DeveloperFocus

Understanding Advanced Java & The Java Virtual Machine (JVM) with Ben Evans

Duration:	3 Days
Cost:	£1495.00 + VAT
Additional Requirements:	Laptop – see spec below

ABOUT THE COURSE

This 3 day course, derived from "The Well-Grounded Java Developer" by Ben Evans & Martijn Verburg, covers a wide range of topics to bring your skill levels up and get you thinking about more advanced coding concepts.

WHAT YOU WILL LEARN

We'll discuss the overall design of the JVM and the guiding principles which underlie it. We'll introduce the JVM's bytecode interpreter as a stack machine, and explain how to use javap to disassemble bytecode. We'll return to the subject of classloading, and consider it from the point of view of the VM, rather than the language.

Moving on, we'll begin to talk specifically about the HotSpot JVM (which forms the basis of both Oracle's JVM and the OpenJDK). We'll introduce OOPs as the runtime representation of Java objects, and discuss KlassOOPs as the VM's view of class metadata. This will set the stage for a full discussion of JIT compilation and the incredible optimization capabilities of HotSpot.

We'll finish off the course by returning to another topic from earlier in the course - reflection.

We'll look inside the platform to see how the reflection capabilities are implemented, to give developers much more insight into what the features actually do, and why a reflective call is so much slower than a regular version.

We'll work primarily with Java 6, although we'll do a couple of advanced topics which make use of new Java 7 features. There'll be plenty of hands-on exercises (at least 25% of the course), which are all derived from real codebases & interview questions.

See Full Course Outline below

WHO SHOULD ATTEND

Intended for experienced Java programmers, who want to get deeper with the platform, the course discusses the more advanced aspects of objects and collections, approaches to representing executing code (such as threadpools), reflection, classloading tools & monitoring.

In the second half of the course, we introduce the major subsystems of the JVM and practical ways to apply this knowledge in real applications.

PRE–REQUISITES

Intermediate Java

MACHINE SPECIFICATON

A recent Windows, Linux or Mac laptop, with Oracle JDK or OpenJDK 7 or 8 installed.

ABOUT THE INSTRUCTORS

Ben Evans

Ben has lived in "Interesting Times" in technology. Whether it's performance testing the auction servers for the Google IPO, building low-latency financial trading systems, writing award-winning websites for some of the biggest films of the 90s, simulating quantum phenomena now being observed at the Large Hadron Collider or helping to provide technology for some of the UK's most vulnerable people - Ben is always doing something interesting.

When he's not running jClarity, Ben spends his time getting involved in community and charity projects. He helps to run the London Java Community and holds a seat on the Java Community Process Executive Committee - the body which governs Java standards.

Ben enjoys fundraising and helping charities, and loves to travel, meet new people, and exchange knowledge and ideas with them. Ben holds a Masters degree in Mathematics from the University of Cambridge and dropped out of a PhD when he realised he was having more fun writing music websites.

Martijn Verburg

Martijn has over 12 years experience as a technology professional and OSS mentor in a variety of environments from start–ups to large enterprises. Some career highlights include overhauling technology stacks and SDLC practices at major IBs, mentoring large vendors on technical community management, and running large distributed development teams at a variety of organisations.

He is the co–leader of the London Java Community (over 2800 developers) and leads the global effort for the Java User Group "Adopt a JSR" and "Adopt OpenJDK" programmes. Martijn's book "The Well–Grounded Java Developer" (with Ben Evans) was published by Manning in 2012.

As a leading expert on technical team optimisation, his talks and presentations are in high demand by major conferences (JavaOne, Devoxx, OSCON, FOSDEM, QCon, etc). Often you'll find him challenging the industry status quo in his alter ego "The Diabolical Developer".

Martijn holds a BSc in Computer Science and a BCA in Information Systems from Victoria University of Wellington

COURSE OUTLINE

1. Introduction
   - History of Java & Implementations
   - The Language & The Platform (non-Java languages)
   - OpenJDK - Using the source code

2. Objects & Collections – Advanced
   - Type safety - serial & conecurrent
   - Visibility & Mutability
   - equals(), hashcode() and related topics
   - The "plurals" classes & handy static methods
   - Java 7: Handling Resources & NIO.2
   - Advanced Generics (type erasure, wildcards, practitioner / designer split)
   - Looking Ahead - Java 8 & Binary Compatibility

3. Representing Execution
   - Callable
   - Inner classes as "clumsy closures"
   - Runnable
   - BlockingQueue
   - Execution Services
   - Designing Multithreaded Apps
   - Looking Ahead - Java 8 Lambda Expressions

4. Reflection
   - Class objects
   - Reflection
   - Issues with Reflection

5. Classloading
   - The hierarchy of classloaders
   - Why you need custom classloading
   - PermGen: Why your App Server/Web Server is killing you

6. VisualVM and MXBeans
   - Introducing VisualVM
   - Basic Tasks
   - Useful Plugins
   - JMX and MXBeans

7. Design Goals
   - The Big Secret – The Power of Runtime Information
   - Security features (no pointer arithmetic, no direct access to memory)
   - Overview of the JVM design (state diagram)

8. The JVM Interpreter
   - The JVM as a Stack Machine
   - Introduction to bytecode
   - Using javap
   - Useful Command-line tools
   - Bytecode families
   - Examples

9. Classloading Revisited
   - The anatomy of a classfile
   - Verification of class files
   - The classloading process in detail

10. OOPs
    - Representing Objects in the heap
    - Mark and Class words
    - klassOops and Class objects
    - Lifecycle of an object
    - Looking Ahead - Java 8 & the removal of Permgen

11. Introduction to Garbage Collection
    - Basic Mark & Sweep
    - Weak Generational Hypothesis
    - The Modern JVM Heap
    - Introducing non-default collectors

12. JIT Compilation
    - Overview – C1, C2, vtables, pointer swizzling, etc
    - The Code Cache
    - Inlining
    - Monomorphic Dispatch
    - Introduction to other compilation features
    - Examining the generated machine code
    - The difficulties of microbenchmarking
    - Examples

13. Reflection Revisited
    - How the JVM handles reflection
    - Diagram of reflective call
    - Why reflection is slow
    - Java 7: Method Handles & invokedynamic