

Laboratorio de Computación IV

Clase 8

Andrés Fortier

¿Consultas?

- Comando: cat y redirect
- Contenidos web: Consulta.
- Herramientas: Sublime.



Hoy

- Comandos: cp y mv.
- Contenidos web: MVC en la web.
- Consulta entrega.

Comandos del día: cp y mv

- cp == copiar archivo(s)

```
$ echo "Hola" > ejemplo.txt
$ mkdir subdirectorio
$ ls -l
total 12
-rw-rw-r-- 1 andres andres      5 abr 12 16:55 ejemplo.txt
drwxrwxr-x 2 andres andres 4096 abr 12 16:55 subdirectorio
```

- cp: origen, destino

```
$ cp ejemplo.txt otro.txt
$ ls -l
$ cat otro.txt
```

Comandos del día: cp y mv

- cp: origen, directorio

```
$ cp ejemplo.txt subdirectorio/  
$ ls -l subdirectorio/  
$ cat subdirectorio/ejemplo.txt
```

```
$ cp *.txt subdirectorio/  
$ ls -l subdirectorio/
```

Comandos del día: cp y mv

- cp -r (Recursivo)

```
$ cd subdirectorio/  
$ mkdir sub2  
$ touch sub2/archivo.txt  
$ cd ..
```

```
.  
├── ejemplo.txt  
├── otro.txt  
└── subdirectorio  
    ├── ejemplo.txt  
    ├── otro.txt  
    └── sub2  
        └── archivo.txt
```

Comandos del día: cp y mv

- cp -r (Recursivo)

```
$ mkdir sub3  
$ cp -r subdirectorio/* sub3/
```

```
.  
├── ejemplo.txt  
├── otro.txt  
├── sub3  
│   ├── ejemplo.txt  
│   ├── otro.txt  
│   └── sub2  
│       └── archivo.txt  
└── subdirectorio  
    ├── ejemplo.txt  
    ├── otro.txt  
    └── sub2  
        └── archivo.txt
```

Comandos del día: cp y mv

- mv == mover o renombrar archivo
- Renombrar archivo

```
$ mv ejemplo.txt renombrado.txt
$ ls -l
total 32
-rw-rw-r-- 1 andres andres      5 abr 12 17:08 otro.txt
-rw-rw-r-- 1 andres andres      5 abr 12 17:08 renombrado.txt
drwxrwxr-x 3 andres andres 4096 abr 12 17:15 sub3
drwxrwxr-x 3 andres andres 4096 abr 12 17:14 subdirectorio
```


Comandos del día: cp y mv

- Mover archivo

```
$ mv renombrado.txt sub3/
$ ls -l
total 28
drwxrwxr-x  4 andres andres 4096 abr 12 17:31 ./
drwxrwxr-x 10 andres andres 4096 abr 12 16:56 ../
-rw-rw-r--  1 andres andres    5 abr 12 17:08 otro.txt
drwxrwxr-x  3 andres andres 4096 abr 12 17:31 sub3/
drwxrwxr-x  3 andres andres 4096 abr 12 17:14 subdirectorio/
$ ls -l sub3/
total 40
-rw-rw-r--  1 andres andres    5 abr 12 17:15 ejemplo.txt
-rw-rw-r--  1 andres andres    5 abr 12 17:15 otro.txt
-rw-rw-r--  1 andres andres    5 abr 12 17:08 renombrado.txt
drwxrwxr-x  2 andres andres 4096 abr 12 17:15 sub2
```

Web MVC



- Vimos cómo servir recursos usando un servidor HTTP

```
$ python -m SimpleHTTPServer 8000
```

- Contenido estático ¿Cómo logramos contenido dinámico?

A lo lejos y hace tiempo: CGI

- *Common Gateway Interface*
- Estándar para
 - Ejecutar un programa /script en el servidor.
 - Retornar el resultado de STDOUT como respuesta.

```
$ mkdir contenidos  
$ cd contenidos  
$ mkdir cgi-bin
```

- Copiar en cgi-bin el archivo `hello.py`

A lo lejos y hace tiempo: CGI

```
#!/usr/bin/python

from datetime import datetime

print 'Content-Type: text/html'
print
print '<html>'
print '<head><title>Hello from Python</title></head>'
print '<body>'
print '<h2>Hello from Python</h2>'
print '<h3>'
print str(datetime.now())
print '</h3>'
print '</body></html>'
```

A lo lejos y hace tiempo: CGI



- En `contenidos` evaluar

```
$ python -m CGIHTTPServer  
Serving HTTP on 0.0.0.0 port 8000 ...
```

- En un web browser ir a
<http://localhost:8000/cgi-bin/hello.py>

A lo lejos y hace tiempo: CGI

```
$ curl -v http://localhost:8000/cgi-bin/hello.py
* Hostname was NOT found in DNS cache
*   Trying 127.0.0.1...
* Connected to localhost (127.0.0.1) port 8000 (#0)
> GET /cgi-bin/hello.py HTTP/1.1
> User-Agent: curl/7.35.0
> Host: localhost:8000
> Accept: */*
>
* HTTP 1.0, assume close after body
< HTTP/1.0 200 Script output follows
< Server: SimpleHTTP/0.6 Python/2.7.6
< Date: Sun, 12 Apr 2015 21:07:06 GMT
< Content-Type: text/html
<
<html>
<head><title>Hello from Python</title></head>
<body>
<h2>Hello from Python</h2>
<h3>
2015-04-12 18:07:06.366699
</h3>
</body></html>
```

CGI en ruby

- Creemos un script CGI en ruby

```
$ cd cgi-bin  
$ gedit hello.rb
```

```
#!/usr/bin/env ruby  
  
puts "HTTP/1.0 200 OK"  
puts "Content-type: text/html\n\n"  
puts "<html><body>This is a test</body></html>"
```

```
$ chmod +x hello.rb  
$ ./hello.rb  
HTTP/1.0 200 OK  
Content-type: text/html  
  
<html><body>This is a test</body></html>
```

CGI en ruby

- En un web browser ir a <http://localhost:8000/cgi-bin/hello.rb>

CGI en ruby

- Creemos otro script CGI en ruby

```
$ cd cgi-bin  
$ gedit otro.rb
```

```
#!/usr/bin/env ruby  
  
require 'cgi'  
cgi = CGI.new('html4')  
  
cgi.out do  
  cgi.html do  
    cgi.body do  
      cgi.h1 {'Hola'} +  
      cgi.p {Time.new()}  
    end  
  end  
end
```

CGI en ruby

```
$ chmod +x otro.rb
```

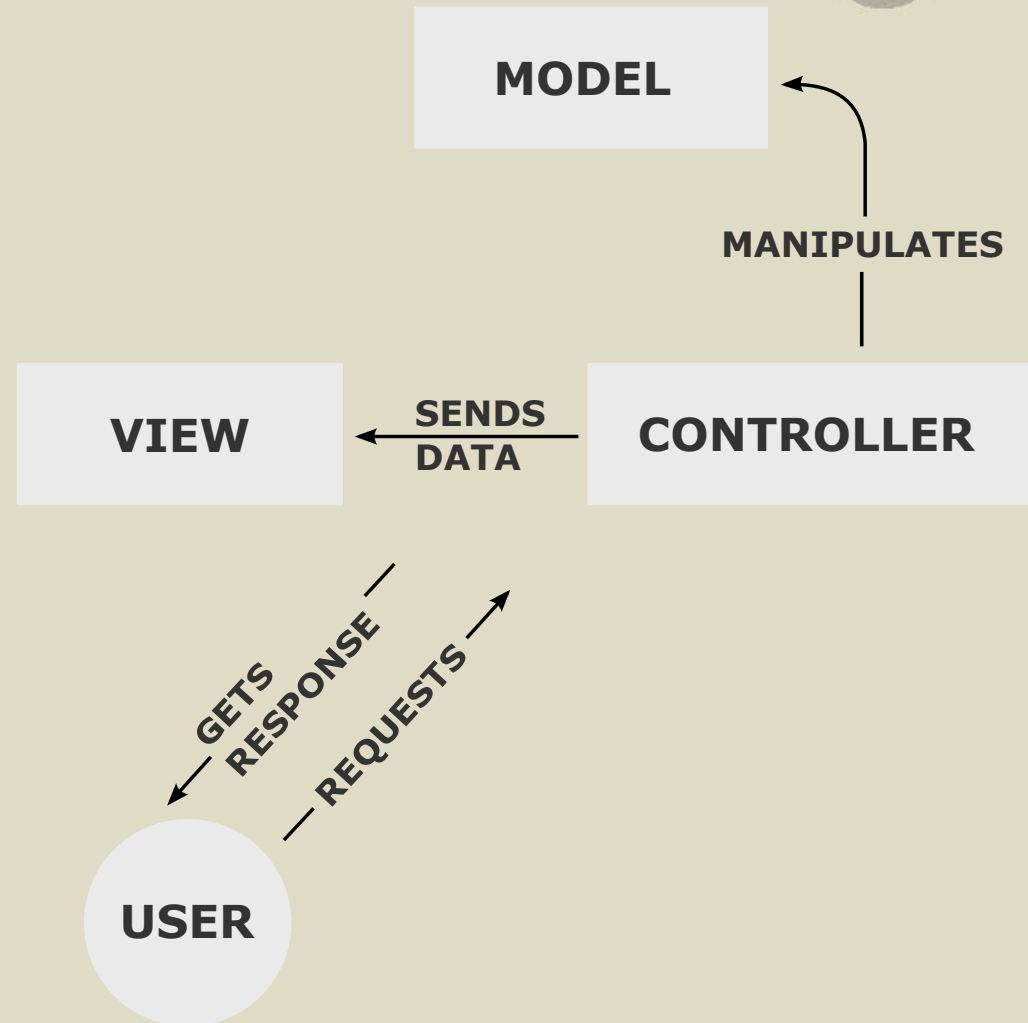
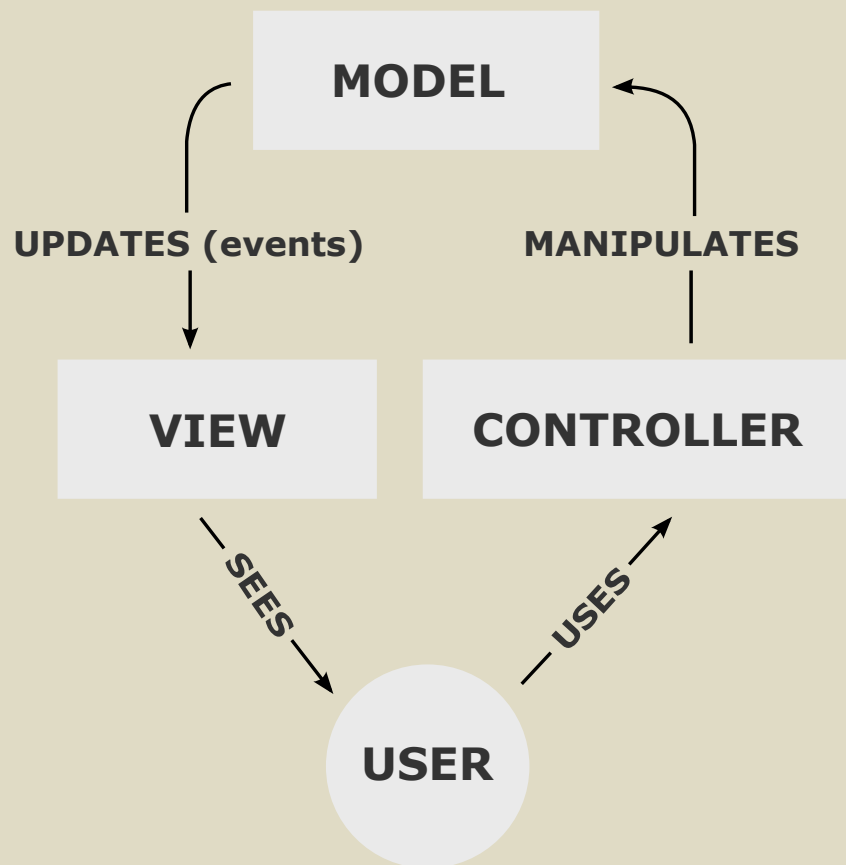
- En un web browser ir a <http://localhost:8000/cgi-bin/otro.rb>

Web MVC




- GCI
 - Un archivo por url, con la lógica de negocios y la generación de la vista.
 - Muy común que no hubiera modelo; SQL + vista.
 - Algunos problemas
 - Incremento en la complejidad.
 - Evolución y Mantenimiento.
 - Testing.

MVC vs “Web MVC”

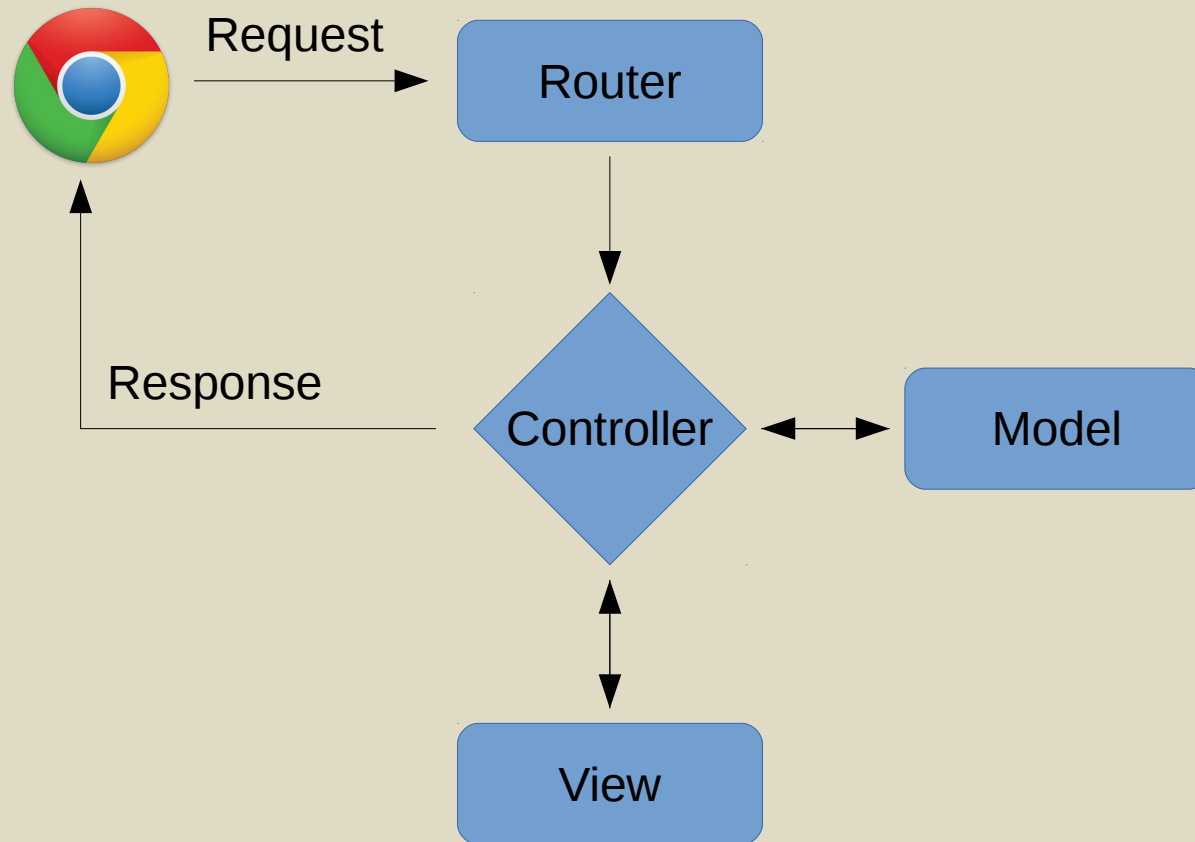


Ruby on Rails / ROR / Rails



- Framework para desarrollar aplicaciones web.
- Model-View-Controller (web).
- *Convention over Configuration.*
- *ActiveRecord* para persistir.
- *Generators.*
- *Templates.*

Rails MVC



Rails MVC

config/routes.rb

```
Rails.application.routes.draw do  
  get 'hello' => 'example#hello'  
end
```

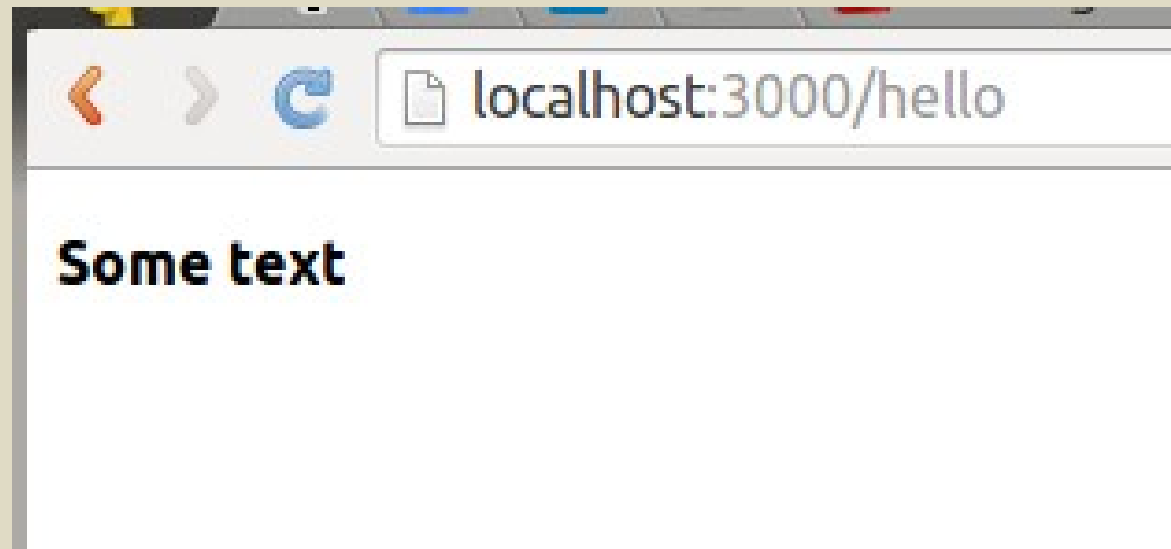
app/controllers/example_controller.rb

```
class ExampleController < ApplicationController  
  def hello  
    @text = "Some text"  
  end  
end
```

app/views/example/hello.erb

```
<p>  
  <strong> <%= @text %> </strong>  
</p>
```

Rails MVC



Tarea para el hogar

- Crear una aplicación Rails de ejemplo

```
$ mkdir ejemplorails

$ cd ejemplorails/

$ rvm use --create ruby-2.0.0-p598@ejemplorails
ruby-2.0.0-p598 - #gemset created
/home/andres/.rvm/gems/ruby-2.0.0-p598@ejemplorails
ruby-2.0.0-p598 - #generating ejemplorails
wrappers.....
Using /home/andres/.rvm/gems/ruby-2.0.0-p598 with gemset
ejemplorails

$ sqlite3 --version
3.8.2 2013-12-06 14:53:30
27392118af4c38c5203a04b8013e1afdb1cebd0d

$ gem install rails -v 4.1.8

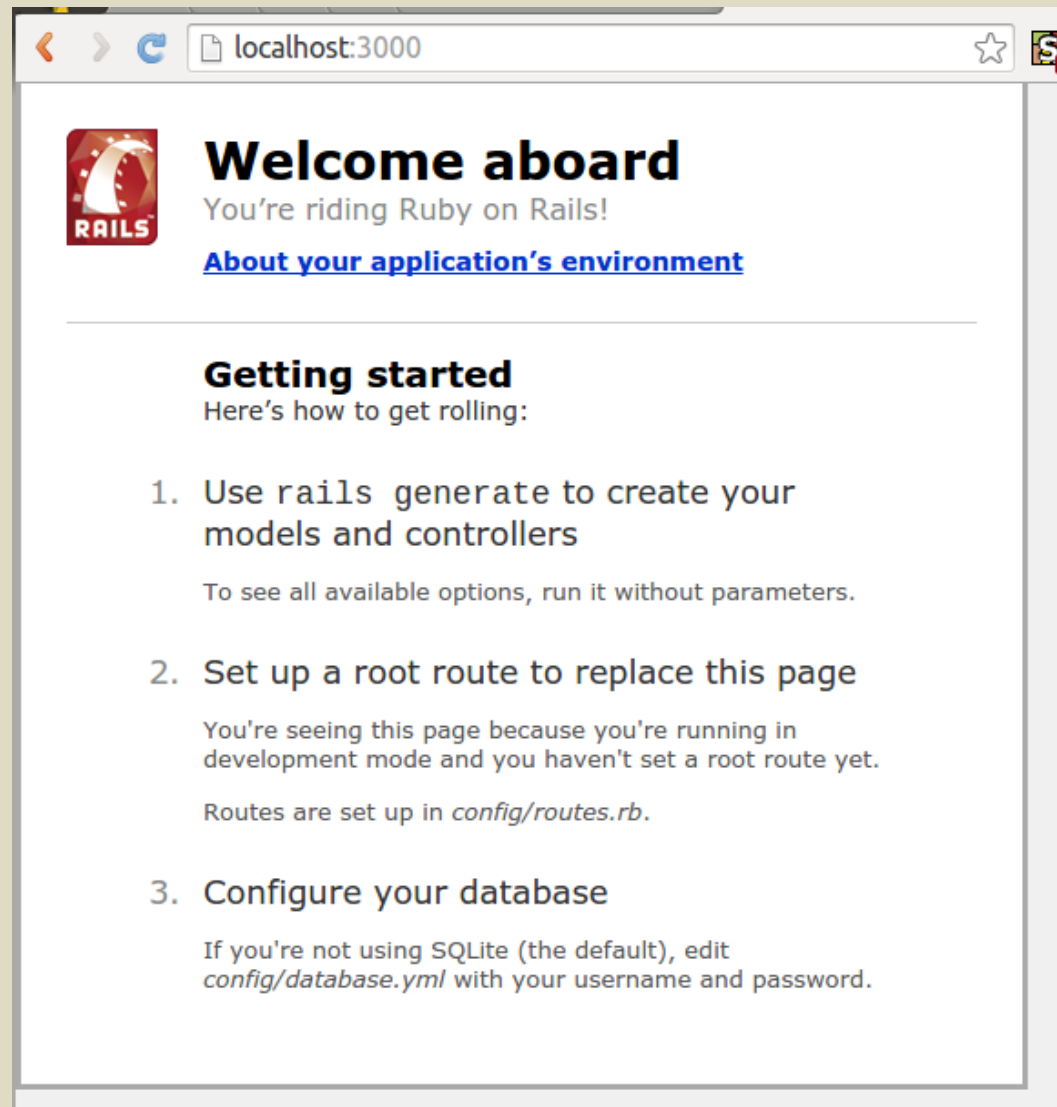
$ rails new . -T
```

Tarea para el hogar

- Crear una aplicación Rails de ejemplo

```
$ bin/rails server
=> Booting WEBrick
=> Rails 4.1.8 application starting in development on
http://0.0.0.0:3000
=> Run `rails server -h` for more startup options
=> Notice: server is listening on all interfaces (0.0.0.0).
Consider using 127.0.0.1 (--binding option)
=> Ctrl-C to shutdown server
```

Tarea para el hogar



Tarea para el hogar

- http://guides.rubyonrails.org/getting_started.html