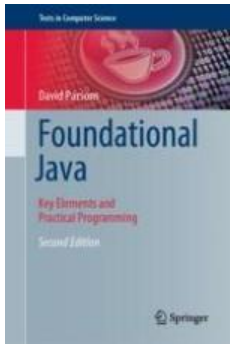


Chapter 1
The Java Story

Foundational Java

Key Elements and Practical Programming

Java

- Major global programming language
  - desk top, web and enterprise systems
  - embedded in mobile phones, Blu-ray players etc.
- ‘virtual machine’ supports a family of related languages
  - Scala, Groovy, versions of Ruby and Python
- Came to wide attention via the web in 1995
  - Popularity riding on the explosion of interest in the Internet and World Wide Web in the mid 1990s

A Brief History of Java

- Started as Sun Microsystems project to build the ‘Star7’ Personal Digital Assistant (PDA)
  - Intended to control all the electronic devices in the home
  - Needed to work on various pieces of hardware, from televisions to toasters
- The Star7 language was called ‘Oak’
  - Later Changed to Java for trademark reasons
- Moved onto the web with the ‘HotJava’ browser
  - able to run small Java programs (known as ‘applets’) within its window
- Java has since matured into one of the key technologies of global software development

Characteristics of Java

- simple
- object-oriented
- distributed
- robust
- secure
- architecture neutral
- portable
- high performance
- multithreaded
- dynamic

Simple

- '*C++ without the knives, guns and clubs*'.
- One major simplification in Java is the way that memory is managed, using largely automatic processes rather than requiring the programmer to do this

Object-oriented

- Object-oriented languages have become common since the 1990s
- Data and processes are ‘encapsulated’ together to provide objects that have both state (data) and behaviour (processes)
- Easier to model the behaviour of the real-world things that we are trying to reflect in software

Distributed

- Trend towards making the machine on the desk less important then the network it is connected to
- Java is designed for network programming:
  - Common Internet protocols
    - HTTP (Hyper Text Transfer Protocol)
    - FTP (File Transfer Protocol)
  - Socket communication
  - Remote method invocation (RMI)
  - Web services

Robust

- Does not behave unpredictably or fail due to programmer error
- Removes the risks of memory pointers from code:
  - Manipulating memory that has not been correctly allocated can crash a program
  - Failing to free up memory that has been finished with leads to ‘memory leaks’ where a program can eventually run out of memory
- Java has only ‘references’ to objects, not pointers
- Garbage collector automatically recovers memory from objects that are no longer needed

Secure

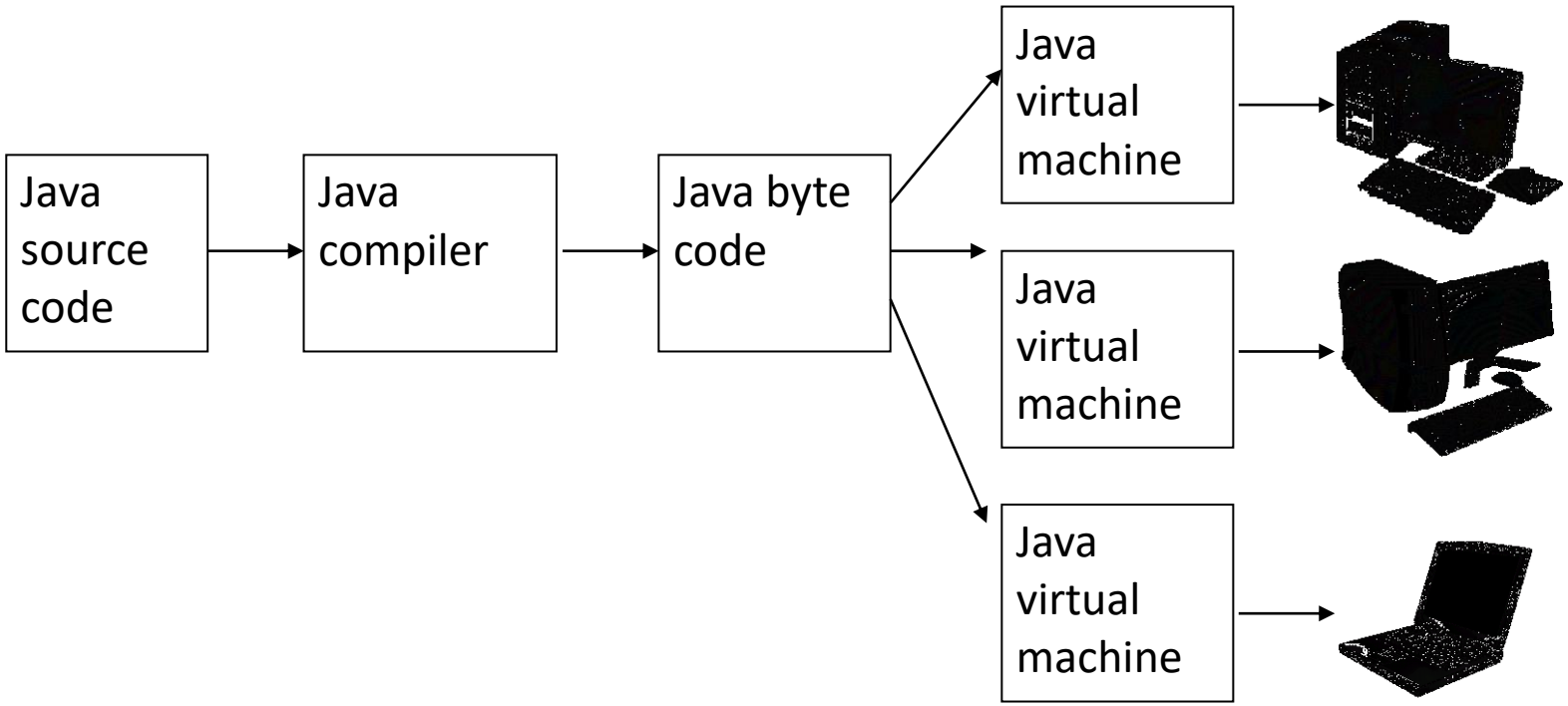
- Security systems built into Java ensure that the code, once written, is not easy to tamper with
- Restrictions placed on what Java applets can do when they are running inside browser

Architecture Neutral

- ‘write once, run anywhere’
- Combines a compiler and an interpreter
- The Java compiler does not convert the source code into an executable for a specific environment
  - it compiles into ‘byte code’
- Byte code can be run on a Java Virtual Machine (JVM), software that interprets the byte code to run on a specific computer
- The same byte code can run on different systems
- Each platform needs to have its own virtual machine

The Java Virtual Machine

- The Java virtual machine (JVM) allows the same byte code to be run on different platforms and operating systems.



Portable

- Architecture neutrality
- Portable definitions of data types
  - e.g. all ‘int’ data types are thirty-two bits long
- Java types are also always signed
  - can contain both positive and negative numbers
  - No ‘unsigned’ types

High performance

- Just-in-time (JIT) compilers speed up the interpretation process
- Native compilers
- Java chips
  - embedded in electronic devices
- HotSpot compiler optimises code while it is running

Multithreaded

- Java makes it easier for programmers to write multithreaded programs that are more efficient than single threaded programs (where only one thing can be happening at any one time)
- Even where the operating system itself is not multithreaded, Java code can be written that uses multiple threads of control

Dynamic

- Java can dynamically change the resources it is using at run time
- Useful in a distributed environment
  - Programs can be flexible in terms of size and behaviour
- Easy to locate objects at run time, even when they are in different places

The JDK and the JRE

- Java installations come in two forms
- Java Software Development Kit (JDK)
  - For writing Java programs
  - includes the Java compiler (javac)
- Java Runtime Environment (JRE)
  - includes the Java Virtual Machine and supporting libraries for running Java code
  - No compiler or other development tools
- A Java Integrated Development Environment (IDE) may include the necessary tools

Java Versions (1)

Version	Year	Highlights
1.0	1995	First public version
1.1	1996	Event handling mechanism for user interfaces
1.2 (Java 2 platform)	1998	Collections Framework, Swing GUI libraries. Java split into three editions
1.3	2000	Updates to existing features, including better sound support
1.4	2002	Performance and security improvements, XML processing, new input/output libraries
5.0 (change in version style)	2004	Generics, annotations, autoboxing, features similar to C#
6	2007	Web services, Java to XML Binding. Update 10 made major changes to JRE footprint.
7	2011	First Oracle and OpenJDK Community versions of Java

Java Versions (2)

Version	Year	Highlights
8	2014	lambda expressions, new date and time API
9	2017	Modules (Project Jigsaw)
10	March 2018	“var” keyword, type inference, release schedule changed to every 6 months (March and September)
11	September 2018	HttpClient, removal of Applet, Web Start and Java Enterprise Edition (Java EE) features
12	March 2019	Preview “switch” expression, changed licensing arrangements
13	September 2019	Preview of Text Blocks, revision of the switch expression
14	March 2020	New NullPointerException message, preview of “records”
15	September 2020	Second preview of changes to the “instanceof” operator, Text Blocks into permanent status.

Major Milestones

- Java became open source in 2006
- Sun Microsystems was acquired by Oracle Corporation in 2010

Java APIs

- Core Java syntax (keywords and fundamental libraries) + Java application programming interfaces (APIs).
- Some of the more specialised libraries (often known as the Java extensions) are not provided with the standard edition of the JDK
- Java Standard Edition, (Java SE)
  - Desktop applications, all the core libraries
- Java Enterprise Edition (Java EE)
  - client server programming
- Java Micro Edition (Java ME)
  - small or embedded devices

Learning Java

- To program in Java successfully we need to understand object-oriented concepts
- Java has a rich syntax and wide-ranging APIs
- It can be used for all kinds of programming, from writing a command line utility to building a distributed client server system or a complex multithreaded real-time system
- Continues to evolve and provides programmers with the tools for coding a host of applications in all kinds of contexts

Some Questions…

- What's the difference between the JDK and the JRE?
- What is byte code?
- If Java is cross platform, why do I have to download different versions of it for Windows, Mac, Linux etc?
- Why is Java popular as a programming language?