Constructors

A constructor is a special method that is used to initialize objects. The advantage of a constructor, is that it is called when an object of a class is created. It can be used to set initial values for fields, now the following example is that creates a constructor:

Example

```
class Clothes
{
   public String color;    // Create a field

   // Create a class constructor for the Clothes class
   public Clothes()
   {
      color = "yellow"; // Set the initial value for color
   }

   public static void main(String[] args)
   {
      Clothes myClothes = new Clothes(); // Create an object of the Clothes Class
      System.out.println(myClothes.color); // Print the value of color
   }
}
```

Note that the constructor name must match the class name, and it can NOT have a return type (like void or int) and that the constructor is called when the object is created.

All classes have constructors by default: if you do NOT create a class constructor yourself, Java creates one for you.

Constructor Parameters

Constructors can also take parameters, which is used to initialize fields.

Example

```
class Clothes
{
   public String color;

   // Create a class constructor with a parameter
   public Clothes(String color)
   {
      this.color = color;
   }

   public static void main(String[] args)
   {
      Clothes myClothes = new Clothes("yellow");
      System.out.println(myClothes.color);
   }
}
```

Explanation

The above example adds a String color parameter to the constructor. Inside the constructor we set this.color to color. When we call the constructor, we pass a parameter to the constructor ("yellow"), which will set the value of color to "yellow".

By the way, you can have as many parameters as you want:

Example

```
class Clothes
{
   public String size;
   public String color;
   public int price;

   // Create a class constructor with multiple parameters
   public Clothes(String size, String color, int price)
   {
```

```
    this.size = size;
    this.color = color;
    this.price = price;
  }

  public static void main(String[] args)
  {
    Clothes myClothes = new Clothes("M", "blue", 500);
    System.out.println("size: " + myClothes.size);
    System.out.println("color: " + myClothes.color);
    System.out.println("price: " + myClothes.price);
  }
}
```

Note that just like other methods, constructors can be overloaded by using different numbers of parameters.

Constructors Save Time

Constructors are very useful, as they help reducing the amount of code:

Without constructor:

```
class Clothes
{
  public String size;
  public String color;
  public int price;

  public static void main(String[] args)
  {
    Clothes myClothes1 = new Clothes();
    myClothes1.size = "M";
    myClothes1.color = "blue";
    myClothes1.price = 500;

    Clothes myClothes2 = new Clothes();
    myClothes2.size = "M";
    myClothes2.color = "pink";
    myClothes2.price = 580;
```

```
        System.out.println(myClothes1.size + " " + myClothes1.color + " " + myClothes1.price);
        System.out.println(myClothes2.size + " " + myClothes2.color + " " + myClothes2.price);
    }
}
```

With constructor:

```
class Clothes
{
    public String size;
    public String color;
    public int price;

    // Create a class constructor with multiple parameters
    public Clothes(String size, String color, int price)
    {
        this.size = size;
        this.color = color;
        this.price = price;
    }
}

class Program
{
    public static void main(String[] args)
    {
        Clothes myClothes1 = new Clothes("M", "blue", 500);
        Clothes myClothes2 = new Clothes("M", "pink", 580);

        System.out.println(myClothes1.size + " " + myClothes1.color + " " + myClothes1.price);
        System.out.println(myClothes2.size + " " + myClothes2.color + " " + myClothes2.price);
    }
}
```