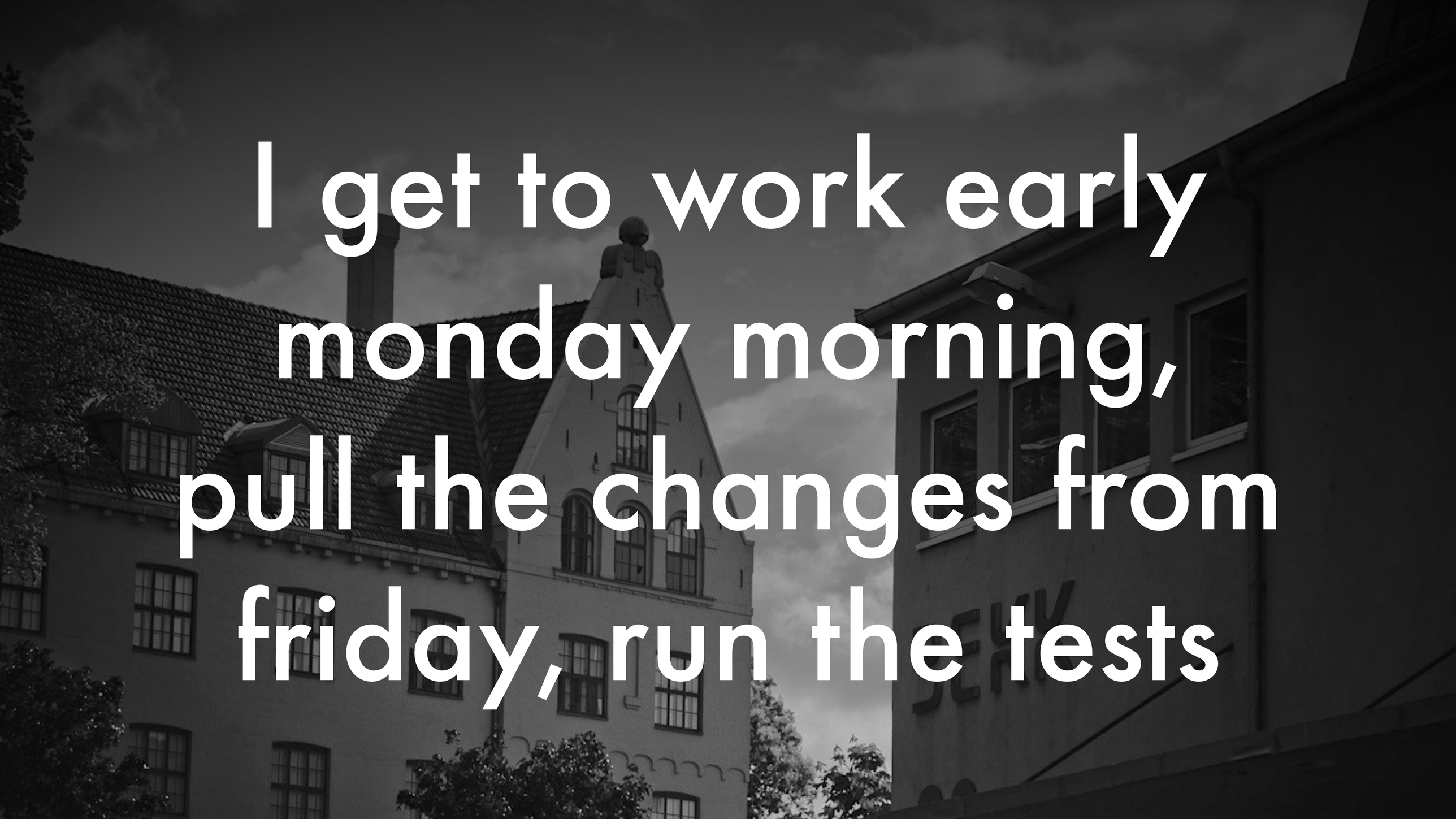
WRITING BEAUTIFUL JAVASCRIPT TESTS

@KIMJOAR

#exampletime
BEKK


I get to work early
monday morning,
pull the changes from
friday, run the tests

Jasmine Spec Runner v2.0

localhost:8000/SpecRunner.html

Jasmine 2.0.2

finished in 0.037s

55 specs, 1 failure

raise exceptions

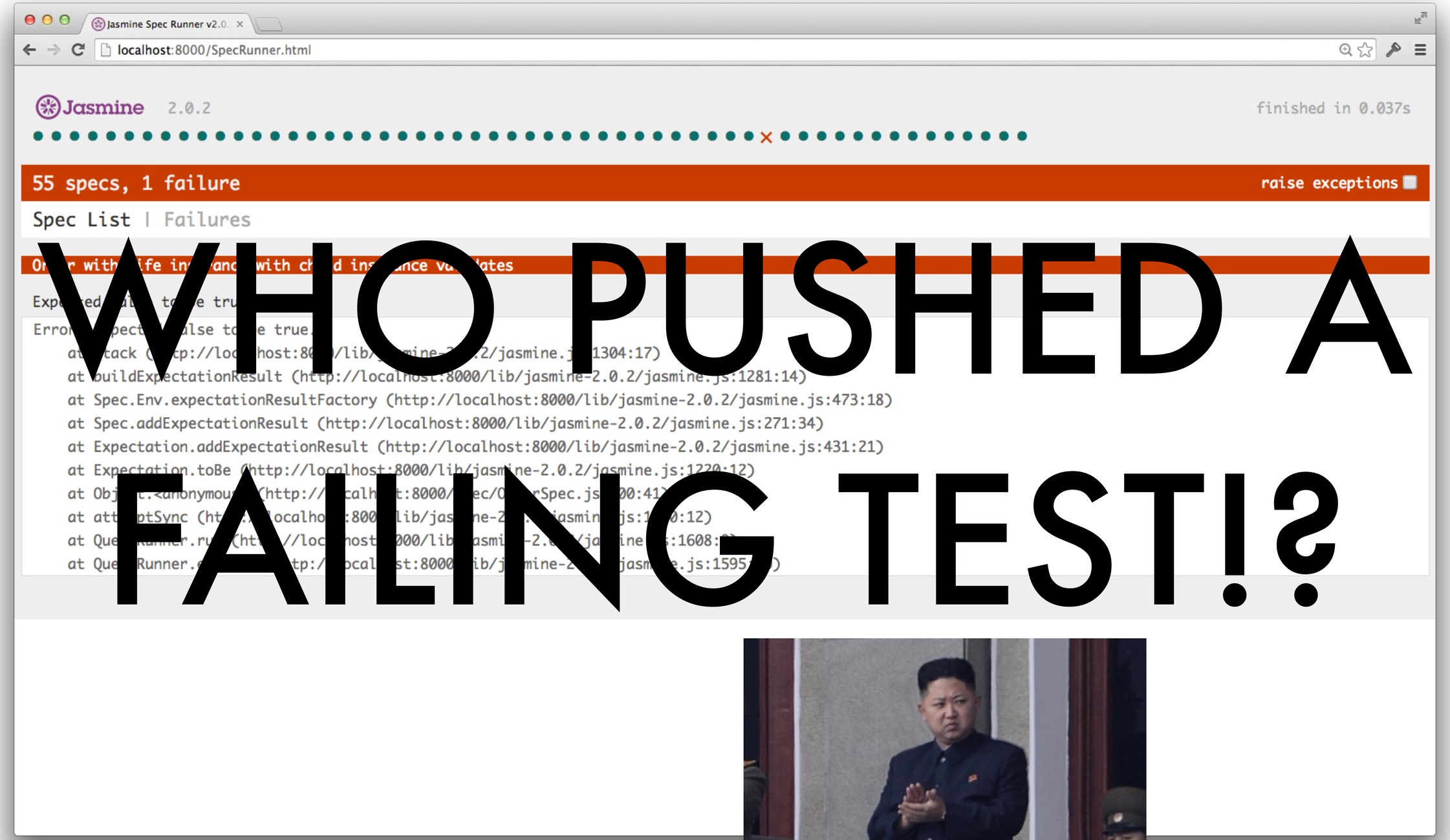
Spec List | Failures

Order with life insurance with child insurance validates

Expected false to be true.

```
Error: Expected false to be true.
    at stack (http://localhost:8000/lib/jasmine-2.0.2/jasmine.js:1304:17)
    at buildExpectationResult (http://localhost:8000/lib/jasmine-2.0.2/jasmine.js:1281:14)
    at Spec.Env.expectationResultFactory (http://localhost:8000/lib/jasmine-2.0.2/jasmine.js:473:18)
    at Spec.addExpectationResult (http://localhost:8000/lib/jasmine-2.0.2/jasmine.js:271:34)
    at Expectation.addExpectationResult (http://localhost:8000/lib/jasmine-2.0.2/jasmine.js:431:21)
    at Expectation.toBe (http://localhost:8000/lib/jasmine-2.0.2/jasmine.js:1220:12)
    at Object.<anonymous> (http://localhost:8000/spec/OrderSpec.js:200:41)
    at attemptSync (http://localhost:8000/lib/jasmine-2.0.2/jasmine.js:1620:12)
    at QueueRunner.run (http://localhost:8000/lib/jasmine-2.0.2/jasmine.js:1608:9)
    at QueueRunner.execute (http://localhost:8000/lib/jasmine-2.0.2/jasmine.js:1595:10)
```

WHO PUSHED A FAILING TEST!?



, however, is green

, however, is green

EXAMPLE OF WHAT A TEST MIGHT LOOK LIKE

```
describe('cart', function() {
    beforeEach(function() {
        // called before every test
        // (often called setup)
    });

    it('calculates correct total price', function() {
        var cart = new Cart();

        cart.addItem({ name: 'Sleeping bag', price: 299, quantity: 2 });
        cart.addItem({ name: 'Tent', price: 499, quantity: 1 });

        expect(cart.totalPrice()).toEqual(1097);
    });

    afterEach(function() {
        // called after every test
        // (often called teardown)
    });
});
```

EXAMPLE OF WHAT A TEST MIGHT LOOK LIKE

```
describe('cart', function() {
    var cart;

    beforeEach(function() {
        cart = new Cart();
    });

    describe('with two orders', function() {
        beforeEach(function() {
            cart.addItem({ name: 'Sleeping bag', price: 299, quantity: 2 });
            cart.addItem({ name: 'Tent', price: 499, quantity: 1 });
        });

        it('calculates correct total price', function() {
            expect(cart.totalPrice()).toEqual(1097);
        });
    });
});
```

Jasmine Spec Runner v2.0

localhost:8000/SpecRunner.html

Jasmine 2.0.2

finished in 0.037s

55 specs, 1 failure

raise exceptions

Spec List | Failures

Order with life insurance with child insurance validates

Expected false to be true.

```
Error: Expected false to be true.
    at stack (http://localhost:8000/lib/jasmine-2.0.2/jasmine.js:1304:17)
    at buildExpectationResult (http://localhost:8000/lib/jasmine-2.0.2/jasmine.js:1281:14)
    at Spec.Env.expectationResultFactory (http://localhost:8000/lib/jasmine-2.0.2/jasmine.js:473:18)
    at Spec.addExpectationResult (http://localhost:8000/lib/jasmine-2.0.2/jasmine.js:271:34)
    at Expectation.addExpectationResult (http://localhost:8000/lib/jasmine-2.0.2/jasmine.js:431:21)
    at Expectation.toBe (http://localhost:8000/lib/jasmine-2.0.2/jasmine.js:1220:12)
    at Object.<anonymous> (http://localhost:8000/spec/OrderSpec.js:200:41)
    at attemptSync (http://localhost:8000/lib/jasmine-2.0.2/jasmine.js:1620:12)
    at QueueRunner.run (http://localhost:8000/lib/jasmine-2.0.2/jasmine.js:1608:9)
    at QueueRunner.execute (http://localhost:8000/lib/jasmine-2.0.2/jasmine.js:1595:10)
```

Jasmine Spec Runner v2.0

localhost:8000/SpecRunner.html

Jasmine 2.0.2

finished in 0.037s

55 specs, 1 failure

raise exceptions

Spec List | Failures

Order with life insurance with child insurance validates

Expected false to be true.

```
Error: Expected false to be true.
    at stack (http://localhost:8000/lib/jasmine-2.0.2/jasmine.js:1304:17)
    at buildExpectationResult (http://localhost:8000/lib/jasmine-2.0.2/jasmine.js:1281:14)
    at Spec.Env.expectationResultFactory (http://localhost:8000/lib/jasmine-2.0.2/jasmine.js:473:18)
    at Spec.addExpectationResult (http://localhost:8000/lib/jasmine-2.0.2/jasmine.js:271:34)
    at Expectation.addExpectationResult (http://localhost:8000/lib/jasmine-2.0.2/jasmine.js:431:21)
    at Expectation.toBe (http://localhost:8000/lib/jasmine-2.0.2/jasmine.js:1220:12)
    at Object.<anonymous> (http://localhost:8000/spec/OrderSpec.js:200:41)
    at attemptSync (http://localhost:8000/lib/jasmine-2.0.2/jasmine.js:1620:12)
    at QueueRunner.run (http://localhost:8000/lib/jasmine-2.0.2/jasmine.js:1608:9)
    at QueueRunner.execute (http://localhost:8000/lib/jasmine-2.0.2/jasmine.js:1595:10)
```

Jasmine Spec Runner v2.0

localhost:8000/SpecRunner.html

Jasmine 2.0.2

finished in 0.037s

55 specs, 1 failure

raise exceptions

Spec List | Failures

Order with life insurance with child insurance validates

Expected false to be true.

```
Error: Expected false to be true.
    at stack (http://localhost:8000/lib/jasmine-2.0.2/jasmine.js:1304:17)
    at buildExpectationResult (http://localhost:8000/lib/jasmine-2.0.2/jasmine.js:1281:14)
    at Spec.Env.expectationResultFactory (http://localhost:8000/lib/jasmine-2.0.2/jasmine.js:473:18)
    at Spec.addExpectationResult (http://localhost:8000/lib/jasmine-2.0.2/jasmine.js:271:34)
    at Expectation.addExpectationResult (http://localhost:8000/lib/jasmine-2.0.2/jasmine.js:431:21)
    at Expectation.toBe (http://localhost:8000/lib/jasmine-2.0.2/jasmine.js:1220:12)
    at Object.<anonymous> (http://localhost:8000/spec/OrderSpec.js:200:41)
    at attemptSync (http://localhost:8000/lib/jasmine-2.0.2/jasmine.js:1620:12)
    at QueueRunner.run (http://localhost:8000/lib/jasmine-2.0.2/jasmine.js:1608:9)
    at QueueRunner.execute (http://localhost:8000/lib/jasmine-2.0.2/jasmine.js:1595:10)
```

Jasmine Spec Runner v2.0

localhost:8000/SpecRunner.html

Jasmine 2.0.2

finished in 0.037s

55 specs, 1 failure

raise exceptions

Spec List | Failures

Order with life insurance with child insurance validates

Expected false to be true.

```
Error: Expected false to be true.
    at stack (http://localhost:8000/lib/jasmine-2.0.2/jasmine.js:1304:17)
    at buildExpectationResult (http://localhost:8000/lib/jasmine-2.0.2/jasmine.js:1281:14)
    at Spec.Env.expectationResultFactory (http://localhost:8000/lib/jasmine-2.0.2/jasmine.js:473:18)
    at Spec.addExpectationResult (http://localhost:8000/lib/jasmine-2.0.2/jasmine.js:271:34)
    at Expectation.addExpectationResult (http://localhost:8000/lib/jasmine-2.0.2/jasmine.js:431:21)
    at Expectation.toBe (http://localhost:8000/lib/jasmine-2.0.2/jasmine.js:1220:12)
    at Object.<anonymous> (http://localhost:8000/spec/OrderSpec.js:200:41)
    at attemptSync (http://localhost:8000/lib/jasmine-2.0.2/jasmine.js:1620:12)
    at QueueRunner.run (http://localhost:8000/lib/jasmine-2.0.2/jasmine.js:1608:9)
    at QueueRunner.execute (http://localhost:8000/lib/jasmine-2.0.2/jasmine.js:1595:10)
```

```
it('validates', function() {
    expect(order.isValid()).toBe(true);
});
```

WHY DID IT FAIL?

WHY DID IT FAIL?

```
beforeEach(function() {
    var ci = new Product({
        type: 'CI',
        price: 105
    });
    cart.addItem(ci);
});
```

CHILD INSURANCE

DOES THIS CHANGE THE ORDER?

```
beforeEach(function() {
    beneficiary1.amount = 1500000;
    beneficiary2.amount = 500000;

    var li = new Product({
        type: 'LI',
        price: 399,
        amount: beneficiary1.amount
            + beneficiary2.amount
    });

    cart.addItem(li);
    order.setPolicyHolder(policyHolder2);
    order.addBeneficiary(beneficiary1);
    order.addBeneficiary(beneficiary2);
});
```

WHY DID IT FAIL?

```
beforeEach(function() {
    policyHolder1 = new PolicyHolder({
        name: "Ola Nordmann",
        ssn: 07099451429,
        telephoneNumber: "+4795732501",
        email: "me@example.org",
        address: {
            street: "Toftes gate 17",
            postCode: 0556,
            postalArea: "Oslo"
        }
    });
    policyHolder2 = new PolicyHolder({
        name: "Testy Testeson",
        ssn: 17099926629,
        telephoneNumber: "+4743032501",
        email: "me2@example.org",
        address: {
            street: "Ravnkollbakken 3",
            postCode: 0970,
            postalArea: "Oslo"
        }
    });

    beneficiary1 = {
        name: "Testy2",
        ssn: "04037335466"
    }
    beneficiary2 = {
        name: "Testy3",
        ssn: "18049938744"
    }
    beneficiary3 = {
        name: "Testy4",
        ssn: "21050682312"
    }

    cart = new Cart();

    order = new Order({
        cart: cart,
        policyHolder: policyHolder1,
        withdrawalDay: 15
    });
});
```

I HAVE NO IDEA

```
beforeEach(function() {
    // ...

    policyHolder2 = new PolicyHolder({
        name: "Testy Testeson",
        ssn: 17099926629,
        telephoneNumber: "+4743032501",
        email: "me2@example.org",
        address: {
            street: "Ravnkollbakken 3",
            postCode: 0970,
            postalArea: "Oslo"
        }
    });

    // ...
});
```

THE PROBLEM?

He got too old to have child insurance

You will,
at some point,
HATE your tests

We need to
learn more
about
writing tests

WHY DO
WE TEST?

because testing in production sucks

But seriously,

IT DEPENDS

HI! I'M @KIMJOAR

I'M A CONSULTANT AT BEKK IN OSLO, NORWAY
I WORK IN 5-10 PERSON TEAMS
FOR 3 MONTHS TO SEVERAL YEARS
FOR LARGE COMPANIES
ON INTERACTIVE AND COMPLEX APPS

TODAY:

PART PHILOSOPHY

PART TECHNIQUES

SHARING EXPERIENCES

BOTH GOOD AND BAD

WHY DO
I TEST?

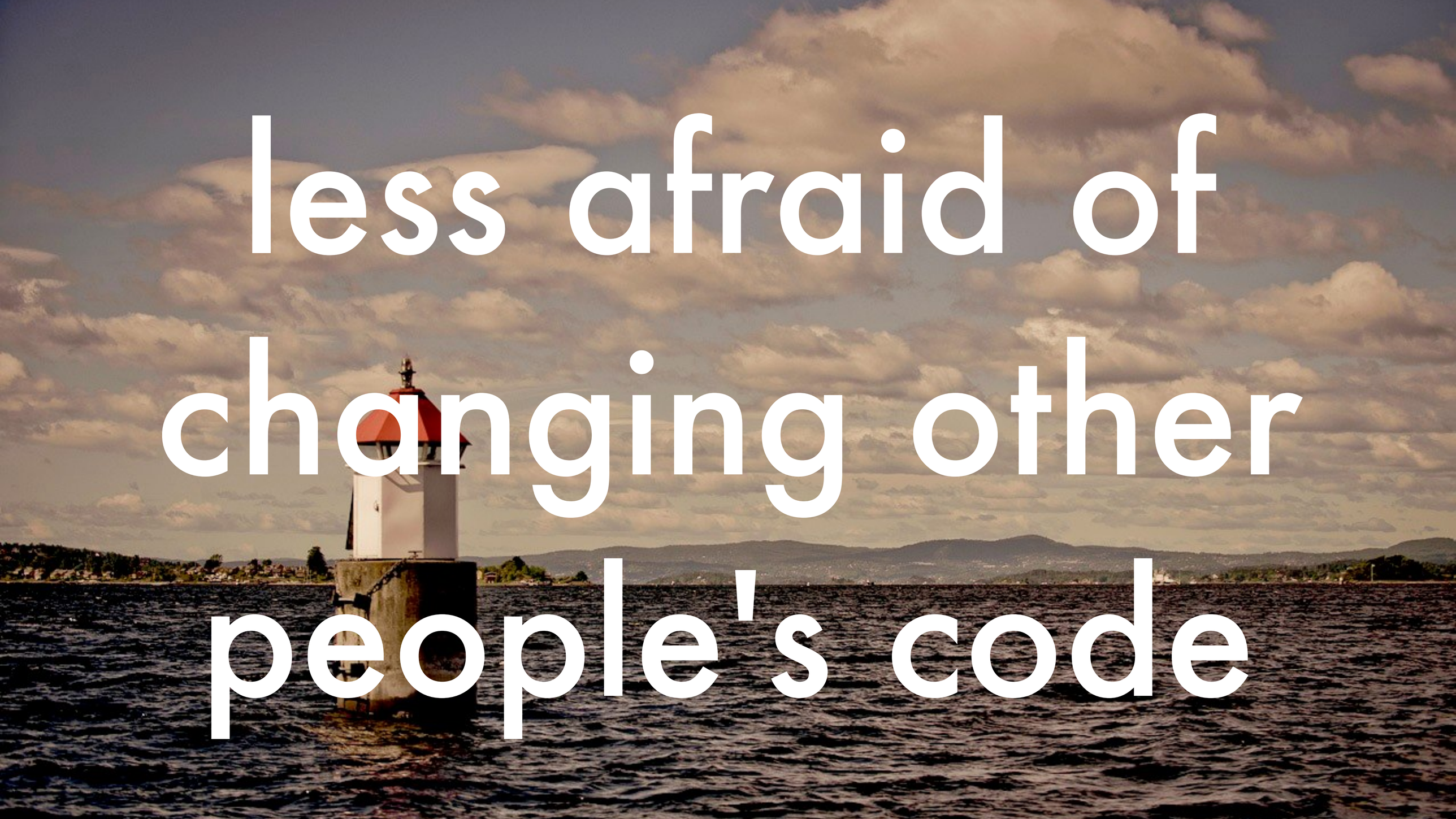
REGRESSIONS
COMMUNICATION
DESIGN

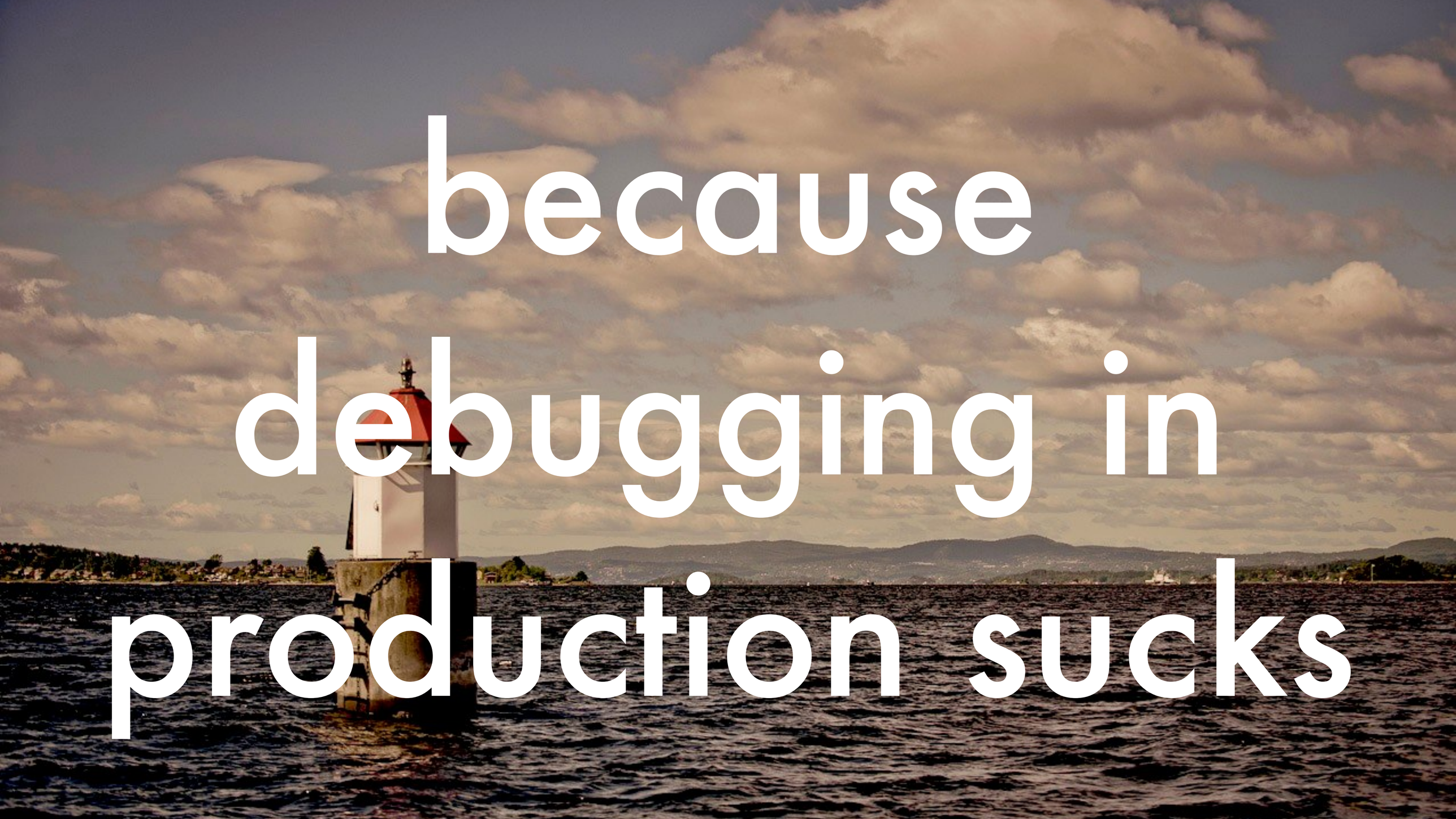
REGRESSIONS

we're building
more complex and
interactive apps

often in
larger teams

less afraid of
changing code

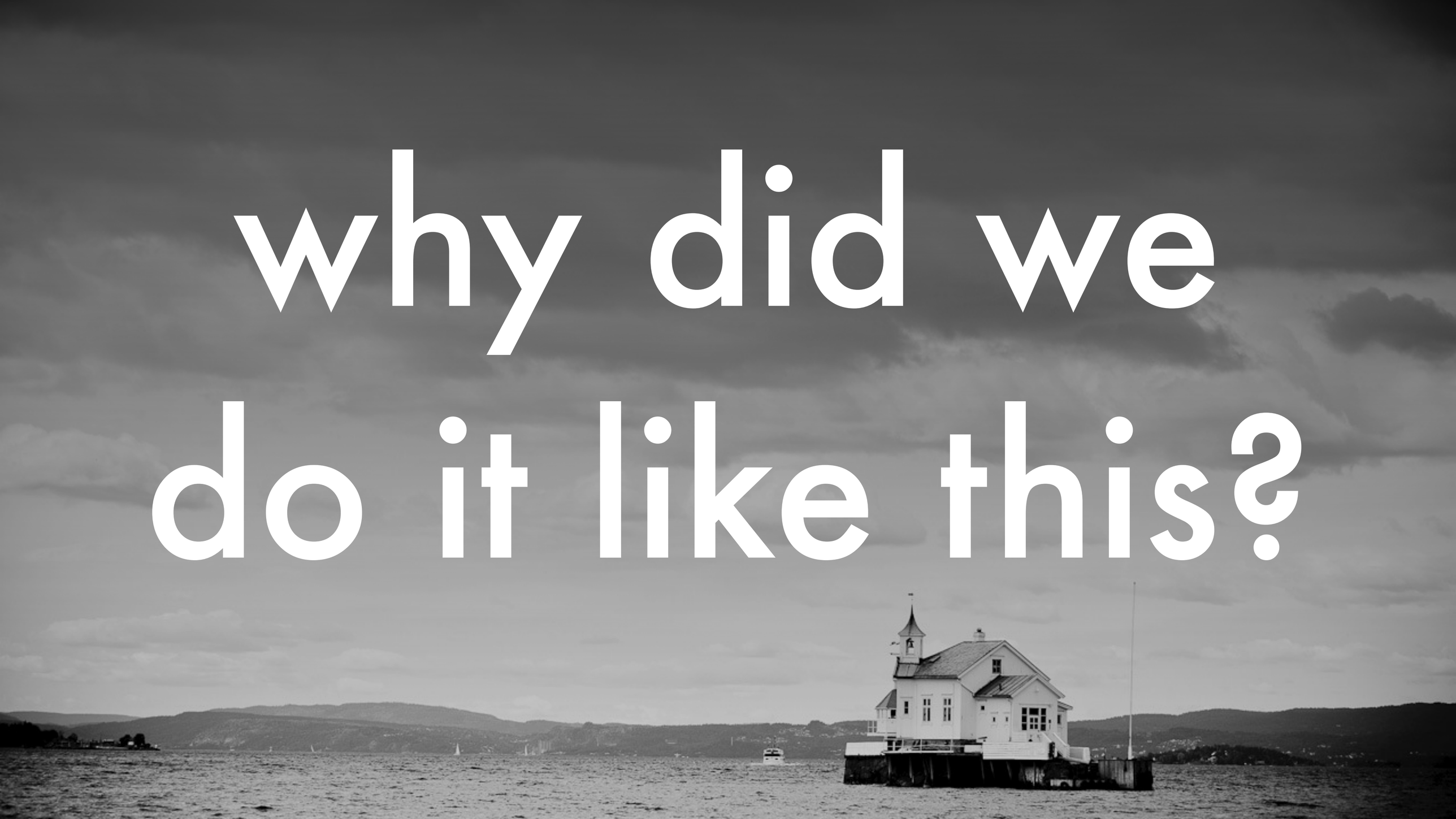

less afraid of
changing other
people's code


because
debugging in
production sucks

fix a bug once

COMMUNICATION

why did we do it like this?



what should it do?

DESIGN

immediate
feedback

keep the
code simple

REGRESSIONS COMMUNICATION DESIGN

so .... what's the problem?

it's easy to write really bad tests

TRY TO
HOW I WRITE
MY TESTS

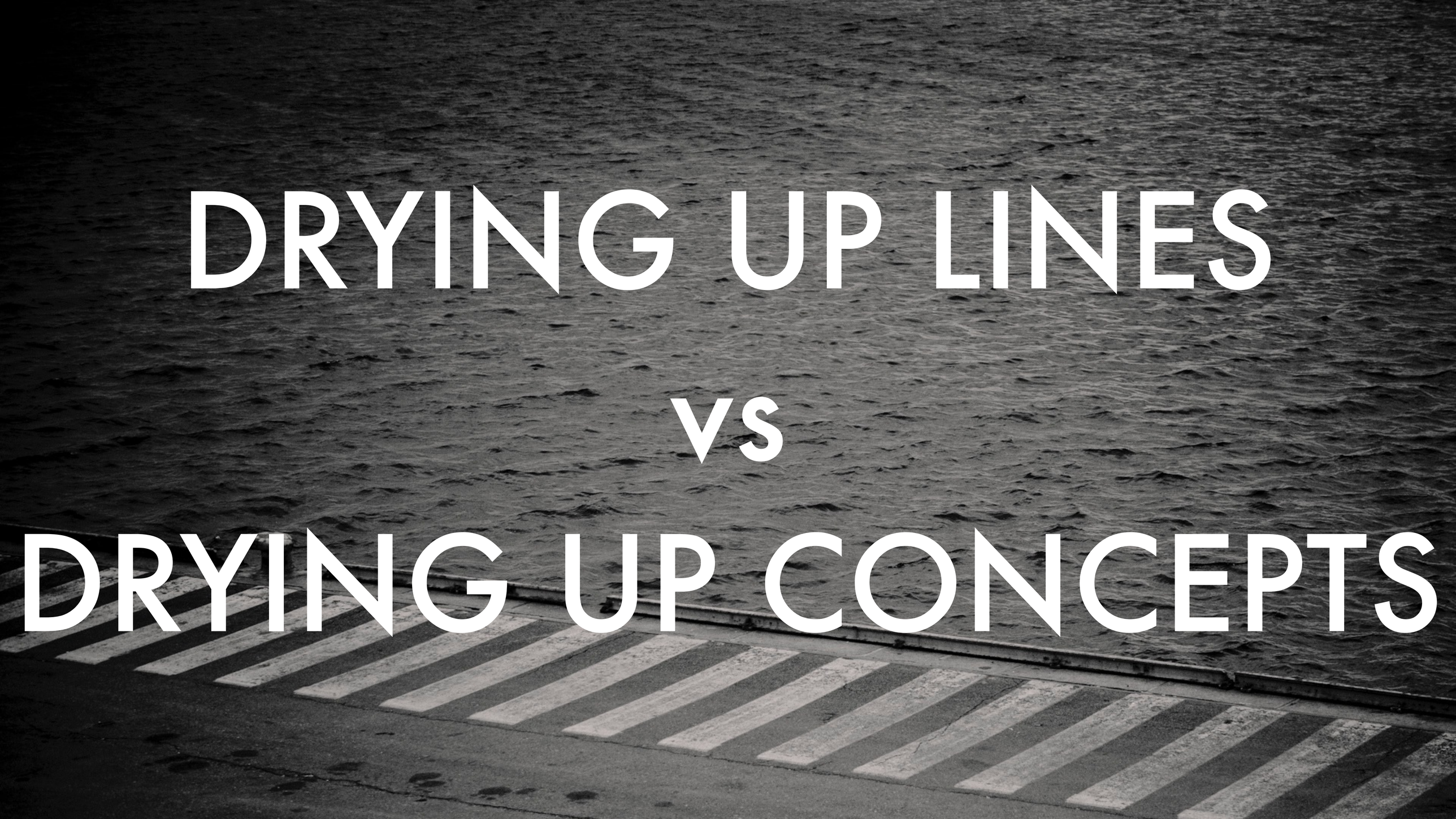
FOCUS ON THE READER

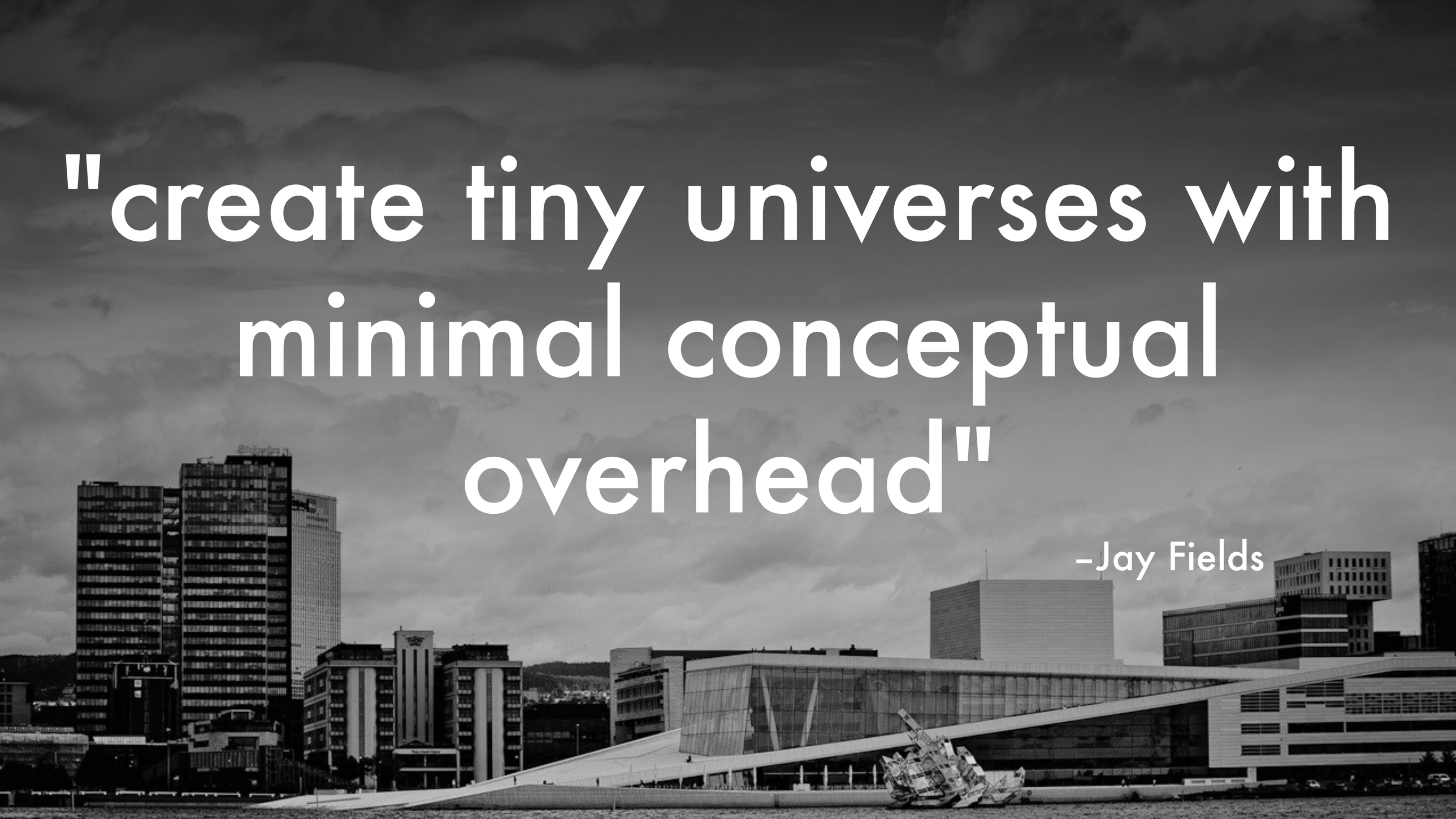
TIME SPENT READING CODE > TIME SPENT WRITING CODE

IN A TEAM YOU SPEND A LOT OF TIME READING OTHERS' CODE

NO SETUP

DRY
ISN'T ALWAYS THE ANSWER


DRYING UP LINES
vs
DRYING UP CONCEPTS



"create tiny universes with minimal conceptual overhead"

–Jay Fields

CREATION HELPERS

```
it('is valid when policy holder is young enough', function() {
    var order = createOrder({
        cart: createCart([{ type: 'CI' }]),
        policyHolder: createPolicyHolder({ age: 14 })
    });

    expect(order.isValid()).toBe(true);
});
```

```
it('is not valid when policy holder is too old', function() {
    var order = createOrder({
        cart: createCart([{ type: 'CI' }]),
        policyHolder: createPolicyHolder({ age: 15 })
    });

    expect(order.isValid()).toBe(false);
});
```

CREATION HELPERS

WITH GOOD DEFAULTS

REVEAL
INTENT

WHAT ARE WE TESTING HERE?

```
it('validates', function() {
    expect(order.isValid()).toBe(true);
});
```

REVEAL
INTENT
FOCUS ON WHY, NOT HOW

DETERMINISTIC

```
beforeEach(function() {
    policyHolder1 = new PolicyHolder({
        name: "Ola Nordmann",
        ssn: 07099451429,
        telephoneNumber: "+4795732501",
        email: "me@example.org",
        address: {
            street: "Toftes gate 17",
            postCode: 0556,
            postalArea: "Oslo"
        }
    });
    policyHolder2 = new PolicyHolder({
        name: "Testy Testeson",
        ssn: 17099926629,
        telephoneNumber: "+4743032501",
        email: "me2@example.org",
        address: {
            street: "Ravnkollbakken 3",
            postCode: 0970,
            postalArea: "Oslo"
        }
    });

    beneficiary1 = {
        name: "Testy2",
        ssn: "04037335466"
    }
    beneficiary2 = {
        name: "Testy3",
        ssn: "18049938744"
    }
    beneficiary3 = {
        name: "Testy4",
        ssn: "21050682312"
    }

    cart = new Cart();

    order = new Order({
        cart: cart,
        policyHolder: policyHolder1,
        withdrawalDay: 15
    });
});
```

REMEMBER THIS ONE?

NO
CONDITIONALS
NO FOR, IF OR WHILE

Midnight
12:01 am
1:00 am
11:00 am
Noon
12:01 pm
1:00 pm
11:59 pm

```
it("handles dates correctly", function() {
    for (var mins = 0; mins < 1440; mins++) {
        var d = new Date(2014, 1, 1, 0, 0, mins, 0)
        var event = new Event({ date: d });

        if (d.getHours() == 0 && d.getMinutes() == 0) {
            expect(event.date).toEqual("Midnight");
        }
        else if (d.getHours() == 12 && d.getMinutes() == 0) {
            expect(event.date).toEqual("Noon");
        }
        else {
            expect(event.date).toEqual(moment(d).format("h:mm a"));
        }
    }
});
```

```
it("handles dates correctly", function() {
    for (var mins = 0; mins < 1440; mins++) {
        var d = new Date(2014, 1, 1, 0, 0, mins, 0)
        var event = new Event({ date: d });

        if (d.getHours() == 0 && d.getMinutes() == 0) {
            expect(event.date).toEqual("Midnight");
        }
        else if (d.getHours() == 12 && d.getMinutes() == 0) {
            expect(event.date).toEqual("Noon");
        }
        else {
            expect(event.date).toEqual(moment(d).format("h:mm a"));
        }
    }
});
```

CAN YOU SPOT THE BUG?

```
it("handles dates correctly", function() {
    for (var mins = 0; mins < 1440; mins++) {
        var d = new Date(2014, 1, 1, 0, 0, mins, 0)
        var event = new Event({ date: d });

        if (d.getHours() == 0 && d.getMinutes() == 0) {
            expect(event.date).toEqual("Midnight");
        }
        else if (d.getHours() == 12 && d.getMinutes() == 0) {
            expect(event.date).toEqual("Noon");
        }
        else {
            expect(event.date).toEqual(moment(d).format("h:mm a"));
        }
    }
});
```

This is seconds, not minutes

CAN YOU SPOT THE BUG?

```
it("handles noon", function() {
    var event = new Event({
        date: createDate({ hours: 12, mins: 0 })
    });

    expect(event.date).toEqual("Noon");
});

it("handles midnight", function() {
    var event = new Event({
        date: createDate({ hours: 0, mins: 0 })
    });

    expect(event.date).toEqual('Midnight');
});
```

SEE IT FAIL
YOU DON'T WANT TO FAIL AT FAILING

QuickTime Player File Edit View Window Help

Jasmine Spec Runner v2.0

localhost:8000/SpecRunner.html

Jasmine 2.0.2 finished in 0.003s

1 spec, 0 failures raise exceptions

Player
increments count when song is finished

PlayerSpec.js (~/dev/presen...mples/async-fail/spec) - VIM2

```
8 describe("Player", function() {
7
6     it("increments count when song is finished", function() {
5         var player = new Player();
4
3         expect(player.countDone).toBe(0);
2
1         player.play('song name', function() {
0             expect(player.countDone).toBe(1);
1         });
2     });
3
4 });
```

PlayerSpec.js 9,43 All

"PlayerSpec.js" 13L, 291C written

THIS TEST WILL NEVER FAIL

MacVim File Edit Tools Syntax Buffers Window Help

Jasmine Spec Runner v2.0

localhost:8000/SpecRunner.html

Jasmine 2.0.2 finished in 0.001s

1 spec, 0 failures raise exceptions

Player
increments count when song is finished

PlayerSpec.js + (~/dev/pres...mples/async-fail/spec) – VIM2

```
8 describe("Player", function() {
7
6     it("increments count when song is finished", function() {
5         var player = new Player();
4
3         expect(player.countDone).toBe(0);
2
1         player.play('song name', function() {
0             expect(player.countDone).toBe(3);
1         });
2     });
3
4 });
```

PlayerSpec.js [+] 9,43 All

THIS TEST WILL NEVER FAIL

Chrome File Edit View History Bookmarks Window Help

Jasmine Spec Runner v2.0

localhost:8000/SpecRunner.html

Jasmine 2.0.2 finished in 0.015s

1 spec, 1 failure raise exceptions

Spec List | Failures

Player increments count when song is finished

Expected 0 to be 3.

```
Error: Expected 0 to be 3.
    at stack (http://localhost:8000/lib/jasm
    at buildExpectationResult (http://localh
    at Spec.Env.expectationResultFactory (ht
    at Spec.addExpectationResult (http://loc
    at Expectation.addExpectationResult (htt
    at Expectation.toBe (http://localhost:80
    at http://localhost:8000/spec/PlayerSpec
    at http://localhost:8000/src/Player.js:9
```

PlayerSpec.js (~/dev/presen...mples/async-fail/spec) - VIM2

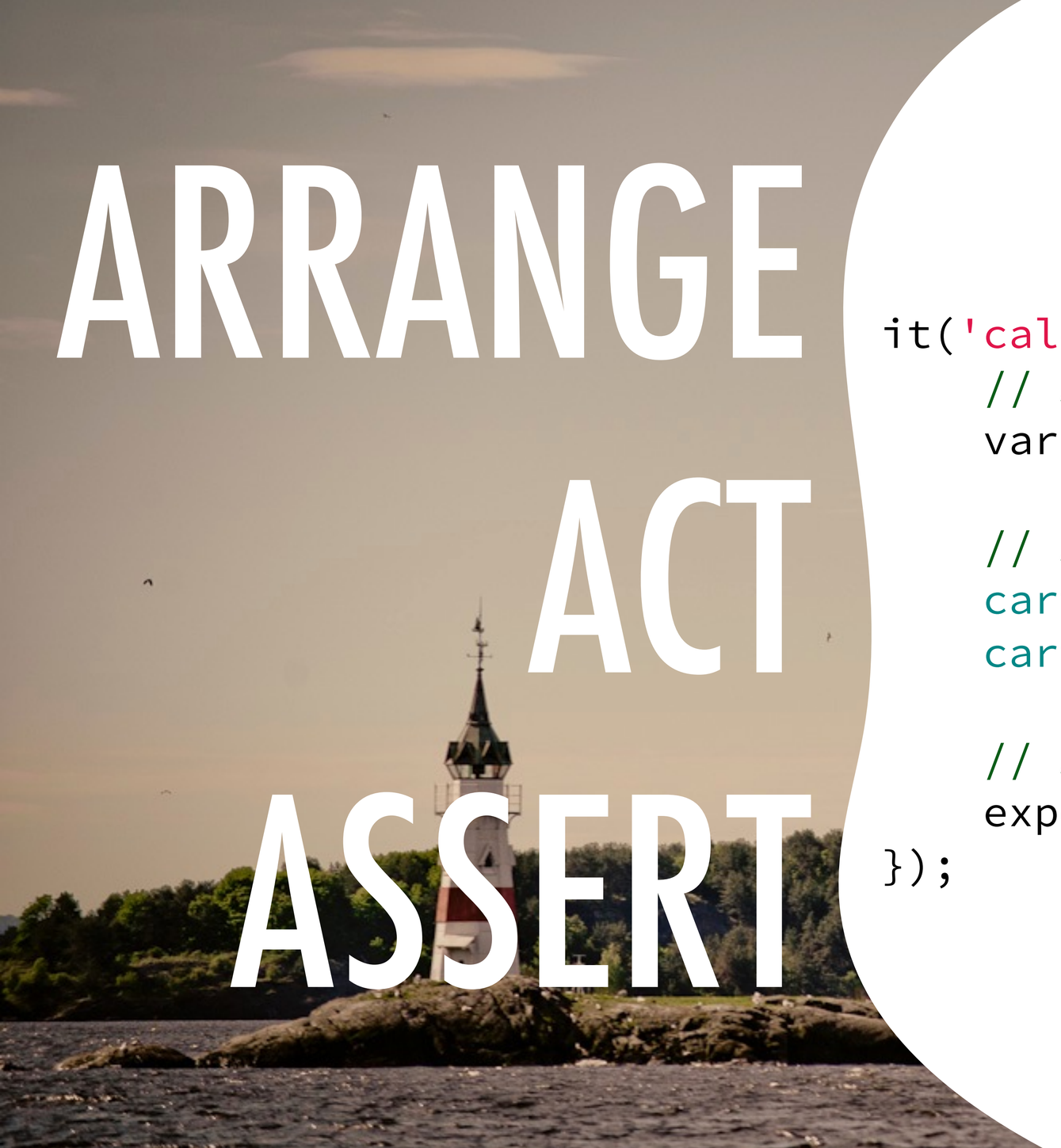
```
9 describe("Player", function() {
8
7     it("increments count when song is finished", function(done) {
6         var player = new Player();
5
4         expect(player.countDone).toBe(0);
3
2         player.play('song name', function() {
1             expect(player.countDone).toBe(3);
0             done();
1         });
2     });
3
4 });
~
~
~
~
~
~
~
~
~
~
~
~
PlayerSpec.js                                   10,19          All
"PlayerSpec.js" 14L, 315C written
```

ASYNC TESTS NEED TO TELL WHEN THEY ARE DONE

Remember, the most important thing your tests do is

FAIL

BEWARE OF ASYNC

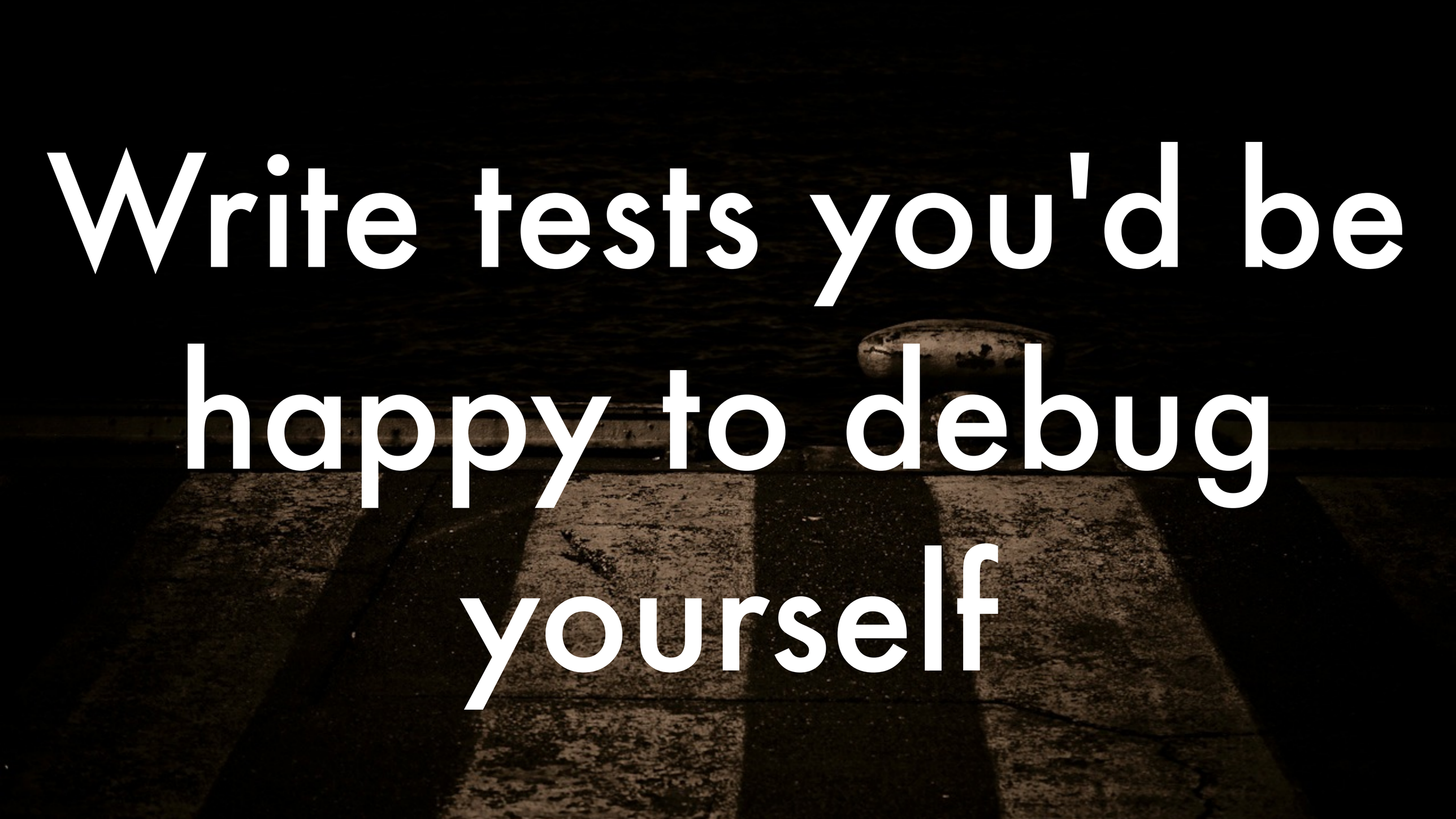



```
it('calculates total price', function() {
    // ARRANGE
    var cart = new Cart();

    // ACT
    cart.addItem(...);
    cart.addItem(...);

    // ASSERT
    expect(cart.totalPrice()).toEqual(1097);
});
```

and perhaps most importantly ...

Tests **SHOULD** to be stupidly simple

What happens when we don't understand our tests?


Write tests you'd be
happy to debug
yourself

MAINTAINING
YOUR TESTS

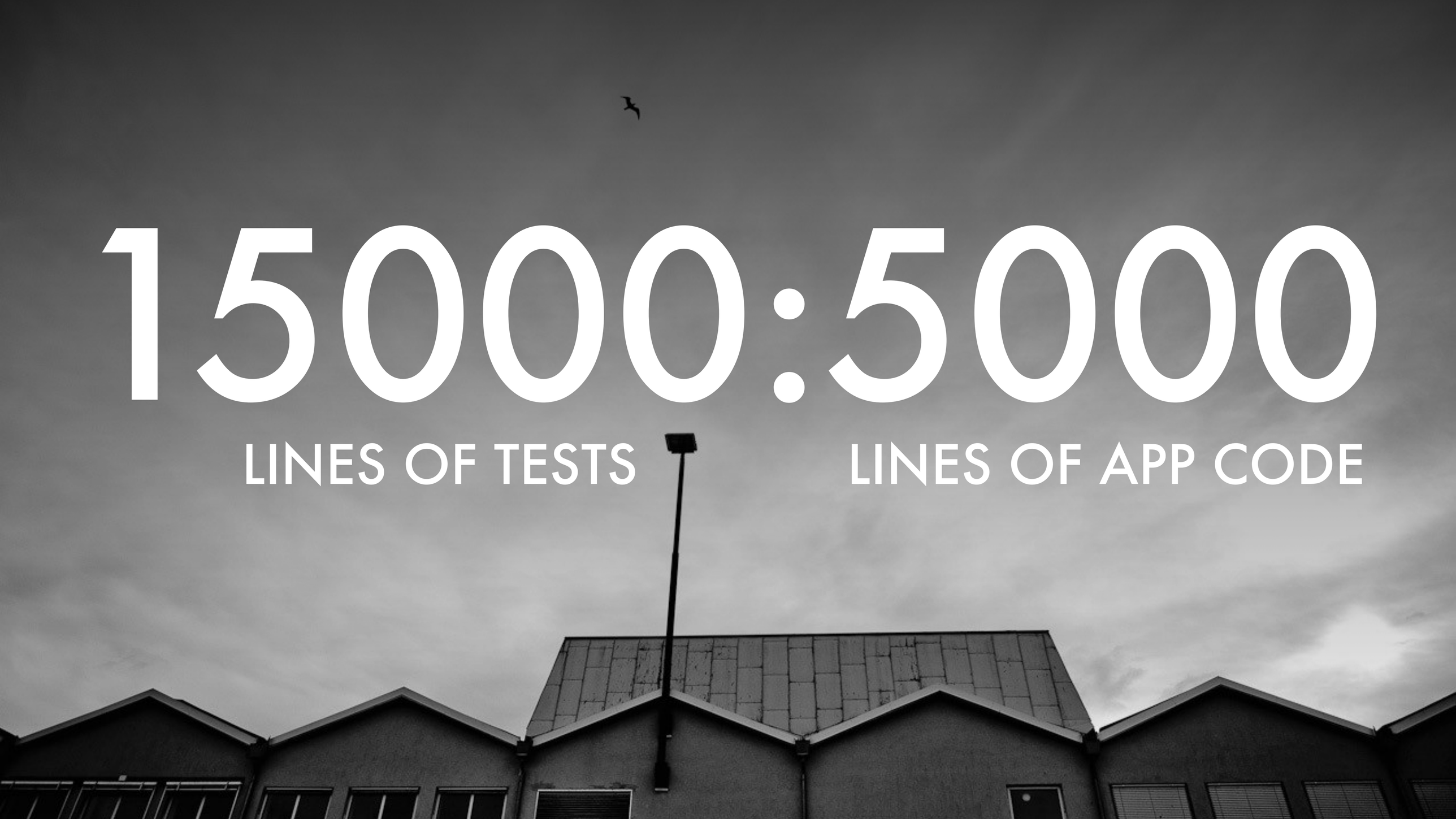
POOR QUALITY TESTS

CAN SLOW DEVELOPMENT TO A CRAWL

FEEL THE PAIN
IF IT DOESN'T FEEL GOOD, FIX IT!

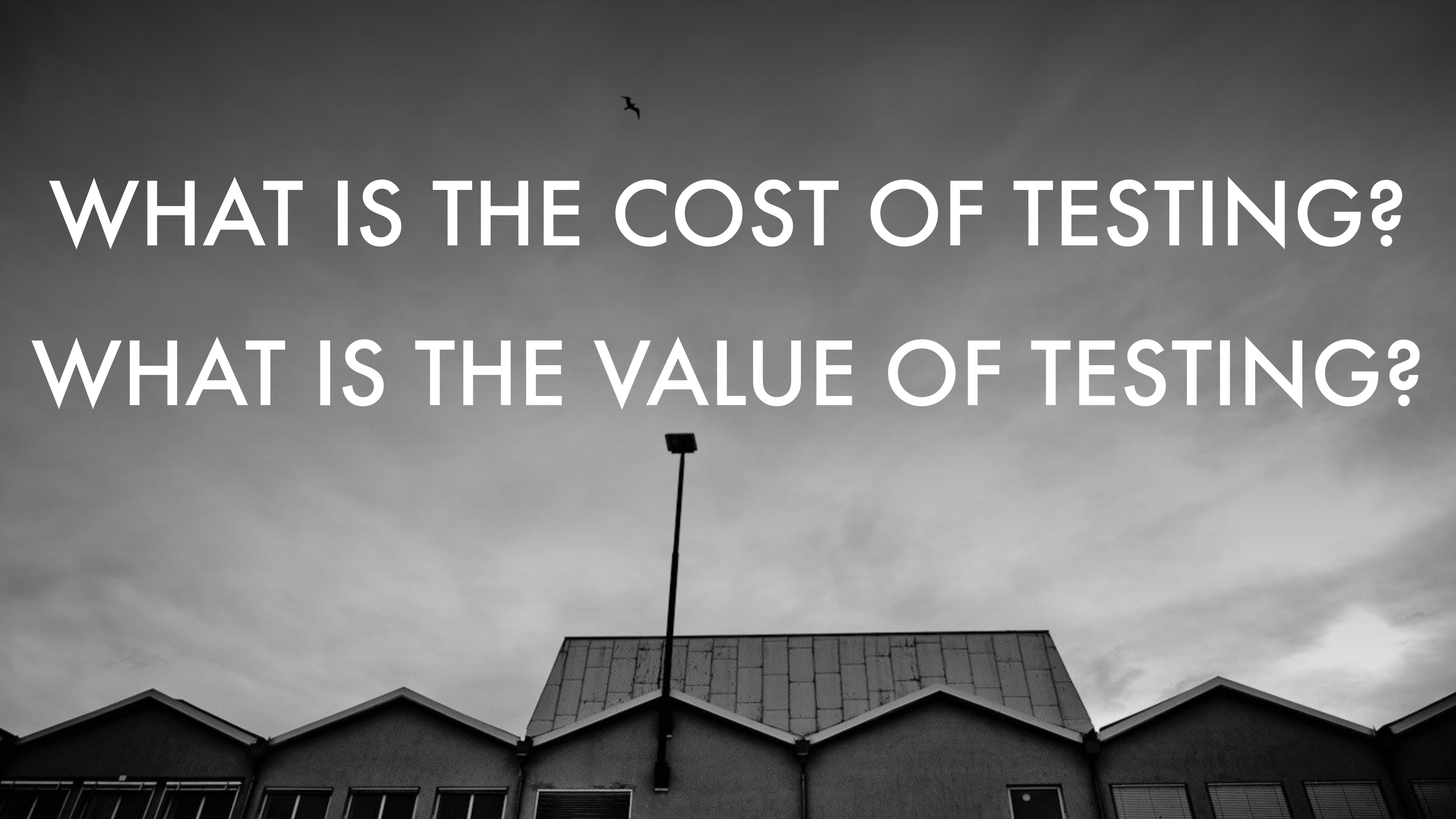
REFACTOR
YOUR TESTS

3:1


15000:5000
LINES OF TESTS
LINES OF APP CODE

"... test the areas that you are most worried about going wrong"

– Martin Fowler


WHAT IS THE COST OF TESTING?
WHAT IS THE VALUE OF TESTING?

LESS
DOGMA-DRIVEN
TESTING, PLEASE

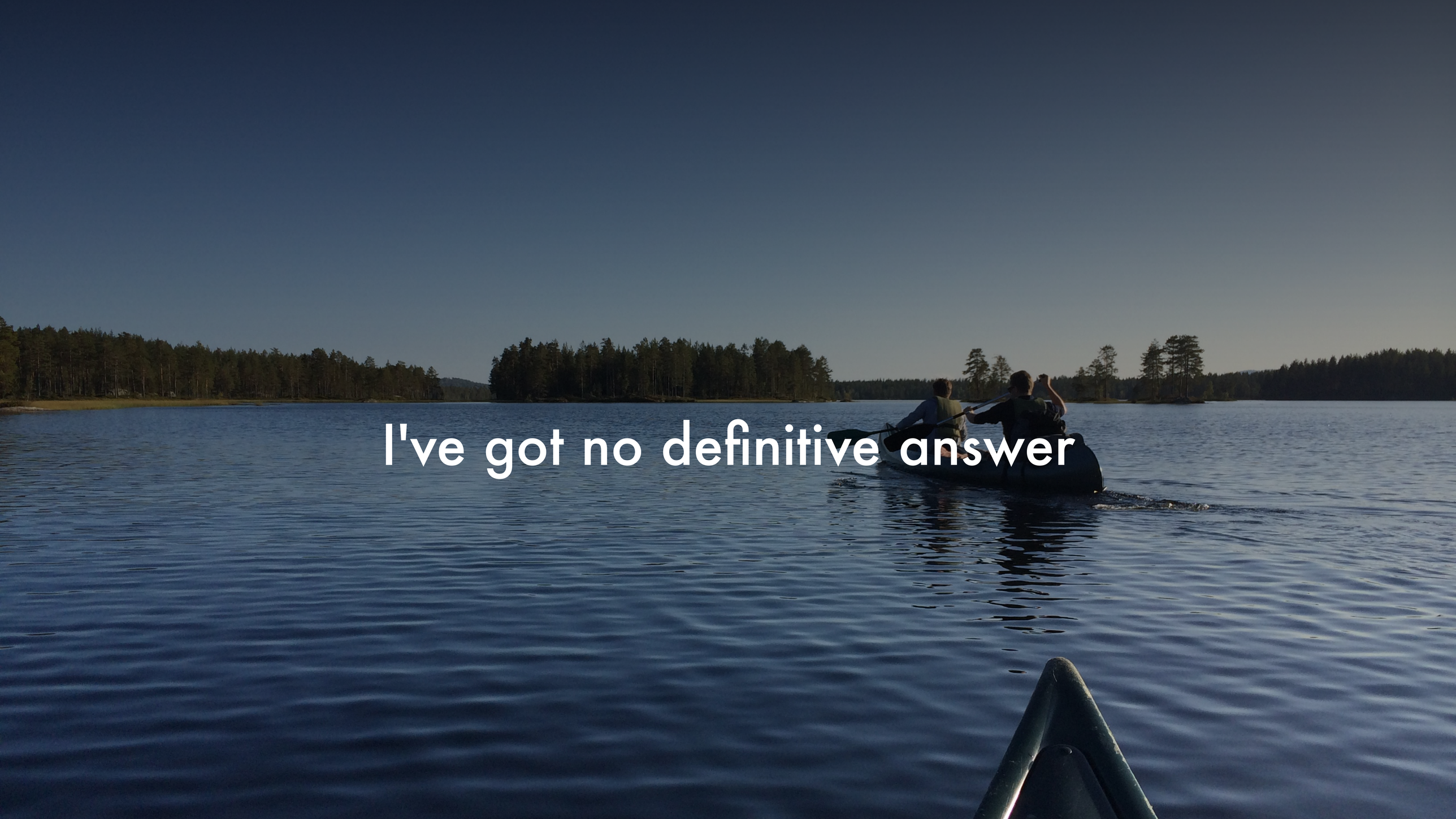
DON'T ACCEPT
INCOMPREHENSIBLE
TESTS

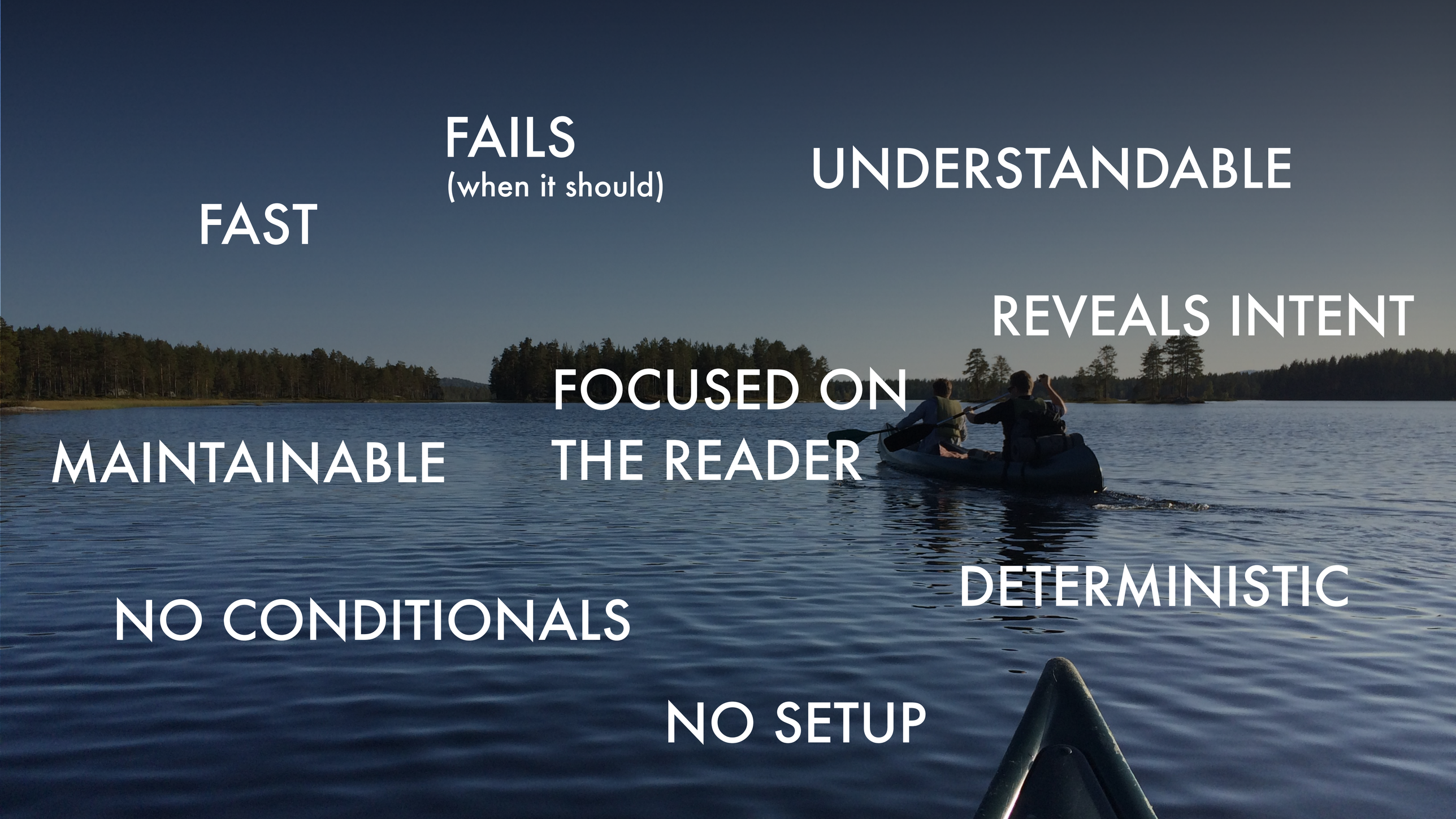
KEEP 'EM
SIMPLE

TESTS CAN
BE AWESOME

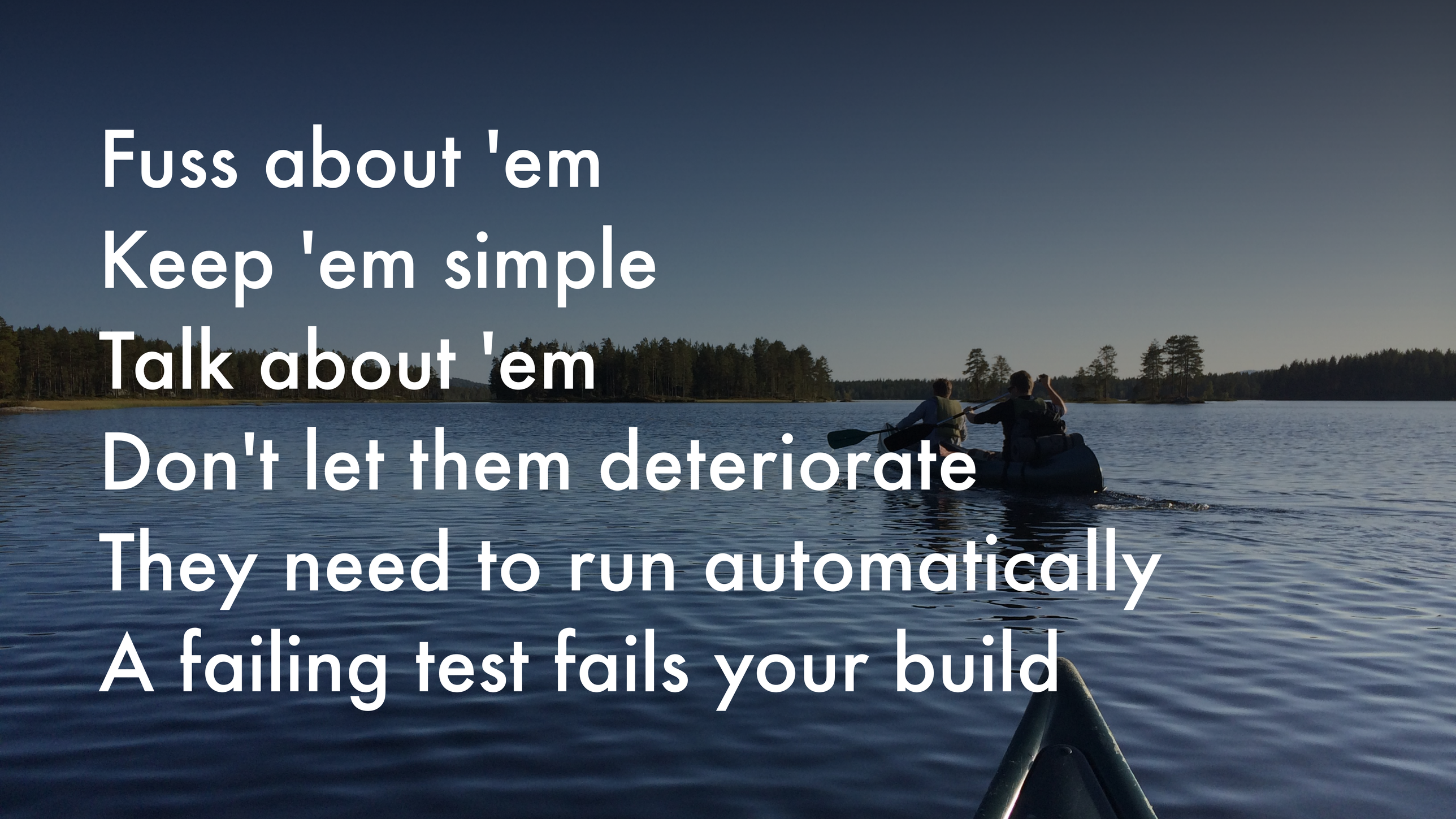
HOW CAN WE *INCREASE* THE VALUE?

SO – WHAT IS A BEAUTIFUL TEST?

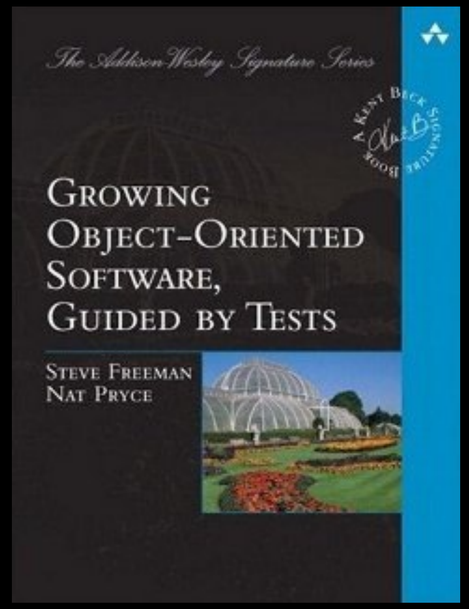

I've got no definitive answer


FAILS
(when it should)
UNDERSTANDABLE
FAST
REVEALS INTENT
FOCUSED ON
THE READER
MAINTAINABLE
DETERMINISTIC
NO CONDITIONALS
NO SETUP

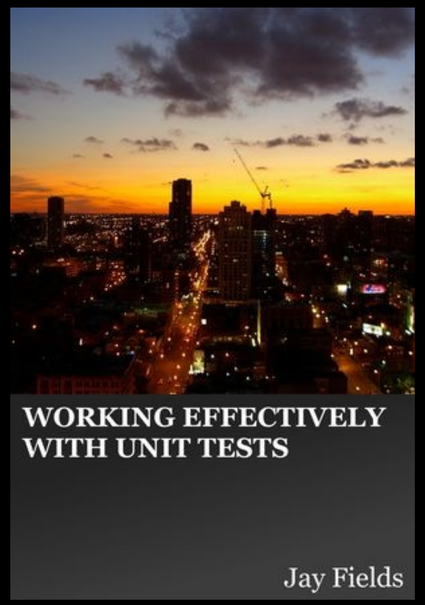
Focus on the craft, not the tools

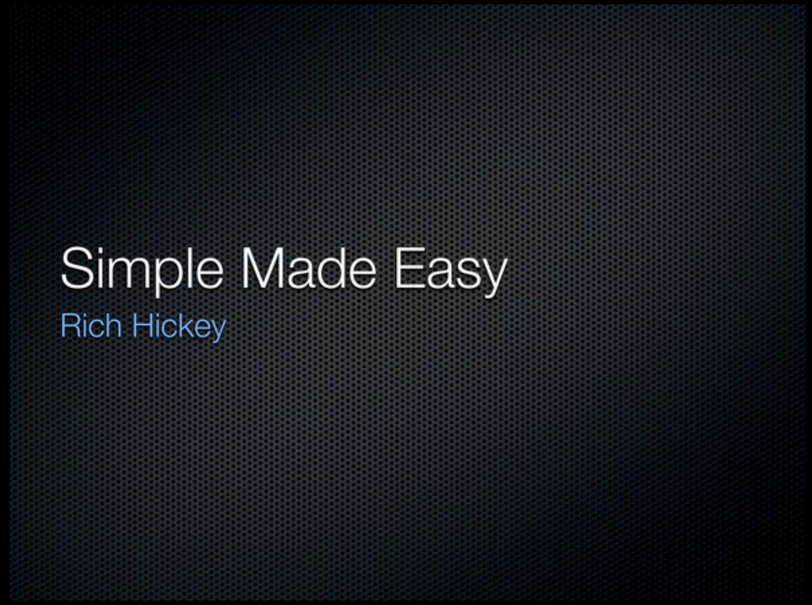


Fuss about 'em
Keep 'em simple
Talk about 'em
Don't let them deteriorate
They need to run automatically
A failing test fails your build



WANT TO LEARN MORE?





THANK YOU
I'm here until Sunday – let me know if you want to talk JavaScript
kimjoar.net