

## Lezione 4: Introduzione a Rails

Matteo Vaccari

<http://matteo.vaccari.name/>  
matteo.vaccari@uninsubria.it

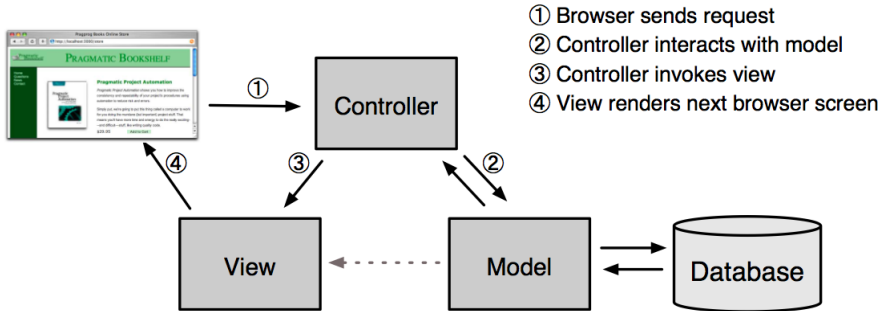


(cc) Matteo Vaccari. Published in Italy.  
Attribution – Non commercial – Share alike 2.5

# Il nostro vecchio amico CGI-quotes

```
#!/usr/bin/env ruby
puts "Content-Type: text/html\r\n"
puts "\r\n"
puts "<html>" # view
puts "<head><title>Quote of the day</title></head>" # view
puts "<body>" # view
puts "<h1>Quote of the day!!</h1>" # view
require 'cgi'
cgi = CGI.new
param = cgi["q"] # interaction
quotes = File.new("/tmp/quotes.txt").read.split("%") # persistency
if param.length > 0 # interaction
  quotes = quotes.select { |quote| quote.include?(param) } # business logic
end
puts "<pre>" + quotes[rand(quotes.size)] + "</pre>" # view
puts "</body>" # view
puts "</html>" # view
```

# II pattern model-view-controller



## Perché Rails?

PHP: quick and dirty

Java: clean but sloooooow

Ruby on Rails: clean and quick!

## Less code is better

- ▶ Succinctness is power

— Paul Graham

- ▶ DRY—Don't Repeat Yourself

Every piece of knowledge must have a single, unambiguous, authoritative representation within a system.

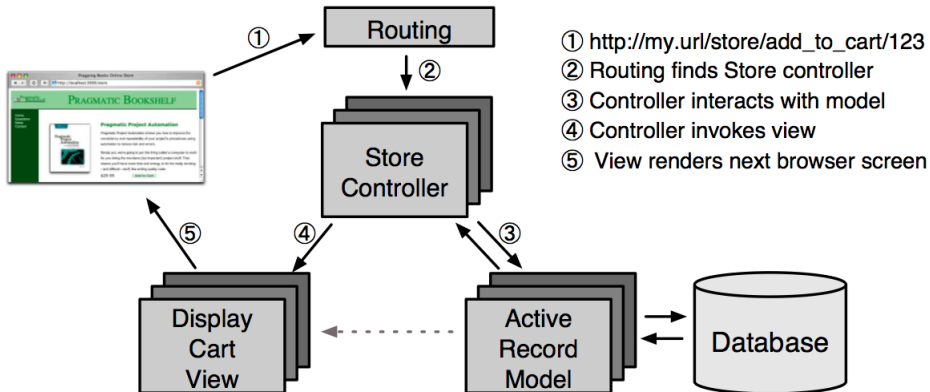
— The Pragmatic Programmers

- ▶ Rails is a framework for writing applications

with joy and less code than most frameworks spend doing XML

— David Heinemeier Hansson

# II pattern MVC in Rails



# Hello world!

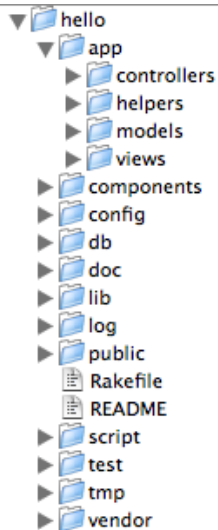
Come ottenere un hello, world?

- ▶ rails hello
- ▶ cd hello
- ▶ script/server
- ▶ script/generate controller Hello
- ▶ def index; render :text => "Hello, world!"; end

# Hello, world! (i)

```
$ rails hello
  create
  create  app/controllers
  create  app/helpers
  create  app/models
  create  app/views/layouts
  create  config/environments
  create  components
      :      :
  create  log/test.log
$
```

# Directory



Hello, world! (ii)

```
$ script/server
```

```
=> Booting WEBrick...
```

```
=> Rails application started on http://0.0.0.0:3000
```

```
=> Ctrl-C to shutdown server; call with -help for options
```

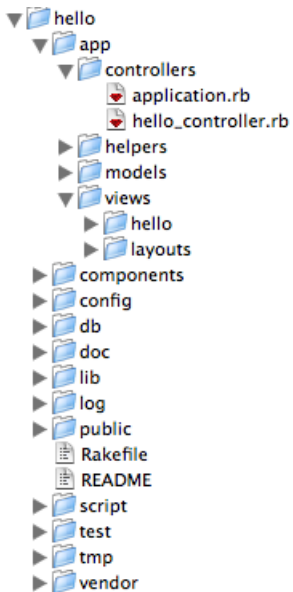
```
[2006-10-17 16:34:44] INFO WEBrick 1.3.1
```

```
[2006-10-17 16:34:44] INFO ruby 1.8.4 (2005-12-24) [i686-darwin8.6.
```

```
[2006-10-17 16:34:44] INFO WEBrick::HTTPServer#start: pid=334 port=
```

# Hello, world! (iii)

```
$ script/generate controller Hello
  exists  app/controllers/
  exists  app/helpers/
  create  app/views/hello
  exists  test/functional/
  create  app/controllers/hello_controller.rb
  create  test/functional/hello_controller_test.rb
  create  app/helpers/hello_helper.rb
$
```



# Hello, world! (iv)

app/controllers/hello\_controller.rb

---

```
class HelloController < ApplicationController
  def index
    render :text => "Hello, world!"
  end
end
```

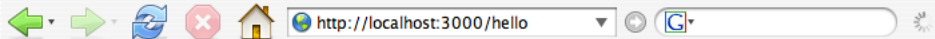
app/views/hello/index.rhtml

---

```
Hello from the view!
```



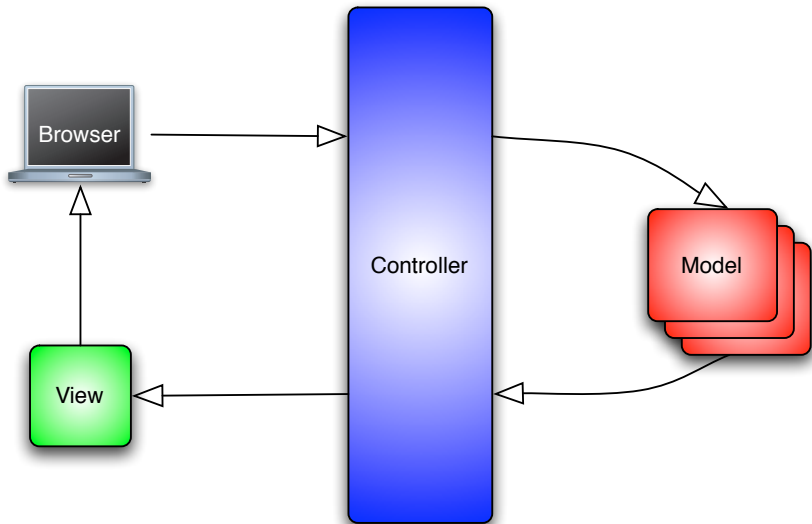
Mozilla Firefox



Hello from the view!

Done

# L'architettura di Rails



## II pattern MVC in Rails

models: Active Record

views: Action View

controllers: Action Controller

## Esercizio: un server di citazioni

Realizzare una pagina che mostri una citazione scelta a caso da un “database” di motti e citazioni.

Il “database” è un semplice file che contiene le citazioni separate dalla stringa “%”

# Un servizio motti & citazioni

Iterazione 0

- ▶ Mostriamo sempre la stessa citazione

# Il modello

app/models/quote.rb

---

```
class Quote
  attr_reader :text

  def initialize(text)
    @text = text
  end
end
```

## Il controller

app/controllers/quotes\_controller.rb

---

```
class QuotesController < ApplicationController
  def index
    @quote = Quote.new("Quidquid latine dictum sit, alte videtur")
  end
end
```

ogni metodo pubblico corrisponde a un'azione:

<http://myserver/quotes/index>

## La view

app/views/quotes/index.rhtml

```
<html>
  <head>
    <title>Motti...</title>
  </head>
  <body>
    <h1>Motti celebri e non</h1>
    <div class='quotation'>
      <h2>Il motto:</h2>
      <pre>
        <%= @quote.text %>
      </pre>
    </div>
  </body>
</html>
```

app/controllers/quotes\_controller.rb

```
class QuotesController < ApplicationController
  def index
    @quote = Quote.new("Quidquid l...")
  end
end
```

Le instance variable del controller sono **iniettate** nella view

# Server citazioni

## Iterazione 1

- ▶ scegliamo una citazione a caso da un file

## Implementiamo l'accesso al file

app/models/quote.rb/hrule

```
class Quote
  attr_reader :text

  def initialize(text)
    @text = text
  end

  def Quote.random
    a = all_texts
    Quote.new a[rand(a.size)]
  end

private
  def Quote.all_texts
    File.new("public/quotes.txt").read.split("%")
  end
end
```

## Il controller deve pescare una citazione a caso

app/controllers/quotes\_controller.rb

---

```
class QuotesController < ApplicationController
  def index
    @quote = Quote.random
  end
end
```

## Iterazione 2

Aggiungiamo alla pagina un campo di testo che permetta di cercare una citazione a caso che contenga una certa parola.

## Iterazione 3

Esercizio:

Aggiungiamo una pagina che contiene 10 citazioni, con due link per vedere la successiva e la precedente paginata di 10 citazioni

Task:

- ▶ aggiungi un metodo al modello
- ▶ aggiungi un'azione al controller
- ▶ crea la nuova view
- ▶ duplicazione nelle views?