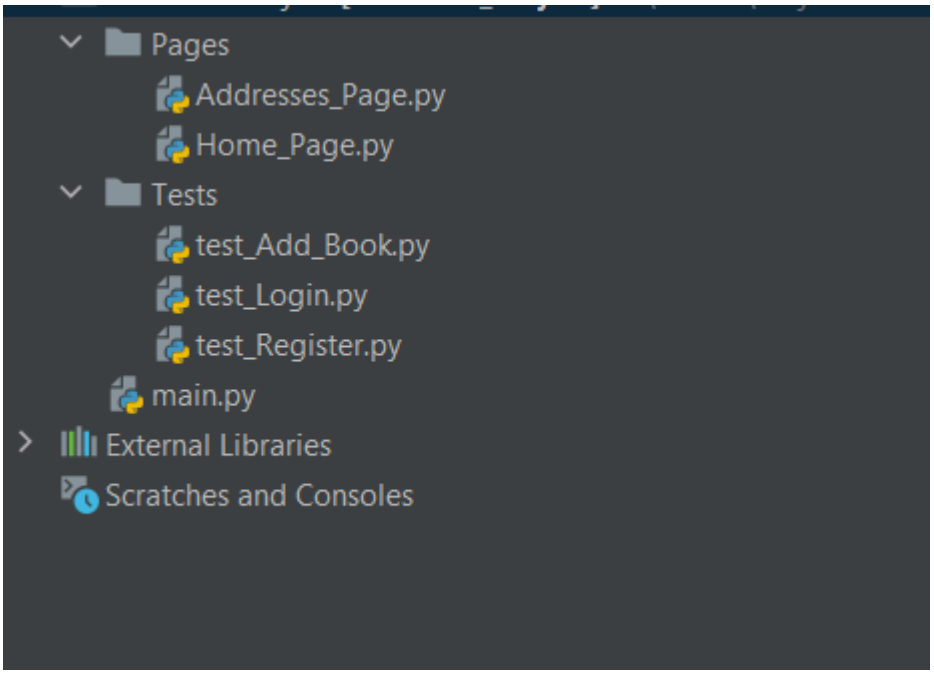
Automatyzacja przypadków testowych przy pomocy Selenium Webdriver

Autor: Adrian Dubel
Chorzów 2020

Projekt został napisany dla strony http://a.testaddressbook.com/ i umieszczony w repozytorium githubie https://github.com/AdrianDubel/Selenium-Project. W projekcie został użyty wzorzec projektowy Page Object Pattern.

Struktura plików:



W folderze “Pages” znajdują się dwa pliki z lokatorami i metodami dla strony Home_Page i Addresses_Page, natomiast w folderze “Tests” znajdują się pliki z testowymi skryptami.

Home_Page:

```
class HomePage:

    def __init__(self, driver):
        self.driver = driver

        self.home_btn_xpath = "//a[@href='/']"
        self.signin_btn_xpath = "//a[@id='sign-in']"
        self.email_input_css = "#session_email"
        self.password_input_css = "#session_password"
        self.submit_xpath = "//input[@name='commit']"
        self.signup_btn_css = "[data-test='sign-up']"
        self.reg_email_input_css = "#user_email"
        self.reg_password_input_css = "#user_password"
        self.title_css = "h1"
        self.alert_css = ".alert"
```

```
    def click_home(self):
        self.driver.find_element_by_xpath(self.home_btn_xpath).click()

    def click_signin(self):
        self.driver.find_element_by_xpath(self.signin_btn_xpath).click()

    def enter_email(self, email):
        self.driver.find_element_by_css_selector(self.email_input_css).send_keys(email)

    def enter_password(self, password):

self.driver.find_element_by_css_selector(self.password_input_css).send_keys(password)

    def click_submit(self):
        self.driver.find_element_by_xpath(self.submit_xpath).click()

    def click_signup(self):
        self.driver.find_element_by_css_selector(self.signup_btn_css).click()

    def enter_new_email(self, email):
        self.driver.find_element_by_css_selector(self.reg_email_input_css).send_keys(email)

    def enter_new_password(self, password):

self.driver.find_element_by_css_selector(self.reg_password_input_css).send_keys(passwor
d)

    def check_header(self):
        title = self.driver.find_element_by_css_selector(self.title_css)
        title_text = title.text
        title_visible = title.is_displayed()
        assert title_text == "Welcome to Address Book"
        assert title_visible == True

    def check_alert(self):
        error = self.driver.find_element_by_css_selector(self.alert_css)
        error_text = error.text
        error_visible = error.is_displayed()
        assert error_text == "Bad email or password."
        assert error_visible == True
```

Addresses_Page:

```
from selenium.webdriver.common.keys import Keys

class AddressPage:

    def __init__(self, driver):
        self.driver = driver

        self.addresses_btn_css = "[data-test='addresses']"
        self.new_address_xpath = "//a[.='New Address']"
        self.firstname_input_css = "#address_first_name"
        self.lastname_input_css = "#address_last_name"
        self.firstaddress_input_css = "#address_street_address"
        self.secondaddress_input_css = "#address_secondary_address"
        self.city_input_css = "#address_city"
        self.state_drop_css = "#address_state"
        self.zip_code_css = "#address_zip_code"
        self.age_input_css = "#address_age"
        self.phone_input_css = "#address_phone"
        self.climbing_check_css = "#address_interest_climb"
        self.submit_address_xpath = "//input[@name='commit']"
        self.alert_css = ".alert"

    def click_address(self):
        self.driver.find_element_by_css_selector(self.addresses_btn_css).click()

    def add_new_address(self):
        self.driver.find_element_by_xpath(self.new_address_xpath).click()

    def enter_first_name(self, name):
        self.driver.find_element_by_css_selector(self.firstname_input_css).send_keys(name)

    def enter_last_name(self, lastname):

self.driver.find_element_by_css_selector(self.lastname_input_css).send_keys(lastname)

    def first_address(self, firstaddress):

self.driver.find_element_by_css_selector(self.firstaddress_input_css).send_keys(firstaddress
)

    def second_address(self, secondaddress):

self.driver.find_element_by_css_selector(self.secondaddress_input_css).send_keys(second
address)
```

```
def enter_city(self, city):
    self.driver.find_element_by_css_selector(self.city_input_css).send_keys(city)

def state(self, state):
    drop = self.driver.find_element_by_css_selector(self.state_drop_css)
    drop.click()
    drop.send_keys(state)
    drop.send_keys(Keys.ENTER)

def enter_zipcode(self, zipcode):
    self.driver.find_element_by_css_selector(self.zip_code_css).send_keys(zipcode)

def enter_age(self, age):
    self.driver.find_element_by_css_selector(self.age_input_css).send_keys(age)

def enter_phone(self, number):
    self.driver.find_element_by_css_selector(self.phone_input_css).send_keys(number)

def climbing(self):
    self.driver.find_element_by_css_selector(self.climbing_check_css).click()

def submit(self):
    self.driver.find_element_by_xpath(self.submit_address_xpath).click()

def verify_alert(self):
    alert = self.driver.find_element_by_css_selector(self.alert_css)
    alert_visible = alert.is_displayed()
    alert_text = alert.text
    assert alert_visible == True
    assert alert_text == "Address was successfully created."
```

I. Przypadki testowe

ID: 001
Tytuł: Próba zalogowania się z poprawnymi danymi
Środowisko: Chrome wersja 90.0.4430.212, Windows 10 Home Edition
Warunki wstępne: .

1. Uruchomiona przeglądarka
2. Na stronie: http://a.testaddressbook.com/
3. Użytkownik niezalogowany

Kroki:

1. Wpisz adres e-mail
2. Wpisz hasło
3. Kliknij przycisk “Sign in”

Oczekiwany rezultat:
Użytkownik zostaje zalogowany. Na stronie pojawia się komunikat “Welcome to Address Book”

ID: 002
Tytuł: Próba zalogowania się z niepoprawnym adresem e-mail
Środowisko: Chrome wersja 90.0.4430.212, Windows 10 Home Edition
Warunki wstępne: .

1. Uruchomiona przeglądarka
2. Na stronie: http://a.testaddressbook.com/
3. Użytkownik niezalogowany

Kroki:

1. Wpisz niepoprawny adres e-mail
2. Wpisz hasło
3. Kliknij przycisk “Sign in”

Oczekiwany rezultat:
Użytkownik nie zostaje zalogowany. Na stronie pojawia się komunikat “Bad email or password.”

ID: 003
Tytuł: Rejestracja nowego użytkownika
Środowisko: Chrome wersja 90.0.4430.212, Windows 10 Home Edition
Warunki wstępne: .

1. Uruchomiona przeglądarka
2. Na stronie: http://a.testaddressbook.com/

Kroki:

1. Kliknij przycisk “Sign up”
2. Wpisz e-mail
3. Wpisz hasło
4. Kliknij przycisk “Sign up”

Oczekiwany rezultat:
Użytkownik zostaje zarejestrowany i zalogowany na stronie. Na stronie pojawia się komunikat “Welcome to Address Book.”

ID: 004
Tytuł: Dodanie nowego adresu
Środowisko: Chrome wersja 90.0.4430.212, Windows 10 Home Edition
Warunki wstępne: .

1. Uruchomiona przeglądarka
2. Na stronie: http://a.testaddressbook.com/
3. Użytkownik zalogowany

Kroki:

1. Kliknij przycisk “Addresses”
2. kliknij przycisk “New Address”
3. Wpisz imię
4. Wpisz nazwisko
5. Wpisz adres
6. Wpisz miasto
7. Wybierz stan
8. Wpisz kod pocztowy
9. Wpisz wiek
10. Wpisz numer telefonu
11. Kliknij przycisk “Create Address”

Oczekiwany rezultat:
Nowy adres zostaje dodany do listy adresów użytkownika.

II. Automatyzacja przypadków testowych przy pomocy Selenium Webdriver

ID: 001, 002:

```
from selenium import webdriver
import unittest
from Pages.Home_Page import HomePage

class LoginTest(unittest.TestCase):

    @classmethod
    def setUpClass(cls):
        cls.driver = webdriver.Chrome()
        cls.driver.implicitly_wait(10)
        cls.driver.maximize_window()

    def test_login_valid(self):
        driver = self.driver

        url = "http://a.testaddressbook.com/sign_in"
        email = "test@wp.pl"
        password = "12345678"

        driver.get(url)
        home_page = HomePage(driver)
        home_page.enter_email(email)
        home_page.enter_password(password)
        home_page.click_submit()
        home_page.check_header()

    def test_login_invalid(self):
        driver = self.driver

        url = "http://a.testaddressbook.com/sign_in"
        email = "test_wrong@wp.pl"
```

```
        password = "12345678"

        driver.get(url)
        home_page = HomePage(driver)
        home_page.enter_email(email)
        home_page.enter_password(password)
        home_page.click_submit()
        home_page.check_alert()

    @classmethod
    def tearDownClass(cls):
        cls.driver.close()
        cls.driver.quit()

if __name__ == '__main__':
    unittest.main()
```

ID: 003

```
from selenium import webdriver
import unittest
from Pages.Home_Page import HomePage
from faker import Faker

class RegisterTest(unittest.TestCase):

    @classmethod
    def setUpClass(cls):
        cls.driver = webdriver.Chrome()
        cls.driver.implicitly_wait(10)
        cls.driver.maximize_window()

    def test_register(self):
        driver = self.driver

        url = "http://a.testaddressbook.com/sign_in"
        faker = Faker()
        email = faker.company_email()
        password = "12345678"

        driver.get(url)
        home_page = HomePage(driver)
        home_page.click_signup()
        home_page.enter_new_email(email)
```

```
        home_page.enter_new_password(password)
        home_page.click_submit()
        home_page.check_header()

    @classmethod
    def tearDownClass(cls):
        cls.driver.close()
        cls.driver.quit()

if __name__ == '__main__':
    unittest.main()
```

ID: 004

```
from selenium import webdriver
import unittest
from Pages.Addresses_Page import AddressPage
from Pages.Home_Page import HomePage
from faker import Faker

class AddAddressTest(unittest.TestCase):

    @classmethod
    def setUpClass(cls):
        cls.driver = webdriver.Chrome()
        cls.driver.implicitly_wait(10)
        cls.driver.maximize_window()

    def test_add_address(self):
        driver = self.driver

        url = "http://a.testaddressbook.com/sign_in"
        email = "test@wp.pl"
        password = "12345678"
        faker = Faker()
        firstname = faker.first_name()
        lastname = faker.last_name()
        address1 = "Street Avenue"
        address2 = "23"
        city = "Los Angeles"
        state = "ke"
        age = "27"
        zip = faker.zipcode()
```

```
        phone = "222333444"

        driver.get(url)
        home_page = HomePage(driver)
        home_page.enter_email(email)
        home_page.enter_password(password)
        home_page.click_submit()

        address_page = AddressPage(driver)
        address_page.click_address()
        address_page.add_new_address()
        address_page.enter_first_name(firstname)
        address_page.enter_last_name(lastname)
        address_page.first_address(address1)
        address_page.second_address(address2)
        address_page.enter_city(city)
        address_page.state(state)
        address_page.enter_zipcode(zip)
        address_page.enter_age(age)
        address_page.enter_phone(phone)
        address_page.submit()
        address_page.verify_alert()

    @classmethod
    def tearDownClass(cls):
        cls.driver.close()
        cls.driver.quit()

if __name__ == '__main__':
    unittest.main()
```

III. Uwagi końcowe

Automatyzacja przypadków testowych powiodła się.