Optional Class Introduction

In previous post we saw how to avoid null Pointer Exception using Optional<T> class. Optional class as written in Java docs is a container object. It may or may-not contain non-null value. If value is present then isPresent() method will return true and get() will return the value.

String str = ....

Optional<String> optional = ....

str can or cannot be null.

optional is never null. But it may or may not contain value (value is either present or empty).

Optional<T> class has private constructor so we cannot use new keyword to create object of Optional<T>.

There are few ways to create Optional<T> object.

* Optional<String> optional = Optional.*of*("Monday");
  + Returns Optional<String> with specified non-null value
  + Throws NullPointerException is value specified is null.
* Optional<String> optional = Optional.*ofNullable*(**null**);
  + *ofNullable* can accept null and non-null values.
  + If value passed is null then it returns empty Optional
  + If value passed is not null then *ofNullable* method calls *of(..)* method
* Optional<String> optional = Optional.*empty*();
  + *empty*() will return empty Optional corresponding to null.