Stepik Academy. Автоматизация тестирования на Python. Вебинар №5

Приручаем отчеты в тестах: когда простого "да всё работает!" недостаточно. Используем скриншоты и заставляем Fail'ы работать на нас.

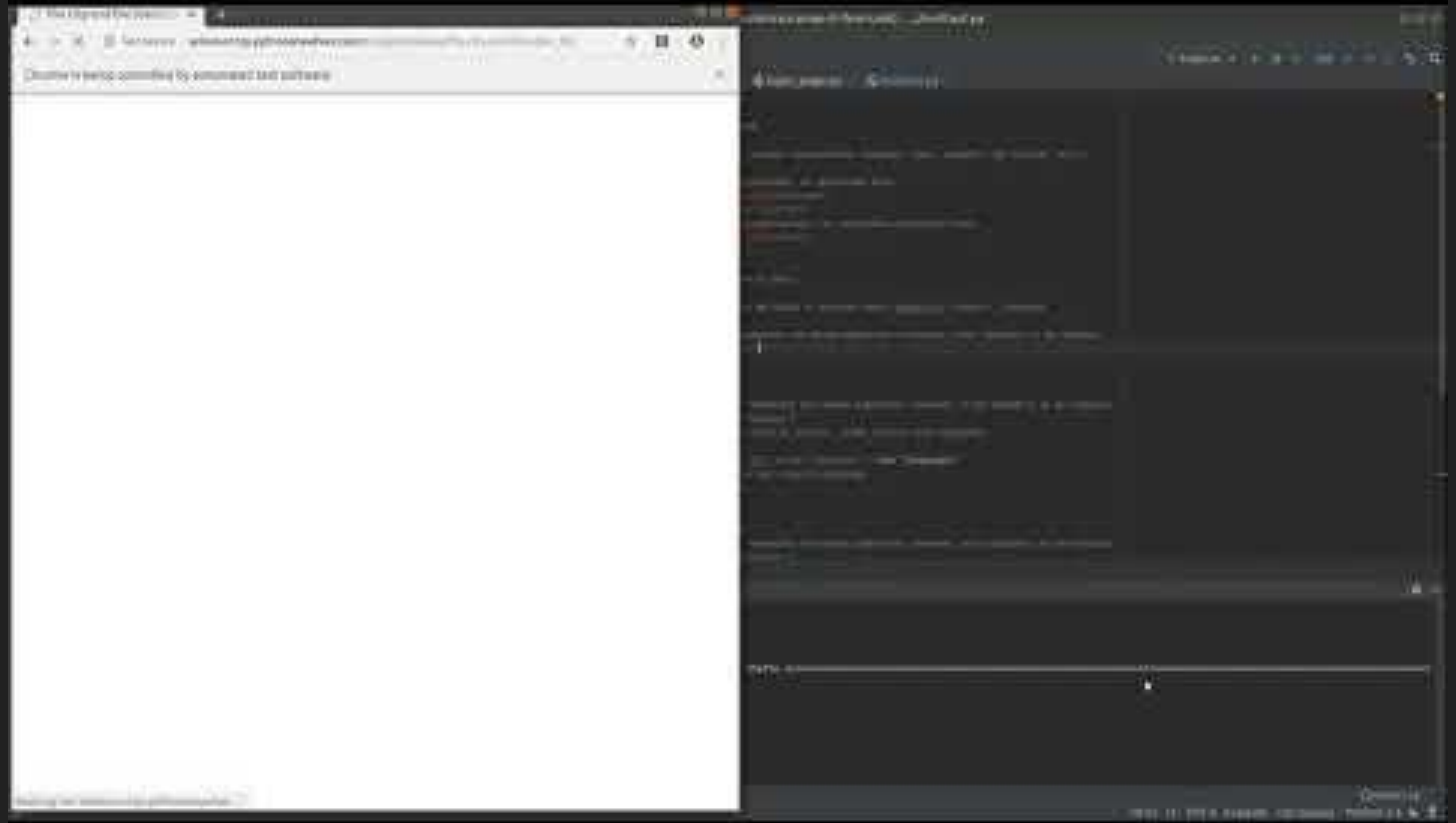
Ещё немного про PyTest

Ещё немного про PyTest

Параллельный запуск тестов с плагином xdist

Ещё немного про PyTest

Параллельный запуск тестов с плагином xdist



Ещё немного про PyTest

Параллельный запуск тестов с плагином xdist

https://pypi.org/project/pytest-xdist/

https://github.com/pytest-dev/pytest-xdist

https://blog.testproject.io/2019/07/16/parallel-test-execution-with-pytest/

https://www.guru99.com/pytest-tutorial.html#9

О чем поговорим сегодня

О чем поговорим сегодня

- Page Object Model - это что и как?

О чем поговорим сегодня

- Page Object Model - это что и как?
- Отчеты о прохождении тестов с Allure

О чем поговорим сегодня

- Page Object Model - это что и как?
- Отчеты о прохождении тестов с Allure
- Визуальное тестирование:
  - С браузерными расширениями
  - Со специальными утилитами

О чем поговорим сегодня

- Page Object Model - это что и как?
- Отчеты о прохождении тестов с Allure
- Визуальное тестирование:
  - С браузерными расширениями
  - Со специальными утилитами
- Скриншот результата теста с помощью возможностей Selenium

Page Object Model

Page Object Model

Page Object Model (Page Object, POM) - паттерн программирования.

Page Object Model

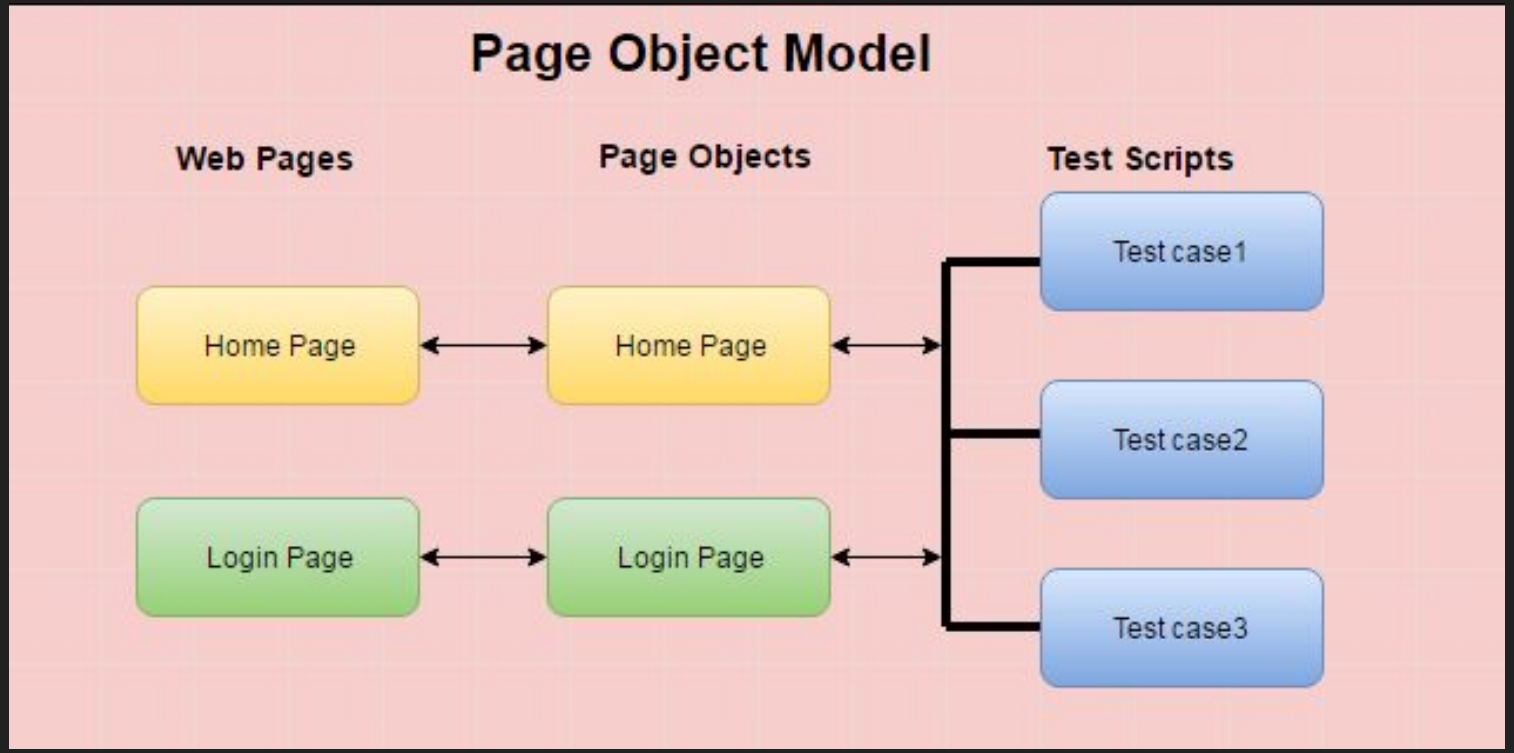
Page Object Model (Page Object, POM) - паттерн программирования.

Каждая страница - экземпляр класса.

Page Object Model

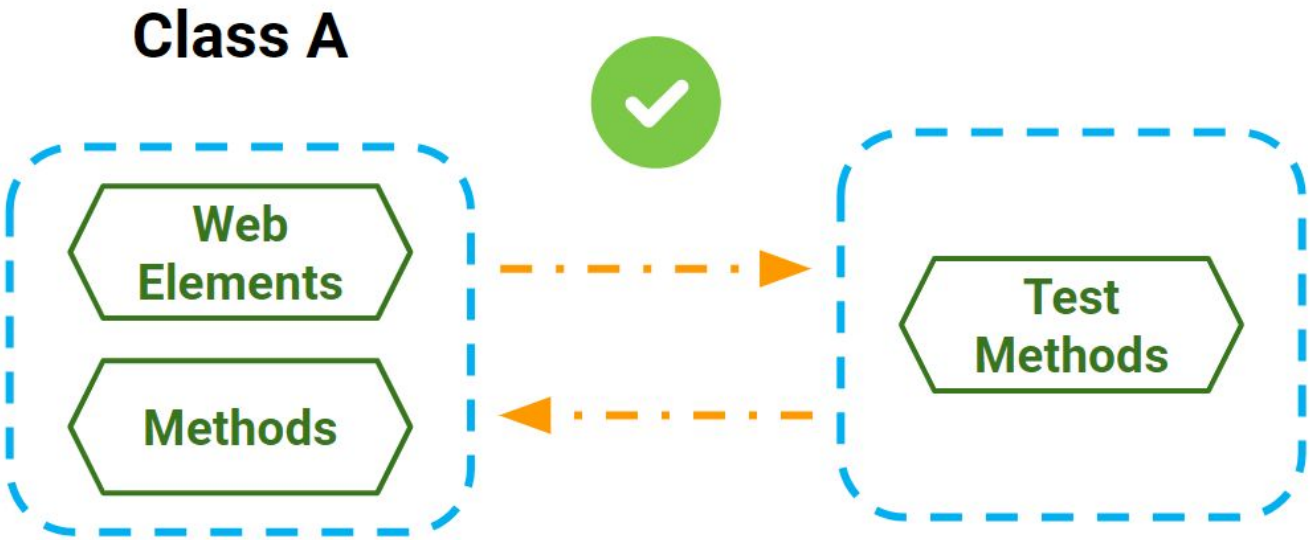
Page Object Model (Page Object, POM) - паттерн программирования.

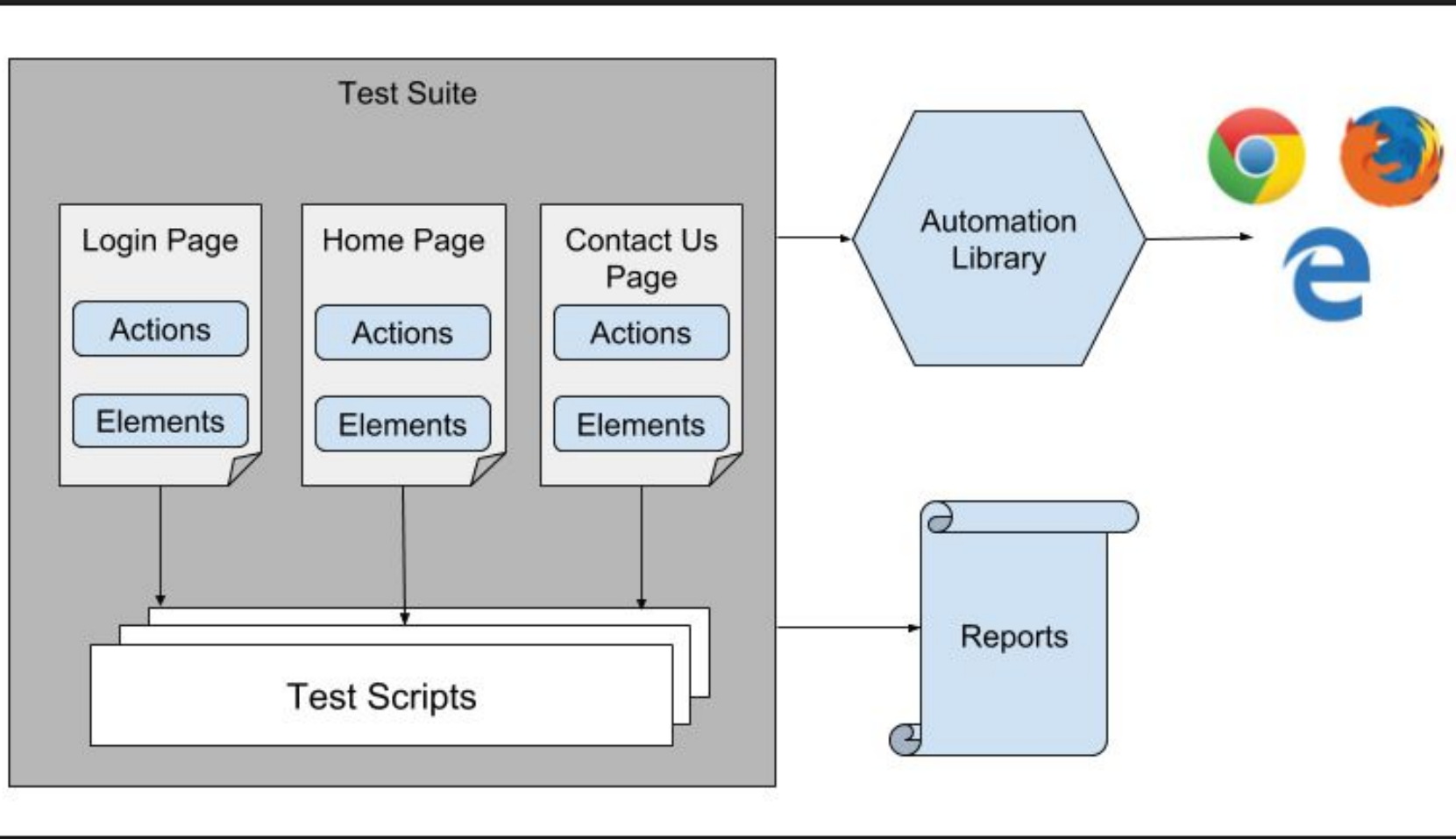
Каждая страница - экземпляр класса.
Действия на странице - методы класса.


Page Object Model
Web Pages
Page Objects
Test Scripts
Home Page
Home Page
Login Page
Login Page
Test case1
Test case2
Test case3

Design Patterns in Automated Testing series

Page Object Model




Test Suite
Login Page
Actions
Elements
Home Page
Actions
Elements
Contact Us Page
Actions
Elements
Test Scripts
Automation Library
Reports

Page Object Model: before

```python
link = "http://selenium1py.pythonanywhere.com/"


def test_user_can_register(browser):
    unique_var = random.randint(1, 1000000)

    browser.get(link)
    login_btn = browser.find_element_by_css_selector("#login_link")
    login_btn.click()

    email = browser.find_element_by_css_selector("#id_registration-email")
    email.clear()
    email.send_keys("{}@example.com".format(unique_var))

    password = browser.find_element_by_css_selector("#id_registration-password1")
    confirmation = browser.find_element_by_css_selector("#id_registration-password2")
    password.send_keys(unique_var, unique_var, unique_var)
    confirmation.send_keys(unique_var, unique_var, unique_var)
    registration_btn = browser.find_element_by_css_selector("[name='registration_submit']")
    registration_btn.click()

    login_success_text = browser.find_element_by_css_selector(".alertinner").text

    assert "Thanks for registering!" in login_success_text
```

Page Object Model: after

```
link = "http://selenium1py.pythonanywhere.com/catalogue/the-shellcoders-handbook_209?promo=midsummer"

class TestSignUp:
    def test_guest_can_go_to_login_link(self, browser, link):
        page = MainPage(browser, link)
        page.open()
        login_page = page.go_to_login_page()
        login_page.fill_email_registration_field()
        login_page.fill_password_registration_field()
        login_page.fill_password_confirmation_field()
        login_page.submit_new_user()
        user_page = UserPage(browser, browser.current_url)
        user_page.should_be_success_confirmation()
```

Page Object Model: after

```
from .base_page import BasePage
from .locators import LoginPageLocators

class LoginPage(BasePage):

    def go_to_login_page(self):
        login_link = self.browser.find_element(*BasePageLocators.LOGIN_LINK)
        login_link.click()

    def fill_email_registration_field(self):
        email_field = self.browser.find_element(*LoginPageLocators.EMAIL_FIELD)
        email_field.send_keys(youremail)

    def fill_password_registration_field(self):
        pw_field = self.browser.find_element(*LoginPageLocators.PW_FIELD)
        pw_field.send_keys(yourpassword)

    def fill_password_confirmation_field(self):
        pw_conf_field = self.browser.find_element(*LoginPageLocators.PW2_FIELD)
        pw_conf_field.send_keys(yourpassword)

    def submit_new_user(self):
        button = self.browser.find_element(*LoginPageLocators.SUBMIT_BUTTON)
        button.click()
```

Page Object Model: after

```
from .base_page import BasePage
from .locators import UserPageLocators

class UserPage(BasePage):
    def should_be_success_confirmation(self):
      success_message = self.browser.find_element(*UserPageLocators.REGISTER_SUCCESS_MODAL)

      assert "Thanks for registering!" in success_message
```

Преимущества Page Object Model

Преимущества Page Object Model

- Избегаем дублирования кода

Преимущества Page Object Model

- Избегаем дублирования кода
- Получаем более высокий уровень абстракции за счет инкапсулирования методов в класс страницы

Преимущества Page Object Model

- Избегаем дублирования кода
- Получаем более высокий уровень абстракции за счет инкапсулирования методов в класс страницы
- Тесты читабельнее и понятнее (больше логики, меньше технического)

Преимущества Page Object Model

- Избегаем дублирования кода
- Получаем более высокий уровень абстракции за счет инкапсулирования методов в класс страницы
- Тесты читабельнее и понятнее (больше логики, меньше технического)
- Одно исправление => поправили везде

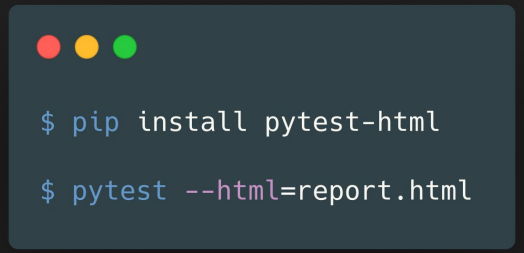
Средства представления отчетов для PyTest

Средства представления отчетов для PyTest

- Плагин PyTest-HTML
  https://github.com/pytest-dev/pytest-html

Средства представления отчетов для PyTest

- Плагин PyTest-HTML
  https://github.com/pytest-dev/pytest-html



report.html

Report generated on 02-Dec-2019 at 13:35:15 by pytest-html v2.0.1

Environment

JAVA_HOME	/home/aozherelyeva/idea-7971
Packages	{'pytest': '5.3.1', 'py': '1.8.0', 'pluggy': '0.13.1'}
Platform	Linux-5.0.0-36-generic-x86_64-with-Ubuntu-18.04-bionic
Plugins	{'xdist': '1.30.0', 'html': '2.0.1', 'rerunfailures': '8.0', 'forked': '1.1.3', 'metadata': '1.8.0'}
Python	3.6.8

Summary

9 tests ran in 84.89 seconds.

(Un)check the boxes to filter the results.

☑ 4 passed, ☐ 0 skipped, ☑ 3 failed, ☑ 2 errors, ☑ 2 expected failures, ☐ 0 unexpected passes, ☐ 0 rerun

Results

Show all details / Hide all details

Result	Test	Duration	Links
Passed *(show details)*	test_main_page.py::test_guest_should_see_login_link	1.39	
Passed *(show details)*	test_product_page.py::TestUserAddToBasketFromProductPage::test_user_can_add_product_to_basket[http://selenium1py.pythonanywhere.com/catalogue/the-city-and-the-stars_95/]	2.25	
Passed *(show details)*	test_product_page.py::TestUserAddToBasketFromProductPage::test_user_cant_see_success_message[http://selenium1py.pythonanywhere.com/catalogue/the-city-and-the-stars_95/]	11.55	
Passed *(show details)*	test_product_page.py::test_guest_cant_see_product_in_basket_opened_from_product_page[http://selenium1py.pythonanywhere.com/catalogue/the-city-and-the-stars_95/]	12.97	
XFailed *(show details)*	test_product_page.py::test_guest_cant_see_success_message_after_adding_product_to_basket[http://selenium1py.pythonanywhere.com/catalogue/the-city-and-the-stars_95/]	4.31	
XFailed *(show details)*	test_product_page.py::test_message_disappeared_after_adding_product_to_basket[http://selenium1py.pythonanywhere.com/catalogue/the-city-and-the-stars_95/]	6.64	
Failed *(hide details)*	test_main_page.py::test_guest_can_go_to_login_link	2.12	

```
browser = <selenium.webdriver.chrome.webdriver.WebDriver (session="0fe65db76c90d19280b703fe8fceb428")>

    def test_guest_can_go_to_login_link(browser):
        link = "http://selenium1py.pythonanywhere.com/catalogue/the-shellcoders-handbook_209?promo=midsummer"
        page = MainPage(browser, link)
        page.open()
        login_page = page.go_to_login_page()
>       login_page.should_be_login_page()
E       AttributeError: 'NoneType' object has no attribute 'should_be_login_page'

test_main_page.py:10: AttributeError
 -----------------------------Captured stdout setup------------------------------

start chrome browser for test..
```

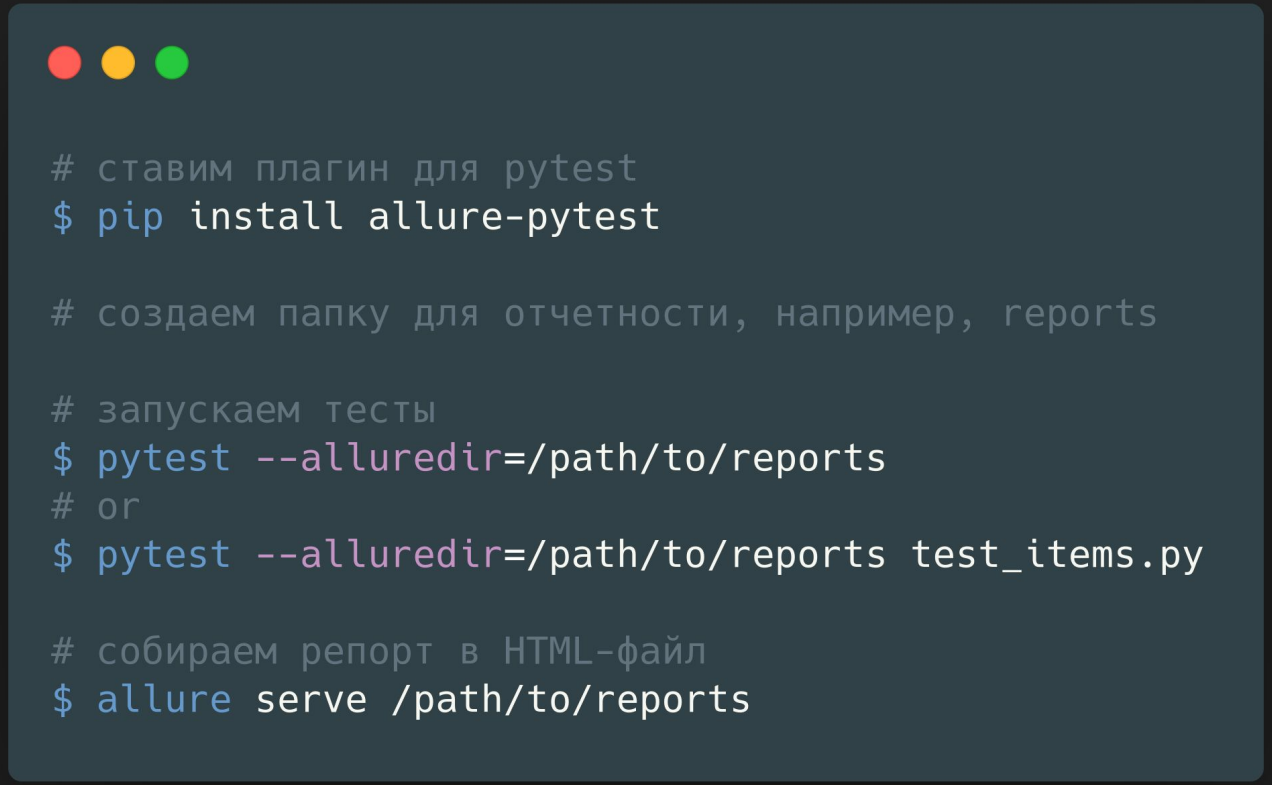
Failed *(hide details)*	test_product_page.py::TestLoginFromProductPage::test_guest_should_see_login_link_on_product_page[http://selenium1py.pythonanywhere.com/catalogue/the-city-and-the-stars_95/]	11.99	

```
self = <test_product_page.TestLoginFromProductPage object at 0x7f4caac5fd68>, browser = <selenium.webdriver.chrome.webdriver.WebDriver (session="513df440197860f2e06651b6c2f5b03a")>
link = 'http://selenium1py.pythonanywhere.com/catalogue/the-city-and-the-stars_95/'

    @pytest.mark.parametrize('link', [product_base_link])
    def test_guest_should_see_login_link_on_product_page(self, browser, link):
        page = ProductPage(browser, link)
```

Средства представления отчетов для PyTest

1. Плагин PyTest-HTML
   https://github.com/pytest-dev/pytest-html

2. Allure Framework
   1. Установить Java
   2. Скачать дистрибутив Allure для командной строки
   3. Добавить JAVA_HOME
   4. Добавить папку с дистрибутивом Allure в PATH


# ставим плагин для pytest
$ pip install allure-pytest

создаем папку для отчетности, например, reports

# запускаем тесты
$ pytest --alluredir=/path/to/reports
# or
$ pytest --alluredir=/path/to/reports test_items.py

# собираем репорт в HTML-файл
$ allure serve /path/to/reports

Allure

- Overview
- Categories
- Suites
- Graphs
- Timeline
- Behaviors
- Packages

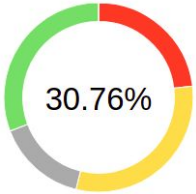
En

Collapse

ALLURE REPORT 12/2/2019

20:10:36 - 15:18:33 (28d 19h)

13
test cases



TREND

There is nothing to show

SUITES 3 items total

test_product_page	2	2	3	2
test_make_report	1	1		
test_main_page		1	1	

Show all

ENVIRONMENT

There are no environment variables

CATEGORIES 2 items total

Product defects	3
Test defects	4

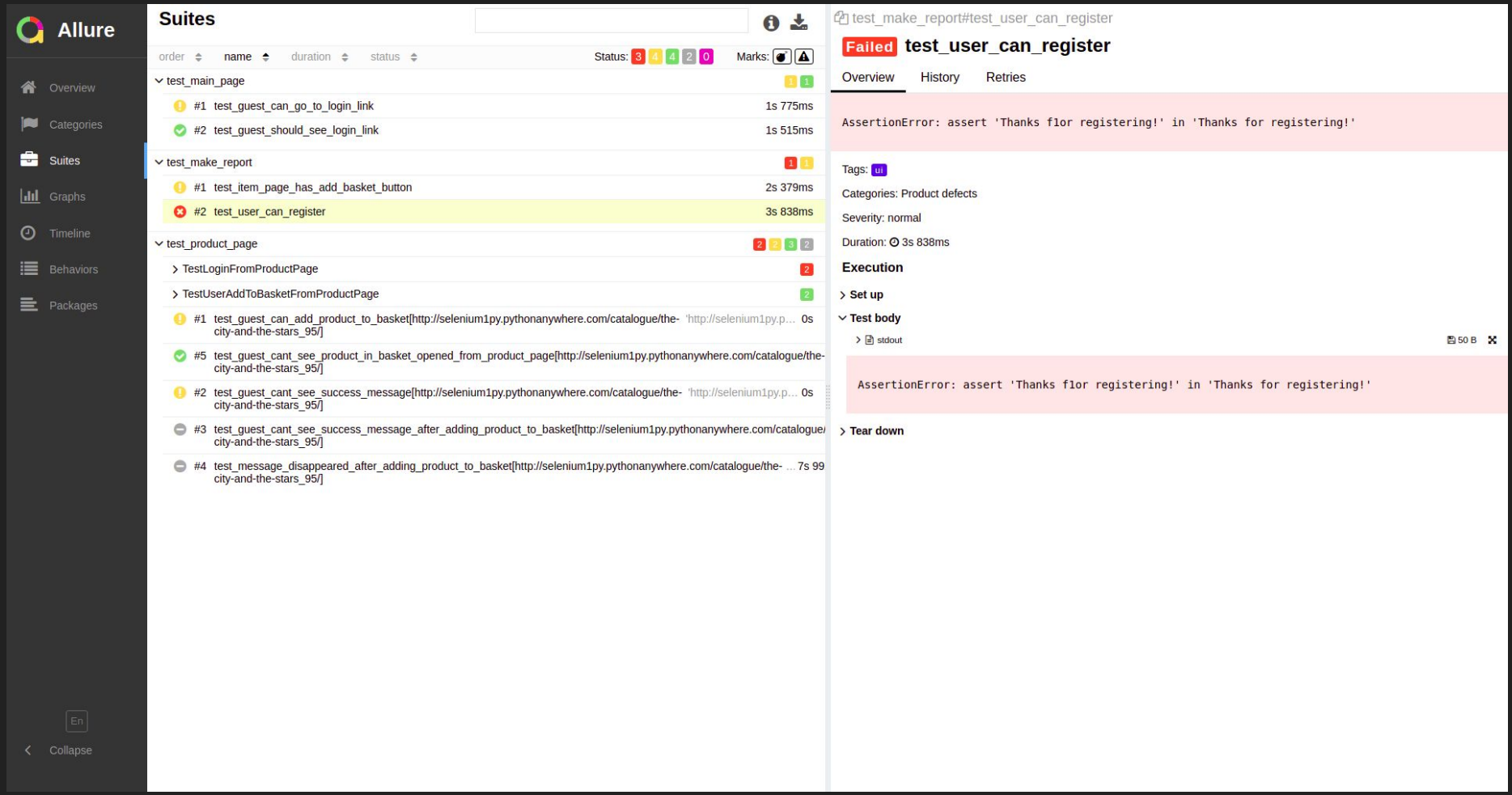
Show all

FEATURES BY STORIES 13 items total

Show all

EXECUTORS

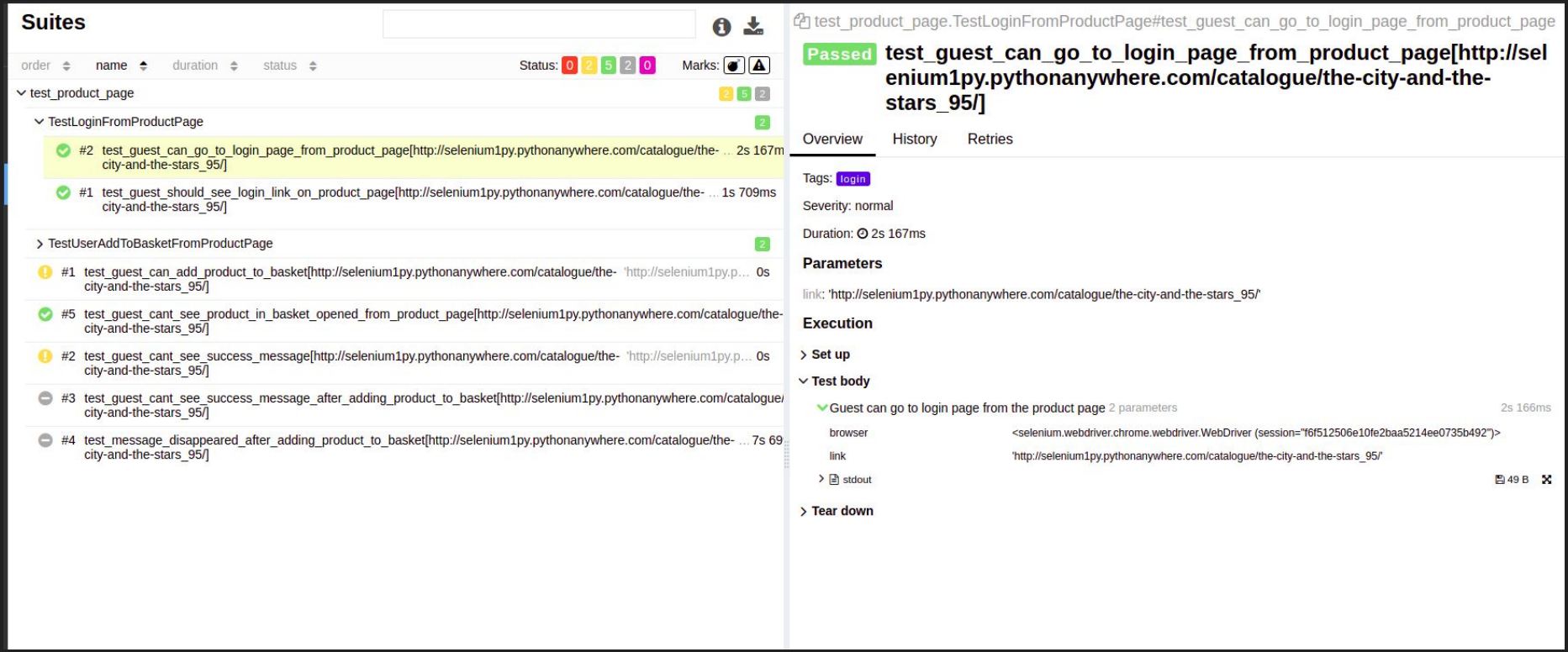
There is no information about tests executors


Allure
Overview
Categories
Suites
Graphs
Timeline
Behaviors
Packages
En
Collapse
Suites
order
name
duration
status
Status: 3 4 4 2 0
Marks:
test_main_page 1 1
#1 test_guest_can_go_to_login_link 1s 775ms
#2 test_guest_should_see_login_link 1s 515ms
test_make_report 1 1
#1 test_item_page_has_add_basket_button 2s 379ms
#2 test_user_can_register 3s 838ms
test_product_page 2 2 3 2
TestLoginFromProductPage 2
TestUserAddToBasketFromProductPage 2
#1 test_guest_can_add_product_to_basket[http://selenium1py.pythonanywhere.com/catalogue/the- 'http://selenium1py.p... 0s
city-and-the-stars_95/]
#5 test_guest_cant_see_product_in_basket_opened_from_product_page[http://selenium1py.pythonanywhere.com/catalogue/the-
city-and-the-stars_95/]
#2 test_guest_cant_see_success_message[http://selenium1py.pythonanywhere.com/catalogue/the- 'http://selenium1py.p... 0s
city-and-the-stars_95/]
#3 test_guest_cant_see_success_message_after_adding_product_to_basket[http://selenium1py.pythonanywhere.com/catalogue/
city-and-the-stars_95/]
#4 test_message_disappeared_after_adding_product_to_basket[http://selenium1py.pythonanywhere.com/catalogue/the- ... 7s 99
city-and-the-stars_95/]
test_make_report#test_user_can_register
Failed test_user_can_register
Overview
History
Retries
AssertionError: assert 'Thanks f1or registering!' in 'Thanks for registering!'
Tags: ui
Categories: Product defects
Severity: normal
Duration: 3s 838ms
Execution
Set up
Test body
stdout
50 B
AssertionError: assert 'Thanks f1or registering!' in 'Thanks for registering!'
Tear down

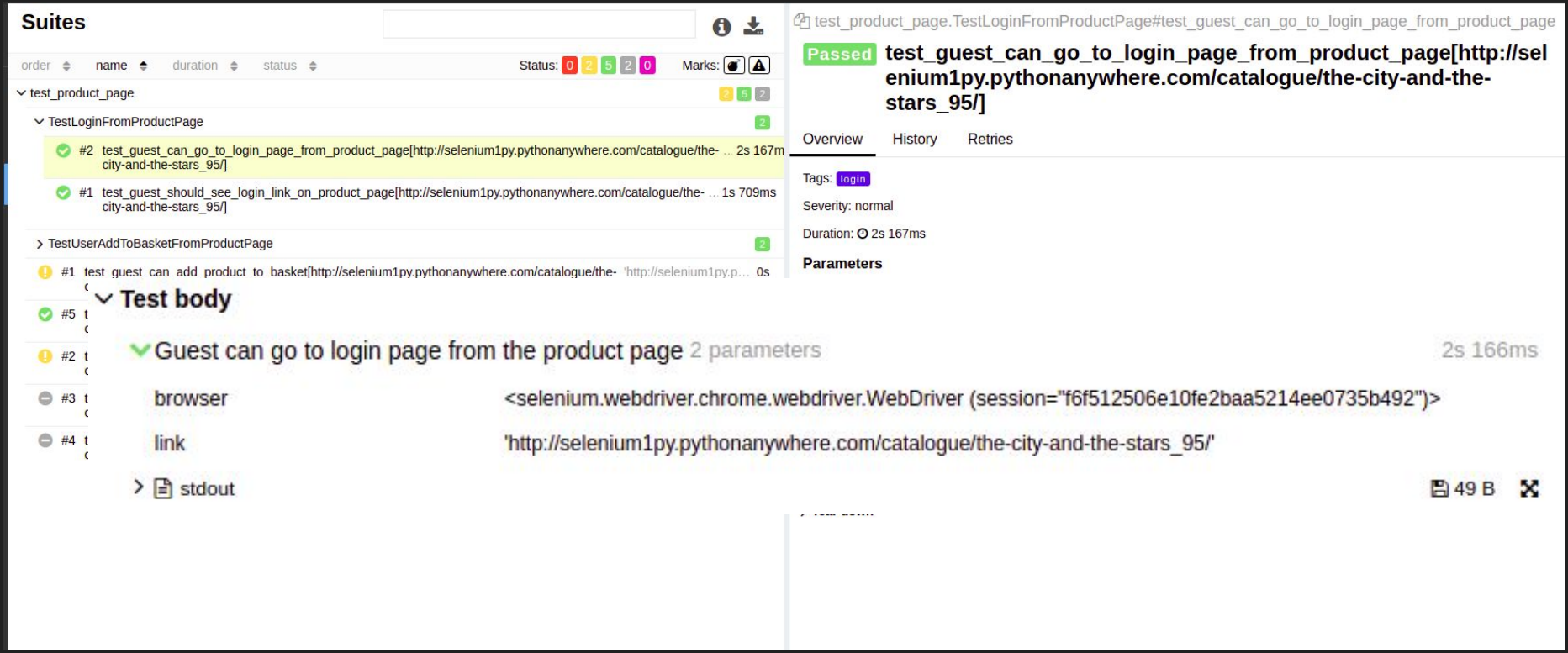
Allure: шаги (steps)

```python
import allure

@allure.step
# test_user_can_add_product_to_basket

@allure.step('User can add a product to the basket')
# User can add a product to the basket
```

Allure: шаги (steps)



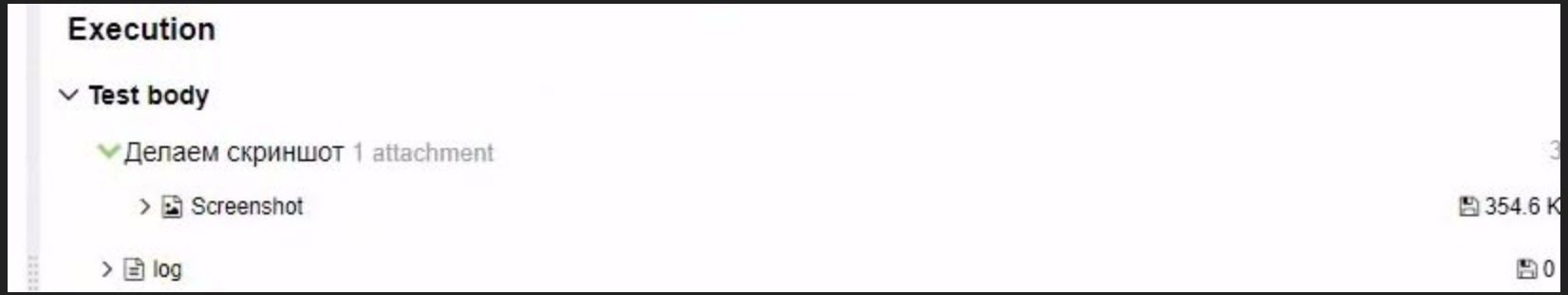
Allure: шаги (steps)



Allure: attachments

```
with allure.step('Делаем скриншот'):
    allure.attach(browser.get_screenshot_as_png(), name='Screenshot', attachment_type=AttachmentType.PNG)
```

Allure: attachments



Визуальное тестирование

Визуальное тестирование

- С помощью расширений для браузера:

Визуальное тестирование

- С помощью расширений для браузера:
  - Page Ruler

Jason Thye

Python Visual Regression Testing

Posted April 16, 2016

Needle, Selenium and Nose in Python

Introduction

In this post, we will have a look at using Needle which allows you to automatically check that your visuals render correctly by taking screenshots of portions of a website and comparing them against known good screenshots. We can then use PerceptualDiff as an image comparison utility to show the difference between the screenshots. We will write a simple Selenium test using Needle and PerceptualDiff to automate the checking of the header bar for the Sydney Morning Herald home page.

Installation

Python

Install Python 2.7.10. Please ensure that you allow the installer to update your PATH. As part of your installation, please also ensure that you install pip, which is a tool that allows easy management of any Python packages that you wish to use. Installers for versions prior to Python 2.7.9 will not have pip bundled, so if you do choose to use an earlier version, please ensure you manually install pip.

Ensure that you have successfully installed Python:

```
bash-3.2$ python --version
Python 2.7.10
```

Ensure that you have successfully installed pip:

```
bash-3.2$ pip --version
pip 6.1.1 from /Library/Frameworks/Python.framework/Versions/2.7/lib/python2.7/site-packages (python 2.7)
```

You can now use the following commands to install the Selenium, Needle and Nose packages:

Визуальное тестирование

- С помощью расширений для браузера:
  - Page Ruler
  - PerfectPixel

Python Visual Regression Testing

Jason Thye

Posted April 18, 2016

Needle, Selenium and Nose in Python

Introduction

In this post, we will have a look at using Needle which allows you to automatically check that your visuals render correctly by taking screenshots of portions of a website and comparing them against known good screenshots. We can then use PerceptualDiff as an image comparison utility to show the difference between the screenshots. We will write a simple Selenium test using Needle and PerceptualDiff to automate the checking of the header bar for the Sydney Morning Herald home page.

Installation

Install Python 2.7.10. Please ensure that you allow the installer to update your PATH. As part of your installation, please also ensure that you install pip, which is a tool that allows easy management of any Python packages that you wish to use. Installers for versions prior to Python 2.7.9 will not have pip bundled, so if you do choose to use an earlier version, please ensure you manually install pip.

Ensure that you have successfully installed Python:

```
bash-3.2$ python --version
Python 2.7.10
```

Ensure that you have successfully installed pip:

```
bash-3.2$ pip --version
pip 6.1.1 from /Library/Frameworks/Python.framework/Versions/2.7/lib/python2.7/site-packages (python 2.7)
```

You can now use the following commands to install the Selenium, Needle and Nose packages:

Визуальное тестирование

- С помощью расширений для браузера:
  - Page Ruler
  - PerfectPixel
  - Full Page Screen Capture

Python Visual Regression Testing

Posted April 16, 2016

Needle, Selenium and Nose in Python

Introduction

In this post, we will have a look at using Needle which allows you to automatically check that your visuals render correctly by taking screenshots of portions of a website and comparing them against known good screenshots. We can then use PerceptualDiff as an image comparison utility to show the difference between the screenshots. We will write a simple Selenium test using Needle and PerceptualDiff to automate the checking of the header bar for the Sydney Morning Herald home page.

Installation

Python

Install Python 2.7.10. Please ensure that you allow the installer to update your PATH. As part of your installation, please also ensure that you install pip, which is a tool that allows easy management of any Python packages that you wish to use. Installers for versions prior to Python 2.7.9 will not have pip bundled, so if you do choose to use an earlier version, please ensure you manually install pip.

Ensure that you have successfully installed Python:

```
bash-3.2$ python --version
Python 2.7.10
```

Ensure that you have successfully installed pip:

```
bash-3.2$ pip --version
pip 6.1.1 from /Library/Frameworks/Python.framework/Versions/2.7/lib/python2.7/site-packages (python 2.7)
```

You can now use the following commands to install the Selenium, Needle and Nose packages:

Визуальное тестирование

- С помощью библиотек:
  - aShot (Java)
    https://github.com/pazone/ashot
  - Needle (Python + Nosetests)
    https://the-creative-tester.github.io/Python-Visual-Regression-Testing/
    https://needle.readthedocs.io/en/latest/

Selenium: делаем скриншот

Selenium: делаем скриншот

```
browser.get_screenshot_as_file(filename)
```

Selenium: делаем скриншот

```
browser.get_screenshot_as_file(filename)
```

```
@pytest.fixture(scope="function")

def browser(request):
    browser_name = request.config.getoption("browser")
    browser = webdriver.Chrome
    yield browser
    print("\nquit browser..")
    browser.save_screenshot('screenshot.png')
    browser.quit()
```

Selenium: делаем скриншот

- **browser.save_screenshot('screenshot.png')**

Другие методы (например, для Remote WebDriver):

- get_screenshot_as_file('screenshot.png')
- get_screenshot_as_png()
- get_screenshot_as_base64() - для embedded images в HTML

```python
import random

link = "http://selenium1py.pythonanywhere.com/"


def test_user_can_register(browser):
    unique_var = random.randint(1, 1000000)

    browser.get(link)
    login_btn = browser.find_element_by_css_selector("#login_link")
    login_btn.click()

    email = browser.find_element_by_css_selector("#id_registration-email")
    email.clear()
    email.send_keys("{}@example.com".format(unique_var))

    password = browser.find_element_by_css_selector("#id_registration-password1")
    confirmation = browser.find_element_by_css_selector("#id_registration-password2")
    password.send_keys(unique_var, unique_var, unique_var)

    confirmation.send_keys(unique_var, unique_var, unique_var)

    registration_btn = browser.find_element_by_css_selector("[name='registration_submit']")
    registration_btn.click()

    login_success_text = browser.find_element_by_css_selector(".alertinner").text

    assert "Thanks for registering!" in login_success_text
```

```python
import pytest
from selenium import webdriver
from datetime import datetime


# Добавляем парсер, который будет считывать опции, которые пользователь передает через терминал при
запуске теста
def pytest_addoption(parser):
    # Браузер: передается через опцию --browser=браузернейм, по умолчанию Хром
    parser.addoption('--browser', action='store', default="firefox",
                     help="Choose browser: chrome or firefox")


@pytest.fixture(scope="function")
def browser(request):
    browser_name = request.config.getoption("browser")
    if browser_name == "chrome":
        print("\nstart chrome browser for test..")
        browser = webdriver.Chrome()
        browser.maximize_window()
    elif browser_name == "firefox":
        print("\nstart firefox browser for test..")
        browser = webdriver.Firefox()
        browser.maximize_window()
    else:
        print("Browser {} still is not implemented".format(browser_name))
    yield browser
    print("\nquit browser..")
    now = datetime.now().strftime('%Y-%m-%d_%H-%M-%S')
    browser.save_screenshot('screenshot-{}.png'.format(now))
    browser.quit()
```

Домашнее задание (финальное) - 2 недели!

Пройти задания модуля 6 (“Пятая неделя: Применение паттерна Page Object Model”);

По примеру теории сгенерировать отчет о прохождении тестов с помощью Allure.

Когда откроется модуль 7 (“Шестая неделя: Финишная прямая”):

1. Оформить тесты в репозитории согласно принципам паттерна Page Object Model;
2. Приготовить код к ревью и отправить на итоговую проверку!

Спасибо за внимание!

Хороших вам выходных :)