Cognizant | Academy
Passion for making a difference

# Exploring JUnit 4.x

Targeted at: Entry Level Trainees

## Session 7: About Test Annotation

# About the Author

| | |
|---|---|
| **Created By:** | B Sai Prasad, (105582) |
| **Credential Information:** | Sun Certified Java Programmer, Microsoft Certified Technology Specialist, PMI-certified Project Management Professional |
| **Version and Date:** | JUnit/PPT/1110/1.0 |

**Cognizant Certified Official Curriculum**


Cognizant | Academy
Passion for making a difference

# Icons Used

**Questions**

**Tools**

**Hands-on Exercise**

**Coding Standards**


**Test Your Understanding**

**Reference**

**Try it Out**

**A Welcome Break**


**Contacts**

Cognizant | Academy
Passion for making a difference

# Session 07: About Test Annotation overview

- **Introduction:**
  - The *@Test* annotation tells JUnit that the **`public void`** method to which it is attached can be run as a test case
  - **`AssertionFailedError`** thrown by the test will be reported by JUnit as a failure
  - If no exceptions are thrown, the test is assumed to have succeeded
  - In this session, associates would learn the two optional parameters of *@Test* annotation

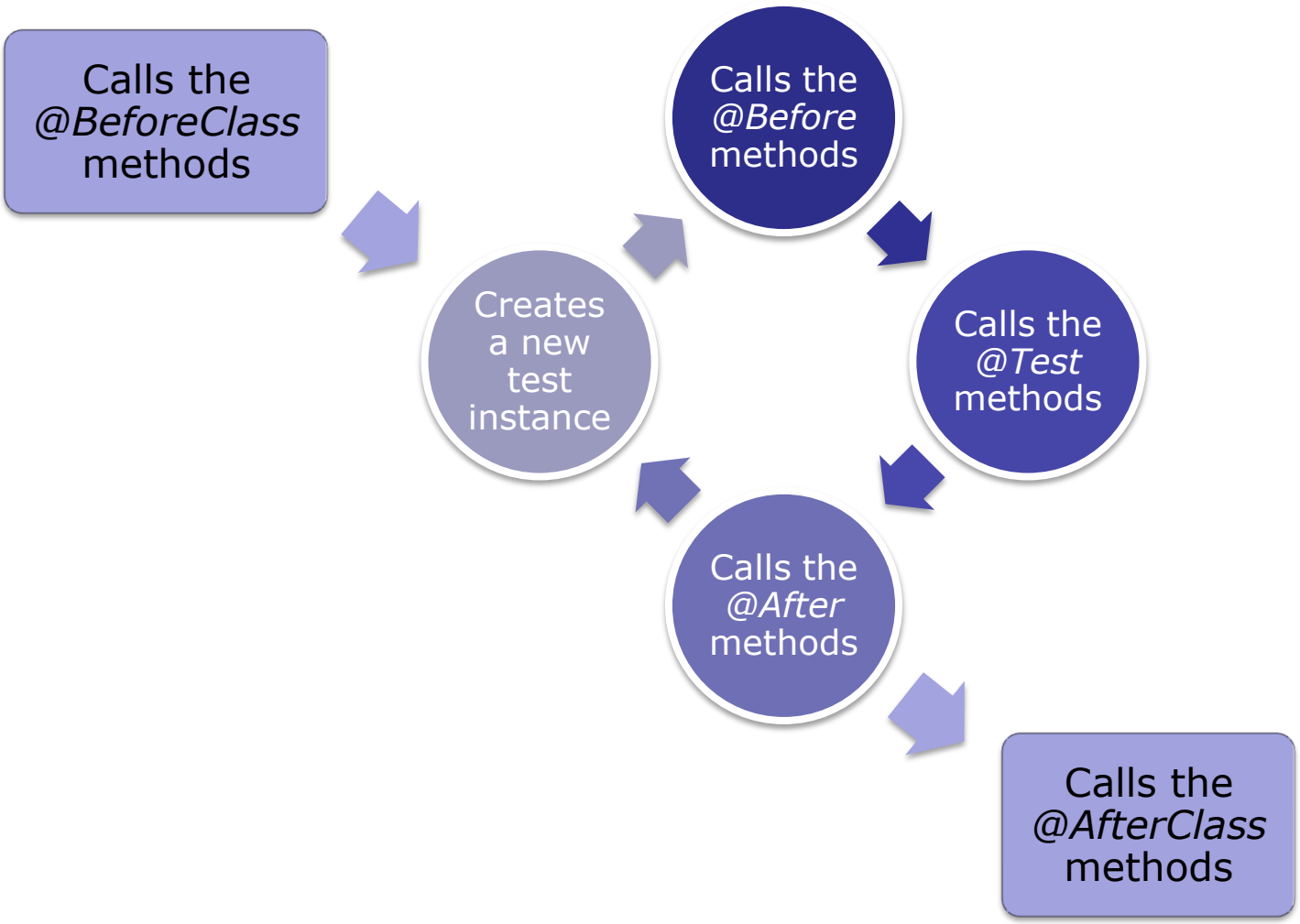
# Session 07- About Test Annotation: Objectives

- **Objectives:**

After completing this chapter, associates will be able to:

- Describe the test execution cycle
- Check for exceptions thrown by test
- Explain the limitation of *expected* parameter
- Use timeouts to fail test that longer than required

# Test Execution Cycle



# Checking For Exceptions

- To check that an exception is correctly thrown under certain circumstances
- To use the *expected* parameter in the *@Test* annotation to test the exception that should be thrown if all goes according to plan

```
@Test(expected = InvalidYearException.class)
public void futureYearsShouldBeInvalid()
            throws InvalidYearException {

      double income = 30000;
      int year = getCurrentYear() + 1;

      taxCalculator.calculateIncomeTax(income, year);
}
```

# Limitation Of Expected Parameter

- Running assertions against the exception message is not possible, if *expected* parameter is used
- Traditional approach ensures that the exception is caught and the message is asserted

```
@Test
public void exceptionShouldIncludeAClearMessage() {
  ...
  try {
    taxCalculator.calculateIncomeTax(income, year);
    fail("calculateIncomeTax() should throw an exception.");
  } catch (InvalidYearException expected) {
    assertEquals(
        expected.getMessage(),
        "No tax calculations available yet for the year 2010");
  }
}
```

# Using Timeouts

- A simple performance testing involves making sure that a particular test always executes within a certain timeframe
- To do this, specify the *timeout* parameter of the @*Test* annotation (in milliseconds)

```
@Test(timeout = 1000)
public void shouldCalculateCorrectTax()
                throws InvalidYearException {

    for (int i = 1; i < 50; i++) {
        ...
        double calculatedTax = ...
        assertEquals(expectedTax, calculatedTax, 0.0);
    }
}
```

# Demonstration

- Use *expected* parameter in *@Test* annotation
- Assert the exception message by catching the exception
- Use *timeout* parameter in *@Test* annotation

- Allow time for questions from participants

# Test Your Understanding


- How do you write a test that passes when an expected exception is thrown?
- Why does JUnit only report the first failure in a single test?
- How do you write a test that succeeds when a exception occurs and executes within a certain timeframe?

# About Test Annotation- Session 7: Summary

- JUnit first constructs a fresh instance of the class and then invokes the annotated method
- The *@Test* annotation supports two optional parameters
- Parameter *expected* declares that a test method should throw the right exception
- Parameter *timeout* causes a test to fail, if it takes longer than a specified amount of clock time (measured in milliseconds)

# About Test Annotation- Session 7 : Source

- Books:
  - JUnit Recipes: Practical Methods for Programmer Testing by *J. B. Rainsberger, Scott Stirling*
  - JUnit in Action by *Vincent Massol, Ted Husted*
- Web:
  - Wiki: http://en.wikipedia.org/wiki/JUnit
  - JUnit: http://www.junit.org/

Cognizant | Academy
Passion for making a difference

# You have completed the Session 7 About Test Annotation



Academy