Werkzeug Cheat Sheet

(Base)Request

environ	the WSGI environment
headers	request headers
args	URL arguments
form	submitted form data
files	submitted files
values	URL args + form data
content_type	the content type
path	the current path
method	the request method (GET ...)

(Base)Response

__init__(response, status, headers, mimetype, content_type)

response	the response iterable
headers	the response headers
status[_code]	status line / code
data	response iterable as string
set_cookie(...)	helper to set cookies

import_string(...)

takes a dotted name and returns the imported object. Second parameter True and on ImportErrors None is returned.

find_modules(...)

pass a dotted name and it will return an iterator over all the modules inside that package. Useful for bootstrapping.

FileStorage

filename	the filename on the client
name	the name of the form field
content_type	the content type
content_length	the length of the file
save(filename)	helper method to copy the upload
stream	the underlaying file pointer

redirect(...)

location	where to redirect? relative to script name or absolute.
code	the redirect code. default is 301

useful codes: 301 permanent; 302 found; 303 see other; 307 temporary

run_server(...)

hostname	the hostname ('localhost')
port	the port (8080)
application	the WSGI application
use_reloader	enable automatic code reloading?
use_debugger	enable the interactive debugger?

responder(...)

a decorator for WSGI applications. The decorated function takes two parameters (env, start_response) and the return value is called with those.

Also: decorate using Request.application to achieve the same however the function is passed a request object rather than (env, start_response)

Example Application

```
from werkzeug import Request,
    Response, run_simple

def index(request):
    return Response('''
       <!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN">
       <title>Index Page</title>
       <h1>Index Page</h1>
       <p><a href="hello?name=World">Hello World</a></p>
    ''')

def hello(request):
    name = request.values.get('name', 'World')
    return Response("Hello %s!",
        mimetype='text/plain')

@Request.application
def application(request):
    if request.path == "/":
        return index(request)
    elif request.path == '/hello':
        return hello(request)
    return Response("Not Found", status=404,
        mimetype='text/plain')

if __name__ == '__main__':
    run_simple('localhost', 4000, application)
```