



Java[®] and OpenJDK[®] compatibility testing

Andrew Haley

Tech Lead, Open Source Java

Why bother with the TCKs?

Why bother with the TCKs?

To fix bugs, silly!

Why bother with the TCKs?

To fix bugs, silly!

Java (TM)

Why bother with the TCKs?

To fix bugs, silly!

Java (TM)

It's very hard to know if you've correctly implemented the specification

Example 1: JSR 166

The JSR 166 implementation in gcj hung partway through the TCK run.

It turned out that there was something I'd missed from reading the JSR:

`Thread.interrupt()` implicitly awakens a thread after `park()`.

Example 1: JSR 166

The JSR 166 implementation in gcj hung partway through the TCK run.

It turned out that there was something I'd missed from reading the JSR:

`Thread.interrupt()` implicitly awakens a thread after `park()`.

It was only possible for me to run the JSR 166 TCK because that TCK was public.
Most TCKs are not public.

Without that TCK, gcj would probably still have the bug.

Example 2: javax.management.MbeanServer

Eclipse running on gcj didn't display its Help in the web browser.

That's all we knew.

After three days of debugging, it call came down to a mistake reading the javadoc specification for `MbeanServer.registerMBean()`.

Example 2: javax.management.MbeanServer

javax.management

Interface MBeanServer

All Superinterfaces:

[MBeanServerConnection](#)



Example 2: javax.management.MbeanServer

javax.management

Interface MBeanServer

All Superinterfaces:

[MBeanServerConnection](#)

registerMBean

```
ObjectInstance registerMBean(Object object,  
                             ObjectName name)  
    throws InstanceAlreadyExistsException,  
          MBeanRegistrationException,  
          NotCompliantMBeanException
```

Registers a pre-existing object as an MBean with the MBean server. If the object name given is null, the MBean must provide its own name by implementing the [MBeanRegistration](#) interface and returning the name from the [preRegister](#) method.

Parameters:

object - The MBean to be registered as an MBean.
name - The object name of the MBean. May be null.

Returns:

An [ObjectInstance](#), containing the [ObjectName](#) and the Java class name of the newly registered MBean. If the contained [ObjectName](#) is n, the contained Java class name is [getMBeanInfo\(n\).getClassName\(\)](#).

Throws:

Example 2: javax.management.MbeanServer

- It turned out that this was because we'd missed something vital.
- There is more to the specification of `MbeanServer.registerMBean()`, but it's not in the javadoc for that method, it's in the text at the top of the page:

When an MBean is registered or unregistered in the MBean server a [MBeanServerNotification](#) Notification is emitted. To register an object as listener to MBeanServerNotifications you should call the MBean server method [addNotificationListener](#) with `ObjectName` the `ObjectName` of the [MBeanServerDelegate](#). This `ObjectName` is:
`JMImplementation:type=MBeanServerDelegate`.

- No non-Classpath implementation of `MbeanServer.registerMBean()` has this bug, because everyone else has run the TCK.

Conclusion 1

- Java implementations with TCK access are much better than those without!
- However, that doesn't mean they meet the specification; it means that your interpretation of the specification agrees with that of the author of the TCK.
- This is still a better situation than C, with no reference implementation and no TCK.

Java is different

- Java is different from languages like C
 - C does not have an official test suite, but an official specification
 - With Java, the TCK is effectively *more* important than the specification: you don't have to meet the specification to be called Java(TM), but you do have to pass the TCK.

Java is different

- Java is different from languages like C
 - C's specification process is entirely open, but most implementations are closed
 - Java's specification process is (mostly) closed, but the Reference Implementation is (or soon will be) open

Java is different

- Java is different from languages like C
 - C's specification process is entirely open, but most implementations are closed
 - Java's specification process is (mostly) closed, but the Reference Implementation is (or soon will be) open
- The Right Thing to do would be to test JSRs by deploying them in real applications before they are final. However, because many JSRs are closed, you aren't allowed to do this.

Problem 1

- The Java SE TCK is confidential
 - We can't pass test cases to non-signatories to work on, can't put test cases into Bugzilla, etc, etc.
 - We can't auto-run any part of the TCK when respinning a package.

Problem 2

- The Java SE TCK is very hard to set up
 - It's taken us 2-3 months to get this far
 - 300+ page TCK manual to read
- Smaller distros will not have the ability to run the TCK on their own

How Red Hat can help the community

We have a standard GNU/Linux TCK configuration

- Follow these rules, install these files
- Much easier than figuring it out for yourself
- We will share this setup with everyone who has signed the JCK access agreement

However...

- There's still lots of work
- There is a number of interactive tests that require someone to sit at the console pressing buttons
- Probably never going to be suitable For Gentoo

So, how well are the tests doing?

So, how well are the tests doing?

We can't tell you! ;-)



Thanks

To Keith Seitz, who has done most of the work

Comments?