Selenium Java Training - Session 33 - Developing a Framework and Automating Live Project (Part 1)

* Create a Maven Quick-start Project
* Add the dependency tags for below:
  + WebDriverManger
  + Selenium Java
  + TestNG
  + Log4j (log4j-api and log4j-core)
  + Apache Commons IO
  + ExtentReports
* Create a Base class
  + Create resources package under 'src/main/java'
  + Create Base class under this package
  + Create a method for initializeBrowser inside the Base class - [View here](https://drive.google.com/file/d/1LVXANb4Dz-pcOMV6G_WkA8HTpxS79AqA/view?usp=sharing)
  + Read the browser name from Properties file, instead of hardcoding it
    - create data.properties file under resources pack
    - Add this property browser=chrome to the file
    - Modify the initializeBrowser code accordingly  - [View here](https://drive.google.com/file/d/1EAosLC_4pK5eTNjeEkvf7FQSp7zRBfJA/view?usp=sharing)
* Create a Class for 'LoginTest' under 'src/test/java' under 'tests' package
  + Create a method say 'login' and annotate with @Test
  + Extend the LoginTest class with Base class
  + Write the code to initialize the browser and open the Application URL
  + Run the LoginTest to see, if this is executing so far.
  + Remove the URL hardcoding and read the URL from the properties file
  + Run the LoginTest to see, if this is executing after this change - [View here](https://drive.google.com/file/d/1lrpkHGHn1_IHrnNK16oi9PzaeDX822jS/view?usp=sharing)
    - Add this property url= <http://tutorialsninja.com/demo/> to the properties file
* Write the remaining code of LoginTest by creating Page Objects
  + Create a package under 'src/main/java' as 'pageobjects'
  + Create a class say 'LandingPage' under this package with page objects - [View here](https://drive.google.com/file/d/1DPYnuM0U1bmxHO7-qeev_zggbE8N0kcr/view?usp=sharing)
  + Create an object for LandingPage from LoginTest and perform required operations - [View here](https://drive.google.com/file/d/1Yt8TQ-ul8WFwGYjFhsbXz42uGT9qiPc6/view?usp=sharing)
  + Run the LoginTest to see, if this is executing so far.
  + Create a class say 'LoginPage' under this package with page objects - [View here](https://drive.google.com/file/d/1PKF5FdfrGamwfrAILcm3niTKq7sOGcAi/view?usp=sharing)
  + Create an object for LoginPage from LoginTest and perform required operations - [View here](https://drive.google.com/file/d/1RSMlwkl_ccLv0Hcl784sl4pRW2TBgR0W/view?usp=sharing)
  + Run the LoginTest to see, if this is executing so far.
  + Create a class say 'AccountPage' under this package with page objects - [View here](https://drive.google.com/file/d/1gz1xPm6CUyaCuTcRx4T6aK0OPEOXk55z/view?usp=sharing)
  + Create an object for AccountPage from LoginTest and perform required operations - [View here](https://drive.google.com/file/d/1Dn45RWoRGzAb50Fli5eQ4MPqcawUSgGf/view?usp=sharing)
  + Run the LoginTest to see, if this is executing so far.
  + Remove the email and password hardcoding and read them from the properties file
  + Write @AfterMethod to close the browser - [View here](https://drive.google.com/file/d/1XsRfjOWlK9fXorwq2xmFZLjHv73WVsCO/view?usp=sharing)
* Parameterize the Test by passing multiple sets of data from DataProvider annotated method
  + Create a DataProvider annotated method and pass the data to the Test annotated method - [View here](https://drive.google.com/file/d/1F2RYi8WgPkgfYv1yTV4qVhjVouuVkn9x/view?usp=sharing)
* Create @BeforeMethod
  + Organize the code for opening the Application - [View here](https://drive.google.com/file/d/1-pttFyM5kt-cukNKTlU7NaqBj43Vbgnp/view?usp=sharing)
* Create some dummy Test Classes with simple SOP statements
  + TestTwo
  + TestThree
  + TestFour
* Convert the Project to TestNG
  + Right click on the Project, Select TestNG > Convert to TestNG
  + Execute the Tests as a group using testng.xml file
* Integrate with Maven to run the Project from the command line
  + Search for Maven TestNG configuration and add the plugin for Maven Surefire plugin having configuration tags
  + Run from the command prompt - mvn test