wordnik

Scalatra 2.2

NOVEMBER 2012

Short introduction And overview of New features

What?

- Routing
- Params
- Json4s
- Akka
- Atmosphere
- Commands

Hello, Scalatra

```
class HelloWorld extends ScalatraServlet {
  get("/hello/:name") {
    <h1>Hello, {params("name")}</h1>
  }
}
```

Hello, Scalatra: Routes

```
class HelloWorld extends ScalatraServlet {
 get("/hello/:name") {
   <h1>Hello, {params("name")}</h1>
 }
}
```

- Matches GET requests to /hello/*
- Captures named parameters from the route (here, :name).
- Matching requests are dispatched to the action.

Hello, Scalatra: Actions

```
class HelloWorld extends ScalatraServlet {
  get("/hello/:name") {
    <h1>Hello, {params("name")}</h1>
  }
}
```

- When a request matches a route, the action is run.
- A response is generated from the action's result. Supports XML literals, Byte arrays, Strings, Files, InputStreams.
- Rendering of action results is customizable.

Hello Scalatra: ScalatraServlet

```
class HelloWorld extends ScalatraServlet {
  get("/hello/:name") {
    <h1>Hello, {params("name")}</h1>
  }
}
```

- ScalatraServlet provides the HTTP DSL.
- Scalatra 2.x is built on Java Servlets, the standard web server API for the JVM.

Why Scalatra?

- Based on Sinatra, which I like to call "the universal web DSL". Clones exist in:
  - Ruby
  - PHP
  - Python
  - JavaScript
  - Java
  - F#
  - Erlang
  - Haskell
  - Clojure
  - Bash
  - Many others

Scalatra Philosophy

- Stateless by default, sessions if you need them.
- Lean by default, integrations if they're popular.
  - Scalate
  - JSON4s

- Akka
- Atmosphere
- Embrace standards
- HTTP
- JVM
- Scala
- Beginner friendly.
- Scale with your app and your expertise.

Routing in depth: Methods

```
get("/hello/:name") { "Hi, "+params("name") }
```

- Routes start with an HTTP method.
  - Get / Head / Options / Connect
  - Post / Put / Patch / Delete / Trace
- Advanced usage: can call at anytime to register routes on the fly.

Routing in depth: Params

```
get("/hello/:name") { "Hi, "+params("name") }
```

- Matches the servlet path.
  - Protip: compose large apps with multiple servlets
- Declare route params with a leading ':’
  - Matches everything up to next '/', '?', or '#'

Routing in depth: Params

Request	Matches?	params("name")
GET /hello	No	
GET /hello/Wordnik	Yes	Wordnik
GET /hello/Irvine	Yes	Irvine
GET /hello/Irvine/CA	No	

Routing in depth: Splats

```
get("/hello/*") { multiParams("splat").mkString(",") }
```

- Splats get stored in the `captures` param.
- If you expect multiple values, you can access it through multiParams.
  - params is just a convenient view of multiParams

Request	multiParams("captures")
GET /hello/	Seq()
GET /hello/Wordnik	Seq("Wordnik")
GET /hello/Wordnik/Cloud/Team	Seq("Wordnik", "Cloud", "Team")

Routing in depth: regex

```
get("""/hello/(\w+)""".r) { "Hello, "+params("captures") }
```

- Use if you have a real hairy route that standard syntax can't handle.
- No named groups yet. Java didn't add named regexes until version 7!

Routing in depth: booleans

```
def userAgent = request.getHeader("User-Agent”)

get("/hello/:name", !userAgent.contains("Lynx")) {
   status = 403
   "You and your fancy, schmancy browser. Get out.” }
```

- Both paths and boolean expressions are converted to route matchers. *Implicitly*
- A route matches if and only if all route matchers match.

Actions in depth: return value

- The return value of an action becomes the body of the response.

Return type	... does
Array[Byte]	renders to output stream
InputStream	copied to output stream
File	written to output stream
Unit	does nothing
Int	no body, sets status
Any	converts to String, renders Any

Actions in depth: render pipeline

- It's just a
- Override

```
protected def renderPipeline: RenderPipeline = {
  case bytes: Array[Byte] =>
    response.outputStream.write(bytes)
  case is: java.io.InputStream =>
    using(is) { util.io.copy(_, response.outputStream) }
}
```

- It's just a partial function.
- Override it. Use orElse to delegate to the default.

Actions in depth: mutable

wordnik

Actions in depth: mutable

```
get("/hello/:name") {
  status = HttpServletResponse.SC_GONE
  contentType = "text/html”
  <h1>Goodbye, cruel world</h1> }
```

- Status and headers are typically set in an imperative fashion.
- Response is thread local. Relax.

Commands

- Capture input
- Validate input

Integrations: JSON4S

- Serialize case classes to json
- Handles xml as well as json

Integrations: Akka

- Takes advantage of servlet 3.0 async support

Integrations: Atmosphere

- WebSocket
- Server side events
- Like socket.io but not socket.io