


Join GitHub today

GitHub is home to over 50 million developers working together to host and review code, manage projects, and build software together.

Sign up

Dismiss

 More documentation

Latest commit 7a75a9e on 23 Feb 2013

History

1 contributor

82 lines (66 sloc) | 2.5 KB

RawBlame

```
1 # Run a WSGI application in a daemon thread
2
3 import bottle
4 import threading
5 import socket
6 import time as _time
7
8 class Server(bottle.WSGIRefServer):
9     def run(self, handler): # pragma: no cover
10         from wsgiref.simple_server import make_server, WSGIRequestHandler
11         if self.quiet:
12             base = self.options.get('handler_class', WSGIRequestHandler)
13             class QuietHandler(base):
14                 def log_request(*args, **kw): pass
15             self.options['handler_class'] = QuietHandler
16         self.srv = make_server(self.host, self.port, handler, **self.options)
17         self.srv.serve_forever(poll_interval=0.1)
18
19 def start_bottle_server(app, port, **kwargs):
20     server_thread = ServerThread(app, port, kwargs)
21     server_thread.daemon = True
22     server_thread.start()
23
24     ok = False
25     for i in range(10):
26         try:
27             conn = socket.create_connection(('127.0.0.1', port), 0.1)
28             except socket.error as e:
29                 _time.sleep(0.1)
30             else:
31                 conn.close()
32                 ok = True
33                 break
34     if not ok:
35         import warnings
36         warnings.warn('Server did not start after 1 second')
37
38     return server_thread.server
39
40 class ServerThread(threading.Thread):
41     def __init__(self, app, port, server_kwargs):
42         threading.Thread.__init__(self)
43         self.app = app
44         self.port = port
45         self.server_kwargs = server_kwargs
46         self.server = Server(host='localhost', port=self.port, **self.server_kwargs)
47
48     def run(self):
49         bottle.run(self.app, server=self.server, quiet=True)
50
51 def app_runner_setup(*specs):
52     '''Returns setup and teardown methods for running a list of WSGI
53     applications in a daemon thread.
54
55     Each argument is an (app, port) pair.
56
57     Return value is a (setup, teardown) function pair.
58
59     The setup and teardown functions expect to be called with an argument
60     on which server state will be stored.
61
62     Example usage with nose:
63
64     >>> setup_module, teardown_module = \
65         webracer.utils.runwsgi.app_runner_setup((app_module.app, 8050))
66     ...
67
68     def setup(self):
69         self.servers = []
70         for spec in specs:
71             if len(spec) == 2:
72                 app, port = spec
73                 kwargs = {}
74             else:
75                 app, port, kwargs = spec
76             self.servers.append(start_bottle_server(app, port, **kwargs))
77
78     def teardown(self):
79         for server in self.servers:
80             server.srv.shutdown()
81
82     return [setup, teardown]
```