**Setting Up and Running Your Selenium Test Scripts**

**Prerequisites**

1. Java Development Kit (JDK)\*\*: Ensure you have JDK 1.8 or higher installed on your system.
2. Apache Maven\*\*: Maven should be installed for dependency management and build automation.
3. Integrated Development Environment (IDE)\*\*: Use an IDE like Eclipse or IntelliJ IDEA for writing and managing your code.
4. Browser Driver\*\*: ChromeDriver, as this project uses the Chrome browser for testing.

**Project Structure**

1. BasePage.java\*\*: Manages WebDriver setup and teardown.
2. HomePage.java\*\*: Contains methods to interact with the Home Page elements.
3. HomeScript.java\*\*: Contains the test cases.
4. pom.xml\*\*: Maven configuration file for dependencies and plugins.

**Setting Up the Project**

1. Install JDK

1. Download and install JDK from the [Oracle website](https://www.oracle.com/java/technologies/javase-downloads.html).
2. Set up the `JAVA\_HOME` environment variable.

2. Install Maven

1. Download and install Maven from the [Maven website](https://maven.apache.org/download.cgi).
2. Set up the `MAVEN\_HOME` environment variable and add it to the `PATH`.

3. Set Up the Project in an IDE

1. Open your IDE and create a new Maven project.
2. Add the provided `pom.xml` file to the project root.

4. Configure WebDriver

1. Download the ChromeDriver from the [ChromeDriver site](https://sites.google.com/a/chromium.org/chromedriver/downloads).
2. Ensure the ChromeDriver executable is in your system's PATH or provide its path in the code.

**Project Files**

**BasePage.java**

This class is responsible for the setup and teardown of the WebDriver. It ensures that each test has a fresh browser instance and that the browser is closed after the test completes.

**Methods:**

* **initializeWebDriver()**:
  + Annotated with @BeforeMethod to run before each test method.
  + Initializes the Chrome WebDriver.
  + Maximizes the browser window.
  + Sets an implicit wait of 10 seconds.
  + Navigates to the specified URL ("<https://www.fitpeo.com/>").
* **tearDown()**:
  + Annotated with @AfterMethod to run after each test method.
  + Quits the WebDriver, closing the browser.

### ****HomePage.java****

This class contains the methods to interact with the elements on the home page, specifically related to the revenue calculator.

#### Fields:

* **WebDriver driver**: WebDriver instance.
* **WebElements**: Various web elements found using @FindBy annotations, such as revenueCalculator, revenuePageHeading, slider, etc.

#### Methods:

* **HomePage(WebDriver driver)**: Constructor that initializes the web elements using PageFactory.initElements.
* **navigateToRevenueCalculator()**:
  + Clicks on the revenue calculator link.
  + Waits for 1 second to ensure the page has loaded.
* **validateRevenueCalculatorPage()**:
  + Returns the text of the revenue page heading.
* **setSlider()**:
  + Uses Actions to move and adjust the slider element.
* **getSliderValueFromTextBox()**:
  + Returns the current value of the slider from the text box.
* **clearTextBoxForSlider()**:
  + Clears the value in the slider's text box.
* **setSliderFromTextBox(String value)**:
  + Sets a specified value in the slider's text box.
* **isSliderUpdated()**:
  + Returns the current value of the slider as a string.
* **isCPT\_99091Selected(), isCPT\_99453Selected(), isCPT\_99454Selected(), isCPT\_99474Selected()**:
  + Each returns a boolean indicating whether the respective checkbox is selected.
* **selectCPT\_99091(), selectCPT\_99453(), selectCPT\_99454(), selectCPT\_99474()**:
  + Each method clicks the respective checkbox.
* **getTotalRecurringReimbursement()**:
  + Returns the text of the total recurring reimbursement element.

**HomeScript.java**

This class contains the test methods. Each test method uses the methods in HomePage to interact with the webpage and perform assertions to validate functionality.

**Methods:**

* **revenueCalculatorPage()**:
  + Navigates to the revenue calculator page.
  + Validates the page heading contains the expected text.
* **claculateTotalRecurringReimbursement()**:
  + Navigates to the revenue calculator page.
  + Clears and sets the slider value.
  + Selects various CPT codes.
  + Validates the total recurring reimbursement.
* **giveSliderInput()**:
  + Navigates to the revenue calculator page.
  + Clears and sets the slider value.
  + Validates the slider value.
* **dragSlider()**:
  + Navigates to the revenue calculator page.
  + Adjusts the slider using drag and drop.
  + Validates the slider value.

#### pom.xml

Maven configuration file to manage dependencies and plugins.

```xml

<project xmlns="http://maven.apache.org/POM/4.0.0" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 https://maven.apache.org/xsd/maven-4.0.0.xsd">

<modelVersion>4.0.0</modelVersion>

<groupId>fitpeo</groupId>

<artifactId

```xml

<version>0.0.1-SNAPSHOT</version>

<properties>

<maven.compiler.source>1.8</maven.compiler.source>

<maven.compiler.target>1.8</maven.compiler.target>

</properties>

<dependencies>

<dependency>

<groupId>org.seleniumhq.selenium</groupId>

<artifactId>selenium-java</artifactId>

<version>4.21.0</version>

</dependency>

<dependency>

<groupId>org.testng</groupId>

<artifactId>testng</artifactId>

<version>7.10.0</version>

<scope>test</scope>

</dependency>

<dependency>

<groupId>io.github.bonigarcia</groupId>

<artifactId>webdrivermanager</artifactId>

<version>5.3.2</version>

</dependency>

<dependency>

<groupId>org.apache.poi</groupId>

<artifactId>poi</artifactId>

<version>5.3.0</version>

</dependency>

<dependency>

<groupId>commons-io</groupId>

<artifactId>commons-io</artifactId>

<version>2.16.1</version>

</dependency>

</dependencies>

<build>

<plugins>

<!-- Maven Compiler Plugin -->

<plugin>

<groupId>org.apache.maven.plugins</groupId>

<artifactId>maven-compiler-plugin</artifactId>

<version>3.8.1</version>

<configuration>

<source>1.8</source>

<target>1.8</target>

</configuration>

</plugin>

<!-- Maven Surefire Plugin for running TestNG tests -->

<plugin>

<groupId>org.apache.maven.plugins</groupId>

<artifactId>maven-surefire-plugin</artifactId>

<version>3.0.0-M7</version>

<configuration>

<suiteXmlFiles>

<suiteXmlFile>testng.xml</suiteXmlFile>

</suiteXmlFiles>

</configuration>

</plugin>

</plugins>

</build>

</project>

```

### How to Set Up and Run the Scripts

#### 1. \*\*Clone or Download the Project\*\*

- Clone the repository from your version control system or download the project files.

#### 2. \*\*Import the Project into Your IDE\*\*

- Open your IDE (Eclipse/IntelliJ IDEA).

- Import the Maven project using the 'Import Project' option.

#### 3. \*\*Configure Maven\*\*

- Ensure that Maven is installed and configured correctly.

- Run `mvn clean install` in the terminal to download the necessary dependencies.

#### 4. \*\*Configure the Browser Driver\*\*

- Download ChromeDriver from [here](https://sites.google.com/a/chromium.org/chromedriver/downloads).

- Ensure ChromeDriver is in your system's PATH or provide its path in `BasePage.java` if necessary.

#### 5. \*\*Run the Test Cases\*\*

- Right-click on `HomeScript.java` in your IDE and select 'Run As > TestNG Test'.

- Alternatively, you can run the tests using Maven by executing the following command in the terminal:

```bash

mvn test

```

#### 6. \*\*Test Execution and Results\*\*

- The test cases will open the browser, perform the actions, and validate the results.

- The results of the tests will be displayed in the TestNG results tab of your IDE or in the terminal.

### Explanation of Each File and Method

#### \*\*BasePage.java\*\*

- \*\*initializeWebDriver()\*\*: Sets up the ChromeDriver, maximizes the window, and navigates to the FitPeo website. It also applies an implicit wait.

- \*\*tearDown()\*\*: Quits the driver after each test method.

#### \*\*HomePage.java\*\*

- \*\*navigateToRevenueCalculator()\*\*: Clicks the revenue calculator link and waits for the page to load.

- \*\*validateRevenueCalculatorPage()\*\*: Returns the heading text of the revenue calculator page.

- \*\*setSlider()\*\*: Drags the slider to a specified position.

- \*\*getSliderValueFromTextBox()\*\*: Gets the current value of the slider from the input box.

- \*\*clearTextBoxForSlider()\*\*: Clears the text box used for slider input.

- \*\*setSliderFromTextBox(String value)\*\*: Sets the slider value by typing in the text box.

- \*\*isSliderUpdated()\*\*: Returns the current value of the slider.

- \*\*isCPT\_99091Selected(), isCPT\_99453Selected(), isCPT\_99454Selected(), isCPT\_99474Selected()\*\*: Checks if the respective CPT code checkbox is selected.

- \*\*selectCPT\_99091(), selectCPT\_99453(), selectCPT\_99454(), selectCPT\_99474()\*\*: Selects the respective CPT code checkbox.

- \*\*getTotalRecurringReimbursement()\*\*: Returns the total recurring reimbursement text.

#### \*\*HomeScript.java\*\*

- \*\*revenueCalculatorPage()\*\*: Validates the presence of the revenue calculator page.

- \*\*claculateTotalRecurringReimbursement()\*\*: Calculates the total recurring reimbursement by selecting various CPT codes and validating the result.

- \*\*giveSliderInput()\*\*: Sets the slider value via the text box and verifies the update.

- \*\*dragSlider()\*\*: Drags the slider to a specific value and verifies the result.

### Notes

- Ensure you have the required permissions to download and execute WebDriver binaries.

- Adjust sleep and wait times according to the performance of your machine and the website's response time. Use explicit waits wherever possible for better reliability.

- Customize the `pom.xml` file as needed for additional dependencies or plugins.