How To Java Tutorials

Lesson 7 Control Flow Statements

Author: Ramy Hakam

TABLE OF CONTENTS

Control Flow Statements

The statements inside your source files are generally Executed from top to bottom, in the order that they appear. Control flow statements, however, break up the flow of Execution by employing decision making, looping, and Branching, enabling your program to conditionally execute Particular blocks of code .

The if-then Statement

The if-then statement is the most basic of all the control Flow statements. It tells your program to execute a Certain section of code only if a particular test evaluates To true.

The if-then Else Statement

The if-then-else statement provides a secondary path of Execution when an "if" clause evaluates to false. .

The Ternary Operator

Which can be thought of as shorthand for an if-then-Else Statement its means like That.
If some Condition is true, assign the value of value1 to Result. Otherwise, assign the value of value2 to result."

CODEING

Creating If-else Demo

The switch Statement

Unlike if-then and if-then-else statements, the switch Statement can have a number of possible execution paths You can suppose that Switch statements is more effective Than a set of if -then statements

CODEING

Creating Switch Demo

An if-then-else statement can test expressions based on ranges Of Values or conditions, whereas a switch statement tests Expressions based only on a single integer, enumerated value, Or String object.

Each break statement terminates the Enclosing switch Statement .Control flow continues with the First statement Following the switch block. The break statements Are necessary Because without them, Statements In switch blocks fall Through: All statements after The Matching case label are Executed in sequence, regardless of The Expression of Subsequent case labels.

CODEING

Creating Code Example

Multiple case

You can Combine more than one case on each block of Switch statements so if you has some different cases that Will execute the same block of code you can combine them

CODEING

```
Creating Code Example
```

Using Strings in switch Statements

In Java SE 7 and later, you can use a String object in the Switch statement's expression.

CODEING

```
Creating Code Example
```

The while Statement

The while statement continually executes a block of Statements while a particular condition is true.
The while statement evaluates expression, which must return a boolean value. If the expression evaluates to true, the While statement executes the statement(s) in the while block.
The while statement continues testing the expression and Executing its block until the expression evaluates to false

The do-while Statement

The Java programming language also provides a do-while Statement. Which is the same as while Statment, But The difference between do-while and while is that do-while Evaluates its expression at the bottom of the loop instead of The top. Therefore, the statements within the do block are Always executed at least once.

CODEING

Creating While Demo

The for Statement

The for statement provides a compact way to iterate over a Range of values. Programmers often refer to it as the "for Loop" because of the way in which it repeatedly loops until A particular condition is satisfied.
He general form of the for statement can be expressed as Follows:

```
for (initialization; termination; increment) {
    statement(s)
}
```

CODEING

Creating For Demo

The enhanced for statement

The for statement also has another form designed for iteration Through Collections and arrays This form is sometimes referred To as the enhanced for statement, and can be used to make Your loops more compact and easy to read.

CODEING

Creating code Example

The break Statement

The break statement has two forms: labeled and unlabeled. You saw the unlabeled form in the previous discussion of The switch statement. You can also use an unlabeled break To terminate a for, while, or do-while loop,

An unlabeled break statement terminates the innermost Switch, for, while, or do-while statement, but a labeled Break terminates an outer statement.

CODEING

Creating Branching demo

The continue Statement

The continue statement skips the current iteration of a for, While, or do-while loop.

The unlabeled form skips to the end of the innermost loop's Body and evaluates the boolean expression that controls the Loop.

A labeled continue statement skips the current iteration of an Outer loop marked with the given label.

CODEING

Creating Code Example

Test Your Self

Assignment

Make a simple Java application That:

1- Create an array of 10 elements

2- Initialize the array elements using For statement With this form

Every element contain the factorial of Its index Ex array[3]= 3!= 3*2*1;

Then print only elements that has 3 As a divisor.

Thank You For Watching

I am still not Better, But I always try .

See You Next