Topics we cover for rest of the course:

* Selenium Webdriver
* Cucumber (BDD Framework)
* TestNG ( Data driven Framework)
* Rest full Services Testing (API Testing)
  + Postman
  + SOAPUI
  + Ready API
  + Rest Assured or Karate Framework
* GIT, Jenkins, Maven

What is Selenium Webdriver?

Selenium is an open source tool which is use to automate Web based applications that are supported by Chrome, FF Mozila, IE, Edge, Safari. Selenium can be written with Java, C#, Ruby, python. In this course we use Java and selenium to automate UI of web applications.

Selenium Grid, is used to run parallel execution of Test cases in different machines.

Selenium IDE, is firefox plugin that is used to record and play back on firefox browser.

Selenium Remote Control, this one was merge with Selenium 2 and then Seelnium Webdriver was created.

What is WebDriver ?

Webdriver is an interface which is can be used with different browser classes such as Chrome, Firefox, IE, Edge, Safar.

Requirements for Selenium Webdriver:

* Java installed
* Eclipse IDE
* Download Jar files from selenium Website
* Download .exe files for Chrome driver, Firefox driver, Edge driver
* We need to create a java project
* We need to import jar files to project library
* We need to create a java class
* We need to write codes to open tekschool website through chrome browser using java and selenium.
* Step 1: Create a java project and name it **Selenium lectures**
* Step 2: add jar files from java build path
* Step 3: create a folder on project level and name it drivers
* Step4: copy chromedriver and past it to drivers folder
* Step 5: create a package and name it basics
* Step6:

Implicitly wait is global wait in selenium and it defines for entire execution. We define certain amount of time for webdriver before it throws an exception. For example when element is not found on UI it will wait for certain amount of time and then it will throw exception.

* How we can open a webpage using selenium?

We can use driver.get and driver.navigate.to() and then pass url as string

* What is difference between driver.get and driver.navigate.to()

Driver.get() will open page and wait until page is loaded while driver.navigate opens page and move to next step without waiting for pageload

* What are navigations command used in selenium?

Driver.navigate.refresh()- to refresh a page

Driver.navigate.back() – to move to previous page we opened

Driver.navigate.forward()- to move to page we opened before navigating back to this page.

* How we can close a browser that is opened by selenium?

Driver.close and driver.quit()

* What is difference between driver.close() and driver.quit() ?

Driver.close will close one tab of browser opened by selenium and driver.quit() will close all tabs that are opened by selenium.

Class codes:

System.setProperty("webdriver.gecko.driver", ".\\drivers\\geckodriver.exe");

WebDriver driver = new FirefoxDriver();

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

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

System.out.println("Website Title " + driver.getTitle());

Thread.sleep(5000);// This is hard stop on execution

driver.navigate().refresh();

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

System.out.println("Website Title " + driver.getTitle());

Thread.sleep(5000);

driver.navigate().back();

Thread.sleep(5000);

driver.navigate().forward();

Thread.sleep(5000);

driver.close();

September 23 Notes:

User Interface (UI ) Elements: Every item on GUI is called Webelement.

Locators: are used to find WebElements in UI.

* ID: Unique and fastest locator ( preferred )
* Name:
* TagName:
* ClassName:
* LinkText:
* CssSelector:
* Xpath:
  + Absolute: starts with / and it starts from top of HTML tag
    - Not recommended to use
    - As element can be change from one location to another
  + Relative: starts with // and it starts from tag of element and syntax: **//tagName[@attribute=’value’]**
    - Recommended to use and we can use multiple attributes,
    - We can use indexing
    - We can use parent to child relationship and vise versa
    - We can use siblings relationship

<input class="ic-Input text" autofocus="autofocus" type="text" name="pseudonym\_session[unique\_id]" id="pseudonym\_session\_unique\_id">

When we inspect elements in UI we have to make sure it is Unique, IF there is ID attribute then we go with ID, and if there is no ID then we can look for unique name attribute, unique className attribute and if we don’t find them then we can look for xpath and css Selector.

Selenium provides a method “findElement()” and this method will find element of UI and the return type is WebElement and we can use locators to find element in UI.

Commands on Webelement:

To enter value we use .sendKeys(“value”);

To click on a button we use .click();

To Find an element in UI using selenium we use driver.findElement(By.LocatorType(“locator”));

We store each UI element as WebElement in Selenium.

We can clear the input value with .clear() method

isDisplayed() method return Boolean value and if element is present it will return true and if not it will return false

IsEnabled() method return Boolean value and if element is enabled, it will return true and if not it will return false

Class code

System.setProperty("webdriver.chrome.driver", ".\\drivers\\chromedriver.exe");

WebDriver driver = new ChromeDriver();

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

driver.manage().timeouts().implicitlyWait(10, TimeUnit.SECONDS);

driver.get("https://canvas.instructure.com/login/canvas");

WebElement email =driver.findElement(By.id("pseudonym\_session\_unique\_id"));

email.sendKeys("instructor@tekschool.us");// this method will send value to email field

Thread.sleep(5000);

WebElement password = driver.findElement(By.name("pseudonym\_session[password]"));

password.sendKeys("123");

Thread.sleep(5000);

WebElement login = driver.findElement(By.xpath("//\*[@id=\"login\_form\"]/div[3]/div[2]/button"));

login.click(); // this method will click on ui element

driver.close();

Class Review: September 27, 2021

WebDriver driver = new ChromeDriver(); or WebDriver driver = new FirefoxDriver();

Why not WebDriver driver = new WebDriver(); ? When we start doing cross browser testing we can create one reference to Webdriver then assign multiple browsers.

What does cross browser testing means? --- means we can test same test cases on different browsers such as chrome, FF, IE, Edge, Safari, opera and headless.

driver.get(): to lunch an application on any browser

driver.navigate().to(): to lunch an application on any browser

driver.close: this closes the current tab of browser

driver.quit(): this closes all tabs of browser

driver.findElement(By.LocatorType()): This method finds an element on UI and stores it as WebElement.

WebElement: any element on User Interface or UI is called WebElement and Selenium provides WebElment interface to store elments we find from UI using Locators.

Locators: are used to find Elements in UI and we have ID, Name, ClassName, CssSelector, Xpath, Linktext, partialLinktext. We mainly use ID, Xpath, CssSelector.

driver.click(): with this method we can click on element.

driver.sendKeys(): with this method we can pass value to input element, it means same thing as we type in a field in UI.

driver.clear(); with this method we can clear the existing value in an input field

isDesplayed: the return type for this method is Boolean and if element is present in UI, it will return True and if it is not present it will return false.

isEnabled: the return type is Boolean, and it check if element is clickable, or element is enabled or not.

driver.getTitle(); this method returns title of website and we can write a verification to make sure title is correct based on expected result.

Waits:

ImplicitlyWait(time, timeUnit); it means how long should driver wait before throwing an exception. For example if an element is not present or the locator is wrong driver will wait for certain amount of time, but if element is present and driver found it, then it will move to next line of code or function. And this is called global wait too, since this wait applies to entire execution.

PageloadTimeOut(Time, timeUnit): it means how long webdriver should wait before throwing an exception. Some applications take more time to load fully while others load within a sec. we have to define this to handle page load of application.

Thread.Sleep(milliseconds ); 1 second = 1000 milliseconds: This is hard stop and it will stop driver from execution for certain amount of time we provide. This is not recommend to use at all, unless you wanna slow down automation and debug the code.

Class Lecture September 27, 2021:

Xpath is one of the most used locators in UI automation. And we have two types xpath:

* Absolute xpath: starts from top of html tag and travels all the way down to element. This is not recommended to use unless the is no other way. Absolute xpath starts with single slash(/) Example for this can be : /html/body/div[2]/div/div/form/fieldset[1]/div[2]/div/input
* Relative xpath: starts with // and it means location of that element tag and it is most used when we don’t have a unique attribute to find an element.

Syntax for basic xpath : //tagName[@attributeName=’valueOfAttribute’]

Example: //input[@name='firstname']

How to handle dynamic elements and elements which does not have attributes?

- If element does not have unique attribute but has a text we can use text function to write an xpath:

Syntax: //tagName[text()=’value of text’]

Example: //span[text()='My Account']

-if element value changes but there is a unique word or character we can use contains() function in xpath:

For example:

<span class="hidden-xs hidden-sm hidden-md">My Account</span>

<span class="hidden-xs hidden-sm hidden-md">My Account User</span>

<span class="hidden-xs hidden-sm hidden-md">My Account Visitor</span>

<span class="hidden-xs hidden-sm hidden-md">My Account Admin</span>

Syntax: //tagName[contains(text(),’Account’)]

Example: //span[contains(text(),'Account')]

September 28, 2021 Class notes:

driver.findElement(By.Locator());

* Returns single element from UI and return type is WebElement
* If no Element found it throws an exception “NoSuchElmenentFound”

driver.FindELements(By.Locator());

* Returns list of Elements which has same locator or tag, and return type is list of WebElement
* If no Elements found it will return an empty list and size of the list will be zero.
* For handling dynamic tables, dynamic dropdows and also finding links on UI.

Drop downs:

- Static Dropdown

* Select Class: we will create Object of Select class and then we can use
* select.Index
* select.value
* select.visibleText
* The value on dropdown is fixed and does not change

- Dynamic Dropdown.

* We need to use FindElements() method and store list of elements from dropdown in a list
* We need to iterate through each index of list
* We need to write .getText method to get text of list
* We need to write .click() method to click or select the value from the dropdown.

September 29, 2021:

* How we can handle DropDowns in selenium?

For Static dropdowns that has select tag we can use Select class and then use methods such as select.value, select.index, select.visible text(), for dynamic dropdowns we should use FindElements() method, then store in a list and then we can write a loop to iterate through the loop and based on conditions we can write an if condition to either click, or enter value.

* What is different between find Element and FindElements ?

findElement finds single element on UI, and return type is WebElement, if element is not found it will throw an exception. While FindElements finds list of elements that has a common locator (tag, xpath) and return type is List of WebElement, if no element is found it will return an empty list.

Action Class in Selenium can enable us to perform below functions:

* Mouse hover to an element
* Right click
* Double click
* Drag and drop
* How to Handle multiple windows in Selenium?

In order to multiple windows we have to switch from parent window to child window. And selenium provides Windowhandles method which the return type is Set. Once we get set of windows we store them in a Set collection then we use Iterator collection to travel from parent window to child window and vise versa.

* How to handle browser based pop ups and alerts in selenium?

In selenium we have switchTo method then we can switch to Alert then if we want to accept we can say.accept or if we want to deny we can select .dismiss and if we want to get text of alert then we can use .getText.

* How to handle the authentication Pop up in Selenium?

In selenium we can pass username and password in concatenation with url to by pass the authentication pop up

url: <http://the-internet.herokuapp.com/basic_auth>

username: admin

password: admin

when we want to by pass or authorize user through authentication pop up we have to follow below syntax:

http://userName:password@the-internet.herokuapp.com/basic\_auth

October 4, 2021

-How to take screenshot in Selenium?

Selenium provides a Interface TakeScreenshot and method called getScreenShotAS and we need to provide Type of file and location once Screenshot is taken.

First we will create object of File class then assign the TakeScreenshot output

Second we will use FileUtiles class.copyFile method to copy and store the file.

Waits in Selenium:

Implicit wait: it is a global wait that we define at the beginning of execution and it will be applicable till end of execution. In Implicit wait we tell webdriver wait for certain amount of time before throwing an exception.

Driver.manage().TimeOut().implicitlywait(waitTime, TimeUnit.TimeFormat);

Explicit wait: is used to tell webdriver to wait for certain conditions and certain element before throwing an exception.

We need to create Object WebdriverWait and then assign the condition and time to wait.

Fluent wait: The Fluent Wait in Selenium is used to define maximum time for the web driver to wait for a condition and frequency which we want to check element conditions before throwing an exception.

Iframes in Selenium: iframes and frames are small html pages inside the UI HTML page, and In order to handle these iframes and frames we need to use driver.SwitchTO().Frame(“ frame id, frame webelement). And if we want to get out of frame after we finish with function we need to use switch to defaultContent method.

How to upload a file using selenium?

Selenium web driver supports uploading file but not downloading file. In Selenium we can use SendKeys method to upload a file.

JavaScriptExecutor Interface: JSE is an interface provided by selenium to handle or execute JS within in current pages in Browser.

* Click with JavaScriptExecutor
* Send Value with JavaScripExecutor
* Handle checkbox to check if it is true or false ( true = selected, false = not selected)
* To refresh a browser
* To get page title
* To find hidden elements with JS
* To scroll down

How to send a value in Selenium with out using sendkeys.

JavaScriptExecutor can handle Calendars selection as well. And also for sending values using JS Executor we need make sure element has Id or an attribute with unique value.

October 5, 2021

BDD Framework:

Framework is a set of codes, tools, and plugins that makes test automation robust, easy to maintain, and reduce redundancy. The major components of a Test Automation Framework are:

* Maven: This is a build management tool, we use maven to store our dependencies (jar files), plugins, and maven build provides compilation, and generating reports after build is done. Maven provides a POM.xml file (Project Object Model) and it is based on xml format.
* POM Design Pattern: We use this pattern for easy maintenance of our UI Elements and to reduce redundancy in our coding. POM Design pattern is implemented by creating separate java classes for each UI page to store it is element. The way we implement POM that we use PageFactory class and initiElements method to initialize the webElements in this class.
* Feature file: Cucumber provides a file with extension “.feature” and Feature file starts with keyword Feature, we have to provide name for this feature. In this feature file we will write Scenario and Scenario Outlines using Gherkin Keywords.
  + Feature file has extension of “ .feature”
  + Scenario is when we write one test case
  + Scenario outline is when we write one test case and we want to run same scenario with different set of data using Examples keyword.
  + Gherkin keywords:
    - Given : This is a precondition
    - When : This is an action
    - Then: This is an expected result
    - And: This keyword can be use to follow actions after When keyword.

For example:

Scenario: Test Login

Given User is on Bank website { this is precondition}

When User enter username<value> and Password< value> { user takes action}

And User click on Login button { following previous action}

Then User should be logged in to account dashboard { This is expected behavior}

Background Keyword: When we have repeated steps at the beginning of each scenario or scenario outline we use Background keyword after Feature keyword and store all steps there. So every time a test scenario is running first it will run steps under Background Keyword then it will run scenario steps.

Scenario Outline: Test Login with multiple users

Given User is on Bank website

When User enter username<userName> and Password< password>

And User click on Login button

Then User should be logged in to account dashboard

Examples:

|UserName|password|

|admin|test@1234|

|customer|customer@1234|

|banker|banker@123|

|guest|guest@1233|

Hooks: in cucumber we have two types of hooks.

* Before Hooks: @Before, before hooks run the block of code before each scenario. Any code that needs to be run before each scenario should be placed in this block of code. For example, type of browser we execute the code, windows property such as maximize the windows, pageload timeout, implicitwait, and get url.
* After Hooks: @After, after hooks run the block of code after each scenario. Any code that needs to be run after each scenario should be place in this block of code. For example, scenario status( fail, pass, skipped), loggers to log the status of scenario, and teardown (close or quit the browser)

Step Definitions: are actual Java classes where we write java methods to perform test automation. Each Step of scenario or scenario outline will be linked with a java method in step definitions. Few points to remember while writing step definitions methods:

* Each step is linked through the annotations (@Given, @When, @Then, @And)

For example a step in scenario can be linked with step definition like below:

Given User is on Bank website < scenario step in feature file with gherkin format>

How above step will be linked or map to step definition java method?

@Given(“^User is on Bank website$”)

Public void user\_is\_on\_Bank\_Website(){

// implementation code

}

* Step name in scenario should be exactly matching the @ annotation line of code in step definition class
* One step can be written n times in different scenarios, but it should have only one step definition method.

Runner Class: is the start point of this framework to execute test scenarios. Test runner runs on top of CucumberOptions and it runs with @RunWith annotation of Junit framework.

* Features : we provide path of folder where we store all our feature files
* Glue: we provide path of package where we store all stepDefinitions classes
* Tags: here we provide tags of scenario, or scenarios, or features
* dryRun: this option checks if any step in scenario miss an implementation step in step definition class
* monochrome: this is for console readability
* strict: this will check if any steps are pending or undefined
* format: this will generate reports

Utility methods: are methods that will be reuse multiple time over course of automation. For example, screenshot method is a utility method, instead of writing the code every time we need to take screenshot, we will write this method in a separate java class then anytime we need it we will call this method. It will help to reduce redundancy and it improved reusability.

OOPS implementation in this framework:

Base class: this class will act as super or parent class for entire project. In this class we define the reference to Webdriver interface, we create object of Logger class from log4j, and we define the entire property of framework that other sub or child classes needs to inherit from parent class. This is concept of Inheritance where child classes will extend property of Parent class.

Encapsulation: When we implement POM (page Object Model) Design pattern we will create Separate Java classes and inside these classes will store UI WebElements in private variables, and access them through public methods. So this is implementing concept of encapsulation. The reason why we do this to limit the access of variables from another POM class if it has same variable in different UI page.

Abstraction: In Test Automation framework the way abstraction is work, we separate the implementation layer from test layer. So UI elements and methods are stored in POM classes and when we test we just access public methods to our test classes, it means we only see what is in method returned, not implementation part.

Interfaces: In Test Automation framework we implement many interfaces, such as Webdriver interface, Screenshot Interface, for Reporting part we implement IReporter Interface that will generate reports at the end of each execution. For Data driven framework we use TestNG listeners that implements ItestListeners interface.

Interview Questions:

* what is difference between scenario and scenario outline in cucumber?
* What is Background Keyword?
* What is Hooks in Cucumber?
* what is Step definitions ?
* How we link scenario step with Step Definitions method in cucumber?
* What is @RunWith in Test Runner
* What is CucumberOptions
* How to run a scenario from Test Runner
* What is the build management tool used in your framework?
* What is the design pattern in your framework?
* What is POM design pattern?
* What is Page Factory?
* Have you implemented OOPS concepts in your framework automation?
* Please walk us through the flow of your framework?
* What is usage of Examples keyword in cucumber?
* What is the extension for feature file
* How you generate reports?
* Why POM design pattern?

October 7, 2021:

December first Week, we will start Resume building, and Interview Preparations. till December 24.

* First you start applying for job on job portals.
* Second you will be contacted by recruiters
  + Are you available in market for new job?
  + What tools you are familiar working with
  + Are you US citizen or green card holder or Work permit EAD.
  + Rate hourly, annually.
  + Few Slots for Technical Screening 30 min – 60 technical screening.
* Technical Screening:
  + Tell me about Yourself & please brief us on your background
    - Coding Language
    - Methodologies
    - Tools
    - UI Automation & API automation
    - CICD tools
    - Frameworks
  + Technical questions starts:
    - Java coding challenges 80%
    - Framework questions
      * Design pattern
      * Feature file
      * Step Definitions
      * Test Runner
      * Maven as Build management tool
    - Selenium
    - SQL
    - Agile
    - API
    - CICD
* Do you have any question for us?
  + Thanks, the interviewer for their time.
  + What will be next step?
  + How many rounds of interview will be there?
  + What framework you guys using in your project?
  + What methodology you guys are following there?
* Final technical Interview:
  + Will be more in detail and with panel of interviewers from team you will work with.
    - Test Lead
    - Developer
    - Hiring Manager
    - SDET
* HR Round of Interview (this is 50/50)
  + Send offer
  + HR Interview

Interview Questions:

What type of tests have you automated?

-We are mainly automating Regression and Smoke Suite. and our smoke suite is focused on testing major functionality of application. While regression is run before each releases.

- How many test cases you have automated per day?

It depends on complexity of test cases, number of steps and dependencies to another test case, with meetings and other works I would say between 1-2 test case.

October 12, 2021:

Link to maven central repository to get dependency xml codes:

<https://mvnrepository.com/>

* Junit dependency
* Selenium dependency
* Cucumber dependency
* Commons io
* Log4j
* Webdriver manager
* Cucumber reporting

Maven life cycles:

Maven clean: This command will clean the target folder and will create fresh target folder to store new dependencies for new build

Maven install: checks if there are any missing dependencies on local .m2 folder if there are any it will download it and then updates the .m2 folder and also if there are any changes happen on dependency version it will update to import new version of that jar file.

Maven Test: This command will run tests under the src/test/java source folder .

In our framework to store application property we will create a file with extension .properties this file is key and value based. It means we will store values and get them by keys. The benefit of property file is to prevent hard coding inside the project and if any changes come we can modify the value in property file. For example url of application. If the url changes we just modify the value in our properties file

Cross browser testing is to run all test cases against different browsers, such as chrome, FF, IE, Edge.

For logger we use log4j.properties file to store log properties.

W create Base class to store property of Webdriver, log4j , properties class and all child classes will inherit these properties from parent class using extend keyword and that is concept of Inheritance one of major OOPS concept used in this framework.

What are OOPs concepts and give us an example of their usage in your framework?

October 13, 2021

We add dependencies based on our need in our POM.xml file. For example if need to add a new libraries then we just get the xml code from maven central repository and add it under <dependencies> tag but outside the <dependency> tag.

Test runner is Engine Start of BDD framework which is based on Cucumber. We create a blank TestRunner class then on top of class name we use @RunWith annotations of Junit framework to run our Test cases. Cucumber provides @CucumberOptions and inside the cucumberOptions we have some properties to define how Cucumber will read and connect feature files with Step Defintions. We have feature where we provide path of feature files,

Glue where we provide path of StepDefinitions classes, we have dryRun to check if any steps in scenario is missing an associated step in StepDefiniton class.

We follow POM Design Pattern and the way we do that is, we create separate Java classes to store UI elements as Private Webelements and access them through public methods. We use PageFactory.initELements method to initialize all UI Webelements in this POM class.

pageFactory class provides an annotation @FindBy which is the same as driver.findElement() but with @FindBy annotation we store UI elements or it will initialize the elements whenever we call them from stepDefinitions class.

Interview Questions:

What is POM Design pattern and its benefits?

What is PageFactory ? and what is difference between POM and PageFactory?

What are key components of Cucumber?

What is feature file ? and what is feature file extension

What is scenario and scenario outline?

What is cucumberOptions?

What is hooks in cucumber?

Lab session Notes: 10/14/2021

The flow for automating a test case using BDD framework with POM Design pattern is in 4 steps.

* Step 1: we create a feature file if there is no feature existed for this scenario. Each feature file starts with word Feature: and name of Feature, Then we write our scenario and scenario steps using gherkin keyword Given, When, Then

Example of Feature file

Feature: Test Feature for Lab

@test

Scenario: Test case

Given user is on Retail website

When user search for 'iphone'

Then iphone should be displayed

* Step 2: We create a Step Definition class for this feature and extends Base class so we can have property of Base class. Then we start writing associated methods for each step of Scenario steps and write @Given(“^Name of Scenario Steps $”) then under this line we write a method to execute the block of code for this step. For example of Step Definiton:

@When("user search for {string}")

public void user\_search\_for(String string) {

// Write code here that turns the phrase above into concrete actions

testPageObject.enterValueForSearchBar(string);

testPageObject.clickOnSearchButton();

}

@Then("iphone should be displayed")

public void iphone\_should\_be\_displayed() {

// Write code here that turns the phrase above into concrete actions

Assert.assertTrue(testPageObject.isIphoneDisplayed());

}

* Step 3: We create separate Java classes to store UI elements as private Webelment and access them with public methods. In our POM classes we create a constructor and inside the constructor we write PageFactory.initELement() method to initialize the webelements of this class.

For example:

public TestPageObject() {

PageFactory.initElements(driver, this);

}

@FindBy(xpath = "//input[@name='search']")

private WebElement searchBar;

@FindBy(xpath = "//button[@type='button' and @class='btn btn-default btn-lg']")

private WebElement searchButton;

@FindBy(xpath = "//img[@title='iPhone']")

private WebElement iphoneImage;

public void enterValueForSearchBar(String value) {

searchBar.sendKeys(value);

}

public void clickOnSearchButton() {

searchButton.click();

}

public boolean isIphoneDisplayed() {

if (iphoneImage.isDisplayed())

return true;

else

return false;

}

* Step 4: we can run this scenario from TestRunner class by writing a tag on top of the scenario and then pass that tag in our Tags in Test Runner CucumberOptions. if a tag is used on top of more than one scenario and scenario outline it will run all of those scenarios. It means we can have a single tag of @SmokeTest on top of 100 Scenarios in different feature files and when we run this tag from test runner it will execute all those scenario in alphabetic order of feature file then from top to bottom inside the feature file.

Class Notes 10/18/2021

* This Framework is based on maven build Management tool.
* The Design pattern is POM (Page Object Model) where we implement PageFactory Class with using @FindBy annotations to store UI elements.
* This Framework is a BDD (Behavior Driven Development) and based on Cucumber.
* In Cucumber we have three major components
  + Feature file where we write our scenario and scenario outline using Gherkin keywords Given, When, Then
  + Step Definitions classes where we write our actual java methods for implementations
  + Test Runner class where we run our scenario and scenario outline using cucumber Options tags.
* We have utility classes where we store our reusable methods.
* We have reporting part where we generate logs using log4j, screenshots, and cucumber io reports.

Class Notes 10/19/2021

In Cucumber we have cucumber DataTables, where we can send a list of data as input and the return type of Cucumber Data Table is List of List or List of Map. If columns have column names then we can use List of Map where rows will be stored in a List interface and Column values will be stored in Map Key and Value based. If columns does not have column names, then we can use List of List where we can give rows in a list interface and column value as index of List interface as well.

For example

Given User is on Website

When User enter personal Information with below data

|firstName|lastName|email|phone|password|

[|alex|conor|alexconor@gmail.com|2021011122|wrongPassword|](mailto:|alex|conor|alexconor@gmail.com|2021011122|wrongPassword|)

Then user create an account

Given User is on Website

When User enter personal Information with below data

[|alex|conor|alexconor@gmail.com|2021011122|wrongPassword|](mailto:|alex|conor|alexconor@gmail.com|2021011122|wrongPassword|)

Then user create an account

What is Map and have you used it in your framework?

Successful

Git and Git hub: Version Control or also known as source control management. Is one of the ways to store and track changes in software code. In Automation we are using source control tool to maintain and share framework codes.

October 25, 2021

TestNG: is an open source Testing Framework which inspired from Junit but with more annotations and it helps testers to make it easer with grouping, parallel execution and parametrizations.

In TestNG every test cases is marked with @Test, if you see @Test on top of a method it is a test. And we can have multiple tests in a java class.

In TestNG we have two ways to supply data to test cases.

1. Trough Parameters and we have to use @Paramaters and provide data from testNG.xml file
2. Through Data Provider annotations.
   1. Data Provider is used to supply data either from inside the data Provider or Excel sheet as input data file. The concept of Data Driven Framework is coming with using of Excel as Input file and running same test case with multiple set of data. It means we can test same functionality of application with different set of data like Scenario outline in cucumber BDD framework. In order to use Excel as an input we need to use apachePoi library to read from excel sheet.
   2. When we are reading data from excel these are steps need to follow:
      1. First we need to provide path of excel workbook
      2. Second we need to provide name of excel sheet where data is stored
      3. Third one we need to write nested loop to iterate through the rows and columns of excel sheet to get the data. Inner loop will travel through the columns and outer loop will travel through the rows.

In TestNG we have assertions: Assertions are basically a way to check if expected result meets actual result. In TestNG we have to two types of Assertions.

1. Hard Assert: execution stops right at the assertion point if actual does not meet the expected result.
2. Soft Assert: execution will not stop if actual does not meet the expected result but it will show test case as failure.

What is difference between Assertion and Verification?

In Assertion if actual does not meet expected result, it will stop execution and fail right away. While verification the execution will continue till end of the test then it will mark test case as failed one.

Parallel Execution: means we should run different set of test cases at the same time, Selenium Grid is main tool for parallel execution but TestNG does support parallel execution as well.

Cross browser Testing: means we should run same test cases in different browsers, so all our test cases should pass in different browsers such as Chrome, Firefox, IE, Edge.

October 27, 2021

What is difference between BDD and Data Driven Framework? And why you prefer one over another?

BDD framework is based on Cucumber and Junit framework where we use Gherkin language to write our test scenarios in feature file, we use BDD framework when we want to test behavior of application based on User acceptance criteria from Business. One of the advantages of BDD framework is that we have feature file where it is written in plain English language and non-technical members of team can understand the test scenarios and what scenarios are covered in this feature.

Data Driven framework is based on TestNG and plain java and selenium coding, where we test same functionality of application with multiple set of data, in this framework we are using Excel an input data with help of Apache POI libraries to read data from excel and test it. This framework is pure coding and it requires coding knowledge for non-technical users to understand the test scenarios and test coverage.

If team is an agile where the interaction is more between BA, Dev and QA then BDD would be a good fit for testing framework, if there is no agile team and it is waterfall testing model, then Data driven framework based on TestNG would be good fit as it does not require too much of maintenance and easier to work on it.

In Sprint Automation and out of Sprint Automation: 20%-30%

If you work with team of Agile and you have sprint of 2 weeks then the automation should be done within the sprint. What it means? It means when a user story comes to sprint backlog and developer starts developing it, we as a QA automation engineer start writing Scenarios based on acceptance criteria from user story, then we write our step definitions, and once story is ready for QA then we start creating POM classes to store UI elements and once it is in QA then we start executing our test cases from Test Runner using Cucumber Tags.

Out Sprint Automation: 60% - 70%

QA automation engineer is not part of scrum team but mainly focusing to automate the existing manual test cases that are already passed and documented. So, we mainly focusing to increase automation percentage of Regression suite from manual to Automation and it is usually 1-2 test cases per day.

TestNG Listeners implements ITestListener interface which provides onTestStart method, TestSucces method, TestFailed method, TestSkipped method and once we implement the interface then we can write implementation codes to log, and write on extent report of TestNG.