Enabling Full-Duplex Communications in APEX

Me

Curt Workman - workmancw@ldschurch.org

- Education
  - University of Utah

- Work
  - Micron Electronics
  - Evans&Sutherland
  - The Church of Jesus Christ of Latter-Day Saints
- Technology
  - Oracle, Java, Business Objects

Agenda

Browser Enabled Full-Duplex Communication

- Definition
- Use Case
- Implementation Options
- WebSocket
- Apex Integration

Definition

Communication That Is:

- Browser Enabled
- Bi-Directional
- Full duplex
- Asynchronous
- Real Time (low latency)

Definition

A picture is worth a thousand words (Examples)

- https://chat.goinstant.com
- http://prezing.com
- http://demo.kaazing.com/forex/

Use Case

Use Cases for Full-Duplex communication

- Social Feeds
- Multiplayer games
- Collaborative editing/coding
- Clickstream data
- Financial tickers
- Multimedia Chat
- Location based apps
- Online education

Implementation Options

Methods of Implementation

- HTTP
- Polling
- Long Polling
- HTTP Streaming
- Plugins (Applet, Flash, etc.)
- WebSocket

Implementation Options

HTTP

- Request/Response Protocol
- Half Duplex
- Bi-directional
- Stateless

Implementation Options

HTTP

HTTP Request/Response Headers

```
Request URL:http://test.com
Request Method:GET
Status Code:200 OK
Request Headersview source
Accept:text/html, */*; q=0.01
Accept-Charset:ISO-8859-1,utf-8;q=0.7,*;q=0.3
Accept-Encoding:gzip,deflate,sdch
Accept-Language:en-US,en;q=0.8
Authorization:Basic c3Jlbm5vOnNlY3VyaXR5
Connection:keep-alive
Host:test.com
Referer:http://test.com
User-Agent:Mozilla/5.0 (Macintosh; Intel Mac OS X 10_7_1) AppleWebKit/535
X-Requested-With:XMLHttpRequest
Query String Parameters (4)
Response Headersview source
Connection:Keep-Alive
Content-Encoding:gzip
Content-Length:22
Content-Type:text/html; charset=UTF-8
Date:Mon, 19 Sep 2011 19:55:20 GMT
Keep-Alive:timeout=5, max=100
Server:Apache
Vary:Accept-Encoding,User-Agent
X-Powered-By:Servlet/2.5 Server: Sun Java System Application Server
```

- Included in each request/response cycle
- Contains no “application data”

Implementation Options

HTTP Drawbacks

- Half Duplex
- Large overhead for request/response headers

Implementation Options

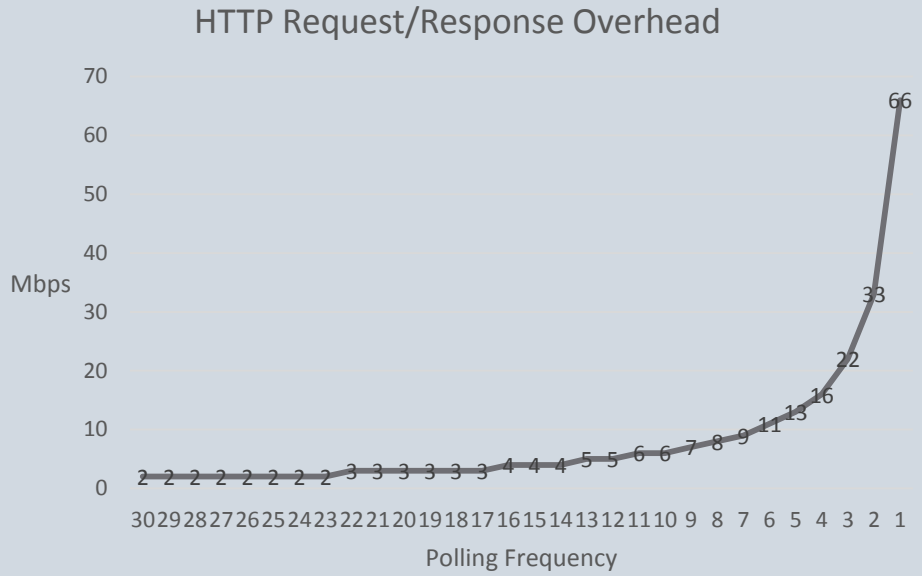
Polling

- Browser periodically polls server via HTTP request
- Server responds even if no message is ready and closes response
- Usually Implemented using XmlHttpRequest JavaScript object
- Bi-directional
- Half Duplex

Implementation Options

Polling Drawbacks

- Overhead of HTTP Request/Response Headers
  - Header size = 800 bytes
  - Client Count = 10,000



- Polling Period needs to be throttled to message frequency.

Implementation Options

Long Polling

- Browser polls server via HTTP request
- Server holds response open until message is ready
- Server sends message and closes response
- Browser initiates new request after receiving response.
- Bi-directional
- Half-duplex

Implementation Options

Long Polling Drawbacks

- Same HTTP request/response headers
- Similar network traffic to regular polling for high frequency messaging

Implementation Options

Streaming

- Browser makes HTTP request
- Server keeps response open via loop construct
- Server sends messages over open response as they occur.

Implementation Options

Streaming Drawbacks

- Client must manage continuous stream of messages over response
  - This consumes memory and must be managed
- Server never completes HTTP response
  - Proxies and firewalls may buffer traffic if response is not closed
- Relatively complex code usually results in adoption of third party frameworks.

Implementation Options

Plugins

- Browser plugins (Silverlight, Java Applets, etc.) are installed in browser.
- Traffic is managed by the browser plugin.
- Full Duplex
- Bi-directional

Implementation Options

Plugin Drawbacks

- Must install plugin
- Not available on all browsers (especially mobile)

Implementation Options

WebSocket

- The holy grail of real time bi-directional full duplex browser based communications (Not Really)
- Overcomes many of the issues with other implementation options.

WebSocket

What is it?

- Standards based communications protocol
  - Internet Engineering Task Force (IETF) RFC 6455
  - W3C WebSocket API
- Component of HTML 5
- Browser Enabled
- Full Duplex
- Bi-directional
- Single Socket
- Real Time (low latency)
- Provides TCP-style network capabilities

WebSocket

IETF RFC 6455 WebSocket Protocol

The protocol has two parts:

- Handshake
- Data Transfer

WebSocket

Protocol opening handshake (Establish Connection)

- Occurs over HTTP request/Response

Request

Required

GET /chat HTTP/1.1
Host: server.example.com
Upgrade: websocket
Connection: Upgrade
Sec-WebSocket-Key: dGhlIHNhbXBsZSBub25jZQ==
Sec-WebSocket-Version: 13

Optional

Origin: http://example.com
Sec-WebSocket-Protocol: chat, superchat
Sec-Websocket-Extensions:
Cookie:

Response

Required

HTTP/1.1 101 Switching Protocols
Upgrade: websocket
Connection: Upgrade
Sec-WebSocket-Accept:
s3pPLMBiTxaQ9kYGzzhZRbK+xOo=

Required

Sec-WebSocket-Protocol: chat
Sec-Websocket-Extensions: extension

WebSocket

Protocol Messages

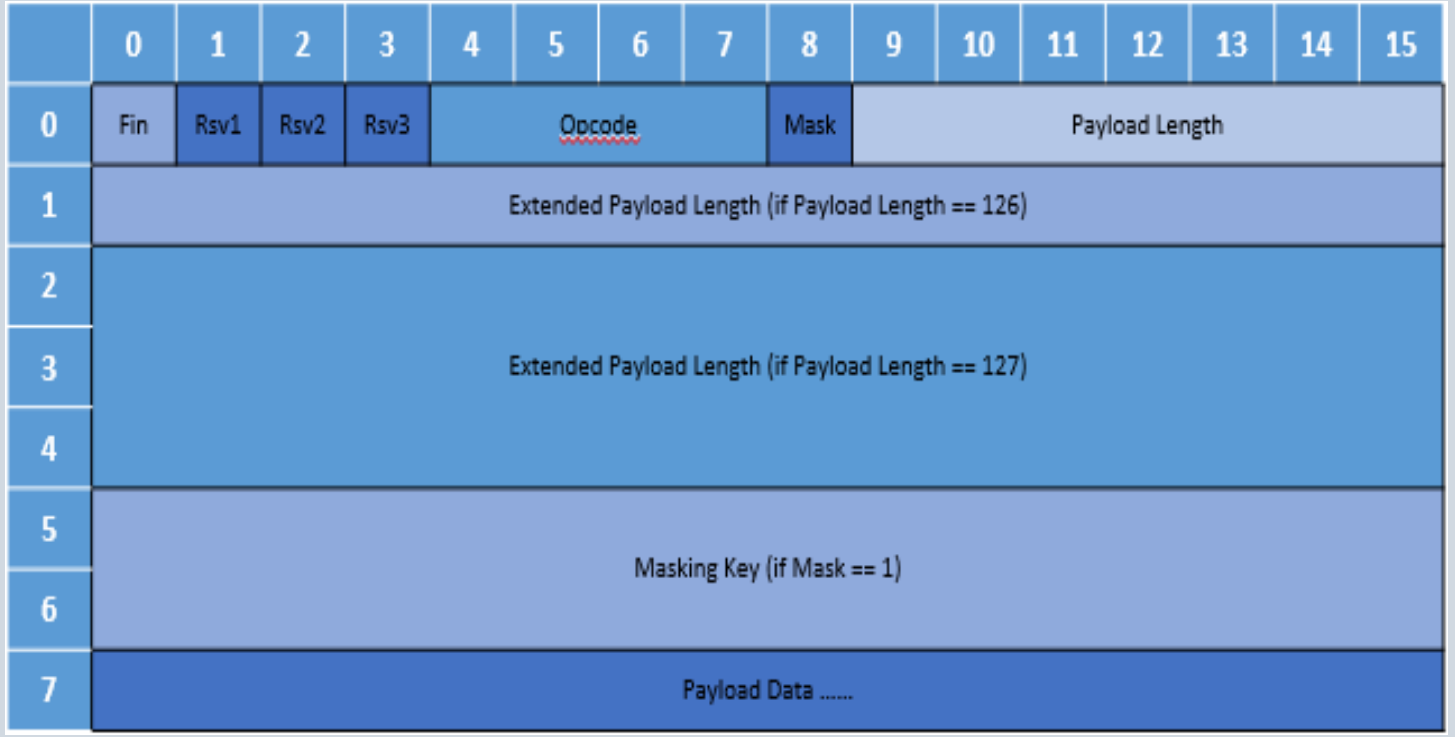
- Once client has established connection via HTTP, WebSocket messages are sent
- A WebSocket message is a logical construct.
  - It is whatever the web application defines as a message.
- Messages are broken into one or more WebSocket frames for transfer

WebSocket

Protocol Frames

A protocol frame has the following structure – Header is only 2 bytes

- Fin – Final Fragment Flag
- Rsv# - Reserved for extensions
- Opcode – Type of Frame
  - Text - 1
  - Binary - 2
  - Close - 8
  - Ping - 9
  - Pong – 10
- Mask – Denotes if message is masked
- Payload length
- Masking Key – Key used to mask data
- Payload Data



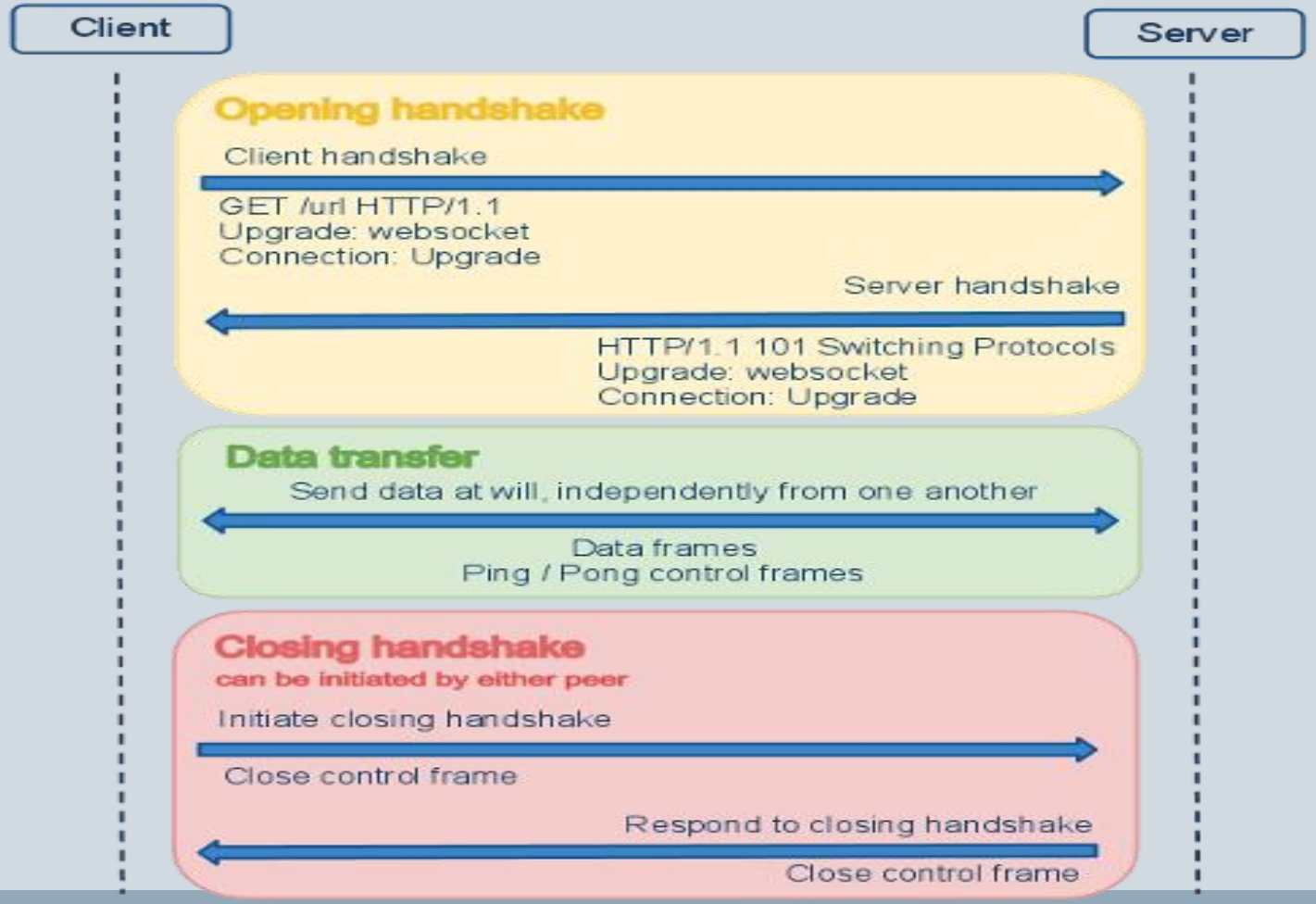
WebSocket

Protocol Closing Message

- Allows for orderly closing of a connection
- May be initiated by either the client or the server
- Is sent as a message with frame type of “close”
- Payload of close frame can supply a close code describing why connection is closing

WebSocket

Protocol Summary



WebSocket

Protocol Server Implementation Options

- Write your own
- Use someone else's
  - Node.js (WebSocket-Node, ws, Socket.io)
  - Java (Jetty)
  - Ruby (EventMachine)
  - Python (pywebsocket, Tornado)
  - Erlang (Shirasu)
  - C++ (libwebsockets)
  - .Net (SuperWebSocket)

WebSocket

W3C WebSocket API

- Standard API that implements the client side of the WebSocket protocol
- Defines a WebSocket Object consisting of:
  - Events
  - Methods
  - Attributes

WebSocket

WebSocket Object – Events

The WebSocket Object dispatches four different events:

- Open – Fires when the server responds to the WebSocket connection request
- Message – Fires when a message is sent from the server
- Error – Fires in response to unexpected errors
- Close – Fires when the WebSocket connection is closed

Each event can be listened for with the following event handlers:

- Open – onopen
- Message – onmessage
- Error – onerror
- Close – onclose

WebSocket

WebSocket Object – Methods

The WebSocket object has two methods:

- Send – Sends a message to the server
- Close – Closes the connection

WebSocket

WebSocket Object – Attributes

The WebSocket Object has three read only attributes:

- readyState – Current state of the connection
  - CONNECTING
  - OPEN
  - CLOSING
  - CLOSED
- bufferedAmount – Number of bytes buffered by the browser.
- Protocol – Protocol negotiated at creation of connection

WebSocket

Simple WebSocket Demo

- Server Implementation
  - ws, http://eiaros.github.io/ws/
  - JavaScript server implementation of the WebSocket protocol
- Client
  - HTML page
  - JavaScript WebSocket Object

Apex Integration

Basic How To

- Create WebSocket server functionality
- Create WebSocket JavaScript client functionality
- Integrate client functionality with UI elements

Apex Integration

Basic WebSocket/Apex Demo

- Server Implementation
  - ws, http://eiaros.github.io/ws/
  - JavaScript server implementation of the WebSocket protocol
  - Server running on Amazon Web Services (AWS)
- Client Implementation
  - APEX application (HTML)
  - JavaScript WebSocket Object
  - Client running on apex.oracle.com

References

Wang, V. Salim, F. Moskovits, P. (2013) *The Definitive Guide to HTML5 WebSocket.* Apress

Stangvik, E http://einaros.github.io/ws

Lindkvist, D. http://www.slideshare.net/ffdead/the-html5-websocket-api

https://goinstant.com

http://prezing.com

http://kaazing.com

Fette, I. https://tools.ietf.org/html/rfc6455#section-5.2

Hickson, I http://www.w3.org/TR/2011/WD-websockets-20110929/#network-intro

http://Apex.oracle.com