Selenium Grid (Continued)

- Configuring Grid
  - Download Selenium Standalone Server from seleniumhq.org
    - This jar file has grid capability
    - Create a folder say jars in the project workspace and copy the downloaded jar file into it
  - Setup Hub
    - Open the command prompt and fire the below command:
      - java -jar F:\OD\NewWorkspace\NewSpace\TP\jars\selenium-server-standalone-3.141.59.jar -role hub
    - You will be getting the message 'Selenium Grid hub will be up and running'
    - By default the hub runs on 4444 port number of the machines IP address
    - Open http://localhost:4444/ or http://192.168.60.232:4444/ to check whether the Hub is really running
      - click on console link and observe that everything will be blank for now
  - Setup Node One
    - Make sure whether Java is already installed in the Node machine by using java -version command
    - Download and Copy the driver.exe files, above selenium-server-standalone jar file and node1.json file (Download sample here) in the node machine and udpate the below provided paths for them
      - Modify the node1.json with the host ip and also the max sessions to 1
    - Open the command prompt and fire the below command:
      - java -Dwebdriver.gecko.driver="F:\OD\NewWorkspace\NewSpace\TP\drivers\geckodriver.exe" -Dwebdriver.chrome.driver="F:\OD\NewWorkspace\NewSpace\TP\drivers\chromedriver.exe" -Dwebdriver.ie.driver="F:\OD\NewWorkspace\NewSpace\TP\drivers\IEDriverServer.exe" -jar F:\OD\NewWorkspace\NewSpace\TP\jars\selenium-server-standalone-3.141.59.jar -role node -nodeConfig F:\OD\NewWorkspace\NewSpace\TP\drivers\node1.json
    - After the node is up and running, refresh the above hub page (http://localhost:4444/) in hub machine to check whether the configured node is connected and displayed as started in hub
    - Explain node1.json
  - Setup Node Two
    - Mask sure wether Java is already installed in the Node machine by using java -version command
    - Download and copy the driver.exe files, selenium-server-standalone jar file and node.2json file (Download sample here) in the node machine and udpate the below provided paths for them
    - Modify the node2.json with the host ip, the max sessions to 1 and port number to a different number from node1
    - Open the command prompt and fire the below command:
      - java -Dwebdriver.gecko.driver="F:\OD\NewWorkspace\NewSpace\TP\drivers\geckodriver.exe" -Dwebdriver.chrome.driver="F:\OD\NewWorkspace\NewSpace\TP\drivers\chromedriver.exe" -Dwebdriver.ie.driver="F:\OD\NewWorkspace\NewSpace\TP\drivers\IEDriverServer.exe" -jar F:\OD\NewWorkspace\NewSpace\TP\jars\selenium-server-standalone-3.141.59.jar -role node -nodeConfig F:\OD\NewWorkspace\NewSpace\TP\drivers\node2.json
    - After the node is up and running, refresh the above hub page (http://localhost:4444/) in hub machine to check whether the configured node is connected and displayed as started in hub
- Base
  - Create a package say 'base' and 'BaseTest.java' under it
  - Create a reusable method say openBrowser(Sting browser) in 'BaseTest.java'
  - Make all the tests classes extent BaseTest class
  - Add dependency tags for selenium java in pom.xml file and create an object for WebDriver in reusable mehtod of 'BaseTest.java'
  - Call the openBrowser() method from all the Test classes
    - Create the complete openBrowser() method
    - We will not be using chromDriver(), firefoxDriver() or internetExplorerDriver(), instead we will be using RemoteWebDriver
    - TestNG will send every test case to the hub and before sending it to hub it has to tell the hub in which circum stances (Win Firefox or Mac Chrome etc) the test cases needs to be executed
    - Define DesiredCapabilities class and assign it to null
    - Assign the objects of DesiredCapabilites based on browsers - View here
    - Assing the object of RemoteWebDriver as shown below
      - driver = new RemoteWebDriver(new URL("http://192.168.60.232:4444/wd/hub"), cap);
    - Create @AfterMethod annotated method say closeBrowser() in BaseTest class to close the browsers after each and every test - View here
      - User Ctlr + C to stop the hub and nodes

- Execute the RunParallelSuites class to run all the tests in parallel mode on Grid

---