320341 Programming in Java

http://cnds.eecs.jacobs-university.de/courses/java-2015

JACOBS
UNIVERSITY

Fall Semester 2015

Lecture 1: Introduction

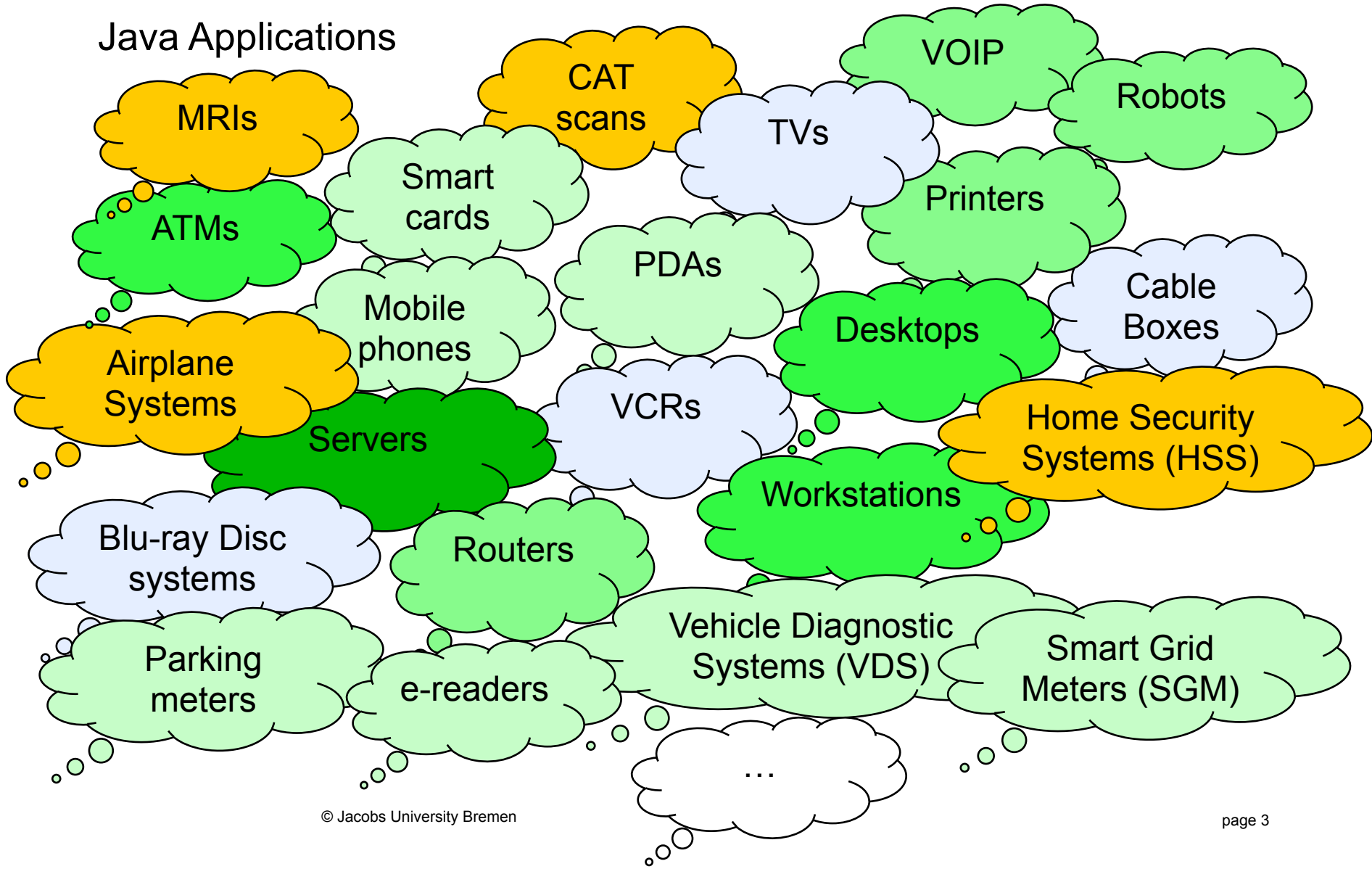
Instructor: Jürgen Schönwälder

Slides: Bendick Mahleko

Outline

- Motivation
- Buzzwords From The Java “White Paper”
- History of Java
- Java Compiler
- Java Runtime Environment
- Language Features
- Homework

Motivation



Motivation

JACOBS UNIVERSITY

Java technology has integrated itself into people's everyday lives

- *There is a high demand for people with Java skills*
- *Java programs run on many devices from cell-phones to servers to clouds*

Some statistics (Meloan, 2012)

- *97% of enterprise desktops run Java*
- *3 billion devices run Java*
- *5 billion Java cards in use*
- *80 million Java-enabled TV devices run Java*
- *9 million Java developers worldwide*
- *100% Blue ray disk players ship with Java*
- *89% of PC desktops run java*
- *Java is the most widely used software development language in the world*

Motivation

JACOBS UNIVERSITY

Java language positives

- High quality and secure execution environment
- Vast library (graphics, networking, databases, collections, multi-threading etc)
- Pleasant syntax
- Comprehensible semantics

Java language Overview

JACOBS UNIVERSITY

Java language has two parts

CORE LANGUAGE (e.g., int, arrays, objects) **+ LIBRARIES**

- The **Core Language** is simple enough to run on mobile phones
- Large collection of standard **libraries** provide “off the shelf” code

Buzzwords

Simple	Portable
Object Oriented	Interpreted
Network savvy	High Performance
Robust	Multithreaded
Secure	Dynamic
Architecture Neutral	

(Gosling, 1995)

Buzzwords

JACOBS UNIVERSITY

Simple

- Core language is easy to program

Object Oriented

- Fundamentally based on the OO notions of classes and objects

Network Savvy

- Extensive library supporting network programming (e.g., TCP/IP, HTTP, FTP)

Buzzwords

JACOBS UNIVERSITY

Robust

- Robust against unintentional errors and malicious code

Secure

- Designed to prevent certain types of attacks (e.g., memory corruption)

Architecture Neutral

- Java compiler generates architecture neutral object file format (bytecode)
- Bytecode can be run anywhere where there is a Java runtime system

Buzzwords

JACOBS UNIVERSITY

Portable

- Java was designed to do “Write Once Run Anywhere” - WORA

Interpreted

- The bytecode is interpreted to the target machine language

High performance

- Interpretation is slow but performance is improved if **JIT** compiler is used to compile “hotspots” of program into machine language and caching it

Buzzwords

JACOBS UNIVERSITY

Multithreaded

- Ease of multithreading in Java makes it attractive for server side development

Dynamic

- Useful for adding code while program is running

History of Java

JACOBS UNIVERSITY

1991

- Sun's internal research project (code-name "Green") to *develop a language for intelligent consumer devices*
- Language Oak was developed based on C++
- Oak renamed to Java

1994

- Java team saw potential of Java on WWW
- Built the "HotJava browser " to demonstrate power of Java
- Inspired the current 'Java crazy'

1996

- First version of Java (Java 1.0) was released
- Found to be very limited; This was followed by version 1.1, but still limited

History of Java

JACOBS UNIVERSITY

1998

- Java 1.2 was released at the JavaOne conference
- Sun's marketing named it

  "Java 2 Standard Edition Software Development Kit Version 1.2"

- Two other additions were introduced
  - "Micro Edition" (Java ME) for embedded devices
  - "Enterprise Edition" (Java EE) for server side programming

Note: Now we have:

- *Java Micro Edition (Java ME)*
- *Java Standard Edition (Java SE)*
- *Java Enterprise Edition (Java EE)*

History of Java

JACOBS UNIVERSITY

Evolution of Java Language

- Version 1.5 was called Version 5.0

- Version 5.0 was the first version to update Java language in a significant way
  - It was introduced in 2004
  - Generic classes, enhanced for-loop, auto-boxing, annotations

- Version 6 : several enhancements (library) over 5.0 version

- Version 7:
  - Released on 28 August 2011
  - Small language enhancements and library improvements
  - New I/O (asynchronous I/O)
  - Fork/ Join framework

History of Java - Summary

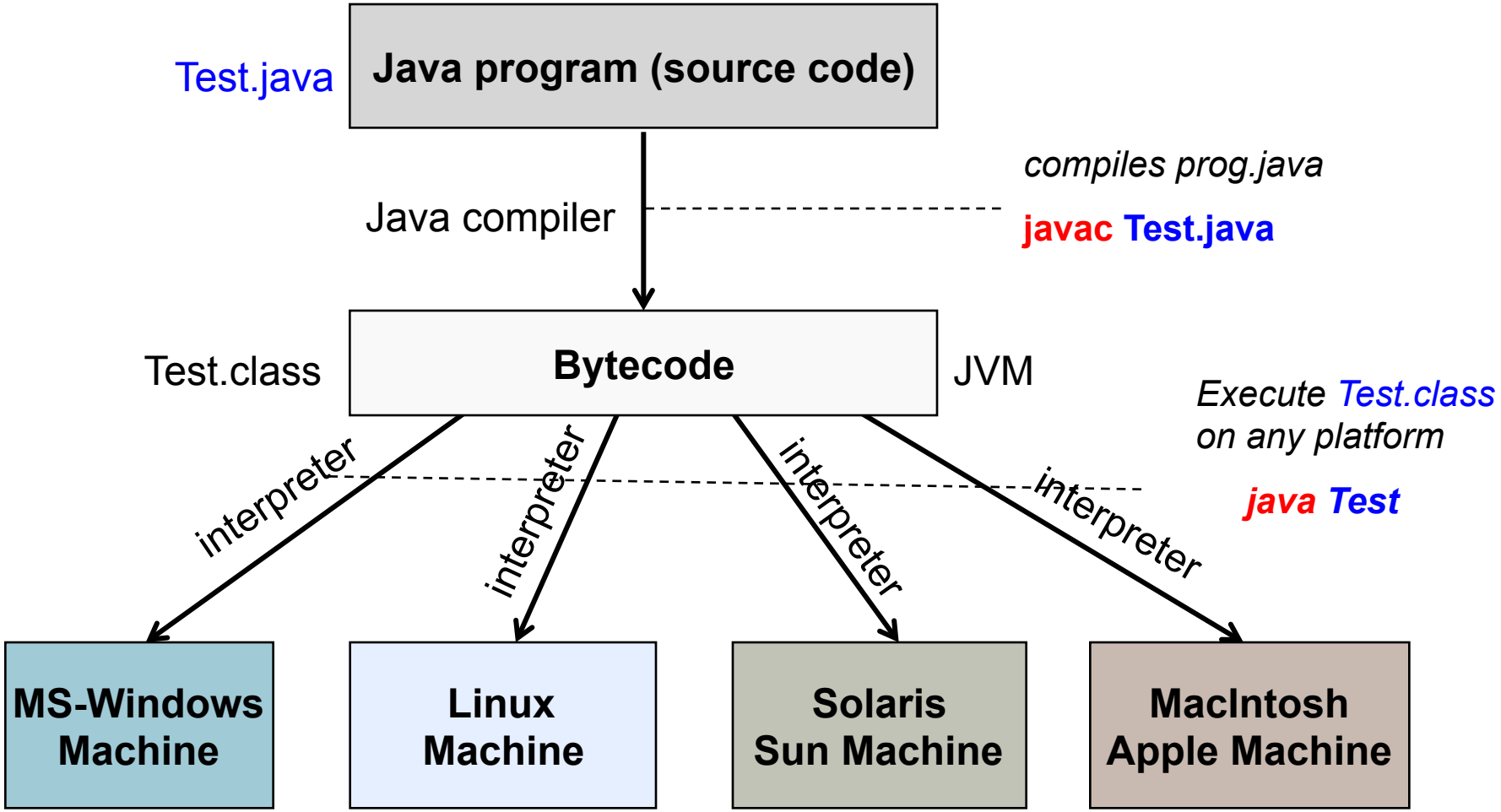
JACOBS UNIVERSITY

Java Version	#Classes	Features & performance
Java 1.02 (1st official release)	250	•Slow •Many bugs •Applets the big thing
Java 1.1	500	•A little faster •More capable & friendlier •Becoming very popular
Java 2 (versions 1.2 – 1.4)	2300	•Much faster (sometimes run at native speeds) •Powerful – 3 flavors: J2ME, J2SE, J2EE •Language of choice for Web-based enterprises & mobile applications
Java 5.0 (version 1.5)	3500	•More power, easier to develop with •Major changes to language & new features

History of Java - Summary

Java Version	#Classes	Features & performance
Java 6 (version 1.6)	-	•Performance enhancements •XML SOAP-based Web Services (JAX-WS) •JDBC4 (enhancements) •Console class •Swing GUI improvements
Java 7 (version 1.7)	-	•Language improvement •Automatic resource management •New file system API •Fork and Join Framework
Java 8 (version 1.8)		•Support for Lambda expressions (closures) •Add bulk data operations for Collections •Add new date, time, and calendar API •Add parallel sorting of arrays •new API for Base64 encoding and decoding • …

Java Compiler



Java Compiler

JACOBS UNIVERSITY

Compiling Java sources

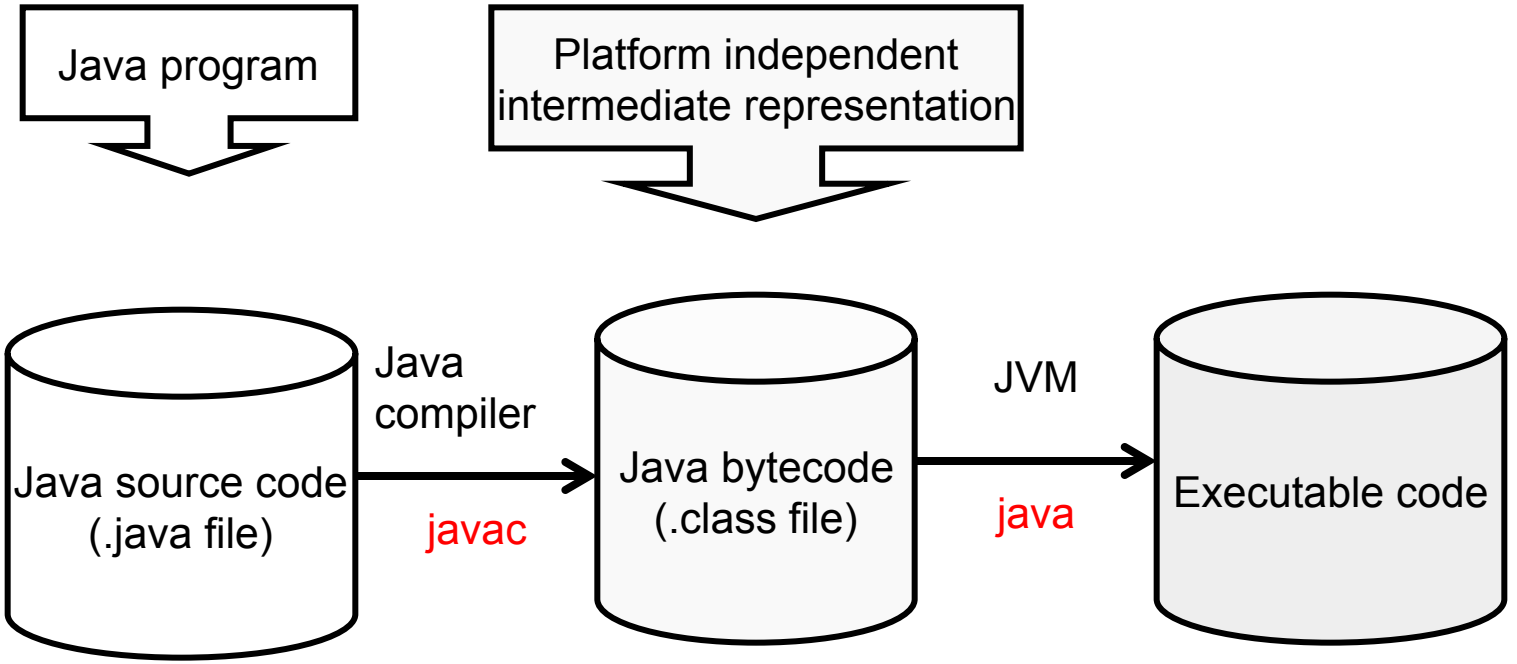
- Source code for each class is saved in a **`.java`** file
- Compile each class to produce a **`.class`** file
- Multiple classes can be packed together in a **`.zip`** or **`.jar`** archive file
- The Java compiler is called "**`javac`**"
- To compile all programs in a directory use "**`javac *.java`**"

Bytecode

- A compiled class is stored in a **`.class`** or **`.jar`** archive file
- It is **`machine neutral object`** and file executable on any machine with a virtual machine

Java Compiler

Compiling Java sources



Java Architecture

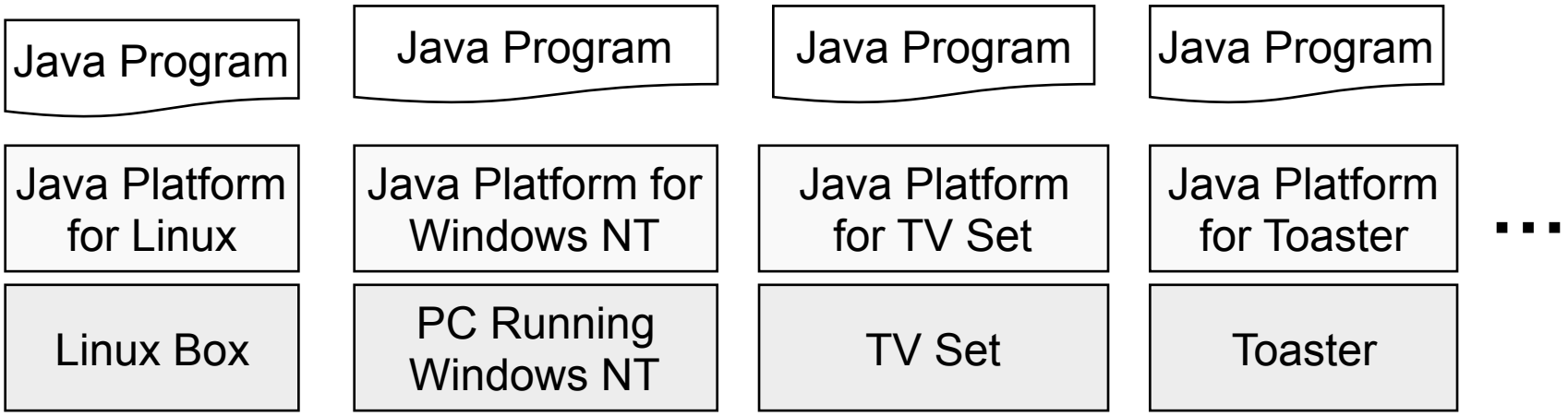
The Java architecture consists of four components

1. Java Programming Language
2. Java Class File format

3. Java API (Application Programming Interface)
   - ❑ Prewritten code that is organized into packages
   - ❑ Java API is divided into three main platforms
     a) Java 2 Platform, Standard Edition (J2SE now called Java SE)
     b) Java 2 Platform, Enterprise Edition (J2EE now called Java EE)
     c) Java 2 Platform, Micro Edition (J2ME now called Java ME)

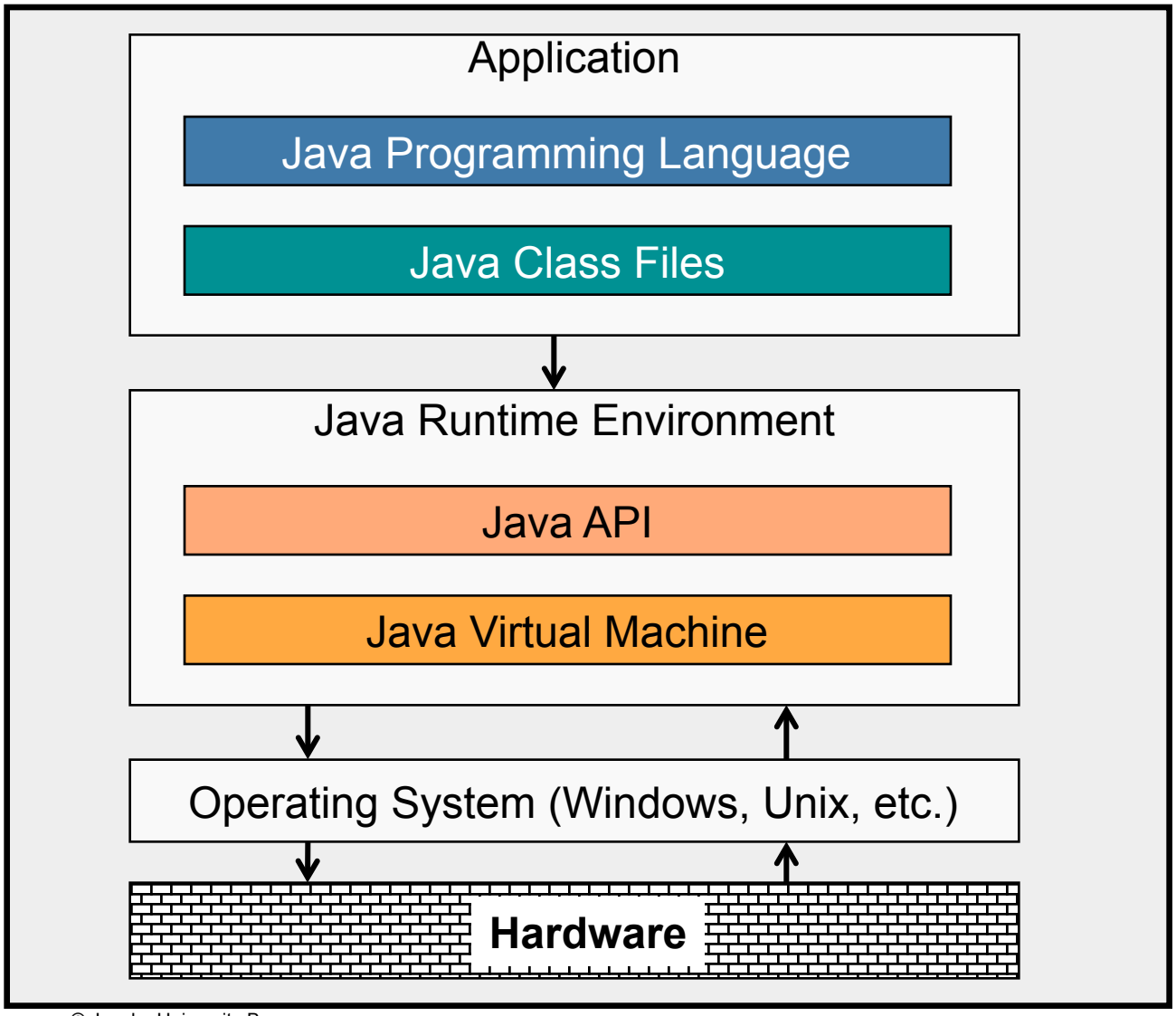
4. Java Virtual Machine (JVM): an abstract computing machine that interprets compiled Java programs

Java Architecture



- **Java platform = Java API + JVM ➔ Java Runtime Environment**

Java Architecture



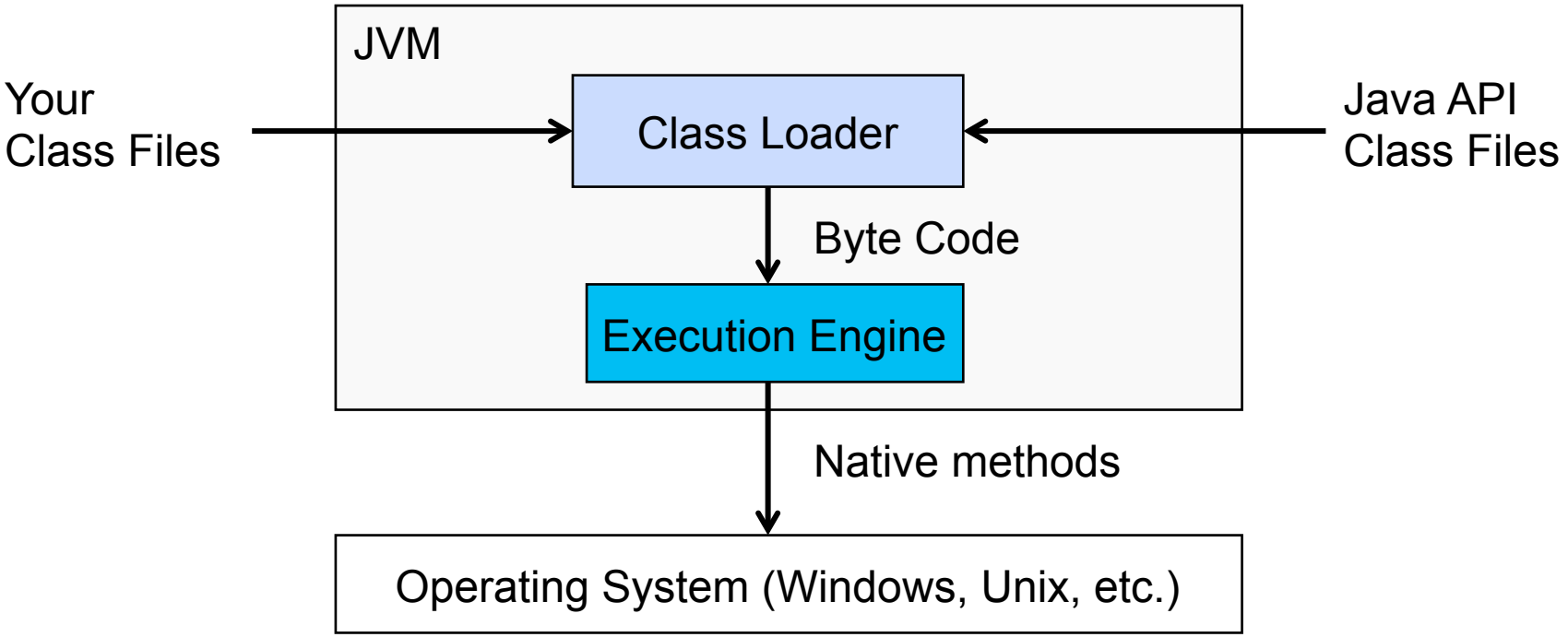
Java Virtual Machine

JACOBS UNIVERSITY

JVM is an *abstract computing machine* that interprets compiled Java programs

JVM specification describes features that every VM should have

JVM loads class files and executes **bytecodes** they contain



Java Virtual Machine

JACOBS UNIVERSITY

The implementation of an execution engine (EE) varies

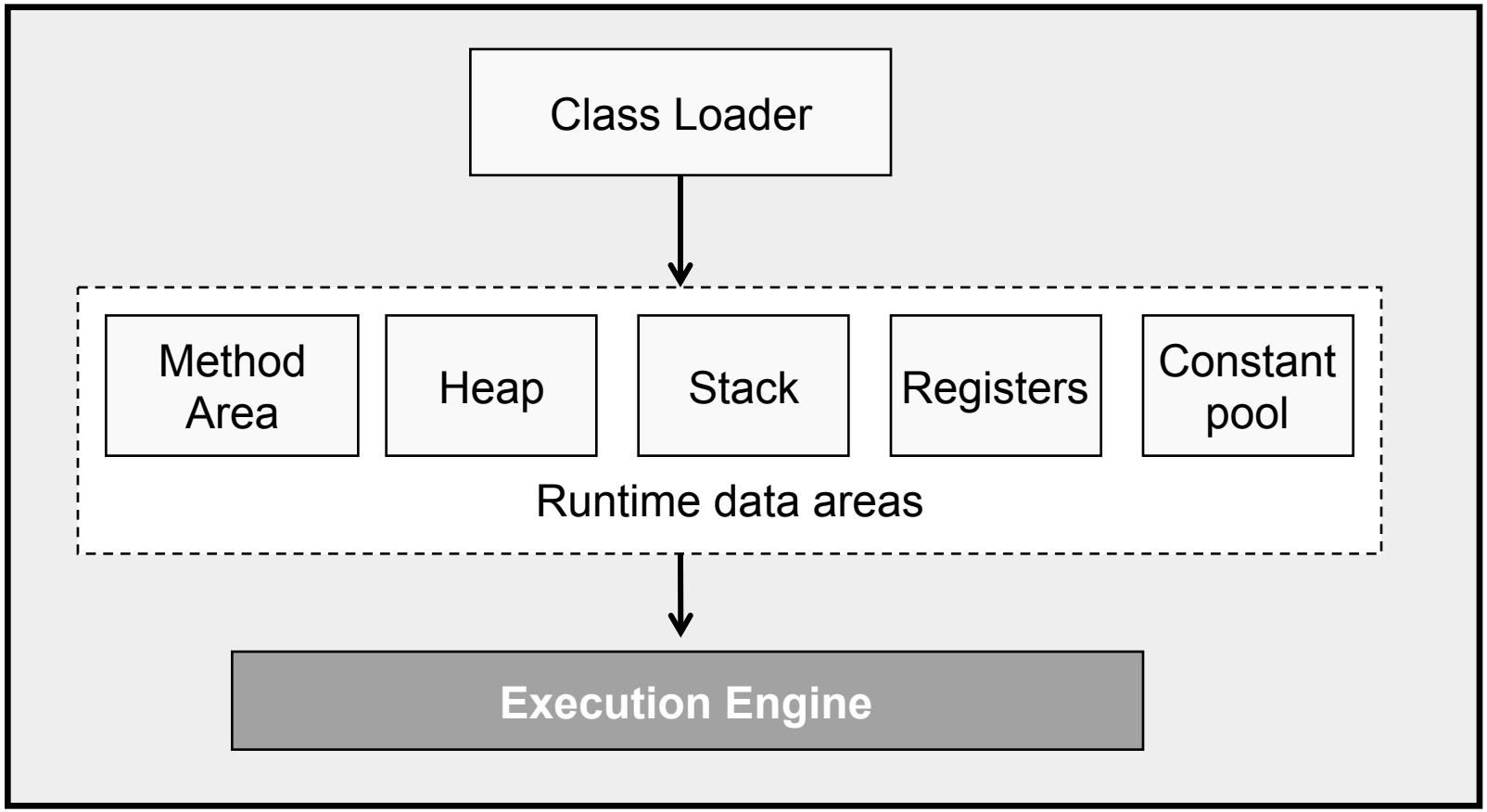
- Interpret **bytecode** one at a time (slow)
- Just in time (**JIT**) compiler (faster, but requires more memory)
- **Adaptive optimizer** (VM monitors “hotspots” compiles them to native code)
- VM built on top of a chip (exec bytecode natively – EE embedded in chip)

Java Runtime Environment

JACOBS UNIVERSITY

JVM needs memory to store temporary data related to code execution

The following components are provided (Spell, 2005)



JVM Runtime Data Areas

JACOBS UNIVERSITY

Heap

- Region of free memory often used for dynamic or temporary allocation

- Provides memory for class and array objects

- Heap memory is reclaimed when reference to an object or array no longer exist – collected by a **garbage collector**

- Programmer maybe allowed to specify initial size of heap (use –mx on Win32 and Solaris)

- OutOfMemoryError exception is generated if the heap runs out of memory

(Spell, 2005)

JVM Runtime Data Areas

JACOBS UNIVERSITY

Stack

- The Stack frame stores the state of method invocations
- Includes data and partial results, local variables and operand stack
- Operand stack stores parameters and return values for most bytecode instr
- Frames makeup JVM stack & store partial results, data and return values
- A Frame is created when a method is invoked and destroyed when a method exists
- StackOverflowError exception generated if a computation requires larger stack than provided (Spell, 2005)

JVM Runtime Data Areas

JACOBS UNIVERSITY

Method Area

- Common storage area shared among all JVM threads
- Stores: method data, field data, bytecode for methods and constructors

Registers

- Reflect current state of machine and updated as bytecode is executed
- Primary register is program counter (pc register)

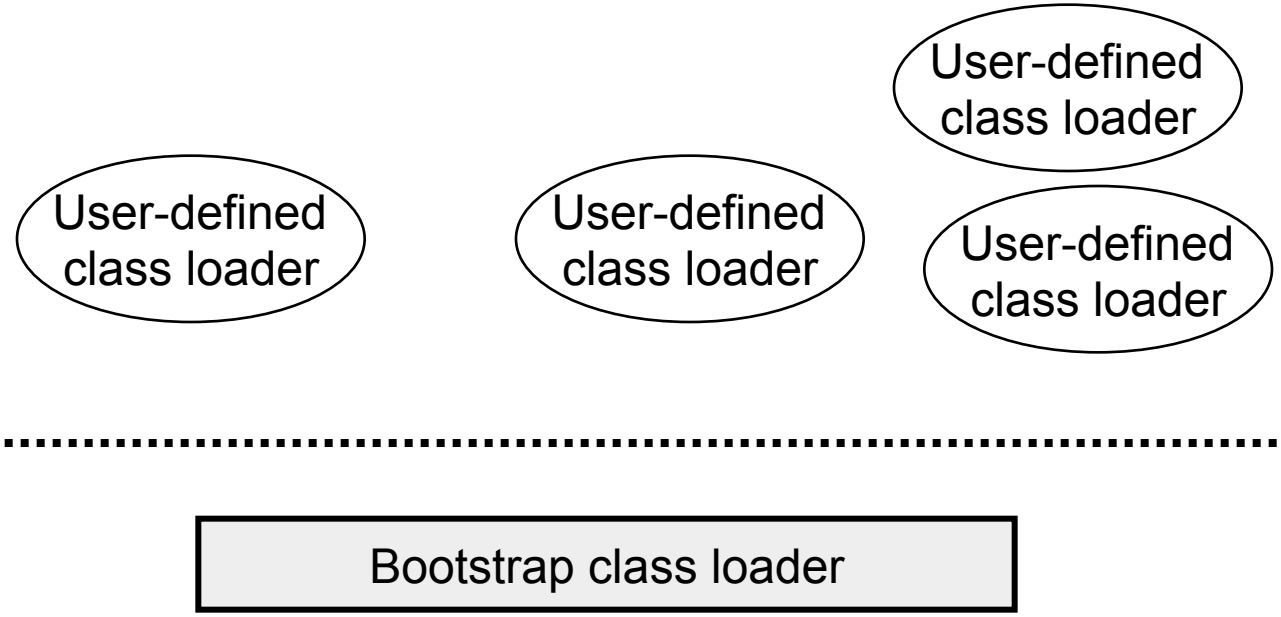
Runtime Constant Pool

- Contains constants including numeric literals and field constants
- Constructed when JVM loads the class file (Spell, 2005)

Java Runtime Environment

Class Loader

- There can be **more than one class loader inside JVM**
- Two types of class loaders
  - **Bootstrap class loader** (There is only one!)
  - **User-defined class loader**



JACOBS UNIVERSITY

Bootstrap Class Loader

- The Bootstrap Class Loader is part of the JVM implementation

- It is also called **primordial class loader, system class loader, default class loader**

- It is the default class loader (loads the Java API classes)

JACOBS UNIVERSITY

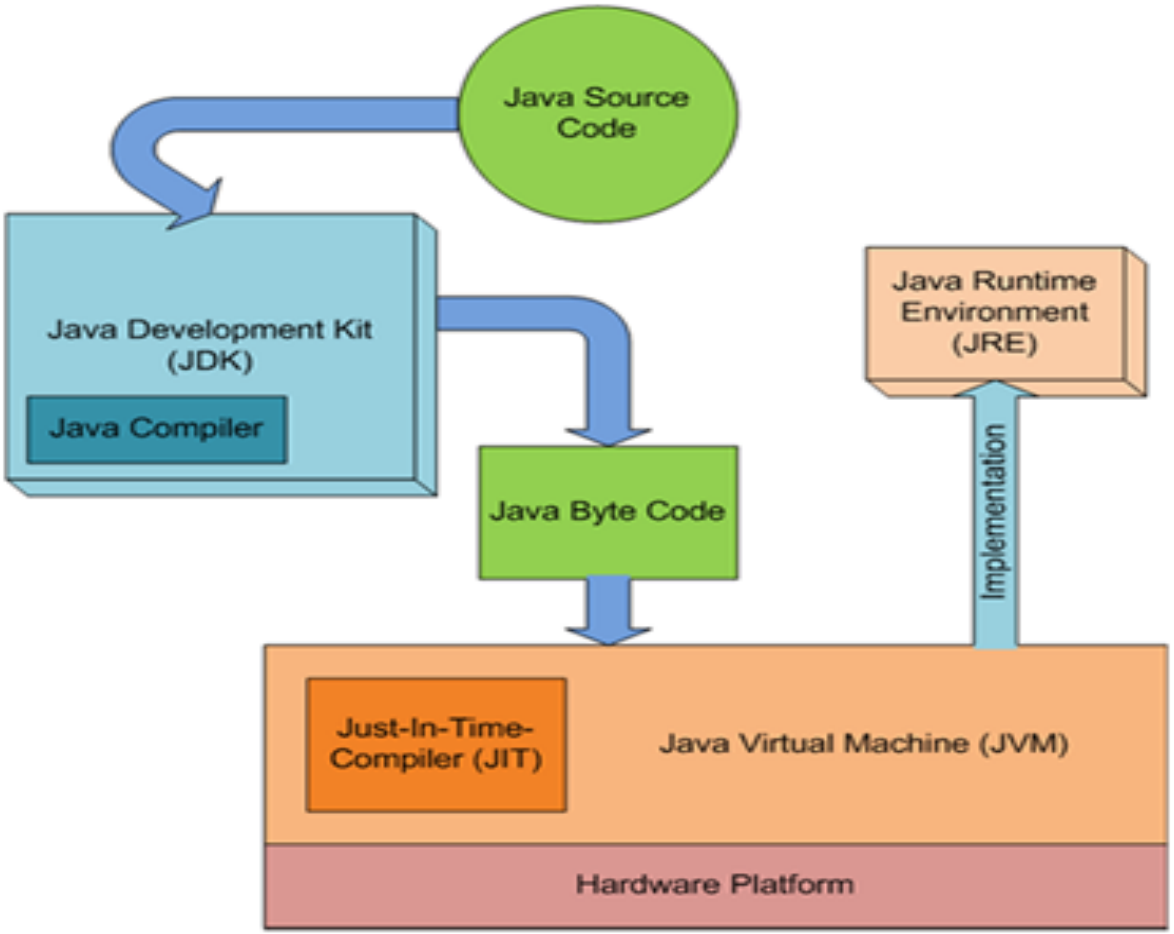
Bootstrap Class Loader

- The Bootstrap Class Loader is part of the JVM implementation

- It is also called **primordial class loader, system class loader, default class loader**

- It is the default class loader (loads the Java API classes)

JRE as an implementation of JVM



Source: http://javapapers.com/core-java/differentiate-jvm-jre-jdk-jit/

Questions?

Reading Material

Radhakrishnan, R., Vijaykrishnan, N., John, L. K., Sivasubramaniam, A., Rubio, J. and Sabarinathan, J.: Java Runtime Systems: Characterization and Architectural Implications. IEEE Transactions on Computers, 50(2), 2001.

Java Products [Online] Available from: http://www.oracle.com/technetwork/java/index.html (Last accessed: 6th September 2012)

Gosling, J. (1995) Java: an Overview [Online] Available from: http://www.cs.dartmouth.edu/~mckeeman/cs118/references/OriginalJavaWhitepaper.pdf (Last Accessed: 6th September 2012).

Java. APress. Available at: http://www.maspick.co.il/Ddd/Apress%20-%20Pro%20Java%20Programming,%202nd%20Edition.pdf

Meloan, S.(2012) JavaOne 2012 Review: Make the Future Java. Retrieved 14 August 2013 from http://www.oracle.com/technetwork/articles/java/javaone12review-1863742.html.