Introduction to Web Development

using Sinatra

Konstantinos Karasavvas

CITY College

August 10, 2013

Table of contents

Multiple Environments

- Different Environments, typically:
    - development
    - test
    - production
- Database adaptors
- Testing libraries
- ...

Multiple Environments: Gemfile

```
source 'http://rubygems.org'

gem 'sinatra', '~> 1.3.2'
gem 'sequel', '~> 3.45.0'
gem 'haml'
gem 'log4r'

group :development, :test do
  gem 'sqlite3'
end

group :production do
 #  gem 'pg'
  gem 'sqlite3'
end
```

```
$ bundle install
...
```

Multiple Environments: Tiny App

```
# user_app8.rb
class UserApp < Sinatra::Application

  configure :development, :test do
    set :db, Sequel.sqlite('db/users.db')
  end

  configure :production do
    # set :db = Sequel.connect('postgres://user:↩
        password@localhost/users')
    set :db, Sequel.sqlite('db/users.db')
  end

  # ...
```

```
# notice, no DB initialisation

class User < Sequel::Model
end
```

Multiple Environments: Tiny App, cont.

```
# config8.ru
#
# Notice, require './user_app8' is now before models

require 'bundler'

Bundler.require

require './user_app8'
require './models/users'

run UserApp.new
```

```
$ rackup -E production config8.ru
...
```

```
$ rackup config8.ru
...
```

Logging

- Why?
    - Diagnostic (something went wrong)
    - Audit (statistic analysis)
- Levels
    - `debug`
    - `info`
    - `warn`
    - `error`
    - `fatal`

Simple Logging: Tiny App

```ruby
# ...

get "/hello_form" do
  logger.info "Entered /hello_form route"
  @title = "Hello Stored User"
  haml :form2, :layout => true
end

# ...
```

File Logging: Tiny App

- Add a logging library to our Gemfile

```
# ...
gem 'log4r'
# ...
```

```
$ bundle install
...
```

File Logging: Tiny App, cont.

```
# config9.ru

require 'bundler'

Bundler.require

require './user_app9'
require './models/users'

logger = Log4r::Logger.new "app"
#logger.outputters << Log4r::Outputter.stdout
logger.outputters << Log4r::Outputter.stderr
log_path = "logs/" + Time.now.strftime("%Y%m%d-%H%M%S") + ".log"
logger.outputters << Log4r::FileOutputter.new('app-file', :↩
    filename => log_path)

run UserApp.new
```

```
$ mkdir logs
```

File Logging: Tiny App, cont.

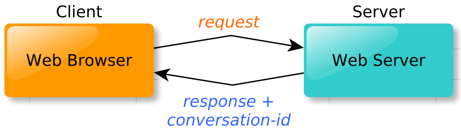
```ruby
# ...

get "/hello_form" do
  logger = Log4r::Logger["app"]
  logger.info "Entered /hello_form route"
  @title = "Hello Stored User"
  haml :form2, :layout => true
end

# ...
```

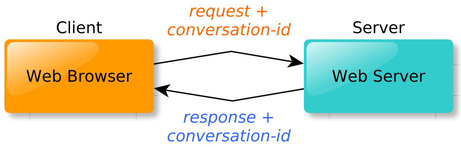
```
$ cat logs/20130731-233135.log
 INFO app: Entered /hello_form route
```

Sessions

- HTTP is stateless
- Sessions allow us to have state between requests
- E.g., the app should *remember*:
  - user logins
  - user adding items in the shopping cart
  - ...
- And has to remember them for multiple clients/conversations
- Implementation
  - HTTP session token
  - Via cookies or GET/POST back to the server

Sessions, cont.





Sessions: Example

```ruby
require 'sinatra'

configure do
  enable :sessions
end

get "/set" do
  session[:user] = "Kostas"
  "User set in session..."
end

get "/get" do
  "User is: #{session[:user]}"
end
```

MIME Types and Attachments

- Multipurpose Internet Mail Extensions (MIME)
  - initially for emails
  - used to describe the content type
- Widely used, e.g. in HTTP

Sessions: Example 1

```
require 'sinatra'

configure do
  mime_type :gif, 'image/gif'
end

get "/user.?:format?" do |format|
   if format == 'gif'
     send_file "screen_bug.gif"
   else
     "Default is text/html"
  end
end
```

Sessions: Example 2

```ruby
require 'sinatra'

get "/user.?:format?" do |format|

  if format == 'gif'
    content_type :gif
    send_file "screen_bug.gif"
  else
    "Default is text/html"
  end
end
```

Sessions: Example 3

```
require 'sinatra'

get "/user.?:format?" do |format|
  if format == 'gif'
    content_type :gif
    send_file "screen_bug.gif"
  else
    "Default is text/html"
  end
end

get "/download/user" do
  attachment "filename.gif"   # name of downloaded file
  send_file "screen_bug.gif"  # name in file system
end
```

Error Handling

- HTTP Status Codes
    - 1xx: Informational
    - 2xx: Success
    - 3xx: Redirection
    - 4xx: Client Error
    - 5xx: Server Error
- Example error codes
    - 403: Forbidden
    - 404: Not Found
    - 500: Internal Server Error
    - ...
- Our app should properly handle those errors

Error Handling: Example 1

```ruby
require 'sinatra'

configure { disable :show_exceptions }

get "/" do
  "Works"
end

not_found do
  "Not found!"
  # haml :"404", layout => true
end

error do
  "Error!"
  # haml :error, layout => true
end

get "/500" do
  raise StandardError, "Forced error"
end
```

Error Handling: Example 2

```ruby
require 'sinatra'

configure { disable :show_exceptions }

error do
  "Error!"
  # haml :"404", layout => true
end

error 403 do
  "A 403 Error!"
  # haml :"403", layout => true
end

get "/403" do
  halt 403
end

get "/500" do
  raise StandardError, "Forced error"
end
```

More Configuration

```ruby
configure do
  set :environment, ENV["RACK_ENV"]
  enable :logging
  set :sessions, true
  set :public_folder, "public"
  disable :show_exceptions
  set :root, "."
  set :static
  set :views, "views"

  # Custom
  set :key, "value"
end
```

```ruby
get "/" do
  settings.key
end
```

Filters

```ruby
require 'sinatra'

before do
  logger.info "Entered 'before' block."
end

after do
  logger.info "Entered 'after' block."
end

get '/' do
  logger.info "Entered '/' route."
  "Testing filters, please consult the log."
end
```

Filters

```
$ ruby filters.rb
[2013-08-01 20:22:23] INFO  WEBrick 1.3.1
[2013-08-01 20:22:23] INFO  ruby 1.9.3 (2012-12-25) [x86_64-linux]
== Sinatra/1.4.3 has taken the stage on 4567 for development with backup from WEBrick
[2013-08-01 20:22:23] INFO  WEBrick::HTTPServer#start: pid=8492 port=4567
I, [2013-08-01T20:22:46.326070 #8492]  INFO -- : Entered 'before' block.
I, [2013-08-01T20:22:46.326424 #8492]  INFO -- : Entered '/' route.
I, [2013-08-01T20:22:46.326644 #8492]  INFO -- : Entered 'after' block.
127.0.0.1 - - [01/Aug/2013 20:22:46] "GET / HTTP/1.1" 200 40 0.0054
localhost - - [01/Aug/2013:20:22:46 EEST] "GET / HTTP/1.1" 200 40
- -> /
```