Java Basics Expanded

Page 1: Introduction to Java

Java is a high-level, platform-independent, object-oriented programming language.

Features of Java:

- Platform Independent: Code can run on any device with JVM.
- Object-Oriented: Follows OOP principles like inheritance, encapsulation, etc.
- Secure: Eliminates explicit pointers and ensures safe code execution.
- Robust: Strong memory management, exception handling.
- Multithreaded: Handles multiple tasks simultaneously.

Why use Java?

- Suitable for web, mobile, and enterprise applications.
- Huge community and extensive libraries.

Example Program:

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

Page 2: Core Concepts in Java

Object-Oriented Programming (OOP):

- Inheritance: Deriving new classes from existing ones.

- Polymorphism: Ability to take multiple forms (method overloading/overriding).

- Encapsulation: Wrapping data and code into a single unit.

- Abstraction: Hiding implementation details and showing only functionality.

Example: Defining a Class and Object

```
class Animal {
    String name;
    void speak() {
        System.out.println("This is an animal.");
    }
}
```

Page 3: Data Types and Operators

Java supports the following data types:

- Primitive: int, float, double, char, boolean, etc.

- Non-primitive: Arrays, Classes, Strings, etc.

Operators:

- Arithmetic (+, -, *, /, %)

- Relational (==, !=, >, <)

- Logical (&&, ||, !)

Example:

```
int a = 10, b = 20;

System.out.println(a + b); // Output: 30
```

Page 4: Control Statements and Loops

Control Statements:

- if-else, switch-case for decision-making.

- Example:

```
if (score > 50) {
    System.out.println("Pass");
} else {
    System.out.println("Fail");
}
```

Loops:

- for, while, do-while for iteration.

```
for (int i = 0; i < 5; i++) {
    System.out.println(i);
}
```

Page 5: Exception Handling in Java

Java provides robust error handling mechanisms.

Try-Catch Example:

```
try {
    int result = 10 / 0;
} catch (ArithmeticException e) {
    System.out.println("Cannot divide by zero: " + e);
}
```

Finally Block:

```
finally {
    System.out.println("Cleanup code here.");
}
```