TCK Development Guidelines version 1.0

For Java™ Compatibility Test Suite Developers

THE NETWORK IS THE COMPUTER™

Sun Microsystems, Inc.
901 San Antonio Road
Palo Alto, CA 94043 USA
650 960-1300 fax 650 969-9131

January 2001 draft tck_guidelines-10_tr7.pdf

Contents

Preface

This document provides guidelines to facilitate development of Technology Compatibility Kits (TCKs) under the Java™ Community Process^{SM} (JCP) Program, to be used for testing implementations of a Java technology specification.

Who Should Use This Book

This book is meant to be used by groups that are interested in developing a TCK, including those internal and external to Sun.

How This Book Is Organized

This book is organized as follows:

Chapter 1, "Introduction to Java™ Compatibility Testing" provides an overview of a number of significant factors that relate to developing TCKs within the Java Community Process (JCP).

Chapter 2, "TCK Components" describes the components of a TCK as defined by the JCP.

Chapter 3, "TCK Appeals Process" provides guidelines and examples for formulating an appeals process.

"TCK Glossary version 1.0" defines important terms.

Related Books

We recommend that you become familiar with the formal procedures for using the Java specification development process within the JCP, which is described in detail on the Web at the following URL:

`http://java.sun.com/aboutJava/communityprocess/`

Note – All references to specific Web URLs are given for the sake of your convenience in locating the resources quickly. These references are subject to changes that are in many cases beyond the control of the authors.

Typographic Conventions Used in This Book

The following table describes the typographic conventions used in this book.

TABLE P-1 Typographic Conventions

Typeface or Symbol	Meaning	Example
`Courier` `AaBbCc123`	The names of commands, files, and directories, variable and method names; on-screen computer output	Edit your `.login` file. Use `ls -a` to list all files. `machine_name% You have mail.`
▼Headline	Inverted triangle alerts reader to an important section or procedure to follow	▼Building the "libjckjni.so" library

`Bold` **`AaBbCc123`**	What you type, contrasted with on-screen computer output	`machine_name%` **`su`**
		`Password:`

`Brackets` `<ABCDE>`	Command-line placeholder: replace with a real name or value	Example 1: To delete a file, type `rm <filename>` Example 2: `<TCK_DIRECTORY>`
Italics *AaBbCc123*	Book titles, new words or terms, or words to be emphasized or set apart from the other text	Read Chapter 6 in *User's Guide*. These are called *class* options. You *must* be root to do this.
\	A backslash at the end of a line indicates that a long line has been broken in two, typically to improve legibility in code	java classname \\ [options for classname]
Indented Line	An alternative to the backslash character as used above, particularly in Win32 platform code	java classname [options for classname]

CHAPTER **1**

Introduction to Java™ Compatibility Testing

This chapter gives an overview of a number of significant factors that relate to developing Java™ Technology Compatibility Kits (TCKs).

1.1 Defining Compatibility Testing

Compatibility testing refers to the methods used to test an implementation of a Java technology specification in order to assure consistency across multiple hardware platforms, operating systems, and other implementations of the same Java technology specification. When this assurance is accomplished by means of a formal process, application developers can then be confident that an application will run in a consistent manner across all tested implementations of the same Java technology specification. This consistent specification-based behavior is a primary function of compatibility testing.

A Java technology is defined as a Java specification and its reference implementation. Examples of Java technologies are Java 2 Standard Edition (J2SE), Java 2 Enterprise Edition (J2EE), and Mobile Information Device Profile (MIDP).

1.2 Compatibility Testing vs. Product Testing

Compatibility testing differs from traditional product testing in a number of ways.

- Unlike product testing, compatibility testing is not primarily concerned with robustness, performance, or ease of use.
- The primary purpose of Java compatibility testing is to determine whether an implementation of a technology is compliant with the specification of that technology.
- Compatibility test development for a given feature relies on a complete specification and reference implementation for that feature.
- Compatibility testing is a means of ensuring correctness, completeness, and consistency across all implementations of a technology specification that are developed.
- The primary goal of compatibility testing is to provide the assurance that an application will run in a consistent manner across all tested implementations of a technology.

1.3 Java Community Process (JCP) and Compatibility Testing

The Java Community Process^{SM} (JCP) Program is the formalization of the open process that Sun Microsystems, Inc. has been using since 1995 to develop and revise Java technology specifications in cooperation with the international Java community.

The JCP specifies that the following three major components must be included as deliverables in a final Java technology release under the direction of the responsible Expert Group:

1. Technology Specification
2. Reference Implementation
3. Technology Compatibility Kit (TCK)

For further information on the JCP see this URL:

`http://java.sun.com/aboutJava/communityprocess/`

1.4 Sun's Compatibility Testing Program

Some Sun Java branding programs require passing appropriate test suites before a Java brand can be used in conjunction with a product. For example, the Java Compatible™ and Java Powered™ brands require this.

This section describes Sun's compatibility testing program with the understanding that an Expert Group is not obligated to strictly follow these particular procedures.

The Sun compatibility testing program allows an implementor of a Java technology product to formally identify the product with the appropriate Java brand. The implementor self-tests an implementation and then notifies Sun when it has passed.

The following steps are performed during the self-test process:

1. Implementor executes the TCK against the implementation under test.
2. Sun assists with testing issues.
3. Implementor notifies Sun that all compatibility requirements have been met.

CHAPTER 2

TCK Components

This chapter describes the components of a TCK as defined by the JCP.

2.1 Components of a TCK

A TCK is a suite of tests, tools, and documentation that allows an implementor of a Java technology specification to determine if the implementation is compliant with the specification. Compatibility tests are usually developed by Sun for the purpose of using them within a comprehensive TCK product that is developed for a particular technology release.

A TCK includes the following components:

- Test Harness

  Defined as the applications and tools that are used for test execution and test suite management.

- TCK Test Suite

  The composite of the actual test cases in a TCK that are executed to test an implementation.

- Documentation

  Includes the specific TCK usage procedures and the compatibility Testing Requirements that apply to the related technology release (usually in the form of a TCK user's guide); also includes a description of the TCK Appeals Process.

- Audit Process (optional)

  Used to better ensure the integrity of a consistent self-testing compatibility program.

These components will now be described in more detail.

2.2 Test Harness

A test harness includes the applications and tools that are used for test execution and test suite management. It will usually support the following features:

- Sequencing of tests, allowing them to be loaded and executed automatically.
- Automated reporting capability to minimize manual errors.
- Ideally, will run tests on any potential target device, however, if permissible by the compatibility rules, may run tests on a computational equivalent when testing requires an emulated environment or a breadboard.

Sun generally uses the JavaTest™ harness in its TCKs.

2.2.1 Sun's Test Suite Harness: JavaTest™

Most TCKs developed at Sun use JavaTest™ for test execution and test suite management. JavaTest can be described as both a Java application and a set of compatibility testing tools. An efficient means of relating the general requirements of a test harness is to describe the functionality of JavaTest.

The JavaTest application is designed to execute tests on varying Java technology implementations from powerful servers to small consumer and embedded systems. JavaTest provides the following functionality:

- User interface for ease of use
- Framework for running tests
- Display of test results
- Failure analysis
- Reporting
- Test results auditing
- Auditable test specification framework
- Distributed testing environment support

2.3 TCK Test Suite

The TCK test suite is a set of tests designed to verify that an implementation of a Java technology complies with the appropriate specification. The individual tests correspond to assertions of the specification. The reference implementation must pass all tests and meet the testing requirements.

Each test in a TCK test suite is composed of one or more test cases that are designated by a test description.

A test case is either source code or written procedures designed to exercise one aspect, piece, or point of a specified assertion. Assertions are statements of required behavior, either positive or negative, that are made within the technology specification.

Test descriptions are composed of machine readable information that describes a test to the test execution harness so that it can correctly process and run a related test.

Most TCKs developed at Sun test run-time functionality of Java technologies, however, there is a TCK which tests compiler implementations.

Ideally, TCK tests are developed to exhibit the following characteristics:

- Hardware and software platform independence (all tests use Java code with use of native and/or platform specific code when no other means exists)
- Implementation independence with no reliance on any implementation specific API or behavior
- Designed so that they:
  - Clearly determine pass/fail status
  - Guarantee that all implementors have the same requirements
  - Guarantee that all implementors have equal access to updates

2.4 Documentation

TCK documentation should contain the following items:

- Description of how to run the test suite
- Test harness documentation
- Testing Requirements which:
  - Identify requirements for successfully passing the test suite
  - Identify additional requirements that cannot be determined by the test suite
- Appeals process for handling test challenges by implementors

2.4.1 TCK User's Guide and the Testing Requirements

TCKs developed by Sun contain a user's guide which describes how to install, configure, and run the applicable TCK. An additional requirement is that each TCK user's guide must contain the Testing Requirements that apply to the related technology to be tested. These Testing Requirements specify how the TCK must be used for compatibility testing.

The Testing Requirements are tailored for each specification. Suggested examples are:

- Other than test configuration files, no test source or binary may be modified in any way.
- Every required test must be able to pass on every publicly documented configuration of a specification implementation.
- The functional programmatic behavior of every binary class and interface must correspond to its specification.
- No subsetting, supersetting, or modification of the public or protected API is allowed except as defined in the specification.
- Any implementation of a specification that is based on a subset of another specification must pass the same corresponding tests and follow appropriately equivalent compatibility rules as the original.

The TCK user's guide also includes information on the required specifications and APIs. The applicable specification intended to be tested by a TCK may be based upon and may reference several Java specifications including those for any of the following: the Java API, the Java programming language, the Java Virtual Machine (JVM).

As such, the tests are usually organized into three primary categories, with possible variations depending on the test suite architecture:
- API tests
- Language tests
- VM tests

2.4.2 TCK Appeals Process Documentation

As part of the documentation released with a TCK, the Expert Group must identify an Appeals Process through which challenges to the TCK can be addressed. This Appeals Process is an escalation process managed by the Maintenance Lead, in which challenges to conformance tests are evaluated and either accepted or rejected. The Maintenance Lead is the member of the Expert Group responsible for maintaining the technology specification.

For a detailed description see Chapter 3, "TCK Appeals Process."

2.5 Optional Audit Process

A formal independent audit process can better ensure the integrity of a consistent self-testing compatibility program.

An audit can be accomplished through:

- Physical testing, which is quite costly, or
- Test results verification, which is usually preferred

If a test results verification program is implemented, then an independent third party might be used to verify that all the required tests were run and have passed.

JavaTest also supports test results verification, and it can be used in a verification program.

CHAPTER **3**

TCK Appeals Process

The Java Community Process includes a test appeals process. The appeals process defines the escalation process in which challenges to compatibility tests are evaluated and either accepted or rejected.

As part of the documentation released with a TCK within the JCP, the Expert Group must identify an appeals process through which challenges to the TCK can be addressed. This appeals process is managed by the Maintenance Lead (the designated expert responsible for maintaining the related technology specification).

3.1 Appeals Process Guidelines

The appeals process is designed by the Expert Group in order to best handle challenges delivered to the Maintenance Lead in a timely fashion. This might be accomplished by giving the Maintenance Lead complete authority to interpret the specification and rule on each test challenge. As another option, the Expert Group can identify a representative team of several technical experts called a Technical Interpretations Committee or TIC, whose members have an understanding of both the specification and the TCK. In this case, the TIC is then responsible for interpreting the specification and making all decisions on test challenges under the direction of the Maintenance Lead.

Any appeals process should outline the following points:

- Who can make challenges to the TCK
- What challenges to the TCK may be submitted
- How these challenges are submitted
- How and by whom challenges are addressed
- How any accepted changes to the TCK are managed

Any appeals process must also adhere to these assumptions:

- All reviews for TCK challenges go through the Maintenance Lead or its designee.
- TCKs are not restricted in the choice of tools, test suite format, or management of the changes due to appeals.
- All implementors are treated equally in terms of access to changes, pass/fail reporting, and compatibility rules and requirements.

3.2 Appeals Process Examples

These examples illustrate the different ways that an Expert Group might formulate an appeals process by using scenarios with varying degrees of complexity.

3.2.1 Example 1: API Test Appeals Process

In this example, the TCK is to be made available to the public under a free license. The Expert Group has chosen the Maintenance Lead as the sole interpreter of tests, so in this case there is no TIC.

- Who can make challenges to the TCK?

  Any party using the TCK as part of an active effort to implement the specification.
- What challenges to the TCK may be submitted?

  Only challenges based on the integrity or relevance of individual tests may be submitted.
- How are challenges to the TCK submitted?

  Any challenge to the TCK must be submitted in writing to the Maintenance Lead with e-mail being acceptable. (See Section 3.3, "Test Challenge and Response Forms," on page 3-25 for examples.)
- How and by whom are challenges addressed?

  The Maintenance Lead will review and respond to all appeals within 10 working days.
- How are accepted changes to the TCK managed?

  If a test is determined to be invalid, the Maintenance Lead will remove the test from the TCK and publish an updated version on the specification web site.

3.2.2 Example 2: J2ME Profile Test Appeals Process

To create a Profile API in J2ME, the Expert Group will (in part) base their specification on building block APIs. In the interest of ensuring that every class and method of any API shared among Java technologies is tested in exactly the same manner, the same TCK tests used to test common parts of the building block APIs should be used wherever feasible for testing the Profile API. For this reason, the test interpretation for challenges to these common tests must be made in conjunction with the building block owner(s). New TCK tests for a Profile that corresponds to extensions to these building block API s are handled normally.

- Who can make challenges to the TCK?

  Any licensee of the Profile technology.

- What challenges to the TCK may be submitted?

  Challenges based on the integrity or relevance of individual tests of both building block APIs or Profile specific APIs.

- How are challenges to the TCK submitted?

  Submitted in writing to the Maintenance Lead. Each challenge must identify the test or set of related tests and a detailed justification of why the test(s) are believed to be invalid.

- How and by whom are challenges addressed?

  The Maintenance Lead is responsible for reviewing and ruling on all test challenges. This will be done with the assistance of the test developers and specification developers (if appropriate and if available). If challenges are made to building block API tests, the Maintenance Lead will work with the originators of the tests to make the decision. The decision will be sent to the test challenger within 10 business days after receipt of the test challenge. If a challenge is made to a building block API test, then the decision will be sent to the test challenger by the Maintenance Lead as soon as it is received from the building block API owner (typically the test challenge will use the appeals process defined for the building block API).

- How are accepted changes to the TCK managed?

  The Maintenance Lead will update the TCK and make changes available to all TCK licensees within 3 working days. Licensees will be notified by e-mail and the updated TCK will be placed on the licensee web site.

3.2.3 Example 3: Appeals Process with a TIC

This example shows one way an appeals process can be defined to handle situations where it is prudent to include more than just the Maintenance Lead for test interpretation. A panel of experts can be chosen from among groups of the following types to form a Technical Interpretations Committee (TIC): the Expert Group, other Java Community Member organizations, TCK test developers, academia. The TIC is run by the Maintenance Lead and is chartered to review all challenges to the TCK by any Java Community Member. The TIC must be comprised of individuals who are technology experts and familiar with the specification, the RI, and the TCK. Ideally, a TIC member should have a thorough knowledge of the technical intricacies and trade-offs behind the details of the specification.

The following statements illustrate an example process that can be used to manage test challenges with a TIC:

- Who can make challenges to the TCK?

  Any licensee of the technology.

- What challenges to the TCK may be submitted?

  Individual or related tests may be challenged for reasons such as:

  - Test is faulty (for example, program logic errors)
  - Specification item covered by the test is ambiguous
  - Test does not match the specification
  - Test assumes unreasonable hardware and/or software requirements

- How are challenges to the TCK submitted?

  Challenges are written and sent to the Maintenance Lead or its designee contesting the validity of one or a related set of TCK tests. A detailed justification for why each test should be invalidated must be included with the challenge.

- How and by whom are challenges addressed?

  The process is managed by the Maintenance Lead or its designee as follows:

  The Maintenance Lead evaluates the challenge and prepares a response. If the appeal is incomplete or unclear, it is returned to the Java Community Member for correction. The Maintenance Lead should check with the test developers (if possible) to review the purpose and validity of the test before writing the response. The Maintenance Lead must also attempt to complete the response within 5 business days. If this is not possible, the Maintenance Lead must tell the Java Community Member when it will be completed (maximum 10 business days from receipt of challenge). If the challenge is identical to a previously rejected challenge (that is, same test and justification), the Maintenance Lead is not required to escalate it to the TIC, but can send the previous TIC response to the Java Community Member. Challenge and response are sent by the Maintenance Lead to the TIC for evaluation. Note that a challenge may affect more than one specification. For example, challenging a Profile TCK test which has been taken from a building block TCK will usually require initiation of that building block API’s appeals process. In addition, if a challenge is clearly correct and a test must be invalidated, to save time the Maintenance Lead may choose to alert the TIC of this fact, and wait 3 working days for any disagreement before invalidating the test. The TIC must make a decision of test validity or invalidity within 10 working days of receipt of the challenge and its response by the Maintenance Lead. If a challenged test has been adopted from another TCK, the test must also be reviewed by its respective TIC or Maintenance Lead. In this case, another 5 working days may be added to the response period for each additional TCK appeals process involved in the decision. All decisions must be documented with an explanation of why test validity was maintained or rejected.

- How are accepted changes to the TCK managed?

  The Java Community Member is informed of the decision and proceeds accordingly. If the test challenge is approved and one or more tests are invalidated, the Maintenance Lead removes the test(s) from the Test Suite or invalidates the test(s) so as to exclude them as part of the testing requirements for all implementors. If the test appeal is rejected, the Java Community Member may choose to escalate the decision to the EC. In addition, an alternate test may be developed by the TIC for use by the test challenger. Such a test would be deemed equivalent to the original test for meeting the testing requirements.

3.2.4 Example 4: JCK Test Appeals Process

The Java Compatibility Kit (JCK) is the product name of the TCK for Java 2 Standard Edition (J2SE). Sun has a well established process for managing challenges to its JCK test suite and plans to continue using a similar process in the future. As the Maintenance Lead, Sun will authorize representatives from Sun's Licensee Engineering group to be the point of contact for all test challenges. Typically this will be the engineer assigned to a company as part of its JCK support group. The following process will be used to manage challenges to JCK tests:

- Who can make challenges to the JCK?

  Only licensees of the JCK

- What challenges to the JCK may be submitted?

  Individual or related tests may be challenged for reasons such as:

  - Test is faulty (for example, has program logic errors)
  - Specification item covered by the test is ambiguous
  - Test does not match the specification
  - Test assumes unreasonable hardware and/or software requirements

- How are challenges submitted?

  Challenges are submitted to the Sun Licensee Engineering contact assigned by Sun to the licensee. Appeals must be in writing, such as described in Section 3.3, "Test Challenge and Response Forms," on page 3-25.

- How and by whom are challenges addressed?

  Sun Licensee Support Engineering coordinates the review and decisions made by test development and specification development engineers. See Section 3.2.5, "JCK Appeals Process Steps," on page 3-24.

- How are approved changes to the JCK managed?

  All tests found to be invalid are placed on the Exclude List for that version of the JCK within 1 business day. The new Exclude List is published to all JCK licensees on the Java Technology Licensee web site. As the Maintenance Lead, Sun has the option of creating an alternative test to address any challenge. Alternative tests (and criteria for their use) will be placed on the Java Technology Licensee web site. Note that passing an alternative test is deemed equivalent with passing the original test.

3.2.5 JCK Appeals Process Steps

1. JCK licensee writes a test challenge to his or her Licensee Support engineer contesting the validity of one or a related set of JCK tests. A detailed justification for why each test should be invalidated must be included with the challenge as described in Section 3.3, "Test Challenge and Response Forms," on page 3-25.

2. Sun Licensee Support Engineering evaluates the challenge. If the appeal is incomplete or unclear, it is returned to the submitting licensee for correction. If all is in order, the Licensee Support engineer will check with the test developers to review the purpose and validity of the test before writing a response. Licensee Support Engineering will attempt to complete the response within 5 business days. If the challenge is similar to a previously rejected test challenge (that is to say., same test and justification), Licensee Support Engineering will send the previous response to the licensee.

3. The challenge and any supporting materials from test developers is sent to the specification engineers for evaluation. A decision of test validity or invalidity is normally made within 15 working days of receipt of the challenge. All decisions will be documented with an explanation of why test validity was maintained or rejected.

4. The licensee is informed of the decision and proceeds accordingly. If the test challenge is approved and one or more tests are invalidated, Sun places the tests on the Exclude List for that version of the JCK (effectively removing the test or tests from the Test Suite). All tests placed on the Exclude List will have a bug

report written to document the decision and it will be made available to all licensees through the bug reporting database. If the test is valid but difficult to pass due to hardware or operating system limitations, Sun may choose to provide an alternate test to use in place of the original test (all alternate tests are made available to the licensee community as a whole).

5. If the test challenge is rejected, the licensee may choose to escalate the decision to the Executive Committee (EC), however, it is expected that the licensee would continue to work with Sun to resolve the issue and only involve the EC as a last resort.

3.3 Test Challenge and Response Forms

Using a designated form can facilitate the process of submitting test challenges and their related responses. The following examples include items that these forms might contain.

3.3.1 Example Test Challenge Form Items

- Test Name
- Spec Involved (could be more than one)
- Test Suite Used (technology name and version #)
- Complaint (complainant name and argument for why test is invalid)

3.3.2 Example Test Challenge Response Form Items

- Test Name
- Spec Involved (could be more than one)
- Test Suite Used (technology name and version #)
- Defense (test defender name and role, for example, test developer or Maintenance Lead, and so forth; also includes the argument for why the test is valid—this can be iterative)
- Implications of test invalidity, for example:
  - Other tests are affected.
  - Test framework is affected thereby creating or exposing ambiguities in the specification, for instance, due to unspecified requirements.
  - Test invalidates reference release, or, creates a hole in the test suite for a portion of testable functionality
- Alternatives (for example, is an alternate test appropriate.)

TCK Glossary version 1.0

Note that there is also a glossary at this URL which contains additional JCP terms:
`http://java.sun.com/aboutJava/communityprocess/glossary.html`

Appeals Process	An escalation process in which challenges to conformance tests are evaluated and either accepted or rejected.
assertion	A statement contained in a Java technology API, VM or language specification to specify some necessary aspect that must then be tested.
assertion testing	A compatibility testing technique based on testing assertions in a specification.
behavior-based testing	A set of test development methodologies that are based on the description, behavior, or requirements of the system under test, not the structure of that system. This is commonly known as black-box testing.
boundary value analysis	A test case development technique which entails developing additional test cases based on the boundaries defined by previously categorized equivalence classes.
Executive Committee (EC)	Composed of Members who guide the evolution of the Java technologies, representing a cross-section of both major stakeholders and other Members of the Java Community.
equivalence class partitioning	A test case development technique which entails breaking a large number of test cases into smaller subsets with each subset representing an equivalent category of test cases.
exclude list	Exclude list files (*.jtx), are used by Sun to supply a list of invalid tests to be filtered out of a test run when using JavaTest. The exclude list provides a level playing field for all implementors by ensuring that when a test is determined to be invalid, then no implementation is required to pass it.

Expert	A Member representative (or an individual who has signed the IEPA) who has expert knowledge and is an active practitioner in the technology covered by the JSR.
Expert Group	The group of Experts who develop or make significant revisions to a Specification.
Individual Expert Participation Agreement (IEPA)	An agreement between Sun Microsystems and an individual that allows that individual to serve on an Expert Group at the invitation of the Specification Lead. There is no fee associated with the IEPA and it is valid until the Expert Group disbands. The IEPA allows individual technical experts who do not represent companies or organizations to participate on Expert Groups.
Java Community Process^{SM} (JCP)	The Java Community Process^{SM} (JCP) program is the formalization of the open process that Sun Microsystems, Inc. has been using since 1995 to develop and revise Java™ technology specifications in cooperation with the international Java community. The JCP is meant to foster the evolution of Java technology in Internet time, and in an open and participative manner.
Java Platform Libraries	The class libraries that are defined for each particular version of a Java Technology in its Specification.
Java Specification (Specification)	A written specification for some aspect of the Java technology. This includes the language, virtual machine, Platform Editions, Profiles, and application programming interfaces.
Java Specification Request (JSR)	The document submitted to the PMO by one or more Members to propose the development of a new Specification or significant revision to an existing Specification.
Java Technology	The combination of a Java specification and its reference implementation. Examples of Java technologies are Java 2 Standard Edition (J2SE), Java 2 Enterprise Edition (J2EE), and Mobile Information Device Profile (MIDP).
JavaTest™	The test harness developed by Sun to manage test execution and result reporting in TCKs.
JavaTest Agent	A small Java™ application that is used in conjunction with JavaTest to run tests on a Java implementation on which it is not possible or desirable to run the main JavaTest application.
JCP	See Java Community Process^{SM} (JCP).
JCP Web Site	The web site where anyone with an Internet connection can stay informed about JCP activities, download draft and final Specifications, and follow the progress of Specifications through the JCP.

JCP Specification Page (Spec Page)	Each Specification approved for development or revision will have a dedicated public web page established on the JCP Web Site to contain a history of the passage of the Specification through the JCP, including a record of the decisions, actions, and votes taken by the EC with respect to the draft Specification.
Maintenance Lead (ML)	The Expert responsible for maintaining the Specification.
Platform Edition Specification (Platform Edition)	A Specification that defines a baseline API set that provides a foundation upon which applications, other APIs, and Profiles can be built. There are currently three Platform Edition Specifications: J2SE, J2EE, and J2ME.
Process Management Office (PMO)	The group within Sun Microsystems that is responsible for administering the JCP and chairing the EC.
Profile Specification (Profile)	A Specification that references one of the Platform Edition Specifications and zero or more other JCP Specifications (that are not already a part of a Platform Edition Specification). APIs from the referenced Platform Edition must be included according to the referencing rules set out in that Platform Edition Specification. Other referenced specifications must be referenced in their entirety.
Reference Implementation (RI)	The prototype or *proof of concept* implementation of a Specification.
structure-based testing	A set of test development methodologies that are based on the internal structure or logic of the system under test, not the description, behavior, or requirements of that system. This is commonly known as white-box or glass-box testing. Compatibility testing does not make use of structure-based test techniques.
TCK	See Technology Compatibility Kit (TCK).
Technology Compatibility Kit (TCK)	The suite of tests, tools, and documentation that allows an implementor of a Java Technology Specification to determine if the implementation is compliant with the Specification.
test	One or more test cases designated by a test description.
test case	The source code and/or written, manual procedures designed to exercise one aspect of a specified assertion.

test description	Machine readable information that describes a test to the test execution harness so that it can correctly process and run the related test. The actual form and type of test description will depend on the attributes of the test suite. When using JavaTest, the test description is a set of test suite specific name/values pairs in either HTML tables or Javadoc-style tags.
test finder	When using JavaTest, a nominated class, or set of classes, that read, verify, and process the files that contain test descriptions in a test suite. All test descriptions that are located or found are handed off to JavaTest for further processing.
test script	A script used by JavaTest, responsible for running the tests and returning the status (pass, fail, error) to JavaTest. The test script must understand how to interpret the test description information returned to it by the test finder.
test suite	The set of conformance tests included in a TCK.
Technical Interpretations Committee (TIC)	A group of technical experts lead by the Maintenance Lead or its designee to investigate and interpret challenges to TCK tests.