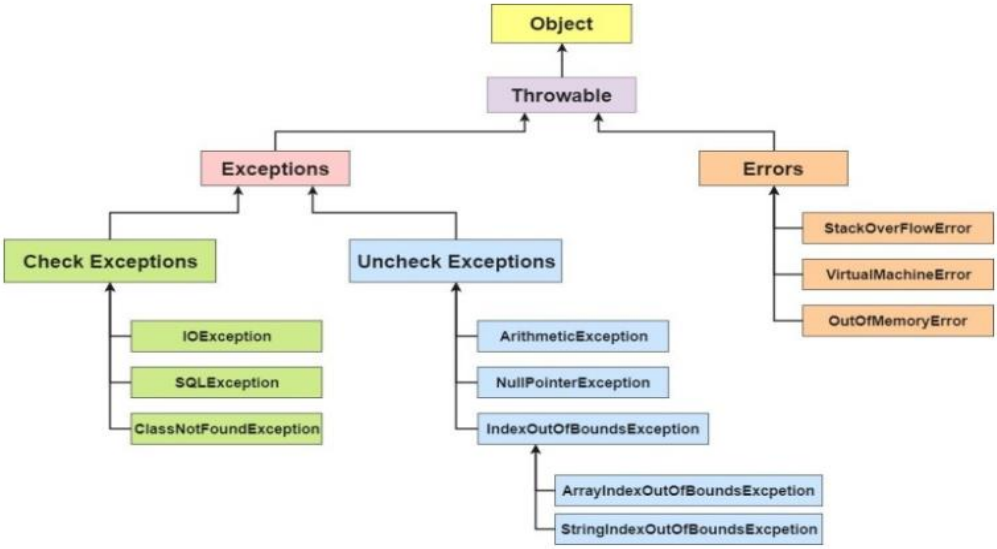
Selinay Altun

11)Exception Türleri

Exception, Türkçe anlamı istisnadır. Exception, kod akışını bozan ve programların yürütülmesi sırasında meydana gelen olaylardır.İki tür exception vardır.Kontrolsüz istisnalar(Unchecked Exceptions) ve Kontrollü istisnalar(Checked Exceptions).



Şekil 1:Exception

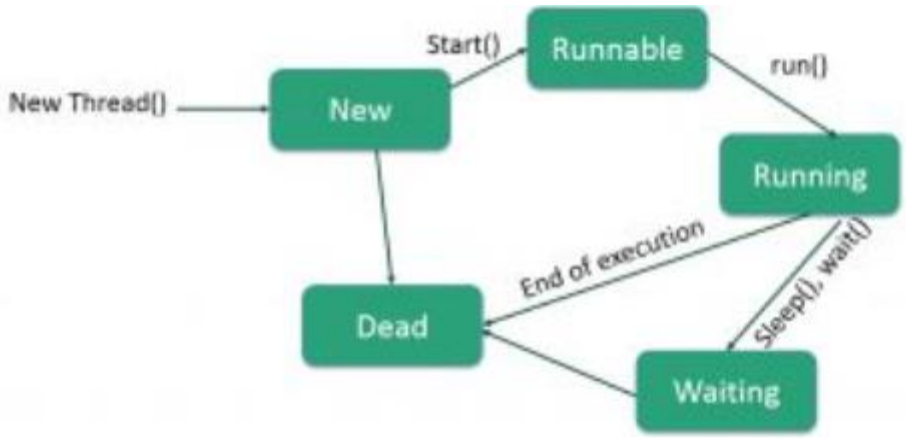
Kontrolsüz İstisnalar(Unchecked Exceptions) : Diğer adı Run-Time Exceptions.Bunun sebebi bu istisnalar çalışma anında meydana gelir.Önlem alınması şart değildir ancak alınırsa çok daha iyi olur. Örnek: ArrayIndexOutOfBoundsException(dizinin olmayan elemanına erişmek) Yani kontrolsüz kodlamadan dolayı meydana gelen istisna tipleridir.

Kontrollü İstisnalar(Checked Exceptions) : Diğer adı Derleme Anı İstisnaları da (Compile-Time Exceptions) .Bunun sebebi bu istisnalar sırasında derleme anında uyarılırız.Önlem alınması gereken ve handle edilene kadar derlenmeyecek istisnalardır.Örnek: FileNotFoundException (yerinde olmayan dosyaya erişmek) , Network (ağ bağlantısının kopması sebebiyle)

13)Thread

Türkçe anlamı iş parçacığıdır.Process'in bir den fazla işi aynı anda yapmasını sağlayan iş parçacıklarına thread denir.Thread, aynı anda sadece bir işi yapar ve processler birden fazla threade sahip olabilir.Processin birden fazla thread çalıştırmasına ise multithreading denir.

Java'da threadler ise aynı anda birden fazla iş yapmak için kullanılır.Multithread uygulamar geliştirerek işlemcilerin daha verimli ve performanslı kullanabiliriz.Günümüzde de birçok önemli sistemlerin alt yapısında ve sıralı işlemlerin önemli olduğu projelerde sıklıkla kullanılmaktadır.Java'da kullanılan iki farklı yöntemi vardır.java.lang.Thread sınıfının extend edilmesi ya da java.lang.Runnable interfacenin implement edilmesidir.



Şekil 2:Thread

- Thread sınıfının extend edilmesi için run methodu override edilir ve gerekli komutlar yazılır.
- Thread objesi Thread constructor'ına runnable edilebilecek bir parametre vererek yaratılabilir.
- RuntimeException fırlatılmaması için oluşturulan obje üzerinden bir kere start methodu çağırılmalıdır.
- Start() metodu çağırılmadan thread çalışmaz.
- Aynı thread sınıfı üzerinden birden çok obje oluşturulabilir.

Java'da multithread kullanıldığında senkronizasyon iyi yapılmalıdır,eğer iyi yapılmazsa beklenmedik sonuçlar ortaya çıkar ve yeterli verim alınmayabilir.

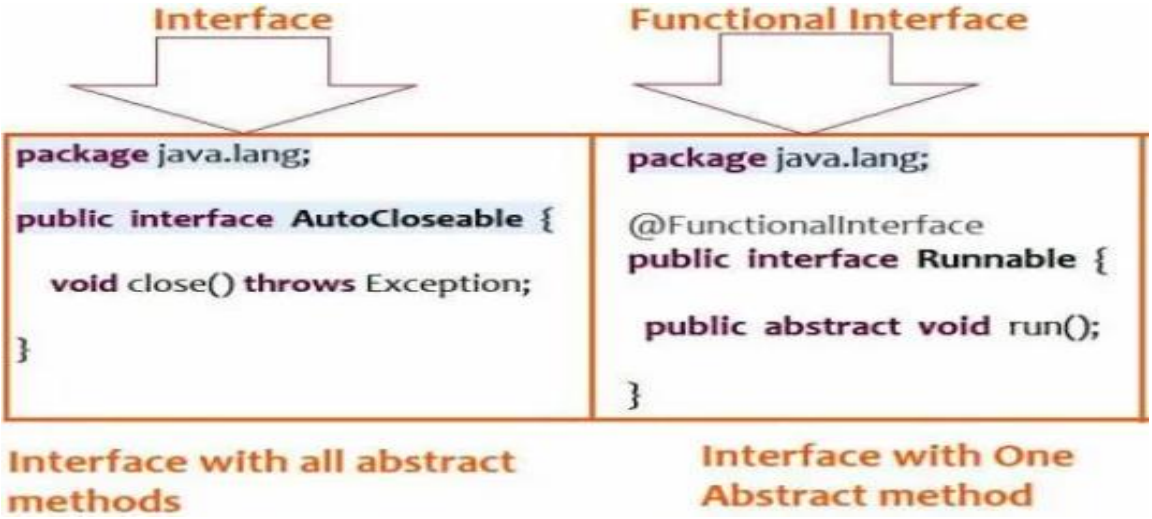
15)Static Method

Static anahtar kelimesini kullanan methodlardır.Bu methodlar sınıfa aittir ancak sınıftan oluşturulan objelere ait değildir.Static methodlar bir yerde çalışır ve sadece bir kere çalışır.Hafızada bir yerde tutulurlar.Static methodlar static olan dataları yönetir.
Static methodlardan doğrudan instance değişkenelere eişilmez.Erişilmek istenen değişkenin objesinin üretilmesi gerekmektedir.
Static methodların bir tane kopyası vardır. static method sadece static methodları çağırabilir.Staticde her nesne oluşmasında tekrar tekrar bellekte bu değişkenin yer tutması için belleğe başvurulmaz çünkü static değişkenin yeri sınıf oluşturulurken açılmıştır.

16) Functional Interface

Functional Interface, Java 8 ile gelmiş özelliklerden biridir. Bir interface'in functional olması için sadece bir adet abstract yani implemente edilmemiş metoda sahip olması yeterlidir. Eğer ilgili interface'in türetildiği interface'de abstract metot varsa bu durumda da Functional Interface olur. Bunun yanında istediği kadar default method barındırılabilir.

Lambda ifadesini kullanabilmemiz için functional interface ihtiyacımız vardır.Yani ,lambda expression'ların kullanılabilmesi için tanımlanırlar. java.util.Function paketinin altında bulunur ve doğrudan koda uygulanabilen yaklaşık 40'tan fazla işlevsel arabirim içerir.



Şekil 3:Functional

@FunctionalInterface anotasyonunun kullanılması zorunlu değildir ancak validasyon yapılması için kullanılabilir.Eğer anotasyon kullanılırsa birden fazla abstract method eklenmesi durumunda compile error verilir.

Thread'ler için kullanılan Runnable Interface'i de bir Functional Interface'dir. Lambda expression'ların kullanılabilmesi için de tanımlanırlar.

17) Lambda Expression

Java 8 ile gelen ve Java 8'in en büyük yeniliklerinden biri olan Lambda Expressions amacı, fonksiyonel programlamayı kolaylaştırıp daha az kod ile hızlı yazılım geliştirmeyi sağlamaktır. Lambda Expressions , Stream API pipeline ile çok çekirdekli ortamların paralel işlem özelliklerinden yararlanır. Lambda Expressions, parametre alabilen ve bir değer döndürebilen kod bloklarıdır.

Aslında Lambda Expressions, metotlara benzerler fakat bir isimleri olmasına gerek yoktur(anonim) ve metotların gövdesinde implement edilir.

Parametresiz tanımlanırsa; () -> expressions

Parametre ile tanımlanırsa tanımlanırsa; (birinciParametre, ikinciParametre) -> expressions

Lambda Expression'da dikkat edilmesi bir konu metodun isimsiz yani anonim bir şekilde oluşturulmasıdır.Aşağıda da bir örnek kod bulunmaktadır.

```
1  package lambdaExpressions;
2
3
4      interface Operation{
5          boolean mathOperation(int number);
6      }
7
8  public class LambdaExp {
9      public static void main(String[] args) {
10         Operation isOddNumber = (number) -> (number % 2) !=0;
11         System.out.println(isOddNumber.mathOperation(-3));
12
13     }
14
15 }
```

18) Predicate and Consumer,Supplier

Predicate, Türkçe olarak doğrulamak anlamına gelmektedir.Bağımsız değişkenin boolean değerli işlevini temsil eder. Java Stream'den gelen verileri filtreler. Predicate, bir argümanı kabul eden ve bir boole değeri döndüren bir test() methoduna sahiptir.Parametre alıp şarta bağlı olarak boolean sonuç döndürür.Gruplama ilemlerinde de kullanılmaktadır.

Consumer,Türkçe olarak tüketici demektir.Parametre alır ve işlem yapar.Bir sonuç döndürmez.Amacı kendisine verilen lambda ifadesini tüketmektir.Kendisine sağlanan girdiyi tüketir.accept() methoduna sahiptir.

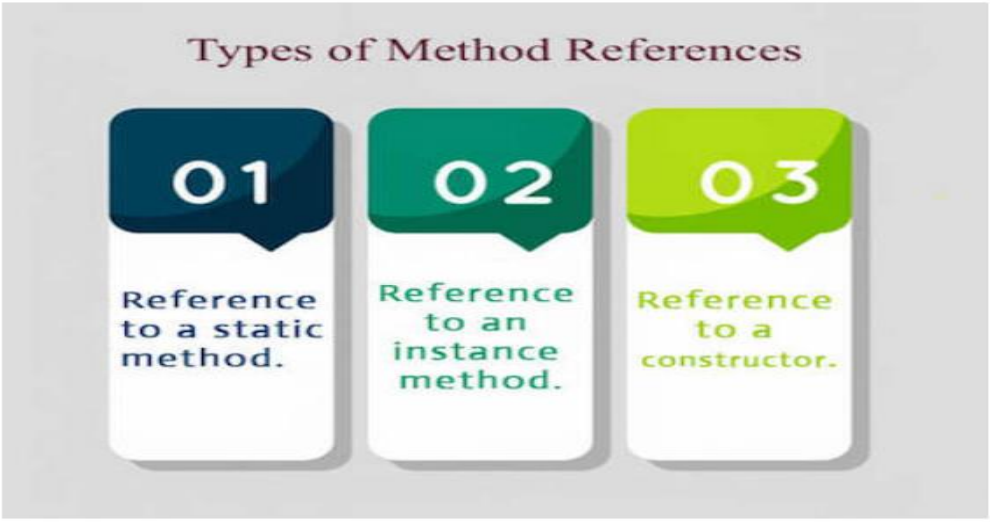
Supplier,Türkçe anlamı ihtiyacı karşılayan/satıcıdır. Consumer ile tam tersi işlem yapar. Parametre almaz ancak sonuç döndürür.get() methoduna sahiptir.Static yöntemleri yoktur.

Farklılıkları:

- Predicate static methoda sahiptir.
- Predicate default olarak and, or, negate methodlarına sahip ,Consumer andThen methoduna sahit ancak Supplier default methoda sahip değil.
- Single Abstract Method olarak Predicate: public boolean test(T t), Consumer: public void accept(T t),Supplier:public R get()

19) Method Reference

Bu özellik Java 8 ile gelmiştir.Method Reference, özel bir lambda ifadesi türüdür. Genellikle mevcut yöntemlere referansta bulunarak basit lambda ifadeleri oluşturmak için kullanılırlar.Zaten bir adı olan yöntemler için kompakt, okunması kolay lambda ifadeleridir. " :: " operatörü bu işlem için kullanılır.Çeşitli method reference kullanımları vardır. Static methods, Constructor gibi.



Static methodlar için:

```
List<String> messages = Arrays.asList("merhaba", "java", "bootcamp");
messages.forEach(StringUtils::capitalize); //capitalize is static method
```

Constructor için:

```
List<String> sehirler = Arrays.asList("Ankara", "Istanbul");
public Cities(String city) {
        this.city = city; }
sehirler.stream() .map(Cities::new)
```

20)Optional

NullPointerException hatasını önlemek amacıyla Java 8 ile gelen yeni özelliklerden biridir. Kodda oluşan NullPointerException hatalarını en aza indirmeyi sağlamayı amaçlayan bir yapıdır. java.util.Optional sınıfını, 15 method içermektedir.

Optional Class'ın birçok avantajı bulunmaktadır. Bunlardan bazıları kodda null kontrolünün yapılmasına gerek kalmamasıdır.Optional Sınıf sayesinde bu kontrol kolaylıkla yapılmaktadır.Bu da kolay ve okunaklı kod yazımını sağlamaktadır.Optional Class kullanımı oldukça kolay ve kullanışlı bir yöntemdir.

3 farklı şekilde nesne oluşturulabilir.empty, boş bir nesne oluşturur.
Örnek: Optional<Example> example= Optional.empty();

Of, null değer almayan nesne oluşturur.ofNullable ise null değer alabilen nesne oluşturur.

Bazı methodlar ve özellikleri aşağıdaki gibidir.

- isPresent(): Null kontrolü sağlar.
- orElse(): Bir şart ile istenilen nesne seçilebilir.
- orElseGet(): Nesnedeki null olan değere varsaydığımız değeri gönderebiliriz.
- ifPresent(): Eğer null değilse şartını sağlar.
- isPresent() : Optional türde olan bir nesnenin kontrolunü sağlar.
- map(): Nesne üzerinde işlemler yapılmasını sağlar.