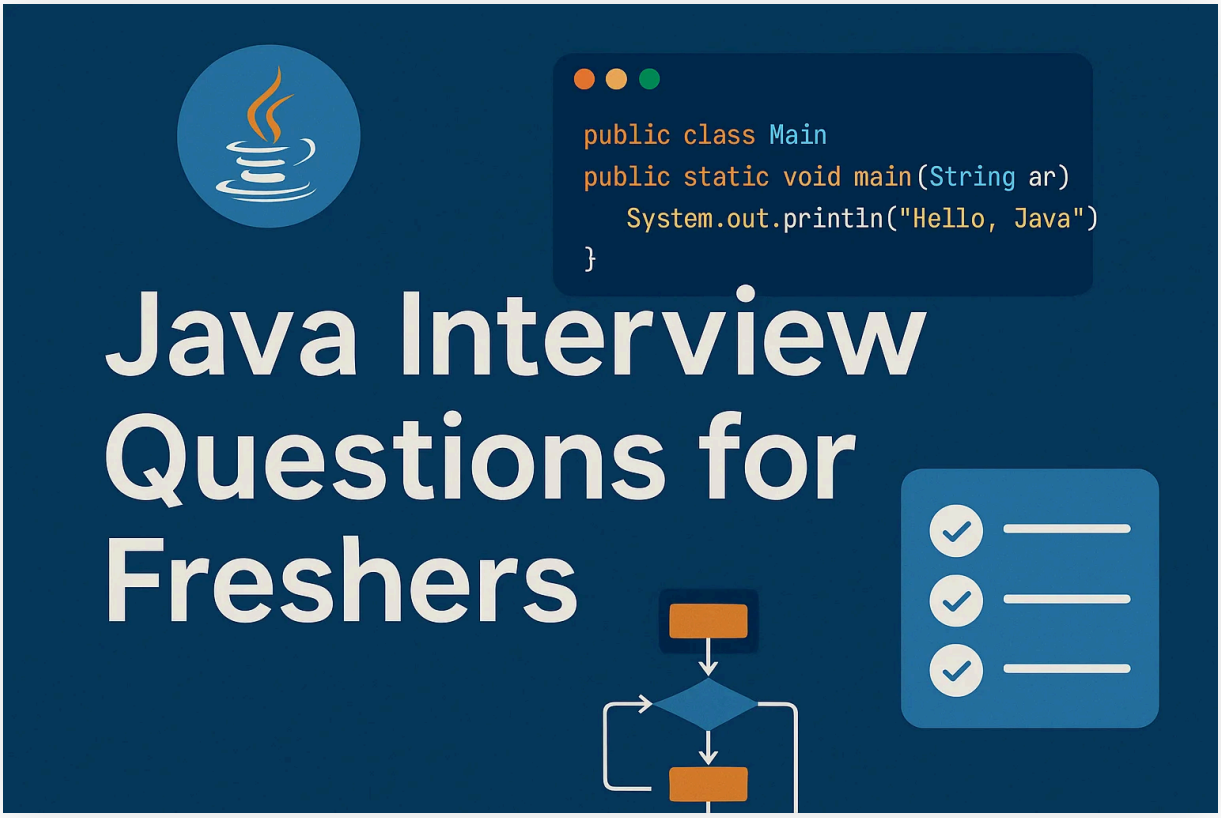
Java Cody

Top 100 Java Interview Questions for Freshers (2025)

May 12, 2025



Are you preparing for your first software developer interview? This comprehensive guide to the **top 100 Java Interview Questions for Freshers** will help you confidently answer questions during Java coding interviews, technical assessments, and oral rounds. These questions cover everything from Java basics to object-oriented programming, exception handling, and collections.

Java Interview Questions for Freshers

1. What is Java?

Java is a high-level, class-based, object-oriented programming language that is platform-independent, thanks to the Java Virtual Machine (JVM). It allows developers to write code once and run it anywhere.

2. What are the differences between JDK, JRE, and JVM?

- **JDK (Java Development Kit)**: Contains JRE + development tools (such as `javac`).
- **JRE (Java Runtime Environment)**: Contains JVM + runtime libraries.
- **JVM (Java Virtual Machine)**: Runs Java bytecode and makes Java platform-independent.

3. What are the primitive data types in Java?

The eight primitive data types in Java are:

- `byte` : 1 byte
- `short` : 2 bytes
- `int` : 4 bytes
- `long` : 8 bytes
- `float` : 4 bytes
- `double` : 8 bytes
- `char` : 2 bytes
- `boolean` : 1 byte

4. Explain the difference between `==` and `equals()` method in Java.

- `==` compares the references (memory address) of two objects.
- `equals()` compares the actual contents (values) of the objects.

```
String s1 = new String("Hello");
String s2 = new String("Hello");

System.out.println(s1 == s2);      // false
System.out.println(s1.equals(s2)); // true
```

5. What is inheritance in Java?

Inheritance is a mechanism where one class acquires the properties and behaviors (methods) of another class. It allows for code reusability.

```
class Animal {
    void sound() {
        System.out.println("Animal makes a sound");
```

```
    }
}

class Dog extends Animal {
    void sound() {
        System.out.println("Dog barks");
    }
}
```

6. What is polymorphism in Java?

Polymorphism means “many shapes.” It allows methods to behave differently based on the object that is calling them. There are two types: compile-time (method overloading) and runtime (method overriding).

```
class Animal {
    void sound() {
        System.out.println("Animal makes a sound");
    }
}

class Dog extends Animal {
    void sound() {
        System.out.println("Dog barks");
    }
}
```

7. Explain the difference between method overloading and method overriding.

- **Method Overloading**: Same method name but different parameters (compile-time polymorphism).
- **Method Overriding**: Same method name and parameters, but in a subclass, it provides its own implementation (runtime polymorphism).

8. What is the `final` keyword in Java?

The `final` keyword can be applied to variables, methods, and classes:

- **Final variable**: Cannot be reassigned after initialization.

- **Final method**: Cannot be overridden.
- **Final class**: Cannot be subclassed.

9. What is the `static` keyword in Java?

The `static` keyword is used for memory management. It indicates that a variable or method belongs to the class rather than any instance of the class.

```
class Example {
    static int count = 0;

    static void display() {
        System.out.println("Static method");
    }
}
```

10. What is the difference between `String` and `StringBuilder` in Java?

- `String` is immutable. Any modification results in a new object being created.
- `StringBuilder` is mutable. It can be modified without creating new objects.

11. What is an exception in Java?

An exception is an event that disrupts the normal flow of a program. It can be caught and handled using `try`, `catch`, and `finally`.

```
try {
    int result = 10 / 0;
} catch (ArithmeticException e) {
    System.out.println("Cannot divide by zero");
} finally {
    System.out.println("This will always be executed");
}
```

12. What are checked and unchecked exceptions in Java?

- **Checked exceptions**: Exceptions that are checked at compile-time (e.g., `IOException`, `SQLException`).

- **Unchecked exceptions**: Exceptions that are checked at runtime (e.g., `NullPointerException`, `ArithmeticException`).

13. What is the use of `super` in Java?

The `super` keyword refers to the superclass and is used to:

- Access superclass methods or fields.
- Call a constructor of the superclass.

```
class Animal {
    void sound() {
        System.out.println("Animal makes a sound");
    }
}

class Dog extends Animal {
    void sound() {
        super.sound(); // Calling superclass method
        System.out.println("Dog barks");
    }
}
```

14. What is the difference between `ArrayList` and `LinkedList` in Java?

- **ArrayList**: Implements `List` interface using a dynamic array. It provides fast random access but slower insertions and deletions.
- **LinkedList**: Implements `List` and `Deque` interfaces using a doubly linked list. It provides faster insertions and deletions but slower random access.

15. What is the `finally` block in Java?

The `finally` block is always executed after a `try-catch` block, regardless of whether an exception is thrown or not.

16. What is the difference between `HashMap` and `TreeMap` in Java?

- **HashMap**: Unordered collection, allows `null` keys and values.
- **TreeMap**: Ordered collection, does not allow `null` keys but allows `null` values.

17. What are interfaces in Java?

An interface is a reference type, similar to a class, that can contain only constants, method signatures, default methods, static methods, and nested types.

```
 interface Animal {
    void sound();
}

class Dog implements Animal {
    public void sound() {
        System.out.println("Dog barks");
    }
}
```

18. **What is an abstract class in Java?**

An abstract class cannot be instantiated. It can have both abstract methods (without implementation) and concrete methods (with implementation).

19. **What is the difference between an interface and an abstract class in Java?**

- **Interface**: Can only contain abstract methods (until Java 8, after which it can have default methods).
- **Abstract class**: Can have both abstract and concrete methods and can maintain state (instance variables).

20. **What is multithreading in Java?**

Multithreading allows multiple threads to run concurrently in a program. This helps in performing multiple tasks simultaneously.

```
 class MyThread extends Thread {
    public void run() {
        System.out.println("Thread running");
    }
}
```

21. **What is synchronization in Java?**

Synchronization is used to control access to a shared resource by multiple threads to avoid data inconsistency.

```
class Counter {
    private int count = 0;

    synchronized void increment() {
        count++;
    }
}
```

22. What is the `volatile` keyword in Java?

The `volatile` keyword ensures that the value of a variable is always read from and written to the main memory, rather than being cached by threads.

23. What is the `transient` keyword in Java?

The `transient` keyword prevents serialization of a variable, meaning it won't be saved when the object is serialized.

```
class Person implements Serializable {
    transient int age;
}
```

24. What is the difference between `StringBuilder` and `StringBuffer`?

- **StringBuilder**: Faster, but not thread-safe.
- **StringBuffer**: Slower, but thread-safe.

25. What is the purpose of `wait()` and `notify()` in Java?

These methods are used for inter-thread communication. A thread can call `wait()` to release the lock and allow other threads to execute. `notify()` wakes up one thread waiting on the object.

26. What is the Singleton Design Pattern in Java?

The Singleton pattern ensures a class has only one instance and provides a global point of access.

```
class Singleton {
    private static Singleton instance;

    private Singleton() {}

    public static Singleton getInstance() {
        if (instance == null) {
            instance = new Singleton();
        }
        return instance;
    }
}
```

27. What is the Observer Design Pattern?

The Observer pattern allows an object (subject) to notify its dependents (observers) when its state changes.

28. What is the Factory Design Pattern?

The Factory pattern provides a way to create objects without specifying the exact class of object that will be created.

29. What is the `clone()` method in Java?

The `clone()` method creates a copy of an object. It is a part of the `Object` class and must be overridden in a class if it implements the `Cloneable` interface.

30. What is the `HashSet` in Java?

`HashSet` is a collection that does not allow duplicate elements and does not guarantee any order.

31. What is the `Iterator` in Java?

An `Iterator` is an interface used to traverse through a collection (like List, Set, etc.).

```
List<String> list = Arrays.asList("A", "B", "C");
Iterator<String> iterator = list.iterator();
while (iterator.hasNext()) {
```

```
        System.out.println(iterator.next());
    }
```

32. What is the difference between `ArrayList` and `Vector` in Java?

- **ArrayList**: Resizable array with no synchronized methods.
- **Vector**: Similar to `ArrayList`, but synchronized.

33. What is the `default` keyword in Java?

Introduced in Java 8, it allows you to define methods in an interface with a default implementation.

```
interface Animal {
    default void sound() {
        System.out.println("Animal sound");
    }
}
```

34. What is the `static` block in Java?

The `static` block is used for static initializations of a class. It runs once when the class is loaded.

```
class Example {
    static {
        System.out.println("Static block");
    }
}
```

35. What is the `instanceof` operator in Java?

The `instanceof` operator checks whether an object is an instance of a specific class or subclass.

```
String s = "Hello";
System.out.println(s instanceof String); // true
```

36. What is the `java.lang` package in Java?

`java.lang` is the default package that is automatically imported. It contains fundamental classes like `String`, `Math`, `Object`, and `System`.

37. What is the difference between `throw` and `throws` in Java?

- `throw`: Used to throw an exception explicitly.
- `throws`: Declares the exceptions that a method may throw.

38. What is the use of `finally` block in exception handling?

The `finally` block is used to execute code after `try-catch` blocks, regardless of whether an exception occurs or not.

39. What is the difference between `String` and `StringBuffer` in Java?

- **String**: Immutable (does not modify the original object).
- **StringBuffer**: Mutable (can modify the original object).

40. What is method reference in Java 8?

A method reference is a shorthand notation of a lambda expression to call a method.

```
List<String> names = Arrays.asList("Alice", "Bob", "Charlie");
names.forEach(System.out::println); // Method reference
```

41. What are streams in Java 8?

Streams allow you to process sequences of elements (such as collections) in a functional style.

42. What is lambda expression in Java 8?

A lambda expression provides a clear and concise way to represent one method interface using an expression.

```
List<String> names = Arrays.asList("John", "Jane", "Doe");
names.forEach(name -> System.out.println(name));
```

43. What is the `Optional` class in Java 8?

`Optional` is a container object which may or may not contain a value, designed to avoid `NullPointerException`.

```
Optional<String> name = Optional.of("John");
name.ifPresent(System.out::println);
```

44. What is the difference between `local` and `anonymous` classes in Java?

- **Local classes**: Declared within a method.
- **Anonymous classes**: Classes without a name, created in the moment.

45. What is serialization in Java?

Serialization is the process of converting an object into a byte stream for storage or transmission.

```
ObjectOutputStream out = new ObjectOutputStream(new
FileOutputStream("object.ser"));
out.writeObject(object);
out.close();
```

46. What is deserialization in Java?

Deserialization is the reverse process, where a byte stream is converted back into an object.

```
ObjectInputStream in = new ObjectInputStream(new
FileInputStream("object.ser"));
Object object = in.readObject();
in.close();
```

47. What is a `Callable` in Java?

A `Callable` is similar to a `Runnable`, but it can return a result or throw an exception.

48. What is a `Future` in Java?

`Future` represents the result of an asynchronous computation, allowing you to retrieve the result or handle exceptions once the computation is done.

49. What is an anonymous inner class in Java?

An anonymous class is a class that is defined and instantiated at the same time, often used for event handling.

50. What are the benefits of using the `Java Collections Framework`?

The `Java Collections Framework` provides a unified architecture for storing and manipulating groups of objects, offering various data structures like `List`, `Set`, and `Map`.

51. What are the types of access modifiers in Java?

Java has four access modifiers:

- `public`: The class, method, or variable is accessible from anywhere.
- `protected`: The method or variable is accessible within the same package or subclasses.
- `private`: The method or variable is accessible only within the same class.
- **Default**: Accessible only within the same package.

52. What is the difference between `ArrayList` and `LinkedList` in Java?

- **ArrayList**: Uses a dynamic array. Access time is fast, but insertions and deletions can be slow due to shifting elements.
- **LinkedList**: Uses a doubly linked list. Insertions and deletions are faster than `ArrayList`, but accessing elements by index is slower.

53. What is the `default` method in Java interfaces?

Introduced in Java 8, `default` methods allow interfaces to have methods with a default implementation, so classes implementing the interface don't have to provide the implementation.

```
interface Animal {
    default void sound() {
        System.out.println("Animal sound");
    }
}
```

54. What is the `static` keyword used for in Java?

The `static` keyword is used to denote class-level members. It means the variable or method is shared among all instances of the class.

55. What is the `this` keyword in Java?

The `this` keyword refers to the current instance of the class and is commonly used to refer to instance variables and methods.

```
class Person {
    String name;

    Person(String name) {
        this.name = name;
    }
}
```

56. What is the difference between `continue` and `break` in Java?

- `continue` : Skips the current iteration of a loop and proceeds to the next iteration.
- `break` : Exits the loop completely.

57. What is method overloading in Java?

Method overloading occurs when a class has multiple methods with the same name but different parameter lists.

```
class Example {
    void print(int a) { System.out.println(a); }
    void print(String a) { System.out.println(a); }
}
```

58. What is method overriding in Java?

Method overriding occurs when a subclass provides its own implementation for a method already defined in the superclass.

```
class Animal {
    void sound() {
        System.out.println("Animal makes a sound");
    }
}

class Dog extends Animal {
    void sound() {
        System.out.println("Dog barks");
    }
}
```

59. What is the difference between `String` and `StringBuffer` in Java?

- `String` is immutable; any change creates a new object.
- `StringBuffer` is mutable; modifications can be made without creating new objects.

60. What is the purpose of the `main` method in Java?

The `main` method is the entry point for any Java application. The JVM calls this method to start program execution.

```
public static void main(String[] args) {
    // Program execution starts here
}
```

61. What is the `instanceof` operator in Java?

The `instanceof` operator checks whether an object is an instance of a specific class or subclass.

```
String s = "Hello";
System.out.println(s instanceof String); // true
```

62. What is the use of the `super` keyword in Java?

The `super` keyword refers to the superclass and is used to:

- Call a superclass method.
- Access superclass constructor.
- Access superclass fields.

63. What is the difference between `final`, `finally`, and `finalize` in Java?

- `final`: Used to declare constants, prevent method overriding, and prevent inheritance.
- `finally`: A block that is always executed after `try-catch`.
- `finalize`: A method that is called by the garbage collector before an object is destroyed.

64. What is the `hashCode()` method in Java?

The `hashCode()` method returns an integer value that represents the hash code of the object, which is used in hash-based collections like `HashMap` and `HashSet`.

65. What is the `equals()` method in Java?

The `equals()` method is used to compare the content of two objects for equality.

```
String str1 = "hello";
String str2 = "hello";
System.out.println(str1.equals(str2));  // true
```

66. What is a `HashMap` in Java?

A `HashMap` is a collection class that implements the `Map` interface, providing a way to store key-value pairs with fast access time.

```
Map<String, Integer> map = new HashMap<>();
map.put("A", 1);
map.put("B", 2);
```

67. What is the `Thread.sleep()` method in Java?

The `Thread.sleep()` method causes the currently executing thread to sleep (pause) for a specified amount of time.

```
    Thread.sleep(1000);  // Sleep for 1 second
```

68. What are lambda expressions in Java?

Lambda expressions allow you to define a method inline and pass it as an argument to a method. They provide a way to write more concise and readable code.

```
    List<String> list = Arrays.asList("Apple", "Banana", "Cherry");
   list.forEach(item -> System.out.println(item));
```

69. What is a `constructor` in Java?

A constructor is a special method used to initialize objects. It has the same name as the class and does not return a value.

```
    class Person {
        String name;

        Person(String name) {
            this.name = name;
        }
    }
```

70. What is an abstract class in Java?

An abstract class cannot be instantiated and is used as a base class for other classes. It can have both abstract (without implementation) and non-abstract methods (with implementation).

71. What is the `clone()` method in Java?

The `clone()` method creates a copy of an object. It is part of the `Object` class and requires the object to implement the `Cloneable` interface.

```
    class Person implements Cloneable {
        String name;
```

```
    public Person clone() throws CloneNotSupportedException {
        return (Person) super.clone();
    }
}
```

72. What is the difference between `ArrayList` and `Vector` in Java?

- `ArrayList` : Resizable array, not synchronized.
- `Vector` : Similar to `ArrayList` , but synchronized, making it thread-safe but slower.

73. What is a `synchronized` block in Java?

A `synchronized` block ensures that only one thread can execute a block of code at a time, providing thread safety.

```
synchronized (this) {
    // critical section
}
```

74. What is the purpose of the `volatile` keyword in Java?

The `volatile` keyword ensures that a variable's value is always read from and written to the main memory, ensuring visibility across threads.

75. What is a `default` constructor in Java?

A default constructor is a constructor provided by Java when no explicit constructor is defined. It initializes the object with default values.

76. What is an inner class in Java?

An inner class is a class defined within another class. It can access the outer class's members, even if they are private.

```
class Outer {
    class Inner {
        void display() {
            System.out.println("Inside inner class");
        }
```

```
        }
    }
```

77. What is the `throw` keyword in Java?

The `throw` keyword is used to explicitly throw an exception in a method or block of code.

```
throw new ArithmeticException("Error occurred");
```

78. What is the difference between `throw` and `throws` in Java?

- `throw` : Used to throw an exception explicitly.
- `throws` : Used in method signatures to declare exceptions that may be thrown by the method.

79. What is the difference between `StringBuilder` and `StringBuffer` in Java?

- `StringBuilder` : Faster and not thread-safe.
- `StringBuffer` : Slower but thread-safe.

80. What is a `Future` in Java?

A `Future` represents the result of an asynchronous computation, which may not have completed yet.

81. What is the `Iterable` interface in Java?

The `Iterable` interface represents any class whose objects can be iterated over using the `for-each` loop.

82. What is a `WeakHashMap` in Java?

A `WeakHashMap` is a type of map where the keys are weak references. If a key is no longer in use, it can be garbage collected.

83. What is a `LinkedHashMap` in Java?

A `LinkedHashMap` maintains the insertion order of elements in addition to providing the functionality of a `HashMap` .

84. What is the `ThreadLocal` class in Java?

`ThreadLocal` provides thread-local variables, meaning each thread has its own independent copy of the variable.

85. What are `ConcurrentHashMap` and its advantages?

`ConcurrentHashMap` is a thread-safe map that allows concurrent updates and provides higher scalability than a synchronized map.

86. What is the `Comparator` interface in Java?

The `Comparator` interface defines a method to compare two objects, allowing custom sorting of elements.

```
class NameComparator implements Comparator<Person> {
    public int compare(Person p1, Person p2) {
        return p1.name.compareTo(p2.name);
    }
}
```

87. What is a `CountDownLatch` in Java?

`CountDownLatch` is a synchronization aid that allows one or more threads to wait until a set of operations in other threads completes.

88. What is `ForkJoinPool` in Java?

`ForkJoinPool` is a special implementation of the `ExecutorService` that helps with parallel processing of tasks that can be recursively

ChatGPT said:

split into smaller sub-tasks.

89. What is the difference between `ExecutorService` and `ThreadPoolExecutor` in Java?

`ExecutorService` is an interface that provides a higher-level replacement for the `Thread` class. `ThreadPoolExecutor` is its default implementation.

90. What is the purpose of the `wait()` method in Java?

The `wait()` method is used for thread synchronization. A thread can call `wait()` to release the lock and allow other threads to access the synchronized block.

91. What is the `notify()` method in Java?

The `notify()` method is used to wake up a single thread that is waiting on an object's monitor (via `wait()`).

92. What is the `notifyAll()` method in Java?

The `notifyAll()` method is used to wake up all threads that are waiting on an object's monitor.

93. What is the `Executor` framework in Java?

The `Executor` framework provides a higher-level replacement for managing thread creation and task execution, simplifying multithreading.

94. What is a `Semaphore` in Java?

A `Semaphore` is a synchronization mechanism that controls access to a shared resource by multiple threads.

95. What is the `Atomic` class in Java?

The `Atomic` classes (like `AtomicInteger`, `AtomicLong`) provide atomic (thread-safe) operations on variables without synchronization.

96. What is a `CyclicBarrier` in Java?

A `CyclicBarrier` is used to make threads wait until a set number of threads have reached a common barrier point.

97. What is a `ReentrantLock` in Java?

`ReentrantLock` is a class that provides explicit locking with additional features like try-lock and timed lock.

98. What is a `ReadWriteLock` in Java?

A `ReadWriteLock` allows multiple readers but only one writer, providing higher concurrency in some situations.

99. What is the `Callable` interface in Java?

The `Callable` interface is similar to `Runnable`, but it can return a result and throw exceptions.

100. What is the difference between `Runnable` and `Callable` in Java?

- `Runnable`: Does not return a result or throw exceptions.
- `Callable`: Can return a result and throw exceptions.

Download Java and Start Coding

Read More :

Top 100 Java Interview Questions for Freshers (2025)

Top 10 Best Java Online Compiler

Java Download: Install Java JDK and Set Up JAVA_HOME

What is a Java Virtual Machine (JVM): JVM Architecture

What is Java Runtime Environment ? Why JRE is Platform Dependent

← PREVIOUS
Top 10 Best Java Online Compiler

Related Posts

Java JDK Oracle: The Complete Guide for Developer

3 Comments / Java / By Java Cody

What is a Java Development Kit(Java JDK): Everything You Need to Know

3 Comments / Java / By Java Cody

Leave a Comment

Logged in as Java Cody. Edit your profile. Log out? Required fields are marked *

Type here..

Post Comment