**import** **static** io.restassured.RestAssured.*given*;

**import** **static** org.hamcrest.Matchers.*equalTo*;

**import** org.testng.annotations.Test;

**import** Payloads.PlacePayload;

**import** reusuableMethods.ReuseMethods;

**import** io.restassured.RestAssured;

**import** io.restassured.path.json.JsonPath;

**public** **class** TestClass {

**public** **static** **void** main(String[] args) {

*addPlace*();

*updatePlace*();

*getPlace*();

}

@Test(priority = 1)

**public** **static** **void** addPlace() {

//Add Place

ReuseMethods.*getURI*();

String response=

*given*().log().all().

queryParam("key", "qaclick123").

header("Content-Type","application/json").

body(PlacePayload.*AddPlace*()).

when().post("maps/api/place/add/json").

then().assertThat().log().all().statusCode(200).

body("scope", *equalTo*("APP")).

header("server", "Apache/2.4.18 (Ubuntu)").

extract().response().asString();

ReuseMethods.*jsonPathConverter*(response).getString("place\_id");

String placeId=ReuseMethods.*jsonPathConverter*(response).getString("place\_id");

String ActualAddress=ReuseMethods.*jsonPathConverter*(response).getString("address");

System.***out***.println("Place id is"+ placeId);

System.***out***.println("Add call is success");

}

//Update the place

@Test(priority = 2)

**public** **static** String updatePlace()

{

ReuseMethods.*getURI*();

String response = *given*().log().all().

queryParam("key", "qaclick123").

header("Content-Type","application/json").

body(PlacePayload.*updatePlace*()).

when().post("maps/api/place/add/json").

then().assertThat().log().all().statusCode(200).

body("scope", *equalTo*("APP")).

header("server", "Apache/2.4.18 (Ubuntu)").

extract().response().asString();

String placeId=ReuseMethods.*jsonPathConverter*(response).getString("place\_id");

String updatedAddress=ReuseMethods.*jsonPathConverter*(response).getString("address");

System.***out***.println("Place id is"+ placeId);

System.***out***.println("update Call is success");

**return** placeId;

}

//get Place

@Test(priority = 3)

**public** **static** **void** getPlace()

{

ReuseMethods.*getURI*();

String response = *given*().log().all().

queryParam("key", "qaclick123").

queryParam("place\_id",*updatePlace*()).

when().get("maps/api/place/get/json").

then().assertThat().log().all().statusCode(200).extract().response().asString();

String updatedAddress=ReuseMethods.*jsonPathConverter*(response).getString("address");

System.***out***.println("Get Call is success");

}

}

**PAYLOADCLASS**

**package** Payloads;

**public** **class** PlacePayload {

**public** **static** String AddPlace()

{

**return** "{\r\n" +

" \"location\": {\r\n" +

" \"lat\": -38.383494,\r\n" +

" \"lng\": 33.427362\r\n" +

" },\r\n" +

" \"accuracy\": 50,\r\n" +

" \"name\": \"Rahul Shetty Academy\",\r\n" +

" \"phone\_number\": \"(+91) 983 893 3937\",\r\n" +

" \"address\": \"29, side layout, cohen 09\",\r\n" +

" \"types\": [\r\n" +

" \"shoe park\",\r\n" +

" \"shop\"\r\n" +

" ],\r\n" +

" \"website\": \"http://rahulshettyacademy.com\",\r\n" +

" \"language\": \"French-IN\"\r\n" +

"}\r\n" +

"";

}

**public** **static** String updatePlace()

{

**return**

"{\r\n" +

" \"location\": {\r\n" +

" \"lat\": -38.383494,\r\n" +

" \"lng\": 33.427362\r\n" +

" },\r\n" +

" \"accuracy\": 50,\r\n" +

" \"name\": \"Rahul Shetty Academy\",\r\n" +

" \"phone\_number\": \"(+91) 983 893 3937\",\r\n" +

" \"address\": \"VRR -806, Summer Walk, Africa\",\r\n" +

" \"types\": [\r\n" +

" \"shoe park\",\r\n" +

" \"shop\"\r\n" +

" ],\r\n" +

" \"website\": \"http://rahulshettyacademy.com\",\r\n" +

" \"language\": \"French-IN\"\r\n" +

"}\r\n" +

"";

}

}

**REUSUABLE METHODCLASS**

package reusuableMethods;

import io.restassured.RestAssured;

import io.restassured.path.json.JsonPath;

public class ReuseMethods {

public static String getURI() {

return RestAssured.baseURI = "https://rahulshettyacademy.com";

}

public static JsonPath jsonPathConverter(String response) {

JsonPath js=new JsonPath(response);

return js;

}

}

Selenium Tests

package com.tests;

import java.awt.AWTException;

import java.awt.Robot;

import java.awt.event.KeyEvent;

import java.io.File;

import java.io.IOException;

import java.util.HashMap;

import java.util.List;

import java.util.Map;

import java.util.UUID;

import java.util.concurrent.TimeUnit;

import org.apache.commons.io.FileUtils;

import org.openqa.selenium.Alert;

import org.openqa.selenium.By;

import org.openqa.selenium.JavascriptExecutor;

import org.openqa.selenium.OutputType;

import org.openqa.selenium.TakesScreenshot;

import org.openqa.selenium.WebDriver;

import org.openqa.selenium.WebElement;

import org.openqa.selenium.chrome.ChromeDriver;

import org.openqa.selenium.chrome.ChromeOptions;

import org.openqa.selenium.htmlunit.HtmlUnitDriver;

import org.openqa.selenium.interactions.Actions;

import org.openqa.selenium.remote.DesiredCapabilities;

import org.testng.Assert;

import org.testng.annotations.Test;

public class SeleniumTests {

// 1.Launch Chrome n other browsers

static WebDriver driver;

public String credentials;

public String credentialsInfo[];

File folder;

public void preRequisites() {

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

driver.manage().deleteAllCookies();

// dynamic wait

driver.manage().timeouts().pageLoadTimeout(40, TimeUnit.SECONDS);

driver.manage().timeouts().implicitlyWait(30, TimeUnit.SECONDS);

}

public void webdriverLaunch()

{

System.setProperty("webdriver.chrome.driver", "C://chromedriver.exe");

ChromeOptions options = new ChromeOptions();

options.addArguments("disable-notifications");

options.addArguments("disable-infobars");

options.addArguments("disable-geolocation");

try {

driver = new ChromeDriver(options);

} catch (Exception e) {

// TODO Auto-generated catch block

e.printStackTrace();

}

System.out.println("Test 1 passed");

}

// 2.locatores in selenium

public void locatorsInSelenium() {

/\*

\* xpath ==> Absolute and relative id, name, linkText,partialLinkText,CSS

\* selector(id,class)

\*/

}

// 3.handling javascript alert

@Test(enabled = false)

public void javascriptAlert() throws InterruptedException {

webdriverLaunch();

try {

driver.get("https://mail.rediff.com/cgi-bin/login.cgi");

} catch (Exception e) {

// TODO Auto-generated catch block

e.printStackTrace();

}

Thread.sleep(5000);

driver.findElement(By.xpath("//\*[@id='login1']")).sendKeys("abc");

driver.findElement(By.xpath("//\*[@title= 'Sign in']")).click();

Alert alert = driver.switchTo().alert();

System.out.println(alert.getText());

alert.accept();

driver.get("https://mail.rediff.com/cgi-bin/login.cgi");

driver.findElement(By.xpath("//\*[@id='login1']")).sendKeys("abc");

driver.findElement(By.xpath("//\*[@title= 'Sign in']")).click();

alert.dismiss();

System.out.println("Test 3 passed");

// file upload

// driver.findElement(By.xpath("//\*[@title= 'Sign

// in']")).sendkeys("/users/dk/extent.html");

}

// 4.handling frames ===> Need to check in Rahul Shetty

// frame id, frame name, frame webelement

@Test(enabled = false)

public void framrHandling() {

webdriverLaunch();

driver.get("https://rahulshettyacademy.com/AutomationPractice/");

preRequisites();

List<WebElement> iframeElements = driver.findElements(By.id("iframe"));

System.out.println("Total number of iframes are " + iframeElements.size());

driver.switchTo().frame("iframe");

driver.findElement((By.xpath("//a[contains(text(),'Courses')]"))).click();

driver.switchTo().defaultContent();

System.out.println("iframe Passed");

}

// 5.mouse movements ==> check the drag and drop part

@Test(enabled = false)

public void mouseMovements() throws InterruptedException, AWTException {

webdriverLaunch();

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

preRequisites();

Thread.sleep(3000);

Robot robot = new Robot();

robot.delay(5000);

robot.keyPress(KeyEvent.VK\_TAB);

robot.keyPress(KeyEvent.VK\_ENTER);

System.out.println("Robot class worked");

Actions action = new Actions(driver);

action.moveToElement(driver.findElement(By.xpath("//\*[contains(text(),'Add-ons')]"))).build().perform();

Thread.sleep(3000);

System.out.println("mouse Movement passed");

driver.findElement(By.xpath("//\*[contains(text(),'SpiceMAX')]")).click();

System.out.println("SpiceMax clicked");

driver.get("https://jqueryui.com/droppable/");

driver.switchTo().frame(0);

/\*

\* action.clickAndHold(driver.findElement(By.

\* xpath("//\*[contains(text(),'Drag me to my target')]")))

\* .moveToElement(driver.findElement(By.xpath("//\*[@id ='droppable']"))

\* .release() .build() .perform();

\*/

}

// 6.dynamic xpath

/\*

\* driver.findElement(By.xpath("//\*[contains(text(),'SpiceMAX')]"));

\* driver.findElement(By.xpath("//\*[starts-with(@id,'SpiceMAX')]"));

\* driver.findElement(By.xpath("//\*[ends-with(@id,'SpiceMAX')]"));

\* driver.get("https://www.ebay.com/"); List<Webelement> linkslist

\* =driver.findElements(By.tagname("a") System.out.println(linkslist.size());

\* for(int i=0; i<linkslist.size();i++) { String linkTest =

\* linkslist.get(i).getText(); System.out.println(linkTest); }

\*/

// 7.waits in selenium

/\*

\* driver.get("url1"); driver.navigate().to("url2"); driver.navigate().back();

\* driver.navigate().forward(); driver.navigate().refresh();

\*/

/\*

\* IMPLICITLY WAIT EXPLICITLY WAIT FLUENT WAIT

\*

\*/

// 8.browser pop ups

/\*

\* Windows Handler

\*/

// 9.headless browser

@Test(enabled = false)

public void testHeadlessBrowser() {

System.setProperty("webdriver.chrome.driver", "C://chromedriver.exe");

ChromeOptions options = new ChromeOptions();

options.addArguments("window-size=1400,800");

options.addArguments("headless");

// options.addArguments("--headless", "--window-size=1920,1200");

driver = new ChromeDriver(options);

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

System.out.println("Headless browser Test Passed");

}

// 9.headless browser 2

@Test(enabled = false)

public void testHeadlessBrowserHTMLUnitDriver() {

// System.setProperty("webdriver.chrome.driver", "C://chromedriver.exe");

// ChromeOptions options = new ChromeOptions();

// options.addArguments("window-size=1400,800");

// options.addArguments("headless");

// options.addArguments("--headless", "--window-size=1920,1200");

driver = new HtmlUnitDriver();

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

System.out.println("HTML Headless browser Test Passed ");

}

// 10.highlight element ==> JS is a class and we have to cast our driver to it.

// executeScript is used to code in JS

public static void highlightElementTest(WebElement element) {

JavascriptExecutor js = ((JavascriptExecutor) driver);

String bgcolor = element.getCssValue("backgroundColor");

for (int i = 0; i < 50; i++) {

changeColor("rgb(0,200,0)", element);

changeColor(bgcolor, element);

}

}

public static void changeColor(String color, WebElement element) {

JavascriptExecutor js = ((JavascriptExecutor) driver);

js.executeScript("arguments[0].style.backgroundColor = '" + color + "'", element);

try {

Thread.sleep(20);

} catch (InterruptedException e) {

}

}

@Test(enabled = false)

public void testHighLightElement() throws InterruptedException, AWTException {

webdriverLaunch();

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

preRequisites();

WebElement addOnButton = driver.findElement(By.xpath("//\*[contains(text(),'Add-ons')]"));

highlightElementTest(addOnButton);

Thread.sleep(3000);

Robot robot = new Robot();

robot.delay(5000);

robot.keyPress(KeyEvent.VK\_TAB);

robot.keyPress(KeyEvent.VK\_ENTER);

System.out.println("HighLight Element Test Passed");

drawBorder(addOnButton);

generateAlerttest("DK test successful");

Alert alert = driver.switchTo().alert();

alert.accept();

System.out.println("Alert Test Passed");

clickElementByJS(addOnButton);

System.out.println("Click Element Test Passed");

refreshBrowserByJS();

System.out.println("Refresh Browser Test Passed");

getTitleByJS();

System.out.println("Get Title Test Passed");

getPageInnerTextByJS();

System.out.println("Get Page Inner Text Passed");

scrollPageDownByJS();

System.out.println("Scroll Page Down Test Passed");

WebElement ele = driver.findElement(By.xpath("//\*[contains(text(),'Terms and Conditions')]"));

scrollIntoViewByJS(ele);

}

// 11.javascript executor

// draw Border

public static void drawBorder(WebElement element) {

JavascriptExecutor js = ((JavascriptExecutor) driver);

js.executeScript("arguments[0].style.border='3px solid red'", element);

}

// generate Alert

public static void generateAlerttest(String message) {

JavascriptExecutor js = ((JavascriptExecutor) driver);

js.executeScript("alert('" + message + "')");

}

// click Element By Javascript Executor

public static void clickElementByJS(WebElement element) {

JavascriptExecutor js = ((JavascriptExecutor) driver);

js.executeScript("arguments[0].click();", element);

}

// refresh the browser by JS

public static void refreshBrowserByJS() {

JavascriptExecutor js = ((JavascriptExecutor) driver);

js.executeScript("history.go(0)");

}

// get Title by JS

public static void getTitleByJS() {

JavascriptExecutor js = ((JavascriptExecutor) driver);

js.executeScript("return document.title;").toString();

}

// getPageInnerText

public static void getPageInnerTextByJS() {

JavascriptExecutor js = ((JavascriptExecutor) driver);

js.executeScript("return document.documentElement.innerText;").toString();

}

// scrollPageDown

public static void scrollPageDownByJS() {

JavascriptExecutor js = ((JavascriptExecutor) driver);

js.executeScript("window.scrollTo(0,document.body.scrollHeight)");

}

// scrollIntoView

public static void scrollIntoViewByJS(WebElement element) {

JavascriptExecutor js = ((JavascriptExecutor) driver);

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

}

// 12.isdisplyed.isenabled.isSelect

@Test(enabled = false)

public void isdisplyed\_isenabled\_isSelectTest() {

webdriverLaunch();

driver.get("https://app.termsfeed.com/profile/login");

preRequisites();

boolean isDisplayedtest = driver.findElement(By.cssSelector("i[class='icon-lock auth-icon']")).isDisplayed();

Assert.assertEquals(true, true, "Display test failed");

boolean isSelectedtest = driver.findElement(By.cssSelector("input[class='custom-control-input']")).isSelected();

Assert.assertEquals(true, true, "Display test failed");

boolean isEnabledtest = driver.findElement(By.xpath("//div[contains(text(),'Log In')]")).isEnabled();

Assert.assertEquals(false, false, "Display test failed");

}

// 13.Dynamic web table ==> Check in Rahul Shetty

// 14.handle calender ==> check in Rahul Shetty

@Test(enabled = false)

public void calendarTest() {

webdriverLaunch();

driver.get("https://jqueryui.com/datepicker/");

preRequisites();

WebElement frameElement = driver.findElement(By.xpath("//\*[@class='demo-frame']"));

// driver.switchTo().frame(0); ==> this is also working fine

driver.switchTo().frame(frameElement);

System.out.println("Inside the frame");

driver.findElement(By.cssSelector("input[id='datepicker']")).click();

}

// \*\*15.handle calendar by javascript executor\*\*

// 16.handle bootstrap dropdown

@Test(enabled = false)

public void bootstrapDropdownTest() {

webdriverLaunch();

driver.get("https://getbootstrap.com/docs/4.0/components/dropdowns/");

preRequisites();

driver.findElement(By.cssSelector("button[id='dropdownMenuButton']")).click();

List<WebElement> dropDownList = driver.findElements(By.xpath("//div[@class='dropdown-menu show']//a"));

System.out.println(dropDownList.size());

for (int i = 0; i < dropDownList.size(); i++) {

dropDownList.get(i).getText();

if (dropDownList.get(i).getText().equalsIgnoreCase("Another action")) {

dropDownList.get(i).click();

break;

}

}

System.out.println("bootstrapDropdownTest passed");

}

// 17.state element exception ==> Need to see the VDO again

// 18. Select Dropdown

// 19.take screenshots and add to extent report

public void getScreenshot(String result) throws IOException {

File src = ((TakesScreenshot) driver).getScreenshotAs(OutputType.FILE);

FileUtils.copyFile(src, new File("C://test//" + result + "screenshot.png"));

}

// 20.locator strategy in webdriver ==> See VDO

// 21.chat bot using selenium ==> check svg elements

@Test(enabled = true)

public void chatBotTest() throws InterruptedException {

webdriverLaunch();

driver.get("https://freecrm.com/");

preRequisites();

Thread.sleep(3000);

driver.findElement(By.xpath("//\*[@class ='intercom-lightweight-app-launcher-icon intercom-lightweight-app-launcher-icon-open']//\*[name()='svg']//\*[local-name()='path'][1]")).click();

}

// 22.use hashmap in selenium

public static HashMap<String, String> getUserLoginInfo() {

HashMap<String, String> userMap = new HashMap<String, String>();

userMap.put("customer", "dk1\_pass1");

userMap.put("admin", "dk2\_pass2");

userMap.put("superAdmin", "dk3\_pass3");

userMap.put("director", "dk4\_pass4");

return userMap;

}

public void loginWithAdminUsertest() {

webdriverLaunch();

driver.get("https://gmail.com/");

preRequisites();

credentials = getUserLoginInfo().get("admin");

credentialsInfo = credentials.split("\_");

driver.findElement(By.name("username")).sendKeys(credentialsInfo[0]);

driver.findElement(By.name("password")).sendKeys(credentialsInfo[1]);

driver.findElement(By.name("submit")).click();

}

public void loginWithCustomerUsertest() {

webdriverLaunch();

driver.get("https://gmail.com/");

preRequisites();

credentials = getUserLoginInfo().get("admin");

credentialsInfo = credentials.split("\_");

driver.findElement(By.name("username")).sendKeys(credentialsInfo[0]);

driver.findElement(By.name("password")).sendKeys(credentialsInfo[1]);

driver.findElement(By.name("submit")).click();

}

// 23.click on back and forward buttons

@Test(enabled = false)

public void backAndforwardtest() {

webdriverLaunch();

driver.get("https://gmail.com/");

System.out.println(driver.getTitle());

preRequisites();

driver.navigate().to("https://freecrm.com/");

System.out.println(driver.getTitle());

driver.navigate().back();

System.out.println(driver.getTitle());

driver.navigate().forward();

System.out.println(driver.getTitle());

driver.navigate().back();

System.out.println(driver.getTitle());

driver.navigate().refresh();

}

// 24.download a file in selenium

public void downloadFileTest() {

// 88908765-87654-987659-876554 is a UUID

folder = new File(UUID.randomUUID().toString());

folder.mkdir(); // ==> this returns an boolean to check the file got created or not

System.setProperty("webdriver.chrome.driver", "C://chromedriver.exe");

ChromeOptions options = new ChromeOptions();

Map<String, Object> prefs = new HashMap<String, Object>();

prefs.put("profile.default\_content\_settings.popups", 0);

prefs.put("download.default\_directory", folder.getAbsolutePath());

options.setExperimentalOption("prefs", prefs);

DesiredCapabilities cap = DesiredCapabilities.chrome();

cap.setCapability(ChromeOptions.CAPABILITY, options);

driver = new ChromeDriver(cap);

}

public void downloadFileTestActual() throws InterruptedException {

driver.get("https//filedoenloadsite.com");

driver.findElement(By.id("abc")).click();

// wait for sometime to download the file

Thread.sleep(2000);

File listOfFiles[] = folder.listFiles();

// The line checks that the directory is not empty

Assert.assertTrue(listOfFiles.length > 0);

// Assert.assertEquals(listOfFiles.length, is(not(0)));

for (File file : listOfFiles) {

// make sure that downloaded file is not empty

// Assert.assertEquals(file.length(), is(not(long)0));

Assert.assertTrue(file.length() > 0);

}

}

// 25.actions class

public void actionsClassTest() {

/\*

\* Mouse Actions in Selenium:

\*

\* doubleClick(): Performs double click on the element

\* clickAndHold(): Performs long click on the mouse without releasing it

\* dragAndDrop(): Drags the element from one point and drops to another

\* moveToElement(): Shifts the mouse pointer to the center of the element

\* contextClick(): Performs right-click on the mouse

\* Keyboard Actions in Selenium:

\* sendKeys(): Sends a series of keys to the element

\* keyUp(): Performs key release

\* keyDown(): Performs keypress without release

\*/

}

// 26.properties file

public void propertitiesFileTest() {

}

//

// 28.use singleton class in selenium

/\*This pattern ensures multiple instances are not created in selenium.Singleton is a creational design pattern,

which ensures that only one object of its kind exists and provides a single point of access to it

for any other code.\*/

public static void initialize(String browserName)

{

browserName = null;

driver = null; // This makes sure multiple instances are not created

if(driver==null) {

if(browserName.equalsIgnoreCase("chrome")) {

System.setProperty("webdriver.chrome.driver", "C://chromedriver.exe");

driver =new ChromeDriver();

}

else if(browserName.equalsIgnoreCase("firefox")) {

System.setProperty("webdriver.chrome.driver", "C://geckodriver.exe");

driver =new ChromeDriver();

}

}

}

public void quit() {

System.out.println("Quitting the browser");

driver.quit();

driver = null; // This makes sure that multiple instances are not created

}

// 32.automate OTP number in selenium

// 34.read and write to and from excel

// 35.Handle SVG elements in selenium

/\* refer chat bot

\* \*/

// 36.multithreading concept in selenium

// 40.handle auth popup in selenium

// 41.broken links ==> Check vdo 29

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*//

// 29.Dynamic xpath coding

// 33.genius in CSS selector

// 39.shadow elements

// 31.jira integration with selenium

// 37.selenium integration with saucelabs

// 38.selenium Grid

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*//

// 42.automate complex graphs

// 43.tricky iframe shadowroot

// 27.write webtables values to excel

// 30.Real time use case of hashmap

}