## Pre-req:

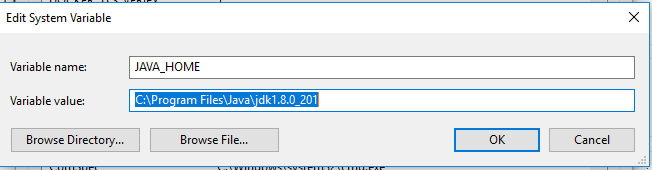
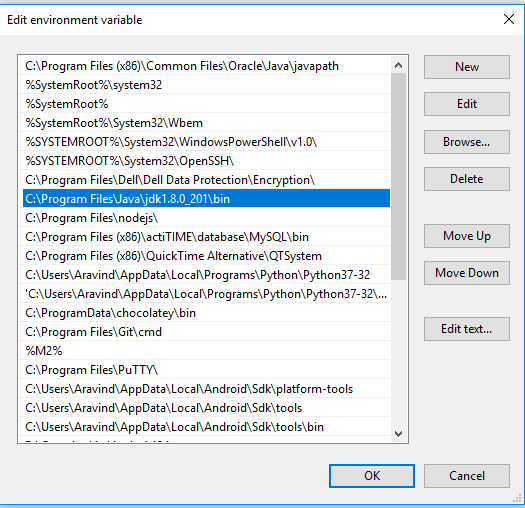
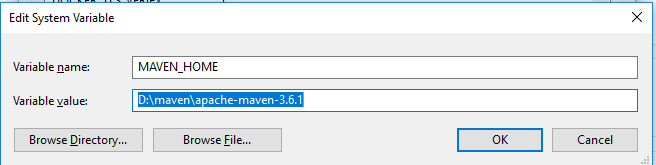
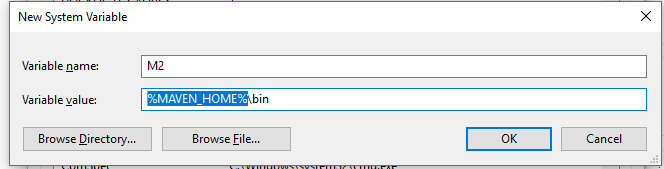
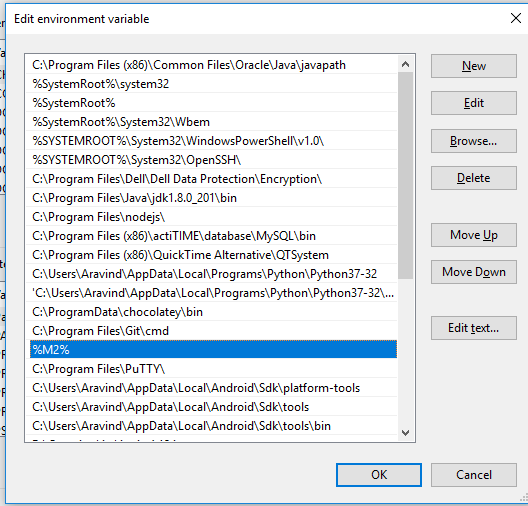
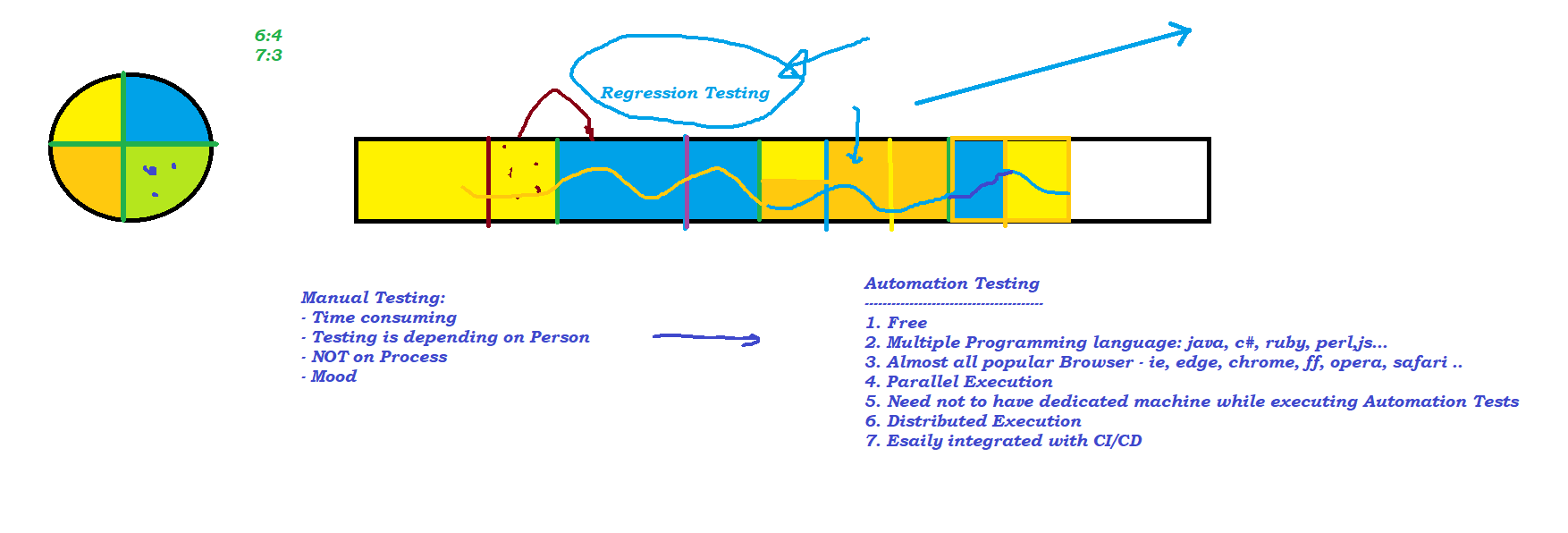
JAVA

1. Basic Syntax of Java – Class, packages, interfaces
2. Data types
   * Primitive data types
   * Derived data types
3. Variables
   * Global variables
     + instance variables
     + static variables
     + constants
   * Local Variables
4. Methods
   * with return type
   * without return type
   * with Arguments
   * without Arguments
   * Overloading
   * Overriding
5. Constructors
   * Default
   * parameterised constructor
6. Access specifiers – private, protected, , public
7. Access Modifiers – static, final, abstract
8. String Class
   * functions – length(), charAt(index), indexOf(char), split(), substring()...
9. String Buffer / String Builder
10. Arrays
11. Enums
12. Wrapper Classes
13. Threads
    * Thread
    * Runnable
14. Exception Handling
    * Compile Time Exception
    * Runtime Exception
15. File Handling
    * Text file, properties file, csv, excel,doc, pdf..
    * Reading File
    * Write into the File
16. Collections
    * List
      + ArrayList
      + LinkedList
      + Vector
    * Set
      + HashSet
      + TreeSet
      + LinkedHashSet
    * Map
      + HashMap
      + TreeMap
      + LinkedHashMap
17. Generics
18. Inner classes
    * anonymous inner classes

Selenium :

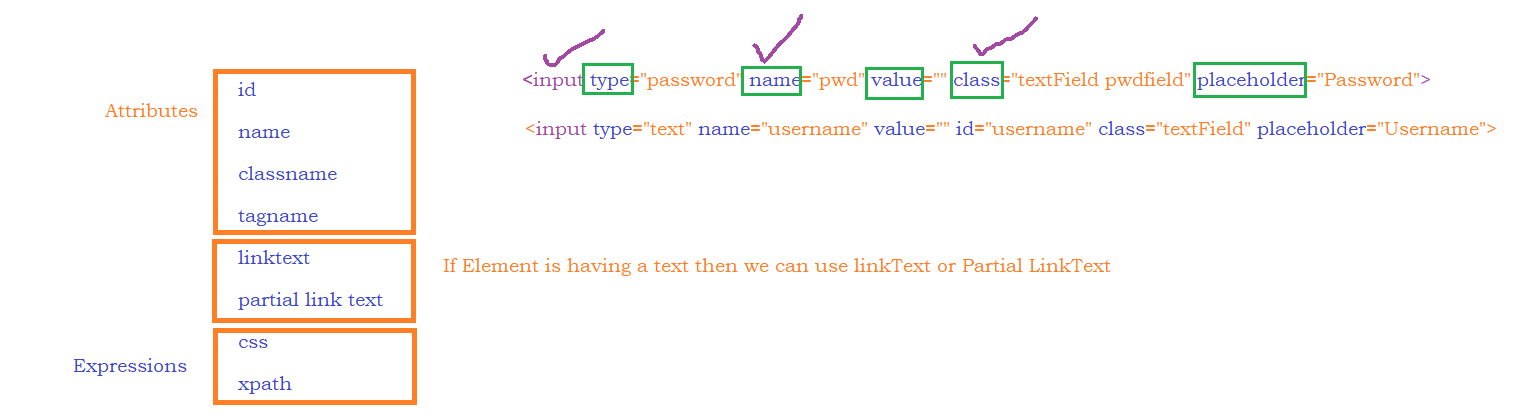
* What is regression Testing
* Why Automation Testing is required
* Why Selenium???????
* Selenium IDE
* ~~Selenium RC~~
* Selenium WebDriver
  + Major
* Selenium GRID
* TestNG
* POM
* Maven
* BDD Cucumber
* CI/CD –
* GIT

Software Required :

* Java
  + Setting Environment Variables
  + JAVA\_HOME
    - 
  + PATH
    - 
* Eclipse
* Maven
  + <https://maven.apache.org/download.cgi>
  + set the Environment Variables
  + MAVEN\_HOME
    - 
  + M2
    - 
  + Updated PATH
    - 
* Application
* Jenkins
* GIT
  + <https://git-scm.com/downloads>
* What is regression Testing
* Why Automation Testing is required
* Why Selenium???????
* 

Selenium IDE

* POC
* Record and playback
* Add-on with FF and Chrome
* Object Identification
  + id
  + name
  + classname
  + tagname
  + linktext
  + partial link text
  + css
  + xpath



SAMPLE HTML CODE:

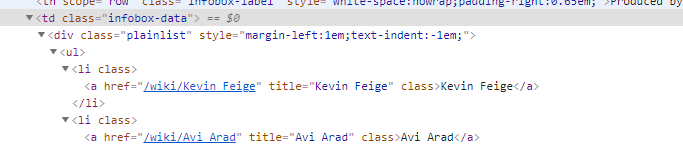


Expressions:

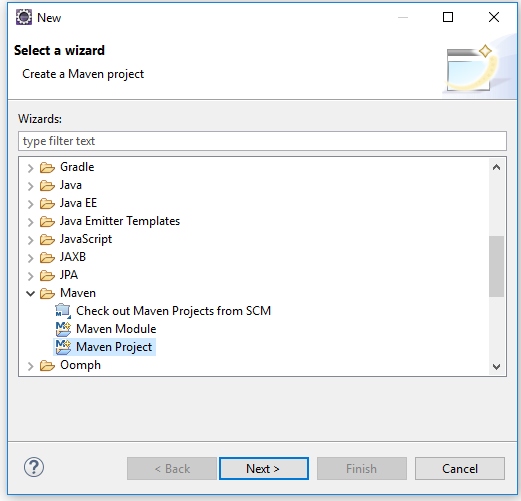
CSS:

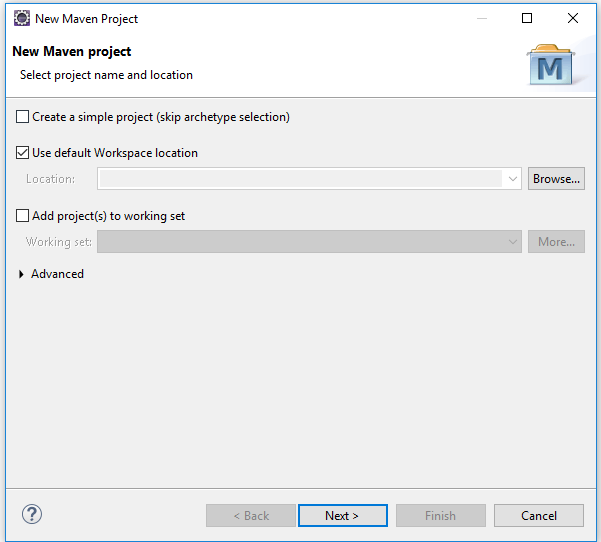
* htmltag[attribute=’value’]
* input[placeholder='Username']
* htmltag#IDvalue OR #IDValue
* htmltag.ClassValue OR .ClassValue
* Parent to child:
  + #loginButton > div

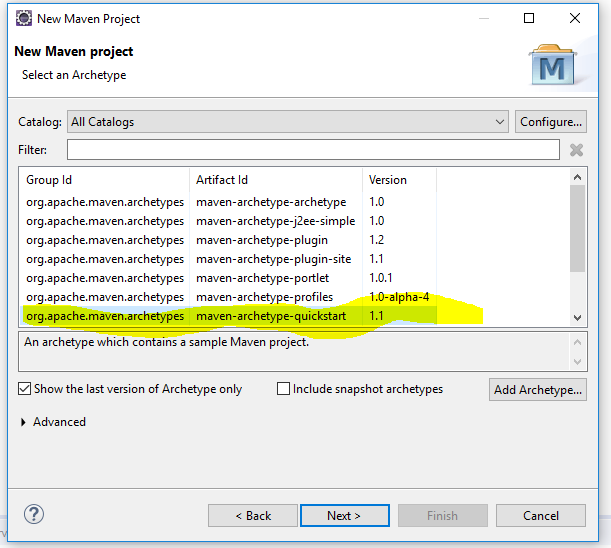
XPATH:

1. **Basic**
   1. Formula - //htmltag[@attribute=’value’]
      * 1. //input[@id='username']
        2. //input[@placeholder='Username']
2. **Function**
   1. text()
      1. Formula - //htmltag[text()=’exact text’]
         1. //div[text()='Login ']
         2. //label[text()='Keep me logged in']
         3. **//a[text()='Autocomplete']**
         4. **//a[text()='Autocomplete' and @class='btn btn-lg']**
   2. contains(arg1,arg2)
      1. arg1 – attribute or text function
         1. //img[contains(@src,'timer')]
         2. //td[contains(text(),'Please')]
      2. arg2 – partial value
   3. starts-with(arg1,arg2)
      1. arg1 – attribute or text function
         1. //\*[starts-with(@id,'ext-gen')]
         2. //h3[starts-with(text(),'About')]
      2. arg2 – partial value
3. **Logical Operator**
   1. and
      1. Formula - //htmltag[@attribute1=’v1’ and @attribute2=’v2’]
      2. //td[(@class='current day' **or** @class='we day' **or** @class='wd day') **and** text()='10']
      3. //input[@type='text' **and** @type='password']
   2. or
      1. Formula - //htmltag[@attribute1=’v1’ or @attribute2=’v2’]
      2. //td[(@class='current day' **or** @class='we day' **or** @class='wd day') **and** text()='10']
      3. //input[@type='text' or @type='password']
   3. not
      1. Formula - //htmltag[@attribute1=’v1’ **and** **not** (@attribute2=’v2’)]
      2. //td[text()='12' **and not** (@class='past day')]
4. **Traverse from parent to child**
   1. //a[@id='loginButton']**/**div
   2. //td[@class='keepLoggedIn']**//**div[text()='Login ']
5. **Traverse from child to parent**
   1. //parente2[parentTag[childXpath]]
   2. *//tr[th[text()='Produced by']]//li*
6. ***AXES Functions***
   1. ***Traverse to following-sibling***
      1. Syntax:
         1. //xpath for ele/following-sibling::siblingTAG
         2. //th[text()='Produced by']/following-sibling::td//li
   2. ***Traverse to preceding-sibling***
      * 1. //xpath for ele/preceding-sibling::siblingTAG
        2. //li[a[span[text()='Upcoming']]]/preceding-sibling::li
   3. ***Traverse till end of the page from any location***
      1. //li[a[span[text()='Upcoming']]]/following::a
   4. ***Traverse till beginning of the page from any location***
      1. //li[a[span[text()='Upcoming']]]/preceding::a
   5. ***Traverse to child***
      1. //a[@id='loginButton']/child::div
   6. ***Traverse to parent***
      1. //div[text()='Login ']/parent::a
   7. ***Traverse to Ancestor***
      1. //a[text()='Kevin Feige']/ancestor::td[@class='infobox-data']
      2. 

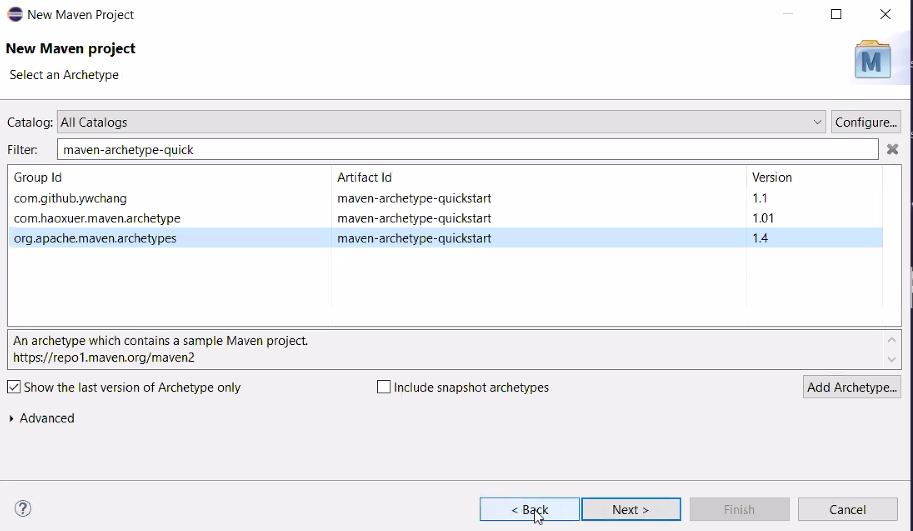
**WEB-DRIVER SETUP USING MAVEN:**

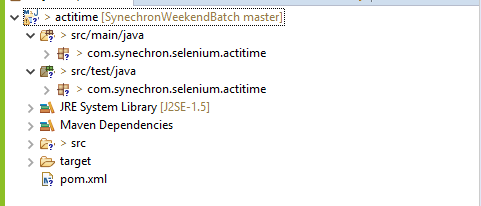
****





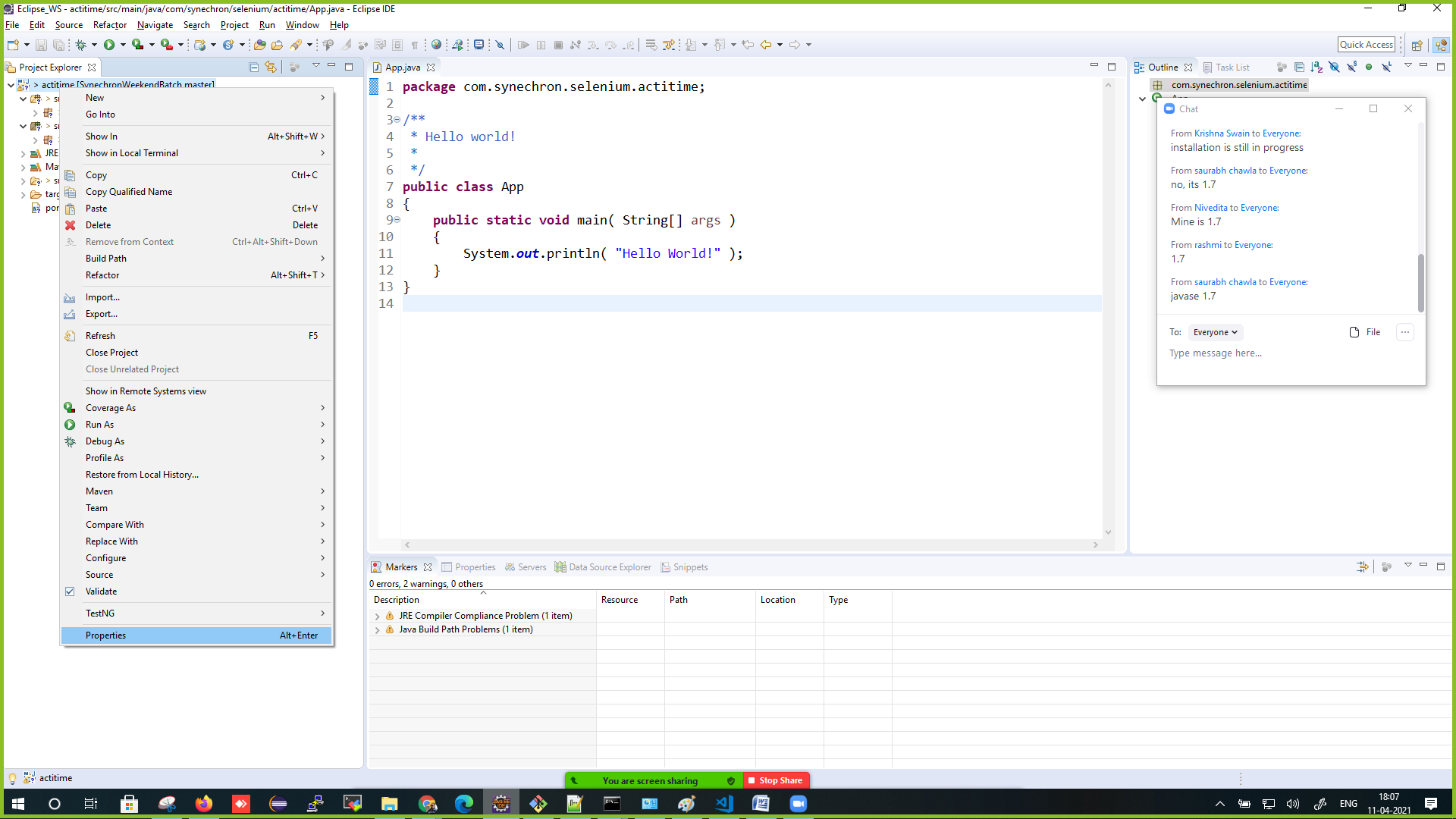
OR

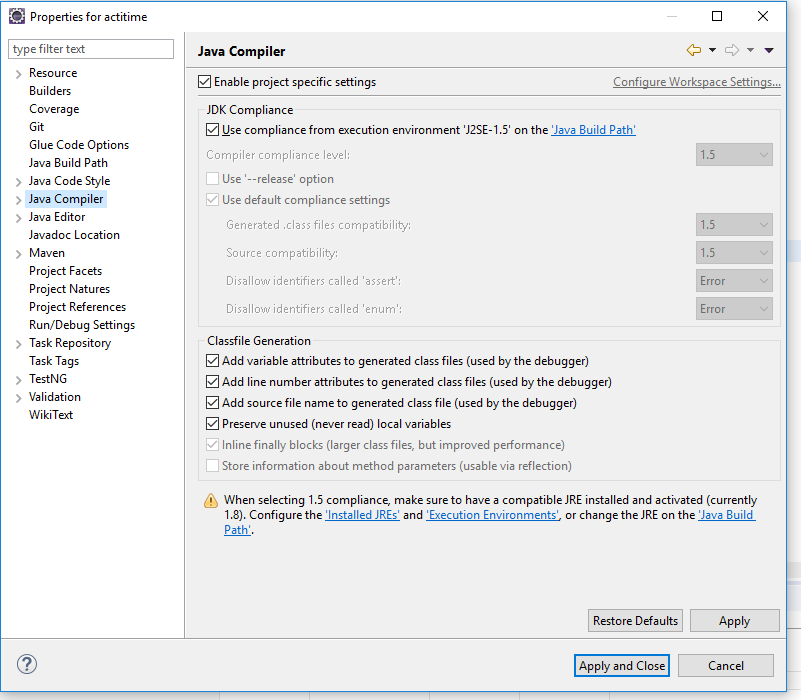


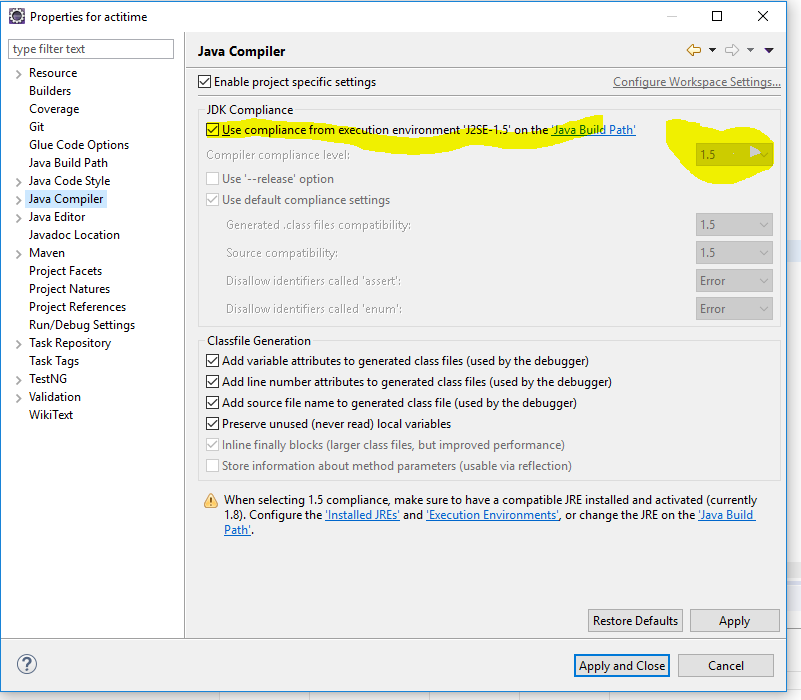


UPDATE COMPILER AND JRE VERSION IN THE PROJECT

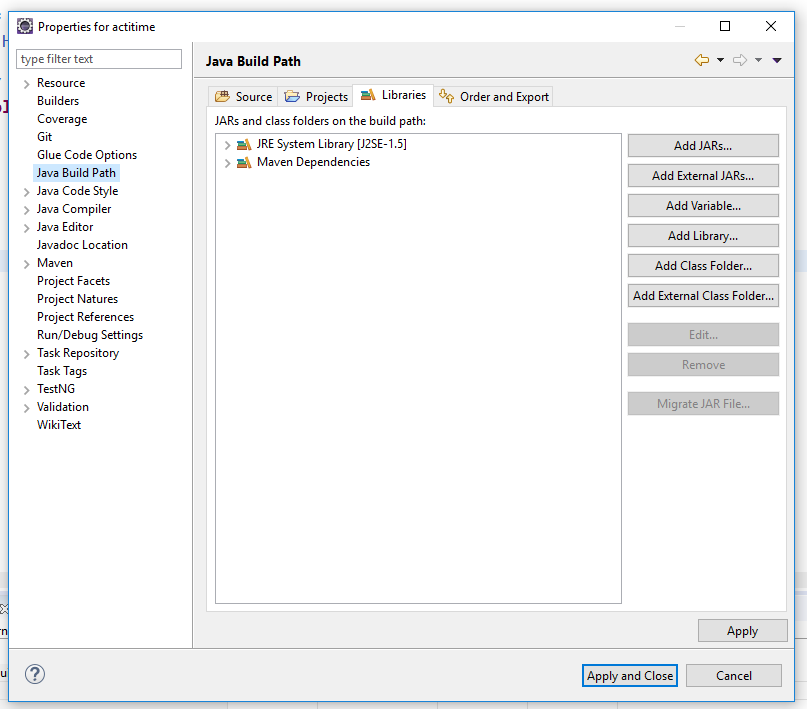
- COMPILER

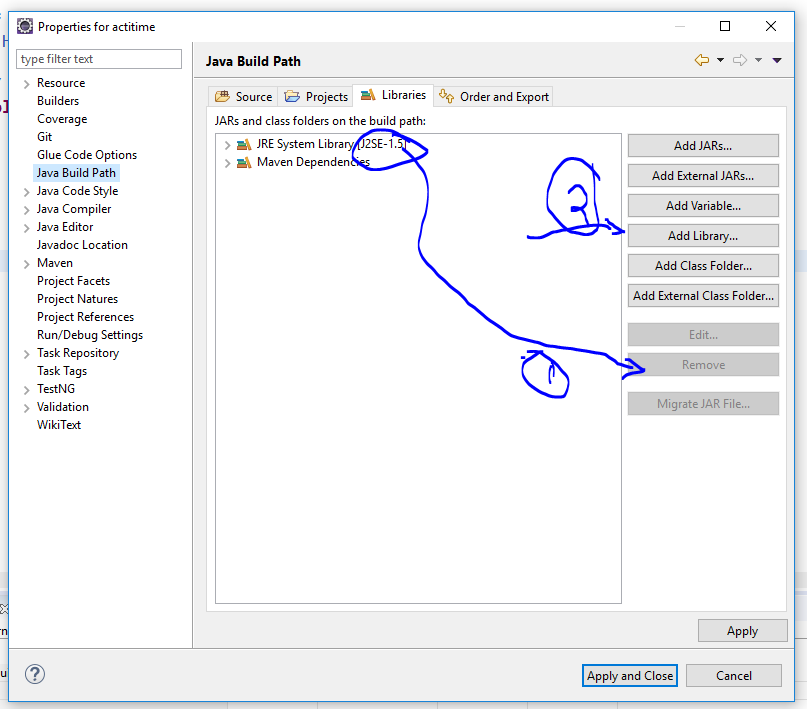


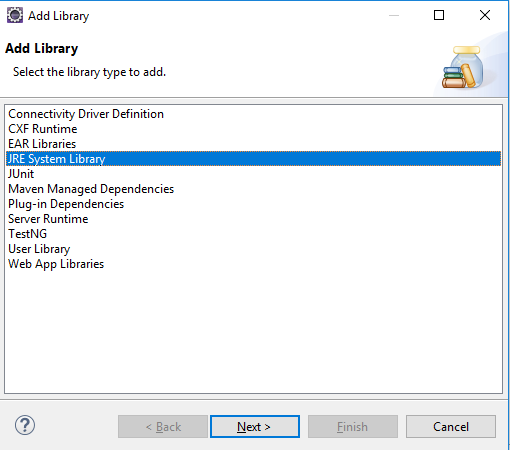


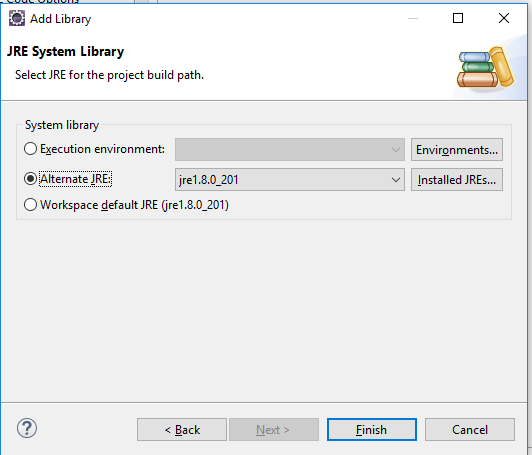


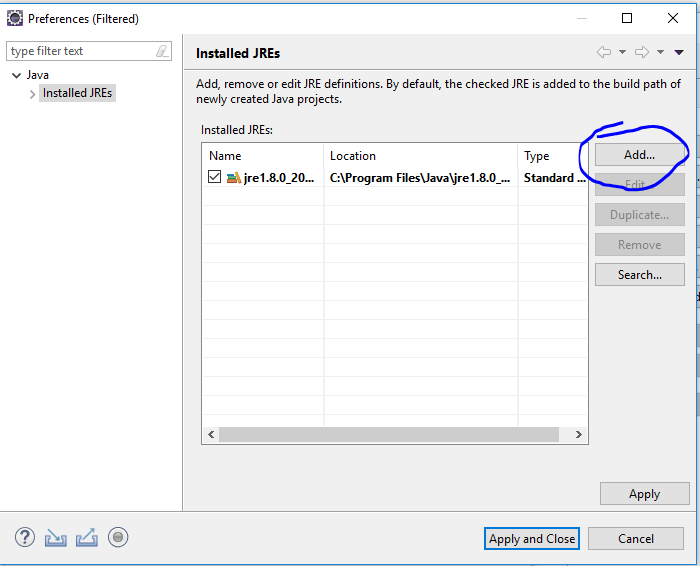
-JRE

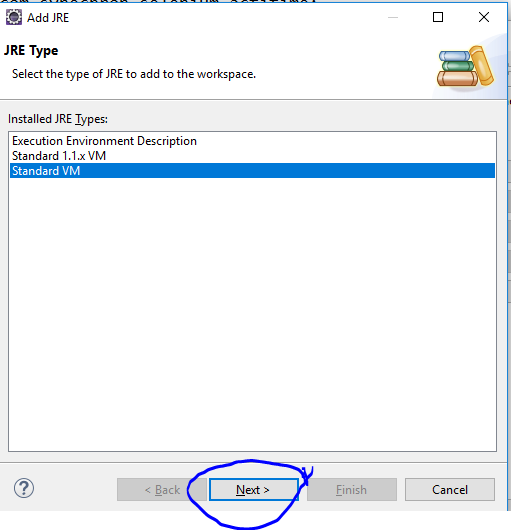


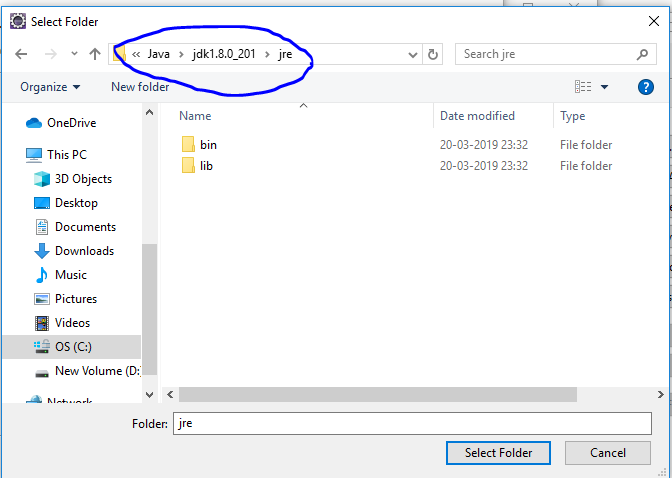




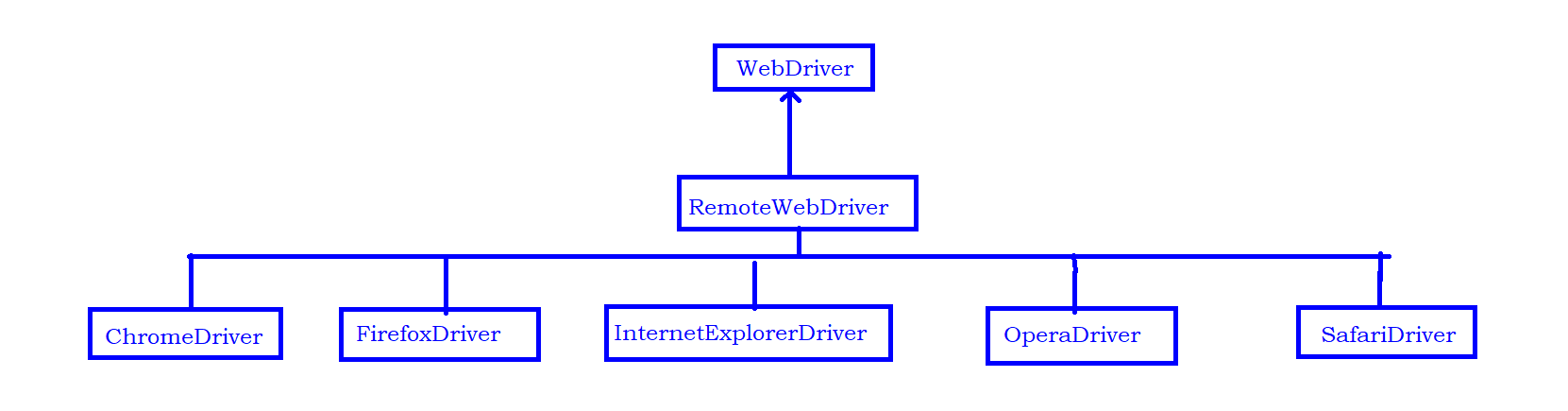




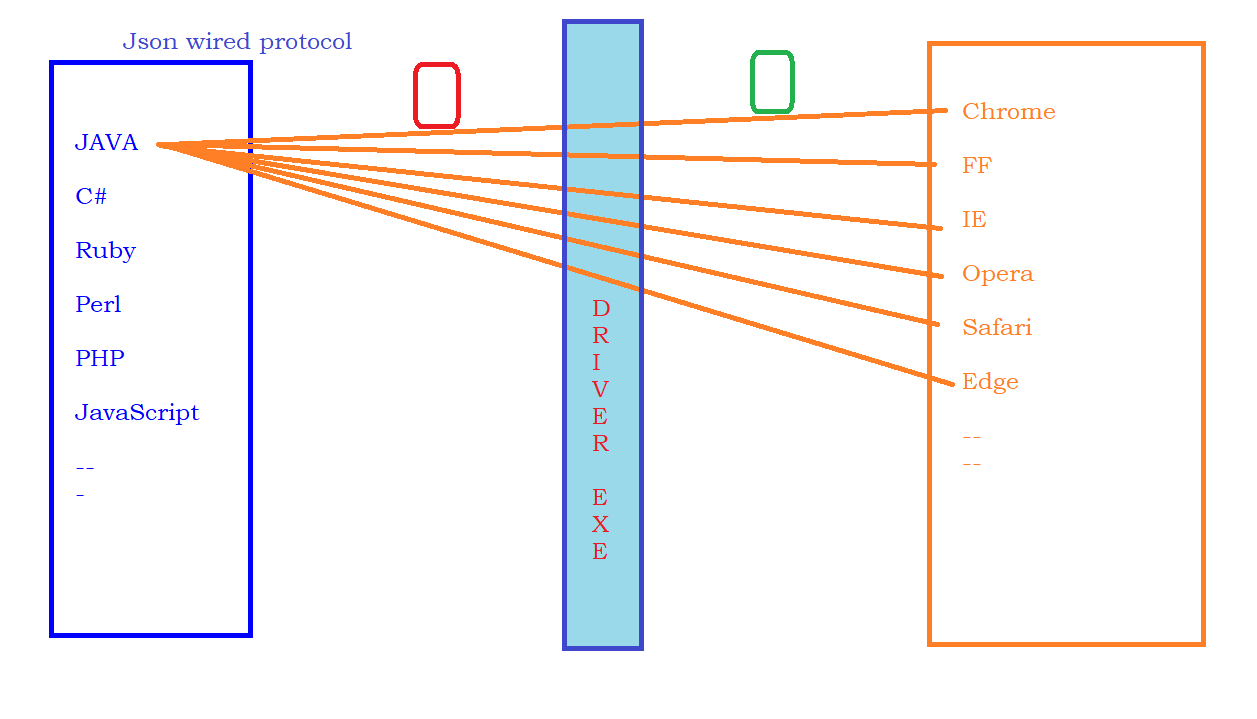




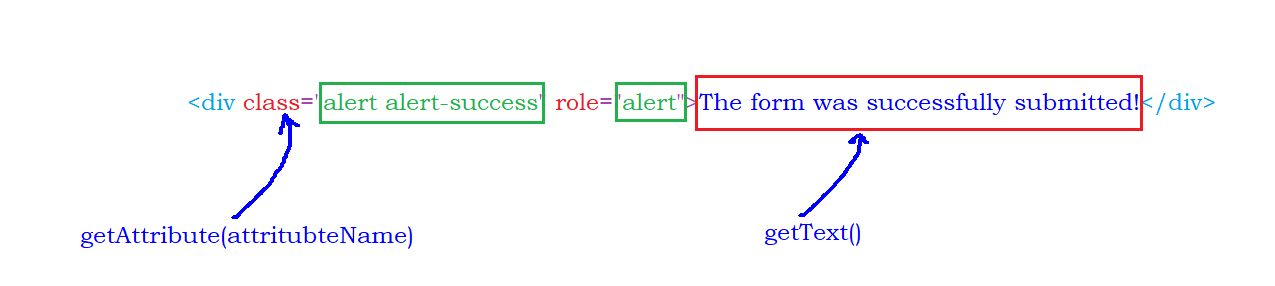
Architecture



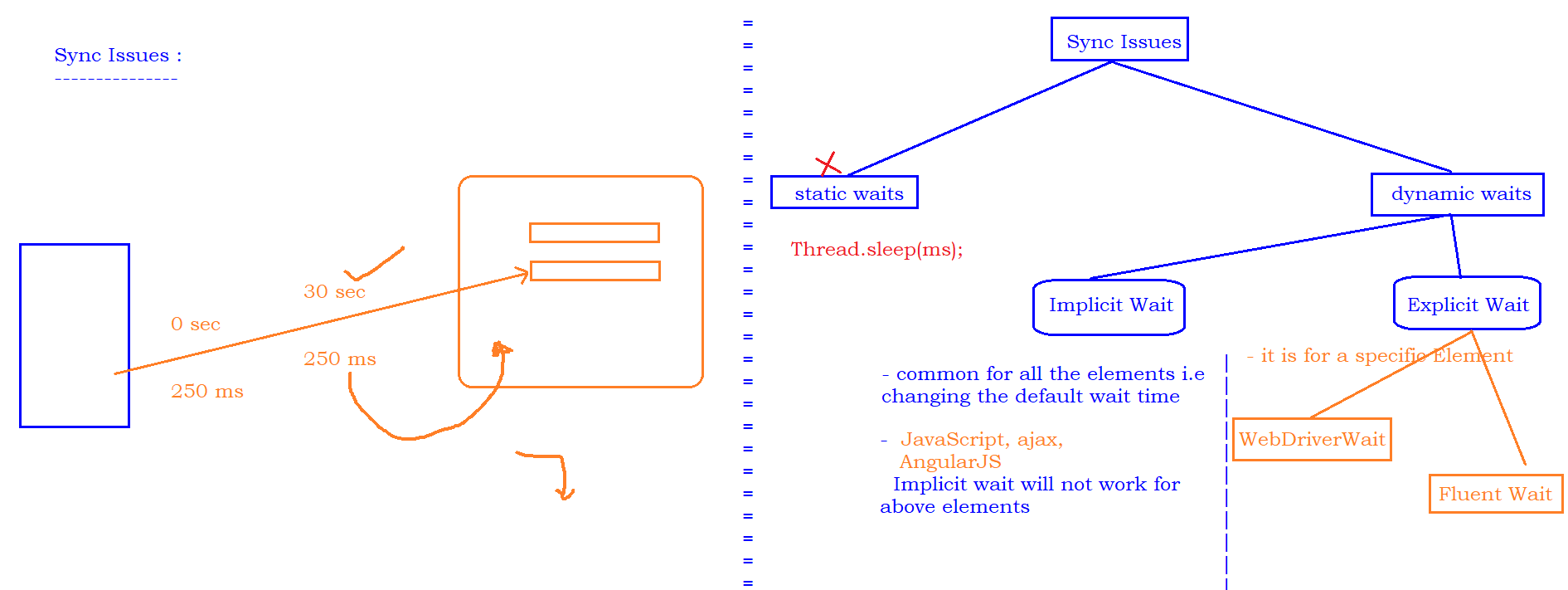
Json Wired Protocol



Reading Text or Attribute at Runtime:



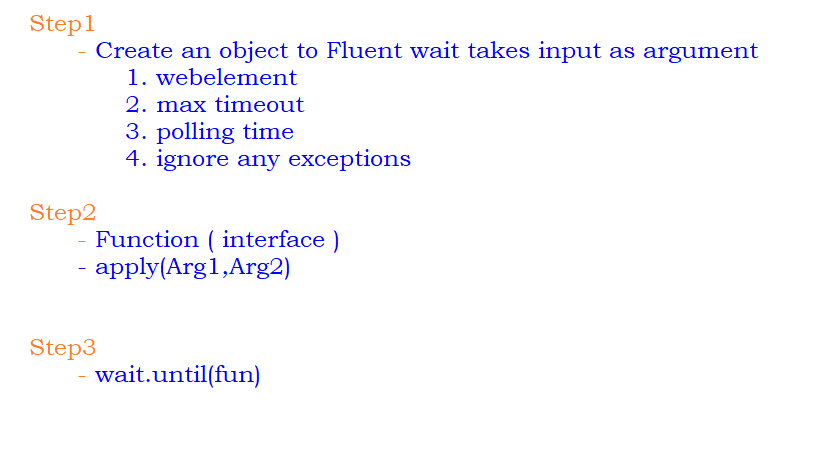
Sync Issues :



WebDriverManager :

<https://mvnrepository.com/artifact/io.github.bonigarcia/webdrivermanager/4.4.0>

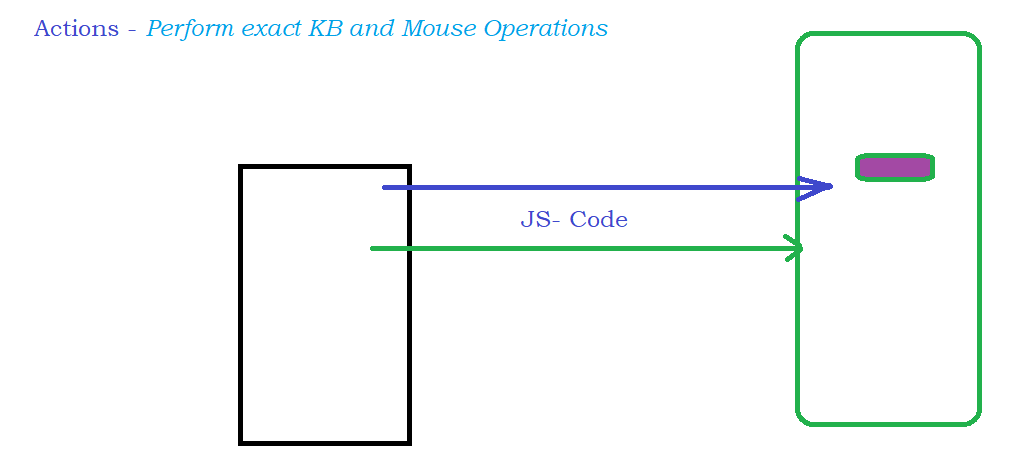
Fluent wait :



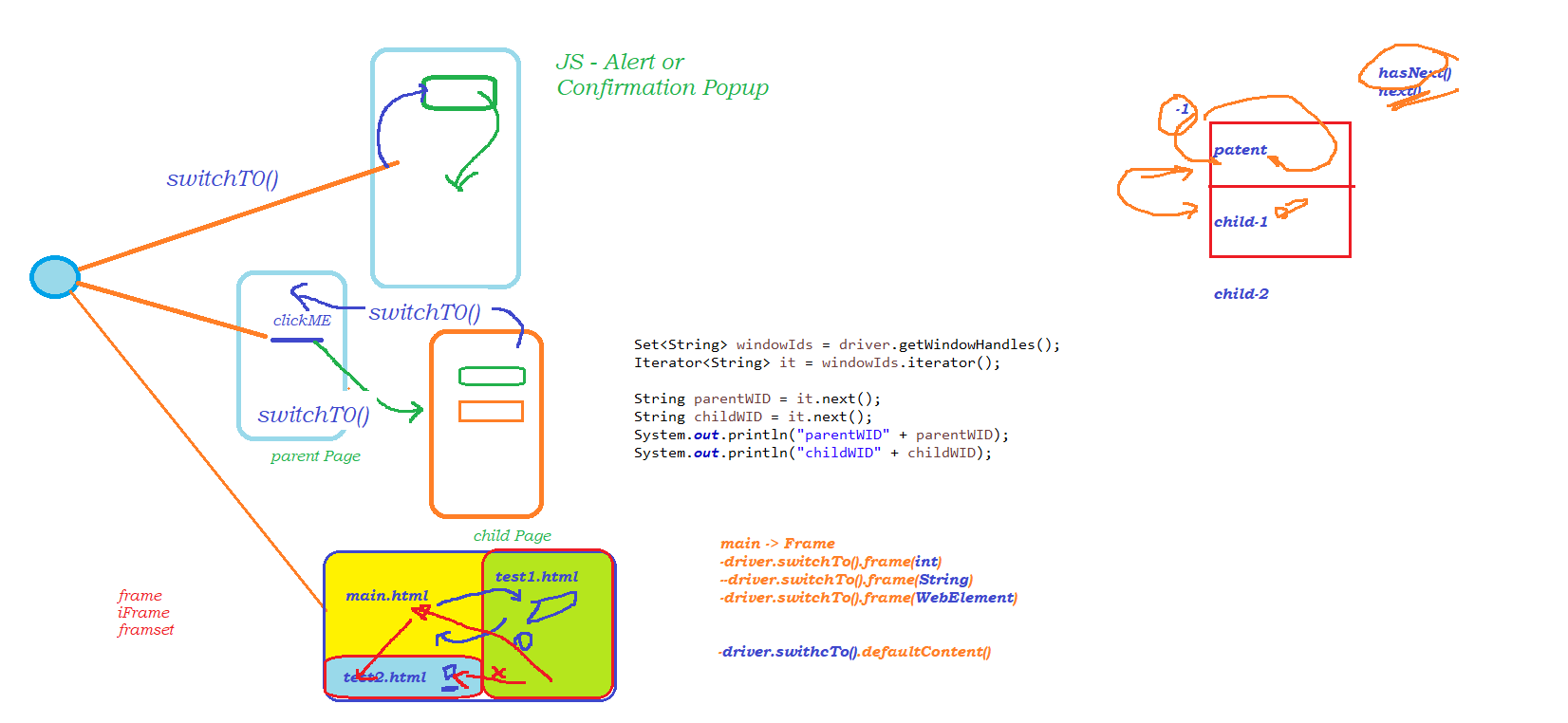
End-End Testing :

1. Execute the test manually (at least 2-3 times) and Identify ,
   1. Test Data required to execute the test
   2. Validations to be performed
   3. the reusable functions
2. Start creating required test data
3. Start writing reusable functions and handle all the validations inside the reusable functions.
4. start calling the functions one by one to complete the Test Automation.

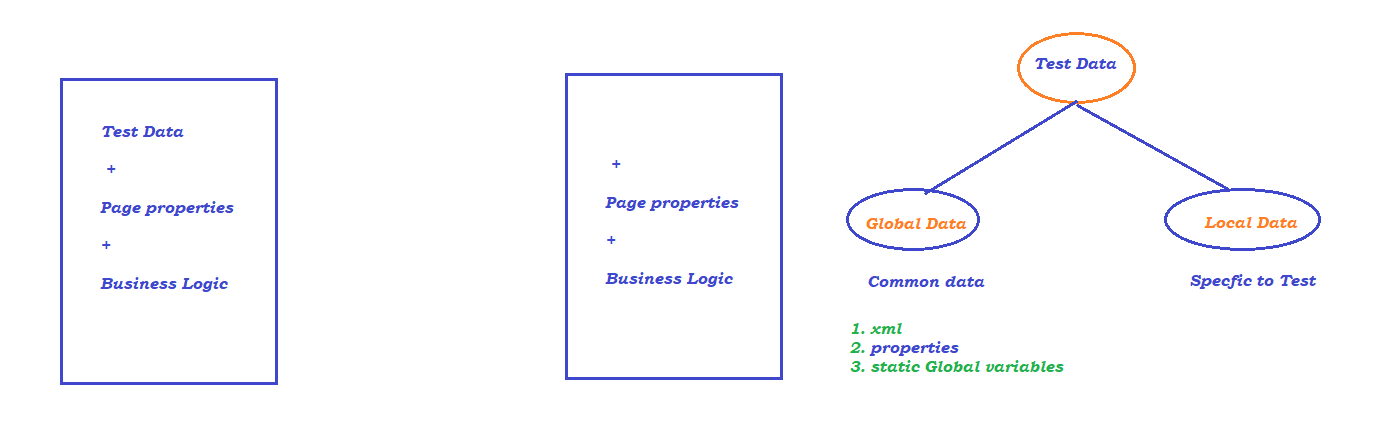
Actions in Selenium

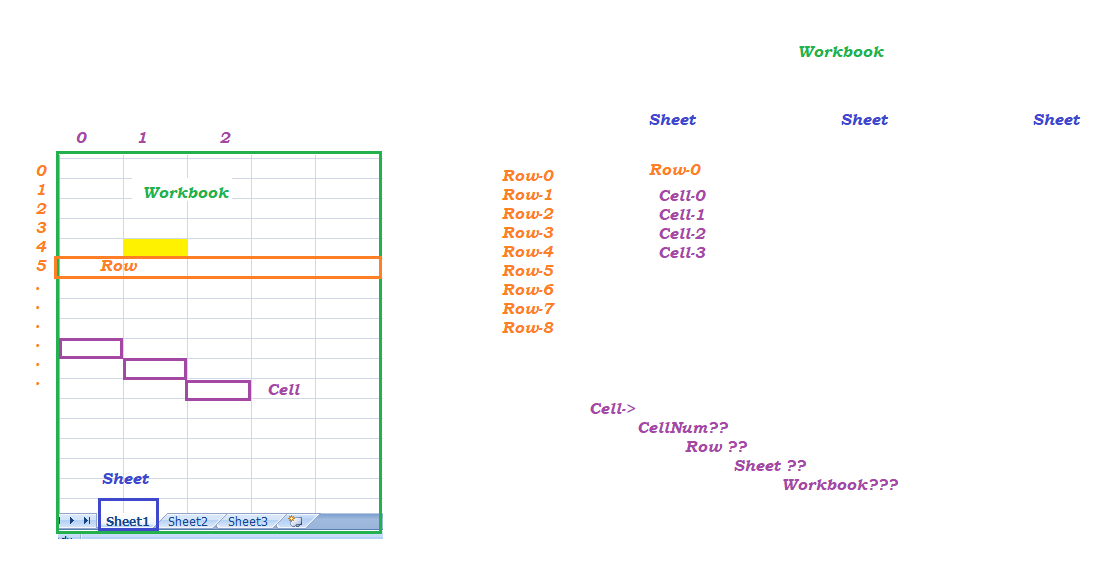


SwitchTo()

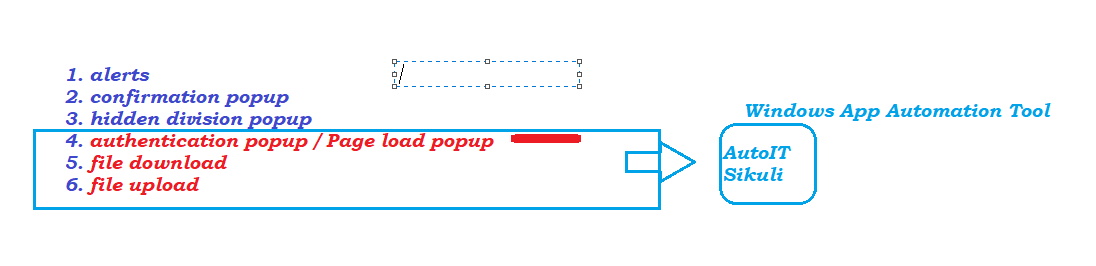


Data Driven Testing :





popup Handling in selenium :



Framework :

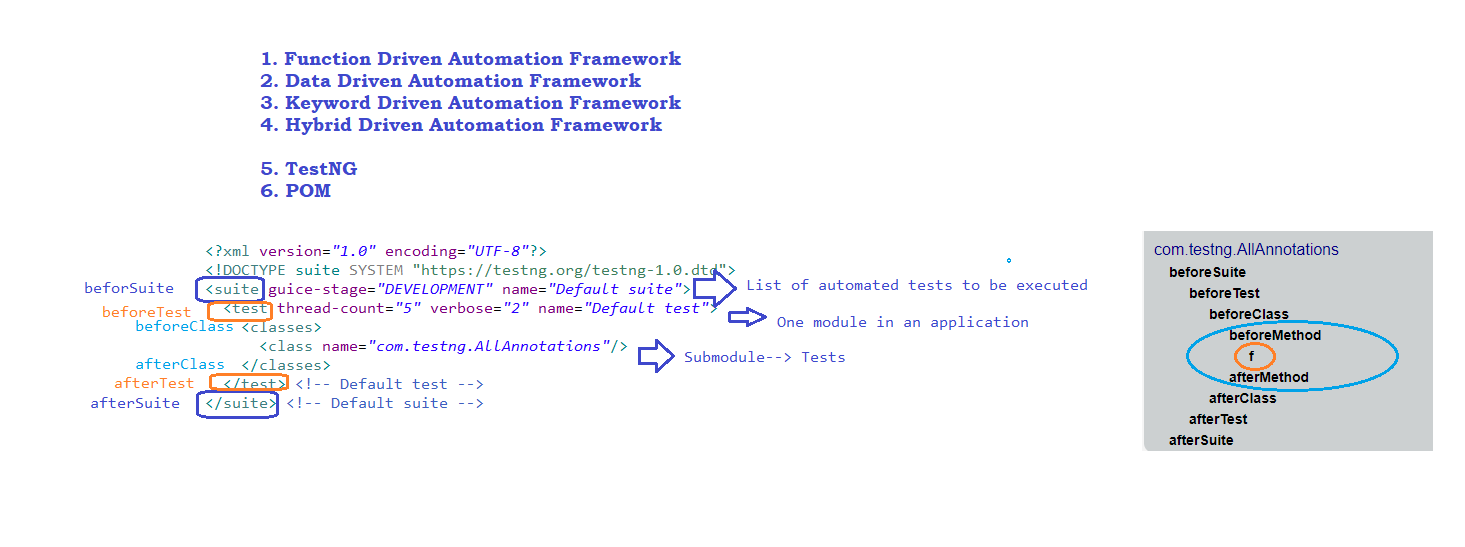
**1. Function Driven Automation Framework**

**2. Data Driven Automation Framework**

**3. Keyword Driven Automation Framework**

**4. Hybrid Driven Automation Framework**

**5. TestNG**

****

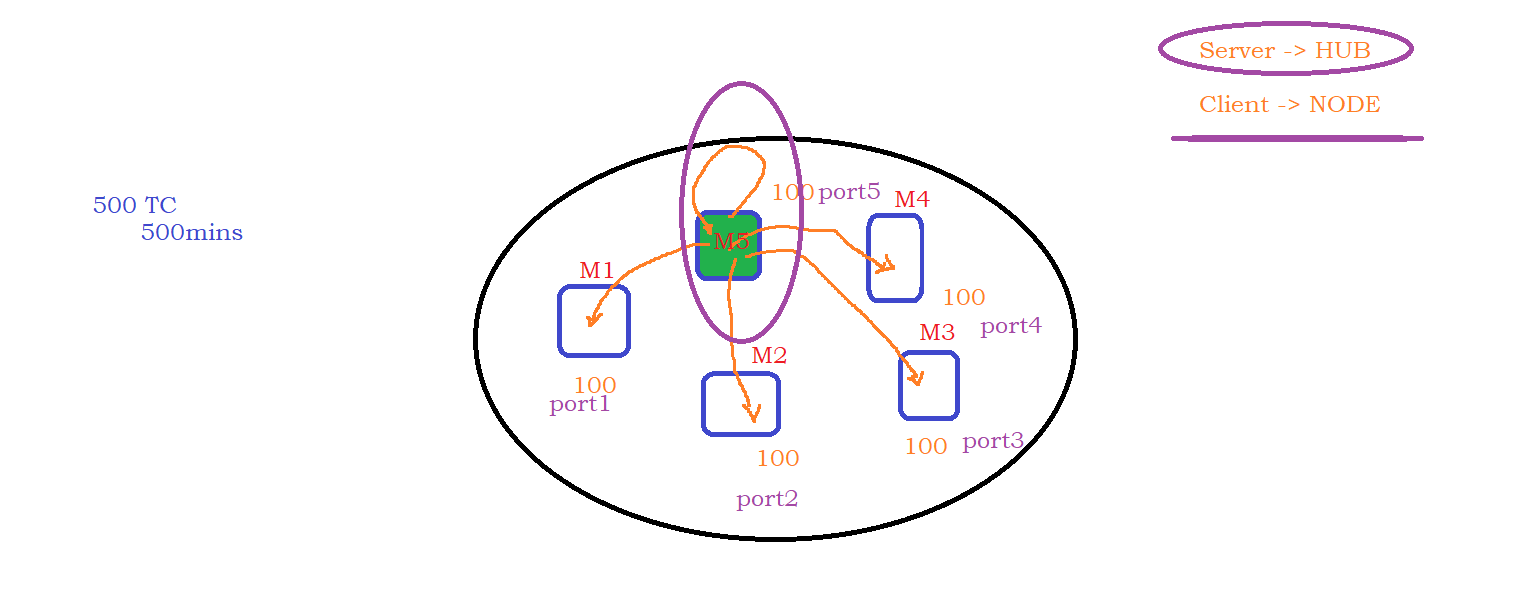
**Validation in TestNG :**

**- Asserts**

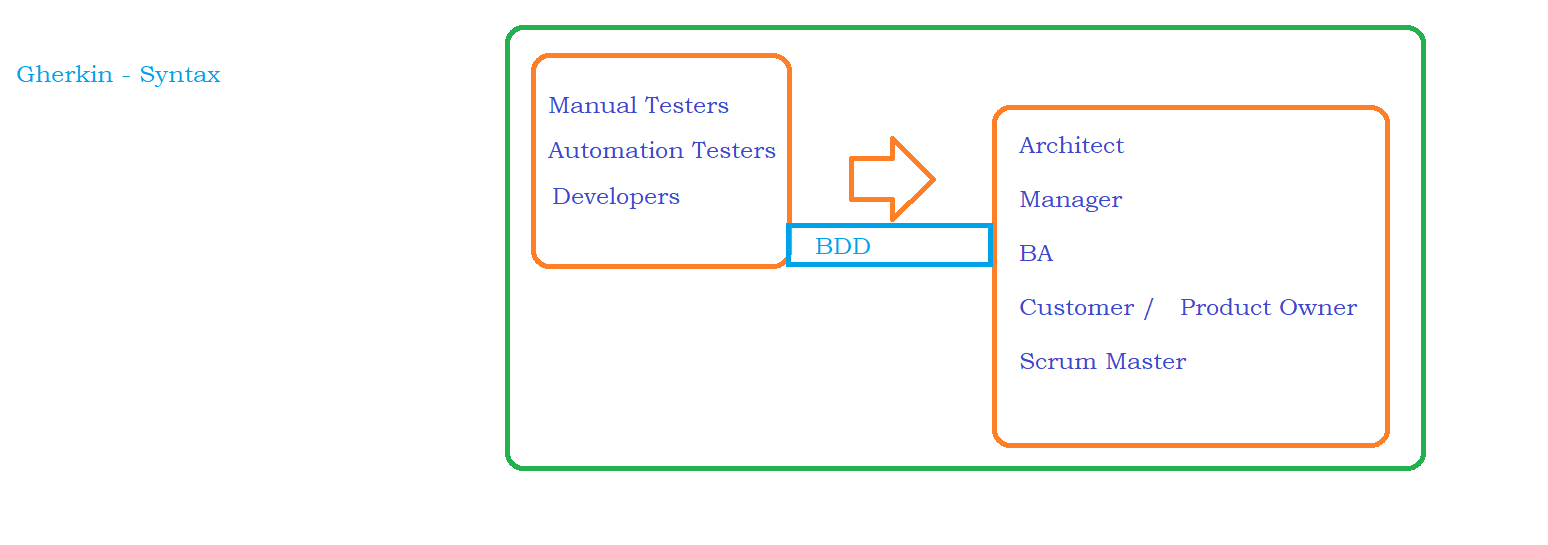
**- Soft Asserts**

**6. POM**

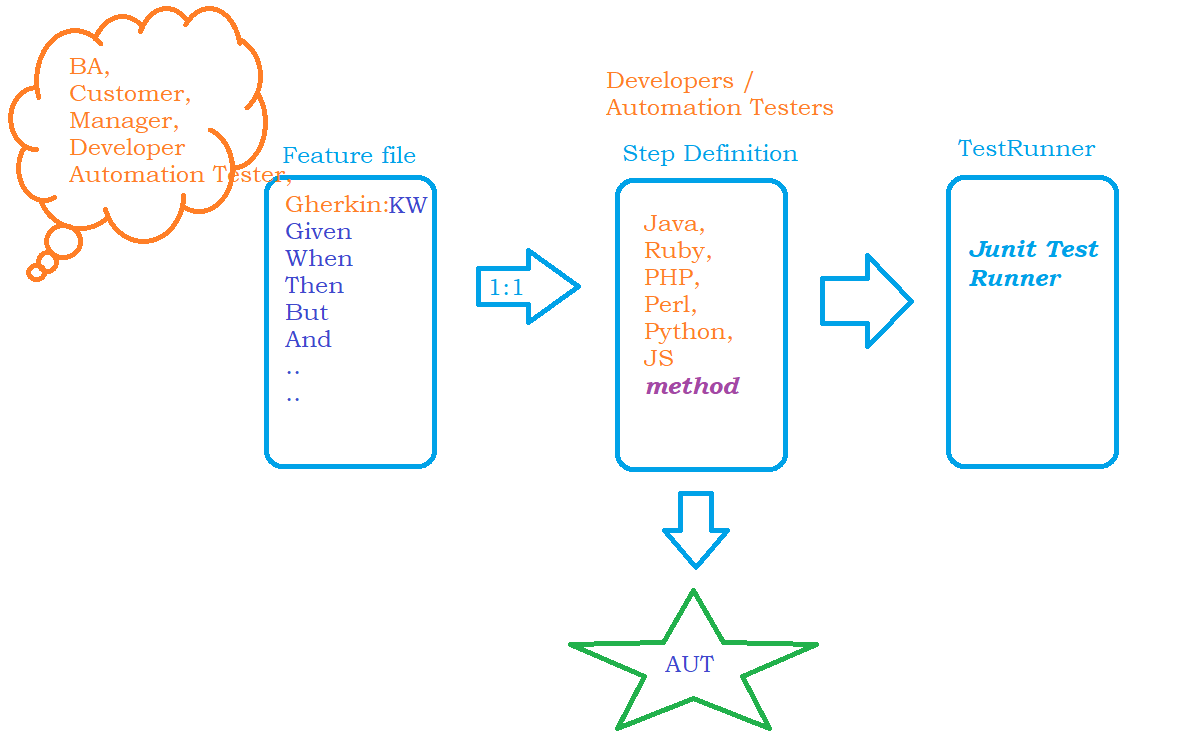
Selenium Grid :

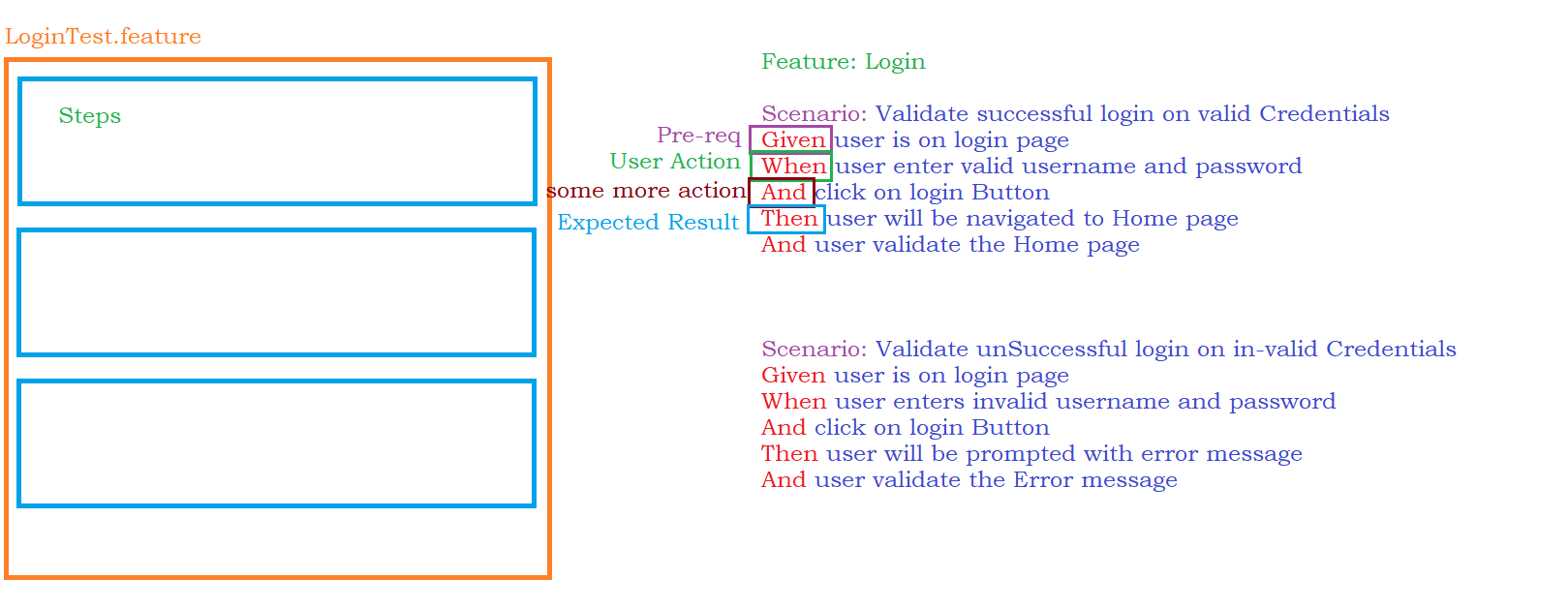


Cucumber :



Components of BDD

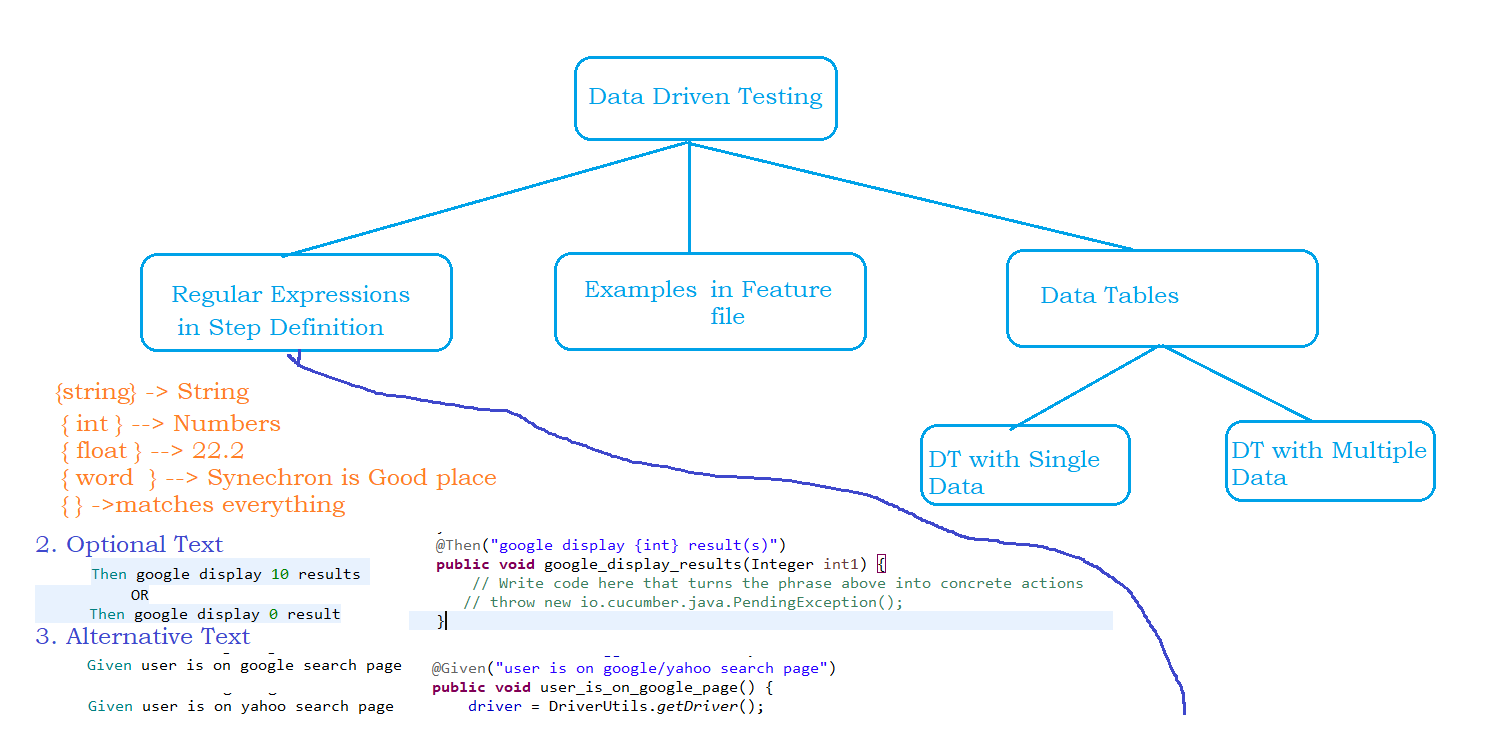




Setup Cucumber Project

1. Create a Maven Project
2. Add / Update the Dependencies
   1. cucumber-java
      1. <https://mvnrepository.com/artifact/io.cucumber/cucumber-java/6.10.3>
   2. cucumber-junit
      1. <https://mvnrepository.com/artifact/io.cucumber/cucumber-junit/6.10.3>
   3. junit
      1. <https://mvnrepository.com/artifact/junit/junit/4.13.2>
3. Install cucumber plugin in Eclipse – Help ->EMP ->search for cucumber
4. Write a new feature file ( file with .feature as extension )
5. Run the feature file
6. copy the missing step definitions from the console and put it in the step definition class
7. Implement all the step definitions with the appropriate programming language choose
8. To execute multiple feature files Create a Junit Test Runner file
9. 

DataDriven Testing :



1. 

## CONTENTS :

### Day -1

1. **Selenium WebDriver Components**

* Introduction to Selenium WebDriver
* Difference in RC and WebDriver
* Features of Selenium WebDriver
* Difference between WebDriver2.0 and 3.0

1. **Environment for Selenium WebDriver**

* Introduction to Eclipse
* Introduction to FireBug
* Practicals: FireBug Exercises
* Introduction to XPath, CSSpath
* Practicals: XPath Exercises
* Practicals: CSSpath Exercises
* Practicals: Security and Performance Testing using FireBug

1. **Process to Create Project and Create Selenium WebDriver Scripts**

* **Practical**: Create scripts to Automate Gmail Test Cases
* **Practical**: Create scripts to use Retrieve value from Web Site
* **Practical**: Create scripts to handle pop-up dialogs
* **Practical**: Create scripts to work with iFrames

### Day **-2**

1. **Selenium WebDriver Project – Build Selenium Java Scripts**

**Practicals**: Build script to automate Live Web site

Create scripts to emulate user interaction with Web site

* Typing text
* Taking action on an existing web element
* Observing if web element is accessible on Web site
* Working with web elements that have same name or id
* Working with web elements that have no name or id
* Navigating across web links
* Navigating across web pages
* Browsing Other sites and Returning to previous sites
* Browsing Other sites and Returning to specific site
* Create scripts to access within a Web Table

1. **Selenium WebDriver Project – Build Selenium Java Scripts**

Synchronization

* Conditional Synchronization
* Unconditional Synchronization
* Implicit and Explicit Wait

1. **Implementation of Action, Select, Random, Alert Class**

**Practicals**: Build scripts to access web elements in a Context-Driven menu (Runtime generated menu)

**Practicals**: Build scripts drag-drop (Runtime generated list of items in drop-drop list)

1. **Cookie Management in Selenium**

**Practicals**: Build scripts to add, delete cookies

**Practicals**: Build scripts to get data of Cookie

### Day- **3**

1. **Capture Snapshot of Web site during script Execution**

**Practicals**: Build script using Java and Selenium to capture screen

1. **DataDriven Tests using Selenium, Java, Excel**

* Overview of variety of Test-Data sources
* Introduction to POM Framework
* Practicals: Reading data from Test-Data source
  + Build script to parse and extract data from Excel Test-Data source
  + Build script to parse and extract data from Text file Test-Data source
  + Build POM Framework

1. **Concept of Distributed Testing using Selenium Grid**

* How Selenium-Grid Works–With a Hub and Nodes
* Practicals: Configuring Selenium-Grid
  + Hub Configuration
  + Node Configuration
  + Common Errors
  + Troubleshooting

1. **Developing reusable script**

**Practicals: End-to-End Project using POM**

**12.0 Introduction to TestNG**

**13. 0 Handling of multi-browser automation**

* How to execute the test on different browser
* Automatically downloading driver executable from WebDriverManager
* Handling child browser and parent browser in selenium

**14.0 Working with popup**

* Handling alert
* Handling poupup from javascript
* Handling authentication popup
* Handling file download popup
* Handling file upload popup
* Overview of AutoIT, Robot class for non-web object handling

**15.0 Integrating Tests with Maven and Jenkins integration**

* Downloading and installing Maven
* Creating project in maven
* Executing test and validating maven reports
* Executing tests from command prompt
* Configuring Jenkins and Executing tests in Jenkins

**16.0 Appium introduction and overview**

* Introduction to Appium
* API walk through
* Advantages of Appium

**17.0 Open source platforms for Executing on cloud**

* overview- Browser stack and Sauce labs
* configuring Sauce lab to execute the tests
* selecting the required Browser with version and executing test

### Day-4

**18.0 Introduction to BDD**

* Overview of Behavior-Driven Development
* BDD and Test-Driven Development
* Cucumber Framework
* Course Pre-Requisites
* Preparing the Development Environment
* Preparing the Selenium and Cucumber Environment
* Step-by-Step Notes on the Environment
* Getting Started with a Project

19.0 Feature Files with Gherkin

* Overview of Feature Files
* Gherkin Syntax
* Writing a Feature
* Writing a Scenario
* Given-When-Then Structure
* Feature Files

20. Cucumber & Java Step Definitions

* Getting Started with Glue Code
* Writing a Step Definition
* Implementing Scenario Steps
* Running a Feature File
* Manage Execution with JUnit TestRunner
* Adding JUnit Assertions

21. Cucumber & Selenium WebDriver

* Getting Started with Selenium WebDriver
* Cucumber Options
* Integrating Selenium and Step Definitions
* JUnit Assertions to Selenium Test
* Cucumber Hooks

Day-5

22. Enhancing Cucumber Framework

* Defining Multiple Scenarios
* Additional Gherkin Keywords
* Cucumber Tags
* Background Keyword
* Creating a Hooks Class

23 Data Parameterization with Gherkin

* Gherkin Variable Placeholders
* Implementing Variable Placeholders with Java
* Gherkin Scenario Outline and Examples
* Java Implementation
* Gerhkin DataTables
* Implementing Cucumber DataTable with Java

24. [Code reusability with regular Expressions](javascript:void(0))

* importance of regular expressions in feature files
* How to reuse functions with different data
* [Cucumber Framework Reports and Test Runner configurations](javascript:void(0))
* Generating reports of cucumber scnearios
* Importance of Attributes in cucumber options
* How we have addressed common problems with cucumber features
* Project code download

25. Framework - TestNG and POM

Day-6

**[26. M](javascript:void(0))**[aven and J](javascript:void(0))**[enkins](javascript:void(0))**

* Importance of Maven in Framework development
* Installing and configuring Maven
* Understanding Terminologies of Maven
* Creating Maven Project and importing into eclipse
* Understanding POM.xml file and its dependencies
* Importance of surefirePlugin in executing Tests
* Importance of Jenkins in Test frameworks
* Install and configure Jenkins
* Configuring Jenkin Settings and Workspace

**27. Source Code Repository in GIT**

* GIT Fundamentals
* GIT Benching and Merging
* Lab: Working With GIT
* Lab: GIT Branching and Merging
* Lab: Code Commit Private GITHub

**28. Continuous Integration and Deployment**

* Continuous Integration Fundamentals
* Continuous Delivery Fundamentals
* CI/CD with Jenkins
* Lab: Install and Configure Jenkins
* Lab: Create a Simple pipeline in Jenkins
* Lab: Configuring Jenkins in CentOS server procured in Lab 1
* Lab: Integrating Jenkins with GIT
* Lab: Configuring Sample Maven Build in Jenkins
* Lab: Integrating Jenkins with Docker
* Lab: Integrating Jenkins with Ansible
* Lab: Configuring End to End Delivery Pipeline in Jenkins
* Lab: Running Continuous Deployment Using Jenkins