Inform Introduction and Reference Guide

Reference Conventions

Text contained within < > signs (example <name of room>) represents a placeholder. When writing your own programs you will need to substitute your own appropriate terms without the < > signs. All other characters, including [and] should be used literally when writing Inform code.

So for example if the instructions say "<type text here>" you might type in Inform "the golden apple"

If the instructions say "[<topic of conversation>]" you might type in Inform "[healthcare]"

Contents

Part I The Three Main Types of Nouns: Rooms, Things, & People.

Part II Interacting with People

Part III Interacting with Characters Part II: Giving Things to Characters and Getting Things from Characters

Part IV Having Characters Respond Differently Depending upon What the Player is Carrying.

Part V Events – Making things happen after certain actions

Part I – The Three Main Types of Nouns: Rooms, Things, & People.

A. Rooms

A room is any location in Inform or, to be more precise, any place where the player is given a chance to enter commands. A room can be an actual room, a location ("the grassy field") or even a part of a location ("upper grassy field" and "lower grassy field").

To create a room with a description the player sees upon entering.

Format:
```
<name of room> is a room. "Cool descriptive text here."
```
Example:
```
Cave is a room. "This is a dank, dark cave."
```

To make additional rooms after the first.

Format:
```
<name of room2> is <direction> of <name of room1>. "<Cool descriptive text of room 2 here>."
```
Example:
```
The forest is south of the cave. "You are in a forest to the south of the cave."
The river is north of the cave. "A shallow river, more of a stream really, runs past on the north side of the cave."
```

Note: The main directions are north, northeast, east, southeast, south, southwest, west, northwest, north. Above and below are also common; change phrase to read "above the" or "below the" not "above of" or "below of".

B. Things

A thing is any object. When a thing is created in Inform, a player can examine, take, and drop that thing by default.

To create a thing and set its starting place in a room

Format:
```
<name of thing> is in <name of room>. The description of
```

the <name of thing> is "<description of the thing>."

Example:
A hammer is in the cave. The description of the hammer is "A rusty old hammer.

To create a thing and give the thing to the player to carry.

Format:
The player is carrying the <name of thing>. The description of the <name of thing> is "description of thing."

Example:
The player is carrying the hammer. The description of the hammer is "A rusty old hammer."

To create a more detailed description of a thing that will display when the user `looks at` or `examines` the thing:

Format:
The description of the <name of thing> is "<description of thing>."

Example:
The player is carrying the hammer. The description of the hammer is "A rusty old hammer.

Notes about things:

- Items do not require descriptions (though descriptions add to the richness of the simulation world) things. Inform will display the names of things in the room if no description is given for them.
- Unless the sentence `<name of thing> is fixed in place` is written, the thing can be taken and dropped by the user.

C. People: Men, Women, and Animals

People are the characters the player will encounter in the game. There are three types of people: men, women, and animals

To create a person and set its starting place in a room:

Format: (3 options based on man, woman, and animal)
<Name of person> is a man. <Name of person> is in <name of

room>.
<Name of person> is a woman. <Name of person> is in <name of room>.
<Name of person> is an animal. Name of person> is in <name of room>.

Examples:
Caesar is a man. Caesar is in the study.
Emmeline Pankhurst is a woman. Emmeline Pankhurst is in the Parliament.
The grizzly bear is an animal. The grizzly bear is in the cave.

Note: The number of sentences can be reduced by combining the statement to create the person and the statement to put the person in a particular room.

Format:

<name of person> is a woman in the <name of room>

Examples:

Emmeline Pankhurst is a woman in the Parliament.

To create a more detailed description of a person that will display when the user `looks at` or `examines` the person:

Format:
The description of <name of person> is "<description of person.>"

Examples:
The description of Caesar is "A man of distinction whose ready access to power has ingrained a look of confidence in his eyes."
The description of Emmeline Pankhurst is "A cheerful looking woman with an impish curve in her smile, she looks ready for trouble."
The description of the grizzly bear is "A bear that is old enough to be grizzled if bears had grey hairs."

Part II – Interacting with People

The player asks a person about something by typing in the command `>ask <character name> about <text>` and tells a character about something by entering the command `>tell <character name> about <text>`. Inform will only understand the text entered by the player if it matches the text specified by the designer exactly. For example, if the designer writes the statement:

```
After telling Marie Antoinette about "[hunger]":
     say "Marie says 'Well, if they cannot afford cake,
let's get them some bread'."
```

The `say` statement will only be activated if the player enters `>tell Marie Antoinette about hunger`.

Often no more flexibility is needed if the designer leaves clues in the form of precise words the player can ask and tell about from the room descriptions.

When more flexibility is needed so the player can enter a range of similar phrases and be understood by Inform — for example "hunger" "hungry people" or "starving peasants", the `understand` statement is used. The purpose writing an understand statement in this context is to allow a player to speak about a topic, without having to know the exact wording of the topic.

To create a series of conversation topics that the player can talk about using a variety of phrases.

Format:

```
Understand "<word or phrase>" or "<word or phrase>" or
"<word or phrase>" as "[<topic name>]".
```

Example:

```
Understand "hunger" or "hungry people" or "the hungry
people" or "starving peasants" or "the starving peasants"
as "[hunger]".
Understand "farewell" or "farewell address" or "last words"
or "his last words" as "[farewell]".
```

To give a response to the player when she asks a character about something:

Format:

```
After asking character name about "[<topic name>]":
     say "<text that the player will see>."
```

Examples:

```
After asking Marie Antoinette about "[hunger]”:
     say "Marie says ‘let them eat cake’.

After asking George Washington about "[farewell]":
     say "‘It would be best,’ muses Washington, ‘ if our
country avoided foreign entanglements’.
```

To give a response to the player when he tells a character about something:

Format:

```
After telling <character name> about "[<topic name>]":
     say "text that the player will see."
```

Examples:

```
After telling Marie Antoinette about "[cake]":
     say "Marie says ‘Well, if they cannot afford cake,
let’s
get them some bread’."

After asking George Washington about "[entanglements]”:
     say "‘I agree. Entanglements can be very binding’."
```

Putting it all together:

```
Understand "hunger" or "hungry people" or "the hungry
people" or "starving peasants" or "the starving peasants"
as "[hunger]."

After telling Marie Antoinette about "<hunger>":
     say "Marie says ‘Well, if they cannot afford cake,
let’s get them some bread’."
```

Notes about Ask, Tell, and Understand:

- Pay attention to the use of brackets around a topic in an understand statement. They are critical. "`[hunger]`" is not the same thing as "`hunger`" to Inform.

Part III – Interacting with Characters Part II: Giving Things to Characters and Getting Things from Characters

Assuming that we have created at least one room, one item, and one character (see the guidelines above for doing so), players can give an item to the character.

To allow the player to give an item to the character and have the character respond

Format:

```
Instead of giving the <item name> to <character name>:
     say "<text that the player will see>";
     now character name is carrying <item name>.
```

Examples:

```
Instead of giving the apple to Hera:
     say "Hera raises her chin in royal triumph.";
     now Hera is carrying the apple;
```

To allow the player to get an item from the character after asking about it

Format:

```
After asking <character name> about <item name>:
     say "<text that the player will see>”;
     now player is carrying <item name>.
```

Examples:

```
After asking Marie about cake:
     say "Marie says, ‘See ? It’s very nice cake.’”;
     now the player is carrying the cake;
```

Notes:

- When more than one statement follows and is associated with an "`instead of ...`" or an " `after ...` " statement as in the cases above, each statement before the final one must be ended with a semicolon, not a period. A semicolon, as in English, indicates a less than full stop between independent clauses that are meant to be read together.

Part IV– Having Characters Respond Differently Depending upon What the Player is Carrying.

Assuming that we have created at least one room, one item, and one character (see the guidelines above for doing so), we can write what is called a conditional statement, so that the character will respond one way if a condition is true and another if the conditions is not true. In this case, the condition is that the player is carrying a certain item.

Format:

```
After asking character name about "[<topic name>]":
      if the player is carrying <item name> begin;
            say "<text that the player will see if she is carrying
      the right thing>."
      otherwise;
            say "<text that the player will see if she is NOT
      carrying the right thing.>";
      end if.
```

Example:

```
After asking Eleanor of Aquitaine about "[state of affairs]":
      if the player is carrying the royal signet ring begin;
            say "Eleanor expounds at length on the troubles facing
Aquitaine.";
      otherwise;
            say "Eleanor says, 'who are you? Why should I discuss
affairs of state with you?.";
      end if.
```

Part V Events – Making unique things happen after certain actions

A. Triggering Different Dialogue/Events During Conversation.

To change character responses when the player uses the same command more than once.

Format:

```
After asking <name of character> about "<topic>":
     say "<Informative and cool text>";

After asking <name of character> about "<topic>" more than
<number> times:
     say "I have told you all I know. Stop asking me about
this.";
```

Example:

```
After asking Cicero about "Antonius":
     say "'I think Antonius is a jerk'";

After asking Cicero about "Antonius" more than one time:
     say "'I have told you all I know. Stop asking me about
this.' Cicero leaves the room.";
     Move Cicero to the Forum.
```

B. Triggering Events When Entering a Room

To make something happen every time the player enters a room.

Format:

```
After going in <name of room>:
     try the player looking;
     say "<text>";
     <any other statements you want to carry out>;
     <last statement ends with a period>.
```

Example: (this code causes the Cicero character to come to the assembly every time the player is there.)

```
After going in the assembly:
```

```
     now Cicero is in the assembly;
     try the player looking;
```

To make something happen (text displayed, etc.) the first time the player enters a room.

Format:

```
After going in <name of room> for the first time:
     try the player looking;
     say "<text>";
```

Example:

```
After going in the assembly for the first time:
     try the player looking;
     say "the assembly is still in the middle of its
     debate"
```

To make something happen (text displayed, etc.) when the player enters a room a specific number of times.

Format:

```
After going in <name of room> for the
<first/second/third/fourth etc.> time:
     try the player looking;
     say "<text>";
```

Example:

```
After going in the assembly for the second time:
try the player looking;
say "the assembly is still in the middle of its debate"
```

To make something happen (text displayed, etc.) when the player enters a room more than a set number of times.

Format:

```
After going in <name of room> more than <number> times:
     try the player looking;
     say "<text>";
```

Example:

```
After going in the assembly more than twice:
     try the player looking;
     say "Nothing new is happening. Explore somewhere else".
```