

twill

twill is a simple scripting language for browsing the Web. Designed for *functional Web testing*.

Author: Titus Brown

Inspired by Cory Dodt's PBP, built around mechanize, BeautifulSoup, and pyparsing.

twill

```
% twill-sh
```

```
>> go http://www.google.com/
```

```
>> formvalue 1 q "guido van rossum"
```

```
>> submit btnI
```

```
>> echo __url__
```

```
http://www.python.org/~guido/
```

twill

```
from twill.commands import *  
  
go('http://www.google.com/')  
formvalue('l', 'q', 'guido van rossum')  
submit('btnI')  
print get_browser.get_url()
```

twill Features

Follows redirects.

Fills & submits forms.

Handles/manipulates cookies.

Allows for simple HTTP authentication.

Simple & easy extension architecture.

Sorry, it doesn't understand JavaScript :(

Organizing simple twill tests

```
% twill-sh <list of files>
```

Running twill tests with nose

```
def setup(module):  
    module.url = testlib.run_server()  
  
def teardown(module):  
    testlib.shutdown(module.url)  
  
def test1():  
    go(url)  
    code(200)  
  
def test2():  
    twill.execute_script('test_stuff.twill', url=url)
```

Voodoo: getting rid of 'run_server'

```
def setup():
    wsgi_app = testlib.get_test_server_as_wsgi()
    twill.add_wsgi_intercept('localhost', 80,
                             wsgi_app)

def teardown():
    twill.remove_wsgi_intercept('localhost', 80)

def test():
    go('http://localhost:80')
    code(200)
```

Writing twill extensions is *easy*

(it's Python, after all!)

File 'hello.py':

```
def hello(what):
```

```
    """
```

```
    >> hello <what>
```

```
    Says 'hello' to <what>
```

```
    """
```

```
    print 'Hello', what, '!'
```

Twill miscellany

A prototype twill-script recorder exists, called 'scotch'.

Twill book will soon be available.

No plans for ruining my life by embedding JavaScript.

Code coverage analysis with figleaf

figleaf is a simple® rewrite of Ned Batchelder's coverage.py.

It's intended for programmatic use.

Basic API:

```
figleaf.start()
```

```
figleaf.stop()
```

```
figleaf.write_coverage('.figleaf')
```

Good code coverage is
necessary but not *sufficient*