**Selenium**

Explain about your current framework with architecture

Perform Drag and Drop of Elements using Selenium Webdriver

package JqueryPackage;

import java.io.IOException;

import java.util.concurrent.TimeUnit;

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.interactions.Actions;

public class JqueryElements {

public static void main(String[] args) {

WebDriver driver = new FirefoxDriver();

driver.manage().timeouts().implicitlyWait(10, TimeUnit.SECONDS);

driver.get("http://jqueryui.com/droppable/#default");

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

Actions act = new Actions(driver);

*// Script for dragging an element and dropping it in another place*

WebElement iFrame = driver.findElement(By.tagName("iframe"));

System.out.println(iFrame.getSize());

driver.switchTo().frame(iFrame);

WebElement From = driver.findElement(By.id("draggable"));

System.out.println(From.getLocation());

WebElement To = driver.findElement(By.id("droppable"));

System.out.println(To.getLocation());

act.dragAndDrop(From, To).build().perform();

}

}

Mouse Hover Action using selenium WebDriver

import java.util.concurrent.TimeUnit;

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.interactions.Actions;

public class mouse\_hover {

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

WebDriver driver = new FirefoxDriver();

*// Wait For Page To Load*

driver.manage().timeouts().implicitlyWait(60, TimeUnit.SECONDS);

*// Go to URL*

driver.get("http://www.myntra.com/");

*// Maximize Window*

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

Actions builder = new Actions(driver);

WebElement mainmenu1 = driver.findElement(By.xxxxx());

builder.moveToElement(mainmenu1 ).build().perform();

Thread.sleep(500); //add a wait

WebElement submenu1= driver.findElement(By.xxxxx()); //Find the submenu

builder.moveToElement(submenu1).click().build().perform();

Thread.sleep(500);

}

}

Read and Write Excel

|  |
| --- |
| import java.io.File;  import java.io.FileInputStream;  import java.io.FileOutputStream;  import org.apache.poi.xssf.usermodel.XSSFSheet;  import org.apache.poi.xssf.usermodel.XSSFWorkbook;  import org.testng.annotations.Test;  public class ReadandWriteExcel {     public static void main(String []args){    try {    // Specify the file path which you want to create or write    File src=new File("./testdata/test.xlsx");    // Load the file    FileInputStream fis=new FileInputStream(src);     // load the workbook     XSSFWorkbook wb=new XSSFWorkbook(fis);    // get the sheet which you want to modify or create     XSSFSheet sh1= wb.getSheetAt(0);   // getRow specify which row we want to read and getCell which column   System.out.println(sh1.getRow(0).getCell(0).getStringCellValue());   System.out.println(sh1.getRow(0).getCell(1).getStringCellValue());   System.out.println(sh1.getRow(1).getCell(0).getStringCellValue());   System.out.println(sh1.getRow(1).getCell(1).getStringCellValue());   System.out.println(sh1.getRow(2).getCell(0).getStringCellValue());   System.out.println(sh1.getRow(2).getCell(1).getStringCellValue());  // here createCell will create column  // and setCellvalue will set the value   sh1.getRow(0).createCell(2).setCellValue("2.41.0");   sh1.getRow(1).createCell(2).setCellValue("2.5");   sh1.getRow(2).createCell(2).setCellValue("2.39");  // here we need to specify where you want to save file   FileOutputStream fout=new FileOutputStream(new File("location of file/filename.xlsx"));  // finally write content   wb.write(fout);  // close the file   fout.close();    } catch (Exception e) {     System.out.println(e.getMessage());    }   }  } |

Database Testing

package softwareTestingMaterial;

import java.sql.Connection;

import java.sql.DriverManager;

import java.sql.ResultSet;

import java.sql.SQLException;

import java.sql.Statement;

public class DBTesting {

     public static void selectQuery() throws SQLException, ClassNotFoundException {

String dbURL = "jdbc:db2://ipAddress:portNumber/dbName";

String username = myUserName;

        String password = myPassword;

        //Load DB2 JDBC Driver

        Class.forName("com.ibm.db2.jcc.DB2Driver");

        //Creating connection to the database

        Connection con = DriverManager.getConnection(dbURL,username,password);

        //Creating statement object

     Statement st = con.createStatement();

     String selectquery = "SELECT \* FROM <tablename> WHERE <condition>";

        //Executing the SQL Query and store the results in ResultSet

     ResultSet rs = st.executeQuery(selectquery);

     //While loop to iterate through all data and print results

     while (rs.next()) {

     System.out.println(rs.getString("transaction\_datetime"));

     }

        //Closing DB Connection

     con.close();

}

}

**Read Excel**

import java.io.FileInputStream;

import java.io.FileNotFoundException;

import java.io.FileOutputStream;

import java.io.IOException;

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

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

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;

//How to read excel files using Apache POI

public class ReadExcel {

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

                        //I have placed an excel file 'Test.xlsx' in my D Driver

FileInputStream fis = new FileInputStream("D:\\Test.xlsx");

XSSFWorkbook workbook = new XSSFWorkbook(fis);

XSSFSheet sheet = workbook.getSheetAt(0);

                        //I have added test data in the cell A1 as "SoftwareTestingMaterial.com"

                        //Cell A1 = row 0 and column 0. It reads first row as 0 and Column A as 0.

Row row = sheet.getRow(0);

Cell cell = row.getCell(0);

                       System.out.println(cell);

System.out.println(sheet.getRow(0).getCell(0));

//String cellval = cell.getStringCellValue();

//System.out.println(cellval);

}

}

**Write Excel**

import java.io.FileInputStream;

import java.io.FileNotFoundException;

import java.io.FileOutputStream;

import java.io.IOException;

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

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

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

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

public class WriteExcel {

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

//create an object of Workbook and pass the FileInputStream object into it to create a pipeline between the sheet and eclipse.

FileInputStream fis = new FileInputStream("D:\\Test.xlsx");

XSSFWorkbook workbook = new XSSFWorkbook(fis);

//call the getSheet() method of Workbook and pass the Sheet Name here.

//In this case I have given the sheet name as “TestData”

                //or if you use the method getSheetAt(), you can pass sheet number starting from 0. Index starts with 0.

XSSFSheet sheet = workbook.getSheet("TestData");

//XSSFSheet sheet = workbook.getSheetAt(0);

//Now create a row number and a cell where we want to enter a value.

//Here im about to write my test data in the cell B2. It reads Column B as 1 and Row 2 as 1. Column and Row values start from 0.

//The below line of code will search for row number 2 and column number 2 (i.e., B) and will create a space.

                //The createCell() method is present inside Row class.

                Row row = sheet.createRow(1);

Cell cell = row.createCell(1);

//Now we need to find out the type of the value we want to enter.

                //If it is a string, we need to set the cell type as string

                //if it is numeric, we need to set the cell type as number

cell.setCellType(cell.CELL\_TYPE\_STRING);

cell.setCellValue("SoftwareTestingMaterial.com");

FileOutputStream fos = new FileOutputStream("D:\\Test.xlsx");

workbook.write(fos);

fos.close();

System.out.println("END OF WRITING DATA IN EXCEL");

}

}

**Highlight the element & javascript executor**

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.testng.annotations.Test;

public class Highlight {

@Test

public void highlightElement() {

System.setProperty("webdriver.gecko.driver", "D:\\Selenium Environment\\Drivers\\geckodriver.exe");

WebDriver driver = new FirefoxDriver();

driver.get("https://www.gmail.com");

WebElement ele = driver.findElement(By.xpath("//\*[@id='Email']"));

JavascriptExecutor js = (JavascriptExecutor) driver;

js.executeScript("arguments[0].setAttribute('style', 'background: yellow; border: 2px solid red;');", ele);

}

}

**What is XPath?**

XPath is defined as **XML path**. **It is a syntax or language for finding any element on the web page using XML path expression**. XPath is used to find the location of any element on a webpage using HTML DOM structure. The basic format of XPath is explained below with screen shot.

Syntax for XPath:

XPath contains the path of the element situated at the web page. Standard syntax for creating XPath is.

Xpath=//tagname[@attribute='value']

// : Select current node.

Tagname: Tagname of the particular node.

@: Select attribute.

Attribute: Attribute name of the node.

Value: Value of the attribute.

Absolute XPath:

It is the direct way to find the element, but the disadvantage of the absolute XPath is that if there are any changes made in the path of the element then that XPath gets failed.

The key characteristic of XPath is that it begins with the single forward slash(/) ,which means you can select the element from the root node.

html/body/div[1]/section/div[1]/div/div/div/div[1]/div/div/div/div/div[3]/div[1]/div/h4[1]/b

Relative xpath:

For Relative Xpath the path starts from the middle of the HTML DOM structure. It starts with the double forward slash (//), which means it can search the element anywhere at the webpage.

You can start from the middle of the HTML DOM structure and no need to write long xpath.

Below is the example of a relative XPath expression of the same element shown in the below screen. This is the common format used to find element through a relative XPath.

Relative xpath: //\*[@class='featured-box']//\*[text()='Testing']

1) Basic XPath: Xpath=//input[@name='uid']

2) Contains(): Xpath=//\*[contains(@type,'sub')]

3) Using OR & AND: Xpath=//\*[@type='submit' OR @name='btnReset']

Xpath=//input[@type='submit' and @name='btnLogin']

4) Start-with function: Xpath=//label[starts-with(@id,'message')]

5) Text(): Xpath=//td[text()='UserID']

6) XPath axes methods:

a) Following: Xpath=//\*[@type='text']//following::input

Xpath=//\*[@type='text']//following::input[1]

b) Ancestor: The ancestor axis selects all ancestors element (grandparent, parent, etc.) of the current node

Xpath=//\*[text()='Enterprise Testing']//ancestor::div[1]

c) Child: Selects all children elements of the current node (Java)

Xpath=//\*[@id='java\_technologies']/child::li

Xpath=//\*[@id='java\_technologies']/child::li[1]

d) Preceding:

Select all nodes that come before the current node

Xpath=//\*[@type='submit']//preceding::input

Xpath=//\*[@type='submit']//preceding::input[1]

e) Following-sibling:

Select the following siblings of the context node. Siblings are at the same level of the current node as shown in the below screen. It will find the element after the current node.

xpath=//\*[@type='submit']//following-sibling::input

f) Parent:

Selects the parent of the current node as shown in the below screen.

Xpath=//\*[@id='rt-feature']//parent::div

g) Self:

Selects the current node or 'self' means it indicates the node itself as shown in the below screen.

One node matching by using "self " axis. It always finds only one node as it represents self-element.

Xpath =//\*[@type='password']//self::input

h) Descendant:

Selects the descendants of the current node as shown in the below screen.

In the below expression, it identifies all the element descendants to current element ( 'Main body surround' frame element) which means down under the node (child node , grandchild node, etc.).

Xpath=//\*[@id='rt-feature']//descendant::a

Xpath=//\*[@id='rt-feature']//descendant::a[1]

Summary:

XPath is required to find an element on the web page as to do an operation on that particular element.

There are two types of XPath:

Absolute XPath

Relative XPath

XPath Axes are the methods used to find dynamic elements, which otherwise not possible to find by normal XPath method

XPath expression select nodes or list of nodes on the basis of attributes like ID , Name, Classname, etc. from the XML document .

CSS/XPath Related Selenium Webdriver Questions.

Q: How would you differentiate between the absolute and relative XPath?

Ans.

Single slash "/" signifies an absolute XPath.

Double Slash "//" represents the relative XPath.

Q: What are the different ways to find the element by CSS class?

Ans. Say, the name of the CSS class is <*MyTest*>. Then, the most simple way to find the element is as follows.

|  |  |
| --- | --- |
|  | //\*[contains(@class, ‘MyTest')] |

If you want an exact comparison to avoid matching texts like “OhhMyTest” or “MyTesting” then use this.

|  |  |
| --- | --- |
|  | //\*[contains(concat(' ', @class, ' '), ' MyTest ')] |

You can even use the <normalize-space> function to remove any leading or trailing whitespaces.

|  |  |
| --- | --- |
|  | //\*[contains(concat(' ', normalize-space(@class), ' '), 'MyTest ')] |

Q: How to specify a child element using XPath?

Ans.

|  |  |
| --- | --- |
|  | //div/a |

Q: How to specify a child element using CSS?

Ans.

|  |  |
| --- | --- |
|  | css=div > a |

Q: How to identify a sub child element using XPath?

Ans.

|  |  |
| --- | --- |
|  | //div//a |

Q: How to identify a sub child element using CSS?

Ans.

|  |  |
| --- | --- |
|  | css=div a |

Q: How do you access an element ID using CSS?

Ans.

|  |  |
| --- | --- |
|  | css=div#sample a |

Q: How would you get to the next sibling?

Ans. You may try this code.

|  |  |
| --- | --- |
|  | tr/td[@class='user’]/following-sibling::td |

Or you can use the below code.

|  |  |
| --- | --- |
|  | tr[td[@class=’user'] ='age']/td[@class='weight'] |

[Detecting element color](https://sqa.stackexchange.com/questions/12698/selenium-webdriver-detecting-element-color)

First of all, we have to get a value of link color using getCssValue method. It can be done by using below code. In the code, Products link’s CSS attribute ‘color’ is stored in a String variable called ‘color’.

|  |  |
| --- | --- |
| 1 | String color = driver.findElement(By.xpath("//a[@href='products/']")).getCssValue("color"); |

The above code will return value in RGB format such as “rgba(36, 93, 193, 1)”. We will convert it into more convenient Hex code using Java. Use below code for it.

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9 | String[] hexValue = color.replace("rgba(", "").replace(")", "").split(",");    int hexValue1=Integer.parseInt(hexValue[0]);  hexValue[1] = hexValue[1].trim();  int hexValue2=Integer.parseInt(hexValue[1]);  hexValue[2] = hexValue[2].trim();  int hexValue3=Integer.parseInt(hexValue[2]);    String actualColor = String.format("#%02x%02x%02x", hexValue1, hexValue2, hexValue3); |

That’s it. You will get a value of color in Hex code after above code is executed. We can add an Assert statement to verify that the color is matching with the expected color.

|  |  |
| --- | --- |
| 1 | Assert.assertEquals("#245dc1", actualColor);  OR  you can user "getcssvalue" in java to get css attributes of element. Below is example  String headerColor = driver.findElement(By.xpath(".//\*[@id='LoginName']/h1")).getCssValue("backgroud-color");  Assert.assertEquals("some message", "#FFFFFF", headerColor); |

**Introduction to AutoIT tool**

1-AutoIt is freeware automation tool that can work with desktop application too.

2-It uses a combination of keystrokes, mouse movement and window/control manipulation in order to automate tasks in a way not possible or reliable with other languages (e.g. VBScript and SendKeys).

**How to write a script in AutoIT?**

For AutoIt scripting, you should have three things ready.

1-AutoIt Editor- Editor helps us to write AutoIt scripts.

2-Tool Finder (Same as firebug on Firefox) – It will help us to identify the element and check their Attributes.

3- AutoIt Help section- This help you to understand about AutoIt functions and what are the parameter it accepts.

Let’s start with Downloading first

**Step 1**– Navigate to AutoIt  official website  <https://www.autoitscript.com/site/autoit/downloads/>  and go to download section or Click here [Download AutoIt](https://www.autoitscript.com/site/autoit/downloads/)

**Step 2**– Click on Download AutoIt and Install

**Step 3**–  Click on Download Editor and Install.

Step 4– Once both installed in your machine check all is installed correctly.

Note- Generally it goes to C:\Program Files\AutoIt3 location if you do not change it

**Step5**– Open SCiTE folder and  Click on SciTE this will open AutoIt Editor

Once all Installed Let’s see how we can write script

Upload File in Selenium Webdriver using Autoit

To Upload File in Selenium Webdriver using Autoit we need to take care of some steps so let’s begin

To upload a file in Selenium Webdriver we will create AutoIT script, which will handle file-uploaded window, and then we will combine Selenium script with AutoIt scripts.

Click on Upload button you will get file uploader we will handle the same using AutoIt.

**Step 1- Open Editor and Finder Tool**

 Step 2– We need to write script to upload file so we will use some method of AutoIt.  
Each method will have some own functionality  
ControlFocus-This will give focus on the window

ControlSetText-This will set the file path

ControlClick-This will click on button

Step 1-

Click on Browse button , a new window will open now open finder tool and Click on Finder tool and drag to the file name as I shown in below screenshot.

This will give all the detail about that window and file name section info; we will use only some attribute like window title, class, and instance.

Open AutoIt Editor and Write Script

In ControlClick method we will give control id of open button

Step 2-

Save the script to a particular location with some unique name.

Note- By default script will be saved as .au3 extension

Step 3– Now Compile the script so for compiling right click on file and Select compile script this will generate a .exe file of the file.

Step 4- Now write Selenium program and add this .exe file and run your program

Here is the code

mport org.openqa.selenium.By;

import org.openqa.selenium.WebDriver;

import org.openqa.selenium.firefox.FirefoxDriver;

public class DemoFileUpload {

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

// This will open Firefox browser

WebDriver driver=new FirefoxDriver();

// This will maximize browser to full screen

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

// This will open respective URL

driver.get("your application url");

// This will click on Upload button

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

// This will invoke AutoIT script here give the path of the script

//and this will throw IO exception so u can use throw or try catch

// In my case I am using throws

Runtime.getRuntime().exec("C:\\Users\\mukesh\_otwani\\Desktop\\AutoItScripts\\blogUpload.exe");

// Once you will run this program AutoIt script will be invoked and respective f//ile will be attached

  }

}

**How to Download files in Selenium Webdriver**

Hello Welcome to Selenium tutorial, today we will see How to Download files using Selenium Webdriver.

In previous post, we have seen how to[upload files using robot class](http://learn-automation.com/upload-file-in-selenium-webdriver-using-robot-class/) and[upload files using AutoIT](http://learn-automation.com/upload-file-in-selenium-webdriver-using-autoit/). Today we will see some different scenario we will see downloading files in Selenium.

Sometime in your application, you have to control some scenario in which you need to download some files by clicking on some link or some button.

Once you start download files/application, you will get one confirmation window, which will ask to save, file or cancel here Webdriver stuck because this is Window’s Pop up. Selenium can handle only Web browser automation not windows based application. Refer below screenshot for more information.

Here download files window is displayed if you try to inspect using firebug you will not get any locators for them.

So using some browser settings (firefoxprofile) we can skip that confirmation window and we can continue with our script.

Here is some setting that we need to modify let’s see how to check these setting

How to Download files in Selenium Webdriver

1- Open [Firefox](http://learn-automation.com/selenium-script-firefox/) browser and in url box type about:config and press enter

2- In Search bar type neverask and enter, here you will find some settings(refer below screenshot)

Now you can see here value is blank so we need to mention which type of file it will not ask if download starts in case.

Note- In this post I am giving values for .exe file(application), in your case if you want to download pdf, excel file etc. you need to mention values (MIME type).

Here you can find their MIME type that will be the values for these settings.

<http://www.sitepoint.com/web-foundations/mime-types-complete-list/>

How to Download files in Selenium Webdriver

Step 1- Create a firefox Profile.

Step 2- set Preferences as per requirement.

Step 3- Open Firefox with firefox profile.

Let us implement the same through Script.

Note- This script will download [Adobe](http://www.adobe.com/) Reader from [Filehippo.com](http://filehippo.com/)

package blog;

import java.util.concurrent.TimeUnit;

import org.openqa.selenium.By;

import org.openqa.selenium.WebDriver;

import org.openqa.selenium.firefox.FirefoxDriver;

import org.openqa.selenium.firefox.FirefoxProfile;

public class DownloadFiles {

public static void main(String[] args) {

// Create a profile

FirefoxProfile profile=new FirefoxProfile();

// Set preferences for file type

profile.setPreference("browser.helperApps.neverAsk.openFile", "application/octet-stream");

// Open browser with profile

WebDriver driver=new FirefoxDriver(profile);

// Set implicit wait

driver.manage().timeouts().implicitlyWait(30, TimeUnit.SECONDS);

// Maximize window

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

// Open APP to download application

driver.get("http://www.filehippo.com/download\_adobe\_reader");

// Click on download

driver.findElement(By.xpath(".//\*[@id='program-header']/div[4]/a[1]")).click();

 }

}

**Double click in selenium**

Actions action = new Actions(driver);

WebElement ele = driver.findElement(By.cssSelector("html>body>div"));

((JavascriptExecutor) driver).executeScript("arguments[0].scrollIntoView();", ele); action.doubleClick(ele).build().perform();

How many types of Webdriver APIs are available in Selenium?

Ans. Below is the list of driver classes that you can use for the browser automation.

AndroidDriver,

ChromeDriver,

EventFiringWebDriver,

FirefoxDriver,

HtmlUnitDriver,

InternetExplorerDriver,

IPhoneDriver,

IPhoneSimulatorDriver,

RemoteWebDriver.

**How would you make sure that a page is loaded using Selenium and Webdriver?**

In Selenium, you can use the below lines of code to check for the successful loading of a web page. The best approach is by selecting an element from the page & stand by till it becomes clickable.

|  |  |
| --- | --- |
| 1  2  3  4  5  6 | selenium.waitForPageToLoad("5000");  // Or  while (!(selenium.isElementPresent("any page element ")==true)) {    selenium.setSpeed("5");    Thread.sleep(5);  } |

Below is the Webdriver specific code to achieve the same objective.

|  |  |
| --- | --- |
| 1  2 | WebDriverWait check = new WebDriverWait(driver, 100);  check.until(ExpectedConditions.anyElement(By.id(id))); |

**How to launch a batch file in a Selenium Webdriver project?**

Ans. It's usual in a test suite to run a batch file or an executable file for setting up the pre-requisites before starting the automation. You can use the below Java code for this purpose.

|  |  |
| --- | --- |
| 1  2 | Process batch = Runtime.getRuntime.exec("path of the batch file");  batch.waitFor(); |

**How do you read a JavaScript variable in Selenium WebDriver?**

Ans. It's easy to access any JavaScript variable from the Selenium Webdriver test scripts. Just you need to use the below Java code snippet.

|  |  |
| --- | --- |
| 1  2  3  4  5 | // Initialize the JS object.  JavascriptExecutor JS = (JavascriptExecutor) webdriver;  // Get the current site title.  String sitetitle = (String)JS.executeScript("return document.title");  System.out.println("My Site Title: " + sitetitle); |

**How to run the selenium IDE test suite from the command line?**

We've given the command to run the SIDE test suite in the next line.

|  |  |
| --- | --- |
| 1 | Java -jar "C:\Selenium Webdriver Questions\selenium-server-standalone-2.33.0.jar" -htmlSuite "\*firefox" "http://www.google.com" " “C:\Selenium Webdriver Questions\SeleniumSuite.HTML" |

**How to run the Selenium Webdriver test from the command line?**

We can run any Selenium Webdriver test written in Java using the following command.

|  |  |
| --- | --- |
| 1 | java -classpath ".;selenium-server-standalone-2.33.0.jar" SampleClass |

What are the different exceptions you face in Selenium Webdriver?

WebDriverException,

NoAlertPresentException,

NoSuchWindowException,

NoSuchElementException,

TimeoutException.

How would you automatically click a screenshot whenever any exception occurs?

For this you will have to use <*EventFiringWebDriver*> class and needs to implement the <*onException*> method of the <*WebDriverEventListener*> interface. See the code example given below.

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24 | WebDriver browser = new FirefoxDriver();    EventFiringWebDriver eventDriver = new EventFiringWebDriver(browser).register(new AbstractWebDriverEventListener() {      @Override    public void onException(Throwable throwable, WebDriver browser) {        // Take the screenshot using the Webdriver.      File screen = ((TakesScreenshot)driver).getScreenshotAs(OutputType.FILE);        // Now you can copy the screenshot somewhere on your system.      FileUtils.copyFile(screen, new File("c:\\Selenium Testing Questions\\screen.png"));    }  });    try {      eventDriver.findElement(By.id("test"));    fail("Caught the Expected exception."); // Intentionally causing the exception for demo.    } catch (NoSuchElementException e) {      // Triggering point for the <onException> event.  } |

**How would you select any particular text using the Selenium Webdriver?**

It seems an easy one at first but you need to do a little more to achieve this.

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7 | driver.get("/");    WebElement item = driver.findElement(By.xpath("//p[contains(text(),'Selenium webdriver quesions')]"));    Actions dummy = new Actions(driver);    dummy.doubleClick(item).build().perform(); |

**Give an example to perform drag and drop action In Selenium WebDriver?**

Yes, we can use the Advanced User Interactions API to perform drag and drop operations in a Selenium Webdriver project.

Code example:

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7 | Actions act = new Actions(driver);    act.dragAndDrop(source\_locator, target\_locator).build().perform();    //Or you can use the below code style.    (new Actions(driver)).dragAndDrop(source\_locator, target\_locator).perform(); |

**How would you fill a text field without calling the sendKeys()?**

It's a bit slower than the sendKeys() method but we do have means to type in a text field. See the Java code given below.

|  |  |
| --- | --- |
| 1  2  3  4  5 | JavascriptExecutor JS = (JavascriptExecutor)webdriver;    JS.executeScript("document.getElementById(User').value='admin@testmail.com'");    JS.executeScript("document.getElementById('Pass').value='######'"); |

**How can you check the state of a checkbox/radio button?**

We can call the isSelected() method to test the status of these elements.

Example Code:

|  |  |
| --- | --- |
| 1 | boolean test = driver.findElement(By.xpath("checkbox/radio button XPath")).isSelected(); |

How would you handle the alert popups in Selenium Webdriver?

First, you’ve to switch the control to the pop up then press the ok or cancel button. After that, turn back to the source page screen.

Code Example:

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11 | String srcPage = driver.getWindowHandle();    Alert pop = driver.switchTo().alert(); // shift control to the alert popup.    pop.accept(); // click on ok button.    pop.dismiss(); // click on cancel button.    // Move the control back to source page.    driver.switchTo().window(srcPage);  // move back to the source page. |

**What is the process to start the IE/Chrome browser?**

If you want to start a browser then, just set the system properties as mentioned below.

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11 | // For the IE web browser.    System.setProperty("webdriver.ie.driver"," iedriver.exe file path");    WebDriver driver = new InternetExplorerDriver();    // For the Chrome web browser.    System.setProperty("webdriver.chrome.driver","chrome.exe file path");    WebDriver driver = new ChromeDriver(); |

**How would you simulate the right click operation in WebDriver?**

You can make use of the Actions class features.

|  |  |
| --- | --- |
| 1  2  3  4  5 | Actions test = new Actions(driver); // Here, driver is the object of WebDriver class.    test.moveToElement(element).perform();    test.contextClick().perform(); |

**How would you select a menu item from a drop down menu?**

There can be following two situations.

If the menu has is using the <*select*> tag then you can call the <*selectByValue()>* or <selectByIndex()> or <selectByVisibleText()> methods of the Select class.

If the menu doesn’t use the <*select*> tag then simply find the XPath of that element and perform the click action for its selection.

**What is the FirefoxDriver, class or an interface? And which interface does it implement?**

*FirefoxDriver* is a Java class, and it implements the <*WebDriver*> interface. It contains the implementations of all the methods available in the <*WebDriver*> interface.

**What is the name of the super interface of the Webdriver?**

SearchContext.

**What is the main difference between the close() and quit() methods?**

close() - it closes the currently active browser window.

quit()- it will close all of the opened browser windows and the browser itself.

**What is the best way to check for the highlighted text on a web page?**

Use the below code to verify the highlighted text for an element on the web page.

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7 | String clr = driver.findElement(By.xpath("//a[text()='TechBeamers']")).getCssValue("color");    String bkclr = driver.findElement(By.xpath("//a[text()='TechBeamers']")).getCssValue("background-color");    System.out.println(clr);    System.out.println(bkclr); |

**How would you use a Selenium variable say "size" from the JavaScript?**

${size}

**What is the Selenese command to show the value of a variable in the log file?**

echo()

**Implicit Wait:**

The implicit wait tells to the WebDriver to wait for certain amount of time before it throws an exception. Once we set the time, WebDriver will wait for the element based on the time we set before it throws an exception. The default setting is 0 (zero). We need to set some wait time to make WebDriver to wait for the required time.

driver.manage().timeouts().implicitlyWait(TimeOut, TimeUnit.SECONDS);

**Explicit Wait:**

Explicit waits are confined to a particular web element. Explicit Wait is code you define to wait for a certain condition to occur before proceeding further in the code.

Explicit wait is of two types:

WebDriverWait

FluentWait

**WebDriverWait:**

WebDriverWait is applied on certain element with defined expected condition and time. This wait is only applied to the specified element. This wait can also throw exception when element is not found.

The following are the Expected Conditions that can be used in Explicit Wait

alertIsPresent()

elementSelectionStateToBe()

elementToBeClickable()

elementToBeSelected()

frameToBeAvaliableAndSwitchToIt()

invisibilityOfTheElementLocated()

invisibilityOfElementWithText()

presenceOfAllElementsLocatedBy()

presenceOfElementLocated()

textToBePresentInElement()

textToBePresentInElementLocated()

textToBePresentInElementValue()

titleIs()

titleContains()

visibilityOf()

visibilityOfAllElements()

visibilityOfAllElementsLocatedBy()

visibilityOfElementLocated()

//WebDriverWait wait = new WebDriverWait(WebDriverRefrence,TimeOut);

WebDriverWait wait = new WebDriverWait (driver, 20);

wait.until(ExpectedConditions.VisibilityofElementLocated(By.xpath(""//button[@value='Save Changes']"")));

**FluentWait:**

FluentWait can define the maximum amount of time to wait for a specific condition and frequency with which to check the condition before throwing an “ElementNotVisibleException” exception.

To say in effortless manner, it tries to find the web element repeatedly at regular intervals of time until the timeout or till the object gets found.

We use Fluent Wait commands mainly when we have web elements which sometimes visible in few seconds and some times take more time than usual to visible. Mainly in Ajax applications.

**Syntax:**

|  |  |
| --- | --- |
| 3  4  5  6  7  8  9  10 | Wait wait = new FluentWait(WebDriver reference)  .withTimeout(timeout, SECONDS)  .pollingEvery(timeout, SECONDS)  .ignoring(Exception.class);    WebElement foo=wait.until(new Function<WebDriver, WebElement>() {  public WebElement applyy(WebDriver driver) {  return driver.findElement(By.id("foo"));  }  }); |

**TestNG Selenium Webdriver Questions.**

**TestNG Annotations:**

**@Test:**Marks a class or a method as a part of the test.

**@BeforeMethod:** A method which is marked with this annotation will be executed before every **@test** annotated method.

**@AfterMethod:** A method which is marked with this annotation will be executed after every **@test** annotated method.

**@BeforeClass:** A method which is marked with this annotation will be executed before ***first @Test*** method execution. It runs only once per class.

**@AfterClass:**A method which is marked with this annotation will be executed after all the test methods in the current class have been run

**@BeforeTest:** A method which is marked with this annotation will be executed before ***first @Test*** annotated method.

**@AfterTest:**A method which is marked with this annotation will be executed when **all @Test** annotated methods complete the execution of those classes which are inside <test> tag in testng.xml file.

**@BeforeSuite:**A method which is marked with this annotation will run **only once before** all tests in the suite have run

**@AfterSuite:**A method which is marked with this annotation will run **once after** execution of all tests in the suite have run

**@BeforeGroups:**This annotated method will run **before the first test run** of that specific group.

**@AfterGroups:**This annotated method will run **after all test methods** of that group completes its execution.

Some other TestNG Annotations, we need to discuss here are mentioned below:

**@Parameters:**This annotation is used to pass parameters to test methods.

**@DataProvider:**If we use @DataProvider annotation for any method that means you are using that method as a data supplier. The configuration of @DataProvider annotated method must be like it always return Object[][] which we can use in @Test annotated method. The @Test method that wants to receive data from this DataProvider needs to use a dataProvider name equals to the name of this annotation.

**@Factory:**Marks a method as a factory that returns objects that will be used by TestNG as Test classes. The method must return Object[ ].

**@Listeners:**This annotation is used with test class. It helps in writing logs and results.

**Test case execution Parallel**

How to run test cases in parallel using TestNG?

we can use “parallel” attribute in testng.xml to accomplish parallel test execution in TestNG

The parallel attribute of suite tag can accept four values:

tests – All the test cases inside <test> tag of testng.xml file will run parallel  
classes – All the test cases inside a java class will run parallel  
methods – All the methods with @Test annotation will execute parallel  
instances – Test cases in same instance will execute parallel but two methods of two different instances will run in different thread.

Exp 1

<suite name="softwaretestingmaterial" parallel="methods">

Exp 2

<?xml version="1.0" encoding="UTF-8"?>

<!DOCTYPE suite SYSTEM "http://testng.org/testng-1.0.dtd">

<suite name="TestSuite" thread-count="3" parallel="methods" >

<test name="testGuru">

<classes>

<class name="TestGuru99MultipleSession">

</class>

</classes>

</test>

</suite>

**How to exclude a particular test group from a test case execution?**

By adding the exclude tag in the testng.xml

<groups>

    <run>

<exclude name="TestGroupNameToExclude"/>

    </run>

</groups>

**How to disable a test case in TestNG ?**

To disable the test case we use the parameter enabled = false to the @Test annotation.

@Test(enabled = false)

**How to skip a @Test method from execution in TestNG?**

By using throw new SkipException()

Once SkipException() thrown, remaining part of that test method will not be executed and control will goes directly to next test method execution.

throw new SkipException("Skipping - This is not ready for testing ");

**How to Ignore a test case in TestNG?**

To ignore the test case we use the parameter enabled = false to the @Test annotation.

@Test(enabled = false)

How to write regular expression In testng.xml file to search @Test methods containing “smoke” keyword.

Regular expression to find @Test methods containing keyword “smoke” is as mentioned below.

<methods>

     <include name=".\*smoke.\*"/>

</methods>

[TestNG: How can I run same test case multiple times?](https://stackoverflow.com/questions/26128289/testng-how-can-i-run-same-test-case-multiple-times)

@Test(invocationCount = 10)

public void testCount() {..}

**What is the use of @Test(threadPoolSize=x)?**

The threadPoolSize attribute tells to form a thread pool to run the test method through multiple threads.

**Note:** This attribute is ignored if invocationCount is not specified

@Test(threadPoolSize = 3, <code class="plain">invocationCount = </code><code class="value">10</code>) public void testCase1(){

What are @Factory and @DataProvider annotation?

@Factory: A factory will execute all the test methods present inside a test class using a separate instance of the respective class with different set of data.

@DataProvider: A test method that uses DataProvider will be executed the specific methods multiple number of times based on the data provided by the DataProvider. The test method will be executed using the same instance of the test class to which the test method belongs.

TestNG Listeners

TestNG Listeners are used to inspect and modify the testing behavior. TestNG listeners always extend org.testng.ITestNGListener marker interface. TestNG listeners can be defined for a test class using org.testng.annotations.Listeners annotation.

Let’s look at some of the important TestNG listeners.

ISuiteListener: We can use this test suite listener to perform some operations when test suite starts and when all the tests are executed. This interface contains two methods – onStart(ISuite suite)and onFinish(ISuite suite) and provides access to test suite object.

ITestListener: We can use this listener to analyze test methods, perform logging. We can also use them to send notifications if any test fails by implementing onTestFailure(ITestResult result)method.

IAnnotationTransformer: We can implement this interface to modify the annotations for any @Testmethod. Note that we can use this annotation only with [TestNG XML](https://www.journaldev.com/21304/testng-xml) configuration.

IAnnotationTransformer2: We can implement this interface to modify the annotations for any method other than @Test method.This annotation can be used with TestNG XML configuration only.

IConfigurable: If a test class implements this interface, its run() method will be invoked instead of each configuration method found.

IConfigurationListener: Listener interface for events related to configuration methods.

IExecutionListener: This listener is used to monitor when a TestNG run starts and ends.

IHookable: If a test class implements this interface, its run() method will be invoked instead of each @Test method found.

IInvokedMethodListener: A listener that gets invoked before and after a method is invoked by TestNG.

IMethodInterceptor: This class is used to alter the list of test methods that TestNG is about to run.

IReporter: This interface can be implemented by clients to generate a report.

Execute failed test cases using TestNG in Selenium – By using “testng-failed.xml”

Steps To follow:

After the first run of an automated test run. Right click on Project – Click on Refresh

A folder will be generated named “test-output” folder. Inside “test-output” folder, you could find “testng-failed.xml”

Run “testng-failed.xml” to execute the failed test cases again.

**TestNG Groups:**

|  |  |
| --- | --- |
|  | package softwareTestingMaterial;  import org.testng.annotations.Test;  public class TestCase1 {  *@Test* (groups = { "smokeTest", "functionalTest" })  public void loginTest(){  System.out.println("Logged in successfully");  }  }  **testng.xml:**  <?xml version="1.0" encoding="UTF-8"?>  <!DOCTYPE suite SYSTEM "http://testng.org/testng-1.0.dtd" >    <suite name="softwaretestingmaterial">  <test name="testngTest">    <groups>  <run>  <include name="smokeTest" />  </run>  </groups>    <classes>  <class name="softwareTestingMaterial.TestCase1" />  </classes>  </test>  </suite>  **What is TestNG Assert and list out common TestNG Assertions?**  TestNG Asserts help us to verify the condition of the test in the middle of the test run. Based on the TestNG Assertions, we will consider a successful test only if it is completed the test run without throwing any exception.  Some of the common assertions supported by TestNG are  assertEqual(String actual,String expected)  assertEqual(String actual,String expected, String message)  assertEquals(boolean actual,boolean expected)  assertTrue(condition)  assertTrue(condition, message)  assertFalse(condition)  assertFalse(condition, message)  **What is Soft Assert in TestNG?**  Soft Assert collects errors during *@Test*. Soft Assert does not throw an exception when an assert fails and would continue with the next step after the assert statement.  If there is any exception and you want to throw it then you need to use assertAll() method as a last statement in the @Test and test suite again continue with next @Test as it is.  **What is Hard Assert in TestNG?**  Hard Assert throws an AssertException immediately when an assert statement fails and test suite continues with next *@Test*  **How to create Group of Groups in TestNG?**  Groups can also include other groups. These groups are called *MetaGroups*. For example, you might want to define a group *all* that includes *smokeTest*and functionalTest. Let’s modify our testng.xml file as follows:  <groups>     <define name="all">  <include name="smokeTest"/>  <include name="functionalTest"/>     </define>     <run>           <include name="all" />     </run>  </groups> |

**Q: What is the time-unit you use in testng time tests?**

Ans. We use *milliseconds* as the time unit in *@Test* or *@TestSuite* methods.

**Q: What is the use of priority in @Test methods and how do you set it?**

Ans. We define the priority in @Test annotated methods to set the order of test execution. Please check the below example.

|  |  |
| --- | --- |
| 1  2  3 | @Test1(priority=0)  @Test2(priority=2)  @Test3(priority=1) |

The above test will run in the following order.

|  |  |
| --- | --- |
| 1 | Test1 => Test3 => Test2 |

**Q: What are the two different ways of generating reports in TestNG?**

Ans. There are two ways to produce a report with Test NG, they are

Use of Listeners: You can implement the <*org.testng.ITestListener*> interface and its methods. Reporting will start when the test begins, finishes, skips, passes or fails.

Use of Reporters: Another way to enable reporting is by implementing the <*org.testng.Reporter*> interface. Unlike the listener interface, here the reporting begins after the whole test suite reaches to the end. The reporting class receives the object which contains the information of the whole test run.

**Q: How to execute the testng test suite from the command line?**

Ans. We don't always need to open Eclipse to run the TestNG project, try using the below command instead.

|  |  |
| --- | --- |
| 1 | java -cp “C:\Selenium Webdriver Questions\testng \lib\\*;C:\Selenium Testing Questions\testng\bin” org.testng.TestNG testng.xml |

**Q: How to turn off the test-output (generated test output folder) in TestNG?**

Ans. In Eclipse, modify the launch/Run configuration and add a <*-usedefaultlisteners false*> option to disable the test output.

**What is Continuous Integration?**

Continuous Integration is abbreviated as CI. Continuous Integration is a development practice which aims to make sure the correctness of a software. After each commit, a suite of tests run automatically and tests a software to ensure whether the software is running without any breaks. If any test fails, we will get immediate feedback say “build is broken”.

In simple words, continuous integration is a process of verifying the correctness of a software.

Some of the continuous integration tools are Jenkins, TeamCity, Bamboo, Travis, CircleCi, Bitbucket, CruiseControl

**What is Maven?**

Maven, a Yiddish word meaning “accumulator of knowledge”. Maven is a build automation tool used primarily for Java projects. It helps in building software, it describes how software is built and it describes its dependencies. It dynamically downloads Java libraries and Maven plug-ins from one or more repositories such as the Maven 2 Central Repository, and stores them in a local cache. We use maven in Selenium as a build tool or project management tool. It helps in managing all project dependencies and ensure an easy build process.

Main Objectives of Maven are as follows:

Making the build process easy

Providing a uniform build system

Providing quality project information

Providing guidelines for best practices development

Allowing transparent migration to new features

What is GIT?

Git is the most widely used modern version control system in the world today for tracking changes in any set of files. Git is an open source project. It is aimed at speed, data integrity, and support for distributed, non-linear workflows. It allows you to commit your work locally and then sync your copy of the repository with the copy on the server.

What is Jenkins?

Jenkins is an open source automation server. It supports us to automate all sorts of tasks related to building, deploying and automating any project. It is cross-platform and can be used on Windows, Mac OS, Linux etc., As a part of Selenium, we use it to build and test our software continuously. It is used for automatic test executions and scheduled builds. Using Jenkins we could also publish results and send email notifications to all the team members. Jenkins provides continuous integration and continuous delivery service for software development.

**Java**

**Inheritance in Java**

The process of obtaining the data members and methods from one class to another class is known as inheritance. It is one of the fundamental features of object-oriented programming.

Important points

In the inheritance the class which is give data members and methods is known as base or super or parent class.

The class which is taking the data members and methods is known as sub or derived or child class.

The data members and methods of a class are known as features.

The concept of inheritance is also known as re-usability or extendable classes or sub classing or derivation.

Interface : class Demo implements X, Y

[Types of Inheritance](https://beginnersbook.com/2013/05/java-inheritance-types/):  
Single Inheritance: refers to a child and parent class relationship where a class extends the another class.

Multilevel inheritance: refers to a child and parent class relationship where a class extends the child class. For example class A extends class B and class B extends class C.

Hierarchical inheritance: refers to a child and parent class relationship where more than one classes extends the same class. For example, class B extends class A and class C extends class A.

Multiple Inheritance: refers to the concept of one class extending more than one classes, which means a child class has two parent classes. Java doesn’t support multiple inheritance, read more about it

**Abstraction in Java**

**Abstraction** is the concept of exposing only the required essential characteristics and behavior with respect to a context.

Hiding of data is known as **data abstraction**. In object oriented programming language this is implemented automatically while writing the code in the form of class and object.

Real Life Example of Abstraction in Java

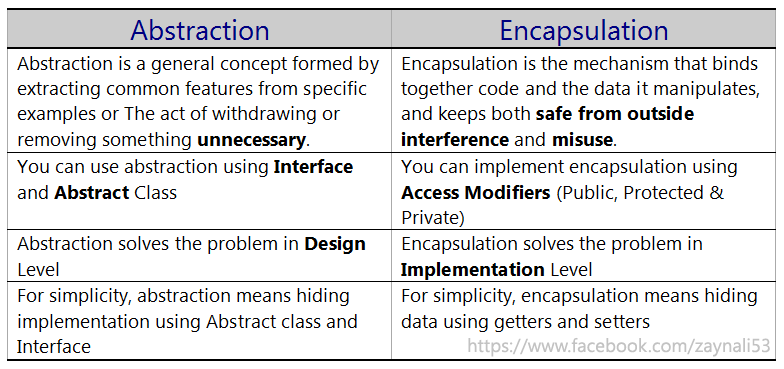
Abstraction shows only important things to the user and hides the internal details, for example, when we ride a bike, we only know about how to ride bikes but can not know about how it work? And also we do not know the internal functionality of a bike.

How to achieve Abstraction ?

There are two ways to achieve abstraction in java

Abstract class (0 to 100%)

Interface (Achieve 100% abstraction)

[](https://www.google.co.in/url?sa=i&rct=j&q=&esrc=s&source=imgres&cd=&ved=0ahUKEwiTlLXkmeDVAhUJChoKHfwvA70QjRwIBw&url=https://stackoverflow.com/questions/742341/difference-between-abstraction-and-encapsulation&psig=AFQjCNH1Q4xI55EdGvmuyffePXWa2JbXdg&ust=1503125716193611)

**Polymorphism**

When one task is performed by different ways i.e. known as polymorphism. For example: to convince the customer differently, to draw something e.g. shape or rectangle etc.

In java, we use method overloading and method overriding to achieve polymorphism.

Another example can be to speak something e.g. cat speaks meaw, dog barks woof etc.

**Encapsulation** is one of the four fundamental OOP concepts. The other three are inheritance, polymorphism, and abstraction.

Encapsulation in Java is a mechanism of wrapping the data (variables) and code acting on the data (methods) together as a single unit. In encapsulation, the variables of a class will be hidden from other classes, and can be accessed only through the methods of their current class. Therefore, it is also known as data hiding.

To achieve encapsulation in Java −

Declare the variables of a class as private.

Provide public setter and getter methods to modify and view the variables values.

Example

Following is an example that demonstrates how to achieve Encapsulation in Java −

/\* File name : EncapTest.java \*/

public class EncapTest {

private String name;

private String idNum;

private int age;

public int getAge() {

return age;

}

public String getName() {

return name;

}

public String getIdNum() {

return idNum;

}

public void setAge( int newAge) {

age = newAge;

}

public void setName(String newName) {

name = newName;

}

public void setIdNum( String newId) {

idNum = newId;

}

}

The variables of the EncapTest class can be accessed using the following program –

\* File name : RunEncap.java \*/

public class RunEncap {

public static void main(String args[]) {

EncapTest encap = new EncapTest();

encap.setName("James");

encap.setAge(20);

encap.setIdNum("12343ms");

System.out.print("Name : " + encap.getName() + " Age : " + encap.getAge());

}

}

Benefits of Encapsulation

The fields of a class can be made read-only or write-only.

A class can have total control over what is stored in its fields.

The users of a class do not know how the class stores its data. A class can change the data type of a field and users of the class do not need to change any of their code.

**What is an interface?**

Interface looks like class but it is not a class. An interface can have methods and variables just like the class but the methods declared in interface are by default abstract (only method signature, no body). Also, the variables declared in an interface are public, static & final by default. We will discuss these points in detail, later in this post.

What is the use of interfaces?

As mentioned above they are used for abstraction. Since methods in interfaces do not have body, they have to be implemented by the class before you can access them. The class that implements interface must implement all the methods of that interface. Also, java programming language does not support multiple inheritance, using interfaces we can achieve this as a class can implement more than one interfaces, however it cannot extend more than one classes.

Declaration  
Interfaces are declared by specifying a keyword “interface”. E.g.:

interface MyInterface

{

/\* All the methods are public abstract by default

\* Note down that these methods are not having body

\*/

public void method1();

public void method2();

}

Interface Implementation

This is how a class implements an interface. It has to provide the body of all the methods that are declared in interface.  
Note: class implements interface but an interface extends another interface.

interface MyInterface

{

public void method1();

public void method2();

}

class XYZ implements MyInterface

{

public void method1()

{

System.out.println("implementation of method1");

}

public void method2()

{

System.out.println("implementation of method2");

}

public static void main(String arg[])

{

MyInterface obj = new XYZ();

obj. method1();

}

}

Key points: Here are the key points to remember about interfaces:  
1) We can’t instantiate an interface in java.

2) Interface provides complete [abstraction](https://beginnersbook.com/2013/03/oops-in-java-encapsulation-inheritance-polymorphism-abstraction/) as none of its methods can have body. On the other hand, [abstract class](https://beginnersbook.com/2013/05/java-abstract-class-method/) provides partial abstraction as it can have abstract and concrete(methods with body) methods both.

3) implements keyword is used by classes to implement an interface.

4) While providing implementation in class of any method of an interface, it needs to be mentioned as public.

5) Class implementing any interface must implement all the methods, otherwise the class should be declared as “abstract”.

6) Interface cannot be declared as private, protected or transient.

7) All the interface methods are by default abstract and public.

8) Variables declared in interface are public, static and final by default.

interface Try

{

int a=10;

public int a=10;

public static final int a=10;

final int a=10;

static int a=0;

}

All of the above statements are identical.

9) Interface variables must be initialized at the time of declaration otherwise compiler will through an error.

interface Try

{

int x;//Compile-time error

}

Above code will throw a compile time error as the value of the variable x is not initialized at the time of declaration.

10) Inside any implementation class, you cannot change the variables declared in interface because by default, they are public, static and final. Here we are implementing the interface “Try” which has a variable x. When we tried to set the value for variable x we got compilation error as the variable x is public static **final** by default and final variables can not be re-initialized.

class Sample implements Try

{

public static void main(String args[])

{

x=20; //compile time error

}

}

11) Any interface can extend any interface but cannot implement it. Class implements interface and interface extends interface.

12) A **class** can implement any **number of interfaces**.

13) If there are **two or more same methods** in two interfaces and a class implements both interfaces, implementation of the method once is enough.

interface A

{

public void aaa();

}

interface B

{

public void aaa();

}

class Central implements A,B

{

public void aaa()

{

//Any Code here

}

public static void main(String args[])

{

//Statements

}

}

14) A class cannot implement two interfaces that have methods with same name but different return type.

interface A

{

public void aaa();

}

interface B

{

public int aaa();

}

class Central implements A,B

{

public void aaa() // error

{

}

public int aaa() // error

{

}

public static void main(String args[])

{

}

}

15) Variable names conflicts can be resolved by interface name e.g:

interface A

{

int x=10;

}

interface B

{

int x=100;

}

class Hello implements A,B

{

public static void Main(String args[])

{

// reference to x is ambiguous both variables are x

System.out.println(x);

System.out.println(A.x);

System.out.println(B.x);

}

}

**Benefits of having interfaces:**

Following are the advantages of using interfaces:

Without bothering about the implementation part, we can achieve the security of implementation

In java, [**multiple inheritance**](https://beginnersbook.com/2013/05/java-multiple-inheritance/) is not allowed, However by using interfaces you can achieve the same . A class can extend only one class but can implement any number of interfaces. It saves you from Deadly Diamond of Death(DDD) problem.

Accessibility Modifiers for Members:

public Members: If members are declared as public inside a class then such members are accessible to the classes which are inside and outside of the package where this class is visible. This is the least restrictive of all the accessibility modifiers.

protected Members:If members are declared as protected then these are accessible to all classes in the package and to all subclasses of its class in any package where this class is visible.

Default Members: When no accessibility modifier is specified for the member then implicitly it is declared as Default. These are accessible only to the other classes in the class’s package.

private Members:This is the most restrictive of all accessibility modifiers. These members are accessible only with in the same class. These are not accessible from any other class within a class’s package also.

Abstract Methods:

If method has a keyword *abstract* in its declaration, then such method/function is called Abstract method. Abstract methods does not have an implementation i.e. method body is not defined; only method prototype is specified in the class definition.

Note:

Only instance methods can be declared as *abstract*.

Since Static methods cannot be overridden declaring abstract static method would of no use.

A Final method cannot be abstract and vice versa.

Methods specified in an Interface are implicitly *abstract*.

Synchronized Methods:

Multiple threads can be executing in a program and at times they might try to execute several methods on the same object simultaneously.

If there is a requirement that only one thread at a time should execute a method in the object, then such methods can be declared as *Synchronized*. Their execution will be mutually exclusive among all threads. At any given time, at the most one thread will be executing a Synchronized method on an object.

Note:  Synchronized methods are also applicable to Static methods of a class.

Native Methods:

Native Methods are also called as Foreign methods. Such methods implementation is not defined in Java but in another programming language.

These methods are specified in the class as method prototypes with prefix with keyword native, no method body is defined in the Java class.

class cNat() {

native void fNat(); // native method declaration

synchronized public void fSyn() {…}; // synchronized method declaration

}

class cmain {

public static void main (String[] args) {

cNat cNatObj =new cNat(); // class instance creation

cNatObj.fNAt(); // accessing native method

}

}

Constructor in Java

**Constructor in java** is a special type of method that is used to initialize the object.

Java constructor is invoked at the time of object creation. It constructs the values i.e. provides data for the object that is why it is known as constructor.

Rules for creating java constructor

There are basically two rules defined for the constructor.

Constructor name must be same as its class name

Constructor must have no explicit return type

Types of java constructors

There are two types of constructors:

Default constructor (no-arg constructor)

Parameterized constructor

class Bike1{

Bike1(){System.out.println("Bike is created");}

public static void main(String args[]){

Bike1 b=new Bike1();

}  }

Transient Fields:

Objects can be stored using serialization. Serialization transforms objects into an output format which is helpful for storing objects. Objects can later be retrieved in the same state as when they were serialized, meaning that fields included in the serialization will have the same values at the time of serialization. Such objects are said to be Persistent.

The fields are declared with keyword Transient in their class declaration if its value should not be saved when objects of the class are written to persistent storage.

class sample implements Serializable{

transient str varStr; // transient field declaration

int varInt; // instance field declaration

}

Volatile Fields:

During execution, complied code might cache the values of fields for efficiency reasons. And as multiple threads will access the same field, caching is not allowed to cause inconsistencies when reading and writing the value in the field.

The *Volatile* modifier can be used to inform the compiler that it should not attempt to perform optimizations on the field which could cause unpredicted results when the field is accessed by multiple threads

class sample {

volatile int varInt; // volatile field declaration

}

3) Accessibility Modifiers for Nested Classes & Interfaces

Nested Interfaces:

Access modifiers can be used in Nested interfaces. An interface declared within another class or interface is called a Nested interface. A top-level interface is the one which is not nested.

Only one type of nested interface is available in Java based on declarative context, Static member interface. These are interfaces defined with keyword Static inside the top-level interface or class or in another Static member class or interface. It can be instantiated like a normal top-level interface or class, no enclosing instance is required for this interface instantiation.

Nested Classes:

Access modifiers are also used in Nested classes. A class declared within another class is called a Nested class. A top-level class is the one which is not nested.

They are 2 categories of Nested classes based on the declarative context. They are

Static member classes

Inner classes

Inner classes are defined in 3 different categories. They are

Non-static member classes

Local classes

Anonymous classes

Static member classes:

Classes which are defined with Static modifier inside the top-level class or other Static member class are called Static member classes. It can be instantiated like a normal top-level class; no enclosing instance is required for this class instantiation.

All the 4 accessibility modifiers i.e. Public, Protected, Package & Private are applicable to Static member classes’ declaration.

Non-Static member classes:

Classes which are defined without Static modifier inside another class are called non-static member classes. An instance of a non-static member class always has an enclosing instance associated with it.

The accessibility modifiers i.e. Public, Protected, Package & Private, abstract, final are applicable to Non-Static member classes’ declaration.

Local classes:

These classes are defined in the context of a block as in a body of the method or normal local block, just as local variables defined in a method body or local block. An instance of a local class has an enclosing instance associated with it, if it is declared in non-static context.

No accessibility modifiers are applicable for Local classes.

Anonymous classes:

These are defined as expressions and instantiated on the fly. An instance of anonymous class has an enclosing instance associated with it, if it is declared in non-static context.

No accessibility modifiers are applicable for Anonymous classes.

Note: A Nested Class or Interface cannot have the same name as any of the enclosing classes or interfaces.

class A {                // Top-level Class

static class B {…}       // Static member class

static interface C {…}   // Static member interface

class D {….}             // Non-static inner class

void func () {

class E {…….}           // Local class in non-static context

static void func1 () {

class F {…}              // Local class in static context

}

B bx = new B () {...} // Anonymous class in non static context

static C cx= new C () {……} // Anonymous class in static context

}

Final keyword in java

It is used to make a variable as a constant, Restrict method overriding, Restrict inheritance. It is used at variable level, method level and class level. In java language final keyword can be used in following way.

Final at variable level

Final at method level

Final at class level



Static keyword in java

The **static keyword** is used in java mainly for memory management. It is used with variables, methods, blocks and nested class. It is a keyword that are used for share the same variable or method of a given class. This is used for a constant variable or a method that is the same for every instance of a class. The main method of a class is generally labeled static.

No object needs to be created to use static variable or call static methods, just put the class name before the static variable or method to use them. Static method can not call non-static method.

In java language static keyword can be used for following

variable (also known as class variable)

method (also known as class method)

block

nested class

Static variable

If any variable we declared as static is known as static variable.

Static variable is used for fulfill the common requirement. For Example company name of employees, college name of students etc. Name of the college is common for all students.

The static variable allocate memory only once in class area at the time of class loading.

Advantage of static variable

Using static variable we make our program memory efficient (i.e it saves memory).

When and why we use static variable

Suppose we want to store record of all employee of any company, in this case employee id is unique for every employee but company name is common for all. When we create a static variable as a company name then only once memory is allocated otherwise it allocate a memory space each time for every employee.

Difference between static and final keyword

static keyword always fixed the memory that means that will be located only once in the program where as final keyword always fixed the value that means it makes variable values constant.

Note: As for as real time statement there concern every final variable should be declared the static but there is no compulsion that every static variable declared as final.

**Collections in Java**

[Java Collection Framework](https://www.javatpoint.com/collections-in-java)

[Hierarchy of Collection Framework](https://www.javatpoint.com/collections-in-java#collectionhierarchy)

[Collection interface](https://www.javatpoint.com/collections-in-java#collectionmethods)

[Iterator interface](https://www.javatpoint.com/collections-in-java#collectioniterator)

Collections in java is a framework that provides an architecture to store and manipulate the group of objects.

All the operations that you perform on a data such as searching, sorting, insertion, manipulation, deletion etc. can be performed by Java Collections.

Java Collection simply means a single unit of objects. Java Collection framework provides many interfaces (Set, List, Queue, Deque etc.) and classes (ArrayList, Vector, LinkedList, PriorityQueue, HashSet, LinkedHashSet, TreeSet etc).

What is Collection in java

Collection represents a single unit of objects i.e. a group.

What is framework in java

provides readymade architecture.

represents set of classes and interface.

is optional.

What is Collection framework

Collection framework represents a unified architecture for storing and manipulating group of objects. It has:

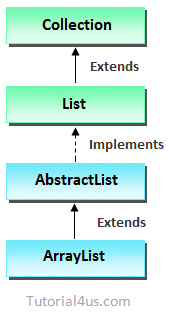
Interfaces and its implementations i.e. classes

Algorithm

**ArrayList in Java**

ArrayList is a replacement of vector class, It is a new class used to store multiple objects.

In ArrayList the data is organizing in the form of cells. Cell values are storing in heap memory and cell address are storing in associative memory.



Points to Remember

ArrayList class is not Synchronized.

ArrayList class elements can be access randomly.

In the ArrayList value will be stored in the same order as inserted.

ArrayList class uses a dynamic array for storing the elements.It extends AbstractList class and implements List interface.

ArrayList class can contain duplicate elements.

ArrayList allows random access because array works at the index basis.

**Note:** ArrayList calss contains same methods like vector.

Creating ArrayList is nothing but creating an object of ArrayList class.

Syntax

ArrayList al=new ArrayList();

Arraylist Constructor

**ArrayList():** This constructor is used for creating an object of ArrayList class.

Syntax

ArrayList al=new ArrayList()

Advantages of ArrayList

ArrayList based applications takes less memory space.

Retrieving the data from ArrayList will take less time.

Performance of ArrayList based applications is more.

Difference Between Vector and ArrayList

|  |  |  |
| --- | --- | --- |
|  | Vector | ArrayList |
| 1 | Vector is legacy Collection Framework (old class). | ArrayList is new Collection Framework. |
| 2 | Vector is Synchronized by default. | ArrayList is not Synchronized. |
| 3 | For retrieving elements from Vector class can be use foreach loop, iterator, listiterator and enumeration. | For retrieving elements from ArrayList class can be use foreach loop, iterator and listiterator. |

Example of ArrayList

import java.util.Arraylist;

class DemoArraylist

{

public static void main(String args[])

{

ArrayList<Integer> al=new ArrayList<Integer>(); // creating arraylist

al.add(10);

al.add(20);

al.add(30);

Iterator itr=al.iterator(); // getting Iterator from arraylist to traverse elements

while(itr.hasNext())

{

System.out.println(itr.next());

}

}

}

Output

10

20

30

Initially hasNext() points to the first element in the list.   
and when you do next() it returns the current element and cursor starts pointing to the next element.  
lets say we have list with following elements :  
1 , 2, 3, 4, 5

So while iterating the list, hasNext() points to 1 initially and using next() we retrieve the value and now the cursor points to next element i.e. 2. When pointer reaches to 5 hasNext() returns false.

**Java LinkedList class**

Java LinkedList class hierarchy

Java LinkedList class uses doubly linked list to store the elements. It provides a linked-list data structure. It inherits the AbstractList class and implements List and Deque interfaces.

The important points about Java LinkedList are:

Java LinkedList class can contain duplicate elements.

Java LinkedList class maintains insertion order.

Java LinkedList class is non synchronized.

In Java LinkedList class, manipulation is fast because no shifting needs to be occurred.

Java LinkedList class can be used as list, stack or queue.

Hierarchy of LinkedList class

As shown in above diagram, Java LinkedList class extends AbstractSequentialList class and implements List and Deque interfaces.

Doubly Linked List

In case of doubly linked list, we can add or remove elements from both side.

java LinkedList class using doubly linked list LinkedList class declaration

Let's see the declaration for java.util.LinkedList class.

public class LinkedList<E> extends AbstractSequentialList<E> implements List<E>, Deque<E>, Cloneable, Serializable

Constructors of Java LinkedList

|  |  |
| --- | --- |
| Constructor | Description |
| LinkedList() | It is used to construct an empty list. |
| LinkedList(Collection c) | It is used to construct a list containing the elements of the specified collection, in the order they are returned by the collection's iterator. |

import java.util.\*;

public class TestCollection7{

 public static void main(String args[]){

  LinkedList<String> al=new LinkedList<String>();

  al.add("Ravi");

  al.add("Vijay");

  al.add("Ravi");

  al.add("Ajay");

  Iterator<String> itr=al.iterator();

  while(itr.hasNext()){

   System.out.println(itr.next());

  }

 }

}

[Test it Now](http://www.javatpoint.com/opr/test.jsp?filename=TestCollection7)

Output:Ravi

Vijay

Ravi

Ajay

**Difference between ArrayList and LinkedList**

ArrayList and LinkedList both implements List interface and maintains insertion order. Both are non synchronized classes.

But there are many differences between ArrayList and LinkedList classes that are given below.

|  |  |
| --- | --- |
| ArrayList | LinkedList |
| 1) ArrayList internally uses **dynamic array** to store the elements. | LinkedList internally uses **doubly linked list** to store the elements. |
| 2) Manipulation with ArrayList is **slow** because it internally uses array. If any element is removed from the array, all the bits are shifted in memory. | Manipulation with LinkedList is **faster** than ArrayList because it uses doubly linked list so no bit shifting is required in memory. |
| 3) ArrayList class can **act as a list** only because it implements List only. | LinkedList class can **act as a list and queue** both because it implements List and Deque interfaces. |
| 4) ArrayList is **better for storing and accessing** data. | LinkedList is **better for manipulating** data. |

Example of ArrayList and LinkedList in Java

Let's see a simple example where we are using ArrayList and LinkedList both.

import java.util.\*;

class TestArrayLinked{

 public static void main(String args[]){

  List<String> al=new ArrayList<String>();//creating arraylist

  al.add("Ravi");//adding object in arraylist

  al.add("Vijay");

  al.add("Ravi");

  al.add("Ajay");

  List<String> al2=new LinkedList<String>();//creating linkedlist

  al2.add("James");//adding object in linkedlist

  al2.add("Serena");

  al2.add("Swati");

  al2.add("Junaid");

  System.out.println("arraylist: "+al);

  System.out.println("linkedlist: "+al2);

 }

}

[**Test it Now**](http://www.javatpoint.com/opr/test.jsp?filename=TestArrayLinked)

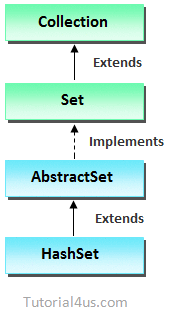
Output:

arraylist: [Ravi,Vijay,Ravi,Ajay]

linkedlist: [James,Serena,Swati,Junaid]

HashSet in Collection Framework

HashSet is **Implementer** class of Set Interface. HashSet supports hashing mechanism to store the value that means all the elements are stored in un-shorted random order (the elements will not be in same order as inserted).



Points to Remember

HashSet are not allows to store duplicate elements.

HashSet allows to store heterogeneous elements.

For retrieving elements from HashSet you can use foreach loop and iterator interface to retrieve the elements.

HashSet is not Synchronized means multiple threads can work Sanctimoniously.

HashSet allows to store null value.

Example of HashSet

import java.util.\*;

class HashSetDemo

{

public static void main(String args[])

{

HashSet<String> hs=new HashSet<String>();

hs.add("Java");

hs.add("C-lang");

hs.add("C++");

hs.add("Java");

System.out.println(hs);

Iterator i=hs.iterator();

System.out.println("Forward Direction");

while(l.hasNext())

{

System.out.println(i.next());

}

}

}

Output

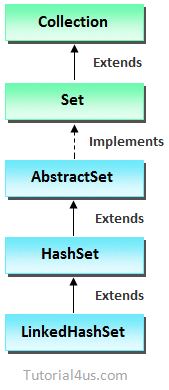
Java

C-lang

C++

LinkedHashSet in Java

LinkedHashSet is **Implementer** class of Set Interface, Which supports hashing mechanism to store the value that means all the elements are stored in the same order as inserted (In unsorted format).



Points to Remember

LinkedHashSet does not allow duplicate elements.

LinkedHashSet allows to store heterogeneous elements

LinkedHashSet is not Synchronized.

For retrieving elements from LinkedHashSet you can use foreach loop and iterator interface.

LinkedHashSet allows null values.

Example of LinkedHashSet

import java.util.\*;

class LHashDemo

{

public static void main(String args[])

{

LinkedHashSet<String> lhs=new LinkedHashSet<String>();

lhs.add("Java");

lhs.add("C-lang");

lhs.add("C++");

lhs.add("Java");

System.out.println(lhs);

Iterator i=lhs.iterator();

System.out.println("Forward Direction");

while(l.hasNext())

{

System.out.println(i.next());

}

}

}

Output

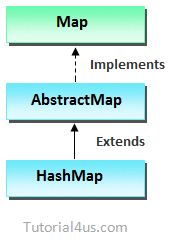
Java

C-lang

C++

HashMap in Java

HashMap are the **Implementer** class of Map Interface, which store the values based on the key.



Points to Remember

HashMap are not allows to store duplicate elements.

HashMap are new collection framework class.

HashMap may have one null key and multiple null values.

For retrieving elements from HashMap you can use foreach loop, Iterator Interface and ListIterator Interface to retrieve the elements.

HashMap is not Synchronized means multiple threads can work Simultaneously.

Example of HashMap

import java.util.\*;

class HashMapDemo

{

public static void main(String args[])

{

HashMap<Integer,String> hm=new HashMap<Integer,String>();

hm.put(1,"Deo");

hm.put(2,"zen");

hm.put(3,"porter");

hm.put(4,"piter");

for(Map.Entry m:hm.entrySet())

{

System.out.println(m.getKey()+" "+m.getValue());

}

}

}

Output

1 Deo

2 zen

3 porter

4 piter

Example of HashMap

import java.util.\*;

class HashMapDemo

{

public static void main(String args[])

{

HashMap<Integer, Float> hm=new HashMap<Integer, Float>();

hm.add(10, 10.5);

hm.add(20,20.5);

hm.add(30,30.5);

hm.add(40,40.5);

hm.add(Null,50.5);

hm.add(Null,60.5); // only one null allow

System.out.println(hm);

System.out.println(hm.values());

System.out.println(hm.keyset());

System.out.println(hm.get(20));

}

}

Output

[null=60.5, 20=20.5, 10=10.5, 40=40.5, 30=30.5]

[60.5, 20.5, 10.5, 40.5, 30.5]

[Null, 20, 10, 30, 40]

[20.5]

HashTable in Java

HashTable is **Implementer** class of Map interface and extends Dictionary class. HashTable does not allows null key and null values, these elements will be stored in a random order.

Points to Remember

HashTable is a legacy class, which will uses hashing technique (the elements will be stored in unsorted or un-order format).

HashTable stored the elements in key values formate.

HashTable does not allows null keys and null values.

HashTable is Synchronized.

HashTable allows heterogeneous elements.

For retrieve elements from HashTable you can use foreach loop, Iterator Interface and Enumeration.

**Note:** HashTable class also contains same methods like HashMap.

Difference between HashMap and HashTable

|  |  |  |
| --- | --- | --- |
|  | HashMap | HashTable |
| 1 | HashMap is a new class in java API. | HashTable is a legacy class in java API. |
| 2 | HashMap is not Synchronized. | HashTable is Synchronized. |
| 3 | HashMap allows maximum one null key and multiple null value. | HashTable does not allows any null key and null values. |

Limitations of HashTable

Hashtable class object is enable to read the data from property file / resource bundle file.

Hashtable class object is unable to develop flexible application.

Example of HashTable

import java.util.\*;

class HashTableDemo

{

public static void main(String args[])

{

HashTable<Integer,String> ht=new HashTable<Integer,String>();

ht.put(1,"Deo");

ht.put(2,"zen");

ht.put(3,"porter");

ht.put(4,"piter");

System.out.println(ht);

}

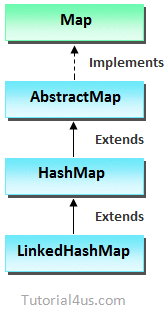
}

Output

[1=Deo, 2=zen, 3=porter ,4=piter]

LinkedHashMap in Java

LinkedHashMap class are **Implementer** class of Map Interface which contains values based on the key. It implements the Map interface and extends HashMap class.



Points to Remember

LinkedHashMap contains only unique elements.

LinkedHashMap have one null key or can have multiple null values.

It is same as HashMap instead maintains insertion order.

Example of LinkedHashMap

import java.util.\*;

class LinkedHashMapDemo

{

public static void main(String args[])

{

LinkedHashMap<Integer,String> tm=new LinkedHashMap<Integer,String>();

lhm.put(1,"Deo");

lhm.put(2,"zen");

lhm.put(3,"porter");

lhm.put(4,"piter");

for(Map.Entry m:lhm.entrySet())

{ System.out.println(m.getKey()+" "+m.getValue());

}

}

}

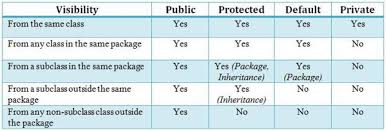
Output

1 Deo

2 zen

3 porter

4 piter

[](https://www.google.co.in/imgres?imgurl=http://4.bp.blogspot.com/-ZGaV8pq-HGs/Vdhz3bY2oiI/AAAAAAAADoQ/Mb0zjguSOR0/s1600/public%2Bvs%2Bprivate%2Bvs%2Bprotected%2Bvs%2Bdefault%2Bmodifier%2Bin%2BJava.jpg&imgrefurl=http://www.java67.com/2015/08/difference-between-public-private-and-protected-in-java.html&docid=qixY-bJ0n0_e7M&tbnid=ARdbtcZ6-A1l1M:&vet=10ahUKEwiFycO13uDVAhWThRoKHacAD0kQMwhGKBcwFw..i&w=1280&h=434&bih=770&biw=1440&q=difference%20between%20string%20and%20stringbuffer%20in%20java%20in%20tabular%20form&ved=0ahUKEwiFycO13uDVAhWThRoKHacAD0kQMwhGKBcwFw&iact=mrc&uact=8)

Difference Between String , StringBuilder and StringBuffer Classes with Example : Java

Today we are going to understand the difference between String , StringBuilder and StringBuffer . As you will find that there are minor differences between the above mentioned classes.  
  
String  
  
String is *immutable*  ( once created can not be changed )object  . The object created as a String is stored in the  Constant String Pool  .   
Every immutable object in Java is thread safe ,that implies String is also thread safe . String can not be used by two threads simultaneously.  
String  once assigned can not be changed.  
  
String  demo = " hello " ;  
// The above object is stored in constant string pool and its value can not be modified.  
  
  
demo="Bye" ;     //new "Bye" string is created in constant pool and referenced by the demo variable              
 // "hello" string still exists in string constant pool and its value is not overrided but we lost reference to the  "hello"string    
  
StringBuffer  
  
StringBuffer is mutable means one can change the value of the object . The object created through StringBuffer is stored in the heap . StringBuffer  has the same methods as the StringBuilder , but each method in StringBuffer is synchronized that is StringBuffer is thread safe .   
  
Due to this it does not allow  two threads to simultaneously access the same method . Each method can be accessed by one thread at a time .  
  
But being thread safe has disadvantages too as the performance of the StringBuffer hits due to thread safe property . Thus  StringBuilder is faster than the StringBuffer when calling the same methods of each class.  
  
StringBuffer value can be changed , it means it can be assigned to the new value . Nowadays its a most common interview question ,the differences between the above classes .  
String Buffer can be converted to the string by using   
toString() method.  
  
StringBuffer demo1 = new StringBuffer("Hello") ;  
// The above object stored in heap and its value can be changed .

demo1=new StringBuffer("Bye");  
// Above statement is right as it modifies the value which is allowed in the StringBuffer  
  
StringBuilder  
  
StringBuilder  is same as the StringBuffer , that is it stores the object in heap and it can also be modified . The main difference between the StringBuffer and StringBuilder is that StringBuilder is also not thread safe.   
StringBuilder is fast as it is not thread safe .    
  
  
StringBuilder demo2= new StringBuilder("Hello");  
// The above object too is stored in the heap and its value can be modified  
demo2=new StringBuilder("Bye");   
// Above statement is right as it modifies the value which is allowed in the StringBuilder  
  
  
----------------------------------------------------------------------------------  
                                  *String*                  *StringBuffer*        *StringBuilder*  
----------------------------------------------------------------------------------                   
Storage Area | Constant String Pool           Heap                       Heap   
Modifiable     |  No (immutable)            Yes( mutable )          Yes( mutable )  
Thread Safe   |           Yes                                  Yes                              No  
 Performance |         Fast                                Very slow                    Fast

Java String Constant Pool: Concept & Mechanism

Java String Constant Pool

When you declare a new string in Java, there are some interesting things that happen behind the scenes. This is a basic string declaration. We create a new string variable called *employee* and give it a value.

|  |
| --- |
| Java basic string declaration |

Not only will Java create the variable *employee*, it will allocate space in the memory for the literal value 'Edgar Allen Poe.' This area in memory is called the string constant pool. It is like a pool of string values that are available to other parts of the program.

Now, if you created another variable, say *employee2*, and ALSO gave it a value of 'Edgar Allen Poe,' Java simply re-uses the value that's already in the pool.

|  |
| --- |
| Java string constant pool literal |

You'll notice the string constant pool sits inside a section of memory is called the heap. This is a part of memory that is used for run-time operations, working with classes and objects. Think of a heap of garden soil you can easily take from as you plant a garden. Java places these new objects there. If you create a hundred more objects, Java will create a hundred more literals atop the heap.

Creating a New Instance of String

If you create a new instance of the String class instead, the constant pool works differently. Let's stay with the employee example and create yet another variable, *employee3*, and also give it the same literal value. This time, however, we will create a new instance of the String class:

|  |
| --- |
| Java create new instance of String |

When this code is processed, Java will act a little differently. Instead of re-using the same literal again, it will create a NEW value in memory. In this case, it does NOT create it in the string constant pool, but in the memory heap.

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Java constant string pool objects   | BASIS FOR COMPARISON | CHECKED EXCEPTION | UNCHECKED EXCEPTION | | --- | --- | --- | | Basic | The compiler checks the checked exception. | The compiler does not check the Unchecked exception. | | Class of Exception | Except "RuntimeException" class all the child classes of the class "Exception", and the "Error" class and its child classes are Checked Exception. | "RuntimeException" class and its child classes, are"Unchecked Exceptions". | | Handling | If we do not handle the checked exception, then the compiler objects. | Even if we do not handle the unchecked exception, the compiler doesn't object. | | Compilation | The program doesn't compile if there is an unhandled checked exception in the program code. | The program compiles successfully even if there is an unhandled unchecked exception in the program code. |   Difference between throw and throws in Java  There are many differences between throw and throws keywords. A list of differences between throw and throws are given below:   |  |  |  | | --- | --- | --- | | No. | throw | throws | | 1) | Java throw keyword is used to explicitly throw an exception. | Java throws keyword is used to declare an exception. | | 2) | Checked exception cannot be propagated using throw only. | Checked exception can be propagated with throws. | | 3) | Throw is followed by an instance. | Throws is followed by class. | | 4) | Throw is used within the method. | Throws is used with the method signature. | | 5) | You cannot throw multiple exceptions. | You can declare multiple exceptions e.g. public void method()throws IOException,SQLException. | |
| Question-1: Write code to filter duplicate elements from an array and print as a list?  findDuplicates.java   |  |  | | --- | --- | | 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35  36  37  38  39  40 | package simple.test;    import java.util.ArrayList;  import java.util.HashSet;  import java.util.List;  import java.util.Set;    public class findDuplicates {    public static void main(String[] args) {    ArrayList<String> list = new ArrayList<String>();    // Form a list of numbers from 0-9.  for (int i = 0; i < 10; i++) {  list.add(String.valueOf(i));  }    // Insert a new set of numbers from 0-5.  for (int i = 0; i < 5; i++) {  list.add(String.valueOf(i));  }    System.out.println("Input list : " + list);  System.out.println("\nFiltered duplicates : " + processList(list));  }    public static Set<String> processList(List<String> listContainingDuplicates) {    final Set<String> resultSet = new HashSet<String>();  final Set<String> tempSet = new HashSet<String>();    for (String yourInt : listContainingDuplicates) {  if (!tempSet.add(yourInt)) {  resultSet.add(yourInt);  }  }  return resultSet;  }  } |     Question-2: Write code to sort the list of strings using Java collection?  sortStrings.java   |  |  | | --- | --- | | 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35  36  37  38 | package simple.test;    import java.util.Arrays;    public class sortStrings {    public static void main(String[] args) throws Exception {    String[] inputList = { "Jan", "Feb", "Mar", "Apr", "May", "Jun", "Jul",  "aug", "Sep", "Oct", "nov", "Dec" };    // Display input un-sorted list.  System.out.println("-------Input List-------");  showList(inputList);    // Call to sort the input list.  Arrays.sort(inputList);    // Display the sorted list.  System.out.println("\n-------Sorted List-------");  showList(inputList);    // Call to sort the input list in case-sensitive order.  System.out.println("\n-------Sorted list (Case-Sensitive)-------");  Arrays.sort(inputList, String.CASE\_INSENSITIVE\_ORDER);    // Display the sorted list.  showList(inputList);  }    public static void showList(String[] array) {  for (String str : array) {  System.out.print(str + " ");  }  System.out.println();  }    } |     Question-3: Write a function to reverse a number in Java?  invertNumber.java   |  |  | | --- | --- | | 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22 | package simple.test;    public class invertNumber {    public long doInvert(long number) {    long invert = 0;  while (number != 0) {  invert = (invert \* 10) + (number % 10);  number = number / 10;  }  return invert;  }    public static void main(String args[]) {  long lnum = 654321;  invertNumber input = new invertNumber();    System.out.println("Input value : " + lnum);  System.out.println("Inverted value : " + input.doInvert(lnum));  }  } |   Question-4: Write a method to check prime no. in Java?  findPrime.java   |  |  | | --- | --- | | 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30 | package simple.test;    import java.util.Scanner;    public class findPrime {    public static void main(String[] args) {  Scanner scan = new Scanner(System.in);  System.out.print("Enter an int value : ");  int input = scan.nextInt();  if (checkPrime(input)) {  System.out.println("Input value " + input + " is a prime number.");  } else {  System.out.println("Input value " + input  + " is not a prime number.");  }  }    public static boolean checkPrime(int n) {  if (n <= 1) {  return false;  }  for (int i = 2; i < Math.sqrt(n); i++) {  if (n % i == 0) {  return false;  }  }  return true;  }  } |     Question-5: Write a Java program to find out the first two max values from an array?  **findTwoMaxValue.java**   |  |  | | --- | --- | | 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29 | package simple.test;    public class findTwoMaxValue {  public void GetTwoMaxValues(int[] nums) {    int maxOne = 0;  int maxTwo = 0;  for (int n : nums) {  if (maxOne < n) {  maxTwo = maxOne;  maxOne = n;  } else if (maxTwo < n) {  maxTwo = n;  }    }    System.out.println("Max1 - " + maxOne);  System.out.println("Max2 - " + maxTwo);  }    public static void main(String[] args) {    int list[] = { 15, 24, 48, 21, 43, 11, 79, 93 };    findTwoMaxValue max = new findTwoMaxValue();  max.GetTwoMaxValues(list);  }  } |     Question-6: Write a Java program to find the longest substring from a given string which doesn’t contain any duplicate characters?  **findSubstr.java**   |  |  | | --- | --- | | 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35  36  37  38  39  40  41  42  43  44  45  46  47  48  49  50  51  52  53  54  55  56  57  58  59  60  61  62  63  64  65  66  67  68  69  70  71  72  73 | package simple.test;    import java.util.HashSet;  import java.util.Set;    public class findSubstr {    private Set<String> stringSet = new HashSet<String>();  private int lstringSet = 0;    public Set<String> findStr(String input) {    // Reset instance data.  stringSet.clear();  lstringSet = 0;    // Set a boolean flag on each char's ASCII value.  boolean[] flag = new boolean[256];  int j = 0;  char[] inputCharArr = input.toCharArray();  for (int i = 0; i < inputCharArr.length; i++) {  char c = inputCharArr[i];  if (flag[c]) {  extractSubString(inputCharArr, j, i);  for (int k = j; k < i; k++) {  if (inputCharArr[k] == c) {  j = k + 1;  break;  }  flag[inputCharArr[k]] = false;  }  } else {  flag[c] = true;  }  }  extractSubString(inputCharArr, j, inputCharArr.length);  return stringSet;  }    private String extractSubString(char[] inputArr, int start, int end) {    StringBuilder sb = new StringBuilder();  for (int i = start; i < end; i++) {  sb.append(inputArr[i]);  }  String subStr = sb.toString();  if (subStr.length() > lstringSet) {  lstringSet = subStr.length();  stringSet.clear();  stringSet.add(subStr);  } else if (subStr.length() == lstringSet) {  stringSet.add(subStr);  }    return sb.toString();  }    public static void main(String a[]) {    findSubstr substr = new findSubstr();    System.out  .println("Actual Strings ------------ | ---- Longest Non-Repeated Strings");  System.out.println("Software\_Programmer"  + "         |         " + substr.findStr("Software\_Programmer"));  System.out.println("Software\_Developer\_In\_Test"  +  "  |         " + substr.findStr("Software\_Developer\_In\_Test"));  System.out.println("developers\_write\_unit\_tests"  +  " |         " + substr.findStr("developers\_write\_unit\_tests"));  System.out.println("javajavbasp.net"  +  "             |         " + substr.findStr("javajavbasp.net"));  }  } |     Question-7: Write Java code to get rid of multiple spaces from a string?  **removeExtraSpaces.java**   |  |  | | --- | --- | | 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20 | package simple.test;    import java.util.StringTokenizer;    public class removeExtraSpaces {      public static void main(String args[]){            String input = "Try    to    remove   extra   spaces.";          StringTokenizer substr = new StringTokenizer(input, " ");          StringBuffer sb = new StringBuffer();            while(substr.hasMoreElements()){              sb.append(substr.nextElement()).append(" ");          }            System.out.println("Actual string: " + input);          System.out.println("Processed string: " + sb.toString().trim());      }  } |     Question-8: Write Java code to identify a number as Palindrome?  **identifyPalindrome.java**   |  |  | | --- | --- | | 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35  36 | package simple.test;    import java.io.BufferedReader;  import java.io.InputStreamReader;    public class identifyPalindrome {    public static void main(String[] args) {    try {  BufferedReader object = new BufferedReader(new InputStreamReader(  System.in));  System.out.println("Input number");  int inputValue = Integer.parseInt(object.readLine());  int n = inputValue;  int rev = 0;  System.out.println("Input value is : ");  System.out.println(" " + inputValue);  for (int i = 0; i <= inputValue; i++) {  int r = inputValue % 10;  inputValue = inputValue / 10;  rev = rev \* 10 + r;  i = 0;  }  System.out.println("Post reversal : " + " ");  System.out.println(" " + rev);  if (n == rev) {  System.out.print("Input value is a palindrome.");  } else {  System.out.println("Input value is not a palindrome.");  }  } catch (Exception e) {  System.out.println("Out of Range.");  }  }  } |     Question-9: Write Java code to swap two numbers without using a temporary variable?  **smartSwapping.java**   |  |  | | --- | --- | | 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22 | package simple.test;    public class smartSwapping {    public static void main(String args[]) {  int numX = 10;  int numY = 20;  System.out.println("Pre-swapping state:");  System.out.println("numX value: " + numX);  System.out.println("numY value: " + numY);  System.out.println("");  numX = numX + numY;  numY = numX - numY;  numX = numX - numY;  System.out.println("Post-swapping state:");  System.out.println("numX value: " + numX);  System.out.println("numY value: " + numY);  }  } |     Question-10: Write a Java program to demonstrate string reverse with and without StringBuffer class?  **invertString.java**   |  |  | | --- | --- | | 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35 | package simple.test;    public class invertString {    public String invertWithStringBuffer(String str) {    StringBuffer buffer = new StringBuffer(str);  buffer.reverse();  return buffer.toString();  }    public String invertWithoutStringBuffer(String str) {    int length = str.length();  String original = str;  String invert = "";  for (int i = length - 1; i >= 0; i--) {  invert = invert + original.charAt(i);  }  return invert;  }    public static void main(String[] args) {    invertString invertStr = new invertString();    System.out.println("Inverted String with StringBuffer class: "  + invertStr.invertWithStringBuffer("987654321"));    System.out.println("");    System.out.println("Inverted String without StringBuffer class: "  + invertStr.invertWithoutStringBuffer("kjihgfedcba"));  }  } |   Reverse String  public void f() {  String input = "Gayatri";  char[] revStr = input.toCharArray();  System.*out*.println(revStr.length);  for(int i=revStr.length-1;i>=0;i--)  System.*out*.print(revStr[i]);  System.*out*.println();  }  public void f1() {  String input1 = "My name is Gayatri";  int index1 = input1.indexOf("is");  System.*out*.println("Index of string is ...."+index1);      }  Remove duplicates from array  public static void main(String args[]){    int array[] = { 10, 20, 30, 20, 40, 40, 50, 60, 70, 80 };// array of ten  // elements  int size = array.length;  System.*out*.println("Size before deletion: " + size);  for (int i = 0; i < size; i++)  {  for (int j = i + 1; j < size; j++)  {  if (array[i] == array[j]) // checking one element with all the  //element  {  while (j < (size) - 1)  {  array[j] = array[j + 1];// shifting the values  j++;  }  size--;  }  }  }  System.*out*.println("Size After deletion: " + size);  for (int k = 0; k < size; k++)  {  System.*out*.println(array[k]); // printing the values  }  }  }  How to find no.of duplicates in array  public static void main(String[] args){  String[] s = {"IBM","Google","Amazon","Google","HP","IBM","Oracle","IBM","HP"};  int duplicate = 0;  String[] sc = s.clone();  for(int i=0; i<s.length;i++)  for (int j=i+1;j<sc.length;j++)  if ((s[i]==sc[j])&&(s[i]!="NULL")){  duplicate++;  sc[j]="NULL";  }  System.*out*.println("Total duplicates - "+duplicate);  }  } |

import java.util.Scanner;

public class MainClass

{

public static void main(String[] args)

{

Scanner sc = new Scanner(System.in);

//Taking rows value from the user

System.out.println(&quot;How many rows you want in this pattern?&quot;);

int rows = sc.nextInt();

System.out.println(&quot;Here is your pattern....!!!&quot;);

for (int i = 1; i <= rows; i++)

{

for (int j = 1; j <= i; j++)

{

System.out.print(j+&quot; &quot;);

}

System.out.println();

}

//Close the resources

sc.close();

}

}

Pattern 1 :

1  
1 2  
1 2 3  
1 2 3 4  
1 2 3 4 5  
1 2 3 4 5 6  
1 2 3 4 5 6 7

public class Pattern {

public static void main(String[] args) {

int rows = 5;

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

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

System.out.print("\* ");

}

System.out.println();

}

}

}

\*

\* \*

\* \* \*

\* \* \* \*

\* \* \* \* \*

1) Mention what is Jenkins? Jenkins is an open source tool with plugin built for continuous integration purpose. The principle functionality of Jenkins is to keep a track of version control system and to initiate and monitor a build system if changes occur. It monitors the whole process and provides reports and notifications to alert.

2) Explain what is continuous integration? In software development, when multiple developers or teams are working on different segments of same web application, we need to perform integration test by integrating all modules. In order to do that an automated process for each piece of code is performed on daily bases so that all your code get tested.

3) What is the requirement for using Jenkins? To use Jenkins you require A source code repository which is accessible, for instance, a Git repository A working build script, e.g., a Maven script, checked into the repository

4) Mention what are the advantages of Jenkins? Advantage of Jenkins include At integration stage, build failures are cached For each code commit changes an automatic build report notification generates To notify developers about build report success or failure, it is integrated with LDAP mail server Achieves continuous integration agile development and test driven development With simple steps, maven release project is automated Easy tracking of bugs at early stage in development environment than production

5) Explain how you can move or copy Jenkins from one server to another? Slide a job from one installation of Jenkins to another by copying the related job directory Make a copy of an already existing job by making clone of a job directory by a different name Renaming an existing job by renaming a directory.

6) Mention what are the commands you can use to start Jenkins manually? To start Jenkins manually, you can use either of the following (Jenkins\_url)/restart: Forces a restart without waiting for builds to complete (Jenkin\_url)/safeRestart: Allows all running builds to complete

7) Mention some of the useful plugins in Jenkin? Some of the important plugins in Jenkin includes Maven 2 project Amazon EC2 HTML publisher Copy artifact Join Green Balls

8) Explain how you can deploy a custom build of a core plugin? To deploy a custom field of a core plugin, you have to do following things Stop Jenkins Copy the custom HPI to $Jenkins\_Home/plugins Delete the previously expanded plugin directory Make an empty file called .hpi.pinned Start Jenkins

9) Explain how can create a backup and copy files in Jenkins? Jenkins saves all the setting, build artifacts and logs in its home directory, to create a back-up of your Jenkins setup, just copy this directory. You can also copy a job directory to clone or replicate a job or rename the directory.

10) Explain how you can clone a Git repository via Jenkins? To clone a Git repository via Jenkins, you have to enter the e-mail and user name for your Jenkins system. For that, you have to switch into your job directory and execute the “git config” command.

11) Explain how you can set up Jenkins job? To create a project that is handled via jobs in Jenkins. Select New item from the menu, once this done enter a name for the job and select free-style job. Then click OK to create new job in Jenkins. The next page enables you to configure your job.

12) Mention what are the two components Jenkins is mainly integrated with? Jenkin is mainly integrated with two components Version Control system like GIT, SVN And build tools like Apache Maven.

Maven

### What is Maven?

Maven is a project management tool. It is based on POM (Project Object Model).

### What aspects are managed by Maven?

* Builds
* Documentation
* Reporting
* SCMs
* Releases
* Distribution

**Q: Why should one use Maven?**  
>> It helps to setup project very quickly and it avoids complicated build files like build.xml. Maven required files like POM.xml; it serves the purpose for Maven only. POM.xml is a collection of dependencies of your Java Project which one can specify to Maven and then Maven will download all of them from the internet and then store it to some repository i.e. local repository, central repository, and remote repository.

>> It helps to not bundle all the jars in your package i.e. in your War file or Ear file because all of them are going store in the repository and wherever you install this application that repository will be used for any dependencies look up. So, your Jar file, War file or Ear file or your bundle deployment will be very light.

**Q: What Maven creates for you?**  
1. Directory Name  
2. Purpose  
3. Project home  
4. Contains the POM.xml and all subdirectories.  
5. Src/main/Java  
6. Contains the deliverable Java source code for the project.  
7. Src/main/resources  
8. Contain the deliverable resources for the project.  
9. Src/test/Java  
10. Contains the testing Java source code.  
11. Src/test/resources  
12. Contains resources necessary for testing.

**Q: What are POM Files in Maven?**  
All your code and resources are placed in the src directory.  
>> The main/Java directory holds your project code.  
>> Compiled code is placed in the largest directory.  
>> The test/Java directory holds your JUnit test code.

**Q: What are Maven’s main objectives?**  
\*\* It helps to make project build easy to work for the users.  
\*\* It easily helps in to migrate from one feature to another or one folder to another.  
\*\* It helps in the new development with proper series of guidelines.  
\*\* It creates a flexible working system for the users. So that you can work uniformly and orderly.

**Q: What is Maven Repository?**  
In Maven a repository is used as a storage folder or a directory to store your projects, your files such as Jar, War or Ear files that can be later used by the Maven application or tool. It works as a whole library of the files that is easily accessible and can be easily located in your system without any trouble and then can be used by Maven.

**Q: How many repositories are there in Maven?**  
There are three types of repository present in Maven. This includes Local Repository, Central Repository, and Remote Repository.  
**Local Repository-** This local repository is located on your local system and it works when you run a maven command. Maven local repository command is %USER\_HOME%/.m2 directory.  
**Central Repository-** Installation from the repository is performed on creating a project from archetype or resolving the dependency.  
**Remote Repository-**This repository is located on the web. It is just a network accessible location that Maven downloads dependencies from. All the artifacts that remote repository contains are open source.

**Q: How to install Maven?**  
Make sure JDK is installed, and ‘JAVA\_HOME’ variable is added as Windows environment variable.  
Add both M2\_HOME and MAVEN\_HOME variable in the Windows environment, and point it to your Maven folder.

**Q: What is the Maven Lifecycle?**  
Lifestyle executed in term of phases:  
1. Maven Steps through phases.  
2. Execution defined in terms of plugin goals.  
3. Execution associated with phases.  
4. Lifecycle completes when all phase executes successfully.

**Q: What is the system requirement for Maven?**  
Maven does not require any high configuration to use. It requires only very minimal and simple system requirements for the users:  
1. Java Deployment Kit  
2. Installed  
3. Configured (JAVA\_HOME)  
4. Internet Connection  
5. For interacting with the repository.  
6. Downloading dependencies.

# **Page Object Model**

### What is the command to check the maven version?

Type the following command on console to know the maven version.

1. mvn -version

### What does the build tool?

* Generates source code (if the auto-generated code is used)
* Generates documentation from source code
* Compiles source code
* Packages compiled code into a JAR or ZIP file
* Installs the packaged code in the local repository, server repository, or central repository

Selenium acts on webelements with the help of their properties such ID, name, XPath, etc. Unlike QTP which has an inbuilt object repository(OR), Selenium has no inbuilt ORs.

Hence we need to build an OR which should also be maintainable and accessible on demand. Page Object Model (POM) is a popular design pattern to create an Object Repository in which each one of those webelements properties are created using a class file.

## Advantages

* POM is an implementation where test objects and functions are separated from each other, thereby keeping the code clean.
* The objects are kept independent of test scripts. An object can be accessed by one or more test scripts, hence POM helps us to create objects once and use them multiple times.
* Since objects are created once, it is easy to access as well as update a particular property of an object.

# **Log4j Logging**

Log4j is an audit logging framework that gives information about what has happened during execution. It offers the following advantages −

* Enables us to understand the application run.
* Log output can be saved that can be analyzed later.
* Helps in debugging, in case of test automation failures.
* Can also be used for auditing purposes to look at the application's health.

## Components

1. Instance of Logger class.

2. Log level methods used for logging the messages as one of the following −

* error
* warn
* info
* debug
* log

# **Java finally block**

**Java finally block** is a block that is used to execute important code such as closing connection, stream etc.

Java finally block is always executed whether exception is handled or not.

Java finally block follows try or catch block.

[next →](https://www.javatpoint.com/throw-keyword)[← prev](https://www.javatpoint.com/nested-try-block)

# **Java finally block**

**Java finally block** is a block that is used to execute important code such as closing connection, stream etc.

Java finally block is always executed whether exception is handled or not.

Java finally block follows try or catch block.

## Why use java finally

* Finally block in java can be used to put "cleanup" code such as closing a file, closing connection etc.

#### Rule: For each try block there can be zero or more catch blocks, but only one finally block.

#### Note: The finally block will not be executed if program exits(either by calling System.exit() or by causing a fatal error that causes the process to abort).

## Java try block

Java try block is used to enclose the code that might throw an exception. It must be used within the method.

Java try block must be followed by either catch or finally block.

## Java Multi catch block

If you have to perform different tasks at the occurrence of different Exceptions, use java multi catch block.

Let's see a simple example of java multi-catch block.

1. **public** **class** TestMultipleCatchBlock{
2. **public** **static** **void** main(String args[]){
3. **try**{
4. **int** a[]=**new** **int**[5];
5. a[5]=30/0;
6. }
7. **catch**(ArithmeticException e){System.out.println("task1 is completed");}
8. **catch**(ArrayIndexOutOfBoundsException e){System.out.println("task 2 completed");}
9. **catch**(Exception e){System.out.println("common task completed");}
11. System.out.println("rest of the code...");
12. }
13. }

#### Rule: At a time only one Exception is occured and at a time only one catch block is executed.

#### Rule: All catch blocks must be ordered from most specific to most general i.e. catch for ArithmeticException must come before catch for Exception .

# **Multi Browser Testing**

Users can execute scripts in multiple browsers simultaneously. For demonstration, we will use the same scenario that we had taken for Selenium Grid. In the Selenium Grid example, we had executed the scripts remotely; here we will execute the scripts locally.

First of all, ensure that you have appropriate drivers downloaded. Please refer the chapter "Selenium Grid" for downloading IE and Chrome drivers.

## Example

For demonstration, we will perform percent calculator in all the browsers simultaneously.

package TestNG;

import org.openqa.selenium.chrome.ChromeDriver;

import org.openqa.selenium.firefox.FirefoxDriver;

import org.openqa.selenium.ie.InternetExplorerDriver;

import java.util.concurrent.TimeUnit;

import org.openqa.selenium.\*;

import org.testng.annotations.\*;

public class TestNGClass {

private WebDriver driver;

private String URL = "http://www.calculator.net";

@Parameters("browser")

@BeforeTest

public void launchapp(String browser) {

if (browser.equalsIgnoreCase("firefox")) {

System.out.println(" Executing on FireFox");

driver = new FirefoxDriver();

driver.get(URL);

driver.manage().timeouts().implicitlyWait(10, TimeUnit.SECONDS);

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

} else if (browser.equalsIgnoreCase("chrome")) {

System.out.println(" Executing on CHROME");

System.out.println("Executing on IE");

System.setProperty("webdriver.chrome.driver", "D:\\chromedriver.exe");

driver = new ChromeDriver();

driver.get(URL);

driver.manage().timeouts().implicitlyWait(10, TimeUnit.SECONDS);

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

} else if (browser.equalsIgnoreCase("ie")) {

System.out.println("Executing on IE");

System.setProperty("webdriver.ie.driver", "D:\\IEDriverServer.exe");

driver = new InternetExplorerDriver();

driver.get(URL);

driver.manage().timeouts().implicitlyWait(10, TimeUnit.SECONDS);

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

} else {

throw new IllegalArgumentException("The Browser Type is Undefined");

}

}

@Test

public void calculatepercent() {

// Click on Math Calculators

driver.findElement(By.xpath(".//\*[@id = 'menu']/div[3]/a")).click();

// Click on Percent Calculators

driver.findElement(By.xpath(".//\*[@id = 'menu']/div[4]/div[3]/a")).click();

// Enter value 10 in the first number of the percent Calculator

driver.findElement(By.id("cpar1")).sendKeys("10");

// Enter value 50 in the second number of the percent Calculator

driver.findElement(By.id("cpar2")).sendKeys("50");

// Click Calculate Button

driver.findElement(By.xpath(".//\*[@id = 'content']/table/tbody/tr/td[2]/input")).click();

// Get the Result Text based on its xpath

String result =

driver.findElement(By.xpath(".//\*[@id = 'content']/p[2]/span/font/b")).getText();

// Print a Log In message to the screen

System.out.println(" The Result is " + result);

if(result.equals("5")) {

System.out.println(" The Result is Pass");

} else {

System.out.println(" The Result is Fail");

}

}

@AfterTest

public void closeBrowser() {

driver.close();

}

}

Create an XML which will help us in parameterizing the browser name and don't forget to mention parallel="tests" in order to execute in all the browsers simultaneously.