

Agile web development with Ruby and Rails

Mr. Charles Woo

Marketing Officer of *Gownfull IME*

Homepage: <http://www.gownfull.com>

Email: charles@gownfull.com

What is Ruby

- It is not a stone! (Of course, you're not attending a jewellery design workshop)
- Open-source programming language by Yukihiro Matsumoto (Japanese guy)
- It is object-oriented
- Numerous libraries are available (Database integration, XML,...)

What will be covered

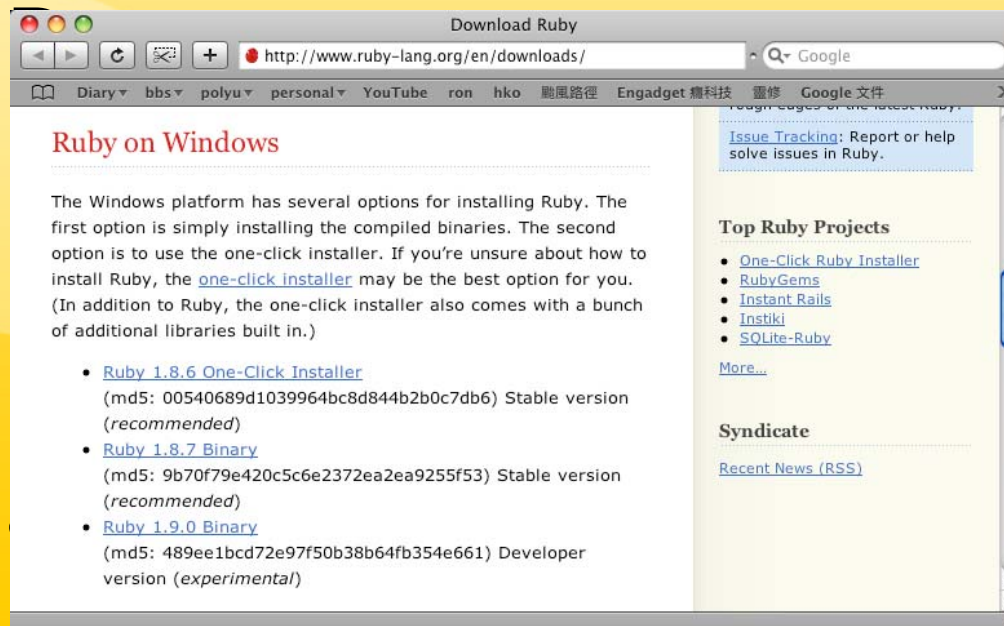
- From Installation to desktop program
- Basic Ruby language
- Basic concept in Relational Database
- OO design methodologies
- Ruby on Rails!! – web application development

1st Objective

- Upon completion of this activity, you will be able to:
 - 1. Install the Ruby compiler
 - 2. Write, compile and execute a Ruby program;
 - 3. Perform simple arithmetic operations

Ruby installation

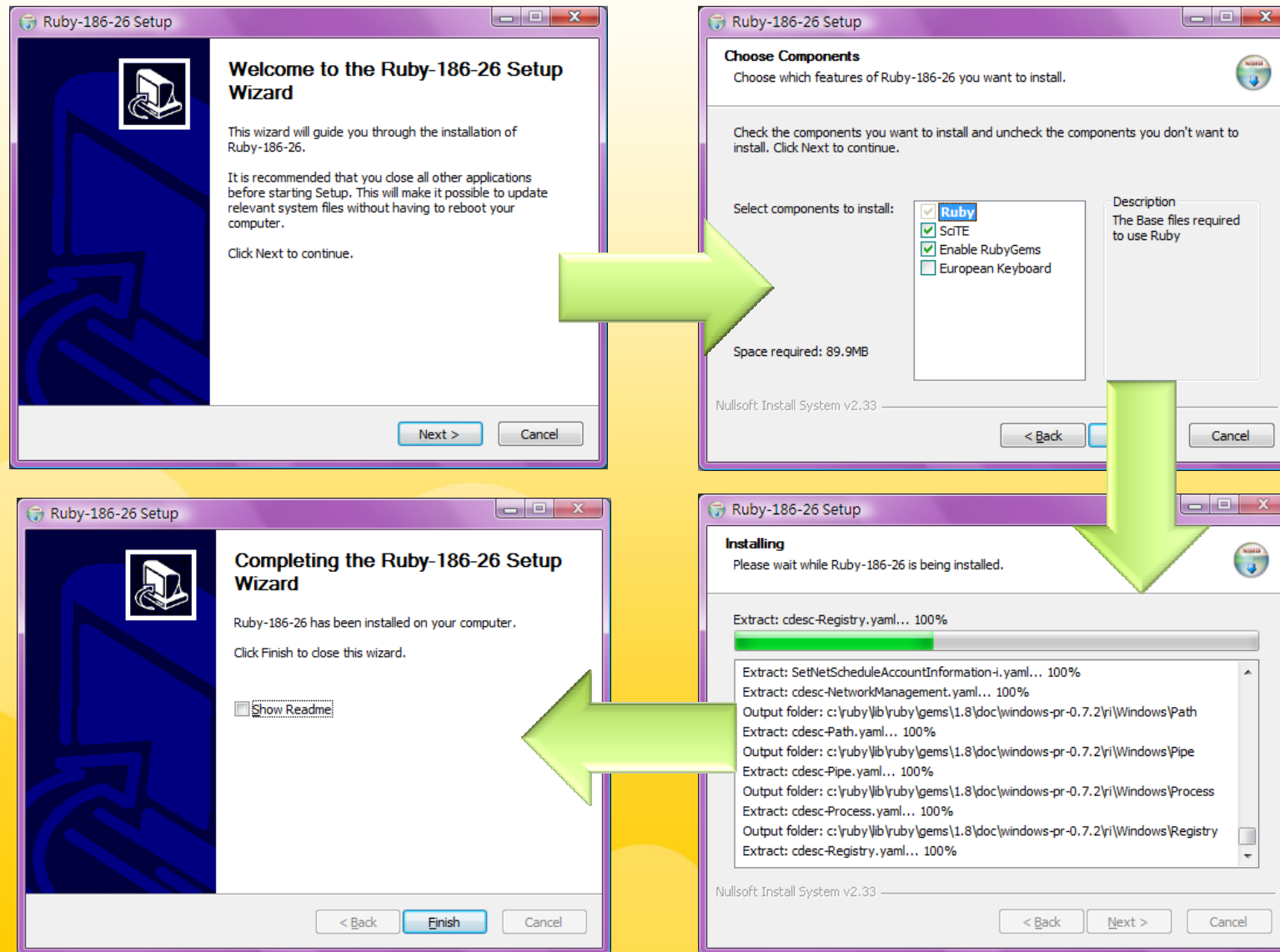
- Ruby compiler can be downloaded on <http://www.ruby-lang.org/en/downloads/>
- You can use Ruby compiler in COMP labs by calling “nalwin32” from start menu →



However, I

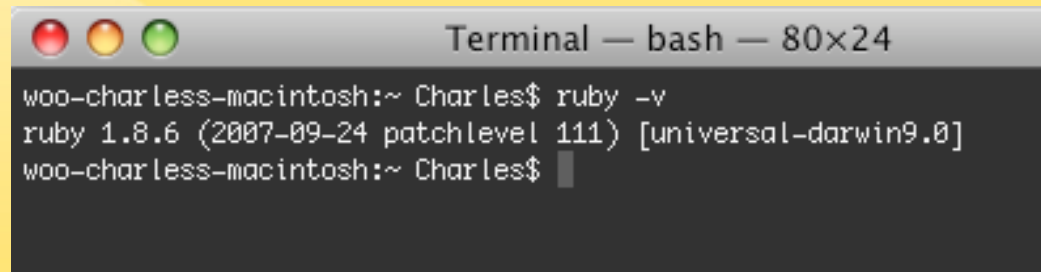
using Mac

Installation of Ruby



Ruby installation

- After installation, check whether the Ruby is successfully installed
- Go to command prompt, enter `ruby -v`

A screenshot of a macOS Terminal window. The title bar reads "Terminal — bash — 80x24". The window has three colored window control buttons (red, yellow, green) on the left. The terminal text shows a user named Charles at a machine named woo-charless-macintosh. The user enters the command `ruby -v`, and the terminal outputs `ruby 1.8.6 (2007-09-24 patchlevel 111) [universal-darwin9.0]`. The prompt then returns to `woo-charless-macintosh:~ Charles$` with a cursor.

```
Terminal — bash — 80x24
woo-charless-macintosh:~ Charles$ ruby -v
ruby 1.8.6 (2007-09-24 patchlevel 111) [universal-darwin9.0]
woo-charless-macintosh:~ Charles$
```

Hello World

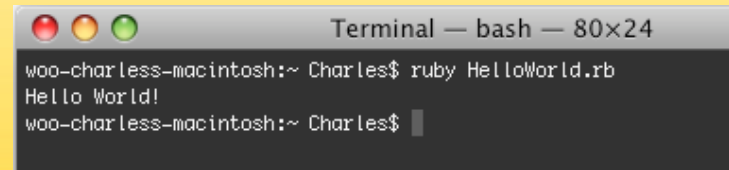
- As a beginner, the 1st small program that you are going to write is “Hello World”
- Source code:

```
puts "Hello World!"
```

- Save the file as HelloWorld.rb

Compile and execute

- Use the command
- “ruby HelloWorld.rb”
- to compile and execute the application

A screenshot of a macOS Terminal window. The title bar reads "Terminal — bash — 80x24". The window has three colored window control buttons (red, yellow, green) on the left. The terminal text shows the user "Charles" at the prompt "woo-charless-macintosh:~ Charles\$". They have entered the command "ruby HelloWorld.rb", which has been executed, resulting in the output "Hello World!". The prompt now shows "woo-charless-macintosh:~ Charles\$" with a cursor.

```
Terminal — bash — 80x24
woo-charless-macintosh:~ Charles$ ruby HelloWorld.rb
Hello World!
woo-charless-macintosh:~ Charles$
```

Getting input from YOU

```
puts "What is your name?"  
name = gets.chomp    #without line break  
puts "How old are you?"  
age = gets  
  
puts 'Hello, ' + name.to_s + ', you are ' +  
age.to_s + ' years old.'
```

Additional exercise

- Remove `chomp` in line 2
- Remove `to_s`

Play with strings

```
# play with strings
str = "Ruby programming is fun."

puts 'What does length do? It returns: ' +
str.length.to_s
puts 'How about slice(5,11)?' +
str.slice(5,11)
puts str.count('programming').to_s
puts str.upcase
puts str.downcase
puts str
```

Task

- Write a Ruby program that accepts a plaintext (i.e. some words inputted by the user) through keyboard. Encrypt it using the Caesar's cipher method as described below
- $A \rightarrow B, B \rightarrow C, C \rightarrow D \dots z \rightarrow \}$
- example

Better start...program structure

```
print _____  
plaintext = _____  
_____  
  
while _____  
#####  plaintext[count]  
    ciphertext = _____ + _____  
    count = count+1  
end  
  
puts ciphertext
```

Playing with Arrays

```
week = ['Sunday', 'Monday', 'Tuesday', 'Wednesday',  
        'Thursday', 'Friday', 'Saturday']
```

```
task = ['Sleeping', 'Resting', 'Studying', 'Cleaning',  
        'Reading', 'Cooking', 'Sleeping']
```

```
count=0
```

```
week.each do |d|  
    puts d + ' is good for ' + task[count]  
    count=count+1  
end
```

Object-oriented concepts in Ruby

- Upon completion of this activity, you will be able to:
- 1. Define a class with variables and methods
- 2. Work with instance variables and local variables;
- 3. Define a subclass
- 4. Write a simple program with OO.

Understanding a Class

```
class Greetings
  @name = ""
  def initialize(name)
    @name = name
  end
  def greet(mesg)
    puts "Hello! " + @name + ". " + mesg
  end

  g1=Greetings.new("Charles")
  g1.greet("Good luck")
  g2=Greetings.new("Kathy")
  g2.greet("Today is a sunny day!")
end
```

Questions

1. What does “@” stand for?
2. What does the method “initialize” do?
3. How to distinguish between instance variables and local variables?

Name → local variable

@name → instance variable (new)

@ @name → Global variable

In-class exercise

You are required to write a class “Pet”.

As in the real world, a pet contains several attributes:

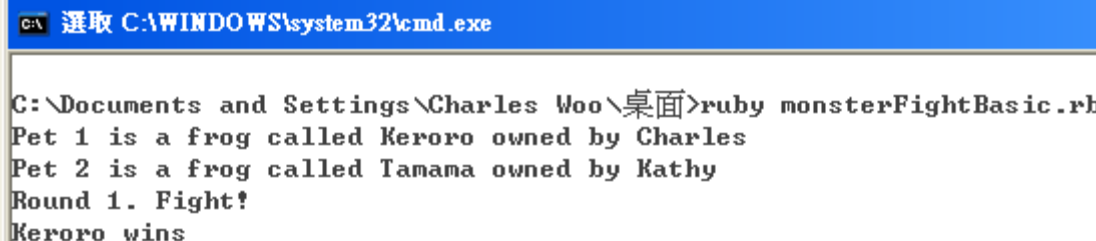
- 1.name: the name of pet (say, Donald, Peter etc.)
- 2.owner: the name of pet owner (say, Donald, Peter etc.)
- 3.weight: the weight of a pet (say, 100, 1000 etc.)
- 4.type: obviously, what that pet is (say, leopard, lizard, tortoise etc.)

Task

The specification of the program is summarized below:

1. Four instance variables as described above should be declared in the Pet class. (Hint: `@name`, `@owner`, `@weight`, `@type`)
2. The method `initialize` should accept four parameters (name, owner, weight, type) and initialize the instance variables.
3. The method `to_s` should be written to return the type of the pet.
4. You should also include the methods `getName`, `getOwner`, `getWeight` to return the values stored in the instance variables of the pet, just similar to accessor (also known as getter) in Java.

5. The `fight` method should accept two parameters. The pet wins if it is the first parameter. The message on the screen capture.



```
選取 C:\WINDOWS\system32\cmd.exe
C:\Documents and Settings\Charles Woo\桌面>ruby monsterFightBasic.rb
Pet 1 is a frog called Keroro owned by Charles
Pet 2 is a frog called Tamama owned by Kathy
Round 1. Fight!
Keroro wins
```

```
class Pet
#define the instance
variables
... ..
#initialize the object
def initialize(*args)
@name,@owner,@weight,@t
ype = args
end

#return the pet type
def to_s
@type
end
#return the pet name
def getName
...
end
```

```
#return the pet owner's name
def getOwner
...
end

#return the pet's weight
def getWeight
...
end

#fight! a pet wins if it is
heavier than its opponent
def fight(...)
if ...
...
else
...
end
end
```

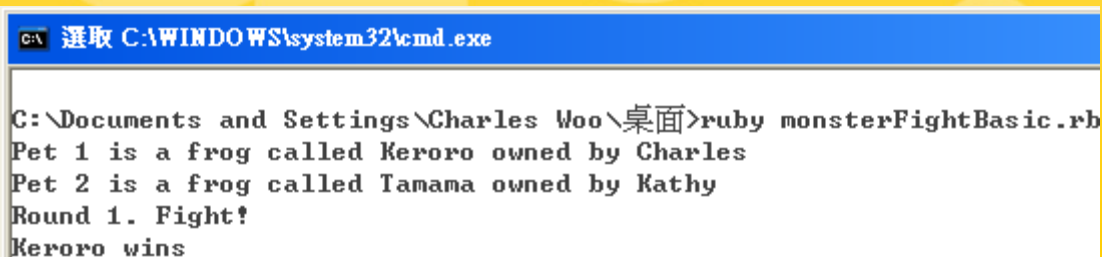
```
#test case goes here.
```

```
p1=Pet.new('Keroro','Charles',30,'frog')  
puts "Pet 1 is a " + p1.to_s + " called " + p1.getName  
+ " owned by " + p1.getOwner + " \n"
```

```
p2=Pet.new('Tamama','Kathy',20, 'frog')  
puts "Pet 2 is a " + p2.to_s + " called " + p2.getName  
+ " owned by " + p2.getOwner + " \n"
```

```
puts "Round 1. Fight!"  
puts p1.fight(p2)
```

```
end
```



The screenshot shows a Windows command prompt window with the title bar "C:\ 選択 C:\WINDOWS\system32\cmd.exe". The command prompt displays the following output:

```
C:\Documents and Settings\Charles Woo\桌面>ruby monsterFightBasic.rb  
Pet 1 is a frog called Keroro owned by Charles  
Pet 2 is a frog called Tamama owned by Kathy  
Round 1. Fight!  
Keroro wins
```

Polymorphism and inheritance

(變態與繼承)

- You know what is 'Object' from the previous task.
- We are going to study the relationship between Shape and Circle. (Refer to handout)

Questions

1. What does “<” stand for?
2. What does “super” mean?
3. How can you see that the Circle class extends the properties of its parent class Shape?

Graphical User Interface using Ruby

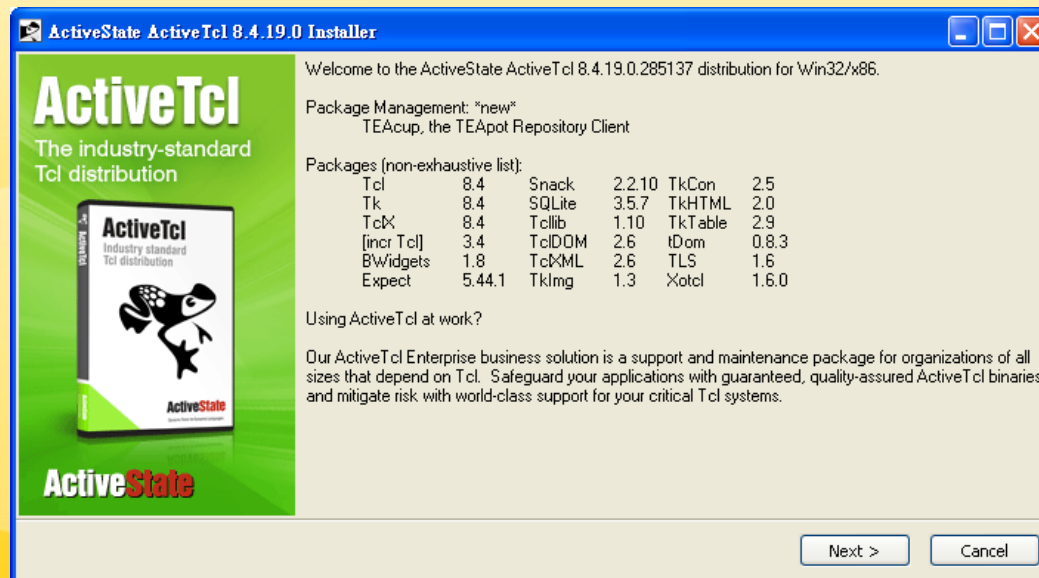
- Upon completion of this activity, you will be able to:
- 1. Setting up the environment for Ruby/ Tk;
- 2. Work with Buttons and Frames;
- 3. Work with Message Boxes;
- 4. Write a simple GUI Application with Ruby/ Tk.

Graphical User Interface using Ruby

- 1. Normally Ruby GUI applications using Ruby/Tk cannot be run successfully even though the Ruby compiler is installed successfully
- 2. A free tool, known as “Active Tcl” can be installed to run Ruby/ Tk applications. You may download “Active Tcl” under “Language Downloads” **for free on** <http://www.activestate.com>

Installation of ActiveTcl

- Few-click installer
- Auto config. with Ruby interpreter/compiler

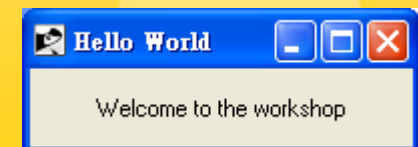


First GUI

```
require "tk"
root = TkRoot.new() { title "Hello World" }
str = "Welcome to the workshop"

lab = TkLabel.new(root) do
  text str
  pack("padx" => 15, "pady" => 10, "side" => "bottom")
end

Tk.mainloop
```



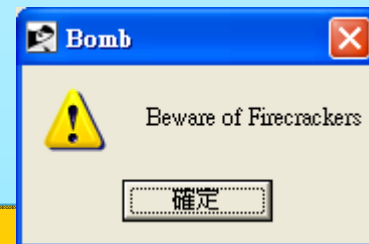
Play with button and popup

```
require "tk"

root = TkRoot.new() { title "Happy New Year" }

btnGo = TkButton.new(root) do
  text "Click me!"
  command proc {
    puts "The button is clicked!"
    messageBox 'type' => 'ok', 'icon' => 'warning',
    'message' => 'Beware of Firecrackers', 'title' =>
    'Bomb'
  }
  pack("padx" => 80, "pady" => 10, "side" => "top",
  "anchor" => "c")
end

Tk.mainloop
```

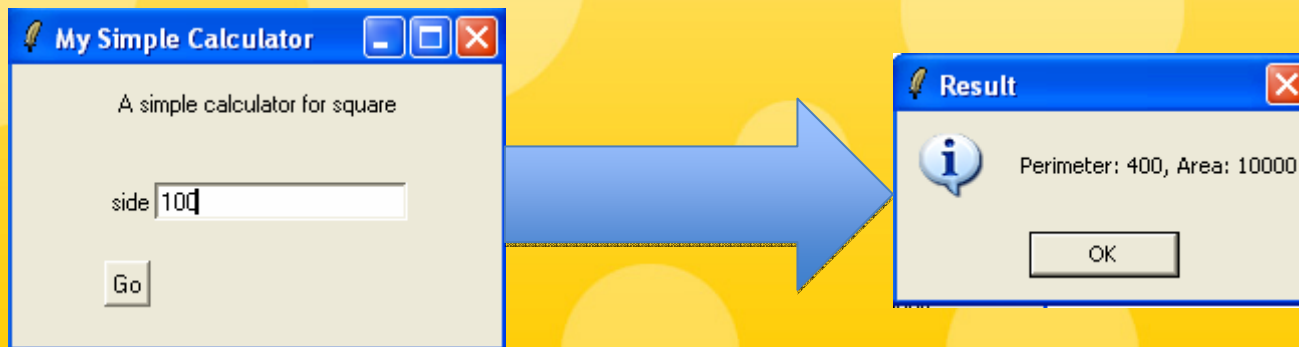


Play with Frame

Refer to the handout

Task

- This is a combined version of previous GUI examples, plus a few new features.
- Your task is to create a simple “SquareCalculator”. It takes the length of a side of a square, and outputs its perimeter and area as shown below.



Task (Con't)

- To train you as a good self-learner, we intentionally not to give any briefing on using textfields. Instead, you are requested to search the information by yourself on the web.
- To give you a better start, the following code segment is provided to you as a kick-off to this task.

Task (Con't)

```
entSide = TkEntry.new(frc) do
  textvariable var_side
  width 20
  pack("side" => "right", "anchor" => "e")
end
```

- Apart from the code segment above, you should also find the statement `var_side.value`, that returns the value (of course, you should cast it to integer by `to_i`) of the textfield, very useful for the calculation.

What is Ruby on Rails?

- Ruby-based Framework
 - Client-Server Web Application
- Faster in Application Development
 - 5-10 times faster than Java (Servlet)
- Active Record: → No need to know SQL
- AJAX Rails: → No need to know XML and JavaScript

What you need for Ruby on Rails?

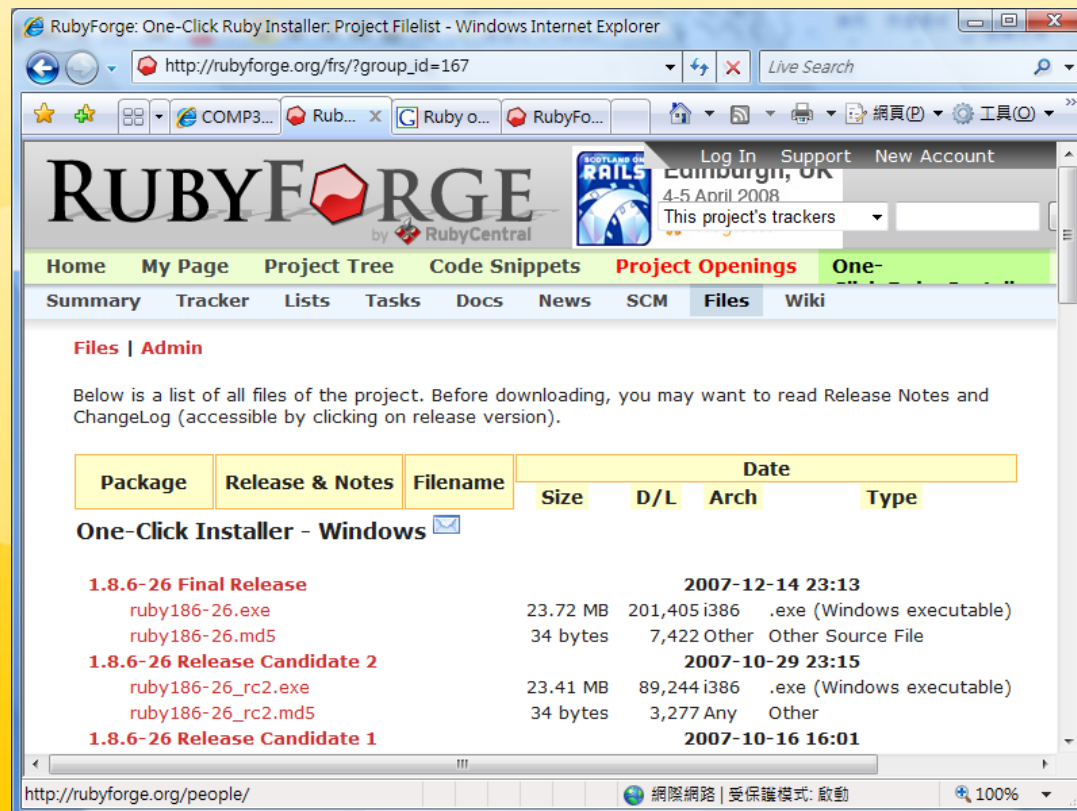
- Ruby (Of course)
- MySQL (Database – Active Record)
- Rails

Installation of RoR

- 1) Ruby and RubyGems
<http://rubyinstaller.rubyforge.org/wiki/wiki.p>
!
- 2) Ruby on Rails
<http://www.rubyonrails.com>
- 3) MySQL
<http://www.mysql.com>
- I will use Appserv instead of MySQL
<http://www.appservnetwork.com>

Install Rails

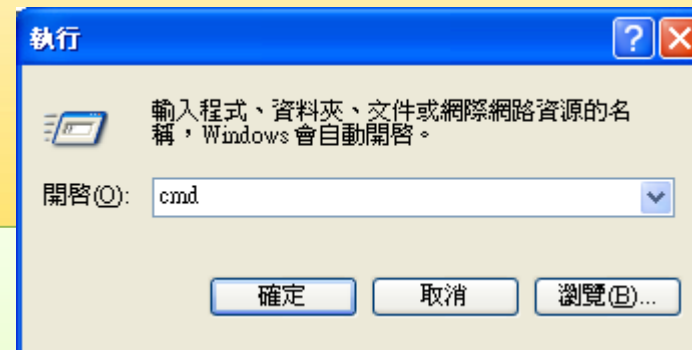
- Windows: one click installer
- Download the latest Release and install



Appendix – InstallationRoR

- Windows: Use RubyGems to install Rails
- Open cmd (command prompt)

```
gem install activesupport -v 1.4.4
gem install activerecord -v 1.15.6
gem install actionpack -v 1.13.6
gem install actionmailer -v 1.3.6
gem install actionwebservice -v 1.2.6
gem install rake -v 0.8.1
gem install rails -v 1.2.6
```



Appendix – InstallationRoR

- Linux and Unix
 - Download Ruby source and build it

```
tar xzf ruby-1.8.6-26.tar.gz
cd ruby-1.8.6-26
./configure
make
make test
sudo make install
```

- Download Ruby gems and build it

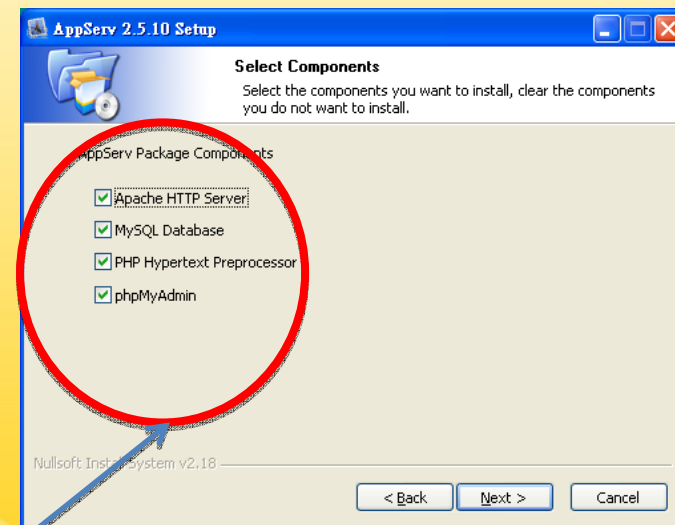
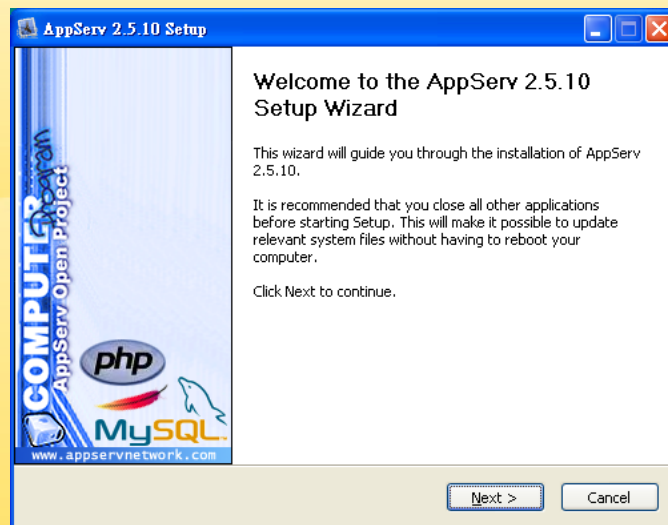
```
tar xzf rubygems-1.2.6.tar.gz
cd rubygems-1.2.6
sudo ruby setup.rb
```

- Use RubyGems to install Rails

```
sudo gem install rails --include-dependencies
```

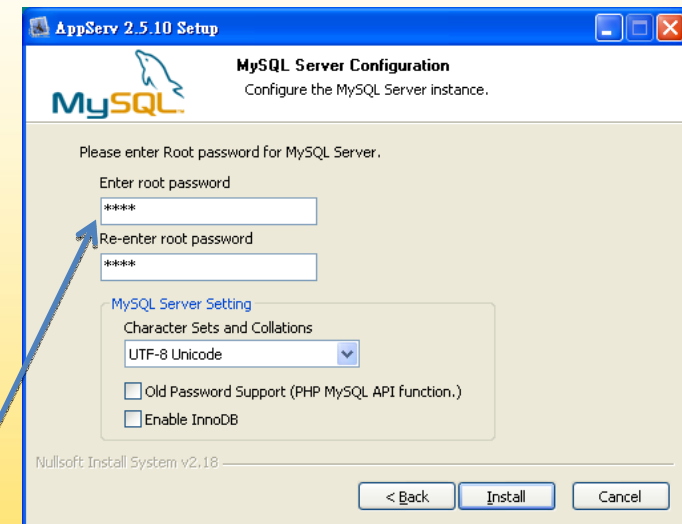
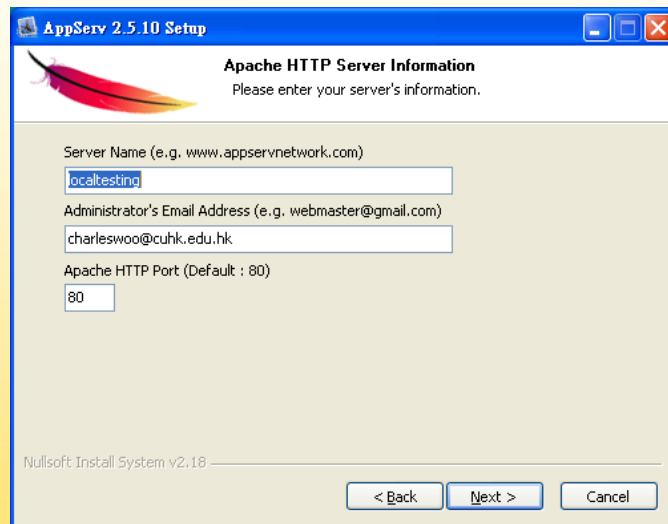
Install Appserv

- Appserv includes Apache server, PHP, MySQL, and its tools → PHPMYAdmin



Select all

Install Appserv

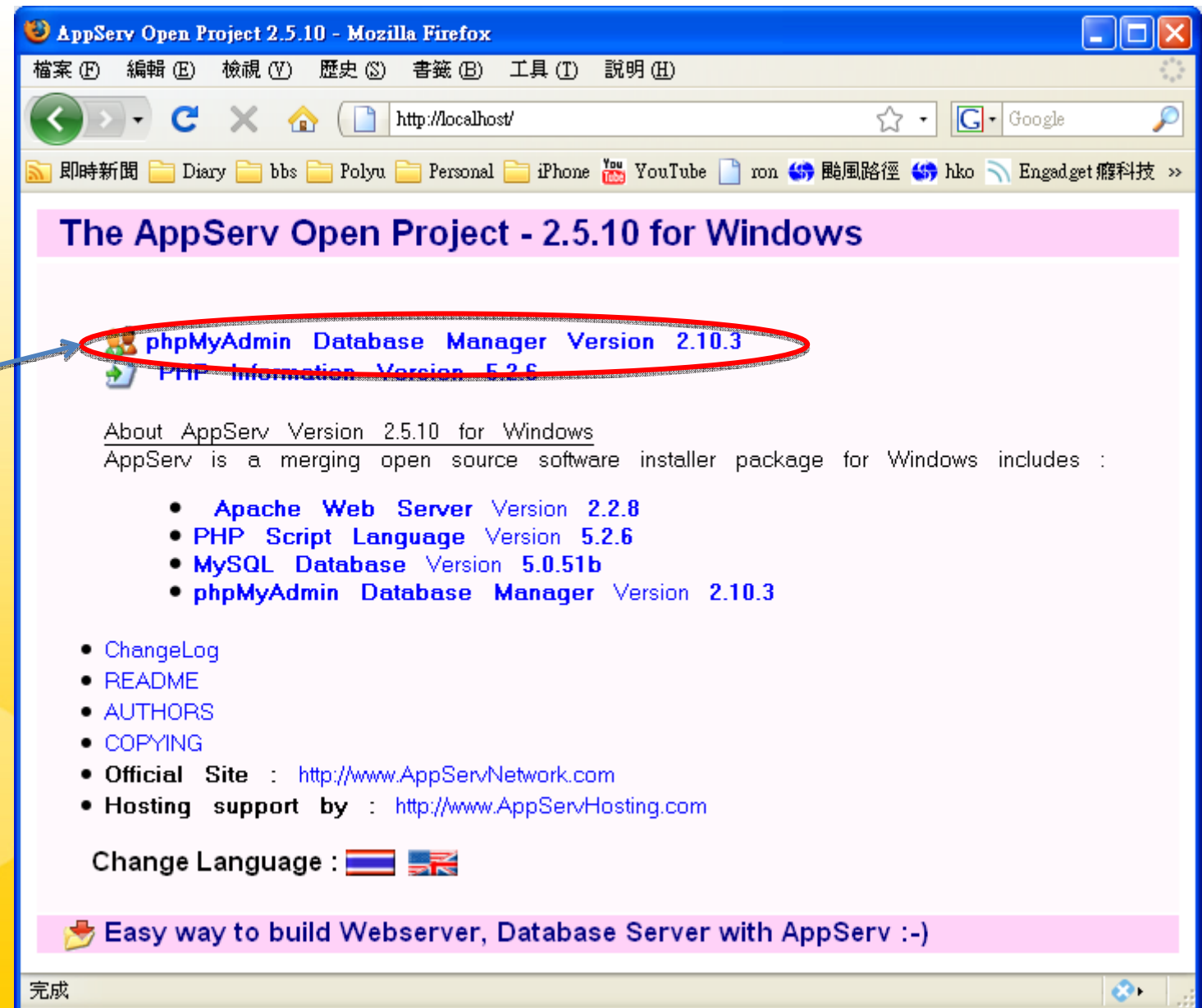


enter the root password



Unblock it

We need
this later

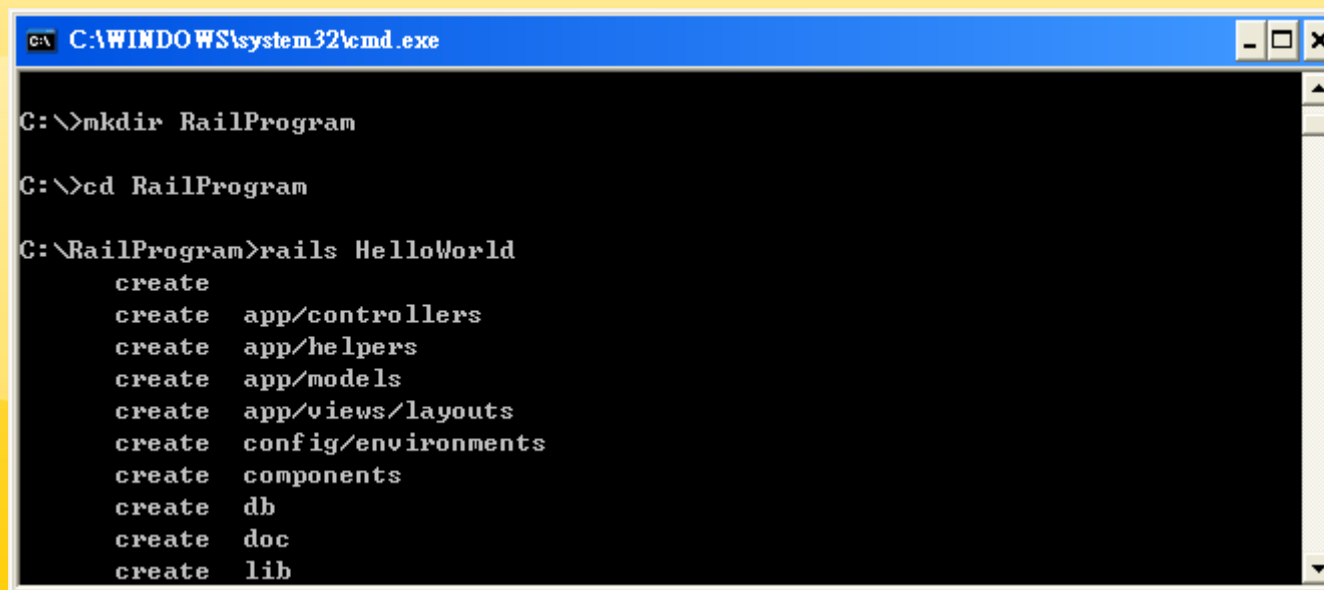


Hello and Welcome to Rails

- From this section, you will learn how to:
 - Use Rails to create an application framework
 - Use the WEBrick web server
 - Generate and add action to controller
 - Create a view template
 - Add 2nd action to a web application
 - Link to actions from views

Creating the 1st Application

- Go to Command Prompt, and select one folder for the project workspace.
- Creating Project command `rails ProjectName`



```
C:\>C:\WINDOWS\system32\cmd.exe

C:\>mkdir RailProgram

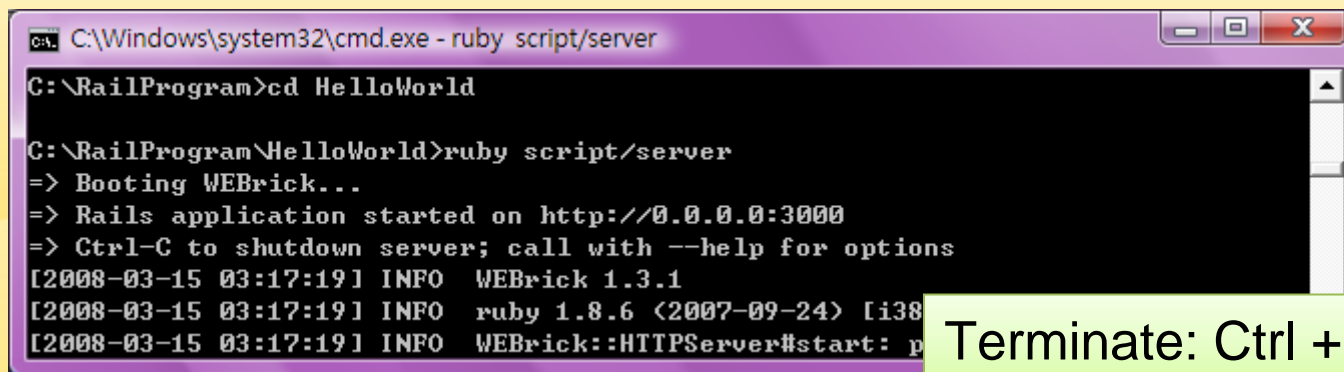
C:\>cd RailProgram

C:\>RailProgram>rails HelloWorld
create
create  app/controllers
create  app/helpers
create  app/models
create  app/views/layouts
create  config/environments
create  components
create  db
create  doc
create  lib
```

Put Ruby on the Web - WEBrick

- Running Application: start up a server

```
ruby script/server
```



The screenshot shows a Windows command prompt window titled "C:\Windows\system32\cmd.exe - ruby script/server". The command prompt shows the following output:

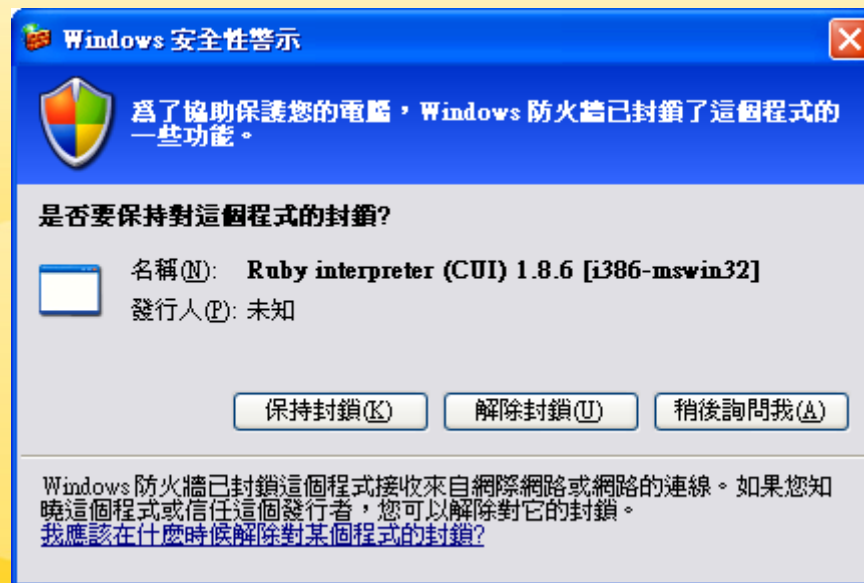
```
C:\RailProgram>cd HelloWorld  
C:\RailProgram\HelloWorld>ruby script/server  
=> Booting WEBrick...  
=> Rails application started on http://0.0.0.0:3000  
=> Ctrl-C to shutdown server; call with --help for options  
[2008-03-15 03:17:19] INFO WEBrick 1.3.1  
[2008-03-15 03:17:19] INFO ruby 1.8.6 (2007-09-24) [i386]  
[2008-03-15 03:17:19] INFO WEBrick::HTTPServer#start: p
```

Terminate: Ctrl + C

- This starts the WEBrick server on port 3000
- Access at <http://localhost:3000>

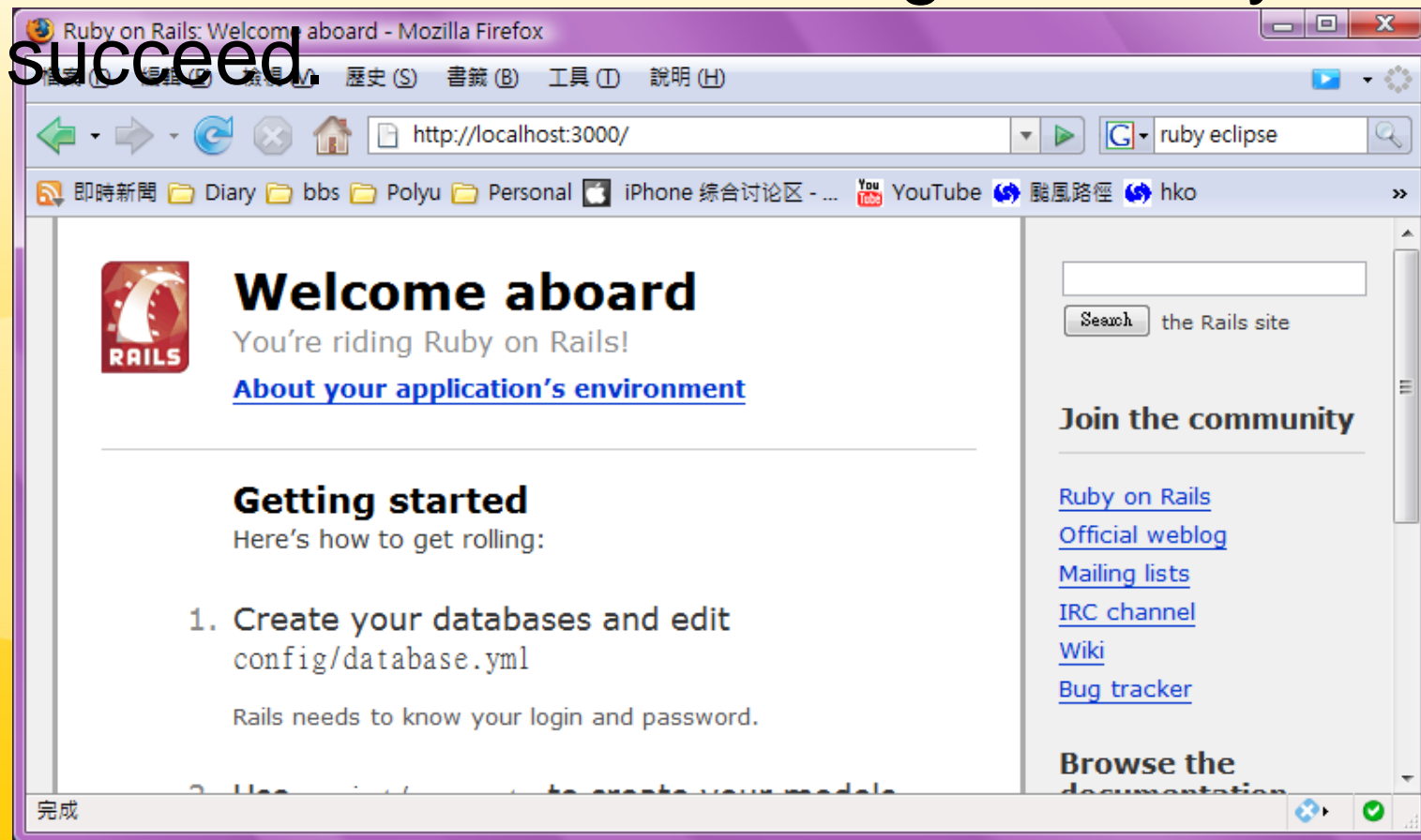
Windows firewall protection

- For testing, you can omit the protection by clicking “Allow”



Put Ruby on the Web - WEBrick

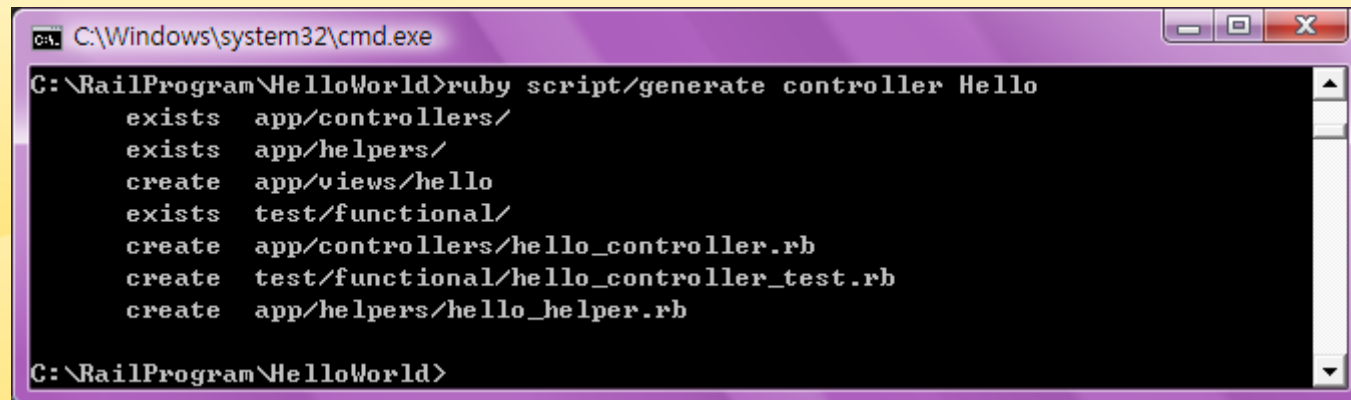
- You will have the following result if you succeed.



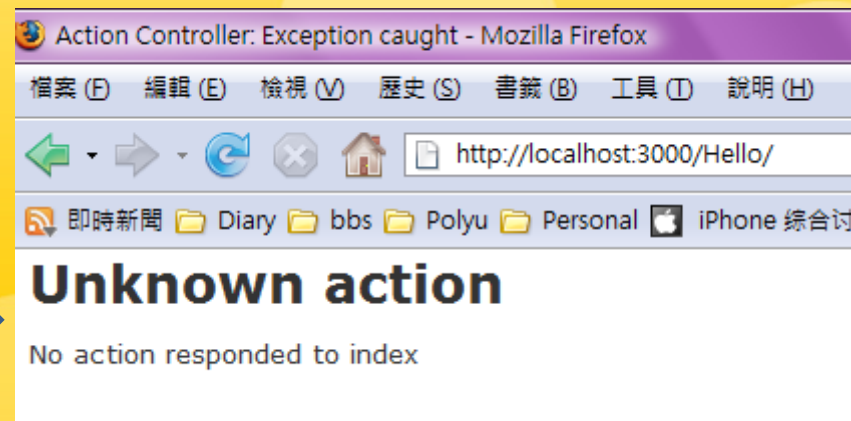
Creating the Controller

- At currently working directly:

ruby script/generate controller Hello



```
C:\Windows\system32\cmd.exe
C:\RailProgram\HelloWorld>ruby 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
C:\RailProgram\HelloWorld>
```



Creating an Action

- Controller is boss of the application, it has various ACTIONS that perform separate task.

- ```
class HelloController <
 ApplicationController
end
```



```
class HelloController<ApplicationController
 def helloMe
 end
end
```

controllers

# Creating an Action

- Start WEBrick server and browse <http://localhost:3000/Hello/helloMe>



- You need a view!

# Creating a View Template

- Controller → handle request from user
- Action → give controller responds to requests
- You need some ways of returning results to user.
- `defName.rhtml` (`helloMe.rhtml`)

# Creating a View Template

- In controller, modify `helloMe` action

```
def helloMe
 @time_now = Time.now
end
```

- Create `helloMe.rhtml` at `\app\views\hello\`

# View – Mixing Ruby and HTML

```
<html>
<head>
<title>Using Ruby on Rails</title>
</head>
<body>
<h1>This is RoR Hello Me!</h1>
 Hope this sharing session would help you.

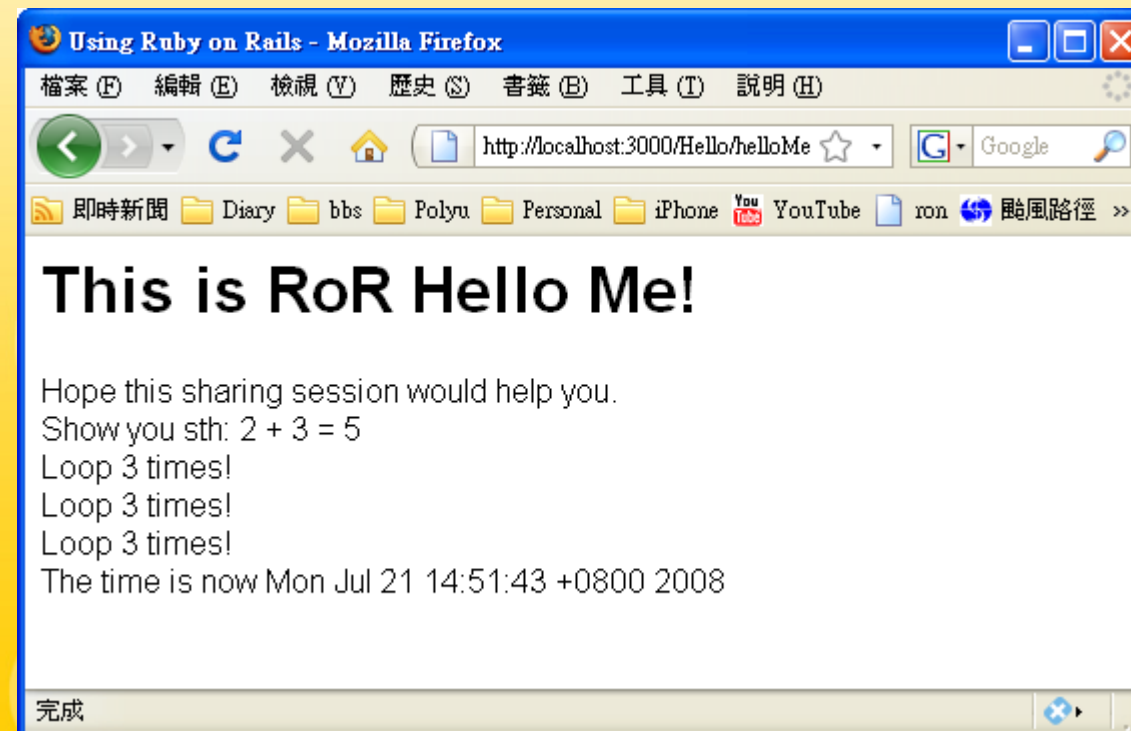
 Show you sth: 2 + 3 = <%= 2 + 3 %>

 <% 3.times do %>
 Loop 3 times!

 <% end %>
 The time is now <%= @time_now %>
</body>
</html>
```

# Creating a View Template

- Pay attention to the code inside the rhtml.
- `<%= %>`, `<% %>` is the defined area of Ruby.

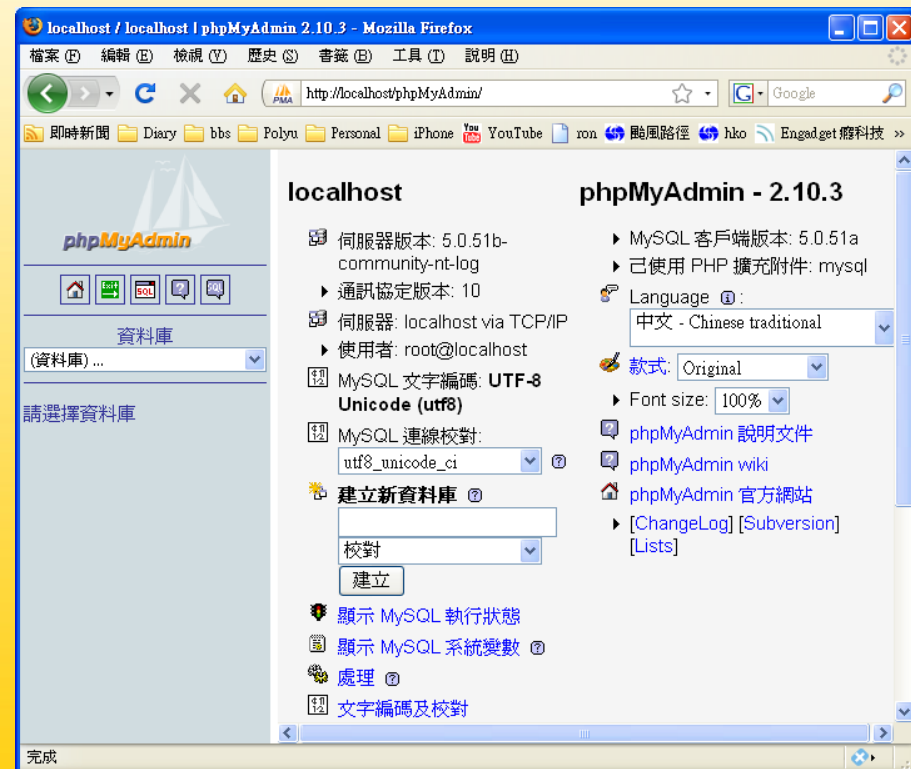
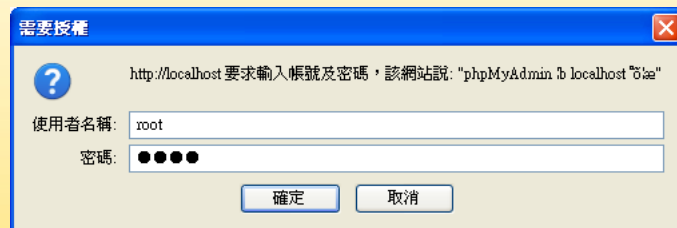


## Adding 2<sup>nd</sup> Action & Linking

- Basically, putting 1 more `def` in controller (Let's say: `yourName`)
- Generate 1 more `rhtml` in view.
- In the `rhtml`, add

```
<%= link_to "Test Text", :action => "yourName" %>
```

- Enter PHPMYAdmin by http://localhost





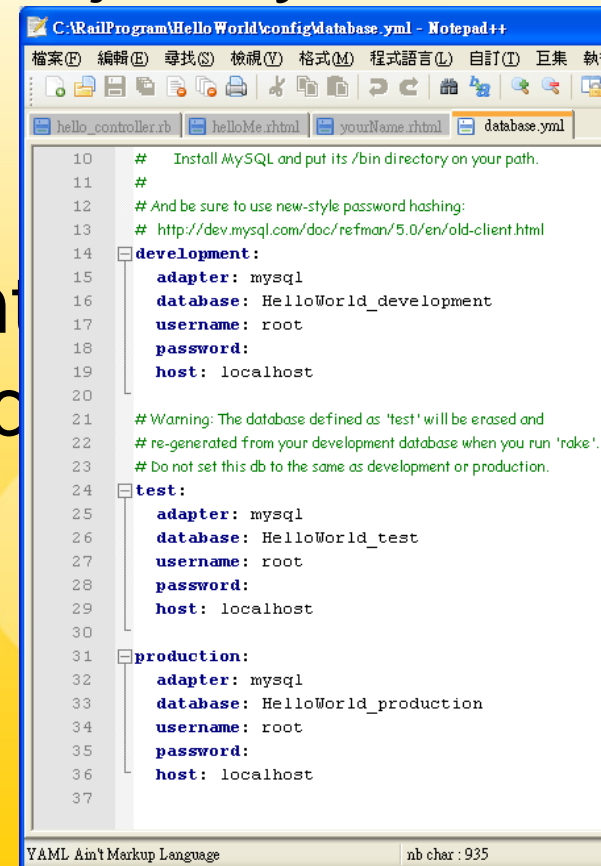
# Configure the database.yml

- After you create the new project, you can

1. Create your databases and edit  
config/database.yml

- In the current development database usage is referred to “development”

- Enter the account you know when installing appserv, or config in MySQL



```
C:\RailsProgram\Hello World\config\database.yml - Notepad++
檔案(F) 編輯(E) 尋找(S) 檢視(V) 格式(M) 程式語言(L) 自訂(I) 巨集(M) 執行(R)
hello_controller.rb helloMe.html yourName.html database.yml
10 # Install MySQL and put its /bin directory on your path.
11 #
12 # And be sure to use new-style password hashing:
13 # http://dev.mysql.com/doc/refman/5.0/en/old-client.html
14 development:
15 adapter: mysql
16 database: HelloWorld_development
17 username: root
18 password:
19 host: localhost
20
21 # Warning: The database defined as 'test' will be erased and
22 # re-generated from your development database when you run 'rake'.
23 # Do not set this db to the same as development or production.
24 test:
25 adapter: mysql
26 database: HelloWorld_test
27 username: root
28 password:
29 host: localhost
30
31 production:
32 adapter: mysql
33 database: HelloWorld_production
34 username: root
35 password:
36 host: localhost
37
YAML Ain't Markup Language nb char : 935
```

# Database & ActiveRecord in Rails

- Create Database and Table

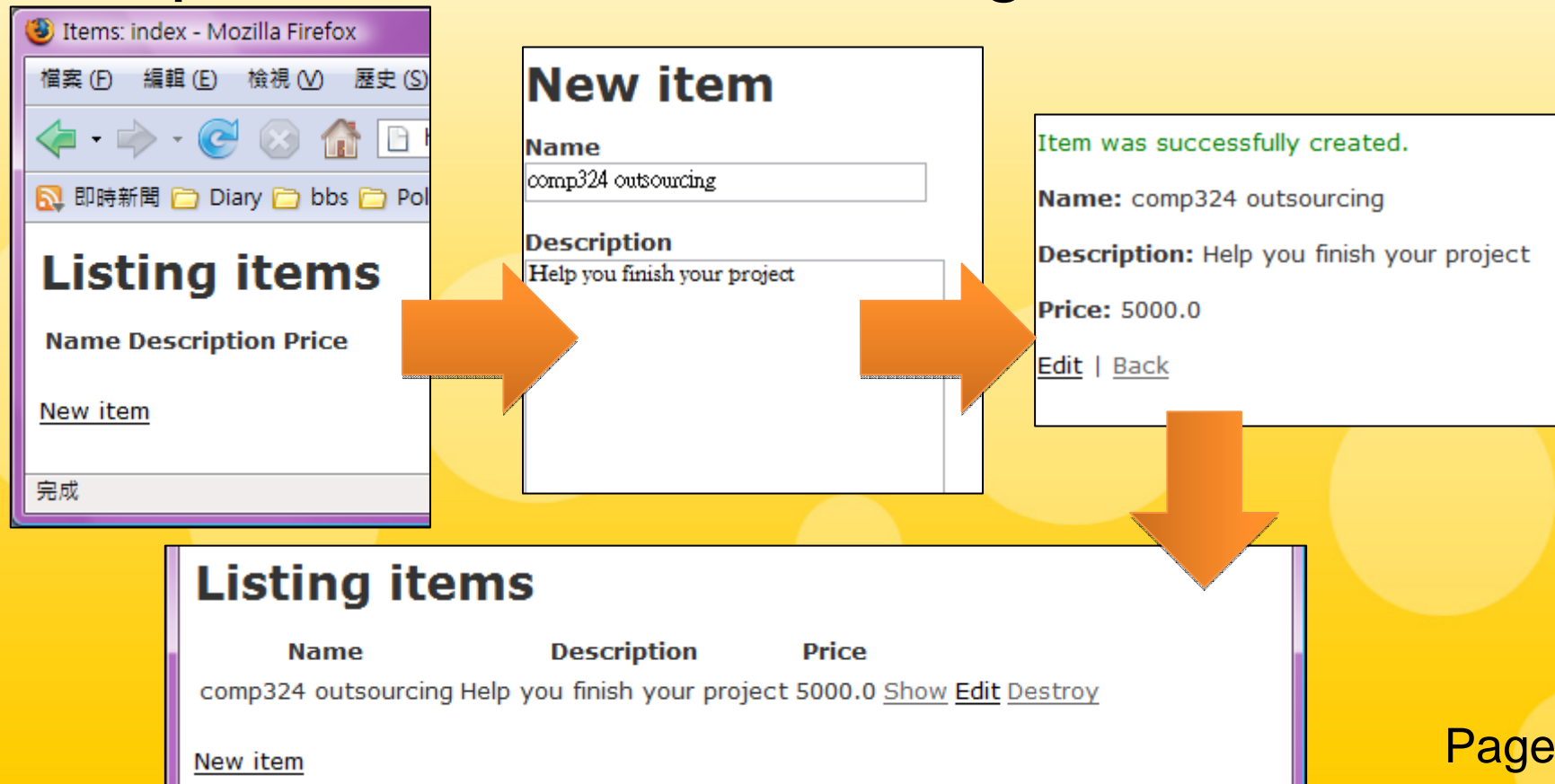
```
CREATE TABLE `items` (
 `id` INT NOT NULL AUTO_INCREMENT PRIMARY KEY ,
 `name` VARCHAR(80) NOT NULL ,
 `description` TEXT NOT NULL ,
 `price` DECIMAL(8, 2) NOT NULL
);
```

- Try the following command after creating the project and Database

```
ruby script/generate scaffold Item Manage
```

# Database & ActiveRecord in Rails

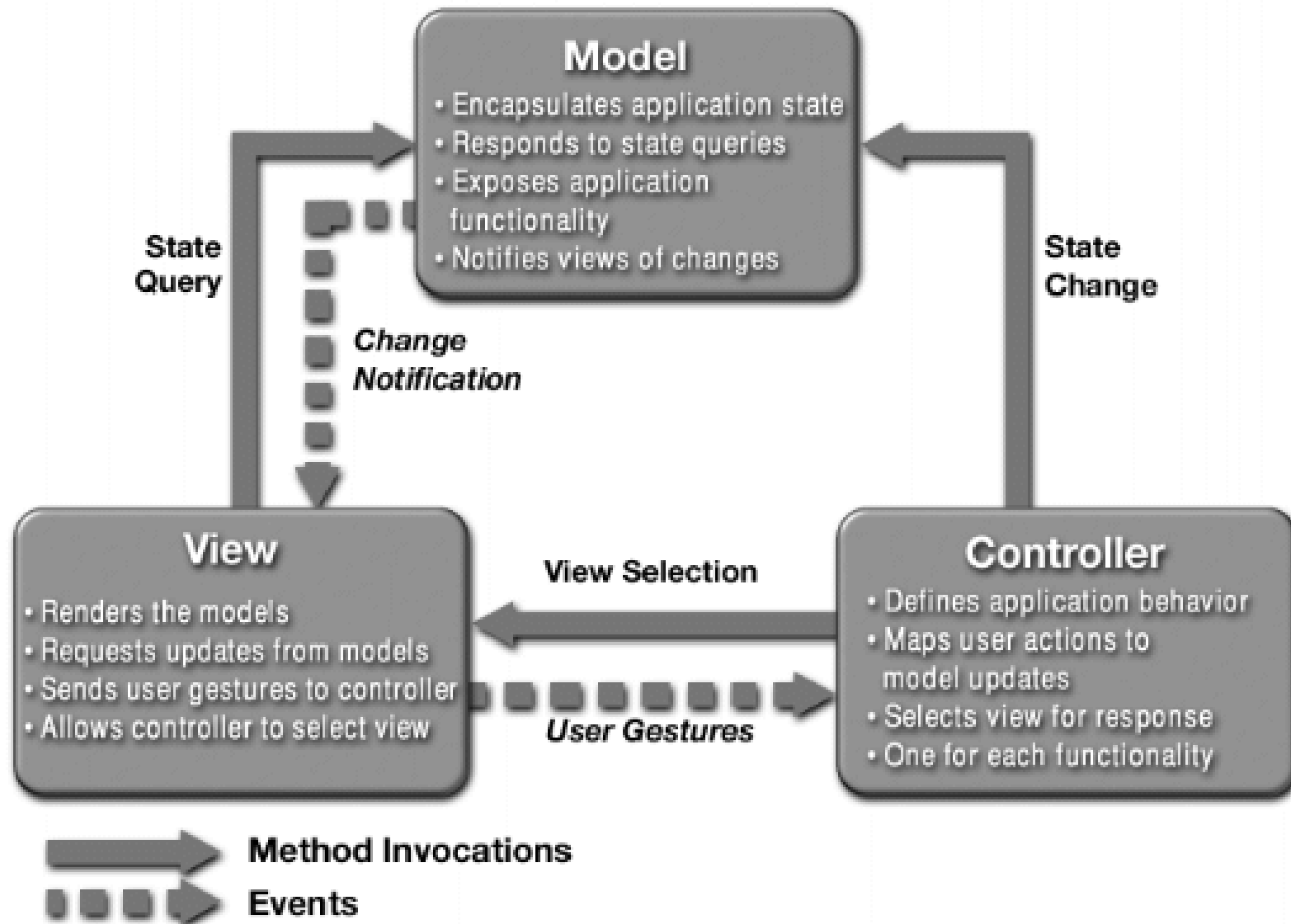
- Run WEBrick and browse <http://localhost:3000/manage>



# Try to investigate...

- Read the rhtml generated after scaffolding
- E.g. reading outside value in input box.  
(`@data = params[:textBoxName]`)
- Similar way to the checkbox, select box, etc...

# Model-View-Controller (MVC)



# Asynchronous Javascript And XML

- Naming at 2005 and starting to bloom
- Integrate the web technology in 20 years (XML, Javascript, Web standard)
- Relocate the model of the browser
- Ajax allows browser to contact server privately
- Advantage: better user experience
- Disadvantage: “寫死人”, “麻煩”

# Flow of Ajax

- 1) One event (e.g. on mouseover) trigger a javascript event
- 2) Javascript creates 1 XML HTTP Request
- 3) HTTP request → Server (Servlet/Rails/...)
- 4) Server returns XML
- 5) Client receives and Asynchronous changes the web elements (List/div/image...by DOM)

# From Ajax to Rails

- A simple non-ajax program

```
rails ajax
cd ajax
ruby script/generate controller ajax show time
```

```
C:\RailProgram\nofAjax>ruby script/generate controller ajax show time
exists app/controllers/
exists app/helpers/
create app/views/ajax
exists test/functional/
create app/controllers/ajax_controller.rb
create test/functional/ajax_controller_test.rb
create app/helpers/ajax_helper.rb
create app/views/ajax/show.rhtml
create app/views/ajax/time.rhtml

C:\RailProgram\nofAjax>
```



# From Ajax to Rails

- At app/controllers/ajax\_controller.rb

```
1 ajax_controller.rb
1 - class AjaxController < ApplicationController
2
3 - def show
4 end
5
6 - def time
7 end
8 end
```

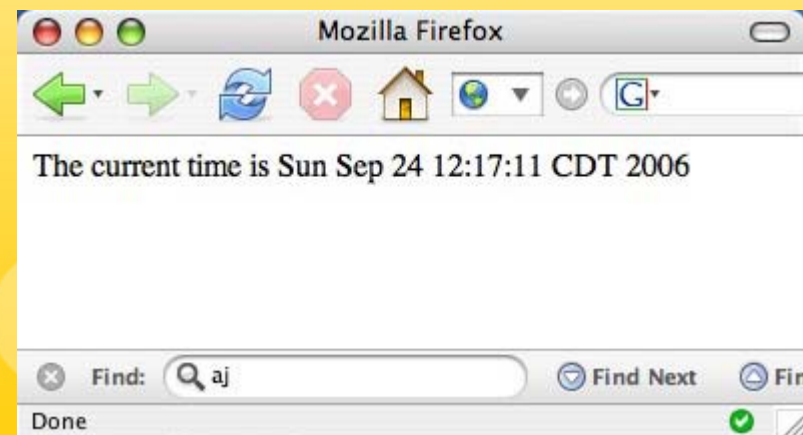
- Modify the app/view/ajax/show.rhtml

```
<h1>Ajax show</h1>
Click this link to show the current <%= link_to
"time", :action => "time" %>.
```

# From Ajax to Rails

- app/controllers/ajax\_controller.rb

```
def time
 render_text "The current time is #{Time.now.to_s}"
end
```



# From Ajax to Rails

- Adding ajax – modify app/view/show.rhtml

```
<%= javascript_include_tag :defaults %>
<h1>Ajax show</h1>
Click this link to show the current
<%= link_to_remote "time", :update => 'time_div',
:url => {:action => "time"} %>.

<div id='time_div'>
</div>
```

- <http://localhost:3000/ajax/show>

# See the following ajax code

```
<script src="/javascripts/prototype.js?1159113688"
type="text/javascript"></script> <script
src="/javascripts/effects.js?1159113688"
type="text/javascript"></script> <script
src="/javascripts/dragdrop.js?1159113688"
type="text/javascript"></script> <script
src="/javascripts/controls.js?1159113688"
type="text/javascript"></script> <script
src="/javascripts/application.js?1159113688"
type="text/javascript"></script> <h1>Ajax show</h1>
Click this link to show the current <a href="#"
onclick="new Ajax.Updater('time_div', '/ajax/time',
{asynchronous:true, evalScripts:true}); return
false;">time.
 <div id='time_div'> </div>
```

# Periodically updates

- Modify the show.rhtml

```
<%= javascript_include_tag :defaults %>
<h1>Ajax show</h1>
Click this link to show the current
<%= javascript_include_tag :defaults %> <h1>Ajax
show</h1> <%= periodically_call_remote :update =>
'time_div', :url => {:action => "time"}, :frequency =>
1.0 %> <div id='time_div'> </div>
```

# Appendix – IDE for Ruby / RoR

- RadRails –  
<http://www.radrails.org/page/download>

# Reference

- Part of the materials in this notes are reproduced with the special permission of *Mr. K. P. Mark , Department of Information System, The City University of Hong Kong*