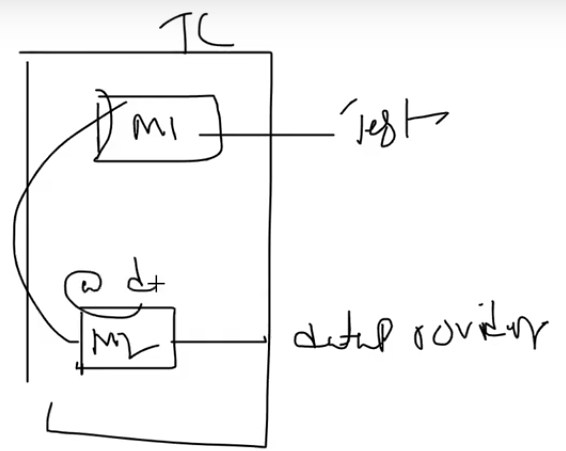
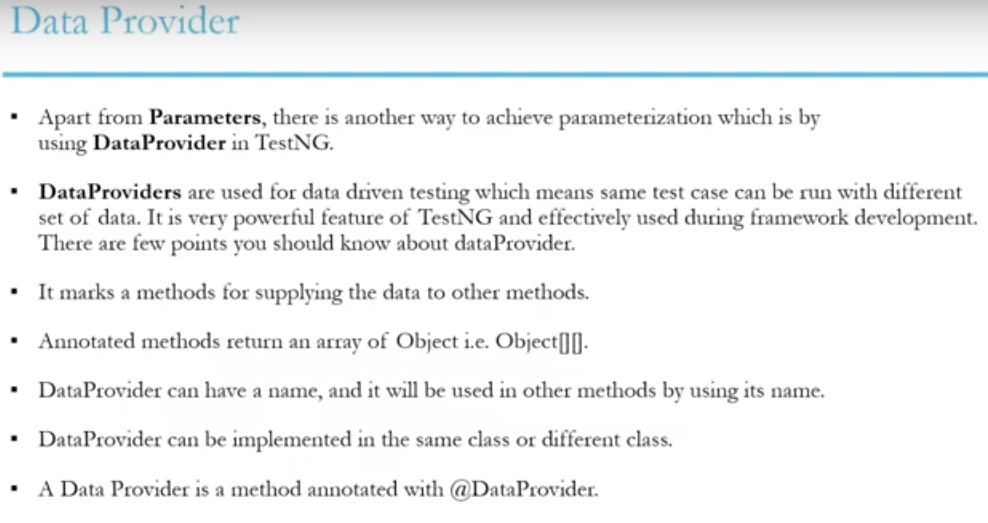
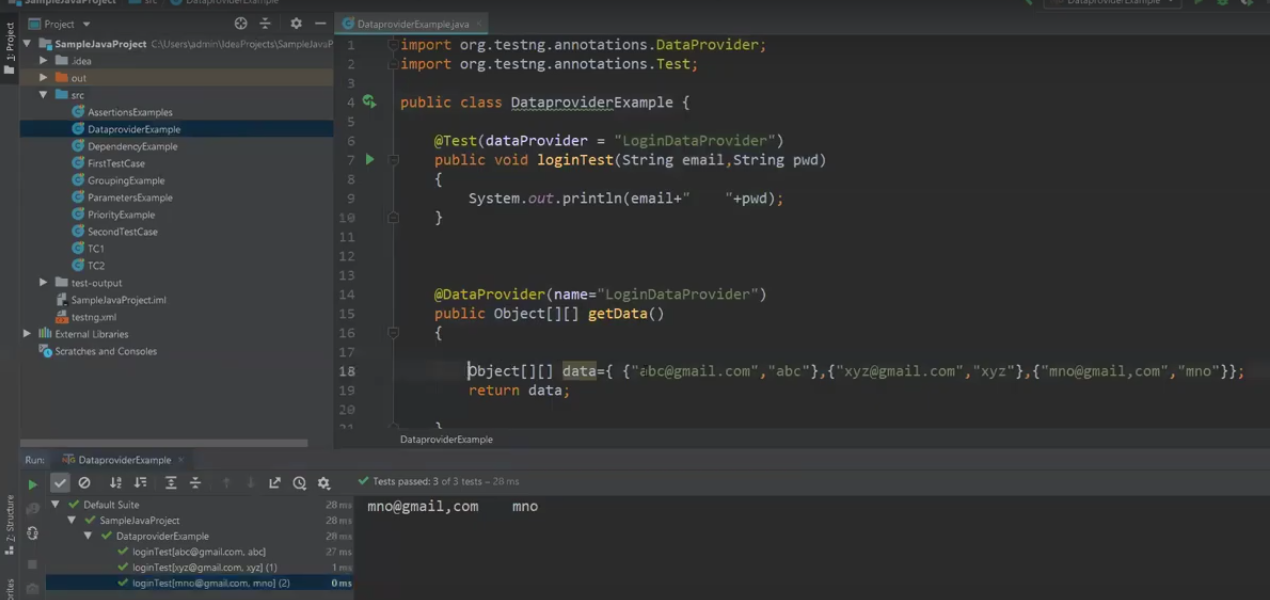
<https://www.youtube.com/watch?v=O8yP672fogE>

http://www.allinoneblogs.com/selenium-tutorials/testng/access-data-from-excel-sheet/

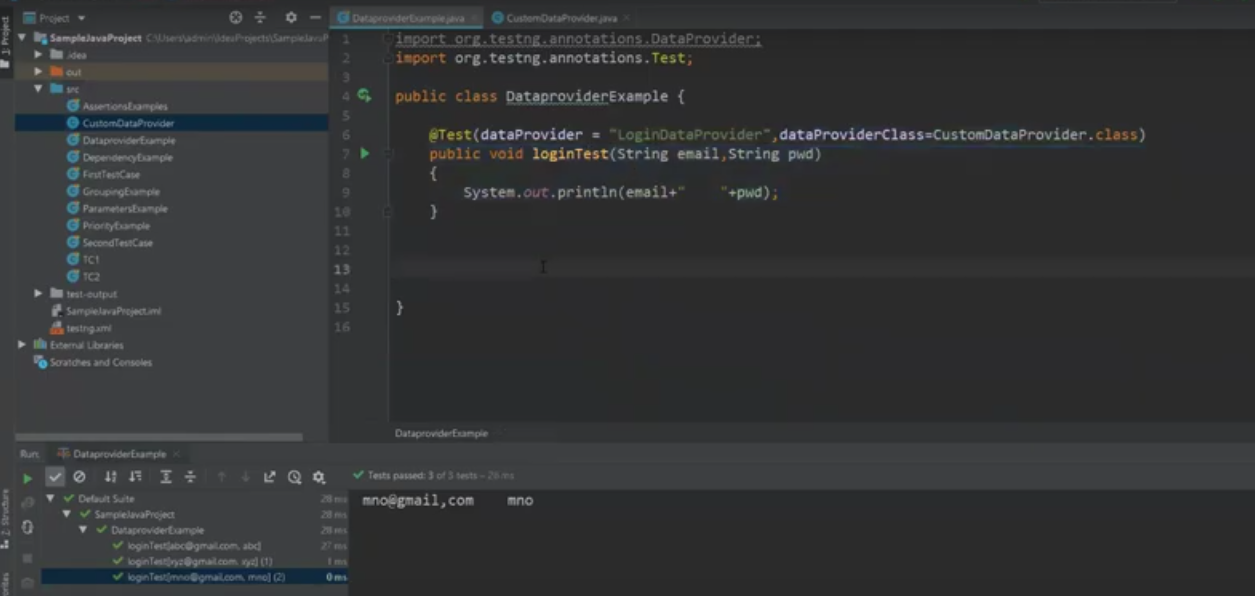
# Tutorial 9:TestNG with intelliJ IDE| DataProvider in TestNG | Parameterization







NOW if my DataProvider method is coming from a different class, how to use it?

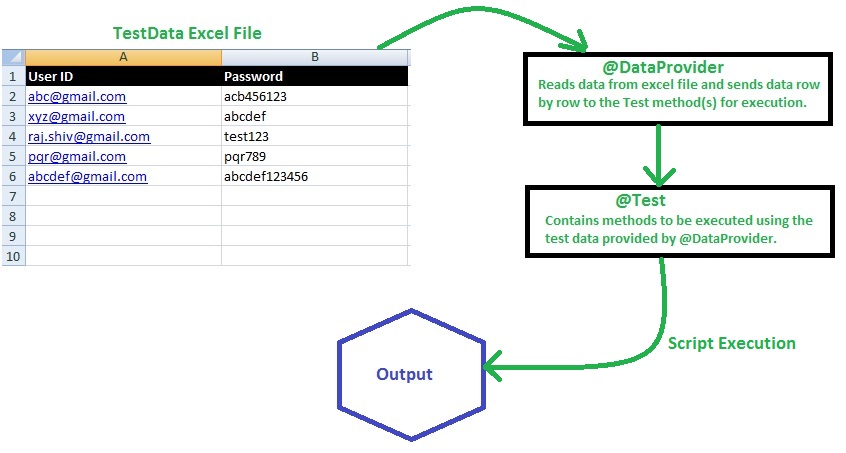


<http://www.allinoneblogs.com/selenium-tutorials/testng/access-data-from-excel-sheet/>

# TestNG – 16 || Access data from Excel sheet using DataProvider.

June 12, 2019 [Ashok Kumar](http://www.allinoneblogs.com/author/allinoneblogs/) [Leave a comment](http://www.allinoneblogs.com/selenium-tutorials/testng/access-data-from-excel-sheet/#respond)

We can use an Excel sheet as a source of input. We can **access data from excel sheet using DataProvider annotation** and pass arguments through @Test methods. Mainly this approach is used in Data Driven Framework.

[](http://www.allinoneblogs.com/wp-content/uploads/2017/10/Data-Driven-Framework.jpg)**Data Driven Framework**

**Data Driven Framework:** A data-driven framework is used to run the same script for a huge amount of test data from any source like properties files, excel files etc. On the other hand, a **Key Driven framework** is based on keywords which are used to control the execution of the script without modifying the actual code. A tester has to just set the keywords which he/she wants to execute during a review cycle. Nowadays, the hybrid framework is used which is the combination of Keyword Driven and Data Driven Framework.

Please refer [Simple example of Key Driven Framework using the excel sheet](http://www.allinoneblogs.com/java-tutorials/file-handling/simple-example-of-key-driven-framework-using-excel-sheet-in-seleniumjava/)

.

# ****DataProvider****

A data provider is defined with @DataProvider annotation. This annotation contains only one attribute its name. If the name is not supplied, the data provider’s name automatically defaults to the method’s name. A data provider returns an array of objects as mentioned below:  
The return type of methods using “@DataProvider(name=”testdata”)” should be Object[Row][Col].  
**Row –** defines the number of rows in excel files  
        – It also defines the total number of times test get repeat and it treats it as individual test case in TestNg  
**Col –** defines number of columns in excel file  
       – defines total number of parameters in the test data

**@DataProvider(name=“excelData”)**  
public Object[ ][ ] readExcel()  
{  
           Object[ ][ ] obj=new Object[2][2];  
           return obj;  
}

**@Test(dataProvider=“excelData”)**  
Public void testMethod(String arg1, String arg2)  
{  
   System.out.println(arg1);  
   System.out.println(arg2);  
}

## ****Selenium Code:****

import java.io.File;

import java.io.FileInputStream;

import java.io.IOException;

import org.apache.poi.xssf.usermodel.XSSFSheet;

import org.apache.poi.xssf.usermodel.XSSFWorkbook;

import org.openqa.selenium.By;

import org.openqa.selenium.WebDriver;

import org.openqa.selenium.chrome.ChromeDriver;

import org.openqa.selenium.support.ui.ExpectedConditions;

import org.openqa.selenium.support.ui.WebDriverWait;

import org.testng.Assert;

import org.testng.annotations.BeforeClass;

import org.testng.annotations.DataProvider;

import org.testng.annotations.Test;

public class TestCase

{

public static WebDriver driver;

public static String targetURL="https://gmail.com";

@BeforeClass

public void launchURL()

{

System.setProperty("webdriver.chrome.driver", "exe\\chromedriver.exe");

driver=new ChromeDriver();

driver.get(targetURL);

}

@DataProvider(name="excelData")

public Object[][] readExcel() throws IOException

{

File file=new File("data\\testData.xlsx");

FileInputStream fis=new FileInputStream(file);

XSSFWorkbook workbook= new XSSFWorkbook(fis);

XSSFSheet sheet=workbook.getSheet("Sheet1");

int totalRows=sheet.getLastRowNum();

int totalColums=sheet.getRow(0).getPhysicalNumberOfCells();

// Read data from excel and store the same in the Object Array.

Object obj[][]=new Object[totalRows][totalColums];

for(int i=0;i<totalRows;i++)

{

obj[i][0]=sheet.getRow(i+1).getCell(0).toString();

obj[i][1]=sheet.getRow(i+1).getCell(1).toString();

}

return obj;

}

@Test(dataProvider="excelData")

public void validateUser(String userName,String password) throws InterruptedException

{

By userNameXpath=By.xpath(".//\*[@id='identifierId']");

By nextBtnXpath=By.xpath(".//\*[@id='identifierNext']");

By passwordXpath=By.xpath(".//\*[@id='password']//input[contains(@name,'password')]");

By passwordNextBtnXpath=By.xpath(".//\*[@id='passwordNext']");

By searchXpath=By.xpath("//input[contains(@aria-label,'Search')]");

wait(userNameXpath);

driver.findElement(userNameXpath).click();

driver.findElement(userNameXpath).clear();

driver.findElement(userNameXpath).sendKeys(userName);

driver.findElement(nextBtnXpath).click();

wait(passwordXpath);

driver.findElement(passwordXpath).click();

driver.findElement(passwordXpath).sendKeys(password);

driver.findElement(passwordNextBtnXpath).click();

try

{

wait(searchXpath);

String windowTitle=driver.getTitle();

if(windowTitle.contains(userName))

{

Assert.assertTrue(true);

logout();

switchAccount();

}

else

{

Assert.assertTrue(false);

}

}

catch(Exception e)

{

switchAccount();

Assert.assertTrue(false);

}

}

public void wait(By locator)

{

WebDriverWait wait=new WebDriverWait(driver, 10);

wait.until(ExpectedConditions.visibilityOfElementLocated(locator));

}

public void logout() throws InterruptedException

{

By myProfileXpath=By.xpath("//\*[contains(@href,'SignOutOptions')]");

By logoutBtnXpath=By.xpath("//\*[contains(@href,'Logout')]");

driver.findElement(myProfileXpath).click();

wait(logoutBtnXpath);

driver.findElement(logoutBtnXpath).click();

}

public void switchAccount()

{

By switchAccountXpath=By.xpath("//div[contains(@aria-label,'Switch account')]");

By useOtherAccountXpath=By.xpath("//\*[@id='identifierLink']");

wait(switchAccountXpath);

driver.findElement(switchAccountXpath).click();

wait(useOtherAccountXpath);

driver.findElement(useOtherAccountXpath).click();

}

}

