Section 3

1. What is selenium framework? Types of framework?
2. Example without DDF?
3. Example of DDF?
4. POM(Page object module) with pagefactory
5. POM with DDF?
6. TestNG //TDD //BDD-cucumber
    1. Advantages
    2. Annotations
    3. keywords
    4. Emailable report
    5. Test-suite
    6. Assertion / Verification
    7. failed.xml file
    8. disable TC execution for suite
    9. Grouping of test cases
    10. Parallel testing
    11. Multi browser testing/C.T
    12. Multi browser parallel testing
    13. TestNG data provider
    14. Difference between TestNG and Junit
7. POM DDF-testNG
8. POM DDF-testNG base and utility class
9. Property file & Listener(capture failed test cases)
10. Mavan project
11. Log 4j & Extend Reporter
12. Framework explanation
13. Interview questions
14. Logical programs
15. Github – push/pull code
16. **Scroll up & dwon**
17. **Scroll right & left**
18. **Jenkins**

1. **What is Selenium Framework? Types of Framework?**

- The Selenium Framework is a **code structure** that **makes** code maintenance easy and efficient. **Without** frameworks, users may place the "code" and "data" at the same location which is neither reusable nor readable.
- Frameworks produce **beneficial outcomes like** increased code reusability, higher portability, reduced cost of script maintenance, better code readability, etc.
- Framework is a set of rules to decide own and follow them in your project.

Types of framework

There are mainly three type of frameworks **created by** Selenium WebDriver **to automate** manual test cases

1. Data Driven
2. Keyword Driven
3. Hybrid (Data Driven+ Keyword Driven)

1. Data Driven

- Data Driven Framework **in** Selenium **is a** method of **separating** Test data **from** the test case.
- **Once** the Test data are **separated** from the test case, it can be **easily** modified for a specific functionality **without** changing the code.
- It is **used** to **fetch** Test Data **from** external files **like** Excel, .csv, .xml or some database tables.

Which framework you have to use in your project/selenium framework?
In the selenium framework/project I have to use data drive framework

2. **Example without DDF?**

```
package Selenium_Framework;

import java.util.concurrent.TimeUnit;

import org.openqa.selenium.By;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.chrome.ChromeDriver;

public class Example_Without_DDF
{
	public static void main(String[] args)
	{
		//To Open Browser
		System.setProperty("webdriver.chrome.driver",
"C:\\Users\\HP\\Downloads\\chromedriver_win32\\chromedriver.exe");
		WebDriver driver = new ChromeDriver();

		//Implicitly Wait
		driver.manage().timeouts().implicitlyWait(20, TimeUnit.SECONDS);

		//To maximize browser
		driver.manage().window().maximize();

		//To enter url/ open kite application
		driver.get("https://kite.zerodha.com/");

		//To enter UN
driver.findElement(By.xpath("//input[@id='userid']")).sendKeys("DV1510");

		//To enter PWD
driver.findElement(By.xpath("//input[@id='password']")).sendKeys("Vijay@123")
;

		//To click on login
		driver.findElement(By.xpath("//button[text()='Login ']")).click();

		//To enter PIN
	driver.findElement(By.xpath("//input[@id='pin']")).sendKeys("959594");

		//To click on continue button
		driver.findElement(By.xpath("//button[text()='Continue ']")).click();

		//get profile name
String actPN = driver.findElement(By.xpath("//span[text()='KV']")).getText();
		System.out.println(actPN);
		String expPN = "KV";

		if(actPN.equals(expPN))
		{
			System.out.println("pass");
		}
		else
		{
			System.out.println("fail");
		}
	}
}
```

3. Example of DDF?

```
package Selenium_Framework;

import java.io.FileInputStream;
import java.io.FileNotFoundException;
import java.io.IOException;
import java.util.concurrent.TimeUnit;

import org.apache.poi.EncryptedDocumentException;
import org.apache.poi.ss.usermodel.Sheet;
import org.apache.poi.ss.usermodel.WorkbookFactory;
import org.openqa.selenium.By;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.chrome.ChromeDriver;

public class Example_DDF
{
	public static void main(String[] args) throws
EncryptedDocumentException, IOException
	{
		//To open browser
		System.setProperty("webdriver.chrome.driver",
"C:\\Users\\HP\\Downloads\\chromedriver_win32\\chromedriver.exe");
		WebDriver driver = new ChromeDriver();
```

```
			//implicitly wait
			driver.manage().timeouts().implicitlyWait(20, TimeUnit.SECONDS);

			//To maximize browser
			driver.manage().window().maximize();

			//To enter url/open kite zerodha application
			driver.get("https://kite.zerodha.com/");

			//excel sheet for data fetch
			FileInputStream file = new FileInputStream("D:\\Dec2020.xlsx");
			Sheet data = WorkbookFactory.create(file).getSheet("DDF");

			//To enter UN
//			String UN = data.getRow(0).getCell(0).getStringCellValue();
//			driver.findElement(By.xpath("//input[@id='userid']")).sendKeys(UN);

			//Other approach for UN
driver.findElement(By.xpath("//input[@id='userid']")).sendKeys(data.getRow(0).
getCell(0).getStringCellValue());

			//To enter PWD
//			String PWD = data.getRow(0).getCell(1).getStringCellValue();
//		driver.findElement(By.xpath("//input[@id='password']")).sendKeys(PWD);

			//Other approach for PWD
driver.findElement(By.xpath("//input[@id='password']")).sendKeys(data.getRow
(0).getCell(1).getStringCellValue());

			//To click on login
			driver.findElement(By.xpath("//button[text()='Login ']")).click();

			//To enter PIN
//			String PIN = data.getRow(0).getCell(2).getStringCellValue();
//			driver.findElement(By.xpath("//input[@id='pin']")).sendKeys(PIN);

			//Other approach for PIN
driver.findElement(By.xpath("//input[@id='pin']")).sendKeys(data.getRow(0).
getCell(2).getStringCellValue());
```

```
		//To click on continue button
		driver.findElement(By.xpath("//button[text()='Continue ']")).click();

		//To get profile name
String actPN = driver.findElement(By.xpath("//span[text()='KV']")).getText();
		System.out.println(actPN);

		//Dynamic profile name- find dynamic xpath for profile name
//String actPN =
driver.findElement(By.xpath("//div[@class='avatar']/span")).getText();
//		System.out.println(actPN);

	String expPN = data.getRow(0).getCell(3).getStringCellValue();

		if(expPN.equals(actPN))
		{
			System.out.println("pass");
		}
		else
		{
			System.out.println("fail");
		}
	}
}
```

Implicitly wait-

- When automation script move from the one webpage to another webpage and sometime page is load or slow internet speed so that time script not immediately fail but it wait till what time we have to give in the implicitly wait.
- When move from one webpage to another webpage that time to match browser speed and selenium speed for that purpose to use implicitly wait.

Where is store your test data?
We store our test data in to excel sheet.

```
package Selenium_Framework;

import java.io.FileInputStream;
import java.io.FileNotFoundException;
import java.io.IOException;
import java.util.concurrent.TimeUnit;

import org.apache.poi.EncryptedDocumentException;
import org.apache.poi.ss.usermodel.Sheet;
import org.apache.poi.ss.usermodel.WorkbookFactory;
import org.openqa.selenium.By;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.chrome.ChromeDriver;

public class Example_DDF1
{
	public static void main(String[] args) throws
EncryptedDocumentException, IOException
	{
		//To open browser
		System.setProperty("webdriver.chrome.driver",
"C:\\Users\\HP\\Downloads\\chromedriver_win32\\chromedriver.exe");
		WebDriver driver = new ChromeDriver();

		//implicitly wait
		driver.manage().timeouts().implicitlyWait(20, TimeUnit.SECONDS);

		//To maximize browser
		driver.manage().window().maximize();

		//To enter url/open kite zerodha application
		driver.get("https://kite.zerodha.com/");

		//excel sheet for data fetch
FileInputStream file = new FileInputStream("D:\\Dec2020.xlsx");
Sheet data = WorkbookFactory.create(file).getSheet("DDF");

		//To enter UN
driver.findElement(By.xpath("//input[@id='userid']")).sendKeys(data.getRow(0).
getCell(0).getStringCellValue());
```

```
				//To enter PWD
		driver.findElement(By.xpath("//input[@id='password']")).sendKeys(data.get
Row(0).getCell(1).getStringCellValue());

				//To click on login
				driver.findElement(By.xpath("//button[text()='Login ']")).click();

				//To enter PIN
driver.findElement(By.xpath("//input[@id='pin']")).sendKeys(data.getRow(0).
getCell(2).getStringCellValue());

				//To click on continue button
				driver.findElement(By.xpath("//button[text()='Continue ']")).click();

				//To get profile name
				String actPN =
driver.findElement(By.xpath("//span[text()='KV']")).getText();
				System.out.println(actPN);

				String expPN = data.getRow(0).getCell(3).getStringCellValue();

				if(expPN.equals(actPN))
				{
					System.out.println("pass");
				}
				else
				{
					System.out.println("fail");
				}
		}
}
```

4. **POM(Page object module) with pagefactory**

POM:

- It is a java design pattern **use for** design of classes **in** Test script.
- Page Object model **is an** object design pattern **in** Selenium, **where**
  1. Web pages are represented as classes &
  2. The various elements on the page are defined as variables on the class.
- In this case we will **use** Page Factory **to** initialize web elements that are **defined** in web page classes or Page Objects.
- POM Strictly **follows** encapsulation concept **where**
  1. **Data member** should be **declared** globally with access level **private**
  2. **Initialize** within a constructor with access level **public** using pagefactory
  3. **Utilize** within a method with access level **public**

Note:

1. **No of D.M.** that need to be **created** under a pom class will **depends** on no of element that need to be **handle** in a webpage.
2. Pom class will **not contain** a main method, to **run** a pom class we **require** another class **with** main() ie. **Test class**
3. Test class will **contain** all the navigation steps **to** test an application

Pom class:

1. Pom class **depends** on webpage present in an application.
2. **For each** webpage pom class will be **created**, no of POM class **depends** on no of webpages **present** in an application.
3. In each POM class D.M./variable are **created** in encapsulation concept **by using** pagefactory.
4. No of D.M. created in POM class will **depend** on no of elements present in a webpage.
5. Each **declared D.M.** should **initialized & utilized** in POM class.

Test class:

1. Test class **depends** on no of Test cases **written** by manual Test engineer.
2. Test class will **contains** navigation steps & inputs that need to be **given** to the components/elements.
3. In test class **data/inputs** that can be given **directly** or **through** external source like Excel sheet.

Pagefactory:

- It is a class which **contains** static method like initElements.
- To initialize D.M./variable in PageFactory we **need** to **use** initElements method **within** the constructor.
  Syntax:
  ```
  PageFactory.initElements(driver, this);
  ```

- initElements will initialize D.M **by** identifying each component present in a webpage by **using** @findBy annotation,which **takes** locator type as an input.
  Syntax:
  ```
  @FindBy(locator Type ="locator value/exression")
  private WebElement D.M. ;
  ```

Working of PageFactory:

1. While executing Test Script initElement method will convert all the data members @findBy annotation to findElement(),this process is known as **basic/early initialization**.--**after** creating object of pom class
   @FindBy(xpath="//span[text()='KV']") private WebElement PN; ----> private WebElement PN = driver.findElement(By.xpath("//span[text()='KV']"));

2. To perform action on component we need to **call a methods**.
3. Before performing each action initElement method will identifies component **present or not**, then it will do complete initialization this process is known as **late/lazy initialization.**

Disadvantage of POM:(why use pom with pagefactory?)

- **POM will initialize all the component before performing actions**, but sometimes application may contains few components which will be **hidden** & displayed **once we perform action on components**, that hidden component will **not be displayed** while pom initializing, so it throws "No such element" exception.
- To overcome drawback of pom, we need to use "PageFactory" which is an extension of pom.

Advantages

1. Reduces code duplication
2. Improve test maintenance
3. Makes the code reusable
4. It is useful in reducing code duplication
5. It makes ease in maintaining the code
6. Makes code readable
7. The code becomes less and optimized

Difference between POM and PageFactory

POM	**PageFactory**
It will initialize all the data member present in class **completely before** performing action on components	It will initialize all the data member present in class performing each action
It will use if webpage is not containing hidden element	It will use if webpage is containing hidden element

@FindBy:

- An annotation used in page factory to locate and declare webelements using different locators
- Below are locators that can be used-all locator types

initElements():

- initElements is a static method in Page Factory class. This method which accept the two parameter such as driver and this keyword.
- Using the initElements method, one can initialize all the web elements located by @FindBy annotation.

Where to use encapsulation in your project?
At the time of POM design we declare private before declaring the variable that time we use encapsulation concept in our project.

NullPoniterException- To get the data from excel sheet and to enter some index(0,5) but that index data is not present so that time get NullPointerExp.

@POM program with page factory without DDF

```
package POM_with_Pagefactory;

import org.openqa.selenium.WebDriver;
import org.openqa.selenium.WebElement;
import org.openqa.selenium.support.FindBy;
import org.openqa.selenium.support.PageFactory;

public class KiteLogin1page
{
	//Step1: Declaration

	@FindBy(xpath=("//input[@id='userid']")) private WebElement UN;
	@FindBy(xpath=("//input[@id='password']")) private WebElement PWD;
	@FindBy(xpath=("//button[text()='Login ']")) private WebElement Login;

	//Step2: Initialization

	public KiteLogin1page(WebDriver driver)
	{
		PageFactory.initElements(driver, this);
	}

	//Step3: usage

	//Enter UN
	public void enterUN()
	{
		UN.sendKeys("DV1510");
	}

	//Enter PWD
	public void enterPWD()
	{
		PWD.sendKeys("Vijay@123");
	}

	//Click on login button
	public void clickOnLoginButton()
	{
		Login.click();
	}
}


package POM_with_Pagefactory;

import org.openqa.selenium.WebDriver;
import org.openqa.selenium.WebElement;
import org.openqa.selenium.support.FindBy;
import org.openqa.selenium.support.PageFactory;

public class KiteLogin2Page
{
	//Step1: Declaration

	@FindBy(xpath=("//input[@id='pin']")) private WebElement Pin;
	@FindBy(xpath=("//button[text()='Continue ']")) private WebElement
ContinueButton;

	//Step2: Initialization
	public KiteLogin2Page(WebDriver driver)
	{
		PageFactory.initElements(driver, this);
	}

	//Step3: Usage
	public void enterPin()
	{
		Pin.sendKeys("959594");
	}
	public void clickOnContinueButton()
	{
		ContinueButton.click();
	}
}
```

```java
package POM_with_Pagefactory;

import org.openqa.selenium.WebDriver;
import org.openqa.selenium.WebElement;
import org.openqa.selenium.support.FindBy;
import org.openqa.selenium.support.PageFactory;

public class KiteHomePage
{
	//Step1: Declaration

	@FindBy(xpath=("//span[text()='KV']")) private WebElement PN;

	//Step2: Initialization

	public KiteHomePage(WebDriver driver)
	{
		PageFactory.initElements(driver, this);
	}

	//Step3: Usage

	public void verifyPN()
	{
		String actPN = PN.getText();

		String expPN = "KV";

		if(expPN.equals(actPN))
		{
			System.out.println("pass");
		}

		else
		{
			System.out.println("fail");
		}
	}
}
```

```
package POM_with_Pagefactory;

import java.util.concurrent.TimeUnit;

import org.openqa.selenium.WebDriver;
import org.openqa.selenium.chrome.ChromeDriver;

public class KiteLoginTest
{
	public static void main(String[] args) throws InterruptedException
	{
		//To open browser
		System.setProperty("webdriver.chrome.driver",
"C:\\Users\\HP\\Downloads\\chromedriver_win32\\chromedriver.exe");
		WebDriver driver = new ChromeDriver();

		//implicitly wait
		driver.manage().timeouts().implicitlyWait(20, TimeUnit.SECONDS);

		//To maximize browser
		driver.manage().window().maximize();

		//To enter url/open kite zerodha application
		driver.get("https://kite.zerodha.com/");

		Thread.sleep(2000);

		KiteLogin1page login1 = new KiteLogin1page(driver);
		login1.enterUN();
		login1.enterPWD();
		login1.clickOnLoginButton();

		KiteLogin2Page login2 = new KiteLogin2Page(driver);
		login2.enterPin();
		login2.clickOnContinueButton();

		KiteHomePage home = new KiteHomePage(driver);
		home.verifyPN();
	}
}
```

@Pow with pagefactory without DDF

```
package POM_with_Pagefactory;

import org.openqa.selenium.WebDriver;
import org.openqa.selenium.WebElement;
import org.openqa.selenium.support.FindBy;
import org.openqa.selenium.support.PageFactory;

public class KiteLoginPage1
{
	//Step1: declaration
	@FindBy(xpath="//input[@id='userid']")      private WebElement UN;
	@FindBy(xpath="//input[@id='password']") private WebElement PWD;
	@FindBy(xpath="//button[text()='Login ']")   private WebElement Login;

	//Step2: initialization
	public KiteLoginPage1(WebDriver driver)
	{
		PageFactory.initElements(driver, this);
	}

	//Step3: usage
	public void enterUN()
	{
		UN.sendKeys("DV1510");
	}
	public void enterPWD()
	{
		PWD.sendKeys("Vijay@123");
	}
	public void clickOnLogin()
	{
		Login.click();
	}
}
```

```
package POM_with_Pagefactory;

import org.openqa.selenium.WebDriver;
import org.openqa.selenium.WebElement;
import org.openqa.selenium.support.FindBy;
import org.openqa.selenium.support.PageFactory;

public class KiteLoginPage2
{
	//Step1: declaration
	@FindBy(xpath="//input[@id='pin']") private WebElement Pin;
	@FindBy(xpath="//button[text()='Continue      ']")      private      WebElement
continutbutton;

	//Step2: initialization
	public KiteLoginPage2 (WebDriver driver)
	{
		PageFactory.initElements(driver, this);
	}

	//Step3: usage
	public void enterPin()
	{
		Pin.sendKeys("959594");
	}
	public void clickOnContinueButtopn()
	{
		continutbutton.click();
	}
}

package POM_with_Pagefactory;

import org.openqa.selenium.WebDriver;
import org.openqa.selenium.WebElement;
import org.openqa.selenium.support.FindBy;
import org.openqa.selenium.support.PageFactory;

public class KiteHomePage1
{
	//Step1: declaration
	@FindBy(xpath="//span[text()='KV']")	private WebElement PN;


	//Step2: initialization
	public KiteHomePage1(WebDriver driver)
	{
		PageFactory.initElements(driver, this);
	}

	//Step3: usage
	public void getPN()
	{
		String actPN = PN.getText();

		String expPN = "KV";

		if(expPN.equals(actPN))
		{
			System.out.println("pass");
		}
		else
		{
			System.out.println("fail");
		}
	}
}
```

```
package POM_with_Pagefactory;

import org.openqa.selenium.WebDriver;
import org.openqa.selenium.WebElement;
import org.openqa.selenium.support.FindBy;
import org.openqa.selenium.support.PageFactory;

public class KiteHomePageClickPN
{
	//Step1: declaration
	@FindBy(xpath="//span[text()='KV']") private WebElement ClickPN;
	@FindBy(xpath="//a[text()=' Logout']")	private WebElement LogOut;

	//Step2: initialization
	public KiteHomePageClickPN(WebDriver driver)
	{
		PageFactory.initElements(driver, this);
	}

	//Step3: usage
	public void clickOnPN()
	{
		ClickPN.click();
	}
	public void logout()
	{
		LogOut.click();
	}
}

package POM_with_Pagefactory;
import java.util.concurrent.TimeUnit;

import org.openqa.selenium.WebDriver;
import org.openqa.selenium.chrome.ChromeDriver;
```

```
public class KiteLoginTest2
{
                public static void main(String[] args) throws InterruptedException
                {
                        //To open browser
                        System.setProperty("webdriver.chrome.driver",
"C:\\Users\\HP\\Downloads\\chromedriver_win32\\chromedriver.exe");
                        WebDriver driver = new ChromeDriver();

                        //implicitly wait
                driver.manage().timeouts().implicitlyWait(20, TimeUnit.SECONDS);

                        //To maximize browser
                        driver.manage().window().maximize();

                        //To enter url/open kite zerodha application
                        driver.get("https://kite.zerodha.com/");

                        Thread.sleep(2000);

                        KiteLoginPage1 login = new KiteLoginPage1(driver);
                        login.enterUN();
                        login.enterPWD();
                        login.clickOnLogin();

                        KiteLoginPage2 login1 = new KiteLoginPage2(driver);
                        login1.enterPin();
                        login1.clickOnContinueButtopn();

                        KiteHomePage1 home1 = new KiteHomePage1(driver);
                        home1.getPN();

                KiteHomePageClickPN click = new KiteHomePageClickPN(driver);
                        click.clickOnPN();
                        click.logout();
```

```
	}
}
```

@POM with PageFactory – to set the method name properly to follow thought

1. To enter value – use set classname elementname
2. To click on – use click classname elementname
3. To get text- use verify classname elementname

```
package POM_with_Pagefactory;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.WebElement;
import org.openqa.selenium.support.FindBy;
import org.openqa.selenium.support.PageFactory;

public class KLogin1Page
{
	//Step1: Declaration
	@FindBy(xpath="//input[@id='userid']") private WebElement UN;
	@FindBy(xpath="//input[@id='password']") private WebElement PWD;
	@FindBy(xpath="//button[text()='Login ']") private WebElement Login;

	//Step2: Initialization
	public KLogin1Page (WebDriver driver)
	{
		PageFactory.initElements(driver, this);
	}

	//Step3: Usage
	public void setKLogin1PageUsername()
	{
		UN.sendKeys("DV1510");
	}
	public void setKLogin1PagePasswprd()
	{
		PWD.sendKeys("Vijay@123");
	}
	public void clickKLogin1PageLoginButton()
	{
		Login.click();
	}
}
```

```
package POM_with_Pagefactory;

import org.openqa.selenium.WebDriver;
import org.openqa.selenium.WebElement;
import org.openqa.selenium.support.FindBy;
import org.openqa.selenium.support.PageFactory;

public class KLogin2Page
{
	//Step1: Declaration
	@FindBy(xpath="//input[@id='pin']") private WebElement Pin;
	@FindBy(xpath="//button[text()='Continue ']") private WebElement
Continue;

	//Step2: Initialization
	public KLogin2Page(WebDriver driver)
	{
		PageFactory.initElements(driver, this);
	}

	//Step3: Usage
	public void setKLogin2PagePin()
	{
		Pin.sendKeys("959594");
	}
	public void clickKLogin2PageContinueButton()
	{
		Continue.click();
	}
}

package POM_with_Pagefactory;

import org.openqa.selenium.WebDriver;
import org.openqa.selenium.WebElement;
import org.openqa.selenium.support.FindBy;
import org.openqa.selenium.support.PageFactory;

public class KHomePage
{
	//Step1: Declaration
	@FindBy(xpath="//span[text()='KV']") private WebElement PN;

	//Step2: Initialization
	public KHomePage(WebDriver driver)
	{
		PageFactory.initElements(driver, this);
	}

	//Step3: Usage
	public void verifyKHomePageProfilename()
	{
		String actPN = PN.getText();

		String expPN = "KV";

		if(expPN.equals(actPN))
		{
			System.out.println("pass");
		}
		else
		{
			System.out.println("fail");
		}
	}

	public void clickKHomePageProfilename()
	{
		PN.click();
	}
}

package POM_with_Pagefactory;

import org.openqa.selenium.WebDriver;
import org.openqa.selenium.WebElement;
import org.openqa.selenium.support.FindBy;
import org.openqa.selenium.support.PageFactory;

public class KProfilePage
```

```
{
	//Step1: Declaration
	@FindBy(xpath="//a[text()=' Logout']") private WebElement Logout;

	//Step2: Initialization
	public KProfilePage(WebDriver driver)
	{
		PageFactory.initElements(driver, this);
	}

	//Step3: Usage
	public void clickKProfilePageLogout()
	{
		Logout.click();
	}
}

package POM_with_Pagefactory;

import java.util.concurrent.TimeUnit;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.chrome.ChromeDriver;

public class KLogoutTest
{
	public static void main(String[] args)
	{
		//To open browser
		System.setProperty("webdriver.chrome.driver",
"C:\\Users\\HP\\Downloads\\chromedriver_win32\\chromedriver.exe");
		WebDriver driver = new ChromeDriver();

		//implicitly wait
		driver.manage().timeouts().implicitlyWait(20, TimeUnit.SECONDS);

		//maximize browser
		driver.manage().window().maximize();

		//To open kite application/enter url
		driver.get("https://kite.zerodha.com/");
```

```
                KLogin1Page login1 = new KLogin1Page(driver);
                login1.setKLogin1PageUsername();
                login1.setKLogin1PagePasswprd();
                login1.clickKLogin1PageLoginButton();

                KLogin2Page login2 = new KLogin2Page(driver);
                login2.setKLogin2PagePin();
                login2.clickKLogin2PageContinueButton();

                KHomePage home = new KHomePage(driver);
                home.verifyKHomePageProfilename();
                home.clickKHomePageProfilename();

                KProfilePage pp = new KProfilePage(driver);
                pp.clickKProfilePageLogout();
        }
}
```

5. POM with DDF?

1. **Test case script 1 to login kite application & to get profile name of user**

- **Convert without method parameter to with method parameter**

```
package POM_with_DDF;

import org.openqa.selenium.WebDriver;
import org.openqa.selenium.WebElement;
import org.openqa.selenium.support.FindBy;
import org.openqa.selenium.support.PageFactory;

public class KiteLogin1Page
{
        //Step1: Declaration

@FindBy(xpath="//input[@id='userid']")      private WebElement UN;
@FindBy(xpath="//input[@id='password']") private WebElement PWD;
@FindBy(xpath="//button[text()='Login ']") private WebElement Login;
```

```
	//Step2: Initialization

		public KiteLogin1Page(WebDriver driver)
		{
			PageFactory.initElements(driver, this);
		}

	//Step3: usage

		public void setKiteLoginTestUsername(String username)
		{
			UN.sendKeys(username);
		}
		public void setKiteLogin1PagePassword(String password)
		{
			PWD.sendKeys(password);
		}
		public void clickKiteLogin1PageLoginButton()
		{
			Login.click();
		}
}


package POM_with_DDF;

import org.openqa.selenium.WebDriver;
import org.openqa.selenium.WebElement;
import org.openqa.selenium.support.FindBy;
import org.openqa.selenium.support.PageFactory;

public class KiteLogin2Page
{
	//Step1: Declaration

@FindBy(xpath="//input[@id='pin']")private WebElement Pin;
@FindBy(xpath="//button[text()='Continue ']")	private WebElement Continue;
```

```
	//Step2: Initialization

	public KiteLogin2Page(WebDriver driver)
	{
		PageFactory.initElements(driver, this);
	}

	//Step3: usage

	public void setKiteLogin2PagePin(String pinvalue)
	{
		Pin.sendKeys(pinvalue);
	}
	public void clickKiteLogin2PageContinueButton()
	{
		Continue.click();
	}
}

package POM_with_DDF;

import org.openqa.selenium.WebDriver;
import org.openqa.selenium.WebElement;
import org.openqa.selenium.support.FindBy;
import org.openqa.selenium.support.PageFactory;

public class KiteHomePage
{
	//Step1: Declaration

	@FindBy(xpath="//span[text()='KV']")	private WebElement PN;


	//Step2: Initialization

	public KiteHomePage(WebDriver driver)
	{
		PageFactory.initElements(driver, this);
	}
```

```
        //Step3: usage
        public void verifyKiteHomePageProfileName(String expPN)
        {
                String actPN = PN.getText();

                if(expPN.equals(actPN))
                {
                        System.out.println("pass");
                }
                else
                {
                        System.out.println("fail");
                }
        }
}

package POM_with_DDF;

import java.io.FileInputStream;
import java.io.FileNotFoundException;
import java.io.IOException;
import java.util.concurrent.TimeUnit;

import org.apache.poi.EncryptedDocumentException;
import org.apache.poi.ss.usermodel.Sheet;
import org.apache.poi.ss.usermodel.WorkbookFactory;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.chrome.ChromeDriver;

public class KiteLoginTest
{
        public static void main(String[] args) throws
EncryptedDocumentException, IOException
        {
                //To load excel sheet
                FileInputStream file = new FileInputStream("D:\\Dec2020.xlsx");
                Sheet sh = WorkbookFactory.create(file).getSheet("DDF");
```

```
		//To open chrome browser
		System.setProperty("webdriver.chrome.driver", 
"C:\\Users\\HP\\Downloads\\chromedriver_win32\\chromedriver.exe");
		WebDriver driver = new ChromeDriver();
		
		//To give implicitly wait
		driver.manage().timeouts().implicitlyWait(20, TimeUnit.SECONDS);
		
		//To maximize browser
		driver.manage().window().maximize();
		
		//To open / enter url of kite app
		driver.get("https://kite.zerodha.com/");
		
		KiteLogin1Page login1 = new KiteLogin1Page(driver);
		
		String UN = sh.getRow(0).getCell(0).getStringCellValue();
		login1.setKiteLoginTestUsername(UN);
		
		String PWD = sh.getRow(0).getCell(1).getStringCellValue();
		login1.setKiteLogin1PagePassword(PWD);
		
		login1.clickKiteLogin1PageLoginButton();
		
		KiteLogin2Page login2 = new KiteLogin2Page(driver);
		
		String PIN = sh.getRow(0).getCell(2).getStringCellValue();
		login2.setKiteLogin2PagePin(PIN);
		
		login2.clickKiteLogin2PageContinueButton();
		
		KiteHomePage home = new KiteHomePage(driver);
		
		String PN = sh.getRow(0).getCell(3).getStringCellValue();
		home.verifyKiteHomePageProfileName(PN);
	}
}
```

2. **Test case script 2 to login kite application & to get profile name of user & to click on profile name and logout the kite application**

- **Convert without method parameter to with method parameter**

```
package POM_with_DDF;

import org.openqa.selenium.WebDriver;
import org.openqa.selenium.WebElement;
import org.openqa.selenium.support.FindBy;
import org.openqa.selenium.support.PageFactory;

public class KiteLogin1Page
{
        //Step1: Declaration

@FindBy(xpath="//input[@id='userid']")      private WebElement UN;
@FindBy(xpath="//input[@id='password']") private WebElement PWD;
@FindBy(xpath="//button[text()='Login ']") private WebElement Login;



        //Step2: Initialization

                public KiteLogin1Page(WebDriver driver)
                {
                        PageFactory.initElements(driver, this);
                }

        //Step3: usage

                public void setKiteLoginTestUsername(String username)
                {
                        UN.sendKeys(username);
                }
                public void setKiteLogin1PagePassword(String password)
                {
                        PWD.sendKeys(password);
                }
                public void clickKiteLogin1PageLoginButton()
                {
                        Login.click();
                }
}


package POM_with_DDF;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.WebElement;
import org.openqa.selenium.support.FindBy;
import org.openqa.selenium.support.PageFactory;

public class KiteLogin2Page
{
        //Step1: Declaration

@FindBy(xpath="//input[@id='pin']")private WebElement Pin;
@FindBy(xpath="//button[text()='Continue ']")     private WebElement Continue;



        //Step2: Initialization

        public KiteLogin2Page(WebDriver driver)
        {
                PageFactory.initElements(driver, this);
        }

        //Step3: usage

        public void setKiteLogin2PagePin(String pinvalue)
        {
                Pin.sendKeys(pinvalue);
        }
        public void clickKiteLogin2PageContinueButton()
        {
                Continue.click();
        }
```

```java
package POM_with_DDF;

import org.openqa.selenium.WebDriver;
import org.openqa.selenium.WebElement;
import org.openqa.selenium.support.FindBy;
import org.openqa.selenium.support.PageFactory;

public class KiteHomePage
{
	//Step1: Declaration

	@FindBy(xpath="//span[text()='KV']")	private WebElement PN;


	//Step2: Initialization

	public KiteHomePage(WebDriver driver)
	{
		PageFactory.initElements(driver, this);
	}

	//Step3: usage
	public void verifyKiteHomePageProfileName(String expPN)
	{
		String actPN = PN.getText();

		if(expPN.equals(actPN))
		{
			System.out.println("pass");
		}
		else
		{
			System.out.println("fail");
		}
	}
	public void clickKiteHomePageProfileName()
	{
		PN.click();
	}
}
```

```
package POM_with_DDF;

import org.openqa.selenium.WebDriver;
import org.openqa.selenium.WebElement;
import org.openqa.selenium.support.FindBy;
import org.openqa.selenium.support.PageFactory;

public class KiteHomeLOut
{
        //Step1: Declaration

        @FindBy(xpath="//a[text()=' Logout']") private WebElement logout;

        //Step2: Initialization

        public KiteHomeLOut(WebDriver driver)
        {
            PageFactory.initElements(driver, this);
        }

        //Step3: usage
        public void clickKiteHomeLogoutLogoutButton()
        {
            logout.click();
        }
}

package POM_with_DDF;

import java.io.FileInputStream;
import java.io.FileNotFoundException;
import java.io.IOException;
import java.util.concurrent.TimeUnit;

import org.apache.poi.EncryptedDocumentException;
import org.apache.poi.ss.usermodel.Sheet;
import org.apache.poi.ss.usermodel.WorkbookFactory;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.chrome.ChromeDriver;

public class KiteLogOutTest
{
	public static void main(String[] args) throws
EncryptedDocumentException, IOException
	{
		//To load excel sheet
		FileInputStream file = new FileInputStream("D:\\Dec2020.xlsx");
		Sheet sh = WorkbookFactory.create(file).getSheet("DDF");

		//To open chrome browser
		System.setProperty("webdriver.chrome.driver",
"C:\\Users\\HP\\Downloads\\chromedriver_win32\\chromedriver.exe");
		WebDriver driver = new  ChromeDriver();

		//To give implicitly wait
		driver.manage().timeouts().implicitlyWait(20, TimeUnit.SECONDS);

		//To maximize browser
		driver.manage().window().maximize();

		//To open / enter url of kite app
		driver.get("https://kite.zerodha.com/");

		KiteLogin1Page login1 = new KiteLogin1Page(driver);
		String UN = sh.getRow(0).getCell(0).getStringCellValue();
		login1.setKiteLoginTestUsername(UN);
					or
login1.setKiteLoginTestUsername(sh.getRow(0).getCell(0).getStringCellValue());

		String PWD = sh.getRow(0).getCell(1).getStringCellValue();
		login1.setKiteLogin1PagePassword(PWD);
					or
		login1.setKiteLogin1PagePassword(sh.getRow(0).getCell(1).getString
		CellValue());
		login1.clickKiteLogin1PageLoginButton();
```

```
                KiteLogin2Page login2 = new KiteLogin2Page(driver);
                String PIN = sh.getRow(0).getCell(2).getStringCellValue();
                login2.setKiteLogin2PagePin(PIN);
                                or
login2.setKiteLogin2PagePin(sh.getRow(0).getCell(2).getStringCellValue());
                login2.clickKiteLogin2PageContinueButton();

                KiteHomePage home = new KiteHomePage(driver);
                String PN = sh.getRow(0).getCell(3).getStringCellValue();
                home.verifyKiteHomePageProfileName(PN);
                                or
                home.verifyKiteHomePageProfileName(sh.getRow(0).getCell(3).
                getStringCellValue());
                home.clickKiteHomePageProfileName();

                KiteHomeLOut logout = new KiteHomeLOut(driver);
                logout.clickKiteHomeLogoutLogoutButton();
        }
}
```

6. TestNG //TDD //BDD-cucumber

TestNG

TestNG is a java unit framework **use** for writting/designing of Test classes.

->Example of normal Test Class
->Example of TestNG Test class

1. Emailable Report

1. Report generation is very **important** when you are **doing** the Automation Testing as well as for Manual Testing.
2. By looking at the result, you can **easily identify** how many test cases are passed, failed and skipped.
3. By looking at the report, you will come to **know** what the **status of the project** is.
4. Selenium web driver is **used** for automating the web-application, **but** it won't generate any reports.
5. The TestNG will **generate** the **default report**.

---Steps to generate Emailable report----

1. Execute Test class and refresh the project.
2. You will get test-output folder.
3. In That folder Right click on the "emailable-report.html" and select the option Open with the web browser **or** double click on it.

Note:

1. If we **use** sop() to display text as a output then result will be **displayed** in console **not** in emailable report.
2. To **display text** in emailable report we need to **use** static method **log** present in Reporter class.

   eg. Reporter.log("String msg", true)

2. Annotation

1. @Test
2. @BeforeMethod
3. @AfterMethod
4. @BeforeClass
5. @AfterClass

1. @Test:- **Use** for execution of test method/TC.

2. @BeforeMethod:- It is **use** for execution of test method **before** execution of every test method with an annotation @Test.

3. @AfterMethod:- It is **use** for execution of test method **after** execution of every test method with an annotation @Test.

4. @BeforeClass:- It is **use** for execution of test method **before** execution of test class.

5. @AfterClass:- It is **use** for execution of test method **after** execution of test class.

->Example of normal Test Class

```
package TestNG_Examples;

import org.openqa.selenium.WebDriver;
import org.openqa.selenium.chrome.ChromeDriver;

public class ExampleOfNormalTestNG
{
	//example without testNG
	public static void main(String[] args) throws InterruptedException
	{
		//To open browser
		System.setProperty("webdriver.chrome.driver", 
"C:\\Users\\HP\\Downloads\\chromedriver_win32\\chromedriver.exe");
		WebDriver driver = new ChromeDriver();
		
		Thread.sleep(5000);
		
		//To open google
		driver.get("https://www.google.com/");
		
		Thread.sleep(5000);
		
		//To close
		driver.close();
	}
}
```

->Example of TestNG Test class

1. To run the program in the testNG we do not use main method instead of that we have to create one regular method and before that we need to call annotation @Test—add testNG library and import.
2. One @Test means one test case so we have to use or mention the multiple @Test using multiple method. Create n no of test cases.
3. Test case or test method is consider when we declare @Test annotation before the method.

```java
package TestNG_Examples;

import org.openqa.selenium.WebDriver;
import org.openqa.selenium.chrome.ChromeDriver;
import org.testng.annotations.Test;

public class ExampleOfTestNG
{
	@Test
	public void m1() throws InterruptedException
	{
		//To open browser
		System.setProperty("webdriver.chrome.driver",
"C:\\Users\\HP\\Downloads\\chromedriver_win32\\chromedriver.exe");
		WebDriver driver = new ChromeDriver();

		Thread.sleep(5000);

		//To open google
		driver.get("https://www.google.com/");

		Thread.sleep(5000);

		//To close
		driver.close();
	}
}
```

Consol output

```
PASSED: m1

===============================================
    Default test
    Tests run: 1, Failures: 0, Skips: 0
===============================================

===============================================
Default suite
Total tests run: 1, Passes: 1, Failures: 0, Skips: 0
===============================================
```

How to identify the how many test cases present in the class how to know?

To identify how many @Test annotation present in the class then to decide total test class present in the class.

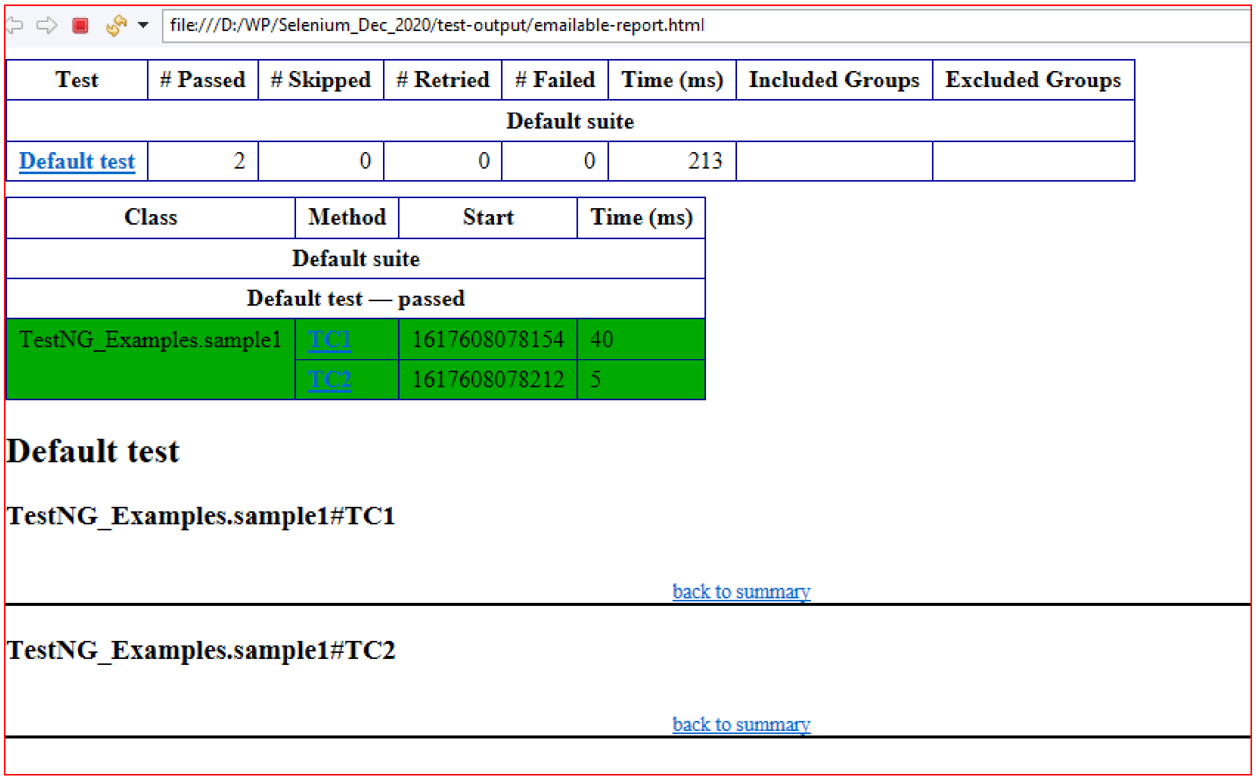
```
package TestNG_Examples;

import org.testng.annotations.Test;

public class sample1
{
	@Test
	public void TC1()
	{
		System.out.println("running TC1");
	}

	@Test
	public void TC2()
	{
		System.out.println("running TC2");
	}
}
```

Consol

```
[RemoteTestNG] detected TestNG version 7.3.0
running TC1
running TC2
PASSED: TC1
PASSED: TC2

===============================================
    Default test
    Tests run: 2, Failures: 0, Skips: 0
===============================================

===============================================
Default suite
Total tests run: 2, Passes: 2, Failures: 0, Skips: 0
===============================================
```

Why use this Emailable Report?---refer above emailable report

- Emailable report send to other person then to copy that emailable report html file and attach to the mail and send.
- Emailable report without output massage



file:///D:/WP/Selenium_Dec_2020/test-output/emailable-report.html

Test	# Passed	# Skipped	# Retried	# Failed	Time (ms)	Included Groups	Excluded Groups
Default suite							
Default test	2	0	0	0	213		

Class	Method	Start	Time (ms)
Default suite			
Default test — passed			
TestNG_Examples.sample1	TC1	1617608078154	40
	TC2	1617608078212	5

Default test

TestNG_Examples.sample1#TC1

back to summary

TestNG_Examples.sample1#TC2

back to summary

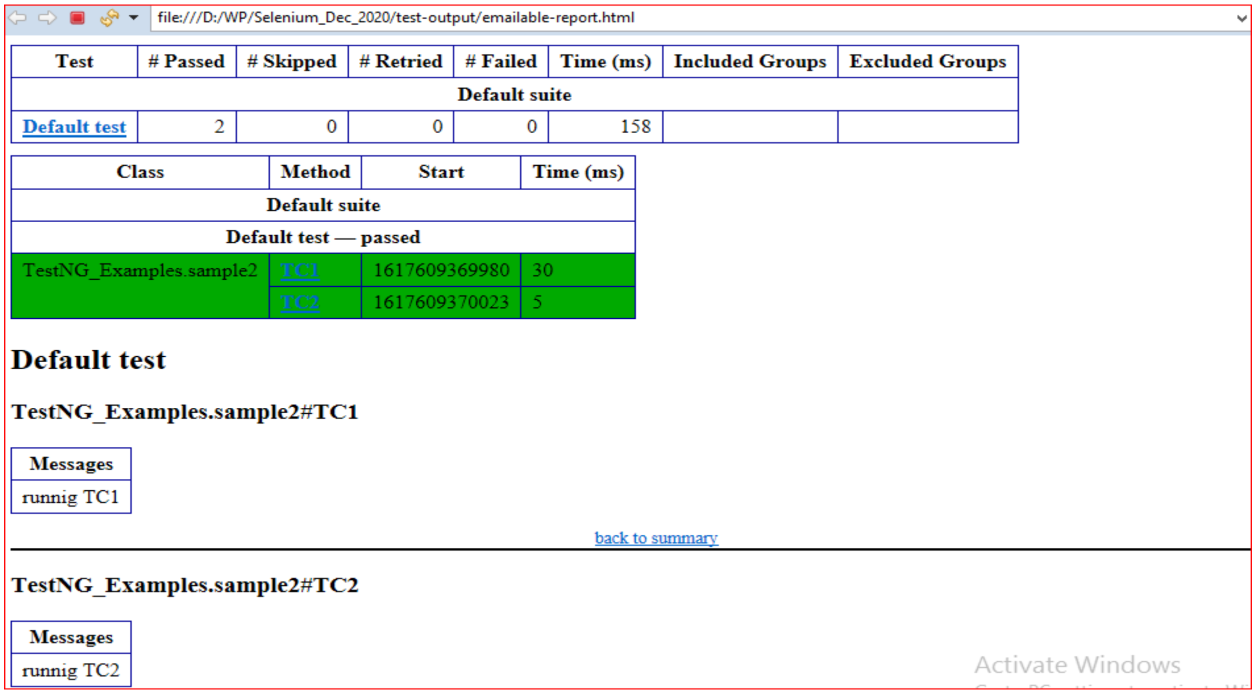
- Output massage seen only consol window and not seen in the emailable report.
- So that time if we need to see consol window massage to emailable report we cant use the sop() printing statement instead of that we need to use one class present in the testNG.
- It is a Reporter it contains a static method i.e. log() then to call this log() method which pass the two parameter. First parameter your massage which accept string argument and second true.
- If second parameter true is not pass so that time output massage not display in the consol window it will display only emailable report.
- So for that reasons we use second parameter ture and it will give if we need to see output massage in the consol window.
- To read output massage in both consol window and emialable report use true

```
package TestNG_Examples;

import org.testng.Reporter;
import org.testng.annotations.Test;

public class sample2
{
	@Test
	public void TC1()
	{
		Reporter.log("runnig TC1",true);
	}

	@Test
	public void TC2()
	{
		Reporter.log("runnig TC2",true);
	}
}
```

consol

```
[RemoteTestNG] detected TestNG version 7.3.0
runnig TC1
runnig TC2
PASSED: TC1
PASSED: TC2

===============================================
    Default test
    Tests run: 2, Failures: 0, Skips: 0
===============================================


===============================================
Default suite
Total tests run: 2, Passes: 2, Failures: 0, Skips: 0
===============================================
```

Emailable report including output massage


file:///D:/WP/Selenium_Dec_2020/test-output/emailable-report.html

Test	# Passed	# Skipped	# Retried	# Failed	Time (ms)	Included Groups	Excluded Groups
Default suite							
Default test	2	0	0	0	158		

Class	Method	Start	Time (ms)
Default suite			
Default test — passed			
TestNG_Examples.sample2	TC1	1617609369980	30
	TC2	1617609370023	5

Default test

TestNG_Examples.sample2#TC1

Messages
runnig TC1

back to summary

TestNG_Examples.sample2#TC2

Messages
runnig TC2

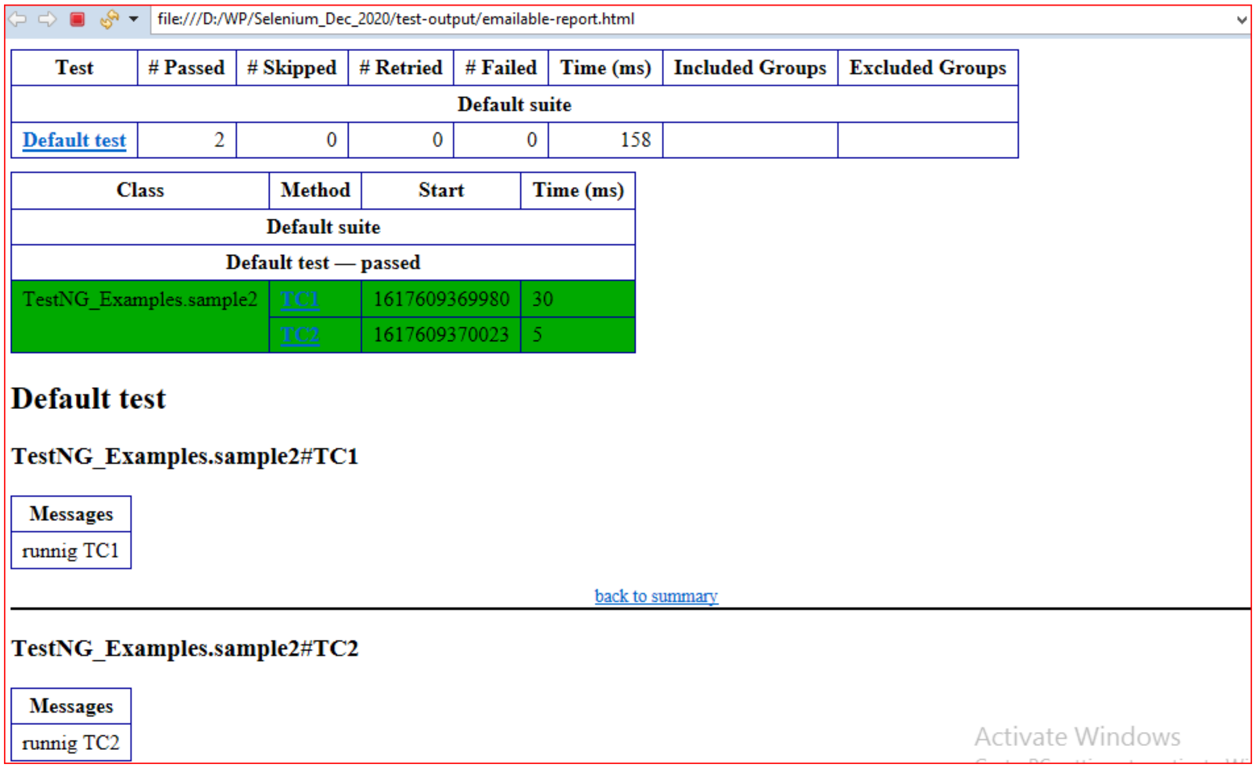
In the log() method second parameter true is not pass

```
package TestNG_Examples;

import org.testng.Reporter;
import org.testng.annotations.Test;

public class sample3
{
	@Test
	public void TC1()
	{
		Reporter.log("runnig TC1");
	}

	@Test
	public void TC2()
	{
		Reporter.log("runnig TC2");
	}
}
```

Consol-not display output massage

```
[RemoteTestNG] detected TestNG version 7.3.0
PASSED: TC1
PASSED: TC2

===============================================
    Default test
    Tests run: 2, Failures: 0, Skips: 0
===============================================


===============================================
Default suite
Total tests run: 2, Passes: 2, Failures: 0, Skips: 0
===============================================
```

But Emailable report to display output massage



file:///D:/WP/Selenium_Dec_2020/test-output/emailable-report.html

Test	# Passed	# Skipped	# Retried	# Failed	Time (ms)	Included Groups	Excluded Groups
Default suite							
Default test	2	0	0	0	158		

Class	Method	Start	Time (ms)
Default suite			
Default test — passed			
TestNG_Examples.sample2	TC1	1617609369980	30
	TC2	1617609370023	5

Default test

TestNG_Examples.sample2#TC1

Messages
runnig TC1

back to summary

TestNG_Examples.sample2#TC2

Messages
runnig TC2

Activate Windows

- Emailable report is override to other separate report not generated. If we run one test case and after some time to run another test case so that time emailable report is override.
- If we want separate report for first test case so to copy that report and store it

2.Annotations

- In consol window display all the output massages precondition, testcase and postconditon, but in the emailable report display only test case output massage.
- In the consol window show only one test case run and pass not show pre or post condition count.
- If two test cases are there & the runs sequence of these test cases are based on the alphabetical order for method name to execute.
- To execute one test case so first execute before method then actual test case and last after method execute.
- Same for second test case to check before method is present of not if present then execute then actual test case execute then check after method present of not then to execute after method.
- If 10 test cases are present that time one before method is execute 10 times and same for after method it is 10 time execute.
- In the annotation sequence is not important. We give any sequence of annotation but it is find out internally sequence.
- Before method and before method take one time in the class because it will execute each time it is no need to take multiple time.

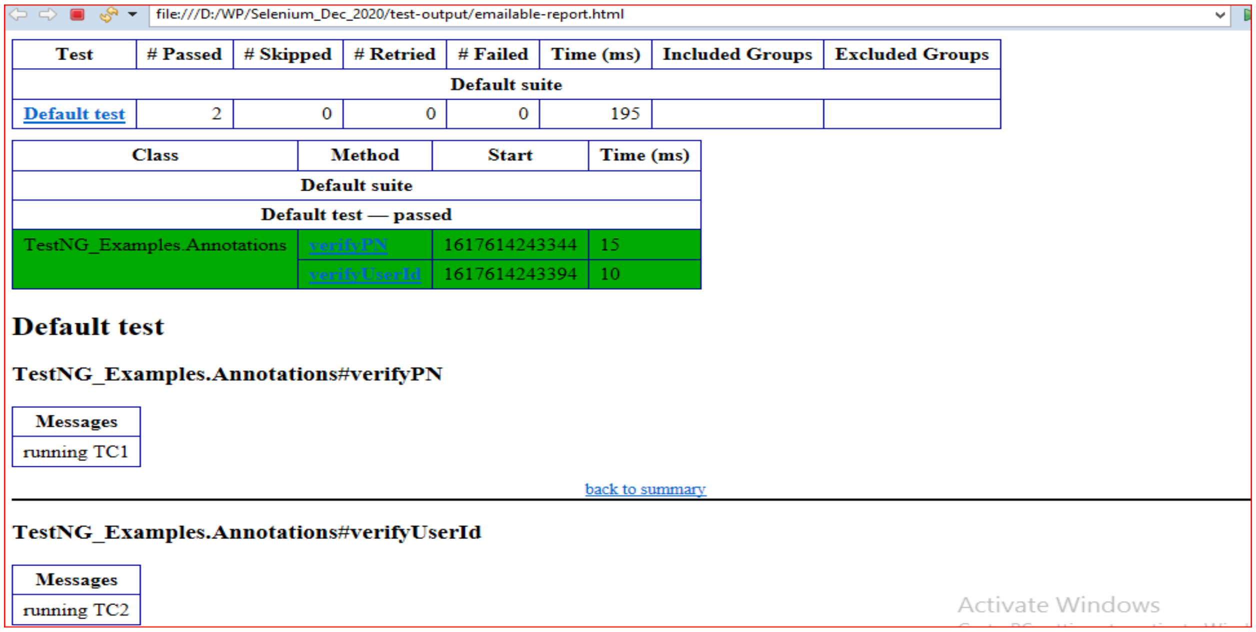
```java
package TestNG_Examples;
import org.testng.Reporter;
import org.testng.annotations.AfterClass;
import org.testng.annotations.AfterMethod;
import org.testng.annotations.BeforeClass;
import org.testng.annotations.BeforeMethod;
import org.testng.annotations.Test;

public class Annotations
{
```

```java
@BeforeClass
public void openBrowser()
{
	Reporter.log("Open browser", true);
}

//precondition
@BeforeMethod
public void loginToApp()
{
	Reporter.log("Login to application", true);
}

//TC1
@Test
public void verifyPN()
{
	Reporter.log("running TC1", true);
}

//TC2
@Test
public void verifyUserId()
{
	Reporter.log("running TC2", true);
}

//postcondition
@AfterMethod
public void logoutFromApp()
{
	Reporter.log("Logout from application", true);
}

@AfterClass
public void closeBrowser()
{
	Reporter.log("close browser", true);
}
```

```
}
```

Consol

```
[RemoteTestNG] detected TestNG version 7.3.0
Open browser
Login to application
running TC1
Logout from application
Login to application
running TC2
Logout from application
close browser
PASSED: verifyPN
PASSED: verifyUserId

===============================================
    Default test
    Tests run: 2, Failures: 0, Skips: 0
===============================================

===============================================
Default suite
Total tests run: 2, Passes: 2, Failures: 0, Skips: 0
===============================================
```



file:///D:/WP/Selenium_Dec_2020/test-output/emailable-report.html

Test	# Passed	# Skipped	# Retried	# Failed	Time (ms)	Included Groups	Excluded Groups
Default suite							
Default test	2	0	0	0	195		

Class	Method	Start	Time (ms)
Default suite			
Default test — passed			
TestNG_Examples.Annotations	verifyPN	1617614243344	15
	verifyUserId	1617614243394	10

Default test

TestNG_Examples.Annotations#verifyPN

Messages
running TC1

back to summary

TestNG_Examples.Annotations#verifyUserId

Messages
running TC2

3. **TestNG Keyword:**

1. invocationCount
2. priority
3. enabled
4. timeOut
5. dependsOnMethods

1. invocationCount:

- Sometimes same test method/TC need to be executed multiple which can be possible by using TestNG keyword "invocationCount"
  eg. invocationCount=5;

2. priority:

- To change test method/TC execution order we need to use TestNG keyword "priority".
  eg. priority=1

Note: priority can be 1. bydefault=0
2. +ve integer
3. -ve integer
4. Duplicate

priority can't be 1. Decimals
2. Variables

3. enabled:

- Disabling a test method/TC in TestNG can be achieved by setting the enabled attribute of the @Test annotation to false.
  eg. enabled=false

4.timeOut:

- If test class contains multiple test methods if one of the test method is time consuming to execute then TestNG bydefault fail that TC & executes the other TC.
  eg. @Test(timeOut=8000)

5. dependsOnMethods:

- If 1 TC execution depends on multiple TC then we need to use "dependsOnMethods" TestNG keyword.
  eg. dependsOnMethods= {"TC name"}

```
//1.invocationCount-run test case with multiple time
	@Test(invocationCount=5)
	public void TC1()
	{
		Reporter.log("running TC1", true);
	}
```

Consol

```
[RemoteTestNG] detected TestNG version 7.3.0
running TC1
running TC1
running TC1
running TC1
running TC1
```

```
//2.priority-by using priority to change the execution order
```

1.Default=0

```
	@Test
	public void TC1()
	{
		Reporter.log("running TC1", true);
	}

	@Test
	public void TC2()
	{
		Reporter.log("running TC2", true);
	}
```

Consol

```
	running TC1
	running TC2
```

2.positive

```
	@Test(priority=2)
	public void TC1()
	{
		Reporter.log("running TC1", true);
	}

	@Test(priority=1)
	public void TC2()
	{
		Reporter.log("running TC2", true);
	}
```

Consol

```
running TC2
running TC1
```

3.negative

```
	@Test
	public void TC1()
	{
		Reporter.log("running TC1", true);
	}

	@Test(priority=-1)
	public void TC2()
	{
		Reporter.log("running TC2", true);
	}
```

Consol

```
running TC2
running TC1
```

4.duplicate

```
	@Test(priority=-1)
	public void TC1()
	{
		Reporter.log("running TC1", true);
	}

	@Test(priority=-1)
	public void TC2()
	{
		Reporter.log("running TC2", true);
	}
```

Consol

```
running TC1
running TC2
```

How to disable execution of particular test case?

By using enable=false we can disable execution of test case

```
//3.enabled=false-disable execution one of test case or incomplete test case not execute of multiple test case

	@Test()
	public void TC1()
	{
		Reporter.log("running TC1", true);
	}

	@Test(enabled=false)
	public void TC2()
	{
		Reporter.log("running TC2", true);
	}
```

Consol

```
running TC1
```

```
//4.timeOut-test case take more time to execute so that time to fail that test case
and execute another test case

        @Test(timeOut=5000)
        public void TC1() throws InterruptedException
        {
                Reporter.log("running TC1", true);
                Thread.sleep(3000);
        }
```

Consol

running TC1

```
        @Test(timeOut=5000)
        public void TC1() throws InterruptedException
        {
                Reporter.log("running TC1", true);
                Thread.sleep(8000);
        }
```

Consol

```
running TC1
FAILED: TC1
org.testng.internal.thread.ThreadTimeoutException                    Method
TestNG_Examples.testNGKyeword.TC1() didn't finish within the time-out 5000
```

```
//5.dependsOnMethods={"method name"}-one test case depends on another
method so that time to use dependsOnMethods
        @Test()
        public void loginToApp()
        {
                Reporter.log("ruuning login to app", true);
        }

        @Test(dependsOnMethods= {"loginToApp"})
        public void logoutFromApp()
        {
```

```
		Reporter.log("ruuning logout from app", true);
	}
```

Consol

```
ruuning login to app
ruuning logout from app
PASSED: loginToApp
PASSED: logoutFromApp
===============================================
    Default test
    Tests run: 2, Failures: 0, Skips: 0
===============================================
===============================================
Default suite
Total tests run: 2, Passes: 2, Failures: 0, Skips: 0
===============================================
```

//5.dependsOnMethods={"method name"}-one test case depends on another method so that time to use dependsOnMethods. One test case fail then depends on method or test case skip

```
	@Test()
	public void loginToApp()
	{
		Reporter.log("ruuning login to app", true);
		Assert.fail();
	}

	@Test(dependsOnMethods= {"loginToApp"})
	public void logoutFromApp()
	{
		Reporter.log("ruuning logout from app", true);
	}
```

Consol

```
ruuning login to app
FAILED: loginToApp
===============================================
    Default test
    Tests run: 2, Failures: 1, Skips: 1
```

```
===============================================
===============================================
Default suite
Total tests run: 2, Passes: 0, Failures: 1, Skips: 1
```

4.Test-Suite:

- It is **xml file** which **contains** all the test classes name which need to be executed.
- It is **use to execute** all/multiple Test classes.
- To add classes in the throughout the project
- Syntax—**What is syntax of test suite?**

```
<suite name="Suite name">
  <test  name="Test name">
    <classes>
      <class name="packageName.className"/>
    </classes>
  </test>
</suite>
```

- **How to create test suite?**
  First to right click on any one class of out of multiple running class→click on TestNG→ convert to TestNG→change name of xml file or same→finish→create one xml file to your project→double click on that→select file source and give class name
- To run three class/multiple classes run at a time
- Test suite follow sequential order means we give class name that manner these are to be execut.

```
package TestNG_Examples;
import org.testng.Reporter;
import org.testng.annotations.Test;

public class demo1
{
	@Test
	public void TC1()
	{
		Reporter.log("runnig TC1",true);
	}

	@Test
	public void TC2()
	{
		Reporter.log("runnig TC2",true);
	}
}

package TestNG_Examples;

import org.testng.Reporter;
import org.testng.annotations.Test;

public class demo2
{
	@Test
	public void TC1()
	{
		Reporter.log("runnig TC1",true);
	}

	@Test
	public void TC2()
	{
		Reporter.log("runnig TC2",true);
	}
}

package TestNG_Examples;
import org.testng.Reporter;
import org.testng.annotations.Test;

public class demo3
{
	@Test
	public void TC1()
	{
```

```
                Reporter.log("runnig TC1",true);
        }
        @Test
        public void TC2()
        {
                Reporter.log("runnig TC2",true);
        }
}
```

Test Suite-

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE suite SYSTEM "https://testng.org/testng-1.0.dtd">
<suite name="Suite">
  <test thread-count="5" name="Test">
    <classes>
      <class name="TestNG_Examples.demo1"/>
      <class name="TestNG_Examples.demo2"/>
      <class name="TestNG_Examples.demo3"/>
    </classes>
  </test> <!-- Test -->
</suite> <!-- Suite -->
```

Or

one suite multiple test are possible but the test name are different and class name also different

Test Suite-

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE suite SYSTEM "https://testng.org/testng-1.0.dtd">
<suite name="Suite">
  <test thread-count="5" name="Test">
    <classes>
      <class name="TestNG_Examples.demo1"/>
      <class name="TestNG_Examples.demo2"/>
      <class name="TestNG_Examples.demo3"/>
    </classes>
  </test> <!-- Test -->
  <test thread-count="5" name="Test1">
    <classes>
      <class name="TestNG_Examples.sample1"/>
      <class name="TestNG_Examples.sample2"/>
      <class name="TestNG_Examples.sample3"/>
    </classes>
  </test> <!-- Test -->
</suite> <!-- Suite -->
```

6. Assertion / Verification

- If we use if else verification process is to verify expected result of a test case and the length of the test case is increase and also test script will take more time for execution.
- To reduce length of test script we need to use Assert class for verification which contains static method.
- Important static methods present in the Assert class and all the static method should be imported for org.testNG.
- Assert class contains static method like
    1. assertEquals()
    2. assertNotEquals()
    3. assertTrue()
    4. assertFalse()
    5. assertNull()
    6. assertNotNull()
    7. fail()
- By using this assert class static method we compare or matching the actual or expected result and to print test case pass or fail.

1. assertEquals()

- assertEquals() method used to verify expected and actual result, if both results are same then output is pass otherwise fail.

```
//1.assertEquals()
	@Test
	public void TC1()
	{
		String exp="hi";
		String act="hi";
		
		Assert.assertEquals(act, exp);
	}
	@Test
	public void TC2()
	{
		String exp="hi";
		String act="hello";
		
		Assert.assertEquals(act, exp, “exp & act results are different”);
	}
```

```
consol
PASSED: TC1
FAILED: TC2
java.lang.AssertionError: exp & act results are different expected [hi] but found
[hello]
```

2. assertNotEquals()

- assertEquals() method used to verify expected and actual result, if both results are not same then output is pass otherwise fail.

```
//2.assertNotEquals()
	@Test
	public void TC1()
	{
		String exp="hi";
		String act="hello";
		
		Assert.assertNotEquals(exp, act);
	}
	@Test
	public void TC2()
	{
		String exp="hi";
		String act="hi";

		Assert.assertNotEquals(exp, act, “both results are same”);
	}
consol
PASSED: TC1
FAILED: TC2
java.lang.AssertionError: both results are same did not expect [hi] but found [hi]
```

3.assertTrue()

- assertTrue() method use to verify condition are true or false, if condition is true output is pass otherwise fail.

```
//3.assertTrue()
	@Test
	public void TC1()
	{
		boolean result = true;

		Assert.assertTrue(result);
	}
	@Test
	public void TC2()
	{
		boolean result = false;

		Assert.assertTrue(result);
	}
Consol
PASSED: TC1
FAILED: TC2
java.lang.AssertionError: expected [true] but found [false]
```

4.assertFalse()

- assertTrue() method use to verify condition are true or false, if condition is false output is pass otherwise fail.

```
//4.assertFalse()
		@Test
		public void TC1()
		{
			boolean result = false;
			
			Assert.assertFalse(result);;
		}
		@Test
		public void TC2()
		{
			boolean result = true;
			
			Assert.assertFalse(result);
		}
```

```
Consol
PASSED: TC1
FAILED: TC2
java.lang.AssertionError: expected [false] but found [true]
```

5. assertNotNull()

- assertNotNull() method is use to verify component or text fields empty or not, if it is not empty output is pass otherwise fail.

```
//5.assertNotNull()
		@Test
		public void TC1()
		{
			String str = "Mangesh";
			
			Assert.assertNotNull(str);
		}
		@Test
		public void TC2()
		{
			String str = null;
			
			Assert.assertNotNull(str);
		}
```

```
Consol
PASSED: TC1
FAILED: TC2
java.lang.AssertionError: expected object to not be null
```

6. assertNull()

- assertNotNull() method is use to verify component or text fields empty or not, if it is empty output is pass otherwise fail.

```
//6.assertNull();
        @Test
        public void TC1()
        {
                String str = null;

                Assert.assertNull(str);
        }
        @Test
        public void TC2()
        {
                String str = "Mangesh";

                Assert.assertNull(str);
        }

}
```

Consol

```
PASSED: TC1
FAILED: TC2
java.lang.AssertionError: expected [null] but found [Mangesh]
```

7.fail()

- This method is use to intentionally fail test method
- In a test class if one of the method is fail then then testNG will stop execution of failed test method and other test method execution is continue.
- In a test class in one of the test method is failed and that test method execution required for other test method execution then other test methods will be skipped.

Disadvantages of assert class

- If a test class containing multiple test method, in one of the test method, multiple verification are presented, while executing if one verification is failed then rest of the verification will not be verified & testNG will execute next method by failing verification field method.

```
//disadvantages of Assert class
    @Test
        public void TC1()
        {
                String exp = "hi";
                String act = "hi";

                Assert.assertNotEquals(act, exp);
                Assert.assertEquals(act, exp);
                Assert.assertNotEquals(act, exp);
        }
```

Consol

```
FAILED: TC1
java.lang.AssertionError: did not expect [hi] but found [hi]
```

SoftAssert Class

- To overcome above drawback we need to use SoftAssert Class
- It is class which contains non-static methods use to do verification
- SoftAssert will do verification if any verification is failed, notifies and execute the rest of the verification in a test method.
- To create the object of the SoftAssert class and call assert class method same
- When to use soft assert class in the test method at the end to use assertAll() main role is here if verification is fail to notify that verification and give output in consol.

@SoftAssert Class

```
package TestNG_Examples;

import org.testng.annotations.Test;
import org.testng.asserts.SoftAssert;
```

```
public class Verification_SoftAssert
{
	//Soft Assert class

	@Test
	public void TC1()
	{
		String exp = "hi";
		String act = "hi";

		SoftAssert soft = new SoftAssert();

		soft.assertEquals(exp, act);
		soft.assertNotEquals(exp, act);
		soft.assertNotEquals(exp, act);

		soft.assertAll();
	}
}
```

Consol

```
FAILED: TC1
java.lang.AssertionError: The following asserts failed:
	did not expect [hi] but found [hi],
	did not expect [hi] but found [hi]
```

How to perform assertion and verification in selenium?

By using Assert class and Soft Assert class we can perform the verification and assertion. There are different method like—method name of assert class by using this method we can perform the verification

7. failed.xml

- While executing the automation scripts, test cases may fail for several reasons. To optimize our next runs, we need to re-run only failed test cases.

Steps to execute failed.xml file

1. Create testng. xml file under project folder.
2. execute testng.xml file
3. In the test-output folder >> testng-failed. xml file will be created.
4. execute "testng-failed. xml"

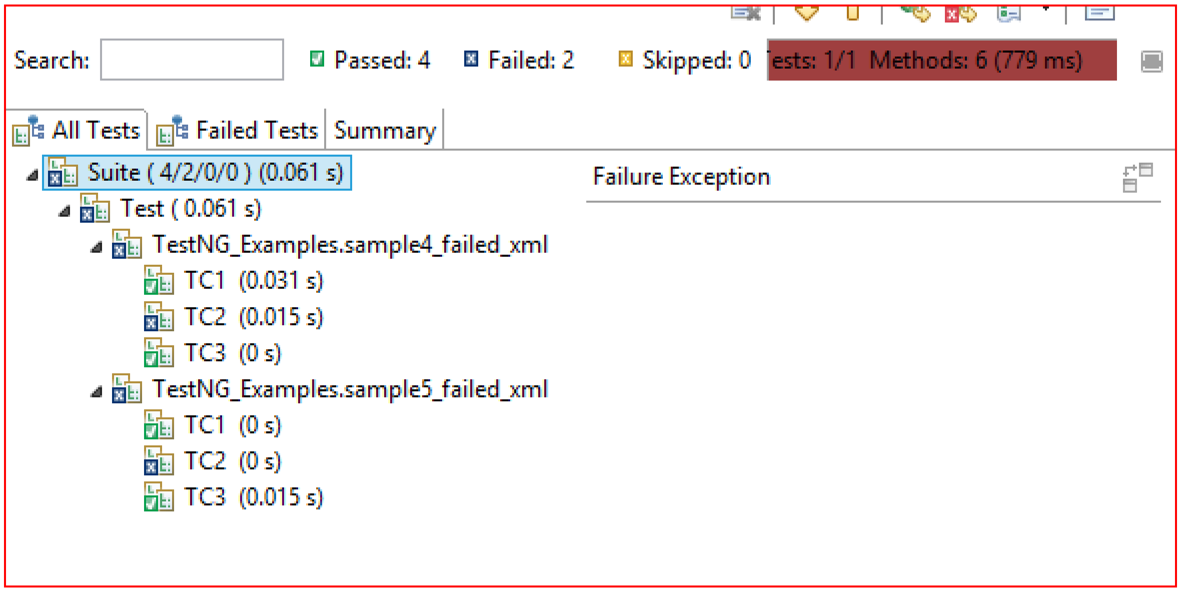
In this way we can execute fail testcases in TestNG class.

Reasons for fail TC

1. envirnment issue
2. script error
3. bug

```
package TestNG_Examples;

import org.testng.Assert;
import org.testng.Reporter;
import org.testng.annotations.Test;

public class sample4_failed_xml
{
	@Test
	public void TC1()
	{
		Reporter.log("running TC1", true);
	}

	@Test
	public void TC2()
	{
		Reporter.log("running TC2", true);

		Assert.fail();
	}
	@Test
	public void TC3()
	{
		Reporter.log("running TC3", true);
	}
}

package TestNG_Examples;

import org.testng.Assert;
import org.testng.Reporter;
import org.testng.annotations.Test;

public class sample5_failed_xml
{
	@Test
	public void TC1()
	{
		Reporter.log("running TC1", true);
	}

	@Test
	public void TC2()
	{
		Reporter.log("running TC2", true);

		Assert.fail();
	}

	@Test
	public void TC3()
	{
		Reporter.log("running TC3", true);
	}
}
```

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE suite SYSTEM "https://testng.org/testng-1.0.dtd">
<suite name="Suite">
  <test thread-count="5" name="Test">
    <classes>
      <class name="TestNG_Examples.sample4_failed_xml"/>
      <class name="TestNG_Examples.sample5_failed_xml"/>

    </classes>
  </test> <!-- Test -->
</suite> <!-- Suite -->
```

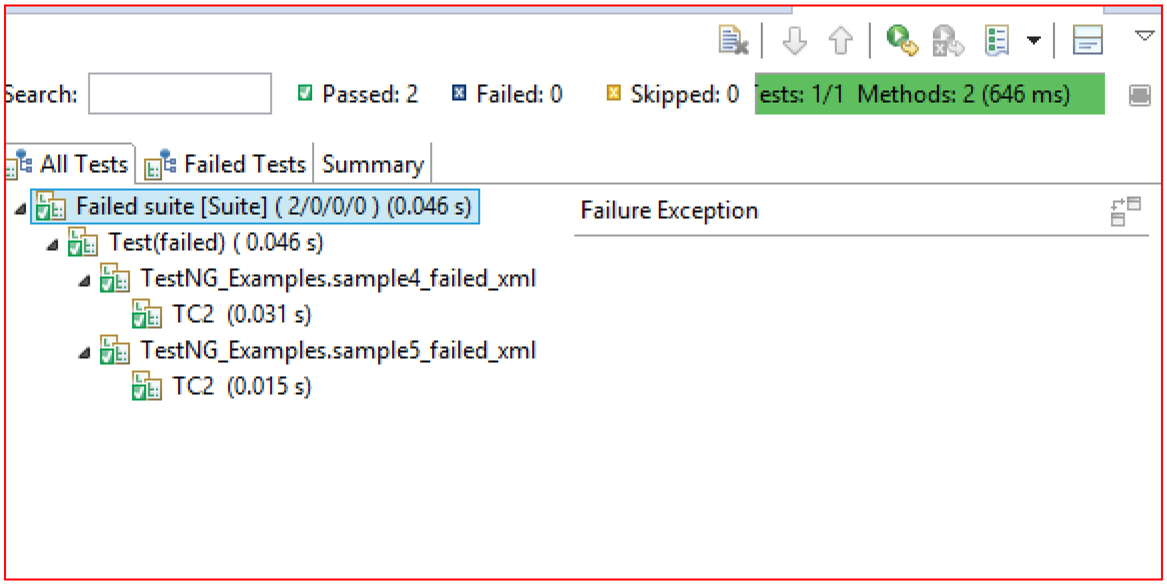


```
Search:            Passed: 4    Failed: 2    Skipped: 0   Tests: 1/1  Methods: 6 (779 ms)

All Tests | Failed Tests | Summary

Suite ( 4/2/0/0 ) (0.061 s)                          Failure Exception
  Test ( 0.061 s)
    TestNG_Examples.sample4_failed_xml
      TC1  (0.031 s)
      TC2  (0.015 s)
      TC3  (0 s)
    TestNG_Examples.sample5_failed_xml
      TC1  (0 s)
      TC2  (0 s)
      TC3  (0.015 s)
```

Consol

```
[RemoteTestNG] detected TestNG version 7.3.0
running TC1
running TC2
running TC3
running TC1
running TC2
running TC3

===============================================
Suite
Total tests run: 6, Passes: 4, Failures: 2, Skips: 0
===============================================
package TestNG_Examples;

import org.testng.Assert;
import org.testng.Reporter;
import org.testng.annotations.Test;

public class sample4_failed_xml
{
	@Test
	public void TC1()
	{
		Reporter.log("running TC1", true);
	}

	@Test
	public void TC2()
	{
		Reporter.log("running TC2", true);

		//Assert.fail();
	}

	@Test
	public void TC3()
	{
		Reporter.log("running TC3", true);
	}
}

package TestNG_Examples;

import org.testng.Assert;
import org.testng.Reporter;
import org.testng.annotations.Test;

public class sample5_failed_xml
{
	@Test
	public void TC1()
	{
			Reporter.log("running TC1", true);
		}

		@Test
		public void TC2()
		{
			Reporter.log("running TC2", true);

			//Assert.fail();
		}

		@Test
		public void TC3()
		{
			Reporter.log("running TC3", true);
		}
}
```

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE suite SYSTEM "https://testng.org/testng-1.0.dtd">
<suite guice-stage="DEVELOPMENT" name="Failed suite [Suite]">
  <test thread-count="5" name="Test(failed)">
    <classes>
      <class name="TestNG_Examples.sample4_failed_xml">
        <methods>
          <include name="TC2"/>
        </methods>
      </class> <!-- TestNG_Examples.sample4_failed_xml -->
      <class name="TestNG_Examples.sample5_failed_xml">
        <methods>
          <include name="TC2"/>
        </methods>
      </class> <!-- TestNG_Examples.sample5_failed_xml -->
    </classes>
  </test> <!-- Test(failed) -->
</suite> <!-- Failed suite [Suite] -->
```



```
Search:            Passed: 2    Failed: 0    Skipped: 0   Tests: 1/1  Methods: 2 (646 ms)

All Tests | Failed Tests | Summary
Failed suite [Suite] ( 2/0/0/0 ) (0.046 s)          Failure Exception
  Test(failed) ( 0.046 s)
    TestNG_Examples.sample4_failed_xml
      TC2  (0.031 s)
    TestNG_Examples.sample5_failed_xml
      TC2  (0.015 s)
```

Consol

```
[RemoteTestNG] detected TestNG version 7.3.0
running TC2
running TC2

===============================================
Failed suite [Suite]
Total tests run: 2, Passes: 2, Failures: 0, Skips: 0
===============================================
```

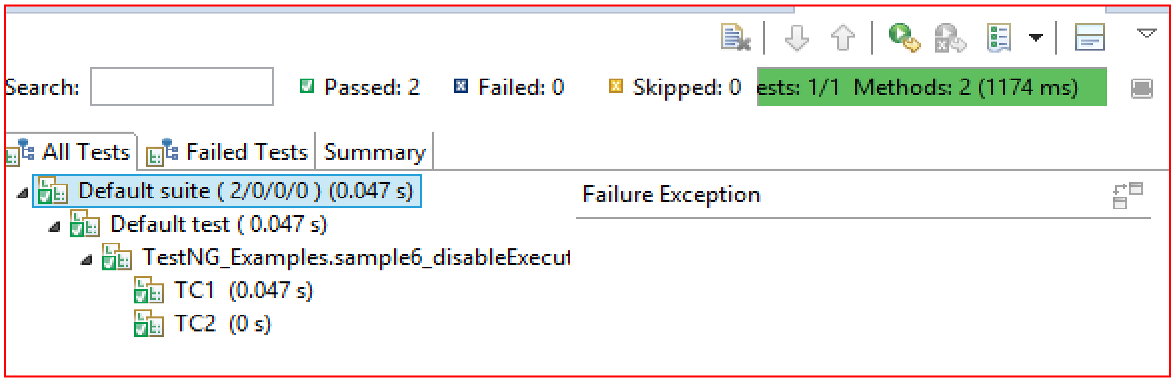
8. Disable TC Execution

1. from Test class using 'enabled=false' keyword
2. from suite using include/exclude keyword

1. from Test class using 'enabled=false' keyword

- **@Test(enabled=false)**

```
package TestNG_Examples;

import org.testng.Reporter;
import org.testng.annotations.Test;

public class sample6_disableTCExecutionFromTestClass
{
        @Test
        public void TC1()
        {
                Reporter.log("running TC1", true);
        }
        @Test
        public void TC2()
        {
                Reporter.log("running TC2", true);
        }
        @Test(enabled=false)
        public void TC3()
        {
                Reporter.log("running TC3", true);
        }
}
```



```
Search:            Passed: 2    Failed: 0    Skipped: 0  ests: 1/1  Methods: 2 (1174 ms)

All Tests | Failed Tests | Summary
◢ Default suite ( 2/0/0/0 ) (0.047 s)          Failure Exception
  ◢ Default test ( 0.047 s)
    ◢ TestNG_Examples.sample6_disableExecut
        TC1  (0.047 s)
        TC2  (0 s)
```

Consol

```
[RemoteTestNG] detected TestNG version 7.3.0
running TC1
running TC2
PASSED: TC1
PASSED: TC2
===============================================
    Default test
    Tests run: 2, Failures: 0, Skips: 0
===============================================
===============================================
Default suite
Total tests run: 2, Passes: 2, Failures: 0, Skips: 0
===============================================
```

2. from suite using include/exclude keyword

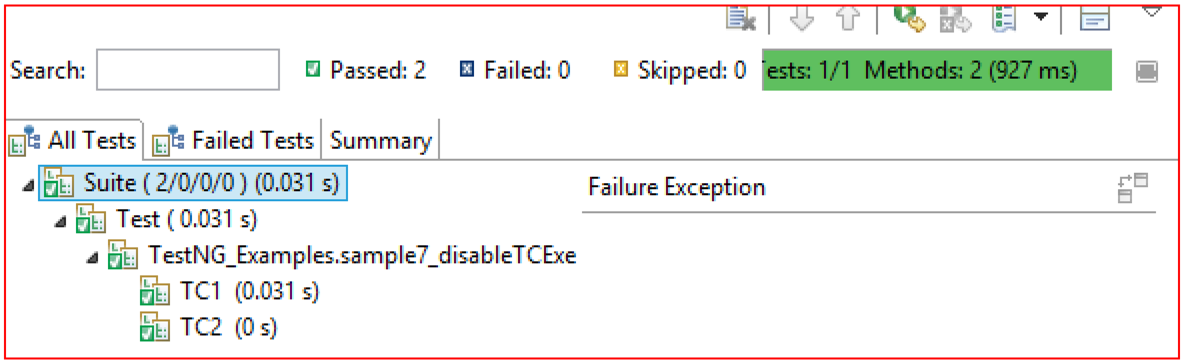
from suite using exclude keyword

- After class immediate to write sytax

```
 <methods>
    <exclude name="test method/test case name"></exclude>
                              Or
    <include name="test method/test case name"></include>
</methods>
```

```
package TestNG_Examples;

import org.testng.Reporter;
import org.testng.annotations.Test;

public class sample7_disableTCExecutionFromSuite
{
	@Test
	public void TC1()
	{
		Reporter.log("running TC1", true);
	}

	@Test
	public void TC2()
	{
		Reporter.log("running TC2", true);
	}

	@Test
	public void TC3()
	{
		Reporter.log("running TC3", true);
	}
}
```

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE suite SYSTEM "https://testng.org/testng-1.0.dtd">
<suite name="Suite">
  <test thread-count="5" name="Test">
    <classes>
      <class name="TestNG_Examples.sample7_disableTCExecutionFromSuite"/>
      <methods>
           <exclude name="TC3"/>
      </methods>
    </classes>
  </test> <!-- Test -->
</suite> <!-- Suite -->
```



```
Search:            Passed: 2    Failed: 0    Skipped: 0   ests: 1/1  Methods: 2 (927 ms)

All Tests | Failed Tests | Summary

Suite ( 2/0/0/0 ) (0.031 s)                      Failure Exception
  Test ( 0.031 s)
    TestNG_Examples.sample7_disableTCExe
      TC1  (0.031 s)
      TC2  (0 s)
```

Consol

```
[RemoteTestNG] detected TestNG version 7.3.0
running TC1
running TC2
===============================================
Suite
Total tests run: 2, Passes: 2, Failures: 0, Skips: 0
===============================================
```

from suite using include keyword

- After class name immediate to write sytax

<methods>

<exclude name="*test method/test case name*"></exclude>

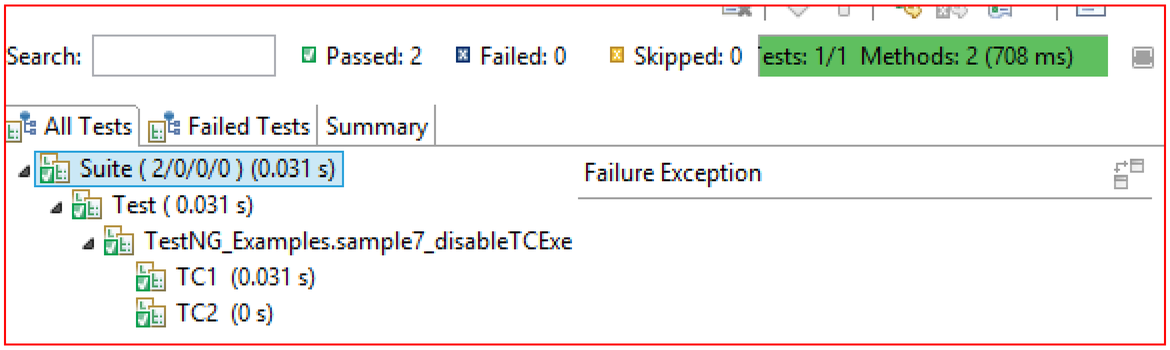
Or

<include name="*test method/test case name*"></include>

</methods>

```
package TestNG_Examples;

import org.testng.Reporter;
import org.testng.annotations.Test;

public class sample7_disableTCExecutionFromSuite
{
	@Test
	public void TC1()
	{
		Reporter.log("running TC1", true);
	}

	@Test
	public void TC2()
	{
		Reporter.log("running TC2", true);
	}

	@Test
	public void TC3()
	{
		Reporter.log("running TC3", true);
	}
}
```

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE suite SYSTEM "https://testng.org/testng-1.0.dtd">
<suite name="Suite">
  <test thread-count="5" name="Test">
    <classes>
      <class name="TestNG_Examples.sample7_disableTCExecutionFromSuite"/>
      <methods>
      <include name="TC1"/>
       <include name="TC2"/>
      </methods>
    </classes>
  </test> <!-- Test -->
</suite> <!-- Suite -->
```



Search: Passed: 2 Failed: 0 Skipped: 0 ests: 1/1 Methods: 2 (708 ms)

All Tests | Failed Tests | Summary

```
Suite ( 2/0/0/0 ) (0.031 s)
  Test ( 0.031 s)
    TestNG_Examples.sample7_disableTCExe
      TC1  (0.031 s)
      TC2  (0 s)
```

Failure Exception

Consol

```
[RemoteTestNG] detected TestNG version 7.3.0
running TC1
running TC2

===============================================
Suite
Total tests run: 2, Passes: 2, Failures: 0, Skips: 0
===============================================
```

9. Grouping of Test cases

- TestNG Groups allow you to perform groupings of different test methods
- Use syntax after test name immediately

```
<groups>
   <run>
     <include name="test method/test case name"></include>
                              Or
     <include name=" test method/test case name "></include>
   </run>
 </groups>
```

```
package TestNG_Examples;

import org.testng.Reporter;
import org.testng.annotations.Test;
```

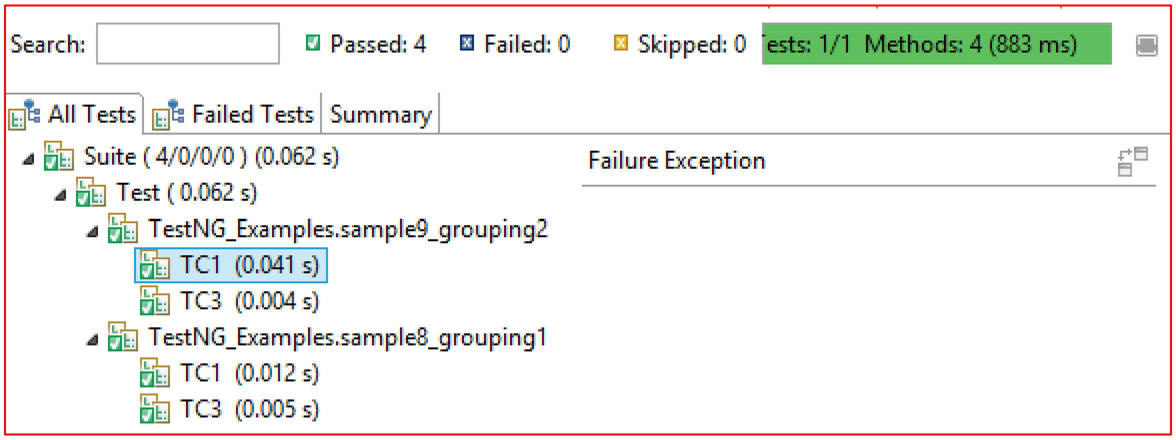
```
public class sample8_grouping1
{

	@Test(groups="order")
	public void TC1()
	{
		Reporter.log("running TC1", true);
	}
	@Test(groups="setting")
	public void TC2()
	{
		Reporter.log("running TC2", true);
	}
	@Test(groups="order")
	public void TC3()
	{
		Reporter.log("running TC3", true);
	}

	@Test(groups="dashboard")
	public void TC4()
	{
		Reporter.log("running TC4", true);
	}

	@Test(groups="dashboard")
	public void TC5()
	{
		Reporter.log("running TC5", true);
	}

	@Test(groups="fund")
	public void TC6()
	{
		Reporter.log("running TC6", true);
	}
}

package TestNG_Examples;
```

```
import org.testng.Reporter;
import org.testng.annotations.Test;

public class sample9_grouping2
{
	@Test(groups="order")
	public void TC1()
	{
		Reporter.log("running TC1", true);
	}

	@Test(groups="setting")
	public void TC2()
	{
		Reporter.log("running TC2", true);
	}

	@Test(groups="order")
	public void TC3()
	{
		Reporter.log("running TC3", true);
	}

	@Test(groups="dashboard")
	public void TC4()
	{
		Reporter.log("running TC4", true);
	}

	@Test(groups="dashboard")
	public void TC5()
	{
		Reporter.log("running TC5", true);
	}

	@Test(groups="fund")
	public void TC6()
	{
		Reporter.log("running TC6", true);
```

```
        }
}
```

@run with order name test cases / test method use include keyword

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE suite SYSTEM "https://testng.org/testng-1.0.dtd">
<suite name="Suite">
  <test thread-count="5" name="Test">
  <groups>
  <run>
  <include name="order"></include>
  </run>
  </groups>
    <classes>
      <class name="TestNG_Examples.sample9_grouping2"/>
      <class name="TestNG_Examples.sample8_grouping1"/>
    </classes>
  </test> <!-- Test -->
</suite> <!-- Suite -->
```



```
Search:    Passed: 4   Failed: 0   Skipped: 0  Tests: 1/1  Methods: 4 (883 ms)
All Tests | Failed Tests | Summary
Suite ( 4/0/0/0 ) (0.062 s)                     Failure Exception
  Test ( 0.062 s)
    TestNG_Examples.sample9_grouping2
      TC1  (0.041 s)
      TC3  (0.004 s)
    TestNG_Examples.sample8_grouping1
      TC1  (0.012 s)
      TC3  (0.005 s)
```

Consol

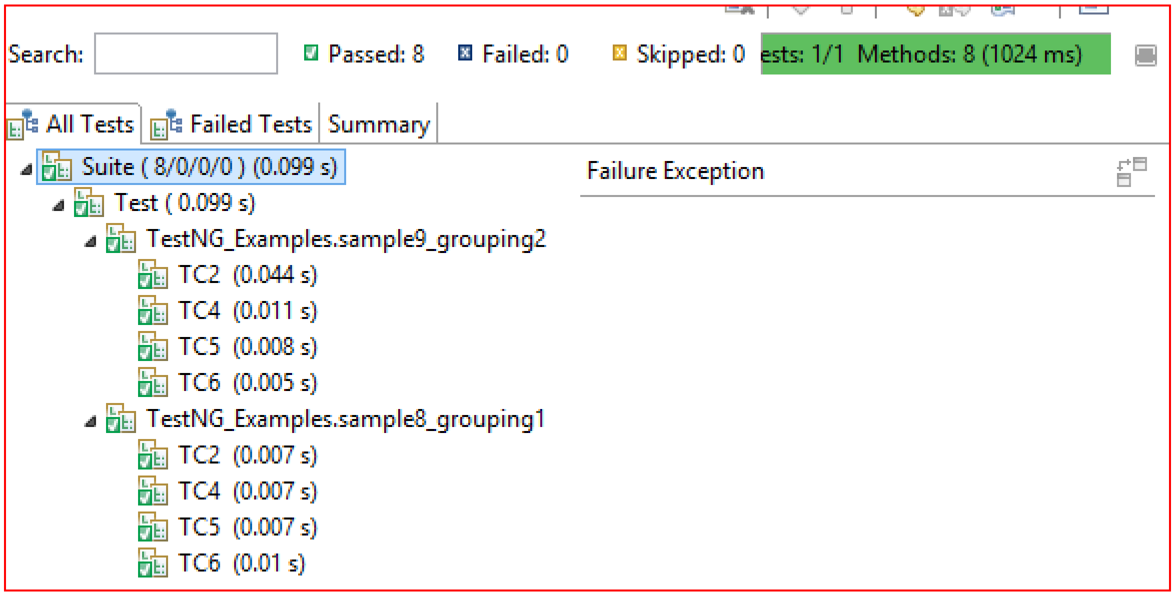
```
[RemoteTestNG] detected TestNG version 7.3.0
running TC1
running TC3
running TC1
running TC3
```

===

Suite
Total tests run: 4, Passes: 4, Failures: 0, Skips: 0

@run without order name test cases / test method—use exclude keyword

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE suite SYSTEM "https://testng.org/testng-1.0.dtd">
<suite name="Suite">
  <test thread-count="5" name="Test">
  <groups>
  <run>
  <exclude name="order"></exclude>
  </run>
  </groups>
    <classes>
      <class name="TestNG_Examples.sample9_grouping2"/>
      <class name="TestNG_Examples.sample8_grouping1"/>
    </classes>
  </test> <!-- Test -->
</suite> <!-- Suite -->
```



```
Search:            Passed: 8    Failed: 0    Skipped: 0  ests: 1/1  Methods: 8 (1024 ms)

All Tests | Failed Tests | Summary
Suite ( 8/0/0/0 ) (0.099 s)                      Failure Exception
  Test ( 0.099 s)
    TestNG_Examples.sample9_grouping2
      TC2  (0.044 s)
      TC4  (0.011 s)
      TC5  (0.008 s)
      TC6  (0.005 s)
    TestNG_Examples.sample8_grouping1
      TC2  (0.007 s)
      TC4  (0.007 s)
      TC5  (0.007 s)
      TC6  (0.01 s)
```

Consol
[RemoteTestNG] detected TestNG version 7.3.0
running TC2
running TC4

```
running TC5
running TC6
running TC2
running TC4
running TC5
running TC6

===============================================
Suite
Total tests run: 8, Passes: 8, Failures: 0, Skips: 0
===============================================
```

10. Parallel testing or parallel execution

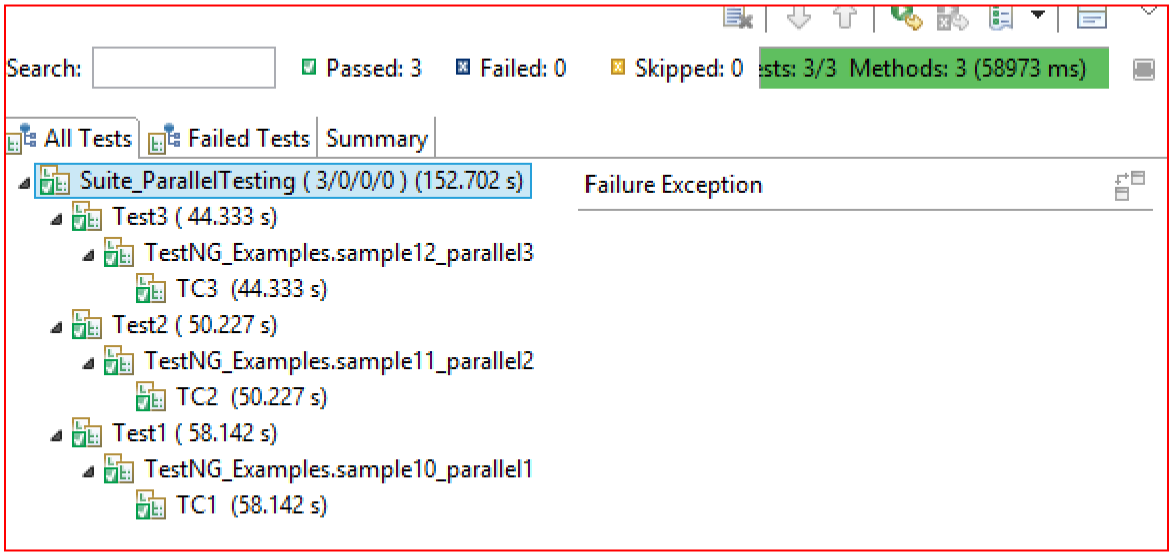
- Parallel testing or parallel execution, as the name suggests, is a **process** of **running** the test case **parallelly** rather than one **after** the other.
- In parallel testing, the program's multiple parts (or modules) execute together, **saving** the testers a lot of time and effort.
- Use in Test suite, **after** suite name immediate to write **parallel="tests"**.
- To create 3 test cases and run parallel and to create multiple test cases.
- 3 chrome browser open for 3 test cases, not 1 chrome browser multiple tab not open. To open new chrome browser each test case in parallel testing.

```
package TestNG_Examples;

import org.openqa.selenium.WebDriver;
import org.openqa.selenium.chrome.ChromeDriver;
import org.testng.annotations.Test;

public class sample10_parallel1
{
	@Test
	public void TC1() throws InterruptedException
	{
		//To open browser
		System.setProperty("webdriver.chrome.driver",
"C:\\Users\\HP\\Downloads\\chromedriver_win32\\chromedriver.exe");
		WebDriver driver = new ChromeDriver();
		//To maximize browser
		driver.manage().window().maximize();

		//To open flipkart
		driver.get("https://www.flipkart.com/");

		//wait 4sec after opening flipkart
		Thread.sleep(4000);

		//close
		driver.close();
	}
}
```

```
package TestNG_Examples;

import org.openqa.selenium.WebDriver;
import org.openqa.selenium.chrome.ChromeDriver;
import org.testng.annotations.Test;

public class sample11_parallel2
{
	@Test
	public void TC2() throws InterruptedException
	{
		//To open browser
		System.setProperty("webdriver.chrome.driver",
"C:\\Users\\HP\\Downloads\\chromedriver_win32\\chromedriver.exe");
		WebDriver driver = new ChromeDriver();

		//To maximize browser
		driver.manage().window().maximize();

		//To open facebook
		driver.get("https://www.facebook.com/");
		//wait 4sec after opening flipkart
		Thread.sleep(4000);

		//close
		driver.close();
	}
}


package TestNG_Examples;

import org.openqa.selenium.WebDriver;
import org.openqa.selenium.chrome.ChromeDriver;
import org.testng.annotations.Test;

public class sample12_parallel3
{
	@Test
	public void TC3() throws InterruptedException
	{
		//To open browser
		System.setProperty("webdriver.chrome.driver",
"C:\\Users\\HP\\Downloads\\chromedriver_win32\\chromedriver.exe");
		WebDriver driver = new ChromeDriver();

		//To maximize browser
		driver.manage().window().maximize();

		//To open flipkart
		driver.get("https://www.google.com/");

		//wait 4sec after opening flipkart
		Thread.sleep(4000);

		//close
		driver.close();
	}
}
```

Consol

```
===============================================
Suite_ParallelTesting
Total tests run: 3, Passes: 3, Failures: 0, Skips: 0
===============================================
```



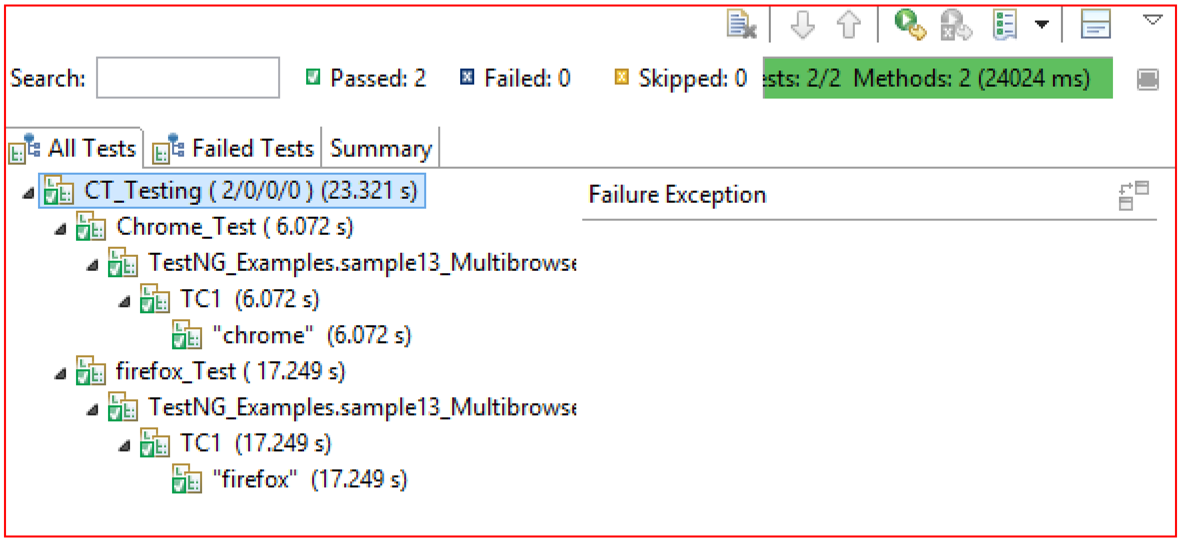
```
Search:            Passed: 3    Failed: 0    Skipped: 0   sts: 3/3  Methods: 3 (58973 ms)

All Tests | Failed Tests | Summary

Suite_ParallelTesting ( 3/0/0/0 ) (152.702 s)        Failure Exception
  Test3 ( 44.333 s)
    TestNG_Examples.sample12_parallel3
      TC3  (44.333 s)
  Test2 ( 50.227 s)
    TestNG_Examples.sample11_parallel2
      TC2  (50.227 s)
  Test1 ( 58.142 s)
    TestNG_Examples.sample10_parallel1
      TC1  (58.142 s)
```

11. Multi browser testing/C.T

- To use @Parameters("variableName / brwoserName") before taking @Test annotation in the test class.
- In the test suite declare parameter name="browserName / variableName " after test name.

```
package TestNG_Examples;

import org.openqa.selenium.WebDriver;

import org.openqa.selenium.chrome.ChromeDriver;

import org.openqa.selenium.firefox.FirefoxDriver;

import org.openqa.selenium.ie.InternetExplorerDriver;

import org.testng.annotations.Parameters;

import org.testng.annotations.Test;


public class sample13_Multibrowser_CT

{

	@Parameters("browserName")
	@Test
	public void TC1(String browserName)
	{
		WebDriver driver = null;
		if(browserName.equals("chrome"))
		{
			System.setProperty("webdriver.chrome.driver",
"C:\\Users\\HP\\Downloads\\chromedriver_win32\\chromedriver.exe");
			driver = new ChromeDriver();
		}
		else if (browserName.equals("firefox"))
		{
			System.setProperty("webdriver.gecko.driver",
"C:\\Users\\HP\\Downloads\\geckodriver.exe");
			driver = new FirefoxDriver();
		}
		else if (browserName.equals("ie"))
		{
			System.setProperty("webdriver.internetexplorer.driver",
"C:\\Users\\HP\\Downloads\\geckodriver.exe");
			driver = new InternetExplorerDriver();
		}
	}
}
```

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE suite SYSTEM "https://testng.org/testng-1.0.dtd">
<suite name="CT_Testing">
  <test thread-count="5" name="Chrome_Test">
  <parameter name="browserName" value="chrome"></parameter>
    <classes>
      <class name="TestNG_Examples.sample13_Multibrowser_CT"/>
    </classes>
  </test> <!-- Test -->

  <test thread-count="5" name="firefox_Test">
  <parameter name="browserName" value="firefox"></parameter>
    <classes>
      <class name="TestNG_Examples.sample13_Multibrowser_CT"/>
    </classes>
  </test> <!-- Test -->
</suite> <!-- Suite -->
```



```
Search:            Passed: 2    Failed: 0    Skipped: 0   ests: 2/2  Methods: 2 (24024 ms)

All Tests | Failed Tests | Summary

CT_Testing ( 2/0/0/0 ) (23.321 s)                    Failure Exception
  Chrome_Test ( 6.072 s)
    TestNG_Examples.sample13_Multibrowse
      TC1  (6.072 s)
        "chrome"  (6.072 s)
  firefox_Test ( 17.249 s)
    TestNG_Examples.sample13_Multibrowse
      TC1  (17.249 s)
        "firefox"  (17.249 s)
```

Consol

```
===============================================
CT_Testing
Total tests run: 2, Passes: 2, Failures: 0, Skips: 0
===============================================
```

12.Multi browser parallel testing

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE suite SYSTEM "https://testng.org/testng-1.0.dtd">
<suite name="CT_Testing" parallel="tests">
  <test thread-count="5" name="Chrome_Test">
  <parameter name="browserName" value="chrome"></parameter>
    <classes>
      <class name="TestNG_Examples.sample13_Multibrowser_CT"/>
    </classes>
  </test> <!-- Test -->

  <test thread-count="5" name="firefox_Test">
  <parameter name="browserName" value="firefox"></parameter>
    <classes>
      <class name="TestNG_Examples.sample13_Multibrowser_CT"/>
    </classes>
  </test> <!-- Test -->
</suite> <!-- Suite -->
```

13. TestNG advantages

1. It has different assertions that helps in checking the expected and actual results.
2. It allows to assign **priority** to test methods
3. It allows to define **dependency** of one test method over other method
4. It provide Detailed (HTML) **reports/ Emailable report**
5. It allows **grouping** of test methods into test class
6. TestNG provides **parallel execution** of test methods
7. TestNG provides **multibrowser/CT testing** of test methods

14. Difference between Junit and TestNG-Junit Old version of TestNG

Parameter	Junit	TestNG
Supports Annotation	It does not support advanced annotation.	It supports advanced annotation.
Parallel test execution	JUnit does not support to run parallel tests.	TestNG can run parallel tests.
Dependency tests	The dependency tests are missing in JUnit.	Dependency tests are present in TestNG.
Grouping tests	Grouping tests together is not possible in JUnit.	Tests can be grouped together and run parallel.

7. POM with DDF and TestNG

POM with DDF program
@To pen Kite application and verify PN & then logout

```
package POM_with_DDF_TestNG;

import org.openqa.selenium.WebDriver;
import org.openqa.selenium.WebElement;
import org.openqa.selenium.support.FindBy;
import org.openqa.selenium.support.PageFactory;

public class KiteLogin1Page
{
	//Declaration

	@FindBy(xpath="//input[@id='userid']") private WebElement un;
	@FindBy(xpath="//input[@id='password']") private WebElement pwd;
	@FindBy(xpath="//button[text()='Login ']") private WebElement login;

	//Initialization

	public KiteLogin1Page(WebDriver driver)
	{
		PageFactory.initElements(driver, this);
	}

	//Usage

	public void setKiteLogin1PageUsername(String username)
	{
		un.sendKeys(username);
	}

	public void setKiteLogin1PagePassword(String password)
	{
		pwd.sendKeys(password);
	}
	public void clickKiteLogin1PageLoginBtn()
	{
		login.click();
	}
}

package POM_with_DDF_TestNG;

import org.openqa.selenium.WebDriver;
import org.openqa.selenium.WebElement;
import org.openqa.selenium.support.FindBy;
import org.openqa.selenium.support.PageFactory;

public class KiteLogin2Page
{
	//Declaration

	@FindBy(xpath="//input[@id='pin']") private WebElement pin;
	@FindBy(xpath="//button[text()='Continue ']") private WebElement
cntBtn;

	//Initialization

	public KiteLogin2Page(WebDriver driver)
	{
		PageFactory.initElements(driver, this);
	}

	//Usage

	public void setKiteLogin2PagePin(String pinValue)
	{
		pin.sendKeys(pinValue);
	}

	public void clickKiteLogin2PageCntBtn()
	{
		cntBtn.click();
	}
}
```

```
package POM_with_DDF_TestNG;

import org.openqa.selenium.WebDriver;
import org.openqa.selenium.WebElement;
import org.openqa.selenium.support.FindBy;
import org.openqa.selenium.support.PageFactory;

public class KiteHomePage
{
	//Declaration
	//find dynamic xpath profile name change every user
	@FindBy(xpath="//div[@class='avatar']/span") private WebElement pn;

	//Initialization

	public KiteHomePage(WebDriver driver)
	{
		PageFactory.initElements(driver, this);
	}

	//Usage
	public void verifyKiteHomePagePN(String expPN)
	{
		String actPN = pn.getText();

		if(expPN.equals(actPN))
		{
			System.out.println("pass");
		}
		else
		{
			System.out.println("fail");
		}
	}

	public void clickKiteHomePagePN()
	{
		pn.click();
	}
}
```

```
package POM_with_DDF_TestNG;

import org.openqa.selenium.WebDriver;
import org.openqa.selenium.WebElement;
import org.openqa.selenium.support.FindBy;
import org.openqa.selenium.support.PageFactory;

public class KiteProfilePage
{
	//Declaration

	@FindBy(xpath="//a[text()=' Logout']") private WebElement logout;

	//Initialization
	public KiteProfilePage(WebDriver driver)
	{
		PageFactory.initElements(driver, this);
	}

	//Usage

	public void clickKiteProfilePageLogoutBtn()
	{
		logout.click();
	}
}

package POM_with_DDF_TestNG;

import java.io.FileInputStream;
import java.io.IOException;
import java.util.concurrent.TimeUnit;

import org.apache.poi.EncryptedDocumentException;
import org.apache.poi.ss.usermodel.Sheet;
import org.apache.poi.ss.usermodel.WorkbookFactory;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.chrome.ChromeDriver;

public class KiteLogoutTest
```

```
{
	public static void main(String[] args) throws InterruptedException, 
EncryptedDocumentException, IOException
	{
		//To load excel sheet
		FileInputStream file = new FileInputStream("D:\\Dec2020.xlsx");
		Sheet sh = WorkbookFactory.create(file).getSheet("DDF");
		
		//To open chrome browser
		System.setProperty("webdriver.chrome.driver", 
"C:\\Users\\HP\\Downloads\\chromedriver_win32\\chromedriver.exe");
		WebDriver driver = new  ChromeDriver();
		
		//To give implicitly wait
		driver.manage().timeouts().implicitlyWait(20, TimeUnit.SECONDS);
		
		//To maximize browser
		driver.manage().window().maximize();
		
		//To open / enter url of kite app
		driver.get("https://kite.zerodha.com/");
		
		//wait for 3sec after opening kite app
		Thread.sleep(3000);
		
		//To create object of pom class
		KiteLogin1Page login1 = new KiteLogin1Page(driver);
login1.setKiteLogin1PageUsername(sh.getRow(0).getCell(0).getStringCellValue()
);
login1.setKiteLogin1PagePassword(sh.getRow(0).getCell(1).getStringCellValue())
;
login1.clickKiteLogin1PageLoginBtn();
		
		KiteLogin2Page login2 = new KiteLogin2Page(driver);
login2.setKiteLogin2PagePin(sh.getRow(0).getCell(2).getStringCellValue());
		login2.clickKiteLogin2PageCntBtn();
		
		KiteHomePage home = new KiteHomePage(driver);
home.verifyKiteHomePagePN(sh.getRow(0).getCell(3).getStringCellValue());
		home.clickKiteHomePagePN();
```

```
                KiteProfilePage profile = new KiteProfilePage(driver);
                profile.clickKiteProfilePageLogoutBtn();
        }
}
```

Consol
pass

POM with DDF & TestNG

```
package POM_with_DDF_TestNG;

import org.openqa.selenium.WebDriver;
import org.openqa.selenium.WebElement;
import org.openqa.selenium.support.FindBy;
import org.openqa.selenium.support.PageFactory;

public class KiteLogin1Page
{
                //Declaration

@FindBy(xpath="//input[@id='userid']")        private        WebElement        un;
@FindBy(xpath="//input[@id='password']") private WebElement pwd;
@FindBy(xpath="//button[text()='Login ']") private WebElement login;

                //Initialization

                public KiteLogin1Page(WebDriver driver)
                {
                      PageFactory.initElements(driver, this);
                }

                //Usage

                public void setKiteLogin1PageUsername(String username)
                {
                    un.sendKeys(username);
                }

                public void setKiteLogin1PagePassword(String password)
                {
                    pwd.sendKeys(password);
                }

                public void clickKiteLogin1PageLoginBtn()
                {
                    login.click();
                }
}

package POM_with_DDF_TestNG;

import org.openqa.selenium.WebDriver;
import org.openqa.selenium.WebElement;
import org.openqa.selenium.support.FindBy;
import org.openqa.selenium.support.PageFactory;

public class KiteLogin2Page
{
                //Declaration

@FindBy(xpath="//input[@id='pin']") private WebElement pin;
@FindBy(xpath="//button[text()='Continue ']") private WebElement cntBtn;

                //Initialization

                public KiteLogin2Page(WebDriver driver)
                {
                    PageFactory.initElements(driver, this);
                }
```

```
			//Usage

			public void setKiteLogin2PagePin(String pinValue)
			{
				pin.sendKeys(pinValue);
			}

			public void clickKiteLogin2PageCntBtn()
			{
				cntBtn.click();
			}
}

package POM_with_DDF_TestNG;

import org.openqa.selenium.WebDriver;
import org.openqa.selenium.WebElement;
import org.openqa.selenium.support.FindBy;
import org.openqa.selenium.support.PageFactory;

public class KiteHomePage
{
			//Declaration
			//find dynamic xpath profile name change every user
@FindBy(xpath="//div[@class='avatar']/span") private WebElement pn;

			//Initialization

			public KiteHomePage(WebDriver driver)
			{
				PageFactory.initElements(driver, this);
			}

			//Usage
```

```
                public String getKiteHomePagePN()
                {
                     String actPN = pn.getText();

                     return actPN;
                }

                public void clickKiteHomePagePN()
                {
                     pn.click();
                }
}

package POM_with_DDF_TestNG;

import org.openqa.selenium.WebDriver;
import org.openqa.selenium.WebElement;
import org.openqa.selenium.support.FindBy;
import org.openqa.selenium.support.PageFactory;

public class KiteProfilePage
{
                //Declaration

@FindBy(xpath="//a[text()=' Logout']") private WebElement logout;

                //Initialization

                public KiteProfilePage(WebDriver driver)
                {
                     PageFactory.initElements(driver, this);
                }
                //Usage

                public void clickKiteProfilePageLogoutBtn()
                {
                    logout.click();
                }
}

package POM_with_DDF_TestNG;

import java.io.FileInputStream;
import java.io.IOException;
import java.util.concurrent.TimeUnit;

import org.apache.poi.EncryptedDocumentException;
import org.apache.poi.ss.usermodel.Sheet;
import org.apache.poi.ss.usermodel.WorkbookFactory;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.chrome.ChromeDriver;
import org.testng.Assert;
import org.testng.annotations.AfterClass;
import org.testng.annotations.AfterMethod;
import org.testng.annotations.BeforeClass;
import org.testng.annotations.BeforeMethod;
import org.testng.annotations.Test;

public class KiteLoginTestNGTest
{
                KiteLogin1Page login1;
                Sheet sh;
                KiteLogin2Page login2;
                KiteHomePage home;
                KiteProfilePage profile;
                WebDriver driver;
                //@BerforeClass pass one time activity
                @BeforeClass
public void openBrowser() throws EncryptedDocumentException, IOException,
InterruptedException
                {
                     //To load excel sheet
FileInputStream file = new
FileInputStream("C:\\Users\\HP\\Documents\\Dec2020.xlsx");
                     sh = WorkbookFactory.create(file).getSheet("testNG");

                     //To open chrome browser
                     System.setProperty("webdriver.chrome.driver",
"C:\\Users\\HP\\Downloads\\chromedriver_win32\\chromedriver.exe");
                     driver = new ChromeDriver();

                     //To give implicitly wait
driver.manage().timeouts().implicitlyWait(20, TimeUnit.SECONDS);

                     //To maximize browser
                     driver.manage().window().maximize();

                     //To open / enter url of kite app
                     driver.get("https://kite.zerodha.com/");

                     //wait for 3sec after opening kite app
                     Thread.sleep(3000);

                     //To create object of POM class
                     login1 = new KiteLogin1Page(driver);
                     login2 = new KiteLogin2Page(driver);
                     home = new KiteHomePage(driver);
                     profile = new KiteProfilePage(driver);
                }
```

```
		@BeforeMethod
		public void loginTOApp()
		{
login1.setKiteLogin1PageUsername(sh.getRow(0).getCell(0).getStringCellValue()
);
login1.setKiteLogin1PagePassword(sh.getRow(0).getCell(1).getStringCellValue())
;
login1.clickKiteLogin1PageLoginBtn();
login2.setKiteLogin2PagePin(sh.getRow(0).getCell(2).getStringCellValue());
login2.clickKiteLogin2PageCntBtn();
		}

		@Test
		public void verifyPN()
		{
			String actPN = home.getKiteHomePagePN();
			System.out.println("actPN="+actPN);
			String expPN = sh.getRow(0).getCell(3).getStringCellValue();
			System.out.println("expPN="+expPN);
			Assert.assertEquals(actPN, expPN,"act&exp results are diff");
		}

		@AfterMethod
		public void logoutFromApp()
		{
			home.clickKiteHomePagePN();
			profile.clickKiteProfilePageLogoutBtn();
		}

		@AfterClass
		public void closeBrowser()
		{
			driver.close();
		}
}
```

8. POM with DDF & TestNG with Base Class & Utility Class

```
package POM_with_DDF_TestNG_Base_UtilityClass;

import org.openqa.selenium.WebDriver;
import org.openqa.selenium.WebElement;
import org.openqa.selenium.support.FindBy;
import org.openqa.selenium.support.PageFactory;

public class KiteLogin1Page
{
                //Declaration

@FindBy(xpath="//input[@id='userid']") private WebElement un;
@FindBy(xpath="//input[@id='password']") private WebElement pwd;
@FindBy(xpath="//button[text()='Login ']") private WebElement login;

                //Initialization

                public KiteLogin1Page(WebDriver driver)
                {
                     PageFactory.initElements(driver, this);
                }

                //Usage

                public void setKiteLogin1PageUsername(String username)
                {
                     un.sendKeys(username);
                }

                public void setKiteLogin1PagePassword(String password)
                {
                     pwd.sendKeys(password);
                }
```

```
                    public void clickKiteLogin1PageLoginBtn()
                    {
                          login.click();
                    }
}

package POM_with_DDF_TestNG_Base_UtilityClass;

import org.openqa.selenium.WebDriver;
import org.openqa.selenium.WebElement;
import org.openqa.selenium.support.FindBy;
import org.openqa.selenium.support.PageFactory;

public class KiteLogin2Page
{
                    //Declaration

@FindBy(xpath="//input[@id='pin']") private WebElement pin;
@FindBy(xpath="//button[text()='Continue ']") private WebElement cntBtn;

                    //Initialization

                    public KiteLogin2Page(WebDriver driver)
                    {
                          PageFactory.initElements(driver, this);
                    }

                    //Usage

                    public void setKiteLogin2PagePin(String pinValue)
                    {
                          pin.sendKeys(pinValue);
                    }

                    public void clickKiteLogin2PageCntBtn()
```

```
			{
				cntBtn.click();
			}
}

package POM_with_DDF_TestNG_Base_UtilityClass;

import org.openqa.selenium.WebDriver;
import org.openqa.selenium.WebElement;
import org.openqa.selenium.support.FindBy;
import org.openqa.selenium.support.PageFactory;

public class KiteHomePage
{
			//Declaration

			//find dynamic xpath profile name change every user
@FindBy(xpath="//div[@class='avatar']/span") private WebElement pn;

			//Initialization

			public KiteHomePage(WebDriver driver)
			{
				PageFactory.initElements(driver, this);
			}

			//Usage

			public String getKiteHomePagePN()
			{
				String actPN = pn.getText();

				return actPN;
			}
			public void clickKiteHomePagePN()
			{
				pn.click();
			}
}

package POM_with_DDF_TestNG_Base_UtilityClass;

import org.openqa.selenium.WebDriver;
import org.openqa.selenium.WebElement;
import org.openqa.selenium.support.FindBy;
import org.openqa.selenium.support.PageFactory;

public class KiteProfilePage
{
			//Declaration

			@FindBy(xpath="//a[text()='	Logout']")	private	WebElement
logout;

			//Initialization
			public KiteProfilePage(WebDriver driver)
			{
				PageFactory.initElements(driver, this);
			}

			//Usage

			public void clickKiteProfilePageLogoutBtn()
			{
				logout.click();
			}
}
```

```
package POM_with_DDF_TestNG_Base_UtilityClass;

import java.util.concurrent.TimeUnit;

import org.openqa.selenium.WebDriver;
import org.openqa.selenium.chrome.ChromeDriver;

			//base class perform only browser related actions

public class baseClass
{
			static WebDriver driver;

			public static void browserOpen() throws InterruptedException
			{
				//To open chrome browser
				System.setProperty("webdriver.chrome.driver",
"D:\\WP\\Selenium_Dec_2020\\browsers\\chromedriver.exe");
				driver = new ChromeDriver();

				//To give implicitly wait
driver.manage().timeouts().implicitlyWait(20, TimeUnit.SECONDS);

				//To maximize browser
				driver.manage().window().maximize();

				//To open / enter url of kite app
				driver.get("https://kite.zerodha.com/");

				//wait for 3sec after opening kite app
				Thread.sleep(3000);
			}
}
```

```
package POM_with_DDF_TestNG_Base_UtilityClass;

import java.io.File;
import java.io.FileInputStream;
import java.io.IOException;

import org.apache.poi.EncryptedDocumentException;
import org.apache.poi.ss.usermodel.Sheet;
import org.apache.poi.ss.usermodel.WorkbookFactory;
import org.openqa.selenium.OutputType;
import org.openqa.selenium.TakesScreenshot;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.io.FileHandler;

public class utilityClass
{
	static Sheet sh;

	public static String getTestData(int rowIndex, int colIndex) throws
EncryptedDocumentException, IOException
	{
		//create generic function

		//To load excel sheet
		FileInputStream file = new
FileInputStream("D:\\WP\\Selenium_Dec_2020\\testData\\Dec2020.xlsx");
		sh = WorkbookFactory.create(file).getSheet("testNG");

String value = sh.getRow(rowIndex).getCell(colIndex).getStringCellValue();

		return value;
	}

			//for screenshot

public static void captureScreenshots(WebDriver driver,int tCID) throws
IOException
		{
	File srs=((TakesScreenshot)driver).getScreenshotAs(OutputType.FILE);
File dest = new
File("D:\\WP\\Selenium_Dec_2020\\Screenshot\\TestCaseNumber_"+tCID+".jpg"
);

                        FileHandler.copy(srs, dest);
                }
        }

package POM_with_DDF_TestNG_Base_UtilityClass;

import java.io.FileInputStream;
import java.io.IOException;
import java.util.concurrent.TimeUnit;

import org.apache.poi.EncryptedDocumentException;
import org.apache.poi.ss.usermodel.Sheet;
import org.apache.poi.ss.usermodel.WorkbookFactory;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.chrome.ChromeDriver;
import org.testng.Assert;
import org.testng.annotations.AfterClass;
import org.testng.annotations.AfterMethod;
import org.testng.annotations.BeforeClass;
import org.testng.annotations.BeforeMethod;
import org.testng.annotations.Test;

public class KiteLoginTestNGTest extends baseClass
{
        KiteLogin1Page login1;
        Sheet sh;
        KiteLogin2Page login2;
        KiteHomePage home;
        KiteProfilePage profile;
        int TCID;

        //@BerforeClass pass one time activity
        @BeforeClass
        public void openBrowser() throws EncryptedDocumentException,
IOException, InterruptedException
        {
```

```
		//to call static method in current class
		browserOpen();

		//To create object of POM class
		login1 = new KiteLogin1Page(driver);
		login2 = new KiteLogin2Page(driver);
		home = new KiteHomePage(driver);
		profile = new KiteProfilePage(driver);
	}

	@BeforeMethod
	public void loginTOApp() throws EncryptedDocumentException,
IOException
	{
		login1.setKiteLogin1PageUsername(utilityClass.getTestData(0,0));
		login1.setKiteLogin1PagePassword(utilityClass.getTestData(0,1));
		login1.clickKiteLogin1PageLoginBtn();
		login2.setKiteLogin2PagePin(utilityClass.getTestData(0,2));
		login2.clickKiteLogin2PageCntBtn();
	}

	@Test
	public void verifyPN() throws EncryptedDocumentException, IOException
	{
		TCID=100;

		String actPN = home.getKiteHomePagePN();
		System.out.println("actPN="+actPN);

		String expPN =utilityClass.getTestData(0,3);
		System.out.println("expPN="+expPN);

		Assert.assertEquals(actPN, expPN,"act&exp results are diff");

		utilityClass.captureScreenshots(driver, TCID);
	}

	@AfterMethod
	public void logoutFromApp()
	{
		home.clickKiteHomePagePN();
		profile.clickKiteProfilePageLogoutBtn();
	}

	@AfterClass
	public void closeBrowser()
	{
		driver.close();
	}
}
```

9. Property file Selenium

- Property file in selenium is use to store the URL, UN and PWD and then call & this important credential can't store in excel sheet. Excel sheet store only the test case data.
- Store the data in the property file in terms of keys and values
- To create a common function in the utility class to read data
- **How to create property file in selenium?**

Step 1) Creating a properties file in eclipse

- Right-click on the main project folder and Select New-> Other->select General -> File and click on 'Next' button->
- Provide a valid file name with the extension '.properties' on the new file resource window and click on 'Finish' button

Step 2) Storing data onto properties file

- Data is stored in properties file in the form of key-value pairs, with the key being unique across the file.
- Open file in Eclipse and store some data

  eg- URL= https://kite.zerodha.com/

Step 3) Reading data from properties file

```
		Properties obj = new Properties();
FileInputStream objfile = new
FileInputStream(System.getProperty("user.dir")+"\\credentials.properties");
		obj.load(objfile);
		String value= obj.getProperty("URL");
```

Only change base and utility class other are same

```
package POM_with_DDF_TestNG_Base_UtilityClass;

import java.io.IOException;
import java.util.concurrent.TimeUnit;

import org.openqa.selenium.WebDriver;
import org.openqa.selenium.chrome.ChromeDriver;

			//base class perform only browser related actions

public class baseClass
{
			static WebDriver driver;

public static void browserOpen() throws InterruptedException, IOException
			{
				//To open chrome browser
				System.setProperty("webdriver.chrome.driver",
"D:\\WP\\Selenium_Dec_2020\\browsers\\chromedriver.exe");
				driver = new ChromeDriver();

				//To give implicitly wait
driver.manage().timeouts().implicitlyWait(20, TimeUnit.SECONDS);

				//To maximize browser
				driver.manage().window().maximize();

				//To open / enter url of kite app
				//driver.get("https://kite.zerodha.com/");

				//call utility class for property file purpose
				driver.get(utilityClass.getPropertyFiledata("URL"));
				//wait for 3sec after opening kite app
				Thread.sleep(3000);
			}
}

package POM_with_DDF_TestNG_Base_UtilityClass;

import java.io.File;
import java.io.FileInputStream;
import java.io.IOException;
import java.util.Properties;

import org.apache.poi.EncryptedDocumentException;
import org.apache.poi.ss.usermodel.Sheet;
import org.apache.poi.ss.usermodel.WorkbookFactory;
import org.openqa.selenium.OutputType;
import org.openqa.selenium.TakesScreenshot;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.io.FileHandler;

public class utilityClass
{
			static Sheet sh;

			public static String getTestData(int rowIndex, int colIndex) throws
EncryptedDocumentException, IOException
			{
				//create generic function

				//To load excel sheet
				FileInputStream file = new
FileInputStream("D:\\WP\\Selenium_Dec_2020\\testData\\Dec2020.xlsx");
				sh = WorkbookFactory.create(file).getSheet("testNG");

String value = sh.getRow(rowIndex).getCell(colIndex).getStringCellValue();
```

```
                        return value;
                }

                                //for screenshot

public static void captureScreenshots(WebDriver driver,int tCID) throws
IOException
                        {
File srs=((TakesScreenshot)driver).getScreenshotAs(OutputType.FILE);

                                File dest = new
File("D:\\WP\\Selenium_Dec_2020\\Screenshot\\TestCaseNumber_"+tCID+".jpg"
);

                                FileHandler.copy(srs, dest);
                        }

public static String getPropertyFiledata(String key) throws IOException
                        {
                                Properties obj = new Properties();
FileInputStream objfile = new
FileInputStream(System.getProperty("user.dir")+"\\file.properties");
                                        obj.load(objfile);
                                        String value= obj.getProperty(key);

                                        return value;
                        }
                }
```

10.How to capture screenshot of only failed test cases?

- To use listener(interface) to get test case status pass, failed or skip.
- To get test case status we need to use listener(interface) in TestNG

- To check status of test case pass, failed and skip we need to use if else to compare the status.
- If status is pass no need to capture screenshot, if test case result s failed so that time to use if else and call utility class method capture screenshot.
- Each test case is execute or complete that time it goes to @After Method and to apply the condition those test cases is run to get status of that run method and compare status pass or fail.
- If it is pass no problem but it is fail so that time to pass the code capture screenshot.
- To add this code in the @AfterMethod, before that we have to declare which listener we have to use here because multiple listener are present in TestNG
- To get the status of the test case for that purpose we need to use ITestResult inside the constructor of @AfterMethod it is a interface or listener. This is give the status and store purpose we need to take one object immediate ITestResult.
- In this object to store the result pass, fail and skip. Result store in this object so by using this result or object we need to use if else statement.
- Inside the if to call the getStatus() after the object dot then to get the status pass, fail and skip

```
package POM_with_DDF_TestNG_Base_UtilityClass;

import java.io.FileInputStream;
import java.io.IOException;
import java.util.concurrent.TimeUnit;

import org.apache.poi.EncryptedDocumentException;
import org.apache.poi.ss.usermodel.Sheet;
import org.apache.poi.ss.usermodel.WorkbookFactory;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.chrome.ChromeDriver;
import org.testng.Assert;
import org.testng.ITestResult;
import org.testng.annotations.AfterClass;
import org.testng.annotations.AfterMethod;
import org.testng.annotations.BeforeClass;
import org.testng.annotations.BeforeMethod;
import org.testng.annotations.Test;

public class KiteLoginTestNGTest extends baseClass
{
			KiteLogin1Page login1;
			Sheet sh;
			KiteLogin2Page login2;
			KiteHomePage home;
			KiteProfilePage profile;
			int TCID;

			//@BerforeClass pass one time activity
			@BeforeClass
			public void openBrowser() throws EncryptedDocumentException,
IOException, InterruptedException
			{
				//to call static method in current class
				browserOpen();

				//To create object of POM class
				login1 = new KiteLogin1Page(driver);
				login2 = new KiteLogin2Page(driver);
				home = new KiteHomePage(driver);
				profile = new KiteProfilePage(driver);
			}

			@BeforeMethod
			public void loginTOApp() throws EncryptedDocumentException,
IOException
			{

			login1.setKiteLogin1PageUsername(utilityClass.getTestData(0,0));
			login1.setKiteLogin1PagePassword(utilityClass.getTestData(0,1));
				login1.clickKiteLogin1PageLoginBtn();
				login2.setKiteLogin2PagePin(utilityClass.getTestData(0,2));
				login2.clickKiteLogin2PageCntBtn();
			}

			@Test
			public void verifyPN() throws EncryptedDocumentException,
IOException
			{
				TCID=400;

				String actPN = home.getKiteHomePagePN();
				System.out.println("actPN="+actPN);

				String expPN =utilityClass.getTestData(0,3);
				System.out.println("expPN="+expPN);

				Assert.assertEquals(actPN, expPN,"act&exp results are diff");

				//utilityClass.captureScreenshots(driver, TCID);
			}

//			@AfterMethod
//			public void logoutFromApp()
//			{
//				home.clickKiteHomePagePN();
//				profile.clickKiteProfilePageLogoutBtn();
//			}

			//failed testcase capture screenshot
			@AfterMethod
			public void logoutFromApp(ITestResult result) throws IOException
			{
```

```
                    if(result.getStatus()==ITestResult.FAILURE)
                    {
                            utilityClass.captureScreenshots(driver, TCID);
                    }

                    home.clickKiteHomePagePN();
                    profile.clickKiteProfilePageLogoutBtn();
              }
              @AfterClass
              public void closeBrowser()
              {
                    driver.close();
              }
}
```

11.Maven project:
How to create Maven projet?
File→New→Project→expand Maven→select Maven Project→Next→Next→enter group id=package name→enter artifact id=project name →finish

Group id: package name
Artifact id: project name

1 jar file- 1 dependency

Maven create slandered structure & no need to add manually jar file also to maintain proper version
src/main/java--- all Pom classes/ library files
src/test/java--- All test classes
Maven dependencies folder
pom.xml-- https://mvnrepository.com/--to add dependency
Maven Create folder also possible as per our requirement
Browser
Screenshot
Testdata

11.Log Generation

- To **generate** the log in selenium framework we **need** to make **use** of **log4j** jar file.
- Log4j is an **open source** logging framework(API)
- By **using** log4j we can **store** selenium automation **flow logs** in file or database.
- Log4j has **3 principle** components
    1. Loggers
    2. Appenders
    3. Layout

1. **Loggers**
   It is **responsible** for logging information. We use Logger's methods to **generate** log statements.

2. **Appenders**
   Apache log4j provides Appender objects which are primarily **responsible** for printing logging messages to **different** destinations such as consoles, files, sockets, NT event logs, etc.
   Use to **deliver** log events to their destination.
   Use to **write** logs in a file.

3. **Layout**
   Layout class and overrides the format() method to **structure** the logging information **according** to a supplied pattern.
   It is **responsible** for logging information in different styles.

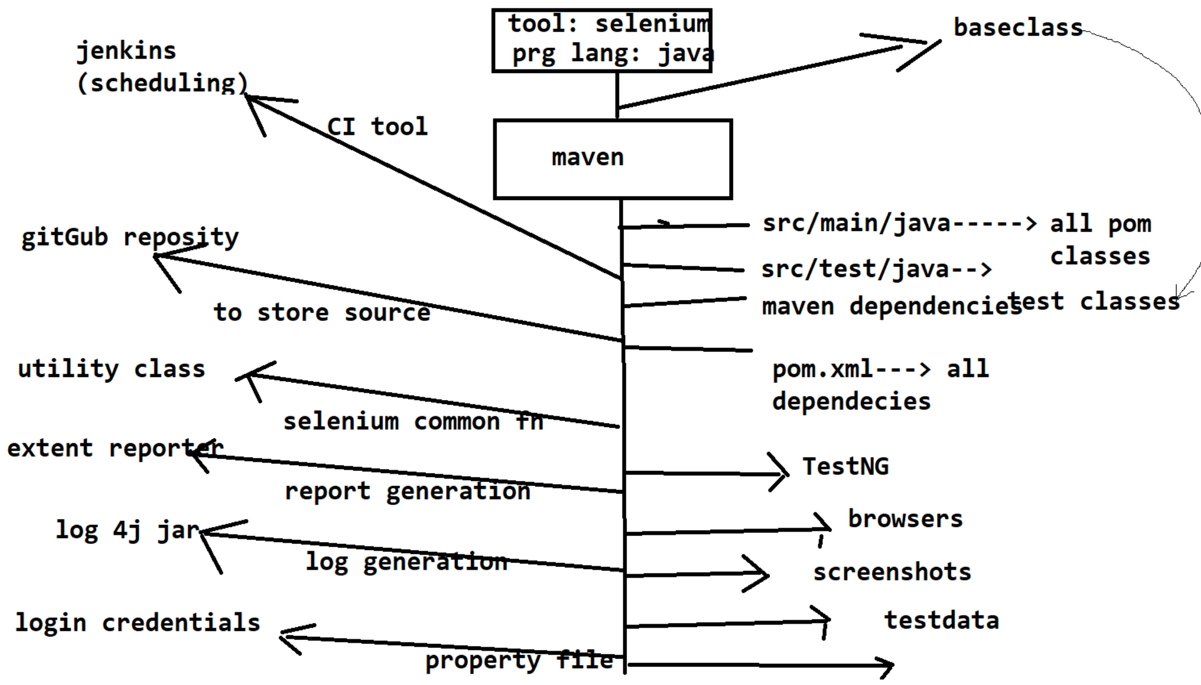
12.Extend Reporter
What are Extent Reports?

- ExtentReports is an **open-source** reporting **library** useful for test automation. It can be easily **integrated** with major testing frameworks **like** JUnit, NUnit, TestNG, etc. These reports are **HTML** documents that depict results as **pie charts**. They also **allow** the **generation** of custom logs, snapshots, and other customized details.

- Extent reporter is a customized **HTML** report which can be **integrated** into selenium webdriver **using** TestNG framework.
- **Advantages**

1. Customizable **HTML** report **with** stepwise & pie chart representation
2. **Display** the **time taken** for test case execution in report
3. **Each test step** can be associated with a **screenshot**
4. Easily **integrated** with TestNG
5. Multiple test cases **runs** can be **within** suite can be tracked easily

13.Framework Explanation



How to explain your selenium framework?

In our framework we use the automation tool as a selenium and programming language java.

In our project we use the maven project. Inside maven project we store all the pom processing , src/main/java- in that we store the all the pom classes, src/test/java- in that we store the all the test classes, maven dependencies- to automatically create a jar file when we add the dependency in the pom.xml , pom.xml- to add/contains all the dependencies like selenium, testNG, apache poi etc.

To design a test class we use our project java unit supporting framework like testNG. So the main advantage to use testNG in our framework like by using testNG we can run/check multiple test classes at the same time by using test suite, it can generate email able report, by using testNG we can perform the parallel execution also compatibility testing so for that reason we use the testNG in our framework.

In our maven project there are different folders like browser, screenshot, and test data etc. so in the browser folder we store the chromedriver.exe file, geckodriver.exe file etc. Then in the screenshot folder to store a capture screenshot when our test script is fail and in the test data folder we store the excel sheet file in that excel sheet file we store test data.

In our project one super most class is present that is a base class. This base class is inherited means extends to each and every test class.

To store the important login credential of application we use property file in our framework. Important login credential of applications like url, un, and pwd etc.

also to generate the logs we use the log4j jar files. When we run the test class so every time logs are store in the separate folder.

To generate the report we use the Extent reporter in our selenium framework.

In our framework there is one important class that is utility class. When we create the common system related function likes fetching data from excel sheet, capture screenshot also handling iframe and webtable. Create a generic function in that class. To create selenium related common function in that utility class.

In our project we use the github as a version control system. By using this github we store the all the daily source code in to central repository.

In project or framework we use the continuous integration tool like Jenkins the main use of this for scheduling purpose. It is also configure to github repository to continuous monitor.

11.GitHub Repository

Create local Repository
right click on project-->Team-->Share project--select checkbox-->select checkbox-->Create repository-->Finish

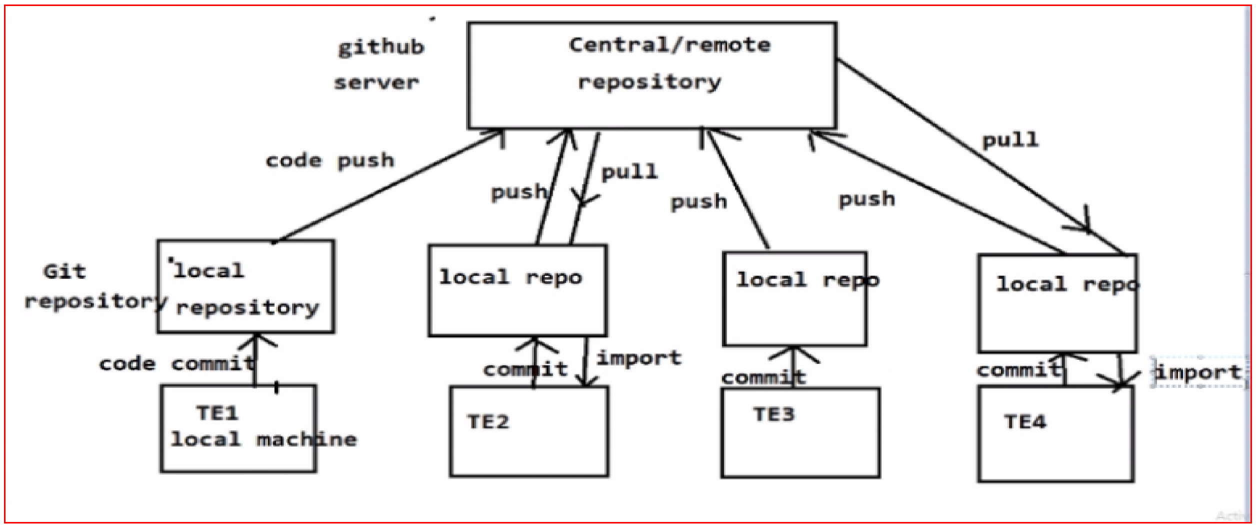
Commit source code from local machine to local repository
right click on project-->Team-->Add to index-->right click on project-->Team-->Commit--->add commit message-->commit

Push source code from local repository(git) to remote repository(bitbucket/github)
right click on project-->Team-->remote-->push-->Enter URL--> Enter UN & PWD-->next-->source ref-->master-->add Specification-->finish

Clone repository
Quick Access-->Open git repository window--> clone a git repository--> Enter repository URL,UN & Pwd--> Next-->select branch & Next--> Finish

Move Project from local repo to working directory
File-->import-->Git-->Project from git-->Existing local repo-->select project-->Finish

Who create central repository?
Project Manager or Team Lead and provide some credential to test engineer like url, username and password.



Interview IMP questions

Total modules: 15 to 20 -- 15

Pom class in each module: 20-40 -- 30

Total pom classes in project: 400 to 600 ---450

Test cases in each module: 70-100 -- 80

Test cases in each Test class: 5 to 10 (8 avg) --8

Test classes in each module: 10-20 -- 10

Test classes in project : 10* 15 : 150 to 200

Total no of test script in project: 1600 to 2000 test scripts in project

Test script per day: depends on complexity of Test script (1 to 4) (avg 2)

test script per month (20 working days): avg (40)

src/main/java- All pom classes

src/test/java- All test classes

src/main/java or src/main/java - library files/config files

Investment Banking(Kite Zeroda) Project Scenarios

1. Quantity field should be by default 1
2. when we click on stock it should show buy ,sell,view chart and other details
3. wishlist--add/remove

4. while selecting order if we select CNC (cash and carry) order ,we can hold the stock for long term --manual
5. While Selecting order,if we select MIS(margine intraday sqare) it should automatically exit at 3-20 pm.
6. if you try to place order ,but you don't have fund in your account then it should display insufficient fund msg
7. if you forget to logout account,at end of every day it should automatically logged out --manual/automation
8. when we placed the order by selecting market price ,price field should be disabled
9. when we placed limit order price field should be editable
10. when we placed bracket order(BO) then stoploss, target, trailing Stop loss should enable
11. when we placed cover order (co) only Stop loss field enabled
12. price field should accept value in multiple of 0.05 (eg 55.05,55,56.15 etc accepted) (55.06,55.07 not accepted)
13. after placing order ,in orders tab it will show Executed/pending/rejected orders
14. in portfolio tab , should show holdings and position of stock.
15. after clicking on user id tab it should show customer details
16. in options quantity field should accept in Lott eg (25 for banknifty,75 for nifty, depends on stock)
17. while adding fund in your demat account ,after clicking on add fund button , different payment options should enabled
18. when we click on withdraw button , withdrawal amount field and proceed button should enabled.
19. when we click on recent withdrawal option,it should show our withdrawal activities
20. when we click on AMO (after market order) option, we should able to placed order after market timing--manual
21. at time of order placing, BSE Or NSE only one should enabled
22. when we click on chart it should show different formats of chart (eg 1d,5d,1M etc) d- day,M- month
23. application should show same price of stock as per NSE or bse ,if we open an application in different platforms or machines, mobile etc ---CT

24. if we placed withdrawal request, ammount should be credit next day before 1 pm (if no hollyday) otherwise it should credit next of hollyday--manual
25. while withdrawing funds ,it should show withdrawable ammount.

OOPs concept in Selenium Framework

1. **Interface:**

   WebDriver driver = new ChromeDriver();

   In this statement WebDriver is nothing but interface in selenium.

2. **UPCASTING:**

   WebDriver driver = new ChromeDriver();

   above statement is nothing but UPCASTING in selenium.

3. **INHERITANCE**

- We **create** a Base Class in the Framework to **initialize** WebDriver interface, WebDriver waits, Property files etc., in the Base Class.
- We **extend** the Base Class in Tests Class. That is nothing but Inheritance in Selenium Framework.

4. **POLYMORPHISM**

   Combination of overloading and overriding is known as Polymorphism.

5. **METHOD OVERLOADING**

- We use **implicit wait** in Selenium. Implicit wait is an **example** of overloading. In Implicit wait we use different time stamps such as SECONDS, MINUTES, HOURS etc.,
- A class having multiple methods with same name but different parameters is called **Method Overloading**.
- In the **iframe** we use method overloading concept in our project. Inside the frame we pass the different argument like frame id, frame name, frame index, & frame webelement. Means same method name but different argrument

  eg. driver.switchTo().frame(); String name, int index

6. **METHOD OVERRIDING**

- **Declaring** a method in **child class** which is **already** present in the **parent class** is called Method Overriding.
- Examples are **get and navigate methods** of different drivers in Selenium.

7. **ENCAPSULATION**

- **All** the **POM classes** in a framework are an **example** of Encapsulation.
- In POM classes, we **declare** the data members using @FindBy and **initialization** of data members will be done using Constructor to **utilize** those in methods.
- Encapsulation is a **mechanism** of **binding code and data together** in a **single unit.**
- Encapsulation is the **process** of **wrapping up code and data together** in a **single unit**. It is used to **hide** the data of a class from another class.
- Encapsulation can be **achieved** when you **declare** all **variables** as **private** and a **public method** in a class to get the values of the variable.

8. **ABSTRACTION**

- In Page Object Model design pattern, we **write** locators (such as id, name, xpath etc.,) in a Page Class.
- We **utilize** these locators in pom class but we **can't see** these locators in the **tests.** Literally we **hide** the **locators** from the **tests.**
- Abstraction is the methodology of **hiding** the implementation of internal details and **showing** the functionality to the users.

Scroll Up & Down

```
package Scroll;

import org.openqa.selenium.JavascriptExecutor;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.chrome.ChromeDriver;

public class ScrollUpDown
{
	public static void main(String[] args) throws InterruptedException
	{
		//To Open Browser

	System.setProperty("webdriver.chrome.driver","C:\\Users\\HP\\Downloads\\
chromedriver_win32 (1)\\chromedriver.exe");
		WebDriver driver = new ChromeDriver();

		//To Wait
		Thread.sleep(3000);

		//Enter Url
		driver.get("http://demo.guru99.com/test/guru99home/");

		//To Maximize
		driver.manage().window().maximize();

		//wait
		Thread.sleep(3000);

		//Scroll down- 2nd parameter positive value
//		JavascriptExecutor js = (JavascriptExecutor)driver;
//		js.executeScript("window.scrollBy(0,1000)");

	      ((JavascriptExecutor)driver).executeScript("window.scrollBy(0,100)");
		//wait
		Thread.sleep(3000);

		//Scroll up- 2nd parameter negative value

		//((JavascriptExecuter)driver).executeScript("window.scrollBy(0,-
500)");
	}
}
```

Scroll Right & Left

```
package Scroll;

import org.openqa.selenium.JavascriptExecutor;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.chrome.ChromeDriver;

public class ScrollRightLeft
{
	public static void main(String[] args) throws InterruptedException
	{
		//To Open Browser

	System.setProperty("webdriver.chrome.driver","C:\\Users\\HP\\Downloads\\
chromedriver_win32 (1)\\chromedriver.exe");
		WebDriver driver = new ChromeDriver();

		//To Wait
		Thread.sleep(3000);

		//Enter Url
		driver.get("http://demo.guru99.com/test/guru99home/");

		//To Maximize
		driver.manage().window().maximize();

		//wait
		Thread.sleep(3000);

		//Scroll right- 1st parameter positive value


	((JavascriptExecutor)driver).executeScript("window.scrollBy(300,0)");

		//wait
		Thread.sleep(3000);

		//Scroll left- 1st parameter negative value

		//((JavascriptExecuter)driver).executeScript("window.scrollBy(-
300,0)");
	}
}
```

Scroll up & down in particular element

```
package Scroll;

import org.openqa.selenium.By;
import org.openqa.selenium.JavascriptExecutor;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.WebElement;
import org.openqa.selenium.chrome.ChromeDriver;

public class scrollIntoView
{
	public static void main(String[] args) throws InterruptedException
	{

		System.setProperty("webdriver.chrome.driver",
        "C:\\Users\\HP\\Downloads\\chromedriver_win32 (1)\\chromedriver.exe");
            WebDriver driver = new ChromeDriver();

            JavascriptExecutor js = (JavascriptExecutor) driver;

            // Launch the application
            driver.get("http://demo.guru99.com/test/guru99home/");

            // Find element by link text and store in variable "Element"
            WebElement ele = driver.findElement(By.linkText("Linux"));
            Thread.sleep(3000);

            // This will scroll the page till the element is found
            js.executeScript("arguments[0].scrollIntoView();", ele);
        }
    }
```

Jenkins :-

- Jenkins is open Source Continuous integration (CI) tool.
- It is cross platform can be used on windows, Linux, mac OS.
- Jenkins main use is to Schedule a particular Job & monitor any Job or application Status
- It fires pre configured actions when a particular Step occurs.

* features
  1. It generates list of all changes done in the repository
  2. can be Configure to email.
  3. we Can Save Test Report & execution history.
  4. Jenkins supports maven for building & testing a project in CI.

Chrone pattern

*	*	*	*	*
minutes in Hour (0-59)	hours in 1 day (0-23)	day of month (1-31)	month in year	day of week (0-7) 0 & 7 both Sunday

eg. every day, every month of year, at 15th min of 13th hr.

15 13 * * *

eg. every 5 min

*/5 * * * *

eg every day at 8 hr

0 8 * * *

* ~~BitBcket~~ BitBucket :-

Bitbucket is a web-based version Control repository for Source Code & development project.
use to keep track of changes in the code.

* Version Control System :-
  ~~It is like~~ ~~[illegible]~~ if more than 1 people are working on Same document at the Same time, then it ~~[illegible]~~ ~~[illegible]~~ eliminates Confusion. (conflict