

Scala.js networking made easy

Olivier Blanvillain

PROGRAMMING METHODS LABORATORY, EPFL

January 26, 2015

This Presentation

1. Transport library
2. Latency compensation framework
3. Example: online multiplayer game

Motivation

- Share
- Many JavaScript APIs
- Many network programming models

Diving in...

```
trait Transport {  
  type Address  
  def listen(): Future[Promise[ConnectionListener]]  
  def connect(remote: Address): Future[ConnectionHandle]  
  def shutdown(): Future[Unit]  
}  
  
trait ConnectionHandle {  
  def handlerPromise: Promise[MessageListener]  
  def write(message: String): Unit  
  def closedFuture: Future[Unit]  
  def close(): Unit  
}  
  
type ConnectionListener = ConnectionHandle => Unit  
type MessageListener = String => Unit
```

```
trait Transport {  
  type Address  
  def listen(): Future[Promise[ConnectionListener]]  
  def connect(remote: Address): Future[ConnectionHandle]  
  def shutdown(): Future[Unit]  
}  
  
trait ConnectionHandle {  
  def handlerPromise: Promise[MessageListener]  
  def write(message: String): Unit  
  def closedFuture: Future[Unit]  
  def close(): Unit  
}  
  
type ConnectionListener = ConnectionHandle => Unit  
type MessageListener = String => Unit
```

```
trait Transport {  
  type Address  
  def listen(): Future[Promise[ConnectionListener]]  
  def connect(remote: Address): Future[ConnectionHandle]  
  def shutdown(): Future[Unit]  
}  
  
trait ConnectionHandle {  
  def handlerPromise: Promise[MessageListener]  
  def write(message: String): Unit  
  def closedFuture: Future[Unit]  
  def close(): Unit  
}  
  
type ConnectionListener = ConnectionHandle => Unit  
type MessageListener = String => Unit
```

```
trait Transport {  
  type Address  
  def listen(): Future[Promise[ConnectionListener]]  
  def connect(remote: Address): Future[ConnectionHandle]  
  def shutdown(): Future[Unit]  
}  
  
trait ConnectionHandle {  
  def handlerPromise: Promise[MessageListener]  
  def write(message: String): Unit  
  def closedFuture: Future[Unit]  
  def close(): Unit  
}  
  
type ConnectionListener = ConnectionHandle => Unit  
type MessageListener = String => Unit
```


Targeted Technologies

- WebSocket
- SockJS
- WebRTC

WebSocket

- Introduction

WebSocket Support

IE	Firefox	Chrome	Safari	Opera	iOS Safari	Opera Mini	Android Browser	Chrome Android
			1 5.1					
8			2 6.1					
9	33	37	7		7.1		4.3	
10	34	38	7.1		8		4.4	
11	35	39	8	26	8.1	8	37	39
TP	36	40		27				
	37	41		28				
	38	42						

Availability: ~84%

SockJS

- Introduction


















SockJS, Supported Transports

<i>Transport</i>	<i>References</i>
websocket (rfc6455)	rfc 6455
websocket (hixie-76)	draft-hixie-thewebsocketprotocol-76
websocket (hybi-10)	draft-ietf-hybi-thewebsocketprotocol-10
xhr-streaming	Transport using Cross domain XHR streaming capability (readyState=3).
xdr-streaming	Transport using XDomainRequest streaming capability (readyState=3).
eventsourcing	EventSource .
iframe-eventsourcing	EventSource used from an iframe via postMessage .
htmlfile	HtmlFile .
iframe-htmlfile	HtmlFile used from an iframe via postMessage .
xhr-polling	Long-polling using cross domain XHR .
xdr-polling	Long-polling using XDomainRequest .
iframe-xhr-polling	Long-polling using normal AJAX from an iframe via postMessage .
jsonp-polling	Slow and old fashioned JSONP polling .

WebRTC

- Introduction

WebRTC Support

IE	Firefox	Chrome	Safari	Opera	iOS Safari	Opera Mini	Android Browser	Chrome Android
			5.1					
8			6.1					
9	33 	37 	7		7.1		4.3	
10	34 	38 	7.1		8		4.4	
11	35 	39 	8	26 	8.1	8	37 	39 
TP	36 	40 		27 				
	37 	41 		28 				
	38 	42 						

Availability: ~54%

Transport Implementations

Platform	WebSocket	SockJS	WebRTC
JavaScript	client	client	client
Play Framework	server	server	-
Netty	both	-	-
Tyrus	client	-	-

THANKS!