Mastering Java Basics

Essential Java Code Snippets for Every Developer

Essential Java Code Snippets for Programmers

Java is a versatile and widely-used programming language, especially for building platform-independent applications. Here are some important Java code snippets that every programmer should be familiar with:

Hello World Program

The classic starting point for any programming language is the "Hello World" program. It demonstrates the basic structure of a Java program.

```
public class HelloWorld {
    public static void main(String[] args) {
        System.out.println("Hello, World!");
    }
}
```

Basic Data Types and Variables

Understanding data types and how to declare variables is fundamental in Java.

```
public class DataTypes {
    public static void main(String[] args) {
        int myNumber = 5;
        float myFloat = 5.99f;
        char myLetter = 'D';
        boolean myBool = true;
        String myText = "Hello";
    }
}
```

Conditional Statements

Conditional statements allow you to execute code based on certain conditions.

```
public class ConditionalStatements {
   public static void main(String[] args) {
      int number = 10;

      if (number > 0) {
         System.out.println("The number is positive.");
      } else if (number < 0) {
         System.out.println("The number is negative.");
      } else {
         System.out.println("The number is zero.");
      }
   }
}
```

Loops

Loops are used to repeat a block of code as long as a specified condition is met.

For Loop

```
public class ForLoop {
   public static void main(String[] args) {
      for (int i = 0; i < 5; i++) {
         System.out.println("Iteration: " + i);
      }
   }
}
```

While Loop

```
public class WhileLoop {
   public static void main(String[] args) {
      int i = 0;
      while (i < 5) {
         System.out.println("Iteration: " + i);
         i++;
      }
   }
}
```

Arrays

Arrays are used to store multiple values in a single variable.

```
public class ArraysExample {
  public static void main(String[] args) {
    String[] cars = {"Volvo", "BMW", "Ford", "Mazda"};
    for (String car : cars) {
      System.out.println(car);
    }
  }
}
```

Methods

Methods are blocks of code that perform a specific task and are executed when called.

```
public class MethodsExample {
  public static void main(String[] args) {
    printMessage("Hello, Java!");
  }

  public static void printMessage(String message) {
    System.out.println(message);
  }
}
```

Classes and Objects

Java is an object-oriented programming language, and understanding classes and objects is crucial.

```
public class Car {
  String model;
  int year;

  public Car(String model, int year) {
    this.model = model;
    this.year = year;
  }

  public void displayInfo() {
    System.out.println("Model: " + model + ", Year: " + year);
  }
```

```
    public static void main(String[] args) {
        Car myCar = new Car("Toyota", 2020);
        myCar.displayInfo();
    }
}
```

Exception Handling

Exception handling is important for managing errors and maintaining the normal flow of an application.

```
public class ExceptionHandling {
    public static void main(String[] args) {
        try {
            int division = 10 / 0;
            System.out.println(division);
        } catch (ArithmeticException e) {
            System.out.println("Cannot divide by zero.");
        } finally {
            System.out.println("Execution completed.");
        }
    }
}
```

File I/O

Reading from and writing to files is a common task in Java.

```
import java.io.FileWriter;
import java.io.IOException;
import java.io.File;
import java.util.Scanner;

public class FileIOExample {
    public static void main(String[] args) {
        // Writing to a file
        try {
            FileWriter writer = new FileWriter("example.txt");
            writer.write("Hello, file!");
            writer.close();
        } catch (IOException e) {
            System.out.println("An error occurred.");
            e.printStackTrace();
```

```
        }

        // Reading from a file
        try {
            File file = new File("example.txt");
            Scanner reader = new Scanner(file);
            while (reader.hasNextLine()) {
                String data = reader.nextLine();
                System.out.println(data);
            }
            reader.close();
        } catch (IOException e) {
            System.out.println("An error occurred.");
            e.printStackTrace();
        }
    }
}
```

These snippets provide a foundation for understanding Java programming. As you continue to learn, you'll build on these basics to develop more complex applications.