# A Rails App in a Single File

## Reverse Archeology Through Ruby Golf

Ryan Alyea @ROFISH

Ryan Alyea

Github & Twitter
@ROFISH


humblebundle.com/weekly

Ryan Alyea @ROFISH

# And now for something completely useless.

Ryan Alyea @ROFISH

# A Rails App in a Single File

Ryan Alyea @ROFISH

# A Rails App in a Single File

## Reverse Archeology Through Ruby Golf

Ryan Alyea @ROFISH

# Don’t try this at work.

Ryan Alyea @ROFISH

You can use Rails modules independently.

# Start with Rack

Ryan Alyea @ROFISH

# config.ru

```
require 'rubygems'
require 'bundler/setup' # Gemfile only lists 'rack'

run Proc.new {|env|
  if env["PATH_INFO"] == "/"
    [200,
      {"Content-Type" => "text/html"},
      ["<h1>Front Page</h1>"]
    ]
  else
    [404,
      {"Content-Type" => "text/html"},
      ["<h1>Not Found</h1>"]
    ]
  end
}
```

# config.ru

```
require 'rubygems'
require 'bundler/setup' # Gemfile only lists 'rack'

run Proc.new {|env|
  if env["PATH_INFO"] == "/"
    [200,
      {"Content-Type" => "text/html"},
      ["<h1>Front Page</h1>"]
    ]
  else
    [404,
      {"Content-Type" => "text/html"},
      ["<h1>Not Found</h1>"]
    ]
  end
}
```

# config.ru

```
require 'rubygems'
require 'bundler/setup' # Gemfile only lists 'rack'

run Proc.new {|env|
  if env["PATH_INFO"] == "/"
    [200,
      {"Content-Type" => "text/html"},
      ["<h1>Front Page</h1>"]
    ]
  else
    [404,
      {"Content-Type" => "text/html"},
      ["<h1>Not Found</h1>"]
    ]
  end
}
```

# config.ru

```
require 'rubygems'
require 'bundler/setup' # Gemfile only lists 'rack'

run Proc.new {|env|
  if env["PATH_INFO"] == "/"
    [200,
      {"Content-Type" => "text/html"},
      ["<h1>Front Page</h1>"]
    ]
  else
    [404,
      {"Content-Type" => "text/html"},
      ["<h1>Not Found</h1>"]
    ]
  end
}
```

All you need is the routes, the controllers, the views, and everything else.

Ryan Alyea @ROFISH

# config.ru

```
# Gemfile lists ‘rack’, ‘actionpack’

require 'rubygems'
require 'bundler/setup’

require 'action_dispatch’
```

# config.ru

```
# Gemfile lists ‘rack’, ‘actionpack’

require 'rubygems'
require 'bundler/setup’

require 'action_dispatch'

routes = ActionDispatch::Routing::RouteSet.new
```

# config.ru

```
# Gemfile lists ‘rack’, ‘actionpack’

require 'rubygems'
require 'bundler/setup’

require 'action_dispatch'

routes = ActionDispatch::Routing::RouteSet.new

routes.draw do
  get '/' => 'mainpage#index'
  get '/page/:id' => 'mainpage#show'
end
```

# config.ru (continued)

```
class MainpageController
end
```

# config.ru (continued)

```
class MainpageController
  def self.action(method)
  end
end
```

# config.ru (continued)

```
class MainpageController
  def self.action(method)
   lambda do
   end
  end
end
```

# config.ru (continued)

```
class MainpageController
  def self.action(method)
   lambda do
    if method == “index"
      [200, {"Content-Type" => "text/html"},
        ["<h1>Front Page</h1>”]
      ]
    elsif method == “show”
      [404, {"Content-Type" => "text/html"},
        ["<h1>Not Found</h1>"]
      ]
    end
   end
  end
end
```

# config.ru (continued)

```
class MainpageController
  def self.action(method)
    controller = self.new
    controller.method(method.to_sym)
  end
end
```

# config.ru (continued)

```
class MainpageController
  def self.action(method)
    controller = self.new
    controller.method(method.to_sym)
  end

  def index(env)
    [200, {"Content-Type" => "text/html"}, ["<h1>Front
Page</h1>"]]
  end

  def show(env)
    [200, {"Content-Type" => "text/html"}, ["<pre>
#{env['action_dispatch.request.path_parameters'][:id]}
</pre>"]]
  end
end
```

# config.ru (partial)

```
require 'action_controller'

class MainpageController < ActionController::Metal
end
```

# config.ru (partial)

```
require 'action_controller'

class MainpageController < ActionController::Metal
  def index
    self.response_body = "<h1>Front Page</h1>"
  end
end
```

# config.ru (partial)

```
require 'action_controller'

class MainpageController < ActionController::Metal
  def index
    self.response_body = "<h1>Front Page</h1>"
  end

  def show
    self.status = 404
    self.response_body =
"<pre>#{env['action_dispatch.request.path_parameters']
[:id]}</pre>"
  end
end
```

# config.ru (overview)

```
#require...

routes = ActionDispatch::Routing::RouteSet.new

routes.draw do
  get '/' => 'mainpage#index'
  get '/page/:id' => 'mainpage#show'
End

class MainpageController
  …
end

run routes
```

# config.ru (overview)

```
#require...

routes = ActionDispatch::Routing::RouteSet.new

routes.draw do
  get '/' => 'mainpage#index'
  get '/page/:id' => 'mainpage#show'
End

class MainpageController
  …
end

run routes
```

# That's Rails in a single Rackup file.

Ryan Alyea @ROFISH

# config.ru (overview)

```
#require...

routes = ActionDispatch::Routing::RouteSet.new

routes.draw do
  get '/' => 'mainpage#index'
  get '/page/:id' => 'mainpage#show'
End

class MainpageController
  …
end

use ActionDispatch::DebugExceptions
run routes
```

# config.ru (partial)

```
class MainpageController < ActionController::Metal
  include AbstractController::Callbacks

  …
end
```

# config.ru (partial)

```
class MainpageController < ActionController::Metal
  include AbstractController::Callbacks
  include ActionController::Redirecting

  …
end
```

# config.ru (partial)

```
class MainpageController < ActionController::Metal
  include AbstractController::Callbacks
  include ActionController::Redirecting
  include ActionController::UrlFor
  include routes.url_helpers

  …
end
```

# config.ru (partial)

```
class MainpageController < ActionController::Metal
  def index
    template = ERB.new(File.read('filename.erb'))
    @local_var = 12345
    self.response_body = template.result(binding)
  end
end
```

# config.ru (partial)

```
class MainpageController < ActionController::Metal
  include AbstractController::Rendering
  include ActionController::Rendering
  include ActionController::ImplicitRender
  def index
    @local_var = 12345
  end


  def render_to_body(*args)
    template = ERB.new File.read("#{params[:action]}.html.erb")
    template.result(binding)
  end
end
```

# config.ru (partial)

```
require 'action_view’

class MainpageController < ActionController::Metal
  include AbstractController::Rendering
  include ActionView::Rendering
  include ActionController::Rendering
  include ActionController::ImplicitRender

  prepend_view_path(‘app/views’)

  def index
    @local_var = 12345
  end
end
```

# config.ru (partial)

```
# include ALL THE THINGS

class MainpageController < ActionController::Base
  prepend_view_path('/path/to/templates')

  def index
    @local_var = 12345
  end
end
```

# config.ru

```
#require...

routes = ActionDispatch::Routing::RouteSet.new

routes.draw do
  get '/' => 'mainpage#index'
  get '/page/:id' => 'mainpage#show'
end

class MainpageController < ActionController::Base
  …
end

run routes
```

That is the bare essentials for a Rails app to go from request to response.

I hope this was enlightening to show modular Rails internals and/or inspired you on your next thin-sized mini app.

Ryan Alyea @ROFISH

Ryan Alyea

Github & Twitter
@ROFISH

This page intentionally blank