

pipver

Generated by Doxygen 1.8.13

Contents

1	Namespace Index	1
1.1	Packages	1
2	Hierarchical Index	3
2.1	Class Hierarchy	3
3	Class Index	5
3.1	Class List	5
4	File Index	7
4.1	File List	7
5	Namespace Documentation	9
5.1	hw Namespace Reference	9
5.1.1	Detailed Description	9
5.1.2	Variable Documentation	9
5.1.2.1	hw	9
5.2	pipver Namespace Reference	9
5.3	pipver.helloworld Namespace Reference	10
5.3.1	Detailed Description	10
5.4	setup Namespace Reference	10
5.4.1	Variable Documentation	10
5.4.1.1	author	10
5.4.1.2	author_email	10
5.4.1.3	classifiers	11

5.4.1.4	cmdclass	11
5.4.1.5	description	11
5.4.1.6	install_requires	11
5.4.1.7	license	11
5.4.1.8	long_description	11
5.4.1.9	name	11
5.4.1.10	packages	11
5.4.1.11	scripts	12
5.4.1.12	url	12
5.4.1.13	version	12
5.5	versioneer Namespace Reference	12
5.5.1	Function Documentation	13
5.5.1.1	do_setup()	13
5.5.1.2	do_vcs_install()	13
5.5.1.3	get_cmdclass()	13
5.5.1.4	get_config_from_root()	14
5.5.1.5	get_root()	14
5.5.1.6	get_version()	14
5.5.1.7	get_versions()	14
5.5.1.8	git_get_keywords()	14
5.5.1.9	git_pieces_from_vcs()	15
5.5.1.10	git_versions_from_keywords()	15
5.5.1.11	pep440_split_post()	15
5.5.1.12	plus_or_dot()	15
5.5.1.13	register_vcs_handler()	16
5.5.1.14	render()	16
5.5.1.15	render_git_describe()	16
5.5.1.16	render_git_describe_long()	16
5.5.1.17	render_pep440()	17
5.5.1.18	render_pep440_branch()	17

5.5.1.19	render_pep440_old()	17
5.5.1.20	render_pep440_post()	17
5.5.1.21	render_pep440_post_branch()	18
5.5.1.22	render_pep440_pre()	18
5.5.1.23	run_command()	18
5.5.1.24	scan_setup_py()	18
5.5.1.25	versions_from_file()	19
5.5.1.26	versions_from_parentdir()	19
5.5.1.27	write_to_version_file()	19
5.5.2	Variable Documentation	19
5.5.2.1	cmd	19
5.5.2.2	CONFIG_ERROR	20
5.5.2.3	errors	20
5.5.2.4	INIT_PY_SNIPPET	20
5.5.2.5	OLD_SNIPPET	20
5.5.2.6	SAMPLE_CONFIG	21
5.5.2.7	SHORT_VERSION_PY	21
6	Class Documentation	23
6.1	HelloWorld Class Reference	23
6.1.1	Detailed Description	23
6.1.2	Constructor & Destructor Documentation	23
6.1.2.1	__init__() [1/2]	24
6.1.2.2	__init__() [2/2]	24
6.1.3	Member Function Documentation	24
6.1.3.1	main() [1/2]	24
6.1.3.2	main() [2/2]	24
6.1.4	Member Data Documentation	24
6.1.4.1	text	24
6.2	NotThisMethod Class Reference	25
6.2.1	Detailed Description	25
6.3	VersioneerBadRootError Class Reference	26
6.3.1	Detailed Description	26
6.4	VersioneerConfig Class Reference	26
6.4.1	Detailed Description	26

7 File Documentation	27
7.1 build/lib/pipver/__init__.py File Reference	27
7.2 pipver/__init__.py File Reference	27
7.3 build/lib/pipver/helloworld.py File Reference	27
7.4 pipver/helloworld.py File Reference	27
7.5 build/scripts-3.6/hw.py File Reference	28
7.6 scripts/hw.py File Reference	28
7.7 setup.py File Reference	28
7.8 versioneer.py File Reference	29
 Index	 31

Chapter 1

Namespace Index

1.1 Packages

Here are the packages with brief descriptions (if available):

hw	Script that implements the Hello World Class	9
pipver	9
pipver.helloworld	Defines the the Hello World Class	10
setup	10
versioneer	12

Chapter 2

Hierarchical Index

2.1 Class Hierarchy

This inheritance list is sorted roughly, but not completely, alphabetically:

Exception	
NotThisMethod	25
VersioneerBadRootError	26
HelloWorld	23
VersioneerConfig	26

Chapter 3

Class Index

3.1 Class List

Here are the classes, structs, unions and interfaces with brief descriptions:

HelloWorld	
The Hello World base class	23
NotThisMethod	25
VersioneerBadRootError	26
VersioneerConfig	26

Chapter 4

File Index

4.1 File List

Here is a list of all files with brief descriptions:

setup.py	28
versioneer.py	29
build/lib/pipver/___init___py	27
build/lib/pipver/helloworld.py	27
build/scripts-3.6/hw.py	28
pipver/___init___py	27
pipver/helloworld.py	27
scripts/hw.py	28

Chapter 5

Namespace Documentation

5.1 hw Namespace Reference

Script that implements the Hello World Class.

Variables

- `hw` = `helloworld.HelloWorld()`

5.1.1 Detailed Description

Script that implements the Hello World Class.

5.1.2 Variable Documentation

5.1.2.1 hw

```
hw = helloworld.HelloWorld()
```

5.2 pipver Namespace Reference

Namespaces

- `helloworld`

Defines the the Hello World Class.

5.3 pipver.helloworld Namespace Reference

Defines the the Hello World Class.

Classes

- class [HelloWorld](#)
The Hello World base class.

5.3.1 Detailed Description

Defines the the Hello World Class.

5.4 setup Namespace Reference

Variables

- [name](#)
- [version](#)
- [cmdclass](#)
- [description](#)
- [long_description](#)
- [url](#)
- [author](#)
- [author_email](#)
- [license](#)
- [packages](#)
- [scripts](#)
- [install_requires](#)
- [classifiers](#)

5.4.1 Variable Documentation

5.4.1.1 author

`author`

5.4.1.2 author_email

`author_email`

5.4.1.3 classifiers

classifiers

5.4.1.4 cmdclass

cmdclass

5.4.1.5 description

description

5.4.1.6 install_requires

install_requires

5.4.1.7 license

license

5.4.1.8 long_description

long_description

5.4.1.9 name

name

5.4.1.10 packages

packages

5.4.1.11 scripts

scripts

5.4.1.12 url

url

5.4.1.13 version

version

5.5 versioneer Namespace Reference

Classes

- class [NotThisMethod](#)
- class [VersioneerBadRootError](#)
- class [VersioneerConfig](#)

Functions

- def [get_root](#) ()
- def [get_config_from_root](#) (root)
- def [register_vcs_handler](#) (vcs, method)
- def [run_command](#) (commands, args, cwd=None, verbose=False, hide_stderr=False, env=None)
- def [git_get_keywords](#) (versionfile_abs)
- def [git_versions_from_keywords](#) (keywords, tag_prefix, verbose)
- def [git_pieces_from_vcs](#) (tag_prefix, root, verbose, runner=[run_command](#))
- def [do_vcs_install](#) (manifest_in, versionfile_source, ipy)
- def [versions_from_parentdir](#) (parentdir_prefix, root, verbose)
- def [versions_from_file](#) (filename)
- def [write_to_version_file](#) (filename, versions)
- def [plus_or_dot](#) (pieces)
- def [render_pep440](#) (pieces)
- def [render_pep440_branch](#) (pieces)
- def [pep440_split_post](#) (ver)
- def [render_pep440_pre](#) (pieces)
- def [render_pep440_post](#) (pieces)
- def [render_pep440_post_branch](#) (pieces)
- def [render_pep440_old](#) (pieces)
- def [render_git_describe](#) (pieces)
- def [render_git_describe_long](#) (pieces)
- def [render](#) (pieces, style)
- def [get_versions](#) (verbose=False)
- def [get_version](#) ()
- def [get_cmdclass](#) (cmdclass=None)
- def [do_setup](#) ()
- def [scan_setup_py](#) ()

Variables

- string `SHORT_VERSION_PY`
- string `CONFIG_ERROR`
- string `SAMPLE_CONFIG`
- string `OLD_SNIPPET`
- string `INIT_PY_SNIPPET`
- `cmd = sys.argv[1]`
- `def errors = do_setup()`

5.5.1 Function Documentation

5.5.1.1 `do_setup()`

```
def versioneer.do_setup ( )
```

Do main VCS-independent setup function for installing Versioneer.

5.5.1.2 `do_vcs_install()`

```
def versioneer.do_vcs_install (
    manifest_in,
    versionfile_source,
    ipy )
```

Git-specific installation logic for Versioneer.

For Git, this means creating/changing `.gitattributes` to mark `_version.py` for export-subst keyword substitution.

5.5.1.3 `get_cmdclass()`

```
def versioneer.get_cmdclass (
    cmdclass = None )
```

Get the custom `setuptools/distutils` subclasses used by Versioneer.

If the package uses a different `cmdclass` (e.g. one from `numpy`), it should be provide as an argument.

5.5.1.4 `get_config_from_root()`

```
def versioneer.get_config_from_root (
    root )
```

Read the project `setup.cfg` file to determine Versioneer config.

5.5.1.5 `get_root()`

```
def versioneer.get_root ( )
```

Get the project root directory.

We require that all commands are run from the project root, i.e. the directory that contains `setup.py`, `setup.cfg`, and `versioneer.py`.

5.5.1.6 `get_version()`

```
def versioneer.get_version ( )
```

Get the short version string for this project.

5.5.1.7 `get_versions()`

```
def versioneer.get_versions (
    verbose = False )
```

Get the project version from whatever source is available.

Returns dict with two keys: `'version'` and `'full'`.

5.5.1.8 `git_get_keywords()`

```
def versioneer.git_get_keywords (
    versionfile_abs )
```

Extract version information from the given file.

5.5.1.9 `git_pieces_from_vcs()`

```
def versioneer.git_pieces_from_vcs (
    tag_prefix,
    root,
    verbose,
    runner = run_command )
```

Get version from 'git describe' in the root of the source tree.

This only gets called if the git-archive 'subst' keywords were **not** expanded, and `_version.py` hasn't already been rewritten with a short version string, meaning we're inside a checked out source tree.

5.5.1.10 `git_versions_from_keywords()`

```
def versioneer.git_versions_from_keywords (
    keywords,
    tag_prefix,
    verbose )
```

Get version information from git keywords.

5.5.1.11 `pep440_split_post()`

```
def versioneer.pep440_split_post (
    ver )
```

Split pep440 version string at the post-release segment.

Returns the release segments before the post-release and the post-release version number (or -1 if no post-release segment is present).

5.5.1.12 `plus_or_dot()`

```
def versioneer.plus_or_dot (
    pieces )
```

Return a + if we don't already have one, else return a .

5.5.1.13 register_vcs_handler()

```
def versioneer.register_vcs_handler (
    vcs,
    method )
```

Create decorator to mark a method as the handler of a VCS.

5.5.1.14 render()

```
def versioneer.render (
    pieces,
    style )
```

Render the given version pieces into the requested style.

5.5.1.15 render_git_describe()

```
def versioneer.render_git_describe (
    pieces )
```

TAG[-DISTANCE-gHEX][-dirty].

Like 'git describe --tags --dirty --always'.

Exceptions:

1: no tags. HEX[-dirty] (note: no 'g' prefix)

5.5.1.16 render_git_describe_long()

```
def versioneer.render_git_describe_long (
    pieces )
```

TAG-DISTANCE-gHEX[-dirty].

Like 'git describe --tags --dirty --always -long'.
The distance/hash is unconditional.

Exceptions:

1: no tags. HEX[-dirty] (note: no 'g' prefix)

5.5.1.17 `render_pep440()`

```
def versioneer.render_pep440 (
    pieces )
```

Build up version string, with post-release "local version identifier".

Our goal: TAG[+DISTANCE.gHEX[.dirty]] . Note that if you get a tagged build and then dirty it, you'll get TAG+0.gHEX.dirty

Exceptions:

1: no tags. git_describe was just HEX. 0+untagged.DISTANCE.gHEX[.dirty]

5.5.1.18 `render_pep440_branch()`

```
def versioneer.render_pep440_branch (
    pieces )
```

TAG[.dev0]+DISTANCE.gHEX[.dirty]] .

The ".dev0" means not master branch. Note that .dev0 sorts backwards (a feature branch will appear "older" than the master branch).

Exceptions:

1: no tags. 0[.dev0]+untagged.DISTANCE.gHEX[.dirty]

5.5.1.19 `render_pep440_old()`

```
def versioneer.render_pep440_old (
    pieces )
```

TAG[.postDISTANCE[.dev0]] .

The ".dev0" means dirty.

Exceptions:

1: no tags. 0.postDISTANCE[.dev0]

5.5.1.20 `render_pep440_post()`

```
def versioneer.render_pep440_post (
    pieces )
```

TAG[.postDISTANCE[.dev0]+gHEX] .

The ".dev0" means dirty. Note that .dev0 sorts backwards (a dirty tree will appear "older" than the corresponding clean one), but you shouldn't be releasing software with -dirty anyways.

Exceptions:

1: no tags. 0.postDISTANCE[.dev0]

5.5.1.21 render_pep440_post_branch()

```
def versioneer.render_pep440_post_branch (
    pieces )
```

```
TAG[.postDISTANCE[.dev0]+gHEX[.dirty]] .
```

The ".dev0" means not master branch.

Exceptions:

```
1: no tags. 0.postDISTANCE[.dev0]+gHEX[.dirty]
```

5.5.1.22 render_pep440_pre()

```
def versioneer.render_pep440_pre (
    pieces )
```

```
TAG[.postN.devDISTANCE] -- No -dirty.
```

Exceptions:

```
1: no tags. 0.post0.devDISTANCE
```

5.5.1.23 run_command()

```
def versioneer.run_command (
    commands,
    args,
    cwd = None,
    verbose = False,
    hide_stderr = False,
    env = None )
```

Call the given command(s).

5.5.1.24 scan_setup_py()

```
def versioneer.scan_setup_py ( )
```

Validate the contents of setup.py against Versioneer's expectations.

5.5.1.25 versions_from_file()

```
def versioneer.versions_from_file (
    filename )
```

Try to determine the version from `_version.py` if present.

5.5.1.26 versions_from_parentdir()

```
def versioneer.versions_from_parentdir (
    parentdir_prefix,
    root,
    verbose )
```

Try to determine the version from the parent directory name.

Source tarballs conventionally unpack into a directory that includes both the project name and a version string. We will also support searching up two directory levels for an appropriately named parent directory

5.5.1.27 write_to_version_file()

```
def versioneer.write_to_version_file (
    filename,
    versions )
```

Write the given version number to the given `_version.py` file.

5.5.2 Variable Documentation

5.5.2.1 cmd

```
cmd = sys.argv[1]
```

5.5.2.2 CONFIG_ERROR

string CONFIG_ERROR

Initial value:

```

1 = """
2 setup.cfg is missing the necessary Versioneer configuration. You need
3 a section like:
4
5 [versioneer]
6 VCS = git
7 style = pep440
8 versionfile_source = src/myproject/_version.py
9 versionfile_build = myproject/_version.py
10 tag_prefix =
11 parentdir_prefix = myproject-
12
13 You will also need to edit your setup.py to use the results:
14
15 import versioneer
16 setup(version=versioneer.get_version(),
17       cmdclass=versioneer.get_cmdclass(), ...)
18
19 Please read the docstring in ./versioneer.py for configuration instructions,
20 edit setup.cfg, and re-run the installer or 'python versioneer.py setup'.
21 """

```

5.5.2.3 errors

def errors = [do_setup\(\)](#)

5.5.2.4 INIT_PY_SNIPPET

string INIT_PY_SNIPPET

Initial value:

```

1 = """
2 from . import {0}
3 __version__ = {0}.get_versions()['version']
4 """

```

5.5.2.5 OLD_SNIPPET

string OLD_SNIPPET

Initial value:

```

1 = """
2 from ._version import get_versions
3 __version__ = get_versions()['version']
4 del get_versions
5 """

```

5.5.2.6 SAMPLE_CONFIG

string SAMPLE_CONFIG

Initial value:

```
1 = """
2 # See the docstring in versioneer.py for instructions. Note that you must
3 # re-run 'versioneer.py setup' after changing this section, and commit the
4 # resulting files.
5
6 [versioneer]
7 #VCS = git
8 #style = pep440
9 #versionfile_source =
10 #versionfile_build =
11 #tag_prefix =
12 #parentdir_prefix =
13
14 """
```

5.5.2.7 SHORT_VERSION_PY

string SHORT_VERSION_PY

Initial value:

```
1 = """
2 # This file was generated by 'versioneer.py' (0.22) from
3 # revision-control system data, or from the parent directory name of an
4 # unpacked source archive. Distribution tarballs contain a pre-generated copy
5 # of this file.
6
7 import json
8
9 version_json = '''
10 %s
11 ''' # END VERSION_JSON
12
13
14 def get_versions():
15     return json.loads(version_json)
16 """
```


Chapter 6

Class Documentation

6.1 HelloWorld Class Reference

The Hello World base class.

Public Member Functions

- `def __init__ (self)`
Initialize the text and call the main function.
- `def main (self)`
Print Hello World.
- `def __init__ (self)`
Initialize the text and call the main function.
- `def main (self)`
Print Hello World.

Public Attributes

- `text`

6.1.1 Detailed Description

The Hello World base class.

6.1.2 Constructor & Destructor Documentation

6.1.2.1 `__init__()` [1/2]

```
def __init__ (
    self )
```

Initialize the text and call the main function.

6.1.2.2 `__init__()` [2/2]

```
def __init__ (
    self )
```

Initialize the text and call the main function.

6.1.3 Member Function Documentation**6.1.3.1** `main()` [1/2]

```
def main (
    self )
```

Print Hello World.

6.1.3.2 `main()` [2/2]

```
def main (
    self )
```

Print Hello World.

6.1.4 Member Data Documentation**6.1.4.1** `text`

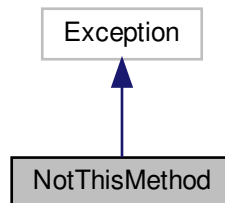
```
text
```

The documentation for this class was generated from the following file:

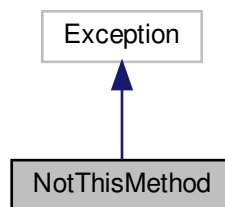
- [build/lib/pipver/helloworld.py](#)

6.2 NotThisMethod Class Reference

Inheritance diagram for NotThisMethod:



Collaboration diagram for NotThisMethod:



6.2.1 Detailed Description

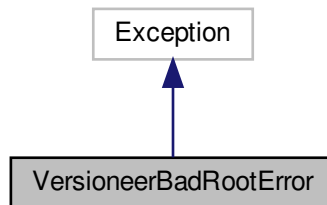
Exception raised if a method is not valid for the current scenario.

The documentation for this class was generated from the following file:

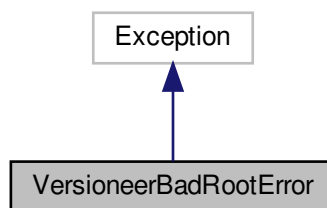
- [versioneer.py](#)

6.3 VersioneerBadRootError Class Reference

Inheritance diagram for VersioneerBadRootError:



Collaboration diagram for VersioneerBadRootError:



6.3.1 Detailed Description

The project root directory is unknown or missing key files.

The documentation for this class was generated from the following file:

- [versioneer.py](#)

6.4 VersioneerConfig Class Reference

6.4.1 Detailed Description

Container for Versioneer configuration parameters.

The documentation for this class was generated from the following file:

- [versioneer.py](#)

Chapter 7

File Documentation

7.1 build/lib/pipver/__init__.py File Reference

Namespaces

- [pipver](#)

7.2 pipver/__init__.py File Reference

Namespaces

- [pipver](#)

7.3 build/lib/pipver/helloworld.py File Reference

Classes

- class [HelloWorld](#)
The Hello World base class.

Namespaces

- [pipver.helloworld](#)
Defines the the Hello World Class.

7.4 pipver/helloworld.py File Reference

Classes

- class [HelloWorld](#)
The Hello World base class.

Namespaces

- [pipver.helloworld](#)

Defines the the Hello World Class.

7.5 build/scripts-3.6/hw.py File Reference

Namespaces

- [hw](#)

Script that implements the Hello World Class.

Variables

- [hw](#) = helloworld.HelloWorld()

7.6 scripts/hw.py File Reference

Namespaces

- [hw](#)

Script that implements the Hello World Class.

7.7 setup.py File Reference

Namespaces

- [setup](#)

Variables

- [name](#)
- [version](#)
- [cmdclass](#)
- [description](#)
- [long_description](#)
- [url](#)
- [author](#)
- [author_email](#)
- [license](#)
- [packages](#)
- [scripts](#)
- [install_requires](#)
- [classifiers](#)

7.8 versioneer.py File Reference

Classes

- class [VersioneerConfig](#)
- class [NotThisMethod](#)
- class [VersioneerBadRootError](#)

Namespaces

- [versioneer](#)

Functions

- def [get_root](#) ()
- def [get_config_from_root](#) (root)
- def [register_vcs_handler](#) (vcs, method)
- def [run_command](#) (commands, args, cwd=None, verbose=False, hide_stderr=False, env=None)
- def [git_get_keywords](#) (versionfile_abs)
- def [git_versions_from_keywords](#) (keywords, tag_prefix, verbose)
- def [git_pieces_from_vcs](#) (tag_prefix, root, verbose, runner=run_command)
- def [do_vcs_install](#) (manifest_in, versionfile_source, ipy)
- def [versions_from_parentdir](#) (parentdir_prefix, root, verbose)
- def [versions_from_file](#) (filename)
- def [write_to_version_file](#) (filename, versions)
- def [plus_or_dot](#) (pieces)
- def [render_pep440](#) (pieces)
- def [render_pep440_branch](#) (pieces)
- def [pep440_split_post](#) (ver)
- def [render_pep440_pre](#) (pieces)
- def [render_pep440_post](#) (pieces)
- def [render_pep440_post_branch](#) (pieces)
- def [render_pep440_old](#) (pieces)
- def [render_git_describe](#) (pieces)
- def [render_git_describe_long](#) (pieces)
- def [render](#) (pieces, style)
- def [get_versions](#) (verbose=False)
- def [get_version](#) ()
- def [get_cmdclass](#) (cmdclass=None)
- def [do_setup](#) ()
- def [scan_setup_py](#) ()

Variables

- string [SHORT_VERSION_PY](#)
- string [CONFIG_ERROR](#)
- string [SAMPLE_CONFIG](#)
- string [OLD_SNIPPET](#)
- string [INIT_PY_SNIPPET](#)
- [cmd](#) = sys.argv[1]
- def [errors](#) = do_setup()

Index

`__init__`
 [pipver::helloworld::HelloWorld](#), [23](#), [24](#)

author
 [setup](#), [10](#)
author_email
 [setup](#), [10](#)

[build/lib/pipver/__init__.py](#), [27](#)
[build/lib/pipver/helloworld.py](#), [27](#)
[build/scripts-3.6/hw.py](#), [28](#)

CONFIG_ERROR
 [versioneer](#), [19](#)

classifiers
 [setup](#), [10](#)

cmd
 [versioneer](#), [19](#)
cmdclass
 [setup](#), [11](#)

description
 [setup](#), [11](#)
do_setup
 [versioneer](#), [13](#)
do_vcs_install
 [versioneer](#), [13](#)

errors
 [versioneer](#), [20](#)

get_cmdclass
 [versioneer](#), [13](#)
get_config_from_root
 [versioneer](#), [13](#)
get_root
 [versioneer](#), [14](#)
get_version
 [versioneer](#), [14](#)
get_versions
 [versioneer](#), [14](#)
git_get_keywords
 [versioneer](#), [14](#)
git_pieces_from_vcs
 [versioneer](#), [14](#)
git_versions_from_keywords
 [versioneer](#), [15](#)

[HelloWorld](#), [23](#)
[hw](#), [9](#)
 [hw](#), [9](#)

INIT_PY_SNIPPET
 [versioneer](#), [20](#)
install_requires
 [setup](#), [11](#)

license
 [setup](#), [11](#)
long_description
 [setup](#), [11](#)

main
 [pipver::helloworld::HelloWorld](#), [24](#)

name
 [setup](#), [11](#)
[NotThisMethod](#), [25](#)

OLD_SNIPPET
 [versioneer](#), [20](#)

packages
 [setup](#), [11](#)
pep440_split_post
 [versioneer](#), [15](#)
[pipver](#), [9](#)
[pipver.helloworld](#), [10](#)
[pipver/__init__.py](#), [27](#)
[pipver/helloworld.py](#), [27](#)
[pipver::helloworld::HelloWorld](#)
 [__init__](#), [23](#), [24](#)
 [main](#), [24](#)
 [text](#), [24](#)
plus_or_dot
 [versioneer](#), [15](#)

register_vcs_handler
 [versioneer](#), [15](#)
render
 [versioneer](#), [16](#)
render_git_describe
 [versioneer](#), [16](#)
render_git_describe_long
 [versioneer](#), [16](#)
render_pep440
 [versioneer](#), [16](#)
render_pep440_branch
 [versioneer](#), [17](#)
render_pep440_old
 [versioneer](#), [17](#)
render_pep440_post
 [versioneer](#), [17](#)

- render_pep440_post_branch
 - versioneer, [17](#)
- render_pep440_pre
 - versioneer, [18](#)
- run_command
 - versioneer, [18](#)
- SAMPLE_CONFIG
 - versioneer, [20](#)
- SHORT_VERSION_PY
 - versioneer, [21](#)
- scan_setup_py
 - versioneer, [18](#)
- scripts
 - setup, [11](#)
- scripts/hw.py, [28](#)
- setup, [10](#)
 - author, [10](#)
 - author_email, [10](#)
 - classifiers, [10](#)
 - cmdclass, [11](#)
 - description, [11](#)
 - install_requires, [11](#)
 - license, [11](#)
 - long_description, [11](#)
 - name, [11](#)
 - packages, [11](#)
 - scripts, [11](#)
 - url, [12](#)
 - version, [12](#)
- setup.py, [28](#)
- text
 - pipver::helloworld::HelloWorld, [24](#)
- url
 - setup, [12](#)
- version
 - setup, [12](#)
- versioneer, [12](#)
 - CONFIG_ERROR, [19](#)
 - cmd, [19](#)
 - do_setup, [13](#)
 - do_vcs_install, [13](#)
 - errors, [20](#)
 - get_cmdclass, [13](#)
 - get_config_from_root, [13](#)
 - get_root, [14](#)
 - get_version, [14](#)
 - get_versions, [14](#)
 - git_get_keywords, [14](#)
 - git_pieces_from_vcs, [14](#)
 - git_versions_from_keywords, [15](#)
 - INIT_PY_SNIPPET, [20](#)
 - OLD_SNIPPET, [20](#)
 - pep440_split_post, [15](#)
 - plus_or_dot, [15](#)
 - register_vcs_handler, [15](#)
 - render, [16](#)
 - render_git_describe, [16](#)
 - render_git_describe_long, [16](#)
 - render_pep440, [16](#)
 - render_pep440_branch, [17](#)
 - render_pep440_old, [17](#)
 - render_pep440_post, [17](#)
 - render_pep440_post_branch, [17](#)
 - render_pep440_pre, [18](#)
 - run_command, [18](#)
 - SAMPLE_CONFIG, [20](#)
 - SHORT_VERSION_PY, [21](#)
 - scan_setup_py, [18](#)
 - versions_from_file, [18](#)
 - versions_from_parentdir, [19](#)
 - write_to_version_file, [19](#)
- versioneer.py, [29](#)
- VersioneerBadRootError, [26](#)
- VersioneerConfig, [26](#)
- versions_from_file
 - versioneer, [18](#)
- versions_from_parentdir
 - versioneer, [19](#)
- write_to_version_file
 - versioneer, [19](#)