kainos®

METODY PROGRAMOWANIA "PAGE OBJECT PATTERN"

PIOTR DUBIELA

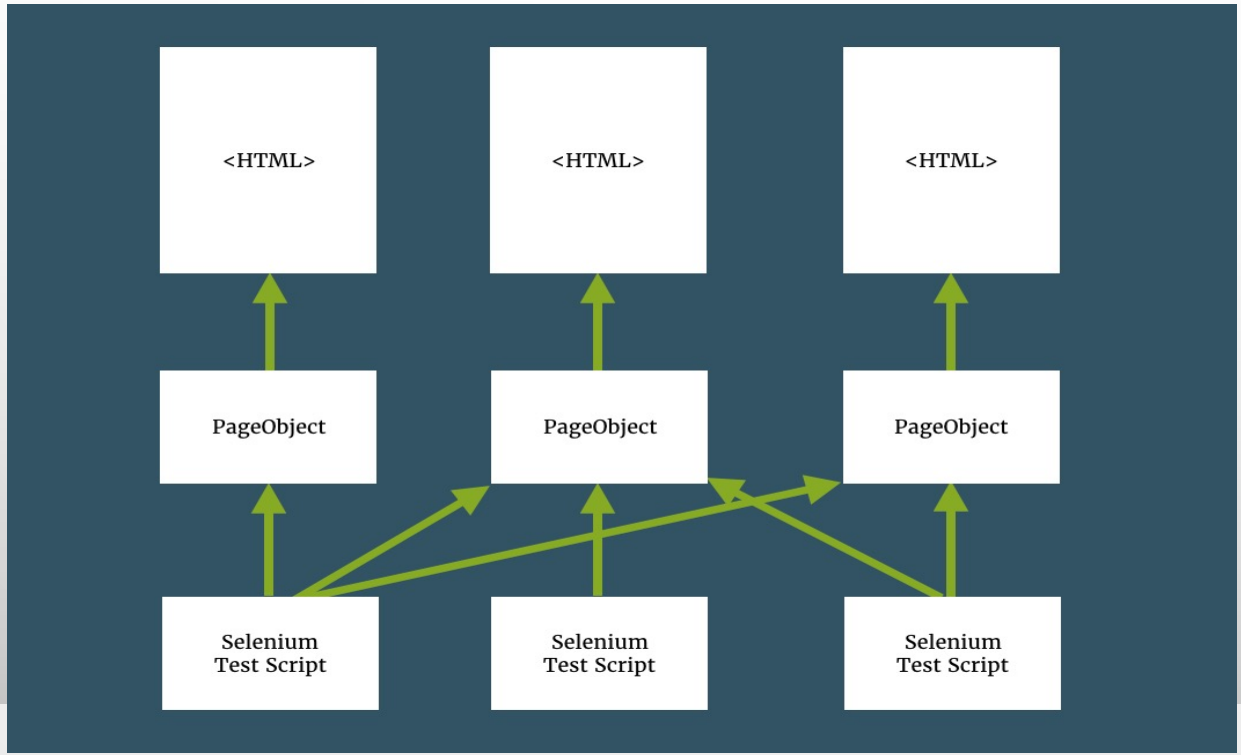
16 XII 2021

PAGE OBJECT PATTERN

- Popularny wzorzec w automatyzacji testów automatycznych end-2-end
- Pozwala na stosunkowo łatwe utrzymywanie testów
- Zapobiega redundancji kodu
- Zmiany na interfejsie graficznym mogą zostać zaktualizowane punktowo w modelu

PAGE OBJECT PATTERN

- Pozwala na oddzielenie od siebie kodu testów I kodu strony



PAGE OBJECT PATTERN

Metody pojedynczego obiektu (klasy) powinny reprezentować funkcje oferowane przez dany widok (niekoniecznie całą stronę – może to być jej wycinek)
Np.

- Koszyk w sklepie będzie umożliwiał usuwanie elementów, sprawdzenie ilości elementów, zwrócenie listy elementów itd.

PAGE OBJECT PATTERN

Różne wyniki działania tej samej akcji powinny być zamodelowane w osobnych metodach np.

- udaneLogowanie() – przeniesie nas do strony głównej
- nieUdaneLogowanie() – pozostawi nas na stronie logowania

PAGE OBJECT PATTERN

Wypełnienie formularza jako kod testowy

```
@Test
public void successfulLogin(){
    driver.findElement(By.id("login")).sendKeys("admin");
    driver.findElement(By.id("passwd")).sendKeys("admin1");
    driver.findElement(By.id("SubmitLogin")).click();
    Assert.assertEquals(driver.findElement(
            By.xpath("//a[@title='View my customer account']"))
            .getText(), "John Smith");
}
```

PAGE OBJECT PATTERN

Wypełnienie formularza z wykorzystaniem Page Object

```
@Test
public void successfullLoginTest(){
    loginPage.successfullLogin(”admin", ”admin1");
    Assert.assertEquals(topNavigation.getUserName(), ”Jan Kowalski");
}
```

PAGE OBJECT PATTERN

Wypełnienie formularza z wykorzystaniem Page Object

```
public MyAccountPage successfullLogin(String email, String password){
    emailInput.sendKeys(email);
    passwordInput.sendKeys(password);
    signInButton.click();
    return new MyAccountPage(driver);
}
```

PAGE OBJECT FACTORY PATTERN

Page Object Factory Pattern

- Rozszerzenie wzorca Page Object Pattern
- Wspierane przez WebDriver
- Pozwala na inicjalizowanie elementów strony w momencie ich użycia
- Zapewnia bardziej swobodny dostęp do elementów
- Poprawia czytelność kodu

PAGE OBJECT FACTORY PATTERN

Lokalizacja i przypisanie pojedynczych elementów

```
@FindBy(id = "email")
private WebElement emailInput;

@FindBy(id="passwd")
private WebElement passwordInput;

@FindBy(id = "SubmitLogin")
private WebElement signInButton;
```

PAGE OBJECT FACTORY PATTERN

Lokalizacja i przypisanie wielu elementów

```
@FindBy(xpath = "//ul[contains(@class,
'product_list')]//a[@class='product-name']")
private List<WebElement> productButtons;
```

PAGE OBJECT FACTORY PATTERN

Niezbędny warunek do działania Page Factory – w przeciwnym razie NPE!

```
public TopNavigation(WebDriver driver) {
    this.driver = driver;
    PageFactory.initElements(driver, this);
}
```

SKRYPT AUTOMATYCZNY Z SELENIUM WEBDRIVER I JAVA

DEMO : Tworzenie testów opartych o Page Objecty

Na podstawie sklepu online

Bibliografia:

1. https://flood.io/wp-content/uploads/2015/05/selenium-logo-DB9103D7CF-seeklogo.com_.png (dostęp 01.05.2019)
2. https://addons.mozilla.org/pl/firefox/addon/selenium-ide/ (dostęp 01.05.2019)
3. http://seleniummaster.com/sitecontent/index.php/introduction-to-selenium-automation/selenium-ide/93-selenium-ide-locating-elements (dostęp 01.05.2019)