**LAB 7: THỰC HÀNH SELENIUM  
 AUTOMATION VÀ REPORT**

**I/ Yêu cầu thực hành**

1. Học viên đã có kiến thức cơ bản về java và TestNG.
2. Học viên đã có kiến thức cơ bản về selenium webdriver, reportNG và Screen capture.
3. Thực hành viết code automation theo TestCase được cho trước.
4. Thực hành với TestNG và đọc kết quả testcase
5. Lưu lại code.

**II/ Thiết lập project.**

**2.1** Tạo Maven project.

<Tham khảo: REF\_1\_CREATE\_MAVEN\_PROJECT.docx>

**2.2**.Cài đặt TestNG

<Tham khảo: REF\_2\_Setup\_TestNG.docx>

**2.3.** Cài đặt Selenium và Chrome driver.

<Tham khảo: REF\_4\_Setup\_Selenium.docx>

**2.4.** Thêm thư viện tích hợp báo cáo.

**1. Thư viện ReportNG**

<https://mvnrepository.com/artifact/org.uncommons/reportng/1.1.4>

khai báo thư viện trong pom.xml với thông tin sau:

| <dependency>  <groupId>org.uncommons</groupId>  <artifactId>reportng</artifactId>  <version>1.1.4</version>  <scope>test</scope>  </dependency> |
| --- |

**2. Thư viện Guice**

<https://mvnrepository.com/artifact/com.google.inject/guice/4.2.2>

khai báo thư viện trong pom.xml với thông tin sau:

| <dependency>  <groupId>com.google.inject</groupId>  <artifactId>guice</artifactId>  <version>4.2.2</version>  </dependency> |
| --- |

**2.5**. Các hàm thông dụng trong Selenium webdriver.

<Tham khảo: REF5\_basic \_func\_webdriver.docx>

**2.6.** Đọc data test từ properties files

<Tham khảo: REF6\_ Properties files.docx>

**III. Thực hành**

**1.** Yêu cầu viết chương trình tự động hóa quá trình sau.

* Gọi trình duyệt Google Chrome.
* Mở URL: [**https://automationexercise.com/products**](https://automationexercise.com/products)
* Đảm bảo người dùng chưa đăng nhập tài khoản
* Thực hiện đăng nhập với thông tin tài khoản (học viên tự tạo tài khoản), đảm bảo đăng nhập thành công với tài khoản trên.
* Thực hiện đăng xuất tài khoản trên và đăng nhập lại với tài khoản mới (học viên tự tạo tài khoản), đảm bảo đăng nhập thành công với tài khoản mới.
* Vân dụng POM (Page Object Model) để thực hành login
* Thực hiện capture ảnh trong trường hợp testcase fail, tạo TestNG report.
* Yêu cầu thời gian chờ tải mỗi trang không quá 30s.

***Gợi ý:***

1. Tạo Maven project có tên **LAB\_7\_SELENIUM**, cài đặt thư viện testNG và selenium, reportNG, GUICE
2. Tạo một folder trong project có lên là **configuration**  
   Tạo một properties file có tên là : **configs.properties**  
   Trong file khao báo thông tin url , emai, pass test.

| base\_url=… email\_1=.. password\_1=… email\_2=… password\_2=... |
| --- |

1. Trong package : src/main/java

Tạo class **PropertiesFileUtils**  
Trong này sẽ khai báo đường dẫn đến file **data.properties**  
 - Phương thức static getProperty(String key) : Lấy giá trị theo key  
 Tham khảo tài liệu: *REF6\_ Properties files.docx*

1. Trong package: src/main/java

Tạo một class **CaptureScreenshot**

Tạo trước một folder có tên .**/ScreenShots** trong project.

Viết phương thức:

**static takeScreenshot(WebDriver driver, String testcaseName)** để chụp ảnh màn hình

# Template

| **public static void** takeScreenshot(WebDriver driver, String testcaseName) {  **try** {  *//Tạo tên ảnh trùng với tên testcase, kiểu ảnh là png* String imageName = ………//code here………..;**+ “.png”**;   *// Thực hiện chụp ảnh màn hình, lấy ra đối tượng file ảnh source.* TakesScreenshot screenshot = (TakesScreenshot) driver;  File source = ………//code here………..;   *//Tạo đối tượng file với tên đã đặt bên trên tại thư mục /ScreenShots,*  *// Sau đó thực viện cope file ảnh nguồn vào file đích đó.* File destiny = **new** File(**"./Screenshots/"** + imageName)  FileHandler.*copy*(source, destiny);   } **catch** (Exception ex) {  System.***out***.println(**"Đã xảy ra lỗi khi chụp màn hình!"**);  ex.printStackTrace();  } } |
| --- |

1. Tiến hành truy cập trang để xác định các vị trí web element

|  | **1.iconSignIn = ?** |
| --- | --- |
|  | **1. inputEmail = ?**  **2. inputPass = ?**  **3. btnSignIn = ?** |
|  | **1. iconSignOut = ?** |

1. Vận dụng POM -Page Object Model để viết code cho thao tác login.  
   Mục đích tránh việc phải lặp lại code, mỗi lần login lại phải viết code đi tìm webElement. Giờ đây, chúng ta chỉ cần viết 1 lần và sử dụng nhiều nơi.

- Trong package : src/main/java  
Viết class **LoginPage** - Class này sẽ làm nhiệm vụ tìm các WebElement tương ứng với: ô văn bản nhập email (**inputEmail**), password (**inputPass**) và button SignIn (**btnSignin**)

Class sẽ bao gồm 3 phương thức chính sau:  
 - **void enterEmail(String email)** : nhập email vào ô văn bản inputEmail  
 - **void enterPassword(String password)** : nhập mật khẩu vào ô văn bản **inputPass  
 - void clickSignIn()** : Thực hiện click chuột vào button SignIn

#Template

| **import** org.openqa.selenium.By; **import** org.openqa.selenium.WebElement; **import** org.openqa.selenium.chrome.ChromeDriver; **import** org.openqa.selenium.support.FindBy; **import** org.openqa.selenium.support.ui.ExpectedConditions; **import** org.openqa.selenium.support.ui.WebDriverWait;  **public class** LoginPage {  public LoginPage(WebDriver driver) {  // TODO Auto-generated constructor stub  wait = new WebDriverWait(driver, 30); // cài đặt thời gian tải trang không quá 30s  }   **public** **void** enterEmail(String email) **throws** InterruptedException {  WebElement inputEmail = wait.until(ExpectedConditions.visibilityOfElementLocated(By……));  *…..//input code here…..*  Thread.*sleep*(2000);  }  **public** **void** enterPassword(String password) **throws** InterruptedException {  *…..//input code here…..*  Thread.*sleep*(2000);  }  **public** **void** clickSignIn() **throws** InterruptedException {  *…..//input code here…..*  Thread.*sleep*(2000);  }  } |
| --- |

1. Trong package **src/test/java** . Thêm class **Lab7DemoTest**  
     
   Trong class này chúng ta sẽ định nghĩa 5 method:

* **Phương thức khởi tạo init()**  
   @BeforeClass

**public** **void** init() {  *…..//input code here….*}  
 *Nhiệm vụ:*   
 - Khởi tạo ChromeDriver.  
 - Set cửa sổ trình duyệt max size :   
 - Cài đặt tải các trang không vượt quá 30s.  
 **driver**.manage().timeouts().pageLoadTimeout(30, TimeUnit.***SECONDS***);

* **Phương thức đăng nhập với tài khoản thứ nhất.**  
   @Test(priority = 0)

**public** **void** TC01\_LoginFirstAccount() **throws** InterruptedException {..}

*Nhiệm vụ:*   
 - Lấy ra thông tin baseUrl, Email1, password1 từ file property.  
 - Thực hiện driver.get() để tải trang theo baseUrl  
 - Tìm kiếm và click chuột vào “Sign In”  
 - Khởi tạo LoginPage từ PageFactory và nhập vào thông tin email, password sau đó click vào button SignIn để đăng nhập với thông tin email1, password 1.

- Đảm đăng nhập thành công bằng cách kiểm tra thành phần “Sign out” có được hiển thị  
 - Đăng xuất bằng click vào SignOut

#Template

| @Test(priority = 0)  **public** **void** TC01\_LoginFirstAccount() {  String URL = PropertiesFileUtils*……//code here….*;  String email1 = PropertiesFileUtils*……//code here….*;  String password1 = PropertiesFileUtils*……//code here….*  //Tải trang  *……//code here….*  WebElement iconSignIn =………*……//code here….*  iconSignIn.click();  LoginPage loginPage = new LoginPage(driver);  PageFactory.initElements(driver, loginPage);  *……//code here….   WebElement iconSignOut = ……//code here….*  Assert.*assertEquals*(*……//code here….*)); //kiểm tra signout    iconSignout.click();  Thread.*sleep*(2000);  } |
| --- |

* **Phương thức đăng nhập với tài khoản thứ hai.**  
   @Test(priority = 1)

**public** **void** TC02\_LoginSecondAccount() **throws** InterruptedException {..}  
  
*Nhiệm vụ:*

- Lấy ra thông tin baseUrl, Email2, password2 từ file property.  
- Thực hiện driver.get() để tải trang theo baseUrl  
- Tìm kiếm và click chuột vào “Sign In”  
- Khởi tạo LoginPage từ PageFactory và nhập vào thông tin email, password sau đó click vào button SignIn để đăng nhập với thông tin email2, password 2.

- Đảm quá trình đăng nhập thành công bằng cách kiểm tra thành phần “Sign out” có được hiển thị  
- Đăng xuất bằng click vào SignOut

#Template

| @Test(priority = 1)  **public** **void** TC02\_LoginSecondAccount() {  String URL = PropertiesFileUtils*……//code here….*;  String email2 = PropertiesFileUtils*……//code here….*;  String password2 = PropertiesFileUtils*……//code here….*  //Tải trang  *……//code here…*    WebElement iconSignIn =………*……//code here….*  iconSignIn.click();  LoginPage loginPage = new LoginPage(driver);  PageFactory.initElements(driver, loginPage);  *……//code here….*  *WebElement iconSignOut = ……//code here….*  Assert.*assertEquals*(*……//code here….,*)); //kiểm tra signout    iconSignout.click();  *Thread.sleep(2000);*  } |
| --- |

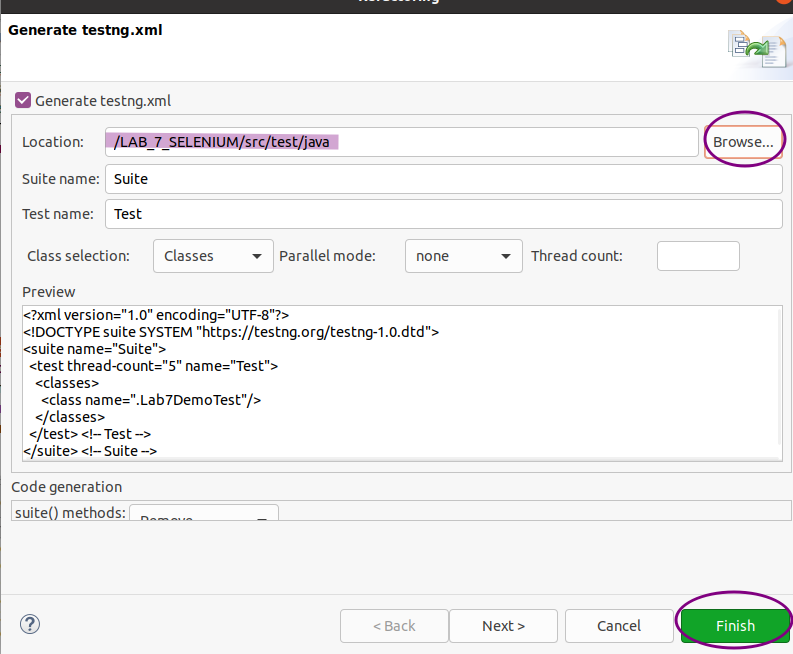
* Phương thức kiểm tra kết quả sau mỗi cho mỗi phương thức **@Test.**Và thực hiện chụp màn hình nếu kết quả Testcase failed

| // Nó sẽ thực thi sau mỗi lần thực thi testcase (@Test)  @AfterMethod  **public** **void** takeScreenshot(ITestResult result) **throws** InterruptedException {  Thread.*sleep*(1000);  // Nếu kết quả failed thì chụp lại màn hình  **if** (ITestResult.***FAILURE*** == result.getStatus()) {  **try** {  CaptureScreenshot.*takeScreenshot*(driver, result.getName());  System.***out***.println("Đã chụp màn hình: " + result.getName());  } **catch** (Exception e) {  System.***out***.println("Exception while taking screenshot " +  e.getMessage());  }  }  } |
| --- |

Tham số đầu vào của hàm là ItestResult chứa kết quả của @Test và tên của testcase: **result.getStatus() & result.getName()**

* Phương thức @AfterClass  
  - Đóng trình duyệt.

1. Tiến hành tạo test suite cho TestNG và tích hợp ReportNG.

- Tạo file test suite : testng.xml  
 Click chuột phải vào project → TestNG → Convert to TestNG.  
   
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
 => chọn đường dẫn đến package test/java của dự án → finish -> một file testng.xml sẽ được tạo ra.  


- Tích hợp ReportNG bằng cách thêm 2 listener vào trong testng.xml

| <**listeners**>  <**listener** class-name="org.uncommons.reportng.HTMLReporter"/>  <**listener** class-name="org.uncommons.reportng.JUnitXMLReporter"/>  </**listeners**> |
| --- |

#File đầy đủ.

| <?xml version=*"1.0"* encoding=*"UTF-8"*?>  <!DOCTYPE suite SYSTEM "https://testng.org/testng-1.0.dtd">  <suite name=*"Suite"*>  <listeners>  <listener class-name=*"org.uncommons.reportng.HTMLReporter"* />  <listener class-name=*"org.uncommons.reportng.JUnitXMLReporter"* />  </listeners>  <test thread-count=*"5"* name=*"Test"*>  <classes>  <class name=*"src/test/java/Lab7DemoTest.java"*/> <!-- class test -->  </classes>  </test> <!-- Test -->  </suite> <!-- Suite --> |
| --- |

9) Chạy testcase.  
 Click chuột phải vào file testng.xml → Run As → TestNg Suite

Sau khi testcase chạy xong, các bạn sẽ cần refresh lại project :  
Click chuột phải vào project → refresh

