MODULE 9: Software Systems and Tools

Lecture 9.4
Java Virtual Machine

Prepared By:

- Scott F. Midkiff, PhD
- Luiz A. DaSilva, PhD
- Kendall E. Giles, PhD

Electrical and Computer Engineering
Virginia Tech

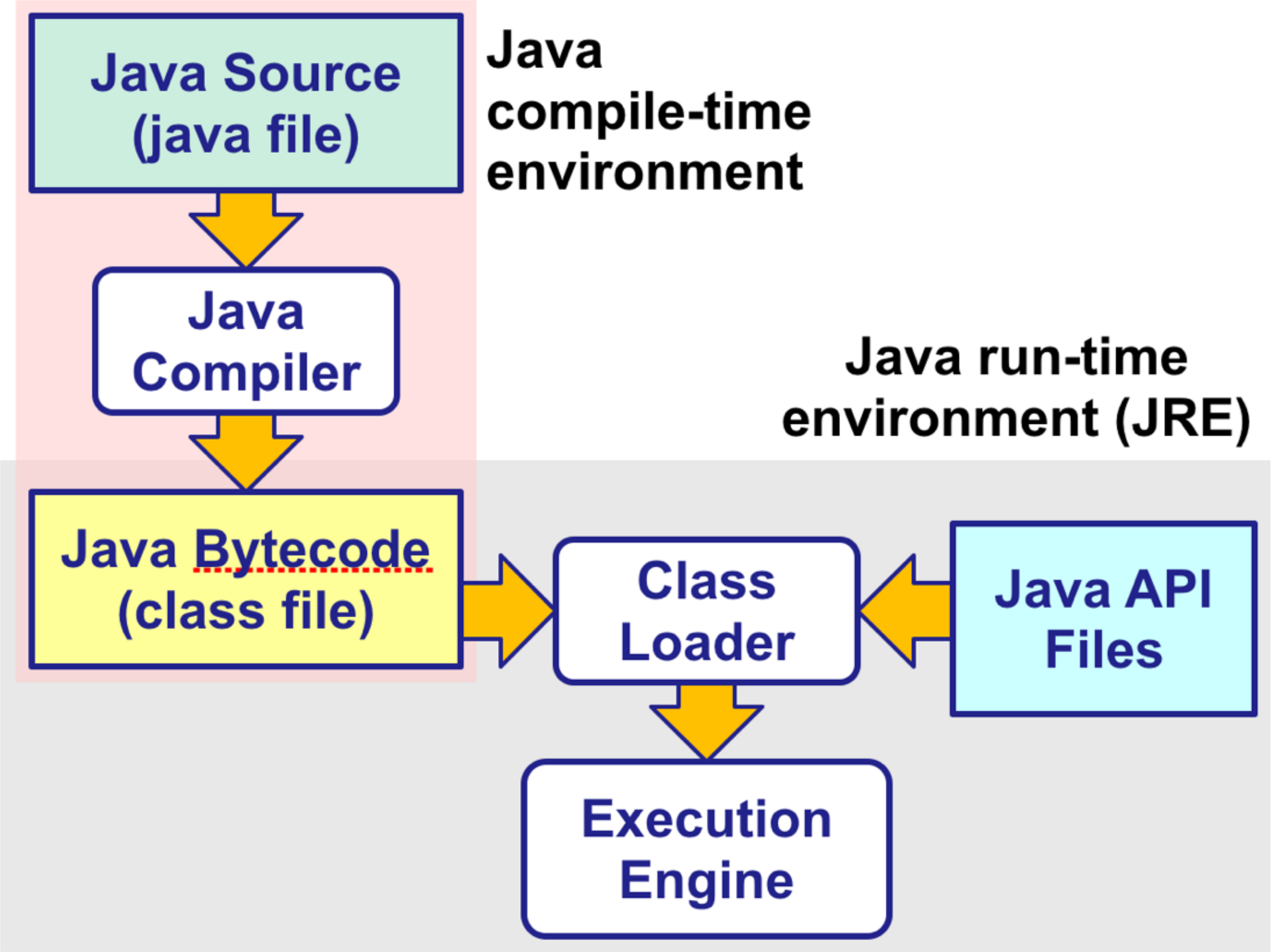
Lecture 9.4 Objectives

- Describe the key components of the Java development environment and the Java Run-time Environment (JRE)
- Describe the key components of the Java Virtual Machine (JVM)
- Describe the functions of the JVM Bytecode Interpreter

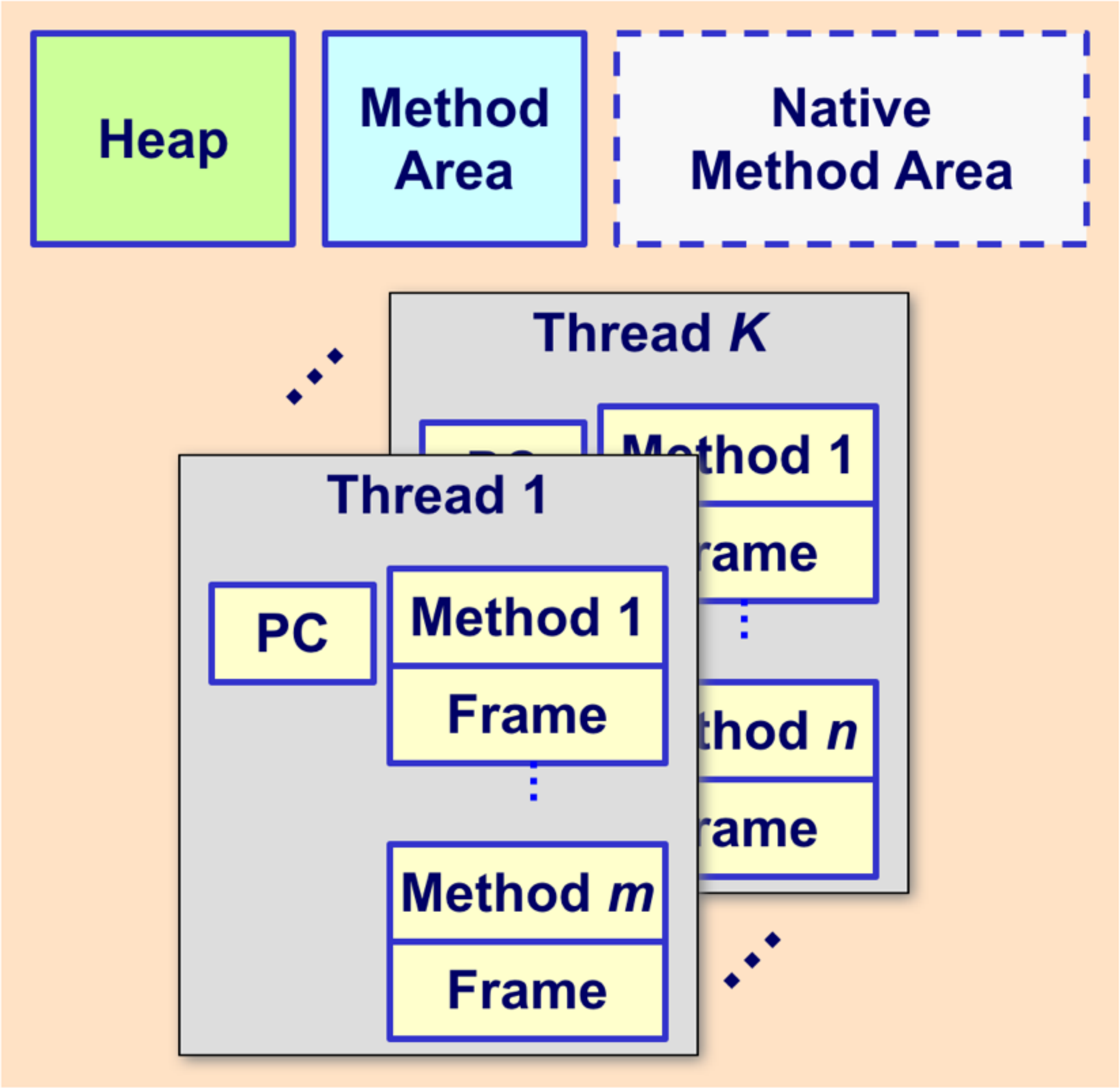
Java Virtual Machine

- The Java Virtual Machine (JVM) is virtual in that it is realized in software, which in turn executes in some other processor and operating system environment
- Java source code is compiled into Java bytecode and then executed on the JVM
- The Java Run-time Environment (JRE) provides the JVM execution engine and the supporting environment
- Stack-based instruction set architecture to simplify porting the JVM to different host processors

Java Development and Execution


Java Source (java file)
Java compile-time environment
Java Compiler
Java run-time environment (JRE)
Java Bytecode (class file)
Class Loader
Java API Files
Execution Engine

JVM Execution Environment



Java Virtual Machine

PC: Program Counter

JVM Execution Environment (2)

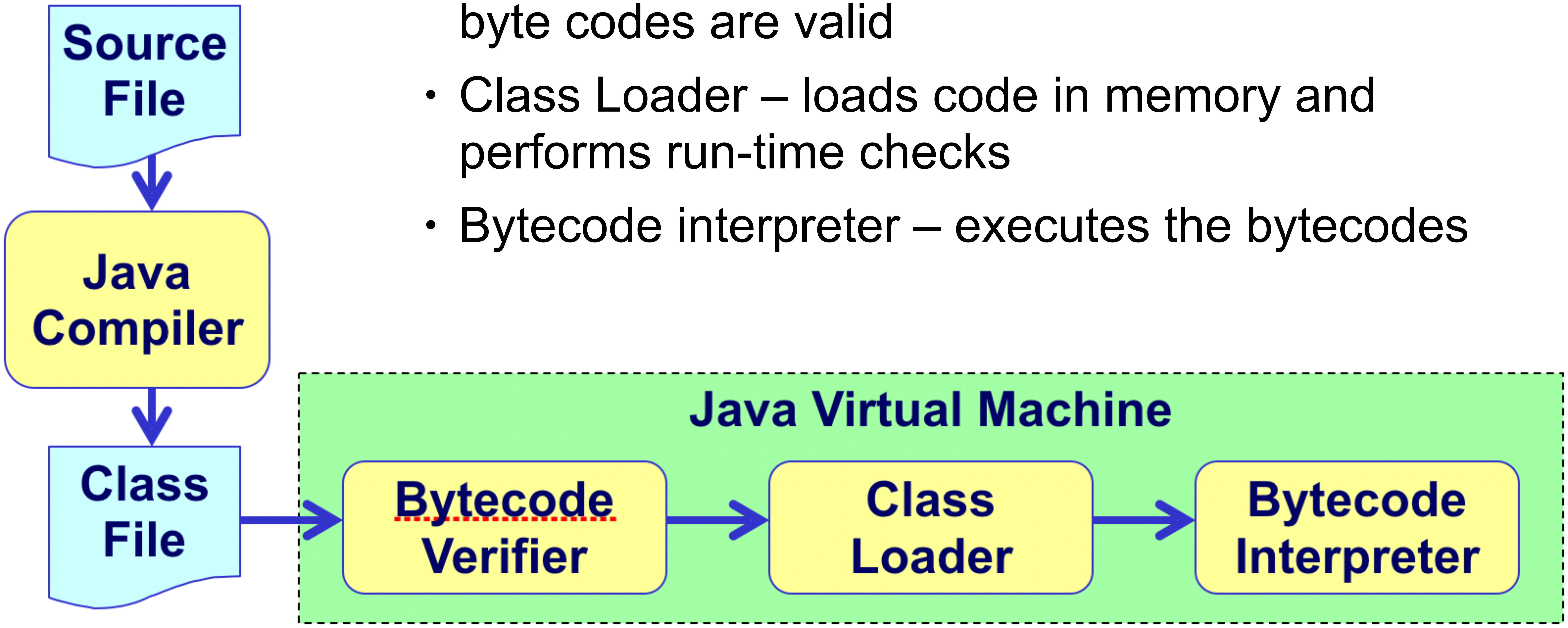
- The JVM supports multiple threads of execution
- Each thread has a program counter and one or more method frames
- Each method frame includes:
    - Reference to bytecodes in a common Method Area – the code
    - Data storage – the data

JVM Execution Environment (3)

- Method area is a common repository for bytecodes
- Heap provides data storage
    - Allocated as needed for execution
    - Garbage collection process frees unused heap storage
- Native Method Area provides interfaces to the host operating system and applications not written in Java

JVM Execution Process

- Bytecode Verifier – ensures file integrity and that byte codes are valid
- Class Loader – loads code in memory and performs run-time checks
- Bytecode interpreter – executes the bytecodes



Six Phases of JVM Bytecode Interpreter

- Request any missing classes or system binary files to complete linking
- Create and initialize the main stack frame and local variables
- Create and begin execution of one or more threads
- Manage heap storage
    - Allocation
    - Garbage collection
- Deallocated resources as threads terminate
- Terminate threads and terminate the JVM when the program ends

CHECK POINT

As a checkpoint of your understanding, please pause the video and make sure you can do the following:

- Describe the key components of the Java development environment and the Java Run-time Environment (JRE)
- Describe the key components of the Java Virtual Machine (JVM)
- Describe the functions of the JVM Bytecode Interpreter

If you have any difficulties, please review the lecture video before continuing.

Summary

- Java source files are compiled into bytecodes, which are then combined with other bytecodes and executed by the JVM
- Each thread of execution has its own program counter and method frames, while other heap memory and bytecode storage are shared
- Key components of the JVM for execution
    - Bytecode Verifier
    - Class Loader
    - Bytecode interpreter
- Bytecode interpreter manages threads and resources

MODULE 9: Software Systems and Tools

Lecture 9.4
Java Virtual Machine

Prepared By:

- Scott F. Midkiff, PhD
- Luiz A. DaSilva, PhD
- Kendall E. Giles, PhD

Electrical and Computer Engineering

Virginia Tech