

Constructor Nedir?

Constructor bir class'dan obje olusturuldugunda, ilk deger atamasini(initialize) yapmak icin kullanilir.

Java'yi en populer yazilim dillerinden biri yapan ozelliklerinin basinda OOP (Object Oriented Programming) olmasi gelir.

OOP konsept temel olarak birden cok olusturmak istedigimiz objeler icin once bir kalip olusturup(class), sonra bu kaliptan istedigimiz kadar obje uretmemizi



ve bu objeleri kullanarak istedigimiz uygulamayi yapabilmemizi ifade eder.



Constructor Nedir?

Constructor bir class'dan obje olusturuldugunda, ilk deger atamasini(initialize) yapmak icin kullanilir.

Ornegin bir okul projesi yapmak istedigimizde ogretmen, ogrenci, personel, ders gibi bir cok objelere ihtiyac vardir.





Ihtiyacimiz olan ogretmenleri olustururken her seferinde bir ogretmen icin hangi bilgileri girmek gerekir diye dusunmek yerine,

tum ogretmenlerin sahip olmasi gereken ortak ozellikleri bir class'da belirleyip, o class'dan istedigimiz kadar ogretmen objesi olusturabiliriz.

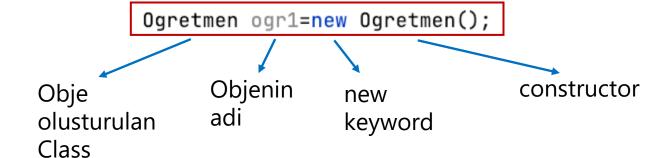
Constructor Nasil Kullanilir?

Constructor bir class'dan olusturulan objelere ilk degerlerini atamak icin new keyword'u ile birlikte kullanilir. Bir Ogretmen Class'i olusturup, ogretmenlerin ortak ozellikleri icin variable ve method'lar olusturalim.

```
public class Ogretmen {
    String isim="isim belirtilmedi";
    String soyisim="Soyisim belirtilmedi";
    String brans="brans belirtilmedi";
}
```

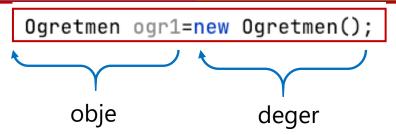
Artik proje kapsamindaki tum class'lardan ogretmen objesi olusturabiliriz.

```
Scanner scan=new Scanner(System.in);
ArrayList<String> ogrenciList=new ArrayList<>();
```



Constructor Nasil Kullanilir?

Constructor bir class'dan olusturulan objelere ilk degerlerini atamak icin new keyword'u ile birlikte kullanilir.



Deger atamasi yapmadan bir obje tanimlayabiliriz ama bu durumda bu objeyi kullanamayiz.

```
Ogretmen ogr1;
System.out.println(ogr1.isim);
```

Olusturdugumuz bir objeyi null pointer'a esitleyebiliriz.

```
Ogretmen ogr2=null;

System.out.println(ogr2.isim);
ogr2.isim="Ali Can";
```

Bu durumda Java Compile time'da size itiraz etmeyecektir ancak, objeyi kullanmak istediginizde, runtime'da NullPointerException verecektir.

Null pointer'a esitlediginiz objeyi kullanmadan deger atamaliyiz.

Constructor Nasil Kullanilir?

Constructor bir class'dan olusturulan objelere ilk degerlerini atamak icin new keyword'u ile birlikte kullanilir.

```
public class Ogretmen {
    String isim="isim belirtilmedi";
    String soyisim="Soyisim belirtilmedi";
    String brans="brans belirtilmedi";
}
```

Yukaridaki Ogretmen class'ini kullanarak bir obje olusturdugumuzda constructor Ogretmen class'inda belirlenmis olan degerleri objeye atayacaktir.

```
Ogretmen ogr1=new Ogretmen();
```

```
System.out.println(ogr1.isim); // isim belirtilmedi
System.out.println(ogr1.soyisim); // Soyisim belirtilmedi
System.out.println(ogr1.brans); // brans belirtilmedi
```

Eger objemizin farkli degerlere sahip olmasini isterseniz, yeni degerleri objeye atamalisiniz

```
ogr1.isim="Ali";
ogr1.soyisim="Can";

System.out.println(ogr1.isim); // Ali
System.out.println(ogr1.soyisim); // Can
System.out.println(ogr1.brans); // brans belirtilmedi
```

Constructor Nasil Olusturulur?

Constructor declaration olarak method'a cok benzer, ancak 2 temel ozellik MUTLAKA OLMALIDIR.

- 1- Ismi mutlaka Class ismi ile ayni olmalidir.
- 2- Return type'i OLMAMALIDIR.

```
public class Ogretmen {
    public Ogretmen(){
    }
    String isim="isim belirtilmedi";
    String soyisim="Soyisim belirtilmedi";
    String brans="brans belirtilmedi";
}
```

Access modifier bizim ihtiyaclarimiza gore public, protected, default veya private olarak belirlenebilir.

Access modifier constructor'in calismasi ile degil, constructor'a baska class'lardan erisimin duzenlenmesi ile ilgilidir.

Constructor'da () icerisine parametre yazilmasi da kodu yazanlarin tercihine birakilmistir, ihtiyaca gore yazilabilir.

Constructor Turleri?

1- Default constructor: Bir Class'ta obje olusturulabilmesi icin Java'nin olusturdugu parametresiz bir constructor'dir.

```
public class Ogretmen {
    String isim="isim belirtilmedi";
    String soyisim="Soyisim belirtilmedi";
    String brans="brans belirtilmedi";
}
```

Yukaridaki Ogretmen Class'inda gorunurde hic constructor olmamasina ragmen, her hangi bir class'da

```
Ogretmen ogr1=new Ogretmen();
```

Yazdigimizda, Ogretmen Class'indan bir obje olusturabiliriz.

Biz bir constructor olusturmadigimiz icin arka planda default constructor calisacak ve obje olusturulacaktir.

Default constructor da parametre olmadigi gibi constructor body'sinde de hic bir atama yoktur, dolayisiyla default constructor kullanilarak olusturulan objeler, Ogretmen class'inda ilk atanan degerlere sahip olurlar.

```
System.out.println(ogr1.isim); // isim belirtilmedi
System.out.println(ogr1.soyisim); // Soyisim belirtilmedi
System.out.println(ogr1.brans); // brans belirtilmedi
```

Constructor Turleri?

2- Parametresiz constructor : Default constructor ile ayni ozelliklerdedir.

Aradaki temel fark, birini biz olustururuz, digerini Java olusturur.

```
public Ogretmen(){
}
```

Default constructor, arka planda calisir ve constructor body'sinde kod yoktur. Dolayisiyla default constructor kullanilarak olusturulan objeler, kaynak class'daki temel ozelliklere sahiptirler.

Genelde parametresiz constructorlar'da da atama olmaz. Ama kod yazan kisilerin atama yapmasina mani birsey de yoktur.

```
public Ogretmen(){
    isim="Hasan";
    soyisim="Kul";
    brans="Matematik";
}
```

```
Ogretmen ogr2=new Ogretmen();

System.out.println("ogretmen 2 isim : "+ogr2.isim); // Hasan
System.out.println("ogretmen 2 soyisim : "+ogr2.soyisim); // Kul
System.out.println("ogretmen 2 brans : "+ogr2.brans); // Matematik
```

Constructor Turleri?

3- Parametreli constructor:
Constructor icinde bizim
parametre olarak gonderdigimiz
degerleri instance variable'lara
atama yaparak, bizi her bir deger
icin tek tek atama yapma
zorunlulugundan kurtarirlar.

```
public Ogretmen(String ism, String soyism, String brns){
   isim=ism;
   soyisim=soyism;
   brans=brns;
}
```

Esitligin solundaki isim, soyisim ve brans instance variable'lar, ism, soyism ve brns ise parametre olarak gonderdigimiz degerlerdir.

```
Ogretmen ogr3=new Ogretmen(ism: "Veli", soyism: "Cem", brns: "Kimya");
```

Boylece obje olustururken constructor'a girdigimiz Veli, Cem ve Kimya parametre olarak constructor'a gidip, constructor body'si icerisinde instance variable'lara atanarak, objenin ozellikleri olurlar

```
System.out.println("ogretmen 3 isim : "+ogr3.isim); // Veli
System.out.println("ogretmen 3 soyisim : "+ogr3.soyisim); // Cem
System.out.println("ogretmen 3 brans : "+ogr3.brans); // Kimya
```

this. Kullanimi

this. Javanin ozel anlam yukledigi bir keyword'dur.

Basina konuldugu variable'in instance variable oldugunu belirtmek icin kullanilir.

```
public Ogretmen(String ism, String soyism, String brns){
   isim=ism;
   soyisim=soyism;
   brans=brns;
}
```

Buyuk bir proje oldugunda parametre isimlerinin variable isimleri ile ayni olmasi, kodu anlamamiz acisindan bize kolaylik saglar.

```
public Ogretmen(String isim, String soyisim, String brans){
    isim=isim;
    soyisim=soyisim;
    brans=brans;
}

String isim="isim belirtilmedi";
String soyisim="Soyisim belirtilmedi";
String brans="brans belirtilmedi";
```

Ancak Java scope geregi, icinde bulundugumuz scope'da isim, soyisim ve brans variable'lari varken instance variable'lara gitmez

this. Kullanimi

this. Javanin ozel anlam yukledigi bir keyword'dur.

Basina konuldugu variable'in instance variable oldugunu belirtmek icin kullanilir.

Java bu karisikligin onune this. keyword kullanimi ile gecmistir.

```
public Ogretmen(String isim, String soyisim, String brans){
    this.isim=isim;
    this.soyisim=soyisim;
    this.brans=brans;
}

String isim="isim belirtilmedi";
String soyisim="Soyisim belirtilmedi";
String brans="brans belirtilmedi";
```

this. sayesinde esitligin solundaki variable'larin instance variable'lar oldugunu belirterek, constructor'in sorunsuz calismasi saglanmistir.