Testing Pyramid

Manchester Digital

The plan

Today
Theory

Slack for questions #bootcamp
Zoom and breakouts

Tomorrow
Practical

Manchester
Digital

Sorry, what do you do?. . .

Test engineer

I'm a . . .

Exploratory tester

Quality Analyst

Software tester

Tester

~~Quality Assurance~~

Load Tester

Performance tester

Software test engineer

Risk tester

Manual Automation Tester

Job title I've actually seen

Frontend tester

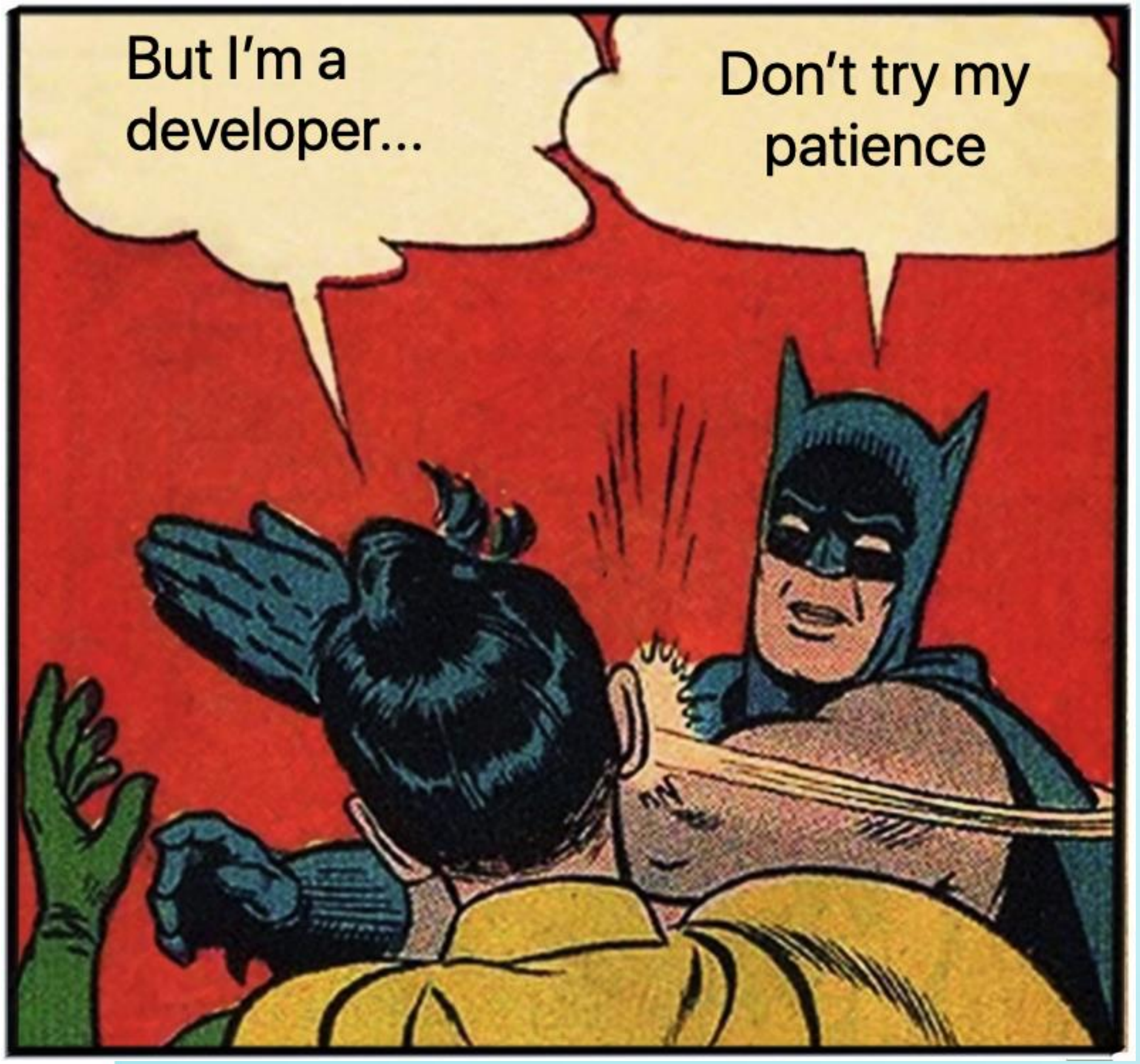
Manual Tester

Penetration Tester

Test Analyst

QA

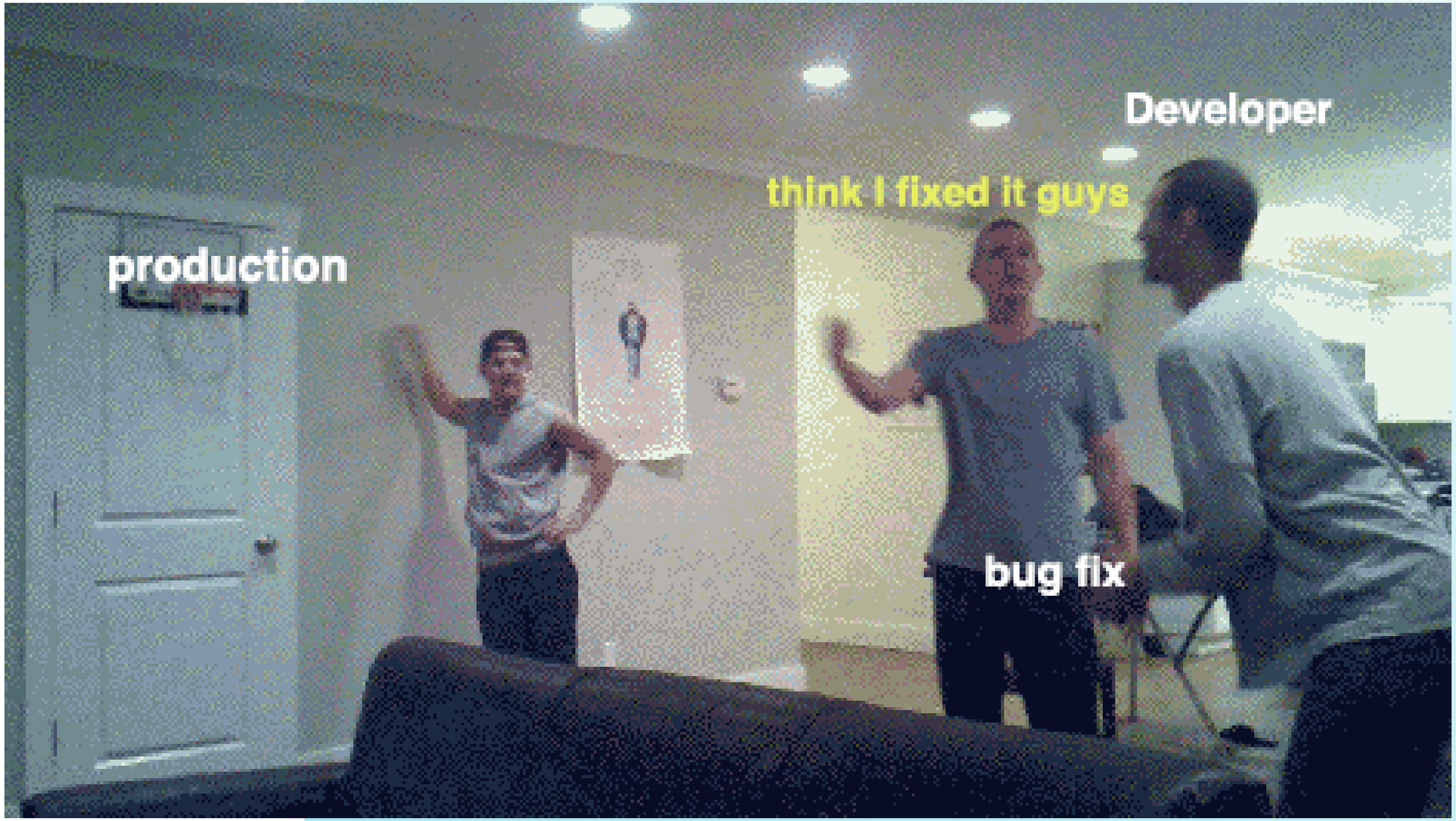
Manchester Digital



Manchester Digital

Why do I need to know about testing?

- The sooner bugs are found the cheaper they are to fix
- Stop faults recurring
- Make sure your code does what the customer wants
- Boundaries between developer and QA/Tester are more blurred



Manchester Digital

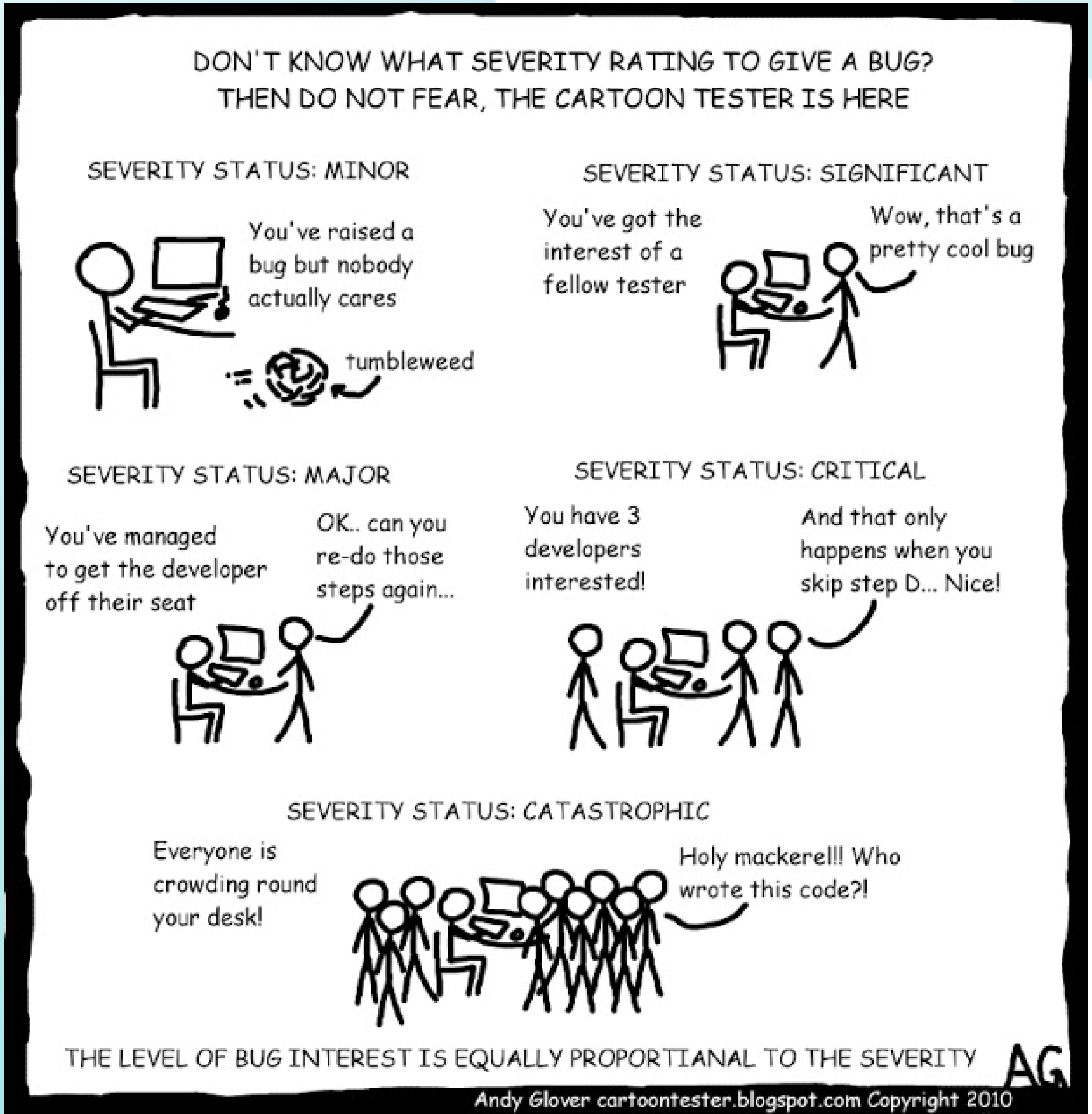
What testing do I need to do?

- Many different kinds of testing
- Lots of different tools and frameworks
- Not all will suit all projects/teams
- Pick what works for you

Manchester Digital

It's good to fail

Manchester Digital


DON'T KNOW WHAT SEVERITY RATING TO GIVE A BUG?
THEN DO NOT FEAR, THE CARTOON TESTER IS HERE
SEVERITY STATUS: MINOR
You've raised a bug but nobody actually cares
tumbleweed
SEVERITY STATUS: SIGNIFICANT
You've got the interest of a fellow tester
Wow, that's a pretty cool bug
SEVERITY STATUS: MAJOR
You've managed to get the developer off their seat
OK.. can you re-do those steps again...
SEVERITY STATUS: CRITICAL
You have 3 developers interested!
And that only happens when you skip step D... Nice!
SEVERITY STATUS: CATASTROPHIC
Everyone is crowding round your desk!
Holy mackerel!! Who wrote this code?!
THE LEVEL OF BUG INTEREST IS EQUALLY PROPORTIANAL TO THE SEVERITY
AG
Andy Glover cartoontester.blogspot.com Copyright 2010

Manchester Digital

Logical Thinking

1

11

21

1211

111221

What's next?

ASDO

XCBN

QPJG

WETL

Which line is the odd one out?

Manchester Digital

Manual Testing

In a group

Breakout room in zoom 4 and a mentor

Manchester Digital

Manual Testing

- Code review
- Running by hand and checking it does what it should
- Time-consuming if you need to do it more than once
- Have to write and maintain manual test cases if you want tests to be repeatable
- May not pick up knock-on effects in complex systems
- Maybe we should look at automated testing...

Manchester Digital

Static Checking

- Analyse code for errors or code standards violations
- Done without running the code
- Catch well-known gotchas
- Helps keep code consistent and readable
- Won't make sure your code works to spec

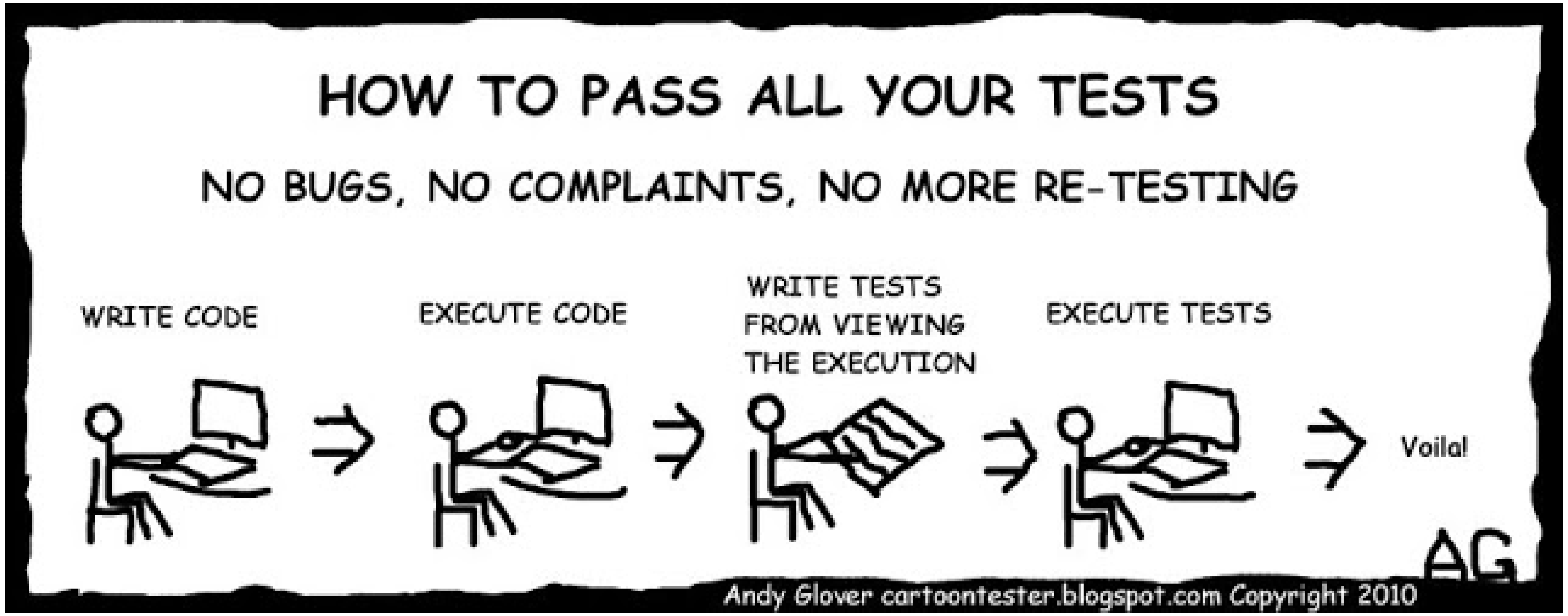
- ADD EXAPMLES

What is a “Unit”?

Manchester Digital

Unit Testing

- Exercise small section of code
- Fast to run
- Gives us confidence to refactor code
- To TDD or not TDD


HOW TO PASS ALL YOUR TESTS
NO BUGS, NO COMPLAINTS, NO MORE RE-TESTING
WRITE CODE
EXECUTE CODE
WRITE TESTS FROM VIEWING THE EXECUTION
EXECUTE TESTS
Voila!
AG
Andy Glover cartoontester.blogspot.com Copyright 2010

Manchester Digital

```
expect(umbrellaOpens).toBe(true)
tests: 1 passed, 1 total
**all tests passed**
```

Manchester Digital

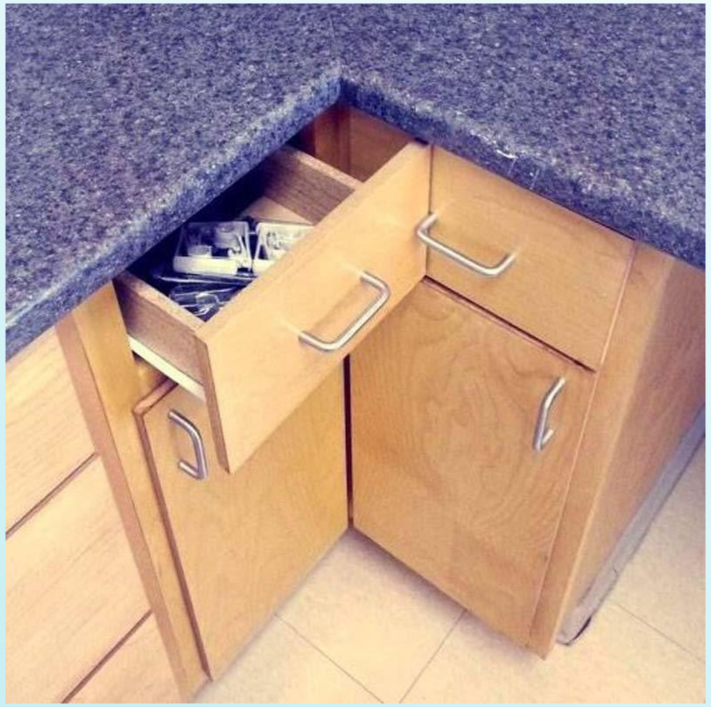
What makes a good test?

- Keep it simple, stupid
- Smaller the better
- Don’t intertwine tests
- Think very clearly about what you want to know from the test

- https://www.maxpou.fr/10-tips-write-better-tests

Manchester Digital

2 Unit Tests 0 Integration tests



Manchester Digital

What is an Integration Test?

Manchester Digital

Integration Testing

- Slower to run
- Use real objects not mocks
- Examples of code and tests, these will be covered tomorrow on practical
- Talk about their understanding and how their companies work

Manchester Digital

Mock Objects

- Mocks vs. Stubs vs. Fakes

- Need much more information around this examples between and the reasons why?

Manchester Digital

Mocking at home

- How does your organisation use mocking?
- What framework?
- How do you do contract testing?

- Examples and how they work now?
- Pros and cons of this approach

Manchester Digital

Contract Testing

- ***Contract testing*** *ensures that a pair of applications or micro-services will work correctly together by checking each service in isolation to ensure the messages it sends or receives conform to a shared understanding that is documented in a "contract".*
- For applications that communicate via HTTP, these "messages" would be the HTTP request and response, and for an application that used queues, this would be the message that goes on the queue.
- In practice, a common way of implementing contract tests is to check that all the calls to your test doubles return the same results as a call to the real application would.
- Contract testing is immediately applicable anywhere where you have two services that need to communicate - such as an API client and a web front-end. Although a single client and a single service is a common use case, contract testing really shines in an environment with many services (as is common for a microservice architecture). Having well-formed contract tests makes it easy for developers to avoid version hell. Contract testing is the killer app for microservice development and deployment.

https://sendgrid.com/blog/quickly-prototype-apis-apiary/

https://docs.pact.io/

Manchester Digital

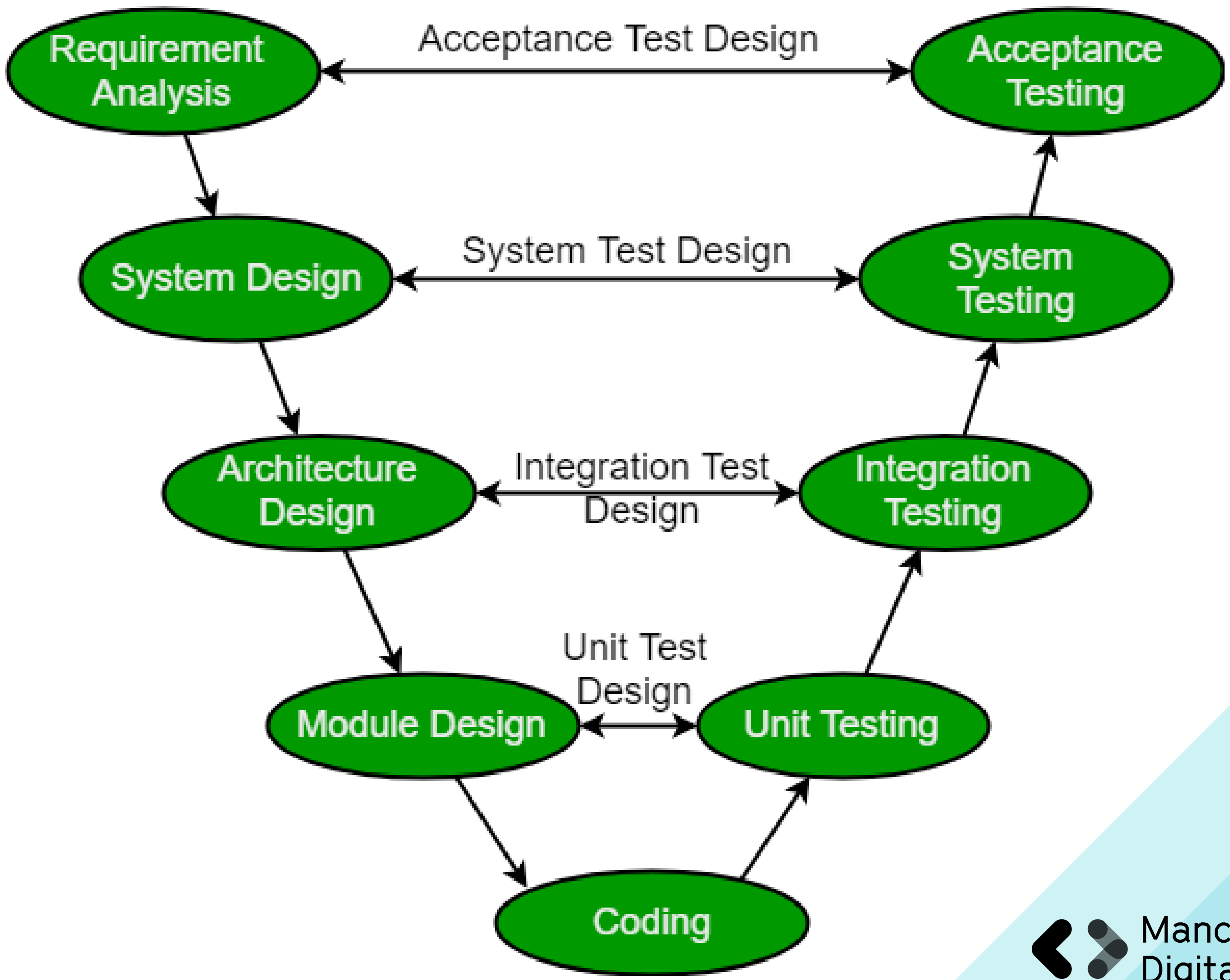
User Interface Testing

- Can catch embarrassing errors
- Often fragile, expensive to maintain
- Tend to be slow
- Hard to run in automated CI

- Make this more exciting – Horror stories and what things have you found?

Manchester Digital

The V-Model



Manchester Digital

Usability

Usability is part of the broader term “user experience” and refers to the ease of access and/or use of a product or website. A design is not usable or unusable *per se*; its features, together with the context of the user (what the user wants to do with it and the user’s environment), determine its level of usability.

- https://uxdesign.cc/10-usability-heuristics-every-designer-should-know-129b9779ac53
- https://www.interaction-design.org/literature/topics/usability

- Explanations, examples and get their experiences

Manchester Digital

Compatibility Tests

- It is a type of non-functional testing.
- Compatibility testing is a type of software testing used to ensure compatibility of the system/application/website built with various other objects such as other web browsers, hardware platforms, users (in case if it’s very specific type of requirement, such as a user who speaks and can read only a particular language), operating systems etc.
- This type of testing helps find out how well a system performs in a particular environment that includes hardware, network, operating system and other software etc.
- It is basically the testing of the application or the product built with the computing environment.
- It tests whether the application or the software product built is compatible with the hardware, operating system, database or other system software or not.

Manchester Digital

Acceptance Tests

- Ensuring Tests meet the specification

- Black box testing
- White box testing

- Rejection tests

- https://danashby.co.uk/2016/08/23/acceptance-testing-what-does-it-mean/

- https://www.developsense.com/blog/2010/08/acceptance-tests-lets-change-the-title-too/

- https://youtu.be/SBhgteA2szg

- Explanations, examples and get their experiences

Manchester Digital

End to End tests

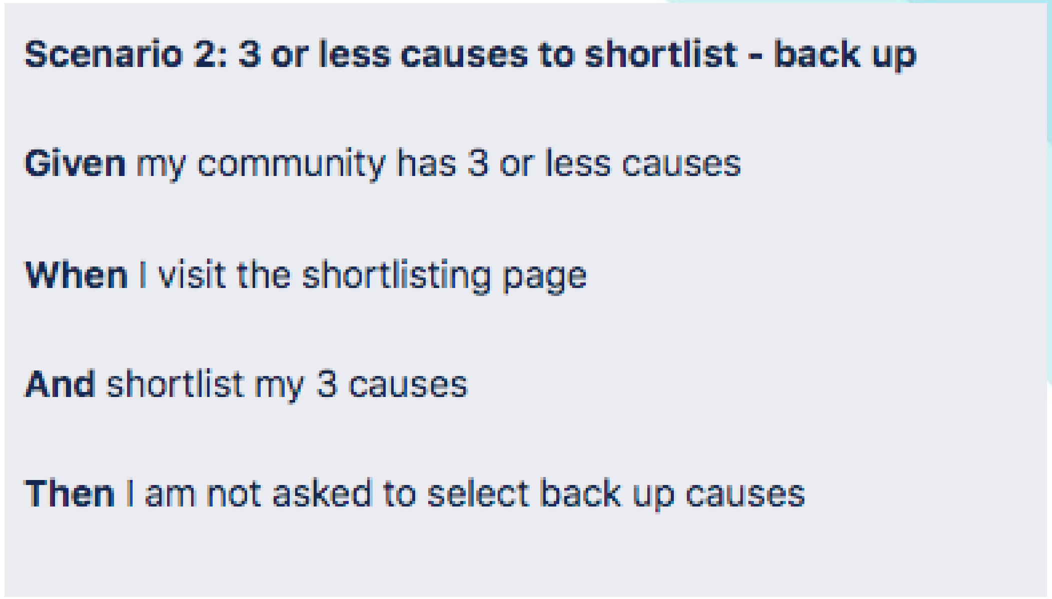
- Start of the journey to the end of the journey
- Mainly using the user journey from start to finish.
- This is important, but can also cause people to go down rabbit holes they didn’t know were there.
- Examples – From the team.
- Tools for capturing or planning I’ve used:
- Mindmaps Google

Manchester Digital

BDD?

- **B**ehaviour
- **D**riven
- **D**evelopment

- Cucumber - docs.cucumber.io/guides
- Gherkin – which is basically a tech way of saying in plain English.



Scenario 2: 3 or less causes to shortlist - back up

Given my community has 3 or less causes

When I visit the shortlisting page

And shortlist my 3 causes

Then I am not asked to select back up causes

BDD

- The BDD approach can largely be divided into two main parts.
  - The first is the practice of using examples written in ubiquitous language to illustrate behaviours (how users will interact with the product).
  - The second part is the practice of using those examples as the basis of automated tests. As well as checking functionality for the user, this ensures the system works as defined by the business throughout the project lifetime.

Other kinds of testing

Do you know what they are and why we do them?

- Code Coverage
- Performance Testing
- Stress Testing
- Load Testing
- Accessibility Testing *
- Usability Testing *
- Compatibility Testing*
- Unit Testing
- Integration Testing
- Acceptance Testing *
- User Interface Testing
- Regression Testing *
- Exploratory Testing *

Manchester Digital

Compatibility

- How do you know which browsers or platforms to test for?

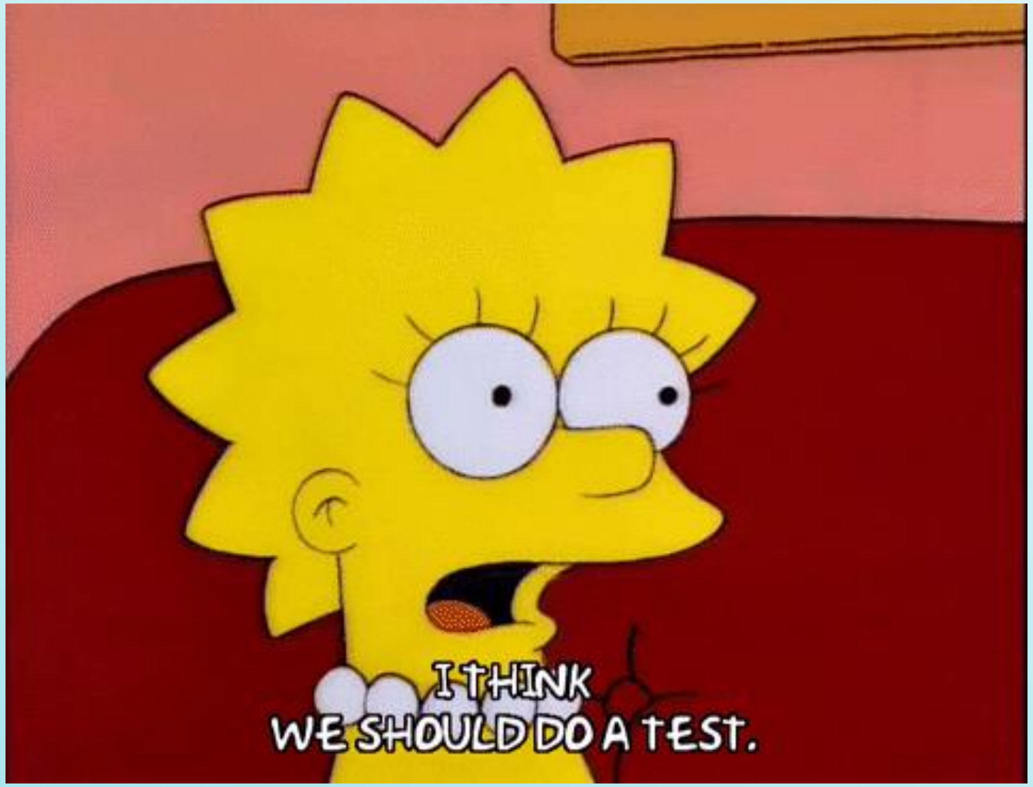
- Target audience
- Google Analytics

- i.e. Browserstack - shows you what it looks like on different browsers

- Make that more exciting, with examples

Manchester Digital

Testing the Bigger Picture

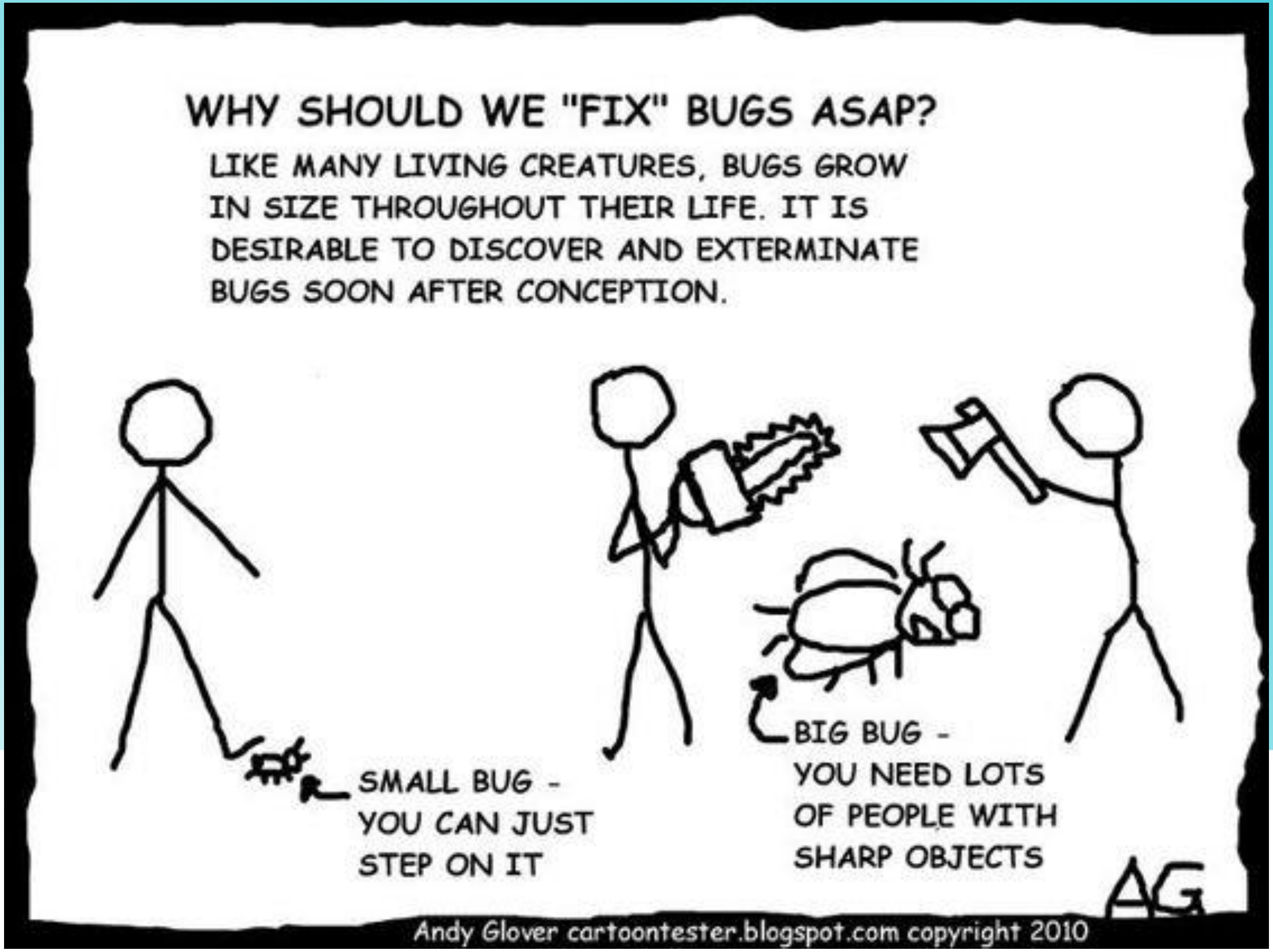


Manchester Digital

Who’s responsible for Quality

Manchester Digital

Everybody tests


WHY SHOULD WE "FIX" BUGS ASAP?
LIKE MANY LIVING CREATURES, BUGS GROW IN SIZE THROUGHOUT THEIR LIFE. IT IS DESIRABLE TO DISCOVER AND EXTERMINATE BUGS SOON AFTER CONCEPTION.
SMALL BUG - YOU CAN JUST STEP ON IT
BIG BUG - YOU NEED LOTS OF PEOPLE WITH SHARP OBJECTS
AG
Andy Glover cartoontester.blogspot.com copyright 2010

Manchester Digital

Costly bugs

- **Ariane 5** - $370m (in 1996) caused by an old bit of code left over from previous Ariane 4 rocket which tried to stuff a 64-bit variable into a 16-bit integer
- **Mars Climate Orbiter -** Quality Assurance had not found the use of an imperial unit in external software, despite the fact that NASA’s coding standards at the time mandated use of metric units
- **Therac-25** - Was designed from the outset to use software based safety systems rather than hardware controls. The removal of these hardware safety measures had tragic consequences, as race conditions in the codebase led to the death of three patients, and caused debilitating injuries to at least three other patients.

Manchester Digital

Manchester Digital

Checking vs Bug hunting

- Tests, even the same test, can be run for different reasons during a project or sprint
- Regression testing is checking, although you will still find bugs
- Exploratory testing, and to some extent Accessibility testing, is bug hunting in unusual places or unexpected workflows.

Manchester
Digital

Regression testing

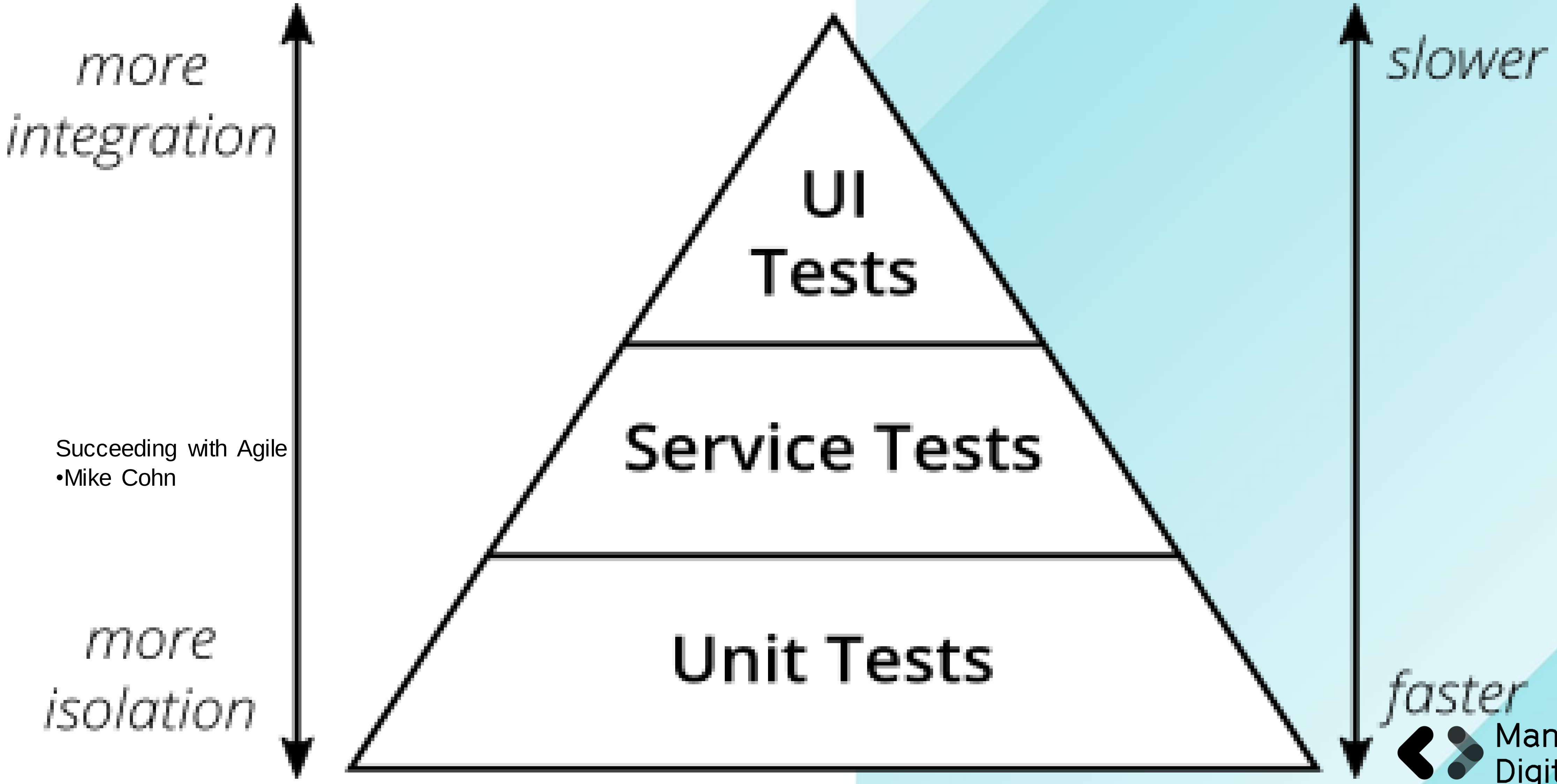
- Checking we’ve not broken anything that was already working
- When/why do regression?
  - New feature is added
  - Performance issue fix
  - Change in requirements and code is modified

Manchester Digital

Manchester Digital

Manchester
Digital

The testing pyramid



Succeeding with Agile
•Mike Cohn

Manchester Digital

Manchester
Digital

Exploratory Testing

- Unscripted testing done to explore a piece of software
- Finds bugs you didn't expect to find
- Different techniques? Charters, sessions, test days
- https://www.satisfice.com/exploratory-testing
- Disadvantages – it's hard to report pass/fail rates if you work for companies/customers who like to see statistics.

Manchester Digital

Accessibility Testing

- It's way too easy to forget about accessibility if you're 'normal'
- Ask around your office and see if anyone is colour blind or dyslexic and ask them about using some websites
- Obvious things are visually impaired or deaf users

Manchester Digital

Accessibility Testing

In a group

Breakout room in zoom 4 and a mentor

Find some websites that don’t work very well.

Manchester Digital

DAY 2

Manchester Digital

- Code Coverage
- Performance Testing
- Stress Testing
- Load Testing
- Accessibility Testing
- Usability Testing
- Compatibility Testing

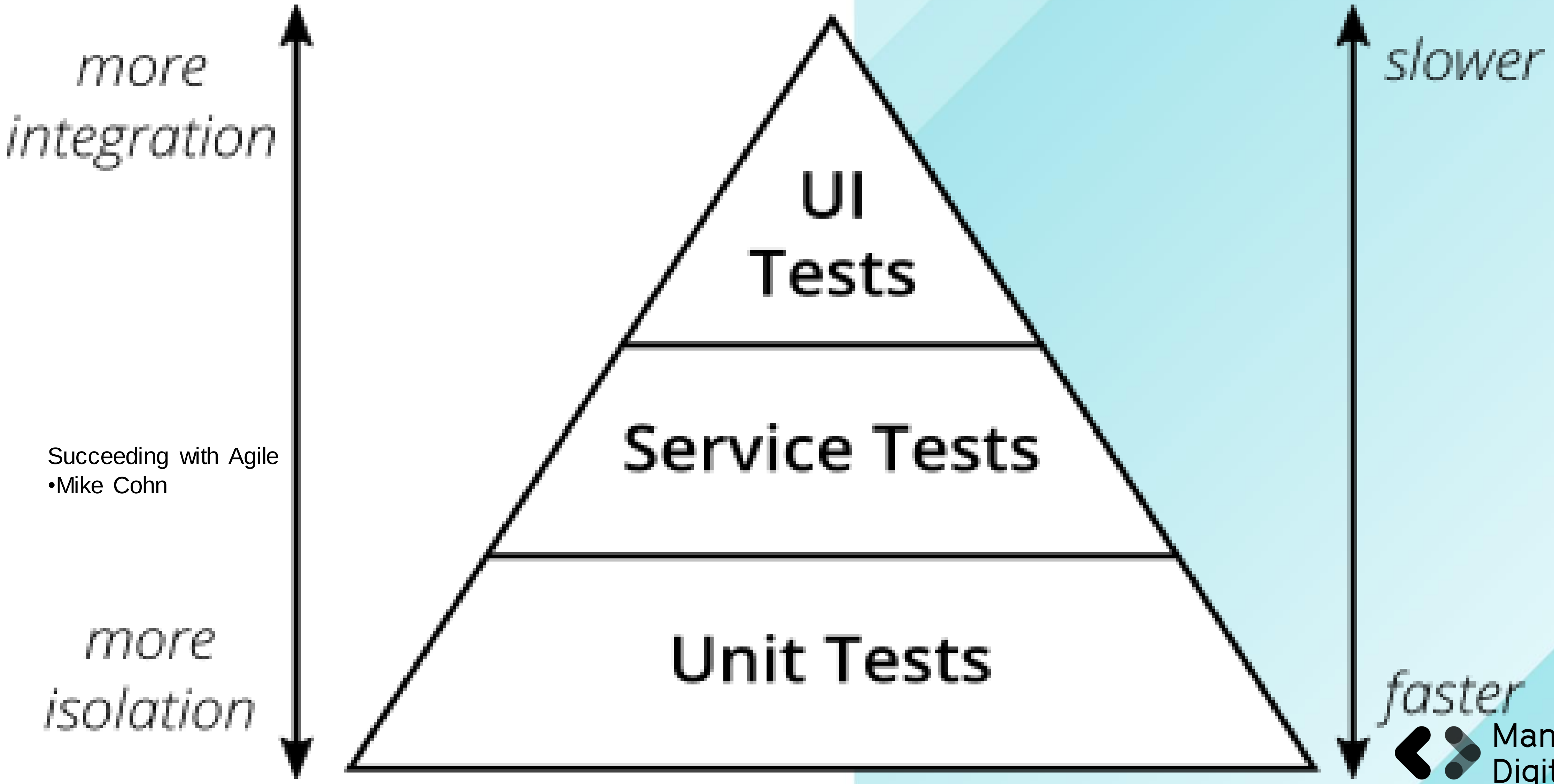
- Unit Testing
- Integration Testing
- Acceptance Testing
- User Interface Testing
- Regression Testing
- Exploratory Testing

Day 2

Introduction

Manchester
Digital

The testing pyramid



Succeeding with Agile
•Mike Cohn

Manchester Digital

RECAP WITH MARTIN

https://martinfowler.com/articles/practical-test-pyramid.html

1. Read article then take a break

Regroup and discuss examples of the different types of tests he talks about

You can look at the sample code application that goes with the article:

https://github.com/hamvocke/spring-testing

(this relies on an API that isn't accesible anymore so I'm not sure it still runs)

Manchester Digital

Test Doubles (a.k.a. mocks)

- Fakes: real implemetation with some functionality, e.g. an in-memory implementation of a database
- Stub: something that returns pre-canned responses
- Mocks: something that records the calls on a mocked object

Fakes

- Useful when building out code
- Can make tests really slow and painful: tests should run in seconds not minutes
- This can lead to people not running tests often
- Can also be unreliable: you are no longer testing a single thind and are relying on the implementation code that is not used in production

Stubs

- Simple
- Can be easy to add, without additional tooling
- Need just enough data to everything works
- Can be used to mock out a part of service yout will build later

Mocks

- Often supplied as a module you can use
- Can also do the job of stubs (and the term is used interchangeably)
- Allows you to check calls made and can be very powerful
- Can also be misused by mocking the wrong thing

- With mocks (and stubs) you need make sure you are testing something real and not just the mock

Manchester
Digital

"Only mock what you own"

- You should only mock code you control and write
- Best practice is not to mock third party libraries or language built-ins
- Doing this leads to bad design and bad tests
- It's quite common, when something in code makes it hard to test
- You tie tests to an implementation that might change
- With mocks (and stubs) you need make sure you are testing something real and not just the mock/implementation
- Mocking the wrong thing can mean you aren't doing this.
- Think you can mock a database call? What about someone accidentally drilling through the cable that connects you to a database?

Manchester Digital

"Only mock what you own"

What you should do

My code needs to get todays date. In Javascript I can write:

```
let now = new Date()
```

What happens when I want to write a test? Today is always changing.

I could mock `Date`

but what happens if it's interface changes?

What happens if I set a return value and the code is called somewhere I don't know about?

Manchester Digital

"Only mock what you own"

What you should do

Create a seam/wrapper object that you control so you can mock it.

```
function today(){return new Date()}
```

Now:

Better code: its clearer and less coupled

Easy to mock: I only need to mock my function

I'm testing against a known contract: this is actually all you can reliably do with third party libraries or services. If I use a real mock I can check and enforce this.

A practical exercise

- You are going to work on Trip Service
- This is a simulation of some of the typical problems you might encounter with legacy code.
- Your task is to make sure there are tests around it and refactor it to make as you think fit (if you have the time).
- The rules:
    - You can't change code unless there are tests around it.
    - Unless you need to change it to write tests and can do so safely i.e. using the automated refactoring tools in your IDE

A practical exercise

- The repo can be found here:
- https://github.com/sandromancuso/trip-service-kata

- The basic idea is that this represents a social network for travellers.
- If you are logged in, and are friends with a user, you can view their trips.

But first a live demo to help you get started.

After this work in your groups there is code for lots of different languages.

Manchester Digital

Testing your front end with automated tests

- Too often testing a front end relies on manual testing
    - Someone goes through a web site manually and makes sures it looks good
    - This is inefficient and takes a lot of time so less used parts don't get tested

- Front end testing can be automated
    - It can be painful and unreliable
    - People don't like doing it

- Selenium is a common tool but can be tricky
    - We are going to take a look at Selenim IDE which can make it easier(-ish)

Manchester Digital

Selenium IDE

- It's a browser extension you can find here:
- https://www.selenium.dev/selenium-ide/

- First a quick demo

- You can practice this using the sites listed here:
- https://www.techbeamers.com/websites-to-practice-selenium-webdriver-online/

Manchester Digital

What to do now

- Have a go at making some tests using Selenium IDE
- Finish the exercise from this morning if you haven't done so
  - If you have got test coverage, look at some refactoring to make it a nice experience
- Do some Katas (as a mob).
  - You can find some here: https://kata-log.rocks/starter
  - Try the classic Gilded Rose:
    - https://github.com/emilybache/GildedRose-Refactoring-Kata/tree/master/python
    - You will need to learn about the Golden Master technique. ask your mentor or me
    - https://understandlegacycode.com/blog/3-steps-to-add-tests-on-existing-code-when-you-have-short-deadlines/

Manchester Digital

Homework

In your company

- Spend a day with a tester.
- How do you mock objects?
- How do you do compatibility testing?
- How do you work out which browsers or mobile devices to be compatible for?
- How do tests fit into Continuous Integration / Delivery?
- Try some mutation testing - can you keep the test suite passing?

Manchester Digital