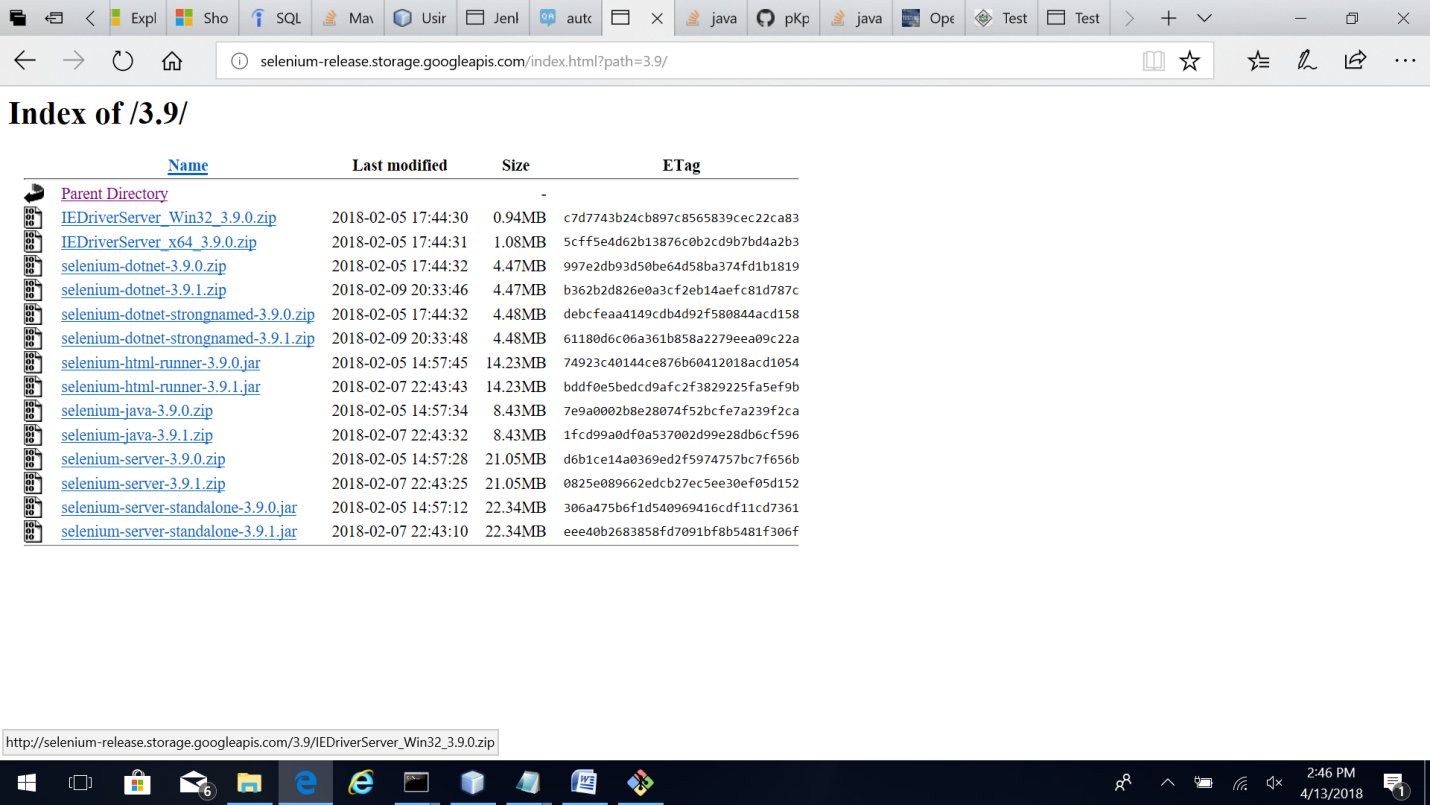
**Running Tests on Selenium Grid**

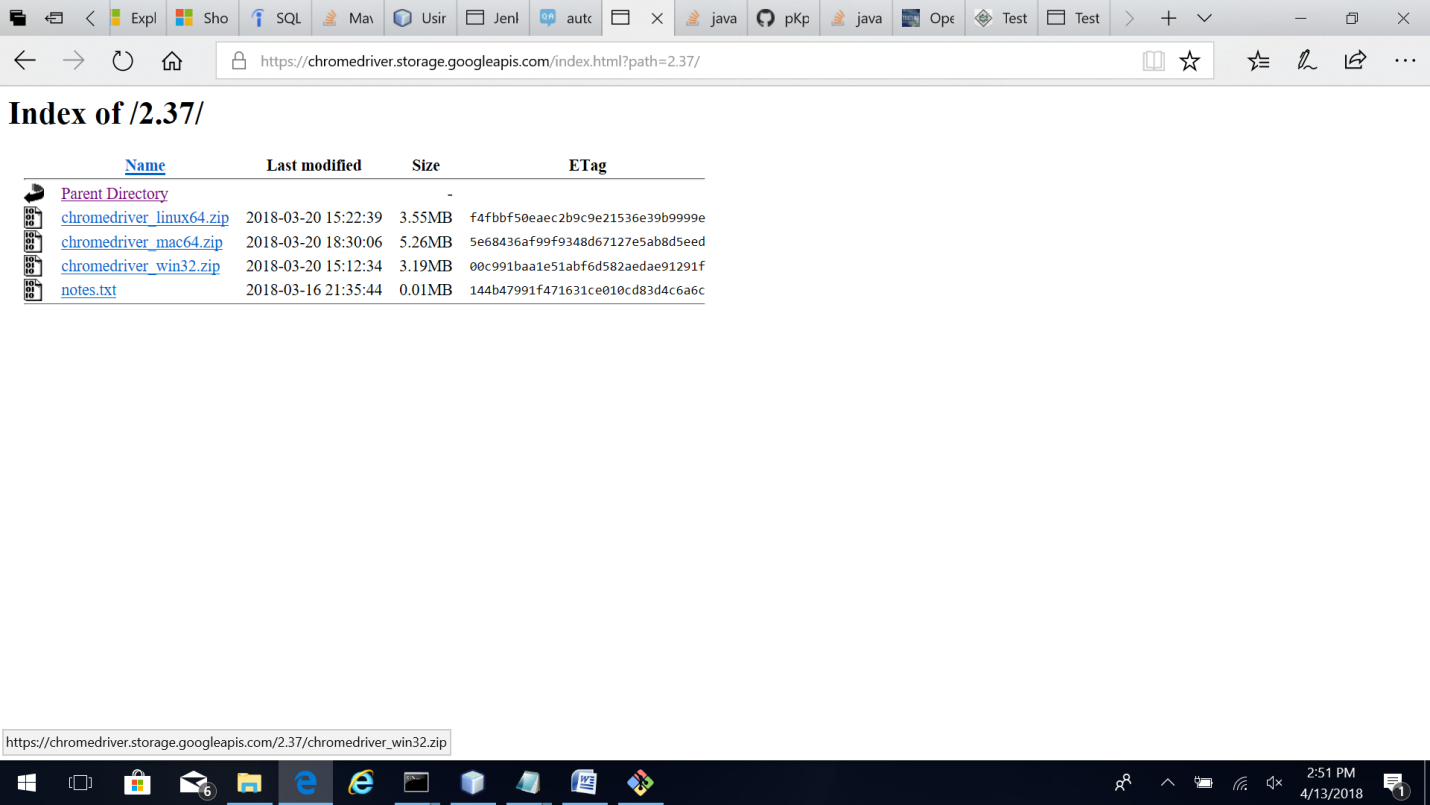
1. Selenium-Grid allows you run your tests on different machines against different browsers in parallel. That is, running multiple tests at the same time against different machines running different browsers and operating systems. Essentially, Selenium-Grid support distributed test execution. It allows for running your tests in a distributed test execution environment.
2. Download Selenium Server Standalone v3.11.0 (selenium-server-standalone-3.11.0.jar) from <https://www.seleniumhq.org/download/>
3. Download individual RemoteWebDriver files for different browsers.
4. For Internet Explorer use the following link:

<http://selenium-release.storage.googleapis.com/index.html?path=3.9/>



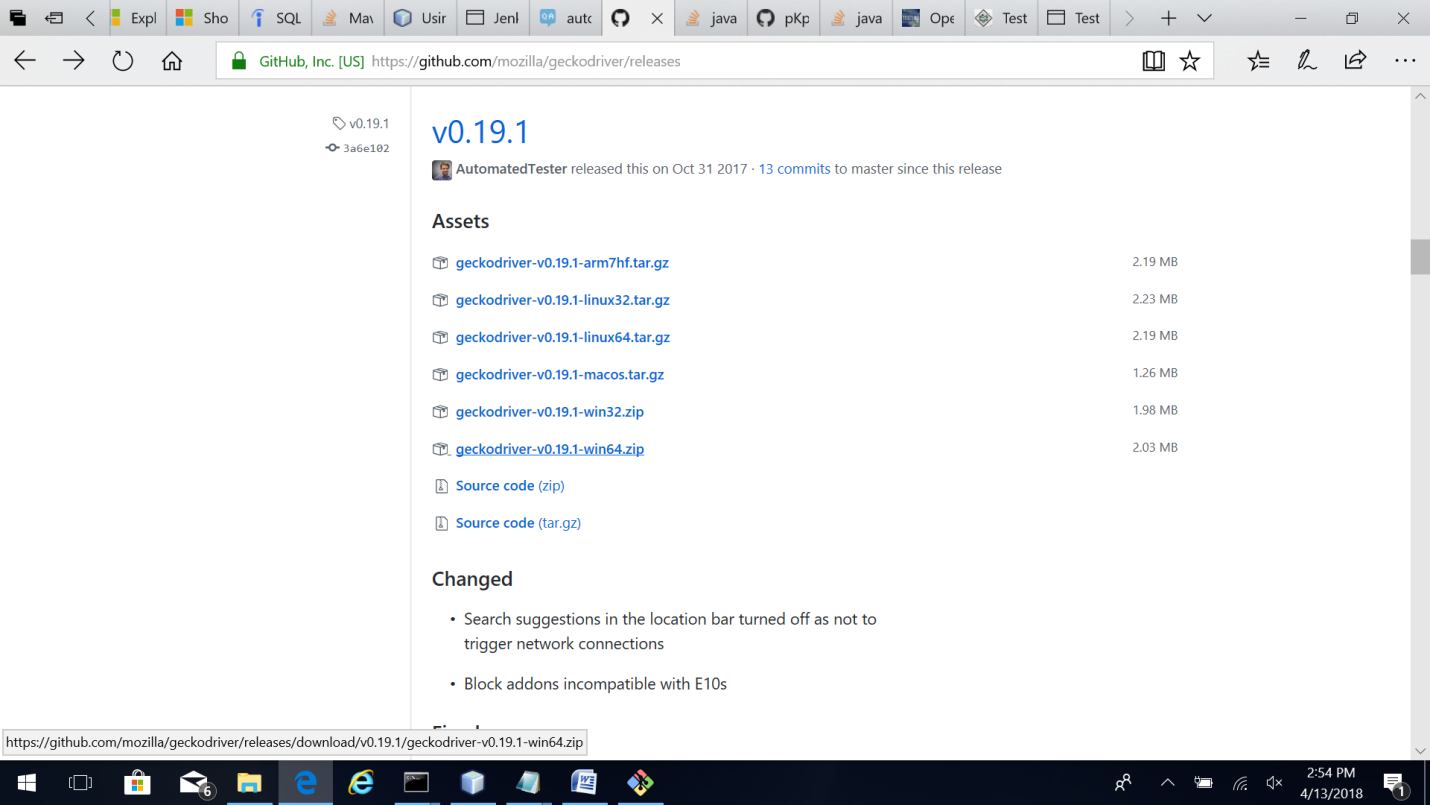
1. For Google Chrome use the following link:

<https://chromedriver.storage.googleapis.com/index.html?path=2.37/>



1. For Firefox Mozilla use the following link:

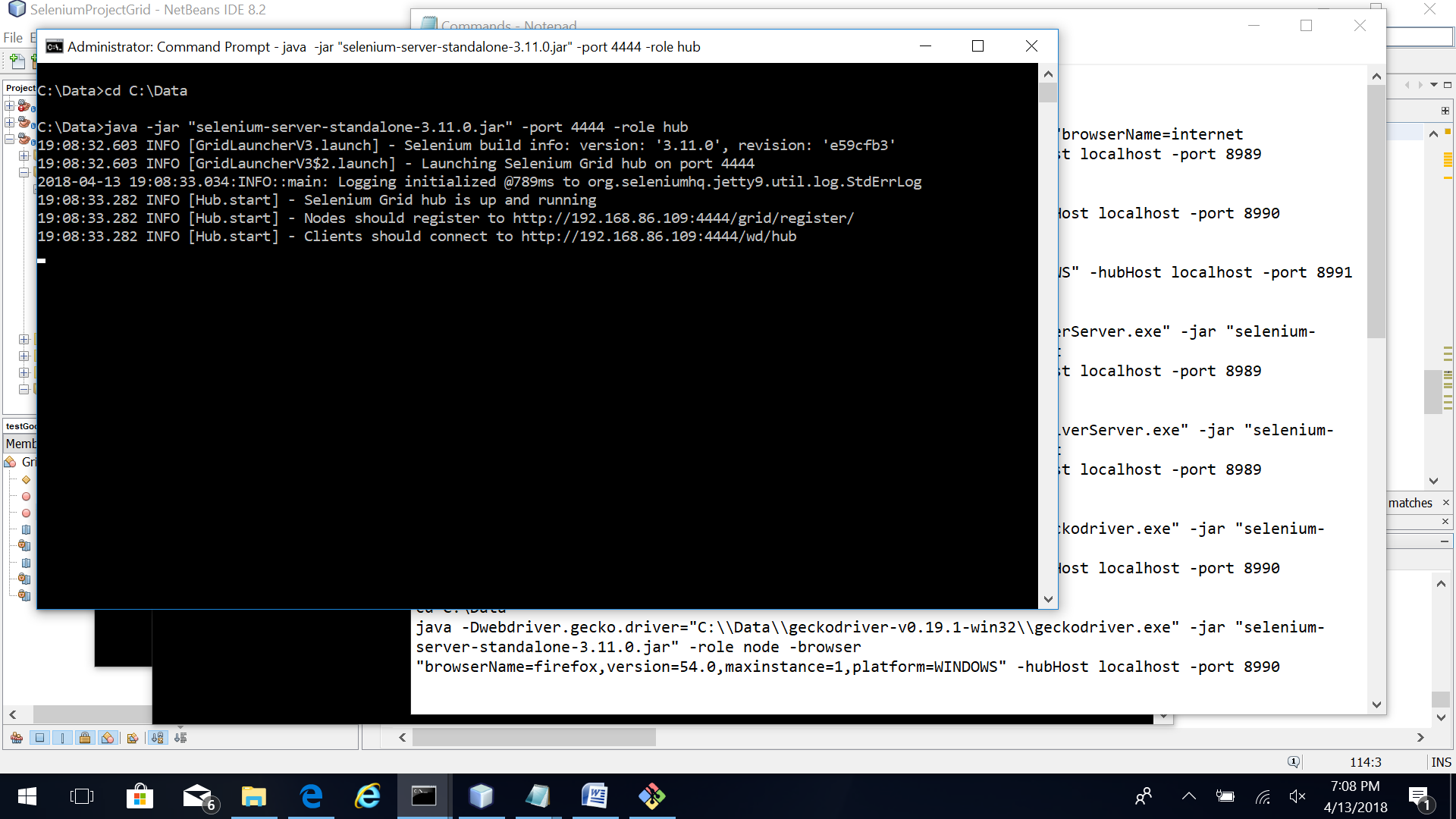
<https://github.com/mozilla/geckodriver/releases>



1. Extract all files in C:\Data directory.
2. Execute the following commands in sequence to start the hub and 3 nodes for Internet Explorer, Chrome and Firefox Browser capabilities respectively. We need to specify browser capabilities supported by the nodes such as Browser name, version, OS and maximum instances of requests supported and ip address of hub and port number used.

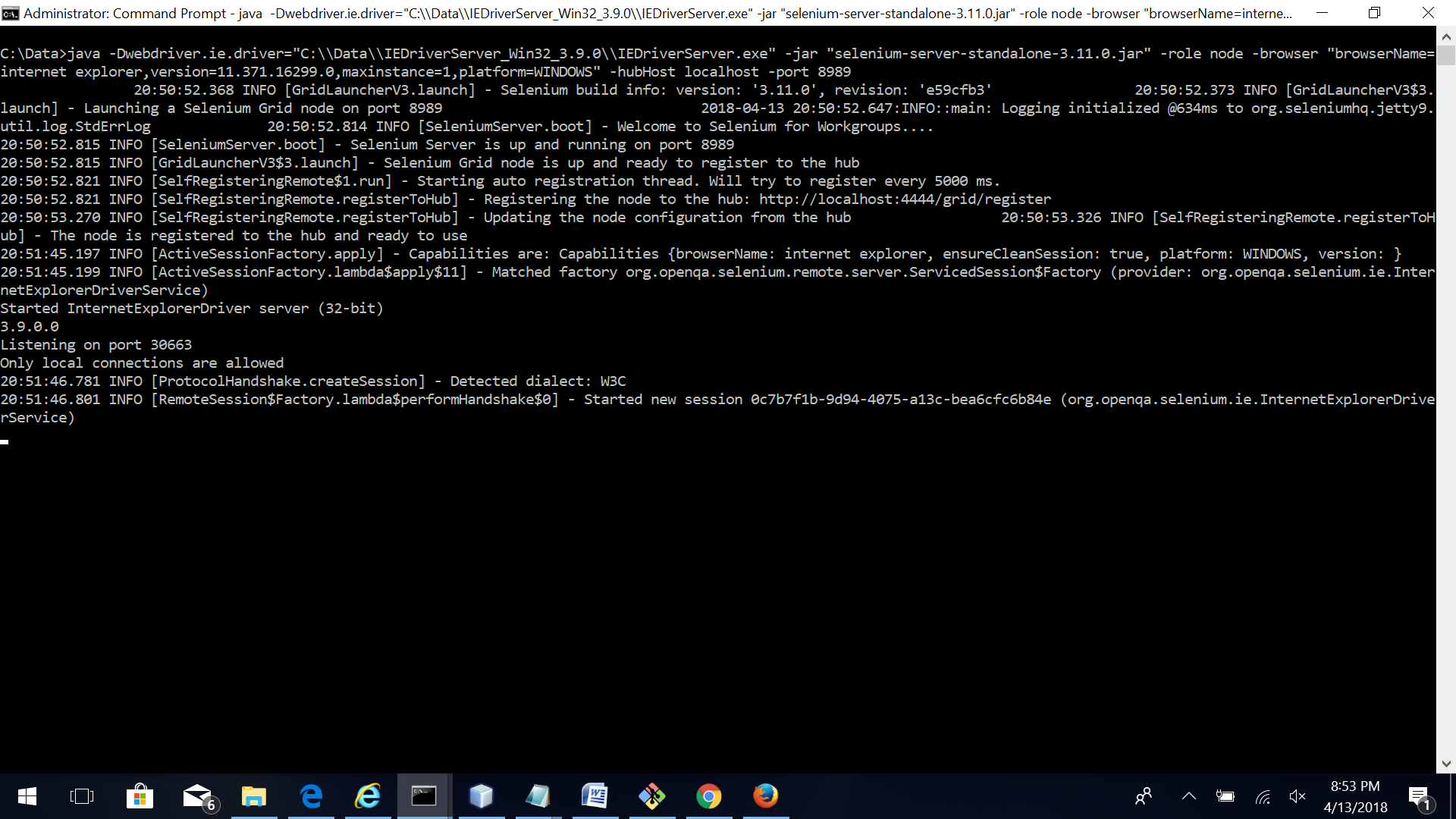
*cd C:\Data*

*java -jar "selenium-server-standalone-3.11.0.jar" -port 4444 -role hub*



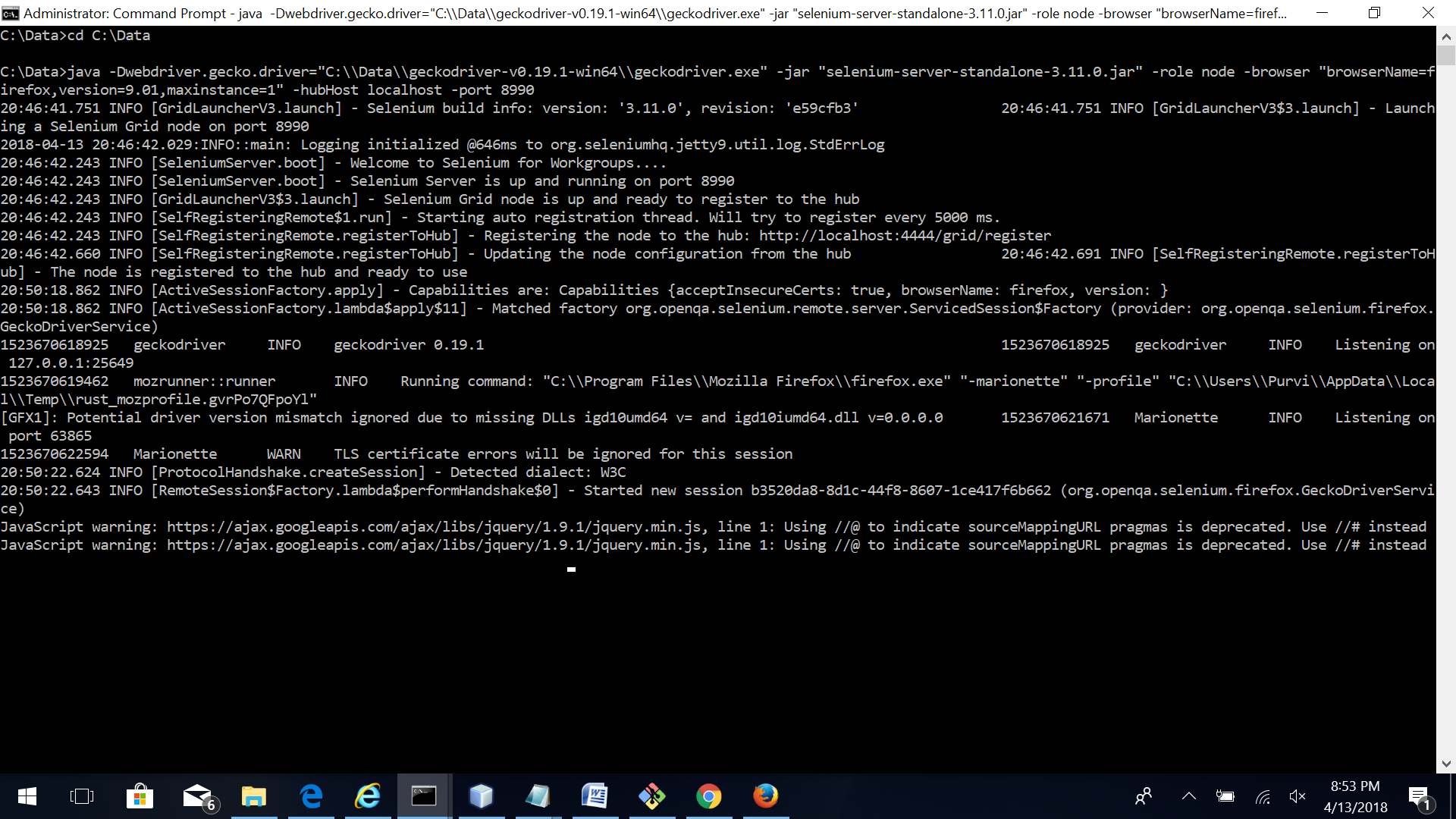
*cd C:\Data*

*java -Dwebdriver.ie.driver="C:\\Data\\IEDriverServer\_Win32\_3.9.0\\IEDriverServer.exe" -jar "selenium-server-standalone-3.11.0.jar" -role node -browser "browserName=internet explorer,version=11.371.16299.0,maxinstance=1,platform=WINDOWS" -hubHost localhost -port 8989*



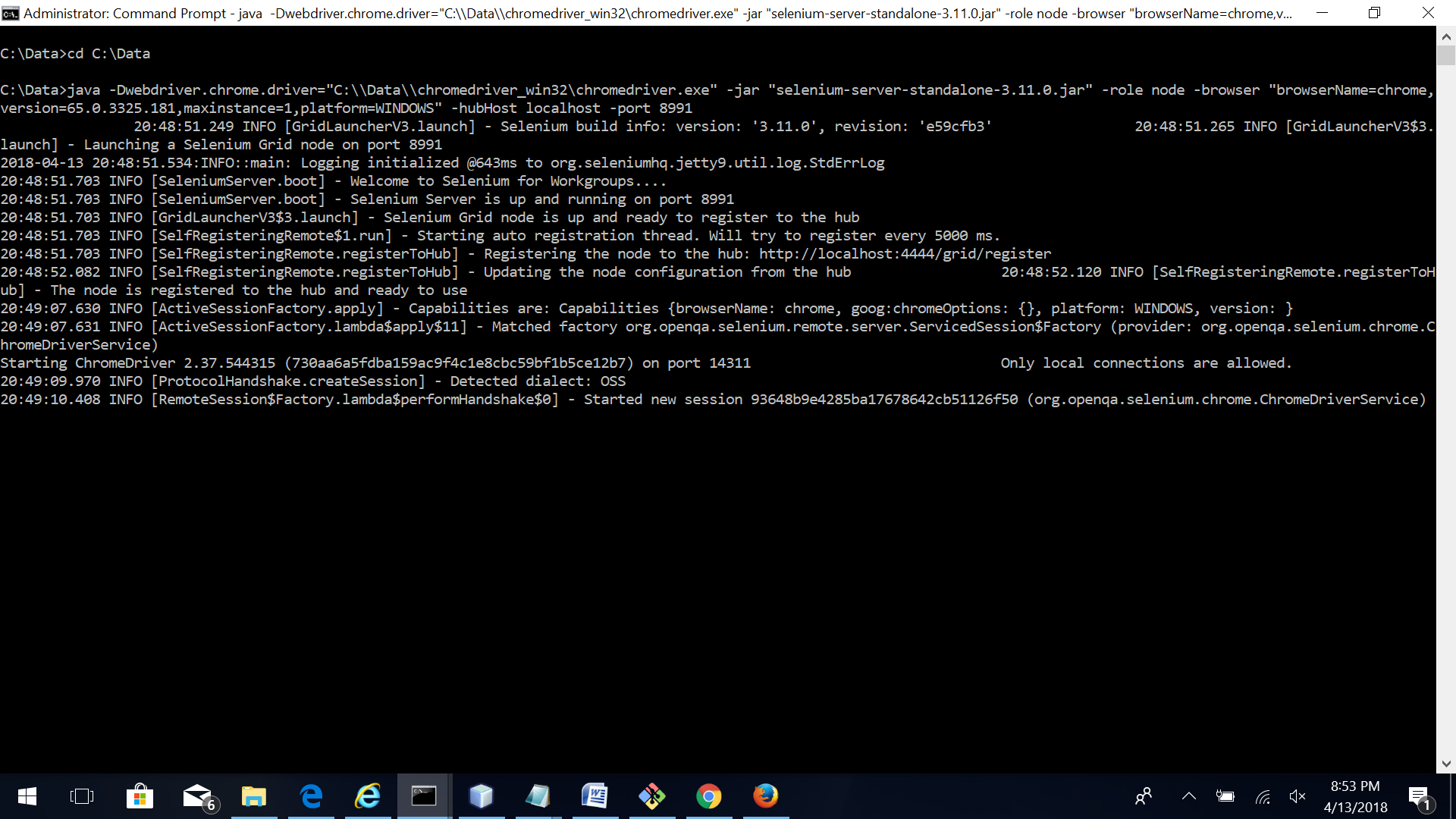
*cd C:\Data*

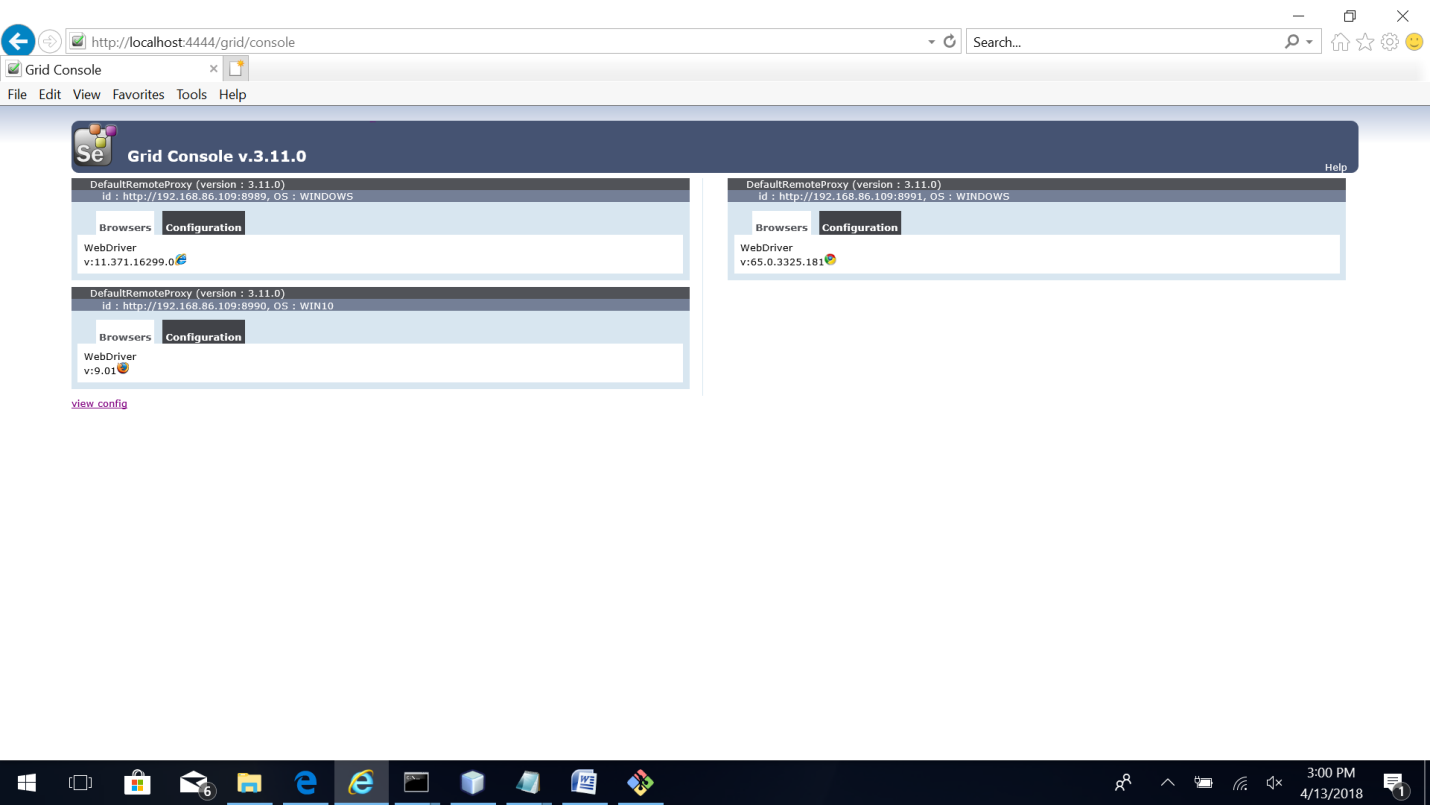
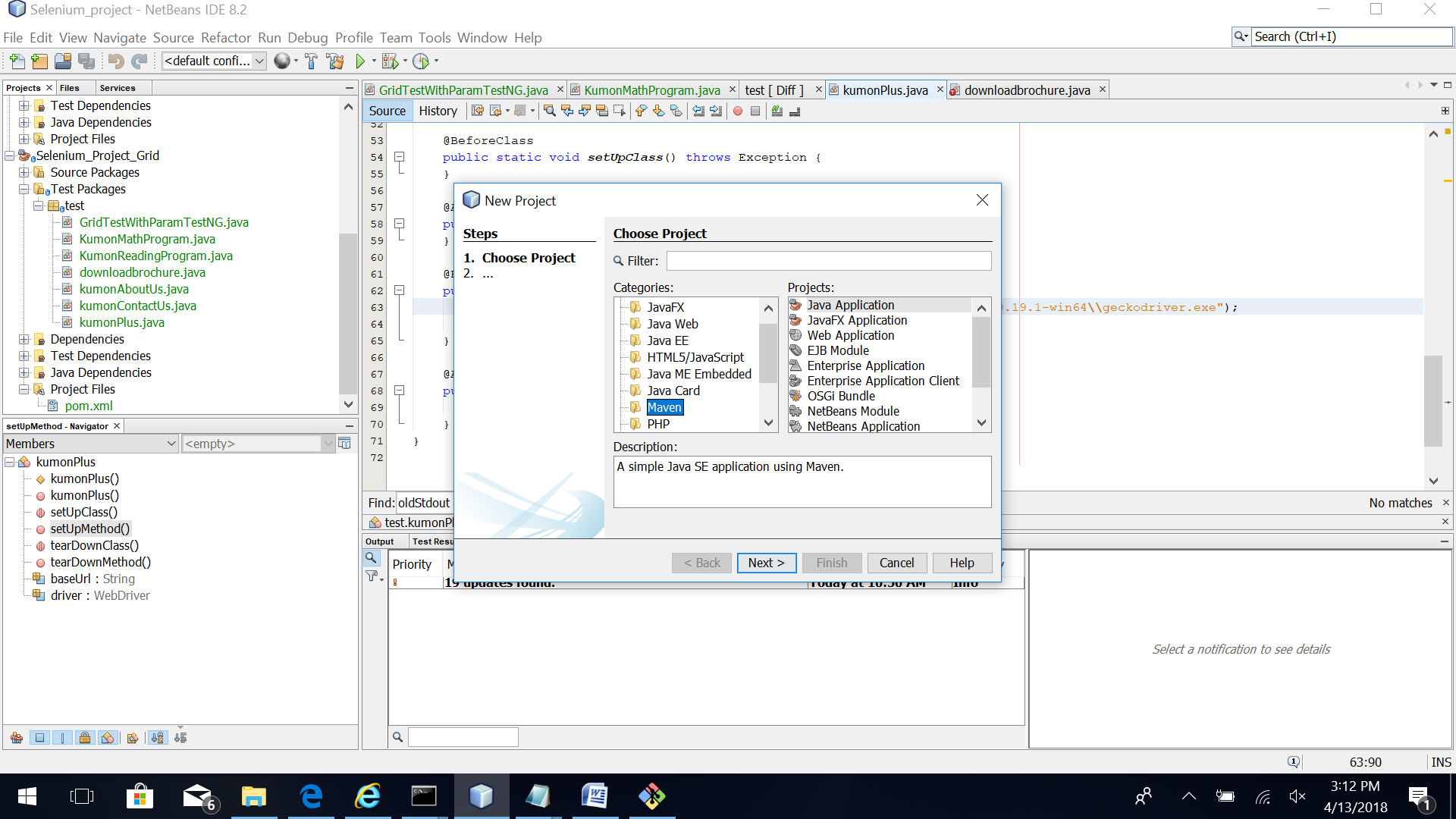
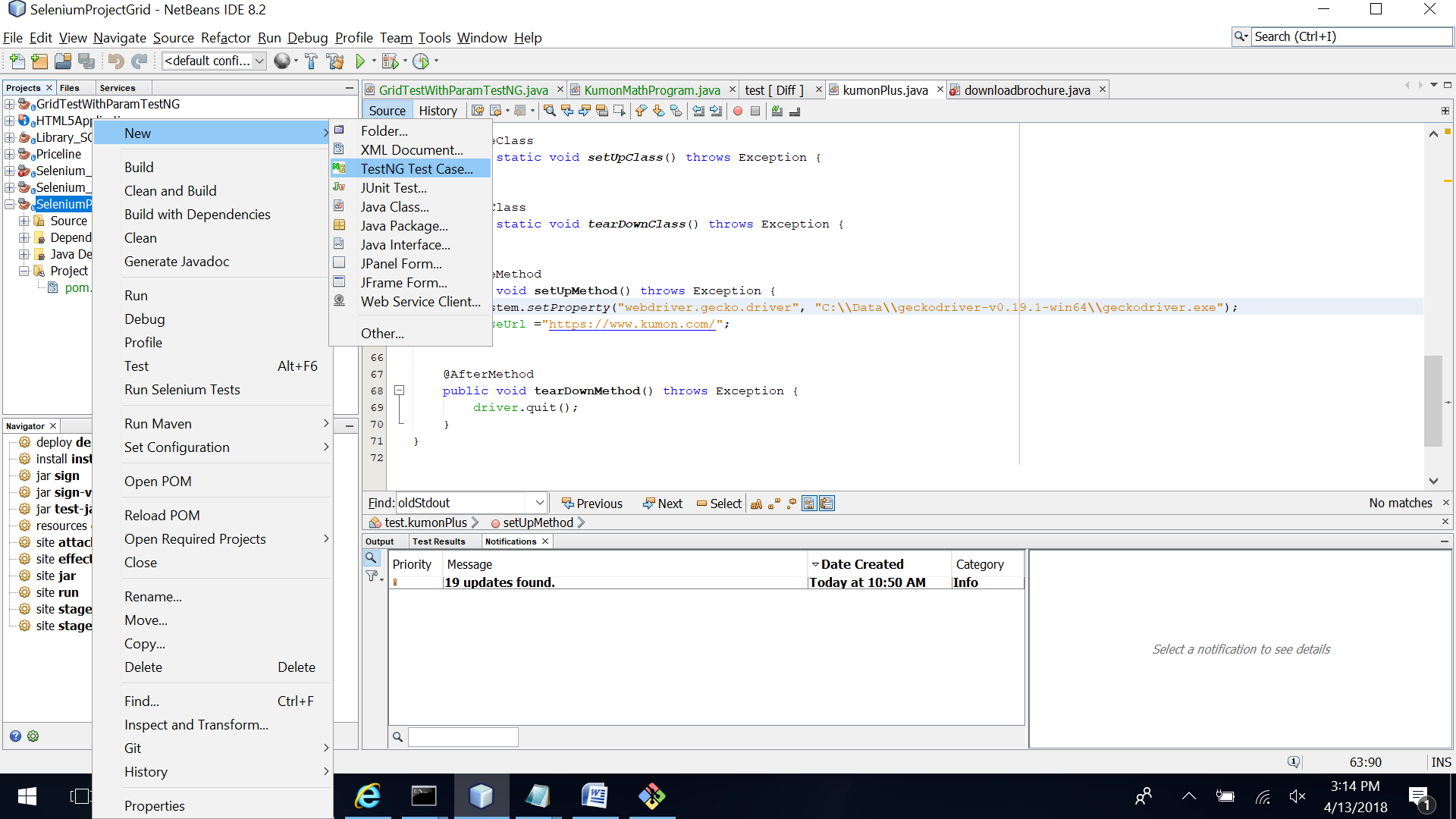
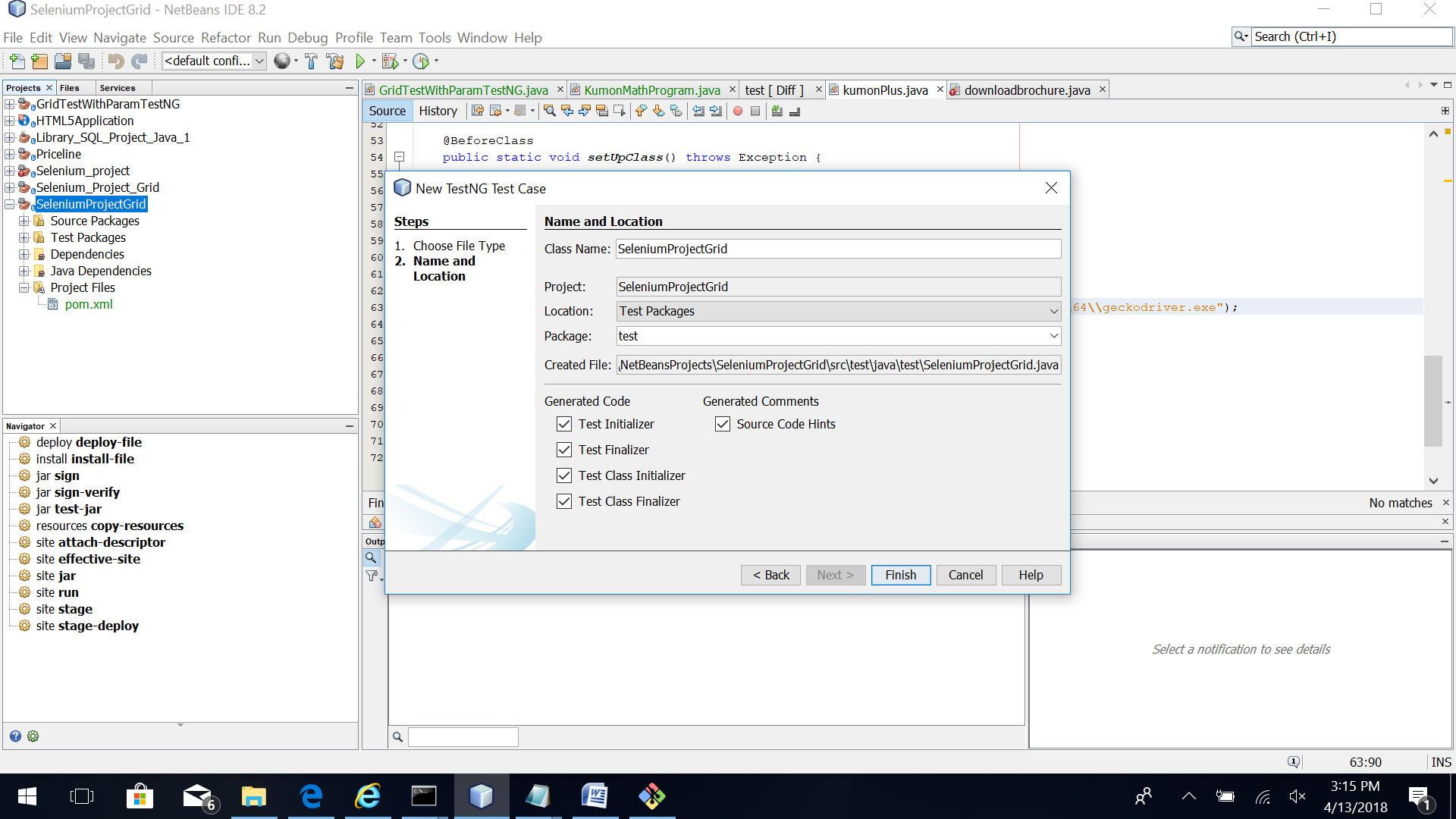
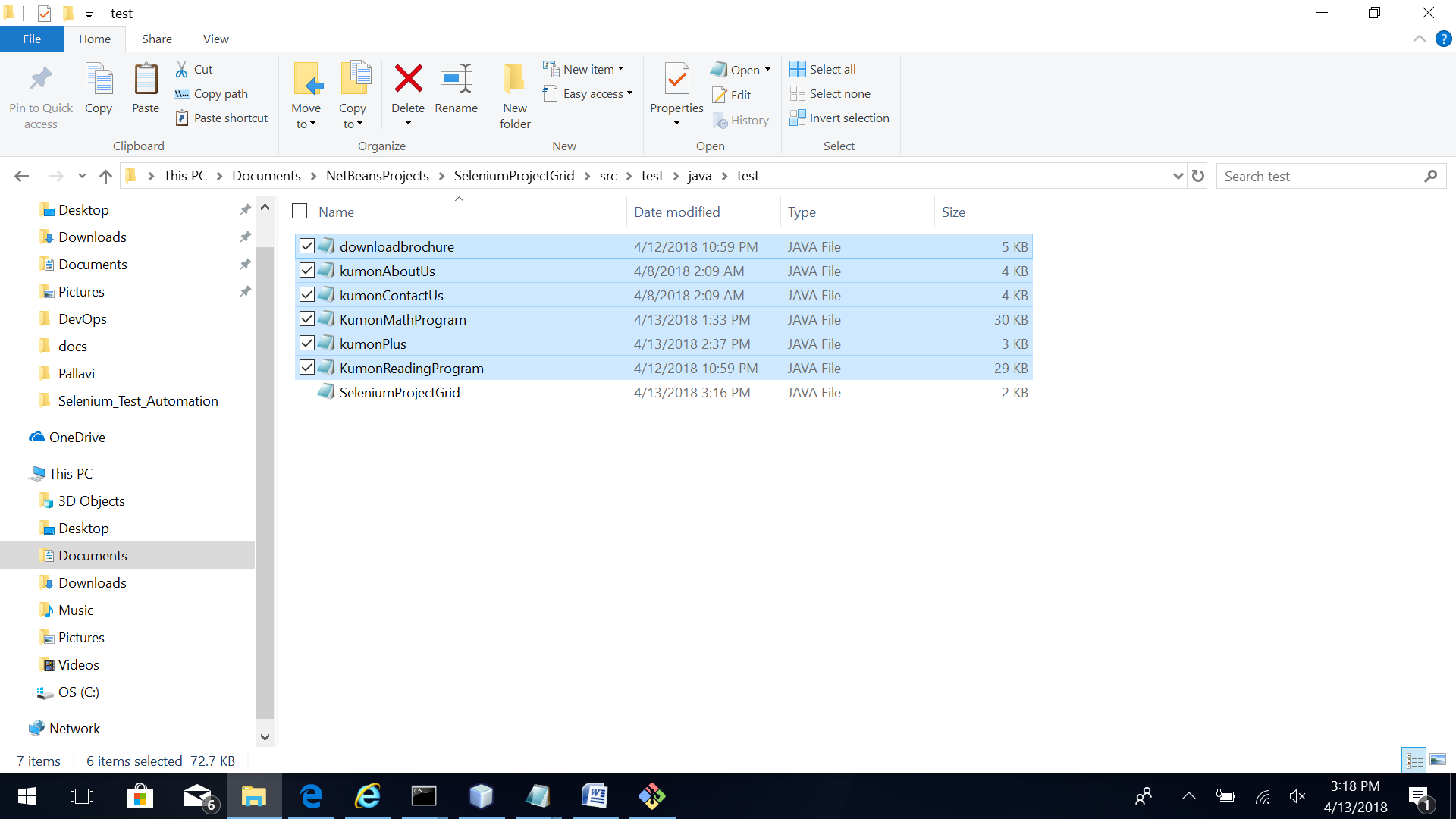
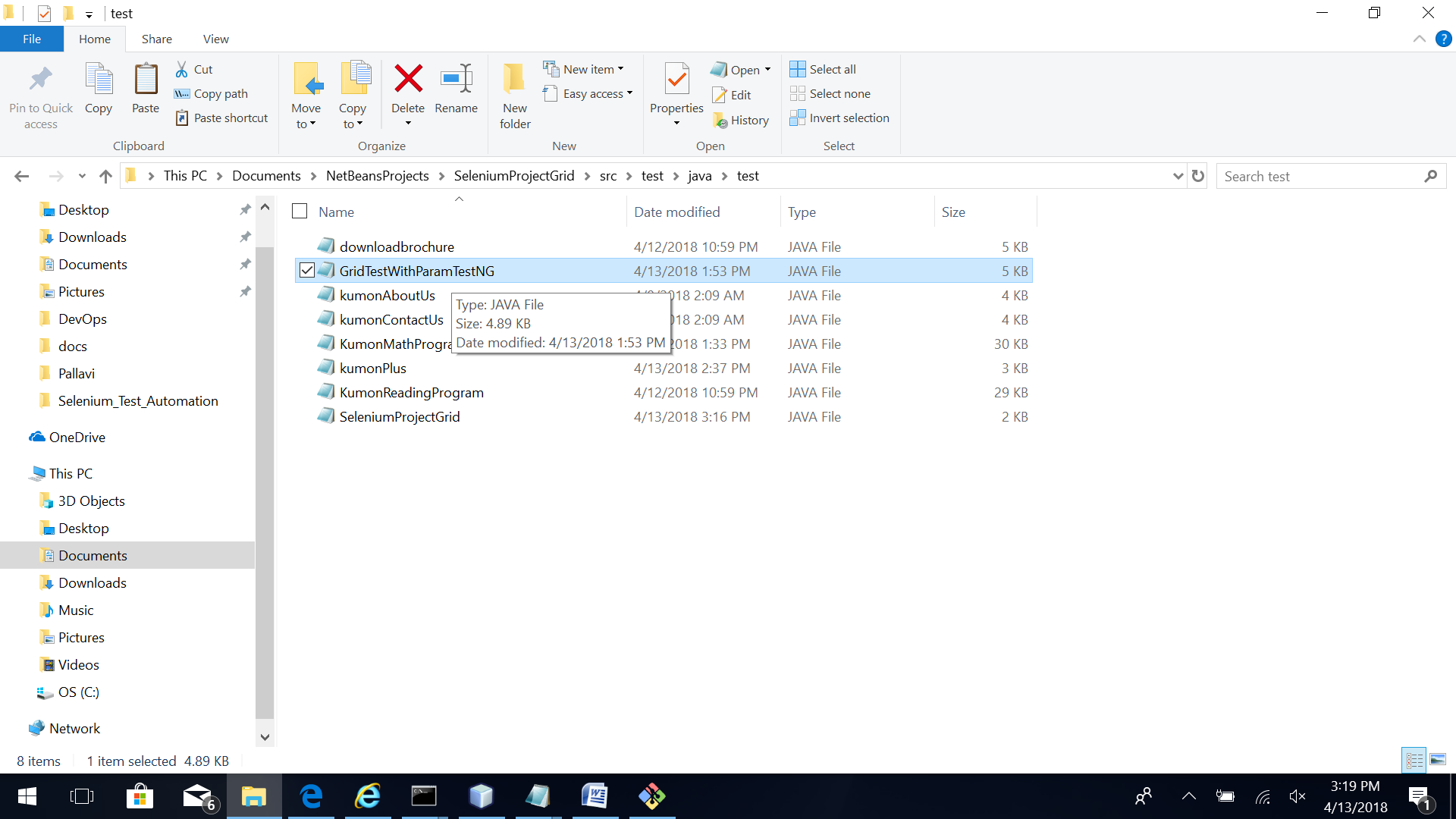
*java -Dwebdriver.gecko.driver="C:\\Data\\geckodriver-v0.19.1-win64\\geckodriver.exe" -jar "selenium-server-standalone-3.11.0.jar" -role node -browser "browserName=firefox,version=9.01,maxinstance=1" -hubHost localhost -port 8990*



*cd C:\Data*

*java -Dwebdriver.chrome.driver="C:\\Data\\chromedriver\_win32\chromedriver.exe" -jar "selenium-server-standalone-3.11.0.jar" -role node –browser "browserName=chrome,version=65.0.3325.181,maxinstance=1,platform=WINDOWS" -hubHost localhost -port 8991*



1. Configuration of the hub can be viewed on http://localhost:4444/grid/console 
2. Create a NetBeans IDE project SeleniumProjectGrid with Maven and Java.
3. Click to create new TestNG test case. Provide class name as “SeleniumProjectGrid” and package as ”test”. Click on Finish. 
4. Add the project test files in the test folder by copy pasting them. These files will be tested on the grid. Resolve the object dependencies by either importing header files or adding dependencies in the pom.xml file.
5. Include this file in the Selenium Grid test by copying it in the test folder - GridTestWithParamTestNG.java 
6. This file contains GridTestWithParamTestNG class and the DataProvider method is used for passing the test data to Selenium Grid RemoteWebDriver using TestNG interface. A DataProvider is a method annotated with *@DataProvider*. A DataProvider returns an array of objects.

@DataProvider(name = "data")

public Object[][] createData() {

return new Object[][] {

{"WINDOWS", "ie", "11.371.16299.0"},

//{"WINDOWS", "ie", ""},

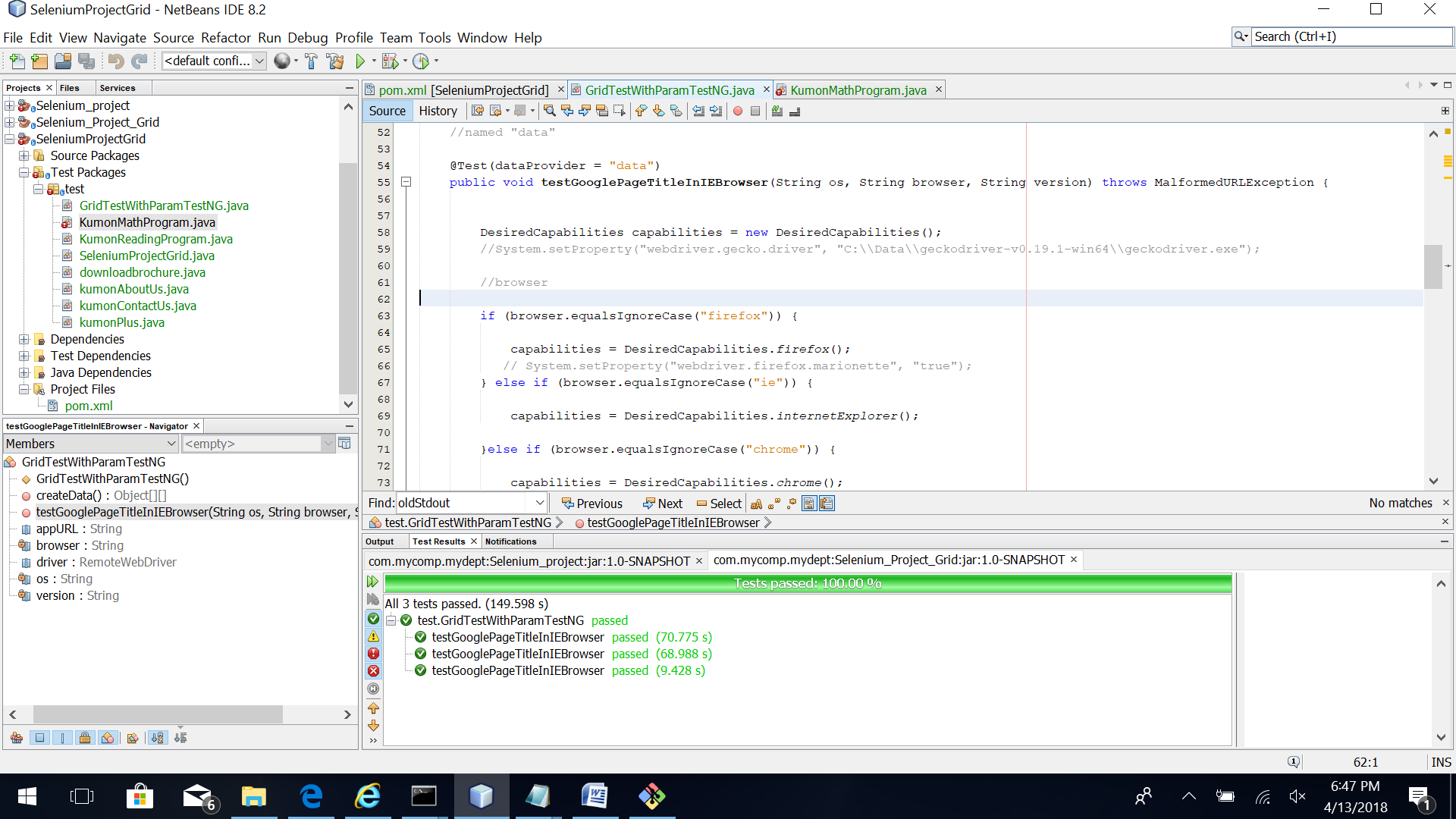
{"WINDOWS", "chrome", "65.0.3325.181"},

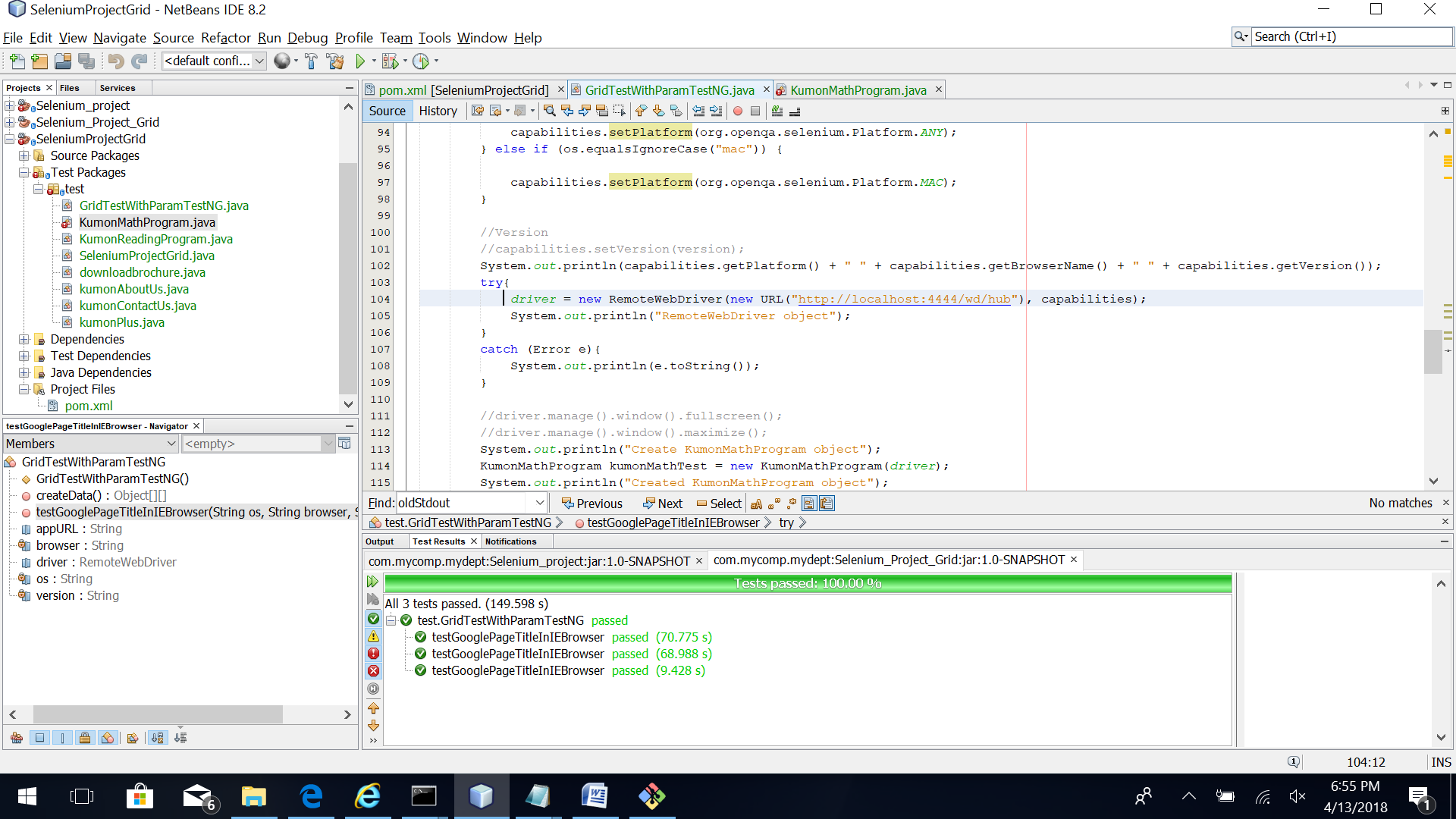
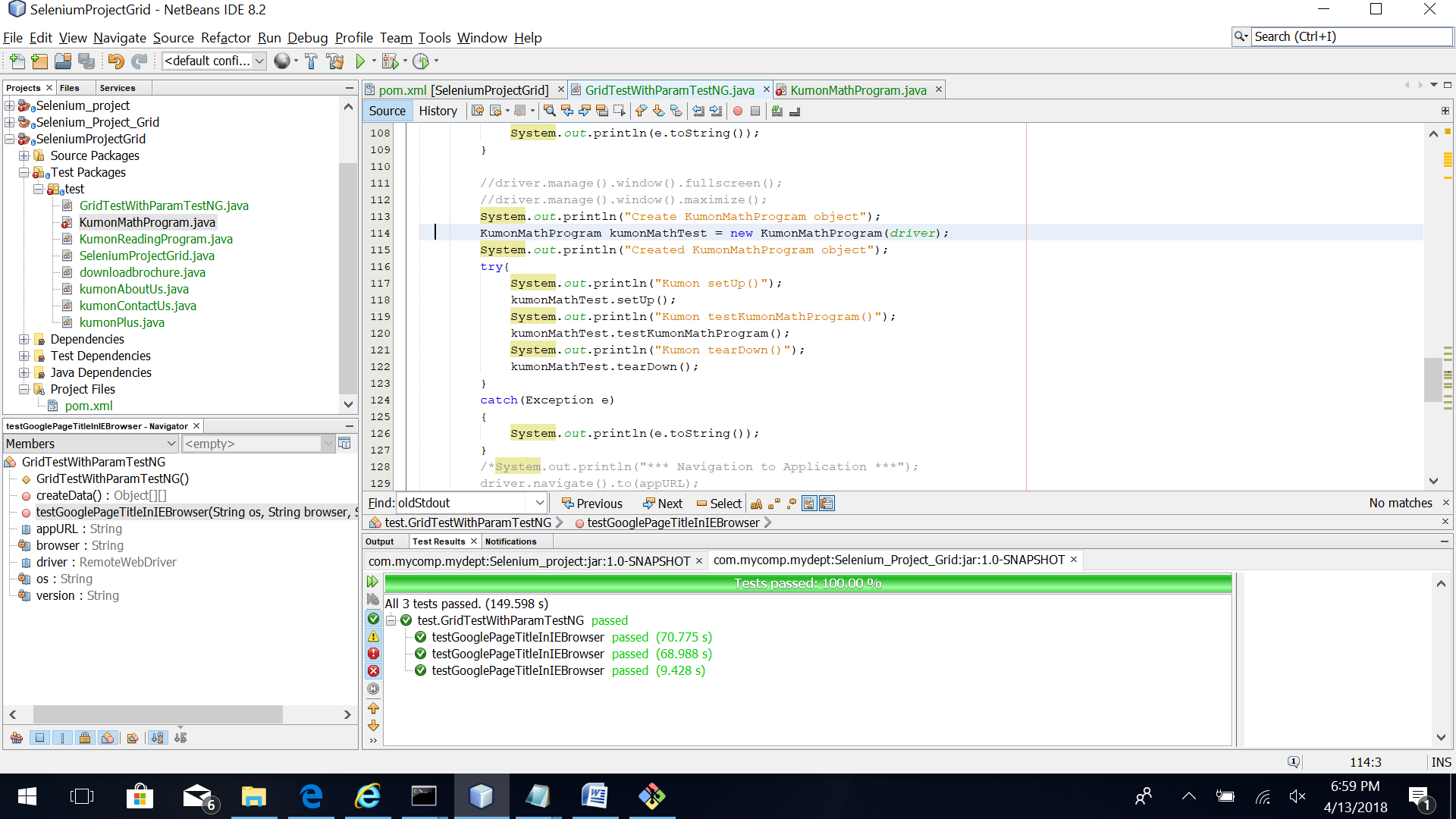
{"", "firefox", "9.0.1"}

};

};

1. The “data” created with @DataProvider method is used in test method of this class and specified before definition of the function as @Test(dataProvider = “data”). OS, browser and version are the parameters that get passed to the @Test function of this class.



1. The browser capabilities are set in DesiredCapabilities Java object and RemoteWebDriver object is created with the URL of Selenium Grid hub. http://localhost:4444/wd/hub
2. Objects of the classes to be tested are created. Then their methods are invoked in sequence to test the classes. Finally the driver.quit () method is called to close the instance of the driver created.
3. Right-click on the project explorer folder and click “Clean and Build”. Right click on “GridTestWithParamTestNG.java” file and select “Test File”. The @Test method of this file is invoked and the tests are ran on different nodes with required capabilities. 