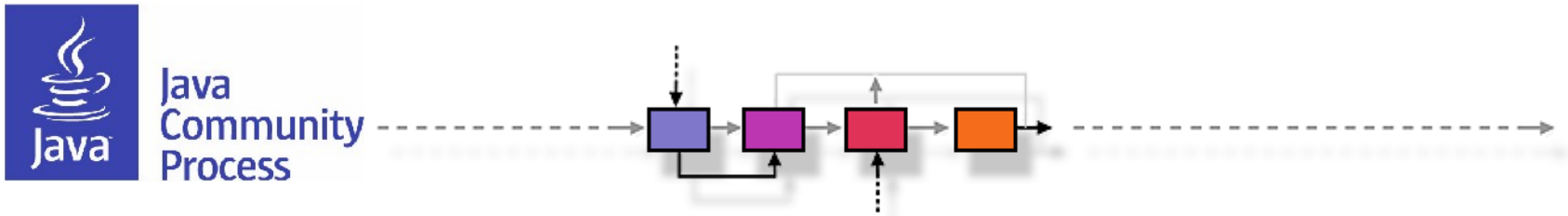


Platform Compatibility Requirements (what are all these Rules, anyway?)

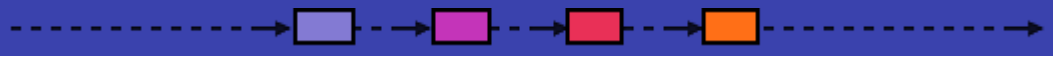
September 26, 2008
patrick@jcp.org

Sun microsystems

Vision

To promote the compatibility and interoperability of Java technology implementations by ensuring that the technologies are well specified and the implementations conform to the specifications.

Design Principles and Goals



- The principles of *Predictability* and *Transparency*, when applied to the Java language and the Java platform, lead to:
  - WORA (Write Once, Run Anywhere).
    - Well-written Java programs will execute with equivalent results on all compatible platforms.
  - WYGIWYS (What You Get Is What You See).
    - Developers and users clearly understand how their programs will behave.

Java Language Design Principles: Predictability

- Program behaviour is tightly specified.
  - Operates down to the bit level.
  - Floating point behavior, hash codes...
- Some performance cost, but worth it...

Java Language Design Principles: Transparency

- Focus is on creating easy-to-read source code.
  - It should be obvious from reading the source how a Java program will behave.
- Some useful features left out for this reason.
  - Preprocessor, macros, operator overloading...
  - Major gains in programmer productivity.
  - Keeps each project from "inventing its own language".

Java Platform Design Principles: Predictability

- All implementations contain same APIs.
- No subsetting or supersetting of standard, specified APIs.
- Behavior is fully specified.
- Class-file semantics are identical across implementations.

Java Platform Design Principles: Transparency

- Vendor value-add is clearly identified.
- No magic modes, no hidden behavior.
- Always compatible.

Results

- Compatible products available from multiple vendors.
- Portable applications run identically on all compatible products.
- Developers and customers understand what is part of the platform and what is vendor-specific.
- No surprises; Java programs and platforms behave as expected.

Program Overview

- The specification defines the requirements.
- The TCK tests whether implementations conform to the specification.
- The *Compatibility Rules* define testing requirements.
- Self-certification of compatibility.
  - Licensees run the tests and follow the rules.
  - They tell Sun when they're compatible.
  - Sun may audit their test results.
- Sun, as well as licensees, must comply.

Test Challenges and Exclusions

- TCK User's Guide defines the appeals process.
- Licensees may assert that a test is invalid:
  - Because of a bug in the test (eg, logic error, or incorrect interpretation of spec), or in the spec (eg, requirements are contradictory).
  - Because the test is biased to a particular implementation.
- Invalid tests are added to the exclude list and will not be run.
- TCK Rules allow the Spec Lead to provide alternate tests.

Testing is not enough

- Compatibility tests can never cover everything.
  - Licensees are free to innovate and add value (subject to constraints).
  - Spec Leads need the flexibility to adapt to new implementations and changing circumstances.
- Therefore, *Compatibility Requirements* (The Rules) define additional conditions that must be met.
- Also read your license.

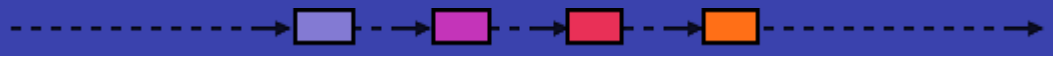
Compatibility Requirements (1)

- Chapter 2 of TCK User's Guide.
- Specify criteria for passing the test suite:
  - All required tests must pass.
  - Tests must not be modified.
- Specify additional (untested, or even untestable) criteria that must be met:
  - All specified classes and interfaces must be provided and must function as specified.
  - Syntax and semantics of the Java language must be preserved.
- Evolve over time to clarify intent, adapt to event.

Compatibility Requirements (2)

- Compatibility is a contractual obligation.
  - You must not ship incompatible products.
- Compatibility is binary.
  - You can't be “almost compatible” or “a little bit incompatible”.
- The JCK contains 120,000 test cases.
  - If you pass 119,999 you are not compatible.
  - If you pass 120,000 and don't meet all the other requirements you are not compatible.

A Brief Tour through the (Java SE) Rules



- Chapter 2 of the JCK User's Guide contains definitions and rules.
- This is only a summary - see the User's Guide for complete details.

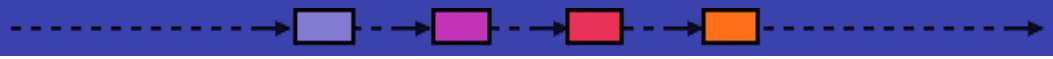
Compatibility in All Modes

- SE1: The Product must be able to satisfy all applicable testing requirements, including passing all Conformance Tests, in every Product Configuration and in every combination of Product Configurations, except only as specifically exempted by these Rules.
- *Product Configuration*: A specific setting or instantiation of an Operating Mode.
- *Operating Mode*: Any Documented option of a *Product* that can be changed by a user in order to modify the behavior of the *Product*.

Why Rule SE1?

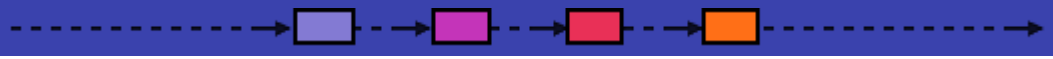
- Make it obvious what's compatible:
  - It's simple - all modes are compatible.
- Applications can always assume a conforming configuration.
  - Users don't have to worry about how they configured the platform or the application.
- No bait and switch:
  - "Yes, we're compatible, but this other mode makes your application run twice as fast."

Exceptions to SE1



- SE1.1: Modes that control Resources.
  - There must be one combination of modes that passes the tests.
- SE1.2: Modes that report only version, usage, or diagnostic information.
- SE1.3: Modes that select the Edition with which the product is compatible.
  - Must meet the compatibility requirements for that Edition.

Example Exceptions for SE1.1

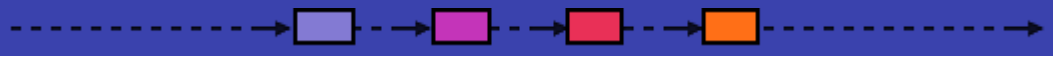


- Test suite may need 128MB of memory to run.
  - Other apps may run in less.
  - OK to require running JCK with 128MB.
- Security permissions may control access to network resources.
  - OK to require permissions needed by JCK.
- Client and server may need to be on same intranet.
  - OK to require for JCK.

Forbidden Operating Modes

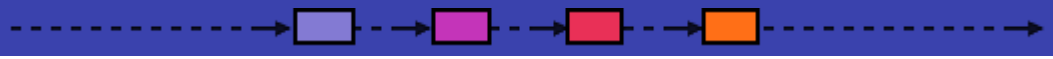
- *Debug mode* that disables all security checks
- *Production mode* that disables error checks
- *Fast mode* that disables functionality

Configurability of the Test Suite



- Only materials supplied with the test suite may be used for testing.
- (SE2) No source or binary test or test property may be modified except as documented in the JCK.
- (SE3) Only the testing tools supplied in the JCK may be used for testing.
- (SE4) The Exclude List can only be modified by the JCP Maintenance Lead.
- (SE8) Only test binaries supplied in the JCK may be used for testing.

Modifications to the Rules



- SE5: The Maintenance Lead can define exceptions to these Rules.
- Such exceptions must be made available to and apply to all licensees.
- Rules will evolve over time.
- No waivers or special considerations are ever given to anyone.

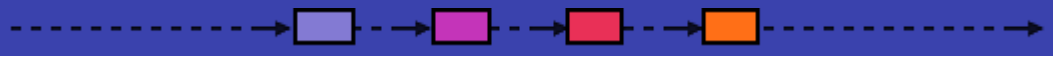
Define the Platform

- SE6: All hardware and software necessary to ensure the product behaves in a conformant manner must be documented and available to all users.
  - Example: If a patch to an operating system is required for the product to perform properly, the patch must be documented and available.

Implement the specified APIs

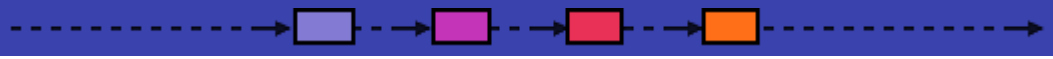
- SE7: The product must contain the full and exact set of public and protected APIs defined in the specification.
- SE7.1: If a product includes multiple Java technologies, it must contain the union of all public and protected APIs of these technologies.
- No subsetting, supersetting, or modification of the public or protected APIs is allowed.
  - Checked by Signature Test.

Exception to Rule SE7



- SE7.2: With approval of Maintenance Lead, a product may include a newer version of an Endorsed Standard than that defined by the specifications
  - *Endorsed Standard*: An API defined through a standards process outside of JCP and included in a Java technology.
    - CORBA, SAX, DOM...
- The Maintenance Lead may issue updated tests.
- Newer versions of *Endorsed Standards* included in a product must be documented.

Class-file Semantics

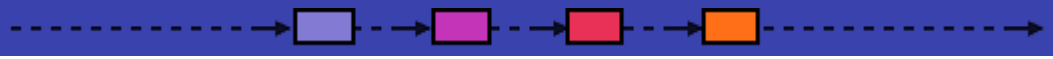


- SE9: The functional programmatic behavior of any binary class or interface must be that defined by the Specifications.
  - API behavior must be consistent with the spec even if no conformance tests check for it.
- Semantics of user-created programs must conform to the Java Language and JVM Specifications.

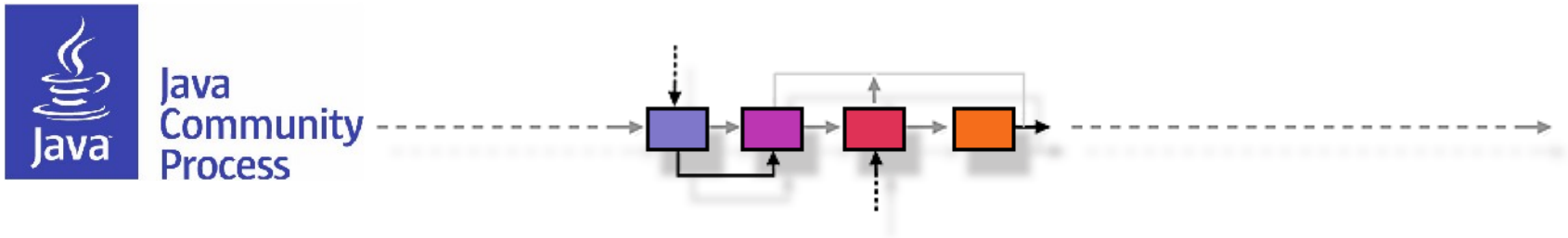
But wait! There's much more!

- If your Product includes a *Development Kit*...
  - There are Rules that specify how you must process Java source-code,
  - And Rules for Schema Compilers and Generators,
  - And Rules for Java-to-WSDL and WSDL-to-Java Tools...
- Read the Fine Print!

Summary



- Conformance is not just about testing.
- The Rules supplement the tests.
  - The spirit as well as the letter of Rules is important.
  - Rules evolve over time.
- Read the Fine Print!



Thank You!

http://jcp.org

Sun microsystems