WebSocket

Salvatore.Loreto@ericsson.com

November 3rd, 2009

WebSocket

- Defines full-duplex communications using a single TCP connection (compared to Comet technologies)
- It is a mechanism for browser-based applications that need two-way communication with servers

WebSocket

- WS is intended to be as close as possible to just exposing raw TCP/IP to JavaScript as possible given the constraints of the Web.

- It is just a layer on top of TCP/IP that adds
  - Web “origin”-based security model for browser
  - Addressing and protocol naming mechanism to support multiple services on one port and multiple host names on one IP address
  - Layers a framing mechanism on top of TCP

WebSocket (Handshake)

- **WebBrowser**(Client) ---> **Server**

GET /text HTTP/1.1\r\n
Upgrade: WebSocket\r\n
Connection: Upgrade\r\n
Host: example.com\r\n
WebSocket-Origin: http://example.com\r\n
WebSocket-Protocol: sample\r\n
…\r\n

The first 3 lines are hard-coded (case and order matters)

The remainder are Unordered ASCII Case-insensitve set of fields

- **WebBrowser(Client)** <--- Server

HTTP/1.1 101 Web Socket Protocol Handshake\r\n
Upgrade: WebSocket\r\n
Connection: Upgrade\r\n
WebSocket-Origin: http://example.com\r\n
WebSocket-Location: ws://example.com/demo\r\n
WebSocket-Protocol: sample\r\n
…\r\n

HTTP **Upgrade** header

- The Upgrade header provides the sender of a message with a means of broadcasting the desire to use another, perhaps completely different, protocol
- If the server is capable, it can send an appropriate response letting the client know that it is okay to use the new protocol. This provides an efficient way to move to other protocols.
- When a server sends a 101 Switching Protocols response, it must include this header.

WebSocket (connection)

- The WebSoket by default uses port 80 for regular Websocket connection;
  Port 80 traffic, however, will often be intercepted by HTTP proxies, which can lead to the connection failing to be established.

- The most realibe method, therefore, is to use TLS encryption and port 443 to connect directly to a WebSocket server.
  More secure, but computationally expansive.

WebSocket (tunnel)

- If the user agent is configured to use an HTTP proxy, then
  - the user agent sends a CONNECT request to the proxy. The CONNECT method asks the proxy to open a TCP connection

    CONNECT example:80 HTTP/1.1
    Host: example.com

  - Once the TCP connection is established, the proxy notifies the user agent by sending

    HTTP/1.1 *200 Connection Established*

    response.

WebSocket (tunnel)

- At this point, the tunnel is set up.

Any data sent by the user agent over the HTTP tunnel will be relayed directly to the outgoing TCPconnection;

and any data sent by the server will be relayed to the user agente over the HTTP tunnel.

WebSocket (Data transfer)

- If the handshake was successful, then the data transfer starts. This is a two-way communication channel where each side can, indeendently from the other, send data

- Data is sent in the form of UTF-8 text.
  Each frame of data starts with 0x00 byte and ends with a 0xFFbyte, with the UTF- text in between.

- The protocol is designed to support other frame types in future. Instead of the 0x00 byte, other bytes might in future be defined.

- Frames denoted by bytes that have the high bit set (0x80 to 0xFF) have a leading length indicator

WebSocket (Data transfer)

- Text Frames use terminator
  - \x00Hello, WebSocket\0xff
- Binary Frames use length prefix
  - \x80\0x10Hello, WebSocket
- Text and binary frames on same WebSocket

WebSocket API

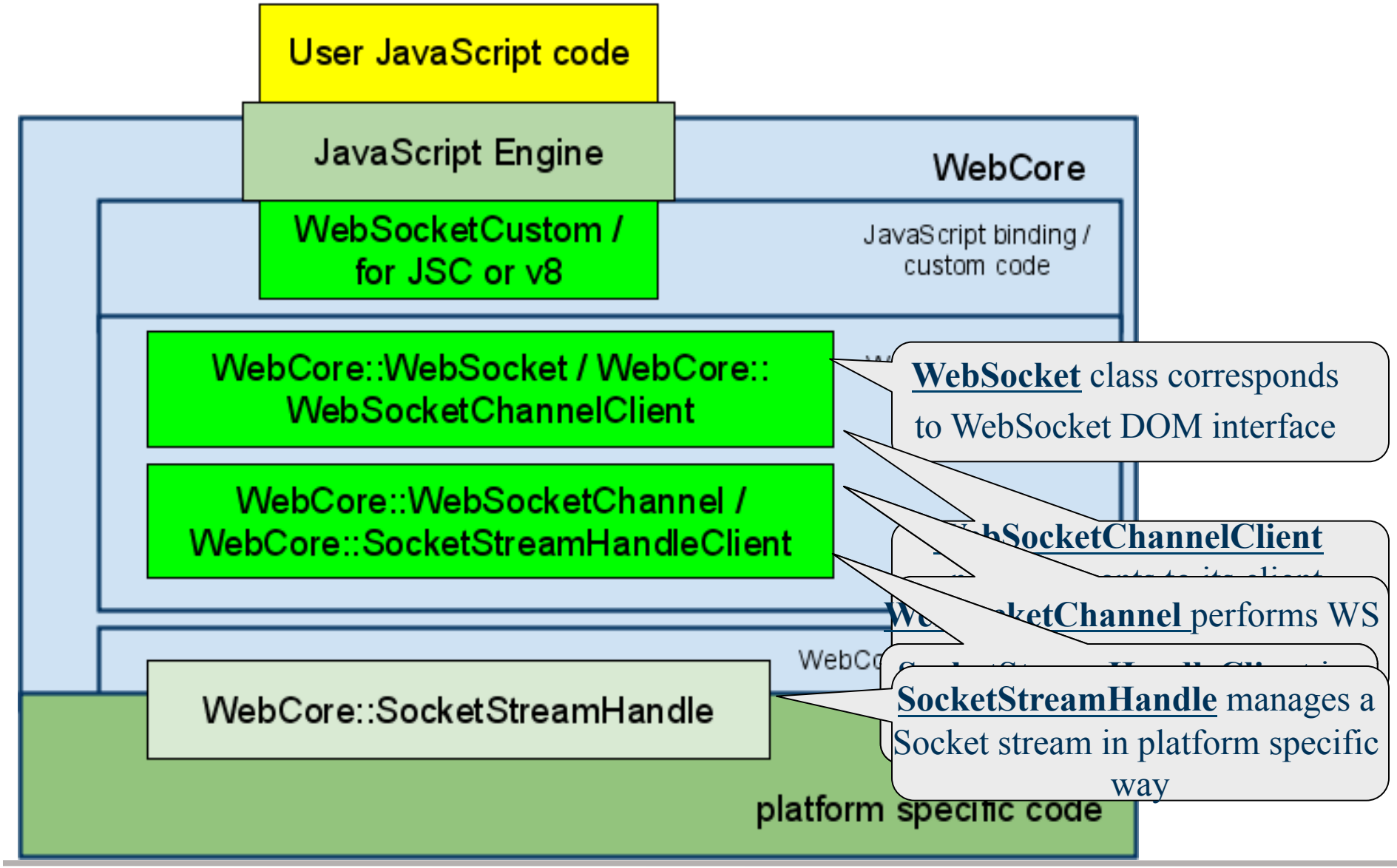
- WebSocket API enables Web pages to use the WebSockets protocol for two-way communication with a remote host.

```
var location = “ws://www.example.org/text”;
var socket = new WebSocket(location);

socket.onopen = function(event) {
                    socket.postMessage(“Hello, WebSocket”); }

socket.onmessage =function(event) { alert(event.data); }

socket.onclose = function(event) { alert(“closed”); }
```

WebSocket scheme: ws, wss

Creating a WebSocket instance

WebSocket Implementation



References

- The Web Sockets API:
  http://dev.w3.org/html5/websockets/
- The Web Socket protocol:
  http://tools.ietf.org/html/draft-hixie-thewebsocketprotocol-54
- Best Practices for the Use of Long Polling and Streaming in Bidirectional HTTP:
  http://www.ietf.org/id/draft-loreto-http-bidirectional-01.txt
- Bidirectional communication for hypertext (HyBi) BoF:
  http://trac.tools.ietf.org/bof/trac/wiki/HyBi