



Introducción a Rails

Rodrigo Rodriguez
rorodr@gmail.com
Skype: rarodriguezr

Conceptos básicos en Web (1/2)

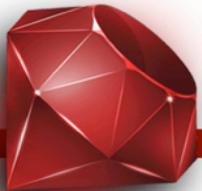


CSS
Javascript

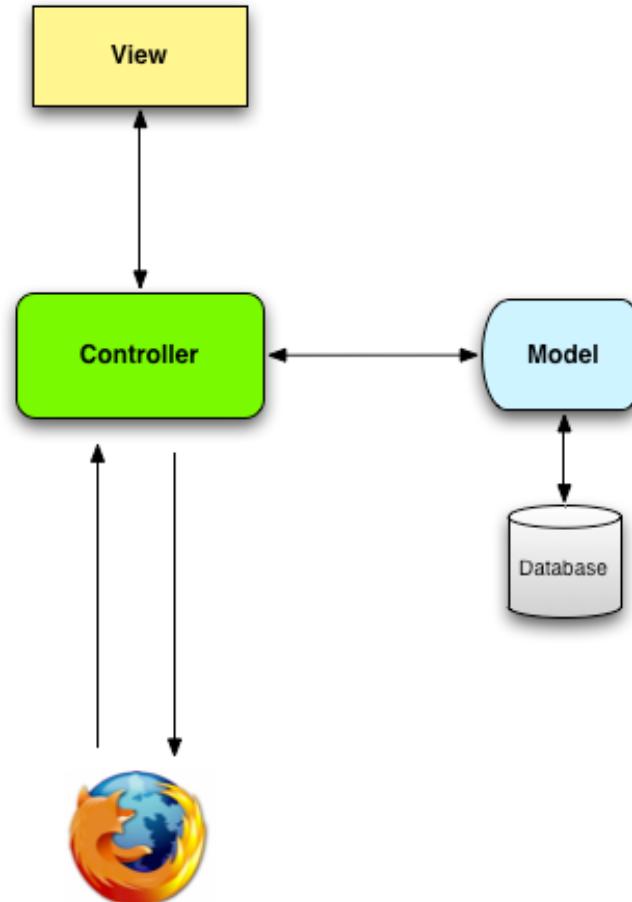
Apache,
Websphere, etc..

MySQL, Oracle,
postgreSQL

Modelo-Vista-Controlador



Patrón de arquitectura, el cual refuerza la separación de la lógica del negocio de la representación lógica asociada con la interfaz de usuario y el acceso a la información requerida por el usuario.



Conceptos básicos en Web (2/2)



- **GUI de usuario:** Interfaz web, la cual es interpretada por el navegador Web. En la gran mayoría de las aplicaciones, se utilizan HTML, CSS y Javascript para desplegar la información
- **Servidor Web:** Aplicación que ejecuta nuestro código en Rails en distintos hilos/procesos.
- **Base de Datos:** Sistema de almacenamiento de la información requerida en la aplicación

RESTful_(1/2)



- REST ó Representational State Transfer: trata de facilitar la manera en que se representa la transferencia de la información.
- Se basa principalmente en 2 principios:
 - Utilizar identificadores del recurso como parte del URL, con el fin de representarlos fácilmente.
 - Transferir representaciones del estado de ese recurso entre componentes del sistema (métodos)

DELETE /photos/17

RESTful_(2/2)



- **Métodos HTTP:**
 - **GET**: devolver un valor
 - **POST**: almacenar un valor
 - **PATCH**: modificar un valor
 - **DELETE**: Eliminar un elemento
- Algunos URLs:

GET /photos/17	DELETE /photos/17
POST /photos	GET /photos
PATCH /photos/17	

Base de Datos



- Sistema de almacenamiento de información
- Existen diversos tipos de BD:
 - **No relacionales (NoSQL)**: No requieren estructuras fijas como tablas, normalmente no soportan operaciones JOIN, ni garantizan completamente Atomicidad, Consistencia, Aislamiento y Durabilidad). Ej. MongoDB, Redis.
 - **Relacionales**: Por lo general muy usados, constan de información almacenada en tablas estructuradas que se pueden relacionar entre sí. Ej: mySQL, postgreSQL, Oracle, etc.

Bases de Datos relacionales (1/2)



- Permiten establecer interconexiones (relaciones) entre los datos (que están guardados en tablas), y a través de dichas conexiones relacionar los datos de ambas tablas
- Se pueden relacionar múltiples tablas entre sí en formato 1-1, 1-N, N-M

users	
id	Integer
name	String
email	String
phone	String
created_at	Datetime

1

videos	
id	Integer
title	String
director	String
productor	String
user_id	Integer
created_at	Datetime

1

SQL – Sintaxis básica



- Structured Query Language: Es un lenguaje para obtener información desde una base de datos relacional. Su sintaxis básica consta de:
 - SELECT: Elegir elementos (columnas) de una tabla
 - WHERE: Elegir la condición que deben de cumplir
 - ORDER BY: definir el orden de los datos

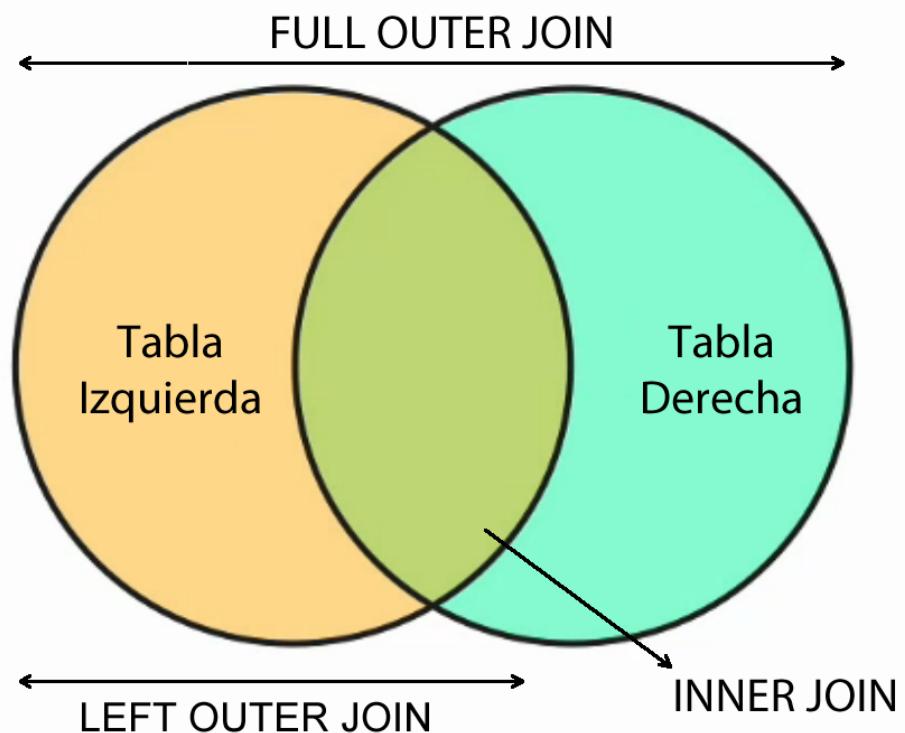
```
SELECT Orders.OrderID, Customers.CustomerName,  
Orders.OrderDate  
FROM Orders, Customers  
WHERE Orders.CustomerID=Customers.CustomerID;
```

SQL – Joins



- Permite relacionar la información de 2 o más tablas
- Existen diversos tipos de Join: INNER_JOIN (JOIN), OUTER_JOIN

```
SELECT Customers.CustomerName,  
Orders.OrderID  
FROM Customers  
INNER JOIN Orders  
ON Customers.CustomerID =  
Orders.CustomerID  
ORDER BY Customers.CustomerName;
```



Ejercicio 1: SQL



- Ingresar a <http://tiny.cc/9nkhxx>
 - Escribir una consulta SQL que muestre únicamente los nombres de todos los Productos cuyo “price” sea menor que 22
 - Escribir una consulta SQL que muestre todos los productos cuyo “Supplier” tenga de “Country” USA o Brazil.

http://www.w3schools.com/sql/trysql.asp?filename=trysql_select_orderby

Rails



- Framework de desarrollo ágil para aplicaciones Web
- Creado por David Heinemeier en 2005
- Es un framework que ha servido de base para la creación de librerías semejantes en otros lenguajes.

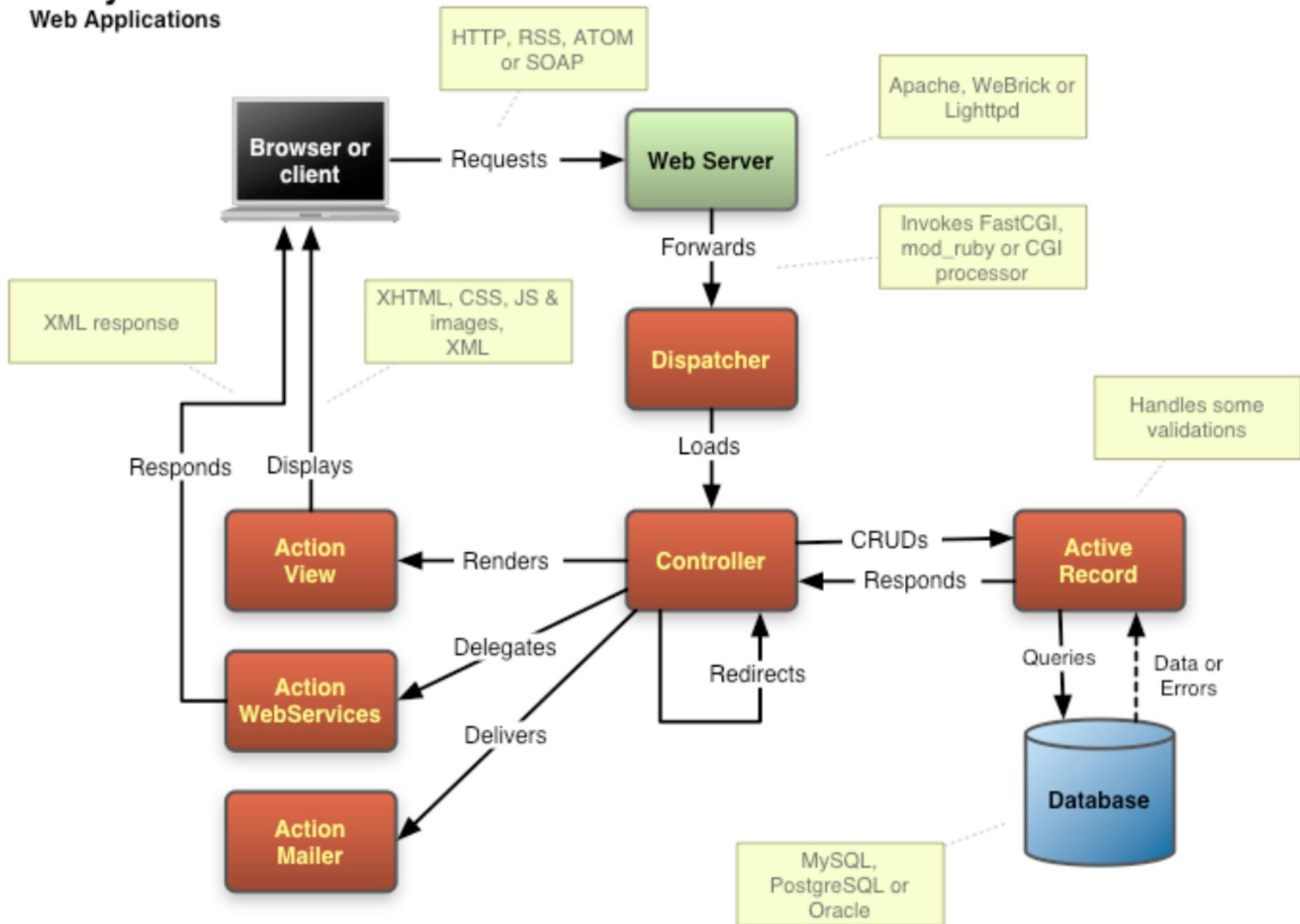
Principios de Rails



- Convención sobre configuración
- DRY (Don't repeat yourself)
- Modelo-Vista-Controlador
- RESTFul

Ruby on Rails

Web Applications



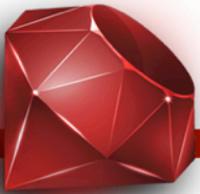
Componentes de Rails



- Action Pack
 - Action Controller
 - Action Dispatch
 - Action View
- Action Mailer
- Active Model
- Active Record
- Active Resource
- Active Support
- Railties ...



Organización de Archivos (1/2)



- **app:** carpeta principal del código, incluye models, views, controllers, helpers, assets.
- **config:** Carpeta principal de configuración, al iniciar la aplicación se cargan muchos de los valores.
- **db:** Archivos para generar cambios a la Base de datos
- **lib:** componentes o librerías genéricas de la aplicación
- **log:** registro (log) de los archivos de aplicación
- **public:** Información pública de la aplicación
- **test/rspec:** tests automatizados para verificación del código.

Organización de Archivos (2/2)



- **tmp**: Archivos temporales
- **vendor**: Librerías externas de la aplicación
- **README**: Descripción breve de la aplicación
- **Rakefile**: Definir tareas disponibles para el comando Rake
- **Gemfile**: Gemas de las cuales depende el proyecto
- **config.ru**: Archivo de configuración para Rack
- **.gitignore**: Archivos que deben ser ignorados por GIT

Instalación de Rails



- En una terminal:
 - `gem install rails`
 - `gem install bundler`

Creación de una aplicación



- Rails provee un generador automático de código/proyecto
- Para crear un nuevo proyecto, solamente se requiere escribir en una terminal:
 - `rails new {APP_NAME}`
- La aplicación generada va a contener la estructura básica del proyecto

Iniciar la aplicación generada



- Rails por defecto contiene un servidor para desarrollo (WebBrick), el cual permite la ejecución de un servidor de aplicación de un solo hilo para nuestra aplicación.
- Para ejecutar el servidor basta correr:
 - rails server
- Por defecto, el servidor corre en el puerto 3000. Por lo que para ver nuestro sitio, hay que escribir en el navegador “localhost:3000”

Utilizando Bundler



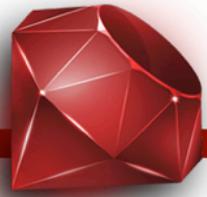
- Bundler permite la instalación automática de todas las gemas que requiere un proyecto de Ruby/Rails.
- Bundler toma de base el archivo “Gemfile” para indicar las dependencias
- Para ejecutar la instalación, basta correr:
 - bundler install

Generando la “magia”



- En Rails, existe un generador de código que permite la creación de migraciones, modelos, controladores y scaffolds.
- Para ejecutar el generador simplemente ocupamos utilizar:
 - rails generate scaffold {MODEL} {ATTR}:{TYPE}

Generador de Scaffold



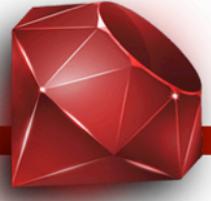
- Crea todo el código que requiere rails para efectuar la administración de una tabla de BD.
- Includes:
 - Migración de BD
 - Agrega registros
 - Un controlador, modelo y helper
 - Todas las vistas (CRUD)
 - Test de unidad/funcional para el código
 - Un layout, scaffold.css y application.js (si no existen)

Migraciones de BD



- Las migraciones en Rails, permite la ejecución de scripts en la BD
- Cada migración debe de ser ejecutada en todas las BD relacionadas al proyecto
- Existen 2 formas distintas de declarar una migración, con “self_up”/“self_down” o con “Change”
- Para ejecutar las migraciones es necesario ejecutar
 - `rake db:migrate`

Manejo de Rutas



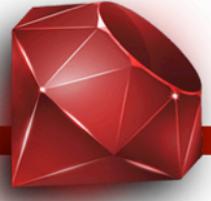
- Un archivo de configuración que permite asociar un URL con un método de un controlador que va a atender la petición
- Rutas (Routes) pueden tener definiciones complejas (Resources) o definir una ruta única
- Rutas pueden contener validaciones y condiciones que deben de cumplir la petición para que se pueda invocar al método asociado.

Convenciones de nombres



Nombre del Modelo	
<i>Tabla</i>	line_items
<i>Archivo</i>	app/models/line_item.rb
<i>Clase</i>	LineItem
Nombres en el Controlador	
<i>URL</i>	http://..../store/list
<i>Archivo</i>	app/controllers/stores_controller.rb
<i>Clase</i>	StoresController
Estructura de la parte visual de la aplicación	
<i>Layout</i>	app/views/layouts/default.html.erb
Convenciones de nombres	
<i>URL</i>	http://..../stores/list
<i>Archivo</i>	app/views/stores/list.html.erb
<i>Helper</i>	module StoresHelper
<i>File</i>	app/helpers/stores_helper.rb

Convenciones de BD



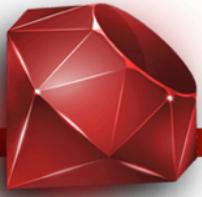
- Las tablas deben de tener su nombre en plural, minúscula, guión bajo entre las palabras y preferiblemente escritas en inglés
 - orders, users, videos, ...
- Las llaves primarias de la tabla deben de llamarse “id” y es preferible utilizar valores autoincrementales para manejar esto.

Rails DB Naming Conventions



- Llaves foráneas (o llave de relación entre tablas), deben de llamarse: 'table_name_id'
- Tablas de relaciones M-N deben de contener los nombres de las tablas relacionadas, ordenadas alfabéticamente y separadas por un guión bajo.

Documentos relacionados



- <http://ruby.railstutorial.org/book/ruby-on-rails-tutorial>
- Agile Development with Rails, fourth Edition
- Rails for Zombies: <http://railsforzombies.org/>
- <http://guides.rubyonrails.org/>



Trabajando con Heroku

Rodrigo Rodríguez

Heroku



- Es un sitio para desplegar aplicaciones Web de manera gratuita (inicialmente solo soportaba Rails, pero ahora soporta otras tecnologías).
- Cuenta con distintos planes, desde gratuitos hasta cuentas profesionales con costos un tanto elevados.
- Utiliza una arquitectura compleja, que le permite que no haya un servidor propio donde se ejecute el código, lo que facilita mucho la instalación de las aplicaciones.

Heroku: Configuración



- La configuración de Heroku es simple:
 - Crear una cuenta en heroku (es gratis)
 - Instalar el Toolbelt de Heroku y asegurarse de que Git está configurado
 - Digitar en una consola:
 - `heroku login`

Más información: <https://devcenter.heroku.com/articles/getting-started-with-ruby#introduction>

Heroku: Crear una aplicación



- Se requiere de la ejecución de cuatro pasos para crear una aplicación en Heroku:
 - “heroku create”: Crear una nueva aplicación y un nuevo repositorio. Se puede ejecutar el comando con un parámetro, que sería el nombre de la aplicación/sitio web.
 - Instalar todas las dependencias del proyecto: “bundle install”
 - Agregar todo el código al control de versiones
 - Digitar: “git push heroku master”

Recursos adicionales



- <https://devcenter.heroku.com/>
- <https://devcenter.heroku.com/categories/ruby>