


JVM Performance and Tuning

Module One Introduction to the JVM



A top-down view of a wooden desk. In the top right, a portion of a silver laptop is visible, showing keys like 'tab', 'Q', 'W', 'E', 'caps lock', 'A', 'S', 'Z', 'fn', 'control', and 'option'. Below the laptop, a pair of black-rimmed glasses lies horizontally. To the right of the glasses is a white ceramic cup filled with dark coffee, with a yellow handle. In the top center, a small green succulent plant in a dark pot sits on the desk. The background is a light-colored wooden surface with a prominent grain pattern.

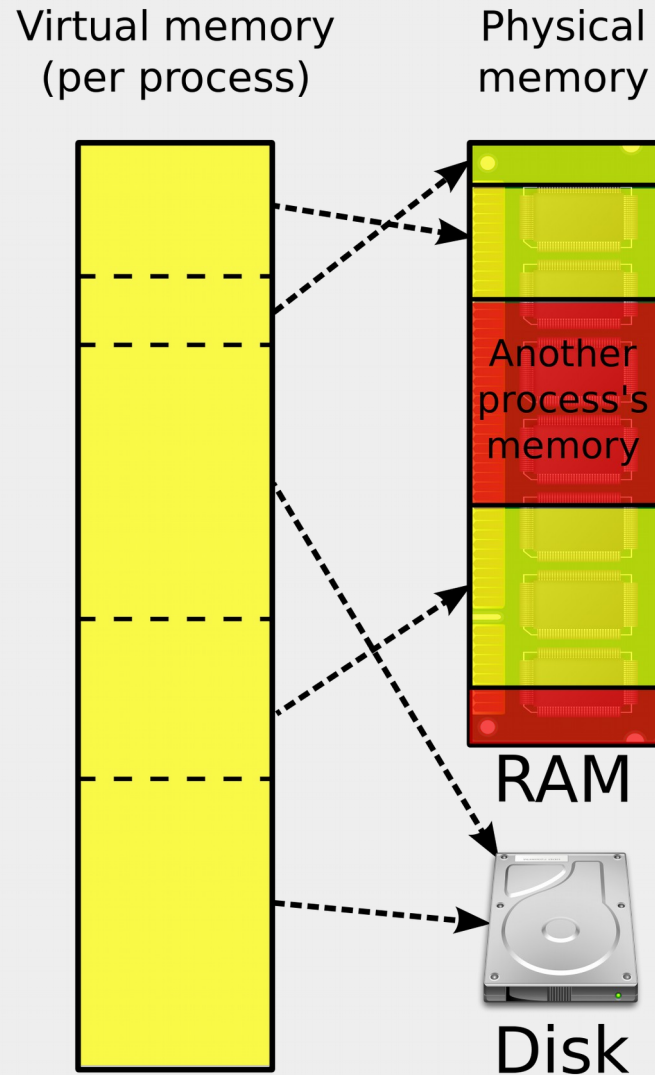
Module Topics

1. **Virtualization**
2. Java and the Java Virtual Machine
3. The Java Ecosystem
4. The JVM Architecture
5. Varieties of JVMs

Virtualization

- Has been around since the 1960s
- A layer of indirection
 - Between hardware and software
 - *Virtual devices and hardware*
 - Between software and software
 - *Emulating operating systems or environments*
 - *Exist in a wide variety of forms*
- JVMs are a specific form of virtualization
 - The JVM involves both layers of indirection

Hardware Virtualization – Virtual Memory



- Allows multiple processes to access the same physical memory
- Allows processes to have arbitrary amounts of memory independent of actual physical memory
- Managed by the operating system
- Uses “abstractions” of real hardware

Hardware Emulation

- Layer of indirection that makes some hardware look like other hardware
- Used to allow portability of:
 - Applications that interact directly with legacy hardware
 - Applications between hardware architectures
- Or to create virtual hardware that doesn't exist
 - The JVM is an emulator for an “imaginary” machine

Wang2200 Emulation



The Wang2200 was the leading office automation system in the 1980s, but the Wang discontinued the product leaving thousands of customers that could no longer run critical software.

Nikawa Inc. wrote a Wang2200 emulator that allowed customers to run all their office automation software on an PC.

Virtual Architecture

```
NEW | 1 | 2 | 3 |
Hercules Version : 3.07
Host name       : raspberrypi
Host OS        : Linux-3.1.9+ #84 Fri Apr 13 12:27:52 BST 2012
Host Architecture : armv6l
Processors      : UP
Chanl Subsys    : 0
Device number   : 0010
Subchannel      : 0004

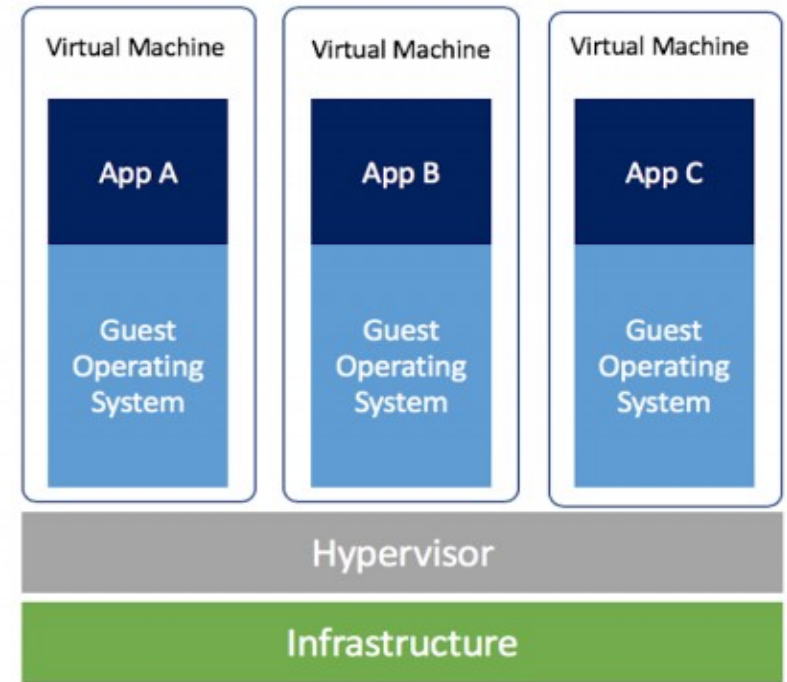
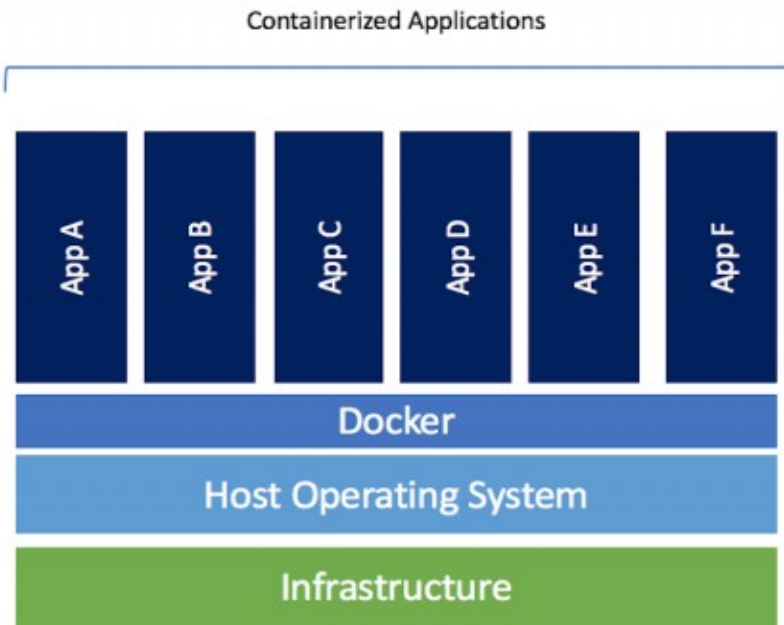
      HHH      HHH      The S/370, ESA/390 and z/Architecture
      HHH      HHH      Emulator
      HHH      HHH
      HHH      HHH      EEEE RRR   CCC U  U L   EEEE SSS
      HHHHHHHHHHHHHHHHHHHHH E   R R C   U  U L   E   S
      HHHHHHHHHHHHHHHHHHHHH EEE RRR   C   U  U L   EEE  SS
      HHHHHHHHHHHHHHHHHHHHH E   R R C   U  U L   E     S
      HHH      HHH      EEEE R  R   CCC UU  LLLL EEEE SSS
      HHH      HHH
      HHH      HHH
      HHH      HHH      My PC thinks it's a MAINFRAME


      Copyright (C) 1999-2010 Roger Bowler, Jan Jaeger, and others
```

Virtual Machines and Containers

- Modern VMs generally are used to allow multiple OS and environments to run on the same hardware
 - E.g. Linux, Win10, Win7 all can run on the same hardware
- Containers are stripped down VMs
 - They run as processes inside an OS
 - But still provide virtualization
 - Considered lightweight VMs

Virtual Machines and Containers



A top-down view of a wooden desk. In the top right corner, a portion of a silver laptop is visible, showing keys like 'tab', 'caps lock', 'shift', 'control', 'option', 'fn', 'Z', 'A', 'S', 'W', 'E', 'Q', '1', '2', '3', '4', '5', '6', '7', '8', '9', '0', 'ESC', and 'F12'. Below the laptop, a pair of black-rimmed glasses lies horizontally. To the left of the glasses is a small, green, succulent-like plant in a dark pot. In the bottom right corner, a white ceramic cup filled with dark coffee sits on the desk. A tablet or another laptop is partially visible at the very bottom edge. The background is a light-colored wooden surface with a prominent grain pattern.

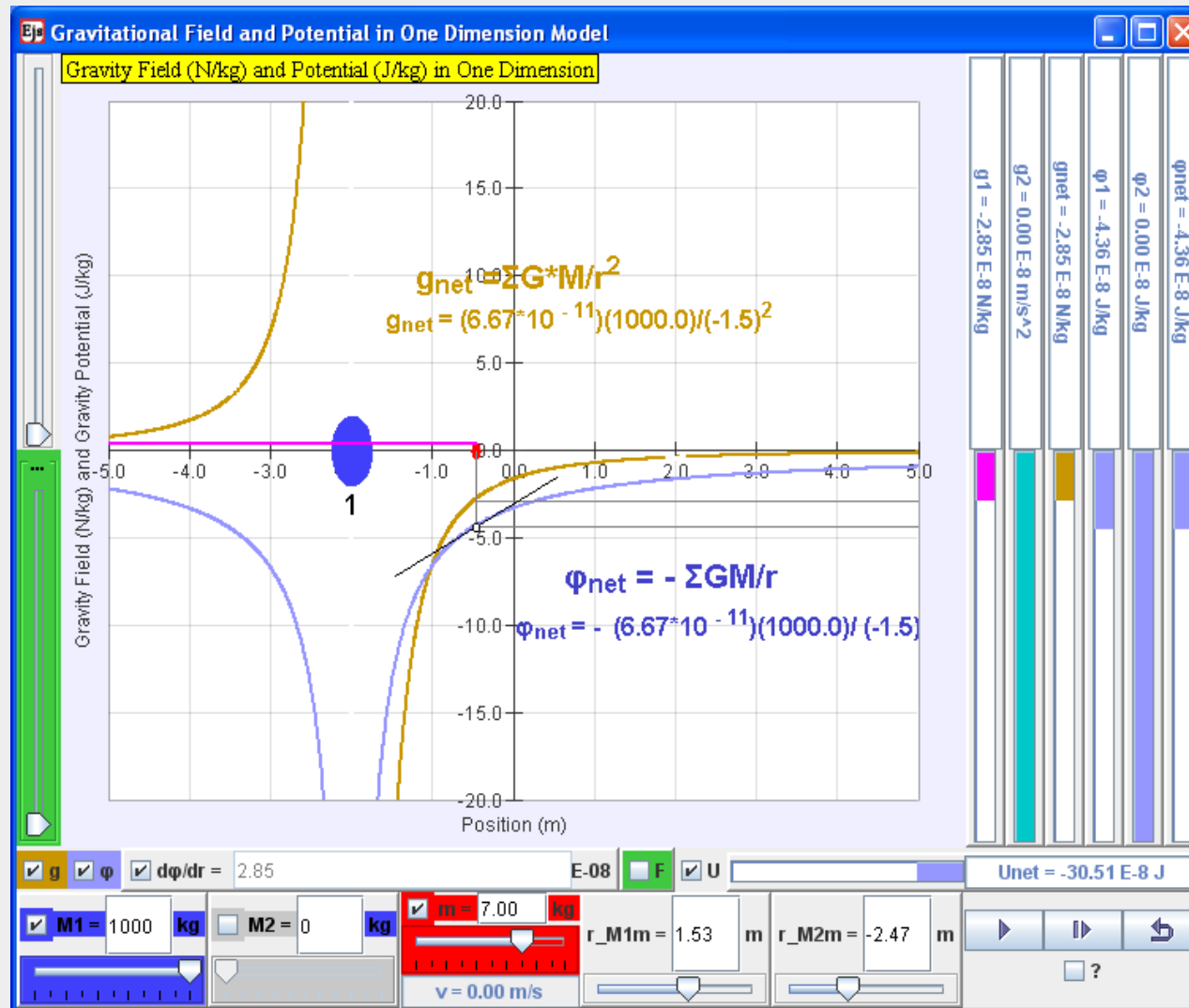
Module Topics

1. Virtualization
- 2. Java and the Java Virtual Machine**
3. The Java Ecosystem
4. The JVM Architecture
5. Varieties of JVMs

Origins of Java

- Java was originally designed to run in a browser
 - Design principle: *write once, run anywhere*
 - Safety was essential – execution of Java code had to be locked down
 - *This is the Java sandbox model*
 - *The JVM did not access the host OS or resources directly*
 - *The JVM only had restricted access*
 - *Eg. could not write to the file system*
- These requirements led to the the VM model
 - Borrowed the idea from other virtualization projects

Java Applet



The Original Sun Model

- Called the “Cathedral and Bazaar”
 - Java development was placed in a separate environment from SUN's main business
 - The specifications for the JVM and Java Language were made freely available
 - Anyone who wanted to could develop and distribute their own version of a JVM and/or Java
 - *But SUN trademarked the name and only allowed it to be used by those products that adhered to the standard*
 - *Notable case: Sun sued MicroSoft for its Java version J++*
 - *Sun won and MicroSoft could not refer to J++ as “Java” -- it later became C#*

The Java Community Process

- SUN implemented a “Community Development Process”
 - The JCP would decide on the direction of Java development
 - Made up of Java users and developers (individuals and organizations)
 - They took Java in a new direction
 - *As a way to connect web applications to existing legacy systems*
 - *Required the use of Java as a stand alone language*
 - *And required the use of Java on the server side*
 - This led to the development of Java Enterprise Architecture
 - *Not of interest to us*
 - *But it did require more robust formulation of the JVM*

The Java Community Process

For the *community*, by the *community*




- The Java Community Process (JCP) is the open, inclusive process to develop and revise Java technology specifications (JSRs), reference implementations (RI), and technology compatibility kits (TCK).
- The JCP program has over 1,000 corporate, individual and Java User Group (JUG) members, & 12,000 registered users.
- More than 350 Java technology specifications are in development in the JCP program; ~two thirds have reached Final Release.
 - Java EE 7 Final Release 2013.
 - Java SE 8 Final Release 2014.
 - Java ME 8 Final Release 2014.

The State of Java

- SUN was purchased by Oracle in 2010
 - Created concerns about the open model of Java
 - OpenJDK is a GPL fork of Java and the JVM supported by Oracle
 - Oracle is moving to a subscription model of Java
 - The future of Oracle Java is uncertain
- However, there are many other JVMs available
 - And the JVM is no longer just for Java
 - Other languages now use the JVM
 - *New languages like Scala, Kotlin*
 - *Ports of legacy languages like COBOL*
 - *Used by modern Big Data apps like Hadoop and Spark*

For this Class

- The Java Model has diverged
 - Java 8 is the most common version in use
 - Oracle started its own Java fork with Java 9
 - Most of the changes do not affect the JVM
 - To keep it simple, we will be using Java 8
- Almost all of what we do is generic
 - Should be applicable to most JVMs “out there”

A top-down view of a wooden desk. In the top right, a portion of a silver laptop is visible, showing keys like 'tab', 'Q', 'W', 'E', 'caps lock', 'A', 'S', 'Z', 'fn', 'control', and 'option'. Below the laptop, a pair of black-rimmed glasses lies horizontally. To the right of the glasses is a white ceramic cup filled with dark coffee. In the top center, a small green succulent plant is in a pot. The background is a light-colored wooden surface with a prominent grain.

Module Topics

1. Virtualization
2. Java and the Java Virtual Machine
- 3. The Java Ecosystem**
4. The JVM Architecture
5. Varieties of JVMs

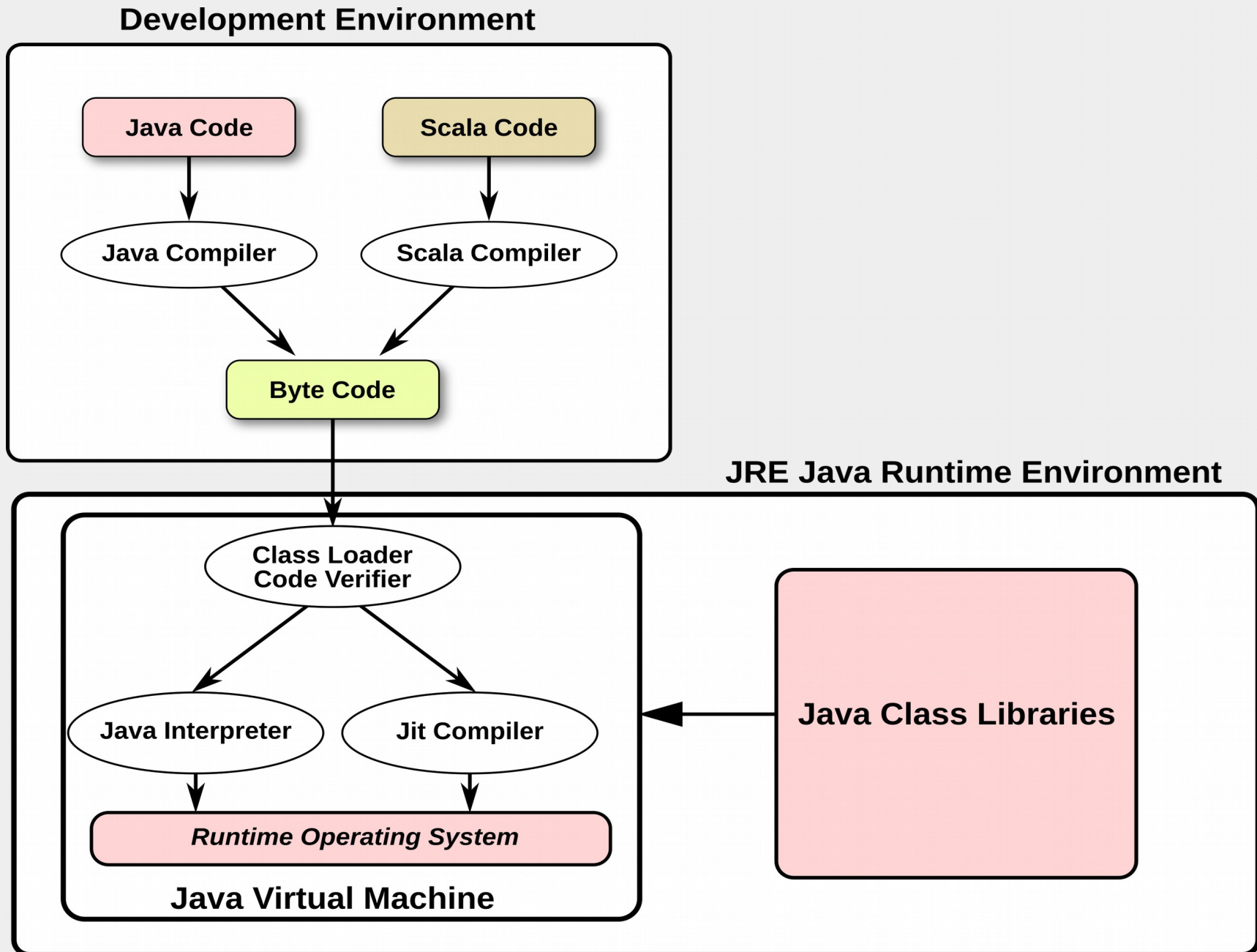
Java “Ecosystem”


- Distributed over three parts
- The Java production environment
 - Where programs are written using the JDK tools and utilities
 - Multiple compilers available from different languages
 - All produce class files consisting of “byte code”
 - Byte code = assembler instructions for hypothetical JVM
- The JRE – Java Runtime Environment
 - OS and architecture specific
 - Consists of the JVM and class libraries

Java “Ecosystem”

- The Java Class libraries
 - Standard Java libraries for a specific architecture
 - Class files need to be linked to library files
- The JVM
 - Class Loader links the class files with library files
 - Controls the execution of the loaded application
 - Translates the byte code to platform specific instructions
 - Just in Time compiler compiles code to improve performance
 - *We will look at this architecture in more detail later*

Java “Ecosystem”



A top-down view of a wooden desk. In the top right corner, a portion of a silver laptop is visible, showing keys like 'tab', 'Q', 'W', 'E', 'caps lock', 'A', 'S', 'Z', 'fn', 'control', and 'option'. Below the laptop, a pair of black-rimmed glasses lies horizontally. To the right of the glasses is a white ceramic cup filled with dark coffee. In the bottom right corner, the top edge of a black tablet or smartphone is visible. At the top center, a small green succulent plant in a dark pot sits on the desk. The wood grain of the desk is prominent, running horizontally.

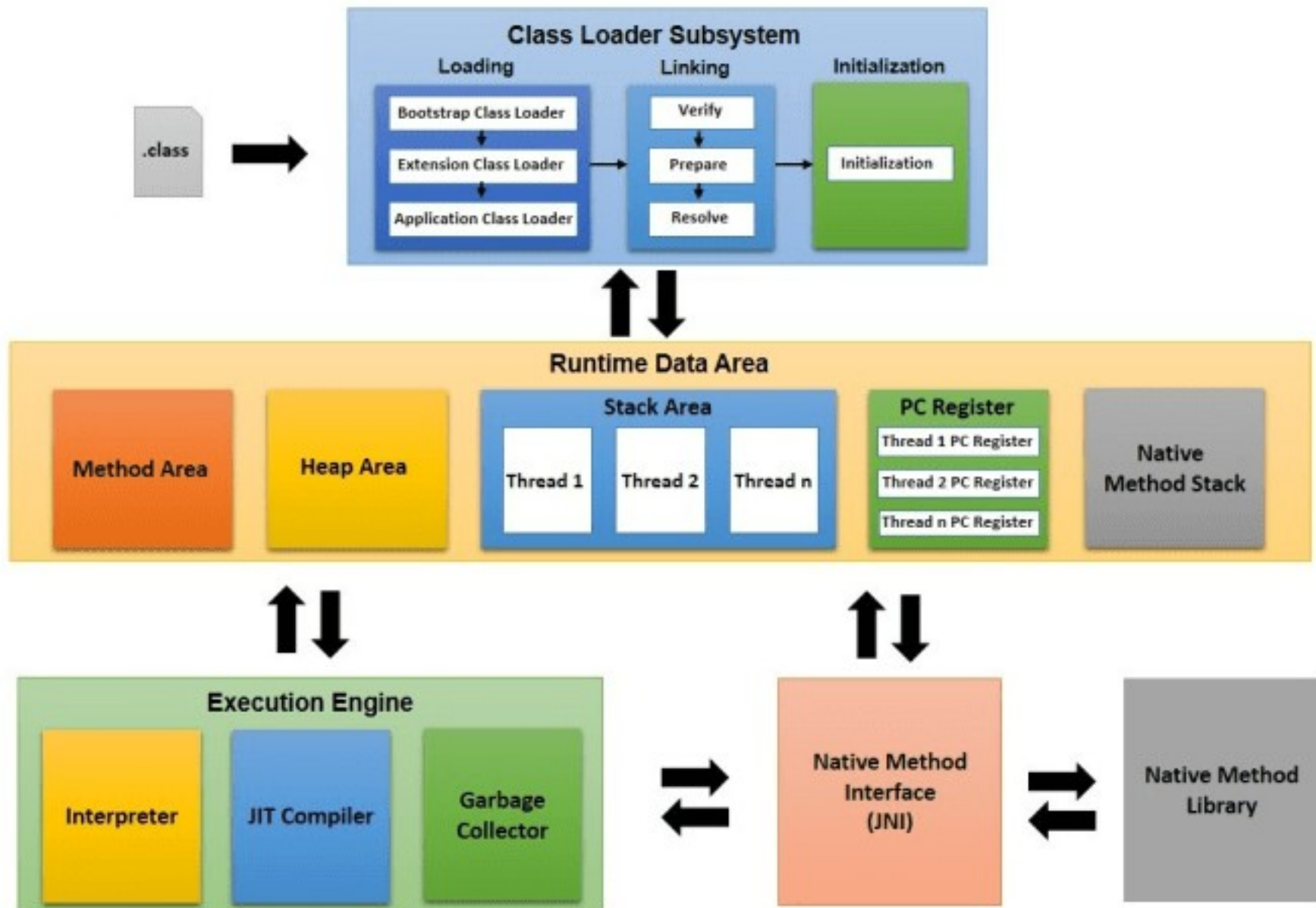
Module Topics


1. Virtualization
2. Java and the Java Virtual Machine
3. The Java Ecosystem
- 4. The JVM Architecture**
5. Varieties of JVMs

JVM Architecture

- The JVM emulates standard OS architecture
- Memory is divided into heap and stack
 - Managed exclusively by the JVM
- Supports multi-threading
 - Maintains thread specific stacks and registers
- JVM can link with and call native methods and code
- Execution area handles translation of byte code to the runtime OS environment

JVM Architecture



A top-down view of a wooden desk. In the top right, a portion of a silver laptop is visible, showing keys like 'tab', 'caps lock', 'shift', 'control', 'fn', 'option', 'alt', 'Z', 'A', 'S', 'W', 'E', 'Q', '1', '2', '3', '4', '5', '6', '7', '8', '9', '0', 'ESC', and 'F12'. Below the laptop, a pair of black-rimmed glasses lies horizontally. To the left of the glasses is a small, green, succulent-like plant in a dark pot. Below the glasses is a white ceramic cup filled with dark coffee, with a yellow handle. In the bottom right corner, the top edge of a black tablet or laptop is visible. The background is a light-colored wooden surface with a prominent grain pattern.

Module Topics


1. Virtualization
2. Java and the Java Virtual Machine
3. The Java Ecosystem
4. The JVM Architecture
5. **Varieties of JVMs**

Why Multiple JVMs?

- SUN/Oracle support a JVM specification
 - Describes how a JVM should operate
 - Does not describe how a JVM should be built
 - Spec describes the JVM interface, not its internals
 - Anyone can build a JVM
- The standard reference implementation is the Oracle supported “HotSpot” JVM
 - Evolves with revisions of the Java versions
 - Most implementations are similar to the reference implementation

Why Multiple JVMs?

- SUN/Oracle support a JVM specification
 - Describes how a JVM should operate
 - Does not describe how a JVM should be built
 - Spec describes the JVM interface, not its internals
 - Anyone can build a JVM
- The standard reference implementation is the Oracle supported “HotSpot” JVM
 - Evolves with revisions of the Java versions
 - Most implementations are similar to the reference implementation

A top-down view of a wooden desk. In the top right, a portion of a silver laptop is visible, showing keys like 'tab', 'caps lock', 'shift', 'fn', 'control', 'option', 'Q', 'W', 'E', 'A', 'S', 'Z'. Above the laptop is a small green succulent in a pot. Below the succulent are a pair of black-rimmed glasses. In the bottom right, there is a white mug filled with dark coffee and a yellow handle. A dark tablet or laptop lid is visible at the very bottom right. A semi-transparent dark rectangle with the text 'Module End' is positioned on the left side of the desk.

Module End