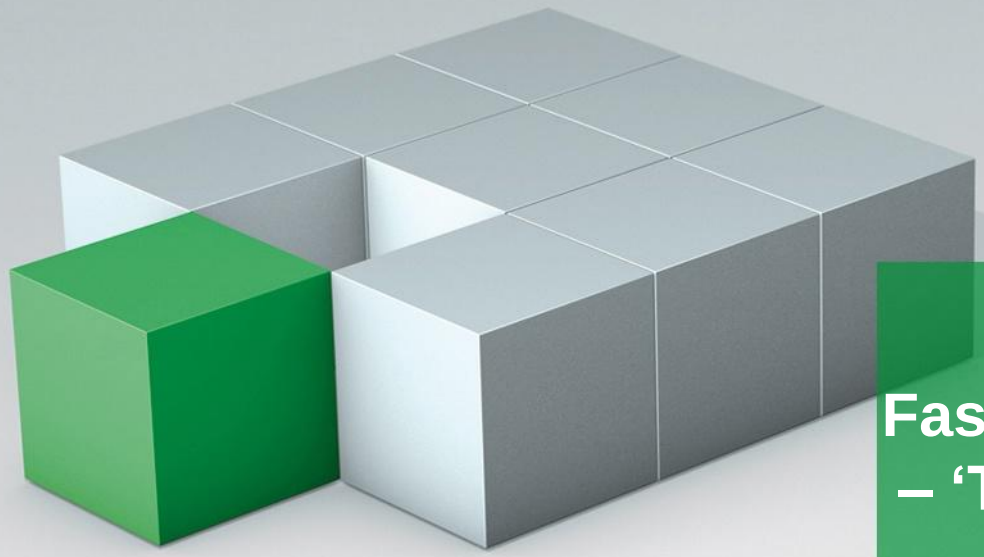
Quality Services and
Testing Solutions at Work

SQS
SOFTWARE QUALITY SYSTEMS



Fast-Track TDD – ‘Testify’ your project

Mike Scott

Agile Testing Days, Berlin October 2009

SQS Group Limited

What's this about?

- Outline an 'ideal' Test-Driven Development environment
  - What are the technical requirements for doing TDD properly?
- Discuss an idea for 'bootstrapping' a TDD process
  - How do we overcome the TDD 'setup risk' for a new project?
- Demo a neat utility called 'Testify' that does this
  - Extended from open source 'TreeSurgeon' wizard
  - Inspired by TDD work in SQS with David Evans

Preface: Agile's Three Levels

- **Values (Organisation level)**
  - People & Interactions, Collaboration
  - Courage, Simplicity, Feedback, Communication
- **Principles (Project level)**
  - Embrace Change, Travel Light, Incremental Change, Working Software, Eliminate Waste, Reduce Cycle Time
- **Practices (Development & Testing level)**
  - **Test-Driven, Continuous Quality, Automation**, Pairing

(Mike's) Principles of Test-Driven Development

- **Any code worth writing, is worth testing**
  - "Code that isn't tested doesn't work – this seems to be the safe assumption" (Kent Beck)
- **Any test worth writing is worth running all the time**
  - Invest time in automating the tests, as execution is fast and free
- **Design the feature by designing the tests first**
  - How should this work, and how will I know when it does work?
- **Fix Regression First**
  - If a previously passing test now fails, fixing it is the highest priority

What are the technical requirements for TDD?

- **Project Automation**
  - Automatically build from latest source code (Continuous Integration)
  - Automatically run one or more test suites (Continuous Quality)
- **Unit Test Framework**
  - Automated white-box tests written by developers
- **Acceptance Test Framework**
  - Automated black-box tests written by testers / analysts / customer
- **Other Automated Tests**
  - GUI tests, non-functional tests, static analysis...
- **Code Coverage Analysis**
  - How much of the code base is being tested by our suites?

Background: the 'Balloon' pattern

- **Start Valid, but Empty**
  - An empty balloon is still a balloon
  - Complete the form or structure of a solution, before adding function
- **Rubber before Air**
  - Don't deliver functionality that cannot be tested
  - Delaying testing is just incurring quality debt
- **Use Iteration Zero**
  - Produce installable / deployable software
  - 100% of automated tests pass, with high test coverage
  - Functionality is trivial (at best)

Introducing 'Testify'

- **Agile TDD Toolset installer and project generator**
  - Developed by Mike Scott of SQS
- **Wizard for creating a new, portable Project**
  - Built on 'Tree Surgeon' for creating portable .NET solution structures
- **Requires only a Project Name**
  - Name becomes an Application Property
  - This is the basis of the first 'acceptance test'
- **Installs Open Source Toolset**
  - Stack of 'best of freed' tools needed for TDD
- **Creates C#.NET code project**
  - Trivial Core : Application.Name = 'YourName'
  - Trivial GUI : 'About' box
  - Trivial Unit & Acceptance Tests
  - Batch file to build project and run all tests in the suite

Demonstration

- **Testify Demo #1**
  - Using the Testify Wizard to:
    - create a new c# project tree
    - Install open-source agile TDD toolset
    - Create example tests

What is in the 'Testify' open source tool stack?

Dot Net Projects

Purpose	Tools	Examples created by Testify
Project Source	(none)	VS2005/VS2008 C# solution
Project Automation	NAnt	Tasks for Clean, Build, Test, Dist...
Unit Test Framework	NUnit	Initial unit tests, SQS helper library of c.30 tests
Acceptance Test Framework	FIT	1 FIT test (application name), 1 Fixture class
Story & Acceptance Test Authoring	Fitnesse, Richnesse	Simple project page hierarchy, 1 acceptance test
Web GUI Test Framework	Selenium	(None yet)
Code Coverage Analysis	NCover	Coverage report generated
Installer	NSIS	Sample .nsi file
Static Code Analysis	FX-Cop	Standard Rules checked

Demonstration

- Testify Demo #2
  - Build and test the application
- Testify Demo #3
  - Skeletons & Templates

What is in the 'Testify' open source tool stack?

Java Projects

Purpose	Tools	Examples created by Testify
Project Source	(none)	Java source
Project Automation	Ant	Tasks for Clean, Build, Test, Dist...
Unit Test Framework	JUnit	Initial unit tests,
Acceptance Test Framework	FIT	1 FIT test (application name), 1 Fixture class
Story & Acceptance Test Authoring	Fitnesse, Richnesse	Simple project page hierarchy, 1 acceptance test
Web GUI Test Framework	Selenium	(None yet)
Code Coverage Analysis	Cobertura	Coverage report generated
Installer	-	
Static Code Analysis	Checkstyle	Standard Rules checked

Where to from here?

- **I like it, how can I try it?**
  - Got a USB drive? See me after…
  - Email Mike.Scott@sqs-uk.com
- **Is it as simple as that to get going with Test-Driven Development?**
  - Well, it depends…
  - Testify is only intended to solve the 'tools and technology' issues
  - SQS have TDD 'Kick-Start' services that address the 'people and process' side of things – talk to us for details
- **The future of Testify: New features in progress**
  - Persistence layer, Service Layer, Nhibernate, Castle, Web front-end
  - 'Mockable clock' with time-based test examples
- **Open Source**
  - Yes, once a coherent design is bedded i
  - Look for notification on www.AgileTesting.org.uk

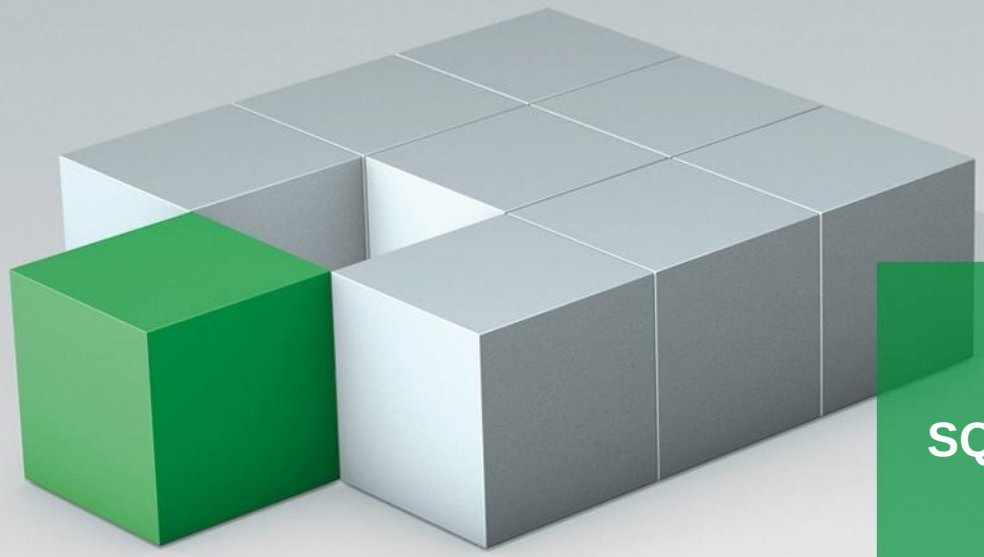
Breaking News:

Testify is now O/S at

http://code.google.com/p/testifywizard/

Quality Services and
Testing Solutions at Work

SOFTWARE QUALITY SYSTEMS



SQS Group Limited

7-11 Moorgate | London, EC2R 6AF, United Kingdom
Tel.: +44 (0) 20 7448 4620 | Fax: +44 (0) 20 7448 4651
E-Mail: info@sqs-uk.com
Internet: www.sqs-uk.com | www.sqs-group.com

Thank you for your attention