Core Interview Questions

**1. What is Abstraction?**

Hiding internal implementation and sharing set of services is called as abstarction.

1. We can achieve abstraction by using “Interface” and “Abstract Class”.

2. We can achieve 100% abstraction using “Interface” and partial by using “Abstract Class”.

3. We can achieve security using abstraction.

4.E.g. ATM machine, Car, Mobile, etc.

**2.What is Encapsulation?**

1. Wrapping of data member and methods called as Encapsulation.

2.We can achieve it by making data members “private”.

3.POJO class is good example of encapsulation.

4. In a class if it has every data member as a “private” then such class is called as tightly encapsulated.

5. E.g. Engine, Gear box within Car, etc.

**3. Difference between Abstraction and Encapsulation?**

**Abstraction Encapsulation**

1.Hiding internal implementation and sharing set. Wrapping of data member and methods.

2.We can achieve abstraction by using “Interface” We can achieve it by making data members and “Abstract Class”. “private”.

3.It increases code. It decreases code.

4. It solves problem at Design level. It solves problem at implementation level.

**4.What is Inheritance?**

1.Acquiring properties of Parent class is called inheritance.

2. It is also called as IS-A relationship.

3. It can be achieved by using “extends” keyword, by making Parent-Child relationship.

4. It helps in reusability of code.

5. E.g. New Car version inherits properties from old versions

**5. What is Polymorphism?**

1. It means one name many forms.

2.It makes reusability simple and also makes code understanding easy.

3.There are two types of Polymorphism : o Run-time Polymorphism (Dynamic Binding, Overriding)

Decision making at Runtime by using runtime object.

Used for adding additional functionality into existing one.

Useful only in Parent-Child Relationship. o Compile-time Polymorphism (Static Binding, Overloading)

4. Best example of polymorphism is “println” method of “printstream” class.

**6. What is Static?**

1.Static is keyword.

2.We can apply static keyword with variables, methods, blocks and classes.

3. The static variable gets memory only once in the class area at the time of class loading. It can be used to refer to the common properties of all objects, for example, the company name of employees, college name of students, etc.

4.A static method belongs to the class rather than the object of a class. It can be invoked without the need for creating an instance of a class. It can access static data member and can change the value of it. It can be used to setup database connection.

5.Static block is used to initialize the static data member. It is executed before the main method at the time of class loading.

**7.What is non-static block?**

1. It is used for non-initializing content
2. Before calling constructor non-static block is executed

**8.What are methods in Exception class?**

1.public String getMessage() Returns a detailed message about the exception that has occurred.

2. public Throwable getCause() Returns the cause of the exception.

3.public String toString() Returns the name of the class concatenated with the result of getMessage().

4. public void printStackTrace() Prints the result of toString() along with the stack trace, the error output stream.

5.public StackTraceElement [] getStackTrace() Returns an array containing each element on the stack trace.

6. public Throwable fillInStackTrace() Fills the stack trace of this Throwable object with the current stack trace, adding to any previous information in the stack trace

**9. What is final, finally, finalized?**

1.The final keyword in java is used to restrict the user. The java final keyword can be used in many context. Final can be used for variables, methods, class. o Final variable once assigned can’t be changed after. o Final method can’t be rewritten, can’t be inherited. o Final class can’t be accessed by creating child of it.

2. The finally is a block that always be executed either there is exception occur inside “try” block or there is no exception occur inside “try” block. In both situation, “finally” block code will be executed.

3. The finalize() is called by the garbage collector on an object when garbage collection determines that there are no more references to the object, it is used to perform cleanup activity.

**10.What is throws keyword?**

1. By using “throws” keyword we can give a chance to caller method to handle the exception.

2. The “throws” keyword is used for propagating the exception.

3. Whenever unchecked exception will occur, it will automatically propagated.

4.Whenever checked exception will occur, we need to write “throws” keyword for propagating the exception.

5.e.g. public void m1( ) throws IO Exception, SQL Exception { }

.

**Explain JDK, JRE and JVM?**

|  |  |  |
| --- | --- | --- |
| **JDK vs JRE vs JVM** | | |
| **JDK** | **JRE** | **JVM** |
| It stands for Java Development Kit. | It stands for Java Runtime Environment. | It stands for Java Virtual Machine. |
| It is the tool necessary to compile, document and package Java programs. | JRE refers to a runtime environment in which Java bytecode can be executed. | It is an abstract machine. It is a specification that provides a run-time environment in which Java bytecode can be executed. |
| It contains JRE + development tools. | It’s an implementation of the JVM which physically exists. | JVM follows three notations: Specification, **Implementation,**and **Runtime Instance**. |

### ****Explain public static void main(String args[]) in Java.****

main() in Java is the entry point for any Java program. It is always written as **public static void main(String[] args)**.

* **public**: Public is an access modifier, which is used to specify who can access this method. Public means that this Method will be accessible by any Class.
* **static**: It is a keyword in java which identifies it is class-based. main() is made static in Java so that it can be accessed without creating the instance of a Class. In case, main is not made static then the compiler will throw an error as **main**() is called by the JVM before any objects are made and only static methods can be directly invoked via the class.
* **void**: It is the return type of the method. Void defines the method which will not return any value.
* **main**: It is the name of the method which is searched by JVM as a starting point for an application with a particular signature only. It is the method where the main execution occurs.
* **String args[]**: It is the parameter passed to the main method.

### ****Why Java is platform independent?****

Java is called platform independent because of its byte codes which can run on any system irrespective of its underlying operating system.

### ****Why Java is not 100% Object-oriented?****

Java is not 100% Object-oriented because it makes use of eight primitive data types such as boolean, byte, char, int, float, double, long, short which are not objects.

### ****What are wrapper classes in Java?****

Wrapper classes convert the Java primitives into the reference types (objects). Every primitive data type has a class dedicated to it. These are known as wrapper classes because they “wrap” the primitive data type into an object of that class. Refer to the below image which displays different primitive type, wrapper class and constructor argument.

### ****What are constructors in Java?****

In Java, constructor refers to a block of code which is used to initialize an object. It must have the same name as that of the class. Also, it has no return type and it is automatically called when an object is created.

There are two types of constructors:

1. **Default Constructor:** In Java, a default constructor is the one which does not take any inputs. In other words, default constructors are the no argument constructors which will be created by default in case you no other constructor is defined by the user. Its main purpose is to initialize the instance variables with the default values. Also, it is majorly used for object creation.
2. **Parameterized Constructor:** The parameterized constructor in Java, is the constructor which is capable of initializing the instance variables with the provided values. In other words, the constructors which take the arguments are called parameterized constructors.

### ****What is singleton class in Java and how can we make a class singleton?****

Singleton class is a class whose only one instance can be created at any given time, in one JVM. A class can be made singleton by making its constructor private.

**What is the difference between Array list and vector in Java?**

|  |  |
| --- | --- |
| **ArrayList** | **Vector** |
| Array List is not synchronized. | Vector is synchronized. |
| Array List is fast as it’s non-synchronized. | Vector is slow as it is thread safe. |
| If an element is inserted into the Array List, it increases its Array size by 50%. | Vector defaults to doubling size of its array. |
| Array List does not define the increment size. | Vector defines the increment size. |
| Array List can only use Iterator for traversing an Array List. | Vector can use both Enumeration and Iterator for traversing. |

**What is the difference between equals() and == in Java?**

Equals() method is defined in Object class in Java and used for checking equality of two objects defined by business logic.

“==” or equality operator in Java is a binary operator provided by Java programming language and used to compare primitives and objects. *public boolean equals(Object o)* is the method provided by the Object class. The default implementation uses == operator to compare two objects. For example: method can be overridden like String class. equals() method is used to compare the values of two objects.

**When can you use the super keyword?**

In Java, the super keyword is a reference variable that refers to an immediate parent class object.

When you create a subclass instance, you’re also creating an instance of the parent class, which is referenced to by the super reference variable.

The uses of the Java super Keyword are-

1. To refer to an immediate parent class instance variable, use super.
2. The keyword super can be used to call the method of an immediate parent class.
3. Super() can be used to call the constructor of the immediate parent class.

**What makes a HashSet different from a TreeSet?**

|  |  |
| --- | --- |
| **HashSet** | **TreeSet** |
| It is implemented through a hash table. | TreeSet implements SortedSet Interface that uses trees for storing data. |
| It permits the null object. | It does not allow the null object. |
| It is faster than TreeSet especially for search, insert, and delete operations. | It is slower than HashSet for these operations. |
| It does not maintain elements in an ordered way. | The elements are maintained in a sorted order. |
| It uses equals() method to compare two objects. | It uses compareTo() method for comparing two objects. |
| It does not permit a heterogenous object. | It permits a heterogenous object. |

**What are the differences between HashMap and HashTable in Java?**

|  |  |
| --- | --- |
| **HashMap** | **Hashtable** |
| It is non synchronized. It cannot be shared between many threads without proper synchronization code. | It is synchronized. It is thread-safe and can be shared with many threads. |
| It permits one null key and multiple null values. | It does not permit any null key or value. |
| is a new class introduced in JDK 1.2. | It was present in earlier versions of java as well. |
| It is faster. | It is slower. |
| It is traversed through the iterator. | It is traversed through Enumerator and Iterator. |
| It uses fail fast iterator. | It uses an enumerator which is not fail fast. |
| It inherits AbstractMap class. | It inherits Dictionary class. |

**What is the importance of reflection in Java?**

Reflection is a runtime API for inspecting and changing the behavior of methods, classes, and interfaces. Java Reflection is a powerful tool that can be really beneficial. Java Reflection allows you to analyze classes, interfaces, fields, and methods during runtime without knowing what they are called at compile time. Reflection can also be used to create new objects, call methods, and get/set field values. External, user-defined classes can be used by creating instances of extensibility objects with their fully-qualified names. Debuggers can also use reflection to examine private members of classes.

**How to not allow serialization of attributes of a class in Java?**

The NonSerialized attribute can be used to prevent member variables from being serialized.  
You should also make an object that potentially contains security-sensitive data nonserializable if possible. Apply the NonSerialized attribute to certain fields that store sensitive data if the object must be serialized. If you don’t exclude these fields from serialisation, the data they store will be visible to any programmes with serialization permission.

**Can you call a constructor of a class inside another constructor?**

Yes, we can call a constructor of a class inside another constructor. This is also called as constructor chaining. Constructor chaining can be done in 2 ways-

1. **Within the same class:** For constructors in the same class, the this() keyword can be used.
2. **From the base class:** The super() keyword is used to call the constructor from the base class.  
   The constructor chaining follows the process of inheritance. The constructor of the sub class first calls the constructor of the super class. Due to this, the creation of sub class’s object starts with the initialization of the data members of the super class. The constructor chaining works similarly with any number of classes. Every constructor keeps calling the chain till the top of the chain.

**Contiguous memory locations are usually used for storing actual values in an array but not in ArrayList. Explain.**

An array generally contains elements of the primitive data types such as int, float, etc. In such cases, the array directly stores these elements at contiguous memory locations. While an ArrayList does not contain primitive data types. An arrayList contains the reference of the objects at different memory locations instead of the object itself. That is why the objects are not stored at contiguous memory locations.

**How is the creation of a String using new() different from that of a literal?**  
When we create a string using new(), a new object is created. Whereas, if we create a string using the string literal syntax, it may return an already existing object with the same name.

**Why is synchronization necessary? Explain with the help of a relevant example.**

Java allows multiple threads to execute. They may be accessing the same variable or object. Synchronization helps to execute threads one after another.  
It is important as it helps to execute all concurrent threads while being in sync. It prevents memory consistency errors due to access to shared memory. An example of synchronization code is-

Public synchronized void increment(){ a++ }

As we have synchronized this function, this thread can only use the object after the previous thread has used it.

**Explain the term “Double Brace Initialization” in Java?**

Double Brace Initialization is a Java term that refers to the combination of two independent processes. There are two braces used in this. The first brace creates an anonymous inner class. The second brace is an initialization block. When these both are used together, it is known as Double Brace Initialization. The inner class has a reference to the enclosing outer class, generally using the ‘this’ pointer. It is used to do both creation and initialization in a single statement. It is generally used to initialize collections. It reduces the code and also makes it more readable.

**Why is it said that the length() method of String class doesn’t return accurate results?**

The length() method of String class doesn’t return accurate results because  
it simply takes into account the number of characters within in the String. In other words, code points outside of the BMP (Basic Multilingual Plane), that is, code points having a value of U+10000 or above, will be ignored.

The reason for this is historical. One of Java’s original goals was to consider all text as Unicode; yet, Unicode did not define code points outside of the BMP at the time. It was too late to modify char by the time Unicode specified such code points.

**What are the differences between Heap and Stack Memory in Java?**

The major difference between Heap and Stack memory are:

|  |  |  |
| --- | --- | --- |
| **Features** | **Stack** | **Heap** |
| **Memory** | Stack memory is used only by one thread of execution. | Heap memory is used by all the parts of the application. |
| **Access** | Stack memory can’t be accessed by other threads. | Objects stored in the heap are globally accessible. |
| **Memory Management** | Follows LIFO manner to free memory. | Memory management is based on the generation associated with each object. |
| **Lifetime** | Exists until the end of execution of the thread. | Heap memory lives from the start till the end of application execution. |
| **Usage** | Stack memory only contains local primitive and reference variables to objects in heap space. | Whenever an object is created, it’s always stored in the Heap space. |

### ****What is a package in Java? List down various advantages of packages.****

Packages in Java, are the collection of related classes and interfaces which are bundled together. By using packages, developers can easily modularize the code and optimize its reuse. Also, the code within the packages can be imported by other classes and reused. Below I have listed down a few of its advantages:

* Packages help in avoiding name clashes
* They provide easier access control on the code
* Packages can also contain hidden classes which are not visible to the outer classes and only used within the package
* Creates a proper hierarchical structure which makes it easier to locate the related classes

### ****Why pointers are not used in Java?****

Java doesn’t use pointers because they are unsafe and increases the complexity of the program. Since, Java is known for its simplicity of code, adding the concept of pointers will be contradicting. Moreover, since JVM is responsible for implicit memory allocation, thus in order to avoid direct access to memory by the user,  pointers are discouraged in Java.

### ****What is JIT compiler in Java?****

JIT stands for Just-In-Time compiler in Java. It is a program that helps in converting the Java bytecode into instructions that are sent directly to the processor. By default, the JIT compiler is enabled in Java and is activated whenever a Java method is invoked. The JIT compiler then compiles the bytecode of the invoked method into native machine code, compiling it “just in time” to execute. Once the method has been compiled, the JVM summons the compiled code of that method directly rather than interpreting it. This is why it is often responsible for the performance optimization of Java applications at the run time.

### ****What are access modifiers in Java?****

In Java, access modifiers are special keywords which are used to restrict the access of a class, constructor, data member and method in another class. Java supports four types of access modifiers:

1. Default
2. Private
3. Protected
4. Public

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Modifier** | **Default** | **Private** | **Protected** | **Public** |
| Same class | YES | YES | YES | YES |
| Same Package subclass | YES | NO | YES | YES |
| Same Package non-subclass | YES | NO | YES | YES |
| Different package subclass | NO | NO | YES | YES |
| Different package non-subclass | NO | NO | NO | YES |

### ****Define a Java Class.****

A class in Java is a blueprint which includes all your data.  A class contains fields (variables) and methods to describe the behavior of an object. Let’s have a look at the syntax of a class.

|  |  |
| --- | --- |
| 1  2  3 | class Abc {  member variables // class body  methods} |

### ****What is an object in Java and how is it created?****

An object is a real-world entity that has a state and behavior. An object has three characteristics:

1. State
2. Behavior
3. Identity

An object is created using the ‘new’ keyword. For example:

ClassName obj = new ClassName();

### ****What is Object Oriented Programming?****

Object-oriented programming or popularly known as OOPs is a programming model or approach where the programs are organized around objects rather than logic and functions. In other words, OOP mainly focuses on the objects that are required to be manipulated instead of logic. This approach is ideal for the programs large and complex codes and needs to be actively updated or maintained.

### ****What are the main concepts of OOPs in Java?****

Object-Oriented Programming or OOPs is a programming style that is associated with concepts like:

1. Inheritance: Inheritance is a process where one class acquires the properties of another.
2. Encapsulation: Encapsulation in Java is a mechanism of wrapping up the data and code together as a single unit.
3. Abstraction: Abstraction is the methodology of hiding the implementation details from the user and only providing the functionality to the users.
4. Polymorphism: Polymorphism is the ability of a variable, function or object to take multiple forms.

### Differentiate between the constructors and methods in Java?

|  |  |
| --- | --- |
| **Methods** | **Constructors** |
| 1. Used to represent the behavior of an object | 1. Used to initialize the state of an object |
| 2. Must have a return type | 2. Do not have any return type |
| 3. Needs to be invoked explicitly | 3. Is invoked implicitly |
| 4. No default method is provided by the compiler | 4. A default constructor is provided by the compiler if the class has none |
| 5. Method name may or may not be same as class name | 5. Constructor name must always be the same as the class name |

### ****What is final keyword in Java?****

**final**is a special keyword in Java that is used as a non-access modifier. A final variable can be used in different contexts such as:

* **final variable**

When the final keyword is used with a variable then its value can’t be changed once assigned. In case the no value has been assigned to the final variable then using only the class constructor a value can be assigned to it.

#### ****final method****

When a method is declared final then it can’t be overridden by the inheriting class.

#### ****final class****

When a class is declared as final in Java, it can’t be extended by any subclass class but it can extend other class.

**What is the difference between break and continue statements?**

|  |  |
| --- | --- |
| **break** | **continue** |
| 1. Can be used in switch and loop (for, while, do while) statements | 1. Can be only used with loop statements |
| 2. It causes the switch or loop statements to terminate the moment it is executed | 2. It doesn’t terminate the loop but causes the loop to jump to the next iteration |
| 3. It terminates the innermost enclosing loop or switch immediately | 3. A continue within a loop nested with a switch will cause the next loop iteration to execute |

### ****What is an infinite loop in Java? Explain with an example.****

An infinite loop is an instruction sequence in Java that loops endlessly when a functional exit isn’t met. This type of loop can be the result of a programming error or may also be a deliberate action based on the application behavior. An infinite loop will terminate automatically once the application exits.

**What is the difference between this() and super() in Java?**

In Java, super() and this(), both are special keywords that are used to call the constructor.

|  |  |
| --- | --- |
| **this()** | **super()** |
| 1. this() represents the current instance of a class | 1. super() represents the current instance of a parent/base class |
| 2. Used to call the default constructor of the same class | 2. Used to call the default constructor of the parent/base class |
| 3. Used to access methods of the current class | 3. Used to access methods of the base class |
| 4.  Used for pointing the current class instance | 4. Used for pointing the superclass instance |
| 5. Must be the first line of a block | 5. Must be the first line of a block |

**What is the difference between this() and super() in Java?**

In Java, super() and this(), both are special keywords that are used to call the constructor.

|  |  |
| --- | --- |
| **this()** | **super()** |
| 1. this() represents the current instance of a class | 1. super() represents the current instance of a parent/base class |
| 2. Used to call the default constructor of the same class | 2. Used to call the default constructor of the parent/base class |
| 3. Used to access methods of the current class | 3. Used to access methods of the base class |
| 4.  Used for pointing the current class instance | 4. Used for pointing the superclass instance |
| 5. Must be the first line of a block | 5. Must be the first line of a block |

**Differentiate between static and non-static methods in Java.**

|  |  |
| --- | --- |
| **Static Method** | **Non-Static Method** |
| 1. *The static* keyword must be used before the method name | 1. No need to use the *static* keyword before the method name |
| 2. It is called using the class (className.methodName) | 2. It is can be called like any general method |
| 3. They can’t access any non-static instance variables or methods | 3. It can access any static method and any static variable without creating an instance of the class |

### Explain the term “Double Brace Initialisation” in Java?

Double Brace Initialization is a Java term that refers to the combination of two independent processes. There are two braces used in this. The first brace creates an anonymous inner class. The second brace is an initialization block. When these both are used together, it is known as Double Brace Initialisation. The inner class has a reference to the enclosing outer class, genertally using the ‘this’ pointer. It is used to do both creation and initialization in a single statement. It is generally used to initialize collections. It reduces the code and also makes it more readable.

### ****What is constructor chaining in Java?****

In Java, constructor chaining is the process of calling one constructor from another with respect to the current object. Constructor chaining is possible only through legacy where a subclass constructor is responsible for invoking the superclass’ constructor first. There could be any number of classes in the constructor chain. Constructor chaining can be achieved in two ways:

1. Within the same class using this()
2. From base class using super()

### ****What is a classloader in Java?****

The **Java ClassLoader** is a subset of JVM (Java Virtual Machine) that is responsible for loading the class files. Whenever a Java program is executed it is first loaded by the classloader. Java provides three built-in classloaders:

1. Bootstrap ClassLoader
2. Extension ClassLoader
3. System/Application ClassLoader

### ****Why Java Strings are immutable in nature?****

In Java, string objects are immutable in nature which simply means once the String object is created its state cannot be modified. Whenever you try to update the value of that object instead of updating the values of that particular object, Java creates a new string object. Java String objects are immutable as String objects are generally cached in the String pool. Since String literals are usually shared between multiple clients, action from one client might affect the rest. It enhances security, caching, synchronization, and performance of the application.

### ****What is the difference between an array and an array list?****

|  |  |
| --- | --- |
| **Array** | **ArrayList** |
| Cannot contain values of different data types | Can contain values of different data types. |
| Size must be defined at the time of declaration | Size can be dynamically changed |
| Need to specify the index in order to add data | No need to specify the index |
| Arrays are not type parameterized | Arraylists are type |
| Arrays can contain primitive data types as well as objects | Arraylists can contain only objects, no primitive data types are allowed |

### ****What is a Map in Java?****

In Java, Map is an interface of Util package which maps unique keys to values. The Map interface is not a subset of the main Collection interface and thus it behaves little different from the other collection types. Below are a few of the characteristics of Map interface:

1. Map doesn’t contain duplicate keys.
2. Each key can map at max one value.

### 

### ****What is collection class in Java? List down its methods and interfaces.****

In Java, the collection is a framework that acts as an architecture for storing and manipulating a group of objects. Using Collections you can perform various tasks like searching, sorting, insertion, manipulation, deletion, etc. Java collection framework includes the following:

* Interfaces
* Classes
* Methods

The below image shows the complete hierarchy of the Java Collection.

**What is Collection?**

1. The Collection is a framework that provides an architecture to store and manipulate the group of objects.

2. It provides operations that you perform on a data such as searching, sorting, insertion and deletion on the group of objects.

3. Collection represents a single unit of objects as a group.

**Why do we use Collection?**

(Explain Difference between Array & Collection)

Array Collection

1. It is Fixed in Size. It is Growable in nature.

2. It can hold only Homogeneous Data Elements. It can hold both Homogeneous and Heterogeneous

With Respect to Memory Arrays are With Respect to Memory Collections are Not Recommended to Use Recommended to Use.

**What is Difference between Collection & Collections?**

Collection Collections

It is an interface. It is class.

It can be used to Represent a Group of It is used to sort and synchronize the

Individual Objects as a Single Entity.. collection elements

It provides the methods that can be used It provides the methods which can be used for various

for data structure operations on a collection

**How Array list works?**

1. When we create object of Arraylist, it create Arraylist instance with default capacity 10.

2.Arraylist capacity increases with formula – New Capacity = ((3/2) x Old Capacity)+1

3. When Arraylist increments with new capacity then data from old Arraylist is copied into new instance and old instance is destroyed.

4.When we add or delete data into the Arraylist then multiple data shift operations are performed.  Arraylist follows Indexing.

**Explain all Constructors of Arraylist?**

1.ArrayList l = new ArrayList(); It creates an Empty ArrayList Object with Default Initial Capacity 10.

2. ArrayList l = new ArrayList(intinitialCapacity); It creates an Empty ArrayList Object with specified Initial Capacity.

3. ArrayList l = new ArrayList(Collection c); It creates an equivalent ArrayList Object for the given Collection Object

**How Linkedlist works? (Why insertion & deletion is fast in Linkedlist?)**

1.When we create an object of Linkedlist and add an element to it.

2.It stores element as a node in which previous & next node address is also stored.

3. Node format = ||prev. node addr.| (value) |next node addr.||

4. Due to previous & next node address is stored, hence while updation or insertion & deletion operation data shift operation need not to perform and it makes Linkedlist fast.

**Define Linkedlist?**

1. It is one of implemented class of List interface in collection framework.

2.It allows duplicate values, Insertion order is preserved & indexing is maintained.

3. It implements Cloneable, Serializable interfaces.

4. It follows doubly linked list structure.

5.It is mostly preferable for insertion & deletion operation.

**What is Map?**

1. Map is used for store different object in the pair of “key” and “value”.

2.In map, “key” should be unique.  Insertion order will not be maintained in Map.

**Difference between Hashmap & Hashtable?**

HashMap Hashtable

1. It is not Synchronised. It is Synchronised.

2.It allows multiple threads at a time. It allows single thread at a time.

3.It is not thread safe. It is thread safe.

4. Null key (once) & Null value is allowed. Null key & Null value is not all

**How HashMap works?**

1. When we create HashMap object, HashMap instance as per default capacity 16 buckets is created.

2.When we perform add (put ( )) operation, it accepts data in key & value format.

3. Internally hashing technique is used, that generates hashcode for key and also calculate index to find bucket location for inserting data in HashMap instance.

4. It will store element at that location as a node format. ||previous node address| (Key) | (Value) |next node address||

5.Now when we perform retrieval (get ( )) operation, it asks for key.

6.Again hashing technique is used and bucket location is identified, then equals ( ) method is used to compare key content and if it returns true then value is retrieved.

**What is Contract between equals () & hashcode ()?**

1. If equals ( ) returns true, then objects must have same hashcodes.

2. If equals ( ) returns false, then objects may or may not have same hashcodes.

3.If hashcodes of objects are same, then we can’t conclude output of equals ( ), it may be true or may be false.

4.If hashcodes of objects are different, then output of equals ( ) must be false.

**Difference between Hashmap & Synchronised (or Concurrent) Hashmap?**

HashMap Synchronised or Concurrent HashMap

1.It is non-Synchronized in nature. It is Synchronized in nature.

2.It is not Thread-safe. It is thread-safe.

3.Performance is high. Performance is low.

4. It can throw ConcurrentModificationException. It doesn’t throw ConcurrentModificationException

**Difference between Comparable & Comparator?**

Comparable Comparator

1.This interface is from java.lang package. This interface is from java.util package.

2.It is used for Default sorting. It is used for Custom sorting.

3.It has only one method i.e. compareTo. It has two methods i.e. compare & equals

4. Programmer decides how sorting is to be done. User decides how sorting is to be done.

**What is Identity Hashmap?**

1. In IdentityHashMap JVM will Use == Operator to Identify Duplicate Keys, which is meant for Reference Comparison.

2. e.g. Integer i=new Integer (5);

Integer i1=new Integer (5);

Map m=new IdentityHashMap();

m.put(i, "java"); m.put(i1, "cjc");

System.out.println(m); => {5=java, 5=cjc}

**What is Hash Collision?**

In HashMap, if two keys have same hashcodes then such situation is called as hash collision.

1. In such case, while adding data, doubly Linkedlist is created to insert data. ||prev. node addr.| (Key1) | (Value1) |next node addr.|| ◊ || | (Key2) | (Value2) | ||

2.And retrieval operation is performed using equals ( ) method.

Java 8 Interview Questions

**List down the new features introduced in Java 8?**

**Answer:** **New features that are introduced in Java 8 are enlisted below:**

* Lambda Expressions
* Method References
* Optional Class
* Functional Interface
* Default methods
* Stream API

**What are Functional Interfaces?**

**Answer:** Functional Interface is an interface that has only one abstract method. The implementation of these interfaces is provided using a Lambda Expression which means that to use the Lambda Expression, you need to create a new functional interface or you can use the predefined [functional interface of Java 8](https://www.softwaretestinghelp.com/java-8-interface-changes/).

The annotation used for creating a new Functional Interface is “**@FunctionalInterface**”.

**What is an optional class?**

**Answer:** Optional class is a special wrapper class introduced in Java 8 which is used to avoid NullPointerExceptions. This final class is present under java.util package. NullPointerExceptions occurs when we fail to perform the Null checks.

**What are the default methods?**

**Answer:** Default methods are the methods of the Interface which has a body. These methods, as the name suggests, use the default keywords. The use of these default methods is “Backward Compatibility” which means if JDK modifies any Interface (without default method) then the classes which implement this Interface will break.

On the other hand, if you add the default method in an Interface then you will be able to provide the default implementation. This won’t affect the implementing classes.

**Syntax:**

|  |
| --- |
| public interface questions{            default void print() {    System.out.println("www.softwaretestinghelp.com");               }      } |

**What are the main characteristics of the Lambda Function?**

**Answer: Main characteristics of the Lambda Function are as follows:**

* A method that is defined as Lambda Expression can be passed as a parameter to another method.
* A method can exist standalone without belonging to a class.
* There is no need to declare the parameter type because the compiler can fetch the type from the parameter’s value.
* We can use parentheses when using multiple parameters but there is no need to have parenthesis when we use a single parameter.
* If the body of expression has a single statement then there is no need to include curly braces.

**How can you create a Functional Interface?**

**Answer:** Although Java can identify a Functional Interface, you can define one with the annotation

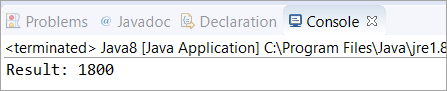
**@FunctionalInterface**

Once you have defined the functional interface, you can have only one abstract method. Since you have only one abstract method, you can write multiple static methods and default methods.

**Below is the programming example of FunctionalInterface written for multiplication of two numbers.**

|  |
| --- |
| @FunctionalInterface // annotation for functional interface  interface FuncInterface {        public int multiply(int a, int b);  }  public class Java8 {       public static void main(String args[]) {          FuncInterface Total = (a, b) -&gt; a \* b;          // simple operation of multiplication of 'a' and 'b'          System.out.println("Result: "+Total.multiply(30, 60));      }  } |

**Output:**

[](https://www.softwaretestinghelp.com/wp-content/qa/uploads/2020/04/8th-Question.png)

**What is Method Reference?**

**Answer:** In Java 8, a new feature was introduced known as Method Reference. This is used to refer to the method of functional interface. It can be used to replace Lambda Expression while referring to a method.

**For Example:** If the Lambda Expression looks like

num -> System.out.println(num)

Then the corresponding Method Reference would be,

System.out::println

where “::” is an operator that distinguishes class name from the method name.

**What is a Predicate? State the difference between a Predicate and a Function?**

**Answer:** Predicate is a pre-defined Functional Interface. It is under java.util.function.Predicate package. It accepts only a single argument which is in the form as shown below,

**Predicate<T>**

| **Predicate** | **Function** |
| --- | --- |
| It has the return type as Boolean. | It has the return type as Object. |
| It is written in the form of **Predicate< T>** which accepts a single argument. | It is written in the form of **Function< T, R>** which also accepts a single argument. |
| It is a Functional Interface which is used to evaluate Lambda Expressions. This can be used as a target for a Method Reference. | It is also a Functional Interface which is used to evaluate Lambda Expressions. In Function< T, R>, T is for input type and R is for the result type. This can also be used as a target for a Lambda Expression and Method Reference. |

**What is a Stream API? Why do we require the Stream API?**

**Answer:** Stream API is a new feature added in Java 8. It is a special class that is used for processing objects from a source such as Collection.

**We require the Stream API because,**

* It supports aggregate operations which makes the processing simple.
* It supports Functional-Style programming.
* It does faster processing. Hence, it is apt for better performance.
* It allows parallel operations.

**What are static methods in Interfaces?**

Static methods, which contains method implementation is owned by the interface and is invoked using the name of the interface, it is suitable for defining the utility methods and cannot be overridden

Spring Interview Questions

### 1) What is Spring?

It is a lightweight, loosely coupled and integrated framework for developing enterprise applications in java.

### 2) What are the advantages of spring framework?

1. Predefined Templates
2. Loose Coupling
3. Easy to test
4. Lightweight
5. Fast Development
6. Powerful Abstraction
7. Declarative support

### 3) What are the modules of spring framework?

1. Test
2. Spring Core Container
3. AOP, Aspects and Instrumentation
4. Data Access/Integration
5. Web

### 4) What is IOC and DI?

IOC (Inversion of Control) and DI (Dependency Injection) is a design pattern to provide loose coupling. It removes the dependency from the program.

### 5) What is the role of IOC container in spring?

IOC container is responsible to:

* create the instance
* configure the instance, and
* assemble the dependencies.

**6) What are the types of IOC container in spring?**

There are two types of IOC containers in spring framework.

1. Bean Factory
2. Application Context.

### 7) What is the difference between Bean Factory and Application Context?

|  |  |
| --- | --- |
| **BeanFactory** | **ApplicationContext** |
| It is an interface defined in org. springframework.beans.factory.**BeanFactory** | It is an interface defined in org. springframework.context.**ApplicationContext** |
| It uses Lazy initialization | It uses Eager/ Aggressive initialization |
| It explicitly provides a resource object using the syntax | It creates and manages resource objects on its own |
| It doesn’t supports internationalization | It supports internationalization |
| It doesn’t supports annotation based dependency | It supports annotation based dependency |

### 8) What is the difference between constructor injection and setter injection?

|  |  |  |
| --- | --- | --- |
| **No.** | **Constructor Injection** | **Setter Injection** |
| 1) | No Partial Injection | Partial Injection |
| 2) | Doesn't override the setter property | Overrides the constructor property if both are defined. |
| 3) | Creates new instance if any modification occurs | Doesn't create new instance if you change the property value |
| 4) | Better for too many properties | Better for few properties. |
|  |  |  |

### 9) What is autowiring in spring? What are the autowiring modes?

Autowiring enables the programmer to inject the bean automatically. We don't need to write explicit injection logic.

The autowiring modes are given below:

|  |  |  |
| --- | --- | --- |
| **No.** | **Mode** | **Description** |
| 1) | no | this is the default mode, it means autowiring is not enabled. |
| 2) | byName | injects the bean based on the property name. It uses setter method. |
| 3) | byType | injects the bean based on the property type. It uses setter method. |
| 4) | constructor | It injects the bean using constructor |

### 10) What are the different bean scopes in spring?

There are 5 bean scopes in spring framework.

|  |  |  |
| --- | --- | --- |
| **No.** | **Scope** | **Description** |
| 1) | singleton | The bean instance will be only once and same instance will be returned by the IOC container. It is the default scope. |
| 2) | prototype | The bean instance will be created each time when requested. |
| 3) | request | The bean instance will be created per HTTP request. |
| 4) | session | The bean instance will be created per HTTP session. |
| 5) | Global session | The bean instance will be created per HTTP global session. It can be used in portlet context only. |

### 11) In which scenario, you will use singleton and prototype scope?

Singleton scope should be used with EJB **stateless session bean** and prototype scope with EJB **stateful session bean**.

**12) What are the transaction management supports provided by spring?**

Spring framework provides two type of transaction management supports:

1. **Programmatic Transaction Management**: should be used for few transaction operations.
2. **Declarative Transaction Management**: should be used for many transaction operations.

**13) What are classes for spring JDBC API?**

1. JdbcTemplate
2. SimpleJdbcTemplate
3. NamedParameterJdbcTemplate
4. SimpleJdbcInsert
5. SimpleJdbcCall.

### 14) What are the different features of Spring Framework?

Following are some of the major features of Spring Framework :

* **Lightweight:** Spring is lightweight when it comes to size and transparency.
* **Inversion of control (IOC):** The objects give their dependencies instead of creating or looking for dependent objects. This is called Inversion Of Control.
* **Aspect oriented Programming (AOP):** Aspect oriented programming in Spring supports cohesive development by separating application business logic from system services.
* **Container:**Spring Framework creates and manages the life cycle and configuration of the application objects.
* **MVC Framework:** Spring Framework’s MVC web application framework is highly configurable. Other frameworks can also be used easily instead of Spring MVC Framework.
* **Transaction Management:** Generic abstraction layer for transaction management is provided by the Spring Framework. Spring’s transaction support can be also used in container less environments.
* **JDBC Exception Handling:** The JDBC abstraction layer of the Spring offers an exception hierarchy, which simplifies the error handling strategy.

**15. What are the various ways of using Spring Framework?**

Spring Framework can be used in various ways. They are listed as follows:

1. As a Full-fledged Spring web application.
2. As a third-party web framework, using Spring Frameworks middle-tier.
3. For remote usage.
4. As Enterprise Java Bean which can wrap existing POJOs (Plain Old Java Objects).

### **16. What’s the difference between @Component, @Controller, @Repository & @Service annotations in Spring?**

**@Component:** This marks a java class as a bean. It is a generic stereotype for any Spring-managed component. The component-scanning mechanism of spring now can pick it up and pull it into the application context.

**@Controller:** This marks a class as a Spring Web MVC controller. Beans marked with it are automatically imported into the Dependency Injection container.

**@Service:** This annotation is a specialization of the component annotation. It doesn’t provide any additional behavior over the @Component annotation. You can use @Service over @Component in service-layer classes as it specifies intent in a better way.

**@Repository:** This annotation is a specialization of the @Component annotation with similar use and functionality. It provides additional benefits specifically for DAOs. It imports the DAOs into the DI container andmakes the unchecked exceptions eligible for translation into Spring DataAccessException.

### **17. What do you understand by @Required annotation?**

@Required is applied to bean property setter methods. This annotation simply indicates that the affected bean property must be populated at the configuration time with the help of an explicit property value in a bean definition or with autowiring. If the affected bean property has not been populated, the container will throw BeanInitializationException.

### **18. What do you understand by @Autowired annotation?**

The **@Autowired** annotation provides more accurate control over where and how autowiring should be done. This annotation is used to autowire bean on the setter methods, constructor, a property or methods with arbitrary names or multiple arguments. By default, it is a type driven injection.

### **19. What do you understand by @Qualifier annotation?**

When you create more than one bean of the same type and want to wire only one of them with a property you can use the **@Qualifier** annotation along with **@Autowired** to remove the ambiguity by specifying which exact bean should be wired.

**20.  What do you understand by @RequestMapping annotation?**

@RequestMapping annotation is used for mapping a particular HTTP request method to a specific class/ method in controller that will be handling the respective request. This annotation can be applied at both levels:

* **Class level**: Maps the URL of the request
* **Method level**: Maps the URL as well as HTTP request method.

### 21.**What are the difference between Spring AOP and AspectJ AOP?**

|  |  |
| --- | --- |
| **Spring AOP** | **AspectJ AOP** |
| Runtime weaving through proxy is done | Compile time weaving through AspectJ Java tools is done |
| It supports only method level PointCut | It suports field level Pointcuts |
| It is DTD based | It is schema based and Annotation configuration |

### 22**. What do you mean by Proxy in Spring Framework?**

An object which is created after applying advice to a target object is known as a Proxy. In case of client objects the target object and the proxy object are the same.

### **23. In Spring, what is Weaving?**

The process of linking an aspect with other application types or objects to create an advised object is called Weaving. In Spring AOP, weaving is performed at runtime.

**24. Name Some of the Design Patterns Used in the Spring Framework?**

* **Singleton Pattern** – singleton-scoped beans
* **Factory Pattern** – Bean Factory classes
* **Prototype Pattern** – prototype-scoped beans
* **Adapter Pattern** – Spring Web and Spring MVC
* **Proxy Pattern** – Spring Aspect-Oriented Programming support
* **Template Method Pattern** – *JdbcTemplate*, *HibernateTemplate*, etc.
* **Front Controller** – Spring MVC *DispatcherServlet*
* **Data Access Object** – Spring DAO support
* **Model View Controller**– Spring MVC

**25. What Is Spring JdbcTemplate Class and How to Use It?**

The Spring JDBC template is the primary API through which we can access database operations logic that we’re interested in:

* Creation and closing of connections
* Executing statements and stored procedure calls
* Iterating over the *ResultSet* and returning results

### **26. What Is Spring WebFlux?**

[Spring WebFlux](https://docs.spring.io/spring/docs/current/spring-framework-reference/web-reactive.html#webflux) is Spring's reactive-stack web framework, and it's an alternative to Spring MVC. In order to achieve this reactive model and be highly scalable, the entire stack is non-blocking.

### **27. What Is the Default Bean Scope in Spring Framework?**

By default, a Spring Bean is initialized as a singleton.

**28. Which Is the Best Way of Injecting Beans and Why?**

The recommended approach is to use constructor arguments for mandatory dependencies and setters for optional ones. This is because constructor injection allows injecting values to immutable fields and makes testing easier.

**29. What is Spring configuration file?**

Spring configuration file is an XML file. This file contains the classes information and describes how these classes are configured and introduced to each other

**30.What is Spring AOP?**

Aspect-oriented programming, or AOP, is a programming technique that allows programmers to modularize crosscutting concerns, or behavior that cuts across the typical divisions of responsibility, such as logging and transaction management. The core construct of AOP is the aspect, which encapsulates behaviors affecting multiple classes into reusable modules.

Spring Boot Interview Question

### **1.  What is Spring Boot?**

Spring Boot is called a microservice framework that is built on top of the spring framework. This can help developers to focus more on convention rather than configuration.

1. The main aim of Spring boot is to give you a production-ready application. So, the moment you create a spring-boot project, it is runnable and can be executed/deployed on the server.
2. It comes with features like autoconfiguration, auto dependency resolution, embedded servers, security, health checks which enhances the productivity of a developer.

### **2. How to create Spring Boot project in eclipse?**

One of the ways to create a spring boot project in eclipse is by using **Spring Initializer.**

You can go to the official website of spring and add details such as version, select maven or Gradle project, add your groupId, artifactId, select your required dependencies and then click on CREATE PROJECT.

Once the project is created, you can download it and extract and import it in your eclipse or STS.

And see your project is ready! To Install Spring Boot in Eclipse – Go to Eclipse IDE, click on “Help”->then go to Eclipse marketplace->and type Spring IDE and click on the finish button.

### **3. How to deploy spring boot application in tomcat?**

Whenever you will create your [spring boot application](https://www.mygreatlearning.com/academy/learn-for-free/courses/dockerize-spring-boot-application/?gl_blog_id=25325) and run it, Spring boot will automatically detect the embedded tomcat server and deploy your application on tomcat.  
After successful execution of your application, you will be able to launch your rest endpoints and get a response.

### **4. What is the difference between Spring and Spring Boot?**

Difference between Spring and Spring boot are as follows:

**Spring –**

1. Is a dependency injection framework.
2. It is basically used to manage the life cycle of [java classes](https://www.mygreatlearning.com/blog/java-tutorial-for-beginners/?gl_blog_id=25325) (beans). It consists of a lot of boilerplate configuration.
3. Uses XML based configuration.
4. It takes time to have a spring application up and running and it’s mainly because of boilerplate code.

**Spring boot-**

1. It is a suite of pre- configured frameworks and technologies which helps to remove boilerplate configuration.
2. Uses annotations.
3. It is used to create a production-ready code.

### **5. What is actuator in spring boot?**

An actuator is one of the best parts of spring boot which consists of production-ready features to help you monitor and manage your application.

With the help of an actuator, you can monitor what is happening inside the running application.  
Actuator dependency figures out the metrics and makes them available as a new endpoint in your application and retrieves all required information from the web. You can identify beans, the health status of your application, CPU usage, and many more with the actuator. By using @Endpoint annotation, you can create a custom endpoint.

### **6. How to change port in spring boot?**

The default port number to start your SpringBoot application is **8080**.

Just to change the port number, you need to add **server.port=8084**c(your port number) property in your application.properties file and start your application.

### **7. How to create war file in spring boot?**

To create a war file in spring boot you need to define your packaging file **as war** in your pom.xml(if it is maven project).

Then just do **maven clean** **and install** so that your application will start building. Once the build is successful, just go into your Target folder and you can see .war file generated for your application.

### **8. What is JPA in spring boot?**

[JPA](https://www.mygreatlearning.com/jpa/free-courses/?gl_blog_id=25325) is basically **Java Persistence API**. It’s a specification that lets you do ORM when you are connecting to a [relational database](https://www.mygreatlearning.com/blog/what-is-rdbms/?gl_blog_id=25325) which is Object-Relational Mapping.

So, when you need to connect from your java application to relational database, you need to be able to use something like JDBC and run [SQL queries](https://www.mygreatlearning.com/blog/sql-tutorial-for-beginners/?gl_blog_id=25325) and then you get the results and convert them into Object instances.

ORM lets you map your entity classes in your [SQL](https://www.mygreatlearning.com/academy/learn-for-free/courses/introduction-to-sql/?gl_blog_id=25325) tables so that when you connect to the database , you don’t need to do query yourself, it’s the framework that handles it for you.

And JPA is a way to use ORM, it’s an API which lets you configure your entity classes and give it to a framework so that the framework does the rest.

### **9. How to save image in database using spring boot?**

1. First configure [mysql](https://www.mygreatlearning.com/academy/learn-for-free/courses/my-sql-basics/?gl_blog_id=25325" \t "_blank) in your spring boot application.
2. Then you can map your entities with your db tables using JPA.
3. And with the help of save() method in JPA you can directly insert your data into your database

@RestController

@RequestMapping("/greatleasrning")

public class Controller {

@Autowired

private final GreatLearningRepository greatLearningRepository;

public Controller(GreatLearningRepository greatLearningRepository) {

}

In above case, your data which may be in [JSON format](https://www.mygreatlearning.com/json-format/free-courses/?gl_blog_id=25325) can be inserted successfully into database.

@RequestMapping(method = RequestMethod.POST)

ResponseEntity<?> insert(@RequestBody Course course) {

greatLearningRepository.save(course);

return ResponseEntity.accepted().build();

}

}

### **10. What is auto configuration in spring boot?**

AutoConfiguration is a process by which Spring Boot automatically configures all the infrastructural beans. It declares the built-in beans/objects of the spring specific module such as JPA, spring security and so on based on the dependencies present in your applications class path.

**For example:** If we make use of Spring JDBC, the spring boot autoconfiguration feature automatically registers the DataSource and JDBCTemplete bean.  
This entire process of automatically declaring the framework specific bean without the need of writing the [xml code](https://www.mygreatlearning.com/xml/free-courses/?gl_blog_id=25325) or java config code explicity  is called Autoconfiguration which is done by springBoot with the help of an annotation called **@EnableAutoconfiguration** alternatively **@SpringBootApplication**.

### **11. How to resolve whitelabel error page in spring boot application?**

This is quite common error in spring boot application which says 404(page not found).

We can mostly resolve this in 3 ways:

1. **Custom Error Controller**– where you will be implementing ErrorController  interface which is provided by SpringFramework and then overriding its getErrorPath() so that you can return a custom path whenever such type of error is occurred.
2. **By Displaying Custom error page**– All you have to do is create an error.html page and place it into the src/main/resources/templates path. The BasicErrorController of of springboot will automatically pick this file by default.
3. **By disabling the whitelabel error page**– this is the easiest way where all you need to do is server.error.whitelabel.enabled property to false in the application.properties file to disable the whitelabel error page.

### **12. How to fetch data from database in spring boot?**

You can use the following steps to connect your application with [MySQL database](https://www.mygreatlearning.com/blog/mysql-tutorial/?gl_blog_id=25325).  
1. First create a database in MySQL with create DATABASE student;  
2. Now, create a table inside this DB:  
CREATE TABLE student(studentid INT PRIMARY KEY NOT NULL AUTO\_INCREMENT, studentname VARCHAR(255));   
3. Create a SpringBoot application and add [JDBC](https://www.mygreatlearning.com/blog/jdbc-tutorial/?gl_blog_id=25325), MySQL and web dependencies.  
4. In application.properties, you need to configure the database.

spring.datasource.url=jdbc:mysql://localhost:3306/studentDetails

spring.datasource.username=system123

spring.datasource.password=system123

spring.jpa.hibernate.ddl-auto=create-drop

5. In your controller class, you need to handle the requests.

package com.student;

import org.springframework.web.bind.annotation.RequestMapping;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.jdbc.core.JdbcTemplate;

import org.springframework.web.bind.annotation.RestController;

@RestController

public class JdbcController {

@Autowired

JdbcTemplate jdbc;

@RequestMapping("/save")

public String index(){

jdbc.execute("insert into student (name)values(GreatLearnings)");

return "Data Entry Successful";

}

}

6. Run the application and check the entry in your Database.

### **13. How to use logger in spring boot?**

There are many logging options available in SpringBoot. Some of them are mentioned below:

* Using log4j2:

import org.apache.logging.log4j.Logger;

import org.apache.logging.log4j.LogManager;

// [...]

Logger logger = LogManager.getLogger(LoggingController.class);

* Using Lombok:

All you need to do is add a dependency called **org.projectlombok**in your pom.xml as shown below:

<dependency>

<groupId>org.projectlombok</groupId>

<artifactId>lombok</artifactId>

<version>1.18.4</version>

<scope>provided</scope>

</dependency>

Now you can create a loggingController and add the **@Slf4j**annotation to it. Here we would not create any logger instances.

@RestController

@Slf4j

public class LoggingController {

@RequestMapping("/logging")

public String index() {

log.trace("A TRACE Message");

log.debug("A DEBUG Message");

log.info("An INFO Message");

log.warn("A WARN Message");

log.error("An ERROR Message");

return "Here are your logs!”;

}

}

So, there are many such ways in spring boot to use logger.

### **14. What is bootstrapping in spring boot?**

One of the way to [bootstrap](https://www.mygreatlearning.com/academy/learn-for-free/courses/intro-to-bootstrap/?gl_blog_id=25325) your spring boot application is using Spring Initializer.  
you can go to the official website of spring  and select your version, and add you groupID, artifactId and all the required dependencies.

And then you can create your restEndpoints and build and run your project.  
There you go, you have bootstrapped your spring boot application.

### **15. How to create jar file in spring boot?**

To create a jar file in spring boot you need to define your packaging file as **jar** in your pom.xml(if it is maven project).

Then just do maven build with specifying **goals as package** so that your application will start building.

Once the build is successful, just go into your Target folder and you can see .jar file generated for you application.

### **16. What is dependency injection in spring boot?**

[Dependency injection](https://www.mygreatlearning.com/spring/tutorials/spring-dependency-injection/?gl_blog_id=25325) is a way through which the Spring container injects one object into another. This helps for loose coupling of components.

**For example:** if class student uses functionality of department class, then we say student class has dependency of Department class. Now we need to create object of class Department in your student class so that it can directly use functionalities of department class is called dependency injection.

### **17. How to store image in MongoDB using spring boot?**

One of the way for storing image in [MongoDB](https://www.mygreatlearning.com/academy/learn-for-free/courses/mongodb-tutorial/?gl_blog_id=25325) is by using Spring Content. And also you should have the below dependency in your pom.xml.

<dependency>

<groupId>com.github.paulcwarren</groupId>

<artifactId>spring-content-mongo-boot-starter</artifactId>

<version>0.0.10</version>

</dependency>

You should have a GridFsTemplate bean in your applicationContext.

@Configuration

public class Config

@Bean

public GridFsTemplate gridFsTemplate() throws Exception {

return new GridFsTemplate(mongoDbFactory(), mappingMongoConverter());

}

...

Now add attributes so that your content will be associated to your entity.

@ContentId

private String contentId;

@ContentLength

private long contentLength = 0L;

@MimeType

private String mimeType = "text/plain";

And last but not the least, add a store interface.

@StoreRestResource(path="greatlearningImages")

public interface GreatLearningImageStore extends ContentStore<Candidate, String> {

}

That’s all you have to do to store your images in mongoDb using Springboot.

### **18. How to configure hibernate in spring boot?**

The important and required dependency to configure hibernate is:

1. **spring-boot-starter-data-jpa**
2. **h2** (you can also use any other database)

Now, provide all the database connection properties in application.properties file of your application in order to connect your JPA code with the database.

Here we will configure H2 database in application.properties file:

spring.datasource.url=jdbc:h2:file:~/test

spring.datasource.driverClassName=org.h2.Driver

spring.datasource.username=test

spring.datasource.password=test

spring.jpa.database-platform=org.hibernate.dialect.H2Dialect

spring.h2.console.enabled=true

spring.h2.console.path=/h2-console

Adding the above properties in your application.properties file will help you to interact with your database using JPA repository interface.

### **19. Mention the advantages of Spring Boot.**

**Advantages of Spring Boot –**

1. It allows convention over configuration hence you can fully avoid XML configuration.
2. SpringBoot reduces lots of development time and helps to increase productivity.
3. Helps to reduce a lot of boilerplate code in your application.
4. It comes with embedded HTTP servers like tomcat, Jetty, etc to develop and test your applications.
5. It also provides CLI (Command Line Interface) tool which helps you  to develop and test your application from CMD.

### **20. Explain what is thyme leaf and how to use thymeleaf?**

Thymeleaf is a server-side java template engine which helps processing and creating [HTML](https://www.mygreatlearning.com/academy/learn-for-free/courses/html-tutorial/?gl_blog_id=25325), [XML](https://www.mygreatlearning.com/blog/xml-tutorial/?gl_blog_id=25325), [JavaScript](https://www.mygreatlearning.com/academy/learn-for-free/courses/intro-to-javascript/?gl_blog_id=25325) , [CSS](https://www.mygreatlearning.com/academy/learn-for-free/courses/css-tutorial/?gl_blog_id=25325), and text. Whenever the dependency in pom.xml (in case of  maven project) is find, springboot automatically configures Thymeleaf to serve dynamic web content.

**Dependency: spring-boot-starter-thymeleaf**

We can place the thyme leaf templates which are just the HTML files in **src/main/resources/templates/** folder so that spring boot can pick those files and renders whenever required.

Thymeleaf will parse the index.html and will replace the dynamic values with its actual value that is been passed from the controller class.  
That’s it, once you run your Spring Boot application and your message will be rendered in web browsers.

### **21. What is the need for Spring Boot DevTools?**

This is one of the amazing features provided by Spring Boot, where it restarts the spring boot application whenever any changes are being made in the code.

 Here, you don’t need to right-click on the project and run your application again and again. Spring Boot dev tools does this for you with every code change.  
**Dependency to be added is: spring-boot-devtools**

The main focus of this module is to improve the development time while working on Spring Boot applications.

### **22. Can we change the port of the embedded Tomcat server in Spring boot?**

Yes, you can change the port of embedded Tomcat server in Spring boot by adding the following property in your **application.properties** file.

server.port=8084

The default port number of the tomcat server to run the spring boot application is 8080, which is further possible to change it.

So we can change the port of tomcat following ways given below:-

* Using application.properties
* Using application.yml
* Using EmbeddedServletContainerCustomizer interface.
* Using WebServerFactoryCustomizer interface.
* Using Command-Line Parameter.

### **23. Mention the steps to connect Spring Boot application to a database using JDBC**

Below are the steps to connect your Spring Boot application to a database using JDBC:

Before that, you need to add required dependencies that are provided by spring-boot to connect your application with JDBC.

**Step 1**: First create a database in MySQL with create DATABASE student;

**Step 2**:  Now, create a table inside this DB:  
CREATE TABLE student(studentid INT PRIMARY KEY NOT NULL AUTO\_INCREMENT,

studentname VARCHAR(255));

**Step 3**: Create a springBoot and add JDBC,mysql and web dependencies.  
**Step 4**: In application.properties, you need to configure the database.

spring.datasource.url=jdbc:mysql://localhost:3306/studentDetails

spring.datasource.username=system123

spring.datasource.password=system123

spring.jpa.hibernate.ddl-auto=create-drop

**Step 5**: In your controller class, you need to handle the requests.

package com.student;

import org.springframework.web.bind.annotation.RequestMapping;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.jdbc.core.JdbcTemplate;

import org.springframework.web.bind.annotation.RestController;

@RestController

public class JdbcController {

@Autowired

JdbcTemplate jdbc;

@RequestMapping("/save")

public String index(){

jdbc.execute("insert into student

(name)values(GreatLearnings)");

return "Data Entry Successful";

}

}

**Step 6**: Run the application and check the entry in your Database.

**Step 7**: You can also go ahead and open the URL and you will see “Data Entry Successful” as your output.

### **24. What are the @RequestMapping and @RestController annotation in Spring Boot used for?**

The **@RequestMapping** annotation can be used at class-level or method level in your controller class.

The global request path that needs to be mapped on a controller class can be done by using **@RequestMapping** at class-level. If you need to map a particular request specifically to some method level.

Below is a simple example to refer to:

@RestController

@RequestMapping("/greatLearning")

public class GreatLearningController {

@RequestMapping("/")

String greatLearning(){

return "Hello from greatLearning ";

}

@RequestMapping("/welcome")

String welcome(){

return "Welcome from GreatLearning";

}

}

The **@RestController** annotation is used at the class level.

You can use @RestController when you need to use that class as a request handler class.All the requests can be mapped and handled in this class.

**@RestController** itself consists **@Controller** and **@ResponseBody** which helps us to remove the need of annotating every method with @ResponseBody annotation.

**Below is a simple example to refer to for use of @RestController annotation:**

@RestController

@RequestMapping(“bank-details”)

public class DemoRestController{

@GetMapping(“/{id}”,produces =”application/json”)

public Bank getBankDetails(@PathVariable int id){

return findBankDetailsById();

}

}

Here, @ResponseBody is not required as the class is annotated with @RestController.

### **25. What do you understand  by auto-configuration in Spring Boot and how to disable the auto-configuration?**

AutoConfiguration is a process by which Spring Boot automatically configures all the infrastructural beans. It declares the built-in beans/objects of the spring-specific module such as JPA, spring-security, and so on based on the dependencies present in your application’s classpath.  
**For example:** If we make use of Spring JDBC, the spring boot autoconfiguration feature automatically registers the DataSource and JDBCTemplete bean.  
This entire process of automatically declaring the framework-specific bean without the need of writing the XML code or java-config code explicitly  is called Autoconfiguration which is done by spring-boot with the help of an annotation called **@EnableAutoconfiguration** alternatively **@SpringBootApplication.**

1. You can exclude the attribute of @EnableAutoConfiguration where you don’t want it to be configured implicity in order to disable the spring boot’s auto-configuration feature.

2. Another way of disabling auto-configuration is by using the property file:

**For example:**

spring.autoconfigure.exclude=

org.springframework.boot.autoconfigure.mongo.MongoAutoConfiguration,

org.springframework.boot.autoconfigure.data.MongoDataConfiguration,

In the above example, we have disabled the autoconfiguration of MongoDB.

### **26. Can you give an example for ReadOnly as true in Transaction management?**

Yes, example for ReadOnly as true in Transaction Management is:

Suppose you have a scenario where you have to read data from your database like if you have a STUDENT database and you have to read the student details such as studentID, and studentName.

 So in such scenarios, you will have to set read-only on the transaction.

### **27. Mention the advantages of the YAML file than Properties file and the different ways to load**

YAML file in Spring boot.

YAML gives you more clarity and is very friendly to humans. It also supports **maps, lists, and other scalar types.**

YAML comes with hierarchical nature which helps in avoiding repetition as well as indentations.

If we have different deployment profiles such as  development, testing, or production and we may have different configurations for each environment, so instead of creating new files for each environment we can place them in a single YAML file.  
But in the case of the properties file, you cannot do that.

**For example:**

spring:

profiles:

active:

-test

---

spring:

profiles:

active:

-prod

---

spring:

profiles:

active:

-development

### **28. What do you understand by Spring Data REST?**

By using Spring Data Rest, you have access to all the RESTful resources that revolves around Spring Data repositories.

Refer the below example:

@RepositoryRestResource(collectionResourceRel = "greatlearning", path = "sample")

public interface GreatLearningRepo extends CustomerRepository< greatlearning, Long> {

}

Now you can use the POST method in the below manner:

{

“Name”:”GreatLearning”

}

And you will get response as follow:

{

“Name”:”GreatLearning”

}

\_\_\_\_\_\_\_\_\_\_

{

"name": "Hello greatlearning "

"\_links": {

"self": {

"href": "<a href="http://localhost:8080/sample/1">http://localhost:8080/ greatlearning /1</a>"

},

" greatlearning ": {

“href": "<a href="http://localhost:8080/sample/1">http://localhost:8080/ greatlearning /1</a>"

}

}

In the above, you can see the response of the newly created resource.

### **31. How to create a login page in spring boot?**

You can create a simple and default login page in spring boot, you can make use of Spring security. Spring security secures all HTTP endpoints where the user has to login into the default HTTP form provided by spring.

We need to add **spring-boot-starter-security** dependency in your pom.xml or build.gradle and a default username and password can be generated with which you can log in.

### **32. What is the main class in spring boot?**

Usually in java applications, a class that has a main method in it is considered as a main class. Similarly, in spring boot applications main class is the class which has a public static void main() method and which starts up the SpringApplicationContext.

### **33. How to use crud repository in spring boot?**

In order to use crud repository in spring boot, all you have to do is extend the crud repository which in turn extends the Repository interface as a result you will not need to implement your own methods.

Create a simple spring boot application which includes below dependency:  
**spring-boot-starter-data-jpa**, **spring-boot-starter-data-rest**

And extend your repository interface as shown below**:**

package com.greatlearning;

import java.util.List;

import org.springframework.data.repository.CrudRepository;

import org.springframework.data.rest.core.annotation.RepositoryRestResource;

@RepositoryRestResource

public interface GreatLearning extends CrudRepository<Candidate, Long>

{

public List<Candidate> findById(long id);

//@Query("select s from Candidate s where s.age <= ?")

public List<Candidate> findByAgeLessThanEqual (long age);

}

### **34. How to run spring-boot jar from the command line?**

In order to run spring boot jar from the command line, you need to update you pom.xml(or build.gradle) of your project with the maven plugin.

<build>

<plugins>

<plugin>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-maven-plugin</artifactId>

</plugin>

</plugins>

</build>

Now, Build your application and package it into the single executable jar. Once the jar is built you can run it through the command prompt  using the below query:

java -jar target/myDemoService-0.0.1-SNAPSHOT.jar

And you have your application running.

### **35. What is Spring Boot CLI and how to execute the Spring Boot project using boot CLI?**

Spring Boot CLI is nothing but a command-line tool which is provided by Spring so that you can develop your applications quicker and faster.

To execute your spring boot project using CLI, you need first to download CLI from Spring’s official website and extract those files. You may see a bin folder present in the Spring setup which is used to execute your spring boot application.

As Spring boot CLI allows you to execute groovy files, you can create one and open it in the terminal.  
And then execute  **./spring run filename.groovy;**

### **36. what is the rest controller in spring boot?**

The **@RestController** annotation is used at the class level.

You can use @RestController when you need to use that class as a request handler class.All the requests can be mapped and handled in this class.

**@RestController** itself consists **@Controller** and **@ResponseBody** which helps us to remove the need of annotating every method with @ResponseBody annotation.

**Below is a simple example to refer to for use of @RestController annotation:**

@RestController

@RequestMapping(“bank-details”)

public class DemoRestController{

@GetMapping(“/{id}”,produces =”application/json”)

public Bank getBankDetails(@PathVariable int id){

return findBankDetailsById();

}

}

Here, @ResponseBody is not required as the class is annotated with @RestController.

### **37. How to handle 404 error in spring boot?**

Consider a scenario, where there are no stockDetails in the DB and still, whenever you hit the GET method you get 200(OK) even though the resource is not found which is not expected. Instead of 200, you should get 404 error.  
So to handle this, you need to create an exception, in the above scenario “StockNotFoundException”.

GetMapping("/stocks/{number}")

public Stock retriveStock(@PathVariable int number)

{

Stock stock = service.findOne(number);

if(Stock ==null)

//runtime exception

throw new StockNotFoundException("number: "+ number);

return stock;

}

Now, create a Constructor from [Superclass](https://www.mygreatlearning.com/blog/java-super-keyword-and-wrapper-class/?gl_blog_id=25325).

Right-click on the file -> Go to Source ->And generate constuctors from superclass-> and check the RuntimeException(String)-> and generate.

And add an annotation called **@ResponseStatus** which will give you 404 (not found) error.

package com.greatlearning;

import org.springframework.http.HttpStatus;

import org.springframework.web.bind.annotation.ResponseStatus;

@ResponseStatus(HttpStatus.NOT\_FOUND)

public class StockNotFoundException extends RuntimeException

{

public StockNotFoundException(String message)

{

super(message);

}

}

Now, you can hit the same URL again and there you go, you get a 404 error when a resource is not found.

### **38. Which is the spring boot latest version?**

The latest version of spring boot is**2.6.0**. It came out with a lot of dependency upgrades, java 15 support and much more.

Yes, now as you are brushed up with spring boot interview questions and answers. We have also tried to cover all the springboot interview questions for experienced professionals. Hope you can easily crack the spring boot interview now!

Please feel free to comment below if you have any queries related to the above questions or answers. Also, do comment if you find any other questions that you think must be included in the above list of questions.