Automated Testing Playbook

Rapid Prototyping

When it's OK to get by without writing automated tests, and when it's time to switch gears and add them.

WORK IN PROGRESS: This material is currently in draft form and under active development. See the GitHub issues page to examine progress.

Permission

You don't have to follow any dogma to be a "good tester" or a "good programmer." Testing is a critically important, underused tool, but beware of overusing it, too.

Exploration

This the phase in which the answers to any of the following questions are not yet clear:

- What is the problem space?
- What patterns exist in the input?

- What is the most effective design?

Not all problems have clear, up-front approaches, and sometimes design insights don't come until you've spent some time thinking hard about the problem. Depending upon the creative process of the individual or team, locking down implementation details with tests at this point may inhibit rapid experimentation leading to valuable insights.

Exploration vs. ...

Vision: If the approach is obvious, or at least clear given previous analysis and exploration, start testing now! That will make it that much easier to move into...

Settlement: When people depend on your code, you're no longer exploring; you've moved into the Settlement phase, where maintaining existing behavior is at least as important as adding new behavior.

Run-once code

This is code that, by definition, can't be reused. At least, it can't be reused in the sense that it will integrate directly with another program, or not in a way that you would be expected to maintain it. Often the entire program is run frequently while developing it, and the output is inspected for updates, i.e. *running the code is the test*.

Tricksy bits

Sometimes an operation is so tricky, it's worth testing regardless of whether you're in the Exploration phase or the code is Run-Once. This is usually true of low-level operations that must handle a number of special cases, and the aggregate output of the program isn't "obviously wrong."

When bugs begin to bite

Bugs usually start coming in one-at-a-time. When they do, if you don't want a full-blown infestation, take the opportunity to test them out of existence *now*. It's also possible that in the early stages, fixing bugs may also mean fixing the design while it's still easy.