Austin JUG
GUI testing with Abbot

Enough with testing the model already!!!

Michael Forsberg

- RealVue Simulation Technologies
  - Software Developer
- Austin JUG
  - certification group coordinator
- Java expierence
  - 2.5 years writing GUI Applications
- BS Computer Science (UT Austin)

Test driven development

- What you get...
  - code that works
  - fast feedback on code changes
  - tests verifing design requirements
  - numerous others
- Traditionally done with unit test.
  - **Unit Tests** are programs written to run in batches and test classes. Each typically sends a class a fixed message and verifies it returns the predicted answer.

Current problems with unit testing applications.

- Unit testing tools are excellent at testing models
  - data structures and APIs
  - fundamental classes and their interactions
- Many developers relys on QA individuals for...
  - Validating GUI components are visually in sync with user interactions.
  - Regresion testing of known interaction bugs
  - Double checking required functionality
- **Abbot provides a solution**

Why a reliance on “model only testing” is bad.

- “Model Only Testing”
  - tests for API components
  - ignoring tests resulting from strange end user interaction.
- Strange states are generally possible.
  - Think, gone to get coffee.

Unit tests promote “Model Only Testing”

- Some thoughts that lead to model only testing...
  - I'll test my component well.
  - I need to be sure this funkey state machine works.
  - Come on, no user with half a brain would do that.

Testing GUI Components.

How in the heck do I test this JTable???

Goals of Abbot

- Scripted control of actions and inspection
- Loose component bindings
- Specify high-level semantic actions, but use low-level OS events to implement them.
- Support live recording of high-level semantic events.
- Extensible user action recording and generation

Main sections of Abbot

- GUI Test Fixtures
- The Robot (A better java.awt.Robot)
  - Testers (For most gui components)
  - Conditional waits (For frames and the such.)
- Component References
  - ComponentFinders, WindowTrackers, Matchers
- Scripts
  - Costello, Recorders

How I use abbot...

- Costello for more QA related tasks.
- Add the Abbot jars to your project and JUnit away.

Costello

- Will be shown during the examples at the end of the presentation

JUnit and Abbot

- Extend ComponentTestFixture
  - Which extends ResolverFixture
    - Which extends TestCase
- ResolverFixture provides ways to find objects
- ComponentFixture provides a robot to interact with the gui components.
- Perform tests as normal, using Finders and Robots to perform GUI interactions

Basic interaction

```
this.getRobot().click(this.getFinder().find
  (customizerModalDialog,
    new Matcher() {
       public boolean matches(
             Component c) {
          if (!(c instanceof JButton))
             return false;
          JButton tmp = (Jbutton)c;
          if (tmp.getText().equals("OK"))
             return true;
          return false;
       }
    }));
```

- Questions (while loading samples)

Thank You and References

- Abbot's Website
  - http://abbot.sourceforge.net/
- UnitTest - definition
  - http://c2.com/cgi/wiki?UnitTests
- FAQ – TestDriven.com
  - http://www.testdriven.com/modules/xoopsfaq/index.php?cat_id=5#q35