# Java Interview Questions

### What is Java

* Java is a general-purpose programming language that is class-based object-oriented programming and designed to have as few implementation dependencies as possible
* It is intended to let the developer to “WRITE ONCE, RUN ANYWHERE”, meaning that complied Java code can be on any platforms that support Java without the need of recompilations.
* Java applications are typically complied to byte code that can run on any Java Virtual Machine regardless of underlying computer architecture.

### Is Java Object Oriented Programming Language?

* Java is not 100% OOPS language

# Selenium Interview Questions

### What is the difference between driver.get() and driver.manage().navigateTo() methods

* Both works same when you use it for the first time, page getting refreshed when we use them again.
* driver.get() is used to navigate to particular URL and waits till the page load. It does not maintain page history or cookies
* driver.manage().navigateTo() is used to navigate to a particular URL and does not wait for the page to load. It maintains page history and cookies and help us to navigate forward and backward.

### Action Class

#### *What is Action class*

* Action class is built-in ability and user facing API to handle various type of Keyboard and Mouse events.
* Use can directly use this class rather than using input devices such as Keyboard and Mouse.
* In Selenium Webdriver, handling these events including operations such as drag and drop, clicking on multiple elements with the help of control key can be done using this API.

#### Different methods in Actions class

* Keyboard interface methods
* Mouse interface methods

#### How to handle action class in Selenium

* Create an object of the Actions class ‘action‘
* Focus on the element using WebDriver: action.moveToElement(element).build().perform();
* Build().perform() is used to compile and execute the actions class.
* Use the different methods under the actions class to perform various operations like click(), drag and drop and so on.

#### Keyboard interface methods

* **sendKeys(keysToSend):** Sends a series of keystrokes onto the element.
* **keyDown(theKey)**: Sends a key press without release it. Subsequent actions may assume it as pressed.
* Example: Keys.ALT, Keys.SHIFT, or Keys.CONTROL);
* **keyUp(theKey)**: Performs a key release.

Mouse Interface methods

* **click():** Clicks on the element.
* **doubleClick ():** Double clicks on the element.
* **contextClick() :** Performs a context-click (right-click) on the element.
* **clickAndHold():** Clicks at the present mouse location without releasing.
* **dragAndDrop(source, target):** Clicks at the source location and moves to the location of the target element before releasing the mouse. source (element to grab, target – element to release).
* **dragAndDropBy(source, xOffset, yOffset):** Performs click-and-hold at the source location, shifts by a given offset value, then frees the mouse. (X offset – to shift horizontally, Y Offset – to shift vertically).
* **moveByOffset(x-offset, y-offset):** Shifts the mouse from its current position (or 0,0) by the given offset. x-offset – Sets the horizontal offset (negative value – shifting the mouse to the left), y-offset – Sets the vertical offset (negative value – shifting the mouse to the up).
* **moveToElement(toElement):** It shifts the mouse to the center of the element.
* release(): Releases the pressed left mouse button at the existing mouse location.

### How to do mouse hover in Selenium

* Mouse hover action is basically an action where a user places mouse over a designated area link hyper link. It can cause some events to be triggered
* Methods provided by Actions class to do mouse hover action

1. moveToElement( WebElement target)
2. moveToElement( WebElement target, int xOffset, int yOffset)

* Coding
  + //Instantiate Action class
  + Actions action = new Actions(driver);
  + // Find the element where we need to do mouse hover
  + WebElement target = driver.findElement(locator);
  + // invoke moveToElement method
  + action.moveToElement(target).perform();
* What really happens
  + **Move Mouse Action:** Mouse gets moved to the middle of the element.
  + Here, the element is scrolled into view and its location gets calculated using getBoundingClientRect.
  + **Build:** build() method is used to generate a composite action containing all actions.
  + In this case, it’s just a single move mouse action. If you observe, we have not invoked it in our above command. This is because the build is executed in the perform method internally.
  + **Perform:** perform() method performs the actions we have specified. But before that, it internally invokes build() method first. After the build, the action is performed.

### How do you take screenshot in Selenium Webdriver?

* Typecast WebDriver instance to Take Screen shot.
  + TakeScreenshot shot = ((TakeScreenshot)WebDriver);
* Call getScreenshotAs method
  + File srcFile = shot.getScreenshotAs(OutputType.FILE);
* Copy file to designated path
  + FileUtility.copyFile (srcFile, new File(path)

The above code will take screenshot of viewable screen

### How do you take screenshot of whole/ full page

* getScreenshotAs() method will take screen shot of the viewable screen only.
* In order to take screenshot of entire page using Selenium WebDriver, we can make use of Ashot().

**Ashot()**

* It is a web driver screenshot utility to capture the entire page of screenshot and it is natively supported from Selenium 3 onwards
* It provides following features
  + It helps capture the entire screen and web element.
  + Beautify screenshot.
  + Provides screenshot comparison.
* Coding

Screenshot screenshot=new AShot().shootingStrategy(ShootingStrategies.viewportPasting(1000)).takeScreenshot(driver);

ImageIO.write(screenshot.getImage(),"PNG",new File("path of the file"));

### How to scroll to an element which don’t have any xpath or any other locator

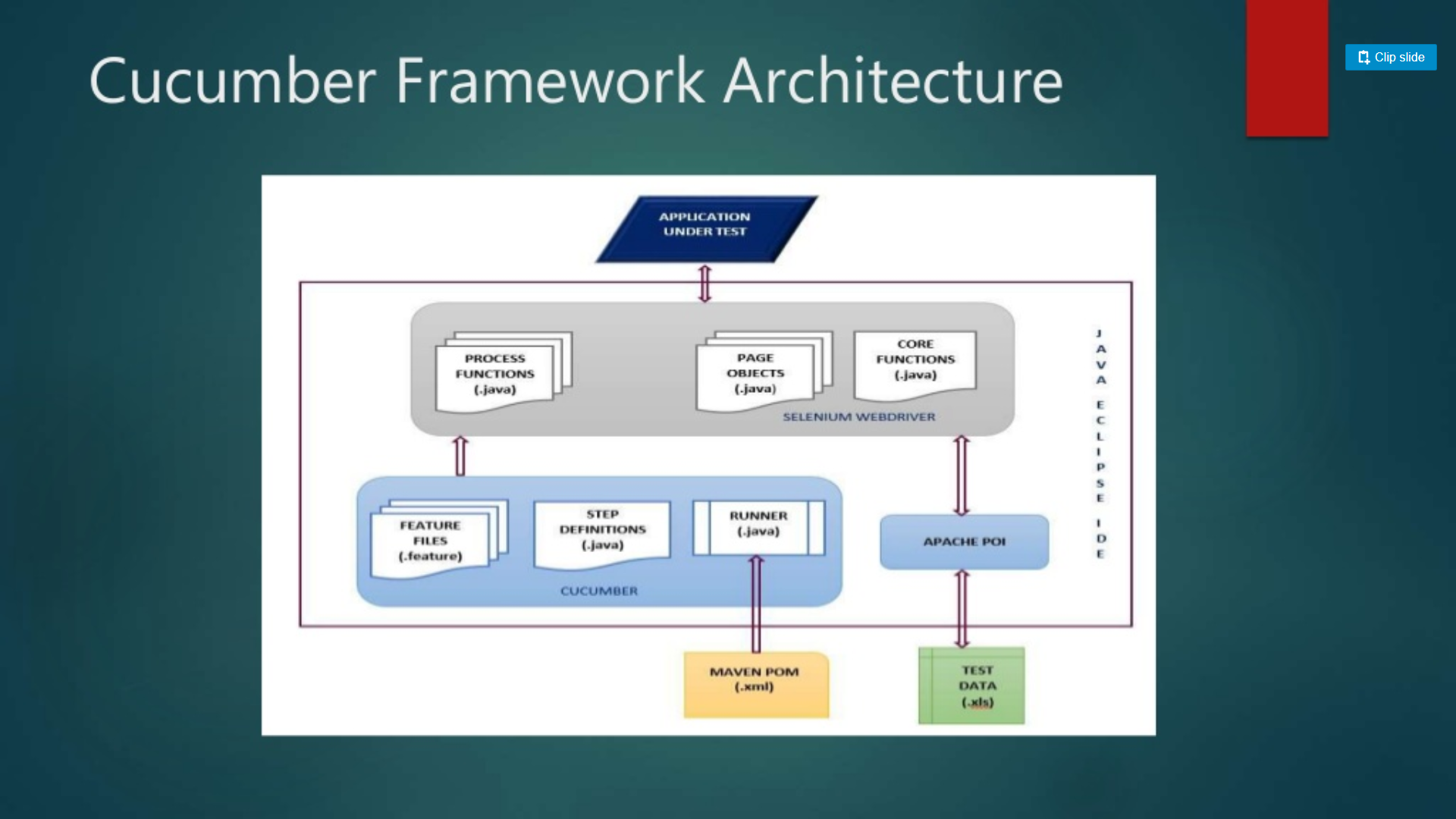
### Why do we use Cucumber Framework

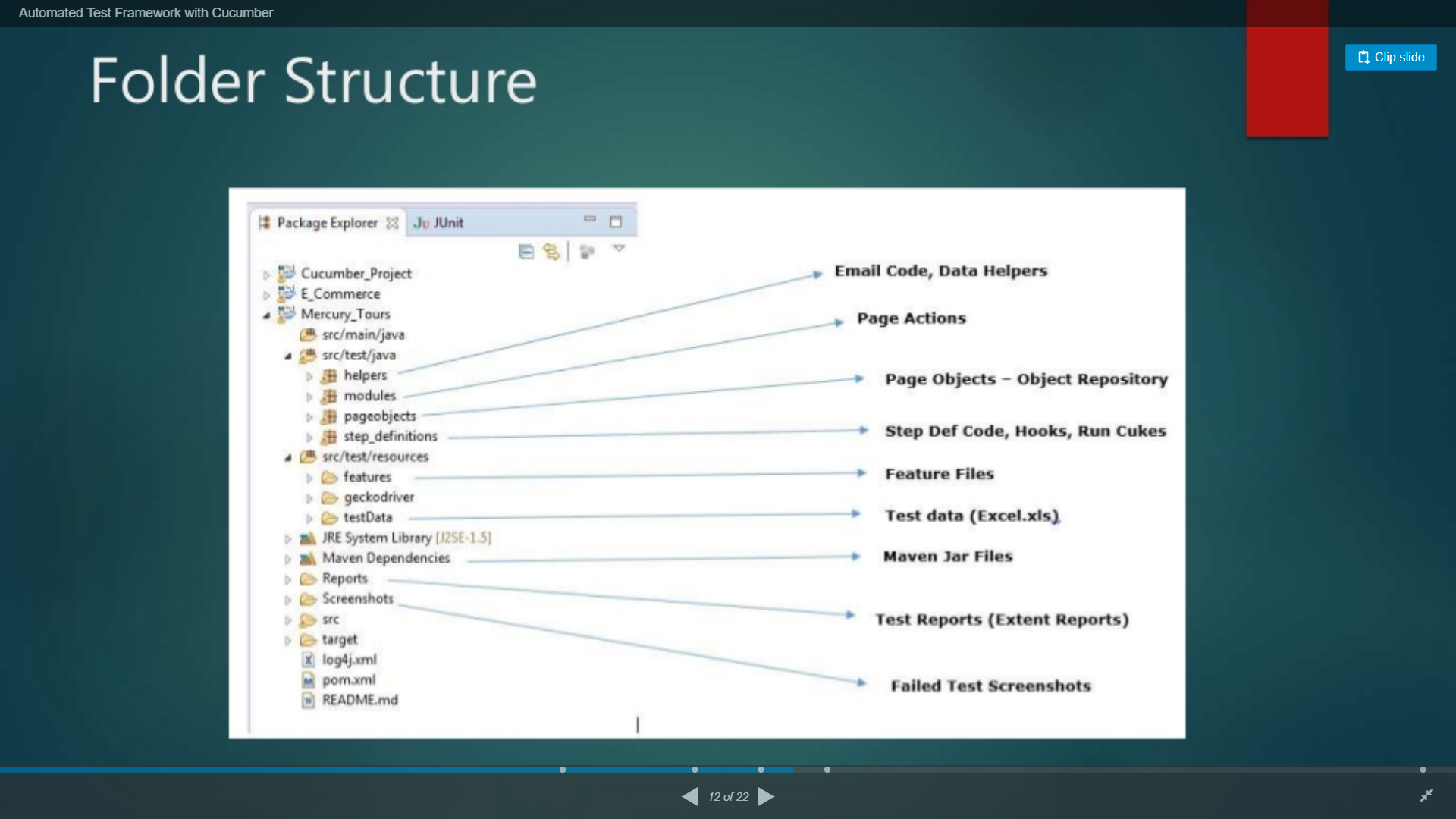
#### What is Cucumber

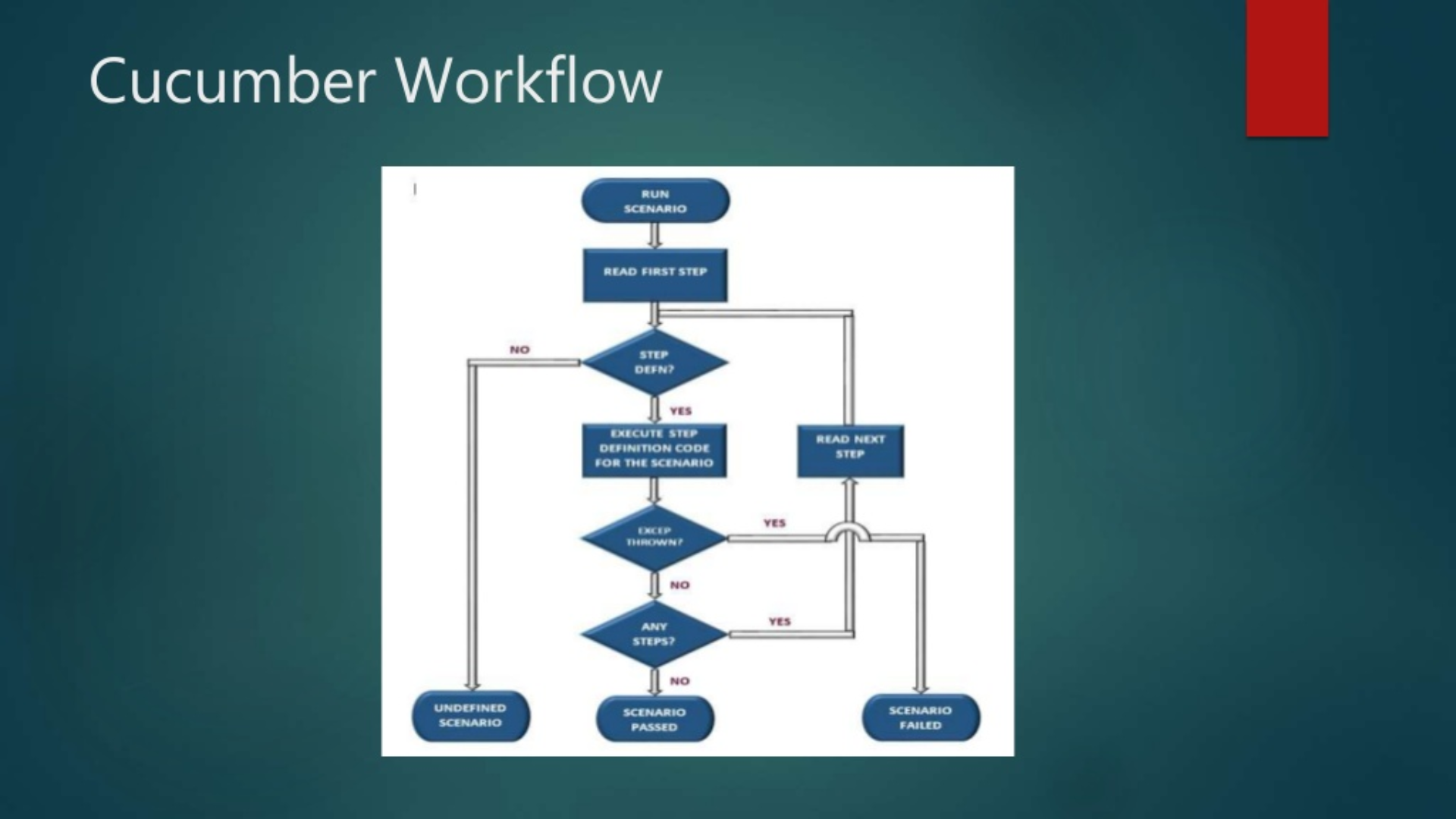
* Cucumber is a tool based on Behaviour Driven Development framework (BDD) which is used to write acceptance tests for web application.
* It allows automation of functional validation in easily readable and understandable format called Gherkin to business Analyst, Dev and testers etc.
* Other tool like Cucumber is JBehave.
* Refer the below link
  + <https://www.softwaretestinghelp.com/cucumber-bdd-tool-selenium-tutorial-30/>

### What is the structure of Cucumber

* Reference link
  + <https://www.slideshare.net/RameshKrishnan35/automated-test-framework-with-cucumber>







### How runner file is connected to other files in Cucumber

* Reference link
  + <http://www.automationtestinghub.com/cucumber-test-runner-class-junit/>

#### What is Cucumber Test Runner Class

* Cucumber test runner class is one of many mechanism using which we can run a Cucumber feature file
* Test runner class also act as a interlink between feature file and step definition class by providing path for both feature file and step definition class.
* There are multiple types of test runners such as Junit runner, Android runner etc., used to run the feature file
* With the test runner class you have option to run either single feature file or multiple feature file
* Code

@RunWith(Cucumber.class)

@CucumberOptions(features="resources/features", glue="")

public class TestRunner\_GoogleHomepage {

}

* @RunWith annotation: This is a JUnit annotation that specifies which runner it has to use to execute this class.
* @CucumberOptions annotation: This annotation provides some important information which will be used to run your cucumber feature file. At the very least, Java should know the location of the feature file as well as the step definition class in your project. CucumberOptions annotation will do just that. There are two parameters
  + The first parameter, called features, provides the location of the feature file.
  + Similarly, the second parameter, called glue, provides the path of the step definition class.

### How to Handle Alerts in Selenium

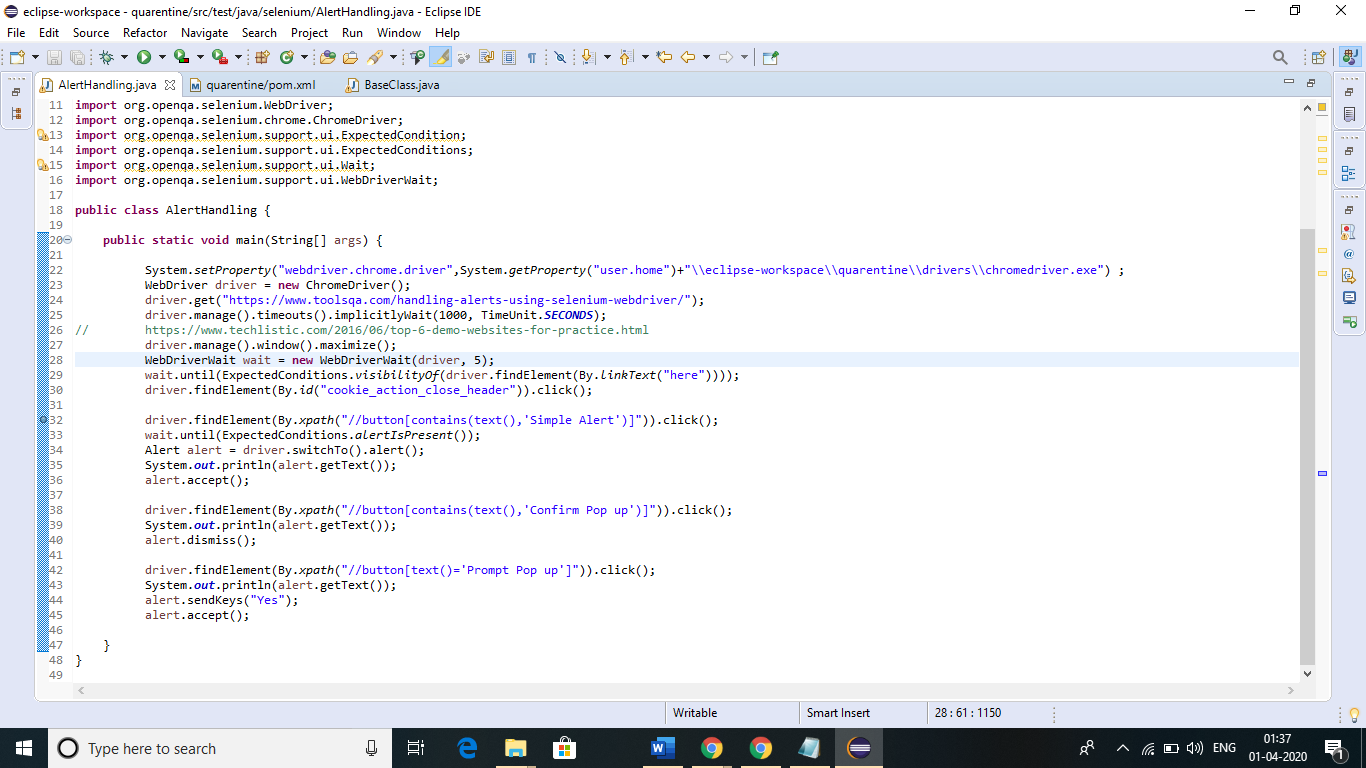
#### What is alert in Selenium

* Alert is small message box which displays on-screen notification
* It gives the user some kind of information or will get permission to perform certain kind of operation
* Sometimes it is used to give warning notification

#### Types of Alerts

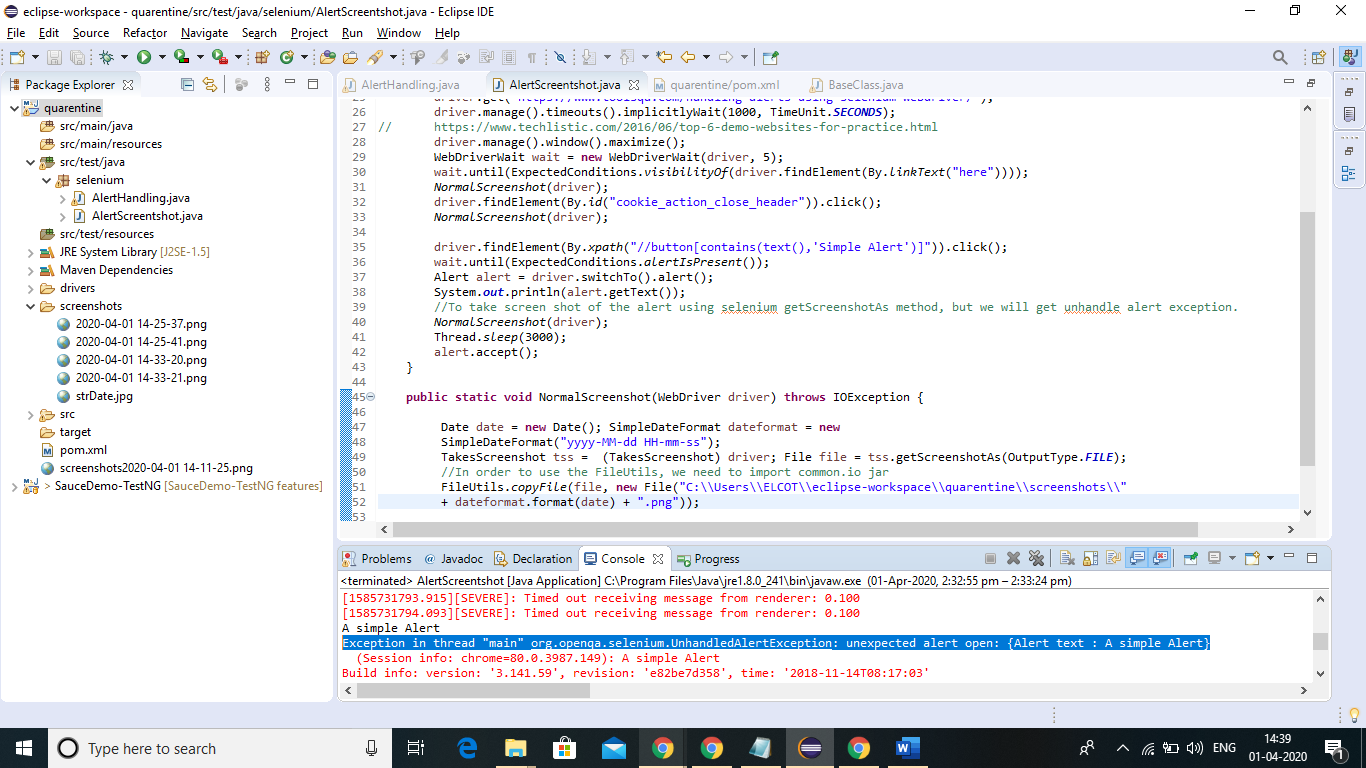
* Simple Alert
* Prompt Alert – Ask some information from user
* Confirmation Alert

#### Coding



#### What happens when you take screen shot of an alert using Selenium getScreenshotAs() method

We will get “Exception in thread "main" org.openqa.selenium.UnhandledAlertException: unexpected alert open: {Alert text : A simple Alert}”



### How to take screenshot of an alert?

#### Steps to capture screenshot using robot class:

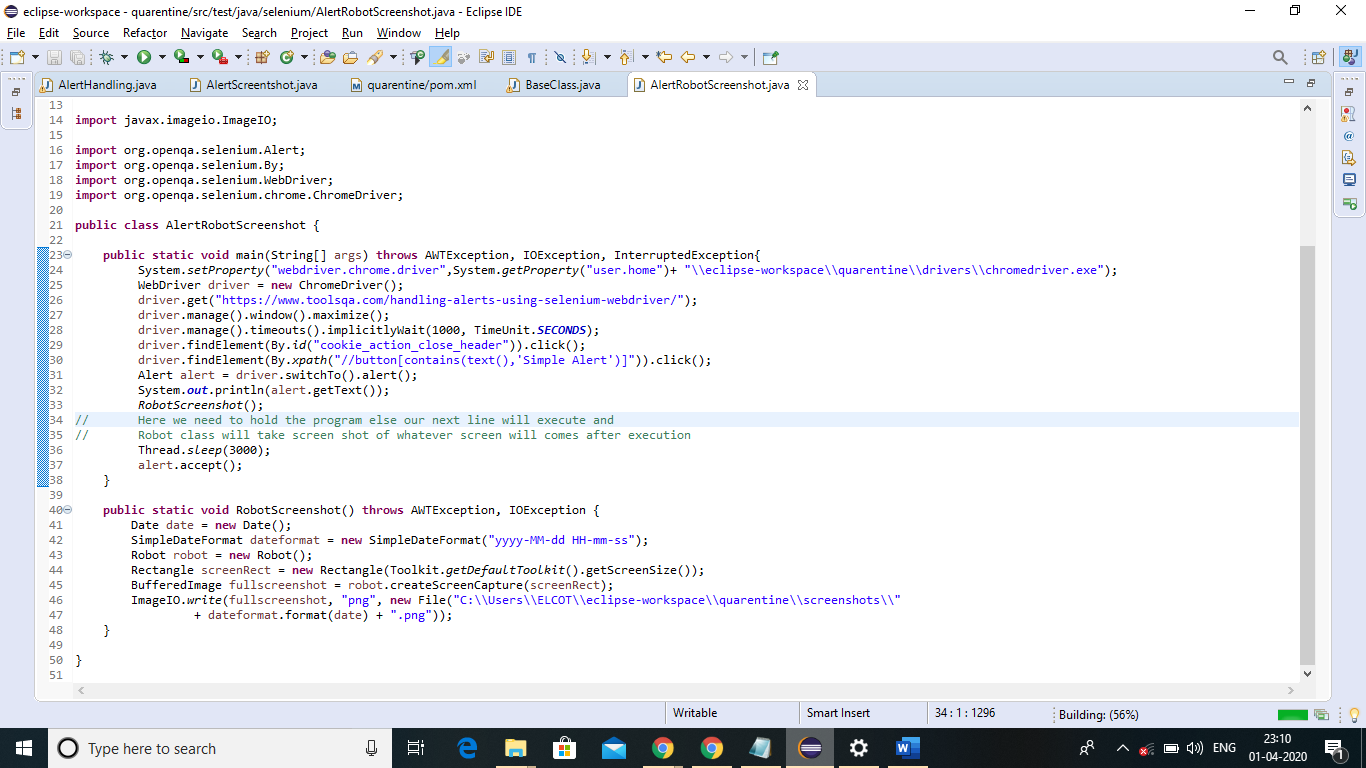
Create an object of Robot class.

Get the screen size as a Rectangle.

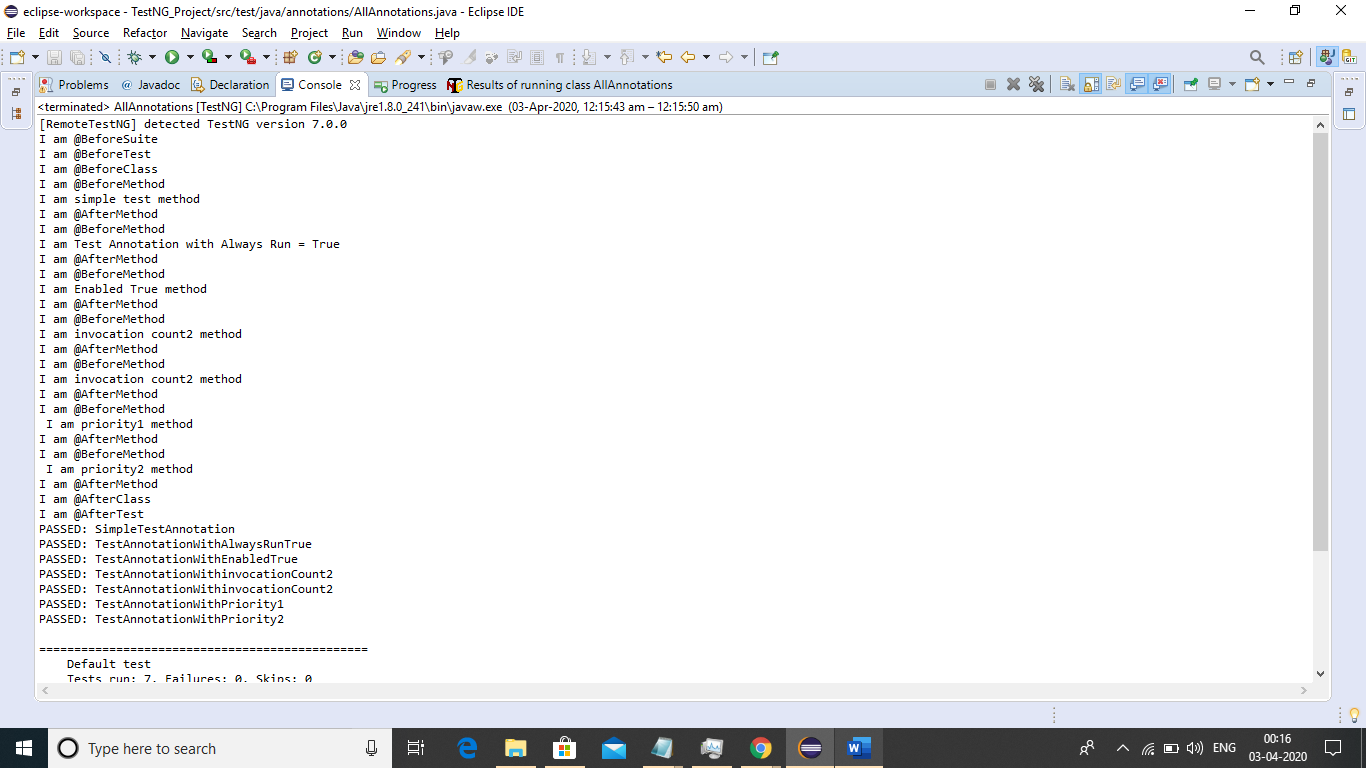
Use createScreenCapture of Robot class and capture screenshot. It will be temp image.

Define destination path for screenshot.

Write temp file in to permanent file.



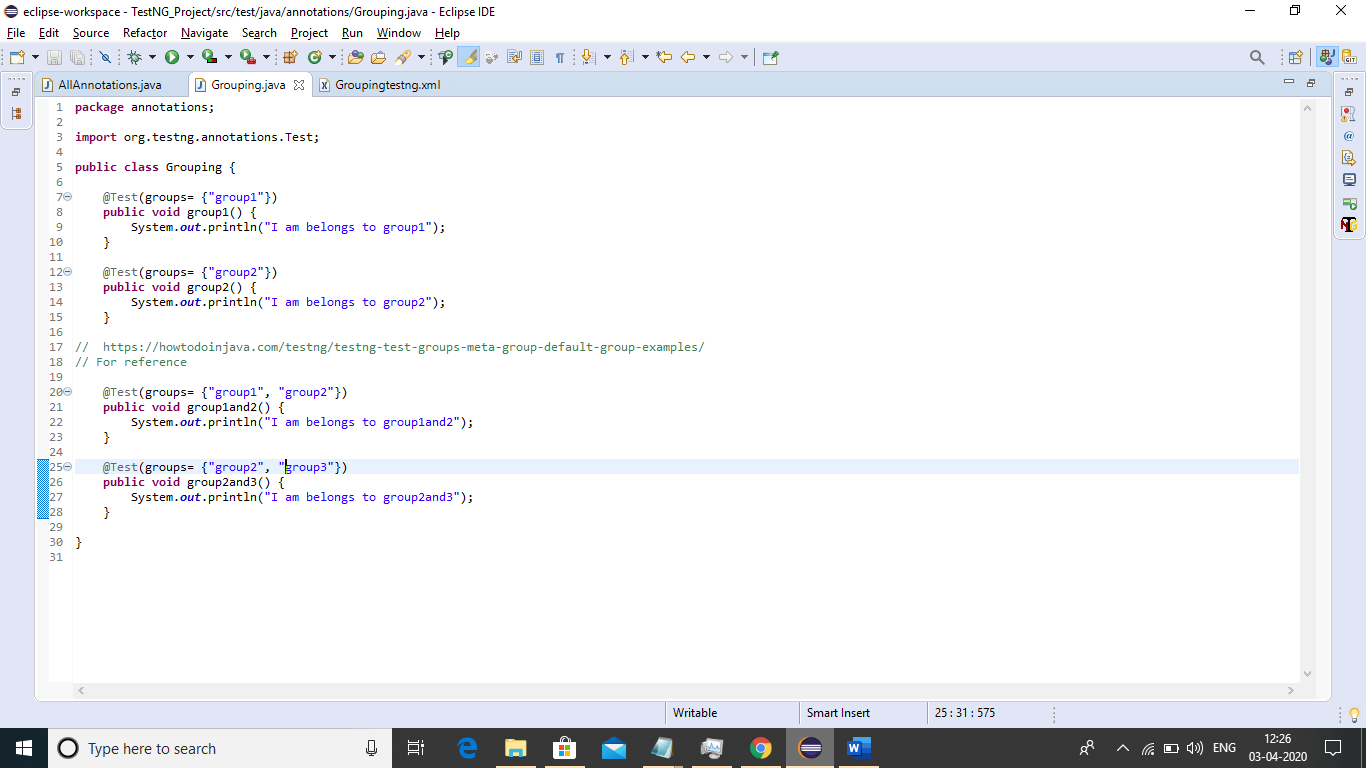
### In which sequence TestNG annotations will run?



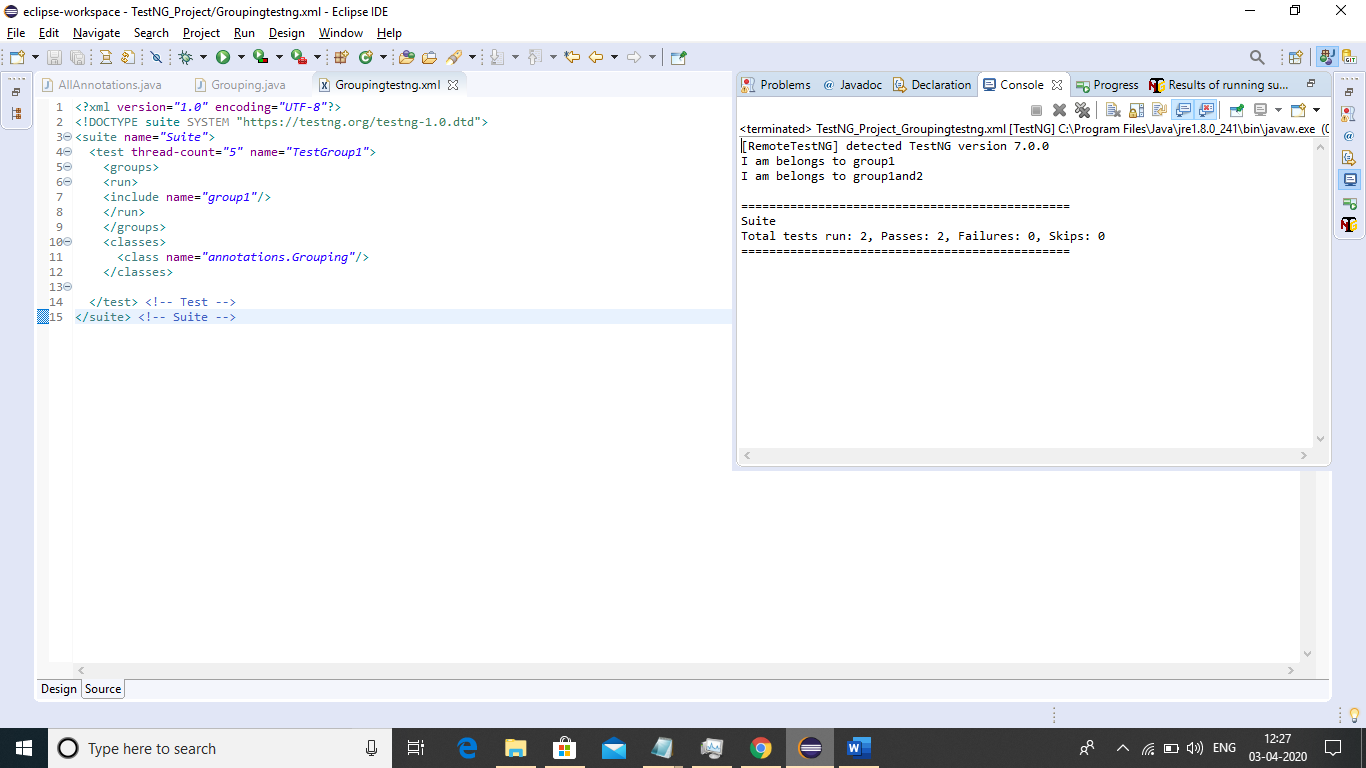
Note: if we put any method without @Test annotation or any other annotation, that particular method is not going to run.

### Grouping in TestNG

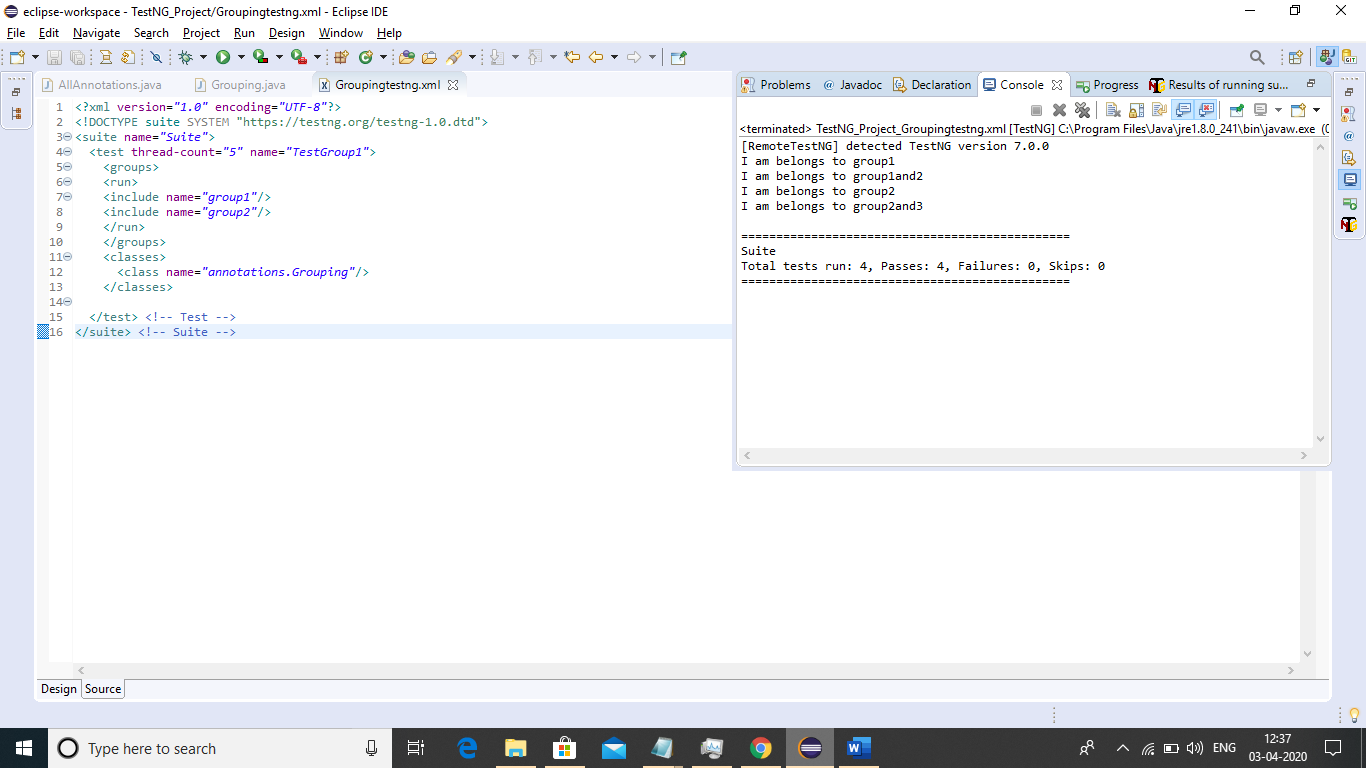
#### Coding:



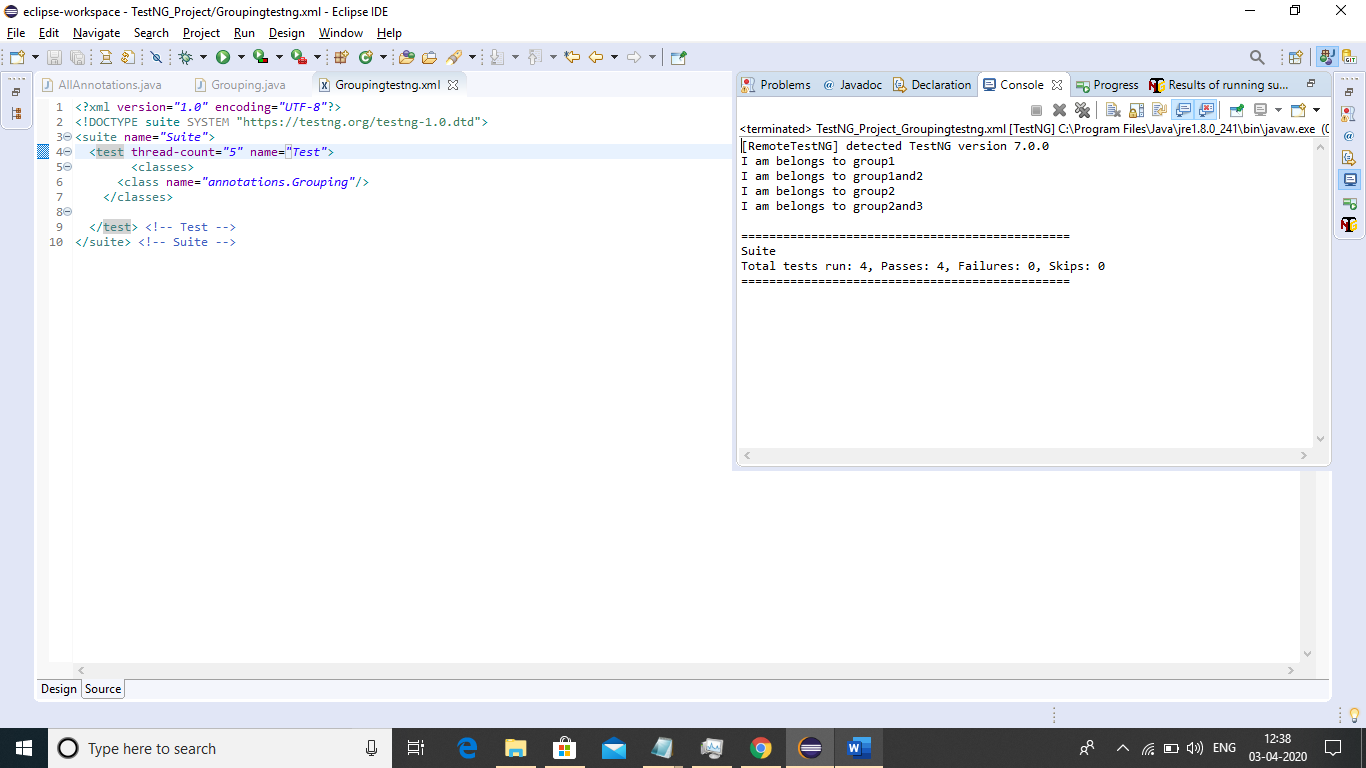
Running a method which belongs to only group1



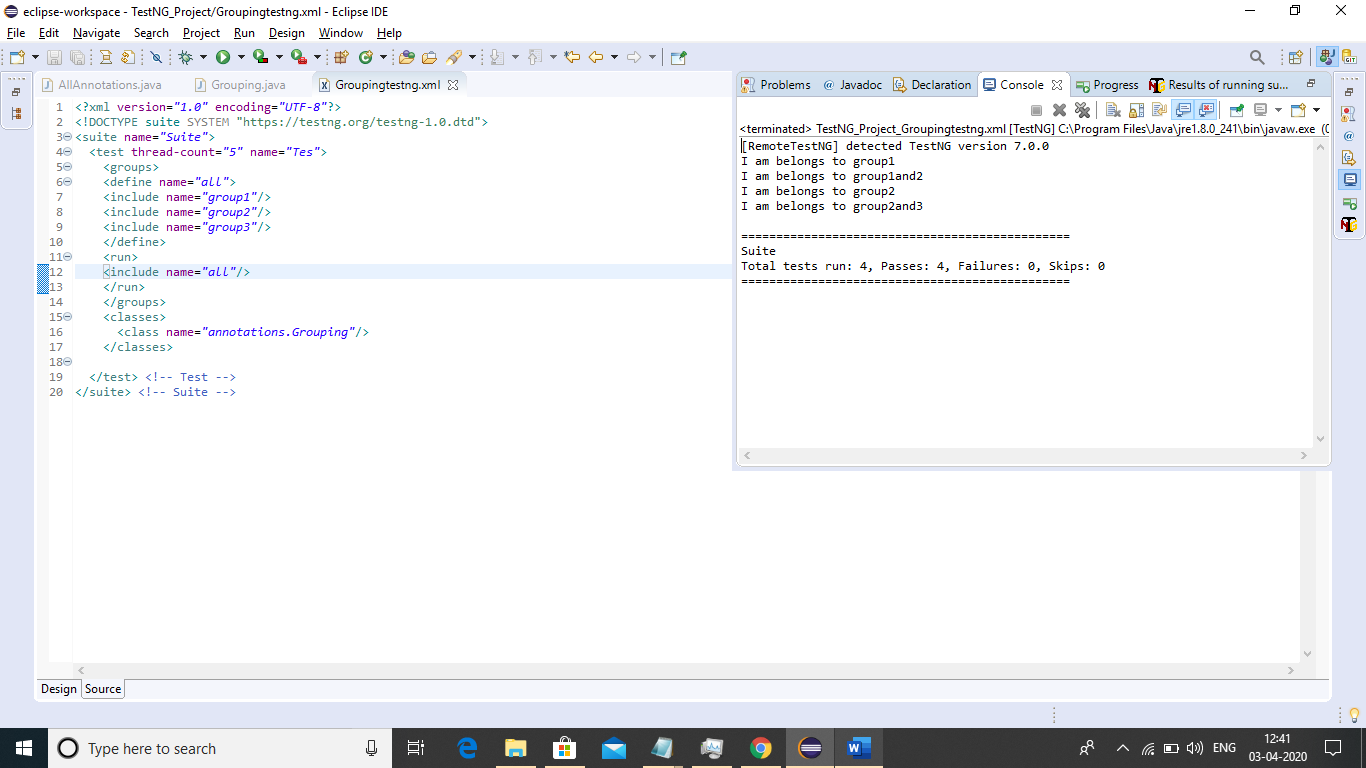
Running a method which belongs to group1, group2



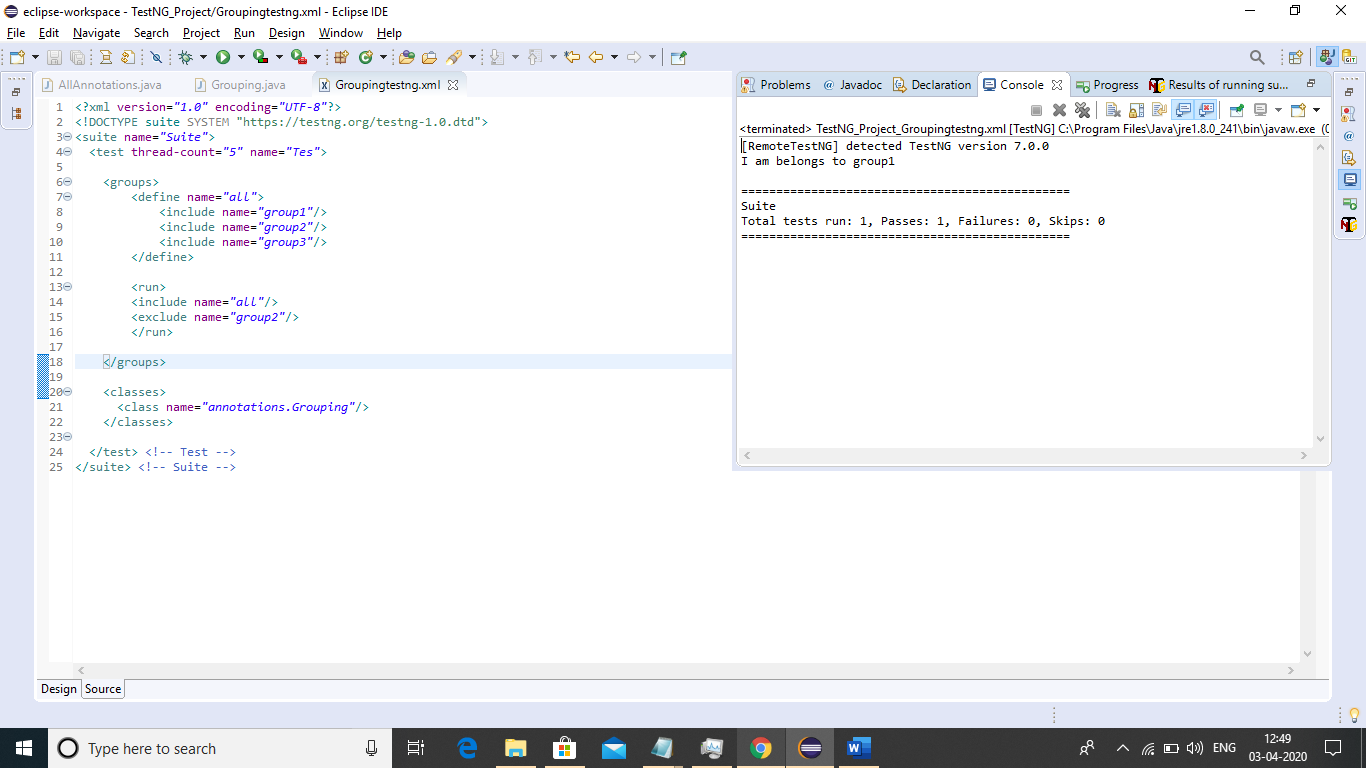
How to run all the groups?



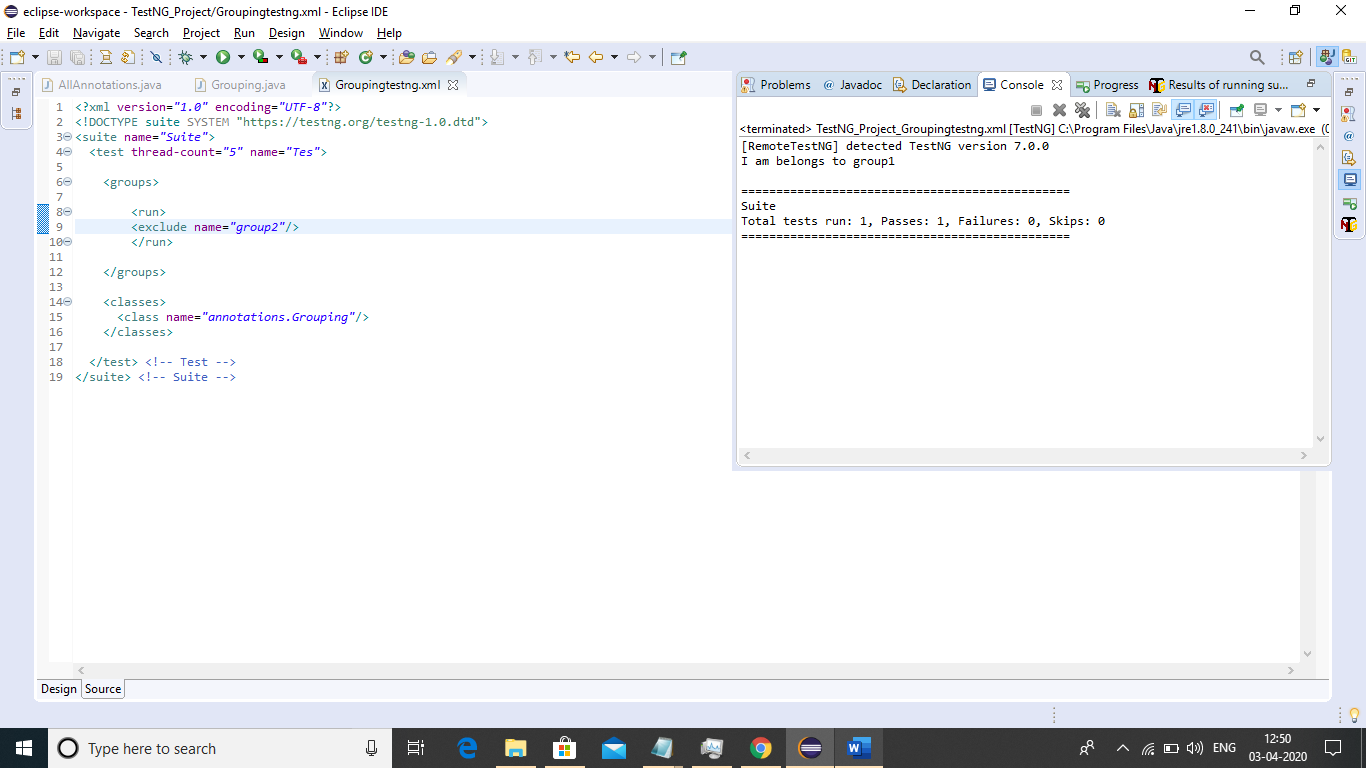
How to run all groups using include tag?



How to run all group but excluding only one group



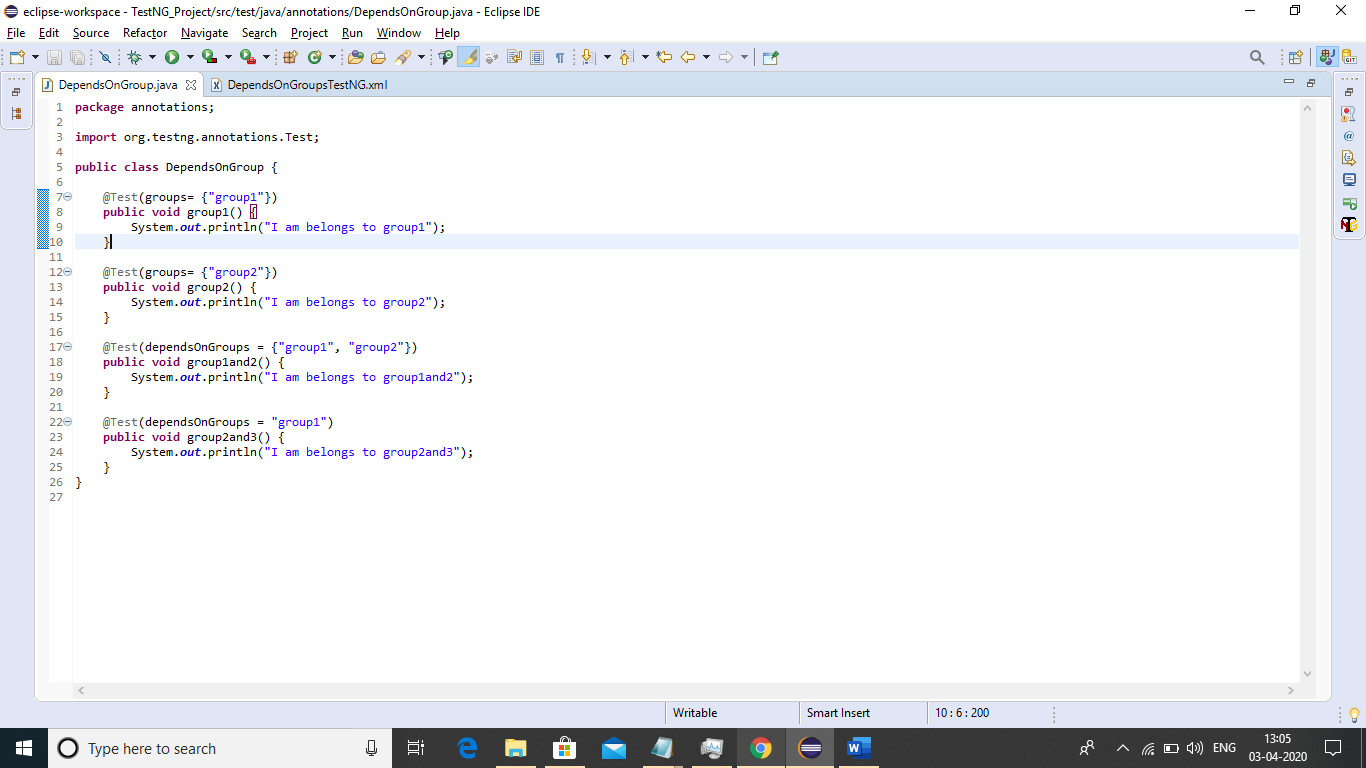
OR

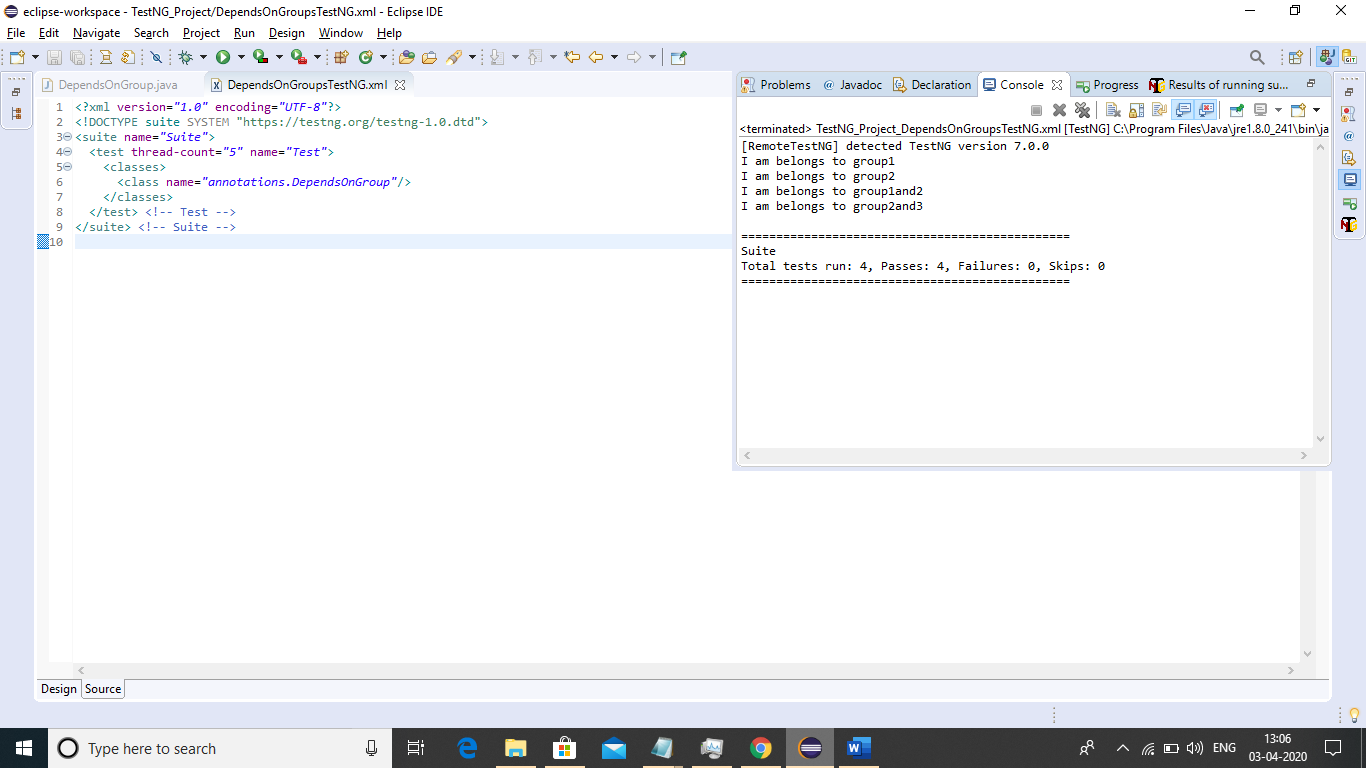


### DependsOnGroup in TestNG

#### When every group works fine!

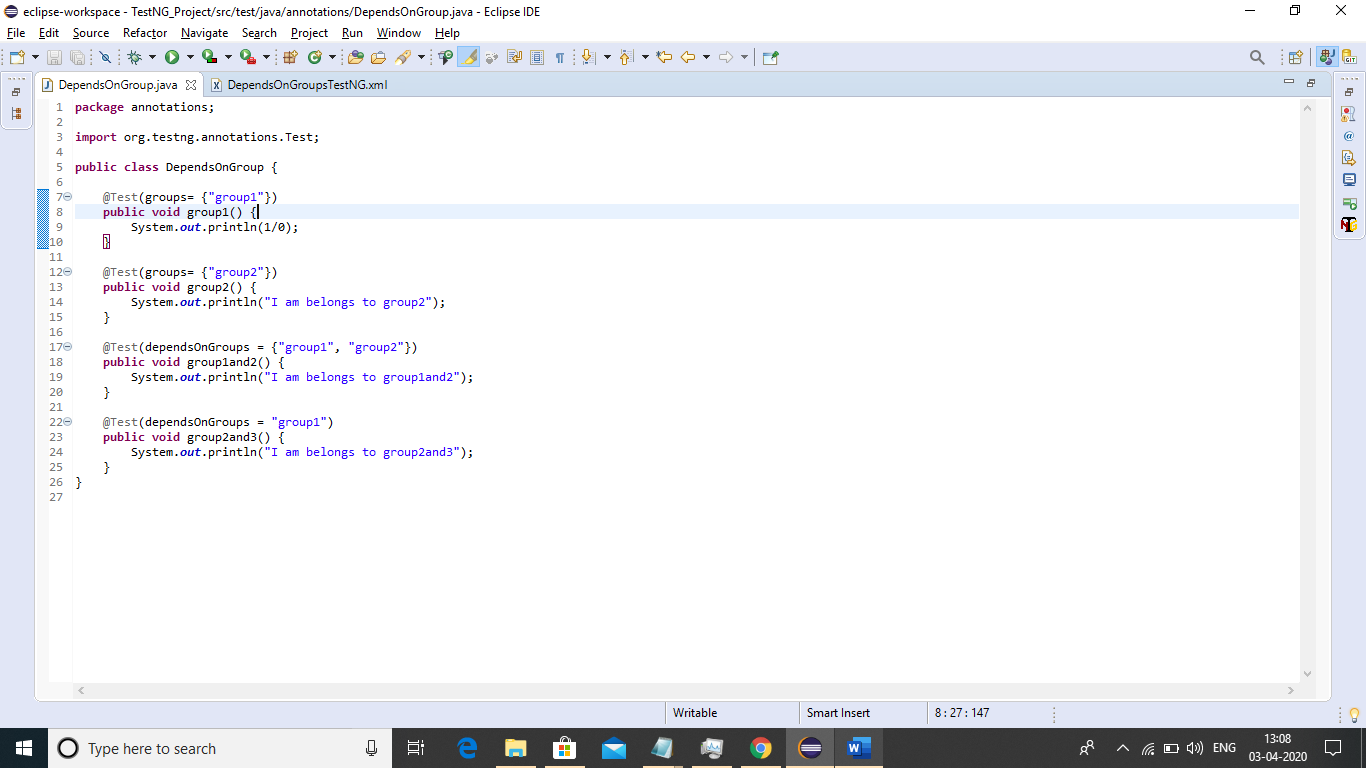
Code:

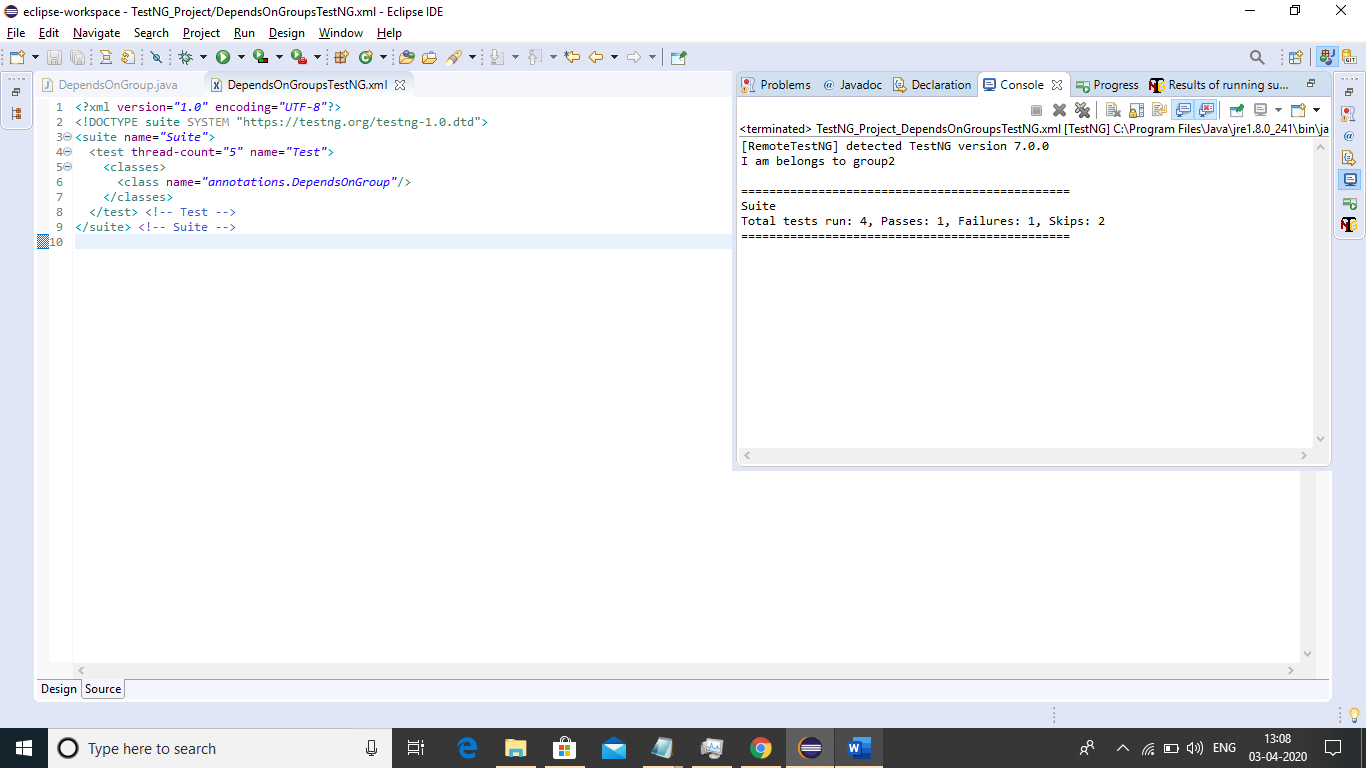




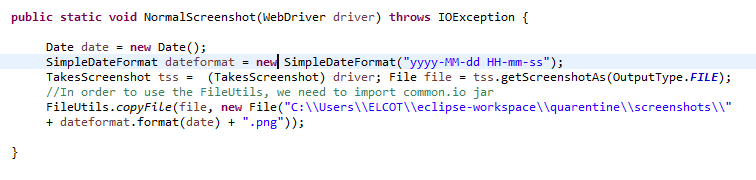
When if one group throws error or not execute

Code:





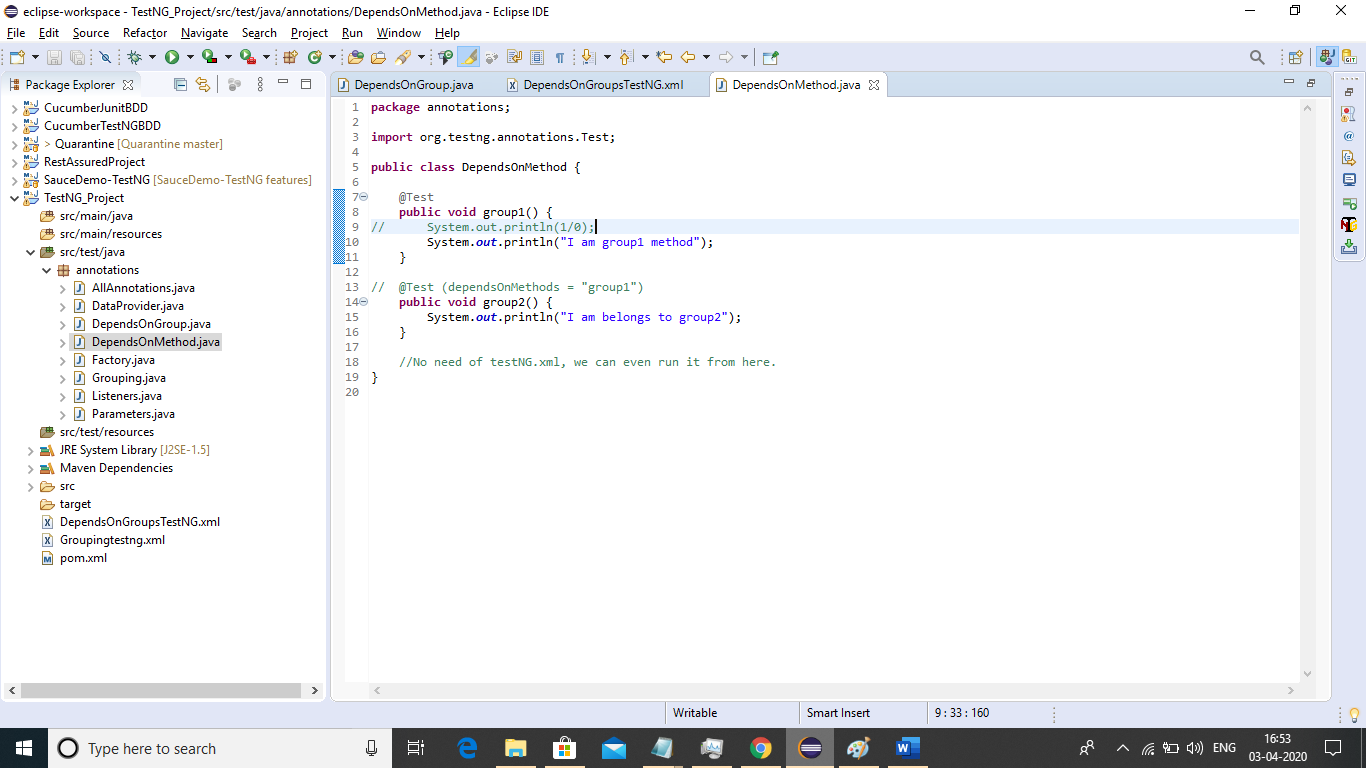
### Taking Screenshots in Selenium

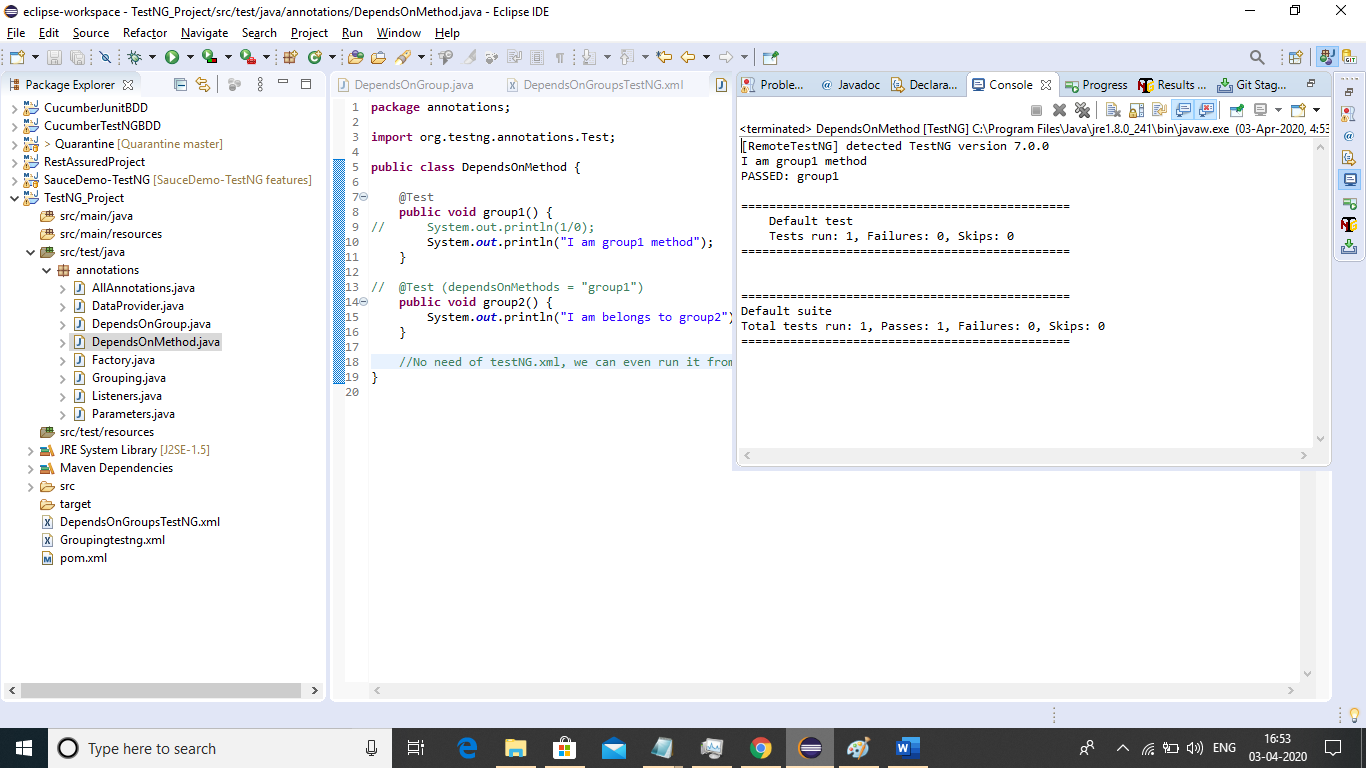


### Depends On method in Test NG

#### When all methods are working fine

Code:





#### When depends on method fails

Code:

