10.8 - What is Enum

In Java, an **enum** is essentially a special type of class that contains a fixed set of constants. Enums are often used in cases where we know all possible values at compile time. Examples include:

- Days of the week (SUNDAY, MONDAY, TUESDAY, etc.)
- Directions (NORTH, SOUTH, EAST, WEST)
- Status codes (RUNNING, FAILED, SUCCESS, etc.)

Enums are particularly useful for situations where a variable can only have one of a small set of predefined values.

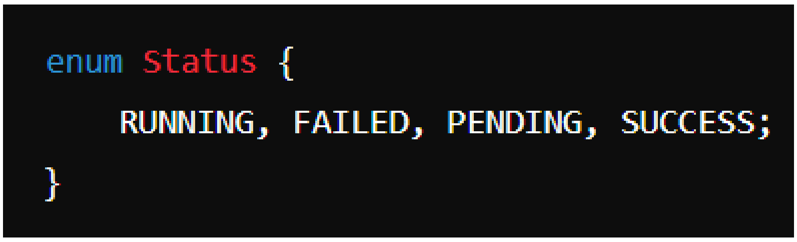
Real-world Example of Enums

Consider how websites return error codes like 404 Page Not Found or 500 Internal Server Error. These codes are constants used to indicate specific conditions, much like Java's enums. In Java, enums allow us to define custom constants for scenarios like:

- Days of the week
- Status codes
- Seasons (WINTER, SPRING, SUMMER, FALL)

Defining an Enum in Java

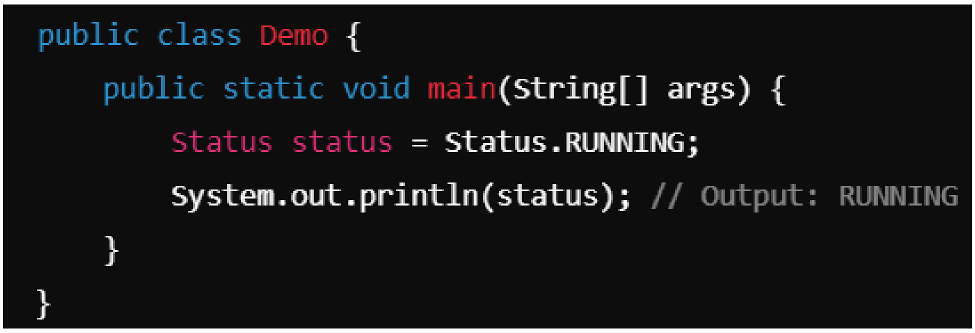
You can define an enum using the enum keyword. For example:



```java
enum Status {
    RUNNING, FAILED, PENDING, SUCCESS;
}
```

Here, the Status enum has four constants: RUNNING, FAILED, PENDING, and SUCCESS.

Example: Using Enums in Code



```java
public class Demo {
    public static void main(String[] args) {
        Status status = Status.RUNNING;
        System.out.println(status); // Output: RUNNING
    }
}
```

In this example, the status variable is of type Status (an enum), and it holds the value RUNNING. You can think of Status.RUNNING as an object representing the RUNNING constant.

Characteristics of Java Enums

- **Named constants:** Enum constants are implicitly public, static, and final. This means they are constants that cannot be changed.
- **Object-oriented:** Although enums look simple, they are full-fledged objects in Java. You can define constructors, instance variables, methods, and even implement interfaces in enums.
- **Indexing:** Enum constants have an implicit order, starting from 0. You can retrieve the index of an enum constant using the ordinal() method.

Methods in Enums

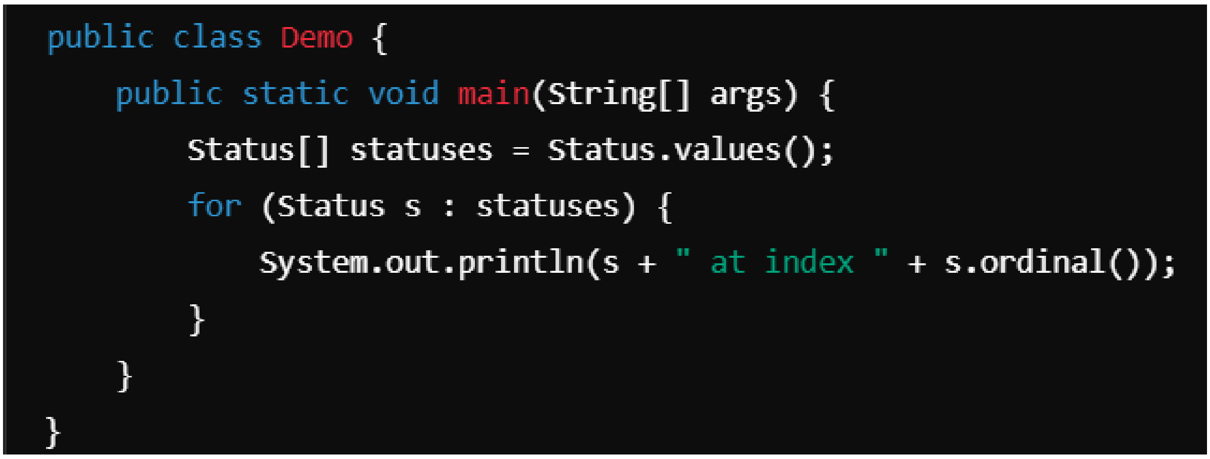
Java provides several useful methods for enums:

1. **ordinal() Method**
   This method returns the index of the enum constant. The first constant is assigned an index of 0, the second 1, and so on.

   System.out.println(Status.RUNNING.ordinal()); // Output: 0

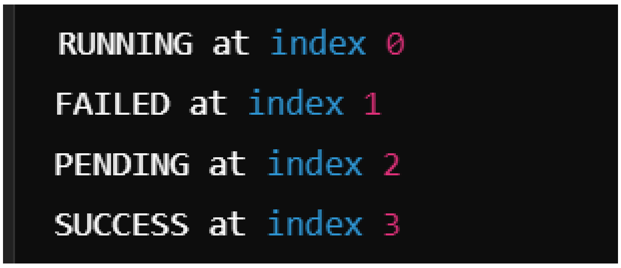
2. **values() Method**
   This method returns an array of all enum constants. It is helpful for iterating through all possible values.

Example:



```
public class Demo {
    public static void main(String[] args) {
        Status[] statuses = Status.values();
        for (Status s : statuses) {
            System.out.println(s + " at index " + s.ordinal());
        }
    }
}
```

Output:



```
RUNNING at index 0
FAILED at index 1
PENDING at index 2
SUCCESS at index 3
```

Conclusion

Enums in Java are powerful and versatile tools. They allow you to define a set of named constants, making code more readable and maintainable. With methods like **ordinal()** and **values(),** enums offer additional functionality that can be very useful when managing predefined constants.