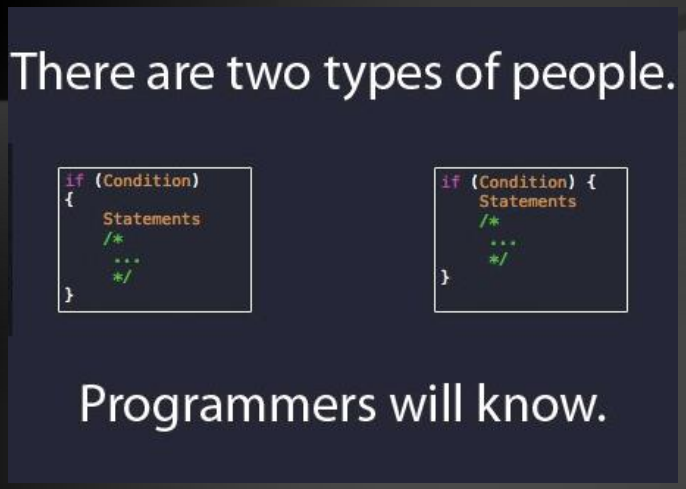
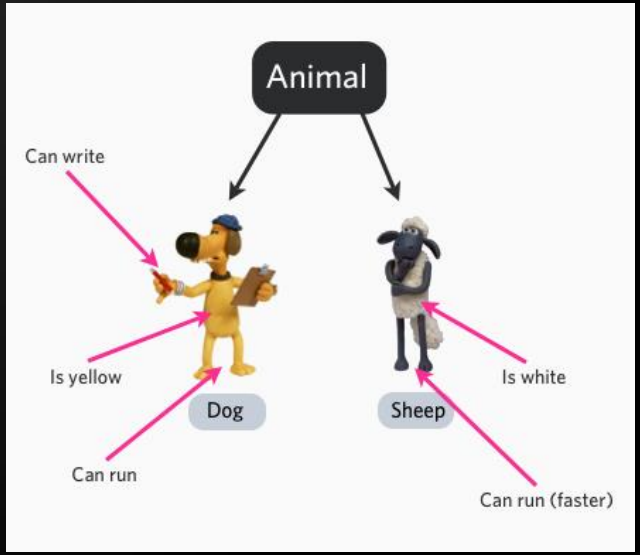
Telerik Academy



Java

Object oriented programming

Dimitar Ivanov
Telerik Software Academy
Learning & Development
http://academy.telerik.com



Telerik Academy

Table of Contents

- Procedural vs OOP
- Class or Object
- Class members
- Class example
- Fields - Declaration

Procedural vs Object Oriented Programming

Procedural vs OOP

- The main unit in procedural programming is function, and unit in object-oriented programming is a class
- Procedural programming concentrates on creating functions, while object-oriented programming starts from isolating the classes, and then follow the implementation of the methods inside of it.
- Procedural programming separates the data of the program from the operations that manipulate the data, while object-oriented programming focus on both of them.

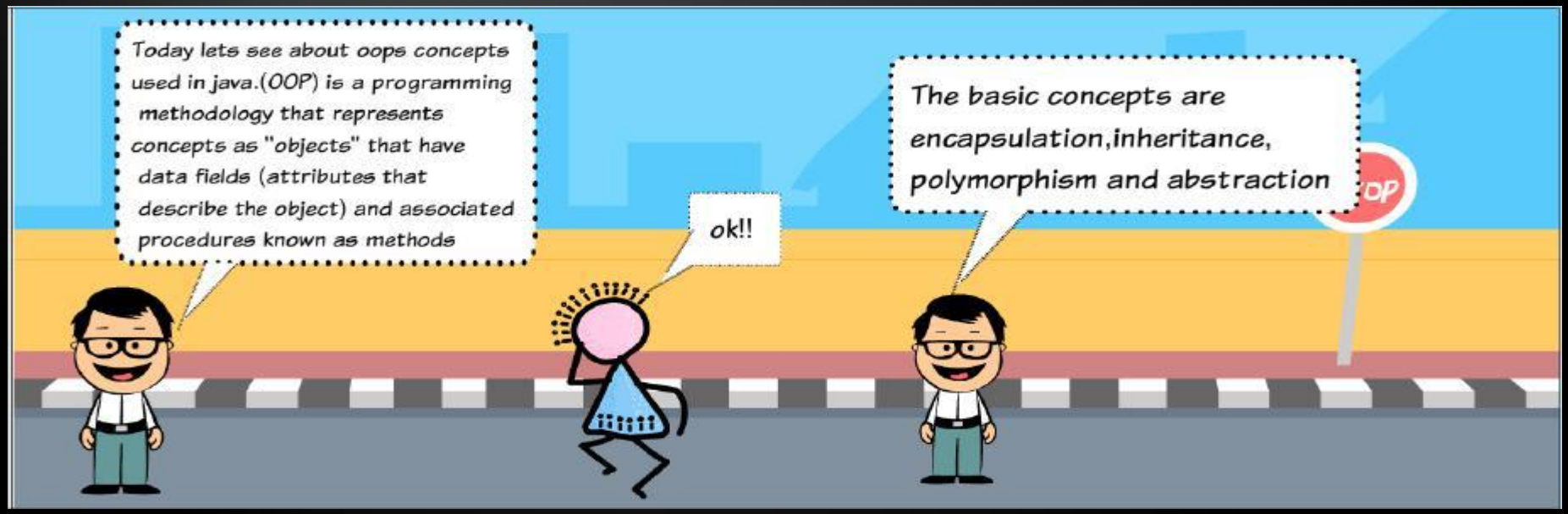
Class ? … Object ?

- "Class" refers to a blueprint. It defines the variables and methods the objects support
- "Object" is a instance of a class. Each object has a class which defines its data and behavior
- Example:
- LinuxConsole sudoBitch = new LinuxConsole();
- sudoBitch.makeASandwitch();

Telerik Academy

Class Members

- One class can have up to three members :
  - Fields: data variables which determine the status of the class or an object
  - Methods: executable code of the class built from statements. It allows us to manipulate/change the state of an object or access the value of the data members
  - Nested classes and nested interfaces



Class example

```
public class Book  {
        String color;
        String title;
        float price;

        public static long nextID = 0;

        public void setColor (String color)   {
                color = color;
        }
}
```

Fields - Declaration

- Field Declaration
  - A type name followed by the field name, and optionally an initialization clause
  - Primitive data type vs. Object reference
    - Boolean, char, byte, short, int, long, float, double
  - Field declarations can be preceded by different modifiers
    - Access control modifiers
    - Static
    - Final

Fields - Declaration

- Access control modifiers
  - **private**: private members are accessible only in the class itself
  - **package**: package members are accessible in classes in the same package and the class itself
  - ***protected***: protected members are accessible in classes in the same package, in subclasses of the class, and in the class itself
  - ***public***: public members are accessible anywhere the class is accessible

Fields - Declaration

- Static
  - only one copy of the static field exists, shared by all objects of this class
  - can be accessed directly in the class itself
  - from outside the class, non-static fields must be accessed through an object reference

Telerik Academy

Fields - Declaration

- Final
  - once initialized, the value cannot be changed
  - often be used to define named constants
  - static final fields must be initialized when the class is initialized
  - non-static final fields must be initialized when an object of the class is constructed

Telerik Academy

Java Overview

http://academy.telerik.com

Free Trainings @ Telerik Academy

- C# Programming @ Telerik Academy
  - csharpfundamentals.telerik.com

- Telerik Software Academy
  - academy.telerik.com

- Telerik Academy @ Facebook
  - facebook.com/TelerikAcademy

- Telerik Software Academy Forums
  - forums.academy.telerik.com