Java switch Statement

In this tutorial, you will learn to use the switch statement in Java to control the flow of your program's execution with the help of examples.

The `switch` statement allows us to execute a block of code among many alternatives.

The syntax of the `switch` statement in Java is:

```
switch (expression) {

  case value1:
    // code
    break;

  case value2:
    // code
    break;

  ...
  ...

  default:
    // default statements
  }
```

How does the switch-case statement work?

The expression is evaluated once and compared with the values of each case.

- If expression matches with value1, the code of `case value1` are executed. Similarly, the code of `case value2` is executed if expression matches with value2.
- If there is no match, the code of the **default case** is executed.

Note: The working of the switch-case statement is similar to the [Java if...else...if ladder]. However, the syntax of the `switch` statement is cleaner and much easier to read and write.

Example: Java switch Statement

```
// Java Program to check the size
// using the switch...case statement

class Main {
  public static void main(String[] args) {

    int number = 44;
    String size;
```

```
    // switch statement to check size
    switch (number) {

      case 29:
        size = "Small";
        break;

      case 42:
        size = "Medium";
        break;

      // match the value of week
      case 44:
        size = "Large";
        break;

      case 48:
        size = "Extra Large";
        break;

      default:
        size = "Unknown";
        break;

    }
    System.out.println("Size: " + size);
  }
}
```

Output:

Size: Large

In the above example, we have used the switch statement to find the size. Here, we have a variable number. The variable is compared with the value of each case statement.

Since the value matches with **44**, the code of `case 44` is executed.

```
size = "Large";
break;
```

Here, the size variable is assigned with the value `Large`.

---

break statement in Java switch...case

Notice that we have been using `break` in each case block.

```
 ...
case 29:
  size = "Small";
  break;
...
```

The `break` statement is used to terminate the **switch-case** statement. If `break` is not used, all the cases after the matching case are also executed. For example,

```
class Main {
  public static void main(String[] args) {

    int expression = 2;

    // switch statement to check size
    switch (expression) {
      case 1:
        System.out.println("Case 1");

        // matching case
      case 2:
        System.out.println("Case 2");

      case 3:
        System.out.println("Case 3");

      default:
        System.out.println("Default case");
    }
  }
}
```

Output

Case 2 Case 3
Default case

In the above example, expression matches with `case 2`. Here, we haven't used the break statement after each case.

Hence, all the cases after `case 2` are also executed.

This is why the `break` statement is needed to terminate the **switch-case** statement after the matching case. To learn more, visit [Java break Statement]

---

default case in Java switch-case

The switch statement also includes an **optional default case**. It is executed when the expression doesn't match any of the cases. For example,

```
class Main {
  public static void main(String[] args) {

    int expression = 9;

    switch(expression) {

      case 2:
        System.out.println("Small Size");
        break;

      case 3:
        System.out.println("Large Size");
        break;

      // default case
      default:
        System.out.println("Unknown Size");
    }
  }
}
```

Output

Unknown Size

In the above example, we have created a **switch-case** statement. Here, the value of expression doesn't match with any of the cases.

Hence, the code inside the **default case** is executed.

```
default:
  System.out.println("Unknown Size);
```

Note: The Java switch statement only works with:

- [Primitive data types]: byte, short, char, and int
- [Enumerated types]
- [String Class]
- [Wrapper Classes]: Character, Byte, Short, and Integer.