Lecture

Intro to TDD (Test driven Development)

basically testing before programming anything. Follows red green refactor loop – everything willl fail, starts red, writes minimal code to make the test pass. if it passes you go back and refactor and if it passes again you go back and repeat

How to know someones code does what its supposed to do?.. you write tests for it so you can test.

Entire test this week is based on TDD. – we write a test for their code. We should do everything in TDD style.

gem is just a library of ruby code.

regression is when someone new joins your group and writes their own feature into your program and when they try to merge it breaks previous code.

bundle exec respec spec/ \*rb 🡪 we should always test the entire test sweep. not single files at a time, because new tests can break previous test codes.

TDD style is different then usual

* first thing we do is describe what were doing.
* describe ‘Product’ do
  + describe ‘#new’
* if you see the word new anywhere in the describe like product = Product.new(xx xxx xx)
* you know we have to initialize something – so directly we go to the Product class and
  + def initialize(name: name, price\_in\_cents: price\_in\_cents)
  + end
* the name: product\_name, price\_in\_cents: price may seem strange in the initialize, this is because of readability, it is just the name of the parameter and the value. Imagine if you have 15 parameters, this makes it easier.
* next says expect(product.name).to eql ‘Book’ 🡪 so we add an instance variable in the Product class 🡪 @name = name, and @price\_in\_cents = price\_in\_cents
* RULE OF THUMB – make EVERYTHING match from the test to your file.
* for the second test – says it must throw exception if name and price are missing(nil)
* so after the initialize we add –
* if name.nil? || price.nil?
  + raise ArgumentError
* end
* if we run this there will still be an arror – because in the test case it says it must raise an ‘InvalidProductError, “Invalid Product”
* so change the raise line above to:
  + raise InvalidProductError, “Invalid Product”
  + but if we run this it will still fail because the program doesn’t know what InvalidProductError is so before even the class Product we add an error class that inherits StandardError class
* class InvalidProductError < StandardError
* end
* Rest in lecture code in email.
* inside tests, they can actually define fake variables not found anywhere else. But you can still use this fake variable to make its own test pass.
* Test code is basically describing everything it wants to do at every level. Other developers looking at your program can use this as documentation to see what youre doing.

Breakout – TDD –