Java Enums

‹ Previous Next ›

Enums

An `enum` is a special "class" that represents a group of **constants** (unchangeable variables, like `final` variables).

To create an `enum`, use the `enum` keyword (instead of class or interface), and separate the constants with a comma. Note that they should be in uppercase letters:

Example

```
enum Level {
  LOW,
  MEDIUM,
  HIGH
}
```

You can access `enum` constants with the **dot** syntax:

```
Level myVar = Level.MEDIUM;
```

Try it Yourself »

Enum is short for "enumerations", which means "specifically listed".

Enum inside a Class

You can also have an `enum` inside a class:

Example

```
public class Main {
  enum Level {
    LOW,
    MEDIUM,
    HIGH
  }

  public static void main(String[] args) {
    Level myVar = Level.MEDIUM;
    System.out.println(myVar);
  }
}
```

The output will be:

```
MEDIUM
```

Try it Yourself »

Enum in a Switch Statement

Enums are often used in `switch` statements to check for corresponding values:

Example

```
enum Level {
  LOW,
  MEDIUM,
  HIGH
}

public class Main {
  public static void main(String[] args) {
    Level myVar = Level.MEDIUM;

    switch(myVar) {
      case LOW:
        System.out.println("Low level");
        break;
      case MEDIUM:
         System.out.println("Medium level");
        break;
      case HIGH:
        System.out.println("High level");
        break;
    }
  }
}
```

The output will be:

```
Medium level
```

Try it Yourself »

Loop Through an Enum

The enum type has a `values()` method, which returns an array of all enum constants. This method is useful when you want to loop through the constants of an enum:

Example

```
for (Level myVar : Level.values()) {
  System.out.println(myVar);
}
```

The output will be:

```
LOW
MEDIUM
HIGH
```

Try it Yourself »

Difference between Enums and Classes

An `enum` can, just like a `class`, have attributes and methods. The only difference is that enum constants are `public`, `static` and `final` (unchangeable - cannot be overridden).

An `enum` cannot be used to create objects, and it cannot extend other classes (but it can implement interfaces).

Why And When To Use Enums?

Use enums when you have values that you know aren't going to change, like month days, days, colors, deck of cards, etc.

‹ Previous Next ›

Top Tutorials

HTML Tutorial
CSS Tutorial
JavaScript Tutorial
How To Tutorial
SQL Tutorial
Python Tutorial
W3.CSS Tutorial
Bootstrap Tutorial
PHP Tutorial
Java Tutorial
C++ Tutorial
jQuery Tutorial

Top References

HTML Reference
CSS Reference
JavaScript Reference
SQL Reference
Python Reference
W3.CSS Reference
Bootstrap Reference
PHP Reference

HTML Colors
Java Reference
Angular Reference
jQuery Reference

Top Examples

HTML Examples
CSS Examples
JavaScript Examples
How To Examples
SQL Examples
Python Examples
W3.CSS Examples
Bootstrap Examples
PHP Examples
Java Examples
XML Examples
jQuery Examples

Get Certified

HTML Certificate
CSS Certificate
JavaScript Certificate
Front End Certificate
SQL Certificate
Python Certificate
PHP Certificate
jQuery Certificate
Java Certificate
C++ Certificate
C# Certificate
XML Certificate