Behavioral - Null Object Design Pattern

Genellikle yazığımız kodların içinde bir nesnenin null olup olmadığının kontrolünü hepimiz olcukça fazla yapıyoruz .Kendim için söyleyeyim null görünce kodun o kısmında ne zaman hata çıkacak diye beklerim genelde. Örneğin veritabanından herhangi bir nesne çekmeye çalıştığımızda önce gelen nesnenin boş olup olmadığını kontrol ederiz ardından boş değilse ona ait çeşitli işlemler gerçekleştiririz. Kodumuzun içinde bu tür null değerlerini fazlaca kullandığımızda hem gereksiz kod tekrarı ve hem hata eğilimi artar hemde kodumuzun okunulabilirliği oldukça azalır. Bu gibi durumlarda uygulayabileceğimiz çözüm için nesneye yönelik programlama dünyasında oldukça fazla kullanılan Null Object tasarım kalıbını kullanmaktayız

Customer class'ımız için base class;

```
abstract class CustomerBase{
   protected String name;
   public abstract boolean isNull();
   public abstract String getName();
}
```

Concrete class'lar

```
class RealCustomer extends CustomerBase{
   public RealCustomer(String newName) {
      this.name = newName;
   }
   @Override
   public boolean isNull() {
      return false;
   }
   @Override
   public String getName() {
      return name;
   }
}
class NullCustomer extends CustomerBase{
   @Override
   public boolean isNull() {
      return true;
   }
   @Override
   public String getName() {
      return "Not available in customer database";
   }
}
```

Burada null gelebilecek bir customer için NullCustomer adında bir class create ettim ve CustomerBase'den extends ettim.Sürekli null kontrolü yapmaktan kaçınmak için.Boilerplate kod ile 3 tane isim create ettim.

```
class CustomerFactory {
   public static final String[] names = {"Rob", "Eva","Angela"};
   public static CustomerBase getCustomer(String name){
      for (int i = 0; i < names.length; i++) {
         if (names[i].equalsIgnoreCase(name)){
            return new RealCustomer(name);
         }
      }
      return new NullCustomer();
   }
}
```

Main

```
public static void main(String[] args) {
    CustomerBase customer1 = CustomerFactory.getCustomer("Rob");
    CustomerBase customer2 = CustomerFactory.getCustomer("Eva");
    CustomerBase customer3 = CustomerFactory.getCustomer("Angela");
    CustomerBase customer4 = CustomerFactory.getCustomer("Luna");
    System.out.println(customer1.getName());
    System.out.println(customer2.getName());
    System.out.println(customer3.getName());
    System.out.println(customer4.getName());
}
```

```
"C:\Program Files\Java\jdk1.8.0_231\bin\java.exe" ...
Rob
Eva
Angela
Not available in customer database
```

Görüldüğü üzere Luna ismi array içerisinde yer almadığı için null nesne olduğu için bizim yazdığımız kodu döndürdü