

Fixtures and Mocks

Managing test data in Rails

Writing a Model: Step 1: Migration

db/migrate/20080826203607_create_authors.rb

```
class CreateAuthors < ActiveRecord::Migration
  def self.up
    create_table :authors do |t|
      t.string :name
    end
  end

  def self.down
    drop_table :authors
  end
end
```

Writing a Model: Step 2: Model

app/models/author.rb

```
class Author < ActiveRecord::Base  
end
```

Writing a Model: Step 3: Test!

test/unit/author_test.rb

```
require 'test_helper'
```

```
class AuthorTest < ActiveSupport::TestCase
  def test_create_author
    author = Author.create(:name => 'Maurice Sendak')
    assert author.saved?
  end
end
```

What about retrieving?

We could do this:

```
test/unit/author_test.rb
```

```
class AuthorTest < ActiveSupport::TestCase
  [...]
  def test_retrieving_without_fixtures
    name = 'Maurice Sendak'
    Author.create(:name => name)
    assert Author.find_by_name(name)
  end
  [...]
end
```

That worked, but What if...

- * You have a more complex model with a lot more fields?
- * The model has associations with other models (e.g. our author has written some books) ?
- * We need to use the same model instance in many tests?

Fixtures: Using the fixture

yml - the very simple Yet Another Markup Language

example:

test/fixtures/authors.yml

```
maurice_sendak:  
  name: Maurice Sendak
```

Fixtures: Using the fixture

test/unit/author_test.rb

```
class AuthorTest < ActiveSupport::TestCase
  fixtures :authors
  [...]
  def test_retrieving_with_fixtures
    assert_nothing_raised do
      Author.find(authors(:maurice_sendak).id)
    end
  end
end
```

Fixtures: More Complex

```
./script/generate model book title:string isbn13:string
```

```
./script/generate model role book_id:integer \  
  author_id:integer
```

app/models/author.rb

```
class Author < ActiveRecord::Base  
  has_many :roles  
  has_many :books, :through => :roles  
end
```

(do the reverse to app/models/book.rb)

Fixtures: More Complex (cont)

test/fixtures/books.yml

```
where_the_wild_things_are:  
  title: Where the Wild Things Are  
  isbn13: 978-0060254926
```

test/fixtures/books_authors.yml

```
maurice_sendak_wrote_where_the_wild_things_are:  
  book: where_the_wild_things_are  
  author: maurice_sendak
```

Wait... what?

Fixtures: Problems and Killer Apps

Problems

- * Fixtures make my test suite slow
- * Fixtures allow invalid data
- * Maintainability Challenges
- * Fixtures are brittle - oh the untold hours
- * Only works on databases. What about testing code that gets data from a web services?

But fixtures have a Killer app

- * test data for methods that use `find_by_sql`

Mocks

Isolating your testing unit

* Why should your controller test depend on your models to be working correctly?

test/functional/book_controller.rb

```
class BooksControllerTest < ActionController::TestCase
  fixture :books

  def test_show
    book = book(:horton_hears_a_who)
    get :show, :id => book.id
    assert_response :success
  end
end
```

Sidebar: Installing Mocha

```
gem install mocha  
at the top of 'test/test_helper.rb' add  
  
require 'mocha'
```

Using a mock

```
class BookControllerTest < ActionController::TestCase
  def test_show
    book_1 = mock('book') # 'book 1' is just a nice label when
                          # looking at exceptions
    Book.expects(:find).with(1).returns(book_1)
    book_1.expects(:title).returns('title')

    get :show, :id => 1
    assert_response :success
    assert_select 'h1#book_title', 'title'
  end
end
```

Source Code

Slides will be linked on Meetup later tonight

http://github.com/myerscarpenter/fixtures_and_mocks