Hands on Scala.js

Li Haoyi, PNWScala 14 Nov 2014
Hands on Scala.js: Agenda

• Intro to Scala.js
• Interactive Web Pages
• Cross-platform libraries
• Client-server integration
• Wrap Up
Intro to Scala.js

- Intro to Scala.js
- Interactive Web Pages
- Cross-platform libraries
- Client-server integration
- Wrap Up
Intro to Scala.js

- Who
- What
- Where
- When
- Why
Intro to Scala.js: Who

- Li Haoyi
  - I work at Dropbox
  - Come talk to me about legacy CoffeeScript code
  - ~10 commits in scala-js/scala-js

- @sjrd/@gzm0
  - Real authors
  - ~2000 commits in scala-js/scala-js
Intro to Scala.js: What

- Scala -> Javascript Compiler
  - Run Scala code in the web browser!

- Respectable Performance
  - 1-3x slower than raw JS, 10x slower than Scala-JVM
    - Probably still 5x faster than python
  - 150-400kb non-gzipped executables
    - Mostly Scala’s bloated collections library
def main() = {
  var x = 0
  while(x < 999){
    x = x + "2".toInt
  }
  println(x)
}
ScalaJS.c.LExample$.prototype.main__V = (function() {
    var x = 0;
    while ((x < 999)) {
        x = ((x + new ScalaJS.c.sci_StringOps().init___T(ScalaJS.m.s_Prefdef().augmentString___T__T("2"))
            ).toInt___I()) | 0)
    };
    ScalaJS.m.s_Prefdef().println__O__V(x)
});
Intro to Scala.js: What

be.prototype.main=function()
{
    for(var a=0;999>a;)
        a=a+(new de).g(S(L(),"2")).ne()|0;
    ee(); L();
    var b=F(fe); ge();
    a=(new he).g(w(a)); b=bc(0,J(q(b,[a])));
    ie(bc(L()),J(q(F(fe),[je(ke(ge()).Vg),b]))))
}
Intro to Scala.js: Where

- http://www.scala-js.org/
- https://github.com/scala-js/scala-js
- https://groups.google.com/forum/#!forum/scala-js
- http://www.scala-js-fiddle.com/
Intro to Scala.js: Where

- Scala.js extends the reach of your Scala
  - Play Websites
  - Node.js modules
  - Chrome Extensions
  - Autodesk Fusion plugins
  - Firefox OS?

- Not just the JVM!
Intro to Scala.js: When

- **June 2013**: Announced at Scaladays
- **Sept 2013**: I got involved in
- **Dec 2013**: v0.1 released at ScalaXchange
- Working towards v1.0 now
Intro to Scala.js: Why
Intro to Scala.js: Why
Good Parts Reconsidered

- I stopped using new years ago.
- I have stopped using Object.create.
- I have stopped using this.
- I have stopped using null.
- I have stopped using falsiness.
Intro to Scala.js: Why
Intro to Scala.js: Why

- Javascript is =(  
  - Rather verbose  
  - *Too* flexible  
  - Hard to write tools  
  - Scary to refactor

- Scala is =)
Interactive Web Pages

- Intro to Scala.js
- Interactive Web Pages
- Cross-platform libraries
- Client-server integration
- Wrap Up
Web Page Takeaways

● Scala.js works
  ○ Conception
  ○ Debugging
  ○ Publishing

● HTML generation using Scalatags rocks

● Working directly with the DOM is much easier with types
Canvas Demos

- Retro Games
- Roll
- Ray Tracer
Cross-platform libraries

- Intro to Scala.js
- Interactive Web Pages
- Cross-platform libraries
- Client-server integration
- Wrap Up
Cross-platform libraries

- **Scalatags**
  - HTML Generation
- **uTest**
  - Unit Testing
- **uPickle**
  - Serialization
- **Scala.Rx**
  - Change Propagation
- **Scalaz**
  - Hardcore FP
- **Shapeless**
  - Hardcore Genericity
- **Monocle**
  - Lenses
- **Parboiled2**
  - Parser Combinators
```scala
private def gitHash = sys.process.Process("git rev-parse HEAD").lines._1.head

  organization := "org.scalaz",
]

  scala.scalajs.sbtplugin.ScalaJSPartitioner.scalaJSBuildSettings ++
  Seq[Sett](
    organization := "com.github.japgolly.fork.scalaz",
  )

scalaVersion := "2.10.4",
crossScalaVersions := Seq("2.9.3", "2.10.4", "2.11.2")
```

```scala
addSbtPlugin("com.typesafe.sbt" % "sbt-osgi" % "0.7.0")
addSbtPlugin("com.eed3si9n" % "sbt-buildinfo" % "0.3.1")
addSbtPlugin("com.eed3si9n" % "sbt-unidoc" % "0.3.1")
+addSbtPlugin("org.scala-lang.modules.scalajs" % "scalajs-sbt-plugin" % "0.5.3")
```
<table>
<thead>
<tr>
<th>Can Use</th>
<th>Can’t Use</th>
</tr>
</thead>
<tbody>
<tr>
<td>java.lang.*</td>
<td>j.l.Thread, j.l.Runtime</td>
</tr>
<tr>
<td>scala.*</td>
<td>s.c.parallel, s.tools.</td>
</tr>
<tr>
<td>Macros: upickle, async</td>
<td>Reflection: pickling, akka</td>
</tr>
<tr>
<td>Scala: Scalaz, Scalatags</td>
<td>Java: Scalatest, Scalate</td>
</tr>
<tr>
<td>XMLHttpRequest, DOM, WebGL, Canvas</td>
<td>Netty, Spray, Swing, OpenGL</td>
</tr>
<tr>
<td>IntelliJ, SBT</td>
<td>Yourkit, VisualVM</td>
</tr>
</tbody>
</table>
Live Coding

Cross-Platform Library

https://github.com/lihaoyi/utest-example-module
Library Takeaways

- Cross-platform libraries targeting JS/JVM work
- Code that works on both platforms can be shared
  - Even tests!
- Code specific/optimized to each platform can be provided separately
Client-Server Integration

- Intro to Scala.js
- Interactive Web Pages
- Cross-platform libraries
- **Client-server integration**
- Wrap Up
Live Coding

Client-Server Integration

https://github.com/spray/spray-template
Client-Server Takeaways

- Wiring Scala.js into any existing project is trivial
- Sharing code between Client/Server is Awesome
  - Constants, algorithms, data-structures, libraries, etc.
- Type-safety makes shared code amazing
- The whole setup actually works!
Wrap Up

- Intro to Scala.js
- Interactive Web Pages
- Cross-platform libraries
- Client-server integration
- **Wrap Up**
Scala.js works!

- Usable for all sorts of projects
- Experience is great
- Future is promising
Things that are Not Great

● Small community
  ○ It’s new, after all

● Scala compiler is slowwww, std lib bloated
  ○ Incremental compilation/DCE helps, but still...

● No big corporate backing
  ○ Just two guys and some extras

● Some rough edges
  ○ Arguably fewer than Javascript itself ^_^
The Future is Now

- Scala.js provides *multiple* web-dev holy-grails
  - Shared code between Client/Server
  - Checked interfaces between Client/Server
  - Sane, shared language between Client/Server
  - Whole-program-checked Client/Server

- Not the future, but *today*
  - Actually ~6 months ago
The Future is Now

```javascript
javascript> [ '10', '10', '10', '10' ].map(parseInt)
[10, NaN, 2, 3]
```

```scalajs
scalajs> List("10","10","10","10").map(parseInt)
List(10, 10, 10, 10)
```
Hands on Scala.js