XTraceback Documentation

Release 0.4.0-rc1

Arthur Noel

May 03, 2014

Contents

Warning: XTraceback is not tested on Windows systems (patches very welcome).

CHAPTER 1

Installation

To install using pip:

```
pip install xtraceback
```

To install with syntax highlighting provided by Pygments:

```
pip install "xtraceback[syntax]"
```

CHAPTER 2

Documentation

2.1 Stdlib Compatibility

XTraceback works as a drop-in replacement for the standard library's `traceback` module and can patch `logging.LogFormatter.formatException()`

XTraceback should be installed as early as possible in your code, or alternatively in a `site` *sitecustomize* module.

2.1.1 traceback

```
>>> import xtraceback
>>> xtraceback.compat.install()
```

This installs `xtraceback.StdlibCompat.print_exception()` as a `sys.excepthook` and patches the `traceback` module replacing the following functions with extended versions from `xtraceback.StdlibCompat`:

- `traceback.format_tb()`
- `traceback.format_exception_only()`
- `traceback.format_exception()`
- `traceback.format_exc()`
- `traceback.print_tb()`
- `traceback.print_exception()`
- `traceback.print_exc()`

2.1.2 logging

```
>>> import logging
>>> import xtraceback
>>> logging.basicConfig()
>>> xtraceback.compat.install_logging(logging.root.handlers[0])
```

2.2 Configuration

For options and their defaults see `xtraceback.XTracebackOptions`. When using stdlib compatibility the `xtraceback.StdlibCompat` *defaults* dictionary should be updated with your overrides - the default instance exists at `xtraceback.compat`:

```
xtraceback.compat.defaults.update(option=value[, ...])
```

2.3 Examples

As a context manager - the stdlib traceback module is monkey patched.

As a sys.excepthook.

```
>>> xtraceback.compat.install_sys_excepthook()
```

In a sitecustomize module.

```
import xtraceback
xtraceback.compat.install()
```

2.4 API

class `xtraceback.`**`XTraceback`**(*etype*, *value*, *tb*, ***options*)

An extended traceback formatter

Parameters

- **etype** (*type*) – The exception type
- **value** (*Exception*) – The exception instance
- **tb** (*traceback*) – The traceback instance
- **options** (*dict*) – Options for this instance

`options` = **None**

Options for xtraceback

`etype` = **None**

The exception type

`value` = **None**

The exception value (instance)

`number_padding` = **None**

Used in XTracebackFrame to determine indent

`tty_stream`

Whether or not our stream is a tty

`color`

Whether or not color should be output

`_highlight`(*string*)

`_str_lines`(*lines*)

_format_lines(*lines*)

_print_lines(*lines*)

_format_tb()

_format_exception_only()

_format_exception()

format_tb()

format_exception_only()

format_exception()

print_tb()

print_exception()

class xtraceback.**XTracebackOptions**(***options*)

XTraceback options

Variables

- **stream** – A file-like object that is the default for print_* methods
- **color** – Flag to force color on or off - if None look to whether the *stream* is a tty
- **print_width** – How many columns wide to print the screen - if None and *stream* is a tty on Unix then fill the available width
- **offset** – A stack frame offset - defaults to 0
- **limit** – Stack frame limit - if None the entire stack is returned

Context Lines of context that are included in traceback entries

_options = {'chain': True, 'stream': None, 'print_width': None, 'globals_module_include': None, 'color': None, 'limit'

_flags = {'show_args': True, 'show_locals': True, 'show_globals': False, 'shorten_filenames': True, 'qualify_methods':

class xtraceback.**TracebackCompat**(***defaults*)

A context manager that patches the stdlib traceback module

Functions in the traceback module that exist as a method of this class are replaced with equivalents that use XTraceback.

Variables

- **NOPRINT** – Exception types that we don't print for (includes None)
- **defaults** – Default options to apply to XTracebacks created by this instance

NOPRINT = (None, <type 'exceptions.KeyboardInterrupt'>)

_factory(*etype*, *value*, *tb*, *limit=None*, ***options*)

_print_factory(*etype*, *value*, *tb*, *limit=None*, *file=None*, ***options*)

format_tb(*tb*, *limit=None*, ***options*)

A shorthand for 'format_list(extract_stack(f, limit)).

format_exception_only(*etype*, *value*, ***options*)

Format the exception part of a traceback.

The arguments are the exception type and value such as given by sys.last_type and sys.last_value. The return value is a list of strings, each ending in a newline.

Normally, the list contains a single string; however, for SyntaxError exceptions, it contains several lines that (when printed) display detailed information about where the syntax error occurred.

The message indicating which exception occurred is always the last string in the list.

format_exception(*etype*, *value*, *tb*, *limit=None*, ***options*)
Format a stack trace and the exception information.

The arguments have the same meaning as the corresponding arguments to print_exception(). The return value is a list of strings, each ending in a newline and some containing internal newlines. When these lines are concatenated and printed, exactly the same text is printed as does print_exception().

format_exc(*limit=None*, ***options*)
Like print_exc() but return a string.

print_tb(*tb*, *limit=None*, *file=None*, ***options*)
Print up to 'limit' stack trace entries from the traceback 'tb'.

If 'limit' is omitted or None, all entries are printed. If 'file' is omitted or None, the output goes to sys.stderr; otherwise 'file' should be an open file or file-like object with a write() method.

print_exception(*etype*, *value*, *tb*, *limit=None*, *file=None*, ***options*)
Print exception up to 'limit' stack trace entries from 'tb' to 'file'.

This differs from print_tb() in the following ways: (1) if traceback is not None, it prints a header "Traceback (most recent call last):"; (2) it prints the exception type and value after the stack trace; (3) if type is SyntaxError and value has the appropriate format, it prints the line where the syntax error occurred with a caret on the next line indicating the approximate position of the error.

print_exc(*limit=None*, *file=None*, ***options*)
Shorthand for 'print_exception(sys.exc_type, sys.exc_value, sys.exc_traceback, limit, file)'. (In fact, it uses sys.exc_info() to retrieve the same information in a thread-safe way.)

class `xtraceback.`**LoggingCompat**(*handler*, *tbcompat*, ***options*)

formatException(*ei*)

2.5 Nose plugin

The plugin is enabled with the *–with-xtraceback* flag. See *nosetests –help* for other options.

The plugin will not work in conjunction with other plugins that patch nose or stdlib hence a second plugin named yanc which colorizes nose output without resorting to monkey patching.

CHAPTER 3

Indices and tables

- *genindex*
- *modindex*
- *search*