

---

# **settei Documentation**

***Release 0.2.0***

**Spoqa, Inc**

**Jul 12, 2016**



<b>1</b>	<b>Loading a configuration is easy</b>	<b>3</b>
<b>2</b>	<b>References</b>	<b>5</b>
2.1	<code>settei</code> — App object holding configuration . . . . .	5
2.1.1	<code>settei.base</code> — Basic app object . . . . .	5
2.1.2	<code>settei.presets</code> — Richer presets for several frameworks . . . . .	6
2.1.3	<code>settei.version</code> — Version data . . . . .	7
<b>3</b>	<b>Indices and tables</b>	<b>9</b>
	<b>Python Module Index</b>	<b>11</b>



Configuration utility for common Python applications and services. FYI, “settei” () means settings in Japanese. :)



---

## Loading a configuration is easy

---

Suppose you use `Flask` with `Settei`.

```
from flask import Flask
from settei import Configuration, config_property

class WebConfiguration(Configuration):
    """Load Configuration::

        [web]
        debug = true

    """

    #: debug option
    debug = config_property('web.debug', bool, default=False)

conf = WebConfiguration.from_path(pathlib.Path('.') / 'dev.toml')
app = Flask(__name__)

if __name__ == '__main__':
    app.run(debug=conf.debug)
```





---

## References

---

### 2.1 settei — App object holding configuration

#### copyright

3. 2016 Spoqa, Inc.

**license** Apache License 2.0, see LICENSE for more details.

#### 2.1.1 settei.base — Basic app object

**class** `settei.base.Configuration` ( *config: typing.Mapping[str, object] = {}, \*\*kwargs* )

Application instance with its settings e.g. database. It implements read-only `Mapping` protocol as well, so you can treat it as a dictionary of string keys.

**classmethod** `from_file` ( *file* ) → `settei.base.Configuration`

Load settings from the given *file* and instantiate an `Configuration` instance from that.

**Parameters** *file* – the file object that contains TOML settings

**Returns** an instantiated configuration

**Return type** `Configuration`

**classmethod** `from_path` ( )

Load settings from the given *path* and instantiate an `Configuration` instance from that.

**Parameters** *path* (`pathlib.Path`) – the file path that contains TOML settings

**Returns** an instantiated configuration

**Return type** `Configuration`

**exception** `settei.base.ConfigWarning`

Warning category which raised when a default configuration is used instead due to missing required configuration.

**class** `settei.base.config_property` ( *key: str, cls: type, docstring: str = None, \*\*kwargs* ) → `None`

Declare configuration key with type hints, default value, and docstring.

#### Parameters

- **key** (`str`) – the dotted string of key path. for example `abc.def` looks up `config['abc']['def']`
- **cls** (`type`) – the allowed type of the configuration

- **docstring** (`str`) – optional documentation about the configuration. it will be set to `__doc__` attribute
- **default** – keyword only argument. optional default value used for missing case. cannot be used with `default_func` at a time
- **default\_func** (`collections.abc.Callable`) – keyword only argument. optional callable which returns a default value for missing case. it has to take an `App` mapping, and return a default value. cannot be used with `default` at a time
- **default\_warning** (`bool`) – keyword only argument. whether to warn when default value is used. does not warn by default. this option is only available when `default` value is provided

**docstring**

(`str`) The properly indented `__doc__` string.

## 2.1.2 settei.preset — Richer presets for several frameworks

### settei.preset.celery — Preset for Celery

**class** `settei.preset.celery.WorkerConfiguration` (`config: typing.Mapping[str, object] = {}, **kwargs`)

The application object mixin which holds configuration for Celery.

**on\_worker\_loaded** (`app`)

Be invoked when a Celery app is ready.

**Parameters** `app` (`celery.Celery`) – a ready celery app

**worker\_broker\_url**

The url of the broker used by Celery. See also Celery's and Kombu's docs about broker urls:

<http://docs.celeryproject.org/en/latest/configuration.html#broker-url>    <http://kombu.readthedocs.org/en/latest/userguide/connections.html#connection-urls>

**worker\_config**

(`typing.Mapping[str, object]`) The configuration mapping for worker that will go to `Celery.conf`.

**worker\_result\_backend**

The backend used by Celery to store task results. See also Celery's docs about result backends:

<http://docs.celeryproject.org/en/latest/configuration.html#celery-result-backend>

### settei.preset.flask — Preset for Flask apps

### settei.preset.logging — Preset for logging configuration

Preset for apps holding `logging` configuration. Logging can be configured through TOML file e.g.:

```
[logging]
version = 1

[logging.loggers.flask]
handlers = ["stderr"]

[logging.loggers."urllib.request"]
```

```
handlers = ["stderr"]

[logging.loggers.werkzeug]
handlers = ["stderr"]

[logging.handlers.stderr]
class = "logging.StreamHandler"
level = "INFO"
stream = "ext://sys.stderr"
```

```
class settei.presets.logging. LoggingConfiguration ( config: typing.Mapping[str, object] =  
                                                    {}, **kwargs)
```

Hold configuration for `logging`.

```
configure_logging ( ) → None  
    Configure logging.
```

### 2.1.3 settei.version — Version data

```
settei.version. VERSION = '0.2.0'  
    (str) The version string e.g. '1.2.3'.
```

```
settei.version. VERSION_INFO = (0, 2, 0)  
    (typing.Tuple`[:class:`int, int, int]`) The triple of version numbers e.g. (1, 2, 3).
```



---

## Indices and tables

---

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



## S

- `settei`, 5
- `settei.base`, 5
- `settei.presets`, 6
  - `settei.presets.celery`, 6
  - `settei.presets.flask`, 6
  - `settei.presets.logging`, 6
- `settei.version`, 7





## C

`config_property` (class in `settei.base`), 5

`Configuration` (class in `settei.base`), 5

`configure_logging()` (`settei.presets.logging.LoggingConfiguration` method), 7

`ConfigWarning`, 5

## D

`docstring` (`settei.base.config_property` attribute), 6

## F

`from_file()` (`settei.base.Configuration` class method), 5

`from_path()` (`settei.base.Configuration` class method), 5

## L

`LoggingConfiguration` (class in `settei.presets.logging`), 7

## O

`on_worker_loaded()` (`settei.presets.celery.WorkerConfiguration` method), 6

## S

`settei` (module), 5

`settei.base` (module), 5

`settei.presets` (module), 6

`settei.presets.celery` (module), 6

`settei.presets.flask` (module), 6

`settei.presets.logging` (module), 6

`settei.version` (module), 7

## V

`VERSION` (in module `settei.version`), 7

`VERSION_INFO` (in module `settei.version`), 7

## W

`worker_broker_url` (`settei.presets.celery.WorkerConfiguration` attribute), 6

`worker_config` (`settei.presets.celery.WorkerConfiguration` attribute), 6

`worker_result_backend` (`settei.presets.celery.WorkerConfiguration` attribute), 6

`WorkerConfiguration` (class in `settei.presets.celery`), 6