|  |  |  |
| --- | --- | --- |
| java | Java & Eclipse |  |
| Eclipse | m2e plugin TestNG plugin |  |
| Maven | Create new Maven Project -Quick Start template GroupID - ArtifactID - ProjectName | A new project with the given name - ArtifactID is created with : pom.xml src/main/java src/main/resources src/test/java src/test/resources |
| pom.xml  Maven Dependencies in pom.xml | Add/Copy required Maven dependencies to pom.xml   **selenium testNg log4J - log4j.api log4j-core poi - poi & poi-ooxml extentreports** | Paste it in the pom.xml -  inside **<dependencies>** tag Save file to build automatically.  Check in Maven Dependencies folder. |
| Project Packages & Folders | 1. In src/test/java- Create packages com.pjtname.pageobjects com.pjtname.testcases com.pjtname.testdata com.pjtname.utilities  2. in Src/test/java - Create folders Configuration Drivers ScreenShots |  |
| Project Packages & Folders | Adding browser driverexes to project path in Src/test/java >Drivers folder > ChromeDriver.exe GeckoDriver.exe IEDriverServer.exe | Copy paste all browserdriver exes to Drivers folder ChromeDriver.exe GeckoDriver.exe IEDriverServer.exe |
| POM - PageObject Model Design Pattern | PageObjects design pattern - @FindBy  Create page objects class of applin pages for PageObjects design pattern. 1. Create a class - In src/test/java> com.pjtname.pageobjects > create class file | PageObjectsModel -Pagefactory deisgn pattern - @FindBy  1.PageFactory class & initElements() method 2.@FindBy ()..WebElement variabename 3. Create a method for each pageobject for Action to be performed on that page object |
| TestCases | Creating TestCases.- BaseClass  Create a base test case class for automation  In src/test/java> com.pjtname.testcases > create class file > BaseClass.java | setup() teardown() CaptureScreenshots fileupload/download conversions alert/popup checks |
| TestNG  1.TestNG jars in pom.xml 2. testng plugin in Eclipse | Creating TestCases.  Create test case class   In src/test/java> com.pjtname.testcases > create class file > LoginTest.java | 1. Use @Test  2. steps for validation in a method 3. get PageObjects of this web page of the applin from its PageObjects class |
| Apache Log4J  1. log4j.api & log4j-core in pom.xml 2. Log4j.property file OR Log4j.xml file - paste to Project Home path | For Creating Logs 1. log4j jars 2. log4j.property/log4j.xml file 3. testscript in testcases classes | For Creating Logs 1. log4j jars in pom.xml 2. log4j.property/log4j.xml file -project home 3. get Logger Class & PropertyConfigurator/DOMCOnfigurator in testscript |
| Config.Properties data file | Creating a configdata.properties file for storing global data  1. config.properties file in Configuration folder 2.ReadConfig.java in utilities package 3. testscript in base class | 1. Adding config.properties to project path in Src/test/java> Configuration folder> 2. In src/test/java> com.pjtname.utilities> ReadConfig.java 3. Import ReadConfig.java in BaseClass.java - create class object |
| Cross Browser Testing  TestNG XML file | Cross Browser Testing  1. Annotation@Parameters("") in class file 2. Different browser options  3. browser drivers path config.properties file 4. Create testng.xml file 5. Add <parameters> tag in testng.xml file |  |
| Extent Reports  1. poi & poi-ooxml jars in pom.xml 2. Extent Reports Config.xml  3. Excel testdata file 4. XLUtils.java - utility file | Creating Extent Reports  1. Extent Reports jars 2. Extent Reports Config.xml  3. Excel testdata file 4. TestNG Listeners - utility file 5. <Listeners> tag in testng.xml  6. Run TestNG XML file | Creating Extent Reports  1. Extent Reports jars in pom.xml 2. Extent Reports Config.xml in project home 3. TestNG Listeners - utility file in src/test/java/com.pjtname.utilities packkge 4. Excel testdata file -in src/test/java/com.pjtname.testdata package 5. <Listeners> tag in testng.xml - add the Listeners utility file in <Listeners classname="packagename.classname"> in testng.xml file 6. Run TestNG XML file - Reports will be generated as per the format given in extentreports.xml/properties file |
| Data Driven Testing Apache poi  1. poi & poi-ooxml jars in pom.xml 2. Excel testdata file 3. XLUtils.java - utility file | Creating DataDrivenTests  1. poi & poi-ooxml jars in pom.xml  2. Excel testdata file with data 3. XLUtils.java - utility file  4. TestCase class file to testNG.xml file | Creating DataDrivenTests  1. poi & poi-ooxml jars in pom.xml  2. Excel testdata file with data 3. XLUtils.java - utility file in src/test/java/com.pjtname.utilities package - to read the excel data file 4. @DataProvider in testcase class and give @DataProvider method name in @Test method  5.Get webpage objects from pageobjects class - Create/Use pageobject class 6.Add this testcase class to tetsng.xml file- RunAs TestNG suite |
| New Test Case | Adding new test cases classes 1. Page object class for the testcase 2. Testcase class for the tests 3. Add the testcase class to testng.xml file | Adding new test cases classes 1. Create page object class for the testcase- in src/test/java> com.pjtname.pageobjects - New> Class>  2. Create testcase class in src/test/java> com.pjtname.testcases- New> Class>  3. Testscript for the tests 4. Add the testcase class to testng.xml file |
| Test Execution | 1. pom.xml - surefireplugin 2. Maven CLI- maven commands 3. Jenkins- simple project 4. Jenkins-maven project |  |
| Jenkins Github |  |  |
|  |  |  |
|  |  |  |