Java SE 7 & Java SE 8

Mark Reinhold

Chief Architect, Java Platform Group

5 October 2010

In a Nutshell

- Java SE 7 *Mid 2011*
  - > JSR 203: More New I/O APIs
  - > JSR 292: “InvokeDynamic”
  - > JSR TBD: Small Language Changes (“Project Coin”)
  - + A considered selection of smaller enhancements
- Java SE 8 *Late 2012*
  - > JSR 308: Type Annotations
  - > JSR TBD: More Small Language Changes
  - > JSR TBD: Lambda Expressions *etc.* (“Project Lambda”)
  - > JSR TBD: Java Native Module System (“Project Jigsaw”)

Java SE 7

- Themes
  - Productivity
  - Performance
  - Multiple Languages
  - Integration
- JSRs
  - JSR 203: More New I/O APIs
  - JSR 292: “InvokeDynamic”
  - JSR TBD: Small Language Changes (“Project Coin”)

SE 7: JSR 203: More New I/O

- What?
  - Asynchronous I/O API
  - Filesystem API
  - Socket options, multicast cleanups, …
- Why?
  - Performance
  - Integration

SE 7: JSR 292: “InvokeDynamic”

- What?
  - First new bytecode since 1995
  - Is of no benefit to the Java language itself
- Why?
  - Dramatically speeds up dynamic languages
    - Better raw performance
    - Higher scalability
    - Expect eventually to beat native Ruby & Python

SE 7: JSR TBD: Small Language Changes

- What?
  - Improved Type Inference for Instance Creation (“diamond”)
  - Enhanced Integer Literals
  - Strings in Switch Statements
  - Automatic Resource Management
  - Simplified Varargs Invocation
  - Multi-Catch
- Why?
  - Improve developer productivity

SE 7: Small enhancements (1/2)

- Core
  - Upgrade Class-Loader Architecture
  - Method to Close a URLClassLoader
  - Concurrency and Collections Updates
    - Fork/Join Framework, *etc.*
  - Unicode 6.0
  - Locale Enhancement
  - TLS 1.2
  - Elliptic-Curve Cryptography (ECC)
  - JDBC 4.1

SE 7: Small enhancements (1/2)

- Client
  - Translucent & Shaped Windows
  - Heavyweight/Lightweight Component Mixing
  - Swing: Nimbus Look-and-Feel
  - Swing: JLayer Component
- Web
  - Update the XML Stack
    - JAXP, JAXB, JAX-WS

Java SE 8

- Themes
  - Productivity
  - Performance
  - Modularity
- JSRs
  - JSR 308: Type Annotations
  - JSR TBD: More Small Language Changes
  - JSR TBD: Lambda Expressions *etc.* (“Project Lambda”)
  - JSR TBD: Java Native Module System (“Project Jigsaw”)

ORACLE

SE 8: JSR 308: Type Annotations

- What?
  - Allow annotations to be written in more syntactic contexts than today
- Why?
  - Further enables static program-checking tools

SE 8: JSR TBD: More Small Language Changes

- What?
  - Collection literals
  - Collection access syntax (perhaps)
- Why?
  - Improve developer productivity

SE 8: JSR TBD: Lambda *etc.*

- What?
  - Lambda expressions (*a.k.a.* “closures”)
  - Default methods
  - Bulk-data operations for the Collections Framework
- Why?
  - Dramatically simplify writing multicore-ready code
  - Modernize the Java language & libraries

SE 8: JSR TBD: Module System

- What?
  - A developer-friendly Java-native module system
    - High fidelity between compile time and run time
  - A (coarsely) modularized Java SE Platform
- Why?
  - Escape from “JAR hell”
    - Eliminate the class path!
  - Scale the platform down
    - To small devices
  - Enable significant performance improvements
    - Especially download and startup time

The preceding is intended to outline our general product direction. It is intended for information purposes only, and may not be incorporated into any contract. It is not a commitment to deliver any material, code, or functionality, and should not be relied upon in making purchasing decisions. The development, release, and timing of any features or functionality described for Oracle’s products remains at the sole discretion of Oracle.