:mod:`readline` --- GNU readline interface

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]readline.rst, line 1); *backlink*

Unknown interpreted text role "mod".

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]readline.rst, line 4)

Unknown directive type "module".

```
.. module:: readline
   :platform: Unix
   :synopsis: GNU readline support for Python.
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]readline.rst, line 8)

Unknown directive type "sectionauthor".

```
.. sectionauthor:: Skip Montanaro <skip@pobox.com>
```

The :mod:`readline` module defines a number of functions to facilitate completion and reading/writing of history files from the Python interpreter. This module can be used directly, or via the :mod:`rlcompleter` module, which supports completion of Python identifiers at the interactive prompt. Settings made using this module affect the behaviour of both the interpreter's interactive prompt and the prompts offered by the built-in :func:`input` function.

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]readline.rst, line 12); *backlink*

Unknown interpreted text role "mod".

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]readline.rst, line 12); *backlink*

Unknown interpreted text role "mod".

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]readline.rst, line 12); *backlink*

Unknown interpreted text role "func".

Readline keybindings may be configured via an initialization file, typically `.inputrc` in your home directory. See Readline Init File in the GNU Readline manual for information about the format and allowable constructs of that file, and the capabilities of the Readline library in general.

Note

The underlying Readline library API may be implemented by the `libedit` library instead of GNU readline. On macOS the :mod:`readline` module detects which library is being used at run time.

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]readline.rst, line 29); *backlink*

Unknown interpreted text role "mod".

The configuration file for `libedit` is different from that of GNU readline. If you programmatically load configuration strings you can check for the text "libedit" in :const:`readline.__doc__` to differentiate between GNU readline and libedit.

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]readline.rst, line 34); *backlink*

Unknown interpreted text role "const".

If you use *editline*/`libedit` readline emulation on macOS, the initialization file located in your home directory is named `.editrc`. For example, the following content in `~/.editrc` will turn ON *vi* keybindings and TAB completion:

```
python:bind -v
python:bind ^I rl_complete
```

Init file

The following functions relate to the init file and user configuration:

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]readline.rst, line 54)

Unknown directive type "function".

```
.. function:: parse_and_bind(string)

   Execute the init line provided in the *string* argument. This calls
   :c:func:`rl_parse_and_bind` in the underlying library.
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]readline.rst, line 60)

Unknown directive type "function".

```
.. function:: read_init_file([filename])

   Execute a readline initialization file. The default filename is the last filename
   used. This calls :c:func:`rl_read_init_file` in the underlying library.
```

Line buffer

The following functions operate on the line buffer:

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]readline.rst, line 72)

Unknown directive type "function".

```
.. function:: get_line_buffer()

   Return the current contents of the line buffer (:c:data:`rl_line_buffer`
   in the underlying library).
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]readline.rst, line 78)

Unknown directive type "function".

```
.. function:: insert_text(string)

   Insert text into the line buffer at the cursor position.  This calls
   :c:func:`rl_insert_text` in the underlying library, but ignores
   the return value.
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]readline.rst, line 85)

Unknown directive type "function".

```
.. function:: redisplay()

   Change what's displayed on the screen to reflect the current contents of the
   line buffer.  This calls :c:func:`rl_redisplay` in the underlying library.
```

History file

The following functions operate on a history file:

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]readline.rst, line 97)

Unknown directive type "function".

```
.. function:: read_history_file([filename])

   Load a readline history file, and append it to the history list.
   The default filename is :file:`~/.history`.  This calls
   :c:func:`read_history` in the underlying library.
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]readline.rst, line 104)

Unknown directive type "function".

```
.. function:: write_history_file([filename])

   Save the history list to a readline history file, overwriting any
   existing file.  The default filename is :file:`~/.history`.  This calls
   :c:func:`write_history` in the underlying library.
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]readline.rst, line 111)

Unknown directive type "function".

```
.. function:: append_history_file(nelements[, filename])

   Append the last *nelements* items of history to a file.  The default filename is
   :file:`~/.history`.  The file must already exist.  This calls
   :c:func:`append_history` in the underlying library.  This function
   only exists if Python was compiled for a version of the library
   that supports it.

   .. versionadded:: 3.5
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]readline.rst, line 122)

Unknown directive type "function".

```
.. function:: get_history_length()
              set_history_length(length)

   Set or return the desired number of lines to save in the history file.
   The :func:`write_history_file` function uses this value to truncate
   the history file, by calling :c:func:`history_truncate_file` in
   the underlying library.  Negative values imply
   unlimited history file size.
```

History list

The following functions operate on a global history list:

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]readline.rst, line 138)

Unknown directive type "function".

```
.. function:: clear_history()

   Clear the current history.  This calls :c:func:`clear_history` in the
   underlying library.  The Python function only exists if Python was
   compiled for a version of the library that supports it.
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]readline.rst, line 145)

Unknown directive type "function".

```
.. function:: get_current_history_length()

   Return the number of items currently in the history.  (This is different from
   :func:`get_history_length`, which returns the maximum number of lines that will
   be written to a history file.)
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]readline.rst, line 152)

Unknown directive type "function".

```
.. function:: get_history_item(index)

   Return the current contents of history item at *index*.  The item index
   is one-based.  This calls :c:func:`history_get` in the underlying library.
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]readline.rst, line 158)

Unknown directive type "function".

```
.. function:: remove_history_item(pos)

   Remove history item specified by its position from the history.
   The position is zero-based.  This calls :c:func:`remove_history` in
   the underlying library.
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]readline.rst, line 165)

Unknown directive type "function".

```
.. function:: replace_history_item(pos, line)

   Replace history item specified by its position with *line*.
   The position is zero-based.  This calls :c:func:`replace_history_entry`
   in the underlying library.
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]readline.rst, line 172)

Unknown directive type "function".

```
.. function:: add_history(line)

   Append *line* to the history buffer, as if it was the last line typed.
   This calls :c:func:`add_history` in the underlying library.
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]readline.rst, line 178)

Unknown directive type "function".

```
.. function:: set_auto_history(enabled)

   Enable or disable automatic calls to :c:func:`add_history` when reading
   input via readline.  The *enabled* argument should be a Boolean value
   that when true, enables auto history, and that when false, disables
   auto history.

   .. versionadded:: 3.6

   .. impl-detail::
      Auto history is enabled by default, and changes to this do not persist
      across multiple sessions.
```

Startup hooks

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]readline.rst, line 196)

Unknown directive type "function".

```
.. function:: set_startup_hook([function])

   Set or remove the function invoked by the :c:data:`rl_startup_hook`
   callback of the underlying library.  If *function* is specified, it will
   be used as the new hook function; if omitted or ``None``, any function
   already installed is removed.  The hook is called with no
   arguments just before readline prints the first prompt.
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]readline.rst, line 205)

Unknown directive type "function".

```
.. function:: set_pre_input_hook([function])

   Set or remove the function invoked by the :c:data:`rl_pre_input_hook`
   callback of the underlying library.  If *function* is specified, it will
   be used as the new hook function; if omitted or ``None``, any
   function already installed is removed.  The hook is called
   with no arguments after the first prompt has been printed and just before
   readline starts reading input characters.  This function only exists
   if Python was compiled for a version of the library that supports it.
```

Completion

The following functions relate to implementing a custom word completion function. This is typically operated by the Tab key, and can suggest and automatically complete a word being typed. By default, Readline is set up to be used by :mod:`rlcompleter` to complete Python identifiers for the interactive interpreter. If the :mod:`readline` module is to be used with a custom completer, a different set of word delimiters should be set.

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]readline.rst, line 219); *backlink*

Unknown interpreted text role "mod".

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]readline.rst, line 219); *backlink*

Unknown interpreted text role "mod".

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]readline.rst, line 227)

Unknown directive type "function".

```
.. function:: set_completer([function])

   Set or remove the completer function.  If *function* is specified, it will be
   used as the new completer function; if omitted or ``None``, any completer
   function already installed is removed.  The completer function is called as
   ``function(text, state)``, for *state* in ``0``, ``1``, ``2``, ..., until it
   returns a non-string value.  It should return the next possible completion
   starting with *text*.

   The installed completer function is invoked by the *entry_func* callback
   passed to :c:func:`rl_completion_matches` in the underlying library.
   The *text* string comes from the first parameter to the
   :c:data:`rl_attempted_completion_function` callback of the
   underlying library.
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]readline.rst, line 243)

Unknown directive type "function".

```
.. function:: get_completer()

   Get the completer function, or ``None`` if no completer function has been set.
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]readline.rst, line 248)

Unknown directive type "function".

```
.. function:: get_completion_type()

   Get the type of completion being attempted.  This returns the
   :c:data:`rl_completion_type` variable in the underlying library as
   an integer.
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]readline.rst, line 255)

Unknown directive type "function".

```
.. function:: get_begidx()
              get_endidx()

   Get the beginning or ending index of the completion scope.
   These indexes are the *start* and *end* arguments passed to the
   :c:data:`rl_attempted_completion_function` callback of the
   underlying library.  The values may be different in the same
   input editing scenario based on the underlying C readline implementation.
   Ex: libedit is known to behave differently than libreadline.
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]readline.rst, line 266)

Unknown directive type "function".

```
.. function:: set_completer_delims(string)
              get_completer_delims()

   Set or get the word delimiters for completion.  These determine the
   start of the word to be considered for completion (the completion scope).
   These functions access the :c:data:`rl_completer_word_break_characters`
   variable in the underlying library.
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]readline.rst, line 275)

Unknown directive type "function".

```
.. function:: set_completion_display_matches_hook([function])

   Set or remove the completion display function.  If *function* is
   specified, it will be used as the new completion display function;
   if omitted or ``None``, any completion display function already
   installed is removed.  This sets or clears the
   :c:data:`rl_completion_display_matches_hook` callback in the
   underlying library.  The completion display function is called as
   ``function(substitution, [matches], longest_match_length)`` once
   each time matches need to be displayed.
```

Example

The following example demonstrates how to use the :mod:`readline` module's history reading and writing functions to automatically load and save a history file named :file:`.python_history` from the user's home directory. The code below would normally be executed automatically during interactive sessions from the user's :envvar:`PYTHONSTARTUP` file.

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]readline.rst, line 292); *backlink*

Unknown interpreted text role "mod".

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]readline.rst, line 292); *backlink*

Unknown interpreted text role "file".

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]readline.rst, line 292); *backlink*

Unknown interpreted text role "envvar".

```
import atexit
import os
import readline

histfile = os.path.join(os.path.expanduser("~"), ".python_history")
try:
    readline.read_history_file(histfile)
    # default history len is -1 (infinite), which may grow unruly
    readline.set_history_length(1000)
except FileNotFoundError:
    pass

atexit.register(readline.write_history_file, histfile)
```

This code is actually automatically run when Python is run in :ref:`interactive mode <tut-interactive>` (see :ref:`rlcompleter-config`).

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]readline.rst, line 312); *backlink*

Unknown interpreted text role "ref".

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]readline.rst, line 312); *backlink*

Unknown interpreted text role "ref".

The following example achieves the same goal but supports concurrent interactive sessions, by only appending the new history.

```
import atexit
```

```
import os
import readline
histfile = os.path.join(os.path.expanduser("~"), ".python_history")

try:
    readline.read_history_file(histfile)
    h_len = readline.get_current_history_length()
except FileNotFoundError:
    open(histfile, 'wb').close()
    h_len = 0

def save(prev_h_len, histfile):
    new_h_len = readline.get_current_history_length()
    readline.set_history_length(1000)
    readline.append_history_file(new_h_len - prev_h_len, histfile)
atexit.register(save, h_len, histfile)
```

The following example extends the :class:`code.InteractiveConsole` class to support history save/restore.

> **System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]readline.rst, line 336); *backlink***
>
> Unknown interpreted text role "class".

```
import atexit
import code
import os
import readline

class HistoryConsole(code.InteractiveConsole):
    def __init__(self, locals=None, filename="<console>",
                 histfile=os.path.expanduser("~/.console-history")):
        code.InteractiveConsole.__init__(self, locals, filename)
        self.init_history(histfile)

    def init_history(self, histfile):
        readline.parse_and_bind("tab: complete")
        if hasattr(readline, "read_history_file"):
            try:
                readline.read_history_file(histfile)
            except FileNotFoundError:
                pass
            atexit.register(self.save_history, histfile)

    def save_history(self, histfile):
        readline.set_history_length(1000)
        readline.write_history_file(histfile)
```