**JUnit 5 vs JUnit 4**

JUnit 5 aims to adapt java 8 style of coding and to be more robust and flexible than JUnit 4. In this post, **JUnit 5 vs JUnit 4**, we will focus on some major differences between junit 4 and junit 5.

**1. JUnit 5 vs JUnit 4 – Annotations**

Most of annotations in both versions are same, but few differs. Here is a quick comparison.

|  |  |  |
| --- | --- | --- |
| **FEATURE** | **JUNIT 4** | **JUNIT 5** |
| Declare a test method | @Test | @Test |
| Execute before all test methods in the current class | @BeforeClass | @BeforeAll |
| Execute after all test methods in the current class | @AfterClass | @AfterAll |
| Execute before each test method | @Before | @BeforeEach |
| Execute after each test method | @After | @AfterEach |
| Disable a test method / class | @Ignore | @Disabled |
| Test factory for dynamic tests | NA | @TestFactory |
| Nested tests | NA | @Nested |
| Tagging and filtering | @Category | @Tag |
| Register custom extensions | NA | @ExtendWith |

**2. Other differences between JUnit 5 and JUnit 4**

**2.1. Architecture**

JUnit 4 has everything bundled into single jar file.

Junit 5 is composed of 3 sub-projects i.e. JUnit Platform, JUnit Jupiter and JUnit Vintage.

1. **JUnit Platform**

It defines the TestEngine API for developing new testing frameworks that runs on the platform.

1. **JUnit Jupiter**

It has all new junit annotations and TestEngine implementation to run tests written with these annotations.

1. **JUnit Vintage**

To support running JUnit 3 and JUnit 4 written tests on the JUnit 5 platform.

**2.2. Required JDK Version**

Junit 4 requires Java 5 or higher.

Junit 5 requires Java 8 or higher.

**2.3. Assertions**

In Junit 4, [org.junit.Assert](https://junit.org/junit4/javadoc/4.12/org/junit/Assert.html) has all assert methods to validate expected and resulted outcomes.  
They accept extra parameter for error message as FIRST argument in method signature. e.g.

|  |
| --- |
| public static void assertEquals(long expected, long actual)  public static void assertEquals(String message, long expected, long actual) |

In JUnit 5, [org.junit.jupiter.Assertions](https://junit.org/junit5/docs/current/api/org/junit/jupiter/api/Assertions.html) contains most of assert methods including additional assertThrows() and assertAll() methods. assertAll() is in experimental state as of today, and is used for grouped assertions.  
JUnit 5 assertions methods also have overloaded methods to support passing error message to be printed in case test fails e.g.

|  |
| --- |
| public static void assertEquals(long expected, long actual)  public static void assertEquals(long expected, long actual, String message)  public static void assertEquals(long expected, long actual, Supplier messageSupplier) |

**2.4. Assumptions**

In Junit 4, [org.junit.Assume](https://junit.org/junit4/javadoc/4.12/org/junit/Assume.html) contains methods for stating assumptions about the conditions in which a test is meaningful. It has following five methods:

1. assumeFalse()
2. assumeNoException()
3. assumeNotNull()
4. assumeThat()
5. assumeTrue()

In Junit 5, [org.junit.jupiter.api.Assumptions](https://junit.org/junit5/docs/current/api/org/junit/jupiter/api/Assumptions.html) contains methods for stating assumptions about the conditions in which a test is meaningful. It has following three methods:

1. assumeFalse()
2. assumingThat​()
3. assumeTrue()

**2.5. Tagging and Filtering**

In Junit 4, @category annotation is used.

In Junit 5, @tag annotation is used.

**2.6. Test Suites**

In Junit 4, @RunWith and @Suite annotation. e.g.

|  |
| --- |
| import org.junit.runner.RunWith;  import org.junit.runners.Suite;    @RunWith(Suite.class)  @Suite.SuiteClasses({          ExceptionTest.class,          TimeoutTest.class  })  public class JUnit4Example  {  } |

In Junit 5, @RunWith, @SelectPackages and @SelectClasses e.g.

|  |
| --- |
| import org.junit.platform.runner.JUnitPlatform;  import org.junit.platform.suite.api.SelectPackages;  import org.junit.runner.RunWith;    @RunWith(JUnitPlatform.class)  @SelectPackages("com.howtodoinjava.junit5.examples")  public class JUnit5Example  {  } |

**2.7. 3rd Party Integration**

In Junit 4, there is no integration support for 3rd party plugins and IDEs. They have to rely on [reflection](https://howtodoinjava.com/java/related-concepts/real-usage-examples-of-reflection-in-java/).

JUnit 5 has dedicated sub-project for this purpose i.e. JUnit Platform. It defines the TestEngine API for developing a testing framework that runs on the platform.