Fundamental concepts

- Class
- Object
- Instance
- method
- parameter
- signature

- Reading: *Chapter 1* of Objects First with Java - A Practical Introduction using BlueJ, © David J. Barnes, Michael Kölling
Objects and classes

• **Class**
  • A class represents a general category of things
  • e.g. Car, Bicycle, Student, Dog

• **Object**
  • Individual objects are created from a class
  • Objects represent ‘things’ from some real-world problem domain
  • e.g: “the red car down there in the car park”

• **Instance**
  • Each created instance of a class possesses its own copy of the characteristics of the class
Example - Student

• The set of all students forms the class Student
• A class describes features in common
• Each individual student is an object of the class Student
• John Smith and Janice Lee are instances of Student
Example - Dog

- The set of all dogs forms the class Dog
- Each individual dog is an object of the class Dog
- Spot, Rover, and Rex are all instances of Dog
Example – Restaurant

- Writing programs is largely about managing complexity.
- How is something complex organized in the real world?
- Consider a restaurant ...
  - Diners want meals
  - Chefs prepare dishes
  - Waiters take orders, and bring food to the tables
  - Busboys collect and wash plates
  - Barmen prepare and serve drinks
  - The maitre’d makes reservations and seats diners

Each type of person provides a narrow range of services. The restaurant involves the co-operative interaction of all the restaurant staff and clients.
A Restaurant

Jan

Chicken and bok choy, please

Bob

Your meal, madam

Ted

No. 14 and No. 67

Ding!

Diner
Waiter
Chef
What is a Waiter?

• A Waiter has the following actions
  • Bring menus
  • Take orders
  • Bring meals

• We can deal with any individual waiter, whether we have met them before or not, based solely on our knowledge of what things a Waiter can do
Review (1)

- **Class**
  - Objects are created from classes. The class describes the kind of object; the objects represent individual instantiations of the class.

- **Object**
  - Java objects model specific objects from the problem domain.

- **Instance**
  - Many similar objects can be created from a single class.
Review questions

• Can a class have several different objects? Discuss

• Can an object have several different classes? Discuss
Methods and parameters

- Objects have operations which can be invoked (Java calls them **methods**).
  - *Q: what methods can be invoked on a Circle object?*

- Methods may have **parameters** to pass additional information needed to execute.
  - *Q: Give some examples of methods from the Circle class and their parameters*

- The header of a method is called its **signature**. It provides information needed to invoke that method.
  - *Q: What is the signature of the changeSize method?*
## State

**circle1 : Circle**

- **private int diameter**: 68
- **private int xPosition**: 230
- **private int yPosition**: 130
- **private String color**: "blue"
- **private boolean isVisible**: true
Two circle objects

- **circle 1**: Diameter: 50, xPosition: 80, yPosition: 30, color: "blue", isVisible: true
- **circle 2**: Diameter: 30, xPosition: 230, yPosition: 75, color: "red", isVisible: true
Other observations

- Many *instances* can be created from a single class.
- The class defines what fields an object has, but each object stores its own set of values (the *state* of the object).

**Parameter**
- Methods can have parameters to provide additional information for a task.

**Signature**
- The header of a method is called its signature. It provides information needed to invoke that method.

**Types**
- Parameters have types. The type defines what kinds of values a parameter can take.
Source code

• Each class has source code (Java code) associated with it that defines its details (fields and methods).

• We will start looking inside source code next week
Return values

• All the methods in the `figures` project have `void` return types; but ...
• ... methods may return a result via a return value.
• Such methods have a non-`void` return type.
• More on this in the next chapter.
Exercise

- Download and open the shapes BlueJ project
- Create objects from the classes Circle, Square, Triangle
  - Right-click on the class and select `new(...)`
- Inspect the objects
  - Double-click on any object in the object bench
- Invoke their methods
  - Right-click on an object and select one of its methods
- Experiment!
  - Making mistakes is OK – it is a good way of learning to program!
Review (2)

• Method
  • We can communicate with objects by invoking methods on them. Objects usually do something if we invoke a method.

• Parameter
  • Methods can have parameters to provide additional information for a task.

• Type
  • Parameters have types. The type defines what kinds of values a parameter can take.

• Signature
  • The header of a method is called its signature. It provides information needed to invoke that method.
Review (3)

• State
  • Objects have state. The state is represented by storing values in fields.

• Method calling
  • Objects can communicate by calling each other’s methods.

• Source code
  • The source code of a class determines the structure and behaviour (the fields and methods) of each of the objects of that class.

• Result
  • Methods may return information about an object via a return value.
Review questions

• Write the signature for a method named AddStudent that has one parameter of type String and does not return a value.

• Write the signature for a method named max that has two parameters both of type int, and returns an int value.
Review these concepts

- Watch David Barnes’ Intro to Key Concepts here:

https://www.youtube.com/watch?v=CPUaTT0Xoo4&list=PLYPW4ErjcnzWB95MVvIKArO6PIfv1fHd&index=1