Rhino & RingoJS: JavaScript on the JVM

Hannes Wallnöfer
http://hns.github.com
@hannesw
"Overall, JavaScript as a system programming language feels a lot like Lisp must have for the programming generation before mine: **minimal syntax, very powerful and orthogonal core abstractions**, and (dare I say it) not much type-checking or busy-work to get in your way."

C. Scott Ananian (litl)
http://cananian.livejournal.com/58744.html
Rhino and RingoJS

- **Rhino**
  - Started at Netscape in 1997
  - Part of Navigator port to Java
  - Mozilla project

- **RingoJS**
  - Started by me in 2009
  - Provide the parts missing for real world software development, especially web applications
Rhino and RingoJS
Rhino

- Robust, optimized codebase
- but showing its age
- very complete
  - interpreter mode
  - compiled mode
  - debugger
  - follows ECMAScript spec rigorously
  - implements JS 1.7 (almost 1.8)
  - implements most of ECMAScript 5
Optimimization

- Rhino used to be one of the faster JVM languages
- But not much movement in the last few years
- Browser JS implementations have roared past Rhino
InvokeDynamic

- Rémi probably told you all about it
- John Rose (Oracle): "Thundering Rhinos" at JavaOne 2010

http://blogs.sun.com/jrose/entry/javaone_in_2010

4x speedup of Richards benchmark (part of V8 benchmark suite) within 50% of V8 by manually editing bytecode
JavaScript objects...

- ... are hashtables
  
  ```javascript
  var x = {foo: 3}
  
  no notion of classes
  x["baz"] = 7;
  ```

- ... have prototypes
  
  ```javascript
  function Foo() {...}
  Foo.prototype.bar = 3;
  var y = new Foo();
  ```

  ```javascript
  var y = Object.create
  ```
How to implement JS objects?

- Obvious solution: hashtables
  - works, but rather slow
- Ideally JS properties are mapped to Java fields
  - easy for the simple case
  - fails for common cases (multiple scripts, dynamic code...)
- Rhino hack: idgen
idgen – custom property handling in Rhino

- Used for built-in types (Object, Array, Function)
- Requires pre-processing of code:

  ```
  // #generated# Last update: 2007-05-09 08:15:15 EDT
  L0: { id = 0; String X = null; int c;
      L: switch (s.length()) {
          case 4: X="name"; id=Id_name; break L;
          case 5: X="arity"; id=Id_arity; break L;
          case 6: X="length"; id=Id_length; break L;
          case 9: c=s.charAt(0);
                  if (c=='a') { X="arguments"; id=Id_arguments; }
                  else if (c=='p') { X="prototype"; id=Id_prototype; }
                  break L;
      }
      if (X!=null && X!=s && !X.equals(s)) id = 0;
      break L0;
  }
  // #/generated#
  ```
- Not faster anymore
Rhino-opt

Experimental Rhino branch
https://github.com/hns/rhino-opt

Various branches:
- companion-scopes
- native-callsites
- non-object-this
Anatomy of a running JS program

Script 1
```javascript
var y = 3;

function A() {
  var x = 2;
  function B() {
    return x * y;
  }
  return B();
}
```

Function Scope
```javascript
x * y
```

parent scope

Function Scope
```javascript
x = 2
```

parent scope

Global Object
```javascript
y = ?
```

Script 2
```javascript
y = "foo";
```
function outer(x) {
    var y;
    for (var i = 0; i < 10000000; i++) {
        y = inner();
    }
    function inner() {
        return x;
    }
    return inner();
}
Unfortunately, improvement is not very relevant for Google V8 Benchmark, which is heavy on objects/classes, not scopes.
Generic mapping of JS objects to Java class

- Use "mixed" approach
  - Generate custom Java classes for JS objects
  - But still allow dynamic property access
- Store Rhino property slots as Java fields
- Generate bytecode to access Java field if defined
RingoJS

Adds features to Rhino for real-world application development

- Modules
- Packages
- Filesystem
- Testing
- HTTP
- and much more
CommonJS

- Started by Kevin Dangoor in 01/2009
- Good initial progress
- Stalled above sync/async divide
- Ratified standards for Modules, Packages, Web Server API
- Proposals for binary data, IO, filesystem
Example: reading lines from a file

```javascript
var fs = require("fs");
var txt = fs.read("git/ringojs/README.md");
```
Example: reading lines from a file

```javascript
var fs = require("fs");
var file = fs.open("git/ringojs/README.md");

var lines = [line for each (line in file)];

lines.filter(function(line) {
    return line.indexOf("build") > -1
}).map(String.toUpperCase).join("\n");
```
POSIX

- Borrowed functionality from JRuby (jnr-posix) - thank you!
- Provides features for getting/setting file permissions and ownership, handling symbolic links.
A web application is a JavaScript function that takes a request object and returns a response object:

```javascript
function app(request) {
    return {
        status: 200,
        headers: {},
        body: ["hello world"]
    };
}
```

Similar to Rack (Ruby) and WSGI (Python), not Java Servlets.
Asynchronous JSGI

- Non-standard extension
- Works with Jetty Continuations/Servlet 3.0

```javascript
function app(request) {
  var response = defer();
  setTimeout(function() {
    response.resolve({
      status: 200,
      headers: {},
      body: ["hello world"]
    });
  }, 2000);
  return response;
}
```
Accessing Java

- LiveConnect: direct mapping between Java and JavaScript

  ```javascript
  var file = new java.io.File("test.txt")
  f.exists()
  f.getName()
  ```

- Easy, natural mapping for 97% of cases

- Except:
  - method overloads
  - creating Java arrays
obj = {
    run: function () {
        print("\nrunning");
    }
}

r = new java.lang.Runnable(obj)
t = new java.lang.Thread(r)
t.start()
Implementing Java interfaces

impl = function () {
    print("running");
}
new java.lang.Thread(impl).start()
JavaAdapter

- Allows subclassing in addition to implementing interfaces
- Allows implementing multiple interfaces
- More verbose

```java
JavaAdapter(
    java_intf_or_class,
    [java_intf, ..., java_intf],
    javascript_object)
```

- Classes must have zero-arg constructor
Using Java's Security Framework

- Scripts can have CodeSource
- Scripts can execute PrivilegedAction on behalf of untrusted code

```javascript
privileged(function() {
    ...
});
```
- Unsolved issues due to dynamic nature of JavaScript
Thanks!