UI Testing with Selenium and Robot Framework

Ed Manlove

irc: t55e

devPyPlTw@verizon.net

Follow along at

github.com/emanlove/pse2012

github.com/emanlove/pse2012

Unit Testing

Integration Testing

Functional Testing

Acceptance Testing

github.com/emanlove/pse2012

Unit Testing

Integration Testing

Functional Testing

Acceptance Testing

Testing the
User Interface
(UI)

github.com/emanlove/pse2012

?

Language	Code Lines
Python	235,385
Javascript	121,994
XML	58,370
HTML	12,985
CSS	8,642
...	...

Source: Ohloh.net

Language	Code Lines
Python	235,385
Javascript	121,994
XML	58,370
HTML	12,985
CSS	8,642
...	...

◁ 28 %

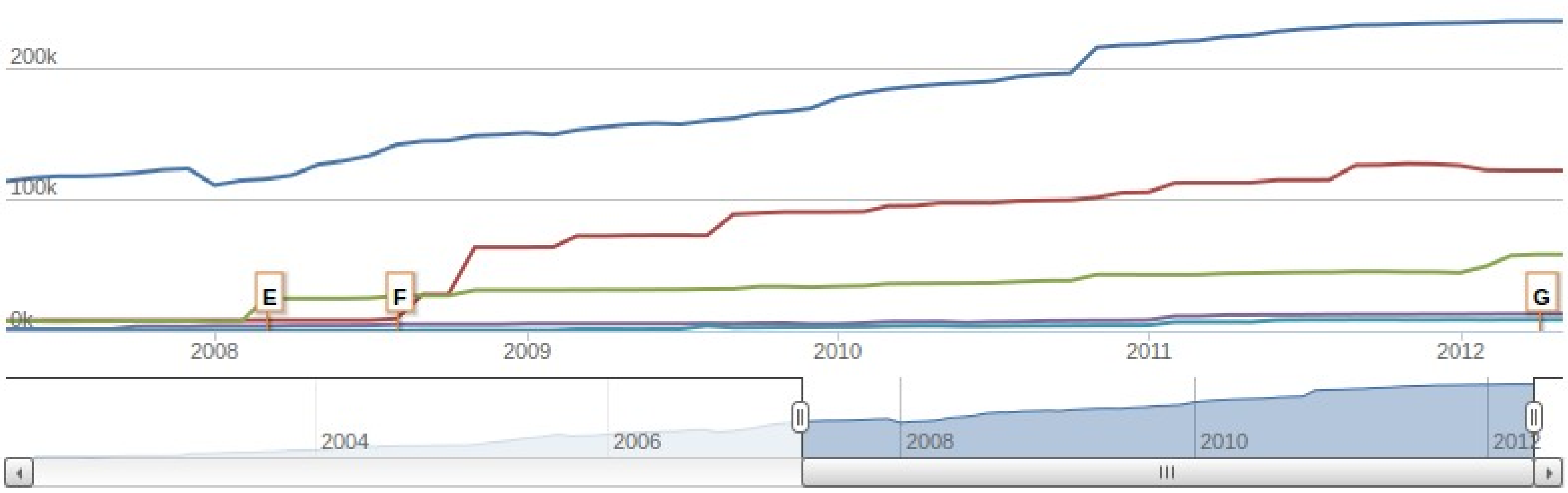
Source: Ohloh.net

Language	Code Lines
Python	235,385
Javascript	121,994
XML	58,370
HTML	12,985
CSS	8,642
...	...

△

1/2

Source: Ohloh.net



Source: Ohloh.net

Language	Code Lines
Python	235,385
Javascript	121,994
XML	58,370
HTML	12,985
CSS	8,642
...	...
?	?

Source: Ohloh.net

Language	Code Lines
Python	235,385
Javascript	121,994
XML	58,370
HTML	12,985
CSS	8,642
...	...
PHP (?!?)	129

Source: Ohloh.net

QUnit ?

Selenium 2.0 / WebDriver

Selenium 2.0 / WebDriver

```
import unittest2 as unittest
import transaction
from plone.app.testing.selenium_layers import SELENIUM_PLONE_FUNCTIONAL_TESTING

class TestPortlets(unittest.TestCase):
    layer = SELENIUM_PLONE_FUNCTIONAL_TESTING

    def setUp(self):
        self.driver = self.layer['selenium']
        self.portal = self.layer['portal']
        self.driver.implicitly_wait(5)

        self.portal.acl_users._doAddUser('member1', 'secret',
                                         ['Member'], [])

    def test_login_failed(self):
        # ensure we have a clean starting point
        transaction.commit()
        self.driver.get(self.portal.absolute_url() + '/login_form')
        self.driver.find_element_by_name('__ac_name').send_keys('member1')
        self.driver.find_element_by_name('__ac_password').send_keys('badpassword')
        self.driver.find_element_by_name('submit').click()
        self.assertTrue("Login failed" in self.driver.page_source)
    def test_login_overlay_successful(self):
        # ensure we have a clean starting point
        transaction.commit()

        self.driver.get(self.portal.absolute_url())

        self.driver.find_element_by_id('personaltools-login').click()
        self.assertTrue(self.driver.find_element_by_id('div.overlay-ajax
form#login_form').is_displayed())

        self.driver.find_element_by_name('__ac_name').send_keys('member1')
        self.driver.find_element_by_name('__ac_password').send_keys('secret')
        self.driver.find_element_by_name('submit').click()

        self.assertTrue("You are now logged in" in self.driver.page_source)
```

```
from plone.testing import Layer

class SeleniumLayer(Layer):
    defaultBases = (z2.ZSERVER_FIXTURE, )

    def testSetUp(self):
        # Start up Selenium
        driver = os.environ.get('SELENIUM_DRIVER', '').lower() or 'firefox'
        webdriver = __import__(
            'selenium.webdriver.%s.webdriver' % driver, fromlist=['WebDriver'])
        args = [arg.strip() for arg in
                os.environ.get('SELENIUM_ARGS', '').split()
                if arg.strip()]
        self['selenium'] = webdriver.WebDriver(*args)

    def testTearDown(self):
        self['selenium'].quit()
        del self['selenium']

SELENIUM_FIXTURE = SeleniumLayer()
SELENIUM_FUNCTIONAL_TESTING = FunctionalTesting(
    bases=(SELENIUM_FIXTURE, ),
    name="SeleniumTesting:Functional")
SELENIUM_PLONE_FUNCTIONAL_TESTING = FunctionalTesting(
    bases=(SELENIUM_FIXTURE, PLONE_FIXTURE),
    name="SeleniumTesting:Functional")
    def test_login_overlay_successful(self):
        # ensure we have a clean starting point
        transaction.commit()

        self.driver.get(self.portal.absolute_url())

        self.driver.find_element_by_id('personaltools-login').click()
        self.assertTrue(self.driver.find_element_by_id('div.overlay-ajax
form#login_form').is_displayed())

        self.driver.find_element_by_name('__ac_name').send_keys('member1')
        self.driver.find_element_by_name('__ac_password').send_keys('secret')
        self.driver.find_element_by_name('submit').click()

        self.assertTrue("You are now logged in" in self.driver.page_source)
    def test_login_overlay_successful(self):
        # ensure we have a clean starting point
        transaction.commit()

        self.driver.get(self.portal.absolute_url())

        self.driver.find_element_by_id('personaltools-login').click()
        self.assertTrue(self.driver.find_element_by_id('div.overlay-ajax
form#login_form').is_displayed())

        self.driver.find_element_by_name('__ac_name').send_keys('member1')
        self.driver.find_element_by_name('__ac_password').send_keys('secret')
        self.driver.find_element_by_name('submit').click()

        self.assertTrue("You are now logged in" in self.driver.page_source)
    def test_login_overlay_successful(self):
        # ensure we have a clean starting point
        transaction.commit()

        self.driver.get(self.portal.absolute_url())

        self.driver.find_element_by_id('personaltools-login').click()
        self.assertTrue(self.driver.find_element_by_id('div.overlay-ajax
form#login_form').is_displayed())

        self.driver.find_element_by_name('__ac_name').send_keys('member1')
        self.driver.find_element_by_name('__ac_password').send_keys('secret')
        self.driver.find_element_by_name('submit').click()

        self.assertTrue("You are now logged in" in self.driver.page_source)
```

Solution: Page Objects

LOCATORS ELEMENTS PAGE

```
from basepage import BasePage
from pageelements import BaseTextElement
from selenium.webdriver.common.by import By
from seleniumwrapper import SeleniumWrapper as wrapper


locators = {
    "username": [By.NAME, "__ac_name"],
    "password": [By.NAME, "__ac_password"],
    "submit": [By.NAME, "submit"],
    "status_message": [By.CSS_SELECTOR, "dl.portalMessage:last-of-type dd"]
}

...
```

LOCATORS ELEMENTS PAGE

```
...

class UsernameElement(BaseTextElement):
    def __init__(self, selenium):
        self.se = selenium
        self.locator = locators["username"]


class PasswordElement(BaseTextElement):
    def __init__(self, selenium):
        self.se = selenium
        self.locator = locators["password"]


class StatusMessageTextElement(BaseTextElement):
    def __init__(self, selenium):
        self.se = selenium
        self.locator = locators["status_message"]

...
```

LOCATORS ELEMENTS PAGE

```
...

class LoginPage(BasePage):

    def __init__(self, selenium):
        self.se = selenium

        self.username = UsernameElement(self.se)
        self.password = PasswordElement(self.se)
        self.status_message = StatusMessageTextElement(self.se)

    def submit(self):
        self.se.find_element(*locators["submit"]).click()
```

```
from plone.seleniumtesting.loginpage import LoginPage

    def test_login_successful(self):

        lpo = LoginPage()
        lpo.open()
        lpo.username = 'member1'
        lpo.password = 'secret'
        lpo.submit()
        self.assertEqual("You are now logged in", lpo.status_message)
```

```
    def test_login_successful(self):
        # ensure we have a clean starting point
        transaction.commit()
        self.driver.get(self.portal.absolute_url() + '/login_form')
        self.driver.find_element_by_name('__ac_name').send_keys('member1')
        self.driver.find_element_by_name('__ac_password').send_keys('secret')
        self.driver.find_element_by_name('submit').click()
        self.assertTrue("You are now logged in" in self.driver.page_source)
```

```
from plone.seleniumtesting.loginpage import LoginPage

    def test_login_successful(self):

        lpo = LoginPage()
        lpo.open()
        lpo.username = 'member1'
        lpo.password = 'secret'
        lpo.submit()
        self.assertEqual("You are now logged in", lpo.status_message)
```

Selenium

Robot Framework

Robot Framework

- Text-based file format
- Keyword-driven approach to testing
- Extensible through custom libraries
- Modular architecture

```
Test Login Successful
    Open login overlay
    Login with credentials  admin  admin
    Overlay should be gone
```

```
*** Test cases ***

Test Login Successful
    Open login overlay
    Login with credentials  admin  admin
    Overlay should be gone

*** Keywords ***

Open login overlay
    Click Element  ${personaltools_login_selector}
    Page Should Contain Element  ${overlay_login_selector}
    Page should contain element  __ac_name

Login with credentials  [Arguments]  ${username}  ${password}
    Input text  __ac_name      ${username}
    Input text  __ac_password  ${password}
    Click Button  Log in
    Wait Until Page Contains Element  ${personaltools_logout_selector}

Overlay should be gone
    Page Should Not Contain Element  ${overlay_login_selector}
```

```
*** Settings ***
Library     Selenium2Library  timeout=0 seconds  implicit_wait=5 seconds

Suite setup

Test setup  Log out and go home

*** Variables ***

${overlay_login_selector} =         css=div.overlay-ajax form#login_form
${personaltools_login_selector} =   css=#personaltools-login
${personaltools_logout_selector} =  css=#personaltools-logout

*** Test cases ***

Test Login Successful
    Open login overlay
    Login with credentials  admin  admin
    Overlay should be gone

*** Keywords ***

Open login overlay
    Click Element  ${personaltools_login_selector}
    Page Should Contain Element  ${overlay_login_selector}
    Page should contain element  __ac_name
...
```

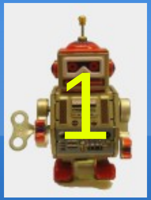
github.com/emanlove/pse2012

- Install Plone

- Install robotframework,

  robotframwork-selenium2library python packages (and dependencies)
- Create test directory and sample test case

- Run the test (c:\ pybot hello_world.txt)



```
[buildout]
parts =
    ...
    robot

auto-checkout +=
    robotframework-selenium2library

[robot]
recipe = zc.recipe.egg
eggs = robotframework
       robotframework-selenium2library
entry-points =
    pybot=robot:run_cli
    rebot=robot.rebot:rebot_cli
```

buildout.cfg (or equivalent)

```
[sources]
...
robotframework-selenium2library     = git
  git://github.com/emanlove/robotframework-selenium2library.git
```

source.cfg (or equivalent)

```
~/plone42$ ./bin/buildout
```

```
. . .

import robot

if __name__ == '__main__':
    robot.run_cli(sys.argv[1:])
```

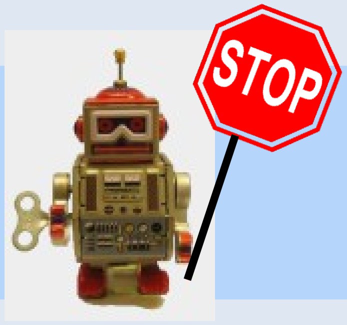
./bin/pybot

```
*** Settings ***
Library     Selenium2Library  timeout=0 seconds  implicit_wait=5 seconds

*** Test cases ***

Hello World
    Go to  http://plone.org
    Click link  Get Involved

    Page should contain  Help make Plone even better
    Page should contain  Create bug fixes, develop new features
```

./acceptance-tests/hello_word.txt

```
~/plone42$ ./bin/pybot acceptance-tests/
```

Issues with



Robot Framework

github.com/emanlove/pse2012

- Visual Editors (TinyMCE, CKEditor)
- New User Interface – CMSUI
- Deco / Tiles / Blocks

github.com/emanlove/pse2012

Thank You

Ed Manlove

irc: t55e

devPyPlTw@verizon.net