MIHAI'S TESTING LECTURE @ DBC

TESTING / RSPEC

LECTURE PLAN

- What and why?
- Two Testing Trinities
- When and how?
- Intro Mihai's Testing Challenge
- … you try it
- Live code Mihai's Testing Challenge

WHAT?

- The Video
- Tests are production code
- RSpec vs Test/Unit
- TDD
- BDD

WHY NOT TEST?

- Don't know how / Never learned
- It takes time to write specs …or does it?
- Quick-and-dirty temp code / No users / No consequences

WHY TEST?

- Mihai's tale of two jobs
- Faster REPL/feedback loop
- Feels safe
- Avoid bugs (and therefore keep customer/boss happy)
- Lets other devs know the intent of your code (!comments)
- Encourage thinking ahead, writing simpler code
  - "What am I trying to do here, anyway?"

RED / GREEN / REFACTOR

- Always start by writing a failing test
  - If green, I will comment out code or change expected value
- Get it to pass, however you can
  - A great time to git commit
- Then refactor safely
  - Run the specs again, obsessively, each time you make a change

GIVEN / WHEN / THEN

- This is the general approach/philosophy of testing
  - And also specific to Feature Testing (Google Gherkin spec)
- GIVEN – Although I have control, I like to test this anyway
- WHEN – run your method
- THEN – test the outcome

WHEN AND HOW?

- Test Eventually vs. Test First
- Test Manually vs. Test Automatically
- What to test in a class
- Destructive vs. Returning Methods
  - What is the outcome we care most about?

RSPEC

- describe / context / it
  - write helpful strings to follow these
- Google "Rspec matchers" or "Rspec cheatsheet"
  - be careful/curious about arrays, Booleans, change
  - let vs. let! vs. before(:each) vs. new local variables
- blocks vs. calling methods