Software Testing Anti-patterns

http://blog.codepipes.com/testing/software-testing-antipatterns.html

Software testing

https://en.wikipedia.org/wiki/Software_testing

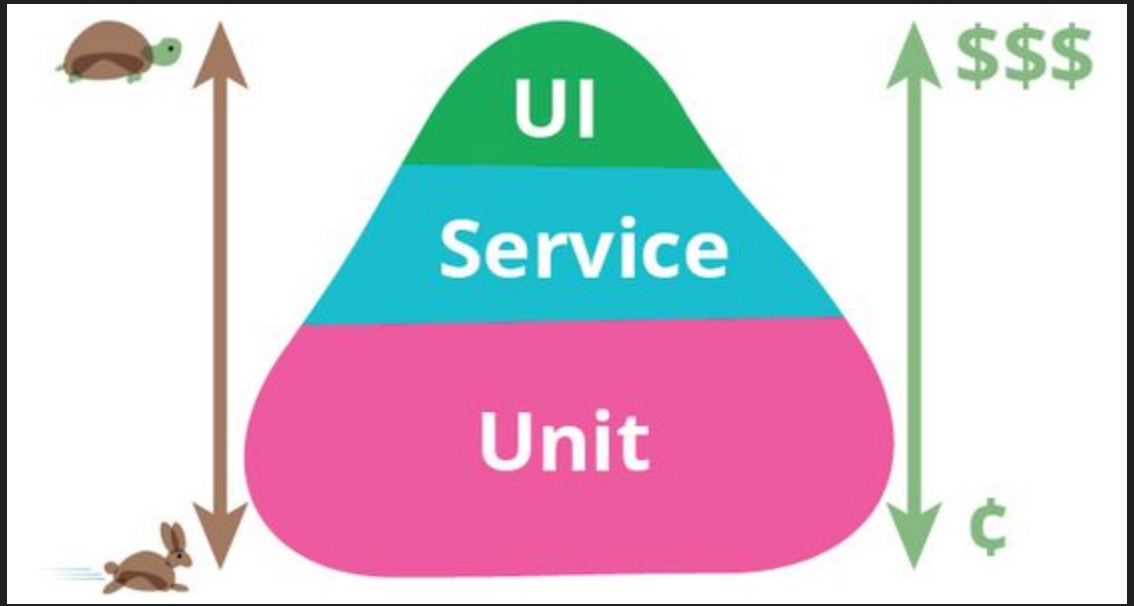
Software design pattern

https://en.wikipedia.org/wiki/Software_design_pattern

Anti-pattern

https://en.wikipedia.org/wiki/Anti-pattern

Test Automation Pyramid



https://martinfowler.com/bliki/TestPyramid.html

Unit/Integration tests

Tests	Focus on	Require	Speed	Complexity	Setup needed
Unit tests	a class/method	the source code	very fast	low	No
Integration tests	a component/service	part of the running system	slow	medium	Yes

http://blog.codepipes.com/testing/software-testing-antipatterns.html

Unit/Integration tests

- uses a database
- uses the network to call another component/application
- uses an external system (e.g. a queue or a mail server)
- reads/writes files or performs other I/O
- does not rely on the source code but instead it uses the deployed binary of the app

http://blog.codepipes.com/testing/software-testing-antipatterns.html

Having unit tests without integration tests

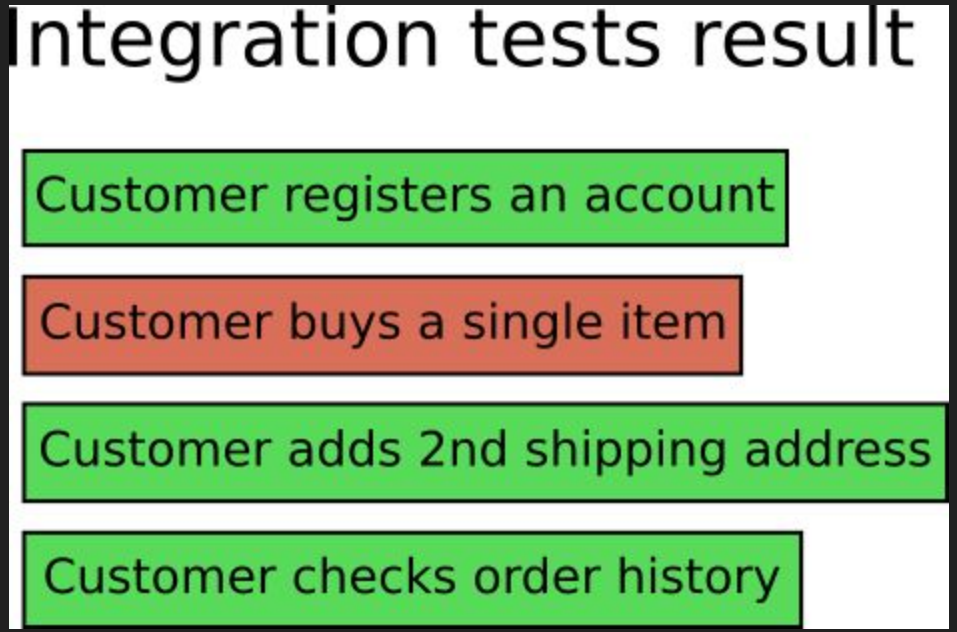
http://blog.codepipes.com/testing/software-testing-antipatterns.html

https://i.pinimg.com/236x/78/17/74/78177442f073d3681a60e7fb853453b3--kitchen-drawers-home-projects.jpg

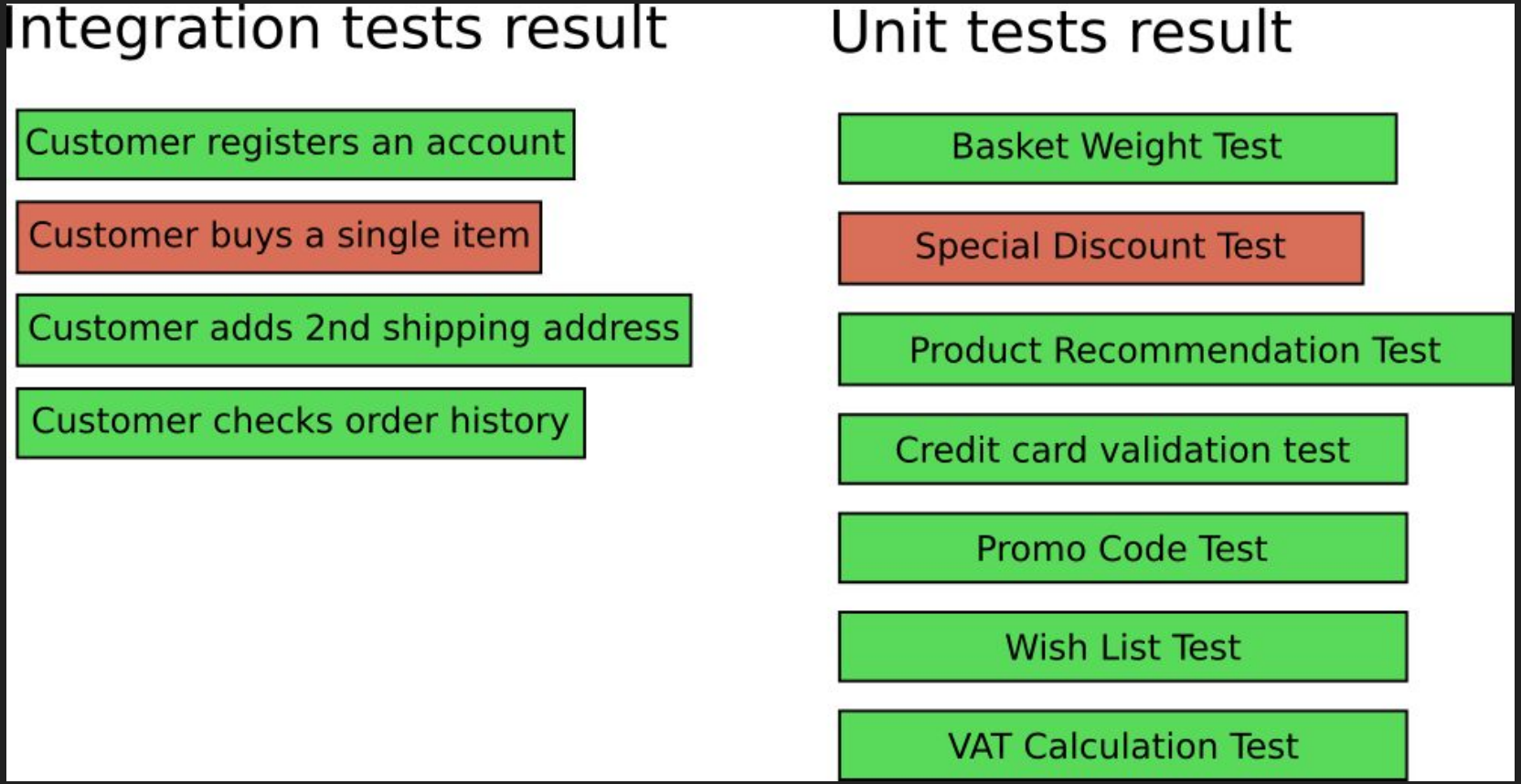
https://i.ytimg.com/vi/0GypdsJulKE/maxresdefault.jpg

Having integration tests without unit tests

http://blog.codepipes.com/testing/software-testing-antipatterns.html



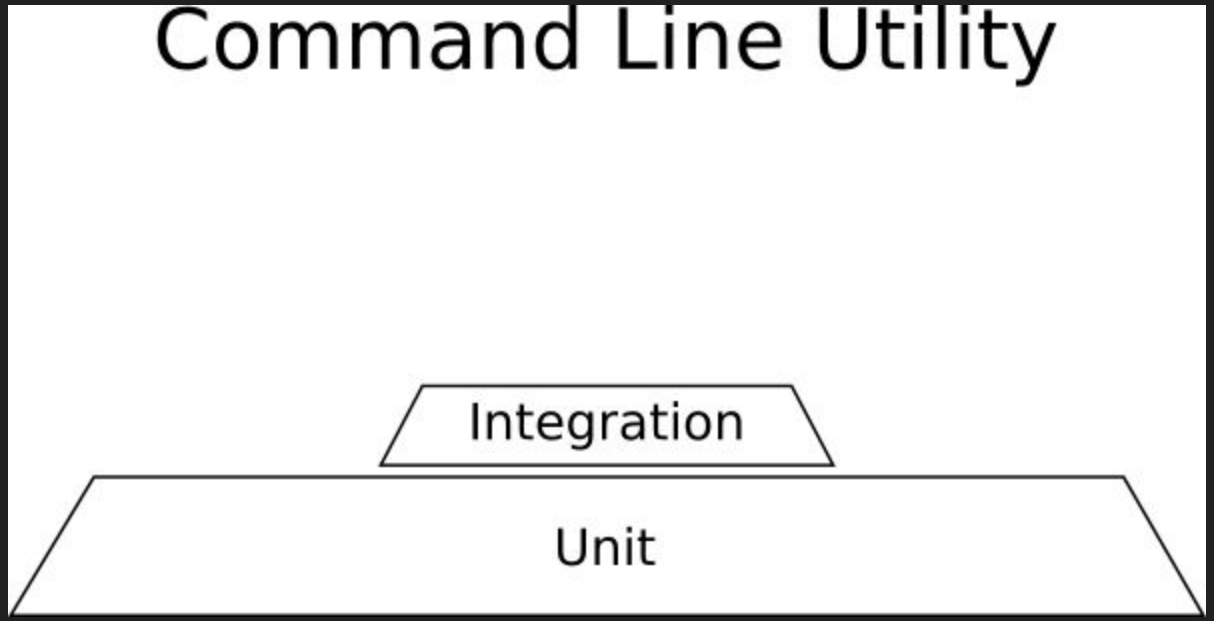
http://blog.codepipes.com/testing/software-testing-antipatterns.html



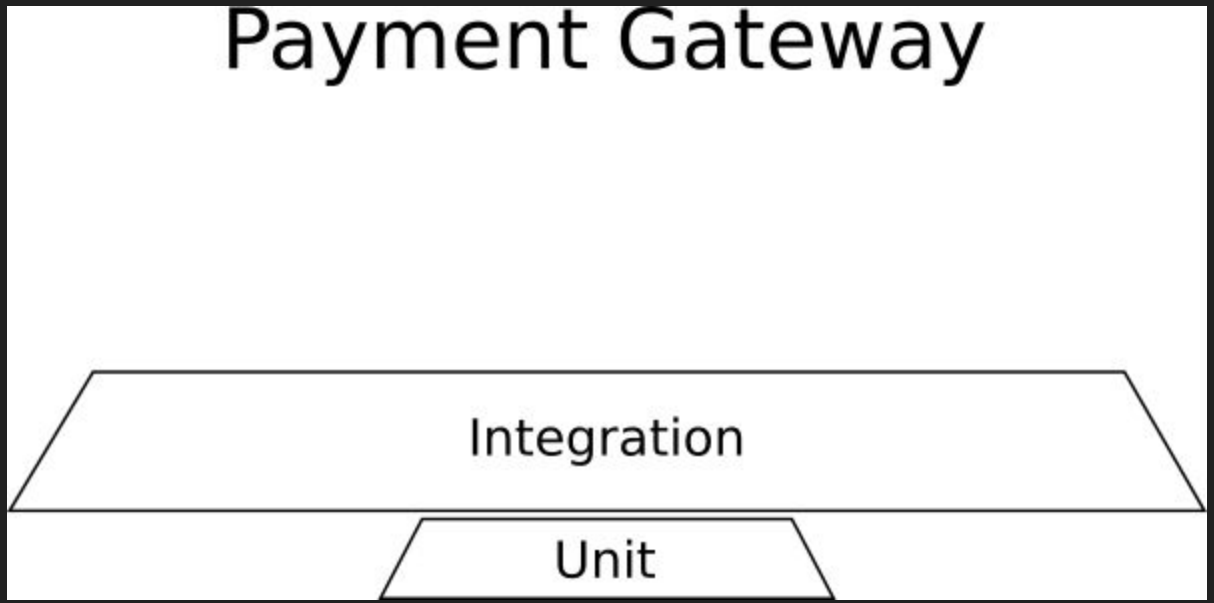
http://blog.codepipes.com/testing/software-testing-antipatterns.html

Having the wrong kind of tests

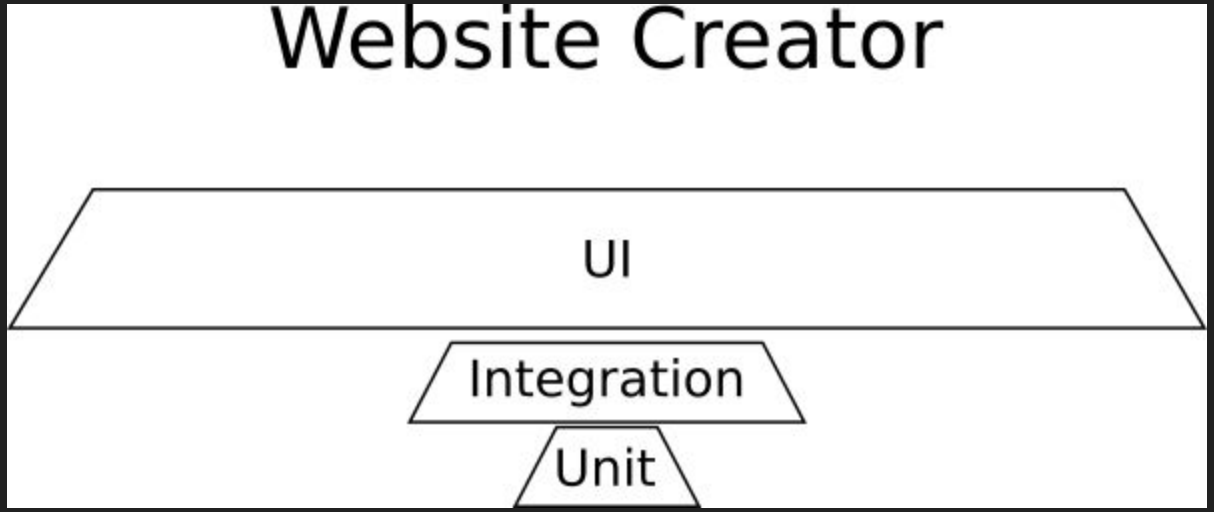
http://blog.codepipes.com/testing/software-testing-antipatterns.html



http://blog.codepipes.com/testing/software-testing-antipatterns.html



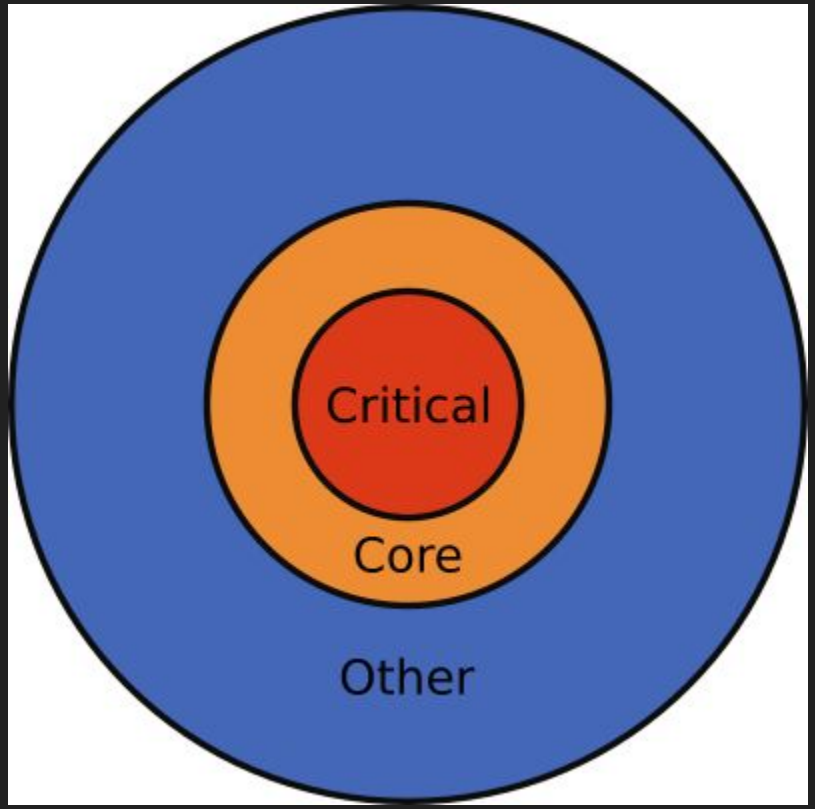
http://blog.codepipes.com/testing/software-testing-antipatterns.html



http://blog.codepipes.com/testing/software-testing-antipatterns.html

Testing the wrong functionality

http://blog.codepipes.com/testing/software-testing-antipatterns.html



http://blog.codepipes.com/testing/software-testing-antipatterns.html

Testing internal implementation

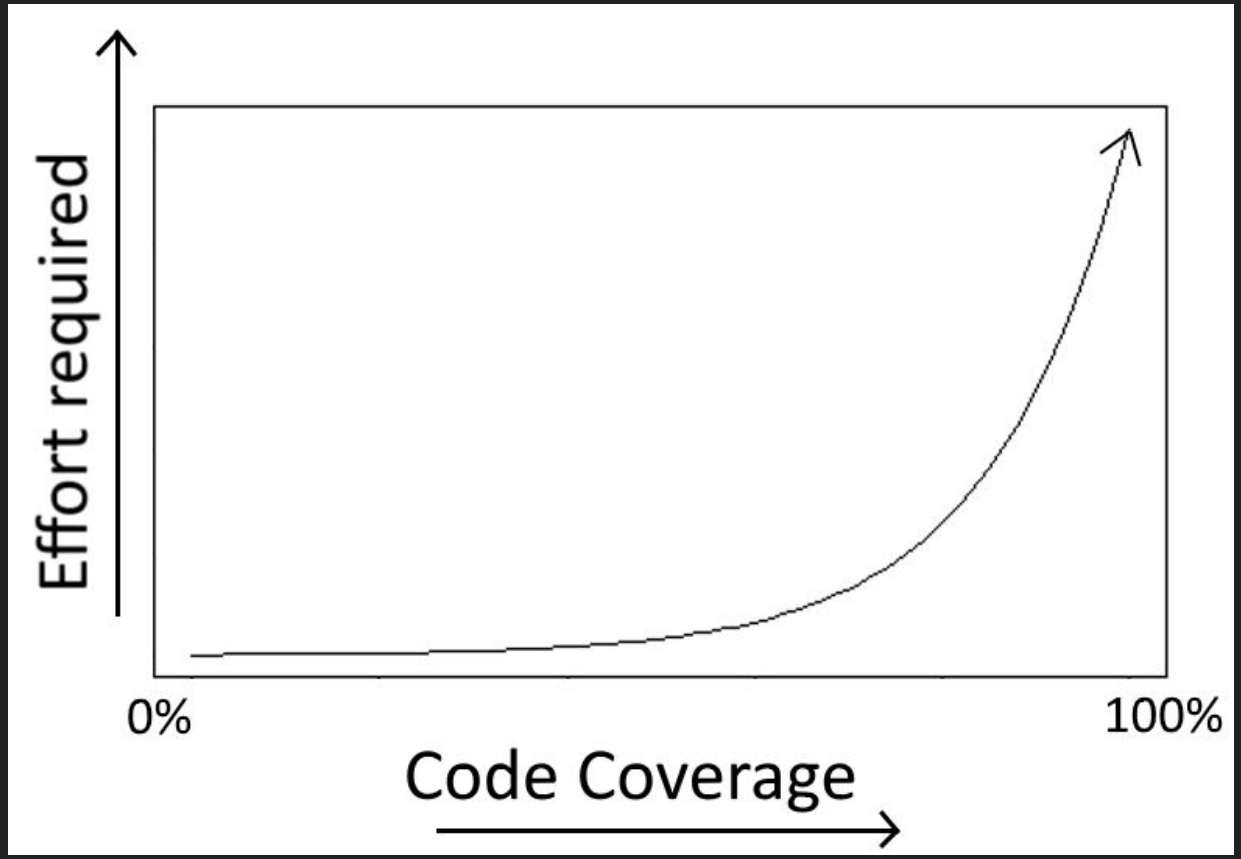
http://blog.codepipes.com/testing/software-testing-antipatterns.html

Paying excessive attention to test coverage

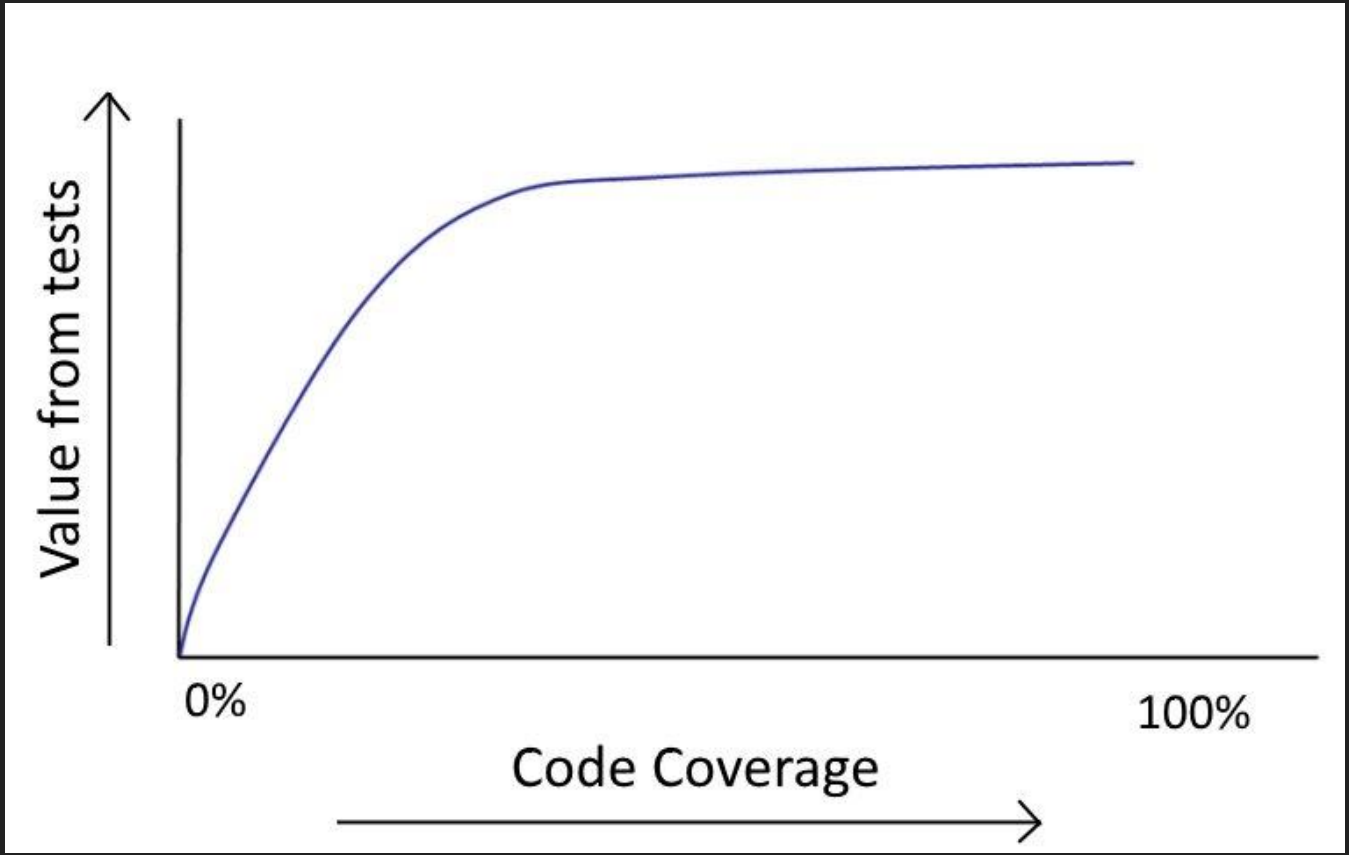
http://blog.codepipes.com/testing/software-testing-antipatterns.html

- PDWT Percent of Developers who Write Tests
- PBCNT Percent of Bugs that Create New tests
- PTVB Percent of Tests that Verify Behavior and not implementation
- PTD Percent of Tests that are Deterministic to total tests

http://blog.codepipes.com/testing/software-testing-antipatterns.html



http://blog.codepipes.com/testing/software-testing-antipatterns.html



http://blog.codepipes.com/testing/software-testing-antipatterns.html

Having flaky or slow tests

http://blog.codepipes.com/testing/software-testing-antipatterns.html

Running tests manually

http://blog.codepipes.com/testing/software-testing-antipatterns.html
8/13 E7

Treating test code as a second class citizen

http://blog.codepipes.com/testing/software-testing-antipatterns.html

Not converting production bugs to tests

http://blog.codepipes.com/testing/software-testing-antipatterns.html

10/13 F#

Treating TDD as a religion

http://blog.codepipes.com/testing/software-testing-antipatterns.html

Writing tests without reading documentation first

http://blog.codepipes.com/testing/software-testing-antipatterns.html
12/13 G#

Giving testing a bad reputation out of ignorance

http://blog.codepipes.com/testing/software-testing-antipatterns.html

13/13 Am7