NESNE YÖNELİMLİ PROGRAMLAMA(Object Oriented Programming/OOP)

Öğr. Gör. Celil ÖZTÜRK
Marmara Üniversitesi
Teknik Bilimler Meslek Yüksekokulu

İçerik

- ✓Metotlar
- ✓Static Metotlar
- ✓Metot Aşırı Yükleme
- ✓Recursive Metotlar

Metotlar

- Metotlar, nesne tabanlı programlamanın en temel kavramlarındandır.
- Metotlar bir defa tanımlanır ve isimleri ile bir çok kez çağrılabilirler.
- Tekrar eden kod bloklarını azaltmak amacıyla kullanılırlar.
- Belli bir kod bloğunun belirli bir görevi gerçekleştirmek için bir araya getirilmesi…
- Programların küçük parçalara bölünmesi !
- C, C#,PHP gibi dillerde fonksiyon olarak isimlendirilirler.
- Kod okunurluğu açısından avantaj sağlarlar. (Readibility)
- Tasarımı genişletilebilir yapmada kullanılırlar.(Extendable)
- Tasarım Değiştirilebilir yapıda olmasını sağlarlar.

Metot Oluştururken Dikkat Edilmesi Gerekenler

- Metot isimleri sayı ile başlamaz.
- Metot isimleri küçük/büyük harf duyarlıdır.
- Metot isimlerinde boşluk bulunmaz.
- *Metot ismi görev ile aynı olmalıdır.

Metot Yapısı

Metot Yapısı

Anahtar_Kelimeler Dönüş Tipi Metot_adi (Metot Parametreleri)
{

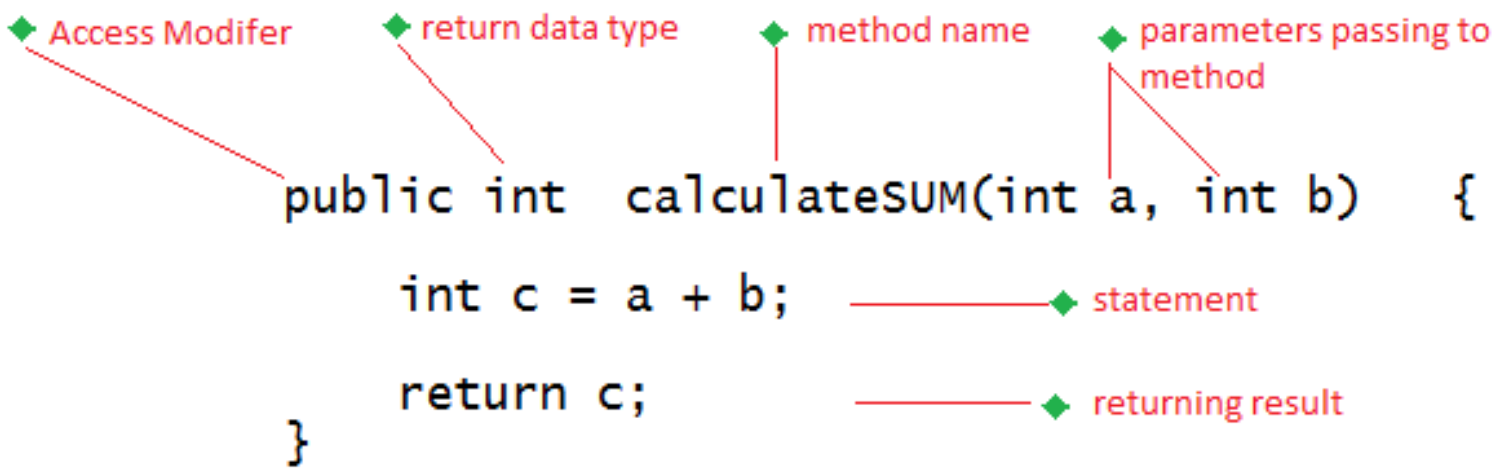
    Metot İçeriği

}

Metot Yapısı

- **Anahtar Kelimeler**:private,protected,public,static…
- **Dönüş tipi**: Metot görevi sonucunda sonuç döndürür.
- **Metot adi**: dışarıdan erişim sağlayabilmek için kullanılan isimdir.Her metot eğer aşırı yükleme(overloading) yapısı dışında ise isimler farklı olmalıdır.
- **Parametre**: Metodun yapacağı işlemler için metoda metot dışından sağlanan/gönderilen veriler. Eğer bir metot parametre almaz ise, parantezler arası boş olmalıdır.

Metot Yapısı



http://www.mainframegurukul.com/tutorials/java/Java-What-is-Class-Object.html

Erişim Belirteçleri(Access Modifier)

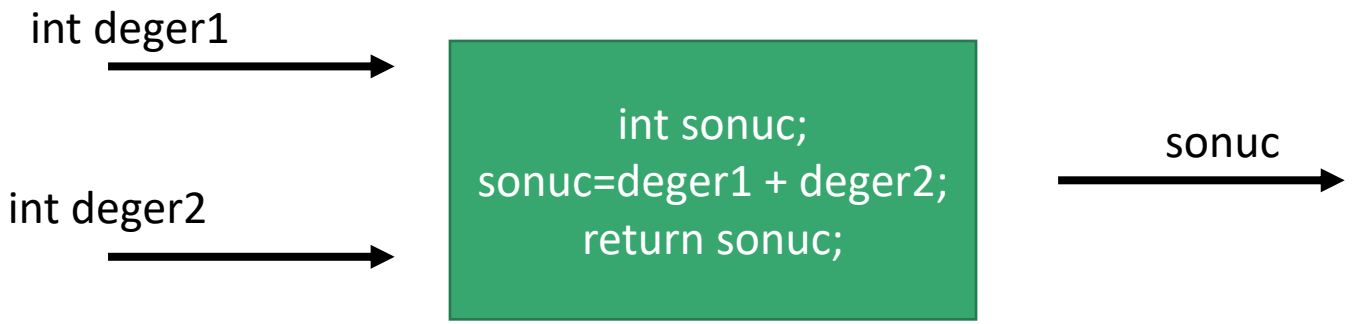
Erişim Belirteci	Açıklama
Public(UML → +)	Sınıf, değişken veya metota erişim sınırsızdır.
Protected (UML → #)	Sınıf, değişken veya metota erişim bulunduğu sınıf ve bu sınıftan türetilen sınıflar ile sınırlıdır.
Private (UML → -)	Sınıfa, değişken veya metota erişim ait olduğu sınıf ile sınırlıdır.

Metotlar

```
private int toplamaYap(int deger1,int deger2)
{
        int sonuc;
        sonuc=deger1 + deger2;
        return sonuc;
}
```

Metotlar

Toplama Yap Metodu Gösterimi



Metotlar

Parametre almayan(parametresiz) Metotlar

Giriş Yok →  **→Çıkış Yok. Mesaj vermek için kullanılır.**

```
public static void baslik()
{
        System.out.println("Marmar Üniv. ");
}

public static void main(String [] args)
{
        System.out.println("Baslik Çağrıldı.");
        baslik();
}
```

Metotlar

Parametre almayan(parametresiz) Metotlar

Giriş Yok →  **→Çıkış Var. Sabit sayılarda kullanılır.**

```
public static double mypi()
{
        return 3.14;
}
```

```
public static void main(String [] args)
{
        System.out.println("Metot Çağrıldı.");
        System.out.println(mypi());
}
```

Metotlar

Parametre alan Metotlar

Giriş var →  **→Çıkış Var.**

```
public static int topla(int x, int y)
{
        int z=x+y;
        return z;
}

public static void main(String [] args)
{
        int s1= 3;
        int s2=5;
        int z= topla(s1,s2);
        System.out.println(“toplam=“+z);
}
```

Metotlar

Parametre alan Metotlar

Giriş var →  **→Çıkış Yok.**

```
public static void topla(int x, int y)
{
        int c=x+y;
        System.out.println(''sonuc:''+c);

}

public static void main(String [] args)
{
        int s1= 3;
        int s2=5;
        topla(s1,s2);

}
```

Metotlar Örnekler

Metot Tanımlama

```
public static void toplama(int sayi1, int sayi2) {
    int sonuc = sayi1 + sayi2;
    System.out.println(sayi1 + " + " + sayi2 + " = " + sonuc);
}

public static void cikarma(int sayi1, int sayi2) {
    int sonuc = sayi1 - sayi2;
    System.out.println(sayi1 + " - " + sayi2 + " = " + sonuc);
}
```

Metot Çağırma

```
public static void main(String[] args) {
    System.out.println("Toplama");
    toplama(5, 6);
    System.out.println("Çıkarma");
    cikarma(10, 2);
```

Metotlar Örnekler

```
public static void buyuksayi(int a, int b) {
        if (a > b) {
            System.out.println("Büyük Sayı: " + a);
        } else if (a < b) {
            System.out.println("Büyük Sayı: " + b);
        } else {
            System.out.println("Sayılar Eşit");
        }
}
```

```
public static void main(String[] args) {
        buyuksayi(10, 6);
        buyuksayi(55, 96);
        buyuksayi(25, 25);
    }
}
```

Metotlar Aşırı Yükleme(Overloading)

- Metotların ayırt edilmeleri için, *sınıf içindeki metot isimlerinin birbirinden farklı olması gerekir.
- Metot Aşırı yükleme kavramı, özel bir durumdur.
- Aynı isimdeki metodun değişik sayıda veya tipteki parametreler ile çağrılabilmesidir.

Metotlar Örnekler Aşırı Yükleme(Overloading)

```
Public static void main (String args [])
{ int sonuc1=toplamaYap(15,20);
  int sonuc2=toplamaYap(12,14,17);
  double sonuc3=toplamaYap(2.11, 6.18);
}
Public static int toplamaYap(int deger1, int deger2)
{  return deger1+deger2; }

Public static int toplamaYap(int deger1, int deger2,int deger3)
{  return deger1+deger2+deger3; }

Public static double toplamaYap(double deger1, double deger2)
{  return deger1+deger2; }
```

Metotlar Örnekler Aşırı Yükleme(Overloading)

```
public class MainClass {

    public static int toplama(int sayi1, int sayi2) {
        int sonuc = sayi1 + sayi2;
        return sonuc;
    }

    public static double toplama(double sayi1, double sayi2) {
        double sonuc = sayi1 + sayi2;
        return sonuc;
    }

    public static void main(String[] args) {
        int tamSayi = toplama(5, 6);
        System.out.println("5 + 6 = " + tamSayi);

        double kusurluSayi = toplama(5.5, 6.5);
        System.out.println("5.5 + 6.5 = " + kusurluSayi);
    }
}
```

Static Metotlar

- Statik metotlar, tanımlandığı sınıfın yeni bir nesnesini oluşturmadan direkt olarak sınıfın adını referans göstererek çağrılabilir.

Metodları ana mainde iki şekilde çağırabiliriz ;

- Eğer metod static değilse :
- **Sınıfİsmi class = new Sınıfİsmi();**
  class.metod();

  Eğer metod static ise :
- **metod();**

Static Metotlar

```
public class Araclar
 {
    public static String depoGoster()
      {
          //Arac depo bilgisi göster
      }
   }
```

```
import metotpackage.metotkavrami.Araclar;
Public class StatikMetotOrnek
{
  public static void main(String[]args)
{
    Araclar.depogoster();
}
```

Metot Yapıları

```java
public class CrunchifyObjectOverriding {
    public static void main(String args[]) {
        Company a = new Company(); // Company reference and object
        Company b = new eBay(); // Company reference but eBay object
        a.address();// runs the method in Company class
        b.address();// Runs the method in eBay class
    }
}

class Company {
    public void address() {
        System.out.println("This is Address of Crunchify Company...");
    }
}

class eBay extends Company {
    public void address() {
        super.address(); // invokes the super class method
        System.out.println("This is eBay's Address...");
    }
}
```

Java Method Overriding Examples and Concepts: Overriding Rules

https://crunchify.com/java-method-overriding-examples-and-concepts-overriding-rules/

Yapıcı Metotlar()

- Yapıcı metotlar, **new** anahtar kelimesi aracılığıyla **yeni bir nesne üretildiğinde** otomatik olarak çalıştırılırlar.
- Yapıcı metodun adı, ait olduğu sınıf adıyla aynı olmak zorundadır.
- Yapıcılar ile bellekte nesneye yer ayrılır.
- Eğer yapıcı metot tanımlanmamış ise, derleme esnasında otomatik olarak parametre almayan gövdesi boş bir yapıcı metot oluşturulur.
- Private,static,final olamazlar.
- Oluşturulan bir nesneye, varsayılan değer atamalarında kullanılabilirler.

```
class Demo
{
     public Demo()
    {
         System.out.println("This is a default constructor");
    }
 }
```

Yapıcı Metotlar() //Parametresiz Constructor

```java
// Create a Main class
public class Main {
  int x;  // Create a class attribute

  // Create a class constructor for the Main class
  public Main() {
    x = 5;  // Set the initial value for the class attribute x
  }

  public static void main(String[] args) {
    Main myObj = new Main(); // Create an object of class Main (This will call the constructor)
    System.out.println(myObj.x); // Print the value of x
  }
}

// Outputs 5
```

https://www.w3schools.com/java/java_constructors.asp#:~:text=A%20constructor%20in%20Java%20is,of%20a%20class%20is%20created.

Yapıcı Metotlar() //Parametreli Constructor

```java
public class Main {
  int x;

  public Main(int y) {
    x = y;
  }

  public static void main(String[] args) {
    Main myObj = new Main(5);
    System.out.println(myObj.x);
  }
}

// Outputs 5
```

```java
public class Main {
  int modelYear;
  String modelName;

  public Main(int year, String name) {
    modelYear = year;
    modelName = name;
  }

  public static void main(String[] args) {
    Main myCar = new Main(1969, "Mustang");
    System.out.println(myCar.modelYear + " " + myCar.modelName);
  }
}

// Outputs 1969 Mustang
```

https://www.w3schools.com/java/java_constructors.asp#:~:text=A%20constructor%20in%20Java%20is,of%20a%20class%20is%20created.

Yapıcı Metotlar()

```
public class Hello { String name; //Constructor Hello(){ this.name =
"BeginnersBook.com"; } public static void main(String[] args) { Hpublic class
Hello {
   String name;
   //Constructor
   Hello(){
      this.name = "BeginnersBook.com";
   }
   public static void main(String[] args) {
      Hello obj = new Hello();
      System.out.println(obj.name);
   }
}
ello obj = new Hello(); System.out.println(obj.name); } }
```

https://beginnersbook.com/2013/03/constructors-in-java/

Yapıcı Metotlar()

```
Id: 10245 Name: Chaitanya
Id: 92232 Name: Negan
```

```java
public class Employee {

   int empId;
   String empName;

   //parameterized constructor with two parameters
   Employee(int id, String name){
       this.empId = id;
       this.empName = name;
   }
   void info(){
        System.out.println("Id: "+empId+" Name: "+empName);
   }

   public static void main(String args[]){
        Employee obj1 = new Employee(10245,"Chaitanya");
        Employee obj2 = new Employee(92232,"Negan");
        obj1.info();
        obj2.info();
   }
}
```

https://beginnersbook.com/2013/03/constructors-in-java/

Recursive(Yinelenen/Recursif/Özyinelenen) Metotlar

- Kendi kendini çağıran metotlardır.Bir dönüş kriteri bildirilene kadar metot kendi kendini çağırır.

```
import java.util.Scanner;


public class JavaOrnekleri {

    public static int Faktoriyel(int sayi)
    {
        if (sayi >= 1)
            return sayi * Faktoriyel(sayi - 1);
        else
            return 1;
    }

    public static void main(String[] args) {
        Scanner reader = new Scanner(System.in);
        System.out.print("Bir Sayı Girin:");

        int num = reader.nextInt();

        System.out.printf("%d Sayısının Faktöriyeli = %d \n", num, Faktoriyel(num));
    }
}
```

https://www.yazilimkodlama.com/java/java-recursive-metot-ile-faktoriyel-hesaplama/

*** Metot imzası(**method signature**) !!!

- Bir metodun imzası, metot adi ve parametre listesidir.(parametre tipleri ve adedi)

public static int max(int num1, int num2) →Metot imzası !

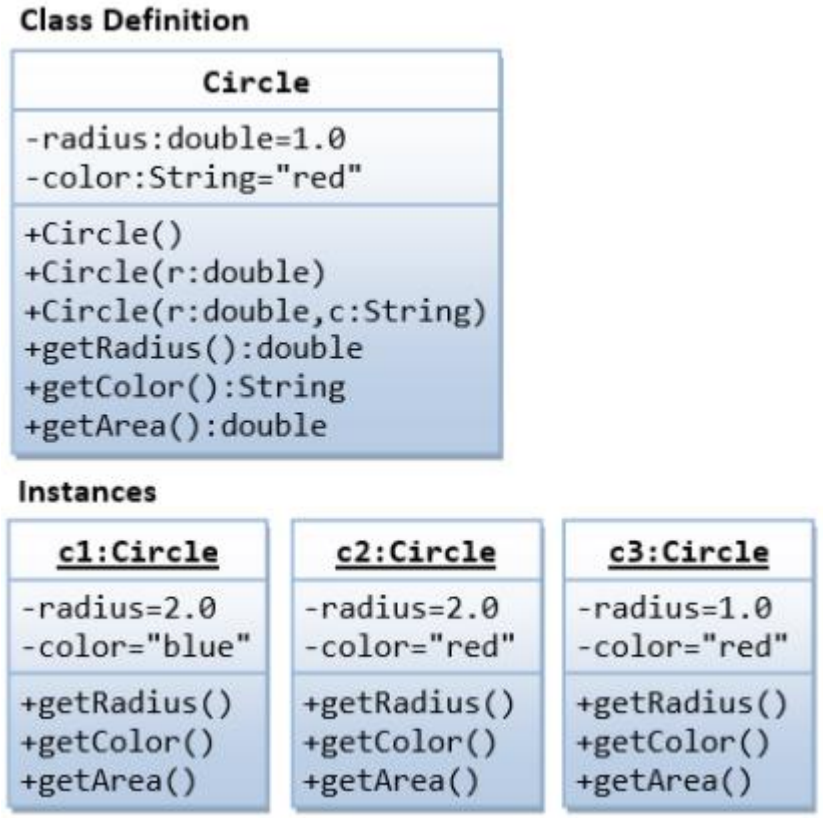
Method Signature Examples

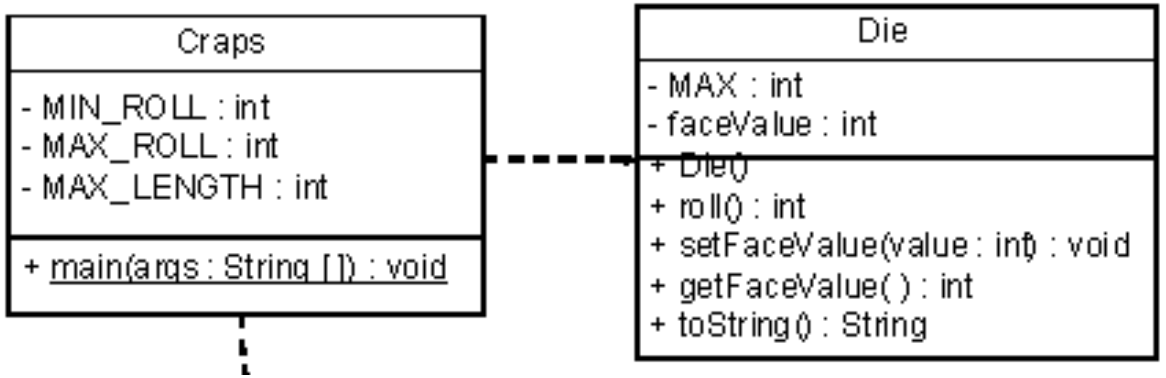
```
public void setMapReference(int xCoordinate, int yCoordinate)
{
//method code
}
```

The method signature in the above example is *setMapReference(int, int)*. In other words, it's the method name and the parameter list of two integers.

https://www.thoughtco.com/method-signature-2034235#:~:text=In%20Java%2C%20a%20method%20signature,name%20but%20accept%20different%20parameters.

Metot UML





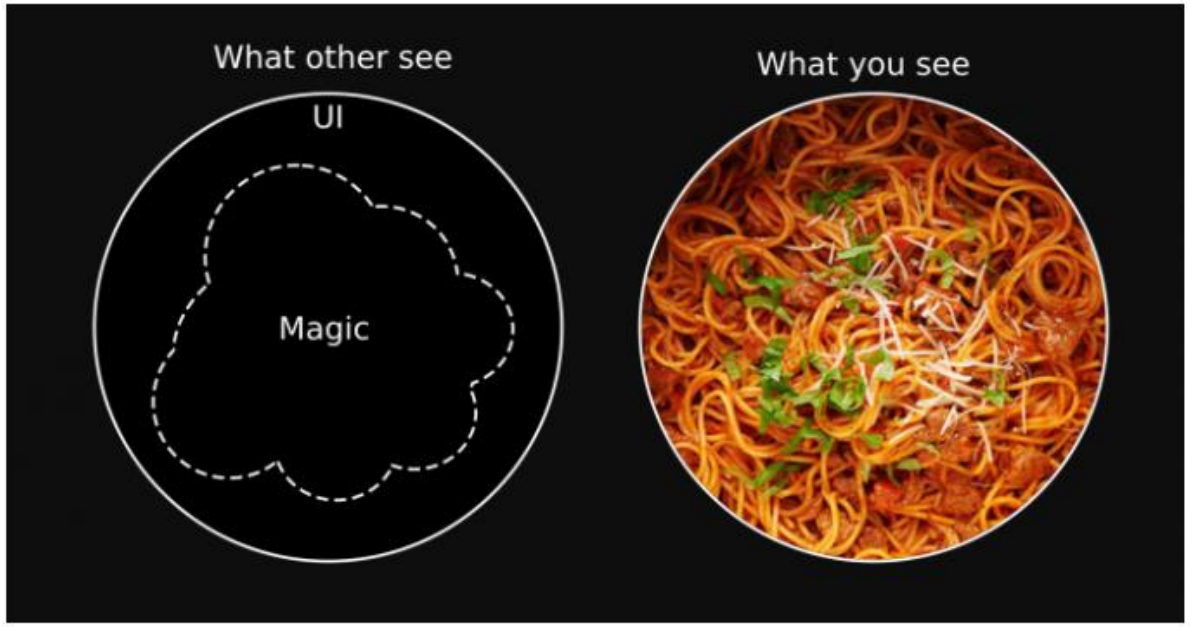
https://www3.ntu.edu.sg/home/ehchua/programming/java/J3a_OOPBasics.html

https://www.cs.umb.edu/~bobw/CS110/Project2/index.html

Metotlar

- **Sihirli Düğme (Magic PushButton) //To make everything Perfect…**
- **Spagetti Kodlama(Spaghetti Coding)**

Bakım ve değişiklik yapılamayacak kadar karmaşık kodlama…



Kaynaklar

- Java ve Java Teknolojileri, *Tevfik KIZILÖREN* – Kodlab Yayınları
- Dr Öğr. Üyesi Aysun Zehra ALTIKARDEŞ- Nesne Yönelimli Programlama Notları
- Dr. Öğr. Üyesi Erbil AKBAY – Nesne Tabanlı Programlama 2 Ders Notları
- https://medium.com/aykiri-yazilimcilar/kaliteli-yazılım-tasarımı-ve-anti-patternler-üzerine-notlar-a8f9ccfb6847/ Deniz Kılınç…
- https://www.mobilhanem.com/temel-java-dersleri-metot-yapisi/
- http://web.firat.edu.tr/iaydin/bmu111/bmu111_bolum7.pdf
- http://yazilimhanem.com/java-metotlar/