Interview Questions

Mastering OOPS Basics in Java Youtube Series

1. Classes and Objects: Syntax and Examples

What is the difference between a class and an object in Java?

A class is a blueprint or template for creating objects. It defines a datatype by bundling data and methods that work on the data. An object is an instance of a class. It is a real-world entity created from the class and can have its own state and behavior.

How do you create an object in Java?

You create an object using the `new` keyword followed by the class name and parentheses. Example:

```
ClassName objectName = new ClassName();
```

Explain the role of constructors in object creation.

Constructors are special methods that initialize objects. When you create an object using the `new` keyword, the constructor is called automatically to set up the initial state of the object.

2. Getters, Setters, and `this` Keyword

Why do we need getters and setters in Java?

Getters and setters provide controlled access to the fields of a class. They allow you to read and modify private data while maintaining encapsulation, ensuring that the internal representation of the object is hidden from the outside world.

What is the purpose of the `this` keyword in Java?

The `this` keyword refers to the current instance of the class. It is often used to differentiate between class fields and parameters with the same name, and to invoke other constructors in the same class.

Can you provide an example of using the `this` keyword?

Yes, consider the following example:

```
public class MyClass {
    private int value;
    public MyClass(int value) {
        this.value = value; // 'this.value' refers to the class field
```

```
    }
    public void setValue(int value) {
        this.value = value; // 'this.value' refers to the class field
    }
    public int getValue() {
        return this.value; // 'this' is optional here
    }
}
```

3. Constructors

What are the different types of constructors in Java?

There are two types of constructors in Java

1. **Default Constructor**: Provided by the compiler if no constructors are defined. It has no parameters.
2. **Parameterized Constructor**: Defined by the programmer, it can accept arguments to initialize the object with specific values.

Can constructors be overloaded in Java?

Yes, constructors can be overloaded in Java. You can have multiple constructors with different parameter lists, allowing objects to be initialized in different ways.

4. Basic Overview of Garbage Collection

What is garbage collection in Java?

Garbage collection in Java is the process of automatically reclaiming memory by removing objects that are no longer in use. It helps in managing memory and preventing memory leaks.

Can you force garbage collection in Java?

You can suggest garbage collection by calling `System.gc()` or `Runtime.getRuntime().gc()`, but you cannot force it. The JVM decides when to perform garbage collection.

5. Static Data, Static Blocks, and Static Functions

What is the purpose of static variables in Java?

Static variables are shared among all instances of a class. They belong to the class itself rather than any specific instance, and only one copy exists regardless of how many objects are created.

What is a static block in Java?

A static block is used to initialize static variables. It is executed when the class is loaded into memory, before any objects of the class are created.

Can you provide an example of a static function?

```
public class MyClass {
    private static int counter = 0;

    public static void incrementCounter() {
        counter++;
    }

    public static int getCounter() {
        return counter;
    }
}
```

Here, `incrementCounter` and `getCounter` are static functions that operate on the static variable `counter`.

6. Final Data, Final Functions, and Final Classes

What does the `final` keyword do when applied to a variable?

When applied to a variable, the `final` keyword makes it a constant. The value of a final variable cannot be changed once it is initialized.

Can a final method be overridden?

No, a final method cannot be overridden by subclasses. It is locked at the class level where it is defined.

What is the significance of a final class?

A final class cannot be subclassed or inherited. This is useful when you want to prevent any modifications to the class by extending it.

7. Abstract Functions and Abstract Classes

What is an abstract class in Java?

An abstract class is a class that cannot be instantiated on its own and is meant to be subclassed. It can contain abstract methods (methods without a body) that must be implemented by subclasses.

How does an abstract method differ from a regular method?

An abstract method is declared without an implementation and must be defined in a subclass. A regular method, on the other hand, has a body and can be executed as is.

Can an abstract class have constructors?

Yes, an abstract class can have constructors, but they are only called when a subclass is instantiated.