

Sinatra

Friday, February 10, 12

1

Not Frank Sinatra

- DSL for writing web applications in Ruby

EdgeCase 

Friday, February 10, 12

2

It's this easy

```
require 'sinatra'
```

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

BOOM!

WEBSERVER!

OK, there's more

Routes

HTTP method with URL matcher

HTTP GET

```
get '/' do
  # show something
end
```

HTTP POST

```
post '/' do
  # create something
end
```

HTTP PUT

```
put '/' do
  # replace something
end
```

HTTP DELETE

```
delete '/' do
  # remove something
end
```

named parameters

```
get '/:name' do
  "Hello #{params[:name]}"
end
```

matching regular expressions

```
get %r{/hello/([\w]+)} do
  "Hello #{params[:captures].first}"
end
```

optional parameters

```
get '/posts.?:format' do
  # GET /posts
  # GET /posts.json
  # GET /posts.xml
end
```

Static Files

served from `./public`

or manually

```
set :public_folder, File.dirname(__FILE__) + '/static'
```

Views / Templates

EdgeCase }
SOFTWARE ARTISANS

EdgeCase }
SOFTWARE CONSULTANTS

EdgeCase }
SOFTWARE CONSULTANTS

```
get '/' do
  # render views/index.html
  haml :index
end
```

custom layouts

```
get '/' do
  haml :index, :layout => post
end
```

render views/index.html embedded in views/post.html

default is views/layout.html

embedded templates

```
get('/:name' do
  erb 'Hello <%= params[:name] %>'
end
```

Filters

before filters

```
before '/protected/*' do  
  authenticate!  
end
```

after filters

```
after do  
  puts response.status  
end
```

Sessions

```
enable :sessions

get '/' do
  session[:answer] = 42
end

get '/answer' do
  session[:answer].to_s
end
```

Error Handling

raise Sinatra::NotFound

```
not_found do
  "This page cannot be found."
end
```

error

```
error do
  'There was an error : ' + env['sinatra.error'].name
end
```

error codes

```
error 403 do
  'Access Verboten!'
end

error 400..500 do
  'Uh oh!'
end
```


custom errors

```
error MyCustomError do
  'There was an error : ' + env['sinatra.error'].name
end
```

Modular Applications

switching to Modular app

```
# my_app.rb
require 'sinatra/base'

class MyApp < Sinatra::Base
  # ... app code ...

  # start server if executed directly
  run! if app_file == $0
end
```

config.ru

config.ru with classic app

```
# config.ru
require './app'
run Sinatra::Application
```

config.ru with modular app

```
# config.ru
require './my_app'
run MyApp
```

Resources

- www.sinatrarb.com/