Rails from the Ground Up!

with Jared Richardson
RoleModel Software

Rails
Part Two
A Blog

Our First App

Create a blog application

Model

database migrations

Unit tests

Sample data

Controllers

Routes

Create/Read/Update/Delete HTML pages

Helpers

Blog

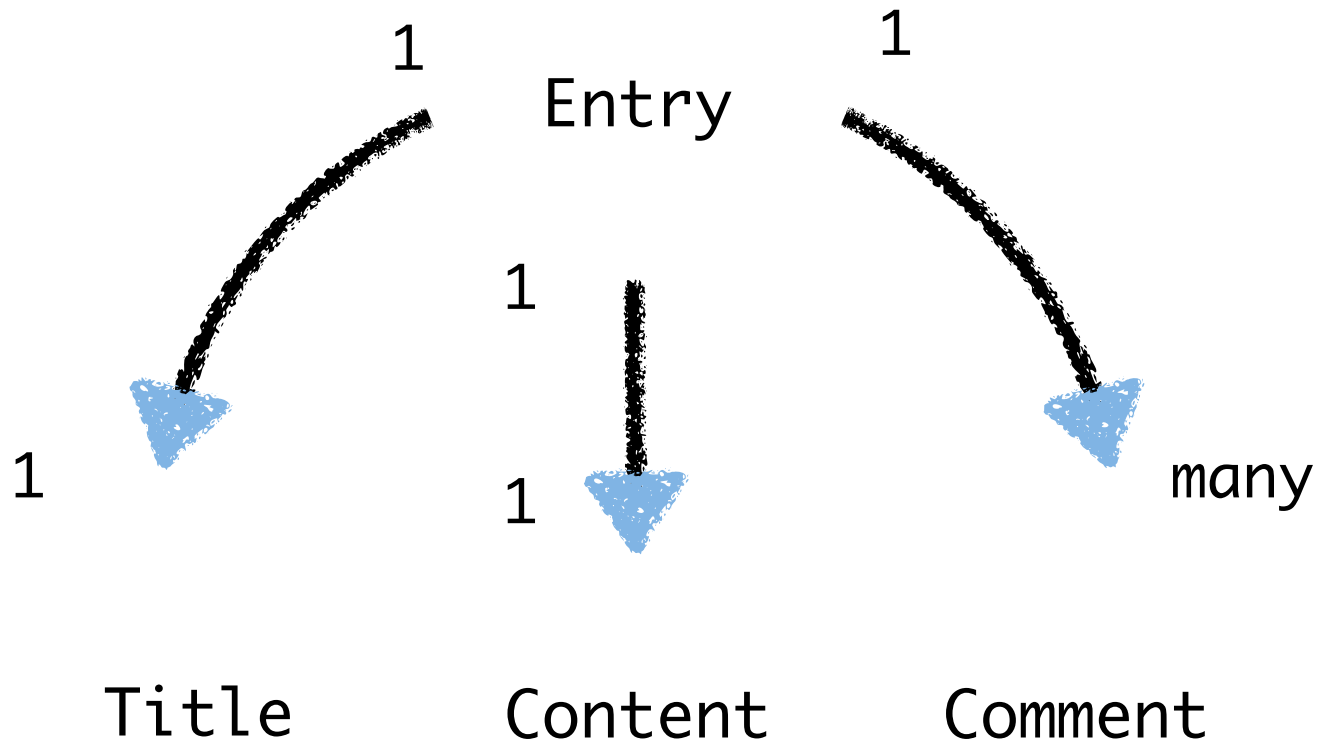
Entry

- has one title
- has one body (content)
- has many comments

Comments

- belong to an article

Blog



Ready?

```
rails new blog

cd blog

rails scaffold Blog_Entry name:string title:string body:text
```

Our First App

~~Create a blog application~~

Model

database migrations

Unit tests

Sample data

Controllers

Routes

Create/Read/Update/Delete HTML pages

Helpers

Examine the Model

app/model/entry.rb

```
class Entry < ActiveRecord::Base

end
```

rails console

```
e = Entry.new
```

What do you see?

```
ActiveRecord::StatementInvalid: Could not find table 'entries'
```

Our First App

~~Create a blog application~~

~~Model~~

database migrations

Unit tests

Sample data

Controllers

Routes

Create/Read/Update/Delete HTML pages

Helpers

Database Migrations

```
rake db:migrate
```

```
jareds-desktop:blog jared$ rake db:migrate
==  CreateEntries: migrating ==================================================
-- create_table(:entries)
   -> 0.0059s
==  CreateEntries: migrated (0.0060s) =========================================
```

Back to the Console

```
rails c

e = Entry.new
```

```
1.9.3-p0 :001 > e = Entry.new
 => #<Entry id: nil, name: nil, title: nil, body: nil, created_at: nil, updated_at: nil>
```

Save It

```
e.save!
```

What do you see?

```
1.9.3-p0 :008 > e.save
   (0.1ms)  begin transaction
  SQL (0.8ms)  INSERT INTO "entries" ("body", "created_at", "name", "title"
, "updated_at") VALUES (?, ?, ?, ?, ?)  [["body", nil], ["created_at", Sun,
 19 Feb 2012 01:45:37 UTC +00:00], ["name", nil], ["title", nil], ["updated
_at", Sun, 19 Feb 2012 01:45:37 UTC +00:00]]
   (1.3ms)  commit transaction
 => true
```

```
INSERT INTO "entries"
("body", "created_at", "name", "title", "updated_at")
VALUES (?, ?, ?, ?, ?)
[["body", nil],
["created_at", Sun, 19 Feb 2012 01:45:37 UTC +00:00],
["name", nil],
["title", nil],
["updated_at", Sun, 19 Feb 2012 01:45:37 UTC +00:00]]
```

Are nils Valid?

Let's fix it

Active Record validations

http://api.rubyonrails.org/classes/ActiveRecord/Associations/ClassMethods.html

validates

```
class Entry < ActiveRecord::Base

  validates :name,  :presence => true

  validates :title, :presence => true,

                    :length => { :minimum => 5 }

end
```

reload!

rails console command

Reloads code on disk

Does NOT re-load existing instances

Recreate & save

```
e = Entry.new

e.save
```

What's Wrong?

```
1.9.3-p0 :032 > e.save
   (0.3ms)  begin transaction
   (0.1ms)  rollback transaction
 => false
1.9.3-p0 :033 >
```

1) Returned False

```
1.9.3-p0 :032 > e.save
   (0.3ms)  begin transaction
   (0.1ms)  rollback transaction
 =>
1.9.3-p0 :033 >
```

2) It's Not This!

```
1.9.3-p0 :032 > e.save
   (0.3ms)  begin transaction
   (0.1ms)  rollback transaction
 => false
```

```
1.9.3-p0 :023 > e.save
   (0.1ms)  begin transaction
   (0.4ms)  UPDATE "entries" SET "name" = 'My First Blog Entry!
', "updated_at" = '2012-02-19 03:01:10.240021' WHERE "entries".
"id" = 4
   (1.1ms)  commit transaction
 => true
```

Saving

```
# runs with no feedback on errors

e.save

# runs and throws exceptions on errors

e.save!
```

errors

```
e = Entry.new

e.valid? # or e.save or e.save!

errors = e.errors

e.errors[:name]

errors.full_messages
```

error listing as a hash

```
errors.full_messages.each{ | msg |
  puts msg
}
```

Fix It!

```
e.errors[:name]  # shows name's errors

e.name="My first blog entry!"

e.valid?

e.errors[:name]

:)
```

What Else?

```
e.errors

e.errors[:title]

["can't be blank",

"is too short (minimum is 5 characters)"]

e.title = "A Freshly Minted Entry!"

e.valid?

TRUE!
```

Save It

```
e.save

What's e's ID?

Entry.all

Oops...

Invalid data
```

Clean Up

```
# wipe out and reset your database

rake db:reset

rails console

# now recreate a blog entry

e = Entry.new
```

Does It Run?

```
rails server
```

http://localhost:3000/Blog

Try Again

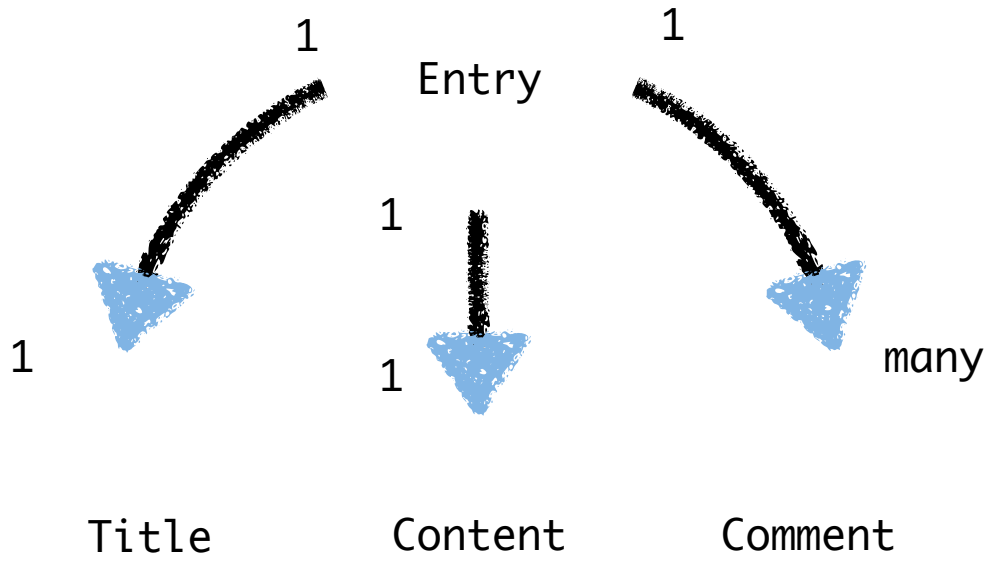
http://locahost:3000/entries

Add a body

(In console) e = Entry.first

e.body

Remember Comments?



Association

has many

Comments

```
rails generate model Comment
      author:string title:string body:text
rake routes
```

Load the comments page

What Did We Miss?

Create a Link

open app/models/*.rb

```ruby
class Comment < ActiveRecord::Base
  belongs_to :entry
end

```

```ruby
class Entry < ActiveRecord::Base
  validates :name,  :presence => true
  validates :title, :presence => true,
                    :length => { :minimum => 5 }
  has_many :comments
end
```

Associations

```
has_one

has_many

belongs_to

has_and_belongs_to_many

# belongs_to in the model w/ the foreign key
```

Our First App

~~Create a blog application~~

~~Model~~

~~database migrations~~

Unit tests

Sample data

Controllers

Routes

Create/Read/Update/Delete HTML pages

Helpers

Unit Tests

Our First App

~~Create a blog application~~

~~Model~~

~~database migrations~~

~~Unit tests~~

Sample data

Controllers

Routes

Create/Read/Update/Delete HTML pages

Helpers

Sample Data

Our First App

~~Create a blog application~~

~~Model~~

~~database migrations~~

~~Unit tests~~

~~Sample data~~

Controllers

Routes

Create/Read/Update/Delete HTML pages

Helpers

Controllers

Our First App

~~Create a blog application~~

~~Model~~

~~database migrations~~

~~Unit tests~~

~~Sample data~~

~~Controllers~~

Routes

Create/Read/Update/Delete HTML pages

Helpers

Routes

Our First App

~~Create a blog application~~

~~Model~~

~~database migrations~~

~~Unit tests~~

~~Sample data~~

~~Controllers~~

~~Routes~~

Create/Read/Update/Delete HTML pages

Helpers

CRUD HTML

Now, you have to tell Rails where your actual home page is located. Open the file `config/routes.rb` in your editor. This is your application's *routing file* which holds entries in a special DSL (domain-specific language) that tells Rails how to connect incoming requests to controllers and actions. This file contains many sample routes on commented lines, and one of them actually shows you how to connect the root of your site to a specific controller and action. Find the line beginning with `root :to` and uncomment it. It should look something like the following:

```
Blog::Application.routes.draw do

  #...
  # You can have the root of your site routed with "root"
  # just remember to delete public/index.html.
  root :to => "home#index"
```

The `root :to => "home#index"` tells Rails to map the root action to the home controller's index action.

Now if you navigate to http://localhost:3000 in your browser, you'll see `Hello, Rails!`.

6.2 Adding a Link

To hook the posts up to the home page you've already created, you can add a link to the home page. Open `app/views/home/index.html.erb` and modify it as follows:

```
<h1>Hello, Rails!</h1>
<%= link_to "My Blog", posts_path %>
```

The `link_to` method is one of Rails' built-in view helpers. It creates a hyperlink based on text to display and where to go – in this case, to the path for posts.

Our First App

~~Create a blog application~~

~~Model~~

~~database migrations~~

~~Unit tests~~

~~Sample data~~

~~Controllers~~

~~Routes~~

~~Create/Read/Update/Delete HTML pages~~

Helpers

Helpers

Our First App

~~Create a blog application~~

~~Model~~

~~database migrations~~

~~Unit tests~~

~~Sample data~~

~~Controllers~~

~~Routes~~

~~Create/Read/Update/Delete HTML pages~~

~~Helpers~~

Next...

Database migrations...