**Enumerated Defined**

**Enumerated** means that something has a defined set of values it can hold. Think of the days of the week. Unless something drastic happens, we will always have Sunday through Saturday. These values could be stored in an enumerated (in Java, we'll use the key word **enum**) variable. They are unchanging and easily accessible.

In programming-speak, the values are **static** and **final**, which really means they can't be changed. Other data types, like int and float, could change during the course of a program.

Java enums are **type-safe**. There is no way to assign a value to the enum that doesn't fit with the defined values. For example, the DayOfWeek enum cannot accept anything other than the days of the week that were defined. If you try to assign a 1 to it, the code will not compile.

Additionally, an enum is a **reference type**, which means it behaves more like a class or an interface. As a programmer, you can create methods and variables inside the enum declaration.

## Syntax

All right, so how do we make use of an enum in Java? The following syntax declares an enumerated (enum) variable for the days of the week:

1. public enum DayOfWeek {
2. SUNDAY,
3. MONDAY,
4. TUESDAY,
5. WEDNESDAY,
6. THURSDAY,
7. FRIDAY,
8. SATURDAY;
9. }

# Java Enums

The **Enum in Java** is a data type which contains a fixed set of constants.

It can be used for days of the week (SUNDAY, MONDAY, TUESDAY, WEDNESDAY, THURSDAY, FRIDAY, and SATURDAY) , directions (NORTH, SOUTH, EAST, and WEST), season (SPRING, SUMMER, WINTER, and AUTUMN or FALL), colors (RED, YELLOW, BLUE, GREEN, WHITE, and BLACK) etc. According to the Java naming conventions, we should have all constants in capital letters. So, we have enum constants in capital letters.

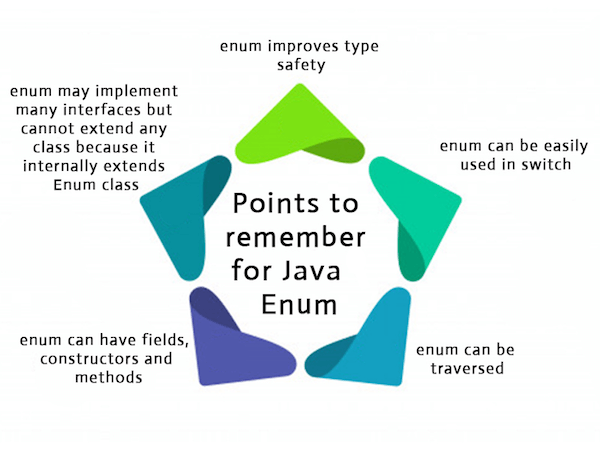
Java Enums can be thought of as classes which have a fixed set of constants (a variable that does not change). The Java enum constants are static and final implicitly. It is available since JDK 1.5.

Enums are used to create our own data type like classes. The **enum** data type (also known as Enumerated Data Type) is used to define an enum in Java. Unlike C/C++, enum in Java is more *powerful*. Here, we can define an enum either inside the class or outside the class.

Java Enum internally inherits the *Enum class*, so it cannot inherit any other class, but it can implement many interfaces. We can have fields, constructors, methods, and main methods in Java enum.

**Points to remember for Java Enum**

* Enum improves type safety
* Enum can be easily used in switch
* Enum can be traversed
* Enum can have fields, constructors and methods
* Enum may implement many interfaces but cannot extend any class because it internally extends Enum class



 class EnumExample1{

 //defining enum within class

 public enum Season { WINTER, SPRING, SUMMER, FALL }

 //creating the main method

 public static void main(String[] args) {

 //printing all enum

 for (Season s : Season.values()){

 System.out.println(s);

 }

 System.out.println("Value of WINTER is: "+Season.valueOf("WINTER"));

 System.out.println("Index of WINTER is: "+Season.valueOf("WINTER").ordinal());

 System.out.println("Index of SUMMER is: "+Season.valueOf("SUMMER").ordinal());



 }}