Fixtures vs. Factories

- Fixtures are set
- Factories make the object
- Fixtures are fast
- Factories are dynamic; will call the object create method, which could have lots of callbacks

Factory Tools and examples

- Factory Girl is the most commonly used Rails factory tool - https://github.com/thoughtbot/factory_girl
- Curate code example for factories https://github.com/projecthydra-labs/curate/tree/develop/spec/factories

Fixture Tools and Examples

- Create data directly as objects in code
- Separate structure and behavior
- Fixtures load data directly into the database
- Destroy all software videocast, don't use either fixtures or factories

Mocking and Stubbing

- What is the difference between a mock and stub
    - Mock Objects and Stubs according to RSpec "The names Mock Object and Test Stub suggest specialized Test Doubles. i.e. a Test Stub is a Test Double that only supports method stubs, and a Mock Object is a Test Double that supports message expectations and method stubs."

Questions on Mocks and Stubs

- How do you handle scenarios when you test many objects?
- How much do you use mocks and stubs in tests?

Code Example from Hydramata::Works

- DatabaseCoordinator from Hydramata::Works and corresponding spec

VCR & HTTP

1. Gem https://github.com/vcr/vcr
2. Return the body and head of the "request"
3. VCR replays an HTTP request
4. Record a VCR Test
5. Example or a fedora request from the hydramata-works projects
6. Tests don't have to rely on a remote connection
7. Hydra::RemoteIdentifier
    a. Spec using VCR and Cassettes that are recorded
8. Customizing VCR to ignore_localhost (via Curate gem)

9. APIs do change frequently
    a. Setup small test suite that runs on occasion tests against the live remote service
10. VCR can ignore certain hosts

JS Testing

- Jeremy hates javascript
- Selenium is an option, you can record your browser interaction
- Selenium can be incompatible with file controllers
- https://github.com/teampoltergeist/poltergeist - phantomjs driver for Capybara testing
- Audrey from dpla uses unit testing frameworks for Javascript
- Javascript tests can be finicky with selenium due to browser timing issues often js testing requires wait calls
- editorial comment - javascript is a joyless ecosystem
- How slow is too slow? Testing suites can be too slow

Other Testing Things

- Integration tests ask “have I done the right thing?” - these are usually slower tests
- Unit tests assert “am I doing it the right way”. - these should run faster
- Tests can facilitate rapid development
- Tests are your documentation for a project
    - Read your tests with an eye towards documentation
- Code coverage - a well documented application should have high code coverage
- High code coverage means you can change things with a high degree of confidence
- Write tests that have value for you
- Tests are for future proofing for your app

Questions

- How do you figure out why tests are slow?
    - “rspec -p”https://docs.google.com/spreadsheets/d/1gPWapLaxtYxn7cBTdI-Ecdjj_HpqqEm3abMY3m88QSU/edit?usp=sharing can profile a unit test suite
    - minitest plugin can check the garbage collection at the end of a test
- Adam - feature tests were very slow because the before setup blocks were firing over and over again
- Doing single tests for “feature” tests can speed things up
- Unit tests tests should have a single assert per test
- Site Prism capybara test - https://github.com/natritmeyer/site_prism
    - Jeremy gives example of specs using SitePrism and Page object models via Site Prism