**Java Enum Important Points**

Below are some of the important points for Enums in Java.

1. All java enum implicitly extends java.lang.Enum class that extends Object class and implements [Serializable](https://www.journaldev.com/2452/serialization-in-java) and [Comparable](https://www.journaldev.com/780/comparable-and-comparator-in-java-example) interfaces. So we can’t extend any class in enum.
2. Since enum is a keyword, we can’t end package name with it, for example com.journaldev.enum is not a valid package name.
3. Enum can implement [interfaces](https://www.journaldev.com/1601/interface-in-java). As in above enum example, it’s implementing Closeable interface.
4. Enum constructors are always private. That’s why its being used to create singleton objects
5. We can’t create instance of enum using new operator.
6. We can declare [abstract methods in java](https://www.journaldev.com/1582/abstract-class-in-java) enum, then all the enum fields must implement the abstract method. In above example getDetail() is the abstract method and all the enum fields have implemented it.
7. We can define a method in enum and enum fields can override them too. For example, toString() method is defined in enum and enum field START has overridden it.
8. Java enum fields has namespace, we can use enum field only with class name like ThreadStates.START
9. Enums can be used in [switch statement](https://www.journaldev.com/588/java-switch-case-string), we will see it in action in the later part of this tutorial.
10. We can extend existing enum without breaking any existing functionality. For example, we can add a new field NEW in ThreadStates enum without impacting any existing functionality.
11. Since enum fields are constants, java best practice is to write them in block letters and underscore for spaces. For example EAST, WEST, EAST\_DIRECTION etc.
12. Enum constants are implicitly static and final
13. Enum constants are final but it’s variable can still be changed. For example, we can use setPriority() method to change the priority of enum constants. We will see it in usage in below example.
14. Since enum constants are final, we can safely compare them using “==” and equals() methods. Both will have the same result.