Optional Class

Before we start discussion regarding Optional<T> class we will first understand what is the need for it.

Let us write a program to find a String in List<String> using startsWith(String str) method of String class. So we will write a method that will pick an element that starts with some prefix.

**public** **static** **void** find(List<String> days, String prefix) {

String found = **null**;

**for** (String str : days) {

**if** (str.startsWith(prefix)) {

found = str;

**break**;

}

}

**if** (found != **null**) {

System.***out***.println(found);

} **else** {

System.***out***.println("No Matches");

}

}

Above code works for the task that we defined. It will loop through the List<String> and search for prefix using startsWith(String str) method of String class. Once it is found it breaks the loop and displays it. If it is not found then we print No Matches.

What is the problem with above code? Well, first we need to make sure that String object found is to be made **null**. Second, we take imperative approach. I have written on [imperative and declarative](http://data-structure-learning.blogspot.com/2015/06/java-imperative-and-declarative.html) approach in Java. Third, we are using mutability all over the method.

For this problem we have written too much code. Let us again write this code. What do we have to do is just filter the collection using filter(..) method and first the first matching element. Else we need to be safe of not getting into NullPointerException. Let us write this in Java 8.

**public** **static** **void** find(**final** List<String> days, **final** String prefix){

**final** Optional<String> found=days.stream()

.filter(day->day.startsWith(prefix))

.findFirst();

**if**(found.isPresent()){

System.***out***.println(found.get());

}

**else**{

System.***out***.println("No Matches");

}

}

Using stream API we wrote a filter criteria and used findFirst() method. The findFirst() method returns Optional<T>. Then we used isPresent() method. It returns true is value in Optional<T> object is not null else it returns false.

This is how we can avoid NullPointerException.