Selenium WebDriver

Agenda

- What is Selenium WebDriver
- WebDriver Features and Drawbacks
- Working of Selenium WebDriver
- Download WebDriver API
- Associate WebDriver jars to Java Project in Eclipse
- Create WebDriver test case
- Execute test case on multiple browsers

What is Selenium WebDriver

- WebDriver is a web automation framework that allows you to execute your tests against different browsers.
- WebDriver is an Java Interface.
- WebDriver is an API(Application Programming Interface)
- WebDriver is one of the component of Selenium suite.
- WebDriver supports multiple languages(Java, C#, Python, ruby etc..)
- WebDriver automates only web based applications(It needs browser)

WebDriver Advantages & Limitations

Advantages:

1) Its open source(free)

2) Supports multiple Operating systems(Windows, Mac, Linux, Unix etc..)

3) Supports multiple languages(Java, C#, Python,...)

4) Support multiple browsers(Chrome, Firefox, IE , etc...)

5) Selenium can integrate third party tools

Ex: AutoIT, Sikuli, Apache poi, TestNG, Extent reports etc...

Limitations:

1) Cannot automate desktop applications

2) Cannot automate images/graphs

Download WebDriver API

- http://www.seleniumhq.org/download/

Selenium Client & WebDriver Language Bindings

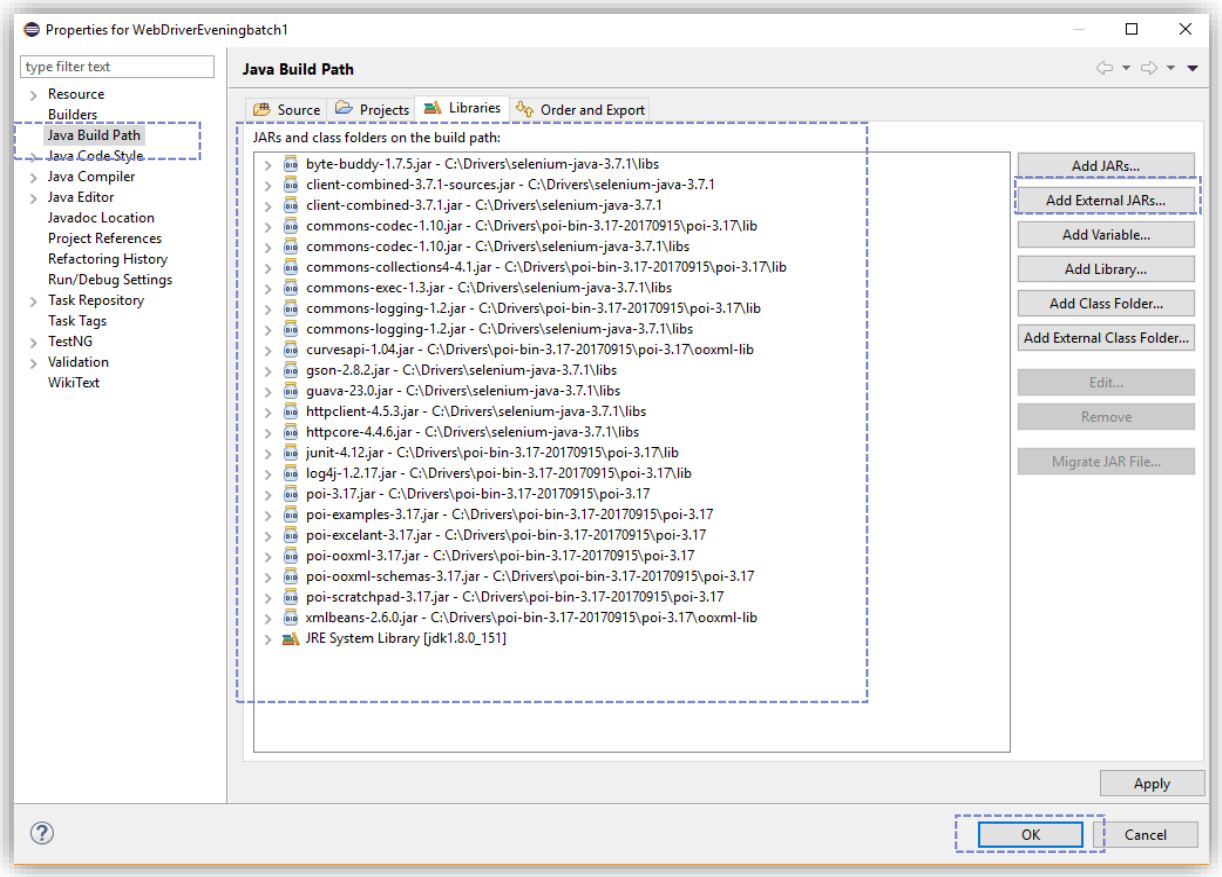
In order to create scripts that interact with the Selenium Server (Selenium RC, Selenium Remote WebDriver) or create local Selenium WebDriver scripts, you need to make use of language-specific client drivers. These languages include both 1.x and 2.x style clients.

While language bindings for other languages exist, these are the core ones that are supported by the main project hosted on GitHub.

Language	Client Version	Release Date			
Java	3.141.59	2018-11-14	Download	Change log	Javadoc
C#	3.14.0	2018-08-02	Download	Change log	API docs
Ruby	3.14.0	2018-08-03	Download	Change log	API docs
Python	3.14.0	2018-08-02	Download	Change log	API docs
Javascript (Node)	4.0.0-alpha.1	2018-01-13	Download	Change log	API docs

Associate WebDriver jars to Java Project in Eclipse

Right click on Project→ Properties→Java Build Path→Libraries→Add External Jars→Browse jar files→Open→OK



Browser Drivers

- **Firefox:** https://github.com/mozilla/geckodriver/releases/download/v0.19.1/geckodriver-v0.19.1-win64.zip
- **Chrome:** http://chromedriver.storage.googleapis.com/2.33/chromedriver_win32.zip
- **IE:** https://goo.gl/bn443g
- **Edge:** https://developer.microsoft.com/en-us/microsoft-edge/tools/webdriver/

System.setProperty()

- Chrome browser

```
System.setProperty("webdriver.chrome.driver", "C:/Drivers/chromedriver_win32/chromedriver.exe");
WebDriver driver=new ChromeDriver(); //ChromeDriver driver=new ChromeDriver();
```

- Firefox

```
System.setProperty("webdriver.gecko.driver", "C://Drivers/geckodriver-v0.23.0-win64/geckodriver.exe");
WebDriver driver=new FirefoxDriver(); //ChromeDriver driver=new ChromeDriver();
```

- IE Browser

```
System.setProperty("webdriver.ie.driver", "C://Drivers/IEDriverServer_Win32_3.14.0/IEDriverServer.exe");
WebDriver driver=new InternetExplorerDriver(); //ChromeDriver driver=new ChromeDriver();
```

- Microsoft Edge//OS Build no:17134.345 App: Microsoft Edge 42.17134.1.0

```
System.setProperty("webdriver.edge.driver", "C://Drivers/MicrosoftEdgeDriver/MicrosoftWebDriver.exe");
WebDriver driver = new EdgeDriver();
```

Demo

1. Open Web Browser.
2. Open URL http://opensource.demo.orangehrmlive.com
3. Enter username (Admin).
4. Enter password (admin).
5. Click on Login.
6. Capture title of the home page.
7. Verify title of the page: OrangeHRM

Challenges to run WebDriver Scripts with IE Browser

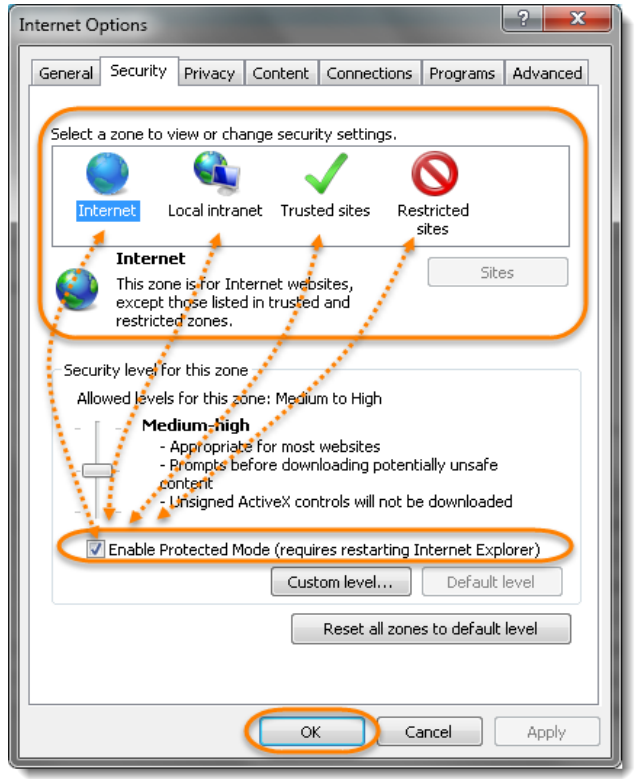
- The path to the driver executable must be set by the webdriver.ie.driver
- Solution: Need to add driver path to Environment variable.

- Protected Mode settings are not the same for all zones.
- Solution:
- Protected Mode settings in IE Browser

  1) Go to Tools > Internet Options and Under Internet Options click on the Security tab.
  2) Click on the Internet zone to Select a zone and to view its Protected Mode property.
  3) Now check the Enable Protected Mode check-box. Whichever one you choose, needs to be set for all the other zones. Means either it can be OFF for all the zones or ON for all the zones.
  4) Repeat this task for all the zones

  - Internet
  - Local Intranet
  - Trusted Sites
  - Restricted Sites

  5) Click *OK* and run the Selenium script again. It will work this time.



Hands On

- **Test Case-1**
  1. Open Browser
  2. Open URL http://demo.nopcommerce.com/
  3. Capture the title of the page
  4. Validate Title of the page.
  5. Expected Title: nopCommerce demo store

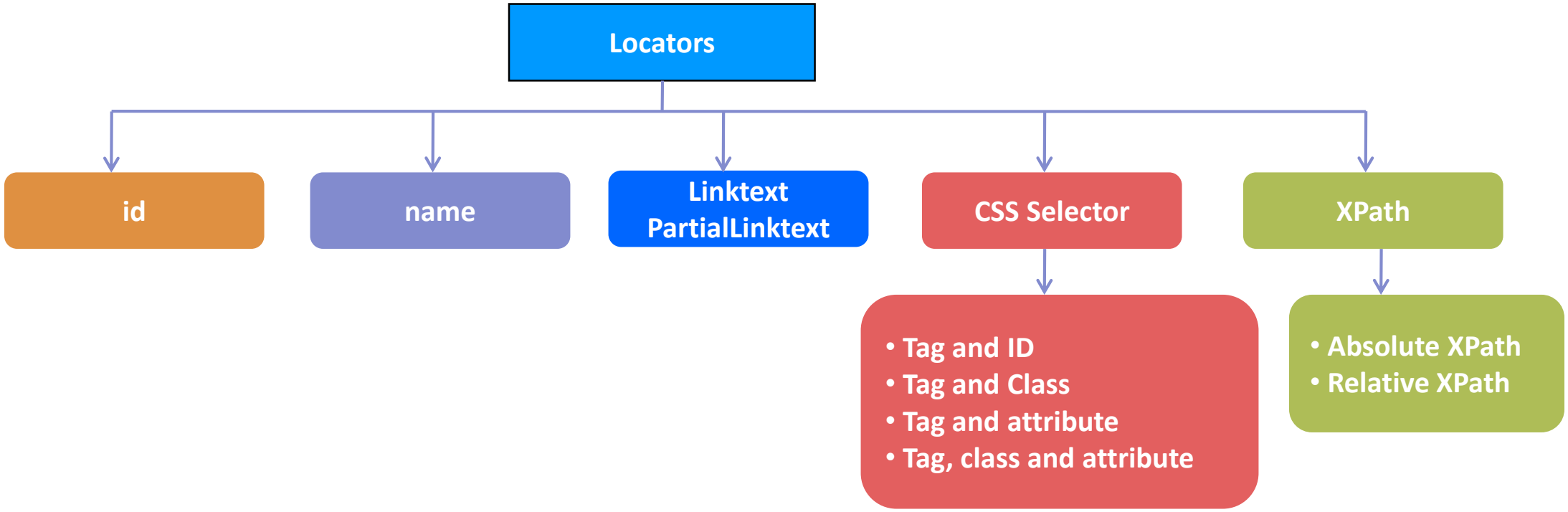
- **Test Case-2**
  1. Open Browser
  2. Open URL http://admin-demo.nopcommerce.com
  3. Enter username (admin@yourstore.com)
  4. Enter password (admin)
  5. Click on Log-In
  6. Capture the title of the page
  7. Validate Title of the page.
  8. Expected Title: Dashboard / nopCommerce administration

Agenda

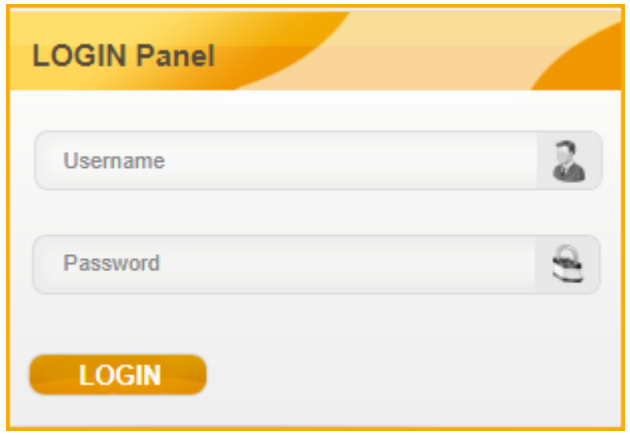
- Locators in Selenium

Locators

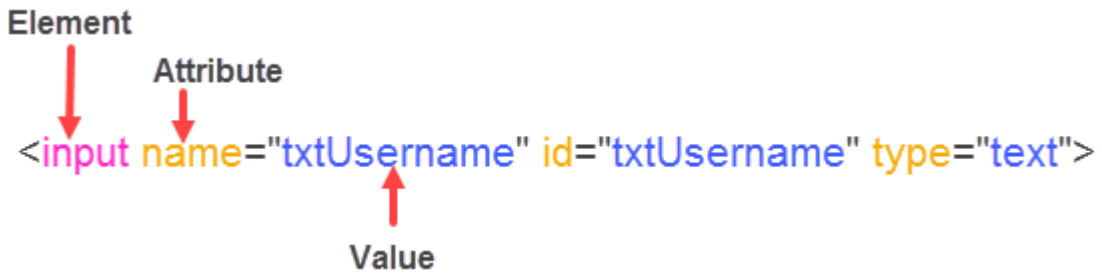
- We can identify various elements on the web using **Locators**.
- Locators are addresses that identify a web element uniquely within the page.



HTML Structure

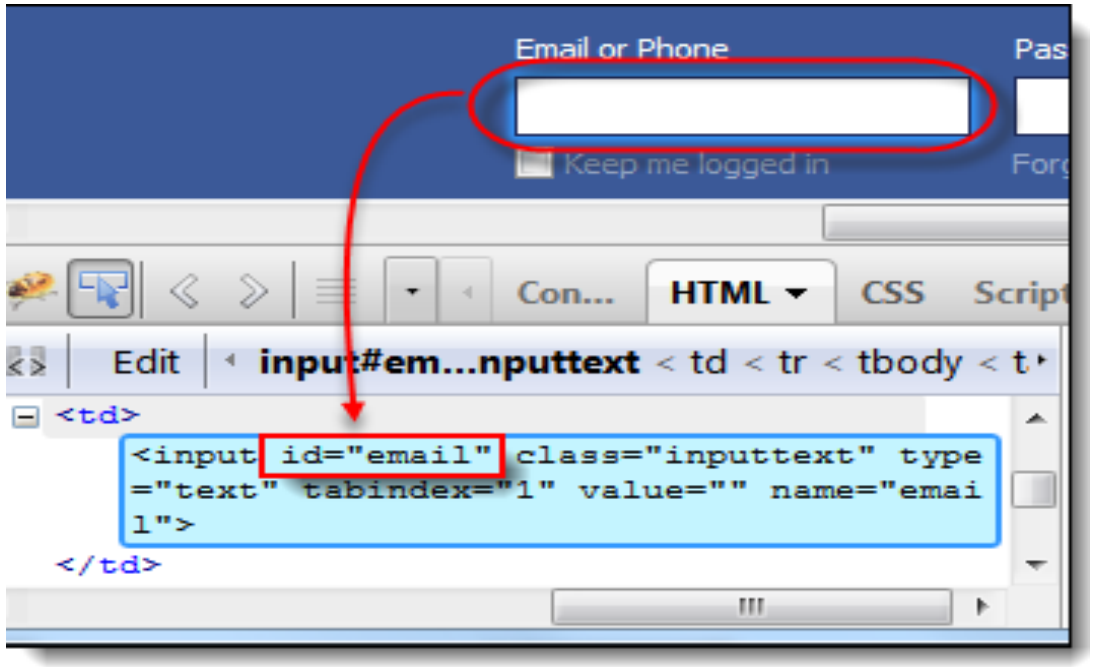



```
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="en" lang="en">
▶<head>…</head>
▼<body>
  ▼<div id="wrapper">
    ▼<div id="content">
      ▶<style type="text/css">…</style>
      ▼<div id="divLogin">
        ▶<div id="divLogo">…</div>
        ▼<form id="frmLogin" method="post" action="/index.php/auth/validateCredentials">
            <div id="logInPanelHeading">LOGIN Panel</div>
          ▼<div id="divUsername" class="textInputContainer">
              <input name="txtUsername" id="txtUsername" type="text">
              <span class="form-hint">Username</span>
            </div>
          ▼<div id="divPassword" class="textInputContainer">
              <input name="txtPassword" id="txtPassword" type="password">
              <span class="form-hint">Password</span>
            </div>
            <div id="divLoginHelpLink"></div>
          ▼<div id="divLoginButton">
              <input type="submit" name="Submit" class="button" id="btnLogin" value="LOGIN">
            </div>
          </form>
        </div>
      </div>
    </div>
  </body>
</html>
```



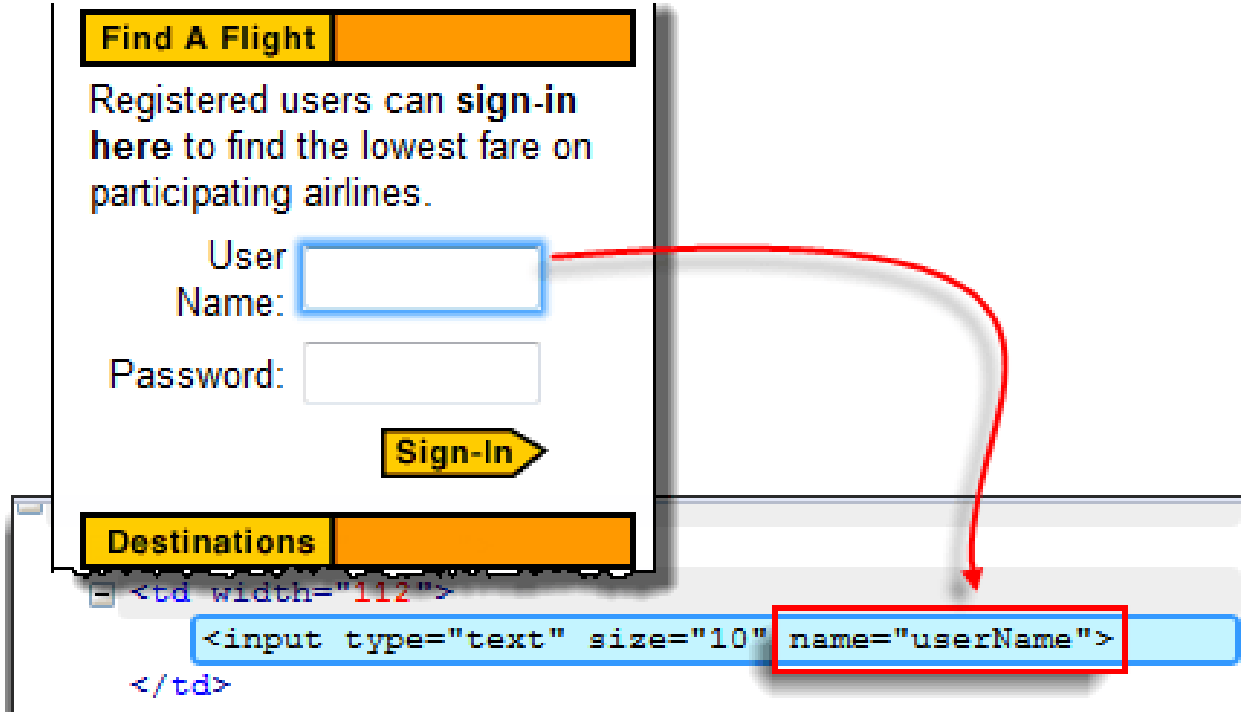
Locating by ID

- driver.findElement(By.id("email")).sendKeys(“xxxxx@gmail.com");



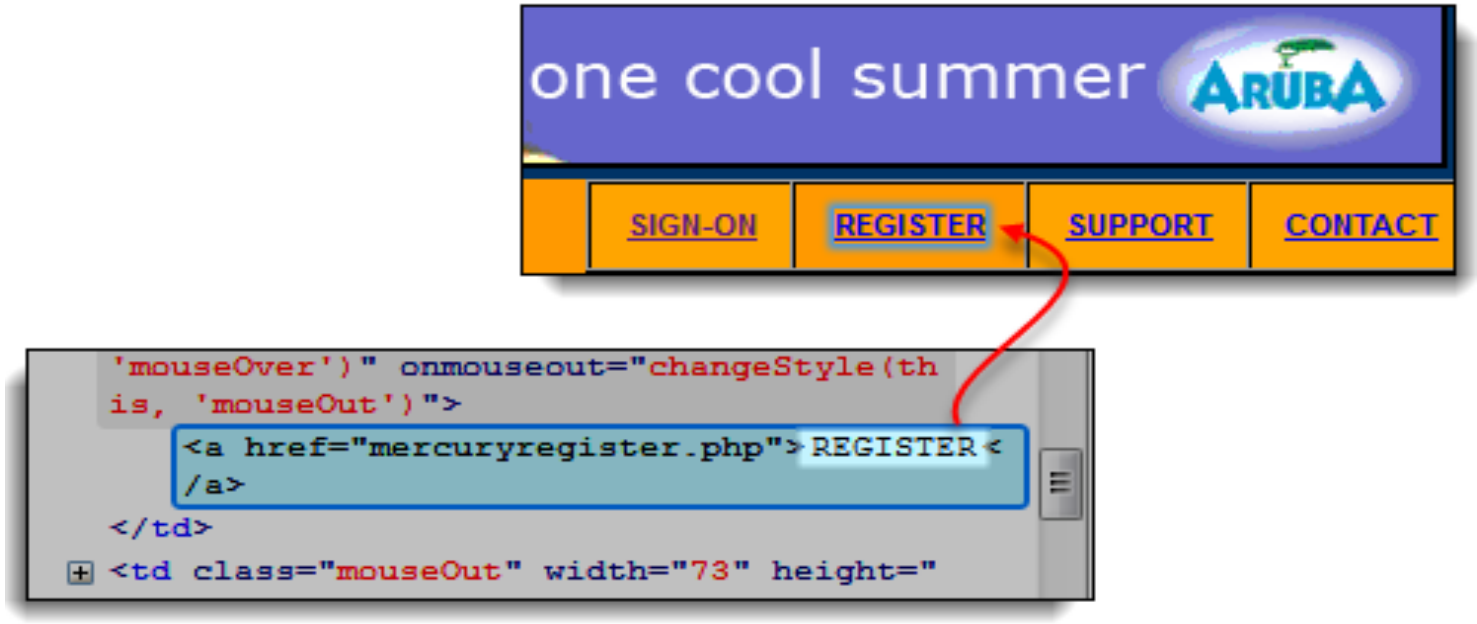
Locating by the name

- driver.findElement(By.name("userName")).sendKeys("mercury");



Locating by the Link Text

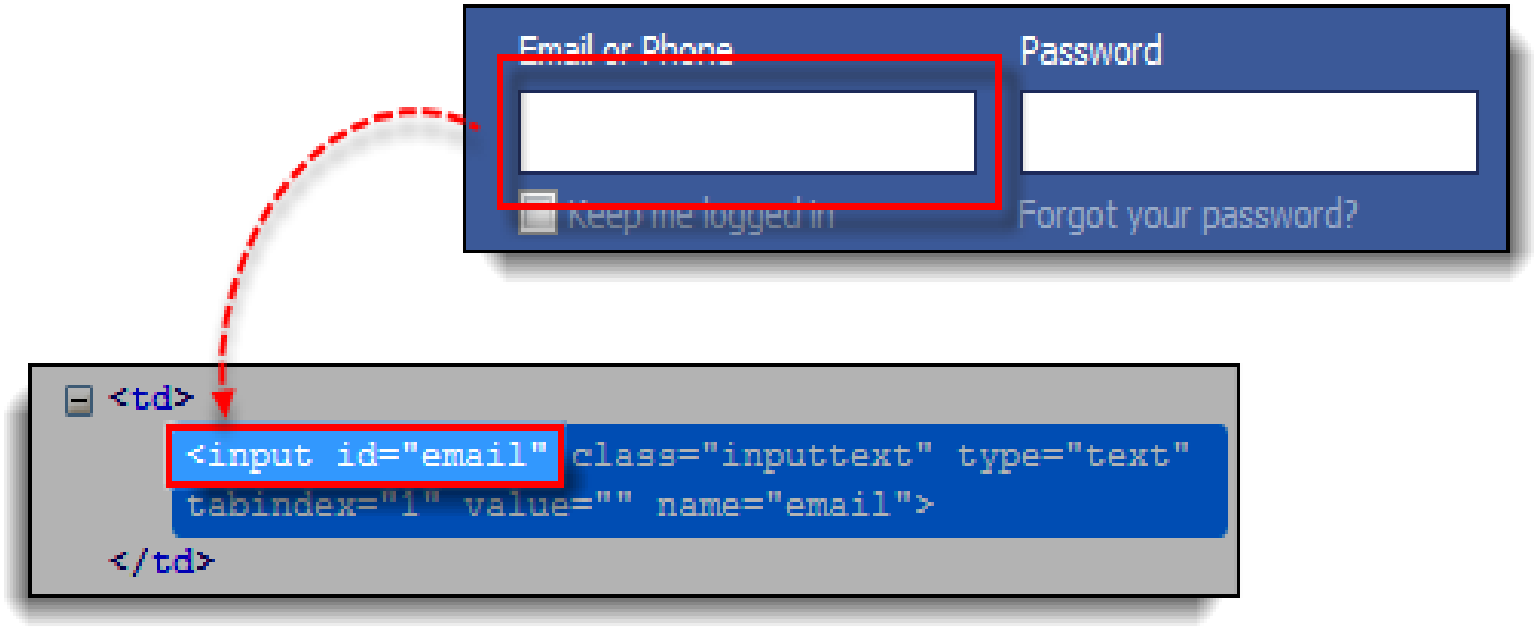
- driver.findElement(By.linkText("REGISTER")).click();
- driver.findElement(By.partialLinkText("REG")).click();



- Tag and ID
- Tag and Class
- Tag and attribute
- Tag, class and attribute

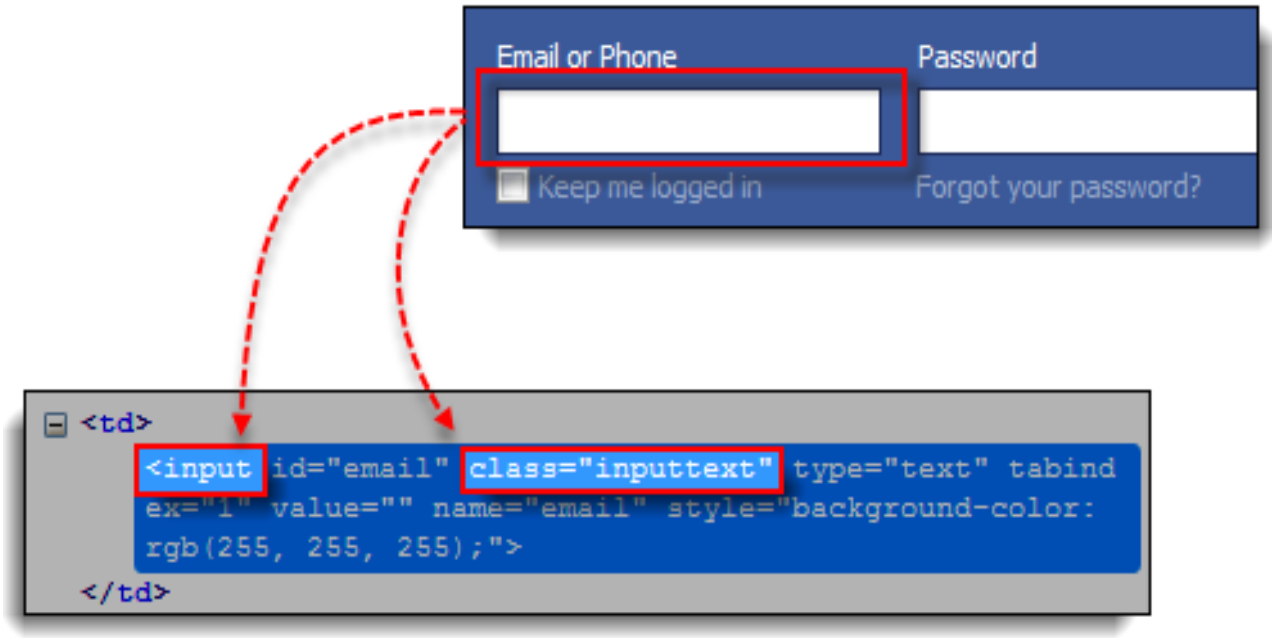
CSS Selector – *Tag* and *ID*

- driver.findElement(By.cssSelector("input#email")).sendkeys("xxxxxxx@gmail.com");



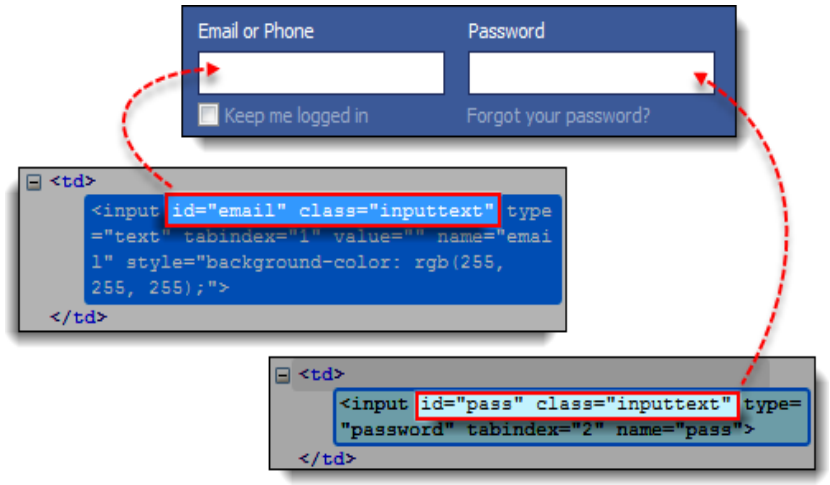
CSS Selector – *Tag* and *Class*

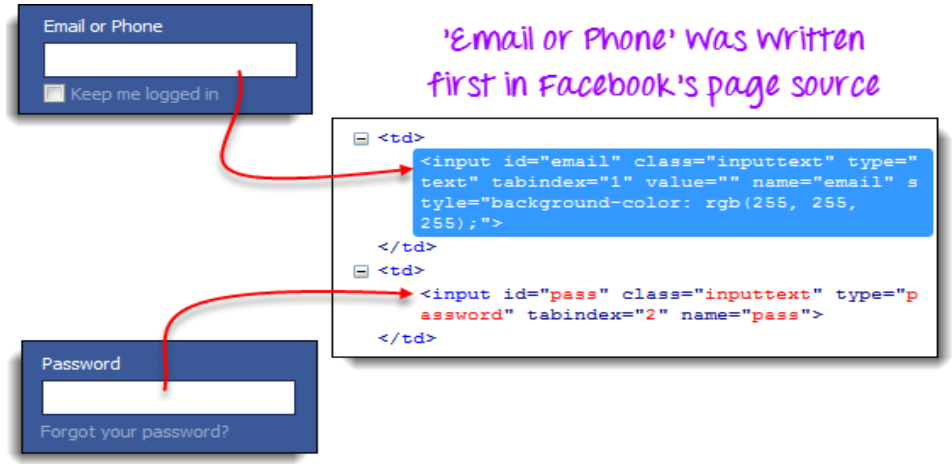
- driver.findElement(By.cssSelector("input.inputtext")).sendkeys("xxxxxx@gmail.com");



Multiple elements have the same tag and class name

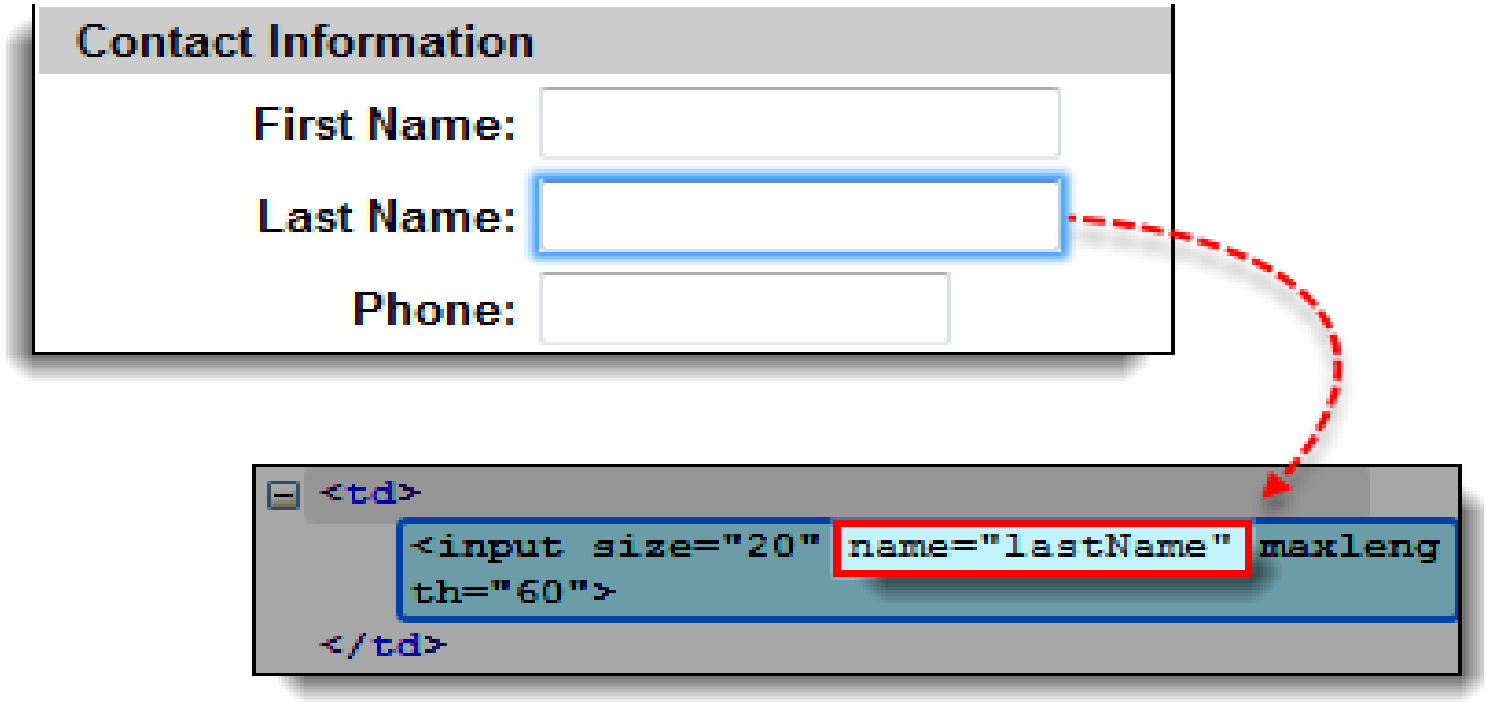
- When multiple elements have the same HTML tag and name, only the first element in source code will be recognized.
- driver.findElement(By.cssSelector("input.inputtext")).sendkeys("xxxxxxxx@gmail.com");





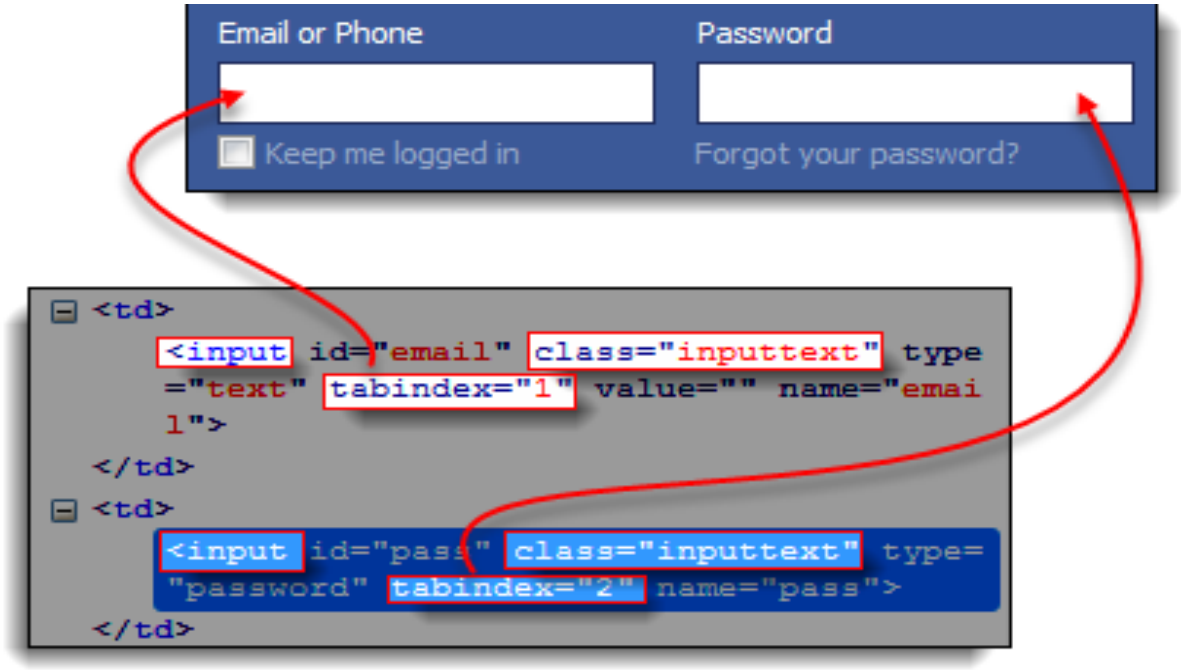
CSS Selector – *Tag* and *Attribute*

- driver.findElement(By.cssSelector("input[name=lastName]")).sendkeys("xxxxx");



CSS Selector - *Tag, class* and *attribute*

- driver.findElement (By.cssSelector("input.inputtext[tabindex=2]").sendkeys("xxxx");



XPath

What is XPath?

- XPath is defined as **XML path**.
- **It is a syntax or language for finding any element on the web page using XML path expression**.
- XPath is used to find the location of any element on a webpage using HTML DOM structure.

Types of X-path

- There are two types of XPath:

1) Absolute XPath

2) Relative XPath

Absolute XPath

- It is the direct way to find the element, but the disadvantage of the absolute XPath is that if there are any changes made in the path of the element then that XPath gets failed.
- It begins with the single forward slash(/) ,which means you can select the element from the root node.
- Below is the example of an absolute xpath expression of the element shown in the below screen.
- Ex:
- /html[1]/body[1]/div[1]/table[1]/tbody[1]/tr[1]/td[2]/table[1]/tbody[1]/tr[4]/td[1]/table[1]/tbody[1]/tr[1]/td[2]/table[1]/tbody[1]/tr[2]/td[3]/form[1]/table[1]/tbody[1]/tr[4]/td[1]/table[1]/tbody[1]/tr[2]/td[2]/input[1]

Relative xpath

- For Relative Xpath the path starts from the middle of the HTML DOM structure.
- It starts with the double forward slash (//), which means it can search the element anywhere at the webpage.
- You can start from the middle of the HTML DOM structure and no need to write long xpath.

Ex:

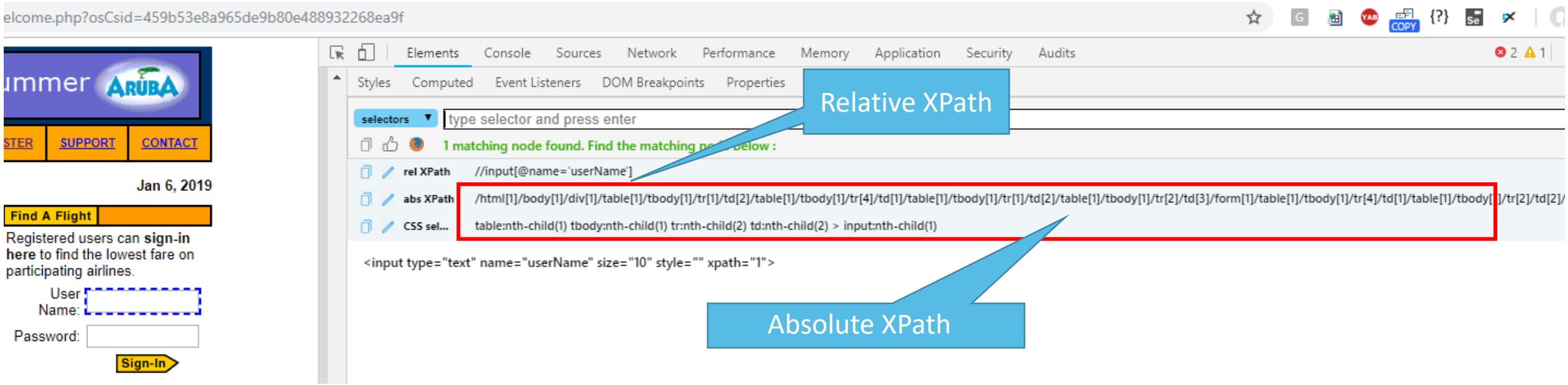
```
//input[@name='userName']
```

Syntax for Relative XPath

- XPath contains the path of the element situated at the web page. Standard syntax for creating XPath is.
- **// :** Select current node.
- **Tagname:** Tagname of the particular node.
- **@:** Select attribute.
- **Attribute:** Attribute name of the node.
- **Value:** Value of the attribute.

Xpath=//tagname[@attribute='value']

Absolute & Relative XPath

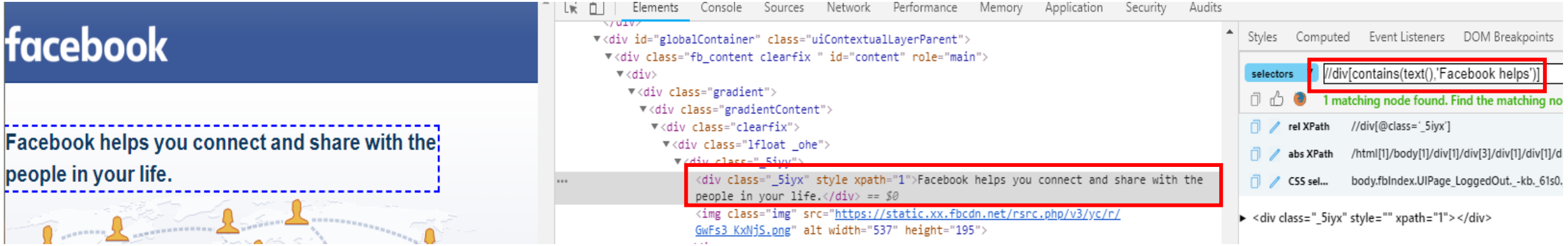


Contains()

- Contains() is a method used in XPath expression. It is used when the value of any attribute changes dynamically, for example, login information.
- The contain feature has an ability to find the element with partial text
- //input[contains(@name,'first')]

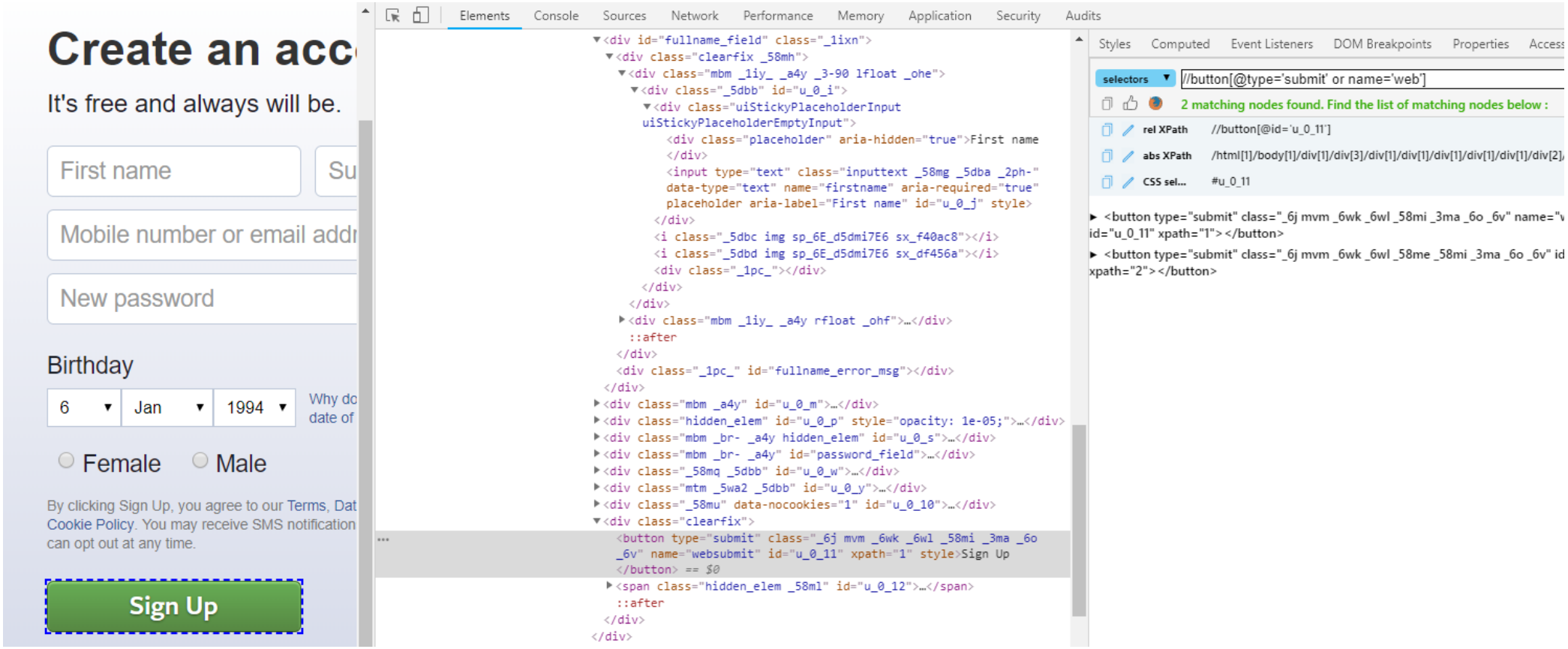
Contains() with text()

- //div[contains(text(),'Facebook helps')]



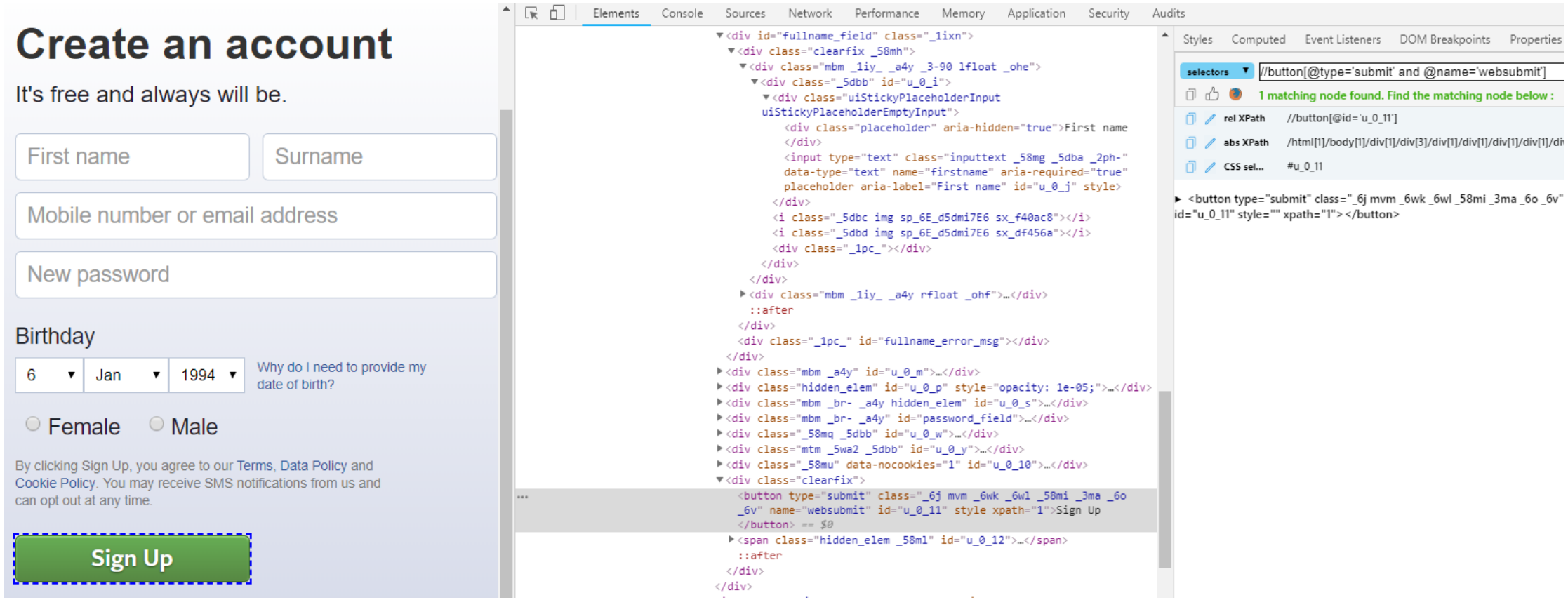
Using OR

- //button[@type='submit' or name='web']



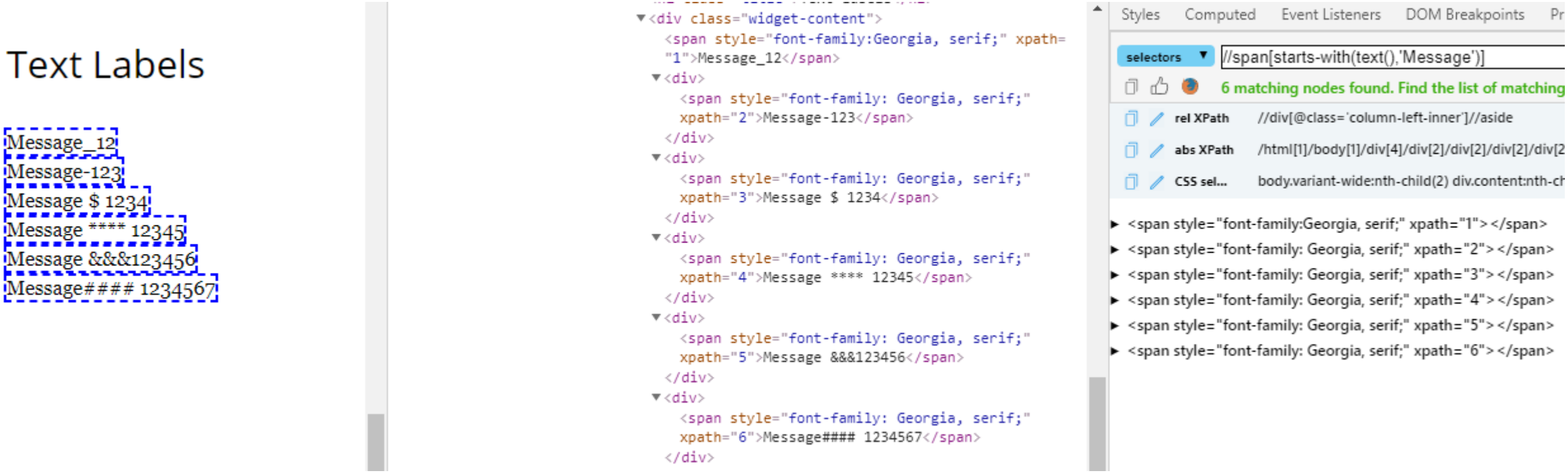
Using AND

- //button[@type='submit' and @name='websubmit']



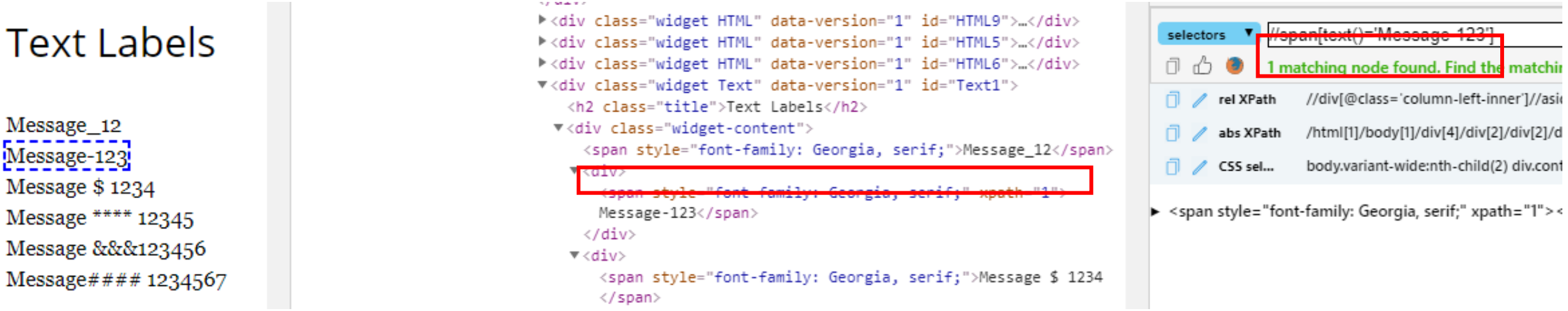
Start-with()

- //span[**starts-with**(text(),'Message')]



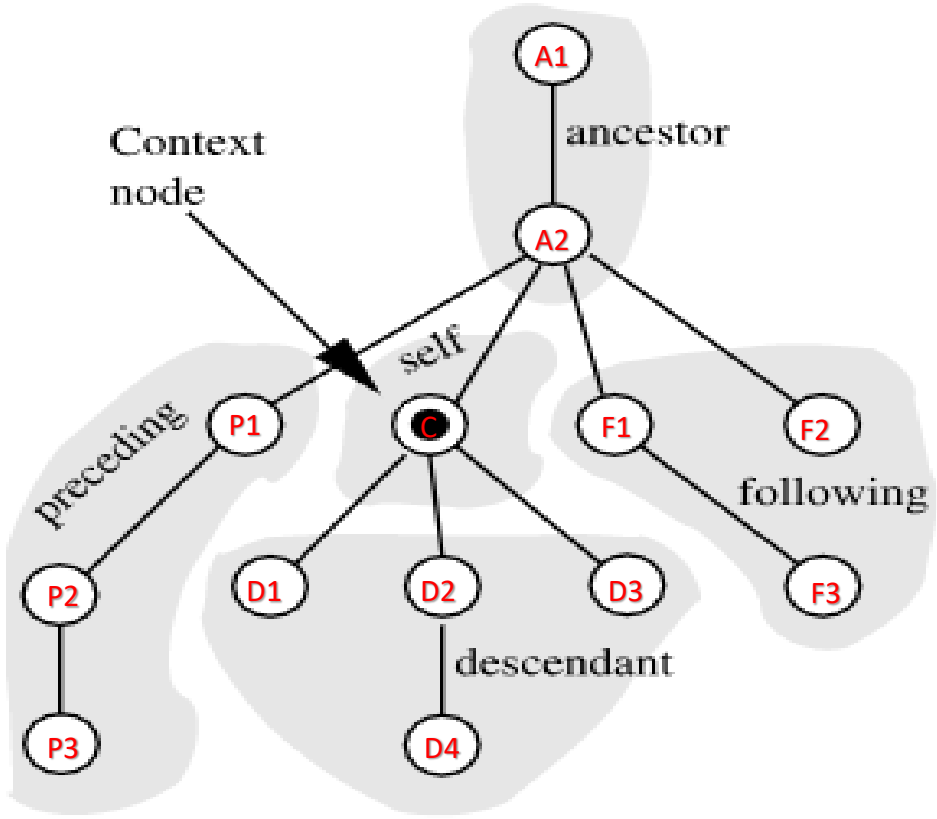
text()

- //span[text()='Message-123']



XPath axes methods

- XPath methods are used to find the complex or dynamic elements.
  - Ancestor
  - Child
  - Parent
  - Preceding
  - Following
  - Self
  - Descendant



Ancestor

- selects all ancestors element (grandparent, parent, etc.) of the current node

//employee/ancestor::*	Select all ancestor node of the employee node.
//ancestor::name	Select all ancestor of the name node in context node.

Child

- select child of the context node.

child::*	Select All child nodes of the context node.
child::employee	Select all child elements of employee node.

Parent

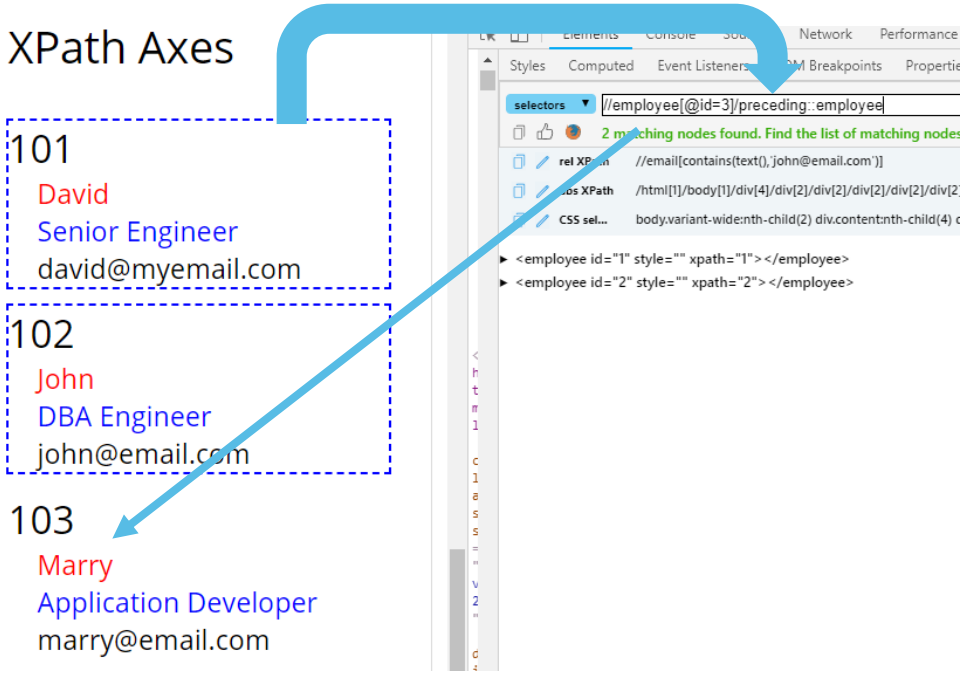
- select the parent node of the context node.

//name/parent::*	Select parent node of the ’name’ context node.
//email/parent::employee	Return result node if employee node is parent node of the context node, otherwise no node found.

Preceding

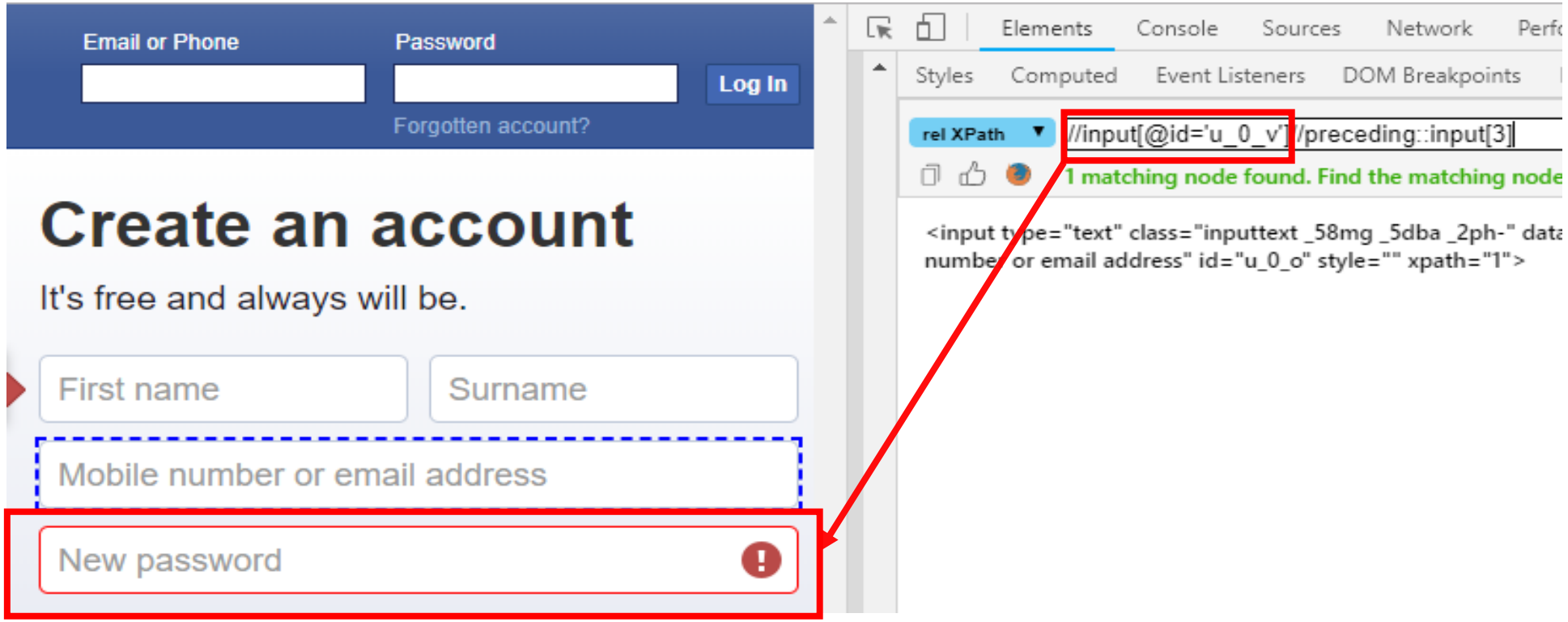
- select all nodes before the context node, excluding attributes node or namespace node.

//employee[@id=3]/preceding::employee	Select all nodes (with child nodes) before the context node.



Preceding...

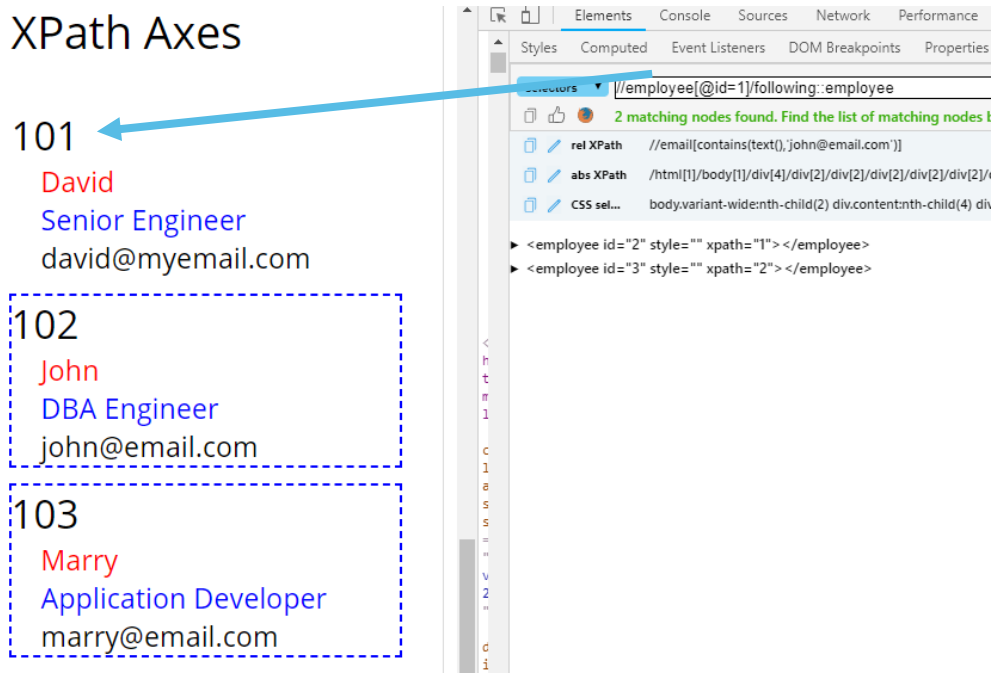
- //input[@id='u_0_v']//preceding::input[3]



Following

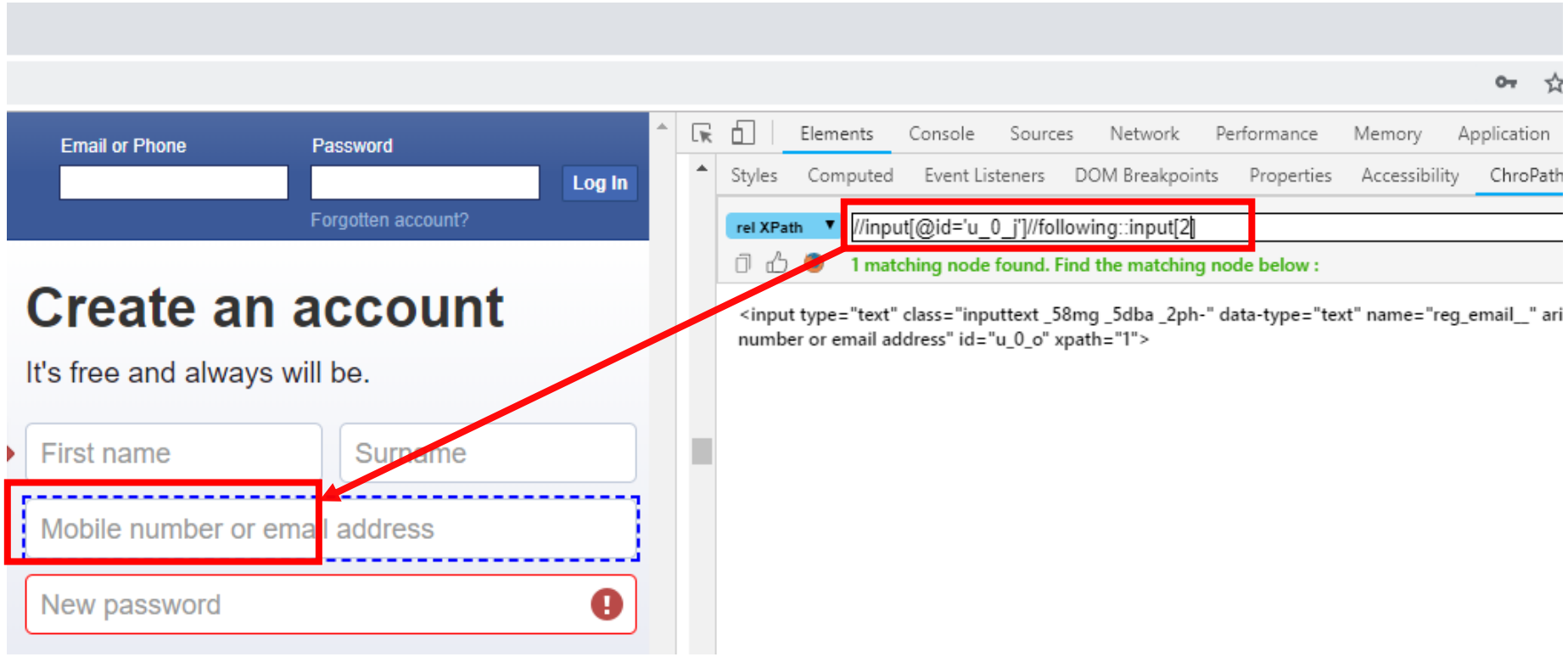
- select all nodes after the context node, excluding attributes node or namespaces node.

//employee[@id=1]/following::employee	Select all nodes (with child nodes) after the context node.



Following..

- //input[@id='u_0_j']//following::input[2]



Self

- Selects the current node ‘name’

Descendant

- Selects all descendants (children, grandchildren, etc.) of the current node

//descendant::employee	Select all descendant of the employee node in context node.

Summary of Locators

Locators	Description	Example
By.className	finds elements based on the value of the "class" attribute	findElement(By.className("someClassName"))
By.cssSelector	finds elements based on the driver's underlying CSS Selector engine	findElement(By.cssSelector("input#email"))
By.id	locates elements by the value of their "id" attribute	findElement(By.id("someId"))
By.linkText	finds a link element by the exact text it displays	findElement(By.linkText("REGISTRATION"))
By.name	locates elements by the value of the "name" attribute	findElement(By.name("someName"))
By.partialLinkText	locates elements that contain the given link text	findElement(By.partialLinkText("REG"))
By.tagName	locates elements by their tag name	findElement(By.tagName("div"))
By.xpath	locates elements via XPath	findElement(By.xpath("//html/body/div/table/tbody/tr/td[2]/table/tbody/tr[4]/td/table/tbody/tr/td[2]/table/tbody/tr[2]/td[3]/form/table/tbody/tr[5]"))

Agenda

- WebDriver commands

Get commands

- get()
- The command launches a new browser and opens the specified URL in the browser instance.
- The command takes a single string type parameter that is usually a URL of application under test

Ex: *driver.get("https://google.com");*

- getTitle()
- The command is used to retrieve the title of the webpage the user is currently working on. A null string is returned if the webpage has no title
- The command doesn't require any parameter and returns a trimmed string value

Ex: *String title = driver.getTitle();*

Get commands

getCurrentUrl()

- The command is used to retrieve the URL of the webpage the user is currently accessing .
- The command doesn’t require any parameter and returns a string value.

Ex: driver.getCurrentUrl();

- getPageSource()
- The command is used to retrieve the page source of the webpage the user is currently accessing
- The command doesn’t require any parameter and returns a string value
- The command can be used with various string operations like contains() to ascertain the presence of the specified string value

  Ex: *boolean result = driver.getPageSource().contains("String to find");*

Browser commands

close(): close() method closes the web browser window that the user is currently working on or we can also say the window that is being currently accessed by the WebDriver. The command neither requires any parameter nor does it return any value.

quit(): quit() method closes down all the windows that the program has opened. Same as close() method, the command neither requires any parameter nor does it return any value.

```
driver.close(); // closes only a single window that is being accessed by the WebDriver instance currently

driver.quit(); // closes all the windows that were opened by the WebDriver instance
```

Conditional commands

- isDispalyed()
- isDisplayed() is the method used to verify presence of a web element within the webpage.
- *Ex: boolean searchIconDisplayed = driver.findElement(By.id("gbqfb")).isDisplayed();*
- isEnabled()
- isEnabled() is the method used to verify if the web element is enabled or disabled within the webpage.
- *Ex: boolean searchIconEnabled = driver.findElement(By.id("gbqfb")).isEnabled();*
- isSelected()
- isSelected() is the method used to verify if the web element is selected or not. isSelected() method is pre-dominantly used with radio buttons, dropdowns and checkboxes.
- *Ex: boolean searchIconSelected = driver.findElement(By.id("male")).isSelected();*

Navigational Commands

- navigate().to() : This command is like the get() command.
  - It opens a new browser window and it will receive / find the page you provided.
  - Able to redirect from the current web page to the expected web page.
- navigate().forward() : A page leads you to the history of the browser.
- navigate().back() : This command is used to return to a page in the browser's history.
- navigate().refresh() : This command is used to refresh the current page.

Hand’s on

- https://testautomationpractice.blogspot.com/

Wait Commands

- Thread.sleep Command
- The sleep code always has to wait for the seconds mentioned within the bracket, even if the work page is ready after 5 seconds. So this test can slow down.
- Ex: *Thread.sleep(5000); // Wait 5 Seconds*
- Implicit Wait:
- Implicit Wait means informing selenium web driver to wait for specific amount of time.
- If the web element is not visible after waiting for that specific point then throw “NoSuchElementException”.
- Ex: *driver.manage().timeouts().implicitlyWait(30,TimeUnit.SECONDS);*

Wait Commands(Con..)

- Explicit Wait:
- In Explicit wait along with wait time we also provide the wait condition for particular WebElement.
- It will wait till the condition or the maximum wait time provided before throwing the Exception *“ElementNotVisibleException”.*
- Wait for the WebDriver to check whether the element exists or to operate on it, visible or enabled or disabled or clickable.
- Ex:

```
// Create a Object wait of WebdriverWait class
WebDriverWait wait = new WebDriverWait(driver,30);
 // Using ExpectedConditions wait until element visibility
wait.until(ExpectedConditions.visibilityOfElementLocated(By.tagName("input")));
```

- **ExpectedConditions** class can be useful in a lot of cases and provides some set of predefined condition to wait for the element. Here are some of them below:

Wait Commands(Con.)

- findElement using locator and store into WebElement element to use in ExpectedConditions class:
  - WebElement element = driver.findElement(By.id("id"));
- **alertIsPresent :**
  - wait.until(ExpectedConditions.alertIsPresent()); // Wait until alert present on page
- **elementToBeClickable:**
  - wait.until(ExpectedConditions.elementToBeClickable(element)); // Wait until element to be clickable on page
- **elementToBeSelected:**
  - wait.until(ExpectedConditions.elementToBeSelected(element)); // Wait until element to be selectable on page
- **frameToBeAvailableAndSwitchToIt:**
  - wait.until(ExpectedConditions.frameToBeAvailableAndSwitchToIt(element)); // Wait until frame is available and frame selected.

Wait Commands(Con.)

- **invisibilityOf:**
  - wait.until(ExpectedConditions.invisibilityOf(element)); // Wait and check element is invisibility
- **presenceOfAllElementsLocatedBy:**
  - wait.until(ExpectedConditions.presenceOfAllElementsLocatedBy((By) element)); // Wait until present element located by.
- **textToBePresentInElement:**
  - wait.until(ExpectedConditions.textToBePresentInElement(element, "Text")); //// Wait until text present on particular an element
- **textToBePresentInElementValue:**
  - // Wait until element value present for a particular element.
  - wait.until(ExpectedConditions.textToBePresentInElementValue(element, "Value"));
- **visibilityOf:**
  - // check element visibility
  - wait.until(ExpectedConditions.visibilityOf(element));
- **titleContains:**
  - // Wait and check title contains or not.
  - wait.until(ExpectedConditions.titleContains("Tilte"));

PageLoadTimeout Command

- Determines the amount of time to wait for the page load to complete before the error is thrown at the end of time.
- If the timeout is negative, then the page load may be indefinite until the pageload.
- Ex:
  - driver.manage().timeouts().pageLoadTimeout(50, TimeUnit.SECONDS);

FluentWait Command

- Fluent Wait uses two parameters to handle wait– **timeout value** and **polling frequency**.
- Fluent wait is another type of Explicit wait and you can define polling and ignore the exception to continue with script execution in case element is not found in webpage.
- First of all, it sets the following values.
  1. Maximum time to wait for any condition.
  2. Frequency to check the success or failure of a specified position.

```
FluentWait<WebDriver> wait = new FluentWait<WebDriver>(driver)
 .withTimeout(timeoutSeconds, TimeUnit.SECONDS)
 .pollingEvery(500, TimeUnit.MILLISECONDS)
 .ignoring(NoSuchElementException.class);
```

Hand's on

- http://newtours.demoaut.com/
- http://demo.nopcommerce.com/

- Assignment-1
- URL: https://goo.gl/RVdKM9
- 1) Capture the title of the page.
- 2) Capture the current URL.
- 3) Check First Name & Last Name text boxes presence and enabled or not.
- 4) Select gender male/female then check the status selected or not.
- 5) Close browser.

- Assignment-2
- URL: http://automationpractice.com
- 1) Capture the title of the page.
- 2) Click on Sign-in Link
- 3) Provide Email and password click on Sign-in button.
- 4) Check the title of the Home page after login
- 5) Quit browser.

- ***Note:*** *Use wait commands wherever it is required.*

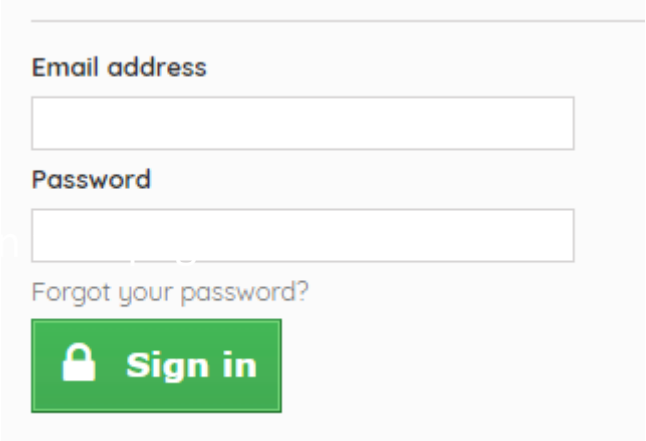
Working with Web Elements

- Text Box/Input Box
- Dropdown/Combo/List box
- Searchable Drop-Down
- Radio Button/Check Box
- Links
- Switch to Alerts
- Switch to Frames/Iframes
- Switch to Windows
- Web Tables
- Date Picker

- Mouse Actions
  - Mouse Hover
  - Right Click
  - Double Click
  - Drag and Drop
  - Slider
  - Resizing
- Tooltips
- Scrolling
- File Download
- File Upload
- Bar Code
- QR Code

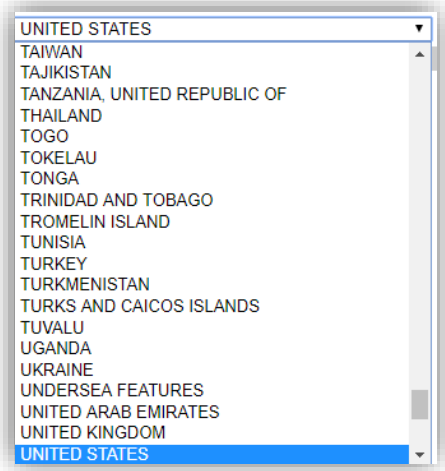
Text Box/Input Box

- sendKeys() :
- clear() :
- isDisplayed():
- isEnabled():

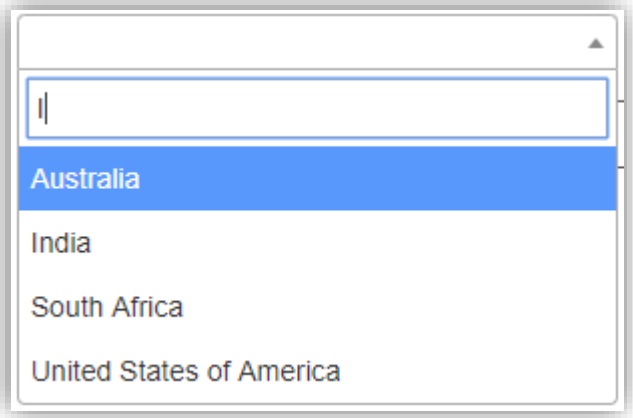


Dropdown/Combo/List box

- getOptions() : Capture options from dropdown
- getOptions().size() : Count options in dropdown
- selectByIndex(): select option using index value
- selectByValue: select option using value
- selectByVisibleText(): select option using visible text
- deselectByIndex(): deselect option using index value
- deselectByValue: deselect option using value
- deselectByVisibleText(): deselect option using visible text

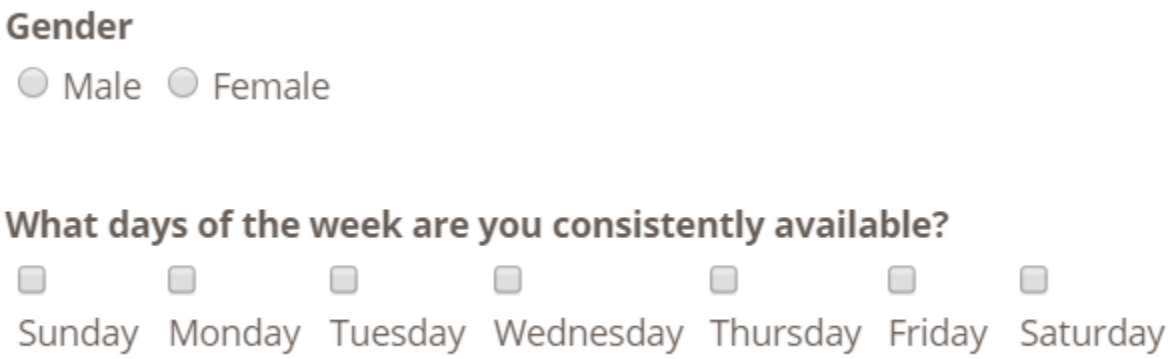


Searchable Drop-Down



Radio Button/Check Box

- click() : select radio button/checkbox
- isDisplayed(): checks radio button/checkbox presence on web page.
- isEnabled(): checks radio button/checkbox is enabled or not



Links

- Operations on Links
  - Count how many links present on a web page
  - Capture all the link on a page
  - Check links are working or not
  - Find Broken Links on a page
- click() : click on link
- isDisplayed(): checks link presence on web page.

Software Testing Tutorials

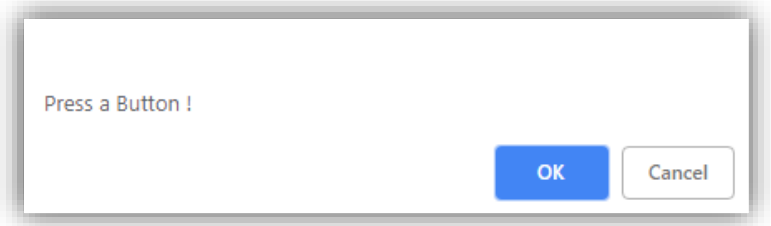
Software Testing Tools Training

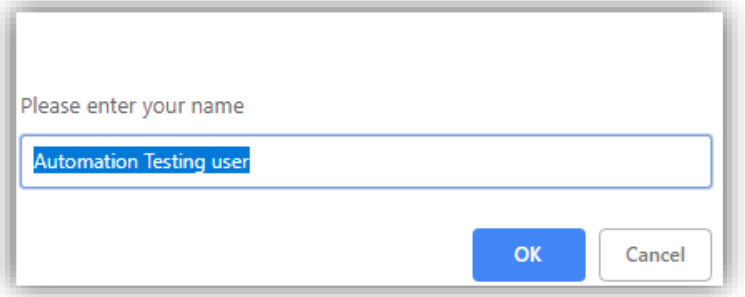
Hand's on

- https://testautomationpractice.blogspot.com/
- http://demo.automationtesting.in/Register.html
- http://demo.nopcommerce.com/register
- http://newtours.demoaut.com/mercuryregister.php
- http://my.monsterindia.com/create_account.html?
- https://money.rediff.com/gainers/bsc/daily/groupa

Alerts

- driver.switchTo().alert() : switch to alert
- driver.switchTo().alert().accept() : switch to alert and close using OK button
- driver.switchTo().alert().dismiss() : switch to alert and close using Cancel button
- driver.switchTo().alert().sendKeys(text) : switch to alert and provide input to text box
- driver.switchTo().alert().getText() : switch to alert and capture the text present on alert box



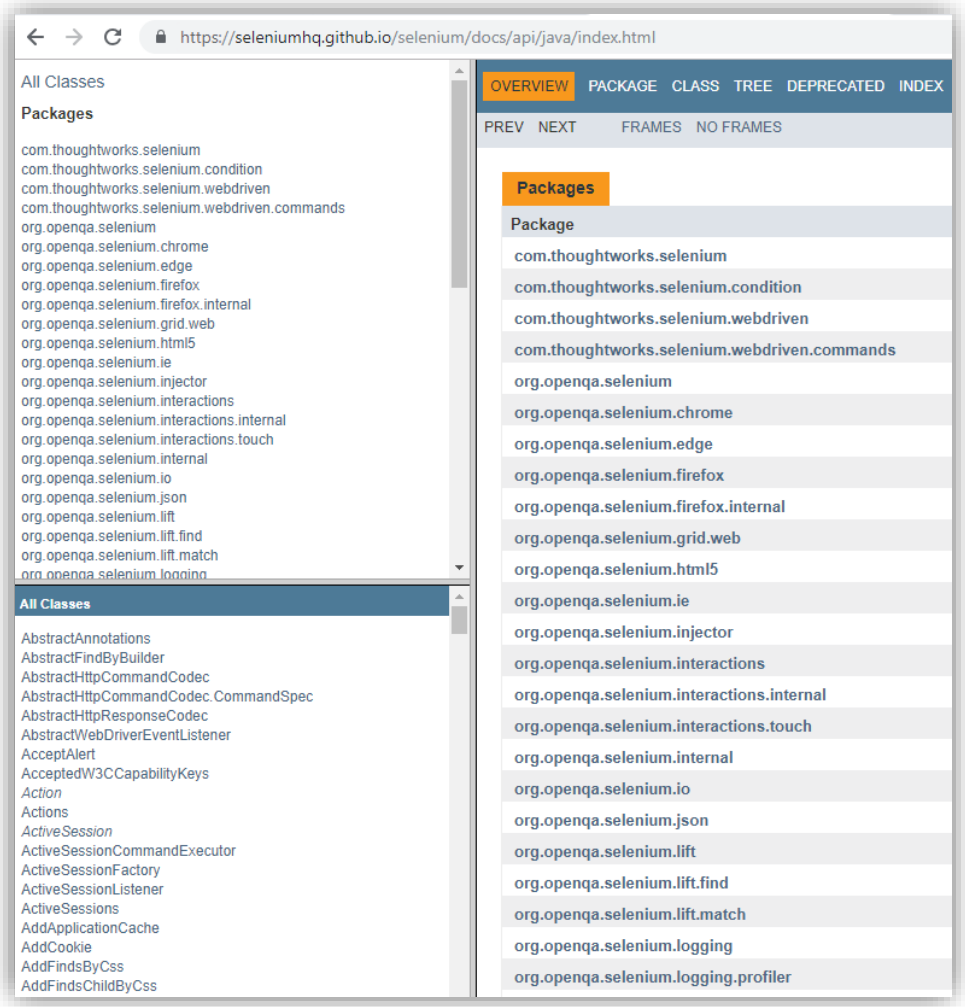


Hand’s on

- https://testautomationpractice.blogspot.com/
- http://demo.automationtesting.in/Alerts.html
- https://mail.rediff.com/cgi-bin/login.cgi

iFrames

- driver.switchTo().frame(name of the frame) :
  - switch to frame using name
- driver.switchTo().frame(WebElement) :
  - switch to frame using WebElement
- driver.switchTo().frame(index) :
  - switch to frame using index
- driver.switchTo().defaultContent():
  - switch to main Page



Hand's on

- http://seleniumhq.github.io/selenium/docs/api/java/index.html
- http://demo.automationtesting.in/Frames.html

Browser Windows

- getWindowHandle()
- The command is used to tackle with the situation when we have more than one window to deal with.
- The command helps us switch to the newly opened window and performs actions on the new window.
- The user can also switch back to the previous window if he/she desires.
- String handle = driver.getWindowHandle();
- driver.switchTo().window(handle);
- getWindowHandles()
- The command is similar to that of “getWindowHandle()” with the subtle difference that it helps to deal with multiple windows i.e. when we have to deal with more than 2 windows.
- Set <String> handles = driver.getWindowHandles();

Hand's on

- http://demo.automationtesting.in/Windows.html

Web Table

- **Web Table Operations:**
  - Counting rows in a table
  - Counting columns in each row
  - Fetch the data from a table
  - Validate data in the table

BookName	Author	Subject	Price
Learn Selenium	Amit	Selenium	300
Learn Java	Mukesh	Java	500
Learn JS	Animesh	Javascript	300
Master In Selenium	Mukesh	Selenium	3000
Master In Java	Amod	JAVA	2000
Master In JS	Amit	Javascript	1000

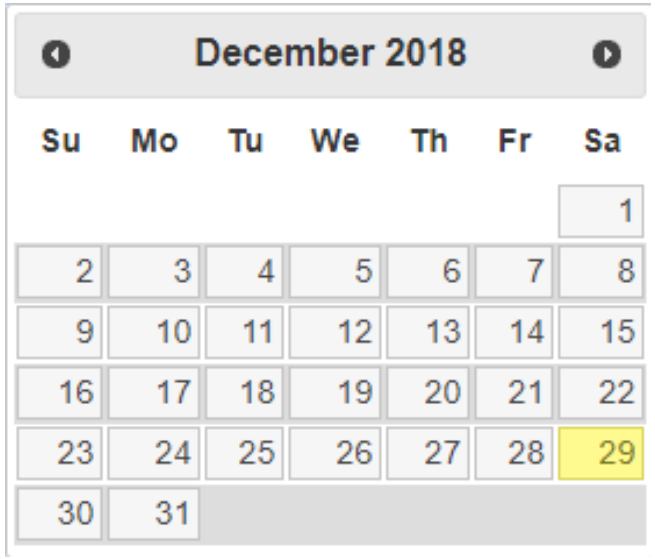
#	Table heading 1	Table heading 2	Table heading 3	Table heading 4	Table heading 5	Table heading 6
1	Table cell	Table cell	Table cell	Table cell	Table cell	Table cell
2	Table cell	Table cell	Table cell	Table cell	Table cell	Table cell
3	Table cell	Table cell	Table cell	Table cell	Table cell	Table cell
4	Table cell	Table cell	Table cell	Table cell	Table cell	Table cell
5	Table cell	Table cell	Table cell	Table cell	Table cell	Table cell

Username	User Role	Employee Name	Status
Admin	Admin		Enabled
fiona.grace	ESS	Fiona Grace	Enabled
hannah.flores	ESS	Hannah Flores	Disabled
jasmine.morgan	ESS	Jasmine Morgan	Enabled
john.smith	ESS	John Smith	Disabled
linda.anderson	ESS	Linda Anderson	Enabled
robert.craig	ESS	Robert Craig	Enabled
russel.hamilton	ESS	Russel Hamilton	Enabled
steven.edwards	ESS	Steven Edwards	Enabled
thomas.fleming	ESS	Thomas Fleming	Enabled

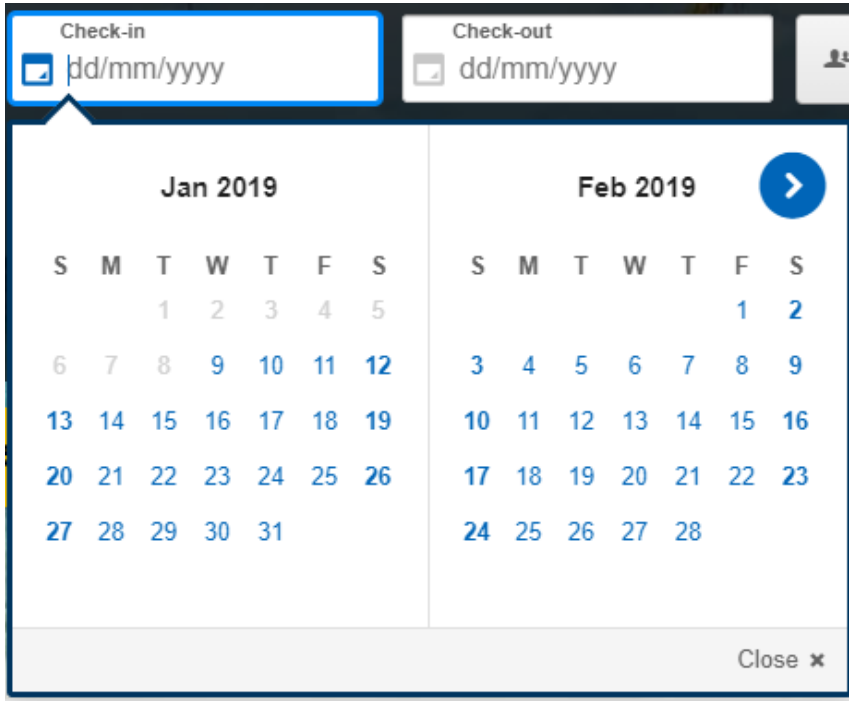
Hand's on

- https://opensource-demo.orangehrmlive.com/
- https://money.rediff.com/gainers/bse/daily/
- https://testautomationpractice.blogspot.com/
- http://admin-demo.nopcommerce.com/Admin/Customer/List
- http://demo.automationtesting.in/WebTable.html
- https://money.rediff.com/gainers/bsc/daily/groupa

Date Picker


December 2018
Su Mo Tu We Th Fr Sa
1
2 3 4 5 6 7 8
9 10 11 12 13 14 15
16 17 18 19 20 21 22
23 24 25 26 27 28 29
30 31


January 2019
Su Mo Tu We Th Fr Sa
30 31 1 2 3 4 5
6 7 8 9 10 11 12
13 14 15 16 17 18 19
20 21 22 23 24 25 26
27 28 29 30 31 1 2
3 4 5 6 7 8 9


Check-in
dd/mm/yyyy
Check-out
dd/mm/yyyy
Jan 2019
S M T W T F S
1 2 3 4 5
6 7 8 9 10 11 12
13 14 15 16 17 18 19
20 21 22 23 24 25 26
27 28 29 30 31
Feb 2019
S M T W T F S
1 2
3 4 5 6 7 8 9
10 11 12 13 14 15 16
17 18 19 20 21 22 23
24 25 26 27 28
Close

Hand's on

- https://testautomationpractice.blogspot.com/
- http://www.phptravels.net/
- https://www.expedia.ca/

Mouse actions using “Actions” Class

- **Mouse Actions**
  - Mouse Hover
    - moveToElement(WebElement).build().perform();
  - Right Click
    - contextClick(WebElement).build().perform();
  - Double Click
    - doubleClick(WebElement).build().perform();
  - Drag and Drop
    - dragAndDrop(Source WebElement, Target WebElement).build().perform()
    - clickAndHold(Source WebElement).moveToElement(Target WebElement).release().build().perform();
  - Slider
    - moveToElement(WebElement).dragAndDropBy(WebElement, 300, 0).build().perform();
  - Resizing
    - moveToElement(WebElement).dragAndDropBy(WebElement, 100, 100).build().perform();

Tooltips

- driver.findElement(WebElement*).getAttribute("title");*

Hand's on

- **Mouse Hover**
- http://opensource.demo.orangehrmlive.com/
- https://www.actitime.com/download
- **Double click**
- http://api.jquery.com/dblclick/
- https://testautomationpractice.blogspot.com/
- **Mouse Right click**
- http://swisnl.github.io/jQuery-contextMenu/demo.html
- **Drag and Drop**
- http://www.dhtmlgoodies.com/scripts/drag-drop-custom/demo-drag-drop-3.html
- https://testautomationpractice.blogspot.com/
- http://jqueryui.com/droppable/#default
- **Slider**
- https://testautomationpractice.blogspot.com/
- https://jqueryui.com/slider/
- **Resizing**
- https://jqueryui.com/resizable/
- https://testautomationpractice.blogspot.com/
- **Tooltip**
- https://www.facebook.com/
- https://jqueryui.com/tooltip/
- **All Mouse actions**
- https://testautomationpractice.blogspot.com/

Scrolling

- **Scroll down by 1000 pixels**

  js.executeScript("window.scrollBy(0,500)");

- **Scroll the page till the element is found**

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

- **Scroll the web page till end.**

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

Hand’s on

- https://www.countries-ofthe-world.com/flags-of-the-world.html
- https://testautomationpractice.blogspot.com/

What is JavaScriptExecutor?

- ***JavaScriptExecutor*** is an interface which provides mechanism to execute Javascript through selenium driver.

Actions we can perform using JavaScript Executor

- Flashing an element
- Drawing a boarder around the element
- Capture title of the page
- Click in some element
- Generate alert info
- Refreshing page
- Scrolling page

Hand’s on

- https://www.twoplugs.com/

File Upload using AutoIT

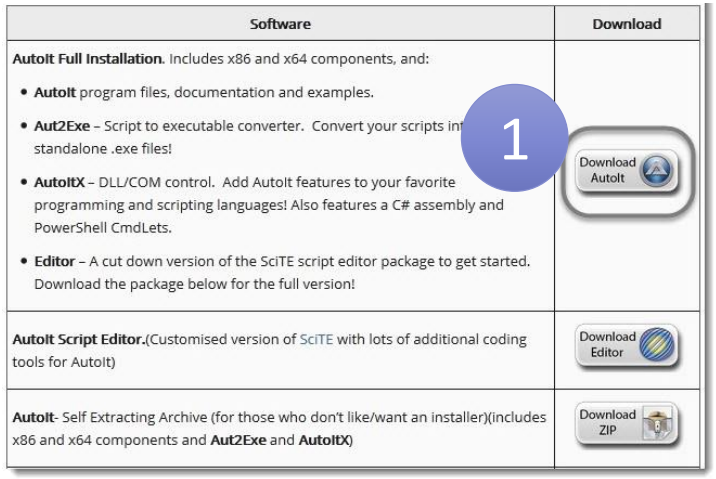
AutoIT

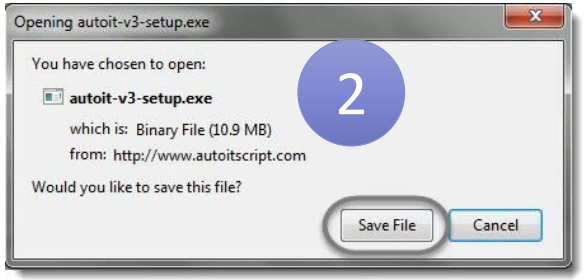
- **AutoIt** v3 is a freeware BASIC-like scripting language designed for automating the Windows GUI and general scripting.

AutoIT Download Links

- **AutoIT**
- https://www.autoitscript.com/cgi-bin/getfile.pl?autoit3/autoit-v3-setup.exe
- **AutoIT Editor**
- https://www.autoitscript.com/cgi-bin/getfile.pl?../autoit3/scite/download/SciTE4AutoIt3.exe

Steps to install AutoIT





Steps to install AutoIT Editor

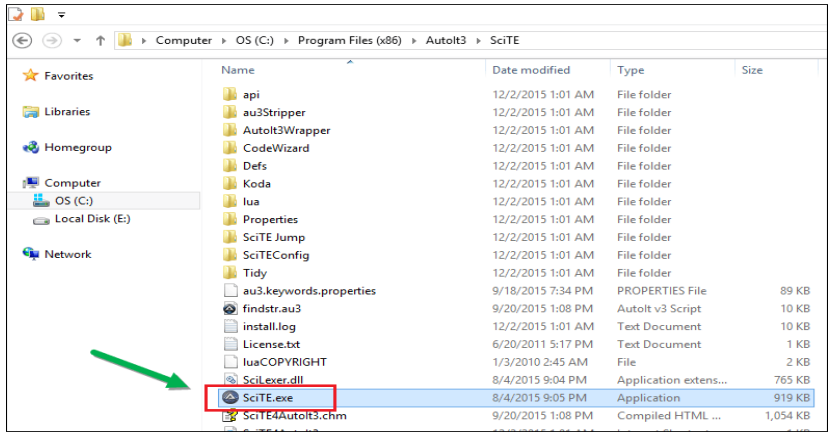
Software	Download
AutoIt Full Installation. Includes x86 and x64 components, and: • **AutoIt** program files, documentation and examples. • **Aut2Exe** – Script to executable converter. Convert your scripts into standalone .exe files! • **AutoItX** – DLL/COM control. Add AutoIt features to your favorite programming and scripting languages! Also features a C# assembly and PowerShell CmdLets. • **Editor** – A cut down version of the SciTE script editor package to get started. Download the package below for the full version!	Download AutoIt
AutoIt Script Editor.(Customised version of SciTE with lots of additional coding tools for AutoIt)	Download Editor

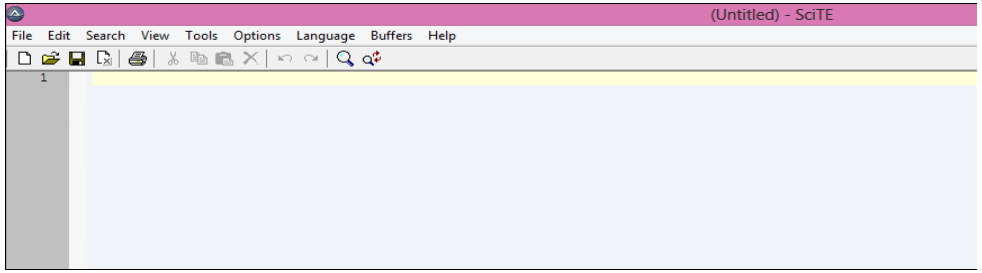
1

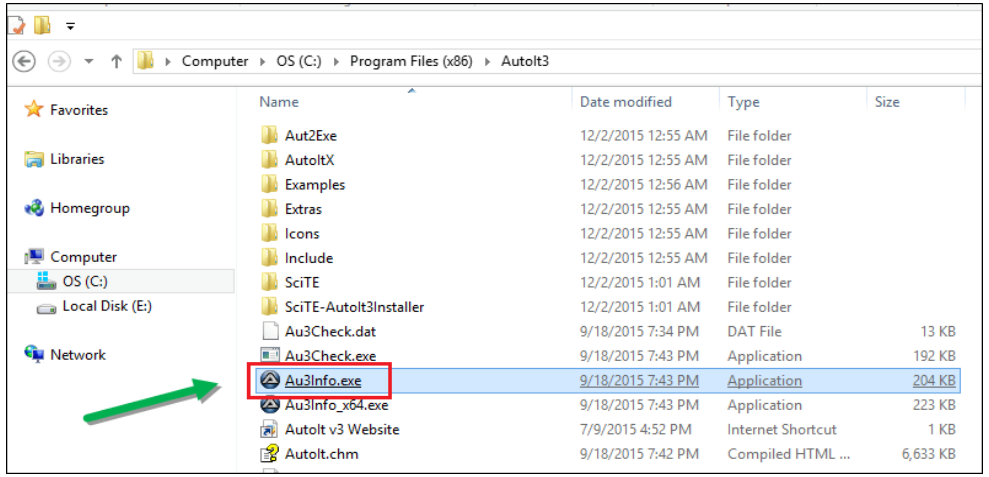
Version	Date updated	Notes
SciTE4AutoIt3.exe (6327Kb)	1/22/2014	Installer containing SciTE and all configuration files plus utilities.Update History. Definition files included: AutoIt v3.3.10.2 and BETA v3.3.11.1

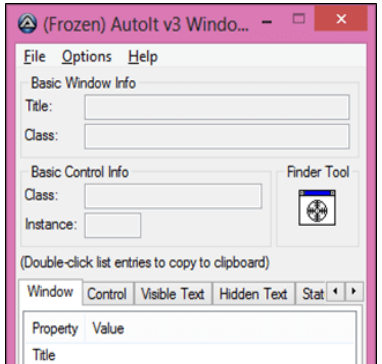
2

Invoking AutoIT & Script Editor









Upload file in Selenium using AutoIt

- Step 1: Identify the Windows control
- Step 2: Build a AutoIt script using identified windows control
- Step 3: Compile the .au3 script and convert it in to .exe file
- Step 4: Call the .exe file in to the Selenium test case

- **Sets input focus to a given control on a window.**
  - ControlFocus(" title "," text ",controlID)
- **Sets text of a control.**
  - ControlSetText(" title "," text ",controlID ," File path which need to upload ")
- **Sends a mouse click command to a given control.**
  - ControlClick(" title "," text ",controlID)
- Example

```
ControlFocus("File Upload","","Edit1")
ControlSetText("File Upload","","Edit1","C:\Users\admin\Desktop\TestFile.txt")
ControlClick("File Upload","","Button1")
```

Upload File using AutoIT Example

```
import java.io.IOException;
import org.openqa.selenium.By;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.WebElement;
import org.openqa.selenium.chrome.ChromeDriver;
import org.openqa.selenium.firefox.FirefoxDriver;

public class UploadFileUsingAutoIT {

    public static void main(String[] args) throws IOException, InterruptedException {

        System.setProperty("webdriver.chrome.driver", "C:/Drivers/chromedriver_win32/chromedriver.exe");
        WebDriver driver = new ChromeDriver();

        driver.get("http://demo.automationtesting.in/Register.html");

        WebElement button=driver.findElement(By.xpath("//*[@id=\"imagesrc\"]"));

        button.click();

        Runtime.getRuntime().exec("C://SeleniumPractice//autoITFiles//fileUpload.exe"); // execute .exe file

    }

}
```

File Upload using Sikuli

Sikuli

- Support all OS
- Sikuli identifies elements based on screenshots/images
- **Sikuli setup**
- Download Sikuli jar file
  - Download link: https://raiman.github.io/SikuliX1/sikulixapi.jar
- Add jar file to project build path in eclipse
- Capture screenshots for file text box & open button save them in a folder.

Upload File using Sikuli Example

```
public class FileUploadDemo {

    public static void main(String[] args) throws FindFailed {

        System.setProperty("webdriver.chrome.driver","C://Drivers/chromedriver_win32/chromedriver.exe");
        WebDriver driver=new ChromeDriver();
        driver.get("https://testautomationpractice.blogspot.com/");

        driver.manage().window().maximize();
        driver.switchTo().frame(0);

        driver.findElement(By.id("RESULT_FileUpload-11")).click();

        //sikuli
        Screen s=new Screen();

        String path="C:\\SeleniumPractice\\sikulifiles\\";
        Pattern fileInputTextBox=new Pattern(path+"filetxtbox.png");
        Pattern openButton=new Pattern(path+"openbtn.png");
        s.wait(fileInputTextBox,20);
        s.type(fileInputTextBox,path+"profilepic.jpg");
        s.click(openButton);

    }
}
```

Download Files

- Approach 1 : Setting Browser File
- Appraoch2: Robot API

Firefox Browser Profile

```
//download files in required location using chrome
FirefoxProfile profile=new FirefoxProfile();

// set Mime type according to your file format
profile.setPreference("browser.helperApps.neverAsk.saveToDisk", "text/plain,application/pdf");

profile.setPreference("browser.download.manager.showWhenStarting", false);

//download files in desired location
profile.setPreference("browser.download.dir","C:\\Downloadedfiles");
profile.setPreference("browser.download.folderList", 2);
profile.setPreference("pdfjs.disabled", true); // only for pdf file

FirefoxOptions option=new FirefoxOptions();
option.setProfile(profile);

System.setProperty("webdriver.gecko.driver","C://Drivers/geckodriver-v0.23.0-win64/geckodriver.exe");
WebDriver driver=new FirefoxDriver(option);
```

Chrome Browser Profile

```
HashMap<String, Object> chromePrefs = new HashMap<String, Object>();

chromePrefs.put("profile.default_content_settings.popups", 0);
chromePrefs.put("download.prompt_for_download", "false");
chromePrefs.put("download.default_directory","C:\\Downloadedfiles"); // set desired download path

ChromeOptions options = new ChromeOptions();
options.setExperimentalOption("prefs", chromePrefs);

DesiredCapabilities cap = DesiredCapabilities.chrome();
cap.setCapability(CapabilityType.ACCEPT_SSL_CERTS, true);
cap.setCapability(ChromeOptions.CAPABILITY, options);

System.setProperty("webdriver.chrome.driver","C://Drivers/chromedriver_win32/chromedriver.exe");
WebDriver driver=new ChromeDriver(cap);  // cap object we need pass
```

Robot API

```
import java.awt.AWTException;
import java.awt.Robot;
import java.awt.event.KeyEvent;
import org.openqa.selenium.By;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.firefox.FirefoxDriver;
import org.openqa.selenium.firefox.FirefoxOptions;
import org.openqa.selenium.firefox.FirefoxProfile;

public class DownloadFilesObFirefoxRobotAPI {

    public static void main(String[] args) throws InterruptedException, AWTException {

        System.setProperty("webdriver.gecko.driver","C://Drivers/geckodriver-v0.23.0-win64/geckodriver.exe");
        WebDriver driver=new FirefoxDriver();

        driver.get("http://the-internet.herokuapp.com/download");

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

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

        //use Robot class to handle downlaod scenario
        Robot robot = new Robot();

        robot.keyPress(KeyEvent.VK_DOWN);// Click on Down arrow to select save radio button on popup window
        robot.keyPress(KeyEvent.VK_ENTER); // Pressed on OK button , it will down load file

    }
}
```

Hand's on

- 1) File Upload
  - https://testautomationpractice.blogspot.com/
  - http://the-internet.herokuapp.com/upload
  - https://blueimp.github.io/jQuery-File-Upload/
- 2) Orange HRM App – Upload Photograph
  - http://opensource.demo.orangehrmlive.com
- Login→PIM→ Add Employee
- 3) Download Files
  - http://the-internet.herokuapp.com/download

Automate BarCode using ZXing API

- ZXing is one the third party API will be used to automate Bar Codes/QR Codes.
- Pre-requisites:
- Download Zxing API jar (javase-3.3.3.jar):
  - https://mvnrepository.com/artifact/com.google.zxing/javase/3.3.3
  - https://mvnrepository.com/artifact/com.google.zxing/core/3.3.3

- Bar-Code generator: https://barcode.tec-it.com

Maven dependencies

```
 <dependency>
    <groupId>com.google.zxing</groupId>
    <artifactId>javase</artifactId>
    <version>3.3.3</version>
 </dependency>
```

```
<dependency>
   <groupId>com.google.zxing</groupId>
   <artifactId>core</artifactId>
   <version>3.3.3</version>
</dependency>
```

Automate QRCode using ZXing API

- ZXing is one the third party API will be used to automate Bar Codes/QR Codes.
- Pre-requisites:
- Download Zxing API jar (javase-3.3.3.jar):
  - https://mvnrepository.com/artifact/com.google.zxing/javase/3.3.3
  - https://mvnrepository.com/artifact/com.google.zxing/core/3.3.3

- QR-Code generator: https://www.the-qrcode-generator.com/

Hand’s on

- 1) Bar Code & QR Code
- https://testautomationpractice.blogspot.com/

Cookies

- A Cookie is comprised of information about the user and their preferences.
- It stores information using a key-value pair.
- It is a small piece of data sent from Web Application and stored in Web Browser, while the user is browsing that website

Selenium commands for Cookies

- driver.manage().getCookies(); // Return The List of all Cookies
- driver.manage().getCookieNamed(arg0); //Return specific cookie according to name
- driver.manage().addCookie(arg0); //Create and add the cookie
- driver.manage().deleteCookie(arg0); // Delete specific cookie
- driver.manage().deleteCookieNamed(arg0); // Delete specific cookie according Name
- driver.manage().deleteAllCookies(); // Delete all cookies

Data Driven Testing using Microsoft Excel

Agenda

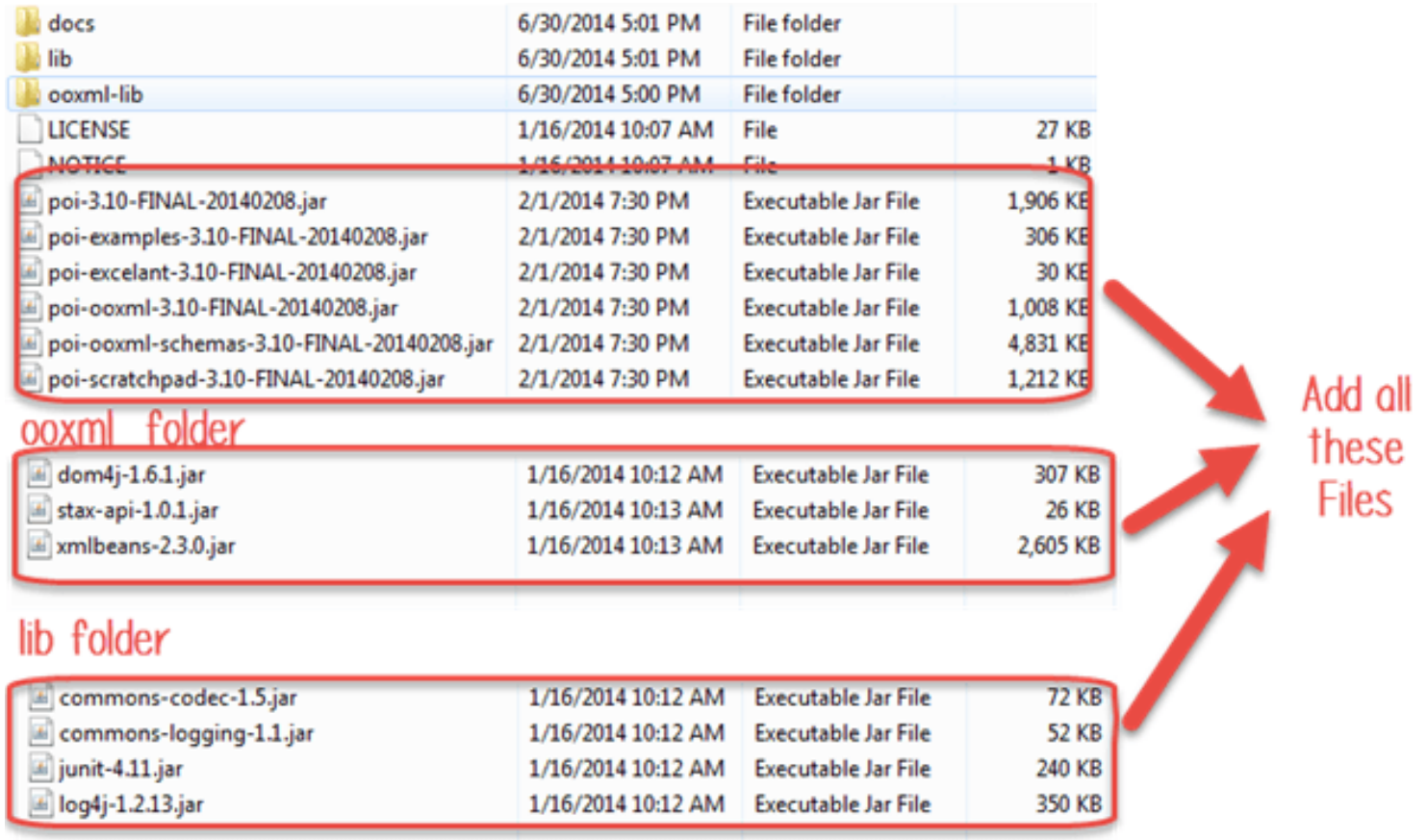
- Read data from Excel
- Write data into Excel
- Data Driven Testing

Apache POI API

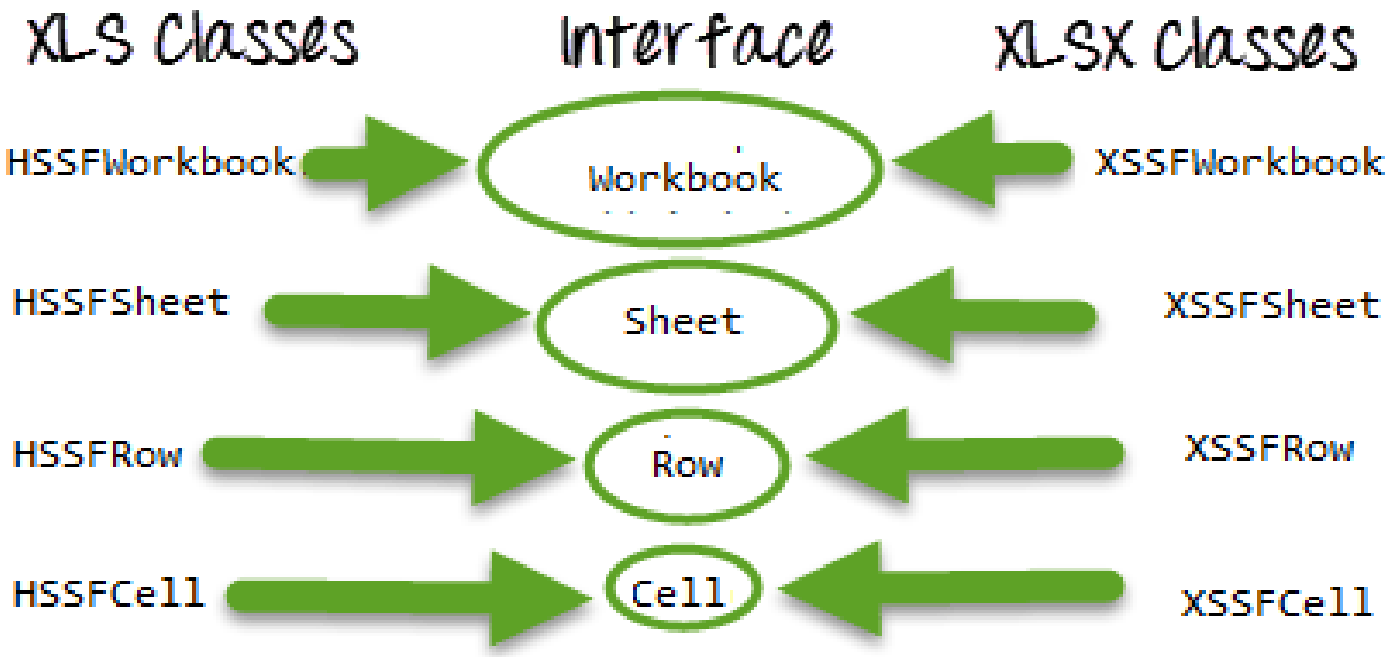
- WebDriver does not support excel directly.
- We can take help of third party drivers.
- We can read and write on excel file with the help of Java IO package and Apache POI library.
- **Latest version POI (P**oor **O**bfuscation **I**mplementation)**jars from**

http://poi.apache.org/download.html

Associate Apache POI drivers to Eclipse WebDriver Project

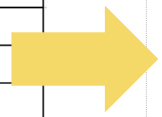


Classes and Interfaces in Apache POI



Read Data from Microsoft Excel

	A	B	C	D
1	EMPID	FIRSTNAME	LASTNAME	SALARY
2	111	ABC	ABC	50000
3	112	PQR	PQR	70000
4	113	MNO	MNO	40000
5	114	XYZ	XYZ	60000
6	115	ABC1	ABC1	50000
7	116	PQR1	PQR1	70000
8	117	MNO1	MNO1	40000
9	118	XYZ1	XYZ1	60000
10	119	ABC2	ABC2	50000
11	120	PQR2	PQR2	70000
12	121	MNO2	MNO2	40000
13	122	XYZ2	XYZ2	60000



```java
import java.io.FileInputStream;
import java.io.IOException;
import org.apache.poi.ss.usermodel.DataFormatter;
import org.apache.poi.xssf.usermodel.XSSFRow;
import org.apache.poi.xssf.usermodel.XSSFSheet;
import org.apache.poi.xssf.usermodel.XSSFWorkbook;

public class ReadingExcel {

    public static void main(String[] args) throws IOException {

        FileInputStream file = new FileInputStream("C://SeleniumPractice/data3.xlsx");
        XSSFWorkbook workbook = new XSSFWorkbook(file);
        XSSFSheet sheet = workbook.getSheet("Sheet1");

        int rownum = sheet.getLastRowNum(); // returns number of rows present in excel sheet
        int colcount = sheet.getRow(0).getLastCellNum(); // returns number of cells present in a row

        for (int i = 0; i <= rownum; i++) {
            XSSFRow row = sheet.getRow(i);

            for (int j = 0; j < colcount; j++) {
                String value = row.getCell(j).toString(); // reading the data from cell
                System.out.print(value + " ");

            }

            System.out.println();

        }
    }
}
```

Write Data into Excel

```java
import java.io.FileOutputStream;
import java.io.IOException;
import org.apache.poi.xssf.usermodel.XSSFRow;
import org.apache.poi.xssf.usermodel.XSSFSheet;
import org.apache.poi.xssf.usermodel.XSSFWorkbook;

public class WritingDataIntoExcel {

    public static void main(String[] args) throws IOException {

        FileOutputStream file = new FileOutputStream("C://SeleniumPractice/testingdata.xlsx");

        XSSFWorkbook workbook = new XSSFWorkbook();

        XSSFSheet sheet = workbook.createSheet("data");

        for (int i = 0; i <= 5; i++) // rows
        {
            XSSFRow row = sheet.createRow(i);

            for (int j = 0; j < 3; j++) // cells/columns
            {
                row.createCell(j).setCellValue("abcedfg");
            }
        }

        workbook.write(file);

        System.out.println("writing excel is completed");
    }
}
```

	A	B	C
1	abcdef	abcdef	abcdef
2	abcdef	abcdef	abcdef
3	abcdef	abcdef	abcdef
4	abcdef	abcdef	abcdef
5	abcdef	abcdef	abcdef
6	abcdef	abcdef	abcdef

Excel Common Utility - Methods

```
public static void setExcelFile(String xlfile, String xlsheet) throws Exception {

    try {

        FileInputStream ExcelFile = new FileInputStream(xlfile);
        wb = new XSSFWorkbook(ExcelFile);
        ws = wb.getSheet(xlsheet);
    } catch (Exception e) {
        throw (e);
    }
}
```

```
public static int getRowCount(String xlfile, String xlsheet) throws IOException {
    fi = new FileInputStream(xlfile);
    wb = new XSSFWorkbook(fi);
    ws = wb.getSheet(xlsheet);
    int rowcount = ws.getLastRowNum();
    wb.close();
    fi.close();
    return rowcount;
}
```

```
public static int getCellCount(String xlfile, String xlsheet, int rownum) throws IOException {
    fi = new FileInputStream(xlfile);
    wb = new XSSFWorkbook(fi);
    ws = wb.getSheet(xlsheet);
    row = ws.getRow(rownum);
    int cellcount = row.getLastCellNum();
    wb.close();
    fi.close();
    return cellcount;
}
```

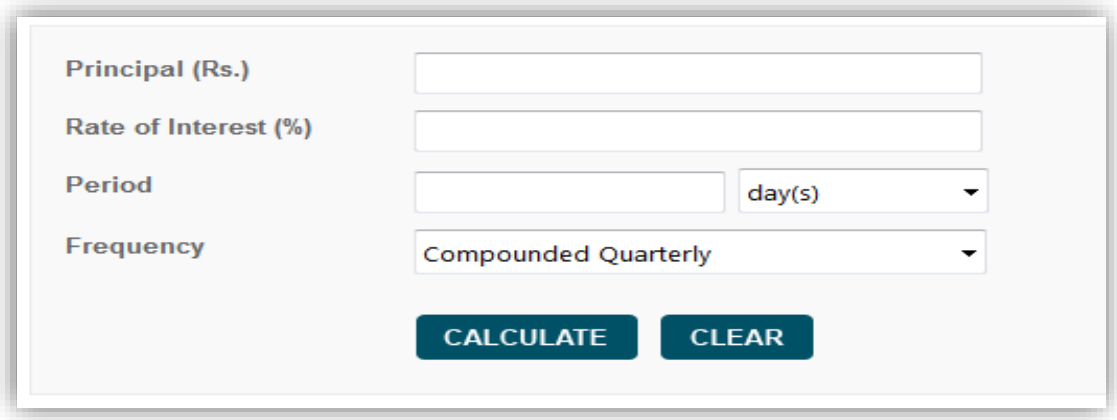
```
public static String getCellData(String xlfile, String xlsheet, int rownum, int colnum) throws IOException {
    fi = new FileInputStream(xlfile);
    wb = new XSSFWorkbook(fi);
    ws = wb.getSheet(xlsheet);
    row = ws.getRow(rownum);
    cell = row.getCell(colnum);
    String data;
    try {
        DataFormatter formatter = new DataFormatter();
        String cellData = formatter.formatCellValue(cell);
        return cellData;
    } catch (Exception e) {
        data = "";
    }
    wb.close();
    fi.close();
    return data;
}
```

```
public static void setCellData(String xlfile, String xlsheet, int rownum, int colnum, String data)
        throws IOException {
    fi = new FileInputStream(xlfile);
    wb = new XSSFWorkbook(fi);
    ws = wb.getSheet(xlsheet);
    row = ws.getRow(rownum);
    cell = row.createCell(colnum);
    cell.setCellValue(data);
    fo = new FileOutputStream(xlfile);
    wb.write(fo);
    wb.close();
    fi.close();
    fo.close();
}
```

Data Driven Test-1 (Read Data from Excel)

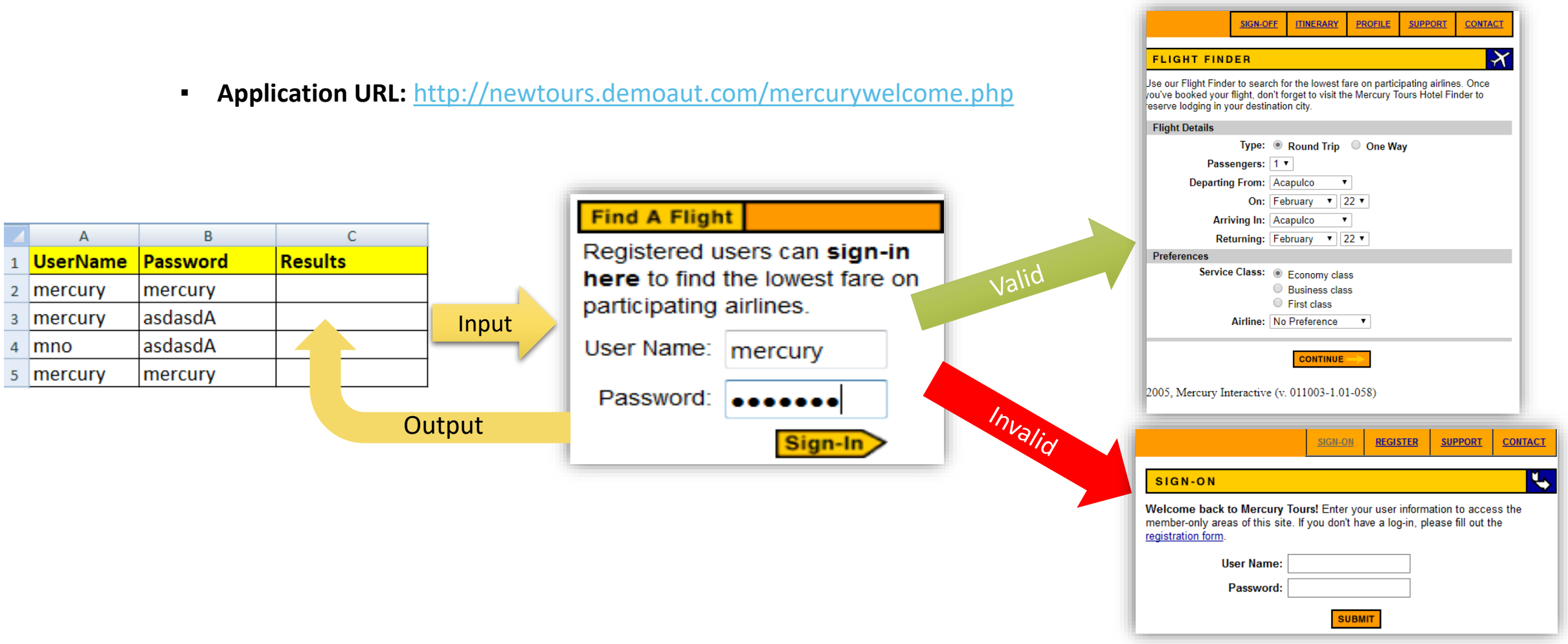
- **Application URL:**
- http://www.moneycontrol.com/fixed-income/calculator/state-bank-of-india-sbi/fixed-deposit-calculator-SBI-BSB001.html

	A	B	C	D	E
1	**Principle**	**Rate of Interest**	**Period(Years)**	**Frequency**	**Maturity Value**
2	2000	10	2	Simple Interest	2400
3	4000	15	5	Simple Interest	8352
4	50000	10	3	Simple Interest	65000
5	75000	6	2	Simple Interest	84000



Data Driven Test-2
(Read Data & Write results into Excel)

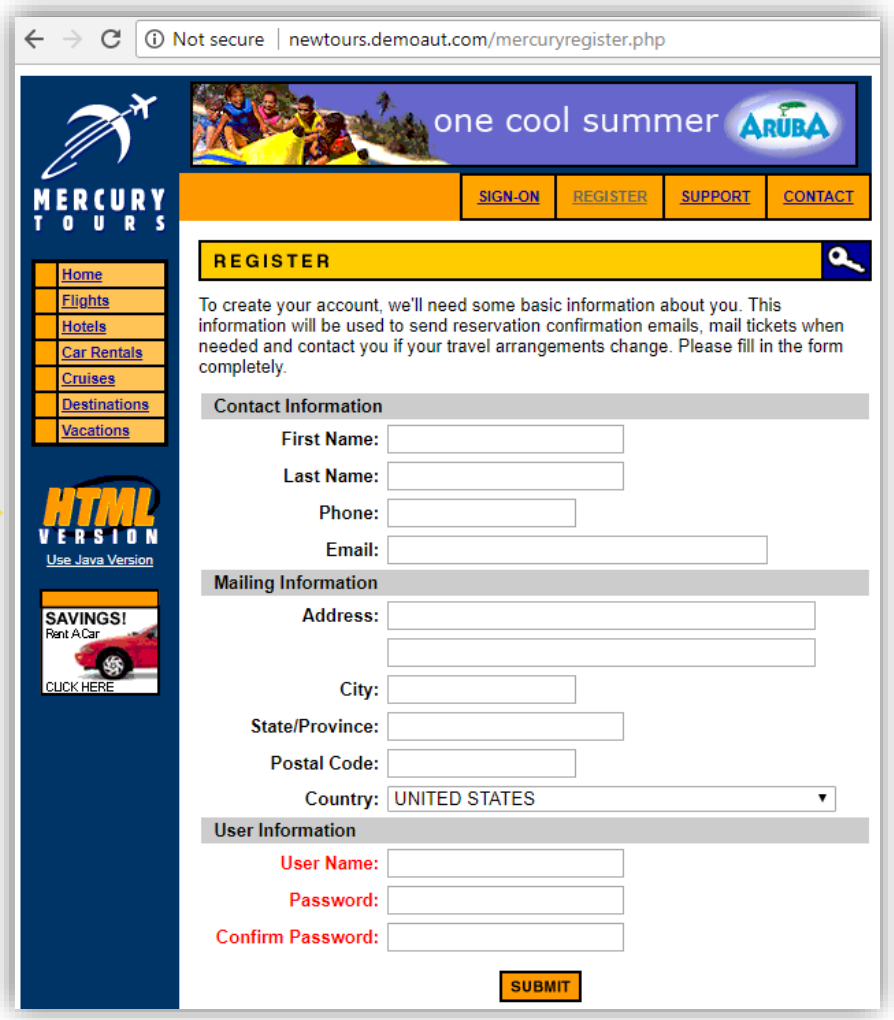
- **Application URL:** http://newtours.demoaut.com/mercurywelcome.php



Data Driven Test-3 (Read Data from Excel)

- **Application URL**
- http://newtours.demoaut.com/mercuryregister.php

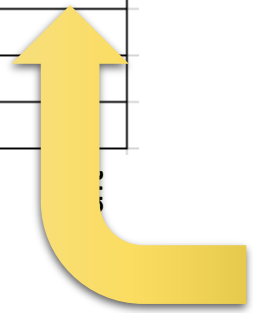
	A	B	C	D	E	F	G	H	I	J	K
1	Fist_Name	Last_Name	Phone	Email	Address	City	State	PostelCode	Country	UserName	Password
2	AAAAA	aaaaa	1234567890	a@gmail.com	HYDERABAD	HYDERABAD	AP	500073	INDIA	aaaaa	aaaaa
3	BBBBB	bbbbb	1234567891	b@gmail.com	CHENNAI	CHENNAI	TN	500074	INDIA	bbbbb	bbbbb
4	CCCCC	ccccc	1234567892	c@gmail.com	BANGOLORE	BANGOLORE	KR	500075	INDIA	ccccc	ccccc
5	DDDDD	ddddd	1234567890	d@gmail.com	DELHI	DELHI	UP	500071	INDIA	ddddd	ddddd
6	EEEEE	eeeee	1234567890	e@gmail.com	MUMBAI	MUMBAI	MH	500079	INDIA	eeeee	eeeee
7	XXXXXX	xxxxxxx	1234567891	x@gmail.com	CHENNAI	CHENNAI	TN	500080	INDIA	xxxxxxx	xxxxxxx
8	YYYYYY	yyyyyy	1234567892	y@gamil.com	BANGOLORE	BANGOLORE	KR	500081	INDIA	yyyyyy	yyyyyy
9	ZZZZZZZ	zzzzzzz	1234567893	z@gmail.com	DELHI	DELHI	UP	500082	INDIA	zzzzzzz	zzzzzzz

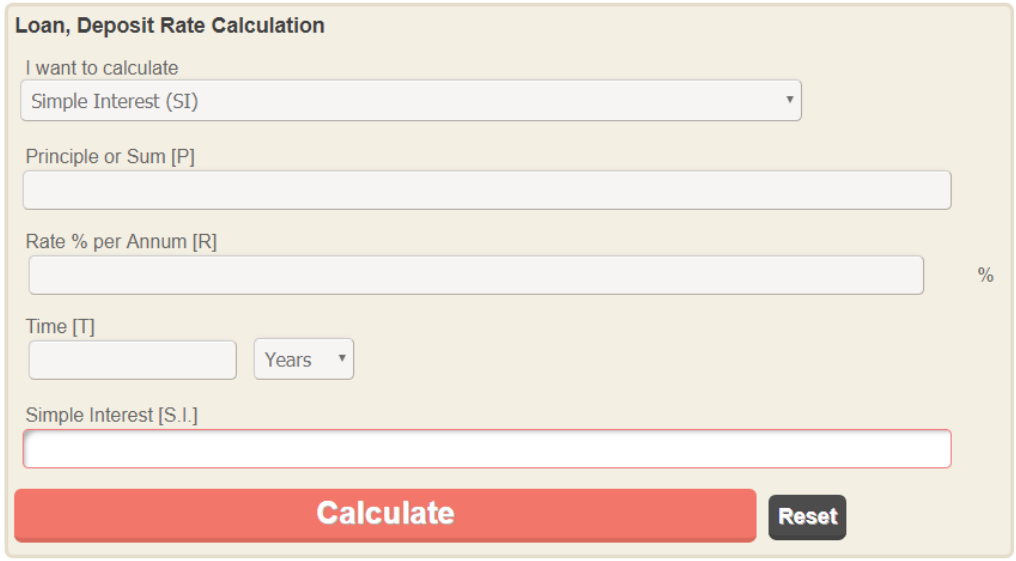


Data Driven Test-4
(Read Data & Write results into Excel)

- **Application URL:** https://www.easycalculation.com/simple-interest.php

	A	B	C	D	E
1	Principle	Interest Rate	Time Period(Years)	Simple Interest	Result
2	200000	10	2	40000	
3	400000	15	5	300000	
4	500000	10	3	150000	
5	750000	6	2	90000	





Headless Browsers

- Headless Browser & headless testing
- When to use headless browser
- Disadvantage of headless browser
- Headless automation in Selenium
- Headless Chrome Browser
- Headless Firefox Browser
- HtmlUnitDriver

Headless Browsers in Selenium Webdriver

- A headless browser is a browser simulation program that **does not have a user interface** (UI less).
- Headless browser programs operate like any other browser, but **do not display any UI**. Selenium **executes its' tests in the background**.
- There are several headless browsers available in the market, the following are the most popular ones:
  - Chrome
  - Firefox
  - HTMLUnit driver
  - PhantomJS

What is headless testing ?

- Executing the web applications' **UI tests without opening a browser's user interface** is called headless browser testing. **Headless browser acts similar to a normal web browser**.

- Testers have full control over the web pages loaded into the headless browsers. Only difference is **you will not see a graphical user interface**.

When to use headless browser testing ?

- We can use headless testing once the cross browser testing is completed and want to run regression test cases in subsequent releases and with continuous integration.
- You have no option other than using headless testing when your machine does not have a GUI, for instance if you want to run your tests in unix.
- It is recommended to use headless browser when tests are executed in parallel as User Interface based browsers consumes a lot of Memory / resources.

Disadvantage of headless browser testing

- Headless browsers are a bad idea. They get you some testing, but nothing like what a real user will see, and they mask lots of problems that only real browsers encounter.
- Hard to debug inconsistent failures on locating elements due to page loading too fast.
- In Real browser as functions are performing in front of user and he can interact with it so he can easily detect where the tests goes fail. And can easily debug if anything goes wrong.
- Headless browsers aren't representing real users, as no user uses the your application without UI. Because it does not have UI, it may not report errors related with images.
- Managing to take screenshot is very difficult in UI less browser

Headless browser automation in Selenium Java

- We can automate the headless browser in selenium, only automation can be performed on headless browser.
- For users, there is no such thing called Headless or UI less browser as their eyes cannot see the UI less browser.
- In this headless browser we can execute the tests created on UI browsers, so debugging occurs on UI browsers only.

  Below are the browsers we are going to automate :
  1. Chrome
  2. Firefox
  3. HtmlUnit browser

Headless Chrome in Selenium

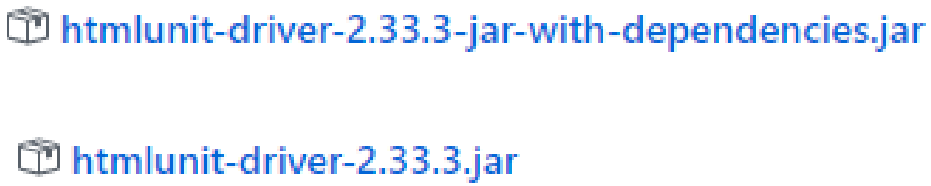
```java
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.chrome.ChromeDriver;
import org.openqa.selenium.chrome.ChromeOptions;

public class HeadLessChrome {

    public static void main(String[] args) {

        //set the driver server exe path
        System.setProperty("webdriver.chrome.driver", "C:/Drivers/chromedriver_win32/chromedriver.exe");

        // set chrome as Headless
        ChromeOptions options = new ChromeOptions();
        options.setHeadless(true);  //options.addArguments("--headless");

        WebDriver driver = new ChromeDriver(options);       //Instantiate Chrome Driver


        driver.get("http://demo.nopcommerce.com/");

        System.out.println("Title of the page:" + driver.getTitle());
        System.out.println("URL of the page:" + driver.getCurrentUrl());

    }
}
```

Headless Firefox Browser in Selenium

```
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.firefox.FirefoxDriver;
import org.openqa.selenium.firefox.FirefoxOptions;

public class HeadLessFirefox {

    public static void main(String[] args) {

        //set the driver server exe path
        System.setProperty("webdriver.gecko.driver", "C://Drivers/geckodriver-v0.23.0-win64/geckodriver.exe");

        // set Firefox as Headless
        FirefoxOptions options = new FirefoxOptions();
        options.setHeadless(true);

        WebDriver driver=new FirefoxDriver(options);        //Instantiate Firefox Driver


        driver.get("http://demo.nopcommerce.com/");

        System.out.println("Title of the page:" + driver.getTitle());
        System.out.println("URL of the page:" + driver.getCurrentUrl());


    }
}
```

HtmlUnitDriver in Selenium

- HtmlUnitDriver is the built-in headless browser in selenium.
- webdriver, HtmlUnitDriver is present in ***org.openqa.selenium.htmlunit*** package
- Unlike Headless Firefox, Chrome, With HtmlUnitDriver we just need to create a object for the class to create a headless browser
- HTMLUnit is completely developed using java.
  - https://github.com/SeleniumHQ/htmlunit-driver/releases



Download HtmlUnitDriver

Third Party Drivers, Bindings, and Plugins

Selenium can be extended through the use of plugins. Here are a number of plugins created and maintained by third parties. For more information on how to create your own plugin or have it listed, consult the docs.

Please note that these plugins are not supported, maintained, hosted, or endorsed by the Selenium project. In addition, be advised that the plugins listed below are not necessarily licensed under the Apache License v.2.0. Some of the plugins are available under another free and open source software license; others are only available under a proprietary license. Any questions about plugins and their license of distribution need to be raised with their respective developer(s).

Third Party Browser Drivers NOT DEVELOPED by seleniumhq

Browser

Mozilla GeckoDriver	latest	change log	issue tracker	Implementation Status
Google Chrome Driver	latest	change log	issue tracker	selenium wiki page
Opera	latest		issue tracker	selenium wiki page
Microsoft Edge Driver			issue tracker	Implementation Status
GhostDriver	(PhantomJS)		issue tracker	SeConf talk
HtmlUnitDriver	latest		issue tracker	
SafariDriver			issue tracker	

2.33.3

rbri released this on Nov 17 · 1 commit to master since this release

Assets 8

- htmlunit-driver-2.33.3-jar-with-dependencies.jar
- htmlunit-driver-2.33.3-jar-with-dependencies.jar.asc
- htmlunit-driver-2.33.3-javadoc.jar
- htmlunit-driver-2.33.3-javadoc.jar.asc
- htmlunit-driver-2.33.3.jar
- htmlunit-driver-2.33.3.jar.asc
- Source code (zip)
- Source code (tar.gz)

HtmlUnitDriver in Selenium (Cont....)

```
import org.openqa.selenium.htmlunit.HtmlUnitDriver;

public class HeadLessHtmlUnitDriver {

    public static void main(String[] args) {

        // create instance for the HtmlUnitWebDriver
        HtmlUnitDriver driver = new HtmlUnitDriver();

        driver.get("http://demo.nopcommerce.com/");

        System.out.println("Title of the page:" + driver.getTitle());
        System.out.println("URL of the page:" + driver.getCurrentUrl());

        driver.quit();
    }

}
```

More Elements to practice

- http://the-internet.herokuapp.com/

TestNG

Agenda

- What is TestNG
- Advantages of TestNG
- Install TestNG in Eclipse
- How to write TestNG Test case
- Annotations in TestNG

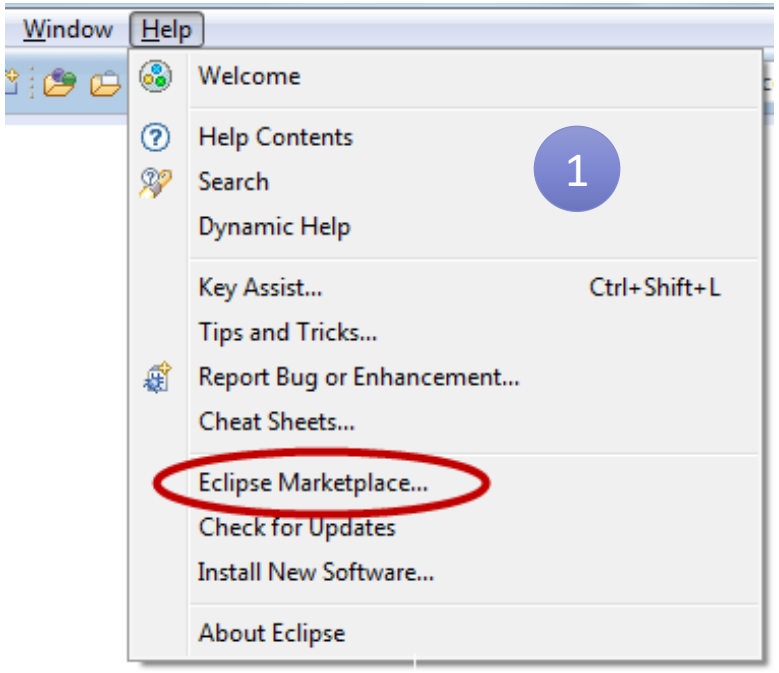
What is TestNG?

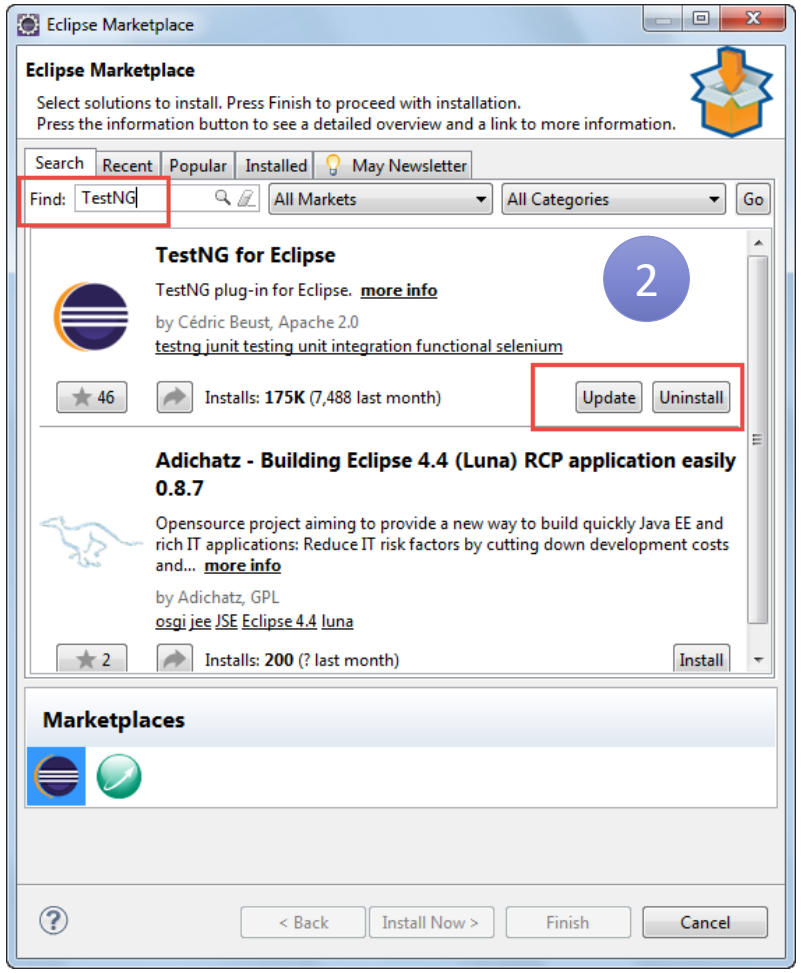
- **TestNG** stands for *Test Next Generation* and it is an open-source test automation framework inspired by JUnit and NUnit.

Advantages of Using TestNG

- HTML reports can be generated
- Supports parallel testing
- Test Cases can be prioritized easily.
- Uses annotations for executing methods
- Supports parameterization
- TestNG does not require main() method.
- **@Test** is used to tell that the method under it is a test case.

Install TestNG in Eclipse





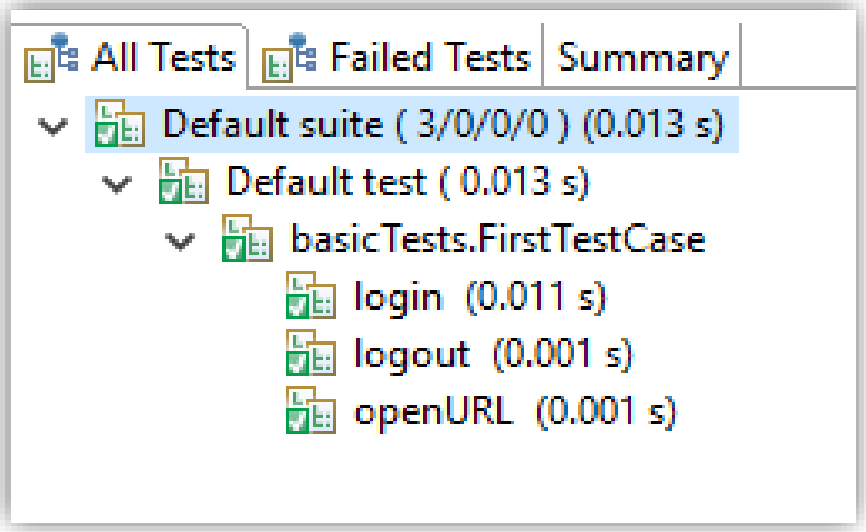
TestNG Test Case

```
import org.testng.annotations.Test;

public class FirstTestCase {

@Test
void openURL()
{
System.out.println(" Open URL of the application");
}

@Test
void login()
{
System.out.println(" this is login functionality");
}

@Test
void logout()
{
System.out.println(" this is logout functionality");
}
}
```

Console Output

```
[RemoteTestNG] detected TestNG version 6.13.1
 this is login functionality
 this is logout functionality
OPen URL of the application
PASSED: login
PASSED: logout
PASSED: openURL
```

Tree Output

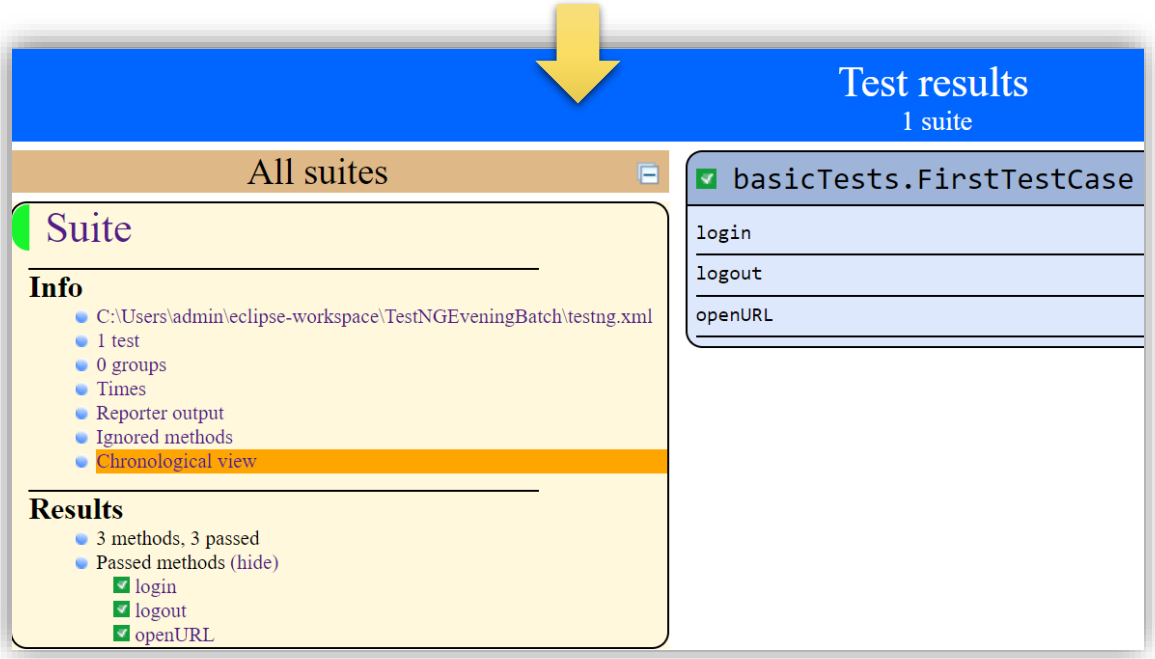


TestNG Report

```java
import org.testng.annotations.Test;

public class FirstTestCase {

@Test
void openURL()
{
System.out.println(" Open URL of the application");
}

@Test
void login()
{
System.out.println(" this is login functionality");
}

@Test
void logout()
{
System.out.println(" this is logout functionality");
}
}
```

Run through XML

```xml
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE suite SYSTEM "http://testng.org/testng-1.0.dtd">
<suite name="Suite">
  <test thread-count="5" name="Test">
    <classes>
      <class name="basicTests.FirstTestCase"/>
    </classes>
  </test> <!-- Test -->
</suite> <!-- Suite -->
```



Annotations

- **@Test** – The method annotated with *@Test* is the main test method in the entire program. Other annotated methods will be executed around this method.
- **@BeforeMethod** – The method annotated with *@BeforeMethod* will run before any test method inside a class is run.
- **@AfterMethod** – The method annotated with *@AfterMethod* will run after every test method inside a class is run.
- **@BeforeClass** – The method annotated with *@BeforeClass* will run once before the first test method in the current class is invoked.
- **@AfterClass** – The method annotated with *@AfterClass* will run once after all the test methods in the current class have run.
- **@BeforeTest** – The method annotated with *@BeforeTest* will run before any test method belonging to a class is run.
- **@AfterTest** – The method annotated with *@AfterTest* will run after all the test methods belonging to a class have run.

Agenda

- Understanding testng.xml
- TestNG Report
- Prioritizing tests
- Dependency methods
- Disable tests
- TestNG batch testing

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE suite SYSTEM "http://testng.org/testng-1.0.dtd">
<suite name="Suite">
  <test  name="Test">
     <classes>
          <class name="basicTests.FirstTestCase"/>
      </classes>
  </test> <!-- Test -->
</suite> <!-- Suite -->
```

Sequencing & Prioritizing

- Annotated methods can be prioritized in TestNG.
- Priority can be assigned as parameters while calling the test cases
- If no priority is defined, then test cases will be executed in alphabetical order.

```
import org.testng.annotations.Test;

public class FirstTestCase {

    @Test(priority=1)
    void search()
    {
        System.out.println("This is search test");
    }

    @Test(priority=2)
    void logout()
    {
        System.out.println(" This is logout test");
    }

    @Test(priority=3)
    void openURL()
    {
        System.out.println(" This is open URL");
    }

    @Test(priority=4)
    void login()
    {
        System.out.println("This is login test");
    }

}
```

Disabling Test case

- In TestNG, we can enable or disable tests as per needs.
- We can control that by setting enable attribute is n @ Test annotation.

enabled=true or enabled=false

This test never be executed in the test case

```
import org.testng.annotations.Test;

public class DisablingTestMethods {

    @Test(priority=1)
    void openURL()
    {
        System.out.println("URL opened");
    }

    @Test(priority=2)
    void login()
        {
        System.out.println("login test");

        }

    @Test(priority=3,enabled=false)
    void search()
    {
        System.out.println("still in progress.....");
    }

    @Test(priority=4,enabled=false)
    void advancedsearch()
    {
        System.out.println("still in progress.....")

    }

    @Test(priority=5)
    void logout()
    {
        System.out.println("loogout test");
    }
}
```

Method dependency

```
import org.testng.Assert;
import org.testng.annotations.Test;

public class DependencyMethods {

    @Test(priority=1)
    void openURL()
    {
        System.out.println("URL opened");
        Assert.assertTrue(true); //pass
    }

    @Test(priority=2,dependsOnMethods= {"openURL"})
    void login()
        {
        System.out.println("login test");
        Assert.assertTrue(true); //pass

        }
    @Test(priority=3,dependsOnMethods= {"openURL","login"})
    void search()
    {
        System.out.println("search test");
        Assert.assertTrue(false); //failed
    }
    @Test(priority=4,dependsOnMethods= {"search"})
    void advancedsearch() //skipped because of dependant method is failed
    {
        System.out.println("advance search test");
        Assert.assertTrue(true);
    }
    @Test(priority=5,dependsOnMethods= {"login"})
    void logout()
    {
        System.out.println("loogout test");
        Assert.assertTrue(true); //pass
    }
}
```

WWW.PAVANONLINETRAININGS.COM

Always Running Tests

```java
import org.testng.Assert;
import org.testng.annotations.Test;

public class DependencyMethods {

    @Test(priority=1)
    void openURL()
    {
        System.out.println("URL opened");
        Assert.assertTrue(true); //pass
    }

    @Test(priority=2,dependsOnMethods= {"openURL"})
    void login()
        {
        System.out.println("login test");
        Assert.assertTrue(true); //pass

        }
    @Test(priority=3,dependsOnMethods= {"openURL","login"})
    void search()
    {
        System.out.println("search test");
        Assert.assertTrue(false); //failed
    }
    @Test(priority=4,dependsOnMethods= {"search"},alwaysRun=true)
    void advancedsearch() //skipped because of dependant method id failed
    {
        System.out.println("advance search test");
        Assert.assertTrue(true);
    }
    @Test(priority=5,dependsOnMethods= {"login"})
    void logout()
    {
        System.out.println("loogout test");
        Assert.assertTrue(true); //pass
    }
}
```

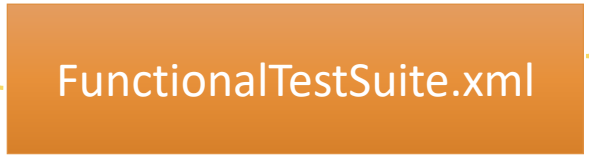
invocationCount in TestNG

- **invocationCount:** This is a TestNG attribute that defines number of times a test method should be invoked or executed before executing any other test method. If invocationCount = 5, then the test method will be executed 5 times before executing next test method.

  Check the below example, In getTitle() test method we are printing the title of the website with invocationCount = 5. And we have another test method secondTest() also.

```
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.chrome.ChromeDriver;
import org.testng.annotations.Test;

public class invocationCount {

    @Test(invocationCount = 5)
    public void getTitle() {
        System.setProperty("webdriver.chrome.driver", "C:/Drivers/chromedriver_win32/chromedriver.exe");
        WebDriver driver = new ChromeDriver();
        driver.get("http://www.pavantestingtools.com/");
        driver.manage().window().maximize();
        System.out.println("Website Title: "+driver.getTitle());
        driver.quit();
    }

    @Test
    public void secondTest() {
        System.out.println("This will be executed at the end");
    }
}
```

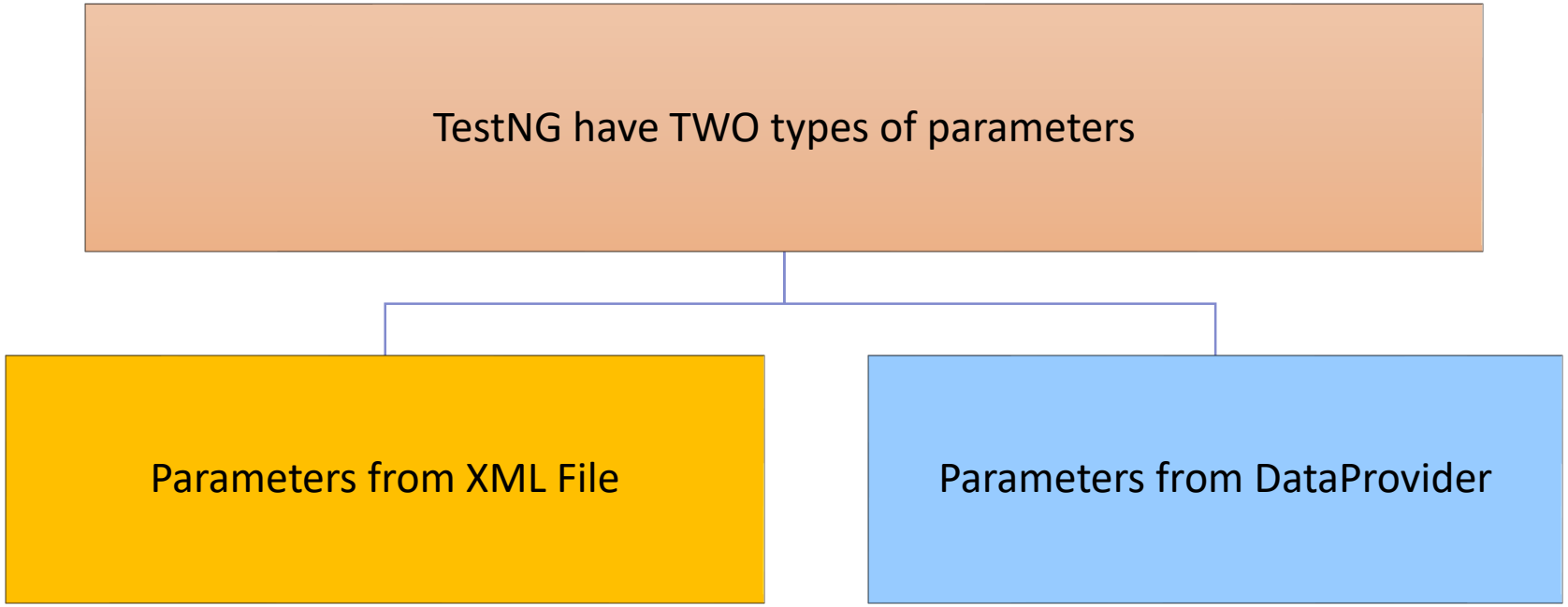
Batch Testing



Test cases	Test Methods
LoginTest.java	loginByemail() loginbyfacebook() loginbytwitter()
SignupTest.java	signupbyemail() signupbyfacebook() signupbytwitter()
PaymentTest.java	paymentindollor() paymentinrupees()
PaymentReturns.java	paymentReturnbybank()

Agenda

- Parameterization
- Passing parameters using XML
- Parallel testing
- Data Provider


TestNG have TWO types of parameters
Parameters from XML File
Parameters from DataProvider

Passing parameters through XML

Param.xml

```xml
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE suite SYSTEM "http://testng.org/testng-1.0.dtd">

<suite name="testsuite">
    <test name="sampletest">
    <parameter name="a" value="welcome" />
        <classes>
            <class name="parameterization.Test1" />
        </classes>
    </test>
</suite>
```

Test1.java

```java
package parameterization;
import org.testng.annotations.Parameters;
import org.testng.annotations.Test;

public class Test1 {

    @Parameters("a")
    @Test
    public void m1(String s)
    {
        System.out.println("the value from xml file is:" +s);
    }
}
```

Passing Multiple parameters through XML

Param2.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE suite SYSTEM "http://testng.org/testng-1.0.dtd">

<suite name="TestSuite" thread-count="3">
    <parameter name="user" value="mercury" />
    <parameter name="passwd" value="mercury" />
    <test name="LoginTest">
        <!-- <parameter name="user" value="mercury" />
        <parameter name="passwd" value="mercury" /> -->
        <classes>
            <class name="parameterization.ParameterWithTestNGXML">
            </class>
        </classes>
    </test>
</suite>
```

ParameterWithTestNGXML.java

```
package parameterization;
import org.openqa.selenium.By;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.chrome.ChromeDriver;
import org.testng.annotations.Parameters;
import org.testng.annotations.Test;

public class ParameterWithTestNGXML {

    @Test
    @Parameters({"user","passwd"})
    public void testParameterWithXML(String usr,String pwd)
    {
        System.setProperty("webdriver.chrome.driver", "C://Drivers//chromedriver_win32/chromedriver.exe");
        WebDriver driver=new ChromeDriver();

        driver.get("http://newtours.demoaut.com/");
        driver.findElement(By.name("userName")).sendKeys(usr);
        driver.findElement(By.name("password")).sendKeys(pwd);
        driver.findElement(By.name("login")).click();
        driver.close();
       }
}
```

Parallel Testing

```java
public class CrossBrowserTest {

    WebDriver driver=null;

    @Parameters("browser")
    @Test
    public void launchBrowser(String br)
    {
        if(br.equals("FF"))
        {
            //Firefox Browser
            System.setProperty("webdriver.gecko.driver","C://Drivers/geckodriver-v0.19.1-win64/geckodriver.exe");
            driver=new FirefoxDriver();
        }

        else if(br.equals("IE"))
        {
            System.setProperty("webdriver.ie.driver","C://Drivers/IEDriverServer_Win32_3.7.0/IEDriverServer.exe");
            driver=new InternetExplorerDriver();
        }

        else if(br.equals("CH"))
        {
            System.setProperty("webdriver.chrome.driver","C://Drivers/chromedriver_win32/chromedriver.exe");
            driver=new ChromeDriver(); // opens the browser
        }
        driver.get("http://newtours.demoaut.com/");

    }

    @Test
    public void verifyTitle()
    {
        Assert.assertEquals(driver.getTitle(), "Welcome: Mercury Tours");
    }

    @AfterClass
    public void closeBrowser()
    {
        driver.quit();
    }

}
```

CrossBrowserTest.java

```xml
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE suite SYSTEM "http://testng.org/testng-1.0.dtd">

<suite name="paralleltesting" parallel="tests">

    <test name="FirefoxTest">
    <parameter name="browser" value="FF" />
        <classes>
            <class name="parameterization.CrossBrowserTest" />
        </classes>
    </test>


    <test name="ChromeTest">
    <parameter name="browser" value="CH" />
        <classes>
            <class name="parameterization.CrossBrowserTest" />
        </classes>
    </test>


    <test name="IETest">
    <parameter name="browser" value="IE" />
        <classes>
            <class name="parameterization.CrossBrowserTest" />
        </classes>
    </test>

</suite>
```

Crossbrowser.xml

Parameters using Dataprovider

- @Parameters annotation is easy but to test with multiple sets of data we need to use Data Provider.
- To fill thousand's of web forms using our testing framework we need a different methodology which can give us a very large dataset in a single execution flow.
- This data driven concept is achieved by **@DataProvider** annotation in TestNG.

Parameters using Data provider

```java
import org.openqa.selenium.By;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.chrome.ChromeDriver;
import org.testng.Assert;
import org.testng.annotations.AfterClass;
import org.testng.annotations.BeforeClass;
import org.testng.annotations.DataProvider;
import org.testng.annotations.Test;

public class DataProviderExample {

    WebDriver driver;

    @BeforeClass
    void setup()
    {
        System.setProperty("webdriver.chrome.driver","C://Drivers/chromedriver_win32/chromedriver.exe");
        driver=new ChromeDriver();
    }

    @Test(dataProvider="users")
    void loginTest(String uname,String pwd)
    {
        driver.get("http://newtours.demoaut.com/");
        driver.findElement(By.name("userName")).sendKeys(uname);
        driver.findElement(By.name("password")).sendKeys(pwd);
        driver.findElement(By.name("login")).click();
        Assert.assertEquals(driver.getTitle(), "Find a Flight: Mercury Tours:");
    }

    @DataProvider(name="users")
    String [][] logindata()
    {
        String data[][]={{"mercury","mercury"},{"asasa","mercury1"},{"mercury2","mercury2"} };
        return data;
    }

    @AfterClass
    void closebrowser()
    {
    driver.quit();
    }

}
```

Data Provider method

Agenda

- TestNG Listeners
- Extent Reports

TestNG Listeners

- **ITestListener** has following methods.
  - **OnStart-** OnStart method is called when any Test starts.
  - **onTestSuccess-** onTestSuccess method is called on the success of any Test.
  - **onTestFailure-** onTestFailure method is called on the failure of any Test.
  - **onTestSkipped-** onTestSkipped method is called on skipped of any Test.
  - **onFinish-** onFinish method is called after all Tests are executed.

- **TestListenerAdapter** Class implemented all above methods.

How to use TestNG Listeners (Without XML)

```java
package testnglisteners;

import org.testng.ITestContext;
import org.testng.ITestListener;
import org.testng.ITestResult;

public class Mylisteners implements ITestListener {
    @Override
    public void onTestStart(ITestResult result) {
        System.out.println(" test is started");
    }

    @Override
    public void onTestSuccess(ITestResult result) {
        System.out.println(" test is passed");
    }

    @Override
    public void onTestFailure(ITestResult result) {
        System.out.println("test is failed");
    }

    @Override
    public void onTestSkipped(ITestResult result) {
        System.out.println("test is skipped");
    }

    @Override
    public void onTestFailedButWithinSuccessPercentage(ITestResult result) {
        // TODO Auto-generated method stub

    }

    @Override
    public void onStart(ITestContext context) {
        // TODO Auto-generated method stub

    }

    @Override
    public void onFinish(ITestContext context) {
        // TODO Auto-generated method stub

    }
}
```

Listeners Class

```java
package testnglisteners;

import org.testng.Assert;
import org.testng.annotations.Listeners;
import org.testng.annotations.Test;

@Listeners(testnglisteners.Mylisteners.class)
public class LoginTest {

    @Test
    void setup()
    {
        Assert.fail();
    }

    @Test
    void loginByEmail()
    {
        Assert.assertTrue(true);
    }

    @Test(dependsOnMethods={"setup"})
    void loginByFacebook()
    {
        Assert.assertTrue(true);
    }
}
```

Test Case

How to use TestNG Listeners (Using XML)

```java
package testnglisteners;
import org.testng.ITestResult;
import org.testng.TestListenerAdapter;

public class Listeners extends TestListenerAdapter {
    public void onTestStart(ITestResult tr) {
        System.out.println(" test is started");
    }

    public void onTestSuccess(ITestResult tr) {
        System.out.println(" test is passed");
    }

    public void onTestFailure(ITestResult tr) {
        System.out.println("test is failed");
    }

    public void onTestSkipped(ITestResult tr) {
        System.out.println("test is skipped");
    }
}
```

Listeners Class

```xml
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE suite SYSTEM "http://testng.org/testng-1.0.dtd">

<suite name="TestNGListeners">

    <listeners>
        <listener class-name="testnglisteners.Listeners" />

    </listeners>

    <test name="Logintest">
        <classes>
            <class name="testnglisteners.LoginTest"></class>
        </classes>
    </test>

</suite>
```

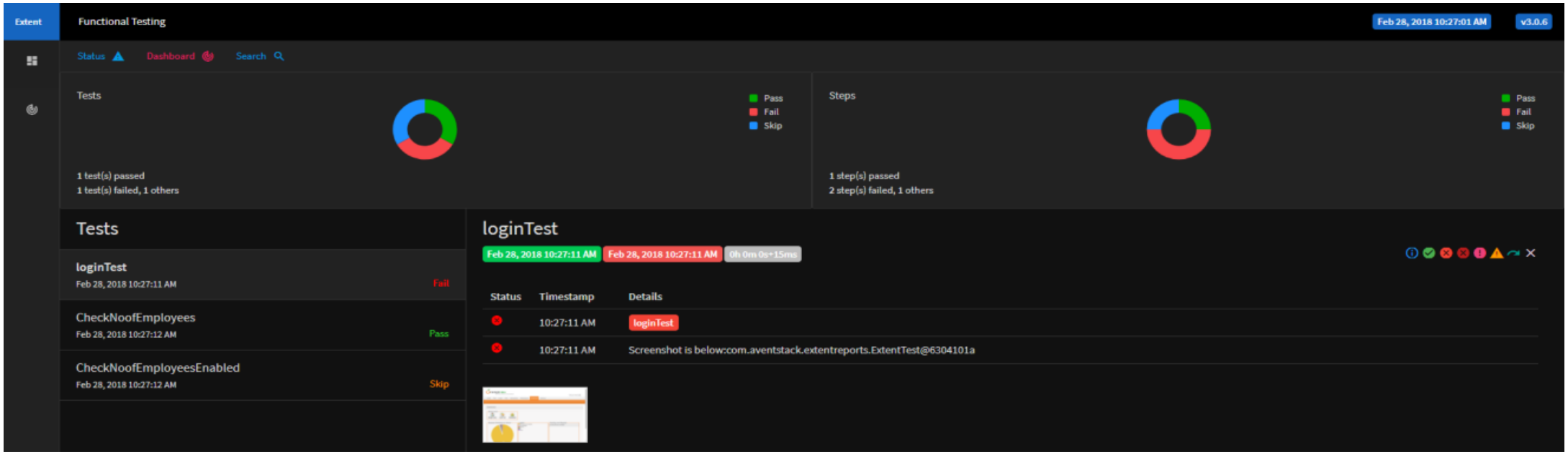
TestNG XML

```java
package testnglisteners;
import org.testng.Assert;
import org.testng.annotations.Test;

public class LoginTest {

    @Test
    void setup()
    {
        Assert.fail();
    }

    @Test
    void loginByEmail()
    {
        Assert.assertTrue(true);
    }

    @Test(dependsOnMethods={"setup"})
    void loginByFacebook()
    {
        Assert.assertTrue(true);
    }
}
```

Test Case

Extent Reports

- Extent Report is third party the most popular and widely used Selenium Reporting tool in the current market.



Generating Extent Reports

- Download extent reports configuration files (API)

http://extentreports.com/community/#

  - ✓ .jar files

- Add all the jars to the project build path

Classes used to generate Extent Reports

- **Template definition:**
  - ExtentHtmlReporter
- **Test Methods status**
  - ExtentReports
- **Detailed info in the report**
  - ExtentTest

Capture Screenshot

```
import java.io.File;
import java.io.IOException;
import org.apache.commons.io.FileUtils;
import org.openqa.selenium.OutputType;
import org.openqa.selenium.TakesScreenshot;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.chrome.ChromeDriver;
import org.testng.annotations.Test;

public class CaptureScreenshot {

    @Test
    void capturescreen() throws IOException
    {
        System.setProperty("webdriver.chrome.driver", "C://Drivers//chromedriver_win32//chromedriver.exe");
        WebDriver driver = new ChromeDriver();

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

        driver.get("http://www.google.com");

        // Take screenshot and store as a file format

        TakesScreenshot ts=(TakesScreenshot) driver;

        File src = ts.getScreenshotAs(OutputType.FILE);

        // now copy the screenshot to desired location using copyFile method

        File trg=new File("C://screenshot/google.png");

        FileUtils.copyFile(src,trg);

    }
}
```

Agenda

- Log4J
- Page Object Model

What is log4J

- Log4j is free open source tool given by Apache foundation for creating log files It helps us to generate a log file in various output target.
- Log Levels – DEBUG ,INFO,WARN,ERROR,FATAL
- **Generating automation logs using Log4j**
  - 1) Log4j.jar
  - 2) log4j.properties

Step1	• Copy log4j.properties file at project home directory
Step2	• Add log4j.jar file to your project build path.
Step3	• Include log code in your test case.

Test Case with log4j

Log4j.properties

```
// Here we have defined root logger
log4j.rootLogger=INFO,CONSOLE,R,HTML,TTCC

// Here we define the appender
log4j.appender.CONSOLE=org.apache.log4j.ConsoleAppender
log4j.appender.R=org.apache.log4j.RollingFileAppender
log4j.appender.TTCC=org.apache.log4j.RollingFileAppender
log4j.appender.HTML=org.apache.log4j.FileAppender

// Here we define log file location
log4j.appender.R.File=./log/testlog.log
log4j.appender.TTCC.File=./log/testlog1.log
log4j.appender.HTML.File=./log/application.html

// Here we define the layout and pattern
log4j.appender.CONSOLE.layout=org.apache.log4j.PatternLayout
log4j.appender.CONSOLE.layout.ConversionPattern= %5p [%t] (%F:%L)- %m%n
log4j.appender.R.layout=org.apache.log4j.PatternLayout
log4j.appender.R.layout.ConversionPattern=%d - %c -%p - %m%n
log4j.appender.TTCC.layout=org.apache.log4j.TTCCLayout
log4j.appender.TTCC.layout.DateFormat=ISO8601
log4j.appender.HTML.layout=org.apache.log4j.HTMLLayout
log4j.appender.HTML.layout.Title=Application log
log4j.appender.HTML.layout.LocationInfo=true
```

Test Case

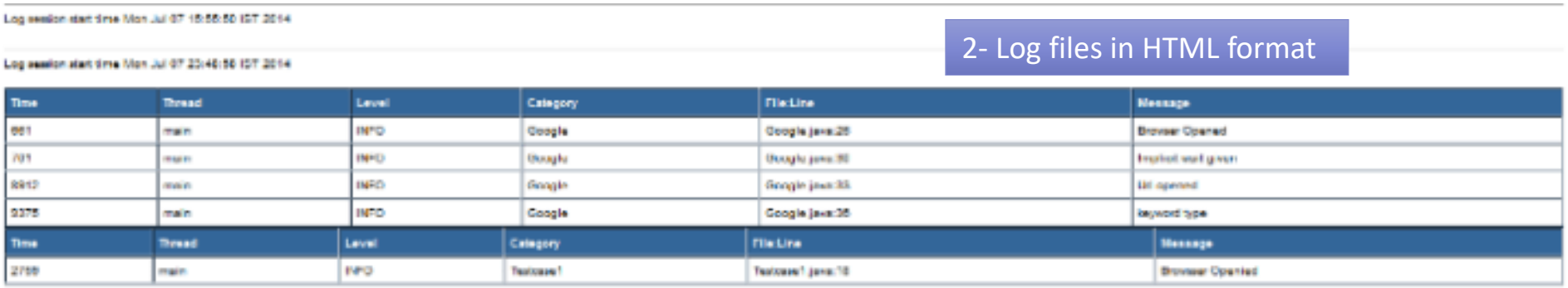
```
import org.apache.log4j.Logger;
import org.apache.log4j.PropertyConfigurator;
import org.testng.Assert;
import org.testng.annotations.Test;


public class Log4JExample {

    public static Logger logger;


    @Test(priority=1)
    public void setup()
    {
        logger=Logger.getLogger("Log4JExample"); // need to pass class name as parameter
        PropertyConfigurator.configure("Log4j.properties");

        logger.info("setup method is starting");
        System.out.println(" this is for setup");
        Assert.assertTrue(true);
        logger.info("setup method passed");
    }

    @Test(priority=2)
    public void login()
    {
        logger.info(" login is started");
        System.out.println("login test");
        Assert.assertTrue(true);
        logger.info(" login is completed");
        logger.info("login is passed");
    }

    @Test(priority=3)
    public void logout()
    {
        logger.info("logout is started");
        System.out.println(" logout test");
        Assert.fail();
        logger.info("logout is failed");
    }

}
```

Log4j logs

File Edit Format View Help

```
2014-07-07 15:55:51,001 - Google -INFO - Browser Opened
2014-07-07 15:55:51,041 - Google -INFO - Implicit wait given
2014-07-07 15:55:59,252 - Google -INFO - Url opened
2014-07-07 15:55:59,715 - Google -INFO - keyword type
2014-07-07 23:48:58,748 - Testcase1 -INFO - Browsser Openied
```

1- Simple log file in text format



Time	Thread	Level	Category	File:Line	Message
001	main	INFO	Google	Google.java:25	Browser Opened
701	main	INFO	Google	Google.java:30	Implicit wait given
9912	main	INFO	Google	Google.java:35	Url opened
9375	main	INFO	Google	Google.java:36	keyword type

Time	Thread	Level	Category	File:Line	Message
2759	main	INFO	Testcase1	Testcase1.java:10	Browsser Openied

2- Log files in HTML format

Page Object Model

- **Page Object Model** is a design pattern to create **Object Repository** for web UI elements.
- Under this model, for each web page in the application, there should be corresponding page class.
- This Page class will find the WebElements of that web page and also contains Page methods which perform operations on those WebElements.
- Name of these methods should be given as per the task they are performing.

3 Approaches

@CacheLookup : Perfomance improvement

```
Approach-1
-------------------------
@FindBy(id="Email")
WebElement txtemail;

public void setemail(String email)
{
txtemail.sendKeys(email);
}

Approach-2
-----------------
@FindBy(how=How.ID, using="Email")
WebElement txtemail;

public void setemail(String email)
{
txtemail.sendKeys(email);
}

Approach-3
--------------------
By Email=By.id("Email")

public void setemail(String email)
{
driver.findElement(Email).sendKeys(email);
}
```

Test Case with Page Object Model

```java
package pageObjectModel;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.WebElement;
import org.openqa.selenium.support.FindBy;
import org.openqa.selenium.support.How;
import org.openqa.selenium.support.PageFactory;

public class LoginPage {

    WebDriver ldriver;

    /*
     * @FindBy(how=How.NAME, using="userName") WebElement txtUsername;
     */

    @FindBy(name = "userName")
    WebElement txtUsername;

    @FindBy(name = "password")
    WebElement txtpassword;

    @FindBy(name = "login")
    WebElement btnsubmit;

    LoginPage(WebDriver rdriver) {
        ldriver = rdriver;
        PageFactory.initElements(rdriver, this);
    }

    public void setUserName(String uname) {
        txtUsername.sendKeys(uname);
    }

    public void setPassword(String pwd) {
        txtpassword.sendKeys(pwd);
    }

    public void clickSubmit() {
        btnsubmit.click();
    }
}
```

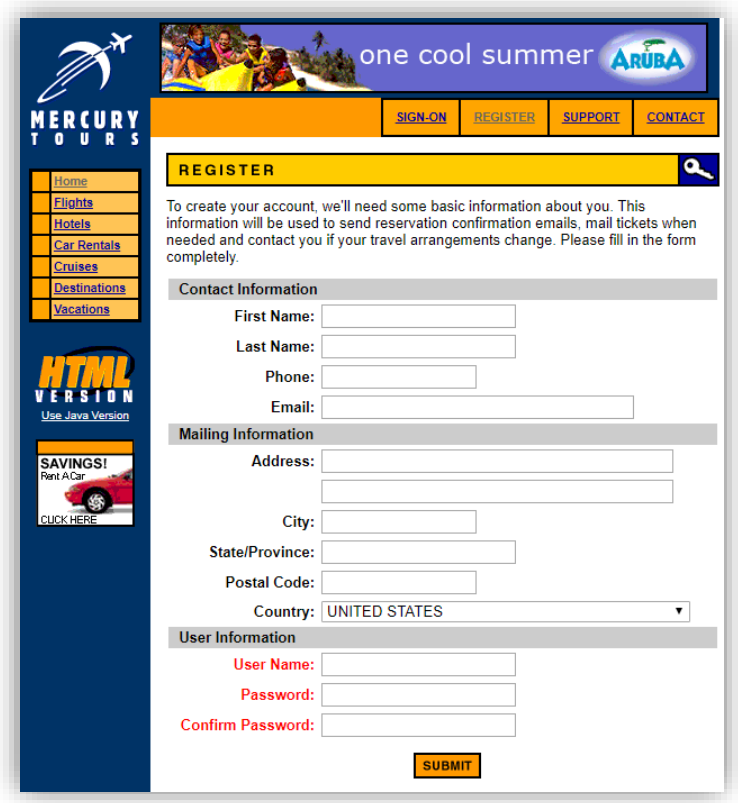
Page Object Class

```java
package pageObjectModel;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.chrome.ChromeDriver;
import org.testng.annotations.BeforeClass;
import org.testng.annotations.Test;

public class LoginTest {

    public static WebDriver driver;

    @BeforeClass
    void setup() {
        System.setProperty("webdriver.chrome.driver", "C://Drivers//chromedriver_win32/chromedriver.exe");
        driver = new ChromeDriver();
        driver.get("http://newtours.demoaut.com/");
    }

    @Test
    void login() {
        LoginPage lp = new LoginPage(driver);
        lp.setUserName("mercury");
        lp.setPassword("mercury");
        lp.clickSubmit();

    }

}
```

Test Case

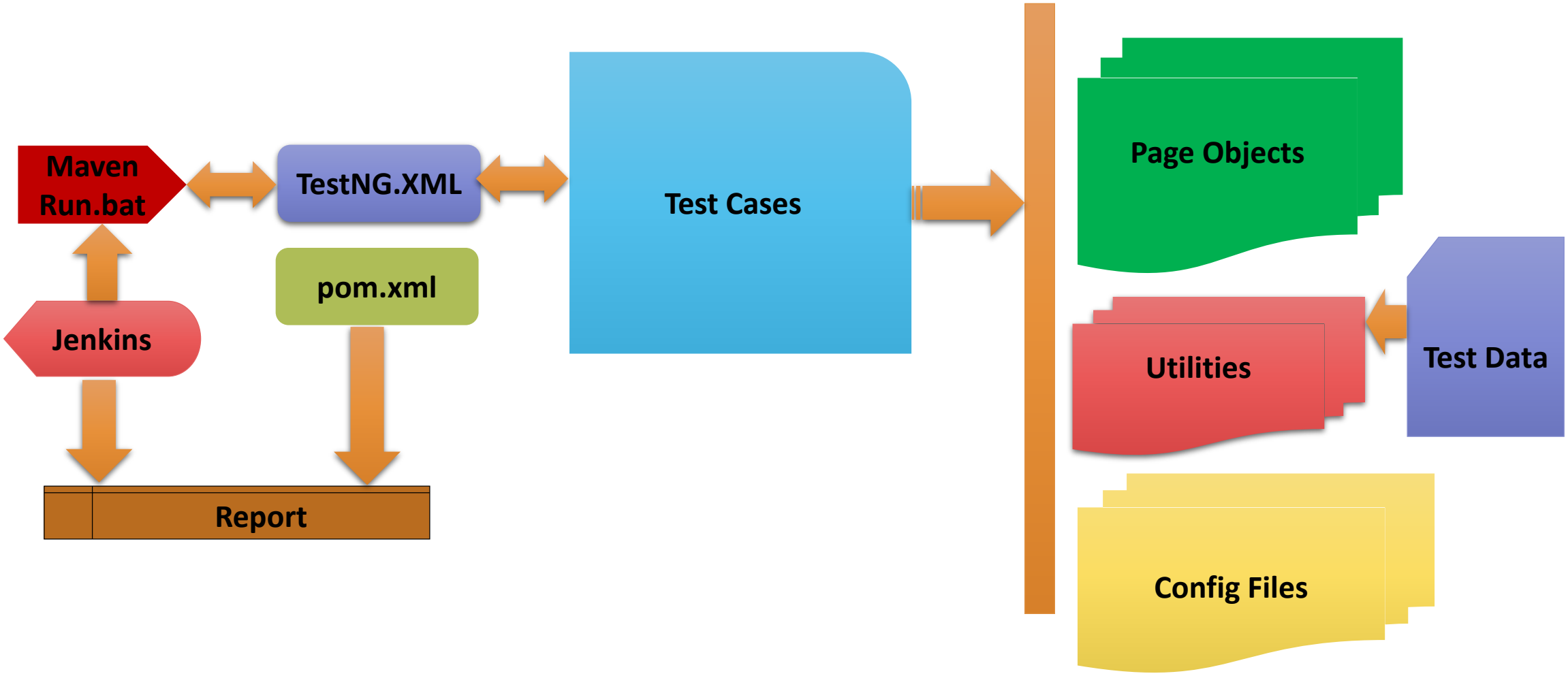
Hand's on

- Develop automation script for Flight Registration using Page Object Model.
- URL: http://newtours.demoaut.com/mercuryregister.php



Hybrid Driven Automation Framework

Automation Framework(Hybrid)



Phase-1: Implementation

1. Create Maven Project
2. Update pom.xml
3. Create Page Objects
4. Create Basic Test case
5. Add logs to test case
6. Read common values from properties file
7. Run test case on desired browser
8. Add extent report
9. Create Data Driven test case
10. Adding new test cases

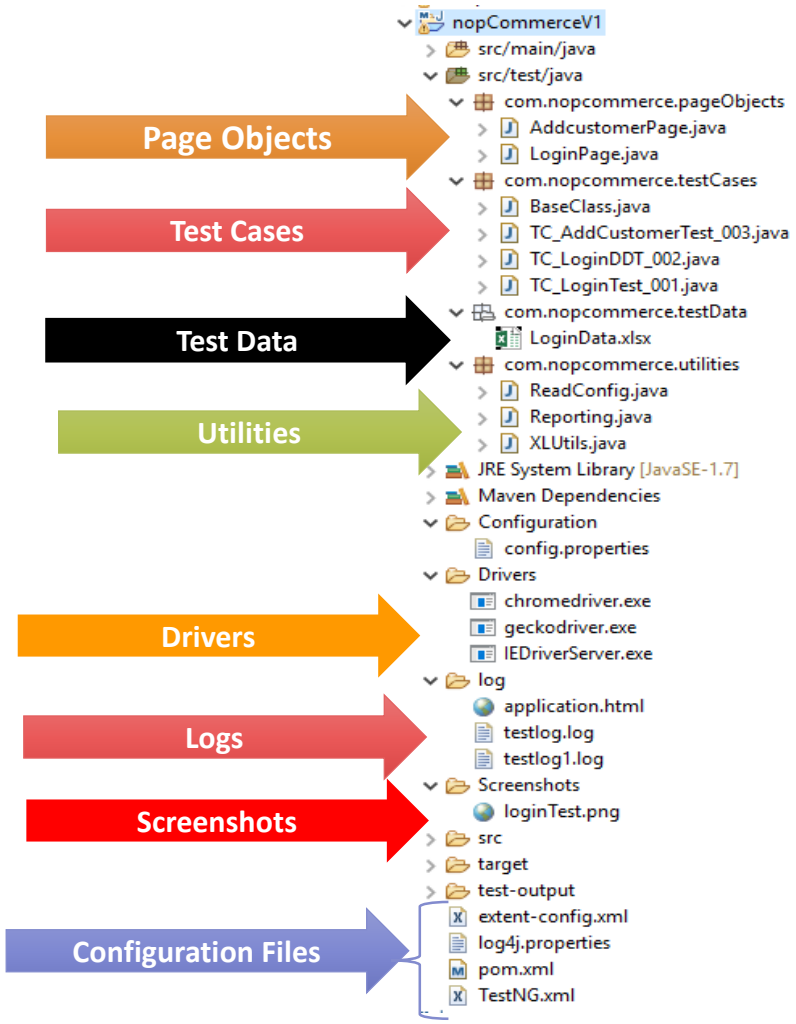
Phase-2: Execution

1. Run test cases with Maven pom.xml
2. Run test cases through Maven CLI (Command Line Interface)
3. Run test cases using run. bat
4. Run test cases using Jenkins (using bat file)

Phase-3: Maintenance

- Creating repository in GITHUB
- Commit the project code in local repository
- Push the project code to GITHUB remote repository from local GIT repository

Framework Structure



Continuous Integration (CI)

Maven

Jenkins

Git

GitHub

CI Tools

- Maven
- Jenkins
- Git
- GitHub

Maven

Run test cases using Maven pom.xml

- You need to add 2 plug-ins to pom.xml File
  - maven-compiler-plugin
  - maven-surefire-plugin
- Right click on pom.xml → Run as Maven test
- **Issue:**
- Error: [ERROR] No compiler is provided in this environment. Perhaps you are running on a JRE rather than a JDK?
- Solution: Eclipse Window → Preferences →Java → Installed JRE's → and check your installed JREs.
- You should have an entry with a JDK there.
- Select the Execution Environment as "*C:\Program Files\Java\jdk1.8.0_151*" → Click OK

Run test cases through Maven CLI

- **Install Maven on Windows OS**
  - Download Maven for Windows
  - http://redrockdigimark.com/apachemirror/maven/maven-3/3.5.2/binaries/apache-maven-3.5.2-bin.zip
- **Add maven path to Windows environment variable.**
  - Right click on MyPC→ Advanced System settings→ Environment Variables
- **Check version of Maven installed on Windows.**
  - c: mvn -version
- **Issue:**
- Error: [ERROR] No compiler is provided in this environment. Perhaps you are running on a JRE rather than a JDK?
- Solution: REFERE LINK: http://roufid.com/no-compiler-is-provided-in-this-environment/
- Run Test cases through command prompt.
  - cd C:\Users\admin\eclipse-workspace\InetBanking
  - mvn clean install

Run automation through run.bat

- Create **run.bat** file which contains below entries

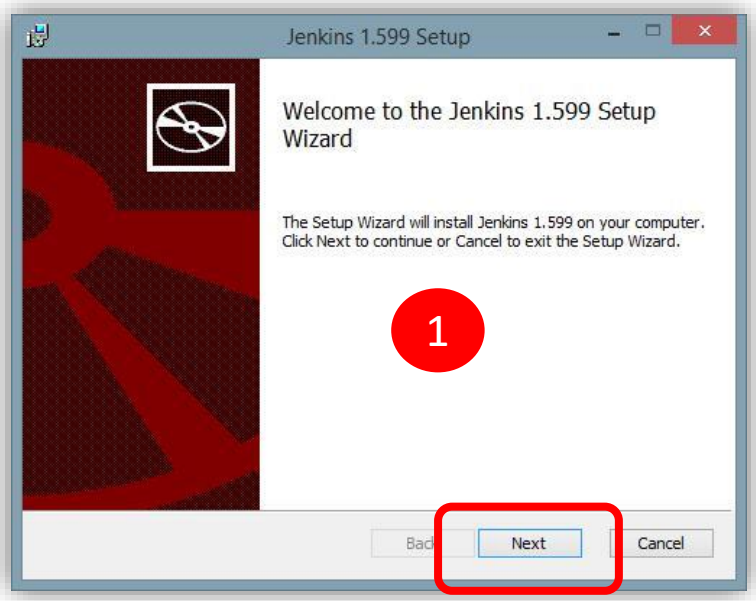
  cd C:\Users\admin\eclipse-workspace\InetBanking

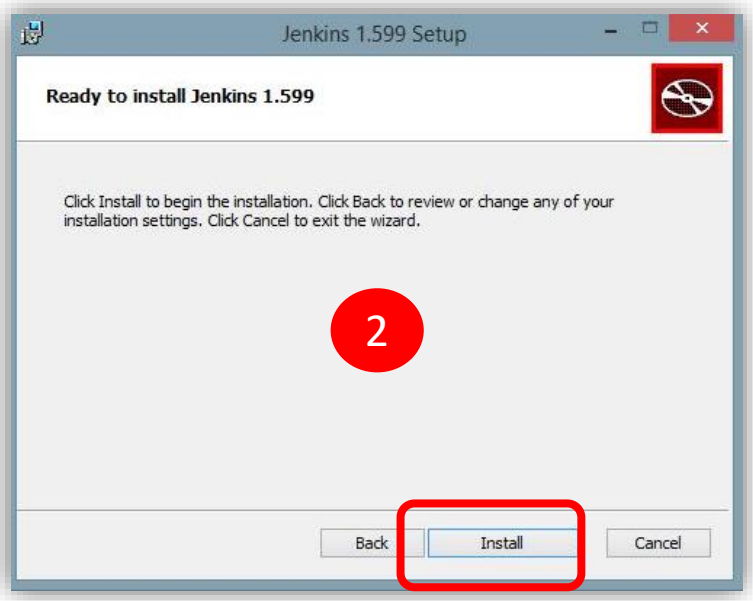
  mvn clean install

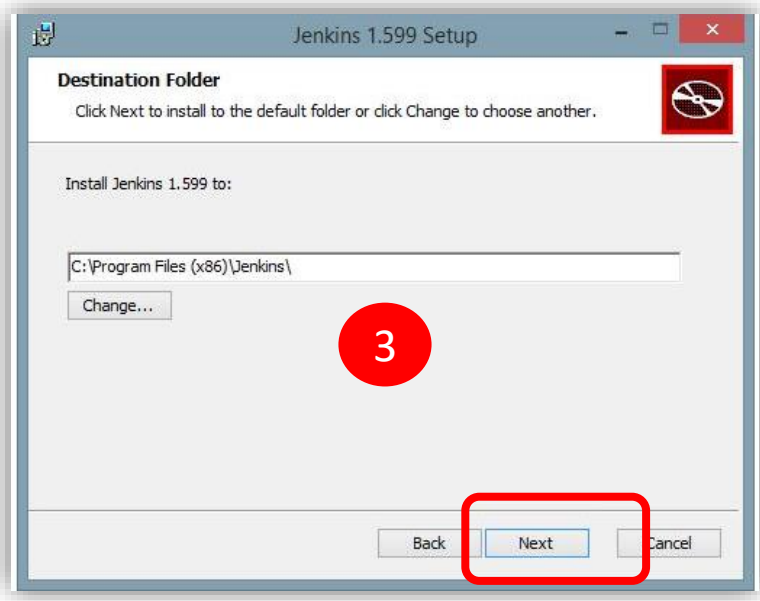
Jenkins installation & Configuration

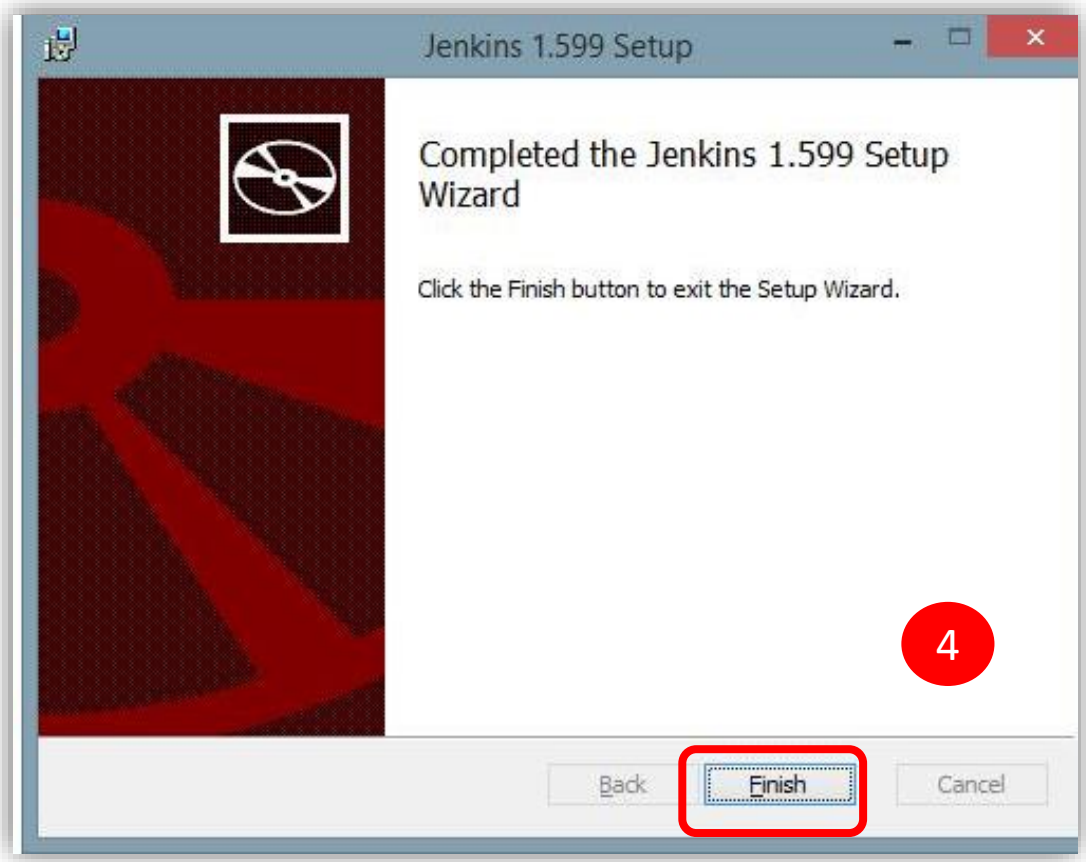
Jenkins Step By step Installation

- Download Link: https://jenkins.io/download/



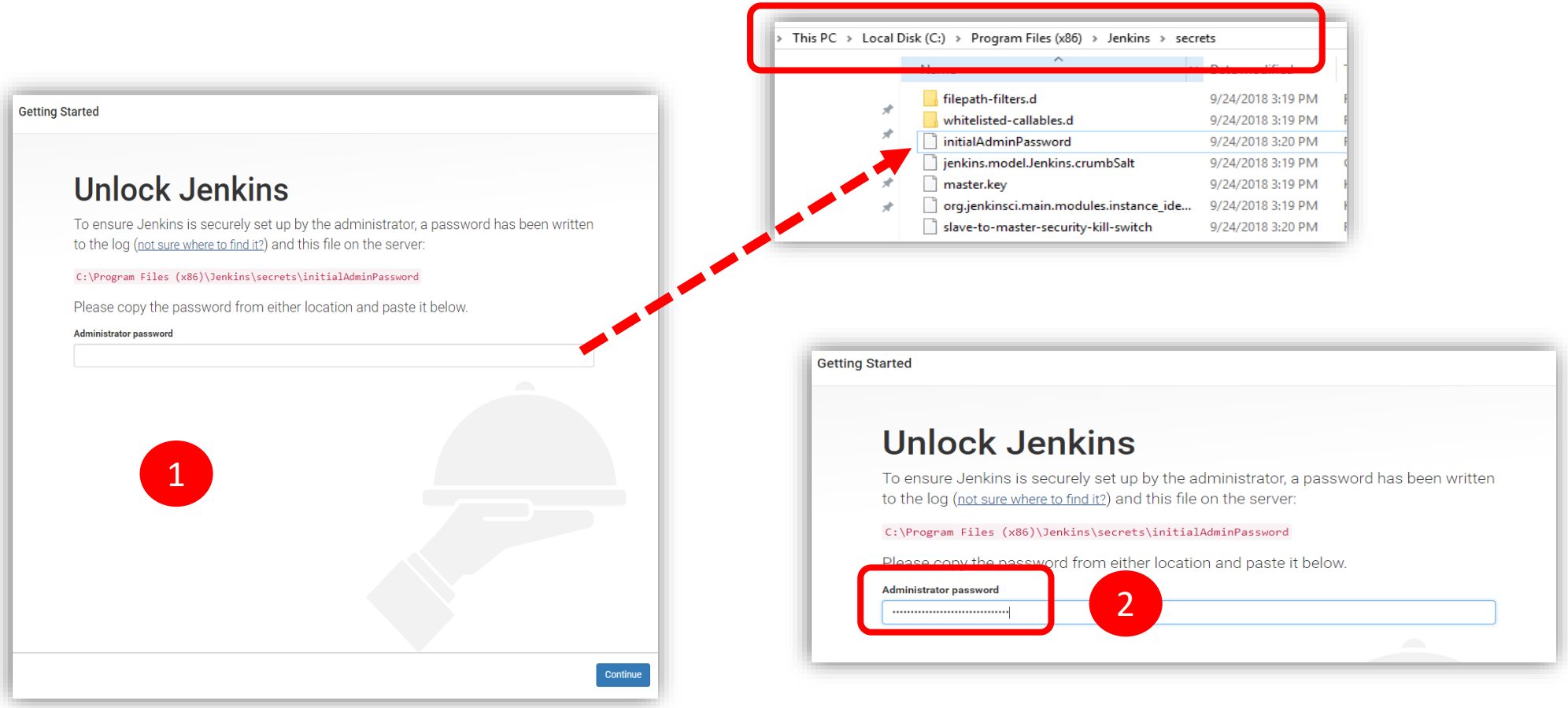




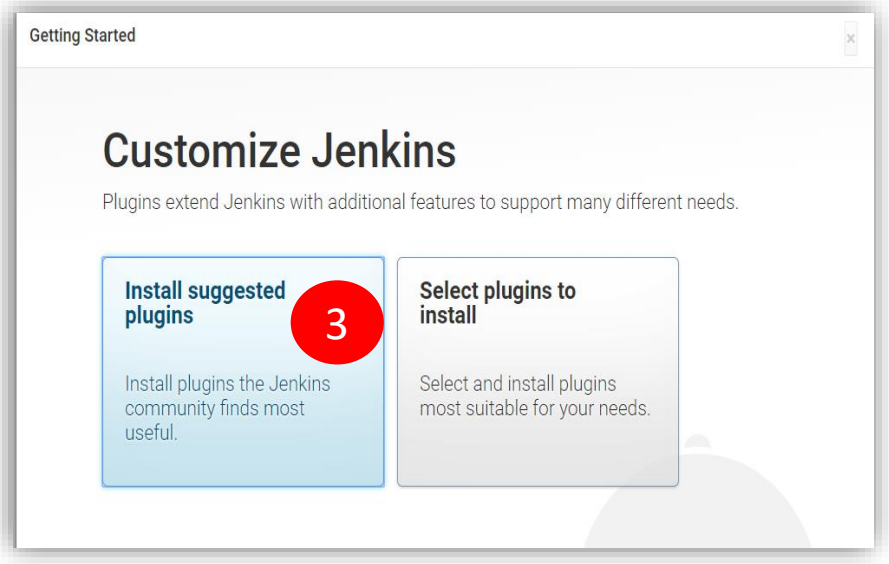

Jenkins 1.599 Setup
Completed the Jenkins 1.599 Setup Wizard
Click the Finish button to exit the Setup Wizard.
4
Back
Finish
Cancel

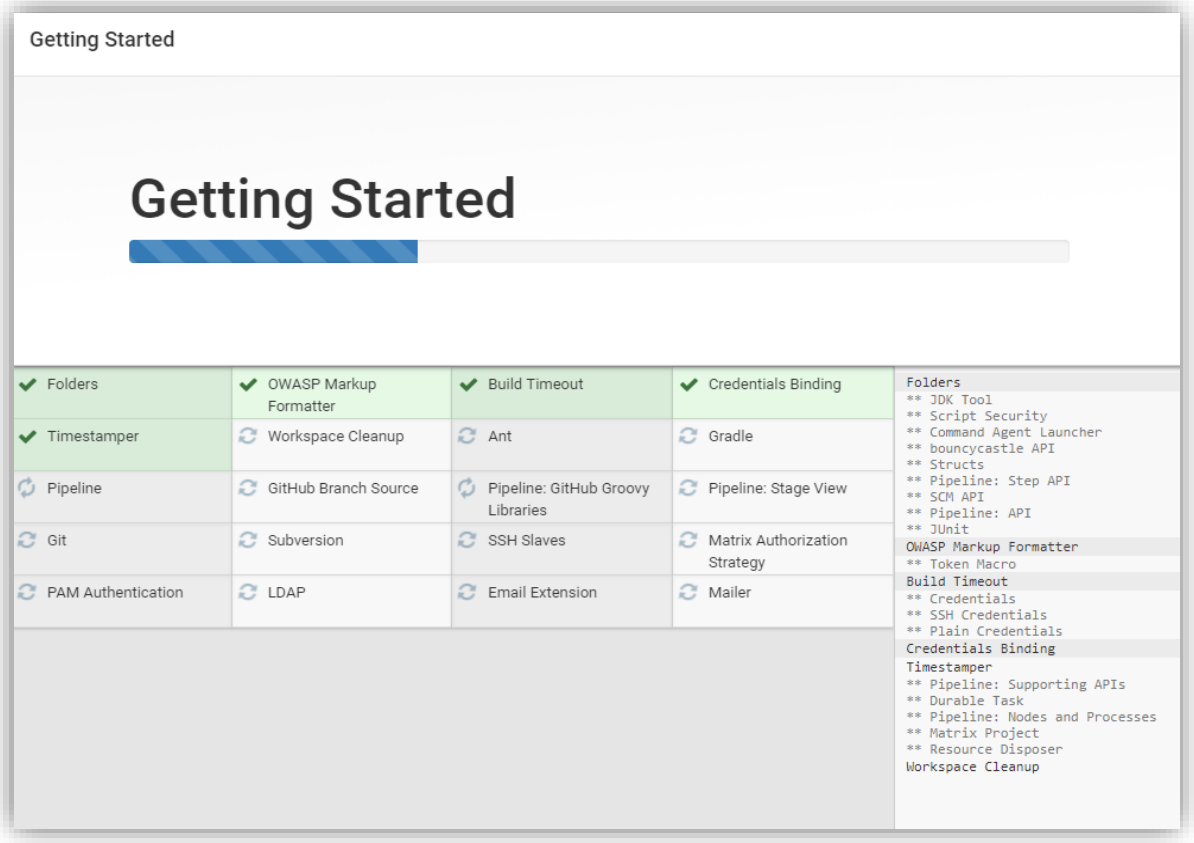
Jenkins Configuration

- URL to open Jenkins: http://localhost:8080

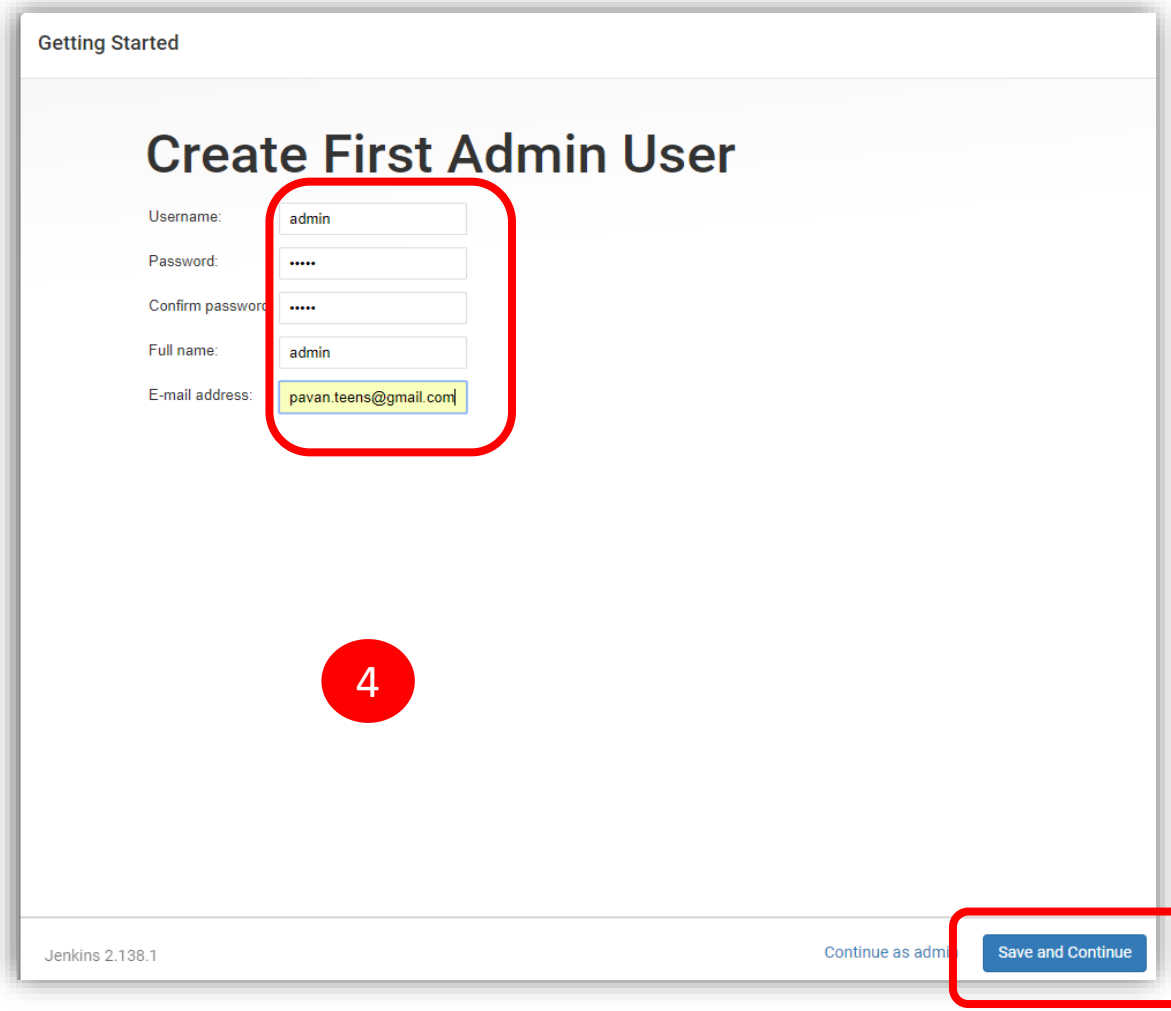


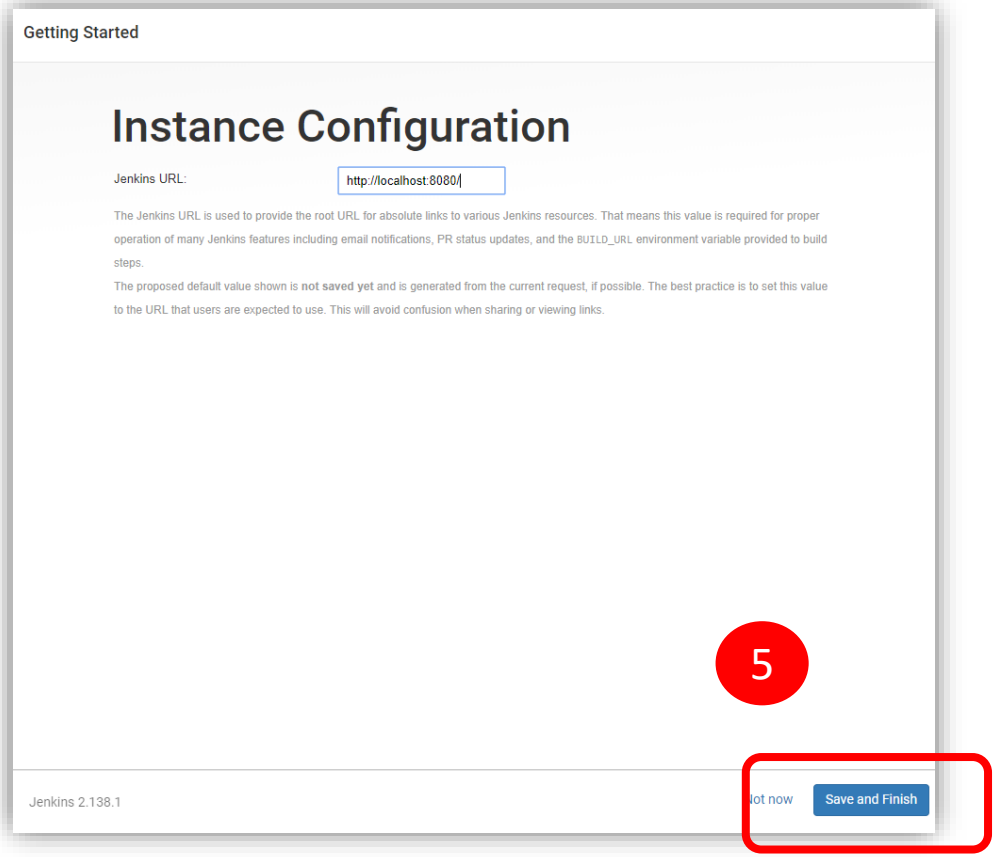
Jenkins Configuration



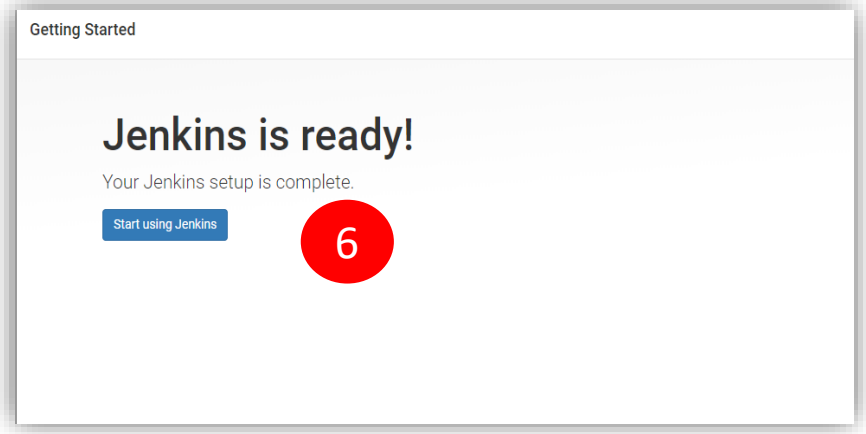


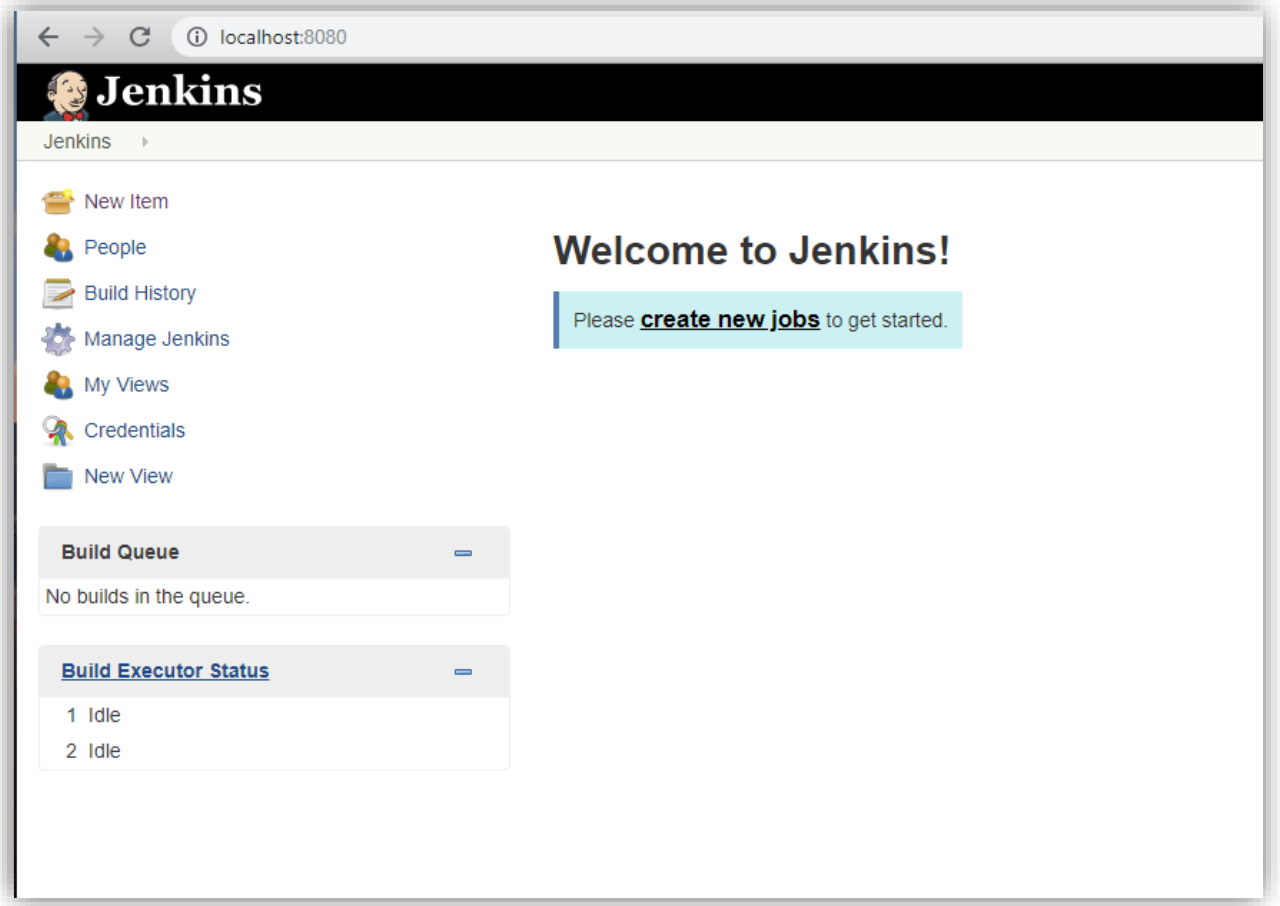
Jenkins Configuration



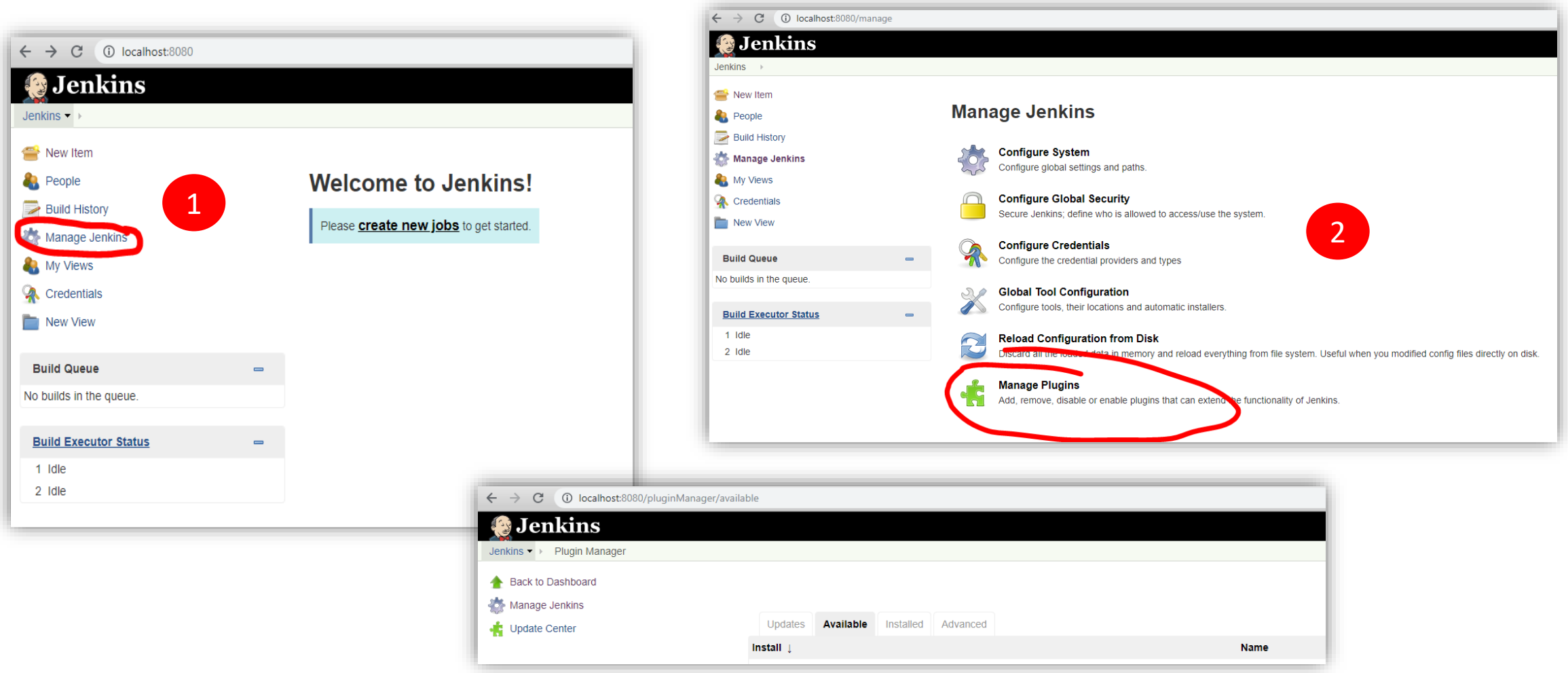


Jenkins Configuration

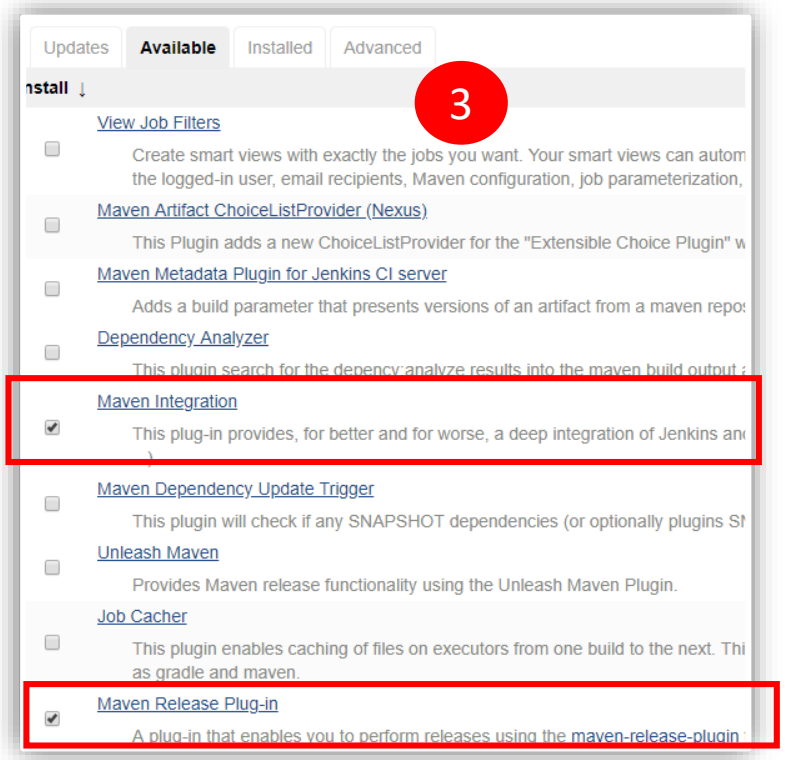


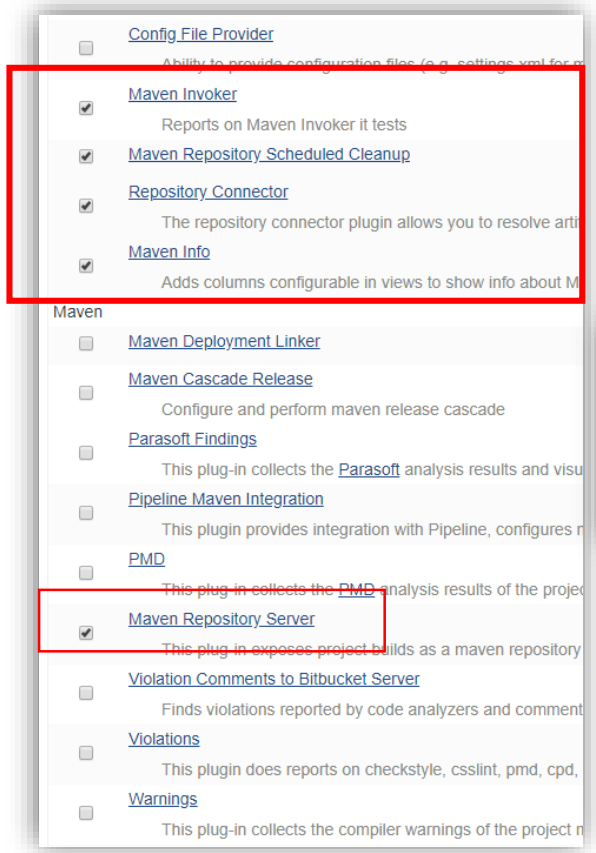


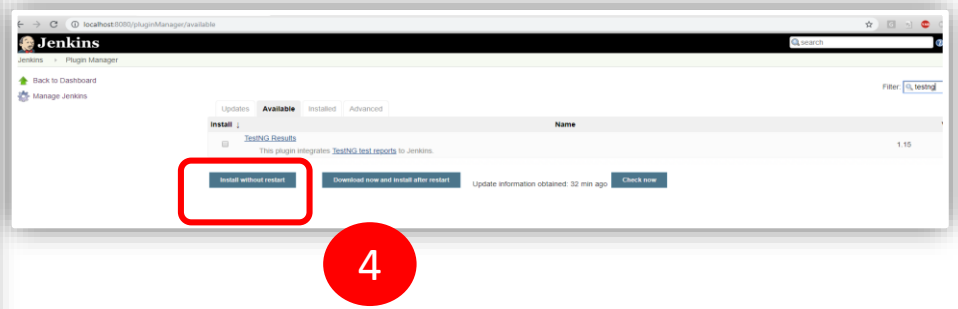
Install Maven Plugins in Jenkins



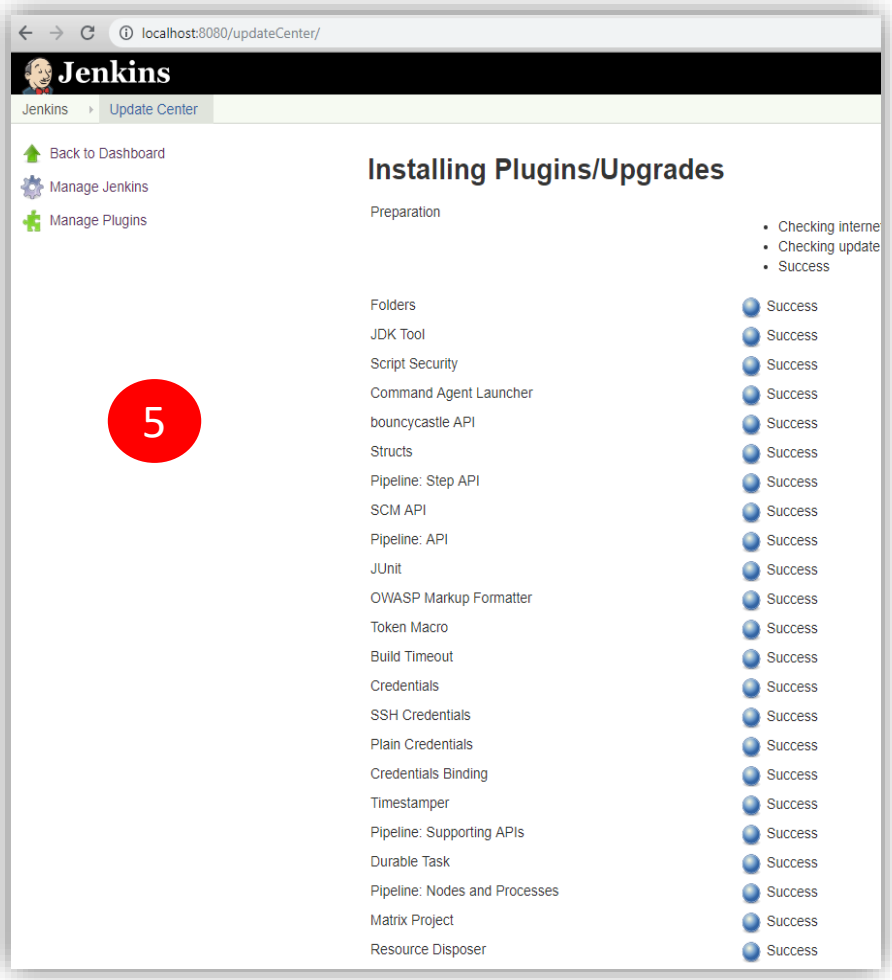
Maven Plugins for Jenkins

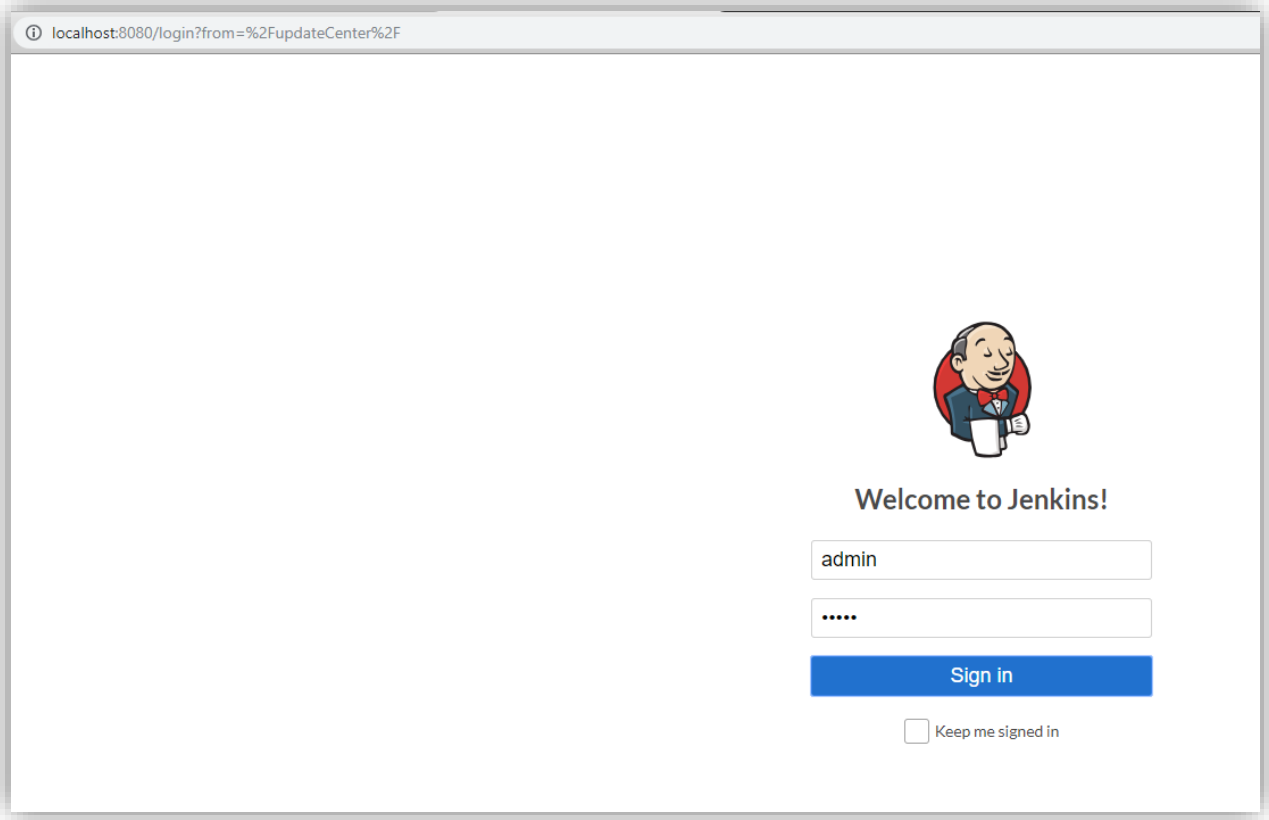






Maven Plugins for Jenkins



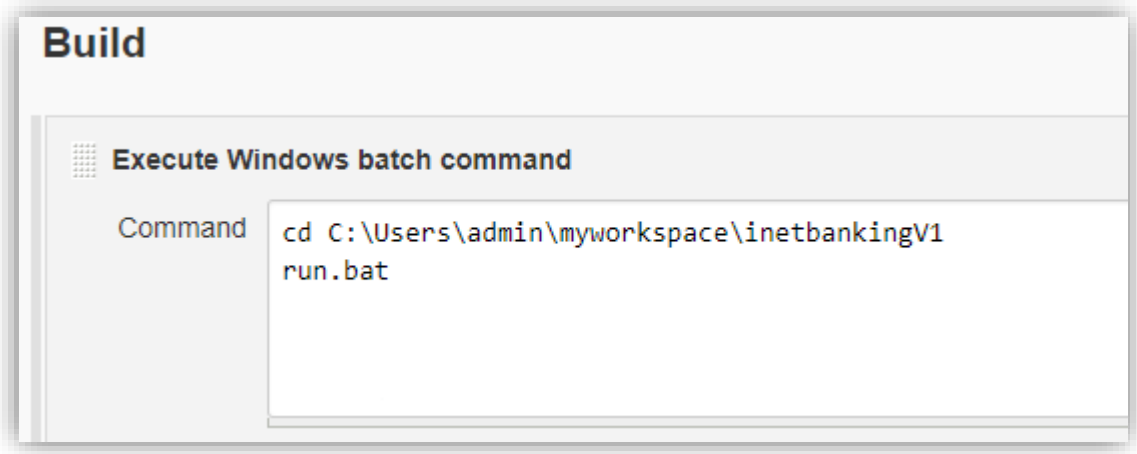


Run Automation through Jenkins

Run Automation using Jenkins Free Style Project

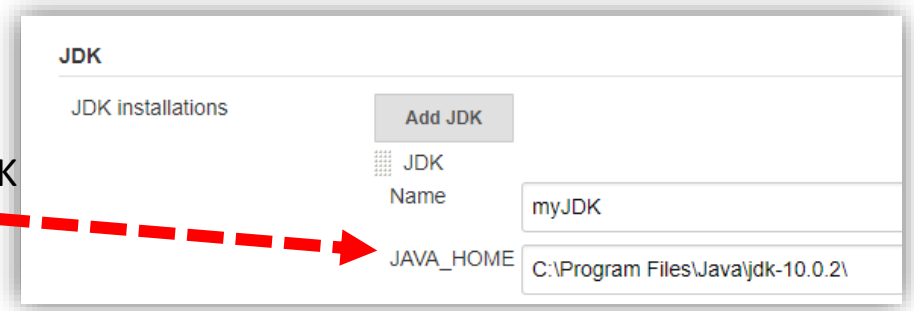
- Jenkins URL: http://localhost:8080/

- **Steps:**
  - New item → provide name of the project → Free style project →OK.
  - Go to build section →Execute windows batch command → Specify path of run.bat file →Save.
  - Go to Dash board → you can see new item(project) is created.

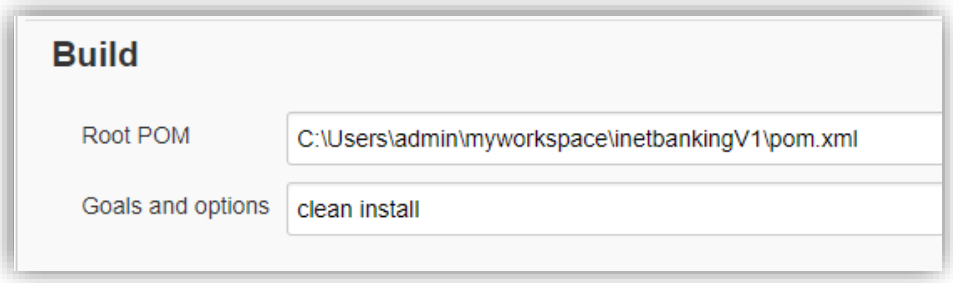


Run Automation using Jenkins Maven Project

- Jenkins URL: http://localhost:8080/
- **Pre-requisite:** JAVA_HOME Configuration in Jenkins
  - Dashboard→ Manage Jenkins→ Global Tool Configuration→ JDK

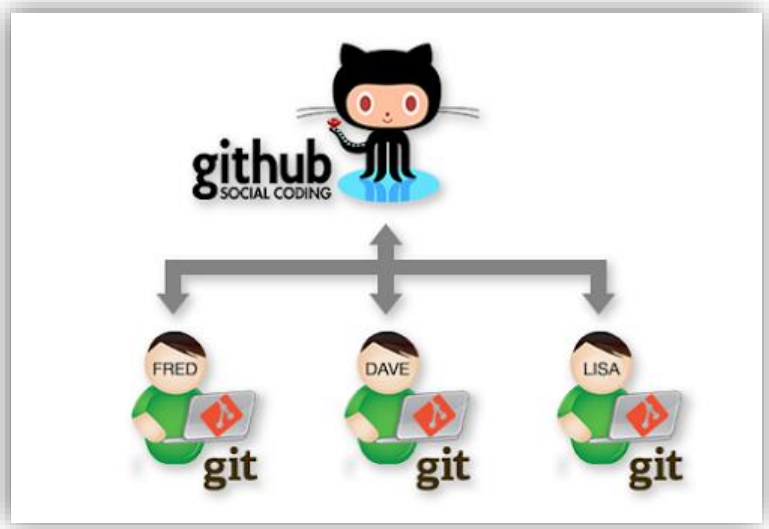


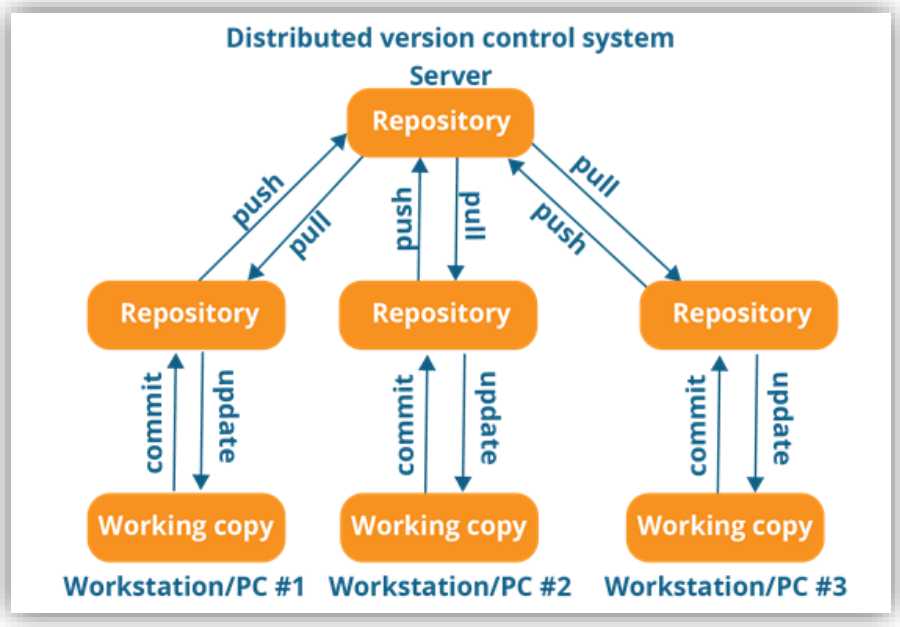
- **Steps:**
- New item → provide name of the project → Maven project →OK.
- Go to build section → Specify path of pom.xml and Goals as shown in picture
- Go to Dash board → you can see new item(project) is created.



Git & GitHub

- **Git** is a revision control system used to track changes in computer files. It's a tool to manage your code & file history while coordinating work remotely on those files with others. **GitHub** is a hosting service for **git** repositories. **Git** is the tool, while **GitHub** is the service to use **git**.

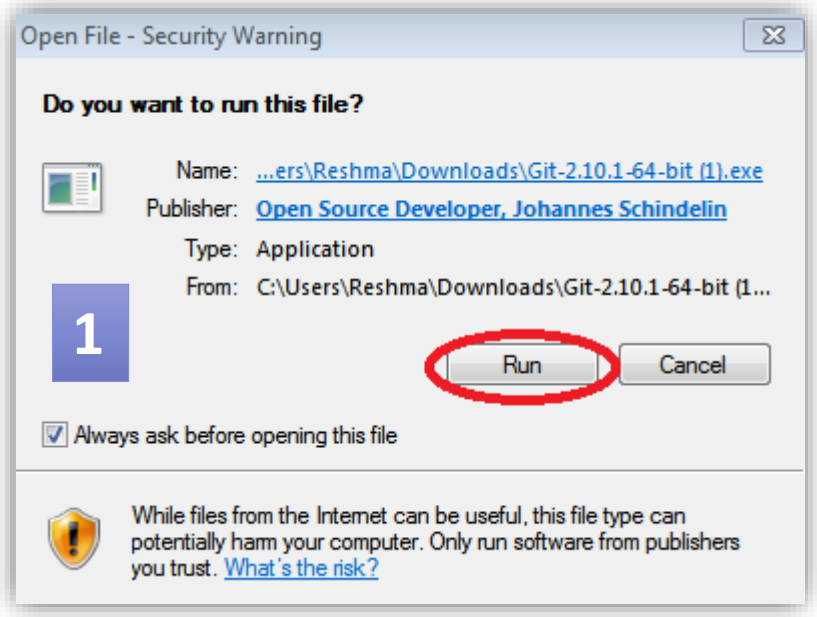


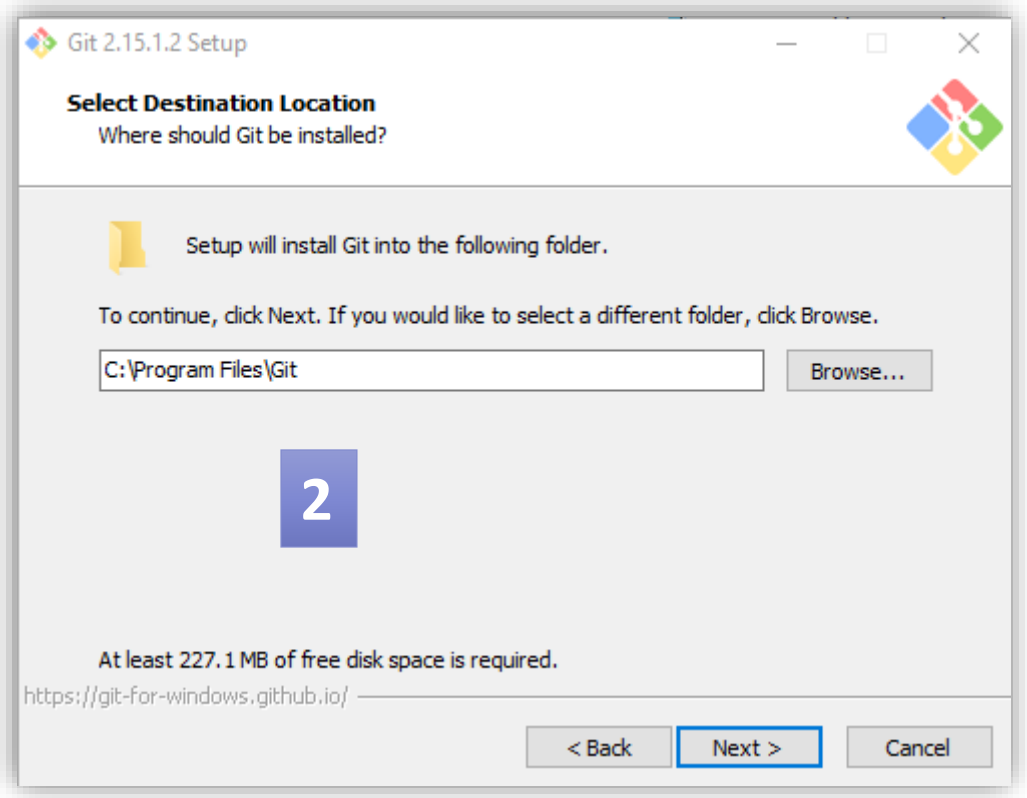


GIT Installation on Windows

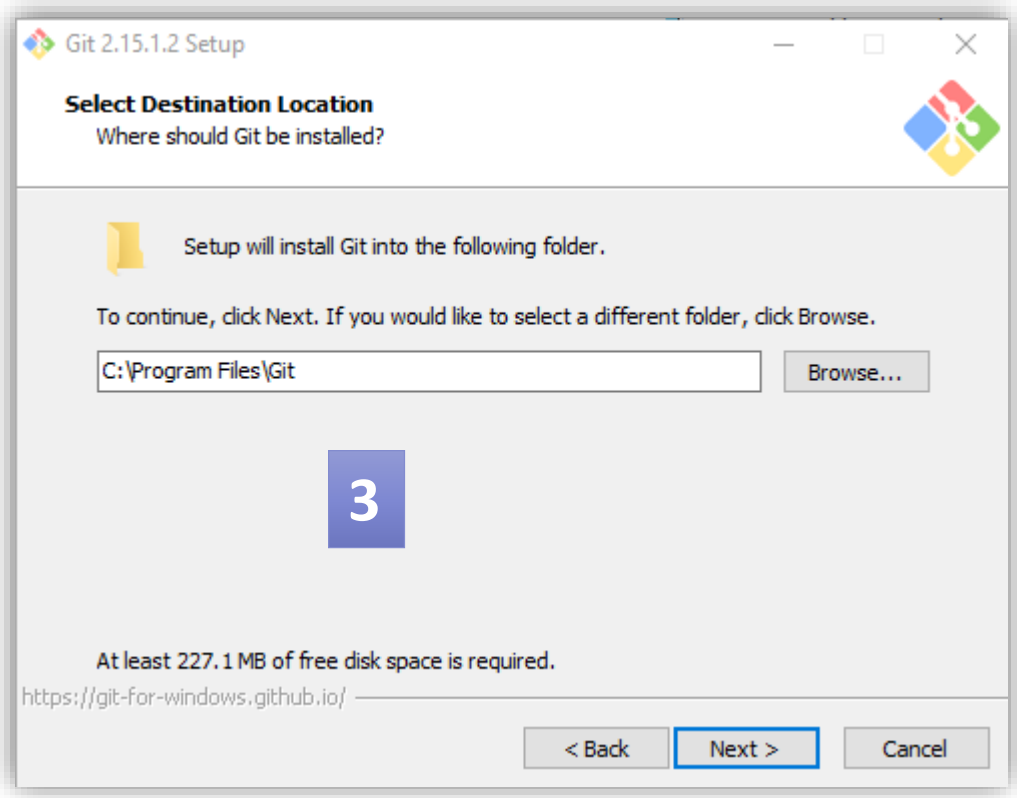
- To download the latest version of Git, click on the link below:
- https://git-scm.com/download/win/

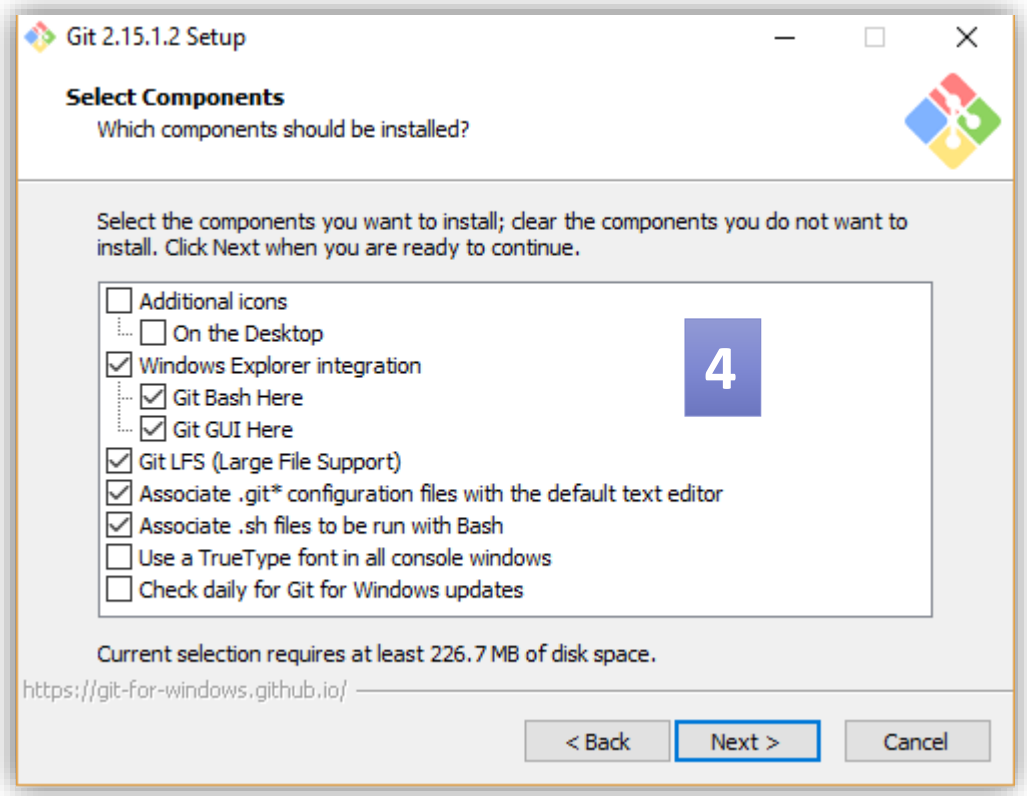
GIT Installation on Windows



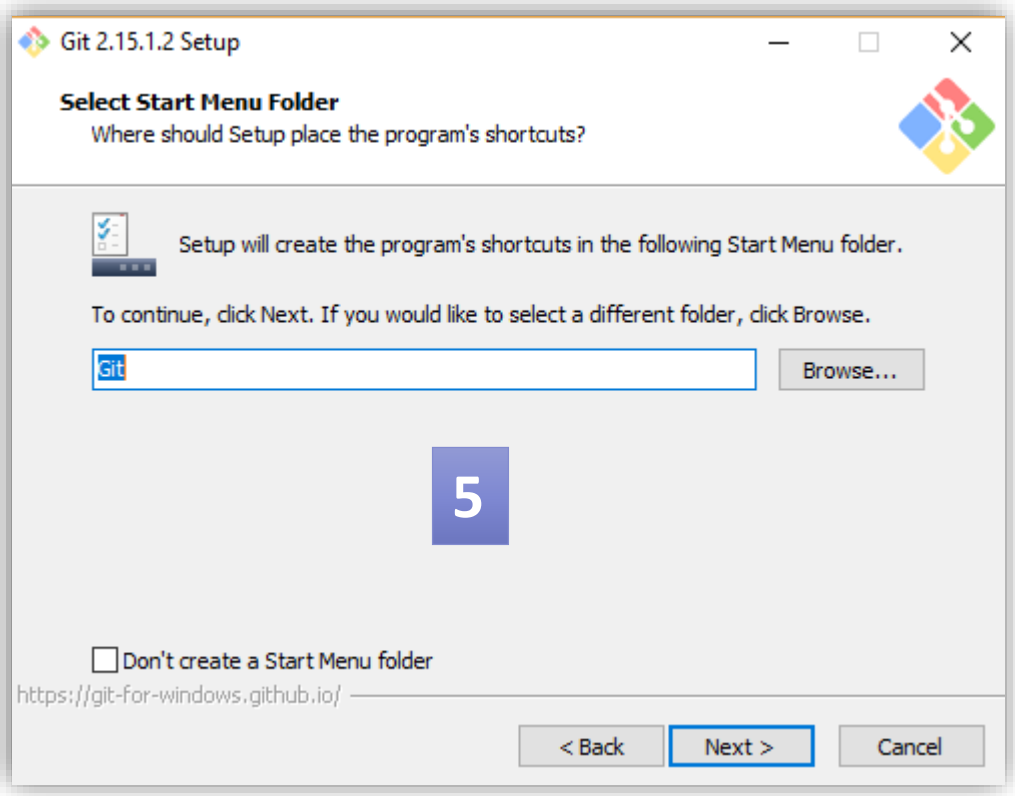


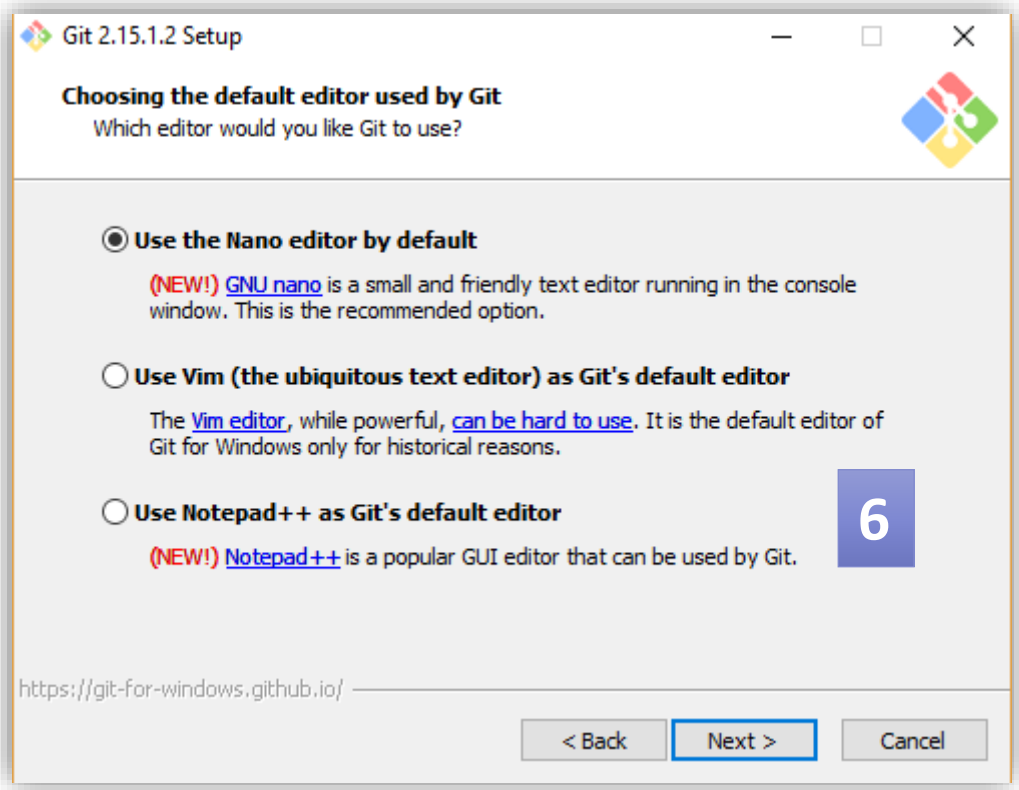
GIT Installation on Windows



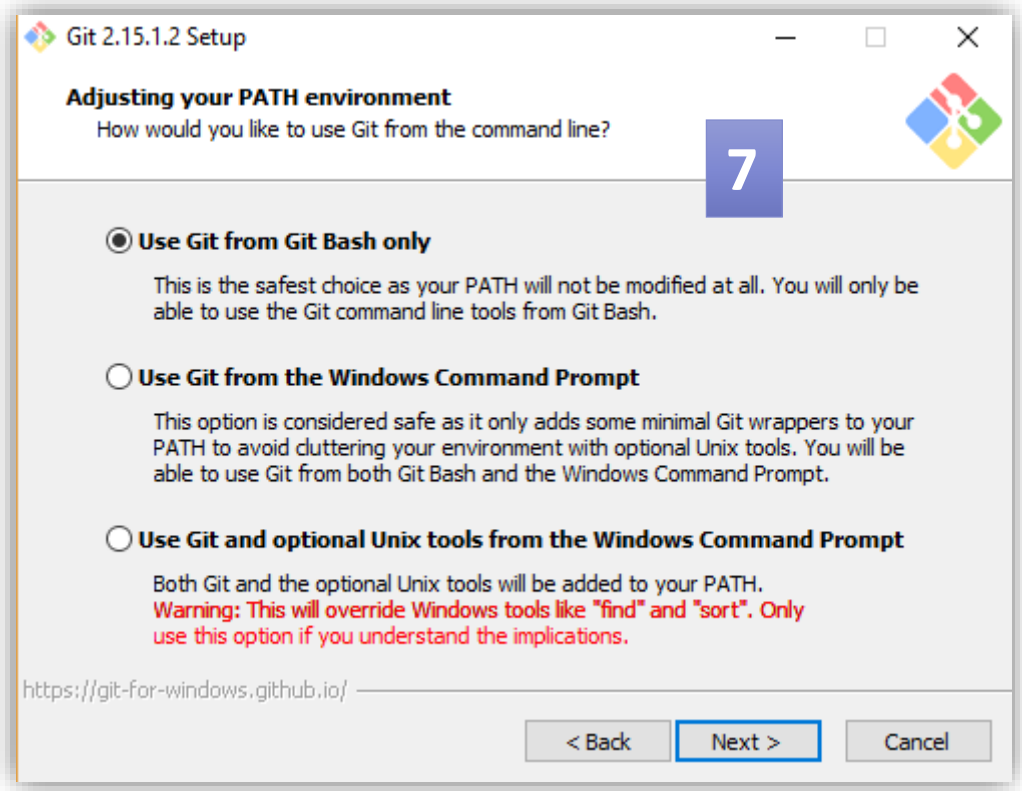


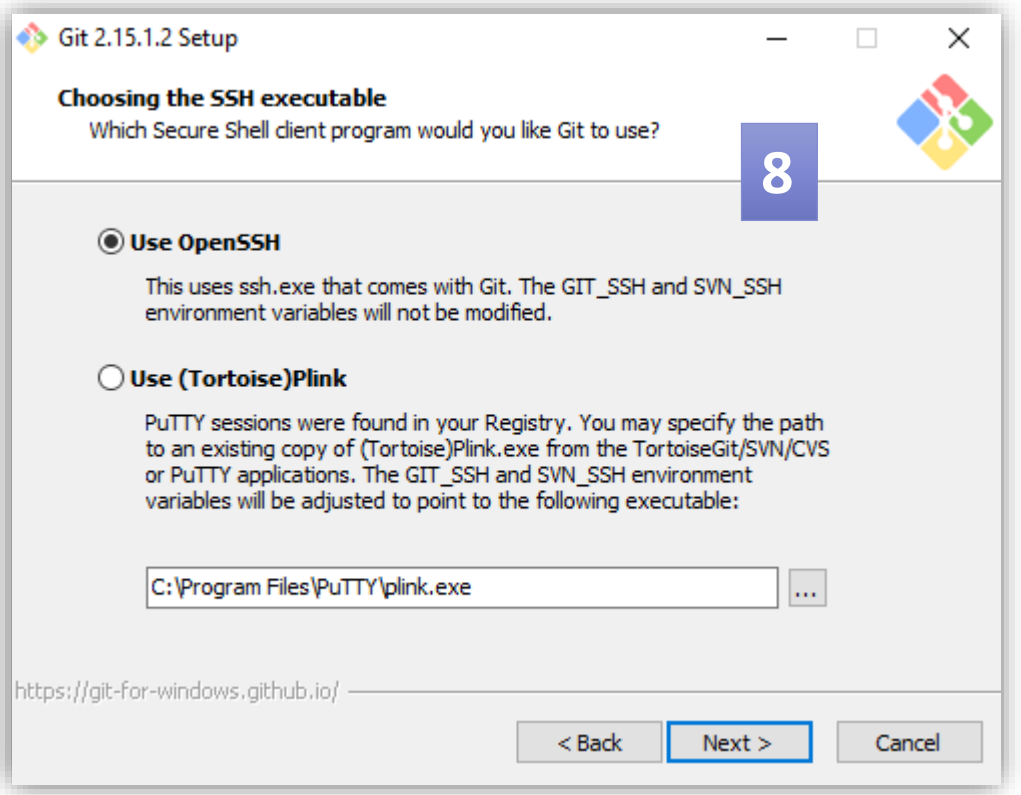
GIT Installation on Windows



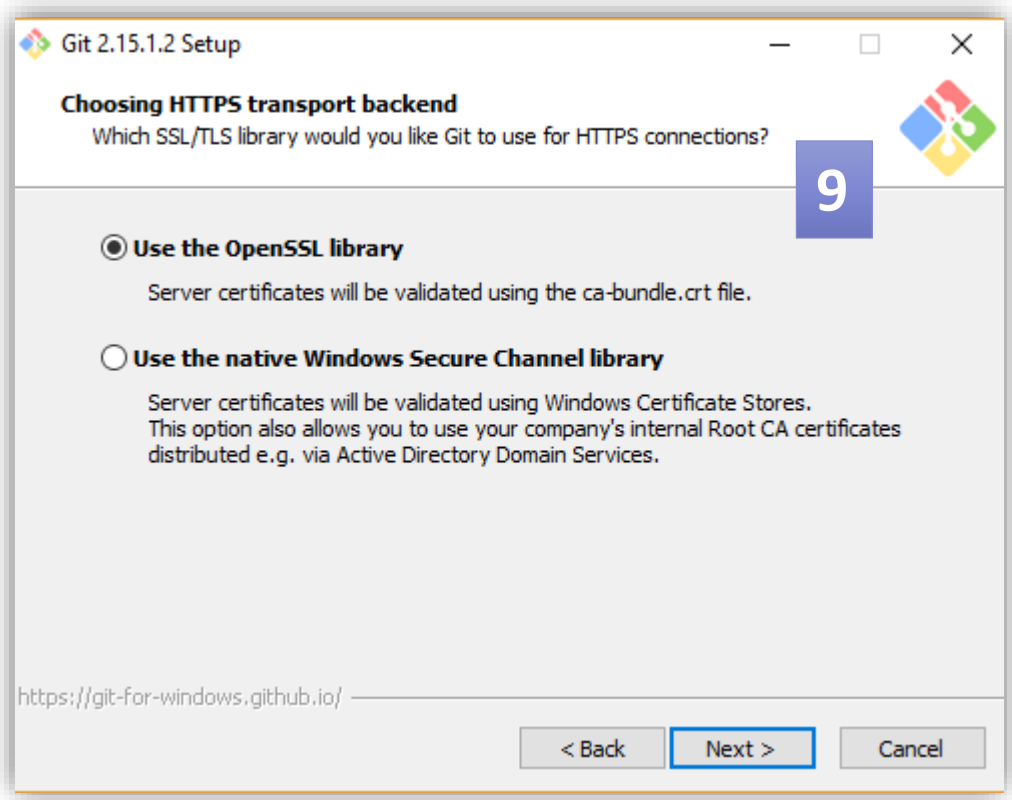


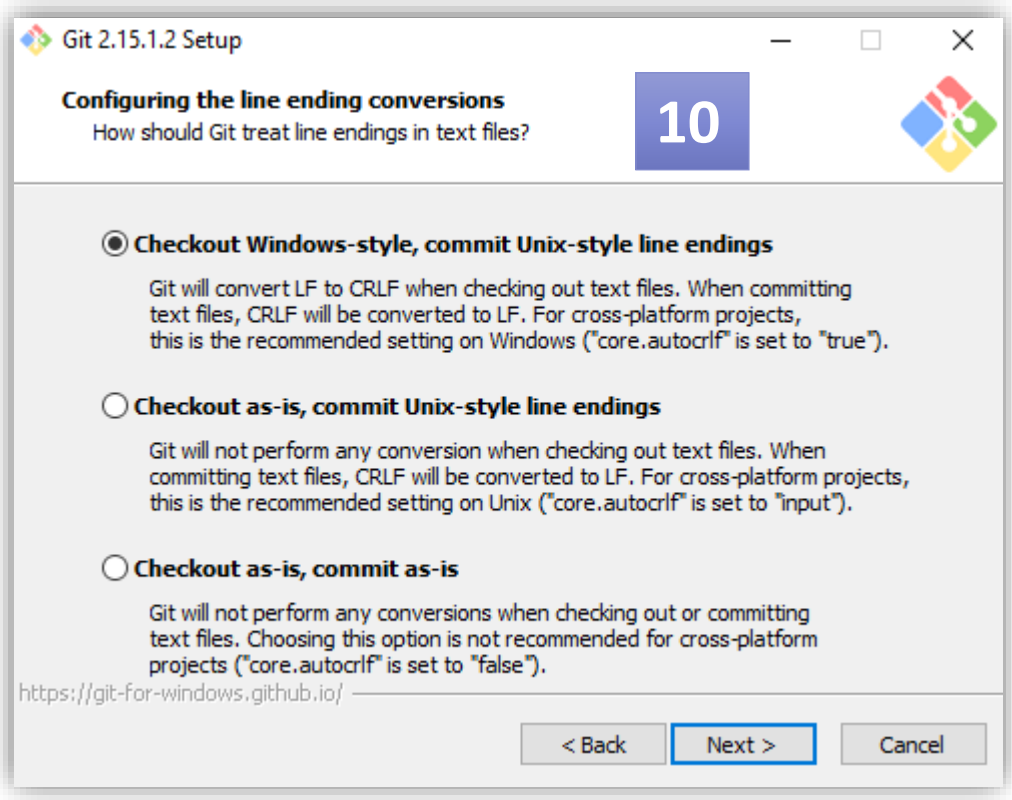
GIT Installation on Windows



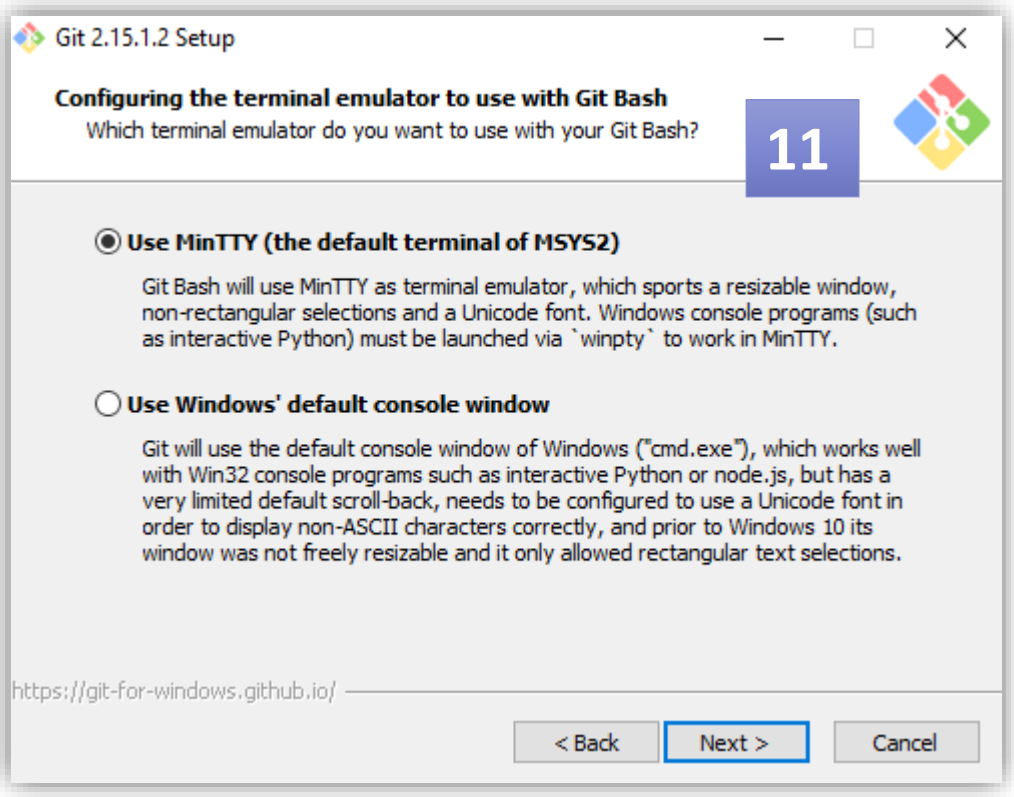


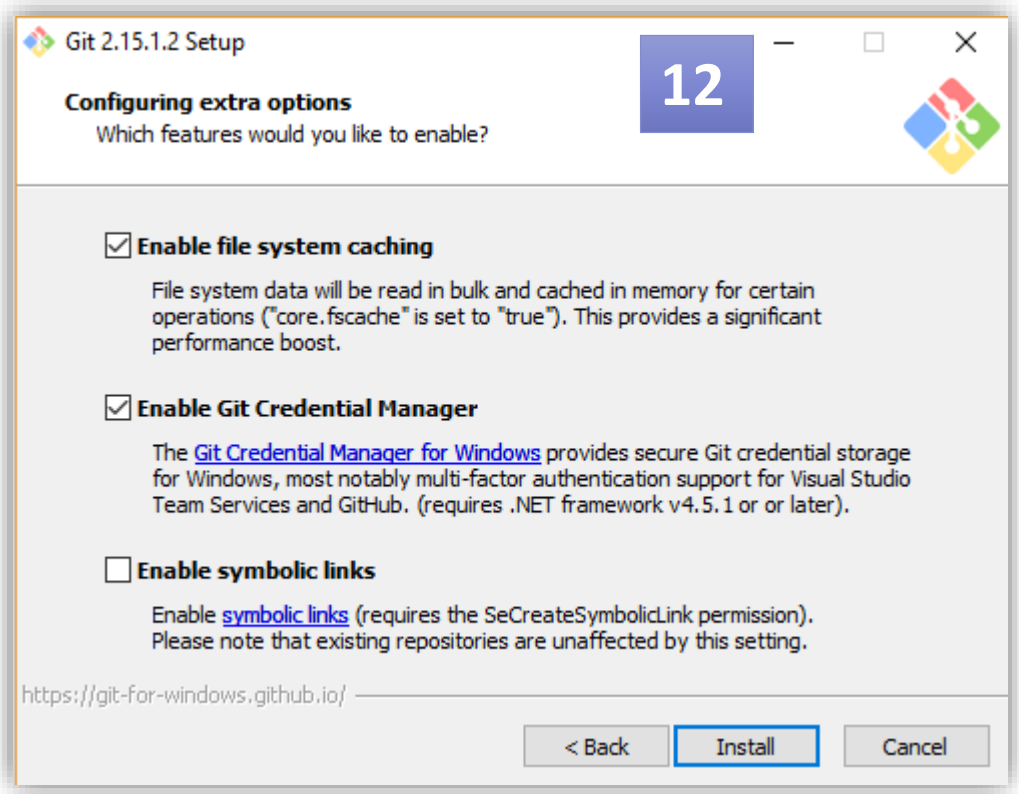
GIT Installation on Windows



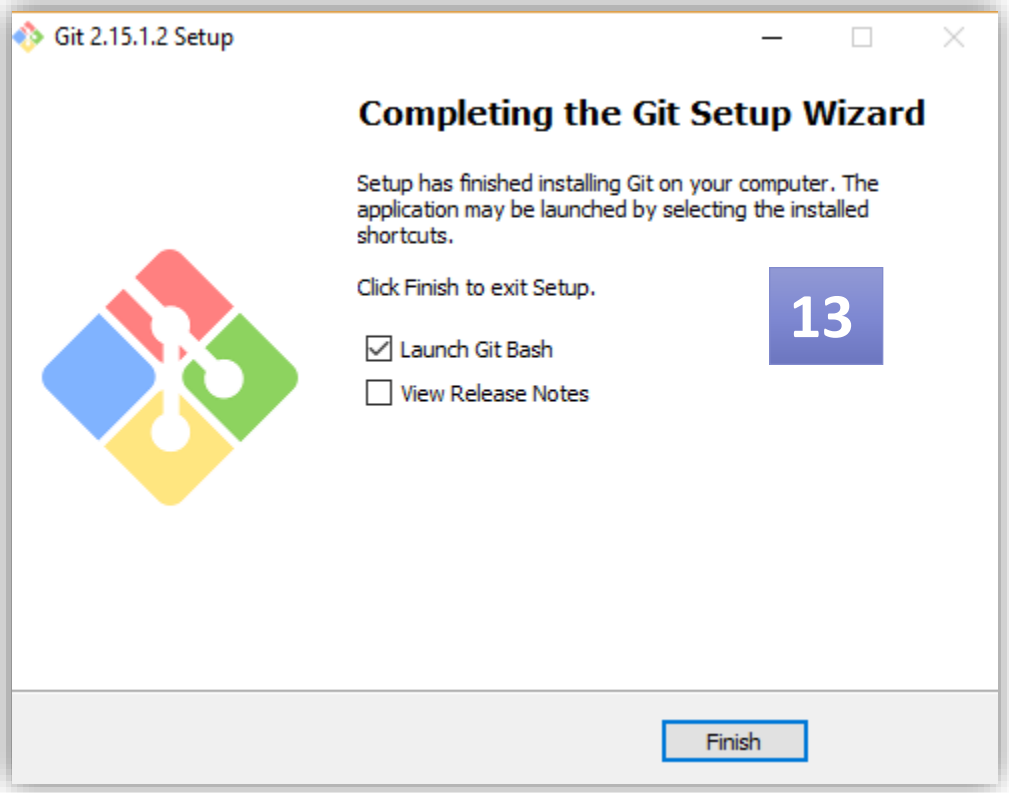


GIT Installation on Windows

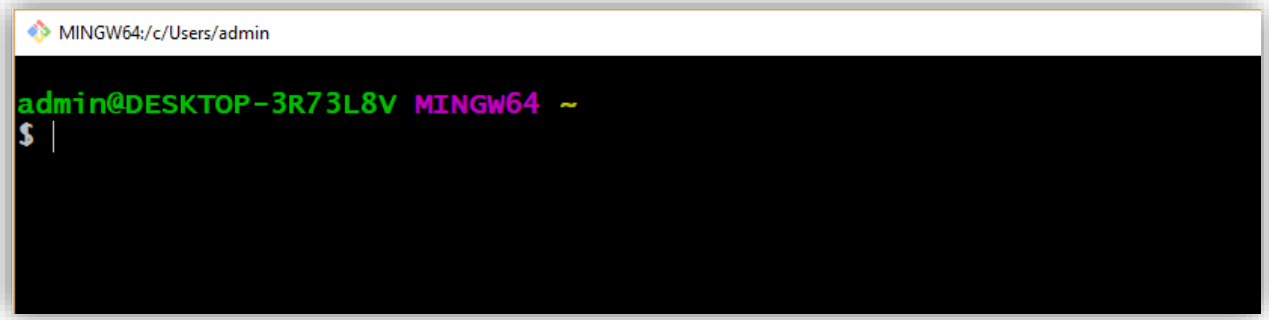




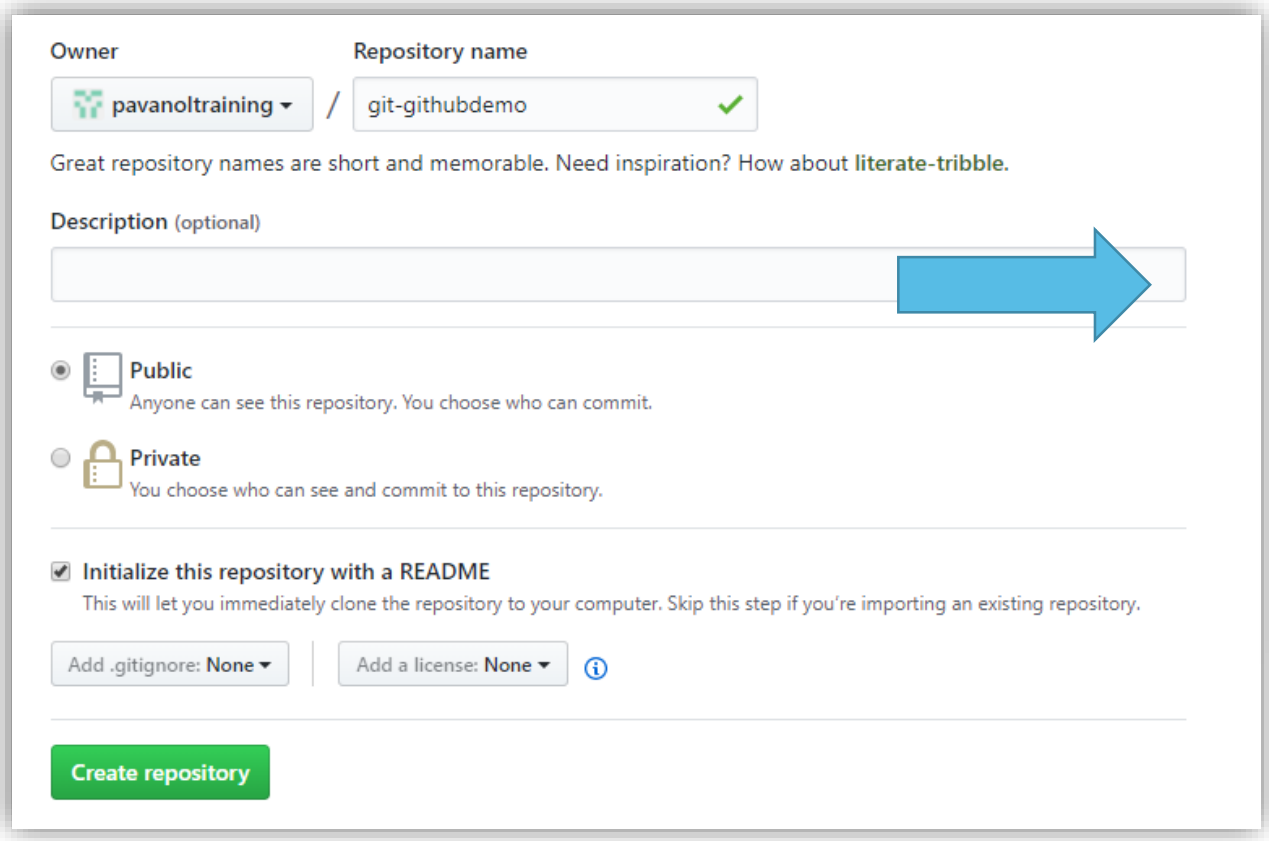
GIT Installation on Windows

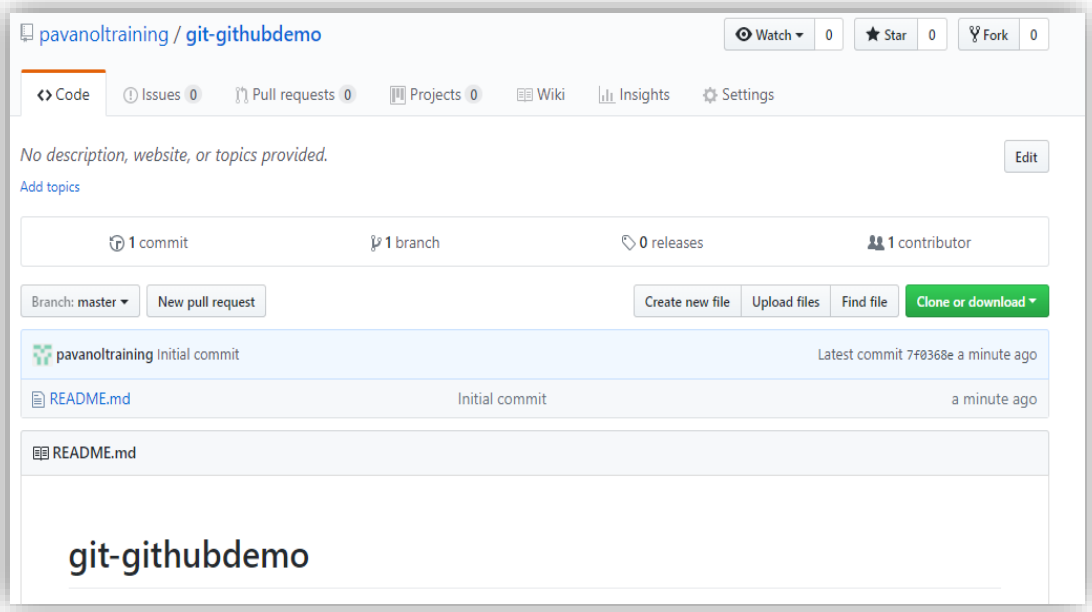


This will launch Git Bash on your screen which looks like the snapshot below:



GITHUB Account





Workflow


Working Directory
Staging Area
Git Repository
Add command
Commit command
Push command
Untracked File
Tracked File
Committed File
Remote File

Git commands

Command	Usage
$ git init	This command is used to start a new repository
$ git remote add origin “URL of git hub repository"	This command is used to add a “remote” repository URL <url> which is referred in other git commands (such as pull or push) with the provided name
$ git status	This command lists all the files that have to be committed.
$ git add -A	This command add all the files to the staging area.
$ git config --global user.name “Your Name" $ git config --global user.email “Your email ID"	This command sets the author name and email address respectively to be used with your commits.
$ git commit -m "This is my first commit!"	This command commits any files you’ve added with the git add command and also commits any files you’ve changed since then.
$ git push -u origin master	This command sends the committed changes of master branch to your remote repository.

Pulling files from Github to git repository

- $ git pull origin master

Step by Step Execution

- **Setup**

  $ git init

  $ git remote add origin "https://github.com/pavanoltraining/inetbankingV1.git"

  $ git status

  $ git add -A

  $ git config --global user.name "pavan"

  $ git config --global user.email "pavanoltraining@gmail.com"

  $ git commit -m "This is my first commit!"
- **Round2:**

  $ git status

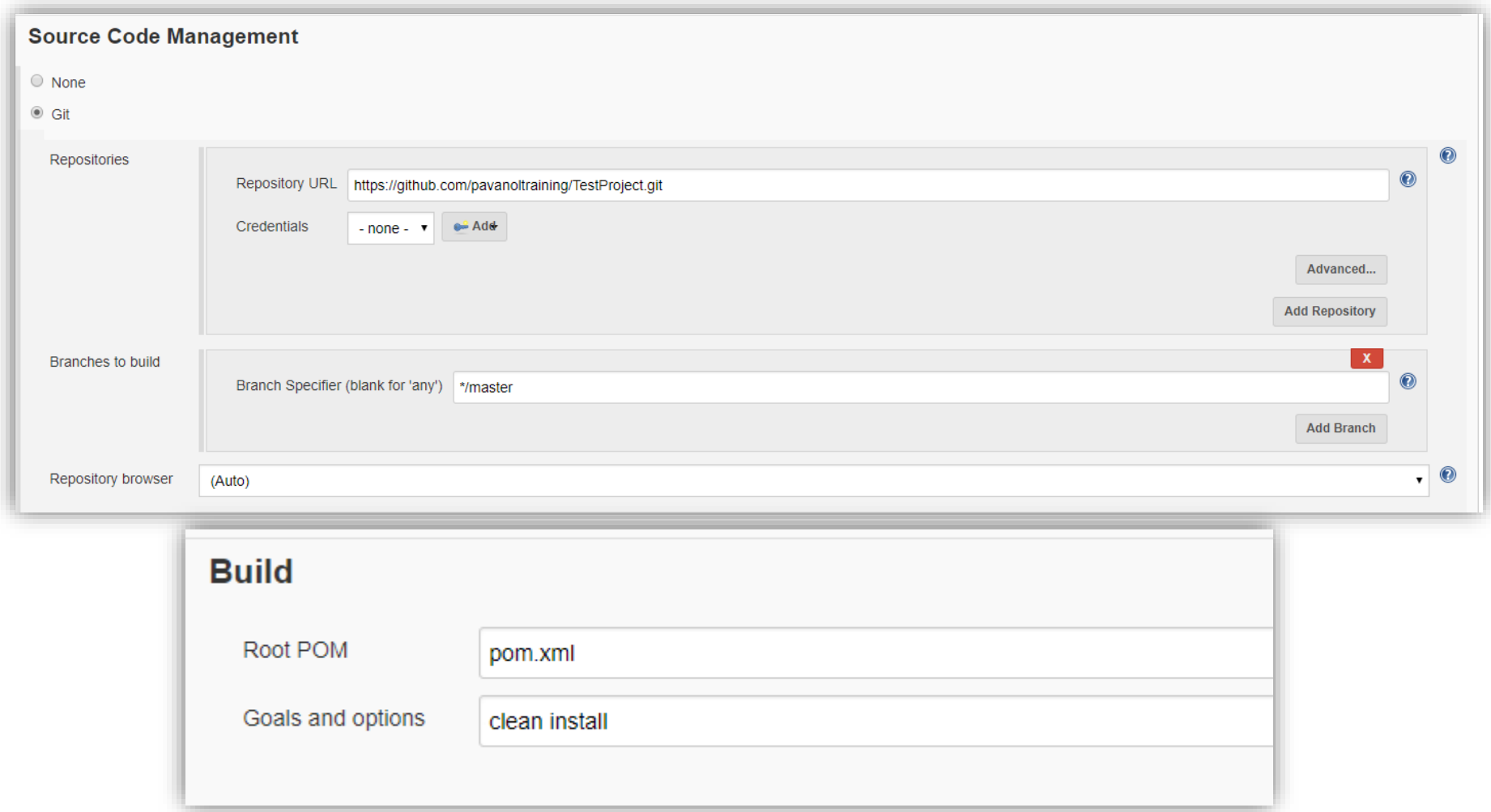
  $ git add -A

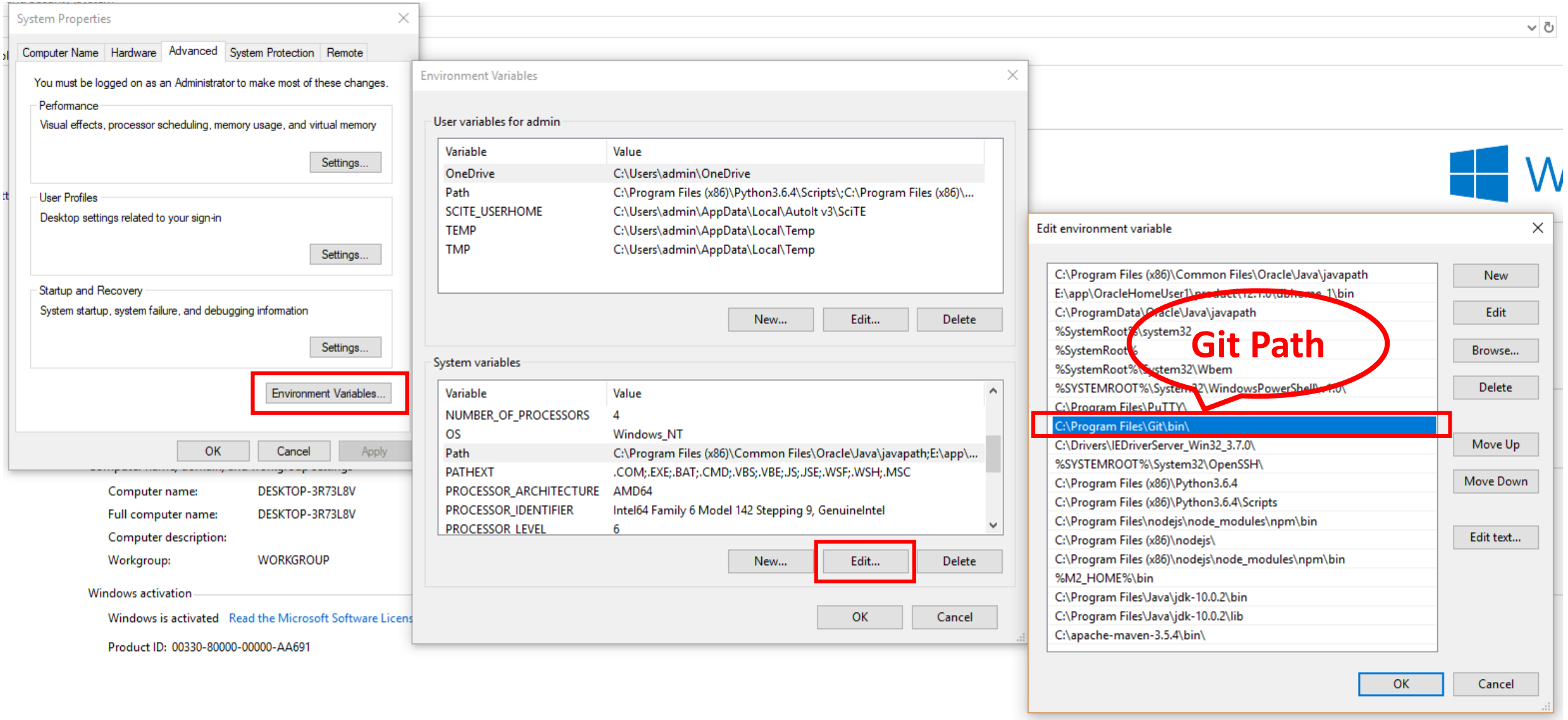
  $ git commit -m "This is my first commit!"
- **Pushing the files from git to Git Hub**

  $ git push -u origin master
- **Pulling files from Github to git repository**

  $ git pull origin master

Run Github project from Jenkins




System Properties
Computer Name
Hardware
Advanced
System Protection
Remote
You must be logged on as an Administrator to make most of these changes.
Performance
Visual effects, processor scheduling, memory usage, and virtual memory
Settings...
User Profiles
Desktop settings related to your sign-in
Settings...
Startup and Recovery
System startup, system failure, and debugging information
Settings...
Environment Variables...
OK
Cancel
Apply
Environment Variables
User variables for admin
Variable
Value
OneDrive
C:\Users\admin\OneDrive
Path
C:\Program Files (x86)\Python3.6.4\Scripts\;C:\Program Files (x86)\...
SCITE_USERHOME
C:\Users\admin\AppData\Local\AutoIt v3\SciTE
TEMP
C:\Users\admin\AppData\Local\Temp
TMP
C:\Users\admin\AppData\Local\Temp
New...
Edit...
Delete
System variables
Variable
Value
NUMBER_OF_PROCESSORS
4
OS
Windows_NT
Path
C:\Program Files (x86)\Common Files\Oracle\Java\javapath;E:\app\...
PATHEXT
.COM;.EXE;.BAT;.CMD;.VBS;.VBE;.JS;.JSE;.WSF;.WSH;.MSC
PROCESSOR_ARCHITECTURE
AMD64
PROCESSOR_IDENTIFIER
Intel64 Family 6 Model 142 Stepping 9, GenuineIntel
PROCESSOR_LEVEL
6
New...
Edit...
Delete
OK
Cancel
Edit environment variable
C:\Program Files (x86)\Common Files\Oracle\Java\javapath
C:\ProgramData\Oracle\Java\javapath
%SystemRoot%\system32
%SystemRoot%
%SystemRoot%\System32\Wbem
C:\Program Files\PuTTY\
C:\Program Files\Git\bin\
C:\Drivers\IEDriverServer_Win32_3.7.0\
%SYSTEMROOT%\System32\OpenSSH\
C:\Program Files (x86)\Python3.6.4
C:\Program Files (x86)\Python3.6.4\Scripts
C:\Program Files\nodejs\node_modules\npm\bin
C:\Program Files (x86)\nodejs\
C:\Program Files (x86)\nodejs\node_modules\npm\bin
%M2_HOME%\bin
C:\Program Files\Java\jdk-10.0.2\bin
C:\Program Files\Java\jdk-10.0.2\lib
C:\apache-maven-3.5.4\bin\
Git Path
New
Edit
Browse...
Delete
Move Up
Move Down
Edit text...
OK
Cancel
Computer name: DESKTOP-3R73L8V
Full computer name: DESKTOP-3R73L8V
Computer description:
Workgroup: WORKGROUP
Windows activation
Windows is activated
Product ID: 00330-80000-00000-AA691

End