The 3 key features of WebRTC

Research Group: Telecommunications Software and Systems Group (TSSG)

Address: Waterford Institute of Technology, West Campus, Carriganore, Waterford, Ireland

Authors: Brendan O'Farrell

Email: bofarrell@tssg.org

Rev Date: July 06, 2012

Rev Number: N/A

Date: July 06, 2012

Description:

This document will look at the key features of WebRTC: Media streams, Peer connection and Data channels.

Table of Contents

1. Key features of WebRTC

The 3 key fetures of WebRTC

- Media streams (getUser media): This is used to gain access to the users camera and microphone. It can also be used to design WebRTC applications.
- Peer connection: This is the engine required to make high quality audio visual calls on the web.
- Data channels: The specification for data channels has yet to ratified. It will be used for such things as, file transfer, games, co-browsing, shared whiteboard, shared document editing and more.

Media Integration Coding examples [https://dvcs.w3.org/hg/audio/raw-file/tip/webaudio/webrtc-integration.html]

1.1. Media streams

A media stream represents a media source, containing one or more synchronized media stream tracks.It can be converted to an object URL and passed to a <video> element. You use the getUser media API to get a media stream for the webcam and microphone. The user is then prompted to allow their consent, an example of this can be seen in Opera 12 where a text box drops down and requests permission from the user to allow use of their camera.

Some examples of using the getUser media. The photobooth app allows you take a photograph of the video image and add different effects to it. This app is great fun for the consumer but commercially it can be used for a multitude of reasons. You could be looking to by a car, the calle can be showing you the car live using his smart phone and you can take snapshots of say the engine number or some damage that might require a quote to repair. This is only the tip of the iceberg.

Another example is face regonisition. We could do away with passwords for online banking, social media sites, online shopping or any interaction which requires a password to gain access to an account. These are only two examples of using getUser media, we can expect to see the number of these apps multiply daily.

IMAGE HERE OF MEDIA STREAM

1.2. Peer connection

This is the engine behind making high quality audio/visual calls on the web.Peer connection allows us to the take the media stream and send it across the web peer to peer. The actual code that implements

PeerConnection is now a part of libjingle [https://developers.google.com/talk/libjingle/]. While PeerConnection has no session protocol and no XMPP/Jingle is required, we've reused many useful components from the libjingle package.WebRTC [http://www.webrtc.org/reference/api-description]

Peer connection is the API for establishing audio video calls (sessions).

Some built in features:

- It establishs peer to peer links
- Manages all the various audio and video codecs
- Sets the encryption
- Tunes the audio video streams to make better use of the bandwidth

Peer connection calling sequences

WebRTC API description [http://www.webrtc.org/reference/api-description]

1.3. Data channels

A data channel is a peer to peer exchange of arbitrary application data. It has low latency, high message rate/throughput and optional unreliable semantics.

Key features of data channels:

- Leverages peer connection session setup
- Multiple simultaneous channels, with priortization
- Reliable and unreliable delievery semantics
- Built in secuirity (DTLS)
- Congestion control
- Can be used with or without audio and video
- Similar API to wesockets

A sample of real world use cases:

- In gaming if you need to send data about say positions, directions. It is more efficent to send them over a peer to peer connection than over HTTP
- Real time text.An example of this is sending code or a process to an engineer who is out on site

- File transfer, no more having to drive to your accounts with boxes of paper
- Remote desktop applications
- Decentralized networks, you can communicate on a private encrypted channel.

At present the specification for data channels has yet to ratified, here are some inital proposals and an interim report.

Data channels WebRTC [http://dev.w3.org/2011/webrtc/editor/webrtc.html#datachannel]

Interim report on data channels from Randell Jesup/IETF [http://lists.w3.org/Archives/Public/public-webrtc/2012Jun/att-0063/W3_Interim_June_2012_Data_Channel.pdf]