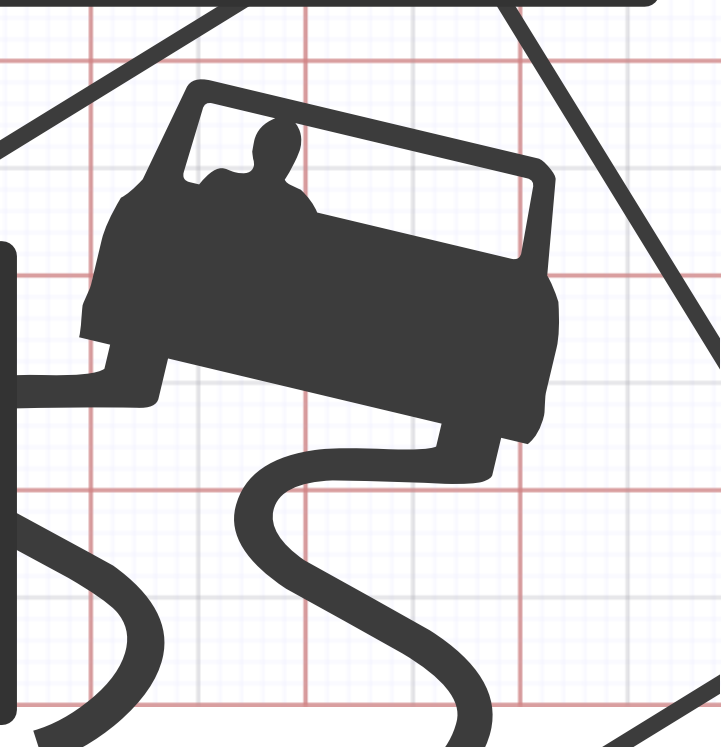




Test Driven Development

AND
GETTING PAID!



Rowan Merewood
Software Eng. / Team Lead
Ibuildings

WHO AM I?

Software Engineer &
Technical Team Lead
at Ibuildings UK

*I want to write good code
and earn a living*

@rowan_m

DETOUR

WHO AM I

Not?

DO NOT

ENTER

I AM NOT A
CONSULTANT

WELL MAYBE A LITTLE

WHO AM I

Not?

DO NOT

ENTER

I AM NOT
SELLING A
BOOK

WHO AM I

Not?

DO NOT

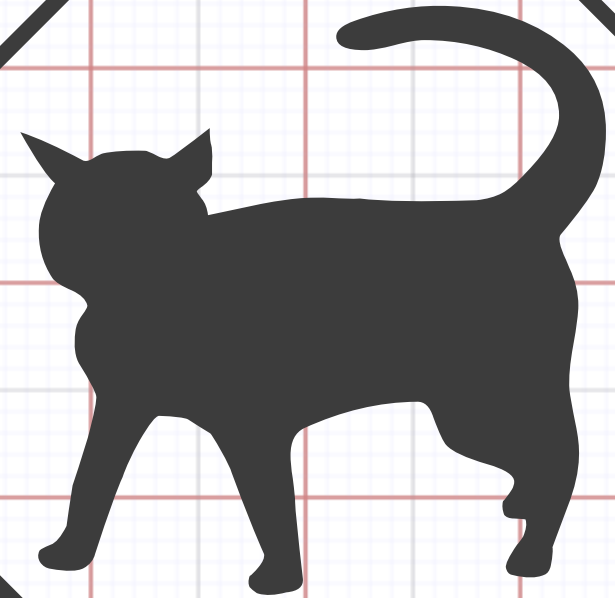
ENTER

I AM NOT
A SLAVE TO
ONE METHOD

THE GOOD

Clean code, smart devs
Latest technology
Building your career

Elitist / Intimidating ?

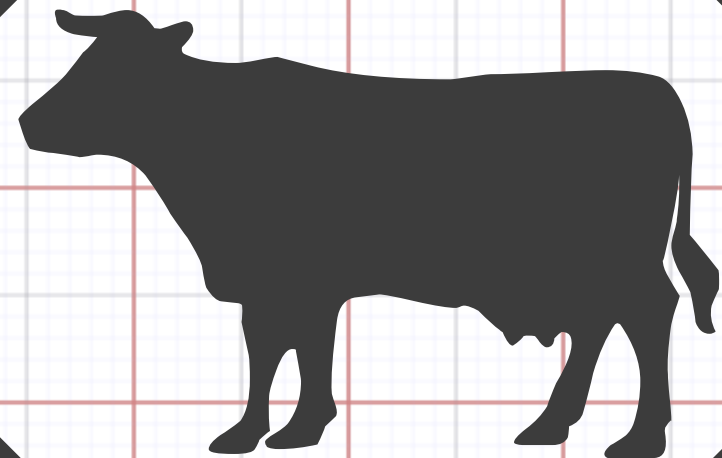


DIFFERENT SITUATIONS

THE BAD

Disgusting code
Devs don't care
Career dead-end

Changes break the app.
Always bug-fixing

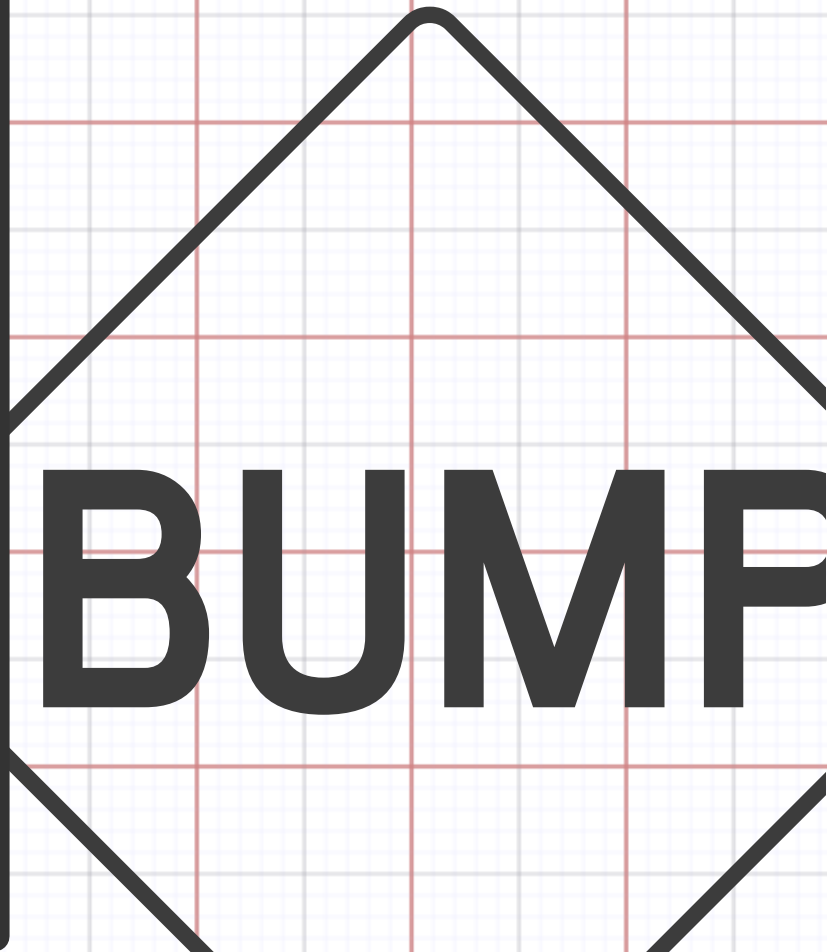


DIFFERENT SITUATIONS

THE UGLY

Good tests are hard
Writing tests takes time
Time is money

You're not an expert
(yet...)



BUMP

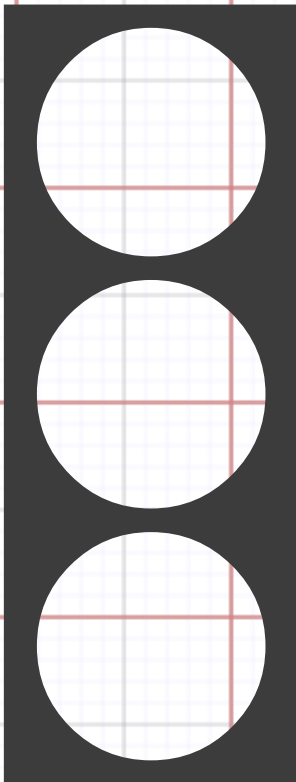
DIFFERENT SITUATIONS

WHAT IS TDD?

1. Decide what you want to do
2. Write a test to show it working
3. Run the test and watch it fail
4. Write just enough code to pass the test
5. Re-run the test (and test suite)
6. Refactor (refine/improve)
7. Re-run tests
8. Repeat

WHAT IS TDD?

SIMPLIFY



RED



GREEN



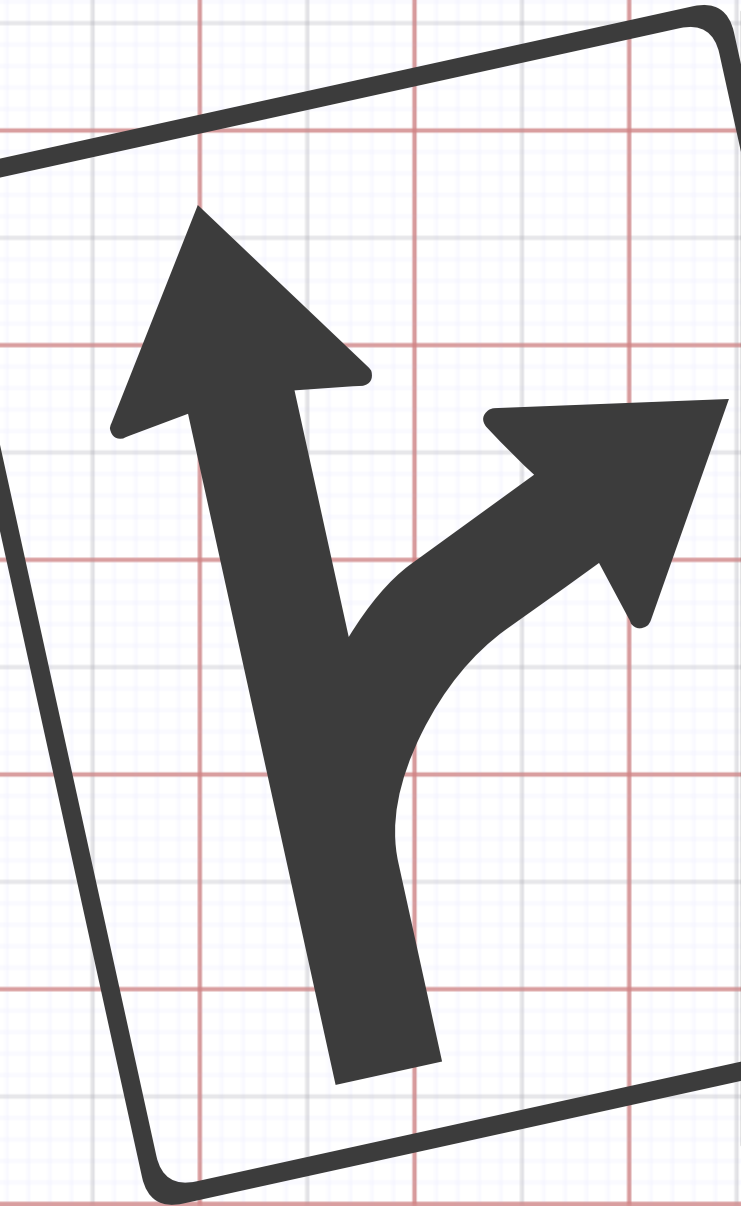
REFACTOR

WHY IS THIS HARD?

Do you know what
you want

before

you code it?



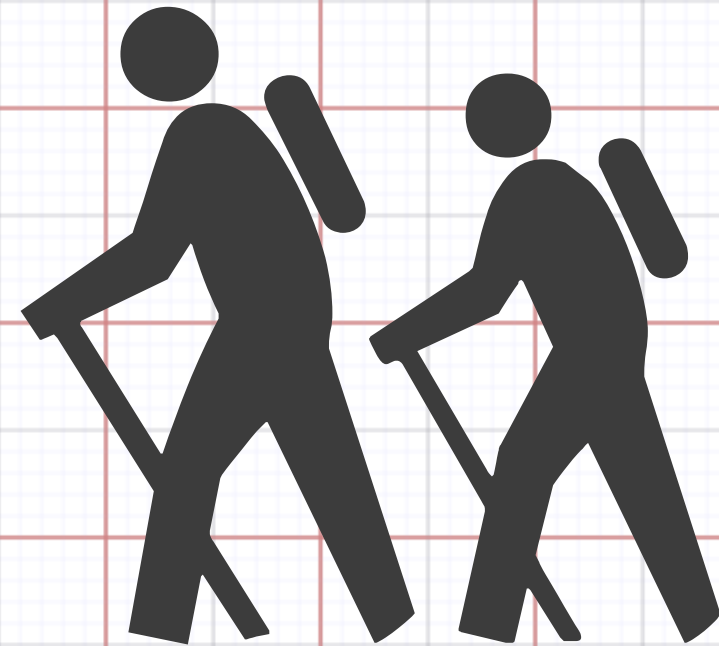
WHY IS THIS HARD?

Does your client
know what
they want?

EVER?

Train yourself to think like a scientist

1. Hypothesis
2. Repeatable Experiment
3. Conclusions



Train yourself to think like a **NINJA** scientist

Kata

(型 or 形, literally: "form")

A set of movements
you repeat again and again
until can do it perfectly.



NINJA WEAPON

DAVE THOMAS

co-author of
The Pragmatic
Programmer



Code Kata

<http://codekata.pragprog.com/>

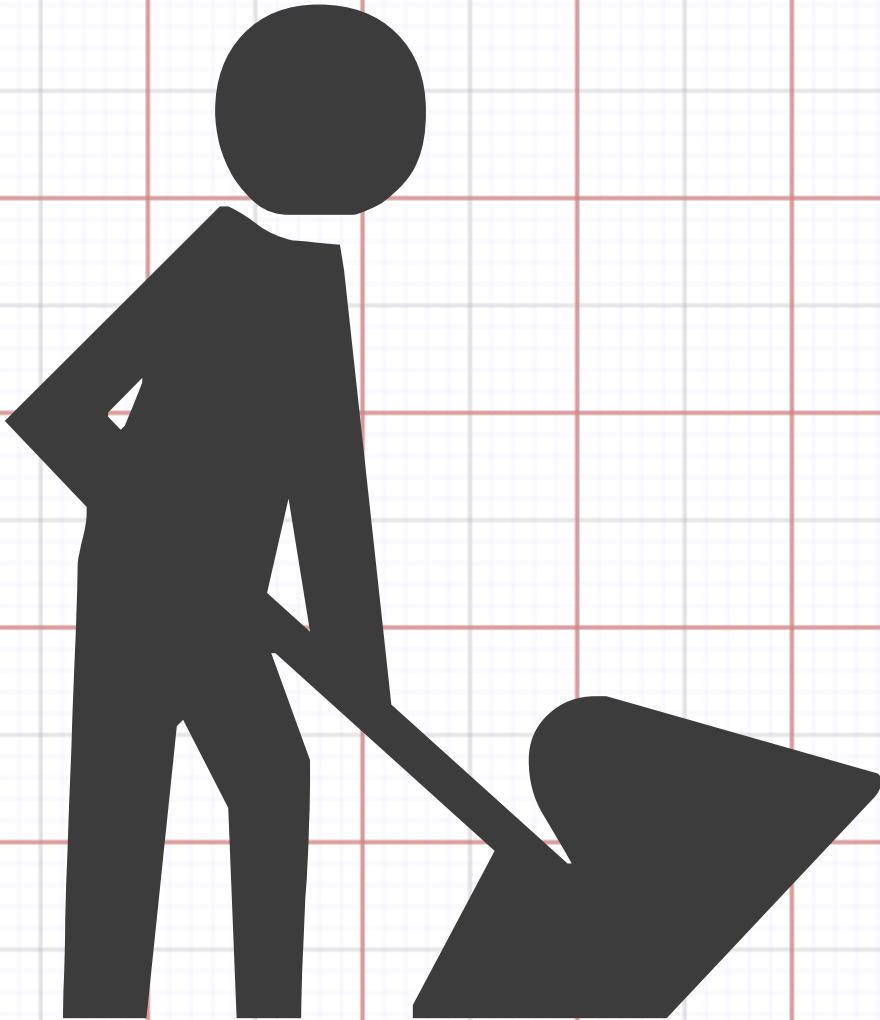
ROY OSHEROVE - TDD Kata

<http://osherove.com/tdd-kata-1/>

Create a simple string calculator with a method `int Add(string numbers)`

The method can take 0, 1 or 2 numbers, and will return their sum (for an empty string it will return 0) for example "" or "1" or "1,2"

ROY OSHEROVE - TDD Kata

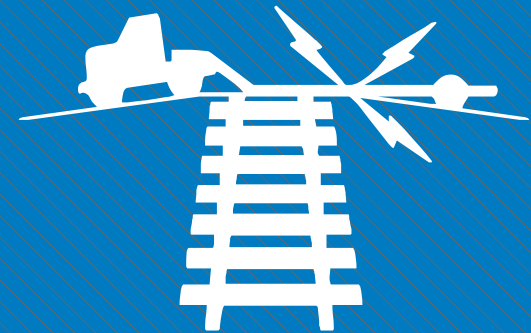


DEMO

Now you're an expert...

WARNING

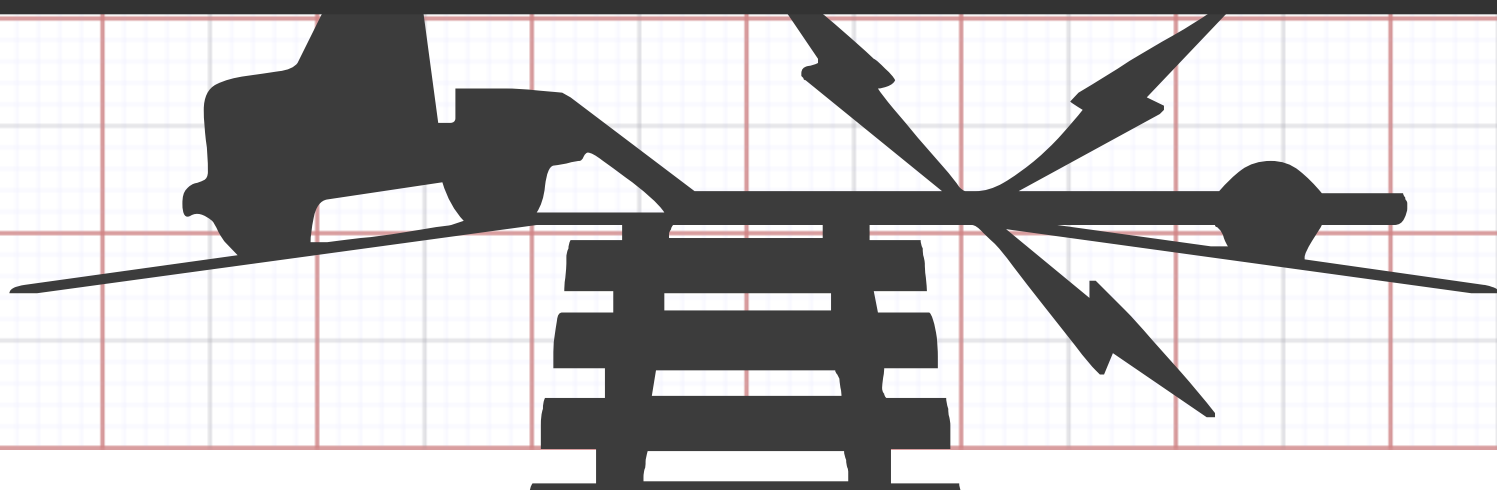
Do not assume you can
just start to do this
in your project



DON'T BE A HERO

Introduce tests all at once...

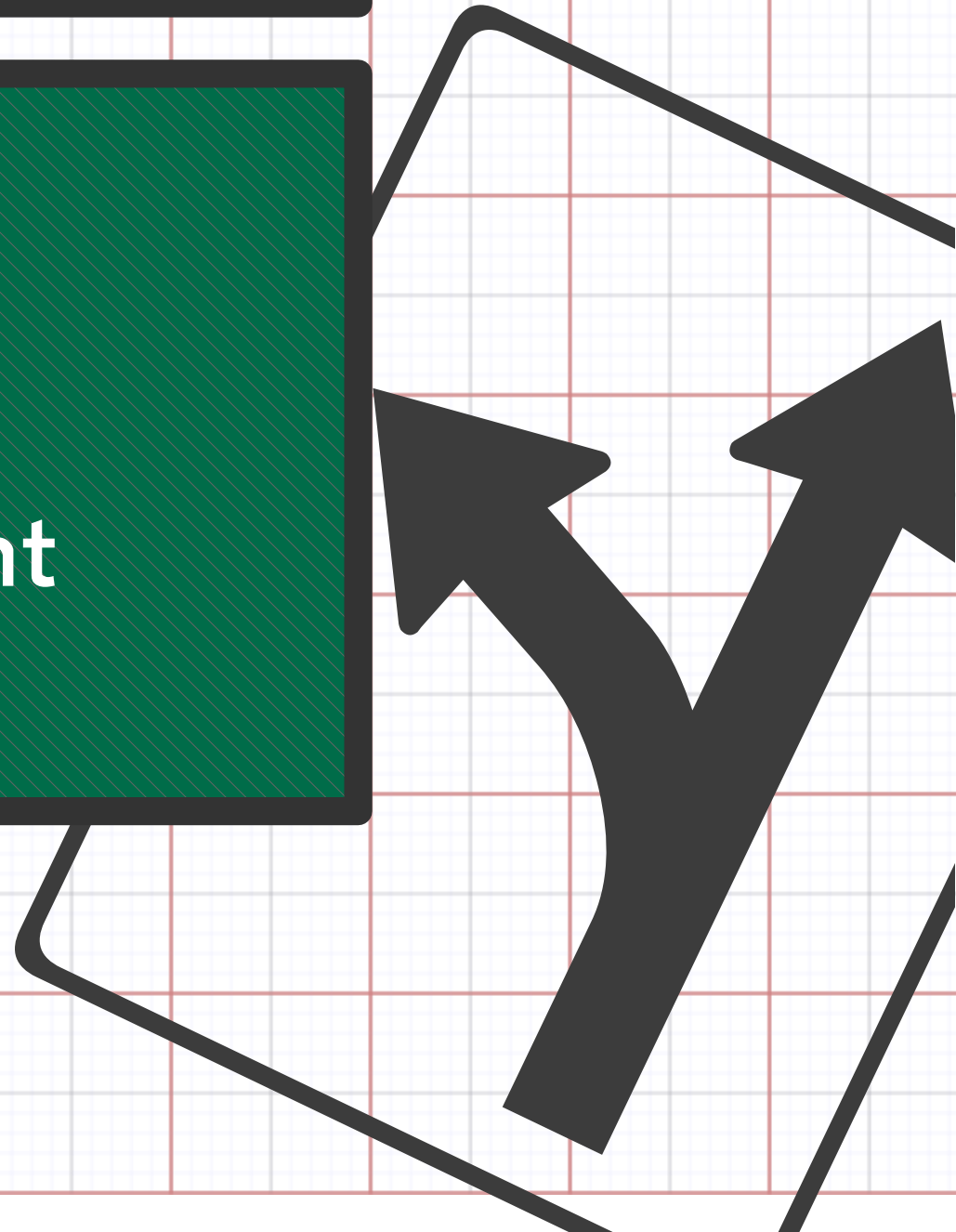
- You will miss your deadline
- Your tests will not be maintained
- Your team will hate you



DIFFERENT APPROACHES

Force tests
on your client

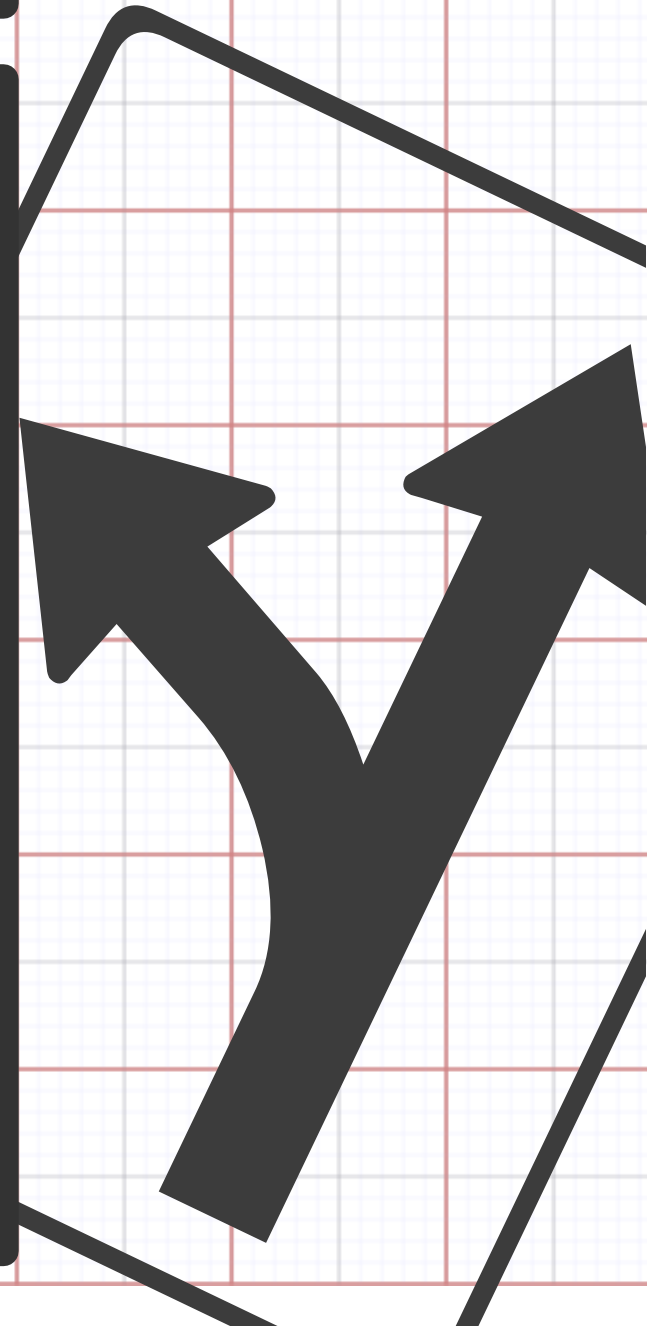
Make your client
want the tests



FORCING TESTS

Add a fixed percentage to your estimates.

Do not compromise your principles.

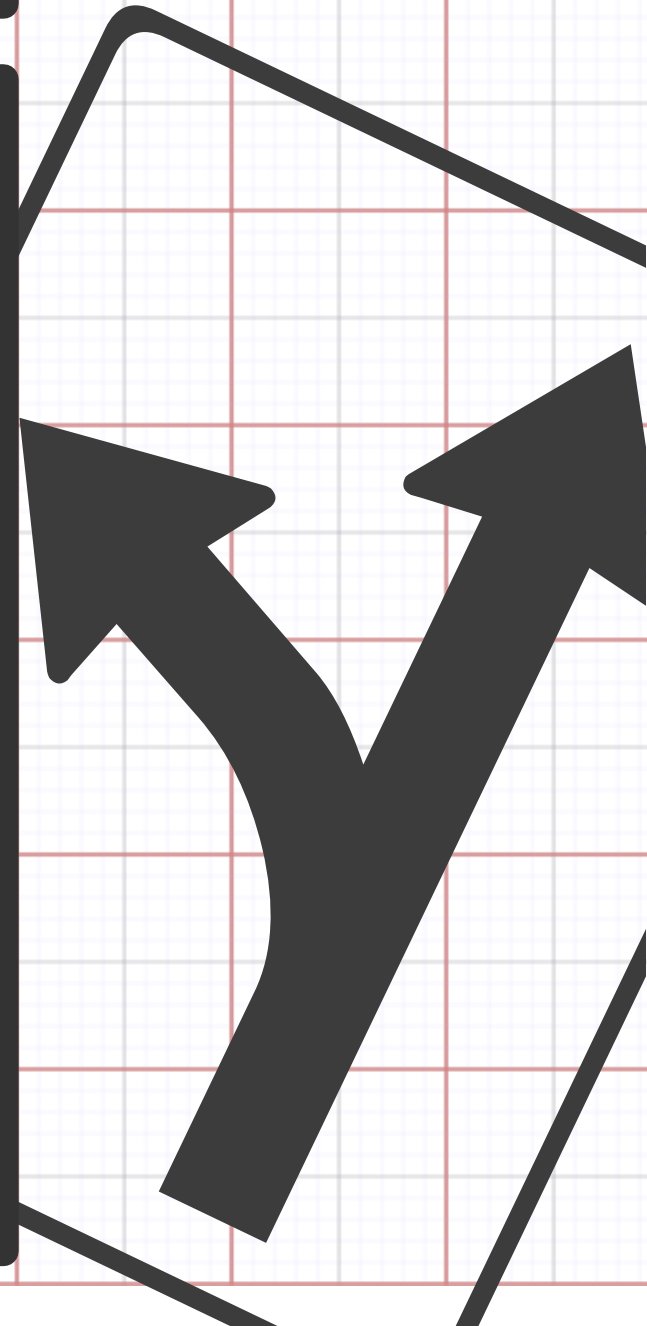


SELLING TESTS

Use tests to define 'done'.

Involve the client
in creating tests.

Make the tests
a deliverable.



SELLING TESTS

Use tests to define 'done'.

Involve the client
in creating tests.

Make the tests
a deliverable.

Fitness Framework
<http://fitnessse.org/>

SELLING TESTS

Use tests to define 'done'.

Involve the client in creating tests.

Make the tests a deliverable.

Fitness Framework
<http://fitnesse.org/>

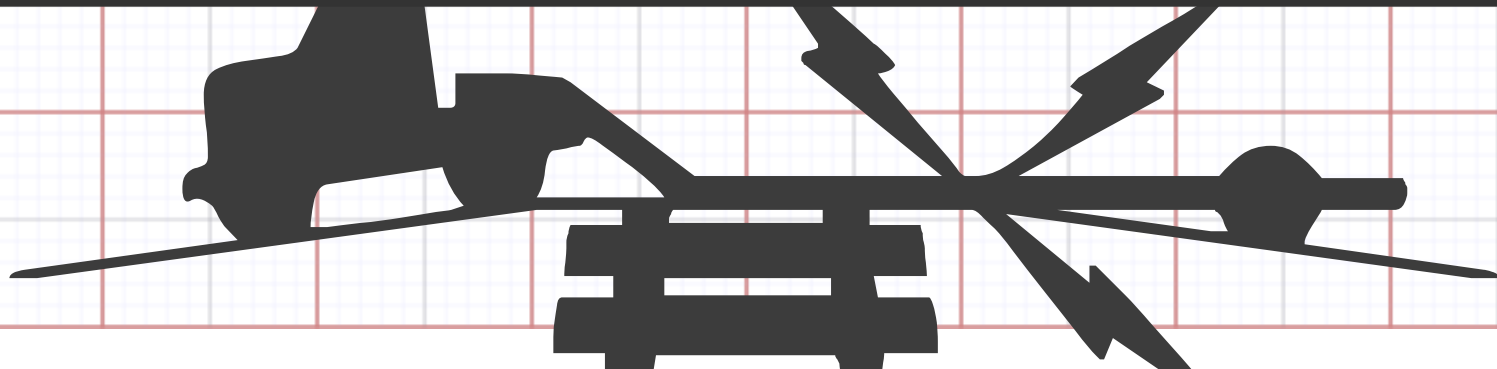
Selenium
<http://seleniumhq.org/>

DON'T BELIEVE THE HYPE

Make sure you don't over-promise.

**Make sure you have the
infrastructure and skills**

NO SILVER BULLETS



INFRASTRUCTURE

AND

CONSTRUCTION

AHEAD

- 1. Unit Testing**
- 2. Acceptance Testing**
- 3. Automated Deployment**
- 4. Continuous Integration**
- 5. Issue Tracking**

INFRASTRUCTURE

AD

CONSTRUCTION

AHEAD

Unit/Acceptance testing
provides the technical base

- 2. Acceptance Testing**
- 3. Automated Deployment**
- 4. Continuous Integration**
- 5. Issue Tracking**

INFRASTRUCTURE

AD

CONSTRUCTION

HEAD

Automated deployment
allows a quick test env.

2. Performance Testing

3. Automated Deployment

4. Continuous Integration

5. Issue Tracking

INFRASTRUCTURE

CONSTRUCTION

Continuous Integration
makes progress visible

- 1.
- 2.
3. Automated Deployment
4. Continuous Integration
5. Issue Tracking

INFRASTRUCTURE

AD

CONSTRUCTION

HEAD

Issue Tracking allows reporting on TDD

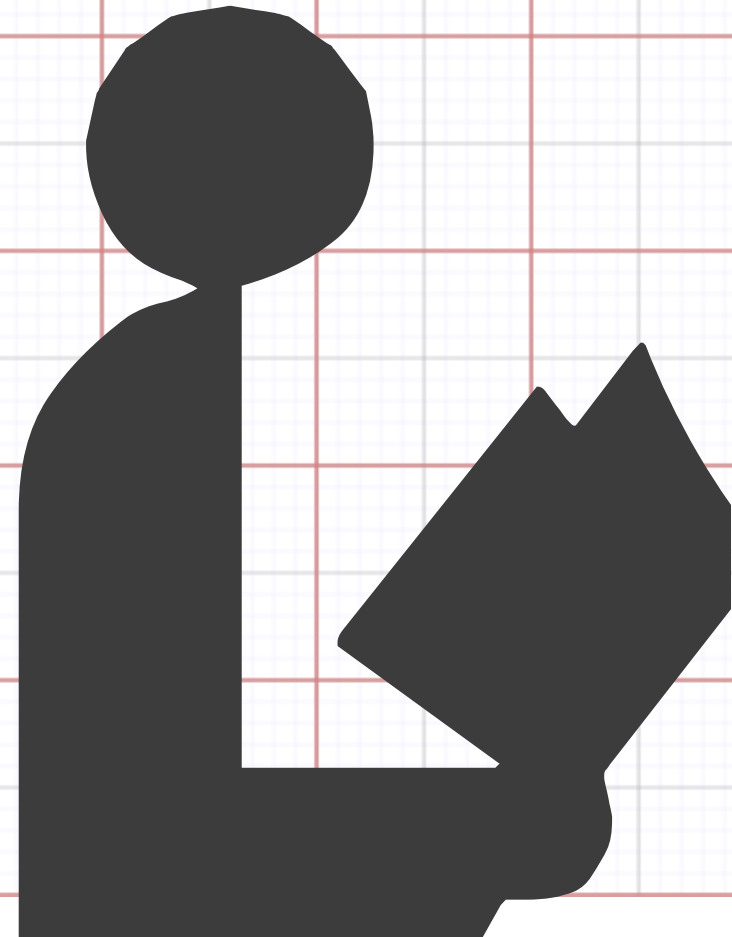
3. Automated Deployment

4. Continuous Integration

5. Issue Tracking

REPORTING

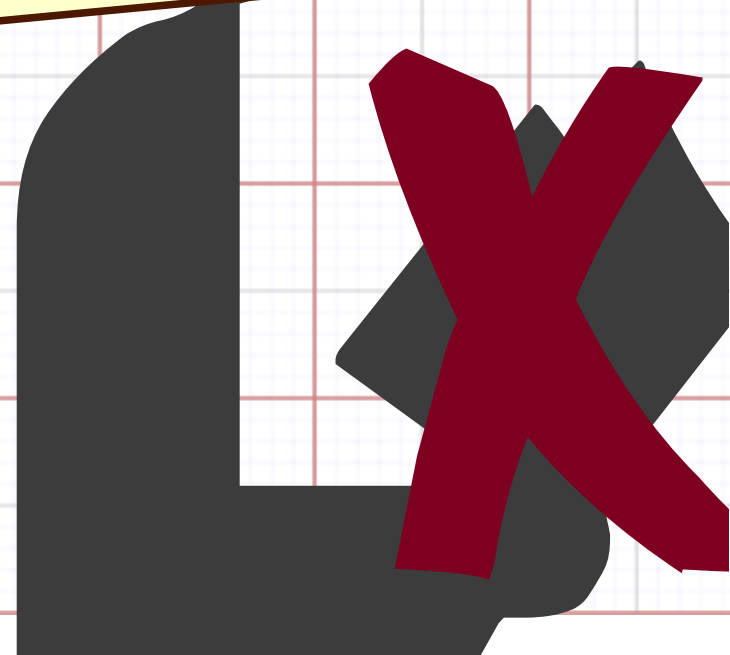
1. Code Coverage
2. Branch Coverage
3. Bug Origin:
 - tested code
 - untested code
4. Test/Dev Time:
 - per feature
 - per story



REPORTING

1. Code Coverage
2. Branch Coverage
3. Bug Origin:
 - tested code
 - untested code
4. Test/Dev Time:
 - per feature
 - per story

Only track a metric if it is useful and encourages the right behaviour!

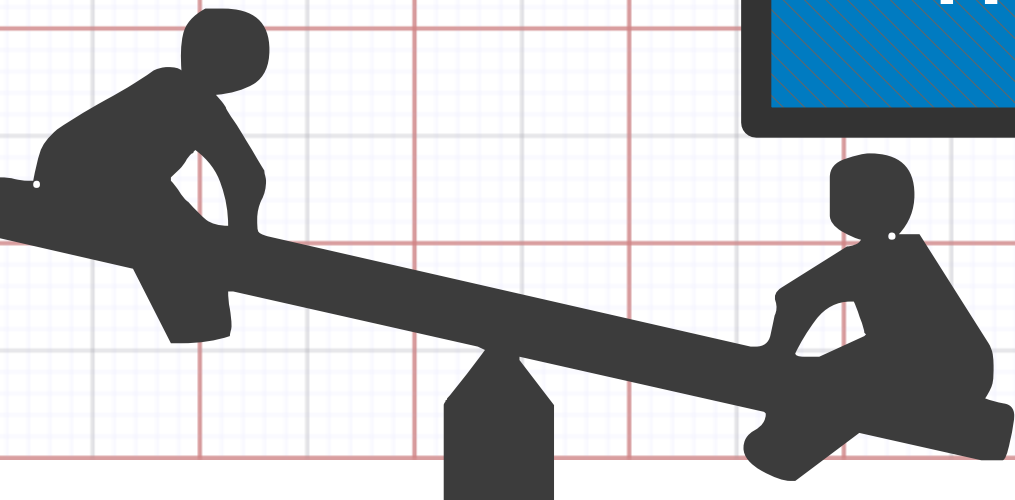


SKILLS & THE TEAM

Owners not heroes

Prepared to fail

Honest & Disciplined



**TDD DOES NOT
CREATE GOOD
CODE**

STOP

WAIT...

WHAT?

**TDD DOES NOT
CREATE GOOD
CODE**

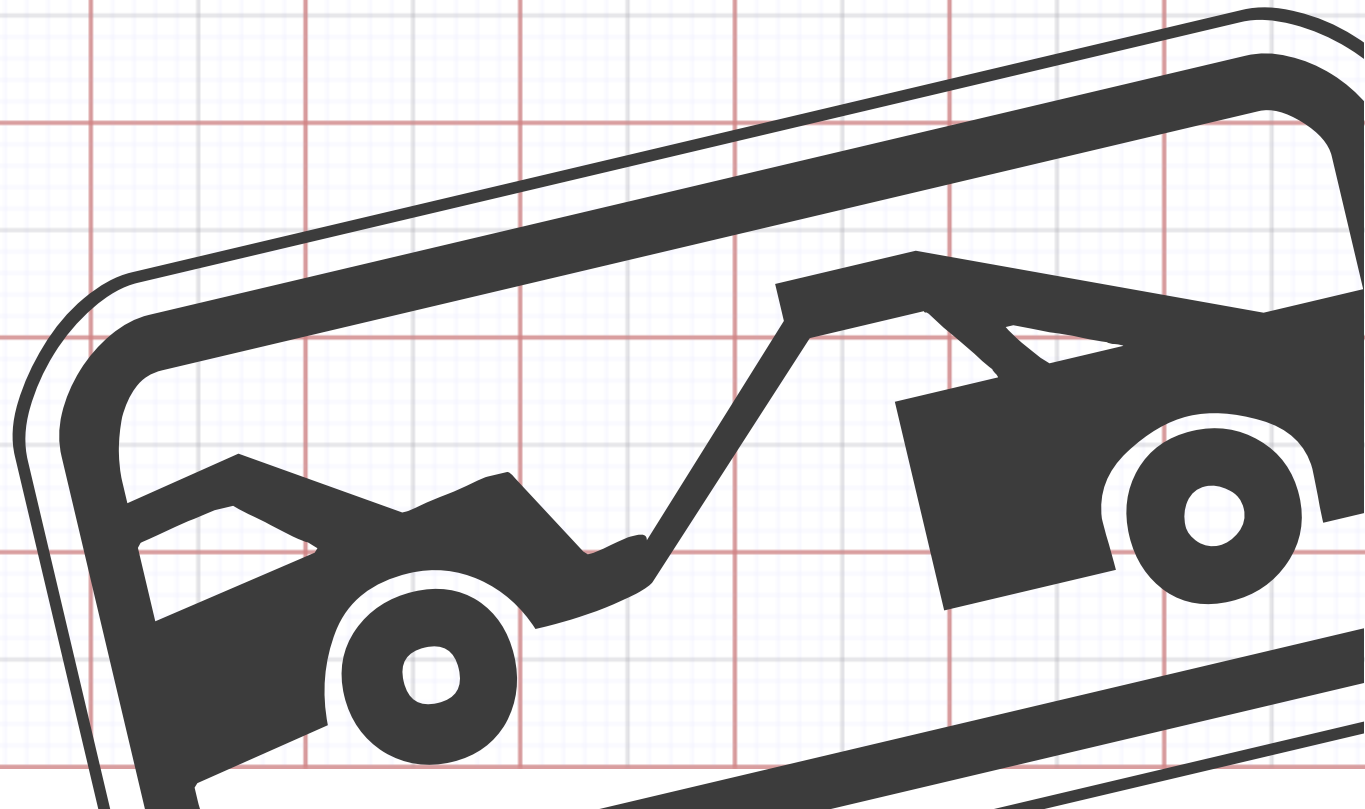
STOP

**GOOD PLAN
GOOD DEV
GOOD CODE**

.....

**BAD PLAN
BAD DEV
BAD CODE**

DISASTER RECOVERY



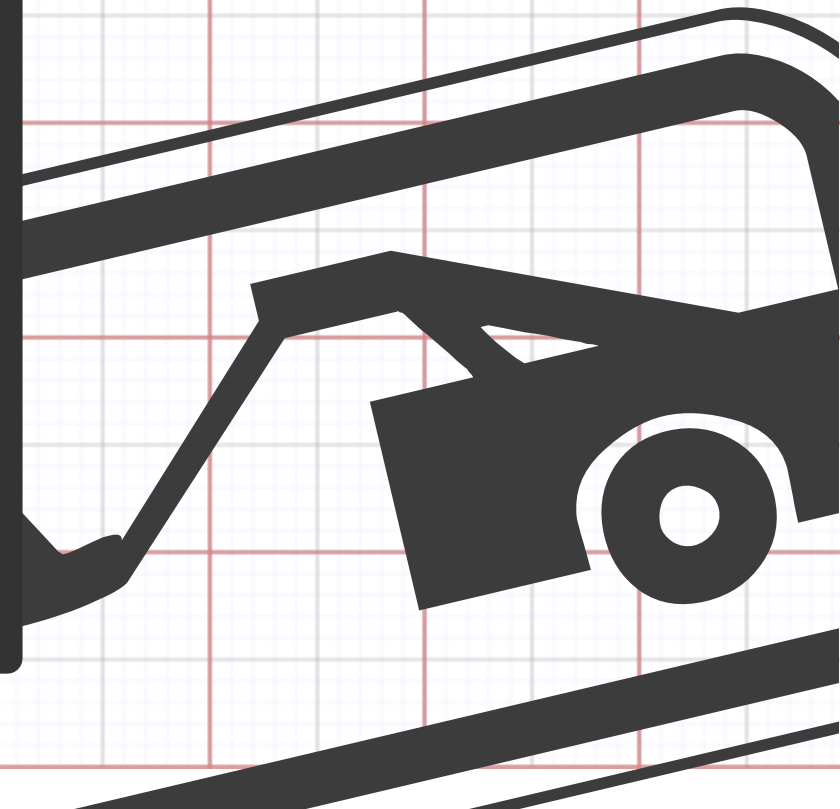
DISASTER RECOVERY

Untestable code ?

Isolate and contain

-or-

Create a testable API



DISASTER RECOVERY

Running late ?

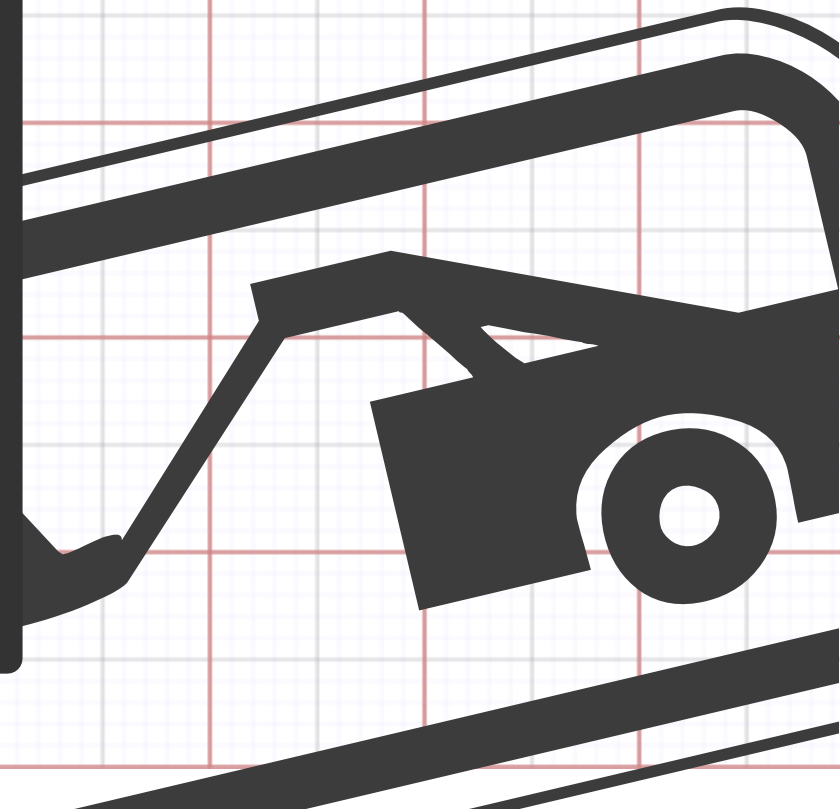
Drop features

-or-

Test the happy path

-or-

Test core only



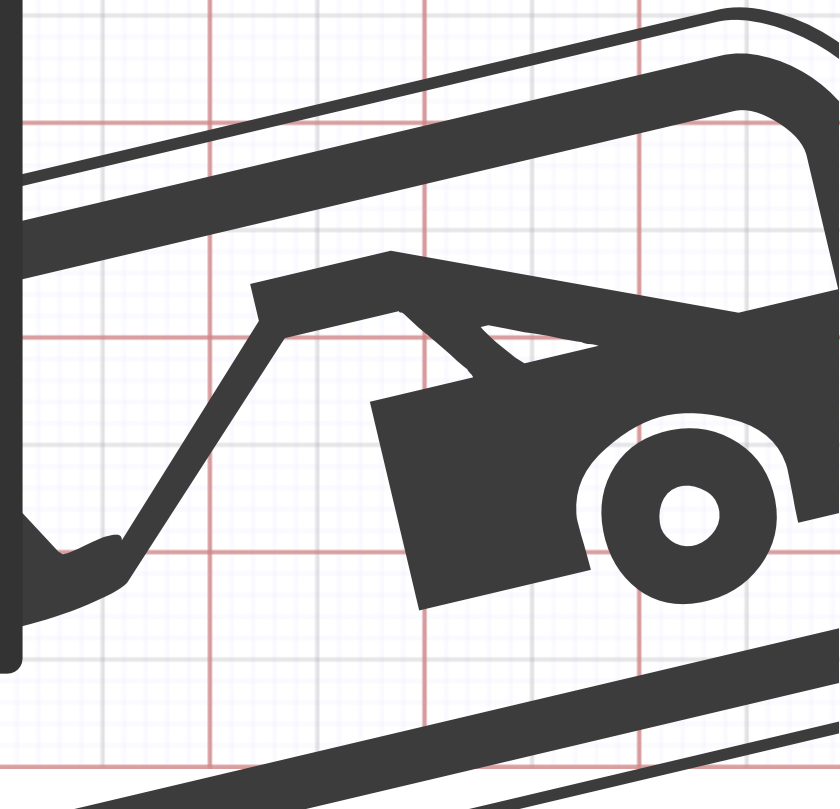
DISASTER RECOVERY

Broken build ?

Fix the test

-or-

Disable the test
(or delete it)



DISASTER RECOVERY

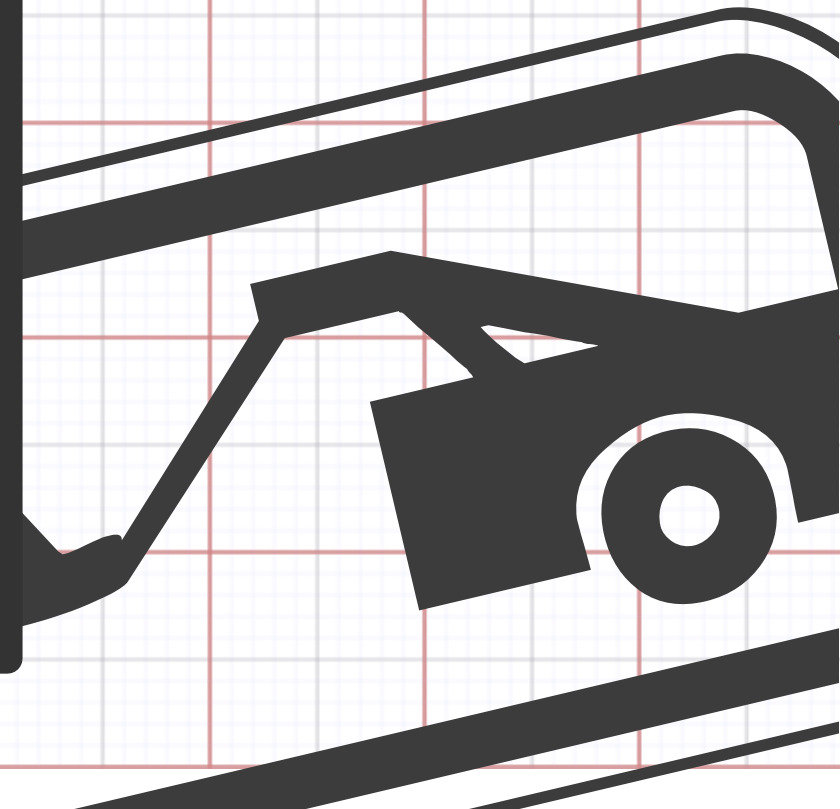
Team doesn't care

Use incentives/games

-or-

Find another job

you'RE WORTH it!



**END
CONSTRUCTION**

QUESTIONS?

@rowan_m

<http://joind.in/3218>

THANK YOU