

# Ruby on Rails

web application dev

about me

**thomas mak**

thomas@cpttm.org.mo

agenda

Ruby Basic

Installation

Creating Rails Proj

# Ruby Basic

# Ruby Basic

## Variable

## Method

## Class

```
foo = "bar"
```

# Ruby Basic

## Variable

## Method

## Class

```
def method_name  
  do_something  
end
```



# Ruby Basic

## Variable

## Method

## Class

```
def hello(name)  
  puts "hello #{name}"  
end
```

# Ruby Basic

## Variable

## Method

## Class

```
"hello #{name}"
```

we can use `#{ }` to put variable in **double quoted** string.



# Ruby Basic

## Variable

## Method

## Class

```
hello "Thomas"  
hello("Thomas")
```

when we call the method with the parameters,  
we can either use the **()** or not.

# Ruby Basic

## Variable

## Method

## Class

```
# Class Def
class Foo
  def hello
    puts "Hello Ruby"
  end
end
```

# Ruby Basic

## Variable

## Method

## Class

```
# Class Def
```

for comments, we use # symbol.  
the comment valid till the line ending.



# Ruby Basic

## Variable

## Method

## Class

```
class Foo

  def hello
    puts "Hello Ruby"
  end

  private
  def internal_method
    do_something
  end
end
```

all methods below the **private** keyword are private.

# Ruby Basic

## Variable

## Method

## Class

```
def self.anther_method  
  something_not_related_to_instances  
end
```

definite the method with **self**. makes it a **class method**.



# Ruby Basic

## Variable

## Method

## Class

```
class Foo
  def initialize
    init_things_here
  end
end
```

an **initialize** method is a constructor method.

# Ruby Basic

## Variable

## Method

## Class

```
class Foo  
end  
  
obj = Foo.new
```

creating instance from class defition.

# Ruby Basic

## Variable

## Method

## Class

```
class Foo
  def initialize
    @bar = 123
  end
end
```

@variable with @ sign is instance variable.



# Ruby Basic

## Variable

## Method

## Class

```
class Foo
  attr_reader :bar
  def initialize
    @bar = 123
  end
end

obj = Foo.new
obj.bar # 123
obj.bar = 456
# NoMethodError: undefined method 'bar='
```

instance variable with **read-only** public access.

# Ruby Basic

## Variable

## Method

## Class

```
class Foo
  attr_accessor :bar
  def initialize
    @bar = 123
  end
end
```

instance variable with **read-write** public access.



# Ruby Basic

## Variable

## Method

## Class

```
class Foo
  attr_accessor :bar
  def initialize
    @bar = 123
  end

  def b=(value)
    @bar = value
    puts "Setting bar value to #{value}"
  end
end
```

and we can define custom setter method.

# Ruby Basic

## Variable

## Method

## Class

```
obj = Foo.new
obj.bar # 123
obj.b = 321 # Setting bar value to 321
obj.bar # 321
obj.b
# NoMethodError: undefined method 'b'
```


testing the Foo class with custom setter method.




## Using the Prompt

Good! You did a bit of math. See how the answer popped out?

Ruby recognizes numbers and mathematic symbols. You could try some other math like:

 `4 * 10`

 `5 - 12`

 `40 / 4`

Type `next` to move to the next lesson when you're finished.

<http://tryruby.org>



Interactive ruby ready.

> `next`

> `next`

>

`=> nil`

**Success!**

> `2+6`

`=> 8`

**Success!**

>

`Enter / Return` → Submit code  
`Up` → Cycle through submitted code  
`clear` → Clear the editor window  
`back` → Return to the previous lesson  
`next` → Move to the next lesson



# Installation

Installation

Mac

Unix

Win

**Installing XCode**

**Installing GCC**

**Installing brew**



# Installation Mac Unix Win



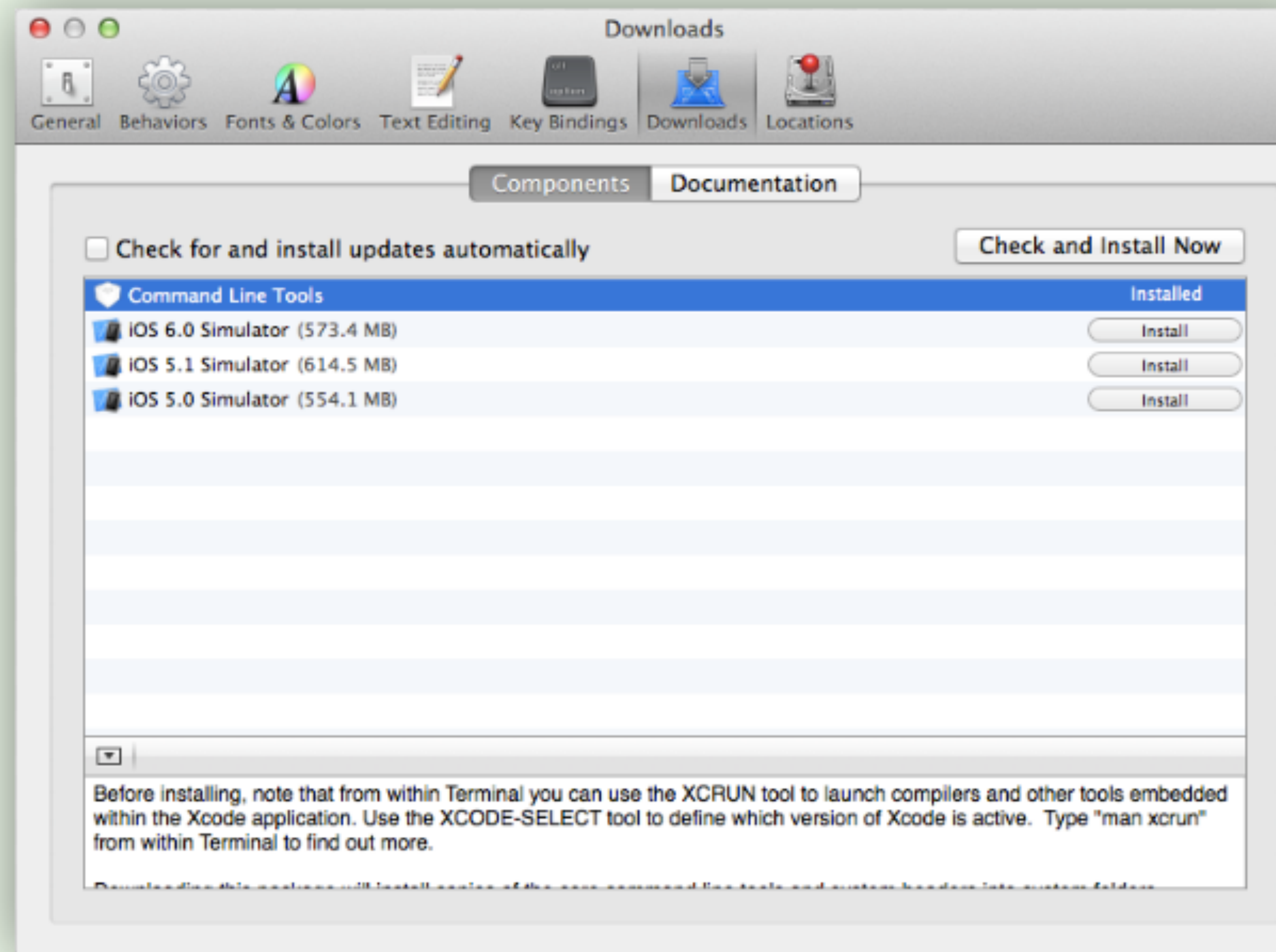


# Installation

## Mac

## Unix

## Win



# Installation

Mac

Unix

Win

```
$ruby -e "$(curl -fsSL https://raw.githubusercontent.com/mxcl/homebrew/go)"
```

installing **homebrew** via one-liner command.  
or check <http://brew.sh> for installation details.



Installation

Mac

Unix

Win

**Installing RVM**

**Installing Ruby**

Installing Git

Installing MySQL

Installing Pow

# Installation

Mac

Unix

Win

```
$ \curl -L https://get.rvm.io | bash -s stable
```

installing **RVM** via one-liner command.  
or check <http://rvm.io> for installation details.



# Installation

## Mac

## Unix

## Win

```
$ rvm install 1.9.3  
$ rvm install 2.0.0
```

with RVM, we can manage multiple versions of Rubies.

in case there is error with `gcc46`, use the following:

```
$ rvm install 1.9.3 --with-tag=clang
```

# Installation

## Mac

## Unix

## Win

```
$ rvm list

rvm rubies

=* ruby-1.9.3-p286 [ x86_64 ]
    ruby-2.0.0-p247 [ x86_64 ]

# => - current
# =* - current && default
#  * - default
```

listing all managed ruby versions.

# Installation

Mac

Unix

Win

```
$ rvm use 1.9.3  
$ gem install rails -v 3.2.8
```

specific the ruby version to use,  
and installing the rails with version specified.



Installation

Mac

Unix

Win

What is gem?

**software package.**

**as a library.**

**reusability.**



Installation

Mac

Unix

Win

<http://rubyinstaller.org>



# Creating Rails Project

# Rails Project

**create project**

framework

file structure

```
$ rails new app_name
```

creating new rails project.



# Rails Project

**create project**

framework

file structure

```
$ rails _3.2.8_ new app_name
```

specify rails version when creating project.



# Rails Project

create project

framework

file structure

# Rails is a framework

carefully designed for web app.

philosophy of best practices.

design pattern injected.

# Rails Project

create project

framework

file structure

app/

config/

db/

log/

public/

vendor/

Gemfile

Rails Project

create project

framework

file structure

app/

major project files.



# Rails Project

create project

framework

**file structure**

**app/**

**assets/**

for js/css files.

**controllers/**

business logic.

**models/**

data and validation.

**views/**

html outputs.

# Rails Project

create project

framework

file structure

## config/

all the configurations.

databases conncetion.

url routing.



**Rails Project**

**create project**

**framework**

**file structure**

**db/**

**database schema.**

**database versions control.**

Rails Project

create project

framework

file structure

log/

for debugging.



# Rails Project

create project

framework

file structure

public/

files that can be view publicly.

images assets.

# Rails Project

create project

framework

file structure

# Gemfile

gem dependency control.

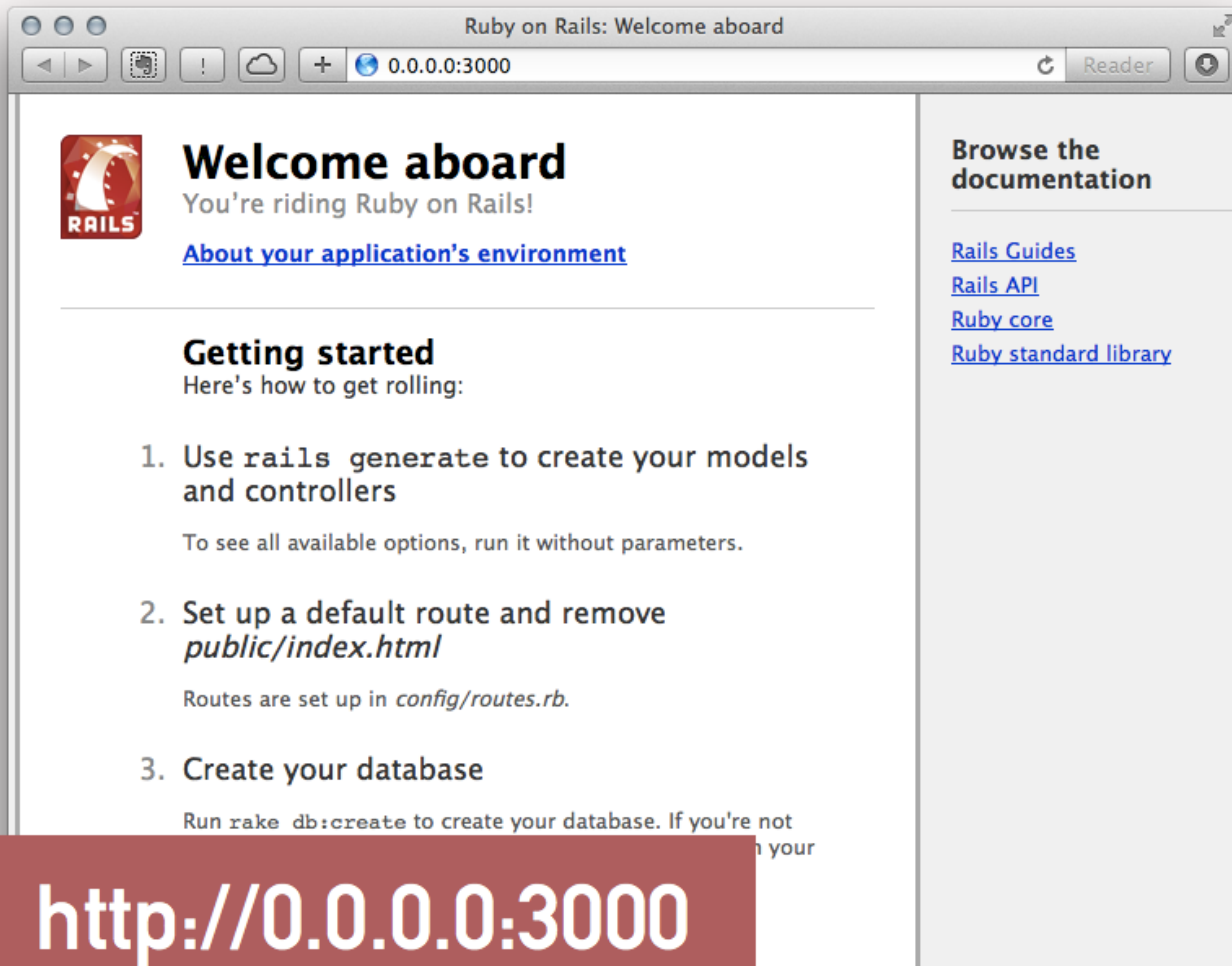


running the development server.

```
$ rails server
=> Booting WEBrick
=> Rails 3.2.8 application starting in development on http://0.0.0.0:3000
=> Call with -d to detach
=> Ctrl-C to shutdown server

[2013-09-06 14:32:54] INFO  WEBrick 1.3.1
[2013-09-06 14:32:54] INFO  ruby 1.9.3 (2012-10-12) [x86_64-darwin12.2.0]
[2013-09-06 14:32:54] INFO  WEBrick::HTTPServer#start: pid=17299 port=3000
```





<http://0.0.0.0:3000>



resources

[tryruby.org](https://tryruby.org)

[nitrous.io](https://nitrous.io)

[guides.rubyonrails.org](https://guides.rubyonrails.org)