Q1). What is the different use of this statement in java?

This is a keyword in java used for referring same class instance variable, same class method, returning same class instance.

Q2). What is run time polymorphism in java?

In java, calling to a method is defined based on the type of object created in runtime. That is run time polymorphism.

Q3). Which method is used to convert String to Char?

by using toCharArray();

Q4). Is it possible to have static method in method overloading?

Yes its possible as method overloading depends only on the number of the argument and data type.

Q5). In a class can a local variable be static?

No

Q6). Is it possible to call finally block after throwing an exception?

Yes. Finally, will always be executed.

Q7). Is finally block always executed?

No. In case of System. exit() it won’t be executed.

Q8). What are the OOPs concept in java?

Inheritance, Abstraction, Polymorphism, Encapsulation, Interface

Q9). Why String is immutable?

As, once the String object is created cannot be modified

Q10). Is multiple inheritance possible in java?

No. Its possible only through interface.

Q11). Can a Interface have complete method.

Yes. After java ver 8, Interface can have complete methods but the method should be static and default.

Q12). Is it possible to have auto widening and auto narrowing in java?

Auto widening is possible but auto narrowing is not.

Q13). Why auto narrowing is not possible in java?

In auto narrowing there is always a loss of memory. So its not possible.

Q14). Where the String objected created?

The string objects are created in String pool area.

Q15). If a String s= “Hi I am Good”. How to separate each word of a String?

By using Split(“ ”) method

Q16). What is encapsulation in java?

In java encapsulation is property, where we hide data member by declaring them as private and try to access them through public methods.

Q17). What is difference in HashMap and HashTable in java?

HashMap can have a Null key and Multiple Null values but HashTable can not have Null key and Null values.

Q18). What is the difference in List and Set in java?

List can have duplicate element but St only have unique element.

Q19). What is recursion in java?

In Java, recursion is a process where a method with in a class calls itself inside the class.

Q20). What is the difference in finally and finalize in java?

Finally, is a java block where finalize is used in garbage collection.

Q21). Can we override static methods in java?

No We cannot override static methods in java.

Q22). What is the difference in Collections and Collection in java.

Collection is an interface in java whereas Collections is a utility class has static java methods inside it.

Q23). What is the difference in Poll and Peak in java collection?

In java collection, poll is used for retrieve data and release memory permanently, but Peak used to retrieve data and release memory temporarily.

Q24). What is boxing and unboxing in java?

In java boxing is used to convert wrapper class to object type and Unboxing is used to get object from wrapper class.

Q25). What is java polymorphism?

In java, an entity is showing different behaviors at different instance of time is known as polymorphism.

Q26). What is java constructor?

When we want to perform an operation/action when an object is created we used constructors.

Q27). What are the different type of constructor?

Java has default and parameterized constructor.

Q28). Which package is used for file reading and writing in java?

Java I/O package is used

Q29). In java how to take input from user?

By using Scanner class

Q30). What  is the difference in array list and linked list ?

Arraylist store memory as continuous memory location but linked list store in random memory locations

Q31). What are different locators in selenium?

ID, name,xpath,CSS,class name, tag name, link text, partial linktext

Q32). What is a web driver ?

Webdriver is an Interface which provide different APIs to perform automation across browsers.

Q33). How to handle select/ dropdown option in selenium?

By using Select class

Q34). How to handle dynamic xpath in selenium?

by using methods such as, text(),Contains(), parent,child,following-siblings,ansestors etc

Q35). How to handle multiple windows in selenium?

By using getWindowHandle()

Q36). What is the difference in getWindowHandle() and getWindowHandles() in selenium?

getWindowHandle() return a String but getWindowHandles() return a collections of String

Q37). How to handle windows popup in selenium?

By using AutoIT

Q38). How to perform keyboard operation in selenium?

By using Action class

Q39). What is Soft assert in selenium?

In selenium soft assert is used to continue the execution without failing the test if any assertion fails. In the end it will collect all exception and fail the test.

Q40). What is ITest result in selenium?

ITest result in selenium is an Listener API used to monitor the execution status of test

Q41). What is the test NG annotations?

@test, @ BeforeMethod, @AfterMethod,@ BeforeClass,@AfterClass, @ Beforegroup, @AfterGroup @BeforeSuite @AfterSuite

Q42). How to Disable an execution of test in TestNG?

@Test(Enable=False);

Q43). How to handle cookies in selenium?

By using getCookies();

Q44). How to delete cookie by name in selenium?

deleteCookiesByNamed(“Name”);

Q45). How to set SSL certificate in selenium?

setAssumeUntrustedCertificateIssuer(False);

Q46). What is Frame handling in selenium?

Frames are different HTML page within a page. In selenium we use driver.switchto.frame()

Q47). In java what is checked exception?

Checked exception are the exception checked by the compiler.

Q48). What is different variable type sin java?

Local , Instance, Static

Q49). What is toString() in java?

toString() is used for String representation of objects.

Q50). What is Selenium grid?

Selenium grid is used to run multiple test across different browsers.

Q51). Write the syntax for to check the xpath in console?

$x[“Value”]

Q52). Explain the Statement WebDriver driver=new Chrome Driver ();

Webdriver is an interface and driver is a reference variable pointing to class chrome driver

Q53). What is Xpath ?

Xpath is XML path in the DOM ( Document object Model)

Q54). Write the syntax for xpath ?

//tagname[@attribute=’Property’]

Q55). How do you select a value from Drop Down ?

* We can select a drop down value from Select Class
* Select sel=new Select ( WebElement)
* selectbyVisibleText or Index or Value

Q56). How do you Select a sub menu from a Main Menu

We can select a sub menu by using Actions Class with move to element method

Q57). What are OOPS Concepts used in Selenium ?

* Below are the following OOPS concepts used
* Inheritance
* Polymorphism
* Encapsulation ( POM )
* Abstraction

Q58). What does the annotation @ Find by do in Selenium

@Find by annotation is used to identify the webelements through its attributes ‘’

Q59). How do you execute a Test Regression Suite in Selenium ?

We can execute the regression suite by creating a test ng xml which includes the test cases

Q60). How can you retrieve the Property from a property file

We can retrieve by using the Properties class and retrieve the key value pair

Q61). Selenium function used for retrieving the attribute or value?

getAttribute(), getText().

Q62). How do get typed text from a textbox?

driver.findElement(By.xpath(“//input[@id=’username’]”)).getAttribute(“value”));

Q63). How do you simulate browser back and forward?

driver.navigate().back();

driver.navigate().forward();

Q64). How do you get the current page URL?

driver.getCurrentUrl();

Q65). How do you achieve synchronization in WebDriver?

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

Q66). How do you clear the contents of a textbox in selenium?

use clear() method.

selenium QA.txt

Displaying selenium QA.txt.

Q67). What is annotations

@Test, @BeforeTest, @AfterTest, @BeforeClass, @AfterClass, @BeforeMethod, @AfterMethod, @BeforeSuite, @AfterSuite.

Q68). What is ghost driver ?

Ghost driver  means it is the JavaScript. It implemented the web driver protocol phantom JS.

Then it can use remote web  driver for back end.

Q69). What is the use of Property file in your framework ?

Properties file is use to store GUI locator and global field database properties .

Mainly it use for JAVA

Q70). What is flash automation

Flash automation means the SWF file is the also in Flash and we can use  HTML file

Q71). How many types of WebDriver APIs using in Selenium?

* Chrome Driver
* Firefox Driver
* Gecko Driver
* InternetExplorer Driver
* Opera Driver
* Android Driver
* iPhone Driver

Q72). Which Operating Systems support to Selenium WebDrivers?

* Windows Operating Systems
* Linux Operating Systems
* Apple Operating Systems

Q73). What is selenium and why it is used by more users?

Selenium is an automation test tool used to test the web application by using the web driver in a browser. It’s an open source so it has large no of user.

Q74). What are the components of selenium?

* Selenium web driver
* Selenium RC
* Selenium Grid
* Selenium IDE

Q75). What are the languages supports in selenium?

* C
* C++
* Java
* Python
* C-sharp
* Ruby

and we need to configure the jars file for each language.

Q76). Write the code to open chrome browser in selenium?

System.setproperty(key,value);

WebDriver driver = new chrome driver ();

diver.close();

Q77). Write the locater to find element in selenium?

By id

By name

By classname

By xpath

By css-selectors

Q78). Write the syntax for Xpath and the two type of xpath?

//tagname[@attributes=”value”]

\*[@attributes=”value”]

Types:

* Absolute xpath
* Relative xpath

Q79). Write the syntax for get element by id and xpath ? And syntax for click button?

Driver.getElement(By.id(“Value”)).sendkeys(“values”);

Driver.getElement(By.Xpath(“//[“Value”]/”));

Driver.getElement(By.id(“Value”).click();

Q80). Write the code to open firefox browser in selenium?

System.setproperty(key,value);

WebDriver driver = new geckodriver ();

diver.close()

Q81). How many test types are support to Selenium?

There are 7 types are supporting

* Integration Testing
* Regression Testing
* Functional Testing
* Smoke Testing
* Responsive Testing
* Cross Browsers Testing
* UI testing (black box)

Q82). How to get the Source code of the webpage?

driver.getPageSource();

Q83). How to disable the Browser level Notifications in Selenium?

ChromeOptions options = new ChromeOptions();

options.addArguments(“–disable-notifications”);

WebDriver driver =new ChromeDriver(options);

Q84). How to go back to main menu from Inner most frame directly?

driver.switchTo().defaultContent();

Q85). How to switch from inner frames to parent frame one by one?

driver.switchTo().parentFrame() ;

Q86). WebDriver interface extends the features from which Interface?

SearchContext interface

Q87). Explain the uses of javaScriptExecutor in Selenium.

* To locate an element if normal locators do not work.
* To pass values instead of sendKeys().
* To generate user defined alert window.
* To capture scrape data(url, title, domain etc).
* To navigate to a url.
* To Scroll the page.
* To open a new Tab.

Q88). How to handle Windows based alert in Selenium?

Selenium does not support this feature.

Q89). How to Resize the browser window?

Dimension d=new Dimension(480, 620);

driver.manage().window().setSize(d);

Q90). What methods are used to scroll the page in javaScriptExecutor?

scrollBy(), scrollTo().

Q91). What is the difference between quit() & close()?

All windows opened by webDriver are closed with quit(). Close() closes only browser window currently webDriver is controlling.

Q92). Can we switch from one frame to another frame through automation?

No we cannot switch from one frame to another frame directly.

Q93). How to switch to a frame?

Browser.switchto.frame()

Q94). How to switch to parent frame?

Browser.switchto.defaultcontent()

Q95). How to retire the data from Encapsulation?

Using getter and setter method we can access the data.

Q96). How to launch a browser ?

static void launchbrowser()

{

try

{

String filepath = System.getProperty("user.dir")+"\\Library\\Driver\\chromedriver.exe";

System.setProperty("webdriver.chrome.driver",filepath );

browser = new ChromeDriver();

}catch(Exception e){

e.printStackTrace();}

}

Q97). How to identify the given drop down and what are the ways?

* Select class
* Robot class

Q98). How to switch to alert ?

Browser.switchto.alert

alert.accept() à ok button

alert.dismiss() à cancel

Q99). How to identify the nth child ?

browser.findElement(By.cssSelector(“div#d3 > form#frm3 > input:nth-child(5)”)).sendKeys(“fifthchild”);

Q100). How many types of wait statements?

* Implicit wait
* Explicit wait
* Fluent Wait

Q101). Does Java support multiple inheritance?

Java does not support Multiple inheritance.

**Q #1) What is Automation Testing?**

Automation testing or Test Automation is a process of automating the manual process to test the application/system under test. Automation testing involves the use of a separate testing tool which lets you create test scripts which can be executed repeatedly and doesn’t require any manual intervention.

**Q #2) What are the benefits of Automation Testing?**

Benefits of Automation testing are:

1. Supports execution of repeated test cases
2. Aids in testing a large test matrix
3. Enables parallel execution
4. Encourages unattended execution
5. Improves accuracy thereby reducing human-generated errors
6. Saves time and money

**Q #3) Why should Selenium be selected as a test tool?**

Selenium

1. is a free and open source
2. have a large user base and helping communities
3. have cross Browser compatibility (Firefox, Chrome, Internet Explorer, Safari etc.)
4. have great platform compatibility (Windows, Mac OS, Linux etc.)
5. supports multiple programming languages (Java, C#, Ruby, Python, Pearl etc.)
6. has fresh and regular repository developments
7. supports distributed testing

**Q #4) What is Selenium? What are the different Selenium components?**

Selenium is one of the most popular automated testing suites. Selenium is designed in a way to support and encourage automation testing of functional aspects of web-based applications and a wide range of browsers and platforms. Due to its existence in the open source community, it has become one of the most accepted tools amongst the testing professionals.

Selenium is not just a single tool or a utility, rather a package of several testing tools and for the same reason, it is referred to as a Suite. Each of these tools is designed to cater different testing and test environment requirements.

The suite package constitutes the following sets of tools:

* [**Selenium Integrated Development Environment (IDE)**](https://www.softwaretestinghelp.com/selenium-ide-download-and-installation-selenium-tutorial-2/) – Selenium IDE is a record and playback tool. It is distributed as a Firefox Plugin.
* **Selenium Remote Control (RC)** – Selenium RC is a server that allows a user to create test scripts in the desired programming language. It also allows executing test scripts within the large spectrum of browsers.
* [**Selenium WebDriver**](https://www.softwaretestinghelp.com/selenium-webdriver-selenium-tutorial-8/) – WebDriver is a different tool altogether that has various advantages over Selenium RC. WebDriver directly communicates with the web browser and uses its native compatibility to automate.
* [**Selenium Grid**](https://www.softwaretestinghelp.com/selenium-grid-selenium-tutorial-29/) – Selenium Grid is used to distribute your test execution on multiple platforms and environments concurrently.

**Q #5) What are the testing types that can be supported by Selenium?**

Selenium supports the following types of testing:

1. Functional Testing
2. Regression Testing

**Q #6) What are the limitations of Selenium?**

Following are the limitations of Selenium:

* Selenium supports testing of only web-based applications
* Mobile applications cannot be tested using Selenium
* Captcha and Barcode readers cannot be tested using Selenium
* Reports can only be generated using third-party tools like TestNG or JUnit.
* As Selenium is a free tool, thus there is no ready vendor support through the user can find numerous helping communities.
* The user is expected to possess prior programming language knowledge.

**Q #7)** **What is the difference between Selenium IDE, Selenium RC, and WebDriver?**

| **Feature** | **Selenium IDE** | **Selenium RC** | **WebDriver** |
| --- | --- | --- | --- |
| Browser Compatibility | Selenium IDE comes as a Firefox plugin, thus it supports only Firefox | Selenium RC supports a varied range of versions of Mozilla Firefox, Google Chrome, Internet Explorer and Opera. | WebDriver supports a varied range of versions of Mozilla Firefox, Google Chrome, Internet Explorer and Opera. Also supports HtmlUnitDriver which is a GUI less or headless browser. |
| Record and Playback | Selenium IDE supports record and playback feature | Selenium RC doesn't supports record and playback feature. | WebDriver doesn't support record and playback feature |
| Server Requirement | Selenium IDE doesn't require any server to be started before executing the test scripts | Selenium RC requires server to be started before executing the test scripts. | WebDriver doesn't require any server to be started before executing the test scripts |
| Architecture | Selenium IDE is a Javascript based framework | Selenium RC is a JavaScript based Framework. | WebDriver uses the browser's native compatibility to automation |
| Object Oriented | Selenium IDE is not an object oriented tool | Selenium RC is semi object oriented tool. | WebDriver is a purely object oriented tool |
| Dynamic Finders (for locating web elements on a webpage) | Selenium IDE doesn't support dynamic finders | Selenium RC doesn't support dynamic finders. | WebDriver supports dynamic finders |
| Handling Alerts, Navigations, Dropdowns | Selenium IDE doesn't explicitly provides aids to handle alerts, navigations, dropdowns | Selenium RC doesn't explicitly provides aids to handle alerts, navigations, dropdowns. | WebDriver offers a wide range of utilities and classes that helps in handling alerts, navigations, and dropdowns efficiently and effectively. |
| WAP (iPhone/Android) Testing | Selenium IDE doesn't support testing of iPhone/Andriod applications | Selenium RC doesn't support testing of iPhone/Android applications. | WebDriver is designed in a way to efficiently support testing of iPhone/Android applications. The tool comes with a large range of drivers for WAP based testing. For example, AndroidDriver, iPhoneDriver |
| Listener Support | Selenium IDE doesn't support listeners | Selenium RC doesn't support listeners. | WebDriver supports the implementation of Listeners |
| Speed | Selenium IDE is fast as it is plugged in with the web-browser that launches the test. Thus, the IDE and browser communicates directly | Selenium RC is slower than WebDriver as it doesn't communicates directly with the browser; rather it sends selenese commands over to Selenium Core which in turn communicates with the browser. | WebDriver communicates directly with the web browsers. Thus making it much faster. |

**Q #8) When should I use Selenium IDE?**

Selenium IDE is the simplest and easiest of all the tools within the Selenium Package. Its record and playback feature makes it exceptionally easy to learn with minimal acquaintances to any programming language. Selenium IDE is an ideal tool for a naïve user.

**Q #9) What is Selenese?**

Selenese is the language which is used to write test scripts in Selenium IDE.

**Q #10)** **What are the different types of locators in Selenium?**

The locator can be termed as an address that identifies a web element uniquely within the webpage. Thus, to identify web elements accurately and precisely we have [different types of locators in Selenium](https://www.softwaretestinghelp.com/using-selenium-xpath-and-other-locators-selenium-tutorial-5/):

* ID
* ClassName
* Name
* TagName
* LinkText
* PartialLinkText
* Xpath
* CSS Selector
* DOM

**Q #11)** **What is the difference between assert and verify commands?**

**Assert:**Assert command checks whether the given condition is true or false. Let’s say we assert whether the given element is present on the web page or not. If the condition is true then the program control will execute the next test step but if the condition is false, the execution would stop and no further test would be executed.

**Verify:**Verify command also checks whether the given condition is true or false. Irrespective of the condition being true or false, the program execution doesn’t halt i.e. any failure during verification would not stop the execution and all the test steps would be executed.

**Q #12) What is an XPath?**

[XPath](https://www.softwaretestinghelp.com/using-selenium-xpath-and-other-locators-selenium-tutorial-5/) is used to locate a web element based on its XML path. XML stands for Extensible Markup Language and is used to store, organize and transport arbitrary data. It stores data in a key-value pair which is very much similar to HTML tags. Both being markup languages and since they fall under the same umbrella, XPath can be used to locate HTML elements.

The fundamental behind locating elements using XPath is the traversing between various elements across the entire page and thus enabling a user to find an element with the reference of another element.

**Q #13) What is the difference between “/” and “//” in Xpath?**

**Single Slash “/” –**Single slash is used to create Xpath with absolute path i.e. the xpath would be created to start selection from the document node/start node.

**Double Slash “//” –** Double slash is used to create Xpath with relative path i.e. the xpath would be created to start selection from anywhere within the document.

**Q #14) What is Same origin policy and how it can be handled?**

The problem of same origin policy disallows to access the DOM of a document from an origin that is different from the origin we are trying to access the document.

Origin is a sequential combination of scheme, host, and port of the URL. For example, for a URL https://www.softwaretestinghelp.com/resources/, the origin is a combination of http, softwaretestinghelp.com, 80 correspondingly.

Thus the Selenium Core (JavaScript Program) cannot access the elements from an origin that is different from where it was launched. For Example, if I have launched the JavaScript Program from “https://www.softwaretestinghelp.com”, then I would be able to access the pages within the same domain such as “https://www.softwaretestinghelp.com/resources” or “https://www.softwaretestinghelp.com/istqb-free-updates/”. The other domains like google.com, seleniumhq.org would no more be accessible.

So, In order to handle the same origin policy, Selenium Remote Control was introduced.

**Q #15)** **When should I use Selenium Grid?**

Selenium Grid can be used to execute same or different test scripts on multiple platforms and browsers concurrently so as to achieve distributed test execution, testing under different environments and saving execution time remarkably.

**Q #16) What do we mean by Selenium 1 and Selenium 2?**

Selenium RC and WebDriver, in a combination, are popularly known as Selenium 2. Selenium RC alone is also referred to as Selenium 1.

**Q #17) Which is the latest Selenium tool?**

WebDriver

**Q #18) How do I launch the browser using WebDriver?**

The following syntax can be used to launch Browser:  
*WebDriver driver =****new****FirefoxDriver();*  
*WebDriver driver =****new****ChromeDriver();*  
*WebDriver driver =****new****InternetExplorerDriver();*

**Q #19) What are the different types of Drivers available in WebDriver?**

The different drivers available in WebDriver are:

* FirefoxDriver
* InternetExplorerDriver
* ChromeDriver
* SafariDriver
* OperaDriver
* AndroidDriver
* IPhoneDriver
* HtmlUnitDriver

**Q #20) What are the different types of waits available in WebDriver?**

There are two [types of waits available in WebDriver](https://www.softwaretestinghelp.com/selenium-webdriver-waits-selenium-tutorial-15/):

1. Implicit Wait
2. Explicit Wait

**Implicit Wait:**Implicit waits are used to provide a default waiting time (say 30 seconds) between each consecutive test step/command across the entire test script. Thus, the subsequent test step would only execute when the 30 seconds have elapsed after executing the previous test step/command.

**Explicit Wait:** Explicit waits are used to halt the execution till the time a particular condition is met or the maximum time has elapsed. Unlike Implicit waits, explicit waits are applied for a particular instance only.

**Q #21)** **How to type in a textbox using Selenium?**

The user can use sendKeys(“String to be entered”) to enter the string in the textbox.

**Syntax:**  
*WebElement username = drv.findElement(By.id(“Email”));*  
*// entering username*  
*username.sendKeys(“sth”);*

**Q #22)** **How can you find if an element in displayed on the screen?**

WebDriver facilitates the user with the following methods to check the visibility of the web elements. These web elements can be buttons, drop boxes, checkboxes, radio buttons, labels etc.

1. isDisplayed()
2. isSelected()
3. isEnabled()

**Syntax:**

**isDisplayed():**  
***boolean****buttonPresence = driver.findElement(By.id(“gbqfba”)).isDisplayed();*

**isSelected():**  
***boolean****buttonSelected = driver.findElement(By.id(“gbqfba”)).isSelected();*

**isEnabled():**  
***boolean****searchIconEnabled = driver.findElement(By.id(“gbqfb”)).isEnabled();*

**Q #23)** **How can we get a text of a web element?**

Get command is used to retrieve the inner text of the specified web element. The command doesn’t require any parameter but returns a string value. It is also one of the extensively used commands for verification of messages, labels, errors etc displayed on the web pages.

**Syntax:**  
*String Text = driver.findElement(By.id(“Text”)).getText();*

**Q #24) How to select value in a dropdown?**

The value in the dropdown can be selected using WebDriver’s Select class.

**Syntax:**

**selectByValue:**  
*Select selectByValue =****new****Select(driver.findElement(By.id(“SelectID\_One”)));*  
*selectByValue.selectByValue(“greenvalue”);*

**selectByVisibleText:**  
*Select selectByVisibleText =****new****Select (driver.findElement(By.id(“SelectID\_Two”)));*  
*selectByVisibleText.selectByVisibleText(“Lime”);*

**selectByIndex:**  
*Select selectByIndex =****new****Select(driver.findElement(By.id(“SelectID\_Three”)));*  
*selectByIndex.selectByIndex(2);*

**Q #25) What are the different types of navigation commands?**

Following are the [navigation commands](https://www.softwaretestinghelp.com/selenium-webdriver-waits-selenium-tutorial-15/):  
**navigate().back()** – The above command requires no parameters and takes back the user to the previous webpage in the web browser’s history.

**Sample code:**  
*driver.navigate().back();*

**navigate().forward()** – This command lets the user to navigate to the next web page with reference to the browser’s history.

**Sample code:**  
*driver.navigate().forward();*

**navigate().refresh()** – This command lets the user to refresh the current web page there by reloading all the web elements.

**Sample code:**  
*driver.navigate().refresh();*

**navigate().to()** – This command lets the user to launch a new web browser window and navigate to the specified URL.

**Sample code:**  
*driver.navigate().to(“https://google.com”);*

**Q #26) How to click on a hyper link using linkText?**

*driver.findElement(By.linkText(“Google”)).click();*

The command finds the element using link text and then click on that element and thus the user would be re-directed to the corresponding page.

The above-mentioned link can also be accessed by using the following command.

*driver.findElement(By.partialLinkText(“Goo”)).click();*

The above command finds the element based on the substring of the link provided in the parenthesis and thus partialLinkText() finds the web element with the specified substring and then clicks on it.

**Q #27)** **How to**[**handle frame in WebDriver**](https://www.softwaretestinghelp.com/selenium-tutorial-18/)**?**

An inline frame acronym as iframe is used to insert another document within the current HTML document or simply a web page into a web page by enabling nesting.

**Select iframe by id**  
*driver.switchTo().frame(“ID of the frame“);*

**Locating iframe using tagName**  
*driver.switchTo().frame(driver.findElements(By.tagName(“iframe”).get(0));*

**Locating iframe using index**

**frame(index)**  
*driver.switchTo().frame(0);*

**frame(Name of Frame)**  
*driver.switchTo().frame(“name of the frame”);*

**frame(WebElement element)**  
**Select Parent Window**  
*driver.switchTo().defaultContent();*

**Q #28) When do we use findElement() and findElements()?**

**findElement():**findElement() is used to find the first element in the current web page matching to the specified locator value. Take a note that only first matching element would be fetched.

**Syntax:**

*WebElement element = driver.findElements(By.xpath(“//div[@id='example']//ul//li”));*  
**findElements():**findElements() is used to find all the elements in the current web page matching to the specified locator value. Take a note that all the matching elements would be fetched and stored in the list of WebElements.

**Syntax:**  
*List <WebElement> elementList = driver.findElements(By.xpath(“//div[@id='example']//ul//li”));*

**Q #29)** **How to find more than one web element in the list?**

At times, we may come across elements of the same type like multiple hyperlinks, images etc arranged in an ordered or unordered list. Thus, it makes absolute sense to deal with such elements by a single piece of code and this can be done using WebElement List.

**Sample Code**

// Storing the list

List <WebElement> elementList = driver.findElements(By.xpath("//div[@id='example']//ul//li"));

// Fetching the size of the list

int listSize = elementList.size();

for (int i=0; i<listSize; i++)

{

// Clicking on each service provider link

serviceProviderLinks.get(i).click();

// Navigating back to the previous page that stores link to service providers

driver.navigate().back();

}

**Q #30) What is the difference between driver.close() and driver.quit command?**

**close()**: WebDriver’s close() method closes the web browser window that the user is currently working on or we can also say the window that is being currently accessed by the WebDriver. The command neither requires any parameter nor does it return any value.

**quit()**: Unlike close() method, quit() method closes down all the windows that the program has opened. Same as close() method, the command neither requires any parameter nor does is return any value.

**Q #31) Can Selenium handle windows based pop up?**

Selenium is an automation testing tool which supports only web application testing. Therefore, windows pop up cannot be handled using Selenium.

**Q #32) How can we handle web-based pop-up?**

WebDriver offers the users a very efficient way to [handle these pop-ups using Alert interface](https://www.softwaretestinghelp.com/handle-alerts-popups-selenium-webdriver-selenium-tutorial-16/). There are the four methods that we would be using along with the Alert interface.

* void dismiss() – The dismiss() method clicks on the “Cancel” button as soon as the pop-up window appears.
* void accept() – The accept() method clicks on the “Ok” button as soon as the pop-up window appears.
* String getText() – The getText() method returns the text displayed on the alert box.
* void sendKeys(String stringToSend) – The sendKeys() method enters the specified string pattern into the alert box.

**Syntax:**  
*// accepting javascript alert*  
*Alert alert = driver.switchTo().alert();*  
*alert.accept();*

**Q #33) How can we handle windows based pop up?**

Selenium is an automation testing tool which supports only web application testing, that means, it doesn’t support testing of windows based applications. However Selenium alone can’t help the situation but along with some third-party intervention, this problem can be overcome. There are several third-party tools available for handling window based pop-ups along with the selenium like AutoIT, Robot class etc.

**Q #34) How to assert the title of the web page?**

*//verify the title of the web page*  
*assertTrue(“The title of the window is incorrect.”,driver.getTitle().equals(“Title of the page”));*

**Q #35) How to mouse hover on a web element using WebDriver?**

WebDriver offers a wide range of interaction utilities that the user can exploit to automate mouse and keyboard events. Action Interface is one such utility which simulates the single user interactions.

Thus, In the following scenario, we have used Action Interface to mouse hover on a drop down which then opens a list of options.

**Sample Code:**

// Instantiating Action Interface

Actions actions=new Actions(driver);

// howering on the dropdown

actions.moveToElement(driver.findElement(By.id("id of the dropdown"))).perform();

// Clicking on one of the items in the list options

WebElement subLinkOption=driver.findElement(By.id("id of the sub link"));

subLinkOption.click();

**Q #36) How to retrieve CSS properties of an element?**

The values of the css properties can be retrieved using a get() method:

**Syntax:**  
*driver.findElement(By.id(“id“)).getCssValue(“name of css attribute”);*  
*driver.findElement(By.id(“id“)).getCssValue(“font-size”);*

**Q #37) How to capture screenshot in WebDriver?**

import org.junit.After;

import org.junit.Before;

import org.junit.Test;

import java.io.File;

import java.io.IOException;

import org.apache.commons.io.FileUtils;

import org.openqa.selenium.OutputType;

import org.openqa.selenium.TakesScreenshot;

import org.openqa.selenium.WebDriver;

import org.openqa.selenium.firefox.FirefoxDriver;

public class CaptureScreenshot {

WebDriver driver;

@Before

public void setUp() throws Exception {

driver = new FirefoxDriver();

driver.get("https://google.com");

}

@After

public void tearDown() throws Exception {

driver.quit();

}

@Test

public void test() throws IOException {

// Code to capture the screenshot

File scrFile = ((TakesScreenshot)driver).getScreenshotAs(OutputType.FILE);

// Code to copy the screenshot in the desired location

FileUtils.copyFile(scrFile, new File("C:\\CaptureScreenshot\\google.jpg"))

}

}

**Q #38) What is Junit?**

[Junit](https://www.softwaretestinghelp.com/selenium-junit-framework-selenium-tutorial-11/) is a unit testing framework introduced by Apache. Junit is based on Java.

**Q #39) What are Junit annotations?**

Following are the JUnit Annotations:

* **@Test:**Annotation lets the system know that the method annotated as @Test is a test method. There can be multiple test methods in a single test script.
* **@Before:**Method annotated as @Before lets the system know that this method shall be executed every time before each of the test methods.
* **@After:**Method annotated as @After lets the system know that this method shall be executed every time after each of the test method.
* **@BeforeClass:**Method annotated as @BeforeClass lets the system know that this method shall be executed once before any of the test methods.
* **@AfterClass:**Method annotated as @AfterClass lets the system know that this method shall be executed once after any of the test methods.
* **@Ignore:**Method annotated as @Ignore lets the system know that this method shall not be executed.

**Q #40)** **What is TestNG and how is it better than Junit?**

[TestNG](https://www.softwaretestinghelp.com/testng-framework-selenium-tutorial-12/) is an advanced framework designed in a way to leverage the benefits by both the developers and testers. With the commencement of the frameworks, JUnit gained enormous popularity across the Java applications, Java developers and Java testers with remarkably increasing the code quality. Despite being easy to use and straightforward, JUnit has its own limitations which give rise to the need of bringing TestNG into the picture. TestNG is an open source framework which is distributed under the Apache Software License and is readily available for download.

TestNG with WebDriver provides an efficient and effective test result format that can, in turn, be shared with the stakeholders to have a glimpse on the product’s/application’s health thereby eliminating the drawback of WebDriver’s incapability to generate test reports. TestNG has an inbuilt exception handling mechanism which lets the program to run without terminating unexpectedly.

There are various advantages that make TestNG superior to JUnit. Some of them are:

* Added advance and easy annotations
* Execution patterns can set
* Concurrent execution of test scripts
* Test case dependencies can be set

**Q #41)** **How to set test case priority in TestNG?**

**Setting Priority in TestNG**

**Code Snippet**

package TestNG;

import org.testng.annotations.\*;

public class SettingPriority {

@Test(priority=0)

public void method1() {

}

@Test(priority=1)

public void method2() {

}

@Test(priority=2)

public void method3() {

}

}

**Test Execution Sequence:**

1. Method1
2. Method2
3. Method3

**Q #42) What is a framework?**

The framework is a constructive blend of various guidelines, coding standards, concepts, processes, practices, project hierarchies, modularity, reporting mechanism, test data injections etc. to pillar automation testing.

**Q #43)** **What are the advantages of the Automation framework?**

**The advantage of**[**Test Automation framework**](https://www.softwaretestinghelp.com/test-automation-frameworks-selenium-tutorial-20/)

* Reusability of code
* Maximum coverage
* Recovery scenario
* Low-cost maintenance
* Minimal manual intervention
* Easy Reporting

**Q #44) What are the different types of frameworks?**

**Below are the different types of frameworks:**

1. **Module Based Testing Framework:** The framework divides the entire “Application Under Test” into the number of logical and isolated modules. For each module, we create a separate and independent test script. Thus, when these test scripts have taken together builds a larger test script representing more than one module.
2. **Library Architecture Testing Framework:** The basic fundamental behind the framework is to determine the common steps and group them into functions under a library and call those functions in the test scripts whenever required.
3. Data Driven Testing Framework: Data Driven Testing Framework helps the user segregate the test script logic and the test data from each other. It lets the user store the test data into an external database. The data is conventionally stored in “Key-Value” pairs. Thus, the key can be used to access and populate the data within the test scripts.
4. **Keyword Driven Testing Framework:** The Keyword Driven testing framework is an extension to Data-driven Testing Framework in a sense that it not only segregates the test data from the scripts, it also keeps the certain set of code belonging to the test script into an external data file.
5. **Hybrid Testing Framework:** Hybrid Testing Framework is a combination of more than one above mentioned frameworks. The best thing about such a setup is that it leverages the benefits of all kinds of associated frameworks.
6. **Behavior Driven Development Framework:** Behavior Driven Development framework allows automation of functional validations in an easily readable and understandable format to Business Analysts, Developers, Testers, etc.

**Q #45) How can I read test data from excels?**

Test data can efficiently be read from excel using JXL or POI API. [See detailed tutorial here](https://www.softwaretestinghelp.com/selenium-framework-design-selenium-tutorial-21/).

**Q #46) What is the difference between POI and jxl jar?**

| **#** | **JXL jar** | **POI jar** |
| --- | --- | --- |
| 1 | JXL supports “.xls” format i.e. binary based format. JXL doesn’t support Excel 2007 and “.xlsx” format i.e. XML based format | POI jar supports all of these formats |
| 2 | JXL API was last updated in the year 2009 | POI is regularly updated and released |
| 3 | The JXL documentation is not as comprehensive as that of POI | POI has a well prepared and highly comprehensive documentation |
| 4 | JXL API doesn’t support rich text formatting | POI API supports rich text formatting |
| 5 | JXL API is faster than POI API | POI API is slower than JXL API |

**Q #47)** **What is the difference between Selenium and QTP?**

| **Feature** | **Selenium** | **Quick Test Professional (QTP)** |
| --- | --- | --- |
| Browser Compatibility | Selenium supports almost all the popular browsers like Firefox, Chrome, Safari, Internet Explorer, Opera etc | QTP supports Internet Explorer, Firefox and Chrome. QTP only supports Windows Operating System |
| Distribution | Selenium is distributed as an open source tool and is freely available | QTP is distributed as a licensed tool and is commercialized |
| Application under Test | Selenium supports testing of only web based applications | QTP supports testing of both the web based application and windows based application |
| Object Repository | Object Repository needs to be created as a separate entity | QTP automatically creates and maintains Object Repository |
| Language Support | Selenium supports multiple programming languages like Java, C#, Ruby, Python, Perl etc | QTP supports only VB Script |
| Vendor Support | As Selenium is a free tool, user would not get the vendor’s support in troubleshooting issues | Users can easily get the vendor’s support in case of any issue |

**Q #48) Can WebDriver test Mobile applications?**

WebDriver cannot test Mobile applications. WebDriver is a web-based testing tool, therefore applications on the mobile browsers can be tested.

**Q #49) Can captcha be automated?**

No, captcha and barcode reader cannot be automated.

**Q #50) What is Object Repository? How can we create an Object Repository in Selenium?**

Object Repository is a term used to refer to the collection of web elements belonging to Application Under Test (AUT) along with their locator values. Thus, whenever the element is required within the script, the locator value can be populated from the Object Repository. Object Repository is used to store locators in a centralized location instead of hardcoding them within the scripts.

In Selenium, objects can be stored in an excel sheet which can be populated inside the script whenever required.

**Q #1) What is JAVA?**

**Ans:** Java is a high-level programming language and is platform independent.

Java is a collection of objects. It was developed by Sun Microsystems. There are a lot of applications, websites and Games that are developed using Java.

**Q #2) What are the features in JAVA?**

**Ans: Features of Java:**

* **Oops concepts**
  + Object-oriented
  + Inheritance
  + Encapsulation
  + Polymorphism
  + Abstraction
* **Platform independent:** A single program works on different platforms without any modification.
* **High Performance:** JIT (Just In Time compiler) enables high performance in Java. JIT converts the bytecode into machine language and then JVM starts the execution.
* **Multi-threaded:** A flow of execution is known as a Thread. JVM creates a thread which is called main thread. The user can create multiple threads by extending the thread class or by implementing Runnable interface.

**Q #3) How does Java enable high performance?**

**Ans:** Java uses Just In Time compiler to enable high performance. JIT is used to convert the instructions into bytecodes.

**Q #4) What are the Java IDE’s?**

**Ans:** Eclipse and NetBeans are the IDE's of JAVA.

**Q #5) What do you mean by Constructor?**

**Ans: The points given below explain what a Constructor is in detail:**

* When a new object is created in a program a constructor gets invoked corresponding to the class.
* The constructor is a method which has the same name as class name.
* If a user doesn’t create a constructor implicitly a default constructor will be created.
* The constructor can be overloaded.
* If the user created a constructor with a parameter then he should create another constructor explicitly without a parameter.

**Q #6) What is meant by Local variable and Instance variable?**

**Ans: Local variables** are defined in the method and scope of the variables that have existed inside the method itself.

**An instance variable** is defined inside the class and outside the method and scope of the variables exist throughout the class.

**Q #7) What is a Class?**

**Ans:** All Java codes are defined in a class. A Class has variables and methods.

**Variables**are attributes which define the state of a class.

**Methods** are the place where the exact business logic has to be done. It contains a set of statements (or) instructions to satisfy the particular requirement.

**Example:**

|  |  |  |
| --- | --- | --- |
| 1 | public class Addition{ //Class name declaration | |
| 2 | int a = 5; //Variable declaration |

|  |  |
| --- | --- |
| 3 | int b= 5; |
| 4 | public void add(){ //Method declaration | |

|  |  |  |
| --- | --- | --- |
| 5 | int c = a+b; | |
| 6 | } |

|  |  |
| --- | --- |
| 7 | } |

**Q #8) What is an Object?**

**Ans:** An instance of a class is called object. The object has state and behavior.

Whenever the JVM reads the “new()” keyword then it will create an instance of that class.

**Example:**

|  |  |
| --- | --- |
| 1 | public class Addition{ |
| 2 | public static void main(String[] args){ | |

|  |  |  |
| --- | --- | --- |
| 3 | Addion add = new Addition();//Object creation | |
| 4 | } |

|  |  |
| --- | --- |
| 5 | } |

The above code creates the object for the Addition class.

**Q #9)What are the Oops concepts?**

**Ans: Oops concepts include:**

* Inheritance
* Encapsulation
* Polymorphism
* Abstraction
* Interface

**Q #10) What is Inheritance?**

**Ans:** Inheritance means one class can **extend**to another class. So that the codes can be reused from one class to another class.

Existing class is known as Super class whereas the derived class is known as a sub class.

**Example:**

|  |  |
| --- | --- |
| 1 | Super class: |
| 2 | public class Manupulation(){ | |

|  |  |
| --- | --- |
| 3 | } |
| 4 | Sub class: | |

|  |  |  |
| --- | --- | --- |
| 5 | public class Addition extends Manipulation(){ | |
| 6 | } |

Inheritance is applicable for public and protected members only. Private members can’t be inherited.

**Q #11) What is Encapsulation?**

**Ans: Purpose of Encapsulation:**

* Protects the code from others.
* Code maintainability.

**Example:**

We are declaring ‘a' as an integer variable and it should not be negative.

|  |  |  |
| --- | --- | --- |
| 1 | public class Addition(){ | |
| 2 | int a=5; |

|  |  |
| --- | --- |
| 3 | } |

If someone changes the exact variable as “***a = -5”***then it is bad.

**In order to overcome the problem we need to follow the below steps:**

* We can make the variable as private or protected one.
* Use public accessor methods such as set<property> and get<property>.

**So that the above code can be modified as:**

|  |  |
| --- | --- |
| 1 | public class Addition(){ |
| 2 | private int a = 5; //Here the variable is marked as private | |

|  |  |
| --- | --- |
| 3 | } |

**Below code shows the getter and setter.**

Conditions can be provided while setting the variable.

get A(){

}

set A(int a){

if(a>0){// Here condition is applied

.........

}

}

For encapsulation, we need to make all the instance variables as private and create setter and getter for those variables. Which in turn will force others to call the setters rather than access the data directly.

**Q #12) What is Polymorphism?**

**Ans:** Polymorphism means many forms.

A single object can refer the super class or sub-class depending on the reference type which is called polymorphism.

**Example:**

|  |  |  |
| --- | --- | --- |
| 1 | Public class Manipulation(){ //Super class | |
| 2 | public void add(){ |

|  |  |
| --- | --- |
| 3 | } |
| 4 | } |

|  |  |  |
| --- | --- | --- |
| 5 | public class Addition extends Manipulation(){ // Sub class | |
| 6 | public void add(){ |

|  |  |
| --- | --- |
| 7 | } |
| 8 | public static void main(String args[]){ | |

|  |  |  |  |
| --- | --- | --- | --- |
| 9 | Manipulation addition = new Addition();//Manipulation is reference type and Addition is reference type | | |
| 10 | | addition.add(); |

|  |  |
| --- | --- |
| 11 | } |
| 12 | } |

Using Manipulation reference type we can call the Addition class “add()” method. This ability is known as Polymorphism.

Polymorphism is applicable for **overriding**and not for **overloading**.

**Q #13) What is meant by Method Overriding?**

**Ans: Method overriding happens if the sub class method satisfies the below conditions with the Super class method:**

* Method name should be same
* Argument should be same
* Return type also should be same

The key benefit of overriding is that the Sub class can provide some specific information about that sub class type than the super class.

**Example:**

public class Manipulation{ //Super class

public void add(){

………………

}

}

Public class Addition extends Manipulation(){

Public void add(){

………..

}

Public static void main(String args[]){

Manipulation addition = new Addition(); //Polimorphism is applied

addition.add(); // It calls the Sub class add() method

}

}

**addition.add()**method calls the add() method in the Sub class and not the parent class. So it overrides the Super class method and is known as Method Overriding.

**Q #14) What is meant by Overloading?**

**Ans:** Method overloading happens for different classes or within the same class.

**For method overloading, subclass method should satisfy the below conditions with the Super class method (or) methods in the same class itself:**

* Same method name
* Different argument type
* May have different return types

**Example:**

public class Manipulation{ //Super class

public void add(String name){ //String parameter

………………

}

}

Public class Addition extends Manipulation(){

Public void add(){//No Parameter

………..

}

Public void add(int a){ //integer parameter

}

Public static void main(String args[]){

Addition addition = new Addition();

addition.add();

}

}

Here the add() method having different parameters in the Addition class is overloaded in the same class as well as with the super class.

**Note:** Polymorphism is not applicable for method overloading.

**Q #15) What is meant by Interface?**

**Ans:** Multiple inheritance cannot be achieved in java. To overcome this problem Interface concept is introduced.

An interface is a template which has only method declarations and not the method implementation.

**Example:**

|  |  |  |
| --- | --- | --- |
| 1 | Public abstract interface IManupulation{ //Interface declaration | |
| 2 | Public abstract void add();//method declaration |

|  |  |  |
| --- | --- | --- |
| 3 | public abstract void subtract(); | |
| 4 | } |

* All the methods in the interface are internally **public abstract void**.
* All the variables in the interface are internally **public static final** that is constants.
* Classes can implement the interface and not extends.
* The class which implements the interface should provide an implementation for all the methods declared in the interface.

|  |  |  |
| --- | --- | --- |
| 1 | public class Manupulation implements IManupulation{ //Manupulation class uses the interface | |
| 2 | Public void add(){ |

|  |  |  |
| --- | --- | --- |
| 3 | …………… | |
| 4 | } |

|  |  |  |
| --- | --- | --- |
| 5 | Public void subtract(){ | |
| 6 | ……………. |

|  |  |
| --- | --- |
| 7 | } |
| 8 | } |

**Q #16) What is meant by Abstract class?**

**Ans:** We can create the Abstract class by using “Abstract” keyword before the class name. An abstract class can have both “Abstract” methods and “Non-abstract” methods that are a concrete class.

**Abstract method:**

The method which has only the declaration and not the implementation is called the abstract method and it has the keyword called “abstract”. Declarations are the ends with a semicolon.

**Example:**

|  |  |
| --- | --- |
| 1 | public abstract class Manupulation{ |
| 2 | public abstract void add();//Abstract method declaration | |

|  |  |  |
| --- | --- | --- |
| 3 | Public void subtract(){ | |
| 4 | } |

|  |  |
| --- | --- |
| 5 | } |

* An abstract class may have a Non- abstract method also.
* The concrete Subclass which extends the Abstract class should provide the implementation for abstract methods.

**Q #17) Difference between Array and Array List.**

**Ans:** **The Difference between Array and Array List can be understood from the below table:**

| **Array** | **Array List** |
| --- | --- |
| Size should be given at the time of array declaration.  String[] name = new String[2] | Size may not be required. It changes the size dynamically.  ArrayList name = new ArrayList |
| To put an object into array we need to specify the index.  name[1] = “book” | No index required.  name.add(“book”) |
| Array is not type parameterized | ArrayList in java 5.0 are parameterized.  Eg: This angle bracket is a type parameter which means a list of String. |

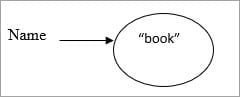
**Q #18) Difference between String, String Builder, and String Buffer.**

**Ans: String:** String variables are stored in “constant string pool”. Once the string reference changes the old value that exists in the “constant string pool”, it cannot be erased.

**Example:**

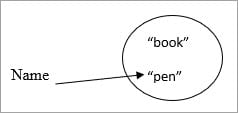
String name = “book”;

**Constant string pool**

.

If the name value has changed from “book” to “pen”.

**Constant string pool**



Then the older value retains in the constant string pool.

**String Buffer:**

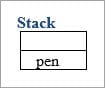
* Here string values are stored in a stack. If the values are changed then the new value replaces the older value.
* The string buffer is synchronized which is thread-safe.
* Performance is slower than the String Builder.

**Example:**

String Buffer name =”book”;



Once the name value has been changed to “pen” then the “book” is erased in the stack.



**String Builder:**

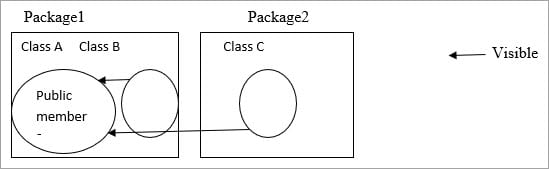
This is same as String Buffer except for the String Builder which is not threaded safety that is not synchronized. So obviously performance is fast.

**Q #19) Explain about Public and Private access specifiers.**

**Ans:** Methods and instance variables are known as members.

**Public:**

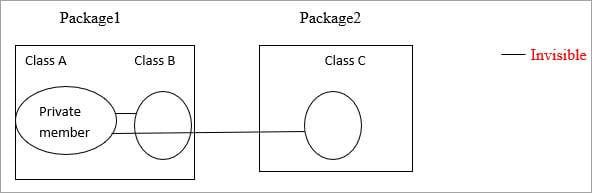
Public members are visible in the same package as well as the outside package that is for other packages.



Public members in Class A are visible to Class B (Same package) as well as Class C (Different package).

**Private:**

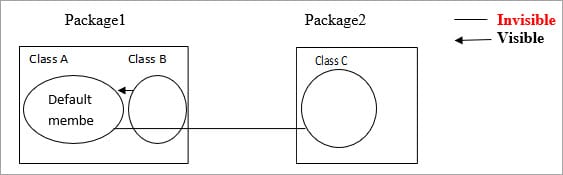
Private members are visible in the same class only and not for the other classes in the same package as well as classes in the outside packages.



Private members in class A is visible only in that class. It is invisible for class  B as well as class C.

**Q #20) Difference between Default and Protected access specifiers.**

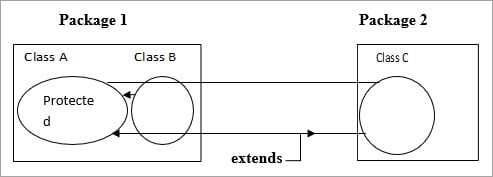
**Ans: Default:**Methods and variables declared in a class without any access specifiers are called default.



Default members in Class A are visible to the other classes which are inside the package and invisible to the classes which are outside the package.

So Class A members are visible to the Class B and invisible to the Class C.

**Protected:**

**             .**

Protected is same as Default but if a class extends then it is visible even if it is outside the package.

Class A members are visible to Class B because it is inside the package. For Class C it is invisible but if Class C extends Class A then the members are visible to the Class C even if it is outside the package.

**Q #21) Difference between HashMap and HashTable.**

**Ans:** **Difference between HashMap and HashTable can be seen below:**

| **HashMap** | **HashTable** |
| --- | --- |
| Methods are not synchronized | Key methods are synchronized |
| Not thread safety | Thread safety |
| Iterator is used to iterate the values | Enumerator is used to iterate the values |
| Allows one null key and multiple null values | Doesn’t allow anything that is null |
| Performance is high than HashTable | Performance is slow |

**Q #22) Difference between HashSet and TreeSet.**

**Ans:** **Difference between HashSet and TreeSet can be seen below:**

| **HashSet** | **TreeSet** |
| --- | --- |
| Inserted elements are in random order | Maintains the elements in the sorted order |
| Can able to store null objects | Couldn’t store null objects |
| Performance is fast | Performance is slow |

**Q #23) Difference between Abstract class and Interface.**

**Ans: Difference between Abstract Class and Interface are as follows:**

**Abstract Class:**

* Abstract classes have a default constructor and it is called whenever the concrete subclass is instantiated.
* Contains Abstract methods as well as Non-Abstract methods.
* The class which extends the Abstract class shouldn’t require implementing all the methods, only Abstract methods need to be implemented in the concrete sub-class.
* Abstract Class contains instance variables.

**Interface:**

* Doesn’t have any constructor and couldn’t be instantiated.
* Abstract method alone should be declared.
* Classes which implement the interface should provide the implementation for all the methods.
* The interface contains only constants.

**Q** **#24)  What is mean by Collections in Java?**

**Ans:** Collection is a framework that is designed to store the objects and manipulate the design to store the objects.

**Collections are used to perform the following operations:**

* Searching
* Sorting
* Manipulation
* Insertion
* Deletion

A group of objects is known as collections. All the classes and interfaces for collecting are available in Java utile package.

**Q #25) What are all the Classes and Interfaces that are available in the collections?**

**Ans:** **Given below are the Classes and Interfaces that are available in Collections:**

**Interfaces:**

* Collection
* List
* Set
* Map
* Sorted Set
* Sorted Map
* Queue

**Classes:**

* Lists:
* Array List
* Vector
* Linked List

**Sets:**

* Hash set
* Linked Hash Set
* Tree Set

**Maps:**

* Hash Map
* Hash Table
* Tree Map
* Linked Hashed Map

**Queue:**

* Priority Queue

**Q #26) What is meant by Ordered and Sorted in collections?**

**Ans:**

**Ordered:**

It means the values that are stored in a collection is based on the values that are added to the collection. So we can iterate the values from the collection in a specific order.

**Sorted:**

Sorting mechanism can be applied internally or externally so that the group of objects sorted in a particular collection is based on properties of the objects.

**Q #27) Explain about the different lists available in the collection.**

**Ans:**Values added to the list is based on the index position and it is ordered by index position. Duplicates are allowed.

**Types of Lists are:**

**Array List:**

* Fast iteration and fast Random Access.
* It is an ordered collection (by index) and not sorted.
* It implements Random Access Interface.

**Example:**

public class Fruits{

public static void main (String [ ] args){

ArrayList <String>names=new ArrayList <String>();

names.add (“apple”);

names.add (“cherry”);

names.add (“kiwi”);

names.add (“banana”);

names.add (“cherry”);

System.out.println (names);

}

}

**Output:**

[Apple, cherry, kiwi, banana, cherry]

From the output, Array List maintains the insertion order and it accepts the duplicates. But not sorted.

**Vector:**

It is same as Array List.

* Vector methods are synchronized.
* Thread safety.
* It also implements the Random Access.
* Thread safety usually causes a performance hit.

**Example:**

public class Fruit {

public static void main (String [ ] args){

Vector <String> names = new Vector <String> ( );

names.add (“cherry”);

names.add (“apple”);

names.add (“banana”);

names.add (“kiwi”);

names.add (“apple”);

System.out.println (“names”);

}

}

**Output:**

[cherry,apple,banana,kiwi,apple]

Vector also maintains the insertion order and accepts the duplicates.

**Linked List:**

* Elements are doubly linked to one another.
* Performance is slow than Array list.
* Good choice for insertion and deletion.
* In Java 5.0 it supports common queue methods peek( ), Pool ( ), Offer ( ) etc.

**Example:**

public class Fruit {

public static void main (String [ ] args){

Linkedlist <String> names = new linkedlist <String> ( ) ;

names.add(“banana”);

names.add(“cherry”);

names.add(“apple”);

names.add(“kiwi”);

names.add(“banana”);

System.out.println (names);

}

}

**Output**

[ banana,cherry,apple,kiwi,banana]

Maintains the insertion order and accepts the duplicates.

**Q #28) Explain about Set and their types in a collection?**

**Ans: Set c**ares about uniqueness. It doesn’t allow duplications. Here “equals ( )” method is used to determine whether two objects are identical or not.

**Hash Set:**

* Unordered and unsorted.
* Uses the hash code of the object to insert the values.
* Use this when the requirement is “no duplicates and don’t care about the order”.

**Example:**

public class Fruit {

public static void main (String[ ] args){

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

names.add(“banana”);

names.add(“cherry”);

names.add(“apple”);

names.add(“kiwi”);

names.add(“banana”);

System.out.println (names);

}

}

**Output:**

[banana, cherry, kiwi, apple]

Doesn’t follow any insertion order. Duplicates are not allowed.

**Linked Hash set:**

* An ordered version of the hash set is known as Linked Hash Set.
* Maintains a doubly-Linked list of all the elements.
* Use this when the iteration order is required.

**Example:**

public class Fruit {

public static void main (String[ ] args){

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

names.add(“banana”);

names.add(“cherry”);

names.add(“apple”);

names.add(“kiwi”);

names.add(“banana”);

System.out.println (names);

}

}

**Output:**

[banana, cherry, apple, kiwi]

Maintains the insertion order in which they have been added to the Set. Duplicates are not allowed.

**Tree Set:**

* It is one of the two sorted collections.
* Uses “Read-Black” tree structure and guarantees that the elements will be in an ascending order.
* We can construct a tree set with the constructor by using comparable (or) comparator.

**Example:**

public class Fruits{

public static void main (String[ ]args) {

Treeset<String> names= new TreeSet<String>( ) ;

names.add(“cherry”);

names.add(“banana”);

names.add(“apple”);

names.add(“kiwi”);

names.add(“cherry”);

System.out.println(names);

}

}

**Output:**

[apple, banana, cherry, kiwi]

TreeSet sorts the elements in an ascending order. And duplicates are not allowed.

**Q #29). Explain about Map and their types.**

**Ans: Map** cares about unique identifier. We can map a unique key to a specific value. It is a key/value pair. We can search a value, based on the key. Like set, Map also uses “equals ( )” method to determine whether two keys are same or different.

**Hash Map:**

* Unordered and unsorted map.
* Hashmap is a good choice when we don’t care about the order.
* It allows one null key and multiple null values.

**Example:**

Public class Fruit{

Public static void main(String[ ] args){

HashMap<Sting,String> names =new HashMap<String,String>( );

names.put(“key1”,“cherry”);

names.put (“key2”,“banana”);

names.put (“key3”,“apple”);

names.put (“key4”,“kiwi”);

names.put (“key1”,“cherry”);

System.out.println(names);

}

}

**Output:**

{key2 =banana, key1=cherry, key4 =kiwi, key3= apple}

Duplicate keys are not allowed in Map.

Doesn’t maintain any insertion order and is unsorted.

**Hash Table:**

* Like vector key, methods of the class are synchronized.
* Thread safety and therefore slows the performance.
* Doesn’t allow anything that is null.

**Example:**

public class Fruit{

public static void main(String[ ]args){

Hashtable<Sting,String> names =new Hashtable<String,String>( );

names.put(“key1”,“cherry”);

names.put(“key2”,“apple”);

names.put(“key3”,“banana”);

names.put(“key4”,“kiwi”);

names.put(“key2”,“orange”);

System.out.println(names);

}

}

**Output:**

{key2=apple, key1=cherry,key4=kiwi, key3=banana}

Duplicate keys are not allowed.

**Linked Hash Map:**

* Maintains insertion order.
* Slower than Hash map.
* Can expect a faster iteration.

**Example:**

public class Fruit{

public static void main(String[ ] args){

LinkedHashMap<Sting,String> names =new LinkedHashMap<String,String>( );

names.put(“key1”,“cherry”);

names.put(“key2”,“apple”);

names.put(“key3”,“banana”);

names.put(“key4”,“kiwi”);

names.put(“key2”,“orange”);

System.out.println(names);

}

}

**Output:**

{key2=apple, key1=cherry,key4=kiwi, key3=banana}

Duplicate keys are not allowed.

**TreeMap:**

* Sorted Map.
* Like Tree set, we can construct a sort order with the constructor.

**Example:**

public class Fruit{

public static void main(String[ ]args){

TreeMap<Sting,String> names =new TreeMap<String,String>( );

names.put(“key1”,“cherry”);

names.put(“key2”,“banana”);

names.put(“key3”,“apple”);

names.put(“key4”,“kiwi”);

names.put(“key2”,“orange”);

System.out.println(names);

}

}

**Output:**

{key1=cherry, key2=banana, key3 =apple, key4=kiwi}

It is sorted in ascending order based on the key. Duplicate keys are not allowed.

**Q #30) Explain the Priority Queue.**

**Ans: Queue Interface**

**Priority Queue:**Linked list class has been enhanced to implement the queue interface. Queues can be handled with a linked list. Purpose of a queue is “Priority-in, Priority-out”.

Hence elements are ordered either naturally or according to the comparator. The elements ordering represents their relative priority.

**Q #31) What is mean by Exception?**

**Ans:** An Exception is a problem that can occur during the normal flow of an execution. A method can throw an exception when something wails at runtime. If that exception couldn’t be handled, then the execution gets terminated before it completes the task.

If we handled the exception, then the normal flow gets continued. Exceptions are a subclass of java.lang.Exception.

**Example for handling Exception:**

|  |  |
| --- | --- |
| 1 | try{ |
| 2 | //Risky codes are surrounded by this block | |

|  |  |
| --- | --- |
| 3 | }catch(Exception e){ |
| 4 | //Exceptions are caught in catch block | |

|  |  |
| --- | --- |
| 5 | } |

**Q #32) What are the types of Exceptions?**

**Ans:** Two types of Exceptions are explained below in detail.

**Checked Exception:**

These exceptions are checked by the compiler at the time of compilation. Classes that extend Throwable class except Runtime exception and Error are called checked Exception.

Checked Exceptions must either declare the exception using throes keyword (or) surrounded by appropriate try/catch.

***E.g.*** ClassNotFound Exception

**Unchecked Exception:**

These exceptions are not checked during the compile time by the compiler.  The compiler doesn’t force to handle these exceptions.

**It includes:**

* Arithmetic Exception
* ArrayIndexOutOfBounds Exception

**Q #33) What are the different ways to handle exceptions?**

**Ans:** **Two different ways to handle exception are explained below:**

**#1) Using try/catch:**

A risky code is surrounded by try block. If an exception occurs, then it is caught by the catch block which is followed by the try block.

**Example:**

|  |  |
| --- | --- |
| 1 | class Manipulation{ |
| 2 | public static void main(String[] args){ | |

|  |  |  |
| --- | --- | --- |
| 3 | add(); | |
| 4 | } |

|  |  |  |
| --- | --- | --- |
| 5 | Public void add(){ | |
| 6 | try{ |

|  |  |
| --- | --- |
| 7 | addition(); |
| 8 | }catch(Exception e){ | |

|  |  |  |  |
| --- | --- | --- | --- |
| 9 | e.printStacktrace(); | | |
| 10 | | } |

|  |  |
| --- | --- |
| 11 | } |
| 12 | } |

**#2) By declaring throws keyword:**

At the end of the method, we can declare the exception using throws keyword.

**Example:**

|  |  |
| --- | --- |
| 1 | class Manipulation{ |
| 2 | public static void main(String[] args){ | |

|  |  |  |
| --- | --- | --- |
| 3 | add(); | |
| 4 | } |

|  |  |  |
| --- | --- | --- |
| 5 | public void add() throws Exception{ | |
| 6 | addition(); |

|  |  |
| --- | --- |
| 7 | } |
| 8 | } |

**Q #34) What are the Advantages of Exception handling?**

**Ans: Given below are the advantages:**

* The normal flow of the execution won’t be terminated if exception got handled
* We can identify the problem by using catch declaration

**Q #35) What are Exception handling keywords in Java?**

**Ans: Given below are the two Exception Handling Keywords:**

**try:**

When a risky code is surrounded by a try block. An exception occurring in the try block is caught by a catch block. Try can be followed either by catch (or) finally (or) both. But any one of the blocks is mandatory.

**catch:**

This is followed by try block. Exceptions are caught here.

**finally:**

This is followed either by try block (or) catch block. This block gets executed regardless of an exception. So generally clean up codes are provided here.

**Q #36) Explain about Exception Propagation.**

**Ans:** Exception is first thrown from the method which is at the top of the stack. If it doesn’t catch, then it pops up the method and moves to the previous method and so on until they are got.

This is called Exception propagation.

**Example:**

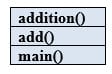
|  |  |
| --- | --- |
| 1 | public class Manipulation{ |
| 2 | public static void main(String[] args){ | |

|  |  |  |
| --- | --- | --- |
| 3 | add(); | |
| 4 | } |

|  |  |  |
| --- | --- | --- |
| 5 | public void add(){ | |
| 6 | addition(); |

|  |  |
| --- | --- |
| 7 | } |

**From the above example, the stack looks like as shown below:**

****

If an exception occurred in the **addition()** method is not caught, then it moves to the method **add()**. Then it is moved to the **main()** method and then it will stop the flow of execution. It is called Exception Propagation.

**Q #37) What is the final keyword in Java?**

**Ans:**

**Final variable:**

Once a variable is declared as final, then the value of the variable could not be changed. It is like a constant.

**Example:**

final int = 12;

**Final method:**

A final keyword in a method that couldn’t be overridden. If a method is marked as a final, then it can’t be overridden by the subclass.

**Final class:**

If a class is declared as final, then the class couldn’t be subclassed. No class can extend the final class.

**Q #38) What is a Thread?**

**Ans:**In Java, the flow of a execution is called Thread. Every java program has at least one thread called main thread, the Main thread is created by JVM. The user can define their own threads by extending Thread class (or) by implementing Runnable interface. Threads are executed concurrently.

**Example:**

|  |  |  |
| --- | --- | --- |
| 1 | public static void main(String[] args){//main thread starts here | |
| 2 | } |

**Q #39) How do you make a thread in Java?**

**Ans:**There are two ways available in order to make a thread.

**#1) Extend Thread class:**

Extending a Thread class and override the run method. The thread is available in java.lang.thread.

**Example:**

|  |  |  |
| --- | --- | --- |
| 1 | Public class Addition extends Thread { | |
| 2 | public void run () { |

|  |  |
| --- | --- |
| 3 | } |
| 4 | } |

The disadvantage of using a thread class is that we cannot extend any other classes because we have already extend the thread class. We can overload the run () method in our class.

**#2) Implement Runnable interface:**

Another way is implementing the runnable interface. For that we should provide the implementation for run () method which is defined in the interface.

**Example:**

|  |  |  |
| --- | --- | --- |
| 1 | Public class Addition implements Runnable { | |
| 2 | public void run () { |

|  |  |
| --- | --- |
| 3 | } |
| 4 | } |

**Q #40) Explain about join () method.**

**Ans:** Join () method is used to join one thread with the end of the currently running thread.

**Example:**

|  |  |  |
| --- | --- | --- |
| 1 | public static void main (String[] args){ | |
| 2 | Thread t = new Thread (); |

|  |  |  |
| --- | --- | --- |
| 3 | t.start (); | |
| 4 | t.join (); |

|  |  |
| --- | --- |
| 5 | } |

From the above code, the main thread started the execution. When it reaches the code ***t.start()*** then ‘thread t’ starts the own stack for the execution. JVM switches between the main thread and ‘thread t’.

Once it reaches the code ***t.join()*** then ‘thread t’ alone is executed and completes its task, then only main thread started the execution.

It is a non-static method. Join () method has overloaded version. So we can mention the time duration in join () method also “.s”.

**Q #41) What does yield method of the Thread class do?**

**Ans:** A yield () method moves the currently running thread to a  runnable state and allows the other threads for execution. So that equal priority threads have a chance to run. It is a static method. It doesn’t release any lock.

Yield () method moves the thread back to the Runnable state only, and not the thread to sleep (), wait () (or) block.

**Example:**

|  |  |  |
| --- | --- | --- |
| 1 | public static void main (String[] args){ | |
| 2 | Thread t = new Thread (); |

|  |  |  |
| --- | --- | --- |
| 3 | t.start (); | |
| 4 | } |

|  |  |  |
| --- | --- | --- |
| 5 | public void run(){ | |
| 6 | Thread.yield(); |

|  |  |
| --- | --- |
| 7 | } |
| 8 | } |

**Q #42) Explain about wait () method.**

**Ans: wait ()** method is used to make the thread to wait in the waiting pool. When a wait () method is executed during a thread execution then immediately the thread gives up the lock on the object and goes to the waiting pool. Wait () method tells the thread to wait for a given amount of time.

Then the thread will wake up after notify () (or) notify all () method is called.

Wait() and the other above-mentioned methods do not give the lock on the object immediately until the currently executing thread completes the synchronized code. It is mostly used in synchronization.

**Example:**

|  |  |  |
| --- | --- | --- |
| 1 | public static void main (String[] args){ | |
| 2 | Thread t = new Thread (); |

|  |  |
| --- | --- |
| 3 | t.start (); |
| 4 | Synchronized (t) { | |

|  |  |  |
| --- | --- | --- |
| 5 | Wait(); | |
| 6 | } |

|  |  |
| --- | --- |
| 7 | } |

**Q #43) Difference between notify() method and notifyAll() method in Java.**

**Ans: Given below are few differences between notify() method and notifyAll() method**

| **notify()** | **notifyAll()** |
| --- | --- |
| This method is used to send a signal to wake up a single thread in the waiting pool. | This method sends the signal to wake up all the threads in a waiting spool. |

**Q #44) How to stop a thread in java? Explain about sleep () method in a thread?**

**Ans:** We can stop a thread by using the following thread methods.

* Sleeping
* Waiting
* Blocked

**Sleep:**

Sleep () method is used to sleep the currently executing thread for the given amount of time. Once the thread is wake up it can move to the runnable state. So sleep () method is used to delay the execution for some period.

It is a static method.

**Example:**

**Thread. Sleep (2000)**

So it delays the thread to sleep 2 milliseconds. Sleep () method throws an uninterrupted exception, hence we need to surround the block with try/catch.

|  |  |  |
| --- | --- | --- |
| 1 | public class ExampleThread implements Runnable{ | |
| 2 | public static void main (String[] args){ |

|  |  |  |
| --- | --- | --- |
| 3 | Thread t = new Thread (); | |
| 4 | t.start (); |

|  |  |
| --- | --- |
| 5 | } |
| 6 | public void run(){ | |

|  |  |
| --- | --- |
| 7 | try{ |
| 8 | Thread.sleep(2000); | |

|  |  |  |  |
| --- | --- | --- | --- |
| 9 | }catch(InterruptedException e){ | | |
| 10 | | } |

|  |  |
| --- | --- |
| 11 | } |

**Q #45) When to use Runnable interface Vs Thread class in Java?**

**Ans:** If we need our class to extend some other classes other than the thread then we can go with the runnable interface because in java we can extend only one class.

If we are not going to extend any class then we can extend the thread class.

**Q #46) Difference between start() and run() method of thread class.**

**Ans:** Start() method creates new thread and the code inside the run () method is executed in the new thread. If we directly called the run() method then a new thread is not created and the currently executing thread will continue to execute the run() method.

**Q #47) What is Multi-threading?**

**Ans:** Multiple threads are executed simultaneously. Each thread starts their own stack based on the flow (or) priority of the threads.

**Example Program:**

|  |  |  |
| --- | --- | --- |
| 1 | public class MultipleThreads implements Runnable | |
| 2 | { |

|  |  |  |
| --- | --- | --- |
| 3 | public static void main (String[] args){//Main thread starts here | |
| 4 | Runnable r = new runnable (); |

|  |  |
| --- | --- |
| 5 | Thread t=new thread (); |
| 6 | t.start ();//User thread starts here | |

|  |  |  |
| --- | --- | --- |
| 7 | Addition add=new addition (); | |
| 8 | } |

|  |  |  |  |
| --- | --- | --- | --- |
| 9 | public void run(){ | | |
| 10 | | go(); |

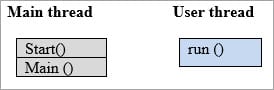
|  |  |  |
| --- | --- | --- |
| 11 | }//User thread ends here | |
| 12 | } |

On the 1st line execution, JVM calls the main method and the main thread stack looks as shown below.

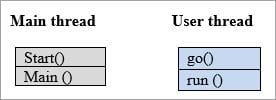


Once the execution reaches, **t.start ()**line then a new thread is created and the new stack for the thread is also created. Now JVM switches to the new thread and the main thread are back to the runnable state.

The two stacks look as shown below.



Now, the user thread executed the code inside the run() method.



Once the run() method has completed, then JVM switches back to the main thread and the User thread has completed the task and the stack was disappeared.

JVM switches between each thread until both the threads are completed. This is called Multi-threading.

**Q #48) Explain thread life cycle in Java.**

**Ans:** **Thread has the following states:**

* New
* Runnable
* Running
* Non-runnable (Blocked)
* Terminated

[](https://cdn.softwaretestinghelp.com/wp-content/qa/uploads/2018/01/Thread-Life-Cycle-in-Java.jpg)

* **New:**

In New state, Thread instance has been created but start () method is not yet invoked. Now the thread is not considered alive.

* **Runnable**:

The Thread is in runnable state after invocation of the start () method, but before the run () method is invoked. But a thread can also return to the runnable state from waiting/sleeping. In this state the thread is considered alive.

* **Running**:

The thread is in running state after it calls the run () method. Now the thread begins the execution.

* **Non-Runnable**(Blocked):

The thread is alive but it is not eligible to run. It is not in runnable state but also, it will return to runnable state after some time.

**Example:** wait, sleep, block.

* **Terminated**:

Once the run method is completed then it is terminated. Now the thread is not alive.

**Q #49) What is Synchronization?**

**Ans:** Synchronization makes only one thread to access a block of code at a time. If multiple thread accesses the block of code, then there is a chance for inaccurate results at the end. To avoid this issue, we can provide synchronization for the sensitive block of codes.

The synchronized keyword means that a thread needs a key in order to access the synchronized code.

Locks are per objects. Every Java object has a lock. A lock has only one key. A thread can access a synchronized method only if the thread can get the key to the objects lock.

For this, we use “Synchronized” keyword.

**Example:**

public class ExampleThread implements Runnable{

public static void main (String[] args){

Thread t = new Thread ();

t.start ();

}

public void run(){

synchronized(object){

{

}

}

**Q #50) What is the disadvantage of Synchronization?**

**Ans:** Synchronization is not recommended to implement all the methods. Because if one thread accesses the synchronized code then the next thread should have to wait. So it makes slow performance on the other end.

**Q #51) What is meant by Serialization?**

**Ans:** Converting a file into a byte stream is known as Serialization. The objects in the file is converted to the bytes for security purposes. For this, we need to implement java.io.Serializable interface. It has no method to define.

Variables that are marked as transient will not be a part of the serialization. So we can skip the serialization for the variables in the file by using a transient keyword.

**Q #52) What is the purpose of a transient variable?**

**Ans:** Transient variables are not part of the serialization process. During deserialization, the transient variables values are set to default value. It is not used with static variables.

**Example:**

transient int numbers;

**Q #53) Which methods are used during Serialization and Deserialization process?**

**Ans:** ObjectOutputStream and ObjectInputStream classes are higher level java.io. package. We will use them with lower level classes FileOutputStream and FileInputStream.

ObjectOutputStream.writeObject**—->**Serialize the object and write the serialized object to a file.

ObjectInputStream.readObject **—>** Reads the file and deserializes the object.

To be serialized, an object must implement the serializable interface. If superclass implements Serializable, then the subclass will automatically be serializable.

**Q #54) What is the purpose of a Volatile Variable?**

**Ans:** Volatile variable values are always read from the main memory and not from thread's cache memory. This is used mainly during synchronization. It is applicable only for variables.

**Example:**

volatile int number;

**Q #55) Difference between Serialization and Deserialization in Java.**

**Ans:** **These are the difference between serialization and deserialization in java:**

| **Serialization** | **Deserialization** |
| --- | --- |
| Serialization is the process which is used to convert the objects into byte stream | Deserialization is the opposite process of serialization where we can get the objects back from the byte stream. |
| An object is serialized by writing it an ObjectOutputStream. | An object is deserialized by reading it from an ObjectInputStream. |

**Q #56) What is SerialVersionUID?**

**Ans:** Whenever an object is Serialized, the object is stamped with a version ID number for the object class. This ID is called the  SerialVersionUID. This is used during deserialization to verify that the sender and receiver that are compatible with the Serialization.

### **1. What are the advantages and disadvantages of Selenium over other testing tools like QTP and TestComplete?**

The differences are listed below.

|  |  |  |  |
| --- | --- | --- | --- |
| ***Selenium vs HP QTP vs TestComplete*** | | | |
| **Features** | **Selenium** | **HP QTP** | **TestComplete** |
| **License** | Open Source | Required | Required |
| **Cost** | Free | High | High |
| **Customer support** | Yes; Open source community | Yes | Yes |
| **Release Cycles/ Development Sprints** | Smaller release cycles with immediate feedback | Smaller release cycles | Agility only |
| **Coding skills** | Very High | Low | High |
| **Environment support** | Windows, Linux, Mac | Only Windows | Windows only (7, Vista, Server 2008 or later OS) |
| **Language support** | Language support | VB Script | VB Script, JS Script, Delphi Script, C++ & C# |

### **2. What are the significant changes in upgrades in various Selenium versions?**

Selenium v1 included only three suite of tools: Selenium IDE, Selenium RC and Selenium Grid. Note that there was no WebDriver in Selenium v1. Selenium WebDriver was introduced in Selenium v2. With the onset of WebDriver, Selenium RC got deprecated and is not in use since. Older versions of RC is available in the market though, but support for RC is not available. Currently, Selenium v3 is in use, and it comprises of IDE, WebDriver and Grid.

IDE is used for recording and playback of tests, WebDriver is used for testing dynamic web applications via a programming interface and Grid is used for deploying tests in remote host machines.

### **3. Explain the different exceptions in Selenium WebDriver.**

Exceptions in Selenium are similar to exceptions in other programming languages. The most common exceptions in Selenium are:

* **TimeoutException:** This exception is thrown when a command performing an operation does not complete in the stipulated time
* **NoSuchElementException:** This exception is thrown when an element with given attributes is not found on the web page
* **ElementNotVisibleException:** This exception is thrown when the element is present in DOM (Document Object Model), but not visible on the web page
* **StaleElementException:** This exception is thrown when the element is either deleted or no longer attached to the DOM

### **4. What is exception test in Selenium?**

An exception test is an exception that you expect will be thrown inside a test class. If you have written a test case in such way that it should throw an exception, then you can use the **@Test**annotation and specify which exception you will be expecting by mentioning it in the parameters. Take a look at the example below: **@Test(expectedException = NoSuchElementException.class)**

Do note the syntax, where the exception is suffixed with .class

### **5. Why and how will you use an Excel Sheet in your project?**

The reason we use Excel sheets is because it can be used as data source for tests. An excel sheet can also be used to store the data set while performing DataDriven Testing. These are the two main reasons for using Excel sheets.

When you use the excel sheet as **data source**, you can store the following:

* **Application URL for all environments**: You can specify the URL of the environment in which you want to do the testing like: development environment or testing environment or QA environment or staging environment or production/ pre-production environment.
* **User name and password credentials of different environments: You can store the access credentials of the different applications/ environments in the excel sheet. You can store them in encoded format and whenever you want to use them, you can decode them instead of leaving it plain and unprotected.**
* **Test cases to be executed**: You can list down the entire set of test cases in a column and in the next column, you can specify either Yes or No which indicates if you want that particular test case to be executed or ignored.

When you use the excel sheet for **DataDriven Test**, you can store the data for different iterations to be performed in the tests. For example while testing a web page, the different sets of input data that needs to be passed to the test box can be stored in the excel sheet.

### **6. How can you redirect browsing from a browser through some proxy?**

Selenium provides a PROXY class to redirect browsing from a proxy. Look at the example below:

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9 | String PROXY = “199.201.125.147:8080”;    org.openqa.selenium.Proxy proxy = new.org.openqa.selenium.Proxy();  proxy.setHTTPProxy(Proxy)   .setFtpProxy(Proxy)   .setSslProxy(Proxy)  DesiredCapabilities cap = new DesiredCapabilities();  cap.setCapability(CapabilityType.PROXY, proxy);  WebDriver driver = new FirefoxDriver(cap); |

### **7. What is POM (Page Object Model)? What are its advantages?**

Page Object Model is a design pattern for creating an Object Repository for web UI elements. Each web page in the application is required to have it’s own corresponding page class. The page class is thus responsible for finding the WebElements in that page and then perform operations on those WebElements.

The advantages of using POM are:

* Allows us to separate operations and flows in the UI from Verification – improves code readability
* Since the Object Repository is independent of Test Cases, multiple tests can use the same Object Repository
* Reusability of code

### **8. What is Page Factory?**

Page Factory gives an optimized way to implement Page Object Model. When we say it is optimized, it refers to the fact that the memory utilization is very good and also the implementation is done in an object oriented manner.

Page Factory is used to initialize the elements of the Page Object or instantiate the Page Objects itself. Annotations for elements can also be created (and recommended) as the describing properties may not always be descriptive enough to differentiate one object from the other.

The concept of separating the Page Object Repository and Test Methods is followed here also. Instead of having to use ‘FindElements’, we use annotations like: **@FindBy** to find WebElement, and **initElements** method to initialize web elements from the Page Factory class.

**@FindBy** can accept **tagName**, **partialLinkText**, **name**, **linkText**, **id**, **css**, **className**& **xpath**as attributes.

### **9. What are the different types of WAIT statements in Selenium WebDriver? *Or the question can be framed like this:* How do you achieve synchronization in WebDriver?**

There are basically two types of wait statements: **Implicit Wait** and **Explicit Wait**.

Implicit wait instructs the WebDriver to wait for some time by polling the DOM. Once you have declared implicit wait, it will be available for the entire life of the WebDriver  instance. By default, the value will be 0. If you set a longer default, then the behavior will poll the DOM on a periodic basis depending on the browser/ driver implementation.

Explicit wait instructs the execution to wait for some time until some condition is achieved. Some of those conditions to be attained are:

* elementToBeClickable
* elementToBeSelected
* presenceOfElementLocated

### **10. Write a code to wait for a particular element to be visible on a page. Write a code to wait for an alert to appear.**

We can write a code such that we specify the XPath of the web element that needs to be visible on the page and then ask the WebDriver to wait for a specified time. Look at the sample piece of code below:

[[](https://www.edureka.co/testing-with-selenium-webdriver)](https://www.edureka.co/testing-with-selenium-webdriver" \t "_blank)

### [Selenium Certification Training](https://www.edureka.co/testing-with-selenium-webdriver" \t "_blank)

* *[Instructor-led Sessions](https://www.edureka.co/testing-with-selenium-webdriver" \t "_blank)*
* *[Real-life Case Studies](https://www.edureka.co/testing-with-selenium-webdriver" \t "_blank)*
* *[Assignments](https://www.edureka.co/testing-with-selenium-webdriver" \t "_blank)*
* *[Lifetime Access](https://www.edureka.co/testing-with-selenium-webdriver" \t "_blank)*

[Explore Curriculum](https://www.edureka.co/testing-with-selenium-webdriver" \t "_blank)

|  |  |
| --- | --- |
| 1  2 | WebDriverWait wait=new WebDriverWait(driver, 20);  Element = wait.until(ExpectedConditions.visibilityOfElementLocated(By.xpath( “<xpath”))); |

Similarly, we can write another piece of code asking the WebDriver to wait until an error appears like this:

|  |  |
| --- | --- |
| 1  2 | WebDriverWait wait=new WebDriverWait(driver, 20);  Element = wait.until(ExpectedConditions.alertIsPresent()); |

### **11. What is the use of JavaScriptExecutor?**

**JavaScriptExecutor** is an interface which provides a mechanism to execute Javascript through the Selenium WebDriver. It provides “**executescript**” and “**executeAsyncScript**” methods, to run JavaScript in the context of the currently selected frame or window. An example of that is:

|  |  |
| --- | --- |
| 1  2 | JavascriptExecutor js = (JavascriptExecutor) driver;  js.executeScript(Script,Arguments); |

### **12. How to scroll down a page using JavaScript in Selenium?**

We can scroll down a page by using window.scrollBy() function. Example:

|  |  |
| --- | --- |
| 1 | ((JavascriptExecutor) driver).executeScript("window.scrollBy(0,500)"); |

### **13. How to scroll down to a particular element?**

To scroll down to a particular element on a web page, we can use the function **scrollIntoView()**. Example:

|  |  |
| --- | --- |
| 1 | ((JavascriptExecutor) driver).executeScript("arguments[0].scrollIntoView();", element); |

### **14. How to handle keyboard and mouse actions using Selenium?**

We can handle special keyboard and mouse events by using **Advanced User Interactions API**. The Advanced User Interactions API contains the Actions and the Action Classes that are needed for executing these events. Most commonly used keyboard and mouse events provided by the Actions class are in the table below:

|  |  |
| --- | --- |
| ***Selenium functions and their explanation*** | |
| **Method** | **Description** |
| **clickAndHold()** | Clicks (without releasing) the current mouse location. |
| **dragAndDrop()** | Performs click-and-hold at the location of the source element, moves. |
| **source, target()** | Moves to the location of the target element, then releases the mouse. |

### **15. What are different types of frameworks?**

The different types of frameworks are:

* Data Driven Framework:-  
  When the entire test data is generated from some external files like Excel, CSV, XML or some database table, then it is called Data Driven framework.
* Keyword Driven Framework:-  
  When only the instructions and operations are written in a different file like an Excel worksheet, it is called Keyword Driven framework.
* Hybrid Framework:-  
  A combination of both the Data Driven framework and the Keyword Driven framework is called Hybrid framework.

### **16. Which files can be used as data source for different frameworks?**

Some of the file types of the dataset can be: excel, xml, text, csv, etc.

### **17. How can you fetch an attribute from an element? How to retrieve typed text from a textbox?**

We can fetch the attribute of an element by using the **getAttribute()** method. Sample code:

|  |  |
| --- | --- |
| 1  2 | WebElement eLogin = driver.findElement(By.name(“Login”);  String LoginClassName = eLogin.getAttribute("classname"); |

Here, I am finding the web page’s login button named ‘Login’. Once that element is found, getAttribute() can be used to retrieve any attribute value of that element and it can be stored it in string format. In my example, I have retrieved ‘classname’ attribute and stored it in LoginClassName.

Similarly, to retrieve some text from any textbox, we can use getText() method. In the below piece of code I have retrieved the text typed in the ‘Login’ element.

|  |  |
| --- | --- |
| 1  2 | WebElement eLogin = driver.findElement(By.name(“Login”);  String LoginText = Login.getText (); |

In the below Selenium WebDriver tutorial, there is a detailed demonstration of locating elements on the web page using different element locator techniques and the basic methods/ functions that can be applied on those elements. 

#### ****Selenium WebDriver Tutorial For Beginners | Edureka****

## **B. Advanced Level – Selenium Interview Question**

From here on, we’ll be looking at the most important Java interview questions for Selenium testers.

### **18. How to send ALT/SHIFT/CONTROL key in Selenium WebDriver?**

When we generally use ALT/SHIFT/CONTROL keys, we hold onto those keys and click other buttons to achieve the special functionality. So it is not enough just to specify **keys.ALT** or **keys.SHIFT** or **keys.CONTROL** functions.

For the purpose of holding onto these keys while subsequent keys are pressed, we need to define two more methods: **keyDown(modifier\_key)** and **keyUp(modifier\_key)**

Parameters: **Modifier\_key (keys.ALT or Keys.SHIFT or Keys.CONTROL)**Purpose: Performs a modifier key press and does not release the modifier key. Subsequent interactions may assume it’s kept pressed.

Parameters: **Modifier\_key (keys.ALT or Keys.SHIFT or Keys.CONTROL)**  
Purpose: Performs a key release.  
Hence with a combination of these two methods, we can capture the special function of a particular key.

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20 | public static void main(String[] args)  {  String baseUrl = “https://www.facebook.com”;  WebDriver driver = new FirefoxDriver();    driver.get("baseUrl");  WebElement txtUserName = driver.findElement(By.id(“Email”);    Actions builder = new Actions(driver);  Action seriesOfActions = builder   .moveToElement(txtUerName)   .click()   .keyDown(txtUserName, Keys.SHIFT)   .sendKeys(txtUserName, “hello”)   .keyUp(txtUserName, Keys.SHIFT)   .doubleClick(txtUserName);   .contextClick();   .build();  seriesOfActions.perform();  } |

### **19. How to take screenshots in Selenium WebDriver?**

You can take a screenshot by using the **TakeScreenshot** function. By using **getScreenshotAs()**method you can save that screenshot. Example:

|  |  |
| --- | --- |
| 1 | File scrFile = ((TakeScreenshot)driver).getScreenshotAs(outputType.FILE); |

### **20. How to set the size of browser window using Selenium?**

To maximize the size of browser window, you can use the following piece of code:  
driver.manage().window().maximize(); – To maximize the window

To resize the current window to a particular dimension, you can use the **setSize()** method. Check out the below piece of code:

|  |  |
| --- | --- |
| 1  2  3 | System.out.println(driver.manage().window().getSize());  Dimension d = new Dimension(420,600);  driver.manage().window().setSize(d); |

To set the window to a particular size, use **window.resizeTo()** method. Check the below piece of code:

|  |  |
| --- | --- |
| 1 | ((JavascriptExecutor)driver).executeScript("window.resizeTo(1024, 768);"); |

To witness a demonstration on setting custom sizes for the browser window and finding various elements on the web page, see the video below.

#### ****Selenium Tutorial For Beginners | Edureka****

### **21. How to handle a dropdown in Selenium WebDriver? How to select a value from dropdown?**

Questions on dropdown and selecting a value from that dropdown are very common Selenium interview questions because of the technicality involved in writing the code.

The most important detail you should know is that to work with a dropdown in Selenium, we must always make use of this html tag: **‘select’**. Without using ‘select’, we cannot handle dropdowns. Look at the snippet below in which I have written a code for a creating a dropdown with three options.

|  |  |
| --- | --- |
| 1  2  3  4  5 | <select id="mySelect">  <option value="option1">Cars</option>  <option value="option2">Bikes</option>  <option value="option3">Trains</option>  </select> |

In this code we use **‘select’**tag to define a dropdown element and the **id**of the dropdown element is ‘myselect’. We have 3 options in the dropdown: Cars, Bikes and Trains. Each of these options, have a ‘value’ attribute also assigned to them. First option from dropdown has value assigned as ‘option1’, second option has value = ‘option2’ and similarly third option has value assigned as ‘option3’.

If you are clear with the concept so far, then you can proceed to the next aspect of choosing a value from the dropdown. This is a 2 step process:

1. Identify the ‘select’ html element (Because dropdowns must have the ‘select’ tag)
2. Select an option from that dropdown element

To identify the ‘select’ html element from the web page, we need to use findElement() method. Look at the below piece of code:

|  |  |
| --- | --- |
| 1  2 | WebElement mySelectElement = driver.findElement(By.id("mySelect"));  Select dropdown = new Select(mySelectElement); |

Now to select an option from that dropdown, we can do it in either of the three ways:

1. dropdown.selectByVisibleText(“Bikes”); → Selecting an option by the text that is visible
2. dropdown.selectByIndex(“1”); → Selecting, by choosing the Index number of that option
3. dropdown.selectByValue(“option2”); → Selecting, by choosing the value of that option

Note that from the above example, in all the three cases, “Bikes” will be chosen from the dropdown. In the first case, we are choosing by visible text on the web page. When it comes to selection by index, 1 represents “Bikes” because indexing values start from 0 and then get incremented to 1 and 2. Finally in case of selection by value attribute, ‘option2’ refers to “Bikes”. So, these are the different ways to choose a value from a dropdown.

### **22. How to switch to a new window (new tab) which opens up after you click on a link?**

If you click on a link in a web page, then for changing the WebDriver’s focus/ reference to the new window we need to use the **switchTo()** command. Look at the below example to switch to a new window:  
**driver.switchTo().window();**

Here, ‘windowName’ is the name of the window you want to switch your reference to.

In case you do not know the name of the window, then you can use the **driver.getWindowHandle()**command to get the name of all the windows that were initiated by the WebDriver. Note that it will not return the window names of browser windows which are not initiated by your WebDriver.

Once you have the name of the window, then you can use an enhanced for loop to switch to that window. Look at the piece of code below.

|  |  |
| --- | --- |
| 1  2  3  4  5 | String handle= driver.getWindowHandle();  for (String handle : driver.getWindowHandles())  {  driver.switchTo().window(handle);  } |

### **23. How do you upload a file using Selenium WebDriver?**

To upload a file we can simply use the command **element.send\_keys(file path).**But there is a prerequisite before we upload the file. We have to use the html tag: **‘input’**and attribute type should be **‘file’**. Take a look at the below example where we are identifying the web element first and then uploading the file.

|  |  |
| --- | --- |
| 1  2  3 | <input type="file" name="uploaded\_file" size="50" class="pole\_plik">  element = driver.find\_element\_by\_id(”uploaded\_file")  element.send\_keys("C:myfile.txt") |

### **24. Can we enter text without using sendKeys()?**

Yes. We can enter/ send text without using **sendKeys()** method. We can do it using JavaScriptExecutor.

How do we do it?  
Using DOM method of, identification of an element, we can go to that particular document and then get the element by its ID (here login) and then send the text by value. Look at the sample code below:

|  |  |
| --- | --- |
| 1  2 | JavascriptExecutor jse = (JavascriptExecutor) driver;  jse.executeScript("document.getElementById(‘Login').value=Test text without sendkeys"); |

### **25. Explain how you will login into any site if it is showing any authentication popup for username and password?**

Since there will be popup for logging in, we need to use the explicit command and verify if the alert is actually present. Only if the alert is present, we need to pass the username and password credentials. The sample code for using the explicit wait command and verifying the alert is below:

|  |  |
| --- | --- |
| 1  2  3 | WebDriverWait wait = new WebDriverWait(driver, 10);  Alert alert = wait.until(ExpectedConditions.alertIsPresent());  alert.authenticateUsing(new UserAndPassword(\*\*username\*\*, \*\*password\*\*)); |

### **26. Explain how can you find broken links in a page using Selenium WebDriver?**

This is a trick question which the interviewer will present to you. He can provide a situation where in there are 20 links in a web page, and we have to verify which of those 20 links are working and how many are not working (broken).

Since you need to verify the working of every link, the workaround is that, you need to send http requests to all of the links on the web page and analyze the response. Whenever you use driver.get() method to navigate to a URL, it will respond with a status of **200 – OK**. 200 – OK denotes that the link is working and it has been obtained. If any other status is obtained, then it is an indication that the link is broken.

But how will you do that?  
First, we have to use the anchor tags <a> to determine the different hyperlinks on the web page. For each <a> tag, we can use the attribute ‘href’ value to obtain the hyperlinks and then analyze the response received for each hyperlink when used in **driver.get()** method.

### **27. Which technique should you consider using throughout the script “if there is neither frame id nor frame name”?**

If neither frame name nor frame id is available, then we can use **frame by index**.

Let’s say, that there are 3 frames in a web page and if none of them have frame name and frame id, then we can still select those frames by using frame (zero-based) index attribute. Each frame will have an index number. The first frame would be at index “0”, the second at index “1” and the third at index “2”. Once the frame has been selected, all subsequent calls on the WebDriver interface willbe made to that frame.

|  |  |
| --- | --- |
| 1 | driver.switchTo().frame(int arg0); |

## **C. TestNG Framework For Selenium – Selenium Interview Questions**

From here on, you’ll read all the TestNG and Selenium Webdriver interview questions for experienced professionals.

### **28. What is the significance of testng.xml?**

I’m pretty sure you all know the importance of TestNG. Since Selenium does not support report generation and test case management, we use TestNG framework with Selenium. TestNG is much more advanced than JUnit, and it makes implementing annotations easy. That is the reason TestNG framewrok is used with Selenium WebDriver.

But have you wondered where to define the test suites and grouping of test classes in TestNG?

It is by taking instructions from the testng.xml file. We cannot define a test suite in testing source code, instead it is represented in an XML file, because suite is the feature of execution. The test suite, that I am talking about is basically a collection of test cases.

So for executing the test cases in a suite, i.e a group of test cases, you have to create a testng.xml file which contains the name of all the classes and methods that you want to execute as a part of that execution flow.

Other advantages of using testng.xml file are:

* It allows execution of multiple test cases from multiple classes
* It allows parallel execution
* It allows execution of test cases in groups, where a single test can belong to multiple groups

### **29. What is parameterization in TestNG? How to pass parameters using testng.xml?**

Parameterization is the technique of defining values in testng.xml file and sending them as parameters to the test class. This technique is especially useful when we need to pass multiple login credentials of various test environments. Take a look at the code below, in which “myName” is annotated as a parameter.

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7 | public class ParameterizedTest1{   @Test   @Parameters("myName")   public void parameterTest(String myName) {   System.out.println("Parameterized value is : " + myName);   }  } |

To pass parameters using testng.xml file, we need to use ‘parameters’ tag. Look at the below code for example:

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10 | <?xml version="1.0" encoding="UTF-8"?>  <!DOCTYPE suite SYSTEM "<http://testng.org/testng-1.0.dtd>" >   <suite name=”CustomSuite">    <test name=”CustomTest”>     <parameter name="myName" value=”John"/>      <classes>       <class name="ParameterizedTest1" />      </classes>    </test>   </suite> |

To extensively understand the working of TestNG and it’s benefit when used with Selenium, watch the below Selenium tutorial video.

#### ****Selenium Training | TestNG Framework For Selenium | Edureka****

### **30. Explain DataProviders in TestNG using an example. Can I call a single data provider method for multiple functions and classes?**

DataProvider is a TestNG feature, which enables us to write DataDriven tests. When we say, it supports DataDriven testing, then it becomes obvious that the same test method can run multiple times with different data-sets. DataProvider is in fact another way of passing parameters to the test method.

**@DataProvider** marks a method as supplying data for a test method. The annotated method must return an Object[] where each Object[] can be assigned to parameter list of the test method.

To use the DataProvider feature in your tests, you have to declare a method annotated by **@DataProvider** and then use the said method in the test method using the ‘dataProvider‘ attribute in the Test annotation.

As far as the second part of the question is concerned, Yes, the same DataProvider can be used in multiple functions and classes by declaring DataProvider in separate class and then reusing it in multiple classes.

### **31. How to skip a method or a code block in TestNG?**

If you want to skip a particular test method, then you can set the ‘enabled’ parameter in test annotation to false.  
@Test(enabled = false)

By default, the value of ‘enabled’ parameter will be true. Hence it is not necessary to define the annotation as true while defining it.

[[](https://www.edureka.co/testing-with-selenium-webdriver)](https://www.edureka.co/testing-with-selenium-webdriver" \t "_blank)

### [Selenium Certification Training](https://www.edureka.co/testing-with-selenium-webdriver" \t "_blank)

[Weekday / Weekend BatchesSee Batch Details](https://www.edureka.co/testing-with-selenium-webdriver" \t "_blank)

### **32. What is soft assertion in Selenium? How can you mark a test case as failed by using soft assertion?**

Soft Assertions are customized error handlers provided by TestNG. Soft Assertions do not throw exceptions when assertion fails, and they simply continue to the next test step. They are commonly used when we want to perform multiple assertions.

To mark a test as failed with soft assertions, call **assertAll()** method at the end of the test.

### **33. Explain what is Group Test in TestNG?**

In TestNG, methods can be categorized into groups. When a particular group is being executed, all the methods in that group will be executed.  We can execute a group by parameterizing it’s name in group attribute of **@Test** annotation. Example: @Test(groups={“xxx”})

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14 | @Test(groups={“Car”})  public void drive(){  system.out.println(“Driving the vehicle”);  }    @Test(groups={“Car”})  public void changeGear() {  system.out.println("Change Gears”);  }    @Test(groups={“Car”})  public void accelerate(){  system.out.println(“Accelerating”);  } |

### **34. How does TestNG allow you to state dependencies? Explain it with an example.**

**Dependency**is a feature in TestNG that allows a test method to depend on a single or a group of test methods. Method dependency only works if the “depend-on-method” is part of the same class or any of the inherited base classes (i.e. while extending a class). Syntax:  
**@Test(dependsOnMethods = { “initEnvironmentTest” })**

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14 | @Test(groups={“Car”})  public void drive(){  system.out.println(“Driving the vehicle”);  }    @Test(dependsOnMethods={“drive”},groups={cars})  public void changeGear() {  system.out.println("Change Gears”);  }    @Test(dependsOnMethods={“changeGear”},groups={“Car”})  public void accelerate(){  system.out.println(“Accelerating”);  } |

### **35. Explain what does @Test(invocationCount=?) and @Test(threadPoolSize=?) indicate.**

**@Test(invocationCount=?)** is a parameter that indicates the number of times this method should be invoked.  
**@Test(threadPoolSize=?)** is used for executing suites in parallel. Each suite can be run in a separate thread.

To specify how many times @Test method should be invoked from different threads, you can use the attribute **threadPoolSize** along with **invocationCount**. Example:

|  |  |
| --- | --- |
| 1  2  3 | @Test(threadPoolSize = 3, invocationCount = 10)  public void testServer() {  } |