<https://www.geeksforgeeks.org/page-object-model-pom/>

https://www.guru99.com/page-object-model-pom-page-factory-in-selenium-ultimate-guide.html

how to use Variables in xpath in POM

String newUser = "NewUser123";

driver.findElement(By.xpath("//td[normalize-space(text())=" + newUser + "]/a")).click();

String newUser = "NewUser123";

driver.findElement(By.xpath("//td[normalize-space(text())='"+newUser+"']/a")).click();

public static void selectEvent(String eventName) {

WebElement eventLink = Browser.instance.findElement(By.xpath("//td[2]/a[1][contains(text(), '"+eventName+"')]")); eventLink.click();

}

==================================================================

Alternative method to use XPATH IN POM from a variable

WorkAround : on page class you can define a method, and can pass the text on the fly from the calling class to click on specific tab

if you want to click any common text on the page. You can create a method as given below and can pass the text on the fly to click on that specific tab on that page

public void clickTab(String tabText){

String tabxpath = "//div[contains(text(), '" + tabText + "')]";

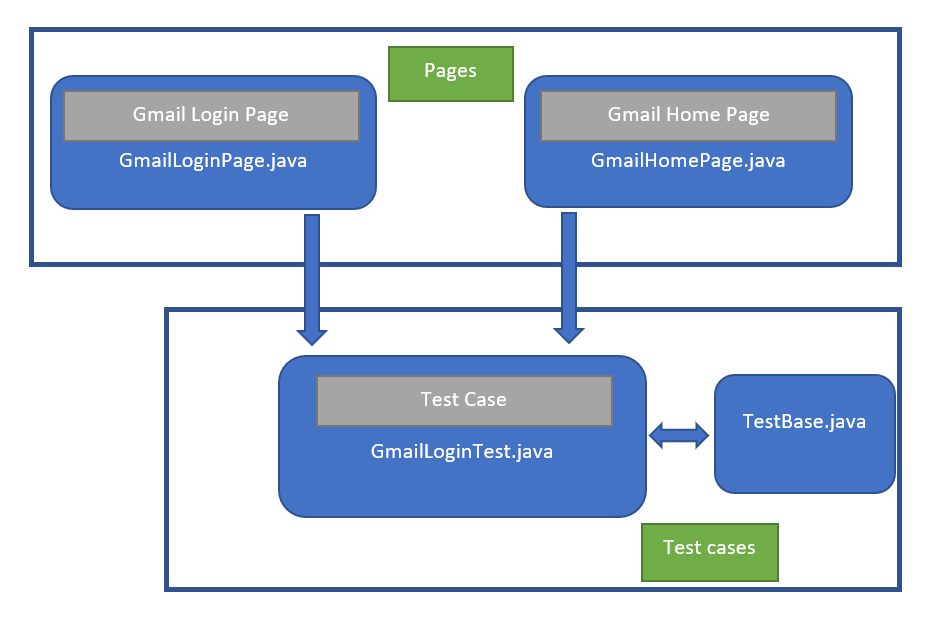
driver.findElement(By.xpath(tabxpath)).click();

}

1. public WebELement customerLink(WebDriver driver, String customerName) {
2. String customerLinkXpath = “//td[@contains(text(),'CUSTOMERNAME']/a";
3. return driver.findElement(By.xpath(customerLinkXpath.Replace("CUSTOMERNAME", customerName));
4. }

Page Object Model (POM)

**What is POM?**  
POM is a design pattern which is commonly used in Selenium for Automating the Test Cases. This design pattern can be used with any kind of framework like keyword-driven, Data-driven, hybrid framework, etc.  
The Page object is an object-oriented class which acts as an interface for the page of your Application under test. Page class contains web elements and methods to interact with web elements. While Automating the test cases, we create the object of these Page Classes and interact with web elements by calling the methods of these classes.

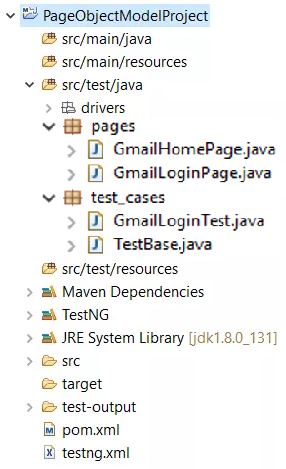
**Page Object Model Design Pattern**  


**Uses of Page Object Model (POM):**  
This Design Pattern is used in Selenium where web pages are represented by a corresponding class and web elements are represented by the variables of the class and all interactions are possible through the methods or say functions of the class.

**Advantages of POM model:**

* **Reusability:** We can reuse the page class if required in different test cases which means we don’t need to write code for identifying the web elements and methods to interact with them for every test case.
* **Maintainability:** As we can see from the above picture test case and page class are different from each other which means we can easily update the code if any new web element is added or existing one updated.
* **Readability:** As we can see in the above picture page code is separated form test code which helps to improve code readability.

**Implementation of Page Object Model:**  
Here I’m taking the example of Gmail application to show the implementation of **POM**.  
The Project Hierarchy of my **Maven Project** is shown below:

**Project Hierarchy**  


**TestBase Class:**  
In this class we create the object of **WebDriver** class, maximize the browser, launching Url etc.

filter\_none

brightness\_4

|  |
| --- |
| package test\_cases;    import java.io.IOException;  import java.util.concurrent.TimeUnit;  import org.openqa.selenium.WebDriver;  import org.openqa.selenium.chrome.ChromeDriver;  import org.testng.annotations.AfterSuite;  import org.testng.annotations.BeforeSuite;    public class TestBase {        public static WebDriver driver = null;        @BeforeSuite      public void initialize() throws IOException {            System.setProperty("webdriver.chrome.driver",          System.getProperty("user.dir") +              "\\src\\test\\java\\drivers\\chromedriver.exe");            driver = new ChromeDriver();            // To maximize browser          driver.manage().window().maximize();            // Implicit wait          driver.manage().timeouts().implicitlyWait(               10, TimeUnit.SECONDS);            // To open Gmail site          driver.get("https:// www.gmail.com");      }        @AfterSuite      // Test cleanup      public void TeardownTest() {          TestBase.driver.quit();      }  } |

**GmailLoginPage Class:**  
In this class, we identify the web elements of Gmail Login Page and methods to interact with these web elements.

filter\_none

brightness\_4

|  |
| --- |
| package pages;    import org.openqa.selenium.WebDriver;  import org.openqa.selenium.WebElement;  import org.openqa.selenium.support.FindBy;  import org.openqa.selenium.support.How;    public class GmailLoginPage {        WebDriver driver;        public GmailLoginPage(WebDriver driver) {          this.driver = driver;      }        // Using FindBy for locating elements      @FindBy(how = How.XPATH, using =              "// input[@type='email']]")      WebElement emailTextBox;        @FindBy(how = How.XPATH, using =              "// input[@type='password']")        WebElement passwordTextBox;      @FindBy(how = How.XPATH, using =              "// div[@role = 'button' and @id  =              'identifierNext']")      WebElement nextButton;        // Defining all the user actions (Methods)      //that can be performed in the Facebook home page        // This method is to set Email in the email text box      public void setEmail(String strEmail) {              emailTextBox.sendKeys(strEmail);          }          // This method is to set Password in the password text box      public void setPassword(String strPassword) {              passwordTextBox.sendKeys(strPassword);          }          // This method is to click on Next Button      public void clickOnNextButton() {          nextButton.click();      }  } |

Similarly, we write the code for Gmail Home Page.

**GmailLoginTest:**  
In this class we test the login page of Gmail.

filter\_none

brightness\_4

|  |
| --- |
| package test\_cases;    import org.openqa.selenium.support.PageFactory;  import org.testng.annotations.Test;  import pages.GmailHomePage;  import pages.GmailLoginPage;    public class GmailLoginTest extends TestBase {        @Test      public void init() throws Exception {            // driver.get("https:// www.gmail.com");          GmailLoginPage loginpage =             PageFactory.initElements(driver,             GmailLoginPage.class);            loginpage.setEmail("abc@gmail.com");          loginpage.clickOnNextButton();          loginpage.setPassword("23456@qwe");          loginpage.clickOnNextButton();      }  } |

### DeveloperApplyPage Page Object

And finally, for our third and last page object for this project, we define one that represents the page containing developer application form. Since we have to deal with a number of form fields here, we define one WebElement variable for every form field. We find each field by their “id” and we define special setter methods for every field that simulate keystrokes for the corresponding fields.

package com.toptal.webpages;

import org.openqa.selenium.WebDriver;

import org.openqa.selenium.WebElement;

import org.openqa.selenium.support.FindBy;

import org.openqa.selenium.support.PageFactory;

public class DeveloperApplyPage {

private WebDriver driver;

@FindBy(tagName = "h1")

WebElement heading;

@FindBy(id="developer\_email")

WebElement developer\_email;

@FindBy(id = "developer\_password")

WebElement developer\_password;

@FindBy(id = "developer\_password\_confirmation")

WebElement developer\_password\_confirmation;

@FindBy(id = "developer\_full\_name")

WebElement developer\_full\_name;

@FindBy(id = "developer\_skype")

WebElement developer\_skype;

@FindBy(id ="save\_new\_developer")

WebElement join\_toptal\_button;

//Constructor

public DeveloperApplyPage(WebDriver driver){

this.driver=driver;

//Initialise Elements

PageFactory.initElements(driver, this);

}

public void setDeveloper\_email(String email){

developer\_email.clear();

developer\_email.sendKeys(email);

}

public void setDeveloper\_password(String password){

developer\_password.clear();

developer\_password.sendKeys(password);

}

public void setDeveloper\_password\_confirmation(String password\_confirmation){

developer\_password\_confirmation.clear();

developer\_password\_confirmation.sendKeys(password\_confirmation);

}

public void setDeveloper\_full\_name (String fullname){

developer\_full\_name.clear();

developer\_full\_name.sendKeys(fullname);

}

public void setDeveloper\_skype (String skype){

developer\_skype.clear();

developer\_skype.sendKeys(skype);

}

public void clickOnJoin(){

join\_toptal\_button.click();

}

public boolean isPageOpened(){

//Assertion

return heading.getText().toString().contains("Apply to join our network as a developer");

}

}

|  |
| --- |
| package pages;    import org.openqa.selenium.WebDriver;  import org.openqa.selenium.WebElement;  import org.openqa.selenium.support.FindBy;  import org.openqa.selenium.support.How;    public class GmailLoginPage {        WebDriver driver;        public GmailLoginPage(WebDriver driver) {          this.driver = driver;      }  1111        // Using FindBy for locating elements      @FindBy(how = How.XPATH, using =              "// input[@type='email']]")      WebElement emailTextBox;        @FindBy(how = How.XPATH, using =              "// input[@type='password']")        WebElement passwordTextBox;      @FindBy(how = How.XPATH, using =              "// div[@role = 'button' and @id  =              'identifierNext']")      WebElement nextButton;        // Defining all the user actions (Methods)      //that can be performed in the Facebook home page        // This method is to set Email in the email text box      public void setEmail(String strEmail) {              emailTextBox.sendKeys(strEmail);          }          // This method is to set Password in the password text box      public void setPassword(String strPassword) {              passwordTextBox.sendKeys(strPassword);          }          // This method is to click on Next Button      public void clickOnNextButton() {          nextButton.click();      }  } |

Similarly, we write the code for Gmail Home Page.

**GmailLoginTest:**  
In this class we test the login page of Gmail.

*filter\_none*

*brightness\_4*

|  |
| --- |
| package test\_cases;    import org.openqa.selenium.support.PageFactory;  import org.testng.annotations.Test;  import pages.GmailHomePage;  import pages.GmailLoginPage;    public class GmailLoginTest extends TestBase {        @Test      public void init() throws Exception {            // driver.get("https:// www.gmail.com");          GmailLoginPage loginpage =             PageFactory.initElements(driver,             GmailLoginPage.class);            loginpage.setEmail("abc@gmail.com");          loginpage.clickOnNextButton();          loginpage.setPassword("23456@qwe");          loginpage.clickOnNextButton();      }  } |

# [How to select a value form drop down list using page object model?](https://sqa.stackexchange.com/questions/22995/how-to-select-a-value-form-drop-down-list-using-page-object-model)

[Ask Question](https://sqa.stackexchange.com/questions/ask)

Asked 3 years, 3 months ago

Active [3 years, 3 months ago](https://sqa.stackexchange.com/questions/22995/how-to-select-a-value-form-drop-down-list-using-page-object-model?lastactivity)

Viewed 14k times

2

1

Here is my code for client registration page.

package Pages;

import org.apache.bcel.generic.Select;

import org.openqa.selenium.WebDriver;

import org.openqa.selenium.WebElement;

import org.openqa.selenium.support.FindBy;

import org.openqa.selenium.support.PageFactory;

public class ClientPage

{

@FindBy(linkText="Admin")

WebElement admin;

@FindBy(linkText="Clients")

WebElement clients;

@FindBy(xpath="//\*[@id='search-container']/a/i")

WebElement search;

@FindBy(id="name")

WebElement name;

@FindBy(name="address")

WebElement address;

@FindBy(id="status")

WebElement select;

//constructor

public ClientPage(WebDriver driver)

{

//initialize elements

PageFactory.initElements(driver,this);

}

public void clickOnadmin()

{

admin.click();

}

public void clickonclient()

{

clients.click();

}

public void clickonsearch()

{

search.click();

}

public void set\_clientname(String na)

{

name.clear();

name.sendKeys(na);

}

public void set\_address(String add)

{

address.clear();

address.sendKeys("add");

}

public void select\_list()

{

Select stat=new Select(driver.findElement(By.id("status")));

stat.selectByVisibleText("active");

}

}

Here is the test page.

package Test;

import org.testng.annotations.Test;

import Pages.ClientPage;

import Pages.HomePage;

import org.testng.Assert;

import org.openqa.selenium.By;

import org.openqa.selenium.JavascriptExecutor;

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.annotations.AfterTest;

import org.testng.annotations.BeforeTest;

public class ClientTest

{

WebDriver driver;

@BeforeTest

public void setup()

{

System.setProperty("webdriver.firefox.marionette","pathToGeckodriver");

driver=new FirefoxDriver();

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

driver.get("http://54.68.159.204/qmsadm");

}

@Test

public void verify()

{

HomePage home=new HomePage(driver);

home.set\_username("");

home.set\_password("");

home.click\_button();

Assert.assertTrue(driver.getPageSource().contains("Hello Sony George"));

}

@Test(priority=1)

public void verify1()

{

ClientPage client=new ClientPage(driver);

client.clickOnadmin();

client.clickonclient();

Assert.assertTrue(driver.getPageSource().contains("Hello Sony George"));

}

@Test(priority=2)

public void verify2()

{

ClientPage client=new ClientPage(driver);

client.clickOnadmin();

client.clickonclient();

client.clickonsearch();

client.set\_clientname("Ajmal police");

client.set\_address("dubai");

}

}

[selenium](https://sqa.stackexchange.com/questions/tagged/selenium) [java](https://sqa.stackexchange.com/questions/tagged/java)

[share](https://sqa.stackexchange.com/q/22995)[improve this question](https://sqa.stackexchange.com/posts/22995/edit)

[edited Oct 12 '16 at 6:05](https://sqa.stackexchange.com/posts/22995/revisions)

asked Oct 10 '16 at 9:30

user21268

[add a comment](https://sqa.stackexchange.com/questions/22995/how-to-select-a-value-form-drop-down-list-using-page-object-model)

## **2 Answers**

[active](https://sqa.stackexchange.com/questions/22995/how-to-select-a-value-form-drop-down-list-using-page-object-model?answertab=active#tab-top)[oldest](https://sqa.stackexchange.com/questions/22995/how-to-select-a-value-form-drop-down-list-using-page-object-model?answertab=oldest#tab-top)[votes](https://sqa.stackexchange.com/questions/22995/how-to-select-a-value-form-drop-down-list-using-page-object-model?answertab=votes#tab-top)

1

First of all, select a value from drop down list is not unique to page object model. Even there was no page object model, you could still do it.

Your question is not clear on which the drop down list is, so here it is: From your ClientPage, you have the following elements Admin, Clients, search, name, address, status; the actions you can perform is clickOnadmin(), clickonclient(), clickonsearch(), set\_clientname(String na), set\_address(String add), select\_list(), verify(), verify1() and verify2().

**It seems the coding style is not the best it can be, can I please suggest you go through your code and make sure that all functions have the same naming convention and all arguments make sense? Below is one way to do it**

* clickOnadmin() should be clickOnAdmin()
* clickonclient() should be clickOnClient()
* clickonsearch() should be clickOnSearch()
* set\_clientname(String na) should be setClientName(String clientName)
* set\_address(String add) should be setAddress(String address)
* select\_list() should be selectList(), this function should have an input argument I think, otherwise, which one do you select from a list?
* verify(), should change its name to indicate its purpose. What does it verify? It verifies if you have navigated to the right page.
* verify1(), it is not clear what you are verifying
* verify2(), it is not clear what you are verifying

In your function verify(), can I please suggest you add a bit more to it? What happens if the page source does not contain "Hello Sony George"? Your code should navigate to the page you intended to navigate OR display a meaningful message, warning the user this is NOT the page you want to navigate to.

If the drop down list you want to select from is the Status, which has None, Inactive, Active and Archived items, you could add

public void statusDropDownList(int index) {

Select drop = new Select(status);

drop.selectByIndex(index);

}