SUnit: latest developments

Sunit Open-Source project

Niall Ross, Cincom
Jan Vrany, eXept

Debugging Suites that use Resources

- Run suite with resources: see failure
  - Debug the failing test: see it pass
  - Debug all tests in suite: see them all pass

What's happening?

- earlier test affected resource-managed state, caused failure
  - e.g. earlier test created data in a table the resource creates in database
- TestCase>>debug tears down resources for each test
  - TestResource's table torn down and recreated – interfering data is gone

- The Fix: new method TestResource>debug
  - a polymorph of TestCase>>debug

TestSuite, TestResult pluggable in base

- Prior ESUG: pluggable in your test runner tools
  - Stephane said “Put it in the base.”
  - Niall said: “Shape-change in base is a big deal …”
- … now: pluggable in the base
- TestCase>>suiteClass: myTestSuiteSubclass

```
TestCase>>suiteClass
    ^SuiteClass ifNil: [TestSuite]
```

- PluggableTestSuite = pluggable TestResult subclasses

  RandomSuite, MinimalResConflictSuite, … ClassifiedTestResult, …

Other Stuff

- TestCase/Suite>>run/run:
  - API methods have better comments

    #run tears down resources

    #run: does not
- TestCase>>asNew
  - override in (very rare) case that your tests have key instvars
- Deprecated methods: (anyone use them?)

  #debugAsFailure, #openDebuggerOnFailingTestMethod, #runCaseAsFailure:
- TestCase>>= and >>hash in base SUnit

Slim SUnit v. Fat SUnit?

- slim base SUnit
  - Load base SUnit for
    - TestCase, TestSuite, TestResult , TestResource (and TestAsserter)
  - Load SUnit<Specific> for
    - Examples of Pluggable TestSuite and TestResult subclasses
    - Examples of other TestCaseOutcomes, e.g. TestSkip
      - assimilated to #expectedFailures
    - Other SUnit OS-project stuff
- fat base SUnit
  - Load base SUnit to get much more (than you want)?