

---

**pep517**

**Nov 06, 2022**



---

## Contents:

---

<b>1</b>	<b>API reference</b>	<b>3</b>
1.1	Calling the build system . . . . .	3
1.2	Subprocess runners . . . . .	5
1.3	Exceptions . . . . .	5
<b>2</b>	<b>Changelog</b>	<b>7</b>
2.1	0.13 . . . . .	7
2.2	0.12 . . . . .	7
2.3	0.11.1 . . . . .	7
2.4	0.11 . . . . .	7
2.5	0.10 . . . . .	8
2.6	0.9.1 . . . . .	8
2.7	0.9 . . . . .	8
<b>3</b>	<b>Indices and tables</b>	<b>9</b>
	<b>Python Module Index</b>	<b>11</b>
	<b>Index</b>	<b>13</b>



**Warning:** The core of this package has been renamed to [pyproject-hooks](#). Please use that package (low level) or [build](#) (higher level) in place of pep517.

This package provides an API to call the hooks defined in [PEP 517](#).



**Warning:** The core of this package (the APIs described below) has been renamed to [pyproject-hooks](#). Please use that package (low level) or [build](#) (higher level) in place of pep517.

## 1.1 Calling the build system

**class** pep517.**Pep517HookCaller** (*source\_dir, build\_backend, backend\_path=None, runner=None, python\_executable=None*)

A wrapper around a source directory to be built with a PEP 517 backend.

### Parameters

- **source\_dir** – The path to the source directory, containing pyproject.toml.
- **build\_backend** – The build backend spec, as per PEP 517, from pyproject.toml.
- **backend\_path** – The backend path, as per PEP 517, from pyproject.toml.
- **runner** – A callable that invokes the wrapper subprocess.
- **python\_executable** – The Python executable used to invoke the backend

The ‘runner’, if provided, must expect the following:

- **cmd**: a list of strings representing the command and arguments to execute, as would be passed to e.g. ‘subprocess.check\_call’.
- **cwd**: a string representing the working directory that must be used for the subprocess. Corresponds to the provided **source\_dir**.
- **extra\_envron**: a dict mapping environment variable names to values which must be set for the subprocess execution.

**get\_requires\_for\_build\_sdist** (*config\_settings=None*)

Identify packages required for building a wheel

Returns a list of dependency specifications, e.g.:

```
["setuptools >= 26"]
```

This does not include requirements specified in pyproject.toml. It returns the result of calling the equivalently named hook in a subprocess.

**get\_requires\_for\_build\_wheel** (*config\_settings=None*)

Identify packages required for building a wheel

Returns a list of dependency specifications, e.g.:

```
["wheel >= 0.25", "setuptools"]
```

This does not include requirements specified in pyproject.toml. It returns the result of calling the equivalently named hook in a subprocess.

**get\_requires\_for\_build\_editable** (*config\_settings=None*)

Identify packages required for building an editable wheel

Returns a list of dependency specifications, e.g.:

```
["wheel >= 0.25", "setuptools"]
```

This does not include requirements specified in pyproject.toml. It returns the result of calling the equivalently named hook in a subprocess.

**prepare\_metadata\_for\_build\_wheel** (*metadata\_directory*, *config\_settings=None*, *\_allow\_fallback=True*)

Prepare a \*.dist-info folder with metadata for this project.

Returns the name of the newly created folder.

If the build backend defines a hook with this name, it will be called in a subprocess. If not, the backend will be asked to build a wheel, and the dist-info extracted from that (unless *\_allow\_fallback* is False).

**prepare\_metadata\_for\_build\_editable** (*metadata\_directory*, *config\_settings=None*, *\_allow\_fallback=True*)

Prepare a \*.dist-info folder with metadata for this project.

Returns the name of the newly created folder.

If the build backend defines a hook with this name, it will be called in a subprocess. If not, the backend will be asked to build an editable wheel, and the dist-info extracted from that (unless *\_allow\_fallback* is False).

**build\_sdist** (*sdist\_directory*, *config\_settings=None*)

Build an sdist from this project.

Returns the name of the newly created file.

This calls the 'build\_sdist' backend hook in a subprocess.

**build\_wheel** (*wheel\_directory*, *config\_settings=None*, *metadata\_directory=None*)

Build a wheel from this project.

Returns the name of the newly created file.

In general, this will call the 'build\_wheel' hook in the backend. However, if that was previously called by 'prepare\_metadata\_for\_build\_wheel', and the same *metadata\_directory* is used, the previously built wheel will be copied to *wheel\_directory*.

**build\_editable** (*wheel\_directory*, *config\_settings=None*, *metadata\_directory=None*)

Build an editable wheel from this project.



Returns the name of the newly created file.

In general, this will call the ‘build\_editable’ hook in the backend. However, if that was previously called by ‘prepare\_metadata\_for\_build\_editable’, and the same metadata\_directory is used, the previously built wheel will be copied to wheel\_directory.

**subprocess\_runner** (*runner*)

A context manager for temporarily overriding the default subprocess runner.

## 1.2 Subprocess runners

These functions may be provided when creating *Pep517HookCaller*, or to *Pep517HookCaller.subprocess\_runner()*.

**pep517.default\_subprocess\_runner** (*cmd*, *cwd=None*, *extra\_environ=None*)

The default method of calling the wrapper subprocess.

**pep517.quiet\_subprocess\_runner** (*cmd*, *cwd=None*, *extra\_environ=None*)

A method of calling the wrapper subprocess while suppressing output.

## 1.3 Exceptions

**exception** **pep517.BackendUnavailable** (*traceback*)

Will be raised if the backend cannot be imported in the hook process.

**exception** **pep517.BackendInvalid** (*backend\_name*, *backend\_path*, *message*)

Will be raised if the backend is invalid.

**exception** **pep517.HookMissing** (*hook\_name*)

Will be raised on missing hooks.

**exception** **pep517.UnsupportedOperation** (*traceback*)

May be raised by build\_sdist if the backend indicates that it can’t.



## CHAPTER 2

---

### Changelog

---

**Warning:** The core of this package has been renamed to `pyproject-hooks`. Please use that package (low level) or `build` (higher level) in place of `pep517`.

#### 2.1 0.13

- Remove support for end-of-life Pythons. Now requires Python3.6+.
- Remove support for `toml` package. Now requires `tomli`.
- Rely on preferred “files” API on Python 3.9 and later (#140).

#### 2.2 0.12

- Add method for pip to check if `build_editable` hook is supported. This is a private API for now.

#### 2.3 0.11.1

- Fix `DeprecationWarning` in `tomli`.

#### 2.4 0.11

- Support editable hooks ([PEP 660](#)).
- Use the TOML 1.0 compliant `tomli` parser module on Python 3.6 and above.
- Ensure TOML files are always read as UTF-8.

- Switch CI to Github actions.

## 2.5 0.10

- Avoid shadowing imports such as `colorlog` in the backend, by moving the `_in_process.py` script into a separate subpackage.
- Issue warnings when using the deprecated `pep517.build` and `pep517.check` modules at the command line. See the [PyPA build project](#) for a replacement.
- Allow building with `flit_core 3.x`.
- Prefer the standard library `unittest.mock` to `mock` for tests on Python 3.6 and above.

## 2.6 0.9.1

- Silence some static analysis warnings.

## 2.7 0.9

- Deprecated the higher level API which handles creating an environment and installing build dependencies. This was not very complete, and the [PyPA build project](#) is designed for this use case.
- New `python_executable` parameter for *Pep517HookCaller* to run hooks with a different Python interpreter.
- Fix for locating the script to run in the subprocess in some scenarios.
- Fix example in README to get `build-backend` correctly.
- Created [documentation on Read the Docs](#)
- Various minor improvements to testing.

## CHAPTER 3

---

### Indices and tables

---

- `genindex`
- `modindex`
- `search`



**p**

pep517, 3





## B

BackendInvalid, 5  
BackendUnavailable, 5  
build\_editable() (*pep517.Pep517HookCaller*  
    *method*), 4  
build\_sdist() (*pep517.Pep517HookCaller method*),  
    4  
build\_wheel() (*pep517.Pep517HookCaller method*),  
    4

## D

default\_subprocess\_runner() (*in module*  
    *pep517*), 5

## G

get\_requires\_for\_build\_editable()  
    (*pep517.Pep517HookCaller method*), 4  
get\_requires\_for\_build\_sdist()  
    (*pep517.Pep517HookCaller method*), 3  
get\_requires\_for\_build\_wheel()  
    (*pep517.Pep517HookCaller method*), 4

## H

HookMissing, 5

## P

pep517 (*module*), 3  
Pep517HookCaller (*class in pep517*), 3  
prepare\_metadata\_for\_build\_editable()  
    (*pep517.Pep517HookCaller method*), 4  
prepare\_metadata\_for\_build\_wheel()  
    (*pep517.Pep517HookCaller method*), 4  
Python Enhancement Proposals  
    PEP 517, 1

## Q

quiet\_subprocess\_runner() (*in module*  
    *pep517*), 5

## S

subprocess\_runner() (*pep517.Pep517HookCaller*  
    *method*), 5

## U

UnsupportedOperation, 5