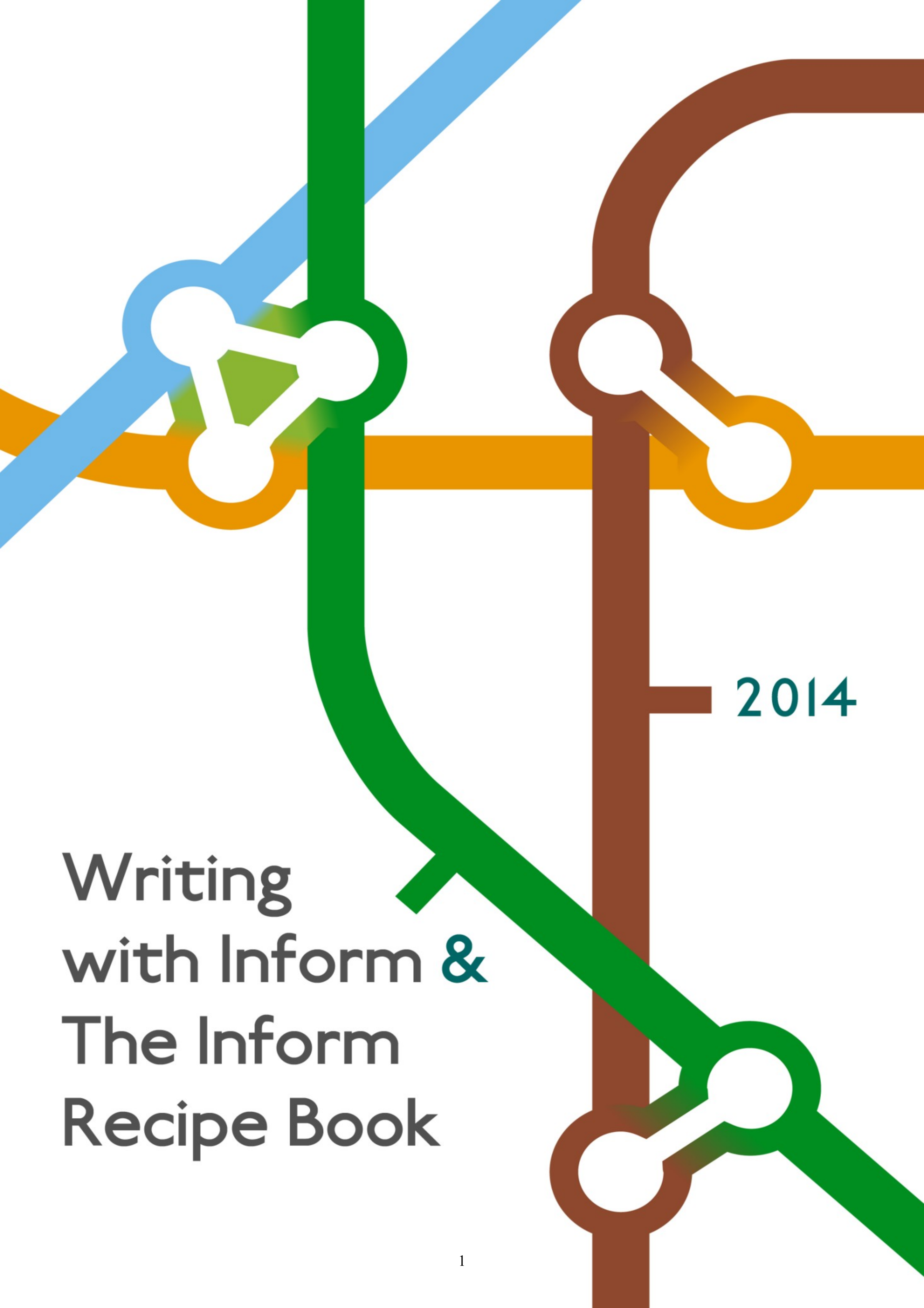


2014

Writing with Inform & The Inform Recipe Book

Part I. Writing with Inform

➡ Start reading here: §1.1. Preface
⬇ Part II. The Inform Recipe Book
↘ Indexes of the examples and definitions

Chapter 1: Welcome to Inform

↖ Contents of *Writing with Inform*
➡ Chapter 2: The Source Text
↘ Indexes of the examples

§1.1. Preface

Welcome to Inform, a design system for interactive fiction based on natural language.

Interactive fiction is a literary form which involves programming a computer so that it presents a reader with a text which can be explored. Inform aims to make the burden of learning to program such texts as light as possible. It is a tool for writers intrigued by computing, and computer programmers intrigued by writing. Perhaps these are not so very different pursuits, in their rewards and pleasures.

> The sheer joy of making things... the fascination of fashioning complex puzzle-like objects of interlocking moving parts and watching them work in subtle cycles... the delight of working in such a tractable medium. The programmer, like the poet, works only slightly removed from pure thought-stuff. He builds his castles in the air, from air, creating by exertion of the imagination. (Frederick P. Brooks, "The Mythical Man-Month", 1972)

Writing with Inform is one of two interlinked books included with Inform: a concise but complete guide to the system. The other book is **The Inform Recipe Book**, a comprehensive collection of examples, showing its practical use. If you are reading this within the Inform application, you will see that the Writing with Inform pages are on "white paper", while the Recipe Book is on "yellow paper".

These notes are arranged so that the reader can, in principle, write whole works of fiction as early as the end of Chapter 3. Each subsequent chapter then extends the range of techniques available to make livelier and more intriguing situations.

This new release of Inform ("Inform 7", the seventh major version since 1993) is a radical departure from most previous approaches to interactive fiction. In particular, it is very different from Inform 6, which newcomers will not need to know anything about. Inform 6 sits inside Inform 7, and is part of the inner workings, but is not visible from the outside. For information about Inform 6, see **www.inform-fiction.org**.

> Programming is best regarded as the process of creating works of literature, which are meant to be read... so we ought to address them to people, not to machines. (Donald Knuth, "Literate Programming", 1981)

★ *See* **Acknowledgements** *for a chance to try out the cross-referencing links in Writing with Inform - click on the red asterisk or the name of the destination to go there*

⬆ Start of Chapter 1: Welcome to Inform

➡ Onward to §1.2. Acknowledgements

⬇ Example 1: ★ **About the examples** An explanation of the examples in this documentation, and the asterisks attached to them. Click the heading of the example, or the example number, to reveal the text.

---

§1.2. Acknowledgements

Inform 7 is dedicated to Emily Short and Andrew Plotkin, whose shrewd and sceptical suggestions made a contribution which can hardly be overstated. A long email correspondence with Andrew entirely subverted my original thoughts about natural-language IF, as he convinced me that the "new model" of rule-based IF was a truer foundation; while Emily's wry, witty analysis and how-about-this? cheered me at low moments, besides providing the impetus and often the specifics for a lot of the best ideas.

From the outset, I have thought of Inform 7 as no longer being a command-line compiler, but a compiler in combination with a humanising user interface. All credit for the reference implementation under Mac OS X belongs to Andrew Hunter. How simple the metaphor of an interactive book with facing pages may seem, but the coding was an enormous challenge. In

2014 Toby Nelson, the author's brother, put months of time into the project by rewriting and modernising the Mac OS X application: sandboxing it for the Mac App Store, giving it a more contemporary design, and much more.

Though David Kinder's Windows application does indeed visually follow the OS X original, the two programs were coded independently, and the programming task taken up by David was formidable indeed. Philip Chimento's Gnome-based user interface for Linux became officially part of the project in November 2007, when the first easy-to-install packages for Ubuntu and Fedora were offered. Philip's efforts were particularly generous since the early stages of Inform-for-Linux were so tentative: for many months, we weren't sure how to go about the project, and during that time Philip quietly wrote us a solution. Adam Thornton continues to support Inform at the command line on Unix-like systems.

Inform in its widest sense incorporates work by so many people that it's simply impossible to thank all of them, but Erik Temple, Dannii Willis, Ron Newcomb, Eric Eve and Juhana Leinonen all deserve special mention. More than 250 users have filed patient and careful bug reports, keeping us on the straight and narrow. They're contributors, too.

It's perhaps surprising that the Inform application gained its first online component only in 2014, but the Public Library of Extensions, and its discussion forum, make a brave step into the modern age. Mark Musante, our Extensions librarian, Dannii Willis and Justin de Vesine have been a great help in setting this up.

The original development of Inform 7 was a long haul, and I would particularly like to thank Sonja Kesserich, David Cornelson and other volunteers for their early testing of a then-fragile system. The final months before the Public Beta release of Inform 7 were made more enjoyable, as well as more productive, by fruitful discussions leading to a cross-platform standard for bibliographic data and cover art. L. Ross Raszewski, who wrote frighteningly efficient reference software in frighteningly little time; the librarians of the IF-Archive, Andrew Plotkin, David Kinder and Paul Mazaitis; and my fellow authors of IF design systems - Mike Roberts (of the Text Adventure Development System); Kent Tessman (of Hugo); and Campbell Wild (of ADRIFT).

This EPUB edition of the documentation was greatly assisted by excellent advice published by Liza Daly, an old friend of Inform's who also helped construct our website.

⬆ Start of Chapter 1: Welcome to Inform

⬅ Back to §1.1. Preface

➡ Onward to §1.3. The facing pages

§1.3. The facing pages

On most computers, Inform runs in a single main window which is an opened book showing two facing pages. As we shall see it behaves as if these pages are in dialogue with each other: for the most part we write on the left hand page and see responses appear on the right. But all is controllable. The margin between the two pages can be dragged back and forth like the slide on a trombone: each page can be made smaller that the other may grow larger. Moreover, each page can display one of a number of displays relevant to the current project, called "panels", one of them being the Documentation panel which displays a screen-readable copy of this manual. The vertical strip of choices at the right hand margin of each page allows you to choose between panels. (The same panel can be showing on both pages at the same time, if that's useful.)

At the start the only panels available are a blank space in which to write the first lines of a new interactive fiction - the Source panel - and this one, the Documentation. Clicking on the other choices will do nothing.

The exception is the Settings panel, which contains some preference settings for the individual project - not the whole application. This is always available, but it controls settings which can be left alone almost all of the time.

⬆ Start of Chapter 1: Welcome to Inform
⬅ Back to §1.2. Acknowledgements
➡ Onward to §1.4. The Go! button

§1.4. The Go! button

Clicking the Go button translates the text in the Source panel into a computer program which enacts the interactive fiction, and automatically sets it going (in the Story panel, which opens as needed).

If the Source is empty of text, Inform will be unable to create anything: it needs at least one name of a location where the drama can unfold. For reasons of tradition, such locations are normally called "rooms", though people have used them to represent anything from grassy fields to states of mind and other metaphorical places.

```
"Midsummer Day"

The Gazebo is a room.
```

Clicking Go with this text in the Source panel will result in a short delay, after which the Story panel will appear, from which we can explore this newly created world: an interactive fiction called "Midsummer Day". It will not be very exciting, since Inform has only five words to go on, but we can add more detail to the source at any point and then click Go again to try out the changes. (Note that there is no need to "quit" these explorations in the Story panel. When Go is clicked, any story already in progress is discarded in favour of the new version.)

The keyboard shortcut Command-R (on Mac OS X), F5 (on Windows), or Ctrl-R (on Linux GNOME) has the same effect as clicking Go.

⬆ Start of Chapter 1: Welcome to Inform

⬅ Back to §1.3. The facing pages

➡ Onward to §1.5. The Replay button

§1.5. The Replay button

Replay works identically to Go, except that it does something further: once the story is created, it automatically plays through the same commands as were typed into the previous version. For instance: suppose we click Go to bring Midsummer Day into being, and find ourselves playing the story. We type "look" and find that there is not much to see. Going back to the source, we add

```
"A white canvas parasol raised up on stakes driven into the grass."
```

so that the source now reads

```
"Midsummer Day"

The Gazebo is a room. "A white canvas parasol raised up on stakes driven into the grass."
```

Instead of clicking Go, we click Replay, and can sit back and watch what has changed. In this example, it only saves us the trouble of typing "look", but once stories become long and elaborate, Replay is invaluable: and especially when we notice in play that something very minor is wrong - a spelling error, say - and want to fix it immediately, without fuss.

⬆ Start of Chapter 1: Welcome to Inform

⬅ Back to §1.4. The Go! button

➡ Onward to §1.6. The Index and Results panels

§1.6. The Index and Results panels

If, when Go! is clicked, the text in the Source panel is not fully understood, then Inform will generate a report of the problems it found, which will open in the "Errors" panel. (Other information is also available in "Errors", but most of it is used for debugging Inform, and can be ignored.)

On the other hand, if the text was fully understood then another new panel will become available: the "Index". This is a cross-referenced index of the source, or rather, of the interactive fiction which has been generated. The Index is only an optional convenience, but becomes more and more helpful as the fiction grows larger. Its exact format does not matter for now.

The icon ↻ always denotes a reference to a particular line in the Source text, that is, to something written in the source: clicking it opens the Source panel and jumps to that position.

The icon 🔍 indicates that more detailed information can be read further down the text in the same panel: clicking it jumps down to this more detailed report.

Lastly, the icon ❓ hints that there is a relevant page of this manual: clicking this opens the Documentation panel and switches to it.

⬆ Start of Chapter 1: Welcome to Inform

⬅ Back to §1.5. The Replay button

➡ Onward to §1.7. The Skein

§1.7. The Skein

The Replay button demonstrates that Inform must be quietly remembering the commands typed into the last run through the story. In fact it remembers, and automatically organises, *every* previous run.

Inform's approach to testing interactive fiction is to treat it as being like the analysis of other turn-based games, such as chess. It would be prohibitively difficult to work out every possible combination of moves: instead, we analyse those which go somewhere, and look for significant choices. Every Queen's Gambit begins with the same first three moves (1. d4, d5; 2. c4), but then there is a choice, as the next move decides whether we have a Queen's Gambit Accepted (dxc4) or Declined (e6). Books about chess often contain great tables of such openings, which run together for a while but eventually diverge. To learn chess, one must explore all of these variations.

Inform's Skein panel is just such a table, built automatically. If we think of the list of typed commands as a thread, then the skein is (as the name suggests) braided together from all these threads. In the display, time begins at the top, with the **start** knot, and the threads of different play-throughs hang downwards from it.

Double-clicking on a command translates the source afresh and replays the story from **start** down to that command, and then stops. We are then free to continue play by typing commands into the Story panel, of course, and these commands will automatically be recorded in the Skein as a new variation of play, diverging from the previous threads.

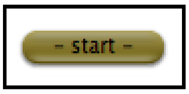
⬆ Start of Chapter 1: Welcome to Inform

⬅ Back to §1.6. The Index and Results panels

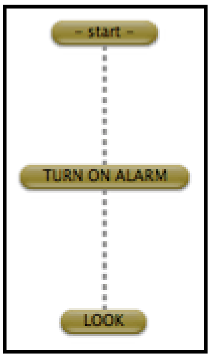
➡ Onward to §1.8. A short Skein tutorial

§1.8. A short Skein tutorial

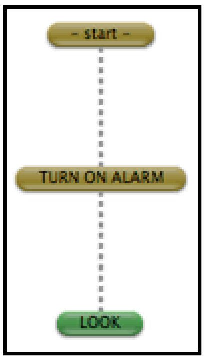
In the following example, we will see how the Skein is woven as different commands are tried. As it happens, the story being played is the example "Witnessed", from Chapter 11, but the details do not matter. When the project has never been played at all, if we switch to the Skein panel (or open it opposite the Story panel) we will only find this:



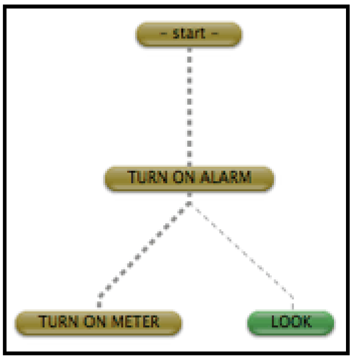
Suppose we click Go for the first time and type two commands in: TURN ON ALARM and then LOOK. Now the Skein shows:



Only one line of play is known to Inform, and it runs downwards in a thread from the special "- start -" knot, which represents the situation before any command has been tried. The useful thing about having past histories recorded like this is that we can revisit them. Suppose we want to go back to the situation after typing only TURN ON ALARM. We could click Go again and type that first command in once more, but now we have an easier method: we simply double-click on the TURN ON ALARM knot. The story restarts by itself, and commands are automatically keyed in to regain the position of play represented by the knot we clicked on - in this example that only keys a single command in, but it might have been hundreds. The Skein now looks like this:



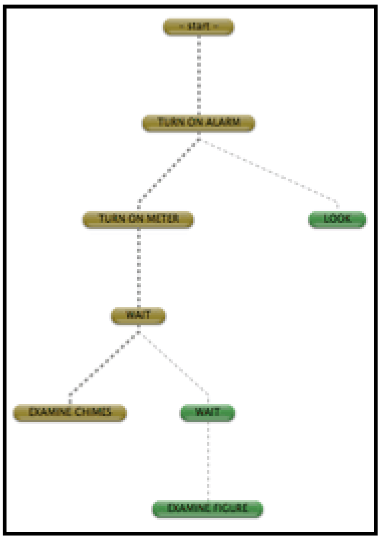
All knots are displayed either as yellow or green. Yellow knot are the ones in the history of the story currently playing. The LOOK knot is green because it hasn't happened in the current story yet - and in fact, it won't happen in the current story, because instead we play TURN ON METER. Now the Skein changes again:



Inform now knows about two ways to play the current project: one consisting of TURN ON ALARM and then TURN ON METER, the other of TURN ON ALARM and then LOOK. Since these only differ after the first turn, Inform displays them as a thread which divides into two after the first turn. Again, LOOK remains green because it hasn't been played in the current story.

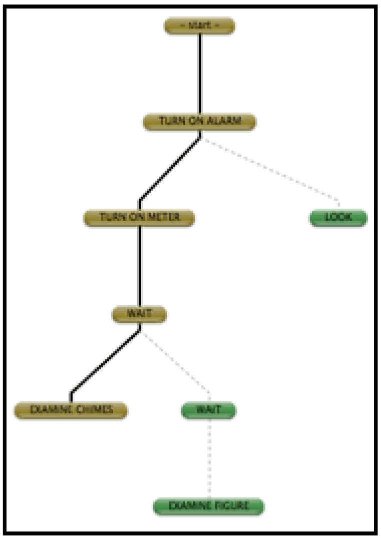
Note also that one of the two possible threads here is drawn more thickly (here it is shown with thick dashes rather than thin). Only one thread is ever drawn thickly -- the one currently being shown in the Transcript panel, which we will come to later on. (That often corresponds to the current line of play, as now, because the Transcript follows what we do unless we choose otherwise.)

After a little more exploration, we reach the following:



At this point we decide that we want to preserve the thread leading to EXAMINE CHIMES - perhaps it's a sequence we are going to want to test often. The Skein can be edited very easily: right-clicking on a knot brings up a contextual menu.

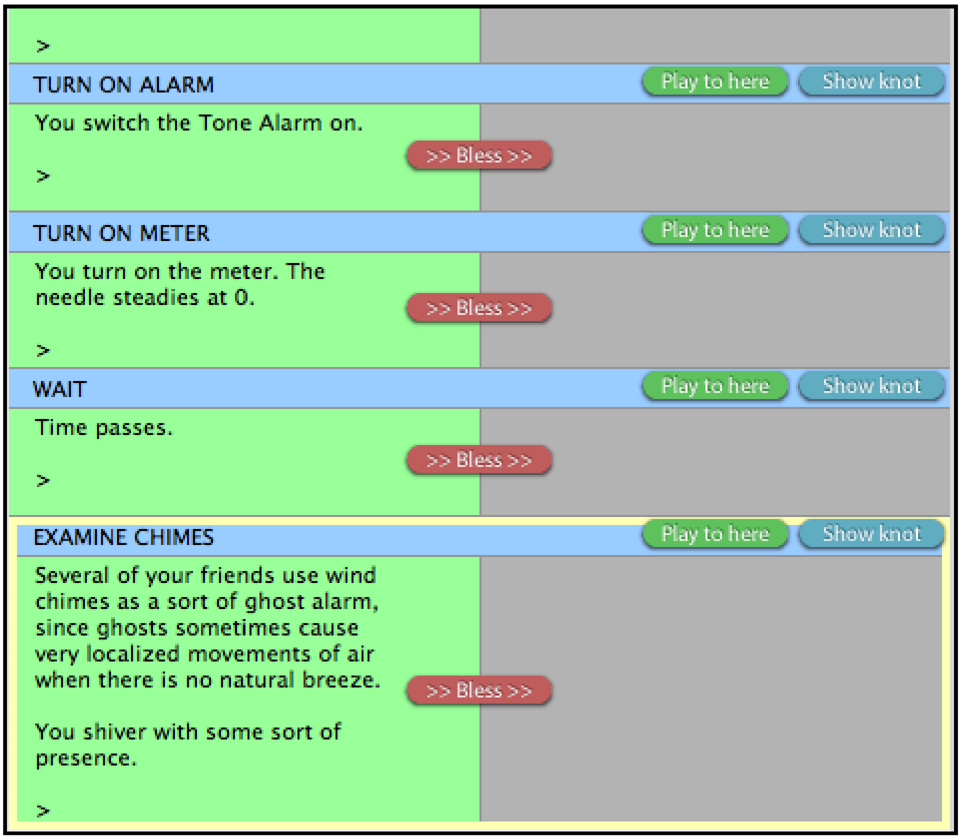
We choose Lock This Thread from the contextual menu, and this makes the thread through to here "locked". That means the knots can't be deleted (unless we unlock them again) - either by our own mistake, or by Inform trimming back no-longer-needed threads of the Skein to keep it manageable in size.



Note that this locked history is now drawn as a solid thread, whereas all the others are unlocked and drawn as dashes.

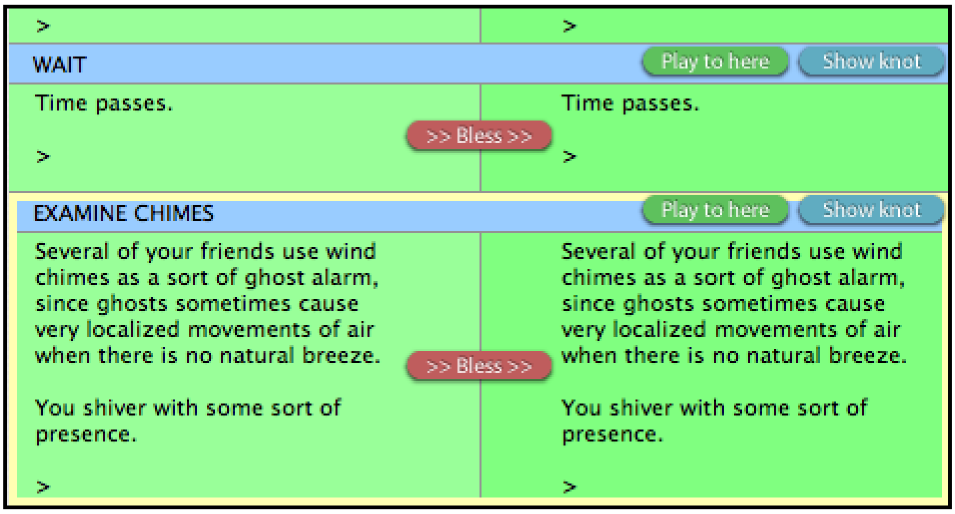
Now we have a securely remembered piece of standard play: it means we can try out the sequence TURN ON ALARM / TURN ON METER / WAIT / EXAMINE CHIMES any time we want to with a double-click on the final knot. This is convenient for testing - but so far it only runs the test: to see whether the test came out well or badly, we have to look through what happened, perhaps by scrolling back in the Story panel to look at the text. And that means that we need to remember what the text should have been like.

In fact, though, Inform can remember for us, using the Transcript panel. This is closely joined to the Skein panel, and it's often convenient to flip between the two. Turning to the Transcript now, we find a two-column view of the story currently being played. The left-hand column shows the text which has been displayed on each turn so far; the right-hand column is empty. The bottom of the Transcript looks like so:



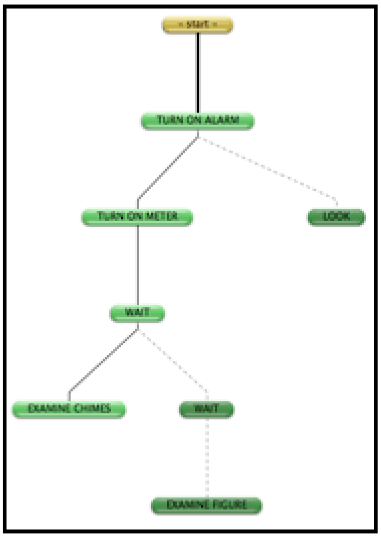
```
>
TURN ON ALARM
You switch the Tone Alarm on.

>
TURN ON METER
You turn on the meter. The
needle steadies at 0.

>
WAIT
Time passes.

>
EXAMINE CHIMES
Several of your friends use wind
chimes as a sort of ghost alarm,
since ghosts sometimes cause
very localized movements of air
when there is no natural breeze.

You shiver with some sort of
presence.

>
```

The empty right-hand column displays the "blessed" transcript - one which the author has approved as being correct. This can be done for each individual knot, using the Bless button joining the columns, but in this case we will bless the whole transcript of this story, using the Bless All button. Now there's text in both columns, and of course the two columns match. (Note that the blessed transcript is in a brighter colour.)



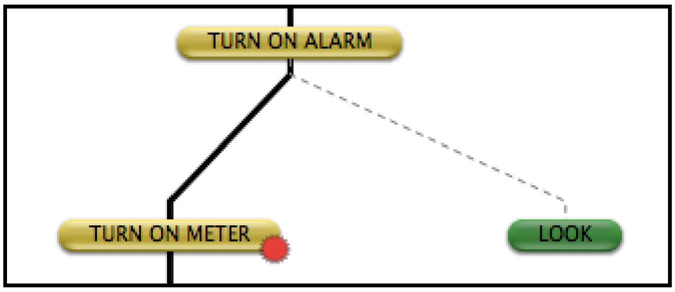
```
>                                    >
WAIT
Time passes.                         Time passes.

>                                    >
EXAMINE CHIMES
Several of your friends use wind     Several of your friends use wind
chimes as a sort of ghost alarm,     chimes as a sort of ghost alarm,
since ghosts sometimes cause         since ghosts sometimes cause
very localized movements of air      very localized movements of air
when there is no natural breeze.     when there is no natural breeze.

You shiver with some sort of         You shiver with some sort of
presence.                            presence.

>                                    >
```

Back in the Skein, we find that the knots which have transcripts have lit up, and are brighter than the others. If we Go, to start a new story, and then look at the Skein:

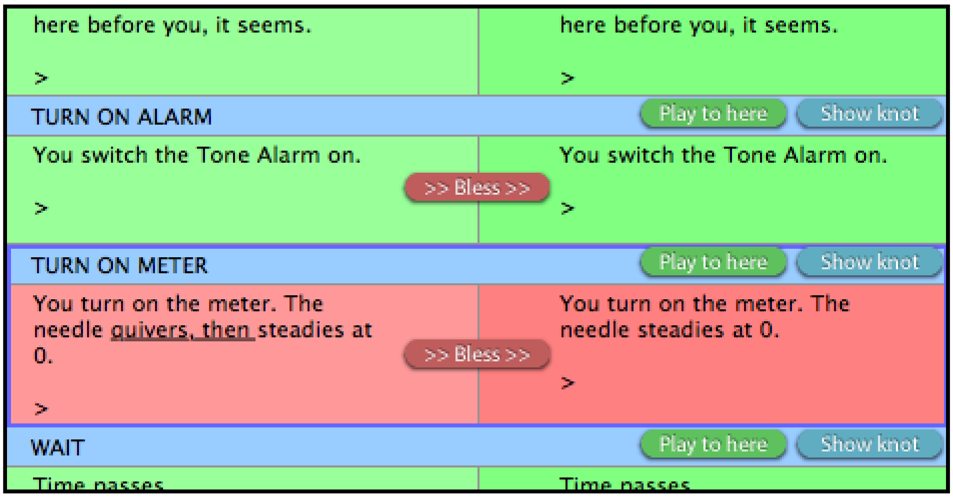


we see that the knots for which we have blessed a transcript are in a brighter green (or a brighter yellow, if they're in the current story being played).

Now suppose we change the source text for the project, so that we make it behave differently. The details don't matter, but suppose we do something which changes the result of the TURN ON METER command, and then run the test again. Now we find:



The red warning badge on the TURN ON METER knot alerts us that the last time this knot was tried (just now, as it happens), the resulting text didn't agree with its blessed transcript. (Red badges can only be seen on bright-coloured knots which have transcripts - for other knots, there's nothing to compare with.) On the other hand, the rest of the yellow current line of play worked out exactly the same as we expected - so no badges. Clicking on the red badge takes us into the Transcript panel at the right place, where the corresponding turn's transcript has also turned red:

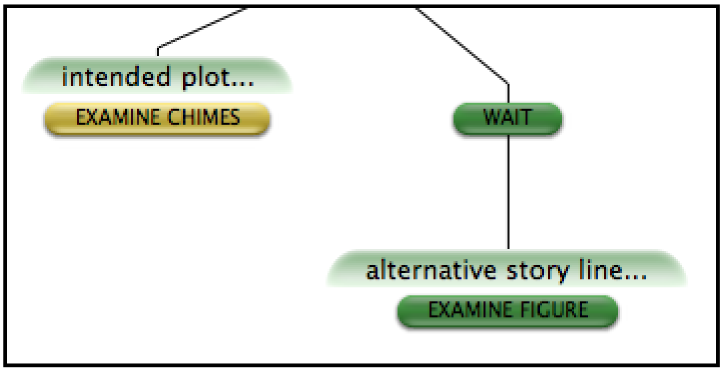


here before you, it seems. >	here before you, it seems. >
TURN ON ALARM	
You switch the Tone Alarm on. >	You switch the Tone Alarm on. >
TURN ON METER	
You turn on the meter. The needle quivers, then steadies at 0. >	You turn on the meter. The needle steadies at 0. >
WAIT	
Time passes	Time passes

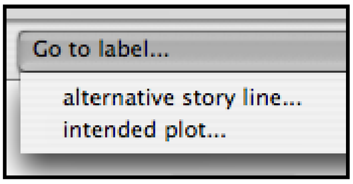
Again, what actually happened is on the left; what should have happened is on the right. The change is shown with underlining - we added the text "quivers, then". If we approve this change, by clicking on the Bless button for the red turn, the amended text will become the correct text to compare against in future runs, and the turn will become green to show that once again all is well. (We can also edit the blessed transcript directly, by double-clicking in the text and typing.) Clicking on the Show knot button takes us back in the skein, at the right place: where we will see that the red warning badge has disappeared.

Some writers of IF like to work backwards from a transcript of the story they want to produce, and for them, the Skein and Transcript combination will be helpful as a running picture of what works so far. Other authors may not use the Skein/Transcript feature at all until right at the end of a project, in testing before publication, when it becomes very important to be able to make small changes in one area without upsetting everything else. Either way, the Skein and Transcript together make a very powerful testing aid.

This tutorial has shown only a short line of play, to keep the pictures small, but for a large project the Skein might run to thousands of knots. It then becomes important to be able quickly to find key knots corresponding to plot developments. To help with that, we can annotate certain knots with any label we choose (by selecting Add Label from the contextual menu):



And this is where the "Labels" gadget at the top of the Skein comes into its own:



since it offers a menu of all the labels in the Skein, and if selected will jump to the one chosen.

The Skein has other abilities too, best explored by experimenting. For instance, we can edit the commands by selecting Edit Knot from the contextual menu, and we can add new knots in the middle of existing lines using the Insert Knot item on that menu. The Play All Blessed option (on the Game menu) is especially powerful: it tests each possible blessed history in turn, trying all of them, and can therefore test very complicated multiple endings and the like in a single click.

Start of Chapter 1: Welcome to Inform

Back to §1.7. The Skein

Onward to §1.9. Summary of the Skein and Transcript

§1.9. Summary of the Skein and Transcript

The Skein records the history of different plays through the current project, and the Transcript records the text of each response, comparing it with a "blessed" or correct version if one is available.

In the Skein each typed command is a "knot". The threads hanging down from the top "-start -" knot are possible histories. Double-click on a knot to play through to there.

Yellow knots are commands played so far in the current story: green knots are possible lines not taken, or not taken yet.

A solid thread is "locked" and protected from deletion (by accident or when Inform trims away loose ends): a dashed thread has no such protection.

A bright knot has a blessed transcript: a darker knot is one which has no blessed transcript. When a bright knot shows a red badge, this means that when last tested its command produced a textual reply which wasn't the same as the blessed transcript. Clicking on the badge shows exactly how.

The thicker thread in the Skein shows the history currently being displayed in the Transcript panel.

Start of Chapter 1: Welcome to Inform

Back to §1.8. A short Skein tutorial

Onward to Chapter 2: The Source Text: §2.1. Creating the world

Examples from Chapter 1: Welcome to Inform

Start of this chapter

Chapter 2: The Source Text

Indexes of the examples

1 ★ **Example** **About the examples** *RB*

An explanation of the examples in this documentation, and the asterisks attached to them. Click the heading of the example, or the example number, to reveal the text.

This is the first of about 400 numbered examples. In a few cases, such as this one, they provide a little background information, but almost all demonstrate Inform source text. The techniques demonstrated tend to be included either because they are frequently asked for, or because they show how to achieve some interesting effect.

The same examples are included in **both** of the books of documentation, but in a different order: in *Writing with Inform*, they appear near the techniques used to make them work; in *The Inform Recipe Book*, they are grouped by the effects they provide. For instance, an example called "Do Pass Go", about the throwing of a pair of dice, appears in the "Randomness" section of *Writing with Inform* and also in the "Dice and Playing Cards" section of *The Inform Recipe Book*. Clicking the italicised WI and RB buttons at the right-hand side of an example's heading switches between its position in each book.

Many computing books quote excerpts from programs, but readers have grown wary of them: they are tiresome to type in, and may only be fragments, or may not ever have been tested. The authors of Inform have tried to avoid this. All but two dozen examples contain entire source texts. A single click on the paste icon ▤ (always placed just left of the double-quoted title) will write the complete source text into the Source panel. All that is then required is to click the Go button, and the example should translate into a working game.

In most cases, typing the single command TEST ME will play through a few moves to show off the effect being demonstrated. (You may find it convenient to create a "scratch" project file for temporary trials like this, clearing all its text and starting again with each new test.)

As part of the testing process which verifies a new build of Inform, each example in turn is extracted from this documentation, translated, played through, and the resulting transcript mechanically checked. So the examples may even work as claimed. But the flesh is weak, and there are bound to be glitches. We would welcome reports, so that future editions can be corrected.

Each example is loosely graded by difficulty: if they were exercises in a textbook, the asterisks would indicate how many marks each question scores. As a general rule:

★ - A simple example, fairly easily guessed.
★★ - A complicated or surprising example.
★★★ - An example needing detailed knowledge of many aspects of the system.
★★★★ - A complete scenario, containing material not necessarily relevant to the topic being demonstrated.

In general, the main text of *Writing with Inform* tries never to assume knowledge of material which has not yet appeared, but the trickier examples almost always need to break this rule.

Chapter 2: The Source Text

↖ Contents of *Writing with Inform*

← Chapter 1: Welcome to Inform

→ Chapter 3: Things

↘ Indexes of the examples

§2.1. Creating the world

Designing an interactive fiction can be divided into two related activities. One is the creation of the world as it appears at the start of play: where and what everything is. The other is to specify the rules of play, which shape how the player interacts with that initially created world. A new Inform project is void and without form, so to speak, with nothing created: but it starts with hundreds of standard rules already in place.

The same division between creating things, and laying down rules, is visible in Inform source text. The creation of the world is done by making unconditional factual statements about it. For example,

```
The wood-slatted crate is in the Gazebo. The crate is a container.
```

Inform calls sentences like these "assertions". The verb is always written in the present tense (thus the crate "is", not "will be"). Further examples are:

```
Mr Jones wears a top hat. The crate contains a croquet mallet.
```

The words "is", "wears" and "contains" are forms of three of the basic verbs built in to Inform. There are only a few built-in assertion verbs, of which the most important are *to be*, *to have*, *to carry*, *to wear*, *to contain* and *to support*. (As we shall see, further assertion verbs can be created if needed.)

The world described by these assertions is the starting condition of the story: what happens when play begins is another matter. If somebody picks up the crate and walks off with it, then it will no longer be in the Gazebo. Mr Jones may remove his hat.

Start of Chapter 2: The Source Text

Back to Chapter 1: Welcome to Inform: §1.9. Summary of the Skein and Transcript

Onward to §2.2. Making rules

§2.2. Making rules

The other kind of sentence tells Inform what should happen in certain circumstances, and reads like an instruction issued to someone:

```
Instead of taking the crate, say "It's far too heavy to lift."
```

This is a "rule", and it changes the crate's behaviour. The player who tries typing "take crate", "pick up the crate" or similar will be met only with the unhelpful reply "It's far too heavy to lift." The many different kinds of thing which the player can do are called "actions", and are always written as participles: "taking ...", for instance, or "putting ... on ...".

Inform is built on a mass of several hundred rules, some quite complex, and it could even be said that Inform *is* that mass of rules. We never see the complexity behind the scenes because the whole aim is to provide a basic, penny-plain, vanilla flavoured sort of realism. It would be surprising if one could put the crate inside itself, so a rule exists to forbid this. It would be surprising if one could drop something which was already on the ground, and so on. These basic rules of realism are the ones which every new Inform project starts with.

A rule always starts with a situation which it applies to, and then follows with one or more things to do. Here's an example where the situation is "Before taking the crate" - the player is just starting to try to pick the box up - and there's a three-step process to follow, but steps 2 and 3 happen only if step 1 comes out in a particular way:

```
Before taking the crate:
    if the player is wearing the hat:
        now the hat is in the crate;
        say "As you stoop down, your hat falls into the crate."
```

The steps to follow here are called "phrases". Inform knows about 400 built-in phrases, but most of them are needed only occasionally. These three are used over and over again:

> **if** tells Inform to do something only if some "condition" holds, here "the player is wearing the hat";
> **now** tells Inform to change the situation, here so that the hat moves to the crate; and
> **say** tells Inform to say something, that is, to write some text for the player to read.

Every one of the built-in phrases has a definition somewhere in this book. The full definition of "say" will come later, but in the simple form above it writes out the given text for the player to read. (Normally this text is simply shown on screen, not spoken aloud, unless software adapted for partially sighted people is being used.) Phrase definitions are all linked to in the Phrases page of a project's Index.

Start of Chapter 2: The Source Text

Back to §2.1. Creating the world

Onward to §2.3. Punctuation

§2.3. Punctuation

An example rule from the previous section demonstrates one of Inform's conventions about punctuation, and is worth pausing to look at again.

```
Instead of taking the crate, say "It's far too heavy to lift."
```

In English grammar, it's usual to regard a full stop as closing its sentence even when it occurs inside quotation marks, provided there is no indication to the contrary, and this is also the rule used by Inform. Thus:

```
The description is "Shiny." It is valuable.
```

is read as equivalent to

```
The description is "Shiny.". It is valuable.
```

Sentence breaks like this occur only when the final character of the quoted text is a full stop, question mark or exclamation mark (or one of these three followed by a close bracket) and the next word begins, in the source code, with a capital letter. A paragraph break also divides sentences, behaving as if it were a full stop.

Material in square brackets [like so] is "comment", in computing jargon: it is considered as being an aside, a private note by the author, and not read in by Inform. This allows us to make notes to ourselves like so:

```
The China Shop is a room. [Remember to work out what happens if the bull gets in
here!]
```

Inform is all about text, so pieces of text are often quoted in Inform source. This example is typical:

```
The description is "Shiny." It is valuable.
```

Quotations always use double-quotation marks, which aren't part of the text. So the description here is just the five letters and full stop in between the marks:

```
Shiny.
```

That seems straightforward, but there are three conventions to watch out for.

1. Square brackets [and] inside quoted text don't literally mean [and]. They're used to describe what Inform should say, but in a non-literal way. For example,

```
"Your watch reads [time of day]."
```

might produce

```
Your watch reads 9:02 AM.
```

These are called "text substitutions". They're highly flexible, and they can take many different forms.

2. Single quotation marks at the edges of words are printed as double. So:

```
"Simon says, 'It's far too heavy to lift.'"
```

produces

```
Simon says, "It's far too heavy to lift."
```

3. Texts which end with sentence-ending punctuation - full stop, question mark, exclamation mark - are printed with a line break after them. So:

```
say "i don't know how this ends";
say "I know just how this ends!";
```

would come out quite differently - this doesn't affect the appearance of the text, but only the position where the next text will appear. Something to be careful about is that this only applies when the punctuation occurs at the end of a "say", as in these examples. (It doesn't apply when a varying textual value is printed, using some text substitution, because then the pattern of where line breaks occur would be unpredictable - sometimes the value might end in a punctuation mark, sometimes not.)

These three punctuation rules for texts feel very natural with practice, and Inform users sometimes don't realise the third rule is even there, because it just seems the right thing to happen. But occasionally the rules get in the way of what we want to do. (For instance, how do we get a literal [or]? What if we want a single quote mark where Inform thinks we want a double, or vice versa?) So we'll come back to these rules in more detail in the chapter on Text.

Inform also reads other punctuation marks. Colon ":" and semicolon ";" turned up in the previous section, in the writing of rules. It also has the more exotic "|" (not a capital I, a vertical stroke) for paragraph breaks outside of quoted text, but people hardly ever need this.

As these examples begin to show, Inform source imitates the conventions of printed books and newspapers whenever there is a question of how to write something not easily fitting into words. The first example of this is how Inform handles headings, but to see why these are so useful we first look at Problems.

★ *See* **How Inform reads quoted text** *for a fuller exploration of the punctuation rules for text*

Start of Chapter 2: The Source Text

Back to §2.2. Making rules

Onward to §2.4. Problems

§2.4. Problems

The language used in the source reads as if it were English aimed at a human reader (and this is intentional: the designer, after all, is a human reader and needs to be able to understand his or her own source), but in reality Inform can only understand a very modest range of sentences and will complain if its limits are passed. Subtler problems arise if the source contains contradictions. For instance, the following "Problem" might be produced:

> **Problem.** You wrote 'A starting pistol is in the cup', but in another sentence 'A Panama hat is on the cup': the trophy cup cannot both contain things and support things, which is what you're implying here. If you need both, the easiest way is to make it either a supporter with a container attached or vice versa. For instance: 'A desk is here. On the desk is a newspaper. An openable container called the drawer is part of the desk. In the drawer is a stapler.'

This is a rather discursive error message, and if a similar problem were to occur in the same run through, it would be curtailed to:

> **Problem.** You wrote 'A firing pistol is in the box', but in another sentence 'A fedora hat is on the box': again, the croquet box cannot both contain things and support things.

Start of Chapter 2: The Source Text

Back to §2.3. Punctuation

Onward to §2.5. Headings

§2.5. Headings

Once the source grows beyond 1000 words or so, it can all too easily become disorganised, and by the time it reaches the size of a novella it can be difficult to find things. (Though nearly all editors provide a Find function, and the keyboard short-cuts Command-F (Mac OS X) or Ctrl-F (Windows) do the trick in the Inform application.)

Inform provides for us to organise the source code in just the way that a printed book would be organised: with headings and subheadings. Firstly, we can put the title at the top. If the first paragraph consists only of a single quoted piece of text, then that's the title; and an author can also be given, as follows:

```
"Spellbreaker" by Dave Lebling
```

We will later see that more bibliographic information can also be placed here, in the same way that the imprint page of a novel comes before the text gets going. The author's name can normally be given without quotation marks, so long as it contains no punctuation. For instance:

> "Three Men in a Boat" by "Jerome K. Jerome"

needs quotes as otherwise the full stop after the K will be mistaken for the end of a sentence.

A sentence which is the only one in its paragraph and which begins with any of the words "volume", "book", "part", "chapter" or "section" is considered to be a heading or a sub-heading. It must not contain a typed line break, and in order to stand alone in its paragraph there should be a skipped line both before and after it. For instance:

> Section 2 - Flamsteed's Balloon

Headings can be written in any format, provided they start with one of the five indicator words, and they are hierarchical: a "Part ..." heading is considered more significant than a "Chapter ..." heading but not so significant as a "Book ..." heading, and so on. (We do not need to use all five kinds of heading.)

⬆ Start of Chapter 2: The Source Text

⬅ Back to §2.4. Problems

➡ Onward to §2.6. Why using headings is a good idea

§2.6. Why using headings is a good idea

Reports of problems, as we have seen, often quote back the source to justify themselves. Rather than quoting line numbers ("Midsummer Day, line 2017" or something similar) Inform uses the ↺ icon. The down side of this is that a glance at the list of problems might give little hint of whereabouts in the source the difficulties lie. Inform therefore makes use of headings to give a general indication:

> In Part the First, Chapter 1 - Attic Area:
>
> **Problem.** You wrote 'South of the Attic is the Winery' ↺, but in another sentence 'South of the Attic is the Old Furniture' ↺: this looks like a contradiction, which might be because I have misunderstood what was meant to be the subject of one or both of those sentences.
>
> In Chapter 2 - Deeper In:
>
> **Problem.** You wrote 'The Disused Observatory is south of the Dark Room' ↺, but in another sentence 'South of the Dark Room is the Cupboard' ↺: again, this looks like a contradiction.

Secondly, headings are used in the Contents page of the Index, and they allow rapid navigation through the source, by jumping to any heading or subheading with a single click.

Finally, headings are used when working out what a name refers to. Suppose the source contains both a "four-poster bed" and also a "camp bed", and we write something like "The pillow is on the bed." Inform decides which bed is meant by giving priority to whichever is defined in the current section (so far), or failing that the current chapter, or current part, or current book, or finally the current volume. This allows us to write, for instance,

```
The four-poster bed is in the Boudoir. The pillow is on the bed.
```

and not have the pillow mysteriously turn up on the camp bed, which hasn't been mentioned since way back in Chapter 2.

↑ Start of Chapter 2: The Source Text
← Back to §2.5. Headings
→ Onward to §2.7. The SHOWME command

§2.7. The SHOWME command

Problem messages are generated when the source text does not make sense to Inform. Even if it does make sense, though, there is no guarantee that it does what the author intends, and the only way to find out is to test the result by playing through it (or asking others to). For the most part one plays as if one were the eventual reader of the work, but sometimes it is highly convenient to have the god-like powers which are an author's prerogative. These are provided by the testing commands, which are present at every stage until the final release version (generated by the Release button). They will be introduced in this manual as they become relevant: here is the first.

The testing command SHOWME prints out a brief summary about a room or thing, and any contents or parts it may have. Typing SHOWME on its own shows the current room, but any item or room in the story, however distant, can be named instead. For instance:

```
>showme
Boudoir - room
    four-poster bed - supporter
    yourself - person
    pillow

>showme diamonds
diamonds - thing
location: in the strongbox on the dresser in the Drawing Room
unlit; inedible; opaque; portable; singular-named; improper-named
description: The diamonds glitter dangerously.
printed name: diamonds
```

Much of this can be seen, and seen more easily, in the World tab of the Index panel: but that only shows the initial state of play, whereas the SHOWME command reveals the situation in mid-story. ("Room", "supporter" and so on are kinds, of which more in Chapter 3.)

★ *See* **High-level debugging commands** for more convenient testing commands like this one

Start of Chapter 2: The Source Text

Back to §2.6. Why using headings is a good idea

Onward to §2.8. The TEST command

§2.8. The TEST command

The only way to thoroughly test a work of IF is to run a complete solution through it, and carefully check the resulting transcript of dialogue. The Skein and Transcript tools of the Inform application are provided for exactly this purpose.

All the same, most works of interactive fiction contain occasional vignettes, either in terms of short scenes of narrative, or in the behaviour of particular things or rooms, which we would like to test without the fuss of using the full story-level Skein tool. The examples in the documentation are like this: in almost every example, typing TEST ME puts the story through its paces.

Solutions or sequences for testing ("scripts") can be defined with sentences like so:

```
Test balloon with "get balloon / blow balloon / drop balloon".
```

This has no effect on the design itself, but ensures that when the story is played, typing "test balloon" will run through the given three commands in sequence, as if we had typed "get balloon" and then "blow balloon" and then "drop balloon".

The name for the test (balloon in this example) has to be a single word. Typing just "test" at the story prompt gives a list of all the test scripts known to the story. Test scripts can make use of each other, for instance:

```
Test all with "test balloon / test door".
```

One convenient way to keep track of the solution for a work being written is to include a test script at the end of each section, and to place a master test script (like "test all") at the top of the source. But different designers will prefer different approaches, and this testing system is no more than an optional convenience.

Many tests will only be sensible in given places, which may be hard to reach from the initial position; or with the aid of given things, which may be difficult to obtain. We are therefore allowed to add stipulations to test scripts:

```
Test balloon with "get balloon / blow balloon / drop balloon" holding the balloon.
```

The "... holding the balloon" means that the balloon will be transferred to the player's ownership immediately before the test script is run, unless it is already held. Similarly:

```
Test jam with "get jam / taste jam / eat jam" in the Kitchen.
```

Or we might want to say both:

```
Test jam with "get jam / taste jam / eat jam" in the Kitchen holding the jam.
```

(Single quotation marks in test scripts are interpreted the same way in test scripts as they are in other text: that is, they are sometimes read as double-quotes unless they appear to be present as apostrophes. The notation ['] forces a single quotation mark if necessary. Similarly, [/] forces a literal forward slash, and prevents the / from being read as dividing up two commands.)

Sometimes when testing it's convenient to get hold of something not easily available at the moment. The testing command "PURLOIN" does this:

```
The jewelled Turkish clockwork hat is in the sealed glass box.

> PURLOIN HAT
[Purloined.]
```

This can also make test scripts shorter, but of course it's important to make sure that people without PURLOIN powers can still play through.

⬆ Start of Chapter 2: The Source Text
⬅ Back to §2.7. The SHOWME command
➡ Onward to §2.9. Material not for release

§2.9. Material not for release

Special testing commands, like "TEST" and "SHOWME", are automatically excluded from the story if it is exported from the Inform application using the Release button. We sometimes want to write our own for-testing-purposes-only code, though, and for this purpose we are allowed to designate whole headings as being "not for release":

```
Section 10 - Open sesame - Not for release

Universal opening is an action applying to nothing.
Understand "open sesame" as universal opening.
Carry out universal opening: now all doors are open.
Report universal opening: say "Open Sesame!"
```

Clearly we do not wish the final reader to be able to type "OPEN SESAME", so this whole heading will be disregarded in the Release version, as will any heading whose name includes "not for release".

Note that if a chapter, say, is marked as "not for release", then its subheadings (mere sections) will also not be for release. If in doubt, check the "Contents" index: if any section is "not for release" then so are all of its subheadings.

The reverse effect is produced by:

```
Section 10 - Open sesame - For release only
```

That is, it marks material included only in a Release version.

⬆ Start of Chapter 2: The Source Text

⬅ Back to §2.8. The TEST command

➡ Onward to §2.10. Installing extensions

§2.10. Installing extensions

The original Inform of 1993 provided no special facilities for "extensions" - in effect, additional packets of rules providing extra features - but the creation and circulation of these extensions soon became a flourishing part of Inform culture. Today's Inform actively promotes sharing of such extensions, both to bring writers together and to support good practice. For the user of an extension, the advantage is clear: why go to great trouble to (say) work out how to make doors open automatically as needed, when somebody else has already perfected this? For the writer of an extension, there is the satisfaction of producing a good solution to a ticklish problem, and contributing to the public good.

Newcomers will probably not need extensions for quite some while, but there is nothing difficult about using them, so a few brief notes are worth giving here. (The final chapter of the documentation covers the writing of new extensions.)

Extensions are identified by name (say "Following People") and also by author (say "Mary Brown"). They need to be installed before they can be used, which means downloading them from the Internet. By far the easiest way to do this is to use the Public Library feature of Inform: then the application can do everything, letting us either choose individual extensions or download them en masse. But it's also possible to install extensions by hand.

When using Inform on Mac OS X, use the File menu item **Show Extensions Folder** to open the relevant folder in the Finder. Each author has a subfolder of this folder, and his or her extensions live inside it.

When using Inform on Windows, this means storing them in the folder

```
My Documents\Inform\Extensions
```

Each author has a subfolder of this folder, and his or her extensions live inside it. Our example extension should therefore be placed as:

```
My Documents\Inform\Extensions\Mary Brown\Following People.i7x
```

When using Inform on Linux, this means storing them in the folder

```
~/Inform/Extensions/
```

where "~" signifies your home folder. (This will have been created for you the first time you ran i7.) Each author has a subfolder of this folder, and his or her extensions live inside it. Our example extension should therefore be placed as:

```
~/Inform/Extensions/Mary Brown/Following People.i7x
```

When using Inform on Linux, this means storing them in the folder

```
~/Inform/Extensions/
```

where "~" signifies your home folder. (This will have been created for you the first time you ran i7.) Each author has a subfolder of this folder, and his or her extensions live inside it. Our example extension should therefore be placed as:

```
~/Inform/Extensions/Mary Brown/Following People.i7x
```

In fact, though, Inform can automatically install extensions for us: we need only select the "Install Extension..." item on the File menu.

The actual extension file should always be named with a ".i7x" suffix, meaning "I7 extension" - for instance, "Following People.i7x".

To provide an example, Emily Short's useful extension "Locksmith" is one of a small number of extensions which come ready-installed as part of the basic Inform package, and need not be downloaded and installed.

Each time that Inform translates any source text, it performs a quick check of the extensions available, and updates its own internal records. A directory of the extensions currently installed can be found by clicking on "Installed Extensions" from the Extensions panel. This is also worth visiting in order to browse the Public Library, a selection of extensions contributed by Inform users.

⬆ Start of Chapter 2: The Source Text

⬅ Back to §2.9. Material not for release

➡ Onward to §2.11. Including extensions

§2.11. Including extensions

We talk about "including" such an extension into a work of IF because the process merges rules and behaviours from the extension with those we have described ourselves. It's not uncommon for contributions by five or six different people to be pooled together this way.

Including an extension is only a matter of writing a single sentence in the source. For instance:

```
Include Locksmith by Emily Short.
```

Note that it is compulsory to name both extension and author.

Many extensions come with their own documentation. Again, follow the "Installed Extensions" link to see what's available from them.

⬆ Start of Chapter 2: The Source Text

⬅ Back to §2.10. Installing extensions

➡ Onward to §2.12. Use options

§2.12. Use options

One more preliminary. Inform has a small number of optional settings which affect the result of translating the source. The sentence:

```
Use American dialect.
```

makes the resulting work of IF use American spellings (except where the designer spells otherwise) and the American convention for spelling out numbers (thus, "one hundred seventeen" not "one hundred and seventeen"). Similarly:

```
Use the serial comma.
```

uses a comma when printing lists: thus "Julian, Dick, George, and Anne" rather than "Julian, Dick, George and Anne". A more profound change is made by

```
Use scoring.
```

which introduces the concept of a numerical score - something which modern authors of interactive fiction often feel is inappropriate, which is why Inform only provides it on request. Two alternative options:

```
Use full-length room descriptions.
Use abbreviated room descriptions.
```

change the normal way room descriptions are shown: normally they are given in full, but in abbreviated mode, they're never given. (The latter is a bad idea in any publicly released story, but is provided for completeness and in case it may help testing.) Alternatively, we can set the traditional Infocom-style of room description to any of VERBOSE, BRIEF and SUPERBRIEF:

```
Use VERBOSE room descriptions.
Use BRIEF room descriptions.
Use SUPERBRIEF room descriptions.
```

The default is now VERBOSE, but until 2010 it was BRIEF.

Next we have:

```
Use undo prevention.
```

which disables the UNDO verb, both in play and after death, for the benefit of stories which are heavily randomised and where we do not want players to keep on UNDOing until they get a random outcome which is to their taste. (Many players consider UNDO to be their birthright, and that any work using this option is an abomination: indeed, it has even been suggested that this section of the Inform documentation be censored. To use the option is to court controversy if not outright hostility.)

We can combine any number of options in a single "Use" sentence, so for example:

```
Use American dialect and the serial comma.
```

brings about both of these changes.

Start of Chapter 2: The Source Text

Back to §2.11. Including extensions

Onward to §2.13. Administering classroom use

§2.13. Administering classroom use

Inform is increasingly used in education, where teachers sometimes need to install it on a whole room of computers at once, and want to monitor their students' progress. There is no special "classroom" version of Inform, but a couple of small administration features in the standard Inform - usually never needed - might be helpful to teachers.

When Inform starts up, it now looks for a file called Options.txt inside the user's home folder for Inform. (On Mac OS X, this is "~/Library/Inform"; on Windows, "My Documents\Inform", and so on.) If the file is present, then the text in it is added to the source text of everything Inform translates.

This must be used only to set use options, specify test commands, and give release instructions. For example, the following is a valid "Options.txt":

```
Use American dialect.
Test fish with "fish/fish with pole/angle".
Release along with source text.
```

The idea is that this file can be used for setting up a standard configuration on multiple machines in a classroom setting. Here the instructor can make sure the Release button will do what she would like, and can arrange for each student's copy of Inform to respond to given Test commands: for instance, if the class has an assignment to create a simulation of a camera, the instructor could set up "Options.txt" so that TEST CAMERA would run through some commands the camera ought to respond to.

A new use option, "Use telemetry recordings.", causes Inform to copy its outcome and problem messages to files in its home folder (see above) as they occur. These files are dated, so that for instance

```
Telemetry 2009-03-25.txt
```

contains all of the recorded activity on 25 March 2009. Telemetry only records the contents of the "Problems" panel - notes of success or failure, and problem messages - and nothing is transmitted via any network, so it isn't really surveillance. The user can deliberately add a note to the current telemetry file by writing something like this in source text:

```
* "I don't get it! What's a kind? Why can't the lamp be lighted?"
```

(This is a way to make a note for the benefit of someone who will read the telemetry file - for instance, to comment on a problem message that has just appeared. Note the double-quotes. Otherwise, it's meant to look like the standard way that beta-testers mark up IF transcripts.)

These two features have been added in response to requests from education users. Let's suppose that Mr Lebling, who teaches 5th grade in Minnesota, wants to set things up just right for his class. He installs Inform on the ten computers they will use, and also copies an Options.txt file from his memory stick onto each one. The Options.txt file reads:

```
Use serial comma.
Use American dialect.
Use telemetry recordings.
```

Now Mr Lebling's class won't be confronted with English spellings, and so on. And most of the kids are happy, but Mr Lebling gets the feeling that young Marc wasn't really paying attention, so after class he checks that day's Telemetry file for that computer to see what Marc was up to, and whether he was stuck on something.

Start of Chapter 2: The Source Text

Back to §2.12. Use options

Onward to §2.14. Limits and the Settings panel

§2.14. Limits and the Settings panel

No computer has unlimited capacity, and a large, complex project may eventually bump its head against the ceiling.

Inform is a system for translating textual descriptions of interactive fiction into "story files". No single format of story file is standard to the IF community. The formats developed over the history of IF differ in three key respects:

```
- the range of computers or devices capable of playing them;
- how large they are, that is, how much play they can express;
- what extra-textual effects they can bring off.
```

Inform can write to two different formats. Neither of these is proprietary, and neither was created by the authors of Inform: each format is a community property, defined by published standards documents. An individual Inform project can make its own choice of story file format, using that project's Settings panel.

Newly created projects are set up with the Glulx format. This has largely taken over from an earlier format called the Z-machine, but Inform can still generate a version 8 Z-machine file (a so-called "z8") if required. The Z-machine is of historic importance, and may continue to be useful for certain tasks where Glulx support is not yet available, but most users will want to keep the Glulx format set all of the time.

Internally, the Inform application uses a tool called Inform 6 (which was once the entire Inform system) to manufacture the story file. There are therefore two ways that large projects can run out of space:

> (a) By exceeding some maximum in Inform 6, or
> (b) By exceeding some fundamental limitation of the current story file format.

In both cases, the Inform application will display a Problems page explaining that the Inform 6 tool has failed to work as intended, and refer us to the "console output" - the text produced by Inform 6 - which is normally ignored, but can be found on the Console tab of the Results panel.

In case (a), Inform 6 will say that a memory setting has been exceeded: it will say what this setting is called (for instance "MAX_ZCODE_SIZE") and what its current value is (for instance 50000). We can then avoid the problem by adding the following use option into the source text:

```
Use MAX_ZCODE_SIZE of 60000.
```

And similarly for every other Inform 6 memory setting. (If the source tries to specify the same setting more than once - which is quite possible if extensions are included, with rival ideas - then the highest value is used.)

Case (b) is only likely to happen with the Z-machine format, since Glulx has a huge capacity; so the cure here is to switch to Glulx in the Settings. But if that's not possible for some reason - say, if we want a story file playable on a tiny handheld computer unable to manage Glulx - we still have a few options. Unless the story is very large (in which case there is little we can do), the "z8" format is most likely to be exhausted for lack of what is called "readable memory", with a message like so:

```
This program has overflowed the maximum readable-memory size of the Z-machine
format. See the memory map below: the start of the area marked "above readable
memory" must be brought down to $10000 or less.
```

followed by a tabulation of how the Z-machine's storage has been used, a large but not very useful diagram. The first time one runs into the problem on a large project, it can be postponed, by adding the following to the source:

```
Use memory economy.
```

(Economy cuts down the verbosity of some of the testing commands, but otherwise subtracts no performance.) Writing this into the source is the equivalent of a diver switching to an emergency oxygen tank: it gives us a generous safety margin, but also tells us that now is the time to wrap things up.

If we hit the problem again, genuine cuts must be made. As a general rule, the most memory-expensive ingredients of an Inform design are various-to-various relations between large kinds such as "thing" or, if there are many rooms, "room". Other than that, if a kind has been festooned with new properties and we have created dozens of items of that kind, then we can get a fairly large saving simply by doing without one of those properties; and so on.

The ultimate memory-saving device, of course, is the one used by book publishers when there are too many pages to bind: to cut the design into two stories, Part I and Part II.

Start of Chapter 2: The Source Text
Back to §2.13. Administering classroom use
Onward to §2.15. What to do about a bug

§2.15. What to do about a bug

In its present guise, Inform is a young piece of software, and bugs are to be expected from time to time. The most obvious bugs are the ones which Inform catches itself, when it confesses that it has halted in failure, or translated the source text into a program which cannot be compiled further. But sometimes it will also happen that Inform will issue a misleading Problem message, or appear to work normally but to produce a story which does not do what it should have done.

It is very helpful for users to report faults, so that the program can be improved for everyone else. To report a fault, please first check with the Inform home page to make sure that the version of Inform you have used to detect the fault is the latest version available. You can find the latest versions at

http://inform7.com/download/

If the bug is still present in the latest version, please report the bug using Inform's bug tracking database, found at

http://inform7.com/mantis/

We can search existing bug reports using the search box at

http://inform7.com/mantis/view_all_bug_page.php

It may be that someone else has already identified the bug and even that a workaround for users is suggested. If not, please make an account at the bug tracking system and submit the requested information to help Inform's maintainers track and fix the fault.

Start of Chapter 2: The Source Text
Back to §2.14. Limits and the Settings panel
Onward to §2.16. Does Inform really understand English?

§2.16. Does Inform really understand English?

No. No computer does, and Inform does not even try to read the whole wide range of text: it is a practical tool for a particular purpose, and it deals only with certain forms of sentence useful to that purpose. Inform source text may look like "natural language", the language we find natural among ourselves, but in the end it is a computer programming language. Many things which seem reasonable to the human reader are not understood by Inform. For instance, Inform understands

```
something which is carried by the player
```

but not (at present, anyway)

```
something which the player carries
```

even though both are perfectly good English. So it is not always safe to assume that Inform will understand any reasonable instruction it is given: when in doubt, we must go back to the manual.

More philosophically, to "understand" involves contextual knowledge. Just because Inform recognises and acts on a sentence, does it really understand what we meant? It will turn out that Inform is both good and bad at this. For instance, from

```
Mr Darcy wears a top hat.
```

Inform will correctly deduce that Darcy is a person, because inanimate objects do not ordinarily wear clothes, and that the top hat is clothing. But it will not automatically know that Darcy is a man rather than a woman because it does not know the social convention implied by "Mr". Moreover, if instead we had written

```
Mr Darcy carries a top hat.
```

then Inform would not guess that the top hat is clothing. This is because it does not have the vast vocabulary and experience of a human reader: it is probably discovering the word "hat" for the first time.

Finally, it is best to avoid ambiguities rather than rely on Inform to know which meaning is patently absurd. For instance, in

```
Heatwave bone breaks clog hospital.
```

(a headline once printed by the *Oxford Mail* newspaper) a human reader quickly realises that there is no clog hospital being broken. But if Inform had been taught the verbs *to break* and *to clog* then that is exactly the conclusion it would have drawn. Or an example which genuinely arose in beta-testing:

```
The life support unit fits the egg.
```

in which Inform construed the verb as *support* and not *fits*, and then created items called "the life" (plural) and "unit fits the egg".

That disclaimer completes the groundwork, and we are ready to begin on simulating a world to explore.

Start of Chapter 2: The Source Text

Back to §2.15. What to do about a bug

Onward to Chapter 3: Things: §3.1. Descriptions

Chapter 3: Things

Contents of *Writing with Inform*
Chapter 2: The Source Text
Chapter 4: Kinds
Indexes of the examples

§3.1. Descriptions

At its simplest, the interactive fiction will be simulating a physical world to explore. The forerunner of today's IF is generally agreed to be a computer simulation by Will Crowther of the exploration of a cave system in the Mammoth and Flint Ridge chain of caves in Kentucky, a part of which might be described in Inform thus:

```
"Cave Entrance"

The Cobble Crawl is a room. "You are crawling over cobbles in a low passage. There is a dim light at the east end of the passage."

A wicker cage is here. "There is a small wicker cage discarded nearby."

The Debris Room is west of the Crawl. "You are in a debris room filled with stuff washed in from the surface. A low wide passage with cobbles becomes plugged with mud and debris here, but an awkward canyon leads upward and west. A note on the wall says, 'Magic word XYZZY'."

The black rod is here. "A three foot black rod with a rusty star on one end lies nearby."

Above the Debris Room is the Sloping E/W Canyon. West of the Canyon is the Orange River Chamber.
```

Here we sketch in four of Crowther's locations, and two objects: just enough to be able to walk around the caves and pick up the rod and the cage. The text in quotation marks will appear verbatim as paragraphs shown to the player as the caves are explored. The first paragraph, as we have seen, is the title of the work. The other quotations describe the places and objects introduced.

If we play this story, we find that we can type TAKE CAGE or TAKE WICKER CAGE, for instance, but not TAKE SMALL CAGE. Inform saw that we called this "a wicker cage" when it first appeared in the source text, and assumed that the player would call it that, too. (Whereas it didn't look inside the descriptive text to allow for TAKE SMALL CAGE or TAKE DISCARDED CAGE or TAKE NEARBY CAGE.) A small limitation here is that probably only the first 9 letters of each word are read from the player's command. This is plenty for handling the wicker cage and the black rod, but it might be embarrassing at a meeting of the Justice League to find that KISS SUPERHERO and KISS SUPERHEROINE read as if they are the same command.

So we have already found that Inform has made some assumptions about what we want, and imposed some limitations on how much computational effort to go to when the work of IF is finally played. If Inform guesses what we need wrongly, we need to know more advanced features of the language in order to overcome these problems. (We shall see how to change the way the player's commands are read in the chapter on Understanding.)

This is often how Inform works: make the standard way of doing things as simple as possible to describe, but allow almost any behaviour to be altered by more elaborate source text. As an example of that, the player begins in the Cobble Crawl because it was the first room created in the source text, but we could instead have written text like:

```
The player is in the Cobble Crawl.
```

to override that. This can make the source text easier to follow if the rooms are sometimes being created in a less obvious way. For example, if we write:

```
The silver bars are in the Y2 Rock Room.
The Cobble Crawl is a room. South of the Crawl is Y2.
```

then the first room to be created will actually be the Y2 Rock Room, so that's where the player will be starting unless we say otherwise.

Start of Chapter 3: Things

Back to Chapter 2: The Source Text: §2.16. Does Inform really understand English?

Onward to §3.2. Rooms and the map

Example 2: ★ **Bic** Testing to make sure that all objects have been given descriptions.

Example 3: ★ **Verbosity 1** Making rooms give brief room descriptions when revisited.

Example 4: ★★ **Slightly Wrong** A room whose description changes slightly after our first visit there.

§3.2. Rooms and the map

Rooms are joined together at their edges by "map connections", most of which are pathways in one of the eight cardinal compass directions: north, northeast (written without a hyphen), east, southeast, south, southwest, west, northwest. We also have up and down, suitable for staircases or ladders. In real life, people are seldom conscious of their compass bearing when walking around buildings, but it makes a concise and unconfusing way for the player to say where to go next, so is generally accepted as a convention of the genre.

Two more directions are provided by Inform: "inside" and "outside". These are best used when one location is, say, a meadow and the other is a woodcutter's hut in the middle of it; we might then say

```
Inside from the Meadow is the woodcutter's hut.
```

The "from" is important, as it clarifies that we intend to link two different locations, not to create an item - the hut - in a single location - the meadow.

A problem which sometimes arises when laying out maps is that Inform allows short forms of room names to be used as abbreviations. This is usually a good idea, but has unfortunate results if we write:

```
The Airport Road is west of the Fish Packing Plant. The Airport is west of the Airport Road.
```

...because "Airport" is taken as a reference to "Airport Road", so Inform makes only two locations, one of which supernaturally leads to itself. We can avoid this by writing:

```
The Airport Road is west of the Fish Packing Plant. A room called the Airport is west of the Airport Road.
```

Using "called" is often a good way to specify something whose name might give rise to confusion otherwise. It always makes something new, and it is also neatly concise, because we can establish something's kind and name in the same sentence. As another example, suppose we want to create a room called "South of the Hut", to south of the Hut. We can't do so like this:

```
South of the Hut is a room. South of the Hut is south of the Hut.
```

...because Inform will read that first sentence as placing a (nameless) room to the south of a room called "Hut". Once again "called" can save the day:

```
South of the Hut is a room called South of the Hut.
```

It is best to use "called" in the simplest way possible, and in particular, best not to use "called" twice in the same sentence. Consider:

```
The kitchen cabinet contains a container called a mixing bowl and a portable supporter called a platter.
```

It is unlikely that anyone would want to name something "a mixing bowl and a portable supporter called a platter", but not impossible, and Inform tends not to be a good judge of what is likely.

(If we really want to get rid of this issue once and for all, starting the source text with the use option "Use unabbreviated object names." will do it, but the effect is drastic. This instructs Inform not to recognise names other than in full. For example:

> West of the Kitchen is the Roaring Range. South of the Range is the Pantry.

is ordinarily read by Inform as constructing three rooms (Kitchen, Roaring Range, Pantry); but with this use option set, it makes four (Kitchen, Roaring Range, Range, Pantry), in two disconnected pieces of map. Handle with care.)

⬆ Start of Chapter 3: Things

⬅ Back to §3.1. Descriptions

➡ Onward to §3.3. One-way connections

⬇ Example 5: ★ **Port Royal 1** A partial implementation of Port Royal, Jamaica, set before the earthquake of 1692 demolished large portions of the city.

⬇ Example 6: ★★ **Up and Up** Adding a short message as the player approaches a room, before the room description itself appears.

⬇ Example 7: ★★★ **Starry Void** Creating a booth that can be seen from the outside, opened and closed, and entered as a separate room.

§3.3. One-way connections

Connections are ordinarily two-way, but do not have to be. One of the map connections in the Mammoth Cave simulation was made by the sentence:

> The Debris Room is west of the Crawl.

Besides reading this sentence at face value, Inform also deduced that the Crawl was probably meant to be east of the Debris Room: in other words, that the path between them is a two-way one. When Inform makes guesses like this, it treats them as being less certain than anything explicitly stated in the source. Inform will quietly overturn its assumption if information comes to hand which shows that it was wrong. That might happen in this case if another sentence read:

> The Hidden Alcove is east of the Debris Room.

These two sentences are not contradictory: Inform allows them both, simply accepting that the world is more complicated than it first assumed. There are relatively few situations where Inform has to make educated guesses, but when it does, it tries always to follow Occam's Razor by constructing the simplest model world consistent with the information in the Source text.

We can even explicitly make a route which turns around as it leads between two rooms:

> West of the Garden is south of the Meadow.

If we want to establish a route which cannot be retraced at all, we can specify that a particular direction leads nowhere:

> East of the Debris Room is nowhere.

Finally, note that Inform's assumptions about two-way directions are only applied to simple sentences. When the source text seems to be saying something complicated, Inform takes it as a precise description of what's wanted. So, for example, in:

```
The Attic is above the Parlour.
The Attic is a dark room above the Parlour.
```

Inform makes guesses about the first sentence, and makes a two-way connection; but it accepts the second sentence more precisely, with just a one-way connection.

Start of Chapter 3: Things

Back to §3.2. Rooms and the map

Onward to §3.4. Regions and the index map

Example 8: **Port Royal 2** Another part of Port Royal, with less typical map connections.

Example 9: **The Unbuttoned Elevator Affair** A simple elevator connecting two floors which is operated simply by walking in and out, and has no buttons or fancy doors.

§3.4. Regions and the index map

Rooms represent individual places to which one can go, but we tend to think of the world around us in larger pieces: we think of a house and a garden, rather than each of the single rooms of the house and all corners of its garden. To Inform a collection of rooms is called a "region", and we can create one like so:

```
The Arboretum is east of the Botanical Gardens. Northwest of the Gardens is the
Tropical Greenhouse.

The Public Area is a region. The Arboretum and Gardens are in the Public Area.
```

The real usefulness of creating regions like "Public Area" will only appear later, when we begin defining rules of play which apply in some areas but not others, but in the mean time we can see the effect by turning to the World tab of the Index. In the World Index, Inform draws a map - or at least a stylised attempt at a diagram of the rooms and their connections: this will not always correspond to how we imagine things, but with any luck it should mostly be right.

Rooms are represented by coloured squares, and the colour-coding is done by region. In the above example, the two "Public Area" rooms are coloured green (as it happens); the Greenhouse, since it belongs to no region, is a neutral grey.

Regions can be put inside each other:

```
The University Parks is a region. The Public Area is in the University Parks.
```

but they are not allowed to overlap other than by one being entirely inside the other.

★ *See* **Improving the index map** *for ways to adjust the way the index map is drawn or exported for publication*

⬆ Start of Chapter 3: Things

⬅ Back to §3.3. One-way connections

➡ Onward to §3.5. Kinds

⬇ Example 10: ★ **Port Royal 3** Division of Port Royal into regions.

§3.5. Kinds

The following description runs to only 33 words, but makes a surprisingly intricate design. It not only places things within rooms, but also places them very specifically with respect to each other:

```
"Midsummer Day"

East of the Garden is the Gazebo. Above is the Treehouse. A billiards table is in the Gazebo. On it is a trophy cup. A starting pistol is in the cup.
```

Inform needs to identify the places and objects being described by the nouns here, and to guess what it can about them. For instance, the pistol can be picked up but not walked inside, whereas the Treehouse is the reverse. (This is obvious to someone who knows what these words mean, less obvious to a computer which does not, but the text contains sufficient clues.) Inform does this by sorting the various nouns into different categories, which are called "kinds". For instance:

```
Garden, Gazebo, Treehouse - room
billiards table - supporter
cup - container
starting pistol - thing
East, up (implied by "above") - direction
```

(A container is something which can contain other things, and a supporter similarly.) For instance Inform knows that if one thing is in another, then the second thing is either a room or a container, and if one thing is on another, the second thing is a supporter. This worked nicely for the design above, but:

```
In the Treehouse is a cardboard box.
```

results in the cardboard box being made only a "thing": because nothing has been put inside it, there is no reason for Inform - which does not know what a cardboard box looks like - to guess that it is a "container". So we need to add:

```
The box is a container.
```

It is rather clumsy to have to write two sentences like this, so we would normally write this instead:

```
In the Treehouse is a container called the cardboard box.
```

Start of Chapter 3: Things

Back to §3.4. Regions and the index map

Onward to §3.6. Either/or properties

Example 11: ★ **First Name Basis** Allowing the player to use different synonyms to refer to something.

Example 12: ★ **Midsummer Day** A few sentences laying out a garden together with some things which might be found in it.

§3.6. Either/or properties

Some containers, like bottles, can be opened: others, like buckets, cannot. If they can be opened, then sometimes they will be open, and sometimes closed. These are examples of properties, which can change during play. The following source sets some properties:

```
The cardboard box is a closed container. The glass bottle is a transparent open
container. The box is fixed in place and openable.
```

There are only four different properties referred to here. Closed means not open, and vice versa, so these two adjectives both refer to the same property. (As might be expected, when a container is open, one can see inside and place things within, or take them out.) The glass bottle and the box being containers is a matter of their kinds, which is something fundamental and immutable, so "container" does not count as a property.

A "transparent" container is one which we can see inside even when it is closed, and the opposite is an "opaque" container.

The property of being "fixed in place" ensures that the player cannot pick the item up and walk away with it: this is useful for such things as oak trees or heavy furniture. The opposite condition is to be "portable".

A container which is "openable" can be opened or closed by the player; as might be expected, the opposite is "unopenable".

With a really large cardboard box, we might imagine that the player could get inside: such a container should be declared "enterable".

Start of Chapter 3: Things

Back to §3.5. Kinds

Onward to §3.7. Properties depend on kind

Example 13: ★ **Tamed** Examples of a container and a supporter that can be entered, as well as nested rooms.

§3.7. Properties depend on kind

Properties depend very much on kind. It makes no sense to ask whether a room is transparent or opaque, for instance, so Inform will not allow this either to be specified or queried.

Another way that kind influences properties can be seen from an earlier example:

```
The Gazebo is a room. A billiards table is in the Gazebo. On it is a trophy cup. A starting pistol is in the cup.
```

The cup, the pistol and the table are all allowed to have the "fixed in place" property, but in fact only the table actually has it: the cup and the pistol are created as "portable" instead. This is because Inform knows that most things are portable, but that supporters - such as the table - are usually fixed in place. If this assumption is wrong, we need only add the line:

```
The table is portable.
```

⬆ Start of Chapter 3: Things

⬅ Back to §3.6. Either/or properties

➡ Onward to §3.8. Scenery

⬇ Example 14: ★ **Disenchantment Bay 1** A running example in this chapter, Disenchantment Bay, involves chartering a boat. This is the first step: creating the cabin.

§3.8. Scenery

As we have just seen, making something "fixed in place" will prevent it from being picked up or moved. But it remains substantial enough to be described in its own paragraph of text when the player visits its location. This can be unfortunate if it has also been described already in the body of the main description for that location. For instance, if we wrote:

```
The Orchard is a room. "Within this quadrille of pear trees, a single gnarled old oak remains as a memory of centuries past." The gnarled old oak tree is fixed in place in the Orchard.
```

This would end up describing the oak twice, once in the paragraph about the Orchard, then again in a list of things within it:

```
Orchard
Within this quadrille of pear trees, a single gnarled old oak remains as a memory of centuries past.

You can see a gnarled old oak tree here.
```

We avoid this by making it "scenery" instead of "fixed in place":

```
The gnarled old oak tree is scenery in the Orchard.
```

Any thing can be scenery, and this does not bar it from playing a part in the story: it simply means that it will be immobile and that it will not be described independently of its room. Being immobile, scenery should not be used for portable objects that are meant to be left out of the room description.

If a supporter is scenery, it may still be mentioned in the room description after all, but only as part of a paragraph about other items, such as

```
On the teak table are a candlestick and a copy of the Financial Times.
```

If the player takes the candlestick and the Times, the teak table will disappear from mention. (Scenery containers do not behave in this way: their contents are assumed to be less immediately visible, and will be mentioned only if the player looks inside them.)

↑ Start of Chapter 3: Things

← Back to §3.7. Properties depend on kind

→ Onward to §3.9. Backdrops

↓ Example 15: ★ **Disenchantment Bay 2** Disenchantment Bay: creating some of the objects in the cabin's description.

↓ Example 16: ★ **Replanting** Changing the response when the player tries to take something that is scenery.

§3.9. Backdrops

It is a cardinal rule that nothing can be in more than one place at the same time, but rules were made to be broken, and an exception is allowed for a special kind of thing called a "backdrop". For instance:

```
"Streaming"

The Upper Cave is above the Rock Pool.

The stream is a backdrop. It is in the Upper Cave and the Rock Pool.
```

Backdrops are ordinarily in the background: if the sky needed to be referred to in the course of play, it might be represented by a backdrop, for instance. Here we have a stream of water running through two rooms, though it might be any number. Backdrops are always fixed in place.

Backdrops can be put in regions as well as rooms, and if so, then they are present at every room in the given region (or regions), as well as any specific rooms they may also be put into. For instance:

```
The Outdoors Area is a region. The Moon is a backdrop. The Moon is in the Outdoors Area. The Moon is in the Skylight Room.
```

The special place "everywhere" can be given as the location of a backdrop to make it omnipresent:

```
The sky is a backdrop. The sky is everywhere.
```

Inform assumes that backdrops are also scenery unless told otherwise, so this will not result in messages like "You can also see the sky here." being included in room descriptions. In the case of the stream above, we could artfully mention it in passing in the room descriptions of the Upper Cave and the Rock Pool.

★ *See* **Moving backdrops** for ways to place backdrops in dynamically changing selections of rooms

---

⬆ Start of Chapter 3: Things

⬅ Back to §3.8. Scenery

➡ Onward to §3.10. Properties holding text

⬇ Example 17: ★ **Disenchantment Bay 3** Disenchantment Bay: adding a view of the glacier.

---

§3.10. Properties holding text

The properties we have seen so far have all been either/or: either open or closed, either transparent or opaque, either fixed in place or portable, either openable or not openable. However, some properties can have a much wider range of possibilities. For instance, the "description" of a room is the text revealed when the player first enters it, or types "look". This needs to be textual: Inform would complain if, for instance, we tried to set the description of something to the number 42. We have already seen a concise way to set the description of a room:

```
The Painted Room is north of the Undertomb. "This is the Painted Room, where strange wall drawings leap out of the dark at the gleam of your candle: men with long wings and great eyes, serene and morose."
```

This does the same thing as:

```
The Painted Room is north of the Undertomb. The description of the Painted Room is "This is the Painted Room, where strange wall drawings leap out of the dark at the gleam of your candle: men with long wings and great eyes, serene and morose."
```

Or even:

```
The Painted Room is north of the Undertomb. The description is "This is the Painted Room, where strange wall drawings leap out of the dark at the gleam of your candle: men with long wings and great eyes, serene and morose."
```

Start of Chapter 3: Things

Back to §3.9. Backdrops

Onward to §3.11. Two descriptions of things

§3.11. Two descriptions of things

The player's first sight of something is the text used as its "initial appearance":

> The plain ring is here. "Cast aside, as if worthless, is a plain brass ring."

This text appears as a separate paragraph in the text describing the Painted Room. It will continue to be used until the first time player picks the ring up (if this ever happens), so it normally describes things in their original, undisturbed context. (Inform uses an either/or property called "handled" for this: something is "handled" if it has at some point been held by the player.)

Thus when a piece of text stands alone as a sentence in its own right, then this is either the "description" of the most recently discussed room, or the "initial appearance" of the most recently discussed thing. Either way, it is used verbatim as a paragraph in the text shown to the player visiting the room in question.

But a thing also has an ordinary "description", which is used to give a close-up look at it. This text is ordinarily only revealed to the player when a command like "examine ring" is keyed in:

> The description of the plain ring is "No better than the loops of metal the old women use for fastening curtains."

★ *See* **Creating a scene** for the description of a scene, which is set in the same way

Start of Chapter 3: Things

Back to §3.10. Properties holding text

Onward to §3.12. Doors

Example 18: ★ **Disenchantment Bay 4** Disenchantment Bay: fleshing out the descriptions of things on the boat.

Example 19: ★★ **Laura** Some general advice about creating objects with unusual or awkward names, and a discussion of the use of printed names.

§3.12. Doors

The map of an interactive fiction is the layout of rooms and the entrances and exits which connect them. So far, these map connections have always run from one room to another, like so:

> The Painted Room is north of the Undertomb.

However, we can also interpose doors between rooms, like so:

> The heavy iron grating is east of the Orchard and west of the Undertomb. The grating is a door.

The second sentence is needed since otherwise Inform will take "heavy iron grating" to be the name of a third room, whereas what we want is for the grating to be something physically present in both the Orchard and in the Undertomb, and acting as a conduit between them. To this end it needs to be a "door", a kind we have not so far seen. In the absence of any other instruction, a newly created door will be fixed in place, closed and openable.

The grating really does come in between the two rooms: the grating is what lies immediately east of the Orchard, not the Undertomb room. So if we wrote the following:

> The Undertomb is east of the Orchard. The heavy iron grating is east of the Orchard and west of the Undertomb. The grating is a door.

then Inform would say that this is a contradiction: we said the Undertomb was east of the Orchard, but then we said that the grating was east of the Orchard.

Inform's "door" kind can be used for all manner of conduits, so the word door need not be taken literally. In Ursula K. Le Guin's beguiling novel "The Tombs of Atuan", from which the above rooms are stolen, it is not a grating which interposes, but:

> The red rock stair is east of the Orchard and above the Undertomb. The stair is an open door. The stair is not openable.

In real life, most doors are two-sided, and can be used from either of the rooms which they join, but this is not always convenient for interactive fiction. Here is a one-sided door:

> The blue door is a door. It is south of Notting Hill. Through it is the Flat Landing.

(Note the use of "it" here as an optional abbreviation.) This will make a door visible only on the Notting Hill side; no map connection will be made in the reverse direction, unless we ask for one.

So much for creating and describing individual doors. Once we need to write about doors in general, we are likely to want a way to find out where a given door sits in the map. The following phrases reveal this:

front side of (object) **... room**

This phrase produces the first of the one or two rooms containing a door - first in the order given in the source text. Example: if

> The red rock stair is east of the Orchard and above the Undertomb.

then "front side of the red rock stair" produces the Orchard. For a one-sided door, this produces the only room containing the door.

back side of (object) **... room**

This phrase produces the last of the one or two rooms containing a door - last in the order given in the source text. Example: if

> The red rock stair is east of the Orchard and above the Undertomb.

then "back side of the red rock stair" produces the Undertomb. A one-sided door has no "back side."

More often, we are dealing with a door and want to know what it leads to, but that depends where we're standing:

other side of (door) **from** (room) **... object**

This phrase produces the room on the other side of the door, as seen from the given vantage point, which needs to be one of its sides. Example: if

> The red rock stair is east of the Orchard and above the Undertomb.

then "other side of the red rock stair from the Undertomb" produces the Orchard, and vice versa.

direction of (door) **from** (room) **... object**

This phrase produces the direction in which the door leads, as seen from the given vantage point, which needs to be one of its sides. Example: if

> The red rock stair is east of the Orchard and above the Undertomb.

then "direction of the red rock stair from the Undertomb" produces up.

★ *See* **Adjacent rooms and routes through the map** for more phrases which can look at the current map layout

---

⬆ Start of Chapter 3: Things

⬅ Back to §3.11. Two descriptions of things

➡ Onward to §3.13. Locks and keys

⬇ Example 20: ★ **Disenchantment Bay 5** Disenchantment Bay: adding the door and the deck to our charter boat.

⬇ Example 21: ★★ **Escape** Window that can be climbed through or looked through.

⬇ Example 22: ★★★ **Garibaldi 1** Providing a security readout device by which the player can check on the status of all doors in the game.

§3.13. Locks and keys

It seems unwise for a door in Notting Hill to be unlocked, so:

```
The blue door is lockable and locked. The matching key of the blue door is the brass
Yale key.
```

Since the second sentence here is a little clumsy, we can equivalently say

```
The brass Yale key unlocks the blue door.
```

Yet a third way to say this is:

```
The blue door has matching key the brass Yale key.
```

This introduces three new properties: a door can be locked or unlocked; lockable or not lockable; and it can have a matching key, which must be another thing. The same thing can be the matching key of many different locks: and note that a door can be locked and even lockable without having a matching key at all, in which case the player trying to open it will be permanently out of luck. Doors are ordinarily unlocked, not lockable, and without a matching key.

Containers can also have locks, in exactly the same way, and are allowed to have the same properties. On the other hand supporters never have locks: it makes no sense to be able to lock a tabletop, for instance, and Inform will not allow any discussion of the matching key of a supporter, or of a supporter being locked or unlocked.

⬆ Start of Chapter 3: Things

⬅ Back to §3.12. Doors

➡ Onward to §3.14. Devices and descriptions

⬇ Example 23: ★ **Disenchantment Bay 6** Disenchantment Bay: locking up the charter boat's fishing rods.

⬇ Example 24: ★★ **Neighborhood Watch** A locked door that can be locked or unlocked without a key from one side, but not from the other.

§3.14. Devices and descriptions

A "device" is another of the standard kinds of thing, and should be used for anything which can be switched on or off: a light switch, say, or a slide projector. Devices are generally machines, clockwork or electrical. A device is always either "switched on" or "switched off", but is switched off unless we specify otherwise.

That makes three kinds of thing which will likely change their appearance according to which of their two possible states they are in: doors and containers, which can be open or closed; and devices, which can be switched on or switched off. We would like to produce text accordingly, and we can do this using Inform's ability to make (almost) any piece of text change with circumstances. For instance:

```
The coffin is an openable container in the Undertomb. "[if open]The lid of a plank coffin yawns open.[otherwise]A plank coffin lies upon the dirt floor of the Tomb."
```

We could use a similar trick to make the appearance of a device change "if switched on". There will be much more about text substitutions, as instructions in square brackets like these are called, in later chapters.

★ *See* **Text with substitutions** *for more on varying what is printed*

⬆ Start of Chapter 3: Things

⬅ Back to §3.13. Locks and keys

➡ Onward to §3.15. Light and darkness

⬇ Example 25: ★ **Disenchantment Bay 7** Disenchantment Bay: making the radar and instruments switch on and off.

⬇ Example 26: ★★ **Down Below** A light switch which makes the room it is in dark or light.

§3.15. Light and darkness

Rooms can be "dark" or "lighted", though they are lighted by default, and are lighted in all the examples we have seen so far.

```
The Sinister Cave is a dark room. "A profoundly disquieting rock formation, apparently sculptured by some demonic hand, this is not a cave in which to relax."
```

When the player is in a dark room, he can still go in various directions, but he cannot see the room description or interact with any of the objects in the room, except those he is holding. This means that, unless we should change the Cave in some way during play, the text above ("A profoundly...") will only be read if the player succeeds in bringing light into the Cave, perhaps by bringing along the following:

```
The flaming torch is in the Sandy Passage. "Stuck loosely into the sand is a flaming torch." The flaming torch is lit.
```

A thing with the property of being "lit" will enable the player to see inside dark rooms, and to carry out other activities requiring light, such as examining items. A lit thing in an open container will still light up a room; a lit thing in a closed container will not, unless the container has been given the "transparent" property.

It is possible to adjust the way darkness behaves, and we will see more on this topic in the chapter on Activities.

★ *See* **Printing a refusal to act in the dark** *for the first of several ways to control what is printed in the dark*

⬆ Start of Chapter 3: Things

⬅ Back to §3.14. Devices and descriptions

➡ Onward to §3.16. Vehicles and pushable things

§3.16. Vehicles and pushable things

Next in the tour of standard kinds is the "vehicle". This behaves like (indeed, is) an enterable container, except that it will not be portable unless this is specified.

```
In the Garage is a vehicle called the red sports car.
```

The player can enter the sports car and then move around riding inside it, by typing directions exactly as if on foot: and the story will print names of rooms with "(in the red sports car)" appended, lest this be forgotten.

We have already seen that some things are portable, others fixed in place. In fact we can also make a third sort of thing: those which, although not portable, can be pushed from one room to another with commands like "push the wheelbarrow north". At a pinch, we might just be willing to allow:

```
The red sports car is pushable between rooms.
```

But of course this is a property which almost any thing can have, not just a vehicle. (Only "almost" because Inform will not allow a door to be pushable between rooms, in the interests of realism rather than surrealism.)

If we need vehicles which the passenger sits on top of, like a horse or a tractor, the standard "vehicle" kind will not be ideal. However, by loading one of the extensions which comes ready-installed:

```
Include Rideable Vehicles by Graham Nelson.
```

...we are provided with two more kinds, "rideable vehicle" and "rideable animal", just right for the tractor and the horse respectively. (As with all extensions, the documentation can be seen by clicking Go on some source which contains the above line, and then turning to the Contents index; or from the Installed Extensions tab of the Extensions panel.)

★ *See* **Going by, going through, going with** *for further ways to customize vehicle behaviour*

⬆ Start of Chapter 3: Things

⬅ Back to §3.15. Light and darkness

➡ Onward to §3.17. Men, women and animals

⬇ Example 27: ★ **Peugeot** A journey from one room to another that requires the player to be on a vehicle.

⬇ Example 28: ★★ **Disenchantment Bay 8** Disenchantment Bay: a pushable chest of ice for the boat.

⬇ Example 29: ★★★ **Hover** Letting the player see a modified room description when he's viewing the place from inside a vehicle.

§3.17. Men, women and animals

Rounding out the standard kinds provided by Inform are four for living things: "person", which is a kind of thing, and "man", "woman" and "animal", all kinds of person. For instance:

```
In the Ballroom is a man called Mr Darcy.
```

For the time being, men and women will be little more than waxworks: they will come to life only when we go beyond the present stage of creating an initial state of the world.

People can be male or female: this is an either/or property for the "person" kind, and it affects play at run-time a little, because the player can use "him" and "her" to refer to male or female people encountered. Men and women are always male and female respectively, and for animals we can choose either way, for example making a stallion male or a nanny goat female. Animals are male unless we say otherwise.

If our animal is instead something like a beetle or an earthworm, where gender doesn't seem to matter or even to exist, we can use the further property "neuter":

```
The spider is a neuter animal in the Bathroom.
```

The Standard Rules don't make people behave differently according to their genders, and the main difference comes down to language: whether we want the animal to be called "her", or "it". Because of the existence of "neuter", we sometimes need to be cautious about the use of the adjective "male": since Inform, partly for historical reasons, uses an either/or property for masculinity, neuter animals are also "male".

Start of Chapter 3: Things
Back to §3.16. Vehicles and pushable things
Onward to §3.18. Articles and proper names
Example 30: **Disenchantment Bay 9** Disenchantment Bay: enter the charter boat's Captain.

§3.18. Articles and proper names

Suppose we have said that:

```
In the Ballroom is a man called Mr Darcy.
```

When the Ballroom is visited, the man is listed in the description of the room as "Mr Darcy", not as "a Mr Darcy". This happened not because Inform recognised that Darcy is a proper name, or even because men tend to have proper names, but because Inform noticed that we did not use "a", "an", "the" or "some" in the sentence which created him. The following shows most of the options:

```
The Belfry is a room. A bat is in the Belfry. The bell is in the Belfry. Some woodworm are
in the Belfry. A man called William Snelson is in the Belfry. A woman called the sexton's
wife is in the Belfry. A man called a bellringer is in the Belfry.

In the Belfry is a man called the vicar. The indefinite article of the vicar is "your local".
```

In the resulting story, we read:

```
You can see a bat, a bell, some woodworm, William Snelson, the sexton's wife, a
bellringer and your local vicar here.
```

The subtlest rule here is in the handling of "the". We wrote "The bell is in the Belfry", but this did not result in the bell always being called "the" bell: in fact, writing "A bell is in the Belfry" would have had the same effect. On the other hand, "A woman called the sexton's wife is in the Belfry." led to the wife always being known as "the" sexton's wife, not "a" sexton's wife, because Inform thinks the choice of article after "called" shows more of our intention than it would elsewhere. These rules will never be perfect in all situations, so we are also allowed to specify indefinite articles by hand, as the vicar's case shows.

"Some" is worth a closer look, because English uses it in several different ways. By introducing the woodworm with "some", above, we established that it was plural. We might imagine that there are many worms, even though they are represented by a single thing in Inform. We can expect to see text in the story such as:

```
You can see some woodworm here.
The woodworm are fixed in place.
```

But suppose we wanted something which there is an amount of, but which is not made up of individual items - a so-called mass noun like "water", or "bread". Now we can write:

```
The water is here. The indefinite article is "some".
```

and this time Inform does not treat the "some water" thing as a plural, so we might read:

```
You can see some water here.
The water is hardly portable.
```

rather than "The water are hardly portable."

Finally, we can override these settings, if they still come out not as we intend, by explicitly changing the either/or properties "singular-named" (vs "plural-named") and "proper-named" (vs "improper-named").

Start of Chapter 3: Things

Back to §3.17. Men, women and animals

Onward to §3.19. Carrying capacity

Example 31: ★ **Belfry** You can see a bat, a bell, some woodworm, William Snelson, the sexton's wife, a bellringer and your local vicar here.

Example 32: ★★ **Gopher-wood** Changing the name of a character in the middle of play, removing the article.

§3.19. Carrying capacity

The containers and supporters created so far have been boundlessly capacious: or rather, though we seldom notice the difference, have had a maximum carrying capacity of 100 items. This is clearly unrealistic for a small purse or a modest mantelpiece. We can impose upper limits with sentences like so:

```
The carrying capacity of the jewelled purse is 2.
```

```
The bijou mantelpiece has carrying capacity 3.
```

Attempts by the player to overfill, or overload, will now be rebuffed with a message such as "There is no room on the mantelpiece".

The player is not a container or a supporter, but nevertheless does have a carrying capacity: this is interpreted to mean the maximum number of items which can be carried at once.

```
The carrying capacity of the player is 4.
```

These restrictions only apply to the player (and other in-world characters): as the omnipotent creators, we are not restrained by them. Nothing prevents this:

```
The carrying capacity of the jewelled purse is 2. The diamond, the ruby and the sapphire
are in the purse.
```

The player will be able to remove all three items, but only put two of them back. (This is probably something we only want very occasionally: perhaps to create a sack stuffed almost to bursting point.)

Start of Chapter 3: Things

Back to §3.18. Articles and proper names

Onward to §3.20. Possessions and clothing

§3.20. Possessions and clothing

We have seen how to place objects in rooms, and in containers or on supporters. But what about people? Perhaps it could be said that they "contain" the fillings in their teeth, or "support" a top hat, but this is not very natural. Inform therefore never speaks of things being "in" or "on" people. Instead, they have two sorts of possessions: the things they carry, and the things they wear. (Body parts, such as arms and legs, are different again: see "parts" below for a clue to how to do these.) Thus:

```
Mr Darcy wears a top hat. Mr Darcy carries a silver sword.
```

In fact, Inform deduces from this not only who owns the hat and the sword, but also that Darcy has the kind "person", because only people can wear or carry.

As all the assertion verbs do, "to wear" and "to carry" have participles which Inform knows about. So we could equally well write:

```
The scarlet coat is worn by Mr Wickham. The duelling pistol is carried by Mr Wickham.
```

If we do not specify who does the wearing, or carrying, then this is assumed to be the player. Thus:

```
A brass lantern and a rusty iron key are carried. The mosquito-repellent hat is worn.
```

It would make no sense to "wear" the key, for instance, so Inform needs to distinguish between what is clothing and what is not. It does this with an either/or property called "wearable": if something has this property then the player will be allowed to wear it, provided it can first be picked up. Anything which is worn by somebody at the start of play is assumed to be wearable (unless we say otherwise). But if nobody is initially wearing the item in question, then we have to be explicit:

```
The player carries a scarlet gown. The gown is wearable.
```

(When we come to asking questions about the current situation, we will need to remember that "to carry" and "to wear" are different. Thus "if Lancelot carries the plate armour" will not be true if he is wearing it rather than carrying it under his arm. As we will later see, we can instead vaguely say "if Lancelot has the plate armour" to mean either carrying or wearing.)

See **To carry, to wear, to have** *for a more detailed explanation of carrying, wearing, and possessing as Inform understands them*

Start of Chapter 3: Things

Back to §3.19. Carrying capacity

Onward to §3.21. The player's holdall

Example 33: **Disenchantment Bay 10** Disenchantment Bay: things for the player and the characters to wear and carry.

§3.21. The player's holdall

When the player has only limited carrying capacity, play is likely to be tiresome, but we can make life easier by providing a way for the player to carry endless items without dozens of free hands to hold them all:

```
"Sackcloth"

The Attic is a room. The old blue rucksack is a player's holdall. The player is wearing the rucksack.

The carrying capacity of the player is 3.

In the Attic are a CD entitled No Smoke Without Fire, a 70s photograph of an American winning Wimbledon, a fraxinus branch, an urn holding your late great-aunt's remains, a convention badge from the American Society of Hypertension and a ghost story by M R James.
```

This example story introduces a new kind of container, the "player's holdall". This is a kind of which most stories will contain at most one example, but in principle there can be any number. A player's holdall is a capacious bag into which the player automatically places surplus items whenever his or her hands are full: trying the above example story and getting the items one by one will give the general idea.

Of course, if the carrying capacity of the player is never reached then there will never be any surplus items and a player's holdall will behave just like any other (portable, usually openable) container.

See **Units** for the tools to implement a more sophisticated capacity system

Start of Chapter 3: Things

Back to §3.20. Possessions and clothing

Onward to §3.22. Food

Example 34: **Disenchantment Bay 11** Disenchantment Bay: making a holdall of the backpack.

§3.22. Food

We have nearly reached the end of the chapter on Things, but one either/or property for things remains: every thing is either "edible" or "inedible". Unless we say otherwise, things are inedible. But for instance we might write:

```
The player carries a Macintosh apple. The Macintosh is edible.
```

(The type of computer is named after a variety of apple descended from a tree cultivated in 1811 by John McIntosh of Ontario.) Edible things are just like inedible ones, except that the player can EAT them. This will usually only consume the foodstuff in question, effectively destroying it, but using techniques from later chapters we could make the consequences more interesting.

⬆ Start of Chapter 3: Things

⬅ Back to §3.21. The player's holdall

➡ Onward to §3.23. Parts of things

§3.23. Parts of things

Everything has one and only one kind. This is both good and bad: good for clarity, bad if something needs to behave in two different ways at once. How might we simulate a car with an ignition key, given that no single thing can be both a "vehicle" and a "device" at the same time?

The Inform world model takes the view that such a car is too complicated to be simulated with a single thing. Instead it should be simulated as a vehicle (the car) which has a device (the ignition) attached. This is done using a third kind of containment to those seen so far ("in..." and "on..."): "part of".

```
"Buttons"

The Confectionary Workshop is a room. The Chocolate Machine is here. "The Chocolate Machine has pride of place. A lever and two buttons, one white, the other brown, seem to be the only controls. On top is a hopper."

A container called the hopper is part of the Chocolate Machine. The lever, the white button and the brown button are parts of the Chocolate Machine.

The Chocolatier's desk is here. "The Chocolatier evidently works at the imposing green-leather topped desk facing the Machine. It has three drawers with brass handles."

The upper drawer, the middle drawer and the lower drawer are parts of the desk. The upper drawer, the middle drawer and the lower drawer are openable closed containers. In the middle drawer is a sugared almond. In the lower drawer is a Battenburg cake. On the desk is a liquorice twist.

The cake, the twist and the almond are edible.
```

The machine and the desk each have several "parts" representing subsidiary pieces of themselves. The desk is a "supporter" (it needs to be, for the liquorice twist to be on top) but also has three "containers" attached, each of which can be opened or closed independently.

In the interests of realism, the standard rules of play protect these composite things. Thus if the desk were to be moved elsewhere (rolling on sugar casters perhaps) then its parts would move with it, and the player is not allowed to detach parts of things: the drawers can be opened or closed, but not pulled out altogether.

Note that rooms and regions are not allowed to have parts. (Rooms are already parts of regions, and to divide up rooms, we can either make several rooms or place containers or other obstacles in a single one.)

Start of Chapter 3: Things

Back to §3.22. Food

Onward to §3.24. Concealment

Example 35: ★★★ **Fallout Enclosure** Adding an enclosure kind that includes both containers and supporters in order to simplify text that would apply to both.

Example 36: ★★★ **Brown** A red sticky label which can be attached to anything in the game, or removed again.

Example 37: ★★★★ **Disenchantment Bay 12** A final trip to Disenchantment Bay: the scenario turned into a somewhat fuller scene, with various features that have not yet been explained.

§3.24. Concealment

Though realism can become tiresome in interactive fiction, there are times when we cannot go along with Inform's normal assumption that all of a person's possessions are visible to everybody else. People are not like containers, which either show all of their holdings or not, according to whether they are open or transparent. If a man is carrying a fishing rod and a wallet, one will be on open show, the other not. Some clothing is outwardly visible, but not all.

Whether or not something is concealed is not like the either/or properties we have seen so far - such as being "open" or "closed" - because it is not really a property of the thing itself, but depends on the habitual behaviour of its current owner. To talk about behaviour we have to use sentences of a kind not seen so far, and which will not fully be explained for some chapters to come.

But straightforward cases are easy to write, if only by imitating the following examples.

Here we make the Cloaked Villain invariably conceal anything she is holding or wearing:

```
Rule for deciding the concealed possessions of the Cloaked Villain: yes.
```

At which point we think about it more carefully, and then rewrite:

```
Rule for deciding the concealed possessions of the Cloaked Villain: if the particular
possession is the sable cloak, no; otherwise yes.
```

(A rule which says neither "yes" nor "no" will decide yes, but it's best to spell out exactly what's wanted.)

Parts are treated exactly as if clothes or items being held, and the following will make the face and inscription on a coin invisible unless the player is holding it - the idea being that they are too small to be seen from farther away.

```
The coin is in the Roman Villa. The face and inscription are parts of the coin. Rule for
deciding the concealed possessions of the coin: if the coin is carried, no; otherwise yes.
```

There is also an either/or property called "described"/"undescribed", intended to be used only as a last resort, but which has the ability to hide something from room descriptions. This not really hiding: the idea is that "undescribed" should be used only for cases where some other text already reveals the item, or where its presence is implicit. Even then, it should only be used when the item is intended to be taken or moved by the player at some point - if the item isn't intended to move, it's much better to make it "scenery". (There's only one commonly-found example - the player's own body, the "yourself", is undescribed.)

Note that the "undescribed" property is automatically removed from anything carried by, worn by or part of the player, even indirectly; and that nothing on top of an "undescribed" supporter will be visible in a room description, even if it itself is "described". (Scenery supporters don't suffer from that restriction, which is one reason scenery is a better option when possible.)

⬆ Start of Chapter 3: Things

⬅ Back to §3.23. Parts of things

➡ Onward to §3.25. The location of something

⬇ Example 38: ★★★ **Search and Seizure** A smuggler who has items, some of which are hidden.

§3.25. The location of something

The model world created by Inform is partitioned into rooms. This means that everything which exists in the model world, exists in one of the rooms. If we write a sentence such as

```
Professor Wilderspin is a man.
```

and say nothing more about Wilderspin, then he does not physically exist at the start of the story: he is said to be "out of play", and stays that way until we move him into one of the rooms. A better metaphor might be that he is waiting in the wings, ready to come onto the stage.

Every thing is either out of play, or can be found in one of the rooms, and the property "location of X" gives us the room in question. The following condition tests, in effect, whether Wilderspin is in play:

```
if the location of Wilderspin is a room, ...
```

Which uses a new phrase:

location of (object) **... room**

This phrase produces the room which, perhaps indirectly, contains the object given. Example: if the player stands in Biblioll College and wears a waistcoat, inside which is a fob watch, then

```
location of the fob watch
```

is Biblioll College. In general, a thing cannot be in two rooms at once, but there are two exceptions: two-sided doors, present on both sides, and backdrops. The "location of" a door is its front side, but a backdrop has no location. (Objects which are not things at all, such as rooms and directions, also have no location.)

We very often want to know the location of the player, and this is more simply called just "the location". (This is actually a value that varies rather than a phrase, but that's a technicality we can ignore here.)

The idea of indirect containment is useful enough to have a name: Inform calls it "enclosure". A thing encloses whatever is a part of itself, or inside itself, or on top of itself, and it also encloses anything that they enclose. And when something moves around, anything it encloses will move with it. In the example above, Biblioll College (a room) and the player (a person) both enclose the fob watch and the waistcoat. (The small print: a door is enclosed by the rooms on both sides; a backdrop is never enclosed.)

Enclosure is only useful when being used as a question. So the following is fine:

```
if the player encloses the fob watch, ...
```

But these will produce problem messages:

```
The player encloses the fob watch. The location of the trilobite is the Museum.
```

because they are too vague. Inform needs to know exactly where the fob watch and the trilobite will begin the story, whereas these sentences leave room for doubt about who or what is actually holding them.

Start of Chapter 3: Things

Back to §3.24. Concealment

Onward to §3.26. Directions

Example 39: ★★ **Van Helsing** A character who approaches the player, then follows him from room to room.

§3.26. Directions

"Direction" is a kind which is quite unlike most of those seen so far. While it has to do with the physical world, a direction does not exactly belong to it. One cannot find "southeast" sitting on a shelf. "Direction" is not a kind of thing, nor a kind of room: it is a kind in its own right.

Every direction has an "opposite" property, which is always another direction. These occur in matched pairs. The opposite of north is south, just as the opposite of south is north. The opposite of southeast is northwest, the opposite of inside is outside, and so on. When Inform reads a sentence like...

```
Bangkok is south of Nakhon Sawan.
```

...it assumes that the opposite map connection is probably also valid, so that

```
Nakhon Sawan is north of Bangkok.
```

The chapter began with the twelve directions built into Inform:

```
north, northeast, east, southeast, south, southwest, west, northwest, up, down, inside,
outside
```

But the built-in set is not always appropriate. Sometimes this is too many; if we wanted to write about a Flatland, for instance, then up and down ought to go. But in practice it is better not to abolish them as directions but instead to forbid travelling in them. (See the Recipe Book for examples.)

But away from our familiar Earth, the usual frame of reference loses its meaning. Terry Pratchett's "Discworld" comedies, set on a rotating disc, use the directions turnwise, widdershins, hubwards and rimwards. On board a Zeppelin airship, which constantly changes its course, the cockpit has no fixed compass bearing from the passenger cabin: it is not very naturally "north". In zero gravity, there is no up or down. Mars does not have a magnetic core, so a compass doesn't work there.

New directions must always be created in opposing pairs, and each **must** be declared with a clear simple sentence of the form "X is a direction." For instance:

```
Turnwise is a direction. The opposite of turnwise is widdershins.
Widdershins is a direction. The opposite of widdershins is turnwise.
Hubwards is a direction. The opposite of hubwards is rimwards.
Rimwards is a direction. The opposite of rimwards is hubwards.
```

It is then possible to write, say, that:

```
Ankh-Morpork is hubwards of Lancre and turnwise from Borogravia.
```

Of course the Map page of the Index for the project normally draws a map based on compass bearings, so it will get a little befuddled by this. But the map drawn in the Index can be given hints to improve its legibility. More on this later, but for now note that

```
Index map with turnwise mapped as east.
```

maps turnwise directions as if they were east, that is, pointing rightwards on the page. (This has no effect on the story file produced; it does not mean turnwise is simply a new name for east; it affects only the look of the Index map, which is only a convenience for the author in any case.)

At one time, directions had to have shortish names (up to three words only), but that's no longer true:

```
Just the tiniest smidge off magnetic north is a direction. The opposite of
just the tiniest smidge off magnetic north is just the tiniest smidge
off magnetic south.
Just the tiniest smidge off magnetic south is a direction. The opposite of
just the tiniest smidge off magnetic south is just the tiniest smidge
off magnetic north.
```

⬆ Start of Chapter 3: Things

⬅ Back to §3.25. The location of something

➡ Onward to Chapter 4: Kinds: §4.1. New kinds

⬇ Example 40: ★★★ **Prisoner's Dilemma** A button that causes a previously non-existent exit to come into being.

⬇ Example 41: ★★★ **The World of Charles S. Roberts** Replacing the ordinary compass bearings with a set of six directions to impose a hexagonal rather than square grid on the landscape.

⬇ Example 42: ★★★★ **Fore** Understand "fore", "aft", "port", and "starboard", but only when the player is on a vessel.

Examples from Chapter 3: Things

⬆ Start of this chapter

➡ Chapter 4: Kinds

↘ Indexes of the examples

2 ★ **Example Bic** RB
Testing to make sure that all objects have been given descriptions.

It may occasionally be useful to check whether all objects in our game have a given property. Here we have a "not for release" section that will run at the start of the game and alert us to any objects lacking description:

```
"Bic"

Section 1 - Testing descriptions - Not for release

When play begins (this is the run property checks at the start of play rule):
    repeat with item running through things:
```

```
    if description of the item is "":
        say "[item] has no description."

Section 2 - Story

The Staff Break Room is a room.

The player carries an orange, a Bic pen, and a napkin. The description of the orange is "It's a small hard pinch-skinned thing from the lunch room, probably with lots of pips and no juice."

The description of the napkin is "Slightly crumpled."
```

3

★ Example Verbosity 1

Making rooms give brief room descriptions when revisited.

RB

By default, the description of a room is printed every time the player enters a room.

On a device with very limited screen space, however, we might wish to supplant that behavior with "brief" descriptions. In Brief mode, Inform prints room descriptions only when the player enters that room for the first time. Afterwards, the text is skipped, for brevity, though the player can see it again at any time by typing LOOK.

As we saw in the previous chapter, we can set "use options" to control certain aspects of the player's experience. One of the use options is the option to

```
Use brief room descriptions.
```

which changes the defaults so that the description of a room is printed only the first time the player enters.

```
"Verbosity"

Use brief room descriptions.

The Wilkie Memorial Research Wing is a room. "The research wing was built onto the science building in 1967, when the college's finances were good but its aesthetic standards at a local minimum. A dull brown corridor recedes both north and south; drab olive doors open onto the laboratories of individual faculty members. The twitchy fluorescent lighting makes the whole thing flicker, as though it might wink out of existence at any moment.

The Men's Restroom is immediately west of this point."

The Men's Restroom is west of the Research Wing. "Well, yes, you really shouldn't be in here. But the nearest women's room is on the other side of the building, and at this hour you have the labs mostly to yourself. All the same, you try not to read any of the things scrawled over the urinals which might have been intended in confidence."
```

```
Test me with "west / east".
```

If we type "test me" during play, these commands will be carried out automatically, and we can see that when we return to the Research Wing, the description is not given a second time.

Some notes: the player can also turn full-length descriptions on or off with the commands "verbose" and "brief", or set a minimal-description setting with the command "superbrief". This power still belongs to the player even if we have set the use option to show brief room descriptions by default.

Moreover, we can ourselves check what the state of the descriptions is, with

```
if set to sometimes abbreviated room descriptions: ...
if set to unabbreviated room descriptions: ...
if set to abbreviated room descriptions: ...
```

Finally, it is possible to exercise more precise control over what the player sees on his first and subsequent visits to a room; see the next example for details.

4

★★ Example **Slightly Wrong**

A room whose description changes slightly after our first visit there.

RB

A fairly common effect in interactive fiction is a room which is described differently on the first visit than on subsequent visits. We can produce this effect as follows:

```
"Slightly Wrong"

Awning is a room. "A tan awning is stretched on tent poles over the dig-site, providing a little shade to the workers here; you are at the bottom of a square twenty feet on a side, marked out with pegs and lines of string. Uncovered in the south face of this square is an awkward opening into the earth."

Slightly Wrong Chamber is south of the Awning. "[if unvisited]When you first step into the room, you are bothered by the sense that something is not quite right: perhaps the lighting, perhaps the angle of the walls. [end if]A mural on the far wall depicts a woman with a staff, tipped with a pine-cone. She appears to be watching you."

Test me with "look / s / look".
```

Note the "[if unvisited]..." in the description of the Slightly Wrong Chamber. A room is considered to be "unvisited" until after the player has seen its description for the first time.

The bracketed text creates a special rule for printing; we will learn more about these in the sections on text with variations and text with substitutions.

Some further fine print: we might write our condition as "if unvisited", "if the location is unvisited", or "if the Chamber is unvisited" -- all of these constructions would be acceptable, but in the absence of more specifics, the condition is understood to apply to the object whose description it is.

5 ★ **Example** **Port Royal 1** *RB*

A partial implementation of Port Royal, Jamaica, set before the earthquake of 1692 demolished large portions of the city.

```
"1691"

Fort James is a room. "The enclosure of Fort James is a large, roughly
hexagonal court walled with heavy stone. The walls face the entrance to Port
Royal Harbour, and the battery of guns is prepared to destroy any enemy ship
arriving."
```

Unless we arrange otherwise, this will be the first room in the game because it is the first we have defined.

For subsequent rooms, we do not have to say explicitly that they are rooms, as long as they are connected to a room on the map. For instance, this will automatically make Thames Street End a room:

```
Thames Street End is south of Fort James. "The ill-named Thames Street runs
from here -- at the point of the peninsula -- all the way east among houses and
shops, through the Fish Market, edging by the round front of Fort Carlisle, to the
point where the town stops and there is only sandy spit beyond. Lime Street,
wider and healthier but not as rich, runs directly south, and to the north the road
opens up into the courtyard of Fort James."

Water Lane is east of Thames Street End. "Here Thames Street -- never very
straight -- goes steeply southeast for a portion before continuing more directly to
the east.

Water Lane runs south toward Queen Street, and facing onto it is the New
Prison -- which, in the way of these things, is neither. It did serve in that capacity
for a time, and in a measure of the villainy which has been usual in Port Royal
from its earliest days, it is nearly the largest building in the town."
```

If we have some concern that the room name will be confused with an existing name, we can be more explicit about it using "called":

```
East of Water Lane is a room called Thames Street at the Wherry Bridge.
Thames Street at the Wherry Bridge has the description "To the southwest is the
fishmarket; directly across the street is the entrance to a private alley through a
brick archway."

The Private Alley is south of Thames Street at the Wherry Bridge. "You're just
outside the tavern the Feathers. To the north, under a pretty little archway, is the
active mayhem of Thames Street, but the alley narrows down to a dead end a
little distance to the south."
```

And now we get "inside", which generates a space treated as its own area on the map.

```
The Feathers is inside from the Private Alley. "Newly built with brick, replacing
the older Feathers tavern that used to stand here. It sells wines in quantity, as
well as serving them directly, and the goods are always of the best quality.
There's a room upstairs for those wanting to stay the night." The Feathers
Bedroom is above the Feathers.
```

And if we like we can declare a number of rooms for which we will come back and write the descriptions later. There is no obligation for the description to occur at the first definition of the room.

```
Lime Street is south of Thames Street End.
```

For efficiency, we can also write multiple sets of connections at once:

```
Queen Street East is east of Queen Street Middle and south of Private Alley.
```

Clicking Go will translate this description into a sketchy but working simulation of Port Royal, in which we can type movement commands like EAST or SOUTH to explore the streets. Looking at the World tab of the Index, we can also see a schematic map of the simulation as it currently stands. Like the rest of the Index, this is provided entirely for the author's benefit, and is not visible to the player. (Though if we do decide that we want players to have access to a printed map while they play, Inform can help: we will return to the layout of Port Royal in the chapter on Publishing.)

The following Test command allows us to type TEST ME and explore the map we just devised:

```
Test me with "s / e / e / s / in".
```

6

★★★ Example Up and Up

RB

Adding a short message as the player approaches a room, before the room description itself appears.

Sometimes when a player moves from one room to another, we want to imply that a considerable amount of time elapses, or that something interesting occurs on the way. In that case, we might want to print more than just the room description itself. Here is how we might define a couple of rooms that are far apart:

```
"Up and Up"

The Plain of the Skull is below the Endless Tower. The description of the Plain of
the Skull is "A vast and trackless plain, enlivened only by the bones of those who
have previously tried and failed to cross. Above you is the Endless Tower, which
rises half-way to the moon."
```

```
The description of the Endless Tower is "From up here the Plain of the Skull seems only a small bald patch: the world is round and most of it is covered with trees. Far off to the southwest is a shimmering surface that might be water; but there are no signs of cities or civilizations, only the lizard-skeletons."
```

And now we borrow from the instructions on Actions to create our actual message. "Before..." introduces a rule that occurs when the player tries to do something; in this case, we will make a Before rule for going to the tower.

```
Before going to the Endless Tower:
    say "You climb... and climb... and climb... The sun sets. The moon rises. The wind begins to blow. You continue to climb..."

The player carries a bit of harness. The description of the harness is "A strip of worked leather and a loop of metal, scavenged from one of the skeletons on the plain. Without it, you might think your entire quest was in vain."

Test me with "look / up".
```

7 ★★★ **Example** **Starry Void** RB

Creating a booth that can be seen from the outside, opened and closed, and entered as a separate room.

Sometimes we may want a room to be visible from the outside in one location, but treated as a separate location when we are inside. The simplest way to do this is to make the exterior form of the object into a door object, and to describe it differently from different vantage points. (Doors in general are described more fully in the Doors section of the Things chapter.)

```
"Starry Void"

The Center Ring is a room.

The magician's booth is a door. "[if the player is in Center Ring]A magician's booth stands in the corner, painted dark blue with glittering gold stars.[otherwise if the magician's booth is closed]A crack of light indicates the way back out to the center ring.[otherwise]The door stands open to the outside.[end if]".
```

Here we've arranged for the booth to be described in the initial room description in different ways depending on where the player is when viewing it. We might like to do the same if the player takes a closer look:

```
Instead of examining the magician's booth in the Center Ring:
    say "It is dark blue and glittering with gold stars. [if the booth is open]The door currently stands open[otherwise]It has been firmly shut[end if]."

Instead of examining the magician's booth in the Starry Void:
    say "The booth door is [if the magician's booth is open]wide open[otherwise]shut, admitting only a thin crack of light[end if]."
```

And now we put it in place:

```
The magician's booth is inside from Center Ring and outside from Starry Void.
```

...and make sure that the booth-and-door object responds to all the names we have used for it in different places:

```
Understand "door" or "of" or "the" or "light" or "crack" or "thin crack" as the booth.

Test me with "examine booth / open door of the booth / in / examine door / close door / look / examine crack of light".
```

A final nice touch, if we're so inclined, is to borrow from the Basic Actions chapter and make the player automatically open the booth door before trying to enter:

```
Before going through the closed magician's booth:
    say "(first opening the door of the booth)[command clarification break]";
    silently try opening the booth.
```

For the contrasting case of a space that is nested inside another place and is not its own room -- say a stall at an open-air market, or a rowboat on a lake -- see the example "Tamed".

---

8 ★ **Example Port Royal 2** RB

Another part of Port Royal, with less typical map connections.

```
"1691"

Thames Street End is a room.
```

If we check out a map of historic Port Royal, we find that Thames Street End bends around the northwest tip of the peninsula and becomes the (very) roughly north/south Fisher's Row. We can't put Fisher's Row south of Thames Street End, though, because Lime Street is already going that way. So instead, let's have a map connection that bends around from west to north:

```
West of Thames Street End is north of Fisher's Row.
```

Now continuing west along Thames Street, or north along Fisher's Row, will bring us around the corner in question. Asymmetric map connections should be used carefully. They're good for representing the layout of the real world, which tends not to be laid out on a convenient square matrix, but if exits are not described clearly they can be disorienting for the player. So let's be sure to make things clear:

```
The description of Fisher's Row is "A waterfront street that runs south towards Chocolata Hole, where the small craft are harboured. It also continues north around the tip of the peninsula from here, turning into the east-west Thames Street."
```

Meanwhile, suppose Fort James is in a prominent position, raised a bit from its surroundings; maybe the player should be able to go down from there, as well as south, to get to Thames Street End.

```
Thames Street End is down from Fort James. Thames Street End is south from Fort James.
```

But we don't want the upward direction to work:

```
Up from Thames Street End is nowhere.

Test me with "n / d / u / w / e / n / s".
```

---

9 ★ **Example** **The Unbuttoned Elevator Affair** *RB*

A simple elevator connecting two floors which is operated simply by walking in and out, and has no buttons or fancy doors.

This is very simple. The interior of the elevator is a single room, but which is mapped east of both of its termini. The reverse map connection, west from the elevator, can only go to a single room, and that's what determines which floor the elevator is on.

```
"The Unbuttoned Elevator Affair"

UNCLE Headquarters is a room. "The steel nerve-center of the free world's battle against the Technological Hierarchy for the Removal of Undesirables and the Subjugation of Humanity. Being against technology, we have only a very simple elevator to the east."

Del Floria's Tailor Shop is a room. "Only trained anti-THRUSH agents recognise the booth in the east wall as a secret elevator."

The Secret Elevator is east of UNCLE Headquarters. The Secret Elevator is east of Del Floria's Tailor Shop.

After going to the Secret Elevator:
    say "The doors automatically close, there is a rush of motion, and they open again.";
    if UNCLE Headquarters is mapped west of the Secret Elevator, now Del Floria's Tailor Shop is mapped west of the Secret Elevator;
    otherwise now UNCLE Headquarters is mapped west of the Secret Elevator;
    continue the action.

Test me with "east / west / east / west".
```

---

10 ★ **Example** **Port Royal 3** *RB*

Division of Port Royal into regions.

```
"1691"
```

We should go ahead and do all our room definitions first...

```
Fort James is a room. "The enclosure of Fort James is a large, roughly
hexagonal court walled with heavy stone. The walls face the entrance to Port
Royal Harbour, and the battery of guns is prepared to destroy any enemy ship
arriving."

Thames Street End is south of Fort James. "The ill-named Thames Street runs
from here -- at the point of the peninsula -- all the way east among houses and
shops, through the Fish Market, edging by the round front of Fort Carlisle, to the
point where the town stops and there is only sandy spit beyond. Most of that
stretch is full of people at all hours. Imported goods are moved off of ships and
taken to distributors; exported goods are brought to be loaded; and there is one
public house and brothel for every ten inhabitants.

Lime Street, wider and healthier but not as rich, runs directly south, and to the
north the road opens up into the courtyard of Fort James."

Lime Street is south of Thames Street End. West of Thames Street End is north
of Fisher's Row. The description of Fisher's Row is "A waterfront street that runs
south towards Chocolata Hole, where the small craft are harboured. It also
continues north around the tip of the peninsula from here, turning into the east-
west Thames Street."

Thames Street End is down from Fort James. Up from Thames Street End is
nowhere.

Water Lane is east of Thames Street End. "Here Thames Street -- never very
straight -- goes steeply southeast for a portion before continuing more directly to
the east.

Water Lane runs south toward Queen Street, and facing onto it is the New
Prison -- which, in the way of these things, is neither. It did serve in that capacity
for a time, and in a measure of the villainy which has been usual in Port Royal
from its earliest days, it is nearly the largest building in the town."

East of Water Lane is a room called Thames Street at the Wherry Bridge.
Thames Street at the Wherry Bridge has the description "To the southwest is the
fishmarket; directly across the street is the entrance to a private alley through a
brick archway."

The Fishmarket is southwest of Thames Street at the Wherry Bridge.

The Private Alley is south of Thames Street at the Wherry Bridge. "You're just
outside the tavern the Feathers. To the north, under a pretty little archway, is the
active mayhem of Thames Street, but the alley narrows down to a dead end a
little distance to the south."

The Feathers is inside from the Private Alley. "Newly built with brick, replacing
the older Feathers tavern that used to stand here. It sells wines in quantity, as
well as serving them directly, and the goods are always of the best quality.
```

```
There's a room upstairs for those wanting to stay the night." The Feathers Bedroom is above the Feathers.

Thames Street by the King's House is east of Thames Street at the Wherry Bridge. "The King's House is reserved for the use of the Governor, but he does not live in it, and it is frequently being rented out to some merchant so that the government will at least derive some value from it. It is nearly the least interesting establishment on Thames Street, and the crowd -- which, to the west, is extremely dense -- here thins out a bit."

Thames Street before Fort Carlisle is east of Thames Street by the King's House. "Here Thames Street, formerly a respectable width, narrows to a footpath in order to edge around the front of Fort Carlisle, underneath the mouths of the cannon.

There are no buildings on the harbour side of Thames Street at this point, which means that you have an unusually good view of the ships at dock, water beyond, and the Blue Mountains rising on the other side of the harbour."

South of Thames Street before Fort Carlisle is a room called Fort Carlisle. The description of Fort Carlisle is "Handsomely arrayed with cannons which you could fire at any moment -- though of course there are ships at dock which might be in the way."

Queen Street End is south of Lime Street.

Queen Street Middle is east of Queen Street End.

Queen Street East is east of Queen Street Middle and south of Private Alley.

Queen Street at the Prison is east of Queen Street East.
```

Now, if we like, we can create regions to distinguish the coast from the portions of town that aren't on the water:

```
Inland is a region. Queen Street End, Queen Street Middle, Queen Street East, Private Alley, Lime Street, and Queen Street at the Prison are in Inland.

Waterfront is a region. Thames Street before Fort Carlisle, Thames Street by the King's House, Thames Street at the Wherry Bridge, Water Lane, Fishmarket, Fisher's Row, and Thames Street End are in Waterfront.
```

There's no rule that regions must be contiguous, so we could if we like make a region consisting just of the two forts:

```
Military Holdings is a region. Fort Carlisle and Fort James are in Military Holdings.
```

And we might make the Feathers Tavern part of the Inland area, but within its own subcategory:

```
Tavern is a region. It is in Inland. Feathers and Feathers Bedroom are in Tavern.
```

Now the index map will be colored to reflect our regions, and later in the game development we would be able to make rules that affect just one region at a time.

11 ★ Example **First Name Basis** RB

Allowing the player to use different synonyms to refer to something.

Sometimes we create objects that we want the player to be able to call by different names: a television that should also answer to "tv" and "telly", for instance, or a refrigerator the player might also call "fridge". In this case, we can use instructions like

```
Understand "tv" and "telly" as the television.
```

to add extra names to the object we've defined.

```
"First Name Basis"

The Crew Lounge is a room. "Deliberately spartan: the crew feels weight restrictions here first, so there aren't any chairs, just a few thin pads on the ground."

The holographic projector is a device in the Crew Lounge. "The one major source of entertainment is the holographic projector, a top of the line Misthon 9000, on which you view every beam you can get." Understand "holo" or "holograph" or "Misthon" or "9000" as the projector.

The description of the projector is "[if switched on]The projector is now playing a documentary about the early politics of the Mars colony.[otherwise]The air above the projector is disappointingly clear.[end if]".
```

(This description is for local color; we will learn more about devices, and conditions like "if switched on", later in this chapter.)

By default, Inform does not understand the names of an object's kind as referring to that object, unless the object has no other name of its own. We can change this, if we like, by defining names that should be applied to everything of a given kind:

```
Lewis and Harper are men in the Crew Lounge. Understand "man" or "guy" or "chap" or "lad" or "male" as a man. Understand "men" or "chaps" or "lads" or "guys" or "males" as the plural of a man.

The description of Lewis is "A wiry, excitable engineer who just signed aboard last week." The description of Harper is "Harper's a good guy: taciturn when sober, affectionate when drunk, but rarely annoying in either state."

Test me with "x holo / x man / lewis / x guy / harper / turn on projector / x holo projector / get men".
```

Inform's naming abilities go considerably further, in fact: we can also instruct it to understand words only under certain circumstances, or only when they appear with other words. Fuller details may be found in the chapter on Understanding.

---

12 ★ **Example Midsummer Day** *RB*

A few sentences laying out a garden together with some things which might be found in it.

```
"Midsummer Day"

East of the Garden is the Gazebo. Above is the Treehouse. A billiards table is in
the Gazebo. On it is a trophy cup. A starting pistol is in the cup. In the Treehouse
is a container called a cardboard box.

Test me with "up / x box / d / e / x table / x cup / x pistol / get cup".
```

---

13 ★ **Example Tamed** *RB*

Examples of a container and a supporter that can be entered, as well as nested rooms.

Within a room, we might have containers and supporters that a player can enter. A chair, stool, table, dais, or pedestal would be an enterable supporter (anything we would describe a person as being "on"); a cage, hammock, or booth would be an enterable container (because we would describe the person as being "inside").

When the player is in or on something, he is able to see the rest of the contents of the room, but a note such as "(in the hammock)" or "(on the poster bed)" is added to the room title when he looks around.

Here is an example to show off the possibilities:

```
"Tamed"

The Center Ring is a room. The cage is in the Center Ring. A lion is an animal in
the cage. The cage is enterable, openable, transparent, and closed.
```

Notice that we made the cage transparent. Strictly speaking it is not made of transparent materials, but we can see into (or out of) a closed cage due to the gaps between the bars, so that from Inform's point of view a cage behaves much like a large sturdy glass box. (If we really wanted to make a distinction between, say, an airtight container and one with perforations, we could do so, but Inform does not model such nuances by default.) If a container is not transparent, we can see into and out of it only when it is open.

Supporters are a bit more straightforward because there is no circumstance in which they separate the player from the rest of the world:

```
The pedestal is in the Center Ring. It is enterable.
```

And in fact we can tell Inform that the player starts on the pedestal with this line:

```
The player is on a pedestal.
```

Now the player will begin there rather than just in the Center Ring.

This last bit is an entirely unnecessary bit of local color, but if we're going to keep getting into and out of the lion's cage, we ought to expect him to take notice:

```
Every turn when the player is in the cage:
    if a random chance of 1 in 2 succeeds, say "The lion eyes you with obvious discontent.";
    otherwise say "Though the lion does not move, you are aware that it is watching you closely."
```

Randomness is explained more completely in the chapter on Change, and every turn rules in the chapter on Time.

Finally, we might want a container whose interior is modeled as its own separate room: say, a magician's booth in which volunteers are made to disappear.

```
The magician's booth is a container in Center Ring. "Off to one side is a magician's booth, used in disappearing acts. The exterior is covered with painted gilt stars." The booth is enterable, open, not openable, and fixed in place.
```

Now we create our other location:

```
Inside from the Center Ring is the Starry Vastness.
```

...which handles the case of the player typing >IN. (We will not assume by default that he wants to get into the cage with the lion, this being obviously perilous.) But we also want to make sure that the player who types >ENTER BOOTH winds up in the same place, so we should add:

```
Instead of entering the magician's booth: try going inside.

Test me with "get in cage / open cage / get in cage / z / close cage / out / open cage / get on pedestal / get off / look / enter booth / out".
```

14 ★ **Example** **Disenchantment Bay 1** RB

A running example in this chapter, Disenchantment Bay, involves chartering a boat. This is the first step: creating the cabin.

To begin with the title:

```
"Disenchantment Bay"
```

There are many Disenchantment Bays across the world, named by eighteenth-century ships' captains - one in Antarctica, another in Tasmania, for instance. The most famous is probably the one where Lewis and Clark's expedition broke through to the Pacific. But ours is the one in Alaska, named in 1791 by a Spanish navigator who had hoped it might lead to the fabled Northwest Passage, and all of this history is beside the point since the game is set in the present day.

```
The Cabin is a room. "The front of the small cabin is entirely occupied with navigational instruments, a radar display, and radios for calling back to shore. Along each side runs a bench with faded blue vinyl cushions, which can be lifted to reveal the storage space underneath. A glass case against the wall contains several fishing rods.

Scratched windows offer a view of the surrounding bay, and there is a door south to the deck. A sign taped to one wall announces the menu of tours offered by the Yakutat Charter Boat Company."
```

We might want to start with the glass case.

```
The Cabin contains a glass case. In the glass case is a collection of fishing rods.
```

Now Inform will have guessed that the case is a container, but its default idea of a container is something like a bucket: permanently open and not able to be opened and shut. We can change that:

```
The case is closed, transparent, and openable.
```

We get a similar set of guesses if we write

```
The bench is in the cabin. On the bench are some blue vinyl cushions.
```

Using "some" rather than "a" or "the" tells Inform that the cushions are to be referred to as a plural object in the future. And because of the "on the bench..." phrase, Inform will guess that the bench is a supporter and that it is fixed in place and cannot be moved from room to room. We do have to tell it that the bench can be sat on, though:

```
The bench is enterable.
```

And now a short script, so that if we type TEST ME, we experiment with the case and bench:

```
Test me with "examine case / get rods / open case / get rods / sit on bench / take cushions / get up"
```

15 ★ **Example** **Disenchantment Bay 2** RB

Disenchantment Bay: creating some of the objects in the cabin's description.

If we compile our last version of the cabin, we get a room where the glass case and the bench are listed separately from the room description, even though they have already been mentioned once. We can prevent this by making the already-mentioned things scenery:

```
"Disenchantment Bay"

The Cabin is a room. "The front of the small cabin is entirely occupied with
navigational instruments, a radar display, and radios for calling back to shore.
Along each side runs a bench with faded blue vinyl cushions, which can be lifted
to reveal the storage space underneath. A glass case against the wall contains
several fishing rods.

Scratched windows offer a view of the surrounding bay, and there is a door
south to the deck. A sign taped to one wall announces the menu of tours offered
by the Yakutat Charter Boat Company."

The Cabin contains a glass case. In the glass case is a collection of fishing rods.
The case is closed, transparent, and openable. The case is scenery.

The bench is in the cabin. On the bench are some blue vinyl cushions. The
bench is enterable and scenery. The cushions are scenery.
```

Generally speaking, it is a good idea to recognize the player's attempts to interact with any objects mentioned in the room description, so we should also provide

```
Some navigational instruments, some scratched windows, a sign, a radar
display, and some radios are scenery in the cabin.

Test me with "examine instruments / x windows / x sign / x display / x radios".
```

The door and the view will need to be done as well, but they are special cases which we will get to shortly.

As noted, making something scenery also means that the player will be prevented from picking it up and carrying it away. This is sensible, though: if an object can be removed from the room where it first appears, we should be careful about mentioning it in the main room description; otherwise, it will continue to be described as present even when someone has carried it off.

16

★ Example **Replanting**

Changing the response when the player tries to take something that is scenery.

RB

By default, "TAKE OAK" in the example above will produce the response "That's hardly portable." This is fine under many circumstances, but also a bit generic, so we might want to override it for a specific game.

```
"Replanting"

The Orchard is a room. "Within this quadrille of pear trees, a single gnarled old oak remains as a memory of centuries past." The gnarled old oak tree is scenery in the Orchard.

Instead of taking some scenery: say "You lack the hulk-like strength."

Test me with "take oak".
```

Here we've used an "instead" rule; we will learn more about these in the section on actions. This allows us to define our own results for taking an object.

Note: "scenery" is a property of an object (about which we will hear more later). So when we use it in rules, we can talk about "some scenery", "something that is scenery", or even "a scenery thing" -- the last one doesn't sound much like English, but is a more plausible construction with other adjectives.

17 ★ Example **Disenchantment Bay 3** RB

Disenchantment Bay: adding a view of the glacier.

Suppose we wanted to have the glacier visible from the Cabin of our boat, and anywhere else we might add to the game:

```
The view of the Malaspina glacier is a backdrop. It is everywhere. The description is "The Malaspina glacier covers much of the nearby slope, and -- beyond it -- an area as large as Rhode Island."
```

18 ★ Example **Disenchantment Bay 4** RB

Disenchantment Bay: fleshing out the descriptions of things on the boat.

Currently we have provided objects for most of what is on the boat, but it's not very interesting to look at. We might want to give some more description to these things.

```
"Disenchantment Bay"

The Cabin is a room. "The front of the small cabin is entirely occupied with navigational instruments, a radar display, and radios for calling back to shore. Along each side runs a bench with faded blue vinyl cushions, which can be lifted to reveal the storage space underneath. A glass case against the wall contains several fishing rods.

Scratched windows offer a view of the surrounding bay, and there is a door south to the deck. A sign taped to one wall announces the menu of tours offered by the Yakutat Charter Boat Company."
```

```
The Cabin contains a glass case. In the glass case is a collection of fishing rods.
The case is closed, transparent, and openable. The case is scenery.

The bench is in the cabin. On the bench are some blue vinyl cushions. The
bench is enterable and scenery. The cushions are scenery.

Some navigational instruments, some scratched windows, a radar display, and
some radios are scenery in the cabin.

The description of the instruments is "Knowing what they do is the Captain's
job."

The description of the windows is "They're a bit the worse for wear, but you can
still get an impressive view of the glacier through them. There were whales
earlier, but they're gone now."

The description of the radar is "Apparently necessary to avoid the larger
icebergs."

The description of the radios is "With any luck you will not need to radio for help,
but it is reassuring that these things are here."
```

The order in which we define these things is fairly open. We could also define an object so:

```
A sign is scenery in the Cabin. The description is "You can get half-day and full-
day sight-seeing tours, and half-day and full-day fishing trips."
```

Where "the description" is assumed to refer to the thing most recently defined, if no object is specified.

```
The view of the Malaspina glacier is a backdrop. It is everywhere. The
description is "The Malaspina glacier covers much of the nearby slope, and --
beyond it -- an area as large as Rhode Island."

Test me with "examine sign / examine glacier / examine instruments / examine
windows / examine radar / examine radios / take the cushions / take the glacier".
```

These last two commands show how scenery and backdrops are automatically impossible for the player to take.

19

★★★ Example **Laura**

RB

Some general advice about creating objects with unusual or awkward names, and a discussion of the use of printed names.

Occasionally it is useful to give something a printed name because we want to call it something extremely long-winded; give one thing a name that is the subset of the name of something else; or use words such as "with" or "and" that are likely to confuse Inform into thinking that the object name ends before it actually does.

Often it is enough to preface these ambiguously-titled things with "a thing called..." or "a supporter called..." or the like, as here:

```
South of Spring Rolls is a room called Hot and Sour Soup.
```

prevents Inform from trying to read "Hot and Sour Soup" as two separate rooms, while

```
The player carries an orange ticket. The player carries a thing called an orange.
```

creates two objects instead of the one orange ticket that would result if the second sentence were merely "The player carries an orange."

Really long names can be a bit cumbersome. For example:

```
The player carries a thing called an incriminating photograph of a woman with
blonde hair.
```

So we might instead give the photograph a printed name:

```
"Laura"

The City of Angels is a room. The incriminating photograph is carried by the
player. The printed name of the incriminating photograph is "incriminating
photograph of a woman with blonde hair".
```

Now we've gotten around any awkwardness with printing the name -- but we also need to understand when the player refers to the photograph. When we define the names of objects under normal circumstances, Inform takes care of this automatically, but if we have especially set the printed name, we must also specially define the appropriate terms for the player to use. For this we need "understand", which will be explained in much more depth in a later chapter:

```
Understand "woman" or "with" or "blonde" or "hair" or "of" or "a" as the
incriminating photograph.

Test one with "x photograph / x incriminating photograph of a woman with blonde
hair / x hair / x blonde / x woman with blonde hair / x incriminating photograph of
a woman".
```

That's probably as far as we really need to go, and if you are satisfied with this behavior, there is no need to read on.

One possible objection to this solution is that Inform will accept some nonsensical formulations as applying to the photograph: for instance, it will allow >EXAMINE PHOTOGRAPH OF, >X BLONDE PHOTOGRAPH WOMAN INCRIMINATING, or even >X OF ...though in the case there were two items with "of" names, the game would disambiguate with a question such as "Which do you mean, the incriminating photograph of a woman with blonde hair or the essence of wormwood?"

Traditionally, Inform has tended to be fairly flexible about word order, preferring to err in the direction of leniency. On the other hand, there are times when we need more exacting rules in order to distinguish otherwise similar cases.

Two features allow us to specify more exactly if we so desire. The first is that, if we specify a whole phrase as the name of something, all the words in that phrase are required, in the order given. Thus "Understand "blonde hair" as the photograph" would require that both "blonde" and "hair" be present, and would not recognize >X BLONDE, >X HAIR BLONDE, or >X HAIR.

Second, we can create tokens, such as "Understand "blonde hair" or "hair" as "[hair]", and then use these tokens in match phrases. This saves a good deal of time when we want to specify a number of different but fussy alternatives. So, for instance, here is a drawing that would not respond to >X OF, or >X BROWN EYES, but would respond to >X DRAWING OF MAN WITH BROWN EYES, >X MAN WITH BROWN EYES, and so on:

```
The drawing is carried by the player. The printed name of the drawing is
"drawing of a man with brown eyes".

Understand "eyes" or "brown eyes" as "[brown eyes]". Understand "man" or
"man with [brown eyes]" or "brown-eyed man" as "[man]". Understand "[man]" or
"drawing of [man]" or "drawing of a [man]" as the drawing.

Test me with "test one / test two".

Test two with "x drawing / x man / x of / x drawing of man / x drawing of a man / x
drawing of a man with brown eyes / x drawing of a brown-eyed man / x brown
eyes".
```

Further refinements are possible: the "privately-named" attribute tells Inform not to try to understand the source name of an object at all, so if we write

```
The purple rabbit is a privately-named thing.
```

...the player will not be able to refer to it as "purple" or "rabbit" or "purple rabbit".

There are also ways to make names to refer to entire kinds of objects (so "dude" will refer to any man in the game); to specify names that only refer to objects in the plural (so GET PICTURES will pick up several pictures together); to reflect an object's properties (so "red apple" works only as long as the apple is in fact red); or even to refer to the object's relationships to other objects (so "bottle of wine" works only when wine is indeed in the bottle). All these refinements are discussed in the chapter on Understanding.

20

★ Example **Disenchantment Bay 5**

RB

Disenchantment Bay: adding the door and the deck to our charter boat.

We mentioned that there is a door out to the deck in our example. The following two sentences will create both the door and the other room:

```
The cabin door is south of the Cabin and north of the Deck. It is a door and
scenery.
```

Now Inform has constructed a generic room called "Deck" to the south. It has neither a description nor any contents yet, but we could fix that in time. It does have a view of the glacier, though, since we defined the glacier view to be everywhere.

21

★★★ Example **Escape**

Window that can be climbed through or looked through.

RB

Suppose we want to offer the player a window he can climb through, instead of a boring ordinary door. Our window will be like a door in that it connects two rooms, appears in both places, and impedes movement when it is shut. But we also want to add that we can look through it and see what lies on the other side; and we further want to understand "climb through window" or "jump through window" as attempts to pass through it.

We'll start by defining a couple of rooms and making the window a door between them.

```
"Escape"

Your Bedroom is a room. The bedroom window is a door. It is west of Your
Bedroom and east of the Grassy Slope.
```

Now we have a "bedroom window" object which can be entered. Now, to catch the case where the player types "LOOK THROUGH WINDOW":

```
Instead of searching the window:
    say "Through the window, you make out [the other side of the window]."
```

The other side of a door is always defined to be the room that we are not currently in when doing the check. When we are in the bedrooom, the other side will be the grassy slope, and vice versa. "Searching" is the action that occurs when the player attempts to LOOK THROUGH something. (To review what grammar gives rise to what actions, we can always consult the Actions portion of the Index.)

Next we want to cover the case where we climb through the window:

```
Instead of climbing the window:
    try entering the window.
```

And because "climb window" is understood but "climb THROUGH window" is not, we will have to borrow from the chapter on Understanding to add some new vocabulary to the game (and we'll add Jump too, while we're at it):

```
Understand "climb through [something]" as climbing. Understand "jump through
[something]" as climbing.
```

Now the final piece: Inform will already keep the player from going through a closed window, but it will say "You can't, since the bedroom window is in the way." This is probably not ideal, so we can replace the instruction thus:

```
Instead of going through the closed window:
    say "The window is shut: you'd break the glass."

Test me with "look through window / climb through window / open window / climb through window / look through window / close window / e / open window / e".
```

22 ★★★★

Example Garibaldi 1

RB

Providing a security readout device by which the player can check on the status of all doors in the game.

Suppose we would like to allow the player to view the status of all the doors functioning in the game; and we want to identify those doors by mentioning which two rooms they connect. The following uses some techniques that will be covered in later chapters, but the basic idea may be obvious:

```
"Garibaldi"

The security readout is a device. The description of the readout is "The screen is blank."

Instead of examining the switched on security readout:
    say "The screen reads: [fixed letter spacing]";
    say line break;
    repeat with item running through doors:
        say line break;
        say " [item] ([front side of the item]/[back side of the item]): [if the item is locked]LOCKED[otherwise]UNLOCKED[end if]";
    say variable letter spacing;
    say paragraph break.
```

It is more or less arbitrary which room winds up as the "front side" and which as the "back", but in this case it hardly matters.

```
The player carries the security readout.

The Docking Bay is a room. The inner airlock is a door. It is north of the Docking Bay and south of the Zocalo. The inner airlock is lockable and unlocked. The outer airlock is lockable and locked. It is a door. It is south of the Docking Bay and north of Space.

The quarantine seal is a door. It is west of the Zocalo and east of Medlab. Quarantine seal is locked.

The security pass unlocks the inner airlock. The player carries the security pass.
```

> Test me with "x readout / turn on readout / x readout / lock inner airlock with security pass / x readout".

23 ★ Example **Disenchantment Bay 6** RB

Disenchantment Bay: locking up the charter boat's fishing rods.

It stands to reason that the captain wouldn't let just anyone meddle with his fishing equipment; maybe he keeps that case locked. We could replace the case description with this one, instead:

> The Cabin contains a glass case. In the glass case is a collection of fishing rods. The case is closed, transparent, openable, lockable, and locked. The case is scenery. The small silver key unlocks the case.

Now there's a silver key that will unlock it -- though since we haven't said where the key is, the player will never be able to find it in the game. (If we look at the World index, we find "small silver key" right at the bottom, not inside any of the rooms. That is as good as not existing at all -- though we usually use the term "out of play" -- but as we will later see, it is possible to have things initially out of play but brought into existence later on.)

24 ★★★ Example **Neighborhood Watch** RB

A locked door that can be locked or unlocked without a key from one side, but not from the other.

Suppose we want a locked door that can be opened with a key, but is also openable by hand without a key from one side only. We start by defining an ordinary lockable door and the key that controls it:

> "Neighborhood Watch"
>
> The shabby door is a door. It is outside from the Studio Apartment and inside from the Rickety Stairwell. The shabby door is locked.
>
> The brass key is carried by the player. It unlocks the shabby door.

The next part is going to require that we modify the normal operation of the "lock" command. "Lock" ordinarily requires that the player supply two objects: a thing he wants to unlock, and the key he wants to use on it. The full command is LOCK DOOR WITH THE KEY, and Inform will not accept simply LOCK DOOR as locking.

Therefore, we're going to need to create our own new variant on the lock verb (and the unlock verb, while we're at it). The full procedure for this is laid out in the chapters on Action and Understanding, but here is an example:

```
Understand "lock [something]" as locking keylessly. Locking keylessly is an
action applying to one thing.
```

Here we've created a new action -- locking something without a key -- and we've told Inform to understand LOCK DOOR as this action, rather than an incomplete command to LOCK DOOR WITH SOMETHING.

Now we add some instructions so that the game will not let us use this keyless unlocking command unless we're in the right place or are properly equipped:

```
Check locking keylessly:
   if the noun is not a door, say "[The noun] is not something you can lock."
instead;
   if the noun is locked, say "[The noun] is already locked." instead;
   if the player carries the brass key and the player is in the Stairwell, try locking
the noun with the brass key instead;
   if the player is in the Stairwell, say "You can't lock the door from this side
without the key." instead.
```

This check rule is performed before the keyless locking action succeeds. The first thing we do is try to use the key if the player is outside and has the key: this way, LOCK DOOR will turn automatically into LOCK DOOR WITH THE KEY, under circumstances where that is both possible and necessary.

The second thing is to check whether the player is outside but keyless, and, if so stop the action from being performed successfully. Here we print a failure message followed by the word "instead", which tells Inform that we've substituted some other outcome for the usual performance of the action.

Now we're reasonably sure that the player is only locking keylessly in the case that he is inside the Studio. (We might have to do a more thorough check for this if there were more than two rooms, but as it is, the player can only be in the Stairwell or in the Studio, so if we have ruled out the Stairwell, we are safe.) So now we want to add what happens when locking-without-a-key command succeeds:

```
Carry out locking keylessly:
   now the noun is locked.
```

That's it. We've just told Inform to make the door be locked. "Now..." syntax will be explained more thoroughly in the chapter on change. But we still haven't described to the player what just happened, so let's provide a description of that, too:

```
Report locking keylessly:
   say "You flip over the deadbolt to lock [the noun]."
```

And now we have to do a similar set of things for unlocking:

```
Understand "unlock [something]" as unlocking keylessly. Unlocking keylessly is
an action applying to one thing.

Check unlocking keylessly:
   if the noun is not a door, say "[The noun] is not something you can lock."
instead;
```

```
    if the noun is unlocked, say "[The noun] is already unlocked." instead;
    if the player carries the brass key and the player is in the Stairwell, try unlocking the noun with the brass key instead;
    if the player is in the Stairwell, say "You can't unlock the door from this side without the key." instead.

Carry out unlocking keylessly:
    now the noun is unlocked.

Report unlocking keylessly:
    say "You flip over the deadbolt to unlock [the noun]."

Test me with "unlock door / drop key / open door / out / close door / lock door / open door / in / get key / out / close door / lock door / unlock door".
```

Some (but not all) of this work is done for you if you like by the Locksmith extension. If you prefer, you can include that extension, then follow the documentation in order to implement the remainder of the scenario. Locksmith takes care of implementing the additional locking and unlocking actions, and provides some other conveniences.

25 ★ Example **Disenchantment Bay 7** RB

Disenchantment Bay: making the radar and instruments switch on and off.

If we would like the player to be able to turn instrumentation on and off, we could add a line to this effect:

```
The radar, the instruments, and the radios are devices.
```

And since the captain is probably not navigating blind, we might also want to say

```
The radar and the instruments are switched on.
```

26 ★★★ Example **Down Below** RB

A light switch which makes the room it is in dark or light.

Suppose we want to have a room with a light switch. Turning the switch off makes the room go dark; turning it on restores the light. This kind of switch is an obvious candidate as a device.

```
"Down Below"

Terrifying Basement is a room. The light switch is a switched on device in the Terrifying Basement. It is fixed in place.
```

Here we define our light switch, and we also make it start out as "switched on". The Terrifying Basement will also start out lit (as all rooms do, by default, unless we specifically say that they are dark). We further say that it is fixed in place to avoid the ludicrous possibility of the player picking it up and carrying it away.

Next we add some instructions to control how turning the light switch on and off affects the room light. These borrow from later chapters on actions, but the gist may be obvious anyway:

```
Carry out switching off the light switch: now the Terrifying Basement is dark.

Carry out switching on the light switch: now the Terrifying Basement is lighted.
```

Inform already has the idea of light and darkness built in; we will see more about this later, and the Phrasebook (in the Index tab) also contains a list of all the adjectives (lighted, dark, etc) which are important to use here.

Speaking of the Index, the Actions tab contains a list of all the grammar that can be used to activate a given command: for instance, the switching action responds to "switch [something]" or "turn on [something]". In this case, we may want to give the player an extra option or two. It would be pretty natural for a player to try >FLIP SWITCH, so let's add that in:

```
Understand "flip [something switched off]" as switching on. Understand "flip
[something switched on]" as switching off. Understand "flip [something]" as
switching on.
```

The nuances of this will be explored in the chapter on Understanding. What is useful to know here is that we have taught Inform to understand that >FLIP LIGHT SWITCH means to turn it on when the switch is already off; if the switch is already on, FLIP SWITCH means to turn the switch off. Depending on the kind of device we are modeling (button? lever? dial?), we might want to write similar lines for commands such as PUSH, PRESS, PULL, TURN, and so on.

Finally, we need to deal with a special case. In general, the player cannot interact with other things in a dark room because he can't see them, but if we adhered strictly to this it would be impossible for him to find the light switch to turn it back on. So we need something from the chapter on Activities to change this:

```
After deciding the scope of the player when the location is the Terrifying
Basement:
   place the light switch in scope.

Upstairs is above the Terrifying Basement.

Test me with "turn off light / look / flip light switch".
```

---

27 ★ **Example Peugeot** RB

A journey from one room to another that requires the player to be on a vehicle.

Let's say that our protagonist is about to flee . Obviously, he can't make the journey on foot; he needs transportation.

```
"Peugeot"

Include Rideable Vehicles by Graham Nelson.

The Lot is a room. The ten-speed bike is a rideable vehicle in the Lot.
```

We make the ten-speed bike a rideable vehicle because we want to say that the player is on it rather than in it. Then our other room:

```
Cambridge is east of the Lot.
```

And now we borrow from the Actions chapter to prevent travel without the proper equipment:

```
Instead of going to Cambridge when the player is not on the ten-speed bike:
    say "It's a long journey to Cambridge: you'll never make it on foot."

After going to Cambridge:
    say "You begin pedalling determinedly.";
    continue the action.

Test me with "e / get on ten-speed bike / e".
```

28

★★ Example **Disenchantment Bay 8**

Disenchantment Bay: a pushable chest of ice for the boat.

RB

We probably do not need a vehicle to ride around our boat, but there might be a heavy ice chest that can only be pushed from room to room:

```
The ice chest is a closed openable container in the Deck. "A very heavy ice
chest sits on the ground." It is fixed in place and pushable between rooms. A
quantity of ice is in the chest. The description is "Ready and waiting just in case
there's any fish needing to be kept cool."
```

This anticipates a later chapter, but it would probably be a good idea to hint to the player, if he tries to take the ice chest, that there is another way to move it:

```
Instead of taking the chest: say "It's too heavy to lift, but you might be able to
push it, and just inch it over the frame of the door."
```

Otherwise, attempts to pick it up will just reply with "That's fixed in place."

29

Example Hover

RB

Letting the player see a modified room description when he's viewing the place from inside a vehicle.

Suppose we want the player to see a modified room description when he's viewing the place from inside a vehicle. There are several conceivable ways of doing this; the example here shows a rather advanced way, but is very flexible and will let us write all sorts of special cases.

```
"Hover"

Use full-length room descriptions.

Emerald City is a room. "All the buildings are spires and none of them have doors." The Vast Desert is west of Emerald City. "[if the player is in a vehicle]Outside, a[otherwise]A[end if] trackless waste stretches as far as the eye can see in every direction."

The hover-bubble is a vehicle in the Emerald City. "Your hover-bubble awaits." The description is "The hover-bubble is a clear globe-shaped vehicle capable of transporting you anywhere you could walk, but faster." Understand "bubble" as the hover-bubble. The hover-bubble contains a chocolate wrapper and a parking ticket.
```

Here's the tricky part, which relies on material from the chapters on Activities and Rulebooks:

```
The container interior rule is listed before the room description body text rule in the carry out looking rules.

This is the container interior rule:
	if the actor is the player and the player is in an enterable thing (called current cage), carry out the describing the interior activity with the current cage.

Describing the interior of something is an activity.
```

Now we've done that, we can write a "rule for describing the interior" of something, which will print whatever we like:

```
Rule for describing the interior of the hover-bubble:
	say "The hover-bubble is transparent, but tints everything outside very faintly lavender."
```

In fact, as a special refinement, we could even say:

```
Rule for describing the interior of the hover-bubble when the hover-bubble contains more than one thing:
	say "The hover-bubble is transparent, but tints everything outside very faintly lavender. Beside you you can see [a list of other things in the hover-bubble]."

Definition: a thing is other if it is not the player.
```

```
Rule for listing nondescript items of the hover-bubble when the player is in the hover-bubble: do nothing.

Test me with "get in bubble / look / west / take all / look / get out / east".
```

And now anything that's beside us in the vehicle will be described during that first paragraph, rather than later on.

30 ★ Example **Disenchantment Bay 9** RB

Disenchantment Bay: enter the charter boat's Captain.

Now finally we can put a Captain in the boat:

```
The Captain is a man in the Cabin. "The captain sits at the wheel, steering the boat and occasionally checking the radar readout."
```

31 ★ Example **Belfry** RB

You can see a bat, a bell, some woodworm, William Snelson, the sexton's wife, a bellringer and your local vicar here.

```
"Belfry"

The Belfry is a room. A bat is in the Belfry. The bell is in the Belfry. Some woodworm are in the Belfry. A man called William Snelson is in the Belfry. A woman called the sexton's wife is in the Belfry. A man called a bellringer is in the Belfry.

In the Belfry is a man called the vicar. The indefinite article of the vicar is "your local".

Test me with "look".
```

32 ★★ Example **Gopher-wood** RB

Changing the name of a character in the middle of play, removing the article.

Suppose that we want a character who starts out with a general epithet ("the bearded man") but is later introduced to the player properly ("Japheth"). In that case, we want to be able to tell Inform to stop using an article once the character has been given a proper name. We can do this like so:

```
"Gopher-wood"
```

```
The Ark is a room. A bearded man is in the Ark.

Instead of examining the bearded man for the first time:
    now the printed name of the bearded man is "Japheth";
    now the bearded man is proper-named;
    say "You peer at him a bit more closely and realize that it's Japheth."
```

Finally, we need to tell Inform to understand the man's name, but only when he's been introduced. For this purpose, we borrow from the chapter on Understanding:

```
Understand "Japheth" as the bearded man when the bearded man is proper-named.

Test me with "x japheth / x man / look / x japheth".
```

33 ★ Example **Disenchantment Bay 10** RB

Disenchantment Bay: things for the player and the characters to wear and carry.

At this point we can dress both the Captain and the player with some appropriate props:

```
The captain wears a baseball cap. The description of the cap is "It says, THE WORST DAY FISHING IS BETTER THAN THE BEST DAY WORKING."

The player is carrying a backpack and a bottle of water. The player is wearing a pair of sunglasses. The description of the sunglasses is "The light off the water and the ice does get pretty bright sometimes."
```

(At present the backpack can't be worn, but see the next version.)

34 ★ Example **Disenchantment Bay 11** RB

Disenchantment Bay: making a holdall of the backpack.

If we wanted, we could make the player's backpack infinitely capacious, so:

```
The backpack is a player's holdall.
```

...And now whenever the player character is unable to hold everything, he will automatically stow some of his possessions therein.

This is only useful if the player doesn't have infinite carrying capacity himself, so perhaps we also need

```
The carrying capacity of the player is 3.
```

Perhaps mercifully, items which are worn are not counted against the player's carrying capacity. We might want to let him take advantage of that, too:

```
The backpack is wearable.
```

This capacity system makes a compromise between the realistic and the absurd: on the one hand, it acknowledges that people can't carry an infinite number of items in their hands, while at the same time providing a sack that can.

Many games will have no use for object-juggling of this kind at all; others will want to be much more rigorous about questions of capacity and volume. Fortunately, it is easy to leave the whole business out by assigning no carrying capacity to anything.

---

35 ★★★ **Example** **Fallout Enclosure** *RB*

Adding an enclosure kind that includes both containers and supporters in order to simplify text that would apply to both.

It may not be immediately obvious why we might want to create new intermediate categories of the kinds hierarchy. But there may be times, for instance, where we would like to make an action that applies in the same way to both containers and supporters, but to nothing else in the game. To avoid creating two nearly-identical rules, we would instead roll the two categories together into one, on the principle that duplicating source text is usually a sign of bad design.

So for instance let's say the player is able to zap objects to make them go away, but any contents -- things inside a container or on top of a supporter -- should always be left as residue. Here's one way we might do this:

```
"Fallout Enclosure"

Section 1 - Procedure

An enclosure is a kind of thing. A container is a kind of enclosure. A supporter is
a kind of enclosure.

Understand "zap [something]" as zapping. Zapping is an action applying to one
thing. The Zapping action has a list of things called the remnants.

Carry out zapping an enclosure:
    if the noun holds something:
        now the remnants is the list of things held by the noun;
        repeat with N running through the remnants:
            move N to the holder of the noun.

Carry out zapping:
    now the noun is nowhere.

Report zapping:
    say "You zap [the noun], destroying [them][if the remnants is not empty] and
```

```
leaving [the remnants with indefinite articles] behind[end if]."

Section 2 - Scenario

SuperDuperMart is a room. SuperDuperMart contains some shelves and a cash register.

The shelves support a bottle of Buffout and a container of Jet.

The cash register contains some prewar money, a coin purse, and a bottle cap. The coin purse contains a prewar nickel. It is closed.

The cash register is closed and locked.

Test me with "zap shelves / zap buffout / zap register / zap purse".
```

36 ★★★ **Example Brown** RB

A red sticky label which can be attached to anything in the game, or removed again.

```
"Brown"

The Shipping Room is a room. The red sticky label is a thing carried by the player. The description of the red sticky label is "It reads: AIRMAIL[if the label is part of something (called the parent)]. It is stuck to [the parent][end if]."

A black crate is in Shipping. The description is "A boring black crate." The brown crate is a thing in Shipping. The description is "An ordinary brown crate."

After examining something when the label is part of the noun:
    say "A bright red sticky label is attached to [the noun]!"
```

Here is the essential point: whenever we ATTACH LABEL TO something, it becomes part of that object.

```
Instead of tying the red sticky label to something:
    now the red sticky label is part of the second noun;
    say "You stick [the label] to [the second noun]."
```

And of course the label cannot be stuck to itself or to more than one thing at a time.

```
Before tying the label to something when the label is part of something:
    if the label is part of the second noun:
        say "[The label] is already stuck to [the second noun]." instead;
    otherwise:
        say "(first freeing the label)[line break]";
        silently try taking the label;
        if the label is part of something, stop the action.
```

```
Instead of tying the red sticky label to the label:
    say "That would ruin the label entirely."

Instead of taking the label when the label is part of something:
    now the player carries the label;
    say "You peel the label off again."
```

Much of the rest is just tidying to make sure that the player's commands are redirected into the right syntax.

```
Instead of tying something to the label:
    try tying the label to the noun.

Instead of putting the label on something:
    try tying the label to the second noun.

Instead of inserting the label into something:
    try tying the label to the second noun.

Understand the commands "stick" or "apply" as "tie".
```

We could have created a new "sticking" action, but to keep the example short we will use the built-in "tying" action instead, and respond to the command "stick" just as if it were "tie".

```
Understand "peel [something]" or "peel off [something]" as taking.

Test me with "i / put label on the black crate / look / x black / x label / get the
label / apply label to brown crate / look / x brown / peel off label / stick label to
label".
```

37 ★★★★ Example **Disenchantment Bay 12** RB

A final trip to Disenchantment Bay: the scenario turned into a somewhat fuller scene, with various features that have not yet been explained.

```
"Disenchantment Bay"

Include Locksmith by Emily Short.

Use scoring.

The Cabin is a room. "The front of the small cabin is entirely occupied with
navigational instruments, a radar display, and radios for calling back to shore.
Along each side runs a bench with faded blue vinyl cushions[if the compartment
is closed], which can be lifted to reveal the storage space underneath[otherwise],
one of which is currently lifted to allow access to the storage compartment
within[end if]. A glass case against the wall contains several fishing rods.
```

Scratched windows offer a view of the surrounding bay, and there is a door south to the deck. A sign taped to one wall announces the menu of tours offered by the Yakutat Charter Boat Company."

The Cabin contains a glass case. In the glass case is a collection of fishing rods. Understand "rod" as the collection. The case is closed, transparent, openable, lockable, and locked. The case is scenery. The small silver key unlocks the case.

The bench is in the cabin. On the bench are some blue vinyl cushions. The bench is enterable and scenery. The cushions are scenery.

A storage compartment is an openable closed container. It is part of the bench. Instead of opening the bench, try opening the storage compartment. Instead of closing the bench, try closing the storage compartment. Instead of pushing or pulling or turning the cushions, try opening the storage compartment. Understand "space" as the storage compartment.

Some nets and a Coke are in the compartment. Understand "net" as the nets. The description of the nets is "They must have something to do with fish as well. Really, you're just here for the sights." The nets are a container.

Some navigational instruments, some scratched windows, a radar display, and some radios are scenery in the cabin. The radar, the instruments, and the radios are devices. The radar and the instruments are switched on.

A screen is part of the radar. The description of the screen is "[if the radar is switched on]Phantom lights move across the screen.[otherwise]The screen is dark.[end if]". Instead of doing something other than examining to the screen, say "It's not good for much but looking at."

The Captain is a man in the Cabin. "The captain sits at the wheel, steering the boat and occasionally checking the radar readout." The captain wears a baseball cap. The description of the cap is "It says, THE WORST DAY FISHING IS BETTER THAN THE BEST DAY WORKING." The captain carries the silver key. The description of the captain is "[The captain] is wearing [a list of things worn by the captain][if the captain carries something] and carrying [a list of things carried by the captain][end if]."

The description of the instruments is "Knowing what they do is the Captain's job." Instead of doing something other than examining to the instruments in the presence of the Captain: say "The Captain glares at you. Clearly you are not welcome to do that."

The description of the windows is "They're a bit the worse for wear, but you can still get an impressive view of the glacier through them. There were whales earlier, but they're gone now." Understand "window" as the windows.

The description of the radar is "Apparently necessary to avoid the larger icebergs."

The description of the radios is "With any luck you will not need to radio for help, but it is reassuring that these things are here."

```
A sign is scenery in the Cabin. The description is "You can get half-day and full-day sight-seeing tours, and half-day and full-day fishing trips."

The view of the Malaspina glacier is a backdrop. It is everywhere. The description is "The Malaspina glacier covers much of the nearby slope, and -- beyond it -- an area as large as Rhode Island." Understand "view of the surrounding bay" or "surrounding bay" as the view.

The cabin door is south of the Cabin and north of the Deck. It is a door and scenery. The description of the Deck is "The whole back half of the boat is open, allowing you to view the surroundings without intervening windows -- if you can stand the cold."

The ice chest is a closed openable container in the Deck. "A very heavy ice chest sits on the ground." It is fixed in place and pushable between rooms.

A quantity of ice is in the Deck. "All around the boat bob chunks of glacier ice." Understand "glacier ice" as the quantity. The description is "Curiously cooled into funny-shaped chunks." The printed name of the quantity is "glacier ice".

Instead of taking the quantity of ice when the player is not carrying the nets:
    if the quantity of ice is handled, continue the action;
    say "You are having a hard time fishing out the ice with your bare hands."

Instead of taking the quantity of ice when the player is carrying the nets:
    if the quantity of ice is handled or the quantity of ice is in the nets, continue the action;
    now the quantity of ice is in the nets;
    say "You scoop up the ice with the net."

Instead of taking the chest: say "It's too heavy to lift, but you might be able to push it, and just inch it over the frame of the door."

The player is carrying a backpack. The player is wearing a pair of sunglasses. The description of the sunglasses is "The light off the water and the ice does get pretty bright sometimes."

The backpack is a player's holdall. The carrying capacity of the player is 3. The backpack is wearable.

Instead of asking the Captain for the key:
    say "'Sure, you can -- well, get me a drink first, would you?'"

Instead of asking the Captain for the key when the Captain is carrying a cold Coke and the Captain is carrying the key:
    move the key to the player;
    say "'Here, knock yourself out.'"

Instead of asking the Captain for the key when the Captain is not carrying the key: say "'I already gave it to you. You didn't lose it, did you?'"

Heat is a kind of value. The heats are cold, cool, room temperature, and warm.
```

```
A beverage is a kind of thing. A beverage can be open or closed. A beverage can be openable or unopenable. A beverage is always edible and openable. A beverage has a heat. A beverage is usually warm. The Coke is a beverage. The beer is a beverage. The beer is in the backpack.

Instead of giving or showing a beer to the Captain:
    say "'I don't drink on the job, thanks,' he says. 'You can help yourself if you want it, though.'"

Instead of giving or showing a Coke to the Captain:
    say "'It needs chilling,' the Captain remarks, disgruntled."

Instead of giving or showing a cold Coke to the Captain:
    move the Coke to the Captain;
    increase the score by 2;
    say "'Ah, thank you,' he says. How he can drink an iced soda on a day like today is an open question, but Alaskans are special."

Every turn when the quantity of ice is in the ice chest:
    repeat with item running through beverages in the ice chest:
        let the current heat be the heat of the item;
        if the current heat is not cold, now the heat of the item is the heat before the current heat.

Before printing the name of a beverage (called the drink):
    say "[heat of the drink] ".

Understand the heat property as describing a beverage.

The maximum score is 5.

After taking the fishing rods:
    end the story finally;
    increase the score by 3;
    say "Success is yours! (Now if only you knew anything about fishing.)"

Test me with "test first / test second / test third".

Test first with "x captain / open case / i / ask captain for the key / give beer to captain / open bench / x nets / get nets / get coke / give coke to captain".

Test second with "s / open chest / drop nets / get glacier ice / get nets / get glacier ice / g / put glacier ice in chest / get coke / put coke in chest".

Test third with "x coke / g / g / g / get coke / n / give coke to captain / ask captain for key / open case / get rods".
```

38 ★★★ Example Search and Seizure

A smuggler who has items, some of which are hidden.

RB

Suppose we want a character who carries hidden objects, but only while he is wearing his jacket. If we deprive him of this, his other possessions become known. Furthermore, if we ever search him, his possessions also become known, and can thereafter be mentioned by us.

```
"Search and Seizure"

Size is a kind of value. The sizes are small, medium, and large. A thing has a size. A thing is usually small.

A thing can be licit or contraband. A thing is usually licit.

A thing can be discovered or secret. A thing is usually secret.
```

Now, we want the character to be able to hide small things if he has some sort of concealing garment on. We also want to be able to see anything that the player has already found once, perhaps by using the >SEARCH PERSON command. So:

```
Rule for deciding the concealed possessions of someone (called the suspect):
    if the particular possession is discovered, no;
    if the suspect wears something and the particular possession is small, yes;
    otherwise no.
```

The following rule, borrowed from a later chapter, assures that any items that are ever mentioned to the player will be treated as known from then on:

```
Before printing the name of something (called discovery): now the discovery is discovered.

The Customs House is a room. The smuggler is a man in the Customs House. The smuggler wears a leather jacket. He carries a bottle of Altairan rum, some raspberries, and a laser pistol. The pistol is large. The jacket is large. The rum and the raspberries are contraband. The description of the smuggler is "He has a bestubbled chin and a sinister eye."

The tourist is a woman in the customs house. The description is "The type who walks off home with a dozen contraband items in her pocket not because she means to steal things but because she's too stupid to understand that the law applies to herself." She wears a tight-fitting dress. The dress is large. The tourist carries a grapevine and an archaeological artifact. The grapevine is large and contraband. The artifact is contraband.

Report examining someone: say "[The noun] is [if the noun is wearing something]wearing [a list of unconcealed things worn by the noun] and [end if]carrying [a list of unconcealed things carried by the noun]."

Report examining someone who is concealing something contraband:
    say "[The noun] looks nervous. You can just tell."

Report examining someone who is carrying an unconcealed contraband thing:
    say "Your eye goes at once to [the list of unconcealed contraband things carried by the noun] which [the noun] is unable to hide";
    if an unconcealed licit thing is had by the noun, say ", though [if the noun is
female]s[end if]he also has [a list of unconcealed licit things had by the noun]";
    say "." instead.
```

Notice that we can talk about what the smuggler wears, what he carries, and what he "has": things the smuggler has can be either worn or carried, so the phrase is useful if we don't care to make that distinction.

```
Instead of searching someone:
    say "[The noun] is revealed to be carrying [a list of things carried by the
noun]."

Instead of confiscating the dress:
    say "You are not allowed to perform strip-searches in the public customs
area."

Understand "confiscate [something]" as confiscating.

Confiscating is an action applying to one thing.

Check confiscating:
    unless the noun is had by someone who is not the player:
        say "You can only confiscate other people's possessions." instead.

Carry out confiscating:
    now the noun is carried by the player.

Report confiscating:
    say "Through the authority vested in you by the power of the Sovereign of
Centauri Proxima, you make [the noun] your own."

Test me with "x smuggler / search smuggler / x smuggler / confiscate jacket / x
smuggler / confiscate rum / confiscate pistol / x smuggler / confiscate raspberries
/ x smuggler".

Test more with "x tourist / confiscate dress / confiscate grapevine / x tourist /
search tourist / confiscate artifact / x tourist".
```

39

★★★ Example **Van Helsing**

RB

A character who approaches the player, then follows him from room to room.

Suppose we want to write a character who tries to be in the same room as the player. We will do this by testing every turn whether the character's location and the player's location are the same; if the answer is no, the character will look for a path to the player's location, then try to move along that path. (We will learn more about finding paths and giving characters instructions later.)

The result will be that if the player ever moves to another location, the character will automatically pursue him.

```
"Van Helsing"

The Drawbridge is a room. North of the Drawbridge is the Immensely Enormous Entry Hall. West of the Entry Hall is the Vast Dining Area. North of the Vast Dining Area is the Colossal Kitchen. The Spooky Guano-filled Attic is above the Entry Hall.

Count Dracula is a man in the Attic.
```

In the following condition, we could also have written "if the location of Count Dracula is not the location", because "location" by itself is always understood to be the player's location. But it seemed better for clarity to write it this way:

```
Every turn:
    if the location of Count Dracula is not the location of the player:
        let the way be the best route from the location of Count Dracula to the location of the player, using doors;
        try Count Dracula going the way;
    otherwise:
        say "'Muhahaha,' says Count Dracula."

Test me with "z / z / n / w / e / u / z / d".
```

40 ★★ **Example** **Prisoner's Dilemma** *RB*

A button that causes a previously non-existent exit to come into being.

We can change the directions in the map in mid-game, though in practice this is rarely necessary. But suppose we do not want a door or any sign of a door to exist before the player takes some action, in this case pressing a button:

```
"Prisoner's Dilemma"

Challenger's Waiting Room is a room. "The challenge is this: to wait as long as you can endure to do so in a room with no features and no clock. If you wait longer than all the other contestants, you win."

The button is fixed in place in the Challenger's Waiting Room. "The only item in view is a black recessed button."

Amid the Cheering Throng is a room.

Instead of pushing the button for the first time:
    change the east exit of the Challenger's Waiting Room to Amid the Cheering Throng;
    change the west exit of the Cheering Throng to the Challenger's Waiting Room;
    say "With a groan of gears, the east wall swings open! If you've lost now, well, you've lost..."

Test me with "e / push button / e / w".
```

Our instructions about pushing the button will be further explained in the chapter on Actions, but the thing to note here is that we can "change (whatever) exit" in order to set or re-set map directions. Notice that we have to set both directions explicitly: changing the east exit of the Waiting Room does not automatically also change the west exit of Amid the Cheering Throng.

This allows greater flexibility in our games but does require an extra line or so of work.

---

41 ★★ **Example** **The World of Charles S. Roberts** *RB*

Replacing the ordinary compass bearings with a set of six directions to impose a hexagonal rather than square grid on the landscape.

Wargaming is an ancient pursuit, but its modern form began as a professional training exercise in 19th-century Prussian staff colleges; since at least as early as H. G. Wells's "Little Wars" (1913) it has been a hobby of "boys from twelve years of age to one hundred and fifty and for that more intelligent sort of girl who likes boys' games and books." The free-form tabletop game used miniature figures and tape-measured movements, and remains the dominant form today. But in the mid-20th century, map grids on printed sheets gave the hobby a sudden new lease of life. They were easier to set up, more interesting to look at, cheaper to sell by mail-order. 1970s sales figures for "Strategy and Tactics", the leading US subscription-based wargame distributor, were very similar to those of Infocom's IF games in the 1980s. And like classical IF, the grid-based wargame parceled up a continuous world into locations.

Grids were initially square, as on a chessboard, but square cells have several disadvantages. Four directions of movement (N, E, S, W) is too few, yet allowing movement in the diagonal directions means allowing tanks to travel about 1.4 times faster northeast than they do north. Square grids also only conform cleanly to man-made landscape features such as buildings in one orientation, and they never fit hills well. (A compromise measure to fix this, cutting the squares into octagons to leave smaller diamond squares at corner intersections, has never caught on.)

But following Charles S. Roberts's American Civil War designs for Avalon Hill of 1958-61 (notably "Chancellorsville" and the second edition of "Gettysburg"), a hexagonal grid became the new standard. Each hexagon is the same distance from the centre of all six of its neighbours, which are at equal angular spacings; and clumps of hexagons fit the shape of lakes, contoured hills, and so forth, much more naturally than clumps of squares do. Hexes also have a certain mystique - an air of "I don't belong in the children's department".

But hexes are tricky for IF, not least because English lacks words for "the direction 60 degrees around from front". Our cognitive view of the world tends to be square, perhaps because our two eyes both face front, in a direction at right angles to the plane of our arms, legs, pelvis and eyes. We reach out sideways at right angles to our walking. Even early hex-grid wargames called the cells "squares", though "hexes" eventually caught on. Still and all:

```
"The World of Charles S. Roberts"

Forward is a direction. Forward has opposite backward. Understand "f" as
forward.
Backward is a direction. Backward has opposite forward. Understand "b" and
"back" as backward.
Forward left is a direction. Forward left has opposite backward right. Understand
"fl" as forward left.
Forward right is a direction. Forward right has opposite backward left.
Understand "fr" as forward right.
Backward left is a direction. Backward left has opposite forward right.
Understand "bl" as backward left.
Backward right is a direction. Backward right has opposite forward left.
Understand "br" as backward right.
```

Now to forbid the use of the compass directions:

> A direction can be hexagonal or squared-off. A direction is usually squared-off. Forward, backward, forward left, forward right, backward left and backward right are hexagonal.
>
> Before going a squared-off direction, say "In this hexagonally-divided landscape, squared-off directions are not allowed." instead.

A slight nuisance is that, with things as they are above, typing BACKWARD produces the response "Which do you mean, backward, backward left or backward right?" To avoid that silly question, we write:

> Does the player mean going backward: it is very likely. Does the player mean going forward: it is very likely.

And now a clump of 37 hexes, in six columns of six or seven rooms each. There are many ingenious ways we could put this map together automatically, but instead we will take a deep breath and write:

> E1 is forward of E2. "Open farmland." E2 is forward of E3. "The edge of woods." E3 is forward of E4. "Deep woodland." E4 is forward of E5. "Deep woodland." E5 is forward of E6. "The rear edge of woods." E6 is forward of E7. "The start of a road leading forward right." E7 is a room. "Grassland."
>
> F1 is forward of F2. "The edge of farmland." F2 is forward of F3. "The edge of woods." F3 is forward of F4. "Clearing in woods." F4 is forward of F5. "Deep woodland." F5 is forward of F6. "A road runs backward left to forward right." F6 is a room. "The edge of grassland."
>
> G1 is forward of G2. "Grassland." G2 is forward of G3. "The edge of farmland." G3 is forward of G4. "A copse of trees." G4 is forward of G5. "The backward edge of woodland." G5 is forward of G6. "A bend in the road, from backward left to backward right." G6 is forward of G7. "Open farmland." G7 is a room. "Open farmland."
>
> H1 is forward of H2. "Grassland, bordered by a hedge to the right." H2 is forward of H3. "The edge of farmland, with a hedge to forward right." H3 is forward of H4. "A copse of trees." H4 is forward of H5. "Open farmland." H5 is forward of H6. "A passing place on the road, which bends forward left to forward right." H6 is a room. "Open farmland."
>
> I1 is forward of I2. "The end of a forward road, blocked by hedges on all sides except backward." I2 is forward of I3. "A straight road runs forward to backward, with long hedges to left and right." I3 is forward of I4. "A straight road runs forward to backward, alongside a long hedge to right." I4 is forward of I5. "A straight road runs forward to backward, alongside a long hedge to right." I5 is forward of I6. "Where three roads, forward, backward left and backward right, meet. Forward right is a thick hedge." I6 is forward of I7. "Open farmland." I7 is a room. "Open farmland."
>
> J1 is forward of J2. "Dense woodland, with a hedge to left." J2 is forward of J3. "Grassland, with a hedge to left." J3 is forward of J4. "The edge of farmland, with a hedge to left." J4 is a room. "Open farmland, with a long hedge blocking movement forward left, backward left or backward." J5 is forward of J6. "A road

```
running forward left to backward right, alongside a hedge." J6 is a room. "Open
farmland."

F1 is forward right of E2 and backward right of E1. F2 is forward right of E3 and
backward right of E2. F3 is forward right of E4 and backward right of E3. F4 is
forward right of E5 and backward right of E4. F5 is forward right of E6 and
backward right of E5. F6 is forward right of E7 and backward right of E6.

G1 is forward right of F1. G2 is forward right of F2 and backward right of F1. G3
is forward right of F3 and backward right of F2. G4 is forward right of F4 and
backward right of F3. G5 is forward right of F5 and backward right of F4. G6 is
forward right of F6 and backward right of F5.

H1 is forward right of G2 and backward right of G1. H2 is forward right of G3 and
backward right of G2. H3 is forward right of G4 and backward right of G3. H4 is
forward right of G5 and backward right of G4. H5 is forward right of G6 and
backward right of G5. H6 is forward right of G7 and backward right of G6.

I3 is forward right of H3 and backward right of H2. I4 is forward right of H4 and
backward right of H3. I5 is forward right of H5 and backward right of H4. I6 is
forward right of H6 and backward right of H5.

J5 is forward right of I6 and backward right of I5. J6 is forward right of I7 and
backward right of I6.
```

And now we have a hexagonally-gridded world. Route-finding will work; prepositional forms like "to be mapped backward left of" exist, just as they should; and in general these directions are just as good as the square ones. (The only thing which doesn't look good is the Index map, where Inform is just unable to draw a picture because it assumes a square grid. But that has no effect on play.)

The landscape is much easier to navigate with a little diagram:

```
To say legend (D - direction):
    let destination hex be the room D from the location;
    if the destination hex is nothing, say " ";
    otherwise say the destination hex.

Carry out looking:
    say "[fixed letter spacing] \ [legend forward] /[line break][legend forward left] ---
- [legend forward right][line break] / \[line break]--< [location] >--[line break] \
/[line break][legend backward left] ---- [legend backward right][line break] /
[legend backward] \[variable letter spacing][line break]".
```

And finally:

```
The player is in I5.

Test me with "f / forward / backward left / bl / br / br / f".
```

42 ★★★★ **Example Fore** *RB*

> Understand "fore", "aft", "port", and "starboard", but only when the player is on a vessel.

Suppose we want to understand shipboard directions, but only when the player is aboard a vessel.

```
"Fore"

Section 1 - Procedure

The starboard is a direction. The starboard has opposite port. Understand "s" as
starboard when the location is nautical.

The port is a direction. The port has opposite starboard. Understand "p" as port
when the location is nautical.

The fore is a direction. The fore has opposite aft. Understand "f" as fore when
the location is nautical.

The aft is a direction. The aft has opposite fore. Understand "a" as aft when the
location is nautical.

Does the player mean going a nautical direction when the location is nautical: it
is very likely.

Index map with fore mapped as north. Index map with aft mapped as south.
Index map with port mapped as west. Index map with starboard mapped as east.
```

And we can even add new ways to talk about the ways things are mapped, borrowing from the Relations chapter. The following will allow us to use "is abaft of" as well as "is aft of":

```
[The verb to be abaft of means the mapping aft relation.]
```

Now, to prevent the player from using NORTH onboard ship, or AFT on land:

```
A room can be nautical or earthbound. A room is usually not nautical. A direction
can be nautical or earthbound. A direction is usually not nautical. Starboard,
port, fore, aft, up, down, the inside and the outside are nautical.

Before going a nautical direction when the location is not nautical, say "Nautical
directions can only be used on board ship."

Before going an earthbound direction when the location is nautical, say
"Compass directions make no sense on board ship, but you can use [list of
nautical directions] instead." instead.

Section 2 - Scenario

The Fish Room is aft of the Spirit Room. Starboard of the Fish Room is the After
Powder Magazine. The Bread Room is aft of the After Powder Magazine.
```

The Fish Room, the Spirit Room, the Bread Room, and the After Powder Magazine are nautical.

The description of the Fish Room is "Absurd quantities of salt fish are kept here, and periodically visited by the cook or someone serving him. It is otherwise an unexceptional little chamber, so far below the waterline that there are no portholes and no external light of any kind. [paragraph break]A narrow doorway leads forward into the Spirit Room, and the After Powder Magazine is starboard."

The description of the Spirit Room is "Despite its ghostly name, this is little more than a closet down at the very navel of the ship, in which alcohol is kept: both for purifying wounds and for drinking. Under normal circumstances there is a guard posted here at every hour, lest anyone take to raiding the larder. The current absence of the guard marine strikes you as a very bad sign indeed. [paragraph break]The only way out is aft."

The description of the Bread Room is "The Bread Room is not only tiny from side to side and front to back: it is also about half the height of a proper room, and the floor slopes up very steeply with the curve of the hull. [paragraph break]What is kept here would not, on land, be dignified by the name of bread: it's hard tack, punishing to the teeth, dry on the tongue, and usually a home to weevils before half the journey is done. [paragraph break]More headroom, and access to the rest of the ship, lies fore through the After Powder Magazine."

The description of the After Powder Magazine is "Kept in near darkness because no one with any sense would bring a naked flame down here: when necessary, it can be lit with a single small lantern made of very thick glass and sealed to keep the sparks within. Sacks of powder are passed up into the higher levels of the ship by the scrubby little boys called 'powder monkeys' -- but none such are here now."

Test me with "north / aft / fore".

Chapter 4: Kinds

↖ Contents of *Writing with Inform*

← Chapter 3: Things

→ Chapter 5: Text

↘ Indexes of the examples

§4.1. New kinds

Values are to Inform what nouns are to English sentences. They represent numbers, times of day, pieces of text, places, people, doors, and so on. Because they have such an enormous variety, and because we often want to talk about what some of them have in common, we need a way to sort all of these different ideas out. That's the main aim of Inform's concept of "kind".

Every value has a kind. The kind of 10 is "number"; the kind of 11:30 PM is "time"; the kind of "jinxed wizards pluck ivy from my quilt" is "text"; and so on. The Kinds index panel shows the kinds present in the current Inform project, which will always include a wide range of built-in kinds, and usually also some new ones created in that project.

Some kinds are more general than others. For example, if we write:

```
Growler is an animal in the Savannah.
```

then Growler is an "animal", which is a kind of "thing", which is a kind of "object". When we talk about "the" kind of Growler, we mean "animal", the most specific one, but actually he belongs to all of those kinds.

As we see from this example, kinds have a whole hierarchy. Some are specialised versions of other kinds; some are not. Browsing the Kinds index shows that Inform builds its model world out of "objects". (That's really what objects are: "object" is a kind of value used to make the ingredients of the model world.) The objects fall into four fundamental kinds, "room", "thing", "direction" and "region", and "thing" is more subdivided still.

All the same, Inform starts out with a fairly simple arrangement. Because taxonomy - setting up kinds for everything - is so difficult, and depends so much on what you want it for, Inform creates relatively few kinds in advance: it has "animal" built in, but not "mammal" or "Bengal tiger". When we need more, we must make them ourselves. Let's see how we might invent these new kinds. The easy one is:

```
A Bengal tiger is a kind of animal.
```

Given that, we can then write:

```
Growler is a Bengal tiger in the Savannah.
```

That's easy enough. Adding "mammal" now looks awkward, though, because it seems to belong in between the two. All Bengal tigers are mammals, but not all animals are. But Inform can sort this out:

```
A mammal is a kind of animal. A Bengal tiger is a kind of mammal.
```

If we look at the Kinds index, we should indeed see a hierarchy:

```
object > person > animal > mammal > Bengal tiger
```

though the diagram is laid out as a sort of tree, just as zoologists do.

As another example, it may seem curious that Inform doesn't usually think "man" is a kind of "animal". This is really because, in English, people don't usually expect something like "if an animal is in the garden" to be true when it's only the Revd Mr Beebe going for a walk. People expect the word "animal" not to mean a human being, despite our common genetic ancestry. But if we want to rearrange Inform's default ideas, we can do that quite easily:

```
A man is a kind of animal. A woman is a kind of animal.
```

Or indeed we could say:

```
A human being is a kind of mammal. A man and a woman are kinds of human being.
```

While this is an ideal way to make new kinds for the model world, we are more restricted in what we can do outside of objects. For instance,

```
A glob is a kind of number.
```

isn't allowed. The numbers are fixed and all exist already; they aren't like Bengal tigers which we can simply imagine, and fill the world with. "Number" is not a concept we can specialise any further. But what we can do is to invent entirely new concepts, like so:

```
A distance is a kind of value.
```

We will see more of these later. (This isn't specialising anything - "value" is a sort of everything-category, and is too big and vague to be a kind.)

Start of Chapter 4: Kinds

Back to Chapter 3: Things: §3.26. Directions

Onward to §4.2. Using new kinds

Example 43: ★★ **Vouvray** Adding synonyms to an entire kind of thing.

§4.2. Using new kinds

This seems a good point to see what we can do with new kinds. Here we invent a new kind to provide a new sort of room:

```
A dead end is a kind of room.
```

Any dead end that we make is also a room, so it has all of the properties and behaviour of a room. For instance, every room is either "lighted" or "dark", and the default is to be lighted. But we can reverse that convention for dead ends, and we can also fill in some other properties:

```
A dead end is a kind of room with printed name "Dead End" and description "This is a
dead end. You'll have to go back the way you came." A dead end is usually dark.

The Undertomb is a dark room. East is a dead end. South is a dead end with printed
name "Collapsed Dead End". Northwest is a dead end called the Tortuous Alcove.

In the Undertomb is the candle lantern. It is lit.
```

As a result of this, three different rooms adjoin the Undertomb, all dead ends. This is much more concise than spelling them out one at a time would be.

Inform often doesn't mind in what order it is told about the world, but it may need to know the name of a kind before that kind can be used. For example,

```
A coffer is a kind of container. In the Crypt is an open coffer.
```

makes sense to Inform and results in the creation of a new thing, just called "coffer" in the absence of any other name to give it, whose kind is "coffer" and which is initially open. Whereas if Inform reads:

```
In the Crypt is an open coffer.
```

without knowing that "coffer" is a kind, it simply makes a thing called "open coffer" (and which is not a container). Inform has to be careful like this: English is simply too overflowing with multiple meanings. An "open railway ticket", for instance, is not a "railway ticket" that one can put objects into.

Start of Chapter 4: Kinds

Back to §4.1. New kinds

Onward to §4.3. Degrees of certainty

§4.3. Degrees of certainty

When we write:

```
A dead end is usually dark.
```

we are saying that it will be dark rather than lighted unless we should specify otherwise. So it would be fine to add:

```
The Tortuous Alcove is lighted.
```

because although dead ends are usually dark, this one is evidently an exception. On the other hand, if we had originally written

```
A dead end is always dark.
```

then Inform would not have permitted any exception to be made, and would have reported a problem if we had tried to make the Tortuous Alcove lighted. Besides "usually" and "always", we can also employ "seldom" and "never", which are their negatives. Thus, "never lighted" means the same as "always dark".

Start of Chapter 4: Kinds

Back to §4.2. Using new kinds

Onward to §4.4. Plural assertions

Example 44: ★ **Odin** Replacing "You see nothing special..." with a different default message for looking at something nondescript.

Example 45: ★ **Something Narsty** A staircase always open and never openable.

§4.4. Plural assertions

As the following examples show, sentences can make several assertions at once by using the plural. Suppose we have defined a kind called "high-up fixture", for instance like so:

```
A high-up fixture is a kind of thing. A high-up fixture is usually fixed in place.
```

Then the following sentence creates two such objects and puts them in their place:

```
The high shelf and the skylight window are high-up fixtures in the Lumber Room.
```

since it is equivalent to saying:

> The high shelf is a high-up fixture. The skylight window is a high-up fixture. The high shelf is in the Lumber Room. The skylight window is in the Lumber Room.

Such plurals are allowed in almost any context, and we could even define two kinds at once:

> Bucket and basket are kinds of container.

Inform constructs plurals by a form of Conway's pluralisation algorithm, which is quite good - for example, it gets oxen, geese (but mongooses), sheep, wildebeest, bream, vertebrae, quartos, wharves, phenomena, jackanapes and smallpox correct. But English is a very irregular language, and multiple-word nouns sometimes pluralise in unexpected ways. So we sometimes need to intervene:

> A brother in law is a kind of man. The plural of brother in law is brothers in law.

We are allowed to define more than one plural for the same singular text, and for the names of things, rooms or kinds, all alternative versions will be used interchangeably. (For instance, Inform defines both "people" and "persons" as plurals of "person".)

Start of Chapter 4: Kinds

Back to §4.3. Degrees of certainty

Onward to §4.5. Kinds of value

Example 46: ★★★ **Get Me to the Church on Time** Using kinds of clothing to prevent the player from wearing several pairs of trousers at the same time.

§4.5. Kinds of value

So much for making new and more specialised kinds of object - for example, new kinds of room, or new kinds of animal. This allows us to describe the physical world in elegant ways, but what about concepts which aren't so physical?

Without getting into philosophy, we can probably agree that numbers like 1, 2, 3, ..., and texts like "Jackdaws love my big sphinx of quartz", are not physical. Inside Inform, those are values, but not objects. Inform already has a good stock of this sort of concept built in, so it may not immediately seem clear why we need to create new ones. But in fact this is very useful. To describe the physical world, we need concepts like (for example) distance and brightness. We want to say that two armchairs are 12 feet apart, or that a given light-bulb is very dim. Here, "twelve feet" and "very dim" are clearly not physical objects; they need to be values, but not objects.

As these two examples suggest, sometimes we want a quantitative way to measure things, sometimes not. Thomas Hardy, in his novel "The Return of the Native", writes:

> When he drew nearer he perceived it to be a spring van, ordinary in shape, but singular in colour, this being a lurid red.

Hardy doesn't tell us that the wavelength of the light is 700nm, he tells us that the colour is "lurid red", and we understand. Later in the same chapter, Hardy writes:

> The loads were all laid together, and a pyramid of furze thirty feet in circumference now occupied the crown of the tumulus.

and now we do have a quantitative measurement: thirty feet. This is how people write about the world, and how they read about it. So Inform needs to provide both sorts of measurement.

(a) Here is a qualitative example. Suppose we would like a candle lantern to burn down, gradually diminishing in brightness. Then we'll need a way to talk about the current strength of the flame, but only in vague terms. Here goes:

```
Brightness is a kind of value. The brightnesses are guttering, weak, radiant and blazing.
```

"Brightness" is now a kind of value on a par with (for instance) "number" or "text". There are only four possible values, named as above. Kinds of value like this, where there are just a few named possibilities, are extremely useful, as we'll see.

(b) Now a quantitative example:

```
Weight is a kind of value. 1kg specifies a weight.
```

The difference here is not the way we create the kind, but the way we tell Inform what the possible values are. Instead of a list, we teach Inform some notation. As a result, "26kg" is now a value, for instance. Quantitative kinds like this are sometimes called "units", because - as in this example - they're often units in the sense of measuring things. Many Inform projects never need units, but they can still be very useful, and they're described in detail in the chapter on "Numbers and Equations".

Start of Chapter 4: Kinds

Back to §4.4. Plural assertions

Onward to §4.6. Properties again

§4.6. Properties again

So now we have seen two fundamental ideas: "value" and "kind". We have seen how to make a source text which refers to many values - for example, Growler the Bengal tiger, 23kg, "Collapsed Dead End", blazing, 7, all values of different kinds.

But we don't just want a way to refer to values, we want to lay out facts about them. Inform understands two sorts of fact, which it calls properties and relations. Properties are about single values in isolation: Growler is male. Relations are about how values interact with each other: Growler likes Bambi. (Or would like to eat Bambi, anyway.) Relations are really the central organising idea of Inform, and we've seen them many times already:

```
Growler is in the Savannah.
```

expresses a relation called "containment" between Growler and the Savannah. Much more about this in the chapter on Relations: for now, let's go back to the simpler idea of properties.

In Inform terms, a "property" is any fact about a value (other than its kind) which the author is allowed to choose. For example,

```
Growler is an animal. Growler is male. The description of Growler is "What immortal
hand or eye could frame thy fearful symmetry?".
```

The first of these sentences talks about Growler's kind, but the other two sentences tell Inform about his properties. Properties are divided into either/or properties - "male" versus "female" - and value properties - such as the description of something, which can be any text.

The Kinds index shows which kinds of object are allowed to have properties. Every object is, so there's no problem with Growler. In general, if Inform can find a sensible way to store properties, then it will allow them. But it won't allow (for example) properties of numbers. There are only a finite number of Bengal tigers in the world (fewer than three thousand, alas), so Inform can easily store individual description texts for each one of them. But there are an infinite range of numbers. (Inform does allow adjectives like "even" or "odd" to be used about number - saying "if 7 is odd" is fine, for example - but they are not properties in the Inform sense, because the author doesn't get to choose. The author can choose whether Growler is a male or female tiger, but not whether 7 is an even or an odd number.)

Start of Chapter 4: Kinds

Back to §4.5. Kinds of value

Onward to §4.7. New either/or properties

§4.7. New either/or properties

Properties can't be handed out completely freely. In the previous chapter, we saw that we were allowed to make a chair "portable" and to make a room "dark". But if we try this the other way round, Inform produces a Problem message. This is because every property must be created in a way which lays out what values are allowed to have it. The Standard Rules built into Inform say that

```
A thing can be fixed in place or portable.
```

and as a result it won't allow "The Savannah is portable" because the Savannah is a room, not a thing.

We must do the same. To go back to our example "dead end" kind:

```
A dead end is either secret or ordinary.
```

This creates just one new property, not two. The names are taken as the two states of a single either/or property: secret means not ordinary, ordinary means not secret. Alternatively, we could just say:

```
A dead end can be secret.
```

in which case the opposite of "secret" would be "not secret".

Now we have a property which can be given to any value of the kind "dead end". We're also free to add to the definitions of kinds which already exist, including those built into Inform: for instance,

```
A room is either indoors or outdoors.
```

If we make the above definitions then all dead ends will be "ordinary" and all rooms "outdoors" unless the source text says otherwise. That is, in the absence of other information it's assumed that an either/or property is not true. We could reverse by writing:

```
A dead end is usually secret. A room is usually indoors.
```

A property can be used by several kinds at once. For example, the built-in either/or property "open" is used by both doors and containers, even though door isn't a kind of container and container isn't a kind of door. In fact, although it's more usual to declare properties for whole kinds, they can actually be given to single values one at a time, if we like:

```
The umbrella is carried by the player. The umbrella can be open.
```

And now the umbrella, which is a thing and not a door or container, can also have the property.

Start of Chapter 4: Kinds

Back to §4.6. Properties again

Onward to §4.8. New value properties

Example 47: ★★ **Change of Basis** Implementing sleeping and wakeful states.

§4.8. New value properties

So much for either/or properties. Now we move on to properties which have values attached. The same principles apply, but the wording is different. For example,

```
A dead end has some text called the river sound. The river sound of a dead end is
usually "a faint whispering of running water". The Tortuous Alcove has river sound "a
gurgle of running water".
```

The property "river sound" is now applicable only to dead ends, so we would not be allowed to talk about "the river sound of the Savannah", say. Moreover, it's required to hold a piece of text. If we tried the following:

```
The river sound of the Tortuous Alcove is 7.
```

...then Inform would object, because the number 7 is the wrong kind of value to go into the "river sound" property. If we need a numerical property, we can try this instead:

> A dead end has a number called the difficulty rating. The Tortuous Alcove has difficulty rating 7.

Suppose that we were to add:

> The Exquisitely Narrow Defile is a dead end.

The Defile must have a river sound, of course, because we said that every dead end would have one. We haven't said what that river sound will be, but Inform can work it out, because we did say this:

> The river sound of a dead end is usually "a faint whispering of running water".

If there are no instructions at all about the value of a property, Inform fills in the default value of the appropriate kind - in this case, it would be a blank text. (A table of the kinds which can be used for properties, and their default values, can be found in the Kinds index.)

Start of Chapter 4: Kinds

Back to §4.7. New either/or properties

Onward to §4.9. Using new kinds of value in properties

Example 48: ★ **Would you...?** Adding new properties to objects, and checking for their presence.

Example 49: ★★ **Straw Boater** Using text properties that apply only to some things and are not defined for others.

§4.9. Using new kinds of value in properties

It turns out to be very useful to create a new kind of value, and then create a property to hold it. So useful, in fact, that Inform provides two shortened forms for doing so. Here is the first, and the second (making "conditions") is in the section following.

Suppose we go back to our example of the candle lantern whose brightness we have to measure. It's clear that what we want to do is to define:

> Brightness is a kind of value. The brightnesses are guttering, weak, radiant and blazing.

And now we can use the technique of the previous section:

> The lantern has a brightness called the flame strength. The flame strength of the lantern is blazing.

This works very nicely. The "flame strength" property is now only allowed to have one of four values we allowed: guttering light, weak light, radiant light or blazing light. So we have succeeded in recording our measurement.

But it seems artificial to call the brightness of the lantern "flame strength", when we could instead simply call it "brightness". Much simpler to write:

```
The lantern has a brightness. The lantern is blazing.
```

Now "brightness" is the name of both the property and the kind of value. What's particularly nice is that we can now use the names of the possible brightnesses - "weak", "blazing" and so on - as adjectives. Inform knows that "The lantern is blazing" must be talking about the brightness property, because "blazing" is a brightness.

Now we can improve our dead ends:

```
A dead end is a kind of room with printed name "Dead End" and description "This is a dead end, where crags in the uneven rock are caught by the [brightness of the lantern] flame you hold aloft. Despite [river sound] there is no sign of the stream." A dead end is usually dark.
```

The "[brightness of the lantern]" is printed not as literal text, but as whatever the brightness currently is. (The square brackets mark it as what is called a text substitution, which will be the subject of the next chapter.) So we get something like this:

```
This is a dead end, where crags in the uneven rock are caught by the blazing flame you hold aloft. Despite a faint whispering of running water there is no sign of the stream.
```

So now we have a lantern, which has a brightness as a property. But we can build on this further if we like. A brightness such as "guttering" is a value, so it can have properties in its own right. That can be quite useful, in fact:

```
A brightness can be adequate or inadequate. A brightness is usually adequate. Guttering is inadequate.
```

This is convenient because it divides up the brightnesses:

```
The player carries a book. The description of the book is "[if the brightness of the lantern is adequate]Many secrets are now yours.[otherwise]No, the print's too tiny by this awful light."
```

And while we're at it, let's give each brightness its own corresponding temperature:

```
Temperature is a kind of value. 100C specifies a temperature.

A brightness has a temperature. The temperature of a brightness is usually 700C. The temperature of blazing is 1400C. The temperature of radiant is 1100C.

The description of the lantern is "The lantern shines with a flame at [temperature of the brightness of the lantern]."
```

(Candle flames are hotter than most people think.)

See **Text with substitutions** *for more on varying what is printed*

⬆ Start of Chapter 4: Kinds

⬅ Back to §4.8. New value properties

➡ Onward to §4.10. Conditions of things

⬇ Example 50: ★ **The Undertomb 1** A small map of dead ends, in which the sound of an underground river has different strengths in different caves.

⬇ Example 51: ★★ **The Undertomb 2** Flickering lantern-light effects added to the Undertomb.

⬇ Example 52: ★★★ **The Crane's Leg 1** A description text that automatically highlights the ways in which the object differs from a standard member of its kind.

⬇ Example 53: ★★★ **Signs and Portents** Signpost that points to various destinations, depending on how the player has turned it.

§4.10. Conditions of things

Now for an even more abbreviated way to create a new kind of value, and at the same time create a property to hold it. Suppose we have something, say a wine cask, which we know is always in one of three different states. We can write:

```
The cask is either customs sealed, liable to tax or stolen goods.
```

This is just like our example of the lantern having possible brightnesses, but it's quicker to do, because we don't need to create or name the kind of value. (The trade-off is that we can't use it for anything else as well.)

Initially the cask will be "customs sealed", the first value we gave. We could now write, for instance,

```
The description of the cask is "A well-caulked Spanish wine cask.
[if liable to tax] It really is a shame to have to pay duty on it!"
```

Or, as a second example, here we're going to allow a whole kind to have the property, not just a single object:

```
Colour is a kind of value. The colours are red, green and white.
A colour can be bright, neutral or flat. Green is neutral.
```

Now in fact these properties are not anonymous: Inform has worked out names for them, even though we didn't give any. The usual arrangement is that the name is the name of the object with the word "condition" tacked on: for instance, "cask condition". So we could write:

```
The printed name of the cask is "wine cask ([cask condition])".
```

so that sometimes this would be "wine cask (liable to tax)", sometimes "wine cask (stolen goods)" and so on.

But only usually, because we might need to define several different conditions of the same thing, and then the names would collide. For instance, suppose we write:

```
A fruit is a kind of thing. A fruit can be citrus, berry, melon, or pome.
```

This makes a property and a kind of value each called "fruit condition". But now suppose we add that:

```
A fruit can be unripened, ripe, overripe, or mushy.
```

This is a quite unrelated property - a fruit could have any combination of these two properties, in fact. Left to itself, Inform will call the second one "fruit condition 2", which isn't really ideal if we ever do need to refer to it in other source text. So we are also allowed to give these conditions names of our own choosing:

```
A fruit can be unripened, ripe, overripe, or mushy (this is its squishiness property).
```

And now the resulting property and kind of value would be called "squishiness".

Start of Chapter 4: Kinds

Back to §4.9. Using new kinds of value in properties

Onward to §4.11. Default values of kinds

§4.11. Default values of kinds

Just about every kind has a "default value". Inform needs this when it knows that something has to be a value of a given kind, but it hasn't been told what the value is. For example, in the previous chapter we saw that every thing has a "description" text, but we also created plenty of things without describing them. So if Inform reads

```
The conference pear is in the bowl.
```

and it isn't told anything else about the pear, what should it set the description of the pear to?

The answer is that Inform knows the description has to be a value of the kind "text", so it uses the default value of "text". Not very interestingly, this is just the blank text "".

Being uninteresting is exactly the idea, of course. The default number is 0, for instance. (Default values are tabulated in the Kinds index.)

It's sometimes useful to be able to refer to the default value of a kind without having to spell out what this is (especially if the kind is something obscure, or we're trying to write a rule for an extension which has to work in situations we don't fully know about).

default value of (name of kind) **... value**

Produces the default value of the kind named. Examples:

```
The silver repeater is here. "You catch sight of a silver repeater watch, hands
immobile at [default value of time]."
```

produces the output:

```
You catch sight of a silver repeater watch, hands immobile at 9:00 am.
```

because nine in the morning is the default time in Inform. If we have:

```
Brightness is a kind of value. The brightnesses are guttering, weak, radiant
and blazing.
```

then "default value of brightness" is guttering, the first brightness created. When it comes to kinds of object, we sometimes have to be a little careful. For example,

```
default value of room
```

is always going to be fine (it's always the first room created in the source text). But

```
default value of vehicle
```

would produce a Problem message if there were no vehicles in the world.

⬆ Start of Chapter 4: Kinds
⬅ Back to §4.10. Conditions of things
➡ Onward to §4.12. Values that vary

§4.12. Values that vary

Sometimes a value important to the simulated world will not naturally belong to any thing or room, and should not be kept in a property. In fact, we have seen a value that varies already: "location", which holds the room in which the story is presently taking place. Here's how we might make a new one:

```
The prevailing wind is a direction that varies. The prevailing wind is southwest.
```

Or "which varies" would also be allowed, as would the more traditional computing term "variable":

```
The prevailing wind is a direction variable. The prevailing wind is southwest.
```

A briefer way to do this is to use the word "initially", which alerts Inform to the possibility that the value will change in future:

```
The prevailing wind is initially southwest.
```

This creates the variable and gives it an initial value all in one sentence.

It's not compulsory to give an initial value. If we do not, Inform will use the default value for its kind. (See the table in the Kinds index.) For example, writing just

```
The grand tally is a number that varies.
```

will start it at the value 0, because that's the default value for numbers.

We can have variables of any of the kinds of value, including new ones, but should watch out for a potential error. If we write:

```
The receptacle is a container that varies.
```

in a world which has no containers at all, Inform will object, because it will be unable to put any initial value into the receptacle variable. A similar complaint will be made if we write:

```
Colour is a kind of value. The fashionable shade is a colour that varies.
```

without ever having defined any colours. Something else we are not permitted is:

```
The receptacle is an open container that varies.
```

because the openness of a given container may change during play, so that the value in the variable might suddenly become invalid even though the variable itself had not changed.

As a final note on kinds, when Inform reads something like this:

```
Peter is a man. The accursed one is initially Peter.
```

it has to make a decision about the kind of "accursed one". Peter is a "man", so that seems like the right answer, but Inform wants to play safe in case the variable later needs to change to a woman called Jane, say, or even a black hat. So Inform in fact creates "accursed one" as an object that varies, not a man that varies, to give us the maximum freedom to use it. If we don't want that then we can override it:

```
Peter is a man. The accursed one is initially Peter.
The accursed one is a man that varies.
```

thus telling Inform exactly what is intended.

⬆ Start of Chapter 4: Kinds

⬅ Back to §4.11. Default values of kinds

➡ Onward to §4.13. Values that never vary

⬇ Example 54: ★★★ **Real Adventurers Need No Help** Allowing the player to turn off all access to hints for the duration of a game, in order to avoid the temptation to rely on them overmuch.

§4.13. Values that never vary

It's sometimes useful to name even values which don't change. For example, suppose the story involves driving, and the same speed limit value comes up in many places. Rather than typing "55" (say) every time it comes up, we might prefer to write:

```
The speed limit is always 55.
```

at the start of the source text, and then talk about "the speed limit" every time we would otherwise have typed "55". Just as the word "initially" alerts Inform that we want the named value to change during play, the word "always" tells it that we don't.

This might seem pointless, because "speed limit" only means the same thing as "55" and takes more typing. But there are two reasons why authors might want to use this feature anyway. One is that it's easier for a human reader to understand the significance of a line like:

```
if the speed is greater than the speed limit, ...
```

Another is that it makes it easier to change our minds about the value, because if we decide we want 70 as the limit and not 55, we only need to make one change at the start of the source text:

```
The speed limit is always 70.
```

which is much easier than combing through a long source text trying to find many individual things which need changing.

"Speed limit" is then a number constant. Any attempt to set this elsewhere, or change its value, will result in a Problem message, and moreover it can be used in contexts where only constant values are allowed. For example,

```
The generic male appearance is always "He is a dude."

Trevor is a man. The description of Trevor is the generic male appearance.
```

means that the SHOWME TREVOR testing command produces, among other data:

```
description: "He is a dude."
```

Start of Chapter 4: Kinds

Back to §4.12. Values that vary

Onward to §4.14. Duplicates

§4.14. Duplicates

Although it is only useful to a limited extent, we can make any number of copies of something:

```
"Polygons"

A shape is a kind of thing. A square is a kind of shape. A triangle is a kind of shape.

The Geometry Lab is a room. In the Geometry Lab are three triangles and two squares.
```

The description "three triangles" makes three identical things, each of the kind "triangle", and similarly for the squares. When the above is compiled, the player can type TAKE TWO TRIANGLES or TAKE ALL THE TRIANGLES and so forth.

Four caveats. Firstly, a counted-out description like "two squares" is only allowed if it combines a number with the name of a kind which is already known (perhaps modified with adjectives, so "two open doors" is fine). If we say:

```
Two circles are in the Lab.
```

without having defined "circle" as a kind in advance, then only a single object will be created - whose name is "two circles". (This is because many natural names start with numbers: "six of clubs", for instance, referring to a single playing card, or "12 Hollywood Close" meaning a single house. We wouldn't want such names to be misinterpreted.)

The second caveat is that excessive duplication is expensive in memory and running time. It is perfectly legal to say

```
In the Lab are 75 triangles.
```

but the resulting story may be a little sluggish: and Inform draws the line at 100, refusing to create more duplicates than that in any single place. If we really need more than about fifty duplicated objects - say, a tombola containing raffle tickets numbered 1 to 1000 - it is usually better to find some less literal way to simulate this: for instance, only having a single raffle ticket, but with a randomly chosen number on it.

If there are very many items in the same place, commands like TAKE ALL and DROP ALL may mysteriously not quite deal with all of them - this is because the parser, the run-time program which deciphers typed commands, has only limited memory to hold the possibilities. It can be raised with a use option like so:

```
Use maximum things understood at once of at least 200.
```

(The default is, as above, 100. Note the "at least".)

Thirdly, note that Inform's idea of "identical" is based on what the player could type in a command to distinguish things. In a few cases this can make items unexpectedly identical. For example:

```
The Lab is a room. A chemical is a kind of thing. Some polyethylene and polyethylene-terephthalate are chemicals in the Lab.
```

results surprisingly in "You can see two chemicals here", because the run-time system truncates the words that are typed - POLYETHYLENE and POLYETHYLENE-TEREPHTHALATE look like the same word in a typed command. So Inform decides that these are indistinguishable chemicals. Typically words are truncated after 9 letters, though (unless the Glulx setting is used) punctuation inside a word, such as an apostrophe, can make this happen earlier. The best way to avoid trouble is simply to use more easily distinguishable names. For example:

```
Some polyethylene and polyethylene terephthalate are chemicals in the Lab.
```

works fine, because now only one chemical can be called TEREPHTHALATE, and that means they can be distinguished.

Finally: numbers up to twelve may be written out in words in the source text, but larger ones must be written as numerals. So "twelve" or "12", but "13" only.

Start of Chapter 4: Kinds

Back to §4.13. Values that never vary

Onward to §4.15. Assemblies and body parts

Example 55: ★★★ **Early Childhood** A child's set of building blocks, which come in three different colours - red, green and blue - but which can be repainted during play.

§4.15. Assemblies and body parts

In the previous chapter, we saw that it was possible to make sub-parts of things. For instance,

```
The white door is in the Drawing Room. The handle is part of the white door.
```

creates a door with an attached handle. But what if we want to say that not just this door, but every door, should have a handle? To do this we first need to create a kind called "handle", since there will clearly need to be many handles. The solution is:

```
A handle is a kind of thing. A handle is part of every door.
```

"Every" is a loaded word and best used sparingly. A sentence like "A handle is part of every handle" would, if taken literally, mean that a handle takes forever to make and is never finished. Inform will reject this, but the moral is clear: we should think about what we are doing with "every".

We will usually want to work with smaller collections - not literally every room, but with a whole set of them all the same. We can do that like so:

```
A silver coin is a kind of thing. A banking room is a kind of room. Five silver coins are in every banking room.
```

The effect of sentences like these is to make what we might call "assemblies" instead of single things. When a banking room is created, so are five more silver coins; when a door is created, so is another handle. Such sentences act not only on items created later on in the source text, but also on all those created so far.

This is especially useful for body parts. If we would like to explore Voltaire's suggestion that history would have been very different if only Cleopatra's nose had been shorter, we will need noses:

```
A nose is a kind of thing. A nose is part of every person.
```

Of course, if we make an assembly like this then we had better remember that the player is also a person and also gets a nose. In fact slightly odd things can happen if we combine this with changing the identity of the player. This works:

```
Cleopatra is a woman in Alexandria. The player is Cleopatra.
A nose is a kind of thing. A nose is part of every person.
```

but if those lines are in reverse order then Cleopatra's nose is assembled before she becomes the player, with the result that it ends up called "Cleopatra's nose" rather than "your nose" in play - which is very regal but probably not what we want. To avoid this, settle the player's identity early on in the source text.

All of the assemblies above make objects. Most make these new objects "part of" existing ones, but as we saw, they can also be "in" or "on" them. In fact, though, assemblies work in much more general ways: they can assemble values of almost any kind, placed in almost any relationship. To make use of that, we need to create a new verb, a topic which won't be covered properly until a later chapter, but here goes:

```
A colour is a kind of value. The colours are red, green and blue.

Liking relates various people to various colours. The verb to like means the liking
relation.

Every person likes a colour.
```

Now every time a person is created, so is a colour which that person will like. If there are two people in the world, the player and Daphne, then we now have five colours: red, green, blue, Daphne's colour and the player's colour. Alternatively, we can assemble the other way around:

```
A person likes every colour.
```

Now we're telling Inform that every time a colour is made, a new person is also made - someone who will like that colour. So this sentence effectively makes three new people, one who likes red, one who likes green, and one who likes blue.

⬆ Start of Chapter 4: Kinds

⬅ Back to §4.14. Duplicates

➡ Onward to §4.16. Names made in assembly

⬇ Example 56: ★ **Being Prepared** A kind for jackets, which always includes a container called a pocket.

⬇ Example 57: ★★ **Model Shop** An "on/off button" which controls whatever device it is part of.

⬇ Example 58: ★★★ **U-Stor-It** A "chest" kind which consists of a container which has a lid as a supporter.

⬇ Example 59: ★★★ **The Night Before** Instructing Inform to prefer different interpretations of EXAMINE NOSE, depending on whether the player is alone, in company, or with Rudolph the Red-nosed Reindeer.

---

§4.16. Names made in assembly

Something skated over in the previous section is the question of how Inform gives names to objects (or other values) it creates in an assembly. The standard thing naming combines the names of what's being assembled. For example:

```
A nose is a kind of thing. A nose is part of every person. Antony and Cleopatra are
people.
```

might result in the creation of "Antony's nose", part of Antony, and "Cleopatra's nose", part of Cleopatra. In this way, Inform names the noses after their owners. It will always do this unless there are multiple indistinguishable things being created, as in the "five silver coins are in every banking room" example: those will all just be called "silver coin".

A small pitfall of this is that if we write:

```
Marcus Tullius Cicero is a person.
```

then although "Marcus Tullius Cicero's nose" and "Cicero's nose" are both valid names for the consular nose, "Marcus's nose" is not.

The standard naming scheme is often about right, but as usual Inform offers a way to improve it in particular cases. For example, if we write:

```
Every room contains a vehicle (called its buggy).
```

then we will find the world full of, say, the Garden buggy, the Patio buggy and so on - instead of the Garden vehicle, the Patio vehicle and so on, which is what we would have had without the "called..." part. Similarly, we could write:

```
A person (called its fan) likes every colour.
Every person likes a colour (called his favourite colour).
```

The former would produce new people with names like "Green's fan", whereas the latter would produce new colours with names like "Daphne's favourite colour".

So much for an informal description. Here is exactly what Inform does:

> (1a) If there is a "called..." text, Inform uses it, expanding out "its" (or "his" or "her" or "their") to a possessive form of the name of the owner, so to speak, and "it" (or "he" or "she" or "they" or "him" or "them") to the name itself.
>
> (1b) If there's no "called..." text, Inform behaves as if we had written "(called its K)", where K is the name of the kind.
>
> (2) If this results in a value which isn't an object being given a name which already exists, Inform tacks on a number to force the new name to be different from existing ones: e.g., "Daphne's colour 2", "Daphne's colour 3", ...

(The reason that (2) doesn't affect objects is that objects are allowed to have names clashing with other objects, or no name at all, whereas other values have to have names belonging to themselves alone.)

Start of Chapter 4: Kinds
Back to §4.15. Assemblies and body parts
Onward to §4.17. Postscript on simulation

§4.17. Postscript on simulation

That concludes our tour through the design of the initial state of a simulated world. We have seen how to create rooms and to stock them with containers, supporters, devices, doors, men and women. The player of such a simulation can explore, move things around, open and close certain doors and containers, lock or unlock them provided a suitable key is found, switch machines on or off, and so on.

But that is about all. There is as yet no element of surprise, no aim or sense of progress to be earned, and no narrative thread. We have painted the backcloth, and laid out the properties, but the actors have yet to take the stage.

Start of Chapter 4: Kinds
Back to §4.16. Names made in assembly
Onward to Chapter 5: Text: §5.1. Text with substitutions

Examples from Chapter 4: Kinds

Start of this chapter
Chapter 5: Text
Indexes of the examples

★★★ **Example Vouvray**

43 Adding synonyms to an entire kind of thing. RB

The Understanding chapter lays out ways to change how the player can refer to objects, but we may not want to wait that long for some of the basic features. Here, for instance, is how to add synonyms that the player can use to refer to an entire kind of object:

```
"Vouvray"

The Wine Emporium is a room. "Set aside, you rather suspect, for tourists: this chamber is barrel-vaulted stone, lined on each side with casks of aging wine. Discarded brochures here and there advertise Wine Tours of the Loire Valley in three different languages, none of them French."

A cask is a kind of thing. A cask is always fixed in place. Understand "cask" or "barrel" as a cask. Understand "casks" or "barrels" as the plural of cask.

The Vouvray cask and the Muscadet cask are casks in the Wine Emporium.

Test me with "get barrels / get barrel / muscadet / x casks / x muscadet cask".
```

---

44 Example **Odin** RB

Replacing "You see nothing special..." with a different default message for looking at something nondescript.

In recent years there has been a strong trend towards providing unique descriptions for all implemented objects. Often this is a good idea, but there are also contexts in which we may want to discourage the player from looking too closely at some things and concentrate his attention on just a few interesting ones.

The trick here is that leaving items completely undescribed leads to rather dull exchanges like this:

```
>x table
You see nothing special about the table.
```

...which can leave the player with the impression that the author was simply too lazy to describe everything. So it can be a good idea to replace that default message with a different one more appropriate to the game. For instance:

```
"Odin"

The House of a Mortal Farmer is a room. "Having two separate rooms, this house testifies to considerable wealth and success at agriculture."

The Bedroom is inside from the House.

A chair is a kind of supporter. A chair is always enterable.
```

```
In the House are a table, two chairs, a basket, and a hearth. On the table is a loaf of bread.

The description of a thing is usually "You give [the noun] a glance, but it is plainly beneath your attention."
```

Because the description is attached to a whole kind ("thing"), it is really a blanket instruction about many objects at once. More specific instructions always override less specific ones, so we can easily make exceptions. For instance, the following will work correctly:

```
The infant is a man in the basket. The description of the infant is "So strong and fat that you wonder whether one of your fellow gods is acquainted with the mistress of the house-- but it's no concern of yours, of course."

Test me with "x table / x chair / x infant".
```

45 ★ Example Something Narsty RB

A staircase always open and never openable.

In a game with many staircases, we might want:

```
A staircase is a kind of door. A staircase is always open. A staircase is never openable.
```

Defining the staircase this way means that we will never be able to get away with (for instance) a folding ladder into the attic which is sometimes closed up. So alternatively we might do

```
"Something Narsty"

A staircase is a kind of door. A staircase is usually open. A staircase is seldom openable.
```

We could then write a rule so that whenever the player types CLIMB [any staircase], the command is diverted to an enter command, while all other doors refuse to respond to such treatment. Still, this kind is now usable, as we see in this trivial example:

```
The ladder is a staircase. It is above the Woodshed and below the Scary Loft.

Test me with "up".
```

But these are refinements for a later chapter.

46 ★★★ Example Get Me to the Church on Time RB

Using kinds of clothing to prevent the player from wearing several pairs of trousers at the same time.

Inform's default handling of wearable things does not make any rules about what can be worn together. Suppose, however, we have a game in which there are a large number of different garments, and we want to keep the player from wearing (say) more than one pair of pants at once:

```
"Get Me to the Church on Time"

A pair of pants and a shirt are kinds of thing. A pair of pants and a shirt are
usually wearable.

Some golf pants are a pair of pants. The description is "Checked in red and
green, with tiny frolicking gophers every few inches."

Some tuxedo trousers are a pair of pants. The description is "Black, pressed,
and slimming."

The frilly shirt is a shirt. The description of the frilly shirt is "She insisted."

The polo shirt is a shirt. The description is "Turquoise and bright yellow, the
colors selected by your golfing buddies."

The player wears the golf pants and the polo shirt. The player carries the tuxedo
trousers and the frilly shirt.

The Wedding Chapel Dressing Room is a room. "The bride's dressing room is a
lavish suite with closets, hangers, dressmaker's dummies, boxes of straight pins
and sewing notions, combs, lotions, brushes, and hair fixatives, plus room for
fifteen female attendants and a photographer. Before they shoved you out of the
room you even got a glimpse of a small reference library including '1001 French
Braids' and 'Corset-Lacing For Beginners.'

This is the groom's dressing room. You get a framed photograph of Elvis, a dusty
mirror, and the floor space of an average toilet stall."

The dusty mirror and the photograph of Elvis are scenery in the Dressing Room.
The description of the mirror is "You can't really get more than a silhouette
impression of yourself." The description of Elvis is "He reminds you that you'd
better get out there before the organist switches to Hound Dog."
```

And now the rule itself, borrowed from a later chapter:

```
Instead of wearing a pair of pants when the player is wearing a pair of pants
(called the wrong trousers):
    say "You'll have to take off [the wrong trousers] first."

Instead of wearing a shirt when the player is wearing a shirt (called the wrong
top):
    say "You'll have to take off [the wrong top] first."
```

```
When play begins:
    say "From the other side of the door, you hear the organist move on from his
instrumental interpretation of 'I Wanna Hold Your Hand' to a somewhat more
spirited rendition of 'Help! I Need Somebody!'. Okay, okay, but you've been
rushing things along since the 16th fairway, and you can't be more than a half-
hour late... Surely that mother of hers can't blame you for that?"

Test me with "i / x trousers / wear trousers / x golf pants / take off golf pants /
wear trousers / x frilly shirt / x polo shirt / wear frilly shirt / doff polo shirt / wear
frilly shirt".
```

If we wanted to, we could make similar kinds for hats, shoes, and so on, and have a simple but effective system of clothing. A more complicated treatment might keep track of layering and describe the player's outfit differently depending on which clothes were outermost -- an example for a later chapter.

---

47 ★★★ **Example** **Change of Basis** RB

Implementing sleeping and wakeful states.

Suppose we want to allow the player to go to sleep some of the time:

```
"Change of Basis"

A person is either awake or asleep. A person is usually awake.
```

The important thing to note here is that it does not work to say "the player is either asleep or awake". This is because the player is not necessarily one specific person or thing during the game: the identity of the player can be changed, as we will see later.

So if we want to make rules about the properties of the player, we should attach these rules to the "person" kind.

```
Linear Algebra Class is a room. "The blackboard is covered with square
arrangements of numbers. These are supposed to convey something to you, but
mostly you're finding them soporific."
```

Now a few rules about changing from one state to the other:

```
Instead of sleeping: now the player is asleep; say "You drop off."

Instead of doing something other than waking up, waiting or sleeping when the
player is asleep:
    say "Ssh! You're sleeping!"

Instead of sleeping when the player is asleep:
    say "Zzzz."

Instead of waking up when the player is asleep:
    now the player is awake;
    say "You come to suddenly, wiping drool from your lips."
```

```
Instead of doing something other than looking or sleeping when the player is awake:
    say "You'd really rather just sleep through this."

Test me with "wake up / sleep / look / z / sleep / wake up / look".
```

48

★ Example **Would you...?**

RB

Adding new properties to objects, and checking for their presence.

For instance, if we want to give some objects a flavor:

```
"Would you...?"

The House is a room. The mouse is an animal in the House.

The player carries some green eggs and a ham.

A food is a kind of thing that is edible. Food has some text called flavor. The flavor of food is usually "Tolerable."
```

Things are, in general, not edible by default, so we have to make them edible specifically in order to allow them to be eaten by the player. Here we've defined food to be edible by default, and we have given it a standard piece of flavor text.

```
The ham and the green eggs are food. The flavor of the green eggs is "Delicious!"

After eating something:
    if the noun provides the property flavor, say "[the flavor of the noun][paragraph break]";
    otherwise say "It's [noun]-flavored."
```

Note that we use "if the noun provides a flavor..." to make sure that the property exists before attempting to use it. Otherwise, there is the risk that we will try to print a property that does not exist, resulting in errors in the game.

We will only get the "It's [noun]-flavored." response if we successfully eat something that is not a food and does not have flavor text. To test this feature, let's suppose something that isn't exactly food but can theoretically be chewed on:

```
The player carries some paper. The paper is edible.

Test me with "eat ham / eat green eggs / eat paper".
```

49

★★★ Example **Straw Boater**

RB

Using text properties that apply only to some things and are not defined for others.

Sometimes we like to give properties to kinds of thing, but not fill them in in all cases. For instance, we might have vehicles that optionally make noise, and those might have a "movement sound".

All properties have a default value, which we can find by looking in the Kinds tab of the index. This is what the property will be set to automatically, if we do not change it ourselves. In the case of a text property, that is ""; so for instance we might use our movement sound thus:

```
"Straw Boater"

Boathouse is a room. "A boathouse circa 1915, which -- though in poor repair -- still suggests Sunday afternoon jaunts taken by women in white gowns and men in straw hats."

North of the Boathouse is the Shallow Water. The description of Shallow Water is "Just south is the boathouse, and beyond it are trees and the marble terrace of the house above. The water deepens to the north."

North of Shallow Water is Deep Water. The description of Deep Water is "From here the boathouse has dwindled invisibly to the south, and you have a broad panorama of the shoreline, all the way down to the Skeleton Point Lighthouse in the southeast."

A vehicle has some text called the movement sound. The sailboat and the motorboat are vehicles in the Boathouse. The movement sound of the motorboat is "VRRRROOOMMMM..." Understand "boat" as the sailboat. Understand "boat" as the motorboat.
```

Note that we haven't given the sailboat any movement sound at all.

```
After going somewhere by a vehicle (called cart):
    if the movement sound of the cart is not "", say "[the movement sound of the cart][paragraph break]";
    continue the action.

Instead of exiting when the player is in a vehicle and the location is not the Boathouse:
    say "You're not dressed for a swim."

Instead of going somewhere when the player is not in a vehicle:
    say "You'd rather not try to make this journey by swimming alone."

Test me with "n / get in sailboat / n / get out / s / get in motorboat / n / n".
```

---

50 ★ **Example The Undertomb 1** RB

A small map of dead ends, in which the sound of an underground river has different strengths in different caves.

This example creates a kind of room, "dead end", and gives each a textual description called its "river sound". Note the use of a text substitution to fill in the appropriate river sound text in each dead end.

```
"The Undertomb"

A dead end is a kind of room with printed name "Dead End" and description
"This is a dead end. You'll have to go back the way you came, consoled only by
[river sound]." A dead end is usually dark.

The Undertomb is a dark room. East is a dead end. South is a dead end with
printed name "Collapsed Dead End". Northwest is a dead end called the
Tortuous Alcove. In the Undertomb is the lantern. It is lit.

A dead end has some text called river sound. The river sound of a dead end is
usually "a faint whispering of running water". The Tortuous Alcove has river
sound "a gurgle of running water".

Test me with "get lantern / e / w / s / n / nw / se".
```

51 ★★★ **Example The Undertomb 2** RB

Flickering lantern-light effects added to the Undertomb.

```
"Undertomb 2"

Section 1 - Procedure

Brightness is a kind of value. The brightnesses are guttering, weak, radiant and
blazing.

A brightness can be adequate or inadequate. A brightness is usually adequate.
Guttering is inadequate.

Temperature is a kind of value. 100C specifies a temperature.

A brightness has a temperature. The temperature of a brightness is usually
700C. The temperature of blazing is 1400C. The temperature of radiant is
1100C.

A dead end is a kind of room with printed name "Dead End" and description
"This is a dead end, where crags in the uneven rock are caught by the
[brightness of the lantern] flame you hold aloft. Despite [river sound] there is no
sign of the stream." A dead end is usually dark.

A dead end has some text called river sound. The river sound of a dead end is
usually "a faint whispering of running water".

Section 2 - Scenario
```

```
The Undertomb is a dark room. "From this dim cross-groined room, exits depart
east, south, and northwest."

East is a dead end. South is a dead end with printed name "Collapsed Dead
End".

Northwest is a dead end called the Tortuous Alcove. The Tortuous Alcove has
river sound "a gurgle of running water".

The player carries a book. The description of the book is "[if the brightness of the
lantern is adequate]Many secrets are now yours.[otherwise]No, the print's too
tiny by this awful light."

In the Undertomb is a lantern. It is lit. The lantern has a brightness. The lantern
is blazing. The description of the lantern is "The lantern shines with a flame at
[temperature of the brightness of the lantern]."

After waiting in the Tortuous Alcove when the brightness of the lantern is not
guttering:
    now the lantern is the brightness before the brightness of the lantern;
    say "You wait so long that your lantern dims a bit."

Test me with "east / west / get lantern / east / west / south / north / northwest /
read book / x lantern / z / x lantern / read book / look / z / x lantern / read book /
look / z / x lantern / read book / look".
```

52 ★★★★ **Example The Crane's Leg 1** *RB*

A description text that automatically highlights the ways in which the object differs from a standard member of its kind.

> When you see a long object, you don't have to think that it is too long if being long is the property given by the nature. It is proved by the fact that a duckling, having short legs, will cry if you try to draw them out by force, and that a crane, having long legs, will protest you with tears if you try to cut them with a knife. -- Zhuangzi

Suppose we have an extremely detailed world model in which every object is characterized by many features -- in this example, material and height, though one could add more. Suppose further that we would like to generate descriptions of these things automatically for the most part, drawing the player's attention only to those aspects of the object that are particularly interesting.

```
"The Crane's Leg"

Material is a kind of value. The materials are wood, glass, stone, cloth, paper,
clay, and metal. A thing has a material.

A height is a kind of value. 3 feet 11 inches specifies a height. A thing has a
height. Definition: a thing is tall if its height is 6 feet 0 inches or more. Definition:
a thing is short if its height is 2 feet 0 inches or less.
```

So far, we have generally dealt with cases where the property of a thing can be a number (such as 3), a value (such as brightness), or a unit (like height, here). It is also possible for a thing to have a property which names another thing, as in "The mother of the baby trout is the large trout" -- where "mother" is a property, and its value, in the case of the baby trout, is large trout. We would define such a property with a line such as "A fish has a thing called the mother."

In practice, though, this is a bit confusing as syntax; moreover, Inform has a much more powerful construct for talking about the ways in which one object relates to another object. A full discussion of this will have to wait for the chapter on Relations. For now, it is enough to say that we can do this:

```
Imitation relates various things to one thing (called the ideal). The verb to imitate means the imitation relation.
```

This will allow us to declare that some objects imitate other objects, like so:

```
A chair is a kind of supporter. A chair is always enterable. A chair is usually wood. The height of a chair is usually 4 feet 0 inches. The ordinary chair is a chair. Every chair imitates the ordinary chair.

A table is a kind of supporter. A table is usually wood. The height of a table is usually 3 feet 8 inches. The ordinary table is a table. Every table imitates the ordinary table.

A rock is a kind of thing. A rock is usually stone. The ordinary rock is a rock. Every rock imitates the ordinary rock. The height of a rock is usually 0 feet 3 inches.

A jug is a kind of container. A jug is usually clay. The ordinary jug is a jug. Every jug imitates the ordinary jug. The height of a jug is usually 0 feet 8 inches.
```

Now each of these types has one ideal representative which has the fundamental attributes of its kind: the ordinary chair is the most chairlike chair imaginable, the ordinary table is the epitome of tableness, and so on. We are also allowed to refer to "the ideal of the chair", thanks to the way we defined imitation. (Again, the relations chapter offers a much more detailed explanation of how relations may be defined.)

```
The description of a thing is usually "[comparison with ideal][run paragraph on]".

To say comparison with ideal:
    let the sample be the ideal of the noun;
    if the sample is not a thing:
        say "Perfectly conforming to your expectations of its type.";
        rule succeeds;
    if the material of the noun is not the material of the sample:
        if the height of the noun is not the height of the sample:
            if the noun is shorter than the sample, say "Unusually short at [height of the noun], and made of [material of the noun].";
            otherwise say "Unusually tall at [height of the noun], and made of [material of the noun].";
        otherwise:
            say "Distinct mostly in being made of [material of the noun].";
```

```
 otherwise:
     if the height of the noun is not the height of the sample:
         if the noun is shorter than the sample, say "Unusually short at [height of
the noun].";
         otherwise say "Unusually tall at [height of the noun].";
     otherwise:
         say "In every respect [a sample]."

The Pleasure Garden is a room. "At the riverbank, a pleasing garden, having
many curving paths and one straight."

The low table is a table in the Pleasure Garden. The height of the low table is 2
feet 3 inches. On the low table is a metal rock called a gold nugget.

Test me with "x table / x nugget".
```

So far the effect is not very deep, but we could take the auto-description a great deal further: providing a larger and more interesting set of variations; or writing a complicated set of rules such that the player only notices height variations when carrying a ruler; or switching between several player-characters, each of whom notices a different subset of characteristics. But these refinements would require more input from later chapters.

---

53 ★★★ Example Signs and Portents

RB

Signpost that points to various destinations, depending on how the player has turned it.

```
"Signs and Portents"

Seven Dials is a room. The description of Seven Dials is "There is a signpost, on
which seven hands swivel and swing, freely as weathercocks. They make your
present road now London, now Abingdon; now Weston-super-Mare, or now
Hell."

Seven Dials contains a signpost. The signpost is scenery. Understand "sign" and
"post" as the signpost.

Destination is a kind of value. The destinations are London, Abingdon, Luton,
Weston-super-Mare, Runnymede, Hell, and Low Noon.

The signpost has a destination.
```

In order to interact with the signpost, we will need to make use of some action rules:

```
Instead of examining the signpost:
    say "[The signpost] currently puts you on the road to [italic type][the
destination of the signpost][roman type], but it swiftly alters again.";
    now the destination of the signpost is the destination after the destination of
the signpost.
```

```
Instead of turning the signpost:
    now the destination of the signpost is the destination after the destination of the signpost;
    say "With a hand's touch you turn the signpost to mark your way for [italic type][the destination of the signpost][roman type]."

Instead of going north in Seven Dials when the destination of the signpost is Hell:
    say "It is a path that goes gently ever down and down with no stumbling block or any distraction at either side; there are no bandits and no tolls.";
    end the story.

Instead of going north in Seven Dials when the destination of the signpost is Low Noon:
    say "A long road whose scenery does not change, nor anything on the horizon move but the sun. When at last you come to Noon, she hangs above your head like a hat.";
    end the story finally.

Test me with "x signpost / n / turn signpost / n / turn signpost / n / turn signpost / n / turn signpost / n".

Test more with "x signpost / n / turn signpost / n / turn signpost / n / turn signpost / n / turn signpost / turn signpost / n".
```

---

54 ★★★ **Example** **Real Adventurers Need No Help** *RB*

Allowing the player to turn off all access to hints for the duration of a game, in order to avoid the temptation to rely on them overmuch.

Suppose we have an action called "asking for help" that gives the player some hints on request. We've also made it possible to turn this feature off, if the player would like to discourage himself from using the hints too much. Now we need a value that varies to keep track of whether hints are currently permitted or currently not permitted. So we might write:

```
"Real Adventurers Need No Help"

A permission is a kind of value. The permissions are allowed and denied.

Hint usage is a permission that varies. Hint usage is allowed.
```

And under the right circumstances, we change hint usage to denied:

```
Check asking for help for the first time:
    say "Sometimes the temptation to rely on hints becomes overwhelming, and you may prefer to turn off hints now. If you do so, your further requests for guidance will be unavailing. Turn off hints? >";
    if player consents:
        now hint usage is denied;
        say "[line break]Truly, a real adventurer does not need hints." instead.
```

Then we can refer back to this value later to decide whether we want to display the hint menu or not:

```
Check asking for help:
    if hint usage is denied, say "You have chosen to eschew hints in this game. Be strong! Persevere!" instead.

Asking for help is an action out of world. Understand "help" or "hint" or "hints" as asking for help.

The Realm of Terribly Unjust Puzzles is a room.

Carry out asking for help:
    say "Fine, since you're weak enough to ask: here is a complete walkthrough: GET EGG. PEEL EGG. SMELL EGG. DIVIDE YOLK INTO THREE PORTIONS. GIVE THE SMALLEST PORTION OF YOLK TO THE GOLDEN GOOSE. ASK THE GOOSE ABOUT WHETHER THE SWAN IS TO BE TRUSTED. GIVE THE LARGEST PORTION OF YOLK TO THE SWAN. DANCE CONGA. EAT MEDIUM PORTION. STAND ON HEAD. WEST."

Test me with "hint".
```

Note that it would probably be kinder to offer the player some intermediate level of help, in the actual event.

---

55 ★★★ **Example** **Early Childhood** *RB*

A child's set of building blocks, which come in three different colours - red, green and blue - but which can be repainted during play.

This would be a one-star example if it were not for the repainting:

```
"Early Childhood 1"

A building block is a kind of thing. A red block, a blue block and a green block are kinds of building block.

The Nursery is a room. In the Nursery are six red blocks, four blue blocks and a green block.

Test me with "look / get red block".
```

But a kind cannot change during play, so this will not do. Instead, the colour will have to be a property of the block. So we might first try this:

```
"Early Childhood 2"

Colour is a kind of value. The colours are red, blue and green. A block is a kind of thing. A block has a colour. A block is usually blue.
```

```
The Nursery is a room. In the Nursery are six red blocks, four blue blocks and a
green block.

Test me with "look / get red block".
```

Which is fine, so far as it goes, but the colour property is not at all visible to the player, who simply sees "eleven blocks". We thought of colour as being something outwardly apparent, but Inform does not know this. To achieve a better effect, we will need features from distant chapters. The first is an activity called "printing the name of":

```
"Early Childhood 3"

Colour is a kind of value. The colours are red, blue and green. A block is a kind
of thing. A block has a colour. A block is usually blue. Before printing the name
of a block: say "[colour] ". Before printing the plural name of a block: say "
[colour] ".

The Nursery is a room. In the Nursery are six red blocks, four blue blocks and a
green block.

Test me with "look / get red block".
```

This too, however, is unsatisfactory. The individual blocks are correctly described, but we are unable to distinguish them during play: we cannot type "take a green block", for instance. And because the blocks are indistinguishable in play, they are still massed together as "eleven blocks" in room descriptions. We need to go one step further:

```
"Early Childhood 4"

Colour is a kind of value. The colours are red, blue and green. A block is a kind
of thing. A block has a colour. A block is usually blue. Before printing the name
of a block: say "[colour] ". Before printing the plural name of a block: say "
[colour] ". Understand the colour property as describing a block.

The Nursery is a room. In the Nursery are six red blocks, four blue blocks and a
green block.
```

And now everything works nicely: the blocks are grouped by colour, and can be referred to by colour, and we can even change the colour of an individual block during play, using a bit of extra trickery from later:

```
Understand "paint [something] [colour]" as painting it. Painting it is an action
applying to one thing and one colour. Check painting it: if the noun is not a block,
say "Paints are only for blocks." instead. Carry out painting it: now the colour of
the noun is the colour understood. Report painting it: say "The block is now [the
colour of the noun]."

Test me with "get red block / get blue block / g / i / look / paint blue block red / i /
look / paint me red".
```

56 ★ Example **Being Prepared** RB

A kind for jackets, which always includes a container called a pocket.

```
"Being Prepared"

A jacket is a kind of thing. A jacket is always wearable.

A pocket is a kind of container. A pocket is part of every jacket. The carrying
capacity of a pocket is always 2.

After examining a jacket:
    let target be a random pocket which is part of the noun;
    say "[The target] contains [a list of things in the target]."
```

Now we've created the rules that will govern any specific jackets we might happen to put in our game: each one will always have one pocket, which will be able to contain no more than two things. The description of "a list of things" is text with a list, which we will learn about further in a few sections.

Next we might want to create the environment and an actual example of the jacket kind:

```
Tent is a room. "A dome made of two flexible rods and a lot of bright green
ripstop nylon. It bills itself as a one-man tent, but you'd call it a two-dwarf tent:
there is no way to arrange yourself on its square floor so that you can stretch out
completely."

The hoodie is a jacket. "Your hoodie is balled up in the corner." The description
of the hoodie is "Both elbows are stained from yesterday's entrenching project."

The hoodie's pocket contains a Swiss army knife and a folded map. The hoodie
is in the Tent.
```

Notice that, since Inform has created a pocket for the hoodie, we can now refer to it by name in our source, giving it any additional properties we need to define. Here we simply put a few items into it.

```
The player wears a whistle. The description of the whistle is "To frighten bears."

Test me with "x hoodie / get hoodie / get knife / get map / i / put hoodie in pocket
/ put whistle in pocket / put map in pocket / put knife in pocket / i".
```

Notice that Inform automatically refuses to put the hoodie into its own pocket: as a default, a container cannot contain something of which it is itself a part.

57 ★★★ Example **Model Shop** RB

An "on/off button" which controls whatever device it is part of.

Suppose we're particularly mechanically-minded and would like a game in which all of our mechanical devices have buttons to turn them on and off.

```
"Model Shop"

An on/off button is a kind of thing.

Instead of pushing an on/off button which is part of a switched off device (called the machine):
    try switching on the machine.
```

Here we are making a rule about how our hypothetical buttons will interact with the machines to which they belong. Instead of pushing... is a rule that pertains to actions, and we will learn more about these in the chapter on actions. "...which is part of a switched off device" provides a specific circumstance - this is only to apply to buttons that are stuck to a machines that can be turned on or off. "(called the machine)" tells Inform that if it finds such a device, it should thereafter refer to it as "the machine." (The called syntax is explained further in the chapter on Change.)

A set of three more rules will complete our instructions about using buttons to control devices:

```
Instead of pushing an on/off button which is part of a switched on device (called the machine):
    try switching off the machine.

Instead of switching on an on/off button which is part of a device (called the machine):
    try switching on the machine.

Instead of switching off an on/off button which is part of a device (called the machine):
    try switching off the machine.
```

Then we hand out buttons with a free hand:

```
One on/off button is part of every device.

The Model Shop is a room. A model train is a fixed in place device in the Model Shop. A toy elephant is a device in the Model Shop.

Every turn when the model train is switched on:
    say "The model train circles your feet, blowing small puffs of steam."

Every turn when the toy elephant is switched on:
    say "The toy elephant waves its trunk at you."

Test me with "push model train's button / push elephant's button / g / switch off model train's button".
```

And now the game will have a model train's button and a toy elephant's button.

It may be that we want (as an added nuance) to add other names for these items. While we would want an assembly to create objects such as "Lucy's hand" and not "Lucy hand", it is entirely reasonable to want to talk about the model train button or the elephant button. We could define these additional names like so:

```
Understand "elephant button" or "button on elephant" as the elephant's button.

Understand "model train" or "model" or "train" as "[train]". Understand "[train]
button" or "button on [train]" as the model train's button.
```

In the second case, we are defining [train] to mean any of the three phrases "train", "model", and "model train"; so "[train] button" will match "model train button" or "train button" or "model button" equally well. See the chapter on Understanding for more on how to create alternative phrasings for the player to use.

---

58 ★★★ **Example U-Stor-It** *RB*

A "chest" kind which consists of a container which has a lid as a supporter.

Suppose we want to write a game in which there are a number of chests. Each of these chests will be a container, but have a lid which is a supporter.

```
"U-Stor-It"

Section 1 - Assemblies and Supporters

A chest is a kind of container. A chest is always openable. A chest is usually
fixed in place. A chest is usually closed. The specification of a chest is
"Represents a container with a separately implemented lid; the lid is itself a
supporter."

A lid is a kind of supporter. A lid is part of every chest. The specification of a lid is
"A supporter attached to a chest, which can only support things when the chest
is closed."
```

(The "specification" of a kind is not really a property, and is used instead to describe the kind in the Index. So the text of these specifications is never found in the game.) Of course, this doesn't get us very far. We will also want the game to correctly interpret variations on "open the chest" and "close the lid", redirecting actions appropriately.

```
Section 2 - Opening and closing

Before opening a lid which is part of a chest (called the item):
    try opening the item instead.

Before closing a lid which is part of a chest (called the item):
    try closing the item instead.
```

```
Before opening a chest (called the box) when something is on a lid (called the obstruction) which is part of the box:
    repeat with item running through things on the obstruction:
        say "(first removing [the item])";
        try taking the item.

Instead of opening a chest when something is on a lid (called the item) which is part of the noun:
    say "You'd have to remove [the list of things on the item] from the lid first." instead.

Instead of looking under a lid which is part of a chest (called the item):
    try opening the item.
```

We may also want to be able to deal with "put in" and "put on" appropriately, even if the player names the wrong part of the object:

```
Section 3 - Insertion and Support

Before inserting something into a lid which is part of a chest (called the item):
    try inserting the noun into the item instead.

Before putting something on a chest when a lid (called the item) is part of the second noun:
    try putting the noun on the item instead.
```

Furthermore, we don't want the player to be able to put things on the lid while the chest is open:

```
Before putting something on a lid which is part of an open chest (called the item):
    say "(first closing [the item])";
    try closing the item.

Instead of putting something on a lid which is part of an open chest (called the item):
    say "[The item] would need to be closed first."
```

And then we may also want a couple of rules for describing our assembled object nicely:

```
Section 4 - Description in Rooms

Instead of examining a closed chest when something is on a lid (called the top) which is part of the noun:
    say "[The noun] is closed, and there [is-are a list of things on the top] on top."

After printing the name of a chest (called the item) while listing contents of a room:
    if a lid (called the second item) which supports something is part of the item:
        say " (on which [is-are a list of things on the second item])";
        omit contents in listing.
```

Now we are free to create entire treasure rooms at a single blow:

```
Section 5 - U-Stor-It Facility

The U-Stor-It Facility is a room. The sea trunk, the shipping crate, and a metal box are chests in the U-Stor-It Facility. The metal box contains a sapphire, a gold coin, and a signed photograph of Babe Ruth.
```

Even though we have never explicitly defined it, the metal box has a "metal box's lid", which we can use at need.

```
The metal box's lid supports a small card. The description of the small card is "It reads, 'Back in 5 mins - Pandora.'"

Test me with "open trunk / x card / open metal box / put all in metal box / get card / put card on box".
```

---

59 ★★★ **Example** **The Night Before** RB

Instructing Inform to prefer different interpretations of EXAMINE NOSE, depending on whether the player is alone, in company, or with Rudolph the Red-nosed Reindeer.

Suppose that we're going to give every person in the game a nose, but we want references to a nose always to mean the nose of someone *else*, if the player is with one other person. Moreover, on some occasions we're going to be in sight of Rudolph, so actions directed at an unspecified nose should always prefer his.

This relies on a somewhat advanced technique from the Understanding chapter, but since it may become useful with assemblies and body parts, it is worth mentioning here.

```
"The Night Before"

The North Pole is a room. "Here it is: the famous Pole. From here you can go south (or south-south, or south-south-by-south); or, alternatively, take refuge inside a red-and-white-striped cabin." The cabin is scenery in the North Pole. Instead of entering the cabin, try going inside.

Santa is a man in the North Pole. "Santa is pacing around in the snow and trying to psych himself up for the big night."

Inside from North Pole is the Candy Cane Cabin. The description of the Cabin is "Striped red and white, but nothing can make this place seem warm and inviting since Mrs. Santa ran off with the Tooth Fairy."

The Ice Shelf is south of North Pole. "The ice here has been smoothed into a kind of runway for easy take-off, and ends in a cliff and cold arctic sea." Donner, Vixen, Blixen, and Rudolph are animals in the Ice Shelf.
```

```
A nose is a kind of thing. A nose is part of every person. The description of Santa's nose is "It's a bit ruddy. You don't like to mention it, but Santa's been dipping heavily into the Grey Goose since Mrs. Santa left town." The description of a nose is usually "Not terribly exciting." The description of Rudolph's nose is "See how it glows!"
```

Next, we'll teach Inform some vocabulary to distinguish between the player and everyone else:

```
Definition: a person is other if it is not the player.

Definition: a thing is selfish if it is part of the player and the player can see an other person.

Instead of examining a selfish nose:
    say "You cross your eyes, but can't get a good look."
```

Here is the part that actually determines the preferences. "Does the player mean..." can result in five outcomes: "it is very unlikely", "it is unlikely", "it is possible" (the neutral default), "it is likely", and "it is very likely". This is discussed in greater detail in the Understanding chapter. Here, we want to discourage references to the player's own nose and encourage references to the nose of Rudolph, so:

```
Does the player mean doing something when the noun is a selfish nose or the second noun is a selfish nose: it is very unlikely.

Does the player mean doing something to Rudolph's nose: it is very likely.
```

And this part is just for decoration:

```
Rule for writing a paragraph about Rudolph:
    say "The reindeer are already harnessed and waiting impatiently. The brilliance of [Rudolph]'s nose casts an eerie red glow over [the list of unmentioned animals in the location]."

Test me with "x nose / x my nose / x santa's nose / in / x nose / out / s / x my nose / x nose / x rudolph's nose / x donner's nose".
```

Chapter 5: Text

Contents of *Writing with Inform*
Chapter 4: Kinds
Chapter 6: Descriptions
Indexes of the examples

§5.1. Text with substitutions

In the previous chapter, we gave properties to certain kinds of things in order to change their appearance and behaviour, and saw brief glimpses of one of Inform's most useful devices: text substitution. The following gives a more complete example:

```
"The Undertomb"

A dead end is a kind of room with printed name "Dead End" and description "This is a dead end. You'll have to go back the way you came, consoled only by [river sound]." A dead end is usually dark.

The Undertomb is a dark room. East is a dead end. South is a dead end with printed name "Collapsed Dead End". Northwest is a dead end called the Tortuous Alcove. In the Undertomb is the lantern. It is lit.

A dead end has some text called river sound. The river sound of a dead end is usually "a faint whispering of running water". The Tortuous Alcove has river sound "a gurgle of running water".
```

The novelty here is the text in square brackets in the first paragraph. They imply more or less what they would when a journalist is quoting something in a newspaper article. The actual words "river sound" are not part of the text. Instead, when Inform prints up the description of a dead end, it will substitute the appropriate river sound in place of these words.

Thus the description of the Collapsed Dead End is "This is a dead end. You'll have to go back the way you came, consoled only by a faint whispering of running water.", whereas the description of the Tortuous Alcove is "This is a dead end. You'll have to go back the way you came, consoled only by a gurgle of running water." As the player explores these dead ends, subtle differences will appear in their room descriptions.

Start of Chapter 5: Text

Back to Chapter 4: Kinds: §4.17. Postscript on simulation

Onward to §5.2. How Inform reads quoted text

§5.2. How Inform reads quoted text

Text is so fundamental to Inform that the basics had to be covered back in Chapter 2, so let's begin this new chapter with a recap.

Literal text is written in double-quotation marks. It's mostly true that what you see is what you get: the literal text "The Hands of the Silversmith" means just

```
The Hands of the Silversmith
```

But four characters are read in unexpected ways: [,], ' and ". The rules are as follows:

Exception 1. Square brackets [and] are used to describe what Inform should say, but in a non-literal way. For example,

```
"Your watch reads [time of day]."
```

might produce

```
Your watch reads 9:02 AM.
```

These are called "text substitutions". They're highly flexible, and they can take many different forms. But as useful as they are, they do seem to stop us from making actual [and] characters come through on screen. To get around that:

say "[bracket]"

This text substitution expands to a single open square bracket, avoiding the problem that a literal [in text would look to Inform like the opening of a substitution. Example:

```
"He [bracket]Lord Astor[close bracket] would, wouldn't he?"
```

prints as "He [Lord Astor] would, wouldn't he?".

say "[close bracket]"

This text substitution expands to a single close square bracket, avoiding the problem that a literal] in text would look to Inform like the closing of a substitution. Example:

```
"He [bracket]Lord Astor[close bracket] would, wouldn't he?"
```

prints as "He [Lord Astor] would, wouldn't he?".

Exception 2. Single quotation marks at the edges of words are printed as double. So:

```
"Simon says, 'It's far too heavy to lift.'"
```

produces

```
Simon says, "It's far too heavy to lift."
```

This is good because typing a double quotation mark inside the quote wouldn't work - it would end the text then and there. Single quotation marks inside words, such as the one in "it's", remain apostrophes.

The rule looks odd at first, but turns out to be very practical. The only problem arises if we need an apostrophe at the start or end of a word, or a double inside one. Again, substitutions can fix this:

say "[apostrophe/']"

This text substitution expands to a single quotation mark, avoiding Inform's ordinary rule of converting literal single quotation marks to double at the edges of words. Example:

```
Instead of going outside, say "Lucy snaps, 'What's the matter? You don't trust
my cookin[apostrophe] mister?'"
```

produces:

```
Lucy snaps, "What's the matter? You don't trust my cookin' mister?"
```

A more abbreviated form would be:

```
Instead of going outside, say "Lucy snaps, 'What's the matter? You don't trust
my cookin['] mister?'"
```

which has exactly the same meaning.

say "[quotation mark]"

This text substitution expands to a double quotation mark. Most of the time this is unnecessary because of Inform's rule of converting literal single quotation marks to double at the edges of words, so it's needed only if we want a double-quote in the middle of a word for some reason. Example:

```
"The compass reads 41o21'23[quotation mark]E."
```

which produces: The compass reads 41o21'23"E. (Note that ["] is not allowed; a double-quotation mark is never allowed inside double-quoted text, not even in a text substitution.)

Exception 3. Texts which end with sentence-ending punctuation - full stop, question mark, exclamation mark - are printed with a line break after them. So:

```
say "i don't know how this ends";
say "I know just how this ends!";
```

would come out quite differently - this doesn't affect the appearance of the text, but only the position where the next text will appear. Again, sometimes this is not what we want - the full rules are complicated enough to be worth a whole section later in the chapter.

Start of Chapter 5: Text

Back to §5.1. Text with substitutions

Onward to §5.3. Text which names things

§5.3. Text which names things

We can put almost any description of a value in square brackets in text, and Inform will work out what kind of value it is and print something accordingly. (Only almost any, because we aren't allowed to use commas or more quotation marks inside a square-bracketed substitution.)

say "[(sayable value)**]"**

This text substitution takes the value and produces a textual representation of it. Most kinds of value, and really all of the useful ones, are "sayable" - numbers, times, objects, rules, scenes, and so on. Example:

```
The description of the wrist watch is "The dial reads [time of day]."
```

Here "time of day" is a value - it's a time that varies, and time is a sayable kind of value, so we might get "The dial reads 11:03 AM."

The values we say most often are objects. If we simply put the name of what we want into square brackets, this will be substituted by the full printed name. We might find:

```
"You admire [lantern]."
= "You admire candle lantern."
```

But this reads oddly - clearly "the" or "a" is missing. So the following substitutions are used very often:

say "[a (object)**]"**

or:

say "[an (object)**]"**

This text substitution produces the name of the object along with its indefinite article. Example:

```
Instead of examining something (called the whatever):
    "You can only just make out [a whatever]."
```

which might produce "You can only just make out a lamp-post.", or "You can only just make out Trevor.", or "You can only just make out some soldiers." The "a" or "an" in the wording is replaced by whatever indefinite article applies, if any.

say "[A (object)**]"**

or:

say "[An (object)**]"**

This text substitution produces the name of the object along with its indefinite article, capitalised. Example:

```
Instead of examining something (called the whatever):
    "[A whatever] can be made out in the mist."
```

which might produce "A lamp-post can be made out in the mist.", or "Trevor can be made out in the mist.", or "Some soldiers can be made out in the mist." The "A" or "An" in the wording is replaced by whatever indefinite article applies, if any.

say "[the (object)**]"**

This text substitution produces the name of the object along with its definite article. Example:

```
Instead of examining something (called the whatever):
    "You can only just make out [the whatever]."
```

which might produce "You can only just make out the lamp-post.", or "You can only just make out Trevor.", or "You can only just make out the soldiers." The "the" in the wording is replaced by whatever definite article applies, if any.

say "[The (object)**]"**

This text substitution produces the name of the object along with its definite article, capitalised. Example:

```
Instead of examining something (called the whatever):
    "[The whatever] may be a trick of the mist."
```

which might produce "The lamp-post may be a trick of the mist.", or "Trevor may be a trick of the mist.", or "The soldiers may be a trick of the mist." The "The" in the wording is replaced by whatever definite article applies, if any.

This may not look very useful, because why not simply put "the", or whatever, into the ordinary text? The answer is that there are times when we do not know in advance which object will be involved. For instance, as we shall later see, there is a special value called "the noun" which is the thing to which the player's current command is applied (thus, if the player typed TAKE BALL, it will be the ball). So:

```
After taking something in the Classroom:
    "You find [a noun]."
```

might produce replies like "You find a solid rubber ball.", "You find an ink-stained blouse.", "You find some elastic bands.", or even "You find Mr Polycarp." (the school's pet hamster, perhaps).

⬆ Start of Chapter 5: Text

⬅ Back to §5.2. How Inform reads quoted text

➡ Onward to §5.4. Text with numbers

§5.4. Text with numbers

When a numerical value is given in a square-bracketed substitution, it is ordinarily printed out in digits. Thus:

```
"You've been wandering around for [turn count] turns now."
```

might print as "You've been wandering around for 213 turns now.", if the story has been played out for exactly that many commands. But if we prefer:

say "[(number) **in words]"**

This text substitution writes out the number in English text. Example:

```
"You've been wandering around for [turn count in words] turns now."
```

might produce "You've been wandering around for two hundred and thirteen turns now." The "and" here is natural on one side of the Atlantic but not the other - so with the "Use American dialect." option in place, it disappears.

Either way, though, there is some risk of the following:

```
You've been wandering around for one turns now.
```

We can avoid this using the special substitution:

say "[s]"

This text substitution prints a letter "s" unless the last number printed was 1. Example:

```
"You've been wandering around for [turn count in words] turn[s] now."
```

produces "... for one turn now." or "... for two turns now." as appropriate. Note that it reacts only to numbers, not to other arithmetic values like times (or, for instance, weights from the "Metric Units" extension).

This only solves one case, but it's memorable, and the case is one which turns up often.

⬆ Start of Chapter 5: Text

⬅ Back to §5.3. Text which names things

➡ Onward to §5.5. Text with lists

⬇ Example 60: ★★★ **Ballpark** A new "to say" definition which allows the author to say "[a number in round numbers]" and get verbal descriptions like "a couple of" or "a few" as a result.

§5.5. Text with lists

We often want running text to include lists of items.

say "[list of (description of objects)**]"**

This text substitution produces a list, in sentence form, of everything matching the description. Example:

```
"Mr Darcy glares proudly at you. He is wearing [list of things worn by Darcy]
and carrying [list of things carried by Darcy]."
```

And, if this were from a dramatisation of the novel by Miss Fielding rather than Miss Austen, we might find:

> Mr Darcy glares proudly at you. He is wearing a pair of Newcastle United boxer shorts and carrying a self-help book.

If the description matches nothing - for instance, if Darcy has empty hands - then "nothing" is printed.

As with all lists in Inform, the serial comma is only used if the "Use serial comma." option is in force. So by default we would get "a fishing pole, a hook and a sinker", rather than "a fishing pole, a hook, and a sinker".

We then need variations to add indefinite or definite articles, and to capitalise the first item. For example,

> "Mr Darcy impatiently bundles [the list of things carried by Darcy] into your hands and stomps out of the room."

might result in

> Mr Darcy impatiently bundles the self-help book and the Christmas card into your hands and stomps out of the room.

say "[a list of (description of objects)**]"**

This text substitution produces a list, in sentence form, of everything matching the description. Each item is prefaced by its indefinite article. Example:

> a maritime bill of lading, some hemp rope and Falconer's Naval Dictionary

say "[A list of (description of objects)**]"**

This text substitution produces a list, in sentence form, of everything matching the description. Each item is prefaced by its indefinite article, and the first is capitalised, so that it can be used at the beginning of a sentence. Example:

> A maritime bill of lading, some hemp rope and Falconer's Naval Dictionary

say "[the list of (description of objects)**]"**

This text substitution produces a list, in sentence form, of everything matching the description. Each item is prefaced by its definite article. Example:

the maritime bill of lading, the hemp rope and Falconer's Naval Dictionary

say "[The list of (description of objects)**]"**

This text substitution produces a list, in sentence form, of everything matching the description. Each item is prefaced by its definite article, and the first is capitalised, so that it can be used at the beginning of a sentence. Example:

The maritime bill of lading, the hemp rope and Falconer's Naval Dictionary

So much for articles. A more insidious problem comes with something like this:

"The places you can go are [list of rooms]."

The trouble is that the list may end up either singular or plural. We might be expecting something like:

The places you can go are Old Bailey, Bridget's Flat and TV Centre.

But if there is only one room, then the result might be:

The places you can go are Bridget's Flat.

which is wrong. We can get around this with careful wording and a slightly different substitution:

"Nearby [is-are list of rooms]."

say "[is-are list of (description of objects)**]"**

This text substitution produces a list, in sentence form, of everything matching the description. The whole list starts with "is" (if there's one item or none) or "are" (more than one). Examples:

is marlin-spike
are maritime bill of lading, hemp rope and Falconer's Naval Dictionary

say "[is-are a list of (description of objects)**]"**

This text substitution produces a list, in sentence form, of everything matching the description. Each item is prefaced by its indefinite article, and the whole list starts with "is" (if there's one item or none) or "are" (more than one). Examples:

is a marlin-spike
are a maritime bill of lading, some hemp rope and Falconer's Naval Dictionary

say "[is-are the list of (description of objects)**]"**

This text substitution produces a list, in sentence form, of everything matching the description. Each item is prefaced by its definite article, and the whole list starts with "is" (if there's one item or none) or "are" (more than one). Examples:

is the marlin-spike
are the maritime bill of lading, the hemp rope and Falconer's Naval Dictionary

say "[a list of (description of objects) **including contents]"**

This text substitution produces a list, in sentence form, of everything matching the description, noting any contents in brackets. This is really intended only to be used by the Standard Rules.

Start of Chapter 5: Text

Back to §5.4. Text with numbers

Onward to §5.6. Text with variations

Example 61: ★ **Control Center** Objects which automatically include a description of their component parts whenever they are examined.

Example 62: ★★ **Tiny Garden** A lawn made up of several rooms, with part of the description written automatically.

§5.6. Text with variations

Text sometimes needs to take different forms in different circumstances. Perhaps it needs an extra sentence if something has happened, or perhaps only one altered word.

say "[if (a condition)**]"**

This text substitution produces no text. It's used only for a side-effect: it says that the text following should be said only if the condition is true. That continues until the end of the text, or until an "[end if]" substitution, whichever comes first. If the "[otherwise]" and "[otherwise if]" substitutions are also present, they allow alternatives to be added in case the condition is false. Example:

```
The wine cask is a container. The printed name of the cask is "[if
open]broached, empty cask[otherwise]sealed wine cask".
```

we find that the cask is described as "a broached, empty cask" when open, and "a sealed wine cask" when closed. A longer example which begins and ends with fixed text, but has two alternatives in the middle:

```
The Customs Wharf is a room. "Amid the bustle of the quayside, [if the cask is
open]many eyes stray to your broached cask. [otherwise]nobody takes much
notice of a man heaving a cask about. [end if]Sleek gondolas jostle at the
plank pier."
```

say "[unless (a condition)**]"**

This text substitution produces no text. It's used only for a side-effect: it says that the text following should be said only if the condition is false. That continues until the end of the text, or until an "[end if]" substitution, whichever comes first. If the "[otherwise]" and "[otherwise if]" substitutions are also present, they allow alternatives to be added in case the condition is true. Example:

```
The Customs Hall is a room. "With infinite slowness, with ledgers and quill
pens, the clerks ruin their eyesight.[unless the player is a woman] They barely
even glance in your direction."
```

say "[otherwise]"

or:

say "[else]"

This text substitution produces no text, and can be used only following an "[if ...]" or "[unless ...]" text substitution. It switches from text which appears if the condition is true, to text which appears if it is false. Example:

```
The wine cask is a container. The printed name of the cask is "[if
open]broached, empty cask[otherwise]sealed wine cask".
```

say "[end if]"

This text substitution produces no text, and can be used only to close off a stretch of varying text which begins with "[if ...]".

say "[end unless]"

This text substitution produces no text, and can be used only to close off a stretch of varying text which begins with "[unless ...]".

say "[otherwise/else **if** (a condition)**]"**

This text substitution produces no text, and can be used only following an "[if ...]" or "[unless ...]" text substitution. It gives an alternative text to use if the first condition didn't apply, but this one does. Example:

```
The wine cask is a container. The printed name of the cask is "[if open]broached, empty cask[otherwise if transparent]sealed cask half-full of sloshing wine[otherwise]sealed wine cask".
```

say "[otherwise/else **unless** (a condition)**]"**

This text substitution produces no text, and can be used only following an "[if ...]" or "[unless ...]" text substitution. It gives an alternative text to use if the first condition didn't apply, and this one is false too.

We sometimes need to be careful about the printing of line breaks:

```
The Cell is a room. "Ah, [if unvisited]the unknown cell. [otherwise]the usual cell."
```

This room description has two possible forms: "Ah, the unknown cell. ", at first sight, and then "Ah, the usual cell." subsequently. But the second form is rounded off with a line break because the last thing printed is a ".", whereas the first form isn't, because it ended with a space. The right thing would have been:

```
The Cell is a room. "Ah, [if unvisited]the unknown cell.[otherwise]the usual cell."
```

allowing no space after "unknown cell."

Start of Chapter 5: Text

Back to §5.5. Text with lists

Onward to §5.7. Text with random alternatives

Example 63: ★ **When?** A door whose description says "...leads east" in one place and "...leads west" in the other.

Example 64: ★★★ **Whence?** A kind of door that always automatically describes the direction it opens and what lies on the far side (if that other room has been visited).

Example 65: ★★★ **Persephone** Separate the player's inventory listing into two parts, so that it says "you are carrying..." and then (if the player is wearing anything) "You are also wearing...".

§5.7. Text with random alternatives

Sometimes we would like to provide a little quirky variation in text, especially in messages which will be seen often. We can achieve this with the "[one of]... [or] ... [or] ..." construction.

say "[one of]"

This text substitution produces no text. It's used only for a side-effect: it switches between a number of alternative texts, which follow it and are divided by "[or]" substitutions, according to a strategy given in a closing substitution. Example:

```
"You flip the coin. [one of]Heads[or]Tails[purely at random]!"
```

Here there are just two alternatives, and the strategy is "purely at random". Exactly half of the time the text will be printed as "You flip the coin. Heads!"; and the other half, "You flip the coin. Tails!".

say "[or]"

This text substitution produces no text, and can be used only in a "[one of]..." construction. It divides alternative wordings. Example:

```
"You flip the coin. [one of]Heads[or]Tails[purely at random]!"
```

There are seven possible endings, each making the choice of which text to follow in a different way:

say "[purely at random]"

This text substitution produces no text, and can be used only to end a "[one of]..." construction. It indicates that the alternatives are chosen uniformly randomly.

say "[then purely at random]"

This text substitution produces no text, and can be used only to end a "[one of]..." construction. It indicates that the alternatives are chosen in sequence until all have been seen, but that after that they are chosen uniformly randomly.

say "[at random]"

This text substitution produces no text, and can be used only to end a "[one of]..." construction. It indicates that the alternatives are chosen at random except that the same choice cannot come up twice running. This is useful to avoid the deadening effect of repeating the exact same message. Example:

```
"The light changes randomly again; now it's [one of]green[or]amber[or]red[at random]."
```

Here we can safely say the light "changes", because the new colour cannot be the same as the one printed the last time.

say "[then at random]"

This text substitution produces no text, and can be used only to end a "[one of]..." construction. It indicates that the alternatives are chosen in sequence until all have been seen, and then after that, at random except that the same choice cannot come up twice running. Example:

```
"Maybe the murderer is [one of]Colonel Mustard[or]Professor Plum[or]Cardinal Cerise[then at random]."
```

say "[sticky random]"

This text substitution produces no text, and can be used only to end a "[one of]..." construction. It indicates that a random choice is made the first time the text is printed, but that it sticks from there on. Example:

```
"The newspaper headline is: [one of]War Casualties[or]Terrorists[or]Banks[sticky random] [one of]Continue To Expand[or]Lose Out[sticky random]."
```

Although the newspaper headline will change with each playing, it will not alter during play.

say "[as decreasingly likely outcomes]"

This text substitution produces no text, and can be used only to end a "[one of]..." construction. It indicates that the alternatives are chosen at random, except that the first is most likely to be chosen, the second is next most likely, and so on down to the rarest at the end. Example:

```
"Zorro strides by, [one of]looking purposeful[or]grim-faced[or]deep in thought[or]suppressing a yawn[or]scratching his ribs[or]trying to conceal that he has cut himself shaving[as decreasingly likely outcomes]."
```

There are six outcomes here: the first is six times as likely as the last, and those in between are similarly scaled, so Zorro cuts himself shaving only once in 21 tries, while he looks purposeful almost a third of the time.

But suppose we want to tuck some useful information in these messages, and we want to be sure that the player will see it. Because all of the above options involve randomness, it's possible that an unlucky player might miss a clue placed into only one variant of the message. One fix for this is to make sure that everything turns up sooner or later:

say "[in random order]"

This text substitution produces no text, and can be used only to end a "[one of]..." construction. A random order is chosen for the alternative passages of text, and they are used in that order as the text is printed again and again. When one random cycle finishes, a new one begins. The effect is somewhat like the "shuffle album" feature on an iPod. Example:

```
"You dip into the chapter on [one of]freshwater fish[or]hairless mammals[or]extinct birds[or]amphibians such as the black salamander[in random order]."
```

One small restriction: if there are more than 32 variations, purely random choices will be printed, and there will be no guarantee that repeats are prevented.

Another fix is to avoid randomness altogether:

say "[cycling]"

This text substitution produces no text, and can be used only to end a "[one of]..." construction. It indicates that the alternatives are used one at a time, in turn: after the last one is reached, we start again from the first. Example:

```
"The pundits discuss [one of]the weather[or]world events[or]celebrity gossip[cycling]."
```

say "[stopping]"

This text substitution produces no text, and can be used only to end a "[one of]..." construction. It indicates that the alternatives are used one at a time, in turn: once the last one is reached, it's used forever after. Example:

> "[one of]The phone rings[or]The phone rings a second time[or]The phone rings again[stopping]."

Finally, here's a convenient shorthand for one of the commonest things needed:

say "[first time]"

or:
say "[only]"

This pair of text substitutions causes whatever is between them to be printed only the first time the text is printed. Example:

> "The screen door squeaks loudly as when you open it. [first time]Well, you'll get used to it eventually. [only]"

This is exactly equivalent to

> "The screen door squeaks loudly as when you open it. [one of]Well, you'll get used to it eventually. [or][stopping]";

but easier to read.

Start of Chapter 5: Text

Back to §5.6. Text with variations

Onward to §5.8. Line breaks and paragraph breaks

Example 66: ★ **Radio Daze** A radio that produces a cycle of output using varying text.

Example 67: ★★ **Camp Bethel** Creating characters who change their behavior from turn to turn, and a survey of other common uses for alternative texts.

§5.8. Line breaks and paragraph breaks

Inform controls the flow of text being said so that it will read, to the player, in a natural way. There are two principles:

(a) pieces of text ending with full stop, exclamation or question marks will be followed by line breaks (or "new lines", as some computer programming languages would call them); and

(b) pieces of text produced by different rules in Inform will be separated by paragraph breaks.

The effect is that authors can forget about paragraph spacing most of the time, but the mechanism is not impossible to fool, so text substitutions are provided to override the usual principles. First, to manipulate line breaks:

say "[line break]"

This text substitution produces a line break. Example:

```
"There is an endless sense of[line break]falling and[line break]falling."
```

Line breaks are not paragraph breaks, so the result is:

```
There is an endless sense of
falling and
falling.
```

with no extra vertical spacing between these lines.

say "[no line break]"

This text substitution produces no text. It's used only for a side-effect: it prevents a line break where Inform might otherwise assume one. Example:

```
"The chorus sing [one of]Jerusalem[or]Rule, Britannia![no line break][at
random]."
```

Here the "[no line break]" stops Inform from thinking that the exclamation mark means a sentence ending - it's part of the name of the song "Rule, Britannia!". So we get

```
The chorus sing Rule, Britannia!.
```

with no line break between the "!" and ".".

And similarly for paragraph breaks. Because Inform can be pretty trigger-happy with these, the first need is for a way to stop them:

say "[run paragraph on]"

This text substitution produces no text. It's used only for a side-effect: it prevents a paragraph break occurring after the present text is printed, in case Inform might be tempted to place one there. Example:

```
Before taking something, say "Very well. [run paragraph on]".
```

This allows the reply to, say, TAKE ENVELOPE to be

```
Very well. Taken.
```

rather than

```
Very well.

Taken.
```

which is how texts produced by different rules would normally be shown. (It's a traditional printer's term. See Oldfield's Manual of Typography, 1892, under "When two paragraphs are required to be made into one, or, in technical language, 'to run on'.")

But sometimes we actually want paragraph breaks in unexpected places. One way is to force them outright:

say "[paragraph break]"

This text substitution produces a paragraph break. Example:

```
"This is not right.[paragraph break]No, something is terribly wrong."
```

Paragraph breaks have a little vertical spacing in them, unlike mere line breaks, so the result is:

```
This is not right.

No, something is terribly wrong.
```

More subtly, we can give Inform the option:

say "[conditional paragraph break]"

This text substitution either produces a paragraph break, or no text at all. It marks a place where Inform can put a paragraph break if necessary; in effect it simulates what Inform does every time a "before" or similar rule finishes. If there is text already printed, and text then follows on, a paragraph break is made. But if not, nothing is done. This is sometimes useful when producing a large amount of text which changes with the circumstances so that it is hard to predict in advance whether a paragraph break is needed or not.

Really finicky authors might possibly want to know this:

if a paragraph break is pending:

This condition is true if text has recently been said in such a way that Inform expects to add a paragraph break at the next opportunity (for instance when the present rule ends and another one says something, or when a "[conditional paragraph break]" is made).

Finally, there are two special sorts of paragraph break for special circumstances. They are mainly used by the Standard Rules, and imitate the textual layout styles of traditional IF.

say "[command clarification break]"

This text substitution produces a line break, and then also a paragraph break if the text immediately following is a room description brought about by having gone to to a different room and looking around, in which case a line break should be added. In traditional IF, this is used when clarifying what Inform thinks the player intended by a given command. Example:

```
say "(first opening [the noun])[command clarification break]";
```

might result in

```
(first opening the valise)
You rummage through the valise for tickets, but find nothing.
```

say "[run paragraph on with special look spacing]"

This text substitution produces no text. It's used only for a side-effect: it indicates that the current printing position does not follow a skipped line, and that further material is expected which will run on from the previous paragraph, but that if no further material turns up then a skipped line would be needed before the next command prompt. (It's very likely that only the Standard Rules will ever need this.)

⬆ Start of Chapter 5: Text

⬅ Back to §5.7. Text with random alternatives

➡ Onward to §5.9. Text with type styles

⬇ Example 68: ★★ **Beekeeper's Apprentice** Making the SEARCH command examine all the scenery in the current location.

§5.9. Text with type styles

Inform does not go in for the use of fonts: a work of IF will be rendered with different fonts on different machines anyway, from tiny personal organisers up to huge workstations. However, it does allow for a modest amount of styling.

say "[bold type]"

This text substitution produces no text. It's used only for a side-effect: to make the text following it appear in bold face. "[roman type]" should be used to switch back to normal. Example:

```
"Jane looked down. [bold type]Danger[roman type], the sign read."
```

say "[italic type]"

This text substitution produces no text. It's used only for a side-effect: to make the text following it appear in italics. "[roman type]" should be used to switch back to normal. Example:

```
"This is [italic type]very suspicious[roman type], said Peter."
```

say "[roman type]"

This text substitution produces no text. It's used only for a side-effect: to return to ordinary Roman type after a previous use of "[bold type]" or "[italic type]".

but there is one other effect we can employ:

say "[fixed letter spacing]"

This text substitution produces no text. It's used only for a side-effect: to make the text following it appear with fixed letter spacing. In variable letter spacing, a lower case "m" is much wider than an "l", which is natural to the eye since it has been printing practice since the Renaissance. Fixed letter spacing is more like typewriting, and it is best used to reproduce typewritten text or printed notices; it can also be convenient for making simple diagrams. Example:

```
"On the door is written: [fixed letter spacing]J45--O-O-O[variable letter spacing]."
```

say "[variable letter spacing]"

This text substitution produces no text. It's used only for a side-effect: to return to ordinary letter spacing after a previous use of "[fixed letter spacing]".

Whichever effect we use, we should be careful to ensure that we return to normal -- roman type and variable letter spacing -- after any specially-treated text has been printed. Combining these effects (for, say, bold fixed-spaced lettering) is not guaranteed to work, though on some platforms it will.

Start of Chapter 5: Text

Back to §5.8. Line breaks and paragraph breaks

Onward to §5.10. Accented letters

Example 69: ★ **Garibaldi 2** Adding coloured text to the example of door-status readouts.

§5.10. Accented letters

Inform 7 is infused by the English language, so it's a challenge using it to write a work of IF in any other language. (With that said, extensions do exist which have made considerable progress on this problem: nil desperandum.) But even a book in English contains occasional quotations or words borrowed from other tongues, so we are going to need more than plain A to Z.

The world has a bewildering range of letters, accents, diacritics, markers and signs. Inform tries to support the widest range possible, but the works of IF produced by Inform are programs which then have to be run on a (virtual) computer whose abilities are more constrained: few players will have an Ethiopian font installed, after all. So a degree of caution is called for.

(a) **Definitely safe to use.** Inform's highest level of support is for the letters found on a typical English typewriter keyboard, including both the $ and £ signs (but not the Yen or Euro symbols ¥ and €), and in addition the following:

```
ä, á, à, ã, å, â and Ä, Á, À, Ã, Å, Â
ë, é, è, ê and Ë, É, È, Ê
ï, í, ì, î and Ï, Í, Ì, Î
ö, ó, ò, õ, ø, ô and Ö, Ó, Ò, Õ, Ø, Ô
```

```
ü, ú, ù, û and Ü, Ú, Ù, Û
ÿ, ý and Ý (but not Ÿ)
ñ and Ñ
ç and Ç
æ and Æ (but not œ or Œ)
ß
¡, ¿
```

These characters can be typed directly into the Source panel, and can be used outside quotation marks: we can call a room the Église, for instance.

(b) **Characters which can safely be used, but will be simplified.** As it reads in the text, Inform silently converts all kinds of dash (en-rules, em-rules, etc.) to simple hyphens; converts the multiplication symbol to a lower case "x"; converts all kinds of space other than tabs (em-spaces, non-breaking spaces, etc.) to simple spaces, and all kinds of quotation marks to "straight" (non-smart) marks.

(c) **Characters which can be used provided they are in quoted text (other than boxed quotations), and which will probably but not certainly be visible to the player**. All other Latin letter-forms, including the œ ligature, East European forms such as ő, ş and ž, and Portuguese forms such as ũ; the Greek and Cyrillic alphabets, with their associated variants and accents; and the principal currency symbols, such as € and ¥. Such characters are not legal in unquoted text: so we could write

```
The Churchyard is a room. The printed name of the Churchyard is "Łodz Churchyard".
```

but not

```
Łodz Churchyard is a room.
```

Moreover, the player is not allowed to type these characters in commands during play: or rather, they will not be recognised if he does. They are for printing only.

(d) **Characters which might work in quoted text, or might not**. The Arabic and Hebrew alphabets are fairly likely to be available; miscellaneous symbols are sometimes legible to the player, sometimes not. Other alphabets are chancier still. (If a work of IF depends on these being visible, it may be necessary to instruct players to use specific interpreters, or to provide a way for the player to test that all will be well.)

Start of Chapter 5: Text

Back to §5.9. Text with type styles

Onward to §5.11. Unicode characters

§5.11. Unicode characters

As we have seen, Inform allows us to type a wide range of characters into the source text, although the more exotic ones may only appear inside quotation marks. But they become more and more difficult to type as they become more obscure. Inform therefore allows us to describe a letter using a text substitution rather than typing it directly.

Unicode characters can be named (or numbered) directly in text. For example:

```
"[unicode 321]odz Churchyard"
```

produces a Polish slashed L. If the Unicode Character Names or Unicode Full Character Names extensions are included, characters can also be named as well as numbered:

```
"[unicode Latin capital letter L with stroke]odz Churchyard"
```

The Unicode standard assigns character numbers to essentially every marking used in text from any human language: its full range is enormous. (Note that Inform writes these numbers in decimal: many reference charts show them in hexadecimal, or base 16, which can cause confusion.) Inform can only handle codes [unicode 32] up to [unicode 65535], so it is not quite so catholic, but the range is still enormous enough that code numbers are unfamiliar to the eye. Inform therefore allows us to use the official Unicode 4.1 names for characters, instead of their decimal numbers, *provided* we have Included the necessary extension like so:

```
Include Unicode Character Names by Graham Nelson.
```

This extension provides names for some 2900 of the most commonly used characters. It means, for instance, that we can write text such as:

```
"Dr Zarkov unveils the new [unicode Hebrew letter alef] Nought drive."
"Omar plays 4[unicode black spade suit] with an air of triumph."
```

Admittedly, these can get a little verbose:

```
"[unicode Greek small letter omega with psili and perispomeni and ypogegrammeni]"
```

But before getting carried away, we should remember the hazards: Inform allows us to type, say, "[unicode Saturn]" (an astrological sign) but it appears only as a black square if the resulting story is played by an interpreter using a font which lacks the relevant sign. For instance, Zoom for OS X uses the Lucida Grande and Apple Symbol fonts by default, and this combination does contain the Saturn sign: but Windows Frotz tends to use the Tahoma font by default, which does not. (Another issue is that the fixed letter spacing font, such as used in the status line, may not contain all the characters that the font of the main text contains.) To write something with truly outré characters is therefore a little chancy: users would have to be told quite carefully what interpreter and font to use to play it.

The "Unicode Character Names" extension, which is pre-installed in the standard distribution of Inform, defines names for the Latin, Greek, Cyrillic, Hebrew and Braille alphabets, together with currency and miscellaneous other symbols, including some for drawing boxes and arrows. It is only optionally installed because even this is quite large: but in case it should still prove inadequate, an alternative can be used:

```
Include Unicode Full Character Names by Graham Nelson.
```

This includes all 12,997 named characters in the 16-bit range of the Unicode 4.1 standard: it is the size of a small novel and its inclusion will slow Inform down. But if you want to experiment with Arabic, ecclesiastical Georgian, Cherokee, Tibetan, Syriac, the International

Phonetic Alphabet, hexagrams or the unified Canadian aboriginal syllabics, "Unicode Full Character Names" (again built into Inform) is the extension for you.

Start of Chapter 5: Text

Back to §5.10. Accented letters

Onward to §5.12. Displaying quotations

Example 70: ★★★ **The Über-complète clavier** This example provides a fairly stringent test of exotic lettering.

§5.12. Displaying quotations

Text is normally printed in between the typed commands of the player, rolling upwards from the bottom of the screen, as if a dialogue is being typed by an old-fashioned teletype. But it can also be displayed in a bolder way, floating above the main text, and this is sometimes used to display quotations.

display the boxed quotation (text)

This phrase displays the given text on screen in an overlaid box. For reasons to do with the way such quotations are plotted onto the screen, their text is treated literally: no substitutions in square brackets are obeyed. The quotation will only ever appear once, regardless of the number of times the "display the boxed quotation ..." phrase is reached. Rather than being shown immediately - and thus, probably, scrolling away before it can be seen - the display is held back until the next command prompt is shown to the player. Example:

```
After looking in the Wabe, display the boxed quotation
    "And 'the wabe' is the grass-plot round
    a sun-dial, I suppose? said Alice,
    surprised at her own ingenuity.

    Of course it is. It's called 'wabe,'
    you know, because it goes a long way
    before it, and a long way behind it --

    -- Lewis Carroll".
```

This was the original example used in *Trinity*, by Brian Moriarty, which invented the idea. A player exploring Kensington Gardens comes upon a location enigmatically called The Wabe; and by way of explanation, this quotation pops up.

Note that exotic accented characters, such as the "Ł" in "Łodz", can't be displayed in boxed quotations. This is only a simple feature, and we should go in search of a suitable extension for fancier screen effects if we would like to do more.

Start of Chapter 5: Text

Back to §5.11. Unicode characters

Onward to §5.13. Making new substitutions

§5.13. Making new substitutions

If we have some textual effect which needs to occur in several different messages, we might want to create a new text substitution for it. For instance:

```
The Missile Base is a room. "[security notice]Seems to be a futuristic missile base." M's Office is east of the Missile Base. "[security notice]Admiral Sir M.- M.- glares up from his desk."

To say security notice:
    say "This area is a Prohibited Place within the meaning of the Official Secrets Act 1939. "
```

This is only the tip of the iceberg in how to define ways to do things using "To...", as we shall see. The definition makes "say the security notice" a new phrase known to Inform. A text substitution is exactly a phrase whose name begins with "say" (well - except for the "say" phrase itself), so the effect is that "[security notice]" is a new text substitution. Several of the examples in this chapter make use of this trick.

Inform often ignores the casing of the text it reads, but sometimes uses it as a clue to meaning. We have already seen that "[an item]" and "[An item]" produce different results, for instance. Similarly, it's possible to define two text substitutions which are the same except for the initial casing. We might write:

```
To say Security Notice:
    say "THIS AREA IS A PROHIBITED PLACE WITHIN THE MEANING OF THE OFFICIAL SECRETS ACT 1939. "
```

And now Inform will act on "[Security Notice]" differently from "[security notice]".

★ *See* **The phrasebook** *for other forms of phrase besides To say...*

Start of Chapter 5: Text

Back to §5.12. Displaying quotations

Onward to Chapter 6: Descriptions: §6.1. What are descriptions?

Example 71: ★ **Fifty Ways to Leave Your Larva** Using text substitution to make characters reply differently under the same circumstances.

Example 72: ★★★ **Fifty Times Fifty Ways** Writing your own rules for how to carry out substitutions.

Examples from Chapter 5: Text

Start of this chapter

Chapter 6: Descriptions

Indexes of the examples

60 ★★★ **Example Ballpark** RB

A new "to say" definition which allows the author to say "[a number in round numbers]" and get verbal descriptions like "a couple of" or "a few" as a result.

Sometimes it is more sensible to describe numbers roughly than in exact terms. For instance, we might want to have our player perceive "many people" rather than "forty-two people" on entering a room. To achieve this, we might write our own "to say" phrase.

```
"Ballpark"

To say (count - a number) in round numbers:
    repeat through the Table of Numerical Approximation:
        if count is less than threshold entry:
            say "[approximation entry]";
            rule succeeds.
```

Phrases will be explained more thoroughly in a later chapter, but as we have already seen in the examples, we can make a "To say..." phrase that will allow us to create our own text substitutions. In this case, we are going to replace the specific number with a vaguer one chosen from a chart, so:

```
Table of Numerical Approximation

    threshold approximation
    1         "no"
    2         "one"
    3         "a couple of"
    6         "a few"
    11        "some"
    21        "many"
    1000      "lots and lots of"
```

The idea here is that we will work our way through the table until we hit a line where the threshold number is higher than the number we want to express, and then print that output: so if we have less than one item, we'll print "no"; if we have more than none but less than two, we'll print "one"; if we have less than three, we'll print "a couple of"; if we have three, four, or five (but not six), we'll print "a few."

```
A room has a number called the population. The population of a room is usually
0. The description of a room is usually "You observe [population of the location in
round numbers] [if population of the location is 1]person [otherwise]people [end
if]here.".

The Stadium is a room. The Hot Dog Stand is west of the Stadium. The
Women's Restroom is south of the Stadium.
```

```
The population of the Stadium is 500. The population of the Hot Dog Stand is 3.
The population of the Restroom is 750.

Test me with "w / e / s".
```

61

★ Example Control Center

RB

Objects which automatically include a description of their component parts whenever they are examined.

It is straightforward to make a rule that anything with parts must mention all those parts during an EXAMINE command:

```
"Control Center"

After examining a thing when something is part of the noun:
    say "[The noun] includes [a list of things which are part of the noun]."

The Control Center is a room. "Here you are at the Control Center of the
universe."

The Universe Management Computer is a fixed in place thing in the Control
Center. "The Universe Management Computer sits directly before you,
unguarded." The description of the Universe Management Computer is "The
computer is so large that you would be unable to operate it all from one position.
Alas, it does not come with a manual."

A chartreuse indicator light, an ennui meter, a golden knob settable to 15,000
positions, a toothpick dispenser, and a button labeled RESTART are part of the
Universe Management Computer.

The command chair is an enterable supporter in the Control Center. It is
pushable between rooms. "Because the computer is too large for you to reach
all of the front panel from a standing position, there is a command chair on
casters which allows you to push back and forth." The description of the
command chair is "Quite ordinary, really, but for the heady rush of power that
comes of sitting in it.". Some casters are part of the command chair.
```

Now whenever we look at any object with components, we will first see the description, then a list of parts which belong to it. The following refinement brings in elements of later chapters, but it may be worth noting: because we've written our rule as an "After examining...", anything that pre-empts the operation of the examine command will also prevent that rule from occurring. So for instance:

```
A hair-thick needle is part of the ennui meter.

Instead of examining the ennui meter: say "You can't be bothered."
```

...would not result in the needle being mentioned.

```
Test me with "x chair / x computer / x ennui meter"
```

62

★★★ Example Tiny Garden

RB

A lawn made up of several rooms, with part of the description written automatically.

Sometimes we want to make a list of something too complicated to express in a say list... phrase. When this happens, we can instead mark all the items we want to mention as "marked for listing".

In this case, we have a lawn area made up of four rooms. We want each room to automatically describe the directions leading to the other parts of the lawn. To do this, we will first determine which directions are relevant and mark those for listing, then list them.

```
"Tiny Garden"

The Herb Garden is a room. "Along this side of the house run your great-aunt's herb beds."

A Grassy Room is a kind of room. The printed name of a Grassy Room is usually "Lawn". The description of a Grassy Room is "The grass underfoot is thick and green. The lawn extends to [grassy directions] from here."
```

The following phrase goes through all the directions in the compass and marks the ones that are interesting to us at the moment.

```
To say grassy directions:
    repeat with that way running through directions:
        if the room that way from the location is a grassy room,
            now that way is marked for listing;
    say "[a list of directions which are marked for listing]";
    now every direction is not marked for listing.

Lawn1 is west of the Herb Garden. It contains a picnic table and a wicker basket. Lawn2 is south of Lawn1 and southeast of Lawn4. Lawn3 is southwest of Lawn1, west of Lawn2, and south of Lawn4. Lawn4 is west of Lawn1. Lawn4 contains a birdbath. The birdbath is fixed in place.

Lawn1, Lawn2, Lawn3, and Lawn4 are Grassy Rooms.

Test me with "w / s / w / n".
```

63

★ Example When?

RB

A door whose description says "...leads east" in one place and "...leads west" in the other.

Very simple, but quite frequently useful:

```
"When?"

The temporal vortex is an open door. It is west of Yesterday and east of Today. "A whirling temporal vortex leads [if the player is in Yesterday]west[otherwise]east[end if]."
```

64 Example Whence?

RB

A kind of door that always automatically describes the direction it opens and what lies on the far side (if that other room has been visited).

It would be fairly tedious reading to have a large game full of doors that describe themselves this way. Nonetheless, if we insisted we could use our knowledge of the map as leverage to make every door in the game describe itself automatically.

To do this, we make use of the phrase "direction of (the door) from (a room)" -- in this case, the direction of the door we're looking at when viewed from the player's location. Thus:

```
"Whence?"

The temporal vortex is an open door. It is west of Yesterday and east of Today.

The initial appearance of a door is usually "Nearby [an item described] leads [if the other side of the item described is visited][direction of the item described from the location] to [the other side][otherwise][direction of the item described from the location][end if]."

Test me with "w / e".
```

Yet a further variation on this, which can automatically understand "the east door" and "the west door" when appropriate, may be found in the example "Whither?".

65 Example Persephone

RB

Separate the player's inventory listing into two parts, so that it says "you are carrying..." and then (if the player is wearing anything) "You are also wearing...".

If we wanted, we might replace the rule for taking inventory as follows:

```
"Persephone"

Instead of taking inventory:
    say "[if the player carries something][We]['re] carrying [a list of things carried by the player][otherwise][We]['re] empty-handed";
```

```
	say "[if the player wears something]. [We]['re] wearing [a list of things worn by the player][end if]."

The Fancy Party is a room. The player carries a sword, a strawberry stem, and 20 credits worth of platinum. The player wears a sash indicating lordhood.

Test me with "i / take off sash / i / drop all / i".
```

---

66 ★ **Example Radio Daze** RB

A radio that produces a cycle of output using varying text.

```
"Radio Daze" by Jon Ingold

The Living Room is a room. "A long couch, set up so you can see your wireless set. Not that you need to see it, of course."

The long couch is an enterable scenery supporter in the living room. Instead of entering the long couch when the radio was switched off, say "Better turn the radio on before you get comfortable." Report entering the couch: say "You settle yourself down to listen." instead. Instead of listening when the radio is switched on, stop the action.

The radio is a device in the living room. The radio is switched off. "[if switched on]The radio burbles on[otherwise]The radio is off[end if]." Check switching off the radio when the player is on the long couch: say "You can't reach the radio from here." instead.

Every turn when the radio is switched on:
	say "[one of]Two characters in the radio play have begun an argument[or]The argument continues[or]The play continues[stopping]: [one of]'Did not!'[or]'Did too!'[or]'Did I?'[or]'You did!'[or]'I couldn't have, Martha!'[or]'But you did, Tom!'[cycling]"

Test me with "sit on couch / turn on radio / sit on couch/ listen / g / g / g / g / g".
```

---

67 ★★★ **Example Camp Bethel** RB

Creating characters who change their behavior from turn to turn, and a survey of other common uses for alternative texts.

```
"Camp Bethel"

Camp Bethel Kitchen is a room.
```

One use for text alternatives is to change the description of a room after first visiting. We've already seen, in the example "Slightly Wrong", how to do this with "[if visited] ... [otherwise] ... [end if]". But since the first description is printed once and the second description on all subsequent occasions, we could equally well write

```
The description of Camp Kitchen is "[one of]You've never been into the kitchen
before, though you've spent many an hour in the dining lodge. The place is
larger than you would have expected, and it has none of the fake rustic touches
of the rest of the camp[or]A tidy, efficient industrial kitchen, without any of the
kitsch rusticity found elsewhere[stopping]."
```

We might also want to liven up the behavior of people and animals, who are probably not doing the exact same thing every time we glance in their direction. There are more complex techniques for modeling the behavior of characters, as we will see in the chapters on Advanced Actions and Activities; but if we just want some textual variety, we might write something like:

```
Jeremy is a man in the Camp Bethel Kitchen. "Jeremy stands at his station, [one
of]peeling white onions[or]briskly dicing onions[or]chopping celery[or]peeling
carrots[or]tying fresh herbs together with string[or]putting all the vegetables into
a large stock pot[or]watching over his boiling vegetable stock[cycling]."
```

And since (textual variation or not) we do want the player to be able to see all these objects:

```
Jeremy carries white onions, celery, carrots, and herbs. Jeremy's station is
scenery in the kitchen. It is a supporter.
```

Jeremy is following a sequence of actions to do an implied task (still somewhat robotically, but it will do for now). Animals might be a bit more capricious, though:

```
Fluffy is an animal in the Camp Bethel Kitchen. "[one of]Fluffy is chasing its
tail[or]Fluffy is staring out the window[or]Fluffy is rubbing itself against your
leg[purely at random]."

A housefly is an animal in the Camp Bethel Kitchen. "A large housefly [one
of]lands on a countertop[or]flies around noisily[or]circles Jeremy's chef hat[at
random]."
```

The housefly's description is merely "at random" rather than "purely at random" because we want to show it doing a different thing each turn, whereas Fluffy could plausibly stare out the window for five turns in a row.

There are more complex ways to change and override the initial descriptions of people and things; if text alternatives do not get us far enough, we can turn to the "rule for writing a paragraph about," documented in the Activities chapter.

Another frequent use of text alternatives is to give characters a bit of variety in things they're likely to say many times in the course of a game:

```
Instead of telling Jeremy about something:
    say "Jeremy looks [one of]surprised[or]intrigued[or]nonplussed[at random].
'[one of]You don't say[or]That's very interesting[or]Do go on[or]I wish I'd known
that sooner[at random]!'".
```

Or, with somewhat more complexity:

```
Instead of asking Jeremy about something:
    say "'[one of]Sorry,[or]I'm afraid[or]Hm,[at random] [one of]I don't know much about that[or]you've got me there[or]I haven't the faintest[at random],' Jeremy [one of]drawls[or]replies[or]comments[or]exclaims[at random]";
    say "[one of][or] huskily[or] throatily[or] silkily[or] in a deep manly voice[as decreasingly likely outcomes]."
```

Notice that, in that last line, our first option is entirely blank. If we put nothing as an element of the text alternatives list, this means that printing nothing at all is a viable alternative. In fact, we've made this the most common probability out of the decreasingly likely outcomes, so that five times in fifteen, or a third of the time the text is run, there will be no modifier printed at all.

```
Test me with "look / g / g / g / ask Jeremy about his feelings for me / ask jeremy about his amnesia / tell Jeremy about my unborn child".
```

As this example (alas) reveals, text alternatives will not go all the way toward making our characters into compelling conversationalists; we will have to wait until we know more about Actions. But at least we have abolished the default responses, and given Jeremy a touch of personality, however witless.

---

68 ★★★ **Example** **Beekeeper's Apprentice** *RB*

Making the SEARCH command examine all the scenery in the current location.

We have to create a suitable action and say what it does, and to repeat what we do through all the scenery items. That needs material from subsequent chapters, but is quite ordinary Inform all the same:

```
"Beekeeper's Apprentice"

Studying the vicinity is an action applying to nothing.

Report studying the vicinity:
    if the location does not contain something which is scenery:
        say "There's little of interest in the [location]." instead;
    repeat with point of interest running through scenery in the location:
        say "[point of interest]: [run paragraph on]";
        try examining the point of interest.

Understand "search" as studying the vicinity.

The Yard is a room.

The hive and the honey are scenery things in the Yard. The description of the hive is "The honeycombed hive is all around you, thrumming with life." The description of the honey is "Wax-sealed honey has been cached in many of the hexagonal nurseries."
```

```
Test me with "search".
```

The reason for this example is to show the use of saying "[run paragraph on]". It means we have output such as:

```
>search
hive: The honeycombed hive is all around you, thrumming with life.

honey: Wax-sealed honey has been cached in many of the hexagonal nurseries.
```

Without the running on, the prompts "hive:" and "honey:" would be separated from the descriptions following them, which would look a little odd.

---

69 ★ **Example Garibaldi 2** RB

Adding coloured text to the example of door-status readouts.

The extension "Basic Screen Effects" provides a few more type styles, in the form of coloured lettering. The colours available are red, yellow, green, blue, white, magenta, and cyan, as well as the usual black; and to restore the player's default screen colour, we say "default letters".

Thus if we wanted to highlight locked and unlocked doors in our security readout example:

```
"Garibaldi"

Include Basic Screen Effects by Emily Short.

The security readout is a device. The description of the readout is "The screen is blank."

Instead of examining the switched on security readout:
    say "The screen reads: [fixed letter spacing]";
    say line break;
    repeat with item running through doors:
        say line break;
        say " [item] ([front side of the item]/[back side of the item]): [if the item is locked][green letters]LOCKED[default letters][otherwise][red letters]UNLOCKED[default letters][end if]";
    say variable letter spacing;
    say paragraph break.

The player carries the security readout.

The Docking Bay is a room. The inner airlock is a door. It is north of the Docking Bay and south of the Zocalo. The inner airlock is lockable and unlocked. The outer airlock is lockable and locked. It is a door. It is south of the Docking Bay and north of Space.
The quarantine seal is a door. It is west of the Zocalo and east of Medlab.
Quarantine seal is locked.

The security pass unlocks the inner airlock. The player carries the security pass.

Test me with "x readout / turn on readout / x readout / lock inner airlock with
security pass / x readout".
```

Note that this extension does not currently produce the desired effects when compiling with the Glulx setting; to see it working, make sure that the settings tab is set to compile to the Z-machine.

---

70 ★★★★ **Example** **The Über-complète clavier** RB

This example provides a fairly stringent test of exotic lettering.

The following example puts Inform's support for exotic lettering through its paces. It was useful in testing Inform but is not a very instructive read: still, it does provide a test story file for interpreters, so we are including the source here as an example.

```
"The Über-complète clavier"

The story headline is "Pushing the Limits of Unicode in IF". The story description
is "This is a demanding test for Unicode compliance by Z-machine interpreters."

Include Unicode Character Names by Graham Nelson.

Include Basic Screen Effects by Emily Short.

The Château Bibliothèque Français is east of the Deutsche Universität Bücherei.
"From this Borgesian construction, doorways lead into anterooms in each of the
four cardinal directions." South of the Bibliothèque is the Miscellany Mañana.
North of the Bibliothèque is the Íslendingabók. East of the Bibliothèque is
Alphabet Soup.

A framed photograph of Icelandic Prime Minister Halldór Ásgrímsson, a ruler
measuring Ångströms, a Bokmål-Lëtzebuergesch Lëtzebuergesch-Bokmål
dictionary and a ticket to Tromsø via Østfold are in the Íslendingabók.

A paper by Karl Weierstraß, a general feeling of Ärger, an old Österreich
passport and the Bach Clavier-Übung open at the fugue à 4 are in the Bücherei.

The painting of École normale superiéure students singing Ça ira, the frankly
lesser-known journal of Niccolò Polo, Così fan tutte on CD, an extract of
Herodotus concerning Artaÿctes and the exit sign reading À BIENTÔT are in the
Bibliothèque.

A wicker basket marked CHLOË is in the Bibliothèque. A ginger cat is in the
basket.
```

A guide to Æsop for naïve æsthetes, Lönnrot's Kalevala, a creed according to the Bahá'ís, FALARÃO magazine, an Estonian poem by Tõnu Trubetsky, a Portuguese-Italian recipe for macarrão, a stripy hanging CANDY PIÑATA bag, a ¿¡Punctuation Turned Upside Down¿¡ pamphlet, an Italian brewers' anti-violence poster declaring BÓTTE NON BÒTTE, a map of È and a dusty book titled The Parnasum of Luís Vaz bearing CAMÕES on its spine are in Miscellany Mañana.

The description of the map is "È is a province in the People's Republic of China."

In Mañana is something called ÂÊÎÔÛ - The Official Journal of the Society for Vowels bearing Circumflexes.

In Mañana is something called âêîôû comic - the youth edition.

The description of Alphabet Soup is "A bewildering place of glyphs, sigils and signs. The Library proper leads back west: steps lead upwards to an Observatory, or downwards into what seems to be a dangerous area. A gaming lounge lies to the south."

The Greek Alphabet, the Cyrillic Alphabet, the Hebrew alphabet, and the embossed plaque are in Alphabet Soup. The description of the Greek alphabet is "αβγδεζηθικλμνξοπρςστυφχψω.". The description of the Hebrew alphabet is "אבגדהוזחטיךכלםמןנסעףפץצקרשת.". The description of the Cyrillic alphabet is "абвгдежзийклмнопрстуфхцчшщъыьэюя.".

```
Instead of examining the plaque:
    say "It seems to be a sign in Braille: ";
    say unicode Braille pattern dots-24, " (I), ",
        unicode Braille pattern dots-1345, " (N), ",
        unicode Braille pattern dots-124, " (F), ",
        unicode Braille pattern dots-135, " (O), ",
        unicode Braille pattern dots-1235, " (R), ",
        unicode Braille pattern dots-134, " (M)."
```

The Gaming Lounge is south of Alphabet Soup. The chess position and the book of puzzle canons are in the Gaming Lounge.

The Georges de la Tour painting Le Tricheur is in the Gaming Lounge. "Hanging on one wall is Georges de la Tour's masterpiece Le Tricheur (the card-sharp). Visible are 8[unicode black diamond suit], 9[unicode black diamond suit], A[unicode black diamond suit], A[unicode black spade suit], 6[unicode black club suit] but not one of them has a [unicode black heart suit]."

The description of Le Tricheur is "If they'd been dice-players instead, they might have thrown [unicode die face-1], [unicode die face-2], [unicode die face-3], [unicode die face-4], [unicode die face-5] or [unicode die face-6], but as it is they stick to cards."

The description of the book of canons is "A typical fugue is no. 13 (Tovey: [unicode eighth note] = 110) in F[unicode music sharp sign] minor, but you can also make out keys like A[unicode music flat sign] and G[unicode music natural sign]."

The empty square text is text that varies. To say empty: say the empty square text.

```
To display the board:
	say empty, empty, empty, empty, empty, empty, unicode black chess king, empty, line break;
	say empty, empty, empty, unicode black chess queen, empty, empty, unicode black chess pawn, empty, line break;
	say unicode black chess pawn, empty, empty, unicode black chess bishop, unicode black chess pawn, empty, empty, unicode black chess pawn, line break;
	say empty, empty, empty, unicode black chess pawn, empty, unicode black chess rook, empty, empty, line break;
	say empty, unicode black chess pawn, empty, unicode white chess pawn, unicode black chess pawn, empty, empty, empty, line break;
	say empty, empty, empty, unicode black chess bishop, unicode white chess queen, empty, unicode white chess pawn, unicode white chess pawn, line break;
	say unicode white chess pawn, unicode white chess pawn, empty, unicode white chess bishop, empty, unicode black chess rook, unicode white chess bishop, empty, line break;
	say empty, unicode white chess knight, empty, empty, unicode white chess rook, empty, unicode white chess rook, unicode white chess king, line break.

Instead of examining the chess position:
	say "Fritz Saemisch - Aron Nimzowitsch, Copenhagen 1923: the Immortal Zugzwang Game. Nimzowitsch (black), observing that white will very soon have to play a terrible move, has just advanced his h pawn for no reason other than to wait. So it is white to play...";
	say "[fixed letter spacing]......k. [line break]...q..p. [line break]p..bp..p [line break]...p.r.. [line break].p.Pp... [line break]...bQ.PP [line break]PP.B.rB. [line break].N..R.RK [variable letter spacing][line break]";
	say "'White must, willy-nilly, eventually throw himself upon the sword', in Nimzowitsch's commentary. ";
	say "We will now try to display the same position using chess-piece symbols in a Unicode font.";
	say fixed letter spacing;
	now the empty square text is " ";
	display the board;
	say variable letter spacing.

The Astrological Observatory is above Alphabet Soup.

The planets are in the Observatory. "Diagrams of the planets are scattered across the dome: Sun [unicode Sun], Mercury [unicode Mercury], Venus [unicode Female Sign], Earth [unicode Earth], Moon [unicode First Quarter Moon] and [unicode Last Quarter Moon], Mars [unicode Male Sign], Jupiter [unicode Jupiter], Saturn [unicode Saturn], Uranus [unicode Uranus], Neptune [unicode Neptune], Pluto [unicode Pluto] and one or two comets [unicode Comet]. Fainter, but all around, you see stars black [unicode black star] and white [unicode white star]."

The constellations are in the Observatory. "Ringing the dome are the constellations: Aries [unicode Aries], Taurus [unicode Taurus], Gemini [unicode Gemini], Cancer [unicode Cancer], Leo [unicode Leo], Virgo [unicode Virgo], Libra [unicode Libra], Scorpius [unicode Scorpius], Sagittarius [unicode Sagittarius], Capricorn [unicode Capricorn], Aquarius [unicode Aquarius], Pisces [unicode Pisces]."

The weather almanac is in the Observatory. The description of the almanac is "Here nightly observers scrawl in hasty abbreviations for the current weather
```

```
conditions: clear weather [unicode Black Sun with Rays], cloudy [unicode cloud],
rain [unicode umbrella], snow [unicode snowman], lightning [unicode lightning],
thunderstorm [unicode thunderstorm]."

The Danger Zone is below Alphabet Soup. The printed name of the Danger
Zone is "[unicode skull and crossbones] Danger Zone [unicode skull and
crossbones]".

The warning signs are in the Danger Zone. "A variety of international-standard
warning standards suggest that this may not be the safest place: [unicode skull
and crossbones], [unicode caution sign], [unicode radioactive sign], [unicode
biohazard sign]."
```

This example text was used to produce a story file which has been tried against both Zoom for Mac OS X and Windows Frotz. The Latin, Greek, Cyrillic and Hebrew text all functioned perfectly on both, but a point of difference showed when writing the Hebrew alphabet: Zoom wrote this right-to-left, Windows Frotz left-to-right. The exotic symbols displayed on Zoom (though others not mentioned above, such as "[unicode staff of hermes]", did not): but most appeared only as black squares on Windows Frotz, exceptions being the astrological signs for Venus and Mars and the musical note.

---

71 ★ Example **Fifty Ways to Leave Your Larva**

RB

Using text substitution to make characters reply differently under the same circumstances.

We can use these substitutions to put together fairly complicated variations within a single piece of text:

```
"Fifty Ways to Leave Your Larva"

The Beekeeper's Palace is a room. Wasp is a woman in the palace. Drone is a
man in the palace.

Instead of kissing someone:
    say "'[denial], [insult]! [boast]!'";
```

In this context, [denial] is understood to refer to the denial property of the noun -- but we could spell it out with "denial of the noun" if we wanted to.

```
A person has some text called denial. The denial of a person is usually "Stand
back". The denial of Drone is "You forget yourself"

A person has some text called insult. The insult of a person is usually
"Grasshopper". The insult of Wasp is "Larva".

A person has some text called boast. The boast of a person is usually "I am
ferocious". The boast of Drone is "I have ferocious allies".
```

And then it would be trivial to insert further rules using these responses:

```
Instead of attacking someone:
    say "'Get away, [insult]!'"

Test me with "kiss wasp / hit wasp / hit drone / kiss drone".
```

72 ★★★ Example **Fifty Times Fifty Ways** RB

Writing your own rules for how to carry out substitutions.

There is only so much we can cram into a text property, so being able to swap in properties is useful but limited. Fortunately, we can also, if we want, create new phrases for how to say things in brackets:

```
"Fifty Times Fifty Ways"

The Beekeeper's Palace is a room. Wasp is a woman in the palace. Drone is a
man in the palace.

A person can be fierce or mellow. Wasp is fierce. Drone is mellow. A person can
be calm or angry. A person is usually calm. A person has some text called insult.
The insult of a person is usually "Grasshopper". The insult of Wasp is "Larva".

Instead of kissing someone:
    say "'[denial for the noun], [insult for the noun]! [boast]!'";
```

Now to provide some meaning to these bracketed forms. We'll start with the easy one:

```
To say boast:
    say "I have ferocious allies".
```

This is a "to say" phrase; we will learn more about phrases in a later chapter, but for now it may be enough to observe that whatever we write after "to say..." becomes a valid substitution in bracketed speech. In this particular case there is no advantage to using the boast token rather than spelling the text out in the quotation, but we might in theory add further instructions to randomize the output, for instance.

To say phrases can be more complex, as well, since we can have them incorporate extra information:

```
To say insult for (speaker - a person):
    if speaker is angry, say "[the insult of the noun]";
    otherwise say "small one".
```

Here where we have (speaker - a person), we are leaving a slot which we can later fill in, madlibs-like, with any person we like. That is why we can write "insult for the noun": we are summoning the To say phrase and telling it to fill in the identity of the unknown speaker with the noun.

This differs from "insult of the noun" in the previous example; in that case, each person had his own insult property, and were merely printing that property out. Here we are actually telling Inform to calculate anew what the insult should be, and giving it some instructions about how to do that.

Our instructions can also get arbitrarily complex:

```
To say denial for (speaker - a person):
    if speaker is calm:
        say "You must not";
    otherwise if speaker is female:
        say "Stand back";
    otherwise:
        say "You forget yourself".

Instead of attacking someone:
    now the noun is angry;
    say "'Get away, [insult]!'"

Test me with "kiss wasp / hit wasp / kiss wasp / kiss drone / hit drone / kiss
drone".
```

So the effects we can get with text substitutions are quite flexible. We could even, if we wanted, fill in the substitutions by random choice, or by selecting items from a long list or table, should we have so bellicose a set of characters that they cannot make do with one or two insulting remarks apiece.

Chapter 6: Descriptions

Contents of *Writing with Inform*

Chapter 5: Text

Chapter 7: Basic Actions

Indexes of the examples

§6.1. What are descriptions?

It is in describing circumstances that Inform really capitalises on the concise, expressive power of natural language, and this chapter brings together the facts about "descriptions".

The simplest descriptions consist of a noun alone. Some refer to single things ("lantern", or "wine cask"), others to kinds of thing ("dead end" or "container"). But we have also seen adjectives alone:

> The oaken desk is **fixed in place**.

Here, "fixed in place" is a description which, to Inform's simple-minded grammar, is a single adjective. And of course adjectives and nouns can be combined:

> The cargo trunk is an **openable container**.

The description "openable container" consists of the noun "container", meaning a kind of thing, and the adjective "openable", which means one of the two possible states of an either/or property held by that thing.

As the next chapter will show, rules also make great use of descriptions:

> Instead of throwing **something** at a **closed openable door**, say "Or you could just use the handle like anyone else, of course."

We have already seen that we can list the items fitting a given description:

> "You look down at [the list of **things in the basket**]."

It's also sometimes convenient to count them up:

number of (description of values) **... number**

This phrase counts the number of values matching the description, which may of course be 0. Example:

```
number of open doors
```

produces the number of doors, anywhere in the model world, which are currently open. A Problem message is produced if the number is potentially infinite, or impractical to count: for instance, Inform rejects "number of odd numbers".

It is because descriptions are so widely useful that they deserve a chapter of their own, and this is it.

↑ Start of Chapter 6: Descriptions
← Back to Chapter 5: Text: §5.13. Making new substitutions
→ Onward to §6.2. Adjectives and nouns

§6.2. Adjectives and nouns

Descriptions can contain a noun, but need not, and can contain any number of adjectives:

```
supporter = the noun supporter
closed = the adjective closed
the open wine cask = the adjective open + the noun wine cask
something portable = (some) + the noun thing + the adjective portable
```

Note that we are not allowed to have more than one noun in the same description (something English occasionally does allow as a coded form of emphasis, as in "the man Jenkins" or "the harlot Helen").

Nouns are simple enough, referring either to kinds or specific things. The noun "something" means "some thing", so is actually a reference to the kind "thing". Inform treats this as having the same meaning as "anything", and all told there are eight special nouns of this kind, but with only three different meanings between them:

```
something = anything
someone = anyone = somebody = anybody
somewhere = anywhere
```

So for instance "anybody male" or "somewhere dark" are valid descriptions. These eight nouns are unusual in being allowed to come at the front of a description: nouns are usually expected to be at the end. (Inform also understands "nothing", "nowhere", "nobody", "no-one" and even "no one", which in a sense are opposites of "something" and the like, but for now we'll look at descriptions of things which do exist rather than don't.)

Start of Chapter 6: Descriptions

Back to §6.1. What are descriptions?

Onward to §6.3. Sources of adjectives

§6.3. Sources of adjectives

We have seen two sorts of adjectives so far: those which refer to either/or properties, like "open" and "closed", and those which come out of new kinds of value. If we define

```
Texture is a kind of value. The textures are rough, stubbly and smooth. Everything has a texture.
```

...then "rough", "stubbly" and "smooth" all become adjectives. (That last sentence "Everything has a texture" was essential, because without it Inform would not know that these words could meaningfully be applied to things.)

In addition to these adjectives, we can create new ones (as we shall see), and a few special adjectives such as "visible", "touchable" and "adjacent" are already defined for us by Inform.

Start of Chapter 6: Descriptions

Back to §6.2. Adjectives and nouns

Onward to §6.4. Defining new adjectives

§6.4. Defining new adjectives

Suppose we want to coin a word for supporters currently supporting something. We can do so with the following sentence:

```
Definition: A supporter is occupied if something is on it.
```

Note the colon, which is essential, and the usage of "it" in the definition part to refer to the object in question. (For this purpose we would write "it" even if we were defining a term about, say, a woman instead of a supporter, so that "she" or "her" might seem more appropriate - but see below.)

This creates the adjective "occupied", and gives it a definition valid for supporters. That restriction on validity means that non-supporters would always fail the description "something occupied"; which might be unfortunate if we wanted to talk about rooms being occupied. We could give a second definition thus:

```
Definition: A room is occupied if a person is in it.
```

These are entirely different senses of the word "occupied" - a mantelpiece is occupied if an invitation is on it, but for a drawing room to be occupied there must be human presence - and

Inform applies whichever sense is relevant when deciding whether or not a given object is "occupied".

Often, though not always, we also want to give a name to the opposite possibility. We can do that as follows:

```
Definition: A room is occupied rather than unoccupied if a person is in it.
```

The "rather than..." part of the definition is optional, but it saves having to write a boringly similar definition of "unoccupied" out in longhand. (Note that Inform does not guess the meaning of "unoccupied" unless it has been explicitly told it. Such guesses are too risky, when so many "un-" words fail to conform to this pattern: "unified", "uncle", "ungulate" and so on.)

Newly defined adjectives cannot be used when creating things, because they are not explicit enough. Inform could not satisfy:

```
The Ballroom is occupied. The bucket is a large container.
```

because there is not enough information: by whom is the Ballroom occupied? How large, exactly? On the other hand, newly defined adjectives are very helpful in conditions and for rules, as we shall see later on.

It is occasionally clumsy having to refer to the subject of a definition using "it". We can avoid this and give the definition better legibility by supplying a name instead. For instance:

```
Definition: a direction (called thataway) is viable if the room thataway from the location is a room.
```

which is a good deal easier to read than

```
Definition: a direction is viable if the room it from the location is a room.
```

★ *See* **New conditions, new adjectives** *for giving more extensive definitions of new adjectives, using phrases*

⬆ Start of Chapter 6: Descriptions

⬅ Back to §6.3. Sources of adjectives

➡ Onward to §6.5. Defining adjectives for values

⬇ Example 73: ★ **Finishing School** The "another" adjective for rules such as "in the presence of another person".

§6.5. Defining adjectives for values

In general, any noun can have adjectives applied to it, and this means that values can have adjectives just as objects can. We have already seen that they can (in some cases, at least) have either/or properties, and this gives them adjectives just as for objects. But we can also write out definitions which apply to values:

```
Definition: A number is round if the remainder after dividing it by 10 is 0.
Definition: A time is late rather than early if it is at least 8 PM.
```

That makes the numbers 20 and 170 but not 37 meet the description "a round number", and the times 8 PM and 11:23 PM but not 9 AM meet the description "a late time". Because they come up fairly often, Inform contains several adjectives for numbers built in:

```
positive - one which is greater than zero (but not 0 itself)
negative - one which is less than zero (but not 0 itself)
even - a number like ..., -4, -2, 0, 2, 4, ...
odd - a number like ..., -5, -3, -1, 1, 3, 5, ...
```

Similarly, two useful adjectives are built in to talk about text:

```
empty - the text "", with no characters in it, not even spaces
non-empty - any text which does have at least one character in
```

Adjectives can have multiple definitions and, as long as each applies to a different sort of noun, there will be no problem. We could write:

```
A thing can be round, square or funny-shaped.
A container can be odd or ordinary.
```

And these definitions of "round" and "odd" will not interfere with the ones applying to numbers, because Inform can always look at the noun to see which definition is meant in any given case. For instance,

```
if the score is round, ...
```

must mean "round" in the sense of numbers, because the score is a number. Inform itself makes good use of this; "empty" also has meanings applying to rulebooks, lists and activities, for instance, as will be seen later.

Although it's more usual to give a definition to apply to a whole kind, we can actually give a specific definition to apply to just a single object or named value. For example:

```
A colour is a kind of value. The colours are red, green and blue.
Definition: red is subtle if the player is female.
Definition: a colour is subtle if it is blue.
```

The first definition of "subtle" takes precedence, of course, since it has the more specific domain - it applies only to red. The effect of this is that, if the player's female, the subtle colours are red and blue; if not, just blue.

Start of Chapter 6: Descriptions

Back to §6.4. Defining new adjectives

Onward to §6.6. Whereabouts on a scale?

Example 74: ★★★ **Only You...** Smoke which spreads through the rooms of the map, but only every other turn.

§6.6. Whereabouts on a scale?

Adjectives are often used in English to give a sense of where something is on a sliding scale. We talk about "a tall man" and "a short man", but without meaning that all men are either tall or short. If pushed, we might say that tall means about 6 feet and up, short means about 5 feet 6 and down, but we more often compare one person's height against another's.

Inform allows us to use adjectives in the same way. For example, every container has a number called its "carrying capacity". We can define:

```
Definition: A container is huge if its carrying capacity is 20 or more.
Definition: A container is large if its carrying capacity is 10 or more.
Definition: A container is standard if its carrying capacity is 7.
Definition: A container is small if its carrying capacity is 5 or less.
```

These definitions are similar to those in the previous section, but have a very specific (and strictly enforced) shape to them. The adjective must be a single word. We have to say "its" (i.e., of it), not the ungrammatical "it's"; we have to specify a property, and a literal value of it, and we must either give an exact value or else conclude with "or more" or "or less". If we create something with one of these properties:

```
The basket is a large container in the Shop. The thimble is a small container in the
Shop. The matchbox is a standard container in the Shop.
```

then they will have the most moderate values they can have, that is, the basket will have carrying capacity 10 and the thimble 5 (and of course the matchbox 7). Both of the following tests will then fail:

```
if the basket is huge ...
if the basket is a small container ...
```

because the basket is neither huge nor small, but somewhere in between.

Sometimes the meaning of adjectives must depend on their context, as we see from the following example, where we assess heights in inches:

```
A person has a number called height. Definition: A man is tall if his height is 72 or more.
Definition: A woman is tall if her height is 68 or more.
```

Inform then judges whether someone is or is not "tall" using different standards for men and for women, and

```
In the Shop are a tall man and a tall woman.
```

creates a man 72 inches tall and a woman 68 inches tall.

Start of Chapter 6: Descriptions

Back to §6.5. Defining adjectives for values

Onward to §6.7. Comparatives

§6.7. Comparatives

The special definitions in the previous section have a further effect. When we define:

```
Definition: A container is large if its carrying capacity is 10 or more.
```

we not only say how to test if something is large (see if its capacity is at least 10) and how to create something large (give it a capacity of exactly 10), we also create a new form of comparison. Thus,

```
if the basket is larger than the thimble ...
if the thimble is not larger than the basket ...
```

are both true. If we also define "huge" and "small", as in the previous section, we also get comparisons "huger than" and "smaller than". Note that "huger than" has exactly the same meaning as "larger than": we can use whichever wording seems more natural. (For bacilli, for instance, we would probably not say "huger than", even though the meaning would be unambiguous.)

We can also compare two things to see if they share the same value of a property. For instance, to go back to the heights example, once we define "tall" and "short", we get that exactly one of the following will be true:

```
if Adam is taller than Eve ...
if Adam is the same height as Eve ...
if Adam is shorter than Eve ...
```

Though it will not always seem natural wording, we can use the comparison "the same P as" for any property P which has a value. Do we think "if the basket is the same carrying capacity as the thimble" is good English? Maybe, maybe not. But we are always at liberty to spell things out in full:

```
if the carrying capacity of the basket is the carrying capacity of the thimble ...
```

Start of Chapter 6: Descriptions

Back to §6.6. Whereabouts on a scale?

Onward to §6.8. Superlatives

§6.8. Superlatives

Lastly, if we define an adjective in this calibrating way, we also automatically benefit from the use of the superlative form. That is, if we define

```
Definition: A container is large if its carrying capacity is 10 or more.
Definition: A container is small if its carrying capacity is 5 or less.
```

then we can talk about things like this:

```
the largest container
the smallest open container
```

Though we should be careful, in the second case, because we might get nothing: maybe all the containers are closed at the moment this is used. And in general there might be several equally large largest containers, in which case we should not rely on getting any particular one of those rather than another.

Note that Inform constructs comparatives and superlatives by a pretty simplistic system. If we want to use these forms for an adjective expressing the relatively large size of a room, we had better go with "roomy" (roomier, roomiest) - not "spacious" (spaciouser, spaciousest).

Start of Chapter 6: Descriptions

Back to §6.7. Comparatives

Onward to §6.9. Which and who

§6.9. Which and who

A description can not only talk about things in terms of themselves, but also in terms of their relationships to the rest of the world. For instance,

```
an open container on the table
a woman inside a lighted room
an animal carried by a man
a woman taller than Mark
something worn by somebody
```

are all valid descriptions. These are really abbreviations, having missed out the words "which is" or "who is", as appropriate:

```
an open container which is on the table
a woman who is inside a lighted room
an animal which is carried by a man
a woman who is taller than Mark
something which is worn by somebody
```

and indeed those are also valid descriptions. The other sentence verbs can all be used here, too. So for instance:

```
a man who does not wear anything
something which supports something
```

And sometimes we should spell out "who is" regardless:

```
a man who is not Sherlock Holmes
```

Since these clauses can be attached to the end of any valid description, descriptions can grow longer still:

```
something worn by a woman who is in a dark room
```

Pedants who flinch when "which" is used to introduce a restrictive clause are welcome to use "that" instead.

⬆ Start of Chapter 6: Descriptions

⬅ Back to §6.8. Superlatives

➡ Onward to §6.10. Existence and there

⬇ Example 75: ★★ **Versailles** A mirror which will reflect some random object in the room.

§6.10. Existence and there

"There" is a curious word in English, which mostly refers to some place which is being talked about - but which can sometimes mean the whole world. In Ian Fleming's novel "From Russia With Love", a chapter narrating a committee meeting of SMERSH officers in Istanbul ends with one of the Russians saying:

```
There is a man called Bond.
```

What does this "there" mean? It really just means that Bond exists. In fact, he's watching the meeting through a concealed periscope, but the SMERSH general doesn't know that. All he is saying is that Bond is out there somewhere, and is not imaginary, or dead.

Inform also allows "there is" (or "there are") to talk about what exists, or does not. This is especially useful if, for some reason, we don't want to give a name to something. For example:

```
There is a door in the Summerhouse.
```

Another reason might be that we want to create something but not put it anywhere. If Inform reads the sentence:

```
There is a man called Bond.
```

then it creates a man, gives him the name Bond, but places him initially off-stage - not in any room, that is, but available to be brought into play later on, like an actor who is not needed until Act II.

"There" also provides a useful way to test what exists:

```
if there is a woman in the Summerhouse, ...
```

Or even:

```
if there is a woman, ...
```

which will be true if the model world contains even a single woman, on-stage or off. The alternative "there are" can also be used:

```
if there are women in the Summerhouse, ...
```

but note that this does not necessarily imply more than one woman is present, despite the plural. If we want that, we have to be more explicit:

```
if there is more than one woman in the Summerhouse, ...
```

or, of course, we needn't use "there is" at all:

```
if more than one woman is in the Summerhouse, ...
```

And we can also test non-existence:

```
if there is nobody in the Summerhouse, ...
if there is nothing on the mantelpiece, ...
```

⬆ Start of Chapter 6: Descriptions
⬅ Back to §6.9. Which and who
➡ Onward to §6.11. A word about in

§6.11. A word about in

What does "in" mean? It's worth just a brief diversion to cover this, because "in" has two subtly different meanings.

Meaning 1. Usually, if X is "in" Y then this is because of containment. A croquet ball is "in" a croquet box, which is "in" the Summerhouse. This is the standard meaning, and is the one which happens if we write something like:

```
The croquet ball is in the box.
```

or if we ask a question like:

```
if the croquet box is in the Summerhouse, ...
```

This kind of "in" talks only about direct containment. If we ask

```
if the croquet ball is in the Summerhouse, ...
```

then the answer is that it isn't - it is in the box which is itself in the Summerhouse, but that's not the same thing.

This is almost always the meaning of "in" that we intend. This is only one of a number of relationships between objects - there are also "part of", "on", "worn by" and "carried by", for example. If we have

```
The bird feed is on the sundial.
```

...then "if the bird feed is in the sundial" won't be true: the relationship here is one called support (being on top of, in effect), not containment. But there's no confusion because "on" and "in" are different words, so it's no problem that they have different meanings.

Meaning 2. Much less common. If X is "in" Y and Y is a region, then the meaning is slightly different. Suppose the Garden Area is a region, and contains several rooms - the Croquet Lawn, the Terrace and so on. Then

```
if the croquet box is in the Garden Area, ...
if the bird feed is in the Garden Area, ...
if the Terrace is in the Garden Area, ...
```

are all true. This seems very natural, but in fact is quite different from the first meaning of "in". It allows rooms (and even other regions) to be "in" a region, and it allows indirect containment.

How Inform decides. So which meaning does Inform use, and when? Since these two meanings are so different, it clearly matters.

The answer is that meaning 1 is always the meaning of "X is in Y" unless Y is explicitly the name of a region. Thus:

```
if the croquet box is in the Garden Area, ...
```

is meaning 2, because "Garden Area" is the name of a region. That seems fair enough, but values are indeed sometimes given names (becoming "variables", or values "that vary"). Suppose "mystery value" is a name for a value which is an object, but which has different identities at different times. Then Inform reads

```
if the croquet box is in the mystery value, ...
```

as meaning 1, because whatever "mystery value" is, it isn't explicitly a region name, even if from time to time it might happen to be equal to a region.

That sometimes makes meaning 2 difficult to express. If we ever need it, and this is fairly rare, we can write it like so:

```
if the croquet box is regionally in the mystery value, ...
```

because "regionally in" is always meaning 2 of "in".

Start of Chapter 6: Descriptions
Back to §6.10. Existence and there
Onward to §6.12. A word about nothing

§6.12. A word about nothing

Like "in", "nothing" has two slightly different meanings, though here there's much less potential for confusion.

Meaning 1. "Nothing" as "no thing". This is the meaning in sentences like:

```
Definition: a container is bare if nothing is in it.
```

And similar for conditions like "if the box contains nothing". It's a word which describes the absence of things: it says that, though there might have been many possible items here, it turned out that there were none.

Meaning 2. "Nothing" as a value. This is much less commonly seen, but sometimes Inform stores a value such as a property (or a variable) which always has to be an object. In some circumstances, "nothing" is then a special value meaning that this is not set at present. For instance,

```
Definition: a container is impossible if its matching key is nothing.
```

The "matching key" property of a container is always an object, but is allowed to be "nothing" when there isn't a matching key anywhere. (If such a container is locked, nobody will ever be able to unlock it.)

How Inform decides. So which meaning does Inform use, and when? The answer is that it depends on the relationship being talked about. When this is "is", values are being compared and we are using meaning 2. But when it is any other relationship, like "is in" - which talks about containment - then we are using meaning 1.

Start of Chapter 6: Descriptions
Back to §6.11. A word about in
Onward to §6.13. To be able to see and touch

§6.13. To be able to see and touch

Two of the adjectives built into Inform are:

```
"visible" - the player can see this
"touchable" - the player can touch this
```

So we can write descriptions such as "someone visible" or "a touchable container". We also have adjectives "invisible" and "untouchable", as might be expected. The visibility adjectives are particularly useful because the following is likely to go wrong:

```
if Helen is in a dark room, ...
```

This tests whether the room is dark, of itself; Helen may in fact be able to see by means of a torch, but the room is still "dark".

We can also talk about what other people can see and touch:

```
something which can be seen by Helen
```

are synonymous. Similarly for touch; and we can write such conditions as

```
if Helen cannot see Agamemnon, ...
if Cressida can see Troilus, ...
```

Note that it is essential to establish who does the seeing and touching: so "something which can be seen" will not be allowed, whereas "something which can be seen by Helen" will.

In fact, inside Inform the adjective "invisible" (for instance) has the following straightforward definition:

```
Definition: Something is invisible if the player cannot see it.
```

The exact definitions of visibility and touchability are complicated, because there are so many ways in which vision and touch can be obstructed, but the gist is that they behave as one would expect. Note that in darkness, nothing is visible, and that nobody can see from one room to another. In general anything invisible is also untouchable, but there are a few exceptions to do with being in the dark. Lastly, the player's own body (usually called "yourself" during play) is both visible (in light) and touchable.

Start of Chapter 6: Descriptions

Back to §6.12. A word about nothing

Onward to §6.14. Adjacent rooms and routes through the map

Example 76: ★★★ **Lean and Hungry** A thief who will identify and take any valuable thing lying around that he is able to touch.

§6.14. Adjacent rooms and routes through the map

Another useful adjective built into Inform is "adjacent". Two rooms are said to be adjacent if there is a map connection between them which does not pass through some barrier such as a door. This is easily tested:

```
if the Hallway is adjacent to the Study ...
```

We usually want to know about the places adjacent to the current scene of the action, so that is what the adjective "adjacent" means when applied to rooms. For instance:

```
if somebody is in an adjacent room, ...
```

As with the case of "visible", the adjective is a cut-down version of the more general relationship. This often happens: "worn" and "carried", for instance, imply "by the player" unless something else is specified.

If we want to ask a more direct question, we can obtain specific map connections as follows. (Recall that every map connection leads either to a door, to a room, or to nothing.) If we know which direction we want to look in, then the easiest thing is to use its relation - every direction in the map, say "north", has its own relation, say "mapped north of". So:

```
if the Ballroom is mapped north of the Hallway, ...
```

Alternatively, and particularly if the direction is not a constant,

room (direction) **from**/of (room) **... room**

This phrase produces the room which the given map direction leads to, or the special value "nothing" if it leads nowhere. If it leads to a door, the result is the room through that door. Examples:

```
say "You look north into [the room north from the Garden]."
if the room north from the Garden is nothing, say "The grass leads nowhere."
```

door (direction) **from**/of (room) **... door**

This phrase produces the door which the given map direction leads to, or the special value "nothing" if it leads nowhere or to a room. Examples:

```
let the barrier be the door north from the Garden;
if the barrier is a door, say "Well, [the barrier] is in the way.";
```

room-or-door (direction) **from**/of (room) **... object**

This phrase produces the object which the given map direction leads to, which will always be either a room, a door or the special value "nothing". The phrase is used mainly by the Standard Rules, for technical reasons, and usually it's better to use "room ... from ..." or "door ... from ..." instead.

The map can be a great sprawling mass of rooms and doors connected together, and it can be quite hard to find a way through it one step at a time.

best route from (object) **to** (object) **... object**

This phrase produces a direction to take in order to get from A to B by the shortest number of movements between rooms, or produces "nothing" if there is no way through at all. Example:

```
The description of the brass compass is "The dial points quiveringly to [best
route from the location to the Lodestone Room]."
```

Best routes are ordinarily forbidden to go through doors, but if the suffix "using doors" is added as an option then any open or openable and unlocked door may be used on the way; and if "using even locked doors" is given, then any door at all will do. Since magnetism is no respecter of property, that seems right here:

```
The description of the brass compass is "The dial points quiveringly to [best
route from the location to the Lodestone Room, using even locked doors]."
```

In practice this simple approach sometimes produces impossible journeys, rather the way Google Maps directions from New York to London would recommend driving down to the docks and then swimming. A more careful approach is to use:

best route from (object) **to** (object) **through** (description of objects) **... object**

This phrase produces a direction to take in order to get from A to B by the shortest number of movements between rooms which match the given description, or produces "nothing" if there is no way through at all. Example:

```
best route from the Drawbridge to the Keep through visited rooms
```

The condition - in this case, that "visited rooms" must be used - also applies to both ends of the journey, so if either Drawbridge or Keep are unvisited then this is "nothing". (Similarly, saying something like "...through containers" would mean there is never a route.)

Lastly, the following phrases can find out how long the journey would be. (They are quite a bit faster than using the "best route..." phrases repeatedly and counting.)

number of moves from (object) **to** (object) **... number**

This phrase produces the number of map connections which must be followed in order to get from A to B by the shortest number of movements between rooms. If A and B are the same, the answer is 0; if there is no route at all, the answer is -1. Example:

```
The description of the proximity gadget is "You are now [number of moves
from the location to the Sundial] moves from the Sundial.";
```

number of moves from (object) **to** (object) **through** (description of objects) **... number**

This phrase produces the number of map connections which must be followed in order to get from A to B by the shortest number of movements between rooms matching the given description. If A and B are the same, the answer is 0; if there is no route at all, or if either A or B fail to match the description themselves, the answer is -1.

Route-finding makes it possible to write quite sophisticated conditions concisely. But these sometimes run slowly, because they call for large amounts of computation. How rapidly Inform can find routes depends on which of two methods it uses. Both have advantages - one is fast but needs large amounts of memory, the other is slow but economical. We can choose between them with one of these two use options:

```
Use fast route-finding.
Use slow route-finding.
```

If neither is specified, "fast" is used where the project uses the Glulx virtual machine (see the Settings panel), and "slow" on the Z-machine, where memory is tighter. Fast route-finding is ideally suited to situations where dozens of characters are constantly route-finding through the map as they meander around in a landscape.

★ *See* **Indirect relations** *for route-finding through a relation rather than the map*

Start of Chapter 6: Descriptions

Back to §6.13. To be able to see and touch

Onward to §6.15. All, each and every

Example 77: ★ **Mistress of Animals** A person who moves randomly between rooms of the map.

Example 78: ★ **All Roads Lead to Mars** Layout where the player is allowed to wander any direction he likes, and the map will arrange itself in order so that he finds the correct "next" location.

Example 79: ★★ **Hotel Stechelberg** Signposts such as those provided on hiking paths in the Swiss Alps, which show the correct direction and hiking time to all other locations.

Example 80: ★★★ **A View of Green Hills** A LOOK [direction] command which allows the player to see descriptions of the nearby landscape.

Example 81: ★★★ **Unblinking** Finding a best route through light-filled rooms only, leaving aside any that might be dark.

§6.15. All, each and every

When testing conditions, we normally talk only about specific things, or else ask if a particular circumstance happens:

```
if the oaken door is open
if a woman is carrying an animal
```

But we can also use "all", "each" or "every" to check the whole range:

```
if each door is open
if anyone is carrying all of the animals
if everybody is in the Dining Room
```

Inform allows other English "determiners" (as they are sometimes called), as well:

```
if some of the doors are open
if most of the doors are open
if almost all of the doors are open
```

are true if at least one case is true, if a majority (any number greater than one half) or at least 80 per cent of the possible cases are true, respectively.

And we can also use "none" and "no". These three are all ways to say the same thing:

```
if no door is open
if all of the doors are not open
if none of the doors is open
```

though it may be clearer style to find a positive way of putting things:

```
if all of the doors are closed
```

All, each and every can be applied to values, too - but only in some cases. For example, suppose we write:

```
Colour is a kind of value. The colours are red, orange, yellow, green, blue, indigo and
violet. A colour can be found or unfound.
```

And suppose that, during play, we assign the "found" property to any colour which the player notices on a wall. We might then want to write conditions like so:

```
if every colour is found
if most of the colours are found
if any colour is found
```

But we always have to bear in mind that Inform might have no reasonable way to decide these questions. It will refuse to allow these, for example:

```
if every number is positive
if any text is palindromic
```

(even supposing the adjective "palindromic" has been defined) - there are practically infinitely many possible numbers and texts, so the search cannot sensibly be done.

Start of Chapter 6: Descriptions

Back to §6.14. Adjacent rooms and routes through the map

Onward to §6.16. Counting while comparing

Example 82: ★★★★ **Revenge of the Fussy Table** A small game about resentful furniture and inconvenient objects.

§6.16. Counting while comparing

Lastly we can also ask for a more specific number of possibilities, like so:

```
if two women are carrying animals
if at most three doors are open
if fewer than 10 portable containers are closed
if all but two of the devices are switched on
if there are more than six locked doors
```

Likewise for "less than", "at least", "all except". Something to watch out for is that

```
if two doors are open
```

will be found true if there are (say) three open doors: after all, if three doors are open, then certainly two doors are. So this is not quite counting. We can be more precise by writing

```
if exactly two doors are open
```

The "all but" counts - say, "if all but two doors are open" - are exact: if, in fact, all of the doors are open then this will be found false.

We can often use these counting forms with values, too. As with the use of "all", this is allowed only if the kind of value is one which can reasonably be searched through. For example:

```
if more than three scenes are happening
if there are more than two non-recurring scenes
```

are allowed because the built-in kind of value "scene" (of which much more later on) has only a small number of possible values.

Lastly, note that the "the" in text like "two of the doors" matters: without it, the phrase will not be recognised as a requirement on the number. (This is to make sure that names of things like "two of hearts" are not misinterpreted.)

⬆ Start of Chapter 6: Descriptions

⬅ Back to §6.15. All, each and every

➡ Onward to Chapter 7: Basic Actions: §7.1. Actions

⬇ Example 83: ★★★ **Yolk of Gold** Set of drawers where the item the player seeks is always in the last drawer he opens, regardless of the order of opening.

Examples from Chapter 6: Descriptions

⬆ Start of this chapter

➡ Chapter 7: Basic Actions

↘ Indexes of the examples

73 ★ **Example Finishing School** RB

The "another" adjective for rules such as "in the presence of another person".

It is often useful to write action rules which apply only when the player is observed by a third party. "In the presence of a person", however, will react even if only the player is in the room, because the player is, of course, a person as well.

A convenient way around this problem is to define an "other" adjective:

```
"Finishing School"

The apple is an edible thing carried by the player.

The Hall is a room. Miss Wicket is a woman in the Hall.

The Dormitory is south of the Hall.

Definition: a person is another if it is not the player.

Instead of eating something in the presence of another person:
    say "Your mannerly upbringing prevents you from eating without a fork or knife
in front of someone."

Test me with "eat apple / south / eat apple".
```

If we did not have "another" here, Inform would interpret even the player as a possible schoolmarm, leading to such lines as "yourself stares at you coldly...". Clearly not quite the thing.

74 ★★★★ **Example Only You...** RB

Smoke which spreads through the rooms of the map, but only every other turn.

Suppose we want to have smoke that spreads from room to room, gradually filling the entire map with a clogging smoke. Having it spread every single turn would make for a pretty rapid diffusion, so we temper this by having it spread only on even-numbered turns, instead. Conveniently, Inform by default already knows about even and odd numbers, so we can write:

```
"Only You..."

Section 1 - The Procedure

Every turn when the turn count is even:
    if every room is smoky, make no decision;
    let previously smoky be whether or not the location is smoky;
    repeat with area running through smoky rooms:
        now every room which is adjacent to the area is smoky;
    if previously smoky is false and the location is smoky:
        say "[The location] is filling rapidly with smoke."

A room can be smoky or unsmoky.

Some air is a backdrop. Air is everywhere. Instead of doing something other
than examining or smelling to air: say "It's just air." Understand "smoke" as the
air when the location is smoky.

Instead of examining the air in a smoky room: say "A thick layer of smoke lies
just under the ceiling."

Instead of smelling the air in a smoky room: say "Agh, acrid." Instead of smelling
a smoky room: try smelling the air.

After looking in a smoky room: say "A thick layer of smoke has gathered under
the ceiling."

Section 2 - The Scenario

The Guide Lodge is a room. "A very spacious room capable of containing
several hundred girls while they eat, talk, or do crafts. It is constructed in a not-
unappealing rustic style, with floor-to-ceiling windows overlooking the lake below,
and a fieldstone hearth at the center." The Guide Lodge is smoky.

The Kitchen is north of the Guide Lodge. "Multiple eight-burner ranges, ovens,
and a walk-in refrigerator: you know the sort of thing."

The Industrial Pantry is east of the Kitchen. "Awe-inspiring quantities of food line
every shelf, from the three-gallon tub of mayonnaise to the 50-pound tub of rice.
Perhaps the most astonishing item is a bag of marshmallows big enough to
double as a futon."

The player is in the Pantry.
The Hallway is west of the Guide Lodge. The description of the Hallway is "A perpetually-crammed hallway which has to handle the overflow line for the toilets." A singed sign is fixed in place in the Hallway. The description of the sign is "Where the edge of the sign has not been burnt, the legible words are '...Can Prevent Forest Fires'."

The Toilets are north of the Hallway. "Always in full use, at least when the 12-to-15s are here."

The Coat Closet is south of the Hallway. "Muddy boots may not be worn inside the lodge; instead, about 250 pair are piled here, along with their owners' damp parkas and umbrellas."

The Craft Supply Room is west of the Hallway. "A holding-depot for jugs of white glue and popsicle sticks."

Test me with "x smoke / z / z / z / z / x smoke / look".
```

---

75 ★★★ **Example Versailles** RB

A mirror which will reflect some random object in the room.

One of the advantages of descriptions is that we can use them to pick an item randomly from a specified category. (For more on this possibility, see the Change chapter sections on randomness.)

For instance, suppose we wanted to create a mirror in which the player would see some item from the room reflected. We might write

```
Instead of searching the mirror:
    say "You see [a random thing in the location] reflected back at you."
```

This is the same as "a random thing which is in the location": phrase "in..." can be used briefly in Inform as it can in English.

But, on a little more thought, we might want to expand on this: the mirror perhaps should reflect not only things that are in the room, but anything that the player can see (even if it's on a supporter or carried by someone). So then we might instead write

```
Instead of searching the mirror:
    say "You see [a random visible thing] reflected back at you."
```

There's still a risk, though, that this will produce the response

```
You see the mirror reflected back at you.
```

because, of course, the mirror is itself visible. So instead we might write

```
Instead of searching the mirror:
    say "You see [a random visible thing which is not the mirror] reflected back at
you."
```

```
"Versailles"

The Hall of Plywood Boards is a room. "The Hall of Mirrors is under
reconstruction: it is currently a dank tunnel enlivened only by short placards
about the history of the room.

As though to mock tourists such as yourself who bought their tickets without
knowing this, the officials have left uncovered a single panel of mirror."

The mirror is scenery in the Hall of Plywood Boards. Understand "panel" or
"panel of mirror" as the mirror. The description is "Lovingly restored to
shimmering brilliance, it suggests how marvelous this room would be if you had
had the good sense to arrive after the renovations were complete."

Some tourists are a person in the Hall of Plywood Boards. The tourists are
scenery.

Instead of searching the mirror:
    say "You see [a random visible thing which is not the mirror] reflected back at
you."

Test me with "x mirror / look in mirror / g".
```

A final note: we use "searching" here because Inform understands both SEARCH THING and LOOK IN THING as the searching action, and the player is most likely to type LOOK IN MIRROR in order to see the reflection there. In the absence of an example, we can discover the relationship between actions and their command vocabulary in one of two ways. A complete list of actions and the vocabulary associated with them is available in the Actions index. Alternatively, we can type ACTIONS at a prompt, followed by LOOK IN MIRROR, and get the response

```
[searching the mirror]
You find nothing of interest.
[searching the mirror - failed the can't search unless container or supporter rule]
```

...which tells us that Inform is understanding the action as "searching the mirror".

---

76 ★★★★ **Example Lean and Hungry** *RB*

A thief who will identify and take any valuable thing lying around that he is able to touch.

```
"Lean and Hungry"

Substance is a kind of value. The substances are silver, gold, and lead.
Everything has a substance. A thing is usually lead.
```

```
Definition: a thing is valuable if it is not lead.

The Limestone Cave is a room. "Not very big, and it doesn't go back far, but
you'd hoped to find some shelter here. Outside it is raining exceptionally hard."

The sinister gentleman is a man in the Cave. "Leaning against the wall is a
sinister gentleman in a threadbare waistcoat." The description is "He looks as
though he might once have been quite well off."

After examining the gentleman:
    say "[The noun] smiles back at you in an unnerving fashion."
```

Now we make the rule that governs the gentleman's behavior. Here we're going to invoke the rules that allow characters besides the player to do actions. More about this can be found in the Advanced Actions chapter:

```
Every turn:
    if the sinister gentleman can touch something valuable (called the treasure)
which is not carried by a person:
        try the gentleman taking the treasure.

Report the gentleman taking something:
        say "[The gentleman] slyly acquires [the noun] and tucks it into his pocket."
instead.
```

That "not carried by a person" prevents the gentleman from stealing from the player (or, less plausibly, from himself). If we did want him to pick pockets, we could just have said "which is not carried by the gentleman".

```
The player is carrying a coin, a bust of Abraham Lincoln, a bottle of sherry, a
small pistol, and a wad of Confederate cash. The coin is gold. The pistol is silver.

A locket is in the Cave. The locket is gold. "A broken gold locket lies on the
ground, a token of your late Mama."

Test me with "get locket / i / drop locket / z / drop cash / z / x gentleman / drop all
/ z / z / look".
```

---

77 ★ **Example** **Mistress of Animals** RB

A person who moves randomly between rooms of the map.

Suppose we want a restless sort of character always pacing from room to room. It is quite easy to use adjacency to achieve this effect:

```
"Mistress of Animals"

Corinth is a room. Athens is east of Corinth. Epidaurus is southeast of Corinth
and east of Mycenae. Mycenae is south of Corinth. Olympia is west of Mycenae.
Argos is south of Mycenae. Thebes is northwest of Athens. Pylos is south of
```

```
Olympia. Sparta is east of Pylos and south of Argos. Delphi is northwest of
Thebes.

Artemis is a woman in Thebes.

Every turn:
    if Artemis is in a room (called the current space):
        let next space be a random room which is adjacent to the current space;
        if Artemis is visible, say "Artemis heads to [the next space].";
        move Artemis to next space;
        if Artemis is visible, say "Artemis arrives from [the current space]."

Test me with "z / z / z / z / z / z".
```

Of course, it helps that Artemis is the sort to like open spaces. The implementation would become more complicated if there were doors which might block transit between these locations.

78

★ Example **All Roads Lead to Mars**

RB

Layout where the player is allowed to wander any direction he likes, and the map will arrange itself in order so that he finds the correct "next" location.

Suppose we want to allow the player to wander freely in any direction, but ourselves maintain control over the order in which he encounters the rooms. This sort of effect emphasizes the order of the story-telling over any kind of rigorous simulation of space; on multiple play-throughs, the player might not find all the same rooms in the same locations.

```
"All Roads Lead to Mars"

Before going a direction (called way) when a room (called next location) is not
visited:
    let further place be the room the way from the location;
    if further place is a room, continue the action;
    change the way exit of the location to the next location;
    let reverse be the opposite of the way;
    change the reverse exit of the next location to the location.

The Open Plain is a room. "A wide-open grassy expanse, from which you could
really go any way at all."

The Hilly Place is a room. "The grassland gives way to a somewhat more hilly
area, though there is still very little to guide you any particular way."

The Stream is a room. "This is the third place you've been today, and so the
stream is welcome. How refreshing!"

Test me with "n / s / e / e".
```

If we wanted still to be able to find routes between places, we could define a relationship of connection between rooms, which we would add to as we went along.

---

79 ★★★ **Example Hotel Stechelberg** *RB*

Signposts such as those provided on hiking paths in the Swiss Alps, which show the correct direction and hiking time to all other locations.

The following rule appends a paragraph to every room description. We need not worry about doors (despite the pass in the Bernese Oberland known figuratively as the "Little Door").

```
"Hotel Stechelberg"

After looking:
    say "Yellow arms on the signpost point:-[line break]";
    repeat with destination running through interesting rooms:
        let the way be the best route from the location to the destination;
        if the way is a direction, say " [way] for [the destination]: [number of moves from the location to the destination] Std."

Hotel Stechelberg is a room. "The wooden hiking inn at the end of the road, with flowerboxes, canton flags, outdoor tables and a triangular paddock for the cows contesting the annual Miss Stechelberg competition. Otto and Marianne do cheerful innkeeper things, while the sun blazes from a gentian-blue sky."

A room can be dull or interesting. A room is usually dull.

North of Hotel Stechelberg is Trummelbachfalle. North of Trummelbachfalle is Lauterbrunnen. Lauterbrunnen is interesting.

Southeast of Hotel Stechelberg is Trachsellauenen. Trachsellauenen is interesting.

Test me with "look".
```

With a bit more work, the result might be:

```
Hotel Stechelberg
The wooden hiking inn at the end of the road, with flowerboxes, canton flags, outdoor tables and a triangular paddock for the cows contesting the annual Miss Stechelberg competition. Otto and Marianne do cheerful innkeeper things, while the sun blazes from a gentian-blue sky.

Yellow arms on the signpost point:-
    north for Lauterbrunnen: 2 Std.
    west for Sefinental: 2 Std.
    west for Schilthorn: 6 Std.
    southeast for Trachsellauenen: 1 Std.
    southeast for Oberhornsee: 3 Std.
```

80

Example A View of Green Hills

RB

A LOOK [direction] command which allows the player to see descriptions of the nearby landscape.

Suppose a game in which the player is wandering an open landscape with long vistas, allowing him to LOOK in some direction, or even look at an adjacent location.

```
"A View of Green Hills"

Corinth is a room. Athens is east of Corinth. Epidaurus is southeast of Corinth and east of Mycenae. Mycenae is south of Corinth. Olympia is west of Mycenae. Argos is south of Mycenae. Thebes is northwest of Athens. Pylos is south of Olympia. Sparta is east of Pylos and south of Argos. Delphi is northwest of Thebes.

Understand "look [direction]" as facing.

Facing is an action applying to one visible thing.

Carry out facing:
    let the viewed item be the room noun from the location;
    if the viewed item is not a room, say "You can't see anything promising that way." instead;
    try looking toward the viewed item.
```

In rules about action handling, "noun" refers to the first object that the player has mentioned in his command, so if the player typed >LOOK WEST, "let the viewed item be the room noun from the location" would be processed as "let the viewed item be the room west from the location", and so on.

We can at need override the default behavior, if it is not going to be appropriate for the player to see the next room over. There is only sky above at any time, so...

```
Instead of facing up:
    say "Above you is bright sky."

Understand "look toward [any adjacent room]" as looking toward. Understand "examine [any adjacent room]" as looking toward.

Looking toward is an action applying to one visible thing.

Carry out looking toward:
    say "You make out [the noun] that way."
```

This design allows us to create descriptions for rooms (as seen from the outside) which will work regardless of where we're looking from. For instance:

```
Instead of looking toward Athens:
    say "Even from here you can make out the silhouette of the Acropolis."

Test me with "look north / look south / look up / look east / east / look west".
```

81 ★★★★ **Example** **Unblinking**

Finding a best route through light-filled rooms only, leaving aside any that might be dark.

RB

Suppose we're simulating a situation where the player needs to travel through lit areas only, but we want to give him some hints about which way might be safe. Here we'll find our best route through light-filled rooms.

The slightly tricky part is that it's not necessarily easy to tell whether a room has a lamp in it. We may say "if the Crypt is lighted", but that only tells us whether it has been declared to be inherently lighted or dark, not whether it happens to contain a light source that the player would be able to see if he went in.

The easiest way to get around this is to create an object -- the light-meter; place it in the target location; and check whether it "can see" a lit object. This preserves all the usual rules about open and closed containers, transparency, etc.

```
"Unblinking"

Section 1 - Procedure

The light-meter is a privately-named scenery thing.

Definition: a room (called the target room) is light-filled:
    if the target room is lighted:
        yes;
    move the light-meter to the target room;
    let the answer be false;
    if the light-meter can see a lit thing:
        now the answer is true;
    now the light-meter is nowhere;
    decide on the answer.
```

That done, we're free to use our best-route phrases to choose a particular route.

```
Section 2 - Scenario

The Tomb of Angels is a room. "This ancient underground tomb is lightless but
for a few shafts from the surface. Everywhere in the shadows are carved angels,
their faces worn away by water and pollution, their wings little more than nubs."

The Upward Path is above the Tomb of Angels. It is dark. "The staircase
switches back on itself many times as it ascends towards the Crash Site."

A container called the sarcophagus is in the Upward Path. It is closed and
openable. "A sarcophagus rests in the niche here, [if open]the lid pushed
aside[otherwise]the lid firmly in place[end if]."
```

```
The Crash Site is above the Upward Path. "The ceiling has wholly caved in here,
and the belly of the spaceship above you is visible -- including the escape
hatch."

A candle is a kind of thing. A candle is usually lit. The player carries four
candles.

After looking when the location is not the Crash Site:
    if the best route from the location to the Crash Site through light-filled rooms is
a direction (called next way):
        say "It looks like there's a safe, lit path [if the number of moves from the
location to the Crash Site through light-filled rooms is 1]straight[otherwise]if you
go[end if] [next way].";
    otherwise:
        say "It looks like there is no fully lit path from here to the Crash Site."

Test me with "up / drop candle / down / up / take the candle / open sarcophagus
/ put candle in sarcophagus / down / up / close sarcophagus / d".
```

An important word of caution: this method would give false negatives if there were a backdrop lightsource, such as the moon, providing light to the Upward Path. This is because backdrops are actually moved around the map by Inform during play, following the player around. So if the moon backdrop is in the Crash Site with the player, it will not be in the Upward Path as well -- even if it's scheduled to move there as soon as the player does.

---

82 ★★★★★ **Example** **Revenge of the Fussy Table** *RB*

A small game about resentful furniture and inconvenient objects.

This example does include a number of features that we haven't met yet, particularly rulebooks. Nonetheless, the basic idea should be relatively clear. Our premise is that the player is faced with a series of disgruntled furnishings, none of which want to be responsible for supporting the cold, hot, drippy, or spiky objects lying around the room.

```
"Revenge of the Fussy Table"

Use scoring.

Section 1 - The Supporters

The Dining Room is a room. "A room of handsome proportions, with an
octagonal dining table on a fine parquet floor. The original set of chairs has been
taken away, leaving only a red velvet dining chair and a blue suede armchair."

The fussy table is a supporter in the Dining Room. It is scenery. The description
is "It's hypochondriac: the result of a sojourn in a superior antique shop, where it
picked up the scent of almond oil, words like provenance, and a secretive shame
about being resurfaced.
```

Chairs don't get along with it.

In its mellowest moods, after a really good dusting, it will mention, casually, a cabinet it once knew, which was twelve feet tall, came from the Winter Palace, and had an asking price of $350K."

A chair is a kind of supporter with carrying capacity 1. A chair is enterable. A chair is usually scenery. Understand "chair" as a chair.

The blue suede armchair and the red velvet dining chair are chairs in the Dining Room. The description of the blue suede is "Quiet, mellow, and with a lingering aroma of clove cigarettes, and possibly something less legal. It doesn't sit up straight, and doesn't let anyone else do so either." The red velvet has the description "In temperament, in bearing, in everything it is the opposite of the blue suede chair. It is concerned for its duty all the time, has a rigidly straight back, and pokes diners in the spine when it suspects them of overdoing things."

The carrying capacity of the player is 2.

Section 2 - Sources of Unpleasantness

A thing can be drippy or dry.

Heat is a kind of value. The heats are luke-warm, cold, and scalding. Everything has a heat. Understand the heat property as describing a thing.

A thing can be spiky or smooth. The cleated left shoe is a wearable spiky thing. It is worn by the player. The cleated right shoe is a wearable spiky thing. It is worn.

Definition: a thing is unpleasant if it is scalding or it is cold or it is spiky or it is drippy.

Definition: a supporter is occupied if something is on it.

Definition: something is contented if it is not concerned.

Definition: a chair is concerned if something unpleasant is on it.

Definition: the table is concerned if something drippy is on it or something scalding is on it or something spiky is on it.

Definition: the ceramic tile is concerned if something spiky is on it.

Definition: a room is concerned if something drippy is in it or something scalding is in it or something spiky is in it or someone which is in it is wearing a spiky thing.

Before printing the name of a drippy thing: say "drippy ". Before printing the name of a cold thing: say "cold ". Before printing the name of a scalding thing: say "scalding ".

Section 3 - What the Player Can Do About It

```
After putting a cold thing on a scalding thing:
    say "[The noun] meets [the second noun]; both shriek in pain. But the
necessary heat transfer occurs.";
    now the heat of the noun is luke-warm;
    now the heat of the second noun is luke-warm.

After putting a scalding thing on a cold thing:
    say "[The noun] meets [the second noun]; both shriek in pain. But the
necessary heat transfer occurs.";
    now the heat of the noun is luke-warm;
    now the heat of the second noun is luke-warm.

After taking a scalding thing:
    say "'Right,' you say. 'I'll just hold onto [the noun], shall I? Because that won't
be in the least uncomfortable for me.'"

After taking a drippy thing:
    say "With a sense of martyrdom, you retrieve [the noun], which dribbles water
down your cuffs."

After entering a chair when the player is wearing a spiky thing (called the clog):
    say "You sit on [the noun], lifting [the list of spiky things worn by the player] so
that the parquet is no longer affronted."

Section 4 - In Which the Furnishings Complain

An every turn rule:
    follow the complaint rules.

The complaint rules is a rulebook.

A complaint rule:
    if something (called the offending item) on the table is drippy:
        say "'Help! Get me a coaster!' screams the table[if the table is visible], its
veneer squirming under [the offending item][otherwise] from the Dining
Room[end if].";
        rule succeeds;
    if something (called the offending item) on the red chair is drippy:
        say "'Oh dear,' murmurs the red chair, as [the offending item] drips into its
velvety seat. 'Oh dear, I will have a damp spot. This is so very -- what will people
think?'";
        rule succeeds;
    if something (called the offending item) on the visible armchair is drippy:
        say "[The offending item] visibly begins degrading the suede where it sits.
The armchair is tactfully silent.";
        rule succeeds;
    if a drippy thing (called the offending item) is in the location and the player is
in the Dining Room:
        say "'Cripes,' says the parquet. 'No one mind me at all. Just leave that
[offending item] right here. You know I'm the most valuable thing in the room?'";
        rule succeeds.

A complaint rule:
    if a cold dry thing (called the offending item) is on the table:
        say "The table shivers under [the offending item].";
        rule succeeds.

A complaint rule:
    if something (called the offending item) on the table is scalding:
        say "'Hey!' protests the table, practically smoking at [the offending item]. 'Heard of a trivet?'";
        rule succeeds;
    if something (called the offending item) on the armchair is scalding:
        say "The armchair doesn't complain about [the offending item]; it just begins, quietly, to give off warm sweet clouds of blue steam, as though its inner opium-nature had been released.";
        rule succeeds;
    if something (called the offending item) in the Dining Room is scalding:
        say "'Yes indeed,' [if the location is the Dining Room]says[otherwise]bellows[end if] the parquet. 'That [offending item] won't leave any sort of mark what-so-ever.'";
        rule succeeds.

A complaint rule:
    if something (called the offending item) on the table is spiky:
        say "The table holds very very very still lest [the offending item] mar its finish. But its resentment is palpable.";
        rule succeeds;
    if something (called the offending item) on the velvet chair is spiky:
        say "[The offending item] stabs the velvet of the red velvet chair, which draws itself up even more sharply in a pose of dutiful martyrdom.";
        rule succeeds;
    if something (called the offending item) which is spiky is on the ceramic tile:
        say "'[The offending item] itches!' cries the ceramic tile.";
        rule succeeds;
    if something (called the offending item) in the location is spiky:
        say "'[The offending item] should not be on a parquet floor,' sings the parquet floor sadly.";
        rule succeeds;
    if someone (called the offending person) in the location is wearing something spiky and the location is the Dining Room:
        say "The parquet floor mutters about the things people wear to walk on parquet floors these days.";
        rule succeeds.

A complaint rule:
    if the table supports something which supports a drippy thing (called the offending item),
        say "'[The offending item] could still theoretically dribble down here,' says the table nonchalantly. 'It's possible.'";
    if the velvet chair supports a concerned thing (called the friend),
        say "'[The friend] seems a little worried,' remarks the velvet chair helpfully."

A complaint rule:
    if the concerned table can see a supporter (called the object of envy) which is not concerned:
        say "The table looks jealously at [the object of envy]. 'Some people have such an easy life.'";
        rule succeeds.
```

```
Instead of going to an adjacent room when the player is in the Dining Room:
    say "'You're not leaving!?' whimpers the table at once."
```

Moreover, every time the player gets close to resolving this issue, his unhelpful companion Alison brings in something else inconvenient. We can use the counting of contented supporters to decide when the player is getting close to winning and it's time for her to bring something else...

```
Section 5 - Source of Further Complications

The Tidy Kitchen is south of the Dining Room. Alison is a woman in the Tidy Kitchen. The kettle is a scalding thing. The ceramic tile is a cold portable supporter. It has carrying capacity 1. The platter is a scalding portable supporter. It has carrying capacity 1. The glass of ice water is a cold, drippy thing.

Instead of drinking the glass of ice water, say "It's not yours, it's for Alison, once she finishes cooking."

Alison carries the kettle, the tile, the platter, and the glass.

An every turn rule:
    follow the behavior rules.

The behavior rules is a rulebook.

A behavior rule:
    if ice water is carried by Alison and the player is in the Dining Room and almost all of the supporters are not concerned:
        let the next victim be a random fixed in place contented supporter;
        move the ice water to the next victim;
        say "Humming to herself, Alison brings in [an ice water] and sets it down smack on [the next victim], then goes back out.";
        rule succeeds.

A behavior rule:
    if platter is carried by Alison and the player is in the Dining Room and almost all of the supporters are not concerned:
        let the next victim be the red velvet chair;
        move the platter to the next victim;
        say "Whistling a jaunty tune, Alison brings in a hot platter and deposits it on [the next victim], then returns to the kitchen.";
        rule succeeds.

A behavior rule:
    if the tile is carried by Alison and the player is in the Dining Room and almost all of the supporters are not concerned:
        let the next victim be a random contented chair;
        move the tile to the next victim;
        say "Humming to herself, Alison brings in [the tile] and sets it down smack on [the next victim], then goes back out.";
        rule succeeds.

A behavior rule:
    if kettle is carried by Alison and the player is in the Dining Room and at least
```

```
three fixed in place supporters are contented:
        let the next victim be the fussy table;
        move the kettle to the next victim;
        say "Just then the kettle boils in the kitchen. Whistling chirpily, Alison brings
it in and sets it down smack on [the next victim], then goes back out.";
        rule succeeds.
```

And just so that the player knows where he stands at the end of each turn:

```
Section 6 - General Assessment

An every turn rule:
    if some of the things are concerned, say "You sense some resentment from
[the list of concerned things]."

An every turn rule:
    if the player is carrying more than one scalding thing:
        let the dropped item be a random scalding thing carried by the player;
        say "You can't hang onto so many burning hot things, and lose your grip on
[the dropped item].[line break]";
        silently try dropping the dropped item.

An every turn rule:
    if almost all of the supporters are concerned,
        say "'I hope everything is going well,' bellows Alison from the other room."
```

Finally, we need some rules to decide when the player is, in fact, finished.

```
Section 7 - Scoring

An every turn rule:
    now score is 5 minus the number of concerned things;
    if the location is concerned, decrement the score;
    if all of the supporters are concerned and the location is concerned, end the
story;
    if none of the supporters are concerned and the location is not concerned,
end the story finally.

The maximum score is 5.

When play begins, now score is 4.

Test me with "i / take off left shoe / get the glass / put the glass on the platter /
get the tile / drop the tile / get the kettle / put the kettle on tile / sit on the blue
armchair".
```

83 ★★★ **Example** **Yolk of Gold** *RB*

Set of drawers where the item the player seeks is always in the last drawer he opens, regardless of the order of opening.

Suppose that for dramatic effect we would like the player to find the thing he is looking for always in the last drawer he opens...

```
"Yolk of Gold"

The Turret is a room. "A cramped little room distinguished chiefly by the spiral staircase that descends from it. The windows look out over the rooftop."

The Rooftop is outside from the Turret.

The staircase is an open unopenable door. It is scenery. It is below the Turret and above the Library. The description is "A winding stair carved out of the single trunk of a massive tree, all in a dark wood; the outside of the stairs left unfinished with the bark still on, but the treads worn smooth by long and constant use."

The description of the Library is "Here, it seems, you have found your mark: books line both walls, a dark carpet lies on the floor, and a strange dress hangs up in a presentation case. And the thing you were told to look for, a desk with three drawers.

A spiral staircase leads up and out."

The cherry desk is scenery in the library. The description is "A deep, satin-lustrous cherry, with scrollwork legs and gilt touches. The years have not been kind, and it has cracked and split in several places; the finish is damaged, and where there is inlaid mother of pearl, it is beginning to come up from its bed. But it is still a sound piece, and features three drawers."

After examining the desk for the first time, say "(Your employers were able to tell you to look for it, but not which drawer to look in. Typical.)"

A drawer is a kind of container. A drawer is always openable and closed. The description of a drawer is "The usual drawer of heavy wood, inadequately smoothed for ease of use."

The top drawer is a drawer. The middle drawer is a drawer. The bottom drawer is a drawer. The top drawer, the middle drawer, and the bottom drawer are part of the desk. A drawer can be explored or unexplored. A drawer is usually unexplored. Instead of searching a closed drawer, try opening the noun.

After opening a drawer when no drawers are explored:
    now the noun is explored;
    say "There is a tremendous screech, but nothing whatsoever inside, not even dust."

After opening an unexplored drawer when exactly one drawer is explored:
    now the noun is explored;
    say "You struggle to open this one a bit more quietly, conscious all the time of noise... but no, it's empty. Just stands to reason."

After closing a drawer:
    say "There is a racket of wood protesting against wood as you do so, which
```

```
makes you wonder if you hadn't better just leave them open from now on. The
unhappy owner is probably going to catch on soon enough anyway."

Before opening an unexplored drawer when exactly two drawers are explored:
    move the mysterious thing to the noun;
    now the noun is explored.

There is a mysterious thing. The description is "A very familiar-looking
hemispherical weight of metal, goldish in tone though perhaps not made of gold.
This one has a slight stickiness about the bottom surface."

Instead of opening the desk when at least one drawer is unexplored:
    let the next drawer be a random unexplored drawer;
    say "(starting with [the next drawer])[line break]";
    try opening the next drawer.

Instead of looking under the desk when no drawers are explored:
    say "You carefully survey the ground around the desk. There don't seem to be
any hidden tripwires or traps to prevent you from having a look in the drawers."

Instead of looking under the desk when at least one drawer is unexplored:
    say "Nothing there either. Thoroughness is a virtue with tedium as a side
effect, as your mother used to say -- but they always counted her a trifle
slapdash."

Instead of looking under the desk when all drawers are explored:
    say "There's nothing down there, but this doesn't come as a great surprise."

Instead of searching the desk when at least one drawer is unexplored:
    move the mysterious thing to the player;
    say "You perform a hasty, squeaky search of ";
    if no drawer is explored, say "all three drawers, discovering nothing and
nothing in rapid succession. But on the third drawer you ";
    if exactly two drawers are unexplored, say "the remaining two drawers.
There's nothing in the second, but in the third you ";
    if exactly one drawer is unexplored, say "the last drawer. In it, you ";
    say "turn up a promising hemispherical object.";
    now all the drawers are explored.
```

Perhaps, just for fun, we have all the other scenery draw the player's attention back to the main point, as well.

```
The carpet is scenery in the Library. The description is "It is too dark for you to
make out details of the antique design, which seems dimly to represent an early
voyage to the moon, with ships and the travelling stars." Understand "rug" as
carpet.

Instead of looking under the carpet:
    if none of the drawers are explored, say "You peel up a corner of the rug
gingerly; nothing results. A perfectly ordinary rug, then." instead;
    if all of the drawers are explored, say "No time for that kind of nonsense.
You'd better get out and away while you can." instead;
    if some of the drawers are explored, say "No time for that nonsense. The
desk's what you want now; what you came for won't be embedded in the
flooring." instead.
```

```
The glass presentation case is transparent scenery in the Library. The description is "The case is taller than you are, framed in wood, with large panels of glass, the better to display the remarkable contents."

Instead of attacking the glass presentation case:
    say "The glass pane of the presentation case shatters, throwing fine glass everywhere, including over the delicate museum-piece inside. Nor does the noise pass unnoted: only a second passes before there are footsteps in the hall, and though you make for the concealing darkness and escape of the turret, you are not swift enough. The servants are soon on you, and you are made to regret, quite painfully, this casual act of vandalism.";
    end the story saying "You have lost your opportunity."

The strange dress is a wearable thing in the glass case. The description is "Not the sort of dress that anyone wears now: such elaboration would be ludicrous. It drips gold -- gilt lace, ruffles of trimmed gold, shimmering golden tracery -- dulled here and there by the sinister black of faceted jet."

Test me with "d / x case / x dress / x carpet / look under carpet / look under desk / x desk / open top drawer / close top drawer / look under desk / open bottom drawer / close bottom drawer / open middle drawer / get thing / look under carpet / look under desk / up".
```

Chapter 7: Basic Actions

Contents of *Writing with Inform*
Chapter 6: Descriptions
Chapter 8: Change
Indexes of the examples

§7.1. Actions

"Actions" are what we get if we try to break down a narrative into its irreducible parts. We might casually say that we are "going shopping", but this involves many smaller steps: going north, going east, entering the shop, examining a loaf of bread, taking it, giving money to the baker, and so on.

An action is an impulse to do something. This may or may not be a reasonable aspiration, and may or may not be achieved. The player's exploration of an interactive fiction is made by a sequence of actions, so much of the designing process comes down to responding to these actions.

We write actions using present participles. For instance, if the player types "take napkin" or "get the napkin" or something similar then the resulting action would be written as:

```
taking the napkin
```

The details of what words the player actually typed are unimportant to us: we deal only in actions.

Every action ends in success or failure. In this context, success means only that the player's intention has been fulfilled. If the player sets out to take the napkin, but finds a million-pound banknote in its folds instead, the action will be deemed to be a failure.

The testing command ACTIONS causes Inform to log every action as it happens, and what its outcome is. (ACTIONS OFF turns this off again.) For instance:

```
>s
[going south]
Security Vault
You can see a metal door here.
[going south - succeeded]
```

```
>close door
[closing metal door]
You close the metal door.
[closing metal door - succeeded]
>take door
[taking metal door]
That's fixed in place.
[taking metal door - failed the can't take what's fixed in place rule]
```

A good way to get a sense of the constant flow of actions is to use this command and then wander around an existing work, trying things out. ACTIONS can also give an insight into the web of rules governing play: there are more than ten different ways an attempt to take something can fail, for instance.

⬆ Start of Chapter 7: Basic Actions

⬅ Back to Chapter 6: Descriptions: §6.16. Counting while comparing

➡ Onward to §7.2. Instead rules

§7.2. Instead rules

An action is ordinarily handled by running it through Inform's extensive rulebooks of what might be called normal behaviour. An action such as "taking the napkin", for instance, will be run through numerous checks to see if it is physically reasonable, and then provided all is well, the napkin will be moved into the possession of the player.

Instead, though, we can bypass the rules to do with an action and do something else:

```
Instead of eating the napkin: say "Why not wait for the actual dinner to arrive?"
```

This is an example of a "rule": a set of circumstances followed by a list of instructions. When those circumstances apply, the instructions are carried out. In the case of an "instead" rule, after this is done the action is immediately ended (and counts as a failure, since the original intention has been thwarted).

A friendly alternative can be used when there is only a single instruction, as here: in such rules the colon can be replaced with a comma. Thus:

```
Instead of eating the napkin, say "Why not wait for the actual dinner to arrive?"
```

⬆ Start of Chapter 7: Basic Actions

⬅ Back to §7.1. Actions

➡ Onward to §7.3. Before rules

⬇ Example 84: ★ **Grilling** A grill, from which the player is not allowed to take anything lest he burn himself.

⬇ Example 85: ★ **Bad Hair Day** Change the player's appearance in response to EXAMINE ME.

§7.3. Before rules

Despite what was said in the previous section, instead rules do not quite bypass all of the usual rules. Inform knows that certain actions require light: for instance,

```
examining the napkin; looking; looking under the dining table
```

and if it is dark then none of these actions will be allowed, and any instead rules about them will not even be reached. Similarly, Inform knows that most actions require physical access to their objects: so "taking the napkin" would be blocked if the napkin were, say, inside a closed glass bottle, whereas "examining the napkin" would not. So an instead rule can only take effect if the action has already passed these basic reasonability tests.

"Before" rules genuinely precede checking of any kind. They also differ from instead rules in that they do not automatically stop the action in its tracks. Rather, they are provided as an opportunity to ensure that something else is done first. For example:

```
Before taking the napkin, say "(first unfolding its delicate origami swan)".
```

whence

```
>GET NAPKIN
(first unfolding its delicate origami swan)
Taken.
```

We have seen that instead rules automatically stop actions, whereas before rules automatically allow them to continue. We sometimes want to change this. The magic word "instead" can therefore be tacked on to any instruction in a before rule, and will have the effect of immediately stopping the action at that instruction. Thus the following two rules are (almost) equivalent:

```
Before taking the key, instead say "It seems to be soldered to the keyhole."

Instead of taking the key, say "It seems to be soldered to the keyhole."
```

It is also possible to be explicit about stopping the action:

stop the action

This phrase stops the current rule, stops the rulebook being worked through, and finally stops the action being processed. Example:

```
Before taking the key:
    say "It seems to be soldered to the keyhole.";
    stop the action.
```

Finally, we can prevent Inform from stopping the action when it otherwise might:

continue the action

This phrase ends the current rule, but in a way which keeps its rulebook going, so that the action being processed will carry on rather than being stopped. Example:

```
Instead of taking the napkin:
    say "(first unfolding its delicate origami swan)[command clarification break]";
    continue the action.
```

An "instead" rule ordinarily stops the action when it finishes, so the "continue the action" is needed to make things carry on. (This rule would have been better written as a "before" rule, in fact, but it shows the idea.)

As a general principle, it is good style to use instead rules whenever blocking actions, and before rules only when it is genuinely necessary to do something first but then to continue: in fact, it is good style to use "stop the action" or "continue the action" as little as possible.

Start of Chapter 7: Basic Actions

Back to §7.2. Instead rules

Onward to §7.4. Try and try silently

Example 86: ★★★ **Democratic Process** Make PUT and INSERT commands automatically take objects if the player is not holding them.

Example 87: ★★★★ **Sand** Extend PUT and INSERT handling to cases where multiple objects are intended at once.

§7.4. Try and try silently

Chapter 2 noted that surveys of Inform source text showed that the three most popular phrases used by authors are "say", "if" and "now". The fourth most popular is "try", which allows us to trigger off actions ourselves, rather than waiting for the player to type something which generates them. Thus:

try (action)

This phrase makes the action, which has to be named literally, take effect now. Example:

```
Instead of entering the trapdoor, try going up.
```

It's as if the player had typed GO UP as a command. Note that the action has to be specific:

```
try eating something;
```

is not allowed, since it doesn't say exactly what is to be eaten.

The word "try" is intended to make clear that there is no guarantee of success. For example:

```
Before locking the front door, try closing the front door.
```

could go wrong in any number of ways - perhaps the door is closed already, perhaps it is not openable, perhaps somebody has wedged it open. It would be safer to write:

```
Before locking the front door:
    try closing the front door;
    if the front door is open, stop the action.
```

There's no need to say anything if closing didn't work, because the closing action will have done that already. A neater approach still is to use:

silently try (action)

or:

try silently (action)

This phrase makes the action, which has to be named literally, take effect now, under the "silent" convention which means that routine messages aren't printed. Example:

```
try silently taking the napkin;
```

Silence is maintained only if this new action, the taking of the napkin, is successful (so if the napkin is successfully taken, the text "Taken." will not appear): if the action should fail, a suitable objection will be voiced as usual.

So now we have:

```
Before locking the front door:
    try silently closing the front door;
    if the front door is open, stop the action.
```

And this is neater because it won't produce a pointless "You close the front door." message.

See **Stored actions** *for how to store up actions as values and try those, too, so that isn't necessary to name the action as literally as in the examples above*

Start of Chapter 7: Basic Actions

Back to §7.3. Before rules

Onward to §7.5. After rules

Example 88: ★ **Fine Laid** Making writing that can be separately examined from the paper on which it appears, but which directs all other actions to the paper.

Example 89: ★ **Hayseed** A refinement of our staircase kind which can be climbed.

§7.5. After rules

There is pleasantly little to be said about "after" rules. If an action has survived all the rules in its way, and has actually succeeded, then we need to give the player a response which acknowledges this. Inform's normal rules will be sufficient to say something undramatic: for instance, if "taking the napkin" has succeeded then it will reply "Taken." to the player.

An after rule is an opportunity to say something more interesting:

```
After taking the diamonds, say "Taken!"
```

(Well, slightly more interesting.) After rules automatically end the action (as a success), which is what we would want in the above case. Allowing it to continue would simply result in "Taken." being printed as well. However, should we really need to do something and then carry on:

```
After taking the diamonds: say "(Mr Beebe looks up sharply.) "; continue the action.
```

Start of Chapter 7: Basic Actions

Back to §7.4. Try and try silently

Onward to §7.6. Reading and talking

Example 90: ★ **Morning After** When the player picks something up which he hasn't already examined, the object is described.

§7.6. Reading and talking

A few actions apply not to items alone, but also involve what might be called conversation. The first is the one used for looking things up in books (which is conversation of a kind, even if the author is not present): "consulting ... about ...". For example,

```
In the Grove is a book of sybilline verses.

After consulting the book about "grove", say "The Grove is a sacred yadda, yadda.
There's a tree, that sort of thing. Wisdom."

After consulting the book about "future events", say "It's a bit, what's the word? Delphic."
```

Note that what follows "about" here is a piece of text in double-quotes, and not the name of something. It can be almost any text at all, and in fact we shall later see (in the chapter on "Understanding") that we can match complicated patterns of words, too.

Similar actions are used for conversing with people:

```
After asking the Sybil about "verses", say "She blushes."

After telling the Sybil about "persians", say "She nods gravely."

After answering the Sybil that "I am mad", say "She sighs."
```

These would be produced by commands like "ask sybil about verses", "tell sybil about persians" and "answer i am mad". Answering is little-used except that it also catches commands like "sybil, something unrecognized", which inexperienced players sometimes type. Asking and telling, however, are important actions and the difference between them is often worth preserving. If you would prefer to make "tell sybil about X" do the same as "ask sybil about X", the following rule would serve:

```
Instead of telling the Sybil about something, try asking the Sybil about it.
```

Games with a lot of conversation often involve great heaps of rules like the ones above, which can be repetitious to type out. We shall also later see (in the chapter on "Tables") that we can tabulate questions and answers in a much more concise way, if we prefer.

★ *See* **Topic columns** *for table-based ways to store and retrieve conversation*

⬆ Start of Chapter 7: Basic Actions

⬅ Back to §7.5. After rules

➡ Onward to §7.7. The other four senses

⬇ Example 91: ★ **Sybil 1** Direct all ASK, TELL, and ANSWER commands to ASK, and accept multiple words for certain cases.

⬇ Example 92: ★ **Lucy** Redirecting a question about one topic to ask about another.

⬇ Example 93: ★★★ **Sybil 2** Making the character understand YES, SAY YES TO CHARACTER, TELL CHARACTER YES, ANSWER YES, and CHARACTER, YES.

⬇ Example 94: ★★★★ **Costa Rican Ornithology** A fully-implemented book, answering questions from a table of data, and responding to failed consultation with a custom message such as "You flip through the Guide to Central American Birds, but find no reference to penguins."

§7.7. The other four senses

The five senses are all simulated with actions. Sight is so informative that it is handled by a whole range of actions: "looking", which describes the general scene; "examining something", which takes a closer look at a specific thing; "looking under something", and so on.

The other senses have one action each: "listening to something", "touching something", "tasting something" and "smelling something". It makes no sense to touch or taste the general scene, but listening and smelling are a different matter: we often just listen, without listening to anything specific. If the player types the command "listen", Inform understands that as listening to the current location: similarly for the bare command "smell". Thus:

```
Instead of listening to the Seashore, say "The song of gulls."

Instead of smelling the Cave, say "Salt and old seaweed."
```

Start of Chapter 7: Basic Actions

Back to §7.6. Reading and talking

Onward to §7.8. Rules applying to more than one action

Example 95: ★★★ **The Art of Noise** Things are all assigned their own noise (or silence). Listening to the room in general reports on all the things that are currently audible.

§7.8. Rules applying to more than one action

A description can include more than one choice of action. For instance:

```
examining or searching the desk
```

matches either of "examining the desk" or "searching the desk". We can have more than two actions, of course:

```
examining, looking under or searching the desk
```

The actions combined like this need to be compatible with each other, at least a little. For instance, this will generate a problem message:

```
waiting or searching the desk
```

because it makes no sense to "wait the desk". On the other hand, this is fine:

```
waiting or searching
```

The general rule is that if we specify one or more objects ("the desk" in the above example), then each of the actions we quote must take at least that many objects.

For example, the following saves us writing the same basic rule three times over:

```
Instead of examining, looking under or searching the desk: say "There's no use poking
around in that old desk."
```

Start of Chapter 7: Basic Actions

Back to §7.7. The other four senses

Onward to §7.9. All actions and exceptional actions

§7.9. All actions and exceptional actions

The special description "doing something" (or "doing anything") matches any action, and "doing something to ..." also allows the noun to be specified.

For instance, the following puts its object out of bounds:

```
Instead of doing something to the cucumber sandwich, say "Lady Bracknell stares
disapprovingly down her pince-nez at you, in a way which no amount of hunger or
curiosity could overcome."
```

We sometimes need to be a little careful here: "waiting" qualifies as "doing something", but not as "doing something to something", because there is no object. "Putting the handbag on the cucumber sandwich" would also not qualify as "doing something to the cucumber sandwich" - only to the handbag.

More often, we would like to restrict the range of allowable actions to a select few. For instance:

```
Instead of doing something other than looking, examining or waiting: say "You must
learn patience."
```

(Or we can write "except" instead of "other than".) Or we might have an object, too:

```
Instead of doing something other than examining, taking or dropping with the dagger:
say "Don't fool around with that dagger. It's exceedingly sharp."
```

Note the "with", which is crucial here. Without it, the rule is subtly different:

```
Instead of doing something other than examining, taking or dropping the dagger: say
"Don't fool around with that dagger. It's exceedingly sharp."
```

This second version matches if the action is, say, taking a shield, or even just looking, because that would be an action other than examining the dagger, taking the dagger or dropping the dagger.

Start of Chapter 7: Basic Actions

Back to §7.8. Rules applying to more than one action

Onward to §7.10. The noun and the second noun

Example 96: ★ **Zodiac** Several variations on "doing something other than...", demonstrating different degrees of restriction.

§7.10. The noun and the second noun

Once we begin applying rules to actions which are not entirely known in advance, we have a problem: there's no way to find out what specifically is happening. Consider the following:

```
Instead of examining something, say "It is none of your concern!"
```

This is fine as far as it goes, but clumsy. What if the player had examined a human being? Then "it" would be inappropriate. A better approach would be this:

```
Instead of examining something, say "[The noun] is none of your concern!"
```

The "noun" and, when necessary, the "second noun" are values which can be used in any rule about actions, and it follows that they can also be substituted into text, as this example demonstrates. Results might include:

```
Lady Bracknell is none of your concern!
```

```
The silver cigarette case is none of your concern!
```

This seems a good moment to mention that if you use "The" in a substitution, then a capitalised "The" will be used so long as this is grammatically correct (Lady Bracknell, as a proper noun, takes no article); "the" becomes a lower-case "the" along the same lines; and "a" a lower-case indefinite article.

```
Instead of examining something in the Drawing Room, say "Under Lady Bracknell's eye,
you feel constrained. Besides, it is only [a noun]."
```

⬆ Start of Chapter 7: Basic Actions

⬅ Back to §7.9. All actions and exceptional actions

➡ Onward to §7.11. In rooms and regions

⬇ Example 97: ★ **Ming Vase** ATTACK or DROP break and remove fragile items from play.

§7.11. In rooms and regions

Three elaborations of action descriptions increase the range of possibilities further.

```
Instead of taking something in the Supernatural Void, say "In this peculiar mist you feel
unable to grasp anything."
```

Like the objects to which the action applies, this location - the "in" clause - can take any description, not just an explicit place like "Supernatural Void":

```
Instead of listening in a dead end, say "You strain to hear further clues as to the course
of the underground river, but to no avail."
```

But we often want a rule to apply in any of a set of rooms: and where, unlike the "dead end" example above, the rooms have nothing much in common except where they happen to lie on a map. For instance, we might want a rule to apply only inside a given building, or a garden consisting of five miscellaneous rooms. If so, we can create a "region" as a convenient way to refer to that group of rooms:

> The Arboretum is east of the Botanical Gardens. Northwest of the Gardens is the Tropical Greenhouse.
>
> The Public Area is a region. The Arboretum and Gardens are in the Public Area.
>
> Instead of eating in the Public Area, say "The curators of the Gardens are ever among you, eagle-eyed and generally cussed."

Start of Chapter 7: Basic Actions

Back to §7.10. The noun and the second noun

Onward to §7.12. In the presence of, and when

§7.12. In the presence of, and when

Relative location can also be important: relative to other people, that is -

> Instead of eating something in the presence of Lady Bracknell, say "Lady Bracknell disapproves thoroughly of gentlemen who snack between meals, and there are few disapprovals in this world quite so thorough as Lady Bracknell's."

As might be guessed, this applies when the action takes place in the same location as the person named: and of course that person can also be described more vaguely ("... in the presence of a woman", say), and can just as easily be an inanimate thing ("... in the presence of the radio set").

Lady Bracknell is a pushover compared to some matriarchs:

> Instead of doing something other than looking, examining or waiting in the presence of the Queen: say "I'm afraid they take what you might call a zero tolerance approach to breaches of court etiquette here."; end the story saying "You have been summarily beheaded".

The last of the optional clauses we can tack on to the description of an action is the most general of all. We can add "when" and then any condition at all, as in:

> Instead of eating something when the radio set is switched on, say "Something about the howling short-wave static puts you right off luncheon."

This supposes that the radio is so loud that it can be heard from any room: we could muffle it so that it's only audible from the room it is in like so:

```
Instead of eating something in the presence of the radio set when the radio set is
switched on, say "Something about the howling short-wave static puts you right off
luncheon."
```

⬆ Start of Chapter 7: Basic Actions

⬅ Back to §7.11. In rooms and regions

➡ Onward to §7.13. Going from, going to

⬇ Example 98: ★ **Beachfront** An item that the player can't interact with until he has found it by searching the scenery.

⬇ Example 99: ★★★ **Today Tomorrow** A few notes on "In the presence of" and how it interacts with concealed objects.

§7.13. Going from, going to

Going is an action defined like any other: it is the one which happens when the player tries to go from one location to another. But it is unlike other actions because it happens in two locations, not just one, and has other complications such as vehicles and doors to contend with. To make it easier to write legible and flexible rules, "going" is allowed to be described in a number of special ways not open to other actions, as demonstrated by the following example story:

```
"Going Going"

The Catalogue Room is east of the Front Stacks. South of the Catalogue Room is the
Musicology Section.

Instead of going nowhere from the Front Stacks, say "Bookcases obstruct almost all
passages out of here."

Instead of going nowhere, say "You really can't wander around at random in the Library."

Before going to the Catalogue Room, say "You emerge back into the Catalogue Room."
```

Note that "going nowhere" means trying a map connection which is blank, and if no rules intervene then "You can't go that way" is normally printed. Unless "nowhere" is specified, descriptions of going apply only when there is a map connection. So "going from the Musicology Section" would not match if the player were trying to go east from there, since there is no map connection to the east. Similarly, "going somewhere" excludes blank connections.

The places gone "from" or "to" can be specific named regions instead of rooms. This is convenient when there are several different ways into or out of an area of map but a common rule needs to apply to all: so, for example,

```
Before going from the Cultivated Land to the Wilderness, ...
Before going nowhere from the Wilderness, say "Tangled brush forces you back."
```

Note that it must be "going nowhere from the Wilderness", not "...in the Wilderness". (Note also the caveat that the regions must be named: "going from a region", or something similarly nonspecific, will not work.)

An important point about "going... from" is that, as mentioned in general terms above, it requires that there is actually a map connection that way: whereas "going... in" does not. Suppose there is no map connection north from the Wilderness. Then:

```
Instead of going north from the Wilderness, say "You'll never read this."
Instead of going north in the Wilderness, say "Oh, it's too cold."
```

The first of these never happens, because it is logically impossible to go north from the Wilderness: but the second does happen. (Technically, this is because "going north" is the action, and "in the Wilderness" a separate condition tacked onto the rule.) This distinction is often useful - it allows us to write rules which apply only to feasible movements.

This may be a good place to mention a small restriction on the ways we can specify an action for a rule to apply to, and how it can be overcome. The restriction is that the action should only involve constant quantities, so that the following does not work:

```
The Dome is a room. The Hutch is north of the Dome. The rabbit is in the Hutch. Before
going to the location of the rabbit, say "You pick up a scent!"
```

because "the location of the rabbit" is a quantity which changes in play (the player can pick up the rabbit and take him to the Dome, for instance). However, we can get around this restriction by defining a suitable adjective, like so:

```
The Dome is a room. The Hutch is north of the Dome. The rabbit is in the Hutch.
Definition: a room is rabbit-infested if it is the location of the rabbit. Before going to a
rabbit-infested room, say "You pick up a scent!"
```

⬆ Start of Chapter 7: Basic Actions

⬅ Back to §7.12. In the presence of, and when

➡ Onward to §7.14. Going by, going through, going with

⬇ Example 100: ★ **Veronica** An effect that occurs only when the player leaves a region entirely.

⬇ Example 101: ★★ **A&E** Using regions to block access to an entire area when the player does not carry a pass, regardless of which entrance he uses.

⬇ Example 102: ★★★ **Bumping into Walls** Offering the player a list of valid directions if he tries to go in a direction that leads nowhere.

⬇ Example 103: ★★★ **Polarity** A "go back" command that keeps track of the direction from which the player came, and sends him back.

§7.14. Going by, going through, going with

Adding to the previous example story, we apply rules which depend on travelling by a particular vehicle:

```
The book trolley is in the Musicology Section. "The book trolley, a sort of motorised tractor for trundling around through the stacks, is parked here." The trolley is a vehicle. Instead of going nowhere by the trolley, say "Don't go crashing the trolley into walls."

Instead of going to the Front Stacks by the trolley, say "The Front Stacks are far too confined for the trolley to manoeuvre into them."
```

And, lastly, rules which apply to movements through particular doors:

```
The green baize door is east of the Catalogue Room and west of the Clerk's Office. The green baize door is an open door.

Before going through the green baize door, say "Through you go..." After going through the green baize door: try looking; say "...and here you are."
```

(Note that these apply whether the action is "going east" or "entering the green baize door", each having the same effect.) The last rule is worth a second look: the normal way that a "going" action is reported is to produce the room description of the new location. So if an "after" rule stops the action before we get to reporting, we have to produce any room description by hand (hence the "try looking" to cause the looking action). Alternatively, we could simply say something and let the normal course of events take place:

```
After going through the green baize door: say "...and here you are:"; continue the action.
```

Finally, going is an action which can also happen while the player is pushing something from one room to another, and we can describe this like so:

```
Instead of going from the Office with the trolley, say "But it looks perfectly placed here. Why push any further?"
```

"Going" is not the only action which moves the player. Another is "exiting", an action which moves the player out of whatever he/she is currently in or on. This action is often caused by the player typing just OUT or GET DOWN, and there's no noun as such. But Inform allows the syntax "exiting from" to make it easier to write rules about the exiting of particular containers or supporters:

```
After exiting from the Mini Cooper:
    say "You painstakingly unpack your limbs from the tiny car."
```

Start of Chapter 7: Basic Actions

Back to §7.13. Going from, going to

Onward to §7.15. Kinds of action

Example 104: ★ **No Relation** A car which must be turned on before it can be driven, and can only go to roads.

Example 105: ★ **Mattress King** Adding extra phrasing to the action to PUSH something in a direction.

Example 106: ★★★ **One Short Plank** A plank bridge which breaks if the player is carrying something when he goes across it. Pushing anything over the bridge is forbidden outright.

Example 107: ★★★★ **Provenance Unknown** Allowing something like PUSH TELEVISION EAST to push the cart on which the television rests.

Example 108: ★★★★ **Zorb** Replacing the message the player receives when attempting to push something that isn't pushable, and also to remove the restriction that objects cannot be pushed up or down.

§7.15. Kinds of action

Especially when people need to react to events going on around them, it is helpful to be able to categorise actions into whole areas of behaviour. For instance:

```
Kissing Mr Carr is unmaidenly behaviour.
Doing something to the painting is unmaidenly behaviour.

Instead of unmaidenly behaviour in the Inn, say "How unmaidenly!"
```

Here a new kind of action called "unmaidenly behaviour" has been created and then used in the description of an instead rule. The convenience of this approach is that when further actions suddenly occur to us as also being unmaidenly - say, attacking Mr Carr - we only need to add a single line:

```
Attacking Mr Carr is unmaidenly behaviour.
```

And this will automatically be reflected in any rules which concern the consequences of failing to be ladylike.

(Note that we were only allowed to say that "Kissing Mr Carr is unmaidenly behaviour." because Inform already knew from earlier sentences - see the example below - that Mr Carr was a person, and therefore that "kissing Mr Carr" made sense as a description of an action.)

Start of Chapter 7: Basic Actions

Back to §7.14. Going by, going through, going with

Onward to §7.16. Repeated actions

Example 109: ★ **Dearth and the Maiden** Our heroine, fallen among gentleman highwaymen, is restrained by her own modesty and seemliness.

Example 110: ★★★ **Mimicry** People who must be greeted before conversation can begin.

§7.16. Repeated actions

We come at last to the final thing which can be specified about an action, and appropriately enough it must be specified with the final words of the description. This is the way to talk about repeated activity:

```
Instead of examining the tapestry for the third time, say "All right, so it's a masterpiece, but is this really the time to make a detailed study?"

Instead of examining the urn at least twice, say "It's an urn. What do you want from me?"

Instead of going nowhere for the 20th time, say "Do stop walking into walls, there's a good fellow."
```

Note that we are allowed to spell out numbers up to twelve in English words, but beyond that must use digits (thus "twelfth" is allowed but not "thirteenth": "13th" should be used instead). The following example is instructive:

```
Instead of taking something for the fourth time, say "No. I'm capricious."
```

This means that it is the fourth time a "taking..." action has been tried, and does not mean that the same item was taken each time. Also, note that we are counting the number of times the action has been tried, not the number of times it succeeded.

Start of Chapter 7: Basic Actions

Back to §7.15. Kinds of action

Onward to §7.17. Actions on consecutive turns

Example 111: ★ **Y ask Y?** Noticing when the player seems to be at a loss, and recommending the use of hints.

Example 112: ★★★★ **A Day For Fresh Sushi** A complete story by Emily Short, called "A Day for Fresh Sushi", rewritten using Inform 7. Noteworthy is the snarky commenter who remarks on everything the player does, but only the first time each action is performed.

§7.17. Actions on consecutive turns

We can also reckon the number of consecutive turns on which an action has been repeated, by talking about "turns" instead of "times", as demonstrated in the following example story. Note also that we are allowed to use the phrase "doing it" to mean "the same description as the previous one except for the part about turns or times", like so:

```
"Waiting Room"

The Antechamber is a room. The tattered copy of Women's Wear Daily is in the Antechamber. Instead of taking the Daily, say "It is stamped NOT TO BE TAKEN AWAY."

Instead of examining the Daily for the first time, say "The best article seems to be about how your star sign affects your best swimsuit colour. Really: that's the best article."

Instead of doing it for the second time, say "You now know a generous amount about a typical week in the life of a weather forecaster."

Instead of doing it for the third time, say "You would now know how to cook herb bread, except that you have already forgotten the names of both of the herbs."

Instead of doing it more than three times, say "Nope, you've drained it of all conceivable sustenance, even the small ads about French farmhouses to let (sleeps 7) and breast reduction surgery (with alarming photographs in sallow light)."

After waiting for four to six turns, say "This is getting mighty dull." After waiting for seven to eight turns, say "Really, exceptionally dull." After waiting for nine turns, end the story saying "You have died of boredom, something previously thought medically impossible".
```

Note once again that numbers above twelve must not be written out, so "more than twelve times" would be acceptable, but we would write "more than 13 times".

Start of Chapter 7: Basic Actions
Back to §7.16. Repeated actions
Onward to §7.18. Postscript on actions

§7.18. Postscript on actions

In this chapter, all actions have been carried out by the player, all have been drawn from the standard stock of built-in actions ("unlocking", "taking", "going" and so forth), and all of those built-in actions have been allowed to work in the standard way - we have seen how to prevent the taking of something, and how to give this unexpected consequences, but not how to make taking work in an entirely different way.

All three of those restrictions will later be lifted in the chapter on "Advanced Actions", but otherwise we have covered the ground thoroughly, and it is time to move on to the techniques enabling us to do more than make tart replies to the player: it is time to change the world.

Start of Chapter 7: Basic Actions
Back to §7.17. Actions on consecutive turns
Onward to Chapter 8: Change: §8.1. Change of values that vary

Examples from Chapter 7: Basic Actions

Start of this chapter
Chapter 8: Change
Indexes of the examples

84 ★ **Example Grilling** RB

A grill, from which the player is not allowed to take anything lest he burn himself.

Descriptions of objects can be used in "Instead" rules: we can not only say "Instead of taking the steak", but also "Instead of taking something" or "Instead of taking something which is on the grill".

That last rule is useful if, for example, we want to prevent the player from interacting with anything on a specific supporter:

```
"Grilling"

The Patio is a room. The Patio contains a grill and an ice chest. The ice chest contains a cold beer. On the grill are a steak and a hot dog.

Mom is a woman in the Patio.

Instead of taking something which is on the grill:
    say "'Hey, you'll burn yourself,' says Mom."

Test me with "get steak / get all from grill / get all".
```

We could just as easily adapt this rule to affect a container: "Instead of taking something which is in the ice chest," for example.

Note also that in older versions of Inform, the pattern "get all from..." was treated differently from "get steak", and had to be accounted for separately. This is no longer the case; this instead of taking... rule will handle all the phrasings which the player might use to try to acquire this object.

85 ★ **Example Bad Hair Day** RB

Change the player's appearance in response to EXAMINE ME.

```
"Bad Hair Day"

The Foyer is a room. "A mirror hangs over the table, tempting you to check your appearance before going in with all the others."

Instead of examining the player:
    say "Oh, stop fussing. You look fine."

Test me with "examine me".
```

86 ★★★ **Example** **Democratic Process** RB

Make PUT and INSERT commands automatically take objects if the player is not holding them.

"Stop" and "Continue" are most useful when we need to write rules that will have to stop the action some of the time but at other times let it pass; so for instance:

```
"Democratic Process"

Before inserting something which is not carried by the player into something:
    if the noun is in the second noun, say "Already done." instead;
    say "(first taking [the noun])[line break]";
    silently try taking the noun;
    if the player is not holding the noun, stop the action.

Before putting something which is not carried by the player on something:
    if the noun is on the second noun, say "Already done." instead;
    say "(first taking [the noun])[line break]";
    silently try taking the noun;
    if the player is not holding the noun, stop the action.

The Assembly Room is a room. "On most days, this room is used for elementary school assemblies; at the moment, it serves as a voting place." The ballot is on the desk. The desk is in the Assembly Room.

The machine is a container in the Assembly Room. "On the ballot machine is a sign which reads 'PUT BALLOTS IN ME :)'." Understand "ballot machine" as the machine.

Test me with "put ballot in machine".
```

87 ★★★★ **Example** **Sand** RB

Extend PUT and INSERT handling to cases where multiple objects are intended at once.

The above example does not quite work when we want the player to be allowed to take multiple objects at once before putting them somewhere: we also need to add a couple of "understand" rules borrowed from many chapters later. While the reasons may not be immediately clear, we include the demonstration here for the sake of thoroughness:

```
"Sand"

Before inserting something which is not carried by the player into something:
    if the noun is in the second noun, say "Already done." instead;
    say "(first taking [the noun]) ";
    silently try taking the noun;
    if the player is not holding the noun, stop the action.

Before putting something which is not carried by the player on something:
    if the noun is on the second noun, say "Already done." instead;
    say "(first taking [the noun])[line break]";
    silently try taking the noun;
    if the player is not holding the noun, stop the action.

Understand "put [things] in [something]" as inserting it into. Understand "put [things] on [something]" as putting it on.

The Closet is a room.

A lentil is a kind of thing. A black-eyed pea is a kind of thing. The closet contains 3 lentils. The Closet contains 14 black-eyed peas. The round tin is a container in the closet. The round tin contains 17 lentils. The square tin is a container in the Closet. The square tin contains 20 black-eyed peas.

Sorting is a scene. Sorting begins when play begins. Sorting ends when all the lentils are in the round tin and all the black-eyed peas are in the square tin. When Sorting ends, end the story finally.

When play begins: say "Thanks to your cruel stepmother, you're not going anywhere until the lentils and peas are sorted."

Test me with "put peas in square tin / put lentils in round tin".
```

88 ★ **Example Fine Laid** *RB*

Making writing that can be separately examined from the paper on which it appears, but which directs all other actions to the paper.

Sometimes it is useful to direct all -- or almost all -- actions from one object to another. For the sake of argument, say we have a sheet of paper with writing on it, and (because we're very meticulous) we want to let the player examine the writing and get a customized response, different from when he just examines the sheet of paper. But for all other purposes -- say, TAKE or TASTE -- we want the two objects to be treated as one.

Here, we approach the problem by changing the noun and/or the second noun of the current action, then issuing a new command to "try the current action". Because we've changed the noun and second noun, the "current action" at this point is different from the one generated originally by the player's command.

```
"Fine Laid"

High Street Stationer is a room.

The sheet of paper is a thing in High Street Stationer. The writing is part of the
sheet of paper.

The description of the sheet of paper is "A beautiful sheet of heavy cream
paper." The description of the writing is "Delicate and spidery."

Instead of tasting the sheet of paper, say "You might need more fiber in your
diet, but this isn't the way.".

Before doing something other than examining when the current action involves
the writing:
    if the writing is the noun, now the noun is the sheet of paper;
    if the writing is the second noun, now the second noun is the sheet of paper;
    try the current action instead.

Test me with "examine sheet of paper / examine writing / get writing / taste
writing".
```

---

89 ★ **Example Hayseed** RB

A refinement of our staircase kind which can be climbed.

Presumably all staircase-type connections between rooms should respond when the player says CLIMB STAIRS (or the equivalent). So

```
"Hayseed"

A staircase is a kind of door. A staircase is usually open. A staircase is seldom
openable.

The ladder is a staircase. It is above the Barn and below the Hayloft.

Instead of climbing a staircase:
    try entering the noun.

Test me with "climb ladder / g".
```

Attempts to climb other types of door will still be treated as useless.

90

★ Example **Morning After**

RB

When the player picks something up which he hasn't already examined, the object is described.

Suppose we want to make the player's life slightly easier by examining everything he picks up, if he hasn't already examined it.

```
"Morning After"

A thing can be examined or unexamined.

After taking something unexamined:
    say "Taken. [run paragraph on]";
    try examining the noun.

Carry out examining something:
    now the noun is examined.
```

Carry out rules are explained in more detail in the chapter on advanced action handling. For now, it may be enough to know that what we put into this carry out rule for examining will happen any time anything is examined, but that it will not interfere with the rest of the predefined behavior of the action. The player will still see the object description and so on, as usual.

```
The Red Door Saloon is a room. "This old place is in pretty bad shape since the
mine shut down. Now there's not much to see but the pair of deep gouges in the
floorboards where they dragged away the Sheriff's corpse with the spurs still on."

Jed is a man in the Red Door Saloon. "At 8:30 AM the only person around is old
Jed, collecting his hangover cure."

The pistol is a thing in the Red Door Saloon. The description of the pistol is "It
ain't too accurate, but for two dollars you can't expect much."

The hangover cure is a thing in the Red Door Saloon. The description of the
hangover cure is "Two yellow egg-yolks unbroken in a red-brown liquid. Yep."

Test me with "x pistol / get all".
```

91

★ Example **Sybil 1**

RB

Direct all ASK, TELL, and ANSWER commands to ASK, and accept multiple words for certain cases.

Sometimes we do not particularly want to deal with all the variations on asking, telling, or answering someone something, but want to direct everything to a single conversational command:

```
"Consulting the Oracle"

The Grove is a room. In the Grove is a woman called the Sybil.

Instead of telling someone about something, try asking the noun about it.
Instead of answering the noun that something, try asking the noun about it.

Instead of asking the Sybil about "persians", say "She nods gravely."
```

And similarly, a difference between GIVE and SHOW is sometimes overkill:

```
Instead of showing something to someone, try giving the noun to the second noun.

The player carries a coin. Instead of giving the coin to the Sybil: move the coin to the Sybil; say "She accepts with a smile."
```

It is also often the case that we want to accept more than one form of a term. For instance

```
Instead of asking the Sybil about "Darius/king", say "Her smile unnerves you."
```

will match either "Darius" or "king". If necessary, we can go a step further and define our own token to match a variety of phrases, like this:

```
Understand "Athenians/Spartans/Greeks" or "hoplite army/forces" as "[Greeks]".
Instead of asking the Sybil about "[Greeks]", say "She looks encouraging."
```

The token "[Greeks]" will match all of "Athenians", "Spartans", "Greeks", "hoplite army", or "hoplite forces". It will not match "hoplite" or "forces" alone; it is important to note that the / divides individual words which are understood equivalently, but does not define entire phrases as equivalent. More about how Inform understands specific phrases can be found in the chapter on Understanding.

```
Test me with "test one / test two".

Test one with "ask sybil about persians / tell sybil about persians / sybil, persians / ask sybil about darius / ask sybil about king".

Test two with "ask sybil about greeks / ask sybil about athenians / ask sybil about hoplite army / ask sybil about hoplite forces / give the coin to the sybil".
```

92

★ Example Lucy

RB

Redirecting a question about one topic to ask about another.

Occasionally we will want to replace the player's question topic with another of our own devising. We can do this in the simplest possible case like so:

```
"Lucy"

The International Boardgame Championship is a room. Lucy is a woman in the Championship.

Instead of asking Lucy about "checkers":
    try asking Lucy about "games".

Instead of asking Lucy about "games",
    say "'I don't like games,' she sniffs."

Test me with "ask lucy about checkers / ask lucy about games".
```

Note that this syntax did not work in older versions of Inform; it is now safe.

---

93 ★★★ Example Sybil 2

RB

Making the character understand YES, SAY YES TO CHARACTER, TELL CHARACTER YES, ANSWER YES, and CHARACTER, YES.

Inform already understands YES, NO, and SORRY as commands in their own right, which can make things a little sticky when we want a character to ask a question of the player. The most important thing is not to cover some of the possible phrasings while ignoring others.

```
"Replies"

The Grove is a room. In the Grove is a woman called the Sybil.

Instead of asking the Sybil to try saying no: try saying no. Instead of asking the Sybil to try saying yes: try saying yes. Instead of asking the Sybil to try saying sorry: try saying sorry.

Instead of answering the Sybil that "yes", try saying yes. Instead of answering the Sybil that "no", try saying no. Instead of answering the Sybil that "sorry", try saying sorry.

Instead of saying yes in the presence of the Sybil:
    say "She looks interested."

Instead of saying no in the presence of the Sybil:
    say "She looks annoyed."

Instead of saying sorry in the presence of the Sybil:
    say "She looks bored."
```

The complexity arises from the fact that we want to handle both YES and SYBIL, YES. If we only had the latter, 'yes' would be treated as a text given to the Sybil, just as in the commands SAY YES TO SYBIL or ANSWER YES. But because we have defined it as a command (so that the player can use it independently), SYBIL, YES is understood as an order to the Sybil to do the YES action.

Fortunately, we can redirect everything, as here, so that the results wind up the same.

And if we want yet another variation not covered by the Inform standard:

```
Understand "tell [someone] [text]" as answering it that. Understand "tell [someone] that [text]" as answering it that.
```

But that is a matter for a later chapter.

```
Test me with "yes / sybil, yes / say yes to sybil / answer yes / tell sybil yes / no / sybil, no / say no to sybil / answer no / tell sybil no / sorry / sybil, sorry / say sorry to sybil / answer sorry / tell sybil sorry".
```

94

★★★ Example Costa Rican Ornithology

RB

A fully-implemented book, answering questions from a table of data, and responding to failed consultation with a custom message such as "You flip through the Guide to Central American Birds, but find no reference to penguins."

The following relies on quite a number of features we haven't met yet: tables, rules for printing names, instructions for understanding the player's commands. It is offered simply as an example of how a fully implemented book might be handled in Inform.

```
"Costa Rican Ornithology"

A book is a kind of thing. Understand "book" as a book. A book has a table name called the contents.

Instead of consulting a book about a topic listed in the contents of the noun:
    say "[reply entry][paragraph break]".

Report consulting a book about:
    say "You flip through [the noun], but find no reference to [the topic understood]." instead.
```

With this "topic understood" phrase, we're telling Inform to print back the word or phrase that the player was attempting to look up. This overrides the more general default response, "You discover nothing of interest in the book."

We now have the essential elements to construct whatever books we like. Now let's have an example of a specific book:

```
The Guide to Central American Birds is a book carried by the player. The contents of the Guide is the Table of Listed Birds.
```

We will come back to the idea of tables and table names later, but for now the important thing is that we have instructed Inform to look up its answers to consulting the bird guide in this form:

```
Table of Listed Birds
```

topic	reply
"[red]" or "[red] bird/macaw"	"You flip through the Guide for a while and eventually discover a reference to the [scarlet macaw], which appears to correspond with what you see before you."
"quetzal/trogon" or "resplendent trogon"	"The entry on the quetzal is quite lyrical, describing its brilliant plumage, flashing and igniting in the sunshine, which is supposedly sufficient to lure birdwatchers from all over the world. Unfortunately, the quetzal is described as being bright emerald in color, with a pink fuzz on its head and a long soft tail 'like a feather boa'. None of these describes your visitor."

The topic column is a bit special: it matches the player's input, and is not meant to be printed out again. Topic columns will be discussed further in the chapter on Tables. (Note also that, however it may appear in the documentation, the topic column should not be spanning multiple lines in our source text.)

We may also compress long or complicated topics by creating bracketed abbreviations, and in fact it's useful to do so now, to explain the red token we just used:

```
Understand "red-orange" or "bird" or "red" or "orange" as the scarlet macaw.
Understand "red-orange" or "red" or "orange" or "scarlet" as "[red]".
```

This technique is discussed further in the chapter on Understanding.

If we wanted more books, we could define those in the same way, giving each its own separate contents table to be used for consultation. But for the sake of the example we will keep it simple, and move on to the scenario itself:

```
The Veranda is a room. "From here you can see a considerable expanse of dense-growing jungle plants, and eventually the open water beyond."

The scarlet macaw is an animal in the veranda. "A vibrantly-colored [scarlet macaw] perches on the rail."

A thing can be known or unknown.

Before printing the name of the scarlet macaw while consulting:
    now the scarlet macaw is known.

Rule for printing the name of the unknown scarlet macaw: if the macaw is unknown, say "red-orange bird of unknown species".

Test me with "look up penguins in the guide / look up quetzal in guide / look up silver nuthatches in the guide / look / look up red bird in the book / look".
```

95 ★★★ **Example** **The Art of Noise** RB

Things are all assigned their own noise (or silence). Listening to the room in general reports on all the things that are currently audible.

This example involves redesigning the LISTEN command, removing its built-in function and replacing that with something more ambitious. We will learn more about how to do this later on.

```
"The Art of Noise"

A thing has some text called sound. The sound of a thing is usually "silence".

The report listening rule is not listed in the report listening to rules.

Carry out listening to something:
    say "From [the noun] you hear [the sound of the noun]."

Instead of listening to a room:
    if an audible thing can be touched by the player, say "You hear [the list of audible things which can be touched by the player].";
    otherwise say "Nothing of note."

Definition: a thing is audible if the sound of it is not "silence".

Before printing the name of something audible while listening to a room:
    say "[sound] from the "

The Sharper Image is a room. The pet rock is a thing in the Sharper Image. The toy car is a thing in the Sharper Image. The sound of the car is "whirring and zooming". The plastic widget is a thing in the Sharper Image. The sound of the plastic widget is "bleeps and bloops". The pointless gadget is a thing in Sharper Image. The sound of the pointless gadget is "buzzbuzzbuzz".

The soundproof case is a transparent openable container in the Sharper Image. It is closed and fixed in place.

Test me with "listen / listen to rock / listen to car / get all / open case / put all in case / listen / close case / listen / listen to car".
```

96

★ Example Zodiac

RB

Several variations on "doing something other than...", demonstrating different degrees of restriction.

Notice that the following two scenarios do not have the same effect:

```
"Zodiac"

The Secluded Alley is a room. The Capricorn Killer is a man in the Secluded Alley.

The player carries a can of mace and a roll of duct tape.
```

```
Instead of doing something other than examining with the Capricorn Killer:
    say "You wouldn't dare!"

Test me with "x me / x killer / touch killer / smell mace".
```

...will prevent the player from doing anything else to the killer, but allow him free range of action with other objects. By contrast, the following will prevent him doing anything other than examining to any item:

```
"Zodiac"

The Secluded Alley is a room. The Capricorn Killer is a man in the Secluded Alley.

The player carries a can of mace and a roll of duct tape.

Instead of doing something other than examining something in the presence of the Capricorn Killer:
    say "You dare not attempt it!"

Test me with "x me / x killer / touch killer / smell mace".
```

But notice that because we specified "examining something", actions that take no object (like sing) are still not affected.

The most strict statement would be

```
"Zodiac"

The Secluded Alley is a room. The Capricorn Killer is a man in the Secluded Alley.

The player carries a can of mace and a roll of duct tape.

Instead of doing something other than looking or examining in the presence of the Capricorn Killer:
    say "You dare not attempt it!"

Test me with "x me / x killer / touch killer / smell mace".
```

Because we left "something" out, now the pattern does not have to match an activity with an object; it will match any activity at all. Singing too will be deemed too risky in the presence of our sinister foe. But we do need to make an exception for "look", or else we won't get even the room description.

---

97 ★ **Example** **Ming Vase** *RB*

ATTACK or DROP break and remove fragile items from play.

In emulation of a certain annoying aspect of the original Adventure, in which there is a Ming vase that cannot safely be dropped:

```
"Ming Vase"

A thing can be strong or fragile. A thing is usually strong.

Instead of attacking or dropping a fragile thing:
    now the noun is nowhere;
    say "[The noun] breaks into thousands of pieces!"

The Cave is a room. The Ming vase is carried by the player. The vase is fragile.
```

We could also implement an additional refinement from Adventure, that a fragile thing is safe if dropped when there is a cushion nearby.

```
The pillow is a portable supporter. It is carried by the player.

Instead of dropping a fragile thing when the pillow is in the location: try putting
the noun on the pillow instead.

After putting a fragile thing on the pillow:
    say "You set [the noun] down gently on the pillow."

Test me with "drop pillow / drop vase / get vase / get pillow / drop vase".
```

In this rule, the pillow is the second noun; if we had a general rule about setting fragile things on soft things, we could say "You set [the noun] down gently on [the second noun]." for the same effect.

If we wanted to be more refined, we would provide extra code so that breaking a container or a supporter would leave behind their contents. We will see how to do that later.

---

98 ★ **Example** **Beachfront** *RB*

An item that the player can't interact with until he has found it by searching the scenery.

Suppose we have our player, a detective, searching for evidence; we don't want him to be able to use this evidence until he has performed the action that reveals it, but after that it should be visible in the room when he looks.

A simple way to do this is to start the object -- an envelope, in this scenario -- out of play, and only move it into the location when the player looks for it:

```
"Beachfront"
```

```
The Stuffy Office is a room. "The windows are closed, making the sultry air even
more unbearable. A narrow slice of Caribbean blue is visible between the scuba
gear rental shop and the recreated 17th century pirate tavern.

The office is cheerfully furnished with wicker chairs and white curtains, but the
tropical decorating scheme stopped at the desk, which is heavy oak and
absolutely covered with papers."

The heavy oak desk is a supporter in the stuffy office. It is scenery. Understand
"paperwork" as the desk.

The creamy envelope is an openable container. The description is "There is no
return address on the outside of the envelope, just the address of the Doctor's
office -- but the legs of the capital A are rubbed down in a characteristic way, and
the top of every R is open. There's no question that it comes from the same
typewriter as the blackmail note." In the envelope is a letter. The envelope can
be found or lost. The envelope is lost.

Instead of searching the desk when the envelope is lost:
    now the envelope is found;
    say "You rifle through the piles of bills and notices; invitations to conventions;
advertisements for high-end prescription drugs; pink carbon sheets bearing
patients['] names and medical identification numbers in spidery, elderly
handwriting. Almost at the bottom of the heap, you find what you were looking
for: a creamy envelope with the address typed.";
    move the envelope to the desk.
```

Here we've changed the property of the envelope to keep track of the fact that it has been found, so that if the player tries again, he won't find anything more.

```
Instead of searching the desk:
    say "Further investigation of the desk reveals nothing else suspicious."
```

Notice that we have two rules that apply to "searching the desk", but one of them has a more specific set of parameters ("when the envelope is lost"). This means that Inform will consult that rule first and use it if it applies; it will only carry out our plain vanilla "instead of searching the desk" rule when the more restricted rule is not relevant.

```
Test me with "x envelope / x desk / search desk / look / get envelope / x
envelope".
```

---

99 ★★★ **Example Today Tomorrow** RB

A few notes on "In the presence of" and how it interacts with concealed objects.

Rules about concealment will affect "in the presence of", too. For instance, suppose we have a man with a pocket pet:

```
"Today Tomorrow"
```

```
The Temporary Employment Office is a room. "'Tomorrow's Temporary Workers - Today!' proclaims the logo over the door. The office is divided into two areas, the inner sanctum where you take calls and fiddle with the computer, and the outer area where workers take skill exams and watch inspirational videos ('Earn your way to partial benefits!', 'Vacation days and you!', 'Temping the Tomorrow Way', etc.)."

Maya is a woman in the Office. "Your coworker Maya sits at her own computer, diligently modifying all the [one of]pay rates in the database down from $9.00 an hour to $8.96[or]billing rates in the database up from $25.00 an hour to $25.04[purely at random]." She wears a trenchcoat. She carries a chihuahua. The description of Maya is "[if Maya is wearing the trenchcoat]She's wearing a trenchcoat, in a bizarre bid to keep your boss unaware of the chihuahua in her pocket. 'Because petsitters are really expensive!' she explained to you in an urgent hiss this morning over the coffeemaker. [otherwise]She looks cold. [end if]To all appearances, she is carrying [a list of unconcealed things carried by Maya]."

Maya's computer is scenery in the Employment Office. The description is "You can't see the screen from here, but she was perfectly happy to tell you what she was doing."

Rule for deciding the concealed possessions of someone (called carrier):
    if the particular possession is the chihuahua and the carrier wears the trenchcoat, yes;
    otherwise no.

Instead of eating something in the presence of the chihuahua:
    say "[The chihuahua] yips at you! Maya looks despairingly at [the noun], which is obviously inciting it."

The desk is scenery in the Office. On the desk are a multi-line telephone and a printer. The description of the printer is "Every morning, this instrument of torture spits out a list of the assignments you have to fill - professional, attractive receptionist with level three Excel certification, at $7.05 an hour; that sort of thing. You spend the ensuing three hours trying to meet its tyrannical demands." The description of the multi-line telephone is "Line three blinks urgently at you. You'll pick it up again as soon as you can remember who it was you put on hold."

The player carries a double bacon cheeseburger and a chocolate shake. Understand "milkshake" or "beverage" as the shake. The description of the cheeseburger is "A minor self-indulgence to make up for the fact that you have to work through lunch." The cheeseburger is edible. The shake is edible. The description of the shake is "It has the consistency of spackle and is no longer cold, but there is some chocolatey goodness in it still." Understand "burger" or "hamburger" as the cheeseburger. Instead of drinking the chocolate shake: try eating the shake instead.

Test me with "x maya / x cheeseburger / drink shake / eat cheeseburger".
```

Unless we somehow get the trenchcoat away from Maya, the chihuahua will not be in view, and will not intervene in our lunch. All very well for the player character, but not so interesting to the story... To this end, we might add an unfortunate event, courtesy of later chapters:

```
The time of day is 11:45 AM.

At 11:47 AM: say "Your boss pokes his head in, temporarily free of the round of
conference calls that occupy all his days. 'Maya,' he says. 'Your coat?' He
shakes his head, clucking sadly. 'It doesn't say professional!' But mercifully Maya
manages to take it off so slowly that he doesn't glimpse her pet before her phone
rings again.";
    now Maya carries the trenchcoat.
```

---

100 ★ Example **Veronica** RB

An effect that occurs only when the player leaves a region entirely.

Suppose that we want to have something happen when the player leaves a region we've defined. "Instead of going from (the region)..." will not suffice for this, because this rule will be invoked every time the player successfully leaves a room within the region, whether or not he is going to a room that is also in the same region.

Instead we need a rule that is a bit more specific, like this:

```
"Veronica"

Neptune is a region.

Tijuana is a room.

High School is north of Tijuana. It is in Neptune.

Detective Offices is west of High School. It is in Neptune.

The player is in High School.

Instead of going from Neptune to a room which is not in Neptune:
    say "It's a bad time to leave Neptune."

Test me with "s / w / e".
```

---

101 ★★ Example **A&E** RB

Using regions to block access to an entire area when the player does not carry a pass, regardless of which entrance he uses.

Rules about going to regions make it easy to exclude the player from a large portion of the map, even if there are many connecting paths to the region. For instance, in this story it would be annoying to have to write a rule about all four exits by which the player could reach the film set area:

```
"A&E"

Winding Street is a room. Winding Street is west of Duck Pond. Sloping Street is north of Winding Street, northwest of Duck Pond, west of Stately Lawn, and southwest of Stately Home. Stately Lawn is north of Duck Pond. Stately Home is north of Stately Lawn.

Film Set is a region. Duck Pond, Stately Lawn, and Stately Home are in Film Set.

Instead of going to Film Set when the player does not carry the VIP Pass: say "A burly studio guard materializes in your path, convincing you that you would prefer to be elsewhere."

The VIP Pass is in the garbage can. The garbage can is in Sloping Street.

After going to the Film Set:
    say "Success! At last you are inside the set of 'Prouder and More Prejudiced'. Next step: locating Mr Firth.";
    end the story finally.

Test me with "e / n / e / get pass / e".
```

102 ★★★ Example **Bumping into Walls** RB

Offering the player a list of valid directions if he tries to go in a direction that leads nowhere.

```
"Bumping into Walls"
```

First we add an instruction to determine which ways lead to other rooms.

```
Definition: a direction (called thataway) is viable if the room thataway from the location is a room.
```

Now we build in the instruction for what Inform should say if the player tries to head in a direction that leads nowhere:

```
Instead of going nowhere:
    let count of exits be the number of viable directions;
    if the count of exits is 0, say "You appear to be trapped in here." instead;
    if the count of exits is 1, say "From here, the only way out is [list of viable directions].";
    otherwise say "From here, the viable exits are [list of viable directions]."
```

There is no theoretical reason why we have to define "count of exits" here: we could, if we wanted, just say "if the number of viable directions is 0", "if the number of viable directions is 1", and so on. However, each calculation of a "viable direction" takes a bit of computing power, so there is some slight savings in not requiring the game to count viable directions more than once in this routine.

```
Dome is a room. North of Dome is North Chapel. South of the Dome is South Chapel. West of the Dome is Western End. Quiet Corner is northwest of the Dome, north of Western End, and west of North Chapel. Loud Corner is east of North Chapel, northeast of Dome, and north of Eastern End. Eastern End is north of Dim Corner and east of Dome. Dim Corner is southeast of Dome and east of South Chapel. Ruined Corner is southwest of Dome, west of South Chapel, and south of Western End.

The Crypt is below the dome.

The church door is east of Eastern End and west of the Courtyard. The church door is a door.

Test me with "u / n / n / e / n / s / u / open door / e / n".
```

103

★★★ Example **Polarity**

A "go back" command that keeps track of the direction from which the player came, and sends him back.

RB

The main trick of this is always to record where the player has gone when he has just moved.

```
"Polarity"

The former location is a room that varies.
```

Here we record where the player has been before moving him; by calling this the "first carry out going rule", we make sure that this rule is followed during the going action before any other pieces of the movement occur. For more detail, see the chapters on advanced actions and on rules.

```
First carry out going rule:
    now the former location is the location.

Understand "go back" as retreating. Understand "back" or "return" or "retreat" as retreating.

Retreating is an action applying to nothing.

Carry out retreating:
    let way be the best route from the location to the former location, using doors;
    if way is a direction, try going way;
    otherwise say "You can't see an open way back."
```

And to deal with the case where the player has not yet moved:

```
When play begins: now the former location is the Dome.
```

```
Instead of retreating when the former location is the location: say "You haven't gone anywhere yet."

Dome is a room. North of Dome is North Chapel. South of the Dome is South Chapel. West of the Dome is Western End. Quiet Corner is northwest of the Dome, north of Western End, and west of North Chapel. Loud Corner is east of North Chapel, northeast of Dome, and north of Eastern End. Eastern End is north of Dim Corner and east of Dome. Dim Corner is southeast of Dome and east of South Chapel. Ruined Corner is southwest of Dome, west of South Chapel, and south of Western End.

The church door is east of Eastern End and west of the Courtyard. The church door is a door.

Test me with "back / n / go back / e / open door / go through door / go back".
```

104

Example No Relation

RB

A car which must be turned on before it can be driven, and can only go to roads.

We need to designate certain rooms as roads. Since the status of being a road will not change during play, we do this with a kind:

```
"No Relation"

A road is a kind of room. Definition: a room is offroad if it is not a road.

Instead of going by a vehicle (called the auto) to somewhere offroad:
    say "You can't drive [the auto] off-road."

Trafalgar Square is a road. "The Square is overlooked by a pillared statue of Admiral Lord Horatio Nelson (no relation), naval hero and convenience to pigeons since 1812."

The National Gallery is north of Trafalgar Square. The Strand is east of Trafalgar Square. The Strand is a road.

The car is a vehicle in Trafalgar Square. The ignition is a device. The ignition is part of the car. Instead of going by the car when the ignition is switched off: say "The ignition is off at the moment." Instead of switching on the car, try switching on the ignition. Instead of switching off the car, try switching off the ignition.

Test me with "get in car / n / e / turn on car / n / e / get out / w / n / s / e / get in car / turn off car / w / turn on ignition / w".
```

(In the course of the writing of Inform 7, much of Trafalgar Square was pedestrianised, making this example already out of date.)

A further technical note: notice "going by a vehicle" in the above rule, rather than "going by something". A rule such as "Instead of going by something..." will be matched whenever the player tries to go some direction while in an enterable object, whether or not that object is actually capable of movement. This is sometimes useful, but in this case we want the warning to apply only when the player is in a vehicle; if we added Trafalgar Square's statue bases to the scenario, we would not want

```
You can't drive the pedestal off-road.
```

So we restrict the rule to "Instead of going by a vehicle..."

105

★ Example **Mattress King**

RB

Adding extra phrasing to the action to PUSH something in a direction.

By default, when the player pushes something a direction, Inform checks to make sure that the object is pushable between rooms. If not, it blocks the action; if so, it carries out a normal going action with the pushed object taken along.

Also by default, this action produces only a description of the new room that we've traveled into. But suppose we would like to print a short message describing the pushing action first:

```
"Mattress King"

Monica's Bedroom is a room. The Living Room is south of Monica's Bedroom. Rachel's Bedroom is south of the Living Room.

After going a direction (called way-pushed) with something (called the thing-pushed):
    say "You push [the thing-pushed] [way-pushed] to [the location].";
    continue the action.

The race car bed is an enterable supporter in Monica's Bedroom. It is pushable between rooms.

Test me with "push bed south".
```

106

★★★ Example **One Short Plank**

RB

A plank bridge which breaks if the player is carrying something when he goes across it. Pushing anything over the bridge is forbidden outright.

```
"One Short Plank"

The East Jungle is a room. The plank bridge is west of the East Jungle and east of the West Jungle. The plank is an open unopenable door. "A precarious plank bridge extends [if the location is West Jungle]east[otherwise]west[end if] across the chasm." The description of the plank is "Extremely fragile and precarious."
```

```
Instead of going through the plank when the player is carrying something:
    say "You step gingerly across the plank, which bows under your weight. But your meagre possessions are the straw which breaks the camel's back!";
    end the story.

After going through the plank:
    say "You step gingerly across the plank, grateful that you're not burdened.";
    continue the action.

There is a feather in the East Jungle.
```

But indeed, why stop there?

```
The gigantic stone ball is a thing in the West Jungle. It is pushable between rooms.

Before going through the plank with something:
    say "Surely you jest." instead.

Test me with "w / e / w / push ball e / e / get feather / w".
```

107

★★★ Example Provenance Unknown

RB

Allowing something like PUSH TELEVISION EAST to push the cart on which the television rests.

Suppose we have a series of items that might be stacked on top of one another -- say a heavy television on a rolling cart, and we want the player to be able to move the cart with PUSH TELEVISION EAST just as well as with PUSH CART EAST.

This takes a little redirection, using a setting action variables rule. This is not a kind of rule we've encountered yet, and in fact we won't meet it until the Advanced Actions chapter; it is included here for the convenience of authors who want to modify the effect of pushing without reading that far ahead:

```
"Provenance Unknown"

Setting action variables for pushing something to:
    if the noun is enclosed by a pushable between rooms thing (called the pushed item) which is in the location:
        now the noun is the pushed item instead.
```

This rule says that any time we push an object that is on top of a stack of pushable objects, we should transfer the action to the item at the bottom of the stack.

The rest is merely a test case.

```
The heavy golden idol is on a roller board. The roller board is on a hovercraft.

The hovercraft, the tea trolley, and the skateboard are pushable between rooms.
The hovercraft is in Zeta Proximan Dig Field.

Zeta Proximan Dig Field is a room. "During the day, the field is massed with
sweating native workers, overseers, and officials from central command. Now
the spades, trowels, brushes, metal detectors, ground probes, plumb lines, and
sighting tripods have been laid aside.

All that remains are the trenches and the fine grey dust that blows slowly across
them; the moonlight; and the just-emerging outlines of an ancient and alien wall."

The Hover-Road is west of the Dig Field. "A long road hastily laid down,
stretching east to west, from the dig site toward the safety of the city."

When play begins:
    say "You have, at last, loaded your illicit cargo without setting off any of the
many and sensitive alarms set here; now it remains only to sneak out of the
area, under the light of Zeta Proxima's lone green moon."

Test me with "push idol west / look / push roller board east / look".
```

108 ★★★ **Example Zorb** *RB*

Replacing the message the player receives when attempting to push something that isn't pushable, and also to remove the restriction that objects cannot be pushed up or down.

There are two aspects of Inform's handling of pushable objects that are particularly prime for modification. One is that we may want to change the language used to refuse the pushing of unpushable objects.

Second, Inform by default assumes that it is impossible to push objects in up or down directions. This makes lots of sense if the player is trying to push a wheelbarrow up a ladder; it makes less sense if instead we're pushing a ball up a slope.

We solve both problems with some syntax borrowed from the chapter on rulebooks: in the first case, we replace the old rule with a new one with more friendly phrasing; in the second, we remove the rule entirely. More about how to do this is described in the rulebooks chapter; and in general we can find out what rules contribute to any given action by looking at the Actions index. In this case, the action is "pushing it to", which has its own set of prerequisites (called check rules) that make sure the object can safely be pushed, before turning processing over to the going action.

```
"Zorb"

Section 1 - Procedure

The new can't push unpushable things rule is listed instead of the can't push
unpushable things rule in the check pushing it to rules.
```

```
This is the new can't push unpushable things rule:
    if the noun is not pushable between rooms:
        say "[The noun] [are] not amenable to being pushed from place to place." instead.

The can't push vertically rule is not listed in any rulebook.
```

And now to provide a scenario where the player can push something up and down a hillside. Most of the rest of the example is there for local color and to provide a way to demonstrate these rule adjustments:

```
Section 2 - Scenario

The Steep Hill is a room. The Crest is above Steep Hill. The Valley is below Steep Hill.

The flat rock is a fixed in place thing in the Steep Hill.

The Zorb is a transparent open enterable container in the Steep Hill. "[if the player props the Zorb]The Zorb rests here, kept from further rolling by your support[otherwise]The Zorb is here[end if].". It is pushable between rooms. The description of the Zorb is "A giant plastic inflatable ball, like a hamster ball for humans[if someone is in the Zorb]. Inside [is-are list of people in the Zorb][end if]."

Lucy is a woman in the Zorb.

Carry out going with the Zorb when the Zorb contains Lucy:
    say "Lucy whoops delightedly as she rides along in the Zorb."

Every turn when the Zorb is not in the Valley and the player does not prop the Zorb:
    let next room be the room down from the location of the Zorb;
    if the player is not in the Zorb and the player can see the Zorb:
        say "The Zorb succumbs to gravity and rolls down toward [the next room].";
    move the Zorb to the next room;
    if the player is in the Zorb:
        say "The Zorb rolls you down the hill!";
        try looking;
    otherwise if the player can see the Zorb:
        say "The Zorb rolls ponderously but inevitably into the vicinity.";

Propping relates one person to one thing. The verb to prop means the propping relation.

Carry out going with the Zorb:
    now the player props the Zorb.

Before doing something when the action requires a touchable noun:
    if the noun is not the Zorb, now the player does not prop the Zorb.

Check waving hands when the player is propping something (called casualty):
    try the player releasing the casualty.
```

```
Carry out entering the Zorb:
    now the player does not prop the Zorb.

Understand "let go of [something]" or "let [something] go" or "release
[something]" or "free [something]" as releasing. Releasing is an action applying
to one thing.

Check releasing:
    if the player carries the noun:
        try dropping the noun instead.

Check releasing:
    if the player does not prop the noun:
        say "You are not supporting [the noun]." instead.

Carry out releasing:
    now the player does not prop the noun.

Report releasing:
    say "You let go of [the noun]."

Test me with "d / push zorb up / look / push zorb up / wave / d / d / push zorb up
/ release zorb / d / push zorb up / touch rock / push the flat rock south".
```

109 ★ **Example Dearth and the Maiden** RB

Our heroine, fallen among gentleman highwaymen, is restrained by her own modesty and seemliness.

The following example, indebted to the late Georgette Heyer, is suggestive:

```
"Dearth and the Maiden"

The Chequers Inn is a room. "The room is panelled and ceilinged in oak, with
blue curtains to the windows and blue cushions on the high-backed settle by the
fire."

An oil painting is in the Inn. "An oil painting hangs upon one wall, a lascivious
work from the Indies in which a very bendy, sloe-eyed courtesan - but no."

A man called Mr Carr is in the Inn. "Standing bashfully aside is one Mr Carr, who
we have been led to understand is by profession a Highwayman (yet whose
visage oddly recalls Lord John Carstares, disgraced eldest son of the Earl of
Wyncham)."

Kissing Mr Carr is unmaidenly behaviour. Doing something to the painting is
unmaidenly behaviour.

Instead of unmaidenly behaviour in the Inn, say "How unmaidenly! Why, one
might just as wantonly strip a rose of its petals, letting each fragrant leaf flutter
slowly to the ground."
```

```
Test me with "examine painting / take painting / kiss mr carr".
```

110 ★★★ **Example** **Mimicry**

People who must be greeted before conversation can begin.

RB

Suppose we want to add a sense of some conversational flow, so that the player is forced to acknowledge the presence of people before beginning detailed conversations with them. We collect all speech actions into a single category:

```
"Mimicry"

Asking someone about something is speech. Telling someone about something is speech. Answering someone that something is speech. Asking someone for something is speech.
```

And then write a general rule.

```
Before speech in the presence of an ungreeted person: try waving hands.
```

One complication is that "asking someone to try doing something", which describes commands such as FRED, GO SOUTH, cannot be made into a kind of action. This requires its own rule:

```
Before asking someone to try doing something in the presence of an ungreeted person: try waving hands.
```

Now we define what greetings are going to look like:

```
Check waving hands:
	unless the player can see someone who is not the player, say "You are alone." instead.

Carry out waving hands:
	say "You nod hello to [the list of ungreeted people who can be seen by the player].";
	now every ungreeted person who can be seen by the player is greeted.

The report waving hands rule is not listed in the report waving hands rulebook.
```

Because of the way we've defined the command, this will now also work if the player waves.

```
A person can be greeted or ungreeted. A person is usually ungreeted. The player is greeted.
```

And now the scenario:

```
The International Convention of Mimes is a room. Lester, Harold, Geoff, Kwame, and Peter are men in the Convention. Elouise is a woman in the Convention.
The Invisible Box is an enterable container in the Convention. "You can detect, from the way people keep leaning on it, an invisible box in the middle of the room."

Lester carries a bowler hat.

Instead of speech in the presence of someone:
    describe poor reception.

Definition: a person is other if it is not the player.

At 9:01 AM:
    move Phineas to the location; say "A mime called Phineas appears from the non-existent bathroom."

Phineas is a man.

A persuasion rule:
    describe poor reception;
    persuasion fails.

To describe poor reception:
    if the player is in the Invisible box,
        say "Everyone convulses with silent laughter as you try to shout from within the invisible box.";
    otherwise
        say "You attempt to convey your meaning with gesture and interpretive dance, but [the list of visible other people] scorn[if the number of visible other people is 1]s[end if] your performance, refusing to respond."

Test me with "ask lester about work / lester, east / ask lester for bowler / lester, nice not talking to you / get in box / ask lester for hat / phineas, east".
```

111 ★ **Example Y ask Y?** RB

Noticing when the player seems to be at a loss, and recommending the use of hints.

Suppose we'd like to watch for signs that the player is floundering, and if we see them, recommend that he try the hints. There are probably more sophisticated diagnostics, but as a first cut, let's assume that a player who repeatedly reviews descriptions of objects he's already seen, looks around the room, and takes inventory, is at a loss for more productive activities. So then...

```
"Y ask Y?"

A thing can be examined or unexamined. A thing is usually unexamined. Carry out examining something: now the noun is examined.

Taking inventory is acting confused. Looking is acting confused. Examining an examined thing is acting confused.
```

```
After acting confused for the sixth turn:
    say "(If you are feeling lost, try typing HELP for suggestions.)"
```

And now we write a scenario which will, alas, rather encourage even a deft and clueful player to play as though he were hopelessly confused:

```
The story headline is "or: Bad Author, No Biscuit".

The description of a thing is usually "Hm. [The item described] reminds you quite a lot of [a random visible thing which is not the item described]."

The Yurt is a room.

Food is a kind of thing. Food is always edible. In the Yurt are a yam and a dish of yakitori. The yam and the yakitori are food. The description of food is "Well, at least it's not [a random edible thing which is not the item described]."

In the Yurt is an animal called a yapok.

The player wears a yukata. The player carries a yataghan.

Every turn:
    if a random chance of 1 in 2 succeeds and something is examined:
        say "Your eye is attracted by some kind of surreptitious movement from [the random examined thing].";
    otherwise if the player carries something and a random chance of 1 in 3 succeeds:
        say "[The random thing carried by the player] tries to slip from your grasp."

Test me with "x yam / x yam / look / x yam / i / look / i / help / quit".
```

And finally a little dollop of perversity from a later chapter:

```
Check quitting the game:
    say "You're sure? ";
    if player consents, say "[line break]You were getting close to a breakthrough, you know.[line break]";
    otherwise stop the action.

Understand "help" as a mistake ("You're doing fine! Just keep at what you're doing now.").
```

112

★★★★ Example **A Day For Fresh Sushi**

RB

A complete story by Emily Short, called "A Day for Fresh Sushi", rewritten using Inform 7. Noteworthy is the snarky commenter who remarks on everything the player does, but only the first time each action is performed.

The following is an almost-completely-faithful rewrite of Emily Short's "A Day for Fresh Sushi", which was originally written using the (very different) Inform 6 programming language. The -- let us be honest and call it a gimmick -- of this game is the evil fish, who has some unpleasant remark to offer on pretty much every action. But the effect would wear off fast if he repeated himself, so these comments need to be single-use only.

Inform 7's repeated action syntax makes it much tidier to write the same scenario, so:

```
"A Day For Fresh Sushi" by Emily Short.

Use scoring.

The story headline is "Your basic surreal gay fish romance".

The Studio is a room. "[if visited]Decorated with Britney's signature flair. It was her innate sense of style that first made you forgive her that ludicrous name. And here it is displayed to the fullest: deep-hued drapes on the walls, the windows flung open with their stunning view of old Vienna, the faint smell of coffee that clings to everything. Her easel stands over by the windows, where the light is brightest.[otherwise]This is Britney's studio. You haven't been around here for a while, because of how busy you've been with work, and she's made a few changes -- the aquarium in the corner, for instance. But it still brings back a certain emotional sweetness from the days when you had just met for the first time... when you used to spend hours on the sofa...

You shake your head. No time for fantasy. Must feed fish.[end if]"

Instead of smelling the Studio:
	say "The evil fish notices you sniffing the air. 'Vanilla Raspberry Roast,' it remarks. 'You really miss her, don't you.'

You glance over, startled, but the fish's mouth is open in a piscine equivalent of a laugh. You stifle the urge to skewer the thing..."

Instead of jumping:
	say "'Er,' says the fish. 'Does that, like, EVER help??'"

Instead of going nowhere:
	say "You can't leave until you've fed the fish. Otherwise, he'll complain, and you will never hear the end of it."

The cabinet is an openable closed container in the Studio. It is fixed in place. "A huge cabinet, in the guise of an armoire, stands between the windows." The description is "Large, and with a bit of an Art Nouveau theme going on in the shape of the doors." Understand "armoire" as the cabinet.

Instead of looking under the cabinet for the first time:
	say "'Dustbunnies,' predicts the fish, with telling accuracy. It executes what for all the world looks like a fishy shudder. 'Lemme tell you, one time I accidentally flopped outta the tank, and I was TWO HOURS on the floor with those things STARING ME IN THE NOSE. It was frightening.'"
```

```
After opening the cabinet for the first time:
   say "'There ya go,' says the fish. 'The girl is getting WARMER.'"

After closing the cabinet for the first time:
   if the fish food is not found, say "'Ooh, what do you think, Bob? I think we're
going to have to dock the girl a few points. HAVE ANOTHER LOOK,
sweetcakes, there's a doll.'"

The cabinet contains some paints and some cloths. The description of the paints
is "A bunch of tubes of oil paint, most of them in some state of grunginess, some
with the tops twisted partway off."

After taking the paints for the first time:
   say "'Boy,' says the fish, apparently to himself, 'I sure hope that's some food
she's finding for me in there. You know, the yummy food in the ORANGE CAN.'"

After examining the paints for the first time:
   say "'Tons of useful stuff in there,' hollers in the fish, in a syncopated burble."

The description of the cloths is "Various colors of drapery that Britney uses to set
up backgrounds and clothe her models. She does a lot of portraiture, so this
comes in handy. It's all a big messy wad at the moment. Organized is not her
middle name." Understand "drapery" or "cloth" as the cloths. The indefinite
article of the cloths is "a heap of". [see 3.17]

Instead of searching or looking under the cloths for the first time:
   now the player is carrying the fish food;
   now the fish food is found;
   say "Poking around the cloths reveals -- ha HA! -- a vehemently orange can of
fish food."

Instead of showing the cloths to the fish:
   say "'What are you, some kind of sadist? I don't want to see a bunch of cloths!
What kind of f'ing good, 'scuse my French, is that supposed to do me? I don't
even wear pants for God's sake!'

He really looks upset. You start wondering whether apoplexy is an ailment
common to fish."

After examining cloths for the first time:
   say "'Whatcha looking at? I can't see through the doors, you know.'"

There is a can of fish food. Understand "canister" as the can. The description is
"A vehemently orange canister of fish food." The fish food can be found or
hidden. The fish food is hidden.

Instead of giving the can to the fish:
   say "'I don't want the whole can, GeniusChyk. Just feed me and we'll ALL be
happy, 'kay?"

Instead of showing the can to the fish:
   say "'That's the ticket, sweetie! Bring it on.'"

Instead of opening the can:
   say "'Oh, for--!' The evil fish breaks out in exasperation and hives. 'Screw the
```

screwing around with the screwtop. SHE never has to do that.'

'Well, SHE is not here,' you reply. 'What do you suggest?'

'>FEED FISH<' says the fish promptly, making fishy faces and pointing at you with his fin. 'Simplicity. Try it.'"

Instead of inserting the can into something:
    say "'HelLLLOOO,' screams the fish. 'Whatever happened to FEEDING MEEE?'"

The easel is a supporter in the Studio. It is scenery. On the easel is a painting. Understand "portrait" or "image" as the painting.

The description of the painting is "Only partway finished, but you can tell what it is: Britney's mother. You only met the old woman once, before she faded out of existence in a little hospice in Salzburg.

In the picture, her hands are grasping tightly at a small grey bottle, the pills to which she became addicted in her old age, and strange, gargoyle-like forms clutch at her arms and whisper in her ears.

But the disturbing thing, the truly awful thing, is the small figure of Britney herself, down in the corner, unmistakable: she is walking away. Her back turned.

You thought she'd finally talked this out, but evidently not. Still feels guilty for leaving. You only barely stop yourself from tracing, with your finger, those tiny slumped shoulders..."

Instead of taking the painting, say "No, you'd better leave it. It'd freak her out if you moved it."

Before examining the painting for the first time:
    say "A ferocious banging from the aquarium attracts your attention as you go to look at the painting. 'Hey!' screams the fish. 'She doesn't like strangers looking at her paintings before they're DOONNNE!'

'Shut up, you,' you reply casually. 'I'm not a stranger.' But the fish puts you off a little bit, and your heart is already in your mouth before you see the painting itself...".

Instead of examining the painting more than once:
    say "Once is really enough. It's pretty much embedded in your consciousness now."

After doing something to the painting:
    say "'So what's it of?' asks the fish, as you turn away. 'She never asks if I want to see them, you know?'

'Her mother,' you respond without thinking.

'Yeah? Man. I never knew my mother. Eggs, that's the way to go.'"

The window is scenery in the Studio. The window can be openable. The window can be open. It is openable and closed. Understand "windows" as the window. The description of the window is "[if open]Through the windows you get a lovely view of the street outside. At the moment, the glass is thrown open, and a light breeze is blowing through.[otherwise]Through the windows, you get a lovely view of the street outside -- the little fountain on the corner, the slightly dilapidated but nonetheless magnificent Jugendstil architecture of the facing building. The glass itself is shut, however.[end if]"

After opening the window for the first time:
    say "'Thank god some air,' says the fish. 'Man, it was getting hard to breathe in here.' Two beats pass. 'Oh wait.'"

The table is scenery in the Studio. On the table is a vase. The vase is an open container. It is not openable.

The description of the table is "A monstrosity of poor taste and bad design: made of some heavy, French-empire sort of wood, with a single pillar for a central leg, carved in the image of Poseidon surrounded by nymphs. It's all scaley, and whenever you sit down, the trident has a tendency to stab you in the knee. But Britney assures you it's worth a fortune." The description of the vase is "A huge vase -- what you saw once described in a Regency romance as an epergne, maybe -- something so big that it would block someone sitting at the table from seeing anyone else also sitting at the table. But it does function nicely as a receptacle for hugeass bouquets of flowers."

Instead of looking under the table for the first time:
    say "'You're not going to find anything down there,' whines the fish. 'I mean, c'mon. It's the fricking floor. Please tell me you can see that. I can see that. I'm a myopic fish in a tank ten feet away and I can tell you there is nothing there but floor.'"

After examining the table:
    say "'That there is MY PA,' says the fish, pointing at the scaley triton figure with one fin."

Instead of inserting something which is not the bouquet into the vase:
say "'Okay, so, what were you, raised in a barn? Normal folks like to use that for flowers. Or so I've observed.'"

After inserting the bouquet into the vase for the first time:
    say "You settle the flowers into the vase and arrange them so that they look sprightly.

'Oooh,' says the fish. 'No one ever changes the plant life in HERE. It's the same seaw--'

'Cut me a break and cork it,' you reply tartly."

The player is carrying a telegram, a bouquet, and a lingerie bag. The player is wearing a chef hat.

The description of the telegram is "A telegram, apparently. And dated three days ago. [fixed letter spacing]TRIUMPH OURS STOP BACK SOON STOP BE SURE

```
TO FEED FISH STOP[variable letter spacing]". [For printing options see 4.13.]
Understand "yellow paper" as the telegram.

After examining the telegram for the first time:
    say "'So,' blubs the evil fish. 'How about it? Little food over here?'"

After examining the telegram:
    choose a random row in the Table of Insulting Fish Comments;
    say "[comment entry][paragraph break]".

Table of Insulting Fish Comments
comment
"'Yeah, yeah,' says the fish. 'You having some trouble with the message, there?
Confused? Something I could clear up for you?'"
"'Oookay, genius kid has some troubles in the reading comprehension
department.' The fish taps his head meaningfully against the side of the tank.
'I'm so hungry I could eat my way out, you get my meaning?'"
"'I'll translate for you,' screams the fish in toothy fury. 'It says GIVE FOOD TO
FISH!! How much more HELP do you NEED???"

The description of the chef hat is "A big white chef hat of the kind worn by chefs.
In this case, you. Just goes to show what a hurry you were in on the way out of
the restaurant." Understand "big" or "white" or "chefs" or "chef's" as the chef hat.
[Inform knows that this is clothing because the player starts out wearing it, so
there's no need to say so separately.]

The aquarium is a transparent open container in the Studio. It is not openable.
"In one corner of the room, a large aquarium bubbles in menacing fashion." The
description of the aquarium is "A very roomy aquarium, large enough to hold
quite a variety of colorful sealife -- if any yet survived." Understand "tank" as the
aquarium.

The aquarium contains some gravel and some seaweed. Understand "little
rocks" as the gravel. Understand "weed" as the seaweed. The description of the
gravel is "A lot of very small grey rocks." The description of the seaweed is
"Fake plastic seaweed of the kind generally bought in stores for exactly this
purpose."

The examine containers rule does nothing when examining the aquarium.

After examining the gravel for the first time:
    say "The fish notices your gaze; makes a pathetic mime of trying to find little
flakes of remaining food amongst the gravel."

After examining the seaweed for the first time:
    say "'Nice, hunh?' blubs the fish, taking a stabbing bite out of one just by way
of demonstration. 'Look so good I could eat it.'"

The aquarium contains an animal called an evil fish. The description of the fish is
"Even if you had had no prior experience with him, you would be able to see at a
glance that this is an evil fish. From his sharkish nose to his razor fins, every
inch of his compact body exudes hatred and danger."

Instead of taking the evil fish:
    say "The fish swims adroitly out of range of your bare hand. 'Hey,' he says,
and the bubbles of his breath brush against your fingers. 'Count yourself lucky I don't bite you right now, you stinking mammal.'"

Instead of attacking the evil fish:
    say "Oh, it's tempting. But it would get you in a world of hurt later on."

Instead of kissing the evil fish:
    say "You're saving all your lovin for someone a lot cuddlier."

After examining the evil fish for the first time:
    say "The fish glares at you, as though to underline this point."

After examining the evil fish for the second time:
    say "'If you're looking for signs of malnutrition,' says the fish, 'LOOK NO FURTHER!!' And it sucks in its gills until you can see its ribcage."

An every turn rule:
    choose a random row in the Table of Fish Banter;
    say "[comment entry][paragraph break]".

Table of Fish Banter
```

comment	used
"'Hey, nice SKIN TONE,' shouts the evil fish. His words reach you in a spitting gurgle of aquarium water. 'You gone over to a pure eggplant diet these days?'"	0
"The evil fish is floating belly up! ...oh, curse. He was toying with you. As soon as he sees you looking, he goes back to swimming around."	0
"The evil fish darts to the bottom of the tank and moves the gravel around with his nose."	0
"The evil fish is swimming around the tank in lazy circles."	0
"The evil fish begins to butt his pointy nose against the glass walls of the tank."	0

```
The description of the bouquet is "Okay, so it's silly and sentimental and no doubt a waste of money, of which there is never really enough, but: you miss her. You've missed her since ten seconds after she stepped aboard the shuttle to Luna Prime, and when you saw these -- her favorites, pure golden tulips like springtime -- you had to have them." Understand "flowers" or "tulip" or "tulips" as the bouquet.

After examining the bouquet for the first time:
    say "'Oh, you shouldn't have,' says the fish. 'For me??'
    You just respond with a livid glare."

    Instead of smelling the bouquet for the first time:
        say "'Mmm-mm,' says the fish. 'Damn, I sure wish I had olfactory abilities. Hey, if I did, I might be even better at noticing the presence or absence of FOOD.'"

    The description of the lingerie bag is "You grant yourself the satisfaction of a little peek inside. You went with a pale, silky ivory this time -- it has that kind of sophisticated innocence, and it goes well with the purple of your skin. A small smirk of anticipation crosses your lips."

    After examining the lingerie bag for the first time:
        say "'What's in THERE?' asks the fish. 'Didja bring me take-out? I don't mind Chinese. They eat a lot of carp, but what do I care? I'm not a carp. Live and let live is what I s--'
```

```
'It's NOT take-out.' You stare the fish down and for once he actually backstrokes a stroke or two. 'It's PRIVATE.'"

After examining the lingerie bag for the second time:
    say "'If it's not take-out, I don't see the relevance!' shouts the fish. 'Food is what you want in this situation. Food for MEEEE.'"

Understand the command "feed" as something new.

Understand "feed [something]" as feeding.

Feeding is an action applying to one visible thing.

Check feeding:
    if the noun is not the evil fish, say "That doesn't make much sense." instead;
    if the player is not carrying the fish food, say "You need the fish food first." instead.

Carry out feeding:
    increment the score;
    say "Triumphantly, you dump the remaining contents of the canister of fish food into the tank. It floats on the surface like scum, but the fish for once stops jawing and starts eating. Like a normal fish. Blub, blub.[paragraph break]";
    say "[bold type] *** TWO HOURS LATER ***[roman type][paragraph break]'So,' Britney says, tucking a strand of hair behind your ear, 'where shall we go for dinner? Since I made the big bucks on this trip, it's my treat. Anywhere you like.'[paragraph break]'I've had a hankering all day,' you admit, as the two of you turn from the shuttle platform and head toward the bank of taxis. 'I could really go for some sashimi right now.'";
    end the story finally.

Before feeding the fish food:
    try feeding the evil fish instead.

When play begins:
    say "You're on the run. You've got a million errands to do -- your apartment to get cleaned up, the fish to feed, lingerie to buy, Britney's shuttle to meet--[paragraph break]The fish. You almost forgot. And it's in the studio, halfway across town from anywhere else you have to do. Oh well, you'll just zip over, take care of it, and hop back on the El. This'll be over in no time.[paragraph break]Don't you just hate days where you wake up the wrong color?[paragraph break]".

The maximum score is 1.

Test me with "x fish / g / kiss fish / x aquarium / x gravel / x seaweed / i / x telegram / x bouquet / smell bouquet / x lingerie / g / x hat / x window / open window / x painting / g / x cabinet / open cabinet / x cloths / search cloths / open food / feed fish".
```

Chapter 8: Change

Contents of *Writing with Inform*
Chapter 7: Basic Actions
Chapter 9: Time
Indexes of the examples

§8.1. Change of values that vary

So far, what we have done in response to the player's commands amounts to little more than a few ripostes. The simulated world does change during play, as the player moves from room to room or picks up things, but all of this is happening automatically, not at our direct instruction. How then can we make the world change?

Recall that the world consists of rooms, in which are things, and that all of these have properties appropriate to their kinds. Some properties are either/or ("open" or "closed" but not both and not neither), while others have values (the "matching key" of a lockable door, for instance). Finally, we may also have created some free-standing values or "variables".

We take the last example first, as it is the simplest. Suppose we have:

```
"Winds of Change"

The prevailing wind is a direction that varies. The prevailing wind is northwest.

The Blasted Heath is a room. "Merely an arena for the play of witches and kings, my
dear, where the [prevailing wind] wind blows."

Instead of waiting when the prevailing wind is northwest:
   say "A fresh gust of wind bowls you over.";
   now the prevailing wind is east.
```

The new phrase here is "now". This automatically checks that the new value is one which makes sense in the given context, so for instance it would not allow either of these:

```
now the prevailing wind is 25;
now the prevailing wind is the Heath;
```

the former being a number, and the latter a room, so that neither is a direction. Similarly, "now" will not allow constant values to be changed. So

```
Colour is a kind of value. The colours are blue, red and mauve.

After pulling the psychedelic lever:
    now blue is mauve.
```

...will result in a problem message; it's like writing "now 1 is 2". The difference between "the prevailing wind" and "blue" is that the wind was declared to be a "direction that varies", whereas blue wasn't.

⬆ Start of Chapter 8: Change

⬅ Back to Chapter 7: Basic Actions: §7.18. Postscript on actions

➡ Onward to §8.2. Changing the command prompt

§8.2. Changing the command prompt

The command prompt is the text printed by Inform to ask the player for another command. Ordinarily this is simply a greater-than-sign, ">", so we tend not to notice it as text at all. Internally, though, it is a variable value called "command prompt", which means we can change it.

For example, this will be a more conversational sort of prompt:

```
When play begins: now the command prompt is "What now? ".
```

Whereas this will be more up-to-the-minute and demanding:

```
When play begins: now the command prompt is "[time of day] >".
```

("Time of day" is another variable value, which is fairly self-explanatory, but will be covered in detail later on.) The prompt can be changed at any point, so can be used to indicate the current situation, or even as a sly way to introduce a sort of conversation between computer and player.

⬆ Start of Chapter 8: Change

⬅ Back to §8.1. Change of values that vary

➡ Onward to §8.3. Changing the status line

⬇ Example 113: ★★★ **Don Pedro's Revenge** Combat scenario in which the player's footing and position changes from move to move, and the command prompt also changes to reflect that.

§8.3. Changing the status line

The status line is the black bar along the top of a story being played, which ordinarily displays the current position; in a story with scoring, it also usually shows the score and number of moves taken. Like the command prompt, it is not fixed but results from values which can be changed: the "left hand status line" and "right hand status line".

The default values are "[the player's surroundings]" for the left hand status line and "[score]/[turn count]" for the right hand status line (if there's scoring; it's blank otherwise). Score and turn count are numbers which vary in play (more about scoring later); "[the player's surroundings]" is a text substitution really intended for just this purpose:

say "[the/-- player's surroundings]"

This text substitution produces a succinct description of where the player is, be this in darkness, in a lighted room or inside an opaque container such as a large packing case. Example:

```
now the left hand status line is "You: [the player's surroundings]";
```

These make useful elements to juggle in redesigning the status line, as in the following example:

```
When play begins:
    now the left hand status line is
        "[the player's surroundings] / [turn count] / [score]";
    now the right hand status line is "Time: [time of day]".
```

The text in the right hand status line should be kept no more than 14 letters long, including any spaces. The left hand status line has more leeway, but should still be kept brief.

★ *See* **Awarding points** *for scoring*

⬆ Start of Chapter 8: Change

⬅ Back to §8.2. Changing the command prompt

➡ Onward to §8.4. Change of either/or properties

⬇ Example 114: ★ **Politics as Usual** Have the status line indicate the current region of the map.

⬇ Example 115: ★★★ **Centered** Replacing the two-part status line with one that centers only the room name at the top of the screen.

§8.4. Change of either/or properties

When we have an either/or property, we can set it like so:

```
Instead of waiting when the oaken door is closed:
    say "There is a slow, creaky click! sort of noise as the door swings open, apparently
```

```
all by itself.";
    now the oaken door is open.
```

If it is open already, nothing changes: in any case nothing is said to the player unless we give explicit instructions to that effect, as we've done here.

Inform protects its model world from accidental damage in several ways, one of which is to ensure that things are not given properties which they are not allowed to have. So this, for instance, will not be accepted:

```
now the oaken door is unvisited
```

More subtle problems arise if it is not possible to tell, when the story is being constructed, what the object in question will be: for instance, if we try to change a randomly chosen object to be "unvisited". Inform therefore makes additional checks during play, printing up a suitable message only if the rules are violated. The net effect is that it is impossible for the oaken door ever to have the "unvisited" property.

⬆ Start of Chapter 8: Change

⬅ Back to §8.3. Changing the status line

➡ Onward to §8.5. Change of properties with values

⬇ Example 116: ★ **Vitrine** An electrochromic window that becomes transparent or opaque depending on whether it is currently turned on.

§8.5. Change of properties with values

Changing properties with values is very similar:

```
now the printed name of the Closet is "Suddenly Spooky Closet"
```

Inform checks three different things to ensure that this change is safe to perform. Firstly, the value must be the right kind for the property in question, so this for instance would be rejected:

```
now the printed name of the Closet is 7
```

Secondly, the object in question has to be allowed to have the given property. This, for instance, would be disallowed:

```
now the initial appearance of the Closet is "Dusty"
```

(since "initial appearance" is a property which only things can have, not rooms). Finally, the object has to actually have the property, not just have the right to have that property. Thus:

```
now the printed name of the Closet is "Suddenly Spooky Closet"
```

...is only permitted if the Closet is designed with a "printed name". In fact this is certain to be true: all rooms and things automatically have a printed name, which is the short boldface description in the case of rooms, and the usual text briefly describing something in the case of things.

"Now" is a simple way to change many things in Inform, but it's cumbersome to change the map of the model world using "now", because the map is such a complicated arrangement. (It's not a property: it's a sort of mesh of relations.) So a special phrase exists to change map connections:

change (direction) **exit of** (room) **to** (room)

This phrase alters the map so that the given map connection is made. Note that connections can be made to rooms, but not doors: the positions of doors are fixed. Example:

```
change the east exit of the Closet to the Tsar's Imperial Dining Salon
```

Since "nothing" is not a room, this doesn't allow us to change the exit to nothing, so there is a separate definition of:

```
change the west exit of the Closet to nothing
```

change (direction) **exit of** (room) **to nothing**/nowhere

This phrase alters the map so that the given map connection is unmade. Example:

```
change the west exit of the Closet to nowhere
```

Altering the map itself is not a very subtle way to adjust when and where the player can move - writing suitable rules is usually a cleaner solution - so this phrase is best avoided unless really needed.

Start of Chapter 8: Change

Back to §8.4. Change of either/or properties

Onward to §8.6. Whose property?

Example 117: ★ **Thirst** A waterskin that is depleted as the player drinks from it.

Example 118: ★ **Thirst 2** A campfire added to the camp site, which can be lit using tinder.

§8.6. Whose property?

This seems a useful point to clarify something already seen. We normally call a property with a value something like:

```
the printed name of the West Ballroom
```

We are sometimes allowed to omit the "of the ..." part, and simply call it "the printed name", for the sake of brevity. For instance, the following room description:

```
The West Ballroom is a room. "A handsome sweep of chequered floor beckons the eye
into the [printed name]."
```

will result in "West Ballroom" being substituted for "[printed name]". Since the text belongs to the West Ballroom, that is assumed to be the owner of any properties named in its description. Similarly:

```
Instead of examining something, say "Hmm, let me see: [printed name]..."
```

Here the owner of the "printed name" is assumed to be the noun referred to in the action - in other words, the "something" alluded to in the rule.

⬆ Start of Chapter 8: Change

⬅ Back to §8.5. Change of properties with values

➡ Onward to §8.7. Moving things

§8.7. Moving things

We have now seen how to change the properties of rooms and things, and also any freestanding values which may have a bearing on the model world. We are not allowed to change the kind of anything during play. Our remaining freedom is to move things around. It would make no sense to move rooms around, because rooms are the fixed reference points in our geography, but anything else is mobile. This even includes things which are supposedly "fixed in place", for unlike the player, we have god-like powers. (There are minor restrictions: backdrops are trickier to move, since they are present in several rooms at once - see the next section. And doors, at the junction between two rooms, cannot be moved.)

Here is how to move something:

move (object) **to** (object)

This phrase moves the first-named object to the second. Example:

```
move the genie's lamp to Aladdin's Cave;
```

The first object named has to be a thing; the destination must be a room, as here, a container, a supporter, or a person. When something is moved, all its parts and contents (and all their contents, and so on) move with it. If the thing being moved is a person, then the destination is required to be a room or an enterable container. (In particular, a person cannot be carried by another person.)

Two options can be used if the object being moved is the player.

```
move the player to Aladdin's Cave, without printing a room description
```

omits the description which would otherwise be produced. A compromise is to use:

```
move the player to Aladdin's Cave, printing an abbreviated room description
```

which gives a full description if the player has never been here before, but only a brief one if it is a familiar scene. These options have no effect for any other objects being moved.

If the destination is a person, like so:

```
move the genie's turban to Aladdin;
```

then it will be carried rather than worn. We could arrange for it to be worn instead by writing

```
now the genie's turban is worn by Aladdin;
```

"Now..." is a much more flexible phrase than "move": more on this shortly.

⬆ Start of Chapter 8: Change

⬅ Back to §8.6. Whose property?

➡ Onward to §8.8. Moving backdrops

⬇ Example 119: ★★ **Meteoric I and II** A meteor in the night sky which is visible from many rooms, so needs to be a backdrop, but which does not appear until 11:31 PM.

§8.8. Moving backdrops

A backdrop can be in several rooms at once. When created, its position can be given as any specific collection of rooms, or as a region, or even as "everywhere". For instance:

```
The Upper Cave is above the Rock Pool. The Ledge is east of the Pool.

The stream is a backdrop. It is in the Upper Cave and the Ledge.
```

Moving backdrops is not like moving other things, because there's no single destination. There are several possibilities:

(a) A backdrop can be moved to a region. If we define:

```
Lower Level is a region. The Rock Pool and the Ledge are in the Lower Level.
```

then we can write either of

```
move the stream to the Lower Level;
now the stream is in the Lower Level;
```

and either way, the stream is now found in the Rock Pool and the Ledge but nowhere else.

(b) A backdrop can be moved to a category of rooms:

move (object) **backdrop to all** (description of objects)

This phrase moves the backdrop so that it is now present in every room matching the given description. Example: If we define

```
A room can be wet or dry. A room is usually dry. The Rock Pool is wet.
```

then we can write

```
move the stream backdrop to all wet rooms;
```

This phrasing, "move the ... backdrop to all ..." is deliberately meant to look unlike the simpler "move ... to ...", to emphasise that this kind of movement is possible only for backdrops.

What then happens is that the stream is present in whichever rooms are currently wet. But the stream's position is ordinarily checked only after movements, for efficiency's sake. So if the player is in a room which suddenly changes from being dry to being wet, the stream will not magically appear (though it will be there if the player goes out and comes in again). If this is not good enough, the phrase "update backdrop positions" can be used to ensure the accuracy of all backdrop locations after a dramatic change:

update backdrop positions

This phrase runs through all backdrops in the model world and makes sure they are correctly in, or not in, the current location, so that everything appears right from the player's point of view. Example:

```
The Upper Cave is above the Rock Pool. The Ledge is east of the Pool. The
stream is a backdrop.

When play begins:
    move the stream backdrop to all wet rooms.

A lever is in the Cave. The lever is fixed in place.

Instead of pulling the lever when the Cave is dry:
    now the Cave is wet;
    now the lever is in the Rock Pool;
    now the lever is portable;
```

```
        update backdrop positions;
        say "The old rusty lever pulls away, and the thin cave wall goes with it, so
    that a stream bursts into the cave, falling to the pool below."
```

(c) A backdrop can be moved to be either everywhere or nowhere:

```
After sleeping:
    say "It's a bright new day!";
    now the stars are nowhere.

After waiting:
    say "Darkness falls rapidly here.";
    now the stars are everywhere.
```

Start of Chapter 8: Change

Back to §8.7. Moving things

Onward to §8.9. Moving the player

Example 120: ★★★ **Orange Cones** Creating a traffic backdrop that appears in all road rooms except the one in which the player has laid down orange cones.

§8.9. Moving the player

The player is a thing, too, and can also be moved, which has the effect of instantaneous transportation, without the need for a suitable map connection to the new location. For instance, these are equivalent:

```
move the player to the Bodleian Library;
now the player is in the Bodleian Library;
```

This will ordinarily result in a room description of the Bodleian Library being printed up, but that might not always be desirable. For instance:

```
Instead of waiting in the Schola Maleficorum:
    say "A bored demon catches your eye (they really do have very inquisitive fingers)
and throws you back out into the Antechamber.";
    move the player to the Antechamber, without printing a room description.
```

Thus tacking on the option "without printing a room description", remembering to add the comma, omits the description which would otherwise be produced. A compromise is to use the option "printing an abbreviated room description": this gives a full description if the player has never been here before, but only a brief one if it is a familiar scene.

The player's point of view can also be moved by shifting to another character. Suppose the story features two people, Alice and Bob, and the player at the keyboard is giving commands to Alice, and seeing everything from her point of view. The phrase:

```
now the player is Bob
```

switches the perspective so that now Bob is the one controlled by the human player, and it's Bob's point of view which counts. The human being at the keyboard may feel a sense of having jumped abruptly from place to place, but in fact neither Alice nor Bob has moved.

A change of player can sometimes cause confusing things to happen, if it takes place as part of a successful action. Suppose there's an action called "possessing", which enables the player to possess somebody else's body; and suppose the player types POSSESS ADELE. The action succeeds, so that the player moves into the mind of Adele. But that means that at the end of the action, the player is no longer the actor - that is, no longer the person who began the action; and consequently, Inform won't use the report rulebook to say what has just happened. It's a strange business, moving into another body.

Start of Chapter 8: Change

Back to §8.8. Moving backdrops

Onward to §8.10. Removing things from play

Example 121: ★★★ **Terror of the Sierra Madre** Multiple player characters who take turns controlling the action.

§8.10. Removing things from play

Some things will occasionally be in a limbo state called being "off-stage": like actors or props not needed in Act II, but perhaps to be brought back on-stage later, they wait on the sidelines. Anything created with no apparent location will start the story off-stage, as in the case of the lamp here:

```
Aladdin's Cave is a room. The genie's lamp is a container.
```

(Such things are easy to see in the World index because they are listed after all of the rooms and their contents, not belonging inside any room.) If we wanted to make this clearer to a human reader, we could add:

```
The lamp is nowhere.
```

to emphasise the point. In this context, "nowhere" means "in no room". Moving the lamp onto the stage-set, so to speak, is easy:

```
now the lamp is in the Cave;
```

or perhaps:

```
now the player is carrying the lamp;
```

and we can whisk it away again like so:

```
now the lamp is nowhere;
```

(We can't say "now the lamp is somewhere" because that's too vague about exactly where it is.) In older builds of Inform, the usual thing was to write "remove the lamp from play", but that's now a deprecated phrase: better to use "nowhere" instead.

remove (object) **from play**

Removes the given object from play, so that it is not present in any room. We are not permitted to remove rooms, or doors, or the player, from play; but we are permitted to remove backdrops, making them disappear from all rooms in which they are present. Example:

```
remove the gold coin from play;
```

We can test whether something is on-stage or off-stage with:

```
if the gold coin is somewhere, ...
if the gold coin is nowhere, ...
```

Inform also understands two adjectives for this:

```
if the gold coin is on-stage, ...
if the gold coin is off-stage, ...
```

Because these are adjectives, they can be used in a few ways which "nowhere" and "somewhere" can't, such as:

```
say "Ah, so many absent friends. Who now remembers [list of off-stage people]?"
```

Note that "on-stage" and "off-stage" apply only to things. Rooms, directions and regions are the stage itself: so it makes no sense to ask the question of whether they are "on-" or "off-". Doors are always on-stage; a backdrop, say "the sky", is always on-stage unless it has been taken off by writing something like "now the sky is nowhere".

⬆ Start of Chapter 8: Change

⬅ Back to §8.9. Moving the player

➡ Onward to §8.11. Now...

⬇ Example 122: ★ **Beverage Service** A potion that the player can drink.

⬇ Example 123: ★ **Spring Cleaning** A character who sulks over objects that the player has broken (and which are now off-stage).

⬇ Example 124: ★★ **Extra Supplies** A supply of red pens from which the player can take another pen only if he doesn't already have one somewhere in the game world.

§8.11. Now...

"Now" has already appeared several times in this chapter, being used like a Swiss army knife to change values of all kinds:

```
now the score is 100;
```

In fact, "now" is by far the most flexible phrase known to Inform.

> **now** (a condition)
>
> This phrase makes the condition become true. Examples:
>
> ```
> now the score is 100;
> now the player is Kevin;
> now the front door is open;
> now Mr Darcy is wearing the top hat;
> now all the doors are open;
> now all of the things in the sack are in the box;
> ```
>
> Inform issues a problem message if the condition asks to do the impossible ("now 3 is an even number") or is vague ("now the duck is not in the Lily Pond") or not in the present tense ("now the front door had been open").

We've now seen all three things which can be done with a condition S which describes the world:

```
S. - The relation holds at the start of play.
if S, ...; - Does the relation hold right now?
now S; - Make the relation hold from now on.
```

For instance,

```
The apple is in the basket.
if the apple is in the basket, ...;
now the apple is in the basket;
```

⬆ Start of Chapter 8: Change

⬅ Back to §8.10. Removing things from play

➡ Onward to §8.12. Increasing and decreasing

⬇ Example 125: ★ **Bee Chambers** A maze with directions between rooms randomized at the start of play.

⬇ Example 126: ★★ **Hatless** It's tempting to use "now..." to distribute items randomly at the start of play, but we need to be a little cautious about how we do that.

⬇ Example 127: ★★★ **Technological Terror** A ray gun which destroys objects, leaving their component parts behind.

§8.12. Increasing and decreasing

Once we begin to deal with named values (or table entries, list entries or other ways to describe places where values are kept), we find that we often want to change them. We could if we wanted always use "now" for this, but it can be a little clumsily worded if we want to increase or decrease something:

```
now the score is the score plus six;
```

Because of that, we have some convenient abbreviations which have the advantage that the value being changed only has to be named once:

increase (a stored value) **by** (value)

This phrase increases the variable, table entry, list entry, or property by the given amount, which must be of a compatible kind. Example:

```
increase the score by 8;
increase the time of day by 5 minutes;
```

decrease (a stored value) **by** (value)

This phrase decreases the variable, table entry, list entry, or property by the given amount, which must be of a compatible kind. Example:

```
decrease the score by 6;
decrease the carrying capacity of the player by 10;
```

An even greater abbreviation can be made when we are changing a number by 1 either way:

increment (a stored value)

This phrase increases the variable, table entry, list entry, or property by 1. Example:

```
increment the score;
```

decrement (a stored value)

This phrase decreases the variable, table entry, list entry, or property by 1. Example:

```
decrement the score;
```

"Increment" and "decrement" are traditional computing terms, though they have been used in engineering for at least a century and in finance for longer still.

Start of Chapter 8: Change
Back to §8.11. Now...
Onward to §8.13. Checking on whereabouts

§8.13. Checking on whereabouts

We have seen that while rooms are fixed, their contents move around, so we will need ways to examine the current whereabouts of things. The following examples show the kind of conditions allowed:

```
if the genie's lamp is in Aladdin's Cave ...
if Aladdin is not in Aladdin's Cave ...
if Aladdin's Cave contains the genie's lamp ...
if the genie's lamp is carried by Aladdin ...
if Aladdin is carrying the genie's lamp ...
if Aladdin does not have the genie's lamp ...
if the table supports the genie's lamp ...
if the table is supporting the genie's lamp ...
if the genie's lamp is supported by the table ...
if the genie's lamp is on the table ...
if the genie's lamp is on top of the table ...
if the genie's lamp is in the cupboard ...
if the genie's lamp is contained in the cupboard ...
if the genie's lamp is inside the cupboard ...
if the genie's lamp is within the cupboard ...
if the wick is part of the genie's lamp ...
```

These are exactly like the assertions which we use to set up the world, except that we make them questions by placing "if" in front. But we shall later see that we can also use three other tenses, not to mention plural forms, so that new verbal forms like "had not been inside" and "were not supported by" are legal here (which they would not be in assertions). What we are not allowed is to contract these verbs with apostrophes: "isn't", "hasn't" and "hadn't" are forbidden.

Overwhelmingly the condition we check most is whether the player is carrying something. The following are therefore equivalent:

```
if the genie's lamp is carried by the player ...
if the genie's lamp is carried ...
```

And similarly for "not carried", "worn" and "not worn". To be precise, if a form of *to be carried* or *to be worn* is not followed by any other description, then "the player" is assumed to be doing the carrying or wearing.

Start of Chapter 8: Change
Back to §8.12. Increasing and decreasing
Onward to §8.14. More flexible descriptions of whereabouts

§8.14. More flexible descriptions of whereabouts

The examples just given were all basically of the form "X *relation* Y" where X and Y were specific names of things. For example,

```
if the genie's lamp is carried by Cinderella ...
if the genie's lamp is inside the cupboard ...
```

Just as actions could be described with patterns to be matched ("taking an open container", say), so can the positions of things. Giving subtler descriptions of our X and Y sometimes broadens the possibilities, sometimes narrows them:

```
if the genie's lamp is carried by a woman ...
if the genie's lamp is inside the closed cupboard ...
```

In the first case, Y is allowed to be one of a whole range of things - any of the women existing in the world. This makes for a broader condition. In the second case, Y has not only to be the cupboard, but at a time when it is closed: which makes for a narrower condition. We can, of course, also vary X:

```
if an animal is inside the cupboard ...
if a container is carried ...
```

And we can even vary both X and Y at once:

```
if a woman is holding an animal ...
```

a condition which will be true if, anywhere in the story's world, any woman is holding any animal.

Start of Chapter 8: Change
Back to §8.13. Checking on whereabouts
Onward to §8.15. Calling names

§8.15. Calling names

Conditions like "if somebody is in an adjacent room" allow complicated tests to be performed with a minimum of fuss, but it's rare that we want to know only whether they are true: more likely we also want to know *which* person, and *which* room.

For this purpose, we are allowed to supply a name for any such vaguely-described object which comes up, and then to use that name thereafter.

```
if somebody is in an adjacent room (called the Hiding Place), say "You hear distant
breathing from [the Hiding Place]."
```

We can even name more than one of the things discovered:

```
Instead of waiting when a woman (called the kidnapper) is holding an animal (called the
pet), say "How can you think of rest when, somewhere out there, [pet] has been cruelly
kidnapped by [the kidnapper]?"
```

Note the brackets, which are essential. The result of typing "wait" is then

```
How can you think of rest when, somewhere out there, a lapdog has been cruelly
kidnapped by Baroness Orczy?
```

Of course, that might be just one of many animals held by women in the story. We shall later see ways to go through all of the possibilities found, performing some action with each in turn.

A calling, if we can use that word, should be made immediately after the noun it refers to, and not left to hang back after any relative clauses. For instance,

```
if something (called the penitential object) held by the player is hot
```

is allowed, but not

```
if something held by the player (called the penitential object) is hot
```

because there is too much potential ambiguity - are we trying to call the player something?

★ *See* **Repeat running through** *for systematically working on everything matching a description*

⬆ Start of Chapter 8: Change

⬅ Back to §8.14. More flexible descriptions of whereabouts

➡ Onward to §8.16. Counting the number of things

⬇ Example 128: ★ **Higher Calling** All doors in the game automatically attempt to open if the player approaches them when they are closed.

---

§8.16. Counting the number of things

It is very often useful to know how many things are in a given situation, and for this purpose we have the "number of ..." construction. For instance:

```
the number of edible things carried
the number of things on the table
the number of people in the Dining Room
```

Whereas "a woman is holding an animal" makes the same test as "an animal is held by a woman", getting the same result, counting is not so even-handed:

```
the number of women holding animals
the number of animals held by women
```

are different questions and, unless the ration is strictly one lapdog per baroness, will have different answers. If Cruella de Vil has 101 dalmatians, they may be very different indeed.

It can also be helpful to count things with no particular location, like so:

```
the number of rooms
the number of closed doors
```

For instance:

```
When play begins, change the right hand status line to "Explored: [number of visited
rooms]/[number of rooms]".
```

Provided that the possible range is finite, we can also use "number of" to count values which match a description. For instance:

```
the number of non-recurring scenes
```

or if we were to define

```
Colour is a kind of value. The colours are red, orange, yellow, green, blue, indigo and
violet.
```

then "the number of colours" would evaluate to 7. As with other ways of talking about whole ranges of values, this only works if the range is manageable. "The number of numbers" cannot sensibly be worked out: there are infinitely many, for all practical purposes, and similarly for "the number of texts".

⬆ Start of Chapter 8: Change
⬅ Back to §8.15. Calling names
➡ Onward to §8.17. Looking at containment by hand

§8.17. Looking at containment by hand

The descriptions outlined in the last few sections are intended to deal with almost all of the routine questions we might have about what currently resides where. It should be a last resort to use the following more primitive way to inspect the world.

holder of (object) **... object**

This phrase produces the container, supporter, carrier, wearer or room in which the object resides.

It's sometimes useful to go the other way. When something has possessions, we can find them out one at a time by running through a list.

first thing held by (object) **... object**

This phrase produces the first of the list of things held by the object. Example:

```
first thing held by Baroness Orczy
```

next thing held after (object) **... object**

This phrase produces the next item of the list of things held by something. Example: suppose Baroness Orczy is carrying a lapdog and a string of pearls.

```
next thing held after the lapdog
```

is then the string of pearls.

Start of Chapter 8: Change
Back to §8.16. Counting the number of things
Onward to §8.18. Randomness

§8.18. Randomness

Sometimes we want to introduce random behaviour into play. We usually do this by generating random values, and then acting differently depending on what they are. The following:

```
a random number from 2 to 5
```

produces, as it suggests, a random number drawn from the choices 2, 3, 4 or 5, each of which is equally likely to come up. In fact, this isn't limited to numbers:

a random (name of kind) **between** (arithmetic value) **and** (arithmetic value) **... value**

or:

a random (name of kind) **from** (arithmetic value) **to** (arithmetic value) **... value**

or:
a random (name of kind) **between** (enumerated value) **and** (enumerated value) **... value**

or:
a random (name of kind) **from** (enumerated value) **to** (enumerated value) **... value**

This phrase produces a uniformly random value in the range given. Examples:

```
a random number from 10 to 99
a random time from 2:31 PM to 2:57 PM
```

If we make a new kind of value:

```
A cloud pattern is a kind of value. The cloud patterns are cumulus,
altocumulus, cumulonimbus, stratus, cirrus, nimbus, nimbostratus.
```

then we can also take random values from it:

```
a random cloud pattern between stratus and nimbus
```

which has three possible outcomes, all equally likely.

We can also use random conditions:

if a random chance of (number) **in** (number) **succeeds:**

This condition is true X/Yths of the time, where X and Y are the numbers. Example:

```
if a random chance of 2 in 3 succeeds, ...
```

Here is a rule which applies only 15% of the time:

```
Instead of waiting when a random chance of 15 in 100 succeeds: ...
```

Testing IF which makes random choices can be rather frustrating, because a problem showing up on one attempt may not show up on another. We can get around this by making use of the fact that computers do not actually generate true randomness, but instead make a sequence of apparently random numbers by applying a complicated formula to each one in order to make the next. The starting point is a number called the "seed", because the next choice grows out of it.

seed the random-number generator with (number)

This phrase changes the seed number as specified. Any random numbers generated after that depend only on the seed. Example: the following sentence will "fix" the process of generating these random numbers so that they are not random at all - the same sequence of random numbers will be produced on each run.

```
When play begins, seed the random-number generator with 1234.
```

The seed value "1234" can be anything positive; a different sequence of random numbers will be produced for each different seed value. A seed value of 0 restores the RNG to properly random behaviour again.

Alternatively, it's possible the "fix" the RNG by clicking the "Make random outcomes predictable when testing" option on the Settings panel. This makes the behaviour predictable whenever the story is played within Inform, but (unlike the rule above) has no effect on the story file once released.

Start of Chapter 8: Change

Back to §8.17. Looking at containment by hand

Onward to §8.19. Random choices of things

Example 129: ★ **Do Pass Go** A pair of dice which can be rolled, and are described with their current total when not carried, and have individual scores when examined.

Example 130: ★ **Lanista 1** Very simple randomized combat in which characters hit one another for a randomized amount of damage.

Example 131: ★ **Weathering** The automatic weather station atop Mt. Pisgah shows randomly fluctuating temperature, pressure and cloud cover.

Example 132: ★★★ **Uptown Girls** A stream of random pedestrians who go by the player.

§8.19. Random choices of things

Writing "a random number" is not allowed, because the possible range is too large, but that was the only reason why not.

a/-- random (description of values) **... value**

This phrase makes a uniformly random choice from values satisfying the description given. Example:

```
a random visited room
a random scene
```

A problem message is issued if the range is too large (for instance, "a random text"). Unexpected results may follow if no value fits the description, unless we are describing objects, in which case the result is the special value "nothing".

For instance:

```
say "You can see [number of adjacent rooms] way[s] from here; how about [random adjacent room]?"
```

But it's important to worry about the possibility that nothing qualifies - here, that no adjacent rooms exist. The above would then say:

```
You can see 0 ways from here; how about nothing?
```

Start of Chapter 8: Change

Back to §8.18. Randomness

Onward to Chapter 9: Time: §9.1. When play begins

Example 133: ★ **Candy** One of several identical candies chosen at the start of play to be poisonous.

Example 134: ★ **Zork II** A "Carousel Room", as in Zork II, where moving in any direction from the room leads (at random) to one of the eight rooms nearby.

Examples from Chapter 8: Change

Start of this chapter

Chapter 9: Time

Indexes of the examples

113 ★★★ **Example Don Pedro's Revenge** RB

Combat scenario in which the player's footing and position changes from move to move, and the command prompt also changes to reflect that.

Suppose our game features a detailed simulated combat between the player character and his opponent. He might have several weapons available, and several types of attack available; and at any given time he might be perched up in the rigging of his ship, standing on the open deck, or boxed in between some barrels. His options will vary depending on his position, and obviously it would detract from the pacing to make the player keep LOOKing in the middle of combat in order to remind himself where he is. Instead, we'll roll this information into the command prompt:

```
"Don Pedro's Revenge"

The Deck of the Helene Marie is a room. "The two crews are embattled all around you, but your attention is reserved for your particular enemy: Don Pedro."
```

```
Table of Random Prompts

    position  prompt
    boxed     "So securely boxed-in that you can really only parry or thrust, you try to "
    boxed     "Trapped between your barrels, you decide to "
    perched   "Able to slice at your attackers but not to advance or retreat, you choose to "
    perched   "Perched up here with the advantage of height (but little mobility), you attempt to "
    free      "Out on the open deck with no impediments, free to advance or retreat, you decide to "

When play begins: reset the prompt.

Every turn: reset the prompt.

To reset the prompt:
    sort the Table of Random Prompts in random order;
    repeat through the Table of Random Prompts:
        if the position entry is the placement of the player:
            now the command prompt is prompt entry;
            stop.

After reading a command: say conditional paragraph break.

A placement is a kind of value. The placements are boxed, perched, free. The player has a placement. The player is free.

Understand "retreat" or "parry" as retreating. Retreating is an action applying to nothing.

Check retreating:
    if the player is perched, say "You can't move backward or parry very successfully from this position." instead.

Carry out retreating:
    now the player is boxed;
    say "You protect yourself, but end up wedged in between two barrels."

Understand "thrust" or "advance" as advancing. Advancing is an action applying to nothing.

Check advancing:
    if the player is perched, say "You can't move forward from here, only slash." instead.

Carry out advancing:
    now the player is free;
    say "You push forward aggressively, making your way to the open deck."

Instead of jumping:
    now the player is perched;
    say "You leap and swing yourself boldly up into the rigging, leaving your attackers beneath you."

Instead of jumping when the player is perched:
    now the player is free;
    say "You leap down from your position, into the middle of the deck."
```

```
Test me with "advance / jump / advance / retreat / jump / retreat / retreat / advance".
```

Of course, this won't be much fun until we also provide the player with a few weapons, some more fighting maneuvers, and, most of all, a Don Pedro to defeat.

---

114 ★ **Example** **Politics as Usual**

Have the status line indicate the current region of the map.

RB

Suppose a game with a large map entirely subdivided into regions. We could define:

```
"Politics as Usual"

When play begins:
    now the right hand status line is "[map region of the location]".

Washington is west of Idaho.

Red is a region. Blue is a region. Idaho is in red. Washington is in blue.

Test me with "e / w".
```

Note that, since regions can be stacked, we technically can be within more than one region at once. In the Port Royal example, for instance, the Tavern region is inside the Inland region. If there is any ambiguity, "the map region of the location" will be construed as "the smallest region that the location belongs to": so we would see "Tavern" rather than "Inland" in the status bar, when the player was in the Feathers or the Feathers Bedroom.

Some extra finesse would be necessary if the names of map regions were very long or if there were some rooms that were not considered to belong to any region at all.

---

115 ★★★ **Example** **Centered**

Replacing the two-part status line with one that centers only the room name at the top of the screen.

RB

If we want to lay out the status line in some other way than with left-hand and right-hand entries, it is possible to do this as well. Later we will learn about the "rule for constructing the status line", but here is a basic effect using this rule and an Inform extension included as part of the standard distribution, called Basic Screen Effects.

```
"Centered"

When play begins:
    say "After months of boring through the Earth's crust in this metal-jawed vehicle, you break through..."
```

```
The Hollow Core is a room. "Truly a magnificent sight: the land curves up away
from you in every direction, covered with the cities and fields of the Core People.
Molten rock runs in the canals, bringing heat and light to every home.

At the center of the Earth hangs a dense black sun."

Include Basic Screen Effects by Emily Short.

Rule for constructing the status line:
    center "[location]" at row 1;
    rule succeeds.

Test me with "look".
```

Basic Screen Effects also provides a mechanism for building complicated status lines of more than one row. To read its documentation, we include the extension, press Go!, and then consult the contents index that results.

116

★ Example Vitrine

RB

An electrochromic window that becomes transparent or opaque depending on whether it is currently turned on.

```
"Vitrine"

Plaza View is a room. "Your uncle's apartment, on loan to you for viewing the
parade and celebrations today. This would be more of a sacrifice on his part if he
weren't currently yachting around Corfu."

The smart window is a device in Plaza View. It is fixed in place. "A vast smart
window [if transparent]overlooks the park[otherwise]has turned to a sheet of
hazy blue[end if]." The smart window can be transparent. The smart window is
transparent. The description is "An electrochromic device which changes shade
and transparency in response to the application of current.

Curtains are so last year."
```

Note the "can be transparent" line. Devices ordinarily are not allowed to have transparency or opaqueness, but we can make an exception in this case. Without that line, attempts to change the transparency of the window will fail.

```
Carry out switching off the window: now the window is transparent.

Carry out switching on the window: now the window is opaque.

Instead of searching a transparent window: say "Isn't it lovely out there?"

Instead of searching an opaque window: say "The window is currently
darkened."
```

```
Test me with "look through window / switch window / look through window /
look".
```

117

★ Example Thirst

A waterskin that is depleted as the player drinks from it.

RB

```
"Thirst"

The player carries a waterskin. The waterskin can be full, partly drained, or
empty. The waterskin is full. Understand "water" as the waterskin.

Instead of drinking the waterskin when the waterskin is empty:
    say "There is no water left."

Instead of drinking the waterskin: if the waterskin is partly drained, now the
waterskin is empty; if the waterskin is full, now the waterskin is partly drained;
say "You drink a long draught."

After printing the name of the waterskin: say " ([waterskin condition])"

Campsite is a room. "It is solid night now, and the stars have come out.
Unfamiliar stars. On the other side of the valley -- a valley round-bottomed but
shallow, like a soup bowl -- burn other campfires, most likely bandits. Their
voices do not carry, but the smoke rises and obscures the starlight over that
way."

A sleepsack is an enterable container in the Campsite. "Your sleepsack is laid
out in a pocket of sandy soil and coarse grass."

The sandy soil, the stars, the distant campfires, and the coarse grass are
scenery in the Campsite. Understand "smoke" as the campfires. Instead of
listening in the presence of your campfire: say "All you hear are the reassuring
snaps and cracks of the sticks in your fire." Understand "campfires" or "fires" as
the distant campfires.

Your campfire is scenery in the Campsite. Instead of pushing, pulling, turning,
tasting, or touching your campfire, say "You would burn yourself." Understand
"fire" as your campfire. The description of your campfire is "A reassuring
protection against wild animals and cold."

The description of the stars is "You invent constellations for them. The slingshot.
The scroll. The heart (upside down)."

Instead of going nowhere when the player is in Campsite:
    say "Now is not the time for wandering, alone in the dark. Better to keep
here[if your campfire is visible], by the fire[end if]."

Singing is an action applying to nothing. Understand "sing" as singing.
```

```
Instead of singing:
    say "You sing, deep and low, a song from home. It is a good night for singing and the song raises your spirits."

Test me with "i / drink water / i / drink water / i".
```

118

★ Example **Thirst 2**

A campfire added to the camp site, which can be lit using tinder.

RB

```
"Thirst"

The player carries a waterskin. The waterskin can be full, partly drained, or empty. The waterskin is full. Understand "water" as the waterskin.

Instead of drinking the waterskin when the waterskin is empty:
    say "There is no water left."

Instead of drinking the waterskin: if the waterskin is partly drained, now the waterskin is empty; if the waterskin is full, now the waterskin is partly drained; say "You drink a long draught."

After printing the name of the waterskin: say " ([waterskin condition])"

Campsite is a room. "It is solid night now, and the stars have come out. Unfamiliar stars. On the other side of the valley -- a valley round-bottomed but shallow, like a soup bowl -- burn other campfires, most likely bandits. Their voices do not carry, but the smoke rises and obscures the starlight over that way."

A sleepsack is an enterable container in the Campsite. "Your sleepsack is laid out in a pocket of sandy soil and coarse grass."

The sandy soil, the stars, the distant campfires, and the coarse grass are scenery in the Campsite. Understand "smoke" as the campfires. Instead of listening in the presence of your campfire: say "All you hear are the reassuring snaps and cracks of the sticks in your fire." Understand "campfires" or "fires" as the distant campfires.

Your campfire is scenery in the Campsite. Instead of pushing, pulling, turning, tasting, or touching your campfire, say "You would burn yourself." Understand "fire" as your campfire. The description of your campfire is "A reassuring protection against wild animals and cold."

The description of the stars is "You invent constellations for them. The slingshot. The scroll. The heart (upside down)."

Instead of going nowhere when the player is in Campsite:
    say "Now is not the time for wandering, alone in the dark. Better to keep here[if your campfire is visible], by the fire[end if]."
```

```
Singing is an action applying to nothing. Understand "sing" as singing.

Instead of singing:
    say "You sing, deep and low, a song from home. It is a good night for singing and the song raises your spirits."

The player carries a tinderbox. The tinderbox contains a flint, a steel, some tinder, and a patch of carbonized cloth. The description of the flint is "A flat grey stone with flaked edges." The description of the steel is "Curved so that you can hold it over the knuckles of your right hand and strike it against the flint. There is a knack to it. Those without the knack end up with bloody knuckles and no fire." The steel is wearable. The description of the tinder is "Dried grass and similar." The description of the cloth is "The little, precious, spark-preserving scraps without which the fire would never begin."

Instead of attacking the flint when the steel is not worn by the player:
    say "You must wear the steel over your knuckles, in order to hit the flint at the best angle."

Instead of attacking the flint when the steel is worn and the cloth is not in the location:
    say "Though you strike the flint sharply with the steel and throw sparks, they have nothing to catch on, since the patch of cloth does not lie beneath."

Instead of attacking the flint when the steel is worn and the cloth is in the location and the cloth is not unlit:
    say "The patch of cloth has already caught."

Instead of attacking the flint:
    now the cloth is glowing;
    say "You strike the flint against the steel and throw sparks onto the patch of cloth; they make tiny circles of orange there, which will only prosper if blown into flame."
```

Realistically, we ought to attach a randomization to this so that each step of the fire-starting has a good chance of failure. But because our player may not be as patient as someone who actually needs a fire started, we allow him to succeed the first time in every case.

```
Ignition is a kind of value. The ignitions are whole, fading, glowing, flaming. A thing has an ignition.

Blowing on is an action applying to one thing. Understand "breathe on [something]" or "blow on [something]" as blowing on.

Instead of blowing on the whole cloth:
    say "There is no point, since no sparks have caught there."

Instead of blowing on the fading cloth:
    now the cloth is flaming;
    say "You blow on the faint sparks on the cloth and turn them into the beginnings of flame."
```

```
Instead of burning the whole tinder:
    if the cloth is not flaming:
        say "The patch of cloth must catch flame before you can light anything with it.";
    otherwise:
        now the tinder is flaming;
        now the cloth is nowhere;
        say "You light the tinder with the patch of cloth, and have the elements of a fire."

Every turn:
    unless the cloth is flaming or the cloth is whole:
        now the ignition of the cloth is the ignition before the ignition of the cloth;
        say "Now the patch of cloth is [ignition of the cloth]."

Some kindling is in the campsite.

Instead of burning the whole kindling:
    if the tinder is not flaming:
        say "You need the tinder to be flaming, first.";
    otherwise:
        now the tinder is nowhere;
        now the kindling is nowhere;
        move the campfire to the location;
        say "You succeed in lighting yourself a proper campfire.";
        now the printed name of Campsite is "By The Campfire".

Test me with "i / drink water / i / drink water / i / wear steel / get flint / get cloth / drop cloth / get tinder / hit flint / blow on cloth / burn tinder / burn kindling / look".
```

---

119 Example **Meteoric I and II** RB

A meteor in the night sky which is visible from many rooms, so needs to be a backdrop, but which does not appear until 11:31 PM.

The game below begins at half past eleven, and one turn later, it's meteor time:

```
"Meteoric I"

The time of day is 11:30 PM.

At 11:31 PM:
    now the meteor is in the great outdoors;
    say "A meteor streaks across the sky.".

The great outdoors is a region. The Spanish Balcony is east of the Inner Court. The Court and Balcony are in the great outdoors. Inside from the Court is the Swimming Pool.

The meteor is a backdrop. Instead of doing something to the meteor, say "The meteor is no longer visible, now nothing more than a memory."
```

```
Test me with "wait / wait / examine meteor / west / examine meteor / in / examine
meteor".
```

Or for something a little slower-moving and with no after-image:

```
"Meteoric II"

The time of day is 4:30 PM.

At 4:31 PM:
    now Phobos is in the great outdoors;
    say "Phobos rises from the western horizon."
At 10:06 PM:
    now Phobos is nowhere;
    say "Phobos sets over the eastern horizon."

The great outdoors is a region. The Martian Balcony is east of the Inner Court.
The Court and Balcony are in the great outdoors. Inside from the Court is the
Heavy Water Swimming Pool.

Phobos is a backdrop. Instead of doing something to Phobos, say "Phobos
orbits a mere 6000km above you, which is practically touching range for
astronomy. On the other hand, astronomy isn't all that practical."

Test me with "wait / wait / examine phobos / west / examine phobos / in /
examine phobos".
```

Though we should not really use Earthly time-keeping, since the Martian day is about half an hour longer than ours.

---

120 ★★★ Example **Orange Cones** RB

Creating a traffic backdrop that appears in all road rooms except the one in which the player has laid down orange cones.

Because we can invent our own adjectives (see the chapter on Phrases), we can make the conditions for a backdrop as simple or as complicated as we like.

In this scenario, we want the player to be able to take, move, and drop orange traffic cones to seal off one street or another. So we create our own "accessible" adjective as follows:

```
Definition: a road is accessible if the orange cones are not in it.
```

...and now

```
move the traffic backdrop to all accessible roads.
```

will tell the traffic backdrop where to appear.

```
"Orange Cones"

A road is a kind of room.

The traffic is a backdrop. It is not scenery. The initial appearance is "Dense traffic snarls the streets, making it difficult to cross even with the lights. Men on motorbikes edge between the cars, and sometimes pull up onto the sidewalks to go around." The description is "It is more or less as usual for this time of day. It's a wonder it ever dissipates, really."

When play begins:
    move the traffic backdrop to all accessible roads.

A line of orange cones are a thing.

Definition: a road is accessible if the orange cones are not in it.

After dropping the orange cones in a road:
    say "With steely determination you begin to lay out the orange cones, blocking access to this segment of street. This produces honking and swearing -- but you persevere.";
    update backdrop positions.

After taking the orange cones:
    say "You go around taking up the orange cones, and within moments the traffic begins to flow into the street again.";
    update backdrop positions.
```

That accomplishes everything we set out to do, but let's add a very simple puzzle to test it out with:

```
The Town Square is a road. North of the Town Square is Candle Street. Candle Street is a road. East of Town Square is Mortar Street. Mortar Street is a road.

The line of orange cones are in Mortar Street.

The player wears a reflective vest and a hard hat.

The manhole cover is a door. "[if location is accessible]Under the cars in the middle of the street, you can just make out the cover of the manhole you need to get into.[otherwise]There's a promising manhole in the middle of the street.[end if]". It is closed and openable. It is below the Town Square and above the Access Tunnel.

Instead of opening the manhole cover when the location is an accessible road:
    say "You can't get anywhere near the manhole cover with all these cars above it."

Instead of taking the orange cones in Town Square:
    if the manhole cover is closed or Town Square is accessible, continue the action;
    otherwise say "You'd better not let the traffic back in until you've closed the manhole. There'll be accidents otherwise."
```

```
After going to the Access Tunnel:
    say "With a stealthy glance left and right, you lower yourself into the access tunnel, thus accomplishing your mission for Chapter 2. To continue your adventure, see Chapter 3: The Vault of Peaquod.";
    end the story finally.
```

And finally, a couple of features from the Activities chapter to make the output more elegant:

```
Rule for writing a paragraph about the orange cones: say "A line of orange cones holds back the traffic from entering here."

Rule for printing the name of the orange cones when the cones are carried by the player:
    say "stack of orange cones".

Test me with "x traffic / open manhole / n / x traffic / s / e / x traffic / get cones / look / x traffic / w / drop cones / look / open manhole / take cones / d".
```

121 ★★★ Example **Terror of the Sierra Madre** RB

Multiple player characters who take turns controlling the action.

Suppose we have a game where we want the player to control two different characters, swapping bodies from one turn to the next. First, the setting, and the two people who will alternately play:

```
"Terror of the Sierra Madre"

The Hay-Strewn Corridor is a room. "[if the player is Maleska]The horse stalls are empty: you have already drained the animals, and carried off their corpses. The house will not long sustain you now.

The window throws on the floor a bright square of malevolent sunlight[otherwise]The stalls for horses run down one side of the room, but the house has long stood empty. A square window without shutters looks out over the ranch, away toward the Sierras[end if]."

Teresa is a woman in the Hay-Strewn Corridor. "Teresa stands opposite you[if Teresa carries something], her fingers wrapped tightly around [a list of things carried by Teresa][end if]." Teresa carries a bulb of garlic and a cross.

Maleska is a man in the Hay-Strewn Corridor. "Maleska watches you from eyes entirely black." Maleska carries a skull.
```

If we tried the text above in Inform, we would find ourselves in the Hay-Strewn Corridor and confronted by both Teresa and Maleska. If "player" is not set to any named person, Inform creates a bland person called "yourself" to represent the player. To avoid this, we set "player" to the person we want to begin as. The player character is normally privately-named, so we'll need to make sure "Maleska" still means what it should.

```
The player is Maleska. Understand "Maleska" as Maleska.
```

Now the Corridor contains just two people, and we arrive on the scene as Maleska, with only Teresa facing us.

At the end of every turn we will use the 'now the player is...' phrase. (This looks as if it simply changes the value of "player": which it does, but it also carries out a complicated operation behind the scenes to effect the switch.)

```
Every turn:
    if the player is Maleska, now the player is Teresa;
    otherwise now the player is Maleska.
```

Our two characters already see the Corridor differently, but let's differentiate them further:

```
Every person has a number called strength. The strength of Teresa is 3. The
strength of Maleska is 5.
```

In this small example, strength is not used for anything, except that we will display it on the status line:

```
When play begins:
    now the command prompt is "[bold type][player][roman type] > ";
    now the left hand status line is "[player]";
    now the right hand status line is "STR: [strength of the player]".
```

That last rule doesn't quite do what we might have expected. When we print "[player]", we find that Inform usually prints "yourself". This is because Inform says "you" to mean Teresa when talking to the player-being-Teresa, and likewise for Maleska. We want to override that in this particular story, because the rapid switches of personality are otherwise hard to follow. So:

```
Rule for printing the name of Teresa: say "Teresa".

Rule for printing the name of Maleska: say "Maleska".

Test me with "look / look".
```

122 ★ **Example** **Beverage Service** RB

A potion that the player can drink.

Some kinds of game objects -- food, for instance -- can only sensibly be used once, and should then be destroyed. The EAT command already implements this, but suppose we also had a category of drinkable potions:

```
"Beverage Service"
```

```
A potion is a kind of thing. The sparkly blue potion is a potion carried by the player.

Level 3 is a room.

Instead of drinking a potion (called the drink):
    now the drink is nowhere;
    say "You quaff [the drink]. It goes down beautifully."

Test me with "drink sparkly / i".
```

---

123 ★ **Example** **Spring Cleaning** *RB*

A character who sulks over objects that the player has broken (and which are now off-stage).

Here we have a destruction action that allows the player to break any fragile items. Once destroyed, these things are removed from play, but we can still refer to them: they are now off-stage. This makes it easy for our sulking character to list the ones that have been destroyed:

```
"Spring Cleaning"

A thing can be tough or fragile. A thing is usually tough.

Instead of attacking something fragile:
    say "You smash [the noun] to smithereens!";
    now the noun is nowhere.

A knick-knack is a kind of thing which is fragile.

Every turn when a knick-knack is off-stage and Granny Blue can see the player:
    say "'Ohh,' whimpers Granny to herself softly. 'How I will miss [the list of off-stage knick-knacks]!'"

The Parlor is a room. Granny Blue is a woman in the Parlor. A china lamb, a porcelain milkmaid, a frolicking Dutch cow, and a crystal unicorn are knick-knacks in the Parlor.

Test me with "break lamb / break milkmaid / break cow / break unicorn".
```

---

124 ★★★ **Example** **Extra Supplies** *RB*

A supply of red pens from which the player can take another pen only if he doesn't already have one somewhere in the game world.

Suppose we have a supply closet in our game from which the player is allowed to take red pens. To keep modeling simple, we only allow him to have one in play at a time, and we test this by seeing whether the red pen is "off-stage" before moving it to his possession.

This approach might seem no different from having a single red pen sitting in the closet, but it may be preferable, for two reasons. First, it's not very plausible for a supply closet to contain nothing but a single red pen (well, assuming a well-regulated supplier, anyway); and second, it gives the player a way to get a new red pen should the original be destroyed in a tragic handwriting accident.

```
"Extra Supplies"

The Supply Closet is a room. A supply of red pens is in the Supply Closet.
Understand "pen" as the supply of red pens when the red pen is not visible.

There is a red pen.

Instead of taking the supply of red pens:
    if the red pen is off-stage:
        move the red pen to the player;
        say "You help yourself to a fresh red pen.";
    otherwise:
        say "You're only allowed one pen at a time. The department secretary is
very strict."

South of the Supply Closet is the Furnace Room. The incinerator is a thing in the
Furnace Room. It is a container. "The incinerator is here, working full blast."

After inserting something into the incinerator:
    now the noun is nowhere;
    say "A fiery blast consumes [the noun]!"

Test me with "get pen / i / get pen / get supply / s / put pen in incinerator / n / get
pen".
```

---

125 ★ **Example** **Bee Chambers** *RB*

A maze with directions between rooms randomized at the start of play.

Mazes are a traditional element of interactive fiction, often consisting of apparently identical rooms with exits that do not work reciprocally and which cause confusion.

The methods of mapping mazes are now fairly well understood and mazes themselves tend to be regarded as tiresome rather than enjoyable by a large portion of the playing audience. However, if we did want to ignore the common wisdom and create a maze, randomly generated at the start of play, here would be one way to go about it:

```
"Maze of Gloom"

A Bee Chamber is a kind of room. The printed name of a Bee Chamber is
usually "Hexagonal Room". The description of a Bee Chamber is usually "Waxy,
translucent walls surround you on six sides; the floor and ceiling are made of the
same material, gently uneven. There are exits in every direction, cut into the
faces or the corners."

Bee1, Bee2, Bee3, Bee4, Bee5, Bee6, Bee7, Bee8, Bee9, and Bee10 are Bee
Chambers.

When play begins:
    now right hand status line is "[number of visited rooms]/[number of rooms]";
    repeat with place running through Bee Chambers:
        now a random Bee Chamber is mapped north of place;
        now a random Bee Chamber is mapped northwest of place;
        now a random Bee Chamber is mapped west of place;
        now a random Bee Chamber is mapped southwest of place;
        now a random Bee Chamber is mapped south of place;
        now a random Bee Chamber is mapped southeast of place;
        now a random Bee Chamber is mapped east of place;
        now a random Bee Chamber is mapped northeast of place;
        now a random Bee Chamber is mapped above place;
        now a random Bee Chamber is mapped below place;
        now a random Bee Chamber is mapped inside place;
        now a random Bee Chamber is mapped outside place.

Test me with "in / out / up / down / n / ne / nw / e / w / sw / se / s".
```

126 ★★★ **Example Hatless** RB

It's tempting to use "now..." to distribute items randomly at the start of play, but we need to be a little cautious about how we do that.

Suppose we want a game in which each scenario starts with the characters wearing hats -- randomly passed out. We might be tempted to write our scenario like this:

```
"Hatless"

The Costumery is a room. Larry, Curly, and Moe are men in the Costumery.
Janine is a woman in the Costumery.

Rule for writing a paragraph about a person (called the target) who wears a hat
(called attire):
    say "[The target] is here, looking stylish in [an attire]."

Rule for writing a paragraph about a hatless person (called the target):
    say "[The target] mopes about, hatless."

A hat is a kind of thing. A hat is always wearable. Definition: a person is hatless
if he is not the player and he does not wear a hat.

The indigo bowler, the polka-dotted fedora, the pink beret, and the scarlet cloche
are hats.
```

```
When play begins:
    now every hat is worn by a random hatless person.
```

And we might hope that this would choose a new hatless person for each hat, but we would be wrong. It will instead choose one hatless person and put all the hats on him -- and everyone else has to go bare-headed. That's clearly no good. Let's try again:

```
"Hatless 2"

The Costumery is a room. Larry, Curly, and Moe are men in the Costumery.
Janine is a woman in the Costumery.

Rule for writing a paragraph about a person (called the target) who wears a hat
(called attire):
    say "[The target] is here, looking stylish in [an attire]."

Rule for writing a paragraph about a hatless person (called the target):
    say "[The target] mopes about, hatless."

A hat is a kind of thing. A hat is always wearable. Definition: a person is hatless
if he is not the player and he does not wear a hat.

The indigo bowler, the polka-dotted fedora, the pink beret, and the scarlet cloche
are hats.

When play begins:
    now every hatless person wears a random hat.
```

But this selects one random hat and assigns it to each hatless person in turn -- so it will only wind up being worn by the last of them (since Inform knows that only one person can wear a hat at a time).

In this case, we do have to expand out our loop so that the game makes an explicit distribution:

```
"Hatless 3"

The Costumery is a room. Larry, Curly, and Moe are men in the Costumery.
Janine is a woman in the Costumery.

Rule for writing a paragraph about a person (called the target) who wears a hat
(called attire):
    say "[The target] is here, looking stylish in [an attire]."

Rule for writing a paragraph about a hatless person (called the target):
    say "[The target] mopes about, hatless."

A hat is a kind of thing. A hat is always wearable. Definition: a person is hatless
if he is not the player and he does not wear a hat.

The indigo bowler, the polka-dotted fedora, the pink beret, and the scarlet cloche
are hats.
```

```
When play begins:
    repeat with item running through hats:
        now the item is worn by a random hatless person.
```

Each time Inform considers the instruction "now the item is worn by a random hatless person", there is one fewer such person to choose from -- so we can guarantee that the hats are distributed one per customer and that all hats are distributed.

Hatless 3 is only guaranteed to work because the number of hats is less than or equal to the number of people; otherwise the final use of random will return "nothing" and then a problem message will appear during play.

127

★★★ Example Technological Terror

RB

A ray gun which destroys objects, leaving their component parts behind.

```
"Technological Terror"

The Decomposition Ray Gun is a thing carried by the player.
```

First we need to define our shooting action:

```
Shooting it with is an action applying to two things.

Check shooting something with something:
    if the player is not carrying the Ray Gun, say "You are pathetically unarmed!" instead;
    if the second noun is not the Ray Gun, say "[The second noun] does not fire." instead;
    if the noun is the Ray Gun, say "Nice trick if you can do it!" instead;
    if the noun is the player, say "That would be disastrous!" instead.
```

Next, some grammar to allow the player to use this action:

```
Understand "shoot [gun] at [something ungunlike]" as shooting it with (with nouns reversed).

Definition: a thing is ungunlike if it is not the gun.

Understand "shoot [something ungunlike] with [gun]" as shooting it with.
Understand "shoot [something] with [something]" as shooting it with.

Understand "shoot [something] at [something]" as shooting it with (with nouns reversed). Understand "fire [gun] at [something ungunlike]" as shooting it with (with nouns reversed). Understand "fire at [something ungunlike] with [gun]" as shooting it with. Understand "fire at [something] with [something]" as shooting it with.
```

Strictly speaking, we only need these last grammar lines (with "understand shoot something...") in order to define an action that the player can take. Adding more grammar lines means that Inform will try to match the most specific ones first, which is useful when the player types something ambiguous and there is one choice that obviously fits this action better than the others. See the chapter on Understanding for a further discussion.

Here we get to use "now..." to give it its destructive effect:

```
Carry out shooting something with something:
    say "ZAP! [The noun] twinkles out of existence! [if something is part of the
noun][The list of things which are part of the noun] clatter to the ground! [end if]
[paragraph break]";
    now every thing which is part of the noun is in the location;
    now the noun is nowhere.

The Deathbot Assembly Line is a room. "Here is the heart of the whole
operation, where your opponents are assembled fresh from scrap metal and bits
of old car." The dangerous robot is a thing in the Assembly Line. "One
dangerous robot looks ready to take you on!" A robotic head, a drill arm, a
needle arm, a crushing leg and a kicking leg are parts of the dangerous robot.

Instead of examining something when something is part of the noun:
    say "[The noun] consists of [a list of things which are part of the noun]."

Test me with "x robot / shoot robot / fire at kicking leg / shoot gun at drill arm /
look".
```

128

★ Example **Higher Calling**

RB

All doors in the game automatically attempt to open if the player approaches them when they are closed.

"Called" is quite useful in the context of rules about going, since go rules often refer to things other than the noun the player typed. For instance, suppose we want to have doors which politely open when the player tries to pass through them:

```
"Higher Calling"

Before going through a closed door (called the blocking door):
    say "(first opening [the blocking door])[line break]";
    silently try opening the blocking door;
    if the blocking door is closed, stop the action.

Dome is a room. North of Dome is North Chapel. South of the Dome is South
Chapel. West of the Dome is Western End. Quiet Corner is northwest of the
Dome, north of Western End, and west of North Chapel. Loud Corner is east of
North Chapel, northeast of Dome, and north of Eastern End. Eastern End is
north of Dim Corner and east of Dome. Dim Corner is southeast of Dome and
east of South Chapel. Ruined Corner is southwest of Dome, west of South
Chapel, and south of Western End.
```

```
The church door is east of Eastern End and west of the Courtyard. The church door is a door.

Test me with "e / e".
```

A fuller set of such rules is included in the Locksmith extension.

129 ★ **Example** **Do Pass Go** RB

A pair of dice which can be rolled, and are described with their current total when not carried, and have individual scores when examined.

```
"Do Pass Go"

Go is a room. "A giant square area, where you and your other pewter ornament friends gather before setting out to purchase London."

The pair of dice is carried by the player.

The pair of dice has a number called first die. The pair of dice has a number called second die. The first die of the pair is 6. The second die of the pair is 6.
Rule for printing the name of the pair of dice while taking inventory: say "pair of dice".
Rule for printing the name of the pair of dice: say "pair of dice showing [first die of the pair plus second die of the pair]".

To say detailed state of the dice:
    if the first die of the pair is the second die of the pair, say "double [first die of the pair]";
    otherwise say "[first die of the pair] and [second die of the pair]".

The description of the pair of dice is "The pair of dice are [if the dice are carried]itching to be rolled[otherwise]showing [detailed state of the dice][end if]."

Rolling is an action applying to one carried thing. Understand "roll [something preferably held]" as rolling.

Check rolling when the noun is not the pair of dice: say "Not something you can roll." instead.
Carry out rolling:
    now the pair of dice is in the holder of the actor;
    now the first die of the pair of dice is a random number from 1 to 6;
    now the second die of the pair of dice is a random number from 1 to 6.
Report rolling:
    say "You roll [detailed state of the dice]."

Test me with "i / roll dice / look / x dice / get dice / x dice / roll dice / roll dice / roll dice / roll dice / roll dice / roll dice / roll dice".
```

Because we remember the states of the individual dice, not just a total, we can make use of the combination rolled.

```
The doubles count is a number that varies.
After rolling:
    if the first die of the pair is the second die of the pair, increment the doubles
count;
    otherwise now the doubles count is 0;
    continue the action.

Jail is a room. "This is Jail, and not the Just Visiting periphery, either."

Every turn when the doubles count is 3:
    say "The blue-uniformed policemen blows his whistle and beckons you
sternly...";
    now the player carries the pair of dice;
    now the player is in Jail;
    now the doubles count is 0.

Every turn when the doubles count is 1 and the player is in Jail:
    say "The warden gruffly releases you.";
    now the player carries the pair of dice;
    now the player is in Go.
```

130

★ Example **Lanista 1**

Very simple randomized combat in which characters hit one another for a randomized amount of damage.

RB

```
"Lanista, Part One"

The Arena is a room. "Sand, blood, iron. These festivals are normally held on
hot days, but the sun has gone behind a cloud and fat drops of rain now and
then spatter the arena floor." The gladiator is a man in the Arena. "A bare-
chested Scythian gladiator faces you, wielding a trident."
```

We start by recording, for each person, a maximum number of points of damage the person can sustain when starting from health, and the current number of points remaining. In the tradition of role-playing games, these are referred to as hit points.

```
A person has a number called maximum hit points. A person has a number
called current hit points.

The maximum hit points of the player is 35. The maximum hit points of the
gladiator is 25.

The current hit points of the player is 35. The current hit points of the gladiator is
25.
```

Now our rule for the actual attack. We want first to calculate how much damage the player's attack does, inflict that damage, and remove the enemy if he's dead; then, if he doesn't die, the enemy counter-attacks, also for a randomized amount of damage, and if this kills the player, the game ends in defeat.

```
Instead of attacking someone:
    let the damage be a random number between 2 and 10;
    say "You attack [the noun], causing [damage] points of damage!";
    decrease the current hit points of the noun by the damage;
    if the current hit points of the noun is less than 0:
        say "[line break][The noun] expires, and is immediately carried away by the Arena slaves!";
        now the noun is nowhere;
        end the story finally;
        stop the action;
    let the enemy damage be a random number between 2 and 10;
    say "[line break][The noun] attacks you, causing [enemy damage] points of damage!";
    decrease the current hit points of the player by the enemy damage;
    if the current hit points of the player is less than 0:
        say "[line break]You expire!";
        end the story.
```

This last bit is a refinement to help the player keep track of how the contest is going:

```
When play begins:
    now the left hand status line is "You: [current hit points of player]";
    now the right hand status line is "Gladiator: [current hit points of gladiator]".

Test me with "hit gladiator / g / g / g".
```

---

131 ★ Example **Weathering**

RB

The automatic weather station atop Mt. Pisgah shows randomly fluctuating temperature, pressure and cloud cover.

```
"Weathering"

A cloud pattern is a kind of value. The cloud patterns are cumulus, altocumulus, cumulonimbus, stratus, cirrus, nimbus, nimbostratus.

The Mount Pisgah Station is a room. "The rocky peak of Mt. Pisgah (altitude 872m) is graced only by an automatic weather station. The clouds, close enough almost to touch, are [a random cloud pattern]. Temperature: [a random number from 7 to 17] degrees, barometric pressure: [950 + a random number from 0 to 15] millibars."

Test me with "look / look / look".
```

---

132 ★★★ Example **Uptown Girls**

RB

A stream of random pedestrians who go by the player.

Suppose we have an urban space we want to populate with random passers-by. These should have a range of characteristics and not always be described in the same way; and once the player has noticed one, he should be able to look at her further, until another pedestrian crosses his path.

```
"Uptown Girls"

Riverside Drive is a room. "There's a pleasant late-afternoon view of the Hudson, and a snap in the air, and you would rather be here than anywhere."

Instead of going a direction, say "Oh, you know where you're going; no need to deviate from the usual path."

Instead of waiting, say "You stroll along enjoying the November crispness."
```

It gets a little annoying to have a random event occurring every single turn of play, so let's introduce some randomness to determine how often the message appears:

```
Every turn when a random chance of 1 in 3 succeeds:
    reset passerby;
    choose a random row in the Table of Atmospheric Events;
    say "[event entry][paragraph break]"

Table of Atmospheric Events
event
"Slowly [a passerby] strolls by, turning to look at you as she passes."
"Some [passerby] nearly bumps into you."
"You dodge to avoid [a passerby]."
"You weave around [a passerby], who has stalled to look into a window."
"There's a ruckus as one of the ubiquitous taxis nearly collides with [a passerby] crossing the street."
"[The passerby] beside you waves to a friend across the street."
"To your left, [a passerby] drops her purse, and swears as she retrieves it."

Hair color is a kind of value. A person has hair color. the hair colors are red-headed, brunette, blonde.

Height is a kind of value. A person has height. The heights are tall, medium-height, short.

Grooming is a kind of value. A person has grooming. The groomings are messy and tidy.

To reset passerby:
    now the hair color of the passerby is a random hair color;
    now the height of the passerby is a random height;
    now the grooming of the passerby is a random grooming.

The passerby is a woman in Riverside Drive. The passerby is scenery. Understand "woman" or "lady" as the passerby. The printed name of the passerby is "[one of]woman[or]lady[purely at random]".
```

At this point we borrow some techniques from later to describe the woman with a random combination of characteristics, and to let the player refer to her by those traits:

```
Before printing the name of the passerby:
    if a random chance of 1 in 4 succeeds, say "[height] ";
    if a random chance of 1 in 4 succeeds, say "[grooming] ";
    if a random chance of 1 in 4 succeeds, say "[hair color] ".

Understand the hair color property as describing the passerby. Understand the height property as describing the passerby. Understand the grooming property as describing the passerby.
```

If we also wanted each of those combinations to mean some more specifically-described woman:

```
Instead of examining a passerby:
    repeat through Table of Passerby Descriptions:
        if hair entry is the hair color of the passerby and height entry is the height of the passerby and grooming entry is the grooming of the passerby, say "[description entry][paragraph break]".

Table of Passerby Descriptions
```

hair	height	grooming	description
red-headed	tall	messy	"An older woman with long red hippie-hair poking out of a ponytail in straggles, and bent to hide how tall she is."
red-headed	medium-height	messy	"A shaggy red-head with shingled hair."
red-headed	short	messy	"Almost an urchin, and very young, with ginger hair and a smudged nose and far too many freckles."
red-headed	tall	tidy	"A precise career woman with henna-red hair."
red-headed	medium-height	tidy	"Her hair is red in the way that lollipops and fire trucks are red: not by nature but by art. The rest of her clothing is pretty ordinary, though."
red-headed	short	tidy	"Thin and small in every sense, with chin-length red-hair. Even high heels do not bring her head much above your shoulder."
brunette	tall	messy	"A Juno-esque woman with dark hair, wearing something resembling a tent."
brunette	medium-height	messy	"An unremarkable woman with dark brown hair and the aura of needing a wash."
brunette	short	messy	"There are mustard stains on the t-shirt of this short brown-haired woman. Estimated age ca. 40. Possibly homeless."
brunette	tall	tidy	"A leggy brunette in business attire."
brunette	medium-height	tidy	"Medium-height, brown-haired, generally nondescript."
brunette	short	tidy	"A neat little dark-haired girl."
blonde	tall	messy	"A tall blonde of about thirteen who looks as though she has not yet figured out how to get her wardrobe to catch up with her rate of growth. Her t-shirt and her pants are too short."
blonde	medium-height	messy	"Black leather pants and the wall-o-hair look."
blonde	short	messy	"One of those shocking platinum blonde types, with a tiger-patterned skirt. Reeally trashy."
blonde	tall	tidy	"Elfin and severe, with perfectly straight hair falling to the middle of the back."
blonde	medium-height	tidy	"A rounded, Marilyn-esque blonde."
blonde	short	tidy	"Pin-precise in a blue-and-white striped suit and a boyish haircut."

```
Test me with "z / z / x passerby / z / z / x passerby".
```

★ Example Candy

133 One of several identical candies chosen at the start of play to be poisonous. RB

Suppose we want to give the player a bag of candies, of which a random one is poisonous. We can pick which one should be poisoned at the start of play, like this:

```
"Candy"

The plural of piece of candy is pieces of candy. A piece of candy is a kind of
thing. A piece of candy is always edible. Four pieces of candy are in the
Halloween bag.

Toxicity is a kind of value. The toxicities are safe and poisonous. A piece of
candy has a toxicity. A piece of candy is usually safe.

The Porch is a room. The player carries the Halloween bag.

After eating a poisonous piece of candy:
   say "Oh, that didn't taste right at all. Oh well!"

When play begins:
   now a random piece of candy is poisonous.

Test me with "eat candy / g / g / g".
```

134 Example Zork II RB

A "Carousel Room", as in Zork II, where moving in any direction from the room leads (at random) to one of the eight rooms nearby.

All we need to do is select the player's destination for him at random:

```
"Zork II"

The Carousel Room is a room.

Instead of going from the Carousel Room:
   move the player to a random adjacent room.
```

To avoid infringing the original game too much, let's try a somewhat different setting:

```
The Games of Chance is north of the Carousel Room. The Haunted Funhouse is
northwest of the Carousel Room. The Ferris Wheel is east of the Carousel
Room. The Topsy-Turvy is northeast of the Carousel Room. The Reproduction
Henge is south of the Carousel Room. The Women's Toilet is southwest of the
Carousel Room. The Men's Toilet is southeast of the Carousel Room. The
Cotton Candy Shop is west of the Carousel Room.
```

```
Test me with "s".
```

And the following means that the test runs consistently even though the numbers are theoretically random. To make them truly random, remove this line.

```
When play begins, seed the random-number generator with 1234.
```

Or if we want to add the refinement that the Carousel Room can be switched off:

```
"Zork II"

The Carousel Room is a room. The spinning machine is a switched on device in
the Carousel Room.
```

And then

```
Instead of going from the Carousel Room when the spinning machine is
switched on:
    move the player to a random adjacent room.

The Games of Chance is north of the Carousel Room. The Haunted Funhouse is
northwest of the Carousel Room. The Ferris Wheel is east of the Carousel
Room. The Topsy-Turvy is northeast of the Carousel Room. The Reproduction
Henge is south of the Carousel Room. The Women's Toilet is southwest of the
Carousel Room. The Men's Toilet is southeast of the Carousel Room. The
Cotton Candy Shop is west of the Carousel Room.

Test me with "turn off machine / s / n / turn on machine / s".

When play begins, seed the random-number generator with 1234.
```

Chapter 9: Time

Contents of *Writing with Inform*

Chapter 8: Change

Chapter 10: Scenes

Indexes of the examples

§9.1. When play begins

With the material from previous chapters, we are now able to set up a simulated world, to respond to the player's actions within it, and to make it change in perhaps unexpected ways.

The resulting experience can be as lively as we want to make it, but so far we have no very good ways to give it any structure: a beginning and an end, for instance, or a sense of plot. This next chapter is all about the passage of time, and it begins at the beginning.

We have already seen an example of how to write a rule which applies just once, at the start of the story:

```
When play begins: say "Welcome to Old Marston Grange, a country house cut off by fog."
```

The "when play begins" rules are checked only at the start of a story, not when a saved session is restored from disc. What happens is that these rules are followed, then the story's banner is printed, then the initial room description is printed up, and then the player is asked for a first command.

Start of Chapter 9: Time

Back to Chapter 8: Change: §8.19. Random choices of things

Onward to §9.2. Awarding points

Example 135: ★ **Clueless** A murderer for the mystery is selected randomly at the beginning of the game.

§9.2. Awarding points

Traditionally-written stories award points throughout play, as an indication of progress. If we want to be traditional, we must first write:

```
Use scoring.
```

Without this, the SCORE, NOTIFY ON and NOTIFY OFF commands do not work; the final score is not shown at the end of a story; and the status line above the player's text area shows only the turn count, not (as is more usual) both the score and the turn count. Changing the "score" (see below) has no visible effect, though it is not actually illegal.

With "Use scoring" in place, we can award points as follows:

```
increase the score by 5;
```

substituting whatever number we feel is appropriate. We should be careful not to give out the same points over and over, that is, not to reward the same basic achievement many times over if the player simply repeats the same action. This, for instance, is open to abuse:

```
After taking the trophy:
    increase the score by 5;
    say "Well done!"
```

The player may simply take the trophy, drop it again, take it again, ... and win five points every time around. We can prevent this by phrasing the rule more carefully:

```
After taking the trophy when the trophy is not handled:
    increase the score by 5;
    say "Well done!"
```

Rather than being an open-ended scoring system, IF normally has a maximum possible score, which can be specified with a sentence like so:

```
The maximum score is 10.
```

The score and maximum score are just numbers that vary, so we can freely change them:

```
After eating the poisoned mushroom:
    now the score is -100.
```

⬆ Start of Chapter 9: Time

⬅ Back to §9.1. When play begins

➡ Onward to §9.3. Introducing tables: rankings

⬇ Example 136: ★★ **Mutt's Adventure** Awarding points for visiting a room for the first time.

⬇ Example 137: ★★★ **No Place Like Home** Recording a whole table of scores for specific treasures.

§9.3. Introducing tables: rankings

Another tradition of interactive fiction is that the player has a current 'rank' according to how far his or her score has got. We can (but need not) choose to provide such rankings, and should do so by specifying a table like this:

```
Table 1 - Rankings

    Score	Rank
    0	"Beginner"
    25	"Amateur Adventurer"
    50	"Novice Adventurer"
    100	"Junior Adventurer"
    200	"Adventurer"
    300	"Master"
    330	"Wizard"
    350	"Master Adventurer"
```

Typographically, tables in Inform look as much as possible like those found in non-fiction books: they can have many columns, so this is only a simple example (drawn from the actual rankings used by Infocom's *Zork I*, 1979). Each line in the source represents one row in the table, and the entries on a line must be separated by at least one tab character. (An entry might of course have several words with spaces in between, so a space is not enough to separate entries: this is the only context when Inform distinguishes between spaces and tabs.) The table must occupy a single whole paragraph, with no skipped lines or missing entries. We are free to use extra tabs to indent it if we like.

Ordinarily we must explicitly ask to use the information presented in a table, but the table of rankings is a special case: Inform uses it automatically, if it is provided at all. To be recognised it must have the name "Rankings" and must have two columns named and arranged as shown. The scores should be given in ascending order. Customarily, the score in the final row should be the maximum possible achieved in the story - so that only a player with maximum possible score can be awarded the final ranking - and the value of "maximum score" is automatically set to this bottom-row value if it has not been explicitly set anywhere else in the source text.

Start of Chapter 9: Time

Back to §9.2. Awarding points

Onward to §9.4. When play ends

§9.4. When play ends

Short of something like a power cut, the story can only end when one of the two participants chooses to end it: either the player, by deciding that enough is enough, or us.

In story-telling, there are many kinds of ending: happy, sad, decisive, bittersweet, surprise. Inform doesn't try to interfere in that kind of artistic choice, but it does need to know one thing about the ending: is it final, or not? Many authors like to make additional menu items available if the player has completed the story right through, but not if she has reached an early or wrong ending. (See the activity "amusing a victorious player", for example.)

end the story

This phrase ends the story at the next opportunity (typically as soon as the current rule ends), with the closing message "The End." The end is not considered final.

end the story finally

This phrase ends the story at the next opportunity (typically as soon as the current rule ends), with the closing message "The End." The end is considered final, and any hidden menu options will be revealed.

end the story saying (text)

This phrase ends the story at the next opportunity (typically as soon as the current rule ends), with the closing message given in the text. The end is not considered final. Example:

```
end the story saying "You have been stymied"
```

end the story finally saying (text)

This phrase ends the story at the next opportunity (typically as soon as the current rule ends), with the closing message given in the text. The end is considered final, and any hidden menu options will be revealed. Example:

```
end the story finally saying "You have defeated Sauron"
```

The closing message is traditionally set out in asterisks:

```
*** The End ***
```

This style is traditional in IF, and goes back to 1980 if not earlier.

We can test the current state like so:

if story has ended:

This condition is true if an end has been declared using one of the "end the story..." phrases.

if story has not ended:

This condition is true if no end has been declared using one of the "end the story..." phrases.

if story has ended finally:

This condition is true if an end has been declared using one of the "end the story finally..." phrases, so that an ending has been reached which the author feels is a completion of the player's experience.

if story has not ended finally:

This condition is true if an end has been declared using one of the "end the story..." phrases, but not "finally", so the author feels that the player can get further experience by playing again and trying different approaches.

The rulebook "when play ends" is the matching bookend to "when play begins". It is followed when the story decides to end (not when the player simply gives up and quits), and before any epitaph like

```
*** You have been poisoned ***
```

appears. For example:

```
When play ends, say "Oh dear."
```

Surprisingly, the end is not always the end:

resume the story

This phrase causes an ended story to resume exactly as if no "end the story..." phrase had been used. Example:

```
When play ends:
    if the story has not ended finally:
        say "Oh dear. Still, here's another chance.";
        resume the story.
```

The phrase is likely to be sensible only as part of a "when play ends" rule. Other traditional uses include giving the player three lives, as in an old-school arcade machine.

⬆ Start of Chapter 9: Time

⬅ Back to §9.3. Introducing tables: rankings

➡ Onward to §9.5. Every turn

⬇ Example 138: ★★★ **Big Sky Country** Allowing the player to continue play after a fatal accident, but penalizing him by scattering his possessions around the game map.

§9.5. Every turn

The passage of time in interactive fiction is broken up into a succession of turns, in each of which the player types a request and is given a response. Usually each such request triggers one action, but sometimes a whole sequence are fired off, as when the player types "get all" in a cluttered room.

As we've seen, the variable "turn count" holds the number of turns of play so far. By convention turn number 0 is the time when Inform prints up the banner and any initial text; it becomes turn number 1 when the player's first command is typed.

One of the last things to happen in each turn is that Inform will apply any rules which have been set to occur "every turn", like so:

```
Every turn, say "The summer breeze shakes the apple-blossom."
```

This is equivalent to writing:

```
An every turn rule: say "The summer breeze shakes the apple-blossom."
```

Note that the text about blossom, which will quickly become tiresome, is said at the end of every turn, not at the beginning, and in particular not before the player's first opportunity to type a command.

As usual when defining rules, we can add stipulations: any condition can be attached using "when".

```
Every turn when the location is the Orchard, say "The summer breeze shakes the apple-blossom."
```

```
Every turn when the player can see the rotting fish, say "Your nose twitches involuntarily."
```

Start of Chapter 9: Time

Back to §9.4. When play ends

Onward to §9.6. The time of day

Example 139: ★★★ **Witnessed 1** A kind of battery which can be put into different devices, and which will lose power after extended use.

Example 140: ★★★★ **Text Foosball** A game of foosball which relies heavily on every-turn rules.

§9.6. The time of day

Inform keeps track of the time of day automatically: play ordinarily begins at 9 AM and each turn takes one minute. In some works of interactive fiction, time of day is crucial: in others, it is irrelevant or even, by a sort of tacit convention, non-existent. So Inform does nothing to act upon the time, or to reveal it to the player, without instruction. Nevertheless it is there.

A sentence like the following allows the initial time to be set up as something other than 9 AM:

```
The time of day is 3:13 PM.
```

Here, "3:13 PM" is a constant value of a kind not seen before: it's a kind of value called "time", and the value "time of day" is a time that varies. After one turn it will be 3:14 PM, then 3:15 PM and so on.

Note that the sentence above is an assertion (a statement about the initial state of affairs), not an instruction which can be part of a rule. It would be equivalent to write:

```
When play begins: now the time of day is 3:13 PM.
```

We more often change "time of day" to take care of drastic events:

```
At the time when the player loses consciousness:
    now the time of day is 10:12 AM;
    say "A mist comes over your vision, and when you come to, it is morning and you are in bed."
```

Start of Chapter 9: Time

Back to §9.5. Every turn

Onward to §9.7. Telling the time

Example 141: ★★ **IPA** Shops which each have opening and closing hours, so that it is impossible to go in at the wrong times, and the player is kicked out if he overstays his welcome.

§9.7. Telling the time

Now that we have the time of day, we can of course use this value in room descriptions and the like:

```
The Clock Chamber is a room. "The dark chamber behind the clock face, a mill-room of
gears which grind down the seconds. Through the glass you can see the reversed
hands reading [the time of day]."
```

It seems odd, though, to read a precise numerical description of the time here: after all, it isn't a digital clock. A friendlier version would use:

say "[(time) **in words]"**

This text substitution produces the given time written out in English sentence form. For example:

```
"Through the glass you can see the reversed hands reading [the time of day in
words]."
```

might produce

```
Through the glass you can see the reversed hands reading twenty to nine.
```

To reiterate an example which came up earlier, we could even work the time of day into the command prompt, which would lend the proper sense of urgency to a story played out against the clock:

```
When play begins: now the command prompt is "[time of day] >".
```

Start of Chapter 9: Time

Back to §9.6. The time of day

Onward to §9.8. Approximate times, lengths of time

Example 142: ★ **Situation Room** Printing the time of day in 24-hour time, as in military situations.

§9.8. Approximate times, lengths of time

Clocks and watches vary considerably in how much detail they show, and we tend not to report the time over-precisely: half-past ten is an elastic concept. The following room description for the Clock Chamber comes across much more naturally:

```
The Clock Chamber is a room. "The dark chamber behind the clock face, a mill-room of
gears which grind down the seconds. Through the glass you can see the reversed
hands reading [the time of day to the nearest five minutes in words]."
```

The phrase "... to the nearest ..." rounds off the given time, just as it sounds; as we'll see later, it can actually round off any arithmetic values, not just times. For instance, "9:58 PM to the nearest ten minutes" is 10:00 PM.

In talking about lengths of time, rather than times of day, it's useful to have these:

(number) **minutes ... time**

This phrase converts numbers into lengths of time. Example:

```
15 minutes
```

Because it's a phrase, not just a notation for writing constants down, the number doesn't have to be given literally:

```
let X be 5;
if the player is in the Slow Room, now X is 10;
let deadline be the time of day plus X minutes;
```

Note that lengths of time can't exceed 1440 minutes.

(number) **hours ... time**

This phrase converts numbers into lengths of time. Example:

```
10 hours
```

Note that lengths of time can't exceed 24 hours.

Start of Chapter 9: Time

Back to §9.7. Telling the time

Onward to §9.9. Comparing and shifting times

§9.9. Comparing and shifting times

Carrying out easy calculations with times is straightforward:

```
The chronometer is in the Clock Chamber. "On one wall is a terribly self-important chronometer showing the time in major world cities. London: [time of day]. Paris: [one hour after the time of day]. Tokyo: [9 hours after the time of day]. Cupertino, California: [7 hours before the time of day]."
```

Here we are using two phrases:

(time) **before** (time) **... time**

This phrase produces a time earlier by the amount given, keeping within the 24 hour clock. Example:

```
7 hours before 5:30 AM
```

produces 10:30 PM.

(time) **after** (time) **... time**

This phrase produces a time later by the amount given, keeping within the 24 hour clock. Example:

```
9 hours after 11 AM
```

produces 8 PM.

Similarly, we have conditions:

if (time) **is before** (time)**:**

This condition is true if the first time occurs earlier in the day than the second. In recognition of the fact that very few stories begin before 4 AM, whereas many run on past midnight, the start of the day is taken to be 4 AM: thus 3:59 AM is after 11:10 PM, but 4:04 AM is before it.

if (time) **is after** (time)**:**

This condition is true if the first time occurs later in the day than the second. In recognition of the fact that very few stories begin before 4 AM, whereas many run on past midnight, the start of the day is taken to be 4 AM: thus 3:59 AM is after 11:10 PM, but 4:04 AM is before it.

⬆ Start of Chapter 9: Time

⬅ Back to §9.8. Approximate times, lengths of time

➡ Onward to §9.10. Calculating times

§9.10. Calculating times

We will occasionally need to perform more complex calculations with time, and in order to do that, we have a way to convert the time of day to numbers. Thus the phrase "the minutes part of ..." takes a time and produces a number from 0 to 59; similarly "the hours part of ..." extracts a number from 0 to 23, using the twenty-four hour clock.

minutes part of (time) **... number**

This phrase converts a time to a number, then takes the result mod 60, which in effect produces the number of minutes after the hours are thrown away. Example:

```
minutes part of 12:41 PM
```

produces 41.

hours part of (time) **... number**

This phrase converts a time to a number, then divides the result by 60, which in effect produces the number of hours after minutes are thrown away. Example:

```
hours part of 8:21 AM
```

produces 8.

To go the other way, we can convert any number to a duration by writing "minutes" or "hours" after it. For instance:

```
The clock error is a number that varies. To thump the mechanism: now the clock error is a random number from -10 to 10.

The broken grandfather clock is in the Chamber. "An erratic grandfather clock seems to say it is [clock error minutes after the time of day]."

When play begins, thump the mechanism. Instead of attacking the broken clock: thump the mechanism; say "You thump the clock, which now reads [clock error minutes after the time of day].".
```

Note that "clock error" is a number, but "clock error minutes" is a time.

Start of Chapter 9: Time

Back to §9.9. Comparing and shifting times

Onward to §9.11. Future events

§9.11. Future events

We often want to arrange for something to happen at some point in the future. Here is yet another timepiece:

```
An egg-timer is in the Chamber. "A plastic egg timer in the shape of a chicken can be
pressed to set it going."

Instead of pushing the egg-timer:
    say "It begins to mark time.";
    the egg-timer clucks in four turns from now.

At the time when the egg-timer clucks:
    say "Cluck! Cluck! Cluck! says the egg-timer."
```

The event here is called "the egg-timer clucks". It only happens if we instruct so, using one of the following phrases:

(rule) **in** (time) **from now**

This phrase causes the given rule to be run at a given time offset from the current time of day. Example:

```
the egg-timer clucks in 18 minutes from now;
```

(rule) **in** (number) **turn**/turns **from now**

This phrase causes the given rule to be run at a given number of turns after the current one. Example:

```
the egg-timer clucks in four turns from now;
```

(rule) **at** (time)

This phrase causes the given rule to be run at a given time of day. Example:

```
the egg-timer clucks at 11:35 AM;
```

If we know in advance what time we want something to happen, we can more simply write:

```
At 4 PM: say "The great bells of the clock tower chime four."
```

(Note that in either case such rules begin with the word "at": they are the only rules allowed to begin with the word "at".)

A small warning: timed events like these only have a chance to occur during the turn sequence, that is, once every turn. In most stories, one turn takes one minute, so there will in due course be a turn happening at exactly (say) 11:35 AM. But if the clock is being advanced faster than this, it's possible that there are turns at (say) 11:32 AM and then not until 11:37 AM. But an event set for 11:35 AM will nevertheless happen -- it will run at the first available turn after that time, which will be 11:37 AM. Events can thus happen up to half an hour late, though Inform cancels them if the elapsed time is greater than that.

The Scenes panel of the Index can be a useful way to see what events have been set.

Start of Chapter 9: Time

Back to §9.10. Calculating times

Onward to §9.12. Actions as conditions

Example 143: ★ **MRE** Hunger that eventually kills the player, and foodstuffs that can delay the inevitable by different amounts of time.

Example 144: ★★ **Totality** To schedule an eclipse of the sun, which involves a number of related events.

Example 145: ★★ **Empire** A train which follows a schedule, stopping at a number of different locations.

Example 146: ★★★ **Hour of the Wren** Allowing the player to make an appointment, which is then kept.

§9.12. Actions as conditions

There are two ways that descriptions of actions can be used as conditions. First, we can simply describe an action, and then the condition will be true if that is what the player is trying to do, and not otherwise:

```
if taking a container, ...
```

This is actually an abbreviation for the longer, some would say preferable form:

```
if we are taking a container, ...
```

Secondly, we can talk about the past as well as the present, which is very useful since interactive fiction often contains situations which are changed by earlier events.

```
Instead of waiting when we have taken the lantern, say "No, your acquisitive nature is
roused now, and simply waiting will no longer do."
```

More on the past tense later follows in the next section: note that "we are taking" has become "we have taken". For the rule to apply, it is not enough that the action "taking the lantern" has been tried: it must have succeeded. Note also that it's enough for any actor in the story to have successfully taken the lantern: it doesn't have to be the player.

⬆ Start of Chapter 9: Time

⬅ Back to §9.11. Future events

➡ Onward to §9.13. The past and perfect tenses

⬇ Example 147: ★ **Night Sky** A room which changes its description depending on whether an object has been examined.

⬇ Example 148: ★★★ **Zero** A box which called "horribly heavy box" after the player has tried to take it the first time.

§9.13. The past and perfect tenses

The remaining sections of this chapter go into more technical ways to think about the progress of the story through time, and can be skipped at a first reading.

Conditions are clauses which require Inform to make a decision: is such-and-such true, or not true? We have already seen conditions attached to rules using "when":

```
Instead of waiting when the Sorting Hat is in the Hall: ...
```

and, as we shall later see, we will often want to write instructions like:

```
if the Sorting Hat is in the Hall, say "Hermione blinks apprehensively."
```

The condition is "the Sorting Hat is in the Hall", and during play this will sometimes be true and sometimes false.

A condition in the form "X is Y" is of course written in the present tense, and refers to the current state of affairs. Three other tenses are allowed. First, the present perfect:

```
if X has been Y ...
```

is true if it has ever been the case that "X is Y" at the start of any turn (or any action). So, for instance,

```
if the gate has been open ...
```

will be valid if and only if the gate has ever been made open by any action (even if it is closed now), or if it started out by being open when play began.

Next is the past tense:

```
if X was Y ...
```

holds if and only if "X is Y" was true at the start of the most recent action. This is convenient when trying to describe what has changed in the course of the action, but sometimes also when making the action itself happen. For instance:

```
if the lantern was switched on, now the lantern is switched off;
if the lantern was switched off, now the lantern is switched on;
```

Completing the set is the past perfect:

```
if X had been Y ...
```

which records whether "X has been Y" was true at the start of the most recent action. All these verbs can of course be negated (though "wasn't" and "hadn't" are disallowed as poor style: we use "was not" and "had not" instead). So for example,

```
if the player had not been in the Ballroom ...
```

is true if the player hadn't visited the Ballroom at the start of the most recent action.

Something we must watch out for is that variables might not have the same values in the past that they have now. As a result, writing conditions such as "if the noun has been open" is a bad idea, because in the past "the noun" very likely referred to something different. It is really only safe to talk in the past tense about definite, fixed things: "if the Great Gates of Kiev have been open" would be fine.

Start of Chapter 9: Time

Back to §9.12. Actions as conditions

Onward to §9.14. How many times?

Example 149: ★ **Tense Boxing** An overview of all the variations of past and present tenses, and how they might be used.

Example 150: ★★ **Bruneseau's Journey** A candle which reacts to lighting and blowing actions differently depending on whether it has already been lit once.

Example 151: ★★ **Elsie** A door that closes automatically one turn after the player opens it.

§9.14. How many times?

There are two further ways to examine the historical record. Given any condition, we can say

```
if (...condition...) for the second time ...
if (...condition...) twice ...
if (...condition...) 2 times...
if (...condition...) two times...
```

(all of which are synonymous: the words once, twice, thrice, one, two, three, four, five, six, seven, eight, nine, ten, first, second, third, fourth, fifth, sixth, seventh, eighth, ninth and tenth all mean what they obviously should). The result is true if the condition holds now and has held for only one previous spell in the past. A condition holding for, say, fifteen consecutive turns without a break counts as only one "time" - so what we mean by "twice" here is that it is true now, was previously false for a while, and was previously true for a while before that, but no more. In effect, then,

```
if the player is in the Ballroom for the third time ...
```

is true if this is the third visit to the Ballroom. We can also say

```
if the player is in the Ballroom for more than the third time ...
```

or similarly "less than", "at least", "at most". It would be more natural, though, to say

```
if the player has been in the Ballroom three times ...
```

The adjective "only" (or equivalently "exactly") can be added to obtain

```
if the player has been in the Ballroom only three times ...
```

To recap, this means there have been exactly three visits to the Ballroom in history, whereas

```
if the player is in the Ballroom for the third time ...
```

means there have been exactly three visits, the third of which is still going on - an important distinction.

Start of Chapter 9: Time

Back to §9.13. The past and perfect tenses

Onward to §9.15. How many turns?

Example 152: ★ **Infiltration** A room whose description changes depending on the number of times the player has visited.

§9.15. How many turns?

So much for "times" - spells in which a condition is true. We can also test the length of time, in turns of play, that something has been true. Thus:

```
if ... for three turns;
```

means that the condition holds now, and held at the start of this turn, at the start of last turn, and at the start of the turn before that. In particular:

```
if the floppy hat has been worn for three turns ...
```

will be false if the hat is not currently worn (even if it has been often in the past) and, on the other hand, will be true if the hat has been worn for twenty turns. Here again we can be more specific. These are synonymous:

```
if the floppy hat is worn for the third turn ...
if the floppy hat has been worn for only 3 turns ...
if the floppy hat has been worn for exactly three turns ...
```

all requiring that the hat wasn't worn four turns ago. As before, "more than", "less than", "at least" and "at most" so forth can also be used - say, "for at least 21 turns".

A warning: we must be careful when writing something like

```
if the noun has been open ...
```

since this tests whether it has ever been true that the noun of the then action was open: not whether the current noun-object has ever been open.

Lastly, note that the beginning of play - when (usually) initial text and a banner is printed, followed by a room description - counts towards these counts. In effect, this is a turn: one in which the player compulsorily performs the looking action, rather than being asked for a command. (By convention it is numbered as turn number 0, and doesn't contribute towards the turn count.)

Start of Chapter 9: Time

Back to §9.14. How many times?

Onward to Chapter 10: Scenes: §10.1. Introduction to scenes

Example 153: ★ **Annoyotron Jr** A child who after a certain period in the car starts asking annoying questions.

Examples from Chapter 9: Time

Start of this chapter

Chapter 10: Scenes

Indexes of the examples

135 ★ **Example Clueless** *RB*

A murderer for the mystery is selected randomly at the beginning of the game.

"When play begins" is the best point to initialize any aspects of the game that are meant to change between playings. For instance, in this scenario, we would randomly select one of the other characters to be guilty of murder:

```
"Clueless"

The murderer is a person that varies.

When play begins:
    now the murderer is a random person who is not the player.

The Billiards Room is a room. Colonel Mustard and Professor Plum are men in
the Billiards Room. Miss Scarlet and Mrs White are women in the Billiards
Room.
```

```
Instead of examining the murderer:
    say "[The noun] certainly looks fiendish!"

Test me with "x mustard / x plum / x scarlet / x white".
```

136

★★★ Example Mutt's Adventure

Awarding points for visiting a room for the first time.

RB

Suppose we want to reward the player the first time he reaches a given room. The "unvisited" attribute is useful for this: unlike such constructions as "going to a room for the first time", it doesn't develop false positives when the player has merely tried to go to the room in question. "Every turn when the player is in a room for the first time" is also unhelpful, because it continues to be true as long as the player is in a room on his first visit there.

```
"Mutt's Adventure"

Use scoring.

Section 1 - Procedure

A room can be scored or unscored.

Carry out going to a unvisited scored room:
    increment the score.

Section 2 - Scenario

The Incan Palace Compound is a room. "After numerous false leads through the jungles of Peru, and an arduous trek along the Amazon, you have arrived, at last, here: at Atagon, the lost city of untold treasure."

The startlingly intricate door is a door. It is inside from Incan Palace Compound and outside from the Treasure Room. "A door carved all over with figures of ancient gods, and protected by an assortment of gears and latches, [if open]stands open[otherwise]blocks progress[end if] towards [the other side of the intricate door]."

The description of the Treasure Room is "To your considerable surprise, the treasure room is stocked with art objects from a vast range of eras and geographical locations: beside the expected pre-Columbian gold there are Cycladic figurines, Chinese Tang-dynasty pottery, purses that might have been stolen from Sutton Hoo. [one of]If the British Museum developed a nasty expectorant cough, this is what you'd find in its hanky.[or][stopping]".

The Treasure Room is scored.

Test me with "in / out / in".
```

137

Example No Place Like Home

RB

Recording a whole table of scores for specific treasures.

Suppose we want to assign scores for a whole range of objects the player might pick up. One systematic way to do this would be with a table of point values for things:

```
"No Place Like Home"

Use scoring.

The Hall of the Gnome King is a room. The emerald cow is a thing in the Hall of
the Gnome King. The ivory chessman is a thing in the Hall of the Gnome King.
The book of incantations is a thing in the Hall of the Gnome King.

Table of Point Values

    item        score
    cow         10
    incantations 4
    chessman    1

Report taking an item listed in the Table of Point Values:
    increase the score by the score entry;
    blank out the whole row.

Test me with "take all".
```

"Blank out the whole row" removes the line from the table, so that each award will occur only once. The player will not be able to earn more and more points by dropping and taking the same item again.

138

Example Big Sky Country

RB

Allowing the player to continue play after a fatal accident, but penalizing him by scattering his possessions around the game map.

Some older games allowed the player to be resurrected after a death, but punished him by distributing his possessions far and wide. Here we emulate that effect.

```
"Big Sky Country"

Use scoring.

When play begins: say "There's a bit of a drive over from Anaconda, Montana,
and then through a couple or three ghost towns, but finally you find what you're
looking for, and strike out on foot..."

Entrance to Devil's Canyon is a room. "You are at the top of a steep road, which
proceeds down into the canyon proper." A sign is in Devil's Canyon. It is fixed in
```

```
place. "An ominous sign has been put up by the local sheriff's office." The
description is "PROCEED AT OWN RISK - NO RESCUES!"

Instead of going down when a random chance of 1 in 3 succeeds:
    say "Whoooops, your footing is not as secure as you thought...";
    end the story.

Dusty Path is below Entrance. "A dusty path, with grey-brown thorny bushes on
either side. Immediately to your right is a sheer drop; far below you can see the
rusting remains of a Model T that some fool tried to drive by here."

Hairpin is below Dusty Path. "A sharp bend in the road, doubling back down
towards the bottom of the canyon. Just north of here there is also a small cavern
of some kind[if the stick pin is in the cavern], which attracts your eye with some
glittery thing[end if]."

The Cavern is north of Hairpin. "Really not much more than a little hollow in the
side of the canyon." In the cavern are a snake and a diamond stick pin. The
snake is an animal. The description of the snake is "You're no expert, but it looks
like a rattler."

Instead of taking the diamond stick pin in the presence of the snake: say "Turns
out the snake is partial to that there pin, and takes exception to your intending to
make off with it."; end the story.
```

In a fuller implementation of this game, we might make it possible to get by the snake, but in this version, it's just going to remain troublesome.

```
Crooked Path is below Hairpin. "You're about two thirds of the way down to the
bottom of the cavern at this point."

At the Spot is below Crooked Path. "This'll be it: a bare patch of ground that
might as well have an X painted right on it."

Rule for supplying a missing noun while digging:
    now noun is the location.

Understand "dig" or "dig hole/here" or "dig in ground/dirt/earth" as digging.
Digging is an action applying to one thing.

Instead of digging at the spot:
    say "You dig and dig, and after a half hour or so, sure enough, you do turn up
a big box of gold! You're going to be richer than God and Bill Gates put
together.";
    increase the score by 5;
    end the story finally.

Instead of digging at the spot when the player does not carry the shovel:
    say "What, without your shovel? That won't work too well."

The player carries a walking stick. The player wears a hat, a whistle, and a
daypack. The daypack contains a mylar blanket, a granola bar, a cellular phone,
a water bottle, a folding shovel, and a photocopied map. The granola bar is
edible. Instead of drinking the water, say "You quench your thirst, for the time
```

```
being." The description of the map is "The map shows the winding path of
Devil's Canyon, with a large X down by the south end. That would be where your
uncle Jesse buried the gold from the train robbery."

The maximum score is 5.

When play ends when the story has not ended finally:
    say "Oh dear, that ought to be fatal! However, if you like I can get you out of
it...

    Shall I? >";
    if the player consents:
        repeat with item running through things had by the player:
            move the item to a random visited room;
        say "A strong wind picks you up and sets you back at [the location], though
perhaps minus a few of your things.";
        resume the story;
        try looking.
```

"If the player consents" is just a convenient way to ask a yes/no question that the player must answer before going on with the game.

139

★★★ Example Witnessed 1

RB

A kind of battery which can be put into different devices, and which will lose power after extended use.

The following example makes fairly ample use of material that we haven't seen yet, but gives some idea of the flexibility of every turn rules. Suppose we want to have a number of electrical devices, all of which may be powered by a set of batteries. The batteries will all need to be discharged as they are used (regardless of what device they happen to be controlling at the moment). So:

```
"Witnessed"

A battery is a kind of thing. A battery has a number called charge. The charge of
a battery is usually 15.

Every turn:
    repeat with hollow running through battery compartments:
        if the hollow is part of a switched on device (called the machine):
            if a battery (called cell) is in the hollow:
                decrement the charge of the cell;
                carry out the warning about failure activity with the machine;
                if the cell is discharged, carry out the putting out activity with the
machine;
            otherwise:
                carry out the putting out activity with the machine.

Warning about failure of something is an activity.
```

```
Rule for warning about failure of a device (called the machine):
    if a random battery compartment which is part of the machine contains a battery (called the power source):
        if the charge of the power source is 2, say "[The machine] is obviously going to go out quite soon."

Putting out something is an activity.

Rule for putting out a device (called the machine):
        say "[The machine] loses power and switches off![line break]";
        silently try switching off the machine.

A battery compartment is a kind of container. A battery compartment is usually closed and openable. One battery compartment is part of every device. Instead of inserting something which is not a battery into a battery compartment, say "Only batteries should go in a battery compartment."
```

And to get rid of annoying messages like "Which would you like to close, the flashlight or the flashlight's battery compartment?" when only the compartment is closable, we might add some understanding instructions:

```
Understand "turn on [device]" as switching on.

Understand "turn off [device]" as switching off.

Understand "open [openable closed thing]" as opening.

Understand "close [openable open thing]" as closing.

Understand "put [something] in [container]" as inserting it into.

Instead of opening a device, try opening a random battery compartment which is part of the noun. Instead of closing a device, try closing a random battery compartment which is part of the noun. Instead of inserting a battery into a device, try inserting the noun into a random battery compartment which is part of the second noun.

Instead of switching on an empty device:
    say "Nothing happens, perhaps because there isn't a charged battery in [the noun]."

Instead of switching on a battery compartment which is part of a device (called the power user), try switching on the power user.

Definition: a device is empty:
    if a random battery compartment which is part of it contains a battery (called the power source):
        if the power source is discharged, yes;
        no;
    yes.

Definition: a battery is discharged if its charge < 1.
```

A light source is a kind of device. Carry out switching on a light source: now the noun is lit. Carry out switching off a light source: now the noun is unlit.

The flashlight is a light source. A D battery is a battery carried by the player.

The cassette recorder is a device. Every turn: if the cassette recorder is switched on, say "The cassette recorder hisses faintly."

Rule for warning about failure of the cassette recorder:
    if a random battery compartment which is part of the cassette recorder contains a battery (called the power source):
        if the charge of the power source is 2, say "The hiss from [the cassette recorder] begins to warble ominously."

The player wears a backpack. The backpack is openable. In the backpack is the flashlight and the cassette recorder.

The description of the cassette recorder is "Useful both for recording your notes and for capturing any odd ghostly sounds you may hear."

The description of the backpack is "An old familiar pack, which you know so well that you can find all its pockets and take things in and out of it in pitch darkness. To avoid it showing up oddly in photographs, it is entirely black, with no shiny or metallic attachments."

The description of the flashlight is "You bought a new one just for this occasion, because you were worried about bringing something too small or light. This is a heavy-duty flashlight with an adjustable-focus beam. The case is made of metal, rather than plastic, and there is a spare light-bulb inside as well. You've put a band of masking tape around the handle and written in your initials in red marker.

There is a piece of red cellophane attached to the business end of the flashlight to keep it from being overly bright."

The red cellophane is part of the flashlight.

Instead of doing something to the red cellophane: say "You need the cellophane on the flashlight so that using it does not completely destroy your night vision."

Thirtieth Street Station is a room. "A huge, high, rectangular room with coffered ceilings, which looks grand but mostly makes you feel lonely and small. There are long benches in rows down the middle of the room, and an information desk with the train times, and a series of ticket windows, none of which matters very much at the moment."

The benches are an enterable supporter. They are scenery in the Station. The information desk is scenery in the Station. Some ticket windows are scenery in the Station. Instead of examining scenery in the Station: say "You're fairly sure that whatever is going on here has nothing to do with [the noun]." Understand "window" as ticket windows.

The mural is fixed in place in Thirtieth Street. "At the north side of the station is a particularly pointless and empty annex to the main room. It is dominated by a huge relief of sorts, and this is what you remember." Understand "metal" or

```
"relief" or "huge" as the mural. The description of the mural is "It is both stylized and confusing, but you think it might be supposed to represent the various tasks and occupations of Philadelphia's population. The portions closer to the ground look as though they have recently been subjected to a light dusting of talcum powder. No unusual prints are evident."

The wind chimes are fixed in place in Thirtieth Street. "Carefully attached to the wall with a piece of duct tape and a hook is a light-weight set of wind chimes. Someone else has been here before you, it seems." The description is "Several of your friends use wind chimes as a sort of ghost alarm, since ghosts sometimes cause very localized movements of air when there is no natural breeze."
```

And this last bit, borrowed from the chapter on Understanding, adds some special instructions to help Inform decide when the player is likely to be referring to a compartment and when he's likely to be referring to the device itself.

```
Does the player mean doing something other than searching to a battery compartment: it is unlikely. [We discourage Inform from choosing a compartment when the player uses just the name of a device or the word 'battery'.]
```

We also need to deal with commands like PUT BATTERY IN FLASHLIGHT, where Inform might construe BATTERY as the D battery, the flashlight's battery compartment, or the cassette recorder's battery compartment -- and might also construe FLASHLIGHT as either the flashlight's battery compartment or the flashlight itself.

```
Does the player mean inserting into a battery compartment:
    if the noun is nothing:
        it is very likely;
    otherwise:
        make no decision.

Does the player mean inserting a battery compartment into: it is very unlikely.

Does the player mean inserting something into a device: it is unlikely.

Does the player mean searching a battery compartment: it is very likely.

Test me with "test first / test second".

Test first with "i / open flashlight compartment / put battery in it / turn on flashlight / take d battery / open cassette compartment / turn on cassette / put battery in cassette compartment / turn on cassette / z / z / z / z".

Test second with "get d battery / put d battery in flashlight compartment / turn on flashlight / z / z / z / z / z / z / turn off flashlight / z / z / turn on flashlight / z".
```

140 ★★★★ **Example** **Text Foosball** RB

A game of foosball which relies heavily on every-turn rules.

Suppose we want a game of foosball in which our opponent acts every turn, but does different things depending on where the ball currently lies. We can put together a sequence of every-turn rules to account for this, as follows:

```
"Text Foosball"

Use scoring.

The Lounge is a room. "The Lounge is appointed with everything necessary to
rest and relaxation: a vending machine, a potted palm, a stack of Entertainment
Weekly issues from 1993, and -- your pride and joy -- a foosball game."

The foosball game is scenery in the Lounge. Understand "table" or "football" or
"foozball" or "fussball" or "soccer" as the foosball game. The game is a
supporter. On the game is a small white ball. The ball can be still, approaching,
receding, or unreachable. The description of the ball is "Currently [small white
ball condition]."

After printing the name of the small white ball, say " ([small white ball
condition])".

When play begins:
    now left hand status line is "You: [score]";
    now right hand status line is "Joey: [Joey's score]".

Some tiny men on sticks are part of the game. Understand "handles" as the tiny
men. The description is "Okay, a couple of the tiny men have had their feet
broken off, and the table surface itself is a bit warped, and the ball resembles a
quail egg in respect of shape and color. This makes for a game of unusual
randomness, but skill is overrated."

Instead of attacking or pulling or pushing the game when the ball is unreachable:
    say "You give the table a good shove, and the ball moves ever-so-slightly.";
    now the ball is still.

Instead of taking the white ball:
    say "You'd forfeit the game if you did that."

Instead of turning the tiny men when the ball is unreachable:
    say "The ball has somehow gotten to a mystical point on the table where it
cannot be reached, no matter what. Close inspection reveals that this point has
been marked in chalk with a tiny X. Not that that does any good."

Instead of turning the tiny men when the ball is approaching:
    if a random chance of 2 in 3 succeeds:
        if a random chance of 1 in 2 succeeds, now the ball is receding;
        otherwise now the ball is still;
        say "[if the ball is still]Thunk. [otherwise]Thwack! [end if]You keep the ball
from reaching its goal! Now it is [small white ball condition].";
    otherwise:
        let Joey score.

To let Joey score:
    now the ball is still;
    now Joey's score is Joey's score + 1;
    say "The ball rolls neatly into your goal, despite your efforts. ";
    if Joey's score < score, say "You put the ball back in the center with a snap. No reason to worry yet; you're still ahead. Joey looks determined, though.";
    otherwise say "After allowing a moment or two for Joey's gloating to pass, you replace it at the center."

Instead of turning the tiny men when a random chance of 1 in 13 succeeds:
    if the ball is unreachable, continue the action;
    now the ball is unreachable;
    say "You hit the ball off-center and it rolls sluggishly into a little dip in the surface of the foosball table. ";
    if Joey's score > 7, say "'You did that on purpose!' Joey exclaims indignantly.";
    otherwise say "You and Joey exchange glances. This is never good."

Instead of turning the tiny men:
    say "You madly rotate the tiny men on sticks! ";
    if a random chance of 1 in 2 succeeds:
        say "Hoorah! You hit the ball!";
        now the ball is receding;
    otherwise:
        say "Somehow you fail to bring your monopodal player into contact with the ball."

Joey is a man in the Lounge. "Joey is hunkered over the foosball handles on his side of the table." Joey can be active or inactive.

Joey's score is a number that varies.

Every turn: now Joey is active.

Every turn when the ball is approaching and Joey is active:
    let total be Joey's score + score;
    if total > 9, make no decision;
    now Joey is inactive;
    let Joey score;
    rule succeeds.

Every turn when the ball is unreachable and Joey is active:
    let total be Joey's score + score;
    if total > 9, make no decision;
    now Joey is inactive;
    say "Joey glares angrily at the stuck ball."

Every turn when the ball is receding and Joey is active:
    let total be Joey's score + score;
    if total > 9, make no decision;
    if the ball is unreachable, make no decision;
    now Joey is inactive;
    if a random chance of 1 in 2 succeeds:
        if a random chance of 1 in 2 succeeds, now the ball is still;
        otherwise now the ball is approaching;
        say "Joey connects with your shot. Now the ball is [small white ball condition]!";
    otherwise:
        now the ball is still;
        say "Joey tries to block, but misses! Back it goes in the center, where it is
[small white ball condition].";
        increment the score.

Every turn when the ball is still and Joey is active:
    let total be Joey's score + score;
    if total > 9, make no decision;
    if the ball is unreachable, make no decision;
    now Joey is inactive;
    if a random chance of 1 in 2 succeeds:
        now the ball is approaching;
        say "Joey hits the ball solidly down towards your goal. Now it is [small white
ball condition].";
    otherwise:
        say "Joey fails to hit the ball in your direction. It remains [small white ball
condition]."

Every turn:
    let total be Joey's score + score;
    if total > 9:
        if Joey's score > score, end the story saying "Rats! Joey wins!";
        if Joey's score < score, end the story finally saying "Victory is yours!";
        if Joey's score is score, end the story saying "A perfect tie."
```

141 ★★ Example IPA

RB

Shops which each have opening and closing hours, so that it is impossible to go in at the wrong times, and the player is kicked out if he overstays his welcome.

Suppose we wanted a game set in a living town, with locations opening and closing at different times of day, and business carrying on as usual. The point might be to force the player to plan his itinerary carefully to hit the right spots at the right times; or we might be writing a more contemplative piece, where part of the enjoyment came from just watching the characters wander around doing their daily business...

```
"IPA"

When play begins: now the right hand status line is "[time of day]".

The time of day is 9:50 AM.

A shop is a kind of room. A shop has a time called the opening hour. The
opening hour of the shop is usually 8 AM. A shop has a time called the closing
hour. A shop usually has closing hour 6 PM.

Check going to a shop (called the target):
    if the time of day is before the opening hour of the target,
        say "[The target] is locked up until [the opening hour of the target]." instead.

Check going to a shop (called the target):
    if the time of day is after the closing hour of the target,
		say "[The target] has been closed since [the closing hour of the target]." instead.

Every turn when the location is a shop:
	let deadline be the closing hour of the location;
	if the deadline is before the time of day:
		let target be a random adjacent room which is not a shop;
		say "You are gently but firmly ushered out, since [the location] is closing.";
		move the player to the target.

The Strip-mall Parking Lot is a room. "Dead Christmas trees are heaped outside the bagel shop. Strips of dirty ice survive along the curb, and in the shadows of the lamp-posts. A wet, almost illegible sheet of algebra homework is plastered to the asphalt.

Pinewood Brewing Supply is at the east end of the lot."

Pinewood Brewing Supply is a shop. It is east of Parking Lot. The opening hour of Pinewood Brewing Supply is 10:00 AM. The closing hour of Pinewood Brewing Supply is 3:30 PM. "Shelves and shelves of malt and hops; large glass carboys, and plastic tubing; empty bottles; bottle-caps; bottle-labeling kits; starters for vinegar, sourdough, root beer.

A sweet malty smell hangs in the air."

Instead of going to Brewing Supply when the time of day is before the opening hour of Brewing Supply for the second time:
	say "You rattle at the door again. 'Hold your horses, for crying out loud,' yells a voice from within."

Noah's Bagels is a shop. It is north of the Parking Lot. The opening hour of Noah's Bagels is 6:00 AM. The closing hour of Noah's Bagels is 11:00 AM. "The selection has been somewhat picked over, leaving you with your choice of Pumpernickel, Asiago, or Everything."

Test me with "e / e / n / z / s / e / z / e / z / z / e".
```

142 ★ **Example** **Situation Room** *RB*

Printing the time of day in 24-hour time, as in military situations.

Though Inform normally prints times in AM/PM terms, it stores the hours and minutes as 24-hour time; so, if we like, we can easily extract that information again thus:

```
"Situation Room"

The Situation Room is a room.

To say (relevant time - a time) as 24h time:
	let H be the hours part of relevant time;
```

```
    let M be the minutes part of relevant time;
    say "[if H is less than 10]0[end if][H][if M is less than 10]0[end if][M]".

When play begins:
    now the time of day is 6:09 PM;
    now the right hand status line is "[time of day as 24h time]".

Test me with "z".
```

---

143 ★ **Example** **MRE** *RB*

Hunger that eventually kills the player, and foodstuffs that can delay the inevitable by different amounts of time.

Many older interactive fiction games required the player to find food and eat on a regular basis in order to avoid death. This effect was often unrealistic (since most people can survive much longer than a few hours without eating) and is often seen as an annoyance. However, for the sake of argument, suppose that we do want to construct a hunger-and-death system.

To make things a little more interesting, we will postulate that different foods are differently filling, so that if the player manages to find something really caloric, he is off the hook on his hunger search for a while.

We will also implement the system so that the player gets messages when he is hungry, then dies a short time later. (The times involved are ludicrously short, but this allows us to see the effects within a simple example. In a real game we would want to allow a considerably longer timer for the hunger to play out.)

First, a little scene-setting:

```
"MRE"

When play begins:
    now the right hand status line is "[time of day]";
    say "The procedure was painless at first: increased strength was the first sign,
followed by a sensation of delayed time, as though everyone around you moved
more slowly. Your ability to dodge and perform feats of agility doubled, then
trebled. You were heralded as a triumph of medicine. They told you there would
be no side effects worth speaking of.

They were wrong."

The Base Camp Larder is a room. "This room has been reinforced after each
incident -- and there have been dozens in the last two months -- so that it now
rivals Fort Knox. Only your new skill and speed enabled you to dodge the motion
sensors, knock out the computerized security system, fool the retinal scanner,
and punch a hole in the steel containment grating. But you're inside now."
```

Now we define our food, and add some special instructions for what happens to our hunger counters when the food is eaten:

```
Food is a kind of thing. Food is usually edible. Food has a time called the satisfaction period. The satisfaction period of a food is usually 5 minutes.

A person can be hungry or replete. The player is hungry.

The Larder contains an apple, a candy bar, and a large plate of pasta. The apple, the candy bar, and the pasta are food. The satisfaction period of the apple is 2 minutes. The satisfaction period of the pasta is 125 minutes.

Check eating something which is not food:
    say "[The noun] might be edible, but it's not what you'd consider really food."

Check eating something when the player is not hungry:
    say "You're not hungry right now."

Carry out eating something:
    now the player is replete;
    hunger resumes in the satisfaction period of the noun from now.
```

The first of those two phrases, "now the player is replete", causes the player to cease to be hungry; the second one sets up a future event in which the hunger sets in again. The length of time until that event depends on how satisfying the specific food is. Now we define that event:

```
At the time when hunger resumes:
    starvation occurs in three minutes from now;
    now the player is hungry.

At the time when starvation occurs:
    if the player is hungry, end the story saying "You have starved".
```

Note "if the player is hungry": it is possible that the starvation event will be set up but the player will eat before it occurs; in that case, we want it not to take effect.

And now, since we really ought to give the player some warning of what is happening to him:

```
Every turn when the player is hungry:
    choose a random row in the Table of Hunger Complaints;
    say "[hunger entry][paragraph break]".

Table of Hunger Complaints
hunger
"Gosh, you're starving."
"It feels as though you haven't eaten in days. Weeks, almost."
"The world seems to slow down and everything becomes sharper and brighter. You are a hunter, a hunter of foodstuffs."
"You find yourself staring at [the random visible thing that is not the player] and wondering how it would taste."

Test me with "eat apple / z / z / z / eat candy bar / z / z / z / z / z / z / z / z / z".
```

144 Example **Totality** RB

To schedule an eclipse of the sun, which involves a number of related events.

First we define the events, and then we create a phrase to schedule them:

```
"Totality"

At the time when the penumbra appears:
    say "The sunlight dies away to an eerie, brownish penumbra."

At the time when the eclipse begins:
    say "The moon passes across the disc of the sun, plunging the world into darkness.";
    now the Chamber is dark.

At the time when the eclipse ends:
    say "The moon passes away from the sun, and a gloomy penumbral light returns.";
    now the Chamber is lighted.

At the time when normal sunlight returns:
    say "The sun shines once more bright in the sky, not to be eclipsed again on this spot for another thirty-seven years."

To schedule an eclipse for (totality - a time):
    the penumbra appears at two minutes before totality;
    the eclipse begins at totality;
    the eclipse ends at three minutes after totality;
    normal sunlight returns at five minutes after totality.
```

Now we make use of the new phrase:

```
When play begins, schedule an eclipse for 3:27 PM.

The Chamber is a room.

The time of day is 3:25 PM.

Test me with "z / z / z / z / z / z / z / z / z".
```

We shall see much more about creating phrases later. Their advantage is that they enable a complicated sequence of operations to be given a meaningful name, and that they can be re-used many times as needed.

145 Example **Empire** RB

A train which follows a schedule, stopping at a number of different locations.

Suppose we want to have a train which, at fixed times, arrives at and leaves stations. It should be possible for the player to get on and off the train when it is stopped, but not while the train is in motion.

```
"Empire"

The Empire Builder Train is a room. The Train has a room called the station. The station of the Train is Seattle.

The description of the Empire Builder is "One of the (relatively) plush long-distance Amtrak trains. You're in a two-story car with toilets and a cafe at one end, not having sprung for a sleeper.

[if the station of the Train is the train]Outside the window there is rapidly-passing countryside.[otherwise]Through the windows you can see the [station of the Train] train station.[end if]"

Instead of exiting when the player is in the Train:
    if the station of the Train is the Train:
        say "The train is not stopped at a station." instead;
    otherwise:
        move the player to the station of the train instead.

Before going outside when the player is in the Train:
    try exiting instead.

Before going inside when the player is in the station of the Train:
    move the player to the Train instead.

Seattle, Edmonds, Everett, Wenatchee, and Spokane are rooms. The description of a room is usually "The scenic train station of [the location][if the location is the station of the train].

The pompously-titled Empire Builder train is pulled up here, soon to continue its journey towards Chicago[end if]."
```

And now our schedule for the train -- somewhat truncated, admittedly, since the full three-day journey from Seattle to Chicago is a bit long even for an ambitious example.

```
At 4:45 PM:
    if the player is in the train or the player is in the station of the train, say "The train pulls out of [the station of the Train]!";
    now the station of the Train is the Train.

At 5:10 PM:
    now the station of the Train is Edmonds;
    if the player is in the train or the player is in the station of the train, say "The train pulls into Edmonds and comes to a stop."

At 5:17 PM:
    if the player is in the train or the player is in the station of the train, say "The train pulls out of [the station of the Train], running north along the shore towards
```

```
Everett.";
    now the station of the Train is the Train.

At 5:39 PM:
    now the station of the Train is Everett;
    if the player is in the train or the player is in the station of the train, say "The train arrives in scenic Everett, WA: the last stop before it turns east and heads over the mountains."

At 5:44 PM:
    if the player is in the train or the player is in the station of the train, say "The train pulls out of [the station of the Train] and turns east.";
    now the station of the Train is the Train.

At 8:39 PM:
    if the player is in the train or the player is in the station of the train, say "In darkness the train rolls into Wenatchee; which is just fine, considering that there is nothing to see here at all.";
    now the station of the Train is Wenatchee.

At 8:44 PM:
    if the player is in the train or the player is in the station of the train, say "The train pulls out of [the station of the Train] and continues east through the darkness towards Spokane.";
    now the station of the Train is the Train.
```

Playing this out would of course require near inhuman patience. Let's set things up so that the player at least doesn't have to wait too long for his first departure:

```
The time of day is 4:43 PM.
```

...and provide fair warning of how slowly time is elapsing.

```
When play begins:
    now the right hand status line is "[time of day]".

Test me with "out / in / z/ z / z / out / z / z / z / z / z / z / z / z / z / z / z / z / z / z / z / z / z / z / z / z / z / z / z / out / in / out / z / z".

Test more with "out / z/ z / z / z / z / z / z / z / z / z / z / z / z / z / z / z / z / z / z / z / z / z / z / z / z / z / z".
```

146 ★★★ Example Hour of the Wren

RB

Allowing the player to make an appointment, which is then kept.

Here we allow the player to set the time at which some future event is going to happen, rather than letting the game decide. We'll need to borrow the syntax for defining new actions from a later chapter:

```
"Hour of the Wren"
```

```
When play begins:
    say "You more or less stumble across them in Central Park: a disparate group
of people, all of different ages, sitting in a circle. They aren't talking to one
another -- in fact, they seem to be trying very hard to ignore one another, like
people in the waiting room of an especially embarrassing kind of doctor. You are
about to go around when a woman in a grey pressed suit comes up to you. Her
suit-skirt is trimmed in lavender cord, and she looks as though she might have
been extremely sharp-dressed in 1944. She hands you a card."

The Circle in the Grass is a room. "No one is looking at you, except for the
secretary, if that is what she is."

The player carries a card. The description of the card is "Typed: 'Active astrology
- dislike your fortunes? change your stars! - make an appointment now - hour of
the wren STILL AVAILABLE.'".

The time of day is 1:55 PM.

Understand "pick [time]" or "choose [time]" or "make appointment for [time]" or
"make an appointment for [time]" as making an appointment for. Making an
appointment for is an action applying to one time.

Carry out making an appointment for:
    say "Fate cannot be commanded more than once."

Instead of making an appointment for the time understood for the first time:
    say "You settle on [the time understood] for your appointment. The woman
makes a note of it in an appointment book, which she carries in a brown paper
bag. 'Excellent choice, ma'am,' she says in a low, urgent voice. 'You'll be very
satisfied.'";
    stars shift at the time understood.

Understand "hour of the wren" as 2:00 PM.

At the time when stars shift:
    end the story saying "insert cataclysm here".

Test me with "x card / make appointment for hour of the wren / z / z / z / z".
```

147 ★ **Example** **Night Sky** *RB*

A room which changes its description depending on whether an object has been examined.

Sometimes a nice effect is to change the way things are described depending on the information the player has gained in the course of play. We could for instance write this:

```
"Night Sky"
```

```
The Planetarium is a room. "[if we have examined the sinister message]A dark
room where it seems something is about to jump out at you![otherwise]A tranquil
dark room with a ceilingful of stars.[end if]"

The sinister message is a thing in the Planetarium. "A message is taped to the
wall." The description is "'BEWARE.'"

Test me with "look / x message / look".
```

On the other hand, beware that this would not work as desired:

```
"Night Sky"

The Planetarium is a room. "[if we have listened to the sinister message]A dark
room where it seems something is about to jump out at you![otherwise]A tranquil
dark room with a ceilingful of stars.[end if]"

The sinister message is a thing in the Planetarium. "A message plays very softly,
so that you would have to listen to hear it." Instead of doing anything other than
listening to the message: say "It's only a sound, after all.". Instead of listening to
the sinister message: say "A voice whispers, 'BEWARE'."

Test me with "listen to message / look".
```

The reason is that our Instead rule has pre-empted normal listening, so Inform considers that we have never successfully heard the message. The moral here is that "if we have..." is useful for tracking events that otherwise proceeded completely normally (picking up ordinary objects, examining things); if we have used instead to make some change, we will have to use a different approach to record that the event did occur as scheduled.

148

★★★ Example Zero

A box which called "horribly heavy box" after the player has tried to take it the first time.

RB

The names of objects might want to change also:

```
"Zero"

Weight is a kind of value. The weights are light, manageable, and horribly heavy.
Everything has a weight.

A thing is usually manageable.
```

This does require a rule for an activity (see the chapter on Activities), but it's a fairly straightforward one:

```
Before printing the name of a horribly heavy thing (called weighty object):
    if we have taken the weighty object, say "[weight] ".
```

```
The Nearly Empty Living Room is a room. "Nearly everything is out, all of Helen's possessions and most of yours." A man called Mr Zero is in the Nearly Empty Living Room. "Mr Zero, despite being heavily paid to assist in this operation, is giving you a look that clearly conveys his lack of interest in budging even one more item." The description of Mr Zero is "Many muscles, no hair."

The book box is a horribly heavy thing in the Living Room. The clothing box is a manageable thing in the Living Room. A broom, a dustpan, some packing tape, and a discarded newspaper are light things in the Living Room.

After taking a horribly heavy thing:
    say "Taken. (Oof.)"

Test me with "get clothing / look / get book box / look / i / drop book box / look".
```

149

Example Tense Boxing

RB

An overview of all the variations of past and present tenses, and how they might be used.

Here we have a box that prints out its current state and its history each time we open and close it:

```
"Tense Boxing"

The Temporal Prism is a room. "A room of angled mirrors, in whose surfaces you can see what is now; what just was; what has always been. A final mirror is broken and its frame gapes blackly."

The mysterious box is in the Temporal Prism. It is an openable closed container.

To assess the box:
    if the box was not open, say "The box was not open.";
    if the box was open, say "The box was open.";
    if the box had not been open, say "The box had not been open.";
    if the box had been open, say "The box had been open.";
    if the box is not open, say "The box is not open.";
    if the box is open, say "The box is open.";
    if the box has not been open, say "The box has not been open.";
    if the box has been open, say "The box has been open."

Before opening the mysterious box:
    say "You are about to open the box.";
    assess the box.

Before closing the mysterious box:
    say "You are about to close the box.";
    assess the box.

After opening the mysterious box:
    say "You now open the box.";
    assess the box.
```

```
After closing the mysterious box:
    say "You now close the box.";
    assess the box.
```

Note that "was..." and "was not..." and so on may describe conditions more complicated than simple properties: we could equally well ask "if the box has been in the sack", "if the box had been carried by the player", and so on.

The past ("if the box was...") and past perfect ("if the box had been...") are especially useful for cases where we want to report on an action after the state of the item has changed; so, for instance:

```
After taking the mysterious box:
    if the box had not been carried by the player, say "You lift the mysterious box for the first time.";
    if the box had been carried by the player, say "You again pick up the mysterious box."

Test me with "open box / close box / open box / take box / drop box / take box".
```

This is in many respects similar to a rule beginning "After taking the mysterious box for the first time...", but it is superior in most circumstances, for two reasons.

First, it will respond correctly even if the player has somehow carried the box before without taking it explicitly: for instance, if another character gave him the box, if the box were moved into his inventory as a result of another action, or if the player carried the box at the start of play. Inform begins its reckoning of time when the game begins, so if the box is defined as being open at the outset, "if the box has been open" will always be true.

Second, "after taking... for the first time" fires only the first time the player attempts to take something. If the player tried to take the box, failed, and then tried again later, the "for the first time..." rule would not fire; our "if the box has not been carried..." rule would.

---

150 ★★ **Example** **Bruneseau's Journey** *RB*

A candle which reacts to lighting and blowing actions differently depending on whether it has already been lit once.

```
"Sire," said the Minister of the Interior to Napoleon, "yesterday I saw the most intrepid man in your Empire." - "What man is that?" said the Emperor brusquely, "and what has he done?" - "He wants to do something, Sire." - "What is it?" - "To visit the sewers of Paris."

This man existed and his name was Bruneseau.

- Victor Hugo, Les Miserables
```

Let's say that our intrepid explorer has a candle that can be lit and blown out again, and should accordingly appear unlit, burning, or partly burnt:

```
"Bruneseau's Journey"

The Sewer Beneath St Denis is a room. "A narrow, stone-lined passageway, with
only a little ledge to walk above the level of the refuse that flows down towards
the Seine."

The candle is carried by the player. The description of the candle is "A candle, [if
the candle has been lit]partially burnt[otherwise]still in pristine condition with
untouched wick[end if]."

Instead of examining the lit candle, say "It burns with a pure heart."

The block burning rule is not listed in the check burning rules.

Instead of burning the lit candle:
    say "The candle is already lit."

Check burning:
    if the noun is not the candle, say "[The noun] cannot profitably be set on fire."

Carry out burning the candle:
    now the candle is lit.

Report burning:
    if the candle had been lit, say "You relight the candle.";
    otherwise say "You light the candle for the first time.".

Understand "blow out [something]" as blowing out. Understand "blow
[something] out" as blowing out. Blowing out is an action applying to one thing.

Carry out blowing out the candle:
    now the candle is unlit.

Report blowing out:
    if the noun is the candle and the candle was lit, say "You blow out [the noun].";
    otherwise say "You blow on [the noun], to little effect."

Test me with "x candle / light candle / x candle / blow out candle / x candle".
```

We must be careful: "if the noun was lit" would throw errors because past-tense rules can only be applied to specific items, not to variables that could be anything.

151 ★★ **Example Elsie** *RB*

A door that closes automatically one turn after the player opens it.

Suppose we want to create an automated door of the sort that closes when it isn't in use. A convenient way is to write a rule that fires "every turn when the sliding door was open". This will be true only if the door was open at the beginning of the turn: if the player just opened it this turn, it stays open, and if it was already closed, it stays closed. Thus:

```
"Elsie"

The axis-ward is a direction. The opposite of axis-ward is hub-ward. Understand "aw" or "axisward" as axis-ward.
The hub-ward is a direction. The opposite of hub-ward is axis-ward. Understand "hw" or "hubward" as hub-ward.

The Ship's Bridge is a room.

The sliding door is a door. It is axis-ward from Bridge and hub-ward from C Deck. The initial appearance is "There is a door in this wall[if closed] -- or at least, the potential for a door, since currently it is sealed, distinguishable from the rest of the wall only by the warning stripes on its surface[end if]."

Every turn when the sliding door was open:
    now the sliding door is closed;
    if the player can see the sliding door:
        say "The sliding door slips back into place, sealing the wall as though there had never been a breach."

After opening the sliding door:
    say "You press the appropriate buttons, and a section of wall slides away."

Test me with "open door / look / enter door / z".
```

152 ★ Example **Infiltration** RB

A room whose description changes depending on the number of times the player has visited.

Suppose we have a location that makes the player uncomfortable, and we want its description to change slightly each time he goes there, to reflect his increasing unease. We also want the door to that room to show whether he is going there for the first time, or whether this is a repeat visit.

We start with an ordinary room:

```
"Infiltration"

The Wasteland is a room. "In its more distant reaches, the Wasteland has a kind of austere beauty, but here beside the Secure Zone it is the worst of all possible worlds. Barrels of toxins are stacked the regulation hundred and fifty feet out; more traditional garbage has simply been flung over the wall, and this category includes one or two corpses roughly and inadequately disguised by black plastic bags. The wall itself has become a canvas for outcasts and exiles, and is covered with obscene paintings, lewd remarks about the inhabitants of the Secure Zone, and a few maudlin epitaphs."
```

Now the door, which will change from saying "leads inside..." to "leads back inside..." when this becomes appropriate:

```
The portal is a door. It is inside from the Wasteland and outside from the Secure Zone. "[if the player is in the Wasteland]To the west, a[otherwise]A[end if] portal in the cinder-block and barbed wire wall leads[if the player is in the Wasteland and the Zone is visited] back[end if] [if the player is in the Wasteland]inside[otherwise]outside[end if]."
```

Here we haven't used any conditions that we didn't know about in previous sections: the portal line only reflects whether the Zone has been visited never or visited once. But the Secure Zone itself makes use of the number of times visited:

```
The Secure Zone has the description "[if the player is in the Zone for the second time]Re-entering the Zone has not made you any more comfortable inside. [end if]Despite your carefully-chosen outfit and the walk you have been practicing, you are sure those inside can tell you don't belong. Not that very many people are visible here[if the player is in the Zone for more than the second time] -- the place seems more desolate than ever[end if]."

Instead of going west in the Wasteland, try going inside. Instead of going east in the Secure Zone, try going outside.
```

And finally, to be sure that the player does see our fancy changing descriptions:

```
Use full-length room descriptions.

Test me with "look / open portal / w / look / e / look / w / e / w".
```

Notice that the description of the Secure Zone changes from visit to visit, but that looking repeatedly during a single visit changes nothing.

153

Example Annoyotron Jr

RB

A child who after a certain period in the car starts asking annoying questions.

```
"Annoyotron Jr"

The Minivan is a room. The Open Road is outside from the Minivan. Pete is a man in the Minivan. "Pete [if the player has been in the Minivan for 3 turns]is starting to look bored[otherwise]is playing with his travel activity book[end if]."

Every turn:
    if the player has been in the Minivan for 5 turns, say "'Are we there [if saying no]now?'[otherwise]yet?' asks Pete.[end if]"

Instead of saying no:
    say "'Oh,' says Pete. There is a blessed, momentary silence."
```

```
Instead of going to the Open Road:
    say "You leap to your death.";
    end the story.

Test me with "z / z / look / g / g / g / no / z / z / z / no / z / out".
```

Chapter 10: Scenes

↖ Contents of *Writing with Inform*
← Chapter 9: Time
→ Chapter 11: Phrases
↘ Indexes of the examples

§10.1. Introduction to scenes

As we have seen, Inform divides up space into individual places called "rooms", and allows us to group rooms together into "regions" if we find that convenient. And Inform also divides time up, into individual turns. These too we can group together: the equivalent of a region is a "scene".

To put this another way, if we think of the interactive fiction as a stage play, then up to now it has simply contained endless dialogue and stage directions - there has been no convenient way to divide up its running time into dramatic episodes, in the same way that a playwright might make Act II take place in the same drawing-room as Act I, but (let us say) six months later, after many things have changed. The script contains cues for one scene to end and another to begin: when those cues are reached, the stage hands rearrange props, actors reposition themselves and so on.

Inform also allows us to create scenes, with cues for them to start and end, and some stage machinery (so to speak) making it easy to move the action on. But interactive fiction is *interactive*, so the metaphor of the theatre only goes so far. We can have several different scenes going on at once - perhaps with the relevant events taking place in different rooms, which the player is free to walk between. And the player may make a choice which changes the story-line, causing scenes to happen which otherwise would not have happened, and so on. Scenes can even be "recurring", that is, can repeat themselves.

So organising the story-line into scenes is not simply a matter of making a list (Scene 1, then Scene 2, then Scene 3, *finis*). It is more like a chart in which one scene can lead in several possible ways to others - a sort of map of time, which as we shall see Inform displays in its "Scenes" index.

↑ Start of Chapter 10: Scenes
← Back to Chapter 9: Time: §9.15. How many turns?
→ Onward to §10.2. Creating a scene

§10.2. Creating a scene

As usual, we only need to say that something is a scene to make it so:

```
Train Stop is a scene.
```

We conventionally write scene names with capital letters, as this demonstrates.

This works, and shows up in the "Scenes" index, but does nothing. We have given no instructions on when it begins - no cue, in stage-play terms - so it never will begin, and even if it did, nobody would notice since it does nothing. First, to give it a beginning:

```
Train Stop begins when the player is in the Station for the third turn.
```

In theory any condition can be used to cue the scene - here, it's "the player is in the Station for the third turn" - but it's wise to look for a state of affairs which will last at least a brief time, because scene changes only happen at the start and end of turns. (Something like "...when examining the timetable" may be true only for a part of the middle of a turn, and so go unnoticed.)

Every scene has two rulebooks attached, one at each end, so to speak. These look very like "when play begins" and "when play ends", and work in the same way. Thus:

```
When Train Stop begins:
    now the Flying Scotsman is in the Station;
    say "The Flying Scotsman pulls up at the platform, to a billow of steam and hammering."

When Train Stop ends:
    now the Flying Scotsman is nowhere;
    if the player is in the Station, say "The Flying Scotsman inches away, with a squeal of released brakes, gathering speed invincibly until it disappears around the hill. All is abruptly still once more."
```

Thus when the scene begins, our imaginary stage-hands wheel in a steam train; when it ends, they get rid of it again. Note that we know where the player will be at the start of the scene, but by the end he may have wandered off across the fields, so we must be careful not to report something he might not be in a position to see.

When Train Stop begins, we printed some text, but we did this by hand. We didn't need to, because Inform automatically prints out the description of a scene (if it has one) when the scene begins. Scenes can have properties, just like objects, and in particular they have the "description" property. For example, we could write:

```
Arrival is a scene. "There's a flourish of trumpets."
```

which saves us the trouble of writing the rule:

```
When Arrival begins: say "There's a flourish of trumpets."
```

We can also write rules like this which apply to a whole variety of scenes at once. For instance:

```
A scene can be bright or dim. A scene is usually dim. Dawn is a bright scene.

When a scene which is bright ends: say "So passes the bright [scene being changed]."
```

Here, instead of naming a scene ("Train Stop"), we've given a description ("a scene which is bright"). When a scene begins, these general rules come before those which name the scene exactly; when it ends, the reverse is true.

⬆ Start of Chapter 10: Scenes

⬅ Back to §10.1. Introduction to scenes

➡ Onward to §10.3. Using the Scene index

⬇ Example 154: ★ **Pine 1** Pine: Using a scene to watch for the solution of a puzzle, however arrived-at by the player.

⬇ Example 155: ★★★ **Entrapment** A scene in which the player is allowed to explore as much as he likes, but another character strolls in as soon as he has gotten himself into an awkward or embarrassing situation.

§10.3. Using the Scene index

But when we test the previous section's example, we find that after a brief wait, the train pulls up: but it never goes away again. We have given instructions on how the scene ends, but not when it ends, and as a result the scene goes on forever once started.

Even with simple story-lines, and this one could hardly be simpler, it is surprisingly easy to overlook something so that the whole story-line is derailed.

The Scenes page of the index is intended to help with this. The Plot section shows all of the scenes and how they are to begin, along with a key to the symbols used on it. One scene always included is "Entire Game", a special scene which, as its name implies, is always being played out. But if we look at the Scene index for the previous example, we will also see our Train Stop scene, and find that it is marked with the red warning symbol for "never ends". Let us fix this:

```
Train Stop ends when the time since Train Stop began is 3 minutes.
```

Note the useful value "time since Train Stop began":

time since (scene) **began ... time**

This phrase produces the time since the named scene began, which only makes sense, of course, if it has indeed begun. Example:

```
time since Entire Game began
```

time since (scene) **ended ... time**

This phrase produces the time since the named scene ended, which only makes sense, of course, if it has indeed ended. Example:

```
time since Formal Dinner ended
```

The actual times, in case they are needed, can be obtained with:

time when (scene) **began ... time**

This phrase produces the time (i.e., the value of the "time of day" variable) at the moment when the given scene began.

time when (scene) **ended ... time**

This phrase produces the time (i.e., the value of the "time of day" variable) at the moment when the given scene ended.

The testing command SCENES causes Inform to monitor the beginning and ending of scenes, and report on them. For instance:

```
>ask monk about lodging
"Welcome a poor traveler for the night?" you ask, rubbing your fingers together to
restore a little feeling.

The monk looks you up and down for a moment and you sense his reaction hanging in
the balance; then he slaps you on the back, hard enough to drive the air from your
lungs. "In."

[Scene 'Greeting' ends]
The monk takes your elbow and pushes you imperiously toward dinner.
[Scene 'Banquet' begins]
```

⬆ Start of Chapter 10: Scenes

⬅ Back to §10.2. Creating a scene

➡ Onward to §10.4. During scenes

⬇ Example 156: ★ **Age of Steam** The railway-station examples so far put together into a short game called "Age of Steam".

§10.4. During scenes

Scenes are not only useful for changing the setting, by moving items or people around and providing a little narration. We can also make the rules different in one scene from another. For instance, at a sleepy country halt there is no reason why one should not walk across the tracks: but if there is a train in the way, that would be impossible.

```
Before going north during the Train Stop, say "The train blocks your way." instead.
```

Any rule can have the clause "during ..." attached, provided that clause goes at the end and either explicitly names a scene, or gives a description of which scenes would match. This is especially useful with "every turn":

```
Every turn during the Train Stop, say "Water is sluiced out of the tank and into the
engine."
```

We can test whether a scene is happening with the adjective "happening":

```
if Train Stop is happening, ...
```

if (scene) **has happened:**

This condition is true if the given scene has both begun and ended.

if (scene) **has not happened:**

This condition is true if the given scene has not ended (or never started).

if (scene) **has ended:**

This condition is true if the given scene ended at least once.

if (scene) **has not ended:**

This condition is true if the given scene has never ended.

We need to be a bit careful: it's possible to set things up so that the Train Stop scene will play out more than once, so "Train Stop is happening" and "Train Stop has happened" might both be true at once.

The kind of value "scene" is one which is allowed to have properties - it has a tick in the "properties" column in the chart in the Kinds index - and this can be very useful in describing scenes. For instance, we could write:

```
A scene can be thrilling or dull. Train Stop is dull.
A scene has a text called cue speech. The cue speech of Train Stop is "All aboard!".
```

Inform has the adjectives "recurring", "non-recurring" and "happening" all built in to describe scenes, and the above would add "thrilling" and "dull". Moreover, the "during" clause of a rule can give a description of a scene as easily as a specific scene name. For instance:

```
Before going north during a dull non-recurring scene, ...
```

Start of Chapter 10: Scenes

Back to §10.3. Using the Scene index

Onward to §10.5. Linking scenes together

Example 157: ★ **Full Moon** Random atmospheric events which last the duration of a scene.

Example 158: ★★ **Space Patrol - Stranded on Jupiter!** We'll be back in just a moment, with more exciting adventures of the... Space Patrol!

Example 159: ★★★ **Bowler Hats and Baby Geese** Creating a category of scenes that restrict the player's behavior.

Example 160: ★★★ **Day One** A scene which plays through a series of events in order, then ends when the list of events is exhausted.

§10.5. Linking scenes together

Let us suppose that somebody gets off the train, after all, so that a second scene follows on.

```
Brief Encounter is a scene. Brief Encounter begins when Train Stop ends.
```

The effect of this is that they occur in sequence. If we add a third to the chain of scenes:

```
Village Exploration is a scene. Village Exploration begins when Brief Encounter ends.
```

...we find another chance to fool ourselves: if we check the Scenes index again, we can see the linkages between these scenes, but we also see that Brief Encounter never ends (despite its name). All we have said is that another scene begins where Brief Encounter leaves off, but it never does, so this is moot.

↑ Start of Chapter 10: Scenes

← Back to §10.4. During scenes

→ Onward to §10.6. More general linkages

↓ Example 161: ★★★ **Pine 2** Pine: Adding a conversation with the princess, in which a basic set of facts must be covered before the scene is allowed to end.

§10.6. More general linkages

We are allowed to link the beginning or end of any scene to the beginning or end of any other scene. So, for instance:

```
Luggage Trouble is a scene. Luggage Trouble begins when Brief Encounter begins.
```

Thus the two scenes run concurrently, at least for a while. We can also add that:

```
Luggage Trouble ends when Brief Encounter ends.
```

This can be useful when a large, complicated scene really contains several smaller sub-scenes.

A special exceptional case is that we can have any scene or scenes starting right at the outset:

```
Railway Meeting is a scene. Railway Meeting begins when play begins.
```

When play ends, of course, all scenes end, so there is no need to say that.

↑ Start of Chapter 10: Scenes

← Back to §10.5. Linking scenes together

→ Onward to §10.7. Multiple beginnings and repeats

↓ Example 162: ★ **The Prague Job** Scenes used to provide pacing while the player goes through his possessions.

↓ Example 163: ★★★ **Entrevaux** Organizing the game by scenes, where each scene has a location and prop lists so that it can be set up automatically.

§10.7. Multiple beginnings and repeats

It is quite allowed for a scene to be linked to several other scenes, and this is useful if several alternate strands of plot are being brought together in a common resolution scene:

```
Bittersweet Ending begins when Stranger's Rejection ends.
Bittersweet Ending begins when Stranger's Acceptance ends.
```

and we can also have the same scene beginning when a condition holds. In general, it will begin the first time it gets any chance to do so.

All scenes are ordinarily set up so that they can happen only once. But sometimes we want them to repeat. Suppose the train calls not once only, but every twenty minutes. We could set this up with two scenes linked back to back like so:

```
Train Stop is a recurring scene. Train Wait is a recurring scene.
Train Wait begins when play begins.
Train Stop begins when Train Wait ends.
Train Wait begins when Train Stop ends.
```

The difference here is that these scenes have been declared as "recurring". In all other respects they are the same as any other scene.

⬆ Start of Chapter 10: Scenes

⬅ Back to §10.6. More general linkages

➡ Onward to §10.8. Multiple endings

⬇ Example 164: ★ **Night and Day** Cycling through a sequence of scenes to represent day and night following one another during a game.

⬇ Example 165: ★★★★ **Pine 3** Pine: Allowing the player to visit aspects of the past in memory and describe these events to the princess, as a break from the marriage-proposal scene.

§10.8. Multiple endings

Interactive fictions vary considerably in the extent to which the player is allowed to influence the story-line. Sometimes the appearance of making choices is wholly bogus, as only one possible line is given more than lip service. But in other works, the player can radically change the outcome, and whole rafts of plot differ between one person's experience and another's. Inform allows multiple endings to its scenes to make this kind of branching story-line easy to achieve.

Any scene can have up to 31 alternate endings, differentiated by name (unless the Z-machine format has been selected on the Settings panel, in which case, 15). These alternates are created as and when conditions are set for them:

```
Brief Encounter ends happily when ...
Brief Encounter ends wisely but sadly when ...
```

"Ends happily" and "ends wisely but sadly" behave just like "ends". We can have rules "When Brief Encounter ends happily, ..." and so forth, in addition to rules "When Brief Encounter ends, ..." - if a rule doesn't specify any particular ending, it applies to all of them.

We can also link rules together from these branches, so

```
Stranger's Acceptance begins when Brief Encounter ends happily.
Stranger's Rejection begins when Brief Encounter ends wisely but sadly.
```

With this set-up and that of the previous section, there are now two possible paths through the story:

```
(i) Train Stop - Brief Encounter - Stranger's Acceptance - Bittersweet Ending
(ii) Train Stop - Brief Encounter - Stranger's Rejection - Bittersweet Ending
```

We might later need to know which of these paths has been taken, and to help with this Inform provides conditions like so:

```
if Brief Encounter ended happily ...
if Brief Encounter did not end happily ...
if Brief Encounter ended wisely but sadly ...
if Brief Encounter did not end wisely but sadly ...
```

(For a scene which repeats, note that these conditions apply only to the most recent repetition: and that such conditions are always false if the scene is currently going on. "Brief Encounter did not end happily" will be true only when the scene has finished but in a different way.)

Start of Chapter 10: Scenes

Back to §10.7. Multiple beginnings and repeats

Onward to §10.9. Why are scenes designed this way?

Example 166: ★★★ **Panache** Replacing the score with a plot summary that records the events of the plot, scene by scene.

Example 167: ★★★ **Pine 4** Pine: Adding a flashback scene that, instead of repeating endlessly, repeats only until the Princess has understood the point.

§10.9. Why are scenes designed this way?

In the part it plays in stories, time is like space. The endings of a scene (along with its beginning) are like the map connections leading out of a room. The Scenes index keeps track of the "map of time" through which these possible story-lines traverse. Some works of IF will have immensely complicated story-lines in only a few rooms, some will have no scenes at all despite a sprawling geography. The Scenes and World index tabs, side by side, show both kinds of map.

Just as Inform uses a simple but practical design for the boundaries between rooms (map connections and doors, that is), it also simplifies transitions between scenes. Scenes are based on states of things: we give circumstances for them to begin or end. There is no phrase with the power to say "make Act II begin right now", so perhaps it is worth explaining why not. The state-based approach was chosen because:

* it guarantees that each action falls entirely inside, or entirely outside, of any given scene (and therefore that "during..." clauses in the conditions for a rule are not affected by rule ordering);

* it ensures that scene changes occur outside actions, like every turn rules;

* it promotes a style of writing which makes it clearer to the reader of the source text when a scene begins and ends, and what conditions are guaranteed to be true during it;

* it makes it possible for the Scenes index page to show this information in a communicative way.

Settings in IF where one revisits the same location but at a different time, or after a dramatic change, have historically been difficult to test properly and prone to mistakes. (The classic example would be where a character killed during Act I reappears unharmed in Act II.) The design of scenes is an attempt to encourage a style of writing which minimises the risk of these accidents.

Since scenes are, in the end, only a convenient way to organise rules, and do nothing that cannot be done by other means, this simplified system of scene changing does not really restrict us.

Start of Chapter 10: Scenes

Back to §10.8. Multiple endings

Onward to Chapter 11: Phrases: §11.1. What are phrases?

Example 168: ★★★★ **Cheese-makers** Scenes used to control the way a character reacts to conversation and comments, using a TALK TO command.

Examples from Chapter 10: Scenes

Start of this chapter

Chapter 11: Phrases

Indexes of the examples

154 ★ **Example Pine 1** RB

Pine: Using a scene to watch for the solution of a puzzle, however arrived-at by the player.

Because scene rules are checked every turn, they can be useful for designing puzzles which have multiple solutions. Instead of deciding the puzzle is "solved" when the player does a certain action, we set up a scene that checks to see whether the player has achieved a certain outcome -- however he accomplished it.

For instance, in this scenario, we're waiting for Sleeping Beauty to wake up, and it doesn't much matter how...

```
"Pine"

A person can be asleep or awake. A person can be active or passive.

The Spinning Tower is a room. "A remote corner of the old castle, reserved for spinning and weaving tasks."

Sleeping Beauty is an asleep woman in the Spinning Tower. "[if asleep]Sleeping Beauty lies here, oblivious to your presence[otherwise]Sleeping Beauty stands
```

```
beside you, looking a little confused[end if]." The description is "She is even
more magnificent than the rumors suggested." Understand "woman" or "girl" or
"princess" or "lady" as Sleeping Beauty.

Discovery is a scene. Discovery begins when play begins. Discovery ends when
Sleeping Beauty is awake. Marriage Proposal is a scene. Marriage Proposal
begins when Discovery ends.

When Discovery ends: say "Throughout the palace you can hear the other
sounds of stirring and movement as the spell of centuries is broken."

Instead of waking an awake person: say "Redundant."

Instead of waking an asleep person: say "Yes, but how?"

Instead of attacking an asleep person:
    now the noun is awake;
    say "[The noun] sits bolt upright. 'Hey! Ow!' So much for that true love's kiss
nonsense."

Instead of kissing an asleep person:
    now the noun is awake;
    say "[The noun] slowly stirs to wakefulness!"

Instead of throwing water at an asleep person:
    now the second noun is awake;
    now the noun is nowhere;
    say "You pour out [the noun] on [the second noun].

[The second noun] wakes, shuddering. 'Agh! I had a terrible dream about
drowning and then-- Hey!'"

The player carries a jug of water. Understand "pour [something] on [something]"
or "splash [something] at/on [something]" as throwing it at.

Test me with "x beauty / wake beauty / pour water on beauty".
```

155 ★★★ **Example Entrapment** RB

A scene in which the player is allowed to explore as much as he likes, but another character strolls in as soon as he has gotten himself into an awkward or embarrassing situation.

The power of scenes lies in their ability to watch for general conditions and move the narrative along whenever these are fulfilled. Instead of waiting for the player to do one specific thing, the game waits for the world to be in a certain condition, before moving to the next stage of the plot.

For instance, suppose we have a story in which the player has been captured for doing something inappropriate at court and is brought in to await a meeting with a palace official. We want to give the player a few minutes to stew, and we want the scene to end with him doing something mildly peculiar or embarrassing, and the official catching him in the act. So we tempt him into trying any of a number of different kooky activities, and just wait until he falls into the trap...

```
"Entrapment"

Waiting Suite is a room. "You find yourself in a narrow room, more cozy than is really comfortable, with dark paneling on all the walls. Underfoot is a thick carpet the color of dried blood. The head of a dragon kit is mounted on the wall."

The wood paneling is scenery in the Waiting Suite. The description is "Just the sort of ornate panels that might conceal a carved switch. You've heard all sorts of rumors about secret rooms and passages in the palace, some of which have not been opened in centuries because no one remembers how to get at them." Understand "panels" or "panel" or "panelling" as the paneling.

Instead of switching on the paneling, say "First you'll have to locate any switches or catches with a careful search."

The thick carpet is scenery in the Waiting Suite. Understand "red" or "blood" or "rug" as the carpet. The description is "A dull, unwelcoming weave, only a touch redder than the wood around you. You discern that it does not lie perfectly flat."

Instead of touching the paneling for the first time: say "You run your hands over the paneling with a methodical touch, knowing exactly what you're looking for but never quite feeling anything that gives or twists; then thump lightly, looking for hollow spaces."

Instead of touching the paneling for the second time: say "With increased vigor, you run your fingers along the borders between panels, then smack each panel sharply at the center. No luck yet, but if you keep at it, you're bound to turn up anything that's there to find."

Instead of attacking the paneling: try touching the paneling. Instead of searching the paneling: try touching the paneling. Understand "knock on [something]" or "tap [something]" or "tap on [something]" as attacking.

After touching the paneling when the player is not confident:
    say "Having polished off all the panels within easy reach, you now have to contort yourself around furniture here and crawl along the floorboards there, hitting each panel three times quite solidly before moving on.";
now the player is embarrassed.

Instead of looking under the carpet for the first time:
    say "You take a corner of the carpet and tug. The floor is sticky, so it doesn't come up on the first try."

A small table is an enterable supporter in the Waiting Suite. On the table is a copy of Dragon Pursuit Today. The description of Dragon Pursuit Today is "Full of glossy illustrations of dragons in various stages of capture, captivity, and destruction. The back of the magazine contains small black-and-white advertisements for hunting kits and the like." Some advertisements and some illustrations are part of Dragon Pursuit Today. The description of the illustrations is "You have the misfortune to look first at the photographs accompanying
```

```
'Cleaning Dragon Splanchna', and feel quite unwell." The description of the advertisements is "Mostly terse ads and phone numbers."

After looking under the table:
    say "It's quite a low table and you have to get down on your knees and poke your head underneath in order to get a good look."

After looking under the carpet:
    say "You pull again at the carpet. There is a tug, then a tearing, as the ancient fabric struggles against the fabric glue. Some of the carpet winds up in your hand and some of it remains in patchy threads adhering to the floor."

After entering the table:
    say "You climb onto the small table, noticing belatedly that you are leaving muddy footprints on its polished surface. Oh well: you can wipe them away again when you get down."

The dragon head is scenery in the Waiting Suite. Understand "kit" or "mouth" as the dragon head. The description is "Its eyes are wide with bewildered surprise; its mouth gapes, its forked tongue protrudes indignantly. From down here it looks as though there's something shiny stuck in its mouth, though you can't tell for sure." The head contains a shiny thing. The description of the shiny thing is "Intriguing but impossible to see clearly." Instead of taking the shiny thing, try searching the dragon head.

Before searching the dragon head:
    if the player is not on the table, try entering the table;
    if the player is not on the table, stop the action.

After searching the dragon head: say "You have a good look inside the dragon's mouth. There's a ball of lucite inside, propping the jaw in display position."

A person can be confident, nervous, or embarrassed. The player is confident.

Touching the paneling is embarrassing behavior. Looking under the carpet is embarrassing behavior. Entering the table is embarrassing behavior. Looking under the table is embarrassing behavior.

Instead of embarrassing behavior:
    if the player is nervous, now the player is embarrassed;
    if the player is confident:
        say "Before you can act, you hear movement from the inner office. You freeze, not quite ready to be discovered in this situation. But no one comes out, and you begin to breathe more easily.";
        now the player is nervous;
    otherwise:
        continue the action.

Causing trouble is a scene. Causing trouble begins when play begins. Causing trouble ends when the player is embarrassed. When Causing trouble ends: say "Just at this inopportune moment, you hear a throat being cleared behind you. 'We can see you now within,' says a dry voice."; end the story saying "To be continued..."

Test me with "switch paneling / touch paneling / g / g / g".
Test more with "x dragon / x shiny / search head / g".
```

...and this scene might lead to another, and so on.

The purpose of an open-ended scene like this might be puzzly or narrative: we might be waiting for the player to get a puzzle solved, or we might be waiting for him to fulfil some plot condition that must be met before we can go on.

156 ★ Example Age of Steam

RB

The railway-station examples so far put together into a short game called "Age of Steam".

The following source is very short and simple, yet it already feels surprisingly interesting in play, because something is going on which the player does not control but must observe. The single scene both starts and finishes.

```
"Age of Steam"

The Station is a room. "Eynforme Halt is a raised platform fringed with cowslip: a whistle-stop with no more than a signal and a water-tank."

The Flying Scotsman is fixed in place. "The Flying Scotsman, fastest train in the world, is now at a dead standstill."

Train Stop is a scene. Train Stop begins when the player is in the Station for the third turn. Train Stop ends when the time since Train Stop began is 3 minutes.

When Train Stop begins:
    now the Flying Scotsman is in the Station;
    say "The Flying Scotsman pulls up at the platform, to a billow of steam and hammering."

When Train Stop ends:
    now the Flying Scotsman is nowhere;
    say "The Flying Scotsman inches away, with a squeal of released brakes, gathering speed invincibly until it disappears around the hill. All is abruptly still once more."

Instead of entering the Flying Scotsman, say "Alas, the [time when Train Stop began] arrival is only to take on water, not to set down or pick up."

Test me with "z / z / z / enter flying scotsman / z / z".
```

157 ★ Example Full Moon

RB

Random atmospheric events which last the duration of a scene.

```
"Full Moon"

Wolf Pursuit is a scene. Wolf Pursuit begins when play begins.

Every turn during Wolf Pursuit, say "[one of]A twig snaps behind you![or]The wind howls in your ears.[or]You feel chilly.[at random]".

When play begins:
    say "You have lost your spectacles, and the lamp, and can see barely further than the next tree. Roots keep trying to trip you, too..."

The Dark Forest is a room. "You are mostly aware that you are not as alone here as you would like, and that the ground is uneven."

Test me with "z / z / z / z".
```

158 ★★★ Example Space Patrol - Stranded on Jupiter! RB

We'll be back in just a moment, with more exciting adventures of the... Space Patrol!

American radio adventure series of the 1950s were unobtrusively sponsored by breakfast cereals, as the following modest example demonstrates. Note that the scene-changing for the commercial break needs to know nothing about the actual programme it breaks into: if Part I were replaced with a different Space Patrol episode, Part II need not be changed at all.

```
"Space Patrol #57 - 1953-10-31 - Stranded on Jupiter!"

Use scoring.

Part I - Serial

Red Spot is a room. "You are in the middle of a vast red oval plain. Overhead, the thick Jovian clouds swirl menacingly, and a fine acrid dust falls instead of rain." Some acrid dust is scenery in the Red Spot. The description of the dust is "The rust-colored dust coats every surface. You've no idea how deep it goes."

Instead of going in Red Spot, say "As you once told Cadet Lucky, Jupiter's a mighty big planet, maybe bigger than Iowa. Why, the Red Spot alone stretches out almost to the horizon."

The player wears a silver uniform and rubber boots. The player carries a shovel and an Analscope. The description of the Analscope is "As you recall from Space Patrol #9 - 1952-11-29 - The Electronic Burglar, the Analscope is a device for locating buried metals. That's what guided you all the way from the orbit of Uranus. (Oh, all right, Neptune.) If only you hadn't crashed!"

The metal plate is a fixed in place container. It is openable and closed. In the metal plate is some water. The description of the metal plate is "Stamped with the distinctive logo of the previous mission."
```

```
Instead of examining the player, say "Your hair clumps together stickily, thanks to the dustfall."

Digging is an action applying to one thing. Understand "dig [something]" or "dig in [something]" as digging.

Instead of digging the dust, try looking under the dust. Instead of looking under the dust when the metal plate is not visible: move the metal plate to the location; say "You brush aside the dust underfoot and -- what were the odds? -- it turns out that you landed just where the previous landing party did, thirteen ill-fated years ago. Here is the metal plate that covers their original well.

But wait! Called by the clanging of your shovel on the plate, a band of Jovian pterodactyls swoop down to attack! You're totally defenceless! You don't have a hope! You're absolutely finished!"; increase the score by 10; move K-Klak to Red Spot.

K-Klak the Pterodactyl is an animal. "K-Klak, leader of the Jupiter Pterodactyls, menaces you. A terrifying creature of scaly wings, with a dragon's tail, K-Klak stands... about 1/8th of an inch tall." Instead of doing something to K-Klak, say "K-Klak makes a frankly panicky noise and leaps backwards, out of your way."

After opening the metal plate: increase the score by 10; say "You have found water! You're saved! K-Klak makes a (very cautiously) pleased noise. Now to find the stolen Brainograph, and track down the crook with the thick Jewish accent and his henchmen with their thick Polish accents..."; end the story finally.

The maximum score is 20.

Part II - Cereal

When play begins, say "Instant Ralstons and Regular Ralstons, the hot whole-wheat cereals in the red and white checkerboard packages present... SPACE PATROL... High adventure in the wild vast reaches of space... Missions of daring in the name of interplanetary justice... Travel into the future as Buzz Corey, Commander-in-Chief of the..."

Last score is a number that varies. Every turn: now the last score is the score.

Ralstons Ad is a scene. Ralstons Ad begins when score is not the last score. Ralstons Ad ends when the Ricechex is consumed.

Include Basic Screen Effects by Emily Short.

When Ralstons Ad begins:
   center "*** We'll be back in just a moment! ***";
   pause the game;
   strip the player;
   move the player to the Kitchen.

When Ralstons Ad ends:
   center "*** And now, back to today's exciting adventure ***";
   pause the game;
   restore the player.
```

```
Saved location is a room that varies. Locker is a container. Wardrobe is a
container.

To strip the player:
    now every thing carried by the player is in the locker;
    now every thing worn by the player is in the wardrobe;
    now saved location is location.

To restore the player:
    now every thing carried by the player is in the Kitchen;
    now every thing in the locker is carried by the player;
    now every thing in the wardrobe is worn by the player;
    move the player to saved location.

The Space Patrol Kitchen is a room. "The nerve center of the Space Patrol! This
is where cadets fill up with their SUPER-FUEL. North leads to the astro control
room, while back south is the cargo hold." A breakfast bowl is in the Kitchen. In
the bowl is Ricechex. Ricechex is edible. The Ricechex can be consumed or
uneaten. The Ricechex is uneaten.

Instead of going north in Kitchen: say "[refusal to leave]". Instead of going south
in Kitchen: say "[refusal to leave]".

Instead of examining the player when Ralstons Ad is happening: say "You are
currently being played by a generically attractive person of about 30, with very
good teeth and well-kept nails."

After eating the Ricechex: say "That's right folks, always start your day the
SPACE PATROL way with a tasty bowl of Ricechex, Wheatchex or good hot
Ralstons. Mmmm Mmmm. You just can't get enough of the sugary goodness in
Ricechex, Wheatchex and good hot Ralstons."; now the ricechex is consumed.

Instead of tasting the Ricechex:
    say "Wow! *wolf-whistle* Man oh man oh man! Yumm-y!"

To say refusal to leave:
    repeat through Table of Refusals:
        say "[nope entry][paragraph break]";
        blank out the whole row;
        rule succeeds;
    say "You can't. Eat your Ricechex."

Table of Refusals
nope
"You can't go that way in the limited universe of this sponsored message."
"Or that way."
"You've already tried that!"
"Why would you want to walk away when you have an alluring bowl of Ricechex
right here?"

Test me with "n / i / x me / x dust / dig dust".

Test ad with "n / s / n / s / n / i / x me / get bowl / taste ricechex / eat ricechex".

Test ending with "x plate / x k-klak / open plate".
```

Episode 57 of "Space Patrol" was actually called "Iron Eaters Of Planet X", just in case the reader feels that any of the foregoing unfairly traduces a work of thoughtful science fiction.

---

159 ★★★ **Example** **Bowler Hats and Baby Geese** *RB*

Creating a category of scenes that restrict the player's behavior.

Scenes can have properties -- a fact that is very useful when it comes to writing a series of scenes that all need to act alike in some respect.

Suppose we have a plot that features a number of scripted scenes, where we need the player to stand still and wait while the events of the scene play out. One way to set this up is to create a property for such scenes -- let's call them "restricted" -- and then write a rule that keeps the player in place while the scene happens:

```
"Bowler Hats and Baby Geese"

Section 1 - The Procedure

A scene can be restricted or free.

Instead of going somewhere during a restricted scene:
    say "Better to stay here for the moment and find out what is going to happen next."
```

And now let's set up our restricted scene. In it, a clown is going to turn up wherever the player is (it doesn't matter where on the map he's gotten to at this point) and do a performance; the player will not be able to leave the area until the performance completes. We'll start with the setting:

```
Section 2 - The Stage and Props

The Broad Lawn is a room. "A sort of fun fair has been set up on this broad lawn, with the House as a backdrop: it's an attempt to give local children something to do during the bank holiday. In typical fashion, everyone is doing a very good job of ignoring the House itself, despite its swarthy roofline and dozens of blacked-out windows."

The House is scenery in the Broad Lawn. The description is "A cautious vagueness about the nature of the inhabitants is generally considered a good idea. They might be gods, or minor demons, or they might be aliens from space, or possibly they are embodiments of physical principles, or expressions of universal human experience, or... at any rate they can run time backward and forward so it warbles like an old cassette. And they're always about when somebody dies. Other than that, they're very good neighbors and no one has a word to say against."

Instead of entering the House:
    say "You can't go in, of course. It's not a house for people."
```

```
The Gazebo is north of the Broad Lawn. "The gazebo is sometimes used for bands, but at the moment has been appropriated for the distribution of lemonade."

The clown is a man. "A clown wearing [a list of things worn by the clown] stands nearby." The description is "He winks back at you."

The clown wears a purple polka-dot bowler hat. He carries a supply of baby geese. The description of the supply of baby geese is "Three or four. Or five. It's hard to count." Understand "goose" or "gosling" or "goslings" as the supply of baby geese.

There are some eggs. The description of the eggs is "A blur, really."

There is a Spanish omelet. The description of the Spanish omelet is "Exquisitely prepared."
```

...And now the scene itself:

```
Section 3 - The Scenes

The Clown Performance is a restricted scene. Clown Performance begins when the turn count is 3.

When Clown Performance begins:
    move the clown to the location.

Every turn during Clown Performance:
    repeat through the Table of Clowning:
        say "[event description entry][paragraph break]";
        blank out the whole row;
        stop.

When Clown Performance ends:
    now the eggs are nowhere;
    now the clown carries the omelet.

Clown Performance ends when the number of filled rows in the Table of Clowning is 0.

Table of Clowning
event description
"A clown with a purple polka-dot bowler hat strides into the vicinity and begins to juggle baby geese."
"While the clown juggles, the baby geese visibly grow older and larger. The clown becomes unnerved."
"In an attempt to resolve the problem, the clown reverses the direction of his juggling. The geese revert to goslings."
"The goslings become smaller and smaller until the clown is juggling goose eggs[replace eggs]."
"The clown throws all the eggs into the air at once and catches them in the bowler hat. He takes a bow; the audience applauds. As a final gesture, he upends his hat to release a perfectly cooked omelet."
```

```
To say replace eggs:
    now the supply of baby geese is nowhere;
    now the clown carries the eggs.

Free Time is a scene. Free Time begins when Clown Performance Ends.

Test me with "scenes / n / z/ z / look / x geese / s / x geese / x eggs / z / s".
```

160 ★★★ **Example** **Day One** RB

A scene which plays through a series of events in order, then ends when the list of events is exhausted.

```
"Day One"

Lecture is a scene. Lecture begins when play begins.

Every turn during Lecture:
    repeat through Table of Lecture Events:
        say "[event entry][paragraph break]";
        blank out the whole row;
        rule succeeds.
```

Here we use a table (see subsequent chapters) to keep track of all the events we wish to have occur during the course of the scene.

```
Table of Lecture Events
event
"'Welcome to Precolumbian Archaeology 101,' thunders Dr Freitag from the front of the class. 'Miss-- yes, you in the back. If you can't find a free seat, how are you going to find Atlantis? Sit down or leave. Now. Thank you.'"
"Freitag stands behinds his desk and lines up the pile of books there more neatly. 'It has come to my attention over previous years that there are two sorts of person who enroll in my class,' he says.

'Some of you will be members of the swim team or women's lacrosse players who have a distribution requirement to fulfill and are under the mistaken impression that archaeology must be easier than psychology. If that description applies to you, I advise you to drop the class now rather than at the midterm break. Under absolutely no circumstances will I ever sign a withdrawal form for someone who is crying at the time. Make a note of that, please.'"
"'The second sort of person,' Dr Freitag says, getting another wind. 'Yes, the second sort of person takes this class because she imagines that it is going to lead to adventure or possibly to new age encounters with dolphins.'

His eye moves over the class, lingering an especially long time on a girl in a patchwork skirt.

'You should also leave now, but since you are probably lying to yourself about the reasons you're here, you will probably not heed my warning and we will be doomed to a semester of one another's company nonetheless.'"
"'Whatever you may tell yourself, you are not here to gain a deeper
```

```
understanding of the world or get in touch with yourself or experience another
culture.'

He paces before the first row of desks, hammering on them one at a time. 'I
know you probably wrote an admissions statement saying that that is what you
hoped to do. Well, too bad. It is not inconceivable that some of you, somehow,
will muddle towards a deeper understanding of something thanks to this class,
but I am not holding my breath, and neither should you.'"
"Freitag takes a breath. 'No, my dear freshwomen, what you are here to do is
learn facts. FACTS. Facts are unpopular in this university and, I am unhappily
aware, at most of the institutions of inferior preparation from which you have
come. Nonetheless, facts it will be. I will expect you to learn names. I will expect
you to learn dates. I will expect you to study maps and I will expect you to
produce evidence of exacting geographical knowledge on the exams. I will
expect you to learn shapes of pottery and memorize masonry designs. There are
no principles you can learn which are more important or more useful than a truly
colossal bank of facts right there in your own head.'"
"'I do not ever want to hear that you do not need to learn things because you will
be able to look them up. This is the greatest fallacy of your computer-semi-
literate generation, that you can get anything out of Google if you need it. Not
only is this demonstrably false, but it overlooks something phenomenally
important: you only know to look for something if you already know it EXISTS. In
short there is no way to fake knowledge, and I am not going to pretend there is.'
He smiles in lupine fashion.

'This class is likely to be the most miserable experience of your four years in
university. Clear?'"
"Everyone is silent."
"The lecture is interrupted by the shrill of a bell."
```

And then we define the scene so that it ends when the table runs out.

```
Lecture ends when the number of filled rows in the Table of Lecture Events is 0.
```

One advantage of this is that we can then edit the events in the scene by changing just the table; the scene will always run the right length and end on the turn when the last event occurs.

And to add a few additional details:

```
Instead of doing something other than waiting, looking, listening or examining
during Lecture:
    say "Dr Freitag glares at you so fiercely that you are frozen into inaction."
```

Notice the careful phrasing of "doing something other than..." so that we do not mention the objects; if we had written "something other than listening to something...", the instead rule would match only action patterns which involved a noun. We state the rule more generally so that it will also match nounless commands such as JUMP and SING, since Freitag will probably take a dim view of those as well.

```
When Lecture ends:
    now Freitag is nowhere;
    say "There is a flurry of movement as your fellow students begin to put away
```

```
their books. Dr Freitag makes his way to the door and is gone before anyone can ask him anything."

The Classroom is a room. Dr Freitag is a man in the Classroom. "Dr Freitag paces before the blackboard."

Test me with "listen / x dr / x me / jump / z / z / z / z / z / x dr".
```

161 ★★★ Example **Pine 2** RB

Pine: Adding a conversation with the princess, in which a basic set of facts must be covered before the scene is allowed to end.

```
"Pine"

A person can be asleep or awake. A person can be active or passive.

The Spinning Tower is a room. "A remote corner of the old castle, reserved for spinning and weaving tasks."

Sleeping Beauty is an asleep woman in the Spinning Tower. "[if asleep]Sleeping Beauty lies here, oblivious to your presence[otherwise]Sleeping Beauty stands beside you, looking a little confused[end if]." The description is "She is even more magnificent than the rumors suggested." Understand "woman" or "girl" or "princess" or "lady" as Sleeping Beauty.

Discovery is a scene. Discovery begins when play begins. Discovery ends when Sleeping Beauty is awake. Marriage Proposal is a scene. Marriage Proposal begins when Discovery ends.

When Discovery ends: say "Throughout the palace you can hear the other sounds of stirring and movement as the spell of centuries is broken."

Instead of waking an awake person: say "Redundant."

Instead of waking an asleep person: say "Yes, but how?"

Instead of attacking an asleep person:
    now the noun is awake;
    say "[The noun] sits bolt upright. 'Hey! Ow!' So much for that true love's kiss nonsense."

Instead of kissing an asleep person:
    now the noun is awake;
    say "[The noun] slowly stirs to wakefulness!"

Instead of throwing water at an asleep person:
    now the second noun is awake;
    now the noun is nowhere;
    say "You pour out [the noun] on [the second noun].
```

```
[The second noun] wakes, shuddering. 'Agh! I had a terrible dream about
drowning and then-- Hey!'"

The player carries a jug of water. Understand "pour [something] on [something]"
or "splash [something] at/on [something]" as throwing it at.
```

So much, we had before. Now, suppose we want a conversation style which allows the player to move conversation forward by asking appropriate questions, but which will keep moving forward even if he doesn't. To this end, we provide a table -- a borrowing from a later chapter. In the table, we record two ways of performing each conversation bit, one which reflects the player's participation, and one in which the character moves things onward:

```
Table of Conversation

    topic                                          reply                                  quip
    "dream/dreams/nightmare/nightmares/sleep" "'Sleep well?' you ask solicitously.


    'Not really,' she replies, edging away from you.   "'Ghastly nightmares,' she remarks. You nod politely."
    So much for that angle."
    "marriage/love/wedding/boyfriend/beau/lover"       "'So,' you say. 'This is a little weird since we just met, but, um.
                                                       Would you like to get married?'


    She looks at you nervously. 'Do I have to?'"    "'I, er,' she says. 'I hope I'm not supposed to marry you or
                                                    something.'"
    "marriage/love/wedding/boyfriend/beau/lover" "'I was told I was going to marry you and inherit the kingdom,' you
                                                    say, apologetically. 'Would that be very bad?'

'Oh, it's not you -- I'm seeing someone,' she says, smiling quickly.

    You try to think how to point out that it's been a  "'Do you think I could go look for someone? I'm seeing him, you
    hundred years since she last saw her                 see, and I think I've been... sick... for a while, so he might be
    boyfriend."                                          worried.'

You try to think how to point out that it's been a hundred years since she last saw
her boyfriend."

    "marriage/love/wedding/boyfriend/beau/lover" "'You've been up here for a hundred years,' you say. An unpleasant
                                                    thought occurs to you. 'Was your young man in the castle
                                                    somewhere?'


    She shakes her    "She goes to the window and looks out at the now-fading thicket of briar. 'That took a while to
    head mutely."     grow,' she observes. 'I've been up here longer than I thought.'

You shrug, uncomfortable."

Instead of asking an awake beauty about a topic listed in the Table of
Conversation:
    now Beauty is passive;
    say "[reply entry][paragraph break]";
    blank out the whole row.
```

The "now Beauty is passive" line prevents her from making any conversation of her own on a turn when we've spoken to her. This keeps the conversation from progressing too quickly.

```
Instead of telling an awake beauty about something: try asking the noun about it.
Instead of asking an asleep person about something:
    say "[The noun] snores."

Marriage Proposal ends when the number of filled rows in the Table of
Conversation is 0.

Every turn during Marriage Proposal:
    if Beauty is active:
        repeat through Table of Conversation:
            say "[quip entry][paragraph break]";
            blank out the whole row;
            make no decision.
```

After we've generated any spontaneous conversation, we return her to her regular active state.

```
Every turn: now Beauty is active.

When Marriage Proposal ends: end the story saying "This is going to take some
explaining."

Test me with "x beauty / wake beauty / pour water on beauty / ask beauty about
sleep / z / ask beauty about marriage".
```

Now we have a scenario in which the player can ask her some questions out of order if he really wants to, but the scene will not end until the basic conversation topics have been exhausted. If we wanted to add some other chit-chat, not as part of the main conversation strand, but by way of optional enrichment, we might make a second conversation table and record alternative outcomes in it.

---

162 ★ **Example** **The Prague Job** *RB*

Scenes used to provide pacing while the player goes through his possessions.

Suppose we want to remind the player that he doesn't have all the time in the world, by starting to nag him when he's nearly, but not entirely, done going over his inventory in preparation for a job.

```
"The Prague Job"

A thing can be seen or unseen. A thing is usually unseen. Carry out examining:
now the noun is seen.

The player carries a lockpick, a smoke bomb, a grappling hook, and a pair of
gloves. The description of the lockpick is "Effective on most kinds of key locks, it
is a gift from your mentor in the discipline, old Wheezy." The description of the
smoke bomb is "Your last of these, so you should rely on it only when other
modes of escape have vanished. It takes effect when dropped, producing a
cloud of purple haze sufficient to fill a medium-sized room." The description of
the grappling hook is "Good for shooting at balconies and other sorts of
```

```
overhang." The gloves are wearable. The description of the gloves is "Black and shiny, with gripping material on the palms. Batman would be jealous."

The Toilet is a room. "The walls are painted an unattractive green; the fixtures are a bit old. But it is the only place in the hostel with any privacy." The Long Hallway is outside from the Toilet.

Reviewing Possessions is a scene. Reviewing Possessions begins when play begins.

Escalating Danger is a scene. Reviewing Possessions ends when Escalating Danger begins. Escalating Danger begins when preparations near completion.

To decide whether preparations near completion:
    if at least two of the things which are carried by the player are seen, yes;
    no.

When Escalating Danger begins: say "Someone pounds on the door of your hideout and yells at you in Czech."

Instead of going from the Toilet during Reviewing Possessions: say "You need to go over your equipment first, and make sure you're ready here."

Instead of going from the Toilet during Escalating Danger: say "You're not done checking over your materials."

Instead of waiting during Escalating Danger: say "There's no time to waste."

Every turn during Escalating Danger: if the time since Escalating Danger began is greater than 1 minute, say "Impatient footsteps pass your door again."

Escalating Danger ends when every thing which is carried by the player is seen. When Escalating Danger ends, say "There -- nothing damaged or torn. You're ready to go."

Mission is a scene. Mission begins when the player is in the Long Hallway. When Mission begins: end the story saying "The game is afoot"

Test me with "i / x lockpick / out / x bomb / out / x hook / x gloves / out".
```

---

163 ★★★★ **Example Entrevaux** RB

Organizing the game by scenes, where each scene has a location and prop lists so that it can be set up automatically.

For some games, it makes sense to organize the entire game around scenes rather than around locations, moving the player when a new scene begins and laying out new props.

To this end, we might extend Inform's default handling of scenes so that each scene has properties to indicate prop lists and locations, and move objects in and out of play automatically as the scenes change. For instance:

```
"Entrevaux"

Part 1 - Procedure

A scene has a room called the starting location.

A scene has a list of objects called the scenery props.

A scene has a list of objects called the inventory props.
```

The starting location is the room to which the player should be moved; scenery props are things that need to be put there when the scene begins; inventory props, things that are given to the player when the scene begins; and the description some printed text to introduce the new scene. We may still occasionally need to have recourse to special "When the Dancing-Lesson begins..." rules for individual scenes, but for the most part this allows us to set scenes up in a consistent and predictable way.

Another point that might be slightly less obvious: sometimes we want to announce a change of location to the player when the scene starts, and sometimes we don't. In particular any scene that starts "when play begins" should probably not explicitly describe the entered room, since that would duplicate the description automatically produced on the first turn of play. So we add a property to track whether any given scene should be announcing its location:

```
A scene can be location-silent or location-loud.
```

And let's say that we also want to announce each new scene as another "chapter" of the game in play, with a pause before the scene begins.

Here we include "Basic Screen Effects" because it will allow us to pause the game for a keypress, then clear the screen before each new chapter:

```
Include Basic Screen Effects by Emily Short.

The chapter counter is a number that varies.

First when a scene (called the current scene) which is not the Entire Game
begins:
    if chapter counter is greater than 0:
        pause the game;
    increment chapter counter;
    say "[bold type]Chapter [chapter counter]: [current scene][roman type]";

Last when a scene (called the current scene) which is not the Entire Game
begins:
    repeat with item running through the scenery props of the current scene:
        move the item to the starting location of the current scene;
    repeat with item running through the inventory props of the current scene:
        move the item to the player;
    if the location is not the starting location of the current scene:
        if the current scene is location-loud:
            move the player to the starting location of the current scene;
        otherwise:
            move the player to the starting location of the current scene, without printing a room description.
```

At the end of each scene, we strike the set and remove all the loose objects from play.

```
When a scene (called the current scene) ends:
    repeat with item running through things which are not fixed in place:
        if the item is not the player:
            now the item is nowhere.

Part 2 - Scenario

Entrevaux Station is a room. "The station building consists of a waiting room and a ticket-selling office so small that only one person can buy a ticket at a time. On the outside wall is a clock that runs twelve minutes late; but since the trains also run twelve minutes, give or take, behind their published schedule, this clock is helpful in establishing reasonable expectations. [paragraph break]Painted on the door is the logo of the Chemin de Fer de Provence, the only railway in France that is not part of the SNCF."

The Hillside Tower is a room. "It's very dark in here, lacking artificial lighting, but from the rough rectangular window you can see a slice of hillside and a little of the river Var."

The window is scenery in the Hillside Tower. The description is "Through it can be seen a slice of wooded hillside and exposed grey-brown cliff. You are in the southern French foothills of the Alps, and the territory is dry. The only respite is the river Var, a milky blue at this time of year, running shallowly over mud and large stones far below your window." Understand "view" or "slice of hillside" or "hillside" or "hill" or "river" or "var" or "mud" or "stones" or "large stones" as the window.

A used ticket is a thing. The description is "A piece of receipt paper indicating that you have paid the one-way fare of 9 euros from Nice. There is a hole punched through it."

A one-euro coin is a thing. The description is "It's a bimetal coin, brassy around the rim and silvery in the center. One side shows western Europe, with unusual prominence given to the UK, and the other side Leonardo da Vinci's four-armed, four-legged man having a nice stretch. It's dated 2002."

Some re-enactors are a person. "Milling about one end of the station is a crowd of medieval re-enactors." The description is "They're dressed in a somewhat aimless range of styles roughly honoring the period of 900-1500 AD. One gentleman is wearing a knobby leather cap; which is a good thing, because there is a rooster standing on his head." Understand "men" or "man" or "gentleman" or "rooster" or "reenactors" or "crowd" or "medieval" or "woman" or "women" as the re-enactors.

A kidnapper is a person. "Your kidnapper is watching you from the corner with his arms folded. You have the impression he's just marking time until someone more important arrives." The description is "He does not look at all like the
```

```
kidnapping sort, but more like a sommelier at a superior restaurant: he wears a
black pinstriped suit and has nicely-manicured hands."

The trolley is an enterable fixed in place container. "The 'train' on which you
arrived is really just a single car, more like a trolley than a proper train."
Understand "car" or "train" as the trolley. The description is "It has a glass front,
so you can see ahead while riding: an innovation among trains."

Arrival is a location-silent scene. "After many days['] journey, you have arrived at
last in Entrevaux, a walled medieval town now chiefly of interest to tourists and
crusade re-enactors."
    The starting location of Arrival is the Entrevaux Station.
    The scenery props of Arrival are { re-enactors, trolley }.
    The inventory props of Arrival are { the used ticket, one-euro coin }.

Arrival begins when play begins. Arrival ends when the time since Arrival began
is 2 minutes.

Abduction is a location-loud scene. "You check into the Hotel Vauban and sleep
deeply enough; it was a long and sticky trip to get here.

Then in the middle of the night something confusing happens. You have the
impression of strangers in your room, and then a searing pain, and you don't
come back to yourself until midmorning of the following day..."
    The starting location of Abduction is the Hillside Tower.
    The scenery props of Abduction are { kidnapper }.

Abduction begins when Arrival ends.

Test me with "i / x re-enactors / z / z / i / x him".
```

164

★ Example Night and Day

RB

Cycling through a sequence of scenes to represent day and night following one another during a game.

Suppose we want to have a sequence of nights and days in our game, with one scene to govern each daylight condition.

```
"Night and Day"

The sun is a backdrop. It is everywhere. The description is "Currently out of
sight."

Night is a recurring scene. Night begins when play begins. Night begins when
Dusk ends. Night ends when the time since Night began is 3 minutes.
```

Notice that our two conditions for the beginning of Night are not in conflict: it will be night-time when the game begins, and then night will also recur every time the Dusk scene ends.

```
When Night begins:
    say "The sun falls below the horizon and the temperature drops abruptly to
well below zero.";
    now the description of the sun is "Currently out of sight."

Dawn is a recurring scene. Dawn begins when Night ends. Dawn ends when the
time since Dawn began is 1 minute.

When Dawn begins:
    say "The sun appears on the horizon.";
    now the description of the sun is "It is tiny and weak.".

Day is a recurring scene. Day begins when Dawn ends. Day ends when the time
since Day began is 3 minutes.

When Day begins:
    say "The sun is now properly up."

Dusk is a recurring scene. Dusk begins when Day ends. Dusk ends when the
time since Dusk began is 1 minute.

When Dusk begins:
    say "The sun has passed across the sky and is on the verge of setting."

Cratered Landscape is a room. "The ground here is [if Night is happening]dim
silver, with the craters visible as darker splotches[otherwise]the color of dried
blood; here and there it is also rippled by impact craters[end if]. The horizon
curves visibly."

Test me with "z / z / z / look / x sun / z / z / z / z / z / z / z".
```

If we run this example and then have a look at the scenes index, we'll see that the cycle is listed through thus:

```
Night (recurring)
    Dawn (recurring)
        Day (recurring)
            Dusk (recurring)
                Night
```

with the second "Night" in italics, to indicate that it is a repetition of the same scene that has already been listed above.

165 ★★★★ **Example Pine 3** RB

Pine: Allowing the player to visit aspects of the past in memory and describe these events to the princess, as a break from the marriage-proposal scene.

```
"Pine"
```

```
Part 1 - The Set-up
```

This is mostly a repeat of what we have already seen, but for the sake of producing a playable scenario, we include it. The new material appears at Part 2.

```
A person can be asleep or awake. A person can be active or passive.

The Spinning Tower is a room. "A remote corner of the old castle, reserved for
spinning and weaving tasks."

Sleeping Beauty is an asleep woman in the Spinning Tower. "[if asleep]Sleeping
Beauty lies here, oblivious to your presence[otherwise]Sleeping Beauty stands
beside you, looking [attitude][end if]." The description is "She is even more
magnificent than the rumors suggested." Understand "woman" or "girl" or
"princess" or "lady" as Sleeping Beauty.

Discovery is a scene. Discovery begins when play begins. Discovery ends when
Sleeping Beauty is awake. Marriage Proposal is a scene. Marriage Proposal
begins when Discovery ends.

When Discovery ends: say "Throughout the palace you can hear the other
sounds of stirring and movement as the spell of centuries is broken."; now
Beauty is passive.

Instead of waking an awake person: say "Redundant."

Instead of waking an asleep person: say "Yes, but how?"

Instead of attacking an asleep person:
now the noun is awake;
    say "[The noun] sits bolt upright. 'Hey! Ow!' So much for that true love's kiss
nonsense."

Instead of kissing an asleep person:
    now the noun is awake;
    say "[The noun] slowly stirs to wakefulness!"

Instead of throwing water at an asleep person:
    now the second noun is awake;
    now the noun is nowhere;
    say "You pour out [the noun] on [the second noun].

[The second noun] wakes, shuddering. 'Agh! I had a terrible dream about
drowning and then-- Hey!'"

The player carries a jug of water. Understand "pour [something] on [something]"
or "splash [something] at/on [something]" as throwing it at.

Table of Conversation

   topic                                    reply                                   quip
   "dream/dreams/nightmare/nightmares/sleep" "'Sleep well?' you ask solicitously.

   'Not really,' she replies, edging away from you.     "'Ghastly nightmares,' she remarks. You nod politely."
   So much for that angle."
   "marriage/love/wedding/boyfriend/beau/lover"      "'So,' you say. 'This is a little weird since we just met, but, um.
                                                     Would you like to get married?'


   She looks at you nervously. 'Do I have to?'"    "'I, er,' she says. 'I hope I'm not supposed to marry you or
                                                   something.'"
   "marriage/love/wedding/boyfriend/beau/lover"    "'I was told I was going to marry you and inherit the kingdom,' you
                                                   say, apologetically. 'Would that be very bad?' This could be
                                                   awkward, considering your family circumstances -- you did promise
                                                   your mother that everything would be better, after this --

'Oh, it's not you -- I'm seeing someone,' she says, smiling quickly.

   You try to think how to point out that it's been a  "'Do you think I could go look for someone? I'm seeing him, you
   hundred years since she last saw her                see, and I think I've been... sick... for a while, so he might be
   boyfriend."                                         worried.'

You try to think how to point out that it's been a hundred years since she last saw
her boyfriend. And try not to think how awkward things would be in your family if
she refuses to marry you."

   "marriage/love/wedding/boyfriend/beau/lover"    "'You've been up here for a hundred years,' you say. An unpleasant
                                                   thought occurs to you. 'Was your young man in the castle
                                                   somewhere?'


   She shakes her     "She goes to the window and looks out at the now-fading thicket of briar. 'That took a while to
   head mutely."      grow,' she observes. 'I've been up here longer than I thought.'

You shrug, uncomfortable."

Instead of asking an awake beauty about a topic listed in the Table of
Conversation:
   now Beauty is passive;
   say "[reply entry][paragraph break]";
   blank out the whole row.

Instead of telling an awake beauty about something: try asking the noun about it.

Instead of asking an asleep person about something:
   say "[The noun] snores."

Marriage Proposal ends when the number of filled rows in the Table of
Conversation is 0.

Every turn during Marriage Proposal:
   if Beauty is active and Beauty is visible:
      repeat through Table of Conversation:
         say "[quip entry][paragraph break]";
         blank out the whole row;
         make no decision.

Every turn: now Beauty is active.

When Marriage Proposal ends: end the story saying "This is going to take some
explaining."
```

So far we haven't much of a chance to affect matters and make them better. Suppose we'd like to add an element to the conversation where we're allowed to tell Beauty about past events -- and explore them a bit; and if the first retelling doesn't go quite as planned, we're allowed to revisit these scenes to hit them with a bit more emphasis.

```
Part 2 - Flashbacks

Instead of asking an awake beauty about a topic listed in the Table of Flashback Material:
    now Beauty is passive;
    say "[reply entry][paragraph break]".

A fact is a kind of thing. The family circumstances is a fact. A fact can be known or unknown. A fact can be current or past.
```

Once known, a fact remains known permanently -- this could be useful if we wanted to make some rules about how Beauty acts when she knows different information. By contrast, a fact is only "current" if it is the last thing discussed. Since a player can mention a fact over and over, he can make it "current" again and again, and thus reactivate the flashback.

```
Table of Flashback Material
```

topic	reply
"poor/poverty/family/money/mother/circumstances" or "family circumstances" or "my family/mother"	"[if family circumstances is unknown]'I wish you'd give some thought to marrying me. You see,' you say, your jaw tensing. 'I wouldn't ask if it weren't for my [family circumstances]...'[otherwise]'I don't think you fully understand the [family circumstances],' you say.[end if]"

```
After printing the name of a fact (called target): now the target is current; now the target is known.
```

This "After printing the name..." rule will be explained later in the chapter on activities; for now, it is enough to know that whenever family circumstances is mentioned in the table of flashback material, this rule will automatically be called. Now the terms under which the flashback happens:

```
Poverty flashback is a recurring scene. Poverty flashback begins when family circumstances is current. When poverty flashback begins: strip the player; move the player to the Woodcutter's Shack.
```

Note the "recurring" here: we want the player to be able to revisit this scene as needed.

```
The Woodcutter's Shack is a room. "Your family lives in a shack in the forest. There are holes in the roof, and in the winter the snow comes in -- rain, too, for that matter. The walls aren't very well-boarded, and don't keep out the wind, and even though you live in the middle of dense woods, you can never gather enough fuel to keep this place fully heated. And then there's the stench. Pigs wander freely in and out, and your three youngest brothers play on the floor."

Pigs are an animal in the shack. The pigs are scenery. The description is "They really are very grubby, dirty animals, but what's worse than that, the value of pigs has declined a lot over the last few decades. This is hard to explain to someone who has been out of touch with the world for a while, but keeping pigs for meat is
```

a dubious prospect when there's less and less for them to forage on." Instead of smelling the pigs: say "They smell the way animals do, when they live among their own refuse."; increase the pity of Beauty by 2.

The brothers are a man. The description of brothers is "Hans, Franz, and Lukas. Twins and then the baby. So young, and growing up fatherless; and soon to be orphaned entirely, if your mother's health does not improve." Understand "brother" or "twin" or "twins" or "baby" or "franz" or "hans" or "lukas" as the brothers.

The untidy bed is scenery in the Shack. Mother is a woman on the untidy bed. The description of mother is "She is wasting away of a slow disease, her skin stretched tautly over bone. She hasn't been the same since your father left." On the bed is a folded letter.

The description of the letter is "Many times read over and creased, the letter explains how your father has gone away with a wealthy countess and will not return. Your mother was not able to read it herself, of course, and had to have it explained to her by the parish priest. Now she keeps it by the bed and crumples it in her fits of delirium."

Instead of kissing or touching Mother for the first time:
    say "You place a gentle kiss on her feverish brow. She looks up at you, her oldest -- yes, never mind that bit -- with a look of sincere trust and admiration.

'You'll find a way through this for us,' she says, squeezing your fingers. 'I know you will.'"; increase the pity of Beauty by 3.

Instead of kissing Mother: say "You have no more heart-rending memories of affection to recount; that one incident will have to serve, for rhetorical purposes."

Instead of waiting in the Shack: say "The wind blows sharply through the walls."

Instead of attacking someone in the Shack:
    say "Though sometimes the conditions of your life make you grouchy and impatient, you would never dream of striking a member of your own family. But from time to time you do feel the temptation."

Beauty has a number called pity. After examining something in the Woodcutter's Shack, increment the pity of Beauty. After examining mother, increase the pity of Beauty by 2. After examining the letter, increase the pity of Beauty by 3.

Poverty flashback ends when waiting or the time since poverty flashback began is five minutes.

When Poverty flashback ends:
    now family circumstances is past;
    say "...you finish describing the miserable circumstances of your home life, and allow your attention to return to the present.";
    restore the player;
    now Beauty is passive;
    if Beauty is sympathetic, say "'Oh dear!' she says. 'What a dreadful life!' She wrings her hands. 'No wonder you are eager to improve your lot...! But --' Her brow clears, a new thought occurring. 'You needn't marry me, you know! We could arrange it differently! I am certain that my father would give you a large

```
reward, instead, and then I would not be separated from my current boyfriend!'";
    otherwise say "She makes a disgusted face, but she doesn't seem nearly so
heart-wrung as you had hoped to make her. Tough audience, these modern
princesses."

Definition: Beauty is sympathetic if the pity of Beauty is greater than 4.

To say attitude:
    if Beauty is sympathetic, say "distressed on your behalf";
    otherwise say "a little confused".
```

And the following is the same as in the Space Patrol example as well: we need a systematic way to remove the player's possessions, then put everything back when the flashback is over:

```
Saved location is a room that varies. Locker is a container. Wardrobe is a
container.

To strip the player:
     now every thing carried by the player is in the locker;
    now every thing worn by the player is in the wardrobe;
    now saved location is location.

To restore the player:
    now every thing carried by the player is in the location;
    now every thing in the locker is carried by the player;
    now every thing in the wardrobe is worn by the player;
    move the player to saved location.

Test me with "x beauty / wake beauty / pour water on beauty / ask beauty about
sleep / tell beauty about poverty / smell pigs / x mother / x letter / kiss mother /
ask beauty about marriage / z / z".
```

Because we haven't changed the endings of the "Marriage Proposal" scene, there is still only one way for this scenario to work out; but at least now the player has the opportunity to alter Beauty's attitude a bit (or not) before the game is done.

---

166 ★★★ Example **Panache**

Replacing the score with a plot summary that records the events of the plot, scene by scene.

RB

If we have a plot that branches and has multiple kinds of outcome, we might well want to assemble these into a plot summary in place of the more traditional score. One way to approach this is to build the scene information into a table, adding information when each scene ends.

We begin with a bit of setup:

```
"Panache"
```

```
The player is in a room called Beneath Roxane's Balcony. Christian is a man in the Balcony. "Christian stands in a spot of moonlight and tries to avoid too obviously glancing at the shadows that conceal you." The description of Christian is "Like you, Christian loves Roxane. Unlike you, he is handsome enough to receive her favor in return. He is the beauty to your brain."

Roxane is a woman in the Balcony. "Above you in the night is Roxane." Roxane can be wooed, skeptical, confused, or annoyed. Roxane is skeptical. The description of Roxane is "The brightest, the most radiant of women -- and in love with an utter fool."

Empty Street is a room. "No one is about at this hour, all alone under a pale moon."

Telling someone about something is speech. Asking someone about something is speech. Answering someone that something is speech.
```

This next portion borrows from the Advanced Actions chapter to allow us to command Christian to do things:

```
A persuasion rule for asking Christian to try speech: persuasion succeeds.

Carry out Christian answering someone that something:
    now Roxane is wooed;
    say "'[noun], [the topic understood].'"

Carry out Christian answering the player that something:
    say "Christian parrots your words back to you." instead.

Carry out Christian telling a skeptical Roxane about something:
    now Roxane is confused;
    say "Christian turns to [the noun]. 'I must tell you about [the topic understood],' he says, and comes to a halt, looking at you for further direction.

    Perhaps you'd better give him exact lines to say. Surely he can't mess up an instruction like 'say hello to Roxane.'" instead.

Carry out Christian asking a skeptical Roxane about something:
    now Roxane is confused;
    say "'So,' says Christian nervously to [the noun]. 'Did you know about [the topic understood]?' But Roxane merely seems puzzled." instead.

Carry out Christian telling a confused Roxane about something:
    now Roxane is annoyed;
    say "Christian begins rambling on witlessly about [the topic understood]." instead.

Carry out Christian asking a confused Roxane about something:
    now Roxane is annoyed;
    say "Christian puts another confused question about [the topic understood]." instead.
```

And now we have enough material to begin writing the scenes:

```
Courting Roxane is a scene. Courting Roxane begins when play begins. Courting Roxane ends in success when Roxane is wooed. Courting Roxane ends in failure when Roxane is annoyed.

When Courting Roxane ends in success:
    record "Seduction by Proxy" in the Table of Events;
    say "Roxane, deeply moved by this sentiment, invites Christian up to her balcony. He scrambles up the ivy and disappears into her bedroom; the last thing you hear is a girlish giggle from above.";
    now Roxane is nowhere; now Christian is nowhere;
    move the player to Empty Street.

When Courting Roxane ends in failure:
    record "Ruining Christian's Chances" in the Table of Events;
    say "Roxane sighs heavily and goes back into her room, slamming the door behind her.

'Thanks very much,' says Christian to you, striding off down the street.";
    now Roxane is nowhere; now Christian is nowhere;
    move the player to Empty Street.

Sulky Ramble is a scene. Sulky Ramble begins when Courting Roxane ends in success. Sulky Ramble ends when the time since Sulky Ramble began is 2 minutes. When Sulky Ramble ends: record "Wandering the Streets, Sulking" in the Table of Events.

Every turn during Sulky Ramble:
    say "You find yourself kicking fenceposts quite without thinking about it."

Smug Ramble is a scene. Smug Ramble begins when Courting Roxane ends in failure. Smug Ramble ends when the time since Smug Ramble began is 2 minutes. When Smug Ramble ends: record "Wandering the Streets, Exultant" in the Table of Events; say "Of course, you will regret this soon enough."

Every turn during Smug Ramble:
    say "You find yourself smiling fiercely at the moon."

To record (occurrence - text) in (target table - a table name):
    choose a blank row in the target table;
    now the event entry is the occurrence.

Table of Events
event
"A Duel of Insults"
with 30 blank rows.

The plot summary rule is listed instead of the announce the score rule in the carry out requesting the score rules.

This is the plot summary rule:
    say "The Plot So Far: [paragraph break]";
    let act number be 0;
    repeat through the table of Events:
        increment act number;
        say " Act [act number]: [event entry][line break]".
```

```
Test me with "christian, ask roxane about love / christian, say your breath smells like ripe taleggio to roxane / score / z / z / score".
```

---

167 ★★★★ **Example** **Pine 4** *RB*

Pine: Adding a flashback scene that, instead of repeating endlessly, repeats only until the Princess has understood the point.

Suppose in addition to our pathetic little family history, we have another secret to convey to the Princess, this one a little more peculiar. She either gets it or she doesn't; once she gets it, we do not revisit that flashback, though it is still possible to keep visiting the poverty flashback.

```
"Pine"

Part 1 - The Set-up

A person can be asleep or awake. A person can be active or passive.

The Spinning Tower is a room. "A remote corner of the old castle, reserved for spinning and weaving tasks."

Sleeping Beauty is an asleep woman in the Spinning Tower. "[if asleep]Sleeping Beauty lies here, oblivious to your presence[otherwise]Sleeping Beauty stands beside you, looking [attitude][end if]." The description is "She is even more magnificent than the rumors suggested." Understand "woman" or "girl" or "princess" or "lady" as Sleeping Beauty.

Discovery is a scene. Discovery begins when play begins. Discovery ends when Sleeping Beauty is awake. Marriage Proposal is a scene. Marriage Proposal begins when Discovery ends.

When Discovery ends: say "Throughout the palace you can hear the other sounds of stirring and movement as the spell of centuries is broken."; now Beauty is passive.

Instead of waking an awake person: say "Redundant."

Instead of waking an asleep person: say "Yes, but how?"

Instead of waiting in the presence of an asleep person (called snorer): say "You are alone with the sound of [the snorer] snoring sonorously."

Instead of attacking an asleep person:
    now the noun is awake;
    say "[The noun] sits bolt upright. 'Hey! Ow!' So much for that true love's kiss nonsense."

Instead of kissing an asleep person:
    now the noun is awake;
    say "[The noun] slowly stirs to wakefulness!"
```

Instead of throwing water at an asleep person:
    now the second noun is awake;
    now the noun is nowhere;
    say "You pour out [the noun] on [the second noun].

[The second noun] wakes, shuddering. 'Agh! I had a terrible dream about drowning and then-- Hey!'"

The player carries a jug of water. Understand "pour [something] on [something]" or "splash [something] at/on [something]" as throwing it at.

Table of Conversation

topic	reply	quip
"dream/dreams/nightmare/nightmares/sleep"	"'Sleep well?' you ask solicitously.	

'Not really,' she replies, edging away from you. So much for that angle."	"'Ghastly nightmares,' she remarks. You nod politely."
"marriage/love/wedding/boyfriend/beau/lover"	"'So,' you say. 'This is a little weird since we just met, but, um. Would you like to get married?'

She looks at you nervously. 'Do I have to? I mean, I'd rather not.'

Well, this could get prickly fast."	"'I, er,' she says. 'I hope I'm not supposed to marry you or something.' Uh oh."
"marriage/love/wedding/boyfriend/beau/lover"	"'I was told I was going to marry you and inherit the kingdom,' you say, apologetically. 'Would that be very bad?' This could be awkward, considering your family circumstances -- you did promise your mother that everything would be better, after this --

'Oh, it's not you -- I'm seeing someone,' she says, smiling quickly.

You try to think how to point out that it's been a hundred years since she last saw her boyfriend."	"'Do you think I could go look for someone? I'm seeing him, you see, and I think I've been... sick... for a while, so he might be worried.'

You try to think how to point out that it's been a hundred years since she last saw her boyfriend. And try not to think how awkward things would be in your family if she refuses to marry you."

"marriage/love/wedding/boyfriend/beau/lover"	"'Do you think you could consider alternatives if he's no longer interested in you?' you suggest.

She gives you the look of a wounded squirrel. 'My father might not approve of my love for the kitchen boy, but his heart is faithful and true!' she exclaims.

'Right; supposing that he's still around, I'm sure that his love won't have faded,' you say, considering your fingernails. Maybe you'd better come clean with her about your identity, after all: she might be more favorably inclined if she understood that you won't interfere in her base-born romances."	"'I don't expect you to understand,' she says in a low whisper. 'I know it is not considered proper for a princess to love -- and such a one as my William, who works in the kitchen --' Her glance dares you to laugh. '-- but I cannot marry you without telling you this truth.'

Right then. Perhaps you'd better tell her your secret, in exchange?"

"marriage/love/wedding/boyfriend/beau/lover"	"'You've been up here for a hundred years,' you say. An unpleasant thought occurs to you. 'Was your young man in the castle somewhere?'

She shakes her head mutely."	"She goes to the window and looks out at the now-fading thicket of briar. 'That took a while to grow,' she observes. 'I've been up here longer than I thought.'

```
You shrug, uncomfortable."

Instead of asking an awake beauty about a topic listed in the Table of Conversation when Marriage Proposal is happening:
    now Beauty is passive;
    say "[reply entry][paragraph break]";
    blank out the whole row.

Instead of telling an awake beauty about something: try asking the noun about it.

Instead of asking an asleep person about something:
    say "[The noun] snores."

Marriage Proposal ends in failure when the number of filled rows in the Table of Conversation is 0.

Every turn during Marriage Proposal:
    if Beauty is active and Beauty is visible:
        repeat through Table of Conversation:
            say "[quip entry][paragraph break]";
            blank out the whole row;
            make no decision.

When Marriage Proposal ends in failure: end the story saying "This is going to take some explaining."

Part 2 - Flashbacks

Instead of asking an awake beauty about a topic listed in the Table of Flashback Material:
    now Beauty is passive;
    say "[reply entry][paragraph break]".

A fact is a kind of thing. A fact can be known or unknown. A fact can be current or past.

The family circumstances is a fact. The secret identity is a fact. The printed name of secret identity is "secret".

Table of Flashback Material
```

topic	reply
"poor/poverty/family/money/mother/circumstances" or "family circumstances" or "my family/mother"	"[if family circumstances is unknown]'I wish you'd give some thought to marrying me. You see,' you say, your jaw tensing. 'I wouldn't ask if it weren't for my [family circumstances]...'[otherwise]'I don't think you fully understand the [family circumstances],' you say.[end if]"
"secret/identity/gender/girl/female/woman" or "secret identity" or "my secret" or "my secret identity" or "my gender"	"[if dramatic revelation ended in failure]'Look,' you say, trying again. 'Pay attention: I need you to understand my [secret identity].'[otherwise]You clear your throat and allow your voice to stray upward, into its natural register and out of this husky false tenor you've been affecting. 'There's, er, something you should know about my [secret identity],' you say...[end if][if dramatic revelation ended in success] She looks impatient. 'I get it, you know,' she says. 'I'm not stupid.'"

```
After printing the name of a fact (called target): now the target is current; now the target is known.
```

Poverty flashback is a recurring scene. Poverty flashback begins when family circumstances is current. When poverty flashback begins: strip the player; move the player to the Woodcutter's Shack.

The Woodcutter's Shack is a room. "Your family lives in a shack in the forest. There are holes in the roof, and in the winter the snow comes in -- rain, too, for that matter. The walls aren't very well-boarded, and don't keep out the wind, and even though you live in the middle of dense woods, you can never gather enough fuel to keep this place fully heated. And then there's the stench. Pigs wander freely in and out, and your three youngest brothers play on the floor."

Pigs are an animal in the shack. The pigs are scenery. The description is "They really are very grubby, dirty animals, but what's worse than that, the value of pigs has declined a lot over the last few decades. This is hard to explain to someone who has been out of touch with the world for a while, but keeping pigs for meat is a dubious prospect when there's less and less for them to forage on." Instead of smelling the pigs: say "They smell the way animals do, when they live among their own refuse."; increase the pity of Beauty by 2.

The brothers are a man in the shack. The brothers are scenery. The description of brothers is "Hans, Franz, and Lukas. Twins and then the baby. So young, and growing up fatherless; and soon to be orphaned entirely, if your mother's health does not improve." Understand "brother" or "twin" or "twins" or "baby" or "franz" or "hans" or "lukas" as the brothers.

The untidy bed is scenery in the Shack. Mother is a woman on the untidy bed. The description of mother is "She is wasting away of a slow disease, her skin stretched tautly over bone. She hasn't been the same since your father left." On the bed is a folded letter.

The description of the letter is "Many times read over and creased, the letter explains how your father has gone away with a wealthy countess and will not return. Your mother was not able to read it herself, of course, and had to have it explained to her by the parish priest. Now she keeps it by the bed and crumples it in her fits of delirium."

Instead of kissing or touching Mother for the first time:
    say "You place a gentle kiss on her feverish brow. She looks up at you, her oldest -- yes, never mind that bit -- with a look of sincere trust and admiration.

'You'll find a way through this for us,' she says, squeezing your fingers. 'I know you will.'"; increase the pity of Beauty by 3.

Instead of kissing Mother: say "You have no more heart-rending memories of affection to recount; that one incident will have to serve, for rhetorical purposes."

Instead of waiting in the Shack: say "The wind blows sharply through the walls."

Instead of attacking someone in the Shack:
    say "Though sometimes the conditions of your life make you grouchy and impatient, you would never dream of striking a member of your own family. But from time to time you do feel the temptation."

Beauty has a number called pity. After examining something in the Woodcutter's Shack, increment the pity of Beauty. After examining mother, increase the pity of

```
Beauty by 2. After examining the letter, increase the pity of Beauty by 3.

Poverty flashback ends when waiting or the time since poverty flashback began
is five minutes.

When Poverty flashback ends:
    now family circumstances is past;
    say "...you finish describing the miserable circumstances of your home life,
and allow your attention to return to the present.";
    restore the player;
    now Beauty is passive;
    if Beauty is clever and Beauty is sympathetic:
        say "'I understand,' she says slowly. 'Yes, I do. I'll do it.' She takes a deep
breath and looks at you. 'We will be king together! and your family will be
royalty!'";
        end the story finally;
    otherwise:
        if Beauty is sympathetic, say "'Oh dear!' she says. 'What a dreadful life!'
She wrings her hands. 'No wonder you are eager to improve your lot...! But --'
Her brow clears, a new thought occurring. 'You needn't marry me, you know! We
could arrange it differently! I am certain that my father would give you a large
reward, instead, and then I would not be separated from my current boyfriend!'";
        otherwise say "She makes a disgusted face, but she doesn't seem nearly
so heart-wrung as you had hoped to make her. Tough audience, these modern
princesses."

Definition: Beauty is sympathetic if the pity of Beauty is greater than 4.

To say attitude:
    if Beauty is sympathetic, say "distressed on your behalf";
    otherwise say "a little confused".

Saved location is a room that varies. Locker is a container. Wardrobe is a
container.

To strip the player:
     now every thing carried by the player is in the locker;
    now every thing worn by the player is in the wardrobe;
    now saved location is location.

To restore the player:
    now every thing carried by the player is in the location;
    now every thing in the locker is carried by the player;
    now every thing in the wardrobe is worn by the player;
    move the player to saved location.

Part 3 - The Other Secret
```

This time, we're waiting for the princess either to understand or not understand -- so we don't want to rerun the scene once it has happened successfully.

```
Beauty has a number called clue count.

Dramatic revelation is a recurring scene. Dramatic revelation begins when
attempting confidence.
```

```
To decide whether attempting confidence:
    if dramatic revelation ended in success, no;
    if secret identity is current, yes;
    no.

When dramatic revelation begins:
    strip the player;
    say "You reminisce about one of the many stops on the way here: you had a long journey from your homeland, and it wasn't made any easier by your poverty, the inability to afford decent inns or plentiful food or any kind of ride along the way.";
    move the player to the Forest Clearing;
    move the pack to the player; now the player wears the trousers; now the player wears the shirt.

Forest Clearing is a room. "It's mid-autumn in your memory, the pool clear and cold, gold and red leaves floating on the surface."

The pool is scenery in the Clearing. Understand "reflection" or "surface" or "water" as the pool. "The pool is cold but beautiful, and the stopping place a welcome rest." The leaves are scenery in the clearing. The description is "Bright gold and orange and red: it's been a sharply chilly autumn, as you have reason to know in detail."

The trousers and the shirt are wearable things. The pack is a container. The pack contains ale, food, and skirt. A distraction is a kind of thing. The ale, the food, the pair of trousers, and the shirt are distractions. The description of a distraction is usually "[The item described] is not the point of this story." The shirt and the trousers are wearable. The description of the trousers is "Borrowed from your oldest brother, who is only a year younger than you. They are too long for your legs and overly snug at the hip, but no one around here pays much attention to fashion, and you're getting away with it, more or less." After examining the trousers, increment the clue count of Beauty.

Instead of examining the player during dramatic revelation:
    increment the clue count of Beauty;
    say "You cannot see yourself without reflection, but you can feel your hair loose and unbound over your shoulders."

Rule for printing the name of the skirt while taking inventory: say "one skirt you have not been able to bring yourself to part with". The description of the skirt is "Made for you by your mother, and it looks quite pretty on you. If your primary plan does not work, you may be forced to wear it again, and hope to catch a male eye... but with luck that will not be necessary." After taking inventory: increment clue count of Beauty.

Swimming is an action applying to nothing. Understand "swim" or "dive" as swimming.

Instead of swimming in the presence of the pool:
    increment clue count of Beauty;
    say "You consider going for a swim, but don't dare be caught unclad and unarmed, not here. There are too many men around, and any of them discovering you here would surely take advantage."
```

```
Instead of searching or drinking the pool:
    increment clue count of Beauty;
    say "You lean over the pool and look carefully at your reflection, your hair
loose and unbound, falling around your face in waves. (That should surely give it
away!)"

Instead of waiting during dramatic revelation: say "You wait for the penny to
drop, for her to understand."

Dramatic revelation ends in failure when waiting or the time since dramatic
revelation began is five minutes. When dramatic revelation ends in failure:
    now secret identity is past;
    restore the player;
    now Beauty is passive;
    say "She wrinkles her nose. 'I don't understand!' she says. 'What are you
trying to tell me?'

    You could weep for womankind. But you don't quite dare spell it out in so
many words, not when someone might come up the stair and overhear a chance
revelation."

Dramatic revelation ends in success when Beauty is clever. When dramatic
revelation ends in success:
    restore the player;
    now Beauty is passive;
    say "'You're -- a girl? Like me?'

'Not much like you,' you say, glancing over her petite frame and pert nose. 'But
female, at any rate.'"

Definition: Beauty is clever if the clue count of Beauty is greater than 2.
```

And now, since we don't really want to return to the rest of the 'marriage proposal' scene once she has learned our ID:

```
Marriage proposal ends in distraction when Dramatic Revelation ends in
success.

Compromise proposal is a scene. Compromise proposal begins when Dramatic
Revelation ends in success. When Compromise Proposal begins: now Beauty is
passive.

Instead of asking an awake beauty about a topic listed in the Table of Secondary
Conversation when Compromise Proposal is happening:
    now Beauty is passive;
    say "[reply entry][paragraph break]";
    blank out the whole row.

Every turn during Compromise Proposal:
    if Beauty is active and Beauty is visible:
        repeat through Table of Secondary Conversation:
            say "[quip entry][paragraph break]";
            if the number of filled rows in the Table of Secondary Conversation is
greater than 1, blank out the whole row;
            make no decision.
```

```
Every turn: now Beauty is active.
```

Notice that we moved the re-activation rule down here so that the Compromise Proposal rule would fire first. There are other more complicated ways of handling order of every turn rules than by relying on text sequence alone; but we will save that for a later chapter. For now it is sufficient to depend on the order in which the rules are declared.

```
Table of Secondary Conversation
```

topic	reply	quip
"girls/me/women/female/truth/identity"	"'Marrying me would be no interference,' you go on. 'You could carry on whatever romances you wished, without your father noticing.' (Probably. You'll let the pragmatic details of this work themselves out later, and hope that any children she has will look vaguely like you.)"	"'Girls can't rescue people.'

```
'Wrong,' you say, feeling a little annoyed. 'But you see why marrying me wouldn't be an interference. You could carry on whatever romances you wished, without your father even noticing.'"
```

"king/man"	"'If you're thinking that a woman can't be the prince -- and then king -- well, there was a woman Pope, once.'
She looks awed."	"The crease in her forehead does not go away. 'But if everyone thinks you are a man... later you would be king!'

```
Before she can go on, you say, 'There was a woman Pope, once. Compared to that, a woman king is nothing.'"
```

"decision/proposal/marriage/choice"	"'So,' you say. 'What do you think?'
[final decision]"	"Her pretty nose twitches again, which you are coming to recognize as a sign of hard mental labor. 'I think I see,' she says. [final decision]"

```
To say final decision:
    if Beauty is sympathetic:
        say "She considers the matter silently for some minutes, then says: 'I will do it. My beloved William will be so glad!' You imagine that William's feelings on the matter will be a tad more complex than that, but do not bother quashing her exuberance...";
        end the story finally;
    otherwise:
        say "'I still don't quite understand why you would want this so badly as to go to all that trouble,' she admits uneasily. Evidently you have not explained enough to her about the poverty of your home life."

Test me with "x beauty / pour water on beauty / ask beauty about sleep / tell beauty about poverty / smell pigs / x mother / x letter / kiss mother / ask beauty about marriage / tell beauty about identity / x me / look in water / i / z / ask beauty about marriage".
```

168

★★★ Example Cheese-makers

RB

Scenes used to control the way a character reacts to conversation and comments, using a TALK TO command.

As we have seen, there are a number of different ways of controlling conversation in interactive fiction, and the best choice of way will depend quite a lot on what kind of work we're writing.

One common model is to replace Inform's default ASK and TELL commands with a TALK TO command. This gives the player less control than he would otherwise have: instead of asking a character about any topic under the sun, he's restricted to seeing (or not seeing) a single sequence of text that the author has written in advance. On the other hand, such a system is harder for the player to break (since he can never ask about a topic that the author hasn't implemented), and easier for the author to tie into plot developments. If we give TALK TO different output at each scene, we get conversation that is always tied to the current state of the plot.

This is a design approach that works best in a game with a large number of short, focused scenes. For other kinds of conversation system design, compare the other examples listed in the Recipe Book.

```
"The Cheese-makers" by Phrynichus.

Chapter 1 - Replacing old talk commands and making a new one
```

Here, using some techniques that will be discussed in the chapter on Understanding, we get rid of Inform's default handling of ASK and TELL, and create our own TALK TO action instead:

```
Understand the commands "ask" and "tell" and "say" and "answer" as something
new.

Understand "ask [text]" or "tell [text]" or "answer [text]" or "say [text]" as a
mistake ("[talk to instead]").

Instead of asking someone to try doing something:
   say "[talk to instead][paragraph break]".

Instead of answering someone that something:
   say "[talk to instead][paragraph break]".

To say talk to instead:
   say "(To communicate in [story title], TALK TO a character.) "

Understand "talk to [someone]" as talking to. Understand "talk to [something]" as
talking to. Talking to is an action applying to one visible thing.

Chapter 2 - Specific scenes and talking
```

Now, suppose we have a situation -- say, a stage play -- in which it is appropriate to talk to different characters at different times. During the prologue of the play, no one else is on-stage, and the player is to address the audience directly:

```
Section 1 - Prologue
```

```
When play begins:
    now right hand status line is "416 BC";
    now left hand status line is "[location]".

Prologue is a scene. Prologue begins when play begins.

The Theater of Dionysus is a room.

The audience is a person in the Theater. "The usual audience looks on: the priests and judges in the front row, and then Athenians, metics, and foreigners." The audience can be prepared or unprepared. The description is "Have you ever seen such a company of perjurers, pathics, and thieves?" Understand "priest" or "priests" or "priest of dionysus" or "judge" or "judges" or "athenians" or "metics" or "foreigners" as the audience.

Instead of talking to the player when the Prologue is happening:
    say "There will be plenty of occasion for muttered asides later in the play, but for now you must prepare the audience for things to come."

Instead of talking to the audience when the Prologue is happening:
    say "Drawing breath, you turn to the audience, and offer them a genial, witty, colorful, and of course crude synopsis of what they are about to see; describing all the characters in unmistakable terms and not omitting the most important of them all, your august self.";
    now the audience is prepared.

Instead of talking to the audience when the Prologue has happened:
    say "You may only direct monologues to the audience when the other actors are off-stage. Otherwise, their characters might have to notice."

Prologue ends when the audience is prepared.
```

But there might follow a scene in which the player shouldn't talk at all:

```
Section 2 - Parodos

Parodos is a scene. Parodos begins when Prologue ends.

When Parodos begins:
    move the chorus to the theater.

Instead of talking to someone during Parodos:
    say "Sssh: this moment belongs to the chorus. They've worked so hard on it, after all."

Parodos ends when the time since Parodos began is 4 minutes.

The chorus is a person. The description is "They are dressed in exaggerated rural costume and feminine masks, as they are meant to represent a company of female cheese-makers from the Spartan-occupied deme of Dekeleia." Understand "cheesewives" or "cheese-makers" or "chorus-leader" as the chorus.

Every turn during Parodos:
    repeat through Table of Choral Events:
        say "[output entry][paragraph break]";
        blank out the whole row;
        make no decision.

Table of Choral Events
output
"The chorus now begins its entry, accompanying with anapestic song its march
up the eisodos."
"The chorus draws nearer, stomping and clomping and swinging their baskets of
cheese."
"You stand aside as the chorus fills the orchestra and dances to and fro."
"The tune of the aulos-player grows more and more frenzied and then breaks
off."
```

This last rule is a refinement borrowing from the Activities chapter, which gives characters different appearances in room descriptions depending on when we happen to look; because of the action of the play, we want to show the chorus and audience doing different things during different scenes.

```
Rule for writing a paragraph about the chorus during Parodos:
    say "The chorus are dancing and singing their way[if the time since Parodos
began is less than 3 minutes] up the long walkways onto the stage[otherwise]
into position in the orchestra[end if]. [The audience] appear to be pricing their
costumes to the nearest obol: woe to the producer who cheats them of their due
share of spectacle."
```

And now a scene in which the player can talk several times to a character (Heracles) but has no useful dialogue with the chorus, the audience, or himself. The prohibition from talking to the audience after the Prologue is already written, but we'll supply some appropriate responses for talking to the player or the chorus during this scene:

```
Section 3 - Episode

Episode is a scene. Episode begins when Parodos ends.

When Episode begins:
    move Heracles to the theater;
    say "The chorus falls silent, which is the cue: Heracles bursts out of the scene
building."

Heracles is a man. The description is "Hard to mistake in his lion skin and boots,
and carrying a formidable club." Heracles wears a lion skin and boots. He carries
a formidable club. Heracles can be placid or annoyed. Heracles is placid.
Heracles can be satisfied, intrigued, or unsatisfied. Heracles is unsatisfied.

Instead of talking to the chorus during Episode:
    say "Your improvised flirtation with the chorus raises no response but a crude
gesture from the chorus-leader, who seems to be modeling the role on Iambe."

Instead of talking to the player during Episode:
    if Heracles is annoyed:
        say "You mutter to yourself about men with more appetite than brain. The
actor playing Heracles ignores you, but it's good odds he's scowling under his
mask. He hates it when anyone but himself ad-libs for attention.";
```

```
    otherwise:
        now Heracles is annoyed;
        say "'By the dog, he'll eat me if he gets a chance,' you mutter aside.
[paragraph break]'What's that you say, my ignoble friend?' demands Heracles,
hefting his club. He's not entirely joking: you've left the script just now."

Instead of talking to Heracles when Heracles is unsatisfied during Episode:
    say "'Dear Heracles, friendly Heracles,' you begin, cringing out of the way as
he responds with one of his affectionate ox-killing punches to the shoulder.
[paragraph break]But Heracles falls still, and looks almost thoughtful, as tell him
you know how he may rout the Spartans, woo all twenty-four lactic ladies, and
tame his savage gut with a bathtubful of porridge. [paragraph break]'Speak on,
little man,' he says.";
    now Heracles is intrigued.

Instead of talking to Heracles when Heracles is intrigued during Episode:
    say "It takes several exchanges for him to wrap his one-inch brain around
your ten-inch plan; but in the end he embraces the scheme, the women, and
your humble self.";
    now Heracles is satisfied.

Every turn when not talking to someone during Episode:
    repeat through Table of Episodic Events:
        say "[output entry][paragraph break]";
        blank out the whole row;
        make no decision.

Table of Episodic Events
output
"With a fart and a roar, Heracles asks the world at large, and you in particular,
where his dinner might be."
"In epic diction, Heracles invites the dairy-mistresses, whey-matrons, and
concubines of curd to supply him a supper from their ample baskets."
"Heracles and the chorus banter about the proclivities of cheese-wives. The
chorus suggest that Heracles, as a son of Zeus, must know something about the
appetites of which they speak."
"Heracles boasts that a man like himself can perform any feat, but only when his
belly is full. Coyly, the matrons prance and dance, skip and gambol out of his
grasp, singing mockingly about heads of garlic and loaves of sesame-crusted
bread."
"The song of the feta fanciers now turns to pots of honey and new-made wine,
borrowing verses from last year's Lenaia winner, 'The Bees'. With a jolt, you
realize that you've missed your cue and the chorus are filling in for you."
"Playing for time, the chorus-leader elaborates a whole banquet: rabbit stew,
shanks of lamb, spitted quails, eels from lake Copais. Heracles looks as near
swooning as any girl fresh from Brauron."
"The chorus-leader extends the list of delicacies to include ox-brains, ham-
hocks, barley, mullet, carrots, pigeons, lentils, radishes, peas, and apples both
wine-dark and golden. The audience shifts on the benches. An expression of
gloom settles over the Priest of Dionysus in the front row."
"Inspired by Euripides['] own Muse, the chorus-leader invents a mock-Alcaean
hymn on the merits of chervil. This is clearly his swan-song: if you don't speak at
last, the play will come to a halt."
"Silence descends."
```

```
Rule for writing a paragraph about Heracles during Episode:
    say "[Heracles] stands at the center of the orchestra, with members of [the
chorus] ranged on either side. [paragraph break][The audience] appear to be
reserving their judgement, though they show signs of restiveness at the usual
jokes: must there be a Heracles in [italic type]every[roman type] play?"

Episode ends successfully when Heracles is satisfied.

When Episode ends successfully:
    say "That, of course, is your cue: you're to come back on as Pan thirty verses
from now, and it takes time to put on the hooves and the woolly-legged
trousers.";
    end the story saying "You exit".

Episode ends disastrously when the number of filled rows in the Table of
Episodic Events is 0.

When Episode ends disastrously:
    end the story saying "The production has crashed to a halt".

Test me with "ask audience about me / tell audience about me / audience, hello /
audience, jump / talk to me / talk to audience / g / talk to chorus / look / x
heracles / talk to me / talk to audience / z / look / talk to heracles / g".
```

Chapter 11: Phrases

↖ Contents of *Writing with Inform*
← Chapter 10: Scenes
→ Chapter 12: Advanced Actions
↘ Indexes of the examples

§11.1. What are phrases?

Phrases are instructions to Inform to do something, or to decide whether something is true or false, or to produce a value, or to say something. Inform has around 350 phrases built-in, and the chapters so far have already defined about 100 of those. In this chapter we'll see some key phrases for organising instructions of what to do, and also see how to define entirely new phrases.

Just to run through the four sorts of phrase with examples:

(a) Phrases to do something. These are the ones used in the body of a rule. For example,

```
When Train Stop begins:
    move the Flying Scotsman to the Station;
    say "The Flying Scotsman pulls up at the platform."
```

Rules like this begin with a "preamble", the beginning part which tells Inform when or how they apply, and then follow on with a list of instructions - here, just two of them. "move ... to ..." and "say ..." are both phrases. Inform provides about 130 of these built-in. It's actually not quite true that they all do something, because one of them is:

do nothing

This phrase does nothing at all. It is very occasionally useful to make a rule which does nothing:

```
This is the largely ineffective rule:
    do nothing.
```

(b) Phrases to decide whether a condition is true. These are the ones which can be used in an "if":

```
if action requires light: ...
```

Not all conditions come from phrases. For example, "if the front door is closed" and "if Peter is wearing the sandals" have meanings which come from the verbs "to be" and "to wear". Inform provides about 60 built-in conditions, which give a friendly wording for questions which would be lengthy or difficult to write in any other way.

(c) Phrases to decide a value. For example:

```
square root of 16
```

produces a number, 4 of course, and can be used whenever a number is expected. Inform provides about 100 built-in phrases like this.

(d) Text substitutions. These are actually just phrases whose definition begins with "To say ...". Example:

```
"It's now [time of day in words]."
```

Inform provides about 60 built-in text substitutions.

⬆ Start of Chapter 11: Phrases

⬅ Back to Chapter 10: Scenes: §10.9. Why are scenes designed this way?

➡ Onward to §11.2. The phrasebook

---

§11.2. The phrasebook

The Phrasebook is Inform's collection of recognised phrases, and it can always be browsed using the Index panel of the same name. Even the smallest project has a good-sized phrasebook, since it contains all of the built-in phrases. But most projects also define new phrases of their own.

Here is a simple definition of a new phrase:

```
To spring the trap:
    say "'Sproing!' go the hinges and, with a flash of silver, the enormous blades whisk together!";
    end the story.
```

Inform allows us to use whatever conventions of layout we prefer, but it's customary to use indentation like this, dividing off the preamble from the phrases which follow. As can be seen, definitions of new phrases look very like rules.

What makes this definition a simple one is that the wording is fixed. The only way to use this would be from another phrase or rule, like so:

```
Instead of entering the cage:
    spring the trap.
```

In the next section we'll see how to give more complicated definitions which, like "move ... to ...", allow for the wording to change with the circumstances.

Start of Chapter 11: Phrases
Back to §11.1. What are phrases?
Onward to §11.3. Pattern matching

§11.3. Pattern matching

In this section, let's make the following new phrase:

```
To admire (item - an object):
    say "You take a long look at [item].".
```

This does very little, of course, but it does allow the wording to be different each time the phrase is used:

```
admire the diamonds;
admire Mr Cogito;
admire the honey sandwich;
```

and our single definition covers all of these possibilities. The bracketed part of the definition, "(item - an object)", tells Inform to expect an object in that position, and Inform enforces this carefully. So this definition might tell Inform what "admire the barricade" means, but not what

```
admire "blue cheese";
admire 63;
```

mean. Unless some other definition sorts the matter out, Inform will reply to uses like this with a Problem message:

```
Problem. You wrote 'admire 63' , but '63' has the wrong kind of value: a number rather than an object.
```

The object does not need to be named literally, but can be anything which works out to be an object: for instance,

```
After dropping something in the Auction House:
    admire the noun.
```

which Inform allows because "noun", here, is a name for the object which is being acted on.

Inform decides which definition to apply in a process called "pattern matching".

The bracketed part of the example definition has the form "(name - kind)". The definition only applies if the text supplied agrees with the "kind" part - for instance, the diamonds agreed with "object", but 63 did not. If the definition does apply, then the Inform works through the rest of the phrase using "name" to mean whatever value matched. For example:

```
To slam shut (box - an open container):
    say "With great panache, you slam shut [the box].";
    now the box is closed.
```

When this phrase is followed, "box" means whatever open container the pattern-matcher found when it was called for. For example, if Inform reads

```
slam shut the Dutch armoire;
```

then it acts on this by following the definition of "slam shut ...", using the Dutch armoire object as the value of "box", so it prints:

```
With great panache, you slam shut the Dutch armoire.
```

and renders it closed.

In fact any description can be given in the definition, and that includes a single, specific value. For instance, we could define:

```
To grant (bonus - a number) points:
    increase the score by the bonus.

To grant (bonus - 7) points:
    say "You shiver uncontrollably."
```

which would withhold this unlucky bounty. That would mean that:

```
grant 7 points;
grant seven points;
```

would each produce uncontrollable shivers, because Inform uses the definition applying to the number 7; but

```
grant six points;
```

would increase the score by 6. In general Inform always follows the principle that more specific definitions take priority over more general ones. So although the definitions:

```
To grant (bonus - a number) points: ...
To grant (bonus - 7) points: ...
```

both apply to the case of "grant 7 points", Inform uses the second, because it's the more specific of the two possibilities.

Sometimes it will not be possible to tell if the value supplied meets the requirements until the story is actually playing. If, at run-time, no definition fits some phrase which has to be carried out, a run-time problem message is produced.

Finally, and more straightforwardly, we can specify variations in wording using slashes between alternative words in a "To ..." definition. For instance:

```
To grant (bonus - a number) point/points: ...
```

allows the final word to be either "point" or "points". Slashes like this can only be used with literal words, not bracketed values, and give alternative forms only of a single word at a time; the alternative "--" means "no word at all", and thus makes it optional:

```
To grant (bonus - a number) point/points/--: ...
```

makes "grant 3" do the same as "grant 3 points".

If we need more variation than that, we should make more than one definition.

Start of Chapter 11: Phrases

Back to §11.2. The phrasebook

Onward to §11.4. The showme phrase

Example 169: ★ **Ahem** Writing a phrase, with several variant forms, whose function is to follow a rule several times.

Example 170: ★★ **Ferragamo Again** Using the same phrase to produce different results with different characters.

§11.4. The showme phrase

We've already seen the SHOWME command, which can be typed into the Story panel to look at the state of something, usually a thing or room. SHOWME is a testing command which has no effect once the work is released; eventual players can't use it.

Inform also has a phrase called "showme", which works in much the same way:

showme (value)

This phrase is intended for testing purposes only. If used in a story file running inside the Inform application, it prints a line of text showing the given value and its kind; in a Released story file, it does nothing at all. Example:

```
When play begins: showme 11.
```

produces

```
number: 11
```

More usefully:

```
Every turn: showme the score.
```

Now, every turn, we get a line in the story's transcript like so:

```
"score" = number: 0
```

Inform uses the quotation marks and equals sign to show that it had to do some work to find the answer. "score" wasn't a constant value - it was a variable, and Inform had to look up the current value.

"showme" is a convenient way to see what's going on inside a phrase which isn't behaving as expected, or to find out the kind of a value. Here are some trickier examples. Suppose our design includes:

```
The matching key of the blue door is the brass Yale key.
```

If we then try this:

```
When play begins:
    showme matching key of the blue door.
```

we get, when the story starts up,

```
"matching key of the blue door" = object: brass Yale key
```

Why is this an "object", when we know that the key is actually a "thing"? After all, if we "showme key" instead, we get:

```
thing: brass Yale key
```

The answer is a little technical: it's because Inform guarantees that the matching key is always an object, but not that it's always a thing - it just happens to be a thing at the moment. There's not really a contradiction, because a "thing" is a kind of "object", so in fact the key is both. If we try "showme matching key", we get something like this:

```
objects valued property: property 23
```

which is even more technical - people never need to print the names of abstract property names during play, so Inform doesn't provide any good way of doing it. It is reduced to printing out an internal ID number ("property 23") instead of the name ("matching key"). This can't be helped: "showme" is a way to lift the lid and see what's going on inside Inform's machinery, and some of the corners are dark.

All the same, "showme" can be very useful in tinkering with rules to make them work properly. It prints nothing at all in a Release version of a project, so it's impossible for these private notes to be shown accidentally to our eventual readers.

Start of Chapter 11: Phrases
Back to §11.3. Pattern matching
Onward to §11.5. Conditions and questions

§11.5. Conditions and questions

A variety of "conditions" have already appeared in this documentation. A condition is a phrase which describes a situation which might be true, or might be false, and examples might include:

```
Mr Kite is in Bishopsgate
the score is greater than 10
Sherlock Holmes suspects a woman
```

These are all examples of sentences, formed by putting nouns either side of a verb, and clearly a wide range of conditions can be written this way. But there are also a few special conditions built into Inform which have a fixed wording, and test questions difficult to address with ordinary sentences. For instance:

if in darkness:

This condition is true if the player currently has no light to see by. Note that the test is more complicated than simply testing

```
if the player is in a dark room, ...
```

since the player might have a torch, or be inside a cage which is itself in a dark room, and so on.

Another example of a condition not easily written as a sentence is:

if player consents:

This condition is unusual in doing something and not simply making a silent check: it waits for the player to type YES (or Y) or NO (or N) at the keyboard, and then is true if the answer was yes. Example:

```
say "Are you quite sure you want to kiss the Queen? ";
if the player consents:
    ...
```

Whether it's put to the player like this or not, testing a condition is really asking a question, and there is always a yes/no answer. In Inform this answer is not usually a value (unlike in some other computer programming languages), but it can be made into one.

Firstly, we need a special kind of value to hold answers like this. It's called "truth state", and it has just two possible values, written as "true" and "false". We then need:

whether or not (a condition) **... truth state**

This phrase converts a condition into its result as a value, which is always either "true" or "false". Example:

```
whether or not 20 is an odd number
```

produces the truth state "false". This is mostly useful for storing up results to look at later:

```
let victory be whether or not all the treasures are in the cabinet;
```

and then subsequently:

```
if victory is true, ...
```

As another example, in most stories this:

```
When play begins:
    showme whether or not in darkness.
```

...will produce a line:

```
"whether or not in darkness" = truth state: false
```

In short, "truth state" is a kind of value like any other. That means it can be the kind of a variable:

```
Salvation earned is a truth state that varies.
```

and it can similarly be used in table columns, lists, or anywhere else where values are allowed.

⬆ Start of Chapter 11: Phrases

⬅ Back to §11.4. The showme phrase

➡ Onward to §11.6. If

⬇ Example 171: ★★ **Proposal** Asking the player a yes/no question which he must answer, and another which he may answer or not as he chooses.

§11.6. If

Inform's most powerful phrases are those which control the others, making them repeat, or be skipped.

if (a condition) **,** (a phrase)

or:

if (a condition)**:**

This phrase causes the single phrase, or block of phrases, following it to be obeyed only if the condition is true. (If the condition must contain a comma for some reason, the block form should be used.) Example:

```
if the red door is open, say "You could try going east?"
```

The sense of an "if" can be reversed by using the word "unless" instead:

unless (a condition) **,** (a phrase)

or:

unless (a condition)**:**

This phrase causes the single phrase, or block of phrases, following it to be obeyed only if the condition is false. (If the condition must contain a comma for some reason, the block form should be used.) Example:

```
unless the red door is closed, say "You could try going east?"
```

"Unless" is clearly unnecessary, but it can be a good way to make the source text easier for humans to read.

As we have seen, there are many different forms of condition in Inform. They usually take a form quite like an assertion sentence, except that they're questions and not statements of fact. For example:

```
if the score is 10, ...
if all of the people are in the Atrium, ...
```

Questions like this are checked by Inform to see if they make sense. The following doesn't, for instance:

```
if 10 is a door, say "Huzzah!";
```

This produces the baffled reply:

Problem. In the line 'if 10 is a door, say "Huzzah!"', I can't determine whether or not '10 is a door', because it seems to ask if a number is some sort of door.

⬆ Start of Chapter 11: Phrases
⬅ Back to §11.5. Conditions and questions
➡ Onward to §11.7. Begin and end

§11.7. Begin and end

In practice it is not enough to apply "if" to a single phrase alone: we want to give a whole list of phrases to be followed repeatedly, or to be followed only if a condition holds.

We do this by grouping them together, and there are two ways to do this. One is as follows:

```
To comment upon (whatever - a thing):
    if whatever is transparent, say "I see right through this!";
    if whatever is an open door:
        say "Oh look, an open door!";
        if whatever is openable, say "But you could always shut it."
```

Here we group two phrases together under the same "if". Note that the comma has been replaced by a colon, and that the indentation in the list of phrases shows how they are grouped together. In the example above, the source moves two tabs in from the margin; the maximum allowed is 25.

Indentation is the convention used in this manual and in the examples, but not everybody likes this Pythonesque syntax. So Inform also recognises a more explicit form, in which the beginning and ending are marked with the words "begin" and "end":

```
To comment upon (whatever - a thing):
    if whatever is transparent, say "I see right through this!";
    if whatever is an open door
    begin;
        say "Oh look, an open door!";
        if whatever is openable, say "But you could always shut it.";
    end if.
```

(Pythonesque because it's a style popularised by the programming language Python, named in turn after "Monty Python's Flying Circus".)

Start of Chapter 11: Phrases

Back to §11.6. If

Onward to §11.8. Otherwise

Example 172: ★ **Matreshka** A SEARCH [room] action that will open every container the player can see, stopping only when there don't remain any that are closed, unlocked, and openable.

Example 173: ★ **Princess and the Pea** The player is unable to sleep on a mattress (or stack of mattresses) because the bottom one has something uncomfortable under it.

§11.8. Otherwise

We often need code which does one thing in one circumstance, and another the rest of the time. We could do this like so:

```
if N is 2:
    ...
if N is not 2:
    ...
```

but this is not very elegant, and besides, what if the action we take when N is 2 changes N so that it becomes something else?

Instead we use "otherwise":

otherwise if (a condition)

or:
otherwise unless (a condition)

or:
otherwise (a phrase)

or:
else if (a condition)

or:
else unless (a condition)

or:
else (a phrase)

This phrase can only be used as part of an "if ...:" or "unless: ...", and provides an alternative block of phrases to follow if the first block isn't followed. Example:

```
if N is 2:
    ...
otherwise:
    ...
```

When there is only a single phrase we can use the shortened form:

```
if N is 2, say "Hooray, N is 2!";
otherwise say "Boo, N is not 2...";
```

We can also supply an alternative condition:

```
if N is 1:
    ...
otherwise if N is 2:
    ...
otherwise if N is greater than 4:
    ...
```

At most one of the "..." clauses is ever reached - the first which works out.

If the chain of conditions being tried consists of checking the same value over and over, we can use a convenient abbreviated form:

if (value) **is:**

This phrase switches between a variety of possible blocks of phrases to follow, depending on the value given. Example:

```
if the dangerous item is:
    -- the electric hairbrush:
        say "Mind your head.";
    -- the silver spoon:
        say "Steer clear of the cutlery drawer."
```

One alternative is allowed to be "otherwise", which is used only if none of the other cases apply, and which therefore guarantees that in any situation exactly one of the blocks will be followed.

```
if N is:
    -- 1: say "1.";
    -- 2: say "2.";
    -- otherwise: say "Neither 1 nor 2.";
```

This form of "if" layout is not allowed to use "begin" and "end" instead of indentation: it would look too messy, and would scarcely be an abbreviation. It is also not allowed to use "unless" instead of "if", because the result would be too tangled to follow.

Start of Chapter 11: Phrases

Back to §11.7. Begin and end

Onward to §11.9. While

Example 174: ★ **Numberless** A simple exercise in printing the names of random numbers, comparing the use of "otherwise if...", a switch statement, or a table-based alternative.

§11.9. While

The next control phrase is "while", which has the form:

while (a condition)**:**

This phrase causes the block of phrases following it to be repeated over and over for as long the condition is true. If it isn't even true the first time, the block is skipped over and nothing happens. Example:

```
while someone (called the victim) is in the Crypt:
    say "A bolt of lightning strikes [the victim]!";
    now the victim is in the Afterlife;
```

We must be careful not to commit mistakes like the following:

```
while eggs is eggs:
    say "again and ";
```

which, as sure as eggs is eggs (which is very sure indeed), writes out

```
again and again and again and again and again and ...
```

forever. (Inform won't prevent this: we will find out the hard way when the story is played.) While we would probably never write anything so blatant as that, the mistake is all too easy to commit in disguised form. We should never design a loop, as repetitions like this are called, without worrying about if and when it will finish.

As with "if", we can use "begin" and "end" instead of a tabulated layout if we want to --

```
while ...
begin;
    ...
end while.
```

(The "begin" of an "if" must of course match an "end if", not an "end while", and so on.)

Experience shows that it is much more legible to lay out "while" loops as blocks, even in these rare cases when only a single phrase forms the body of the block.

Start of Chapter 11: Phrases

Back to §11.8. Otherwise

Onward to §11.10. Repeat

§11.10. Repeat

The other kind of loop in Inform is "repeat". The trouble with "while" is that it's not obvious at a glance when or whether the loop will finish, and nor is there any book-keeping to measure progress. A "repeat" loop is much more predictable, and is more or less certain to finish.

There are several forms of "repeat", of which the simplest is similar to the old FOR/NEXT loop from the home-computer programming language BASIC, for those with long memories:

repeat with (a name not so far used) **running from** (arithmetic value) **to** (arithmetic value)

or:

repeat with (a name not so far used) **running from** (enumerated value) **to** (enumerated value)**:**

This phrase causes the block of phrases following it to be repeated once for each value in the given range, storing that value in the named variable. (The variable exists only temporarily, within the repetition.) Example:

```
repeat with counter running from 1 to 10:
    ...
```

This, and runs through the given phrases ten times. Within those phrases, a special value called "counter" has the value 1 the first time through, then the value 2, then 3 and so on up to 10. (It can of course be called whatever we like: this is only an example.) The range can be from any kind where ranges make sense - anything on which arithmetic can be done, so for instance

```
repeat with moment running from 4 PM to 4:07 PM:
    ...
```

and also any enumeration:

```
Colour is a kind of value. The colours are red, orange, yellow, green, blue,
indigo and violet.

...
    repeat with hue running from orange to indigo:
        ...
```

We are allowed to "nest" loops, that is, to put one inside another.

```
To plot a grid with size (S - a number):
    repeat with x running from 1 to S:
        say "Row [x]:";
        repeat with y running from 1 to S:
            say " [y]";
        say "."
```

If we then write

```
plot a grid with size 5;
```

then the result is

```
Row 1: 1 2 3 4 5.
Row 2: 1 2 3 4 5.
Row 3: 1 2 3 4 5.
Row 4: 1 2 3 4 5.
Row 5: 1 2 3 4 5.
```

Thus the innermost phrase, the say which mentions "y", happens 25 times.

Whenever dealing with numbers in Inform we may need to remember that if the Settings for the project are set to use the Z-machine, the range is restricted to -32768 up to 32767. Repeating with a counter up to exactly 32767 is hazardous, because the counter can never break through this barrier: it's infinity, so far as Inform is concerned, and that can cause the repetitions to go on forever. (On Glulx, numbers can be very much larger.)

Start of Chapter 11: Phrases

Back to §11.9. While

Onward to §11.11. Repeat running through

Example 175: ★ **Wonka's Revenge** A lottery drum which redistributes the tickets inside whenever the player spins it.

§11.11. Repeat running through

Inform is not used very much for numerical work, so the kind of repeat loop described in the previous section is not much used. Inform's natural domain is really the world of things and rooms, so the following kind of repeat is much more useful.

repeat with (a name not so far used) **running through** (description of values)**:**

This phrase causes the block of phrases following it to be repeated once for each value matching the description, storing that value in the named variable. (The variable exists only temporarily, within the repetition.) Example:

```
repeat with item running through open containers:
    ...
```

If there are no containers, or they are all closed, the phrases will not be followed at all. Inform will issue a Problem message if the range of the loop may be infinite: for example, it won't allow:

```
repeat with X running through odd numbers:
    ...
```

On the other hand it will allow:

```
repeat with T running through times:
    ...
```

which repeats 1440 times, starting with T at midnight and finishing at 11:59 PM. See the Kinds index for which kinds of value can be repeated through.

As with counting the "number of ..." objects satisfying some property, we can run through a wide variety of possibilities - any description whose range is possible for Inform to search. For example:

```
repeat with dinner guest running through the people in the Dining Room:
    ...

repeat with possession running through things carried:
    ...

repeat with event running through non-recurring scenes which are happening:
    ...
```

The following lists the whereabouts of all men in lighted rooms:

```
repeat with suspect running through the men who are in a lighted room:
    say "[The suspect] is in [the location of the suspect].";
```

One small note of caution: if what the "repeat" loop does is to change the things being repeated through, changing in particular whether items not yet reached will qualify to be repeated through, the results can be unexpected. Rather than writing "repeat with X running through D", it may be safer to try "while there is D (called X)", though note that this will only finish if X is always changed so that it no longer qualifies.

⬆ Start of Chapter 11: Phrases

⬅ Back to §11.10. Repeat

➡ Onward to §11.12. Next and break

⬇ Example 176: ★★ **Strictly Ballroom** People who select partners for dance lessons each turn.

§11.12. Next and break

So "repeat" and "while" phrases cause a block of other phrases to be repeated, over and over. The number of repetitions and the flow of "control" has so far been controlled only by the way the original loop was described.

But in fact it's also possible to change this from inside the block being repeated, using these:

next

This phrase can only be used inside a "repeat" or "while" block, and causes the current repetition of the block to finish immediately. That either means the next repetition begins, or (if we are already at the last one) the loop ends too. Example:

```
repeat with X running from 1 to 10:
    if X is 4, next;
    say "[X] ".
```

produces the text "1 2 3 5 6 7 8 9 10 ", with no "4" because the "say" phrase was never reached on the fourth repetition.

In Monopoly terms, "next" is "Advance to Go" rather than "go directly, do not pass Go, do not collect $200" - the next iteration begins with the variable, if there is one, having cleanly moved on to the next value, just as if the loop had been run through in the normal way. ("Next" is called "continue" in a fair number of programming languages, so Inform issues a specific problem message to help people who forget this.)

break

This phrase can only be used inside "repeat", "while" block, and causes both the current repetition and the entire loop to finish immediately. Example:

```
repeat with X running from 1 to 10:
    if X is 7, break;
    say "[X] ".
```

produces the text "1 2 3 4 5 6 ", with nothing after "6" because the loop was broken at that point. The "say" wasn't reached on the 7th repetition, and the 8th, 9th and 10th never happened.

Start of Chapter 11: Phrases

Back to §11.11. Repeat running through

Onward to §11.13. Stop

§11.13. Stop

Now that it's possible to define phrases where different things are done in different circumstances, we sometimes want to halt early. This is what "stop" is for.

stop

This phrase causes the current rule to end immediately. It is most often used in the definition of other phrases:

```
To judge the score:
    if the score is 0, stop;
    say "The score is [score in words] more than it was a half-hour ago."
```

In the case when the score is 0, the "stop" ends the phrase immediately, so that the subsequent text is printed only if the score is not 0.

"Stop" can also be used in action rules, though this is not very good style - it's clearer to use "stop the action", which is exactly equivalent.

Start of Chapter 11: Phrases
Back to §11.12. Next and break
Onward to §11.14. Phrase options

§11.14. Phrase options

There are sometimes several slightly different ways to perform a given task but which have substantially the same definition. In the following example:

```
To go hiking, into the woods or up the mountain:
    if into the woods, say "Watch out for badgers.";
    if up the mountain, say "Better take your compass.";
    say "You go hiking."
```

...a phrase has been set up which can be used in three ways:

```
go hiking;
go hiking, into the woods;
go hiking, up the mountain;
```

Note that commas must be used to divide these "phrase options" from the rest of the text of the phrase. Within the definition of the phrase, the option's name is a valid condition, and

```
if up the mountain, ...
```

tests whether it is set; we can also test if it is not set using:

```
if not up the mountain, ...
```

A more substantial example from the Standard Rules is given by a phrase used mostly for internal, technical reasons:

list the contents of (object)

This phrase produces a list of all things whose holder is the given object, according to Inform's traditional conventions for room descriptions and inventory listings. Example:

```
list the contents of Marley Wood, as a sentence, with newlines
and including all contents;
```

Where this is possible, it's generally better to use "[list of things in ...]" instead, which produces the same result in an acceptable way for the middle of a sentence.

Note that this phrase is allowed to have multiple options specified, whereas "go hiking" above was not: this is because it was defined thus:

```
To list the contents of (something - an object), with newlines, indented, as a sentence,
including contents, including all contents, giving inventory information, giving brief
inventory information, using the definite article, listing marked items only, prefacing with
is/are, not listing concealed items, suppressing all articles and/or with extra indentation:
...
```

The significant difference is the word "and/or" instead of "or", which signals that more than one option can apply at a time.

Start of Chapter 11: Phrases

Back to §11.13. Stop

Onward to §11.15. Let and temporary variables

Example 177: ★★ **Equipment List** Overview of all the phrase options associated with listing, and examples of how to change the inventory list into some other standard formats.

§11.15. Let and temporary variables

A variable, as we have seen, is a name for a value which changes, though always remaining of the same kind. For instance, if "target" is a number variable (or "number that varies") then it may change value from 2 to 4, but not from 2 to "fishknife".

To make complicated decisions, phrases often need to remember values on a temporary basis. We have already seen this for the counter in a "repeat" loop, which exists only inside that loop, and then is no longer needed.

We can also make temporary variables using "let":

let (a name not so far used) **be** (value)

or:

let (a temporary named value) **be** (value)

This phrase creates a new temporary variable, starting it with the value supplied. The variable lasts only for the present block of phrases, which certainly means that it lasts only for the current rule. Examples:

```
let outer bull be 25;
let the current appearance be "reddish brown";
let the special room be Marley Wood;
```

The kinds of these are deduced from the values given, so that, for instance,

```
say "The outer bull scores [the outer bull in words] when you practice archery
in [special room]."
```

produces

```
The outer bull scores twenty-five when you practice archery in Marley Wood.
```

The variable name should be a new one; if it's the name of an existing one, then the kinds must agree. So:

```
let outer bull be 25;
let outer bull be 50;
```

is a legal combination, because the second "let" simply changes the value of the existing "outer bull" variable to a different number.

let (a name not so far used) **be** (name of kind)

This phrase creates a new temporary variable of the given kind. The variable lasts only for the present block of phrases, which certainly means that it lasts only for the current rule. Example:

```
let inner bull be a number;
```

The variable created holding the default value for that kind - in this case, the number 0. A handful of very obscure kinds have no default values, and then a problem message is produced. Inform also disallows:

```
let the conveyance be a vehicle;
```

because temporary variables aren't allowed to have kinds more specific than "object". (This is a good thing: suppose there are no vehicles in the world?) It's quite safe in such cases to use

```
let the conveyance be an object;
```

instead, which creates it as the special object value "nothing".

Temporary variables made by "let" are only temporarily in existence while a phrase is being carried out. Their values often change: we could say

```
let x be 10;
now x is 11;
```

for instance, or indeed we could "let x be 10" and then "let x be 11". But although we are allowed to change the value, we are not allowed to change the kind of value. The name "x" must always have the same kind of value throughout the phrase to which it belongs, so the following will not be allowed:

```
let x be 45;
now x is "Norway";
```

(The difference between using "let" and "now" here is that "let" can create a new temporary variable, whereas "now" can only alter things already existing: on the other hand, "now" can change many other things as well, whereas "let" applies only to temporary variables.)

⬆ Start of Chapter 11: Phrases

⬅ Back to §11.14. Phrase options

➡ Onward to §11.16. New conditions, new adjectives

⬇ Example 178: ★ **M. Melmoth's Duel** Three basic ways to inject random or not-so-random variations into text.

§11.16. New conditions, new adjectives

We can create new conditions by defining a phrase with "to decide whether" (or equivalently "to decide if"):

```
To decide whether danger lurks:
    if in darkness, decide yes;
    if the Control Room has been visited, decide no;
    decide yes.
```

If the player is indeed in darkness, the decision is "yes" because the "decide yes" stops the process right there. We can now write, for instance,

```
if danger lurks, ...
```

In fact, "danger lurks" is now a condition as good as any other, and can be used wherever a condition would be given. Rules can apply only "when danger lurks", for instance.

yes

or:
decide yes

This phrase can only be used in the definition of a phrase to decide whether a condition holds. It ends the decision process immediately and makes the condition true.

no

or:
decide no

This phrase can only be used in the definition of a phrase to decide whether a condition holds. It ends the decision process immediately and makes the condition false.

We can also supply definitions of adjectives like this. So far, new adjectives have been defined like so:

```
Definition: a supporter is occupied if it is described and something is on it.
```

If we want to give a definition which involves more complex logic, we can use a special form allowing us to make arbitrary decisions. In this longer format, the same definition would look like so:

```
Definition: a supporter is occupied:
    if it is undescribed, decide no;
    if something is on it, decide yes;
    decide no.
```

Here "it" refers to the supporter in question. Note that there are now two colons in this sentence, one after "Definition", the other after the clause being defined. But that apart, it's a phrase like any other: it must end in "yes" or "no" just as the "danger lurks" example must. "Decide no" and "decide yes" are needed so often that they can be abbreviated by leaving out "decide":

```
Definition: a supporter is occupied:
    if it is undescribed, no;
    if something is on it, yes;
    no.
```

Start of Chapter 11: Phrases

Back to §11.15. Let and temporary variables

Onward to §11.17. Phrases to decide other things

Example 179: ★★★ **Owen's Law** OUT always means "move to an outdoors room, or else to a room with more exits than this one has"; IN always means the opposite.

§11.17. Phrases to decide other things

A condition is a yes/no decision, but we can also take decisions where the result is a value. Suppose we want to create a concept of the "grand prize", which will have different values at different times in play. Each time the "grand prize" is referred to, Inform will have to decide what its value is, and the following tells Inform how to make that decision:

```
To decide which treasure is the grand prize:
    if the Dark Room has been visited, decide on the silver bars;
    decide on the plover's egg.
```

Note that we have to say what kind the answer will be: here it's a kind of thing called "treasure" (which we're supposing has already been created), and as it turns out only two treasures are ever eligible anyway (we're also supposing that the plover's egg and the silver bars are treasures already created, of course). And note also that the phrase must in all cases end with a "decide on ..." to say what the answer is:

decide on (value)

This phrase can only be used in the body of a definition of a phrase to decide a value. It causes the calculation to end immediately, with the outcome being the given value, which must be of the kind expected. Example:

```
To decide which number is double (N - a number):
    let D be N times N;
    decide on D.
```

Now that we have "grand prize" created, we can use it just as we would use any other value, so for instance:

```
if taking the grand prize, ...
```

As this is something of a dialect difference between English speakers, "what" and "which" are synonymous here, i.e., we could equally well write something like:

```
To decide what number is the target score: ...
```

(A phrase to decide if something-or-other is exactly the same thing as a phrase to decide a truth state, and indeed, if we want to then we can use "decide on T", where T is a truth state, in its definition. For instance:

```
To decide if time is short:
    if the time of day is after 10 PM, decide on true;
    ...
    decide on whether or not Jennifer is hurried.
```

"Decide on true" is exactly equivalent to the more normally used "decide yes", and of course it is optional. The last line is more interesting since it effectively delegates the answer to another condition.)

Start of Chapter 11: Phrases

Back to §11.16. New conditions, new adjectives

Onward to §11.18. The value after and the value before

Example 180: ★ **Witnessed 2** A piece of ghost-hunting equipment that responds depending on whether or not the meter is on and a ghost is visible or touchable from the current location.

Example 181: ★★★ **A Haughty Spirit** Windows overlooking lower spaces which will prevent the player from climbing through if the lower space is too far below.

§11.18. The value after and the value before

A point which has come up several times in recent chapters is that enumerated kinds of value have a natural ordering. For example, if we write:

```
Colour is a kind of value. The colours are red, orange, yellow, green, blue, indigo and violet.
```

...then we not only have seven possible values, we have put them into a sequence, in order of their naming. We can't perform arithmetic on colours, of course, but we can perform comparisons on them. Thus "red < yellow" is true, while "green >= violet" is not. (More on comparisons in the chapter on Numbers and Equations, which also covers arithmetic.)

It's also sometimes useful to get at the sequence directly. First, the two ends:

first value of (name of kind) **... value**

This phrase produces the first-created value of the given kind, which should be an enumeration. Example: if we have

```
Colour is a kind of value. The colours are red, orange, yellow, green, blue, indigo and violet.
```

then "first value of colour" is red.

last value of (name of kind) **... value**

This phrase produces the last-created value of the given kind, which should be an enumeration. Example: if we have

> Colour is a kind of value. The colours are red, orange, yellow, green, blue, indigo and violet.

then "last value of colour" is violet.

And now how to step forward and back:

(name of kind) **after** (enumerated value) **... value**

This phrase produces the next-created value of the given kind, which should be an enumeration. Example: if we have

> Colour is a kind of value. The colours are red, orange, yellow, green, blue, indigo and violet.

then "colour after orange" is yellow.

(name of kind) **before** (enumerated value) **... value**

This phrase produces the previous-created value of the given kind, which should be an enumeration. Example: if we have

> Colour is a kind of value. The colours are red, orange, yellow, green, blue, indigo and violet.

then "colour before blue" is green.

Start of Chapter 11: Phrases

Back to §11.17. Phrases to decide other things

Onward to Chapter 12: Advanced Actions: §12.1. A recap of actions

Example 182: ★ **Entropy** All objects in the game have a heat, but if not kept insulated they will tend toward room temperature (and at a somewhat exaggerated rate).

Example 183: ★★★ **The Hang of Thursdays** Turns take a quarter day each, and the game rotates through the days of the week.

Examples from Chapter 11: Phrases

Start of this chapter

169

Example Ahem

RB

Writing a phrase, with several variant forms, whose function is to follow a rule several times.

As we see in the example here, it is possible to use slashed variations in more than one place in a phrase, and to offer a number of separate forms. The main rule of thumb to remember is that value inputs for the phrase should always be separated by some text; so

```
To do/follow (chosen rule - a rule) exactly/precisely/just/-- (N - a number)
time/times:
    ....
```

would cause a problem when we tried to call it with

```
follow the throat-clearing rule 2 times.
```

In general, we probably don't need to make our phrase definitions quite so flexible as this, but it's a good idea to account for "a" vs. "the", and for the possibility of using singular and plural forms, especially when writing extensions or other source to be shared.

```
"Ahem"

To do/follow (chosen rule - a rule) exactly/precisely/just (N - a number)
time/times:
    repeat with index running from 1 to N:
        follow chosen rule.

This is the throat-clearing rule:
    say "'Ahem,' says [a random visible person who is not the player]."

After waiting:
    do the throat-clearing rule just one time.

Instead of listening:
    follow the throat-clearing rule precisely three times.

Instead of smelling:
    follow the throat-clearing rule exactly 2 times.

Chateau Marmont is a room. Tom, Jack, Zsa-Zsa, and Wilma-Faye are people in
the Chateau. Zsa-Zsa and Wilma-Faye are women.

Test me with "wait / smell / listen".
```

170 ★★★ Example **Ferragamo Again** RB

Using the same phrase to produce different results with different characters.

Here we use phrases that match individual items where possible, and the general kind otherwise:

```
"Ferragamo Again"

The Break Room is a room. Vanessa, Tina, and Lisa are women in the Break Room. Mark and Holman are men in the Break Room.

Understand the commands "ask" and "tell" and "answer" as something new.

Understand "talk about [any subject]" as talking about. Talking about is an action applying to one visible thing.

Understand "talk about [text]" as talking randomly about. Talking randomly about is an action applying to one topic. Carry out talking randomly about: say "Mostly you're interested in [the list of subjects]."

Carry out talking about something:
    now the previous subject is the noun.

Report talking about something:
    say "You chat for a while about [the noun]."

A subject is a kind of thing. Assyrian vowel sounds, designer handbags, and instant run-off voting are subjects. Understand "linguistics" and "mute" and "stop" as sounds. Understand "prada" and "tods" and "coach" and "carmen marc valvo" as designer handbags. Understand "reform" and "election" and "election fraud" and "two-party system" and "Diebold" as instant run-off voting.

To say (annoyed-person - a person) gestures in irritation:
    say "[The annoyed-person] sighs heavily. [run paragraph on]"

To say (annoyed-person - Vanessa) gestures in irritation:
    say "[The annoyed-person] takes off her glasses and polishes them on her sleeve. [run paragraph on]".

To say (annoyed-person - Holman) gestures in irritation:
    say "Holman bobs his head. [run paragraph on]"

The previous subject is a subject that varies.

Instead of talking about something for more than one turn:
    if the noun is the previous subject, say "[a random visible person who is not the player gestures in irritation]Maybe you should let this one go.[line break][paragraph break]";
    otherwise continue the action.
```

```
Test me with "talk about chocolate / talk about vowel sounds / g / talk about
handbags / talk about prada / talk about tods".
```

171

★★★ Example **Proposal**

Asking the player a yes/no question which he must answer, and another which he may answer or not as he chooses.

RB

Suppose we want to ask the player a question where he might say yes or no in response. There are two possible forms of this: the modal question where the player must pick one to proceed, and the non-modal question where he might also type other verbs.

```
"Proposal"

The story genre is "A Worked Example about Yes/No Questions".

Section 1 - Asking a Modal Yes/No Question

When play begins:
    say "Do you like Mr Spruce? ";
    if player consents, now Spruce is handsome;
    otherwise now Spruce is ugly;
    say paragraph break.

Section 2 - Mr Spruce's Non-Modal Question

Use full-length room descriptions.

The Conservatory is a room. "You are in a room full of plants."

Mr Spruce is a man in the Conservatory. Mr Spruce can be apprehensive or
calm. Mr Spruce is calm. Mr Spruce can be handsome or ugly.

At 9:02 AM: say "Mr Spruce flings himself to his knees and implores you to
become his lawfully wedded wife.";
    now Mr Spruce is apprehensive;
    Mr Spruce gives up in two minutes from now.

At the time when Mr Spruce gives up:
    say "Mr Spruce sighs heavily, seeing that you don't intend to reply. 'Never
mind, my dear, I'll ask later. Perhaps I should have spoken to your Papa first...
yes, a gently-bred female... no wonder...'";
    now Mr Spruce is calm;
    Mr Spruce departs in one minute from now.

At the time when Mr Spruce departs:
    if the player can see Mr Spruce, say "Mr Spruce takes his leave of you.";
    otherwise say "Mr Spruce pokes his head in to say that he is leaving.";
    end the story saying "Well, that is over..."
```

```
Instead of saying yes in the presence of an ugly apprehensive Mr Spruce:
	now Mr Spruce is calm;
	say "Remembering what your mother said to you about the stock exchange and Dear Papa, you close your eyes and accept Mr Spruce.";
	end the story saying "Alas for your maiden hopes."

Instead of saying yes in the presence of a handsome apprehensive Mr Spruce:
	now Mr Spruce is calm;
	say "You are silent with delight for a moment before you say yes, yes!";
	end the story saying "How Genevieve Stubbs will cry!"

Instead of saying no in the presence of an ugly apprehensive Mr Spruce:
	now Mr Spruce is calm;
	say "Gently you inform Mr Spruce that it is impossible. He seems less deflated than you had expected.";
	end the story saying "Odd, that..."

Instead of saying no in the presence of a handsome apprehensive Mr Spruce:
	now Mr Spruce is calm;
	say "You lower your eyes and refuse petulantly, hoping to stir him to a more ardent repetition of these same requests. But then -- alack! -- he says 'I see how it is!' in a strangled voice, and strides from the room!";
	end the story saying "A fatal error!"
```

And since the player might SAY YES TO SPRUCE, we had better reroute the relevant options:

```
Instead of answering Mr Spruce that "no", try saying no.

Instead of answering Mr Spruce that "yes", try saying yes.

Instead of asking Mr Spruce to try saying yes, try saying yes.

Instead of asking Mr Spruce to try saying no, try saying no.

Instead of saying sorry, try saying no.

Instead of asking Mr Spruce to try saying sorry, try saying no.

Instead of answering Mr Spruce that "sorry", try saying no.

Test me with "z / z / z / yes".

Test more with "z / z / z / no".
```

172

★ Example **Matreshka**

RB

A SEARCH [room] action that will open every container the player can see, stopping only when there don't remain any that are closed, unlocked, and openable.

```
"Matreshka"

Ransacking is an action applying to one thing.

Check ransacking:
    if the noun is not the location, say "You can hardly search [the noun] from
here." instead.

Carry out ransacking:
    while the player can see a closed openable unlocked container (called target):
        say "[target]: [run paragraph on]";
        try opening the target.

Report ransacking:
    say "You can see nothing further worth searching."

The Russian Gift Shop is a room. In the Russian Gift Shop is a large wooden
doll. It is closed and openable. In the large wooden doll is a medium wooden
doll. It is closed and openable. In the medium wooden doll is a small wooden
doll. It is closed and openable. In the small wooden doll is a tiny solid wooden
doll.
```

And now we need to borrow from a later chapter for the command that will make this work:

```
Understand "search [any visited room]" as ransacking.

Test me with "search gift shop".
```

173 ★ Example Princess and the Pea RB

The player is unable to sleep on a mattress (or stack of mattresses) because the bottom one has something uncomfortable under it.

The main point here is that we need to figure out where the stack meets the floor:

```
"Princess and the Pea"

The Topmost Turret is a room. A mattress is a kind of supporter. A mattress is
always enterable. A mattress is portable.

A large mattress is a mattress in the Turret. A medium mattress is a mattress in
the Turret. A small mattress is a mattress in the Turret.

Instead of sleeping when the player is on a mattress (called the bed):
    let the item be the bed;
    while the holder of the item is not a room:
        let the item be the holder of the item;
    say "You can still feel something very uncomfortable under [the item]."
```

```
Instead of sleeping:
    say "You can't sleep standing up!"

Instead of looking under a mattress, say "You scout around, but are unable to
determine what's causing you this discomfort. If only your maid Winnie were
here. She's very good at this."

Test me with "sleep / enter small / sleep / get up / get small / put small on
medium / get on small / sleep / get up / g / get medium / put medium on large /
get on small / look / sleep".
```

174 Example **Numberless** RB

A simple exercise in printing the names of random numbers, comparing the use of "otherwise if...", a switch statement, or a table-based alternative.

```
"Numberless"

The Rambling Warren is a room.

When play begins:
    let N be a random number between 1 and 5;
    if N is 1:
        say "N is one.";
    otherwise if N is 2:
        say "N is two.";
    otherwise if N is 3:
        say "N is three.";
    otherwise:
        say "N is more than the number of your toes."
```

The final "otherwise" here will fire only if none of the earlier conditions applies; we could leave it out and print nothing in the case that N is 4 or 5.

The more compact way to do this is to create a list of values that our number could match; in many programming languages this is called a switch statement. For example:

```
When play begins:
    let Y be a random number between 6 and 10;
    if Y is:
        -- 6: say "Six is the magic number!";
        -- 7: say "The number of the day is seven!";
        -- otherwise: say "Today's magic number is boring."
```

As a final option, we can use a construction we've seen only briefly before now: a table. The use of tables will be explained more fully in their own chapter, but here we see in brief that we can assign a number of values to one column of a table and then use that table to look up output:

```
When play begins:
    let X be a random number between 11 and 14;
    if X is a number listed in the Table of Switching, say "[output entry][paragraph break]";
    otherwise say "X is greater than the number of your noses!"

Table of Switching

    number  output
    11      "X is eleven!"
    12      "X is twelve!"
    13      "X is thirteen!"

Test me with "z".
```

As we shall see, things other than text can be stored in tables, so we could also use a table as a way to look up objects or even rules to carry out.

175

★ Example Wonka's Revenge

RB

A lottery drum which redistributes the tickets inside whenever the player spins it.

```
"Wonka's Revenge"

The Caribou Lodge is a room. "Hundreds of expectant faces are turned your way from every table." A lottery drum is in the Lodge. "Before you is the lottery drum[if we have spun the drum], ready to disgorge a ticket[otherwise], waiting to be spun[end if]." In the drum are a red ticket, an orange ticket, a yellow ticket, a green ticket, a blue ticket, a purple ticket, and a ticket of pure gold. The drum is closed and openable.

Understand "spin [something]" as spinning.

Spinning is an action applying to one thing.

Check spinning: if the noun is an open container which contains something, say "[The list of things in the noun] would fly out." instead.

Carry out spinning a container:
    shuffle the contents of the noun.

Report spinning:
    if the noun contains something, say "You rattle [if the noun is transparent][the list of things in the noun][otherwise]the stuff[end if] in [the noun].";
    otherwise say "Nothing results of your shaking [the noun]."
```

Inform keeps track of the order in which things have been put into a container. If we want to change that order without the player's intervention, we can move the things ourselves.

```
To shuffle the contents of (basket - a container):
    let moves be the number of things in the basket;
    repeat with counter running from 1 to moves:
        move a random thing in the basket to the basket.

After opening the drum when we have spun the drum for the first time:
    if something (called the pick) is in the drum:
        try searching the drum;
        say "[The pick] it is, then.";
        silently try taking the pick;
        if the pick is the ticket of pure gold, end the story finally;
        otherwise end the story saying "Oh well, better luck next time."

Test me with "open drum / look in drum / close drum / spin drum / open drum".
```

---

176 ★★ **Example Strictly Ballroom** RB

People who select partners for dance lessons each turn.

Many simple repetitions can effectively be done with a "now..." instruction: it is quicker to say

```
now every person is angry
```

than

```
repeat with offended party running through people:
    now the offended party is angry.
```

Repeat comes in handy when we have something a bit more complicated to do with each item:

```
"Strictly Ballroom"

A person can be alert or occupied. A person is usually alert.

When play begins:
    now the player is occupied.

Dance is a kind of value. The dances are waltzes, polkas, cha-chas, charlestons, fox-trots, tangos, lambadas, and two-steps.

The current round is a dance that varies.

Manner is a kind of value. The manners are swiftly, primly, energetically, suavely, seductively, elegantly, and badly.

Every turn: now the current round is a random dance.

Every turn:
    repeat with dancer running through people who are not the player:
        if dancer is alert:
            now dancer is occupied;
            let partner be a random alert person who is not the dancer;
            if partner is a person:
                now partner is occupied;
                say "[The dancer] [the current round][if a random chance of 1 in 5 succeeds] [a random manner][end if] with [partner]. ";
            otherwise:
                say "[paragraph break][The dancer] is forced to be a wallflower. Poor [dancer]. ";
    say paragraph break.
```

Notice we did not say "repeat with dancer running through alert people who are not the player...". This is because Inform would draw up a list of alert people at the beginning of the repeat, and not take into account which people became occupied partway through the repetition. If we want to make sure that each person dances only with one other person, we have to continue checking alertness each time we run through the repetition.

After all the partners are assigned, we can set up for the next turn by making everyone alert again, and for this we do not need "repeat":

```
Every turn: now every person is alert; now the player is occupied.

Before doing something to someone: now the noun is occupied.

Before doing something when the second noun is a person: now the second noun is occupied.

Instead of doing something to someone: say "You successfully distract [the noun]."

The Pacific Ballroom is a room. "A rather utilitarian space at the moment, since this is a class and not a party." Timmy, Tommy, Joey, George, Mary, Martha, Yvette, McQueen, Linus, and Patricia are people in the Pacific Ballroom.

Test me with "z / ask linus about blanket / z / z".
```

177 Example **Equipment List**

RB

Overview of all the phrase options associated with listing, and examples of how to change the inventory list into some other standard formats.

Most of the phrase options above are relatively self-explanatory; a few are less so. Here is an overview:

"With newlines" tells Inform to put a new line before each listed object. Indented tells it to indent contents of objects, when listing these.

"Giving inventory information" means to append information such as (closed) or (being worn) to objects.

"As a sentence" means to put "and" before the last object and commas between them; this is usually not used in conjunction with newline listing. "As a sentence" obeys whatever conventions about the use of the serial comma we may have established with the "Use serial comma" option.

"Including contents" means to list the contents of open or transparent containers and all supporters, whereas including all contents means to list the contents of all containers, even opaque closed ones.

"Tersely", perhaps unexpectedly, puts parentheses around objects listed as the contents of other objects.

"Giving brief inventory information" omits most of the inventory tags, such as "(open)" and "(worn)", but does list "(closed)" for closed containers which might not otherwise be obviously openable.

"Using the definite article" means prefixing objects with "the", if applicable, rather than "a".

"Listing marked items only" means including only objects that have already been declared "marked for listing".

"Prefacing with is-are" means that Inform will write "is" before the list if it contains only one item, and "are" if the list contains more than one.

"Not listing concealed items" means to omit from the list anything which is scenery.

Finally, "with extra indentation" means that the whole list should be indented slightly, in emulation of the default inventory listing.

With this information, we can try rewriting the inventory behavior to emulate the standard or to explore alternate versions:

```
"Equipment List"

The Watery Room is a room. The player carries a snorkel and a waterproof
sack. The waterproof sack contains an undersea map, a diving guide, a cup, and
500 Argentine pesos. The cup contains a worm. The player wears a swimsuit
and a pair of flippers. The sack is openable and open.

Inventory listing style is a kind of value. The inventory listing styles are tall, wide,
curt, minimal, divided tall, and divided wide. Current inventory listing style is an
inventory listing style that varies.

Understand "inventory [inventory listing style]" as requesting styled inventory.
Requesting styled inventory is an action applying to an inventory listing style. It is
an action out of world.

Carry out requesting styled inventory:
    now current inventory listing style is the inventory listing style understood.

Report requesting styled inventory:
    say "Inventory listing is now set to [current inventory listing style]."
```

We begin by emulating the standard inventory listing style:

```
Instead of taking inventory when current inventory listing style is tall:
   if the number of things enclosed by the player is 0, say "You are empty-handed." instead;
   say "You are carrying: [line break]";
   list the contents of the player, with newlines, indented, giving inventory information, including contents, with extra indentation.
```

Here we offer the alternative of listing everything together as a paragraph:

```
Instead of taking inventory when current inventory listing style is wide:
   if the number of things enclosed by the player is 0, say "You are empty-handed." instead;
   say "You are carrying ";
   list the contents of the player, giving inventory information, as a sentence, including contents;
   say "."
```

This may be unsatisfactory, however. Items that are inside other items are not set off from those merely carried by the player. One way around this is to use terse listing, giving such descriptions as "a waterproof sack (in which are an undersea map, a diving guide, a cup (in which is a worm) and a 500 Argentine pesos)" as opposed to the more confusing " a waterproof sack (open), inside which are an undersea map, a diving guide, a cup, inside which is a worm and a 500 Argentine pesos".

```
Instead of taking inventory when current inventory listing style is curt:
   if the number of things enclosed by the player is 0, say "You are empty-handed." instead;
   say "You are carrying ";
   list the contents of the player, tersely, giving brief inventory information, as a sentence, including contents;
   say "."
```

If, using the above style, we close the sack, we will still get "(closed)" after the sack's listing. The following minimalist listing style abolishes even that nicety:

```
Instead of taking inventory when current inventory listing style is minimal:
   if the number of things enclosed by the player is 0, say "You are empty-handed." instead;
   say "You are carrying ";
   list the contents of the player, tersely, as a sentence, including contents;
   say "."
```

If we want to list worn things separately from carried things, we have occasion to put "listing marked items only" to work:

```
Instead of taking inventory when the current inventory listing style is divided wide:
   if the number of things enclosed by the player is 0, say "You are empty-handed." instead;
   say "You are wearing ";
   now all things enclosed by the player are unmarked for listing;
   now all things worn by the player are marked for listing;
```

```
    if no things worn by the player are marked for listing, say "nothing";
    otherwise list the contents of the player, as a sentence, listing marked items only;
    say ".[paragraph break]";
    say "You are carrying ";
    now all things carried by the player are marked for listing;
    now all things worn by the player are unmarked for listing;
    if no things carried by the player are marked for listing, say "nothing";
    otherwise list the contents of the player, as a sentence, tersely, giving brief inventory information, listing marked items only;
    say ".[paragraph break]".
```

And similarly for a tall divided inventory:

```
Instead of taking inventory when the current inventory listing style is divided tall:
    if the number of things enclosed by the player is 0, say "You are empty-handed." instead;
    if the player carries something:
        now all things enclosed by the player are unmarked for listing;
        now all things carried by the player are marked for listing;
        say "You are carrying: [line break]";
        list the contents of the player, with newlines, indented, giving inventory information, including contents, with extra indentation, listing marked items only;
    if the player wears something:
        now all things enclosed by the player are unmarked for listing;
        now all things worn by the player are marked for listing;
        say "You are wearing: [line break]";
        list the contents of the player, with newlines, indented, including contents, with extra indentation, listing marked items only.

Test me with "i / inventory wide / i / inventory curt / i / close sack / i / open sack / inventory minimal / i / close sack / i / open sack / inventory divided wide / i / inventory divided tall / i / drop all / i / take all / take off swimsuit / take off flippers / i / i divided wide / i / wear swimsuit / drop all / i".
```

178

★ Example M. Melmoth's Duel

RB

Three basic ways to inject random or not-so-random variations into text.

```
"M. Melmoth's Duel"

Saint-Germain-des-Prés is a room. "Haunt of artists, of the coffee-drinking sort, and of cafés, of the artist-haunted sort, you once again find yourself outside M. Melmoth's hotel. Today [one of]the recently-fallen rain runs down the gutters of the 6th[or]sunlight glints even off the blackened windows of the Abbey[or]crowds of vulgar children play chase around the lampposts[at random], and you long to be indoors."

The Hôtel d'Alsace is inside from Saint-Germain-des-Prés. "Typical. Oscar writes you a letter announcing his own imminent demise - 'My wallpaper and I are fighting a duel to the death. One or other of us has got to go.' - and then you get
```

```
there and he's out, no doubt procuring paint the colour of absinthe, if he isn't
procuring the painter."

Tint is a kind of value. The tints are green, aquamarine and darkish purple.

The wallpaper is fixed in place in the Hôtel. The wallpaper has a tint. "In this
light, the wallpaper has a distinctly [tint of the wallpaper] wash. [if the tint of the
wallpaper is darkish purple]You particularly dislike purple.[end if]"

Before going to the Hôtel: now the wallpaper is a random tint.

After going from the Hôtel, say "You leave, shaking your head. But within twenty-
four hours, you are back, as you always knew you would be."

Test me with "in / out / look / in / out / look".
```

---

179 ★★★★ **Example** **Owen's Law** *RB*

OUT always means "move to an outdoors room, or else to a room with more exits than this one has"; IN always means the opposite.

Suppose we want the game to interpret "GO OUT" as "move towards an outdoors room, or towards a room with more exits than the current room", while "GO IN" means "move toward a room with fewer exits, or towards an indoors room". Thus going in repeatedly within a building would lead towards dead-ends, while going out repeatedly would lead towards the center of the building and then towards an exit to the outside world.

We start by encoding these rules as definitions:

```
"Owen's Law"

A room can be indoors or outdoors. A room is usually indoors.

Definition: a room is outward:
    if it is not adjacent, no;
    if it is indoors and the location is outdoors, no;
    if it is outdoors and the location is indoors, yes;
    if the number of rooms adjacent to it is greater than the number of rooms
adjacent to the location, yes;
    otherwise no.

Definition: a room is inward:
    if it is not adjacent, no;
    if it is outdoors and the location is indoors, no;
    if it is indoors and the location is outdoors, yes;
    if the number of rooms adjacent to it is less than the number of rooms
adjacent to the location, yes;
    otherwise no.

Instead of going nowhere when the noun is outside: try exiting.
Instead of exiting when the player is in a room:
    if at least one room is outward:
        let the destination be a random outward room;
        let the way be the best route from the location to the destination;
        say "(that is, [way])[command clarification break]";
        try going the way instead;
    otherwise:
        say "It's not entirely obvious which way you mean. ";
        carry out the listing available exits activity.

Instead of going inside when the room inside from the location is not a room and
at least one room is inward:
    if more than one room is inward:
        carry out the listing available exits activity;
    otherwise:
        let the destination be a random inward room;
        let the way be the best route from the location to the destination;
        say "(that is, [way])[command clarification break]";
        try going the way instead.

Instead of going nowhere:
    carry out the listing available exits activity.
```

This "listing available exits" is a refinement borrowed from a future chapter, which allows us to specify special listing and printing rules:

```
Listing available exits is an activity.

Rule for listing available exits:
    if going inside and an adjacent room is inward:
        say "From here 'in' could reasonably mean [a list of adjacent inward
rooms].";
        rule succeeds;
    if exiting and an adjacent room is outward:
        say "From here 'out' could reasonably mean [a list of outward adjacent
rooms].";
        rule succeeds;
    say "From here you can go [a list of adjacent rooms]."

Before printing the name of a room (called the target) while listing available exits:
    let aim be the best route from the location to the target;
    say "[aim] to the ".

Rule for printing the name of an unvisited room which is not the location:
    say "unknown location".

Dune is an outdoors room. "Hundreds of feet of dune stretch west to the beach,
crisscrossed with dune-buggy tracks and the footprints of birds. To the east is a
low-lying, boxy concrete installation."

Ocean Shores Military Installation is east of the Dune. It is an outdoors room.
"The World War II emplacements, built in case of Japanese invasion, have never
been destroyed, though with all the weapons and furnishings gone it is difficult to
make much sense of the original structure. A doorway leads west into concrete-
```

```
lined darkness; a rusty but reliable ladder ascends to a walkway overlooking the sea."

Walkway is above Ocean Shores Military Installation. "From here you have a long view of the dunes and the Pacific Ocean, complete with the rotting hull of a long-stranded vessel."

Dark Echoing Room is inside from Ocean Shores Military Installation. Dank Dripping Room is east of Dark Echoing Room. Narrow Room is south of Dark Echoing Room. Small Sealed Chamber is north of Dark Echoing Room. Room Smelling of Animal Urine is north of Dank Dripping Room. The description of a room is usually "It is dark in here, and feels unsafe."

Test me with "e / u / d / in / s / out / n / out / e / in / out / out / out".
```

180 ★ **Example Witnessed 2** RB

A piece of ghost-hunting equipment that responds depending on whether or not the meter is on and a ghost is visible or touchable from the current location.

```
"Witnessed"

The player carries a device called a Trifield Natural EMF Meter. The description of the Meter is "This cost a pretty penny off the internet, but it's worth it: according to the website it has been programmed by PhD physicists to ignore manmade sources of fields and to respond only to paranormal EMF changes.

It also features an optional Tone Alarm, which can be turned on to indicate when readings spike. If the alarm is off, the meter just reads out the magnetic and electric field levels on a scale from 0-100 microteslas, or 0-1000 V/m.

Since both fields are important, you keep the meter set to SUM mode. The meter has its own optional backlighting, so that you can see the reading even if your flashlight is off. Currently it is reading at [meter setting]." A Tone Alarm is part of the Meter. It is a device. The description of the Tone Alarm is "The Tone Alarm will make a noise, if the EMF picks up a spike."

To decide what number is meter setting:
    if the meter is switched off, decide on 0;
    if a ghost is touchable, decide on 35;
    if a ghost is visible, decide on 12;
    decide on 0.

After switching on the meter:
    say "You turn on the meter. The needle steadies at [meter setting]."

Every turn: if the meter setting is greater than 10 and the Tone Alarm is switched on, say "[The Tone Alarm] shrieks."

Thirtieth Street Station is a room. "A huge, high, rectangular room with coffered ceilings, which looks grand but mostly makes you feel lonely and small. There
```

```
are long benches in rows down the middle of the room, and an information desk with the train times, and a series of ticket windows, none of which matters very much at the moment."

The benches are an enterable supporter. They are scenery in the Station. The information desk is scenery in the Station. Some ticket windows are scenery in the Station. Instead of examining scenery in the Station: say "You're fairly sure that whatever is going on here has nothing to do with [the noun]." Understand "window" as ticket windows.

The mural is fixed in place in Thirtieth Street. "At the north side of the station is a particularly pointless and empty annex to the main room. It is dominated by a huge relief of sorts, and this is what you remember." Understand "metal" or "relief" or "huge" as the mural. The description of the mural is "It is both stylized and confusing, but you think it might be supposed to represent the various tasks and occupations of Philadelphia's population. The portions closer to the ground look as though they have recently been subjected to a light dusting of talcum powder. No unusual prints are evident."

The wind chimes are fixed in place in Thirtieth Street. "Carefully attached to the wall with a piece of duct tape and a hook is a light-weight set of wind chimes. Someone else has been here before you, it seems." The description is "Several of your friends use wind chimes as a sort of ghost alarm, since ghosts sometimes cause very localized movements of air when there is no natural breeze."

A ghost is a kind of person. The pale figure is a ghost.

At 9:03 AM: move the pale figure to the location; say "You shiver with some sort of presence."

Test me with "turn on alarm / turn on meter / z / z / z / x figure".
```

181 ★★★★ Example **A Haughty Spirit** *RB*

Windows overlooking lower spaces which will prevent the player from climbing through if the lower space is too far below.

Suppose we have a game in which the player can climb through windows which overlook rooms below. We want him to be allowed to climb out windows to reach a room on the same level or at most one level lower than the one he's on; otherwise, he should get a refusal, saying that he would break his neck.

To figure out the height distance between the start room and the destination room, we might have a repeat loop look at all the directions one has to follow along the "best route" path between the two rooms, and record any ups and downs; then subtract the number of "up" steps from the number of "down" steps, and report what remains.

```
"A Haughty Spirit"
```

```
To decide what number is the distance (first place - a room) rises above (second place - a room):
    let the total distance be the number of moves from the first place to the second place;
    if the total distance is less than 1, decide on 0;
    let count of down moves be 0;
    let count of up moves be 0;
    let next place be the first place;
    repeat with counter running from 1 to the total distance:
        let the way be the best route from the next place to the second place;
        if the way is down, let count of down moves be the count of down moves plus 1;
        if the way is up, let the count of up moves be the count of up moves plus 1;
        let next place be the room the way from next place;
    let the decision be the count of down moves minus the count of up moves;
    decide on the decision.
```

Now we just have to create windows and some action rules for interacting with them...

```
A window is a kind of thing. A window is always fixed in place. A window can be open or closed. A window is usually closed. A window can be openable or unopenable. A window is usually openable.

Understand "climb through [something]" as entering. Understand "jump through/out [something]" as entering.

Before entering a closed window:
    say "[The noun] would have to be opened first." instead.

Instead of entering a window:
    if the noun overlooks a room (called the far side):
        let fall be the distance the location rises above the far side;
        if fall is greater than 1, say "You'd break your neck." instead;
        say "You tumble into [the far side].";
        move the player to the far side;
    otherwise:
        say "There's nowhere to go."

Instead of examining a window:
    say "[The noun] [if the noun is open]opens over[otherwise]gives a view of[end if] [the list of rooms overlooked by the noun]."
```

Here we must anticipate a little from the chapter on Relations, and provide ourselves with a way of keeping track of how windows and rooms relate to one another:

```
Overlooking relates various windows to various rooms. The verb to overlook means the overlooking relation. The initial appearance of a window is usually "[The item described] overlooks [the list of rooms overlooked by the item described]."

The Square Keep is above the Winding Staircase. The Winding Staircase is above the Motte. A crown and a broken sword are in the Motte. The Bailey is west of the Motte.
The long window is in the Keep. The long window overlooks the Bailey and the
Motte. The narrow window is in the Winding Staircase. The narrow window
overlooks the Bailey.

Test me with "jump through window / open window / jump through window / d / x
narrow window / open window / climb through window / e / up / down".
```

We could then add rules to allow the player to look through windows and see things in the rooms below, but that would require more material from later chapters.

182

Example **Entropy**

RB

All objects in the game have a heat, but if not kept insulated they will tend toward room temperature (and at a somewhat exaggerated rate).

```
"Entropy"

Heat is a kind of value. The heats are frosty, cold, cool, room temperature,
warm, hot, and scalding. Everything has a heat. The heat of a thing is usually
room temperature.

Every turn:
    repeat with item running through things which are not in an insulated
container:
        if the heat of the item is greater than room temperature, now the heat of the
item is the heat before the heat of the item;
        if the heat of the item is less than room temperature, now the heat of the
item is the heat after the heat of the item.

Definition: a container is insulated if it is closed and it is opaque.

The vacuum thermos is an opaque closed openable container carried by the
player. In the vacuum thermos is a frosty thing called an ice cube.

Every turn:
    if the heat of the ice cube is greater than cold:
        if the ice cube is visible, say "The ice cube melts! 'HA ha,' says Maxwell, in
a very unsporting, some might say demonic, way.";
        now the ice cube is nowhere.

Before printing the name of something: say "[heat] ".

Equilibrium is a room. "A perfectly smooth chamber sealed from the outside
world. You can't at this moment work out where the exit is, though possibly that
is just because the lighting is so very very even and diffuse. And doesn't come
from anywhere that you can see, either."

Maxwell is a man in Equilibrium. "Maxwell perches awkwardly on a stool across
from you[if Maxwell has something], holding [a list of things carried by Maxwell]
[end if]." He is carrying a box of Chinese food. The Chinese food is scalding. "A
discarded [item described] lies on the floor." The description of Maxwell is "He
has the faintly peevish look of one who has not been properly fed."
```

```
Every turn when Maxwell has the food:
    if the heat of the Chinese food is greater than warm, say "Maxwell takes a bite, and swears.";
    if the heat of the Chinese food is warm, say "Maxwell eats as fast as he can, enjoying the food while it's at just the right temperature.";
    if the heat of the Chinese food is less than warm:
        say "Maxwell sadly stabs at his leftovers with a chopstick, but does not try to eat any more.";
        move the food to the location.

Test me with "z / z / open thermos / close thermos / open thermos".
```

183 ★★★ **Example** **The Hang of Thursdays** *RB*

Turns take a quarter day each, and the game rotates through the days of the week.

```
"The Hang of Thursdays"

The Stage is a room. Rule for printing the name of the stage: say "[current weekday] [current time period]" instead.

A weekday is a kind of value. The weekdays are Saturday, Sunday, Monday, Tuesday, Wednesday, Thursday, Friday. The current weekday is a weekday that varies. The current weekday is Saturday.

A time period is a kind of value. The time periods are morning, afternoon, evening, night. The current time period is a time period that varies. The current time period is afternoon.

This is the new advance time rule:
    if the current time period is less than night:
        now the current time period is the time period after the current time period;
    otherwise:
        now the current time period is morning;
        now the current weekday is the weekday after the current weekday.
```

Now we need to borrow from a later chapter to make these instructions apply to the passage of time:

```
The new advance time rule is listed instead of the advance time rule in the turn sequence rules.

Test me with "z / z / z / z / z".
```

Chapter 12: Advanced Actions

Contents of *Writing with Inform*
Chapter 11: Phrases
Chapter 13: Relations
Indexes of the examples

§12.1. A recap of actions

Actions are impulses to do something, which arise sometimes through typed commands:

```
>examine tapestry
```

and sometimes through "try" phrases occurring in other rules:

```
Before examining the tapestry, try switching the ultraviolet light on.
```

Every action either succeeds or fails, though failure may not be a bad thing (something better may have happened). Besides any rules applied in the source text, actions are subject to basic realism rules. A general rule ensures that actions are rejected if the actor would need to touch something which is out of reach, or see something which is invisible; and a couple of hundred other built-in rules police individual actions. For instance, if the ACTIONS testing command has been used to switch monitoring on, then:

```
>unlock cage with watermelon
[unlocking cage with watermelon]
That doesn't seem to fit the lock.
[unlocking cage with watermelon - failed the can't unlock without the correct key rule]
```

Actions generated by "try" phrases are allowed to run "silently", which means that if nothing out of the way happens and they succeed, then nothing is printed. For instance:

```
Before examining the tapestry: say "(Switching on the lamp first.)"; silently try switching
the ultraviolet light on.
```

There are many ways to impose extra rules on actions, and we have seen three main kinds: Before rules, intended so that preliminary activities like the one above can happen before the action is tried; Instead rules, which block or divert the intention, or may cause something spectacularly different to happen; and After rules, which allow for unexpected consequences after the action has taken place.

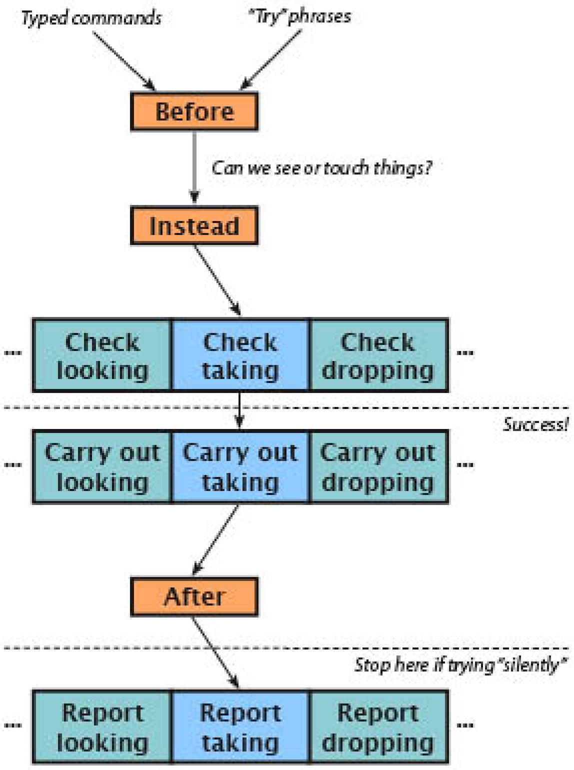
Start of Chapter 12: Advanced Actions

Back to Chapter 11: Phrases: §11.18. The value after and the value before

Onward to §12.2. How actions are processed

§12.2. How actions are processed

The following flow chart shows the natural course of events when Inform deals with a new action - a "taking" action in the case drawn. For quite a long time, the action may still fail, and it may be that nothing actually happens: but eventually a deciding line is crossed, and once that happens the action will certainly succeed.



The coloured boxes on this chart represent "rulebooks", that is, collections of rules with a common purpose. The orange boxes for Before, Instead and After were covered in the Basic Actions chapter, but the blue boxes are new. The orange boxes are where we put rules to handle unusual situations, or unexpected events: special rules to cover the opening of a container which happens to be booby-trapped, or walking through a doorway into a room where a surprise party is about to be sprung.

Blue boxes hold the mundane, everyday rules - the generic ways that particular actions behave. Every action provides these: "Check" rules, to see if it makes sense - for instance, to see that the player is not trying to take his or her own body, or a whole room, or something he or she already has; then "Carry out" rules, to actually do what the action is supposed to do - in the case of taking, to move an object into the player's possession; and finally "Report" rules, which tell the player what has happened - perhaps printing up "Taken."

When we create a new action, we add a new column to the blue rows in this diagram. As we shall see, we can also put new rules into the existing blue boxes: for instance, if we wanted to increase physical realism by forbidding the player to carry more than a certain weight, we would want to add a new "check taking" rule, and this is entirely legal.

In subsequent chapters, we will see ways to intervene at almost every point in the above diagram - from how "Can we see or touch things?" is reckoned, to each and every individual rule in all of these books. Action-processing may be the single most important thing Inform does, so the system is designed to be immensely flexible. On the other hand, that does make it a lot to take in at one look. Newcomers should probably concentrate on "Instead" and "After" as the basic tools for designing the situations turning up routinely in interactive fiction. There are guidelines at the end of this chapter offering advice on which tricks to use when it comes to more complicated needs.

Start of Chapter 12: Advanced Actions
Back to §12.1. A recap of actions
Onward to §12.3. Giving instructions to other people

§12.3. Giving instructions to other people

So far, all actions have been carried out by the player: which is fine for exploring the passive world of an empty warehouse, but less good for a drama in which other characters have to be contended with. In fact, an action can be carried out by anybody - by any instance of the "person" kind, that is, which includes all the men, women and animals in the story, and not only the player.

In interactive fiction, players conventionally ask other characters to do something with commands like so:

```
> will, go west
```

Clearly "will, go west" should not produce the same action as "go west", because a different person will be trying it: this person is called the "actor", and while the actor is ordinarily the player, here it is the character called Will. Inform distinguishes these two actions like so:

```
going west
asking Will to try going west
```

As a result, we can write rules like so:

```
Instead of asking Will to try going west, say "Will scratches his head, baffled by this talk of westward. Is not the ocean without bound?"
```

To write rules like this, we sometimes want to generalise about who is supposed to do the deed. To do this we can refer to "person asked", just as the "noun" stands for whatever noun was typed:

```
Instead of asking somebody to try taking something, say "I don't think we ought to tempt [the person asked] into theft, surely?"
```

So if the player types "Algy, take sandwich", the "person asked" would be Algy; the "noun" would be the sandwich; and there would be no "second noun".

⬆ Start of Chapter 12: Advanced Actions

⬅ Back to §12.2. How actions are processed

➡ Onward to §12.4. Persuasion

⬇ Example 184: ★ **Virtue** Defining certain kinds of behavior as inappropriate, so that other characters will refuse indignantly to do any such thing.

⬇ Example 185: ★★★ **Latris Theon** A person who can accept instructions to go to new destinations and move towards them according to the most reasonable path.

§12.4. Persuasion

"Asking ... to try ..." actions run through their Before and Instead rules like any other actions, but then (if no rule has intervened) something different happens: Inform has to decide whether the person asked consents to try the action or not. By default, the answer is always no, and text like the following will be printed:

```
> will, go west
Will has better things to do.
```

However, we can intervene to make the answer "yes", using a special kind of rule which produces a yes/no answer. The following examples show how we can give broad or narrow permission, as we choose:

```
Persuasion rule for asking people to try going: persuasion succeeds.
Persuasion rule for asking Will to try going west: persuasion succeeds.
```

Such a rule can either declare that "persuasion succeeds", or that "persuasion fails", or make no decision and leave it to another rule to say. If it decides that persuasion fails, it is also allowed to say something, describing why: in that event, the standard message ("Will has better things to do.") is suppressed. For example,

```
Persuasion rule for asking Will to try going:
    say "Will looks put out, and mutters under his breath.";
    persuasion fails.
```

The following rule, which is really only suitable for testing, makes everybody infinitely obliging:

```
Persuasion rule for asking people to try doing something: persuasion succeeds.
```

Supposing that Will does decide to cooperate, a new action is generated:

```
Will going west
```

and this is then subject to all of the usual action machinery. For instance, we could write a rule such as:

```
Instead of Will going west, say "He runs out into the waves, but soon returns, rueful."
```

So in this case the new action ("Will going west") failed: but the original action, "asking Will to try going west", is still deemed to have succeeded - after all, Will *did* try. To put it more formally, "asking X to try A" succeeds if the persuasion rules succeed, and otherwise fails.

Note also that "Instead of..." rules written for other people will be treated by Inform as failures, even if we write something like

```
Instead of Will pulling the cord:
    say "The bell rings."
```

and thus may produce unsatisfactory results such as

```
>WILL, PULL CORD
The bell rings.

Will is unable to do that.
```

If we wish to write new successful actions for another character, we will need to create appropriate carry out and report rules for them: these will be explained in the sections to follow.

(Finally, note that the mechanism Inform uses to see if we have printed a refusal message of our own, in the event of persuasion rules failing, can be fooled if we write a persuasion rule explicitly ending with a "[paragraph break]" text substitution.)

⬆ Start of Chapter 12: Advanced Actions

⬅ Back to §12.3. Giving instructions to other people

➡ Onward to §12.5. Unsuccessful attempts

⬇ Example 186: ★ **The Hypnotist of Blois** A hypnotist who can make people obedient and then set them free again.

⬇ Example 187: ★ **Police State** Several friends who obey you; a policeman who doesn't (but who takes a dim view of certain kinds of antics).

§12.5. Unsuccessful attempts

Suppose, finally, that Will not only consents to try the action, but it also survives its passage through Before and Instead rules. What happens then? In principle, what happens to Will is exactly what would have happened to the player in his place. For instance:

```
> will, go east
Will leaves to the east.
```

If on the other hand Will's attempt is frustrated because one of the checking rules stops him, then Will's action fails. For instance, if Will tries going northeast but there is no room to northeast, one of the rules checking the "going" action will stop him. We will then see this:

```
> will, go northeast
Will is unable to do that.
```

This is rather a generic message, and we may want something more interesting. We can provide that using yet another special kind of rule:

```
Unsuccessful attempt by Will going: say "Will blunders around going nowhere, as usual."
```

Even that is still a little generic, though, because it treats all of the various ways that "going" can fail as the same. If we have ACTIONS switched on, we can see what goes on behind the scenes when we ask Will to walk into a door:

```
>will, go west
[asking Will to try going west]
[(1) Will going west]
[(1) Will going west - failed the can't go through closed doors rule]
Will blunders around going nowhere, as usual.
[asking Will to try going west - succeeded]
```

(The "(1)" lets us know that a new action is starting during the old one, and before the old one finishes: sometimes we go up to three or four deep, though seldom more in practical cases.) We can now rewrite the "unsuccessful attempt" rule like so:

```
Unsuccessful attempt by Will going:
    if the reason the action failed is the can't go through closed doors rule, say "Will looks doubtful and mumbles about doors.";
    otherwise say "Will blunders around going nowhere, as usual."
```

The value "reason the action failed" is set to whichever checking rule threw out the action which Will tried. The names of these rules try to be self-explanatory - at any rate, those with gnomic names are not useful for this sort of thing, and can be ignored - and can be found out either using ACTIONS or by consulting the Actions index.

Finally, note that "unsuccessful attempt" rules apply only when the person in question is being asked to perform the action by somebody else - as in the examples above.

Start of Chapter 12: Advanced Actions

Back to §12.4. Persuasion

Onward to §12.6. Spontaneous actions by other people

Example 188: ★★ **Generation X** A person who goes along with the player's instructions, but reluctantly, and will get annoyed after too many repetitions of the same kind of unsuccessful command.

§12.6. Spontaneous actions by other people

The player's actions happen not only when he types a command, but can also happen spontaneously as a result of a "try" phrase.

```
try going west
try asking Will to try going west
```

The latter might, of course, result in Will trying going west: or it might not - that depends on the persuasion rules. But as the author, we have the ultimate powers of persuasion, and can make Will act in any way we like, without asking:

```
try Will going west
```

Nobody in the simulated world requested this: it is an impulse felt by Will alone, so that - from the player's point of view - Will is acting spontaneously. The player need not be anywhere nearby, and may never know what happened. Recall that when actions work their way down through the flow-chart, they are stopped before reaching the "report" stage - when the player is told about them - if they are running "silently". This is also where Inform stops an action which is not witnessed by the player.

To repeat a point in the previous section: "unsuccessful attempt" rules do not apply to actions which the author has caused to happen, using "try". When such actions fail, they invoke no special set of rules. Indeed, when "try" causes somebody other than the player to try an action, nothing will be printed to report back on success or failure. If Will can't go west, that's his problem.

Note that the text "try Will going west" involves the actor's name immediately placed next to the action he is to try, which in a very few cases might cause ambiguities. If the actor's name contains a participle like "going" - say, if Will's full name turned out to be Mr Will Going - then we would have to write out the action name in full, using "trying" to clarify matters:

```
try Will Going trying going west
```

Start of Chapter 12: Advanced Actions

Back to §12.5. Unsuccessful attempts

Onward to §12.7. New actions

Example 189: ★ **IQ Test** Introducing Ogg, a person who will unlock and open a container when the player tells him to get something inside.

Example 190: ★★★★ **Boston Cream** A fuller implementation of Ogg, giving him a motivation of his own and allowing him to react to the situation created by the player.

§12.7. New actions

It is not often that we need to create new actions, but a large work of interactive fiction with no novelty actions is a flavourless dish. Here we shall create an action for photographing things.

```
The Ruins is a room. "You find this clearing in the rainforest oddly familiar." The camera is in the Ruins. "Your elephantine camera hangs from a convenient branch."
```

```
Photographing is an action applying to one visible thing and requiring light.
```

In theory that text is already sufficient to make the new action, but what we have so far is rudimentary to say the least. The two qualifications give Inform the useful information that we cannot photograph in the dark, and that we need to be photographing something - not, as in the case of waiting or taking inventory, acting without reference to any particular thing external to ourselves.

The word "visible" here tells Inform that we do not need to be able to touch the thing in question: a line of sight is good enough. These two stipulations were necessary because the default arrangement is that any object must be in touching range, and that most actions can be performed in darkness. (Also, note that if you invent an action which needs to apply to directions like "north" or "south", you need to make this apply to visible things, because the object used inside Inform to represent the idea of "north" can be seen but not touched. So for understanding purposes, "visible thing" is understood as meaning any visible thing or direction: it's more general than "thing", not more specific.)

Occasionally, when writing general rules about actions, it can be useful to find out what the current action's requirements are: the following conditions do what they suggest.

if action requires a touchable noun:

This condition is true if the action being processed is one whose (first) noun is an object which needs to be touchable by the actor. For example, it's true for "taking", but false for "examining".

if action requires a touchable second noun:

This condition is true if the action being processed is one whose second noun is an object which needs to be touchable by the actor. For example, it's true for "putting the brick in the sack", but false for "throwing the brick at the window".

if action requires a carried noun:

This condition is true if the action being processed is one whose (first) noun is an object which needs to be carried by the actor. For example, it's true for "dropping", but false for "taking".

if action requires a carried second noun:

This condition is true if the action being processed is one whose second noun is an object which needs to be carried by the actor.

if action requires light:

This condition is true if the action being processed is one which can only be performed if the actor has light to see by. For example, it's true for "examining", but false for "dropping".

As further examples, here we create "blinking" and "scraping X with Y". Note the use of "it" to indicate that the name of an object should go here.

```
Blinking is an action applying to nothing. Scraping it with is an action applying to two things.
```

The photographing action now exists, but with two provisos: (a) it never happens, because Inform does not know what commands by the player should cause it, and (b) even if it were to happen, nothing would follow, because Inform does not know what to do. (There are no check, carry out or report rules yet.)

The first problem is easily overcome:

```
Understand "photograph [something]" as photographing.
```

We will return to the whole subject of parsing, as this process of understanding the player's commands is called, later. But this gives the gist of it.

★ *See* **Understand** *for the full story*

Start of Chapter 12: Advanced Actions

Back to §12.6. Spontaneous actions by other people

Onward to §12.8. Irregular English verbs

Example 191: ★ **Red Cross** A DIAGNOSE command which allows the player to check on the health of someone.

Example 192: ★★★ **Frizz** Liquid flows within containers and soaks objects that are not waterproof; any contact with a wet object can dampen our gloves.

Example 193: ★★★ **3 AM** A shake command which agitates soda and makes items thump around in boxes.

§12.8. Irregular English verbs

Our three example actions can be recognised in play using the following:

```
Understand "photograph [something]" as photographing.
```

```
Understand "blink" as blinking.
```

```
Understand "scrape [something] with [something]" as scraping it with.
```

The last of these examples shows why Inform does not risk generating this automatically: English is so full of irregular verbs. Inform could have guessed "blink" and "photograph", but might then have opted for "scrap" instead of "scrape".

Inform does risk automatically generating the past participle of an action. (Many past participles are never needed, so the stakes are lower if Inform gets this wrong.) What usually happens is that the "-ing" is replaced with "-ed", thus photographed, blinked, scraped - but Inform has a dictionary of some 460 irregular exceptions, such as caught, fled, crossbred, taken, woven. So with luck Inform will guess correctly. If not, we can get around this like so:

```
Squicking is an action with past participle squacked, applying to one thing.
```

Start of Chapter 12: Advanced Actions

Back to §12.7. New actions

Onward to §12.9. Check, carry out, report

§12.9. Check, carry out, report

The normal behaviour of an action is specified by its three associated rulebooks - check, carry out and report. In the case of our "photographing" example, these rulebooks will be:

> **Check photographing**. Here, for instance, we need to verify that the player has the camera. If any of our checks fail, we should say why and stop the action. If they succeed, we say nothing.

> **Carry out photographing**. At this stage no further checking is needed (or allowed): the action now definitively takes place. At this point we might, for instance, deduct one from the number of exposures left on the film in the camera, or award the player points for capturing something interesting for posterity. But we should say nothing.
>
> **Report photographing**. At this stage no further activity is needed (or allowed): whatever effect the action had, it has happened and is now over. All we can do is to say what has taken place.

So far we have not really gone into the business of what rulebooks are, and we don't do so here either - suffice to say that we can now create whatever rules we need:

```
A check photographing rule:
    if the camera is not carried:
        say "You can hardly photograph without a camera, now can you?" instead.
```

In fact, writing "a check photographing rule" is over-formal. We can more simply label our rules like so:

```
Check photographing:
    if we have photographed the noun:
        say "You've already snapped [the noun]." instead.

Report photographing: say "Click!"
```

For the sake of brevity, photography has no interesting consequence (no points to be won, no film to use up), so there are no carry out rules here. Note the way we used the word "instead" once again to stop actions in their tracks.

We can continue to add rules at any point, and a classic thing that happens when testing a new work is that the designer realises there is a case which has not been thought of:

```
Check photographing:
    if the noun is the camera:
        say "That would require some sort of contraption with mirrors." instead.
```

Start of Chapter 12: Advanced Actions

Back to §12.8. Irregular English verbs

Onward to §12.10. Action variables

Example 194: ★ **The Dark Ages Revisited** An electric light kind of device which becomes lit when switched on and dark when switched off.

Example 195: ★★ **Paddington** A CUT [something] WITH [something] command which acts differently on different types of objects.

Example 196: ★★★ **Delicious, Delicious Rocks** Adding a "sanity-check" stage to decide whether an action makes any sense, which occurs before any before rules, implicit taking, or check rules.

Example 197: ★★★ **Noisemaking** Creating a stage after the report stage of an action, during which other characters may observe and react.

§12.10. Action variables

For some complex situations, it can be useful to keep track of a few values throughout the processing of the action. This is not an everyday occurrence: in the Standard Rules, for instance, only two or three out of 90 actions need to do this. But suppose we want to write a more deluxe version of our "photographing" action. This time, rather than having a single thing called the "camera", we will provide a whole range of possible cameras, varying in quality:

```
Photographing is an action applying to one visible thing and requiring light. Understand "photograph [something]" as photographing.

The Studio is a room. Sally is a woman in the Studio. A foam-lined tote bag is in the Studio.

A camera is a kind of thing. A camera has a number called picture quality. The digital SLR camera is a camera in the tote bag. The player carries a camera called the instant one-shot camera. The picture quality of the SLR camera is 10. The picture quality of the one-shot is 2. Definition: a camera is sharp if its picture quality is 5 or more.
```

And we will want the photographing action to have the player use the best-quality camera which comes to hand. We will give the action a variable called the 'camera photographed with', thus:

```
The photographing action has an object called the camera photographed with.
```

Every action's variables must be named differently from those of all other actions, because there are some "before" rules (for instance) which take effect for many different actions, and which might need access to any of their variables. So action variables should be named in a way marking out to which action they belong. The best way to do this is to include the past participle of the action name - just as "camera photographed with" contains the past participle "photographed" of the action "photographing".

This value is created when the action begins, and disappears when the action ends. (If the action should happen a second time before the first time was completed, a second copy of the value is created, leaving the original undisturbed.) When the action begins, the value starts out as something neutral - so if it is a number, it starts out as 0, if a text, it starts out as the blank text "", and so on. Here it is an object, so it starts out as nothing - the value meaning no object at all. But of course we want to give it a value ourselves. We can do that using the "setting action variables" rulebook. For instance:

```
Setting action variables for photographing:
    now the camera photographed with is the sharpest camera which is carried by the actor.
```

The "setting action variables" rulebook is run through before even the before rules, and it has no power to stop or change the action. Its rules should say nothing and do nothing other than to set rulebook variables like this one. Note that it is intended to work for any actor, not only the player: so rather than referring to the player as the performer of the action, we need to write "the actor", as in the example above. (See subsequent sections for more on actors.)

We can now write rules such as:

```
A check photographing rule:
    if the camera photographed with is nothing:
        say "You can hardly photograph without a camera, now can you?" instead.
```

Only rules to do with the photographing action - before, instead, after, check, carry out, or report rules, and so on - are allowed to see the 'camera photographed with' value: it's the private property of the action.

A further elaboration allows us to make rules about photographing neater to write. If we create our variable like so:

```
The photographing action has an object called the camera photographed with (matched
as "using").
```

...then we are now allowed to add an optional 'using ...' clause onto a description of the action. The clause has to be introduced with a single word: here, it's 'using'. For instance, we could write rules such as

```
Instead of photographing something using the one-shot camera:
    say "But you promised to give this to Sally's nephew."

Check photographing something using the noun:
    say "That would require some sort of contraption with mirrors." instead.

Report photographing something using a sharp camera:
    say "You feel cool and important as the shutter clicks."
```

(This is the method used by the Standard Rules to attach optional clauses such as 'to', 'with' and 'through' to the going action.)

⬆ Start of Chapter 12: Advanced Actions

⬅ Back to §12.9. Check, carry out, report

➡ Onward to §12.11. Making actions work for other people

⬇ Example 198: ★ **Removal** TAKE expanded to give responses such as "You take the book from the shelf." or "You pick up the toy from the ground."

⬇ Example 199: ★ **Further Reasons Why All Poets Are Liars** The young William Wordsworth, pushing a box about in his room, must struggle to achieve a Romantic point of view.

⬇ Example 200: ★ **The Second Oldest Problem** Adapting the going action so that something special can happen when going from a dark room to another dark room.

⬇ Example 201: ★★ **Puff of Orange Smoke** A system in which every character has a body, which is left behind when the person dies; attempts to do something to the body are redirected to the person while the person is alive.

⬇ Example 202: ★★★ **Croft** Adding special reporting and handling for objects dropped when the player is on a supporter, and special entering rules for moving from one supporter to another.

---

§12.11. Making actions work for other people

The "photographing" action now works very nicely when the player does it. But not when others try. Suppose that neither the player, nor Clark Gable, is holding the camera:

```
>photograph clark
You can hardly photograph without a camera, now can you?

>clark, photograph me
>
```

An uncanny silence. What has happened is that the rules written so far are all implicitly restricted to the player only. This is because when we write -

```
Check photographing:
    if the camera is not carried:
        say "You can hardly photograph without a camera, now can you?" instead.
```

the action is "photographing", not "Clark photographing". In the next few sections we shall see how to make the rules work nicely for everybody. This is a little bit harder, so it should be noted right away that in many projects there is no need. In a story which has no other characters who succumb to persuasion, for instance, only the player will ever try the action.

Start of Chapter 12: Advanced Actions

Back to §12.10. Action variables

Onward to §12.12. Check rules for actions by other people

Example 203: ★★ **The Man of Steel** An escaping action which means "go to any room you can reach from here", and is only useful to non-player characters.

Example 204: ★★★ **Trying Taking Manhattan** Replacing the inventory reporting rule with another which does something slightly different.

Example 205: ★★★★ **Under Contract** Creating a person who accepts most instructions and reacts correctly when a request leads implicitly to inappropriate behavior.

§12.12. Check rules for actions by other people

If we want to impose the restriction about carrying the camera on other people, we need a rule like the following:

```
Check someone photographing: if the person asked does not carry the camera, stop the
action.
```

Implicitly, that "someone" excludes the player. Note that we say nothing in this rule, stopping the action without a word: after all, Clark might well be out of sight when trying this. If he is within sight, then we read:

```
>clark, photograph me
Clark Gable is unable to do that.
```

We saw before that Inform's built-in rules all have handy names (the "can't drop what's already dropped rule", and such), and that these are useful when writing better "unable to..."

messages. So for a deluxe version, we end up with:

```
Check someone trying photographing (this is the other people can't photograph without
the camera rule): if the person asked does not carry the camera, stop the action.
```

And now, with ACTIONS on, we find that:

```
>clark, photograph me
[asking Clark Gable to try photographing yourself]
[(1) Clark Gable photographing yourself]
[(1) Clark Gable photographing yourself - failed the other people can't photograph
without the camera rule]
Clark Gable is unable to do that.
[asking Clark Gable to try photographing yourself - succeeded]
```

which means that we could have, say,

```
Unsuccessful attempt by Clark photographing:
    if the reason the action failed is the other people can't photograph without the camera
rule, say "Clark is too suave to be embarrassed. 'Frankly, my dear, I don't have a
camera.'";
    otherwise say "Clark tries, and fails, to take a photograph."
```

Start of Chapter 12: Advanced Actions

Back to §12.11. Making actions work for other people

Onward to §12.13. Report rules for actions by other people

Example 206: ★ **Get Axe** Changing the check rules to try automatically leaving a container before attempting to take it. (And arranging things so that other people will do likewise.)

Example 207: ★★★ **Barter Barter** Allowing characters other than the player to give objects to one another, accounting for the possibility that some items may not be desired by the intended recipients.

§12.13. Report rules for actions by other people

Report rules for the player's actions are easy to write, and for many actions, they are not much harder for other people either:

```
Report photographing: say "Click!"
Report someone photographing: say "Click! [The person asked] takes a snapshot of [the
noun]."
```

But once other people are involved, we have to go to some trouble to get all of the possibilities right. Here is a case which did not immediately occur to the author of the "going" action, for instance:

```
>get in cage
You get into the cage.
>clark, get in automobile
Clark Gable gets into the automobile.
```

```
>clark, push cage west
Clark Gable goes west in the automobile, pushing the cage in front, and you along too.

The Lot (in the cage)
In the Lot you can see an automobile (in which is Clark Gable).
```

We said before that report rules are skipped if the action is running "silently", or if the action is one that the player does not witness. But that is also a tricky concept. Inform's doctrine is that you witness an action if you can see any of the actor, the noun or the second noun at either the beginning or the end of the action; except that being able to see a backdrop does not count. Thus if Clark Gable, in Beverly Hills, photographs the Hollywood sign then we do not witness this from Sunset Boulevard merely because we, too, can see the Hollywood sign.

While the report rules for actions by the player must actually report something, report rules for other people's actions are under no such obligation. For instance, if Clark unlocks a door from the other side to the player, then this counts as an action that the player witnesses - and after all, it could be argued that the player should hear the key turning in the lock - but in fact the standard rules for reporting locking choose to say nothing.

⬆ Start of Chapter 12: Advanced Actions

⬅ Back to §12.12. Check rules for actions by other people

➡ Onward to §12.14. Actions for any actor

⬇ Example 208: ★★ **The Man of Steel Excuses Himself** Elaborating the report rules to be more interesting than "Clark goes west."

⬇ Example 209: ★★★ **Fate Steps In** Fate entity which attempts to make things happen, by hook or by crook, including taking preliminary actions to set the player up a bit.

§12.14. Actions for any actor

In the previous sections, we created a new action by providing one set of rules for the player and another for anybody else who might try to perform it. These rules began with action descriptions in one of the following forms:

```
Instead of taking a container, ...
Instead of P taking a container, ...
```

The first form implies that the player must be performing the action: the second allows for any person matching P to be the action, except that this person must not be the player. That means that all rules seen so far either affect only the player, or only other people.

This is often convenient, but sometimes we need to set up a complicated action which really does work in the same way for every actor - for instance, the built-in Inform actions provided by the Standard Rules aim to do this. We can write such rules thus:

```
Instead of an actor taking a container, ...
```

Here the rule applies to anyone who tries taking a container, player or not. Inside such a rule, the special value 'the actor' is the person performing the action. For instance, the Standard Rules include this one:

```
Carry out an actor wearing (this is the standard wearing rule):
    now the actor wears the noun.
```

Start of Chapter 12: Advanced Actions
Back to §12.13. Report rules for actions by other people
Onward to §12.15. Out of world actions

§12.15. Out of world actions

The actions seen so far are all impulses causing the protagonist inside the fictional world to do something, or at least try to. But when the player types "quit" or "save", that is not a request for anything to happen in the fictional world: it is an instruction to the program simulating that world. In fact, just the same, such requests are treated as actions, but of a special category called "out of world" actions. They do not cause time to pass by, so the turn counter does not advance, nor does this command cycle count as a turn at all; and they are altogether exempt from "Before", "Instead" and "After" rules. Only the player is allowed to try them.

We can also create new out-of-world actions. Suppose we want a dialogue like so:

```
>ROOMS
You have been to 1 out of 8 rooms.
```

Here is a complete implementation:

```
Requesting the room tally is an action out of world.
Report requesting the room tally: say "You have been to [number of visited rooms] out of [number of rooms] room[s]."
Understand "rooms" as requesting the room tally.
```

It is important not to use "out of world" actions for anything affecting what goes on in the fictional world, or realism will collapse, and action-processing may also fail to work in the usual way. "Out of world" actions should be reserved for providing commands like ROOMS, which monitor events rather than participate in them.

Start of Chapter 12: Advanced Actions

Back to §12.14. Actions for any actor

Onward to §12.16. Reaching inside and reaching outside rules

Example 210: ★ **Spellbreaker** P. David Lebling's classic "Spellbreaker" (1986) includes a room where the game cannot be saved: here is an Inform implementation.

Example 211: ★★★ **A point for never saving the game** In some of the late 1970s "cave crawl" adventure games, an elaborate scoring system might still leave the player perplexed as to why an apparently perfect play-through resulted in a score which was still one point short of the supposed maximum. Why only 349 out of 350? The answer varied, but sometimes the last point was earned by never saving the game - in other words by playing it right through with nothing to guard against mistakes (except perhaps UNDO for the last command), and in one long session.

§12.16. Reaching inside and reaching outside rules

The flow chart back at the start of this chapter shows that, early on in processing an action (between Before and Instead), Inform asks the question "Can we see or touch things?" This is where it enforces the requirements in the action's definition:

```
Photographing is an action applying to one visible thing and requiring light.
Scraping it with is an action applying to two things.
```

Seeing and touching are two different questions, which Inform answers in different ways. We shall see ways to modify or entirely alter what can be seen using the "deciding the scope of something" activity when we get to the Understanding and Activities chapters, and later in this chapter we will change the definition of touchability. What both have in common is that they are complicated questions, affected by the circumstances. We cannot simply declare that the player can touch a given lever, or can see in a given room: we must arrange for there to be no barriers between the player and the lever, or for there to be a light source in the room.

An example of rules applying to given objects is provided by the way that Inform decides whether the player can reach something or not. For instance, suppose the following:

```
The Laboratory is a room. In the Laboratory is a conical flask. The flask is closed and
transparent. In the flask is an antibumping granule.
```

The player will be able to examine the granule but not to take it, as that would require reaching through glass. Suppose the player does type TAKE GRANULE: then Inform looks for potential barriers between the player and the granule, and of course finds the conical flask. If, as in this case, the thing to be touched is on the inside, then Inform asks the "reaching inside" rules for permission. There are two reaching inside rules built in to Inform:

```
can't reach inside rooms rule
can't reach inside closed containers rule
```

and in fact the second of these rules will cause the taking action to fail, because the conical flask is a closed container. (The other rule has to do with a player in one room able to see another room through, say, a telescope - merely having a line of sight doesn't give the ability to reach into the frame.)

Symmetrically, Inform also has "reaching outside" rules, used if the player is inside something and wants to reach an object in the wider room. (From a bed, probably yes; from a cage, probably no.) This ordinarily contains just one rule:

```
can't reach outside closed containers rule
```

Start of Chapter 12: Advanced Actions

Back to §12.15. Out of world actions

Onward to §12.17. Visible vs touchable vs carried

Example 212: ★★ **Carnivale** An alternative to backdrops when we want something to be visible from a distance but only touchable from one room.

§12.17. Visible vs touchable vs carried

To recap, actions are created like so:

```
Photographing is an action applying to one visible thing and requiring light.
Depositing it in is an action applying to two things.
Taking inventory is an action applying to nothing.
```

Actions can involve up to two different things. We can place additional requirements on any of these things by describing them as a "visible thing", "touchable thing" or "carried thing". (If we simply say "thing" or "things", as in the second example, Inform assumes the requirement to be "touchable".) These three conditions are increasingly strong:

- To be "visible", something needs only to be possible to refer to by the player, which in practice means that it must be visible to the player-character. The noun or second noun produced by any action resulting from a command at the keyboard will always satisfy this minimal condition.

- To be "touchable", the player-character must be able to physically touch the thing in question: this normally means that it must be in the same room, and there must be no physical barriers in between.

- To be "carried", the player-character must (directly) carry the thing in question. (But if the player types a command using an action requiring something "carried", like WEAR HAT, the thing in question - the hat - will sometimes be picked up automatically. This is called "implicit taking", and results in text like "(first taking the top hat)" being printed.)

If an action involves two things, they need not have the same requirement as each other:

```
Waving it at is an action applying to one carried thing and one visible thing.
```

Thus to "wave magic wand at banyan tree", the player must be holding the wand, but need only be able to see the tree.

Note one special case. Requirements on touchability are waived in the case of "try" actions applied to people other than the player where the things they would need to touch are doors or backdrops. (This is a compromise to avoid difficulties arising from the ambiguous locations of such items.)

⬆ Start of Chapter 12: Advanced Actions

⬅ Back to §12.16. Reaching inside and reaching outside rules

➡ Onward to §12.18. Changing reachability

⬇ Example 213: ★★ **Eddystone** Creating new commands involving the standard compass directions.

⬇ Example 214: ★★★ **Slogar's Revenge** Creating an amulet of tumblers that can be used to lock and unlock things even when it is worn, overriding the usual requirement that keys be carried.

§12.18. Changing reachability

The question of what the player can, and cannot, reach to touch is important in interactive fiction. It contains some of the subtlest ideas in the model world, though they often go unnoticed. For instance, if a key is on a shelf which is part of a closed box, can we reach for the key? This comes down to whether the shelf, described only as "part of" the box, is on the inside or the outside: and in fact, because it cannot know which is the case, Inform allows either. So in general it is best to regard "parts" as being exterior parts, but to avoid having parts on containers that might in the course of play be closed up with the player inside.

We can, if we wish, change the principles of what can be touched by writing new reaching inside or reaching outside rules. Returning to the example of the conical flask:

```
A rule for reaching inside the flask: say "Your hand passes through the glass as if it were not there, chilling you to the bone."; allow access.
```

(Or this could equally be called "a reaching inside rule for the flask".) More generally, we could give the usual flexible description of what the rule applies to:

```
A rule for reaching inside open containers: say "Your hands seem enigmatically too large for [the container in question]."; deny access.
```

The "container in question" is the one to which the rule is being applied. Note that a reaching inside rule can "deny access" (stopping with failure), or "allow access" (stopping with success), or neither, in which case the decision is left up to any subsequent rules in the rulebook to make. If none of them decide, access is allowed.

If it seems possible that these rules will be employed by people other than the player, then we need to write them a little more carefully, and in particular we need to ensure that they print nothing for other people. In the first case below, anybody can reach through the glass; in the second case, only the player cannot reach into open containers.

```
A rule for reaching inside the flask:
    if the person reaching is the player, say "Your hand passes through the glass as if it were not there, chilling you to the bone.";
    allow access.
```

```
A rule for reaching inside open containers:
    if the person reaching is the player:
        say "Your hands seem enigmatically too large for [the container in question].";
        deny access.
```

The "person reaching" is, as its name suggests, the person trying to reach through the barrier in question.

⬆ Start of Chapter 12: Advanced Actions

⬅ Back to §12.17. Visible vs touchable vs carried

➡ Onward to §12.19. Changing visibility

⬇ Example 215: ★ **Magneto's Revenge** Kitty Pryde of the X-Men is able to reach through solid objects, so we might implement her with special powers that the player does not have...

⬇ Example 216: ★ **Waterworld** A backdrop which the player can examine, but cannot interact with in any other way.

⬇ Example 217: ★★★ **Dinner is Served** A window between two locations. When the window is open, the player can reach through into the other location; when it isn't, access is barred.

§12.19. Changing visibility

Ordinarily, Inform has a simple model for visibility: it is either fully light or it is fully dark, and certain actions are impossible in the dark, such as examining something.

We first need to remember that darkness affects what actions are even tried, as far as the player's typed commands go. If the player is in a dark room, and there is a screwdriver on the floor, the command EXAMINE SCREWDRIVER will not try any action: the screwdriver is not "in scope", which means that the parser thinks the player does not have any means of knowing it exists. (The rules for scope can be modified - see the chapter on Activities.) But let's suppose that the player types EXAMINE BOOK, and is holding the book in question. The book is now "in scope", so the action "examining the book" is tried.

Some actions require light to be present, and "examining" is one of those. So Inform consults the visibility rules to see if it can go ahead. By default, there is only one visibility rule, which says "yes" in the light and "no" in darkness. Here, though, we create another one:

```
Visibility rule when in darkness:
    if examining the book:
        say "You have to squint. Still...";
        there is sufficient light;
    there is insufficient light.
```

A visibility rule must always conclude "there is sufficient light", or "there is insufficient light", or else do nothing and leave it to other rules to decide.

It is a possibly unexpected fact that "looking" does not require light, but instead behaves differently in darkness - it prints a pseudo-room-description such as

```
Darkness
It is pitch dark, and you can't see a thing.
```

instead of printing the description of the player's current room. This means that the "looking" action is unaffected by visibility rules. All the same, what "looking" does in the dark can be changed by using the two activities "printing the name of a dark room" and "printing the description of a dark room" (see the Activities chapter for details).

Start of Chapter 12: Advanced Actions

Back to §12.18. Changing reachability

Onward to §12.20. Stored actions

Example 218: ★ **Flashlight** Visibility set so that looking under objects produces no result unless the player has a light source to shine there (regardless of the light level of the room).

§12.20. Stored actions

As we have seen, to describe an action fully takes a complicated little bundle of information - we need to know what is to be done, who will do it, and what it will be done to. There are times when we would like to remember an action and look back on it later (perhaps many turns later, after many other actions have taken effect) - but this is not easy to do with only the techniques we have seen so far. There are quite a few cases to get right, and it would be easy to not store quite enough of the details.

Fortunately, Inform provides a kind of value called "action" which can do all of this automatically. (In older versions of Inform this was called "stored action", but the word "stored" is now unnecessary, and makes no difference.) As with most other kinds of value, actions can be held in variables, "let" values, properties or table columns. For example:

```
The best idea yet is an action that varies.
```

creates a variable called "the best idea yet" which holds an action.

This will normally be created holding the default value - the player waiting. We really only have two ways to make more interesting actions. One is by typing them out explicitly, like so:

```
now the best idea yet is pushing the button;
```

Here "pushing the button" is a constant of the kind "action", so it goes into happily into "best idea yet" in the same way that a number like 3 could go into a number that varies. The action must be specific in every respect, so "taking something" or "doing something" will not work - "taking something" is really a general description of many possible actions, not an action in its own right.

The other way to produce a useful action is:

current action ... action

This phrase produces the action currently being processed as a value - it literally stores the action, and remembers, if necessary, the exact wording of the player's command at the time it was stored - so that even actions arising from commands like LOOK UP X100 IN THE CODE BOOK can be stored faithfully. Examples:

```
let the present whim be the current action;
say "How you would like to be [current action].";
```

This only makes sense if an action is currently going on, so it shouldn't be used in "every turn" rules, for instance.

So much for making actions: now for making use of them. The first obvious idea is to store up an action for several turns and then have it take effect later. That's easily done: just as we can "try" any action written out explicitly, so we can also try a stored one. The phrase to do this has exactly the same wording either way, since it does the same thing either way.

But actions can still be useful even if we never intend to try them. For one thing, we can say them, and this produces a fairly natural description of what the action is:

```
Before doing something in the presence of the bearded psychiatrist: say "'Zo, the
subject vishes to engage in [the current action]. Zis is very interesting.'"
```

will produce text such as:

```
"So, the subject vishes to engage in rubbing the fireman's pole. Zis is very interesting."
```

One of Inform's most convenient features is its ability to test if the action being processed matches vague or complicated descriptions of whole classes of actions. For example,

```
if the best idea yet is taking something, ...
```

works even though "taking something" is not a single action; it's a description which could apply to many different actions (taking a box, taking a ball, and so on). What Inform tests is whether the "best idea yet" value, a single action, fits this description or not. We can be even vaguer:

```
if the best idea yet is doing something to the lever, ...
```

Just occasionally, this can lead to ambiguities. For instance,

```
if the current action is wearing something, ...
```

fails because Inform thinks "wearing" is meant in the sense of the current action having clothes on, so it produces a problem message. To avoid this, simply write:

```
if the current action is trying wearing something, ...
```

which can't be misunderstood. Something else to be aware of is that the terms "actor", "noun" and so on will refer to that action: for instance, in

```
if the best idea yet is taking the noun, ...
```

"noun" here refers to the noun in "best idea yet", not to its meaning outside of this phrase (if indeed it has such a meaning).

When dealing with actions, we sometimes want to know what they are dealing with. We can extract this information using the following phrases:

action name part of (action) **... action name**

This phrase produces the action name part of an action. Example: suppose the current actor is Algy, who is throwing the brick at Biggles. Then

```
action name part of the current action = throwing it at action
```

noun part of (action) **... object**

This phrase produces the (first) noun of an action. Example: suppose the current actor is Algy, who is throwing the brick at Biggles. Then

```
noun part of the current action = the brick
```

If the noun is something other than an object, this produces just "nothing", the non-object.

second noun part of (action) **... object**

This phrase produces the second noun of an action. Example: suppose the current actor is Algy, who is throwing the brick at Biggles. Then

```
second noun part of the current action = Biggles
```

If the second noun is something other than an object (for instance for the command SET DIAL TO 3417 it would be the number 3417), this produces just "nothing", the non-object.

actor part of (action) **... object**

This phrase produces the person who would be carrying out the action if it were being tried. Example: suppose the current actor is Algy, who is throwing the brick at Biggles. Then

```
actor part of the current action = Algy
```

The following phrase is a convenient shorthand form:

if (action) **involves** (object)**:**

This condition is true if the object appears as any of the actor, the noun or the second noun in the action. Example:

```
if the current action involves Algy
```

would be true for "give revolver to Algy", "Algy trying flying the Sopwith Camel", "examine Algy" and so on, but false for "ask Raymond about secret airfield".

action of (an action) **... action**

This phrase is now seldom needed. It produces a literally typed action as a value. Example:

```
now the best idea yet is the action of pushing the button;
```

Nowadays in most contexts we can just type "pushing the button" as a value, and that will work fine, so this phrase is retained only to keep old code working.

⬆ Start of Chapter 12: Advanced Actions

⬅ Back to §12.19. Changing visibility

➡ Onward to §12.21. Guidelines on how to write rules about actions

⬇ Example 219: ★ **Bosch** Creating a list of actions that will earn the player points, and using this both to change the score and to give FULL SCORE reports.

⬇ Example 220: ★ **Cactus Will Outline Us All** For every character besides the player, there is an action that will cause that character to wither right up and die.

⬇ Example 221: ★★★ **Actor's Studio** A video camera that records actions performed in its presence, and plays them back with time-stamps.

⬇ Example 222: ★★★ **Anteaters** The player carries a gizmo that is able to record actions performed by the player, then force him to repeat them when the gizmo is dropped. This includes storing actions that apply to topics, as in "look up anteater colonies in the guide".

§12.21. Guidelines on how to write rules about actions

Looking at the action-processing diagram, there seem to be a bewildering number of ways to intervene. For instance, suppose it must be fatal to pick up a land mine. All six of the following rules would do the business:

```
Before taking the land mine: end the story saying "Bang!"
Instead of taking the land mine: end the story saying "Bang!"
Check taking the land mine: end the story saying "Bang!"
Carry out taking the land mine: end the story saying "Bang!"
After taking the land mine: end the story saying "Bang!"
Report taking the land mine: end the story saying "Bang!"
```

So which should we use? Of course, we could decide that it really doesn't matter: what works, works. But it is a good idea to play along with the conventions used by Inform, if only because that will make our rules interact better with each other and with rules by other people which we may someday want to borrow. So this chapter ends by offering a few guidelines. Let us suppose that we have some effect which we want to achieve.

1. Are we just trying to correct the player's typing?

For instance, responding to the command "STEAL GOLDEN EAGLE" with a reply like "To steal something, just try to TAKE it." It is bad style to make a special action for this, which does nothing except to print up this text: better is to use the "Understand ... as a mistake" technique, which will come up in the chapter on Understanding.

2. Does the effect apply only to a particular situation, or is it a general phenomenon?

In other words, does the effect apply only to particular people, things or places, or is it a generic rule of play? In the case of the land mine, this is an easy question to answer: it is a unique situation. On the other hand, stopping the player from carrying unduly heavy weights would be a generic rule of play.

Rules like the one saying that photography is only possible if one holds the camera are, by convention, also counted as generic rules of play: they are not really special rules about the camera, but apply to all possible acts of photography anywhere, so are actually generic.

Sometimes we can choose our own answer to this question, and go either way. Suppose we want a certain place to be muddy-floored, affecting things that happen there. One way would be to write exceptional rules applying to that one room. But we could alternatively create a general concept of muddiness ("A room can be clean or muddy. A room is usually clean.") and then regard the new behaviour as being a set of generic rules applying in muddy rooms. We could then, of course, create a second muddy room with much greater ease, or transplant these rules to other works and have muddy rooms in those too.

3. Particular situations: use Instead or After (or sometimes Before).

The next question is: does the effect kick in after the hoped-for action has taken place, or not? In the case of the land mine, to answer that means deciding whether we think the detonator is sensitive to the slightest touch - in which case the explosion would happen at the first touch, and should be in an "Instead" rule - or whether one must actually pick up and disturb the mine - in which case an "After" should be used.

That leaves us a choice of two rulebooks if the effect takes place when the merest impulse towards the action is felt: "Before" and "Instead". Which to use? In cases of doubt, choose "Instead". But if the effect is intended to absolutely suppress all such impulses - for instance, in a silent examination room there must be no talking - then "Before" might be more appropriate. We could imagine that someone about to say something first has a mental impulse to speak, then opens his mouth so that it becomes visible to others that some talking is about to go on, and finally utters words. Here are three possible responses:

> "You cannot contemplate breaking this smothering silence." (*Before*)
> "The invigilator stares you down through her horn-rimmed glasses." (*Instead*)
> "Everyone turns, appalled, as the silence is broken like the surface of a swimming pool by a falling elephant." (*After*)

4. Generic situations: work with action rules (or sometimes Before).

(a) If the effect takes place only when an action is definitely being tried, then we should use one of the action's three rulebooks: check, carry out or report. Check rules should do nothing, and should say nothing either unless they block the action (in which case, they should say why).

Carry out rules must not block the action - it is too late for that - and should not say anything - that hasn't happened yet. (There are a few exceptions: if the action is to look at something, then carrying it out is in a sense the same thing as reporting it. But in all cases of doubt, a carry out rule should say nothing.) Adding a carry out rule to an existing action can make it do something extra.

Report rules must neither block the action nor do anything. Working with new report rules is a way to make more natural-seeming, or more informative, messages appear. For instance, an effect where we want to be able to see through a door when it is first opened ("You open the panelled door, through which is the Board Room.") would be a case for a report rule.

In all cases, it is good style to write check, carry out or report rules in such a way that they could be used in other works too, or in situations that could conceivably have happened in this one (even if in fact it never does). We may one day want to put our new rules into an extension to be used by other people or in other projects, after all.

In this chapter, we have only seen the addition of new rules. We could add a new "check taking" rule, for instance, with the techniques seen so far. But what if the effect we want is not a matter of adding a rule but taking away, or restricting the applicability, of an existing one? In that case, we will need to say that the rule "does nothing" under certain circumstances (see the Rulebooks chapter). The check, carry out and report rules for all of the built-in actions are named, and they appear in the Actions index.

(b) If the effect takes place to divert or supplement an action, before that action actually takes place, then this should be done with a "Before" rule. This is the biggest practical use of "Before" rules: to try other actions, either instead or as well as the one just getting under way.

For example, if we want an automatic mechanism to try opening a container before taking something inside it, that would be a classic case for "Before". Indeed, that is the only way it could work - "Before" rules have a chance to get in before the touchability conditions are tested.

If we wanted a special "stealing" action for the act of taking another person's possessions, we might want to divert any taking action for such items into our new "stealing" action - that too would be a "Before". This would ensure that any "Instead" rules to do with taking do not apply.

5. Changing the behaviour of out-of-world actions.

Remember that "Instead", "Before" and "After" do not apply: so use "check" rules to forbid certain out-of-world activities, or specify that their rules do nothing (see the Rulebooks chapter).

Finally...

These are only guidelines. The system is designed to be flexible in order to give the author the widest possible range of options, and nobody should feel ashamed of making use of them.

Start of Chapter 12: Advanced Actions

Back to §12.20. Stored actions

Onward to Chapter 13: Relations: §13.1. Sentence verbs

Examples from Chapter 12: Advanced Actions

Start of this chapter

Chapter 13: Relations

Indexes of the examples

184 ★ **Example Virtue** RB

Defining certain kinds of behavior as inappropriate, so that other characters will refuse indignantly to do any such thing.

```
"Virtue"

The Cloister is a room. Lady Teresa is a woman in the Cloister. Mother Margaret
is a woman in the Cloister.

Attacking someone is misbehavior. Kissing someone is misbehavior.

Instead of asking someone to try misbehavior: say "[The person asked] stiffens,
offended by your suggestion."

Test me with "kiss margaret / margaret, kiss me".
```

185 ★★★ **Example Latris Theon** RB

A person who can accept instructions to go to new destinations and move towards them according to the most reasonable path.

To begin with, we create an action for going to a named place. All that this action will do is to change that person's hoped-for destination: the actual moving around comes later.

```
"Latris Theon"

A person has a room called destination.

Understand "go to [any room]" as going vaguely.

Going vaguely is an action applying to one visible thing.

Carry out someone going vaguely:
	now the destination of the person asked is the noun.

Report someone going vaguely:
	say "[The person asked] looks amused, but accepts the commission to go to [the noun]."
```

It stands to reason the player plays Zeus or at the very least Apollo, but let's not let this go to the player's head. Note that the following rule applies to the player, but not to anyone else, so HERMES, GO TO ATHENS will work but GO TO ATHENS will not.

```
Carry out going vaguely:
	say "You're too thoroughly lost."
```

And finally we recreate Greece and one of its heroes.

```
Corinth is a room. Athens is east of Corinth. Epidaurus is southeast of Corinth and east of Mycenae. Mycenae is south of Corinth. Olympia is west of Mycenae. Argos is south of Mycenae. Thebes is northwest of Athens. Pylos is south of Olympia. Sparta is east of Pylos and south of Argos. Delphi is northwest of Thebes.

Hermes is a man in Corinth. The destination of Hermes is Corinth. [So he is initially happy where he is.] Persuasion rule for asking Hermes to try going vaguely: persuasion succeeds. [But he is open to suggestions.]

Every turn when the destination of Hermes is not the location of Hermes:
	let the right direction be the best route from the location of Hermes to the destination of Hermes;
	try Hermes going the right direction.

Test me with "hermes, go to athens / e".
```

It simplifies matters that our map of Greece makes it possible to reach any location from any other location, by some sequence of movements: if there were an isolated location -- say, Crete -- with no map connection to the mainland, then we would have to worry about the "right direction" not being a direction at all. The following version of Hermes' trekking rule is protected against the possibility:

```
Every turn when the destination of Hermes is not the location of Hermes:
    let the right direction be the best route from the location of Hermes to the destination of Hermes;
    if the right direction is a direction, try Hermes going the right direction.
```

186

★ Example **The Hypnotist of Blois**

RB

A hypnotist who can make people obedient and then set them free again.

```
"The Hypnotist of Blois"

A person is either hypnotized or alert. A person is usually alert.

Persuasion rule for asking a hypnotized person to try doing something:
    persuasion succeeds.

Understand "hypnotize [someone]" as hypnotizing.

Hypnotizing is an action applying to one thing.

Check hypnotizing:
    if the noun is hypnotized, say "Already done." instead.

Carry out hypnotizing:
    now the noun is hypnotized.

Report hypnotizing:
    say "[The noun] slips under your control."

Instead of waking someone hypnotized:
    now the noun is alert;
    say "[The noun] returns abruptly to consciousness."
```

We will learn more about check rules for other characters shortly, but the following prevents the hypnosis patient from trying to hypnotize us in turn:

```
Check someone hypnotizing someone:
    stop the action.

Maison de la Magie is a room. "In a darkened room, a few hundreds of paces from the chateau of Blois, you give to tourists three shows a day: displaying to them power they do not comprehend and spectacles they do not deserve."

A volunteer is a woman in the Maison. "A volunteer from the audience stands facing you, [if alert]skeptically awaiting hypnosis[otherwise]her face worshipful and obedient[end if]." The printed name of the volunteer is "volunteer from the audience". The description is "A distracted, susceptible woman." The volunteer wears a t-shirt and a baseball hat.

The player wears a top hat and a cape.
```

```
Test me with "volunteer, remove hat / hypnotize volunteer / look / volunteer, remove hat / wake volunteer / volunteer, wear hat".
```

187 ★ **Example** **Police State** *RB*

Several friends who obey you; a policeman who doesn't (but who takes a dim view of certain kinds of antics).

```
"Police State"

Seventh Avenue is a room. "The bars are all closed now, and there aren't any good clubs to go to, so you're on your own for open-air entertainment."

Persuasion rule for asking the policeman to try doing something: persuasion fails.

Persuasion rule for asking someone to try doing something: persuasion succeeds.
```

Note that the policeman will never get to the second persuasion rule, so he will always refuse to do the player's nefarious bidding.

```
Charles, Thomas, and Larry are men in Seventh Avenue. Patricia is a woman in Seventh Avenue.
```

And here's an unnecessary aesthetic touch from a later chapter, which will round up the descriptions of your friends into a single paragraph:

```
Rule for writing a paragraph about someone who is not the policeman:
    let X be the number of visible people who are not the policeman;
    say "It's just [X in words] of you now: [a list of visible people who are not the policeman]. But it sure has been a rip-roaring evening."

The policeman is a man in Seventh Avenue. "A policeman with a very guarded expression is watching you."

Singing is an action applying to nothing. Understand "sing" as singing.

Report singing:
    say "A little the worse for wear, you sing."

Smelling a person is disorderly conduct. Tasting a person is disorderly conduct. Jumping is disorderly conduct. Singing is disorderly conduct.

Instead of someone trying disorderly conduct in the presence of the policeman:
    say "The policeman arrests [the person asked]!";
    now the person asked is nowhere;
    the rule succeeds.
```

```
Instead of disorderly conduct in the presence of the policeman:
	end the story saying "The policeman arrests you!"

Test me with "charles, look / charles, jump / look / policeman, sing / thomas, taste policeman / patricia, sing / look / jump".
```

Notice the difference between the two rules about disorderly conduct: the one for other people says 'the rule succeeds' to make sure that the action is counted as a success and not (as normally happens with instead rules) a failure. Most of the time we don't care whether actions are judged successes or failures, but it matters here, because if we type CHARLES, JUMP and the result fails, then text such as 'Charles is unable to.' will be printed - which would get in the way. So we declare the action a success.

188

Example Generation X

RB

A person who goes along with the player's instructions, but reluctantly, and will get annoyed after too many repetitions of the same kind of unsuccessful command.

```
"Generation X"

The Volunteer Center is a room. "A fairly spartan office, though there are a few attractive posters from advertising campaigns of the past, and an ominous map charting the deforestation of Brazil."

The desk is scenery in the Volunteer Center. "Your standard metal desk, with a drawer for postal supplies and the like." On the desk is a pile of leaflets and a pile of business cards. The power stapler is a device on the desk. A drawer is part of the desk. It is openable and closed. In the drawer are a sponge and a roll of bulk-mail stamps.

Instead of doing something:
	if examining, continue the action;
	if searching, continue the action;
	if looking, continue the action;
	if asking Jenna to try doing something, continue the action;
	say "The whole point of this exercise is to train Jenna. Once you have her on the envelope-stuffing, you can go make some calls."

Instead of answering Jenna that something:
	say "You're going to have to break the instructions down to the simplest ones possible, given that she's in the mood to misunderstand on purpose."

A cardboard box is in the Center. In the cardboard box is a pile of empty envelopes. The box is openable and closed. The description is "A recycling symbol is prominent on the side, which makes you feel a little bit better about using a mailing campaign for this cause."

Jenna is a woman in the Volunteer Center. "Your daughter Jenna is here, barely visible through the cloud of resentment." The description is "Jenna [if jenna
carries something]has [the list of things carried by Jenna][otherwise]returns your stare, with added hostility[end if]."

A persuasion rule for asking Jenna to try doing something: persuasion succeeds.

Unsuccessful attempt by Jenna doing something:
    repeat through Table of Retorts:
        if the reason the action failed is the cause entry:
            say "[response entry][paragraph break]";
            rule succeeds;
    say "'I don't see how I'm supposed to do that, Mom,' Jenna says."

Table of Retorts
```

cause	response
can't take yourself rule	"'Is that like 'get a grip on yourself' or something?' Jenna asks, momentarily diverted."
can't take what's fixed in place rule	"[physical impossibility]"
can't take scenery rule	"[physical impossibility]"
can't take what's already taken rule	"[already done]"
can't drop what's already dropped rule	"[already done]"
can't wear what's already worn rule	"[already done]"
can't take off what's not worn rule	"[already done]"
can't close what's already closed rule	"[already done]"
can't open what's already open rule	"[already done]"
can't switch off what's already off rule	"[already done]"
can't switch on what's already on rule	"[already done]"
can't unlock what's already unlocked rule	"[already done]"
can't lock what's already locked rule	"[already done]"

```
To say physical impossibility:
    say "'Maybe you should've brought someone a little stronger,' Jenna says. 'Like the Incredible Hulk.' "

To say already done:
    repeat through Table of Bored Remarks:
        say "[response entry]";
        blank out the whole row;
        rule succeeds;
    say "'Okay, I'm going to be, like, in the car,' says Jenna. 'Outside.' ";
    end the story saying "Jenna has gotten fed up".

Table of Bored Remarks
response
"'Did that,' says Jenna."
"'Check,' says Jenna."
"'Yeah, Mom, I already did that, okay?'"
"'Look, if I have to be here doing dumb stuff, could you at least tell me to do stuff that isn't already done?' Jenna asks wearily."
"Jenna gives a great upward sigh, riffling her bangs. 'Once again... there is totally no point.'"

Test me with "e / x jenna / jenna, get stapler / jenna, get stapler / x jenna / jenna, drop stapler / jenna, drop stapler / jenna, open box / jenna, open box / jenna,
switch stapler on / jenna, switch stapler on / jenna, take desk / jenna, open box /
jenna, open box".
```

189

★ Example IQ Test

RB

Introducing Ogg, a person who will unlock and open a container when the player tells him to get something inside.

We have already seen how Before... rules can generate implicit actions for the player, so that the player will, for instance, open doors before trying to walk through them. The same can be done for characters other than the player; so for instance:

```
"IQ Test"

The Donut Shop is a room. "Vibrantly decorated in donut colors: pink, brown,
and cream."

Ogg is a man in the Donut Shop. "Ogg is slumped in the corner[if Ogg carries
something] with [a list of things carried by Ogg][end if]. He wears a nametag
which says 'HELLO MY NAME IS OG.'" Understand "og" as Ogg. Ogg wears a
nametag. The description of the nametag is "Sadly misspelled."

The Donut Shop contains a transparent closed openable locked lockable
container called a case. The case contains some cake donuts. The donuts are
edible.

The matching key of the case is a silver key. The silver key is carried by Ogg.

A persuasion rule for asking someone to try doing something:
    persuasion succeeds.

Before someone opening a locked thing (called the sealed chest):
    if the person asked is carrying the matching key of the sealed chest, try the
person asked unlocking the sealed chest with the matching key of the sealed
chest;
    if the sealed chest is locked, stop the action.

Before someone taking something which is in a closed container (called the shut
chest):
    try the person asked opening the shut chest;
    if the shut chest is closed, stop the action.

The block giving rule is not listed in the check giving it to rules.

Test me with "open case / get donuts / og, get donuts / og, give me the donuts /
eat donuts".
```

(A more detailed set of before rules for people using doors and locks is provided in Emily Short's Locksmith extension.)

190 Example Boston Cream

RB

A fuller implementation of Ogg, giving him a motivation of his own and allowing him to react to the situation created by the player.

```
"Boston Cream"

Use scoring.

The Donut Shop is a room. "Vibrantly decorated in donut colors: pink, brown, and cream."

Ogg is a man in the Donut Shop. "Ogg is slumped in the corner[if Ogg carries something] with [a list of things carried by Ogg][end if]. He wears a nametag which says 'HELLO MY NAME IS OG.'" Understand "og" as Ogg. Ogg wears a nametag. The description of the nametag is "Very neatly written."

The Donut Shop contains a transparent closed openable locked lockable enterable container called a case. The case is fixed in place. The case contains some cake donuts, some jelly donuts, and some apple fritters. "The [if unopenable]damaged[otherwise]gleaming[end if] donut case [if something is in the case]contains [a list of things in the case][otherwise]has been stripped of its contents[end if]." The cake donuts, the jelly donuts, and the apple fritters are edible.

The matching key of the case is a silver key. The silver key is in a mesh basket. The mesh basket is closed, transparent, and openable. It is in the Donut Shop.

Before someone taking something which is carried by the player:
    if the person asked cannot touch the player:
        say "Ogg looks with a fixed frown at [the noun].";
        continue the action;
    say "[The person asked] comes up and shakes your arm until you drop [the noun].";
    say line break;
    silently try dropping the noun;
    stop the action.

Before someone unlocking a locked thing with something which is not carried by the person asked:
    try the person asked taking the second noun;
    stop the action.

Before someone opening a locked thing (called the sealed chest):
    if the person asked can see the matching key of the sealed chest:
        if the matching key of the sealed chest is enclosed by the sealed chest:
            say "Seeing the paradoxical location of [the matching key of the sealed chest], [the person asked] gives a howl of rage.";
            increment the score;
            end the story finally saying "You have thwarted Ogg";
        otherwise:
            try the person asked unlocking the sealed chest with the matching key of the sealed chest;
        stop the action.
```

```
Before someone taking something which is in a closed container (called the shut chest):
    try the person asked opening the shut chest;
    stop the action.

Ogg has a number called hunger. The hunger of Ogg is 0.

Every turn:
    increment the hunger of Ogg;
    if the hunger of Ogg is 2 and Ogg is visible, say "Ogg's stomach growls.";
    if the hunger of Ogg is 3 and Ogg is visible:
        if Ogg can see an edible thing (called the target):
            say "Ogg eyes [the target] with obvious intent.";
        otherwise:
            say "Ogg glances at you in a disturbingly shifty way.";
    if the hunger of Ogg is greater than 3:
        if Ogg carries an edible thing (called the target):
            try Ogg eating the target;
        otherwise:
            let new target be a random edible thing which can be seen by Ogg;
            if the new target is a thing:
                try Ogg taking the new target;
            otherwise:
                if Ogg can touch the player, end the story saying "Ogg is gnawing your ankle";
                otherwise try Ogg taking the player.

The crumbs are a thing. "Crumbs of [the list of edible things which cannot be seen by the player] lie scattered over the whole floor."

Instead of asking Ogg to try doing something when Ogg cannot touch the player:
    say "Ogg tilts his head and shrugs, unable to hear your instruction clearly."

Instead of asking Ogg to try eating something:
    say "It's not as though Ogg really needs any encouragement in that department, is it?"

Definition: Ogg is hungry if the hunger of Ogg is greater than 2.

Persuasion rule for asking Ogg to try doing something: if Ogg is hungry, persuasion fails; persuasion succeeds.

Persuasion rule for asking Ogg to try giving something edible to the player:
    persuasion fails.

Unsuccessful attempt by Ogg doing something:
    if the reason the action failed is a failing listed in the Table of Ogg Retorts:
        say "[reply entry][paragraph break]";
    otherwise:
        say "Ogg looks adorably confused."

Table of Ogg Retorts

    failing                             reply
    can't take people's possessions rule "'Ogg too polite.'"
    can't take other people rule        "'Ogg not that greedy.'"
    can't take scenery rule               "'[The noun] very very heavy.'"
    can't take what's fixed in place rule "'[The noun] very heavy."
    can't drop what's not held rule       "'Hunh?'"

Carry out Ogg eating an edible thing:
    move the crumbs to the holder of Ogg;
    now the hunger of Ogg is 0.

Report Ogg unlocking something with something:
    say "Ogg struggles a bit with [the second noun] in the lock of [the noun], but does eventually succeed. 'Hunh!' says Ogg.";
    stop the action.

Carry out Ogg opening the case when the case has been open:
    now the case is unopenable.

Instead of closing the unopenable case:
    say "The glass panels are no longer properly seated in their tracks, and the case cannot be closed ever again."

Report Ogg opening the unopenable case:
    say "Ogg forces [the case] so hard that it does break.";
    stop the action.

Report Ogg opening the case:
    say "Ogg yanks [the noun] open with such force that you fear for its structural integrity.";
    stop the action.

Report Ogg taking something edible:
    say "Ogg acquires [the noun] with a look of tender affection.";
    stop the action.

Report Ogg eating something:
    say "Ogg chows down on [the noun], scattering crumbs in all directions.";
    stop the action.

Report Ogg eating something when the number of visible edible things is 1:
    say "Ogg eats [the noun] in his trademark style. You can no longer bear to watch.";
    stop the action.

After entering the case:
    say "You climb inside the case, folding yourself up uncomfortably."

After locking the case with something when the player is in the case:
    say "You turn the key firmly in the lock -- amazing it locks from within, but it does -- and settle yourself for a long wait, hoping this thing is not air-tight."

The maximum score is 1.

Test me with "open mesh / get key / unlock case / open case / enter case / close case / lock case / wait / wait".
```

191

★ Example **Red Cross**

RB

A DIAGNOSE command which allows the player to check on the health of someone.

```
"Red Cross"

Diagnosing is an action applying to one visible thing. Understand "diagnose [something]" as diagnosing.

Check diagnosing:
    if the noun is not a person, say "Only people can have diagnoses." instead.

Carry out diagnosing:
    say "You're not a medical expert."

Instead of diagnosing the player:
    say "You feel fine!"

The Clinic is a room. Fred is a man in the Clinic. The player carries a tongue-depressor.

Test me with "diagnose me / diagnose fred / diagnose tongue-depressor".
```

192

★★★ Example **Frizz**

RB

Liquid flows within containers and soaks objects that are not waterproof; any contact with a wet object can dampen our gloves.

Suppose we have some items that get wet in contact with other damp things; in particular, if we touch anything wet while wearing a pair of gloves, the gloves too get damp. This requires that we be systematic about detecting all cases where contact occurs. So:

```
"Frizz"

A thing can be waterproof or porous. A container is usually waterproof. An animal is usually waterproof.

A thing can be sodden or dry.

A container can be waterfilled or empty.

Before printing the name of a sodden thing: say "sodden ". Understand the sodden property as describing a thing.

The player wears a pair of woolly gloves. Instead of wearing a sodden thing: say "You dubiously contemplate [the noun], and decide it's best to wear dry clothing."
```

```
Every turn:
    follow the liquid distribution rules.

The liquid distribution rules is a rulebook.

A liquid distribution rule:
    repeat with item running through containers:
        if the item is open and the item is empty and the item is in a waterfilled container:
            now the item is waterfilled;
            if the player can see the item, say "[The item] fills up with water, of course.".

A liquid distribution rule:
    repeat with item running through things in a waterfilled container:
        if the item is porous and the item is dry:
            if the player can see the item, say "[The item] soaks through.";
            now the item is sodden.

A liquid distribution rule:
    repeat with item running through sodden things in the airing cupboard:
        if the item is not The Last Man, now the item is dry.

The last liquid distribution rule:
    if the player carries a dry copy of The Last Man, end the story finally.

Instead of examining a waterfilled container:
    say "[The noun] is full of water[if the noun contains something visible]; it also contains [the list of things in the noun][end if]."

Instead of examining a container:
    say "Dry inside[if the noun contains something visible], and containing [a list of things in the noun]."

A book is a kind of thing. Instead of examining a sodden book, say "[The noun] is too soaked to read, thanks to someone's carelessness." Understand "book" as a book.

The copy of The Last Man is a book. The description of The Last Man is "Mary Shelley's very own original copy, loaned to you under the strictest of agreements[if the pond encloses the copy]. How it came to be in its present position is a long story, and not important at the moment: the critical thing is not to blame oneself - who could have anticipated the cricket bat, anyway? - but to fix it immediately, before anything worse occurs[end if]." The The Last Man is in the ziploc bag. The ziploc bag is waterproof and empty. It is openable, transparent, and closed.

The Back Garden is a room. The Back Garden contains a pond. The pond is a waterfilled container. In the pond is the ziploc bag. A carp is in the pond. The carp is an animal. Instead of taking the carp, say "You're not fast enough."

Rule for writing a paragraph about the pond:
    if the pond encloses the Last Man:
        say "It is a beautiful day just at the end of spring and beginning of summer. The sun shines, the trees blossom, the world conspires in Edenic cheerfulness.
```

```
You can take in none of it. Your eye is on [the Last Man].";
    otherwise:
        say "The pond[if something is in the pond] (containing [the list of things in the pond])[end if] remains a figure of menace in an otherwise lushly verdant landscape."

Before printing the name of the Last Man when the Last Man is enclosed by the Pond: if the Last Man is dry, say "delicate, valuable ".

After printing the name of something (called the target) while writing a paragraph about something:
    if the Last Man is enclosed by the Pond:
        if the target is in something (called the parent), say " - which is in [the parent]".

The Kitchen is west of the Back Garden. "Not in any sense your province: Mrs Peaswell gets agitated if anyone besides herself so much as boils a kettle. In general it is best to sneak through, disturbing as little as possible and preserving the cosy domestic fiction that academics cannot cook.

The nearby stairwell leads up to the second floor."

The Stairwell is above the Kitchen. "Halfway up and halfway down." The airing cupboard is a container in the Stairwell. "An airing cupboard here contains the heating system, and is the ideal place for restoring wet items to a dry state." The airing cupboard is fixed in place.
```

Now: whenever the player definitely touches something, we want to follow certain rules about the transfer of liquid. These rules need to come after an action's check rules (to make sure the action really occurs) and before the carry out rules (so that nothing has moved or changed yet). So we'll borrow from the chapter on rulebooks to create a whole new stage to the action, occurring between the check and the carry out phases:

```
The post-check rules are a rulebook.

This is the post-check stage rule:
    abide by the post-check rules.

The post-check stage rule is listed before the carry out stage rule in the specific action-processing rules.

A post-check rule (this is the dry glove rule):
    if we get wet:
        if the player wears the gloves and the gloves are dry:
            now the gloves are sodden;
            say "(soaking your gloves in the process)";
    continue the action.

A post-check rule (this is the wet glove rule):
    if the player wears the sodden gloves:
        if the The Last Man must be touched and the Last Man is not sodden:
            say "(soaking the parched pages of The Last Man with the rude touch of your sodden gloves)";
            now the The Last Man is sodden;
        continue the action.

Before doing something when the player does not wear the gloves:
    if The Last Man must be touched:
        if The Last Man is dry, say "[The The Last Man] is too precious to endanger when you are not wearing gloves." instead;
        otherwise say "You hesitate instinctively, then recollect that you can hardly harm [The The Last Man] any more than it has already been harmed...".

To decide whether we get wet:
    if the noun is not a thing, no;
    if the noun dampens us, yes;
    if the second noun is not a thing, no;
    if the second noun dampens us, yes;
    no.

To decide whether (item - a thing) dampens us:
    if the item is not liquiferous, no;
    if item must be touched, yes;
    no.

To decide whether (item - a thing) must be touched:
    if the item is the noun and the action requires a touchable noun, yes;
    if the item is the second noun and the action requires a touchable second noun, yes;
    no.

Definition: a thing is liquiferous:
    if it is sodden, yes;
    if it is in a waterfilled container, yes;
    no.

Test me with "x book / x bag / get bag / take off gloves / w / u / put gloves in cupboard / z / open bag / touch book / push book / turn book / get gloves / wear gloves / get book".

Test disaster with "open bag".

Test mishandling with "get bag / open bag / get book".
```

193 ★★★★ **Example 3 AM** RB

A shake command which agitates soda and makes items thump around in boxes.

```
"3 AM"

Understand "shake [something preferably held]" as shaking.

Shaking is an action applying to one carried thing.
```

```
Carry out shaking:
    say "Nothing results of your shaking [the noun]."

Instead of shaking a closed container when something is in the noun:
    say "Something rattles inside [the noun]."

Instead of shaking a closed transparent container when something is in the noun:
    say "Inside [the noun] there are banging noises produced by [the list of things contained by the noun]."

Instead of shaking an open container which contains something:
    say "[The list of things contained by the noun] might fly out."

The Wawa is a room. "A convenience store, if you like to call it that, vending the usual assortment of chips, donuts, soda, and beer. There is something of a line at the sandwich counter."

The box of enrobed cakes is in the Wawa. "A box of Tastykake Enrobed Cakes has fallen off its shelf." The description is "'Enrobed Cakes' is a fancy term for 'strange sponge-like baked good, covered in a thin shell of waxy chocolate'. They are addictive, but not in a way that lets you respect yourself in the morning." The box is a closed openable container. In the box is a cake.

Instead of opening the box, say "The Wawa clerks frown on the consumption of unpurchased foodstuffs."

The can of root beer is a closed openable container carried by the player. The can of root beer is either agitated or calm.
```

Because the can of root beer should have some reactions to having been shaken later in the game, we need to borrow a few ideas from the chapter on Time:

```
Instead of shaking the can of root beer:
    the can calms down in five turns from now;
    say "You give the can a good hard shake.";
    now the can is agitated.

Instead of listening to the can: say "It sounds [if agitated]fizzy[otherwise]calm[end if]!"

At the time when the can calms down:
    now the can is calm.

The sticky mess is fixed in place. "There is a sticky mess on the ground."

Instead of opening the agitated can of root beer:
    now the can of root beer is nowhere;
    now the sticky mess is in the location;
    say "You open the can and fizzing sweet soda goes absolutely everywhere."

Instead of opening the calm can of root beer when the can has been agitated:
    now the can of root beer is nowhere;
```

```
	say "The root beer is disappointingly flat. That's what you get for shaking it up!"

Test me with "get box / shake box / open box / shake box / listen to can / shake can / listen to can / wait / wait / wait / wait / wait / listen to can / open can".
```

194 ★ **Example The Dark Ages Revisited** RB

An electric light kind of device which becomes lit when switched on and dark when switched off.

This will be explored more in subsequent examples, but one of the things we can do with carry out rules is extend the function of existing commands so that they do more, or have special effects in specific situations. For instance, suppose we want to have a class of electric light:

```
"The Dark Ages Revisited"

An electric light is a kind of device. Carry out switching on an electric light: now the noun is lit. Carry out switching off an electric light: now the noun is unlit.
```

This will not affect the behavior of any other devices when switched; it will also not change the way in which switching lights on and off is reported. The player will still see "You switch the sodium lamp on." or the like. In this case that is probably what we want. If we wanted a special way of describing turning on electric lights as opposed to all other devices, we could also add an after rule for the electric light class. Adding this rule to the carry out train does guarantee, though, that in no case will we manage to make the lamp lit without actually making it switched on (or vice versa).

```
The Stooped Corridor is a room. "A low, square-cut corridor, running north to south, stooping you over."

The sodium lamp is an electric light in the Stooped Corridor. "[if switched on]The sodium lamp squats on the ground, burning away.[otherwise]The sodium lamp squats heavily on the ground.[end if]". The description is "It is a heavy-duty archaeologist's lamp, [if switched off]currently off.[otherwise]blazing with brilliant yellow light.[end if]"

Instead of burning the sodium lamp, try switching on the lamp.
```

So far so easy. Since we've built the description of its light or darkness into the lamp's description, though, we may want to get rid of the "...is switched on" line that automatically follows when we look at something. For this we do need to borrow from a later chapter:

```
The examine devices rule is not listed in the carry out examining rules.

Test me with "x lamp / switch lamp on / look / x lamp".
```

195

★★★ Example Paddington

RB

A CUT [something] WITH [something] command which acts differently on different types of objects.

Suppose we intend a game in which the player needs to cut things open on a regular basis. We'll want to check whether he has the proper tools handy, and deal graciously with commands such as CUT [something] when no tool is specified. So:

```
"Paddington"

A blade is a kind of thing.

Understand "cut [something] with [something]" as cutting it with.

Instead of cutting something:
   if a blade (called the edge) is held by the player,
      try cutting the noun with the edge;
   otherwise say "Your fingernails are not sharp enough."

Cutting it with is an action applying to two things.

Check cutting it with:
   if the noun is a person, say "That would hurt." instead;
   if the second noun is not a blade, say "[The second noun] has not got enough
of a blade." instead.

Carry out cutting it with:
   increment the count of rips of the noun.

Report cutting it with:
   say "You slash [the noun] with [the second noun]."
```

We'll need a way to account for all these cuts and rips.

```
Definition: a thing is ripped if the count of rips of it > 0. A thing has a number
called the count of rips. After examining something ripped, say "You see [the
count of rips of the noun in words] rip[s] in [the noun][if something is in the noun],
revealing [a list of things in the noun][end if]."
```

Moreover, because open containers normally list their contents when examined but we'd prefer Paddington's to be mentioned in the ripping paragraph:

```
The examine containers rule does nothing when examining the teddy bear.
```

So far, so good. But suppose that we'd like cutting also to make containers be permanently open and impossible to close again. We could write an "instead" rule, but that would mean that only our instead instructions would take effect, overriding the normal cutting it with rules entirely. Better would be to add a second carry out rule:

```
Carry out cutting a container with something:
    now the noun is open;
    now the noun is unopenable.
```

Now our rule will occur whenever a container is cut, but play will still go on to the reporting stage. And indeed we can add more of these, of varying degrees of specificity:

```
Carry out cutting something which is part of something with something:
    move the noun to the player.

Carry out cutting the quilt with something:
    now the description of the quilt is "Horribly tattered."
```

For that matter, we might want to add a report rule as well, to occur after the "You slash..." rule, so that every time the player cuts something open which has contents, the contents will be listed.

```
Report cutting it with:
    if the noun is open and the noun contains something,
        say "Visible within [is-are a list of things in the noun]."
```

This time we do not add the condition to the rule (i.e., Report cutting an open noun...) If we did, this report rule would be more specific than the general report rule, and would occur first.

```
The Safehouse is a room.

The teddy bear is a closed thing in the Safehouse. The description is "Fluffy[if
the head is part of the bear], with an outsized head[otherwise], but headless[end
if]." The head is a closed part of the teddy bear. In the bear is a large wad of
stuffing. In the head are a small wad of stuffing and a packet of smuggled
diamonds.

The quilt is in the Safehouse. The description is "An old but comforting quilt."

The player carries a blade called a switchblade.
```

Here is a final nicety to get rid of the "which is closed" statement on our closed unopenable teddy bear, using an "activity" rule:

```
After printing the name of a closed unopenable container:
    omit contents in listing.

Test me with "cut quilt with bear / cut quilt with switchblade / examine quilt / cut
bear with switchblade / again / examine bear / cut head with switchblade / get
diamonds / mourn loss of innocence".
```

196 ★★★ **Example Delicious, Delicious Rocks** RB

Adding a "sanity-check" stage to decide whether an action makes any sense, which occurs before any before rules, implicit taking, or check rules.

In some cases, we may want to add new stages to action processing. One possibility is a stage where we check the sanity of what the player is trying to do before executing any of the other commands; so that we avoid, for instance

```
>EAT ROCK
(first taking the rock)
That's plainly inedible.
```

Here is how we might insert such a stage in our action processing, using rulebook manipulation.

```
"Delicious, Delicious Rocks"

Section 1 - Procedure

The sanity-check rules are a rulebook.

This is the sanity-check stage rule:
   abide by the sanity-check rules.

The sanity-check stage rule is listed after the before stage rule in the action-processing rules.

Section 2 - Scenario

Candyland is a room. The lollipop tree is an edible thing in Candyland. The genuine rock is a thing in Candyland.

Sanity-check eating an inedible thing:
   say "Your digestion is so delicate -- you're sure [the noun] wouldn't agree with you." instead.

Test me with "eat lollipop / eat rock".
```

Notice that now Inform does not try taking the rock before rejecting the player's attempt to eat it.

It is of course possible to get the same effect with

```
Before eating an inedible thing:
   say "Your digestion is so delicate -- you're sure [the noun] wouldn't agree with you." instead.
```

...and in a small game with few rules, there's not much reason to add an extra stage. The ability to modify the stages of action processing becomes useful when we have a fairly large game with sophisticated modeling and want to be sure that some kinds of message (such as the sanity-check) are always handled before other things that we might be doing at the before stage (such as generating implicit actions like opening doors before going through them).

197 **Example** **Noisemaking** *RB*

Creating a stage after the report stage of an action, during which other characters may observe and react.

Suppose the current sequence of action handling is not quite enough for us: we'd also like to have a stage after reporting, where other characters can react to the player character's behavior after it has already happened and been reported on screen. Having such a stage is unlike using "after", because after occurs before reports and prevents them from being printed. So, for instance, we could allow the player to do any of a range of different actions that make loud noises, and have a nervous bird that reacts to all of them by flying away afterward.

To do this, we can add a new rule into the specific action-processing rules. (For a list of these, see the Rules index.) Moving rules around and adding new ones requires syntax that we will learn in the chapter on Rulebooks, but the present example is fairly straightforward:

```
"Noisemaking"

Section 1 - Procedure

The other-player response rule is listed after the report stage rule in the specific action-processing rules.

This is the other-player response rule:
    follow the observation rules.

The observation rules is a rulebook.

Section 2 - Scenario

Country Lane is a room. West of Country Lane is Outside the Farmhouse. East of Country Lane is Village Center. North of Country Lane is Open Field.

The player carries a drum.

The black crow is an animal in Country Lane.

Singing is an action applying to nothing. Understand "sing" as singing.

Report singing:
    say "You hum a little ditty."

Singing is a loud action.

Attacking the drum is a loud action.
```

```
The block attacking rule is not listed in any rulebook.

Report attacking something:
    say "THWACK!"

An observation rule for loud action in the presence of the black crow:
    let N be a random adjacent room;
    if N is a room, move the black crow to N;
    say "The crow, startled, flies off to [N]."

Test me with "sing / g / n / hit drum".
```

198 ★ **Example** **Removal** RB

TAKE expanded to give responses such as "You take the book from the shelf." or "You pick up the toy from the ground."

Suppose that we want to change the reporting of "take" so that the player is always told something like "You take the book from the shelf." or "You pick up the toy from the ground." In order to generate these reports, we will need to know where the object started, even though by the time we are printing the output, the object will have moved.

```
"Removal"

The Pharmacy is a room. A desk and a trash can are in the Pharmacy. The pill-counter, a prescription, and a computer are on the desk. The computer is fixed in place. The pill-counter contains some Vicodin. The trash can contains an empty box.

The taking action has an object called previous locale (matched as "from").
```

The previous locale could in theory be either a thing or a room, so we make it "an object" -- that is, the most generic possible kind, to which both things and rooms belong. Now we record what the previous locale is at the beginning of each taking action:

```
Setting action variables for taking:
    now previous locale is the holder of the noun.

Report taking something from the location:
    say "You pick up [the noun] from the ground." instead.

Report taking something:
    say "You take [the noun] from [the previous locale]." instead.

Test me with "get all".
```

199 ★ **Example** **Further Reasons Why All Poets Are Liars** *RB*

The young William Wordsworth, pushing a box about in his room, must struggle to achieve a Romantic point of view.

We begin with the location and its fittings, and we create a kind of value which names the different internal positions we will allow.

```
"Further Reasons Why All Poets Are Liars"

Nook Obscure is a room. "Above the College kitchens, which make a humming sound, less tuneable than bees, but hardly less industrious, with shrill notes of sharp command and scolding intermixed: and below Trinity's loquacious clock, who never lets the quarters, night or day, slip by him unproclaimed, and tells the hours twice over with a male and female voice. In short, the kind of rubbish room they give to a northern villager. But you get a bed and a high shelf all of your own. And you long to find some Romantic way to look out of the window."

The window, the shelf and the bed are scenery in the Nook Obscure. The shelf and the bed are supporters. The bed is enterable.

Internal position is a kind of value. The internal positions are nowhere at all, over by the window, under the shelf and near the bed.

The box is an enterable supporter in Nook Obscure. The current box position is an internal position that varies. The current box position is near the bed. "Your packing case, stamped W. WORDSWORTH (KENDAL), is [current box position]." Instead of taking the box, say "It is filled with your peerless rock collection and too heavy to lift, but could be pushed." Instead of opening the box, say "It is securely nailed shut."
```

We create an action, "pushing it over to", for pushing a box around on the floor of a single location. (Calling this "pushing it over to" prevents clashes with the existing "pushing it to" action, which is for pushing things from one room to another.) Almost half of the text which defines the action is concerned with the two action variables, but they make the implementation of everything else so much easier that we end up writing less than if we hadn't used them.

```
Understand "push [box]" as a mistake ("You can push the box to the window, the bed or the shelf.").

Understand "push [something] to [something]" as pushing it over to. Pushing it over to is an action applying to two things.

The pushing it over to action has an internal position called the old position.
The pushing it over to action has an internal position called the new position.
Setting action variables for pushing something over to something:
    now the old position is the current box position;
    now the new position is nowhere at all;
    if the second noun is the window, now the new position is over by the window;
    if the second noun is the bed, now the new position is near the bed;
    if the second noun is the shelf, now the new position is under the shelf.
```

```
Check pushing it over to:
	if the noun is not the box, say "That's not something you can push." instead;
	if the player is on the bed, say "You can't reach from here." instead;
	if the player is on the noun, say "Not while you are standing on [the noun]." instead;
	if the new position is nowhere at all, say "You can only push [the noun] to the window, the bed or the shelf." instead;
	if the new position is the old position, say "The [noun] is already [new position]." instead.
Carry out pushing it over to:
	now the current box position is the new position.
Report pushing it over to:
	say "With some effort, you shove [the noun] from [old position] to [new position]."
```

Everything which remains simply provides a couple of puzzles to test this arrangement.

```
Euclid's Elements is on the shelf. Understand "euclid" or "book" as the Elements.
Instead of taking something (called the item) which is on the shelf:
	if the player is on the box and the current box position is under the shelf, continue the action;
	say "You cannot reach [the item], which is up on the shelf."

Instead of examining the window:
	say "This window opens rather unpromisingly onto the chapel wall opposite, so even granted the moonlight it is dark in here. Still, surely there's a poem here somewhere?"

Instead of examining the window when the player is on the bed:
	say "Just a blank patch of chapel wall."

Instead of examining the window when the player is on the box:
	if the current box position is near the bed:
		say "Tantalisingly, you are not quite able to spy the statue.";
	otherwise if the current box position is under the shelf:
		say "All you can see is the antechapel wall, and the dull silver gleam of the pealing organ.";
	otherwise:
		say "At last! You can just, standing on tiptoes on the box right up at the window, make out the top of the statue! Of such epiphanies are Poesy born. Let's see now... oh yes...[paragraph break]And from my pillow, looking forth by light[line break]Of moon or favouring stars, I could behold[line break]The antechapel where the statue stood[line break]Of Newton with his prism and silent face,[line break]The marble index of a mind for ever[line break]Voyaging through strange seas of Thought, alone.";
		end the story finally.

Test me with "get on bed / x window / get off / x window /get elements / get on box / x window / get elements / push box to shelf / get off / push box to shelf / get on box / get elements / x window / get off / push box to window / get on box / x window".
```

200 ★ Example The Second Oldest Problem

RB

Adapting the going action so that something special can happen when going from a dark room to another dark room.

Text in this example is drawn from Will Crowther's original 1976 FORTRAN implementation of ADVENTURE, the founding work of the genre, whose source code was rediscovered by Dennis G. Jerz in 2007. Note the capitals: the program ran on an early computer without lower case lettering. They look a little mimsy now, but picture them glowing green on an old-style cathode ray tube monitor in a darkened room late at night.

The problem alluded to is that the player is forbidden to walk between two dark rooms, so that he must always have light to see by from at least one end of any movement. Writing source text to achieve this is tricky to get right in every case, because the determination of light is hard to do. Here we interleave the necessary rules into the existing "going" action, using a new action variable to record the number of ends which are dark as experienced by the player, which might be 0, 1 or 2:

```
"THE SECOND OLDEST PROBLEM"

The going action has a number called the dark terminus count.
Setting action variables for going:
    now the dark terminus count is 0;
    if in darkness, increment the dark terminus count.
The last carry out going rule:
    if in darkness, increment the dark terminus count;
    if the dark terminus count is 2, end the story saying "YOU FELL INTO A PIT AND BROKE EVERY BONE IN YOUR BODY!" instead.
```

And now three early rooms to try this out.

```
COBBLE CRAWL is a room. "YOU ARE CRAWLING OVER COBBLES IN A LOW PASSAGE. THERE IS A DIM LIGHT AT THE EAST END OF THE PASSAGE."

DEBRIS ROOM is west of COBBLE CRAWL. "YOU ARE IN A DEBRIS ROOM, FILLED WITH STUFF WASHED IN FROM THE SURFACE. A LOW WIDE PASSAGE WITH COBBLES BECOMES PLUGGED WITH MUD AND DEBRIS HERE,BUT AN AWKWARD CANYON LEADS UPWARD AND WEST."

AWKWARD CANYON is west of DEBRIS ROOM. "YOU ARE IN AN AWKWARD SLOPING EAST/WEST CANYON."

DEBRIS ROOM and AWKWARD CANYON are dark.

Rule for printing the name of a dark room: say "DARKNESS" instead.
Rule for printing the description of a dark room: say "IT IS NOW PITCH BLACK. IF YOU PROCEED YOU WILL LIKELY FALL INTO A PIT." instead.

Test me with "w / e / w / w".
```

This is only the second oldest problem in the IF literature: the earliest puzzle is unlocking the steel grate which bars entrance to the cave.

---

201 ★★★ **Example** **Puff of Orange Smoke** RB

A system in which every character has a body, which is left behind when the person dies; attempts to do something to the body are redirected to the person while the person is alive.

Suppose we want to let the player kill characters, leaving behind corpses.

```
"Puff of Orange Smoke"

Paraguay is a room. Bolivia is north of Paraguay. Lydia is a woman in Paraguay.
"Lydia is, as usual, here." The description of Lydia is "Long, long legs and a
sarcastic attitude." Instead of touching Lydia: say "'Watch it, sailor,' she snaps."

A body is a kind of thing. A body is a part of every person. Instead of touching a
body: say "[The noun] is grotesquely inert."

The description of Lydia's body is "Long, long legs and no attitude at all." The
initial appearance of Lydia's body is "Lydia's corpse is sprawled at your feet."
```

Using our "part of every person..." line, we've conveniently assigned one body per person. Since we're going to separate people from their bodies when the bodies die, though, we also want a more permanent relation that will help us keep track of which bodies used to belong to which people:

```
Spirit-possession relates one person to one body. The verb to be owner of
means the spirit-possession relation.

When play begins:
    repeat with victim running through people:
        let the corpse be a random body which is part of the victim;
        now the victim is owner of the corpse.
```

When Lydia is alive, we want >TOUCH LYDIA'S BODY to mean the same thing as >TOUCH LYDIA, so we use the setting action variables rules as a convenient point at which to reassign the action:

```
Setting action variables when the noun is a body which is part of a person
(called owner):
    now the noun is the owner.

Setting action variables when the second noun is a body which is part of a
person (called owner):
    now the second noun is the owner.
```

This doesn't change Inform's idea about what action is being performed; just about the object it's being performed on. The rest of the action will now proceed as if the player had typed >TOUCH LYDIA.

Along similar lines, once Lydia is dead, we want >MOVE LYDIA to mean >MOVE LYDIA'S BODY if the body is in view:

```
Setting action variables when the noun is a dead person and the noun is owner
of a visible body (called the mortal remains):
    now the noun is the mortal remains.
```

The trick is, though, that >MOVE LYDIA will only be understood if there is something called Lydia that the player can see and refer to, even after she's dead. There are various ways to do this, but the least painful here will be to make the deceased Lydia permanently visible, by putting her in an always-accessible backdrop. The backdrop itself will never be mentioned in the game, and we should make its name something that the player is unlikely to type casually; we don't want the player to interact with it directly. So:

```
The worldview is a privately-named backdrop. It is everywhere. The spirit-world
is a privately-named transparent closed unopenable container. It is part of the
worldview.

Definition: a person is dead if he is in the spirit-world.
```

It's also possible that the player will type something like >X LYDIA when Lydia's corpse is not in view, so we should have an appropriate answer to that as well:

```
Before doing something to a dead person:
    say "[The noun] is dead; or had you blocked that out?" instead.
```

Because the before rules happen after the setting action variables rules, this will only ever happen if the corpse is not visible.

Now we define the attack itself, which should discard the body, move the spirit to its eternal resting place, and describe the event to the player:

```
Instead of attacking someone:
    let the corpse be a random body which is part of the noun;
    move the corpse to the location;
    move the noun to the spirit-world;
    say "With a single blow, you rid the world of [the noun]."
```

And finally a trick borrowed from the chapter on understanding, so that we can refer to "Lydia's body" while Lydia is alive, but "Lydia's corpse" only after Lydia has died:

```
Understand "corpse" as a body when the item described is not part of a person.

Test me with "x body / x lydia's body / touch lydia's body / x corpse / kill lydia /
look / x lydia's body / x lydia's corpse / x corpse / x lydia / touch lydia / lydia, hello
/ n / x lydia / touch lydia / lydia, hello".
```

202 ★★★ Example Croft

RB

Adding special reporting and handling for objects dropped when the player is on a supporter, and special entering rules for moving from one supporter to another.

Suppose that we have a design in which the player spends lots of time on enterable supporters, and in which we want to report certain actions -- dropping things onto those supporters, or leaping from one to another -- in a new way. We might begin by adding some action variables to help us keep track of the situation:

```
"Croft"

The dropping action has an object called the container dropped into (matched as "into").

The dropping action has an object called the supporter dropped onto (matched as "onto").

Rule for setting action variables for dropping:
    if the actor is in a container (called C), now the container dropped into is C;
    if the actor is on a supporter (called C), now the supporter dropped onto is C.

Report dropping a heavy thing onto a metallic thing:
    say "You drop [the noun], and [the supporter dropped onto] clangs protestingly." instead.

Report someone dropping a heavy thing onto a metallic thing:
    say "[The actor] drops [the noun] onto [the supporter dropped onto], which clangs protestingly." instead.

A thing can be heavy or light. A thing can be metallic or ordinary. A thing is usually ordinary. A thing is usually light.

The Ancient Cambodian Temple is a room. "A vast space built for ancient and forgotten rituals. The stone floor crawls with vermin. Well above the floor, and separated by some feet, are twin platforms built into the wall: the one carved of jointed wood, the other of sheets of graven bronze."

A platform is a kind of supporter. A platform is always enterable. A platform is usually scenery.

The bronze platform is a metallic platform in the Temple. Lara is a woman. She is on the bronze platform. She wears safari pants and a tank top. She carries a gun and a map. The gun is heavy.

The wood platform is an ordinary platform in the Temple. The player is on the wood platform. The player carries a rope, an Ancient Cambodian/English Phrasebook, a pickaxe, and a precious idol. The idol and the pickaxe are heavy.

Persuasion rule: persuasion succeeds.
```

```
The entering action has an object called the place left (matched as "from").
    Check entering a platform from a platform:
        if actor is the player, say "You leap into midair to cross the distance...";
    otherwise say "[The actor] leaps gracefully across the distance...";
    move the actor to the holder of the noun, without printing a room description.
```

Because this rule occurs before the "implicitly pass through other barriers rule", that rule will not occur when we move from platform to platform; we'll use our own custom rule instead.

```
Rule for setting action variables for entering:
    now the place left is the holder of the actor.

Report entering a platform from a platform:
    say "You land in a cat-like crouch on [the noun]." instead.

Report Lara entering a platform from a platform:
    say "Lara lands soundlessly on [the noun][if the noun supports the player]
beside you[end if]." instead.

Report entering a platform from the location:
    say "You jump, catch the edge of [the noun] in your hands, and -- exerting
considerable upper-body strength -- pull yourself up onto it." instead.

Report Lara entering a platform from a location:
    say "Lara jumps, catches the edge of [the noun], and is standing upright on it,
all in less time than it takes to tell."

Instead of examining a person who is not the player:
    say "[The noun] carries [list of things carried by the noun] and wears [list of
things worn by the noun]."

Instead of climbing a platform, try entering the noun.

Test me with "Lara, drop map / lara, drop gun / drop idol / enter bronze platform /
drop pickaxe / get off / climb wood".
```

203

★★ Example **The Man of Steel**

RB

An escaping action which means "go to any room you can reach from here", and is only useful to non-player characters.

It will sometimes be handy to write actions that are only available to the non-player characters and not to the player. To do this, we just define an action which has no "understand": the player will never be able to specify this command himself, but other characters can try it.

This is particularly useful for creating abstract or general actions for when we want a character to eat something, wear something, or go somewhere, but aren't too particular as to what the object is; as here, where we just want Clark to move away from the kryptonite, regardless of direction:

```
"The Man of Steel"

Escaping is an action applying to nothing.

Carry out someone escaping:
    let space be the holder of the person asked;
    let place be a random room which is adjacent to the space;
    let way be the best route from the space to the place;
    try the person asked going way.

Every turn:
    if Clark Kent can see kryptonite, try Clark Kent escaping.

The Daily Planet Newsroom is a room.

Perry White's Office is west of the Newsroom. In Perry White's Office are a desk
and a poster of Elvis. On the desk is a lead-lined box. The box is openable. In
the box is the green kryptonite crystal.

The Supply Closet is east of the Newsroom. The Elevator Bank is north of the
Newsroom.

Clark Kent is a man in the Newsroom. "Clark [if Clark can see the
kryptonite]looks a bit ill[otherwise]is here, frowning as he revises his latest
article[end if]."

Test me with "west / get box / east / close box / east / west / north / south / west".
```

204 ★★★ **Example** **Trying Taking Manhattan**

Replacing the inventory reporting rule with another which does something slightly different.

RB

Inform has built-in commands for other people, and sometimes we may want to adjust the way these work without completely disabling and replacing the command. Suppose, for instance, that instead of

```
Kermit the Frog looks through his possessions.
```

we'd like someone taking inventory to report what he's got, thus:

```
Kermit the Frog says, "I seem to be carrying a microphone and wearing a hat
and a trenchcoat."
```

To do this, we could replace the built-in report rule with a different one.

```
"Trying Taking Manhattan"

The loud inventory rule is listed instead of the report other people taking
inventory rule in the report taking inventory rules.
```

```
This is the loud inventory rule:
    unless the player is the person asked:
        say "[The person asked] says, 'I seem to be carrying [a list of things carried by the person asked][if the person asked is wearing something] and wearing [a list of things worn by the person asked][end if].'"

Persuasion rule for asking someone to try doing something: persuasion succeeds.

Grand Central Station is a room. "Here you are in New York, New York. Any minute now someone is going to burst into song."

Kermit the Frog is a man in Grand Central Station. "Kermit the Frog stands nearby, enjoying being green." Kermit is wearing a hat and a trenchcoat. He is carrying a microphone.

Test me with "inventory / kermit, inventory".
```

205 ★★★★ Example **Under Contract**

RB

Creating a person who accepts most instructions and reacts correctly when a request leads implicitly to inappropriate behavior.

```
"Under Contract"

The Sound Stage is a room. The description is "A somewhat creaky and unconvincing reproduction of the docks of San Francisco."

Clark Gable is a man in the Sound Stage. "Clark is here, carrying [the list of things carried by Clark][if Clark wears anything] and wearing [the list of things worn by Clark][end if]." Clark carries a swagger stick. Clark carries an open openable player's holdall called a briefcase. The briefcase contains a signed contract. Clark wears a fedora and a pair of pants. A handle is part of the briefcase.

The player carries an open openable player's holdall called a frilly bag. The carrying capacity of the player is 2. The player wears a sparkly scarf and a slinky dress. A strap is part of the bag. The tent is a portable enterable container.

Instead of taking off the slinky dress: say "Rowr! Not that kind of movie, babe."

Instead of taking off the pants: say "That's not your job, whatever the society pages might suggest."

Unsuccessful attempt by Clark doing something:
    repeat through table of Clark Retorts:
        if the reason the action failed is the cause entry:
            say "[response entry][paragraph break]";
            rule succeeds;
    say "'I don't think that's in the script,' says Clark dubiously."
```

Table of Clark Retorts

cause	response
can't take yourself rule	"'I'm always self-possessed,' Clark remarks. You've heard that line before, but it sounds so much more convincing from him."
can't take other people rule	"'I don't think it would be appreciated if I tried to do that to [the noun],' he rumbles."
can't take component parts rule	"'I don't want to rip [the noun] out,' Clark remarks."
can't take people's possessions rule	"'I don't cotton to acting like a thief,' Clark replies. 'It ain't proper.'"
can't take what you're inside rule	"'Do you see where I am, babe?' Clark demands."
can't take what's already taken rule	"[already done]"
can't take scenery rule	"'I'm not the stunt man, darling,' he says with a wry twinkle."
can't take what's fixed in place rule	"'I'm not the stunt man, darling,' he says with a wry twinkle."
can't exceed carrying capacity rule	"Clark grins. 'I've only got so many hands, darling,' he says."
can't insert into closed containers rule	"[physical impossibility]"
can't go that way rule	"[physical impossibility]"
can't go through closed doors rule	"[physical impossibility]"
can't enter closed containers rule	"[physical impossibility]"
can't exit closed containers rule	"[physical impossibility]"
can't drop yourself rule	"'We're inseparable, me and me,' Clark replies, with a smile."
can't drop what's already dropped rule	"[already done]"
can't drop what's not held rule	"'Not under my control, [the noun],' replies Clark."
can't drop clothes being worn rule	"[salacious retort]"
can't put something on itself rule	"'I lack the dexterity,' says Clark. Oh, he's so modest."
can't put onto what's not a supporter rule	"'[The second noun] won't support a thing,' says Clark reprovingly."
can't put clothes being worn rule	"[salacious retort]"
can't insert clothes being worn rule	"[salacious retort]"
can't give worn items rule	"[salacious retort]"
can't wear what's not clothing rule	"'Costuming just gets stranger every year,' says Clark. 'In short: no.'"
can't wear what's already worn rule	"[already done]"
can't eat unless edible rule	"'What're you trying to do, poison me?'"
can't eat clothing without removing it first rule	"[salacious retort]"
can't take off what's not worn rule	"[already done]"
can't close what's already closed rule	"[already done]"
can't open what's already open rule	"[already done]"
can't switch off what's already off rule	"[already done]"
can't switch on what's already on rule	"[already done]"
can't unlock what's already unlocked rule	"[already done]"
can't lock what's already locked rule	"[already done]"

```
To say already done:
    repeat through Table of Clark's Bored Remarks:
        say "[response entry]";
        blank out the whole row;
        rule succeeds;
    say "'Already done.' "
```

```
Table of Clark's Bored Remarks
response
"'I have anticipated your every demand.'"
"'We've been through this part of the script already,' Clark remarks, with just a hint of creeping tedium."
"'Right right, already taken care of...'"
"'I'm a believer in rehearsal,' Clark says, sounding more bored than ever, 'but we really have got all this done already.'"

To say salacious retort:
	repeat through Table of Clark's Flirtatious Remarks:
		say "[response entry]";
		blank out the whole row;
		rule succeeds;
	say "Clark, driven past the point of endurance, seizes you in his arms and kisses you.";
	end the story finally.

Table of Clark's Flirtatious Remarks
response
"Clark allows his mustache to quirk at the suggestion."
"Clark wiggles his eyebrows at you."
"'That kind of thing isn't in my contract, sweetie,' says Clark. 'If you're interested you'll have to ask off-camera.'"
"'I'd feel so... bare,' Clark says, with a sidelong look at you."
"'You first,' says Clark."
"Clark's stare is intense. You may be on the verge of breaking him."

To say physical impossibility:
	repeat through Table of Clark's Frustrated Denials:
		say "[response entry]";
		blank out the whole row;
		rule succeeds;
	say "Clark makes a helpless gesture about his ability to move himself, or parts of himself, through solid objects. "

Table of Clark's Frustrated Denials
response
"Clark frowns. 'I can't reach through things -- unless you were planning a special effect?'"
"'Look,' says Clark. 'If you want a stage magician, hire one.'"
"Clark just groans."
"Clark says drily, 'Is this in the script, or are we doing improvisation now?'"

The block giving rule is not listed in the check giving it to rules. The block showing rule is not listed in the check showing it to rules. The report smelling rule is not listed in the report smelling rules. The report listening rule is not listed in the report listening to rules.

Carry out listening to something:
	do nothing.

Report listening to something: say "Your attention bears no interesting result."

Carry out smelling something:
	do nothing.
```

```
Report smelling something: say "Your attention bears no interesting result."

Report someone listening to something: say "[The person asked] concentrates, listening."

Report someone smelling something: say "[The person asked] sniffs at [the noun]."

Instead of asking someone for something: try asking the noun to try giving the second noun to the player.

Carry out showing something to someone: say "You reveal [the noun] to [the second noun]."

Carry out Clark showing something to someone:
    if the second noun is the player:
        say "Clark shows you [the noun]. [run paragraph on]";
        try examining the noun;
    otherwise:
        say "Clark reveals [the noun] to [the second noun]."

Instead of asking someone to try saying yes: try saying yes. Instead of asking someone to try saying no: try saying no. Instead of asking someone to try saying sorry, try saying sorry.
```

And now we make Clark a very amenable type:

```
A persuasion rule for asking people to try doing something: persuasion succeeds.
```

There are quite a few actions for which no automatic other-character behavior is provided at all, because they always end in failure when the player tries them. Currently the response is not very interesting. Let's spice it up a bit:

```
Burning something is useless action. Waking up is useless action. Thinking is useless action. Cutting is useless action. Jumping is useless action. Tying something to something is useless action. Drinking something is useless action. Swinging is useless action. Rubbing is useless action. Setting something to something is useless action. Waving hands is useless action. Buying is useless action. Climbing is useless action. Sleeping is useless action. Kissing is useless action. Throwing something at something is useless action. Attacking is useless action. Asking something about something is useless action. Telling something about something is useless action. Answering something that something is useless action. Waking something is useless action.

A persuasion rule for asking people to try useless action:
    say "Clark Gable frowns. 'I don't mean to be difficult, but I can't see any point in that.'";
    persuasion fails.

A persuasion rule for asking Clark to try taking off the pants:
    say "He gives you a look of mocking exasperation. Apparently you'll have to take a (slightly) more subtle approach.";
    persuasion fails.
```

```
Check someone giving something to someone (this is the can't give worn
items rule):
    if the person asked wears the noun, stop the action.
```

And because we do not want Clark automatically taking off the pants as the result of a drop action:

```
Instead of Clark taking off the pants: do nothing.
```

We know that this can only come about as a default action, because we have arranged matters so that he cannot be persuaded directly.

```
Test me with "give dress to clark / clark, take off pants / clark, eat pants / clark,
give me the pants / clark, drop pants / clark, put the pants in the briefcase / again
/ g / g / g"
```

---

206 ★ **Example** **Get Axe** RB

Changing the check rules to try automatically leaving a container before attempting to take it. (And arranging things so that other people will do likewise.)

We could now re-write the check rules so that any time someone (the player or someone else) tries to pick up a container which he is in, he will first get out:

```
"GET AXE"

This is the clever can't take what you're inside rule:
    if the person asked is in the noun, try the person asked exiting;
    if the person asked is in the noun, rule fails.

The clever can't take what you're inside rule is listed instead of the can't take
what you're inside rule in the check taking rules.

Attic is a room. The unused coffin is in the Attic. The coffin is enterable and
openable and open. Raskolnikov is a man in the coffin.

Persuasion rule for asking Raskolnikov to try doing something:
    persuasion succeeds.

Test me with "raskolnikov, get coffin".
```

---

207 ★★★ **Example** **Barter Barter** RB

Allowing characters other than the player to give objects to one another, accounting for the possibility that some items may not be desired by the intended recipients.

By default, if we make no modifications, telling one player to give something to another will fail, even if persuasion succeeds. This is because the default behavior of the GIVE command is interrupted by the "block giving rule" -- since in many cases we do not want people to exchange objects freely.

However, suppose that we do want characters to be able to exchange articles freely: we allow persuasion to succeed and turn off the "block giving rule".

```
"Barter Barter"

The block giving rule is not listed in the check giving it to rules.

A persuasion rule for asking people to try giving: persuasion succeeds.

The Trading Post is a room.

Meriwether Lewis is a man in the Trading Post. He carries a fluffy handmade quilt and a bag of beans. The beans are edible.

William Clark is a man in the Trading Post. He carries leather slippers, a journal, and a loaf of bread. The bread is edible. The slippers are wearable.

Instead of examining someone:
    say "[The noun] is carrying [the list of things carried by the noun]."
```

And now we might want to implement a way to keep track of whether the recipient character wants what's being offered:

```
Check someone trying giving something to someone (this is the sneering refusal rule):
    if the second noun dislikes the noun, stop the action.

Unsuccessful attempt by someone trying doing something:
    if the reason the action failed is the sneering refusal rule, say "'Would you care for [the noun]?' [the person asked] asks solicitously of [the second noun].

But [the second noun] refuses [the noun] disdainfully.";
    otherwise say "[The person asked] just appears bewildered by this improbable instruction."

Distaste relates one person to various things. The verb to dislike means the distaste relation.

Clark dislikes the beans. Lewis dislikes the bread.
```

Since we've defined this as a relation, we could change what the characters like and dislike during the course of the game, freely; for instance, characters might grow hungry and suddenly like all the edible articles.

```
Test me with "x lewis / x clark / clark, give the slippers to lewis / clark, give the bread to lewis".
```

208

★★★ Example The Man of Steel Excuses Himself

RB

Elaborating the report rules to be more interesting than "Clark goes west."

Report rules can be a good point at which to add local color: while Inform provides default descriptions of character behavior, these are rather generic and can stand to be customized.

For instance, if we wanted to liven up our previous Clark Kent example:

```
Report Clark Kent going a direction (called the way):
    say "[one of]With a particularly weak excuse[or]Muttering[at random] about [random excuse subject], Clark heads [way]." instead.

To say random excuse subject:
    choose a random row in the Table of Lame Excuses; say "[reply entry]".

Table of Lame Excuses
reply
"needing a paper-clip"
"wanting an English-Tuvalu dictionary"
"walking a neighbor's dog"
"hearing air-raid sirens"
"having drunk too much coffee"
"thinking he smells smoke"
"wondering where Lois got to"
"needing to speak to Jimmy"
"noticing the Good Year blimp"
```

It's good to be careful, as the library report rules have been designed and tested to describe every contingency (going through doors, going in vehicles, etc.): so when replacing a report rule, we should try to consider all the possible variations of the action that we might want to describe.

However, in this case, our scenario is so simple that there are no doors, vehicles, or pushable objects, so we're safe in giving Clark a very simple reporting scheme.

209

★★★★ Example Fate Steps In

RB

Fate entity which attempts to make things happen, by hook or by crook, including taking preliminary actions to set the player up a bit.

One of the nice things about before rules for actions is that they allow us to express some planning for characters other than the player: we've already seen how this works, a bit. But we could also use before rules to write plans for an abstract story-driving entity, rather than for other individual characters. This story-driver could be in charge of all the non-player characters, as well as spontaneous or natural changes in the environment, shaping the narrative around the player's behavior.

The following example is a very simple one, but the same concept could be worked out in a great deal more complexity, with all sorts of alternative procedures available to our story-manager:

```
"Fate Steps In"

Fate is a woman. After deciding the scope of the player: place Fate in scope. The description of Fate is "Not smiling." Instead of doing something other than examining to fate: say "As if."

Every turn: try fate tripping.

Tripping is an action applying to nothing.

Carry out someone tripping:
    if something dangerous (called the trap) is in the location:
        say "Lise chooses this moment to lick her fingers -- it's not gross, it's natural, you decide -- stand up, and head for the door. Unfortunately, her path crosses directly over [the trap]. There is a vaudevillesque moment where you try to warn or catch her; the next moment she's on the floor, looking shocked and also in quite a lot of pain. 'I'm not sure,' she says to you steadily but with unfocused gaze, 'but I think I might have broken my tailbone.'";
        end the story saying "Well, she's paying attention to you now".

Before someone tripping when the location does not contain a dangerous thing: try the person asked making a mess instead.

Making a mess is an action applying to nothing.

Carry out someone making a mess:
    let calamitous object be a random visible supporter which supports at least three things;
    if calamitous object is a supporter:
        say "[The calamitous object] tips over, spilling [the list of things on calamitous object] all over the place.";
        move the calamitous object to the location;
        now every thing on the calamitous object is in the location.

Definition: a thing is dangerous if it is not the carton and it is not the table and it is not a person.

Before someone making a mess when a safe supporter (called target) is visible:
    if Lise carries something, try Lise putting a random thing carried by Lise on the target instead.

Instead of someone making a mess when the tray is on the table:
    say "Just at that moment, a large blond man-thing in a red jacket walks more or less through you, and you come into violent contact with the table, knocking [the list of the things on the table] onto the floor.";
    now every thing on the table is in the location;
    now every thing on the tray is in the location instead.
```

```
Definition: a supporter is safe if the number of things on it is less than two.

McQuerry Dining Hall is a room.

The table is scenery in the dining hall. The table is a supporter.

Lise Fitzwallace is a woman in the Dining Hall. "Lise is at the nearest table, not apparently paying any attention to you." The description of Lise is "A capella singer, women's rugby champion, general object of attention from all genders. Unlikely to notice you unless fate smiles broadly." Lise carries a fork, a napkin, an empty glass, and a plate of half-eaten eggplant parmesan.

Report Lise putting something on something:
    say "Lise, still deep in thought, absently puts [the noun] on [the second noun]." instead.

The carrying capacity of the player is 2. The carton of chocolate milk is in the Hall. "There's a carton of milk beside you, which you set down for a moment -- but you do want it."

Instead of taking something when the player carries the tray:
    say "You've got both hands full with this tray."

The player carries the tray. On the tray is some macaroni and some overdone chicken. The macaroni and the chicken are edible. The tray is portable.

Test me with "get milk / put tray on table / get milk".

Test again with "drop tray".
```

210 ★ **Example Spellbreaker** RB

P. David Lebling's classic "Spellbreaker" (1986) includes a room where the game cannot be saved: here is an Inform implementation.

The answer is easy, but there is a trap:

```
Check saving the game when the location is the Vault: say "That spell does not work here." instead.
```

The trap is that "Before saving the game...", which might have been our first guess, does not work: because out of world actions are exempt from Before, Instead and After rules.

"Spellbreaker" pulls this unpleasant, but in context witty, stunt as part of a situation which is engineered to force the player to reason through a weighing-objects puzzle using the perfect strategy rather than by guesswork. The illusion that the situation is fair - not rigged against the player, that is - would collapse if the player could save the game and keep retrying possibilities in the light of knowledge gained from earlier attempts. The moral of this story is that any attempt to use in-world situations to influence out-of-world commands should be *extremely uncommon*.

211

★★★★ Example **A point for never saving the game**

RB

In some of the late 1970s "cave crawl" adventure games, an elaborate scoring system might still leave the player perplexed as to why an apparently perfect play-through resulted in a score which was still one point short of the supposed maximum. Why only 349 out of 350? The answer varied, but sometimes the last point was earned by never saving the game - in other words by playing it right through with nothing to guard against mistakes (except perhaps UNDO for the last command), and in one long session.

Here is one way to score this point with Inform:

```
Check saving the game for the first time: decrement the score.
```

That has the right effect, but it just isn't sneaky enough. Instead let us quietly keep track of how many times the player saves:

```
Check saving the game: increment the number of saves.
When play ends: if the number of saves is 0, increment the score.
```

Sneakier, certainly, but now we could get the bonus even if the game ends earlier on, so finally:

```
When play ends: if the number of saves is 0 and the score is 349, increment the
score.
```

212

★★★ Example **Carnivale**

RB

An alternative to backdrops when we want something to be visible from a distance but only touchable from one room.

Suppose we want to make an object that (unlike a backdrop) is definitely located in one room, but can be seen from far off. We want to allow the player to interact with it from a distance, but only using those actions that require visibility. Other actions should be denied:

```
"Carnivale"

The Fairground is a region. Park Entrance, By the Wheel, and Candy Stand are
in Fairground. Candy Stand is north of By the Wheel. Park Entrance is west of
Candy Stand and northwest of By the Wheel.
```

```
The ferris wheel is scenery in By the Wheel. "It is extravagantly tall and carries several dozen glass gondolas for riders."

The description of By the Wheel is "You stand at the foot of an enormous ferris wheel, which turns far too quickly and never seems to stop for new riders."

The description of Park Entrance is "You are now just inside the gates. Behind you snakes a triple line of fairgoers all the way down the length of the valley to the railway station. Roughly southeast of here is the ferris wheel, towering over the other attractions."

The description of Candy Stand is "A hut in pale pink and baby blue dispenses marshmallow death's-heads, sugar-beetles, and other such treats. The giant ferris wheel is just off to the south from here."
```

As the descriptions make clear, the ferris wheel should be visible from everywhere in the fair, so we'll borrow a line from the Activities chapter to make that happen:

```
After deciding the scope of the player:
    if the location is in Fairground, place the ferris wheel in scope.
```

"Scope" determines what the player can interact with; by writing this rule, we make Inform understand all commands that refer to the ferris wheel when the player is anywhere in the fairground, instead of responding with

```
You can't see any such thing.
```

as it normally would.

Now, by default, if the player were to type TOUCH FERRIS WHEEL while in another room, he would get the response

```
You can't reach into By the Wheel.
```

This may not be quite what we want, but we can replace this text with our own reaching inside rule:

```
Rule for reaching inside a room:
    say "You can only look from this distance.";
    deny access.
```

And because our accessibility rules are considered before the "Instead" phase, we can write the following rule confident that it will apply only when the player is in fact in range to touch the ferris wheel:

```
Instead of touching the ferris wheel:
    say "You don't dare: it's spinning too fast."

Test me with "x ferris wheel / touch ferris wheel / se / x ferris wheel / touch ferris wheel".
```

213 ★★★ Example Eddystone

RB

Creating new commands involving the standard compass directions.

Using the compass directions in commands is a little bit finicky because directions are forbidden to figure in any interactions involving touch. (Really, directions are more a concept than an object; this is a compromise situation.) In any case, if we want to write a new command involving these, we need to be sure to specify that the direction is a visible thing. For instance:

```
"Eddystone"

The Lighthouse is a room. "A lonely place, but in these tense times, no one but the lighthouse keeper and a few trusted agents are allowed on the grounds at all, for fear of sabotage."

The light is a fixed in place thing in the Lighthouse. "At the center of the room is the light itself, a 1000-Watt tungsten halogen light powered by diesel generator, and having a visible range of twenty-six nautical miles." Understand "lamp" as the light. It is lit. The light has a a direction called heading. The heading of the light is north.

A room is usually dark.

Understand "turn [something] [a direction]" as reorienting it to. Reorienting it to is an action applying to two things.
```

will give us

```
>turn light northeast
You must name something more substantial.
```

To avoid this mystifying result:

```
"Eddystone"

The Lighthouse is a room. "A lonely place, but in these tense times, no one but the lighthouse keeper and a few trusted agents are allowed on the grounds at all, for fear of sabotage."

The light is a fixed in place thing in the Lighthouse. "At the center of the room is the light itself, a 1000-Watt tungsten halogen light powered by diesel generator, and having a visible range of twenty-six nautical miles." Understand "lamp" as the light. It is lit. The light has a direction called heading. The heading of the light is north.

A room is usually dark.

Understand "turn [something] [a direction]" as reorienting it to. Reorienting it to is an action applying to one thing and one visible thing.

Instead of turning the light, say "Try turning the light to the direction of your choice."
```

```
Check reorienting it to: if the noun is not the light, say "You couldn't do so meaningfully." instead; if the second noun is up or the second noun is down, say "The light only points in compass directions." instead.
```

And now that's done, we have a little fun calculating where the beam hits:

```
Carry out reorienting it to:
    now the heading of the light is the second noun;
    let way be the heading of the light;
    let place be the room way from the Lighthouse;
    while place is a room and place is lower than Lighthouse:
        let place be the room way from the place;
    if place is not a room, now the beam is nowhere;
    otherwise move beam to the place.

Report reorienting it to: say "The light now points [heading of the light][if the beam is in a room], spotlighting [the holder of the beam][otherwise], into empty space[end if]."

The beam is a lit thing. Understand "light" or "brilliant" as the beam. "Brilliant light from the lighthouse floods the whole area." It is fixed in place. Instead of doing something other than examining to the beam: say "The light is, of course, intangible." The description is "The light is coming from the lighthouse, since the lamp is apparently pointed this way."

Altitude is a kind of value. 200 ft specifies an altitude. A room has an altitude. The altitude of a room is usually 50 ft. The altitude of the Lighthouse is 100 ft.

Definition: a room is low if its altitude is 20 ft or less.

The Jetty is south of the Lighthouse. "During daylight hours, a fine place to catch almost unlimited supplies of crayfish. Less entertaining by night."

North of the Lighthouse is the Uphill Road. The altitude of Uphill Road is 75 ft. North of Uphill Road is Hilltop. The altitude of Hilltop is 110 ft. The description of Hilltop is "The highest natural point around for miles; sometimes you will sit up here and watch for the lighthouse supply ship, the Lady Loch."

Northeast of the Lighthouse is Open Field. East of the Lighthouse is Stanley Creek Valley. The description of Stanley Creek is "This place used to have some other name meaning Ghost Valley in the aboriginal language, but it was piously renamed by missionaries." Train Trestle is east of the Stanley Creek Valley. "The now-abandoned track of the Bush Pacific Railway runs here, above Stanley Creek." The altitude of Train Trestle is 100 ft.

Before going from a room (called source) to a room (called destination):
    if source is lower than destination:
        say "It's an uphill climb...";
    otherwise:
        if destination is lower than source, say "You're heading downhill now...";
        otherwise say "It's a straight shot."

Test me with "turn lamp / turn lamp down / turn lamp east / e / e".
```

All very loosely based on the Eddystone Point lighthouse of Tasmania, built in 1889, and forbiddingly remote even today. George Isaacs, a child growing up in the lighthouse, remembers the plentiful crayfish.

214

★★★ Example **Slogar's Revenge**

RB

Creating an amulet of tumblers that can be used to lock and unlock things even when it is worn, overriding the usual requirement that keys be carried.

Under most circumstances, locking and unlocking require the player to be carrying the key he uses to unlock something. This makes sense -- unless the key is on a keychain, or on a chain around his neck, for instance. So here we explore one way to circumstantially override the carrying requirements, while still making sure that the player cannot unlock the door if the unlocking tool is nowhere in sight.

In essence, we are rewriting the carrying requirements rule with a different one of our own devising, and swapping it in only at those moments when it is correct to do so.

```
"Slogar's Revenge"

Section 1 - Procedure

The amulet carrying rule substitutes for the carrying requirements rule when locking something with the Amulet of Tumblers.

The amulet carrying rule substitutes for the carrying requirements rule when unlocking something with the Amulet of Tumblers.
```

We can now replace the usual behavior of the carrying requirements rule (to check whether the player is carrying something and, if not, to generate an implicit take) with a similar rule of our own; note that "if the player has the second noun" is a more compact way to write "if the player carries the second noun or the player wears the second noun":

```
This is the amulet carrying rule:
    if the player has the second noun:
        continue the action;
    say "(first picking up the amulet)[command clarification break]";
    try silently taking the second noun;
    if the player is not carrying the second noun:
        stop the action;

Section 2 - Scenario

The Daunting Dungeon is a room.

West of the Daunting Dungeon is the Disturbing Door. The Disturbing Door is a door. West of the Disturbing Door is the Fallow Field.
```

```
The Disturbing Door is closed and locked.

The player wears the Amulet of Tumblers. The Amulet of Tumblers unlocks the Disturbing Door.

Test me with "unlock disturbing door with amulet / open door / west / remove amulet / close door / lock disturbing door with amulet / drop amulet / unlock disturbing door with amulet".
```

For a more systematic handling of the keychain problem (and a number of other refinements to the behavior of doors), see the Locksmith extension included with Inform.

---

215 ★ **Example Magneto's Revenge** *RB*

Kitty Pryde of the X-Men is able to reach through solid objects, so we might implement her with special powers that the player does not have...

```
"Magneto's Revenge"

The School for Gifted Youngsters is a room. Kitty Pryde is a woman in the School for Gifted Youngsters. "Kitty Pryde waits for you to say why you summoned her out of class." The description is "You see nothing special about Kitty Pryde. But that is only because she has learned to conceal her mutant powers: in fact she has the alternative name Shadowcat, and the ability to phase through solid objects."

The glass box is a container in the School for Gifted Youngsters. It is closed, openable, and transparent. In the box is a quantity of poisonous gas. In the box is a thing called the message from Magneto.

Instead of examining the message when the player is not carrying the message:
    say "You'd need to be holding it before you could read it."

Instead of opening the glass box: say "You're deterred by the swirling bottle-green mist within."

Check someone taking the gas (this is the gaseous object rule): rule fails.

Unsuccessful attempt by someone taking the gas: say "The gas isn't something one can pick up in one's bare hands."

Every turn:
    if the player can touch the gas:
        say "The gas has reached your lungs!";
        end the story.

A rule for reaching inside something:
    if the person reaching is Kitty, allow access.
```

```
Persuasion rule for asking someone to try doing something: persuasion succeeds.

Test me with "get message / kitty, get message / open box / kitty, open box".
```

Now the player can get Kitty to take the message without releasing the poisonous gas:

```
Kitty Pride waits for you to tell her why you summoned her out of class.

You can also see a glass box (closed) (in which are some poisonous gas and a message from Magneto) here.

>kitty, get message
Kitty Pryde picks up a message from Magneto.
```

216

★ Example **Waterworld**

RB

A backdrop which the player can examine, but cannot interact with in any other way.

It's tempting to handle the player's inability to interact with something with a simple instead rule:

```
"Waterworld 1"

A view is a kind of backdrop. Instead of doing something other than examining to a view, say "You are too far from [the noun] to do anything but look."

The sun is a view. It is everywhere. The description is "A blazing sun makes you wish you had never been born."

The Sahara is a room. North of the Sahara is More Sahara. North of More Sahara is Yet Further Sahara.

Test me with "x sun / get sun / n / x sun / n / x sun".
```

Unfortunately, the rule does not address the case where the object in question is the second noun; so for instance the following example reveals the difficulty:

```
"Waterworld 2"

A view is a kind of backdrop. Instead of doing something other than examining to a view, say "You are too far from [the noun] to do anything but look."

The player carries a rope.

The sun is a view. It is everywhere. The description is "A blazing sun makes you wish you had never been born."
The Sahara is a room. North of the Sahara is More Sahara. North of More Sahara is Yet Further Sahara.

Test me with "x sun / get sun / n / x sun / n / x sun / tie rope to the sun".
```

...where the response here behaves as though the sun is in reach. If we had a fully implemented tying action, the player would (even more disastrously) be allowed to lasso celestial objects.

We could add a second instead rule as well:

```
"Waterworld 3"

A view is a kind of backdrop.

Instead of doing something other than examining when the noun is a view:
    say "You are too far from [the noun] to do anything but look."
Instead of doing something other than examining when the second noun is a view:
    say "You are too far from [the second noun] to do anything but look."

The player carries a rope.

The sun is a view. It is everywhere. The description is "A blazing sun makes you wish you had never been born."

The Sahara is a room. North of the Sahara is More Sahara. North of More Sahara is Yet Further Sahara.

Test me with "x sun / get sun / n / x sun / n / x sun / tie rope to sun".
```

This produces acceptable output again, but there is a more elegant way, one that works better with Inform's existing world model. Currently the default model assumes that accessibility -- whether the player can reach something or not -- is checked between the Before... rules and the Instead... rules. We can add our own accessibility rules, including this one to govern whether views are accessible. So for instance:

```
"Waterworld 4"

A view is a kind of backdrop.

The can't touch views rule is listed before the access through barriers rule in the accessibility rulebook.

Accessibility rule (this is the can't touch views rule):
    if the action requires a touchable noun and the noun is a view:
        say "You are too far from [the noun] to do anything but look." instead;
    if the action requires a touchable second noun and the second noun is a view:
        say "You are too far from [the second noun] to do anything but look." instead;

The player carries a rope.
The sun is a view. It is everywhere. The description is "A blazing sun makes you wish you had never been born."

The Sahara is a room. North of the Sahara is More Sahara. North of More Sahara is Yet Further Sahara.

Test me with "x sun / get sun / n / x sun / n / x sun / tie rope to sun".
```

Now our new accessibility rule fits into its proper stage.

217 ★★ Example **Dinner is Served** RB

A window between two locations. When the window is open, the player can reach through into the other location; when it isn't, access is barred.

```
"Dinner is Served"

Street in Kolonaki is a room. "There is a single round table out on the street here, and a window more or less at knee level looks down into the Olive Tree Gyro Shop, which is partly basement."

The Street contains a round table. The table is scenery. On the round table is a plate. On the plate are a gyro and a mound of fresh potates. The plate is portable. The potates and the gyro are edible. The description of potates is "They'd be called french fries, at home, but these are steak-cut and fried in olive oil." The description of the gyro is "Dripping garlic-yogurt sauce."

Olive Tree Gyro Shop is inside from Street in Kolonaki. Kostis is a man in the Gyro Shop. In the Shop is a stand. On the stand is a rotating column of cooking lamb flesh. In the shop is a closed, openable container called a drinks refrigerator. The refrigerator contains a can of Mythos beer and a can of Coke Light.
```

Here's the part that allows reaching through the window.

We replace the usual rule that says the player can never reach into a room with one that more specifically checks whether we are trying to reach through the window. If we aren't, we return the usual refusal. If we are, we return a custom refusal if the window is closed ("You can't reach through the closed window"), but allow access if the window is open.

```
The can't reach through closed window rule is listed instead of the can't reach inside rooms rule in the reaching inside rules.

This is the can't reach through closed window rule:
    let reaching through the window be false;
    if the container in question is a room and the container in question is not the location:
        if the container in question is the Street and the location is the Olive Tree Gyro Shop:
            now reaching through the window is true;
        if the container in question is the Gyro Shop and the location is the Street:
            now reaching through the window is true;
        if reaching through the window is true:
            if the window is closed:
                say "You can't reach through the closed window.";
                deny access;
            otherwise:
                allow access;
        otherwise:
            say "You can't reach into [the container in question] from here.";
            deny access.
```

And the rest is window-dressing.

```
After looking when a room (called the next room) is adjacent:
    try examining the next room.

Instead of examining a supporter, say "On [the noun] [is-are a list of things on
the noun]." Instead of examining an open container, say "In [the noun] [is-are a
list of things in the noun]."

The window is a backdrop. It is in the Street and the Shop. The window can be
openable. The window can be open. The window is openable and closed.
Instead of searching the window in the Street: try examining the shop. Instead of
searching the window in the Shop: try examining the street.

Understand "examine [any adjacent room]" as examining.

Instead of examining a room:
    say "Over in [the noun], you can see [a list of visible things in the noun]."

After deciding the scope of the player:
    if the player is in the Street, place the Shop in scope;
    if the player is in the Shop, place the Street in scope.

Test me with "examine shop / open refrigerator / open window / examine shop /
open refrigerator / get beer / in / examine street / out / get gyro / close window /
put gyro in refrigerator / open window / put gyro in refrigerator".
```

218 ★ **Example Flashlight** RB

Visibility set so that looking under objects produces no result unless the player has a light source to shine there (regardless of the light level of the room).

```
"Flashlight"

The Schoolhouse is a room. "Though normally comfortable, the room is dark and
menacing during the storm; rain sheets on the windows, and you can barely see
the flash of the lighthouse only a few miles away."
```

```
The cabinet is a fixed in place openable container in the Schoolhouse. The
hurricane lantern is a thing in the Schoolhouse. "A hurricane lantern hangs from
a peg." The lantern is lit.

Visibility rule when looking under something:
    if the player is carrying a lit thing (called lamp):
        say "You shine [the lamp] under [the noun]...";
        there is sufficient light;
    there is insufficient light.

There is a marble. The marble can be found or lost. The marble is lost.

Instead of looking under the cabinet when the marble is lost:
    move the marble to the player;
    now the marble is found;
    say "Billy's lost marble! So that's where it got to!"

Test me with "look under cabinet / get lantern / look under cabinet".
```

Because visibility is checked before instead rules, this discovery will (correctly) occur only when the player does have enough light.

219

★ Example **Bosch**

Creating a list of actions that will earn the player points, and using this both to change the score and to give FULL SCORE reports.

RB

We could, if we wanted, make a table of stored actions all of which represent things that will earn points for the player. For instance:

```
"Bosch"

Use scoring.

The Garden of Excess is a room. The gilded lily is an edible thing in the Garden
of Excess.

The Pathway to Desire is west of the Garden of Excess. The emerald leaf is in
the Pathway.

Table of Valuable Actions
```

relevant action	point value	turn stamp
taking the emerald leaf	15	-1
eating the gilded lily	5	-1

(And our list would presumably continue from there, in the full game.)

```
The maximum score is 25.
```

```
After doing something:
    repeat through Table of Valuable Actions:
        if the current action is the relevant action entry and turn stamp entry is less than 0:
            now the turn stamp entry is the turn count;
            increase the score by the point value entry;
    continue the action.

Understand "full score" or "full" as requesting the complete score. Requesting the complete score is an action out of world.

Check requesting the complete score:
    if the score is 0, say "You have not yet achieved anything of note." instead.

Carry out requesting the complete score:
    say "So far you have received points for the following: [line break]";
    sort the Table of Valuable Actions in turn stamp order;
    repeat through the Table of Valuable Actions:
        if the turn stamp entry is greater than 0:
            say "[line break] [relevant action entry]: [point value entry] points";
    say line break.

Test me with "eat lily / w / full score / get leaf / full".
```

This system is tidy, but limited: we cannot give actions interesting names in the score list, like "seducing the pirate's daughter" or "collecting a valuable artifact". So it will not be ideal in all situations, but it has the virtue of being easy to extend, and of listing all of the player's successes in the order in which they occurred in his play-through.

---

220 ★ **Example** **Cactus Will Outlive Us All** *RB*

For every character besides the player, there is an action that will cause that character to wither right up and die.

```
"Cactus Will Outlive Us All"

Death Valley is a room. Luckless Luke and Dead-Eye Pete are men in the Valley. A cactus is in the Valley. Persuasion rule: persuasion succeeds.

A person has an action called death knell. The death knell of Luckless Luke is pulling the cactus. The death knell of Dead-Eye Pete is Luke trying dropping the cactus.

Before an actor doing something:
    repeat with the victim running through people in the location:
        let the DK be the death knell of the victim;
        if the DK is not waiting and the current action is the DK:
            say "It looks as if [the DK] was the death knell for [the victim], who looks startled, then nonexistent.";
            now the victim is nowhere.
```

If we leave it at that, then pulling the cactus will kill Luckless Luke but then say "Nothing obvious happens.", which seems like a bit of an anti-climax. So we add a special case response for that one:

```
After pulling the cactus when Luckless Luke was in the location:
    say "That's a real shame."

Test me with "get cactus / drop cactus / luke, get cactus / luke, drop cactus / pull cactus / look".
```

221 ★★★ Example **Actor's Studio**

A video camera that records actions performed in its presence, and plays them back with time-stamps.

RB

Here we construct a video camera to track and play back actions:

```
"The Actor's Studio"

Section 1 - The Video Camera

The video camera is a thing carried by the player.

Table of Videotape

    recorded action     time stamp
    waiting             9:00 AM
    with 25 blank rows.

Mode is a kind of value. The modes are idle, recording, and playing back. The video camera has a mode. The video camera is idle.

Understand "play back" as playing back. Instead of switching on the camera, try tuning the camera to recording. Instead of switching off the camera, try tuning the camera to idle.

The description of the video camera is "It is currently [mode]; its available settings are idle, recording, and playing back."

Understand "set [camera] to [a mode]" as tuning it to. Tuning it to is an action applying to one thing and one mode.

Instead of setting the camera to something:
    say "The available settings are idle, recording, and playing back."

Check tuning it to:
    if the noun is not the camera, say "Only the video camera can be set to [the mode understood]." instead.

Carry out tuning it to:
    now the mode of the noun is the mode understood.
```

```
Report tuning it to:
    say "You set [the noun] to [mode understood]."

After an actor doing something when the video camera is recording:
    if the current action is tuning the video camera to recording, make no decision;
    if the number of blank rows in the Table of Videotape is greater than zero:
        choose a blank row in the Table of Videotape;
        now the recorded action entry is the current action;
        now the time stamp entry is the time of day;
    otherwise:
        now the video camera is idle;
        say "The video camera runs out of recording memory and switches off.";
    continue the action.

Every turn when the video camera is playing back:
    say "On the camera screen, you see [run paragraph on]";
    let starting playback be false;
    repeat through the Table of Videotape:
        if the recorded action entry is not waiting:
            now starting playback is true;
            say "[line break] -- [if the actor part of the recorded action entry is the
player]you [end if][the recorded action entry], time stamped at [time stamp
entry]";
            blank out the whole row;
    if starting playback is false, say "only static.";
    otherwise say paragraph break.

Section 2 - The Scenario

The Actor's Studio is a room. Lucas is a man in the Actor's Studio. Persuasion
rule: persuasion succeeds.

The Studio contains an edible thing called a croissant.

Test me with "set camera to recording / x lucas / lucas, take inventory / lucas, eat
croissant / set camera to playing back / z".
```

Notice that both Lucas' implied taking action (picking up the croissant) and his eating action are recorded on the same move.

222 ★★★ Example **Anteaters** RB

The player carries a gizmo that is able to record actions performed by the player, then force him to repeat them when the gizmo is dropped. This includes storing actions that apply to topics, as in "look up anteater colonies in the guide".

```
"Anteaters"

A book is a kind of thing. Understand "book" as a book. A book has a table name
called the contents.
```

```
Report consulting a book about:
    say "You flip through [the noun], but find no reference to [the topic understood]." instead.

Instead of consulting a book about a topic listed in the contents of the noun:
    say "[reply entry][paragraph break]".

The Guide to Desert Fauna is a book. The contents of the Guide is the Table of Critters.

Table of Critters
```

topic	reply
"spines"	"You flip through the Guide for a while and eventually realise that spines are flora, not fauna."
"anteater colonies"	"The giant anteater, which grows to six feet in size and can kill a jaguar, is a solitary animal, found in many habitats, including grasslands, deciduous forests and rainforests. It does not form colonies. That's ants. They're actually quite easy to tell apart."

```
Death Valley is a room. The Guide is in the Valley.

The gizmo is in Death Valley. The gizmo has an action called idea. The description of the gizmo is "The gizmo is hard to describe, but it projects an idea of [idea]."

Before when the player carries the gizmo and the idea of the gizmo is waiting:
    say "[The gizmo] eagerly soaks up the whole idea of [the current action].";
    now the idea of the gizmo is the current action.

After dropping the gizmo:
    say "The percussion of the fall seems to have shaken the gizmo's idea loose! There's nothing for it now but [idea of the gizmo].";
    try the idea of the gizmo;
    now the idea of the gizmo is waiting.

Test me with "get guide / look up spines in guide / x gizmo / get gizmo / i / x gizmo / drop gizmo / get gizmo / look up anteater colonies in guide / x gizmo / drop gizmo".
```

Chapter 13: Relations

Contents of *Writing with Inform*
Chapter 12: Advanced Actions
Chapter 14: Adaptive Text and Responses
Indexes of the examples

§13.1. Sentence verbs

Descriptions of things - "open door", "people in the Drawing Room" - have already had a whole chapter to themselves. But descriptions are only half of the story of Inform's highly flexible language for talking about places, things and circumstances: this chapter is the other half, and is about the "sentence". Of course all text is made up of sentences, but Inform has a more specific meaning than that. Consider the following pieces of source text:

```
The mouse is in the teapot.
```

```
Every turn when the mouse is in the teapot, say "A tail hangs out of the spout."
```

```
Instead of taking the mouse:
    say "The mouse slips from your hand and disappears into the teapot!";
    now the mouse is in the teapot.
```

What these three extracts have in common is the sentence "the mouse is in the teapot". Such a sentence can be used in three different ways: to declare the original state of the world, to ask during play if the world currently has that state, or to change things during play so that it does.

Actually, though, only definite sentences about the present can be used in all three ways. A vague instruction like

```
now Mr Darcy can see the mouse;
```

will fail, because there are so many ways in which Darcy might be able to see the mouse that Inform has no way to know how to arrange matters. And this by contrast is not merely difficult but impossible:

> now Mr Darcy has never seen the mouse;

Which cannot be arranged because the past cannot be changed.

Verbs also turn up inside the more complicated descriptions. For instance,

> things which are in the teapot
> people who can see the mouse

are both descriptions, not sentences, but they contain "to be" and "to be able to see" respectively.

This chapter is about the verbs which can be used in sentences and descriptions. Inform involves many other features which use verbs - the action "taking the mouse" and the phrase "end the story" both use forms of verbs (to take and to end) - but this chapter has nothing to do with them: so for the sake of clarity, we will call verbs that occur in sentences "sentence verbs".

⬆ Start of Chapter 13: Relations
⬅ Back to Chapter 12: Advanced Actions: §12.21. Guidelines on how to write rules about actions
➡ Onward to §13.2. What sentences are made up from

§13.2. What sentences are made up from

A sentence consists of two nouns with a verb between them. Usually, the two nouns are descriptions, as in:

> Mr Collins is in a lighted room.

Here "Mr Collins" and "a lighted room" are descriptions. But there are sentences where one or both of the nouns is a value of some other kind. For instance, in

> if the score is greater than 10, ...

the sentence "the score is greater than 10" consists of two number values ("the score" and "10") connected by a verb part ("is greater than").

This chapter is about getting the most out of sentences by defining new verbs to express ideas not already built in to Inform. Before we can define a new sentence verb, however, we must first look at the meanings of verbs: which Inform calls "relations".

Start of Chapter 13: Relations

Back to §13.1. Sentence verbs

Onward to §13.3. What are relations?

Example 223: ★★★ **Formal syntax of sentences** A more formal description of the sentence grammar used by Inform for both assertions and conditions.

§13.3. What are relations?

Relations are what sentences express. They are yes/no questions about pairs of things: for example, to say that the coin is in the purse is to say that a particular relation ("being in") is true about a specific pair of things (the coin, the purse). It is neither a fact about the coin nor about the purse, but about the two together.

Inform comes with a number of relations built in, almost all of which have been used in previous chapters already. The following table names some of the more useful ones, giving examples of sentences to bring them about:

```
containment relation - The coin is in the purse.
support relation - The coin is on the table.
incorporation relation - The coin is part of the sculpture.
carrying relation - The coin is carried by Peter.
wearing relation - The jacket is worn by Peter.
possession relation - if Mr Darcy has a rapier...
adjacency relation - The Study is east of the Hallway.
visibility relation - if Darcy can see Elizabeth...
touchability relation - if Darcy can touch Elizabeth...
```

These relation names do not trip off the tongue, but they relatively seldom need to be referred to.

The same meaning can often be expressed by using several different verbs, or using the same verb in several different ways, as in the following examples:

```
The coin is in the purse.
The purse contains the coin.
The coin is contained by the purse.
```

all of which boil down to saying that the coin and purse satisfy the containment relation. Because of that, *relations are not the same as verbs*. To create a new idea, we will need first to create a new relation, and only then can we set up a verb which allows us to talk about that relation.

Start of Chapter 13: Relations

Back to §13.2. What sentences are made up from

Onward to §13.4. To carry, to wear, to have

§13.4. To carry, to wear, to have

Inform has altogether five mutually exclusive ways in which one thing can be physically joined to another one:

```
containment relation - The coin is in the purse.
support relation - The coin is on the table.
incorporation relation - The coin is part of the sculpture.
carrying relation - The coin is carried by Peter.
wearing relation - The jacket is worn by Peter.
```

This is why we cannot have

```
The coin is on the table.
The coin is part of the table.
```

simultaneously, and it is a rare exception to the general rule that having one relation does not affect having another.

But there is also a sixth relation used in Inform for these meanings: the possession relation, which is the meaning of the verb "to have". At first sight this looks the same as the carrying relation, but in fact it is a convenient shorthand for "carrying or wearing", provided for conditions rather than assertions:

```
if Mr Darcy has a wet shirt ...
```

will be true during play if he is either carrying or wearing the shirt.

Still another relation exists which can be tested, but not declared to be true or false: the concealment relation, which is the meaning of the verb "to conceal". So we can ask:

```
if Mr Darcy conceals a fob watch ...
```

↑ Start of Chapter 13: Relations

← Back to §13.3. What are relations?

→ Onward to §13.5. Making new relations

↓ Example 224: ★ **Celadon** Using the enclosure relation to let the player drop things which he only indirectly carries.

↓ Example 225: ★ **Interrogation** A wand which, when waved, reveals the concealed items carried by people the player can see.

---

§13.5. Making new relations

We can create new relations like so:

```
Loving relates various people to one person.
```

Every relation has a name which ends with the word "relation", and in this case the name is "loving relation". While the name is often just two words long, as here, it doesn't have to be:

```
Adept sensitivity relates one person to one vehicle.
```

makes the "adept sensitivity relation". (The limit is 32 words.)

In such a definition, we have to say what kind of thing appears on the left and right of any relation, and also whether "one" or "various" possibilities can exist. In the example

```
Loving relates various people to one person.
```

what we are saying is that only people love; that they only love people; and that each person loves only one other person (at any given moment).

The "various" part comes in because, for instance, we might have:

> Verenka *loving relation* Stankevich
> Liubov *loving relation* Stankevich

so that various people (Verenka and Liubov, to name but two) love one person (Stankevich). But we are forbidding anyone to love two other people at the same time: Stankevich must decide which of them to love, or pick someone else, or no-one at all. Similarly, we would not allow

> Liubov *loving relation* Belinsky

It is sometimes convenient to give a name to the other side of a relationship, so to speak. We might imagine:

```
Pet-ownership relates various animals to one person (called the owner).
```

It would then make sense to talk about "the owner of Loulou", and we could have phrases like "now Flaubert is the owner of Loulou" or "if the owner of Loulou is a woman..." and so forth. This, however, would not be allowed:

```
Pet-ownership relates various animals (called the pet) to one person.
```

because "the pet of Flaubert" would be ambiguous: he might have owned dozens.

↑ Start of Chapter 13: Relations
← Back to §13.4. To carry, to wear, to have
→ Onward to §13.6. Making reciprocal relations

§13.6. Making reciprocal relations

The relationships described in this chapter so far are by no means always reciprocated. For instance, if a stone is on a table, then it is never true that the table is also on the stone. And the question may not even be meaningful to ask. If Peter wears a jacket, the jacket does not even have the possibility of wearing Peter.

But sometimes we do want a relation which works both ways equally well. These are simple to set up:

Meeting relates people to each other.

The effect is that various people know various other people, and this is always reciprocated. If Daisy knows Sophie then, automatically, Sophie knows Daisy. This even-handedness is maintained throughout play, so that whatever changes are made it is always true that if A knows B then B knows A.

And similarly for a reciprocal relation between one and another:

Marriage relates one person to another.

In this case, we can again give a name to the partner under a relation:

Marriage relates one person to another (called the spouse).

and now, for instance, we may have that the spouse of John is Yoko and the spouse of Yoko is John.

Since many of these examples have involved people, it might be worth mentioning again that any kind can be involved, not just the "person" kind.

Start of Chapter 13: Relations

Back to §13.5. Making new relations

Onward to §13.7. Relations in groups

Example 226: ★★★ **Four Cheeses** A system of telephones on which the player can call distant persons and have conversations.

§13.7. Relations in groups

Finally, there is a kind of relation which binds even more strongly.

Nationality relates people to each other in groups.

This is a kind of relation which divides people up: we might wish to have all the Icelandic people related to each other, all the Peruvians to each other, and so on. If there were a Pacific island called Informia with one inhabitant, then that person would be related only to himself. As time goes by, we could imagine people emigrating, and so on, so that these groupings would switch: perhaps everyone would leave Belgium and, for a while, there would be no Belgian nationals at all.

The testing command RELATIONS prints out the current state of all the relations created in the source code. For instance:

```
>relations
Overlooking relates various rooms to various rooms:
    The Pub >=> the Garden
    The Garden >=> the Shrubbery
    The Shrubbery >=> the Sundial Plot
Friendship relates people to each other:
    Mr Wickham <=> Sophie
    Charlotte <=> Sophie
Marriage relates one person to another:
    Mr Wickham == Sophie
```

That can produce a lot of output. To see only a single relation, or to see it at some intermediate point in a calculation, there's also a testing phrase:

> **show relation** (relation of values to values)
>
> This phrase is for testing purposes only. It shows the current state of the named relation, that is, it shows which values relate to which other ones, where it's possible to do this in any sensible way.

But this is a phrase - not a typed command.

Start of Chapter 13: Relations

Back to §13.6. Making reciprocal relations

Onward to §13.8. The built-in verbs and their meanings

Example 227: ★ **Transmutations** A machine that turns objects into other, similar objects.

Example 228: ★★★ **Otranto** A kind of rope which can be tied to objects and used to anchor the player or drag items from room to room.

§13.8. The built-in verbs and their meanings

It is all very well to define new relations, but this does nothing if there is no way to assert that they are true, or to ask whether they are true or false. That requires a verb: in fact, a relation is nothing more than what Inform uses as the "meaning" of a verb. The assertion verbs built in to Inform have the following built-in relations as their meanings:

```
Verb - Relation
to be - equality relation
to have - possession relation
to contain - containment relation
to support - support relation
to carry - carrying relation
to wear - wearing relation
to incorporate - incorporation relation
```

Two of Inform's built-in relations are expressed using prepositions instead:

```
Preposition - Relation
to be part of - (reversed) incorporation relation
to be adjacent to - adjacency relation
```

It would be easy to make verbs for these if we wanted ("to adjoin", say) using the techniques of the next section.

The verb *to be* is grammatically different from any other, and its meaning is too complicated to be fully expressed by any one relation. A great deal of the Inform program is given over to its "meaning", which we are not allowed to change or imitate. The "equality relation" is simple enough, and is the one implied by conditions like

```
if the score is 20, ...
```

but *to be* can have more complicated implications - "if Mr Wickham is hungry" clearly doesn't test whether two quantities are equal. Fortunately the other verbs are much simpler.

There are a few other built-in verbs, as can be seen in the Index, but these are mostly for experts only. For example:

```
Verb - Relation
to mean - meaning relation
to provide - provision relation
```

"To mean" can be used to make new verbs, as we'll soon see. Provision is to do with whether something can have a given property: for example, "if R provides the property lighted" tests whether R is able to have this property, not whether it actually has it at the moment.

Start of Chapter 13: Relations
Back to §13.7. Relations in groups
Onward to §13.9. Defining new assertion verbs

§13.9. Defining new assertion verbs

Here is an example definition of a new verb:

```
The verb to sport means the wearing relation.
```

Once this is done, we can write the assertion

```
Mr Wickham sports a Tory rosette.
```

which will do the the same thing as

```
Mr Wickham wears a Tory rosette.
```

because both verbs have the same relation as their meaning.

Earlier versions of Inform needed to be told how to make other parts of the verb, but that's rarely true now. Just writing:

```
The verb to sport means the wearing relation.
```

is enough for Inform to understand "he sports", "they sport", "he sported", "it is sported", "he is sporting", "he had sported" and so on. It works with irregular verbs, too; it has a very comprehensive dictionary. But it's legal to spell out the conjugation if need be:

```
The verb to sport (he sports, they sport, he sported, it is sported) implies the knowledge
relation.
```

Occasionally it's convenient to have the relation the other way around. For instance:

```
The verb to grace means the reversed wearing relation.
```

With that defined, these two sentences have identical meanings:

```
Mr Wickham sports a Tory rosette.
A Tory rosette graces Mr Wickham.
```

Reversed in this sense means that the things related - the subject and object of the verb - are the other way round.

The Phrasebook index contains all the verbs associated with assertions, in the Verbs section. When we add new verbs to our source, those will appear in the Phrasebook as well.

The verbs above ("to grace", "to sport") are short ones, but we're free to make them longer than that. For example:

```
The verb to cover oneself with means the wearing relation.

Peter is covering himself with a tent-like raincoat.
```

Here we have "to cover oneself with", four words long; the limit is 29.

⬆ Start of Chapter 13: Relations

⬅ Back to §13.8. The built-in verbs and their meanings

➡ Onward to §13.10. Defining new prepositions

⬇ Example 229: ★ **Unthinkable Alliances** People are to be grouped into alliances. To kiss someone is to join his or her faction, which may make a grand alliance; to strike them is to give notice of quitting, and to become a lone wolf.

⬇ Example 230: ★★★ **The Unexamined Life** An adaptive hint system that tracks what the player needs to have seen or to possess in order to solve a given puzzle, and doles out suggestions accordingly. Handles changes in the game state with remarkable flexibility, and allows the player to decide how explicit a nudge he wants at any given moment.

§13.10. Defining new prepositions

The term preposition is used here, a little loosely, to mean anything which we add to the verb *to be* in order to talk about some relation or other. We have seen many examples already, such as:

```
To be in - The ball is in the box.
To be part of - The lever is part of the slot machine.
```

These are defined just the way verbs are. Compare the following:

```
Suspicion relates various people to one person.

The verb to suspect means the suspicion relation.

The verb to be suspicious of means the suspicion relation.
```

The result of this is that

```
Hercule Poirot suspects Colonel Hotchkiss.
Hercule Poirot is suspicious of Colonel Hotchkiss.
```

are exactly equivalent, and so are these two descriptions:

```
somebody who suspects Colonel Hotchkiss
somebody suspicious of Colonel Hotchkiss
```

While most prepositions are short ("in", "part of", "suspicious of"), they're free to be longer if need be ("inordinately far away from"): the limit is 30 words, which should be ample.

We can also define verbs as auxiliaries, like so:

```
The verb to be able to approach means the approachability relation.
```

Now we can ask if Poirot "can approach" Hotchkiss, and so on.

⬆ Start of Chapter 13: Relations

⬅ Back to §13.9. Defining new assertion verbs

➡ Onward to §13.11. Indirect relations

⬇ Example 231: ★ **The Abolition of Love** A thorough exploration of all the kinds of relations established so far, with the syntax to set and unset them.

⬇ Example 232: ★ **Swerve left? Swerve right? Or think about it and die?** Building a marble chute track in which a dropped marble will automatically roll downhill.

⬇ Example 233: ★ **Beneath the Surface** An "underlying" relation which adds to the world model the idea of objects hidden under other objects.

⬇ Example 234: ★★★ **Bogart** Clothing for the player that layers, so that items cannot be taken off in the wrong order, and the player's inventory lists only the clothing that is currently visible.

§13.11. Indirect relations

We have already seen, in the chapter on Descriptions which is a forerunner of this one, that Inform provides not only "adjacent" as a way of seeing if one room is directly connected to another, but also "the best route from A to B", which allows us to see if any sequence of moves connects them.

Something similar - in fact, simpler - is allowed for any relation between objects. Suppose we would like to go sledging: we can go downhill, but not up. Some quite distant places may be reachable, while others close by may not be, even if lower than us, because they would involve climbing again at some point. The following would implement this:

```
Overlooking relates various rooms to various rooms.

The verb to overlook means the overlooking relation.

The Garden overlooks the Shrubbery. The Folly overlooks the Garden. The Shrubbery overlooks the Sundial Plot. The Old Ice House overlooks the Garden.

After looking:
    say "This wintry vantage point overlooks [the list of rooms overlooked by the location].";
    let the way be the next step via the overlooking relation from the location to the Sundial Plot;
    if the way is a room, say "To sledge downhill to the Sundial, aim for [the way].";
    otherwise say "It is not possible to sledge downhill to the Sundial."
```

Here we're making use of:

next step via (relation of values to values) **from** (object) **to** (object) **... object**

This phrase tries to find a shortest route between the two given endpoints, using the given relation of objects to determine single steps. Example:

```
next step via the overlooking relation from the Folly to the Chinese Lake
```

The result is the special object value "nothing" if the two endpoints are the same or if no route exists.

number of steps via (relation of values to values) **from** (object) **to** (object) **... number**

This phrase tries to find the length of a shortest route between the two given endpoints, using the given relation of objects to determine single steps. Example:

```
number of steps via the overlooking relation from the Folly to the Chinese Lake
```

The result is 0 if the two endpoints are the same, or -1 if no route exists.

Another example would be the "six degrees of separation" game, where it is claimed that any two people on Earth are connected by a sequence of up to six acquaintances. In an Inform implementation, we might talk about "the next step via the friendship relation from George Bush to Saddam Hussein", for instance, a phrase likely to evaluate to Donald Rumsfeld, and then

```
the number of steps via the friendship relation from George Bush to Saddam Hussein
```

would be... but that would be telling.

As with route-finding through the map, finding "the next step via" a relation can be slow. For instance, suppose we have dozens of articles of clothing all partially revealing each other, connected by two relations - overlying and underlying. Then "the next step via" these relations allows us to establish what can be worn on top of what else. If we need to calculate this often, and there are enormous wardrobes of clothes to choose from, speed starts to matter.

Once again there is a choice of algorithms: "fast" and "slow", where "fast" needs much more memory. To make route-finding for a given relation "fast", we have to declare it that way:

```
Overlying relates various garments to various garments with fast route-finding.
Overlapping relates various garments to each other with fast route-finding.
```

Otherwise, the "slow" method will be used.

This "with fast route-finding" note can only be added to various-to-various relations. (Although route-finding through various-to-one and one-to-various relations is fully supported, it exploits the relative simplicity of these problems to use a more efficient algorithm than either "fast" or "slow".)

★ *See* **Adjacent rooms and routes through the map** *for route-finding through the map rather than a relation*

↑ Start of Chapter 13: Relations

← Back to §13.10. Defining new prepositions

→ Onward to §13.12. Relations which express conditions

↓ Example 235: ★★★ **The Problem of Edith** A conversation in which the main character tries to build logical connections between what the player is saying now and what went immediately before.

§13.12. Relations which express conditions

One last way to create a new relation and, in many ways, the easiest of all. If we write:

```
Contact relates a thing (called X) to a thing (called Y) when X is part of Y or Y is part of
X. The verb to be joined to means the contact relation.
```

then we would be able to talk about a handle being joined to a door, and a door being joined to a handle, and so on. We are not allowed to declare:

```
The hook is joined to the line.
```

because the question of whether they are joined is not for us to decide: that will be for the condition to determine, whenever we test it. Similarly, we cannot meaningfully write

```
now the hook is joined to the line;
```

(and Inform will not let us) because this relation is not something we can force either way: we can make it come true by other means, maybe, but we cannot simply make it true by saying so. Lastly, this kind of relation is restricted in that we are not allowed to find paths or calculate numbers of steps through it.

So this way to define relations is, on the face of it, just a sort of verbal trick to write conditions in a more attractive way. The more flexible, changeable relations in previous sections have much greater expressive power. All the same, it is nice to be able to write -

```
Nearness relates a room (called A) to a room (called B) when the number of moves from B to A is less than 3. The verb to be near means the nearness relation.
```

and then to be able to write rules like:

```
Instead of listening when the location is near the Sundial: say "You hear a splashing of water."
```

As with other relations, there's no reason why we have to use objects. For example:

```
Material is a kind of value. The materials are wood and metal. A thing has a material.

Materiality relates a thing (called X) to a material (called Y) when Y is the material of X. The verb to be made of means the materiality relation.
```

which enables us to write:

```
if the cube is made of wood, ...
say "The carpenter looks at [the list of things which are made of wood].";
```

And here is a mathematical one:

```
Divisibility relates a number (called N) to a number (called M) when the remainder after dividing M by N is 0. The verb to divide means the divisibility relation. The verb to be a factor of means the divisibility relation.
```

We now find that "2 divides 12", "5 is not a factor of 12" and "12 is divisible by 3" are all true. Again, we are only really gaining a nice form of words, but improving the clarity of the source text is never a bad thing.

Start of Chapter 13: Relations

Back to §13.11. Indirect relations

Onward to §13.13. Relations involving values

Example 236: ★ **Wainwright Acts** A technical note about checking the location of door objects when characters other than the player are interacting with them.

Example 237: ★★★ **A Humble Wayside Flower** Relations track the relationships between one character and another. Whenever the player meets a relative of someone he already knows, he receives a brief introduction.

§13.13. Relations involving values

Although most of the examples in this chapter have involved objects, relations can connect almost any values together. We can create relations in groups, one to various relations, various to one relations, one to one relations, and various to various relations for any combination of kinds. For example:

```
Partnership relates various texts to various texts.

The verb to belong with means the
partnership relation.

"cheese" belongs with "crackers".
"clam" belongs with "chowder".
```

How might we make use of this? Clearly it would be impractical to keep trying:

```
if "caviar" belongs with "aardvarks", ...
if "caviar" belongs with "abacuses", ...
...
```

to find out what "caviar" belongs with. It's still harder to find out if it belongs with anything at all -- in theory we would have to try every possibility, which of course is impossible. Instead we have these phrases:

if (value) **relates to** (name of kind) **by** (relation of values to values)**:**

This condition is true if the value V is such that V relates to something by the given relation. Example: suppose partnership relates various texts to various texts. Then we can test

```
if "chalk" relates to a text by the partnership relation, ...
```

if (name of kind) **relates to** (value) **by** (relation of values to values)**:**

This condition is true if the value V is such that something relates to V by the given relation. Example: suppose partnership relates various texts to various texts. Then we can test

```
if a text relates to "cheese" by the partnership relation, ...
```

If a partner does exist, then we can find it with:

(name of kind) **to which**/whom (value) **relates by** (relation of values to values) **... value**

or:

(name of kind) **that**/which/whom (value) **relates to by** (relation of values to values) **... value**

This phrase produces an Y such that the given value V relates to Y by the given relation. Example: suppose partnership relates various texts to various texts. Then we can obtain

```
the text to which "chalk" relates by the partnership relation
```

which might be, say, "cheese". It's a run-time problem to use this if no such Y exists.

(name of kind) **that**/which/who **relates to** (value) **by** (relation of values to values) **... value**

This phrase produces an X such that X relates to the given value V by the given relation. Example: suppose partnership relates various texts to various texts. Then we can obtain

```
the text which relates to "cheese" by the partnership relation
```

which might be, say, "chalk". It's a run-time problem to use this if no such X exists.

Of course, there might be many answers to this question, so perhaps these are neater:

list of (name of kind) **that**/which/who **relate to** (value) **by** (relation of values to values) **... value**

This phrase produces a list of all the X such that X relates to the given value V by the given relation. Example: suppose partnership relates various texts to various texts. Then we can obtain

```
list of texts which relate to "cheese" by the partnership relation
```

which might be, say, { "chalk", "grapes", "macaroni" }. The answer might be the empty set, but that's not a problem.

list of (name of kind) **to which**/whom (value) **relates by** (relation of values to values) **... value**

or:

list of (name of kind) **that**/which/whom (value) **relates to by** (relation of values to values) **... value**

This phrase produces a list of all Y such that the given value V relates to Y by the given relation. Example: suppose partnership relates various texts to various texts. Then we can obtain

```
list of texts to which "chalk" relates by the partnership relation
```

which might be, say, { "cheese", "blackboard", "cliffs" }. The answer might be the empty set, but that's not a problem.

Finally, it's sometimes useful to get at the list of all values which can appear on the left or right hand side of a relation. We need tongue-twister like wording to do it, but:

list of (name of kind) **that**/which/whom (relation of values to values) **relates ... value**

This phrase produces a list of all X which relate to anything under the given relation. Example: suppose partnership relates various texts to various texts. Then we can obtain

```
list of texts which the partnership relation relates
```

list of (name of kind) **to which**/whom (relation of values to values) **relates ... value**

or:

list of (name of kind) **that**/which/whom (relation of values to values) **relates to ... value**

This phrase produces a list of all Y which anything relates to under the given relation. Example: suppose partnership relates various texts to various texts. Then we can obtain

```
list of texts which the partnership relation relates to
```

For efficiency reasons, there are no guarantees about what order these lists have - but they can of course always be sorted when found.

Start of Chapter 13: Relations

Back to §13.12. Relations which express conditions

Onward to §13.14. Relations as values in their own right

Example 238: ★ **Meet Market** A case in which relations give characters multiple values of the same kind.

Example 239: ★★★ **For Demonstration Purposes** A character who learns new actions by watching the player performing them.

§13.14. Relations as values in their own right

As we've seen, most relations have names - "containment relation", for instance. These are themselves values in Inform, though there are a few restrictions on how they are used. (Relations can contain a colossal amount of data, so we don't want to have to copy them casually.)

Consider these two examples:

```
Parity relates a number (called N) to a number (called M) when N minus M is even.

Joint magnitude relates a number (called N) to a number (called M) when N plus M is
greater than 7.
```

Here "parity relation" and "joint magnitude relation" are both values of the same kind: "relation of numbers to numbers". In general, every relation is a value of kind "relation of K to L", for the appropriate kinds K and L. So the parity relation doesn't have the same kind as the containment relation, for example. Because it often happens that K and L are the same, we can just say "relation of K" in this case, so we could equally say that the kind of the parity relation is "relation of numbers".

This is useful to know when writing phrases like so:

```
To chart (R - a relation of numbers):
    repeat with N running from 1 to 5:
        repeat with M running from 1 to 5:
            if R relates N to M, say "[N] <=> [M] ";
        say "[line break]";
```

and now "chart parity relation" will work nicely, but "chart visibility relation" will be rejected (as it should be, because it relates things, not numbers). In general, if R is any relation, we can write

```
if R relates X to Y, ...
now R relates X to Y;
now R does not relate X to Y;
```

to test, set and unset a relation R between two values. (Inform checks that the values X and Y have the right kind and produces a problem message if not.)

Several useful adjectives can be applied to relations:

```
"empty" - nothing relates to anything else
"symmetric" - by definition X relates to Y if and only if Y relates to X
"equivalence" - this is a relation "in groups", or an "equivalence relation"
"one-to-one" - it relates one K to one L
"one-to-various" - similarly
"various-to-one" - similarly
"various-to-various" - similarly
```

So for example it's possible to ask

```
if R is a symmetric one-to-one relation of texts, ...
```

With some relations, it's possible to clear them out by writing:

```
now R is empty;
```

and with temporary relations (see the next section), it's even possible to change their valencies (one-to-one vs. one-to-various, etc.) using "now", but only when they are empty. The exceptions where "empty" can't be used are those which can't be changed at all, like the parity relation above, and a few built-in cases such as the support, containment and incorporation relations, where emptying would dissolve the model world in a disastrous way.

Start of Chapter 13: Relations

Back to §13.13. Relations involving values

Onward to §13.15. Temporary relations

Example 240: ★ **Number Study** The parity and joint magnitude relations explored.

§13.15. Temporary relations

So far in this chapter, we've only seen relations which exist permanently during play. The relationships might change - sometimes Red Riding Hood would be in the Woodcutter's Cottage, sometimes not - but the relations themselves were eternal.

In fact, though, we can also create relations to be dynamic data structures, like lists:

let (a name not so far used) **be** (description of relations of values to values)

This phrase creates a new temporary variable, and sets its value to the identity of a newly created and equally temporary relation. These last only for the present block of phrases, which certainly means that they exist only in the current rule. Example:

```
let the password dictionary be a relation of texts;
```

This makes a purely temporary various-to-various relation between texts, which lasts as long as the temporary value "password dictionary" lasts. By default, relations are various-to-various, but we could instead write, say:

```
let the nicknames catalogue be a various-to-one relation of texts;
```

Such a relation exists only in the current phrase, and is destroyed when the phrase finishes, like any other "let". Of course there's no verb whose meaning is this relation, but that's no obstacle, because we can manipulate it using "relates":

```
now the nicknames catalogue relates "Trudy" to "Snake-eyes";
```

(At present such a relation cannot be used outside its own phrase.)

⬆ Start of Chapter 13: Relations
⬅ Back to §13.14. Relations as values in their own right
➡ Onward to §13.16. What are relations for?

§13.16. What are relations for?

It is easy to say what verbs are for: they are to express relations. But what are relations for?

Inform 7's focus on relations between objects is unusual as an approach to interactive fiction; the concept does not exist in most design systems, or rather, it does but is submerged. Traditional design systems do, after all, have the spatial relations of being inside, on top of, and so on. It could well be said that these are the only relationships that inanimate objects ever have. A stone can be on top of a table, and if so then that expresses their entire association.

This is because the stone, and the table, have no opinions, emotions, knowledge or memory. If the stone is taken away and then put back, nothing has changed. People, on the other hand, tend to remember having met each other before; they like being in some places, but not others; their behaviour depends on who, or what, is nearby. Being conscious, they have internal states, unlike the stone. Relations are a simple but powerful way to express and talk about such connections, and although they have numerous uses in physical contexts too, they are at their most powerful when helping to make the characters of interactive fiction come alive.

Start of Chapter 13: Relations

Back to §13.15. Temporary relations

Onward to Chapter 14: Adaptive Text and Responses: §14.1. Tense and narrative viewpoint

Example 241: ★★★ **Murder on the Orient Express** A number of sleuths (the player among them) find themselves aboard the Orient Express, where a murder has taken place, and one of them is apparently the culprit. Naturally they do not agree on whom, but there is physical evidence which may change their minds...

Example 242: ★★★ **What Not To Wear** A general-purpose clothing system that handles a variety of different clothing items layered in different combinations over different areas of the body.

Example 243: ★★★★ **Mathematical view of relations** Some notes on relations from a mathematical point of view, provided only to clarify some technicalities for those who are interested.

Example 244: ★★★★ **Graph-theory view of relations** Some notes on relations from the point of view of graph theory.

Examples from Chapter 13: Relations

Start of this chapter

Chapter 14: Adaptive Text and Responses

Indexes of the examples

223 ★★★★ **Example Formal syntax of sentences** *RB*

A more formal description of the sentence grammar used by Inform for both assertions and conditions.

An entire grammar for the whole mass of Inform would not be linguistically interesting: it contains many convenient wordings which are not really part of a grand pattern. Inform does, however, have a formal notion of a Sentence, a grammatical structure which we shall call S. It is almost true that conditions ("if the flowerpot is on the wall") have the same grammar as assertions ("The flowerpot is on the wall") and "now" phrases ("now the flowerpot is on the wall"). All three use the S grammar, so we could define an assertion as "S.", say that "if S", "while S", "when S" and so on are conditions, and say that "now S" defines the "now" declaration.

Grammatical sentences do not necessarily make sense, of course. Many perfectly grammatical assertions in fact give rise to problem messages:

> The wicker basket is not in the kitchen. (*Unhelpfully negative.*)
> The wicker basket has been in the kitchen. (*Talks about a time which never existed.*)
> The wicker basket is full. (*Full of what? Too vague.*)
> The wicker basket is the ginger cat. (*Demonstrably false.*)

Whereas the first three, at least, would be sensible as conditions. So saying that assertions are "just like" conditions is a little misleading: what they have in common is S, the underlying grammar they each use as a starting-point.

To define S, we break it up into subsidiary structures. The most important is the Description Phrase (DP), examples of which include "the red basket", "somewhere lighted" and "an empty open container". Clearly sentences include DPs, but they also include other ingredients. The general pattern used in Inform is very simple:

```
1. S = DP + VP
2. VP = Verb + DP
```

where VP is another structure, the Verb Phrase. For instance:

```
S (The horseman wears a riding helmet)
   = DP (The horseman) + VP (wears a riding helmet)
VP (wears a riding helmet)
   = Verb (wears) + DP (a riding helmet)
```

In that example, the Verb was the single word "wears". More generally, Inform allows a Verb to include adverbs and prepositions, to be negated, and to come in any of four tenses, so the following are all valid examples of Verb in our grammar:

```
wore
carries
is carried by
had not been inside
```

Although we are not going through the definition of Description Phrases in detail, it is worth noticing how "which" and "who" behave:

```
3a. DP = DP + which + VP
3b. DP = DP + who + VP
```

Thus "an open container which is in the Ballroom" can be broken down as:

```
DP (an open container) + which + VP (is in the Ballroom)
```

To understand compounds like "something in a container", we have to invent a new grammatical structure for "in a container" and similar: let's call this a Relative Phrase (RP).

```
4. DP = DP + RP
```

Thus "an open container in the Ballroom" is DP (an open container) + RP (in the Ballroom). Relative Phrases have two different forms:

```
5a. RP = Preposition + DP
5b. RP = Participle + DP
```

so that "in a container" is an example of 5a. An example of 5b would be

```
RP (worn by Mr Darcy) = Participle (worn by) + DP (Mr Darcy)
```

That is nearly it, but not quite: we must go back to the "almost" in the statement above that assertions and conditions "almost" have the same grammar S. The difference arises from a curious irregularity in English called subject-verb inversion (see the *Oxford English Grammar* at 3.22F), whereby assertions can be reversed but not conditions. For instance,

```
In the Garden is a sunflower.
```

This does not follow the pattern S = DP + VP, because "in the garden" is not a DP: indeed, it is not a noun at all. To make sense of this sentence, Inform reverses it to "A sunflower is in the Garden", which does indeed follow DP + VP. Hence the final rule:

```
6 (assertions only). S = RP + Verb + DP
```

So the condition "if in the garden is a sunflower..." fails because rule 6 does not apply to the grammar for conditions: while occasional poetic uses of subject-verb inversion do turn up in conditions ("If On A Winter's Night A Traveller", say), they are rare in ordinary English usage, and illegal in Inform. That completes the S grammar, so to recap:

```
1. S = DP + VP
2. VP = Verb + DP
3a. DP = DP + which + VP
3b. DP = DP + who + VP
4. DP = DP + RP
5a. RP = Preposition + DP
5b. RP = Participle + DP
6 (assertions only). S = RP + Verb + DP
```

224

★ Example **Celadon**

RB

Using the enclosure relation to let the player drop things which he only indirectly carries.

By default, Inform only lets the player drop those things which he is carrying -- that is, those directly in his possession. Things inside satchels or on portable trays have to be taken first.

If we want to change this behavior, we might add a dropping rule that distinguishes between carrying and mere enclosure (introduced back in "The location of something" in the chapter on Things):

```
"Celadon"

The Tea Room is a room. The player carries a black lacquer tray. The lacquer tray is portable. On the lacquer tray are a celadon teapot and a napkin.

Before dropping something:
    if the player does not carry the noun and the player encloses the noun:
```

```
		say "(first taking [the noun] from [the holder of the noun])[command clarification break]";
		silently try taking the noun;
		if the player does not carry the noun, stop the action.

Instead of taking the napkin:
	say "It seems to be stuck to the tray, possibly by an underlying wad of gum."

Test me with "i / drop teapot / i / look / drop teapot / drop napkin / i / drop tray".
```

---

225 ★ **Example** **Interrogation** RB

A wand which, when waved, reveals the concealed items carried by people the player can see.

```
"Interrogation"

The X-Ray Vision Wand is carried by the player.

Instead of waving the X-Ray Vision Wand:
	say "Disappointingly, nothing happens."

Instead of waving the X-Ray Vision Wand when the player can see someone who is concealing something:
	say "The wand glows green. Immediately you see on the monitor [a list of things which are concealed by people who can be seen by the player]."

After printing the name of a thing (called target) which is carried by someone while waving the wand:
	say " (carried by [a random person who carries the target])"

The Interrogation Chamber is a room. "Despite its gothic name, this room is a rather civilized place for your work, with large plate-glass windows and a fitted carpet."

A thing can be secret or obvious. A thing is usually obvious.

Brian is in the Interrogation Chamber."Brian lounges against the wall." Brian carries a quantity of plastic explosive. The explosive is secret.

Janine is in the Interrogation Chamber. "Janine toys nervously with a laptop bag." Janine carries a chocolate biscuit, a laptop bag, and a microfilm. The microfilm is secret. The laptop bag is wearable. In the bag is a laptop computer.

Rule for deciding the concealed possessions of something: if the particular possession is secret, yes; otherwise no.

Instead of examining someone: say "[The noun] is openly carrying [a list of unconcealed things carried by the noun]."

Test me with "wave wand / examine janine / examine brian".
```

226

Example Four Cheeses

RB

A system of telephones on which the player can call distant persons and have conversations.

```
"Four Cheeses"

Section 1 - Telephones and Connections

A telephone is a kind of thing. Understand "phone" as a telephone.

Understand "call [any telephone] on [something]" as calling it on. Understand "call [any telephone]" as calling it on. Understand the commands "dial" or "phone" or "telephone" as "call". Understand "call [any known person]" as calling it by name on.

Connection relates one thing to another (called the other party).

The verb to reach means the connection relation.

Calling it on is an action applying to one visible thing and one thing.

Check calling it on:
    if the second noun is not a telephone, say "[The second noun] is unlikely to be much use in that respect." instead;
    if the second noun is the noun, say "You get a busy signal." instead.

Carry out calling it on:
    if a person (called the listener) can see the noun, now the player reaches the listener.
```

Because we've said that connection is a reciprocal, one-to-one relationship, Inform will do the rest of the bookkeeping: if (for instance) we telephone someone else, the first connection will be broken automatically.

```
Report calling it on:
    say "'Hello?' says [the other party of the player]."
```

To avoid annoyance, we should also let the player use CALL #### as well as CALL #### ON TELEPHONE. A rule from the chapter on Activities comes in handy here:

```
Rule for supplying a missing second noun while calling something on:
    assign a phone.

To assign a phone:
    if the player can touch a telephone (called the current phone):
        say "(on [the current phone])[line break]";
        now the second noun is the current phone;
    otherwise:
        say "You don't have a phone handy."
```

Things might be a little more complicated if we had cell phones that could be moved around, but for right now the player can only touch a maximum of one phone at a time.

Suppose we further want to allow the player to call people up by name, but only if they've already been encountered or are familiar to the player for some reason.

```
A person can be known or unknown.

Understand "call [any known person]" as calling it by name on.

Understand "call [any known person] on [something]" as calling it by name on.

Rule for supplying a missing second noun while calling something by name on:
    assign a phone.

Calling it by name on is an action applying to one visible thing and one thing.

Check calling it by name on:
    if the noun is in the location, say "[The noun] is right here." instead.

Carry out calling it by name on:
    if the noun can touch a telephone (called the link), try calling the link on the second noun;
    otherwise say "You can't reach [the noun]." instead.

Before calling something on something when the player reaches someone:
    say "(first ending your conversation with [the other party of the player]) [command clarification break]";
    end current conversation.

Understand "hang up [something]" as hanging up.

Hanging up is an action applying to one thing.

Check hanging up:
    if the noun is not a telephone, say "You can't hang up [the noun]." instead;
    if the player does not reach someone, say "You're not on the line with anyone." instead.

Carry out hanging up:
    now the player does not reach anyone.

Report hanging up:
    say "You put down [the noun], cutting the connection."

Before going somewhere when the player reaches someone:
    say "(first hanging up on [the other party of the player])[command clarification break]";
    end current conversation.
```

And finally we want to make sure that calling random other numbers produces a sensible result:

```
Understand "call [text]" as misdialling. Misdialling is an action applying to one
topic. Carry out misdialling: say "The phone rings and rings but no one answers."

Understand "call 911" or "call 999" or "call police" or "call fire department" as a
mistake ("After strict warnings, you've given up making prank calls to emergency
services.").

Before misdialling when the player reaches someone:
    say "(first ending your conversation with [the other party of the player])
[command clarification break]";
    end current conversation.

To end current conversation:
    let the current phone be a random telephone which can be touched by the
player;
    silently try hanging up the current phone.

After deciding the scope of the player while the player reaches someone:
    place the other party of the player in scope, but not its contents.
```

A note about this scope addition: the player can refer to the other party whenever he has the other person on the phone. He can't, however, see or refer to anything that person might be holding or wearing, thanks to the "but not its contents" option.

Furthermore, the player can't actually do anything to that person that requires touching. That's because of the reaching inside rules, which govern whether the player can reach through intervening barriers such as rooms. (See the Advanced Actions chapter for more about changing reachability.) There are two things we might want to be careful about, though.

First, we should specifically disallow the player from looking at the person on the other end of the line. Since sight doesn't require touching, the reaching inside rules will not be consulted about a command such as EXAMINE BOSS or LOOK UNDER BOSS. We can, however, intervene in such cases using the visibility rules, which are consulted for any actions that "require light" (including EXAMINE and LOOK UNDER). Here again we borrow some options from the Advanced Actions chapter:

```
To decide whether acting through the line:
    if the noun is something and the location of the noun is not the location of the
player:
        yes;
    if the second noun is something and the location of the second noun is not the
location of the player:
        yes;
    no.

Visibility rule when acting through the line:
    there is insufficient light.

Rule for printing a refusal to act in the dark when acting through the line:
    say "You're not on a video phone, so you can only hear." instead.
```

Second, though the existing reaching inside rules are adequate to stop us from touching the person on the other end of the line, the response that's currently printed is a bit generic: it just says "You can't reach into [the room containing the person]." Let's add our own custom reply, instead:

```
A rule for reaching inside a room (called destination):
    if the other party of the player is enclosed by the destination:
        say "Though you're on the line with [the other party of the player], you can't physically reach to [the destination].";
        deny access.

Section 2 - Conversation over the Phone, In General
```

This portion supplies a simple method of conversation; but we could substitute some completely different conversation system if appropriate. The effect of the telephones is that we are allowed to talk to characters in distant locations under certain circumstances, after which the usual conversation rules apply.

```
Instead of listening to a telephone when the player reaches someone:
    say "You can hear [the other party of the player] breathing."

Before listening to someone when the player cannot touch the noun:
    say "[The noun] is waiting for you to carry on the conversation." instead.

A person has a table name called chatter.

Before telling someone about something:
    try asking the noun about it instead.

Before answering someone that something:
    say "Best to confine your conversation to questions and answers." instead.

Before asking someone about something:
    if the topic understood is a topic listed in the chatter of the noun, say "[reply entry][paragraph break]" instead;
    otherwise say "[The noun] does not reply." instead.

Section 3 - The Scenario

The Guard House is a room. "Here you spend all your nights. Bullet-proof windows offer a panoramic view of serene cliffs, palm trees, and a moonlit ocean. Occasionally someone is foolish enough to try a cliff ascent or even an attack by helicopter, but lately things have been pretty quiet.

The mansion is up the hill behind you, security lights ablaze."

The grey telephone is a telephone in the Guard House. Understand "6885" as the grey telephone. "Before you is a grey telephone. In black marker someone has written on it: MAIN OFFICE 2802."

Before going a direction in the Guard House, say "And leave your post? The boss would have you flayed. No kidding." instead.
```

In a game where the player could walk around, we would of course want to add a before rule so that he automatically hung up any phone he was using before leaving the room.

The Main Office is a room. The boss is a known woman in the Main Office. A telephone called the red telephone is in the Main Office. Understand "2802" as the red telephone.

The Guild is a room. The ninja is an unknown man in the Guild. A telephone called the black telephone is in the Guild. Understand "4431" as the black telephone.

Potter's Pizza is a room. The pizza delivery boy is a known man in Pizza. A telephone called the saucy telephone is in Pizza. Understand "8885" as the saucy telephone.

The chatter of the boss is the Table of Boss Conversation. The chatter of the delivery boy is the Table of Pizza Conversation. The chatter of the ninja is the Table of Ninja Conversation.

After calling the red telephone on something for the first time:
    say "'Yes?' asks the boss. Her voice is especially husky this evening. Maybe that night of passion isn't so far off after all."

Table of Boss Conversation

topic	reply
"love/passion/tonight/night" or "night of passion"	"'...Sorry, what?' she asks. 'I wasn't listening.' Oh. Maybe she'd go for some pizza, though."
"pizza"	"'I'd love some. No pepperoni, though,' she says, sounding dreamy. Yes, this is definitely time for a call to your old friend, the pizza boy."
"imminent ninja attack"	"'Don't worry about it,' says the boss crisply. 'I have everything under control.'"

Table of Pizza Conversation

topic	reply
"pizza"	"'Pepperoni special tonight!' he says proudly."
"pepperoni"	"'Pepperoni is included free on ALL our pizzas,' he says proudly."
"no pepperoni"	"'Well, I don't see why you'd want that,' replies the boy sniffily. 'It's free!'"
"jalapeno"	"'Sorry, we're out of jalapenos this evening. There was a run on them.'"
"sausage"	"'Sausage, sure, we can do you sausage.'"
"canadian bacon"	"'There's currently an embargo on Canadian pig products.'"
"cheese"	"'We use four kinds,' says the boy, then lowers his voice confidentially. 'Actually, two of them are the same. Nobody ever counts. The stringy one, the one that comes in dollops and the orangey one. You know.'"
"pineapple"	"'We could put pineapple on there, sure,' says the delivery boy, in a tone that lets you know his opinion of people who order fruit-based pizzas."
"pineapple and garlic"	"'What kind of crazy combination is that?' demands the delivery boy, finally losing all self-control."
"delivery"	"'Well, I don't know,' says the boy in a worried voice. 'Last time I came there were attack dogs. And ninjas.'"
"massive gratuity"	"'There's no use in a big tip you don't live to spend,' says the delivery boy quite firmly."

Table of Ninja Conversation

topic	reply
"imminent attack"	"'Yes, still on for tonight,' confirms the voice at the other end of the line."
"pizza delivery	"The voice, in tones of velvet, indicates that it cannot guarantee the safety of any delivery

```
boy"               persons whomsoever."
```

```
Test me with "call 2802 / examine boss / ask boss about night of passion / ask
boss about pizza / listen to telephone / call delivery boy on telephone / ask boy
about cheese / tell boy about no pepperoni / ask boy about delivery / tell boy
about massive gratuity / attack boy".
```

...and it more or less writes itself from there.

---

227 ★ **Example** **Transmutations** *RB*

A machine that turns objects into other, similar objects.

Suppose we want to have a machine in our game that can transmute one item into another, similar object with different properties: a bag of jelly beans into a bag of jewels, for instance. Thus each item will be associated with some number of equivalents -- the other objects it can turn into. This is a handy use for group relations:

```
"Transmutations"

Workshop is a room.

Transmutation relates things to each other in groups. The verb to become
means the transmutation relation.

Definition: a thing is transmutable if it becomes more than one thing. [* It always
becomes itself.]

A thing can be valuable. Something valuable called a bag of jewels is carried by
the player. It becomes the bag of gunpowder and the bag of jelly beans.

A thing can be dangerous. The bag of gunpowder is a dangerous thing.

The bag of jelly beans is an edible thing.

The machine is fixed in place in the workshop.

The can't insert into what's not a container rule does nothing when inserting
something into the machine.

Check inserting something which is not transmutable into the machine:
    instead say "You can't transmute that."

To decide which thing is new form of (obj - edible thing): decide on a random
valuable thing which becomes obj.

To decide which thing is new form of (obj - dangerous thing): decide on a
random edible thing which becomes obj.
```

```
To decide which thing is new form of (obj - valuable thing): decide on a random dangerous thing which becomes obj.

Carry out inserting something into the machine:
    now the noun is nowhere;
    now the player carries the new form of the noun.

Report inserting something edible into the machine:
    say "The machine clicks, whirrs, and spits out [a new form of the noun]. You're rich!";
    rule succeeds.

Report inserting something dangerous into the machine:
    say "The machine clicks, whirrs, and in a shower of flavor crystals, spits out [a new form of the noun].";
    rule succeeds.

Report inserting something valuable into the machine:
    say "The machine clicks, whirrs, and with a violent roar, spits out [a new form of the noun].";
    rule succeeds.

    Test me with "i / put jewels in machine / i / put gunpowder in machine / i / put beans in machine".
```

In this example we have only defined a single set of transmutable objects, but we could easily expand to include other groups.

(Thanks to Jesse McGrew for proposing this example.)

228 ★★★ **Example Otranto** *RB*

A kind of rope which can be tied to objects and used to anchor the player or drag items from room to room.

The range of things one might want to do with a rope in a work of interactive fiction is fairly overwhelming. One might, in theory, swing from ropes; use them to tie containers shut; cut them up into smaller ropes; tie them together into longer ropes; employ them as fuses; bind other characters with them, or the player character.

Our rope implementation is, by these lights, reasonably simple, but it does account for the possibility of tying and untying both ends; using ropes to descend into lower rooms; pulling objects tied to the far end of the rope; and dragging objects from place to place.

```
"Otranto"
```

We start by coming up with a rope.

```
A rope is a kind of thing.
```

```
Definition: a thing is nonrope if it is not a rope. [The perfect idiocy of this statement notwithstanding, having a shortcut will come in very handy later]

Attachment relates things to each other in groups. The verb to be stuck to means the attachment relation.

Definition: a thing is tied if the number of things stuck to it is greater than 1.

Definition: a thing is free if it is not tied.

Definition: a rope is free if the number of nonrope things stuck to it is less than 2.

Definition: a thing is hindering if it is stuck to the noun and it is not within the location.

A thing can be round or unevenly shaped. A thing is usually round.

Definition: something is anchored if it is fixed in place or it is scenery or it is part of an anchored thing.

Definition: something is draggable if it is not had by the player and it is not the player and it is not anchored.
```

Now, we want a rope to be described in terms of the way it is tied, when it's described in a room description.

```
Rule for writing a paragraph about a rope (called the coil):
    if the coil is stuck to something which is in a room (called the next room) which is not the location:
        let the way be the best route from the location to the next room;
        if the way is up or the way is down:
            say "[The coil] runs [way] into [the next room].";
        otherwise:
            say "[The coil] snakes across the floor [way] towards [the next room].";
    otherwise:
        say "There is [a coil] here[if the coil is stuck to a visible nonrope thing], tied to [the list of nonrope visible things which are stuck to the coil][end if]."

To decide what room is the home of (item - a thing):
    if item is a door:
        let front cut be the number of moves from the location to the front side of the item;
        let back cut be the number of moves from the location to the back side of the item;
        if front cut is -1, let front cut be 999;
        if back cut is -1, let back cut be 999;
        if the location encloses the item, decide on the location;
        if front cut is greater than back cut, decide on the back side of the item;
        decide on the front side of the item;
    decide on the location of the item.

Rule for writing a paragraph about a nonrope thing (called the anchor) which is stuck to a rope (called the coil):
    if the coil is in an adjacent room:
        let the next room be the home of the coil;
        let the way be the best route from the location to the next room;
        if the way is up or the way is down:
            say "[The coil] runs [way] from [the anchor] into [the next room].";
        otherwise:
            say "From [the anchor] runs [a coil], heading off toward [the way].";
    otherwise:
        if the coil is stuck to something which is not visible,
            say "[The coil] is tied to [the anchor][if the coil is stuck to something in an adjacent room (called the next room)], and from there runs off towards [the next room][end if]."
```

We need a way to account for it when it's being carried, as well.

```
After printing the name of a rope (called the tied object) while taking inventory:
    if something nonrope is stuck to the tied object:
        say " (attached to [the list of nonrope things which are stuck to the tied object])";
    otherwise:
        say " (with both ends free)".
```

And, indeed, whenever the player examines a rope, we should see what's connected.

```
Instead of examining a rope (called the cord) when something is stuck to the cord:
    say "[The noun] is tied to [the list of secondary things which are stuck to the noun]."
```

Similarly, any time the player looks at something tied to a rope.

```
After examining the player when the player is stuck to something which is not the player:
    say "You're currently lashed to [the list of secondary things stuck to the noun]."

After examining something which is stuck to something secondary:
    say "[The noun] is currently attached to [the list of secondary things stuck to the noun]."
```

We also need to make sure that the rope can be interacted with properly even when it's partly in the next room.

```
After deciding the scope of the player:
    if something stuck to a rope (called the coil) is in the location, place the coil in scope.

A reaching inside rule:
    if the noun is a rope:
        let the anchor be a random visible thing stuck to the noun;
        if the anchor is touchable, allow access.
```

Now tying:

```
Before tying something to a rope:
    if the noun is stuck to the second noun, say "[The noun] and [the second noun] are already tied together." instead;
    if the second noun is not free, say "[The second noun] has no ends free."
instead;
    if the noun is round, say "You can't realistically tie anything to [the noun]."
instead.

Instead of tying a rope to something:
    try tying the second noun to the noun.

Instead of tying something to a rope:
    now the noun is stuck to the second noun;
    say "You loop [the second noun] around [the noun] and knot firmly."

Instead of tying something to a nonrope tied thing:
    let the coil be a random rope stuck to the second noun;
    try tying the noun to the coil.

Instead of tying a nonrope tied thing to something:
    let the coil be a random rope stuck to the noun;
    try tying the second noun to the coil.

Instead of tying a free nonrope thing to a free nonrope thing:
    if the player carries a free rope (called the coil):
        try tying the noun to the coil;
        if the noun is stuck to the coil and the coil is free:
            try tying the second noun to the coil;
    otherwise:
        say "You lack the requisite spare rope."

Understand "untie [something] from [something]" as untying it from. Understand
"untie [something]" as untying it from.

Rule for supplying a missing second noun while untying something from:
    if the number of secondary things stuck to the noun is 0, say "[The noun] is
already entirely free." instead;
    if the noun is a rope:
        if the number of touchable nonrope things which are stuck to the noun > 1:
            say "You'll have to say which thing you want to untie [the noun] from.";
            rule fails;
        otherwise:
            if the number of touchable nonrope things stuck to the noun is 0, say
"You can't reach [the random nonrope thing stuck to the noun]." instead;
            let the tied object be a random touchable nonrope thing which is stuck to
the noun;
            say "(from [the tied object])[line break]";
            now the second noun is the tied object;
    otherwise:
        if the noun is stuck to a rope (called the tied object):
            say "(from [the tied object])[line break]";
            now the second noun is the tied object.

Untying it from is an action applying to two things.

Before untying a rope from something: try untying the second noun from the
noun instead.
```

```
Before untying something from a rope:
    if the second noun is not held:
        say "(first picking up [the second noun])[line break]";
        try taking the second noun.

Check untying it from:
    unless the noun is stuck to the second noun or the second noun is stuck to the noun,
        say "[The noun] and [the second noun] are already not tied together." instead.

Carry out untying it from:
    now the noun is not stuck to the second noun.

Report untying it from:
    say "Untied."
```

Another part of the fun of a rope is that you can drag things from another room:

```
After reading a command: now every thing is unmentioned.

Before pulling something anchored: say "[The noun] is firmly anchored." instead.

Instead of pulling something tied:
    if the noun is unmentioned:
        say "The impulse is transmitted to [the list of pullable things stuck to the noun].";
        repeat with item running through pullable things stuck to the noun:
            say "[item]: [run paragraph on]";
            try pulling the item;
        if the noun is a rope and the noun is not within the location:
            if the number of nonrope hindering things is 0, move the noun to the location;
    otherwise:
        continue the action.

Before pulling something which is not visible:
    if the noun is anchored:
        say "[The noun] resists, for whatever reason." instead;
    otherwise:
        let space be the holder of the noun;
        let way be the best route from the space to the location;
        if the way is a direction:
            move the noun to the location;
            say "[The noun] [if the way is up]rises[otherwise]slides[end if] into view." instead;
        otherwise:
            move the noun to the location;
            say "[The noun] slides into view." instead.

Definition: a thing is secondary if it is not the noun. Definition: a thing is pullable if it is not the noun and it is not the player.
```

A player who is tied to things should also have some restrictions on his ability to move.

```
Before going a direction (called the way) when the player has something (called
the link) which is stuck to something anchored (called the anchor):
    let the next room be the home of the anchor;
    if the next room is not a room, continue the action;
    if the next room is the location:
        if the link is stuck to at least two anchored things,
            say "You can't go far while you're carrying [the link] tied to [the list of
anchored things stuck to the link]." instead;
    otherwise:
        let the safe way be the best route from the location to the next room;
        if the safe way is the way:
            if the player is not stuck to the anchor, say "(coiling up your rope again as
you go...)";
        otherwise:
            if the safe way is a direction,
                say "While you have [the link] you can't really head any direction but
[best route from the location to the next room]." instead;
            otherwise say "You're tied up here." instead.

Before going a direction (called the way) when the player is stuck to something
anchored (called the anchor):
    let the next room be the home of the anchor;
    if the next room is not a room, continue the action;
    if the next room is the location:
        if the player is stuck to at least two anchored things,
            say "You can't go far while you're tied to [the list of anchored things stuck
to the player]." instead;
    otherwise:
        if the best route from the location to the next room is the way:
            say "(coiling up your rope again as you go...)";
        otherwise:
            say "Your attachments prevent you going any way but [best route from
the location to the next room]." instead.
```

Sometimes, if the player is tied to a movable object, the moved object will move with him.

```
After going somewhere when the player has something (called the link) which is
stuck to something draggable:
    if the player is not stuck to the link:
        say "You drag along behind you [the list of draggable things which are stuck
to the link].";
        now every draggable thing which is stuck to the link is in the location;
    continue the action.

Report going somewhere when the player is stuck to something draggable:
    say "You drag along behind you [the list of draggable things which are stuck to
the player].";
    now every draggable thing which is stuck to the player is in the location.
```

And now the actual game and puzzles.

```
Use full-length room descriptions.

The Fallow Field is a room. "The very land is gloomy, the earth plowed into
untended rows that yield no fruit, shadowed by the castle to the north. A chasm,
no doubt the product of some upheaval of the earth, opens before your feet.". An oak stump is fixed in place in the Field. "From an oak stump, a few hopeful shoots grow." A hempen rope is a rope in the field. It is stuck to the oak stump and the wooden chest. The stump is unevenly shaped.

The Chasm is below the Field. "Your person is most uncomfortably pressed on every side by the closeness of the walls; to which you may add as a further inconvenience, that the irregularity of the floor making it difficult to walk upright." An iron key is in the Chasm. "An iron key nestles in the cleft of earth, its age indicated by its implausibly great size."

The wooden chest is a unevenly shaped closed openable container in the Chasm. The description of the wooden chest is "A handsome, solid case not long committed to its dank enclosure, or it would long since have rotted." Rule for printing the name of the wooden chest when the chest is not handled: say "deadweight". Understand "dead" or "weight" or "deadweight" as the chest. Before pulling the wooden chest: now the chest is handled.

In the chest is a heavy dagger. The description of the dagger is "Set with red jewels and of a wicked aspect."

Before going down from the Field when the player is not stuck to something anchored:
    say "You don't quite dare simply leap into the darkness without some anchor." instead.

Before going down from the Field:
    let anchor be a random anchored thing which is stuck to the player;
    say "You lower yourself gingerly, hoping that [the anchor] holds your weight..."

Before going up from the Chasm:
    if the player cannot touch a rope which is stuck to an anchored thing which is in the Field, say "And how, precisely, do you mean to do that?" instead.

The Castle Hall is north of the Field. "All is desolate: the great hall has no roof, nor is there any glass in the windows. A staircase without banister ascends inside the wall to a musician's gallery without song."

The Musician's Gallery is above the Castle Hall. "Of its former cheery aspect only this remains to the Gallery: that chevrons of red and yellow are painted on the wall. But as these are streaked with rain and grime, the banister pulled away, the roof open to the sky, and the corners made a nesting place for birds, the consolation thereby afforded is but slight."

The pointed door is north of the Musician's Gallery and south of the Sinister Attic. It is a closed locked openable door. "A pointed door of particularly grim and uninviting aspect leads north." The pointed door is lockable and unevenly shaped. The description of the pointed door is "A door coming to a gothic point and fitted with iron fittings of great strength. The handle looks particularly well-attached." The iron key unlocks the pointed door.

A rule for reaching inside the Musician's Gallery:
    allow access.
```

```
A rule for reaching inside the Sinister Attic:
    allow access.

Instead of opening the pointed door for the first time:
    say "When you rattle at the door, there arises from beyond a terrible shrill
noise as though something beyond exults in its imminent release."

After opening the trapped pointed door when the player can see the pointed
door:
    say "Thousands of bats fly from the pointed door, attacking you!";
    end the story.

After opening the pointed door when the player cannot see the pointed door:
now the pointed door is untrapped; continue the action.

The pointed door can be trapped or untrapped. The pointed door is trapped.

Before pulling the pointed door: try opening the pointed door instead.

The player is unevenly shaped.

After going to the sinister attic:
    say "You have arrived at the goal of your quest!";
    end the story finally.

Test me with "x rope / pull rope / get chest / untie rope from chest / tie rope to me
/ down / get key / up / untie rope from stump / north / up / unlock pointed door
with key / open it / tie rope to door / down / pull rope / up / north".

Test death with "x rope / pull rope / get chest / untie rope from chest / tie rope to
me / down / get key / up / untie rope from stump / north / up / unlock pointed
door with key / open it / g"
```

229

Example **Unthinkable Alliances**

RB

People are to be grouped into alliances. To kiss someone is to join his or her faction, which may make a grand alliance; to strike them is to give notice of quitting, and to become a lone wolf.

The following is best tested by experimentally kissing and/or attacking, and typing RELATIONS after every change to see the effect.

```
"Unthinkable Alliances"

Unthinkable Solutions is a room. Sophie, Daisy, Ryan and Owen are in
Unthinkable Solutions.

Alliance relates people to each other in groups. The verb to help means the
alliance relation.

Sophie helps Ryan. Daisy helps Ryan. Owen helps the player.
```

```
Instead of kissing someone (called the blessed one):
    say "Smack!";
    now the player helps the blessed one.

Instead of attacking someone (called the vilified one):
    say "Smack!";
    now the player does not help the vilified one.

Test me with "relations / kiss sophie / relations / hit ryan / relations".
```

230 ★★★ **Example** **The Unexamined Life** *RB*

An adaptive hint system that tracks what the player needs to have seen or to possess in order to solve a given puzzle, and doles out suggestions accordingly. Handles changes in the game state with remarkable flexibility, and allows the player to decide how explicit a nudge he wants at any given moment.

Hint systems in IF come in a variety of flavors: some are a static, prewritten set of guidelines (which might exist in a menu or outside the game entirely); others are built in as part of the program, and attempt to adapt to the situation the player currently faces. Adaptive hints have the advantage that they are less likely to reveal information for which the player is not ready, and the disadvantage that they are more work for the author.

The exercise here is to write an adaptive hint system that will both respond in agile ways to the state of the world model and require a minimum of authorial fussing. We also want the player to be able to ask for a hint about any object he encounters in the game world: this will let him be specific and avoid accidentally receiving hints about the wrong puzzles.

Our baseline assumption is that a player may find a puzzle unsolvable for one of two reasons: he either hasn't seen the relevant clue, or he hasn't got the relevant equipment. If these are true, then he should be given hints about how to find this information, and then once he has it, more specific hints about the puzzle itself -- ending, as a last resort, with the exact command(s) he will need to use in order to bring about the solution.

In practice, there are other possibilities, but this will do for an example.

We begin by defining our relations:

```
"The Unexamined Life"

Use scoring.

Explaining relates one thing to various things. The verb to explain means the
explaining relation.
```

```
Instead of hinting about something when something unexamined (called the clue) explains the noun:
    say "You're still missing some information that might be useful to understanding the problem. [More]";
    if player consents, try hinting about the clue.

Requiring relates one thing to various things. The verb to require means the requiring relation.

Instead of hinting about something when the noun requires something (called the implement) which is not carried by the player:
    say "You're missing an object that might be useful to resolving this problem. [More]";
    if player consents, try hinting about the implement.

Hinting about is an action applying to one visible thing. Understand "hint about [any thing]" as hinting about.
```

This allows us to create the most absolutely generic sort of hint -- boring, perhaps, but in practice the player often just needs a nudge about what part of the game world he should be examining for a solution:

```
Carry out hinting about:
    if something explains the noun, say "You might want to review [the list of things which explain the noun]. ";
    if the noun requires something:
        say "You should be sure that you have [the list of things required by the noun]. ";
    otherwise:
        say "Sorry, I can't advise you further on that.".
```

These things cover hinting about objects that are themselves puzzles. But what if the player asks for hints about a tool or piece of information because he doesn't know how to apply it yet? We might want to give some guidance there, as well.

```
Carry out hinting about something which explains something (called target):
    if target is unseen, say "[The noun] might prove useful information, sooner or later." instead;
    otherwise say "You could examine [the noun]." instead.

Carry out hinting about something which is required by something:
    say "[The noun] might be useful to have. [More]";
    if player consents:
        if a seen thing requires the noun, say "[The noun] may help with [the list of seen things which require the noun]." instead;
        otherwise say "There are [number of things which require the noun in words] problems for which [the noun] might come in handy." instead.
```

Now we have these general hints written, but we want to pre-empt them if the player has not yet fulfilled all the prerequisites.

```
Instead of hinting about something unseen:
    if the noun is visible:
        now the noun is seen;
        continue the action;
    say "Perhaps you should explore further. ";
    if the ultimate location of the noun is an unvisited room:
        try hinting about the ultimate location of the noun;
    otherwise:
        if the ultimate location of the noun is the location:
            say "You're in the correct room right now[if the visible shell of the noun is a thing]. Try further exploring [the visible shell of the noun][end if].";
        otherwise:
            try hinting about the ultimate location of the noun.

Instead of hinting about a visited room:
    say "There's a room you've visited, but you haven't exhausted all there is to see there. [More]";
    if player consents:
        say "Try going back to [the noun]. [More]";
        if player consents, direct player to the noun.

Instead of hinting about an unvisited room:
    say "There's a room you haven't yet visited. [More]";
    if player consents, direct player to the noun.

To direct player to (goal - a room):
    let way be the best route from location to the goal, using even locked doors;
    if way is a direction, say "Try going [way] to start your explorations.";
    otherwise say "Sorry, the route is an indirect one.".

Instead of hinting about a portable seen thing which is not visible:
    if the noun is scenery, continue the action;
    say "You have seen the item you need to solve this problem, but it's not in sight at the moment. [More]";
    if player consents:
        try hinting about the ultimate location of the noun.
```

And this business of "seen" things requires, of course, that we keep track:

```
A thing can be seen or unseen. A thing is usually unseen. The player is seen.
After printing the name of something (called target): now the target is seen.
```

That "After printing..." rule means that as soon as the game automatically prints the name of an object, it tags that object as having been "seen" by the player. This requires just a little care on our part, that we never mention an object without using the game's printing rules. Still, it is much easier than most other possible forms of bookkeeping.

We also need to deal with the question of whether the player has examined an object, for those objects whose descriptions carry vital information:

```
A thing can be examined or unexamined. A thing is usually unexamined. Carry out examining something: now the noun is examined.
```

In practice, there might be other ways of getting vital facts, and in a more sophisticated puzzle game we might need a more sophisticated model to track this. But examined or unexamined will do for now.

```
To decide what room is the ultimate location of (item - a thing):
    let place be the holder of the item;
    while the place is a thing:
        let the place be the holder of the place;
    if the place is a room, decide on the place.

To decide what thing is the visible shell of (item - a thing):
    if item is visible, decide on the item;
    let place be the holder of the item;
    while place is a thing and place is not visible:
        let place be the holder of the place;
    if the place is visible, decide on the place.

To say more:
    say "[paragraph break]Shall I go on? > ".
```

That covers most of the generic hints, but let's also add some slightly more precise hints about a few kinds of objects that are especially important in the model world. These hints will probably not be very interesting to a seasoned IF veteran, but a novice player who does not know the wording or cannot guess what something might be for may still find them useful:

```
Carry out hinting about a locked lockable thing:
    say "You could unlock [the noun] with [the matching key of the noun]." instead.

Instead of hinting about a locked thing when the matching key of the noun is not
carried by the player:
    if the player can see the matching key of the noun:
        say "Perhaps [the matching key of the noun] would help.";
    otherwise:
        say "[The noun] is locked. There must be a key around somewhere.
[More]";
        if player consents, try hinting about the matching key of the noun.

Carry out hinting about a closed openable unlocked thing:
    say "You could open [the noun]." instead.

Carry out hinting about an open door:
    say "You could enter [the noun]." instead.

Carry out hinting about an unexamined thing:
    say "You might find out something if you examine [the noun]." instead.

Carry out hinting about an edible thing:
    say "You could eat [the noun]." instead.

Carry out hinting about a wearable thing:
    say "You could wear [the noun]." instead.

Carry out hinting about a pushable between rooms thing:
    say "You could push [the noun] some direction." instead.
```

Now to the actual objects in the game:

```
The Crypt is a room. "This squat, barrel-vaulted chamber runs roughly north-
south. Along either side are the graves of Saxon kings and early bishops of the
church long since gone to dust -- one [tomb] in particular looks undisturbed."
```

Notice that we used the bracketed tomb here: the tomb is scenery, and if we do not use the name-printing function, Inform will not register that we have mentioned it to the player.

```
The tomb is scenery in the Crypt. The tomb is openable and closed. The silver
dagger is a thing in the tomb. Understand "tombs" as the tomb. The description
of the silver dagger is "Gleaming in a soft light all its own. Its blade is figured with
running deer and its hilt is made of horn." The wight requires the silver dagger.
The tomb requires the pry bar.

Instead of opening the tomb when the player does not carry the pry bar:
    say "The lids are stone, too heavy for you to raise without some implement."
```

Now we can add specific hints to replace the generic ones:

```
Carry out hinting about the tomb:
    say "The lids are heavy, but you can open them when you carry the pry bar."
```

The rest of the hint system ensures that the player will not see this final suggestion until he has the pry bar, since the tomb "requires" the pry bar. Having the hint there doesn't excuse us from providing some alternate wording in case the player solves this not-very-difficult conundrum on his own, though:

```
Understand "pry [something] with [something preferably held]" as unlocking it
with. Understand the commands "lever" or "prise" as "pry".

Instead of unlocking something with the pry bar, try opening the noun.

The wight is a man in the Crypt. "[The wight] lurks near the south exit." The
description of wight is "Old English [italic type]wiht[roman type]: a thing, a
creature. It is little more than the memory of a life ill-lived, but it lingers here."
Understand "wiht" or "creature" or "ghost" as the wight.

Instead of going south in the presence of wight:
    say "The wight breathes chill into your face.

Your head swims, and you are aware that you no longer have the willpower to go
in that direction."

Fresh Air is south from the Crypt.

After going to Fresh Air:
    increment the score;
    say "Congratulations, you have escaped!";
    end the story finally.

The inscription is fixed in place in the Crypt. "Someone has painstakingly carved
[an inscription] into the wall above the door." The description is "Squinting, you
```

```
decipher the Latin text: [italic type]Silver causes harm to those that live though dead[roman type]." The inscription explains wight.

The Treasure Chamber is north of the Crypt. "The walls are thick, the high windows promisingly barred with iron. But for all this there is no hint of any valuable stores remaining."

The pry bar is in the Treasure Chamber. "One of the window bars, rusted from its place, lies in a puddle of water." Understand "window" or "bars" as the pry bar. The description of the pry bar is "A few feet long, and not entirely rusted into uselessness yet."

Instead of giving the dagger to wight:
    say "The wight recoils, appalled."

Carry out hinting about wight:
    say "You will have to find some way to get wight to come in physical contact with the silver dagger, which he will certainly not do willingly. [More]";
    if player consents, say "You could, for instance, throw it at him." instead;
    otherwise stop the action.

Understand "touch [something] with [something]" as putting it on (with nouns reversed). Understand "hit [someone] with [something]" as putting it on (with nouns reversed).

Instead of attacking the wight:
    say "You can't force yourself to approach close enough for hand to hand combat: if, indeed, the wight has hands."

Instead of putting the dagger on wight:
    say "The wight fades out of your way without ever coming into contact with the dagger. Perhaps a more projectile method would work better."

Instead of putting something on wight:
    say "The wight dodges you."

Instead of throwing the dagger at wight:
    now the wight is nowhere;
    move the dagger to the location;
    increment the score;
    say "The dagger passes through its airy form with a rending like the rip of silk. The fragments dissipate at once."

The maximum score is 2.

Test me with "hint about wight / north / get bar / south / open tomb / get dagger / south / hint about wight / read inscription / hint about wight / attack wight / throw dagger at wight / south".
```

Note that, if using TEST ME to run through the solution on the Z-machine, we will have to answer a few yes/no questions along the way.

For Glulx, the code should instead read something like

```
Test me with "hint about wight / y / north / get bar / south / open tomb / get dagger / south / hint about wight / y / read inscription / hint about wight / y / attack wight / throw dagger at wight / south".
```

231 ★ **Example The Abolition of Love** RB

A thorough exploration of all the kinds of relations established so far, with the syntax to set and unset them.

Suppose we are modeling a complex society seething with interpersonal relations of every kind.

```
"The Abolition of Love"

Section 1 - Relation types

Loving relates one person to one person.

Noticing relates various people to one person.

Impressing relates one person to various people.

Fancying relates various people to various people.

Acquaintance relates people to each other.

Marriage relates one person to another.

Alliance relates people to each other in groups.

The Chapel is a room. Elizabeth, Wickham and Darcy are people in the Chapel. Mr Bennett and Mrs Bennett are people in the Chapel. Georgiana is a person in the Chapel.

The verb to love means the loving relation.

The verb to notice means the noticing relation.

The verb to impress means the impressing relation.

The verb to fancy means the fancying relation.

The verb to know means the acquaintance relation.

The verb to be married to means the marriage relation.

The verb to be related to means the alliance relation.

Elizabeth loves Darcy. Elizabeth fancies Darcy. Elizabeth notices Darcy. Elizabeth impresses Darcy.
```

```
Mr Bennett is related to Mrs Bennett and Elizabeth. Mr Bennett is married to Mrs Bennett.

Georgiana is related to Darcy.
```

Now we want ways to set and unset all of these relations. (In the interests of thoroughness, we may get a bit far-fetched here. It is not recommended in practice that we make the player guess the verb "traduce".)

```
Section 2 - Setting and Unsetting Love (1-1)

Understand "infatuate [someone] with [someone]" as infatuating it with.
Infatuating it with is an action applying to two visible things.

Carry out infatuating it with:
   now the noun loves the second noun.

Report infatuating it with:
   say "Now [the noun] loves [a random person loved by the noun][if the second noun loves someone], while [the second noun] loves [a random person loved by the second noun][end if]."

Understand "embitter [someone] toward [someone]" as embittering it toward.
Embittering it toward is an action applying to two visible things.

Carry out embittering it toward:
   now the noun does not love the second noun.

Report embittering it toward:
   say "[The noun] sees [the second noun] in a different light and no longer feels any affection."
```

Because love is a 1-1 relation, a person cannot love more than one other character at a time. Whenever we set a character to love a new person, that person ceases to love the character loved before. It is a fickle world.

One to various relations are a bit more open: we can say someone impresses multiple other characters, and our additions to the list do not override the initial ones.

```
Section 3 - Setting and Unsetting Impressed (1-V)

Understand "commend [someone] to [someone]" as commending it to.
Commending it to is an action applying to two visible things.

Carry out commending it to:
   now the noun impresses the second noun.

Report commending it to:
   say "[The second noun] takes a very decided interest in [the noun]."

Understand "traduce [someone] to [someone]" as traducing it to. Traducing it to is an action applying to two visible things.
```

```
Carry out traducing it to:
    now the noun does not impress the second noun.

Report traducing it to:
    say "[The second noun], hearing your story, decides not to be at all impressed
with [the noun]."
```

And because this is a one-to-various relation, we can also make statements which set multiple relations at once, so:

```
Understand "celebrate [someone]" as celebrating. Celebrating is an action
applying to one visible thing.

Carry out celebrating:
    now the noun impresses every person.

Report celebrating:
    say "[The list of people who are impressed by the noun] take a very decided
interest in [the noun]."

Understand "slander [someone]" as slandering to. Slandering to is an action
applying to one visible thing.

Carry out slandering to:
    now every person is not impressed by the noun.

Report slandering to:
    say "Now [the noun] impresses [the list of people who are impressed by the
noun]."
```

Note that the above unsetting is not equivalent to "now the noun does not impress every person" -- which would be ambiguous in spoken English, as well. Various-to-one relations are similar:

```
Section 4 - Setting and Unsetting Noticing (V-1)

Understand "draw the attention of [someone] to [someone]" as drawing the
attention of it to. Drawing the attention of it to is an action applying to two visible
things.

Carry out drawing the attention of it to:
    now the noun notices the second noun.

Report drawing the attention of it to:
    say "[The noun] glances thoughtfully in the direction of [the second noun]."

Understand "distract [someone] from [someone]" as distracting it from.
Distracting it from is an action applying to two visible things.

Carry out distracting it from:
    now the noun does not notice the second noun.
```

```
Report distracting it from:
    say "You distract [the noun] from [the second noun]."

Understand "draw attention to [someone]" as drawing attention to. Drawing attention to is an action applying to one visible thing.

Carry out drawing attention to:
    now every person notices the noun.

Report drawing attention to:
    say "You quickly cause everyone to attend to [the noun]."

Understand "outshine [someone]" as outshining. Outshining is an action applying to one visible thing.

Carry out outshining:
    now every person does not notice the noun.

Report outshining:
    say "You quickly distract everyone from [the noun]."

Section 5 - Setting and Unsetting Fancying (V-V)

Understand "flatter [someone]" as flattering. Flattering is an action applying to one thing.

Carry out flattering:
    now every person fancies the noun.

Report flattering:
    say "You draw down universal admiration for [the noun] by casting him or her in a flattering light."

Understand "unflatter [someone]" as unflattering. [Okay, okay, but it's four am.] Unflattering is an action applying to one thing.

Carry out unflattering:
    now every person does not fancy the noun.

Report unflattering:
    say "[The noun] gives everyone a universal disgust."

Understand "admire [someone]" as admiring. Admiring is an action applying to one thing.

Carry out admiring: now the player fancies the noun.

Report admiring: say "You find you rather fancy [the noun]."

Understand "loathe [someone]" as loathing. Loathing is an action applying to one thing.

Carry out loathing: now the player does not fancy the noun.
```

```
Report loathing: say "You take [the noun] in disgust."

Understand "cause chaos" as causing chaos. Causing chaos is an action applying to nothing.

Carry out causing chaos:
	now every person fancies every person.

Report causing chaos: say "Now everyone fancies everyone else, which is quite an inconvenient state of affairs."

Understand "relieve chaos" as relieving chaos. Relieving chaos is an action applying to nothing.

Carry out relieving chaos:
	now every person is fancied by no one.

Report relieving chaos: say "Now no one fancies anyone, which is safe but tedious."
```

Our options for setting and unsetting symmetrical relations are more limited again:

```
Section 6 - Setting and Unsetting Marriage (1-1 Symmetrical)

Understand "marry [someone] to [someone]" as uniting it in matrimony with. Uniting it in matrimony with is an action applying to two visible things.

Carry out uniting it in matrimony with:
	now the noun is married to the second noun.

Report uniting it in matrimony with:
	say "You perform the marriage of [the noun] to [the second noun], joining them to the family of [a list of people related to the noun]."

Understand "divorce [someone] from [someone]" as divorcing it from. Divorcing it from is an action applying to two visible things.

Check divorcing it from:
	if the noun is not married to the second noun, say "[The noun] is not married to [the second noun] anyway." instead.

Carry out divorcing it from:
	now the noun is not married to the second noun.

Report divorcing it from:
	say "[The noun] is now not married to [the second noun]."
```

When we unset the symmetrical relation on one side, it is automatically set or unset on the other. It is not necessary to say "the second noun is married to the noun" or "the second noun is not married to the noun", even though that is the case.

```
Section 7 - Setting and Unsetting Acquaintance (V-V Symmetrical)
```

```
Understand "introduce [someone] to [someone]" as introducing it to. Introducing it to is an action applying to two visible things.

Carry out introducing it to:
	now the noun knows the second noun.

Report introducing it to:
	say "You introduce [the noun] to [the second noun]. Now [the noun] is acquainted with [the list of people who are known by the noun], and [the second noun] is acquainted with [the list of people who are known by the second noun]."

Understand "announce [someone]" as announcing. Announcing is an action applying to one visible thing.

Carry out announcing:
	now every person knows the noun.

Report announcing:
	say "You announce [the noun] to the whole assembled company."

Understand "ostracise [someone]" as ostracising. Ostracising is an action applying to one visible thing.

Carry out ostracising:
	now every person does not know the noun.

Report ostracising:
	say "You cause everyone present to forget and pretend not to be acquainted at all with [the noun]."
```

And finally, setting groups:

```
Section 8 - Setting and Unsetting Familial Relations (Groups)

Understand "make [someone] adopt [someone]" as forcing it to adopt. Forcing it to adopt is an action applying to two visible things.

Carry out forcing it to adopt:
	now the noun is related to the second noun.

Report forcing it to adopt:
	say "Now [the second noun] is related to [the list of people related to the second noun]."

Understand "make [someone] disown [someone]" as forcing it to disown. Forcing it to disown is an action applying to two visible things.

Carry out forcing it to disown:
	now the second noun is not related to the noun.

Report forcing it to disown:
	say "Now [the second noun] is related to [the list of people who are related to the second noun], and [the noun] is related to [the list of people who are related to the noun]."
```

Notice that when we say "the second noun is not related", we remove that person from the group: they are now in a separate group of their own, while the rest of the group's members remain related to one another.

And finally, a long litany of test cases, complete with the relations lists:

```
Test acquaintance with "relations / introduce darcy to elizabeth / introduce darcy to wickham / announce mr bennett / relations / ostracise wickham / introduce georgiana to wickham / relations".

Test impression with "commend georgiana to elizabeth / relations / celebrate Mrs bennett / relations / traduce mrs bennett to darcy / relations / slander mrs bennett / relations".

Test notice with "draw the attention of darcy to elizabeth / relations / draw attention to mr bennett / relations / distract darcy from mr bennett / relations / outshine mr bennett / relations".

Test love with "embitter elizabeth toward darcy / relations / infatuate elizabeth with wickham / relations".

Test marriage with "marry elizabeth to darcy / relations / divorce elizabeth from darcy / relations".

Test alliance with "make mr bennett adopt georgiana / relations / make mrs bennett disown georgiana / relations".

Test fancying with "admire elizabeth / relations / loathe elizabeth / relations / flatter elizabeth / relations / unflatter elizabeth / relations / cause chaos / relations / relieve chaos / relations".

Test me with "test acquaintance / test impression / test notice / test love / test alliance / test fancying / test marriage".
```

232 ★ **Example Swerve left? Swerve right? Or think about it and die?** *RB*

Building a marble chute track in which a dropped marble will automatically roll downhill.

Suppose we have marbles that roll downhill across our map, in a life-size version of one of those marble-chute toys. We might now want to keep track of both compass relationships and which-room-slopes-into-which, so we make a new relation:

```
"Swerve left? Swerve right? Or think about it and die?"

Overlooking relates various rooms to various rooms.

The verb to overlook means the overlooking relation.

A thing can be spherical or lumpy. A marble is a kind of thing. A marble is always spherical. The player carries a marble called a red marble. The player carries a
```

```
marble called an agate marble. The player carries a marble called a blue cloudy marble.

The Long Yellow Slide is north of the Funnel. The Long Yellow Slide overlooks the Blue Funnel. The Ski-jump is below the Blue Funnel. The Blue Funnel overlooks the Ski-jump. The Ski-jump overlooks the Landing Bowl. The Landing Bowl overlooks the Snake Run. The Landing Bowl is north of the Snake Run. The Snake Run overlooks the Goal. The Snake Run is north of the Goal.

Definition: a room is sloping if it overlooks a room.
```

And let's say we want the player to be allowed to slide, too, since that would be much more fun than just watching the marbles go:

```
Understand "sit" as sitting down. Sitting down is an action applying to nothing. Check sitting down: if the player is spherical, say "You are already seated." Carry out sitting down: now the player is spherical. Report sitting down: say "You sit, ready to slide wherever fate takes you."

Understand the command "stand" as something new.

Understand "stand" or "stand up" as standing up. Standing up is an action applying to nothing. Check standing up: if the player is lumpy, say "You are already standing." Carry out standing up: now the player is lumpy. Report standing up: say "You get to your feet."
```

Now a rule to control what happens to all our sliding and rolling objects:

```
Every turn:
    repeat with item running through spherical things which are in sloping rooms:
        let the current space be the holder of the item;
        let the final space be a random room which is overlooked by the current space;
        if the player can see the item and the item is a marble, say "[The item] rolls out of the room toward [the final space].[line break]";
        if the player is the item, say "You keep sliding...";
        move the item to the final space;
        if the player can see the item and the item is a marble, say "[The item] rolls into the room from [the current space].[line break]".
```

Since the Ski-jump overlooks the Landing Bowl, the marble will be able to fly through the air to its destination, even though there is no map connection to allow the player to cross. We might want to let the player make it across this barrier also, so:

```
Instead of jumping in a sloping room:
    say "You leap...";
    move the player to a random room overlooked by the location.
```

Because overlooking is various-to-various, we could include that element popular in marble chute toys, the splitter:

```
The Downhill Splitter is north of the Long Yellow Slide. "The green plastic chute runs downhill towards a Y-junction, forcing incoming marbles right or left."
The Downhill Splitter overlooks the Long Yellow Slide and the Purple Snaking
Passage. The Purple Snaking Passage is southeast of the Downhill Splitter. The
Purple Snaking Passage overlooks the Landing Bowl. The Purple Snaking
Passage is above the Landing Bowl.

The player is in the Downhill Splitter.

Test me with "drop red / drop blue / sit / z / stand up / drop agate / sit / z / z / z / z
/ z".
```

233 ★ **Example** **Beneath the Surface** RB

An "underlying" relation which adds to the world model the idea of objects hidden under other objects.

The standard world model provides for the idea of containers and supporters, but this is not the only way that objects can relate to one another in the real world. Here we try adding the idea of concealment beneath another object:

```
"Beneath the Surface"

Section 1 - In Which our Terms are Defined

Underlying relates various things to one thing. The verb to underlie means the
underlying relation. The verb to be under means the underlying relation. The
verb to be beneath means the underlying relation.

Instead of looking under a thing which is underlaid by something (called the lost
object):
    say "You find [the list of things which underlie the noun]!";
    now every thing which underlies the noun is carried by the player;
    now every thing which underlies the noun does not underlie the noun.

Hiding it under is an action applying to one carried thing and one thing.
Understand "put [something preferably held] under [something]" as hiding it
under. Understand "hide [something preferably held] under [something]" as
hiding it under. Understand the commands "shove" and "conceal" and "stick" as
"hide".

Check hiding it under:
    if the second noun is not fixed in place, say "[The second noun] wouldn't be a
very effective place of concealment." instead.

Carry out hiding it under:
    now the noun is nowhere;
    now the noun underlies the second noun.

Report hiding it under:
    say "You shove [the noun] out of sight beneath [the second noun]."

Section 2 - In Which They are Put To Use
```

```
The Room of Hidden Objects is a room. It contains a sofa, an easy chair, and a rug. The sofa supports a lime-green pillow and an innocent-looking Chinese finger toy. The rug is fixed in place. The chair is a supporter.

A treasure map underlies the easy chair. A skeleton is beneath the sofa. A blueprint of Atlantis, a lexicon of Linear A, and the key to Jimmy Hoffa's Mausoleum are under the rug.

Test me with "look under the sofa / look under the rug / look under the easy chair / hide lexicon under rug".
```

234

Example **Bogart**

RB

Clothing for the player that layers, so that items cannot be taken off in the wrong order, and the player's inventory lists only the clothing that is currently visible.

We have two things to keep track of with our layering clothing: what currently is covering something else; and what *can* cover something else. This implementation goes for a fairly simple treatment, assuming that each item of clothing will completely conceal those beneath it, and that we are not implementing entire sets of shirts, jackets, etc. But it will do for a demonstration.

```
"Bogart"

Section 1 - Clothing Behavior
```

First we make our relation to represent what *is* underneath another item:

```
Underlying relates one thing to various things. The verb to underlie means the underlying relation. The verb to be under implies the underlying relation.
```

And now we prevent taking a lower layer off before the thing that is worn over it:

```
Before taking off something which underlies something (called the impediment) which is worn by the player:
say "(first removing [the impediment])[command clarification break]";
silently try taking off the impediment;
if the noun underlies something which is worn by the player, stop the action.

Check taking off:
   if the noun underlies something (called the impediment) which is worn by the player, say "[The impediment] [are] in the way." instead.

Carry out taking off:
   now the noun is not underlaid by anything.

Report taking off something:
   say "[We] [are] now wearing [a list of uppermost things worn by the player]." instead.
```

```
Definition: a thing is uppermost if it is not under something.
```

That covers order of clothing removal, but we also want to restrict what can be worn on top of what else. This time we need Inform to have some idea of what customarily can be layered on top of what other clothing:

```
Overlying relates one thing to various things. The verb to overlie means the overlying relation.

Covering relates a thing (called A) to a thing (called B) when the number of steps via the overlying relation from A to B is greater than 0. The verb to cover means the covering relation.
```

With these definitions, we can say that a jacket should go over a shirt and a shirt over an undershirt (say), and then Inform will know that a jacket will cover both shirt and undershirt.

```
Before wearing something when something (called the impediment) which covers the noun is worn by the player:
    while the player wears something which covers the noun:
        say "(first removing [the impediment])[command clarification break]";
        silently try taking off the impediment;
        if the player is wearing the impediment, stop the action.

Carry out wearing:
    if the noun covers something (called the hidden item) worn by the player, now the hidden item underlies the noun.

Instead of looking under something which is worn by the player:
    if something (called the underwear) underlies the noun, say "[We] [peek] at [the underwear]. Yup, still there.";
    otherwise say "Just [us] in there."

Instead of taking inventory:
    say "[if the player carries something][We]['re] carrying [a list of things carried by the player][else][We]['re] empty-handed[end if][if the player wears something]. [We] [are] wearing [a list of uppermost things worn by the player][end if]."

To peek is a verb.
```

Notice that our inventory only describes the things that the player can see as the upper layer of clothing.

```
Section 2 - The Scenario

The Trailer is a room. "A full-length mirror is the main amenity in here, and that suits you just fine." The full-length mirror is scenery in the Trailer. Instead of examining or searching the mirror, try taking inventory.

The player wears a fedora, a jacket, a shirt, some undershorts, an undershirt, some slacks, a pair of socks, and a pair of shoes.
The shirt underlies the jacket. The pair of socks underlies the pair of shoes. The undershorts underlie the slacks. The undershirt underlies the shirt.

The jacket overlies the shirt. The shoes overlie the socks. The slacks overlie the undershorts. The shirt overlies the undershirt.

Test me with "x mirror / remove fedora / remove jacket / remove shirt / remove slacks / remove undershirt / remove shoes / remove socks / remove shorts / remove undershorts".
```

If we further wanted to prevent the player from taking off clothes in inappropriate places, we might add something like this:

```
Instead of taking off something in the presence of someone who is not the player:
    say "[We] [are] far too modest to strip in public."
```

---

235 ★★★

Example The Problem of Edith

RB

A conversation in which the main character tries to build logical connections between what the player is saying now and what went immediately before.

Suppose that we have a core set of issues we want to be able to bring up with all the characters, and we want characters to draw intelligent connections between different conversation topics. We will need some model of how things relate to one another, so:

```
"The Problem of Edith"

Suggestion relates things to each other. The verb to suggest means the suggestion relation.

A subject is a kind of thing. The current subject is a thing that varies. greeting is a subject.

Understand "ask [someone] about [any subject]" as asking it about the subject.
Understand "tell [someone] about [any subject]" as asking it about the subject.

Asking it about the subject is an action applying to one thing and one visible thing.

Carry out asking it about the subject:
    say "'Hmm, [the second noun],' says [the noun]. ";
    relate the current subject with the second noun;
    now the current subject is the second noun.
```

And if we wanted to offer the player some hints about angles he could pursue:

```
Instead of thinking:
    say "You contemplate [a list of things suggested by the current subject]."
```

For that matter, we could use the same system to have characters make sense of any physical evidence the character shows them:

```
Instead of showing something which suggests the current subject to someone:
    say "[The second noun] nods impatiently."

Instead of showing something to someone:
    let the next subject be the next step via the suggestion relation from the noun
to the current subject;
    if the next subject is a subject:
        try asking the second noun about the subject the next subject;
    otherwise:
        say "[The second noun] shrugs."

When play begins:
    now the left hand status line is "Discussing: [current subject]";
    now the right hand status line is " ".

Broughton Hall is a room. Lady Uckfield is a woman in Broughton Hall. "Lady
Uckfield sits at her desk, looking wholly composed."

The nasty letter is a thing carried by the player. The nasty letter suggests
infidelity and penmanship. The ten-pound note is carried by the player. It
suggests money.

Infidelity is a subject. Infidelity suggests marriage and divorce. Marriage
suggests love. Marriage, love, and divorce are subjects.

Penmanship is a subject. Penmanship suggests education. Education is a
subject. Class status and money are subjects. Class status suggests education.
Money suggests class status and marriage.

The current subject is divorce.
```

Now we can define what gets said when the subject is changed, regardless of whether the segue was introduced in speech or by a shown object. Since rows are blanked after use, the speaker will never repeat herself; if we provide more than one line about the same pair of topics, the first one will be used, then the second, and so on, until the table runs out:

```
To relate (initial - a subject) with (next - a subject):
    repeat through Table of Remarks:
        if the initial is starting entry and the next is the final entry:
            say "[comment entry][paragraph break]";
            blank out the whole row;
            rule succeeds;
    say paragraph break.

Table of Remarks

    starting   final     comment
```

divorce	love	"'As it seems to me, all the love is on one side,' she says crisply. 'And that rarely works.'"
divorce	love	"'Stop making that plea: it won't work.'"
divorce	infidelity	"'Frankly, I rather think there would have been cause enough for divorce without the perversely plentiful evidence of unfaithfulness.'"
divorce	money	"'If you mean that the divorce will be expensive, I know it,' she says. 'But I can think of no happier investment.'"
marriage	money	"'If you wish me to understand that it was a marriage for money, you could have spared your energy. That was patent from the outset.'"
infidelity	money	"'I'm sorry, but I don't see how having married for money excuses a subsequent infidelity.'"

If we had more than one character in the scenario, we could provide multiple tables, but this will do to demonstrate the idea.

Of course, we can override specific instances, if we want the character always to say the same thing regardless of how we came to this point:

```
Instead of asking Lady Uckfield about the subject penmanship:
    now the current subject is penmanship;
    say "She sighs. 'So few people write really beautifully these days.'"

Test me with "think / ask lady about infidelity / show nasty letter to lady / show
note to lady / think / ask lady about divorce / ask lady about love / ask lady about
marriage / ask lady about divorce / ask lady about love / ask lady about
penmanship".
```

We would have to be careful about this system, since we have applied a various-to-various relation to every single object in the game. In practice it would probably be wisest to restrict it a bit, with judicious definitions of kind and so on.

236 ★ Example **Wainwright Acts** RB

A technical note about checking the location of door objects when characters other than the player are interacting with them.

Suppose we wanted to write rules for a character who will interact with doors in other locations even when the player is not present. This poses a little challenge: doors are actually single objects, and -- with the same shuffling of stage properties that applies to backdrops -- they are moved as needed to represent the door object in whatever room contains the player.

That means that it isn't safe to rely on a phrase like

```
if an open door is in the location of Bernard
```

because, even if Bernard's location is connected by doors to other places, the actual representation of that door may not be "in" Bernard's location, from the model's point of view, at this exact moment.

This does not, of course, mean that we can't ask this question; just that we have to be a little cleverer about how we phrase it. Every door has properties that correspond to the two locations

linked:

```
the front side of the blue door (a room, which is arbitrarily one side of the door)
the back side of the blue door (arbitrarily the other side)
```

We can make this information easier to check with a conditional relation, like so:

```
Liminality relates a door (called X) to a room (called Y) when the front side of X is Y or the back side of X is Y. The verb to be a threshold of means the liminality relation.
```

And this allows us to write rules that have characters interacting with doors even in the player's absence:

```
"Wainwright Acts"

The Waiting Room is a room. The waiting room door is west of the Waiting Room and east of the Gents' Loo. The Waiting Room door is an open door. "The waiting room door [if open]stands open[otherwise]is shut firmly[end if]."

Sir Humphrey is a man in the Gents' Loo.

Liminality relates a door (called X) to a room (called Y) when the front side of X is Y or the back side of X is Y. The verb to be a threshold of means the liminality relation.

Definition: a person is other if he is not the player.

Every turn:
    repeat with indiscreet one running through other people:
        repeat with port running through open doors that are a threshold of the location of the indiscreet one:
            if the port is a threshold of the location and the indiscreet one is not in the location:
                say "Through [the port], you overhear [the indiscreet one] discussing [one of]his hopes for your imminent resignation[or]your wife's infidelity[or]your financially straitened circumstances[or]ways to avoid attending your birthday party[or]your halitosis[as decreasingly likely outcomes]."

Test me with "z / z / z / w / z / e / close door / z".
```

237 ★★★ Example **A Humble Wayside Flower** *RB*

Relations track the relationships between one character and another. Whenever the player meets a relative of someone he already knows, he receives a brief introduction.

First we define the relationships we choose to acknowledge:

```
"A Humble Wayside Flower"

Marriage relates one person to another (called the spouse). The verb to be married to means the marriage relation.
```

```
Fatherhood relates one person (called father) to various people. The verb to engender means the fatherhood relation.
```

For brevity, we will ignore the existence of mothers. It is a sad world.

```
Siblinghood relates a person (called A) to a person (called B) when a person who engenders A engenders B. The verb to be sibling to means the siblinghood relation.

Family relates a person (called A) to a person (called B) when A is married to B or A engenders B or B engenders A or A is sibling to B. The verb to be related to means the family relation.

A person can be known or unknown. After printing the name of an unknown person (called the alien):
    if a known person (called the contact) is related to the alien:
        say " ([relation between alien and contact] of [the contact])";
        now the alien is known;
        rule succeeds.

To say relation between (first party - a person) and (second party - a person):
    if the first party is married to the second party:
        if the first party is female, say "wife";
        otherwise say "husband";
        rule succeeds;
    if the first party is sibling to the second party:
        if the first party is female, say "sister";
        otherwise say "brother";
        rule succeeds;
    if the first party engenders the second party:
        say "father";
        rule succeeds;
    if the second party is the father of the first party:
        if the first party is female, say "daughter";
        otherwise say "son";
        rule succeeds.

Pere Blanchard's Hut is a room. Percival Blakeney is a known man in the Hut. Marguerite is a woman in the Hut. Percival is married to Marguerite. Outside from the Hut is the Garden. Louise is a woman in the Garden. The Road to Paris is west of the Garden. Armand St Just is a man in the Road. Louise is married to Armand. Monsieur St Just is a man. He engenders Armand and Marguerite.

Test me with "out / west / east / west".
```

Monsieur St Just never appears on the scene in this piece, but if we did put him somewhere the player could find him, he, too, would be properly introduced.

238

Example **Meet Market**

A case in which relations give characters multiple values of the same kind.

RB

We have already seen that we can give things value properties -- a lamp has a brightness, for instance. Relations give us additional flexibility: since we may relate various things to various values, it is possible to describe a thing as having more than one value at the same time.

To demonstrate:

```
"Meet Market" by "K M and Eric Rossing"

Feature is a kind of value. The features are snub-nosed, gangly, comely, bright-eyed, and sulky.

Appearance relates various persons to various features. The verb to appear means the appearance relation.

Meet Market is a room.

Bob, Carol, Ted, and Alice are people in the Meet Market.

Bob appears snub-nosed and gangly.

Ted appears sulky and snub-nosed.

Carol appears comely and bright-eyed.

Alice appears bright-eyed and comely.

Yourself appears sulky and gangly.

Instead of looking:
	say "The snub-nosed ones: [list of people who appear snub-nosed][line break]";
	say "The gangly ones: [list of people who appear gangly][line break]";
	say "The comely ones: [list of people who appear comely][line break]";
	say "The bright-eyed ones: [list of people who appear bright-eyed][line break]";
	say "The sulky ones: [list of people who appear sulky][paragraph break]".

Test me with "look".
```

The same logic might be used to provide characters who have complex mood states: a person might be angry and sad, not merely one or the other -- feelings being what they are.

239 ★★★ Example **For Demonstration Purposes** *RB*

A character who learns new actions by watching the player performing them.

Suppose we want to have a character who can dynamically learn new actions by observing the player performing them. We could do this by adding the actions to a list of things the character can do, but using a relation to express the same idea allows for tidier, easier-to-read code.

Thanks to Jesse McGrew for the initial design of this example.

```
"For Demonstration Purposes"

Section 1 - Procedure

Capability relates various people to various stored actions. The verb to be
capable of means the capability relation.

Persuasion rule:
    let CA be the current action with no specific actor;
    if the person asked is capable of CA:
        persuasion succeeds;
    otherwise:
        say "[The person asked] look[s] confused. Maybe a demonstration would
help.";
        persuasion fails.

The action requester is an object that varies. The action requester variable
translates into I6 as "act_requester".

To decide which action is the current action with no specific actor:
    let old actor be the person asked;
    let old requester be the action requester;
    now the person asked is the player;
    now the action requester is nothing;
    let CA be the current action;
    now the person asked is the old actor;
    now the action requester is the old requester;
    decide on CA.

The learning by observation rule is listed after the report stage rule in the specific
action-processing rules.

Definition: a person is other if he is not the player.

This is the learning by observation rule:
    repeat with the viewer running through other people who can see the player:
        if the player is the actor and viewer is not capable of the current action:
            say "[The viewer] watches your behavior with interest. Seems like [they]
[are] learning.";
            now the viewer is capable of the current action.

Section 2 - Scenario

The Daily Planet is a room. Clark is here. He is a man.
```

```
When play begins:
    now Clark is capable of taking inventory.

Test me with "Clark, inventory / Clark, x me / x me / Clark, x me".
```

---

240 ★ **Example Number Study** RB

The parity and joint magnitude relations explored.

This is the same case given above, but expanded just slightly to demonstrate that the names of the relations can also be printed, if we like:

```
"Number Study"

Abstraction is a room.

Parity relates a number (called N) to a number (called M) when N minus M is even.

Joint magnitude relates a number (called N) to a number (called M) when N plus M is greater than 7.

To chart (R - a relation of numbers):
    repeat with N running from 1 to 5:
        repeat with M running from 1 to 5:
            if R relates N to M, say "[N] <=> [M] by [R][line break]";

When play begins:
    let L be { parity relation, joint magnitude relation };
    repeat with R running through L:
        chart R.
```

As this shows, we can even form lists of relations. The kind of L is "list of relations of numbers".

---

241 ★★★ **Example Murder on the Orient Express** RB

A number of sleuths (the player among them) find themselves aboard the Orient Express, where a murder has taken place, and one of them is apparently the culprit. Naturally they do not agree on whom, but there is physical evidence which may change their minds...

The following example creates two new relations, and two new verbs, in order to set up a tangled web of intrigue.

```
"Murder on the Orient Express"
```

```
The Dining Car is a room. Lord Peter is a man in the Dining Car. Sherlock Holmes is a man in the Dining Car. Miss Marple is a woman in the Dining Car. Adam Dalgliesh is a man in the Dining Car.

Suspecting relates various people to one person.

The verb to suspect means the suspecting relation.

Dalgliesh suspects Holmes. Holmes suspects Lord Peter. Lord Peter suspects Holmes. Miss Marple suspects the player.

Exculpating relates one thing to various people.

The verb to exculpate means the exculpating relation.

The silver bullet exculpates the player. The pipe ash exculpates Holmes. The poison pen letter exculpates Lord Peter. The poison pen letter exculpates Miss Marple. [Poor Dalgliesh. I guess he did it.]

The pipe ash, the letter and the silver bullet are carried.
```

Given this, we can then set up elaborate rules:

```
Instead of showing something to a person who suspects the player:
    say "'You would say that,' remarks [the second noun] darkly.".

Instead of showing something which exculpates the player to someone:
    say "'How striking!' says [the second noun]. 'Almost I begin to distrust myself.'".

Test me with "show the letter to miss marple / show the silver bullet to holmes".
```

And so on: "if Dalgliesh suspects someone who is exculpated by something carried by the player...", for instance, makes a fitting final example for this chapter. The description

```
someone who is exculpated by something carried by the player
```

expresses a complicated idea in very few words, and in such a way that a passer-by looking at the source text would immediately see what was meant.

The moral is that relations allow sophisticated patterns of behaviour to be created in a way that reads back naturally as English.

---

242 ★★★ **Example** **What Not To Wear** RB

A general-purpose clothing system that handles a variety of different clothing items layered in different combinations over different areas of the body.

```
"What Not To Wear"

Section 1 - Overlying and Underlying
```

We start by borrowing some of the same ideas from the Bogart example, but we're also going to make a kind called "garment-element". This kind will include both garments (objects of clothing) and body parts (things that can be covered by clothing); using it allows us to restrict the way our underlying and overlying relations apply, which will make them a bit faster at run-time.

```
A garment-element is a kind of thing.

Underlying relates various garment-elements to various garment-elements with fast route-finding. The verb to underlie means the underlying relation. The verb to be under implies the underlying relation.

Check taking off:
    if the noun underlies something (called the impediment) which is worn by the player, say "[The impediment] [are] in the way." instead.

Carry out taking off:
    now the noun is not underlaid by anything.

Report taking off something:
    say "[We] [are] now wearing [a list of uppermost things worn by the player]." instead.

Definition: a garment-element is uppermost if it is not under something opaque.
```

Here we've expanded on the previous ideas of 'uppermost' because it is possible for an upper layer to reveal what lies beneath: a tie, a clear plastic trenchcoat, an open-knit sweater, etc. We'll make such items transparent.

```
Before taking off something which underlies something which is worn by the player:
    while the noun underlies something (called the impediment) which is worn by the player:
        say "(first removing [the impediment])[command clarification break]";
        silently try taking off the impediment;
        if the noun underlies the impediment, stop the action.

Overlying relates various garment-elements to various garment-elements. The verb to overlie means the overlying relation.

Covering relates a garment-element (called A) to a garment-element (called B) when the number of steps via the overlying relation from A to B is greater than 0. The verb to cover means the covering relation.

Before wearing something when a garment which covers the noun is worn by the player:
    while the player wears a garment (called the impediment) which covers the noun:
        say "(first removing [the impediment])[command clarification break]";
        silently try taking off the impediment;
        if the player is wearing the impediment, stop the action.

Carry out wearing:
    repeat with hidden item running through things worn by the player:
        if the noun covers the hidden item, now the hidden item underlies the noun.

Instead of looking under something which is worn by the player:
    if something (called the underwear) underlies the noun, say "[We] [peek] at
[the underwear]. Yup, still there.";
    otherwise say "Just [us] in there."

Instead of taking inventory:
    say "[if the player carries something][We]['re] carrying [a list of things carried
by the player][else][We]['re] empty-handed[end if][if the player wears something].
[We] [are] wearing [a list of uppermost garments worn by the player][end if]."

To peek is a verb.

Section 2 - Regional Coverage
```

Here we draw in the idea that different clothes go over different areas of the body, and that they should be in competition with each other only if both sets of clothes belong at the same level over the same body area.

```
Before wearing something:
    let N be the layering depth of the noun;
    repeat with item running through things worn by the player:
        if the layering depth of the item is N and the item covers a body-part which
is covered by the noun:
            say "(first taking off [the item])[command clarification break]";
            silently try taking off the item;
            if the player wears the item, stop the action.
```

This may seem like overkill, but it allows us to create garments that cover different subsets of the body -- pants and shirt vs. a dress, for instance.

```
To decide what number is the layering depth of (chosen garment - a thing):
    let N be 0;
    if the chosen garment covers a body-part (called base):
        let N be the number of steps via the overlying relation from the chosen
garment to the base;
    decide on N.
```

To help with modeling, we'll give everyone body parts, broken down according to their relevance to clothing:

```
A body-part is a kind of garment-element. A torso, a seat, a head, pair of legs,
and pair of feet are kinds of body-part.
```

If we wanted to allow gloves, we might put in hands as well; but this is enough for now.

```
One head is part of every person. One torso is part of every person. One pair of
legs is part of every person. One pair of feet is part of every person. One seat is
part of every person.
```

And now we make some categories of clothing:

```
A garment is a kind of garment-element. A garment can be transparent. A pair of
pants, a pair of underpants, a foundation garment, a pair of socks, a pair of
shoes, a jacket, a hat, a dress, and a shirt are kinds of garment.

The plural of pair of pants is pairs of pants. The plural of pair of underpants is
pairs of underpants. The plural of pair of socks is pairs of socks. The plural of
pair of shoes is pairs of shoes.

A pair of pants, a pair of underpants, a foundation garment, a pair of socks, a
pair of shoes, a jacket, a hat, a dress, and a shirt are usually wearable.

When play begins:
    now every pair of socks overlies every pair of feet;
    now every pair of shoes overlies every pair of socks;
    now every pair of underpants overlies every seat;
    now every pair of pants overlies every pair of underpants;
    now every foundation garment overlies every torso;
    now every jacket overlies every shirt;
    now every jacket overlies every dress;
    now every hat overlies every head;
    now every dress overlies every pair of underpants;
    now every dress overlies every foundation garment.

Section 2 - The Scenario

The Dressing Room is a room.

The player carries some capris, some jeans, a corset, a plunge bra, a thong,
boy-shorts, black satin D'Orsay pumps, brown leather boots, a camisole, a
cocktail dress, a bolero, a cashmere shrug, a sheer wrap, and a linen tunic.

The woolly socks are a pair of socks.
The D'Orsay pumps and the brown leather boots are pairs of shoes.
The thong and the boy-shorts are pairs of underpants.
The capris and the jeans are pairs of pants.
The tunic is a shirt.
The camisole, the corset, and the plunge bra are foundation garments.
The cocktail dress is a dress.
The bolero, the cashmere shrug, and the sheer wrap are jackets. The shrug and
the wrap are transparent.

Test me with "wear capris / wear jeans / i / wear thong / i / wear dress / wear
corset / wear dress / i / wear wrap / i / wear boots / wear pumps / i".
```

243 ★★★★ **Example** **Mathematical view of relations** RB

Some notes on relations from a mathematical point of view, provided only to clarify some technicalities for those who are interested.

Inform uses the term "relation" in a broader sense than mathematics. Properly speaking, the term "relation" in its mathematical sense only applies to the case where the domain for the left and right objects are the same: for simplicity's sake, let us talk only about the case where they are.

In mathematics, the properties most often looked for in a relation are that it should be:

(a) Reflexive: A <=> A for every A. This is not especially useful for Inform, and seldom appears in practical examples.

(b) Symmetric: A <=> B if and only if B <=> A. Generally, Inform relations are not symmetric, but there are two important cases which are:

```
Meeting relates people to each other.
Marriage relates one person to another.
```

These are automatically symmetric, so that to assert one way round is to assert the other as well.

(c) Transitive: A <=> B and B <=> C means that A <=> C as well. Again, Inform relations are not generally transitive. In many relations, there can be long chains of things, each perhaps related to the one in front and the one behind, so that there is some indirect sense in which the two ends of the chain are connected to each other: but they are not related as such. For instance, a journey across the map might pass through ten rooms, each adjacent to the last and next, but the two ends would not themselves be adjacent. The concept we need is the "transitive closure" of the original relation, defined as the smallest transitive relation including the original. If R is a relation between "things", then the following:

```
TC relates a thing (called A) to a thing (called B) when the number of steps via R
from A to B is greater than 0.
```

is the transitive closure of R. In particular,

```
Accessibility relates a room (called A) to a room (called B) when the number of
moves from B to A is greater than 0. The verb to be accessible from means the
accessibility relation.
```

calculates the transitive closure of adjacency. Here, though, the way we normally understand "accessible from" suggests that it would be better to write:

```
Accessibility relates a room (called A) to a room (called B) when the number of
moves from B to A is at least 0.
```

which is reflexive as well as transitive. The usefulness of Inform's "next step via R from A to B" construction, in a wide variety of settings, reflects the importance of transitivity as an idea.

A relation which has all three properties of being reflexive, symmetric and transitive is called an "equivalence relation". (If all the map connections are two-way, then the accessibility relation above is symmetric and therefore a full equivalence relation: but if not, it may not be.) Inform has a special construction for making equivalence relations:

```
Nationality relates people to each other in groups.
```

This language - "in groups" - relies on the standard theorem that every equivalence relation on a set naturally defines a partition of that set, and vice versa. The "groups" referred to are what are normally called "equivalence classes". (Inform does little with these equivalence classes: it might be interesting to do so, in effect forming quotient kinds.)

---

244 ★★★★ Example **Graph-theory view of relations** *RB*

Some notes on relations from the point of view of graph theory.

One way to look at a relation is to regard it as a directed graph: that is, a collection of things ("vertices") with arrows drawn between them ("edges"). We write our items A, B, C, ... on a piece of paper: then, if A relates to B, we draw an arrow pointing from A to B, and so on. If we made this drawing for the adjacency relation, we would more or less have reconstructed the map (or at least a simplified one which does not care about precise directions, like the famous diagram of the London Underground). But the drawing can be made for any relation. If we define:

```
Suspecting relates various people to one person.
```

then, in the corresponding graph, each "vertex" will have at most one arrow leading away from it - though there could be many (or none) leading towards. Conversely, a one-to-various relation produces a graph where each vertex has at most one arrow coming in. A one-to-one relation means that the picture consists of some vertices on their own, with no arrows, a few perhaps with looped arrows leading from and to themselves, and then a collection of pairs joined by arrows. On the other hand, a various-to-various relation is just a free-for-all, with no restrictions on the arrows. The relations:

```
Meeting relates people to each other.
Marriage relates one person to another.
```

always have the property of working both ways round, and these are easiest to visualise by forgetting the direction of the arrows, so that they just become lines joining the vertices.

Inform uses a different algorithm for finding routes ("the next step via R from A to B") in each of these cases, and internally it stores relations in different formats in the different cases, because it makes a big difference to the efficiency of Inform to minimise the storage required for a relation and the time taken to explore it.

All the cases are benign except for "various to various" - the most useful - and for its closely related symmetrical version, "relates... to each other". Inform cannot afford to assume that the relation will be "sparse" (that is: that no vertex will have more than a certain number of arrows, or that the total number of arrows will be small), because it has no idea how the arrows will come and go during play. It therefore uses 1 bit of storage for each pair of objects. This sounds harmless, but if there are 200 rooms, there are 40,000 pairs of rooms, which means a 5000-byte allocation of storage (plus a handful of bytes as overhead). Gratuitous various-to-various relations are therefore not a good idea.

There is a standard algorithm for calculating shortest paths through a directed graph, but Inform does not always use it, because there is not always memory to store the required matrix of partial results. Inform's slow method, likely to be used on the Z-machine, requires a storage overhead which is equal to the number of vertices, not the square of that number, but the worst-case running time can be bad: if there are N vertices, and the diameter of graph (the longest distance between vertices) is D, then the running time is proportional to D times N. The worst case in finding routes from A to B is when almost every vertex can reach B, some across long trails, but A cannot. In the case of finding routes across the game's map, this must be multiplied further by the number of possible directions - usually 16.

This does not sound too awful, but if one is trying to find (say) "the most distant room from A", that means a further loop and now the running time will be D times N squared. Extension writers will need to be careful of this kind of thing: it is easy to write highly cool prototypes which work terribly slowly on larger, more realistic maps.

Chapter 14: Adaptive Text and Responses

↖ Contents of *Writing with Inform*
← Chapter 13: Relations
→ Chapter 15: Numbers and Equations
↘ Indexes of the examples

§14.1. Tense and narrative viewpoint

A conspicuous difference between interactive fiction and a traditional novel is the point of view from which it's told. Inform usually produces text like:

```
You can see a grey cat in the basket.
```

where a novel would usually write:

```
He saw a grey cat in the basket.
```

Standard interactive fiction (IF) is second person singular, and present tense; most novels are told in the third person singular, and past tense.

But these are just conventions - a few novels, for example, use the so-called present historic ("Napoleon looks up at the sky and sighs. Must Ney always be so doubting?"), and plenty are told in the first person singular ("I always get the shakes before a drop."). Inform allows some of this flexibility, too. The two values:

```
story viewpoint
story tense
```

control the style of the text produced. The story viewpoint has to be one of the values:

```
first person singular
second person singular
third person singular
first person plural
second person plural
third person plural
```

(which are actually the six possible values of a kind called "narrative viewpoint"), while the story tense must be one of:

```
past tense
present tense
future tense
perfect tense
past perfect tense
```

(from a kind called "grammatical tense"). Combining these gives 30 possibilities in all, though only a few are at all commonly used.

It's important to make a very large caveat here: Inform uses these settings in producing the replies ("responses") by the built-in actions, but the only way for all of our own text to have a particular tense or narrative viewpoint is to write it that way. If we write:

```
The Taj Mahal is a room. "You stand and admire the Taj Mahal."

When play begins:
    now the story viewpoint is first person plural;
    now the story tense is past tense.
```

then we're likely to see the following peculiar transcript:

```
Taj Mahal
You stand and admire the Taj Mahal.

>e
We couldn't go that way.
```

That's because the response ("We couldn't go that way") was constructed to follow the settings for viewpoint and tense, but the fixed text of the room description wasn't. In fact there are ways to write the room description so that it would adapt itself automatically, as we'll see, but it takes a fair amount of work. More simply:

```
The Taj Mahal is a room. "I stood and admired the Taj Mahal."

When play begins:
    now the story viewpoint is first person plural;
    now the story tense is past tense.
```

In short, tense and viewpoint switching is neat, but it isn't magic.

If we want to write text which will work in whatever the current tense is, the following turn out to be useful little conveniences:

say "[here]"

Produces "here" if the story tense is the present tense, and "there" otherwise.

say "[now]"

Produces "now" if the story tense is the present tense, and "then" otherwise.

⬆ Start of Chapter 14: Adaptive Text and Responses

⬅ Back to Chapter 13: Relations: §13.16. What are relations for?

➡ Onward to §14.2. Adaptive text

§14.2. Adaptive text

Paying attention to the tense and viewpoint is one reason why text might need to adapt. Another is that it might need to adapt according to whether nouns are singular or plural, or whether it talks about the player or some third party. For example, the following rule isn't ideal:

```
Instead of taking: say "[The noun] is pinned down by Dr Zarkov's force field."
```

Most of the time it's fine ("The V-ray is pinned down by Dr Zarkov's force field"), but then:

```
> GET ME
You is pinned down by Dr Zarkov's force field.
> GET CONDENSERS
The condensers is pinned down by Dr Zarkov's force field.
```

Which is a little unfortunate. But the correction is very easy:

```
Instead of taking: say "[The noun] [are] pinned down by Dr Zarkov's force field."
```

The result is much better: "The V-ray is pinned down..."; "You are..."; "The condensers are...". In fact, it's also convenient because it adapts to the story viewpoint and story tense: "The condensers will be pinned down..."; "He was pinned down...".

How does Inform do this? The answer is not that "[are]" is a specially-written text substitution. In fact Inform can do this with any verb that it has a definition of. For example,

```
"[The noun] [carry] too much static charge."
```

would also adapt itself - "The V-ray carries too much static charge", and so on. There aren't many verbs built in to Inform, but "[have]" and "[carry]" and "[wear]" and "[can]" may be useful, and "[can see]" and "[can touch]". Negative forms like "[are not]" are also available:

```
"[The noun] [cannot touch] the ionizer terminal."
```

might produce "The V-ray will not be able to touch the ionizer terminal.", for example.

As these examples hint, the verb adapts itself to the most recently printed object name. All of this only works if the previous object's name is printed from a substitution. So:

```
"[The condensers] [are] working."
```

will work -- correctly forming "The condensers are working.", "The condensers will be working." or "The condensers were working.", according to the story tense -- but

```
"The condensers [are] working."
```

probably won't work. Inform doesn't have any way to understand the raw text outside of the text substitution marks "[" and "]", and it doesn't recognise "The condensers" as being something's name.

Something else to be careful with is the use of lists. If we write this:

```
"[The condensers] and [the V-ray] [are] smashed by Voltan's birdmen."
```

then Inform is likely to print:

```
The condensers and the V-ray is smashed by Voltan's birdmen.
```

because it looks at the most recently named object - the V-ray, singular - to decide whether to use "is" or "are". On the other hand, Inform gets this right:

```
"[The list of things on the bench] [are] smashed by Voltan's birdmen."
```

Because Inform constructs the list itself, it's able to appreciate that the things listed are jointly the subject of the verb, and it uses that information to decide on "is" or "are". So:

```
The condensers and the V-ray are smashed by Voltan's birdmen.
The Atomic Furnace shovel is smashed by Voltan's birdmen.
```

Start of Chapter 14: Adaptive Text and Responses
Back to §14.1. Tense and narrative viewpoint
Onward to §14.3. More on adapting verbs

§14.3. More on adapting verbs

If we need an adaptive message with a verb which doesn't belong to Inform's built-in set, all we need do is define it. In the previous chapter we defined verbs by giving them meanings, but in fact that's optional. For example:

```
To retrofit is a verb.
```

defines a verb without telling Inform what it means. Inform will throw a Problem message if we try to write text like:

> Flash retrofits the meteor beam.

because, after all, it doesn't know what "retrofit" means. But it does still know how to print it, so this works:

> "[The actor] [retrofit] the Mecha-Mole."

which might come out as "Dale retrofits the Mecha-Mole", or "Barin's archers retrofitted the Mecha-Mole", and so on.

This is especially neat for writing a single response to an action which works regardless of who the actor was. For example, the Standard Rules include:

> say "[The actor] [put] [the noun] on [the second noun]."

And this can make either:

> You put the revolver on the table.
> General Lee puts the revolver on the table.

⬆ Start of Chapter 14: Adaptive Text and Responses

⬅ Back to §14.2. Adaptive text

➡ Onward to §14.4. Adapting text about the player

⬇ Example 245: ★★★ **Fun with Participles** Creating dynamic room descriptions that contain sentences such as "Clark is here, wasting time" or "Clark is here, looking around" depending on Clark's idle activity.

⬇ Example 246: ★★★ **Variety** Suppose we want all of our action responses to display some randomized variety. We could do this by laboriously rewriting all of the response texts, but this example demonstrates an alternative.

⬇ Example 247: ★★★ **Variety 2** This builds on the Variety example to add responses such as "You are now carrying the fedora" that describe relations that result from a given verb, as alternate responses.

⬇ Example 248: ★★★★ **Narrative Register** Suppose we want all of our action responses to vary depending on some alterable quality of the narrator, so that sometimes they're slangy, sometimes pompous or archaic.

---

§14.4. Adapting text about the player

In second-person-singular IF, the player is always "you". Many messages look like so:

> "You have twenty minutes remaining."

where the subject, or the object, of the sentence is "you". But what if we want to have this text adapt itself to different narrative viewpoints?

The solution is to use the following:

```
"[We]" or "[we]"
"[Us]" or "[us]"
"[Our]" or "[our]"
"[Ours]" or "[ours]"
"[Ourselves]" or "[ourselves]"
```

The capitalised and uncapitalised versions are identical except, of course, that the initial letter of the resulting text is upper case in one but not the other. As examples of these:

```
"[We] [carry] the Queen's warrant."
"The birds drop pebbles on [us]. Right on [our] heads!"
"[Ours] [are] the burdens of office, which [we] take on [ourselves]."
```

Notice that all five of these forms are differently worded, in English. That's the reason why we use the plural to write them - the traditional second person plural forms would be "you", "you", "your", "yours" and "yourself", so we wouldn't know if "[you]" was supposed to be the subject or the object of the verb. So the convention with all of these adaptive forms is that we use "we" and its variations. (That's also why the verbs are written in the plural - " [are]", not "[is]"; "[carry]", not "[carries]".)

⬆ Start of Chapter 14: Adaptive Text and Responses

⬅ Back to §14.3. More on adapting verbs

➡ Onward to §14.5. Adapting text referring to other things

§14.5. Adapting text referring to other things

The family in the previous section - "[we]", "[us]", "[our]", "[ours]", "[ourselves]" - always referred to the player. But we also sometimes want to refer to other things without naming them. For example, how should we adapt this?

```
> EXAMINE TREE
It has no clear outline in this misty netherworld.
```

We can easily make the verb adapt - change the "has" to "[have]" - but the trick here is to make the "It" adapt to cases where what's examined is plural, or animate. What we want is:

```
Instead of examining in the Netherworld:
    say "[regarding the noun][They] [have] no clear outline in this misty netherworld."
```

For example, this produces:

```
> EXAMINE ME
You have no clear outline in this misty netherworld.
> EXAMINE MARK
He has no clear outline in this misty netherworld.
> EXAMINE DRUMS
They have no clear outline in this misty netherworld.
```

Note that we have to say "[regarding the noun]", not just start in with "[They]", because nothing has been named so far in the sentence - so Inform doesn't know what object it refers to. "[regarding the noun]" prints nothing, and simply tells the printing part of Inform that the subject has changed.

This isn't always needed:

```
"[We] [have] a look at [the noun], but [they] [are] just too big."
```

works fine, because printing "[the noun]" changes the subject to that, and then "[they]" agrees with it automatically. The text might come out, for example, as:

```
I had a look at Peter Rabbit, but he was just too big.
You have a look at the chessmen, but they are just too big.
We have a look at ourselves, but we are just too big.
```

We have a family of five text substitutions here, matching those in the previous section:

```
"[They]" or "[they]"
"[Them]" or "[them]"
"[Their]" or "[their]"
"[Theirs]" or "[theirs]"
"[Themselves]" or "[themselves]"
```

There's also the peculiar impersonal non-object for English sentences like "It is raining" or "There are books":

```
"[It]" or "[it]"
"[There]" or "[there]"
```

These look pointless - but consider the two texts

```
"[We] [take] [the noun]. It [rain] harder."
"[We] [take] [the noun]. [It] [rain] harder."
```

The first one risks printing "We took the scissors. It rain harder.", because it makes "[rain]" agree with "scissors", which are plural. But the second text makes "[rain]" agree with "[it]". And, as a convenience:

```
"[It's]" or "[it's]"
"[There's]" or "[there's]"
```

do the obvious thing using the current story tense.

Finally, we occasionally want to agree with a number:

```
"Honestly, [dud count][regarding the dud count] of these [are] broken."
```

Start of Chapter 14: Adaptive Text and Responses
Back to §14.4. Adapting text about the player
Onward to §14.6. Adapting demonstratives and possessives

§14.6. Adapting demonstratives and possessives

Consider the following message: how might we make this adaptive?

```
> MEASURE TOP SHELF
You really are not tall enough to reach that.
```

The verbal part is easy enough, but "that" needs a new feature.

```
"[We] really [are not] tall enough to reach [regarding the noun][those]."
```

This could then adapt to, say,

```
> MEASURE JAM TARTS
He really was not tall enough to reach those.
```

Notice that it's "[regarding the noun][those]", not just "[those]". If we wrote "[those]", Inform would make it agree with the player, who was printed earlier in the sentence by the "[We]".

Lastly, how about:

```
> PUT TEAPOT IN MOUSEHOLE
The teapot's height is just too great.
```

This time we want:

```
"[regarding the noun][Possessive] height is just too great."
```

which might adapt to, say,

```
Our height is just too great.
Alice's height will be just too great.
```

Actually, "[regarding ...]" can be used for a description of possibly many items, too. For example:

Every turn when the player carries something:

```
say "Every possession is a worry. I wonder if [regarding things carried by the player]
[they] still [look] okay in your pocket?"
```

So if the player carries just a single coin, say, this automatically becomes:

Every possession is a worry. I wonder if it still looks okay in your pocket?

but if the player carries a pair of scissors (a single plural-named item) or a coin and an iPhone, it becomes:

Every possession is a worry. I wonder if they still look okay in your pocket?

Once again these text substitutions are available in capitalised and uncapitalised forms:

```
"[Those]" or "[those]"
"[Possessive]" or "[possessive]"
```

In fact "[Those]" and "[those]" do subtly different things, besides the capital letter, because "[Those]" expects to be the subject of the sentence and "[those]" the object, and this makes a difference if the noun in question is a person. If the noun is an odious person called Tilly then

```
"[regarding the noun][Those] is unacceptable."
"You've never liked [regarding the noun][those]."
```

would come out as "She is unacceptable" - so "[Those]" becomes "She" - but "You've never liked her" - so "[those]" becomes "her". If we need these in different cases, we can explicitly ask for that:

```
"[those in the nominative]"
"[Those in the accusative]"
```

Start of Chapter 14: Adaptive Text and Responses

Back to §14.5. Adapting text referring to other things

Onward to §14.7. Can, could, may, might, must, should, would

Example 249: ★ **Olfactory Settings** Some adaptive text for smelling the flowers, or indeed, anything else.

§14.7. Can, could, may, might, must, should, would

English uses so-called "modal verbs" to change a sentence so that it talks about something only possibly happening. For example, the sentence "Fred goes to school" can be modified to "Fred must go to school", "Fred should go to school" or even "Fred might go to school".

Inform supports the use of modal verbs in text substitutions. For example,

```
"[Fred] [might go] to school."
```

would in the present tense come out as "Fred might go to school.", but could alternatively be "Fred might have gone to school." As this example shows, all that's needed is to take a verb we'll call V - this case, "go" - and we can write any of these:

```
"[can V]" or "[cannot V]" or "[can't V]"
"[could V]" or "[could not V]" or "[couldn't V]"
"[may V]" or "[may not V]" or "[mayn't V]"
"[might V]" or "[might not V]" or "[mightn't V]"
"[must V]" or "[must not V]" or "[mustn't V]"
"[should V]" or "[should not V]" or "[shouldn't V]"
"[would V]" or "[would not V]" or "[wouldn't V]"
```

That helps us to handle informal usages like this one:

```
"You can't go that way."
```

To make this message adaptive, we write:

```
"[We] [can't go] that way."
```

which can adapt in surprising ways -- "They won't be able to go that way.", for example.

Note that the verb V has to be one that Inform knows. But that's easy:

```
To discombobulate is a verb.
```

and then

```
"[Fred] [might not discombobulate] so easily."
```

could produce "Fred might not have discombobulated so easily", for example.

⬆ Start of Chapter 14: Adaptive Text and Responses
⬅ Back to §14.6. Adapting demonstratives and possessives
➡ Onward to §14.8. Adapting contractions

§14.8. Adapting contractions

Contractions usually take the form of part of a word being missed out and replaced by an apostrophe. We've already seen "[can't]", "[couldn't]", "[mayn't]", "[mightn't]", "[mustn't]", "[shouldn't]" and "[wouldn't]", for example. But Inform supports other contractions, too, as follows.

The English verbs "to be" and "to have" are unique in having contracted forms, which we can write "['re]" and "['ve]", like this:

```
"[We]['ve] got rhythm. [We]['re] cool."
```

which might produce, say, "I've got rhythm. I'm cool.", or "He'll have rhythm. He'll be cool.", or "You had got rhythm. You were cool." (The contractions don't appear in the past tense; but the spacing fixes itself automatically.)

The Standard Rules often use a special text substitution for responses like this one:

```
"[They're] hardly portable."
```

This is exactly like "[Those]['re] hardly portable" except that if the plural is needed, Inform prints "They're hardly portable" rather than the correct, but not quite idiomatic, "Those're hardly portable". (If we wrote "[They]['re] ...", that would get the plural form right, but then the singular would be "It's hardly portable" not "That's hardly portable".)

Only a few English verbs have contracted negative forms, beyond those already mentioned. Inform knows these informal forms:

```
"[aren't]"
"[don't]"
"[haven't]"
"[won't]"
```

For example,

```
Instead of taking something:
    say "[The noun] [are] pinned down by Dr Zarkov's force field. [They] [aren't] free to move. [They] [can't] move. [They] [won't] move. [They] [haven't] a chance to move. Anyhow, [they] [don't] move."
```

can produce variations like these:

```
The condensers are pinned down by Dr Zarkov's force field. They aren't free to move. They can't move. They won't move. They haven't a chance to move. Anyhow, they don't move.

You were pinned down by Dr Zarkov's force field. You weren't free to move. You couldn't move. You wouldn't move. You hadn't a chance to move. Anyhow, you didn't move.
```

Start of Chapter 14: Adaptive Text and Responses
Back to §14.7. Can, could, may, might, must, should, would
Onward to §14.9. Verbs as values

§14.9. Verbs as values

Each verb known to Inform is actually a value of the kind "verb". To refer to a verb as a value, we have to put the word "verb" in front, as in these examples:

```
the verb contain, the verb might, the verb provoke
```

all of which appear in the Standard Rules.

Two adjectives are provided for use with verbs: "modal" (or "non-modal") to pick out verbs like might, could, should, and so on; and "meaningful" (or "meaningless") to pick out verbs which have a defined meaning as an Inform relation. For example, in the Standard Rules, the verb contain is meaningful, the verb might is modal, and the verb provoke is meaningless.

If V has a meaning as a relation of objects, then "meaning of V" produces that relation. For example,

```
showme the meaning of the verb contain;
showme the meaning of the verb provoke;
```

produces:

```
"meaning of the verb contain" = relation of objects: containment relation
"meaning of the verb provoke" = relation of objects: equality relation
```

As this demonstrates, if a verb has no meaning, or its meaning doesn't relate to objects, we get just the equality relation.

In fact, Inform even defines a verb "to mean": it's meaningful, and its meaning is the meaning relation. Thus:

```
if the verb mean means the meaning relation...
```

is true. More usefully, we can search our vocabulary like this:

```
the list of verbs meaning the containment relation
```

which, unless any non-Standard Rules definitions have been added, produces:

```
list of verbs: {verb contain}
```

Note that the meaning relation can't be changed at run-time: it is not clear what it would even mean to do something like -

```
now the verb contain means the wearing relation;
```

with the story already started, so this will produce a problem message.

say "[adapt (verb)**]"**

Adapts the given verb to the current story tense and story viewpoint. For example, "you [adapt the verb provoke]" might produce "you provoke".

say "[adapt (verb) **from** (narrative viewpoint)**]"**

Adapts the given verb to the current story tense but the given viewpoint. For example, "he [adapt the verb provoke from the third person singular]" might produce "he provokes".

say "[adapt (verb) **in** (grammatical tense)**]"**

Adapts the given verb to the given tense but the current story viewpoint. For example, "you [adapt the verb provoke in the past tense]" might produce "you provoked".

say "[adapt (verb) **in** (grammatical tense) **from** (narrative viewpoint)**]"**

Adapts the given verb to the given tense and viewpoint. For example, "we [adapt the verb provoke in the future tense from the first person plural]" might produce "we will provoke".

say "[negate (verb)**]"**

Adapts the given verb to the current story tense and story viewpoint, giving it a negative sense. For example, "you [negate the verb provoke]" might produce "you do not provoke".

say "[negate (verb) **from** (narrative viewpoint)**]"**

Adapts the given verb to the current story tense but the given viewpoint, giving it a negative sense. For example, "he [negate the verb provoke from the third person singular]" might produce "he does not provoke".

say "[negate (verb) **in** (grammatical tense)**]"**

Adapts the given verb to the given tense but the current story viewpoint, giving it a negative sense. For example, "you [negate the verb provoke in the past tense]" might produce "you did not provoke".

say "[negate (verb) **in** (grammatical tense) **from** (narrative viewpoint)**]"**

Adapts the given verb to the given tense and viewpoint, giving it a negative sense. For example, "we [negate the verb provoke in the future tense from the first person plural]" might produce "we will not provoke".

Note that the verb doesn't have to be named explicitly for use by the adapt or negate phrases, so for example:

```
To decide which text is the rendering of (V - verb) (this is my rendering):
    decide on "[negate V in the past perfect tense]".

When play begins:
    showme my rendering applied to the list of meaningful verbs.
```

produces:

```
"my rendering applied to the list of meaningful verbs" = list of texts: {"had not had", "had not related", "had not meant", "had not provided", "had not contained", "had not supported", "had not incorporated", "had not enclosed", "had not carried", "had not held", "had not worn", "had not been able to see", "had not been able to touch", "had not concealed", "had not unlocked"}
```

Lastly, we can get at three other useful parts of a verb, too. These aren't adaptive, of course: a verb only has one infinitive form.

say "[infinitive of (verb)**]"**

Produces the infinitive of the given verb. Note that this is without a "to": for example, "[infinitive of the verb carry]" is "carry", not "to carry".

say "[past participle of (verb)**]"**

Produces the past participle of the given verb. For example, "[past participle of the verb carry]" is "carried". Warning: because modal verbs like "should" or "might" are defective in English, this will produce odd results on them - "shoulded" and "mighted", for example.

say "[present participle of (verb)**]"**

Produces the present participle of the given verb. For example, "[present participle of the verb carry]" is "carrying". Warning: because modal verbs like "should" or "might" are defective in English, this will produce odd results on them - "shoulding" and "mighting", for example.

Start of Chapter 14: Adaptive Text and Responses

Back to §14.8. Adapting contractions

Onward to §14.10. Responses

Example 250: ★★ **History Lab** We create phrases such as "the box we took" and "the newspaper Clark looked at" based on what has already happened in the story.

Example 251: ★★ **Relevant Relations** An example of how to create room descriptions that acknowledge particular relations using their assigned verbs, rather than by the heavily special-cased code used by the standard library.

§14.10. Responses

Most of the text which the player sees is drawn from the source, but mixed in with this are messages apparently added by Inform itself - usually in the form of short sentences saying that something has been done, or that something can't be done. Such pieces of text are called "responses", because they are almost always replies to commands. For example:

```
> EAST
You can't go that way.

> JUMP
You jump on the spot.
```

Responses like this, which don't appear anywhere in the source text, come from one of the extensions being used; most often from the Standard Rules, the "extension" which is automatically included in every project. The SR contain many small rules, and almost all of these are capable of producing one or two standard responses. These are labelled with the rule's name and then a bracketed letter - (A), (B), (C), ... as needed so that every response has its own unique name. There's nothing very mysterious about how this is done. For example, here is a rule with one response:

```
Carry out taking inventory (this is the print empty inventory rule):
    if the first thing held by the player is nothing,
        say "[We] [are] carrying nothing." (A) instead.
```

which makes the familiar text "You are carrying nothing." a response named:

```
print empty inventory rule response (A)
```

These names are actually values, belonging to the kind "response". Because of that, if we try this:

```
say "Hmm: [print empty inventory rule response (A)]"
```

Inform will produce

```
Hmm: print empty inventory rule response (A)
```

since we gave Inform a value to print, and that's just what it then did. As an alternative:

say "[text of (response)**]"**

This text substitution writes out the current text of the given response.

Thus,

```
say "Hmm: [text of print empty inventory rule response (A)]"
```

produces

```
Hmm: You are carrying nothing.
```

Start of Chapter 14: Adaptive Text and Responses

Back to §14.9. Verbs as values

Onward to §14.11. Changing the text of responses

§14.11. Changing the text of responses

These responses are named so that they can be changed. Most IF authors dislike one or two of the existing responses, and some would like to change almost all of them to give the text a different style; and extensions for IF in languages other than English change literally every response, of course.

It's very easy to change responses:

```
The print empty inventory rule response (A) is "Your hands are, like, totally empty.
Lame."
```

and we can even do this dynamically during play:

```
now the print empty inventory rule response (A) is "Your hands ...";
```

just as if we were setting a variable.

Start of Chapter 14: Adaptive Text and Responses

Back to §14.10. Responses

Onward to §14.12. The RESPONSES testing command

Example 252: ★ **Responsive** Altering the standard inventory text for when the player is carrying nothing.

§14.12. The RESPONSES testing command

In practice we can't change these responses unless we know what they're called. One way to find out is just to read through the extensions we're using, but that's a laborious process. A more practical answer is to type:

```
> RESPONSES
```

which replies by listing the sets of responses currently available; for example, it says that RESPONSES 1 is the set of responses for the Standard Rules. We can then type exactly that:

```
> RESPONSES 1
Standard Rules:
    block vaguely going rule response (A): "You'll have to say which compass direction to
go in."
    print the final prompt rule response (A): "> [run paragraph on]"
    ...
```

and so on. This lists all of the responses, rule by rule, along with their current texts.

Start of Chapter 14: Adaptive Text and Responses

Back to §14.11. Changing the text of responses

Onward to Chapter 15: Numbers and Equations: §15.1. How do we measure things?

Examples from Chapter 14: Adaptive Text and Responses

Start of this chapter

Chapter 15: Numbers and Equations

Indexes of the examples

245 ★★★ **Example Fun with Participles** RB

Creating dynamic room descriptions that contain sentences such as "Clark is here, wasting time" or "Clark is here, looking around" depending on Clark's idle activity.

Mostly the Standard Rules use verbs adapted to finite forms ("he jumped", "we take the hammer", and so on). But Inform can also produce participles to describe actions that are ongoing: "he is carrying the fedora" or "taking the hammer..."

In this example, we give non-player characters actions to perform and then have Inform dynamically describe what they're doing when the player chooses to look.

We start by establishing the idea that a verb can describe a particular action:

```
"Fun with Participles"

Section 1 - Descriptive Functionality
```

```
Describing relates various verbs to various action names. The verb to describe means the describing relation.

To look around is a verb. The verb look around describes the looking action.

To stand about is a verb. The verb stand about describes the waiting action. To look bored is a verb. The verb look bored describes the waiting action. To waste time is a verb. The verb waste time describes the waiting action.

To jump is a verb. To leap is a verb. To pirouette is a verb. The verb jump describes the jumping action. The verb leap describes the jumping action. The verb pirouette describes the jumping action.
```

Now we need to give every character some sort of idle activity. By default, we'll have people just be waiting, but allow for that idle activity to change into something more interesting if the player has told them to do something else.

```
A person has an action name called the current idle. The current idle of a person is usually the waiting action.

Rule for writing a paragraph about someone (called chosen person) when a verb describes the current idle of the chosen person:
	say "[The chosen person] [are] here, [present participle of a random verb that describes (the current idle of the chosen person)]."

Instead of someone doing something:
	now the current idle of the person asked is (the action name part of the current action);
	continue the action.

A persuasion rule:
	persuasion succeeds.

Section 2 - Scenario

Lab is a room. The fedora is a wearable thing in the Lab. Clark is a man in the Lab.
```

And just to give past participles a test-drive as well, let's make Clark a bit of a drama king:

```
After Clark doing something when a verb describes (the action name part of the current action):
	say "'Fine, have it your way!' Clark exclaims. 'But I have [past participle of a random verb that describes (the action name part of the current action)] for the last time!'";
	rule succeeds.

Test me with "look / Clark, jump / look / Clark, look / look / Clark, wait".
```

246 ★★★ Example Variety RB

Suppose we want all of our action responses to display some randomized variety. We could do this by laboriously rewriting all of the response texts, but this example demonstrates an alternative.

Verbs can be related to other things by relations. We've seen that it's possible for a verb to "mean" a relationship. But we can also create a relation between verbs and actions. For instance, we can tell Inform that "take", "get", and "acquire" are all valid ways to describe the action of taking, and then allow it to pick a verb randomly to describe whatever action just occurred.

```
"Variety"

Section 1 - Descriptive Functionality

Describing relates various verbs to various action names. The verb to describe means the describing relation.

To take is a verb. To acquire is a verb. To get is a verb.

The verb take describes the taking action. The verb acquire describes the taking action. The verb get describes the taking action.

To drop is a verb. To put down is a verb. To discard is a verb. The verb drop describes the dropping action. The verb put down describes the dropping action. The verb discard describes the dropping action.

To sniff is a verb. To smell is a verb. The verb sniff describes the smelling action. The verb smell describes the smelling action.

To jump is a verb. To leap is a verb. To pirouette is a verb. The verb jump describes the jumping action. The verb leap describes the jumping action. The verb pirouette describes the jumping action.

After an actor doing something when the noun is nothing and a verb describes (the action name part of the current action) (this is the apply random verbs to describing nounless actions rule):
    say "[The actor] [verb rendering applied to a random verb that describes (the action name part of the current action)].";
    rule succeeds.

After an actor doing something to something when a verb describes (the action name part of the current action) (this is the apply random verbs to describing actions rule):
    say "[The actor] [verb rendering applied to a random verb that describes (the action name part of the current action)] [the noun].";
    rule succeeds.

To decide which text is the rendering of (V - verb) (this is verb rendering):
    decide on "[adapt V]".

Section 2 - Scenario
```

```
Lab is a room. The table is here. The bat and the ball are on the table.

Test me with "get ball / drop ball / get bat / drop bat / smell ball".
```

247

Example Variety 2

RB

This builds on the Variety example to add responses such as "You are now carrying the fedora" that describe relations that result from a given verb, as alternate responses.

Some of our default actions establish relations between items in the world, and reporting on the relation ("You are now carrying the fedora") can be a valid response alongside reporting on the action itself ("You take the fedora").

To do this, we need to teach Inform explicitly which relations are the results of actions, then check this when reporting on actions:

```
"Variety 2"

Section 1 - Descriptive Functionality

Describing relates various verbs to various action names. The verb to describe
means the describing relation.

Table of Action Results

    related action          relation
    the taking action       the carrying relation
    the wearing action      the wearing relation
    the taking off action   the carrying relation

To take is a verb. To acquire is a verb. To get is a verb.

The verb take describes the taking action. The verb acquire describes the taking
action. The verb get describes the taking action.

To drop is a verb. To put down is a verb. To discard is a verb. The verb drop
describes the dropping action. The verb put down describes the dropping action.
The verb discard describes the dropping action.

To sniff is a verb. To smell is a verb. The verb sniff describes the smelling action.
The verb smell describes the smelling action.

To jump is a verb. To leap is a verb. To pirouette is a verb. The verb jump
describes the jumping action. The verb leap describes the jumping action. The
verb pirouette describes the jumping action.

To don is a verb. The verb don describes the wearing action.

To doff is a verb. The verb doff describes the taking off action.
```

```
After an actor doing something when the noun is nothing and a verb describes (the action name part of the current action) (this is the apply random verbs to describing nounless actions rule):
    say "[The actor] [verb rendering applied to a random verb that describes (the action name part of the current action)].";
    rule succeeds.

After an actor doing something to something when a verb describes (the action name part of the current action) (this is the apply random verbs to describing actions rule):
    let current action name be the action name part of the current action;
    if a random chance of 1 in 2 succeeds and the current action name is a related action listed in the Table of Action Results:
        choose a row with the related action of current action name in the Table of Action Results;
        let R be the relation entry;
        let subject be the actor;
        let chosen object be the noun;
        say "[The subject] [are] now [present participle of a random verb that means R] [the chosen object].";
    else:
        say "[The actor] [verb rendering applied to a random verb that describes (the action name part of the current action)] [the noun].";
    rule succeeds.

To decide which text is the rendering of (V - verb) (this is verb rendering):
    decide on "[adapt V]".

To say infinitive of (V - a verb): (- {V}(1); -).

To say past participle of (V - a verb): (- {V}(2); -).

To say present participle of (V - a verb): (- {V}(3); -).

Section 2 - Scenario

Lab is a room. The fedora is a wearable thing in the Lab.

Test me with "wear the fedora / take off the fedora / wear fedora / take off fedora".
```

248 ★★★★ **Example Narrative Register** *RB*

Suppose we want all of our action responses to vary depending on some alterable quality of the narrator, so that sometimes they're slangy, sometimes pompous or archaic.

As we saw in "Variety", we can associate verbs with particular actions and call them up as needed. If we do that, though, we can also store additional information about those verbs and use that information to select the ideal verb to use in a particular situation.

In this example, we create a table of verbs and their meanings, together with some connotative information. Each time we report an action, we then score all the available verbs to decide which is the most suitable to use at the moment. This allows us to change the narrator's diction change mid-game and have the action descriptions change as well.

Moreover, because we're using adaptive verbs, these responses will automatically inflect properly even if we change the story tense and viewpoint.

```
"Narrative Register"

Section 1 - Descriptive Functionality

Describing relates various verbs to various action names. The verb to describe means the describing relation.

To take is a verb. To acquire is a verb. To get is a verb. To gain is a verb. To obtain is a verb. To pick up is a verb. To bag is a verb. To procure is a verb. To score is a verb. To grab is a verb. To snag is a verb. To snatch is a verb. To collect is a verb.

To drop is a verb. To put down is a verb. To discard is a verb. To throw away is a verb. To dispose of is a verb. To set down is a verb. To toss aside is a verb. To ditch is a verb. To abandon is a verb. To dump is a verb. To jettison is a verb. To abjure is a verb. To foresake is a verb. To dispense with is a verb.

After an actor doing something to something when a verb describes (the action name part of the current action) (this is the apply random verbs to describing actions rule):
    score the relevant verbs;
    sort the Table of Verb Meanings in reverse relevance order;
    choose row 1 in the Table of Verb Meanings;
    let top score be the relevance entry;
    sort Table of Verb Meanings in random order;
    repeat through the Table of Verb Meanings:
        if relevance entry is top score:
            say "[The actor] [verb rendering applied to (word entry)] [the noun].";
            erase relevance;
            rule succeeds.

To decide which text is the rendering of (V - verb) (this is verb rendering):
    decide on "[adapt V]".

To score the relevant verbs:
    repeat through the Table of Verb Meanings:
        if the meaning entry is (the action name part of the current action):
            increase relevance entry by 1;
            repeat with chosen connotation running through connotations entry:
                if the chosen connotation is listed in the current register:
                    increase relevance entry by 1;
                otherwise:
                    decrease relevance entry by 1.

To erase relevance:
    repeat through Table of Verb Meanings:
        now relevance entry is 0.

A tonality is a kind of value. The tonalities are pompous, archaic, slangy, upbeat, downbeat.

Connoting relates various verbs to various tonalities. The verb to connote means the connoting relation.

The current register is a list of tonalities that varies. The current register is { }.

When play begins:
    repeat through the Table of Verb Meanings:
        now the word entry describes the meaning entry;
        now relevance entry is 0;
        repeat with chosen tone running through the connotations entry:
            now the word entry connotes the chosen tone.

Table of Verb Meanings
```

word	meaning	connotations	relevance (a number)
the verb take	the taking action	{ }	
the verb acquire	the taking action	{ pompous }	
the verb get	the taking action	{ }	
the verb gain	the taking action	{ }	
the verb obtain	the taking action	{ pompous }	
the verb pick up	the taking action	{ }	
the verb bag	the taking action	{ slangy }	
the verb score	the taking action	{ slangy, upbeat }	
the verb procure	the taking action	{ archaic }	
the verb grab	the taking action	{ slangy }	
the verb snag	the taking action	{ slangy }	
the verb snatch	the taking action	{ slangy }	
the verb collect	the taking action	{ }	
the verb discard	the dropping action	{ pompous }	
the verb drop	the dropping action	{ }	
the verb put down	the dropping action	{ }	
the verb toss aside	the dropping action	{ }	
the verb ditch	the dropping action	{ slangy }	
the verb throw away	the dropping action	{ }	
the verb dispose of	the dropping action	{ }	
the verb set down	the dropping action	{ }	
the verb abandon	the dropping action	{ downbeat }	
the verb dump	the dropping action	{ downbeat }	
the verb abjure	the dropping action	{ archaic }	
the verb foresake	the dropping action	{ archaic }	
the verb jettison	the dropping action	{ pompous }	
the verb dispense with	the dropping action	{ pompous }	

```
Section 2 - Changing Tone Mid-Game

Understand "new tone" as changing the tone. Changing the tone is an action out of world.

Carry out changing the tone:
    now the current register is { };
    if a random chance of 1 in 4 succeeds:
        say "Your narrator will now adopt an ordinary tone.";
        rule succeeds;
    let rando be a random tonality;
    add rando to the current register, if absent;
    say "Your narrator will now be [rando]."
```

```
Section 3 - Scenario

Lab is a room. The table is here. The bat and the ball are on the table.

Test me with "get ball / drop ball / get bat / drop bat / new tone / get all / drop all /
new tone / get all / drop all".
```

249 ★ **Example Olfactory Settings** RB

Some adaptive text for smelling the flowers, or indeed, anything else.

While this isn't very interesting as IF, it runs through most of the adaptive-text tricks.

```
"Olfactory Settings"

The Doghouse is a room. "Not so much a place as a state of being."

The player carries a ticket to the opera, some papers, and a bouquet of flowers.
The bouquet is ambiguously plural.

Instead of eating something inedible, say "[The noun] [don't] seem likely to agree
with [us] at all. [We][']d be wiser to leave [regarding the noun][them] alone."

Instead of touching something: say "[regarding the noun][Those] [are] all prickly."

Instead of smelling something: say "[Our] nose [regarding nothing][are] too weak
to get much smell from [regarding the noun][those]."

Instead of smelling the bouquet: say "[regarding the noun][They]['re] lovely."

Instead of tasting something:
    say "Whew, [regarding the noun][are] [those] ever nasty!"

Test me with "x ticket / eat it / eat them / touch it / touch them / smell it / smell
them / taste it / taste them / x papers / eat it / eat them / touch them / smell them
/ taste them / x bouquet / eat it / eat them / touch them / smell them / taste
them".
```

250 ★★★ **Example History Lab** RB

We create phrases such as "the box we took" and "the newspaper Clark looked at" based on what has already happened in the story.

The examples Variety and Narrative Register show how verbs can be associated with particular actions. Here, we use the same principle so that we can report to the player what was last done to a particular object, either by the player or by someone else.

To do this, we need to use the idea of stored actions from the Advanced Actions chapter.

```
"History Lab"

Section 1 - Procedure

An object has an action called the last action.

Describing relates various verbs to various action names. The verb to describe means the describing relation.

To take is a verb. The verb take describes the taking action.
To drop is a verb. The verb drop describes the dropping action.
To look at is a verb. The verb look at describes the examining action.
To examine is a verb. The verb examine describes the examining action.

After an actor doing something to something:
    if a verb describes the action name part of the current action:
        now the indefinite article of the noun is "the";
        now the last action of the noun is the current action;
    continue the action.

After printing the name of something (called item):
    if the last action of the item is not waiting and the last action of the item is not the current action:
        let chosen action-name be the action name part of the last action of the item;
        let chosen actor be the actor part of the the last action of the item;
        if a verb describes the chosen action-name:
            let the chosen verb be a random verb that describes the chosen action-name;
            say " [if the chosen actor is the player][we][else][chosen actor][end if] [adapt chosen verb in past tense]";

Section 2 - Scenario

Lab is a room. It contains a box. The box contains a newspaper. Clark is a man in the Lab.

A persuasion rule:
    persuasion succeeds.

Test me with "x box / look / x newspaper / look / clark, x newspaper / clark, get box / clark, drop box / look / take box / i / smell box / i".
```

Notice that smelling the box does not change the box's description because we haven't gotten around to defining a smell or sniff verb.

251 ★★★ Example **Relevant Relations** RB

An example of how to create room descriptions that acknowledge particular relations using their assigned verbs, rather than by the heavily special-cased code used by the standard library.

Suppose that we wanted authors to be able to indicate which relations should or should not be included in room descriptions, and have the system dynamically honor that instruction.

Inform already knows about verbs for describing supporting, containment, carrying, and wearing, so we could write a set of instructions to handle such cases. To do this, we're using the "writing a paragraph about" activity, which is described in the chapter on activities.

```
"Relevant Relations"

Section 1 - Procedure

Rule for writing a paragraph about something (called item):
    now the current paragraph is { };
    say "[one of][regarding item]There [are] [an item] here[or][We] [can see] [an item] here[at random]. [run paragraph on]";
    follow the descriptive rules for the item;
    repeat with new item running through the current paragraph:
        now the prior named object is nothing;
        if new item is not the item:
            follow the descriptive rules for the new item;
    say paragraph break.

Rule for writing a paragraph about someone (called chosen person):
    now the current paragraph is { };
    say "[one of][regarding chosen person][The chosen person] [are] here[or][We] [can see] [a chosen person] here[at random]. [run paragraph on]";
    follow the descriptive rules for the chosen person;
    repeat with new item running through the current paragraph:
        now the prior named object is nothing;
        if new item is not the chosen person:
            follow the descriptive rules for the new item;
    say paragraph break.

The descriptive rules are an object-based rulebook.

Definition: a container is see-through:
    if it is transparent:
        yes;
    if it is open:
        yes;
    no.

A descriptive rule for a see-through container (called item) (this is the describe contents rule):
    describe the containment relation for item.

A descriptive rule for a supporter (called item):
    describe the support relation for item.
```

```
A descriptive rule for a person (called item):
    describe the wearing relation for the item.

A descriptive rule for a person (called item):
    describe the carrying relation for the item.

The current paragraph is a list of things that varies.

Before printing the name of something (called mentioned target) while writing a
paragraph about something:
    add the mentioned target to the current paragraph, if absent.

To describe (R - a relation of objects) for (item - a thing):
    if a thing to which item relates by R is a thing:
        say "[The item with pronoun] [verb rendering applied to a random verb that
means R] [the list of things to which item relates by R with indefinite articles].
[run paragraph on]"

To decide which text is the rendering of (V - verb) (this is verb rendering):
    decide on "[adapt V]".

To say (T - a thing) with pronoun:
    if T is the prior named object:
        say "[regarding T][They]";
    else:
        say "[The T]"

Section 2 - Scenario

The Space Elevator is a room. "Mercifully, there aren't any windows. The ability
to see how far up you are would almost certainly make you ill."

The luggage rack is a supporter in the Space Elevator. The suitcase is a closed
openable container on the luggage rack. The bouquet is on the luggage rack.

Clark is a man in the Space Elevator. Clark is carrying a box of cupcakes. Clark
is wearing a t-shirt. The description of the box of cupcakes is "They're the latest
confection from Red Velvet Planet, the Martian bakery."

Persuasion rule: persuasion succeeds.
```

We can if we like then add alternate names for these relations that will be randomly swapped in some of the time. For instance:

```
To sport is a verb. The verb to sport means the wearing relation.

To hold up is a verb. The verb to hold up means the support relation.

Test me with "clark, drop the box / look / clark, take the suitcase / look / clark, get
bouquet".
```

One might, hypothetically, imagine going even further than this and simply designating relations as either "important" or "unimportant" -- perhaps changing the relation's designation at runtime. Relations are not themselves allowed to have properties, however.

252 **Example Responsive** *RB*

Altering the standard inventory text for when the player is carrying nothing.

The most straightforward way to alter the response text for something in the standard rules is to select the Index tab, then select Actions, then pick the particular action whose text we'd like to alter. Under action details, there will be icons that look like speech bubbles.

Clicking on the speech bubble will show what the current response text is, and give us an option called "set".

If we click "set", this will automatically paste in the response name that we need to change. We can put this inside a "when play begins" rule in order to make that change take effect from the start of the game, like so:

```
"Responsive"

An Anonymous B613 Cell is a room. "There isn't much to see in this bare room. What there is, you've already seen sometime in the last twenty years."

When play begins:
	now print empty inventory rule response (A) is "[We] [have] absolutely nothing.".

Test me with "i".
```