



RUBY ON RAILS

“Web development that doesn't hurts”

PARTE II: ROR

- Contenido
 - Introducción al framework
 - Componentes clave
 - Testing
 - JavaScript // CoffeeScript
 - LiveController
 - RailsAvanzado
 - APIs y Servicios

INTRODUCCIÓN A RAILS

- Creado por David Hanson (HDD)
- 2004 se publica la primera beta
- La empresa de HDD, 37Signals, es la encargada de gestionar el core
- Sale a partir del proyecto Basecamp de 37S
- La combinación con Ruby hace un framework muy potente y flexible. Reduce el time 2 market.

MITOS SOBRE RAILS

- Rails es un framework demasiado nuevo
- Rails es difícil de desplegar en sistemas
- Rails no tiene comunidad
- No escala
- No es multi-thread (comparémoslo con el siguiente...)
- No es concurrente

USAN RAILS

EA

MicroSites

YellowPages

Twitter

algo le queda

**NewYork
Times**

Groupon

GitHub

Ask.fm

Basecamp

Hulu

ThemeForrest

SlideShare

Scribid

VK

Shopify

change.org

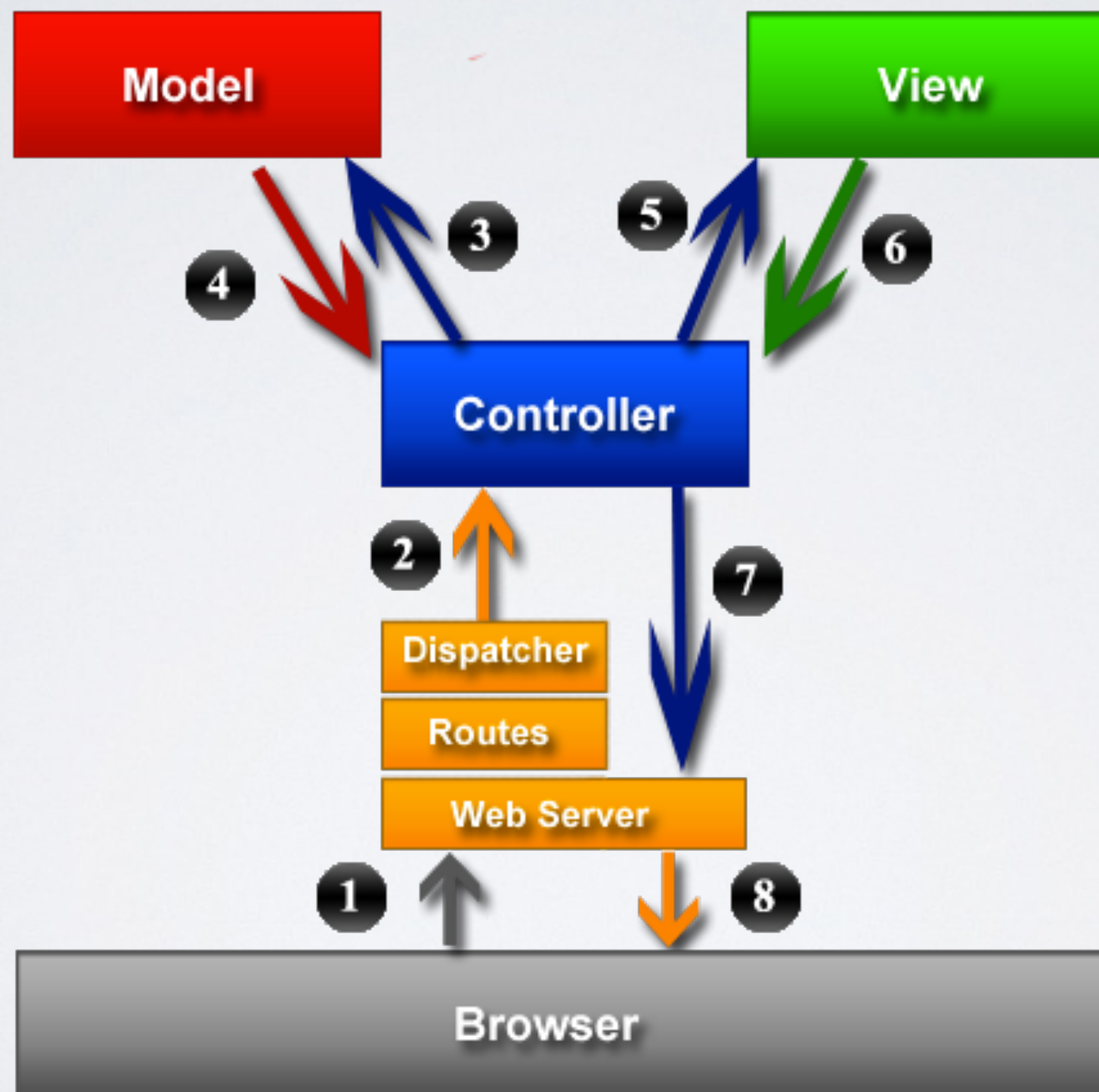
CARACTERÍSTICAS

- Arquitectura MCV
- ORM (ActiveRecord)
- Convención sobre configuración
- Principios DRY
- Embedded Ruby para las vistas (*.erb)
- jQuery como framework de JavaScript por defecto

CARACTERÍSTICAS

- Testing completo incluido
- Conexiones permanentes con LiveController
- Gestión de la caché básica o avanzada (Russian doll caching)
- Turbolinks
- Sistema de comandos muy completo

MVC



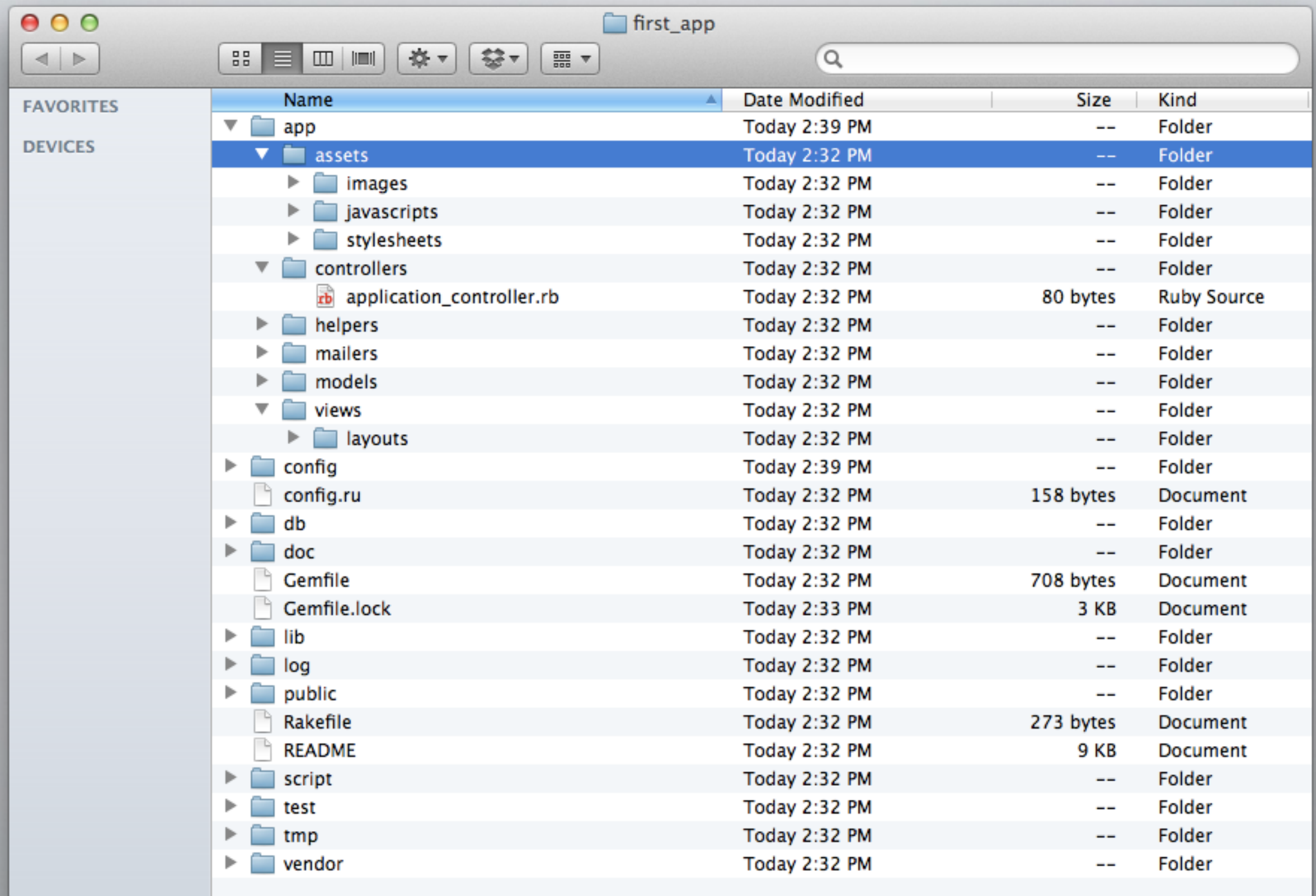
CREACIÓN DE UNA APP

```
# Creación de una app  
$ rails new nombre_app
```

.....

```
# Variaciones  
-m "plantilla.rb"      # usa un template  
-d "base_de_datos"     # modifica el adapter  
-edge                  # usa vers. edge
```

FICHEROS



Name	Date Modified	Size	Kind
app	Today 2:39 PM	--	Folder
assets	Today 2:32 PM	--	Folder
images	Today 2:32 PM	--	Folder
javascripts	Today 2:32 PM	--	Folder
stylesheets	Today 2:32 PM	--	Folder
controllers	Today 2:32 PM	--	Folder
application_controller.rb	Today 2:32 PM	80 bytes	Ruby Source
helpers	Today 2:32 PM	--	Folder
mailers	Today 2:32 PM	--	Folder
models	Today 2:32 PM	--	Folder
views	Today 2:32 PM	--	Folder
layouts	Today 2:32 PM	--	Folder
config	Today 2:39 PM	--	Folder
config.ru	Today 2:32 PM	158 bytes	Document
db	Today 2:32 PM	--	Folder
doc	Today 2:32 PM	--	Folder
Gemfile	Today 2:32 PM	708 bytes	Document
Gemfile.lock	Today 2:33 PM	3 KB	Document
lib	Today 2:32 PM	--	Folder
log	Today 2:32 PM	--	Folder
public	Today 2:32 PM	--	Folder
Rakefile	Today 2:32 PM	273 bytes	Document
README	Today 2:32 PM	9 KB	Document
script	Today 2:32 PM	--	Folder
test	Today 2:32 PM	--	Folder
tmp	Today 2:32 PM	--	Folder
vendor	Today 2:32 PM	--	Folder

ENTORNOS & CONF

- En la carpeta environments encontramos diferentes configuraciones. Tres por defecto.
 - Development
 - Test
 - Production
- database.yml guarda la configuración de la bbdd

CREACIÓN DE UNA APP

```
# Creación de una app  
$ rails new nombre_app
```

.....

```
# Variaciones  
-m "plantilla.rb"      # usa un template  
-d "base_de_datos"     # modifica el adapter  
-edge                  # usa vers. edge
```


SCAFFOLDING

- Técnica de andamiaje
 - generación rápida de los principales componentes de un objeto
 - se indica el objeto, sus atributos y el tipo de dato
 - fichero de “migración”, modelo, controlador, vistas, tests, helpers, javascript y css

SCAFFOLDING

Creación de un scaffold

\$ rails g scaffold *modelo* [*atributos...*]

otros generadores

\$ rails g layout *nombre*

\$ rails g model *nombre atributos*

\$ rails g controller *nombre métodos*

FICHEROS DE MIGRACIÓN

- Contienen las instrucciones para trabajar sobre las tablas de las base de datos
- Son independientes del SGBD
- Gestiona la marcha atrás en caso de error con el rollback

FICHEROS DE MIGRACIÓN

```
class CreatePosts < ActiveRecord::Migration
  def change
    create_table :posts do |t|
      t.string :titulo
      t.text :contenido

      t.timestamps
    end
  end
end
```


FICHEROS DE MIGRACIÓN

- Las migraciones además gestionan:
 - carga de datos programadas en el seed.rb
 - la versión de la bbdd actual y controla que migración debe ejecutar
 - create, drop, truncate de la base de datos

FICHEROS DE MIGRACIÓN

Podemos usar los siguientes comandos rake

\$ rake db:create	# crea la bbdd
\$ rake db:migrate	# carga migraciones
\$ rake db:seed	# relleno de bbdd
\$ rake db:rollback	# marcha atrás

FICHEROS DE MIGRACIÓN

- Las migraciones además gestionan:
 - carga de datos programadas en el seed.rb
 - la versión de la bbdd actual y controla que migración debe ejecutar
 - create, drop, truncate de la base de datos

BLOG EN 20 MINUTOS

```
# Vamos a crear nuestra primera app rails
```

```
$ rails new my_blog
```

```
$ cd my_blog
```

```
# configuramos la base de datos
```

```
$ rails g scaffold Post title:string  
content:text
```

```
$ rails g scaffold Comment content:text  
post:references
```

```
$ rake db:create
```

```
$ rake db:migrate
```

```
$ rails server
```