**How to commit and push Maven Project into Github**

1. First creat a new repository in GitHub

New - Give name and select readme checkbox - and create

2. Generate access token in GitHub

Profile icon- settings - developer settings - token classic - generate new token(classic) - check repo - generate token - copy and save the token in somewhere

3. Clone the newly created repository to Eclipse

From GitHub – click Code -select http code url and copy that URL .

Go to Eclipse - window - show view - other - Git - Git repository - open – clone(a small page will be displayed in the current window) - clone url – (the all the details related to the git repository automatically displayed in the next window )next-next-select main - next – finish.

4. Push project to Git

From eclipse - right click the project - team - share project - click repository drop-down and select the added repository - finish

Again right click on project - team - commit - add needed file from unstaged to staged - commit and push

Commit : Commit - committing is the process which records changes in the repository.

The "commit" command is used to save your changes to the local repository.

Push: Push - pushing sends the recent commit history from your local repository up to GitHub.

Commit and push

Main

Branch

**Debugging in Eclipse**

Resume(F8):-Resumes normal execution of the program until the next breakpoint is reached.

Step over :-Executes the current line of code and goes to the next line without stepping into any method calls or associated scope (e.g. loops and conditions) of the current line.

Step into:- Executes the current line of code and dives into the next line of code in the program execution. If the current line calls a method, the debugger steps into the method.

Step Return:-(F7)Steps back out of the current method and returns to the caller of the method

**How to PUSH and commit to GITHUB from Intellij idea IDE**

Create a new repository in github

While creating a repository no need **to tick add a ReadME checkbox.**

### then copy the url in the title…or push an existing repository from the command line.

### Sample repository

git remote add origin https://github.com/AnasNazar880/sample\_git.git

Create a new project in the Intellij idea .

Then open Terminal in the intellije and there we have powershell in the top in the terminal .we have to change it .in order to do change that go to file-settings-search Terminal -click terminal in the tool menu and change the shell path to cmd.exe file and click apply and ok button.

Then come to the terminal ,there we have Microsoft window version will be seen.

**Git Initialization**

Select the project and right click then goto open in -exlorer-we have the project folder-open the project folder -then type “git init”in the terminal and then go to view option in the explorer -show -hidden files.there we can see a **.Git file** is created in the project folder.

**Link the Repository/staging**

Copy paste the git repository from the github git remote add origin <https://github.com/AnasNazar880/sample_git.git> to the terminal in intellij.

Then type **git add .** command in the terminal.

At this time the local repository is linked with github repository .

**Commit the git** (ready to push )

using the command **git commit -am “any comment”(for committing a group of files)** in the terminal .then we will get a list of files that need to be commit in the terminal .

**git commit “copy paste the file location ”** command for committing a file to git.

**PUSH the git**

* PUSH the git using the command **git push** in the terminal.
* If you are using first time pushing you have to authorize through a popup window and click authorize through browser.
* While making changes in the program, in order to view that changes in the program click git icon in the left bottom in the window.
* Commit and push can be done through the terminal by using different git commands as well as the IntelliJ UI.

**To see changes in the program in Git Repo**

The changes in the codes will not be seen in the git repository. in order to see that changes in the git repo ,click Git in the top of the window .and click commit .then again click git -push .

**SELENIUM (version 4.10.0)-History**

* Selenium is one of the most widely used open source Web UI (User Interface) automation testing suite.
* it is for automating web applications for testing purposes
* Selenium supports automation across different browsers, platforms and programming languages.
* Current version of selenium is 4.10.0
* Official website of Selenium is https://www.selenium.dev/
* Selenium was originally developed by **Jason Huggins in 2004** as an internal tool at **ThoughtWorks**.
* **Shinya Kasatani** developed Selenium Integrated Development **Environment (IDE)** in 2006 as a Firefox plugin that helps create tests.

**Automation Testing**

* Testing an web application using a programming language without human interpretation.
* Automation testing uses the specialized tools to automate the execution of manually designed test cases without any human intervention.
* Automation testing tools can access the test data, controls the execution of tests and compares the actual result against the expected result.

**Advantages of Automation Testing**

* It provides rapid feedback to developers.
* It provides disciplined documentation of test cases.
* Less error prone as compared to manual testing.
* It provides unlimited iterations of test case execution.
* Automation testing supports frequent regression testing.

While considering Automation Testing in a firm ,Consider the following points

* Automation testing is costlier/difficult for a small firm to implement .so the company will think twice while go for automation.
* Budget of the project will be considered .
* Return of investment will be considered.
* If the number of testcases for testing is considerably lower .AT is not used
* Feasibility
* Based on the repetitiveness of testing of testcases such as regression testing ,smoke testing.

**Why selenium with java**

* Java is more **widely used in commercial applications** than other programming languages like Python, **so integrating Selenium tests is easier.**
* Programs **written in Java are faster than** other popular languages like Python
* Java  has a **massive community that provides large-scale support** with a variety of resources.

**Features of Selenium**

* Selenium is an open source and portable Web testing Framework.

Selenium supports various operating systems, browsers and programming languages.

* Programming Languages: C#, Java, Python, PHP, Ruby, Perl, and JavaScript
  + - Operating Systems: Android, iOS, Windows, Linux, Mac, Solaris.
    - Selenium Supports various programming languages because **Selenium Directly communicates with HTML elements in the application or Fronded of web application (UI) such as HTML Tags and Attributes.**
  + Browsers: Google Chrome, Mozilla Firefox, Internet Explorer, Edge, Opera, Safari, etc.
* **Cross Browser Testing** is possible in Selenium .

**Flavors of Selenium or Components of Selenium**

1,Selenium‐IDE is the Integrated Development Environment for building Selenium test cases.

2,Selenium‐RC allows the test automation developer to use a programming language for maximum flexibility and extensibility in developing test logic.

Selenium‐RC provides an [API](https://tekslate.com/api-testing-interview-questions) (Application Programming Interface) and library for each of its supported languages: [**HTML**](https://tekslate.com/html-tutorials)**,**[**Java**](https://tekslate.com/javascript-tutorials)**,**[**C#**](https://tekslate.com/c-sharp-tutorials)**, Perl, PHP,**[**Python**](https://tekslate.com/tutorials-python)**, and**[**Ruby**](https://tekslate.com/ruby-rails-tutorials)

**3. Selenium-Grid**

Selenium Grid is a **tool used together with Selenium RC to run parallel tests across different machines and different browsers all at the same time.** Parallel execution means running multiple tests at once.

**4. Selenium Web Driver/Selenium 2.0**

* The WebDriver proves itself to be better than both Selenium IDE and Selenium RC in many aspects.

**Comparison of Components of Selenium**

|  |  |  |  |
| --- | --- | --- | --- |
| Functionality | Selenium IDE | Selenium RC | Selenium Web driver |
| SERVER | it requires no server to start execution of test cases. | It requires the server to start execution of test cases. | It requires no server to start execution of test cases. |
| Navigation | Cannot handle navigation. | Cannot handle navigation | Can handle navigation |
| Alerts | Cannot handle alerts. | Cannot handle alerts. | Can handle alerts |
| Mouse Actions | Cannot handle mouse actions. | Cannot handle mouse actions. | Can handle mouse actions. |
| Architecture | Derived from Javascript. | Derived from Javascript. | Not derived from Javascript. |
| Dropdown | Cannot handle dropdown. | Cannot handle dropdown. | Can handle dropdown. |
| iPhone/Android | Cannot perform iPhone/Android testing. | Cannot perform iPhone/Android testing. | Can perform iPhone/Android testing with the help of Android Driver, iPhone Driver. |

How to setup an Selenium based project

Create a maven project named SeleniumCommands ,java as the language and type of project is Maven. Add dependencies for selenium (go to Maven repositories for selenium in the web and copy the dependency and paste it in pom.xml folder in the project)Then create class named BrowserLaunch

Browser Launch

Webdriver is an interface implements to the ChromeDrive,FireFoxDriver,(classes) etc

Driver is an Object reference ;it is named by the user.

|  |  |  |
| --- | --- | --- |
| 1 | Google Chrome Browser | Webdriver driver=new ChromeDriver(); |
| 2 | FireFox Browser | Webdriver driver=new FireFoxDriver(); |
| 3 | Microsoft Edge Browser | Webdriver driber=new EdgeDriver(); |
|  |  |  |

**Browser Commands**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Number | Commands | Syntax | Usage/definition | Example |
| 1 | driver.manage().window().maximize(); |  | For maximizing the browser window |  |
| 2 | driver.get(appUrl) | ***get(String arg0)***: ***void*** | ***Load*** a new web page in the current browser window. | driver.get("https://www.go"); or |
| 3 | ***driver.getTitle();*** | ***getTitle(): String*** | This method fetches the ***Title*** of the current page. | String Title = driver.getTitle(); |
| 4 | ***driver.getPageSource();*** | ***getPageSource()*** | returns the ***Source Code*** of the page |  |
| 5 | ***driver.close();*** | ***close(): void*** | Close only the current window the WebDriver . | driver.close(); |
| 6 | ***driver.quit();*** | ***quit(): void*** | method ***Closes*** all windows opened by the *WebDriver* | driver.quit(); |
| 7 | driver.manage().deleteallcookies |  | To delete cookies | driver.manage().deleteallcookies() |

**Web Elements Commands**

FindElement command is used o **uniquely identify a web element** within the web page.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| sl no | **Commands** | **Syntax** | **definition** | **Example** |
| **1** | **sendKeys()** | *PasswordField.sendKeys(“text”);* | allows the user to type content automatically into an editable field while executing tests | passwordField.sendKeys("Hyrin@2023"); |
| **2** | FindElement(By.id(“ ”)) | driver.findElement(By.*id*("Email")); | to uniquely identify a web element within the web page. | WebElement emailField = driver.findElement(By.*id*("Email")); |
| **3** |  |  |  |  |

**Locators in Selenium**

* Locators in Selenium WebDriver provide mechanisms for identifying HTML elements on the page.
* **The identification of the WebElement (**or HTML element) is made using locators in Selenium WebDriver..
* web locators for locating WebElements on the page.

**Types of Locators in Selenium**

1. ID
2. Name
3. ClassName
4. LinkText
5. PartialLinkText
6. TagName
7. CssSelector
8. xpath

|  |  |  |
| --- | --- | --- |
| locator | Description | Synatx |
| ID | Identify the webelement using id attribute | driver.FindElement(By.id(“id value”)); |
| Name | Identify the Webelement using Name attribute | driver.FindElemrnt(By.Name(“name value”)); |
| ClassName | Use the *Class* attribute for identifying the object | driver.findElement(By.className(“classValue”)); |
| LinkText |  |  |
| Partial LinkText |  |  |
| TagNmae |  |  |
| CssSelector |  |  |
| xpath | Use XPath to locate the WebElement | driver.findElement(By.xpath(“xpathValue”)); |

* ID locator in Selenium is the most preferred and fastest way to locate desired WebElements on the page. ID Selenium locators are unique for each element in the DOM.
* ClassName locators are not so used .