More Automated Tests

Admin

- Midterm
  - Take home or in class?
- Odd Schedule next week

Why Test?

- You are beginning to do some automated tests
- So have we answered the question why test?

Why Test?

- You are beginning to do some automated tests
- So have we answered the question why test?
  - Or have you just surrendered to the fact that I made you do it
- Hopefully you start to see how it can be helpful

How should we test?

- The How should we test question is easy
  - **If** the function just takes a param for the input and returns a value as output.
  - We've seen that
  - Its a big **IF**
  - Right?

Mocks

- A Mock is a fake/test (I know) version of a module
  - Or even a function
- E.g.
  - We mock the database connection and tell create a fake database module that will return just our test data
- Mock library built into python3
  - Was part of the unittest module in legacy python
- Mock part of JUnit in Java (superseded by mockito today)

Should we Mock?

- Lots of people have been calling for limiting Mocks in recent years.
  - http://blog.metaobject.com/2014/05/why-i-don-mock.html
  - https://medium.com/javascript-scene/mocking-is-a-code-smell-944a70c90a6a
- Even Martin Fowler is pushing for a different interpretation of Mocks
  - https://medium.com/javascript-scene/mocking-is-a-code-smell-944a70c90a6a
- So if we don't want to Mock willy-nilly – when should we?

When should you Mock

- Mock at least when:
  - the act of testing is going to use a consumable resource
    - Print an invoice
  - Money is involved
    - Charge a credit card
  - Danger is involved
    - Launch missile
  - Privacy is involved
    - Print medical records
- Did I miss any?

Mocks

- Still want to run these tests without Mocks on rare occasions
  - But most tests can use mocked versions of these services.

Mocks, Stubs and Fakes

- Lots of debate in the community today about Mocks, Stubs and Fakes,
  - (See Fowler link in previous slide)
  - What do we mean by these?

Mocks, Stubs and Fakes

- Lots of debate in the community today about Mocks, Stubs and Fakes,
  - What do we mean by these?
  - Usually:
  - Stub: function/class/service is replaced with one that has little/no functionality
  - Fake: function/class/service is replaced with one that responds with very much fake data
    - Like my post to slack
  - Mocks
    - allow for testers to provide specific response for given input

Mock Example

- Java example from Dmitriy Yefremov
  - `public interface Phonebook {`
  - `  String setNumber(String name, String number);`
  - `  String getNumber(String name);`
  - `}`
  - And now
  - `import static org.mockito.Mockito.*;`
  - `// create a mock`
  - `Phonebook phonebook = mock(Phonebook.class);`
  - `// set expectations`
  - `when(phonebook.getNumber("Alice")).thenReturn("123 4567890");`
  - `Possibly more when statements here.`

Monkey Patching

- What are we talking about?

Monkey Patching

- What are we talking about?
  - Monkey patching: changing the way code works at run time
  - Why Monkey patching?
    - Two contenders
    - Monkeying around with the code
    - Pun on stealthy guerrilla patching of code

Monkey Patching

- Most Modern Languages support Monkey patching
  - Python, ruby, C#, and workarounds in golang
  - Older languages no: C, c++, JAVA
  - (objective-C did support it)
  - Kotlin doesn’t support this but there is discussion in the community.
    - Kotlin??
    - https://proandroiddev.com/kotlin-a-massive-leap-forward-78251531f616
  -

Usual use: Monkey Patching

- We are using your library
  - Your library has a bug
  - Or doesn’t provide a feature
  - We write code in our code that is injected into your library for our program
  - Your library now works differently till the process ends

Testing: Monkey Patching

- In python (and some other languages) monkey patching can be used in place of Mocks
  - In fact with pytest this is preferred.
  - But we don’t want monkey patches to last till end of process, so pytest uses test specific monkey patches
  - and then puts things back when test function ends.