

# Building Web Apps with Rack and Sinatra

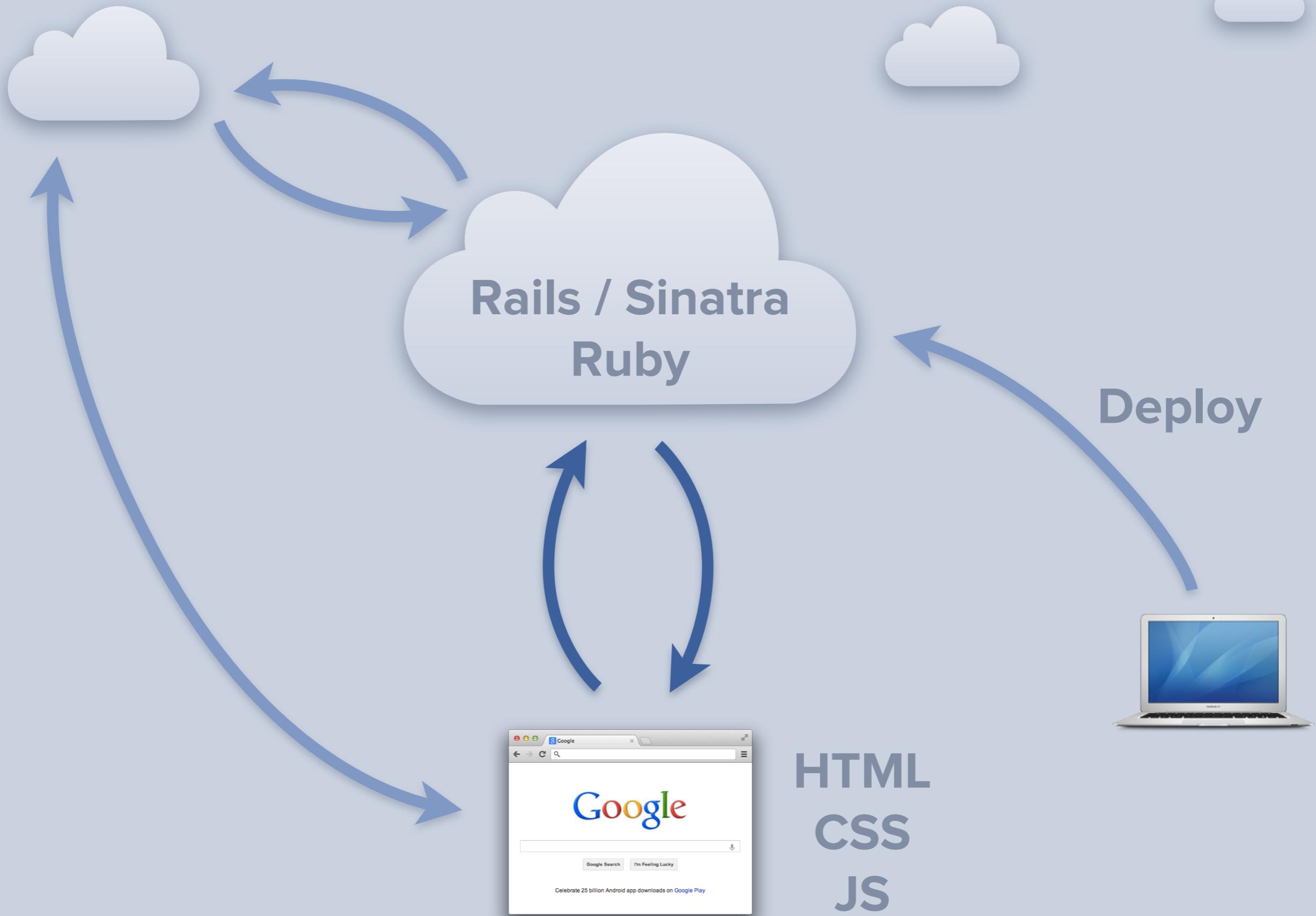
@blacktm

# Objectives

- ✓ Describe the HTTP request process
- ✓ Define Rack, describe its purpose
- ✓ Build a simple Rack web app
- ✓ Define, build, and use middleware
- ✓ Define DSL (compare with “framework”)
- ✓ Build a simple Sinatra app

# Things You'll Need

- ✓ `gem install thin`
- ✓ `gem install sinatra`
- ✓ `gem install heroku`
- ✓ Project files and notes at  
[blacktm.com](http://blacktm.com)



HTTP Request

The diagram illustrates the flow of an HTTP request through a stack of components. On the left, a vertical arrow labeled "HTTP Request" points downwards. To the right of this arrow is a large, light-gray cloud-like shape containing four rectangular boxes, each with a dark-gray border. From top to bottom, the boxes are labeled: "Your App", "Rack", "Middleware", and "Web Server".

Your App

Rack

Middleware

Web Server

HTTP Request

The diagram illustrates the flow of an HTTP request through a stack of components. A large, light-gray cloud shape contains four nested rectangular boxes. From bottom to top, the boxes are labeled: "Web Server", "Middleware", "Rack", and "Your App". To the left of the stack, a vertical arrow points downwards, with the text "HTTP Request" written vertically along its path. The arrow has a curved section that loops back towards the top of the stack before continuing straight down.

Your App

Rack

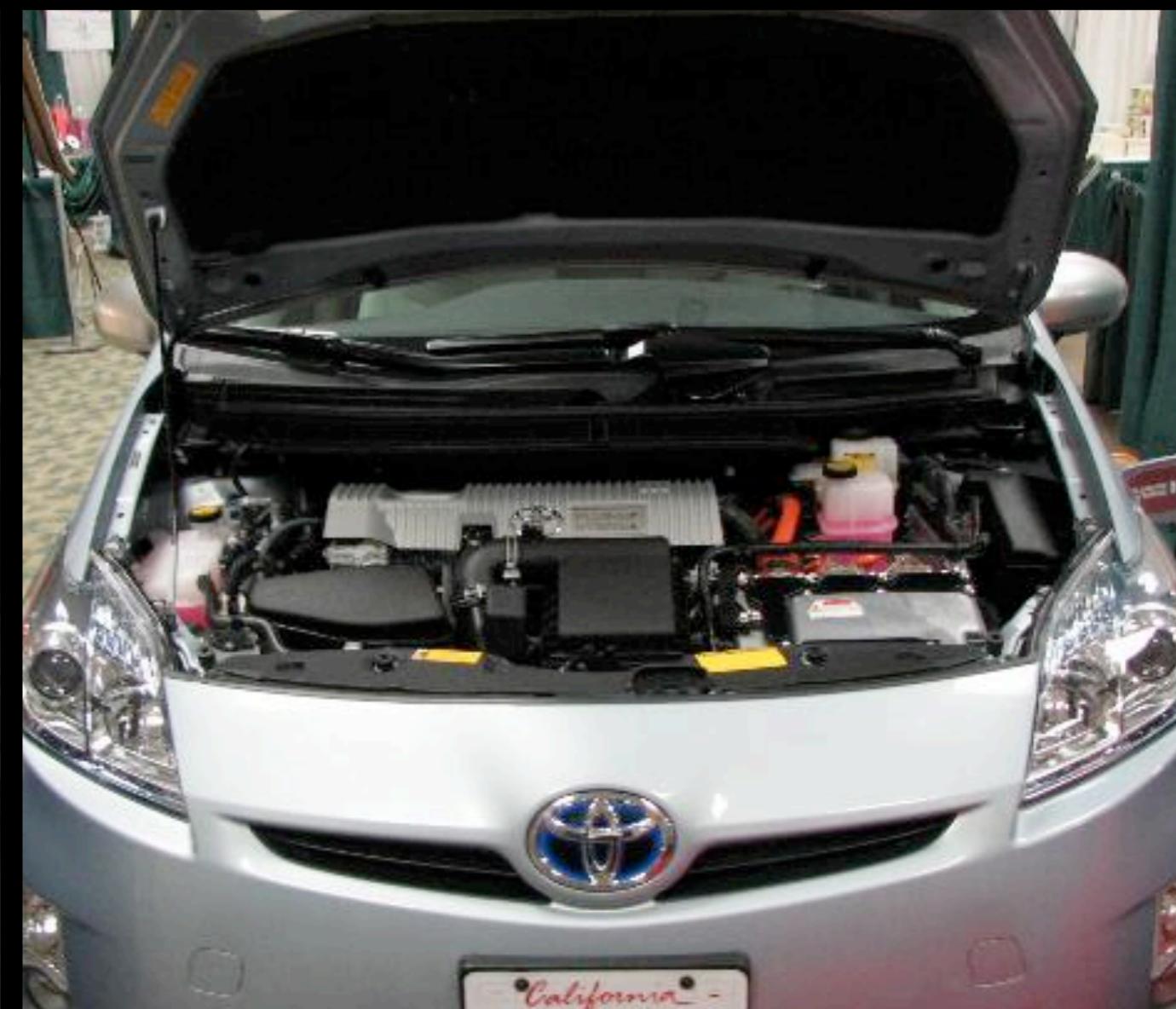
Middleware

Web Server

# What is Rack?

A modular web server interface.

(Connects your Ruby app to the web.)





A common interface.

Your App



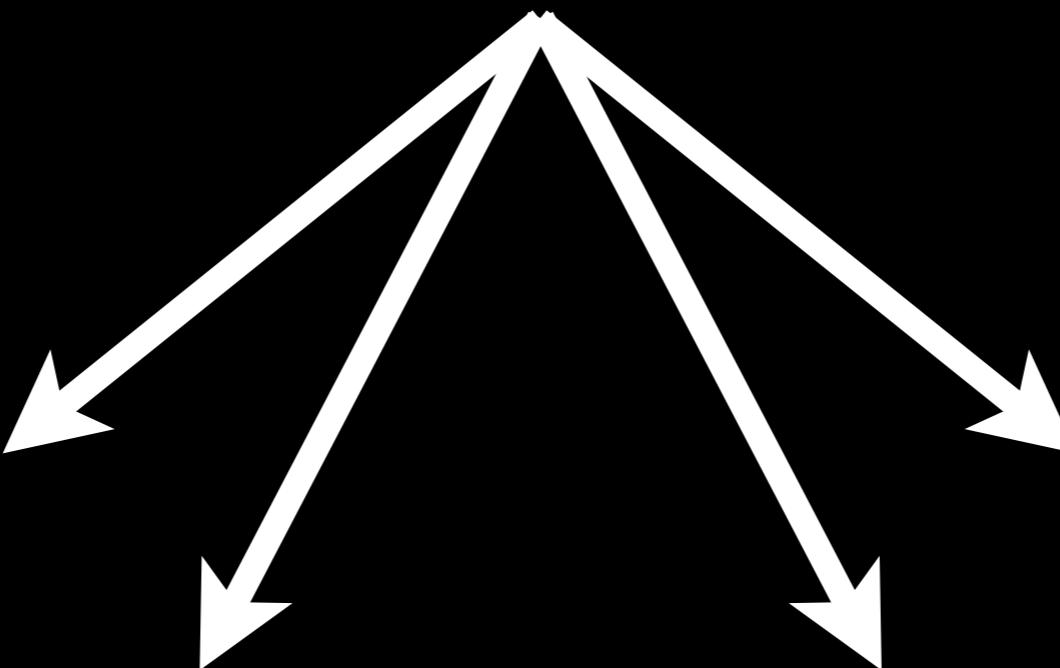
Rack

WEBrick

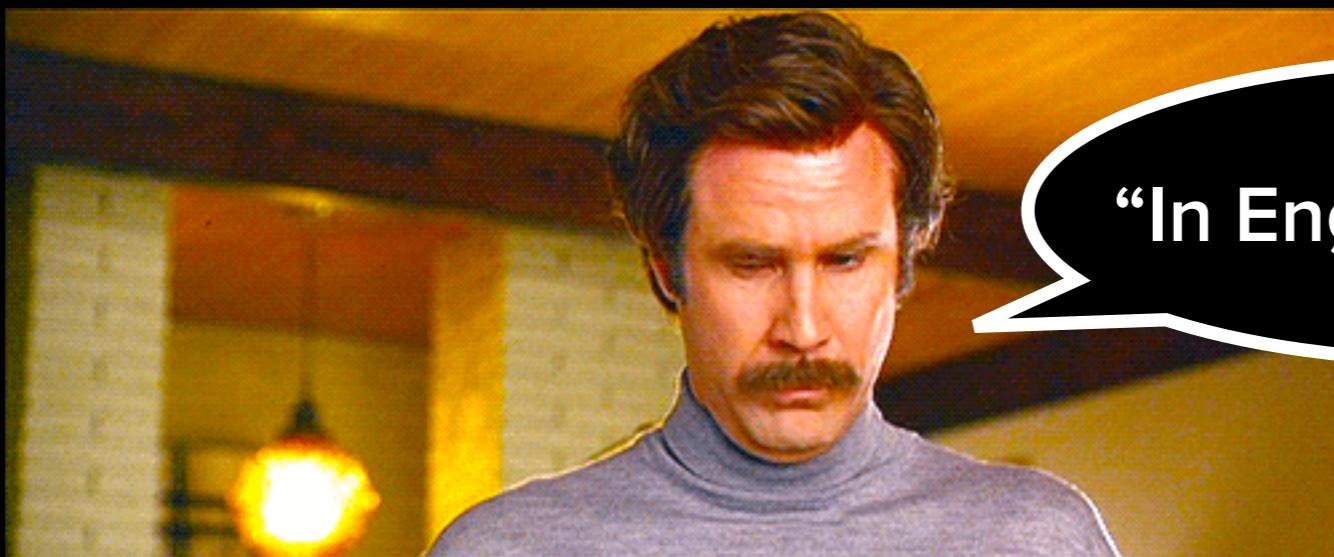
CGI

Mongrel

Thin



# What is a Rack app?



A Ruby object that responds to a “call” method, taking a single hash parameter, and returning an array which contains the response status code, response headers, and response body as an array of strings.

Open “simple\_rack” and edit the “config.ru” file.

---

```
# config.ru - rackup file
require 'rack'

class MyApp
  def call(env)
    headers = { "Content-Type" => "text/html" }
    [200, headers, ["Hello world!"]]
  end
end

run MyApp.new
```

---

```
$ rackup config.ru ← Run it!
```

# A very basic server.

`http://0.0.0.0:9292`

`#=> Hello world!`

`http://0.0.0.0:9292/there/isnt/a/route/here`

`#=> Hello world!`

# Who cares?

- ✓ You just built a web app. Damn.
- ✓ Important in understanding (and building) web frameworks (e.g. Rails)
- ✓ Middleware!

HTTP Request

The diagram illustrates the flow of an HTTP request through a stack of components. A large, light-gray cloud shape contains four nested rectangular boxes. From bottom to top, the boxes are labeled: "Web Server", "Middleware", "Rack", and "Your App". To the left of the stack, a vertical arrow points downwards, with the text "HTTP Request" written vertically along its path. The arrow has a curved section that loops back towards the top of the stack before continuing straight down.

Your App

Rack

Middleware

Web Server

**HTTP Request**

The diagram illustrates the flow of an HTTP request through a stack of components. A large, light-gray cloud shape contains four nested rectangular boxes. From bottom to top, the boxes are labeled: **Web Server**, **Middleware**, **Rack**, and **Your App**. To the left of the stack, a vertical arrow points downwards, with the text **HTTP Request** written vertically along its path. The arrow has a curved section where it wraps around the middle of the stack.

Your App

Rack

Middleware

Web Server

# Let's explore Middleware!

1. Open “rack\_middleware”
2. Stop the server (ctrl-c)
3. Run: \$ rackup config.ru
4. Open: <http://0.0.0.0:9292>  
or: <http://localhost:9292>

Rack::Lint::LintError at / 0.0.0.0:9292

# Rack::Lint::LintError at /

No Content-Type header found

Ruby /Users/Tom/.rvm/gems/ruby-1.9.3-p194/gems/rack-1.4.1/lib/rack/lint.rb: in assert, line 19  
Web GET 0.0.0.0/

Jump to:

[GET](#) | [POST](#) | [Cookies](#) | [ENV](#)

## Traceback (innermost first)

```
/Users/Tom/.rvm/gems/ruby-1.9.3-p194/gems/rack-1.4.1/lib/rack/lint.rb: in assert
  19.         raise LintError, message ...
/Users/Tom/.rvm/gems/ruby-1.9.3-p194/gems/rack-1.4.1/lib/rack/lint.rb: in check_content_type
  476.       assert("No Content-Type header found") ...
/Users/Tom/.rvm/gems/ruby-1.9.3-p194/gems/rack-1.4.1/lib/rack/lint.rb: in _call
  54.         check_content_type status, headers ...
/Users/Tom/.rvm/gems/ruby-1.9.3-p194/gems/rack-1.4.1/lib/rack/lint.rb: in call
  36.         dup._call(env) ...
/Users/Tom/.rvm/gems/ruby-1.9.3-p194/gems/rack-1.4.1/lib/rack/showexceptions.rb: in call
  24.         @app.call(env) ...
/Users/Tom/.rvm/gems/ruby-1.9.3-p194/gems/rack-1.4.1/lib/rack/commonlogger.rb: in call
  20.         status, header, body = @app.call(env) ...
/Users/Tom/.rvm/gems/ruby-1.9.3-p194/gems/rack-1.4.1/lib/rack/chunked.rb: in call
  43.         status, headers, body = @app.call(env) ...
/Users/Tom/.rvm/gems/ruby-1.9.3-p194/gems/rack-1.4.1/lib/rack/content_length.rb: in call
  14.         status, headers, body = @app.call(env) ...
/Users/Tom/.rvm/gems/ruby-1.9.3-p194/gems/thin-1.5.0/lib/thin/connection.rb: in block in pre_process
  81.           response = @app.call(@request.env) ...
/Users/Tom/.rvm/gems/ruby-1.9.3-p194/gems/thin-1.5.0/lib/thin/connection.rb: in catch
  79.             catch(:async) do ...

```

Rack::Lint::LintError at /  
No Content-Type header found

Ruby /Users/Tom/.rvm/gems/ruby-1.9.3-p194/gems/rack-1.4.1/lib/rack/lint.rb: in assert, line 19  
Web GET 0.0.0.0/

Jump to:

[GET](#) | [POST](#) | [Cookies](#) | [ENV](#)

**Traceback** (innermost first)

```
/Users/Tom/.rvm/gems/ruby-1.9.3-p194/gems/rack-1.4.1/lib/rack/lint.rb: in assert
  19.         raise LintError, message ...
/Users/Tom/.rvm/gems/ruby-1.9.3-p194/gems/rack-1.4.1/lib/rack/lint.rb: in check_content_type
  476.     assert("No Content-Type header found") ...
/Users/Tom/.rvm/gems/ruby-1.9.3-p194/gems/rack-1.4.1/lib/rack/lint.rb: in _call
  54.     check_content_type status, headers ...
/Users/Tom/.rvm/gems/ruby-1.9.3-p194/gems/rack-1.4.1/lib/rack/lint.rb: in _call
  36.     dup._call(env) ...
/Users/Tom/.rvm/gems/ruby-1.9.3-p194/gems/rack-1.4.1/lib/rack/showExceptions.rb: in call
  24.     @app.call(env) ...
/Users/Tom/.rvm/gems/ruby-1.9.3-p194/gems/rack-1.4.1/lib/rack/commonlogger.rb: in call
  20.     status, header, body = @app.call(env) ...
/Users/Tom/.rvm/gems/ruby-1.9.3-p194/gems/rack-1.4.1/lib/rack/chunked.rb: in call
  45.     status, headers, body = @app.call(env) ...
/Users/Tom/.rvm/gems/ruby-1.9.3-p194/gems/rack-1.4.1/lib/rack/contentLength.rb: in call
  14.     status, headers, body = @app.call(env) ...
/Users/Tom/.rvm/gems/ruby-1.9.3-p194/gems/thin-1.5.0/lib/thin/connection.rb: in block in pre_process
  81.         response = @app.call(@request.env) ...
/Users/Tom/.rvm/gems/ruby-1.9.3-p194/gems/thin-1.5.0/lib/thin/connection.rb: in catch
  79.         catch(:async) do ...
...
```

# Rack's Default Middleware (in development mode):

## Rack::CommonLogger

## Rack::ShowExceptions

## Rack::Lint



```
# app.rb

class MyApp
  def call(env)
    puts "Hello from MyApp"
    return [200, {}, ["Hello"]]
  end
end
```



No "Content-Type"

Solution?

Middleware!

# Open “config.ru”

---

```
class ContentType
  def initialize(app)
    @app = app
  end

  def call(env)
    status, headers, body = @app.call(env)
    puts "Hello from ContentType"

    # TODO: Add "Content-Type" to the HTTP
    #       headers and return the response.
  end
end
```

# Add the “Content-Type” to the headers.

```
# TODO: Add "Content-Type" to the HTTP  
#       headers and return the response.  
headers.merge!( "Content-Type" => "text/html" )  
return [status, headers, body]
```

```
class FinishSentence
  def initialize(app)
    @app = app
  end

  def call(env)
    status, headers, body = @app.call(env)
    puts "Hello from FinishSentence"

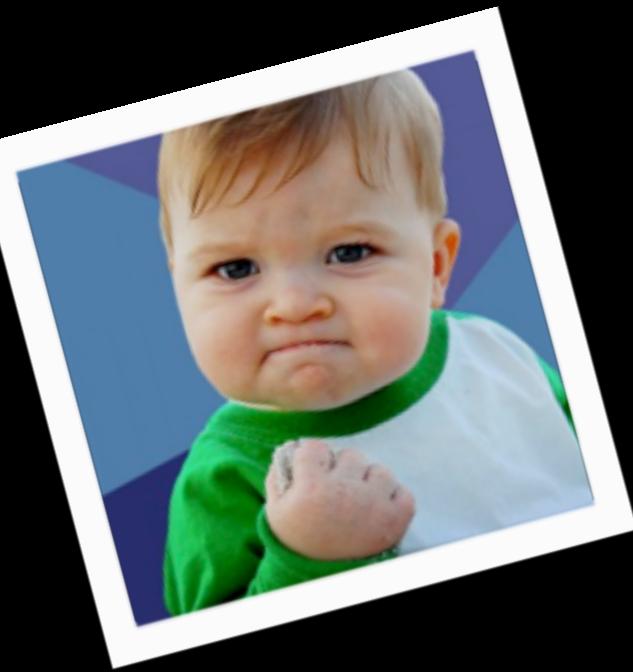
    # TODO: Add " world!" to the HTTP
    #       body and return the response.
  end
end
```

# Add the “ world!” to the body.

```
# TODO: Add " world!" to the HTTP  
#        body and return the response.  
  
body.push(" world!")  
return [status, headers, body]
```

# Use the middleware.

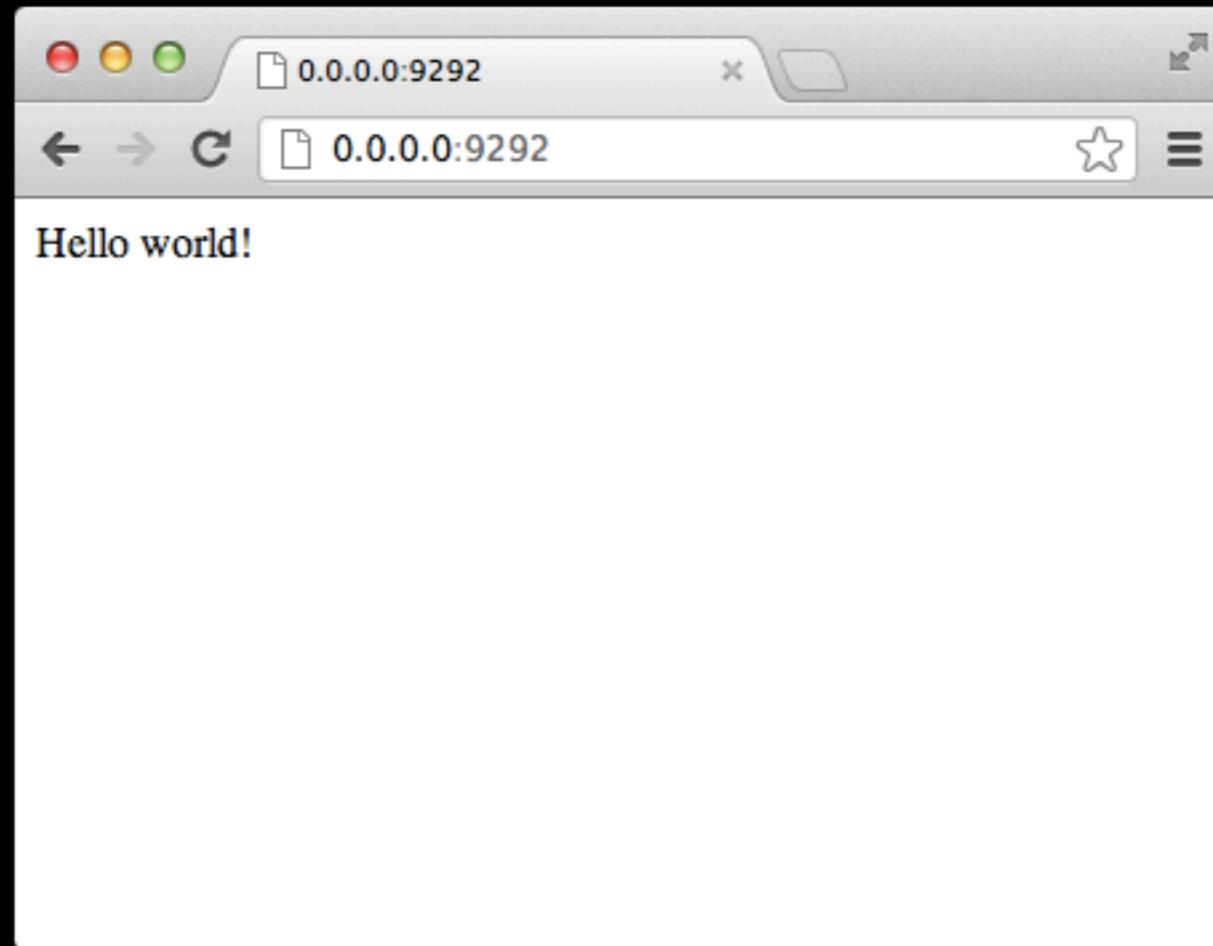
```
# TODO: Tell the app to use the "ContentType"  
#       and "FinishSentence" middleware.  
use ContentType  
use FinishSentence
```



# Run it!

```
$ rackup config.ru
```

```
http://0.0.0.0:9292
```



```
127.0.0.1 - - [13/Oct/2012 21:18:58] "GET / HTTP/1.1" 200 12 0.0009
```

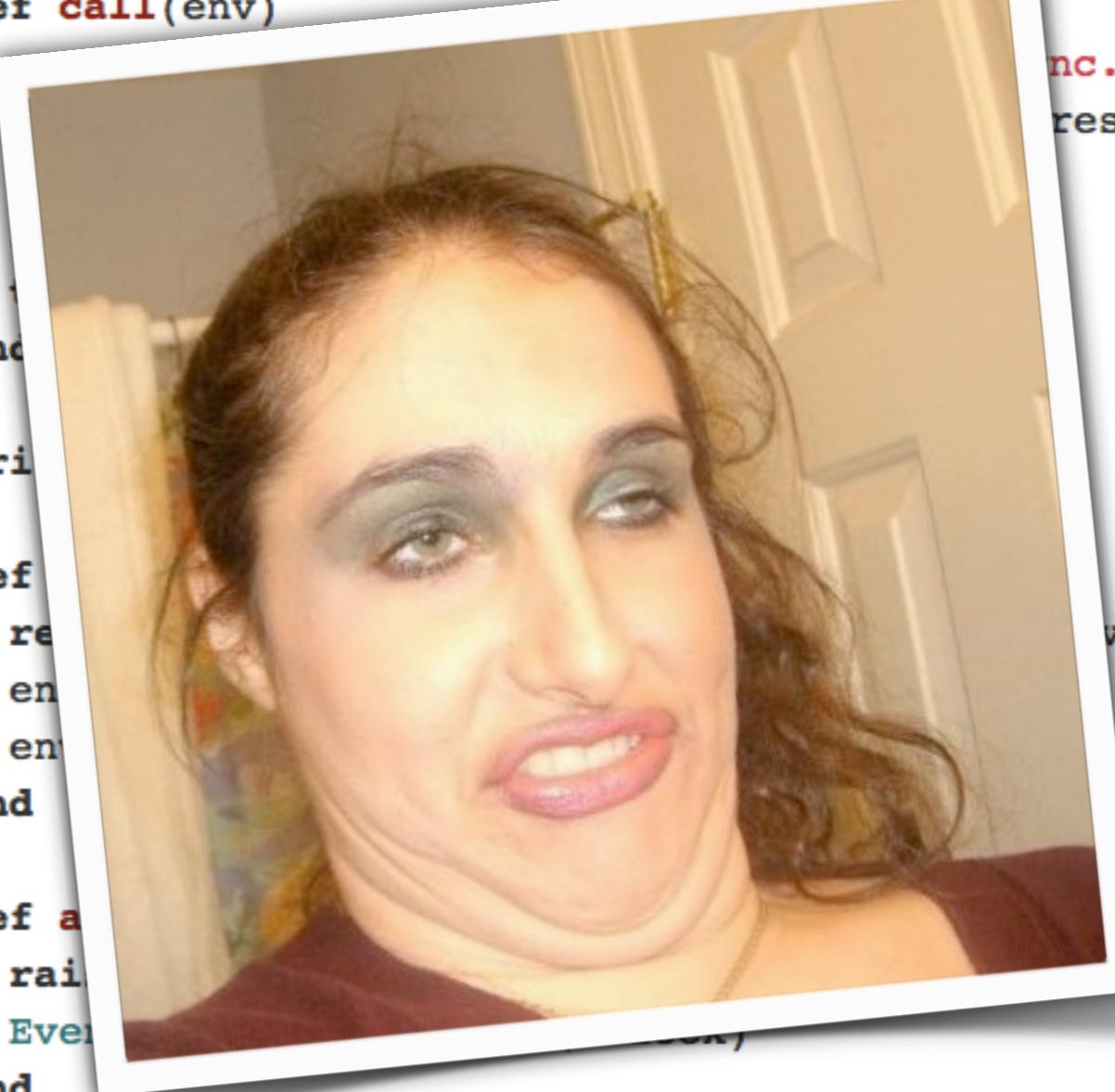
```
rack_middleware:$ rackup config.ru
>> Thin web server (v1.5.0 codename Knife)
>> Maximum connections set to 1024
>> Listening on 0.0.0.0:9292, CTRL+C to stop
{
  Hello from MyAPP
  Hello from FinishSentence
  Hello from ContentType
  127.0.0.1 - - [14/Nov/2012 17:01:07] "GET / H"
```

# What's wrong with this?

```
97  class ExtendedRack < Struct.new(:app)
98    def call(env)
99      result, callback = app.call(env), env['async.callback']
100     return result unless callback and async?(*result)
101     after_response { callback.call result }
102     setup_close(env, *result)
103     throw :async
104   end
105
106  private
107
108  def setup_close(env, status, header, body)
109    return unless body.respond_to? :close and env.include? 'as
110    env['async.close'].callback { body.close }
111    env['async.close'].errback { body.close }
112  end
113
114  def after_response(&block)
115    raise NotImplementedError, "only supports EventMachine at
116    EventMachine.next_tick(&block)
117  end
```

# What's wrong with this?

```
97  class ExtendedRack < Struct.new(:app)
98    def call(env)
99      env['HTTP_X_ACCEL_INTERNAL'] = "/internal"
100     env['HTTP_X_ACCEL_REDIRECT'] = "http://#{env['HTTP_HOST']}/internal"
101    env['HTTP_X_ACCEL_REDIRECT_TYPE'] = "auto"
102    env['HTTP_X_ACCEL_REDIRECT_ASIS'] = true
103    env['HTTP_X_ACCEL_REDIRECT_STATUS'] = 200
104  end
105
106  private
107
108  def self.call(env)
109    response = Rack::Response.new
110    env.each do |key, value|
111      response[key] = value if ENV['RACK_ENV'] == 'development' || value.included?
112    end
113
114    def response.append_header(key, value)
115      raise "Header already present: #{key}" if response.headers[key]
116      response[key] = value
117    end
118  end
119
```





# Sinatra

Put this in  
your pipe

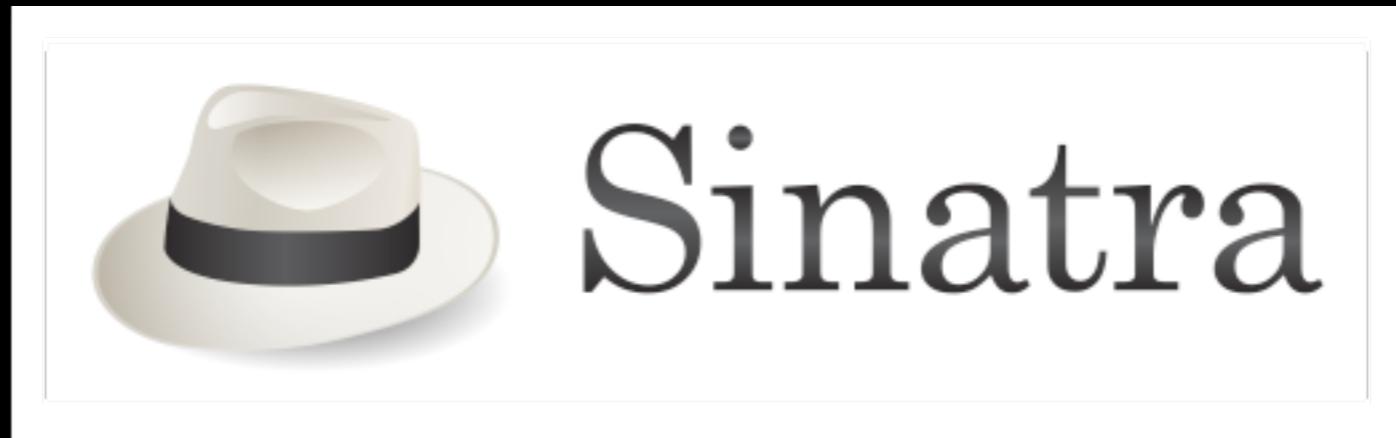
and smoke it

[README](#)  
[DOCUMENTATION](#)  
[CONTRIBUTE](#)  
[CODE](#)  
[CREW](#)  
[ABOUT](#)

```
require 'sinatra'
```

```
get '/hi' do
  "Hello World!"
end
```

```
$ gem install sinatra
$ ruby hi.rb
== Sinatra has taken the stage ...
>> Listening on 0.0.0.0:4567
```



“Domain Specific Language”



“Sinatra is a DSL for quickly creating web applications in Ruby with minimal effort.”

A “human interface” to Rack.



HTTP Request

The diagram illustrates the flow of an HTTP request through a stack of components. On the left, a vertical arrow labeled "HTTP Request" points downwards. To the right of this arrow is a large, light-gray cloud-like shape containing four rectangular boxes, each with a dark-gray border. From top to bottom, the boxes are labeled: "Your App", "Rack", "Middleware", and "Web Server".

Your App

Rack

Middleware

Web Server

HTTP Request

The diagram illustrates the flow of an HTTP request through a stack of components. On the left, a large cloud-like shape contains the text "HTTP Request". A curved arrow points from this text down towards a stack of three rectangular boxes. The top box is labeled "Your App", the middle box is labeled "Rack" with a nested box labeled "Middleware", and the bottom box is labeled "Web Server". All components are contained within a large, light-gray circular area.

Your App

Rack

Middleware

Web Server

HTTP Request

The diagram illustrates the flow of an HTTP request through a stack of application layers. On the left, a large cloud-like shape contains the text "HTTP Request". A thick, curved arrow originates from the bottom of this shape and points downwards towards a series of rectangular boxes stacked vertically. From top to bottom, the layers are labeled: "Your App", "Sinatra", "Rack", "Middleware", and "Web Server". Each layer is enclosed in its own rectangular box, which is then contained within a larger, rounded rectangle that encompasses the entire stack.

Your App

Sinatra

Rack

Middleware

Web Server

# The Showdown

Plain 'ol Rack

```
class MyApp
  def call(env)
    headers = { "Content-Type" => "text/html" }
    [200, headers, ["Hello world!"]]
  end
end
```

---

Sinatra

```
get '/' do
  "Hello world!"
end
```

Open “simple\_sinatra” and edit the “config.ru” file.

---

```
# config.ru
require 'sinatra'

get '/' do
  "Hello Sinatra!"
end

run Sinatra::Application
```

---

```
$ rackup config.ru ← Run it!
```

What else can  do?

```
# config.ru
require 'sinatra'

get '/' do
    "Hello Sinatra!"
end

get '/hello/:name' do |n|
    "Hello #{n}!"
end

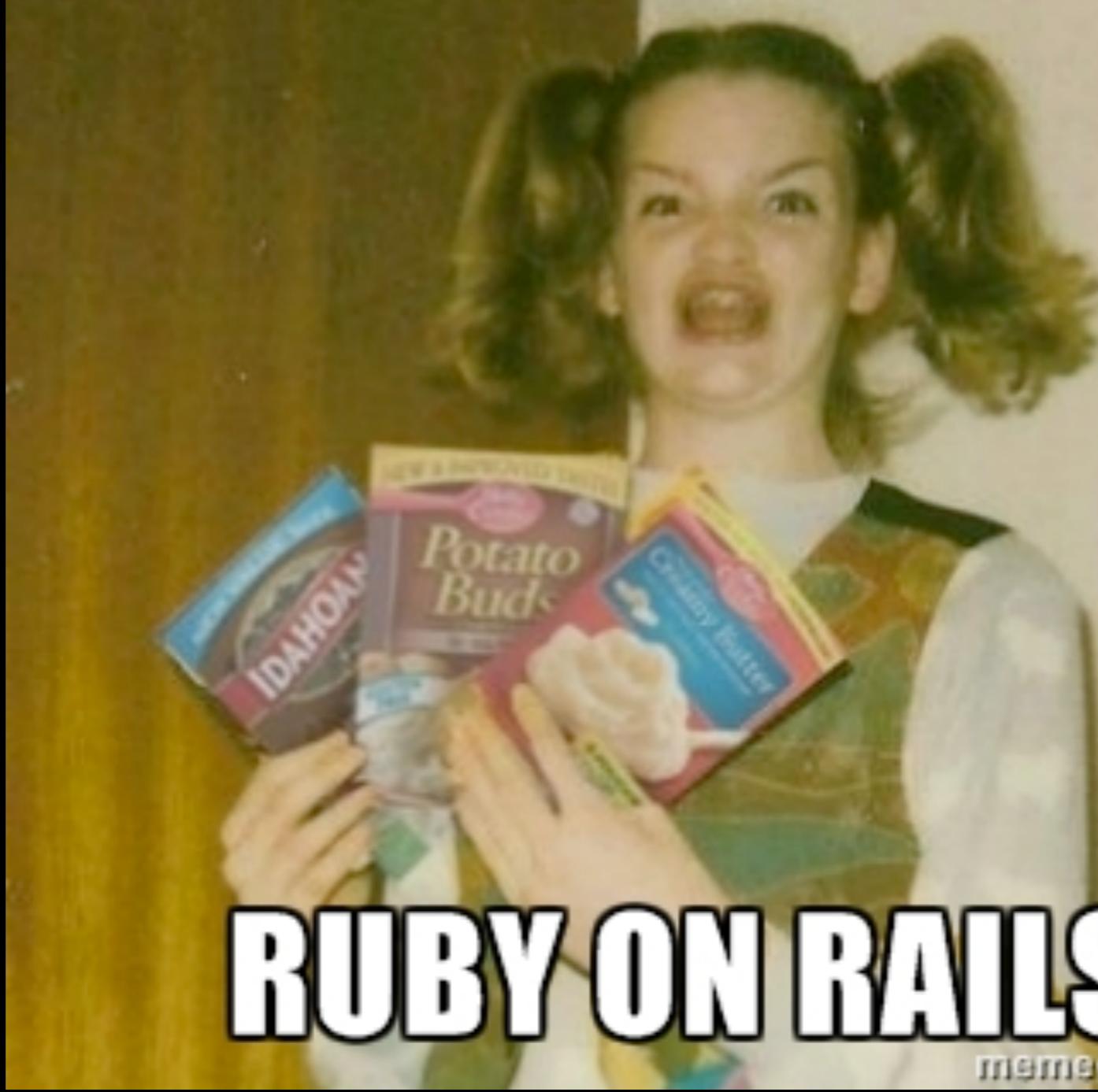
get '/wildcard/*' do
    request.inspect
end

error 404 do
    "OMG, 404!"
end

get '/breakit' do
    500
end

run Sinatra::Application
```

# ERMAHGERD



# RUBY ON RAILS

memegenerator.net

# Rails

“A full-stack web application framework.”

email

models

database

views

ajax

routes

sessions

orm

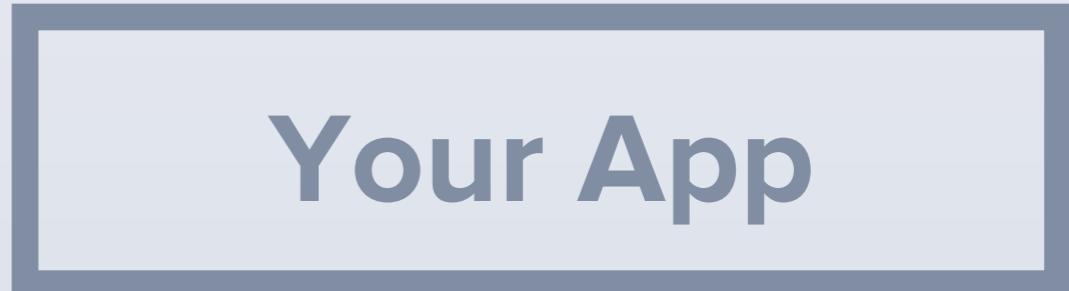
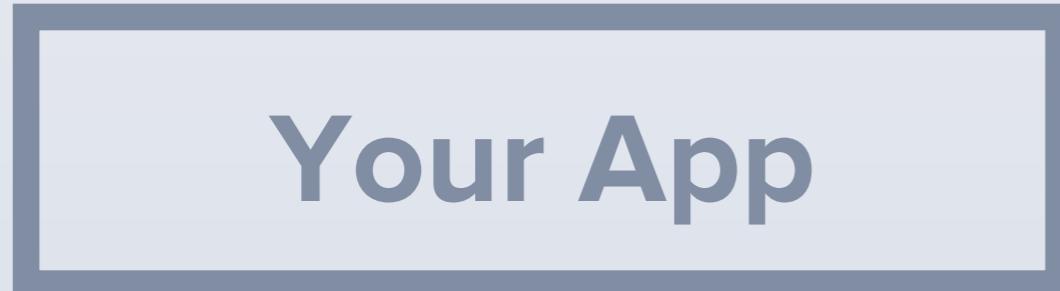
tasks

controllers

csrf

logging

tests



Your App

RAILS

Your App

Sinatra

Rack

Middleware

Web Server

Rack

Middleware

Web Server





Vacation rentals, private room

Where are you going?

BROWSE SIGN UP LOG IN

Tweet Pin It Google +1 Email Like 3

# Airstream B&B by Vintage Vacations

Camper/RV - Entire home/apt • Ashey Rd, Ryde, PO33 2UT, United Kingdom

Photos Maps Street View

Feedback

Heroku Status

Subscribe to Notifications

## heroku status

### current status and incident report

Production

99.92% October uptime

99.95% August - October

i

Development

98.47% October uptime

99.13% August - October

i

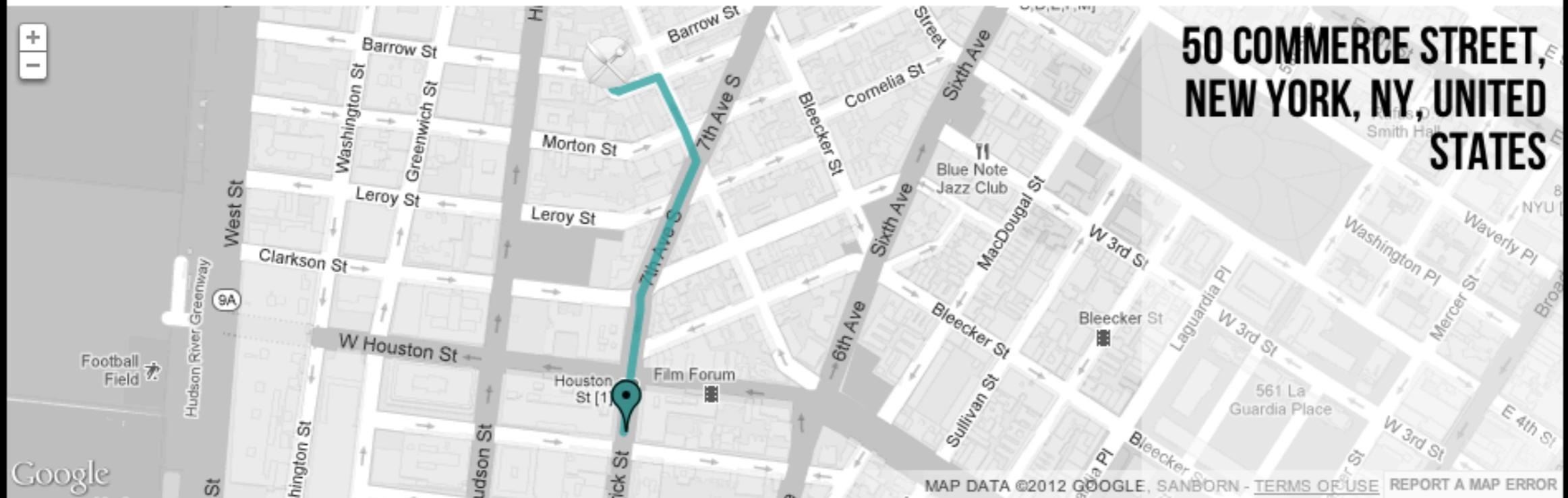
Production Apps Operational

Now

Development Apps & Tools Operational

# WHY DON'T YOU FUCKING GO TO COMMERCE

50 COMMERCE STREET,  
NEW YORK, NY, UNITED  
STATES



MY LOCATION IS  
FUCKING WRONG

NO, THAT PLACE  
LOOKS LIKE SHIT

ACTUALLY, I'M FUCKING  
THIRSTY



@WTFSIGTE

CREATED BY

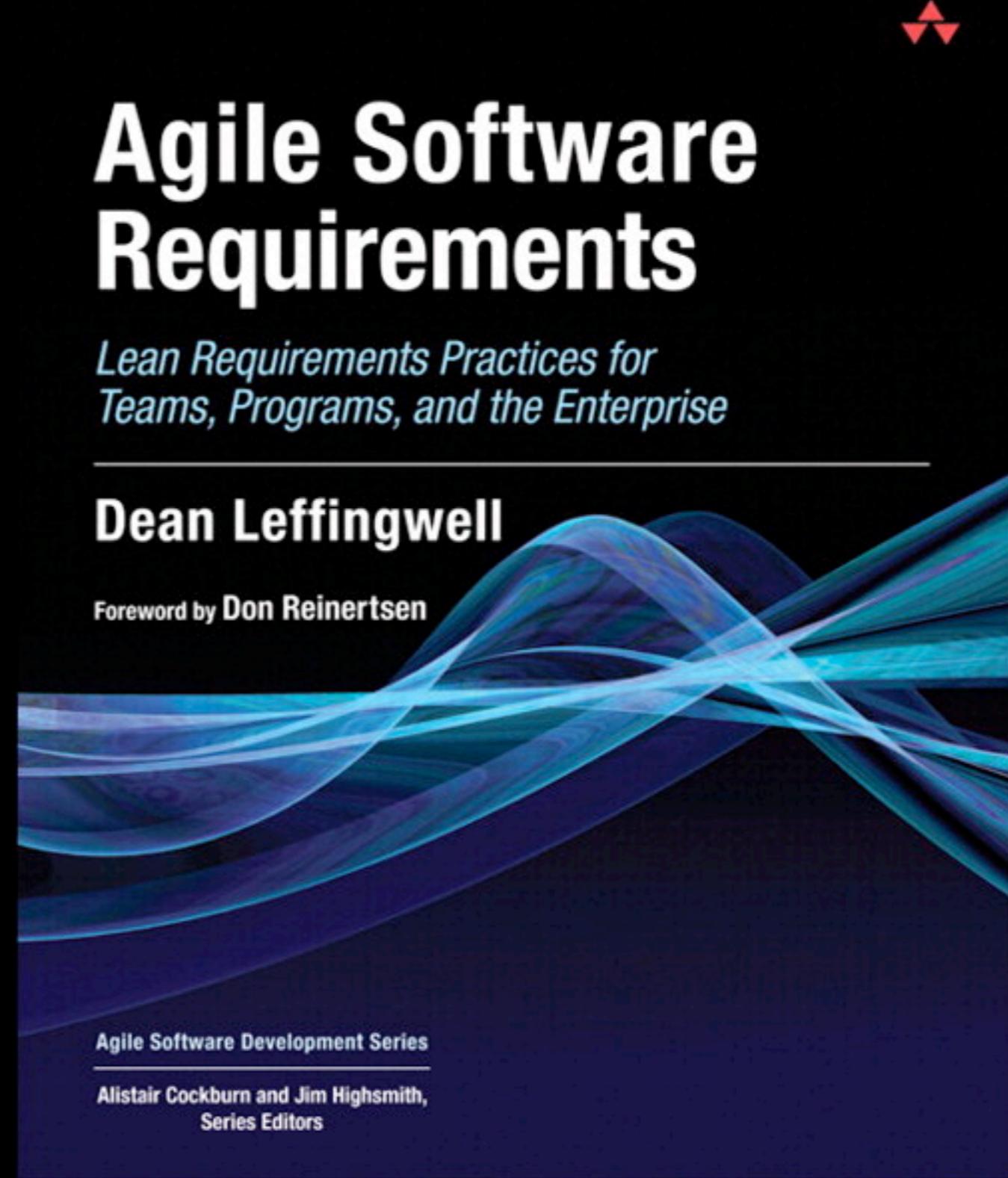
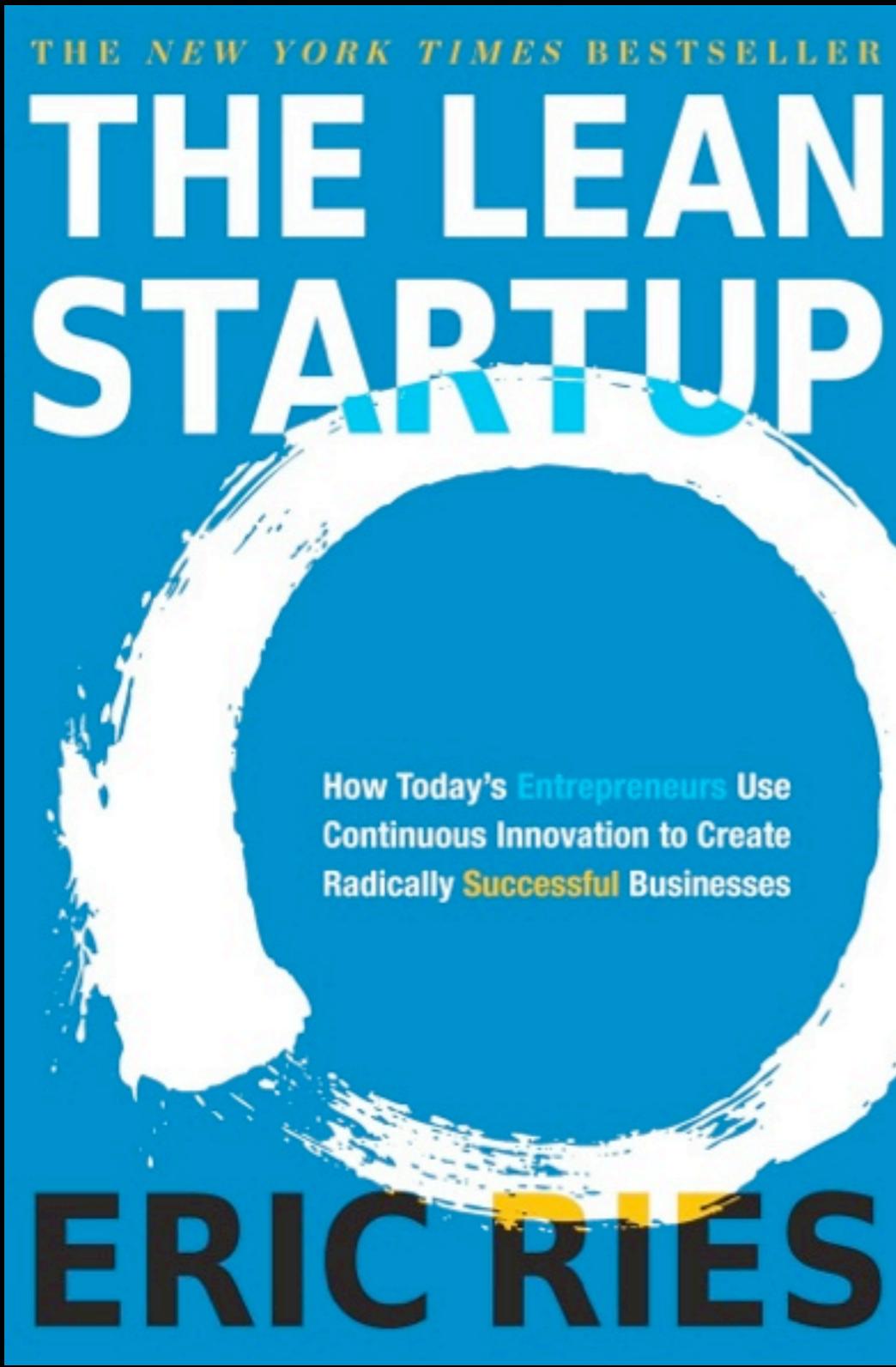
Like 56k

Tweet 2,254

+1 522

me





# OVERBUILT F-650



**DIESEL POWER**  
THE VOICE OF THE TURBODIESEL ENTHUSIAST  
[dieselpowermag.com](http://dieselpowermag.com)



# Messages App

Rack + Sinatra + DataMapper + Heroku

The screenshot shows a web application interface titled "Tom's Messages". The title bar is dark blue with white text. Below it, the main content area has a light gray background. Three messages are listed vertically, each enclosed in a thin gray border:

- Bob Saget**  
Hey man, I just sent you a message!  
2012-11-14T04:39:37+00:00
- Jimmy John**  
I have sammiches.  
2012-11-14T04:41:12+00:00
- Some Guy**  
This is liek pery neat.  
2012-11-14T04:42:25+00:00

# Thanks!

@blacktm