JUnit 5 Release Notes

Table of Contents

5.6.3 . . . 1
- JUnit Platform . . . 2
- JUnit Jupiter . . . 2
- JUnit Vintage . . . 2

5.6.2 . . . 2
- JUnit Platform . . . 2
- JUnit Jupiter . . . 2
- JUnit Vintage . . . 3

5.6.1 . . . 3
- JUnit Platform . . . 3
- JUnit Jupiter . . . 3
- JUnit Vintage . . . 3

5.6.0 . . . 3
- Overall Improvements . . . 4
- JUnit Platform . . . 4
- JUnit Jupiter . . . 5
- JUnit Vintage . . . 6

5.5.2 . . . 7
- Overall Improvements . . . 7
- JUnit Platform . . . 7
- JUnit Jupiter . . . 7
- JUnit Vintage . . . 7

5.5.1 . . . 8
- JUnit Platform . . . 8
- JUnit Jupiter . . . 8
- JUnit Vintage . . . 8

5.5.0 . . . 8

This document contains the *change log* for all JUnit 5 releases since 5.5 GA.

Please refer to the User Guide for comprehensive reference documentation for programmers writing tests, extension authors, and engine authors as well as build tool and IDE vendors.

5.6.3

Date of Release: October 26, 2020

Scope: Bug fixes since 5.6.2

For a complete list of all *closed* issues and pull requests for this release, consult the 5.6.3 milestone page in the JUnit repository on GitHub.

JUnit Platform

No changes.

JUnit Jupiter

No changes.

JUnit Vintage

Bug Fixes

- The internal `JUnit4VersionCheck` class — which verifies that a supported version of JUnit 4 is on the classpath — now implements a lenient version ID parsing algorithm in order to support custom version ID formats such as `4.13.1`, `4.13-patch_1`, etc.

5.6.2

Date of Release: April 10, 2020

Scope: Bug fixes since 5.6.1

For a complete list of all *closed* issues and pull requests for this release, consult the 5.6.2 milestone page in the JUnit repository on GitHub.

JUnit Platform

Bug Fixes

- `ReflectionSupport.findNestedClasses()` no longer detects inner class cycles for classes that do not match the supplied `Predicate`. For example, JUnit Jupiter no longer throws an exception if an inner class cycle is detected in a nested class hierarchy whose inner classes are not annotated with `@Nested`.

JUnit Jupiter

Bug Fixes

- Test discovery no longer halts with an exception for inner class hierarchies that form a cycle if such inner classes are not annotated with `@Nested`.

JUnit Vintage

Bug Fixes

- Generating display names from JUnit 4 `Descriptions` now falls back to `getDisplayName` if `getMethodName` returns a blank `String` instead of throwing an exception.

5.6.1

Date of Release: March 22, 2020

Scope: Bug fixes since 5.6.0

For a complete list of all *closed* issues and pull requests for this release, consult the 5.6.1 milestone page in the JUnit repository on GitHub.

JUnit Platform

Bug Fixes

- In order to avoid file locking issues on Microsoft Windows, URLs are no longer cached when loading `junit-platform.properties` files.
- The presence of multiple `junit-platform.properties` files on the classpath now only results in a warning if the files have different URLs.

JUnit Jupiter

Bug Fixes

- `TestInstancePreDestroyCallback` extensions are now invoked in reverse registration order when `PER_CLASS` test instance lifecycle semantics have been configured.

JUnit Vintage

No changes.

5.6.0

Date of Release: January 20, 2020

Scope:

- New `@EnabledForJreRange` and `@DisabledForJreRange` execution conditions
- `@Order` allows to specify relative order
- Parameter names are included in default display names of parameterized test invocations

- Improvements to `@CsvSource` and `@CsvFileSource`
- New `TestInstancePreDestroyCallback` extension API
- Performance improvements and bug fixes for the Vintage engine
- Improved error reporting for failures during test discovery and execution
- Support for using `any()` and `none()` in tag expressions
- `org.junit.platform.console` now provides a `java.util.spi.ToolProvider`
- `DiscoverySelectors` for tests in inherited nested classes
- OSGi metadata
- Minor bug fixes and improvements

For a complete list of all *closed* issues and pull requests for this release, consult the 5.6 M1, 5.6 RC1, and 5.6 GA milestone pages in the JUnit repository on GitHub.

Overall Improvements

- Gradle Module Metadata is now published for all artifacts.
- OSGi metadata is now published in all binary JARs.
- Javadoc now contains a module API overview page.

JUnit Platform

Bug Fixes

- The `EventConditions.nestedContainer()` method in the Engine Test Kit now correctly handles events from multiple levels of nested classes.
- Module `org.junit.platform.launcher` now reads `java.logging` due to usage of types in package `java.util.logging`.

Deprecations and Breaking Changes

- The `Launcher` now propagates errors during test discovery by default instead of only logging and thereby potentially hiding them. In order to restore the old, lenient behavior, you can set the `junit.platform.discovery.listener.default` configuration parameter to `logging`.
- To support the above feature consistently, a new `EngineDiscoveryListener` interface was introduced. `TestEngine` implementations should now notify the listener that can be accessed via the `EngineDiscoveryRequest.getDiscoveryListener()` method about each processed `DiscoverySelector`. Test engines that use `EngineDiscoveryRequestResolver` do not have to make any changes.
- In the `EngineTestKit` API, the `all()`, `containers()`, and `tests()` methods in `EngineExecutionResults` have been deprecated in favor of the new `allEvents()`, `containerEvents()`, and `testEvents()` methods, respectively. The deprecated methods will be removed in JUnit Platform 1.7.0.

New Features and Improvements

- Running all tests with any tags or without any tags at all is now supported by using the special expressions `any()` and `none()`.
- `ReflectionSupport.findNestedClasses(…)` now detects cycles within inner class hierarchies. Consult the Javadoc for details.
- New methods in `DiscoverySelectors` to select and execute individual tests in inherited nested classes, via specific selectors (`NestedClassSelector` and `NestedMethodSelector`).
- New `printFailuresTo(PrintWriter, int)` method in `TestExecutionSummary` that allows one to specify the maximum number of lines to print for exception stack traces.
- `TestExecutionSummary.Failure` is now serializable.
- `ThrowableCollector.toTestExecutionResult()` is now public.
- Exceptions thrown by test engines during discovery and execution are now reported to `TestExecutionListeners`.
- The `junit-platform-commons` module no longer has a dependency on the `java.compiler` module (in terms of the Java Module System). Specifically, a new internal utility has been introduced in `PackageUtils` that implements functionality equivalent to `javax.lang.model.SourceVersion.isName(CharSequence)` from the `java.compiler` module.
- Module `org.junit.platform.console` now provides a `java.util.spi.ToolProvider` implementation that can be acquired by `ToolProvider.findFirst("junit")` when running on Java 9 or above.

JUnit Jupiter

Bug Fixes

- Method `assertIterableEquals()` in `Assertions` no longer throws a `StackOverflowError` when comparing iterables with components that themselves implement `Iterable`.
- When `@Nested` is used, the temporary directory is now also injected into instance fields of enclosing classes annotated with `@TempDir`.

Deprecations and Breaking Changes

- `@EnabledIf` and `@DisabledIf` have been removed from Jupiter's API. Script-based condition APIs and their supporting implementations were deprecated in JUnit Jupiter 5.5 with the intent to remove them in JUnit Jupiter 5.6. Users must now rely on a combination of other built-in conditions or create and use a custom implementation of `ExecutionCondition` to evaluate the same conditions.
- The default `@Order` value for non-annotated `@RegisterExtension` fields and test methods is now `Integer.MAX_VALUE / 2` instead of `Integer.MAX_VALUE`. If you had previously assigned extension fields or test methods an explicit order greater than `Integer.MAX_VALUE / 2`, this may be a breaking change for you.

New Features and Improvements

- Support for multi-character delimiters in `@CsvSource` and `@CsvFileSource`.
- Support for custom `null` values in `@CsvSource` and `@CsvFileSource`.
- Documented support for comments in CSV files loaded via `@CsvFileSource`.
- Auto-detection of enum type from method signature for `@EnumSource`.
- Parameter names are now included in the default display name of a `@ParameterizedTest` invocation (if they are present in the bytecode). The `{argumentsWithNames}` pattern can also be used in custom names.
- New `@EnabledForJreRange` and `@DisabledForJreRange` annotations for enabling or disabling test execution over a range of JRE versions.
- `@EnabledIfEnvironmentVariable`, `@DisabledIfEnvironmentVariable`, `@EnabledIfSystemProperty`, and `@DisabledIfSystemProperty` may now be used as *repeatable* annotations. In other words, it is now possible to declare each of those annotations multiple times on a test interface, test class, or test method.
- `JAVA_15` has been added to the `JRE` enum for use with JRE-based execution conditions.
- The `@TempDir` extension now makes an attempt to delete non-writable files by making them writable first.
- The default `@Order` value for non-annotated `@RegisterExtension` fields and test methods is now `Integer.MAX_VALUE / 2` instead of `Integer.MAX_VALUE`. This allows `@Order` annotated fields and methods to be explicitly ordered after non-annotated fields and methods. For example, this allows *before* callback extensions to be registered last and *after* callback extensions to be registered first, relative to other programmatically registered extensions.
- New `junit.jupiter.execution.timeout.mode` configuration parameter to control whether timeouts are applied to tests. Supported values include `enabled`, `disabled`, and `disabled_on_debug`.
- New `TestInstancePreDestroyCallback` interface that defines the API for extensions that wish to process test instances **after** they have been used in tests and **before** they are destroyed.
- New `TypeBasedParameterResolver<T>` abstract base class that serves as a generic adapter for the `ParameterResolver` API and simplifies the implementation of a custom resolver that supports parameters of a specific type.
- `InvocationInterceptor` extensions may now explicitly `skip()` an intercepted invocation. This allows executing the invocation by other means — for example, in a forked JVM.
- Discovery of `@Nested` test classes that form a cycle now results in an exception that halts execution of the JUnit Jupiter test engine instead of infinite recursion.

JUnit Vintage

Bug Fixes

- JUnit 3 suites with duplicate test names are now reported correctly.

New Features and Improvements

- To support adoption of the recent JUnit 4.13 release, the Vintage engine now requires the new version in its POM and Gradle Module Metadata. However, if you absolutely have to stay on 4.12, you can safely downgrade the dependency manually because the Vintage engine will remain compatible with 4.12.
- Performance improvements for projects with a large number of tests.
- Performance improvements for test classes with a large number of methods.

5.5.2

Date of Release: September 8, 2019

Scope: Bug fixes since 5.5.1

For a complete list of all *closed* issues and pull requests for this release, consult the 5.5.2 milestone page in the JUnit repository on GitHub.

Overall Improvements

- Published artifacts have been fixed regarding module descriptors.
  - Binary JAR files contain `module-info.class`.
  - Source JAR files contain `module-info.java`.
  - Javadoc JAR files contain neither `module-info.class` nor `module-info.java`.

JUnit Platform

No changes.

JUnit Jupiter

Bug Fixes

- The `JupiterTestEngine` no longer crashes without executing any tests if JUnit 4 is on the classpath but Hamcrest is not. Specifically, initialization of the `OpenTest4JAndJUnit4AwareThrowableCollector` class no longer fails if the `org.junit.internal.AssumptionViolatedException` class cannot be loaded from the classpath due to a missing Hamcrest dependency.

JUnit Vintage

No changes.

5.5.1

Date of Release: July 20, 2019

Scope: Bug fixes since 5.5.0

For a complete list of all *closed* issues and pull requests for this release, consult the 5.5.1 milestone page in the JUnit repository on GitHub.

JUnit Platform

No changes.

JUnit Jupiter

Bug Fixes

- Fix test discovery and execution of inherited `@Nested` classes.

JUnit Vintage

No changes.

5.5.0

Date of Release: June 30, 2019

Scope:

- Declarative `@Timeout` support
- New `InvocationInterceptor` extension API
- New `LifecycleMethodExecutionExceptionHandler` extension API
- Deprecation of script-based conditions (`@EnabledIf` and `@DisabledIf`)
- Configurable test discovery implementation for `TestEngine` authors
- Explicit Java module descriptors
- Various minor improvements and bug fixes

For complete details consult the 5.5.0 Release Notes online.