

Vital Rails

Ruby on Rails Training

1

EdgeCase

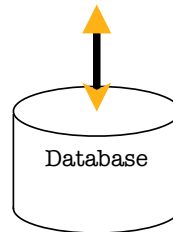
software artisans

2

Model Testing

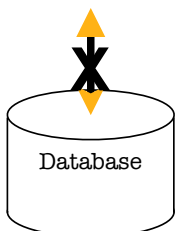
3

M C V



4

M C V



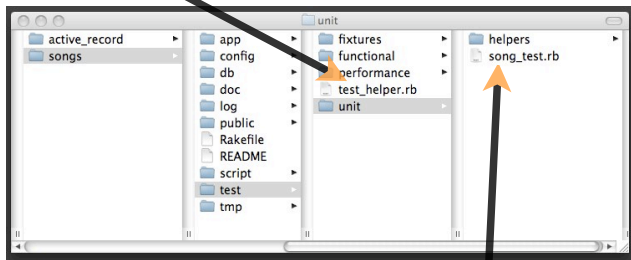
5

Song

- Tracks the name of the Song
- Tracks the duration of the Song

6

Test Helper Code



Model Test Files

7

```
require 'test_helper'

class SongTest < ActiveSupport::TestCase
  # Replace this with your real tests.
  test "the truth" do
    assert true
  end
end
```

8

```
require 'test_helper'

class SongTest < ActiveSupport::TestCase
  # Replace this with your real tests.
  test "the truth" do
    assert true
  end
end
```

Rails Specific Test Case!

9

```
require 'test_helper'

class SongTest < ActiveSupport::TestCase
  # Replace this with your real tests.
  test "the truth" do
    assert true
  end
end
```

Rails testing helpers

10

```
require 'test_helper'

class SongTest < ActiveSupport::TestCase
  # Replace this with your real tests.
  test "the truth" do
    assert true
  end
end
```

Default Rails Test Method ...
Get rid of this ASAP!

11

```
class SongTest < ActiveSupport::TestCase

  test "Saving Name" do
    song = Song.create(:name => "Unity",
                      :duration => 240)

    assert_equal "Unity", song.name
    assert_equal 240, song.duration
  end
end
```

12

Anything bother you about this test?

```
test "Saving Name" do
  song = Song.create(:name => "Unity",
                    :duration => 240)

  assert_equal "Unity", song.name
  assert_equal 240, song.duration
end
```

13

```
test "Saving Name" do
  song = Song.create(:name => "Unity",
                  :duration => 240)

  assert_equal "Unity", song.name
  assert_equal 240, song.duration
end
```

Touches the database

14

Anemic Tests

```
test "Saving Name" do
  song = Song.create(:name => "Unity",
                    :duration => 240)

  assert_equal "Unity", song.name
  assert_equal 240, song.duration
end
```

15

Test the Behavior

16

```
test "Name can't be blank" do
  song = Song.new(:name => nil,
                 :duration => 240)

  assert ! song.valid?
  assert song.errors.on(:name)
  assert_match(/needs to be included/,
              song.errors.on(:name).to_s)
end
```

17

Avoid the Database!

```
test "Name can't be blank" do
  song = Song.new(:name => nil,
                 :duration => 240)

  assert ! song.valid?
  assert song.errors.on(:name)
  assert_match(/needs to be included/,
              song.errors.on(:name).to_s)
end
```

18

3 Checks!

(1) Make sure it is invalid

```
song = Song.new(:name => nil,  
                :duration => 240)  
  
assert ! song.valid?  
assert song.errors.on(:name)  
assert_match(/needs to be included/,  
             song.errors.on(:name).to_s)  
  
end
```

```
test "Name can't be blank" do  
  song = Song.new(:name => nil,  
                  :duration => 240)
```

```
  assert ! song.valid?  
  assert song.errors.on(:name)  
  assert_match(/needs to be included/,  
               song.errors.on(:name).to_s)  
  
end
```

Introduce error on
one of parameters

19

20

3 Checks!

(2) Make sure the right field has the error

```
do  
  nil,  
  :duration => 240)  
  
assert ! song.valid?  
assert song.errors.on(:name)  
assert_match(/needs to be included/,  
             song.errors.on(:name).to_s)  
  
end
```

21

3 Checks!

(3) Make sure the right error has occurred

```
do  
  nil,  
  :duration => 240)  
  
assert ! song.valid?  
assert song.errors.on(:name)  
assert_match(/needs to be included/,  
             song.errors.on(:name).to_s)  
  
end
```

22

Rinse, Repeat

(duration presence)

```
test "Duration can't be blank" do  
  song = Song.new(:name => "Unity",  
                  :duration => nil)  
  
  assert ! song.valid?  
  assert song.errors.on(:duration)  
  assert_match(/n[o']t.*blank/,  
               song.errors.on(:duration).to_s)  
  
end
```

23

Rinse, Repeat

(duration is a number)

```
test "Duration must be a number" do  
  song = Song.new(:name => "Unity",  
                  :duration => "X")  
  
  assert ! song.valid?  
  assert song.errors.on(:duration)  
  assert_match(/(n't|not).*number/,  
               song.errors.on(:duration).to_s)  
  
end
```

24

Time to Refactor

25

Extract Method

```
test "Name can't be blank" do
  song = Song.new(:name => nil,
                  :duration => 240)
```

```
    assert ! song.valid?
    assert song.errors.on(:name)
    assert_match(/needs to be included/,
                  song.errors.on(:name).to_s)
  end
```

Really useful stuff

26

Extract Method

```
private

def assert_errors_on(obj, field, pattern)
  assert ! obj.valid?
  assert obj.errors.on(field)
  assert_match(pattern,
    obj.errors.on(field).to_s) if pattern
end
```

27

New Version of Test

```
test "Name can't be blank" do
  song = Song.new(:name => nil,
                  :duration => 240)
  assert_errors_on(song, :name,
    /needs to be included/)
end
```

28

New Version of Test

```
test "Name can't be blank" do
  song = Song.new(:name => nil,
                  :duration => 240)
  assert_errors_on(song, :name,
    /needs to be included/)
end
```

Variations of this is
used over and over

29

Extract Valid Parameters

```
VALID_OPTIONS = {
  :name => "Unity",
  :duration => 240
}
```

```
test "Can Create Valid Song" do
  song = Song.new(VALID_OPTIONS)
  assert song.valid?
end
```

30

New Version of Test

```
test "Name can't be blank" do
  song = Song.new(
    VALID_OPTIONS.merge(:name => nil))
  assert_errors_on(song, :name,
                    /needs to be included/)
end
```

31

New Version of Test

```
test "Name can't be blank" do
  song = Song.new(
    VALID_OPTIONS.merge(:name => nil))
  assert_errors_on(song, :name,
                    /needs to be included/)
end
```

Emphasizes the
parameter in error

32

Can We Go Farther?

33

Move Valid Options

```
VALID_OPTIONS = {
  :name => "Unity", :duration => 240
}
```



```
class Song
  def self.valid_options
    { :name => "Unity", :duration => 240 }
  end
end
```

34

New Assertion Method

```
def assert_validates_presence_of(
  klass, field,
  pattern=/n[o']t.*blank/)

  bad_opts = klass.valid_options.
    merge(field => nil)
  instance = klass.new(bad_opts)

  assert_errors_on(instance, field,
                    pattern)
end
```

35

New Version of Test

```
test "Name can't be blank" do
  song = Song.new(
    VALID_OPTIONS.merge(:name => nil))
  assert_errors_on(song, :name,
                    /needs to be included/)
end
```



```
test "Name can't be blank" do
  assert_validates_presence_of(
    Song, :name, /needs to be included/)
end
```

36

Next Level: Class Method

```
class SongTest < ActiveSupport::TestCase
  assert_validates_presence_of :name,
    /needs to be included/
end
```

- Change to class method
- generate a test method
- autodetect Song from class name

37

Next Level: Class Method

```
class SongTest < ActiveSupport::TestCase
  assert_validates_presence_of :name,
    /needs to be included/
end
```

**Left as an exercise
for the reader**

- Change to class method
- generate a test method
- autodetect Song from class name

38

Rinse, Repeat
(for other validation types)

39

(for discussion)

Is it worth your time to
test Rails validations
and relations?

40

Business Logic?

41

```
test "Duration Converts to Min and Sec" do
  song = Song.new(Song.valid_options)
  assert_equal [3, 30],
    song.duration_minutes_seconds
end
```

```
class Song
  def duration_minutes_seconds
    [ duration / 60, duration % 60 ]
  end
end
```

42

Business Logic

- Shouldn't need to hit the database
- Normal TDD/BDD testing that we use for non-rails objects

43

When to Hit the Database

44

```
class Song
  LONG_SONG = 300

  def self.find_long_songs
    find :all,
      :conditions =>
        ['duration > ?', LONG_SONG],
      :order => :duration
  end
end
```

45

```
test "Finding Long Song" do
  # Given
  Song.create!(:name => "Short1",
               :duration => 100)
  Song.create!(:name => "Short2",
               :duration => 300)
  Song.create!(:name => "Long1",
               :duration => 301)
  Song.create!(:name => "Long2",
               :duration => 400)

  # When
  songs = Song.find_long_songs

  # Then
  assert_equal ['Long1', 'Long2'],
    songs.map { |s| s.name }.sort
end
```

46

Remember ...

Avoid the Database

(except for finders)

47

Remember ...

Don't Test the Framework

48

Remember ...

Test Behavior

49

Remember ...

Refactor Your Tests

(as well as your code base)

50

LAB 3-1

Model Testing

51

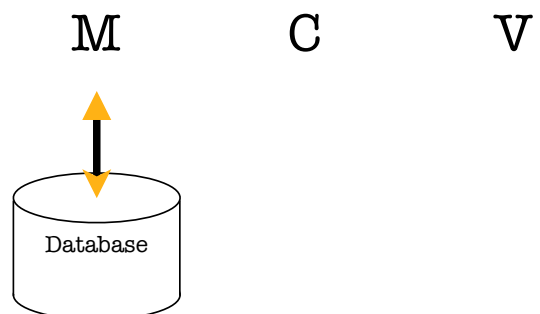
Model Testing

- Add tests for all the validations in your model
- Add test for any "business logic" that you have implemented in your model

52

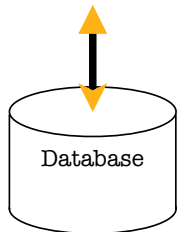
Controller Testing

53



54

M C V

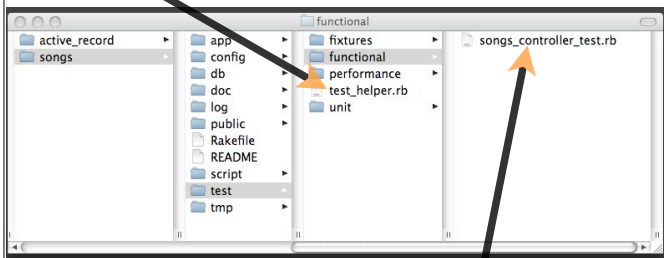


55

Controller / View Testing

56

Test Helper Code



Controller Test Files

57

```
require 'test_helper'

class SongsControllerTest < ActionController::TestCase
  # Replace this with your real tests.
  test "the truth" do
    assert true
  end
end
```

Controller Specific Test Case
(controllers are strange beasts)

58

```
class SongsControllerTest <
  ActionController::TestCase
  test "GET on index populates @songs" do
    get :index

    assert_response :success
    assert_template :index
    assert assigns(:songs)
    assert_equal [], assigns(:songs)
  end
end
```

59

```
class SongsControllerTest <
  ActionController::TestCase
  test "GET on index populates @songs" do
    get :index

    assert_response :success
    assert_template :index
    assert assigns(:songs)
    assert_equal [], assigns(:songs)
  end
end
```

Name the test:
(1) What operation GET
(2) What controller action
(3) What we expect

60

(an aside)

```
class SongsControllerTest <
  ActionController::TestCase
  test "GET on index populates @songs" do
    get :index

    assert_response :success
    assert_template :index
    assert assigns(:songs)
    assert_equal [], assigns(:songs)
  end
end
```

Perform the action

Operations are: get, put,
post, delete, and head

```
get :new, :song => {
  :name => "Unity",
  :duration => "210",
}
```

You can pass parameters to get
(and post/put/delete/head)

```
class SongsControllerTest <
  ActionController::TestCase
  test "GET on index populates @songs" do
    get :index

    assert_response :success
    assert_template :index
    assert assigns(:songs)
    assert_equal [], assigns(:songs)
  end
end
```

Check the response

Possible responses
are: :success, :redirect,
:missing, and :error

```
class SongsControllerTest <
  ActionController::TestCase
  test "GET on index populates @songs" do
    get :index

    assert_response :success
    assert_template :index
    assert assigns(:songs)
    assert_equal [], assigns(:songs)
  end
end
```

Check the selected template name

**Check that the proper instance
variables have been assigned**

```
class SongsControllerTest <
  ActionController::TestCase
  test "GET on index populates @songs" do
    get :index

    assert_response :success
    assert_template :index
    assert assigns(:songs)
    assert_equal [], assigns(:songs)
  end
end
```


**Then make sure they have
been assigned the right value.**

```
class SongsControllerTest <
  ActionController::TestCase
  test "GET on index populates @songs" do
    get :index

    assert_response :success
    assert_template :index
    assert assigns(:songs)
    assert_equal [], assigns(:songs)
  end
end
```

```
class SongsControllerTest <
  ActionController::TestCase
  test "GET on index populates @songs" do
    get :index

    assert_response :success
    assert_template :index
    assert_assigns(:songs)
    assert_equal 11, assigns(:songs)
  end
end
```



But is this the right value?

67

Test Data

- Where does the Test Data come from?
 - Fixtures
 - Yuck - Slow, Inflexible, Brittle!
 - Mocks
 - Fast & Flexible, but brittle if not used carefully.
 - Other

68

Confession Time

69

Given

- Test Data for controllers is hard
- And controller logic should be simple
- ...

70

I tend to do little
controller unit testing

71

But what about view
testing?

72

You can test views ...

```
assert_tag :div,  
  :attributes => { :class => "warning" }
```

But there's little logic in views to check

73

LAB 3-2

Controller Testing

74

Controller Testing

- Add a controller test for the actions in the presentation controller

75

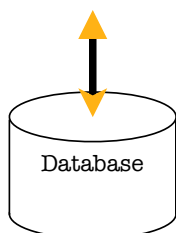
Integration Testing

76

M

C

V



77

Focus

- It all works together
- The whole system is correct
- It keeps working together

78

Options

(front-end)

- Test::Unit or RSpec
- Cucumber

79

Options

(front-end)

- Test::Unit or RSpec
- **Cucumber**

80



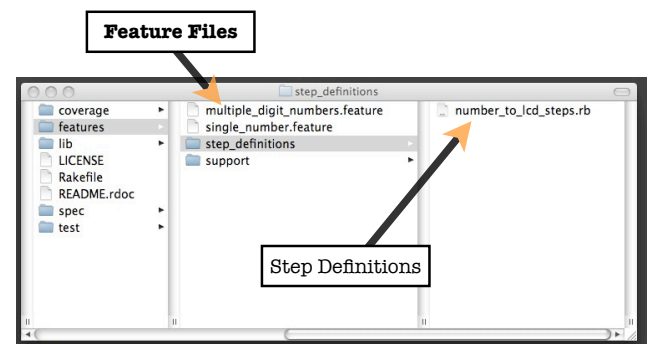
81

External DSL for Integration Testing

82

```
[sudo] gem install cucumber
```

83



84

```
Feature: Test::Unit
  In order to please people who like Test::Unit
  As a Cucumber user
  I want to be able to use assert* in my
  step definitions.

Scenario: assert_equal
  Given x = 5
  And y = 5
  Then I can assert that x == y
```

85

```
Feature: Test::Unit
  In order to please people who like Test::Unit
  As a Cucumber user
  I want to be able to use assert* in my
  step definitions.
```

```
Scenario: assert_equal
  Given x = 5
  And y = 5
```

User Story

- (1) Start of the file
- (2) Can contain any text
- (3) Ends with "Scenario" on a separate line.

86

Scenario

- (1) Begins with the word Scenario
- (2) Remaining words are used used for the description
- (3) Followed by a list of steps

```
Feature: Test::Unit
  In order to please people who like Test::Unit
  As a Cucumber user
  I want to be able to use assert* in my
  step definitions.
```

```
Scenario: assert_equal
  Given x = 5
  And y = 5
  Then I can assert that x == y
```

87

Steps

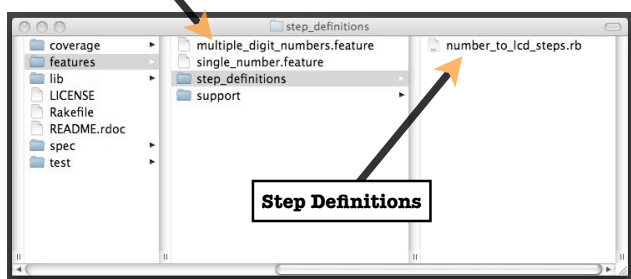
Words: Given, When, Then, But, And

```
Feature: Test::Unit
  In order to please people who like Test::Unit
  As a Cucumber user
  I want to be able to use assert* in my
  step definitions.
```

```
Scenario: assert_equal
  Given x = 5
  And y = 5
  Then I can assert that x == y
```

88

Feature Files



Step Definitions

```
Given /^(w+) = (w+)$/ do |var, value|
  instance_variable_set("@#{var}", value)
end
```

89

90

Pattern

Matches line in feature file

```
Given /(\w+) = (\w+)\$/ do |var, value|
  instance_variable_set("@#{var}", value)
end
```

91

Matching Groups

Yields to block via parameters

```
Given /(\w+) = (\w+)\$/ do |var value|
  instance_variable_set("@#{var}", value)
end
```

92

Implementation

Anything you want it to be ...

```
Given /(\w+) = (\w+)\$/ do |var, value|
  instance_variable_set("@#{var}", value)
end
```

93

Hooks

One Time Initialization

```
World do
  @object = Object.new
end
```

94

Hooks

Per-Scenario Initialization

```
Before do
  @calc = Calculator.new
end
```

Hooks

Per-Scenario Initialization

```
After do |scenario|
  if scenario.status.index(:failed)
    # Report Failure
  end
end
```

95

Scenario Outline: Add two numbers

```
Given I have entered <input_1> into the calculator
And I have entered <input_2> into the calculator
When I press <button>
Then the result should be <output> on the screen
```

Examples:

input_1	input_2	button	output
20	30	add	50
2	5	add	7
0	40	add	40

96

Scenario Outline: Add two numbers
 Given I have entered <input_1> into the calculator
 And I have entered <input_2> into the calculator
 When I press <button>
 Then the result should be <output> on the screen

Examples:

input_1	input_2	button	output
20	30	add	50
2	5	add	7
0	40	add	40

97

Scenario Outline: Add two numbers
 Given I have entered <input_1> into the calculator
 And I have entered <input_2> into the calculator
 When I press <button>
 Then the result should be <output> on the screen

Examples:

input 1	input 2	button	output
20	30	add	50
2	5	add	7
0	40	add	40

Repeated
 for each line in the table

98

Cucumber

Behaviour Driven Development
 with elegance and joy

on Rails

99

Options

(back-end)

- Rails Integration Tests
- Selenium Browser Testing
- Watir / FireWatir / SafariWatir
- Webrat

100

Options

(back-end)

- Selenium Browser Testing
- Watir / FireWatir / SafariWatir
- **Webrat**

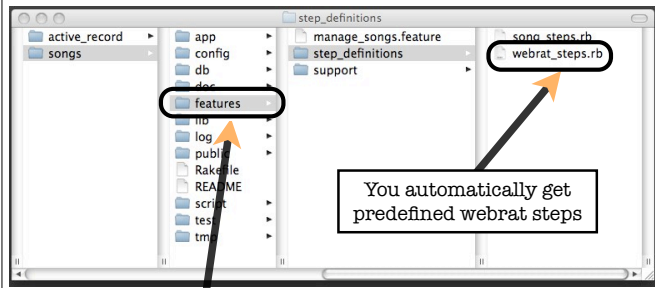
101

script/generate cucumber --testunit

-- or --

script/generate cucumber --rspec

102



Added Feature Directory
(and other minor stuff)

You automatically get
predefined webrat steps

103

`script/generate feature model_name`

104

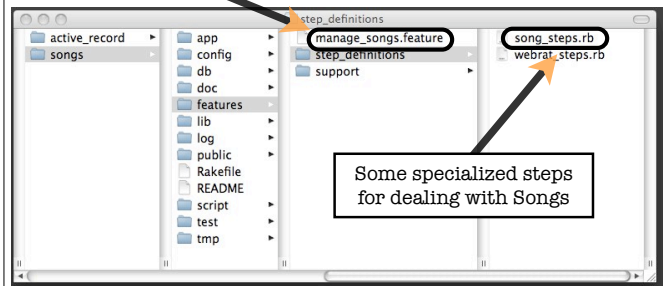
`script/generate feature model_name`

“Don’t get addicted to this
generator – you’re better off writing
these by hand in the long run.”

-- github documentation

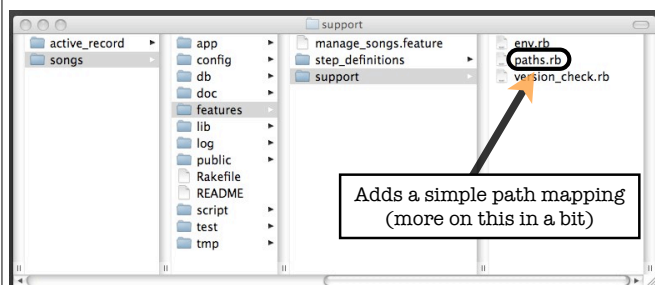
105

Manage Songs story
(with some predefined scenarios)



Some specialized steps
for dealing with Songs

106



Adds a simple path mapping
(more on this in a bit)

107

What's A Feature / Story

108

```

Feature: Manage songs
  In order to [goal]
    [stakeholder]
    wants [behaviour]

Scenario: Register new song
  Given I am on the new song page
  And I press "Create"

Scenario: Delete song
  Given the following songs:
  ||
  ||
  ||
  ||
  ||
  When I delete the 3rd song
  Then I should see the following songs:
  ||
  ||
  ||
  ||
  ||

```

109

Feature Description

```

Feature: Manage songs
  In order to manage
  As a user, I
  want to see my song list

```

For comments only

110

Scenario

```

Scenario: Register new song
  Given I am on the new song page
  When I fill in "Unity" for "Song Name"
  When I fill in "123" for "Song Duration"
  And I press "Save Song"
  Then I should see "Unity"
  And I should see "123"

```

111

Scenario

```

Scenario: Register new song
  Given I am on the new song page
  When I fill in "Unity" for "Song Name"
  When I fill in "123" for "Song Duration"
  And I press "Save Song"
  Then I should see "Unity"
  And I should see "123"

```

112

Scenario

```

Scenario: Register new song
  Given I am on the new song page
  When I fill in "Unity" for "Song Name"
  When I fill in "123" for "Song Duration"
  And I press "Save Song"
  Then I should see "Unity"
  And I should see "123"

```

Name	Duration	show	delete
Allentown	263	show	delete
Upstream Girl	189	show	delete
Jan Isaccant Mass	369	show	delete
Angry Young Man	323	show	delete
A Matter of Trust	309	show	delete
Only the Good Die Young	201	show	delete
Baby Grand	369	show	delete
Big Shot	280	show	delete
Honesty	139	show	delete
Crowdnight	139	show	delete
Unity	123	show	delete

113

- NO:
 - Input field names or IDs
 - No form IDs
 - No explicit, hard coded URLs
 - No HTTP method specification

114

Easy to Change

115

How is this mapped?

```
Scenario: Register new song
  Given I am on the new song page
  When I fill in "Unity" for "Song Name"
  When I fill in "123" for "Song Duration"
  And I press "Save Song"
  Then I should see "Unity"
  And I should see "123"
```

```
Given /^I am on (.+)/ do |page_name|
  visit path_to(page_name)
end
```

116

How is this mapped?

```
Scenario: Register new song
  Given I am on the new song page
  When I fill in "Unity" for "Song Name"
  When I fill in "123" for "Song Duration"
  And I press "Save Song"
  Then I should see "Unity"
  And I should see "123"
```

```
Given /^I am on (.+)/ do |page_name|
  visit path_to(page_name)
end
```

117

How is this mapped?

```
Scenario: Register new song
  Given I am on the new song page
  When I fill in "Unity" for "Song Name"
  When I fill in "123" for "Song Duration"
  And I press "Save Song"
  Then I should see "Unity"
  And I should see "123"
```

Matches

```
Given /^I am on (.+)/ do |page_name|
  visit path_to(page_name)
end
```

118

How is this mapped?

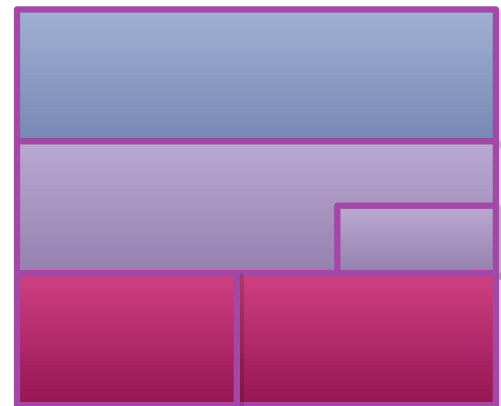
```
Scenario: Register new song
  Given I am on the new song page
  When I fill in "Unity" for "Song Name"
  When I fill in "123" for "Song Duration"
  And I press "Save Song"
  Then I should see "Unity"
  And I should see "123"
```

Webrat
One of many
useful operations

support/paths.rb
Maps "nice" page
names to URLs

```
Given /^I am on (.+)/ do |page_name|
  visit path_to(page_name)
end
```

119



120

Copyright 2007-2009, EdgeCase, LLC. All rights reserved.

120

Scenario: Register new song
Given I am on the new song page
When I fill in "Unity" for "Song Name"

121 Copyright 2007-2009, EdgeCase, LLC. All rights reserved.

```
Given /^the following songs:$/ do |songs|
  Song.create!(songs.hashes)
end
```

```
def path_to(page_name)
  ...
end
```

122 Copyright 2007-2009, EdgeCase, LLC. All rights reserved.

```
visit(url)
click_button(id)
```

```
Song.find(id)
```

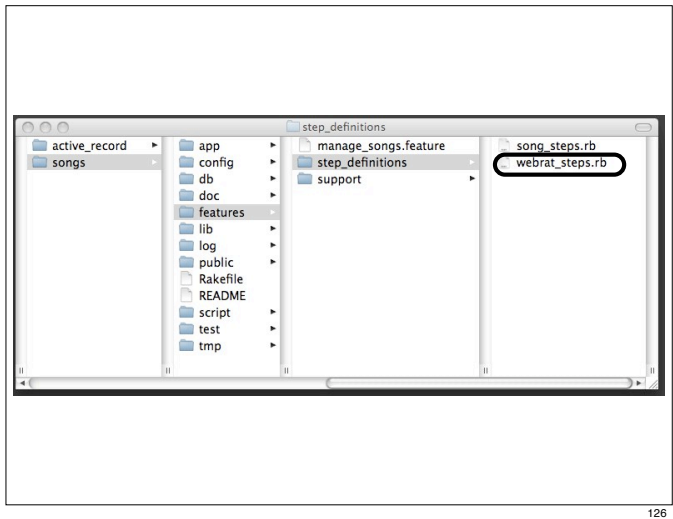
123 Copyright 2007-2009, EdgeCase, LLC. All rights reserved.

Translation Layer

124 Copyright 2007-2009, EdgeCase, LLC. All rights reserved.

What's Available

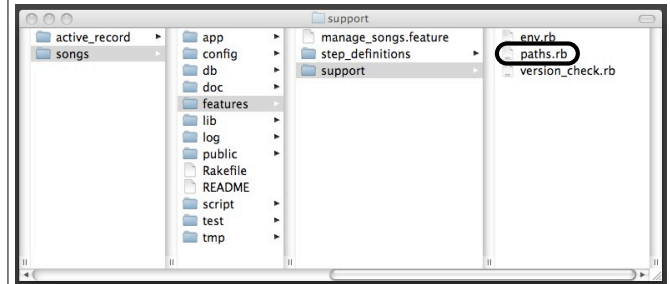
125



From webrat_steps.rb

```
Given /^I am on (.+)$/
When /^I go to (.+)$/
When /^I press "([^"]*)"$/ do |link|
  When /^I follow "([^"]*)"$/ within "([^"]*)"$/
  When /^I fill in "([^"]*)" with "([^"]*)"$/
  When /^I fill in "([^"]*)" for "([^"]*)"$/
  When /^I fill in the following:$/
  When /^I select "([^"]*)" from "([^"]*)"$/
  When /^I select "([^"]*)" as the date and time$/
  When /^I select "([^"]*)" as the "([^"]*)" date and time$/
  When /^I select "([^"]*)" as the time$/
  When /^I select "([^"]*)" as the "([^"]*)" time$/
  When /^I select "([^"]*)" as the date$/
  When /^I select "([^"]*)" as the "([^"]*)" date$/
  When /^I check "([^"]*)"$/
```

127



128

From pages.rb

```
def path_to(page_name)
  case page_name
  when /the home\s?page/
    '/'
  when /the new song page/
    new_song_path
  when /the songs page/
    songs_path
  when /the "(.*)" song page/
    song_path(Song.find_by_name($1))
  else
    raise "Can't find mapping from \"#{page_name}\" +
      "to a path.\n" +
      "Now, go and add a mapping in #{__FILE__}"
  end
end
```

Standard Stuff

129

From pages.rb

```
def path_to(page_name)
  case page_name
  when /the home\s?page/
    '/'
  when /the new song page/
    new_song_path
  when /the songs page/
    songs_path
  when /the "(.*)" song page/
    song_path(Song.find_by_name($1))
  else
    raise "Can't find mapping from \"#{page_name}\" +
      "to a path.\n" +
      "Now, go and add a mapping in #{__FILE__}"
  end
end
```

Added by generate feature

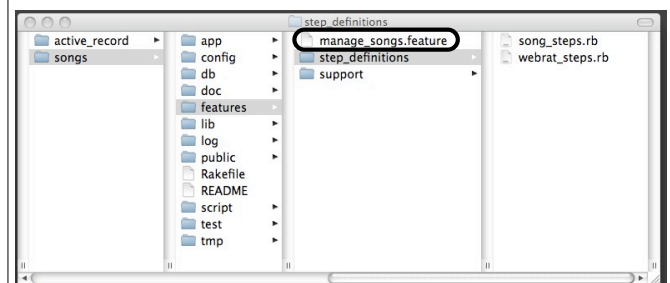
130

From pages.rb

```
def path_to(page_name)
  case page_name
  when /the home\s?page/
    '/'
  when /the new song page/
    new_song_path
  when /the songs page/
    songs_path
  when /the "(.*)" song page/
    song_path(Song.find_by_name($1))
  else
    raise "Can't find mapping from \"#{page_name}\" +
      "to a path.\n" +
      "Now, go and add a mapping in #{__FILE__}"
  end
end
```

Added by hand

131



132

From manage_songs.rb

```
Scenario: Delete song
  Given the following songs:
    |name|duration|
    |Unity|10|
    |The Parts of Ruby|11|
    |White Sandy Beach of Hawaii|12|
    |Over the Rainbow|13|
  When I delete the 3rd song
  Then I should see the following songs:
    |Name|Duration|
    |Unity|10|
    |The Parts of Ruby|11|
    |Over the Rainbow|13|
```

133

```
Scenario: Delete song
  Given the following songs:
    |name|duration|
    |Unity|10|
    |The Parts of Ruby|11|
    |White Sandy Beach of Hawaii|12|
    |Over the Rainbow|13|
  When I delete the 3rd song
  Then I should see the following songs:
    |Name|Duration|
    |Unity|10|
    |The Parts of Ruby|11|
    |Over the Rainbow|13|
```

Given
Initial conditions
(4 songs)

134

From manage_songs.rb

```
Scenario: Delete song
  Given the following songs:
    |name|duration|
    |Unity|10|
    |The Parts of Ruby|11|
    |White Sandy Beach of Hawaii|12|
    |Over the Rainbow|13|
  When I delete the 3rd song
  Then I should see the following songs:
    |Name|Duration|
    |Unity|10|
    |The Parts of Ruby|11|
    |Over the Rainbow|13|
```

When
Code to Test
(delete a song)

135

From manage_songs.rb

```
Scenario: Delete song
  Given the following songs:
    |name|duration|
    |Unity|10|
    |The Parts of Ruby|11|
    |White Sandy Beach of Hawaii|12|
    |Over the Rainbow|13|
  When I delete the 3rd song
  Then I should see the following songs:
    |Name|Duration|
    |Unity|10|
    |The Parts of Ruby|11|
    |Over the Rainbow|13|
```

Then
Post-conditions
(only 3 songs left)

136

Running Cucumber

```
$ rake -T cucumber
(in /Users/jim/Documents/Presentations/VitalRails/src/songs)
rake cucumber # Alias for cucumber:ok
rake cucumber:all # Run all features
rake cucumber:ok # Run features that should pass
rake cucumber:wip # Run features that are being worked on
```

137

138

Terminal window showing Cucumber command and output. A callout box labeled "Invoke" points to the command.

```
[dev]$ rake cucumber:ok
(in /Users/jim/Documents/Presentations/VitalRails/src/songs)
/System/Library/Frameworks/Ruby.framework/Versions/1.8/usr/bin/ruby -I "/Library/Ruby/Gems/1.8/gems/cucumber-0.4.2/lib:" "/Library/Ruby/Gems/1.8/gems/cucumber-0.4.2/bin/cucumber" --color --tags @wip --strict --format pretty
Feature: Manage songs
  In order to manage
    As a user, I
      want to see my song list

Scenario: Register new song # features/manage_songs.feature:6
  Given I am on the new song page # features/step_definitions/webrat_steps.rb:11
  When I fill in "Unity" for "Song Name" # features/step_definitions/webrat_steps.rb:35
  When I fill in "123" for "Song Duration" # features/step_definitions/webrat_steps.rb:35
  And I press "Save Song" # features/step_definitions/webrat_steps.rb:19
  Then I should see "Unity" # features/step_definitions/webrat_steps.rb:123
  And I should see "123" # features/step_definitions/webrat_steps.rb:123

Scenario: Delete song # features/manage_songs.feature:14
  Given the following songs: # features/step_definitions/song_steps.rb:1
    | name | duration |
    | Unity | 10 |
    | The Parts of Ruby | 11 |
    | White Sandy Beach of Hawaii | 12 |
    | Over the Rainbow | 13 |
```

139

Terminal window showing Cucumber command and output. A callout box labeled "Steps" points to the command, and another callout box labeled "Green is passing" points to the output.

```
[dev]$ rake cucumber:ok
(in /Users/jim/Documents/Presentations/VitalRails/src/songs)
/System/Library/Frameworks/Ruby.framework/Versions/1.8/usr/bin/ruby -I "/Library/Ruby/Gems/1.8/gems/cucumber-0.4.2/lib:" "/Library/Ruby/Gems/1.8/gems/cucumber-0.4.2/bin/cucumber" --color --tags @wip --strict --format pretty
Feature: Manage songs
  In order to manage
    As a user, I
      want to see my song list

Scenario: Register new song # features/manage_songs.feature:6
  Given I am on the new song page # features/step_definitions/webrat_steps.rb:11
  When I fill in "Unity" for "Song Name" # features/step_definitions/webrat_steps.rb:35
  When I fill in "123" for "Song Duration" # features/step_definitions/webrat_steps.rb:35
  And I press "Save Song" # features/step_definitions/webrat_steps.rb:19
  Then I should see "Unity" # features/step_definitions/webrat_steps.rb:123
  And I should see "123" # features/step_definitions/webrat_steps.rb:123

Scenario: Delete song # features/manage_songs.feature:14
  Given the following songs: # features/step_definitions/song_steps.rb:1
    | name | duration |
    | Unity | 10 |
    | The Parts of Ruby | 11 |
    | White Sandy Beach of Hawaii | 12 |
    | Over the Rainbow | 13 |
```

140

Terminal window showing Cucumber command and output. A callout box labeled "Where Found" points to the command, and another callout box labeled "File and line number where step is defined" points to the output.

```
[dev]$ rake cucumber:ok
(in /Users/jim/Documents/Presentations/VitalRails/src/songs)
/System/Library/Frameworks/Ruby.framework/Versions/1.8/usr/bin/ruby -I "/Library/Ruby/Gems/1.8/gems/cucumber-0.4.2/lib:" "/Library/Ruby/Gems/1.8/gems/cucumber-0.4.2/bin/cucumber" --color --tags @wip --strict --format pretty
Feature: Manage songs
  In order to manage
    As a user, I
      want to see my song list

Scenario: Register new song # features/manage_songs.feature:6
  Given I am on the new song page # features/step_definitions/webrat_steps.rb:11
  When I fill in "Unity" for "Song Name" # features/step_definitions/webrat_steps.rb:35
  When I fill in "123" for "Song Duration" # features/step_definitions/webrat_steps.rb:35
  And I press "Save Song" # features/step_definitions/webrat_steps.rb:19
  Then I should see "Unity" # features/step_definitions/webrat_steps.rb:123
  And I should see "123" # features/step_definitions/webrat_steps.rb:123

Scenario: Delete song # features/manage_songs.feature:14
  Given the following songs: # features/step_definitions/song_steps.rb:1
    | name | duration |
    | Unity | 10 |
    | The Parts of Ruby | 11 |
    | White Sandy Beach of Hawaii | 12 |
    | Over the Rainbow | 13 |
```

141

Terminal window showing Cucumber command and output. A callout box labeled "Stats" points to the command, and another callout box labeled "Stats" points to the output.

```
[dev]$ rake cucumber:ok
(in /Users/jim/Documents/Presentations/VitalRails/src/songs)
/System/Library/Frameworks/Ruby.framework/Versions/1.8/usr/bin/ruby -I "/Library/Ruby/Gems/1.8/gems/cucumber-0.4.2/lib:" "/Library/Ruby/Gems/1.8/gems/cucumber-0.4.2/bin/cucumber" --color --tags @wip --strict --format pretty
Feature: Manage songs
  In order to manage
    As a user, I
      want to see my song list

Scenario: Register new song # features/manage_songs.feature:6
  Given I am on the new song page # features/step_definitions/webrat_steps.rb:11
  When I fill in "Unity" for "Song Name" # features/step_definitions/webrat_steps.rb:35
  When I fill in "123" for "Song Duration" # features/step_definitions/webrat_steps.rb:35
  And I press "Save Song" # features/step_definitions/webrat_steps.rb:19
  Then I should see "Unity" # features/step_definitions/webrat_steps.rb:123
  And I should see "123" # features/step_definitions/webrat_steps.rb:123

Scenario: Delete song # features/manage_songs.feature:14
  Given the following songs: # features/step_definitions/song_steps.rb:1
    | name | duration |
    | Unity | 10 |
    | The Parts of Ruby | 11 |
    | White Sandy Beach of Hawaii | 12 |
    | Over the Rainbow | 13 |

4 scenarios (4 passed)
18 steps (18 passed)
0m0.401s
Fails: 0
```

142

Cucumber with an error

Terminal window showing Cucumber command and output. A callout box labeled "Where Found" points to the command, and another callout box labeled "File and line number where step is defined" points to the output.

```
[dev]$ rake cucumber:ok
(in /Users/jim/Documents/Presentations/VitalRails/src/songs)
/System/Library/Frameworks/Ruby.framework/Versions/1.8/usr/bin/ruby -I "/Library/Ruby/Gems/1.8/gems/cucumber-0.4.2/lib:" "/Library/Ruby/Gems/1.8/gems/cucumber-0.4.2/bin/cucumber" --color --tags @wip --strict --format pretty
Feature: Manage songs
  In order to manage
    As a user, I
      want to see my song list

Scenario: Register new song # features/manage_songs.feature:6
  Given I am on the new song page # features/step_definitions/webrat_steps.rb:11
  When I fill in "Unity" for "Song Name" # features/step_definitions/webrat_steps.rb:35
  When I fill in "123" for "Song Duration" # features/step_definitions/webrat_steps.rb:35
  And I press "Save Song" # features/step_definitions/webrat_steps.rb:19
  Then I should see "Unity" # features/step_definitions/webrat_steps.rb:123
  And I should see "123" # features/step_definitions/webrat_steps.rb:123

Scenario: Delete song # features/manage_songs.feature:14
  Given the following songs: # features/step_definitions/song_steps.rb:1
    | name | duration |
    | Unity | 10 |
    | The Parts of Ruby | 11 |
    | White Sandy Beach of Hawaii | 12 |
    | Over the Rainbow | 13 |
```

143

Cucumber with an error

Terminal window showing Cucumber command and output. A callout box labeled "Failing Step" points to the output.

```
[dev]$ rake cucumber:ok
(in /Users/jim/Documents/Presentations/VitalRails/src/songs)
/System/Library/Frameworks/Ruby.framework/Versions/1.8/usr/bin/ruby -I "/Library/Ruby/Gems/1.8/gems/cucumber-0.4.2/lib:" "/Library/Ruby/Gems/1.8/gems/cucumber-0.4.2/bin/cucumber" --color --tags @wip --strict --format pretty
Feature: Manage songs
  In order to manage
    As a user, I
      want to see my song list

Scenario: Register new song # features/manage_songs.feature:6
  Given I am on the new song page # features/step_definitions/webrat_steps.rb:11
  When I fill in "Unity" for "Song Name" # features/step_definitions/webrat_steps.rb:35
  When I fill in "123" for "Song Duration" # features/step_definitions/webrat_steps.rb:35
  And I press "Save Song" # features/step_definitions/webrat_steps.rb:19
  Then I should see "Unity" # features/step_definitions/webrat_steps.rb:123
  And I should see "123" # features/step_definitions/webrat_steps.rb:123

Scenario: Delete song # features/manage_songs.feature:14
  Given the following songs: # features/step_definitions/song_steps.rb:1
    | name | duration |
    | Unity | 10 |
    | The Parts of Ruby | 11 |
    | White Sandy Beach of Hawaii | 12 |
    | Over the Rainbow | 13 |
```

144

Cucumber with an error

HTML Dump

```
Scenario: Show a single song                                     # features/monage_songs.feature
  In the following songs:                                         # features/step_definitions.rb
    Name | Id | Duration |
    -----|-----|-----|
    Unity | 112 |          |
    The Parts of Ruby | 112 |          |
  And as on the "Unity" song page                                # features/step_definitions.rb
  Then I should see "Hello"                                       # features/step_definitions.rb
  expected the following element's content to include "Unity":

  # Runs: Your music on the web
  Name: Unity
  Duration: 11
  Back
  Navigation
  Songs
  Create a Song
  Artists
  Create an Artist
  Footer

  -false is not true, (Test::Unit::AssertionFailedError)
  /system/Library/Frameworks/Ruby.framework/Versions/1.8/usr/lib/ruby
  /system/Library/Frameworks/Ruby.framework/Versions/1.8/usr/lib/ruby
  /system/Library/Frameworks/Ruby.framework/Versions/1.8/usr/lib/ruby
  /system/Library/Frameworks/Ruby.framework/Versions/1.8/usr/lib/ruby
  /system/Library/Frameworks/Ruby.framework/Versions/1.8/usr/lib/ruby
  /features/step_definitions/wombat_steps.rb:124:in `should see'
  /features/step_definitions/wombat_steps.rb:43:in `Then I should see "Unity"'

  And I should not see "The Parts of Ruby" # features/step_definitions.rb

Failing Scenarios:
cucumber: features/monage_songs.feature:37 # Scenario: Show a single song
```

145

Cucumber with an error

Skipped Steps

[illegible]

146

Cucumber with missing steps

Undefined Step

No match found in any
step definition file.

```
Scenario: Show a single song # Features/manage_songs.feature:37
Given the following songs: # features/step_definitions/song_steps.rb:1
  I name | duration |
    Unity | 11 |
    The Parts of Ruby | 12 |
And I am on the "Unity" song page # features/step_definitions/webroot_steps.rb:11
Then I should see an undefined step # Features/manage_songs.feature:44
  Undefined step: "The Parts of Ruby" # features/step_definitions/webroot_steps.rb:123
Features/manage_songs.feature:44: In "Then I should try an undefined step" # features/step_definitions/webroot_steps.rb:123
And I should see "The Parts of Ruby" # features/step_definitions/webroot_steps.rb:145
```

147

Cucumber with missing steps

Skipped Step

Can't continue because we have an undefined step

```
Scenario: Show a single song                                # features/manage_songs.feature:37
  Given the following songs:                                # features/step_definitions/song_steps.rb:1
    | name | duration |
    | Unity | 11 |
    | The Parts of Ruby | 12 |
  And I am on the "unity" song page                        # features/step_definitions/webchat_steps.rb:11
  And I should see "Unity"                                 # features/step_definitions/webchat_steps.rb:123
  Then I should try an undefined step                      # features/manage_songs.feature:44
  Undefined step: "I should try an undefined step"         # features/step_definitions/undefined_steps.rb:1
  features/manage_songs.feature:44:in "Then I should try an undefined step"
  And I should see "11"                                     # features/step_definitions/webchat_steps.rb:123
  And I should not see "The Parts of Ruby"                 # features/step_definitions/webchat_steps.rb:145
```

148

Cucumber with missing steps

Undefined Step

No match found in any
step definition file.

```
You can implement step definitions for undefined steps with these snippets:
```

```
Then /^I should try an undefined step$/ do
  pending
end
```

Copy this into a step definition file.

149

Live Demo:

Add Scenario for editing a song

150

LAB 3-2

Controller Testing

151

Cucumber Testing

- Write cucumber tests for your Presentation app features.

152

Development Lifecycle

153

Write a new (failing)
Cucumber Scenario

|

Write a unit test for a
portion of the
scenario

|

Implement Code to
pass the unit test

|

Scenario
Done?

154