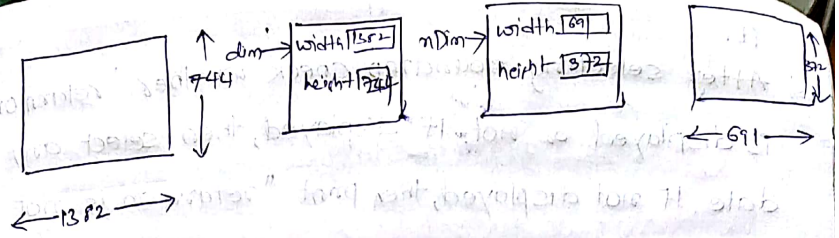


Write a note on various getmethods (or) WAP to demonstrate

getAttribute()
getText()

```
psvm
{
String url = "https://www.magentocom";
ChromeDriver driver = new ChromeDriver();
driver.manage().window().maximize();
driver.manage().timeouts().implicitlyWait(20, TimeUnit.SECONDS);
driver.get(url);
WebElement myAcct = driver.findElement(By.LinkText("My Account"));
System.out.println(myAcct.getTagName());
System.out.println(myAcct.getText());
System.out.println(myAcct.getAttribute("href"));
```

o/p: a
My Account
https://www.magento.com

Write a script to demonstrate isSelected() & isDisplayed()
(or)
Write a note on boolean functions

Scenario

→ Open the chrome browser and navigate to clearTrip.com website
→ Check whether round trip is selected or not. If yes, display a msg "round trip is selected". If not, then select

Scanned by CamScanner

it.

→ After selecting roundtrip check whether 'return on' is displayed or not. If displayed, then select any date. If not displayed, then print "return on is not displayed".

```
psvm
{
String url = "https://www.cleartrip.com/";
ChromeDriver driver = new ChromeDriver();
driver.manage().window().maximize();
driver.manage().timeouts().implicitlyWait(20,TimeUnit.seconds);
driver.get(url);
WebElement roundtrip = driver.findElement(By.id("RoundTrip"));
if (roundtrip.isSelected()==true)
{
 System.out.println("Roundtrip is selected");
}
else
{
 roundtrip.click();
}
WebElement rDate = driver.findElement(By.id("Retampdate"));
if (rDate.isDisplayed()==true)
{
 rDate.sendKeys ("Wed, 26 Dec, 2018");
}
else
{
 System.out.println("return on is not displayed");
}
}
}
```

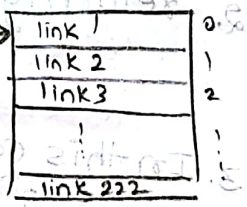
Scanned by CamScanner

Write a script to demonstrate isEnabled()

```
psvm
{
String url = "https://www.account.magento.com/customer/account/create/";

ChromeDriver driver = new ChromeDriver();
driver.manage().window().maximize();
driver.manage().timeouts().implicitlyWait(20,TimeUnit.seconds);
driver.get(url);
WebElement register = driver.findElement(By.id("registerSubmit"));
System.out.println(register.isEnabled());
Thread.sleep(30000);
s.o.p(register.isEnabled());
```

Write a script to display the count of total no of links in the magento application. After displaying total no of links, display the coorresponding linktexts associated with all the links.




```
psvm
{
String url = "https://www.magento.com/";
ChromeDriver driver = new ChromeDriver();
driver.manage().window().maximize();
driver.manage().timeouts().implicitlyWait(20,TimeUnit.SECONDS);
driver.get(url);
List<WebElement> links = driver.findElements(By.tagName("a"));
int size = links.size();                    // 219 (less than 222 bcoz, some links are hidden)
system.out.println(size);
for(int i=0; i<size; i++)
{
s.o.p(links.get(i).getText());              // some blank spaces appear, bcoz some links have no linkText
}
```

Scanned by CamScanner

Note: ① To get the return type automatically, press

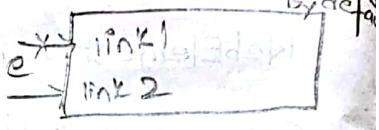
" Cntrl 2 L "

② To comment, select & press cntrl shft /

The above program can also be written by 'for each' loop as shown below.

```
for(WebElement e : Links)
{
  s.o.p(e.getText());
}
```

pick one link and store in e of type Webelement (starts from 0 by default)



Differences between findElement() & findElements()

1. It is used to find one single webElement.	1. It is used to find multiple webelements.
2. return type is WebElement	2. Return type is List<WebElement>
3. In this case, if the webelement is not identified, then NoSuchElement exception is displayed.	3. In this case, if the web element is not identified, [] is displayed (empty list)

Write a script to count the total no of links present in the magento application. Iterate through all the links and when MyAccount link is found, click on it.

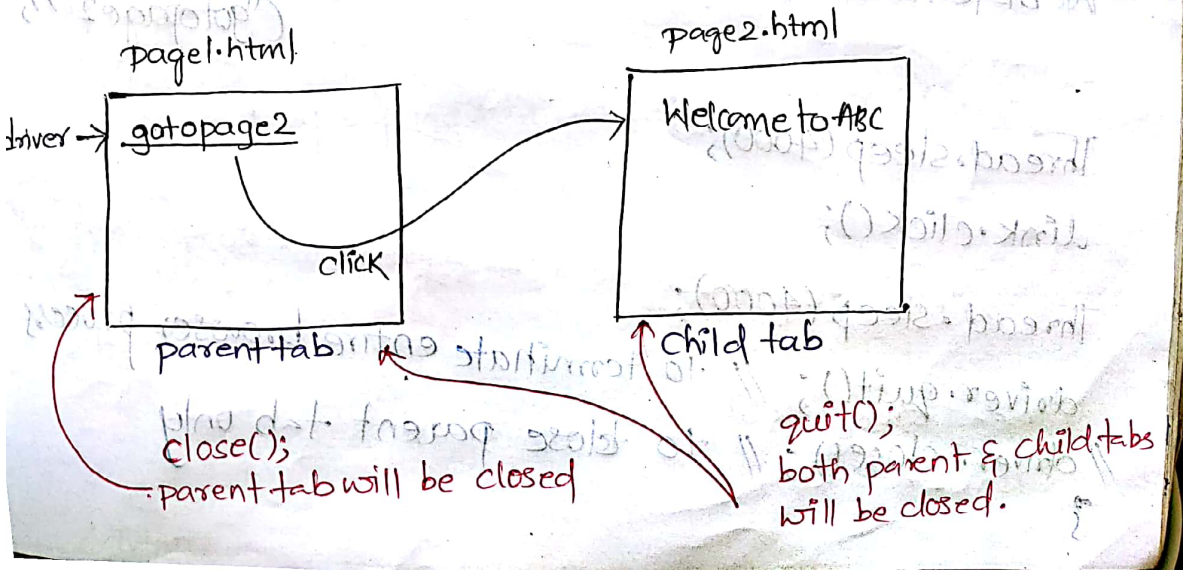
```
psvm(String args[]) throws InterruptedException
{
String url = "https://www.magento.com/";
```

Scanned by CamScanner

```
ChromeDriver driver = new ChromeDriver();
driver.manage().window().maximize();
driver.manage().timeouts().implicitlyWait(20, TimeUnit.seconds);
driver.get(url);
List<WebElement> links = driver.findElements(By.tagName("a"));
int size = links.size();
S.O.P("total num of links"+size);
for(WebElement e: links)
{
 String text=e.getText();
  if(text.contains("MyAccount"))
   {
     e.click();
     break;
   }
}
```

Differences b/w close() & quit()

1. Incase of close(), only the parent tab will be closed.
   Incase of quit(), both parent & child tabs would be closed.
2. In the industry, using quit() is recommended after completing all the tasks because quit() will terminate the entire process.



Scanned by CamScanner

page1.html

```
<html>
  <body>
    <a href = "page2.html" target = "blank">gotopage2</a>
  </body>
</html>
```

opens new tab

page2.html

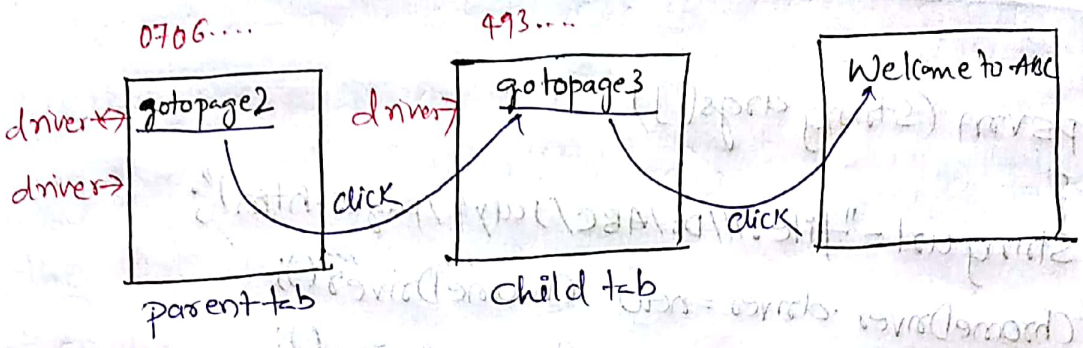
```
<html>
  <body>
    <h1> Welcome to ABC </h1>
  </body>
</html>
```

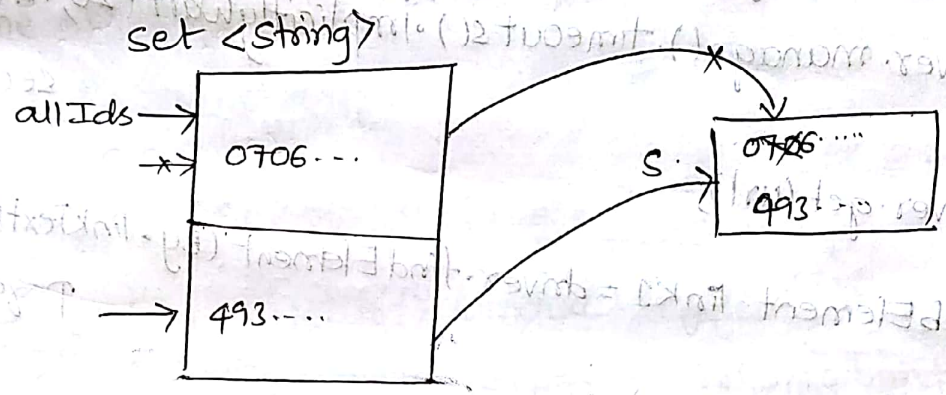
```
psvm (String [] args)
{
String url = "file:///D:/ABC/July1/WebPages/Page1.html";
ChromeDriver driver = new ChromeDriver();
driver.manage().window().maximize();
driver.manage().timeouts().implicitlyWait(20,TimeUnit.
                                                SECONDS);
driver.get(url);
WebElement link = driver.findElement(By.linkText
                                    ("gotopage2"));

Thread.sleep(4000);
link.click();
Thread.sleep(4000);
driver.quit(); // to terminate entire browser process
// driver.close(); // to close parent tab only
}
```

Scanned by CamScanner

switching from one tab to other tab
(or)
switching from one window to other window





page1.html

```
<html>
  <body>
    <a href = "page2.html" target = "blank"> gotopage2 </a>
  </body>
</html>
```

page2.html

```
<html>
  <body>
    <a href = "page3.html"> gotopage 3 </a>
  </body>
</html>
```

page3.html

```
<html>
  <body>
    <h1> Welcome to ABC </h1>
  </body>
</html>
```

Scanned by CamScanner

Syntax:-

```
for(String s : allIds)
{
  driver.switchTo().window(s);
}
```

```
psvm (String args[])
{
String url = "file:///D:/ABC/July1/page1.htm)";
ChromeDriver.driver = new ChromeDriver();
driver.manage().window().maximize();
driver.manage().timeouts().implicitlyWait(20, TimeUnit.
                                          SECONDS);
driver.get(url);
WebElement link1 = driver.findElement(By.linkText("go
                                          page2"));
Thread.sleep(4000);
link1.click();
String parentId = driver.getWindowHandle();
System.out.println("Parent id is "+parentId);
Set<String> allIds = driver.getWindowHandles();
S.O.P(allIds);
for(String s : allIds)
{ driver.switchTo().window(s);
}
WebElement link2 = driver.findElement(By.linkText("go
                                          page3"));
Thread.sleep(5000);
link2.click();
```

Scanned by CamScanner

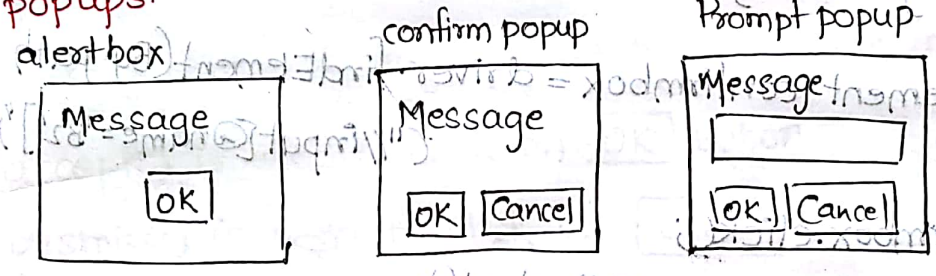
```
Thread.sleep(5000);
driver.quit();
}
}
```

→ In the above pgm, the for loop is designed in such a way that, it will always make the driver to point to the last tab. If we have to select specific tabs then the above pgm is not suitable.

→ The above pgm can be written in a different way as shown below, [instead of for loop use ArrayList]

```
ArrayList<String> al = new ArrayList<String>(allIds);
driver.switchTo().window(al.get(1));
```

Write a script to handle popups (or) WAS to handle alert popups.



Script to handle alert box

```
psvm(String[] args)
{
String url = "https://www.echoecho.com/javascript4.htm";
ChromeDriver driver = new ChromeDriver();
driver.manage().window().maximize();
driver.manage().timeouts().implicitlyWait(20,TimeUnit.seconds);
driver.get(url);
WebElement alertbox = driver.findElement(By.xpath("//input[@name='B1']"));
```

Scanned by CamScanner

```
alertbox.click();
Alert a = driver.switchTo().alert();
Thread.sleep(5000);
s.o.p (a.getText());
Thread.sleep(5000);
a.accept();
 }
}
```

Write a script to handle confirm box popup

```
psvm(string[] args)
{
String url = "http://www.echoecho.com/javascript4.htm";
ChromeDriver driver = new ChromeDriver();
driver.manage().window().maximize();
driver.manage().timeouts().implicitlyWait(20, TimeUnit.
                                                SECOND);
driver.get(url);
WebElement confirmbox = driver.findElement(By.path
                        ("//input[@name='B2']"));
confirmbox.click();
Alert a = driver.switchTo().alert();
Thread.sleep(5000);
s.o.p(a.getText());
Thread.sleep(5000);
a.dismiss();
```

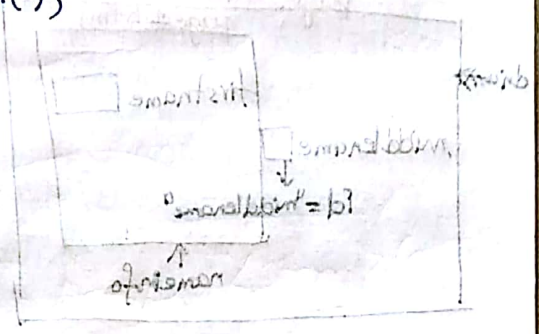
Script to handle prompt box popup

```
psvm(String[] args)
{
String url = "http://www.echoecho.com/javascript4.htm";
```

Scanned by CamScanner

```
FirefoxDriver driver = new FirefoxDriver();
driver.manage().window().maximize();
driver.manage().timeouts().implicitlyWait(20, TimeUnit.SECONDS);
driver.get(url);
WebElement promptbox = driver.findElements(By.xpath("//
                        input[@name='B3']"));

promptbox.click();
Alert a=driver.switchTo().alert();
Thread.sleep(5000);
s.o.p (a.getText());
Thread.sleep(5000);
a.sendkeys("12345");
Thread.sleep(5000);
a.accept();
}
}
```



Note: getText() is used to get the text from the alertbox.

- accept() is used to click on OK button.
- dismiss() is used to click on cancel button
- sendkeys() is used to type the text in the textbox of promptbox popup.

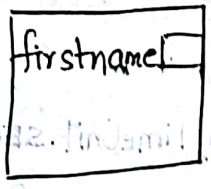
Handling Frames

→ A Webpage inside another webpage (or) a webpage embedded into another webpage is reffered to as frame.

→ In html, in order to create frames, we should use <iframe> tag

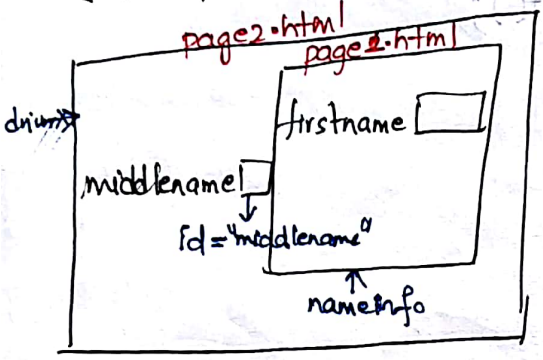
Scanned by CamScanner

page1.html




```
<html>
  <body>
  firstname<input type="text" id="fname" />
  </body>
</html>
```

page2.html




```
<html>
  <body>
    middlename<input type="text" id="middlename" />
    <iframe src="page1.html" name="new entry" id="nameinfo"></iframe>
  </body>
</html>
```

switching to frame can be done using 3 ways:

(i) driver.switchTo().frame(index)

(ii) driver.switchTo().frame("id/name")

(iii) xpath

Scanned by CamScanner

```
psvm (String[] args)
{
string url = "file:///D:/ABC/July 1/WebPage/Frame/page2.html";
ChromeDriver driver = new ChromeDriver();
driver.manage().window().maximize();
driver.manage().timeouts().implicitlyWait(20,TimeUnit.SECONDS);
driver.get(url);
WebElement mname = driver.findElement(By.id("middlename"));
mname.sendKeys ("ramesh");
//driver.switchTo().frame(0);
//or
//driver.switchTo().frame("nameinfo");
//or
driver.switchTo().frame("nameinfo");
WebElement frame = driver.findElement(By.xpath("//input
                                    [@id='firstname']"));

frame.sendKeys ("sachin");
}
}
```

Mouse and Keyboard Events

Scenario to be automated:

(i) Goto amazon.in website

(ii) Move to 'shopby' category

(iii) Move to 'echo and alexa'.

(iv) Under echo and alexa, move to all new echo, dot and click on it using mouse functions.

Scanned by CamScanner

```
psvm (String [] args)
{
String url = "https://www.amazon.in/";
ChromeDriver driver = new ChromeDriver();
driver.manage().window().maximize();
driver.manage().timeouts().implicitlyWait(20, TimeUnit.seconds);
driver.get(url);
Actions a = new Actions(driver);
WebElement shopby = driver.findElement(By.xpath("//span
                    [text()='shop by']"));
WebElement echoandalexa = driver.findElement(By.xpath
                    ("//span[text()='Echo & Alexa']"));
WebElement echodot = driver.findElement(By.xpath("
                    //span[text()='All new Echo Dot']
a.moveToElement(shopby).build().perform();
Thread.sleep(5000);
a.moveToElement(echoandAlexa).build().perform();
Thread.sleep(5000);
a.moveToElement(echodot).click().build().perform()
}
}
```

The above pgm can be written in an alternate way as shown below

```
a.moveToElement(shopby).pause(4000).moveToElement(ech
andalexa).pause(4000).moveToElement(echodot).pause
(4000).click().build().perform();
```

Scanned by CamScanner

Keyboard events

scenario to be automated

1. Open any browser and navigate to www.cleartrip.com
2. Identify the from field using the mouse functions and click on it.
3. Type 'tri' in the from field.
4. Press the downward arrow key 5 times, to select the city tripoli.
5. Press the enter key
6. Automate the above scenario using mouse and Keyboard functions.

```
psvm (String [] args) throws InterruptedException {
    String url = "https://www.cleartrip.com/";
    FirefoxDriver driver = new FirefoxDriver();
    driver.manage().window().maximize();
    driver.manage().timeouts().implicitlyWait(20,TimeUnit.seco
                                                    NDS);
    driver.get(url);
    Actions a = new Actions(driver);
    WebElement from = driver.findElement(By.id("FromTag"))
                                                    ;

    a.moveToElement(from).click().build().perform();
    a.sendKeys("tri").pause(2000).sendKeys(Keys.ARROW-DOW
                                                    N)
    .pause(2000).sendKeys(Keys.ARROW-DOWN).pause(2000).send
    Keys(Keys.ARROW-DOWN).sendKeys(Keys.ARROW-DOWN).
    sendKeys(Keys.ARROW-DOWN).sendKeys(Keys.ENTER).
    build().perform();
}
```

Scanned by CamScanner

Scenario to be automated:

1. Open any browser and navigate to google.com
2. Move to the search bar and click on it and type 'ABC FOR TECHNOLOGY TRAINING' in uppercases and press enter

```
psvm (String[] args) throws InterruptedExeception {
String url = "https://www.google.com/";
ChromeDriver driver = new ChromeDriver();
driver.manage().window().maximize();
driver.manage().timeouts().implicitlyWait(20, TimeUnit.SECONDS);
driver.get(url);
Actions a = new Actions(driver);
WebElement search = driver.findElement(By.xpath
("//input[@id='1st-ib']"));
a.moveToElement(search).click().KeyDown(Keys.SHIFT).
sendKey("abc for technology training').sendKeys(Keys.ENT
.build().perform();
}
}
```

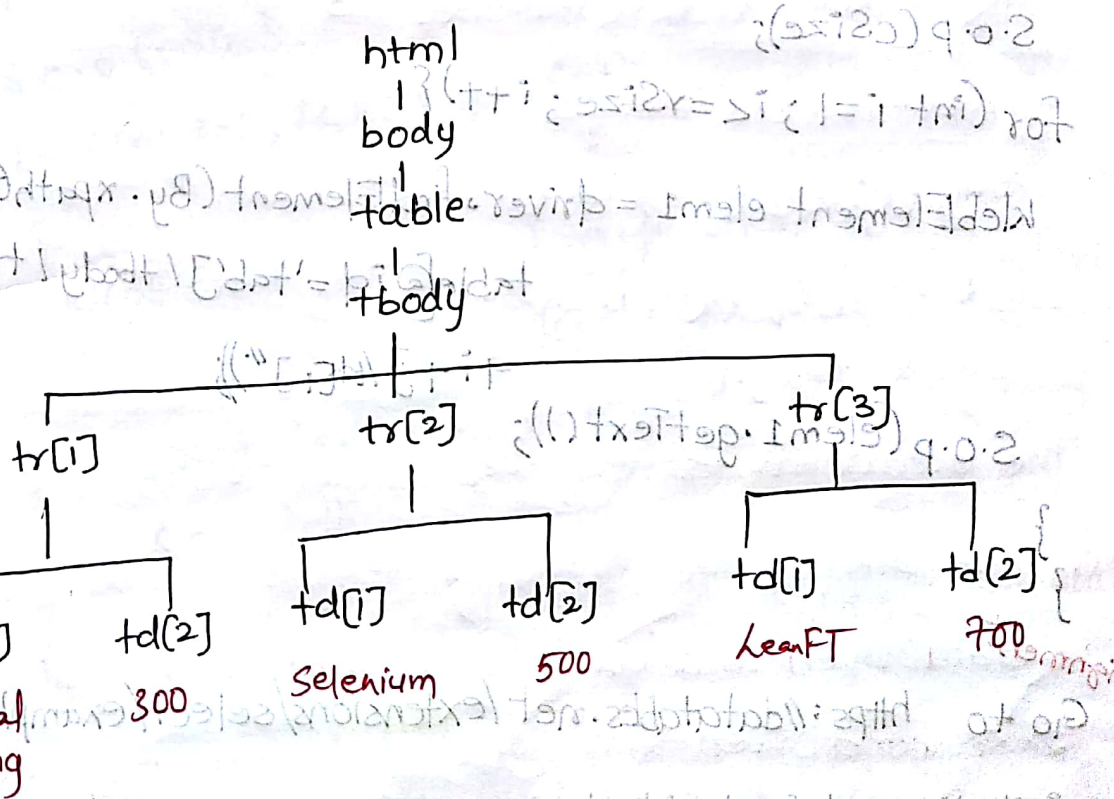
SCRIPT to handle Tables

From the below table print all the course names

Manual Testing	300
Selenium	500
LeanFT	700

Scanned by CamScanner

```
<html>
  <body>
    <table border="1" id='tab'>   <= To identify Particular Table (using id)
      <tbody>
        <tr>
          <td> Manual Testing </td>
          <td> 300 </td>
        </tr>
        <tr>
          <td> Selenium </td>
          <td> 500 </td>
        </tr>
        <tr>
          <td> LeanFT </td>
          <td> 700 </td>
        </tr>
      </tbody>
    </table>
  </body>
</html>
```



Scanned by CamScanner

```
class TableDemo{
psvm(String[] args){
String url = "file:///D:/ABC/July17/WebPages/Table2.html";
ChromeDriver driver = new ChromeDriver();
driver.manage().window().maximize();
driver.manage().timeouts().implicitlyWait(20, TimeUnit.SECONDS);
driver.get(url);
List<WebElements> rows = driver.findElements(By.xpath("
                                //table[@id='tab']/tbody/tr"));

int rSize = rows.size();
S.o.p(rSize);
List<WebElement> cols = driver.findElements(By.xpath
                        ("//table[@id='tab']/tbody
                                /tr[1]/td"));

int cSize = cols.size();
S.o.p(cSize);
for(int i=1; i<=rSize; i++){
WebElement elem1 = driver.findElement(By.xpath("//
                        table[@id='tab']/tbody/tr["
                                +i+"]/td[1]"));

S.o.p(elem1.getText());
}
}
```

Assignment

1. Go to https://datatables.net/extensions/select/examples/initialisation/simple.html

2.

Scanned by CamScanner

```
psvm(String [] args) {
String url = "https://datatables.net/extensions/select/examples
             /initialisation/simple.html";

ChromeDriver driver = new ChromeDriver();
driver.manage().window().maximize();
driver.mange().timeouts().implicitlyWait(20,TimeUnit.seconds);
driver.get(url);
List<WebElement> rows = driver.findElements(By.xpath("//
                        table[@id='example']/tbody/tr"))
                        ;

  int rSize = rows.size();
  System.out.println(rSize);
List<WebElement> cols = driver.findElement(By.xpath("//
                        table[@id='example']/tbody/tr[1]/
                        td"));
int cSize = cols.size();
S.O.P(cSize);
for(int i=1;i<=rSize;i++)
 {
  WebElement element = driver.findElement(By.xpath("//table
                       [@id='example']/tbody/tr["+i+"]/

    String text = element.getText();
    S.O.P(text);
 }
```

For the above table, extract salary info and calculate the sum and display

```
for(int i=1;i<=rSize;i++)
{ WebElement element = driver.findElement(By.xpath("
                       //table[@id='example']/tbody/
                       tr["+i+"]/td[2]"));
```

Scanned by CamScanner

```
String text = element.getText();
text = text.replace("$","");
text = text.replace(",","");
s.o.p(text);
sum = Integer.parseInt(text) + sum;
}
s.o.p("the sum is " +sum);
}
```

Frameworks

Frameworks is a set of rules or deadlines to be followed in automation process to achieve 100% efficiency or better performance.

Framworks mainly includes 3 phrases

(i) Framework design

(ii) Framework implementation

(iii) Framework excecution

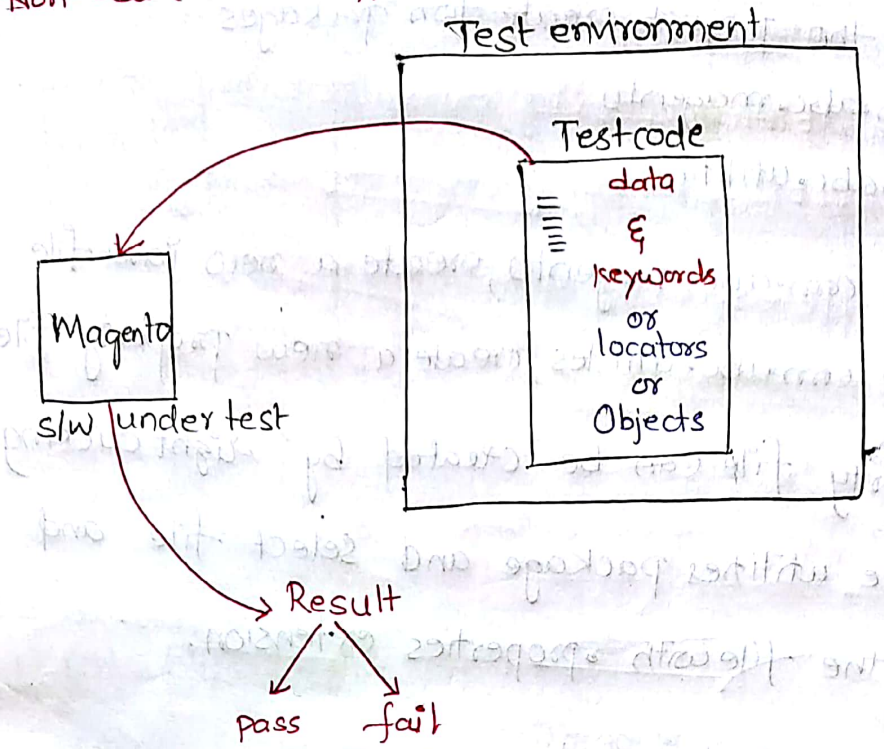
Framework design

The automation test engineer during automation proce can used any of the following designs

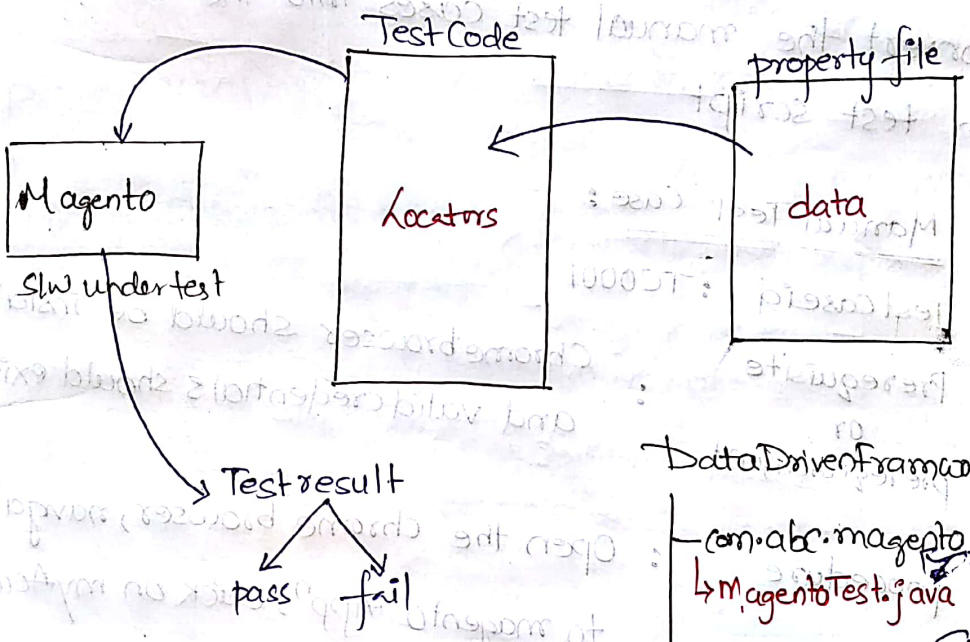
1. DataDrivenFramework
2. Keyword Driven Framework
3. Hybrid Driven Framework
4. Page Object Model
5. page Factory Model
6. TestNG Framework

Scanned by CamScanner

Non-data driven approach



Data driven approach



Data Driven framework

- com.abc.magento
  - MagentoTest.java (Test code)
- com.abc.utilities (property file)
  - Datadriven.properties

Steps involved in creation of data driven framework:

1. Create a new java project and give a name as the data driven framework
2. Add the selenium jars and the driver softwares to the project.

Scanned by CamScanner

3. Inside the project, create two packages
   (i) com.abc.magento
   (ii) com.abc.utilities

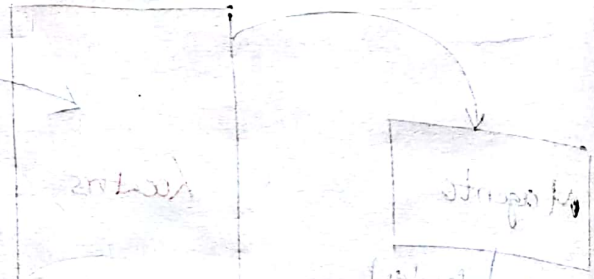
4. Inside com.abc.magento, create a new java file

5. Inside com.abc.utilites, create a new property file.

6. Property file can be created by right clicking on the utilities package and select file and save the file with .properties extension.

Framework implementation

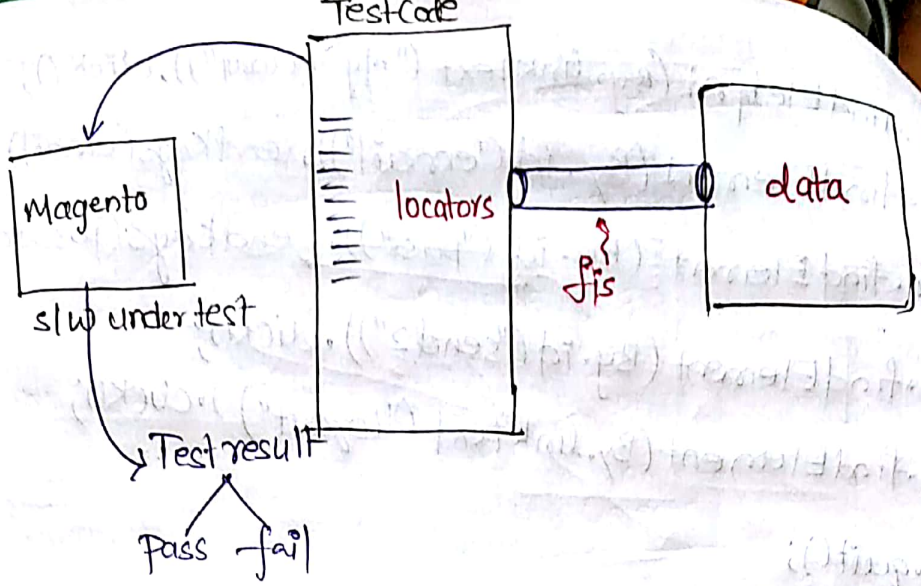
1. In this phase, the automation test engineer would convert the manual test cases into the test code or test script



Manual Test Case:

TestCaseid	: TC0001
Prerequisite or prerequirement	: Chrome browser should be installed and valid credentials should exist
procedure	: Open the chrome browser, navigate to magento App^n, click on myAccount, enter email, enter the pw, click on login. & click on logout.
Expected Result	: MagentoAppn, should be logged out successfully.

Scanned by CamScanner



data driven properties

1. url - https://www.magento.com
2. email - subramanyaraj87@gmail.com
3. pass - Welcome123

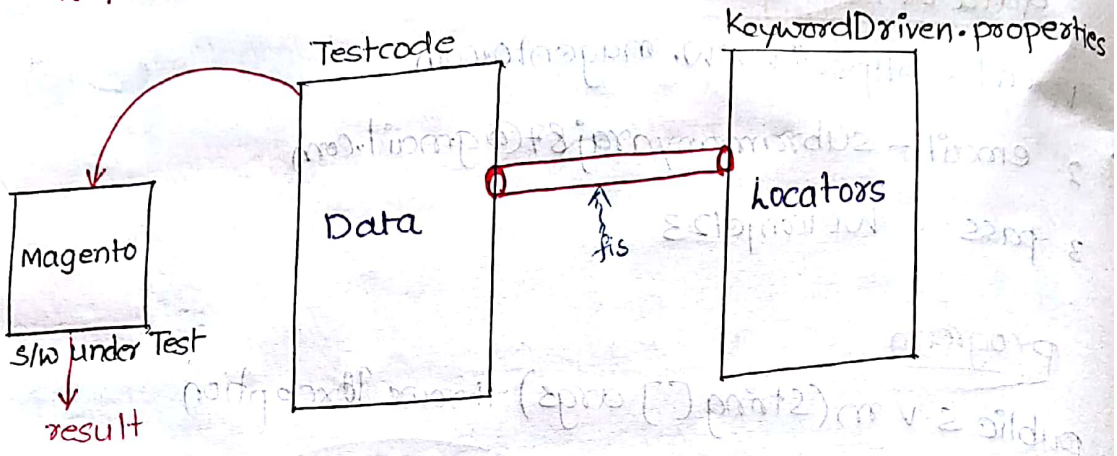
program

```
public s v m(String [] args) throws IOException
{
FileInputStream fis = new FileInputStream("D:\\ABC\\July\\
                      DataDrivenFrameWork\\src\\
Properties p = new Properties();
p.load(fis);
String url = p.getProperty("url");
String email = p.getProperty("email");
String password = p.getProperty("password");
ChromeDriver driver = new ChromeDriver();
driver.manage().window().maximize();
driver.manage().timeouts().implicitlyWait(20, TimeUnit.SECONDS);
driver.get(url);
```

Scanned by CamScanner

```
driver.findElement(By.linkText("My Account")).click();
driver.findElement(By.id("email")).sendKeys(email);
driver.findElement(By.id("pass")).sendKeys(password);
driver.findElement(By.id("send2")).click();
driver.findElement(By.linkText("Logout")).click();
driver.quit();
```

Keyword driven Framework



Jars & Drivers

KeywordDrivenFramework

- com.abc.magento ← test code
  - MagentoTest.java
- com.abc.utilities ← property file
  - KeywordDriven.properties

Property file:

```
myAcct  = //a[text()='My Account']
email   = //input[@id='email']
pass    = //input[@id='pass']
login   = //button[@id='send2']
logout  = //a[text()='logout']
```

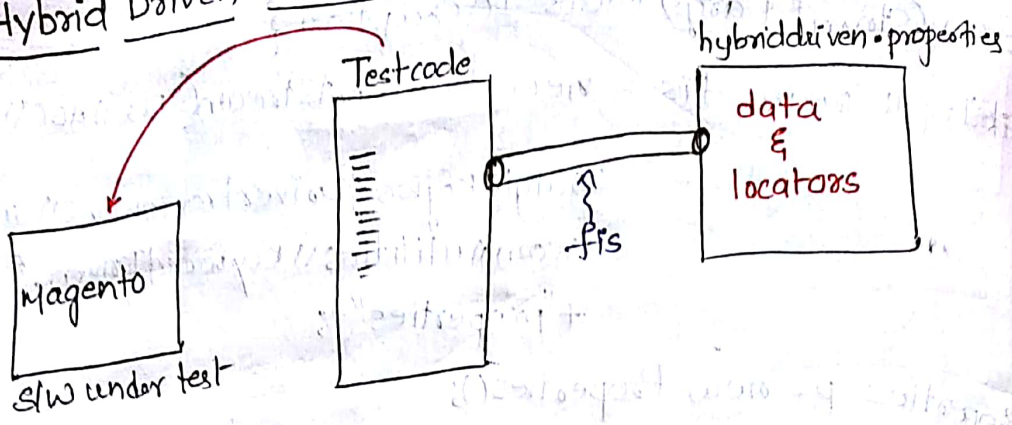
Scanned by CamScanner

testcode :

```
psvm (String[] args) throws IOException {
FileInputStream fis = new FileInputStream("D:\\ABC\\
                    July\\KeywordDriverframework\\src
                    \\com\\utilities\\KeywordDriven."
                    +"properties");
Properties p=new Properties();
p.load(fis);
String myAcctLocator = p.getProperty("My Acct");
String emailLocator =p.getProperty("email");
String passLocator =p.getProperty("login");
String logoutLocator =p.getProperty("logout");
ChromeDriver driver = new ChromeDriver();
driver.manage().window().maximize();
driver.manage().timeout().implicitlyWait(20,TimeUnit.seconds);
driver.get("https://www.magento.com");
driver.findElement(By.xpath(myAcctLocator)).click();
driver.findElement(By.xpath(emailLocator)).sendKeys
                    ("suchendra.abc@gmail.com");
driver.findElement(By.xpath(passLocator)).sendKeys("
                    Welcome123");
driver.findElement(By.xpath(loginLocator)).click();
driver.findElement(By.xpath(logoutLocator)).click();
driver.quit();
```

Scanned by CamScanner

Hybrid Driven Framework



Jars & Drivers

HybridDrivenFramework
- com.abc.magento ← Testcode
  - MagentoTest.java
- com.abc.utilities ← propertyfile
  - hybridDriven.properties

Propertyfile:

```
url = https://www.magento.com
un = sucheendra.abc@gmail.com
password = Welcome123
myAcct = //a[text()='My Account']
email = //input[@id='email']
pass = //input[@id='pass']
login = //button[@id='send2']
logout = //a[text()='log out']
```

Test code:

```
psvm(String[] args) throws IOException {
FileInputStream fis = new FileInputStream("D:\\ABC\\HybridDrivenFramework\\src\\com\\abc\\utilities\\hybridDriven.properties");
Properties p = new Properties();
```

Scanned by CamScanner

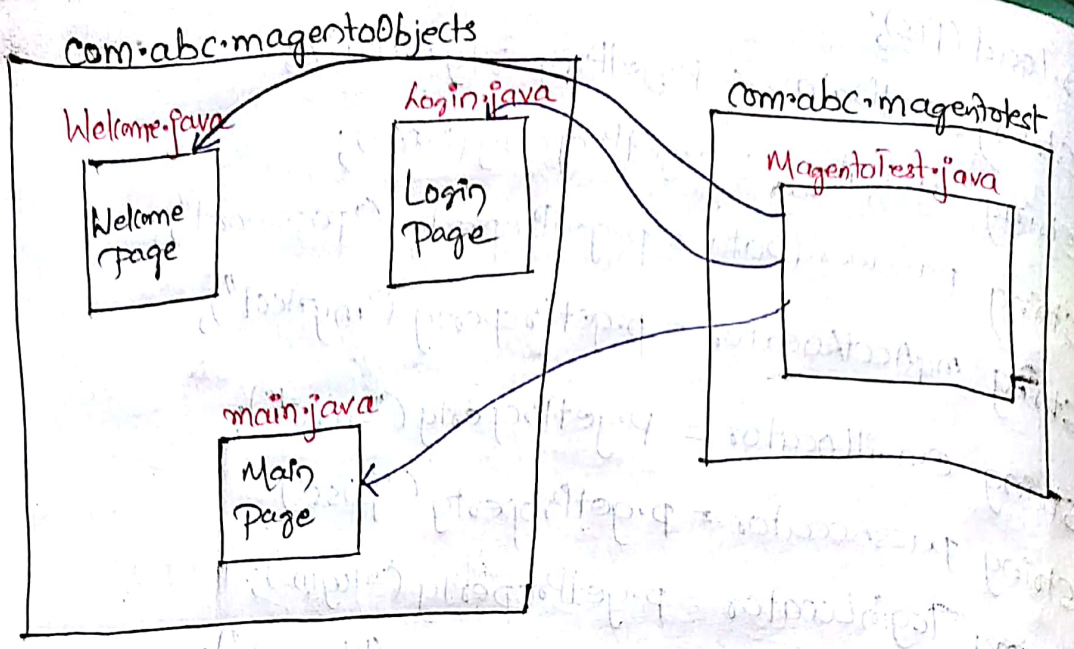
```
p.load(fis);
String urlData = p.getProperty("url");
String unData = p.getProperty("un");
String passwordData = p.getProperty("password");
String myAcctLocator = p.getProperty("myAcct");
String emailLocator = p.getProperty("email");
String passLocator = p.getProperty("pass");
String loginLocator = p.getProperty("login");
String logoutLocator = p.getProperty("logout");
ChromeDriver driver = new ChromeDriver();
driver.manage().window().maximize();
driver.manage().timeouts().implicitlyWait(20, TimeUnit.seconds);
driver.get(urlData);
driver.findElement(By.xpath(myAcctLocator)).click();
driver.findElement(By.xpath(emailLocator)).sendKeys(unData);
driver.findElement(By.xpath(passLocator)).sendKeys(passwordData);
driver.findElement(By.xpath(loginLocator)).click();
driver.findElement(By.xpath(logoutLocator)).click();
driver.quit();
```

Page Object Model

Jars & Drivers

PageObjectModel
- com.abc.magentoobjects
  - Welcome.java
  - login.java
  - main.java
- com.abc.magentotest
  - MagentoTest.java

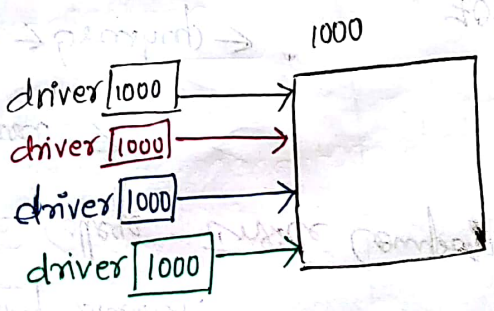
Scanned by CamScanner




```
Class Welcome{
WebDriver driver;
    By myAcct = By.LinkText("My Account");
public Welcome(WebDriver driver){
  this.driver = driver;
}
public void clickOnMyAccount(){
 driver.findElement(myAcct).click();
 }
}
Class Login{
WebDriver driver;
   By email = By.id("email");
   By pass = By.id("pass");
   By login = By.id("send2");
public Login(WebDriver driver){
   this.driver = driver;
}
public void typeEmail(String em){
driver.findElement(email).sendKeys(em);
}
public void typePassword(String pw){
driver.findElement(pass).sendKeys(pw);
}
public void clickOnLogin(){
driver.findElement(login).click();
```

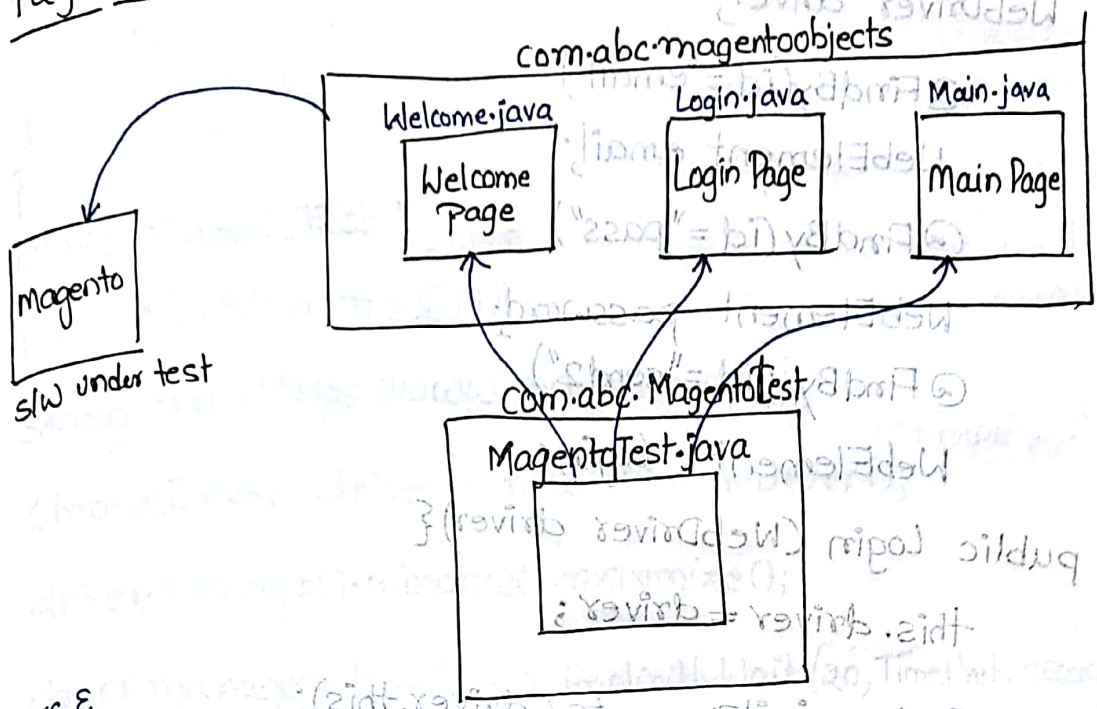
Scanned by CamScanner

```
}
}
class Main {
WebDriver driver;
By Logout = By.LinkText("Logout");
public Main(WebDriver driver){
 this.driver=driver;
}
public void clickOnLogout(){
driver.findElement(Logout).click();
}
}
class MagentoTest{
public static void main(String[] args){
WebDriver driver = new ChromeDriver();
driver.manage().window().maximize();
driver.get("https://www.magento.com");
Welcome w = new Welcome(driver);
w.clickOnMyAcct();
Login l = new Login(driver);
l.typeEmail("sucheendra.abc@gmail.com");
l.typePassword("Welcome123");
l.clickOnLogin();
Main m = new Main(driver);
 m.clickOnLogout();
 driver.quit();
}
}
```



Scanned by CamScanner

Page Factory Model



Jars & Drivers

pageFactoryModel

- com.abc.magentoobjects
  - Welcome.java
  - Login.java
  - Main.java
- com.abc.magentotest
  - MagentoTest.java

```
class Welcome{
WebDriver driver;
   @FindBy (LinkText = "My Account")
    WebElement myAcct;
public Welcome (WebDriver driver){
      this.driver = driver;
   PageFactory.initElements(driver, this);
}
public void clickOnMyAcct(){
   myAcct.click();
  }
}
```

Scanned by CamScanner

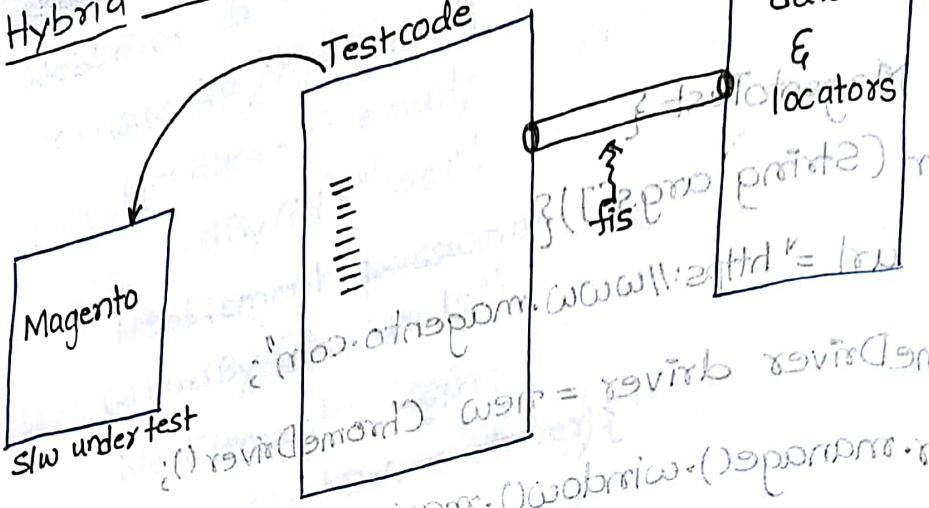
```
class Login{
WebDriver driver;
    @FindBy(id="email")
     WebElement email;
    @FindBy(id="pass")
     WebElement password;
    @FindBy(id="send2")
     WebElement Login;
public Login(WebDriver driver){
    this.driver=driver;
PageFactory.initElements(driver,this);
}
public void typeEmail(String em){
 email.sendKeys(em);
}
public void typePassword(String pw){
 password.sendKeys(pw);
}
public void ClickOnLogin(){
  login.click();
}
 class Main{
klebDriver driver;
 @FindBy(LinkText="Log Out")
 WebElement LogOut;
public Main(WebDriver driver){
    this.driver=driver;
PageFactory.initElements(driver,this);
```

Scanned by CamScanner

```
}
public void clickOnLogout() {
  LogOut.click();
}
}
class MagentoTest {
p.s.v.m (String args[]){
String url = "https://www.magento.com";
ChromeDriver driver = new ChromeDriver();
driver.manage().window().maximize();
driver.manage().timeOuts().implicitlyWait(20, TimeUnit.SECONDS);
driver.get(url);
Welcome w = new Welcome(driver);
w.clickOnMyAcct();
Login L = new Login(driver);
L.typeEmail("sucheendra.abc@gmail.com");
L.typepassword("Welcome123");
L.ClickOnLogin();
      Main m = new Main(driver);
      m.clickOnLogout();
      driver.quit();
  }
}
```

Scanned by CamScanner

Hybrid driven Framework using excel



excel

old version
97-2003
.xls

new version
2007-2016
.xlsx

.xls
HSSFWorkbook
HSSFSheet
HSSFRow
HssfCell

Book
Sheet
Row
cell

.xlsx
XSSFWorkbook
XSSFSheet
XSSFRow
XSSFCell

Jars & drivers

HybridDrivenFrameworkExcel
- com.abc.magento (Testcode)
  - MagentoTest.java
- com.abc.utilities (files)
  - hybrid.xlsx

Scanned by CamScanner

TestcaseID	Procedure	Action	Objects	Data
TC001	Open chrome browser	Open		
TC002	launch magento app'n	Navigate		https://www.magento.com
TC003	click on MyAcct	Click	//a[text()='My Account']	
TC004	enter the email	Type	//input[@id='email']	sucheendra.abc@gmail.com
TC005	enter the password	Type	//input[@id='pass']	Welcome123
TC006	click on login	Click	//button[@id='send2']	
TC007	click on logout	Click	//a[text()='Log Out']	
TC008	close the browser	Close		

Steps involved in downloading poi jars:

1. Open any browser and search for apache poi
2. Click on the first link, click on download link.
3. Scroll down to the end of webpage and click on one link Binary artifacts.
4. search for poi-bin-3.17-20170915.zip and click on it
5. Extract the zip file and place it in selenium components folder.

Adding the poi jars to the newly created project

1. Go to eclipse, select the project, right click Go to build path, click on configure build path.
2. Go to Libraries click on Add External Jars
3. Select all the jar files present in poi-3.17 folder
4. Go to lib folder, add all the jars present in lib folder
5. Go to ooxml-lib folder, add all the jar files and

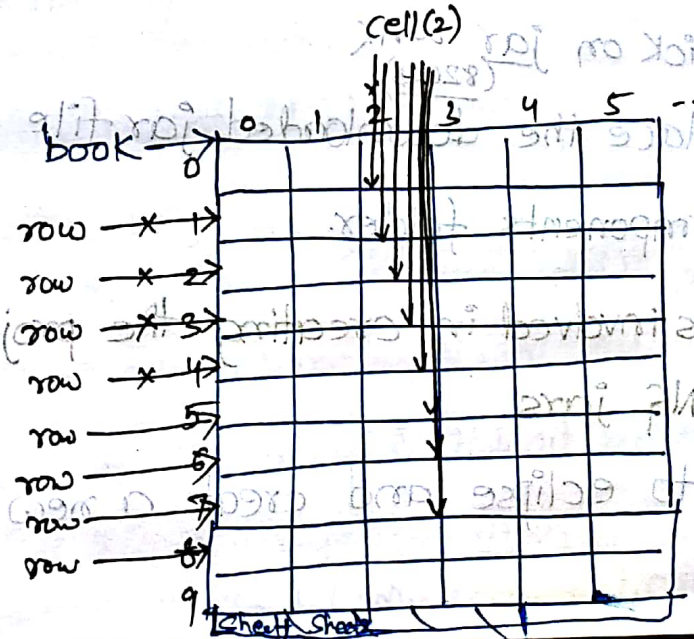
Scanned by CamScanner

click on Apply and Close

```
class MagentoTest {
    public static XSSFWorkbook book;
    public static XSSFSheet sheet;
    public int numofRows;
    public static XSSFRows row;
    public static XSSFCell cell;
    public static String data;
    public static String action;
    public static WebDriver driver;
    public static String getCellValues (int rownum, int cellnum)
    {
        row = sheet.getRow(row.num);
        cell = row.getCell (cellnum);
        data = cell.getStringcellvalue();
        returndata;
    }
    p.s.v.m(String args[]){
    FileInputStream fis = new FileInputStream ("----hybrid.xlsx")
        book = new XSSFWorkbook (fis);
        sheet = book.getSheetAt (0);
    numofRows = sheet.getPhysicalNumberOfRows();
        S.O.p(numOfRows);
    for(int i=1; i<numOfRows; i++)
    {
        action = getCellValues (i,2);
```

Scanned by CamScanner

```
s.o.p(action);
switch (action) {
case "open":
    driver = new ChromeDriver();
    driver.manage().window().maximize();
    driver.manage().timeOuts().implicitlyWait(20,TimeUnit.
                                                SECONDS);
    break;
case "navigate":
    driverget(getCellValues(i,4));
    break;
case "click":
    driver.findElement(By.xpath(getCellValues(i,3)).click()
                                                ;
    break;
case "type":
    driver.findElement(By.xpath(getCellValues(i,3)).
        sendKeys(getCellValues(i,4));
    break;
case "close";
    driver.quit();
    break;
}
}
}
}
```



Scanned by CamScanner

Steps involved in installing TestNG:

1. Open eclipse, go to help and click on eclipsemarketplace
2. In the find textbox, type TestNG and click on search
3. In the 'TestNG for eclipse' section, click on install.
4. After waiting for sometime, TestNG for eclipse checkbox will be displayed, enable the checkbox and then click on confirm.
5. Click on 'I Accept the terms' radio button and click on 'finish.'

TestNG Framework (Test Next Generation)

1. It is a framework which is used to test the software under test and obtain report.

Steps involved in downloading TestNG Jars

1. Open the browser and browse for MVNrepository
2. Click on the first link
3. In the search box, type testng and click on search.
4. Click on TestNG link
5. Click on 6.14.3 link
6. Click on jar link (820KB)
7. Place the downloaded jar file in the selenium components folder.

Steps involved in creating the project and adding the testNG jars.

1. Go to eclipse and create a new Java Project

Scanned by CamScanner

2. Right click on the newly created java project. Go to build path and click on configure buildpath.
3. Click on add external jars button.
4. Select the TestNG jar file and click on open.
5. Click on Apply and Close
6. Add Selenium jars and driver softwares.

Program:

```
public class MagentoDemo
{
@Test
public void positiveCredential2()
{
ChromeDriver driver = new ChromeDriver();
driver.manage().window().maximize();
driver.manage().timeOuts().implicitlyWait(20, TimeUnit.seconds);
driver.get("https://www.magento.com");
driver.findElement(By.linkText("My Account")).click();
driver.findElement(By.id("email")).sendKeys("sucheudra.abc@gmail.com");
driver.findElement(By.id("pass")).sendKeys("Welcome123");
driver.findElement(By.id("send2")).click();
driver.findElement(By.linkText("Log Out")).click();
}
public void positiveCredential1()
{
ChromeDriver driver = new ChromeDriver();
driver.manage().window().maximize();
driver.manage().timeOuts().implicitlyWait(20, TimeUnit.seconds);
driver.get("https://www.magento.com");
driver.findElement(By.linkText("My Account")).click();
```

Scanned by CamScanner

```
driver.findElement(By.id("email")).sendKeys("sucheendra.abc@gmail.com");
driver.findElement(By.id("pass")).sendKeys("Welcome123");
driver.findElement(By.id("send2")).click();
driver.findElement(By.LinkText("Log Out")).click();
}
}
}
```

Note: A testNG class consists of tests methods

→ Any Method which is annotated using @test is referred to as a test method.

→ A testNG class can contain any number of test methods

→ test method is also referred to as a test case.

→ When multiple test() are present, ascii value of the method name is compared. the one with < ascii value, is executed first

reports generated in testNG

1. Once the testNG class is excuted, select project, right click and click on refresh. Once we refresh the project, one folder by name 'testOutput' would appear.

→ Inside the testOutput folder we will have two types of reports.

(i) index.html

(ii) emailablereport.html

@BeforeMethod & @AfterMethod

@BeforeMethod:

→ Such methods which are annotated using @Befremeth they would get excecuted before the execution of each testcase or test method

Scanned by CamScanner

@AfterMethod : Such methods which are annotated using @AfterMethod, they will get executed after the execution of each test case or test method.

BeforeMethod
@Test
public void positiveCredential1() { ⟶ TestMethod
}
After method
Before method
@Test
public void positiveCredential2() { ⟶ TestMethod
}
AfterMethod

```
public class Demo{
  @BeforeMethod
  public void beforeMethod(){
    System.out.println("Before method executed");
  }
@AfterMethod
public void afterMethod(){
  System.out.println("After method executed");
}
@Test
public void positiveCredential1(){
  System.out.println("Inside positive credential1");
}
```

Scanned by CamScanner

```
    @Test
    public void positiveCredential2() {
        System.out.println("Inside positive Credential2");
    }
}
```

output:

```
Before method executed
inside positive Credential1
after method executed
Before method executed
inside positive Credential2
after method executed
```

program:

```
public class BeforeAfterMethodDemo
{
    ChromeDriver driver;

    @BeforeMethod
    public void openBrowser()
    {
        driver = new ChromeDriver();
        driver.manage().window().maximize();
        driver.manage().timeouts().implicitlyWait(20,TimeUnit
        driver.get("https://www.magento.com");
    }
    @AfterMethod
    public void terminate()
    {
        driver.quit();
```

Scanned by CamScanner

```
}
@Test
public void loginValidation()
{
 driver.findElement(By.linkText("My Account")).click();
 driver.findElement(By.id("email")).sendKeys("sucheendra.abc=
                                              @gmail.com");
 driver.findElement(By.id("pass")).sendKeys("Welcome123");
 driver.findElement(By.id("send2")).click();
 driver.findElement(By.linkText("Log out")).click();

}
@Test
public void clickRegister(){
    driver.findElement(By.linkText("My Account")).click();
    driver.findElement(By.xpath("//span[text()='Register']")).
                                                    click();
 }
}
```

@BeforeClass & @AfterClass

@BeforeClass : Such methods which are created using @BeforeClass would get executed only once before the execution of before() of the current class.

@AfterClass : Such methods which are created using @AfterClass would get executed only once after the execution of after() in the current class.

Scanned by CamScanner

Program:

```
public class Demo{

  @BeforeClass
  public void beforeClass(){
      S.o.p("Before class executed");
  }

  @AfterClass
  public void afterClass()
  {
    system.out.println("After class executed");
  }

  @BeforeMethod
  public void beforeMethod()
  {
    S.o.p("Before method executed");
  }

  @AfterMethod
  public void afterMethod()
  {
    S.o.p("After method executed");
  }

  @Test
  public void positiveCredential1()
  {
    S.o.p("Inside positive credential 1");
  }

  @Test
  public void positiveCredential2()
  {
    S.o.p("Inside positive credential 2");
  }
```

Scanned by CamScanner

Output:

```
Before Class executed
Before method executed
Inside positive credential 1
After Method executed
Before Method executed
Inside positive Credential 2
After method executed
After Class executed.
```

Priorities in the TestNG

The order of execution of the test methods or test cases inside the TestNG class is based on the alphabetical order or the ascii values. However we can change the order of execution of the test cases depending on the requirements using the concept of priority.

Syntax: `Test(priority = priority-num)`

Test method without priority

Program:

```
public class Demo {

@Test
public void a-method(){
  s.o.p("a-method executed");
}
```

Scanned by CamScanner

```
@Test
public void b_method()
{
    s.o.p("b_method executed");
}
@Test
public void c_method()
{
    s.o.p ("c_method executed");
}

@Test
public void d_method()
{
    s.o.p ("d_method executed");
}
@Test
public void e_method()
{
    s.o.p("e_method executed");
}
```

output

```
a_method executed
b_method executed
c_method executed
d_method executed
e_method executed
```

Scanned by CamScanner

with priority :

```
class Demo{
@Test(priority = 33)
public void a_method()
{ s.o.p ("a_method executed");
 }
@Test(priority = 0)
public void b_method()
{s.o.p ("b_method executed");
 }
@Test(priority = 42)
public void c_method()
{ s.o.p ("c_method executed");
 }
@Test(priority = 6)
public void d_method()
{
 s.o.p ("d_method executed");
 }
@Test(priority = 99)
public void e_method()
{
 s.o.p ("e_method executed");
 }
}
```

O/P:

```
b_method executed
d_method executed
a_method executed
c_method executed
e_method executed
```

Scanned by CamScanner