You can’t test this?! Hammertime!

...

Frances Buontempo
and
Steve Love

You can’t test this

…

Break it down...

You can’t test this

- You can’t run the whole env on your machine
- You aren’t allowed to see real data
- You’re just a dev
- I’m a BA and can’t run the code til it’s in UAT
- The team used to test but it held up a release
- You have code to write, the testers will test it for you later

You **can't** test this

- The code uses other parts that don't exist yet
  - Code versions and DB schema or library changes go hand-in-hand
- I haven't finished it yet
  - You ever tasted food you are cooking before it's done?
  - Or maths proofs – lemmas first then the main theorem (then corollaries)
- Random numbers, so does something different every time
- Embedded software, I don't have the hardware yet
- Button/observer handlers, all in 'main'
- Structure code to be testable

Redefine “random”

So lie about random

Write a “random” provider to

Return a zero (or one)

Corollary: Never trust a singleton.

```
void some_function(int x)
{
    int shifted = x + 2;
    std::cout << shifted * std::rand();
}

int main()
{
    std::srand(std::time(0));
    some_function(10);
}
```


You can't **test** this

- You don't know what it's supposed to do yet
- It writes to the screen
- Formal proofs?
  - Do you need to be *certain* of *everything*?
  - Is it OK to be nearly sure?
  - Is "some" better than "none"?
- Some things are *hard* to replicate in test
  - Multi-user / multi-threaded
  - Environmental problems - servers going down, trees falling on the line

Test: clay pot for assaying c.f. crucible

@fbuontempo @IAmSteveLove

Redefine “screen”

Return a string/stream

(or send one in)

Corollary: Never trust a singleton.

You can't test **this**

- Database procedures, schema updates
- It's just a script
- Interaction with 3rd party library/software
  - Which isn't written yet
- API design
  - Microservices (PACT etc)
- It's a fail-over script
  - And I only have one machine!
- Changes over time
  - Tomorrow is 24 hours away so any test takes 24 hours

Time is relative

So lie about DateTime.now();

Send in a time provider

Corollary: Never trust a singleton.

Questions, so many questions

Why do we test?

Dispell Doubt	Confidence for you? Your customers?
	What do you mean by confidence?
	No regressions e.g. seagull merge
Discussion	With business people, your team...
	Concert examples are easier to point at/reason about
Documentation	What it does, why e.g. happy path, edge cases
	How to use it
Discovery	What it does do?
	Is it slow?

MINE

Mine?

DIY.DESPAIR.COM

What do we test?

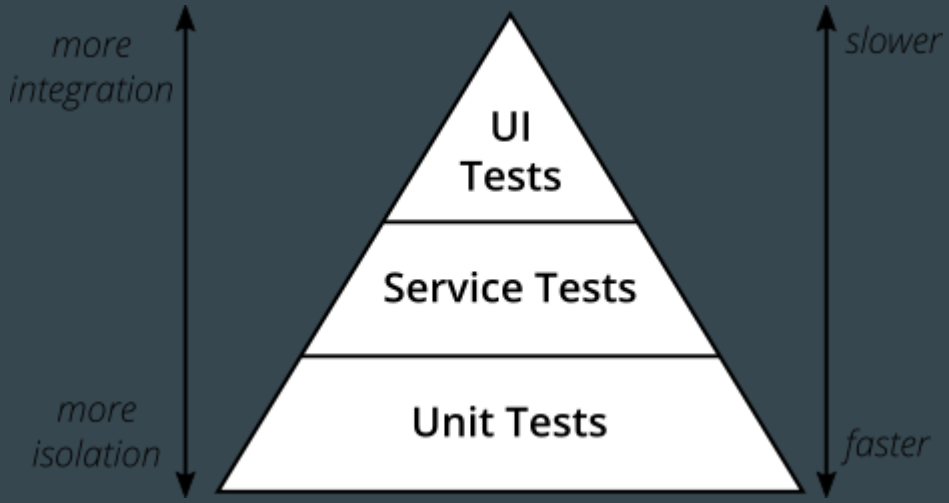
- A developer's patience?
- Behaviour?
- Load?
- Database?
- UI?
- Nothing broke?
- Coverage?
- Performance?
- Code?
- Endpoints/URLs/REST gateways?

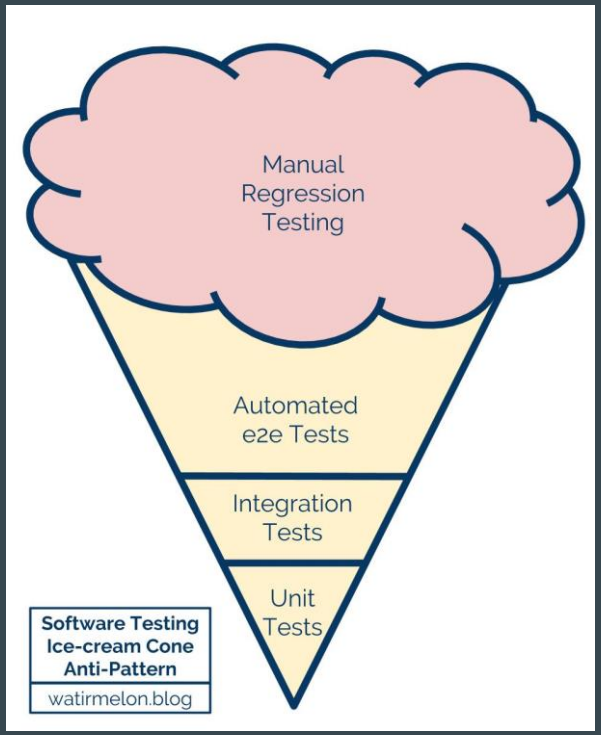
Test ALL the things!

How do we test?

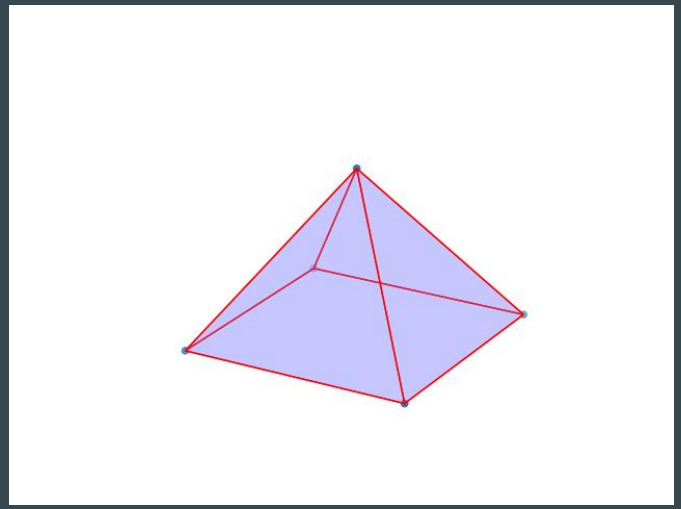
- End to end
- “Integration”
- “Units”
- Pen testing
- Automatically (CI etc)
- With a testing framework
- In the language it was written in?
- Inside out/outside in
- TDD
- BDD
- Chaos monkey
- Manually
- With runtime asserts
- Any language you please?

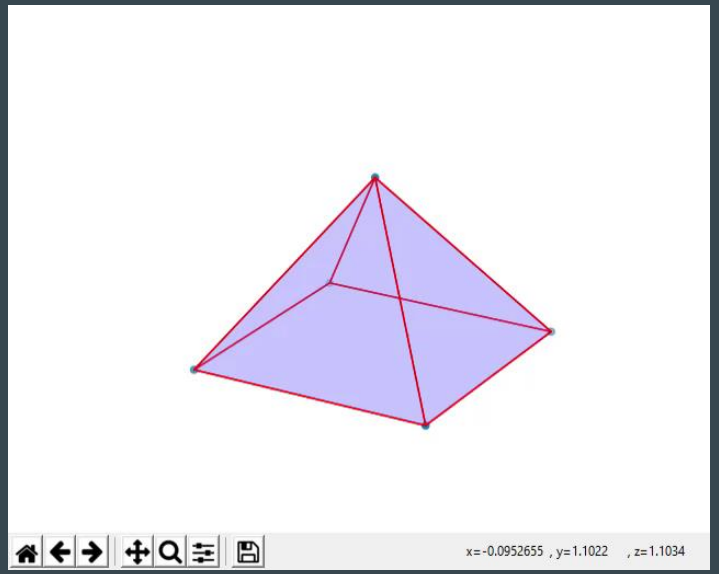
Caveat: Testing “pyramid”/ice cream cone





No ice cream – unstable!





When do we test this?

- When it's released?
  - Or the weekend/day/an hour before
- When we've finished coding?
- Before each commit?
- Before we code?
- As we code?
- When we get a turn on UAT, after we've committed the code, when we get a turn, ...

Test early

Test often

Where do we test?

- Only on my machine, on CI, in prod/the cloud?
- Only in UAT?
- Only using the official test framework?
- In your head
  - Go for a walk and think about what might not work
- And by the way - did you release what you actually tested?

Whose responsibility is it anyway?

- QA team/testers
- Your team-mates
- Your customers/users?
- Or....you?

Having someone else test your stuff has its benefits...but can you be sure it'll happen? What about WHEN it'll get done? And don't get it thrown straight back at you because it crashes/won't start....

Excuses

@fbuontempo @IAmSteveLove

I did test this, but

- It took hours
- What it's supposed to do under these circumstances?
- The tests failed
- I have no idea what went wrong
- I can't debug the tests
- People using the software hate it because
  - It doesn't do what they want
  - It's slow
  - It's the wrong colour

But I did test this

Beware words. If a data scientist says they tested their code ask what they did. If they talk about accuracy, you need to talk about the difference between

evaluating a model

and

testing code

But I’m replicating the code in the test

- E.g. Fizz Buzz
  - Talk to Kevlin!
- Think about the properties you need to always hold

But we don’t have real data yet!

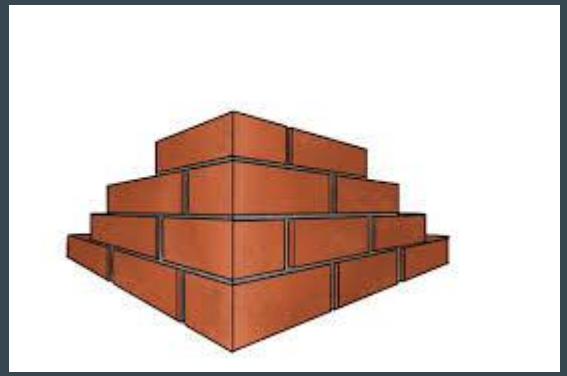
- Why do people insist on real data?
  - “Normal usage more important than edge cases”
- Will that give you edge/corner cases?
- What are you trying to test?

The numbers are wrong

ProTip: Send in zeros and check for discontinuities

A True Story

"I'm not an engineer but I don't understand how you can test something you haven't built yet."



But I don’t have an environment to test in!

- UAT
  - is broken
  - being used by someone else
- I don’t have a dev
  - db,
  - message queue,
  - enough licences

It’s all going wrong

Help! I’m debugging a test!

- Failing test first => good messages
- Learn to drive your test framework
  - Find out what fail messages look like

It takes too long

- Elastic Search: 72 hours ish to run them all
- Kon Mari the tests once in a while
- It's OK to separate fast tests (that run all the time) from slow tests (that are kicked manually late on a Friday)
- Measure coverage
  - Of code and requirements
  - Are all your tests pulling their weight?
- Using dates and times takes ages
  - The differences between today and tomorrow take 24 hours

But how do I do this?

Good fail messages...

- Compare false != true is NOT a good message
- Actual v expected backwards
    - Order varies between frameworks
- By the way, “Null argument exception” is the right message for an unexpected null argument

Assert true

```
import unittest
from fizz_buzz import *

class FizzBuzzTests(unittest.TestCase):
    def test_three_is_fizz(self):
        self.assertTrue(fizz_buzz(3) == "fizz")
```


```
FAIL: test_three_is_fizz (test_fizz_buzz.FizzBuzzTests)
----------------------------------------------------------------------
Traceback (most recent call last):
  File "/home/fran/spiel/hammer/test_fizz_buzz.py", line 6, in
test_three_is_fizz
        self.assertTrue(fizz_buzz(3) == "fizz")
AssertionError: False is not true
```


Assert equal

```
import unittest
from fizz_buzz import *

class FizzBuzzTests(unittest.TestCase):
    def test_three_is_fizz(self):
        self.assertEqual(fizz_buzz(3), "fizz")
```


FAIL: test_three_is_fizz (test_fizz_buzz.FizzBuzzTests)

Traceback (most recent call last):

File "/home/fran/spiel/hammer/test_fizz_buzz.py", line 6, in test_three_is_fizz

self.assertEqual(fizz_buzz(3), "fizz")

AssertionError: 3 != 'fizz'

Expected but was: order matters

```
from hamcrest import *
import unittest

class TestStuff(unittest.Testcase):
    def test_five_is_buzz(self):
        assert_that('buzz', equal_to(5))

AssertionError:
Expected: <5>
     but: was 'buzz'
```


Read the docs

- Try out the ways of expressing a message
- Can you write custom messages in your framework?
  - Do you need to?
- Learn how to drive the tools you are using
- Tl; dr; **Write a failing test first**
  - See what the message says

Mocks, stubs and stunt doubles

Fake it til they make it

Does the coverage hit the code or just the mocks?

Maybe try some mutation testing

...

How do you test std::midpoint?

- The simple problem of computing a value between two other values is surprisingly subtle in general.
  - https://www.youtube.com/watch?v=sBtAGxBh-XI
  - http://www.open-std.org/jtc1/sc22/wg21/docs/papers/2019/p0811r3.html

Possible problems? Overflow? It's ok to look at code and list what might go wrong

How do you test std::midpoint?

```
template <class Integer>
constexpr Integer midpoint0(Integer a,
                            Integer b) noexcept
{
    return (a + b)/2;
}
```

- The overflow is in Java's binary search (or was for years)
- Code has bugs, but you can find them and write tests for them

Midpoint tests....

```
template <class Integer>
constexpr Integer midpoint0(Integer a,
Integer b) noexcept
{
    return a+((b-a)/2);
}
```


Midpoint tests....

- Also overflows! (for some inputs e.g. a + billion, b - billion). So add a new test and continue.
- Also, what's the midpoint (integer) between 0 and 1?

Even more midpoint tests

```
template <class Integer>
constexpr Integer midpoint1(Integer a,
Integer b) noexcept
{
    using U = std::make_unsigned_t<Integer>
    return Integer(U(a)+(U(b)-U(a))/2);
}
```


Even more midpoint tests

```
midpoint(1, 6) = 3
midpoint(6, 1) == -2147483645
```

These were just ints. You should see what happens with a signed char. After many refinements, we got there. See Marshall's talk.

Midpoint ...

- You don’t have to be a TDD purist
- It’s OK to add new tests for edge cases as you find them
  - Does midpoint(1, 4) == midpoint(4, 1)?
- Use tests to demonstrate options, e.g. for edge cases, and use these to start a discussion

How do you test machine learning?

Jupyter/Data bricks etc note books

- The structure is hard to test
- One long function
- Randomness inside
- Results are a often a graph

Rant!

1. Approval tests for graphs (and other outputs)
2. Keep the “code” in the notebook minimal.
3. Write functions.
4. Call functions you have in modules that you test.
5. Deploy sanely (and in a repeatable fashion)
6. Use version control

How do you test code with random numbers in?

- Seeds, but that's one path
- Try using 0 (or 1)
- Averages?
- Property based testing

How do you test code with dates and times in

- A date provider means you can move forward a day in one line of code
  - Rather than waiting 24 hours
- Never trust a singleton.

One more thing...

- Round-tripping stuff
  - serialise/deserialise (not using files)
  - See https://stackoverflow.com/questions/24299692/why-is-a-round-trip-conversion-via-a-string-not-safe-for-a-double (and some recent bug fix in C++ - see Nico' book)

Hammertime....

@fbuontempo @IAmSteveLove

Fuzzers

- Try random stuff!

Property based testing

- A type of fuzzer?

Mutation testing

- Cosmic ray

Machine Learning

Can your “machine” learn to test the code?

Pexx /Moles

Most test-generation frameworks are rubbish, by the way

Acceptance testing

- BDD
- Gilded rose kata
- Record what it does now FTW
- Approval testing

Be your own chaos monkey

- Kill processes
- Unplug cables
- Send in nonsense

Think like a tester!

Coverage

- Are you testing the same thing over and over
- Code coverage – lots of tools
  - Requirements coverage – BDD???? Ask the BAs
- Are your tests pulling their weight?
  - All of them?

TTF

- Time to feedback
- Make it quicker!

You got this

- How do we test?
  - Break it down
  - Test structure
  - Use tests to drive discussions and communication
- Where do we test?
  - In as many environments as we can afford
  - Use your imagination. No really. Like just think about this stuff!
- What do we test?
  - What's the worst thing that could happen?
  - What could possibly go wrong?
  - Think like a tester!

Make your life easier

- Why do we test?
  - So it doesn't go wrong in front of the customer is a good answer to that
  - So I don't get called at 3am because it crashed
  - So I can deploy the code at any time, including during the day, and be confident it'll just work
  - So I can go to the pub/home/gym (Damn you Covid!)

You can test this!
Hammertime!
...