

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

ХАРКІВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ ІМ. В. Н. КАРАЗІНА

ФАКУЛЬТЕТ КОМП'ЮТЕРНИХ НАУК

КАФЕДРА БЕЗПЕКИ ІНФОРМАЦІЙНИХ СИСТЕМ І ТЕХНОЛОГІЙ

Лабораторна робота №7

з навчальної дисципліни

**«Математичні методи та технології тестування та верифікації
програмного забезпечення»**

Виконала:

студентка групи КС-23

Рузудженк С.Р.

Перевірив:

доцент

Нарсжній О. П.

Лабораторна робота №7

з навчальної дисципліни

«Математичні методи та технології тестування та верифікації програмного забезпечення»

Тема: «Шаблон проектування Page Object»

Мета: написання тестового сценарію з використанням шаблону проектування Page Object.

Хід роботи

Запускаємо *IntelliJIDEA*, створюємо проект *Maven*. Додаємо відповідні залежності у файл *pom.xml*.

```
<dependencies>
  <dependency>
    <groupId>org.seleniumhq.selenium</groupId>
    <artifactId>selenium-java</artifactId>
    <version>3.14.0</version>
  </dependency>
  <dependency>
    <groupId>junit</groupId>
    <artifactId>junit</artifactId>
    <version>RELEASE</version>
  </dependency>
</dependencies>
```

Збираємо проект з тестами: в першому ми заходимо на сторінку <https://pn.com.ua/> та перевіряємо наявність елемента «ІТ услуги» на сторінці; у другому – заходимо на сторінку <https://pn.com.ua/> та обираємо категорію «Диваны», потім першого популярного виробника і перевіряємо, що всі відфільтровані товари належать йому. Для цього нам необхідні наступні класи:

- **HomePage**, який задає необхідні дії в браузері;

```
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.WebElement;
import org.openqa.selenium.support.FindBy;
import org.openqa.selenium.support.PageFactory;
```

Вказуємо поля класу:

```
private WebDriver driver;
@FindBy(
    xpath = ".*//*[@id='column-center']/section/div[1]/a")
private WebElement computerCategory;

@FindBy(
    xpath = ".*//*[@id='column-center']/section/div[5]/ul/li[1]/a")
private WebElement sofasCategory;

@FindBy(
    xpath = ".*//*[@id='producers-filter-block']/div[3]/span[3]/a")
private WebElement popularBrand;

@FindBy(
    xpath = ".*//*[@class = 'td-logo-block']/a/span[1]")
private WebElement home;
```

Оголошення елементів відбувається декларативно за допомогою анотації `@FindBy`. Також пошук елементів можна організувати, використовуючи в методах `driver.findElement`.

Створюємо конструктор:

```
public HomePage(WebDriver driver) {
    this.driver = driver;
    PageFactory.initElements(driver, page: this);
}
```

Створюємо необхідні нам методи:

```
public ComputerPage go_to_homePage(){
    this.home.click();
    return new ComputerPage(this.driver);
}

public ComputerPage chooseComputerCategory(){
    this.computerCategory.click();
    return new ComputerPage(this.driver);
}

public ComputerPage chooseSofasCategory(){
    this.sofasCategory.click();
    return new ComputerPage(this.driver);
}

public ComputerPage choosePopularBrand(){
    this.popularBrand.click();
    return new ComputerPage(this.driver);
}
```

- **ComputerPage**, в якому реалізуються тести;

Поля класу:

```
private WebDriver driver;
@FindBy(
    xpath = ".*//*[@id='column-center']/section/div[4]/article[3]/div/div")
private WebElement ITService;

@FindBy(xpath = ".*//*[@id='column-center']/" +
    "section/div[3]/*li/*div[2]/div[1]/*[contains(text(), 'AMF ')]")
private List<WebElement> list;

private ArrayList<String> stringList;
```

Конструктор:

```
public ComputerPage(WebDriver driver) {
    this.driver = driver;
    PageFactory.initElements(driver, page: this);
}
```

Методи:

```
public String getTextITService() {
    return this.ITService.getText();
}

public ArrayList getElements(){
    stringList = new ArrayList<String>();
    for(int i = 0; i < this.list.size(); ++i) {
        this.stringList.add((this.list.get(i)).getText());
    }
    return stringList;
}

public boolean toCompare(){
    boolean condition = true;
    for(int i = 0; i < this.list.size(); ++i) {
        System.out.println(this.stringList.get(i));
        condition = ((this.stringList.get(i)).equals((this.list.get(i)).getText()));
    }
    return condition;
}
```

- **Test**, який містить власне тести;

Конструктор и поля:

```
String siteHomePage = "https://pn.com.ua/";
private WebDriver driver;
private HomePage homePage;
private ComputerPage computerPage;

public Test() {
}
```

Тести:

```

@Before
public void testBeforeClass() throws Exception {
    System.setProperty("webdriver.chrome.driver", "C:\\Users\\Anna\\Desktop\\it\\chromedriver.exe");
    this.driver = new ChromeDriver();
    this.driver.get(this.siteHomePage);
}

@org.junit.Test
public void testComputerCategory() throws Exception {
    this.homePage = PageFactory.initElements(this.driver, HomePage.class);
    this.computerPage = this.homePage.chooseComputerCategory();
    System.out.println(this.computerPage.getTextITService());
    this.computerPage.getTextITService();
    this.driver.manage().timeouts().implicitlyWait(2L, TimeUnit.MINUTES);
    System.out.println("We're looking for element: "+this.computerPage.getTextITService());
    String string = this.computerPage.getTextITService().substring(0, 21);
    System.out.println("We found element: "+string);
    Assert.assertEquals(this.computerPage.getTextITService(), string);
}

@org.junit.Test
public void testAMFpresence() throws InterruptedException {
    this.homePage = PageFactory.initElements(this.driver, HomePage.class);
    this.computerPage = this.homePage.chooseSofasCategory();
    this.computerPage = this.homePage.choosePopularBrand();
    System.out.println(this.computerPage.getElements());
    this.driver.manage().timeouts().implicitlyWait(1L, TimeUnit.MINUTES);
    System.out.println("\nCOMPARING\n");
    Assert.assertTrue(message: "Something have gone wrong!", this.computerPage.toCompare());
}

@After
public void testAfterClass() {
    this.driver.quit();
}

• Main.

import org.openqa.selenium.WebDriver;
import org.openqa.selenium.chrome.ChromeDriver;
import org.openqa.selenium.support.PageFactory;

public class Main {
    public Main() {
    }

    public static void main(String[] args) throws InterruptedException {
        String siteHomePage = "https://pn.com.ua/";
        System.setProperty("webdriver.chrome.driver",
            "C:\\Users\\Anna\\Desktop\\it\\chromedriver.exe");
        WebDriver driver = new ChromeDriver();
        driver.get(siteHomePage);
        HomePage homePage = (HomePage)PageFactory.initElements(driver, HomePage.class);
        ComputerPage computerPage = homePage.chooseComputerCategory();
        computerPage = homePage.go_to_homePage();
        computerPage = homePage.chooseSofasCategory();
        computerPage = homePage.choosePopularBrand();
        computerPage = homePage.go_to_homePage();
    }
}

```

Обидва тести пройшли успішно.

✓ Test	23s 518ms
✓ testComputerCategory	11s 507ms
✓ testAMFpresence	12s 11ms

Висновки

Отже, Page Object – це шаблон проектування, який широко використовується в автоматизованому тестуванні та дозволяє розділити логіку виконання тестів від їх реалізації. Він моделює сторінки застосунка, що тестується, в якості об'єктів у коді.

У результаті ми маємо окремі класи, що відповідають за роботу з HTML кожної конкретної веб-сторінки. Такий підхід значно зменшує об'єм коду, що повторюється, оскільки одні й ті ж самі об'єкти сторінок можна використовувати в різних тестах.

Таким чином, під час виконання лабораторної роботи, було написано тестовий сценарій за допомогою шаблону проектування *PageObject*, а також вдало протестовано даний сценарій (тести пройшли успішно).