

## CSCI 209: Software Development

Sara Sprenkle  
sprenkles@wlu.edu

Sept 9, 2016

Sprenkle - CSCI209

1

## What is Programming?

"If you don't think carefully, you might think that programming is just typing statements in a programming language."

--Ward Cunningham

"Any fool can write code that a computer can understand. Good programmers write code that *humans* can understand."

-- Martin Fowler

"Refactoring: Improving the Design of Existing Code"

Sept 9, 2016

Sprenkle - CSCI209

2

## Discussion: What Is *Good* Software?

- What are its outcomes?
- What are the characteristics of the software?
- How can we write good software?
- What are short-term vs long-term goals?

Sept 9, 2016

Sprenkle - CSCI209

3

## Characteristics of *Good* Software?

- Free of bugs
  - Robust, reliability, stability
- Code is easy to read, extend, maintain
  - Readability, extensibility, maintainability
- Application is easy to use
  - Usability
- Efficiency
- Scalability

→ Referred to as the *\*ilities*

Sept 9, 2016

Sprenkle - CSCI209

4

## Course Content

- Software Design Principles
- Java
  - **Statically** typed language
- Software development, productivity tools
  - Eclipse
  - Version Control Systems

Sept 9, 2016

Sprenkle - CSCI209

5

## What to Expect from this Class

- Programming intensive
  - Variety of assignments and projects
  - More freedom in design, *\*ilities*
    - Larger portion of your grade
    - Correctness is **NOT** enough
  - Building on large library of classes
  - Read others' code! *Learn from the good and the bad*
  - Building larger applications
- Compare/Contrast with Python
  - PL design; what's the best PL for your needs
- Learning on your own
  - Online resources

Sept 9, 2016

Sprenkle - CSCI209

6

## Learning Objectives

- Discuss software development and practices **knowledgably**, using appropriate **terminology**
- Design, implement, test, and document efficient applications of **increasing size and complexity**
- Understand the designs and implementations of **others**
- Use a **version control system**
- Use many of the capabilities of the **Eclipse IDE**
- Test and debug large applications **systematically**, using standard tools
- Understand **design principles** such as DRY and shy
- Discuss the benefits and limitations of a **statically typed language**

Sept 9, 2016

Sprenkle - CSCI209

7

## Feedback from an Alumnus

"I am schooling everyone at work on OO design and Java. Seriously, keep pounding OO design principles in. It is incredibly practical. I'm teaching CS majors and Computer Engineering grads about this. It's crazy how some (good) technical schools don't stress this more."

Sept 9, 2016

Sprenkle - CSCI209

8

## Class Details

- **Course Web Site** <http://www.cs.wlu.edu/~sprenkle/cs209>
  - Example code, lecture slides, readings, resources
  - Course wiki
- **Optional Textbooks**
  - Use plentiful online resources instead
- **Participation**
  - Class discussions

Sept 9, 2016

Sprenkle - CSCI209

9

## Class Details

- **Programming Assignments**
  - Hands-on learning
  - Various sizes
  - To start, a lot of short ones
- 1 Testing Project
- 2 Exams
- Team Final Project

Sept 9, 2016

Sprenkle - CSCI209

10

## Course Dynamics

- **Professor's Responsibilities:**
  - Be **prepared** for class
  - Provide constructive feedback to students
  - Treat students with **respect**
  - **Challenge and encourage** students
  - Make material as clear as possible
- **Student's Responsibilities**
  - Be **prepared** for class (do readings and homework)
  - Give **attention and effort** in class to learning
  - Ask questions (**during class** and via email)
  - Use professor's office hours
  - Let professor know if something is going wrong
  - Treat other students and professor with **respect**

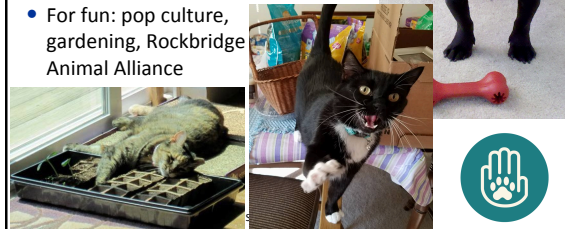
Sept 9, 2016

Sprenkle - CSCI209

11

## My Bio

- From Dallastown, PA
- Education: B.S., Gettysburg College; M.S., Duke University; Ph.D., University of Delaware
- For fun: pop culture, gardening, Rockbridge Animal Alliance



### My Research Interests

- General: Software engineering
- Automated testing of web applications
  - Develop algorithms
  - Implement in tools
  - Empirical studies
    - Try ideas out, see what actually happens, analyze
- Digital Humanities
  - Ancient Graffiti Project

Sept 9, 2016 Sprenkle - CSCI209 13

### My Research Interests

- Email received recently






Sept 9, 2016 Sprenkle - CSCI209 14

### Student Survey

- Class years
- Any experience with Java

Sept 9, 2016 Sprenkle - CSCI209 15

### What is Java? ... and, why should I learn it?

- From Sun Microsystems 
  - 1995, James Gosling and Patrick Naughton
  - Specifications
- Object-oriented
- Rich and **large** library 
- Develop cross-platform applications
  - Web, desktop, embedded
  - Frameworks
- Widely used 

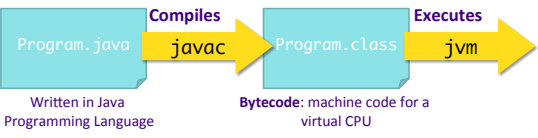
Sept 9, 2016 <http://www.tiobe.com/tiobe-index/> 16

### What is Java?

- Java Programming Language
- Java Virtual Machine
- Java Class Libraries

Sept 9, 2016 Sprenkle - CSCI209 17

### Overview: Writing, Executing Java Programs



```

    graph LR
      A[Program.java  
Written in Java  
Programming Language] -- Compiles (javac) --> B[Program.class  
Bytecode: machine code for a  
virtual CPU]
      B -- Executes (jvm) --> C[ ]
    
```

Sept 9, 2016 Sprenkle - CSCI209 18

### Java Programming Language

- Similar to Python
- But *entirely* object-oriented\*

Label the Python equivalents

**Step 1:**

Program.java

Compiler  
**javac**

Program.class

Written in Java Programming Language

**Bytecode:** machine code for a virtual CPU

Sept 9, 2016
Sprengle - CSC1209
19

### Java Virtual Machine (JVM)

**Step 2:**

Program.class

Bytecode

Mac JVM

UNIX JVM

...

Windows JVM

CPU  
(machine code)

CPU  
(machine code)

CPU  
(machine code)

- Same **bytecode** executes on each platform
- Don't need to provide the source code

Sept 9, 2016
Sprengle - CSC1209
20

### Java Virtual Machine (JVM)

- Emulates the CPU
  - Usually specified in software (rather than hardware)
- Executes the program's **bytecode**
  - Bytecode: virtual machine code
- JVMs available for each Java-supported platform
  - Enables program *portability*
- HotSpot VM
  - Code dynamically compiled to machine code
- Garbage Collection

Sept 9, 2016
Sprengle - CSC1209
21

### Traditional Way

Program

Platform-specific  
Compiler

Platform-specific Executable

- Executable is not portable

- How does Java's approach ease distribution of software?
- How is (I) Java and (II) the traditional approach the same and different from Python's approach?

Sept 9, 2016
Sprengle - CSC1209
22

### Java Class Libraries

- Pre-defined classes
  - Included with Java Development Kit (JDK) and Java Runtime Environment (JRE)
  - View the available classes online: <http://docs.oracle.com/javase/8/docs/api/>
- Similar in purpose to modules included with Python

Sept 9, 2016
Sprengle - CSC1209
23

		Java Language										
		java	javac	javadoc	jar	javap	jdeps	Scripting				
Tools & Tool APIs		Security	Monitoring	JConsole	VisualVM	JMC	JFR					
		JPDA	JVM TI	IDL	RMI	Java DB	Deployment					
		Internationalization	Web Services	Troubleshooting								
Deployment		Java Web Start			Applet / Java Plug-in							
User Interface		JavaFX										
Toolkits		Swing	Java 2D	AWT	Accessibility							
		Drag and Drop	Input Methods	Image I/O	Print Service	Sound						
Integration Libraries		IDL	JDBC	JNDI	RMI	RMI-IOP	Scripting					
		Beans	Security	Serialization	Extension Mechanism							
Other Base Libraries		JMX	XML_JAXP	Networking	Override Mechanism							
		JNI	Date and Time	Input/Output	Internationalization							
		lang and util										
lang and util Base Libraries		Math	Collections	Ref Objects	Regular Expressions							
		Logging	Management	Instrumentation	Concurrency Utilities							
		Reflection	Versioning	Preferences API	JAR	Zip						
Java Virtual Machine		Java HotSpot Client and Server VM										

Image from <http://docs.oracle.com/javase/8/docs/index.html>

Sept 9, 2016
Sprengle - CSC1209
24

## Benefits of Java

- Rapid development of programs
  - Large library of classes, including GUIs, Enterprise-level applications, Web applications
- Portability
  - Run program on multiple platforms without recompiling
- Statically-typed language
  - Compiling: find some errors before execution!
  - Can give performance boost by doing optimizations

Sept 9, 2016

Sprengle - CSC209

25

## Which 'Java' is this class about?

- Java programming language
- Java class libraries
- Use the JVM but won't learn about how it works
  - For more information:
    - <http://docs.oracle.com/javase/specs/>

Sept 9, 2016

Sprengle - CSC209

26

## Aside: JavaScript vs Java

- JavaScript is **not** Java
  - JavaScript is a *scripting* language, primarily embedded in HTML, executed by Web browsers



```
<script type="text/javascript">
function myFunction() {
    return ("Hello, have a nice day!")
}
</script>
</head>
<body>
<script type="text/javascript">
    document.write(myFunction())
</script>
```

Sept 9, 2016

Sprengle - CSC209

27

## Java Development Kit

- JDK: Java Development Kit
  - SDK: Software Development Kit
  - Free from Oracle
  - Contains
    - **javac**: Java compiler
    - **java**: Java Virtual Machine
    - Java class libraries
- <http://www.oracle.com/technetwork/java/javase/downloads/index.html>

Sept 9, 2016

Sprengle - CSC209

28

## Java Development Kit: JDK

- Installed on Linux machines
  - Java 1.8 should be reachable using default path
  - To see which executable you're executing use
    - which java
    - Should be `/usr/bin/java`
- Run `java -version` to determine which version you're running
- You can download the JDK for your machine from <http://www.oracle.com/technetwork/java/javase/downloads/index.html>
- JRE is for *running* Java applications
  - Does not include the compiler

Sept 9, 2016

Sprengle - CSC209

29

## Course Logistics

- Before class: read slides
  - No textbook; slides are the textbook
  - Execute examples
    - Download examples from course web site
  - Answer the questions
- In class
  - Discussion of questions
  - Problem-solving, programming

Sept 9, 2016

Sprengle - CSC209

30

## Looking Ahead

- Next time – discussion, in class work
  - Example programs
  - Data types
- Your To Do
  - Check out course web site
  - Read Monday's pre-class readings
  - Post answers to Sakai forum
    - What is your development process?
    - How will you succeed in course?
- Optional: install Java 8 on your home machine