

# Introduction

Ye Yang

Stevens Institute of Technology

# About the Course

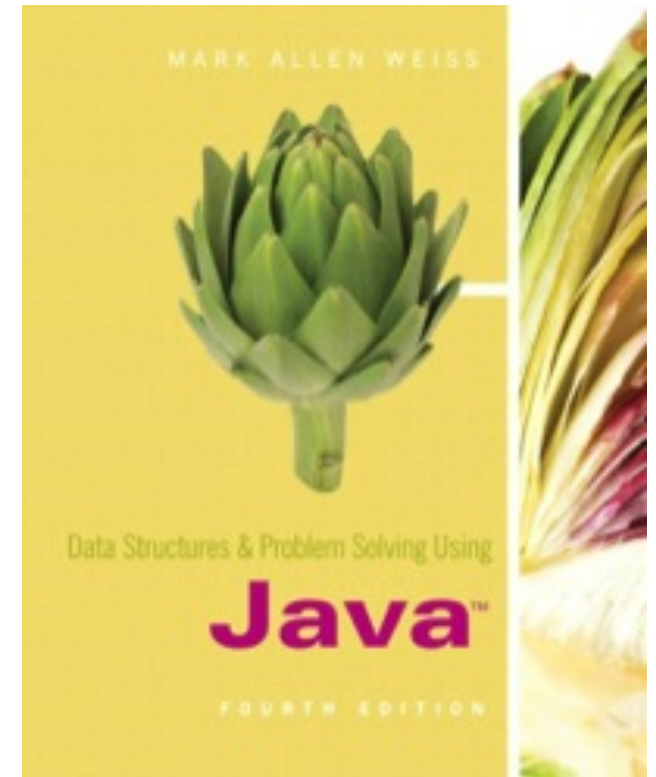
- This is an introductory course on Object-Oriented Software Engineering
  - Basic Terminologies in OO Programming
  - Java language
  - Commonly used data structures and algorithms
  - How to analyze these data structures and algorithms
- Lectures
  - Tuesdays: 12:25-1:15pm; Babbio 203
  - Fridays: 12-1:50pm; Babbio 203

# Staff

- Instructor: Ye Yang
  - Office hour: 2-4pm Tuesday or by appointment
  - Office: Babbio 537
  - Email: [ye.yang@stevens.edu](mailto:ye.yang@stevens.edu)
- Grader: Herbert J Zieger III
  - Email: [hzieger@stevens.edu](mailto:hzieger@stevens.edu)

# Textbook

- *Data Structures and Problem Solving Using Java*, by Mark Allen Weiss, Addison-Wesley 2009-10-07. 4th edition (October 7, 2009). ISBN-13: 978-0321541406
  - Errata:  
<http://users.cs.fiu.edu/~weiss/dsaajava2/errata.html>
  - Source Code:  
<http://users.cs.fiu.edu/~weiss/dsaajava2/code/>



# Additional Reference on Java

- Head First Java, 2nd Edition, by Kathy Sierra and Bert Bates. O'Reilly Media, Inc. 2005. ISBN: 0596009208.



# Course Linkages



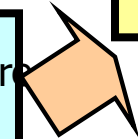
## E115: Introduction to Programming

- Basic programming skills



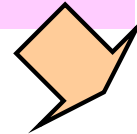
## SSW315: Object Oriented Software Development

- Reinforce problem solving skills
- Apply data structure and algorithms in Java programming



## SSW215: Individual Software Engineering

- Basic software engineering practices in Python



## SSW345: Model-Based Software Engineering

- Developing and managing complex projects
- Reinforce formal modeling and analysis skills

# Course Objectives

- Upon completion of the course, students will be able to:
  - construct object-oriented software using simple data structures and algorithms
  - perform simple time and space analyses of traditional computing problems and solutions
  - design, implement and perform unit testing of small Java applications



# Why Java?

- Industry standard (for now)
- Large ecosystem
- Not tied to any particular architecture (Java Virtual Machine)
- Other advantages include security and extensibility



# Grading

- Homework (25%)
  - 6 individual homework assignments
  - Code that does not compile will not be accepted
  - No late submission without consent from the instructor
- Quizzes (5%)
  - Unscheduled pop quizzes
  - No makeup quiz for absent day
- Weekly lab (20%)
  - ~30 minutes during Friday lectures
  - No late submission without consent from the instructor
  - No makeup lab for absent day
- Midterm (20%)
  - Oct. 22; Closed book.
- Final Exam (30%)
  - Dec. 6; Closed book.

# Academic Honesty

- You must include the Stevens' Honor Code in the beginning of your assignments or exam papers.
  - e.g. as comments or on top of your handwritten sheets
- Plagiarism is easy to catch.
- All homework and exams in this class are individual assignments.
- No collaboration.

# Expectations

- Attend class
  - Ask questions
- Read assigned text
- Start homework early
- Write well and clearly
- Get help when you need it

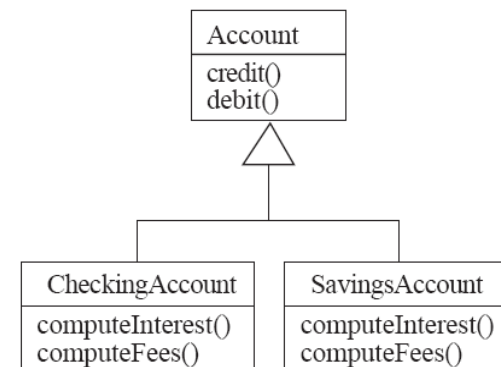
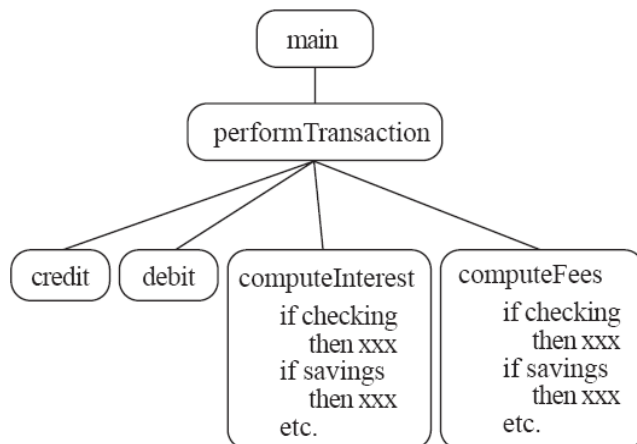
# Comparison of Two Paradigms

- Procedural paradigm (figure to the left):

- Software is organized around the notion of *procedures*
- *Procedural abstraction*
  - Works as long as the data is simple
- *Adding data abstractions*
  - Groups together the pieces of data that describe some entity
  - Helps reduce the system's complexity.
    - Such as *Records* and *structures*

- Object oriented paradigm (figure to the right):

- Organizing procedural abstractions in the context of data abstractions



# Object Oriented paradigm

- An approach to the solution of problems in which all computations are performed in the context of objects.
  - The objects are instances of classes, which:
    - are data abstractions
    - contain procedural abstractions that operate on the objects
  - A running program can be seen as a collection of objects collaborating to perform a given task

# The Language of Object-Oriented Programming

- Object: An object is a repository of data.
- Class: A class is a type of object.
- Method: A procedure or function that operates on an object or a class. A method is associated with a particular class.
- Inheritance: A class may inherit properties from a more general class.
- Polymorphism: The ability to have one method call work on several different classes of objects, even if those classes need different implementations of the method call.
- Object-Oriented: Each object knows its own class and which methods manipulate objects in that class.

# JAVA

- Java allows you to store data in variables, but first you must **declare** them, and specify their **type**.
  - Ruby: `x = 1`
  - Scheme: `(let ((x 1)) )`
  - Java: `int x; x = 1;`
- This Java declaration does two things.
  - It allocates a chunk of memory big enough to store an integer, which Java calls type "int".
  - It names the variable (chunk of memory) "x".


# Variables

- Variables are used not just to store numbers, but also to **reference** objects.
- There are two ways to get classes of objects to play with:
  - Use one defined by somebody else. Java has tons of pre-defined classes you can use. Many come in the "Java standard library" provided with every Java compiler.
  - Define your own.



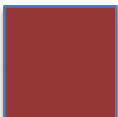


# String Class Example

- Java has a built-in class called String
  - String myString;
    - This does not create a String object. Instead, it declares a variable (chunk of memory) that can store a **reference** to a String object.

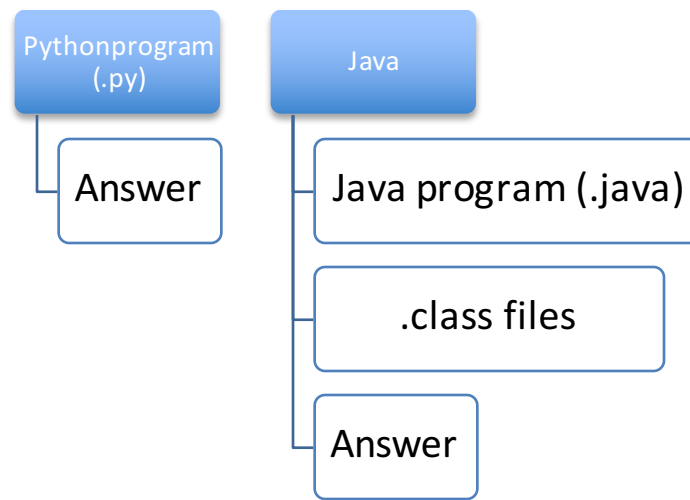
myString  (a variable, not an object)

- myString = new String();
  - This line performs two distinct steps.
    - First, the phrase "new String()" is called a **constructor**. It constructs a brand new String object.
    - Second, the assignment "=" causes myString to **reference** the object. You can think of this as myString pointing to the object.

myString    (a String object)

# JAVA Compilation

- Unlike Ruby or Python, Java programs must be compiled before you can run them.
- **Compilation** converts your written code to a machine-readable bytecode.
- The advantage is a faster program than one written in Scheme.
- The disadvantage is that you have to wait for it to compile.

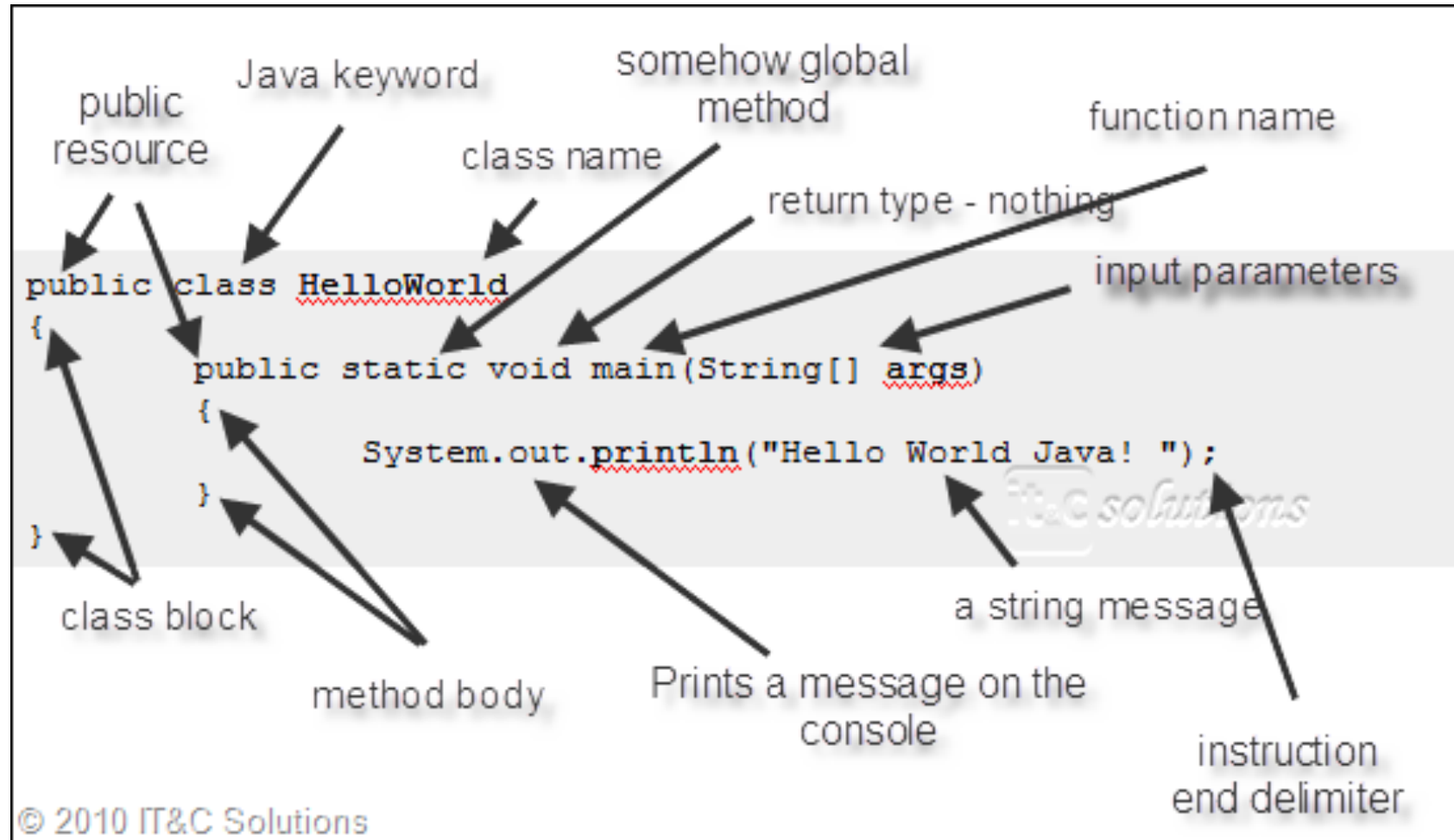


# HelloWorld.java

```
class HelloWorld {  
    public static void main (String[] args) {  
        System.out.println("Hello, world");  
    }  
}
```

- What are the classes? Objects? Methods?
- What does this program do?

# Anatomy of a Java Class



# Activity

- Install JAVA, Eclipse, JDK
- Read Chapter 1-2

# Next Lecture

- Using Objects
- Defining Classes