import org.openqa.selenium.WebDriver;

import org.openqa.selenium.By;

public class NameLocator

{

String fName;

static WebDriver driver;

public WebDriver setupDriver()

{

/\*Invoke the getWebDriver method from the DriverSetup File\*/

DriverSetup obj = new DriverSetup();

driver = obj.getWebDriver();

return driver;

}

public String getNameLocator()

{

/\*Identify the Firstname

Get the placeholder value

Store the placeholder value in the static variable fName.\*/

fName = driver.findElement(By.id("firstname")).getAttribute("placeholder");

return fName;

}

public static void main(String[] args)

{

NameLocator namLocator=new NameLocator();

namLocator.setupDriver();

String name=namLocator.getNameLocator();

System.out.println("The name is "+name);

}

}

//Add required imports

import org.openqa.selenium.WebDriver;

import org.openqa.selenium.By;

import org.openqa.selenium.WebElement;

public class PageLocator //DO NOT Change the class Name

{

String lName;

WebElement lNameElm;

static WebDriver driver;

public WebDriver createDriver() //DO NOT change the method signature

{

//Invoke getWebDriver method from DriverSetup and return it

DriverSetup obj = new DriverSetup();

driver = obj.getWebDriver();

return driver;

}

public WebElement getPageLocator(WebDriver driver) //DO NOT change the method signature

{

/\*Replace this comment by the code statement to get the WebElement of 'Lastname'\*/

/\*Find the element by id \*/

lNameElm = driver.findElement(By.id("lastname"));

return lNameElm;

}

public String getName(WebElement element) //DO NOT change the method signature

{

//Get the attribute value from the element and return it

lName= element.getAttribute("placeholder");

return lName;

}

public static void main(String[] args){

PageLocator pl=new PageLocator();

//Add required code

driver =pl.createDriver();

WebElement elmn =pl.getPageLocator(driver);

String name = pl.getName(elmn);

System.out.println("The name is "+name);

}

}

3)

import org.openqa.selenium.WebDriver;

import org.openqa.selenium.By;

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

import org.openqa.selenium.WebElement;

public class Registration //DO NOT Change the class Name

{

static String baseUrl; //Assign the value for baseUrl

static WebDriver driver;

public WebDriver createDriver() //DO NOT change the method signature

{

//Implement code to create Driver from DriverSetup and return it

//use getWebDriver method

DriverSetup obj = new DriverSetup();

driver = obj.getWebDriver();

return driver;

}

public void navigate(WebDriver driver) //DO NOT change the method signature

{

// set the url to baseUrl and navigate ("http://webapps.tekstac.com/InvoiceUpdates/")

baseUrl = "http://webapps.tekstac.com/InvoiceUpdates/";

driver.get(baseUrl);

}

public Select getCategoryElement(WebDriver driver) //DO NOT change the method signature

{

/\*Replace this comment by the code statement to get the driver\*/

/\* Select Value "Utility Invoice" from 'Category' (drop-down) and return the select element

\* Declare the drop-down element as an instance of the Select class.

\* Return select object \*/

Select droplist = new Select(driver.findElement(By.cssSelector("select")));

droplist.selectByValue("utility invoice");

return droplist;

}

public void setFormValues(WebDriver driver) //DO NOT change the method signature

{

/\*Using the driver, Find the elements by id and send the values to the elements \*/

/\* Send 'Rakesh' for 'name'

'123' for 'number'

'new user' for user type

'Utility Invoice' for select

'1000' for 'amount'

'9876543210' for phone

'New User Invoice' for 'comments'

\*/

driver.findElement(By.id("name")).sendKeys("Rakesh");

driver.findElement(By.id("number")).sendKeys("123");

driver.findElement(By.id("newUser")).click();

getCategoryElement(driver);

driver.findElement(By.name("amount")).sendKeys("1000");

driver.findElement(By.name("num")).sendKeys("9876543210");

driver.findElement(By.name("comments")).sendKeys("New User Invoice");

driver.findElement(By.id("submit")).click();

}

public WebElement getSuccessMessageElement(WebDriver driver) //DO NOT change the method signature

{

//Get the message element and return it

//After registration a success message is displayed. Identify and return the WebElement of the same.

WebElement result = driver.findElement(By.id("result"));

return result;

}

public String getSuccessMessage(WebElement element) //DO NOT change the method signature

{

//Get the attribute value from the WebElement of success message displayed and return it.

String text = element.getText();

return text;

}

public static void main(String[] args){

Registration reg=new Registration();

//Add required code

driver = reg.createDriver();

reg.navigate(driver);

reg.setFormValues(driver);

WebElement welmn = reg.getSuccessMessageElement(driver);

String text1 = reg.getSuccessMessage(welmn);

System.out.println(text1);

}

}

import org.openqa.selenium.WebDriver;

import org.openqa.selenium.By;

public class NameLocator

{

static String page1, page2,page3,page4,page5,baseUrl;

static WebDriver driver;

public WebDriver setupDriver()

{

/\*Replace this comment by the code statement to get the driver\*/

DriverSetup obj = new DriverSetup();

driver = obj.getWebDriver();

return driver;

}

public void navigate1()

{

/\*Replace this comment by the code statement to navigate to baseUrl \*/

baseUrl = "https://www.selenium.dev/";

driver.get(baseUrl);

}

public void getPageTitle1()

{

/\*Replace this comment by the code statement to get page title of "https://selenium.dev/"\*/

/\* Assign the page title to variable 'page1' \*/

page1= driver.getTitle();

}

public void navigate2()

{

/\*Replace this comment by the code statement to navigate to https://www.google.co.in/ \*/

driver.navigate().to("https://www.google.co.in/");

}

public void getPageTitle2()

{

/\*Replace this comment by the code statement to get page title of https://www.google.co.in/ \*/

/\* Assign the page title to variable 'page2' \*/

page2= driver.getTitle();

}

public void navigateBack()

{

/\*Replace this comment by the code statement to navigate back \*/

driver.navigate().back();

}

public void getPageTitle3()

{

/\*Replace this comment by the code statement to get page title of backed page \*/

/\* Assign the page title to variable 'page3' \*/

page3= driver.getTitle();

}

public void navigateForward()

{

/\*Replace this comment by the code statement to navigate forward \*/

driver.navigate().forward();

}

public void getPageTitle4()

{

/\*Replace this comment by the code statement to get page title of the forwarded page \*/

/\* Assign the page title to variable 'page4' \*/

page4= driver.getTitle();

}

public void refreshPage()

{

/\*Replace this comment by the code statement to refresh the page \*/

driver.navigate().refresh();

}

public void getPageTitle5()

{

/\*Replace this comment by the code statement to get page title of refreshed page \*/

/\* Assign the page title to variable 'page5' \*/

page5= driver.getTitle();

}

public static void main(String[] args)

{

NameLocator namLocator=new NameLocator();

//Implement code here

namLocator.setupDriver();

namLocator.navigate1();

namLocator.getPageTitle1();

namLocator.navigate2();

namLocator.getPageTitle2();

namLocator.navigateBack();

namLocator.getPageTitle3();

namLocator.navigateForward();

namLocator.getPageTitle4();

namLocator.refreshPage();

namLocator.getPageTitle5();

}

}

5)

import org.openqa.selenium.WebDriver;

import org.openqa.selenium.By;

public class Registration

{

String fName;

static WebDriver driver;

static String baseUrl;

public WebDriver setupDriver()

{

/\* Invoke the getWebDriver method

Set value of BaseUrl

Launch the app using get() with baseUrl \*/

DriverSetup obj = new DriverSetup();

driver = obj.getWebDriver();

baseUrl = "http://webapps.tekstac.com/Shopify/";

driver.get(baseUrl);

return driver;

}

public void setElements()

{

/\*Using the driver, Find the elements by id

Set the values to the elements

Register the form\*/

driver.findElement(By.id("firstname")).sendKeys("Rahul");

driver.findElement(By.id("lastname")).sendKeys("Dravid");

driver.findElement(By.id("username")).sendKeys("Rahul Dravid");

driver.findElement(By.id("pass")).sendKeys("12345");

driver.findElement(By.id("reg")).click();

}

public static void main(String[] args)

{

Registration reg=new Registration();

/\* Implement the code here \*/

reg.setupDriver();

reg.setElements();

}

}

6)

import org.openqa.selenium.WebDriver;

import org.openqa.selenium.By;

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

import org.openqa.selenium.WebElement;

public class Registration

{

static WebDriver driver;

static String baseUrl;

public WebDriver setupDriver()

{

//Assign the value for baseUrl

/\* Get the driver, and launch the app using get() with baseUrl \*/

DriverSetup obj = new DriverSetup();

driver = obj.getWebDriver();

baseUrl = "http://webapps.tekstac.com/Shopify/";

driver.get(baseUrl);

return driver;

}

public void setElements()

{

/\*Using the driver, Find the elements by id and send the values to the elements\*/

driver.findElement(By.id("firstname")).sendKeys("Mithali");

driver.findElement(By.id("lastname")).sendKeys("Raj");

driver.findElement(By.id("username")).sendKeys("Mithali Raj");

Select drpdwn = new Select(driver.findElement(By.id("selectcity")));

drpdwn.selectByValue("mas");

driver.findElement(By.xpath("//\*[@value='female']")).click();

driver.findElement(By.id("pass")).sendKeys("MR@123");

driver.findElement(By.id("reg")).click();

}

public static void main(String[] args)

{

Registration reg=new Registration();

//Implement Code Here

reg.setupDriver();

reg.setElements();

}

}

CSSLocator

//Add required imports

import java.util.List;

import org.openqa.selenium.By;

import org.openqa.selenium.WebDriver;

import org.openqa.selenium.WebElement;

public class CSSLocator //DO NOT change the class name

{

static WebDriver driver;

public WebDriver createDriver() //DO NOT change the method signature

{

//Implement code to create Driver from DriverSetup and return it

DriverSetup obj = new DriverSetup();

driver = obj.getWebDriver();

return driver;

}

public WebElement getCSSLocator(WebDriver driver) //DO NOT change the method signature

{

/\*Replace this comment by the code statement to get the Web element of username\*/

/\*Find and return the element \*/

WebElement elmn = driver.findElement(By.cssSelector("input#username"));

return elmn;

}

public String getName(WebElement element) //DO NOT change the method signature

{

//Get the attribute value from the element and return it

String val = element.getAttribute("placeholder");

return val;

}

public static void main(String[] args){

CSSLocator pl=new CSSLocator();

//Add required code

driver = pl.createDriver();

WebElement elmn = pl.getCSSLocator(driver);

String value = pl.getName(elmn);

System.out.println(value);

}

}

Absolute xpath locator

//Add required imports

import java.util.List;

import org.openqa.selenium.By;

import org.openqa.selenium.WebDriver;

import org.openqa.selenium.WebElement;

public class AbsoluteXpathLocator //DO NOT Change the class Name

{

static WebDriver driver;

public WebDriver createDriver()

{

//Implement code to create Driver from DriverSetup and return it

DriverSetup obj = new DriverSetup();

driver = obj.getWebDriver();

return driver;

}

public WebElement getAbsoluteXpathLocator(WebDriver driver)//DO NOT change the method signature

{

/\*Replace this comment by the code statement to get the Web element of logo \*/

/\*Find and return the element \*/

WebElement val = driver.findElement(By.xpath("/html/body/form/div[1]/img"));

return val;

}

public String getName(WebElement element) //DO NOT change the method signature

{

//Get the attribute value from the element and return it

String value = element.getAttribute("src");

return value;

}

public static void main(String[] args){

AbsoluteXpathLocator pl=new AbsoluteXpathLocator();

//Add required code

pl.createDriver();

WebElement elm = pl.getAbsoluteXpathLocator(driver);

String val = pl.getName(elm);

System.out.println(val);

}

}

Relativexpathlocator

//Add required imports

import java.util.List;

import org.openqa.selenium.By;

import org.openqa.selenium.WebDriver;

import org.openqa.selenium.WebElement;

public class RelativeXpathLocator //DO NOT Change the class Name

{

static WebDriver driver;

public WebDriver createDriver() //DO NOT change the method signature

{

//Implement code to create Driver from DriverSetup and return it

DriverSetup obj = new DriverSetup();

driver = obj.getWebDriver();

return driver;

}

public WebElement getRelativeXpathLocator(WebDriver driver)//DO NOT change the method signature

{

/\*Replace this comment by the code statement to get the Web element \*/

/\*Find and return the element \*/

WebElement element = driver.findElement(By.xpath("//\*[@id='ttab']/tbody/tr[2]/td[3]"));

return element;

}

public String getName(WebElement element)//DO NOT change the method signature

{

//Get the attribute value from the element and return it

String text = element.getText();

return text;

}

public static void main(String[] args){

RelativeXpathLocator pl=new RelativeXpathLocator();

//Add required code

pl.createDriver();

WebElement elm = pl.getRelativeXpathLocator(driver);

String val = pl.getName(elm);

System.out.println(val);

}

}

Handling RegEx Selenium

import org.openqa.selenium.WebDriver;

import org.openqa.selenium.By;

import org.openqa.selenium.WebElement;

import java.util.regex.Matcher;

import java.util.regex.Pattern;

public class NameLocator { //DO NOT change the class name

public static String baseUrl; //Assign 'http://webapps.tekstac.com/Handling\_Regular\_Expression/' for baseUrl

public static WebDriver driver;

public static WebElement elem;

public WebDriver createDriver()

{

//Create driver. Assign it to static variable 'driver' and return it

DriverSetup obj = new DriverSetup();

driver = obj.getWebDriver();

baseUrl="http://webapps.tekstac.com/Handling\_Regular\_Expression/";

return driver;

}

public void navigate(WebDriver driver){

//Navigate to the baseUrl

driver.get(baseUrl);

}

public void setFormValues(WebDriver driver)

{

//set the value for 'Shipment for user' and submit form

driver.findElement(By.id("userId")).sendKeys("Shamili");

driver.findElement(By.id("track")).click();

}

public WebElement getNameResultElement(WebDriver driver) {

//Find the element of 'Shamili' and return it

WebElement element =driver.findElement(By.xpath("//\*[@id='result']/table/tbody/tr[1]/td[2]"));

return element;

}

public WebElement getShipmentResultElement(WebDriver driver) {

//Find the element of 'SHIP1236' and return it

WebElement element1 =driver.findElement(By.xpath("//\*[@id='result']/table/tbody/tr[2]/td[2]"));

return element1;

}

public WebElement getEmailResultElement(WebDriver driver) {

//Find the element of 'shamili93@gamil.com' and return it

WebElement element2 =driver.findElement(By.xpath("//\*[@id='result']/table/tbody/tr[4]/td[2]"));

return element2;

}

public boolean validateEmail(String eMailID) {

//Validate email using regex.

String regex = "\\b[A-Z0-9a-z-]+@[a-z]+\\.[a-z]{2,4}\\b";

Pattern pattern = Pattern.compile(regex);

Matcher matcher = pattern.matcher(eMailID);

if (!matcher.matches()) {

return false;

}else{

return true;

}

}

public boolean validateShipmentId(String shipmentId) {

//Validate shipmentId using regex

String regex = "[A-Z0-9]{8}";

Pattern pattern = Pattern.compile(regex);

Matcher matcher = pattern.matcher(shipmentId);

if (!matcher.matches()) {

return false;

}else{

return true;

}

}

public static void main(String[] args)

{

NameLocator reg=new NameLocator();

//Add required code here

reg.createDriver();

reg.navigate(driver);

reg.setFormValues(driver);

reg.getNameResultElement(driver);

elem = reg.getShipmentResultElement(driver);

String id = elem.getText();

elem = reg.getEmailResultElement(driver);

String email = elem.getText();

reg.validateEmail(email);

reg.validateShipmentId(id);

}

}

## xPathAncestor

import org.openqa.selenium.WebDriver;

import org.openqa.selenium.By;

import org.openqa.selenium.WebElement;

import java.util.concurrent.TimeUnit;

public class NameLocator //DO NOT Change the class name

{

//Use the declared variables for stroing required values

static String fName;

static WebDriver driver;

public WebDriver setupDriver() //DO NOT Change the method Signature

{

/\* Replace this comment by the code statement to create and return the driver \*/

/\* Naviaget to the url 'http://webapps.tekstac.com/AddressBook/' \*/

DriverSetup obj = new DriverSetup();

driver = obj.getWebDriver();

driver.get("http://webapps.tekstac.com/AddressBook/");

return driver;

}

public String getNameLocator() //DO NOT Change the method Signature

{

/\*Using the driver, Find the element ancestor and its text and assign the text to 'fName' \*/

/\*Close the driver\*/

fName =driver.findElement(By.xpath("//\*[@id='xxx']/tbody/tr[1]/td[1]/ancestor::div[@id='t1']")).getText();

driver.close();

return fName;

}

public static void main(String[] args)

{

NameLocator namLocator=new NameLocator();

//Add required code here

namLocator.setupDriver();

String val = namLocator.getNameLocator();

System.out.println(val);

}

}

Work with alerts

import org.openqa.selenium.WebDriver;

import org.openqa.selenium.By;

import org.openqa.selenium.WebElement;

import java.util.concurrent.TimeUnit;

import org.openqa.selenium.Alert;

public class AlertTest //DO NOT Change the class Name

{

public static WebDriver driver;

public WebDriver createDriver() //DO NOT change the method signature

{

//Implement code to create Driver from DriverSetup, assign it to 'static' variable and return it

DriverSetup obj = new DriverSetup();

driver = obj.getWebDriver();

return driver;

}

public Alert getAlertElement(WebDriver driver) //DO NOT change the method signature

{

//Find the 'click' buttton and click it.

//Locate the 'Alert' element and return it

driver.findElement(By.name("submit")).click();

Alert alert = driver.switchTo().alert();

return alert;

}

public static void main(String[] args) throws InterruptedException //DO NOT change the method signature

{

AlertTest at=new AlertTest();

//Implement code here

at.createDriver();

Alert al = at.getAlertElement(driver);

al.accept();

}

}

## Applying POI - 1

import java.io.File;

import java.io.FileInputStream;

import java.io.FileOutputStream;

import java.util.Arrays;

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

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

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

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

import org.apache.poi.ss.usermodel.\*;

public class CusRegExcel { //Do NOT change the class name

public static Object[][] readExcelData(String sheetName){ //DO NOT change the method signature

//Implement code to read data from excel sheet in a 2-D array. Return the array

XSSFRow row;

XSSFCell cell;

String[][] excelData = null;

try {

FileInputStream inputStream = new FileInputStream(System.getProperty("user.home") + File.separator +"CustReg.xlsx");

XSSFWorkbook workbook = new XSSFWorkbook(inputStream);

// get 0th sheet data

XSSFSheet mySheet = workbook.getSheet(sheetName);

// get number of rows from sheet

int numRows = mySheet.getLastRowNum()+1;

int numCols = mySheet.getRow(0).getLastCellNum();

excelData = new String[numRows][numCols];

for (int i=0; i<numRows; i++) {

row = mySheet.getRow(i);

for (int j=0; j<numCols; j++) {

cell = row.getCell(j);

String value1 = String.valueOf(cell);;

excelData[i][j] = value1;

}

}

} catch (Exception e) {

e.printStackTrace();

}

return excelData;

}

public static String getExcelPath(String sheetName)

{

String path = null;

//Implement code to locate the excel file. Return the filepath

try{

path = System.getProperty("user.home") + File.separator +"CustReg.xlsx";

FileInputStream fis = new FileInputStream(path);

XSSFWorkbook workbook = new XSSFWorkbook(fis);

XSSFSheet sheet = workbook.getSheet(sheetName);

} catch (Exception e) {

e.printStackTrace();

}

return path;

}

public static void main(String[] args) throws Exception

{

CusRegExcel customer = new CusRegExcel();

//Add required code

customer.getExcelPath("customervalid");

customer.readExcelData("customervalid");

}

}

## Applying POI - 2

import java.io.File;

import java.io.FileInputStream;

import java.io.FileOutputStream;

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

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

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

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

import org.apache.poi.ss.usermodel.\*;

import java.util.\*;

public class CusRegExcel { //DO NOT change the class name

//Wrire the below values into excel sheet. DO NOT change the values

static String[] fields1 = new String[] { "Tester", "32" , "XYZ", "2323245235", "test1@gmail.com"};

static String[] fields2 = new String[] { "Tester1", "33" , "ABC", "4323245125", "test2@gmail.com"};

static String[] fields3 = new String[] { "Tester2", "34" , "KLM", "1323245235", "test3@gmail.com"};

public static String getExcelPath(String sheetName) { //DO NOT change the method signature

//Implement code to locate excelsheet.

//Return the filepath

String path = null;

//Implement code to locate the excel file. Return the filepath

try{

path = System.getProperty("user.home") + File.separator +"CustReg.xlsx";

FileInputStream fis = new FileInputStream(path);

XSSFWorkbook workbook = new XSSFWorkbook(fis);

XSSFSheet sheet = workbook.getSheet(sheetName);

} catch (Exception e) {

e.printStackTrace();

}

return path;

}

public static void writeExcelData(String sheetName) { //DO NOT change the method signature

//Implement code to write the given fields1,fields2,fields3 data to excel sheet.

//Write from row zero(without any headers)

XSSFRow nRow;

try {

FileInputStream inputStream = new FileInputStream(System.getProperty("user.home") + File.separator +"CustReg.xlsx");

XSSFWorkbook workbook = new XSSFWorkbook(inputStream);

// get 0th sheet data

XSSFSheet mySheet = workbook.getSheet(sheetName);;

// get number of rows from sheet

Map < String, String[] > map = new TreeMap ();

// Call put(Object key, Object value) method to add the entry in the map.

map.put( "1", fields1);

map.put( "2", fields2);

map.put( "3", fields3);

Set < String > key = map.keySet(); // Return type of keySet method is a Set.

int row = 2;

// Iterating rows using Enhanced for loop.

for (String str : key)

{

nRow = mySheet.createRow(row++);

String [] strArray = map.get(str);

int cell = 0;

for (String obj : strArray)

{

Cell nCell = nRow.createCell(cell++);

nCell.setCellValue((String)obj);

}

}

// Write the workbook in the file system.

FileOutputStream fos = new FileOutputStream(System.getProperty("user.home") + File.separator +"CustReg.xlsx");

workbook.write(fos);

fos.close();

} catch (Exception e) {

e.printStackTrace();

}

}

public static void main(String[] args) throws Exception

{

CusRegExcel customer = new CusRegExcel();

//Add required code

customer.getExcelPath("customervalid");

customer.writeExcelData("customervalid");

}

}

## Applying POI With POM - 1

import java.io.File;

import java.io.FileInputStream;

import java.io.FileOutputStream;

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

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

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

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

import org.apache.poi.ss.usermodel.\*;

import org.apache.poi.ss.usermodel.DataFormatter;

import java.io.IOException;

public class CusRegExcel { //Do not change the class name

//Use this declaration to store values parsed from excel

public static String[] customerData=new String[5];

public static String[] readExcelData(String fileName) throws IOException{ //Do not change the method signature

//Implement code to read data from excel file. Store the values in 'customerData' array. Return the array. \*/

XSSFRow row;

XSSFCell cell;

FileInputStream inputStream = new FileInputStream(System.getProperty("user.home") + File.separator +fileName);

XSSFWorkbook workbook = new XSSFWorkbook(inputStream);

DataFormatter df = new DataFormatter();

// get 0th sheet data

XSSFSheet mySheet = workbook.getSheetAt(0);

// get number of rows from sheet

int numRows = mySheet.getLastRowNum()+1;

int numCols = mySheet.getRow(0).getLastCellNum();

for (int i=0; i<numRows; i++) {

row = mySheet.getRow(i);

for (int j=0; j<numCols; j++) {

cell = row.getCell(j);

String value1 = df.formatCellValue(cell);

customerData[j]= value1;

}

}

return customerData;

}

}

import java.util.regex.Pattern;

import java.util.concurrent.TimeUnit;

import org.junit.\*;

import static org.junit.Assert.\*;

import static org.hamcrest.CoreMatchers.\*;

import org.openqa.selenium.\*;

import org.openqa.selenium.firefox.FirefoxDriver;

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

import java.io.IOException;

public class SeleniumTestForm { //Do not change the class name

public static WebDriver driver;

public static String[] Data=new String[5];

public void createDriver() { //Do not change the method signature

//Implement code to create driver and assign it to 'static' driver variable

DriverSetup obj = new DriverSetup();

driver = obj.getWebDriver();

}

public void testSeleniumTestForm() { //Do not change the method signature

//Read the data from excel sheet. Sheet name is 'customervalid'

//find the elements in the form and set values parsed from excel sheet. Submit the form for registration

try{

Data = CusRegExcel.readExcelData("CustReg.xlsx");

}catch(Exception e){

e.printStackTrace();

}

driver.findElement(By.name("cname")).sendKeys(Data[0]);

driver.findElement(By.name("age")).sendKeys(Data[1]);

driver.findElement(By.name("address")).sendKeys(Data[2]);

driver.findElement(By.name("phonenumber")).sendKeys(Data[3]);

driver.findElement(By.name("email")).sendKeys(Data[4]);

driver.findElement(By.id("submit")).click();

}

public void closeBrowser(){

//close the browser

driver.close();

}

public static void main(String[] args) throws Exception

{

CusRegExcel customer = new CusRegExcel();

//Add required code

SeleniumTestForm sf = new SeleniumTestForm();

sf.createDriver();

sf.testSeleniumTestForm();

sf.closeBrowser();

}

}

## Applying POI With POM - 2

import java.util.regex.Pattern;

import java.util.concurrent.TimeUnit;

import org.junit.\*;

import static org.junit.Assert.\*;

import static org.hamcrest.CoreMatchers.\*;

import org.openqa.selenium.\*;

import org.openqa.selenium.chrome.ChromeDriver;

import org.openqa.selenium.firefox.FirefoxDriver;

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

public class SeleniumTestForm { //DO NOT change the class name

//Use this declarations to store respective values

public static WebDriver driver;

public static String result;

public static String msg;

public void createDriver() {

//Implement code to create driver and assign it to 'static' driver variable

DriverSetup obj = new DriverSetup();

driver = obj.getWebDriver();

}

public void testSeleniumTestForm(String weight,String transportMode) throws Exception {

//'weight' and 'transportMode' parameter are used to send the values to the form

//find the elements in the form and set values as per description. Submit the form.

//Find the element of the message displayed after submit. Store it in 'msg' variable.

//Store the 'pass' or 'fail' in the 'result' variable

driver.findElement(By.id("weight")).sendKeys(weight);

driver.findElement(By.id(transportMode)).click();

driver.findElement(By.id("calculate")).click();

msg= driver.findElement(By.id("result")).getText();

if(msg.equals("Dear Customer, your total shipping cost is $200"))

result ="pass";

else

result ="fail";

}

public void closeBrowser(){

//close the browser

driver.close();

}

public static void main(String[] args) throws Exception

{

SeleniumTestForm st=new SeleniumTestForm();

//Add code here

st.createDriver();

st.testSeleniumTestForm("200", "air");

st.closeBrowser();

CargoShipping cs = new CargoShipping();

cs.writeExcelData("cargo.xlsx", result);

}

}

import java.io.File;

import java.io.FileInputStream;

import java.io.FileOutputStream;

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

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

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

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

import org.apache.poi.ss.usermodel.\*;

public class CargoShipping //DO NOT change the class name

{

public static void writeExcelData(String fileName,String result) throws Exception { //Do not change the method signature

//Write the Test result to the excel sheet

FileOutputStream out = new FileOutputStream((System.getProperty("user.home") + File.separator +fileName));

XSSFWorkbook workbook = new XSSFWorkbook();

XSSFSheet mySheet = workbook.createSheet("TestCase");

XSSFRow row = mySheet.createRow(0);

// Specific cell number

Cell cell = row.createCell(0);

// putting value at specific position

cell.setCellValue(result);

// writing the content to Workbook

workbook.write(out);

out.close();

workbook.close();

}

}

## TestNG Annotations - 1

import java.io.IOException;

import java.lang.annotation.Annotation;

import java.lang.reflect.Method;

import java.util.List;

import org.openqa.selenium.WebDriver;

import org.testng.Assert;

import org.testng.TestNG;

import org.testng.annotations.\*;

import org.testng.annotations.AfterSuite;

import org.testng.annotations.BeforeSuite;

import org.testng.annotations.Test;

import org.testng.collections.Lists;

//ADD the required TestNG annotations on respective methods

public class Ex1TestNG { //DO NOT Change the class Name

//Do NOT change these below declarations.

public static WebDriver driver;

String name;

String password;

@BeforeSuite

public void createDriver() { //DO NOT change the method signature

//Implement code to create Driver from DriverSetup and assign it to 'static' driver variable

DriverSetup ds = new DriverSetup();

driver = ds.getWebDriver();

}

@Test(groups= {"init"})

public void initializeName() { //DO NOT change the method signature

name="Tom";

}

@Test(groups= {"init"})

public void initializePassword() { //DO NOT change the method signature

password="Tom";

}

@Test(dependsOnGroups= {"init"})

public void signin() { //DO NOT change the method signature

System.out.println("signin");

}

@AfterSuite

public void closeBrowser() { //DO NOT change the method signature

driver.close();

}

public static void main(String[] args){ //DO NOT change the method signature

Ex1TestNG page=new Ex1TestNG();

//Implement the required code

TestNG testNG = new TestNG();

testNG.setTestClasses(new Class[] {

Ex1TestNG.class });

testNG.run();

}

}

[TestNG Annotations - 2](https://cognizant-assessment.tekstac.com/mod/vpl/view.php?id=972)

import java.io.IOException;

import java.lang.annotation.Annotation;

import java.lang.reflect.Method;

import java.util.Iterator;

import java.util.List;

import org.junit.AfterClass;

import org.junit.BeforeClass;

import org.openqa.selenium.WebDriver;

import org.testng.Assert;

import org.testng.TestNG;

import org.testng.annotations.AfterSuite;

import org.testng.annotations.BeforeMethod;

import org.testng.annotations.BeforeSuite;

import org.testng.annotations.BeforeTest;

import org.testng.annotations.Listeners;

import org.testng.annotations.Test;

import org.testng.collections.Lists;

@Listeners(FeatureTest.class) //DO NOT remove or alter this line. REQUIRED FOR TESTING

public class Ex2TestNG { //DO NOT Change the class Name

public static WebDriver driver;

String name;

String password;

@BeforeSuite

public void createDriver() { //DO NOT change the method signature

//Create driver and assign it to 'static' driver variable

DriverSetup ds = new DriverSetup();

driver = ds.getWebDriver();

}

@BeforeTest

public void resetName() { //DO NOT change the method signature

//NO implementation required. ONLY proper annotation is required.

}

@Test(dependsOnMethods= {"initializeName","initializePassword"})

public void signin() { //DO NOT change the method signature

System.out.println("signin");

}

@Test (priority=1)

public void initializeName() { //DO NOT change the method signature

name="Tom";

}

@Test (priority=0)

public void initializePassword() { //DO NOT change the method signature

password="Tom";

}

@AfterSuite

public void closeBrowser() { //DO NOT change the method signature

//close the driver

driver.close();

}

public void invokeTest() { //DO NOT change the method signature

//Implement code to invoke test using TestNG

//DO NOT change the method signature

TestNG testNG = new TestNG();

testNG.setTestClasses(new Class[] {

Ex2TestNG.class });

testNG.run();

}

public static void main(String[] args) {

Ex2TestNG ex2=new Ex2TestNG();

//Implement any required code.

ex2.invokeTest();

}

}

## Shipment Cost - TestNG With DataProvider -1

import static org.junit.Assert.assertEquals;

import java.io.File;

import java.io.FileInputStream;

import java.util.\*;

import org.apache.poi.ss.usermodel.DataFormatter;

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

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

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.WebElement;

import org.openqa.selenium.firefox.FirefoxDriver;

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

import org.testng.TestNG;

import org.testng.annotations.BeforeMethod;

import org.testng.annotations.BeforeSuite;

import org.testng.annotations.DataProvider;

import org.testng.annotations.Listeners;

import org.testng.annotations.Test;

import org.testng.collections.Lists;

import junit.framework.Assert;

@Listeners(FeatureTest.class) //DO NOT remove or alter this line. REQUIRED FOR TESTING

public class Ex3TestNG //DO NOT Change the class Name

{

public static WebDriver driver;

@BeforeSuite

public void createDriver() { //DO NOT change the method signature

//Create driver and assign it to 'static' driver variable

DriverSetup ds = new DriverSetup();

driver = ds.getWebDriver();

}

@DataProvider

public Object[][] shipmentData() //DO NOT change the method signature

{

//Parse data from Shipmentdetails.xlsx and return the 2-dimensional array

XSSFRow row;

XSSFCell cell;

String[][] excelData = null;

try {

FileInputStream inputStream = new FileInputStream(System.getProperty("user.home") + File.separator +"ShipmentDetails.xlsx");

XSSFWorkbook workbook = new XSSFWorkbook(inputStream);

// get 0th sheet data

XSSFSheet mySheet = workbook.getSheet("Data");;

// get number of rows from sheet

int numRows = mySheet.getLastRowNum()+1;

int numCols = mySheet.getRow(0).getLastCellNum();

excelData = new String[numRows-1][numCols];

for (int i=1; i<numRows; i++) {

row = mySheet.getRow(i);

for (int j=0; j<numCols; j++) {

cell = row.getCell(j);

String value1 = String.valueOf(cell);;

excelData[i-1][j] = value1;

}

}

} catch (Exception e) {

e.printStackTrace();

}

return excelData;

}

//DO NOT change the method signature

@Test(dataProvider="shipmentData")

void testShipment (String orginPort,String destinationPort,String railModeCharge,String roadModeCharge,String airModeCharge) throws InterruptedException

{

//Select the souce/destination as specified in description.

// Test the tabled data against the excel data as specified in description.

Select origin = new Select(driver.findElement(By.name("origin\_id")));

origin.selectByIndex(0);

Select destination = new Select(driver.findElement(By.name("destination\_id")));

destination.selectByIndex(1);

driver.findElement(By.name("submit")).click();

String rail = driver.findElement(By.xpath("//\*[@name='charge']/tbody/tr[3]/td[2]")).getText();

String road = driver.findElement(By.xpath("//\*[@name='charge']/tbody/tr[2]/td[2]")).getText();

String air = driver.findElement(By.xpath("//\*[@name='charge']/tbody/tr[4]/td[2]")).getText();

String p = driver.findElement(By.xpath("//p")).getText();

try{

Assert.assertEquals(rail,railModeCharge);

}catch(AssertionError e){

System.out.println("The Rail mode Charge doesnt match");

}

try{

Assert.assertEquals(road,roadModeCharge);

}catch(AssertionError e){

System.out.println("he Road mode Charge doesnt match");

}

try{

Assert.assertEquals(air,airModeCharge);

}catch(AssertionError e){

System.out.println("he Air mode Charge doesnt match");

}

}

public void shipment() {

//Invoke the test using TestNG ONLY HERE.

TestNG testNG = new TestNG();

//Provide the list of test classes

testNG.setTestClasses(new Class[] {

Ex3TestNG.class });

testNG.run();

}

public static void main(String[] args) {

Ex3TestNG ex3=new Ex3TestNG();

//Implement any required code.

ex3.shipment();

}

}

## Commodity Details - TestNG With DataProvider - 1

import static org.junit.Assert.assertEquals;

import java.io.File;

import java.io.FileInputStream;

import java.util.Iterator;

import java.util.List;

import org.apache.poi.ss.usermodel.DataFormatter;

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

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

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.WebElement;

import org.openqa.selenium.chrome.ChromeDriver;

import org.openqa.selenium.firefox.FirefoxDriver;

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

import org.testng.TestNG;

import org.testng.annotations.BeforeMethod;

import org.testng.annotations.BeforeSuite;

import org.testng.annotations.DataProvider;

import org.testng.annotations.Listeners;

import org.testng.annotations.Test;

import org.testng.collections.Lists;

import junit.framework.Assert;

@Listeners(FeatureTest.class) //DO NOT remove or alter this line. REQUIRED FOR TESTING

public class Ex4TestNG //DO NOT Change the class Name

{

WebDriver driver;

XSSFSheet testSheet;

static int testCount=1;

@BeforeSuite

public void createDriver() { //DO NOT change the method signature

//Create driver and assign it to 'static' driver variable

DriverSetup ds = new DriverSetup();

driver = ds.getWebDriver();

}

@DataProvider(name ="commodityData")

public Object[][] commodityData() //DO NOT change the method signature

{

//Parse data from CommodityDetails.xlsx and return the 2-dimensional array

XSSFRow row;

XSSFCell cell;

String[][] excelData = null;

try {

FileInputStream inputStream = new FileInputStream(System.getProperty("user.home") + File.separator +"CommodityDetails.xlsx");

XSSFWorkbook workbook = new XSSFWorkbook(inputStream);

// get 0th sheet data

testSheet = workbook.getSheet("Data");;

// get number of rows from sheet

int numRows = testSheet.getLastRowNum()+1;

int numCols = testSheet.getRow(0).getLastCellNum();

excelData = new String[numRows-1][numCols];

for (int i=1; i<numRows; i++) {

row = testSheet.getRow(i);

for (int j=0; j<numCols; j++) {

cell = row.getCell(j);

String value1 = String.valueOf(cell);;

excelData[i-1][j] = value1;

System.out.println(excelData[i-1][j]);

}

}

} catch (Exception e) {

e.printStackTrace();

}

return excelData;

}

@Test(dataProvider="commodityData")

void testCommodity (String name,String weight,String length,String width,String height) throws InterruptedException

{

//Pass the form data parsed from excel sheet and submit form

//Find the elements of the newly displayed row on the page and compare the values against the excel data as mentioned in the description

for(int i=2;i<4;i++){

driver.findElement(By.name("name")).sendKeys(name);

driver.findElement(By.name("weight")).sendKeys(weight);

driver.findElement(By.name("length")).sendKeys(length);

driver.findElement(By.name("width")).sendKeys(width);

driver.findElement(By.name("height")).sendKeys(height);

driver.findElement(By.name("add")).click();

String nme = driver.findElement(By.xpath("//\*[@id='myTable']/tbody/tr["+i+"]/td[1]")).getText();

String wt = driver.findElement(By.xpath("//\*[@id='myTable']/tbody/tr["+i+"]/td[2]")).getText();

String lgth = driver.findElement(By.xpath("//\*[@id='myTable']/tbody/tr["+i+"]/td[3]")).getText();

String wdth = driver.findElement(By.xpath("//\*[@id='myTable']/tbody/tr["+i+"]/td[4]")).getText();

String ht = driver.findElement(By.xpath("//\*[@id='myTable']/tbody/tr["+i+"]/td[5]")).getText();

try{

Assert.assertEquals(nme,name);

}catch(AssertionError e){

System.out.println("name doesnt match");

}

try{

Assert.assertEquals(wt,weight);

}catch(AssertionError e){

System.out.println("weight doesnt match");

}

try{

Assert.assertEquals(lgth,length);

}catch(AssertionError e){

System.out.println("length doesnt match");

}

try{

Assert.assertEquals(wdth,height);

}catch(AssertionError e){

System.out.println("width doesnt match");

}

try{

Assert.assertEquals(ht,length);

}catch(AssertionError e){

System.out.println("height doesnt match");

}

}

}

public void commodity() {

//Invoke the test using TestNG ONLY HERE.

TestNG testNG = new TestNG();

//Provide the list of test classes

testNG.setTestClasses(new Class[] {

Ex4TestNG.class });

testNG.run();

}

public static void main(String[] args) {

Ex4TestNG ex4=new Ex4TestNG();

//Implement any required code.

ex4.commodity();

}

}