Real Life Test Maintenance

Hello!

I AM GIL ZILBERFELD

www.gilzilberfeld.com

www.everydayunittesting.com

@gil_zilberfeld

For the exercises:

https://github.com/gilzilberfeld/test-maintenance-exercises

https://bit.ly/2JaM3Pb

Test Maintenance

What is it?

@gil_zilberfeld

Maintenance starts here and now

It requires a plan

@gil_zilberfeld

Once we got a map

We can create a plan of where we want to go

High level view

- What is important for the product?
- What are the risks?
- What should we focus on?
- At this stage and the short-mid term future


LET'S DO SOME MAPPING!
imgflip.com

Kayak

- Important: Commerce, Immediacy
- Risks: Dependency on other systems

  Information, reliability, accuracy
- Focus: Integration

What about Facebook?

Importance, risk, focus

Facebook

- Important: Ad sales, privacy, trust
- Risks Losing money, finding out what we really do, showing relevant information
- Focus Complexity, new algorithms

Exercise: What would Facebook do?

- Developing a new algorithm for fake news identification
- Only UI tests
- 1 tester
- Lots of FB data lying around

What about Candy Crush Saga?

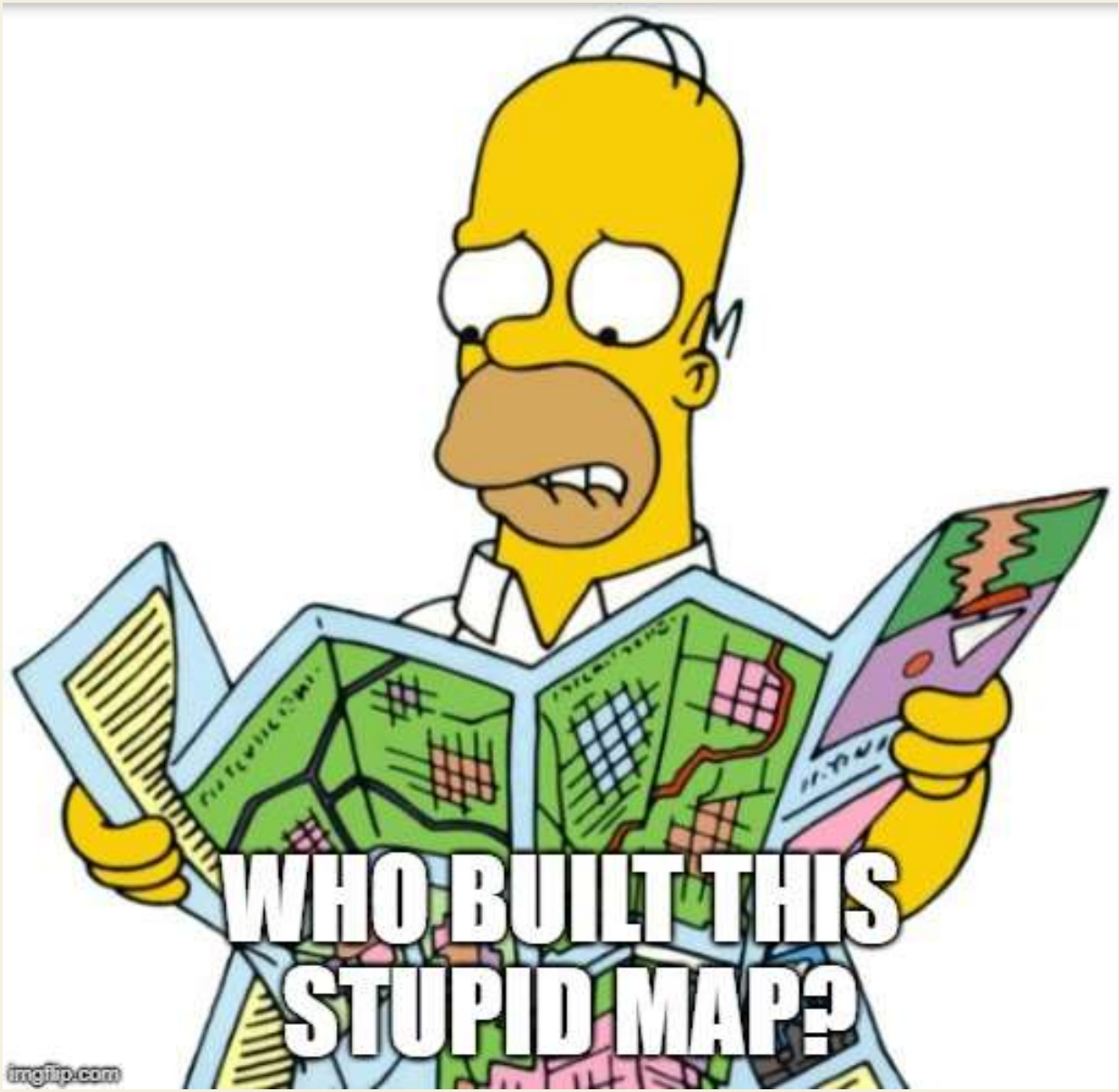
Importance, risk, focus

Candy Crush Saga

- Important — Hooking, selling upgrades, balance
- Risks — Competitors, keeping whales interested
- Focus — Usability, quick testing for new levels

Exercise: What would King do?

- Adding a new level
- Just unit tests for the engine
- Needs to work on all platforms
- Easy to hack the system
- Good API and engine testing skills, UI not so much


WHO BUILT THIS
STUPID MAP?
imgflip.com

Testing cartography

- The tests we have
- The workflows we cover (value)
- The risks we answer
- The costs incurred by the tests
- The dependencies we rely on
- The architecture
- The resources
- The skills

The value of our tests

- Do existing tests give value?
- Cover the right areas
- Find bugs
- Stable
- Pinpoint the problems

@gil_zilberfeld

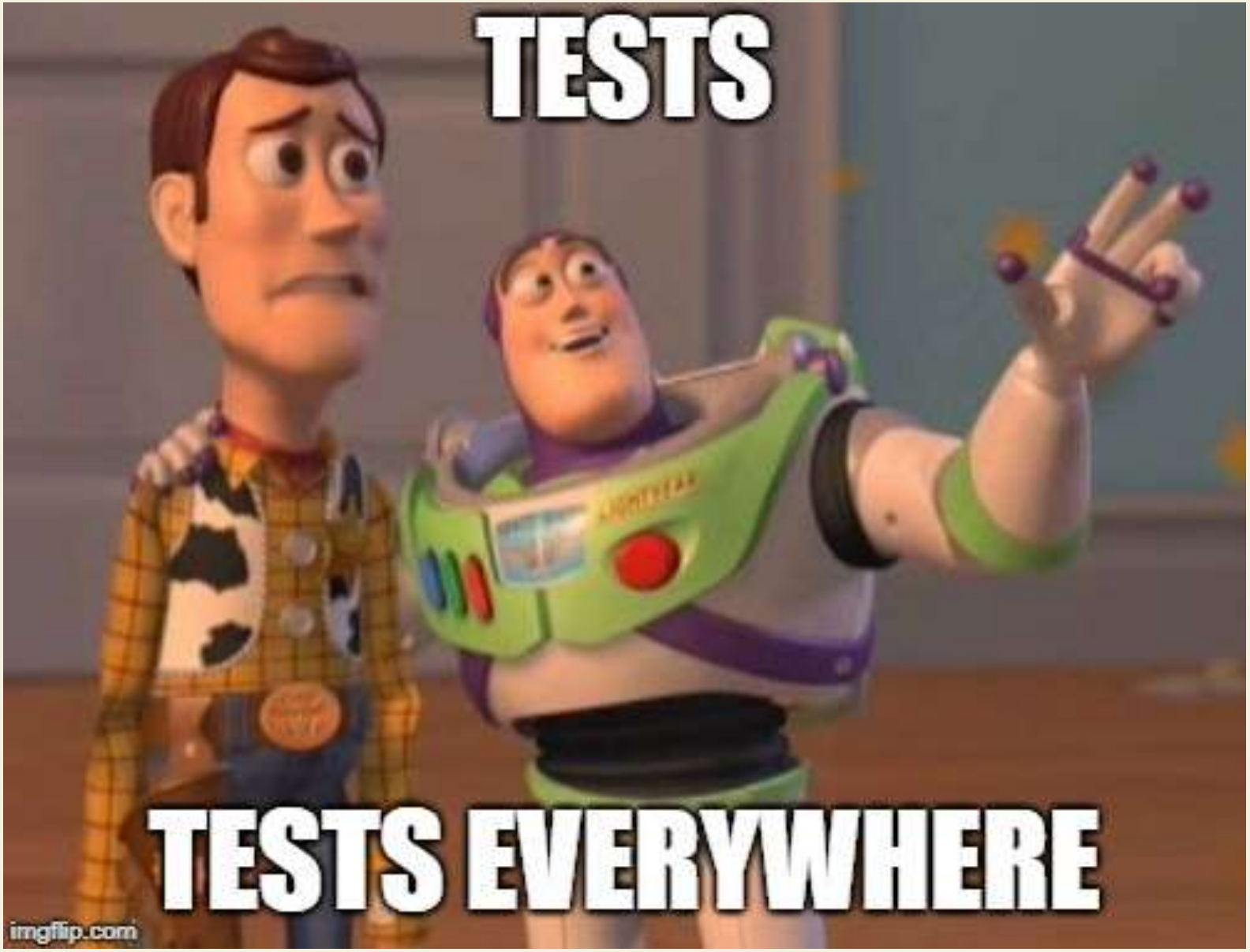
What to track

- “Main” Workflow coverage
- Area stability
- Workflow stability
- Manual regression testing time
- Time to feedback
- Deliverability
- Escaped bugs that we could have found(by customer or internally)

What to do

- Add tests where needed
- Move “older stable” area tests to later build cycles
- Move “newer unstable” area tests to earlier cycles

The suite level



@gil_zilberfeld

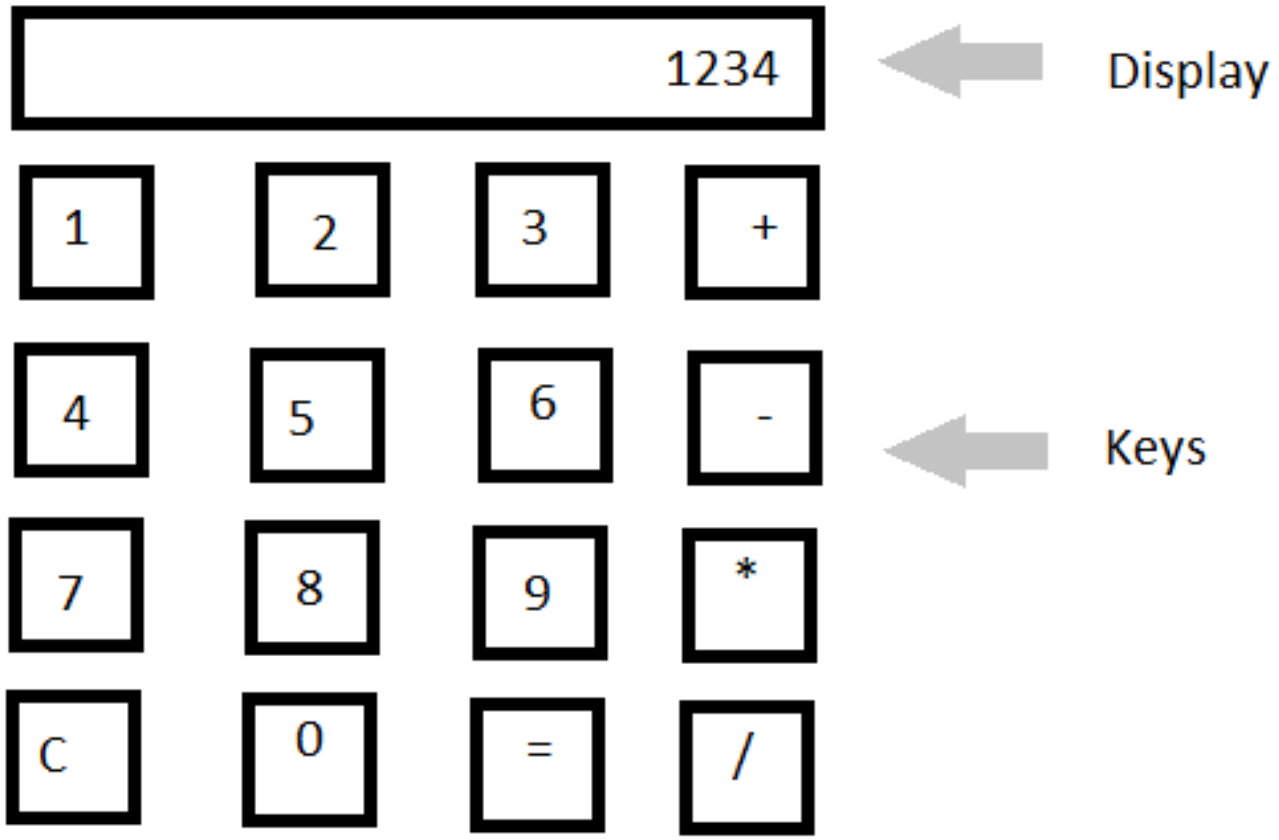
Test suite clean up

- Organize sanity, regression and acceptance
- Delete “Ignored” tests
- Fix grouping for test cohesion

Those annoying tests


SO I WRITE THE TEST ONCE AND DON'T LOOK AT IT EVER AGAIN
YOU KNOW THERE'S A BETTER WAY, RIGHT?
imgflip.com

Calculator Display Requirements



Exercise: Unit Tests

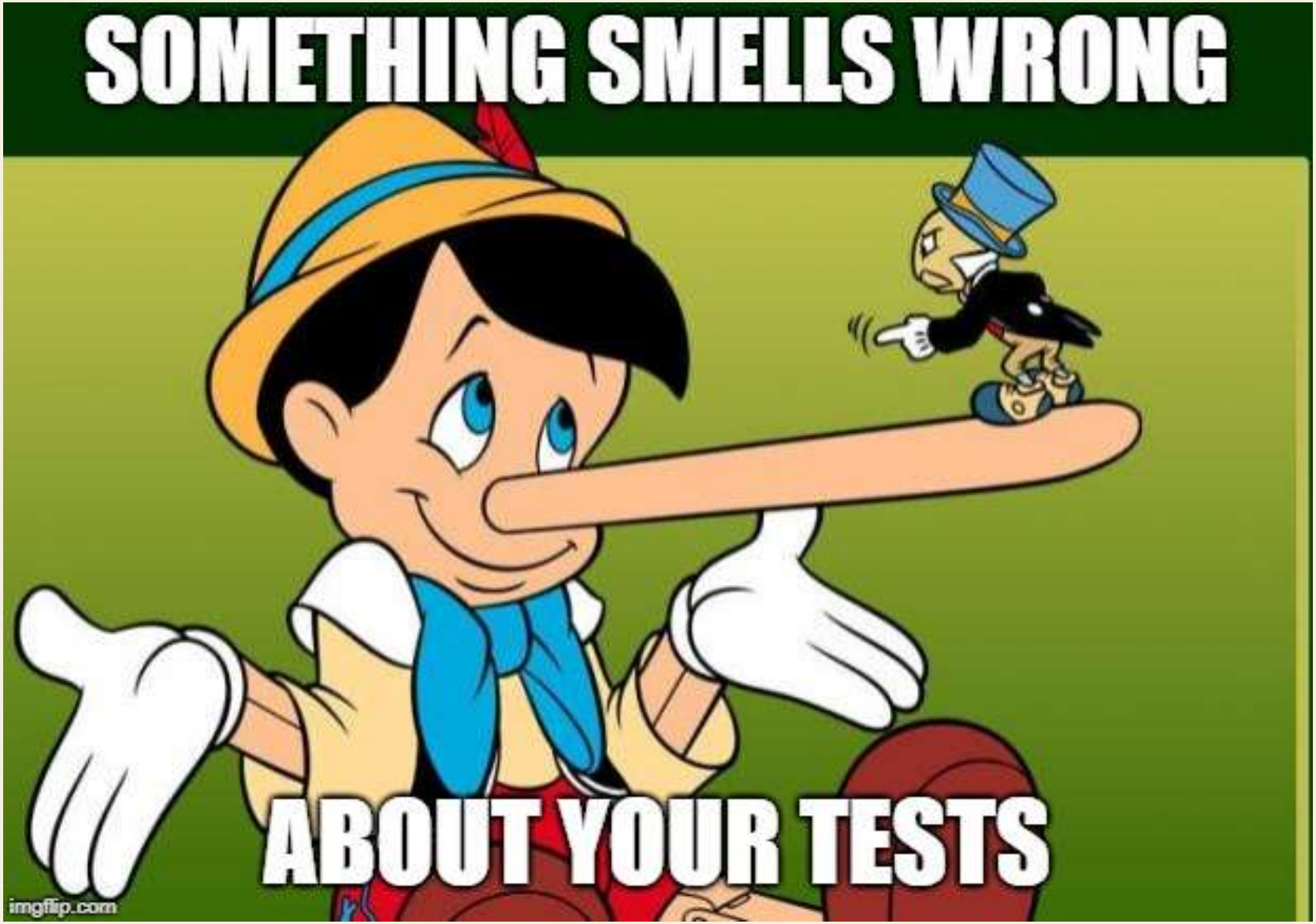
- Find issues
- Refactor and/or make suggestions


THE LOOK YOU GET
WHEN YOU NOTICE
THOSE TEST SMELLS
imgflip.com

Uninformative tests

- Tests that don't point to the problem
- Not enough information on failure
- Checking too many operations
- No overlapping between test types

(triangulation)


SOMETHING SMELLS WRONG
ABOUT YOUR TESTS
imgflip.com

Unreliable tests

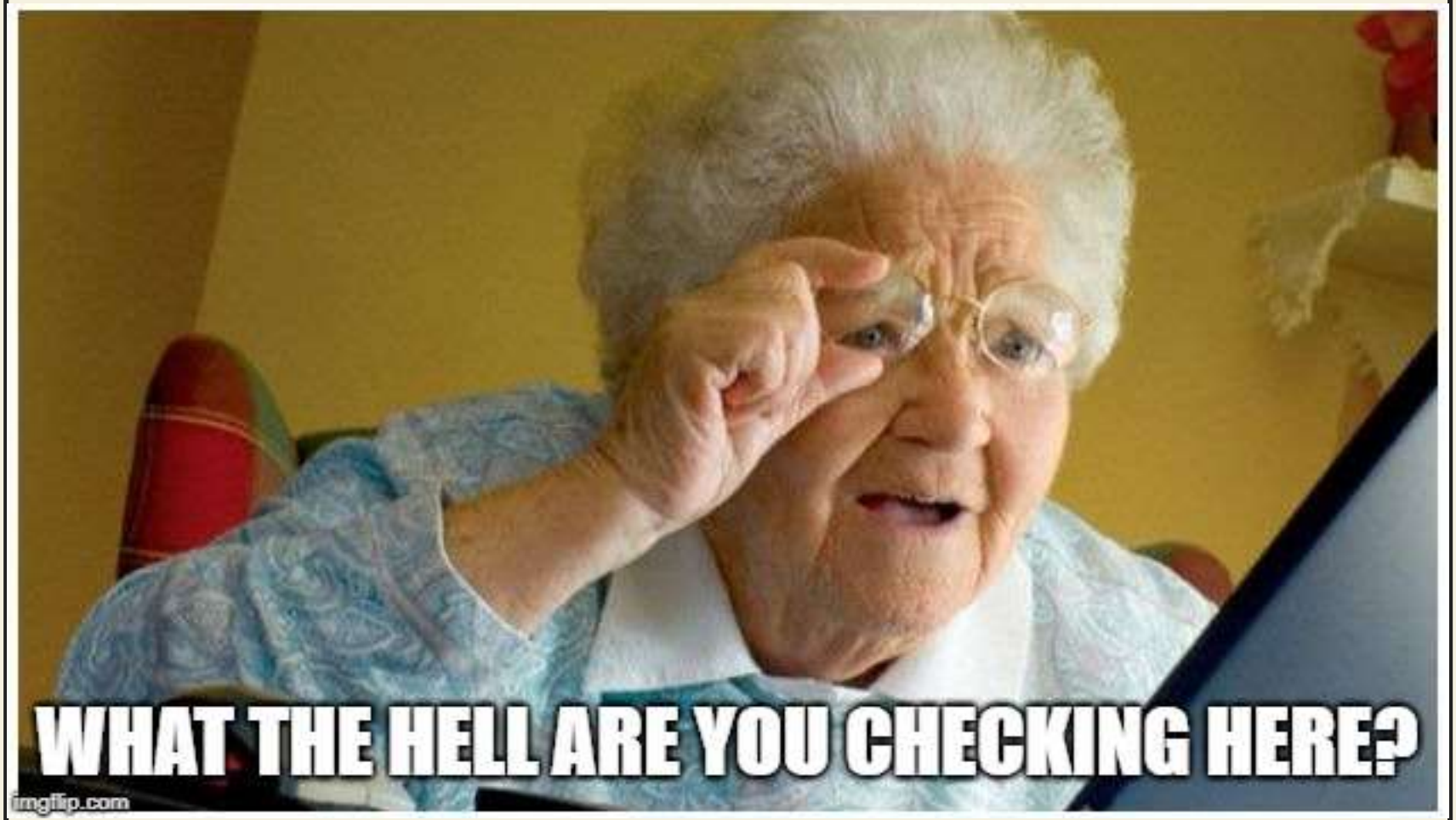
- Inability to run everywhere, anytime
- Flaky tests
- Dependency on unstable resources, platforms
- Test isolation (resources, initialization, cleanup)

Misleading tests

- Data transformation
- tests that always pass
- Checking the wrong thing
- Unclear test names
- Readability

Misleading tests

- Logic in tests
- Code matching
- No assertions

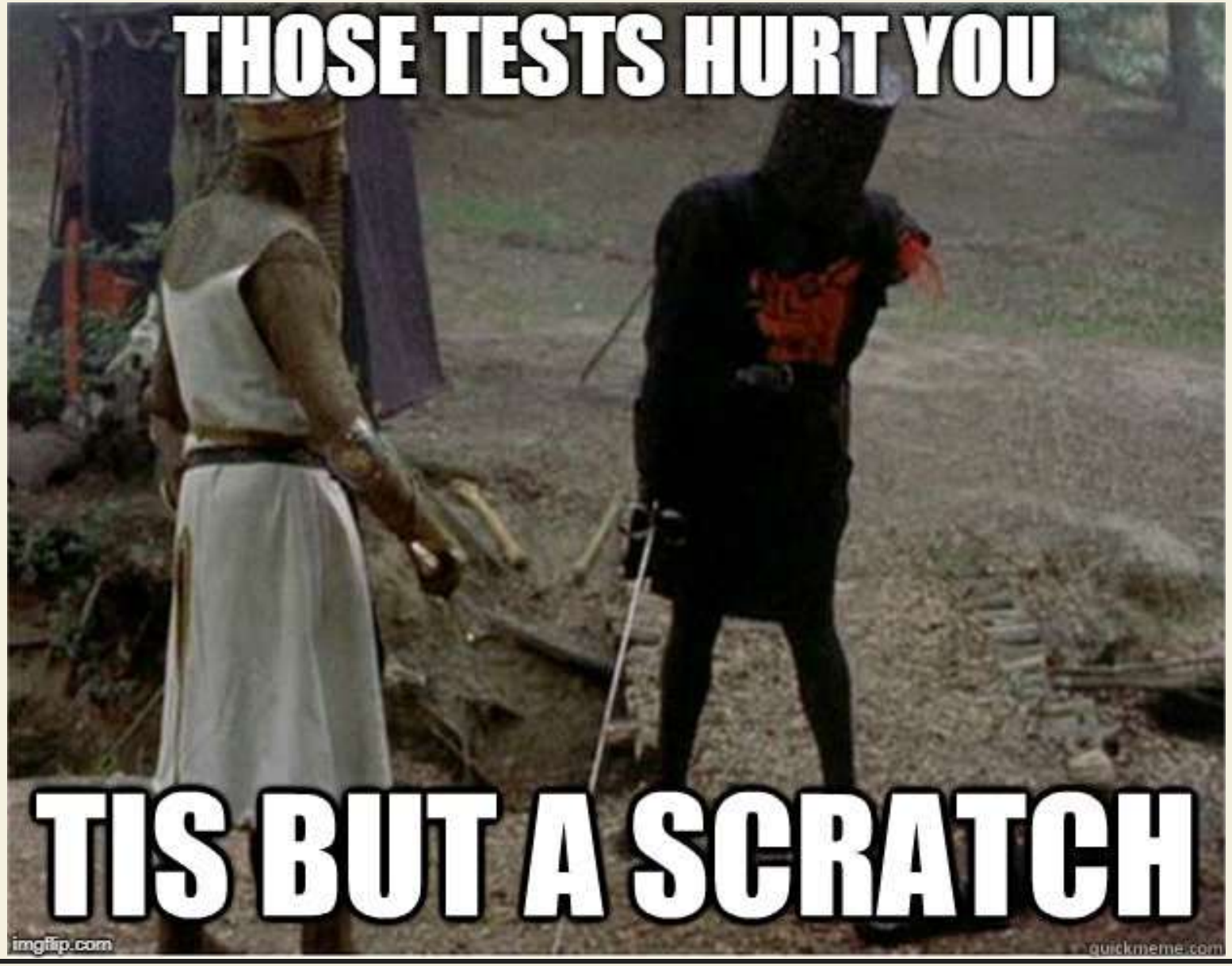

WHAT THE HELL ARE YOU CHECKING HERE?
imgflip.com

Maintenance issues

- Copy-paste, duplication
- Where do I add the next test
- Tests that do same thing “just to be on the safe side”
- Verbose and big setup, no framework (Page object)

Tests that hurt you

- Test run length
- Build cycle run length


THOSE TESTS HURT YOU
TIS BUT A SCRATCH
imgflip.com
quickmeme.com

Exercise: Integration tests

- Make suggestions on the test suite
- Also consider the unit tests!
- Make changes for improvement

@gil_zilberfeld

Test reviews

Why?

Why review?

- Find problems early
- Fight complexity
- Knowledge sharing
- Collective code ownership
- Maintain conventions

Ask: Can I maintain this code?

Who reviews?

- Team lead
- Architect
- Developer
- Peers
- The person who can bring the most value

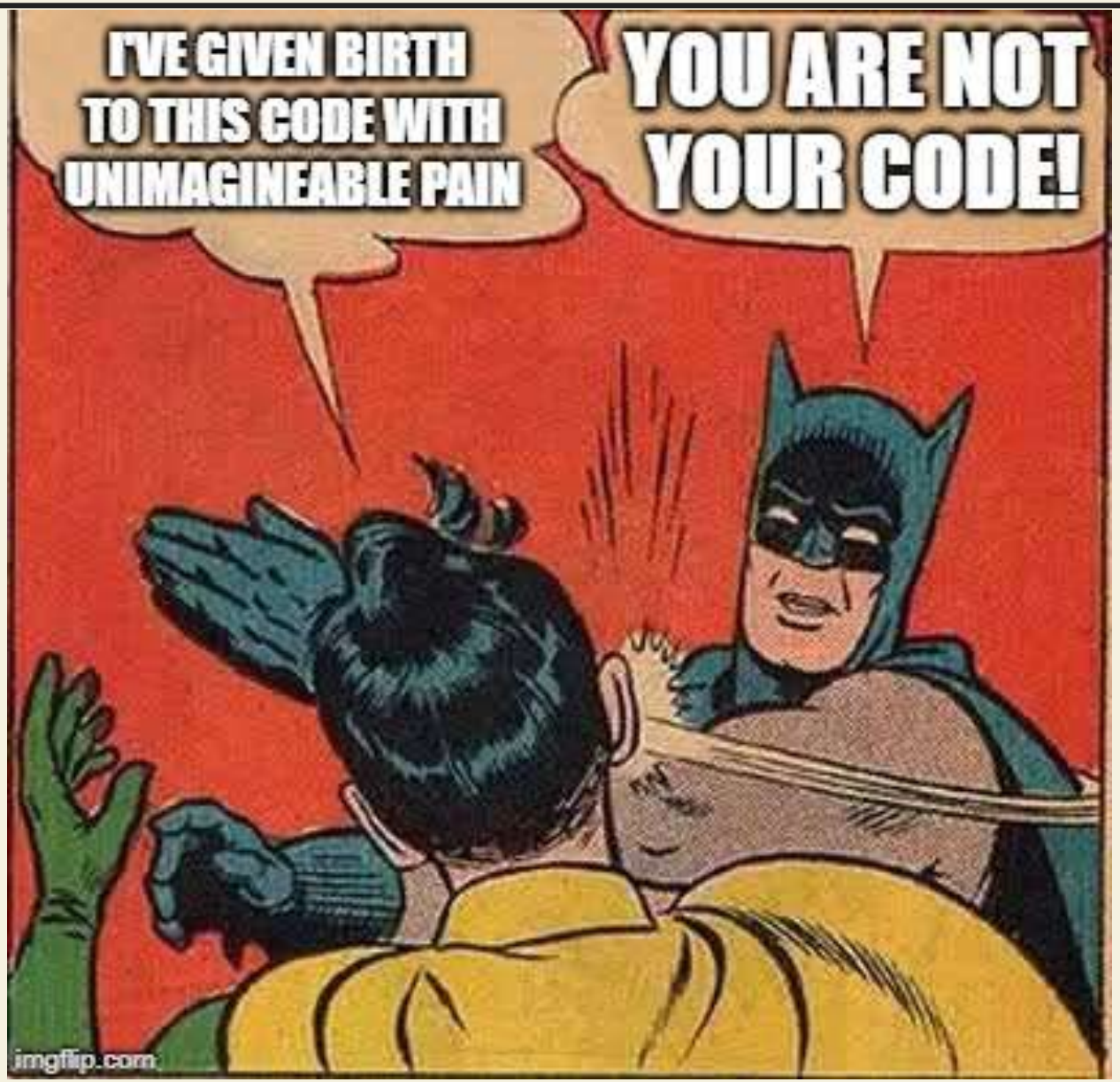
@gil_zilberfeld

Short reviews

- Short reviews means higher availability
- No more than 60 minutes
- No more than 200 lines of code
- Less is better

Async review

- Smell of attitude
- Better together
- Better for distributed teams
- Consider the commit status



@gil_zilberfeld

Errors will be found!

@gil_zilberfeld

When to review?

- Before committing
- After merging
- When you're ready

What to review?

- Functionality
- Are all necessary cases covered?
- Are they written correctly?
- Is the test run time quick?
- Remove ignored tests

What to review?

- “What if” scenarios
- English
- Performance
- Impact on other systems
- Dependency on other systems

What to review?

- Compliance
  - Conventions
  - Tools
  - Legal

@gil_zilberfeld

Pair programming and testing

- Better together
- No need to review!
- Be nice

Exercise: Review the changes

- Pair with someone
- Express views
- Try to stick to the patterns

@gil_zilberfeld

Thanks!

ANY QUESTIONS?

You can find me at:

@gil_zilberfeld

http://www.GilZilberfeld.com

http://www.EverydayUnitTesting.com

http://www.fastee.im