Inform7 Cheat Sheet

v0.2

by Mark-Oliver Reiser

March 2011

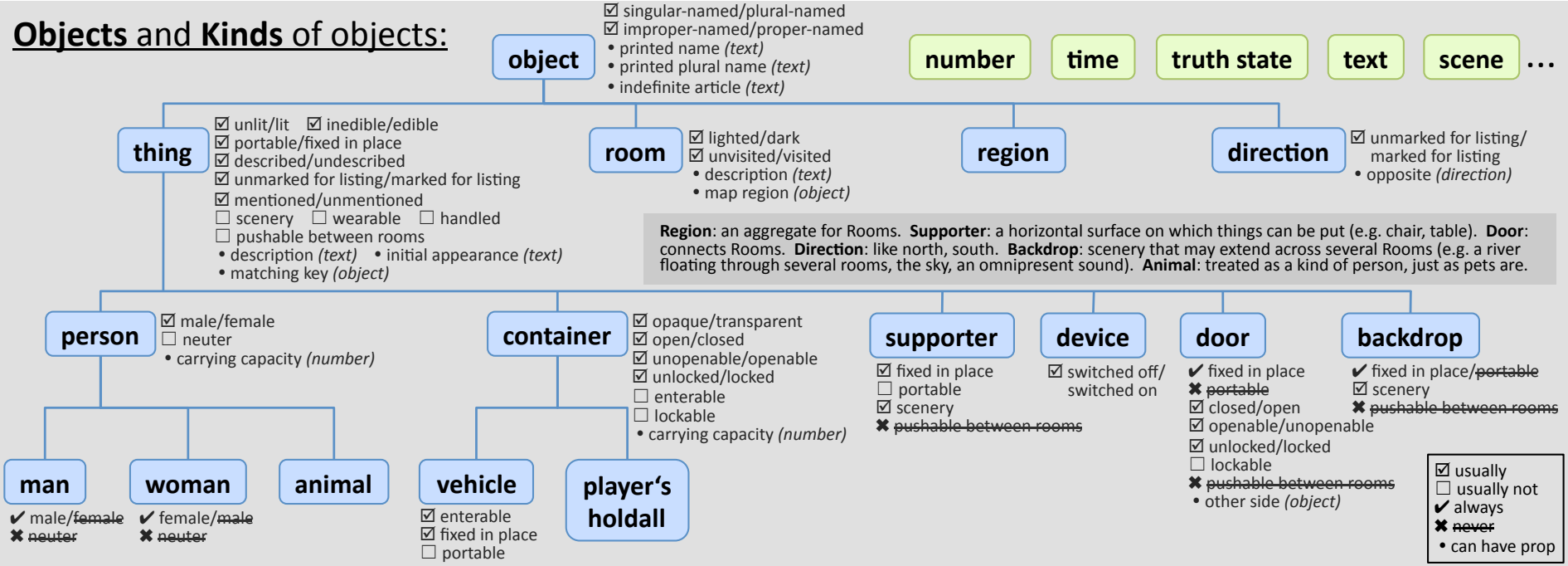
most examples taken from

Writing with Inform by Graham Nelson
(part of the Inform7 documentation)

and

Inform7 for Programmers by Ron Newcomb

Objects and **Kinds** of objects:



Building blocks of Inform7 source text:

Phrases are the fundamental building blocks.

Some phrases may stand alone in the source (esp. those stating facts about the game world).

The entrance hall is a room .

} **Phrase** (stand alone)

Other phrases may not appear alone, because Inform7 does not know when to execute them. ***Rules*** *are one way to specify when to execute a phrase.*

Preamble (of the rule) — **Rule** — **Action** — **Description**

Instead of taking a large container :
say "[The noun] seems a bit too heavy."

Text replacement

} **Phrase** (in a rule)

Phrases

Phrases are the fundamental building blocks of Inform7 source text; they can have very different uses.
A phrase ends with a full stop which may be omitted if the phrase ends with a string ending with "." or "!". Several phrases can be concatenated into a composite phrase by separating them with a semicolon.

Say "Bob".
Say "Hi."
Say "Hello" ; say " world."

Phrases to state facts about the game world

The <object> **is a** <description>. [the articles are optional]

The Lobby is a room.
The cave is a room.
The treasure chest is a locked container in the cave.

Phrases to do something

Say … .	Say "Hello world."
Move … **to** … .	Move the chest to the lobby.
Now … **is** … .	Now the chest is unlocked.
Change … **to** … .	Change the chest to unlocked. *(Deprecated)*

Phrases to take a decision

If <condition> **then** <phrase>.

If <condition> **then** <phrase>; **otherwise** <phrase>.

If <condition>, <phrase>.

If <condition>, <phrase>; **otherwise** <phrase>.

```
If <condition> begin;          [may all appear on a single line]
    <phrases>;
otherwise if <condition>;
    <phrases>;
otherwise;                     [note the semicolon]
    <phrases>;
end if.
```

```
If <condition>:                [phrases must appear on separate lines]
    <phrases>;
otherwise if <condition>:
    <phrases>;
otherwise:
    <phrases>.
```

```
If <variable> is:
    -- <value>: <phrase>;
    -- <value>: <phrase>;
    -- otherwise: <phrase>.
```

Phrases to repeat other phrases

```
Repeat with <variable> running from <value> to <value> begin;
    <phrases>;
end repeat.
```

```
Repeat with <variable> running through <description of values>
begin;
    <phrases>;
end repeat.
```

```
Repeat through <table> begin;
    <phrases>;
end repeat;
```

```
While <condition> begin;
    <phrases>;
end while.
```

Phrases to compute a value

(called to „decide" a value in Inform7 parlance)

square root of 16

Phrases to introduce local variables

Let <variable> **be** <value>.

Let X be 5.

Custom phrases

To <phrase pattern>: <phrases>.

Use **(** <variable> **-** <kind> **)** in phrase pattern to introduce parameters:

To plainly greet (friend - a person): … [named parameter]
To simply/plainly greet (friend - a person): … [/ marks alternatives]
To simply/plainly/-- greet (friend - a person): [-- empty alternative]
    Say „Hi [friend]."

Custom conditions:

To decide whether <phrase pattern>:
    <phrases>; **decide yes/no.** *OR* **decide on** <truth value>.

To decide whether danger lurks:
    if in darkness, decide yes;
    no. ['decide' is optional]
To decide whether (pants - a thing) is/are on fire:
    decide on whether or not a random chance of 1 in 2 succeeds.

Custom value computations:

To decide which/what <returned kind> **is** <phrase pattern>:
    <phrases>; **decide on** <value>.

To decide which number is double (N - a number):
    decide on N times N.

Custom say phrases:

To say <phrase pattern>: <phrases>.

To say He-Shе for (P - a person):
    if P is female, say "She"; otherwise say "He".
say "[He-Shе for Chris] glances at you.";

Rules

A rule consists of a preamble and one or more phrases. The preamble tells Inform7 when to execute the phrases.

> *In fact, this is a simplification: the preamble gives a name to the rule (optional) and tells Inform7 in which rulebook the rule should be put (optional). However, since the main difference between rulebooks is the occasion on which the contained rules will be applied, you can think of the preamble as a specification of when to execute the rule's phrases.*

<preamble> : <phrases>.

If the preamble begins with **Before**, **After**, **Instead of**, **Every turn**, or **When**, and there is only one phrase, an alternative notation is available:

<preamble> , <phrase>.

The following are NOT rule definitions:

To … : … .	[defines a new phrase, see above]
At … : … .	[defines a timed event]
Definition … : … .	[defines new adjective, see Descriptions]

Creating a rule

(1) creating an unnamed rule put in a rulebook:

A/an <rulebook name> **rule:** <phrases>. [indefinite article!!]

The indefinite article and the word "rule" are optional. So, given the rulebook named "every turn", the following are all legal:

An every turn rule: say "Howdy."
Every turn rule: say "Hi."
Every turn: say "Hi."

(2) creating a named rule not put in any rulebook:

This is the <rule name> **rule:** <phrases>. [definite article!!]

This is the notorious greet rule: say "Hi."

(3) creating a named rule put in a rulebook:

A/an <rulebook name> **rule (this is the** <rule name> **rule):**
    <phrases>.

An every turn rule (this is the greet rule): say "Hi."
Every turn (this is the greet rule): say "Hi."

Circumstances when putting a rule in a rulebook

first/last <rulebook> **about/for/of/on/rule** <action>
while/when <condition> **during** <scene> **:** <phrases> .

first/last tells Inform7 where to put the rule in the rulebook.
about/for/of/on/rule are just fillers to improve readability, but can help avoid ambiguities by separating the rulebook name from what follows:

Instead of kissing Clark, say "No thanks." [all equivalent]
Instead rule about kissing Clark: say "No thanks."
Instead kissing Clark: say "No thanks."

Built-in rulebooks

when play begins **every turn** **when play ends**	
before **instead** **after**	Executed during action processing, see Section "Main Cycle" below.
visibility **does the player mean** **persuasion** **unsuccessful attempt**	See Writing with Inform, Chapter 12.19. On persuasion and unsuccessful attempts see Writing with Inform, Chapter 12.5-6.
reaching inside **reaching outside**	
for each action: **check** <action name> **carry out** <action name> **report** <action name>	Executed during action processing, see Section "Main Cycle" below.
for each scene: **when** <scene name> **begins** **when** <scene name> **ends**	See Writing with Inform, Chapter 10.
for each activity: **before** <activity name> **for** <activity name> **after** <activity name>	Dealing with activities is usually not required, more elegant alternatives exist in most cases. Writing with Inform, Chapter 17.

Ending a rule

Ending a rule with an outcome – no matter if 'success' or 'failure' – will immediately end the entire rulebook, i.e. no more rules will be processed.

rule succeeds. [end rule with outcome 'success']
rule fails. [end rule with outcome 'failure']
make no decision. [end rule with no outcome]

… instead. [end rule with outcome 'failure']
stop the action. [end rule with outcome 'failure']
continue the action. [end rule with no outcome]

End with a result value:

rule succeeds with result <value> . [accordingly for "rule fails" etc.]

Rulebooks can define other, so-called **named outcomes**, that look like phrases. E.g. the visibility rulebook defines:

there is sufficient light; [end rule with named outcome]
there is insufficient light;

Phrases for rules

follow <rule> / <rulebook> . [apply rule, throw away result]
follow <rule> / <rulebook> **for** <value>. [for a value based rulebook]

consider <rule> **for** <value> . [apply rule, throw away result]

abide by <rule> **for** <value> . [apply rule, ***end*** with its result]
anonymously abide by <rule> **for** <value>.

<kind> **produced by** <rule producing values> **for** <value> .

Follow *and* **consider** *are equal except with respect to procedural rules (which are deprecated).* **Anonymously abide** *means that from the outside it will seem as if the abiding rule was not involved at all and the result was produced by <rule>; with* **abide** *it seems as if the abiding rule had produced the result itself.*

Custom rulebooks

<rulebook name> **is a** <value> **based** **rulebook** **producing** <value>.

The <rulebook name> **has default failure / success / no outcome.**

The <rulebook name> **has outcomes** <outcomes> .

Adding rulebook variables (global to all rules in the rulebook):

The <rulebook name> **has a** <kind> / <type> **called** <variable name> .

Appraisal rulebook is a rulebook.
The cat behavior rules are a rulebook producing an animal.
The probability rules have outcomes it is likely, it is possible and it is unlikely.

Descriptions

A description identifies one or more objects. It can be used when creating new objects (a new object will be created such that it matches the description) and when denoting existing objects.

A description consists of zero or one noun and zero or more adjectives:

<adjectives> <noun>

Nouns

The <noun> is the name of a kind, the name of an object or one of eight special nouns (with only three different meanings):

something = **anything**	equivalent to the noun **thing**
someone = **somebody** = **anyone** = **anybody**	equivalent to the noun **person**
somewhere = **anywhere**	equivalent to the noun **room**

the open wine cask [adjective open + noun wine cask]
something portable [the noun thing + the adjective portable]

Built-in Adjectives

visible **touchable** **adjacent** **visited** **in**	
which/who:	
all / each / every / everybody **some / most / almost all** **none / no / nobody**	

a light cloak [which is] worn by a woman [who is] in a dark room

Custom adjectives

(1) ordinary boolean or many-valued properties on kinds
A person can be grumpy or happy.

(2) definitions:

Definition: a <kind> **is** <adjective> **rather than** <adjective>
if <condition> **.**

Definition: A room is occupied if a person is in it.
Definition: A container is large rather than small if its carrying capacity is 20 or more.

Comparatives/superlatives are automatically generated:
the largest container

(3) a generic "to decide whether" boolean function (see above; cannot be combined with other adjectives; but works on everything, not only objects).

Actions

An action is something the player's character can do to interact with the game world, e.g. moving around, taking things, opening a door.

Two things are important to note about actions:
(1) Actions are usually triggered by the player by entering a command via the keyboard, but actions may also be triggered internally by our code (through special phrases, see below).
(2) For the many different, synonymous phrases the player can enter to achieve the same effect, only a single standardized action will be created.
Example: entering "take ivory key", "take key", or "get key" will all produce the same, uniform action "taking the ivory key" (assuming that there is no other key around that might cause an ambiguity).

<verb in present participle> <noun> <noun>
in <room> **in the presence of** <person>
while/when <condition> **during** <scene>
for the <nth> **time** **for** <n> **turns**

doing something other than <action> [all actions except <action>]

taking the ivory key
taking something
eating in the Gardens
eating something in the presence of Lady Bracknell
examining the key for the second time [spelling n out is ok up to 12]
examining the key for the 20th time
examining the key for 2 turns [only if consecutive]
waiting for **four to six** turns
waiting for **more than 13** turns

Going

The action "going" happens when the player's character moves between rooms. Some special forms simplify common cases:

going from <room> [when leaving the room]
going to <room> [when entering the room]
going from <room> **to** <room> [when entering from 2nd room]
going through <door>

going <direction> **from** <room> [only if route exists!!]
going <direction> **in** <room> [also if route does not exists!!]

going nowhere from <room> [player tries unmapped route]
(note the "from" here, which is inconsistent with previous lines)

going … by <vehicle> [only if in the vehicle]

going … with <thing> [if pushing a thing btw. rooms]

exiting from <vehicle or container or supporter>

Kinds of actions

<action> **is** <action kind name> **.**

Kissing Mr Carr is unmaidenly behavior.
Doing something to the painting is unmaidenly behavior.
Instead of unmaidenly behaviour in the Inn, say "How unmaidenly!"

Phrases related to actions

try <action> [perform action as if user typed command]
try silently <action> [same, but no output on success, only if failed]

Custom Actions

<action name> **is an action applying to nothing .**

<action name> **is an action applying to one visible / touchable / carried** <kind or type> **requiring light .**

<action name> <propositions> **is an action applying to two visible / touchable / carried** <kind/type> **requiring light .**
<action name> <propositions> **is an action applying to one visible / touchable / carried** <kind/type> **and one visible / touchable / carried** <kind/type> **requiring light .**

Use **it** *in the list of propositions to denote the direct object.*

Blinking is an action applying to nothing.
Photographing is an action applying to one visible thing and requiring light.
Scraping it with is an action applying to two things.
Adjusting it to is an action applying to one thing and one number.

Understanding

Tells Inform7 how to map commands entered by the player to actions.

Understand <string> **as** <action> **(with nouns reversed) .**

<string> must start with a definite word but may contain substitutions afterwards. The substitutions must match the action's objects in number and type:

Understand "blink" as blinking.
Understand "photograph [someone]" as photographing.
Understand "photograph [an open door]" as photographing.
Understand "photograph" as photographing. [error, too few objects]
Understand "photograph [someone] standing/sitting next to [something]" as photographing. [error, too many objects]
Understand "adjust [something] to [a number]" as adjusting it to.

Understand "deposit [something] in [a container]" as inserting it into.
Understand "fill [a container] with [something]" as inserting it into (with nouns reversed).

Introducing synonyms and discarding commands:

Understand <word> **as the plural of** <object> **.**
Understand <word> **as** <kind/value> **.**
Understand <word> **as** <new token> **.** [creates new token]
Understand the command <word> **as** <word> **.**
Understand the command <word> **as something new.**

Understand "dog" as the St Bernard.
Understand "birds" as the plural of duck. [in addition to "ducks"]
Understand "machine" as a device.
Understand "eleventy-one" as 111.
Understand "by/near/beside" as "[beside]". [creates new token]
Understand the command "access" as "open".

More sophisticated substitutions, a.k.a. tokens:

"… in/inside/into …"	alternatives for a single word
[something]	match anything within reach/sight
[any <thing>**]**	match also things out of reach/sight
[something preferably held]	resolve ambiguity by pref. held things
[things]	like [something], but accepts a list of things or a vague plural like "all"
[things inside] … [something]	match only things inside 2nd object
[other things] … [something]	match only other things than 2nd obj.
[text]	matches any raw text; no parsing
[something related by reversed <relation>**]**	matches things related to (e.g. contained in) the understood kind

Understand "put [other things] in/into [something]" as inserting it into.
Understand "help on [text]" as getting help about.

A box is a kind of container. Understand "box of [something related by containment]" as a box. Red box is a box. Some crayons are in the red box. [--> now player can type TAKE BOX OF CRAYONS]

Understanding things by their property:

Understand the <property> **property as describing** <kind/object> **.**

The china pot can be unbroken or broken. The china pot is unbroken. Understand the unbroken property as describing the pot. [-> now player can type TAKE UNBROKEN POT, but only if pot is unbroken]

Relations

Relations are used to track how certain objects are related to others.

Built-in relations

Relation	Verb	Preposition
equality relation	**to be**	**to be**
containment relation	**to contain**	**to be in**
support relation	**to support**	**to be on**
carrying relation	**to carry**	**to be carried by**
wearing relation	**to wear**	**to be worn by**
possession relation *= carrying OR wearing*	**to have**	**to have**
incorporation relation	**to incorporate**	**to be part of**
visibility	**to be able to see**	
touchability	**to be able to touch**	
adjacency	**to be adjacent to** **to be** <direction> **of**	**to be** <direction> **of**

Custom relations

<relation name> **relates one/various** <kind> **to one/various** <kind> **with fast route-finding.**
<relation name> **relates one/various** <kind> **to another / each other in groups with fast route-finding.**

Loving relates various people to one person.
Marriage relates one person to another. [reciprocal relation]
Meeting relates people to each other. [reciprocal relation]
Nationality relates people to each other in groups. [in-group relation]

Reciprocal is what mathematicians call reflexive; "in groups" is transitive.

Relations can be created with a definition of which objects are actually related to which other objects:

<relation name> **relates one/various** <kind> **(called** <variable>**) to one/various** <kind> **(called** <variable>**) when** <condition>**.**

Contact relates a thing (called X) to a thing (called Y) when X is part of Y or Y is part of X.

How to use relations

Relations cannot be used directly; instead, you have to define verbs for them that formulate yes/no questions about the relation of two objects.

The verb <infinitive> **(he** <present singular>**, they** <present plural>**, he** <past>**, it is** <past participle>**, he is** <present participle>**) implies the reversed** <relation name> **relation.**

reversed = subject and object of verb are mapped reversely to relation.

The verb to sport (he sports, they sport, he sported, it is sported, he is sporting) implies the wearing relation.
The verb to grace (he graces, they grace, he graced, it is graced, it is gracing) implies the reversed wearing relation.
The verb to cover oneself with (he covers himself with, they cover themselves with, …) implies the wearing relation.

Now we can write in the source:

Mr Wickham sports a Tory rosette.
A Tory rosette graces Mr Wickham.
Peter is covering himself with a tent-like raincoat.

Also, the existing verb "to be" can be augmented to ask about a relation:

The verb to be <preposition> **implies the** <relation name> **relation.**

The verb to be suspicious of implies the suspecting relation.

Now we can write in the source:

Hercule Poirot is suspicious of Colonel Hotchkiss.

Where to use relations

(1) in assertions
(2) in conditions

Phrases for relations

next step via <relation> **from** <object> **to** <object>
number of steps via <relation> **from** <object> **to** <object>

These are special to value relations (i.e. non-object on one or both sides):

<kind> **to which/whom** <value> **relates by** <value relation>
<kind> **that/which/whom** <value> **relates to by** <value relation>

<kind> **that/which/who relates to** <value> **by** <value relation>

list of <kind> **to which/whom** <value> **relates by** <value relation>
list of <kind> **that/which/whom** <value> **relates to by** <valuerelation>

list of <kind> **that/which/who relate to** <value> **by** <value relation>

... (and more; cf. Writing with Inform, Chapter 13.13)

Tables

Creating a table / list

Table <number> / **Table of** <name> / **Table** <number> – <name>
<name column #1> **TAB** <name column #2> ...
<value> / – – / <kind>* **TAB** <value> / – – / <kind>* ...
...
with <number> **blank rows**

TAB = one or more tab chars ; * <kind> only in first row of empty column

Table 2.1 - Selected Elements

Element	Symbol	Atomic number	Atomic weight
"Hydrogen"	"H"	1	1
"Iron"	"Fe"	26	56
"Zinc"	"Zn"	30	65
"Uranium"	"U"	92	238

Referring to entries of a table

Referring to a table (written as <table> below):

the Table <number> / **of** <name>

the Table 2.1
the Table of Selected Elements

Referring to entries in a table (written as <table entry> below):

<column name> **in row** <number> **of** <table>

<column name> **of** <value> **in** <table>

<column name> **corresponding to** <column name> **of** <value>
in <table>

Choose a/the row <number> **in/from** <table>.
Choose a/the row with <column name> **of** <value> **in/from** <table>.
Choose a/the blank row in/from <table>.
Choose a/the random row in/from <table>.

After choosing a row, we can refer to an entry simply with:

<column name> **entry**

Reading from & writing to a table

As usual, you can use these <table entry> references to read a value from the table or write a value to a table:

<table entry> [just denoting an entry retrieves its value]
<table entry> **is** <value>. ["is" changes the entry's value]

symbol in row 3 of Table 2.1 [retrieves value "Zn"]
symbol in row 3 of Table 2.1 is "Cn" [replaces "Zn" with "Cn"]
element corresponding to symbol of "Fe" in Table 2.1 [retrieves "Iron"]

number of blank/filled rows in/from Table <number> / **of** <name>

repeat through <table> : ...
repeat through <table> **in reverse order** : ...
repeat through <table> **in reverse** <column name> **order** : ...

blank out <table entry>
blank out the whole row / the whole <column name> **column in Table** <number> / **of** <name>
blank out the whole of Table <number> / **of** <name>

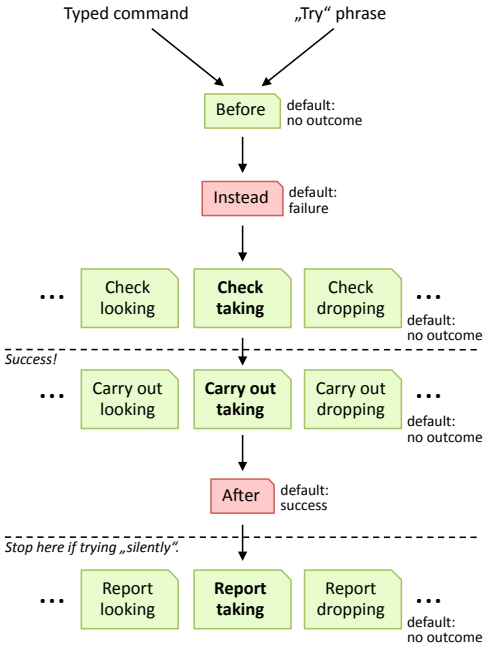
Defining objects and values with tables

<description> **are defined by** <table>.
<kind of value>**s are defined by** <table>.

See Writing with Inform, Chapters 15.16, 17 for details.

Main Cycle

A game of interactive fiction can be seen as an endless cycle of processing actions (either triggered by player-typed commands or by "try" phrases in the source) resulting in messages being printed out and/or changes being effected to the game world.



Each box represents a rulebook being followed during action processing. Red boxes indicate rulebooks with a default outcome (i.e. matching rules will stop the action by default), green boxes indicate rulebooks with no default outcome (i.e. matching rules will continue the action by default).

Miscellaneous

Names

Names may contain spaces:
The launching base is a room.

In names of rules, rulebooks, actions etc. the keyword **rule**, **rulebook**, **action**, resp. is optional if not required to avoid ambiguities:
the advance time rule = advance time
the instead rules = the instead rulebook = instead

Conditions

rule succeeded/failed	true if most recently followed rule ended in success / failure
outcome of the rulebook is <named outcome> **outcome**	true if the outcome of the most recently followed rulebook is <named outcome>
<value> **relates to** <kind> **by** <value relation>	
<kind> **relates to** <value> **by** <value relation>	
there is a <table entry> **there is no** <table entry>	
<table name> **is empty**	
the <object> **is a** <column name> **listed in Table** <number> / **of** <name>	

What else ... ?

This Cheat Sheet intends to illustrate the basic structures in the Inform7 language, not to provide a comprehensive list of all available phrases, rules, etc. However, the Inform7 index provides such lists, esp.:
Phrasebook Index
Rules Index
Actions Index