Specialists and other myths

CAST 2007
July 9-11, 2007
Bellevue WA

Michael Kelly
www.MichaelDKelly.com

Testing Techniques; Innovations and Applications

Why are we so bad at this?

"I can't do..."

- automated testing
- performance testing
- exploratory testing
- usability testing
- security testing

Why not?

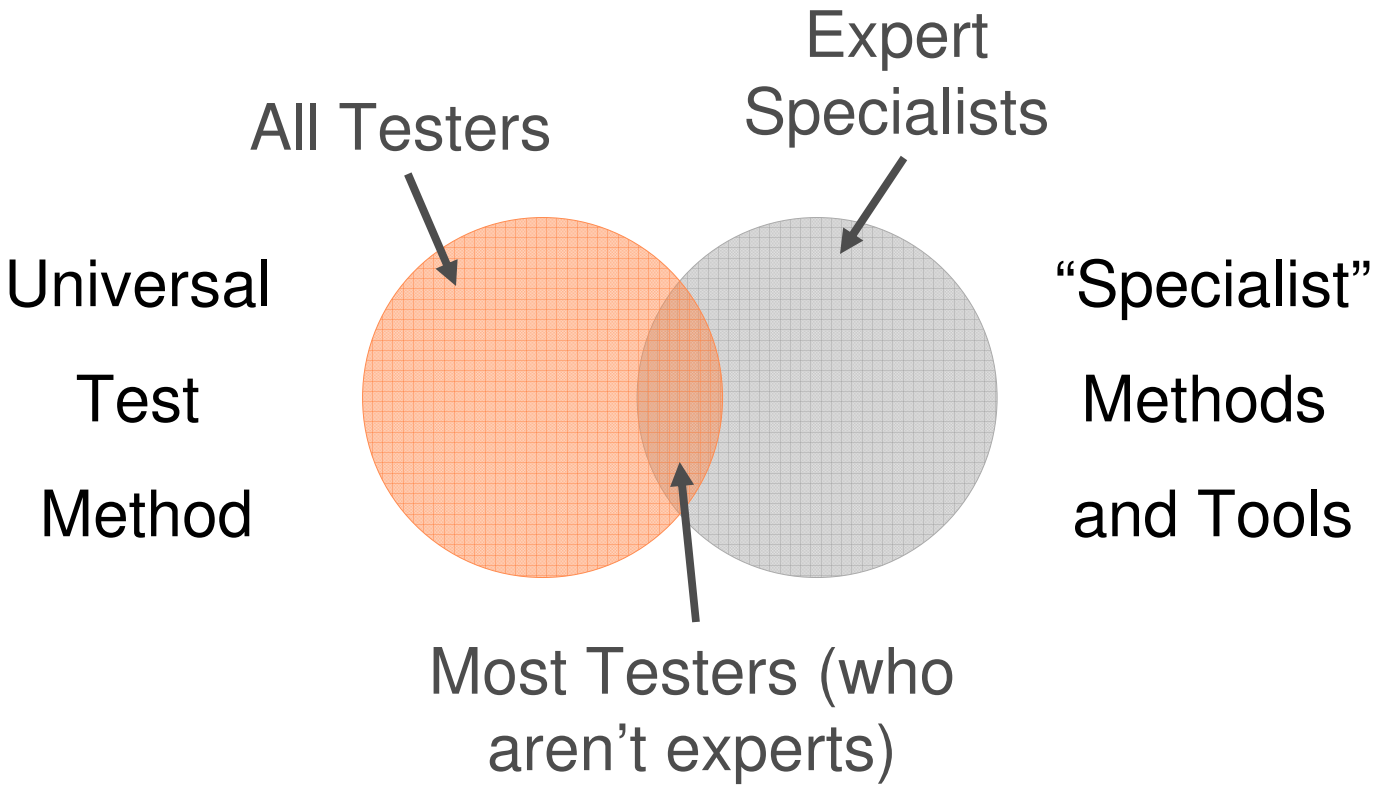
One day…

The Universal Testing Method

1. Model the test space
2. Determine coverage
3. Determine oracles
4. Determine test procedures
5. Configure the test system
6. Operate the test system
7. Observe the test system
8. Evaluate the test results
9. Report test results

Bach, James and Michael Bolton. Rapid Software Testing version 2.1.2. Satisfice, Inc. 1995-2007.

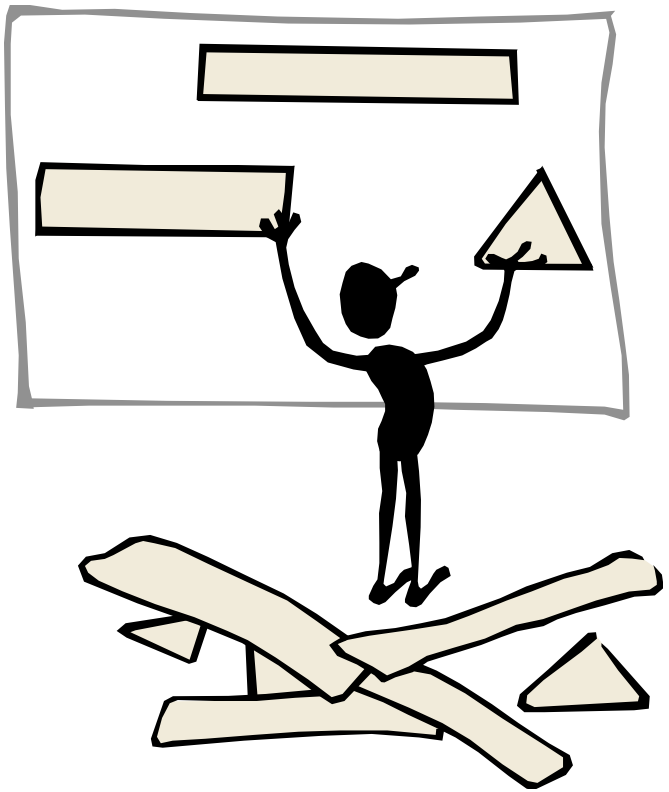
Who are the experts?



Model the test space

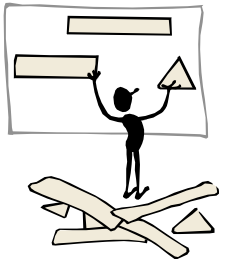
When you model the test space, you are:

- composing, describing, and working with mental models of the things you are exploring
- identifying the relevant dimensions, variables, and dynamics of the application



Formal vs. Informal models

Model the test space



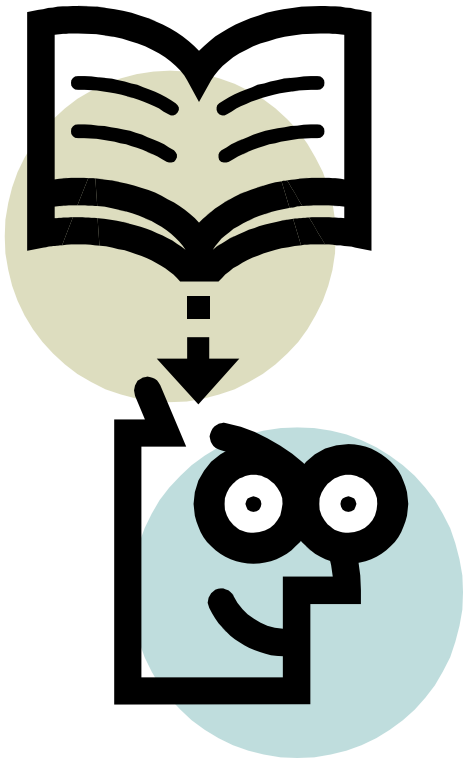
Examples:

- Try developing a UCML diagram for a large performance test
- Use FCCCUTSVIDS for application touring while exploring
- Find or develop a state model of the application for automation
- Try to find comparable products for usability testing
- Use analysis tools such as the SPI Dynamics suite of tools to outline the application for security testing

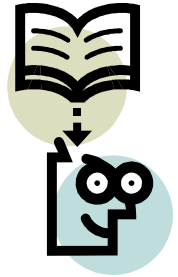
Determine oracles

An oracle is the mechanism for determining whether the application has passed or failed a test.

- Some oracles are specified in advance or are made formal
- Some oracles are heuristic



Determine oracles



Examples:

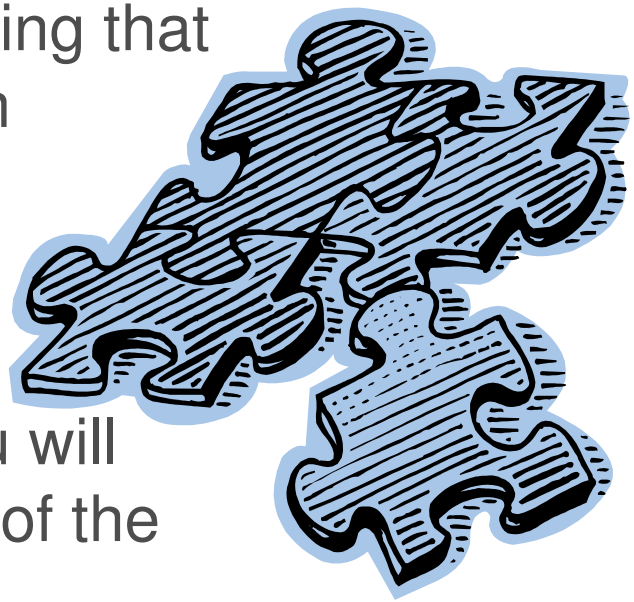
- Identify or specify performance SLAs for key features within the application
- Use the HICCUPP consistency heuristics for both functional and usability testing while exploring
- Identify another application for use in a parallel test for either manual exploratory testing or for test automation
- Identify general error patters or verification point data sources for your automation
- Look for known security exploits in the application or technology you're testing

Determine coverage

Test coverage is what you're testing. It comprises the dimensions of the product evaluated while testing.

Things to think about:

- Risk: *why* you are covering that aspect of the application
- Technique: *how* you will cover that aspect of the application
- Environment: *where* you will be covering that aspect of the application



Determine coverage

Examples:

- A great heuristic for starting a coverage outline for both manual and automated testing is James Bach's "San Francisco Depot" (SFDPO) mnemonic
- For performance testing, I'll again default to UCML; try adding specifics about your data
- Identify groups of users, personas, or profiles for usability testing
- Try developing simple threat models or explore creating misuse cases for application security

What really makes a specialist special?

I like to think it's their advanced understanding the methods and tools of the type of testing in which they've specialized.

Determine test procedures

A test procedure is simply the instructions for setting up a test, how to execute the test, and for how to evaluate the results.

Might address:

- Pre-conditions and post-conditions
- Testing environments
- Testing tools
- Timing

Even if you can't write them:

- Review and provide feedback
- Research specific aspects
- Compare and contrast with the other testing you are doing on the project

Formal vs. Informal

Determine test procedures

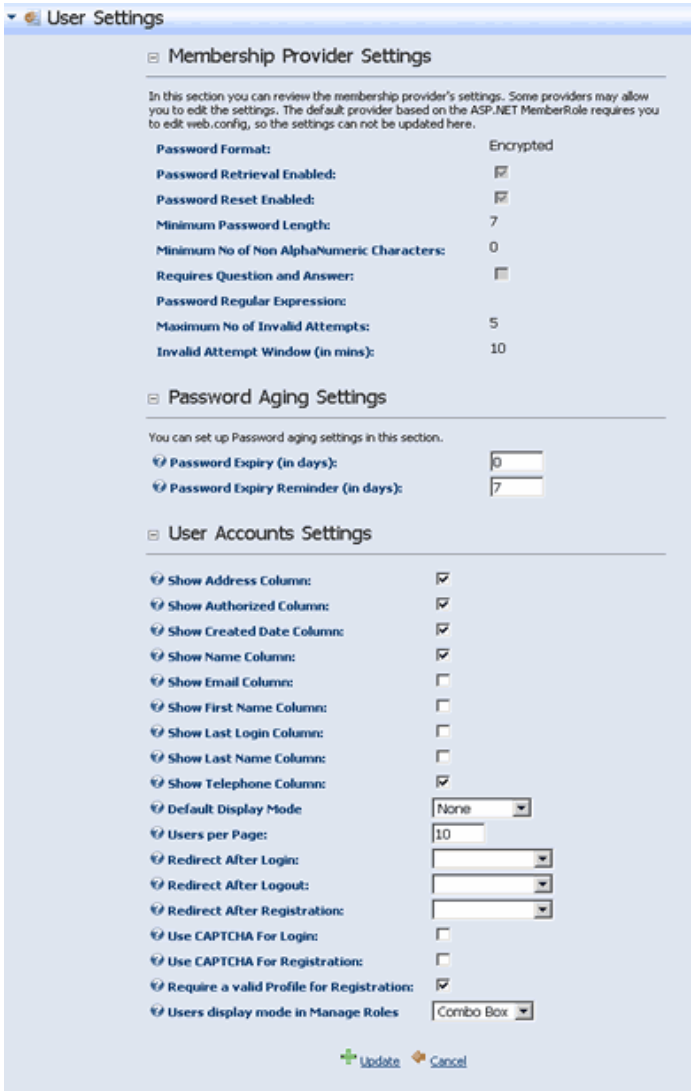
Examples:

- The process of logging test results using a central test management tool like IBM Rational TestManager or HP Mercury TestDirector
- The process of managing and reporting results in session based test management
- Standards for where to place transaction timers, verification points, and naming conventions performance and automated scripts
- Process for gathering information on user interaction with the system during a formal usability testing session (video recording, desktop recording, interviews and surveys, etc…)
- Who to notify within the organization before you start your security testing

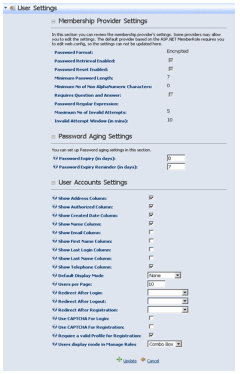
Configure the test system

Configuring the test system is where you ensure you have everything you need to support your testing. This can include installing or configuring any of the following:

- the application you are testing (proxies, servers, configuration settings, logs, etc…)
- applications for parallel testing (or other oracles)
- testing tools and scripting languages
- video or desktop recording tools
- setting up the *your* environment (notepads, monitors, input methods, etc…)



Configure the test system



Examples:

- Install necessary tools, applications, and languages needed to support your exploratory testing
- Set up an performance monitoring dashboard using Introscope or WhatsUp
- Set up customer analytics software to capture usage during usability testing or beta testing
- Use virtual environments to execute your automated scripts on many different platforms and configurations
- Configure security scanners and analyzers for your environment

Operate the test system

Operating the test system is where you make and manage your contact with the object of your study. You use the various configurations you've outlined, you follow the procedures you've established for better control of experimental conditions.

- OFAT vs. MFAT
- Using lab procedures
- Using tools to extend your reach into the application you're testing
- Automated vs. manual operation

Operate the test system

Examples:

- Executing automated scripts
- Executing performance scenarios
- Scanning the system or server for security exploits

Brain Off

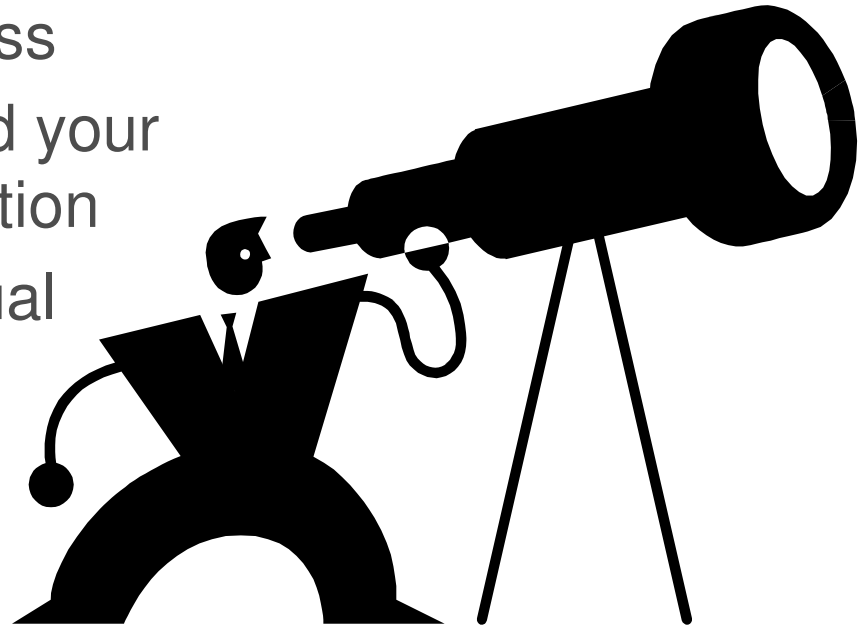
- Executing an exploratory test session
- Holding a formal usability test

Brain On

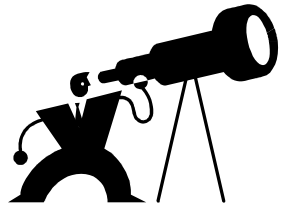
Observe the test system

Gathering empirical data about the object of your study; collecting different kinds of data, or data about different aspects of the object; using procedures for rigorous observations.

- Inattentional blindness
- Using tools to extend your view into the application
- Automated vs. manual observations



Observe the test system



Examples:

- Using IBM PureCoverage to capture code coverage information while executing automated regression suites
- Looking at virtual user histograms and system monitors while performance tests are running to better anticipate where additional monitoring might be helpful
- Evaluate hidden fields, URLs, session variables, or source code in a browser while performing web security testing
- Using BB TestAssistant to capture tester interaction with the system during exploratory testing
- Noticing a user's (or even another tester's) facial expressions while they are testing, listening to their voice for evidence of frustration

Evaluate the test results

Taking your observations and making sense of them. Not only recording the results, but determining if they are in fact the correct results, or the complete set of results needed to fulfill the mission of your testing.

It's determining what story your testing is telling you up to this point.

Evaluate the test results

Examples:

- Correlate virtual user load with response times, CPU and memory usage, and disk and network I/O
- Determine if a failed automated test case is an application failure, a script failure, or an intermittent problem
- Debrief another tester to hear the story of their testing so far so you can make a qualitative analysis
- Compare time on task, accuracy, recall, and emotional response to baseline data for a user population
- Evaluate likelihood of an exploit after a vulnerability is identified

Report test results

Make a credible, professional report of your work to your clients in oral and written form.

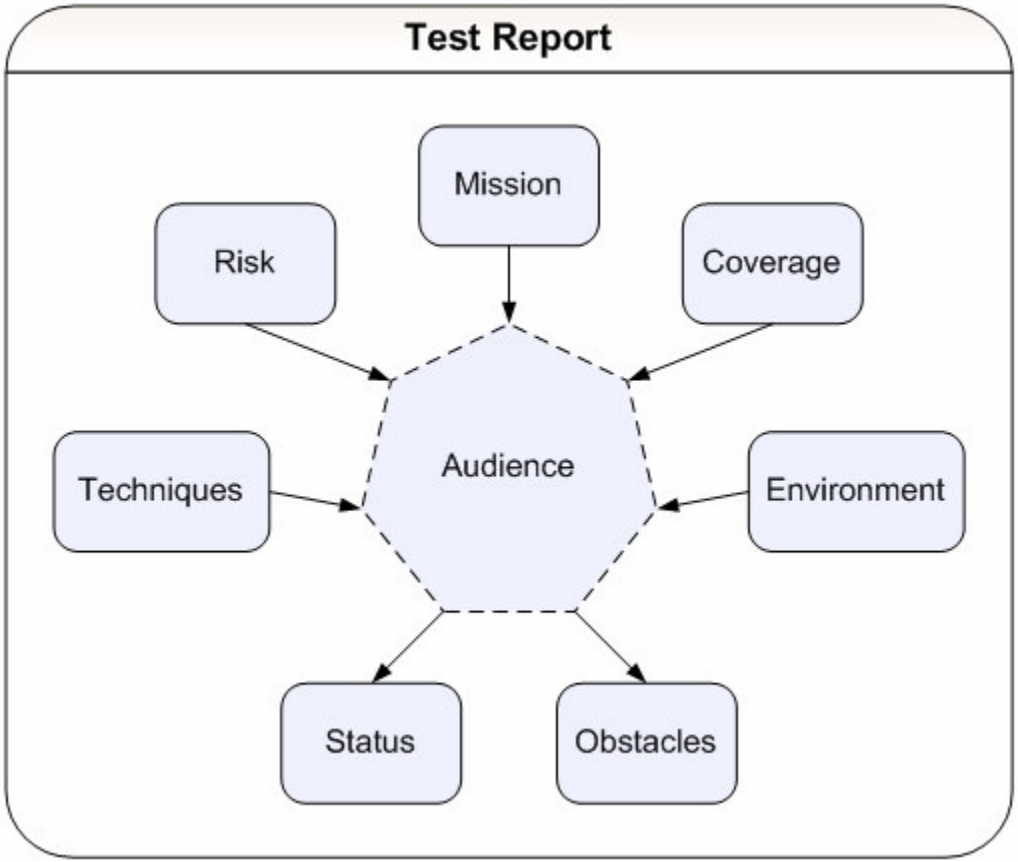
- Telling the story of your progress so far
- Writing a test summary report
- The hallway conversation status report

Report test results

Example:

- I use MCOASTER to frame my reports (for all types of testing)

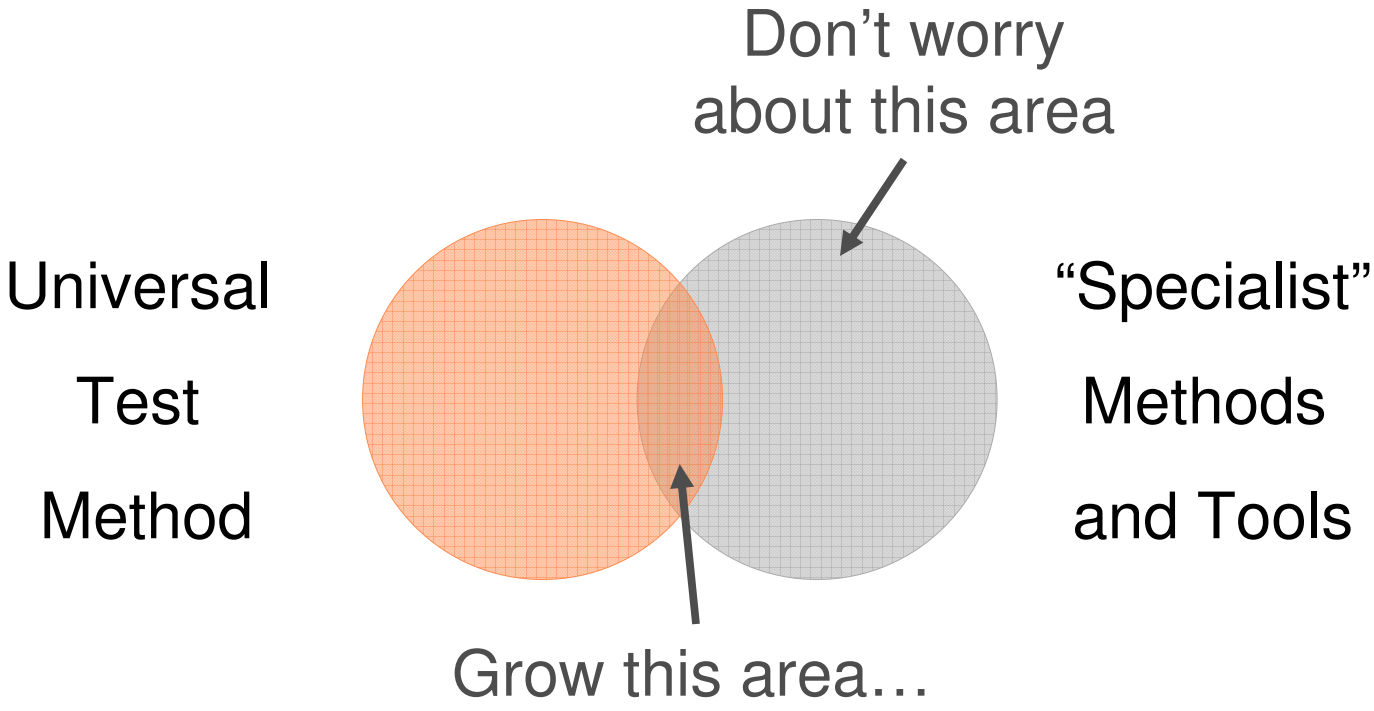
- Works at multiple levels:
  - Executive summary
  - Test summary
  - Test detail



What can you do?

If the only thing special are the methods and tools, then focus on those…

Focus on where there is overlap



Learn about them

Self-Education for Testers…

- ***Touring***: *I read a survey piece.*
- ***Experiencing***: *I build an example; or do the activity.*
  - ***Serendipity***: *I learn from unexpected events.*
  - ***Teacher***: *I go see someone.*
  - ***Reading***: *I find famous books and papers.*
  - ***Global Supermind***: *I tour Google.*
  - ***Standards***: *I discover what is considered "correct."*
  - ***Communities***: *I find a forum or professional association.*
  - ***Conferences/Classes***: *I attend with a critical attitude.*
  - ***Browsing***: *I skim and riffle.*
  - ***Acquisition***: *I gather a library.*
  - ***Testing***: *I contrast alternatives, critique, or consider extremes.*
  - ***Teaching***: *I try to explain it.*

Bach, James. "Self-Education for Testers." Satisfice, Inc. 2006. <http://www.associationforsoftwaretesting.org/conference/cast2006/James_Bach_Self-Education_for_Testers.pdf>.

Practice them

- Contributing to open source projects
- Beta testing
- Pair testing/programming
- Adopt “parallel thinking”
- Search for bugs in the wild
- Learn systems thinking
- Teaching and writing
- Participate in conferences and workshops

Kelly, Michael. “How Do You Practice Software Testing?” InformIT.com 12 Aug 2005.

Questions / Comments / Counter Experiences

Michael Kelly
mike@michaeldkelly.com
www.MichaelDKelly.com