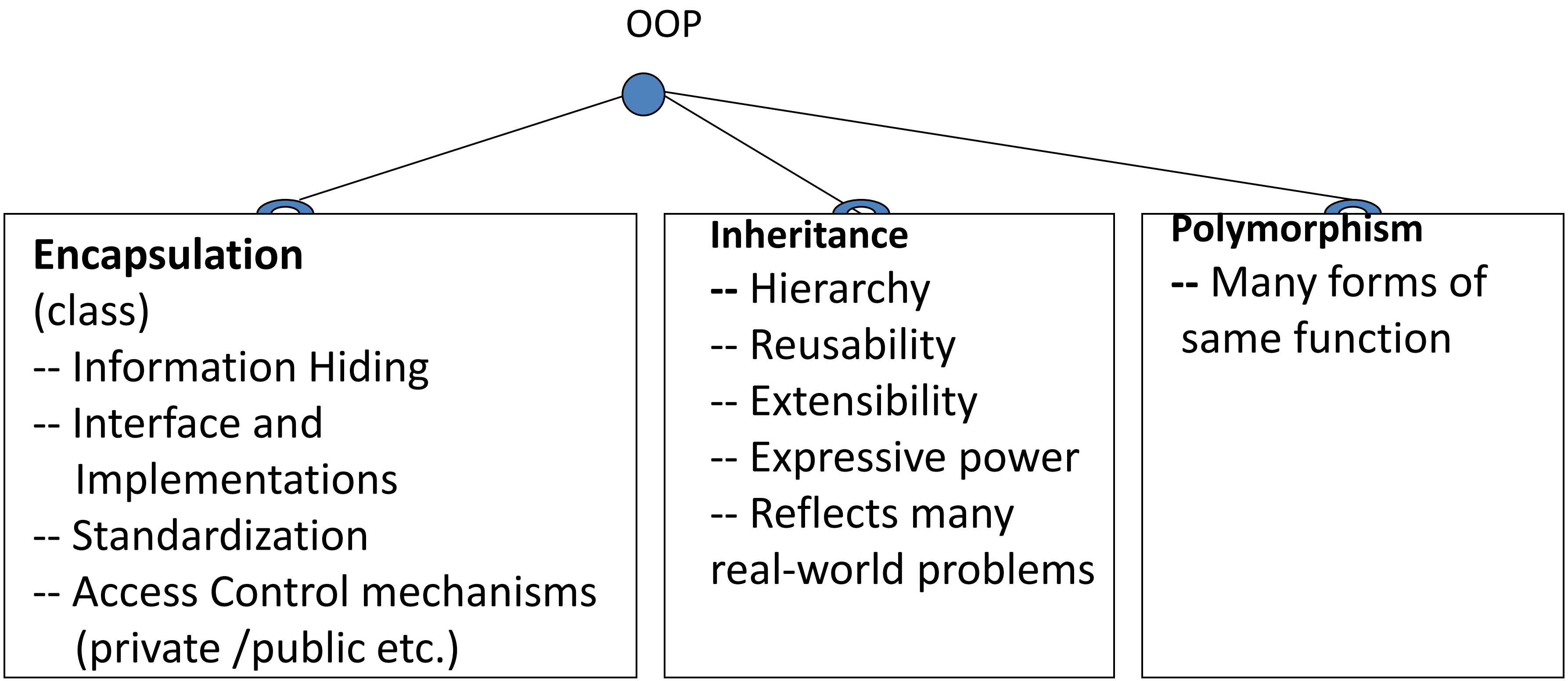
Introduction to Object-oriented Program Design

B. Ramamurthy

Object-Oriented Principles



OOP

Encapsulation
(class)
-- Information Hiding
-- Interface and
Implementations
-- Standardization
-- Access Control mechanisms
(private /public etc.)

Inheritance
-- Hierarchy
-- Reusability
-- Extensibility
-- Expressive power
-- Reflects many
real-world problems

Polymorphism
-- Many forms of
same function

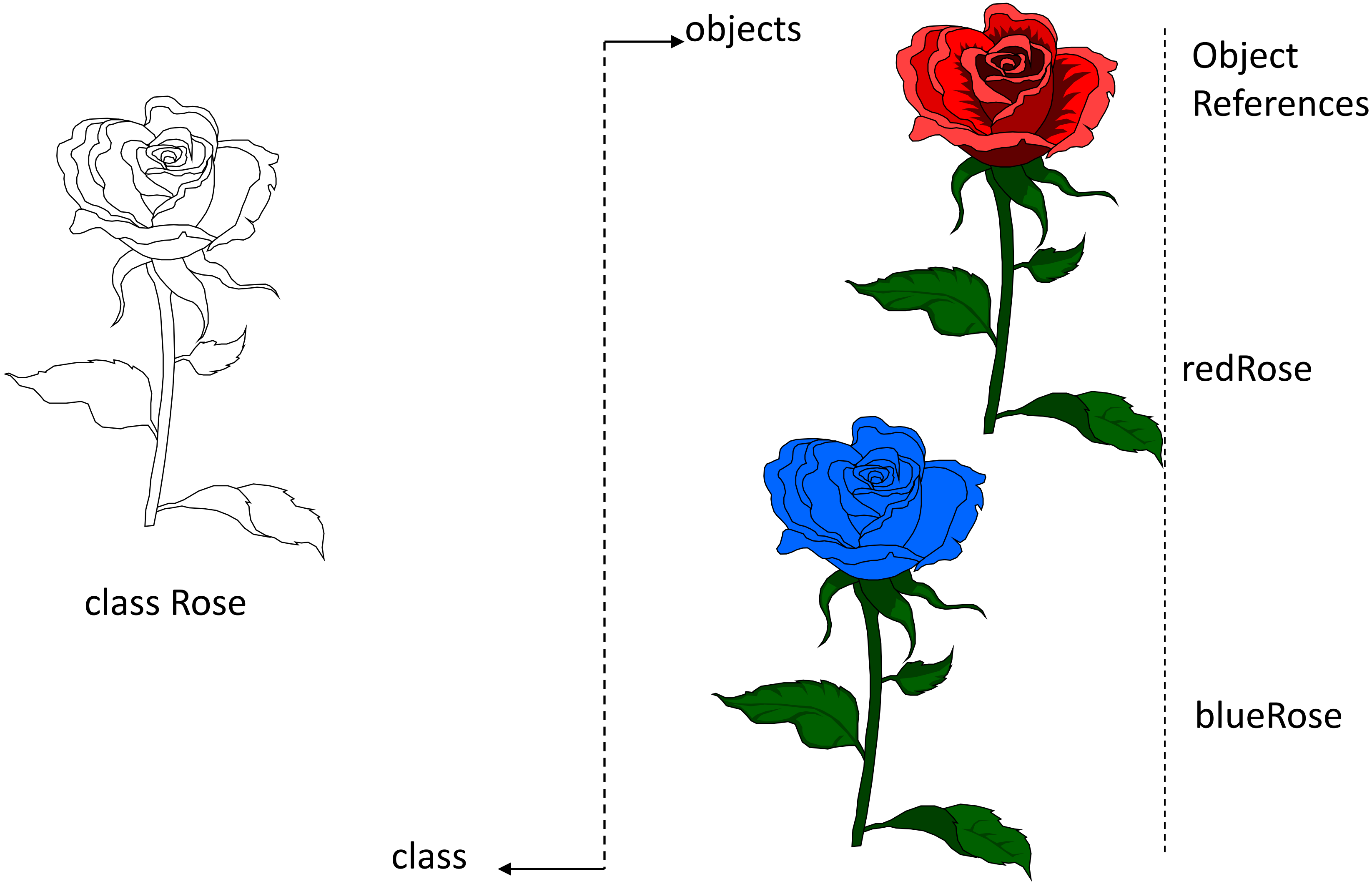
What is an Object?

- Object-oriented programming supports the view that programs are composed of objects that interact with one another.
- How would you describe an object?
- Using its characteristics (has a ----?) and its behaviors (can do ----?)
- Object must have unique identity (name) : Basketball, Blue ball
- Consider a ball:
  - Color and diameter are characteristics (Data Declarations)
  - throw, bounce, roll are behaviors (Methods)

Classes are Blueprints

- A class defines the general nature of a collection of objects of the same type.
- The process creating an object from a class is called instantiation.
- Every object is an instance of a particular class.
- There can be many instances of objects from the same class possible with different values for data.

Example



class Rose

objects

class

Object References

redRose

blueRose

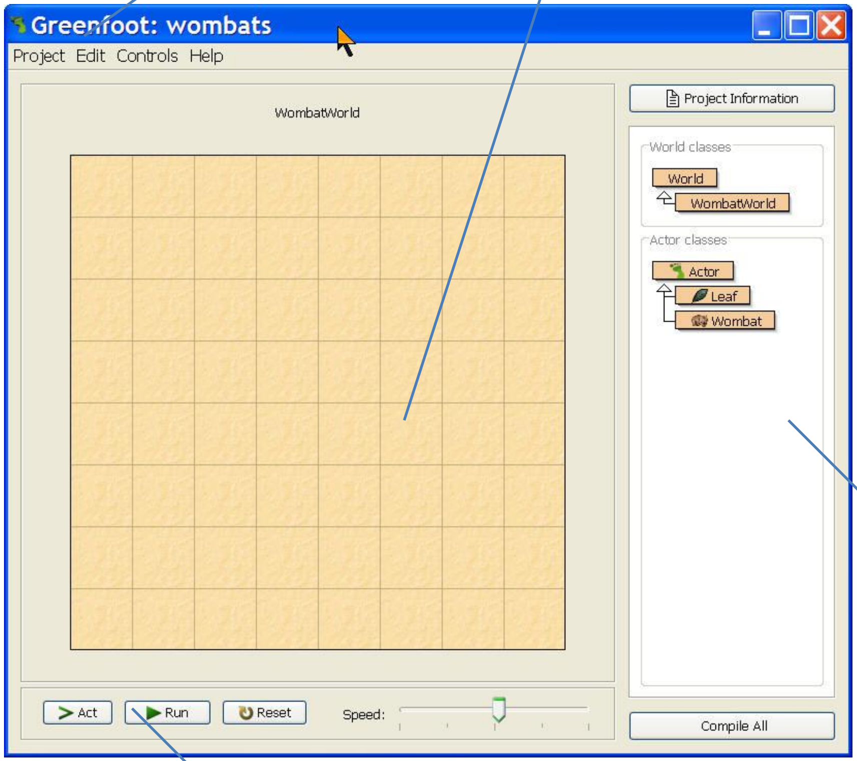
Bouquet Class

Many instances of real bouquet **objects** of this **Class** can be instantiated

Greenfoot

Project controls

World



Greenfoot is a 2D environment to develop Java programs

Greenfoot interface

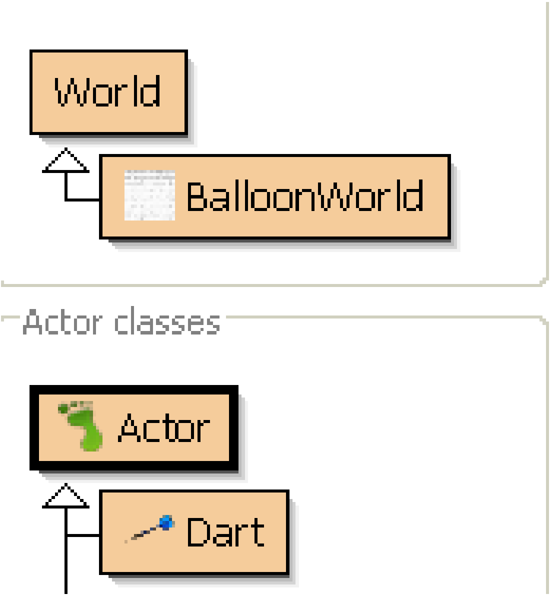
Class diagram

Book scenarios available at **http://www.greenfoot.org/book/**

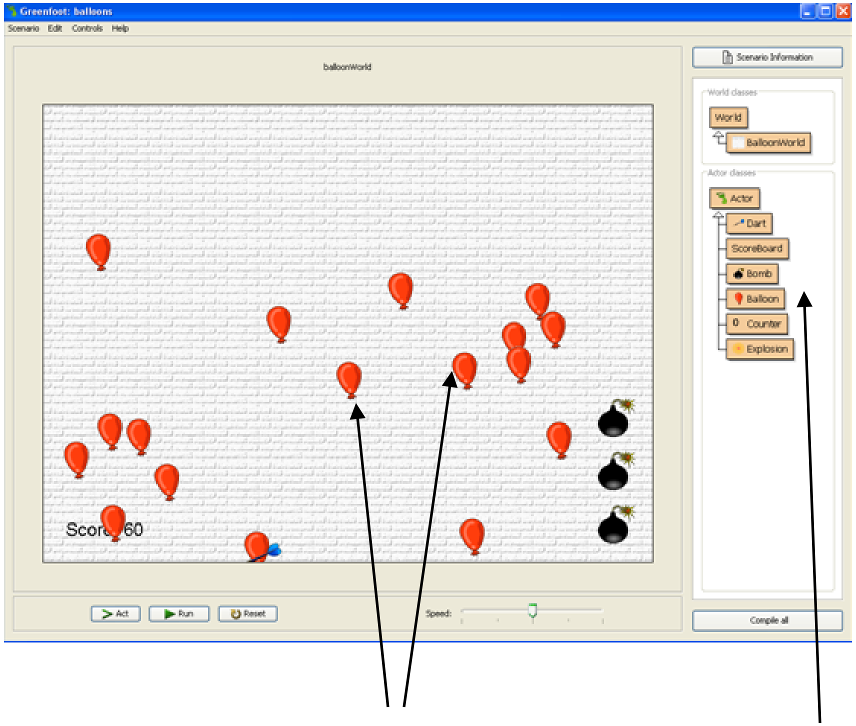
Execution controls

Worlds and Actors

- Greenfoot has two main classes
  - World
    - Place to hold and display actors
    - Has a width and height and cell size
      - Of type integer (int)
    - Has a background image
  - Actor
    - Actors know how to act
    - Actors have a x and y location in the world



Objects and Classes

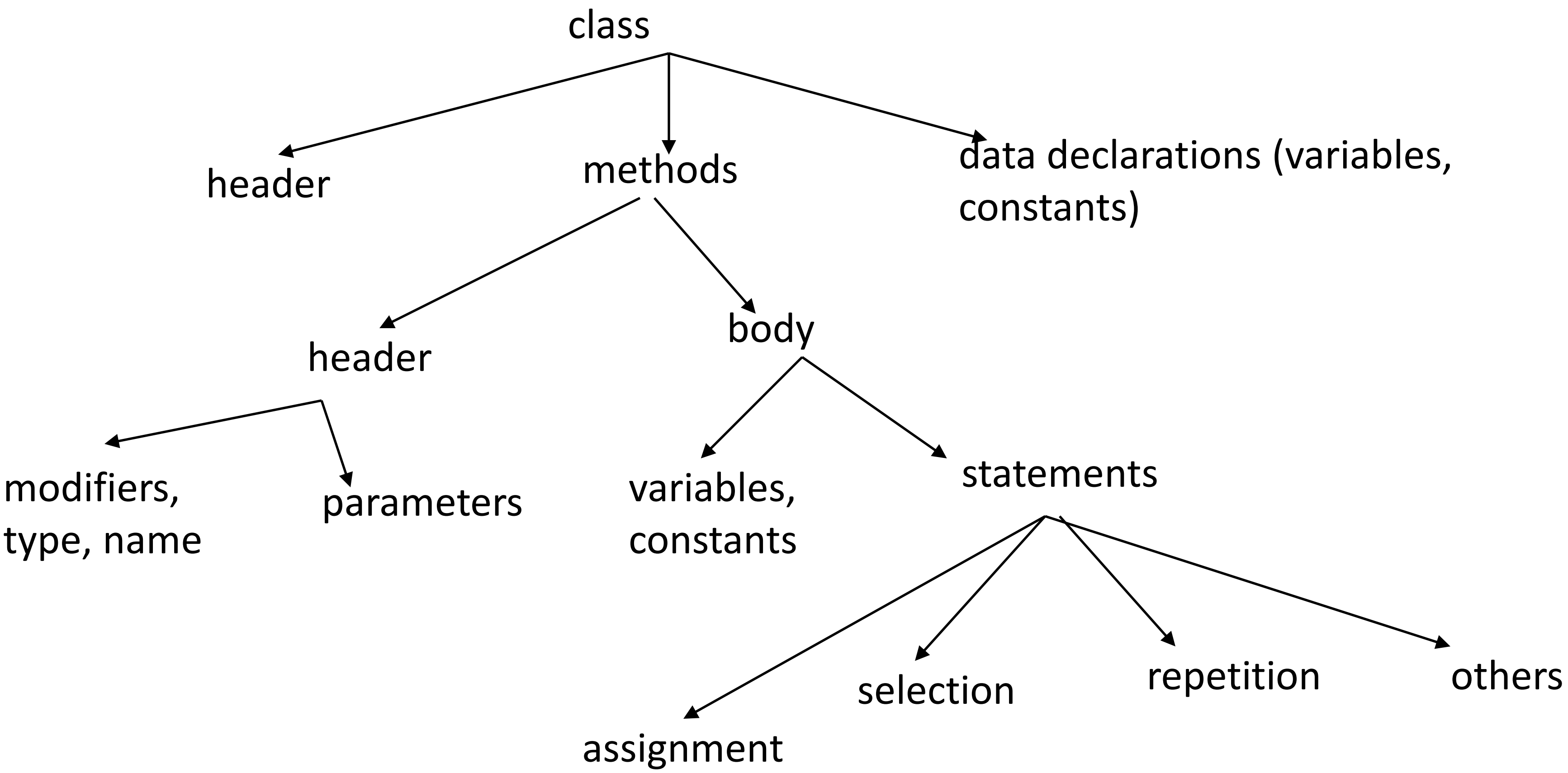


Objects

Classes

- Classes define what objects know and can do
  - There is one Balloon class
- Objects do the action
  - There can be many objects of the same class
  - There are many balloon objects

Elements of a Class



class

header

methods

data declarations (variables, constants)

header

body

modifiers, type, name

parameters

variables, constants

statements

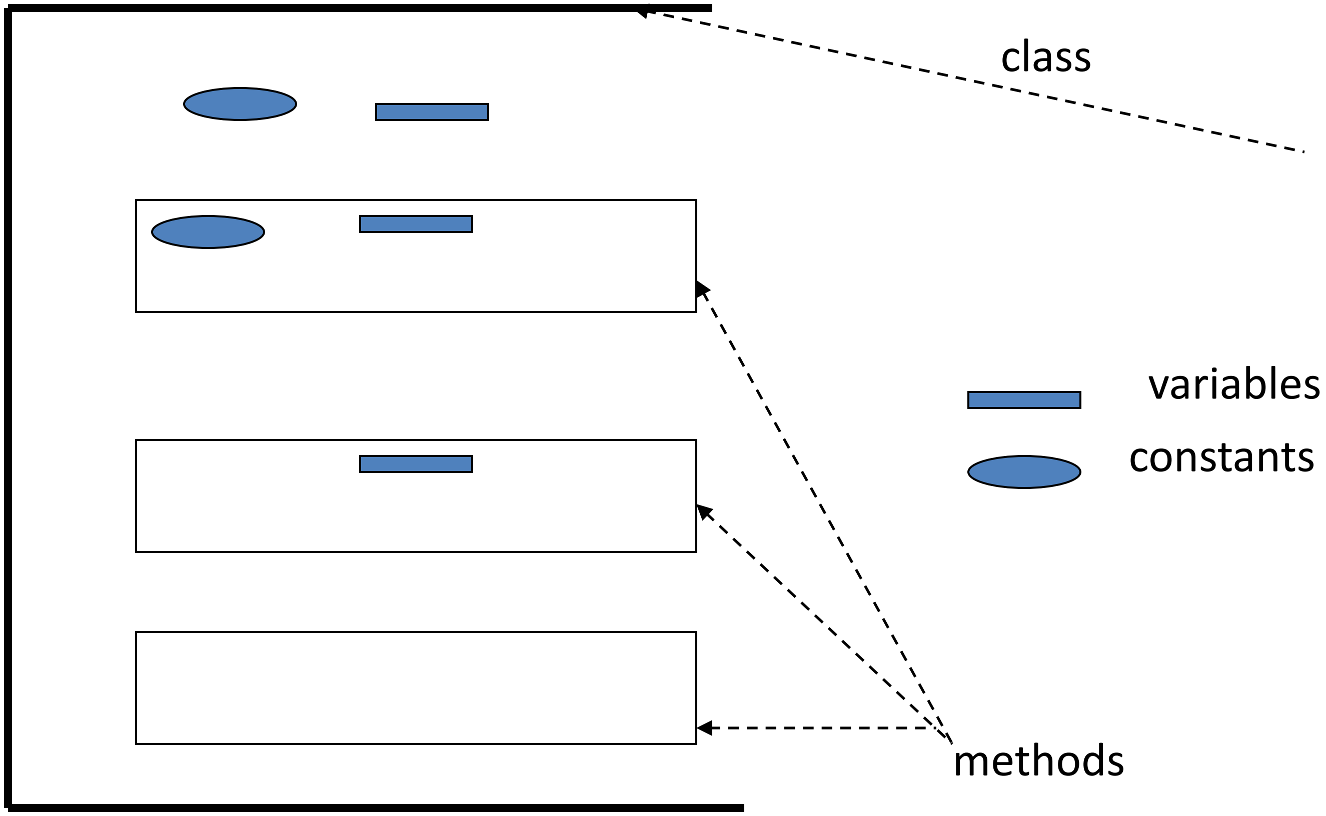
assignment

selection

repetition

others

Class Structure



Defining Classes

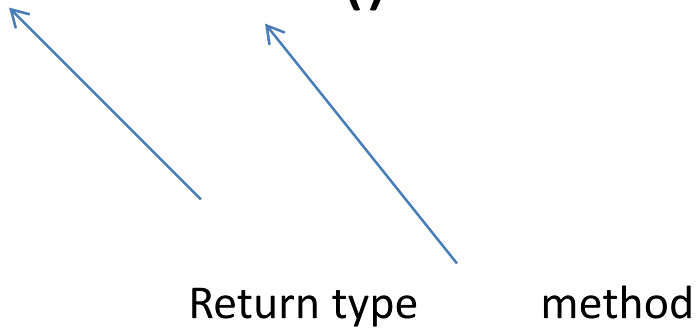
- Syntax:
- **class class_name {**
- **data-declarations**
- **constructors**
- **methods }**
- Constructors are special methods used for instantiating (or creating) objects from a class.
- Data declarations are implemented using variable and constant declarations.

Operator new and “dot”

- new operator creates a object and returns a reference to that object.
- After an object has been instantiated, you can use dot operator to access its methods and data declarations (if you have access permissions).

Interacting with objects: methods

- void move()
- void turnLeft()



Return type

method

Instantiation : Examples

- new Wombat()
- new Leaf()

constructor

Parameters

- void setSize (int size)

parameter

10 things

1. Greenfoot interface
2. Class
3. Object
4. World class
5. Actor class
6. Instantiate an object
7. Methods and data (defining a class)
8. Return type
9. Parameters
10. OOP (Object oriented principles)