# **FitPeo Assignment:**

# **Selenium Automation Test in Java**

## **Overview**

This document describes the steps of a Selenium test written in Java that automates the following tasks on the FitPeo website:

1. Navigate to the homepage and then to the Revenue Calculator page.
2. Scroll down to the slider section and adjust the slider.
3. Update the text field and validate the corresponding slider value.
4. Select multiple CPT codes using checkboxes.
5. Validate the total recurring reimbursement value.
6. Verify that the header displaying total recurring reimbursement shows the expected value.

The test uses TestNG for managing test execution and assertions.

## **Prerequisites**

* **Java Development Kit (JDK)** installed on the system.
* **Selenium WebDriver** library included in the project.
* **TestNG** framework for running tests and performing assertions.
* **ChromeDriver** installed and properly configured.

## **Code Walkthrough**

### **Step 1: Import Dependencies**

import java.util.concurrent.TimeUnit;

import org.openqa.selenium.By;

import org.openqa.selenium.JavascriptExecutor;

import org.openqa.selenium.Keys;

import org.openqa.selenium.WebDriver;

import org.openqa.selenium.WebElement;

import org.openqa.selenium.chrome.ChromeDriver;

import org.openqa.selenium.interactions.Actions;

import org.testng.Assert;

import org.testng.annotations.AfterClass;

import org.testng.annotations.BeforeClass;

import org.testng.annotations.Test;

* **Selenium WebDriver** classes such as By, WebDriver, WebElement, and JavascriptExecutor are imported to interact with the browser and perform actions.
* **TestNG** annotations (@BeforeClass, @Test, @AfterClass) are used to manage the test lifecycle and assertions.

### **Step 2: Setup Method**

@BeforeClass

public void setup() {

System.setProperty("webdriver.chrome.driver", "/home/garikipatij/Downloads/chromedriver-linux64 (1)/chromedriver-linux64/chromedriver");

driver = new ChromeDriver();

driver.manage().window().maximize();

}

* **@BeforeClass**: This method is executed before any test methods are run. It sets up the WebDriver and maximizes the browser window.
* **WebDriver Initialization**: The ChromeDriver is configured using the System.setProperty() method, providing the path to the chromedriver executable.

### 

### 

### **Step 3: Test Method**

@Test

public void test() {

// 1. Navigate to the FitPeo Homepage

driver.get("https://www.fitpeo.com/");

driver.manage().timeouts().implicitlyWait(5, TimeUnit.SECONDS);

* **@Test**: The main test method starts here.
* **Navigation**: The browser navigates to the FitPeo homepage using driver.get(). Implicit wait is set to allow elements to load.

### **Step 4: Navigate to the Revenue Calculator Page**

driver.findElement(By.xpath("/html/body/div[1]/div/header/div/div[3]/div[6]/a")).click();

* **Navigate to Revenue Calculator**: A click action is performed on the link to navigate to the Revenue Calculator page using XPath.

### **Step 5: Scroll Down to the Slider Section**

JavascriptExecutor js = (JavascriptExecutor) driver;

WebElement slider = driver.findElement(By.xpath("//span[@class='MuiSlider-root MuiSlider-colorPrimary MuiSlider-sizeMedium css-16i48op']"));

js.executeScript("arguments[0].scrollIntoView();", slider);

* **Scroll to Slider**: JavaScriptExecutor is used to scroll the page to the slider element to ensure it's visible for interaction.

### 

### **Step 6: Adjust the Slider**

### WebElement sliderThumb = driver.findElement(By.cssSelector("span[data-index='0'] input"));

Actions actions = new Actions(driver);

actions.dragAndDropBy(sliderThumb, 94, 0).perform();

* **Adjust the Slider**: Using Actions class, the slider is dragged horizontally by 94 pixels, changing its value.
* We can move using coordinates to match the slider appropriately.
* As I’m not flexible with the extension, I have taken approx. value to the given value i.e., 823.

### **Step 7: Update the Text Field**

j

driver.findElement(By.id(":r0:")).sendKeys(Keys.BACK\_SPACE);

driver.findElement(By.id(":r0:")).sendKeys(Keys.BACK\_SPACE);

driver.findElement(By.id(":r0:")).sendKeys(Keys.BACK\_SPACE);

driver.findElement(By.id(":r0:")).sendKeys("560");

* **Update Text Field**: The code clears the text field and enters a value of 560.

### **Step 8: Validate the Slider Value**

String value = driver.findElement(By.cssSelector("span[data-index='0'] input")).getAttribute("value");

Assert.assertEquals(value, "560");

System.out.println("Success: Attribute value is as expected i.e 560");

* **Validation**: The value of the slider is checked to ensure it matches the expected value (560). The test passes if the values match.

### **Step 9: Update Text Field to Another Value**

driver.findElement(By.id(":r0:")).sendKeys(Keys.BACK\_SPACE);

driver.findElement(By.id(":r0:")).sendKeys(Keys.BACK\_SPACE);

driver.findElement(By.id(":r0:")).sendKeys(Keys.BACK\_SPACE);

driver.findElement(By.id(":r0:")).sendKeys("820");

* **Change Text Field Value**: The text field is cleared and updated with the new value (820).

### **Step 10: Select CPT Codes**

WebElement scrolldown = driver.findElement(By.xpath("//\*[text()='CPT-99457']"));

js.executeScript("arguments[0].scrollIntoView();", scrolldown);

driver.findElement(By.xpath("(//span[contains(@class, 'MuiCheckbox-root')]//input[@type='checkbox'])[1]")).click();

driver.findElement(By.xpath("(//span[contains(@class, 'MuiCheckbox-root')]//input[@type='checkbox'])[2]")).click();

driver.findElement(By.xpath("(//span[contains(@class, 'MuiCheckbox-root')]//input[@type='checkbox'])[3]")).click();

driver.findElement(By.xpath("(//span[contains(@class, 'MuiCheckbox-root')]//input[@type='checkbox'])[8]")).click();

* **Select Checkboxes**: Various checkboxes corresponding to CPT codes are selected by clicking on them.

### 

### **Step 11: Validate Total Recurring Reimbursement**

String actual = driver.findElement(By.xpath("(//p[contains(@class, 'MuiTypo')]//p[@class='MuiTypography-root MuiTypography-body1 inter css-1bl0tdj'])[4]")).getText();

String expected = "$110700";

Assert.assertEquals(actual, expected);

System.out.println("Validation is success.");

* **Validate Reimbursement**: The text value for the total recurring reimbursement is extracted and compared with the expected value ($110700).

### **Step 12: Verify the Header**

String te = driver.findElement(By.xpath("(//p[contains(@class, 'MuiTypo')])[7]")).getText();

System.out.println(te);

* **Verify Header**: The header displaying the total recurring reimbursement is printed to the console for verification.

### **Step 13: Teardown**

@AfterClass

public void teardown() {

driver.close();

}

* **@AfterClass**: This method is executed after the test is completed. It closes the browser window to clean up the WebDriver session.