
WebDispatch Documentation

Release 1.0b1

Atsushi Odagiri

November 18, 2014

1	Basic Usage	3
1.1	Install	3
1.2	URL Dispatch	3
1.3	Method Dispatch	3
1.4	Action Dispatch	3
2	Tips and Tricks	5
2.1	integrate with WebOb	5
2.2	generate absolute url on backend of reverse proxy.	5
3	Reference	7
4	Indices and tables	9
	Python Module Index	11


```
1  from webob import Request
2  from webob.dec import wsgify
3  from webdispatch import URLDispatcher
4  from webdispatch.mixins import URLMapperMixin
5
6  class MyRequest(Request, URLMapperMixin):
7      pass
8
9  app = URLDispatcher()
10
11 @wsgify(RequestClass=MyRequest)
12 def index(request):
13     url = request.generate_url("hello", xname="webdispatch")
14     return '<a href="%s">Hello</a>' % (url,)
15
16 @wsgify(RequestClass=MyRequest)
17 def hello(request):
18     return "Hello %s" % request.urlvars['xname']
19
20 app.add_url('home', '/', index)
21 app.add_url('hello', '/hello/{xname}', hello)
22
23 if __name__ == '__main__':
24     from wsgiref.simple_server import make_server
25
26     httpd = make_server('0.0.0.0', 8080, app)
27     httpd.serve_forever()
```

Contents:

Basic Usage

1.1 Install

```
$ pip install WebDispatch
```

1.2 URL Dispatch

```
webdispatch.urldispatcher.URLDispatcher

def hello(envIRON, start_response):
    start_response("200 OK",
        [('Content-type', 'text/plain')])
    return ["hello"]

def goodbye(envIRON, start_response):
    start_response("200 OK",
        [('Content-type', 'text/plain')])
    return ["goodby %s" % envIRON['wsgiorg.routing_args'][1]['name']]

from webdispatch import URLDispatcher
application = URLDispatcher()
application.add_url('hello', '/hello', hello)
application.add_url('goodbye', '/good-bye', goodbye)

from wsgiref.simple_server import make_server

httpd = make_server('0.0.0.0', 8080, application)
httpd.serve_forever()
```

this is very primitive wsgi application.

1.3 Method Dispatch

```
webdispatch.methoddispatcher.MethodDispatcher
```

1.4 Action Dispatch

Tips and Tricks

2.1 integrate with WebOb

WebOb has wsgify decorator that makes callable wsgi application.

WebDispatch provides `webdispatch.mixins.URLMapperMixin` to add `generate_url` method to Request class.

use generator mixin:

```
class MyRequest(Request, URLMapperMixin):
    pass

@wsgify(RequestClass=MyRequest)
def view(request):
    return "Hello"
```

2.2 generate absolute url on backend of reverse proxy.

When application is forwarded by reverse proxy, below headers are given.

- HTTP_X_FORWARDED_SERVE
- HTTP_X_FORWARDED_HOST
- HTTP_X_FORWARDED_FOR
- HTTP_X_FORWARDED_SCHEME
- HTTP_X_FORWARDED_PROTO

use `paste.deploy.config.PrefixMiddleware`.

```
[pipeline:main]
pipeline =
    prefix
    app

[filter:prefix]
use = egg:PasteDeploy#prefix
scheme = https
```

Reference

urldispatcher

class webdispatch.urldispatcher.**URLDispatcher** (*applications=None, extra_environ=None, **kwargs*)

dispatch applications with url patterns.

add_subroute (*pattern*)

create new URLDispatcher routed by pattern

add_url (*name, pattern, application*)

add url pattern dispatching to application

detect_view_name (*environ*)

detect view name from environ

on_view_not_found (*environ, start_response*)

called when views not found

class webdispatch.urldispatcher.**URLGenerator** (*environ, urlmapper*)

generate url form parameters and url patterns.

generate (*name, **kwargs*)

generate full qualified url for named url pattern with kwargs

make_full_qualified_url (*path*)

append application url to path

class webdispatch.urldispatcher.**URLMapper** (*converters=None*)

find application matched url pattern.

add (*name, pattern*)

add url pattern for name

generate (*name, **kwargs*)

generate url for named url pattern with kwargs

lookup (*path_info*)

lookup url match for path_info

methoddispatcher

class webdispatch.methoddispatcher.**ActionDispatcher** (*action_var_name='action'*)

wsgi application dispatching actions to registered classes

detect_view_name (*environ*)

get view name from routing args

on_view_not_found (*environ*, *start_response*)
called when action is not found

register_actionhandler (*action_handler*)
register class as action handler

class webdispatch.methoddispatcher.**MethodDispatcher** (***kwargs*)
dispatch applications with request method.

detect_view_name (*environ*)
convert request method to view name

on_view_not_found (*_*, *start_response*)
called when valid view is not found

webdispatch.methoddispatcher.**action_handler_adapter** (*handler_cls*, *action_name*)
wraps class to wsgi application dispatching action

useful mixin classes

class webdispatch.mixins.**URLMapperMixin**
mixin to add `generate_url()` method.

generate_url (*name*, ***kwargs*)
generate url with urlgenerator used by urldispatch

urlmapper
get urlmapper object from wsgi environ

Indices and tables

- *genindex*
- *modindex*
- *search*

W

`webdispatch.methoddispatcher`, [7](#)

`webdispatch.mixins`, [8](#)

`webdispatch.urldispatcher`, [7](#)

A

`action_handler_adapter()` (in module `webdispatch.methoddispatcher`), 8

`ActionDispatcher` (class in `webdispatch.methoddispatcher`), 7

`add()` (`webdispatch.urldispatcher.URLMapper` method), 7

`add_subroute()` (`webdispatch.urldispatcher.URLDispatcher` method), 7

`add_url()` (`webdispatch.urldispatcher.URLDispatcher` method), 7

D

`detect_view_name()` (`webdispatch.methoddispatcher.ActionDispatcher` method), 7

`detect_view_name()` (`webdispatch.methoddispatcher.MethodDispatcher` method), 8

`detect_view_name()` (`webdispatch.urldispatcher.URLDispatcher` method), 7

G

`generate()` (`webdispatch.urldispatcher.URLGenerator` method), 7

`generate()` (`webdispatch.urldispatcher.URLMapper` method), 7

`generate_url()` (`webdispatch.mixins.URLMapperMixin` method), 8

L

`lookup()` (`webdispatch.urldispatcher.URLMapper` method), 7

M

`make_full_qualified_url()` (`webdispatch.urldispatcher.URLGenerator` method), 7

`MethodDispatcher` (class in `webdispatch.methoddispatcher`), 8

O

`on_view_not_found()` (`webdispatch.methoddispatcher.ActionDispatcher` method), 7

`on_view_not_found()` (`webdispatch.methoddispatcher.MethodDispatcher` method), 8

`on_view_not_found()` (`webdispatch.urldispatcher.URLDispatcher` method), 7

R

`register_actionhandler()` (`webdispatch.methoddispatcher.ActionDispatcher` method), 8

U

`URLDispatcher` (class in `webdispatch.urldispatcher`), 7

`URLGenerator` (class in `webdispatch.urldispatcher`), 7

`URLMapper` (class in `webdispatch.urldispatcher`), 7

`urlmapper` (`webdispatch.mixins.URLMapperMixin` attribute), 8

`URLMapperMixin` (class in `webdispatch.mixins`), 8

W

`webdispatch.methoddispatcher` (module), 7

`webdispatch.mixins` (module), 8

`webdispatch.urldispatcher` (module), 7