Gil Zilberfeld (TestinGil)

Hello!

I AM GIL ZILBERFELD

http://www.testingil.com/

Twitter: @gil_Zilberfeld
Telegram: @TestinGil

Download the code from GitHub

A LONG
TIME AGO
IN A GALAXY
FAR, FAR AWAY...
AT A
TEST.IL MEETUP
imgflip.com

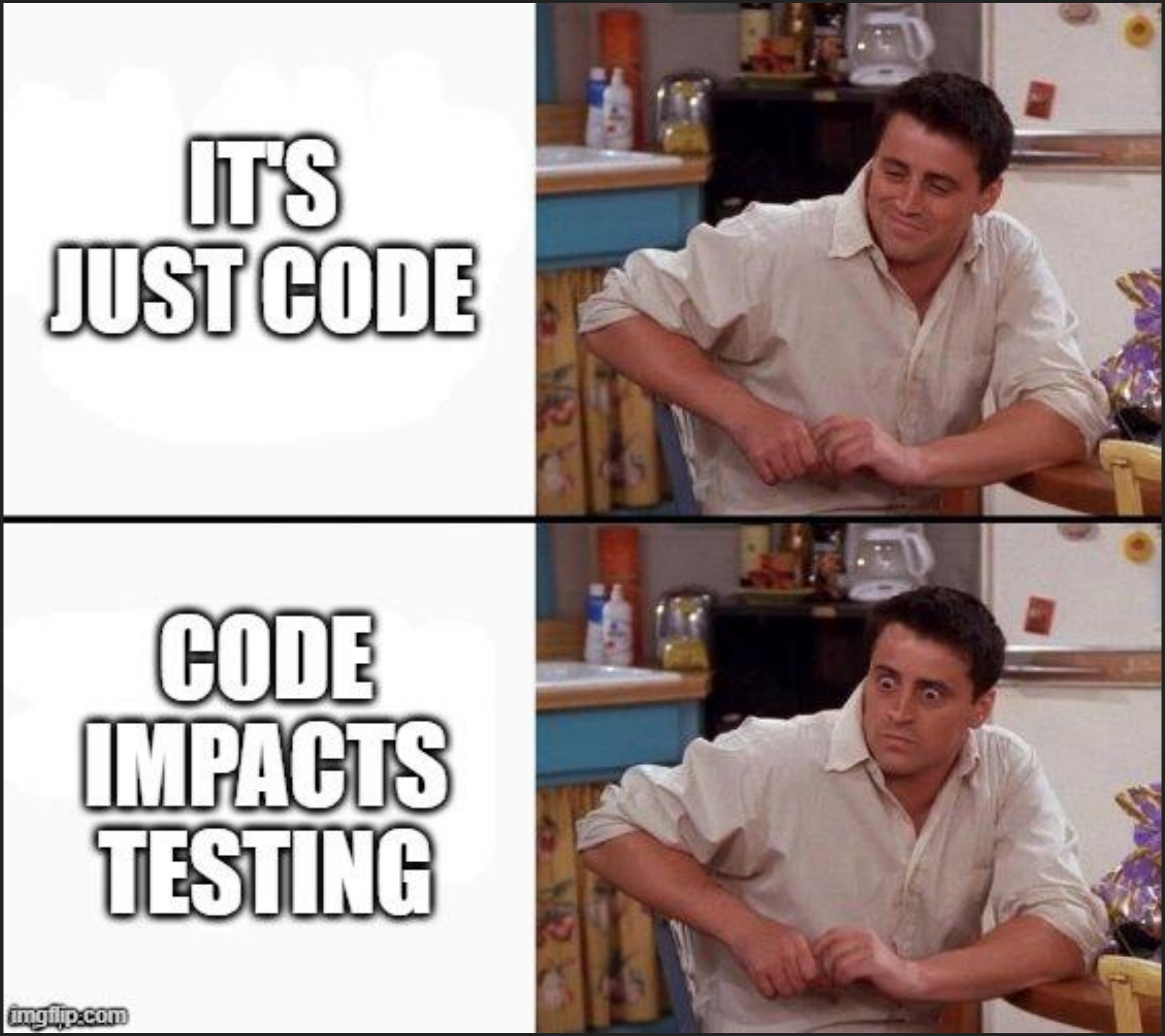
THE CONVERSATION
imgflip.com

The conversation

- Developers don’t want to change code for tests
- The design is so brilliant already
- They've worked so much at it
- And many more…

Fallacies and misconceptions

- The job is done when code gets pushed
- The black box fallacy
- Code and architecture doesn’t have any visible correlation with testing


IT'S JUST CODE
CODE IMPACTS TESTING
imgflip.com

Developer blind spots

- How the code will be tested
- In what context
- The cost of testing
- The effect of developer testing on system tests

Videos
Maps
Weather
Presidential Alert
NOT MY PROBLEM
Dismiss
iTu
FaceTime
imgflip.com

Produce code

That works

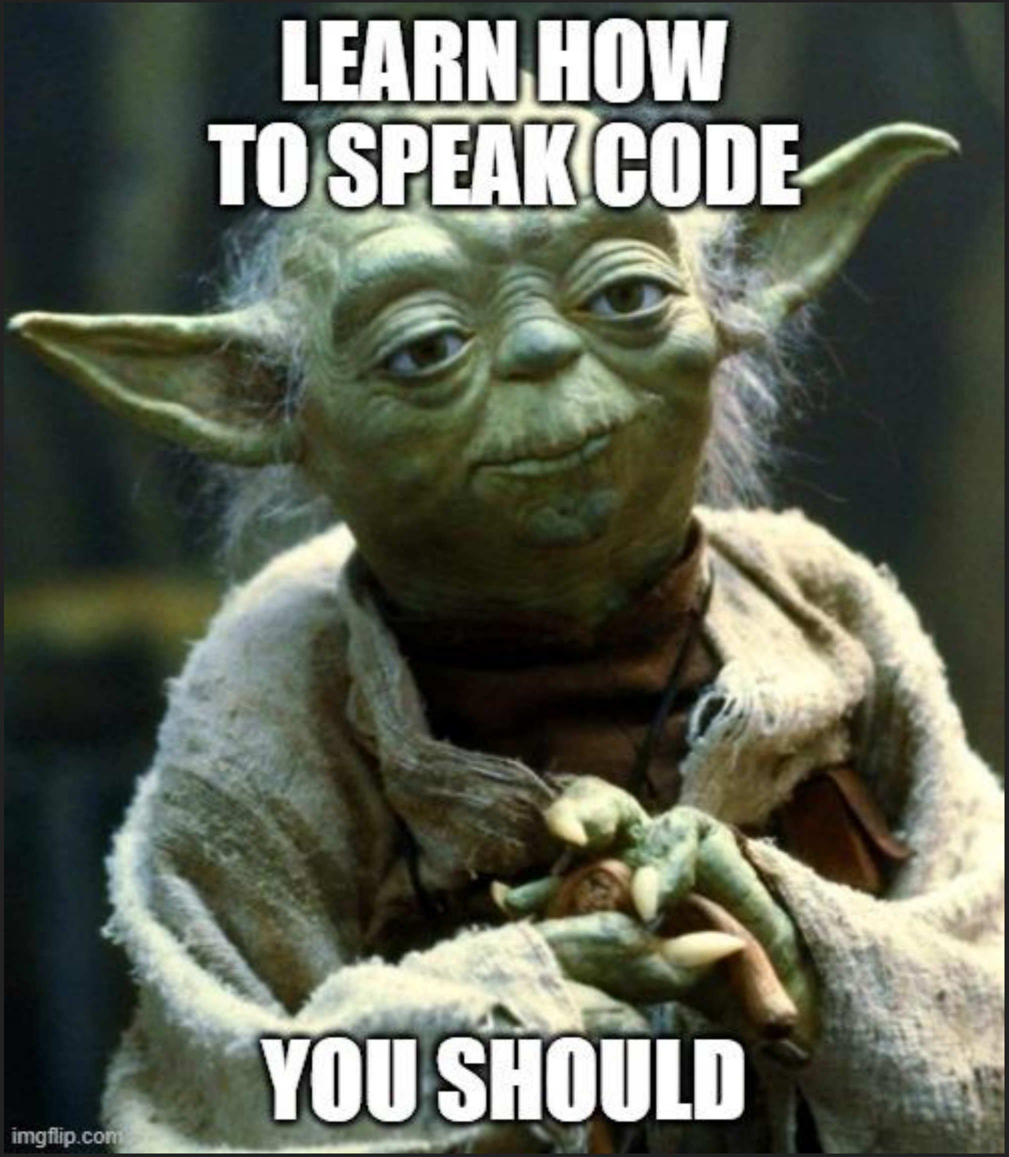
And easy to maintain

Videos
Maps
Weather
Presidential Alert
MY PROBLEM
Dismiss
iTu
FaceTime
imgflip.com

Report the most comprehensive information to stakeholders

How do we align the mission statements?

@gil_zilberfeld


LEARN HOW
TO SPEAK CODE
YOU SHOULD
imgflip.com

DEMO DAY
imgflip.com

NYPD
DUPLICATION
imgflip.com

Duplication

- The problem
  - Same code appears in multiple locations
- The impact
  - We need to check multiple scenarios for better coverage and reporting
  - Both internal and APIs
  - More testing effort, and usually less coverage
- Let's see an example

Duplication

- The solution
  - Identify the pattern
  - Refactor into libraries
- But, but, but…
  - No buts
  - Really

VISIBILITY
imgflip.com

Visibility

- The problem
  - Cannot call code directly or access it for assertions
- The impact
  - We need to write complex scenarios for better coverage and reporting
  - If we can at all
  - More setup effort, and usually less coverage
- Let’s see an example

Visibility

- The solution
  - Make it public!
  - Expose methods and data through APIs
- But, but, but…
  - “Someone may call it”
  - Using a programming language construct for hiding something should really be the last resort
  - APIs need management
  - Maybe it should be public

DEPENDENCY INJECTION
imgflip.com

Dependency Injection

- The problem
  - Dependencies are created and controlled only inside the code
- The impact
  - We cannot modify, mock or replace them
  - Test scenarios cannot run
  - Or require mind-bending setup
  - In which case, you probably just won’t run it
- Let’s see an example

Dependency injection

- The solution
  - Pass the dependency as a parameter
  - Or use a DI framework
  - Or use a redirection configuration
  - Now we can use mock dependencies for different scenarios
- But, but, but…
  - “My brilliant design!”
  - Is not testable, therefore its brilliance is in question
  - By the way, Spring is a dependency too

MOCKABILITY
imgflip.com

Mockability

- The problem
  - Dependencies are created without ability for extension
- The impact
  - We cannot mock or replace them
  - Test scenarios cannot run
  - Or require mind-bending setup
  - In which case, you probably just won't run it
- Let's see an example

Mockability

- The solution
  - Don’t make them final (or sealed, or whatever)
  - Use replaceable dependencies with DI like in-memory databases and mocked servers
- But, but, but…
  - “My brilliant design!”
  - You keep saying that word, I’m not sure it means what you think it means

MODULARITY
imgflip.com

Modularity

- The problem
  - Code has a lot of dependencies in it
  - Breaking the SRP and ISP (SOLID)
  - Services are big
- The impact
  - Test setup requires supplying many dependencies than the tested code really needs
  - More setup effort, if we don’t break in the process
  - Test run is longer
- Let’s see an example

Modularity

- The solution
  - Break it down!
  - Smaller code and small services
  - Less dependencies mean smaller setup
- But, but, but…
  - “My brilliant design!”
  - Really means: “I’m not changing my code for you (and maybe it’s too risky)”
  - It makes for other problems like god files, classes and methods. Maintenance is horrid. Don't do that.

Being part of the conversation

Before, during and after coding

Joint meetings

- Design reviews
- Code reviews
- Test planning
- Test design
- Test results
- Demos

QUESTIONS
imgflip.com

Asking questions - Design

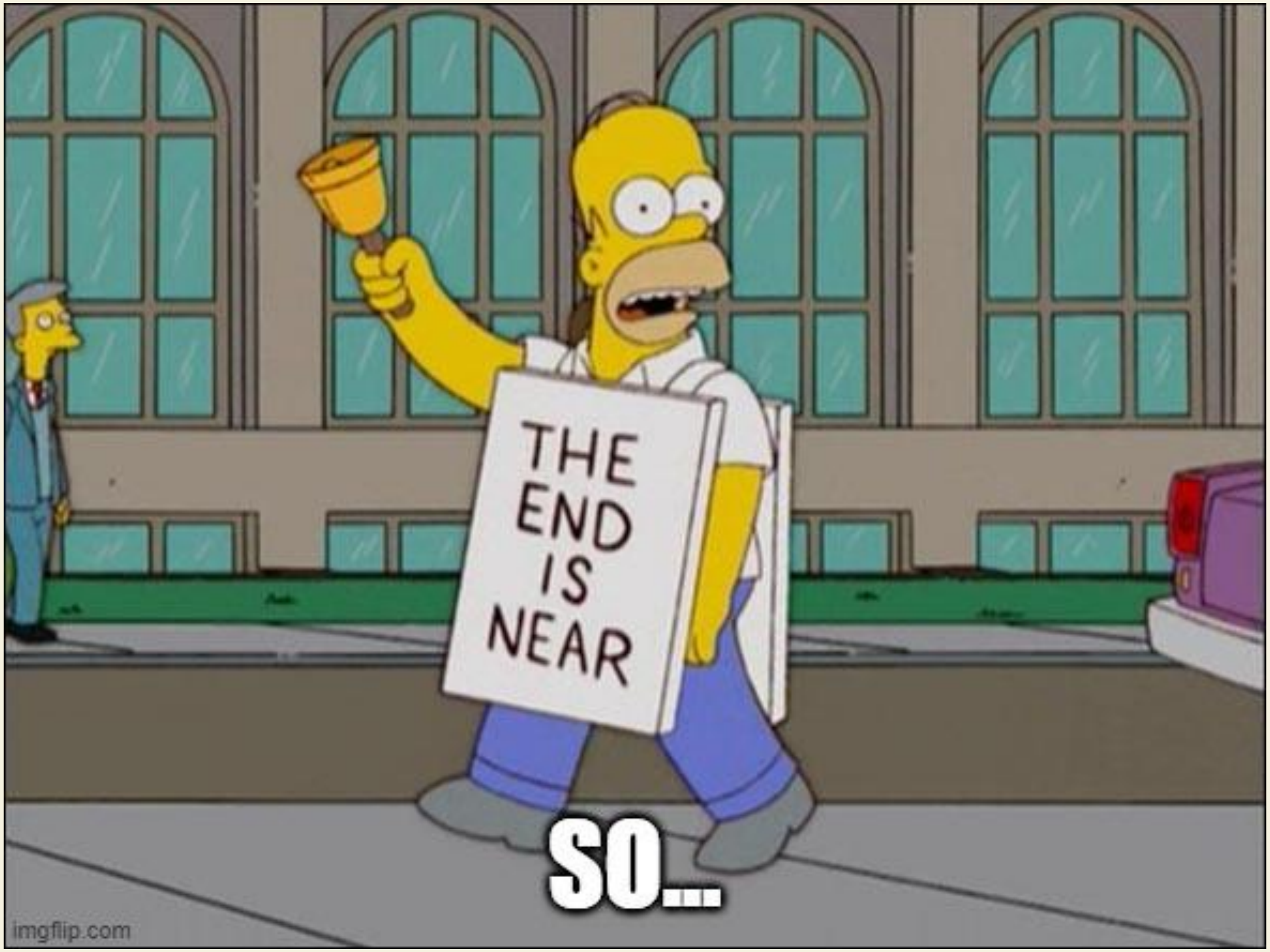
- Where is the risky code?
- What old features will be impacted by the new code?
- What are the dependencies?
- Will we be able to "mock" them?

Asking questions - Planning

- What scenarios are going to be automated?
- By what?
  - Unit, integration, API
- What don’t they cover?
- What is the cost for improving testability?
  - What do we get in return?
- What should we prepare so we can test scenarios after that?

Retros

- What makes it hard to test?
- What makes it easy to test?
- Also, define hard and easy


THE END IS NEAR
SO...
imgflip.com

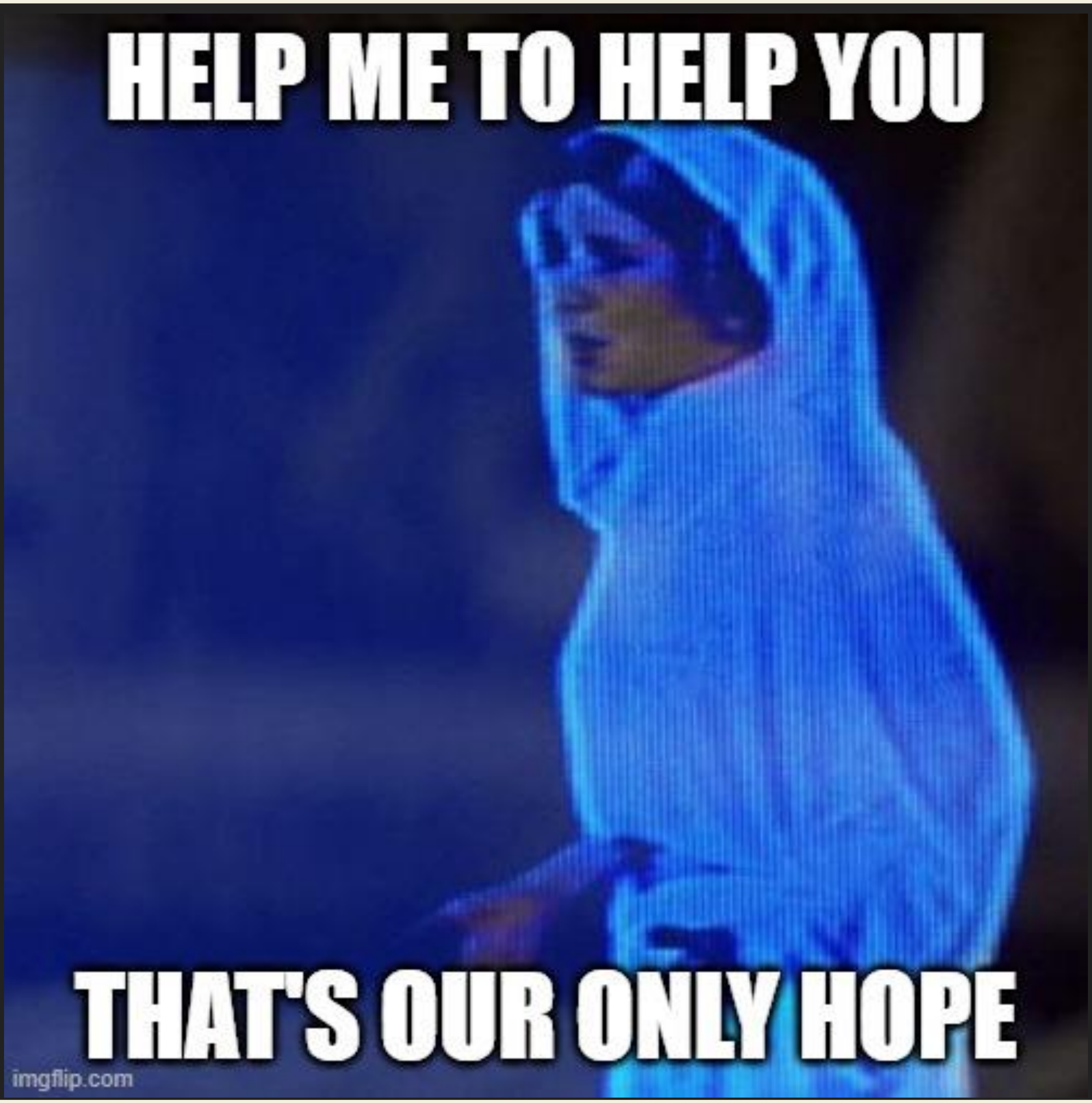
"If you brought us a functionally correct design that required our app to run on a million AWS instances, we'd casually say "no, that's not valid."

https://www.geepawhill.org/2020/12/11/the-steering-premise/

If a design is not testable, it is not a valid design

We need to help the developers do their job

@gil_zilberfeld


HELP ME TO HELP YOU
THAT'S OUR ONLY HOPE
imgflip.com

Thanks!

ANY QUESTIONS?

You can find me at:

LinkedIn: Profile, Company

XING: Profile, Group:

Twitter: @gil_zilberfeld

http://www.testingil.com