

scalaz.http

scalaz, http, sbt

we put big ideas into small devices

overview

- ▶ Talk
- ▶ Demo
- ▶ Talk

scalaz 4

what is it?

general functions that are
not available in the core
API

more?

scalaz 4

HTTP, database, actors, type classes, arrows, applicative functors, B-K Tree, Kleisli, monoid, memoisation.

Doesn't work on Scala 2.8 yet, branch is in development.

web?

slinky \Rightarrow scalaz.http

scapps ⇒

scalaz.http.scapps

slinky

slinky

- ▶ 2nd generation API
- ▶ Low-ish level APIs
- ▶ Models HTTP; request, response, status, method, headers, etc.
- ▶ Can use directly, or with higher level constructs (scapps)

code?

pattern matching

```
request match {  
  case MethodPath(GET, "/hello") =>  
    OK(ContentType, "text/html") << transitional << say("hello")  
  case MethodPath(GET, "/say") => {  
    val phrase = (request ! "phrase") > (_.mkString)  
    phrase ? (BadRequest, OK) << strict << say(phrase | "Pass phrase")  
  }  
}
```

routing

```
val routes: Kleisli[Option, Request[Stream], Response  
[Stream]] =  
  List(  
    exactPath("/") >=> GET >=> webRoot _,  
    startsWith("/api") >=> List(  
      exactPath("/") >=> GET >=> apiUsage _,  
      startsWith("/register") >=> POST >=> apiRegister _,  
      startsWith("/registrants") >=> GET >=> apiRegistrants _,  
      startsWith("/search") >=> GET >=> apiSearch _  
    )  
  )
```

```
def route(implicit request: Request[Stream],  
  servletRequest: HttpServletRequest) = routes(request)
```

“actions”

```
def apiUsage(request: Request[Stream]) = {  
  implicit val r = request  
  val content =  
    <div>  
      <h2>Register</h2>  
      <p>Endpoint: {a("/api/register", "/api/register")}</p>  
    </div>  
  ConferenceContent("Conference API", content)  
}
```

rest

```
def routes : Kleisli[Option, Request[Stream], Response[Stream]] = {  
  dir("/") >=> List(  
    GET >=> List(  
      "/" >=> index _,  
      "/new" >=> formForCreate _,  
      dir("/") >=> withParam("edit") >=> (_.asItemRequest) >=> formForEdit _,  
      dir("/") >=> (_.asItemRequest) >=> item _  
    ).sumList,  
    PUT >=> dir("/") >=> (_.asItemRequest) >=> updateItem _,  
    POST >=> "/" >=> createItem _,  
    DELETE >=> dir("/") >=> (_.asItemRequest) >=> deleteItem _  
  ).sumList  
}
```

work

```
request match {  
case Path(DbId(db, id)) =>  
  val couched = ...  
  couched(id) {  
    case (OK.toInt, ri, Some(entity)) =>  
      id match {  
        case ("style.css") => ...  
        case (id) => ...  
      }  
    case (NotModified.toInt, ri, _) => Some(cache_heds(ri, NotModified))  
    case (NotFound.toInt, _, _) => None  
  }  
}
```

html

```
val html =  
  <html xmlns="http://www.w3.org/1999/xhtml">  
  <head><title>Ze head</title></head>  
  <body>  
    <p>Ze body</p>  
  </body>  
  </html>  
OK(ContentType, "text/html") << strict << html
```

json

```
val it = Map('foo -> "Bar", 'baz -> "Quux")  
OK(ContentType, "application/json") << it.jsonString
```

demo

what's next?

What's next?

- ▶ Scalaz 4-ise
- ▶ Flesh out scapps style
 - ▶ Routing - <http://bitbucket.org/nkpart/scapps/.../>
RestfulRoute.scala
- ▶ Scala 2.8
- ▶ Database

Code

- ▶ Project: <http://code.google.com/p/scalaz/>
- ▶ Demo: <http://github.com/tomjadams/slinky-demo/tree/master>
- ▶ Scapps proto: <http://bitbucket.org/nkpart/scapps/>
- ▶ Scala 2.8: <http://scalaz.googlecode.com/svn/branches/scala-2.8/>
- ▶ Real example: <http://technically.us/code/x/sling-shot/>

Contact

Tom Adams

Chief technologist & co-founder

tom@mogeneration.com

<http://mogeneration.com/>

mo
gen
era
tion