02-MAR-2020

-----------

POM Framework

-------------

In Page Object Model framework, Object identification code and validation code are seperated

-------------------------------------------------- -------------------------------------------------------------

(Package-1: Object Repository) (Package-2: Test Cases)

Class-1{page-1 WebElement Identification code} Class-1:{Scenario-1 Validation code}

Class-2:{page-2 WebElement Identification code} Class-2:{Scenario-2 Validation code}

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

In this approach we create individual classes for every page webelements with identification code and then we create individual functions for each Webelement

to return identification value.

Advantages of POM:

------------------

In future or next versions if the object properties are changed, then we need to update only WebElement identification code

(i.e. Object Repository package classes only)

-- It will increase Readibility of the script

-- It will reduce duplication of code

-- It is early for maintainibility

------------------------------------------------------------------------------------------------------------------------------------------------------------------

\*\* In POM framework we use Page Factory class and @FindBy annotationn

Page Factory:

------------

Using this class we can initialize webelement locator value classes in required class.

Syntax:

------

ClassName Obj= PageFactory.initElements (driver,ClassName.class);

@FindBy

-------

Using this annotation we can assign Webelement locator value

Syntax:

-------

@FindBy (locatorname= locator\_value)

WebElement ele;

----------------------------------------------------------------------------------------------------------------------------------------------------------------

ex: Create script to validate login functionality in http://rediffmail.com application using POM Framework with PageFactory annotation @FindBy

Procedure:

Step1:

create 2 packages

package-1: "object.repository"

package-2: "test.cases"

Step 2:

create 2 non-executable classes in "object.repository" package

class-1: RediffHomePg

Class-2: RediffLoginPg

Step 3:

assign home page webelement locator value to the object

using @FindBy annotation in "RediffHomePg" class

syntax:

@FindBy(Locator="Locator value")

WebElement obj;

Step 4:

create method/function to return obj from "RediffHomePg" class

syntax:

public WebElement methodname(){

return(obj);

}

Script in "RediffHomePg" class

-------------------------

public class RediffHomePg {

//Assign locator value

@FindBy(linkText="Sign in")

WebElement signin;

//to return obj

public WebElement signinObj(WebDriver driver){

return(signin);

}

}

Step 5: assign Login page webelement locator values to the objects

using @FindBy annotation in "RediffLoginPg" class and create

method/functions to return each obj

class

script in "RediffLoginPg" class

--------------------------------

public class RediffLoginPg {

//Assign locator value

@FindBy(name="login")

WebElement emailId;

//Assign locator value

@FindBy(name="passwd")

WebElement pwd;

//Assign locator value

@FindBy(name="proceed")

WebElement go;

//to return obj

public WebElement emailObj(WebDriver driver){

return(emailId);

}

public WebElement pwdObj(WebDriver driver){

return(pwd);

}

public WebElement goObj(WebDriver driver){

return(go);

}

}

Step 6: create class in "test.cases" package to validate login functionality

By initializing Webelement locator values from “object.repository” package classes using “PageFactory” class

Syntax:{to initialize objects from "object.repository" package classes}

ClassName Obj= PageFactory.initElements(driver, ClassName.class);

public class RediffLoginValidation {

public static void main(String[] args) throws InterruptedException {

WebDriver driver= new FirefoxDriver();

driver.get("http://rediffmail.com");

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

//to initialize objects from "object.repository" package classes

RediffHomePg RHP= PageFactory.initElements(driver, RediffHomePg.class);

RediffLoginPg RLP= PageFactory.initElements(driver, RediffLoginPg.class);

RHP.signinObj(driver).click();

RLP.emailObj(driver).sendKeys("Madhukar");

RLP.pwdObj(driver).sendKeys("Mercury");

RLP.goObj(driver).click();

Thread.sleep(3000);

String pgTitle= driver.getTitle();

if (pgTitle.contains("Inbox")){

System.out.println("Successful login operation");

}

else{

System.out.println("Unsuccessful login operation");

}

}

}

-----------------------------------------------------------------------------------------------------------------------------------------------------

\*\* I.Q.

PROPERTIES FILE

---------------

It is also called as configuration properties file.

Where we can maintain resuable test data in properties file like application path,user id,pwd,locator values,.... etc

Each parameter in properties file is stored as pair of strings, in key and value format, where each key is one line

Ex. Uid= livetech

To read Properties file we use "Properties class" object

Following are the methods provided by Properties class

(i) load(): to load the properties file in current test case

(ii) get Property(): to read key value from properties file

-- Properties file can be used on many packages

-- It uses "Properties class"

=======================================================================END OF CLASS===================================================================================