Interview Questions – Part II

Softech Solutions Inc.

www.softechsolutionsgroup.com

saney.alam@softechsolutionsgroup.com

Interview Questions – Part II

Q1: Difference between == and .equals() ?

Ans. "equals" is the method of object class which is supposed to be overridden to check object equality, whereas "==" operator evaluate to see if the object handlers on the left and right are pointing to the same object in memory.

x.equals(y) means the references x and y are holding objects that are equal. x==y means that the references x and y have same object.

Sample code:

String x = new String("str");

String y = new String("str");

System.out.println(x == y); // prints false

System.out.println(x.equals(y)); // prints true

Q2: Why is String immutable in Java?

Ans. 1. String Pool - When a string is created and if it exists in the pool, the reference of the existing string will be returned instead of creating a new object. If string is not immutable, changing the string with one reference will lead to the wrong value for the other references.

Example -

String str1 = "String1";

String str2 = "String1"; // It doesn't create a new String and rather reuses the string literal from pool

// Now both str1 and str2 pointing to same string object in pool, changing str1 will change it for str2 too

2. To Cache its Hashcode - If string is not immutable, One can change its hashcode and hence it's not fit to be cached.

3. Security - String is widely used as parameter for many java classes, e.g. network connection, opening files, etc. Making it mutable might possess threats due to interception by the other code segment.

Q3: Explain the scenarios to choose between String, StringBuilder and StringBuffer ?

Or

What is the difference between String, StringBuilder and StringBuffer ?

Ans. If the Object value will not change, use String Class because a String object is immutable.

If the Object value can change and will only be modified from a single thread, use StringBuilder because StringBuilder is unsynchronized (means faster).

If the Object value may change, and can be modified by multiple threads, use a StringBuffer because StringBuffer is thread safe (synchronized).

Q4: What are the difference between composition and inheritance in Java?

Ans. Composition - has-a relationship between classes.

Inheritance - is-a relationship between classes.

Composition - Composing object holds a reference to composing classes and hence relationship is loosely bound.

Inheritance - Derived object carries the base class definition in itself and hence its tightly bound.

Composition - Used in Dependency Injection

Inheritance - Used in Runtime Polymorphism

Composition - Single class objects can be composed within multiple classes.

Inheritance - Single class can only inherit 1 Class.

Composition - It's the relationship between objects.

Inheritance - It's the relationship between classes.

Q5: Which are the different segments of memory?

Ans. 1. Stack Segment - Contains primitives, Class / Interface names and references.

2. Heap Segment - Contains all created objects in runtime, objects only plus their object attributes (instance variables), Static variables are also stored in heap.

3. Code Segment - The segment where the actual compiled Java bytecodes resides when loaded.

Q6: Why do we need Inner classes? Can't we just work with outer classes wherever we implement Inner classes?

Ans. Yes, we can substitute outer classes wherever we need to have inner classes but Inner classes have advantage in certain cases and hence preferred -

Ease - Why to implement a class outside if its objects are only intended to be part of an outer object. It's easy to define the class within another class if the use is only local.

Protection - Making a call an outer exposes a threat of it being used by any of the class. Why should it be made an outer class if its object should only occur as part of other objects?

For example - You may like to have an class address whose object should have a reference to city and by design that's the only use of city you have in your application. Making Address and City as outer class exposes City to any of the Class. Making it an inner class of Address will make sure that its accessed using object of Address.

Q7: What are different ways of object creation in Java?

Ans. Using new operator - new xyzClass()

Using factory methods - xyzFactory.getInstance( )

Using newInstance( ) method - (Class.forName(xyzClass))emp.newInstance( )

By cloning an already available object - (xyzClass)obj1.clone( )

Q8: What are points to consider in terms of access modifier when we are overriding any method?

Ans. 1. Overriding method cannot be more restrictive than the overridden method.

Reason: in case of polymorphism, at object creation jvm look for actual runtime object. Jvm does not look for reference type and while calling methods it look for overridden method.

If by means subclass were allowed to change the access modifier on the overriding method, then suddenly at runtime when the JVM invokes the true objects version of the method rather than the reference type's version then it will be problematic.

2. In case of subclass and superclass define in different package; we can override only those method which have public or protected access.

3. We cannot override any private method because private methods cannot be inherited and if method cannot be inherited then method cannot be overridden.

Q9: What are different ways to create String Object? Explain.

Ans.

String str = new String ("abc”);
String str1 = "abc”;

When we create a String using double quotes, JVM looks in the String pool to find if any other String is stored with same value. If found, it just returns the reference to that String object else it creates a new String object with given value and stores it in the String pool.

When we use new operator, JVM creates the String object but don’t store it into the String Pool. We can use intern() method to store the String object into String pool or return the reference if there is already a String with equal value present in the pool.

Q10: What do you mean by "Java is a statically typed language”?

Ans. It means that the type of variables is checked at compile time in Java. The main advantage here is that all kinds of checking can be done by the compiler and hence will reduce bugs.

Q11: What is the difference between declaration, instantiation and initialization?

Ans. Declaration is intimation to the compiler about the nature of Data a reference is going to hold.

For example - List myList;

Instantiation is reservation of memory.

For example

myList = new ArrayList();

Initialization or construction is setting the default values for member elements.

For example

myList = new ArrayList(mySet);

** Example 2nd is both for instantiation as well as initialization. The only difference is that 2nd will initialize the member elements to their default values whereas 3rd will initialized it with the elements from set.

Q12: What are the common uses of "this" keyword in java ?

Ans. "this" keyword is a reference to the current object and can be used for following -

1. Passing itself to another method.

2. Referring to the instance variable when local variable has the same name.

3. Calling another constructor in constructor chaining.

Q13: Does garbage collection guarantee that a program will not run out of memory?

Ans. Garbage collection does not guarantee that a program will not run out of memory. It is possible for programs to use up memory resources faster than they are garbage collected. It is also possible for programs to create objects that are not subject to garbage collection.

Q14: What are the benefits of JSON over XML ?

Ans. Lighter and faster than XML as on-the-wire data format

Object Representation - Information is presented in object notations and hence better understandable.

Easy to parse and conversion to objects for information consumption.

Support multiple data types - JSON supports string, number, array, Boolean whereas XML data are all string.

Q15: What is Lazy Initialization in Hibernate ?

Ans. It's a feature to lazily initialize dependencies, relationship and associations from the Database. Any related references marked as @OneToMany or @ManyToMany are loaded lazily i.e. when they are accessed and not when the parent is loaded.

Q16: What is a Lambda Expression? What's its use?

Ans. It’s an anonymous method without any declaration.

Lambda Expression is useful to write shorthand Code and hence saves the effort of writing lengthy Code.

It promotes Developer productivity, Better Readable and Reliable code.

Q17: Which of the following combination of keywords is illegal in Java?

A. static and transient

B. transient and final

C. static and synchronized

D. abstract and final

Ans. abstract and final

Q18: What are different types of classes?

Ans. There are different verticals in which Java Classes can be classified.

1. Access - Public , Private , default or Protected.

2. Packaging - System, library or User Defined

3. Structure - Outer or Inner

4. Object Derivation - Abstract Class or Concrete Class.

5. Object Creation - Normal, Singleton , Doubleton , Immutable or Enum.

6. Functionality - String , Util , Stream etc.

Q19: How can we run a java program without making any object?

Ans. By putting code within static method. With Java 6 and earlier versions, even static block can be used.

Q20: Can we use both "this()" and "super()" in a constructor ?

Ans. No, because both this and super should be the first statement.

Q21: What is the difference between final, finally and finalize() ?

Ans. final - constant variable, objects cannot be de-referenced, restricting method overriding, restricting class sub classing.

finally - handles exception. The finally block is optional and provides a mechanism to clean up regardless of what happens within the try block. Use the finally block to close files or to release other system resources like database connections, statements etc.

finalize() - method helps in garbage collection. A method that is invoked before an object is discarded by the garbage collector, allowing it to clean up its state.

Q22: What are the ways to avoid LazyInitializationException ?

Ans. 1. Set lazy=false in the hibernate config file.

2. Set @Basic(fetch=FetchType.EAGER) at the mapping.

3. Make sure that we are accessing the dependent objects before closing the session.

4. Force initialization using Hibernate.initialize

5. Using Fetch Join in HQL.

Q23: Why every object constructor automatically call super() in Object before its own constructors?

Ans. Derived object carries the body of its class as well as the body of the parent class. Its body (member elements) is initialized using its own class constructor whereas the body (member elements) carried from the parent class are initialized using super class constructor. So In order to initialize the elements of the parent class before its own elements are even initialized, super is called.

Q24: Describe what happens when an object is created in Java?

Ans. 1. Memory is allocated from heap to hold all instance variables and implementation-specific data of the object and its superclass's. Implementation-specific data includes pointers to class and method data.

2. The instance variables of the objects are initialized to their default values.

3. The constructor for the most derived class is invoked. The first thing a constructor does is call the constructor for its super classes. This process continues until the constructor for java.lang.Object is called, as java.lang.Object is the base class for all objects in java.

4. Before the body of the constructor is executed, all instance variable initializers and initialization blocks are executed. Then the body of the constructor is executed. Thus, the constructor for the base class completes first and constructor for the most derived class completes last.

Q25: Difference between TreeMap and HashMap?

Ans. They are different the way their elements are stored in memory. TreeMap stores the Keys in order whereas HashMap stores the key value pairs randomly.

Q26: Difference between HashMap and Hashtable?

Ans. Hashtable is synchronized whereas HashMap is not. HashMap allows null values whereas Hashtable doesn't allow null values.

Q27: What will this code print?

```
String a = new String ("TEST");
String b = new String ("TEST");
if(a == b) {
  System.out.println ("TRUE");
} else {
  System.out.println ("FALSE");
}
```

Ans. FALSE. == operator compares object references, a and b are references to two different objects, hence the FALSE. .equals method is used to compare string object content.

Q28: Describe, in general, how java's garbage collector works?

Ans. The Java runtime environment deletes objects when it determines that they are no longer being used. This process is known as garbage collection. The Java runtime environment supports a garbage collector that periodically frees the memory used by objects that are no longer needed. The Java garbage collector is a mark-sweep garbage collector that scans Java dynamic memory areas for objects, marking those that are referenced. After all possible paths to objects are investigated, those objects that are not marked (i.e. are not referenced) are known to be garbage and are collected.

Q29: What are the methods of Object Class?

Ans. clone() - Creates and returns a copy of this object.

equals() - Indicates whether some other object is "equal to" this one.

finalize() - Called by the garbage collector on an object when garbage collection determines that there are no more references to the object.

getClass() - Returns the runtime class of an object.

hashCode() - Returns a hash code value for the object.

toString() - Returns a string representation of the object.

notify(), notifyAll(), and wait() - Play a part in synchronizing the activities of independently running threads in a program.

Q30: Why Char array is preferred over String for storing password?

Ans. String is immutable in java and stored in String pool. Once it's created it stays in the pool until unless garbage collected, so even though we are done with password it's available in memory for longer duration and there is no way to avoid it. It's a security risk because anyone having access to memory dump can find the password as clear text.