Fall 2017

Data Structures Lecture 1

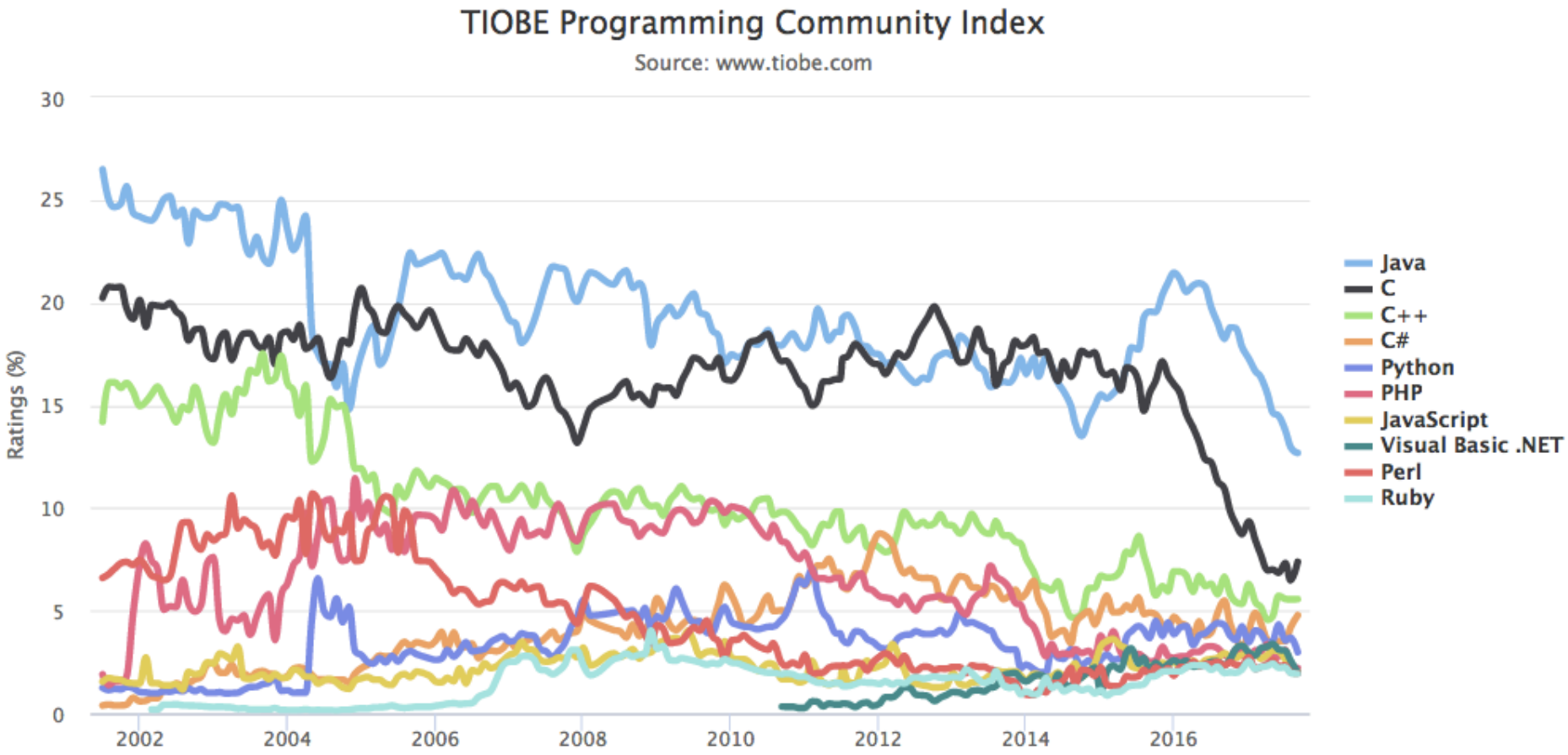
Fang Yu

Software Security Lab.
Dept. Management Information Systems,
National Chengchi University

A brief review of Java programming

Popularity of Programming Languages

Source: https://www.tiobe.com/tiobe-index/



History of PL Popularity

Sep 2017	Sep 2016	Change	Programming Language	Ratings	Change
1	1		Java	12.687%	-5.55%
2	2		C	7.382%	-3.57%
3	3		C++	5.565%	-1.09%
4	4		C#	4.779%	-0.71%
5	5		Python	2.983%	-1.32%
6	7	^	PHP	2.210%	-0.64%
7	6	v	JavaScript	2.017%	-0.91%
8	9	^	Visual Basic .NET	1.982%	-0.36%
9	10	^	Perl	1.952%	-0.38%
10	12	^	Ruby	1.933%	-0.03%
11	18	^^	R	1.816%	+0.13%

About Java

Java is

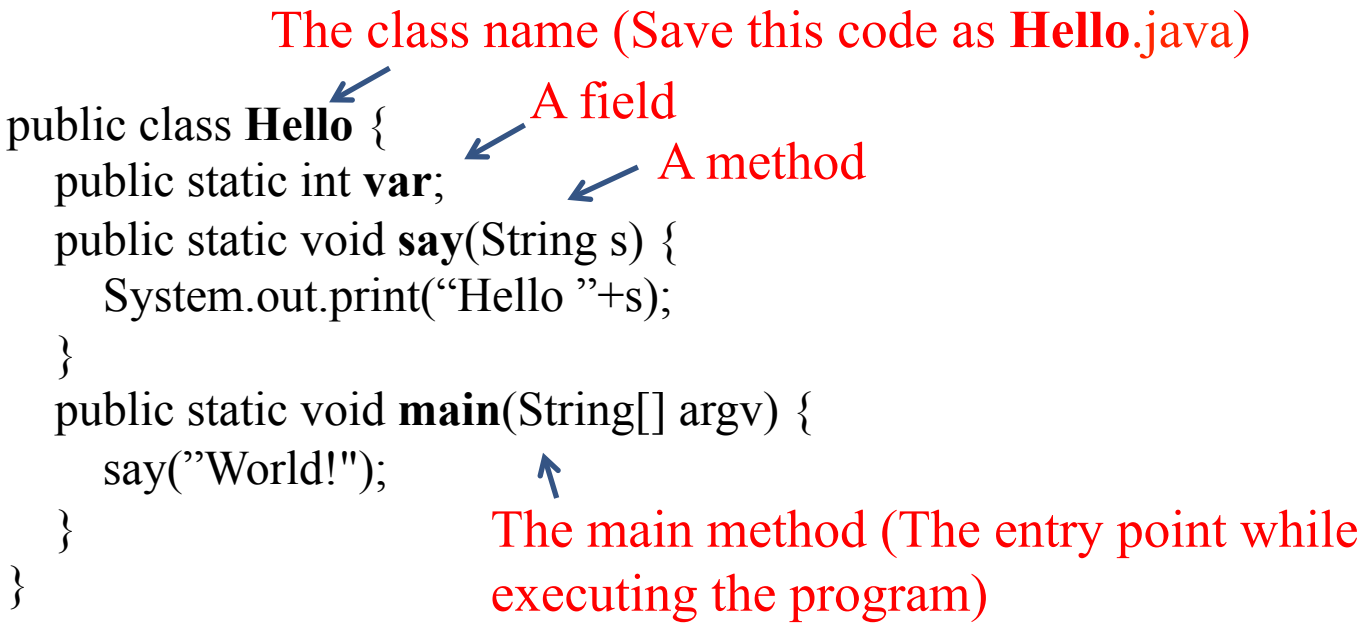
- One of the most popular languages in the past years: Simple, object-oriented, distributed, interpreted, robust, secure, architecture-neutral, portable, multi-threaded, dynamic, and more.

Three main elements: Class, Type, and Object

- An object is the basic unit in Java
- A class defines the type of an object

Java Programming Basics

- A class consists of
  - fields (to store data)
  - methods (to define operations that can act on data)



Modifiers

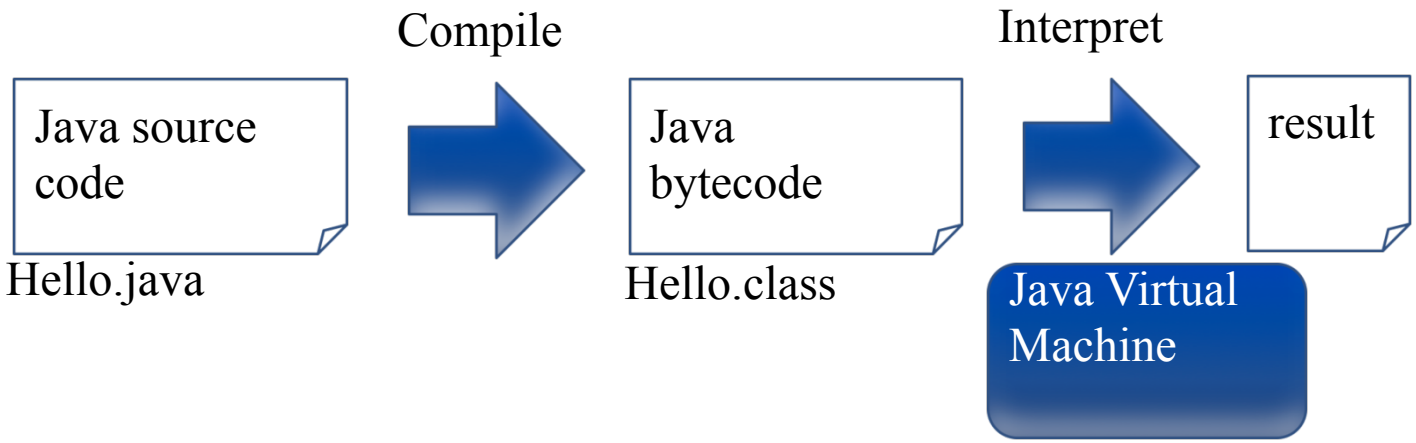
"public" indicates that anyone can run/extend/ import this class

"static" indicates the field/method belongs to the class, not objects

```
public class Hello {
    public static int var;
    public static void say(String s) {
        System.out.print("Hello "+s);
    }
    public static void main(String[] argv) {
        say("World!");
    }
}
```

"void" indicates that the method returns nothing

How Java works



- Execute your code in command lines
  - “javac Hello.java” to generate Hello.class
  - “java Hello” to execute the bytecode

Example: Operator

- Operators are similar to C++
  - E.g., =, +, - , *, /, %
- A simple example:
- Sum 1 to 100 using a formula

```
public class Example {
        public static void main(String[] argv) {
                int n = 100;
                System.out.println("1+2+...+" + n + " = " + ( n * (n + 1) / 2));
        }
}
```

```
javac Example.java
java Example

1+2+...+100 = 5050
```

Example: Loop

- Sum 1 to 100 using a method with for-loop

```
public class Example {
        public static int sum(int n) {
                return n*(n+1)/2;
        }
        public static void main(String[] argv) {
                int n = 100;
                System.out.println("1+2+...+"+n+" = " + sum(n));
        }
}
```

```
javac Example.java
java Example

1+2+...+100 = 5050
```

Example: Loop

- Sum 1 to 100 using a method with for-loop

```
public class Example {
        public static int sum(int n) {
                int total = 0;
                for (int i = 1; i <= n; i++) { total += i; }
                return total;
        }
        public static void main(String[] argv) {
                int n1 = 100, n2=200;
                System.out.println("1+2+...+"+n+" = " + sum(n));
        }
}
```

```
javac Example.java
java Example

1+2+...+100 = 5050
```

Example: Loop

- Sum 1 to 100 using a method with for-loop

```
public class Example {
        public static int sum(int n) {
                int total = 0;
                for (int i = 1; i <= n; i++) { total += i; }
                return total;
        }

        public static int sum2(int n1, int n2) {
                int total = 0;
                for (int i = n1; i <= n2; i++) { total += i; }
                return total;
        }

        public static void main(String[] argv) {
                int n1 = 100, n2=200;
                System.out.println(n1+“+...+"+n2+" = " + sum2(n1,n2));
        }
}
```

javac Example.java
java Example

1+2+...+100 = 5050

Use pre-defined class library

- Use java.util.Scanner for getting inputs
- The Scanner class reads the input stream and divides it into tokens by delimiters (whitespace)
- The Scanner class includes the following methods:

hasNext()	Return true if there is another token
next()	Return the next token
hasNext**Type**()	Return true if there is another token that can be interpreted as the **Type**
next**Type**()	Return the next token that can be interpreted as the **Type**

Use pre-defined class library

- Import the package

  import java.util.Scanner;

- Construct a Scanner object:

  Scanner in = **new** Scanner(System.in);

- Call its method:

  e.g., in.nextInt() or in.hasNext()

Example: Get a user input

- Sum using java.util.Scanner class

```
import java.util.Scanner;
public class Example {
        public static int sum(int n) {
                int total = 0;
                for (int i = 1; i <= n; i++) { total += i; }
                return total;
        }
        public static void main(String[] argv) {
                Scanner in = new Scanner(System.in);
                System.out.print(“Enter n: ”);
                int n = in.nextInt();
                System.out.println("1+2+...+"+n+" = " + sum(n));
        }
}
```

```
javac Example.java
java Example

Enter n: 100
1+2+...+100 = 5050
```

Example: Get user inputs

- Sum using java.util.Scanner class

```
import java.util.Scanner;
public class Example {
        public static int sum(int n) {
                int total = 0;
                for (int i = 1; i <= n; i++) { total += i; }
                return total;
        }
        public static int sum(int n1, n2) {
                int total = 0;
                for (int i = n1; i <= n2; i++) { total += i; }
                return total;
        }
        public static void main(String[] argv) {
                Scanner in = new Scanner(System.in);
                System.out.print(“Enter n1 n2: ”);
                int n1 = in.nextInt();
                int n2 = in.nextInt();
                System.out.println(n1+”+...+"+n2+" = " + sum(n1,n2));
        }
}
```

```
javac Example.java
java Example

Enter n1 n2: 10 100
10+...+100 = 5005
```

About Eclipse

Eclipse is

- An Integrated Development Environment (IDE) for Java and also many other languages
- An open source platform (free!)
- Maintained by many software development leaders like IBM and Borland

Eclipse Extension

Furthermore, Eclipse

- provides a common environment that companies can modify and customize by creating **plug-ins**
- These plug-ins can **add functionality** to Eclipse like modeling, UML, XML, metrics, reliability reports, and other information.
- The Eclipse web site has a list of links to many popular **plug-in repositories**

Learn Eclipse and Java

- Eclipse and Java tutorials. Watch this if you are a total beginner.
  http://eclipsetutorial.sourceforge.net/index.html
- A nice introduction to eclipse by L. Willaims et al. NCSU.
  http://agile.csc.ncsu.edu/SEMaterials/tutorials/eclipse/
- A nice java/eclipse tutorial on youtube:
  http://www.youtube.com/watch?v=UGmhks4K13g

Homework 1 (Due on 9/21)

- BMI Calculator:
  - BMI = (Weight in Kilograms / (Height in Meters x Height in Meters))
- Enter Height and Weight, return BMI and
  - "You are not in shape. Actually, you are not even close." if BMI >= 30
  - "To be honest, you are not in shape." if 30 > BMI >= 26
  - "You are in shape" if 26 > BMI >= 20
  - "You are under shape" if 20 > BMI
- Use Eclipse to write/execute/debug your java code
- Upload your code using WM5 (**no** direct copy accepted)
- TAs will show you "clear" hints to do so in Monday's lab

Coming up…

- The first lab is scheduled on Monday, Sep 18, 12:00-2:00pm. TAs will talk about Eclipse and HW1 (formats for automatic grading)
- We will discuss object oriented design and abstract data type next week
- Read TB Chapter 1 and Chapter 2
- **We will have a makeup course on Sep 30**
- **We will not have courses on Oct 5 and Oct. 12**