Testing with JUnit

Running a test case:

1. Get the component to a known state (set up).
2. Cause some event (the test case).
3. Check the behaviour.
    - Record pass/fail
    - Track statistics

- Typically we want to do a lot of test cases so it makes sense to automate.
- Test cases are mostly similar in structure, so we can generalize them.

JUnit

- JUnit is a framework for writing repeatable tests.
- classes for structuring test cases.
- runners to run test cases and collect statistics.
  - `junit.textui.TestRunner` – Text based
  - `junit.swingui.TestRunner` – Swing based
  - `junit.awtui.TestRunner` – AWT based
- Normally used by extending `TestCase` with specifics for testing a particular class/system/component.

Test Fixture

- Usually some set of test cases operate on a similar set of objects — the *test fixture*.
- Fixture is implemented by member variables of extension (subclass) of `junit.framework.TestCase`.
- Override two methods:
  - `protected void setUp()` — initialize fixture prior to each test case.
  - `protected void tearDown()` — clean up fixture after each test case.

Test Cases

- By default, methods named `test`*Something* are test cases.
- Write one method `test`... for each test case.
- Use `assert`*XXX* from `junit.framework.Assert` (a parent of `TestCase`) to evaluate results.
    - `assertEquals`
    - `assertTrue`
    - `assertFalse`
    - `assertSame`
    - `assertNotSame`
    - `fail` — for when you know a test has failed.

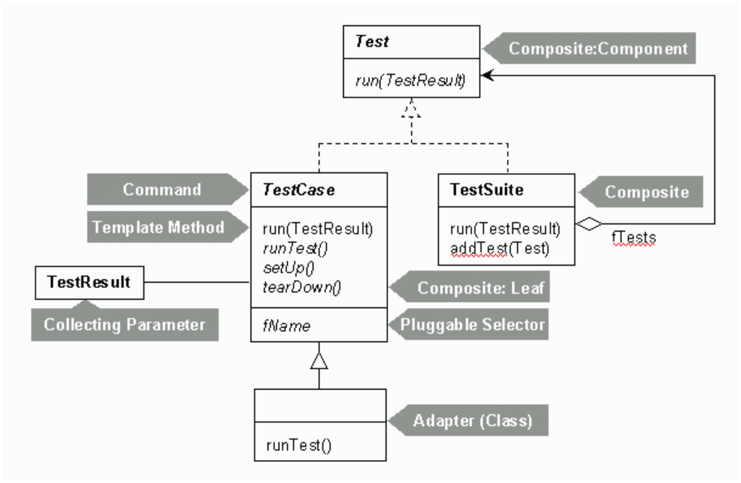
Test Suite

- To run a group of tests together, create a test suite.
- Simplest to simply implement:

```
public static Test suite() {
  return new TestSuite(YourTestClass.class);
}
```

- Will form test suite containing all methods that start with "test".
- Can also use no-argument constructor and explicitly add tests with `addTest`.

Design of JUnit



References

There's lots of info at `http://www.junit.org`, including:

- Kent Beck and Erich Gamma.
  JUnit A Cook's Tour, 2004.
- Kent Beck and Erich Gamma.
  JUnit Cookbook, 2004.