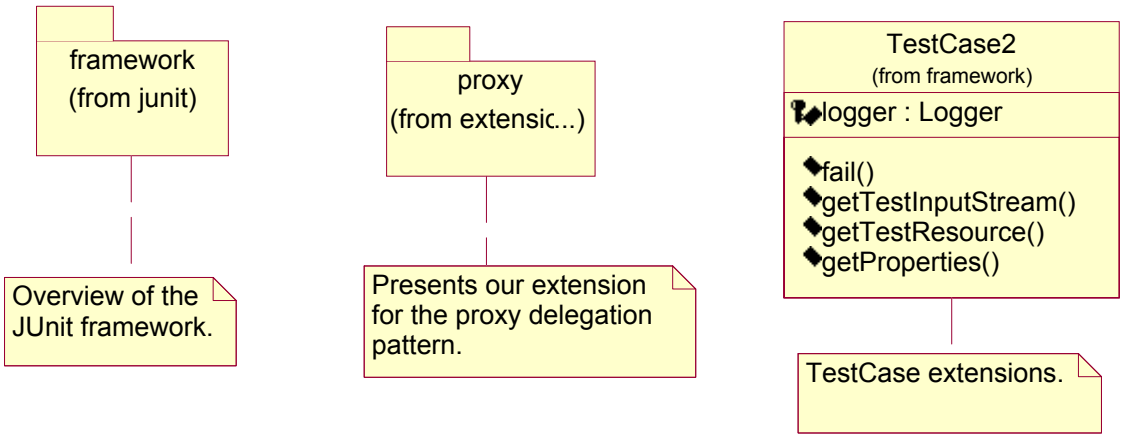
This module provides some extensions to the JUnit framework, including:

- Support for a proxy delegation pattern. This is useful when creating a test suite for an API that will have multiple implementations.

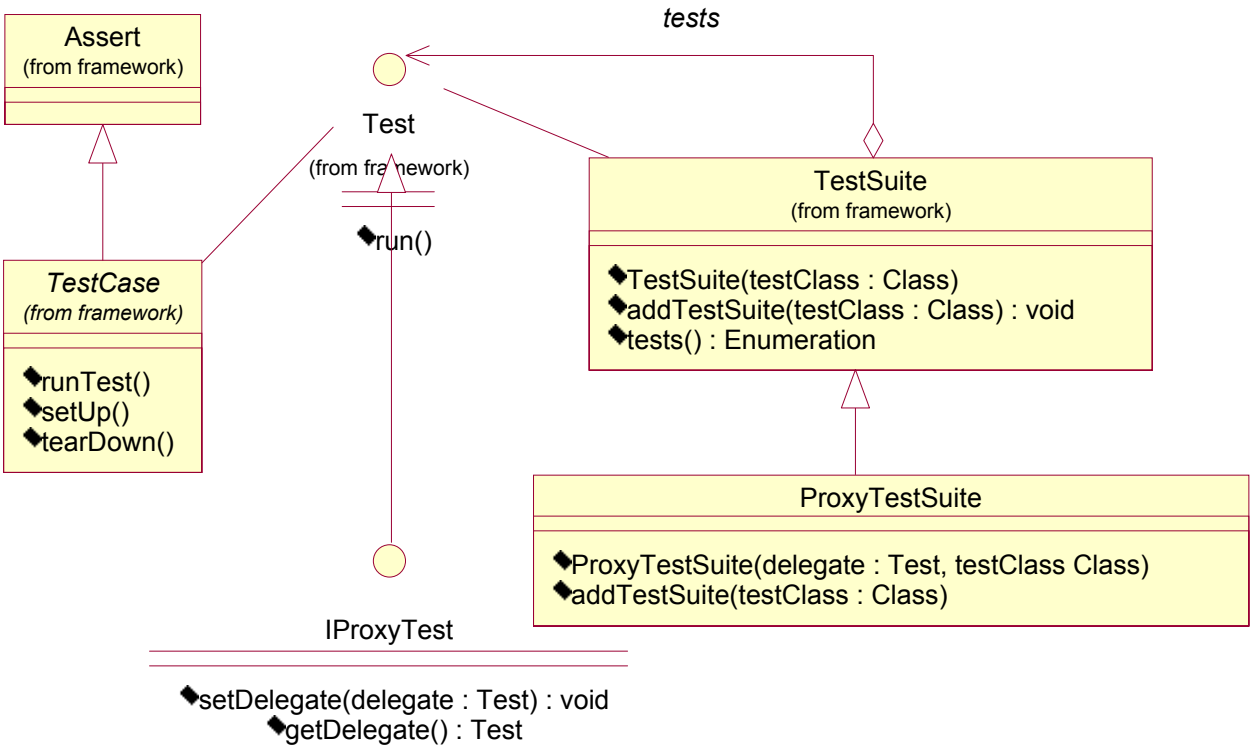
- Support for hierarchical properties.

- Support for loading test resources.

- Various trivial convenience methods.



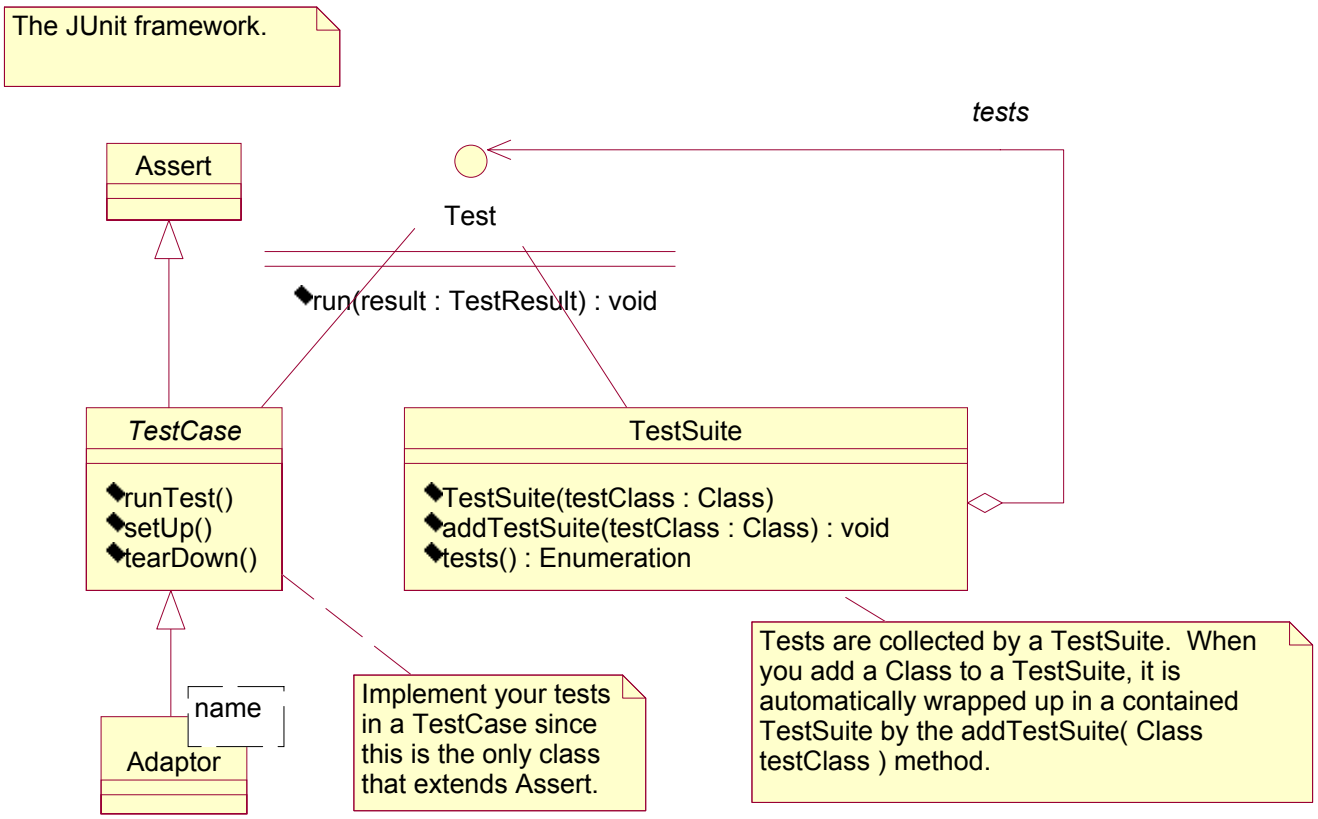
Proxy and delegation pattern.



The use of the proxy and delegation pattern with JUnit is appropriate when you want to factor apart a test suite for some API and you have multiple implementations of that API with the consequence that implementation specific fixtures must be established for testing. Under these conditions you have a multiple inheritance problem since the API test suite needs to extend the implementation specific test cases, but the API test suites must be independent of the specific implementations.

The JUnit framework makes stateless invocations of tests. In order to solve the multiple inheritence problem, you need to be able to provide one item of state to each of the tests in the API test suite -- a delegate Test. Normally the delegate Test will expose some abstract behavior and the implementation specific TestCase will provide a concrete implementation of that behavior. The tests in the API test suite will implement the IProxyTest interface so that they can gain access to that delegate. The implementation specific test case extends TestCase.

The use of the proxy and delegation pattern makes it easy to repeat a test suite for different configurations as well. The delegate TestCase typically implements a method that reads Properties that are used to establish the configuration.


The JUnit framework.
tests
Assert
Test
run(result : TestResult) : void
TestCase
runTest()
setUp()
tearDown()
TestSuite
TestSuite(testClass : Class)
addTestSuite(testClass : Class) : void
tests() : Enumeration
name
Adaptor
Implement your tests in a TestCase since this is the only class that extends Assert.
Tests are collected by a TestSuite. When you add a Class to a TestSuite, it is automatically wrapped up in a contained TestSuite by the addTestSuite(Class testClass) method.

Some extension methods.