

DEVELOPER DISCIPLINES

CHRIS HOWE-JONES

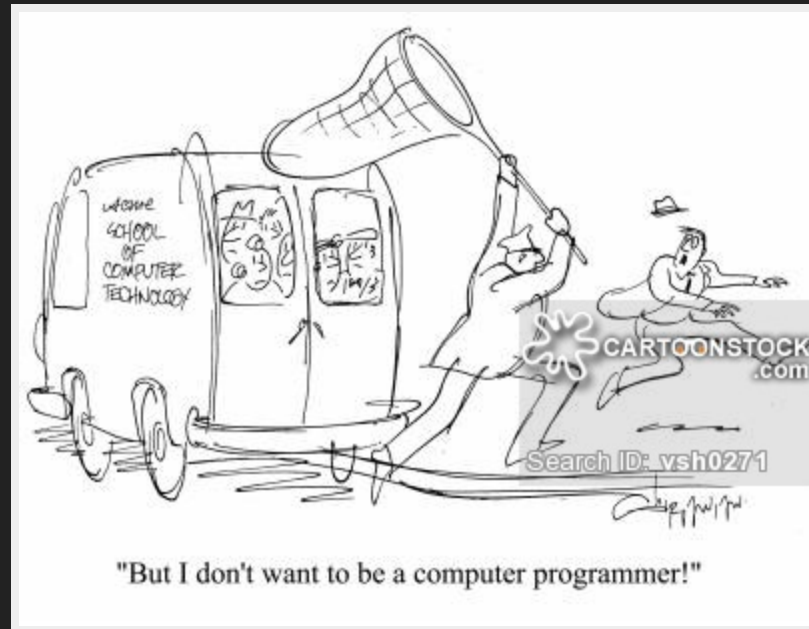
@AGILE_GEEK

14TH MARCH 2016

DEVELOPER DISCIPLINES

Who wants to be a Developer?

DEVELOPER



What makes a 'Developer'?

- Why it matters?
- Why am I qualified to talk to you?

WHO AM I?

Name: Chris Howe-Jones

Job Title: Technical Navigator

Twitter: @agile_geek

Github: <http://github.com/chrishowejones>

Blog: <http://chrishowejones.wordpress.com>

LinkedIn: <https://uk.linkedin.com/in/chrishowejones>

Organiser:

London Clojurians, Clojure eXchange, Functional
Programming eXchange

Speaker:

FP North East, Agile North East, London
Clojurians, Dynamo Conference

WORKED FOR?



A word cloud featuring various client names in two colors: green and purple. The names are arranged in a non-uniform, overlapping manner. The largest words are 'Opencast Software' in purple and 'Sage' in green. Other prominent names include 'Tecsphere' in green, 'Strategic System Solutions' in purple, 'Devcycle' in green, and 'NHS' in purple. Smaller names include 'Insure The Box', 'HMRC', 'Credit Swiss', 'Mastodon C', 'Capgemini', 'JHC', 'Citi Bank', 'ICommunico', 'Goal Group', 'Lloyds Bank', 'Onsiteformz', 'Deutsche Bank', 'JP Morgan Chase', 'Ibacus', and 'ORE Catapult'.

Insure The Box
HMRC
Mastodon C
Sage
Credit Swiss
Capgemini
JHC
Citi Bank
Tecsphere
ICommunico
Opencast Software
Goal Group
Strategic System Solutions
Lloyds Bank
Deutsche Bank
Onsiteformz
Devcycle
NHS
Ibacus
JP Morgan Chase
ORE Catapult

JOB?

A word cloud of various job titles. The word 'Developer' is the largest and most central, rendered in a large, red, serif font. To its right, 'CTO' is also large, in a dark brown, serif font. Other titles are arranged around them in various sizes and colors (red, purple, brown). The titles include: Analyst, Analyst Programmer, Lean Coach, Tech Lead, Software Architect, Technical Architect, Team Leader, Scrum Master, Solutions Architect, Agile Coach, Enterprise Architect, Managing Director, Data Architect, and Project Manager.

Analyst

Analyst Programmer

Lean Coach

Tech Lead

Software Architect

Technical Architect

Team Leader

Scrum Master

Solutions Architect

Agile Coach

Enterprise Architect

Managing Director

Data Architect

Project Manager

Developer

CTO

LANGUAGES?



TECHNOLOGIES AND DISCIPLINES?



SO?

Why do you care?

Because I'm the person who will employ you!

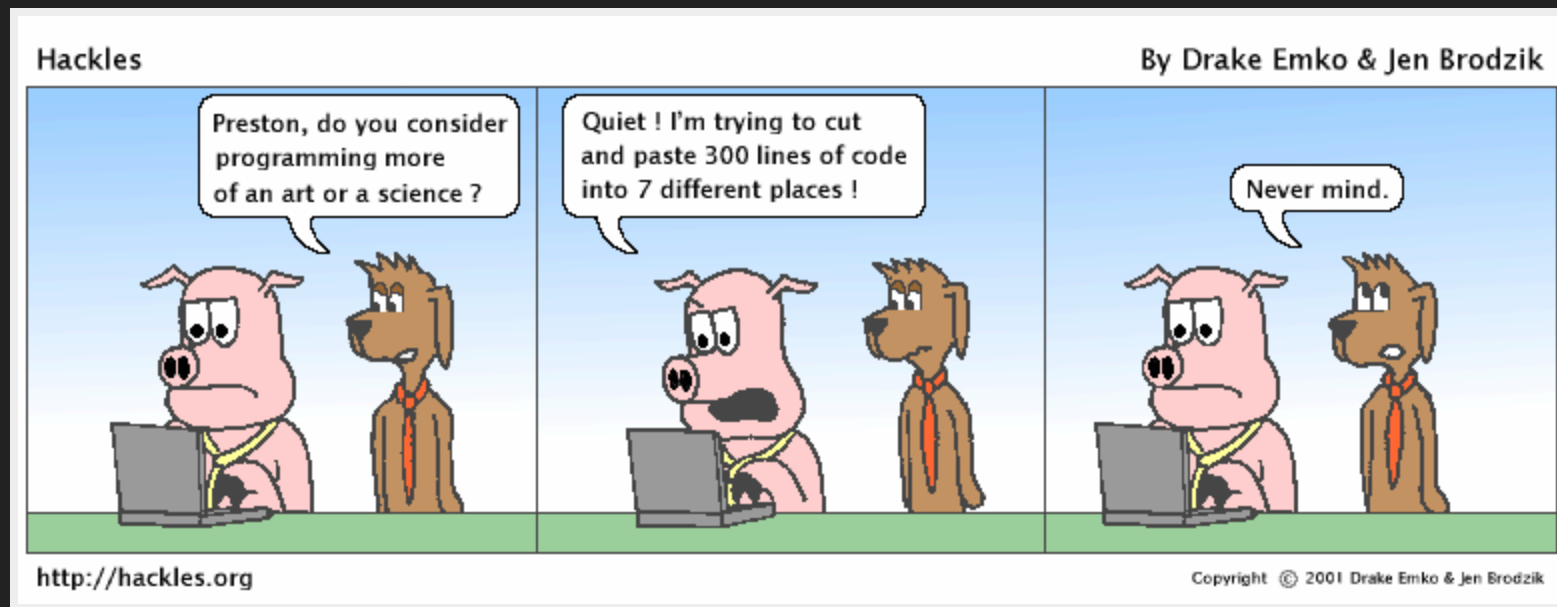
Or someone just like me

WHAT DISCIPLINES?

Don't I just need to program?

What else do I need?

SOFTWARE DEVELOPMENT IS A COMPLEX WORLD



MULTI-DISCIPLINED TEAMS

- Developers/Programmers
- Testers/QA
- Business Analysts
- Scrum Masters
- Product Owners
- Project Managers
- Tech Leads
- Team Leaders
- Product Managers

COMPLEX ENVIRONMENT

- Polyglot Programming
 - multiple languages
- Polyglot Persistence
 - multiple persistence technologies
- Cloud Deployment
 - elastic virtualised environments
 - servers on demand

DISCIPLINES

- TDD
- BDD
- Test Pyramid
 - Unit,
 - System,
 - Integration,
 - Performance,
 - Stress,
 - Failure,
 - Load,
 - Functional

- SDLC
 - 'Waterfall',
 - Scrum,
 - DSDM,
 - Kanban
- Version Control
 - frequent,
 - small,
 - incremental

- Continuous Integration
- Continuous Deployment
- Automated testing
- Automated build
- Static code analysis
- Peer review
- Functional review
- Refactoring
- Debugging

- SOLID principles

- *Single Responsibility Principle - a class should have only a single responsibility*
- *Open for Extension, Closed for Modification*
- *Liskov Substitution Principle - "objects in a program should be replaceable with instances of their subtypes without altering the correctness of that program."*
- *Interface Segregation Principle - "many client-specific interfaces are better than one general-purpose interface."*
- *Dependency Inversion Principle - one should "Depend upon Abstractions. Do not depend upon concretions."*

- Referential transparency
- Immutability
- Reduced side effects

WHAT'S MOST IMPORTANT?

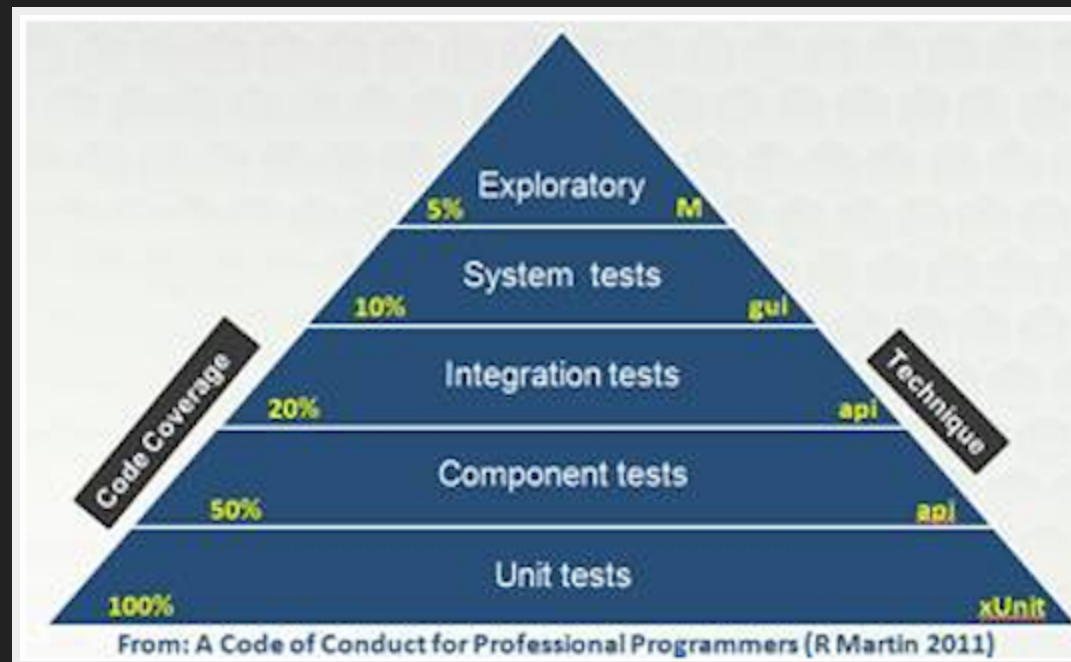
I had a manager who asked?

How do you know when you've finished?

- You decide what success is
- You write tests to prove it
- You implement your code
- You run your tests
- They pass

YOU'RE DONE!

TEST, TEST, TEST AGAIN...



TYPES OF TESTING?

Only really two types of test...

- *Developer Tests*
- *User Tests*

DEVELOPER TESTS

- Unit
 - Integration
 - System
 - Non functional
 - Performance
 - Security
 - Stress
 - Resilience
- ...etc

USER TESTS

Functional

- Smoke tests
- Sanity testing
- Explorative testing
- Regression testing
- Usability testing
- Accessibility testing
- Acceptance testing

TESTS TO DISCOVER DESIGN

Test Driven Development

TEST DRIVEN DEVELOPMENT

What is Test Driven Development?

WHAT?

Incremental process to build low level design through the feedback mechanism of tests, written tests first.

WHY?

- Tests code (automated and run in build).
- Evolves design constantly checking against tests for 'completeness'.
- Provides automated 'safety net' to catch 'breaking changes'.
- Enables rapid and radical design changes in future.
- Avoids 'big ball of mud'

Good design is testable, And design that isn't testable is bad.

"I haven't got time to test that..." If it's worth building, it's worth testing.

If it's not worth testing, why are you wasting your time working on it?

Tests are your first users.

Tests can be your documentation

If TDD hurts..you're doing it wrong.

"The story of the unforeseen requirement."

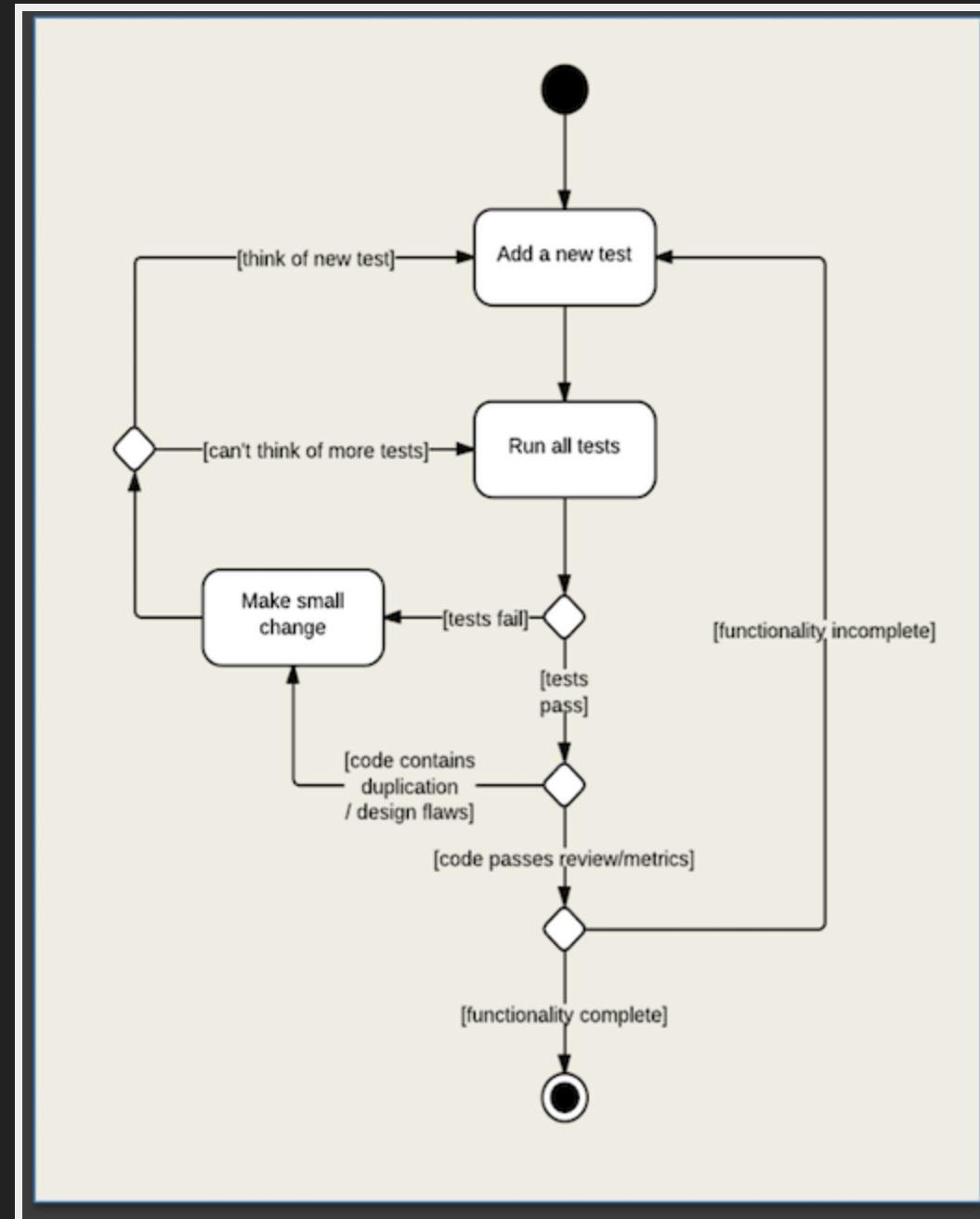


www.phdcomics.com

HOW?

1. Write a (failing) test.
2. Run a (failing) test.
3. Write the (simplest) implementation to get the test to pass.
4. Run test (if pass do next step else do 3)
5. Refactor implementation (and/or test) to remove duplication.
6. Repeat from step 1.

HOW?



WHEN?

ALL THE TIME

TDD?

Test Driven

Design

VERSION CONTROL - SCM

Git, Subversion, CVS, VSS, Mercurial, PVCS

WHAT IT'S NOT

- Backup
- Centralised code sharing



WHAT IT IS

It's a Time Machine

Small, incremental changes

Record of events

Who

What

When

Why

WHAT CAN YOU DO WITH IT?

Rewind time

Try out a change safely

Integrate code across team

WHY IS IT IMPORTANT?

It tells a story.

It protects you from mistakes.

It enables review.

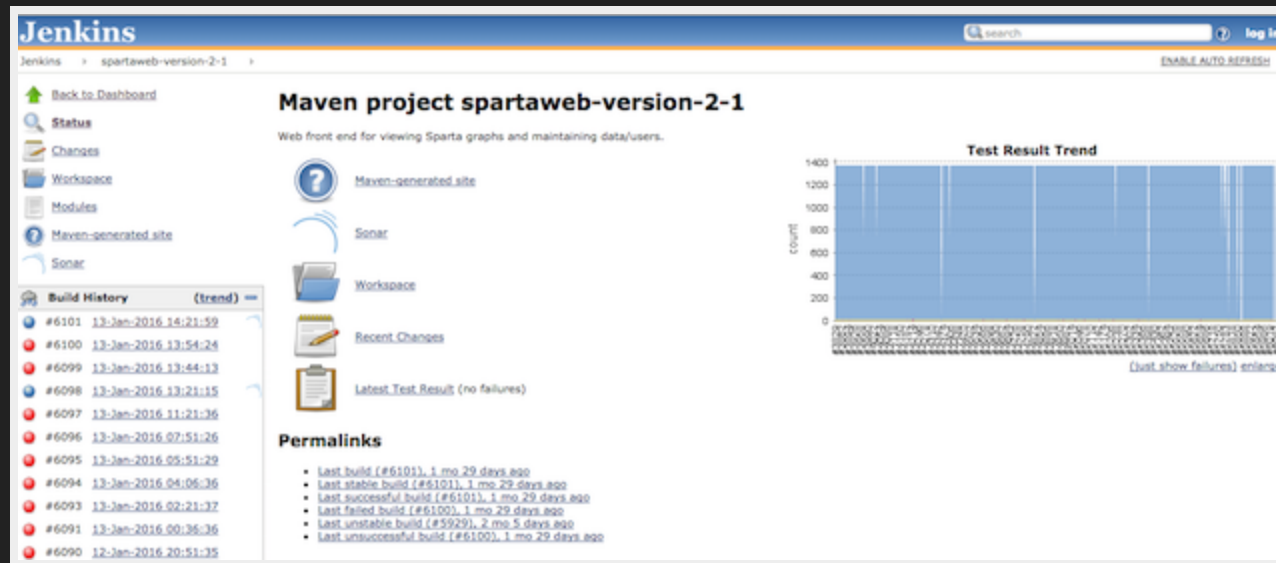
OTHER IMPORTANT DISCIPLINES?

CONTINUOUS INTEGRATION

"Continually integrating (and testing) code across a team."

- Merging code regularly.
- Regression testing.

CONTINUOUS BUILD




"Continually build (and testing) code across a team."


- Automated builds.

Jenkins


Jenkins > spartaweb-version-2-1 > #6101

[Back to Project](#)
[Status](#)
[Changes](#)
[Console Output](#)
[View Build Information](#)
[Polling Log](#)
[Git Build Data](#)
[Test Result](#)
[Redeploy Artifacts](#)
[Aggregated Test Result](#)
[See Fingerprints](#)
[Previous Build](#)


**Build #6101 (13-Jan-2016 14:21:59)**



No changes.




[Started by an SCM change](#)




Revision: 1ac0583710761588da02f24bc692a99a9b9bc333

- origin/Version-2.1




[Test Result](#) (no failures)



[Aggregated Test Result](#) (no tests)

Module Builds

 [spartaweb](#) 13 min

- Build on merges.
- Tests run on build.
- E.g. Jenkins, TeamCity, etc.

CONTINUOUS DEPLOYMENT

"Continually deploy (verified) code to 'live'."

- Automated deployment.
- 'Push Button' deployment.

STATIC CODE ANALYSIS TOOLS

The screenshot displays the SonarQube web interface with the following sections:

- Home** (top navigation)
- Configuration** | **Log In** | **Search** (top right)
- Helicopter View** (left sidebar):
 - Projects
 - Super Heroes
 - Recent Activity
 - Reviews
 - Dependencies
 - Motion chart
 - Views
- Sonar as a Service** (left sidebar, below Helicopter View)
- CloudBees** (left sidebar, below Sonar as a Service)
- Forges** (table):

Name	Lines of code	SQALE Rating
Forges	8,021,272	B
Apache	4,103,834	B
Others	1,965,713	B
JBoss	560,122	B
OW2	511,805	B
Sourceforge	363,581	B
Codehaus	273,439	B
GoogleCode	123,846	B
QPS4J	68,101	B
SpringSource	50,831	B

9 results
- High Quality Benchmark** (table):

Name	Lines of code	Coverage	SQALE Rating
High Quality Benchmark	108,405	70.9%	B
Logback-Parent	20,377	52.4%	B
Sonar	63,354	70.7%	B
Spring Batch	24,674	87.5%	B

3 results
- Super Heroes loving Unit Tests** (table):

Name	Lines to cover	Coverage
rafael.hertzog@gmail.com	7,413	91.3%
bayard	6,179	88.9%
scolecbourne	11,920	88.3%
jahlborn	7,211	86.9%
castagna	26,776	77.3%
ngn	8,849	76.2%
godin	6,806	76.0%
ltheussl	7,040	76.0%
joehni	7,417	75.2%
Johan Svensson	5,767	71.6%
niailp	5,931	69.1%
Simon Brandhof	21,886	67.6%
Mattias Persson	14,511	64.2%
paul	5,465	63.3%
poutsma	8,269	63.1%
ceki	8,097	61.1%
alesj	9,749	60.7%

17 results
- Projects** (heatmap):

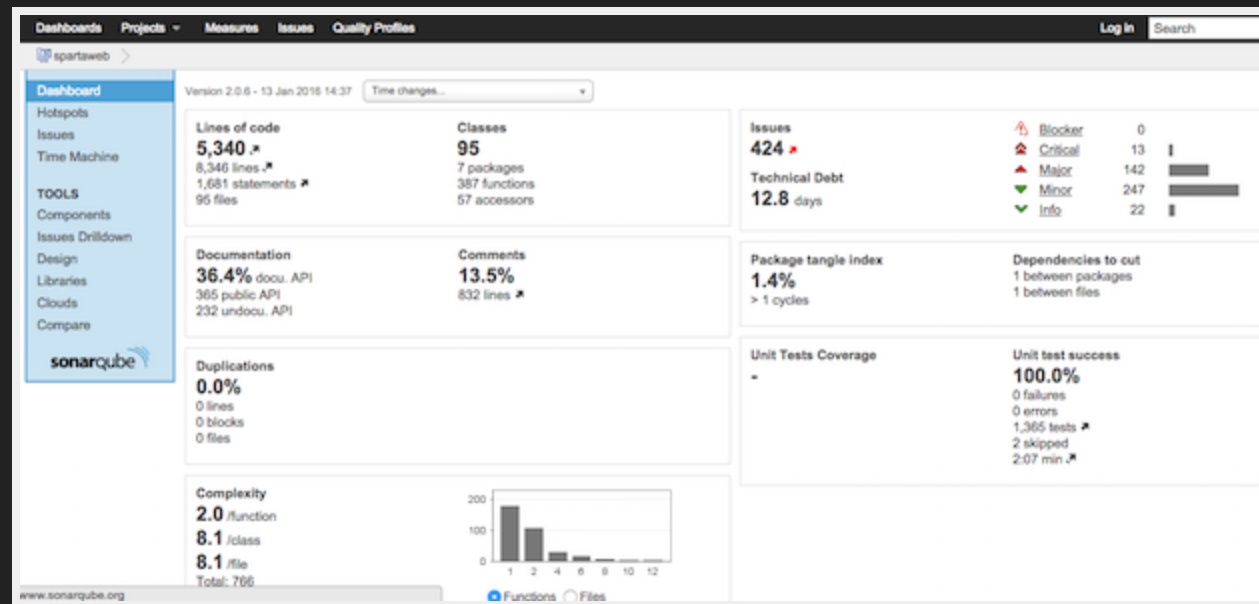
Size: Lines of code (0.0% to 100.0%)

Color: 0.0% to 100.0%

Coverage: 0.0% to 100.0%

Heatmap showing project coverage across various categories (e.g., Apache CXF, Silverpeas, Apache Jackrabbit, OpenEJ, Apache Build, OpenJ, JOnAS, Apache Tomcat, Jaha Project, Jetape, Apache Enterprise, Cayer, Wicket, Apache Sling, Jett, XWiki, Neo4j, Jenkins, Cast, Boni, Apache, Maven, Hudson, Apache, Confluence, Camel, PDFB, Sonar, All, Apache, JBP, Apache, Do, XW, Apache, Pict, JBoss Application Server, Struts, Active, JBoss, Apache, JBP, Apache, Do, XW, Apache, Pict, JBoss Application Server, Struts, Active, JBoss, Apache, JBP, Apache, Do, XW, Apache, Pict).

"Analyse code for style and common 'bugs'."



- C# - Resharper, FxCop, StyleCop...
- Java - Sonar, CheckStyle, FindBugs...

AUTOMATED CODE FORMATTER

"Automatically format code to 'team' or 'organisational' standards."

PROPERTY OR GENERATIVE TESTING

"In computer science, a property testing algorithm for a decision problem is an algorithm whose query complexity to its input is much smaller than the instance size of the problem. Typically property testing algorithms are used to decide if some mathematical object (such as a graph or a boolean function) has a "global" property, or is 'far' from having this property, using only a small number of 'local' queries to the object."

WHAT?

"A high level approach to testing in the form of abstract invariants [that] functions should satisfy universally."

WHAT?

"Property-based tests make statements about the output of your code based on the input, and these statements are verified for many different possible inputs."

COMMAND LINE IS YOUR FRIEND

"Don't be afraid of the command line."

**WHAT OTHERS SAY ARE IMPORTANT
DISCIPLINES?**



"Take baby steps and always look for stepping stones, be obsessed with feedback, keep it simple, communicate as much as you can"

Giuseppe Capizzi, Developer Pivotal Labs

Co-organiser: CodeLovers

Speaker: Bergamo Linux User Group, CodeLovers, Milano XP User Group, ClojureBridge



*"...SCM, tests, simplifying, static analysis, reviews
to share knowledge." "...thinking, talking,
sharing..."*

Glen Mailer, Freelance Software Developer

Organiser Sheffield Geeks

Speaker: Sheffield Geeks, Clojure eXchange 2015



"deliberate practice using fast feedback."

Philip Potter, Tech lead on registers at @gdsteam

Speaker: London Clojurians, EuroClojure, Clojure eXchange,
CodeMesh

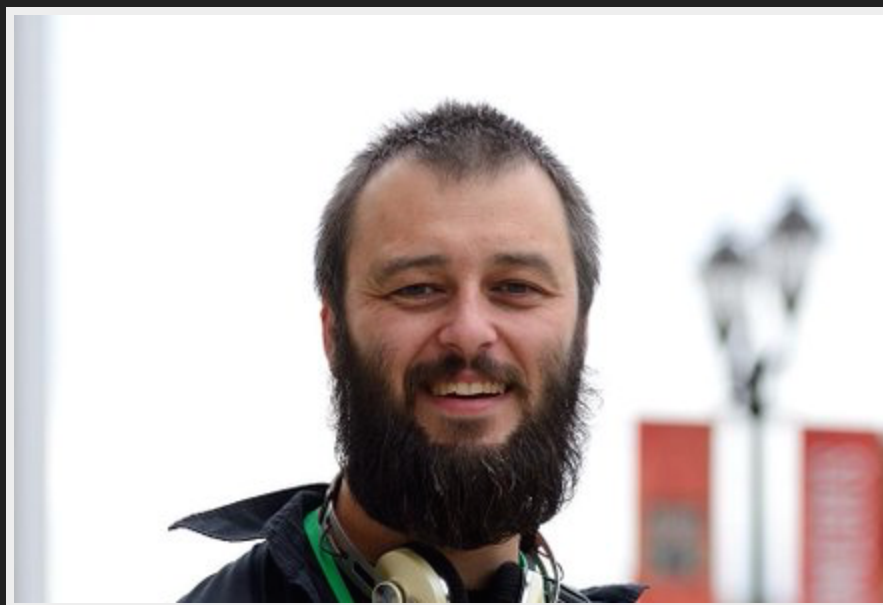
*"code as if you are going to support your own
code...because you might have to."*

Mazhar Iqbal, Tech lead LLoyds Bank



"learn your short-cut keys."

Stephen Hobbs, Software Developer at Orchid Software



"frequent commits to source control."

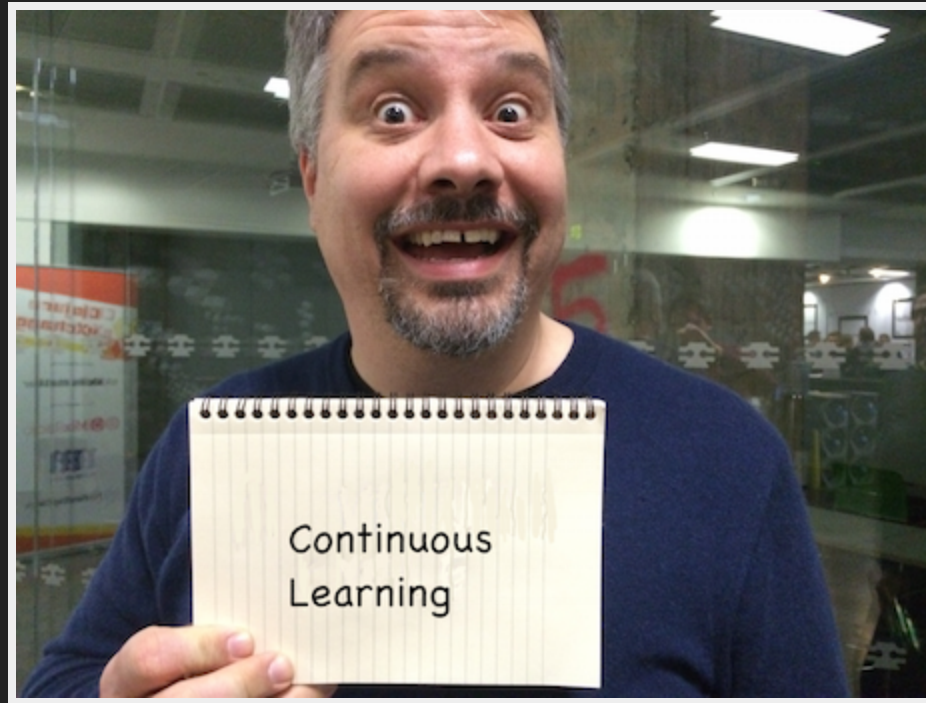
Chris Ford, Functional Composer at Thoughtworks

Speaker: London Clojurians, EuroClojure, Clojure eXchange, Strange Loop - St Louis, GOTO Berlin



"Simplifying code, document processes, range based estimates, low coupling, code formatters, lint, automation, few side effects."

Michael Langford, Founder and iOS Development Director Rowdy Labs, Atlanta GA



Bruce Durling, Co-founder & CTO Mastodon C, London
Co-founder London Clojurians, Co-organiser Clojure eXchange,
EuroClojure board, Co-organiser Functional Programming
Exchange.

Speaker: Data Science, Clojure, Functional Programming.