:mod:`code` --- Interpreter base classes

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

Unknown directive type "module".

```
.. module:: code
   :synopsis: Facilities to implement read-eval-print loops.
```

Source code: :source:`Lib/code.py`

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]code.rst`, line 7); *backlink*

Unknown interpreted text role "source".

The `code` module provides facilities to implement read-eval-print loops in Python. Two classes and convenience functions are included which can be used to build applications which provide an interactive interpreter prompt.

This class deals with parsing and interpreter state (the user's namespace); it does not deal with input buffering or prompting or input file naming (the filename is always passed in explicitly). The optional *locals* argument specifies the dictionary in which code will be executed; it defaults to a newly created dictionary with key `'__name__'` set to `'__console__'` and key `'__doc__'` set to `None`.

Closely emulate the behavior of the interactive Python interpreter. This class builds on :class:`InteractiveInterpreter` and adds prompting using the familiar `sys.ps1` and `sys.ps2`, and input buffering.

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]code.rst`, line 28); *backlink*

Unknown interpreted text role "class".

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]code.rst`, line 33)

Unknown directive type "function".

```
.. function:: interact(banner=None, readfunc=None, local=None, exitmsg=None)

   Convenience function to run a read-eval-print loop.  This creates a new
   instance of :class:`InteractiveConsole` and sets *readfunc* to be used as
   the :meth:`InteractiveConsole.raw_input` method, if provided.  If *local* is
   provided, it is passed to the :class:`InteractiveConsole` constructor for
   use as the default namespace for the interpreter loop.  The :meth:`interact`
   method of the instance is then run with *banner* and *exitmsg* passed as the
   banner and exit message to use, if provided.  The console object is discarded
   after use.

   .. versionchanged:: 3.6
      Added *exitmsg* parameter.
```

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]code.rst`, line 48)

Unknown directive type "function".

```
.. function:: compile_command(source, filename="<input>", symbol="single")

   This function is useful for programs that want to emulate Python's interpreter
   main loop (a.k.a. the read-eval-print loop).  The tricky part is to determine
   when the user has entered an incomplete command that can be completed by
   entering more text (as opposed to a complete command or a syntax error).  This
   function *almost* always makes the same decision as the real interpreter main
   loop.

   *source* is the source string; *filename* is the optional filename from which
   source was read, defaulting to ``'<input>'``; and *symbol* is the optional
   grammar start symbol, which should be ``'single'`` (the default), ``'eval'``
   or ``'exec'``.

   Returns a code object (the same as ``compile(source, filename, symbol)``) if the
   command is complete and valid; ``None`` if the command is incomplete; raises
   :exc:`SyntaxError` if the command is complete and contains a syntax error, or
   raises :exc:`OverflowError` or :exc:`ValueError` if the command contains an
   invalid literal.
```

Interactive Interpreter Objects

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]code.rst`, line 75)

Unknown directive type "method".

```
.. method:: InteractiveInterpreter.runsource(source, filename="<input>", symbol="single")

   Compile and run some source in the interpreter. Arguments are the same as for
   :func:`compile_command`; the default for *filename* is ``'<input>'``, and for
   *symbol* is ``'single'``.  One of several things can happen:

   * The input is incorrect; :func:`compile_command` raised an exception
     (:exc:`SyntaxError` or :exc:`OverflowError`).  A syntax traceback will be
     printed by calling the :meth:`showsyntaxerror` method.  :meth:`runsource`
     returns ``False``.

   * The input is incomplete, and more input is required; :func:`compile_command`
     returned ``None``. :meth:`runsource` returns ``True``.

   * The input is complete; :func:`compile_command` returned a code object.  The
     code is executed by calling the :meth:`runcode` (which also handles run-time
     exceptions, except for :exc:`SystemExit`). :meth:`runsource` returns ``False``.

   The return value can be used to decide whether to use ``sys.ps1`` or ``sys.ps2``
   to prompt the next line.
```

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

Unknown directive type "method".

```
.. method:: InteractiveInterpreter.runcode(code)

   Execute a code object. When an exception occurs, :meth:`showtraceback` is called
   to display a traceback.  All exceptions are caught except :exc:`SystemExit`,
   which is allowed to propagate.

   A note about :exc:`KeyboardInterrupt`: this exception may occur elsewhere in
   this code, and may not always be caught.  The caller should be prepared to deal
   with it.
```

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]code.rst`, line 108)

Unknown directive type "method".

```
.. method:: InteractiveInterpreter.showsyntaxerror(filename=None)

   Display the syntax error that just occurred.  This does not display a stack
   trace because there isn't one for syntax errors. If *filename* is given, it is
   stuffed into the exception instead of the default filename provided by Python's
   parser, because it always uses ``'<string>'`` when reading from a string. The
   output is written by the :meth:`write` method.
```

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]code.rst`, line 117)

Unknown directive type "method".

```
.. method:: InteractiveInterpreter.showtraceback()

   Display the exception that just occurred.  We remove the first stack item
   because it is within the interpreter object implementation. The output is
   written by the :meth:`write` method.

   .. versionchanged:: 3.5 The full chained traceback is displayed instead
      of just the primary traceback.
```

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]code.rst`, line 127)

Unknown directive type "method".

```
.. method:: InteractiveInterpreter.write(data)

   Write a string to the standard error stream (``sys.stderr``). Derived classes
   should override this to provide the appropriate output handling as needed.
```

Interactive Console Objects

The :class:`InteractiveConsole` class is a subclass of :class:`InteractiveInterpreter`, and so offers all the methods of the interpreter objects as well as the following additions.

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

Unknown interpreted text role "class".

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

Unknown interpreted text role "class".

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]code.rst`, line 143)

Unknown directive type "method".

```
.. method:: InteractiveConsole.interact(banner=None, exitmsg=None)

   Closely emulate the interactive Python console. The optional *banner* argument
   specify the banner to print before the first interaction; by default it prints a
   banner similar to the one printed by the standard Python interpreter, followed
   by the class name of the console object in parentheses (so as not to confuse
   this with the real interpreter -- since it's so close!).

   The optional *exitmsg* argument specifies an exit message printed when exiting.
   Pass the empty string to suppress the exit message. If *exitmsg* is not given or
   ``None``, a default message is printed.

   .. versionchanged:: 3.4
      To suppress printing any banner, pass an empty string.

   .. versionchanged:: 3.6
      Print an exit message when exiting.
```

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]code.rst`, line 162)

Unknown directive type "method".

```
.. method:: InteractiveConsole.push(line)

   Push a line of source text to the interpreter. The line should not have a
   trailing newline; it may have internal newlines.  The line is appended to a
   buffer and the interpreter's :meth:`runsource` method is called with the
   concatenated contents of the buffer as source.  If this indicates that the
   command was executed or invalid, the buffer is reset; otherwise, the command is
   incomplete, and the buffer is left as it was after the line was appended.  The
   return value is ``True`` if more input is required, ``False`` if the line was
   dealt with in some way (this is the same as :meth:`runsource`).
```

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]code.rst`, line 174)

Unknown directive type "method".

```
.. method:: InteractiveConsole.resetbuffer()

   Remove any unhandled source text from the input buffer.
```

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\cpython-main\Doc\library\[cpython-main][Doc][library]code.rst`, line 179)

Unknown directive type "method".

```
.. method:: InteractiveConsole.raw_input(prompt="")

   Write a prompt and read a line.  The returned line does not include the trailing
   newline.  When the user enters the EOF key sequence, :exc:`EOFError` is raised.
   The base implementation reads from ``sys.stdin``; a subclass may replace this
   with a different implementation.
```