JUnit 5 Release Notes

Stefan Bechtold, Sam Brannen, Johannes Link, Matthias Merdes, Marc Philipp, Christian Stein

Version 5.3.0-SNAPSHOT

Table of Contents

This document contains the *change log* for all JUnit 5 releases since 5.1 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.

5.3.0-M1

Date of Release:

Scope:

For a complete list of all *closed* issues and pull requests for this release, consult the 5.3 M1 milestone page in the JUnit repository on GitHub.

JUnit Platform

Bug Fixes

- Full stacktrace is printed to the console, when running the `ConsoleLauncher` in `--details verbose` mode.

Deprecations and Breaking Changes

- The `junit-platform-gradle-plugin` has been discontinued and is no longer released as part of JUnit 5. Please use Gradle's native support for running tests on the JUnit Platform (requires Gradle 4.6 or higher) instead.

New Features and Improvements

- New overloaded variant of `isAnnotated()` in `AnnotationSupport` that accepts `Optional<? extends AnnotatedElement>` instead of `AnnotatedElement`.

JUnit Jupiter

Bug Fixes

-

Deprecations and Breaking Changes

-

New Features and Improvements

- New support for the IBM AIX operating system in `@EnabledOnOs` and `@DisabledOnOs`.
- New `assertThrows` methods in `Assertions` provide a more specific failure message if the supplied lambda expression or method reference returns a result instead of throwing an exception.

- New `arguments()` static factory method in the `Arguments` interface that serves as an *alias* for `Arguments.of()`. `arguments()` is intended to be used via `import static`.
- New `get<Class>(index)` Kotlin extension method to make `ArgumentsAccessor` friendlier to use from Kotlin.

JUnit Vintage

Bug Fixes

-

Deprecations and Breaking Changes

-

New Features and Improvements

-

5.2.0

Date of Release: April 29, 2018

Scope: JUnit BOM, support for Maven Surefire 2.21.0 allowing builds with Java 9 and Java 10, *argument aggregation* and *widening primitive conversion* for arguments in parameterized tests, external factory methods for `@MethodSource`, as well as various minor improvements and bug fixes.

For a complete list of all *closed* issues and pull requests for this release, consult the 5.2 M1, 5.2 RC1, and 5.2 GA milestone pages in the JUnit repository on GitHub.

Overall Improvements

- JUnit BOM: To ease dependency management using Maven or Gradle, a *Bill of Materials* POM is now provided under the `org.junit:junit-bom:5.2.0` Maven coordinates.

JUnit Platform

Bug Fixes

- Tag expressions containing spaces are now supported in the JUnit Platform Maven Surefire provider.
- Duplicate `--config` keys supplied to the `ConsoleLauncher` are now reported properly.
- Exceptions thrown in `Node.after()` (in the `HierarchicalTestEngine` infrastructure) no longer mask earlier exceptions.

New Features and Improvements

- The JUnit Platform Surefire Provider (`junit-platform-surefire-provider`) now works with and requires Surefire `2.21.0` which allows it to be used with Java 9 and Java 10.
- The default *include pattern* for filtering class names now matches test classes whose names either start with `Test` or end with `Test` or `Tests`.
    - This pattern is used by the `ConsoleLauncher`, the JUnit Platform Gradle Plugin, and the `JUnitPlatform` runner.

JUnit Jupiter

Bug Fixes

- Exceptions thrown by an `AfterAllCallback` no longer mask exceptions thrown at the class level when using the `@TestInstance(PER_CLASS)` lifecycle mode.

New Features and Improvements

- New `assertDoesNotThrow()` methods in `Assertions` which assert that the execution of a given code block does *not* throw any kind of exception.
- New `fail()` method in `Assertions` makes it possible to fail a test without an explicit failure message.
- Implicit support for *widening primitive conversion* for an argument supplied to a `@ParameterizedTest`.
    - For example, a parameterized test annotated with `@ValueSource(ints = { 1, 2, 3 })` can be declared to accept an argument of type `int`, `long`, `float`, or `double`.
- `@MethodSource` now supports `static` factory methods declared in external classes referenced by *fully qualified method name*.
- Support for aggregation of multiple `@ParameterizedTest` arguments into a single object.
    - For details, see Argument Aggregation.

JUnit Vintage

No changes.

5.1.1

Date of Release: April 8, 2018

Scope: Bug fixes and minor improvements since the 5.1.0 release

For a complete list of all *closed* issues and pull requests for this release, consult the 5.1.1 milestone page in the JUnit repository on GitHub.

JUnit Platform

New Features and Improvements

- New `findAllClassesInModule()` method in `ReflectionSupport` which enables third-party `TestEngine` implementations to scan for classes in modules — for example, when processing a `ModuleSelector` during the discovery phase.

JUnit Jupiter

New Features and Improvements

- The `ParameterContext` API used in `ParameterResolver` implementations now includes the following convenience methods for looking up annotations on parameters. Extension authors are strongly encouraged to use these methods instead of those provided in the core `java.lang.reflect.Parameter` API due to a bug in `javac` on JDK versions prior to JDK 9 which causes lookups for annotations on parameters in inner class constructors to fail consistently — for example, when resolving a parameter for a `@Nested` test class constructor.
  - `boolean isAnnotated(Class<? extends Annotation> annotationType)`
  - `Optional<A> findAnnotation(Class<A> annotationType)`
  - `List<A> findRepeatableAnnotations(Class<A> annotationType)`

JUnit Vintage

No changes.

5.1.0

Date of Release: February 18, 2018

Scope:

- Discovering tests in Java 9 modules
- Improved Kotlin support
- Programmatic extension registration via `@RegisterExtension`
- Tag expression language for filtering tests to be executed
- Annotation-based conditional test execution with support for environment variables, system properties, operating systems, JRE versions, and dynamic scripts
- Various improvements for writing parameterized tests
- Refinements to the `ExtensionContext` API
- Support for re-running individual dynamic tests, parameterized tests, and test template invocations within an IDE

For complete details consult the 5.1.0 Release Notes online.