SR3 - Activities A/sr3

Purpose

The built-in activities and their default stock of rules; the locale description mechanism.

A/sr3.§21 Locale activities; §22 Locale Implementation; §23 Printing the Locale Description; §24 Choosing Notable Locale Objects; §25 Printing a Locale Paragraph

---

§**1.** These must not be created until the basic rulebooks are in place, because creating any activity automatically creates three rulebooks as well.

Once again, there are no activities or assumptions about them built into NI, but the template I6 layer expects the following 26 activities to be created and in this order. (That is, the order here must exactly match that of the `*_ACT` constant definitions made in `Definitions.i6t`.) The activities are fairly completely described in the Inform documentation, so the only notes here concern implementation.

§**2.** We start with activities used in describing things and rooms.

Most of the activities begin with all three rulebooks empty, since by default they do not intervene. This one, however, has a "last" for-rule, because it's required to do something definite: it must actually print a name, and the last for-rule is the default way to do this. (The assumption is that any other for-rule added by the user will halt the rulebook before the default implementation is reached.) We also include a before rule which marks any item whose name is being printed with the "mentioned" property, for reasons to be found below.

```
Part SR3 - Activities

Section SR3/1 - Definitions

Printing the name of something (documented at act_pn) is an activity. [0]

Before printing the name of a thing (called the item being printed)
    (this is the make named things mentioned rule):
    now the item being printed is mentioned.

The standard name printing rule is listed last in the for printing the name rulebook.
The standard name printing rule translates into I6 as "STANDARD_NAME_PRINTING_R".

Printing the plural name of something (documented at act_ppn) is an activity. [1]

Rule for printing the plural name of something (called the item) (this is the standard
    printing the plural name rule):
    say the printed plural name of the item.
The standard printing the plural name rule is listed last in the for printing
the plural name rulebook.

Printing a number of something (documented at act_pan) is an activity. [2]

Rule for printing a number of something (called the item) (this is the standard
    printing a number of something rule):
    say "[listing group size in words] ";
    carry out the printing the plural name activity with the item.
The standard printing a number of something rule is listed last in the for printing
a number rulebook.
```

§**3.** When they occur in room descriptions, names of things are sometimes supplemented by details:

```
Printing room description details of something (documented at act_details) is an activity. [3]
```

§**4.** Names of things are often formed up into lists, in which they are sometimes grouped together:

```
Listing contents of something (documented at act_lc) is an activity. [4]
The standard contents listing rule is listed last in the for listing contents rulebook.
The standard contents listing rule translates into I6 as "STANDARD_CONTENTS_LISTING_R".
Grouping together something (documented at act_gt) is an activity. [5]
```

§**5.** And such lists of names are formed up in turn into room descriptions. Something which is visible in a room can either have a paragraph of its own or can be relegated to the list of "nondescript" items at the end.

```
Writing a paragraph about something (documented at act_wpa) is an activity. [6]
```

§**6.** When these paragraphs have all gone by, the nondescript items left over are more briefly listed: the following activity gets the chance to change how this is done.

```
Listing nondescript items of something (documented at act_lni) is an activity. [7]
```

§**7.** Darkness behaves, for room description purposes, a little as if it were a room in its own right. Until the 1990s that was almost always how darkness was implemented in IF programs: this persists in I6, but not I7, where the existence of a room-which-is-not-a-room would break type safety.

```
Printing the name of a dark room (documented at act_darkname) is an activity. [8]
Printing the description of a dark room (documented at act_darkdesc) is an activity. [9]
Printing the announcement of darkness (documented at act_nowdark) is an activity. [10]
Printing the announcement of light (documented at act_nowlight) is an activity. [11]
Printing a refusal to act in the dark (documented at act_toodark) is an activity. [12]
The look around once light available rule is listed last in for printing the
announcement of light.
This is the look around once light available rule:
    try looking.
```

§**8.** Two special forms of printing: the status line at the top of the screen, refreshed every turn during play, and the banner which appears at or close to the start of play:

```
Constructing the status line (documented at act_csl) is an activity. [13]
Printing the banner text (documented at act_banner) is an activity. [14]
```

§**9.** Now a brace of activities to intervene in how the Inform parser does its parsing, arranged roughly in chronological order of their typical use during a single turn of a typed command, its parsing, and final conversion into an action.

The unusual notation "(future action)" here allows NI to parse rule preambles for these activities in a way which would refer to the action which might, at some point in the future, be generated – during parsing we don't of course yet know what that action is, but there is always a current guess at what it might be.

```
Reading a command (documented at act_reading) is an activity. [15]
Deciding the scope of something (future action) (documented at act_ds) is an activity. [16]
Deciding the concealed possessions of something (documented at act_con) is an activity. [17]
Deciding whether all includes something (future action) (documented at act_all)
    is an activity. [18]
The for deciding whether all includes rules have outcomes it does not (failure) and
    it does (success).
Clarifying the parser's choice of something (future action) (documented at act_clarify)
    is an activity. [19]
Asking which do you mean (future action) (documented at act_which) is an activity. [20]
Printing a parser error (documented at act_parsererror) is an activity. [21]
Supplying a missing noun (documented at act_smn) is an activity. [22]
Supplying a missing second noun (documented at act_smn) is an activity. [23]
Implicitly taking something (documented at act_implicitly) is an activity. [24]
```

§**10.** Here are the default rules for the behaviour of ALL:

```
Rule for deciding whether all includes scenery while taking (this is the
    exclude scenery from take all rule): rule fails.
Rule for deciding whether all includes people while taking (this is the
    exclude people from take all rule): rule fails.
Rule for deciding whether all includes fixed in place things while taking (this
    is the exclude fixed in place things from take all rule): rule fails.
```

§**11.** The supplying activities are linguistically interesting, for reasons gone into in the paper *Interactive Fiction, Natural Language and Semantic Analysis*: English verbs do not naturally seem to feature optional nouns. Indeed, we say "it rained on Tuesday" where "it" refers to nothing at all, merely because we can't bring ourselves to leave a gap and say just "rained on Tuesday". A better example here would be "it sounded like rain", because we do the same to convey the idea of listening ambiently rather than to any single thing: listening appears to be rare among actions in that it can equally well take a noun as not. Just as English handles this problem by supplying a spurious "it" which appears to mean "the world at large", so Inform handles it by supplying the current location, with the same idea in mind. And the same applies to the sense of smell, which can be similarly defocused.

```
Rule for supplying a missing noun while an actor smelling (this is the ambient odour rule):
    now the noun is the location.
Rule for supplying a missing noun while an actor listening (this is the ambient sound rule):
    now the noun is the location.
```

§**12.** The following rule is something of a dodge to provide a better parser response to commands like GO or BRETT, GO. (Putting the rule here, and giving it a name, allows the user to override it and thus accept the idea of vague going after all.)

```
Rule for supplying a missing noun while an actor going (this is the block vaguely going rule):
    issue library message going action number 7.
```

§**13.** The very first rules: those invoked at the earliest possible moment when the virtual machine is starting up. We need this hook, really, for the Glulx VM, which requires various styles to be created.

```
Starting the virtual machine (documented at act_startvm) is an activity. [25]
The enable Glulx acceleration rule is listed first in for starting the virtual machine.
The enable Glulx acceleration rule translates into I6 as "ENABLE_GLULX_ACCEL_R".
```

§**14.** And the very last rules of all. The obituary is a rare example of a sequence of events in the I6 library having been rolled up into an activity, partly because it's one of the few clear-cut moments where several unconnected things happen in succession.

```
Amusing a victorious player (documented at act_amuse) is an activity. [26]
Printing the player's obituary (documented at act_obit) is an activity. [27]
The print obituary headline rule is listed last in for printing the player's obituary.
The print final score rule is listed last in for printing the player's obituary.
The display final status line rule is listed last in for printing the player's obituary.
The print obituary headline rule translates into I6 as "PRINT_OBITUARY_HEADLINE_R".
The print final score rule translates into I6 as "PRINT_FINAL_SCORE_R".
The display final status line rule translates into I6 as "DISPLAY_FINAL_STATUS_LINE_R".
```

§**15.** There is one last question: the one which usually reads "Would you like to RESTART, RESTORE a saved game, or QUIT?", but which sometimes provides other options too. The "ask the final question rule" handles this, and does so by repeatedly calling the following activity:

```
Handling the final question is an activity. [28]
```

§**16.** It follows that this activity *must* at least sometimes do something dramatic to the execution state: perform a quit, for instance. Four primitive rules are available for the drastic things which the activity might wish to do, but these are not placed in any rulebook: instead they are available for anyone who wants to call them. (In the default implementation below, we put references to them into a table.)

```
The immediately restart the VM rule translates into I6 as "IMMEDIATELY_RESTART_VM_R".
The immediately restore saved game rule translates into I6 as "IMMEDIATELY_RESTORE_SAVED_R".
The immediately quit rule translates into I6 as "IMMEDIATELY_QUIT_R".
The immediately undo rule translates into I6 as "IMMEDIATELY_UNDO_R".
```

§**17.** We structure the activity so that the printing of the question and typing of the answer take place at the "before" stage, and then the parsing and acting upon this answer take place at the "for" stage. Reading the keyboard is the last rule in "before". With the "for" stage, the idea is that any extra rule slipped in by the user can take precedence over the default implementation, so the latter is the last there, too.

```
The print the final question rule is listed in before handling the final question.
The print the final prompt rule is listed in before handling the final question.
The read the final answer rule is listed last in before handling the final question.
The standard respond to final question rule is listed last in for handling the final question.

This is the print the final prompt rule: say "> [run paragraph on]".

The read the final answer rule translates into I6 as "READ_FINAL_ANSWER_R".
```

§**18.** That clears away the underbrush and reduces us to two matching tasks: (i) to print the question, (ii) to parse the answer, given that we want to be able to vary the set of choices available.

We do this by reading the options from the Table of Final Question Options. (See below for its default contents.) Each row is an option, whose wording must be placed in the topic column. The final question wording entry can either be text describing the option – e.g., "perform a RESTART" – or can be left blank, making the option a secret one, omitted from the question but still recognised as an answer. The only if victorious entry can be set to make the option available only after a victorious ending, not after a loss; Infocom's traditional AMUSING option behaved thus. Finally, the table specifies what to do if the option is taken: either it provides a rule, or an activity to carry out. (If it provides only an activity, but that activity is empty, then the option is omitted from the question and not recognised as an answer.)

```
Section SR3/2 - Final Question

This is the print the final question rule:
    let named options count be 0;
    repeat through the Table of Final Question Options:
        if the only if victorious entry is false or the story has ended finally:
            if there is a final response rule entry
                or the final response activity entry [activity] is not empty:
                if there is a final question wording entry, increase named options count by 1;
    if the named options count is less than 1, abide by the immediately quit rule;
    say "Would you like to ";
    repeat through the Table of Final Question Options:
        if the only if victorious entry is false or the story has ended finally:
            if there is a final response rule entry
                or the final response activity entry [activity] is not empty:
                if there is a final question wording entry:
                    say final question wording entry;
                    decrease named options count by 1;
                    if the named options count is 0:
                        say "?[line break]";
                    otherwise if the named options count is 1:
                        if the serial comma option is active, say ",";
                        say " or ";
                    otherwise:
                        say ", ";
```

§**19.** And the matching rule to parse and respond to the answer:

```
This is the standard respond to final question rule:
    repeat through the Table of Final Question Options:
        if the only if victorious entry is false or the story has ended finally:
            if there is a final response rule entry
                or the final response activity entry [activity] is not empty:
                if the player's command matches the topic entry:
                    if there is a final response rule entry, abide by final response rule entry;
                    otherwise carry out the final response activity entry activity;
                    rule succeeds;
    issue miscellaneous library message number 8.
```

§**20.** The table of final options is the only material under the heading "Section SR/Q", to make it easy for users to replace with entirely different tables.

These settings are the traditional ones used by Inform since 1995 or so. The UNDO option has customarily been a "secret", though not much of one, since it somewhat cheapens the announcement of a calamity to be immediately offered the chance to reverse it: death, where is thy sting?

```
Section SR3/3 - Final question options

Table of Final Question Options
final question wording only if victorious topic final response rule final response activity
"RESTART" false "restart" immediately restart the VM rule --
"RESTORE a saved game" false "restore" immediately restore saved game rule --
"see some suggestions for AMUSING things to do" true "amusing" -- amusing a victorious player
"QUIT" false "quit" immediately quit rule --
"UNDO the last command" false "undo" immediately undo rule --
```

§**21. Locale activities.** A "locale description" is a segment of the text produced by LOOK: the "locale" is a clutch of objects at a given level in the object tree. Most room descriptions consist of a top line, a description of the place, and then a single (though often, as here, multi-paragraph) locale:

```
Sentier Le Corbusier
A coastal walk along the rocky shore between Nice and Menton.
```
now the locale for the room Sentier Le Corbusier:
```
A translucent jellyfish has been washed up by the waves.

You can also see a bucket and a spade here.
```

A locale typically contains a run of paragraphs specific to interesting items, especially those not yet picked up, followed by a paragraph which lists the "nondescript" items – those not given paragraphs of their own, such as the bucket and spade. (Some items, though – typically scenery, but also for instance the player – are not even nondescript and do not appear at all.) A locale can contain no interesting paragraphs, or no list of nondescript items, or can even contain neither: that is, it can be entirely empty.

When the player is in or on top of something, multiple locales are described:

```
Sentier Le Corbusier (in the golf cart)
A coastal walk along the rocky shore between Nice and Menton.
```
now the locale for the room Sentier Le Corbusier:
```
A translucent jellyfish has been washed up by the waves.

You can also see a bucket and a spade here.
```
now the locale for the golf cart:
```
In the golf cart you can see a map of Villefranche-sur-Mer.
```

To sum up, the text produced by LOOK consists of a header (produced by the carry out looking rules) followed by one or more locale descriptions (produced by the activity below).

§**22. Locale Implementation.** When describing a locale, we keep a Table of interesting objects, each associated with a priority – a number indicating how important, and therefore how near to the top of the description, the object is. A special syntax allows us to create the Table with exactly the same number of rows as there are things in the model world: thus, in the worst case where all things are in a single locale, we still will not run out of table rows. (We do this rather than creating a large but fixed-size table because memory is very short in some Z-machine I7 works, so we want to take only what we might actually need. The table structure is not as wasteful as it might look: an experiment with using a number property of things instead showed that this table was actually more efficient, because of the property numbering overhead in the Z-machine memory representation of objects.)

```
Section SR3/4 - Locale descriptions - Unindexed

Table of Locale Priorities
notable-object (an object) locale description priority (a number)
-- --
with blank rows for each thing.

To describe locale for (O - object):
    carry out the printing the locale description activity with O.

To set the/-- locale priority of (O - an object) to (N - a number):
    if O is a thing:
        if N <= 0, now O is mentioned;
        if there is a notable-object of O in the Table of Locale Priorities:
            choose row with a notable-object of O in the Table of Locale Priorities;
            if N <= 0, blank out the whole row;
            otherwise now the locale description priority entry is N;
        otherwise:
```

```
        if N is greater than 0:
            choose a blank row in the Table of Locale Priorities;
            now the notable-object entry is O;
            now the locale description priority entry is N;
```

§**23. Printing the Locale Description.** This is handled by the "printing the locale description" activity. The before stage works out which objects might be of interest; the for stage actually prints paragraphs; the after stage is initially empty, but can be used to insert all kinds of interesting information into a room description.

We count the paragraphs printed in a global variable, not an activity variable, since it needs to be consulted in sub-activities whose rules are outside what would be its scope; that doesn't matter, though, since locale descriptions are not nested. (If they were, the above table would fail in any case.)

(1) Disaster would ensue if the user tampered with the "initialise locale description rule", but nobody is likely to do this other than intentionally.
(2) The "find notable locale objects rule" in fact only runs a further activity, the "choosing notable locale objects" activity. The task here is to identify the objects which might by virtue of their location appear in the locale, and to assign each of them a priority number.
(3) The "interesting locale paragraphs rule" goes through all of the notable objects chosen at stage (2), in order of priority, and offers each to yet another activity: the "printing a locale paragraph about" activity. This can either print a paragraph related to the item in question, or demote it as being not even nondescript (by changing its priority to 0). The default is to do nothing, in which case the item becomes nondescript.
(4) The "you-can-also-see rule" prints what is, ordinarily, the final paragraph of the locale description, listing the nondescript items. It goes to some trouble to find out whether these all have a common object tree parent, listing them with "list contents of" if they do: this is so that people who have written rules such as "Rule for printing the name of the blur while listing contents..." will take effect, because the "listing contents" activity will be going on. Provided that the notable objects chosen in (2) are all children of the locale domain, this will always happen. If the user should add rules to make quite different objects also notable, then the "you-can-also-see rule" has to resort to listing in a way which doesn't use the "listing contents" activity – since the list is not in fact a list of the contents of anything.

```
Printing the locale description of something (documented at act_pld) is an activity.

The locale paragraph count is a number that varies.

Before printing the locale description (this is the initialise locale description rule):
    now the locale paragraph count is 0;
    repeat with item running through things:
        now the item is not mentioned;
    repeat through the Table of Locale Priorities:
        blank out the whole row.

Before printing the locale description (this is the find notable locale objects rule):
    let the domain be the parameter-object;
    carry out the choosing notable locale objects activity with the domain;
    continue the activity.

For printing the locale description (this is the interesting locale paragraphs rule):
    let the domain be the parameter-object;
    sort the Table of Locale Priorities in locale description priority order;
    repeat through the Table of Locale Priorities:
        [say "[notable-object entry]...";]
        carry out the printing a locale paragraph about activity with the notable-object entry;
    continue the activity.

For printing the locale description (this is the you-can-also-see rule):
```

```
let the domain be the parameter-object;
let the mentionable count be 0;
repeat with item running through things:
    now the item is not marked for listing;
repeat through the Table of Locale Priorities:
    [say "[notable-object entry] - [locale description priority entry].";]
    if the locale description priority entry is greater than 0,
        now the notable-object entry is marked for listing;
    increase the mentionable count by 1;
if the mentionable count is greater than 0:
    repeat with item running through things:
        if the item is mentioned:
            now the item is not marked for listing;
    begin the listing nondescript items activity with the domain;
    if the number of marked for listing things is 0:
        abandon the listing nondescript items activity with the domain;
    otherwise:
        if handling the listing nondescript items activity:
            if the domain is a room:
                if the domain is the location, say "You ";
                otherwise say "In [the domain] you ";
            otherwise if the domain is a supporter:
                say "On [the domain] you ";
            otherwise if the domain is an animal:
                say "On [the domain] you ";
            otherwise:
                say "In [the domain] you ";
            say "can [if the locale paragraph count is greater than 0]also [end if]see ";
            let the common holder be nothing;
            let contents form of list be true;
            repeat with list item running through marked for listing things:
                if the holder of the list item is not the common holder:
                    if the common holder is nothing,
                        now the common holder is the holder of the list item;
                    otherwise now contents form of list is false;
                if the list item is mentioned, now the list item is not marked for listing;
            filter list recursion to unmentioned things;
            if contents form of list is true and the common holder is not nothing,
                list the contents of the common holder, as a sentence, including contents,
                    giving brief inventory information, tersely, not listing
                    concealed items, listing marked items only;
            otherwise say "[a list of marked for listing things including contents]";
            if the domain is the location, say " here";
            say ".[paragraph break]";
            unfilter list recursion;
        end the listing nondescript items activity with the domain;
continue the activity.
```

§**24. Choosing Notable Locale Objects.** By default, the notable objects are exactly the children of the domain, and they all have equal priority (1). Since table sorting is stable, and thus preserves the row order of rows with equal priority, the eventual order of listing is by default the same as the order in which things are added to the table, which in turn is the object-tree traversal order.

```
Choosing notable locale objects of something (documented at act_cnlo) is an activity.

For choosing notable locale objects (this is the standard notable locale objects rule):
    let the domain be the parameter-object;
    let the held item be the first thing held by the domain;
    while the held item is a thing:
        set the locale priority of the held item to 5;
        now the held item is the next thing held after the held item;
    continue the activity.
```

§**25. Printing a Locale Paragraph.** By default there are four kinds of "interesting" locale paragraph, and the following setup is fairly complicated because it implements conventions gradually built up between 1978 and 2008.

To recap, this activity is run on each notable thing in turn, in priority order. (It is only run on notable things for efficiency reasons.)

The basic principle is that, at every stage, we should consider an item only if it is not "mentioned" already. This will happen if it has been named by a previous paragraph, but also if it has been explicitly marked as such to get rid of it. In considering an item, we have three basic options:

(a) Print a paragraph about the item and mark it as mentioned – this is good for interesting items deserving a paragraph of their own.
(b) Print a paragraph, but do not mark it as mentioned – this is only likely to be useful if we want to print information related to the item without mentioning the thing itself. (For instance, if the presence of a mysterious parcel resulted in a ticking noise, we could print a paragraph about the ticking noise without mentioning the parcel, which would then appear later.)
(c) Mark the item as mentioned but print nothing – this gets rid of the item, ensuring that it will not appear in the final "you can also see" sentence, and will not be considered by subsequent rules.
(d) Do nothing at all – the item then becomes "nondescript" and appears in the final "you can also see" sentence, unless somebody else mentions it in the mean time.

Briefly, then, the following is the standard method:

(1) The "don't mention player's supporter in room descriptions rule" excludes anything the player is directly or indirectly standing on or, less frequently, in. The header of the room description has probably already said something like "Boudoir (on the four-poster bed)", so the player can't be unaware of this item.
(2) The "don't mention scenery in room descriptions rule" excludes scenery.
(3) The "don't mention undescribed items in room descriptions rule" excludes the player object. (It's redundant to say "You can also see yourself here.") At present nothing else in I7 is "undescribed" in this sense.
(4) The "set pronouns from items in room descriptions rule" adjusts the meaning of pronouns like IT and HER to pick up items mentioned. Thus if a room description ends "Mme Tourmalet glares at you.", then HER would be adjusted to mean Mme Tourmalet.
(5) The "offer items to writing a paragraph about rule" gives the "printing a paragraph about" activity a chance to intervene. We detect whether it does intervene or not by looking to see if it has printed any text.
(6) The "use initial appearance in room descriptions rule" uses the initial appearance property of an object which has never been handled as a paragraph.
(7) The "describe what's on scenery supporters in room descriptions rule" is a somewhat controversial feature: whereas the rest of Inform's room description conventions are generally consensus, this one is much disliked by some users for its occasional inappropriateness. It prints text such as "On the

mantelpiece is a piece of chalk." for items which, like the mantelpiece, are scenery mentioned (we assume) in the main room description. (It is assumed that scenery supporters make their contents more prominently visible than scenery containers, which we do not announce the contents of.) The ability to modify, replace or abolish this rule was one of the main motivations to break room description up into activities in March 2008.

```
Printing a locale paragraph about something (documented at act_plp) is an activity.

For printing a locale paragraph about a thing (called the item)
    (this is the don't mention player's supporter in room descriptions rule):
    if the item encloses the player, set the locale priority of the item to 0;
    continue the activity.

For printing a locale paragraph about a thing (called the item)
    (this is the don't mention scenery in room descriptions rule):
    if the item is scenery, set the locale priority of the item to 0;
    continue the activity.

For printing a locale paragraph about a thing (called the item)
    (this is the don't mention undescribed items in room descriptions rule):
    if the item is undescribed:
        set the locale priority of the item to 0;
    continue the activity.

For printing a locale paragraph about a thing (called the item)
    (this is the set pronouns from items in room descriptions rule):
    if the item is not mentioned, set pronouns from the item;
    continue the activity.

For printing a locale paragraph about a thing (called the item)
    (this is the offer items to writing a paragraph about rule):
    if the item is not mentioned:
        if a paragraph break is pending, say "[conditional paragraph break]";
        carry out the writing a paragraph about activity with the item;
        if a paragraph break is pending:
            increase the locale paragraph count by 1;
            now the item is mentioned;
            say "[command clarification break]";
    continue the activity.

For printing a locale paragraph about a thing (called the item)
    (this is the use initial appearance in room descriptions rule):
    if the item is not mentioned:
        if the item provides the property initial appearance and the
            item is not handled and the initial appearance of the item is
            not "":
            increase the locale paragraph count by 1;
            say "[initial appearance of the item]";
            say "[paragraph break]";
            if a locale-supportable thing is on the item:
                repeat with possibility running through things on the item:
                    now the possibility is marked for listing;
                    if the possibility is mentioned:
                        now the possibility is not marked for listing;
                say "On [the item] ";
                list the contents of the item, as a sentence, including contents,
                    giving brief inventory information, tersely, not listing
                    concealed items, prefacing with is/are, listing marked items only;
```

```
            say ".[paragraph break]";
        now the item is mentioned;
    continue the activity.

Definition: a thing (called the item) is locale-supportable if the item is not
scenery and the item is not mentioned and the item is not undescribed.

For printing a locale paragraph about a thing (called the item)
    (this is the describe what's on scenery supporters in room descriptions rule):
    if the item is [not undescribed and the item is] scenery and
        the item does not enclose the player:
        if a locale-supportable thing is on the item:
            set pronouns from the item;
            repeat with possibility running through things on the item:
                now the possibility is marked for listing;
                if the possibility is mentioned:
                    now the possibility is not marked for listing;
            increase the locale paragraph count by 1;
            say "On [the item] ";
            list the contents of the item, as a sentence, including contents,
                giving brief inventory information, tersely, not listing
                concealed items, prefacing with is/are, listing marked items only;
            say ".[paragraph break]";
    continue the activity.
```