

testng.xml

<!DOCTYPE suite SYSTEM "https://testng.org/testng-1.0.dtd">

<suite name="Cucumber Test Suite" parallel="false">

<!-- Note: The testng.xml file for the package name, should be

within the package folder, to avoid the red line error.-->

<!-- Replace with your "packageName.mainJavaClassNameWithoutDotJava" for running the

related Given, When, Then methods (via the Selenium Documentation page

object model (non-main) Java classes' Java Objects);

1) The number of (non-main) Java classes for the three groups of websites

is \*three\* in total for the Given, When, and Then methods of the main Java

class TestDesktopWebsites.java respectively; or

2) The number of (non-main) Java classes for the three smartphone

applications is \*three\* in total for Given, When, Then of the main Java

class TestAndroidApps.java, for which the Given and When methods each use one

object of first (non-main) Java class' object respectively, and the Then method

used one object each of the second and third (non-main) Java classes -->

<test name="Cucumber TestDesktopWebsites">

<classes>

<class name="hellocucumber.TestDesktopWebsites"/>

</classes>

</test>

<!-- Additional note: As Selenium\_Android is from another IntelliJ IDEA (Community)

Project, and the related files were copied from Selenium\_Android, the filepath

is not allowed to use Selenium\_Tutorial (this project)'s Selenium\_Screenshot\_Images

folder. This applies to all four Selenium\_Android Java files' bottom of the

"public static String takeScreenshot(WebDriver driver) throws IOException" method(s).

Therefore, the Selenium\_Screenshot\_Images folders for Selenium\_Tutorial and Selenium\_Android

do not contain the other project's screenshot images respectively. -->

<test name="Cucumber TestAndroidApps">

<classes>

<class name="hellocucumber.TestAndroidApps"/>

</classes>

</test>

<!-- Observation:

Both "Spark" ExtentReports reached the target folder of the Selenium\_Tutorial project.

The App ExtentReport was created about 1 minute after the Desktop Website ExtentReport.

As compared to running each project's respective feature files

separately via each project, running both features together in a suite keeps having

errors frequently.

For the case of the first, and third smartphone applications which were used, we need

to be able to click on the permission buttons (which cannot be detected by

the Appium Inspected Weblink) once, in time. Clicking twice means the related test for

the Then method of the main Java class TestAndroidApps.java would fail.

-->

</suite>

Selenium\_Tutorial.feature

Feature: Learning Google Chrome desktop web browser automation with IntelliJ Idea,

Java, Selenium, and Cucumber

Scenario: How to find WebElements which cannot be found with name, id, className,

and partialLinkText

Given that the easiest way to find WebElements is by driver.findElement; and driver.findElements which returns a List WebElement

When it is not possible to do so for certain WebElements in this way

Then we use driver.findElement By.tagName"...", for which ... can be i for <i> or button for <button>

Selenium\_Android.feature

Feature: Learning Android smartphone application automation with IntelliJ Idea,

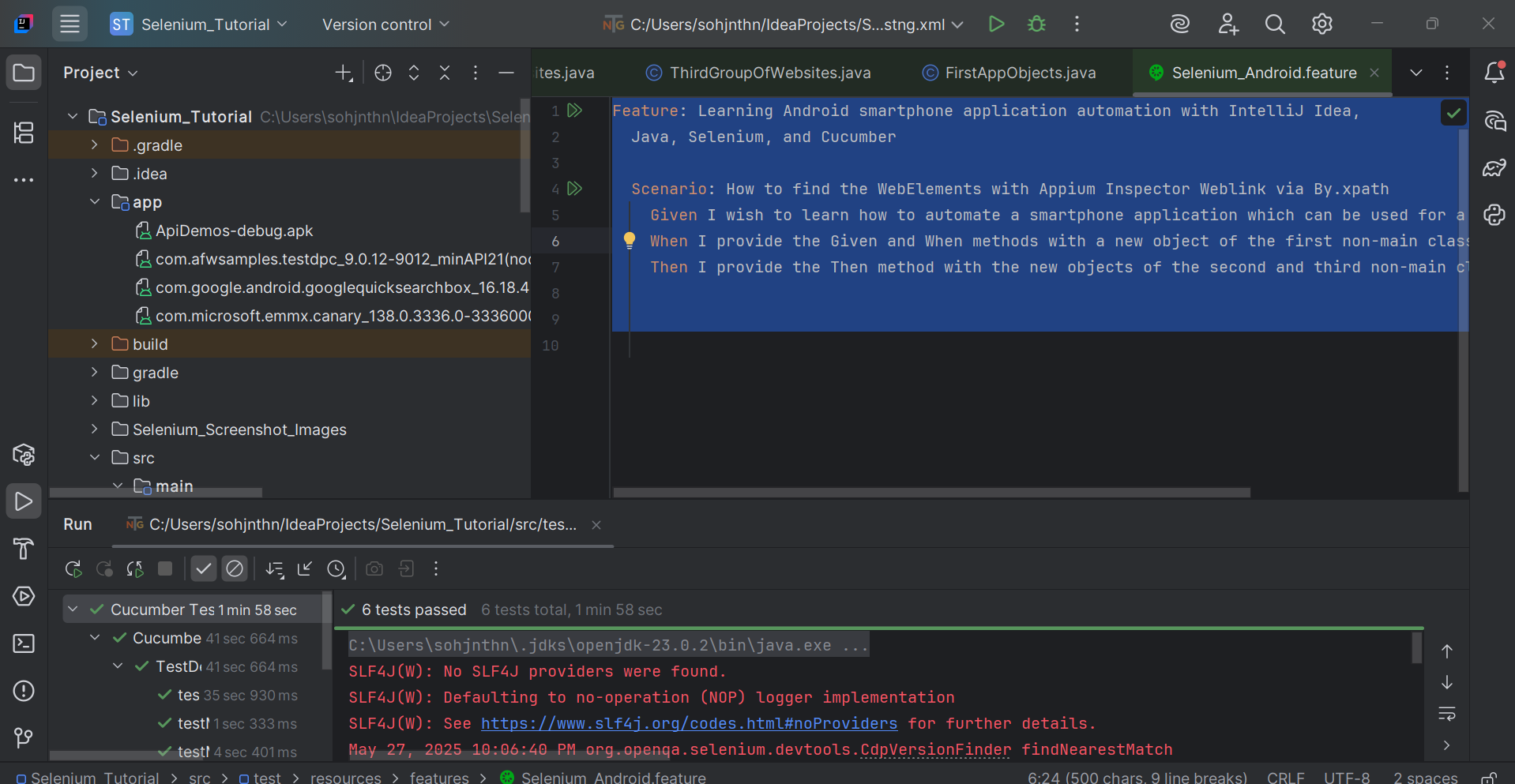
Java, Selenium, and Cucumber

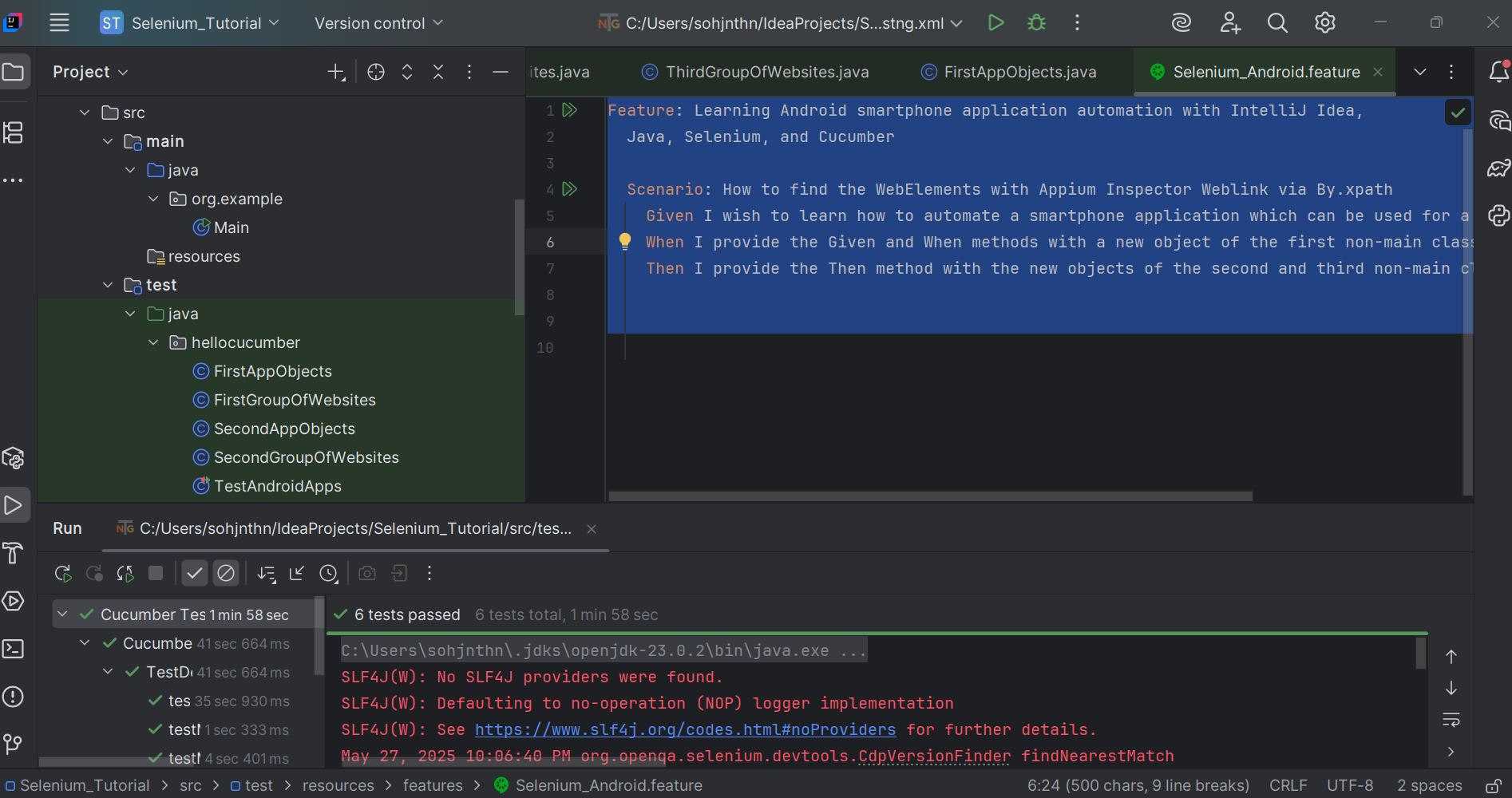
Scenario: How to find the WebElements with Appium Inspector Weblink via By.xpath

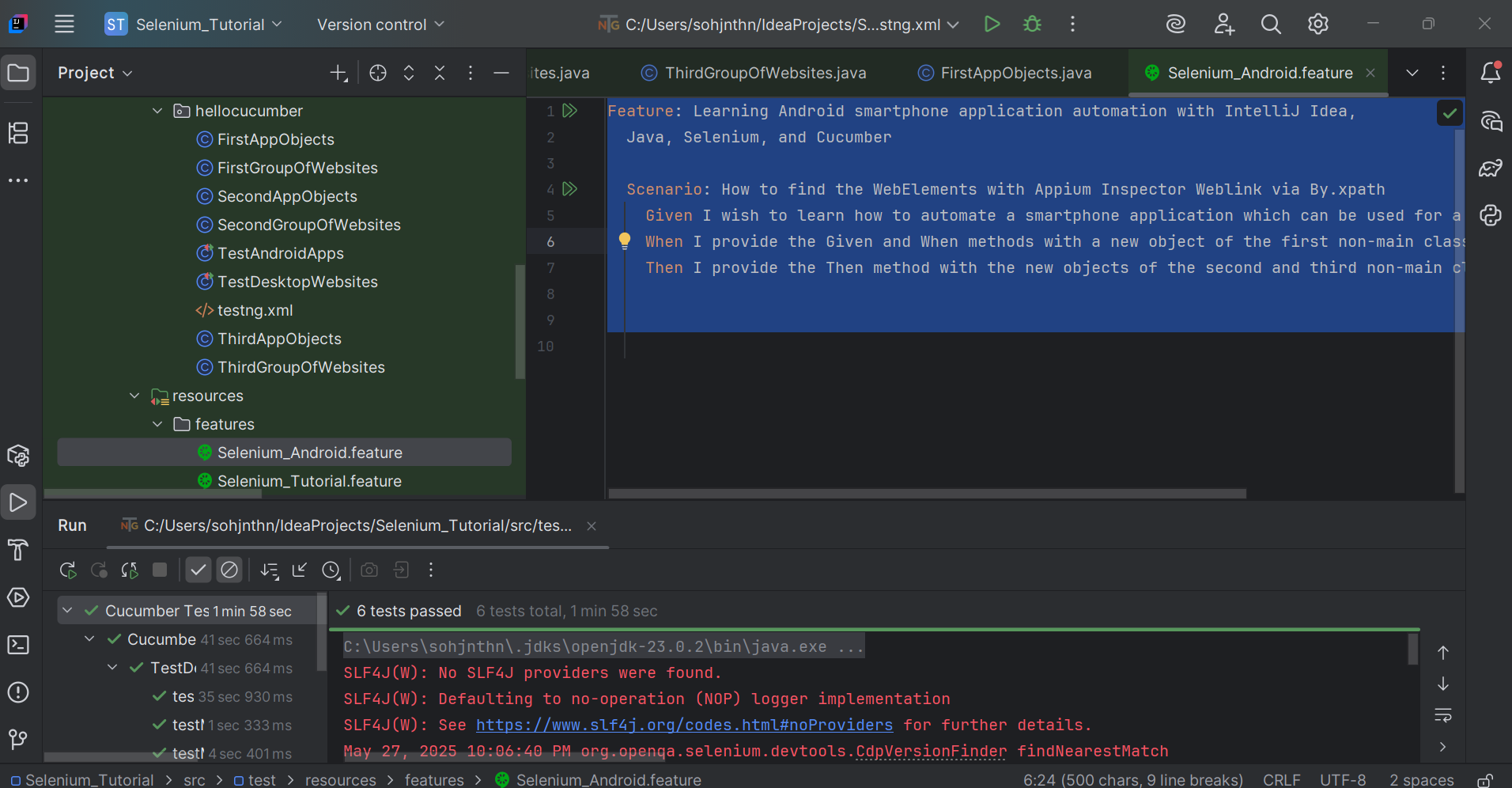
Given I wish to learn how to automate a smartphone application which can be used for a real Android 15 device

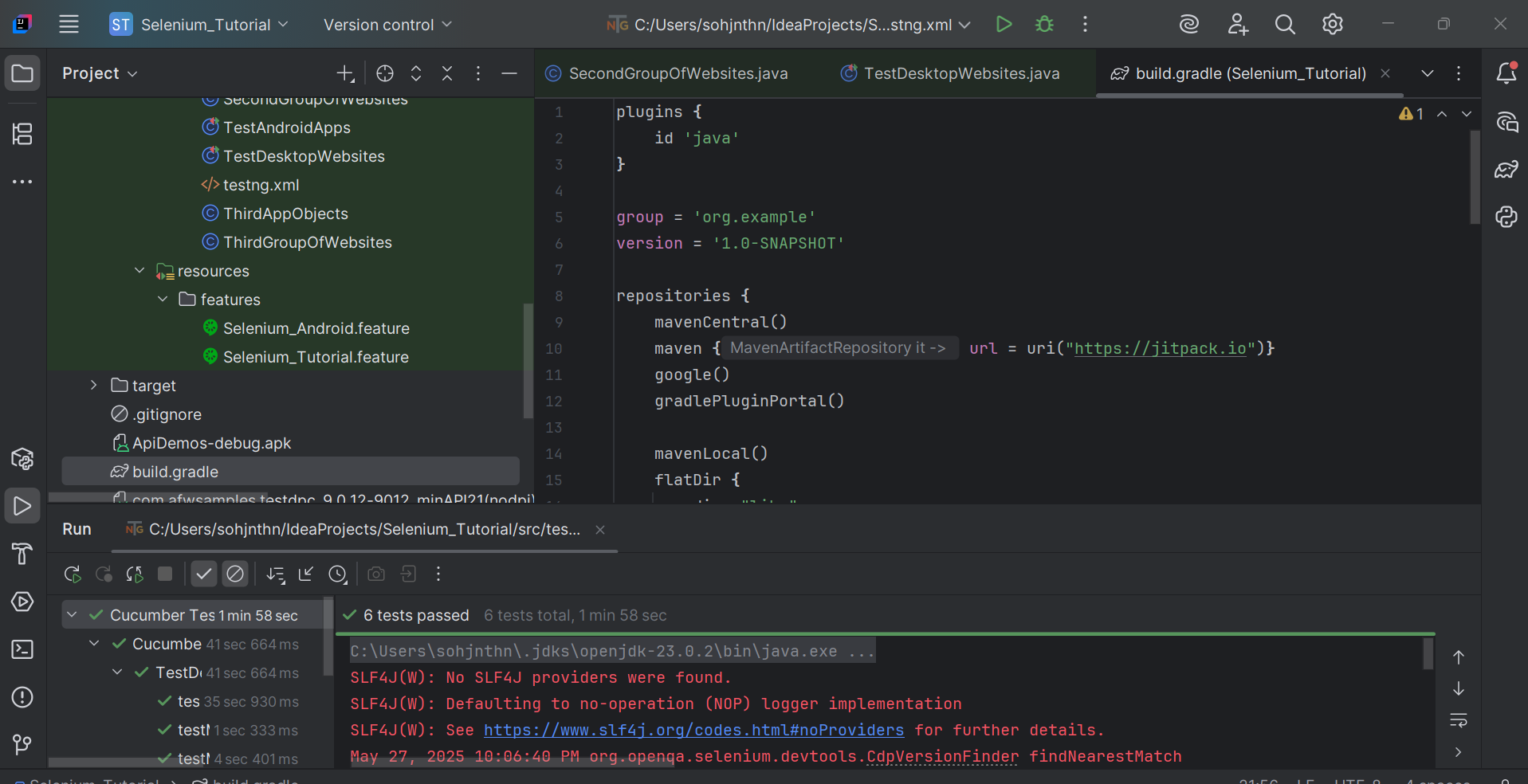
When I provide the Given and When methods with a new object of the first non-main class

Then I provide the Then method with the new objects of the second and third non-main classes









build.gradle

plugins **{**

id 'java'

**}**

group = 'org.example'

version = '1.0-SNAPSHOT'

repositories **{**

mavenCentral()

maven **{** url = uri("https://jitpack.io")**}**

google()

gradlePluginPortal()

mavenLocal()

flatDir **{**

dirs "libs"

**}**

**}**

dependencies **{**

implementation 'io.cucumber:cucumber-testng:7.22.2'

implementation 'org.testng:testng:7.11.0'

implementation 'io.appium:java-client:9.4.0'

implementation 'io.cucumber:cucumber-junit:7.22.1'

implementation 'org.junit.jupiter:junit-jupiter-api:5.12.2'

implementation 'org.junit.jupiter:junit-jupiter-engine:5.12.2'

implementation 'io.cucumber:cucumber-core:7.22.1'

implementation 'io.cucumber:cucumber-spring:7.22.1'

implementation 'org.apache.groovy:groovy-all:4.0.26'

implementation 'io.cucumber:cucumber-gherkin:7.22.1'

implementation 'io.cucumber:gherkin:32.1.1'

implementation 'io.cucumber:gherkin-utils:9.2.0'

implementation 'io.cucumber:cucumber-jvm-groovy:6.10.4'

implementation 'io.cucumber:cucumber-groovy:6.10.4'

implementation 'io.cucumber:cucumber-java:7.22.1'

implementation 'org.seleniumhq.selenium:selenium-chrome-driver:4.32.0'

implementation 'com.aventstack:extentreports:5.1.2';

implementation 'org.springframework:spring-beans:6.2.6'

implementation 'org.springframework:spring-core:6.2.6'

implementation 'org.springframework:spring-web:6.2.6'

implementation 'org.springframework:spring-test:6.2.6'

implementation 'org.springframework:spring-webmvc:6.2.6'

implementation 'org.springframework:spring-jms:6.2.6'

implementation 'org.springframework:spring-messaging:6.2.6'

implementation 'org.springframework:spring-aop:6.2.6'

implementation 'org.springframework:spring-orm:6.2.6'

implementation 'org.springframework:spring-context-support:6.2.6'

implementation 'org.springframework:spring-context:6.2.6'

implementation 'org.springframework:spring-tx:6.2.6'

implementation 'org.springframework:spring-jdbc:6.2.6'

**}**

test **{**

useJUnitPlatform()

**}**

TestDesktopWebsites.java

package hellocucumber;

import io.cucumber.testng.AbstractTestNGCucumberTests;

import com.aventstack.extentreports.ExtentReports;

import com.aventstack.extentreports.ExtentTest;

import com.aventstack.extentreports.MediaEntityBuilder;

import com.aventstack.extentreports.Status;

import com.aventstack.extentreports.reporter.ExtentSparkReporter;

import io.cucumber.java.en.Given;

import io.cucumber.java.en.Then;

import io.cucumber.java.en.When;

import io.cucumber.junit.Cucumber;

import io.cucumber.junit.CucumberOptions;

import io.cucumber.spring.CucumberContextConfiguration;

import org.junit.runner.RunWith;

import org.openqa.selenium.OutputType;

import org.openqa.selenium.By;

import org.openqa.selenium.Keys;

import org.openqa.selenium.WebDriver;

import org.openqa.selenium.WebElement;

import org.openqa.selenium.TakesScreenshot;

import org.openqa.selenium.chrome.ChromeDriver;

import org.openqa.selenium.io.FileHandler;

import org.testng.annotations.Test;

import java.io.File;

import java.io.IOException;

import java.util.List;

import java.time.LocalDateTime;

import java.util.Objects;

//TIP To <b>Run</b> code, press <shortcut actionId="Run"/> or

// click the <icon src="AllIcons.Actions.Execute"/> icon in the gutter.

@CucumberContextConfiguration

@RunWith(Cucumber.class)

@CucumberOptions(

// features should start with "src/test/resources/features/xxxxxxxxxxx.feature",

features = "src/test/resources/features/Selenium\_Tutorial.feature",

// glue starts with "src/test/java/#package for the Test Java class with the Given;When;Then annotations#/",

glue={"src/test/java/hellocucumber/"},

// plugin = = {"pretty", "html:target/cucumber-reports"}

plugin = {"pretty", "html:target/cucumber-reports"}

)

public class TestDesktopWebsites {

// Test.java class should be within src > test

private WebDriver driver1 = new ChromeDriver();

private String websiteTitle = "";

private String capturedScreenshotImageFilepathString = "";

private String currentLocalDateTimeForExtentSparkReporter = *currentLocalDateTimeWithDdMmYyFormat*();

private ExtentReports extentReport = new ExtentReports();

// should be target/Spark.html, not an actual filepath starting from C:\\

private ExtentSparkReporter extentSparkReporter = new ExtentSparkReporter("target/Spark\_" + currentLocalDateTimeForExtentSparkReporter + ".html");

private ExtentTest extentTest = extentReport.createTest("Selenium\_Tutorial Test");

private FirstGroupOfWebsites firstGroupOfWebsites;

private SecondGroupOfWebsites secondGroupOfWebsites;

private ThirdGroupOfWebsites thirdGroupOfWebsites;

public TestDesktopWebsites() throws IOException {

}

// Note that this method is not executed, if running the feature with the Given, When, Then lines

void testMethod() {

System.*out*.println("testing the IntelliJ IDEA - Help - Testing - Create Tests - Right-click to generate Test Method for JUnit5.");

}

@org.junit.Test

@org.junit.jupiter.api.Test

@org.testng.annotations.Test

@Given("that the easiest way to find WebElements is by driver.findElement; and driver.findElements which returns a List WebElement")

public void testMethod1() throws IOException {

// this line only works without an error, in a public method, as it is a public void method

// the other three lines (originally above this fourth line) are public methods

extentReport.attachReporter(extentSparkReporter);

FirstGroupOfWebsites firstGroupOfWebsites = new FirstGroupOfWebsites((ChromeDriver) driver1, extentTest, extentSparkReporter, extentReport);

System.*out*.println("testing");

}

@org.junit.Test

@org.junit.jupiter.api.Test

@org.testng.annotations.Test

@When("it is not possible to do so for certain WebElements in this way")

public void testMethod2 () throws IOException {

SecondGroupOfWebsites secondGroupOfWebsites = new SecondGroupOfWebsites((ChromeDriver) driver1, extentTest, extentSparkReporter, extentReport);

}

@org.junit.Test

@org.junit.jupiter.api.Test

@org.testng.annotations.Test

@Then("we use driver.findElement By.tagName\"...\", for which ... can be i for <i> or button for <button>")

public void testMethod3 () throws IOException {

ThirdGroupOfWebsites thirdGroupOfWebsites = new ThirdGroupOfWebsites((ChromeDriver) driver1, extentTest, extentSparkReporter, extentReport);

}

public static String currentLocalDateTimeWithDdMmYyFormat() throws IOException {

char dayFirstDigitChar = '0';

char daySecondDigitChar = '0';

char monthFirstDigitChar = '0';

char monthSecondDigitChar = '0';

char yearFirstDigitChar = '0';

char yearSecondDigitChar = '0';

char yearThirdDigitChar = '0';

char yearFourthDigitChar = '0';

char hourFirstDigitChar = '0';

char hourSecondDigitChar = '0';

char minuteFirstDigitChar = '0';

char minuteSecondDigitChar = '0';

char secondFirstDigitChar = '0';

char secondSecondDigitChar = '0';

char subsecondFirstDigitChar = '0';

char subsecondSecondDigitChar = '0';

char subsecondThirdDigitChar = '0';

char subsecondFourthDigitChar = '0';

char subsecondFifthDigitChar = '0';

char subsecondSixthDigitChar = '0';

// Issue - Missing these three digits

char subsecondSeventhDigitChar = '0';

char subsecondEighthDigitChar = '0';

char subsecondNinthDigitChar = '0';

// How to take Selenium screenshot image

LocalDateTime localDateTime = LocalDateTime.*now*();

String initialLocalDateTimeString = localDateTime.toString();

String initialLocalDateTimeStringDaySubstring = initialLocalDateTimeString.substring(8, 10);

String initialLocalDateTimeStringDashMonthDashSubstring = initialLocalDateTimeString.substring(4, 8);

String initialLocalDateTimeStringYearSubstring = initialLocalDateTimeString.substring(0, 4);

String initialLocalDateTimeStringTimeSubString = initialLocalDateTimeString.substring(10);

// Not providing as (10, 29) which caused an issue, as found out that intialLocalDateTimeString can be have 26, 27, 28, or 29 characters

String finalLocalDateTimeString = initialLocalDateTimeStringDaySubstring + initialLocalDateTimeStringDashMonthDashSubstring + initialLocalDateTimeStringYearSubstring + initialLocalDateTimeStringTimeSubString;

if(finalLocalDateTimeString.length()>=1) {

// index 0 character; 1st character

dayFirstDigitChar = finalLocalDateTimeString.charAt(0);

}

if(finalLocalDateTimeString.length()>=2) {

// index 1 character; 2nd character

daySecondDigitChar = finalLocalDateTimeString.charAt(1);

}

if(finalLocalDateTimeString.length()>=4) {

// index 3 character; 4th character

monthFirstDigitChar = finalLocalDateTimeString.charAt(3);

}

if(finalLocalDateTimeString.length()>=5) {

// index 4 character; 5th character

monthSecondDigitChar = finalLocalDateTimeString.charAt(4);

}

if(finalLocalDateTimeString.length()>=7) {

// index 6 character; 7th character

yearFirstDigitChar = finalLocalDateTimeString.charAt(6);

}

if(finalLocalDateTimeString.length()>=8) {

// index 7 character; 8th character

yearSecondDigitChar = finalLocalDateTimeString.charAt(7);

}

if(finalLocalDateTimeString.length()>=9) {

// index 8 character; 9th character

yearThirdDigitChar = finalLocalDateTimeString.charAt(8);

}

if(finalLocalDateTimeString.length()>=10) {

// index 9 character; 10th character

yearFourthDigitChar = finalLocalDateTimeString.charAt(9);

}

if(finalLocalDateTimeString.length()>=12) {

// index 11 character; 12th character

hourFirstDigitChar = finalLocalDateTimeString.charAt(11);

}

if(finalLocalDateTimeString.length()>=13){

// index 12 character; 13th character

hourSecondDigitChar = finalLocalDateTimeString.charAt(12);

}

if(finalLocalDateTimeString.length()>=15) {

// index 14 character; 15th character

minuteFirstDigitChar = finalLocalDateTimeString.charAt(14);

}

if(finalLocalDateTimeString.length()>=16) {

// index 15 character; 16th character

minuteSecondDigitChar = finalLocalDateTimeString.charAt(15);

}

if(finalLocalDateTimeString.length()>=18) {

// index 17 character; 18th character

secondFirstDigitChar = finalLocalDateTimeString.charAt(17);

}

if(finalLocalDateTimeString.length()>=19) {

// index 18 character; 19th character

secondSecondDigitChar = finalLocalDateTimeString.charAt(18);

}

if(finalLocalDateTimeString.length()>=21) {

// index 20 character; 21th character

subsecondFirstDigitChar = finalLocalDateTimeString.charAt(20);

}

if(finalLocalDateTimeString.length()>=22) {

// index 21 character; 22nd character

subsecondSecondDigitChar = finalLocalDateTimeString.charAt(21);

}

if(finalLocalDateTimeString.length()>=23) {

// index 22 character; 23rd character

subsecondThirdDigitChar = finalLocalDateTimeString.charAt(22);

}

if(finalLocalDateTimeString.length()>=24) {

// index 23 character; 24th character

subsecondFourthDigitChar = finalLocalDateTimeString.charAt(23);

}

if(finalLocalDateTimeString.length()>=25) {

// index 24 character; 25th character

subsecondFifthDigitChar = finalLocalDateTimeString.charAt(24);

}

if(finalLocalDateTimeString.length()>=26) {

// index 25 character; 26th character

subsecondSixthDigitChar = finalLocalDateTimeString.charAt(25);

}

if(finalLocalDateTimeString.length()>=27) {

// index 26 character; 27th character

subsecondSeventhDigitChar = finalLocalDateTimeString.charAt(26);

}

if(finalLocalDateTimeString.length()>=28) {

// index 27 character; 28th character

subsecondEighthDigitChar = finalLocalDateTimeString.charAt(27);

}

if(finalLocalDateTimeString.length()>=29) {

// index 28 character; 29th character

subsecondNinthDigitChar = finalLocalDateTimeString.charAt(28);

}

// There seems to be an error for which the two digits for day are incorrect, so localDateTime.getDayOfmonth is used(), instead of dayFirstDigitChar and daySecondDigitChar

return localDateTime.getDayOfMonth() + "-" + monthFirstDigitChar + monthSecondDigitChar + "-" + yearFirstDigitChar + yearSecondDigitChar + yearThirdDigitChar + yearFourthDigitChar + "T" + hourFirstDigitChar + hourSecondDigitChar + minuteFirstDigitChar + minuteSecondDigitChar + secondFirstDigitChar + secondSecondDigitChar + "." + subsecondFirstDigitChar + subsecondSecondDigitChar + subsecondThirdDigitChar + subsecondFourthDigitChar + subsecondFifthDigitChar + subsecondSixthDigitChar + subsecondSeventhDigitChar + subsecondEighthDigitChar + subsecondNinthDigitChar;

}

}

FirstGroupOfWebsites.java

package hellocucumber;

import com.aventstack.extentreports.ExtentReports;

import com.aventstack.extentreports.ExtentTest;

import com.aventstack.extentreports.MediaEntityBuilder;

import com.aventstack.extentreports.Status;

import com.aventstack.extentreports.reporter.ExtentSparkReporter;

import org.apache.tools.ant.types.resources.First;

import org.openqa.selenium.\*;

import org.openqa.selenium.chrome.ChromeDriver;

import org.openqa.selenium.io.FileHandler;

import java.io.File;

import java.io.IOException;

import java.time.LocalDateTime;

import java.util.List;

import java.util.Objects;

public class FirstGroupOfWebsites {

private String websiteTitle = "";

private String capturedScreenshotImageFilepathString = "";

public FirstGroupOfWebsites (ChromeDriver driver, ExtentTest extentTest, ExtentSparkReporter extentSparkReporter, ExtentReports extentReport) throws IOException {

System.*out*.println("testing");

String capturedScreenshotImageFilepathString = new String("");

// Captured screenshot image filename to be used for ExtentReport test step - Pass; or Fail

capturedScreenshotImageFilepathString = *takeScreenshot*(driver);

driver.get("https://demo.guru99.com/");

// Captured screenshot image filename to be used for ExtentReport test step - Pass; or Fail

capturedScreenshotImageFilepathString = *takeScreenshot*(driver);

// The web browser tab's title (depends on the website)

String websiteTitle = driver.getTitle();

System.*out*.println("The website title is \"" + websiteTitle + "\".");

// Note: Not all web browser tab titles work

if (Objects.*equals*(websiteTitle, "Guru99 Bank Home Page")) {

// requires actual filepath of takeScreenshot(WebDriver object)

System.*out*.println("The current web browser tab title is " + "\"" + websiteTitle + "\"; which should be \"Guru99 Bank Home Page\".");

extentTest.addScreenCaptureFromPath(capturedScreenshotImageFilepathString).pass(MediaEntityBuilder.*createScreenCaptureFromPath*(capturedScreenshotImageFilepathString).build()).log(Status.*PASS*, "The current web browser tab title is " + "\"" + websiteTitle + "\"; which should be \"Guru99 Bank Home Page\".");

} else {

System.*out*.println("The current web browser tab title cannot be retrieved; which should be \"Guru99 Bank Home Page\".");

extentTest.addScreenCaptureFromPath(capturedScreenshotImageFilepathString).pass(MediaEntityBuilder.*createScreenCaptureFromPath*(capturedScreenshotImageFilepathString).build()).log(Status.*FAIL*, "The current web browser tab title cannot be retrieved; which should be \"Guru99 Bank Home Page\".");

}

WebElement emailInputField = driver.findElement(By.*name*("emailid"));

// Captured screenshot image filename to be used for ExtentReport test step - Pass; or Fail

capturedScreenshotImageFilepathString = *takeScreenshot*(driver);

// check for empty String for email input field value

//String emailInputFieldText = emailInputField.getText();

// Only for testing purposes, to ensure that WebElement object.getText() works

// Capture screenshot image, and input field text value immediately after sending the String to the input field

if (emailInputField.isDisplayed()) {

emailInputField.sendKeys("abc@gmail.com");

// Captured screenshot image filename to be used for ExtentReport test step - Pass; or Fail

capturedScreenshotImageFilepathString = *takeScreenshot*(driver);

}

// actual verification for pass or fail

// check for email input field value being "abc@gmail.com"

// emailInputFieldText = emailInputField.getText();

// Note: Text input field cannot allow Selenium to find the text with WebElement object.getText();

if (emailInputField.isDisplayed()) {

extentTest.addScreenCaptureFromPath(capturedScreenshotImageFilepathString).pass(MediaEntityBuilder.*createScreenCaptureFromPath*(capturedScreenshotImageFilepathString).build()).log(Status.*PASS*, "The email input field value's text is the correct \"abc@gmail.com\".");

} else {

extentTest.addScreenCaptureFromPath(capturedScreenshotImageFilepathString).pass(MediaEntityBuilder.*createScreenCaptureFromPath*(capturedScreenshotImageFilepathString).build()).log(Status.*FAIL*, "The email input field value's text is not \"abc@gmail.com\".");

}

String emailText = "abc@gmail.com";

int emailLength = emailText.length();

// To use backspace to clear the initially entered emailText

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

emailInputField.sendKeys(Keys.*BACK\_SPACE*);

}

// check for the email input field value being ""

// emailInputFieldText = emailInputField.getText();

// Captured screenshot image filename to be used for ExtentReport test step - Pass; or Fail

capturedScreenshotImageFilepathString = *takeScreenshot*(driver);

if (emailInputField.isDisplayed()) {

extentTest.addScreenCaptureFromPath(capturedScreenshotImageFilepathString).pass(MediaEntityBuilder.*createScreenCaptureFromPath*(capturedScreenshotImageFilepathString).build()).log(Status.*PASS*, "The email input field value's text is the correct \"\".");

} else {

extentTest.addScreenCaptureFromPath(capturedScreenshotImageFilepathString).pass(MediaEntityBuilder.*createScreenCaptureFromPath*(capturedScreenshotImageFilepathString).build()).log(Status.*FAIL*, "The email input field value's text is not \"\".");

}

//emailInputFieldText = emailInputField.getText();

extentTest.addScreenCaptureFromPath(capturedScreenshotImageFilepathString).pass(MediaEntityBuilder.*createScreenCaptureFromPath*(capturedScreenshotImageFilepathString).build()).log(Status.*PASS*, "The email input field value \"abc@gmail.com\" has been removed by the Backspace key.");

emailInputField.sendKeys("abc@gmail.com");

// Captured screenshot image filename to be used for ExtentReport test step - Pass; or Fail

capturedScreenshotImageFilepathString = *takeScreenshot*(driver);

//emailInputFieldText = emailInputField.getText();

// check for email input field value being "abc@gmail.com"

if (emailInputField.isDisplayed()) {

extentTest.addScreenCaptureFromPath(capturedScreenshotImageFilepathString).pass(MediaEntityBuilder.*createScreenCaptureFromPath*(capturedScreenshotImageFilepathString).build()).log(Status.*PASS*, "The email input field value's text is the correct \"abc@gmail.com\".");

} else {

extentTest.addScreenCaptureFromPath(capturedScreenshotImageFilepathString).pass(MediaEntityBuilder.*createScreenCaptureFromPath*(capturedScreenshotImageFilepathString).build()).log(Status.*FAIL*, "The email input field value's text is not \"abc@gmail.com\".");

}

// The Submit button

WebElement submitButton = driver.findElement(By.*name*("btnLogin"));

if (submitButton.isDisplayed()) {

submitButton.click();

// Captured screenshot image filename to be used for ExtentReport test step - Pass; or Fail

capturedScreenshotImageFilepathString = *takeScreenshot*(driver);

}

// The web browser tab's title (depends on the website)

websiteTitle = driver.getTitle();

System.*out*.println("The website title is \"" + websiteTitle + "\".");

// Note: Not all web browser tab titles work

if (Objects.*equals*(websiteTitle, "Guru99 Bank Home Page")) {

// requires actual filepath of takeScreenshot(WebDriver object)

System.*out*.println("The current web browser tab title is " + "\"" + websiteTitle + "\"; which should be \"Guru99 Bank Home Page\".");

extentTest.addScreenCaptureFromPath(capturedScreenshotImageFilepathString).pass(MediaEntityBuilder.*createScreenCaptureFromPath*(capturedScreenshotImageFilepathString).build()).log(Status.*PASS*, "The current web browser tab title is " + "\"" + websiteTitle + "\"; which should be \"Guru99 Bank Home Page\".");

} else {

System.*out*.println("The current web browser tab title cannot be retrieved; which should be \"Guru99 Bank Home Page\".");

extentTest.addScreenCaptureFromPath(capturedScreenshotImageFilepathString).pass(MediaEntityBuilder.*createScreenCaptureFromPath*(capturedScreenshotImageFilepathString).build()).log(Status.*FAIL*, "The current web browser tab title cannot be retrieved; which should be \"Guru99 Bank Home Page\".");

}

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

// Captured screenshot image filename to be used for ExtentReport test step - Pass; or Fail

capturedScreenshotImageFilepathString = *takeScreenshot*(driver);

// The web browser tab's title (depends on the website)

websiteTitle = driver.getTitle();

System.*out*.println("The website title is \"" + websiteTitle + "\".");

// Note: Not all web browser tab titles work

if (Objects.*equals*(websiteTitle, "Search - Microsoft Bing")) {

extentTest.addScreenCaptureFromPath(capturedScreenshotImageFilepathString).pass(MediaEntityBuilder.*createScreenCaptureFromPath*(capturedScreenshotImageFilepathString).build()).log(Status.*PASS*, "The current web browser tab title is the correct " + "\"" + websiteTitle + "\"; which should be \"Search - Microsoft Bing\".");

} else {

extentTest.addScreenCaptureFromPath(capturedScreenshotImageFilepathString).pass(MediaEntityBuilder.*createScreenCaptureFromPath*(capturedScreenshotImageFilepathString).build()).log(Status.*FAIL*, "The current web browser tab title is the incorrect" + "\"" + websiteTitle + "\"; which should be \"Search - Microsoft Bing\".");

}

// Microsoft Bing search bar

WebElement microsoftBingSearchBar = driver.findElement(By.*name*("q"));

// String microsoftBingSearchBarText = microsoftBingSearchBar.getText();

// Selenium does not allow text input field value to be retrieved with WebElement object.getText();

// Search bar should initially have an empty String

if (microsoftBingSearchBar.isDisplayed()) {

microsoftBingSearchBar.sendKeys("MapleStorySEA Unfunded Amino");

// Captured screenshot image filename to be used for ExtentReport test step - Pass; or Fail

capturedScreenshotImageFilepathString = *takeScreenshot*(driver);

}

if (microsoftBingSearchBar.isDisplayed()) {

extentTest.addScreenCaptureFromPath(capturedScreenshotImageFilepathString).pass(MediaEntityBuilder.*createScreenCaptureFromPath*(capturedScreenshotImageFilepathString).build()).log(Status.*PASS*, "The Microsoft Bing search term input is the correct \"MapleStorySEA Unfunded Amino\".");

microsoftBingSearchBar.sendKeys(Keys.*ENTER*);

// Captured screenshot image filename to be used for ExtentReport test step - Pass; or Fail

capturedScreenshotImageFilepathString = *takeScreenshot*(driver);

} else {

extentTest.addScreenCaptureFromPath(capturedScreenshotImageFilepathString).pass(MediaEntityBuilder.*createScreenCaptureFromPath*(capturedScreenshotImageFilepathString).build()).log(Status.*FAIL*, "The Microsoft Bing search term input is the incorrect \"MapleStorySEA Unfunded Amino\".");

microsoftBingSearchBar.sendKeys(Keys.*ENTER*);

// Captured screenshot image filename to be used for ExtentReport test step - Pass; or Fail

capturedScreenshotImageFilepathString = *takeScreenshot*(driver);

}

websiteTitle = driver.getTitle();

System.*out*.println("The website title is \"" + websiteTitle + "\".");

// Note: Not all web browser tab titles work

// Microsoft Bing - First Page of search results - Default All tab

if (Objects.*equals*(websiteTitle, "MapleStorySEA Unfunded Amino - Search")) {

System.*out*.println("The current web browser tab title is " + "\"" + websiteTitle + "\".");

extentTest.addScreenCaptureFromPath(capturedScreenshotImageFilepathString).pass(MediaEntityBuilder.*createScreenCaptureFromPath*(capturedScreenshotImageFilepathString).build()).log(Status.*PASS*, "The Microsoft Bing search has been triggered, to reach the webpage with the web browser tab title \"MapleStorySEA Unfunded Amino - Search\".");

} else {

System.*out*.println("The current web browser tab title cannot be retrieved.");

extentTest.addScreenCaptureFromPath(capturedScreenshotImageFilepathString).pass(MediaEntityBuilder.*createScreenCaptureFromPath*(capturedScreenshotImageFilepathString).build()).log(Status.*FAIL*, "The Microsoft Bing search has not been triggered, to reach the webpage with the web browser tab title \"MapleStorySEA Unfunded Amino - Search\".");

}

// MapleStory Unfunded Amino (Microsoft Bing search results - Search tab)

//websiteTitle = driver.getTitle();

// System.out.println("The website title is \"" + websiteTitle + "\".");

// Search engine result Search tab

// id is based on the line above highlighted text for inspected element

// Still on the same initial Search Results webpage

// Should only check for whether the WebElement searchTab is displayed

WebElement searchTab = driver.findElement(By.*id*("b-scopeListItem-copilotsearch"));

if (searchTab.isDisplayed()) {

searchTab.click();

// Captured screenshot image filename to be used for ExtentReport test step - Pass; or Fail

capturedScreenshotImageFilepathString = *takeScreenshot*(driver);

extentTest.addScreenCaptureFromPath(capturedScreenshotImageFilepathString).pass(MediaEntityBuilder.*createScreenCaptureFromPath*(capturedScreenshotImageFilepathString).build()).log(Status.*PASS*, "The Microsoft Bing search results - Search tab has been reached.");

} else {

searchTab.click();

// Captured screenshot image filename to be used for ExtentReport test step - Pass; or Fail

capturedScreenshotImageFilepathString = *takeScreenshot*(driver);

extentTest.addScreenCaptureFromPath(capturedScreenshotImageFilepathString).pass(MediaEntityBuilder.*createScreenCaptureFromPath*(capturedScreenshotImageFilepathString).build()).log(Status.*FAIL*, "The Microsoft Bing search results - Search tab has not been reached.");

}

// Go back to All tab from Search tab webpage

// id is based on the line above highlighted text for inspected element

WebElement allTab = driver.findElement(By.*id*("b-scopeListItem-web"));

if (allTab.isDisplayed()) {

allTab.click();

// Captured screenshot image filename to be used for ExtentReport test step - Pass; or Fail

capturedScreenshotImageFilepathString = *takeScreenshot*(driver);

extentTest.addScreenCaptureFromPath(capturedScreenshotImageFilepathString).pass(MediaEntityBuilder.*createScreenCaptureFromPath*(capturedScreenshotImageFilepathString).build()).log(Status.*PASS*, "The Microsoft Bing search results - All tab has been reached.");

} else {

allTab.click();

// Captured screenshot image filename to be used for ExtentReport test step - Pass; or Fail

capturedScreenshotImageFilepathString = *takeScreenshot*(driver);

extentTest.addScreenCaptureFromPath(capturedScreenshotImageFilepathString).pass(MediaEntityBuilder.*createScreenCaptureFromPath*(capturedScreenshotImageFilepathString).build()).log(Status.*FAIL*, "The Microsoft Bing search results - All tab has not been reached.");

}

websiteTitle = driver.getTitle();

System.*out*.println("The website title is \"" + websiteTitle + "\".");

// Note: Not all web browser tab titles work

if (Objects.*equals*(websiteTitle, "MapleStorySEA Unfunded Amino - Search")) {

System.*out*.println("The current web browser tab title is " + "\"" + websiteTitle + "\".");

extentTest.addScreenCaptureFromPath(capturedScreenshotImageFilepathString).pass(MediaEntityBuilder.*createScreenCaptureFromPath*(capturedScreenshotImageFilepathString).build()).log(Status.*PASS*, "The Microsoft Bing search results webpage has been reached.");

} else {

System.*out*.println("The current web browser tab title cannot be retrieved.");

extentTest.addScreenCaptureFromPath(capturedScreenshotImageFilepathString).pass(MediaEntityBuilder.*createScreenCaptureFromPath*(capturedScreenshotImageFilepathString).build()).log(Status.*FAIL*, "The Microsoft Bing search results webpage has not been reached.");

}

// web browser search engine result uses partialLinkText is not working, due to YouTube videos being displayed at the right sometimes

//WebElement requiredSearchResultLink = driver.findElement(By.partialLinkText("Featured | [MapleStorySEA] Unfunded Tips Amino - Amino Apps"));

// Due to video section appearing at the right

// Which causes less text for the Microsoft Bing search results hyperlink to be displayed

WebElement requiredSearchResultLink = driver.findElement(By.*partialLinkText*("Featured | [MapleStorySEA] Unfunded Tips"));

// Use tagName "a" instead

// Cannot use tagName, keeps changing position in List <WebElement>

//List <WebElement> tagNameAWebElementList = driver.findElements(By.tagName("a"));

//WebElement requiredSearchResultLink = tagNameAWebElementList.get(30);

// String requiredSearchResultPartialLinkTextString = requiredSearchResultPartialLinkText.getText();

//String requiredSearchResultLinkText = requiredSearchResultLink.getText();

//if (requiredSearchResultPartialLinkText.isDisplayed() && requiredSearchResultPartialLinkTextString.equals("Featured | [MapleStorySEA] Unfunded Tips Amino - Amino Apps")) {

if (requiredSearchResultLink.isDisplayed()) {

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

//requiredSearchResultPartialLinkText.sendKeys(Keys.DOWN);

requiredSearchResultLink.sendKeys(Keys.*DOWN*);

}

// Captured screenshot image filename to be used for ExtentReport test step - Pass; or Fail

capturedScreenshotImageFilepathString = *takeScreenshot*(driver);

extentTest.addScreenCaptureFromPath(capturedScreenshotImageFilepathString).pass(MediaEntityBuilder.*createScreenCaptureFromPath*(capturedScreenshotImageFilepathString).build()).log(Status.*PASS*, "The Microsoft Bing search results - bottom of the first page been reached; by using the WebElement for requiredSearchResultLink \"Featured | [MapleStorySEA] Unfunded Tips Amino - Amino Apps\".");

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

//requiredSearchResultPartialLinkText.sendKeys(Keys.UP);

requiredSearchResultLink.sendKeys(Keys.*UP*);

}

*takeScreenshot*(driver);

extentTest.addScreenCaptureFromPath(capturedScreenshotImageFilepathString).pass(MediaEntityBuilder.*createScreenCaptureFromPath*(capturedScreenshotImageFilepathString).build()).log(Status.*PASS*, "The Microsoft Bing search results - top of the first page has been reached; by using the WebElement for requiredSearchResultLink \"Featured | [MapleStorySEA] Unfunded Tips Amino - Amino Apps\".");

//requiredSearchResultPartialLinkText.click();

requiredSearchResultLink.click();

//}

}

// Problem: Search engine search result opens in a new web browser tab

// New ChromeDriver object needs to be created to open a new web browser window

driver.get("https://aminoapps.com/c/maplestorysea/home/");

capturedScreenshotImageFilepathString = *takeScreenshot*(driver);

extentTest.addScreenCaptureFromPath(capturedScreenshotImageFilepathString).pass(MediaEntityBuilder.*createScreenCaptureFromPath*(capturedScreenshotImageFilepathString).build()).log(Status.*PASS*, "The Amino website has been reached.");

websiteTitle = driver.getTitle();

System.*out*.println("The website title is \"" + websiteTitle + "\".");

// Note: Not all web browser tab titles work

if (Objects.*equals*(websiteTitle, "Featured | [MapleStorySEA] Unfunded Tips Amino")) {

System.*out*.println("The current web browser tab title is the correct " + "\"" + websiteTitle + "\"; and should be \"Featured | [MapleStorySEA] Unfunded Tips Amino\".");

extentTest.addScreenCaptureFromPath(capturedScreenshotImageFilepathString).pass(MediaEntityBuilder.*createScreenCaptureFromPath*(capturedScreenshotImageFilepathString).build()).log(Status.*PASS*, "The current web browser tab title is the correct " + "\"" + websiteTitle + "\"; and should be \"Featured | [MapleStorySEA] Unfunded Tips Amino\".");

} else {

System.*out*.println("The current web browser tab title is the incorrect " + "\"" + websiteTitle + "\"; and should be \"Featured | [MapleStorySEA] Unfunded Tips Amino\".");

extentTest.addScreenCaptureFromPath(capturedScreenshotImageFilepathString).pass(MediaEntityBuilder.*createScreenCaptureFromPath*(capturedScreenshotImageFilepathString).build()).log(Status.*FAIL*, "The current web browser tab title is the incorrect " + "\"" + websiteTitle + "\"; and should be \"Featured | [MapleStorySEA] Unfunded Tips Amino\".");

}

WebElement aminoSearchBar = driver.findElement(By.*className*("nav-search-input"));

//String aminoSearchBarText = aminoSearchBar.getText();

if (aminoSearchBar.isDisplayed()) {

aminoSearchBar.sendKeys("MapleStory Unfunded Amino");

// non-empty input has been entered

// Text input field value cannot be found by Selenium

//aminoSearchBarText = aminoSearchBar.getText();

capturedScreenshotImageFilepathString = *takeScreenshot*(driver);

extentTest.addScreenCaptureFromPath(capturedScreenshotImageFilepathString).pass(MediaEntityBuilder.*createScreenCaptureFromPath*(capturedScreenshotImageFilepathString).build()).log(Status.*PASS*, "The Amino search term has been provided as the correct \"MapleStorySEA Unfunded Amino\".");

} else {

aminoSearchBar.sendKeys("MapleStory Unfunded Amino");

// non-empty input has been entered

// aminoSearchBarText = aminoSearchBar.getText();

capturedScreenshotImageFilepathString = *takeScreenshot*(driver);

extentTest.addScreenCaptureFromPath(capturedScreenshotImageFilepathString).pass(MediaEntityBuilder.*createScreenCaptureFromPath*(capturedScreenshotImageFilepathString).build()).log(Status.*FAIL*, "The Amino search term has been provided as not \"MapleStorySEA Unfunded Amino\".");

}

String aminoSearchTerm = "MapleStorySEA Unfunded Amino";

int aminoSearchTermLength = aminoSearchTerm.length();

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

aminoSearchBar.sendKeys(Keys.*BACK\_SPACE*);

}

// Captured screenshot image filename to be used for ExtentReport test step - Pass; or Fail

capturedScreenshotImageFilepathString = *takeScreenshot*(driver);

//aminoSearchBarText = aminoSearchBar.getText();

//if (aminoSearchBarText.equals("")) {

//extentTest.addScreenCaptureFromPath(capturedScreenshotImageFilepathString).pass(MediaEntityBuilder.createScreenCaptureFromPath(capturedScreenshotImageFilepathString).build()).log(Status.PASS, "The Amino search term has been removed by the Backspace key; and the current Amino search term is \"" + aminoSearchBarText + "\".");

//} else {

//extentTest.addScreenCaptureFromPath(capturedScreenshotImageFilepathString).pass(MediaEntityBuilder.createScreenCaptureFromPath(capturedScreenshotImageFilepathString).build()).log(Status.FAIL, "The Amino search term has not been removed by the Backspace key; and the current Amino search term is \"" + aminoSearchBarText + "\".");

//}

aminoSearchBar.sendKeys("MapleStorySEA Unfunded Amino");

// Captured screenshot image filename to be used for ExtentReport test step - Pass; or Fail

capturedScreenshotImageFilepathString = *takeScreenshot*(driver);

//aminoSearchBarText = aminoSearchBar.getText();

//if(aminoSearchBarText.equals("MapleStorySEA Unfunded Amino")) {

//extentTest.addScreenCaptureFromPath(capturedScreenshotImageFilepathString).pass(MediaEntityBuilder.createScreenCaptureFromPath(capturedScreenshotImageFilepathString).build()).log(Status.PASS, "The Amino search term has been re-provided as the correct \"" + aminoSearchBarText + "\", which should be \"MapleStorySEA Unfunded Amino\".");

//} else {

//extentTest.addScreenCaptureFromPath(capturedScreenshotImageFilepathString).pass(MediaEntityBuilder.createScreenCaptureFromPath(capturedScreenshotImageFilepathString).build()).log(Status.FAIL, "The Amino search term has been re-provided as the incorrect \"" + aminoSearchBarText + "\", which should be \"MapleStorySEA Unfunded Amino\".");

//}

aminoSearchBar.sendKeys(Keys.*ENTER*);

// Captured screenshot image filename to be used for ExtentReport test step - Pass; or Fail

capturedScreenshotImageFilepathString = *takeScreenshot*(driver);

//websiteTitle = driver.getTitle();

List<WebElement> tagNameH3WebElementList = driver.findElements(By.*tagName*("h3"));

WebElement popularPostsText = tagNameH3WebElementList.get(0);

String popularPostsTextString = popularPostsText.getText();

// This test step usually fails due to changing web browser tab title

// So changed to use fixed non-hyperlink text "Popular Posts", for the "Popular Posts" section

// For this case, the tag is h3

System.*out*.println("The Popular Posts' section's Popular Posts title for entering a search term from an Amino community is \"" + popularPostsTextString + "\".");

// Note: Not all web browser tab titles work

if (Objects.*equals*(popularPostsTextString, "Popular posts")) {

System.*out*.println("The Popular Posts' section's Popular Posts title for entering a search term from an Amino community is the correct " + "\"" + popularPostsTextString + "\"; which should be \"Popular Posts\".");

extentTest.addScreenCaptureFromPath(capturedScreenshotImageFilepathString).pass(MediaEntityBuilder.*createScreenCaptureFromPath*(capturedScreenshotImageFilepathString).build()).log(Status.*PASS*, "The Popular Posts' section's Popular Posts title for entering a search term from an Amino community is the correct " + "\"" + popularPostsTextString + "\"; which should be \"Popular Posts\".");

} else {

System.*out*.println("The Popular Posts' section's Popular Posts title for entering a search term from an Amino community is the incorrect " + "\"" + popularPostsTextString + "\"; which should be \"Popular Posts\".");

extentTest.addScreenCaptureFromPath(capturedScreenshotImageFilepathString).pass(MediaEntityBuilder.*createScreenCaptureFromPath*(capturedScreenshotImageFilepathString).build()).log(Status.*FAIL*, "The Popular Posts' section's Popular Posts title for entering a search term from an Amino community is the correct " + "\"" + popularPostsTextString + "\"; which should be \"Popular Posts\".");

}

driver.get("https://aminoapps.com/c/maplestorysea/home/");

// Captured screenshot image filename to be used for ExtentReport test step - Pass; or Fail

capturedScreenshotImageFilepathString = *takeScreenshot*(driver);

websiteTitle = driver.getTitle();

System.*out*.println("The website title is \"" + websiteTitle + "\".");

// Note: Not all web browser tab titles work

if (Objects.*equals*(websiteTitle, "Featured | [MapleStorySEA] Unfunded Tips Amino")) {

System.*out*.println("The current web browser tab title is " + "\"" + websiteTitle + "\".");

extentTest.addScreenCaptureFromPath(capturedScreenshotImageFilepathString).pass(MediaEntityBuilder.*createScreenCaptureFromPath*(capturedScreenshotImageFilepathString).build()).log(Status.*PASS*, "The Amino webpage is the correct \"" + websiteTitle + "\"; which should be \"Featured | [MapleStorySEA] Unfunded Tips Amino\".");

} else {

System.*out*.println("The current web browser tab title cannot be retrieved.");

extentTest.addScreenCaptureFromPath(capturedScreenshotImageFilepathString).pass(MediaEntityBuilder.*createScreenCaptureFromPath*(capturedScreenshotImageFilepathString).build()).log(Status.*FAIL*, "The Amino webpage is the incorrect \"" + websiteTitle + "\"which should be \"Featured | [MapleStorySEA] Unfunded Tips Amino\".\");.");

}

WebElement privacyNoticeAcceptButton = driver.findElement(By.*className*("confirm-close"));

if (privacyNoticeAcceptButton.isDisplayed()) {

privacyNoticeAcceptButton.click();

// Captured screenshot image filename to be used for ExtentReport test step - Pass; or Fail

capturedScreenshotImageFilepathString = *takeScreenshot*(driver);

extentTest.addScreenCaptureFromPath(capturedScreenshotImageFilepathString).pass(MediaEntityBuilder.*createScreenCaptureFromPath*(capturedScreenshotImageFilepathString).build()).log(Status.*PASS*, "The privacy notice pop-up's Accept button has been clicked on, so the pop-up has been closed.");

} else {

// Captured screenshot image filename to be used for ExtentReport test step - Pass; or Fail

capturedScreenshotImageFilepathString = *takeScreenshot*(driver);

extentTest.addScreenCaptureFromPath(capturedScreenshotImageFilepathString).pass(MediaEntityBuilder.*createScreenCaptureFromPath*(capturedScreenshotImageFilepathString).build()).log(Status.*FAIL*, "The privacy notice pop-up's Accept button has not been clicked on.");

}

// findElement is only for the first matching object for the case of multiple objects with the same name/id/className

WebElement freeCharacterSlotExpansionCoupon = driver.findElement(By.*className*("overflow-hidden"));

if (freeCharacterSlotExpansionCoupon.isDisplayed()) {

freeCharacterSlotExpansionCoupon.click();

}

websiteTitle = driver.getTitle();

System.*out*.println("The website title is \"" + websiteTitle + "\".");

// Note: Not all web browser tab titles work

if (Objects.*equals*(websiteTitle, "Reddit advice | [MapleStorySEA] Unfunded Tips Amino")) {

System.*out*.println("The current web browser tab title is " + "\"" + websiteTitle + "\".");

// Captured screenshot image filename to be used for ExtentReport test step - Pass; or Fail

capturedScreenshotImageFilepathString = *takeScreenshot*(driver);

extentTest.addScreenCaptureFromPath(capturedScreenshotImageFilepathString).pass(MediaEntityBuilder.*createScreenCaptureFromPath*(capturedScreenshotImageFilepathString).build()).log(Status.*PASS*, "The Amino webpage is the correct \"" + websiteTitle + "\"; which should be \"Reddit advice | [MapleStorySEA] Unfunded Tips Amino\".");

} else {

System.*out*.println("The current web browser tab title cannot be retrieved.");

// Captured screenshot image filename to be used for ExtentReport test step - Pass; or Fail

capturedScreenshotImageFilepathString = *takeScreenshot*(driver);

extentTest.addScreenCaptureFromPath(capturedScreenshotImageFilepathString).pass(MediaEntityBuilder.*createScreenCaptureFromPath*(capturedScreenshotImageFilepathString).build()).log(Status.*FAIL*, "The Amino webpage is the incorrect \"" + websiteTitle + "\"; which should be \"Reddit advice | [MapleStorySEA] Unfunded Tips Amino\".");

}

driver.get("https://aminoapps.com/c/maplestorysea/home/");

// Captured screenshot image filename to be used for ExtentReport test step - Pass; or Fail

capturedScreenshotImageFilepathString = *takeScreenshot*(driver);

extentTest.addScreenCaptureFromPath(capturedScreenshotImageFilepathString).pass(MediaEntityBuilder.*createScreenCaptureFromPath*(capturedScreenshotImageFilepathString).build()).log(Status.*PASS*, "The Amino webpage has been reached.");

websiteTitle = driver.getTitle();

System.*out*.println("The website title is \"" + websiteTitle + "\".");

// Note: Not all web browser tab titles work

if (Objects.*equals*(websiteTitle, "Featured | [MapleStorySEA] Unfunded Tips Amino")) {

System.*out*.println("The current web browser tab title is the correct " + "\"" + websiteTitle + "\"; which should be \"Featured | [MapleStorySEA] Unfunded Tips Amino\".");

// Captured screenshot image filename to be used for ExtentReport test step - Pass; or Fail

capturedScreenshotImageFilepathString = *takeScreenshot*(driver);

extentTest.addScreenCaptureFromPath(capturedScreenshotImageFilepathString).pass(MediaEntityBuilder.*createScreenCaptureFromPath*(capturedScreenshotImageFilepathString).build()).log(Status.*PASS*, "The current web browser tab title is the correct " + "\"" + websiteTitle + "\"; which should be \"Featured | [MapleStorySEA] Unfunded Tips Amino\".");

} else {

System.*out*.println("The current web browser tab title is the incorrect " + "\"" + websiteTitle + "\"; which should be \"Featured | [MapleStorySEA] Unfunded Tips Amino\".");

// Captured screenshot image filename to be used for ExtentReport test step - Pass; or Fail

capturedScreenshotImageFilepathString = *takeScreenshot*(driver);

extentTest.addScreenCaptureFromPath(capturedScreenshotImageFilepathString).pass(MediaEntityBuilder.*createScreenCaptureFromPath*(capturedScreenshotImageFilepathString).build()).log(Status.*FAIL*, "The current web browser tab title is the incorrect " + "\"" + websiteTitle + "\"; which should be \"Featured | [MapleStorySEA] Unfunded Tips Amino\".");

}

// findElements is for more than one matching object with the same class/name/id/partialLinkText which cannot be uniquely identified

// Base this on the Inspect - Ctrl+F order or position, search with "" marks

// className for this case is "overflow-hidden"

List<WebElement> fourOverflowHiddenClassnameWebElementList = driver.findElements(By.*className*("overflow-hidden"));

// Spell Trace and Star Force Enhancement

WebElement requiredThirdOverflowHiddenClassnameElement = fourOverflowHiddenClassnameWebElementList.get(2);

if (requiredThirdOverflowHiddenClassnameElement.isDisplayed()) {

requiredThirdOverflowHiddenClassnameElement.click();

// Captured screenshot image filename to be used for ExtentReport test step - Pass; or Fail

capturedScreenshotImageFilepathString = *takeScreenshot*(driver);

extentTest.addScreenCaptureFromPath(capturedScreenshotImageFilepathString).pass(MediaEntityBuilder.*createScreenCaptureFromPath*(capturedScreenshotImageFilepathString).build()).log(Status.*PASS*, "The Amino webpage has been reached.");

} else {

// Captured screenshot image filename to be used for ExtentReport test step - Pass; or Fail

capturedScreenshotImageFilepathString = *takeScreenshot*(driver);

extentTest.addScreenCaptureFromPath(capturedScreenshotImageFilepathString).pass(MediaEntityBuilder.*createScreenCaptureFromPath*(capturedScreenshotImageFilepathString).build()).log(Status.*FAIL*, "The Amino webpage has not been reached.");

}

websiteTitle = driver.getTitle();

System.*out*.println("The website title is \"" + websiteTitle + "\".");

// Note: Not all web browser tab titles work

if (Objects.*equals*(websiteTitle, "Unfunded Training Advice from Level 1 to Level 200? | [MapleStorySEA] Unfunded Tips Amino")) {

System.*out*.println("The current web browser tab title is the correct " + "\"" + websiteTitle + "\"; and should be \"Unfunded Training Advice from Level 1 to Level 200? | [MapleStorySEA] Unfunded Tips Amino\".");

extentTest.addScreenCaptureFromPath(capturedScreenshotImageFilepathString).pass(MediaEntityBuilder.*createScreenCaptureFromPath*(capturedScreenshotImageFilepathString).build()).log(Status.*PASS*, "The current web browser tab title is the correct " + "\"" + websiteTitle + "\"; and should be \"Unfunded Training Advice from Level 1 to Level 200? | [MapleStorySEA] Unfunded Tips Amino\".");

} else {

System.*out*.println("The current web browser tab title is the incorrect " + "\"" + websiteTitle + "\"; and should be \"Unfunded Training Advice from Level 1 to Level 200? | [MapleStorySEA] Unfunded Tips Amino\".");

extentTest.addScreenCaptureFromPath(capturedScreenshotImageFilepathString).pass(MediaEntityBuilder.*createScreenCaptureFromPath*(capturedScreenshotImageFilepathString).build()).log(Status.*FAIL*, "The current web browser tab title is the incorrect " + "\"" + websiteTitle + "\"; and should be \"Unfunded Training Advice from Level 1 to Level 200? | [MapleStorySEA] Unfunded Tips Amino\".");

}

driver.get("https://aminoapps.com/c/maplestorysea/home/");

// Captured screenshot image filename to be used for ExtentReport test step - Pass; or Fail

capturedScreenshotImageFilepathString = *takeScreenshot*(driver);

websiteTitle = driver.getTitle();

System.*out*.println("The website title is \"" + websiteTitle + "\".");

// Note: Not all web browser tab titles work

if (Objects.*equals*(websiteTitle, "Featured | [MapleStorySEA] Unfunded Tips Amino")) {

System.*out*.println("The current web browser tab title is the correct " + "\"" + websiteTitle + "\"; and should be \"Featured | [MapleStorySEA] Unfunded Tips Amino\".");

extentTest.addScreenCaptureFromPath(capturedScreenshotImageFilepathString).pass(MediaEntityBuilder.*createScreenCaptureFromPath*(capturedScreenshotImageFilepathString).build()).log(Status.*PASS*, "The current web browser tab title is the correct " + "\"" + websiteTitle + "\"; and should be \"Featured | [MapleStorySEA] Unfunded Tips Amino\".");

} else {

System.*out*.println("The current web browser tab title is the incorrect " + "\"" + websiteTitle + "\"; and should be \"Featured | [MapleStorySEA] Unfunded Tips Amino\".");

extentTest.addScreenCaptureFromPath(capturedScreenshotImageFilepathString).pass(MediaEntityBuilder.*createScreenCaptureFromPath*(capturedScreenshotImageFilepathString).build()).log(Status.*FAIL*, "The current web browser tab title is the incorrect " + "\"" + websiteTitle + "\"; and should be \"Featured | [MapleStorySEA] Unfunded Tips Amino\".");

}

// findElements is for more than one matching object with the same class/name/id/partialLinkText which cannot be uniquely identified

// Base this on the Inspect - Ctrl+F order or position, search with "" marks

// className for this case is "overflow-hidden"

List<WebElement> sevenLabelClassnameWebElementList = driver.findElements(By.*className*("label"));

// Spell Trace and Star Force Enhancement

WebElement requiredSecondLabelClassnameElement = sevenLabelClassnameWebElementList.get(1);

if (requiredSecondLabelClassnameElement.isDisplayed()) {

requiredSecondLabelClassnameElement.click();

// Captured screenshot image filename to be used for ExtentReport test step - Pass; or Fail

capturedScreenshotImageFilepathString = *takeScreenshot*(driver);

extentTest.addScreenCaptureFromPath(capturedScreenshotImageFilepathString).pass(MediaEntityBuilder.*createScreenCaptureFromPath*(capturedScreenshotImageFilepathString).build()).log(Status.*PASS*, "The Amino webpage has been reached.");

} else {

// Captured screenshot image filename to be used for ExtentReport test step - Pass; or Fail

capturedScreenshotImageFilepathString = *takeScreenshot*(driver);

extentTest.addScreenCaptureFromPath(capturedScreenshotImageFilepathString).pass(MediaEntityBuilder.*createScreenCaptureFromPath*(capturedScreenshotImageFilepathString).build()).log(Status.*FAIL*, "The Amino webpage has not been reached.");

}

websiteTitle = driver.getTitle();

System.*out*.println("The website title is \"" + websiteTitle + "\".");

// Note: Not all web browser tab titles work

if (Objects.*equals*(websiteTitle, "Latest | [MapleStorySEA] Unfunded Tips Amino")) {

System.*out*.println("The current web browser tab title is the correct " + "\"" + websiteTitle + "\"; and should be \"Latest | [MapleStorySEA] Unfunded Tips Amino\".");

extentTest.addScreenCaptureFromPath(capturedScreenshotImageFilepathString).pass(MediaEntityBuilder.*createScreenCaptureFromPath*(capturedScreenshotImageFilepathString).build()).log(Status.*PASS*, "The current web browser tab title is the correct " + "\"" + websiteTitle + "\"; and should be \"Latest | [MapleStorySEA] Unfunded Tips Amino\".");

} else {

System.*out*.println("The current web browser tab title is the incorrect " + "\"" + websiteTitle + "\"; and should be \"Latest | [MapleStorySEA] Unfunded Tips Amino\".");

extentTest.addScreenCaptureFromPath(capturedScreenshotImageFilepathString).pass(MediaEntityBuilder.*createScreenCaptureFromPath*(capturedScreenshotImageFilepathString).build()).log(Status.*FAIL*, "The current web browser tab title is the incorrect " + "\"" + websiteTitle + "\"; and should be \"Latest | [MapleStorySEA] Unfunded Tips Amino\".");

}

}

public static String takeScreenshot(WebDriver driver) throws IOException {

char dayFirstDigitChar = '0';

char daySecondDigitChar = '0';

char monthFirstDigitChar = '0';

char monthSecondDigitChar = '0';

char yearFirstDigitChar = '0';

char yearSecondDigitChar = '0';

char yearThirdDigitChar = '0';

char yearFourthDigitChar = '0';

char hourFirstDigitChar = '0';

char hourSecondDigitChar = '0';

char minuteFirstDigitChar = '0';

char minuteSecondDigitChar = '0';

char secondFirstDigitChar = '0';

char secondSecondDigitChar = '0';

char subsecondFirstDigitChar = '0';

char subsecondSecondDigitChar = '0';

char subsecondThirdDigitChar = '0';

char subsecondFourthDigitChar = '0';

char subsecondFifthDigitChar = '0';

char subsecondSixthDigitChar = '0';

// Issue - Missing these three digits

char subsecondSeventhDigitChar = '0';

char subsecondEighthDigitChar = '0';

char subsecondNinthDigitChar = '0';

LocalDateTime localDateTime = LocalDateTime.*now*();

String initialLocalDateTimeString = localDateTime.toString();

String initialLocalDateTimeStringDaySubstring = initialLocalDateTimeString.substring(8, 10);

String initialLocalDateTimeStringDashMonthDashSubstring = initialLocalDateTimeString.substring(4, 8);

String initialLocalDateTimeStringYearSubstring = initialLocalDateTimeString.substring(0, 4);

// Not providing as (10, 29) which caused an issue, as found out that intialLocalDateTimeString can be have 26, 27, 28, or 29 characters

String initialLocalDateTimeStringTimeSubString = initialLocalDateTimeString.substring(10);

String finalLocalDateTimeString = initialLocalDateTimeStringDaySubstring + initialLocalDateTimeStringDashMonthDashSubstring + initialLocalDateTimeStringYearSubstring + initialLocalDateTimeStringTimeSubString;

if(finalLocalDateTimeString.length()>=1) {

dayFirstDigitChar = finalLocalDateTimeString.charAt(0);

}

if(finalLocalDateTimeString.length()>=2) {

daySecondDigitChar = finalLocalDateTimeString.charAt(1);

}

if(finalLocalDateTimeString.length()>=4) {

monthFirstDigitChar = finalLocalDateTimeString.charAt(3);

}

if(finalLocalDateTimeString.length()>=5) {

monthSecondDigitChar = finalLocalDateTimeString.charAt(4);

}

if(finalLocalDateTimeString.length()>=7) {

yearFirstDigitChar = finalLocalDateTimeString.charAt(6);

}

if(finalLocalDateTimeString.length()>=8) {

yearSecondDigitChar = finalLocalDateTimeString.charAt(7);

}

if(finalLocalDateTimeString.length()>=9) {

yearThirdDigitChar = finalLocalDateTimeString.charAt(8);

}

if(finalLocalDateTimeString.length()>=10) {

yearFourthDigitChar = finalLocalDateTimeString.charAt(9);

}

if(finalLocalDateTimeString.length()>=12) {

hourFirstDigitChar = finalLocalDateTimeString.charAt(11);

}

if(finalLocalDateTimeString.length()>=13){

hourSecondDigitChar = finalLocalDateTimeString.charAt(12);

}

if(finalLocalDateTimeString.length()>=15) {

minuteFirstDigitChar = finalLocalDateTimeString.charAt(14);

}

if(finalLocalDateTimeString.length()>=16) {

minuteSecondDigitChar = finalLocalDateTimeString.charAt(15);

}

if(finalLocalDateTimeString.length()>=18) {

secondFirstDigitChar = finalLocalDateTimeString.charAt(17);

}

if(finalLocalDateTimeString.length()>=19) {

secondSecondDigitChar = finalLocalDateTimeString.charAt(18);

}

if(finalLocalDateTimeString.length()>=21) {

subsecondFirstDigitChar = finalLocalDateTimeString.charAt(20);

}

if(finalLocalDateTimeString.length()>=22) {

subsecondSecondDigitChar = finalLocalDateTimeString.charAt(21);

}

if(finalLocalDateTimeString.length()>=23) {

subsecondThirdDigitChar = finalLocalDateTimeString.charAt(22);

}

if(finalLocalDateTimeString.length()>=24) {

subsecondFourthDigitChar = finalLocalDateTimeString.charAt(23);

}

if(finalLocalDateTimeString.length()>=25) {

subsecondFifthDigitChar = finalLocalDateTimeString.charAt(24);

}

if(finalLocalDateTimeString.length()>=26) {

subsecondSixthDigitChar = finalLocalDateTimeString.charAt(25);

}

if(finalLocalDateTimeString.length()>=27) {

subsecondSeventhDigitChar = finalLocalDateTimeString.charAt(26);

}

if(finalLocalDateTimeString.length()>=28) {

subsecondEighthDigitChar = finalLocalDateTimeString.charAt(27);

}

if(finalLocalDateTimeString.length()>=29) {

subsecondNinthDigitChar = finalLocalDateTimeString.charAt(28);

}

TakesScreenshot screenshot = ((TakesScreenshot) driver);

File sourceFile = screenshot.getScreenshotAs(OutputType.*FILE*);

File destinationFile = new File("C:\\Users\\sohjnthn\\IdeaProjects\\Selenium\_Tutorial\\Selenium\_Screenshot\_Images\\" + dayFirstDigitChar + daySecondDigitChar + "-" + monthFirstDigitChar + monthSecondDigitChar + "-" + yearFirstDigitChar + yearSecondDigitChar + yearThirdDigitChar + yearFourthDigitChar + "T" + hourFirstDigitChar + hourSecondDigitChar + minuteFirstDigitChar + minuteSecondDigitChar + secondFirstDigitChar + secondSecondDigitChar + "." + subsecondFirstDigitChar + subsecondSecondDigitChar + subsecondThirdDigitChar + subsecondFourthDigitChar + subsecondFifthDigitChar + subsecondSixthDigitChar + subsecondSeventhDigitChar + subsecondEighthDigitChar + subsecondNinthDigitChar + "\_screenshot.png");

FileHandler.*copy*(sourceFile, destinationFile);

return "C:\\Users\\sohjnthn\\IdeaProjects\\Selenium\_Tutorial\\Selenium\_Screenshot\_Images\\" + dayFirstDigitChar + daySecondDigitChar + "-" + monthFirstDigitChar + monthSecondDigitChar + "-" + yearFirstDigitChar + yearSecondDigitChar + yearThirdDigitChar + yearFourthDigitChar + "T" + hourFirstDigitChar + hourSecondDigitChar + minuteFirstDigitChar + minuteSecondDigitChar + secondFirstDigitChar + secondSecondDigitChar + "." + subsecondFirstDigitChar + subsecondSecondDigitChar + subsecondThirdDigitChar + subsecondFourthDigitChar + subsecondFifthDigitChar + subsecondSixthDigitChar + subsecondSeventhDigitChar + subsecondEighthDigitChar + subsecondNinthDigitChar + "\_screenshot.png";

}

}

SecondGroupOfWebsites.java

package hellocucumber;

import com.aventstack.extentreports.ExtentReports;

import com.aventstack.extentreports.ExtentTest;

import com.aventstack.extentreports.MediaEntityBuilder;

import com.aventstack.extentreports.Status;

import com.aventstack.extentreports.reporter.ExtentSparkReporter;

import org.openqa.selenium.OutputType;

import org.openqa.selenium.TakesScreenshot;

import org.openqa.selenium.WebDriver;

import org.openqa.selenium.chrome.ChromeDriver;

import org.openqa.selenium.io.FileHandler;

import java.io.File;

import java.io.IOException;

import java.time.LocalDateTime;

public class SecondGroupOfWebsites {

private String websiteTitle = "";

private String capturedScreenshotImageFilepathString = "";

public SecondGroupOfWebsites (ChromeDriver driver, ExtentTest extentTest, ExtentSparkReporter extentSparkReporter, ExtentReports extentReport) throws IOException {

// For displaying a sentence for ExportReport

driver.get("https://aminoapps.com/c/maplestorysea/recent/");

capturedScreenshotImageFilepathString = *takeScreenshot*(driver);

extentTest.addScreenCaptureFromPath(capturedScreenshotImageFilepathString).pass(MediaEntityBuilder.*createScreenCaptureFromPath*(capturedScreenshotImageFilepathString).build()).log(Status.*PASS*, "When it is not possible to retrieve WebElements using name/id/className/partialLinkText.");

System.*out*.println("It is not possible to do so for certain WebElements in this way.");

}

public static String takeScreenshot(WebDriver driver) throws IOException {

char dayFirstDigitChar = '0';

char daySecondDigitChar = '0';

char monthFirstDigitChar = '0';

char monthSecondDigitChar = '0';

char yearFirstDigitChar = '0';

char yearSecondDigitChar = '0';

char yearThirdDigitChar = '0';

char yearFourthDigitChar = '0';

char hourFirstDigitChar = '0';

char hourSecondDigitChar = '0';

char minuteFirstDigitChar = '0';

char minuteSecondDigitChar = '0';

char secondFirstDigitChar = '0';

char secondSecondDigitChar = '0';

char subsecondFirstDigitChar = '0';

char subsecondSecondDigitChar = '0';

char subsecondThirdDigitChar = '0';

char subsecondFourthDigitChar = '0';

char subsecondFifthDigitChar = '0';

char subsecondSixthDigitChar = '0';

// Issue - Missing these three digits

char subsecondSeventhDigitChar = '0';

char subsecondEighthDigitChar = '0';

char subsecondNinthDigitChar = '0';

LocalDateTime localDateTime = LocalDateTime.*now*();

String initialLocalDateTimeString = localDateTime.toString();

String initialLocalDateTimeStringDaySubstring = initialLocalDateTimeString.substring(8, 10);

String initialLocalDateTimeStringDashMonthDashSubstring = initialLocalDateTimeString.substring(4, 8);

String initialLocalDateTimeStringYearSubstring = initialLocalDateTimeString.substring(0, 4);

// Not providing as (10, 29) which caused an issue, as found out that intialLocalDateTimeString can be have 26, 27, 28, or 29 characters

String initialLocalDateTimeStringTimeSubString = initialLocalDateTimeString.substring(10);

String finalLocalDateTimeString = initialLocalDateTimeStringDaySubstring + initialLocalDateTimeStringDashMonthDashSubstring + initialLocalDateTimeStringYearSubstring + initialLocalDateTimeStringTimeSubString;

if(finalLocalDateTimeString.length()>=1) {

dayFirstDigitChar = finalLocalDateTimeString.charAt(0);

}

if(finalLocalDateTimeString.length()>=2) {

daySecondDigitChar = finalLocalDateTimeString.charAt(1);

}

if(finalLocalDateTimeString.length()>=4) {

monthFirstDigitChar = finalLocalDateTimeString.charAt(3);

}

if(finalLocalDateTimeString.length()>=5) {

monthSecondDigitChar = finalLocalDateTimeString.charAt(4);

}

if(finalLocalDateTimeString.length()>=7) {

yearFirstDigitChar = finalLocalDateTimeString.charAt(6);

}

if(finalLocalDateTimeString.length()>=8) {

yearSecondDigitChar = finalLocalDateTimeString.charAt(7);

}

if(finalLocalDateTimeString.length()>=9) {

yearThirdDigitChar = finalLocalDateTimeString.charAt(8);

}

if(finalLocalDateTimeString.length()>=10) {

yearFourthDigitChar = finalLocalDateTimeString.charAt(9);

}

if(finalLocalDateTimeString.length()>=12) {

hourFirstDigitChar = finalLocalDateTimeString.charAt(11);

}

if(finalLocalDateTimeString.length()>=13){

hourSecondDigitChar = finalLocalDateTimeString.charAt(12);

}

if(finalLocalDateTimeString.length()>=15) {

minuteFirstDigitChar = finalLocalDateTimeString.charAt(14);

}

if(finalLocalDateTimeString.length()>=16) {

minuteSecondDigitChar = finalLocalDateTimeString.charAt(15);

}

if(finalLocalDateTimeString.length()>=18) {

secondFirstDigitChar = finalLocalDateTimeString.charAt(17);

}

if(finalLocalDateTimeString.length()>=19) {

secondSecondDigitChar = finalLocalDateTimeString.charAt(18);

}

if(finalLocalDateTimeString.length()>=21) {

subsecondFirstDigitChar = finalLocalDateTimeString.charAt(20);

}

if(finalLocalDateTimeString.length()>=22) {

subsecondSecondDigitChar = finalLocalDateTimeString.charAt(21);

}

if(finalLocalDateTimeString.length()>=23) {

subsecondThirdDigitChar = finalLocalDateTimeString.charAt(22);

}

if(finalLocalDateTimeString.length()>=24) {

subsecondFourthDigitChar = finalLocalDateTimeString.charAt(23);

}

if(finalLocalDateTimeString.length()>=25) {

subsecondFifthDigitChar = finalLocalDateTimeString.charAt(24);

}

if(finalLocalDateTimeString.length()>=26) {

subsecondSixthDigitChar = finalLocalDateTimeString.charAt(25);

}

if(finalLocalDateTimeString.length()>=27) {

subsecondSeventhDigitChar = finalLocalDateTimeString.charAt(26);

}

if(finalLocalDateTimeString.length()>=28) {

subsecondEighthDigitChar = finalLocalDateTimeString.charAt(27);

}

if(finalLocalDateTimeString.length()>=29) {

subsecondNinthDigitChar = finalLocalDateTimeString.charAt(28);

}

TakesScreenshot screenshot = ((TakesScreenshot) driver);

File sourceFile = screenshot.getScreenshotAs(OutputType.*FILE*);

File destinationFile = new File("C:\\Users\\sohjnthn\\IdeaProjects\\Selenium\_Tutorial\\Selenium\_Screenshot\_Images\\" + dayFirstDigitChar + daySecondDigitChar + "-" + monthFirstDigitChar + monthSecondDigitChar + "-" + yearFirstDigitChar + yearSecondDigitChar + yearThirdDigitChar + yearFourthDigitChar + "T" + hourFirstDigitChar + hourSecondDigitChar + minuteFirstDigitChar + minuteSecondDigitChar + secondFirstDigitChar + secondSecondDigitChar + "." + subsecondFirstDigitChar + subsecondSecondDigitChar + subsecondThirdDigitChar + subsecondFourthDigitChar + subsecondFifthDigitChar + subsecondSixthDigitChar + subsecondSeventhDigitChar + subsecondEighthDigitChar + subsecondNinthDigitChar + "\_screenshot.png");

FileHandler.*copy*(sourceFile, destinationFile);

return "C:\\Users\\sohjnthn\\IdeaProjects\\Selenium\_Tutorial\\Selenium\_Screenshot\_Images\\" + dayFirstDigitChar + daySecondDigitChar + "-" + monthFirstDigitChar + monthSecondDigitChar + "-" + yearFirstDigitChar + yearSecondDigitChar + yearThirdDigitChar + yearFourthDigitChar + "T" + hourFirstDigitChar + hourSecondDigitChar + minuteFirstDigitChar + minuteSecondDigitChar + secondFirstDigitChar + secondSecondDigitChar + "." + subsecondFirstDigitChar + subsecondSecondDigitChar + subsecondThirdDigitChar + subsecondFourthDigitChar + subsecondFifthDigitChar + subsecondSixthDigitChar + subsecondSeventhDigitChar + subsecondEighthDigitChar + subsecondNinthDigitChar + "\_screenshot.png";

}

}

ThirdGroupOfWebsites.java

package hellocucumber;

import com.aventstack.extentreports.ExtentReports;

import com.aventstack.extentreports.ExtentTest;

import com.aventstack.extentreports.MediaEntityBuilder;

import com.aventstack.extentreports.Status;

import com.aventstack.extentreports.reporter.ExtentSparkReporter;

import org.openqa.selenium.\*;

import org.openqa.selenium.chrome.ChromeDriver;

import org.openqa.selenium.io.FileHandler;

import java.io.File;

import java.io.IOException;

import java.time.LocalDateTime;

import java.util.List;

import java.util.Objects;

public class ThirdGroupOfWebsites {

private String websiteTitle = "";

private String capturedScreenshotImageFilepathString = "";

public ThirdGroupOfWebsites (ChromeDriver driver, ExtentTest extentTest, ExtentSparkReporter extentSparkReporter, ExtentReports extentReport) throws IOException {

// Therefore, the Cucumber Feature version requires driver4 to be initialised for the Then method

driver.get("https://aminoapps.com/c/maplestorysea/recent/");

String oneString = "1";

// i is a valid tagName, which is displayed starting with "<i "

List<WebElement> tagNameIWebElementList = driver.findElements(By.*tagName*("i"));

WebElement aminoHomeIcon = tagNameIWebElementList.get(2);

if (aminoHomeIcon.isDisplayed()) {

aminoHomeIcon.click();

}

websiteTitle = driver.getTitle();

System.*out*.println("The website title is \"" + websiteTitle + "\".");

// Note: Not all web browser tab titles work

if (Objects.*equals*(websiteTitle, "Featured | [MapleStorySEA] Unfunded Tips Amino")) {

System.*out*.println("The current web browser tab title is the correct \"" + websiteTitle + "\"; and should be the correct \"Featured | [MapleStorySEA] Unfunded Tips Amino\".");

} else {

System.*out*.println("The current web browser tab title is the incorrect \"" + websiteTitle + "\"; and should be the correct \"Featured | [MapleStorySEA] Unfunded Tips Amino\".");

}

// button is a valid tagName, which is displayed starting with "<button "

// testing for the Create Post button - pop-up

List<WebElement> tagNameButtonWebElementList = driver.findElements(By.*tagName*("button"));

WebElement createPostButton = tagNameButtonWebElementList.get(1);

if(createPostButton.isDisplayed()) {

createPostButton.click();

// Captured screenshot image filename to be used for ExtentReport test step - Pass; or Fail

capturedScreenshotImageFilepathString = *takeScreenshot*(driver);

//actual pop-up cannot be captured, can only prove that the pop-up was triggered by clicking on the Create Post button

extentTest.addScreenCaptureFromPath(capturedScreenshotImageFilepathString).pass(MediaEntityBuilder.*createScreenCaptureFromPath*(capturedScreenshotImageFilepathString).build()).log(Status.*PASS*, "The Amino webpage's Create Post pop-up is displayed.");

System.*out*.println("Create Post button was clicked on to trigger the pop-up. The pop-up is not displayed at this point in time, but is displayed later.");

} else {

// Captured screenshot image filename to be used for ExtentReport test step - Pass; or Fail

capturedScreenshotImageFilepathString = *takeScreenshot*(driver);

extentTest.addScreenCaptureFromPath(capturedScreenshotImageFilepathString).pass(MediaEntityBuilder.*createScreenCaptureFromPath*(capturedScreenshotImageFilepathString).build()).log(Status.*FAIL*, "The Amino webpage's Create Post button was not clicked on, so the Create Post pop-up is not displayed later.");

}

// trying to capture pop-up UI in ExtentReport

// So not possible to provide not working if else - else

// Captured screenshot image filename to be used for ExtentReport test step - Pass; or Fail

capturedScreenshotImageFilepathString = *takeScreenshot*(driver);

extentTest.addScreenCaptureFromPath(capturedScreenshotImageFilepathString).pass(MediaEntityBuilder.*createScreenCaptureFromPath*(capturedScreenshotImageFilepathString).build()).log(Status.*PASS*, "The Amino webpage's Create Post pop-up is displayed now.");

// Test for whether non-text-input field text can be retrieved

WebElement tagLine = driver.findElement(By.*className*("tagline"));

String taglineText = tagLine.getText();

// Captured screenshot image filename to be used for ExtentReport test step - Pass; or Fail

capturedScreenshotImageFilepathString = *takeScreenshot*(driver);

if(taglineText.equals("MapleStorySEA follows the original Korea MapleStory (by Nexon Korea) 's content.")){

extentTest.addScreenCaptureFromPath(capturedScreenshotImageFilepathString).pass(MediaEntityBuilder.*createScreenCaptureFromPath*(capturedScreenshotImageFilepathString).build()).log(Status.*PASS*, "The Amino webpage's tagline text is displayed as the correct \"" + taglineText + "\"; which should be \"MapleStorySEA follows the original Korea MapleStory (by Nexon Korea) 's content.\".");

} else {

extentTest.addScreenCaptureFromPath(capturedScreenshotImageFilepathString).pass(MediaEntityBuilder.*createScreenCaptureFromPath*(capturedScreenshotImageFilepathString).build()).log(Status.*FAIL*, "The Amino webpage's tagline text is displayed as the incorrect \"" + taglineText + "\"; which should be \"MapleStorySEA follows the original Korea MapleStory (by Nexon Korea) 's content.\".");

}

// Testing if "Explore your interests and " for the pop-up can be accessed

// tagName is "p"

//List <WebElement> tagNamePWebElementList = driver.findElements(By.tagName("p"));

//WebElement exploreYourInterestsAndFindYourCommunitiesFullstopText = tagNamePWebElementList.get(1);

//String exploreYourInterestsAndFindYourCommunitiesFullstopTextString = exploreYourInterestsAndFindYourCommunitiesFullstopText.getText();

// Note: Even though .getText() works, which seems to provide the same text

// This has been verified by the system to not match

// So this has to be tested by checking for whether the WebElement object is displayed

// Explore your interests and find your communities.

List<WebElement> imgTagList = driver.findElements(By.*tagName*("img"));

WebElement topLeftCornerAminoLogoButton = imgTagList.get(0);

// There is some delay, so we need to use the WebElement which can be found

// before and after the pop-up is displayed

for(int i=0; i<1000; i++) {

System.*out*.println("The Amino webpage's tagline text is displayed as the correct \"Explore your interests and find your communities.\"; which should be \"Explore your interests and find your communities.\", the pop-up UI would take time to load.");

}

// The following element cannot be found with a for loop for 1000 times for a single System.out.println statement

// Then the required text of the pop-up can be found

//List <WebElement> tagNamePWebElementList = driver.findElements(By.tagName("p"));

//WebElement exploreYourInterestsAndFindYourCommunitiesFullstopText = tagNamePWebElementList.get(1);

//String exploreYourInterestsAndFindYourCommunitiesFullstopTextString = exploreYourInterestsAndFindYourCommunitiesFullstopText.getText();

capturedScreenshotImageFilepathString = *takeScreenshot*(driver);

if(topLeftCornerAminoLogoButton.isDisplayed()){

extentTest.addScreenCaptureFromPath(capturedScreenshotImageFilepathString).pass(MediaEntityBuilder.*createScreenCaptureFromPath*(capturedScreenshotImageFilepathString).build()).log(Status.*PASS*, "The pop-up's text is correct; and should be \"Explore your interests and find your communities.\".");

} else {

extentTest.addScreenCaptureFromPath(capturedScreenshotImageFilepathString).pass(MediaEntityBuilder.*createScreenCaptureFromPath*(capturedScreenshotImageFilepathString).build()).log(Status.*FAIL*, "The pop-up's text is incorrect; and should be \"Explore your interests and find your communities.\".");

}

extentReport.flush();

// Note: Unable to verify any element of pop-up

// Number of valid tagName elements end as per last related non-pop-up WebElement

}

public static String takeScreenshot(WebDriver driver) throws IOException {

char dayFirstDigitChar = '0';

char daySecondDigitChar = '0';

char monthFirstDigitChar = '0';

char monthSecondDigitChar = '0';

char yearFirstDigitChar = '0';

char yearSecondDigitChar = '0';

char yearThirdDigitChar = '0';

char yearFourthDigitChar = '0';

char hourFirstDigitChar = '0';

char hourSecondDigitChar = '0';

char minuteFirstDigitChar = '0';

char minuteSecondDigitChar = '0';

char secondFirstDigitChar = '0';

char secondSecondDigitChar = '0';

char subsecondFirstDigitChar = '0';

char subsecondSecondDigitChar = '0';

char subsecondThirdDigitChar = '0';

char subsecondFourthDigitChar = '0';

char subsecondFifthDigitChar = '0';

char subsecondSixthDigitChar = '0';

// Issue - Missing these three digits

char subsecondSeventhDigitChar = '0';

char subsecondEighthDigitChar = '0';

char subsecondNinthDigitChar = '0';

LocalDateTime localDateTime = LocalDateTime.*now*();

String initialLocalDateTimeString = localDateTime.toString();

String initialLocalDateTimeStringDaySubstring = initialLocalDateTimeString.substring(8, 10);

String initialLocalDateTimeStringDashMonthDashSubstring = initialLocalDateTimeString.substring(4, 8);

String initialLocalDateTimeStringYearSubstring = initialLocalDateTimeString.substring(0, 4);

// Not providing as (10, 29) which caused an issue, as found out that intialLocalDateTimeString can be have 26, 27, 28, or 29 characters

String initialLocalDateTimeStringTimeSubString = initialLocalDateTimeString.substring(10);

String finalLocalDateTimeString = initialLocalDateTimeStringDaySubstring + initialLocalDateTimeStringDashMonthDashSubstring + initialLocalDateTimeStringYearSubstring + initialLocalDateTimeStringTimeSubString;

if(finalLocalDateTimeString.length()>=1) {

dayFirstDigitChar = finalLocalDateTimeString.charAt(0);

}

if(finalLocalDateTimeString.length()>=2) {

daySecondDigitChar = finalLocalDateTimeString.charAt(1);

}

if(finalLocalDateTimeString.length()>=4) {

monthFirstDigitChar = finalLocalDateTimeString.charAt(3);

}

if(finalLocalDateTimeString.length()>=5) {

monthSecondDigitChar = finalLocalDateTimeString.charAt(4);

}

if(finalLocalDateTimeString.length()>=7) {

yearFirstDigitChar = finalLocalDateTimeString.charAt(6);

}

if(finalLocalDateTimeString.length()>=8) {

yearSecondDigitChar = finalLocalDateTimeString.charAt(7);

}

if(finalLocalDateTimeString.length()>=9) {

yearThirdDigitChar = finalLocalDateTimeString.charAt(8);

}

if(finalLocalDateTimeString.length()>=10) {

yearFourthDigitChar = finalLocalDateTimeString.charAt(9);

}

if(finalLocalDateTimeString.length()>=12) {

hourFirstDigitChar = finalLocalDateTimeString.charAt(11);

}

if(finalLocalDateTimeString.length()>=13){

hourSecondDigitChar = finalLocalDateTimeString.charAt(12);

}

if(finalLocalDateTimeString.length()>=15) {

minuteFirstDigitChar = finalLocalDateTimeString.charAt(14);

}

if(finalLocalDateTimeString.length()>=16) {

minuteSecondDigitChar = finalLocalDateTimeString.charAt(15);

}

if(finalLocalDateTimeString.length()>=18) {

secondFirstDigitChar = finalLocalDateTimeString.charAt(17);

}

if(finalLocalDateTimeString.length()>=19) {

secondSecondDigitChar = finalLocalDateTimeString.charAt(18);

}

if(finalLocalDateTimeString.length()>=21) {

subsecondFirstDigitChar = finalLocalDateTimeString.charAt(20);

}

if(finalLocalDateTimeString.length()>=22) {

subsecondSecondDigitChar = finalLocalDateTimeString.charAt(21);

}

if(finalLocalDateTimeString.length()>=23) {

subsecondThirdDigitChar = finalLocalDateTimeString.charAt(22);

}

if(finalLocalDateTimeString.length()>=24) {

subsecondFourthDigitChar = finalLocalDateTimeString.charAt(23);

}

if(finalLocalDateTimeString.length()>=25) {

subsecondFifthDigitChar = finalLocalDateTimeString.charAt(24);

}

if(finalLocalDateTimeString.length()>=26) {

subsecondSixthDigitChar = finalLocalDateTimeString.charAt(25);

}

if(finalLocalDateTimeString.length()>=27) {

subsecondSeventhDigitChar = finalLocalDateTimeString.charAt(26);

}

if(finalLocalDateTimeString.length()>=28) {

subsecondEighthDigitChar = finalLocalDateTimeString.charAt(27);

}

if(finalLocalDateTimeString.length()>=29) {

subsecondNinthDigitChar = finalLocalDateTimeString.charAt(28);

}

TakesScreenshot screenshot = ((TakesScreenshot) driver);

File sourceFile = screenshot.getScreenshotAs(OutputType.*FILE*);

File destinationFile = new File("C:\\Users\\sohjnthn\\IdeaProjects\\Selenium\_Tutorial\\Selenium\_Screenshot\_Images\\" + dayFirstDigitChar + daySecondDigitChar + "-" + monthFirstDigitChar + monthSecondDigitChar + "-" + yearFirstDigitChar + yearSecondDigitChar + yearThirdDigitChar + yearFourthDigitChar + "T" + hourFirstDigitChar + hourSecondDigitChar + minuteFirstDigitChar + minuteSecondDigitChar + secondFirstDigitChar + secondSecondDigitChar + "." + subsecondFirstDigitChar + subsecondSecondDigitChar + subsecondThirdDigitChar + subsecondFourthDigitChar + subsecondFifthDigitChar + subsecondSixthDigitChar + subsecondSeventhDigitChar + subsecondEighthDigitChar + subsecondNinthDigitChar + "\_screenshot.png");

FileHandler.*copy*(sourceFile, destinationFile);

return "C:\\Users\\sohjnthn\\IdeaProjects\\Selenium\_Tutorial\\Selenium\_Screenshot\_Images\\" + dayFirstDigitChar + daySecondDigitChar + "-" + monthFirstDigitChar + monthSecondDigitChar + "-" + yearFirstDigitChar + yearSecondDigitChar + yearThirdDigitChar + yearFourthDigitChar + "T" + hourFirstDigitChar + hourSecondDigitChar + minuteFirstDigitChar + minuteSecondDigitChar + secondFirstDigitChar + secondSecondDigitChar + "." + subsecondFirstDigitChar + subsecondSecondDigitChar + subsecondThirdDigitChar + subsecondFourthDigitChar + subsecondFifthDigitChar + subsecondSixthDigitChar + subsecondSeventhDigitChar + subsecondEighthDigitChar + subsecondNinthDigitChar + "\_screenshot.png";

}

}

TestAndroidApps.java

package hellocucumber;

import com.aventstack.extentreports.ExtentReports;

import com.aventstack.extentreports.ExtentTest;

import com.aventstack.extentreports.reporter.ExtentSparkReporter;

import io.appium.java\_client.android.AndroidDriver;

import io.cucumber.java.en.Given;

import io.cucumber.java.en.Then;

import io.cucumber.java.en.When;

import io.cucumber.junit.Cucumber;

import io.cucumber.junit.CucumberOptions;

import org.junit.runner.RunWith;

import org.openqa.selenium.remote.DesiredCapabilities;

import java.io.IOException;

import java.net.URL;

import java.time.LocalDateTime;

//TIP To <b>Run</b> code, press <shortcut actionId="Run"/> or

// click the <icon src="AllIcons.Actions.Execute"/> icon in the gutter.

//@CucumberContextConfiguration

@RunWith(Cucumber.class)

@CucumberOptions(

// features should start with "src/test/resources/features/xxxxxxxxxxx.feature",

features = "src/test/resources/features/Selenium\_Android.feature",

// glue starts with "src/test/java/#package for the Test Java class with the Given;When;Then annotations#/",

glue={"src/test/java/hellocucumber/"},

// plugin = = {"pretty", "html:target/cucumber-reports"}

plugin = {"pretty", "html:target/cucumber-reports"}

)

public class TestAndroidApps {

// Test.java class should be within src > test

AndroidDriver driver1;

AndroidDriver driver2;

AndroidDriver driver3;

AndroidDriver driver4;

String websiteTitle = "";

String capturedScreenshotImageFilepathString = "";

String currentLocalDateTimeForExtentSparkReporter = *currentLocalDateTimeWithDdMmYyFormat*();

ExtentReports extentReport = new ExtentReports();

// should be target/Spark.html, not an actual filepath starting from C:\\

ExtentSparkReporter extentSparkReporter = new ExtentSparkReporter("target/Spark\_" + currentLocalDateTimeForExtentSparkReporter + ".html");

ExtentTest extentTest = extentReport.createTest("Selenium\_Android Test");

FirstAppObjects firstAppObjects;

SecondAppObjects secondAppObjects;

ThirdAppObjects thirdAppObjects;

public TestAndroidApps() throws IOException {

}

// Note that this method is not executed, if running the feature with the Given, When, Then lines

void testMethod() {

System.*out*.println("testing the IntelliJ IDEA - Help - Testing - Create Tests - Right-click to generate Test Method for JUnit5.");

}

@org.junit.Test

@org.junit.jupiter.api.Test

@org.testng.annotations.Test

@Given("I wish to learn how to automate a smartphone application which can be used for a real Android 15 device.")

public void testMethod1() throws IOException {

// this line only works without an error, in a public method, as it is a public void method

// the other three lines (originally above this fourth line) are public methods

extentReport.attachReporter(extentSparkReporter);

DesiredCapabilities desiredCapabilities = new DesiredCapabilities();

desiredCapabilities.setCapability("platformName", "Android");

desiredCapabilities.setCapability("appium:platformVersion", "15.0");

desiredCapabilities.setCapability("appium:app", "C:\\Users\\sohjnthn\\IdeaProjects\\Selenium\_Tutorial\\app\\com.afwsamples.testdpc\_9.0.12-9012\_minAPI21(nodpi)\_apkmirror.com.apk");

desiredCapabilities.setCapability("appium:automationName", "UiAutomator2");

driver1 = new AndroidDriver(new URL("http://127.0.0.1:4723/wd/hub"), desiredCapabilities);

firstAppObjects = new FirstAppObjects(driver1, extentTest, extentSparkReporter, extentReport);

}

@org.junit.Test

@org.junit.jupiter.api.Test

@org.testng.annotations.Test

@When("I provide the Given and When methods with a new object of the first non-main class.")

public void testMethod2() throws IOException {

firstAppObjects = new FirstAppObjects(driver1, extentTest, extentSparkReporter, extentReport);

}

@org.junit.Test

@org.junit.jupiter.api.Test

@org.testng.annotations.Test

@Then("I provide the Then method with the new objects of the second and third non-main classes.")

public void testMethod3() throws IOException {

DesiredCapabilities desiredCapabilities = new DesiredCapabilities();

desiredCapabilities.setCapability("platformName", "Android");

desiredCapabilities.setCapability("appium:platformVersion", "15.0");

desiredCapabilities.setCapability("appium:app", "C:\\Users\\sohjnthn\\IdeaProjects\\Selenium\_Tutorial\\app\\ApiDemos-debug.apk");

desiredCapabilities.setCapability("appium:automationName", "UiAutomator2");

driver2 = new AndroidDriver(new URL("http://127.0.0.1:4723/wd/hub"), desiredCapabilities);

secondAppObjects = new SecondAppObjects(driver2, extentTest, extentSparkReporter, extentReport);

desiredCapabilities = new DesiredCapabilities();

desiredCapabilities.setCapability("platformName", "Android");

desiredCapabilities.setCapability("appium:platformVersion", "15.0");

desiredCapabilities.setCapability("appium:app", "C:\\Users\\sohjnthn\\IdeaProjects\\Selenium\_Tutorial\\app\\com.microsoft.emmx.canary\_138.0.3336.0-333600000\_minAPI26(armeabi-v7a)(nodpi)\_apkmirror.com.apk");

desiredCapabilities.setCapability("appium:automationName", "UiAutomator2");

driver3 = new AndroidDriver(new URL("http://127.0.0.1:4723/wd/hub"), desiredCapabilities);

thirdAppObjects = new ThirdAppObjects(driver3, extentTest, extentSparkReporter, extentReport);

}

public static String currentLocalDateTimeWithDdMmYyFormat() throws IOException {

char dayFirstDigitChar = '0';

char daySecondDigitChar = '0';

char monthFirstDigitChar = '0';

char monthSecondDigitChar = '0';

char yearFirstDigitChar = '0';

char yearSecondDigitChar = '0';

char yearThirdDigitChar = '0';

char yearFourthDigitChar = '0';

char hourFirstDigitChar = '0';

char hourSecondDigitChar = '0';

char minuteFirstDigitChar = '0';

char minuteSecondDigitChar = '0';

char secondFirstDigitChar = '0';

char secondSecondDigitChar = '0';

char subsecondFirstDigitChar = '0';

char subsecondSecondDigitChar = '0';

char subsecondThirdDigitChar = '0';

char subsecondFourthDigitChar = '0';

char subsecondFifthDigitChar = '0';

char subsecondSixthDigitChar = '0';

// Issue - Missing these three digits

char subsecondSeventhDigitChar = '0';

char subsecondEighthDigitChar = '0';

char subsecondNinthDigitChar = '0';

// How to take Selenium screenshot image

LocalDateTime localDateTime = LocalDateTime.*now*();

String initialLocalDateTimeString = localDateTime.toString();

String initialLocalDateTimeStringDaySubstring = initialLocalDateTimeString.substring(8, 10);

String initialLocalDateTimeStringDashMonthDashSubstring = initialLocalDateTimeString.substring(4, 8);

String initialLocalDateTimeStringYearSubstring = initialLocalDateTimeString.substring(0, 4);

String initialLocalDateTimeStringTimeSubString = initialLocalDateTimeString.substring(10);

// Not providing as (10, 29) which caused an issue, as found out that intialLocalDateTimeString can be have 26, 27, 28, or 29 characters

String finalLocalDateTimeString = initialLocalDateTimeStringDaySubstring + initialLocalDateTimeStringDashMonthDashSubstring + initialLocalDateTimeStringYearSubstring + initialLocalDateTimeStringTimeSubString;

if(finalLocalDateTimeString.length()>=1) {

// index 0 character; 1st character

dayFirstDigitChar = finalLocalDateTimeString.charAt(0);

}

if(finalLocalDateTimeString.length()>=2) {

// index 1 character; 2nd character

daySecondDigitChar = finalLocalDateTimeString.charAt(1);

}

if(finalLocalDateTimeString.length()>=4) {

// index 3 character; 4th character

monthFirstDigitChar = finalLocalDateTimeString.charAt(3);

}

if(finalLocalDateTimeString.length()>=5) {

// index 4 character; 5th character

monthSecondDigitChar = finalLocalDateTimeString.charAt(4);

}

if(finalLocalDateTimeString.length()>=7) {

// index 6 character; 7th character

yearFirstDigitChar = finalLocalDateTimeString.charAt(6);

}

if(finalLocalDateTimeString.length()>=8) {

// index 7 character; 8th character

yearSecondDigitChar = finalLocalDateTimeString.charAt(7);

}

if(finalLocalDateTimeString.length()>=9) {

// index 8 character; 9th character

yearThirdDigitChar = finalLocalDateTimeString.charAt(8);

}

if(finalLocalDateTimeString.length()>=10) {

// index 9 character; 10th character

yearFourthDigitChar = finalLocalDateTimeString.charAt(9);

}

if(finalLocalDateTimeString.length()>=12) {

// index 11 character; 12th character

hourFirstDigitChar = finalLocalDateTimeString.charAt(11);

}

if(finalLocalDateTimeString.length()>=13){

// index 12 character; 13th character

hourSecondDigitChar = finalLocalDateTimeString.charAt(12);

}

if(finalLocalDateTimeString.length()>=15) {

// index 14 character; 15th character

minuteFirstDigitChar = finalLocalDateTimeString.charAt(14);

}

if(finalLocalDateTimeString.length()>=16) {

// index 15 character; 16th character

minuteSecondDigitChar = finalLocalDateTimeString.charAt(15);

}

if(finalLocalDateTimeString.length()>=18) {

// index 17 character; 18th character

secondFirstDigitChar = finalLocalDateTimeString.charAt(17);

}

if(finalLocalDateTimeString.length()>=19) {

// index 18 character; 19th character

secondSecondDigitChar = finalLocalDateTimeString.charAt(18);

}

if(finalLocalDateTimeString.length()>=21) {

// index 20 character; 21th character

subsecondFirstDigitChar = finalLocalDateTimeString.charAt(20);

}

if(finalLocalDateTimeString.length()>=22) {

// index 21 character; 22nd character

subsecondSecondDigitChar = finalLocalDateTimeString.charAt(21);

}

if(finalLocalDateTimeString.length()>=23) {

// index 22 character; 23rd character

subsecondThirdDigitChar = finalLocalDateTimeString.charAt(22);

}

if(finalLocalDateTimeString.length()>=24) {

// index 23 character; 24th character

subsecondFourthDigitChar = finalLocalDateTimeString.charAt(23);

}

if(finalLocalDateTimeString.length()>=25) {

// index 24 character; 25th character

subsecondFifthDigitChar = finalLocalDateTimeString.charAt(24);

}

if(finalLocalDateTimeString.length()>=26) {

// index 25 character; 26th character

subsecondSixthDigitChar = finalLocalDateTimeString.charAt(25);

}

if(finalLocalDateTimeString.length()>=27) {

// index 26 character; 27th character

subsecondSeventhDigitChar = finalLocalDateTimeString.charAt(26);

}

if(finalLocalDateTimeString.length()>=28) {

// index 27 character; 28th character

subsecondEighthDigitChar = finalLocalDateTimeString.charAt(27);

}

if(finalLocalDateTimeString.length()>=29) {

// index 28 character; 29th character

subsecondNinthDigitChar = finalLocalDateTimeString.charAt(28);

}

// There seems to be an error for which the two digits for day are incorrect, so localDateTime.getDayOfmonth is used(), instead of dayFirstDigitChar and daySecondDigitChar

return localDateTime.getDayOfMonth() + "-" + monthFirstDigitChar + monthSecondDigitChar + "-" + yearFirstDigitChar + yearSecondDigitChar + yearThirdDigitChar + yearFourthDigitChar + "T" + hourFirstDigitChar + hourSecondDigitChar + minuteFirstDigitChar + minuteSecondDigitChar + secondFirstDigitChar + secondSecondDigitChar + "." + subsecondFirstDigitChar + subsecondSecondDigitChar + subsecondThirdDigitChar + subsecondFourthDigitChar + subsecondFifthDigitChar + subsecondSixthDigitChar + subsecondSeventhDigitChar + subsecondEighthDigitChar + subsecondNinthDigitChar;

}

}

FirstAppObjects.java

package hellocucumber;

import com.aventstack.extentreports.ExtentReports;

import com.aventstack.extentreports.ExtentTest;

import com.aventstack.extentreports.MediaEntityBuilder;

import com.aventstack.extentreports.Status;

import com.aventstack.extentreports.reporter.ExtentSparkReporter;

import io.appium.java\_client.android.AndroidDriver;

import io.appium.java\_client.pagefactory.AndroidFindBy;

import org.openqa.selenium.\*;

import org.openqa.selenium.io.FileHandler;

import java.io.File;

import java.io.IOException;

import java.time.LocalDateTime;

public class FirstAppObjects {

// First App

@AndroidFindBy(xpath = "//android.widget.TextView[@text=\"Policy management\"]")

WebElement policyManagementText;

AndroidDriver driver;

String capturedScreenshotImageFilepathString = "";

public FirstAppObjects(AndroidDriver driver, ExtentTest extentTest, ExtentSparkReporter extentSparkReporter, ExtentReports extentReport) throws IOException {

policyManagementText = driver.findElement(By.*xpath*("//android.widget.TextView[@text=\"Policy management\"]"));

if(policyManagementText.isDisplayed()){

capturedScreenshotImageFilepathString = *takeScreenshot*(driver);

extentTest.addScreenCaptureFromPath(capturedScreenshotImageFilepathString).pass(MediaEntityBuilder.*createScreenCaptureFromPath*(capturedScreenshotImageFilepathString).build()).log(Status.*PASS*, "Able to find \"Policy Management\".");

System.*out*.println("Pass");

} else {

capturedScreenshotImageFilepathString = *takeScreenshot*(driver);

extentTest.addScreenCaptureFromPath(capturedScreenshotImageFilepathString).pass(MediaEntityBuilder.*createScreenCaptureFromPath*(capturedScreenshotImageFilepathString).build()).log(Status.*FAIL*, "Unable to find \"Policy Management\".");

System.*out*.println("Fail");

}

}

public static String takeScreenshot(WebDriver driver) throws IOException {

char dayFirstDigitChar = '0';

char daySecondDigitChar = '0';

char monthFirstDigitChar = '0';

char monthSecondDigitChar = '0';

char yearFirstDigitChar = '0';

char yearSecondDigitChar = '0';

char yearThirdDigitChar = '0';

char yearFourthDigitChar = '0';

char hourFirstDigitChar = '0';

char hourSecondDigitChar = '0';

char minuteFirstDigitChar = '0';

char minuteSecondDigitChar = '0';

char secondFirstDigitChar = '0';

char secondSecondDigitChar = '0';

char subsecondFirstDigitChar = '0';

char subsecondSecondDigitChar = '0';

char subsecondThirdDigitChar = '0';

char subsecondFourthDigitChar = '0';

char subsecondFifthDigitChar = '0';

char subsecondSixthDigitChar = '0';

// Issue - Missing these three digits

char subsecondSeventhDigitChar = '0';

char subsecondEighthDigitChar = '0';

char subsecondNinthDigitChar = '0';

LocalDateTime localDateTime = LocalDateTime.*now*();

String initialLocalDateTimeString = localDateTime.toString();

String initialLocalDateTimeStringDaySubstring = initialLocalDateTimeString.substring(8, 10);

String initialLocalDateTimeStringDashMonthDashSubstring = initialLocalDateTimeString.substring(4, 8);

String initialLocalDateTimeStringYearSubstring = initialLocalDateTimeString.substring(0, 4);

// Not providing as (10, 29) which caused an issue, as found out that intialLocalDateTimeString can be have 26, 27, 28, or 29 characters

String initialLocalDateTimeStringTimeSubString = initialLocalDateTimeString.substring(10);

String finalLocalDateTimeString = initialLocalDateTimeStringDaySubstring + initialLocalDateTimeStringDashMonthDashSubstring + initialLocalDateTimeStringYearSubstring + initialLocalDateTimeStringTimeSubString;

if(finalLocalDateTimeString.length()>=1) {

dayFirstDigitChar = finalLocalDateTimeString.charAt(0);

}

if(finalLocalDateTimeString.length()>=2) {

daySecondDigitChar = finalLocalDateTimeString.charAt(1);

}

if(finalLocalDateTimeString.length()>=4) {

monthFirstDigitChar = finalLocalDateTimeString.charAt(3);

}

if(finalLocalDateTimeString.length()>=5) {

monthSecondDigitChar = finalLocalDateTimeString.charAt(4);

}

if(finalLocalDateTimeString.length()>=7) {

yearFirstDigitChar = finalLocalDateTimeString.charAt(6);

}

if(finalLocalDateTimeString.length()>=8) {

yearSecondDigitChar = finalLocalDateTimeString.charAt(7);

}

if(finalLocalDateTimeString.length()>=9) {

yearThirdDigitChar = finalLocalDateTimeString.charAt(8);

}

if(finalLocalDateTimeString.length()>=10) {

yearFourthDigitChar = finalLocalDateTimeString.charAt(9);

}

if(finalLocalDateTimeString.length()>=12) {

hourFirstDigitChar = finalLocalDateTimeString.charAt(11);

}

if(finalLocalDateTimeString.length()>=13){

hourSecondDigitChar = finalLocalDateTimeString.charAt(12);

}

if(finalLocalDateTimeString.length()>=15) {

minuteFirstDigitChar = finalLocalDateTimeString.charAt(14);

}

if(finalLocalDateTimeString.length()>=16) {

minuteSecondDigitChar = finalLocalDateTimeString.charAt(15);

}

if(finalLocalDateTimeString.length()>=18) {

secondFirstDigitChar = finalLocalDateTimeString.charAt(17);

}

if(finalLocalDateTimeString.length()>=19) {

secondSecondDigitChar = finalLocalDateTimeString.charAt(18);

}

if(finalLocalDateTimeString.length()>=21) {

subsecondFirstDigitChar = finalLocalDateTimeString.charAt(20);

}

if(finalLocalDateTimeString.length()>=22) {

subsecondSecondDigitChar = finalLocalDateTimeString.charAt(21);

}

if(finalLocalDateTimeString.length()>=23) {

subsecondThirdDigitChar = finalLocalDateTimeString.charAt(22);

}

if(finalLocalDateTimeString.length()>=24) {

subsecondFourthDigitChar = finalLocalDateTimeString.charAt(23);

}

if(finalLocalDateTimeString.length()>=25) {

subsecondFifthDigitChar = finalLocalDateTimeString.charAt(24);

}

if(finalLocalDateTimeString.length()>=26) {

subsecondSixthDigitChar = finalLocalDateTimeString.charAt(25);

}

if(finalLocalDateTimeString.length()>=27) {

subsecondSeventhDigitChar = finalLocalDateTimeString.charAt(26);

}

if(finalLocalDateTimeString.length()>=28) {

subsecondEighthDigitChar = finalLocalDateTimeString.charAt(27);

}

if(finalLocalDateTimeString.length()>=29) {

subsecondNinthDigitChar = finalLocalDateTimeString.charAt(28);

}

TakesScreenshot screenshot = ((TakesScreenshot) driver);

File sourceFile = screenshot.getScreenshotAs(OutputType.*FILE*);

File destinationFile = new File("C:\\Users\\sohjnthn\\IdeaProjects\\Selenium\_Tutorial\\Selenium\_Screenshot\_Images\\" + dayFirstDigitChar + daySecondDigitChar + "-" + monthFirstDigitChar + monthSecondDigitChar + "-" + yearFirstDigitChar + yearSecondDigitChar + yearThirdDigitChar + yearFourthDigitChar + "T" + hourFirstDigitChar + hourSecondDigitChar + minuteFirstDigitChar + minuteSecondDigitChar + secondFirstDigitChar + secondSecondDigitChar + "." + subsecondFirstDigitChar + subsecondSecondDigitChar + subsecondThirdDigitChar + subsecondFourthDigitChar + subsecondFifthDigitChar + subsecondSixthDigitChar + subsecondSeventhDigitChar + subsecondEighthDigitChar + subsecondNinthDigitChar + "\_screenshot.png");

FileHandler.*copy*(sourceFile, destinationFile);

return "C:\\Users\\sohjnthn\\IdeaProjects\\Selenium\_Tutorial\\Selenium\_Screenshot\_Images\\" + dayFirstDigitChar + daySecondDigitChar + "-" + monthFirstDigitChar + monthSecondDigitChar + "-" + yearFirstDigitChar + yearSecondDigitChar + yearThirdDigitChar + yearFourthDigitChar + "T" + hourFirstDigitChar + hourSecondDigitChar + minuteFirstDigitChar + minuteSecondDigitChar + secondFirstDigitChar + secondSecondDigitChar + "." + subsecondFirstDigitChar + subsecondSecondDigitChar + subsecondThirdDigitChar + subsecondFourthDigitChar + subsecondFifthDigitChar + subsecondSixthDigitChar + subsecondSeventhDigitChar + subsecondEighthDigitChar + subsecondNinthDigitChar + "\_screenshot.png";

}

}

SecondAppObjects.java

package hellocucumber;

import com.aventstack.extentreports.ExtentReports;

import com.aventstack.extentreports.ExtentTest;

import com.aventstack.extentreports.MediaEntityBuilder;

import com.aventstack.extentreports.Status;

import com.aventstack.extentreports.reporter.ExtentSparkReporter;

import io.appium.java\_client.android.AndroidDriver;

import io.appium.java\_client.pagefactory.AndroidFindBy;

import org.openqa.selenium.\*;

import org.openqa.selenium.io.FileHandler;

import java.io.File;

import java.io.IOException;

import java.time.LocalDateTime;

public class SecondAppObjects {

// Second App

@AndroidFindBy(xpath = "//android.widget.TextView[@content-desc=\"App\"]")

WebElement appTab;

@AndroidFindBy(xpath = "//android.widget.TextView[@content-desc=\"Search\"]")

WebElement searchTab;

@AndroidFindBy(xpath = "//android.widget.TextView[@content-desc=\"Invoke Search\"]")

WebElement invokeSearchTab;

@AndroidFindBy(xpath = "//android.widget.EditText[@resource-id=\"io.appium.android.apis:id/txt\_query\_prefill\"]")

WebElement prefillQueryTextField;

String capturedScreenshotImageFilepathString = "";

public SecondAppObjects(AndroidDriver driver, ExtentTest extentTest, ExtentSparkReporter extentSparkReporter, ExtentReports extentReport) throws IOException {

WebElement appTab = driver.findElement(By.*xpath*("//android.widget.TextView[@content-desc=\"App\"]"));

if (appTab.isDisplayed()) {

capturedScreenshotImageFilepathString = *takeScreenshot*(driver);

extentTest.addScreenCaptureFromPath(capturedScreenshotImageFilepathString).pass(MediaEntityBuilder.*createScreenCaptureFromPath*(capturedScreenshotImageFilepathString).build()).log(Status.*PASS*, "App tab is displayed.");

} else {

capturedScreenshotImageFilepathString = *takeScreenshot*(driver);

extentTest.addScreenCaptureFromPath(capturedScreenshotImageFilepathString).pass(MediaEntityBuilder.*createScreenCaptureFromPath*(capturedScreenshotImageFilepathString).build()).log(Status.*FAIL*, "App tab is not displayed.");

}

appTab.click();

WebElement searchTab = driver.findElement(By.*xpath*("//android.widget.TextView[@content-desc=\"Search\"]"));

if (searchTab.isDisplayed()) {

capturedScreenshotImageFilepathString = *takeScreenshot*(driver);

extentTest.addScreenCaptureFromPath(capturedScreenshotImageFilepathString).pass(MediaEntityBuilder.*createScreenCaptureFromPath*(capturedScreenshotImageFilepathString).build()).log(Status.*PASS*, "Search tab is displayed.");

} else {

capturedScreenshotImageFilepathString = *takeScreenshot*(driver);

extentTest.addScreenCaptureFromPath(capturedScreenshotImageFilepathString).pass(MediaEntityBuilder.*createScreenCaptureFromPath*(capturedScreenshotImageFilepathString).build()).log(Status.*FAIL*, "Search tab is not displayed.");

}

searchTab.click();

WebElement invokeSearchTab = driver.findElement(By.*xpath*("//android.widget.TextView[@content-desc=\"Invoke Search\"]"));

if (invokeSearchTab.isDisplayed()) {

capturedScreenshotImageFilepathString = *takeScreenshot*(driver);

extentTest.addScreenCaptureFromPath(capturedScreenshotImageFilepathString).pass(MediaEntityBuilder.*createScreenCaptureFromPath*(capturedScreenshotImageFilepathString).build()).log(Status.*PASS*, "Invoke Search tab is displayed.");

} else {

capturedScreenshotImageFilepathString = *takeScreenshot*(driver);

extentTest.addScreenCaptureFromPath(capturedScreenshotImageFilepathString).pass(MediaEntityBuilder.*createScreenCaptureFromPath*(capturedScreenshotImageFilepathString).build()).log(Status.*FAIL*, "Invoke Search tab is not displayed.");

}

invokeSearchTab.click();

WebElement prefillQueryTextField = driver.findElement(By.*xpath*("//android.widget.EditText[@resource-id=\"io.appium.android.apis:id/txt\_query\_prefill\"]"));

prefillQueryTextField.sendKeys("Test the text.");

if (prefillQueryTextField.isDisplayed()) {

capturedScreenshotImageFilepathString = *takeScreenshot*(driver);

extentTest.addScreenCaptureFromPath(capturedScreenshotImageFilepathString).pass(MediaEntityBuilder.*createScreenCaptureFromPath*(capturedScreenshotImageFilepathString).build()).log(Status.*PASS*, "Able to enter the text \"Test the text.\".");

System.*out*.println("Pass");

} else {

capturedScreenshotImageFilepathString = *takeScreenshot*(driver);

extentTest.addScreenCaptureFromPath(capturedScreenshotImageFilepathString).pass(MediaEntityBuilder.*createScreenCaptureFromPath*(capturedScreenshotImageFilepathString).build()).log(Status.*FAIL*, "Unable to enter the text \"Test the text.\".");

System.*out*.println("Fail");

}

}

public static String takeScreenshot(WebDriver driver) throws IOException {

char dayFirstDigitChar = '0';

char daySecondDigitChar = '0';

char monthFirstDigitChar = '0';

char monthSecondDigitChar = '0';

char yearFirstDigitChar = '0';

char yearSecondDigitChar = '0';

char yearThirdDigitChar = '0';

char yearFourthDigitChar = '0';

char hourFirstDigitChar = '0';

char hourSecondDigitChar = '0';

char minuteFirstDigitChar = '0';

char minuteSecondDigitChar = '0';

char secondFirstDigitChar = '0';

char secondSecondDigitChar = '0';

char subsecondFirstDigitChar = '0';

char subsecondSecondDigitChar = '0';

char subsecondThirdDigitChar = '0';

char subsecondFourthDigitChar = '0';

char subsecondFifthDigitChar = '0';

char subsecondSixthDigitChar = '0';

// Issue - Missing these three digits

char subsecondSeventhDigitChar = '0';

char subsecondEighthDigitChar = '0';

char subsecondNinthDigitChar = '0';

LocalDateTime localDateTime = LocalDateTime.*now*();

String initialLocalDateTimeString = localDateTime.toString();

String initialLocalDateTimeStringDaySubstring = initialLocalDateTimeString.substring(8, 10);

String initialLocalDateTimeStringDashMonthDashSubstring = initialLocalDateTimeString.substring(4, 8);

String initialLocalDateTimeStringYearSubstring = initialLocalDateTimeString.substring(0, 4);

// Not providing as (10, 29) which caused an issue, as found out that intialLocalDateTimeString can be have 26, 27, 28, or 29 characters

String initialLocalDateTimeStringTimeSubString = initialLocalDateTimeString.substring(10);

String finalLocalDateTimeString = initialLocalDateTimeStringDaySubstring + initialLocalDateTimeStringDashMonthDashSubstring + initialLocalDateTimeStringYearSubstring + initialLocalDateTimeStringTimeSubString;

if(finalLocalDateTimeString.length()>=1) {

dayFirstDigitChar = finalLocalDateTimeString.charAt(0);

}

if(finalLocalDateTimeString.length()>=2) {

daySecondDigitChar = finalLocalDateTimeString.charAt(1);

}

if(finalLocalDateTimeString.length()>=4) {

monthFirstDigitChar = finalLocalDateTimeString.charAt(3);

}

if(finalLocalDateTimeString.length()>=5) {

monthSecondDigitChar = finalLocalDateTimeString.charAt(4);

}

if(finalLocalDateTimeString.length()>=7) {

yearFirstDigitChar = finalLocalDateTimeString.charAt(6);

}

if(finalLocalDateTimeString.length()>=8) {

yearSecondDigitChar = finalLocalDateTimeString.charAt(7);

}

if(finalLocalDateTimeString.length()>=9) {

yearThirdDigitChar = finalLocalDateTimeString.charAt(8);

}

if(finalLocalDateTimeString.length()>=10) {

yearFourthDigitChar = finalLocalDateTimeString.charAt(9);

}

if(finalLocalDateTimeString.length()>=12) {

hourFirstDigitChar = finalLocalDateTimeString.charAt(11);

}

if(finalLocalDateTimeString.length()>=13){

hourSecondDigitChar = finalLocalDateTimeString.charAt(12);

}

if(finalLocalDateTimeString.length()>=15) {

minuteFirstDigitChar = finalLocalDateTimeString.charAt(14);

}

if(finalLocalDateTimeString.length()>=16) {

minuteSecondDigitChar = finalLocalDateTimeString.charAt(15);

}

if(finalLocalDateTimeString.length()>=18) {

secondFirstDigitChar = finalLocalDateTimeString.charAt(17);

}

if(finalLocalDateTimeString.length()>=19) {

secondSecondDigitChar = finalLocalDateTimeString.charAt(18);

}

if(finalLocalDateTimeString.length()>=21) {

subsecondFirstDigitChar = finalLocalDateTimeString.charAt(20);

}

if(finalLocalDateTimeString.length()>=22) {

subsecondSecondDigitChar = finalLocalDateTimeString.charAt(21);

}

if(finalLocalDateTimeString.length()>=23) {

subsecondThirdDigitChar = finalLocalDateTimeString.charAt(22);

}

if(finalLocalDateTimeString.length()>=24) {

subsecondFourthDigitChar = finalLocalDateTimeString.charAt(23);

}

if(finalLocalDateTimeString.length()>=25) {

subsecondFifthDigitChar = finalLocalDateTimeString.charAt(24);

}

if(finalLocalDateTimeString.length()>=26) {

subsecondSixthDigitChar = finalLocalDateTimeString.charAt(25);

}

if(finalLocalDateTimeString.length()>=27) {

subsecondSeventhDigitChar = finalLocalDateTimeString.charAt(26);

}

if(finalLocalDateTimeString.length()>=28) {

subsecondEighthDigitChar = finalLocalDateTimeString.charAt(27);

}

if(finalLocalDateTimeString.length()>=29) {

subsecondNinthDigitChar = finalLocalDateTimeString.charAt(28);

}

TakesScreenshot screenshot = ((TakesScreenshot) driver);

File sourceFile = screenshot.getScreenshotAs(OutputType.*FILE*);

File destinationFile = new File("C:\\Users\\sohjnthn\\IdeaProjects\\Selenium\_Tutorial\\Selenium\_Screenshot\_Images\\" + dayFirstDigitChar + daySecondDigitChar + "-" + monthFirstDigitChar + monthSecondDigitChar + "-" + yearFirstDigitChar + yearSecondDigitChar + yearThirdDigitChar + yearFourthDigitChar + "T" + hourFirstDigitChar + hourSecondDigitChar + minuteFirstDigitChar + minuteSecondDigitChar + secondFirstDigitChar + secondSecondDigitChar + "." + subsecondFirstDigitChar + subsecondSecondDigitChar + subsecondThirdDigitChar + subsecondFourthDigitChar + subsecondFifthDigitChar + subsecondSixthDigitChar + subsecondSeventhDigitChar + subsecondEighthDigitChar + subsecondNinthDigitChar + "\_screenshot.png");

FileHandler.*copy*(sourceFile, destinationFile);

return "C:\\Users\\sohjnthn\\IdeaProjects\\Selenium\_Tutorial\\Selenium\_Screenshot\_Images\\" + dayFirstDigitChar + daySecondDigitChar + "-" + monthFirstDigitChar + monthSecondDigitChar + "-" + yearFirstDigitChar + yearSecondDigitChar + yearThirdDigitChar + yearFourthDigitChar + "T" + hourFirstDigitChar + hourSecondDigitChar + minuteFirstDigitChar + minuteSecondDigitChar + secondFirstDigitChar + secondSecondDigitChar + "." + subsecondFirstDigitChar + subsecondSecondDigitChar + subsecondThirdDigitChar + subsecondFourthDigitChar + subsecondFifthDigitChar + subsecondSixthDigitChar + subsecondSeventhDigitChar + subsecondEighthDigitChar + subsecondNinthDigitChar + "\_screenshot.png";

}

}

ThirdAppObjects.java

package hellocucumber;

import com.aventstack.extentreports.ExtentReports;

import com.aventstack.extentreports.ExtentTest;

import com.aventstack.extentreports.MediaEntityBuilder;

import com.aventstack.extentreports.Status;

import com.aventstack.extentreports.reporter.ExtentSparkReporter;

import io.appium.java\_client.android.AndroidDriver;

import io.appium.java\_client.pagefactory.AndroidFindBy;

import org.openqa.selenium.\*;

import org.openqa.selenium.io.FileHandler;

import java.io.File;

import java.io.IOException;

import java.time.LocalDateTime;

import java.util.Objects;

public class ThirdAppObjects {

// Third App

@AndroidFindBy(xpath = "//android.widget.EditText[@resource-id=\"com.microsoft.emmx.canary:id/url\_bar\"]")

WebElement edgeBrowserDefaultWebpageSearchBar;

@AndroidFindBy(xpath = "//android.widget.TextView[@resource-id=\"com.microsoft.emmx.canary:id/line\_1\"]\n")

WebElement requiredSearchOption;

@AndroidFindBy(xpath = "//android.widget.FrameLayout[@content-desc=\"Web View\"]")

WebElement searchResultsWebpageView;

@AndroidFindBy(xpath = "//android.widget.FrameLayout[@content-desc=\"Web View\"]")

WebElement aminoWebpageView;

String capturedScreenshotImageFilepathString = "";

public ThirdAppObjects(AndroidDriver driver, ExtentTest extentTest, ExtentSparkReporter extentSparkReporter, ExtentReports extentReport) throws IOException {

String oneString = "1";

// Due to the need to manually click on multiple sign in confirmation buttons

for(int i=0; i<1000000; i++) {

System.*out*.println("The app requires time to load.");

}

edgeBrowserDefaultWebpageSearchBar = driver.findElement(By.*xpath*("//android.widget.EditText[@resource-id=\"com.microsoft.emmx.canary:id/url\_bar\"]"));

if(edgeBrowserDefaultWebpageSearchBar.isDisplayed()){

capturedScreenshotImageFilepathString = *takeScreenshot*(driver);

extentTest.addScreenCaptureFromPath(capturedScreenshotImageFilepathString).pass(MediaEntityBuilder.*createScreenCaptureFromPath*(capturedScreenshotImageFilepathString).build()).log(Status.*PASS*, "The Edge Browser default webpage search bar is displayed.");

System.*out*.println("Pass");

} else {

capturedScreenshotImageFilepathString = *takeScreenshot*(driver);

extentTest.addScreenCaptureFromPath(capturedScreenshotImageFilepathString).pass(MediaEntityBuilder.*createScreenCaptureFromPath*(capturedScreenshotImageFilepathString).build()).log(Status.*FAIL*, "he Edge Browser default webpage search bar is not displayed.");

System.*out*.println("Fail");

}

edgeBrowserDefaultWebpageSearchBar.click();

if(edgeBrowserDefaultWebpageSearchBar.isDisplayed()){

capturedScreenshotImageFilepathString = *takeScreenshot*(driver);

extentTest.addScreenCaptureFromPath(capturedScreenshotImageFilepathString).pass(MediaEntityBuilder.*createScreenCaptureFromPath*(capturedScreenshotImageFilepathString).build()).log(Status.*PASS*, "The Edge Browser default webpage search bar is displayed.");

System.*out*.println("Pass");

} else {

capturedScreenshotImageFilepathString = *takeScreenshot*(driver);

extentTest.addScreenCaptureFromPath(capturedScreenshotImageFilepathString).pass(MediaEntityBuilder.*createScreenCaptureFromPath*(capturedScreenshotImageFilepathString).build()).log(Status.*FAIL*, "he Edge Browser default webpage search bar is not displayed.");

System.*out*.println("Fail");

}

edgeBrowserDefaultWebpageSearchBar.sendKeys("MapleStorySEA Unfunded Amino");

if(edgeBrowserDefaultWebpageSearchBar.isDisplayed()){

capturedScreenshotImageFilepathString = *takeScreenshot*(driver);

extentTest.addScreenCaptureFromPath(capturedScreenshotImageFilepathString).pass(MediaEntityBuilder.*createScreenCaptureFromPath*(capturedScreenshotImageFilepathString).build()).log(Status.*PASS*, "The Edge Browser default webpage search bar contains the entered search term \"MapleStorySEA Unfunded Amino\".");

System.*out*.println("Pass");

} else {

capturedScreenshotImageFilepathString = *takeScreenshot*(driver);

extentTest.addScreenCaptureFromPath(capturedScreenshotImageFilepathString).pass(MediaEntityBuilder.*createScreenCaptureFromPath*(capturedScreenshotImageFilepathString).build()).log(Status.*FAIL*, "The Edge Browser default webpage search bar does not contain the entered search term \"MapleStorySEA Unfunded Amino\".");

System.*out*.println("Fail");

}

requiredSearchOption = driver.findElement(By.*xpath*("//android.widget.TextView[@resource-id=\"com.microsoft.emmx.canary:id/line\_1\"]\n"));

if(requiredSearchOption.isDisplayed()){

capturedScreenshotImageFilepathString = *takeScreenshot*(driver);

extentTest.addScreenCaptureFromPath(capturedScreenshotImageFilepathString).pass(MediaEntityBuilder.*createScreenCaptureFromPath*(capturedScreenshotImageFilepathString).build()).log(Status.*PASS*, "The Edge Browser default webpage search bar contains the entered search term \"MapleStorySEA Unfunded Amino\"");

System.*out*.println("Pass");

} else {

capturedScreenshotImageFilepathString = *takeScreenshot*(driver);

extentTest.addScreenCaptureFromPath(capturedScreenshotImageFilepathString).pass(MediaEntityBuilder.*createScreenCaptureFromPath*(capturedScreenshotImageFilepathString).build()).log(Status.*FAIL*, "The Edge Browser default webpage search bar does not contain the entered search term \"MapleStorySEA Unfunded Amino\"");

System.*out*.println("Fail");

}

requiredSearchOption.click();

// There may be a very short delay

for (int i=0; i<2; i++) {

System.*out*.println("The required search results webpage is still loading.");

}

searchResultsWebpageView = driver.findElement(By.*xpath*("//android.widget.FrameLayout[@content-desc=\"Web View\"]"));

// The following line cannot be used as the WebElement is not found

// WebElement requiredSearchResultLink = driver.findElement(By.partialLinkText("Featured | [MapleStorySEA] Unfunded Tips"));

if(searchResultsWebpageView.isDisplayed()){

capturedScreenshotImageFilepathString = *takeScreenshot*(driver);

extentTest.addScreenCaptureFromPath(capturedScreenshotImageFilepathString).pass(MediaEntityBuilder.*createScreenCaptureFromPath*(capturedScreenshotImageFilepathString).build()).log(Status.*PASS*, "The required search results webpage is reached.");

System.*out*.println("Pass");

} else {

capturedScreenshotImageFilepathString = *takeScreenshot*(driver);

extentTest.addScreenCaptureFromPath(capturedScreenshotImageFilepathString).pass(MediaEntityBuilder.*createScreenCaptureFromPath*(capturedScreenshotImageFilepathString).build()).log(Status.*FAIL*, "The required search results webpage is not reached.");

System.*out*.println("Fail");

}

// For "//android.widget.FrameLayout[@content-desc=\"Web View\"]"

// sending the key KEYS.TAB would cause the test to fail

// We can only click on it

// And we cannot select any specific individual part as an element

// Noting that what was clicked on is not the correct search result for the default Featured tab, but is the next partialLinkText

// And would have an error if the next thing to click on is a video

searchResultsWebpageView.click();

// unable to detect any more

// driver.findElement(By.xpath("//android.widget.FrameLayout[@content-desc=\"Web View\"]"));

// The loading for UI change is very slow, we try to capture three screenshot images

for(int i=0; i<1000000; i++) {

System.*out*.println("A nearby search result would be clicked on.");

}

if (Objects.*equals*(oneString, "1")) {

capturedScreenshotImageFilepathString = *takeScreenshot*(driver);

extentTest.addScreenCaptureFromPath(capturedScreenshotImageFilepathString).pass(MediaEntityBuilder.*createScreenCaptureFromPath*(capturedScreenshotImageFilepathString).build()).log(Status.*PASS*, "A nearby search result would be clicked on.");

System.*out*.println("Pass");

} else {

capturedScreenshotImageFilepathString = *takeScreenshot*(driver);

extentTest.addScreenCaptureFromPath(capturedScreenshotImageFilepathString).pass(MediaEntityBuilder.*createScreenCaptureFromPath*(capturedScreenshotImageFilepathString).build()).log(Status.*FAIL*, "A nearby search results webpage's search results would not not be clicked on.");

System.*out*.println("Fail");

}

for(int i=0; i<1000000; i++) {

System.*out*.println("A nearby search result would be clicked on.");

}

if (Objects.*equals*(oneString, "1")) {

capturedScreenshotImageFilepathString = *takeScreenshot*(driver);

extentTest.addScreenCaptureFromPath(capturedScreenshotImageFilepathString).pass(MediaEntityBuilder.*createScreenCaptureFromPath*(capturedScreenshotImageFilepathString).build()).log(Status.*PASS*, "A nearby search result would be clicked on.");

System.*out*.println("Pass");

} else {

capturedScreenshotImageFilepathString = *takeScreenshot*(driver);

extentTest.addScreenCaptureFromPath(capturedScreenshotImageFilepathString).pass(MediaEntityBuilder.*createScreenCaptureFromPath*(capturedScreenshotImageFilepathString).build()).log(Status.*FAIL*, "A nearby search results webpage's search results would not not be clicked on.");

System.*out*.println("Fail");

}

for(int i=0; i<1000000; i++) {

System.*out*.println("A nearby search result would be clicked on.");

}

if (Objects.*equals*(oneString, "1")) {

capturedScreenshotImageFilepathString = *takeScreenshot*(driver);

extentTest.addScreenCaptureFromPath(capturedScreenshotImageFilepathString).pass(MediaEntityBuilder.*createScreenCaptureFromPath*(capturedScreenshotImageFilepathString).build()).log(Status.*PASS*, "A nearby search result would be clicked on.");

System.*out*.println("Pass");

} else {

capturedScreenshotImageFilepathString = *takeScreenshot*(driver);

extentTest.addScreenCaptureFromPath(capturedScreenshotImageFilepathString).pass(MediaEntityBuilder.*createScreenCaptureFromPath*(capturedScreenshotImageFilepathString).build()).log(Status.*FAIL*, "A nearby search results webpage's search results would not not be clicked on.");

System.*out*.println("Fail");

}

extentReport.flush();

}

public static String takeScreenshot(WebDriver driver) throws IOException {

char dayFirstDigitChar = '0';

char daySecondDigitChar = '0';

char monthFirstDigitChar = '0';

char monthSecondDigitChar = '0';

char yearFirstDigitChar = '0';

char yearSecondDigitChar = '0';

char yearThirdDigitChar = '0';

char yearFourthDigitChar = '0';

char hourFirstDigitChar = '0';

char hourSecondDigitChar = '0';

char minuteFirstDigitChar = '0';

char minuteSecondDigitChar = '0';

char secondFirstDigitChar = '0';

char secondSecondDigitChar = '0';

char subsecondFirstDigitChar = '0';

char subsecondSecondDigitChar = '0';

char subsecondThirdDigitChar = '0';

char subsecondFourthDigitChar = '0';

char subsecondFifthDigitChar = '0';

char subsecondSixthDigitChar = '0';

// Issue - Missing these three digits

char subsecondSeventhDigitChar = '0';

char subsecondEighthDigitChar = '0';

char subsecondNinthDigitChar = '0';

LocalDateTime localDateTime = LocalDateTime.*now*();

String initialLocalDateTimeString = localDateTime.toString();

String initialLocalDateTimeStringDaySubstring = initialLocalDateTimeString.substring(8, 10);

String initialLocalDateTimeStringDashMonthDashSubstring = initialLocalDateTimeString.substring(4, 8);

String initialLocalDateTimeStringYearSubstring = initialLocalDateTimeString.substring(0, 4);

// Not providing as (10, 29) which caused an issue, as found out that intialLocalDateTimeString can be have 26, 27, 28, or 29 characters

String initialLocalDateTimeStringTimeSubString = initialLocalDateTimeString.substring(10);

String finalLocalDateTimeString = initialLocalDateTimeStringDaySubstring + initialLocalDateTimeStringDashMonthDashSubstring + initialLocalDateTimeStringYearSubstring + initialLocalDateTimeStringTimeSubString;

if(finalLocalDateTimeString.length()>=1) {

dayFirstDigitChar = finalLocalDateTimeString.charAt(0);

}

if(finalLocalDateTimeString.length()>=2) {

daySecondDigitChar = finalLocalDateTimeString.charAt(1);

}

if(finalLocalDateTimeString.length()>=4) {

monthFirstDigitChar = finalLocalDateTimeString.charAt(3);

}

if(finalLocalDateTimeString.length()>=5) {

monthSecondDigitChar = finalLocalDateTimeString.charAt(4);

}

if(finalLocalDateTimeString.length()>=7) {

yearFirstDigitChar = finalLocalDateTimeString.charAt(6);

}

if(finalLocalDateTimeString.length()>=8) {

yearSecondDigitChar = finalLocalDateTimeString.charAt(7);

}

if(finalLocalDateTimeString.length()>=9) {

yearThirdDigitChar = finalLocalDateTimeString.charAt(8);

}

if(finalLocalDateTimeString.length()>=10) {

yearFourthDigitChar = finalLocalDateTimeString.charAt(9);

}

if(finalLocalDateTimeString.length()>=12) {

hourFirstDigitChar = finalLocalDateTimeString.charAt(11);

}

if(finalLocalDateTimeString.length()>=13){

hourSecondDigitChar = finalLocalDateTimeString.charAt(12);

}

if(finalLocalDateTimeString.length()>=15) {

minuteFirstDigitChar = finalLocalDateTimeString.charAt(14);

}

if(finalLocalDateTimeString.length()>=16) {

minuteSecondDigitChar = finalLocalDateTimeString.charAt(15);

}

if(finalLocalDateTimeString.length()>=18) {

secondFirstDigitChar = finalLocalDateTimeString.charAt(17);

}

if(finalLocalDateTimeString.length()>=19) {

secondSecondDigitChar = finalLocalDateTimeString.charAt(18);

}

if(finalLocalDateTimeString.length()>=21) {

subsecondFirstDigitChar = finalLocalDateTimeString.charAt(20);

}

if(finalLocalDateTimeString.length()>=22) {

subsecondSecondDigitChar = finalLocalDateTimeString.charAt(21);

}

if(finalLocalDateTimeString.length()>=23) {

subsecondThirdDigitChar = finalLocalDateTimeString.charAt(22);

}

if(finalLocalDateTimeString.length()>=24) {

subsecondFourthDigitChar = finalLocalDateTimeString.charAt(23);

}

if(finalLocalDateTimeString.length()>=25) {

subsecondFifthDigitChar = finalLocalDateTimeString.charAt(24);

}

if(finalLocalDateTimeString.length()>=26) {

subsecondSixthDigitChar = finalLocalDateTimeString.charAt(25);

}

if(finalLocalDateTimeString.length()>=27) {

subsecondSeventhDigitChar = finalLocalDateTimeString.charAt(26);

}

if(finalLocalDateTimeString.length()>=28) {

subsecondEighthDigitChar = finalLocalDateTimeString.charAt(27);

}

if(finalLocalDateTimeString.length()>=29) {

subsecondNinthDigitChar = finalLocalDateTimeString.charAt(28);

}

TakesScreenshot screenshot = ((TakesScreenshot) driver);

File sourceFile = screenshot.getScreenshotAs(OutputType.*FILE*);

File destinationFile = new File("C:\\Users\\sohjnthn\\IdeaProjects\\Selenium\_Tutorial\\Selenium\_Screenshot\_Images\\" + dayFirstDigitChar + daySecondDigitChar + "-" + monthFirstDigitChar + monthSecondDigitChar + "-" + yearFirstDigitChar + yearSecondDigitChar + yearThirdDigitChar + yearFourthDigitChar + "T" + hourFirstDigitChar + hourSecondDigitChar + minuteFirstDigitChar + minuteSecondDigitChar + secondFirstDigitChar + secondSecondDigitChar + "." + subsecondFirstDigitChar + subsecondSecondDigitChar + subsecondThirdDigitChar + subsecondFourthDigitChar + subsecondFifthDigitChar + subsecondSixthDigitChar + subsecondSeventhDigitChar + subsecondEighthDigitChar + subsecondNinthDigitChar + "\_screenshot.png");

FileHandler.*copy*(sourceFile, destinationFile);

return "C:\\Users\\sohjnthn\\IdeaProjects\\Selenium\_Tutorial\\Selenium\_Screenshot\_Images\\" + dayFirstDigitChar + daySecondDigitChar + "-" + monthFirstDigitChar + monthSecondDigitChar + "-" + yearFirstDigitChar + yearSecondDigitChar + yearThirdDigitChar + yearFourthDigitChar + "T" + hourFirstDigitChar + hourSecondDigitChar + minuteFirstDigitChar + minuteSecondDigitChar + secondFirstDigitChar + secondSecondDigitChar + "." + subsecondFirstDigitChar + subsecondSecondDigitChar + subsecondThirdDigitChar + subsecondFourthDigitChar + subsecondFifthDigitChar + subsecondSixthDigitChar + subsecondSeventhDigitChar + subsecondEighthDigitChar + subsecondNinthDigitChar + "\_screenshot.png";

}

}