relevance

Sinatra

Glenn Vanderburg
Relevance, Inc.
glenn@thinkrelevance.com

www.sinatrarb.com

Sinatra

README
DOCUMENTATION
CONTRIBUTE
BLOG
CODE
ABOUT

Put this in your pipe

```
require 'rubygems'
require 'sinatra'
get '/hi' do
  "Hello World!"
end
```

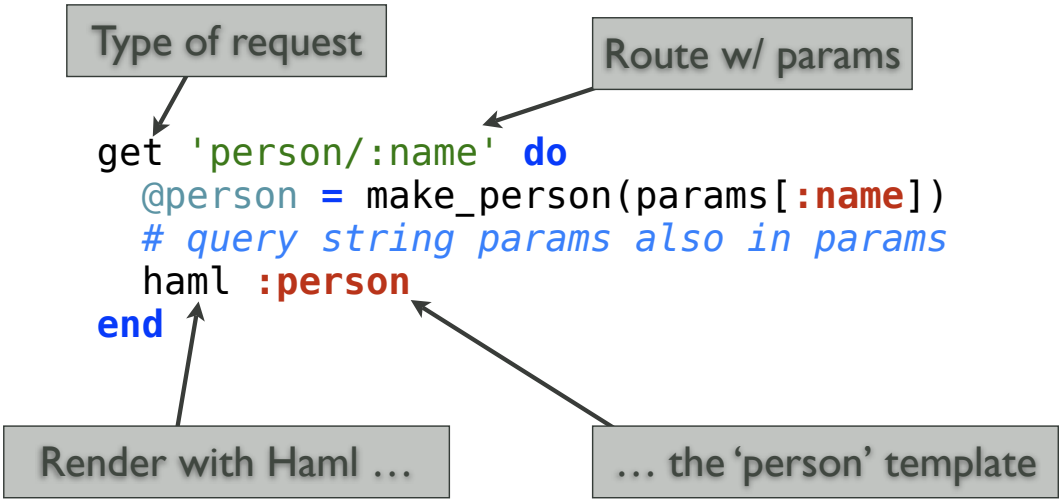
and smoke it

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

- Very small web framework
  - ≈1300 lines of Ruby
- Just the web:
  - HTTP dispatching
  - Request parsing
  - Response generation
  - No persistence
- "Camping without the LSD"

- Built on Rack
- Not a toy!
  - Quick-and-dirty web apps
  - Apps of mostly REST services
  - Example: Castronaut
    http://github.com/relevance/castronaut
- Uses Builder, Erb, Haml/Sass for templating
  - Docs oriented toward Haml

Basics



Configure

```
configure do
  set :conn, create_db_connection
  set :cache = {}
  # other things to do only once
end

# access as options.conn, options.cache
```

relevance

Helpers

```
helpers do
  def format_person(person)
    "#{person.first.first}. #{person.last}"
  end
end
```

relevance

Templates

- Can be in `views/`
- Or inline:

```
template :greeting do
  'Hi, <%= @person.first %>!'
end

__END__

@@ farewell
Goodbye, <%= @person.first %>.
```


More …

- Post, Put, Delete using `post`, `put`, and `delete`.
- Before filters:

```
before do
  @note = 'Hi!'
  request.path_info = '/foo/bar/baz'
end
```

- Filters or actions can `halt` a request:

```
halt
halt 'please try again'
halt 401, 'go away!'
```

More …

- Actions can `pass` to later actions:

```
get '/guess/:who' do
  pass unless params[:who] == 'Frank'
  "You got me!"
end

get '/guess/*' do
  "You missed!"
end
```

- Support for testing using test/unit, test-spec, Rspec, Bacon

Environments

```
configure :production do
  # ...
end

configure :development, :test do
  # ...
end
```

```
$ ruby bing.rb -e production
```

```
$ RACK_ENV=production ruby bing.rb
```

Error Handling

```
not_found do
  'This is nowhere to be found'
end

error do
  'Sorry there was a nasty error - ' +
      env['sinatra.error'].message
end

error MyCustomError do
  'So what happened was...' + env['sinatra.error'].message
end
```


Rack

- Standard Ruby web server interface: http://rack.rubyforge.org/
- Good middleware interface
- Comes with support for logging, debugging, URL routing, authentication, and session handling.

Rack

```
use Rack::Lint

use Rack::Auth::Basic do |username, password|
  username == 'admin' && password == 'secret'
end

require 'my_custom_middleware'
use MyCustomMiddleware
```

relevance

Why Sinatra's Cool

- The Ruby community refactors itself.
- Throwaway web apps.
- Less magic.
- Small, service-oriented apps.
  - Ripe for integrating with others.
- For an example of a "big" Sinatra app, look at Castronaut.

relevance

relevance

Sinatra

Glenn Vanderburg
Relevance, Inc.
glenn@thinkrelevance.com

Talk and sample code at:
http://github.com/glv/sinatra_talk