WebDispatch Documentation

Release

2011, Atsushi Odagiri

October 20, 2011

CONTENTS

```
from webob import Request
from webob.dec import wsgify
from webdispatch import URLDispatcher
from webdispatch.mixins import URLMapperMixin

class MyRequest(Request, URLMapperMixin):
    pass

app = URLDispatcher()

@wsgify(RequestClass=MyRequest)
def index(request):
    url = request.generate_url("hello", xname="webdispatch")
    return '<a href="%s">Hello</a>' % (url,)

@wsgify(RequestClass=MyRequest)
def hello(request):
    return "Hello %s" % request.urlvars['xname']

app.add_url('home', '/', index)
app.add_url('hello', '/hello/{xname}', hello)

if __name__ == '__main__':
    from wsgiref.simple_server import make_server

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

Contents:

CHAPTER

ONE

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

CHAPTER

TWO

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
```

CHAPTER

THREE

REFERENCE

CHAPTER

FOUR

INDICES AND TABLES

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