

# JUnit 5 Release Notes

## Table of Contents

5.6.0-M1 .....	1
Overall Improvements .....	2
JUnit Platform .....	2
JUnit Jupiter .....	2
JUnit Vintage .....	3
5.5.2 .....	4
Overall Improvements .....	4
JUnit Platform .....	4
JUnit Jupiter .....	4
JUnit Vintage .....	4
5.5.1 .....	5
JUnit Platform .....	5
JUnit Jupiter .....	5
JUnit Vintage .....	5
5.5.0 .....	5

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.

## 5.6.0-M1

**Date of Release:** October 21, 2019

### Scope:

- New `@EnabledForJreRange` and `@DisabledForJreRange` execution conditions
- `@Order` allows to specify relative order
- Improvements to `@CsvSource` and `@CsvFileSource`
- Improved error reporting for failures during test discovery and execution
- Performance improvements and bug fixes for the Vintage engine
- `org.junit.platform.console` now provides a `java.util.spi.ToolProvider`
- `DiscoverySelectors` for tests in inherited nested classes

For a complete list of all *closed* issues and pull requests for this release, consult the [5.6 M1](#) milestone page in the JUnit repository on GitHub.

# Overall Improvements

- `Gradle Module Metadata` is now published for all artifacts.

## JUnit Platform

### Bug Fixes

- 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

- New `printFailuresTo(PrintWriter, int)` method in `TestExecutionSummary` that allows one to specify the maximum number of lines to print for exception stack traces.
- 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.
- Exceptions thrown by test engines during discovery and execution are now reported to `TestExecutionListeners`.
- 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.
- New methods in `DiscoverySelectors` to select and execute individual tests in inherited nested classes, via specific selectors (`NestedClassSelector` and `NestedMethodSelector`).

## JUnit Jupiter

## Bug Fixes

- 

## 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`.
- New `@EnabledForJreRange` and `@DisabledForJreRange` annotations for enabling or disabling test execution over a range of JRE versions.
- 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 `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.

## JUnit Vintage

### Bug Fixes

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

## Deprecations and Breaking Changes

- 

## New Features and Improvements

- 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.