

Nuts and Bolts of WebSocket

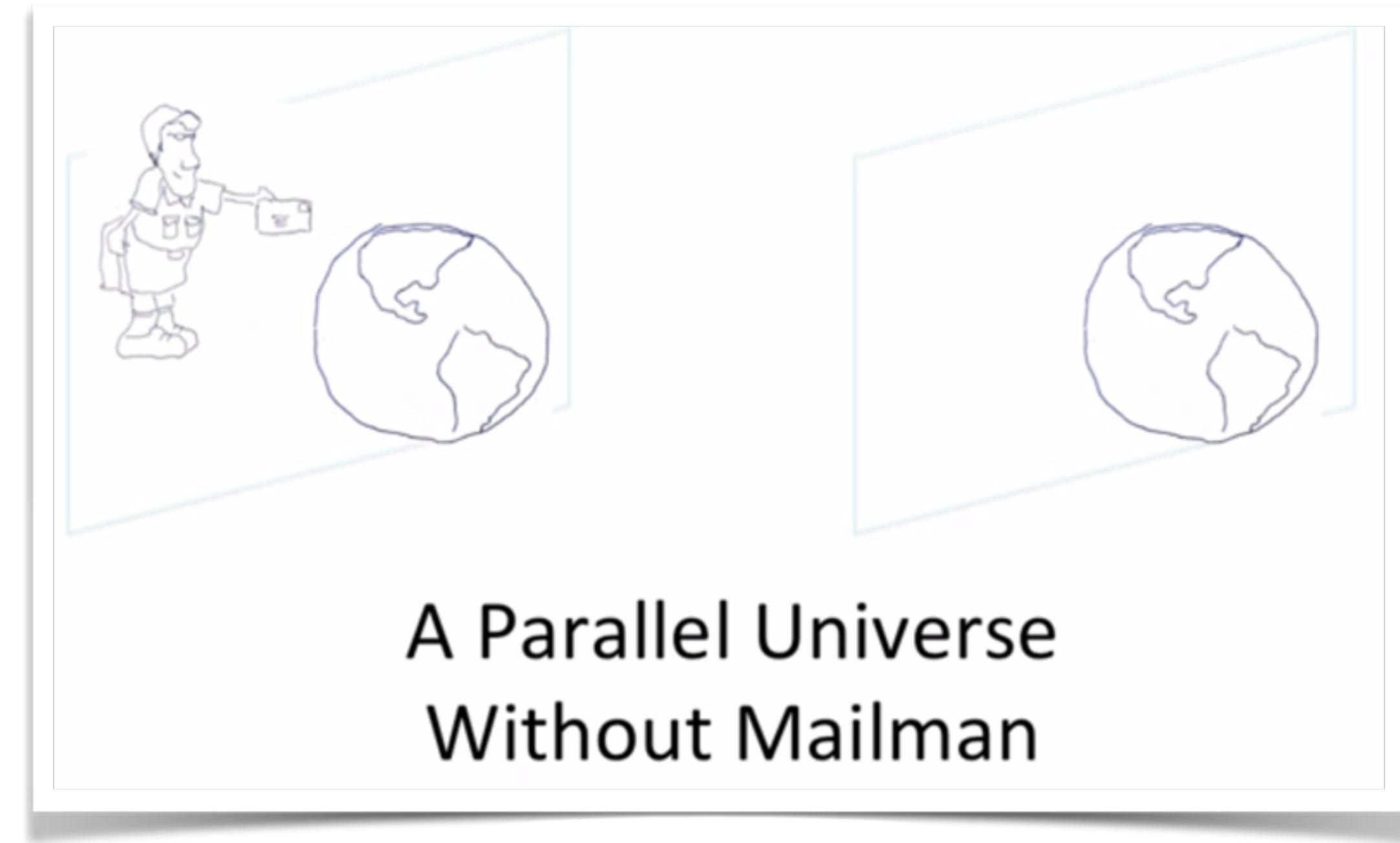
**Arun Gupta, Red Hat, @arungupta
Peter Moskovits, Kaazing, @peterm_kaazing**



Agenda

- Introduction
- WebSocket and Node.js
- WebSocket using JSR 356
 - Server
 - Client
- Securing WebSocket
- Embedded WebSocket
- Load Balance WebSocket
- Pub/Sub over WebSocket
 - STOMP over WebSocket
 - MQTT over WebSocket
- REST and SSE
- Scalability
- Debugging
- Production Tips

The “long” story of WebSocket





“Limitations” of HTTP



“Limitations” of HTTP

- Client-driven

“Limitations” of HTTP

- Client-driven
- Half-duplex

Devoxx
Java
WebSockets



“Limitations” of HTTP

- Client-driven
- Half-duplex
- Verbose



“Limitations” of HTTP

- Client-driven
- Half-duplex
- Verbose
- New TCP connection



“Hello World” HTTP

```
POST /websocket-vs-rest-payload/webresources/rest HTTP/1.1\r\n
Host: localhost:8080\r\n
Connection: keep-alive\r\n
Content-Length: 11\r\n
User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 10_9_1) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/32.0.1700.107 Safari/537.36\r\n
Origin: chrome-extension://hgmloofddfdnphfgcellkdfbjeloo\r\n
Content-Type: text/plain \r\n
Accept: */*\r\n
Accept-Encoding: gzip,deflate,sdch\r\n
Accept-Language: en-US,en;q=0.8\r\n
\r\n
```

```
HTTP/1.1 200 OK\r\n
Connection: keep-alive\r\n
X-Powered-By: Undertow 1\r\n
Server: Wildfly 8 \r\n
Content-Type: text/plain\r\n
Content-Length: 11 \r\n
Date: Fri, 21 Feb 2014 21:27:53 GMT \r\n
\r\n
```



“Hello World” HTTP

```
POST /websocket-vs-rest-payload/webresources/rest HTTP/1.1\r\n
Host: localhost:8080\r\n
Connection: keep-alive\r\n
Content-Length: 11\r\n
User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 10_9_1) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/32.0.1700.107 Safari/537.36\r\n
Origin: chrome-extension://hgmloofddfdnphfgcellkdfbjeloo\r\n
Content-Type: text/plain \r\n
Accept: */*\r\n
Accept-Encoding: gzip,deflate,sdch\r\n
Accept-Language: en-US,en;q=0.8\r\n
\r\n
```

663 bytes

```
HTTP/1.1 200 OK\r\n
Connection: keep-alive\r\n
X-Powered-By: Undertow 1\r\n
Server: Wildfly 8 \r\n
Content-Type: text/plain\r\n
Content-Length: 11 \r\n
Date: Fri, 21 Feb 2014 21:27:53 GMT \r\n
\r\n
```



How WebSocket solves it ?



How WebSocket solves it ?

- Bi-directional (client-driven)



How WebSocket solves it ?

- Bi-directional (client-driven)
- Full-duplex (half-duplex)



How WebSocket solves it ?

- Bi-directional (client-driven)
- Full-duplex (half-duplex)
- Lean protocol (verbose)



How WebSocket solves it ?

- Bi-directional (client-driven)
- Full-duplex (half-duplex)
- Lean protocol (verbose)
- Single TCP connection (new TCP)



What is WebSocket ?



What is WebSocket ?

- Bi-directional, full-duplex, communication channel over a single TCP connection



What is WebSocket ?

- Bi-directional, full-duplex, communication channel over a single TCP connection
- Originally proposed as part of HTML5



What is WebSocket ?

- Bi-directional, full-duplex, communication channel over a single TCP connection
- Originally proposed as part of HTML5
- IETF-defined **Protocol**: RFC 6455



TM

What is WebSocket ?

- Bi-directional, full-duplex, communication channel over a single TCP connection
- Originally proposed as part of HTML5
- IETF-defined **Protocol**: RFC 6455
- W3C-defined **JavaScript API**



How does it work ?



How does it work ?

- Upgrade HTTP to WebSocket (single TCP connection)



How does it work ?

- Upgrade HTTP to WebSocket (single TCP connection)
- Send data frames in both direction (bi-directional)



How does it work ?

- Upgrade HTTP to WebSocket (single TCP connection)
- Send data frames in both direction (bi-directional)
- Send messages independent of each other (full-duplex)



Ш

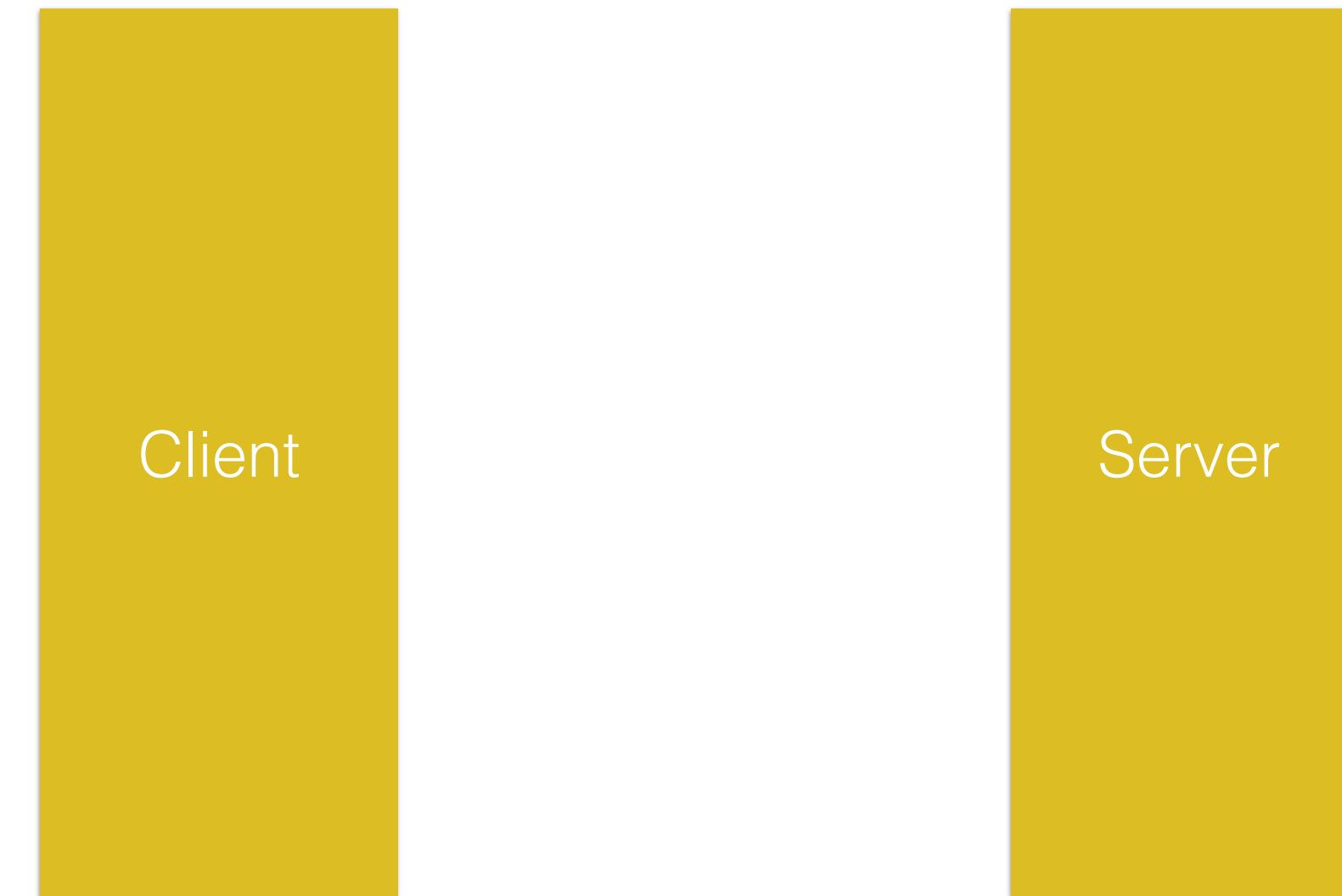


How does it work ?

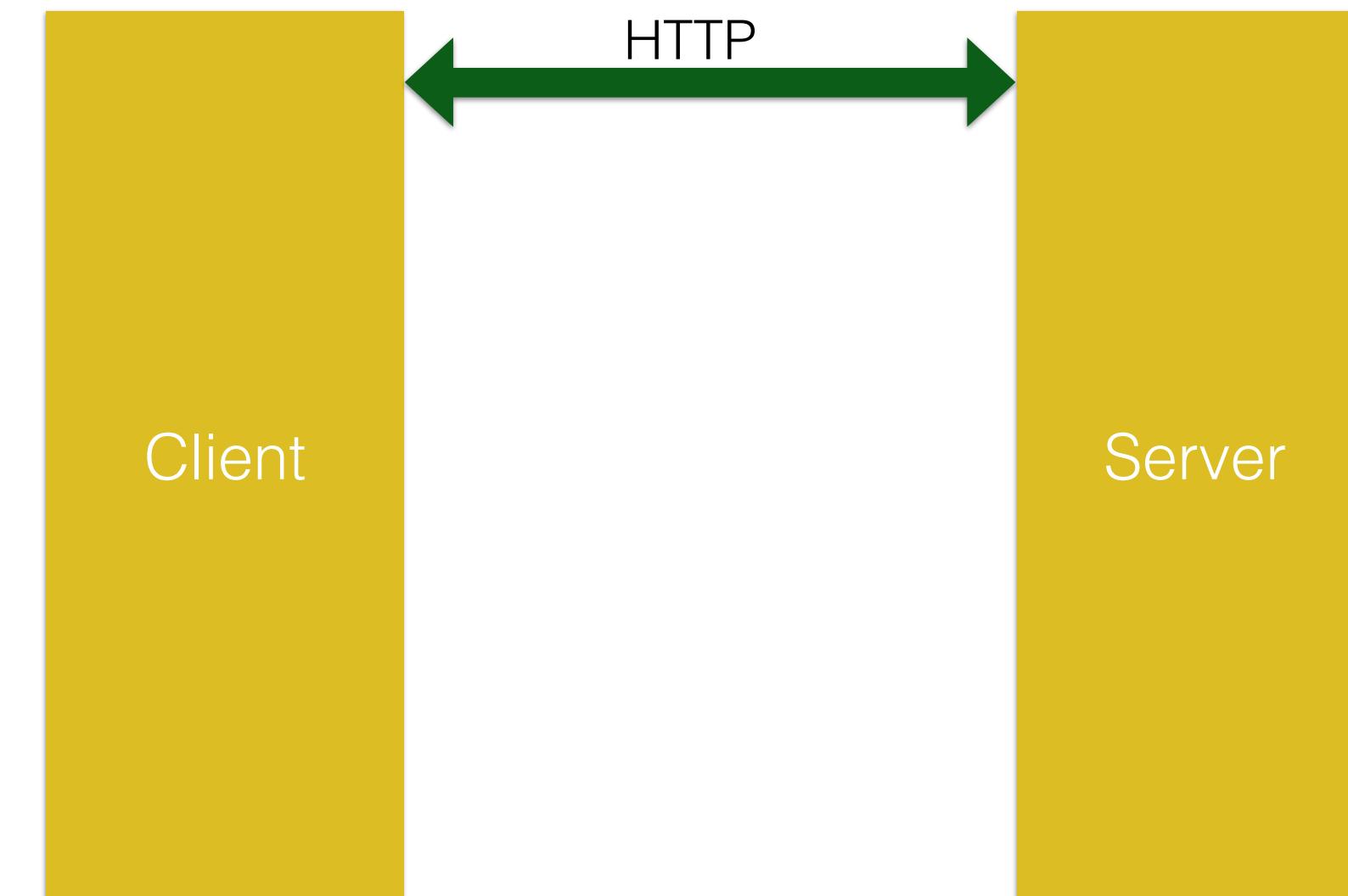
- Upgrade HTTP to WebSocket (single TCP connection)
- Send data frames in both direction (bi-directional)
- Send messages independent of each other (full-duplex)
- End the connection



How does it work ?

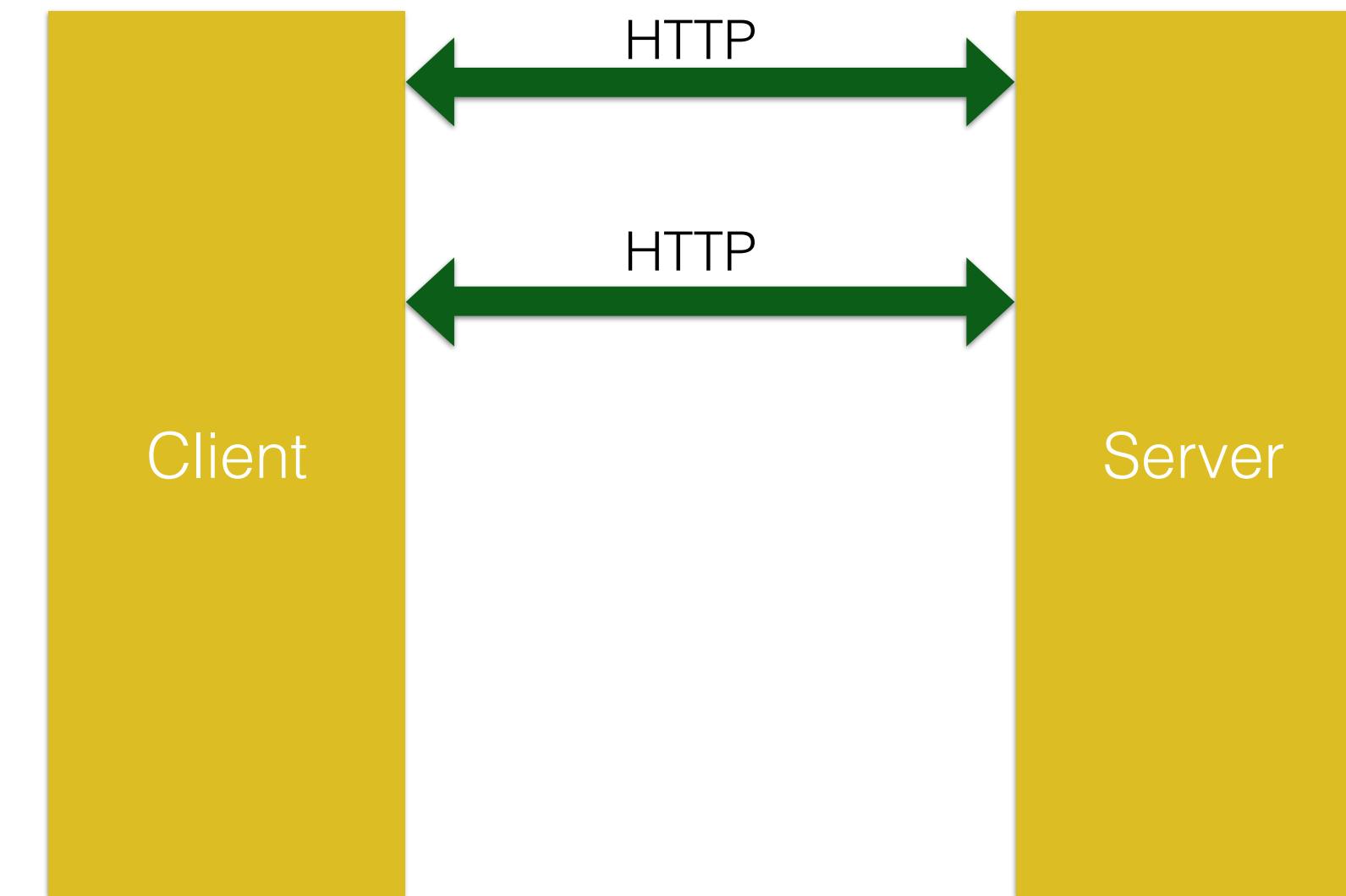


How does it work ?

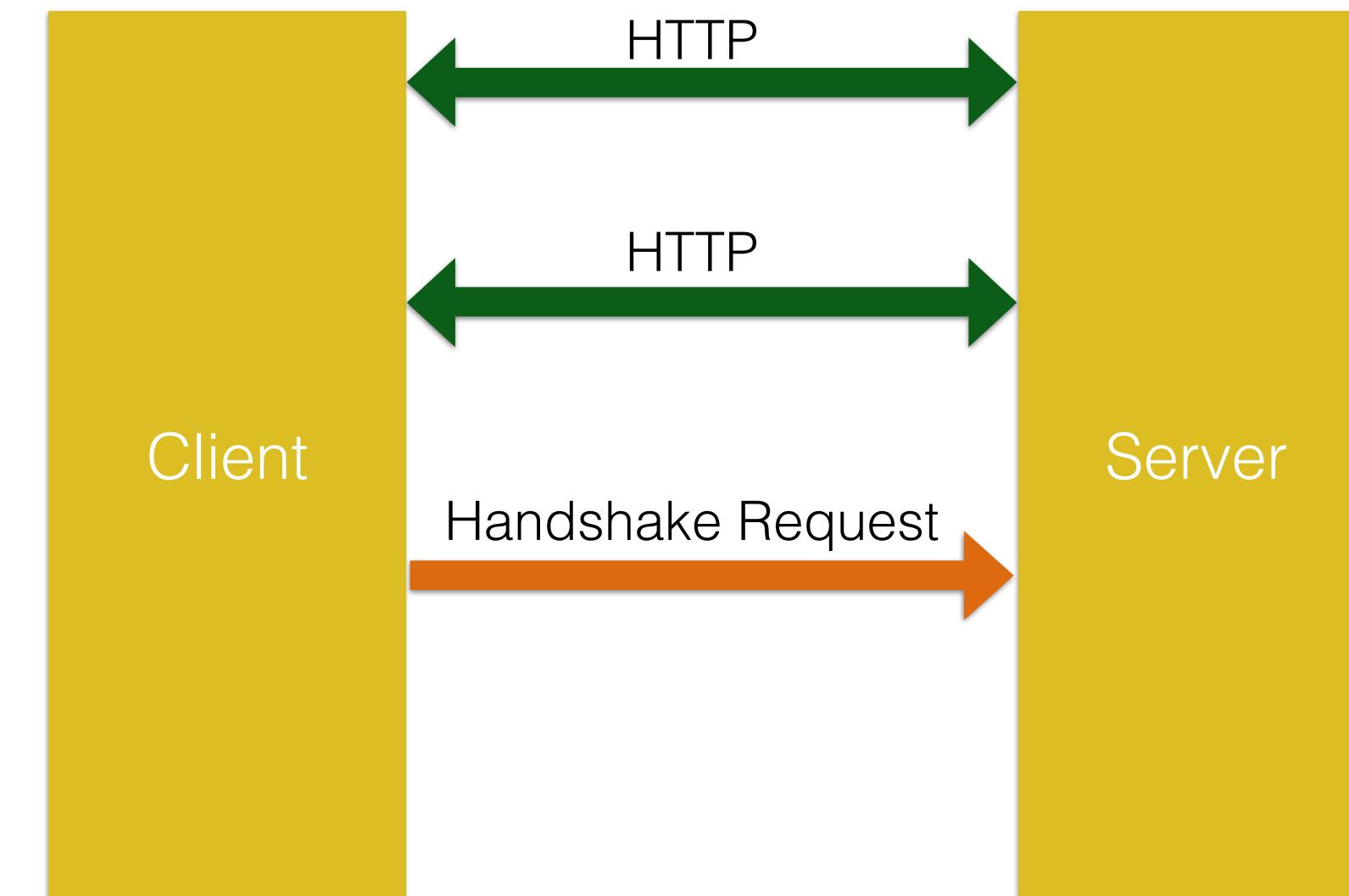




How does it work ?

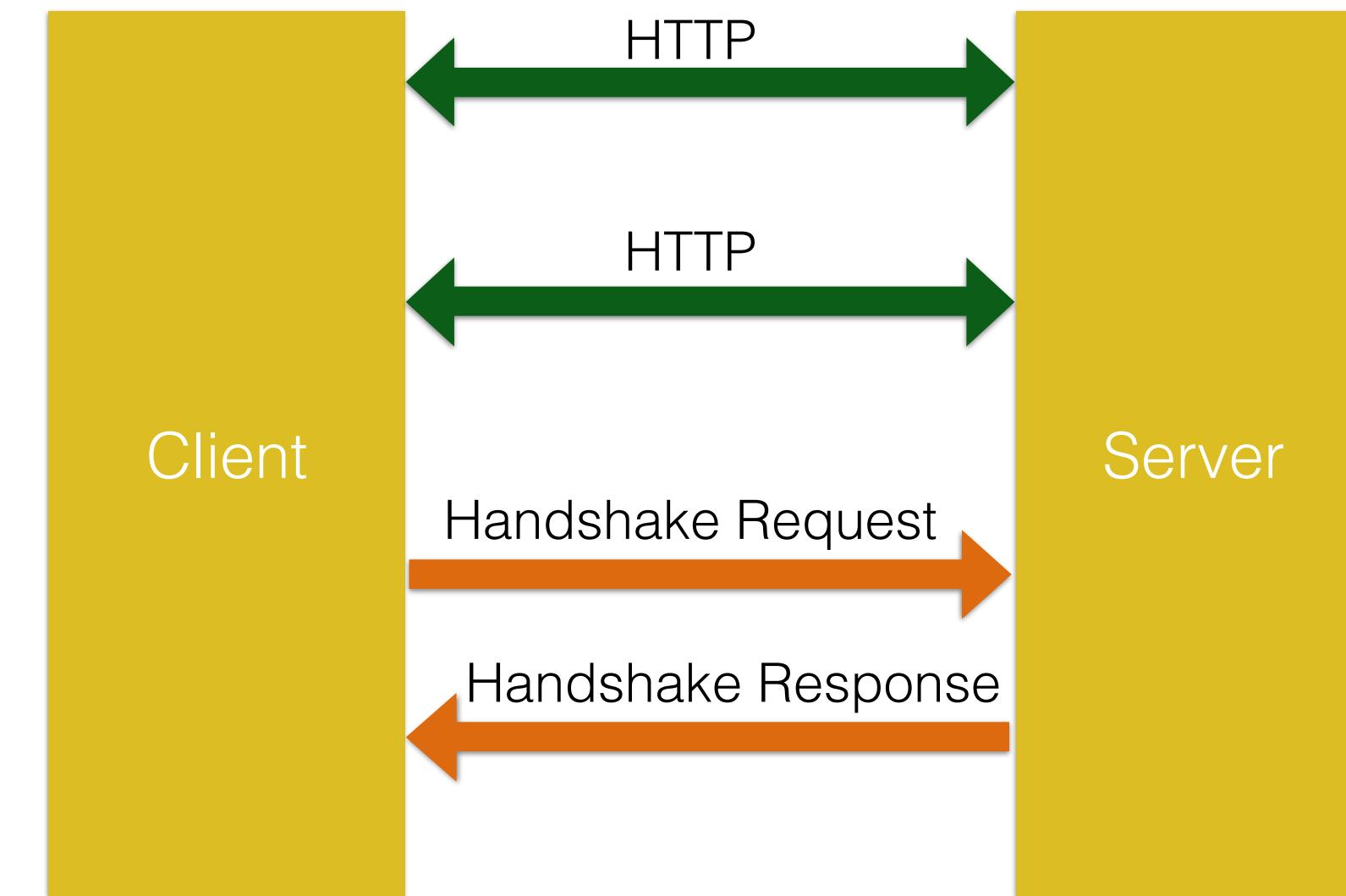


How does it work ?





How does it work ?





Handshake Request

```
GET /chat HTTP/1.1
Host: server.example.com
Upgrade: websocket
Connection: Upgrade
Origin: http://example.com
Sec-WebSocket-Key: dGhIHNhbXBsZSBub25jZQ==
Sec-WebSocket-Protocol: chat, superchat
Sec-WebSocket-Version: 13
```



Handshake Request

GET /chat HTTP/1.1

Host: server.example.com

Upgrade: websocket

Connection: Upgrade

Origin: http://example.com

Sec-WebSocket-Key: dGhIHNhbXBsZSBub25jZQ==

Sec-WebSocket-Protocol: chat, superchat

Sec-WebSocket-Version: 13



Handshake Request

```
GET /chat HTTP/1.1
Host: server.example.com
Upgrade: websocket
Connection: Upgrade
Origin: http://example.com
Sec-WebSocket-Key: dGhIIHNhbXBsZSSub25jZQ==
Sec-WebSocket-Protocol: chat, superchat
Sec-WebSocket-Version: 13
```



Handshake Response

HTTP/1.1 101 Switching Protocols

Upgrade: websocket

Connection: Upgrade

Sec-WebSocket-Accept: s3pPLMBiTxaQ9kYGzzhZRbK+xOo=

Sec-WebSocket-Protocol: chat



Handshake Response

HTTP/1.1 101 Switching Protocols

Upgrade: websocket

Connection: Upgrade

Sec-WebSocket-Accept: s3pPLMBiTxaQ9kYGzzhZRbK+xOo=

Sec-WebSocket-Protocol: chat



Handshake Response

HTTP/1.1 101 Switching Protocols

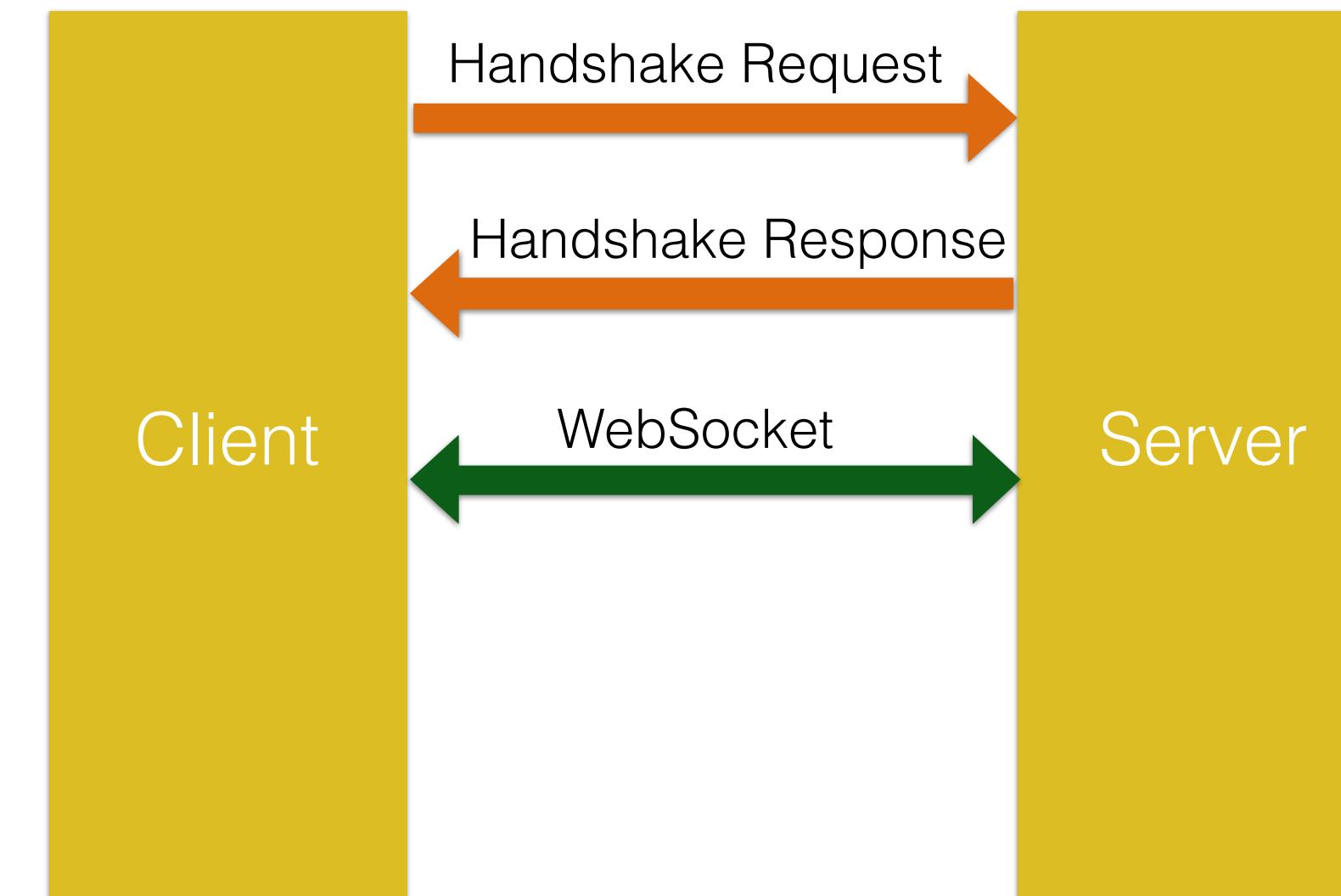
Upgrade: websocket

Connection: Upgrade

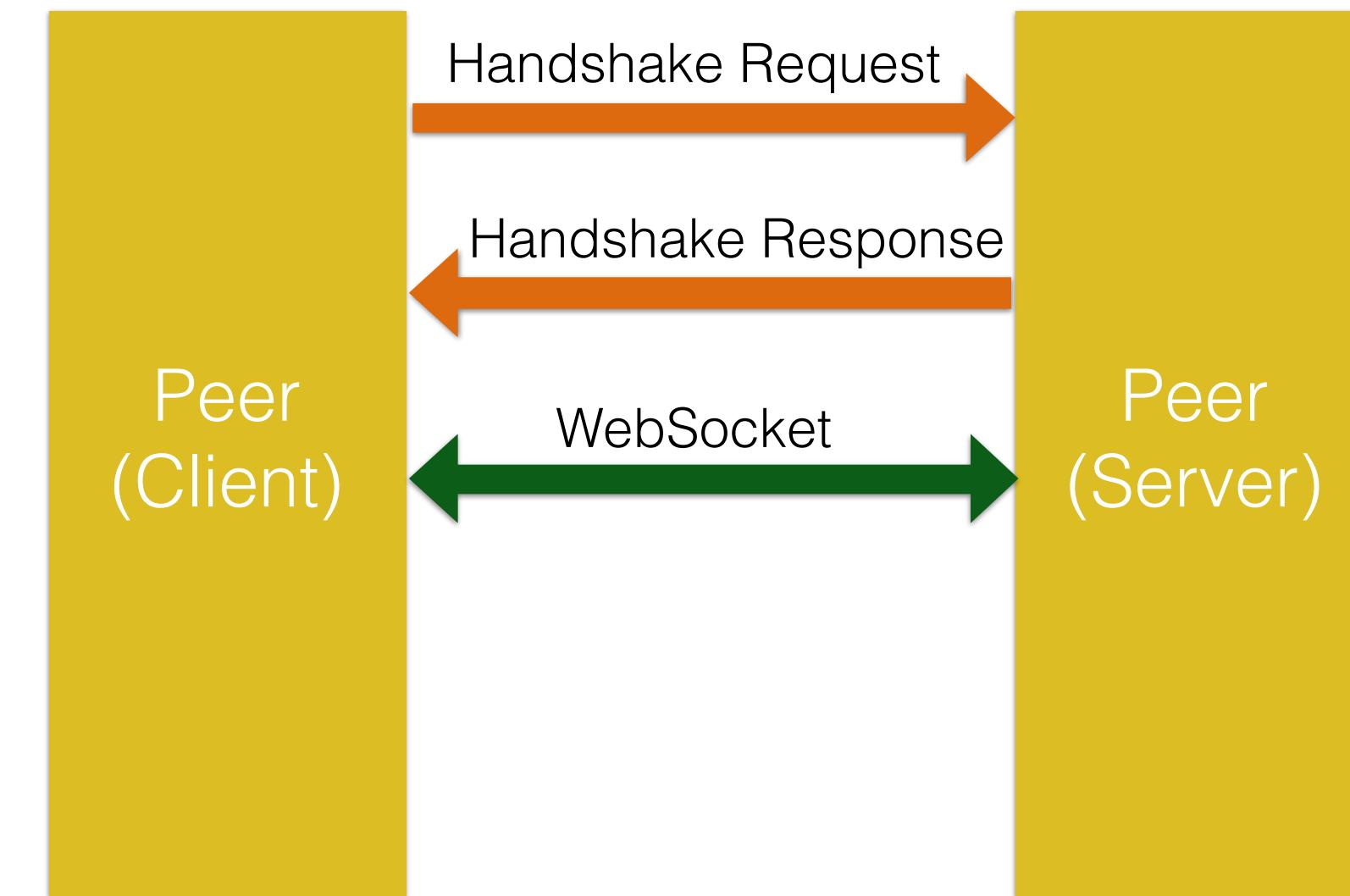
Sec-WebSocket-Accept: s3pPLMBiTxaQ9kYGzzhZRbK+xOo=

Sec-WebSocket-Protocol: chat

How does it work ?

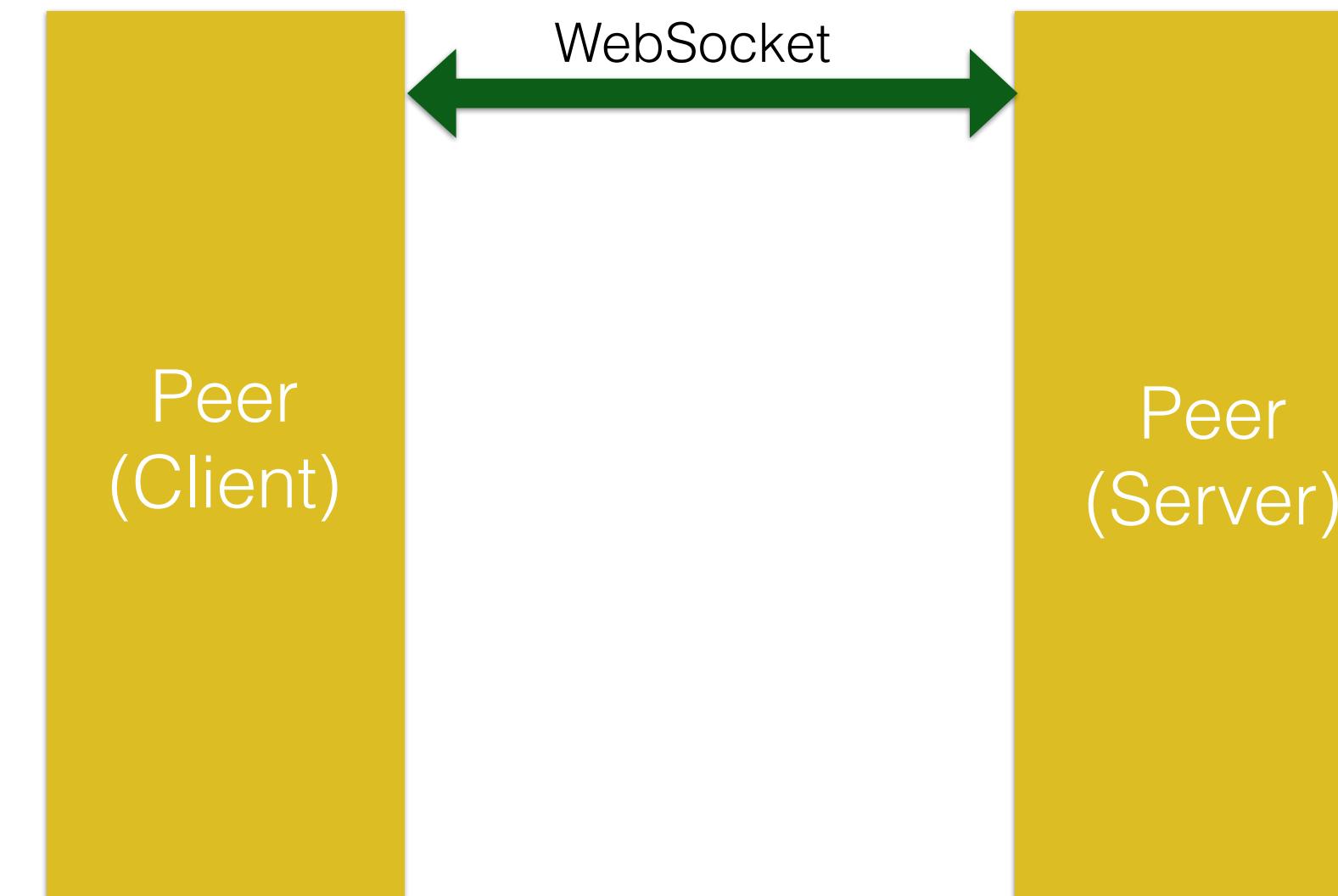


How does it work ?

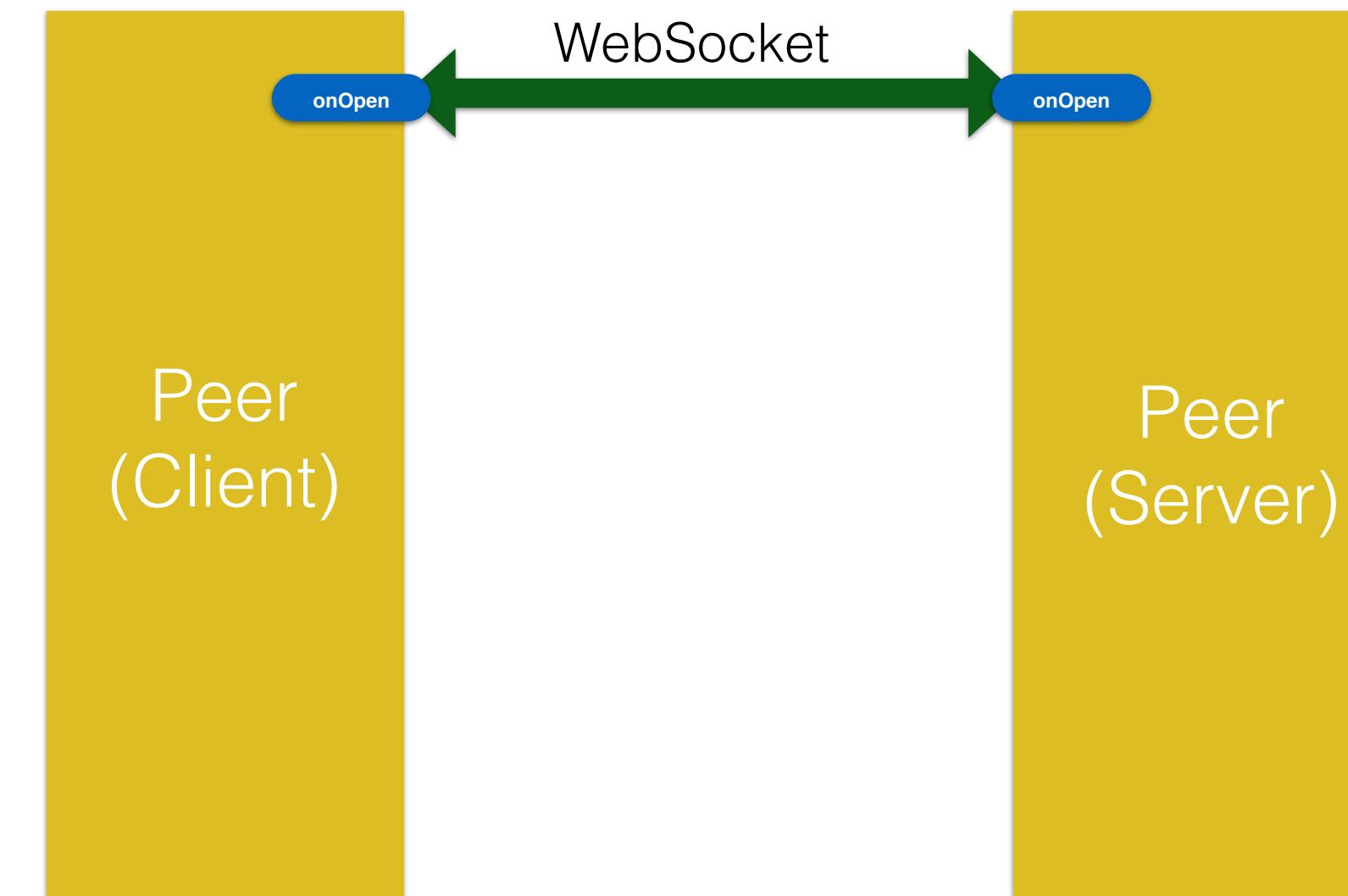




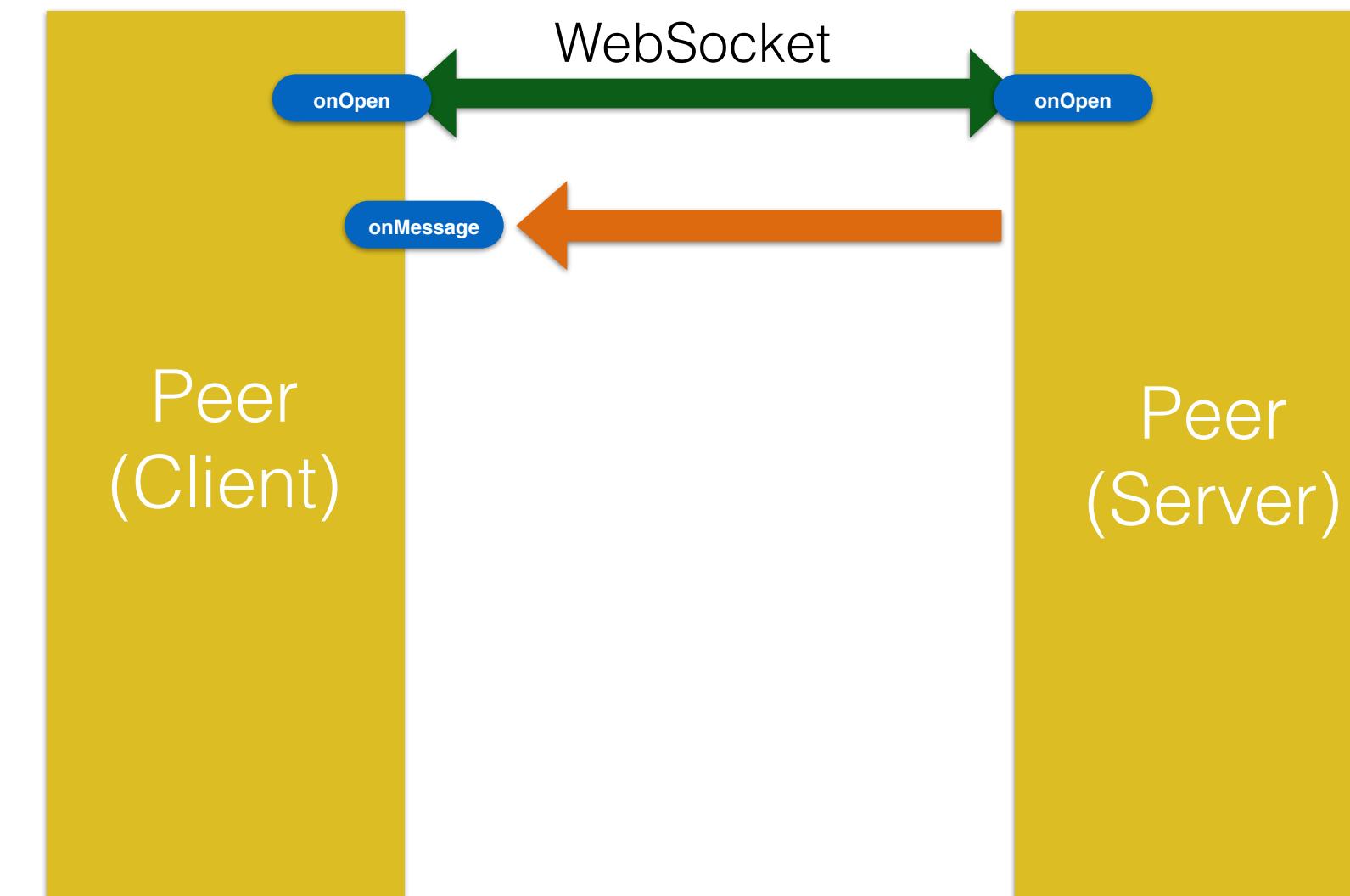
How does it work ?



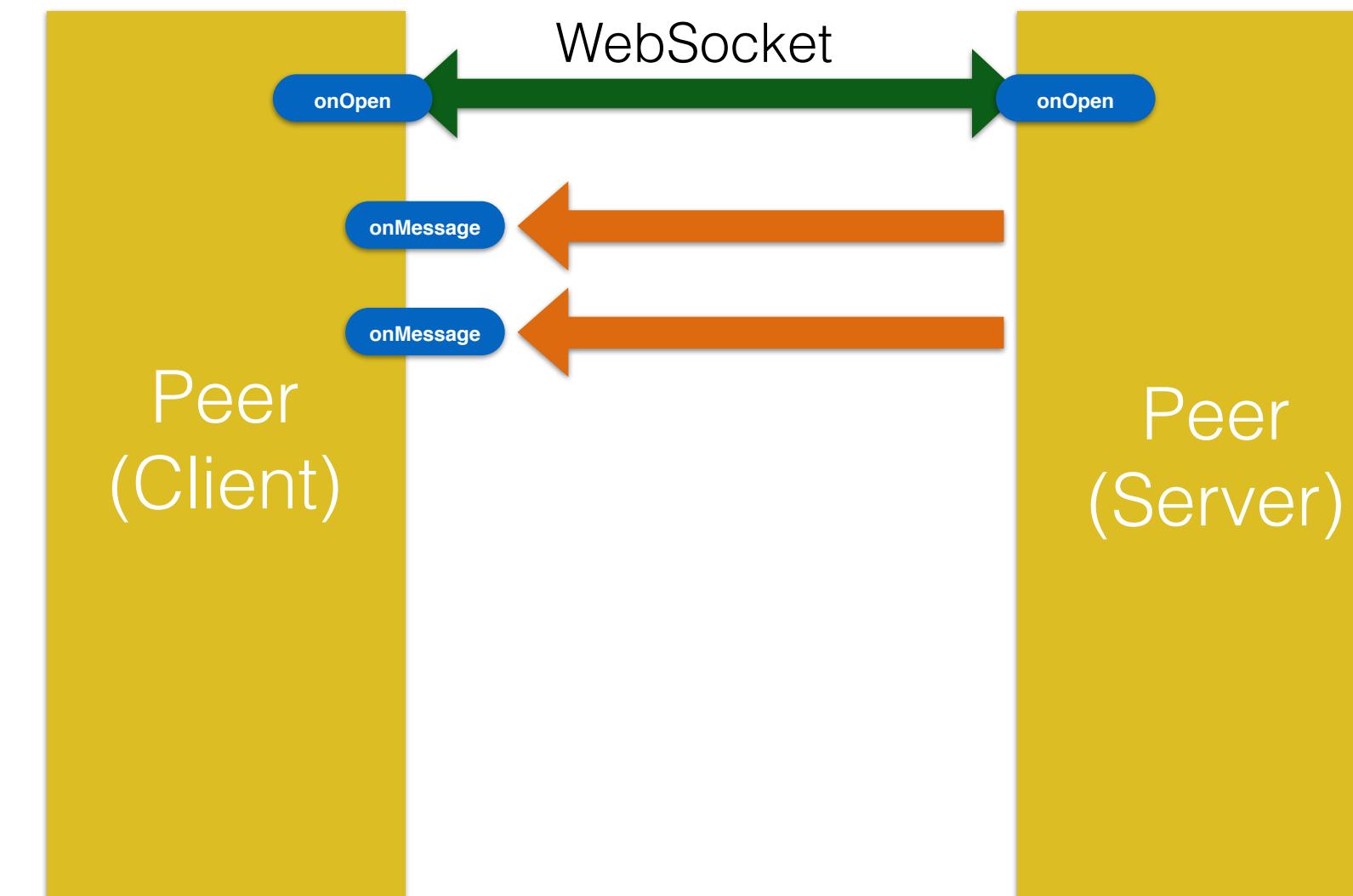
How does it work ?



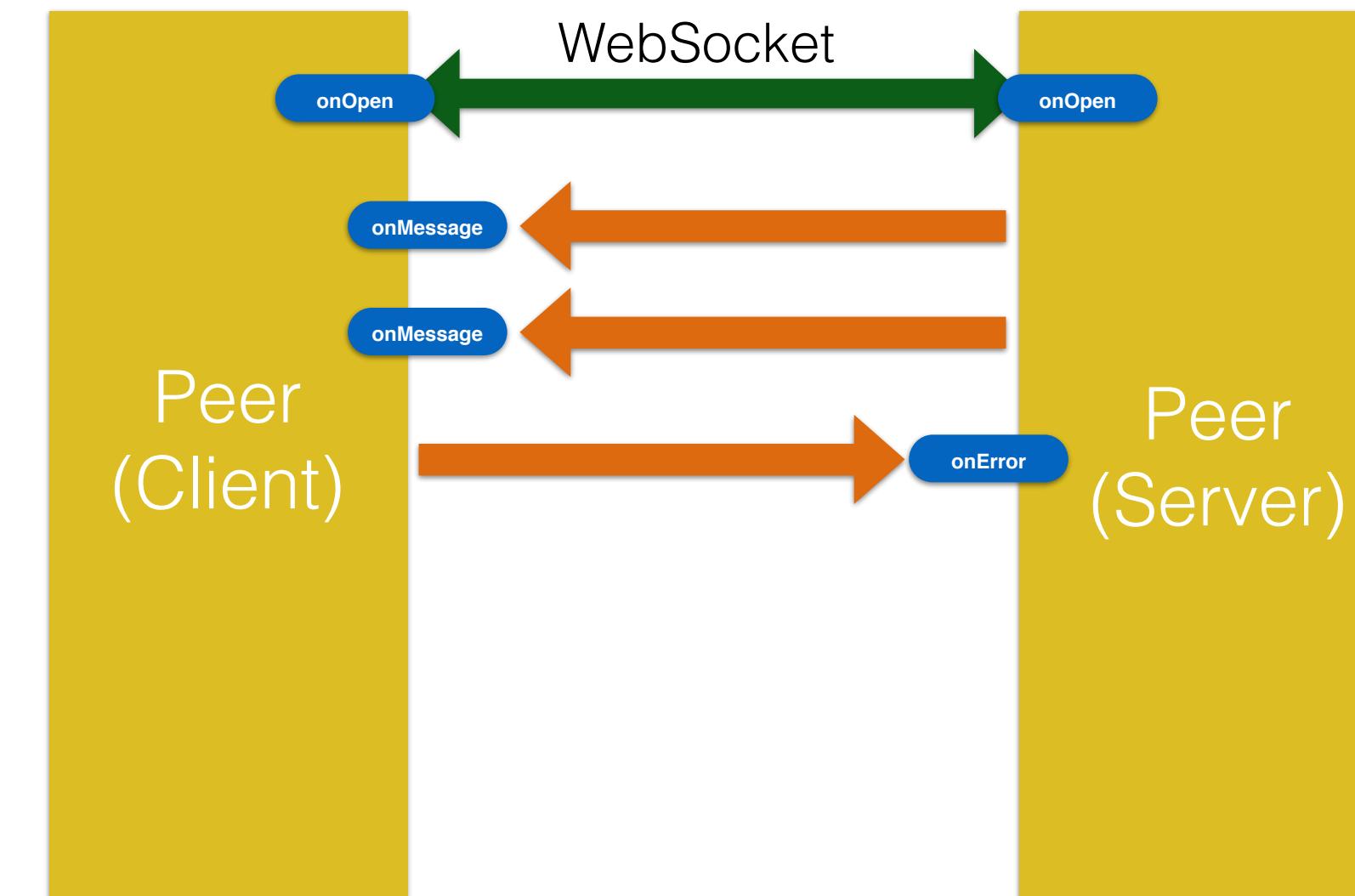
How does it work ?



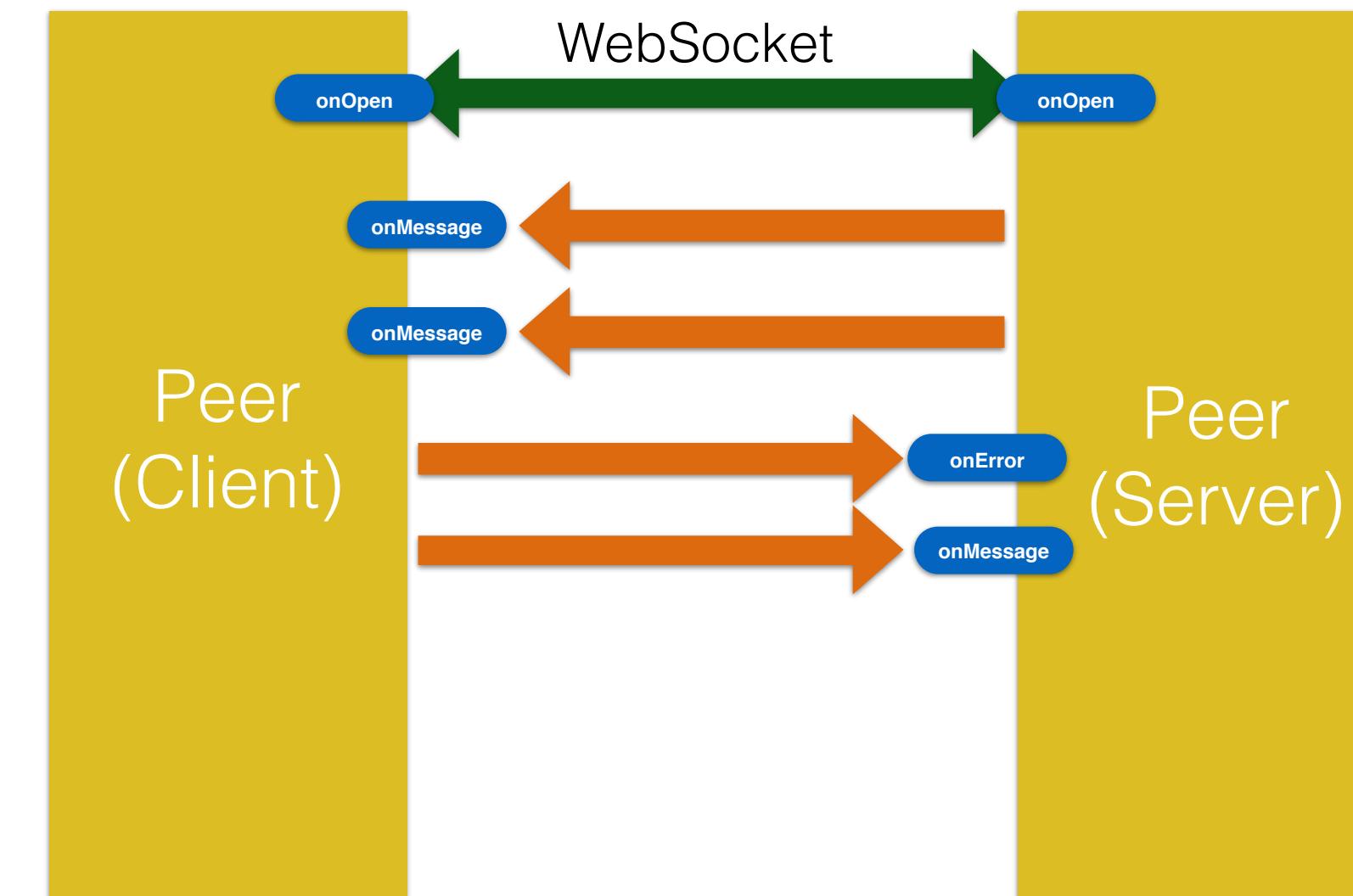
How does it work ?



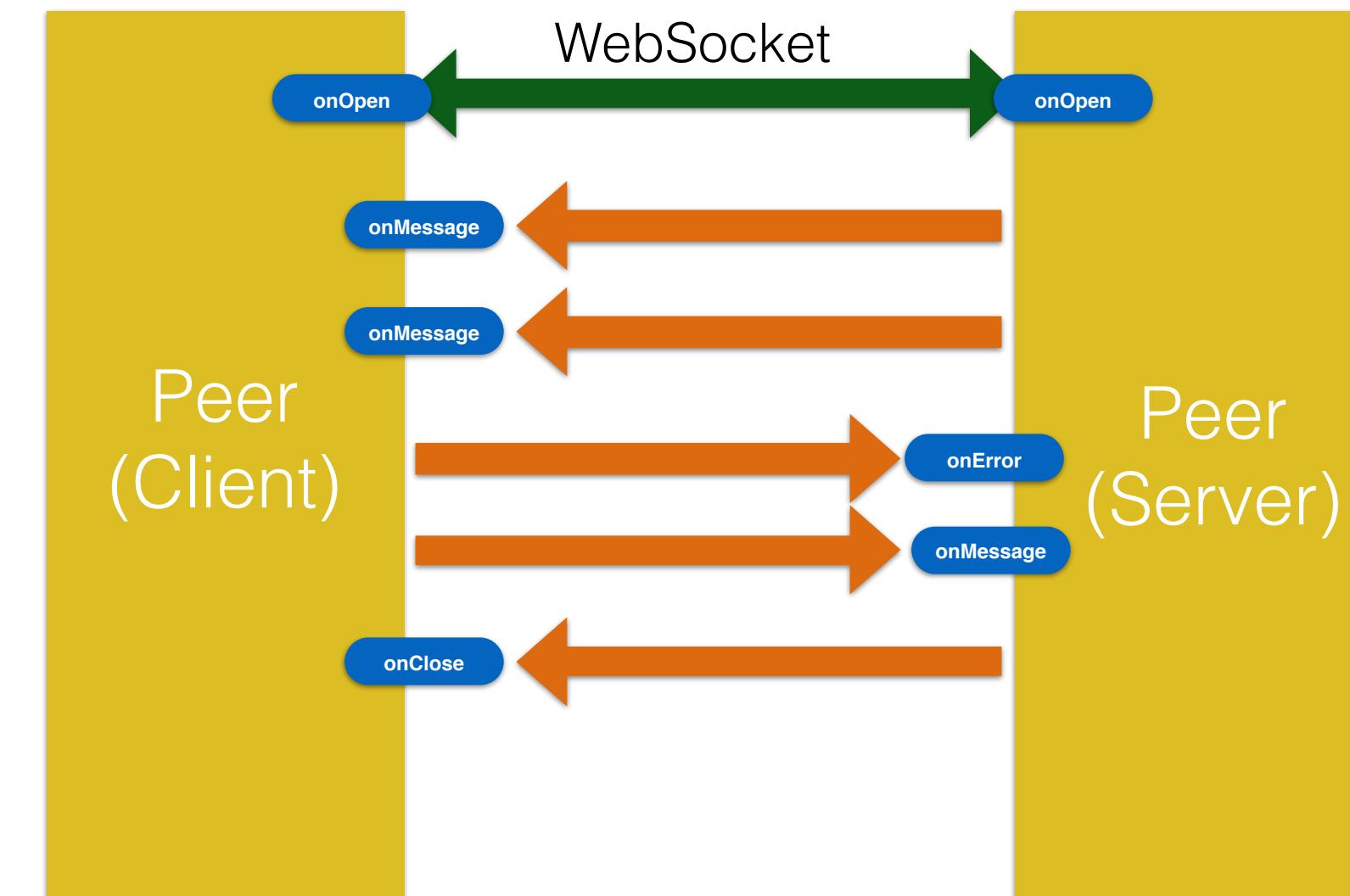
How does it work ?



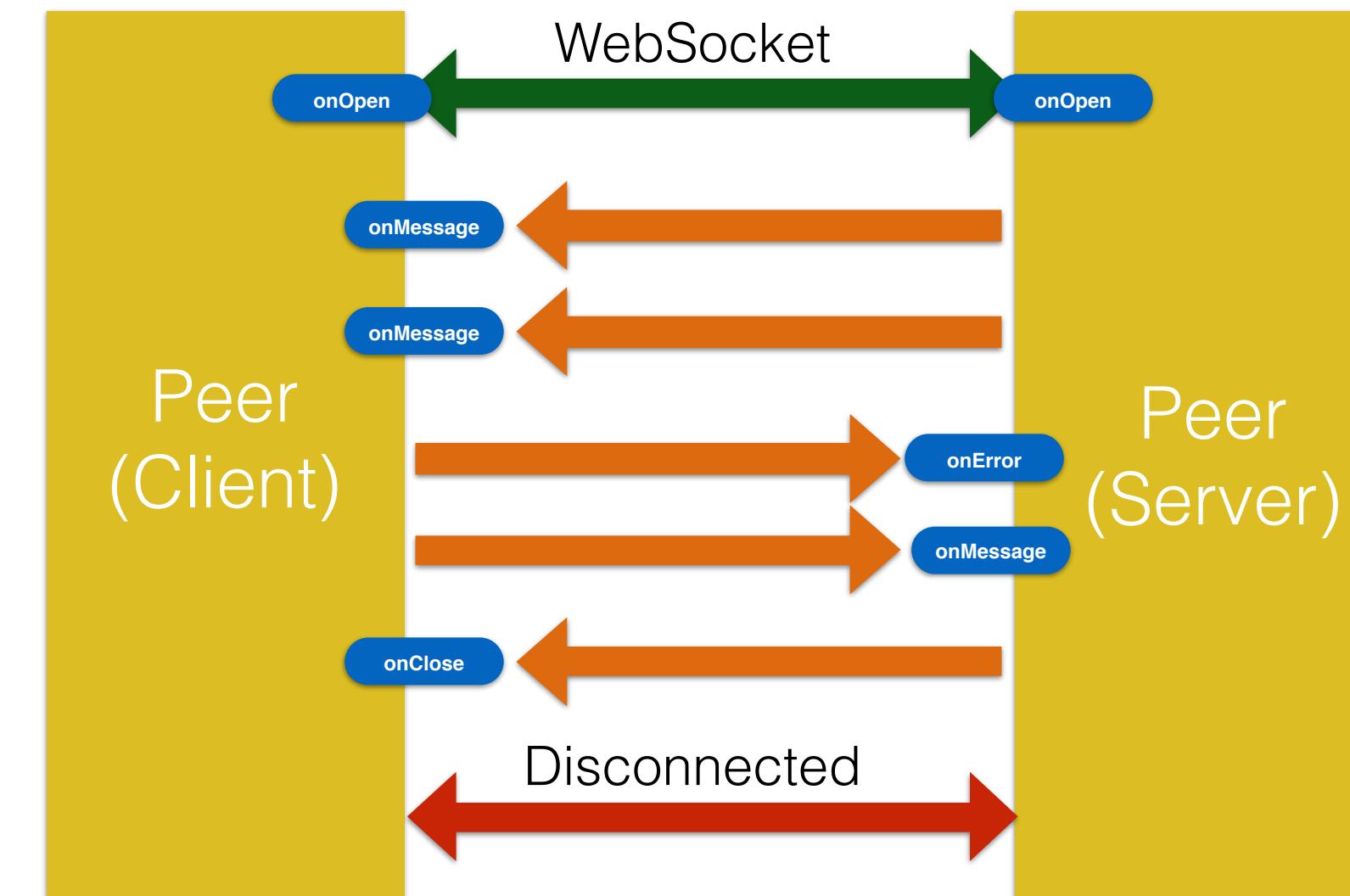
How does it work ?



How does it work ?



How does it work ?





WebSocket Subprotocols



WebSocket Subprotocols

- Facilitates application layer protocols



WebSocket Subprotocols

- Facilitates application layer protocols
- Registered in a Subprotocol name registry



WebSocket Subprotocols

- Facilitates application layer protocols
- Registered in a Subprotocol name registry
 - Identifier, common name, definition



WebSocket Subprotocols

- Facilitates application layer protocols
- Registered in a Subprotocol name registry
 - Identifier, common name, definition
 - www.iana.org/assignments/websocket/websocket.xml#subprotocol-name



WebSocket Subprotocols

- Facilitates application layer protocols
- Registered in a Subprotocol name registry
 - Identifier, common name, definition
 - www.iana.org/assignments/websocket/websocket.xml#subprotocol-name
 - STOMP, XMPP, MQTT, SOAP, ...



WebSocket Extensions



WebSocket Extensions

- Add capabilities to the base protocol



WebSocket Extensions

- Add capabilities to the base protocol
- Multiplexing

<http://tools.ietf.org/html/draft-tamplin-hybi-google-mux>



WebSocket Extensions

- Add capabilities to the base protocol
- Multiplexing
<http://tools.ietf.org/html/draft-tamplin-hybi-google-mux>
- Compression: Only non-control frames/messages



WebSocket Extensions

- Add capabilities to the base protocol
- Multiplexing
<http://tools.ietf.org/html/draft-tamplin-hybi-google-mux>
- Compression: Only non-control frames/messages
 - Per-frame
<http://tools.ietf.org/html/draft-tyoshino-hybi-websocket-perframe-deflate>



WebSocket Extensions

- Add capabilities to the base protocol
- Multiplexing
<http://tools.ietf.org/html/draft-tamplin-hybi-google-mux>
- Compression: Only non-control frames/messages
 - Per-frame
<http://tools.ietf.org/html/draft-tyoshino-hybi-websocket-perframe-deflate>
 - Per-message
<http://tools.ietf.org/html/draft-ietf-hybi-permessage-compression>



WebSocket JavaScript API

```
[Constructor(DOMString url, optional (DOMString or DOMString[]) protocols)]
interface WebSocket : EventTarget {
  readonly attribute DOMString url;

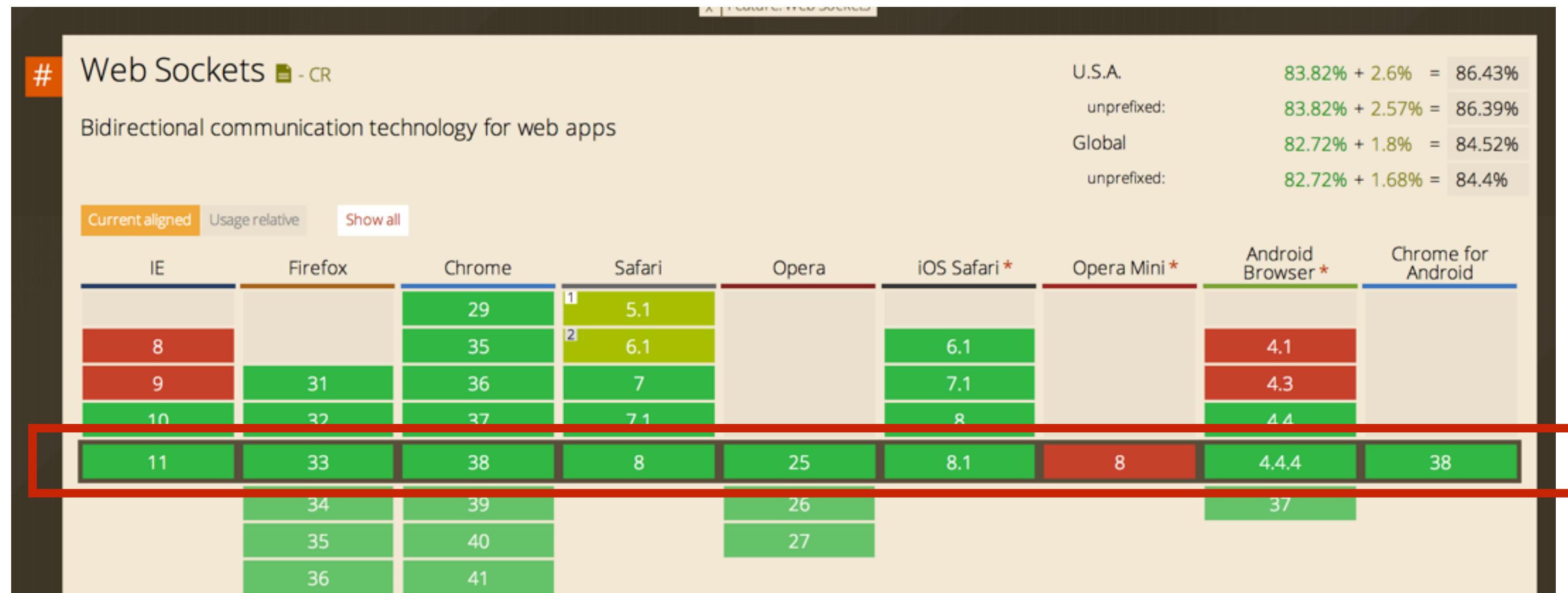
  // ready state
  const unsigned short CONNECTING = 0;
  const unsigned short OPEN = 1;
  const unsigned short CLOSING = 2;
  const unsigned short CLOSED = 3;
  readonly attribute unsigned short readyState;
  readonly attribute unsigned long bufferedAmount;

  // networking
    attribute EventHandler onopen;
    attribute EventHandler onerror;
    attribute EventHandler onclose;
  readonly attribute DOMString extensions;
  readonly attribute DOMString protocol;
  void close([Clamp] optional unsigned short code, optional DOMString reason);

  // messaging
    attribute EventHandler onmessage;
    attribute DOMString binaryType;
  void send(DOMString data);
  void send(Blob data);
  void send(ArrayBuffer data);
  void send(ArrayBufferView data);
};
```

www.w3.org/TR/websockets

Support in Browsers

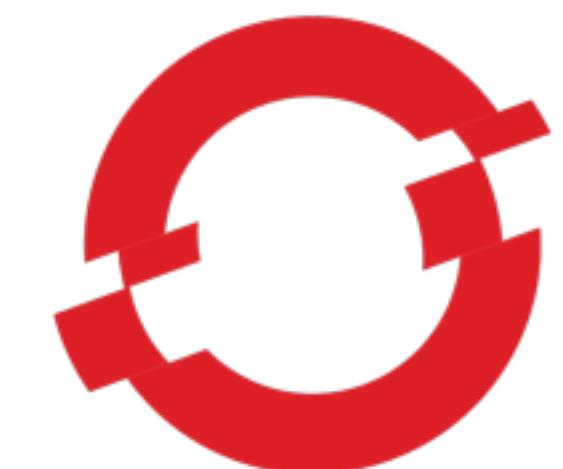


caniuse.com/websockets



Developer Tools

Devoxx™





Java API for WebSocket



Java API for WebSocket

- API for WebSocket server and client endpoint



Java API for WebSocket

- API for WebSocket server and client endpoint
- Annotated: `@ServerEndpoint`, `@ClientEndpoint`





Java API for WebSocket

- API for WebSocket server and client endpoint
 - Annotated: `@ServerEndpoint`, `@ClientEndpoint`
 - Programmatic: `Endpoint`



Java API for WebSocket

- API for WebSocket server and client endpoint
 - Annotated: `@ServerEndpoint`, `@ClientEndpoint`
 - Programmatic: `Endpoint`
 - WebSocket opening handshake negotiation



Java API for WebSocket

- API for WebSocket server and client endpoint
 - Annotated: `@ServerEndpoint`, `@ClientEndpoint`
 - Programmatic: `Endpoint`
 - WebSocket opening handshake negotiation
 - Lifecycle Callback methods



Java API for WebSocket

- API for WebSocket server and client endpoint
 - Annotated: `@ServerEndpoint`, `@ClientEndpoint`
 - Programmatic: `Endpoint`
 - WebSocket opening handshake negotiation
 - Lifecycle Callback methods
 - Integration with Java EE technologies





Annotated Endpoint

```
import javax.websocket.*;  
  
@ServerEndpoint("/hello")  
public class HelloBean {  
    @OnMessage  
    public String sayHello(String name) {  
        return "Hello " + name;  
    }  
}
```



WebSocket Annotations

- Class-level annotations
 - `@ServerEndpoint`: Turns a POJO in a server endpoint
 - `@ClientEndpoint`: Turns a POJO in a client endpoint



WebSocket Annotations

- Method-level annotations
 - `@OnMessage`: Intercepts WebSocket messages
 - `@OnOpen`: Intercepts WebSocket open events
 - `@OnClose`: Intercepts WebSocket close events
 - `@OnError`: Intercepts WebSocket error events



WebSocket Annotations

- Parameter-level annotations
- `@PathParam`: Matches path segment of a URI-template



@ServerEndpoint attributes



@ServerEndpoint attributes

- value: Relative URI or URI template e.g. '/hello' or '/chat/{subscriber-level}'



@ServerEndpoint attributes

- value: Relative URI or URI template e.g. '/hello' or '/chat/{subscriber-level}'
- decoders: list of message decoder classnames



@ServerEndpoint attributes

- **value**: Relative URI or URI template e.g. '/hello' or '/chat/{subscriber-level}'
- **decoders**: list of message decoder classnames
- **encoders**: list of message encoder classnames



@ServerEndpoint attributes

- **value**: Relative URI or URI template e.g. '/hello' or '/chat/{subscriber-level}'
- **decoders**: list of message decoder classnames
- **encoders**: list of message encoder classnames
- **subprotocols**: list of the names of the supported subprotocols



Chat Server

```
@ServerEndpoint("/chat")
public class ChatBean {
    static Set<Session> peers = Collections.synchronizedSet("...");
    @OnOpen
    public void onOpen(Session peer) {
        peers.add(peer);
    }
    @OnClose
    public void onClose(Session peer) {
        peers.remove(peer);
    }
    @OnMessage
    public void message(String message) {
        for (Session peer : peers) {
            peer.getBasicRemote().sendObject(message);
        }
    }
}
```



Chat Server Simplified

```
@ServerEndpoint("/chat")
public class ChatBean {
    @OnMessage
    public void message(String message, Session endpoint) {
        for (Session peer : endpoint.getOpenSessions()) {
            peer.getBasicRemote().sendObject(message);
        }
    }
}
```



<https://github.com/javaee-samples/javaee7-samples/tree/master/websocket/chat>



Custom Payloads

```
@ServerEndpoint(  
    value="/hello",  
    decoders={MyMessageDecoder.class},  
    encoders={MyMessageEncoder.class}  
)  
public class MyEndpoint {  
    . . .  
}
```



Custom Payloads: Text decoder

```
public class MyMessageDecoder implements Decoder.Text<MyMessage> {  
    public MyMessage decode(String s) {  
        JSONObject jsonObject = Json.createReader("...").readObject();  
        return new MyMessage(jsonObject);  
    }  
  
    public boolean willDecode(String string) {  
        . . .  
        return true;  
    }  
  
    . . .  
}
```



Custom Payloads: Text encoder

```
public class MyMessageDecoder implements Encoder.Text<MyMessage> {  
    public String encode(MyMessage myMessage) {  
        return myMessage.jsonObject.toString();  
    }  
  
    . . .  
}
```



Custom Payloads: Binary decoder

```
public class MyMessageDecoder implements Decoder.Binary<MyMessage> {  
    public MyMessage decode(byte[] s) {  
        . . .  
        return myMessage;  
    }  
  
    public boolean willDecode(byte[] string) {  
        . . .  
        return true;  
    }  
  
    . . .  
}
```



Method Signatures



Method Signatures

- Exactly one of the following
 - **Text:** String, boolean, Java primitive or equivalent class, Reader, any type for which there is a decoder
 - **Binary:** byte[], ByteBuffer, byte[] and boolean, ByteBuffer and boolean, InputStream, any type for which there is a decoder
 - **Pong messages:** PongMessage



Method Signatures

- Exactly one of the following
 - **Text:** String, boolean, Java primitive or equivalent class, Reader, any type for which there is a decoder
 - **Binary:** byte[], ByteBuffer, byte[] and boolean, ByteBuffer and boolean, InputStream, any type for which there is a decoder
 - **Pong messages:** PongMessage
 - An optional Session parameter



Method Signatures

- Exactly one of the following
 - **Text:** String, boolean, Java primitive or equivalent class, Reader, any type for which there is a decoder
 - **Binary:** byte[], ByteBuffer, byte[] and boolean, ByteBuffer and boolean, InputStream, any type for which there is a decoder
 - **Pong messages:** PongMessage
- An optional **Session** parameter
- 0..n String parameters annotated with **@PathParam**



Method Signatures

- Exactly one of the following
 - **Text:** String, boolean, Java primitive or equivalent class, Reader, any type for which there is a decoder
 - **Binary:** byte[], ByteBuffer, byte[] and boolean, ByteBuffer and boolean, InputStream, any type for which there is a decoder
 - **Pong messages:** PongMessage
- An optional **Session** parameter
- 0..n String parameters annotated with **@PathParam**



Sample Messages



Sample Messages

- void m(String s);



Sample Messages

- void m(String s);
- void m(Float f, @PathParam("id") int id);



Sample Messages

- void m(String s);
- void m(Float f, @PathParam("id")int id);
- Product m(Reader reader, Session s);



Sample Messages

- void m(String s);
- void m(Float f, @PathParam("id")int id);
- Product m(Reader reader, Session s);
- void m(byte[] b); or void m(ByteBuffer b);



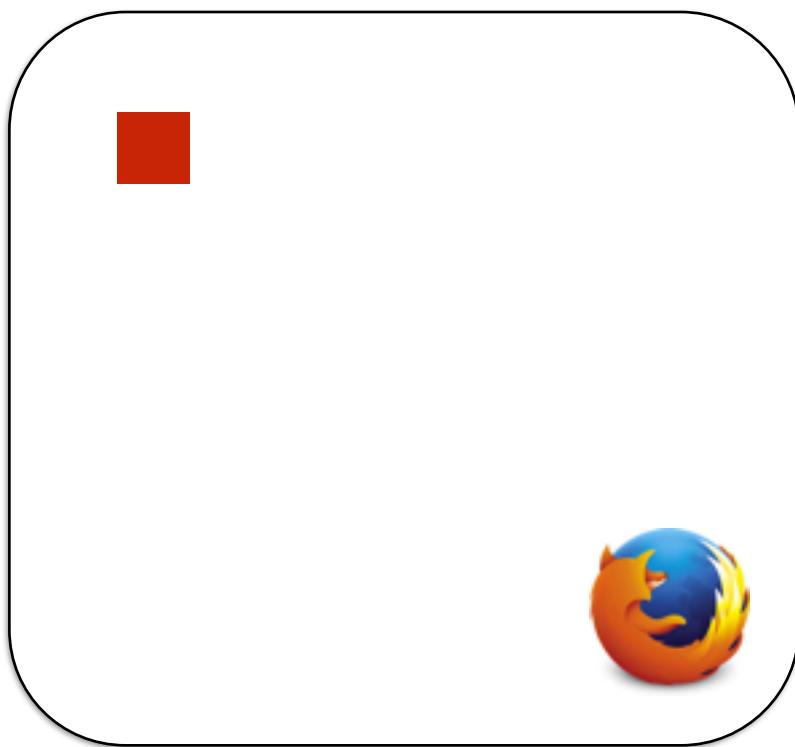
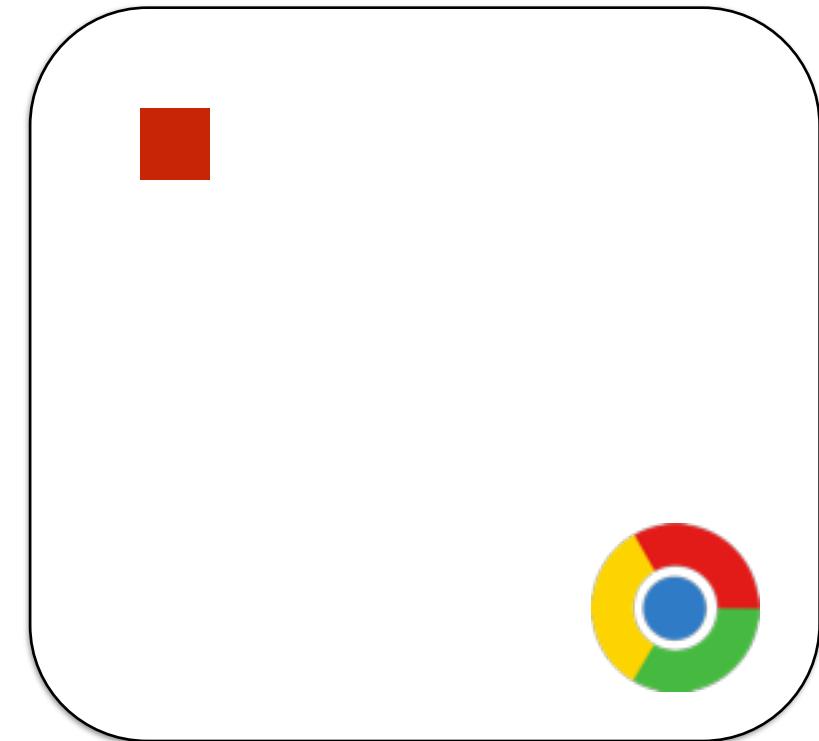
Sample Messages

- void m(String s);
- void m(Float f, @PathParam("id")int id);
- Product m(Reader reader, Session s);
- void m(byte[] b); or void m(ByteBuffer b);
- Book m(int i, Session s, @PathParam("isbn")String isbn, @PathParam("store")String store);

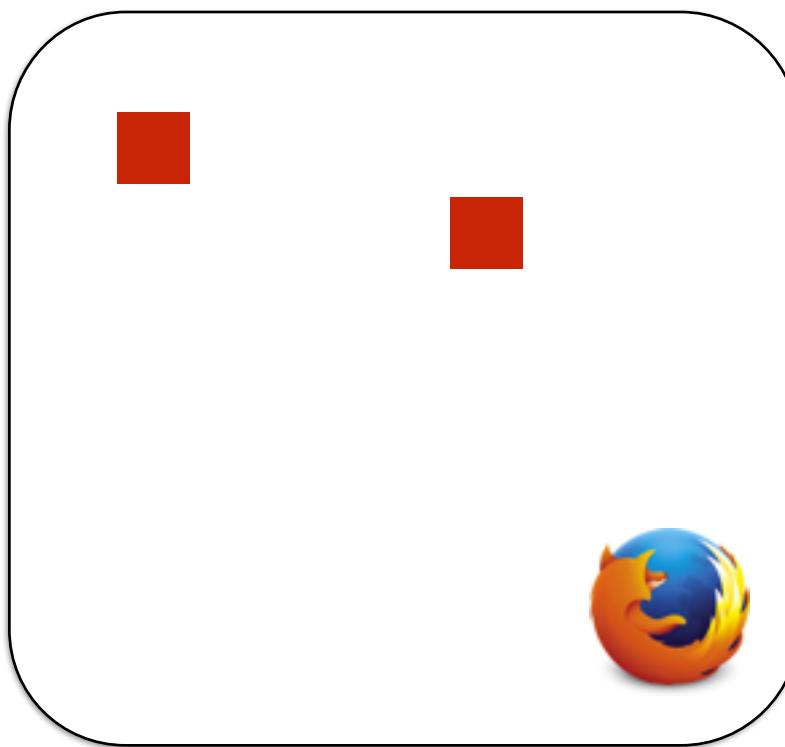
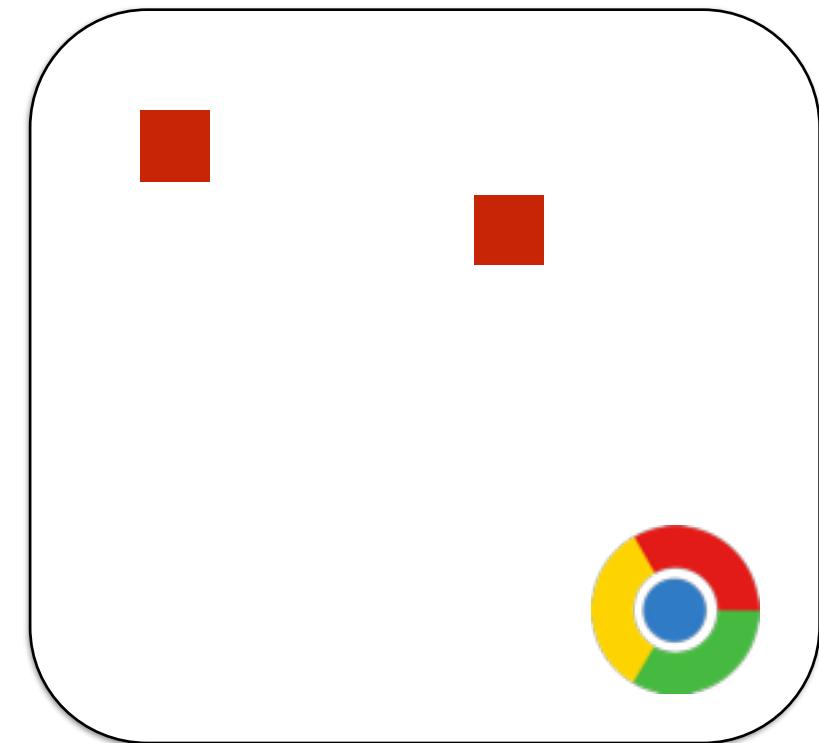
Devoxx™



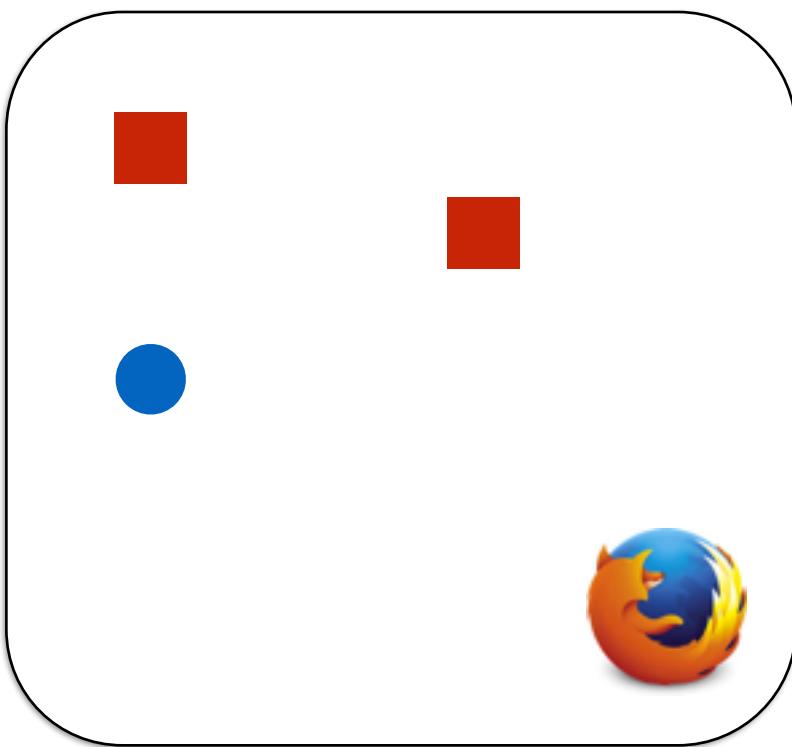
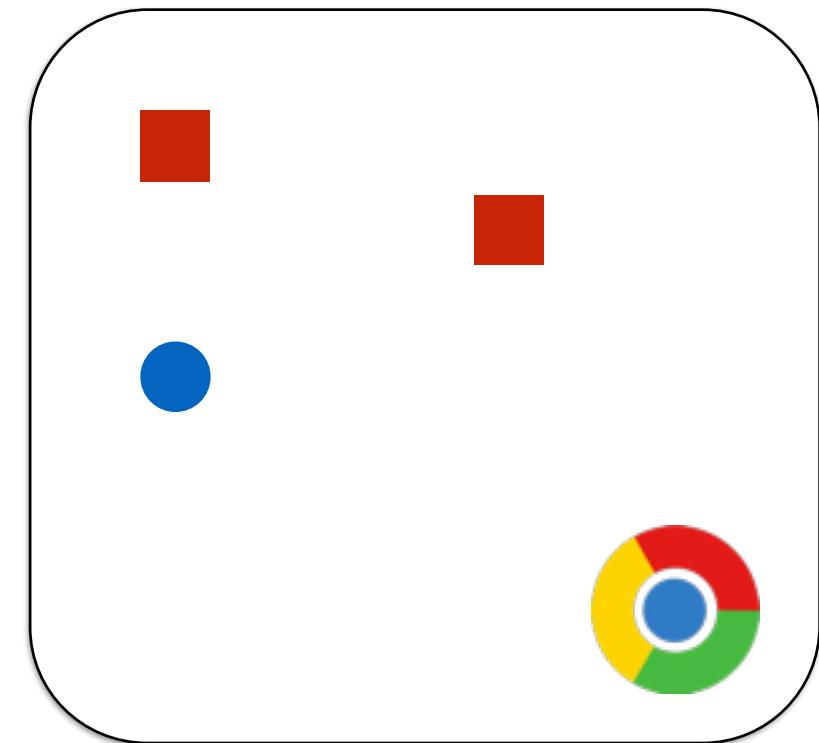
TM
Devoxx



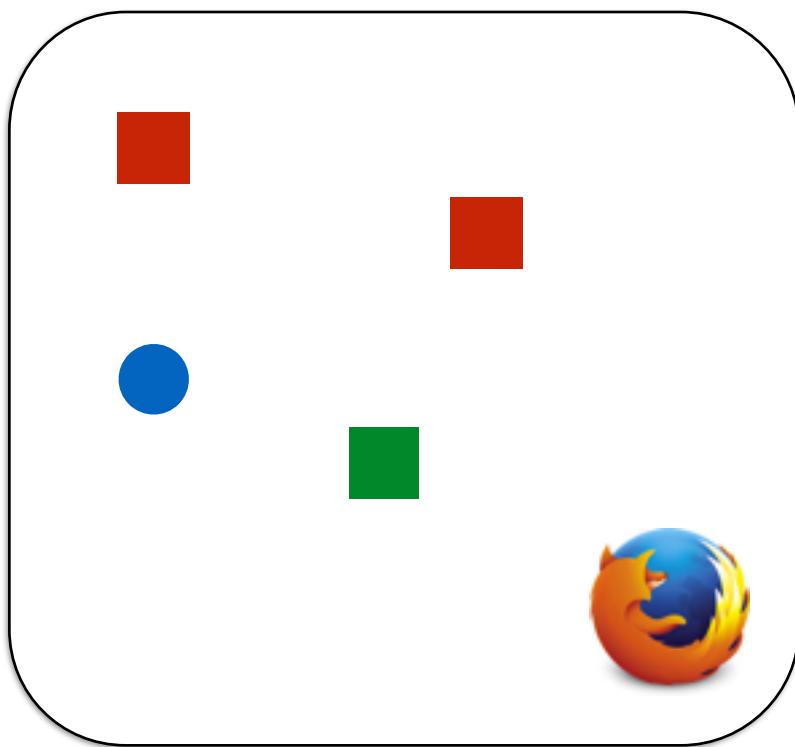
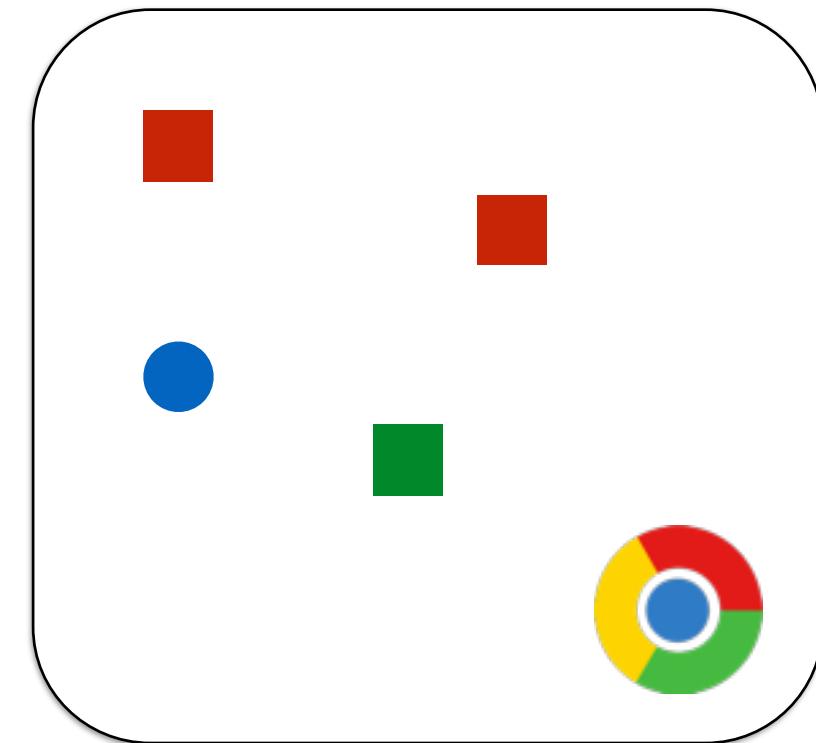
TM
Devoxx



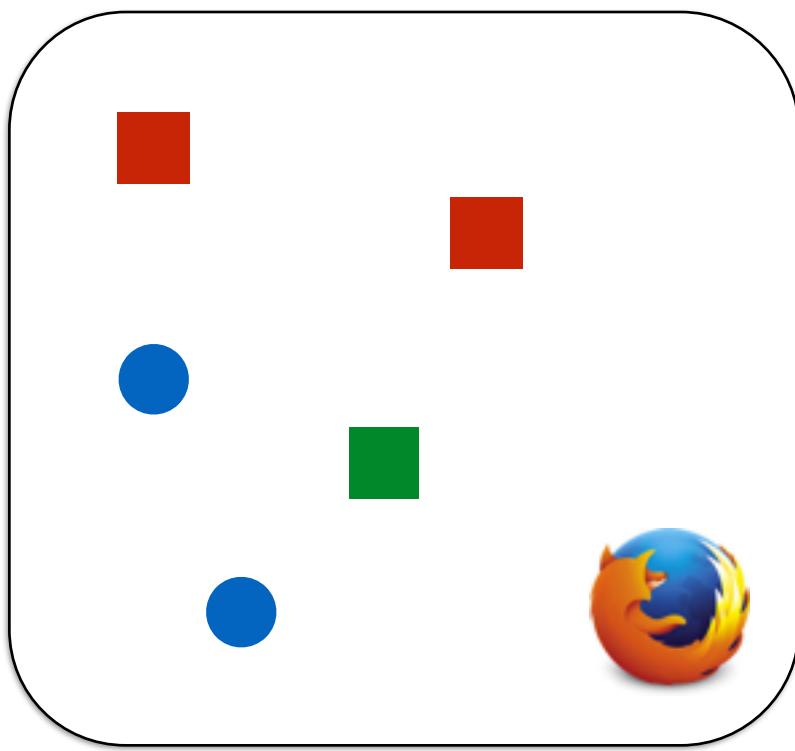
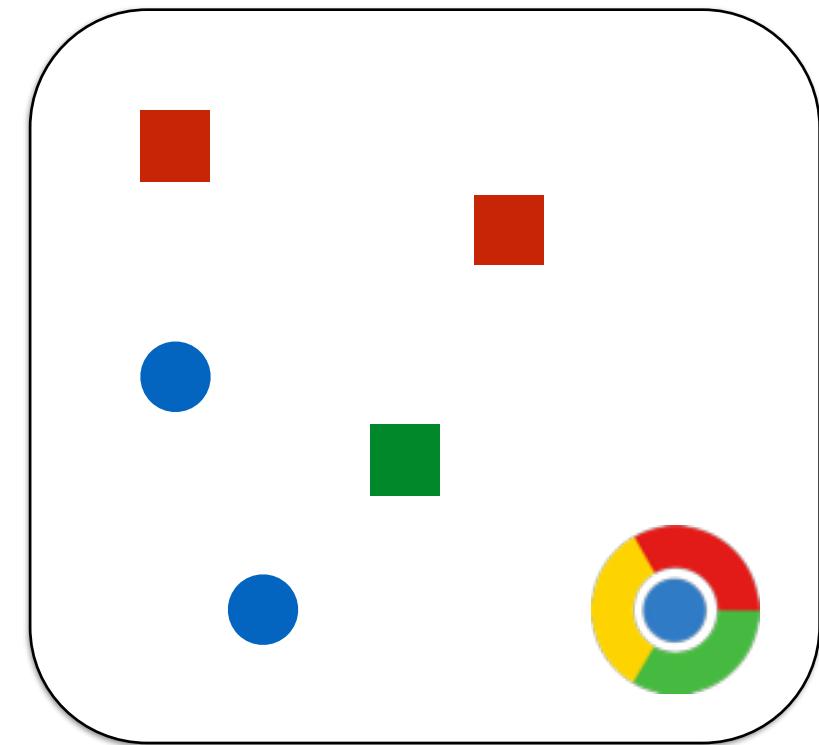
TM
Devoxx

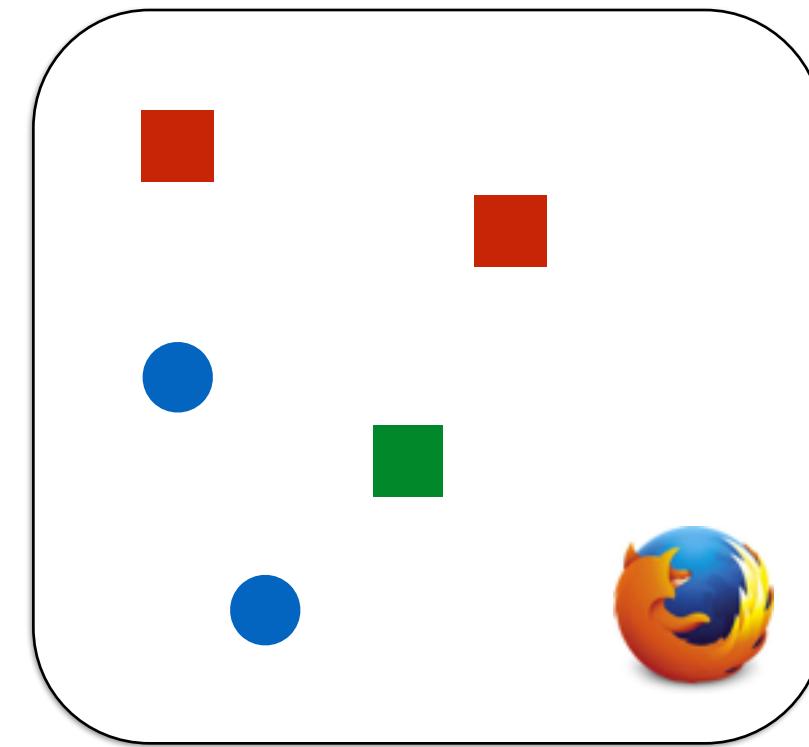
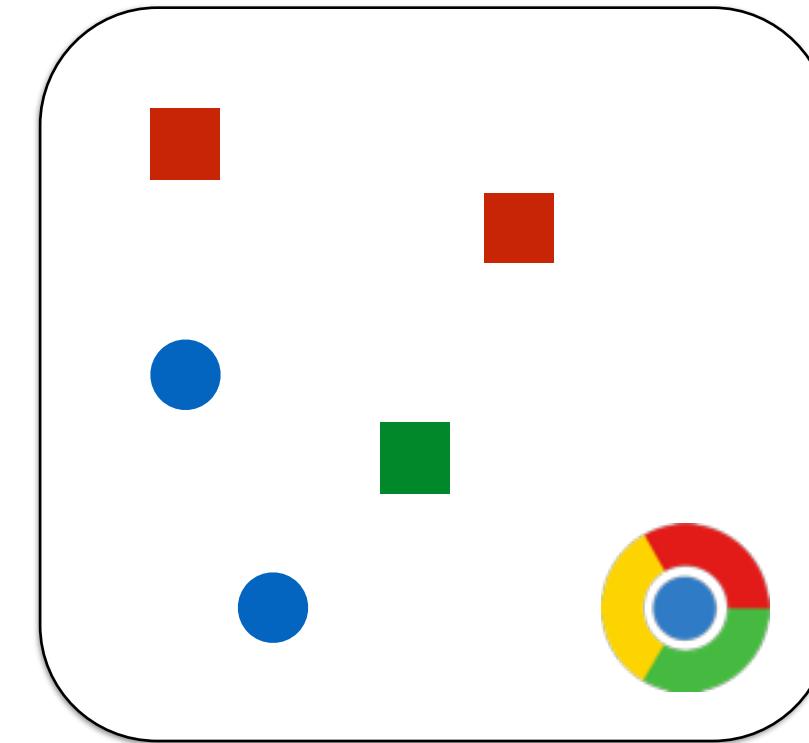


TM
Devoxx



TM
Devoxx





<https://github.com/javaee-samples/javaee7-samples/tree/master/websocket/whiteboard>



URI Template Matching

```
@ServerEndpoint("/chat/{roomid}")
public class ChatServer {
    @OnMessage
    public void receiveMessage(
        @PathParam("roomid")String roomId) {
        . . .
    }
}
```



Client Endpoint

```
@ClientEndpoint
```

```
public class HelloClient {  
    @OnMessage public void message(  
        String message,  
        Session session) {  
        // . . .  
    }  
}
```



Client Endpoint

```
@ClientEndpoint
```

```
public class HelloClient {  
    @OnMessage public void message(  
        String message,  
        Session session) {  
        // . . .  
    }  
}
```

```
WebSocketContainer c = ContainerProvider.getWebSocketContainer();  
c.connectToServer(HelloClient.class, "hello");
```



lorem ipsum dolor



lorem ipsum dolor



<https://github.com/javaee-samples/javaee7-samples/tree/master/websocket/google-docs>



Programmatic Endpoint

```
public class ChatServer extends Endpoint {  
    @Override  
    public void onOpen(Session session) {  
        session.addMessageHandler(new MessageHandler.Text() {  
            public void onMessage(String message) {  
                try {  
                    session  
                        .getBasicRemote()  
                        .sendText(message);  
                } catch (IOException ex) { }  
            }  
        } );  
    }  
}
```



Programmatic Endpoint Config

```
public class MyEndpointConfig implements ServerApplicationConfig {  
  
    @Override  
    public Set<ServerEndpointConfig> getEndpointConfigs(  
        Set<Class<? extends Endpoint>> set) {  
        return new HashSet<ServerEndpointConfig>() {  
            {  
                add(ServerEndpointConfig.Builder  
                    .create(ChatServer.class, "/chat")  
                    .build());  
            }  
        };  
    }  
  
    @Override  
    public Set<Class<?>> getAnnotatedEndpointClasses(Set<Class<?>> set) {  
        return Collections.emptySet();  
    }  
}
```



<https://github.com/javaee-samples/javaee7-samples/tree/master/websocket/endpoint-singleton>



Securing WebSockets



Securing WebSockets

- Origin-based security model



Securing WebSockets

- Origin-based security model
- Sec-xxx keys can not be set using XMLHttpRequest
- Sec-WebSocket-Key, Sec-WebSocket-Version



Securing WebSockets

- Origin-based security model
- Sec-xxx keys can not be set using XMLHttpRequest
 - Sec-WebSocket-Key, Sec-WebSocket-Version
- User-based security using Servlet security mechanism
 - Endpoint mapped by `ws://` is protected using security model defined using the corresponding `http://` URI
 - Authorization defined using `<security-constraint>`



Securing WebSockets

- Origin-based security model
- Sec-xxx keys can not be set using XMLHttpRequest
 - Sec-WebSocket-Key, Sec-WebSocket-Version
- User-based security using Servlet security mechanism
 - Endpoint mapped by **ws://** is protected using security model defined using the corresponding http:// URI
 - Authorization defined using <security-constraint>
- Transport Confidentiality using **wss://**
 - Access allowed over encrypted connection only



Cross-Origin Resource Sharing



Cross-Origin Resource Sharing

- Relaxes same-origin restrictions to network requests



Cross-Origin Resource Sharing

- Relaxes same-origin restrictions to network requests
- Servers include Access-Control-Allow-Origin HTTP header



Cross-Origin Resource Sharing

- Relaxes same-origin restrictions to network requests
- Servers include Access-Control-Allow-Origin HTTP header
- Access-Control-* headers
 - Max-Age
 - Allow-Methods
 - Allow-Headers
 - ...



Cross-Origin Resource Sharing

- Relaxes same-origin restrictions to network requests
- Servers include Access-Control-Allow-Origin HTTP header
- Access-Control-* headers
 - Max-Age
 - Allow-Methods
 - Allow-Headers
 - ...
- www.w3.org/TR/cors/



User-based Security

<https://github.com/javaee-samples/javaee7-samples/tree/master/websocket/endpoint-security>



TLS-based Security

<https://github.com/javaee-samples/javaee7-samples/tree/master/websocket/endpoint-wss>



Embedded WebSocket



Embedded WebSocket

- Undertow New web server in WildFly 8



Embedded WebSocket

- Undertow New web server in WildFly 8
- Blocking and non-blocking based on NIO

Devoxx
Java



Embedded WebSocket

- Undertow New web server in WildFly 8
- Blocking and non-blocking based on NIO
- Composition/handler-based architecture



Embedded WebSocket

- Undertow New web server in WildFly 8
- Blocking and non-blocking based on NIO
- Composition/handler-based architecture
- Lightweight and fully embeddable



Embedded WebSocket

- Undertow New web server in WildFly 8
- Blocking and non-blocking based on NIO
- Composition/handler-based architecture
- Lightweight and fully embeddable
- Supports Servlet 3.1 and HTTP Upgrade



Embedded WebSocket

- Undertow New web server in WildFly 8
- Blocking and non-blocking based on NIO
- Composition/handler-based architecture
- Lightweight and fully embeddable
- Supports Servlet 3.1 and HTTP Upgrade
- mod_cluster supported

X
X
O
>
H
D

Undertow is awesome!



```
techempower@lg01:~$ wrk -d 30 -c 256 -t 40 http://10.0.3.2:8080/byte
Running 30s test @ http://10.0.3.2:8080/byte
  40 threads and 256 connections
    Thread Stats      Avg      Stdev     Max   +/- Stdev
      Latency    247.05us    3.52ms  624.37ms  99.90%
      Req/Sec    27.89k     6.24k   50.22k  71.15%
  31173283 requests in 29.99s 3.83GB read
    Socket errors: connect 0, read 0, write 0, timeout 9
Requests/sec: 1039305.27
Transfer/sec:   130.83MB
```

This is output from [Wrk](#) testing a single server running [Undertow](#) using conditions similar to Google's test (1-byte response body, no HTTP pipelining, no special request headers) **1.039 million requests per second.**

<http://www.techempower.com/blog/2014/03/04/one-million-http-rps-without-load-balancing-is-easy/>



git@github.com:undertow-io/undertow.git



Load Balance WebSocket



Load Balance WebSocket

- Reverse proxy



Load Balance WebSocket

- Reverse proxy
- Apache module: `mod_proxy_wstunnel`



Load Balance WebSocket

- Reverse proxy
 - Apache module: `mod_proxy_wstunnel`
 - Only vertical scaling



Load Balance WebSocket

- Reverse proxy
 - Apache module: `mod_proxy_wstunnel`
- Only vertical scaling
- No session replication



<http://blog.arungupta.me/2014/08/load-balance-websockets-apache-htpd-techtip48/>



Pub/Sub over WebSocket

<https://github.com/arun-gupta/kaazing-openshift-cartridge>

TM
X
O
>
S
D

STOMP over WebSocket

Stomp 



STOMP over WebSocket

Stomp

- STOMP: Simple Text Oriented Messaging Protocol





STOMP over WebSocket

Stomp

- STOMP: Simple Text Oriented Messaging Protocol
- Interoperable wire format: any client, any broker



STOMP over WebSocket

Stomp 

- STOMP: Simple Text Oriented Messaging Protocol
- Interoperable wire format: any client, any broker
- Messaging interoperability among languages and platforms



STOMP over WebSocket

Stomp 

- STOMP: Simple Text Oriented Messaging Protocol
- Interoperable wire format: any client, any broker
- Messaging interoperability among languages and platforms
 - Unlike JMS



STOMP over WebSocket

Stomp 

- STOMP: Simple Text Oriented Messaging Protocol
- Interoperable wire format: any client, any broker
- Messaging interoperability among languages and platforms
 - Unlike JMS
- REST for messaging: CONNECT, SEND, SUBSCRIBE, ...



STOMP over WebSocket

Stomp 

- STOMP: Simple Text Oriented Messaging Protocol
- Interoperable wire format: any client, any broker
- Messaging interoperability among languages and platforms
 - Unlike JMS
- REST for messaging: CONNECT, SEND, SUBSCRIBE, ...
- Map STOMP frames to WebSocket frames



<https://github.com/arun-gupta/wildfly-samples/tree/master/websocket-stomp>

TM
X
O
>
S
D

MQTT over WebSocket

Stomp 



MQTT over WebSocket

Stomp

- Light-weight pub/sub messaging over TCP



MQTT over WebSocket

Stomp

- Light-weight pub/sub messaging over TCP
- Designed for “small foot print” or limited bandwidth



MQTT over WebSocket

Stomp

- Light-weight pub/sub messaging over TCP
- Designed for “small foot print” or limited bandwidth
- MQTT 3.1.1 just became an OASIS standard



MQTT over WebSocket

Stomp

- Light-weight pub/sub messaging over TCP
- Designed for “small foot print” or limited bandwidth
- MQTT 3.1.1 just became an OASIS standard
- Plain byte array message payload



MQTT over WebSocket

Stomp 

- Light-weight pub/sub messaging over TCP
- Designed for “small foot print” or limited bandwidth
- MQTT 3.1.1 just became an OASIS standard
- Plain byte array message payload
- Quality-of-Service: 0 (TCP), 1 (at least once), 2 (missed messages)

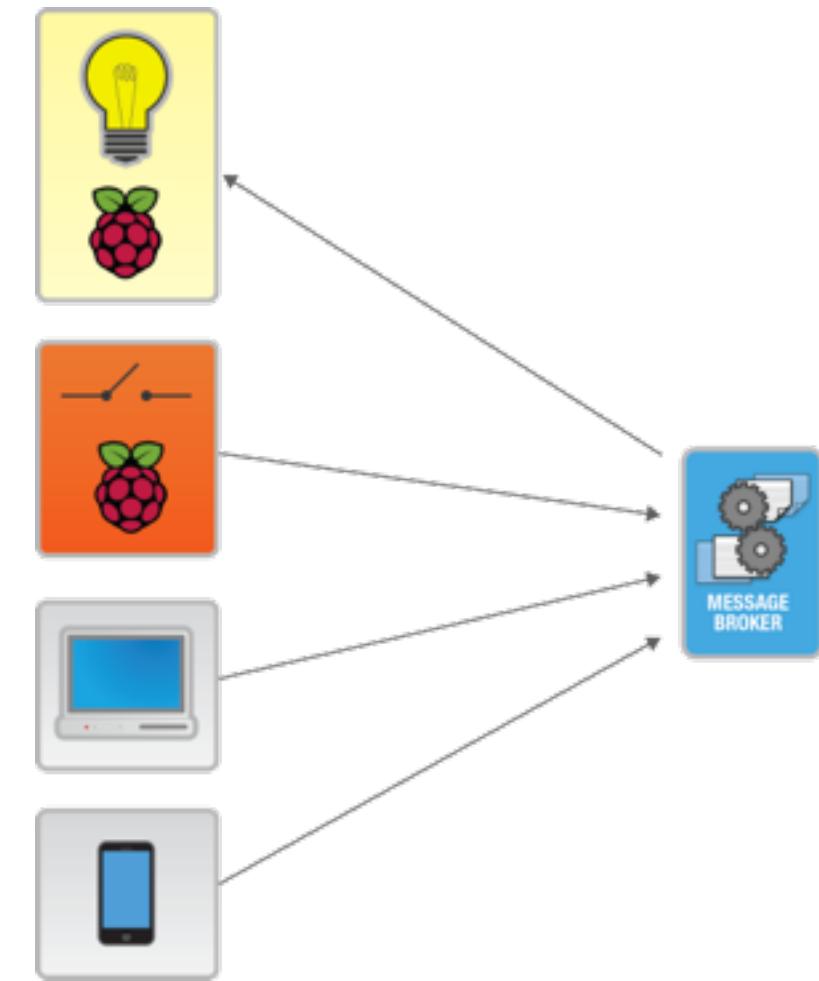


MQTT over WebSocket

Stomp

- Light-weight pub/sub messaging over TCP
- Designed for “small foot print” or limited bandwidth
- MQTT 3.1.1 just became an OASIS standard
- Plain byte array message payload
- Quality-of-Service: 0 (TCP), 1 (at least once), 2 (missed messages)
- “Last will and testament” publish a message after client goes offline

Devoxx™



<http://blog.kaazing.com/2013/10/01/controlling-physical-devices-on-the-real-time-web-kaazing-iot-talk-at-javaone-2013/>



Compare with REST

<https://github.com/javaee-samples/javaee7-samples/tree/master/websocket/websocket-vs-rest-payload>

<https://github.com/javaee-samples/javaee7-samples/tree/master/websocket/websocket-vs-rest>

TM

X
X
O
>
S
D

Server-Sent Events



Server-Sent Events

- Part of HTML5 Specification



Server-Sent Events

- Part of HTML5 Specification
- Server-push notifications



Server-Sent Events

- Part of HTML5 Specification
- Server-push notifications
- Cross-browser JavaScript API: EventSource



Server-Sent Events

- Part of HTML5 Specification
- Server-push notifications
- Cross-browser JavaScript API: EventSource
- Message callbacks



Server-Sent Events

- Part of HTML5 Specification
- Server-push notifications
- Cross-browser JavaScript API: EventSource
- Message callbacks
- MIME type: text/eventstream



EventSource API

```
[Constructor(DOMString url, optional EventSourceInit eventSourceInitDict)]
interface EventSource : EventTarget {
  readonly attribute DOMString url;
  readonly attribute boolean withCredentials;

  // ready state
  const unsigned short CONNECTING = 0;
  const unsigned short OPEN = 1;
  const unsigned short CLOSED = 2;
  readonly attribute unsigned short readyState;

  // networking
  [TreatNonCallableAsNull] attribute Function? onopen;
  [TreatNonCallableAsNull] attribute Function? onmessage;
  [TreatNonCallableAsNull] attribute Function? onerror;
  void close();
};

dictionary EventSourceInit {
  boolean withCredentials = false;
};
```



WebSockets and SSE ?



WebSockets and SSE ?

WebSocket

Server-Sent Event



WebSockets and SSE ?

WebSocket

Over a custom protocol

Server-Sent Event

Over simple HTTP



WebSockets and SSE ?

WebSocket

Over a custom protocol

Full-duplex, bi-directional

Server-Sent Event

Over simple HTTP

Server-push only, client-server OOB



WebSockets and SSE ?

WebSocket

Over a custom protocol

Full-duplex, bi-directional

Native support in most browsers

Server-Sent Event

Over simple HTTP

Server-push only, client-server OOB

Can be poly-filled to backport



WebSockets and SSE ?

WebSocket

Over a custom protocol

Full-duplex, bi-directional

Native support in most browsers

Not straight forward protocol

Server-Sent Event

Over simple HTTP

Server-push only, client-server OOB

Can be poly-filled to backport

Simpler protocol



WebSockets and SSE ?



WebSockets and SSE ?

WebSocket

Server-Sent Event



WebSockets and SSE ?

WebSocket

Application-specific reconnection

Server-Sent Event

Built-in support for reconnection and event id



WebSockets and SSE ?

WebSocket

Application-specific reconnection

Require server and/or proxy configurations

Server-Sent Event

Built-in support for reconnection and event id

No server or proxy change required



WebSockets and SSE ?

WebSocket

Application-specific reconnection

Require server and/or proxy configurations

Text and Binary

Server-Sent Event

Built-in support for reconnection and event id

No server or proxy change required

Text only



WebSockets and SSE ?

WebSocket

Application-specific reconnection

Require server and/or proxy configurations

Text and Binary

Pre-defined message handlers

Server-Sent Event

Built-in support for reconnection and event id

No server or proxy change required

Text only

Pre-defined and arbitrary



What makes them scalable ?



What makes them scalable ?

- No HTTP/TCP opening/closing connections



What makes them scalable ?

- No HTTP/TCP opening/closing connections
 - Handshake over a single TCP connection



What makes them scalable ?

- No HTTP/TCP opening/closing connections
 - Handshake over a single TCP connection
 - HTTP connections have short connection timeout (5 secs for Apache)



What makes them scalable ?

- No HTTP/TCP opening/closing connections
 - Handshake over a single TCP connection
 - HTTP connections have short connection timeout (5 secs for Apache)
- Elimination of HTTP headers (cookies, content-type, user-agent, ...)



What makes them scalable ?

- No HTTP/TCP opening/closing connections
 - Handshake over a single TCP connection
 - HTTP connections have short connection timeout (5 secs for Apache)
- Elimination of HTTP headers (cookies, content-type, user-agent, ...)
 - Reduces bandwidth dramatically



What makes them scalable ?

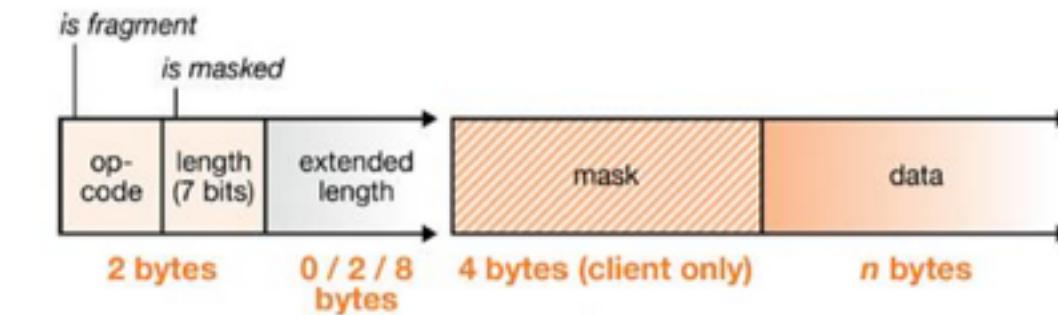


What makes them scalable ?

- Minimal data framing

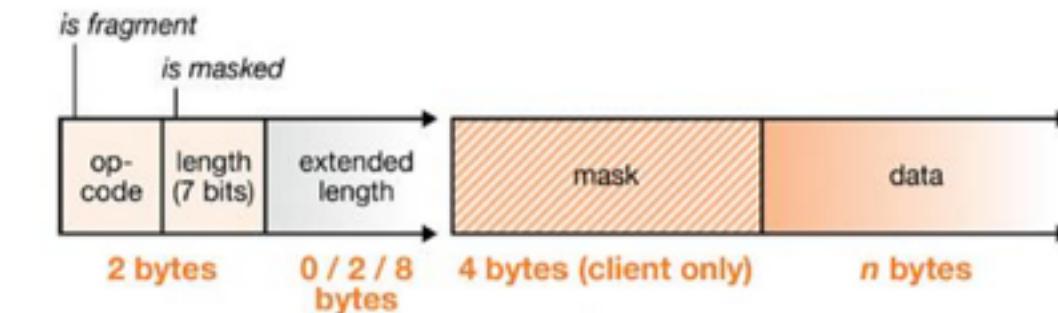
What makes them scalable ?

- Minimal data framing
 - 2-14 bytes overhead after handshake



What makes them scalable ?

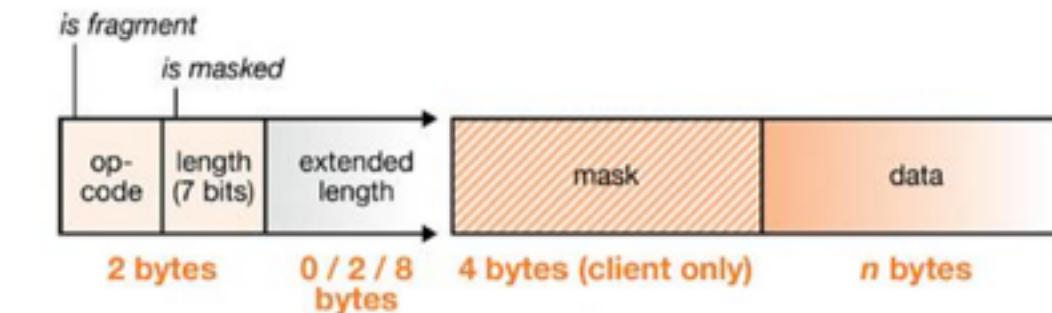
- Minimal data framing
 - 2-14 bytes overhead after handshake
- Maintaining a TCP connection on server is relatively inexpensive





What makes them scalable ?

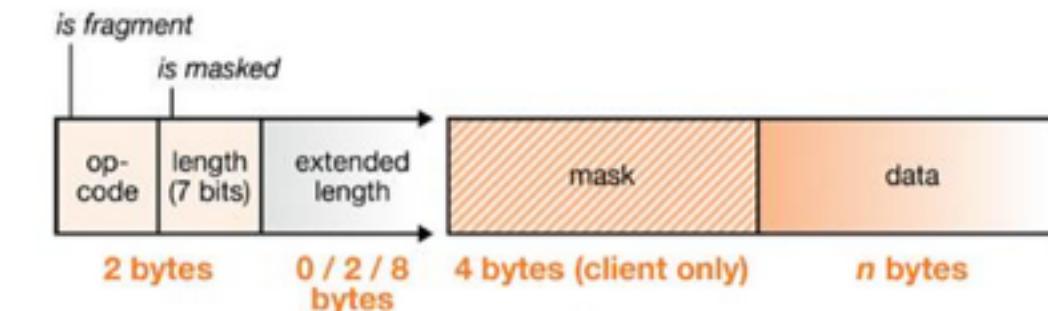
- Minimal data framing
 - 2-14 bytes overhead after handshake
- Maintaining a TCP connection on server is relatively inexpensive
- Smaller data fragments can be sent without out a request/response





What makes them scalable ?

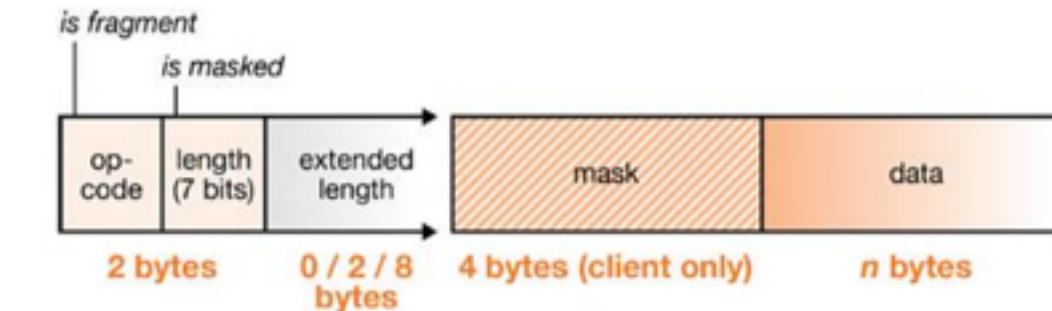
- Minimal data framing
 - 2-14 bytes overhead after handshake
- Maintaining a TCP connection on server is relatively inexpensive
- Smaller data fragments can be sent without out a request/response
 - Live pushes, e.g. stock sticker





What makes them scalable ?

- Minimal data framing
 - 2-14 bytes overhead after handshake
- Maintaining a TCP connection on server is relatively inexpensive
- Smaller data fragments can be sent without out a request/response
 - Live pushes, e.g. stock sticker
 - Lower latency





What makes them scalable ?



What makes them scalable ?

- WebSockets are good at scaling vertically



What makes them scalable ?

- WebSockets are good at scaling vertically
- HTTP servers are typically configured to log start/completion of HTTP request, not so for WebSocket



What makes them scalable ?

- WebSockets are good at scaling vertically
- HTTP servers are typically configured to log start/completion of HTTP request, not so for WebSocket
- Polling and Long-polling is a waste of bandwidth, WebSockets are more elegant



What makes them scalable ?

- WebSockets are good at scaling vertically
- HTTP servers are typically configured to log start/completion of HTTP request, not so for WebSocket
- Polling and Long-polling is a waste of bandwidth, WebSockets are more elegant
- Number of concurrent clients depend upon FD settings



Debugging WebSockets

TM



Elements Network Sources Timeline Profiles Resources Audits Console NetBeans

Preserve log Disable cache

Filter All Documents Stylesheets Images Media Scripts XHR Fonts TextTracks WebSockets Other Hide data URLs

Name Path Headers Frames Cookies

localhost

```

Request URL: ws://localhost:61614/
Request Method: GET
Status Code: 101 Switching Protocols
▼ Request Headers view source
  Accept-Encoding: gzip,deflate, sdch
  Accept-Language: en-US,en;q=0.8
  Cache-Control: no-cache
  Connection: Upgrade
  Cookie: JSESSIONID=214bf52c5ba8b5942e641f2ec4a5; treeForm_tree-hi=
  Host: localhost:61614
  Origin: http://localhost:8080
  Pragma: no-cache
  Sec-WebSocket-Extensions: permessage-deflate; client_max_window_bits
  Sec-WebSocket-Key: 8tJpUUPfH0byKGq0AgcUBA==
  Sec-WebSocket-Protocol: v10.stomp, v11.stomp
  Sec-WebSocket-Version: 13
  Upgrade: websocket
  User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 10_10_0) AppleWebKit/537.36 (KHTML, like Gecko) Chrome

▼ Response Headers view source
  Connection: Upgrade
  Sec-WebSocket-Accept: WFICtNMveMhuj0lLD4LuJKy1sI=
  Sec-WebSocket-Protocol: v10.stomp
  Upgrade: WebSocket

```



Elements Network Sources Timeline Profiles Resources Audits Console NetBeans

Preserve log Disable cache

Filter All Documents Stylesheets Images Media Scripts XHR Fonts TextTracks WebSockets Other

Name Path Headers Frames Cookies

localhost

Data
MESSAGE content-length:4 expires:0 destination:/queue/myQueue subscription:sub-0 priority:4 message
SUBSCRIBE id:sub-0 destination:/queue/myQueue
SEND destination:/queue/myQueue content-length:4 test
CONNECTED server:ActiveMQ/5.10.0 heart-beat:10000,10000 session:ID:Arungs-MacBook-Pro.local-492-
CONNECT login:admin passcode:admin accept-version:1.1,1.0 heart-beat:10000,10000



TM
X
O
>
U
D

← → ⌄ chrome://net-internals/#events&q=type:SOCKET%20is:active

Events ▾ capturing events (4641)

(?) type:SOCKET is:active 3 of 572

	ID	Source Type	Description
1596087	SOCKET	localhost:8080	
1596088	SOCKET	localhost:8080	
1596381	SOCKET	localhost:61614	

1596381: SOCKET
localhost:61614
Start Time: 2014-10-23 10:30:37.952

```
t=143883 [st= 0] +SOCKET_ALIVE [dt=?]
    --> source_dependency = 1596378 (CONNECT_JOB)
t=143883 [st= 0] +TCP_CONNECT [dt=0]
    --> address_list = ["[:1]:61614"]
t=143883 [st= 0] TCP_CONNECT_ATTEMPT [dt=0]
    --> address = "[:1]:61614"
t=143883 [st= 0] -TCP_CONNECT
    --> source_address = "[:1]:57613"
t=143883 [st= 0] +SOCKET_IN_USE [dt=?]
    --> source_dependency = 1596376 (HTTP_STREAM_JOB)
t=143884 [st= 1] SOCKET_BYTES_SENT
    --> byte_count = 616
t=143885 [st= 2] SOCKET_BYTES_RECEIVED
    --> byte_count = 164
t=143886 [st= 3] SOCKET_BYTES_SENT
    --> byte_count = 89
t=143888 [st= 5] SOCKET_BYTES_RECEIVED
    --> byte_count = 134
t=144739 [st= 856] SOCKET_BYTES_SENT
    --> byte_count = 61
t=147073 [st= 3190] SOCKET_BYTES_SENT
    --> byte_count = 54
t=147077 [st= 3194] SOCKET_BYTES_RECEIVED
    --> byte_count = 196
t=153901 [st=10018] SOCKET_BYTES_RECEIVED
    --> byte_count = 3
t=154226 [st=10343] SOCKET_BYTES_SENT
    --> byte_count = 7
```

Debugging WebSockets



No.	Time	Source	Destination	Protocol	Length	Info
11	9.489449000	::1	::1	HTTP	648	GET /HelloWebSocket/ HTTP/1.1
13	9.491601000	::1	::1	HTTP	2134	HTTP/1.1 200 OK (text/html)
18	9.669322000	::1	::1	HTTP	501	GET /HelloWebSocket/echo HTTP/1.1
20	9.669489000	::1	::1	HTTP	543	GET /favicon.ico HTTP/1.1
22	9.670298000	::1	::1	HTTP	205	HTTP/1.1 101 Switching Protocols
24	9.671010000	::1	::1	HTTP	1624	HTTP/1.1 404 Not Found (text/html)
26	12.411987000	::1	::1	WebSocket	98	WebSocket Text [FIN] [MASKED]
28	12.413161000	::1	::1	WebSocket	108	WebSocket Text [FIN]
30	13.011122000	::1	::1	WebSocket	98	WebSocket Text [FIN] [MASKED]
32	13.013172000	::1	::1	WebSocket	108	WebSocket Text [FIN]



Production Tips



Production Tips

- Proxy can be evil and make WebSockets unusable



Production Tips

- Proxy can be evil and make WebSockets unusable
 - Issue: Remove “Upgrade” header



Production Tips

- Proxy can be evil and make WebSockets unusable
 - Issue: Remove “Upgrade” header
 - Fix: Set timeout, remove after onOpen called



Production Tips

- Proxy can be evil and make WebSockets unusable
 - Issue: Remove “Upgrade” header
 - Fix: Set timeout, remove after onOpen called
 - Issue: Close connection after X idle time



Production Tips

- Proxy can be evil and make WebSockets unusable
 - Issue: Remove “Upgrade” header
 - Fix: Set timeout, remove after onOpen called
 - Issue: Close connection after X idle time
 - Fix: Application-level heartbeat



Production Tips

- Proxy can be evil and make WebSockets unusable
 - **Issue:** Remove “Upgrade” header
 - **Fix:** Set timeout, remove after onOpen called
 - **Issue:** Close connection after X idle time
 - **Fix:** Application-level heartbeat
 - **Issue:** Not allow to pass through at all



Production Tips

- Proxy can be evil and make WebSockets unusable
 - Issue: Remove “Upgrade” header
 - Fix: Set timeout, remove after onOpen called
 - Issue: Close connection after X idle time
 - Fix: Application-level heartbeat
 - Issue: Not allow to pass through at all
 - Fix: Fall back on long-polling



Production Tips



Production Tips

- Load Balancer



Production Tips

- Load Balancer
 - Issue: Don't work with WebSocket, e.g. Amazon ELB



Production Tips

- Load Balancer
 - Issue: Don't work with WebSocket, e.g. Amazon ELB
 - Fix: ELB configured to use TCP, but no session affinity



Production Tips

- Load Balancer
 - Issue: Don't work with WebSocket, e.g. Amazon ELB
 - Fix: ELB configured to use TCP, but no session affinity
- Browsers



Production Tips

- Load Balancer
 - Issue: Don't work with WebSocket, e.g. Amazon ELB
 - Fix: ELB configured to use TCP, but no session affinity
- Browsers
 - Issue: IE 6, 7, 8, 9 and Safari 5 do not support WebSocket



Production Tips

- Load Balancer
 - Issue: Don't work with WebSocket, e.g. Amazon ELB
 - Fix: ELB configured to use TCP, but no session affinity
- Browsers
 - Issue: IE 6, 7, 8, 9 and Safari 5 do not support WebSocket
 - Fix: Fallback using Atmosphere, Socket.IO, SockJS



Production Tips

- Load Balancer
 - Issue: Don't work with WebSocket, e.g. Amazon ELB
 - Fix: ELB configured to use TCP, but no session affinity
- Browsers
 - Issue: IE 6, 7, 8, 9 and Safari 5 do not support WebSocket
 - Fix: Fallback using Atmosphere, Socket.IO, SockJS
- Inconsistencies in JSR 356

TM

Atmosphere



X
O
V
W
S
D

A vertical column of large, bold, black-outlined characters: X, O, V, W, S, D, arranged from top to bottom.



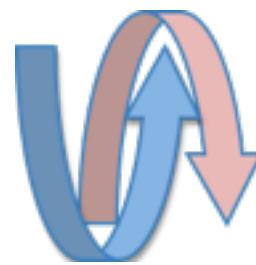
Atmosphere



- Java/JavaScript framework



Atmosphere



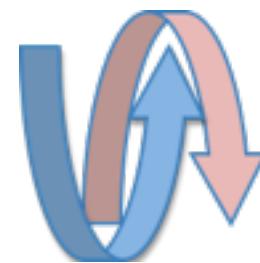
- Java/JavaScript framework
- Portable asynchronous applications



Ш



Atmosphere



- Java/JavaScript framework
- Portable asynchronous applications
- Fallback to long-polling in absence of WebSocket



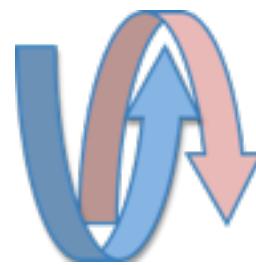
Atmosphere

- Java/JavaScript framework
- Portable asynchronous applications
- Fallback to long-polling in absence of WebSocket
- Containers: Netty, Jetty, GlassFish, Tomcat, JBoss, WildFly, WebLogic, Resin, WebSphere





Atmosphere

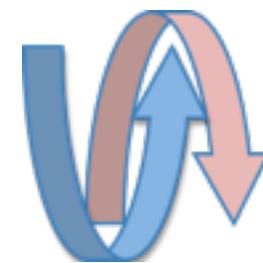


- Java/JavaScript framework
- Portable asynchronous applications
- Fallback to long-polling in absence of WebSocket
- Containers: Netty, Jetty, GlassFish, Tomcat, JBoss, WildFly, WebLogic, Resin, WebSphere
- Browsers: Firefox, Chrome, IE (6x+), Opera, Safari, Android





Atmosphere



- Java/JavaScript framework
- Portable asynchronous applications
- Fallback to long-polling in absence of WebSocket
- Containers: Netty, Jetty, GlassFish, Tomcat, JBoss, WildFly, WebLogic, Resin, WebSphere
- Browsers: Firefox, Chrome, IE (6x+), Opera, Safari, Android

<https://github.com/javaee-samples/javaee7-samples/tree/master/websocket/atmosphere-chat>



Resources

- Material: github.com/arun-gupta/nuts-and-bolts-of-websocket