Objects and Classes
Lecture 1

Waterford Institute of Technology

March 26, 2014

John Fitzgerald

Fundamental Programming

Course Content

Course comprises

- Lectures
- Labs
  - Exercises
- Assignments
  - Closely related to Exercises

Waterford Institute *of* Technology

INSTITIÚID TEICNEOLAÍOCHTA PHORT LÁIRGE

Course Content

Lectures

Lectures

- Closely based on textbook
  Objects First with Java
  (3rd & 5th editions).
- Cover core subset Java language
  - Lecture introduces new material
  - Gist of lecture applied in lab

Waterford Institute *of* Technology
INSTITIÚID TEICNEOLAÍOCHTA PHORT LÁIRGE

Course Content

Labs

Labs

- Closely coupled to lectures
- Lab topic selection influenced by
  - *Objects First* textbook
  - Oracle's *The Java Tutorials*
  - Udacity *Intro to Programming*
  - Future ICTSkills web dev labs
  - Archived materials

Waterford Institute *of* Technology
INSTITIÚID TEICNEOLAÍOCHTA PHORT LÁIRGE

Programming

Learning tips

Learn to program by

- Reading supplied code
- Writing your own code

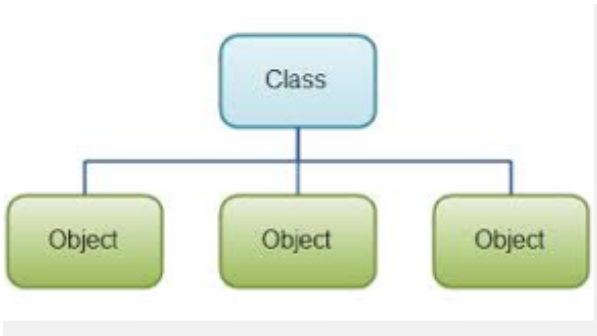
Complete all labs

- Read the supplied lab code
- Complete exercises
- Occasionally work in pairs
- Interact with colleagues

Object Oriented Programming (OOP)

What is OOP?

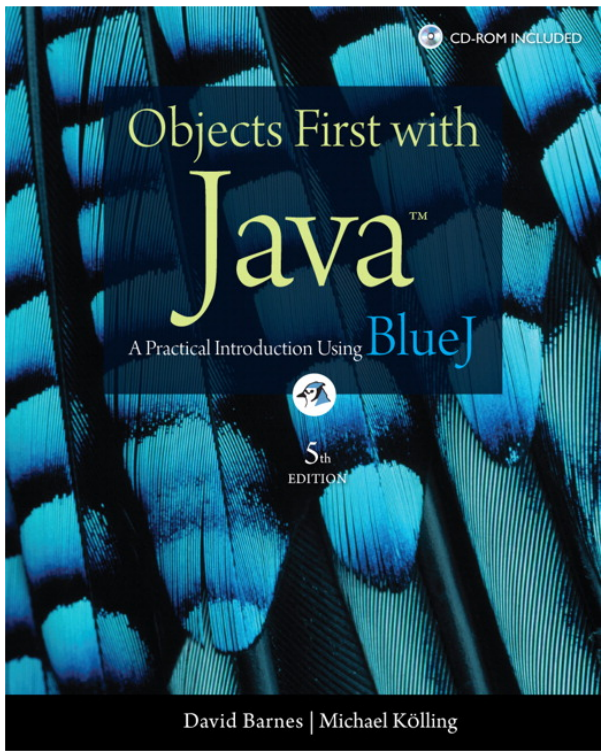
- A fundamental programming style
  - Represents concepts as objects
  - Objects: created from classes
  - Objects: contain data
  - Objects: have methods
  - Methods: can perform actions
  - Data: define state of each object
  - Methods: invoked or called



Applying Object Oriented Programming

Tools employed

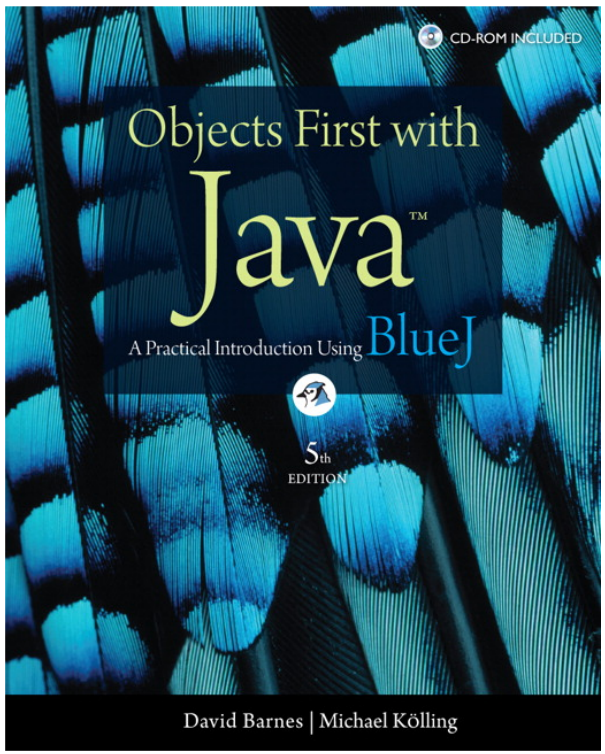
- Language: Java
  - Brief history
  - Popularity of language
  - Languages strengths and weaknesses



Applying Object Oriented Programming

BlueJ

- Development environment: BlueJ
  - What is BlueJ?
  - Brief history
  - Strengths and weaknesses



Java Programming Language

Selected facts

- Developed at Sun Microsystems
- Released in 1995
- Oracle Corporation buys Sun (2010)
- Compiled to bytecode
- Runs on Java Virtual Machine (JVM)
- Computer-architecture independent
- One of most widely used languages

Integrated Development Environment (IDE)

What is an IDE?

- A software application
- Purpose: assistance to software developers
- Example IDEs
  - Visual C++
  - Eclipse
  - NetBeans
  - DrJava
  - BlueJ

IDE Components

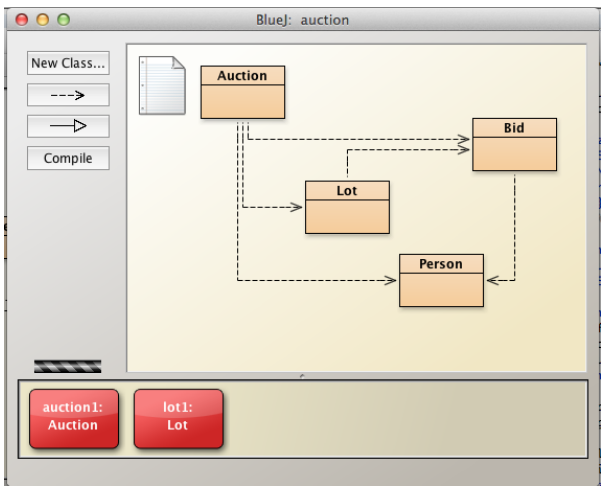
Typically included

- Text editor
- Build tools
- Debugger
- Unit testing
- Intelligent code completion
- Version control

BlueJ IDE

Description

- IDE for Java programming
- Designed for education
- Feasible for small software projects
- Fundamentally different to other IDEs
  - Graphical emphasis
  - Objects first
  - Incremental changes to existing code
  - Code samples represent realistic problems
  - Avoids starting with blank page



OOP teaching

Traditional approach

- Complex IDEs
  - Eclipse
  - NetBeans
- Large text books (1000 pages +)
- Begin with blank page
- Immediate introduction to many complex concepts
- OO concepts introduced several weeks into course

OOP teaching

BlueJ approach

BlueJ guidelines

- Simple IDE: BlueJ
- *Objects First with Java* compact textbook
- Begin with existing code
- Gradual introduction new concepts
- Immediate introduction of objects

Classes and Objects

Example of class

- Abstract description or specification of some entity
- Example
  - Car with following specification:
    - Make
    - Model
    - Color

Example of object

- Specific instance of an entity defined by its class
- Example: car object
  - Make: VW
  - Model: Golf
  - Color: red

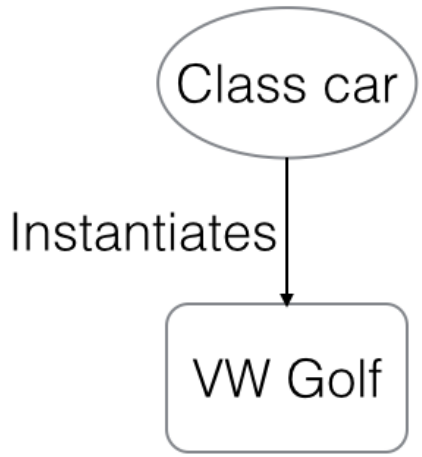
Concepts

Class

- Describes, defines or specifies objects
- Car class broadly descriptive
  - Actual make, model not known to class

Car object clearly specified

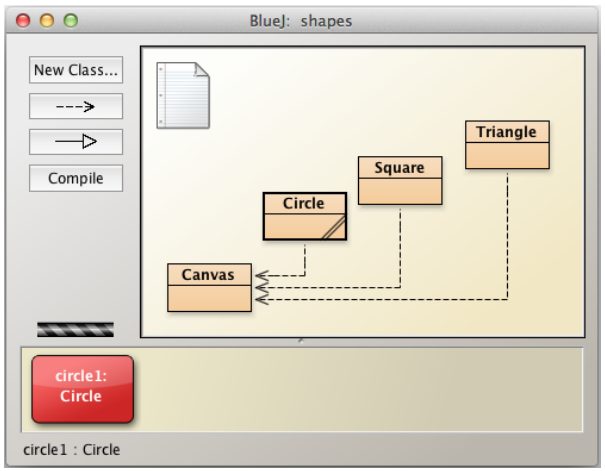
- Object created in conformance with class specification
- Exact make, model known



Create objects with BlueJ

Open Shapes project in BlueJ

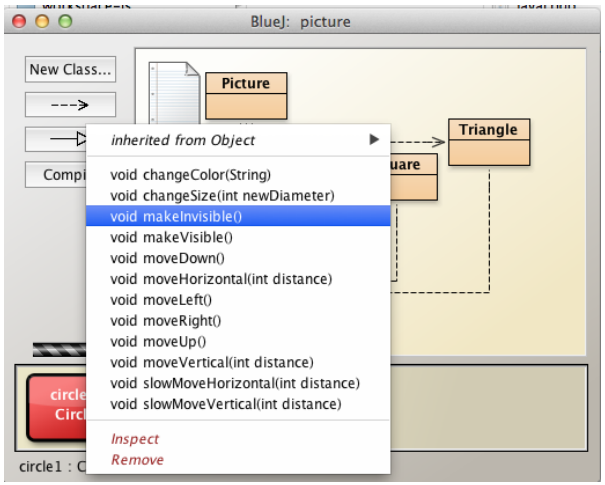
- Select Circle class & right click
- Choose new Circle
- Accept default name
- Circle object now on object workbench



Method invocation

Still working with Shapes project

- Right click on Circle object
- Select makeInvisible
- Object disappears
- Select makeVisible
- Object reappears



Method description

What is a method?

- A program within the class
- Methods
  - Perform actions
  - Can optionally return data
- Circle method actions:
  - Make circle object appear
  - Make circle object disappear
  - Change color circle object
  - Change size circle object

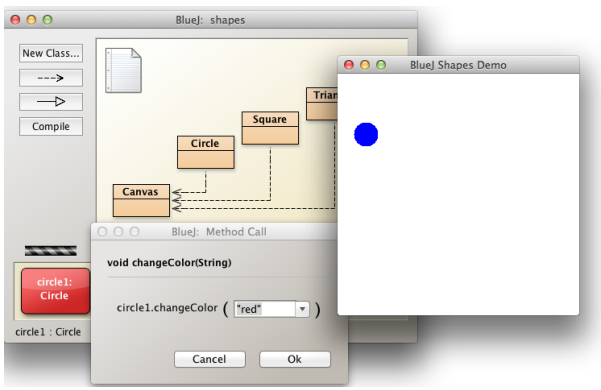
void: no data returned

```
void makeVisible();
void makeInvisible();
void changeColor(String);
void changeSize(int newDiameter);
```

Parameters

What are parameters?

- Some methods require no further information.
  - Example: *makeInvisible()*
- Others require additional information.
  - Additional information termed parameters
  - Zero, one or more parameters permitted.
  - Example *changeColor(String)*
  - String a Java object representing new color.



Method Signature

What is method signature?

- Consider method *voidChangeColor(String)*.
- This is method signature
- *void* means no value sent back to caller by method
- *changeColor* is method name
- *String*, Java data type, is *formal parameter*
- it green, for example, is *actual parameter*

```
void changeColor(String);
```

Data types

Signature of method:

- Informs number of parameters
- Informs type of each parameter
- Eight primitive data types, e.g:
  - int: represents integer values, e.g. 10, 25
  - boolean: may be *true* or *false*
- String: is an object. Represents text, e.g. "color", "10"

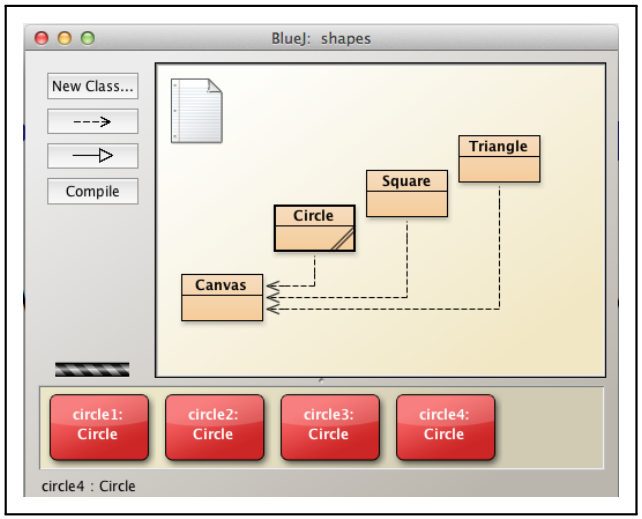
Some primitive Java types

int
float
double
long
boolean

Multiple instances

Many objects may be created from single class.

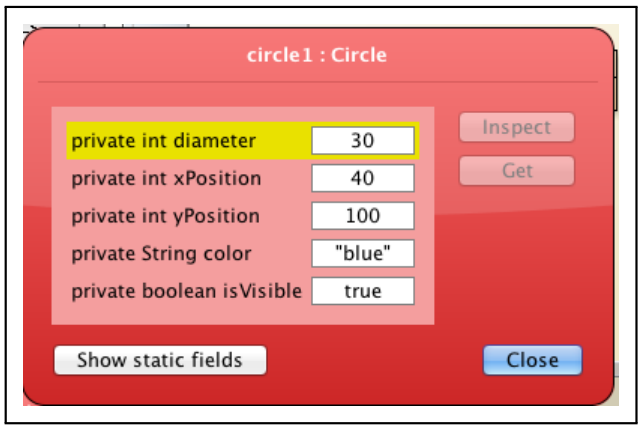
- Object: an instance of a class
- Instantiating class produces object
- Each object can have own set of internal data



State

Objects have state.

- Class Circle has fields
- Circle object field *color* has value (attribute)
  - Example: "blue"
- Object state: set of all values of all fields



More about objects?

What is an object?

- Entity produced in accordance with Class specification
- Class could be likened to blueprint + user guide
- Objects of same class have same fields
- Field types & names defined in Class, not objects
- But, field values reside in objects, not Class (in general)
- Field values typically differ across objects