EnumSet and EnumMap

Before we move on, I want to talk about two more classes in the collections framework, specifically created to support enum types more efficiently.

You can use any List, Set, or Map, with an enum constant.

The EnumSet, and EnumMap, each has a special implementation that differs from the HashSet or HashMap.

These implementations make these two types extremely compact and efficient.

There's no special list implementation for enum types.

The EnumSet

The EnumSet is a specialized Set implementation for use with enum values.

All of the elements in an EnumSet must come from a single enum type.

The EnumSet is abstract, meaning we can't instantiate it directly.

It comes with many factory methods to create instances.

In general, this set has much better performance than using a HashSet, with an enum type.

Bulk operations (such as containsAll and retainAll) should run very quickly, in constant time, O(1), if they're run on an enumSet, and their argument is an EnumSet.

The EnumMap

The Enum Map is a specialized Map implementation for use with enum type keys.

The keys must all come from the same enum type, and they're ordered naturally by the ordinal value of the enum constants.

This map has the same functionality as a HashMap, with O(1) for basic operations.

The enum key type is specified during construction of the EnumMap, either explicitly by passing the key type's class, or implicitly by passing another EnumSet.

In general, this map has better performance than using a HashMap, with an enum type.

Two Types of EnumSet implementations

Enum sets are represented internally as bit vectors, which is just a series of ones and zeros.

A one indicates that the enum constant (with an ordinal value that is equal to the index of the bit) is in the set.

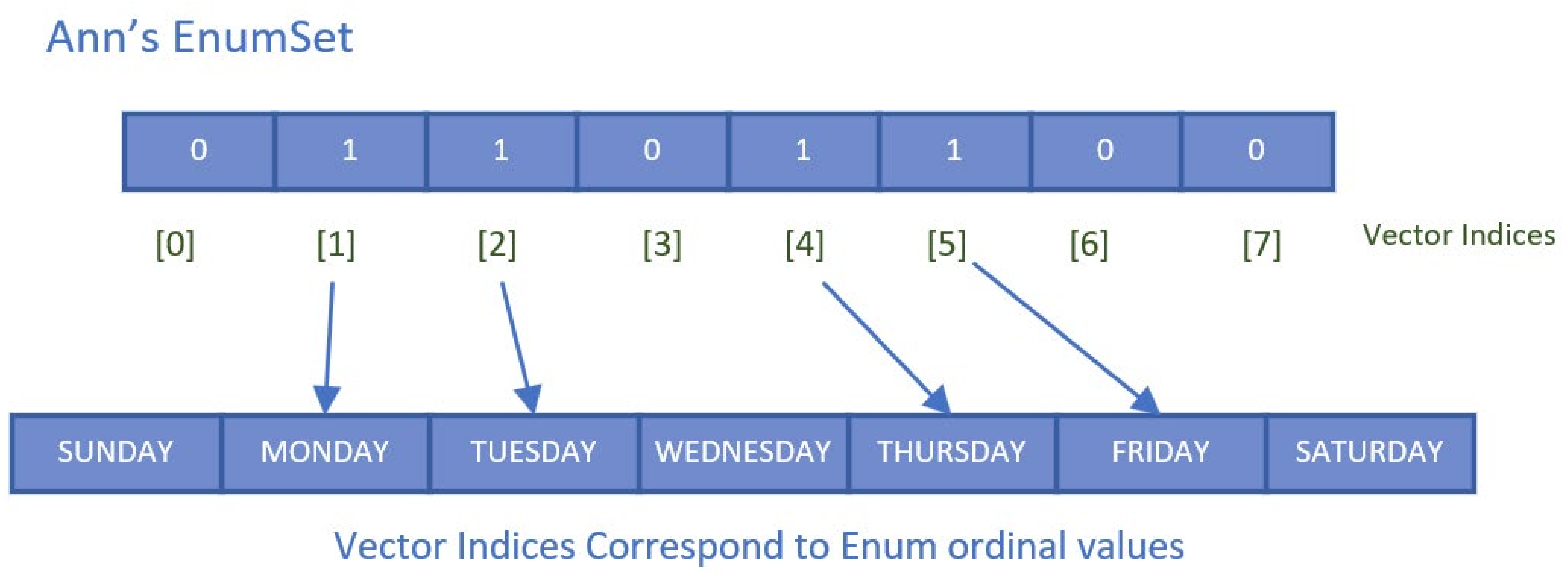
A zero indicates the enum constant is not in the set.

Using a bit vector allows all set operations to use bit math, which makes it very fast.

A RegularEnumSet uses a single long as its bit vector, which means it can contain a maximum of 64 bits, representing 64 enum values.

A JumboEnumSet gets returned if you have more than 64 enums.

Ann's Work Day EnumSet

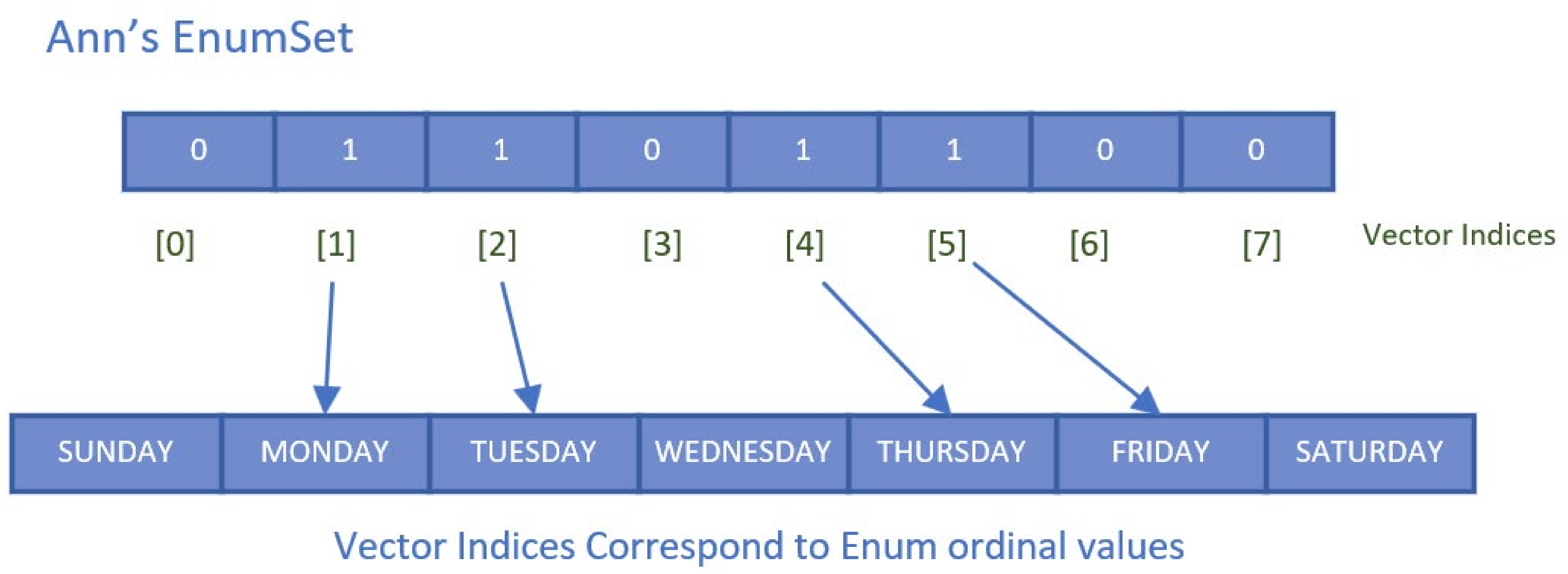


This slide is a visual representation of the EnumSet for Ann's work days.

It's size is 7, for the 7 possible values (based on the number of constants in the WeekDay Enum).

Any weekday that's part of her set, will be set to 1, at the index that corresponds to, the weekday ordinal value.

Ann's Work Day EnumSet



MONDAY has an ordinal value of 1 in our WeekDays enum, and the value in the underlying bit vector, at position 1, is a 1.

This means MONDAY is part of Ann's EnumSet.