



# Software 1



## Recitation No. 12: JUnit

June 3, 2006

# Software Testing

---

## ■ Motivation:

- Automatic formal proofs can be impossible
- Manual proofs are costly

## ■ Definition:

- Running the software on a set of input data

## ■ Goals:

- Finding faults
- Providing confidence of (probable) correctness and of satisfying all requirements

# Software Testing (cont.)

---

- Granularity Levels:
  - Unit Testing (developers)
  - Integration Testing (QA)
  - System testing
  - User Acceptance Testing
  - Regression Testing
- “Testing is not a phase, but a lifestyle”

# Unit Tests

---

Two Types:

- **Black-box tests:**

- assert the contract
- independence of the implementation

- **Coverage tests (Glass-box tests)**

- assert each piece of code

# JUnit

---

- A testing framework for Java programs
- Useful mainly for unit testing
- Integrated with Eclipse
- home-page: <http://www.junit.org>

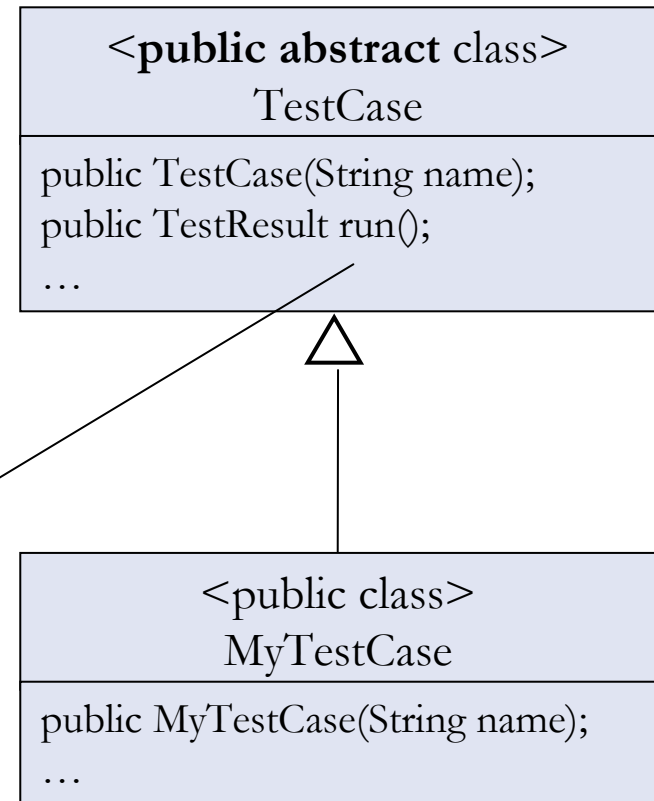


# JUnit Framework

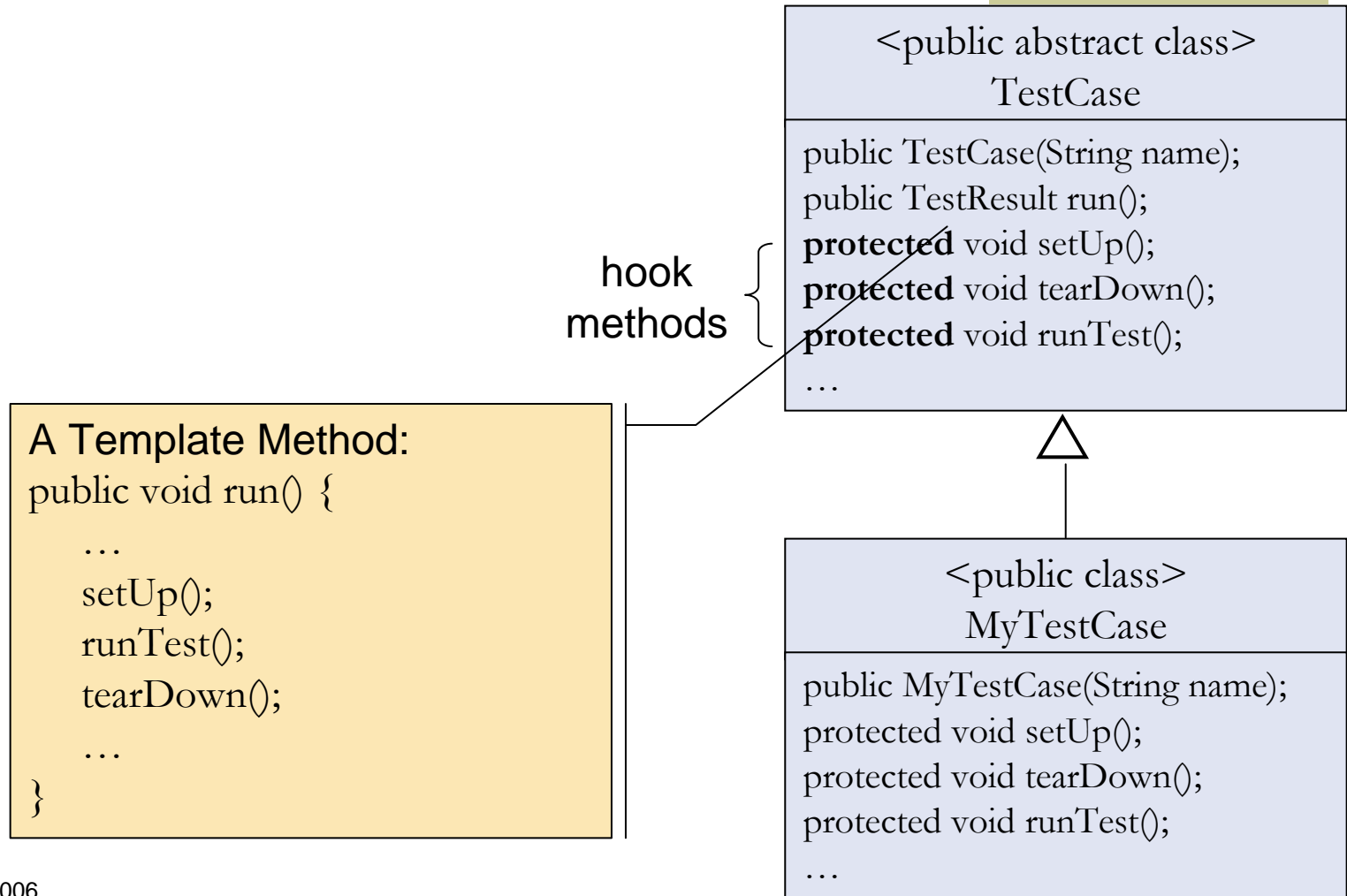
## The TestCase Class:

- The fixture to run tests
- A command object.
  - design pattern
  - encapsulates a command as an object

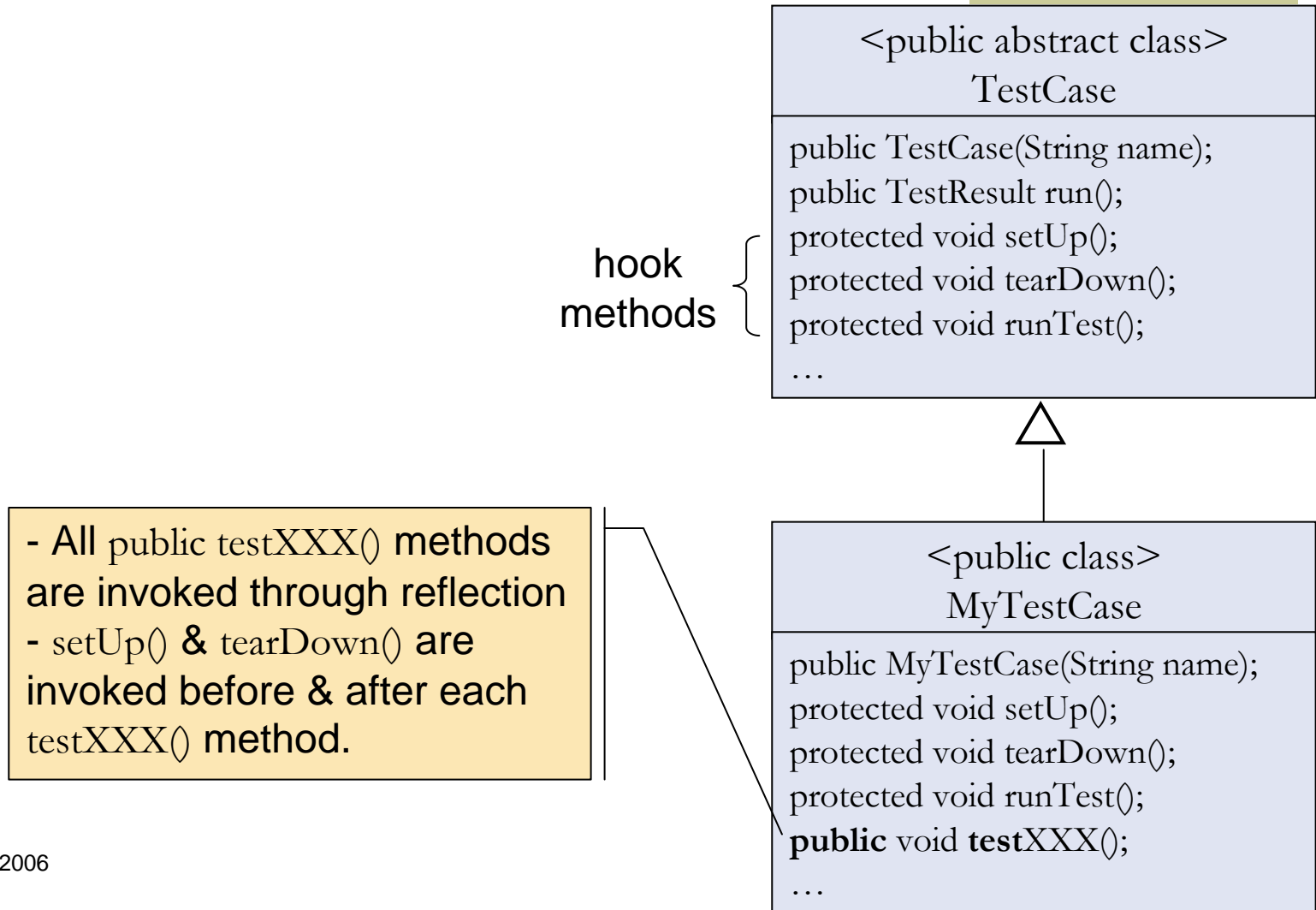
A Template Method  
(design pattern)



# JUnit Framework (cont.)



# JUnit Framework (cont.)

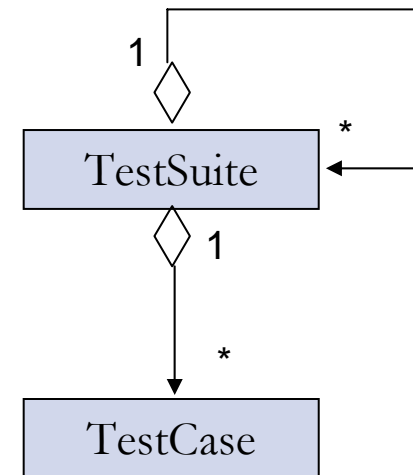




# JUnit Framework

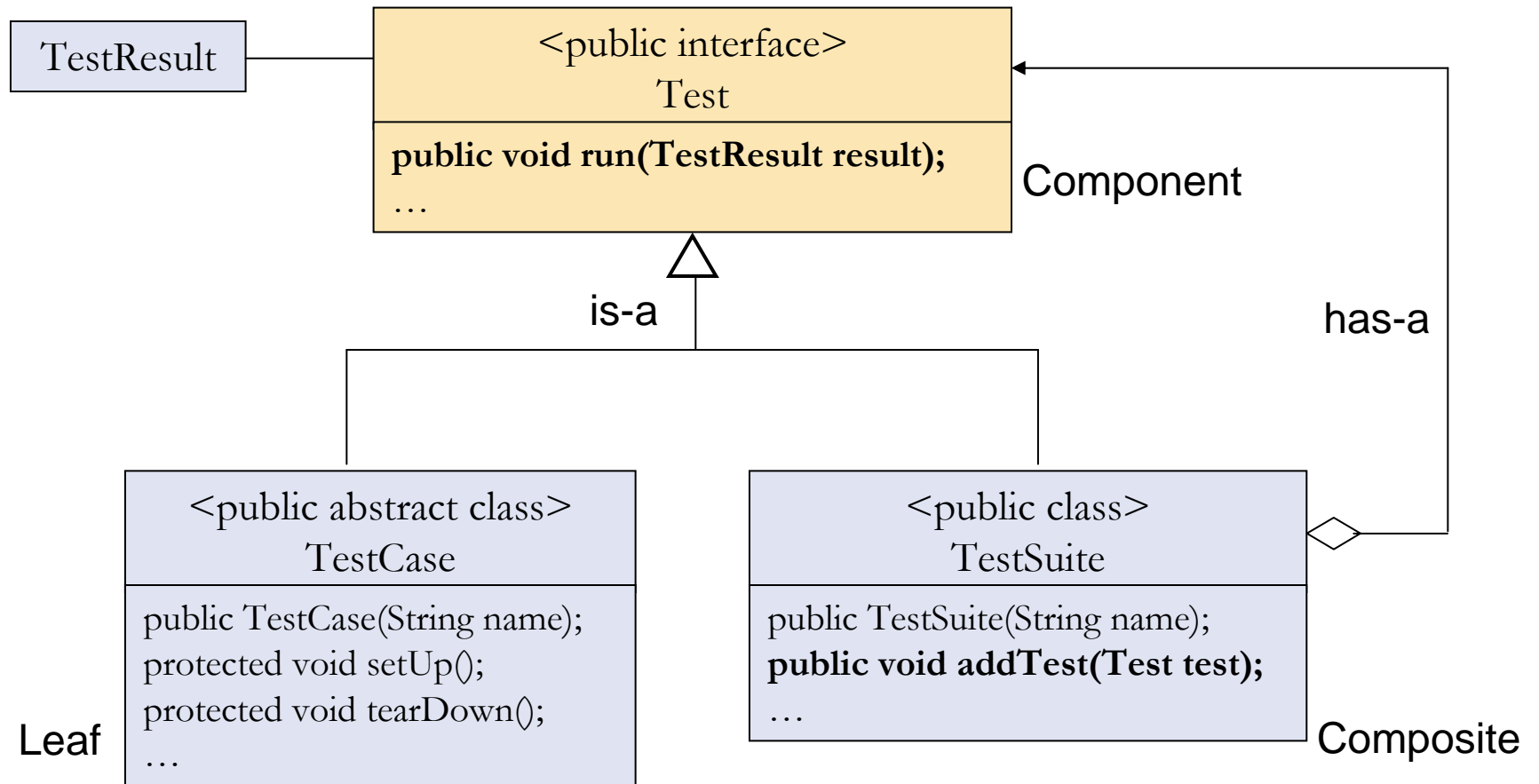
## The TestSuite Class:

- TestCases can be grouped into TestSuites
- Grouping is implemented using the Composite design pattern.
- A Composite is a group of objects in which some objects contains others



# JUnit Framework (cont.)

## ■ The Composite Design Pattern:

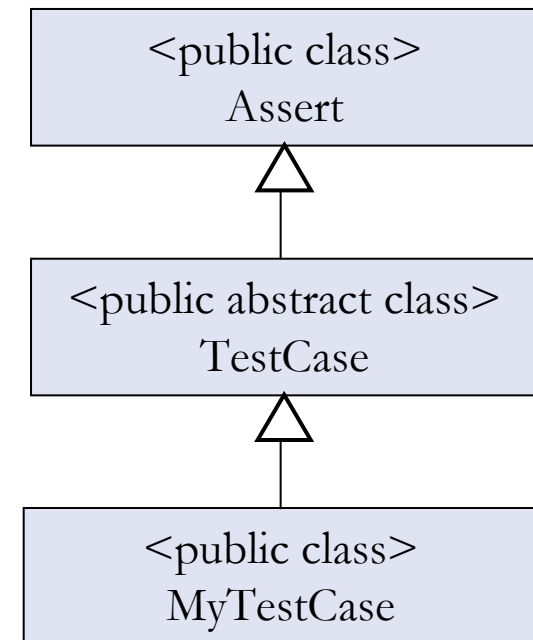


# JUnit Framework (cont.)

## The Assert Class:

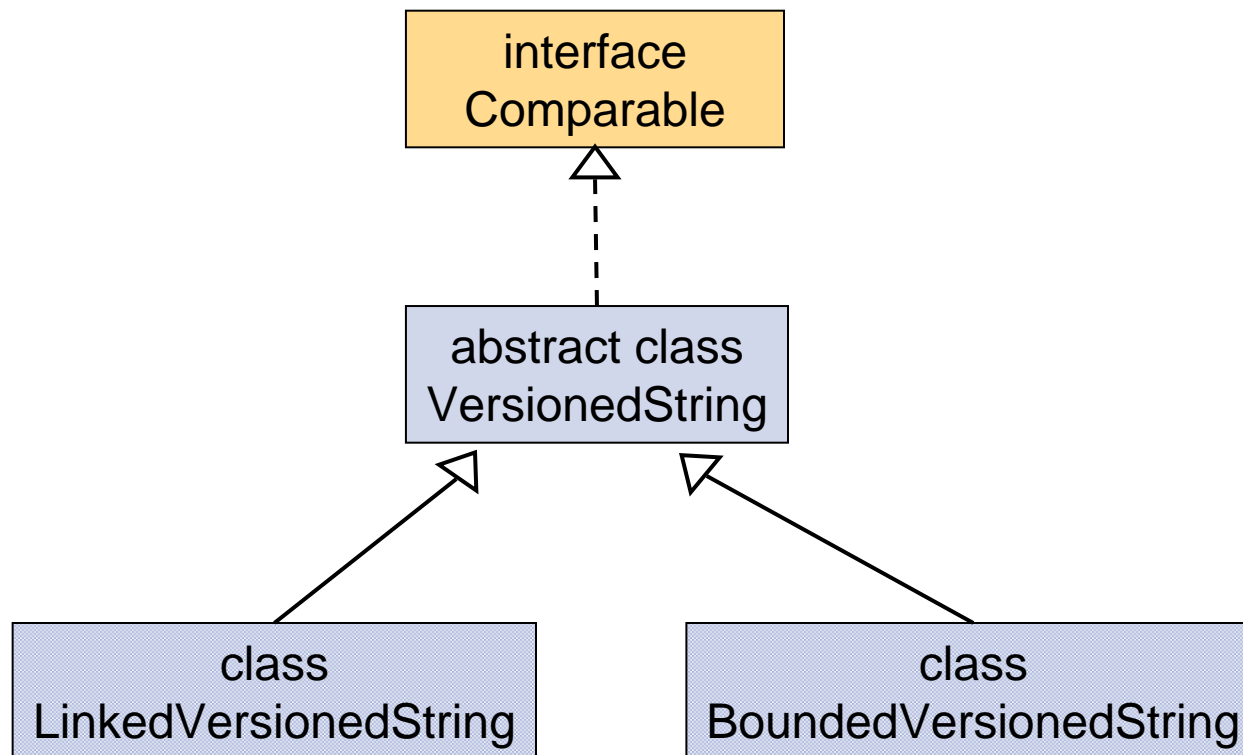
### ■ Defines a set of assert methods:

- `assertEquals(...)`
- `assertFalse(boolean condition);`
- `assertTrue(boolean condition)`
- `assertNotNull(Object object);`
- `fail();`
- more...



# JUnit with Eclipse

Demonstrating black-box tests for:



# JUnit with Eclipse (cont.)

---

Writing a test case for `LinkedVersionedString`:

- Define a class named `LinkedVersionedStringTest`:
  - Extend `junit.framework.TestCase`
  - Override the `setUp()` and `tearDown()` methods
  - Write `testXXX()` methods with no arguments
  - No need to write a `main()` method

# JUnit with Eclipse (cont.)

## ■ Black-box tests for `LinkedVersionedString`:

- *Initial State*: `length() == 0`

- `add(String s)`:

*Requires*:  $s \neq \text{null}$

*Ensures*:  $\text{length}() = \text{old length}() + 1$

$\text{getVersion}(\text{length}()) = s$

- `int length()`:

*Requires*: *nothing*

*Ensures*:  $\text{return\_value} = \text{Number of calls to add() so far}$

- `String getVersion(int i)`:

*Requires*:  $0 < i \leq \text{length}()$

*Ensures* :  $\text{return\_value} \neq \text{null}$

inputs:

$0 < i < \text{length}()$

$0 < i = \text{length}()$