Objects First with Java

A Practical Introduction using BlueJ

David J. Barnes

Michael Kölling

Course Contents

- Introduction to object-oriented programming…
- … with a strong software engineering foundation…
- … aimed at producing and maintaining large, high-quality software systems.

Buzzwords

inheritance

responsibility-driven design

encapsulation

iterators

overriding

coupling

cohesion

interface

javadoc

mutator methods

collection classes

polymorphic method calls

Goals

- Sound knowledge of programming principles
- Sound knowledge of object-orientation
- Able to critically assess the quality of a (small) software system
- Able to implement a small software system in Java

Book

David J. Barnes & Michael Kölling

Objects First with Java
A Practical Introduction using BlueJ

Pearson Education, 2003
ISBN 0-13-044929-6.

Webpage

The course web page is at

www.cs.vt.edu/~cs1054/fall2003/

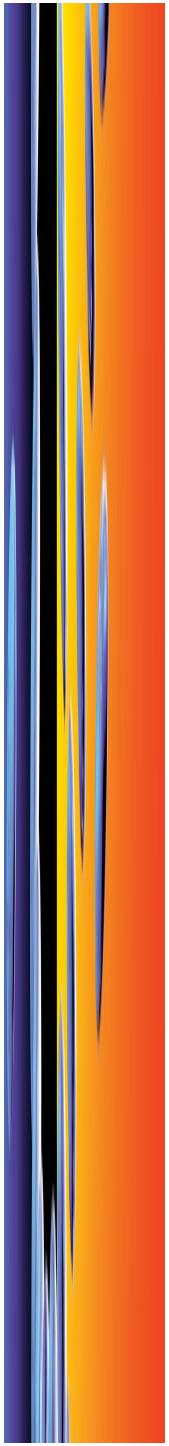
Please check it regularly.
It will be used for announcements and distribution of material.

Course overview (1)

- Objects and classes
- Understanding class definitions
- Object interaction
- Grouping objects
- More sophisticated behavior - libraries
- Well-behaved objects - testing, maintaining, debugging
- Designing classes

Course overview (2)

- Inheritance
- Polymorphism
- Extendable, flexible class structures
- Handling errors
- Designing applications



Demo

Fundamental concepts

- object
- class
- method
- parameter
- data type

Objects and classes

- objects
  - represent ‘things’ from the real world, or from some problem domain (example: “the red car down there in the car park”)
- classes
  - represent all objects of a kind (example: “car”)

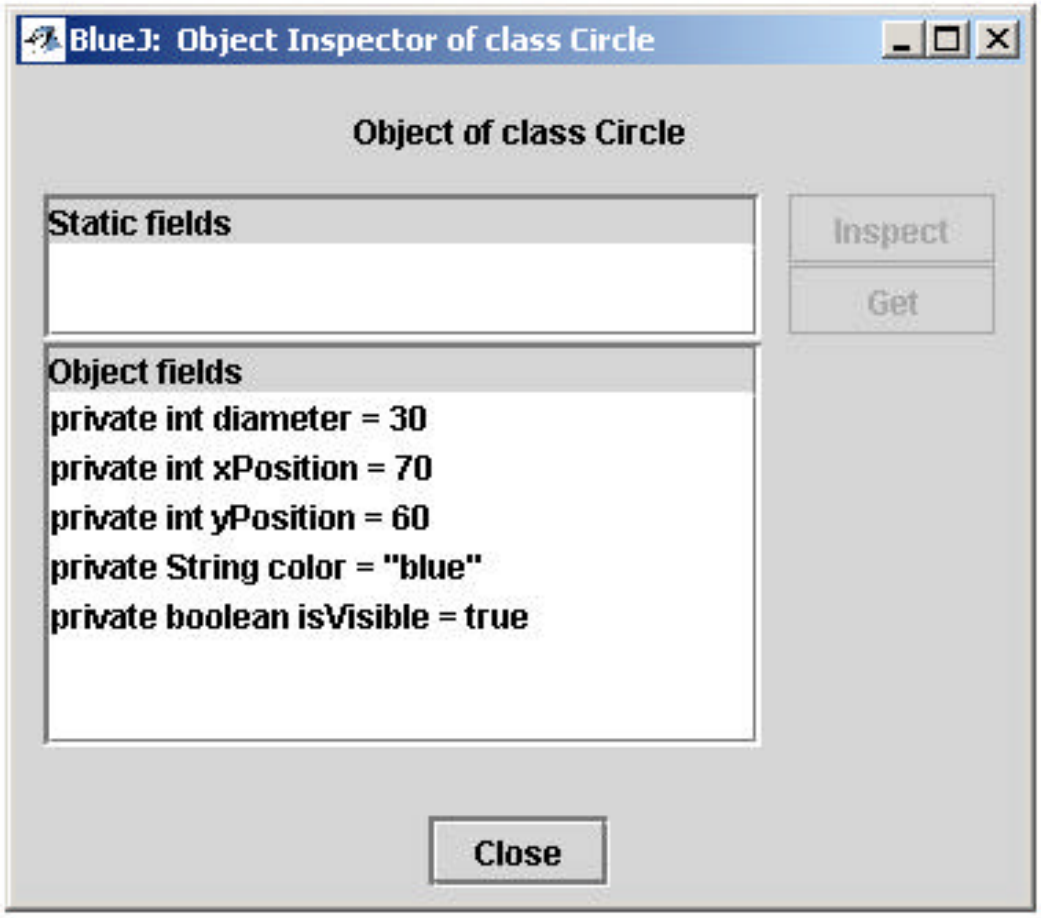
Methods and parameters

- objects have operations which can be invoked (Java calls them *methods*)
- methods may have parameters to pass additional information needed to execute

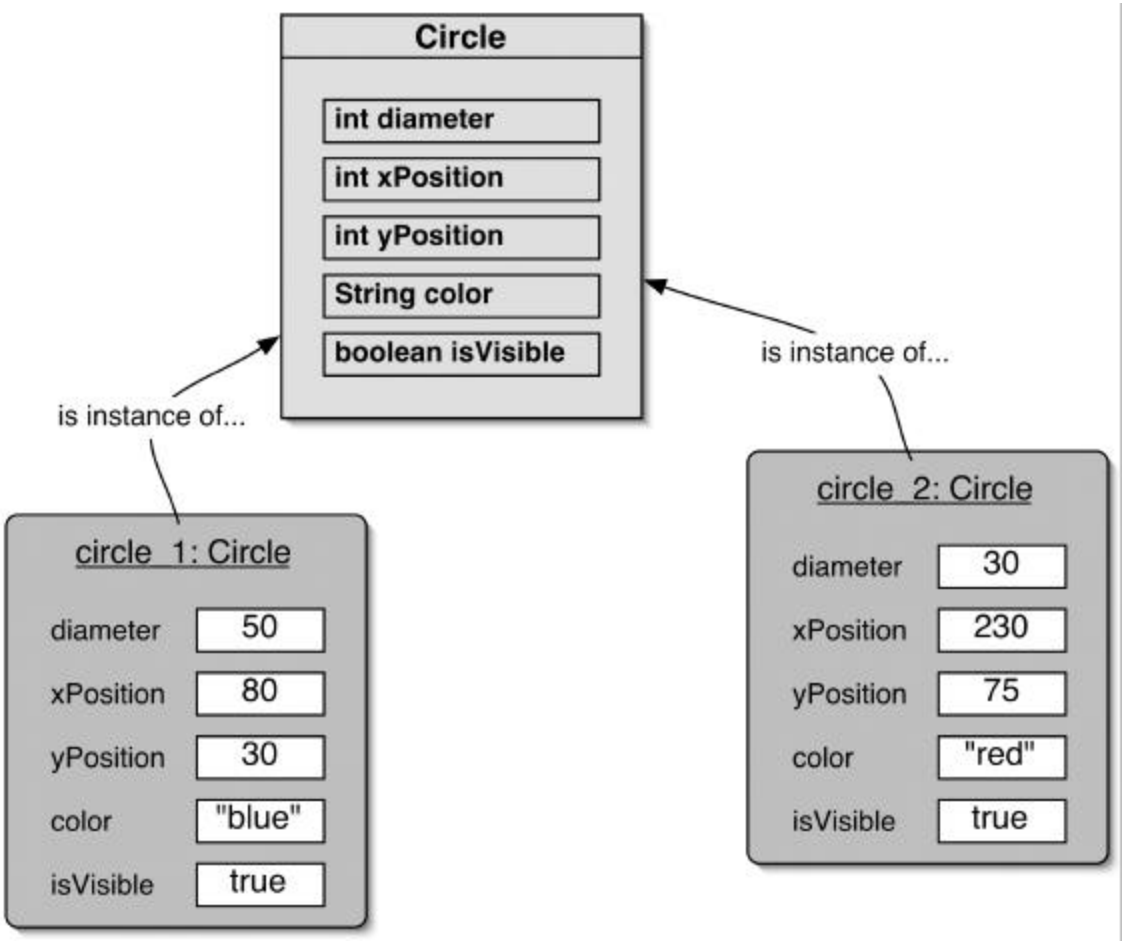
Other observations

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

State


BlueJ: Object Inspector of class Circle
Object of class Circle
Static fields
Inspect
Get
Object fields
private int diameter = 30
private int xPosition = 70
private int yPosition = 60
private String color = "blue"
private boolean isVisible = true
Close

Two circle objects



Source code

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

Return values

- Methods may return a result via a return value.