

Objectives

- Introduction to Java
- Basics of Java Syntax
- Java fundamentals
 - Print statements

Open Poll Everywhere

Talk: Design for Sustainability on Earth and in Space



Danielle Wood

Assistant Professor of Media Arts and Sciences, Aeronautics and Astronautics; Director of the Space Enabled Research Group, Massachusetts Institute of Technology

Thursday, September 21, 5:00 pm
Stackhouse Theater

Extra credit opportunity:
Post in Canvas discussion forum

<https://my.wlu.edu/mudd-center/programs-and-events/2023-2024-ethics-of-design/danielle-wood>

Review: Version Control

- What are the features/functionality/benefits of version control?
- What are some of the common Git commands and what do they do?
- What is git? Vs what is GitHub?
- How did the git lab go? Make sense? Have questions?

Review slides and lab

Review: Version Control Systems

- Track versions, changes
- Collaboration
- Documenting authorship, changes
- Sandbox
- Back up, restore

Review: Common Git Commands

Command	What it does
clone	Clones a repository – sets up your repository so that you can coordinate
add <file>	Adds the <i>file</i> to the staging area
commit	Commits all the staged files (locally)
push	Push all your changes to the remote → You need your code to be pushed so that I can see it.
branch	List all local branches
branch <name>	Creates a new branch named <i>name</i>
checkout <name>	Switches to the branch named <i>name</i>



<https://xkcd.com/1597/>

Typical Git Workflow

1. Clone repository

- Git's default branch is `main`

2. Branch from `main` to a work-in-progress branch

- Work on `feature/next step/...`

3. When complete, merge branch back into `main`

- Optionally, push `main`

4. Switch back to and continue in work-in-progress branch (either same branch or new one)

5. Repeat

Reminder: Reloading Assignments

- Reload assignment pages whenever you return to them
 - Get most recent updates
 - I may have addressed issues that students alerted me to

INTRODUCTION TO JAVA

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
- Widely used
 - Frameworks to enable easier development



ORACLE®



Feedback from Alumnus

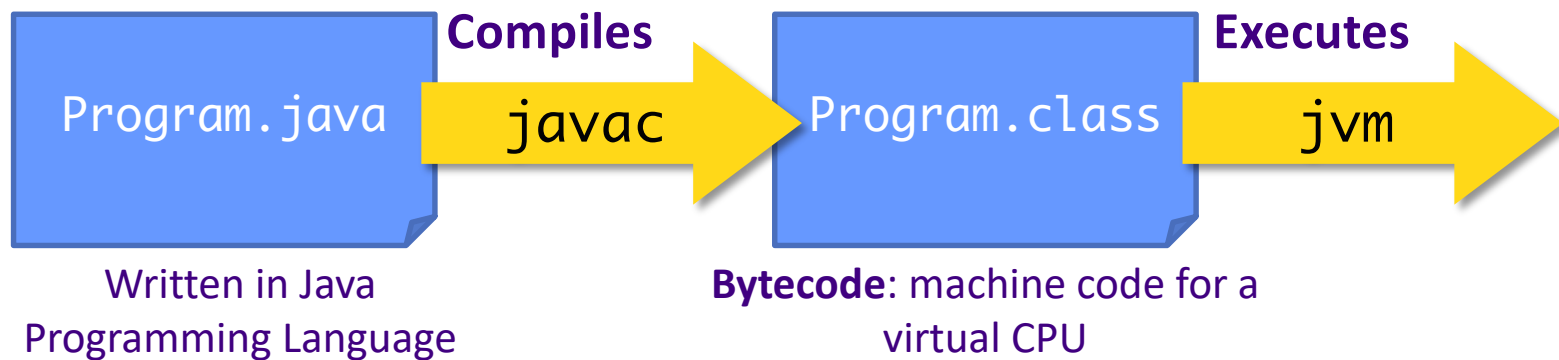
Also thank you for the Software Engineering course in Java!
My team codes in C++ and java knowledge have been very helpful in ramping up 😊

Goal: Transferrable skills

What is Java?

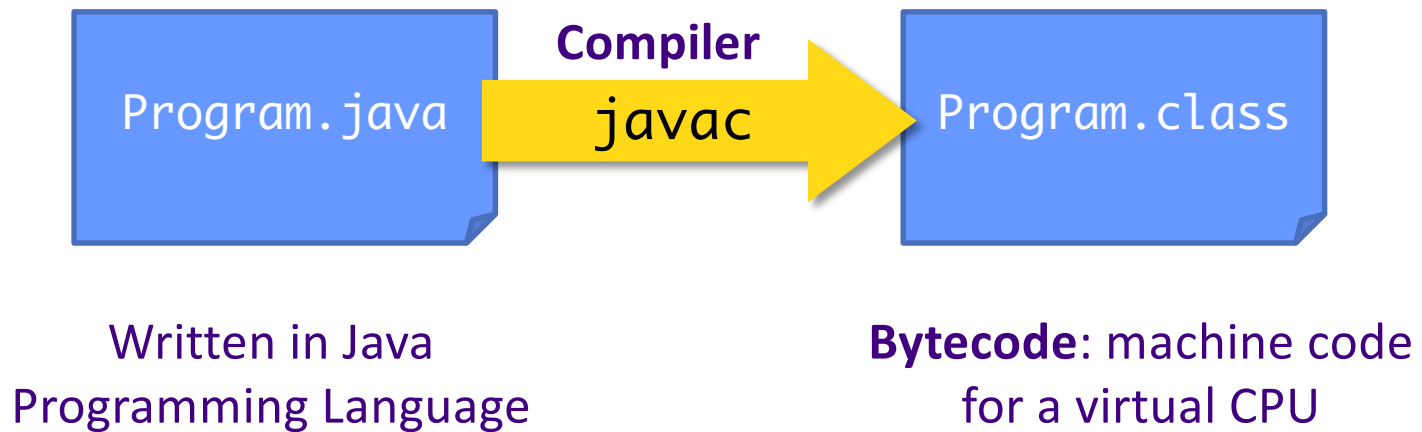
- Java Programming Language
- Java Virtual Machine
- Java Class Libraries

Overview: Compiling, Executing Java Programs



Compiling Java Programs

Step 1:

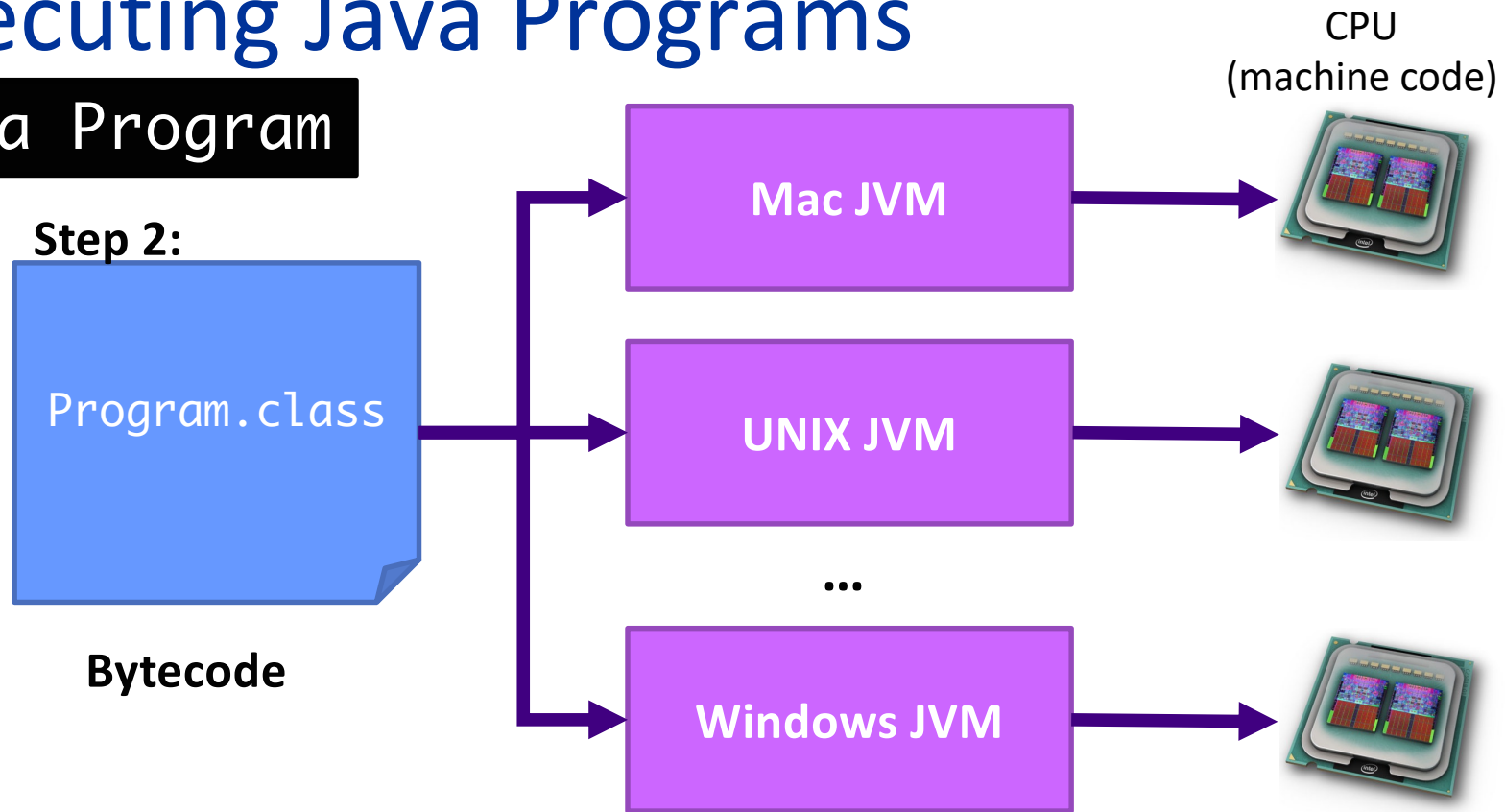


```
javac Program.java
```

- Compiler catches some errors
- Need to fix those errors and recompile

Executing Java Programs

java Program



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

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

Traditional (C/C++) Program Execution



- Example: I use my Mac-specific compiler to compile program into a Mac-specific executable
- Limitation: Executable is not portable

How does Java's approach affect distribution of software?



2 - How does software being Java-based affect its distribution?



Makes it harder because you have to install a JVM on every machine

A

Makes it more secure because you don't provide the source code

B

Makes it easier because same bytecode can be run on multiple platforms

C

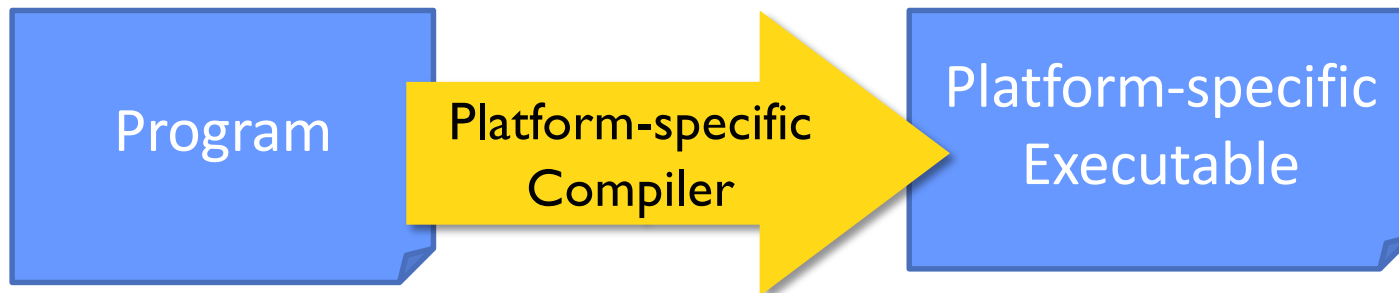
Makes it easier because many machines already have Java installed

D

None of the above

E

Traditional (C/C++) Program Execution

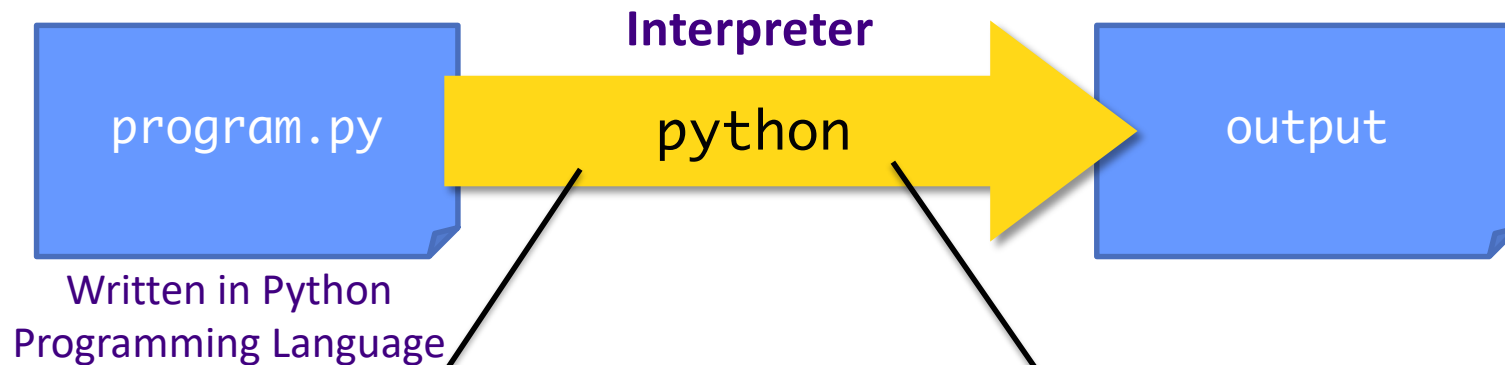


- Example: I use my Mac-specific compiler to compile program into a Mac-specific executable
- Limitation: Executable is not portable to any OS

What is Python's approach?

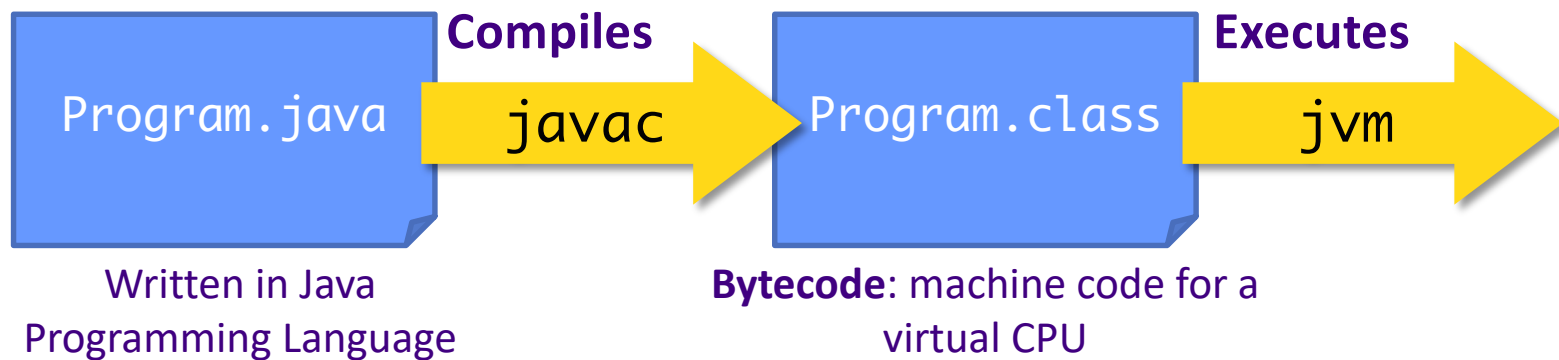
How are (I) Java and (II) the traditional approach the same and different from Python's approach?

Executing Python Programs



1. Syntax validation
 - exit if not valid
2. Translate Python code to Python bytecode
 - Bytecode stored in `__pycache__` directory
3. Python virtual machine execute Python bytecode

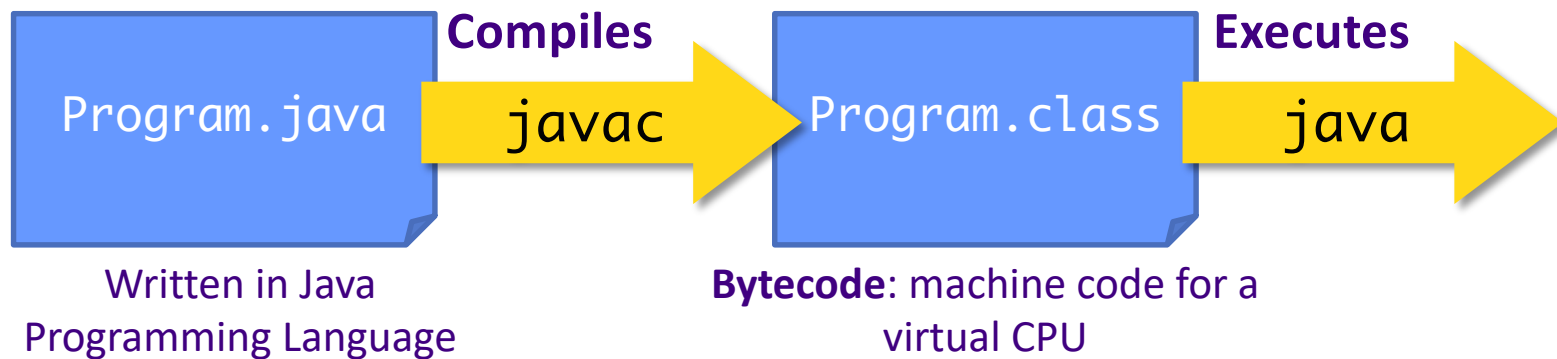
Overview: Compiling, Executing Java Programs



JDK: Java Development Kit

- Contains

- **javac**: Java compiler
- **java**: Java Virtual Machine
- Java class libraries



Java Class Libraries

- Pre-defined classes

- Included with Java Development Kit (JDK) and Java Runtime Environment (JRE)

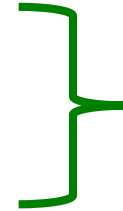
- View the available classes online:

- <https://docs.oracle.com/en/java/javase/17/docs/api/index.html>

- Similar in purpose to *modules* available for Python

What is Java?

- Java Programming Language
- Java Class Libraries



What this course
is about

- Java Virtual Machine

- Use the JVM but won't learn about how it works
- For more information on JVM:

<http://docs.oracle.com/javase/specs/>

Bringing It Together: 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
- Compiled
 - Find some errors before execution!
 - Statically typed
 - Can give performance boost by doing optimizations

LET'S PROGRAM!

Example Java Program: Hello.java

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

What are your observations about this program?
What can you figure out?

Example Java Program

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

- Everything in Java is inside a **class**
 - Java is *entirely* object-oriented*
 - This class is named **Hello**

Java: Files and Class Definitions

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

Defines the class Hello

Hello.java

- Name of the file **must** match the name of the class
 - E.g., Hello.java
- In general, each Java program file contains **one** class definition

Java: Blocks of Code

Blocks of code marked with { }


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

Defines the class Hello

Hello.java

Java: Access Modifiers

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



Access Modifier:

controls if other classes can use code in this class

Java: Method Definitions

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

method

- This class contains one *method* definition:
main

The `main` Method

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

- Similar to `main` in Python
 - But *must be* associated with a *class*
- Must take one parameter: an *array* of Strings
 - For command-line arguments
- Must be **public static**
- Must be **void**: data type of what method returns (nothing)
- `main` is *automatically* called when program is executed

Example Java Program

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

- Method contains one line of code
 - What do you think it does?

Java: Print Statements

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

- Calls the **println** method on the **System.out** object
- **println** takes one parameter, a **String**
- Displays string on terminal, terminates the line with new line (**\n**) character

Java: Comments

```
/**
 * Our first Java class: displays Hello!
 * @author Sara Sprenkle
 */
public class Hello {
    public static void main(String[] args) {
        //print a message
        System.out.println("Hello!");
    }
}
```

- Comments: `/* */` or `//`
➤ `/** */` are special **JavaDoc** comments

Looking Ahead

- Read textbook: Chapter 1 through 1.4: Lets look at a Java Program
- Complete Java compilation and execution before next class