```
Help on module joker:

NAME
    joker - Carry out directives such as save, restore, and quit.

FILE
    c:\nickmontfort-curveship-814de6e\joker.py

CLASSES
    exceptions.Exception(exceptions.BaseException)
        StartupError

    class StartupError(exceptions.Exception)
     |  Exception occurring during session startup or in loading a spin.
     |
     |  Method resolution order:
     |      StartupError
     |      exceptions.Exception
     |      exceptions.BaseException
     |      __builtin__.object
     |
     |  Methods defined here:
     |
     |  __init__(self, msg)
     |
     |  ----------------------------------------------------------------------
     |  Data descriptors defined here:
     |
     |  __weakref__
     |      list of weak references to the object (if defined)
     |
     |  ----------------------------------------------------------------------
     |  Data and other attributes inherited from exceptions.Exception:
     |
     |  __new__ = <built-in method __new__ of type object>
     |      T.__new__(S, ...) -> a new object with type S, a subtype of T
     |
     |  ----------------------------------------------------------------------
     |  Methods inherited from exceptions.BaseException:
     |
     |  __delattr__(...)
     |      x.__delattr__('name') <==> del x.name
     |
     |  __getattribute__(...)
     |      x.__getattribute__('name') <==> x.name
     |
     |  __getitem__(...)
     |      x.__getitem__(y) <==> x[y]
     |
     |  __getslice__(...)
```

```
 |      x.__getslice__(i, j) <==> x[i:j]
 |
 |      Use of negative indices is not supported.
 |
 |  __reduce__(...)
 |
 |  __repr__(...)
 |      x.__repr__() <==> repr(x)
 |
 |  __setattr__(...)
 |      x.__setattr__('name', value) <==> x.name = value
 |
 |  __setstate__(...)
 |
 |  __str__(...)
 |      x.__str__() <==> str(x)
 |
 |  __unicode__(...)
 |
 |  ----------------------------------------------------------------------
 |  Data descriptors inherited from exceptions.BaseException:
 |
 |  __dict__
 |
 |  args
 |
 |  message

FUNCTIONS
    comment(_, world, discourse)
        A comment has been entered. Just continue.

    concept_info(tokens, world, discourse)
        Describes items or actions based on a particular actor's concept.

    count_commands(_, world, discourse)
        Returns a report on the number of commands issued so far.

    count_directives(_, world, discourse)
        Return a report on the number of directives issues so far.

    count_unrecognized(_, world, discourse)
        Returns a report on the number of unrecognized inputs so far.

    exits(_, world, discourse)
        Lists the exits from the focalizer's current room.

    inputs(tokens, world, discourse)
        Returns a report listing all requested inputs.
```

```
joke(tokens, world, discourse)
    Handles directives -- inputs that deal with the program state.

light(_, world, discourse)
    Returns a report on the focalizer's compartment's light level.

load_fiction(file_name, required, defaults)
    Loads a fiction file.

load_file(file_name, required, defaults, module_type)
    Loads either an interactive fiction or a spin file.

    Improved filename parsing thanks to Max Battcher.

load_spin(existing_spin, spin_file)
    Loads one spin file and returns an updated spin.

narrating(tokens, world, discourse)
    Returns a report describing the current spin.

narrating_commanded(tokens, world, discourse)
    Changes the commanded actor.

narrating_dynamic(tokens, world, discourse)
    Changes whether the spin is dynamic.

narrating_focalizer(tokens, world, discourse)
    Changes the focalizing actor.

narrating_narratee(tokens, world, discourse)
    Changes which actor (if any) is the narratee.

narrating_narrator(tokens, world, discourse)
    Changes which actor (if any) is the narrator.

narrating_order(tokens, world, discourse)
    Changes the order.

narrating_perfect(tokens, world, discourse)
    Changes whether narration is in the perfect by default.

narrating_player(tokens, world, discourse)
    Changes the player character (both focalizer and commanded).

narrating_progressive(tokens, world, discourse)
    Changes whether narration is in the progressive by default.

narrating_speed(tokens, world, discourse)
    Changes the default speed of narration.
```

```
narrating_time(tokens, world, discourse)
    Changes the narrator's position in time relative to events.

narrating_timewords(tokens, world, discourse)
    Turns time_words on or off.

narrating_uses(tokens, world, discourse)
    Loads a new spin (parameters for telling) from a file.

prologue(_, world, discourse)
    Returns a reply containing the prologue.

recount(tokens, world, discourse)
    Returns a report and a reply with narration of previous events.

report(kind, *params)
    Prepare a report text using the MESSAGE dictionary.

restart(_, world, discourse)
    Restarts the game and emit an appropriate report.

restore(tokens, world, discourse)
    Restores the game and emit an appropriate report.

room_name(_, world, discourse)
    Give the name of the focalizer's current room.

save(tokens, world, discourse)
    Save the fiction/game/world and emit an appropriate report.

session_startup(version)
    Return strings to be presented, centered, as a session begins.

set_role(role, tokens, world, discourse)
    Name or change a narrative role, such as focalizer.

show_frontmatter(discourse)
    Return a string with the title, headline, and credits.

show_prologue(data)
    Return a string containing the prologue, if there is one.

terminate(_, world, discourse)
    Quits the game after emitting an appropriate report.

    Since 'quit' is a builtin function, this one is called 'terminate.'

ticks(_, world, discourse)
    Returns a report on how many ticks (time units) have passed.
```

```
    title(_, world, discourse)
        Returns a reply containing the frontmatter.

    undo(tokens, world, discourse)
        Undoes a turn and emits an appropriate report.

    update_spin(existing_spin, new_spin)

    wc_info(tokens, world_or_concept, world, discourse)
        Reports on the world or a concept: tree, item info, actions.

    world_info(tokens, world, discourse)
        Describes the world's items or actions.

DATA
    MESSAGE = {'are': '[] are the [].', 'concept_missing_item': "The item ...
    __author__ = 'Nick Montfort'
    __copyright__ = 'Copyright 2011 Nick Montfort'
    __license__ = 'ISC'
    __status__ = 'Development'
    __version__ = '0.5.0.0'

VERSION
    0.5.0.0

AUTHOR
    Nick Montfort
```