Java SE 17 Unit Testing with JUnit

Understanding JUnit

Jim Weaver

Developer, Trainer and Author

www.codeweaver.org

Java SE 17 Unit Testing with JUnit

Version Check

Version Check

This course was created by using:

- Java SE 17
- Junit 5.8.2

What is JUnit?

The most widely used unit testing library for Java

- Write test code to verify production code
- Test libraries provide tools to verify results from the code under test

The library that spread unit testing as a developer practice

> “Never in the field of software development have so many owed so much to so few lines of code.”

Martin Fowler regarding JUnit

Outcomes

Create and run JUnit tests

Apply a wide range of assertions made available by Junit to verify code

Control the execution of tests

- Lifecyle of tests
- Conditional and repeating test execution

Improve test result reports

Understand some of the advantages of writing and having unit tests

Won't cover test-driven or test-first development – another course covers

Prerequisites

Just know a little Java!

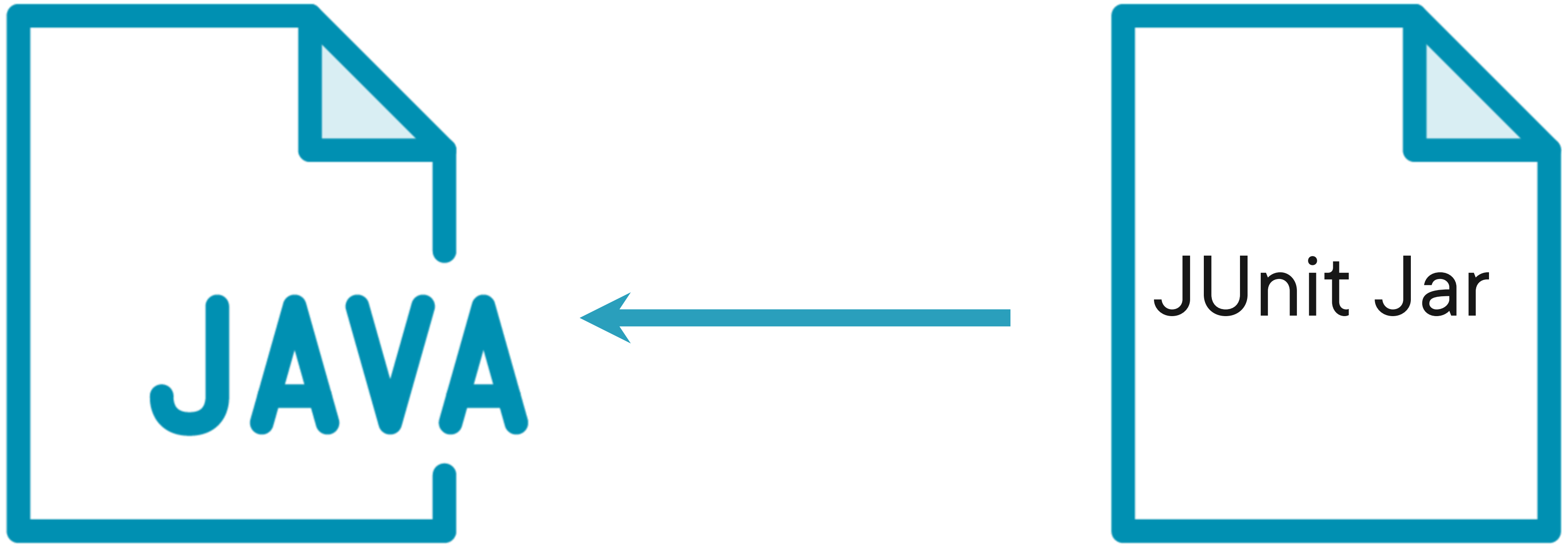
Up Next:
Understanding the JUnit Library

Understanding the JUnit Library

Java project dependency

Java archive (jar) file(s)

Need to be on classpath



```java
public class DurationParserTest {

   @Test
   public void translateWeeksToDays() {
      assertEquals(7, DurationParser.parseDays("1 week"));
   }

   @Test
   public void returnZeroIfUnitNotRecognized() {
      assertEquals(0, DurationParser.parseDays("2 blarghs"));
   }
. . .
```

Unit Tests are Contained in Java Class Files

- **Classes containing tests are part of the project, but normally kept separate from production classes**
- **A single unit test class may contain multiple tests, one method for each test**

JUnit5 was released in 2017

It is a complete rewrite

Takes advantage of Java 8 features

- Primarily lambda expression support

Composed of multiple jar files

- Jupiter (writing tests)
- Platform (running tests and tool integration)
- Vintage (older JUnit test support)

Up Next:
Following Along

Following Along

Requirements

JDK SE 17 or higher

Maven 3.8 or higher

An IDE – using IntelliJ for demonstration

A Git client

- Alternatively, use zip archive of code

Up Next:
Installing and Running JUnit