# JUNIT

## IRP Learning Program 2014

People matter, results count.

# Objectives

- Purpose:
  - To learnt the Unit testing by this tool
- Product:
  - Introduction
  - Basic Annotations
  - Avanced Testing
  - Integration Testing
  - Beyond JUnit
- Process:
  - Basic Junit strategy
  - Advanced Level Testing.

# Table of Contents

## Junit Overview

- Manual Testing - Automated Testing

- JUnit is a unit testing framework for the Java Programming Language.

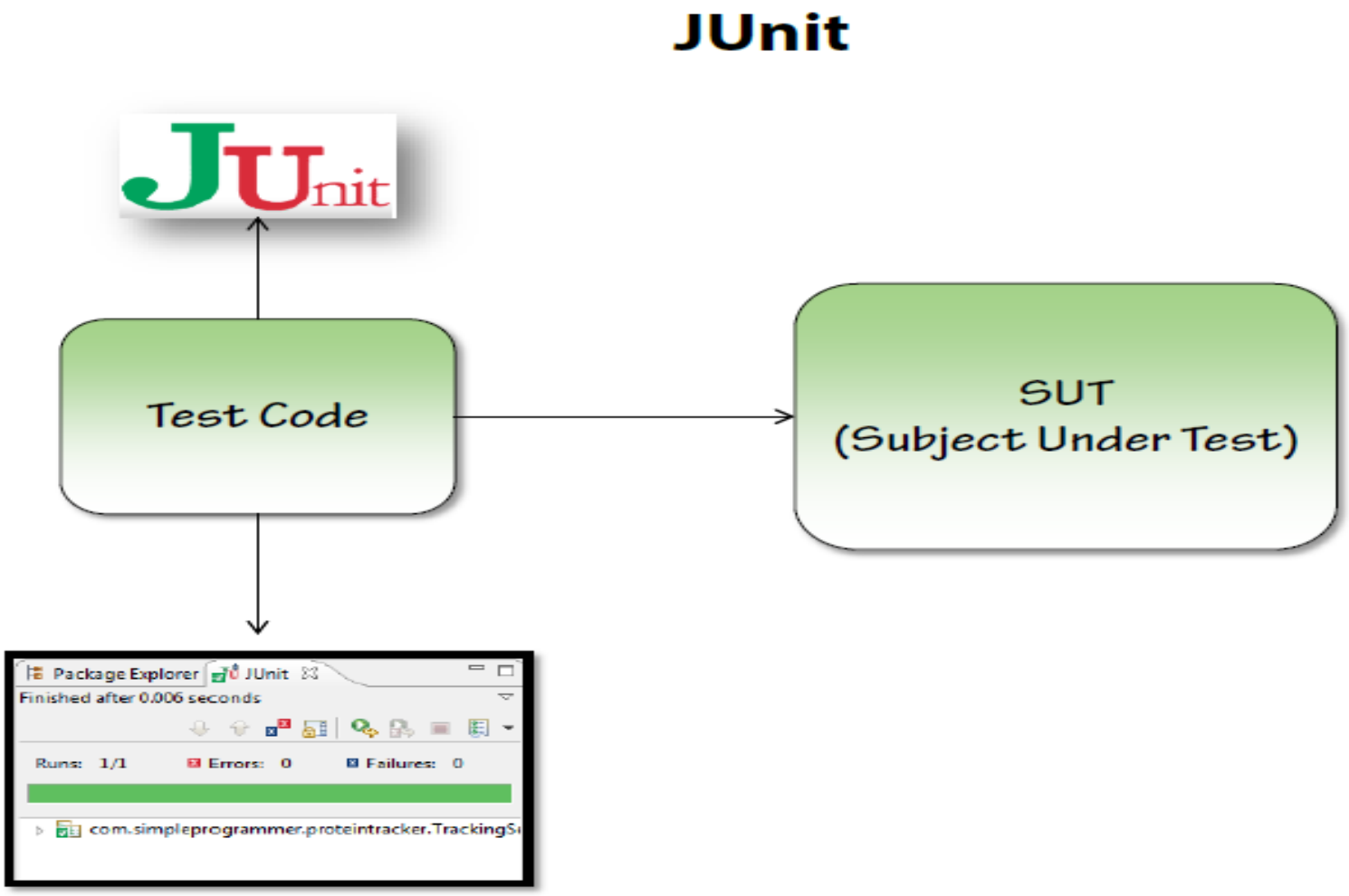
# Junit Overview

**Features:**

- open source framework
- Annotations
- Asserts
- Test setup and teardown
- Exception testing
- Test suites
- Parameterized testing
- Assumptions
- Rules
- Theories
- Integration with popular build systems
- Test Runners
- Simple and elegant

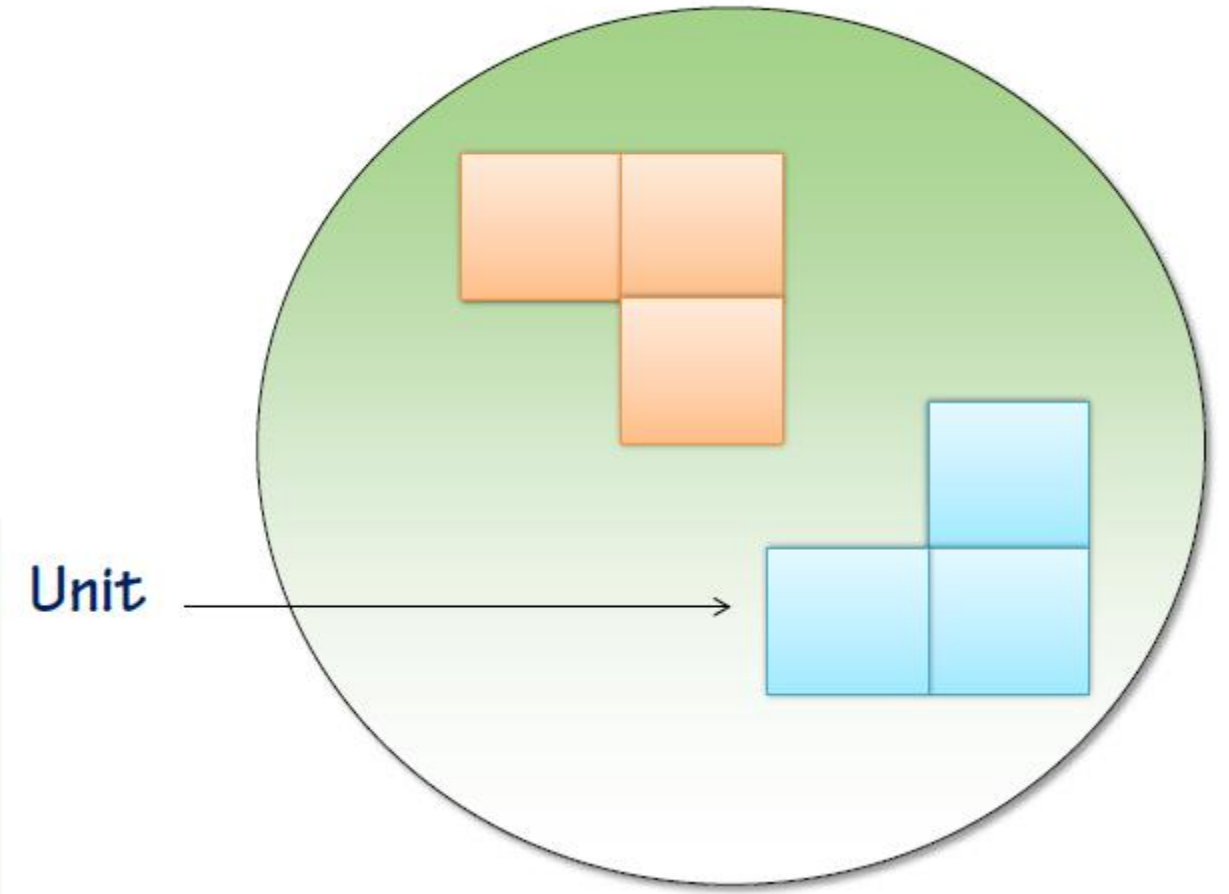
# Junit Overview

## Unit Test Case



# Junit Overview

## What is Unit Testing?



# Junit Overview

## History of Junit:

- Introduced by "**Kent Beck**"
- The origin of the elite fighting force known as "J Unit"
- Creator of Sunit which led to Junit and all the derivatives thereof
  - 1994 → Sunit
  - 1997 → Junit
  - 2000 → Junit.org
  - 2002 → Eclipse supports
- Junit still continues to evolve

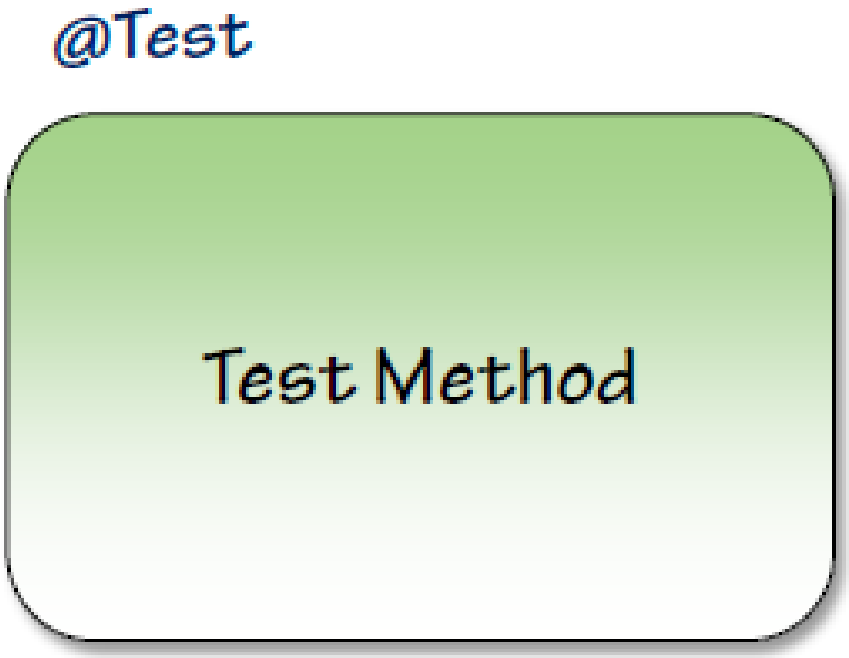
# Junit Overview

## Setting up Junit

- Directly use from IDE (Eclipse)
- Download the jars and manually include it in IDE.

# Junit Basics

JUnit Test Methods



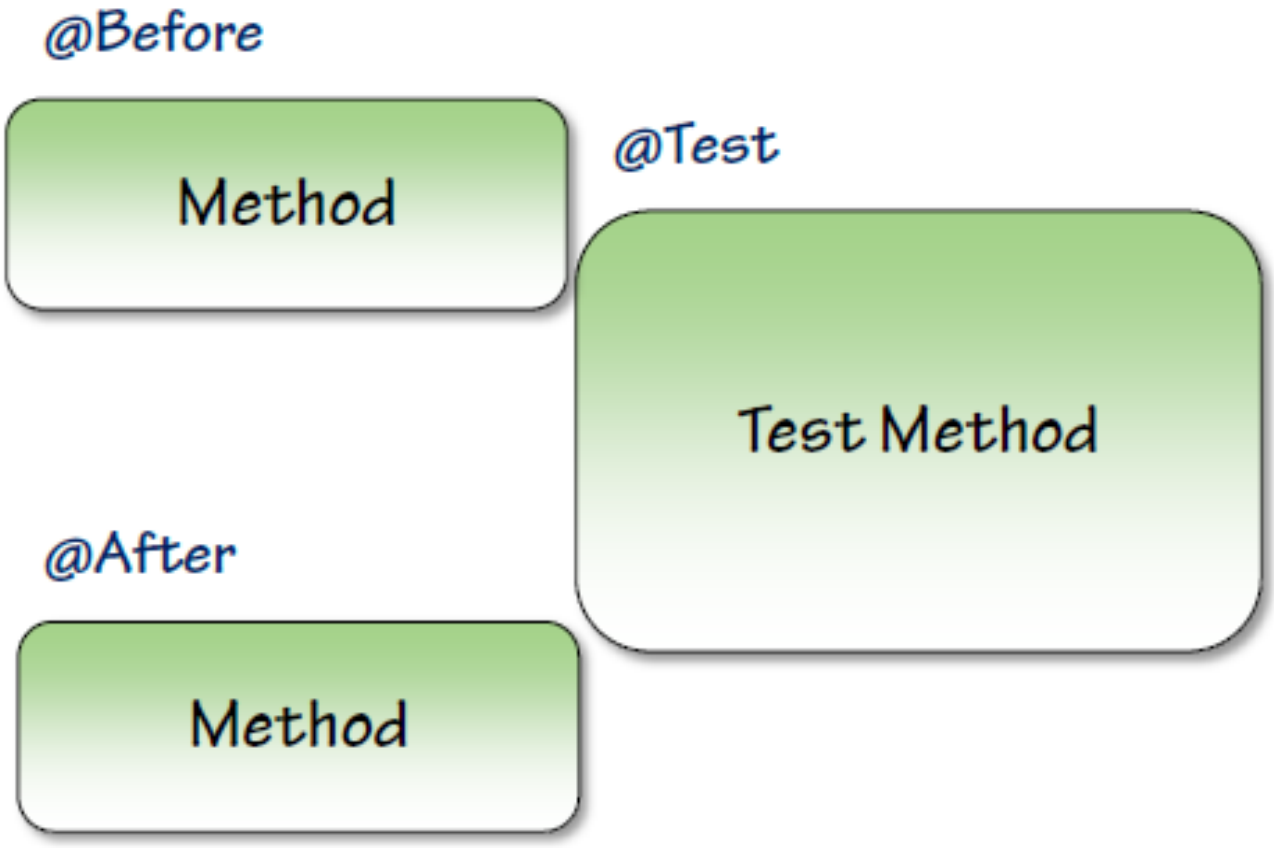
# Junit Basics

## JUnit Annotations (Basic)

- @Test
- @Before - setUp
- @After - tearDown
- @BeforeClass
- @AfterClass
- @Ignore
- @Test(expected = Exception.class)
- @Test(timeout = 100)

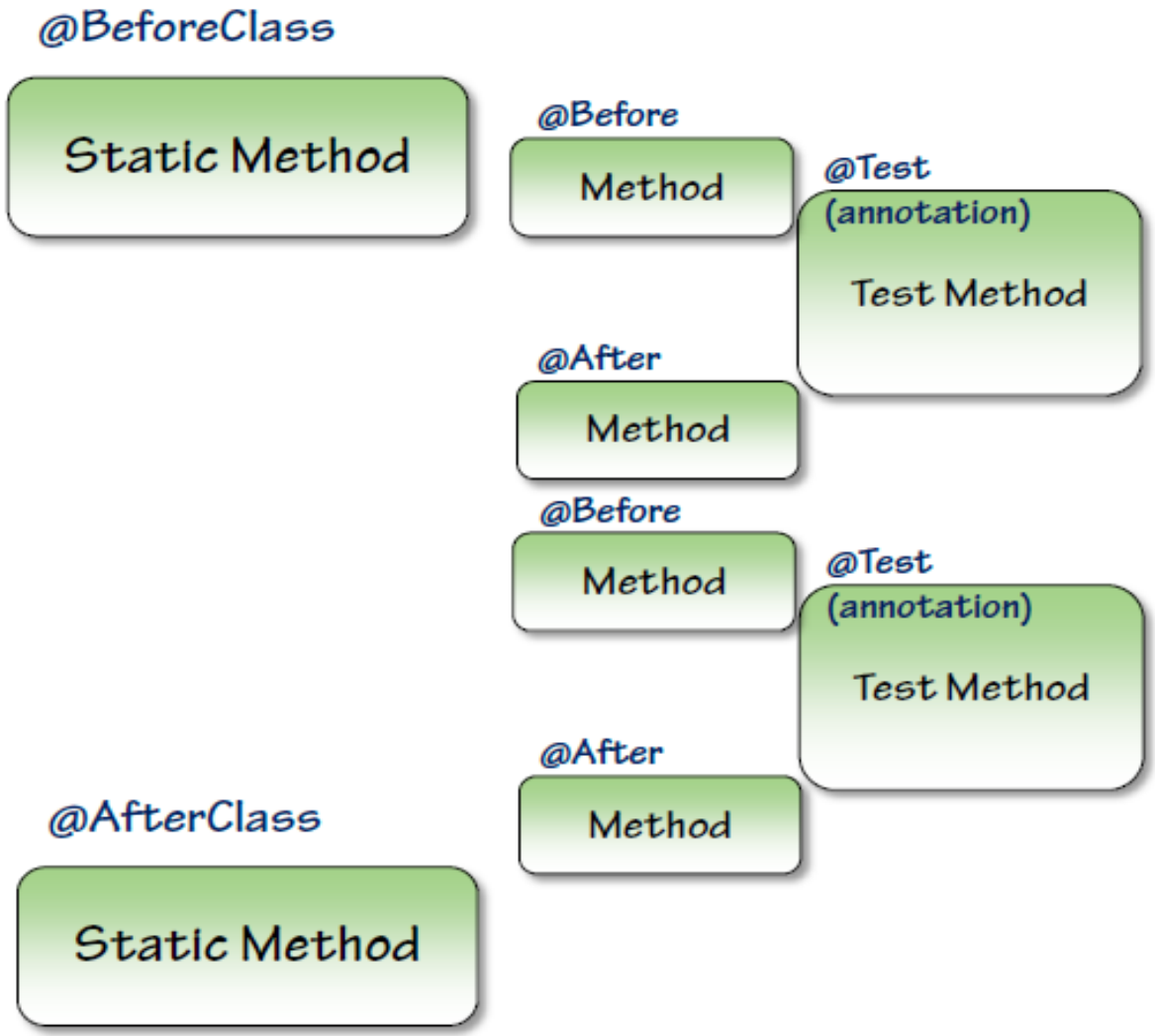
# Junit Basics

## Before And After:



# Junit Basics

## BeforeClassand AfterClass

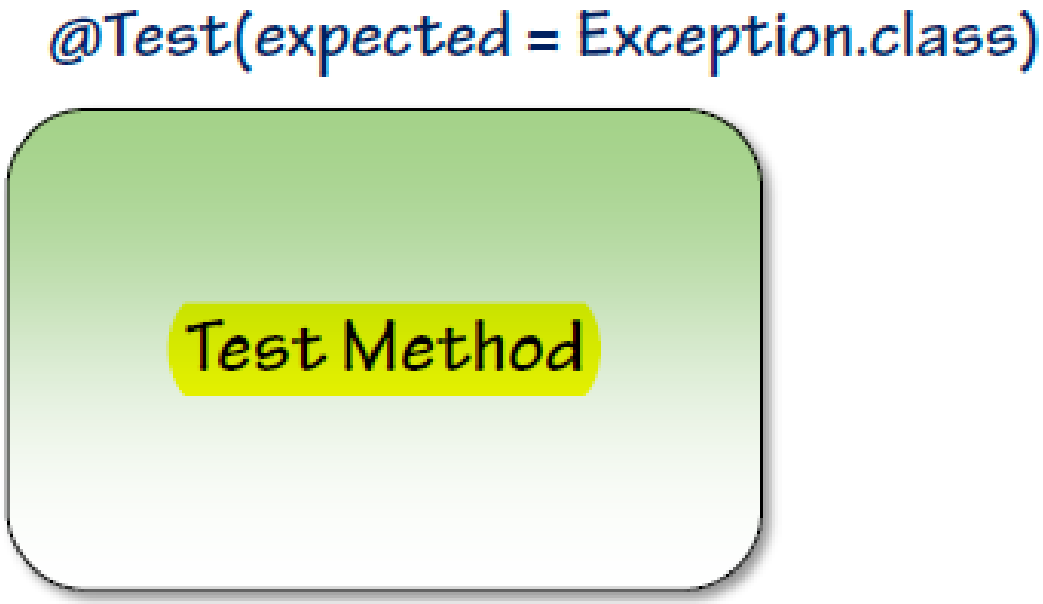




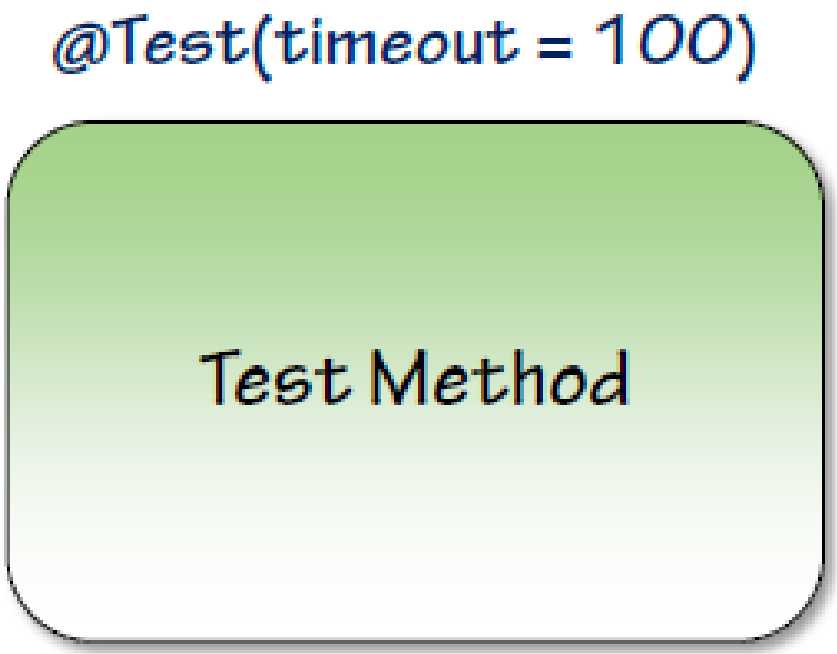
# Junit Basics

## Exception Testing



# Junit Basics

## Timing Out:





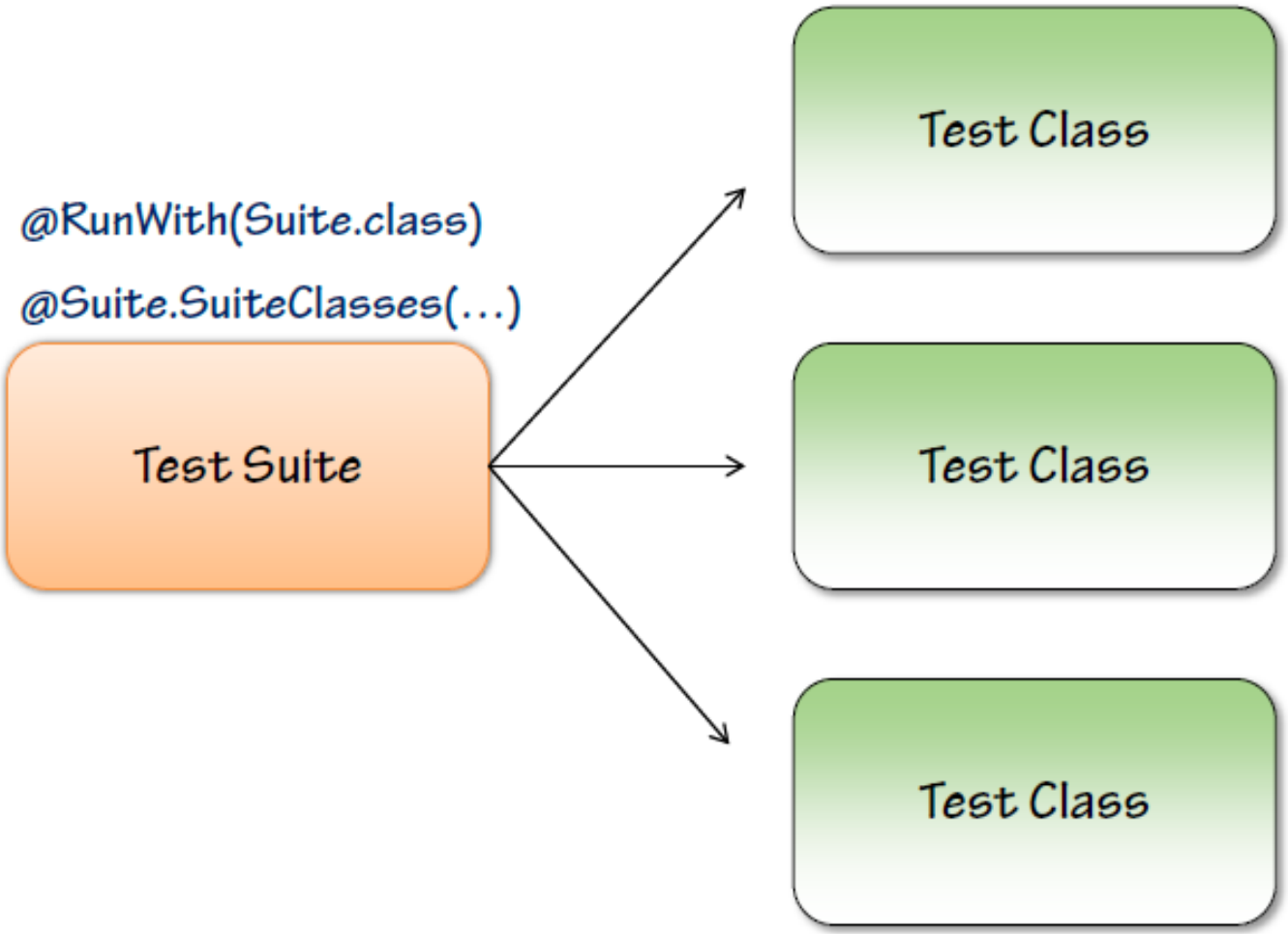
# Junit Basics

## Assertions (Basic)

- assertArrayEquals
- assertEquals
- assertTrue
- assertFalse
- assertNull
- assertNotNull
- assertSame
- assertNotSame
- fail

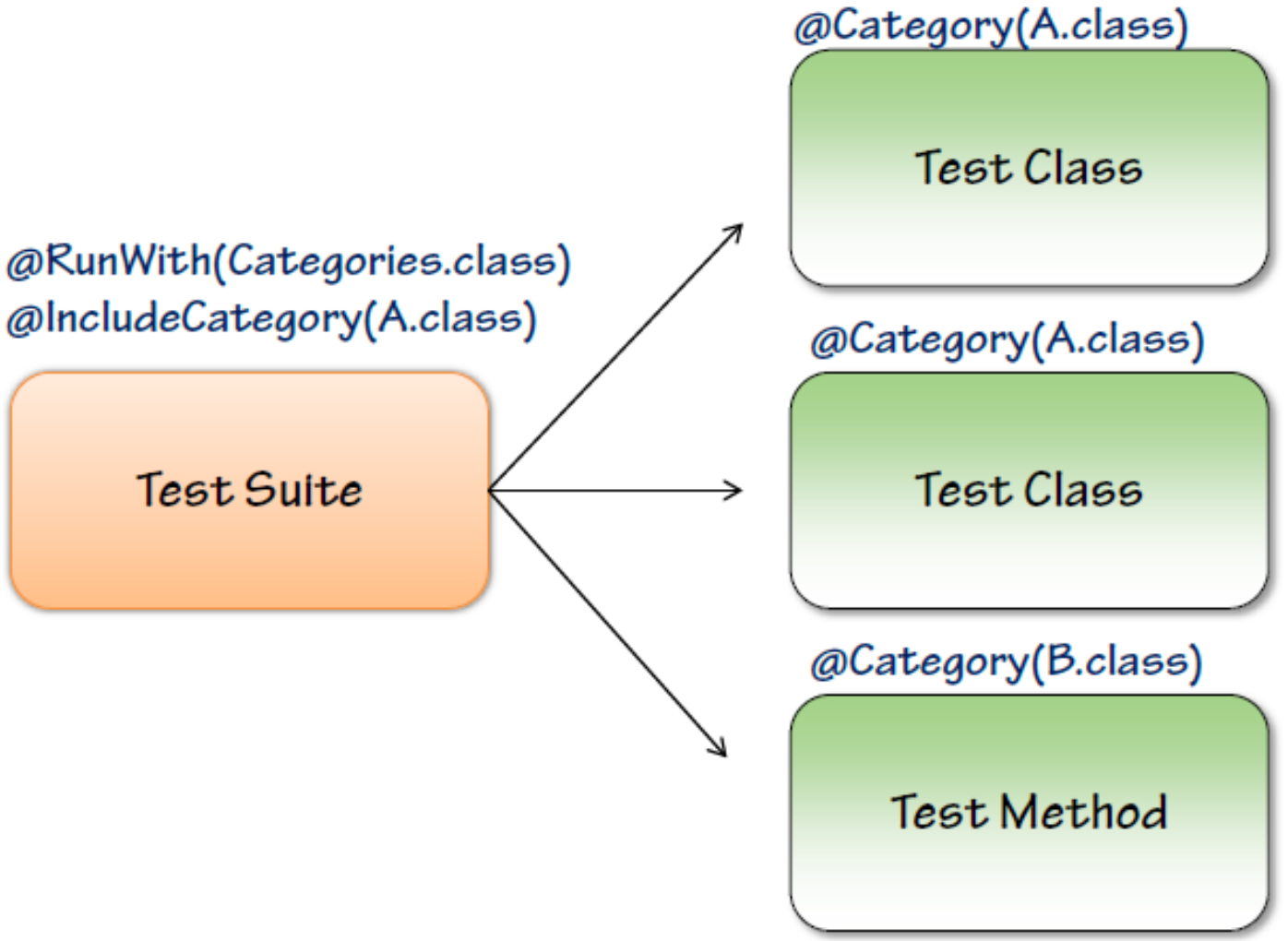
# Advanced Junit

## Test Suites



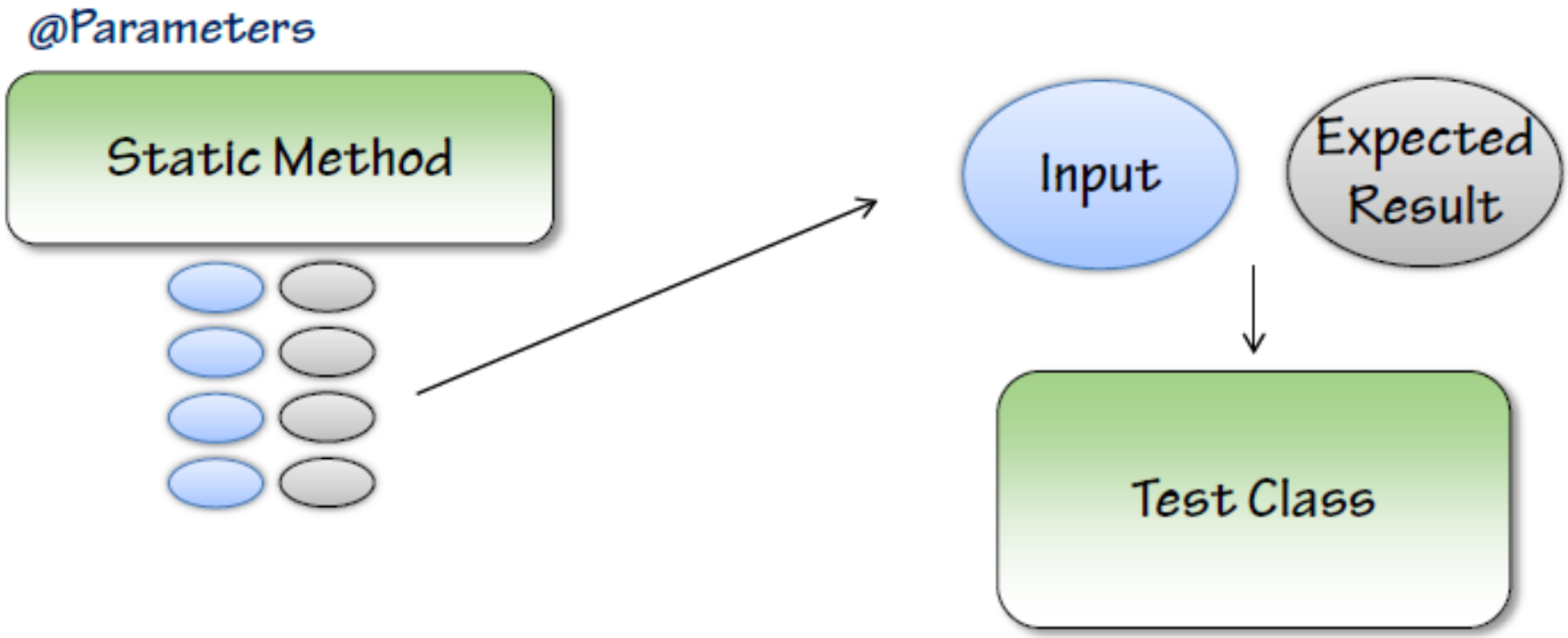
# Advanced Junit

## Categories



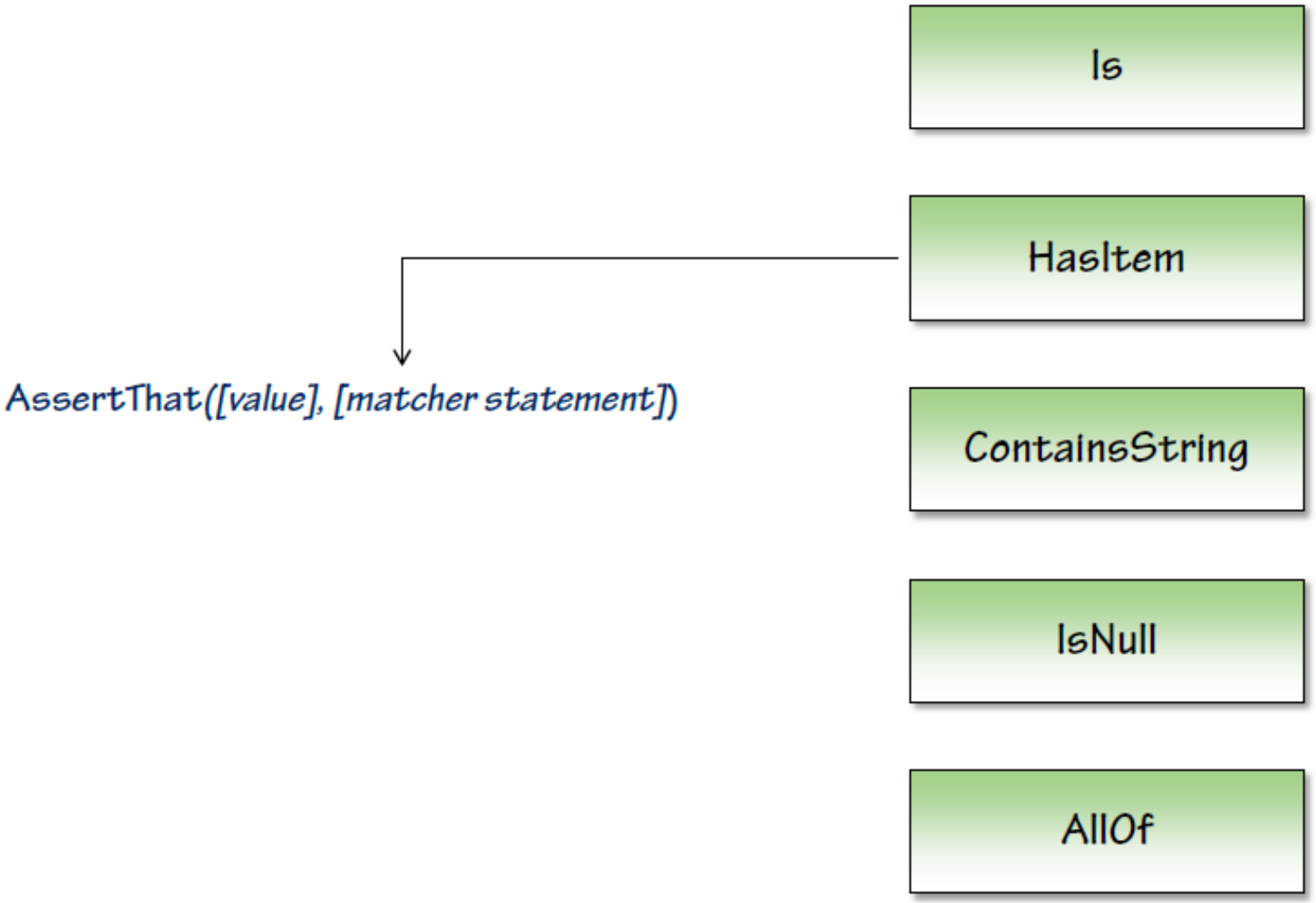
# Advanced Junit

## Parameterized Tests



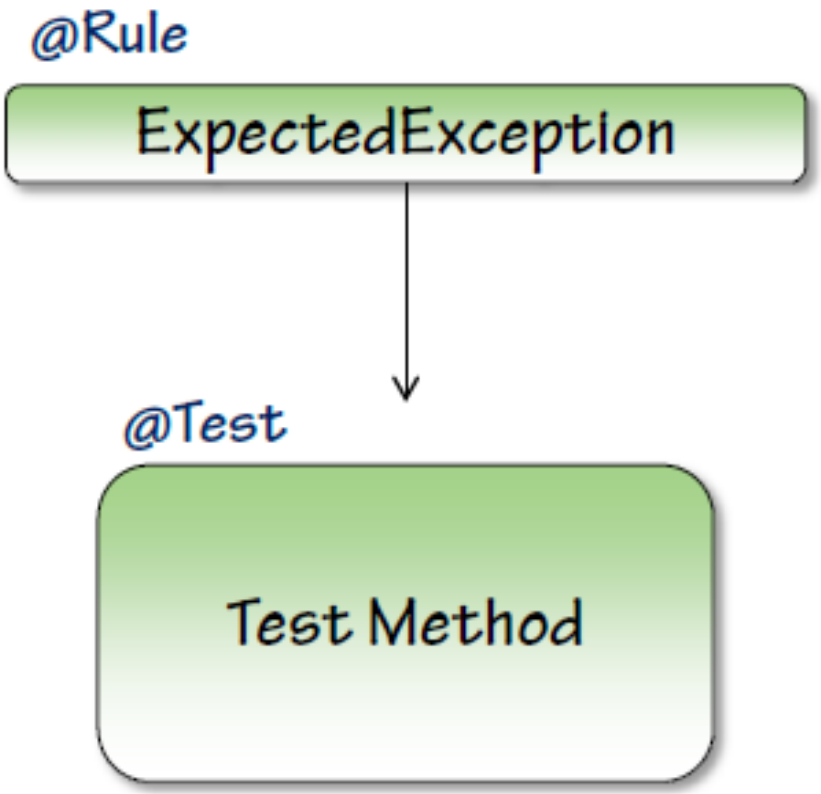
# Advanced Junit

## Advanced Assertions



# Advanced Junit

## Advanced Exception Testing



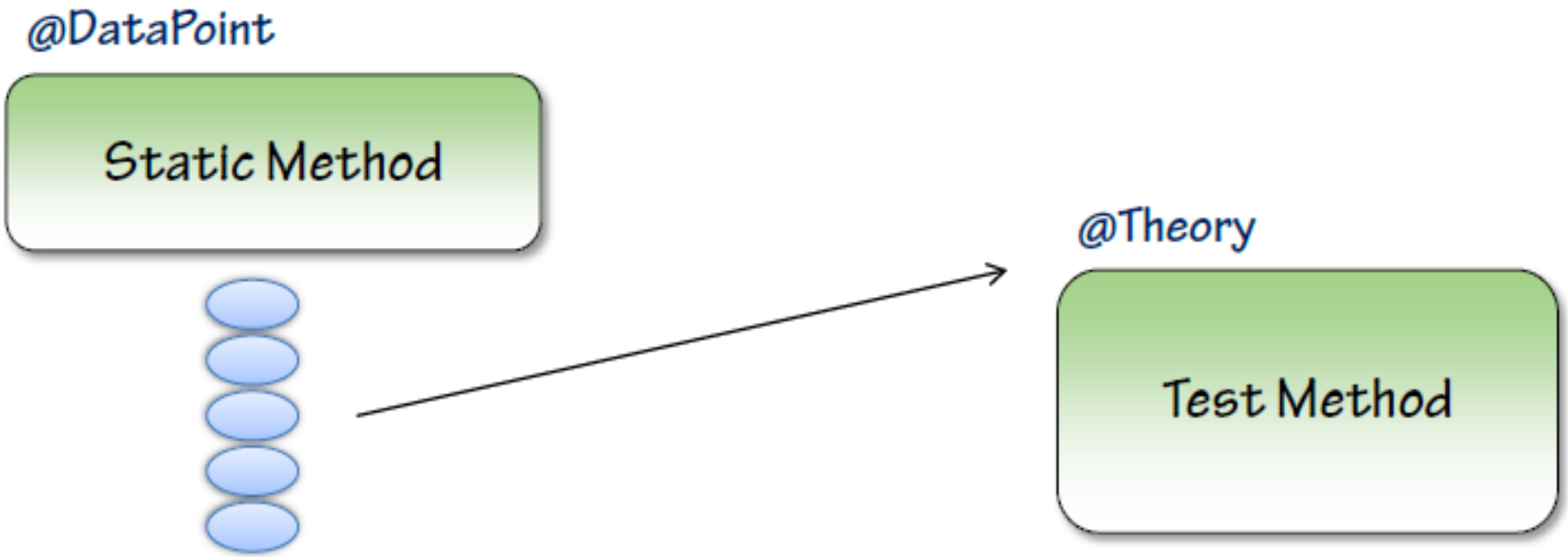
# Advanced Junit

## Rules - @ClassRule

- TemporaryFolder
- ExternalResource
- ErrorCollector
- Verifier
- TestWatcher
- TestName
- Timeout
- ExpectedException
- RuleChain

# Advanced Junit

## Theories



# Integrating JUnit

# Integrating JUnit

- Console Runners
- Command Prompt Runners

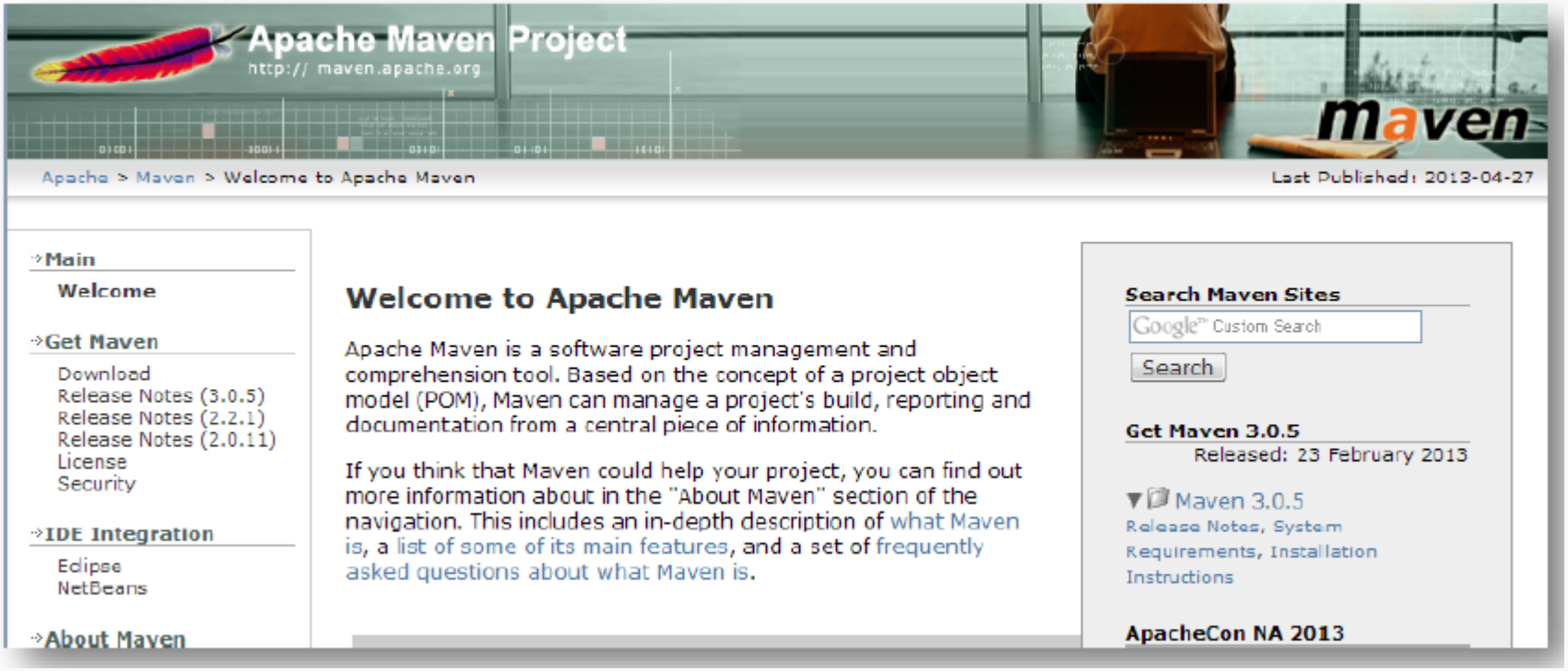
- **Org.junit.core.JUnitCore** class used to start the runners.

# Integrating JUnit

## Junit And Ant

# Integrating JUnit

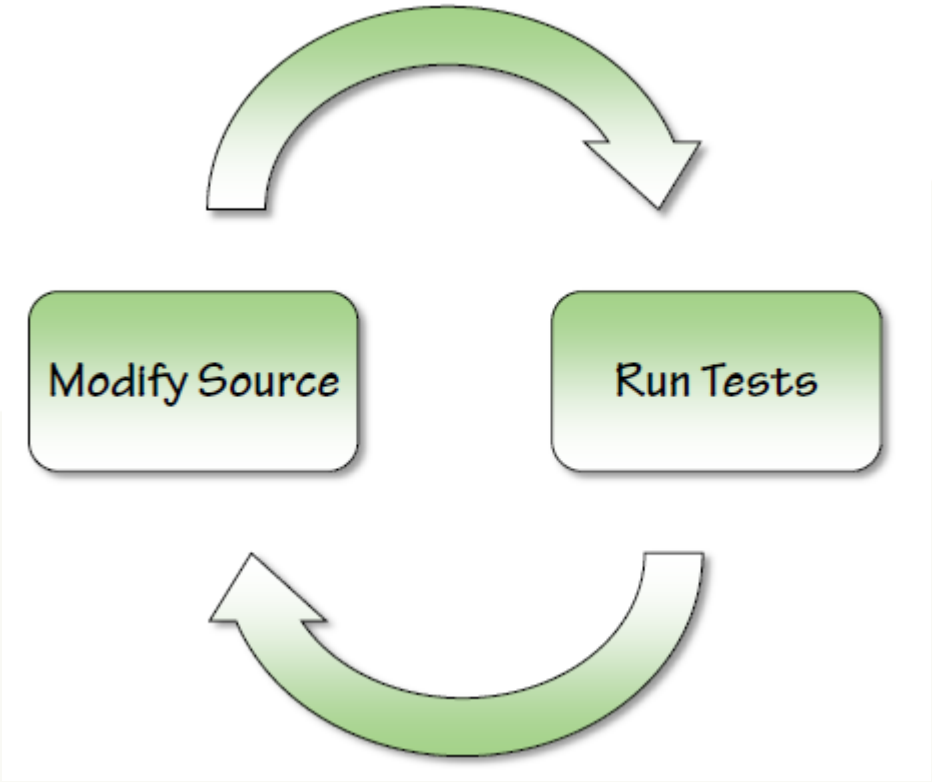
## Junit And Maven



## Eclemma - CodeCoverage

# Beyond JUnit



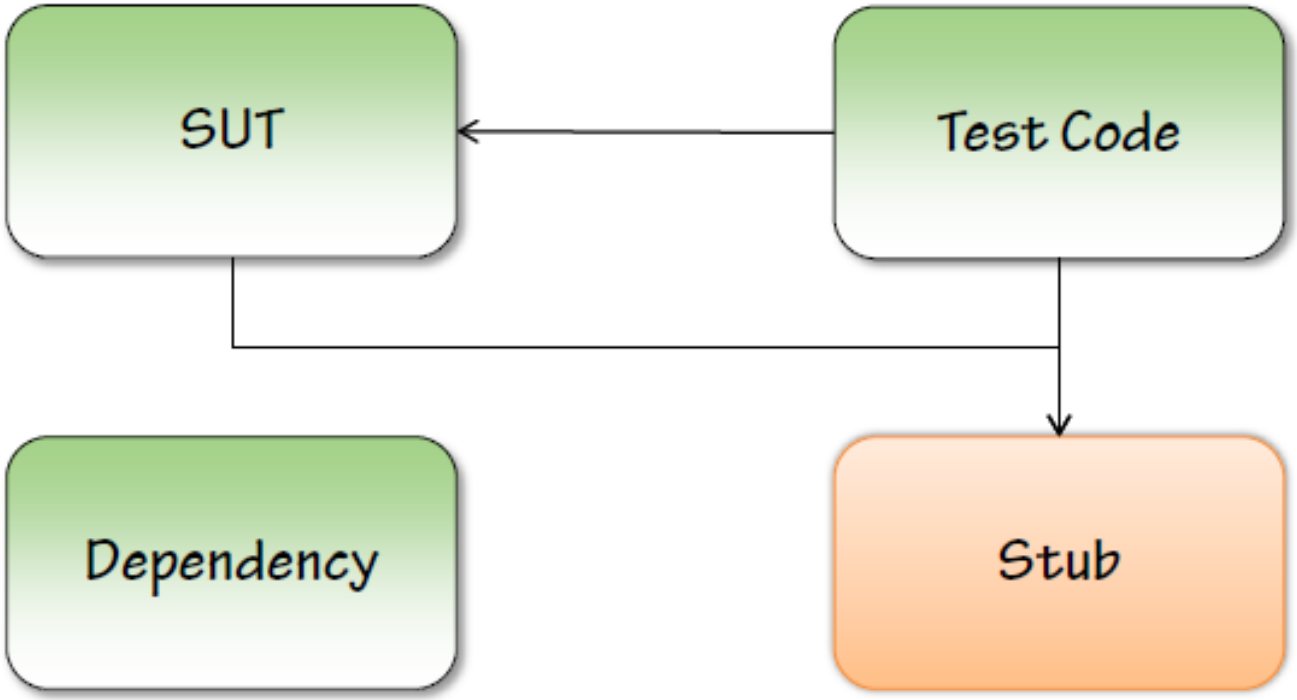


# Beyond JUnit

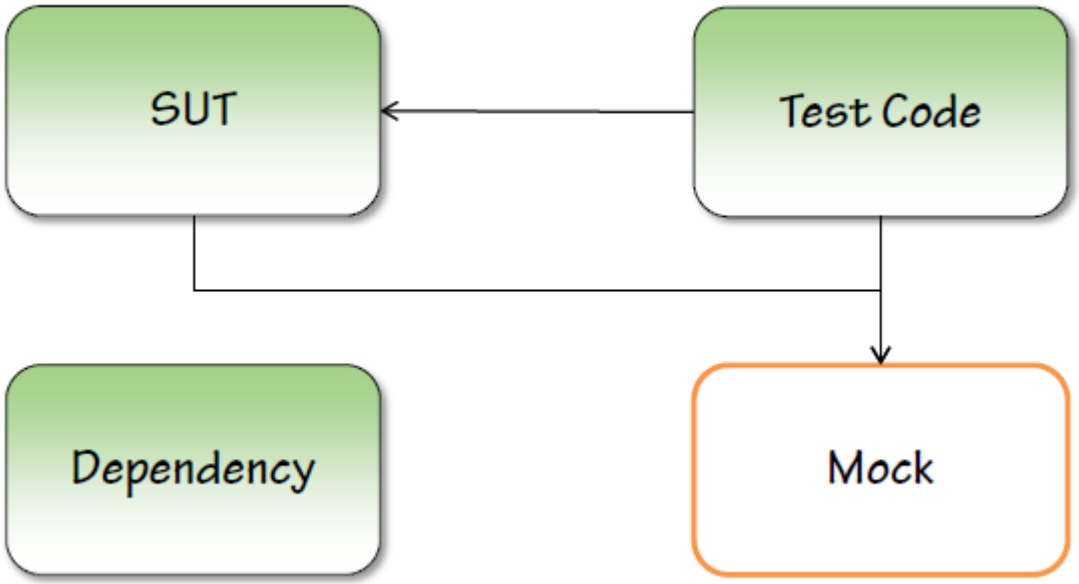
## Dependencies

## Stubs



# Beyond JUnit

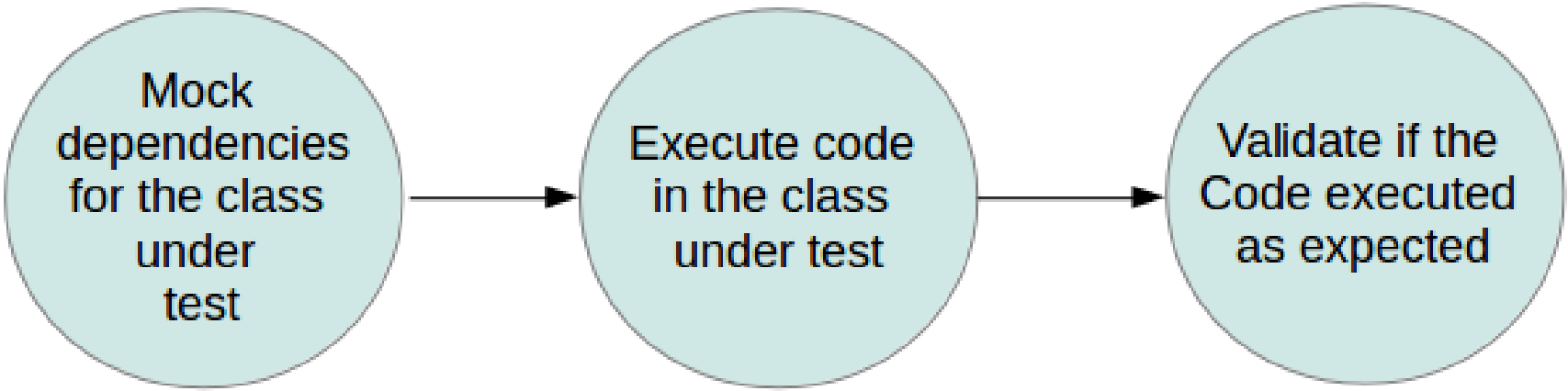
## Mocks



# Mock

- A *dummy object* is passed around but never used, i.e., its methods are never called. Such an object can for example be used to fill the parameter list of a method.
- *Fake* objects have working implementations, but are usually simplified. For example, they use an in memory database and not a real database.
- A *mock object* is a dummy implementation for an interface or a class in which you define the output of certain method calls.

# Mock



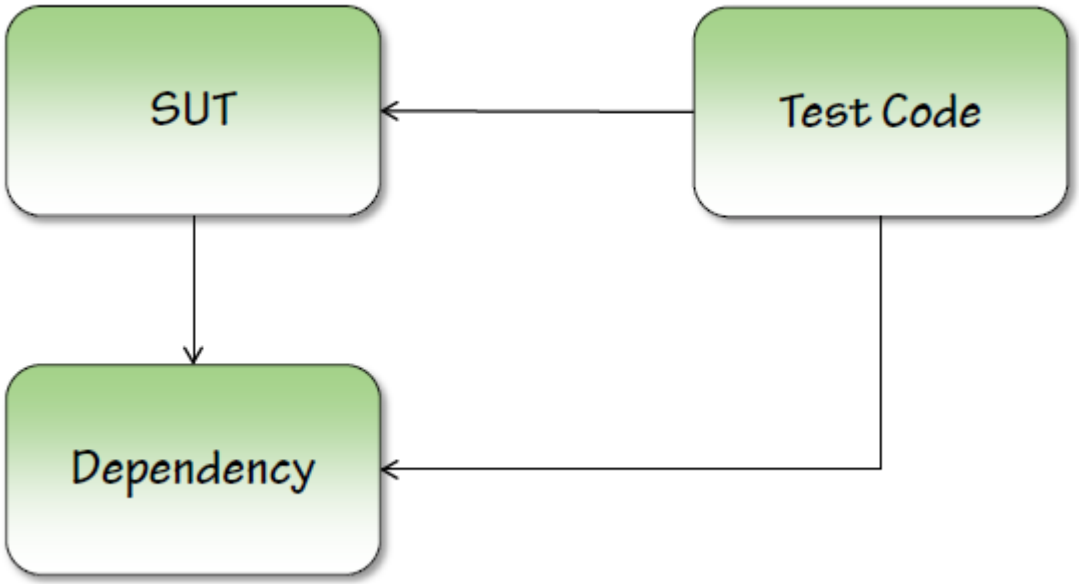
# Mock

Mockito has certain limitations. It can not test the following constructs:

- final classes
- anonymous classes
- primitive types

# Beyond JUnit

## Integration Testing



# Beyond JUnit

- SMS notifier
- Selenium

# Recap

timeout

Annotations

Assertions

JUnit

SUnit

Test-Case

Exception

TestSuite

# Thank You For Your Time

People matter, results count.