Selector Documentation

Release 0.9.0

Luke Arno

Sep 27, 2017

Contents

1 selector Module 3

2 Indices and tables 5

Python Module Index 7

Narrative documentation available from http://github.com/lukearno/selector

Contents:

CHAPTER 1

selector Module

selector - WSGI handler delegation. (AKA routing.)

class `selector.`**`ByMethod`**

Bases: `object`

Base class for dispatching to method named by REQUEST_METHOD.

class `selector.`**`EnvironDispatcher`**(*rules*)

Bases: `object`

Dispatch based on list of rules.

exception `selector.`**`MappingFileError`**

Bases: `exceptions.Exception`

Raised to signal a syntax error in a mapping file.

class `selector.`**`MiddlewareComposer`**(*app*, *rules*)

Bases: `object`

Compose middleware based on list of rules.

class `selector.`**`Naked`**

Bases: `object`

Naked object style dispatch base class.

exception `selector.`**`PathExpressionParserError`**

Bases: `exceptions.Exception`

Raised to signal a syntax error in a path expression.

class `selector.`**`Selector`**(*mappings=None*, *prefix=''*, *parser=None*, *wrap=None*, *mapfile=None*, *consume_path=True*)

Bases: `object`

WSGI middleware for URL paths and HTTP method based delegation.

`add`(*path*, *method_dict=None*, *prefix=None*, ***http_methods*)

Add a mapping.

HTTP methods can be specified in a dict or using key word args, but kwargs will override if both are given.

select (*path*, *method*)
Figure out which app to delegate to or send 404 or 405.

slurp (*mappings*, *prefix=None*, *parser=None*, *wrap=None*)
Slurp in a whole list (or any iterable) of mappings.

Mappings take the form of

```
(PATH_EXPRESSION, HTTP_METHOD_TO_WSGI_APP_DICT)
```

slurp_file (*filename*, *prefix=None*, *parser=None*, *wrap=None*)
Read mappings from a simple text file. (See README.md.)

static status404 (*environ*, *start_response*)
Default WSGI 404 app.

static status405 (*environ*, *start_response*)
Default WSGI 405 app.

class selector.**SimpleParser** (*patterns=None*)
Bases: object

Callable to turn path expressions into regexes with named groups.

```
SimpleParser()("/hello/{name}") == r"^\/hello\/(?P<name>[^\^.]+)$"
```

See README.md for details.

default_pattern = 'chunk'

end = '}'

oend = ']'

ostart = '['

start = '{'

selector.**expose** (*obj*)
Set obj._exposed = True and return obj.

selector.**method_not_allowed** (*environ*, *start_response*)
Default WSGI 405 app.

selector.**not_found** (*environ*, *start_response*)
Default WSGI 404 app.

selector.**opliant** (*meth*)
Decorate a bound wsgi callable taking args from wsgiorg.routing_args.

```
class App(object):
    @opliant
    def __call__(self, environ, start_response, arg1, arg2, foo='bar'):
        ...
```

selector.**pliant** (*func*)
Decorate an unbound wsgi callable taking args from wsgiorg.routing_args.

```
@pliant
def app(environ, start_response, arg1, arg2, foo='bar'):
    ...
```

CHAPTER 2

Indices and tables

- genindex
- modindex
- search

Python Module Index

S

selector, 3

Index

A

add() (selector.Selector method), 3

B

ByMethod (class in selector), 3

D

default_pattern (selector.SimpleParser attribute), 4

E

end (selector.SimpleParser attribute), 4
EnvironDispatcher (class in selector), 3
expose() (in module selector), 4

M

MappingFileError, 3
method_not_allowed() (in module selector), 4
MiddlewareComposer (class in selector), 3

N

Naked (class in selector), 3
not_found() (in module selector), 4

O

oend (selector.SimpleParser attribute), 4
opliant() (in module selector), 4
ostart (selector.SimpleParser attribute), 4

P

PathExpressionParserError, 3
pliant() (in module selector), 4

S

select() (selector.Selector method), 4
Selector (class in selector), 3
selector (module), 3
SimpleParser (class in selector), 4
slurp() (selector.Selector method), 4
slurp_file() (selector.Selector method), 4
start (selector.SimpleParser attribute), 4
status404() (selector.Selector static method), 4
status405() (selector.Selector static method), 4