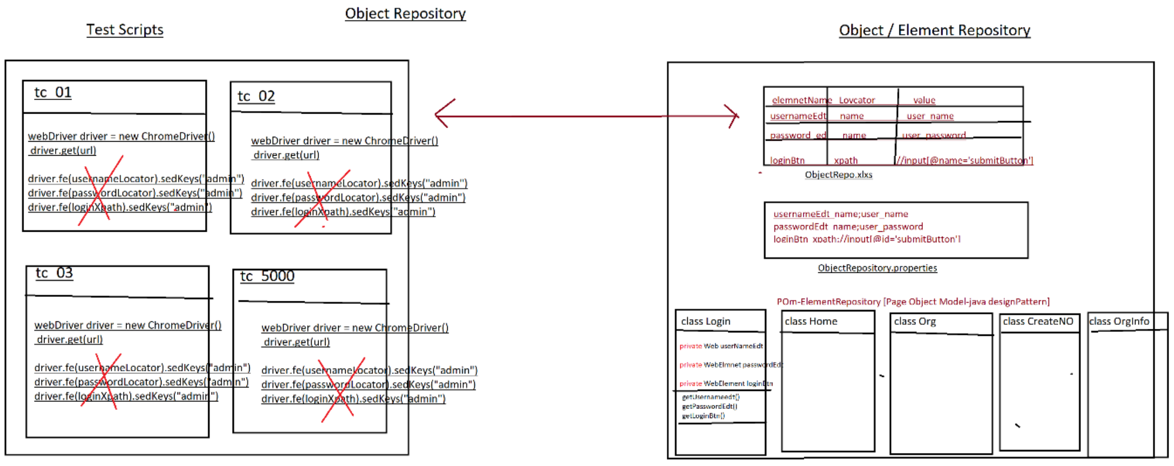
1. What is Object/Elements/POM Repository

Its collection of element locators & business libraries in one place & its developed using POM design pattern



```
webDriver driver = new ChromeDriver()
 driver.get(url)

driver.fe(usernameLocator).sedKeys("admin")
driver.fe(passwordLocator).sedKeys("admin")
driver.fe(loginXpath).sedKeys("admin")
```

elemnetName	Lovcator	value
usernameEdt	name	user_name
password_ed	name	user_password
loginBtn	xpath	//input[@name='submitButton']

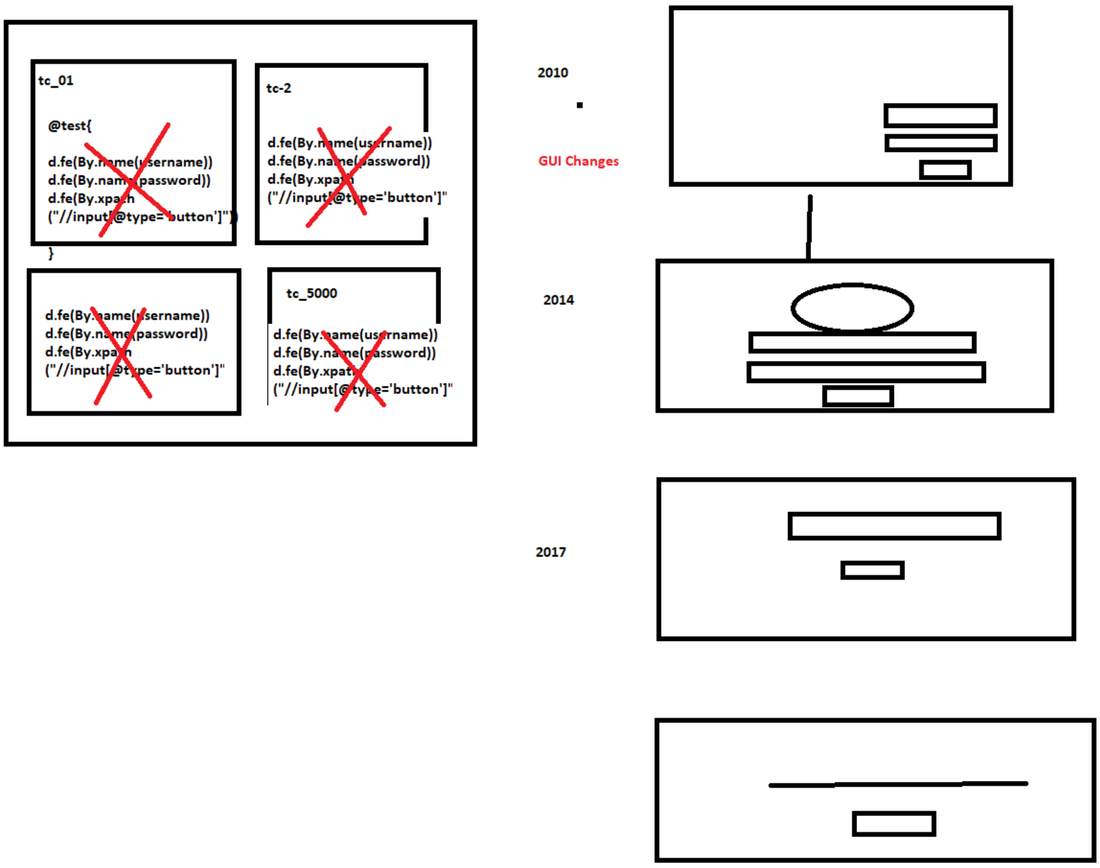
```
usernameEdt  name;user_name
passwordEdt  name;user_password
loginBtn  xpath://input[@id='submitButton']
```

```
private Web userNameEdt
private WebElmnet passwordEdt
private WebElement loginBtn
getUsernameedt()
getPasswordEdt()
getLoginBtn()
```

2. Why Object repository ?

As per the rule of the automation, we should not hardcode[fixed]elements with in test Scripts, instead we should get elements from Object Repository , because in Agile process due to frequent requirement changes , modification & maintenance of elements is tedious job

EG : below is the example of Gmail application GUI changes



```
tc_01

 @test{

 d.fe(By.name(username))
 d.fe(By.name(password))
 d.fe(By.xpath
 ("//input[@type='button']")
 }
```

```
tc-2

d.fe(By.name(username))
d.fe(By.name(password))
d.fe(By.xpath
("//input[@type='button']"
```

```
d.fe(By.name(username))
d.fe(By.name(password))
d.fe(By.xpath
("//input[@type='button']"
```

```
tc_5000

d.fe(By.name(username))
d.fe(By.name(password))
d.fe(By.xpath
("//input[@type='button']"
```

3. What is the advantages repository?
    a. Reusability of elements, no need to write xpath & other locators again & again
    b. Modification in Repository is easy, when GUI changes frequently
    c. Maintenance is easy, because all the elements we kept in one place
    d. Test Script Code Optimized via business reusable libraries
    e. More Readability
    f. Test Script development is faster due to business lib
    g. Test Script is more robust
    h. Handle Stale Elements Exception.
4. What is POM?

POM is a java design pattern preferred by google to develop object repository.

5. Why POM ?

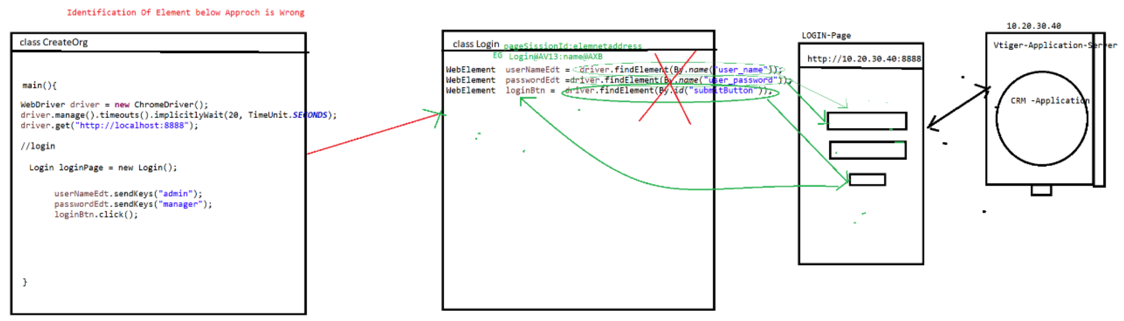
It's a well-organized structured design pattern, where we can maintain all the web elements in page wise, due to POM design pattern maintains & modification is easy & faster.

6. Advantages of POM:
    1. Well organized structure
    2. Handle stale element exception.
    3. maintains & modification of element is easy
    4. We can directly store Web Elements in java class
    5. Better fit for Agile processes
    6. Support Auto heal feature

7. Why @FindBy(locator) annotation instead of driver.findElement("locator")

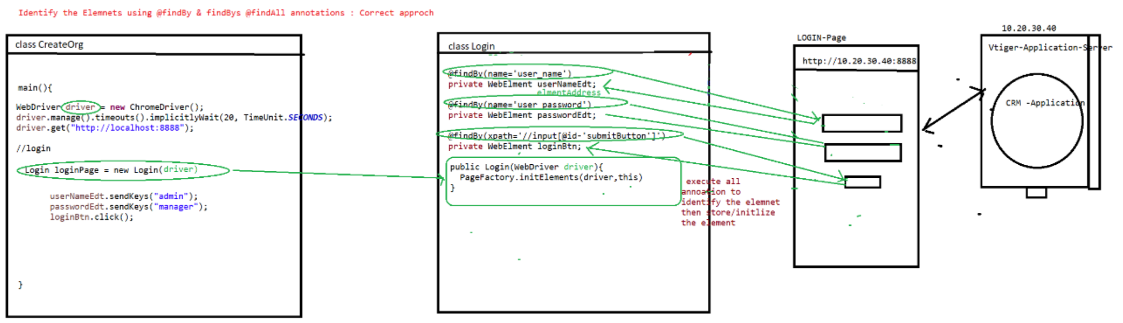
   Ans : to avoid staleElementReferenceException

8. What is staleElementReferenceException ?
   It's one of selenium Execption , whenever webdriver try to identify an element , element was available in GUI, but at time of performing an action on the elements element was not recognized due to page got refreshed or elements may become old or element not attached to page in such case we get staleElementReferenceException


Identification Of Element below Approch is Wrong

class CreateOrg
```
 main(){

WebDriver driver = new ChromeDriver();
driver.manage().timeouts().implicitlyWait(20, TimeUnit.SECONDS);
driver.get("http://localhost:8888");

//login

  Login loginPage = new Login();

      userNameEdt.sendKeys("admin");
      passwordEdt.sendKeys("manager");
      loginBtn.click();

 }
```

class Login
```
WebElement  userNameEdt = driver.findElement(By.name("user_name"));
WebElement  passwordEdt =driver.findElement(By.name("user_password"));
WebElement  loginBtn =  driver.findElement(By.id("submitButton"));
```


Identify the Elemnets using @findBy & findBys @findAll annotations : Correct approch

class CreateOrg
```
 main(){

WebDriver driver = new ChromeDriver();
driver.manage().timeouts().implicitlyWait(20, TimeUnit.SECONDS);
driver.get("http://localhost:8888");

//login

Login loginPage = new Login(driver)

      userNameEdt.sendKeys("admin");
      passwordEdt.sendKeys("manager");
      loginBtn.click();

 }
```

class Login
```
@findBy(name='user_name')
private WebElment userNameEdt;

@findBy(name='user_password')
private WebElment passwordEdt;

@findBy(xpath='//input[@id-'submitButton']')
private WebElment loginBtn;

public Login(WebDriver driver){
  PageFactory.initElements(driver,this)
}
```
elmentAddress
execute all annoation to identify the elemnet then store/initlize the element
LOGIN-Page
http://10.20.30.40:8888
10.20.30.40
Vtiger-Application-Server
CRM -Application

9. Rules of POM

Rule 1 : create separte java class for every page in a application & class name should be same page name

Rule 2 : Identify all the elements using @findBy & @findAll , @findbys annotations & store them in speific pom / java class (Element declartion)

Rule3 : For Every POM class create Constructor to get an Object of the class & initialize the Page Elements , in order to initialize all the page Elements we should use Pagefactory.initElement() (Element initialization)

Rule 4 : declare all the WebElements as private & provide getters methods to accesses elements in testScripts class [this processes is called Encapsulation]

Note : to create getters mtds inside the java class fallow below steps

➔place cursor inside the class➔ Right click➔source ➔generate getters & setters ➔ select the getters check box ➔ click on ok button

Rule 5 : Go to every page & identify the reusable business libraries & implement them in same POM class

10. Difference between POM & PageFactory design pattern?

POM is java design pattern, where will maintain all the Web element locator in well-organized manner

Pagefactory it's an extended design pattern of POM , which is used to create an Object to POM classes , & at the time of object creation it will execute all @findBy @findbys annotation then initialize all the elements

Difference between @findBy , @findAll &@findBys annotation

All annotation available in Selenium webdriver, its traditional ways to identify the elements in GUI.

@findBy : used to identify the element using one locator or one condition

@findAll : it contains multiple @findBy annotation , it mean we can identify the same element using multiple locator (multiple conditions) , it will use OR condition during execution of locator

```
@findALL({ @findBy(@id='username') , @findBy(name='user')})

Private Webelements userNAmeEdt;
```

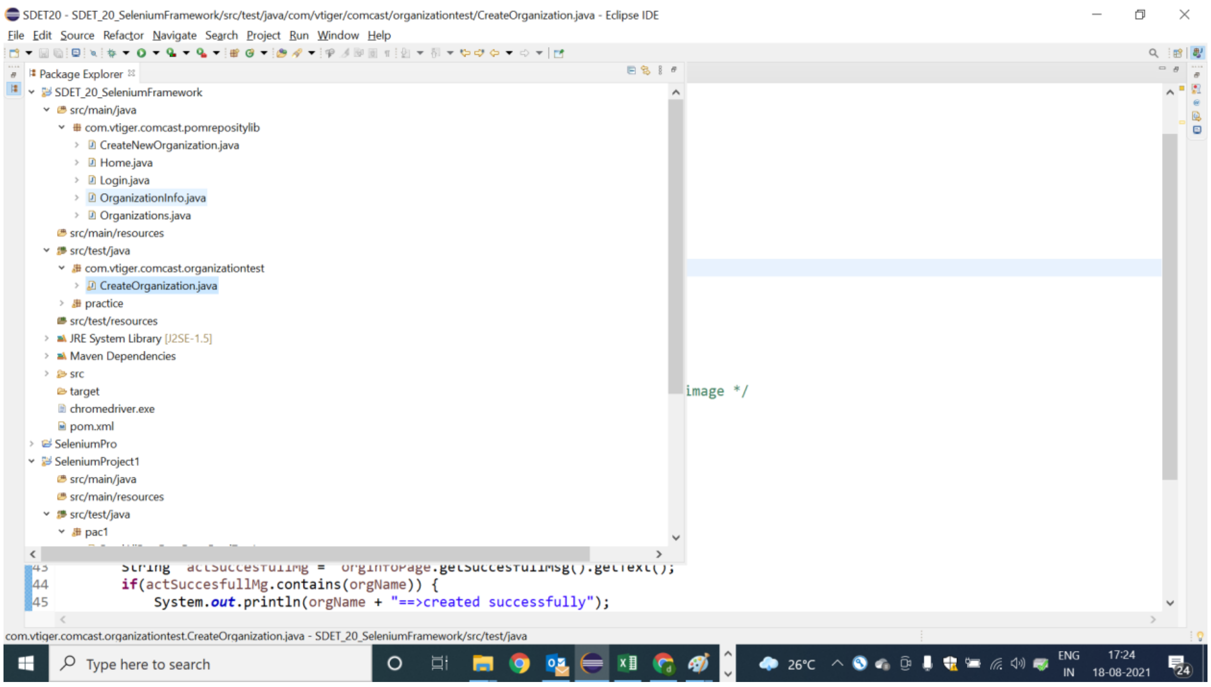
Note : using above concepts we can achieve **Autohealing** technique

AutoHealing : during execution , if one locator fails to identify the element , it will retry to identify the same element using another locator

@FindBys : it contains multiple @findBy annotation , it mean we can identify the elements using multiple locator (multiple conditions) , it will use AND condition to during execution of locator

```
@findBys({ @findBy(@id='username') , @findBy(name='user')})

Private Webelements userNAmeEdt;
```

Project Structure



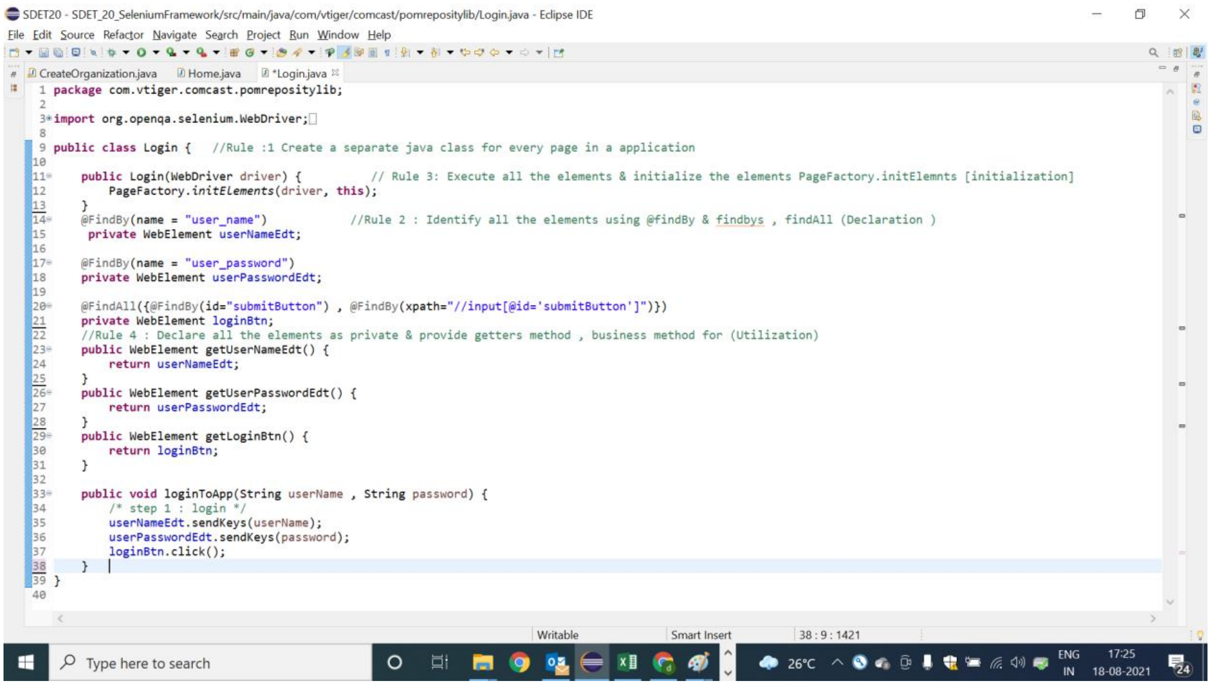
SDET20 - SDET_20_SeleniumFramework/src/test/java/com/vtiger/comcast/organizationtest/CreateOrganization.java - Eclipse IDE

```
SDET_20_SeleniumFramework
  src/main/java
    com.vtiger.comcast.pomrepositylib
      CreateNewOrganization.java
      Home.java
      Login.java
      OrganizationInfo.java
      Organizations.java
  src/main/resources
  src/test/java
    com.vtiger.comcast.organizationtest
      CreateOrganization.java
    practice
  src/test/resources
  JRE System Library [J2SE-1.5]
  Maven Dependencies
  src
  target
  chromedriver.exe
  pom.xml
SeleniumPro
SeleniumProject1
  src/main/java
  src/main/resources
  src/test/java
    pac1
```

```
image */
        String  actSuccesfullMg  =  orgInfoPage.getSuccessfullMsg().getText();
        if(actSuccesfullMg.contains(orgName)) {
            System.out.println(orgName + "==>created successfully");
```

com.vtiger.comcast.organizationtest.CreateOrganization.java - SDET_20_SeleniumFramework/src/test/java

Pom Classes : Login

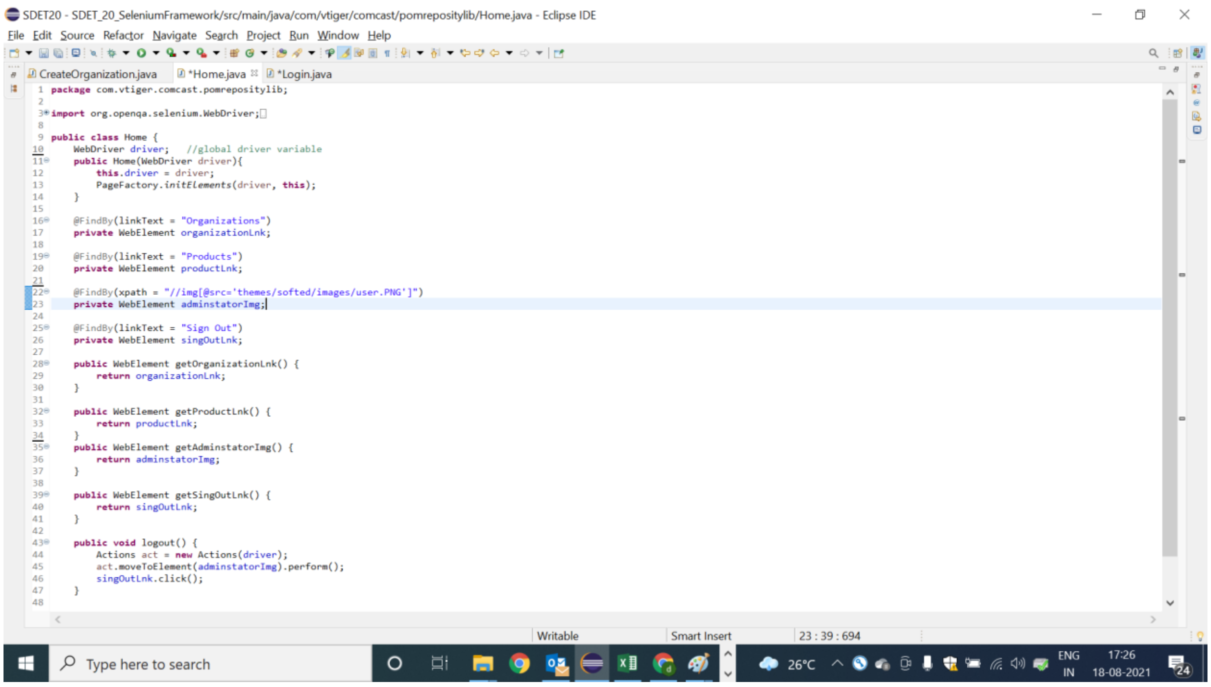


SDET20 - SDET_20_SeleniumFramework/src/main/java/com/vtiger/comcast/pomrepositylib/Login.java - Eclipse IDE

```java
package com.vtiger.comcast.pomrepositylib;

import org.openqa.selenium.WebDriver;

public class Login {   //Rule :1 Create a separate java class for every page in a application

    public Login(WebDriver driver) {           // Rule 3: Execute all the elements & initialize the elements PageFactory.initElemnts [initialization]
        PageFactory.initElements(driver, this);
    }
    @FindBy(name = "user_name")             //Rule 2 : Identify all the elements using @findBy & findbys , findAll (Declaration )
     private WebElement userNameEdt;

    @FindBy(name = "user_password")
    private WebElement userPasswordEdt;

    @FindAll({@FindBy(id="submitButton") , @FindBy(xpath="//input[@id='submitButton']")})
    private WebElement loginBtn;
    //Rule 4 : Declare all the elements as private & provide getters method , business method for (Utilization)
    public WebElement getUserNameEdt() {
        return userNameEdt;
    }
    public WebElement getUserPasswordEdt() {
        return userPasswordEdt;
    }
    public WebElement getLoginBtn() {
        return loginBtn;
    }

    public void loginToApp(String userName , String password) {
        /* step 1 : login */
        userNameEdt.sendKeys(userName);
        userPasswordEdt.sendKeys(password);
        loginBtn.click();
    }
}
```

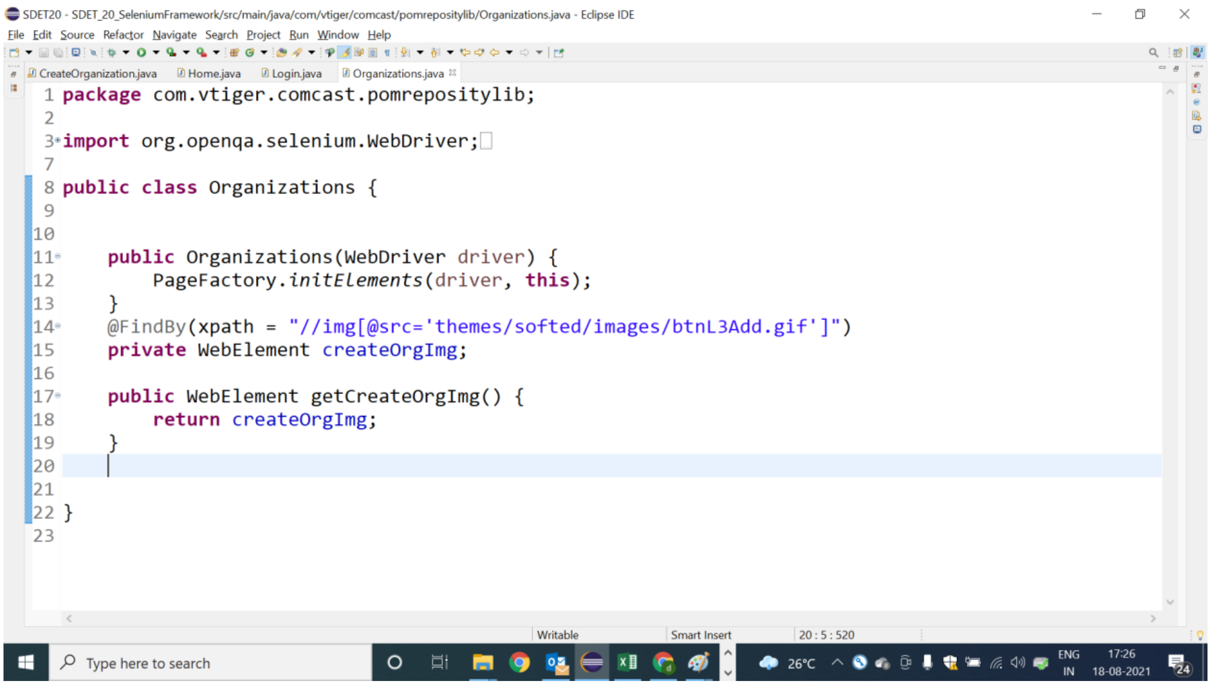
Writable | Smart Insert | 38 : 9 : 1421

Pom Classes : Home


SDET20 - SDET_20_SeleniumFramework/src/main/java/com/vtiger/comcast/pomrepositylib/Home.java - Eclipse IDE

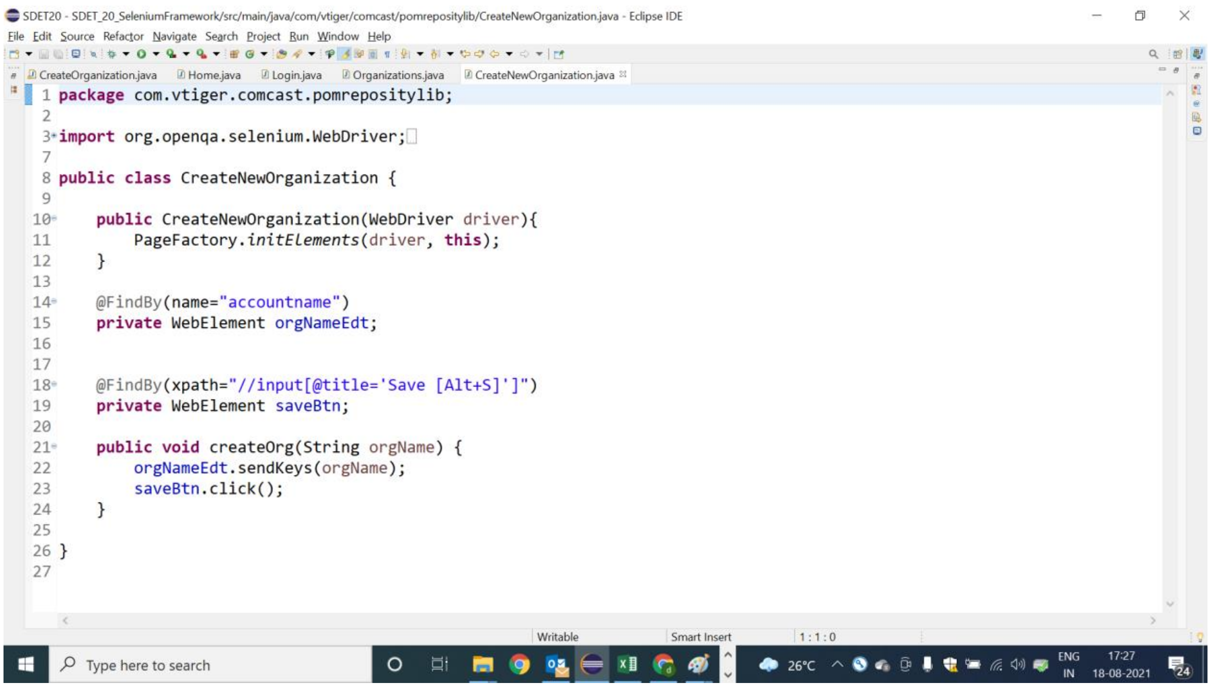
```java
package com.vtiger.comcast.pomrepositylib;

import org.openqa.selenium.WebDriver;

public class Home {
    WebDriver driver;   //global driver variable
    public Home(WebDriver driver){
        this.driver = driver;
        PageFactory.initElements(driver, this);
    }

    @FindBy(linkText = "Organizations")
    private WebElement organizationLnk;

    @FindBy(linkText = "Products")
    private WebElement productLnk;

    @FindBy(xpath = "//img[@src='themes/softed/images/user.PNG']")
    private WebElement adminstatorImg;

    @FindBy(linkText = "Sign Out")
    private WebElement singOutLnk;

    public WebElement getOrganizationLnk() {
        return organizationLnk;
    }

    public WebElement getProductLnk() {
        return productLnk;
    }
    public WebElement getAdminstatorImg() {
        return adminstatorImg;
    }

    public WebElement getSingOutLnk() {
        return singOutLnk;
    }

    public void logout() {
        Actions act = new Actions(driver);
        act.moveToElement(adminstatorImg).perform();
        singOutLnk.click();
    }

```

POM class : Organization


SDET20 - SDET_20_SeleniumFramework/src/main/java/com/vtiger/comcast/pomrepositylib/Organizations.java - Eclipse IDE

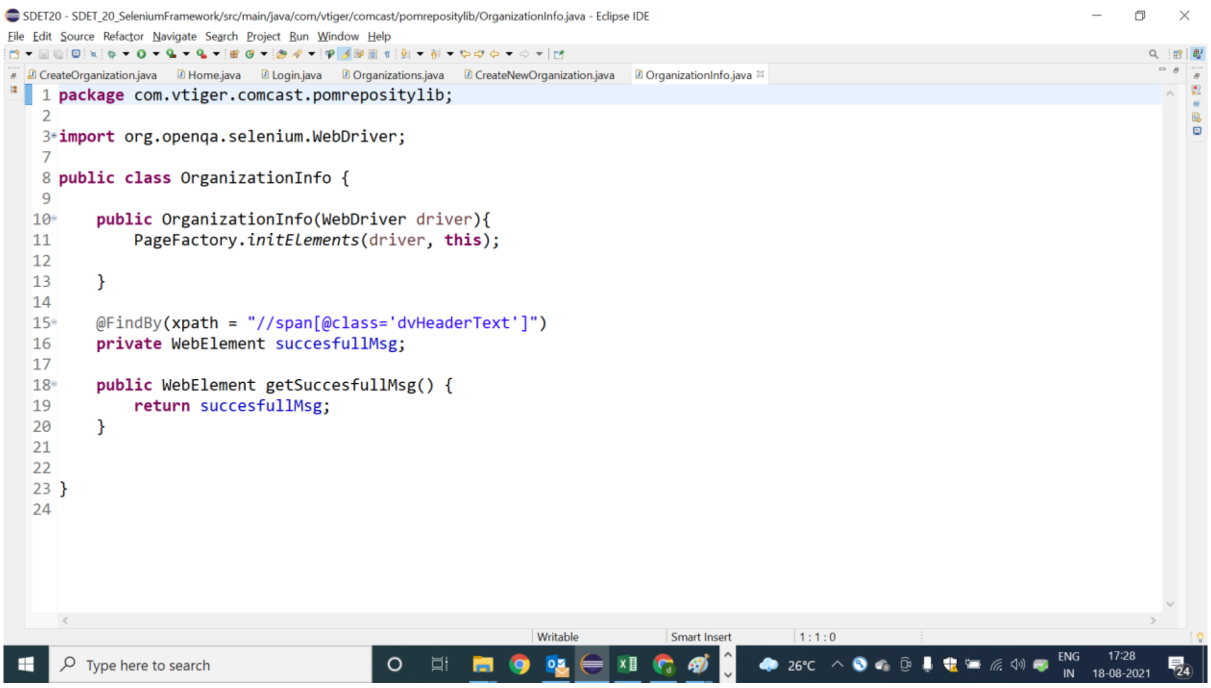
```java
package com.vtiger.comcast.pomrepositylib;

import org.openqa.selenium.WebDriver;

public class Organizations {


    public Organizations(WebDriver driver) {
        PageFactory.initElements(driver, this);
    }
    @FindBy(xpath = "//img[@src='themes/softed/images/btnL3Add.gif']")
    private WebElement createOrgImg;

    public WebElement getCreateOrgImg() {
        return createOrgImg;
    }


}

```

POM class : Create new Organization Page


SDET20 - SDET_20_SeleniumFramework/src/main/java/com/vtiger/comcast/pomrepositylib/CreateNewOrganization.java - Eclipse IDE

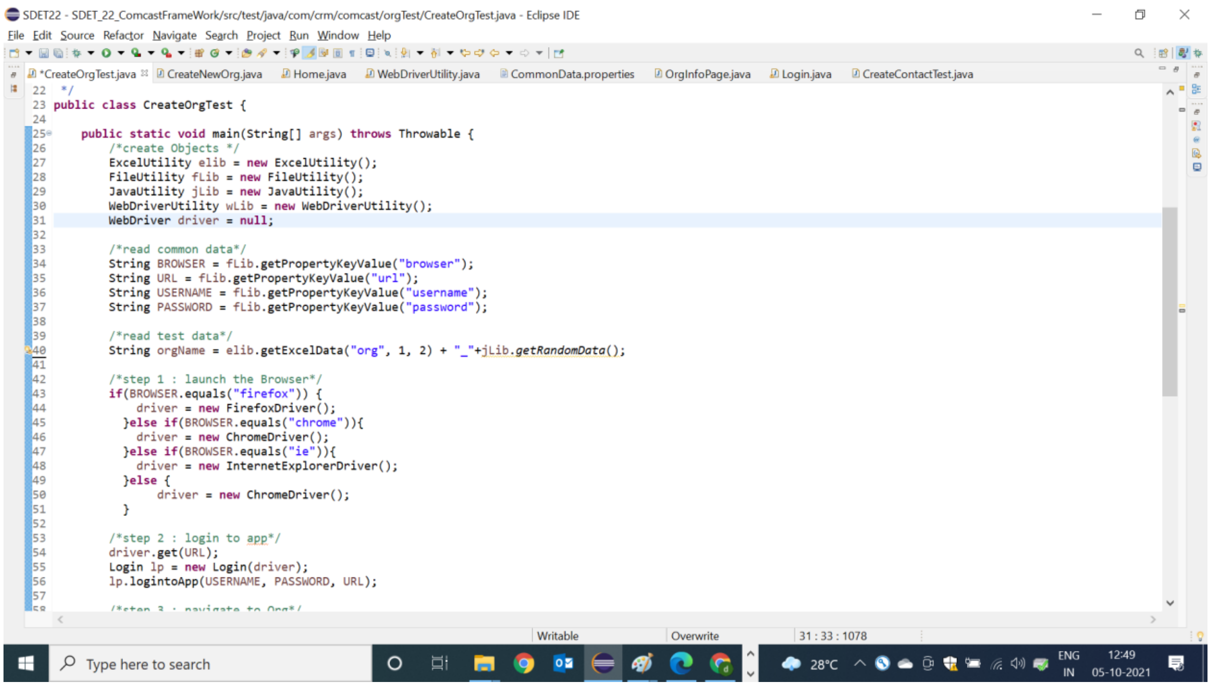
```java
package com.vtiger.comcast.pomrepositylib;

import org.openqa.selenium.WebDriver;

public class CreateNewOrganization {

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

    @FindBy(name="accountname")
    private WebElement orgNameEdt;


    @FindBy(xpath="//input[@title='Save [Alt+S]']")
    private WebElement saveBtn;

    public void createOrg(String orgName) {
        orgNameEdt.sendKeys(orgName);
        saveBtn.click();
    }

}
```

POm class : OrganizationInfo

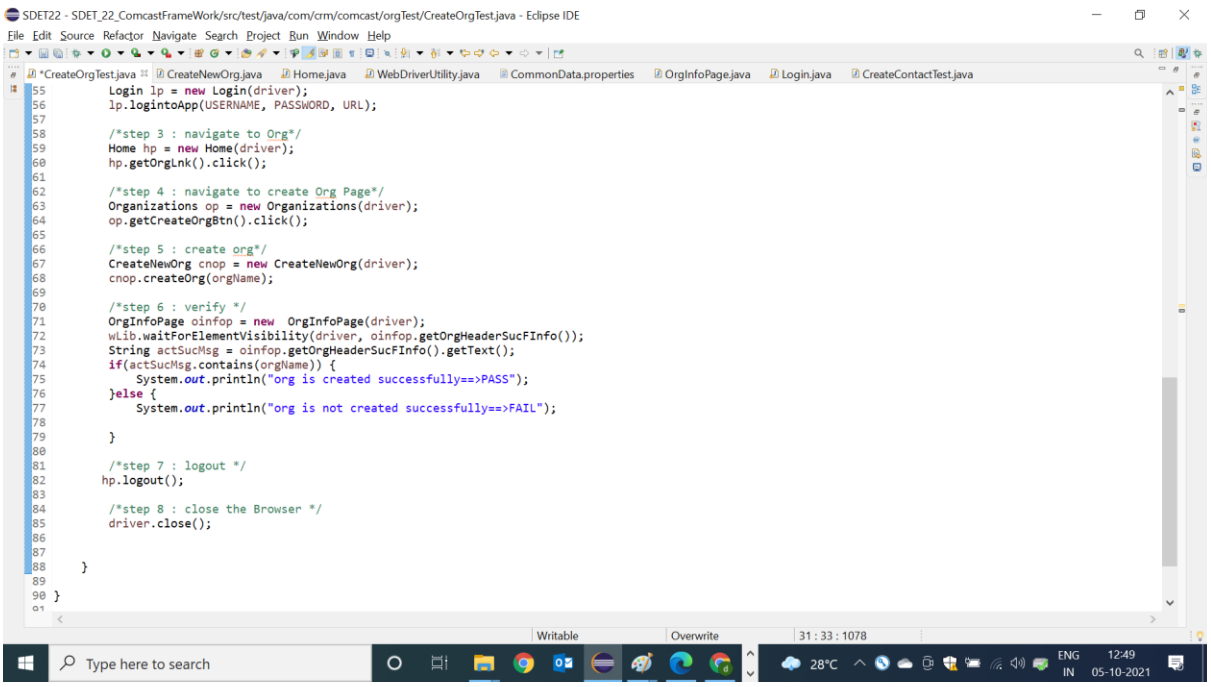

SDET20 - SDET_20_SeleniumFramework/src/main/java/com/vtiger/comcast/pomrepositylib/OrganizationInfo.java - Eclipse IDE

```java
package com.vtiger.comcast.pomrepositylib;

import org.openqa.selenium.WebDriver;

public class OrganizationInfo {

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

    }

    @FindBy(xpath = "//span[@class='dvHeaderText']")
    private WebElement succesfullMsg;

    public WebElement getSuccesfullMsg() {
        return succesfullMsg;
    }


}
```

Test Scripts using POM class



```java
 */
public class CreateOrgTest {

    public static void main(String[] args) throws Throwable {
        /*create Objects */
        ExcelUtility elib = new ExcelUtility();
        FileUtility fLib = new FileUtility();
        JavaUtility jLib = new JavaUtility();
        WebDriverUtility wLib = new WebDriverUtility();
        WebDriver driver = null;

        /*read common data*/
        String BROWSER = fLib.getPropertyKeyValue("browser");
        String URL = fLib.getPropertyKeyValue("url");
        String USERNAME = fLib.getPropertyKeyValue("username");
        String PASSWORD = fLib.getPropertyKeyValue("password");

        /*read test data*/
        String orgName = elib.getExcelData("org", 1, 2) + "_"+jLib.getRandomData();

        /*step 1 : launch the Browser*/
        if(BROWSER.equals("firefox")) {
            driver = new FirefoxDriver();
          }else if(BROWSER.equals("chrome")){
            driver = new ChromeDriver();
          }else if(BROWSER.equals("ie")){
            driver = new InternetExplorerDriver();
          }else {
               driver = new ChromeDriver();
          }

        /*step 2 : login to app*/
        driver.get(URL);
        Login lp = new Login(driver);
        lp.logintoApp(USERNAME, PASSWORD, URL);

```



```java
        Login lp = new Login(driver);
        lp.logintoApp(USERNAME, PASSWORD, URL);

        /*step 3 : navigate to Org*/
        Home hp = new Home(driver);
        hp.getOrgLnk().click();

        /*step 4 : navigate to create Org Page*/
        Organizations op = new Organizations(driver);
        op.getCreateOrgBtn().click();

        /*step 5 : create org*/
        CreateNewOrg cnop = new CreateNewOrg(driver);
        cnop.createOrg(orgName);

        /*step 6 : verify */
        OrgInfoPage oinfop = new  OrgInfoPage(driver);
        wLib.waitForElementVisibility(driver, oinfop.getOrgHeaderSucFInfo());
        String actSucMsg = oinfop.getOrgHeaderSucFInfo().getText();
        if(actSucMsg.contains(orgName)) {
            System.out.println("org is created successfully==>PASS");
        }else {
            System.out.println("org is not created successfully==>FAIL");

        }

        /*step 7 : logout */
       hp.logout();

        /*step 8 : close the Browser */
        driver.close();


    }

}
```