HTML5 WEBSOCKETS

Brad Drysdale
*Director of Technology*
*KAAZING*
WebSockets
The Web Communication Revolution

Brad Drysdale
Director of Technology - Kaazing
Single Trader Desktop
Next Generation Web-based…

Gaming/Betting Platform

Single Trader Desktop
Next Generation Web-based…

Gaming/Betting Platform

Single Trader Desktop

Real-time Gambling
Next Generation Web-based…

Smart Metering
Gaming/Betting Platform
Single Trader Desktop
Social Networking
RFID Tracking
eComm
On-line Gaming
Real-time Gambling
Monitoring/Dashboards
IPTV
Logistics & Supply Chain
Reaching the Masses…

WWW
Extending your business across the Web means $$$
Yet you say…

“I can already do this today”
Hang on…

Can you really?
Is your proposed solution…

- Low Latency, Real-time Data?
- Bandwidth Efficient?
- Open Standards?
- Require Plugins? (Note: IE10)
- Platform Neutral?
- Seamless support for Mobile/Tablet OS?
- Cloud Ready?
- Future Proofed?
- Web Scale?
Is your proposed solution…

- Low Latency, Real-time Data?
- Bandwidth Efficient?
- Open Standards?
- Require Plugins? (Note: IE10)
- Platform Neutral?
- Seamless support for Mobile/Tablet OS?
- Cloud Ready?
- Future Proofed?
- Web Scale?
- Truly Web Competitive???
So what’s new…
Here’s how you get Web Competitive
Welcome HTML5

- HTML5 is the next set of W3C HTML standards
- Offers new and enhanced features as building blocks for next generation RIAs
- Industry standard backed by Google, Apple, Mozilla, Microsoft, Cisco, etc
- Many of the browser vendors have already implemented several of these features
- The race is on to implement the rest and be the best
HTML5 Features

- HTML5 features:
  - New forms and media (audio/video) elements
  - New APIs
    - Canvas
    - Web Workers
    - Geolocation
    - Offline storage
    - WebSockets
    - Communication APIs
  - Lots of other cool stuff which is content for a different talk
Let’s revisit the good old days…
Client-Server Architecture

TCP socket

Thick Client

Full duplex

Back-end server

1011
Full duplex transmission of rich business protocols between server to client
Client-Server Architecture

Now let’s extend this to the Web!

Full duplex transmission of rich business protocols between server to client
Out spending again…

Middleware.
Out spending again…
What is this stuff?

Middleware.
Hint is in the name…

Middleware.
HTTP Web Architecture

HTTP

Browser

Internet

Middleware

Back-end server

Half Duplex

Full duplex
Middleware is the glue between HTTP and TCP
HTTP Is Not Full Duplex
Half-Duplex Web Architecture
The Legacy Web Stack

- Designed to serve static documents
  - HTTP
  - Half duplex communication
- High latency
- Bandwidth intensive
  - HTTP header traffic approx. 800 to 2000 bytes overhead per request/response
- Complex architecture
  - Not changed since the 90’s
  - Plug-ins
  - Polling / long polling
  - Legacy application servers
- Expensive to “Webscale” applications
Squeeze every last drop…
• Ajax applications use various “hacks” to simulate real-time communication
  • Polling - HTTP requests at regular intervals and immediately receives a response
  • Long Polling - HTTP request is kept open by the server for a set period
  • Streaming - More efficient, but not complex to implement and unreliable
• Excessive HTTP header traffic, significant overhead to each request response
Hack the Web for Real-Time

Polling

Request
Response

Server

Browser

Time: Requests every n seconds

Long-Polling

Request
Response

Server

Browser

Time: Requests every n seconds

Streaming

Request
Response

Server

Browser

Time: Requests every n seconds

Request Response Overhead

Google Instant search single key press = 649 Bytes
Yahoo single character search = 1432 Bytes
GET /PollingStock//PollingStock HTTP/1.1
Host: localhost:8080
User-Agent: Mozilla/5.0 (Windows; U; Windows NT 5.1; en-US; rv: 1.9.1.5) Gecko/20091102 Firefox/3.5.5
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/q=0.8
Accept-Language: en-us
Accept-Encoding: gzip, deflate
Accept-Charset: ISO-8859-1,utf-8;q=0.7,*;q=0.7
Keep-Alive: 300
Connection: keep-alive
Referer: http://localhost:8080/PollingStock/
Cookie: showInheritedConstant=false;
showInheritedProtectedConstant=false; showInheritedProperty=false;
showInheritedProtectedProperty=false; showInheritedMethod=false;
showInheritedProtectedMethod=false; showInheritedEvent=false;
showInheritedStyle=false; showInheritedEffect=false;
HTTP Response Headers

- Total (unnecessary) HTTP request and response header information overhead: 871 bytes (example)
- Overhead can be as much as 2000 bytes
HTTP Header Traffic Analysis

• Example network throughput for HTTP request and response headers associated with polling
  • **Use case A:** 1,000 clients polling every second:
    • Network throughput is \((871 \times 1,000) = 871,000\) bytes = 6,968,000 bits per second (\(~6.6\ Mbps\))
  • **Use case B:** 10,000 clients polling every second:
    • Network throughput is \((871 \times 10,000) = 8,710,000\) bytes = 69,680,000 bits per second (\(~66\ Mbps\))
  • **Use case C:** 100,000 clients polling every second:
    • Network throughput is \((871 \times 100,000) = 87,100,000\) bytes = 696,800,000 bits per second (\(~665\ Mbps\))
About Ajax and Comet

• Great toilet cleaners…
• Ajax (Asynchronous JavaScript and XML) is used to build highly interactive Web apps
  • Content can change without loading the entire page
  • User-perceived low latency
• "Real-time" often achieved through polling and long-polling
• Comet lack of a standard implementation
• Comet adds lots of complexity
Traditional vs Web

- **Traditional Computing**
  - Full-duplex bidirectional TCP sockets
  - Access any server on the network

- **Web Computing**
  - Half-duplex HTTP request-response
  - HTTP polling, long polling fraught with problems
  - Lots of latency, lots of bandwidth, lots of server-side resources
  - Bespoke solutions became very complex over time
Complexity does not scale
The Web gets a new Superhero
Enter HTML5 WebSocket!
HTML5 WebSocket

- WebSockets provide an improved Web Comms fabric
- Consists of W3C API and IETF Protocol
- Provides a full-duplex, single socket over the Web (even using ports 80 and 443)
- Traverses firewalls, proxies, and routers seamlessly
- Leverages Cross-Origin Resource Sharing
- Share port with existing HTTP content
- Can be secured with TLS (much like HTTPS)
The New Web Architecture

WebSocket

Browser

Internet

Full duplex

Back-end server
The New Web Architecture

Regain the full duplex transmission of rich business protocols between server to client
Regain the full duplex transmission of rich business protocols between server to client, across the Web, across the Cloud.
//Checking for browser support
if (window.WebSocket) {
    document.getElementById("support").innerHTML = "HTML5 WebSocket is supported";
} else {
    document.getElementById("support").innerHTML = "HTML5 WebSocket is not supported";
}
Browser Support for WebSocket

- Chrome
- Safari
- Firefox (need to turn on)
- Opera 10.7 (need to turn on)
- Internet Explorer 9+ Beta
WebSocket Emulation

• Kaazing WebSocket Gateway
  • [http://www.kaazing.com/download](http://www.kaazing.com/download)
  • Makes WebSocket work in all browsers today (including I.E. 6)

• Flash WebSocket implementation
  • Requires opening port on the server's firewall
How do I use: WebSocket API

```javascript
// Create new WebSocket
var mySocket = new WebSocket("ws://www.WebSocket.org");

// Associate listeners
mySocket.onopen = function(evt) {
    alert("Connection open…");
};

mySocket.onmessage = function(evt) {
    alert("Received message: " + evt.data);
};

mySocket.onclose = function(evt) {
    alert("Connection closed…");
};
```
// Sending data
mySocket.send("WebSocket Rocks!")

// Close WebSocket
mySocket.close();
WebSocket Handshake

**Client wants**
**ws://example.com/chat**

**Server accepts**

**REQUIRED**
- GET /chat HTTP/1.1
- Host: server.example.com
- Upgrade: websocket
- Connection: Upgrade
- Sec-Websocket-Key: 16-byte nonce, base64 encoded
- Sec-Websocket-Version: 6

**OPTIONAL**
- Sec-Websocket-Origin: http://example.com
- Sec-Websocket-Protocol: protocol [,protocol]*
- Sec-Websocket-Extensions: extension [,extension]*
- Cookie: cookie content & other cookie-related headers

**REQUIRED**
- HTTP/1.1 101 "Switching Protocols" or other description
- Upgrade: websocket
- Connection: Upgrade
- Sec-Websocket-Accept: 20-byte MD5 hash in base64

**OPTIONAL**
- Sec-Websocket-Protocol: protocol
- Sec-Websocket-Extensions: extension [,extension]*
WebSocket Frames

- Frames have a few header bytes
- Data may be text or binary
- Frames from client to server are masked (XORed w/ random value) to avoid confusing proxies

![Diagram of WebSocket Frame structure]

- mask: 4 bytes (client only)
- opcode: 2 bytes
- length: 2 bytes
- extended length: 2 bytes
- extension data: n bytes
- application data
Reduction in Network Traffic

• With WebSocket, each frame has only several bytes of packaging (a 500:1 or even 1000:1 reduction)
• No latency involved in establishing new TCP connections for each HTTP message
• Dramatic reduction in unnecessary network traffic and latency
• Remember the Polling HTTP header traffic? 665 Mbps network throughput for just headers
### HTTP Header Traffic Analysis

<table>
<thead>
<tr>
<th>Client</th>
<th>Overhead Bytes</th>
<th>Overhead Mbps</th>
</tr>
</thead>
<tbody>
<tr>
<td>1,000</td>
<td>871,000</td>
<td>~6.6*</td>
</tr>
<tr>
<td>10,000</td>
<td>8,710,000</td>
<td>~66</td>
</tr>
<tr>
<td>100,000</td>
<td>87,100,000</td>
<td>~665</td>
</tr>
</tbody>
</table>

* 871,000 bytes = 6,968,000 bits = ~6.6 Mbps
## WebSocket Framing Analysis

<table>
<thead>
<tr>
<th>Client</th>
<th>Overhead Bytes</th>
<th>Overhead Mbps</th>
</tr>
</thead>
<tbody>
<tr>
<td>1,000</td>
<td>2,000</td>
<td>~0.015*</td>
</tr>
<tr>
<td>10,000</td>
<td>20,000</td>
<td>~0.153</td>
</tr>
<tr>
<td>100,000</td>
<td>200,000</td>
<td>~1.526</td>
</tr>
</tbody>
</table>

* 2,000 bytes = 16,000 bits (~0.015 Mbps)
HTTP versus WebSockets

Example: Entering a character in a search field with auto suggestion

<table>
<thead>
<tr>
<th></th>
<th>HTTP traffic*</th>
<th>WebSocket Traffic*</th>
</tr>
</thead>
<tbody>
<tr>
<td>Google</td>
<td>788 bytes, plus 1 byte</td>
<td>2 bytes, plus 1 byte</td>
</tr>
<tr>
<td>Yahoo</td>
<td>1737 bytes, plus 1 byte</td>
<td>2 bytes, plus 1 byte</td>
</tr>
</tbody>
</table>

* Header information for each character entered into search bar

**WebSockets reduces bandwidth overhead up to 1000x**
Polling vs. Web Sockets

- Use Case A: Polling 6,968,000 bits per second, Web Sockets 16,000
- Use Case B: Polling 69,680,000 bits per second, Web Sockets 160,000
- Use Case C: Polling 696,800,000 bits per second, Web Sockets 1,500,000
“Reducing kilobytes of data to 2 bytes…and reducing latency from 150ms to 50ms is far more than marginal. In fact, these two factors alone are enough to make WebSocket seriously interesting to Google.”

—Ian Hickson (Google, HTML5 spec lead)
“The world is moving to HTML5”
—Apple

“The Web has not seen this level of transformation, this level of acceleration, in the past ten years… we're betting big on HTML5”
—Vic Gundotra, VP of Engineering, Google

“In a nutshell, we love HTML5, we love it so much we want it to actually work.”
—Dean Hachamovitch, General Manager for Internet Explorer, Microsoft

“I had no idea there was so much HTML5 already in play”
—Tim O’Reilly
The New Web Stack

- Designed for full-duplex high performance transactional Web
  - HTTP & HTML5 WebSocket
  - Full duplex communication
- Lower latency
- Reduced bandwidth
- Simplified architecture
- Massive scalability
WebSockets Architecture

Browser Support for WebSocket

- Chrome
- Safari
- Firefox (need to turn on)
- Opera 10.7 (need to turn on)
- Internet Explorer 9+ Beta
Server Support

- Kaazing WebSocket Gateway
- Apache mod_pywebsocket
- Jetty
- phpwebsockets
- web-socket-ruby
- Yaws (Erlang)
- Node.js / Socket.io

- This slide is forever out of date…
Got WebSocket!

Now what?
Discovering WebSockets
Got WebSocket. Now What?

• Major upgrade for web traffic, use it!
• Build high performance, scalable messaging for web apps
• Extend the reach of *any* TCP-based protocol you want, all the web to the browser
• The browser is a true client of that protocol – powerful paradigm shift
• Aggregate data and apply business logic at the client
### Example: Financial Apps

![Financial Apps Chart](http://www.kaazing.com/demos/jsStock/stock-jquery.html)

<table>
<thead>
<tr>
<th>Company</th>
<th>Symbol</th>
<th>Price</th>
<th>Change</th>
<th>Sparkline</th>
<th>Open</th>
<th>Low</th>
<th>High</th>
</tr>
</thead>
<tbody>
<tr>
<td>THE WALT DISNEY COMPANY</td>
<td>DIS</td>
<td>27.45</td>
<td>0.36</td>
<td></td>
<td>27.09</td>
<td>24.39</td>
<td>29.80</td>
</tr>
<tr>
<td>GARMIN LTD.</td>
<td>GRMN</td>
<td>34.30</td>
<td>-0.49</td>
<td></td>
<td>34.79</td>
<td>31.32</td>
<td>38.26</td>
</tr>
<tr>
<td>SANDISK CORPORATION</td>
<td>SNDK</td>
<td>18.82</td>
<td>-1.42</td>
<td></td>
<td>20.24</td>
<td>18.22</td>
<td>22.26</td>
</tr>
<tr>
<td>GOODRICH CORPORATION</td>
<td>GR</td>
<td>51.90</td>
<td>-0.44</td>
<td></td>
<td>52.34</td>
<td>47.11</td>
<td>57.57</td>
</tr>
<tr>
<td>NVIDIA CORPORATION</td>
<td>NVDA</td>
<td>13.46</td>
<td>-0.39</td>
<td></td>
<td>13.85</td>
<td>12.47</td>
<td>15.23</td>
</tr>
<tr>
<td>CHEVRON CORPORATION</td>
<td>CVX</td>
<td>67.89</td>
<td>-0.41</td>
<td></td>
<td>68.30</td>
<td>61.48</td>
<td>75.12</td>
</tr>
<tr>
<td>THE ALLSTATE CORPORATION</td>
<td>ALL</td>
<td>32.63</td>
<td>1.61</td>
<td></td>
<td>31.02</td>
<td>27.92</td>
<td>34.11</td>
</tr>
<tr>
<td>EXXON MOBIL CORPORATION</td>
<td>XOM</td>
<td>67.73</td>
<td>1.21</td>
<td></td>
<td>66.52</td>
<td>59.87</td>
<td>73.17</td>
</tr>
<tr>
<td>METLIFE INC.</td>
<td>MET</td>
<td>35.64</td>
<td>-0.09</td>
<td></td>
<td>35.73</td>
<td>32.16</td>
<td>39.30</td>
</tr>
<tr>
<td>J.C. PENNEY COMPANY INC.</td>
<td>JCP</td>
<td>32.66</td>
<td>-0.29</td>
<td></td>
<td>32.95</td>
<td>29.66</td>
<td>36.24</td>
</tr>
<tr>
<td>OFFICEMAX INCORPORATED</td>
<td>OMX</td>
<td>12.22</td>
<td>-0.19</td>
<td></td>
<td>12.41</td>
<td>11.17</td>
<td>13.65</td>
</tr>
<tr>
<td>AETNA INC.</td>
<td>AET</td>
<td>27.30</td>
<td>0.43</td>
<td></td>
<td>26.87</td>
<td>24.19</td>
<td>29.56</td>
</tr>
<tr>
<td>CONOCOPHILLIPS</td>
<td>COP</td>
<td>43.59</td>
<td>-3.03</td>
<td></td>
<td>46.62</td>
<td>41.96</td>
<td>51.27</td>
</tr>
<tr>
<td>UNITEDHEALTH GROUP INC.</td>
<td>UNH</td>
<td>24.43</td>
<td>0.07</td>
<td></td>
<td>24.36</td>
<td>21.93</td>
<td>26.79</td>
</tr>
</tbody>
</table>
Example: Financial Apps

Currency Pairs

USDGBP [↑] 0.4593 [×] 0.4426
GBPUSD [↑] 0.0000 [×] 0.0022
USDJPY 74.9977 [×] 59.0821
USDEUR 0.7056 [×] 0.6171
USDILS 3.8735 [×] 2.0698

Buy ▲ Sell ▲
Buy ▲ Sell ▲
Buy ▼ Sell ▲
Buy ▲ Sell ▲

Spot Prices

USDAXU 2,700
USDAXAG 60

Portfolio Valuations

USDGBP £ 0
USDJPY ¥ 0
USDEUR € 0
USDILS m 0

GBPCHF CHF 0
GBPUSD $ 0
GBPEUR € 0
GBPRUB py.: 0

Market Indices

DOW 15410 +63 +0.4%
NASDAQ 1392 +51 +3.8%
S&P 500 839 +0 +0.0%
FTSE 100 5840 -69 -1.2%
DAX 4015 -99 -2.4%
CAC 40 2276 +7 +0.3%
Nikkel 11431 +133 +1.2%
Hang Seng 15582 -190 -1.2%
Shanghai 2144 -27 -1.2%

News Feed: New York Times

Wall Street's Long History of Protests [+]  
Wall Street Goes to Sefcon II on Swaps [+]  
Congress is Asked to Approve 3 Trade Pacts [+]  
Rhapsody to Acquire Napster in Deal With Best Buy [+]  
Diamondback Beset With Redemptions, Summer Losses [+]  

Executions

no recent trades
WebSocket-Based Quake II

http://code.google.com/p/quake2-gwt-port
Example: HTML5 Frogger

To use your mobile phone as a joystick, please go to http://demo.kaazing.com/frogger/ws and enter the following pin: 7713

Listening to JMS topic: /topic/7713
JMS Received: State=0&0&0.14132948997152392

http://demo.kaazing.com/frogger
Possibilities…

- Low latency Financial and Trading apps
- Online in-game betting and live auctions
- Social networking
- Performance and monitoring dashboards
- RFID and GPS Tracking
- Sports and news broadcasting applications
- Supply chain and inventory management
- Smart meters
- Next generation web application of your choice!
Your cool [HTML5 WebSocket] App Here...

http://iseeaday.blogspot.com/
Unconstrained Web
- Financial Services
- Transportation and Logistics
- Telecommunications
- Utilities
- Social Networking

Cloud Computing
- Server to Server communication
- Distributed Internet applications over any TCP protocol
- Services on demand

3G & 4G Mobile Networking
- Significant bandwidth reduction
- New Service Delivery
- New Customer Experience