Integration testing

- Integration testing tests the flow through the application.
- It is like testing a story that a customer gives at the start of the application.
  - user goes to the store index page. they select a product, adding to their cart. Then they check out….
- testing such a story will involve multiple controllers, views, and models

Functional vs. Integration Testing

- Details are fairly similar
- Think of Integration test as tying together multiple functional tests into one test that matches a user case
- convenience methods
  - follow_redirect
  - post_via_redirect/get_via_redirect
    - Keep following redirects to get to the appropriate handling method
  - https(true)
  - https?

A complete integration test for depot

- When we create models and controllers, the unit and functional tests are created automatically.

- Integration tests are not. You have to generate them.

```
depot> rails generate integration_test user_stories
      invoke  test_unit
      create    test/integration/user_stories_test.rb
```

Integration testing in the depot

- Let look at user_stories_test in the depot application.
  - depot_r

- We can run it by

  rake test:integration user_stories

  - the '_test' part of the file name is added automatically

Performance testing

- Server bashing – how much can we do before it dies?
  - Create 100 orders in 3 seconds?
  - against 1000 products?

  - First to create a thousand products – we will need to use a dynamic fixture
  - test\fixtures\performance\products.yml (note where it is located)

```
<% 1.upto(1000) do |i| %>
  product_<%= i %>:
  id: <%= i %>
  title: Product Number <%= i %>
  description: My description
  image_url: product.gif
  price: 1234
<% end %>
```

Performance test script

- Let look at test/performance/order_speed_test.rb
- and run it by ruby –Itest test/performance/order_speed_test.rb
  - -I is the load path (similar to classpath in java)

Other performance checks

- rails benchmarker 'method'
  - try this with ‘User.encrypted_password(“ksdjfkj”,”sdfsdfs”)’
- rails profiler 'Method‘
- Real load testing is hard
- This might be useful for some purpose to get a good start
- alternative – load stressing tools
  - scripts to generate requests and submit them
  - from another machine...
- Mocks/stubs for completing applications
  - like the credit card transaction – can’t do that each time we test

Rake commands summary

Tasks	Description
`rake test`	Runs all unit, functional and integration tests. You can also simply run `rake` as the *test* target is the default.
`rake test:benchmark`	Benchmark the performance tests
`rake test:functionals`	Runs all the functional tests from `test/functional`
`rake test:integration`	Runs all the integration tests from `test/integration`
`rake test:plugins`	Run all the plugin tests from `vendor/plugins/*/**/test` (or specify with `PLUGIN=_name_`)
`rake test:profile`	Profile the performance tests
`rake test:recent`	Tests recent changes
`rake test:uncommitted`	Runs all the tests which are uncommitted. Supports Subversion and Git
`rake test:units`	Runs all the unit tests from `test/unit`