


JUnit 5 Introduction

By Robert Smieja

Overview

- What is JUnit 5?
 - JUnit 5 Setup/Requirements
 - Standard JUnit 5 Test
 - JUnit Vintage
 - New JUnit 5 Features
 - DisplayName/Tagging
 - Assertions
 - Nested Tests
 - Dependency Injection
 - Interfaces/Default Methods
 - Parameterized Tests
 - Dynamic Tests
- 

What is JUnit 5?

- JUnit 5 is the latest version of JUnit, a framework to write Unit tests
 - Re-imagined to take advantage of new Java 8 features
- JUnit 5 is composed of 3 modules:
 - JUnit Platform - Found for launching testing frameworks, defines the *TestEngine* API
 - JUnit Jupiter - The new JUnit 5 *TestEngine*
 - JUnit Vintage - A project providing a *TestEngine* to run JUnit 3 and JUnit 4 tests
- Current version is JUnit5 - Milestone 4



JUnit 5 Setup/Requirements

- Java 8
- IDE
 - Eclipse Oxygen (4.7) w/ [Patch](#)
 - IntelliJ IDEA 2017.1 or later
- Build Tools
 - [Gradle](#)
 - [Maven](#)
- Other options
 - Console Launcher
 - Use JUnit 4 to `@RunWith(JUnitPlatform.class)`



Standard JUnit 5 Test

- The basics of JUnit have stayed the same
 - [StandardJUnit5Tests.java](#)

<u>JUnit 4</u>	<u>JUnit 5</u>
@Before	@BeforeEach
@BeforeClass	@BeforeAll
@After	@AfterEach
@AfterClass	@AfterAll
@Ignore	@Disabled
@RunWith	@ExtendWith



JUnit Vintage

- JUnit Vintage is the name of the *TestEngine* that runs JUnit 3/4 tests
- All classes related to JUnit 5 are in their own package
 - org.junit.jupiter
- Limited support for JUnit @Rules
 - Supported types are ExternalResource, Verifier, and ExpectedException
- To add support, make sure the JUnit Vintage jar exists
 - Group: org.junit.vintage, Artifact: junit-vintage-engine
 - [JUnit4Tests.java](#)



New JUnit 5 Features - @DisplayName/@Tag

- @DisplayName
 - New annotation for Classes and Methods
 - Used by test runners and test reporting for display purposes
 - [DisplayNameTests.java](#)
- @Tag
 - New annotation for Classes and Methods
 - Used by test runners and test reporting for filtering purposes
 - [TagTests.java](#)



New JUnit 5 Features - Assertions

- New types of Assertions that leverage Java 8's Lambda Expressions:
 - Grouped Assertions - [GroupedAssertionTests.java](#)
 - Timeout - [TimeoutAssertionTests.java](#)
 - Exceptions - [ExceptionAssertionTests.java](#)



New JUnit 5 Features - Nested Tests

- New `@Nested` annotation allows use of non-static inner classes
 - Java does not allow *static* members in inner-classes
 - `@BeforeAll` and `@AfterAll` do not work
 - [NestedTests.java](#)



New JUnit 5 Features - Dependency Injection

- Test constructors and test methods are now allowed to take arguments
 - Enables Dependency Injection
- JUnit 5 uses registered *ParameterResolver* to inject the correct type
 - [DependencyInjectionTests.java](#)



New JUnit 5 Features - Interfaces/Default Methods

- Java 8 allows default methods on interfaces
- JUnit 5 fully supports Java 8 interfaces
 - *@Test, @BeforeEach, @AfterEach, etc*
 - *@Tag, @ExtendWith*
 - [InterfaceTests.java](#)



New JUnit 5 Features - Parameterized Tests

- Parameterized Tests allow running the same test with different arguments
- Requires an additional dependency
 - Group: org.junit.jupiter, Artifact: junit-jupiter-params
- Tests are declared using *@ParameterizedTest*
- Requires a “Source” for arguments
 - [ParameterizedTests.java](#)



New JUnit 5 Features - Dynamic Tests

- New annotation *@TestFactory* enables methods that create Tests
 - Returns a Stream of `DynamicTest` at runtime
 - Does not execute *@BeforeEach* or *@AfterEach* between tests
 - [DynamicTests.java](#)



Questions?



Additional Resources

- JUnit 5 Website - <http://junit.org/junit5/>
- JUnit 5 User Guide - <http://junit.org/junit5/docs/current/user-guide/>
- JUnit 5 Samples - <https://github.com/junit-team/junit5-sample>
- Code from presentation - <https://github.com/robertsmieja/junit5-examples>

