

# PAGE OBJECT MODEL



- WHAT IS  
**BUILDER PATTERN**  
IN POM ?



@rajatt95

SUBSCRIBE



# 1. PROJECT SETUP

```
<!-- https://mvnrepository.com/artifact/org.seleniumhq.selenium/selenium-java -->
<dependency>
  <groupId>org.seleniumhq.selenium</groupId>
  <artifactId>selenium-java</artifactId>
  <!-- <version>4.10.0</version> -->
  <version>${selenium.java.version}</version>
</dependency>

<!-- https://mvnrepository.com/artifact/io.github.bonigarcia/webdrivermanager -->
<dependency>
  <groupId>io.github.bonigarcia</groupId>
  <artifactId>webdrivermanager</artifactId>
  <version>${webdrivermanager.version}</version>
</dependency>

<!-- https://mvnrepository.com/artifact/org.testng/testng -->
<dependency>
  <groupId>org.testng</groupId>
  <artifactId>testng</artifactId>
  <version>${testng.version}</version>
  <scope>test</scope>
</dependency>
```



@rajatt95



## 2. Declare Web Elements

```
/******  
// Data Members | Properties  
// Web Elements Declaration  
/******  
  
// Without Page Factories  
1 usage  
private By textBox_Username = By.xpath(xpathExpression: "//*[@id=\"user-name\"]");  
1 usage  
private By textBox_Password = By.xpath(xpathExpression: "//input[@placeholder='Password']");  
1 usage  
private By button_Login = By.id("login-button");
```



**@rajatt95**



# 3. Operations on Web Elements

```
/**
 * Methods | Tasks | Functionality | Behavior
 * Operations on the Web Elements
 */

/**
 * Fills the username field with the provided username.
 *
 * @param username The username to be filled in the username field.
 */
1 usage  👤 "Rajat"
public LoginPage fill_TextBox_Username(String username){
    driver.findElement(textBox_Username).sendKeys(username);
    // This is the example of BUILDER PATTERN
    // After this operation, User remains on the same page
    return this;
}

/**
 * Clicks the login button.
 */
1 usage  new *
public void click_Button_Login() { driver.findElement(button_Login).click(); }
```



**@rajatt95**



# 4. Create Test and call Objects

```
@Test
public void test_Login_Functionality_Valid_Credentials(){

    getDriver().get("https://www.saucedemo.com/");

    LoginPage loginPage = new LoginPage(getDriver());

    // This is the example of BUILDER PATTERN
    // After this operation, User remains on the same page
    LoginPage
        .fill_TextBox_Username("standard_user")
        .fill_TextBox_Password("secret_sauce")
        .click_Button_Login();

    HomePage homePage = new HomePage(getDriver());

    // Assert.assertEquals() -
    // Boolean values as Actual and Expected
    // String values as Actual and Expected
    // This is the example of POLYMORPHISM (OOPS Concept) - Compile Time (Method Overloading)
    // It allows us to perform a single action in different ways.
    Assert.assertEquals(homePage.get_Element_Icon_Twitter().isDisplayed(), expected: true,
        message: "Assertion for Twitter icon presence in the Footer");

    String msg_FooterText_Actual = homePage.get_Element_Msg_Footer().getText();
    String msg_FooterText_Expected = "© 2023 Sauce Labs. All Rights Reserved. Terms of Service | Privacy Policy";
    Assert.assertEquals(msg_FooterText_Actual, msg_FooterText_Expected,
        message: "Assertion for message present in the Footer");
}
```



**@rajatt95**





# RAJAT VERMA

SUBSCRIBE

SOFTWARE TEST ENGINEER



Gmail: [rajatvermaa95@gmail.com](mailto:rajatvermaa95@gmail.com)



Github Profile: <https://github.com/rajatt95>



Github Page: <https://rajatt95.github.io>



LinkedIn: <https://www.linkedin.com/in/rajatt95>



Topmate: <https://topmate.io/rajatt95>



Telegram: <https://t.me/rajatt95>



Instagram: <https://www.instagram.com/rajattvermaa95>



Youtube: <https://www.youtube.com/@rajatt95>