pydocstyle Documentation

Release 1.0.0

Amir Rachum

April 01, 2016

Contents

(formerly pep257)

pydocstyle is a static analysis tool for checking compliance with Python docstring conventions.

pydocstyle supports most of PEP 257 out of the box, but it should not be considered a reference implementation.

CHAPTER 1

Quick Start

1. Install

```
pip install pydocstyle
```

2. Run

```
$ pydocstyle test.py
test.py:18 in private nested class `meta`:
        D101: Docstring missing
test.py:22 in public method `method`:
        D102: Docstring missing
...
```

3. Fix your code :)

Contents:

1.1 Usage

1.1.1 Installation

Use pip or easy_install:

```
pip install pydocstyle
```

Alternatively, you can use `pydocstyle.py` source file directly - it is self-contained.

1.1.2 Command Line Interface

Usage

```
Usage: pydocstyle [options] [<file|dir>...]

Options:
  --version             show program's version number and exit
  -h, --help            show this help message and exit
  -e, --explain         show explanation of each error
  -s, --source          show source for each error
  -d, --debug           print debug information
```

```
  -v, --verbose          print status information
  --count                print total number of errors to stdout
  --select=<codes>       choose the basic list of checked errors by specifying
                         which errors to check for (with a list of comma-
                         separated error codes). for example:
                         --select=D101,D202
  --ignore=<codes>       choose the basic list of checked errors by specifying
                         which errors to ignore (with a list of comma-separated
                         error codes). for example: --ignore=D101,D202
  --convention=<name>    choose the basic list of checked errors by specifying
                         an existing convention. Possible conventions: pep257
  --add-select=<codes>   amend the list of errors to check for by specifying
                         more error codes to check.
  --add-ignore=<codes>   amend the list of errors to check for by specifying
                         more error codes to ignore.
  --match=<pattern>      check only files that exactly match <pattern> regular
                         expression; default is --match='(?!test_).*\.py' which
                         matches files that don't start with 'test_' but end
                         with '.py'
  --match-dir=<pattern>
                         search only dirs that exactly match <pattern> regular
                         expression; default is --match-dir='[^\.].*', which
                         matches all dirs that don't start with a dot
```

Return Code

0	Success - no violations
1	Some code violations were found
2	Illegal usage - see error message

Configuration Files

`pydocstyle` supports *ini*-like configuration files. In order for `pydocstyle` to use it, it must be named one of the following options, and have a `[pydocstyle]` section.

- `setup.cfg`
- `tox.ini`
- `.pydocstyle`
- `.pydocstylerc`

When searching for a configuration file, `pydocstyle` looks for one of the file specified above *in that exact order*. If a configuration file was not found, it keeps looking for one up the directory tree until one is found or uses the default configuration.

Note: For backwards compatibility purposes, **pydocstyle** supports configuration files named `.pep257`, as well as section header `[pep257]`. However, these are considered deprecated and support will be removed in the next major version.

Available Options

Not all configuration options are available in the configuration files. Available options are:

- convention
- select
- ignore
- add_select
- add_ignore
- match
- match_dir

See the *Usage* section for more information.

Inheritance

By default, when finding a configuration file, pydocstyle tries to inherit the parent directory's configuration and merge them to the local ones.

The merge process is as follows:

- If one of select, ignore or convention was specified in the child configuration - Ignores the parent configuration and set the new error codes to check. Otherwise, simply copies the parent checked error codes.
- If add-ignore or add-select were specified, adds or removes the specified error codes from the checked error codes list.
- If match or match-dir were specified - use them. Otherwise, use the parent's.

In order to disable this (useful for configuration files located in your repo's root), simply add inherit=false to your configuration file.

Note: If any of select, ignore or convention were specified in the CLI, the configuration files will take no part in choosing which error codes will be checked. match and match-dir will still take effect.

Example

```
[pydocstyle]
inherit = false
ignore = D100,D203,D405
match = *.py
```

1.2 Error Codes

1.2.1 Grouping

Missing Docstrings	
D100	Missing docstring in public module
D101	Missing docstring in public class
D102	Missing docstring in public method
D103	Missing docstring in public function
D104	Missing docstring in public package
D105	Missing docstring in magic method
Whitespace Issues	
D200	One-line docstring should fit on one line with quotes
D201	No blank lines allowed before function docstring
D202	No blank lines allowed after function docstring
D203	1 blank line required before class docstring
D204	1 blank line required after class docstring
D205	1 blank line required between summary line and description
D206	Docstring should be indented with spaces, not tabs
D207	Docstring is under-indented
D208	Docstring is over-indented
D209	Multi-line docstring closing quotes should be on a separate line
D210	No whitespaces allowed surrounding docstring text
D211	No blank lines allowed before class docstring
Quotes Issues	
D300	Use “”“triple double quotes”“”
D301	Use r”“” if any backslashes in a docstring
D302	Use u”“” for Unicode docstrings
Docstring Content Issues	
D400	First line should end with a period
D401	First line should be in imperative mood
D402	First line should not be the function’s “signature”
D403	First word of the first line should be properly capitalized

1.2.2 Default Checks

Not all error codes are checked for by default. The default behavior is to check only error codes that are part of the PEP257 official convention.

All of the above error codes are checked for by default except for D203.

1.3 Release Notes

pydocstyle version numbers follow the Semantic Versioning specification.

1.3.1 1.0.0 - January 30th, 2016

Major Updates

- The project was renamed to **pydocstyle** and the new release will be 1.0.0!

New Features

- Added support for Python 3.5 (#145).
- Classes nested inside classes are no longer considered private. Nested classes are considered public if their names are not prepended with an underscore and if their parent class is public, recursively (#13, #146).
- Added the D403 error code - "First word of the first line should be properly capitalized". This new error is turned on by default (#164, #165, #170).
- Added support for `.pydocstylerc` and as configuration file name (#140, #173).

Bug Fixes

- Fixed an issue where a `NameError` was raised when parsing complex definitions of `__all__` (#142, #143).
- Fixed a bug where D202 was falsely reported when a function with just a docstring and no content was followed by a comment (#165).
- Fixed wrong `__all__` definition in main module (#150, #156).
- Fixed a bug where an `AssertionError` could occur when parsing `__future__` imports (#154).

1.4 Older Versions

Note: Versions documented below are before renaming the project from **pep257** to **pydocstyle**.

1.4.1 0.7.0 - October 9th, 2015

New Features

- Added the D104 error code - "Missing docstring in public package". This new error is turned on by default. Missing docstring in `__init__.py` files which previously resulted in D100 errors ("Missing docstring in public module") will now result in D104 (#105, #127).
- Added the D105 error code - "Missing docstring in magic method'. This new error is turned on by default. Missing docstrings in magic method which previously resulted in D102 error ("Missing docstring in public method") will now result in D105. Note that exceptions to this rule are variadic magic methods - specifically `__init__`, `__call__` and `__new__`, which will be considered non-magic and missing docstrings in them will result in D102 (#60, #139).
- Support the option to exclude all error codes. Running pep257 with `--select=` (or `select=` in the configuration file) will exclude all errors which could then be added one by one using `add-select`. Useful for projects new to pep257 (#132, #135).
- Added check D211: No blank lines allowed before class docstring. This change is a result of a change to the official PEP257 convention. Therefore, D211 will now be checked by default instead of D203, which required a single blank line before a class docstring (#137).
- Configuration files are now handled correctly. The closer a configuration file is to a checked file the more it matters. Configuration files no longer support `explain`, `source`, `debug`, `verbose` or `count` (#133).

Bug Fixes

- On Python 2.x, D302 ("Use u"""" for Unicode docstrings") is not reported if *unicode_literals* is imported from *__future__* (#113, #134).

- Fixed a bug where there was no executable for *pep257* on Windows (#73, #136).

1.4.2 0.6.0 - July 20th, 2015

New Features

- Added support for more flexible error selections using `--ignore`, `--select`, `--convention`, `--add-ignore` and `--add-select` (#96, #123).

Bug Fixes

- Property setter and deleter methods are now treated as private and do not require docstrings separate from the main property method (#69, #107).
- Fixed an issue where pep257 did not accept docstrings that are both unicode and raw in Python 2.x (#116, #119).
- Fixed an issue where Python 3.x files with Unicode encodings were not read correctly (#118).

1.4.3 0.5.0 - March 14th, 2015

New Features

- Added check D210: No whitespaces allowed surrounding docstring text (#95).
- Added real documentation rendering using Sphinx (#100, #101).

Bug Fixes

- Removed log level configuration from module level (#98).
- D205 used to check that there was *a* blank line between the one line summary and the description. It now checks that there is *exactly* one blank line between them (#79).
- Fixed a bug where `--match-dir` was not properly respected (#108, #109).

1.4.4 0.4.1 - January 10th, 2015

Bug Fixes

- Getting `ImportError` when trying to run pep257 as the installed script (#92, #93).

1.4.5 0.4.0 - January 4th, 2015

> **Warning:** A fatal bug was discovered in this version (#92). Please use a newer version.

New Features

- Added configuration file support (#58, #87).
- Added a `--count` flag that prints the number of violations found (#86, #89).
- Added support for Python 3.4, PyPy and PyPy3 (#81).

Bug Fixes

- Fixed broken tests (#74).
- Fixed parsing various colon and parenthesis combinations in definitions (#82).

- Allow for greater flexibility in parsing `__all__` (#67).
- Fixed handling of one-liner definitions (#77).

1.4.6 0.3.2 - March 11th, 2014

First documented release!

1.5 License

CHAPTER 2

Credits

pydocstyle is a rename and continuation of pep257, a project created by Vladimir Keleshev.

Maintained by Amir Rachum.