# Ruby on Rails Tutorial
# Chapter 3
# Mostly Static Pages

Alan Hecht

# Agenda

- Start the ongoing project used throughout the book
- Introduction to RSpec
- Introduction to TDD
- Dynamic Rails pages

# Create Application

- rails new sample_app -T
- -T option says not to use the Test::Unit framework for testing
- Add Rails, Sqlite, and RSpec gems to the Gemfile

# Initialize RSpec

- rails generate rspec:install

- Creates:
  - .rspec options file
  - ‘spec’ directory
  - spec_helper.rb for RSpec configuration

# Commit & Deploy Application

- Create improved README file
- Create GitHub repository
- Push commited changes to GitHub
- Deploy to Heroku

# Static Pages

- HTML files served from the public directory
- public/index.html is the home page by default

# Static Pages with Rails

- Create a Git branch to work on the feature

- Generate a Pages controller for home & contact actions
  - rails generate controller Pages home contact

# Static Pages with Rails

- Routes generated for the home & contact actions

```
SampleApp::Application.routes.draw do
  get "pages/home"
  get "pages/contact"
```

- Methods generated for the Pages controller actions

- Views containing static HTML generated for the corresponding controller actions

# Testing Tools

- Autotest & Spork are optional
  - Healthy percentage of people don't use Autotest

- If using Autotest, be sure to install something like autotest-fs
  - Monitors file system changes
  - Polling will fry your laptop and kill the battery

# Test Driven Development

- Is about design as well as testing

- Design the code from the perspective of someone who uses it “outside in”

# Test Driven Development

- Write a failing test
  - Helps ensure that the test is useful
- Write code to make the test pass
  - Only write enough code to make the test pass
- Refactor
  - Tests ensure the code still works
- Repeat

# When not (gasp) to use TDD

- When not sure how to solve a problem
  - In XP, this is a “spike”
- When using it means that nothing will get done
  - Easy to get overwhelmed, especially when learning Ruby & Rails
  - At least write tests after the fact

# RSpec Tests

```
describe PagesController do
  describe "GET 'home'" do
    it "should be successful" do
      get 'home'
      response.should be_success
    end
  end
end
```

- Good advice not to test views & helpers
  - Integration tests for views
- Run with ‘bundle exec rspec spec/’

# Add ‘about’ Page

- Write a failing controller test
  - Add ‘render_views’ class method call to ensure page is there
- Add an ‘about’ controller method
  - Need to add a route & view as well

# Slightly Dynamic Pages - Red

- Add a title test
  - ‘have_selector’ checks for an HTML element in the response
  - Do not have to search for complete element text

```
it "should have the right title" do
  get 'home'
  response.should have_selector("title",
                                :content => "Ruby On Rails Tutorial Sample App | About")
end
```

# Slightly Dynamic Pages - Green

- Add HTML to about.html.erb, contact.html.erb, and home.html.erb to make the tests pass

# Slightly Dynamic Pages Refactor

- HTML pages contain a lot of duplication
  - Don’t Repeat Yourself (DRY)
- Add instance variables for the page title
- Add a layout to define title and page structure
  - HTML from the view inserted at “yield” statement

# Conclusion

- All the tests still pass
- Merge the code back into the ‘master’ branch
  - Delete the branch
- Push to GitHub