Real-time With WebRTC

Lightning Talk by Luis Rosario

What is WebRTC?

WebRTC is a free, open project that provides browsers and mobile applications with Real-Time Communications (RTC) capabilities via simple APIs. The WebRTC components have been optimized to best serve this purpose.

Supported Platforms

The WebRTC javascript API is as low level as you can get with real time communication.

Most companies use other 3rd party libraries to Interact with WebRTC API.

Note:
If you are interested in being a WebRTC developer, you will want to know how to interact with WebRTC API directly before abstracting away with any 3rd party libraries.

Some Companies That Use WebRTC:

Google Hangouts

Facebook Messenger

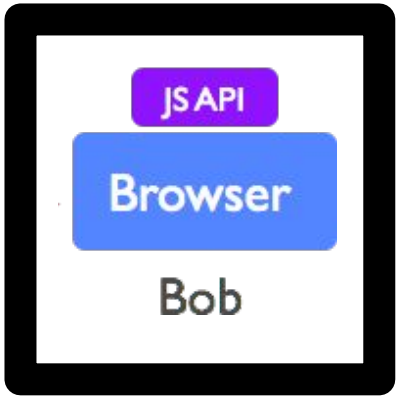
Amazon Chime

Goto Meeting

Zoom

PEER TO PEER COMMUNICATION PART 1

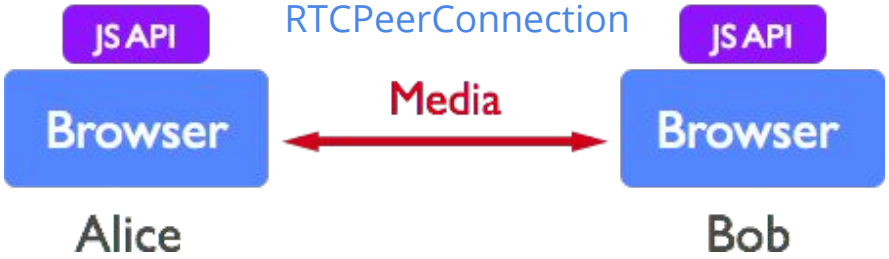
Peer to peer communication is done directly through the browser known as the client. Each client can connect to the WebRTC API and interact with camera and audio inputs on the local machine.



PEER TO PEER COMMUNICATION PART 2

In order for Alice and Bob to interact with each other, a peer connection must be established. This is initiated by interacting with the WebRTC API and creating a RTCPeerConnection.

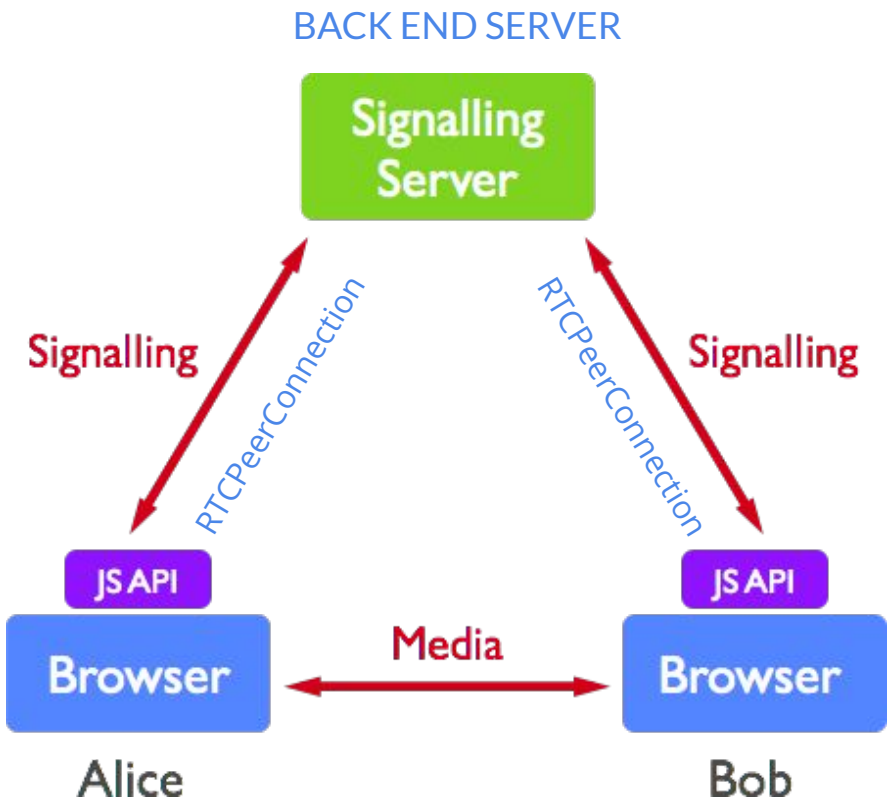
Note: This is a basic example for a two-way connection. Most production apps require multi peer connections.



PEER TO PEER COMMUNICATION PART 3

Most production applications utilize a signaling server in order to handle multiple peer connections similar to how chat rooms behave. A signaling server will utilize some type of websocket technology for the communication to backend of the server.

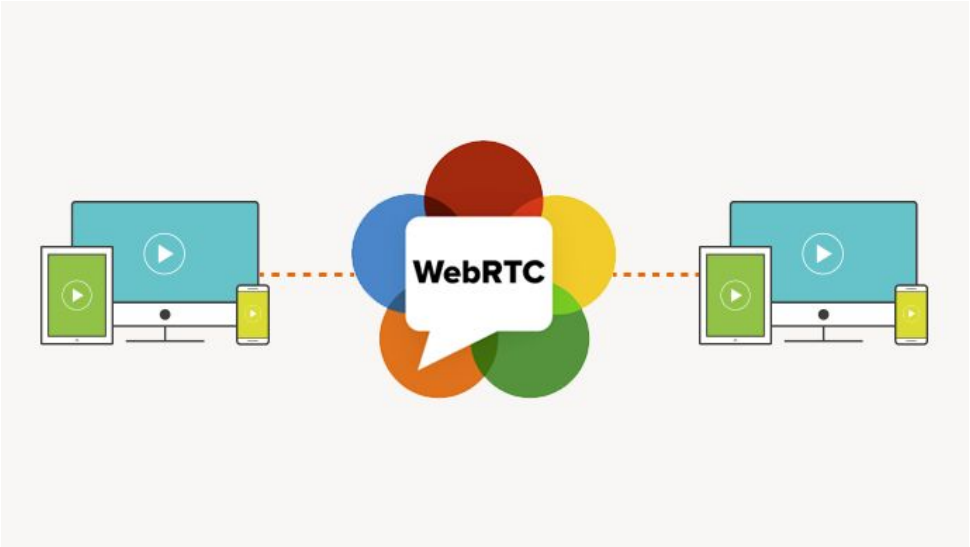
Ex. Socket IO



PEER TO PEER COMMUNICATION PART 4

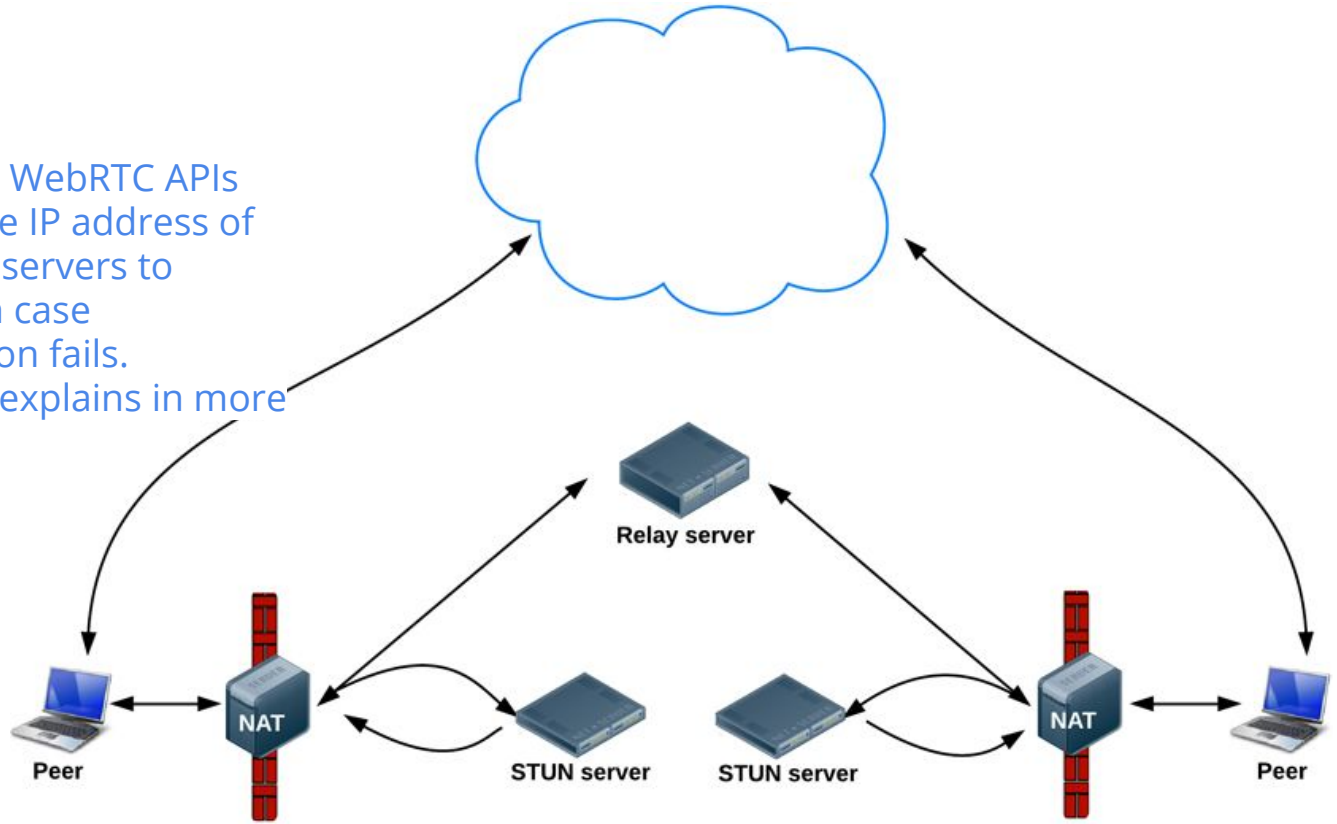
WebRTC is designed to work peer-to-peer, so users can connect by the most direct route possible.

However, WebRTC is built to cope with real-world networking: client applications need to traverse NAT gateways and firewalls, and peer to peer networking needs fallbacks in case direct connection fails.



PEER TO PEER COMMUNICATION PART 5

As part of this process, the WebRTC APIs use STUN servers to get the IP address of your computer, and TURN servers to function as relay servers in case peer-to-peer communication fails. (WebRTC in the real world explains in more detail.)



Common Methods In Use When Accessing the API

getUserMedia()

capture audio and video

RTCPeerConnection

stream audio and video between users

RTCDataChannel

stream data between users

Download Repo

Tutorial Source: CodeLabs

https://github.com/lr001dev/webRTCLightningTalk

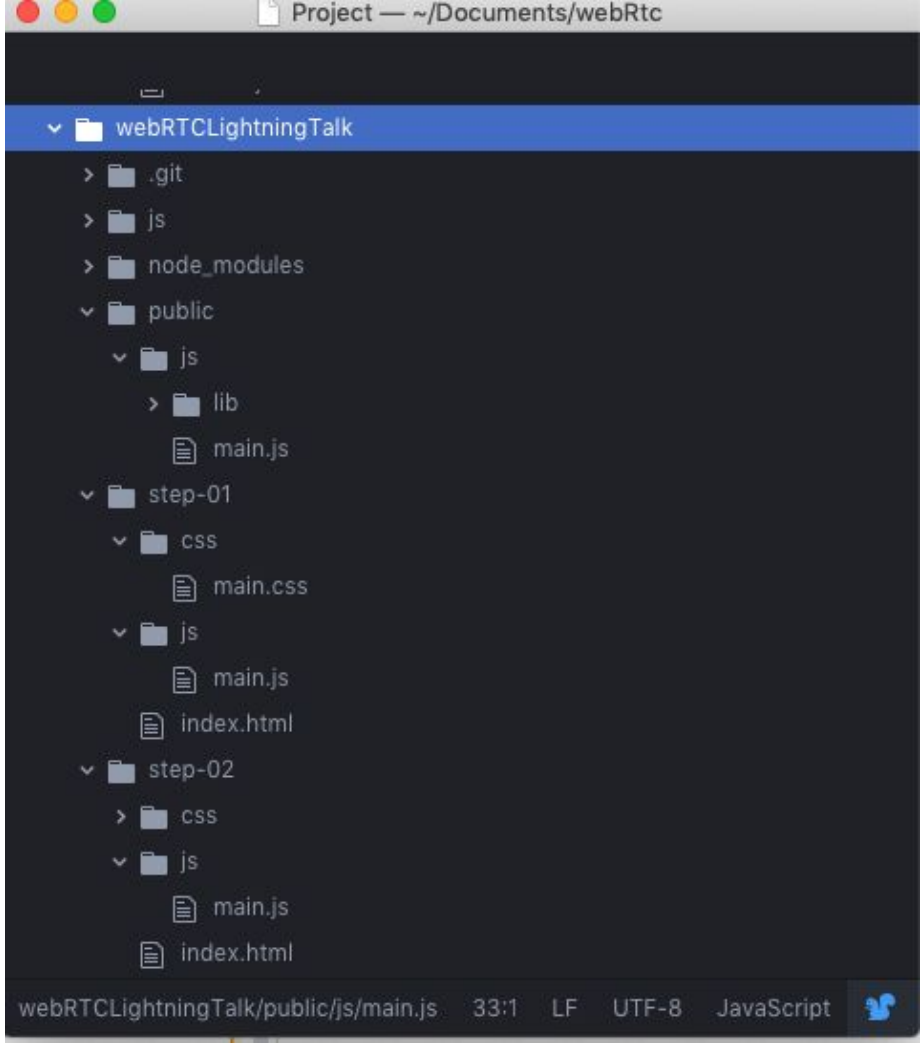
Public/js/main.js is where you place any code to test.

Copy code from step-01/js/main.js or step-02/js/main.js to

Public/js/main.js

to run each example

Step 1 is loaded by default



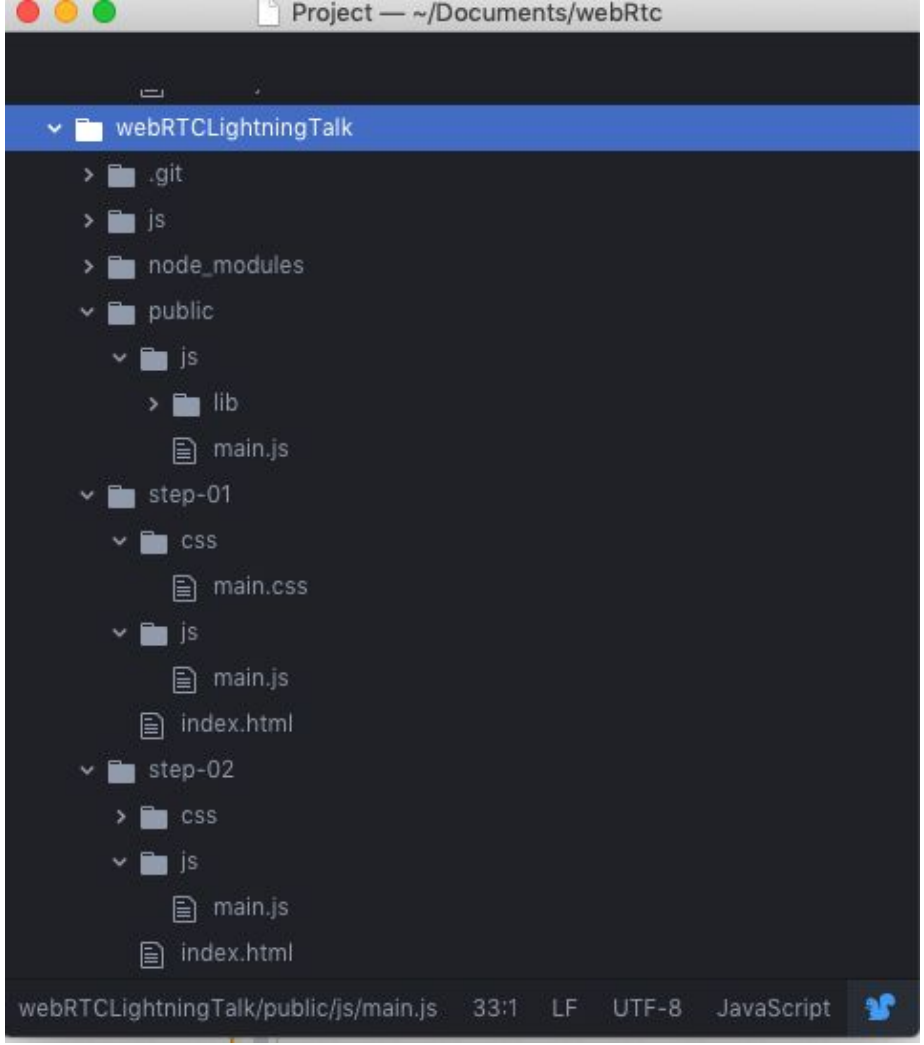
Inside webRTCLightningTalk run

>> nodemon server.js

Visit localhost:3000

Step 1 allows you to take control of your webcam

Step 2 Establishes an RTCPeerConnections for two way communication



RESOURCES

Dive in to WebRTC API:

WebRTC Open Source Project

Good Blog About WebRTC:

https://www.html5rocks.com/en/tutorials/webrtc/basics/

Full CodeLabs Tutorial:

CODELABS

Active Scalable Open Source 3rd Party Library:

MEDIASOUP

Developer Driven White Label Pay As You Go Service:

TOKBOX