CS 683 Emerging Technologies
Fall Semester, 2006
Doc 19 Simple Book Store App
Oct 31, 2006

Copyright ©, All rights reserved. 2006 SDSU & Roger Whitney, 5500 Campanile Drive, San Diego, CA 92182-7700 USA. OpenContent (http://www.opencontent.org/opl.shtml) license defines the copyright on this document.

References

Agile Web Development with Rails, Thomas & Hanson, The Pragmatic Bookshelf, 2005

Agile Web Development with Rails 2nd Ed Bl.16 October 25, Thomas & Hanson, The Pragmatic Bookshelf, PDF

MySQL 3.x & Rails Issue, Post on rail.trechno-weenie.net
http://72.14.253.104/search?q=cache:ZjlYt82GCYIJ:rails.techno-weenie.net/question/2005/12/25/migrations_with_mysql_3_23+ruby+on+rails+mysql+InnoDb+migrations&hl=en&gl=us&ct=clnk&cd=3

Book Example

Database Setup

Run MySql (or PostgreSQL) database

Create the databases:

- cs683BookStore_development
- cs683BookStore_test
- cs683BookStore_production

config/database.yml

```
development:
  adapter: mysql
  database: cs683BookStore_development
  username: whitney
  password:
  host: localhost

test:
  adapter: mysql
  database: cs683BookStore_test
  username: whitney
  password:
  host: localhost

production:
  adapter: mysql
  database: cs683BookStore_production
  username: whitney
  password:
  host: localhost
```

Create Rails Site & Model

```
Al 10 -> rails cs683BookStore
Al 11 -> cd cs683BookStore

Al 12 -> ruby script/generate model book
      exists  app/models/
      exists  test/unit/
      exists  test/fixtures/
      create  app/models/book.rb
      create  test/unit/book_test.rb
      create  test/fixtures/books.yml
      create  db/migrate
      create  db/migrate/001_create_books.rb
```

The Model

app/models/book.rb

```
class Book < ActiveRecord::Base
end
```

Creating Books Table

Edit db/migrate/001_create_books.rb to be:

```
class CreateBooks < ActiveRecord::Migration
  def self.up
    create_table :books do |t|
      t.column :title, :string
      t.column :author, :string
      t.column :date, :date
    end
  end

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

Rails Convention

Model singular	-> Table plural
Book	books

Run the migration by:

```
Al 13->rake db:migrate
(in /Users/whitney/rails/cs683BookStore)
== CreateBooks: migrating ==================
-- create_table(:books, {:options=>""})
        -> 0.1035s
== CreateBooks: migrated (0.1063s) ===========
```

Setting Versions

To set the database to migration version X
rake db:migrate VERSION=X

```
Al 14->rake db:migrate VERSION=0
(in /Users/whitney/rails/cs683BookStore)
== CreateBooks: reverting ============================
-- drop_table(:books)
   -> 0.4035s
== CreateBooks: reverted (0.4079s) ====================

Al 15->rake db:migrate VERSION=1
(in /Users/whitney/rails/cs683BookStore)
== CreateBooks: migrating ============================
-- create_table(:books, {:options=>""})
   -> 0.4582s
== CreateBooks: migrated (0.4611s) ====================
```

Some Database Glitches

MySQL 3.x & Rails Migration Issue

Opinionated Software

MySQL 3.x Table Create syntax uses 'TYPE=InnoDB'

MySQL 4.x & 5.x Table Create syntax uses 'ENGINE=InnoDB'

Rails Migration hard codes MySQL 4.x & 5.x syntax

MySQL 3.x & Rails Migration Solution 1

Set the options in create table to either:

''

'TYPE=InnoDB'

```
class CreateBooks < ActiveRecord::Migration
  def self.up
    create_table :books, :options => '' do |t|
      t.column :title, :string
      t.column :author, :string
      t.column :date, :date
    end
  end

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

You will not be able to use Unit tests with this solution

MySQL 3.x & Rails Migration Solution 2

Edit Rails Source Code mysql_adapter.rb

/usr/lib/ruby/gems/1.8/gems/
activerecord-1.14.4/lib/active_record/connection_adapters/mysql_adapter.rb

Change create_table to be:

```
def create_table(name, options = {}) #:nodoc:
  #super(name, {:options => "ENGINE=InnoDB"}.merge(options))
  super(name, options)
end
```

Location and activerecord version may differ

MySQL 3.x & Rails Migration Solutions 3 & 4

Don't use MySQL 3.x

Don't use migrations wth MySQL 3.x

PostgreSQL & Rails

Use only lower case names for
tables
columns

Accessing/Testing Database

Auto-Generated Unit Test

test/unit/book_test.rb

```
require File.dirname(__FILE__) + '/../test_helper'

class BookTest < Test::Unit::TestCase

  # Replace this with your real tests.
  def test_truth
    assert true
  end

end
```

Creating the Test tables

Al 16-> rake db:test:prepare

Every time you change the tables you must run this command to create the tables in the test database

Tests use the test database set up in you database.yml file

Q. Why do we have to do this manually?
Rails should run this for us if needed

Running the Test

```
Al 17->ruby test/unit/book_test.rb
Loaded suite test/unit/book_test
Started
..
Finished in 0.192902 seconds.

1 tests, 1 assertions, 0 failures, 0 errors
```

If the test runs successfully you have accessed the datebase

The empty test is useful

Adding Some Test Data

Edit test/fixtures/books.yml so it is:

```
# Read about fixtures at http://ar.rubyonrails.org/classes/Fixtures.html
ruby_book:
  id: 1
  title:   Programming Ruby
  author: Thomas
  date: 2005-01-01
rails_book:
  id: 2
  title:  Agile Web Development with Rails
  author: Thomas & Hansson
  date: 2005-8-10
```

Do not use any tabs in this file

A Real Test

test/unit/book_test.rb

```
require File.dirname(__FILE__) + '/../test_helper'

class BookTest < Test::Unit::TestCase
  fixtures :books

  def test_Read
    ruby = Book.find_by_title('Programming Ruby')
    assert_equal(ruby.author, 'Thomas')
    assert_equal(ruby.id, 1)
    assert_equal(ruby.date, Date.new(2005,1,1))
  end
end
```

Running the Test

```
Al 18->ruby test/unit/book_test.rb
Loaded suite test/unit/book_test
Started
..
Finished in 0.12832 seconds.

1 tests, 3 assertions, 0 failures, 0 errors
```

Default Views

Creating a Controller

```
Al 19->ruby script/generate controller bookStoreAdmin
        exists  app/controllers/
        exists  app/helpers/
        create  app/views/book_store_admin
        exists  test/functional/
        create  app/controllers/book_store_admin_controller.rb
        create  test/functional/book_store_admin_controller_test.rb
        create  app/helpers/book_store_admin_helper.rb
```

Using Default Views

Edit
app/controllers/book_store_admin_controller.rb
so it is:

```
class BookStoreAdminController < ApplicationController
   scaffold :book
end
```

http://127.0.0.1:3000/book_store_admin

Scaffolding

http://127.0.0.1:3000/book_store_admin

Listing books

Title Author Date

New book

Scaffolding

http://127.0.0.1:3000/book_store_admin/new

New book

Title

Cat in the Hat

Author

Dr. Suess

Date

2001 October 31

Create

Scaffolding

http://127.0.0.1:3000/book_store_admin/list

Book was successfully created

Listing books

Title	Author	Date			
Cat in the Hat	Dr. Suess	2001-10-31	Show	Edit	Destroy

New book

Controller Names => urls

BookStoreAdminController => http://127.0.0.1:3000/book_store_admin/

CatRatBAtController => http://127.0.0.1:3000/cat_rat_b_at/

Use of Default Views

Default views are generated on the fly

Allows you to work model without dealing with views

Generating Views

Generating Concrete Views

```
Al 20->ruby script/generate scaffold book
      exists  app/controllers/
      exists  app/helpers/
      create  app/views/books
      exists  test/functional/
  dependency  model
      exists    app/models/
      exists    test/unit/
      exists    test/fixtures/
    identical    app/models/book.rb
        skip    test/unit/book_test.rb
        skip    test/fixtures/books.yml
      create  app/views/books/_form.rhtml
      create  app/views/books/list.rhtml
      create  app/views/books/show.rhtml
      create  app/views/books/new.rhtml
      create  app/views/books/edit.rhtml
      create  app/controllers/books_controller.rb
      create  test/functional/books_controller_test.rb
      create  app/helpers/books_helper.rb
      create  app/views/layouts/books.rhtml
      create  public/stylesheets/scaffold.css
```

Generates concrete versions of default views so we can modify them

Generated app/controllers/books_controller.rb

```
class BooksController < ApplicationController
  def index
    list
    render :action => 'list'
  end

  # GETs should be safe (see http://www.w3.org/2001/tag/doc/whenToUseGet.html)
  verify :method => :post, :only => [ :destroy, :create, :update ],
         :redirect_to => { :action => :list }

  def list
    @book_pages, @books = paginate :books, :per_page => 10
  end

  def show
    @book = Book.find(params[:id])
  end

  def new
    @book = Book.new
  end
```

app/controllers/books_controller.rb

```
def create
  @book = Book.new(params[:book])
  if @book.save
    flash[:notice] = 'Book was successfully created.'
    redirect_to :action => 'list'
  else
    render :action => 'new'
  end
end

def edit
  @book = Book.find(params[:id])
end

def update
  @book = Book.find(params[:id])
  if @book.update_attributes(params[:book])
    flash[:notice] = 'Book was successfully updated.'
    redirect_to :action => 'show', :id => @book
  else
    render :action => 'edit'
  end
end

  def destroy
    Book.find(params[:id]).destroy
    redirect_to :action => 'list'
  end
end
```

Layout

Template for pages

app/views/layouts/books.rhtml

```
<html>
<head>
  <title>Books: <%= controller.action_name %></title>
  <%= stylesheet_link_tag 'scaffold' %>
</head>
<body>

<p style="color: green"><%= flash[:notice] %></p>

<%= yield  %>

</body>
</html>
```

Generated CSS

public/stylesheets/scaffold.css

```
body { background-color: #fff; color: #333; }

body, p, ol, ul, td {
  font-family: verdana, arial, helvetica, sans-serif;
  font-size:   13px;
  line-height: 18px;
}

pre {
  background-color: #eee;
  padding: 10px;
  font-size: 11px;
}

a { color: #000; }
a:visited { color: #666; }
a:hover { color: #fff; background-color:#000; }

.fieldWithErrors {
  padding: 2px;
  background-color: red;
  display: table;
}

#errorExplanation {
  width: 400px;
  border: 2px solid red;
  padding: 7px;
  padding-bottom: 12px;
  margin-bottom: 20px;
  background-color: #f0f0f0;
}

#errorExplanation h2 {
  text-align: left;
  font-weight: bold;
  padding: 5px 5px 5px 15px;
  font-size: 12px;
  margin: -7px;
  background-color: #c00;
  color: #fff;
}

etc.
```

app/views/books/new.rhtml

```
<h1>New book</h1>

<%= start_form_tag :action => 'create' %>
  <%= render :partial => 'form' %>
  <%= submit_tag "Create" %>
<%= end_form_tag %>

<%= link_to 'Back', :action => 'list' %>
```

view is placed inside the layout

This view calls a partial -
code & html shared between views

app/views/books/_form.rhtml

```
<%= error_messages_for 'book' %>

<!--[form:book]-->
<p><label for="book_title">Title</label><br/>
<%= text_field 'book', 'title'  %></p>

<p><label for="book_author">Author</label><br/>
<%= text_field 'book', 'author'  %></p>

<p><label for="book_date">Date</label><br/>
<%= date_select 'book', 'date'  %></p>
<!--[eoform:book]-->
```

Form partial use in previous slide

Partial file name start with an underscore