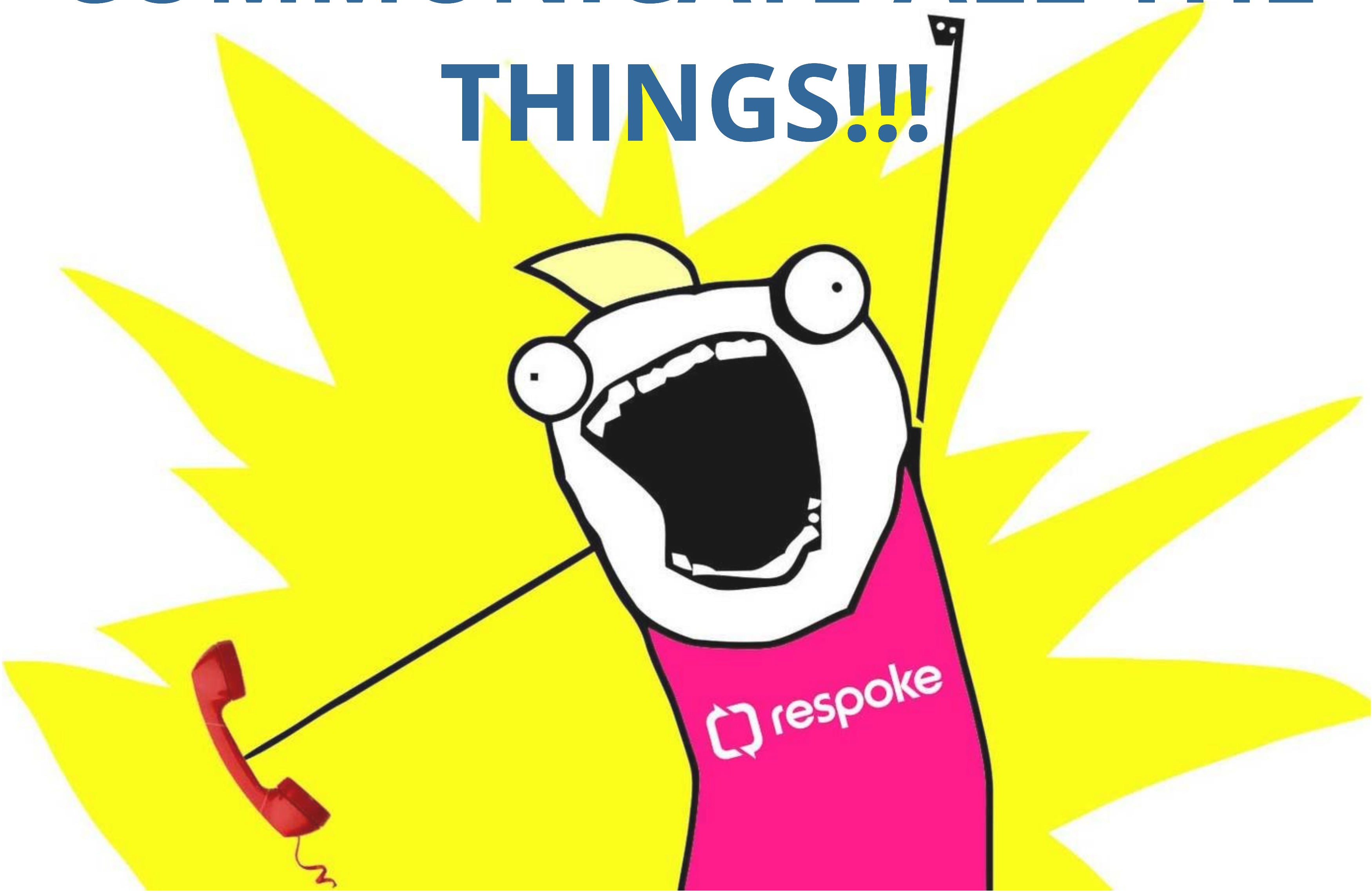
COMMUNICATE ALL THE THINGS!!!



Follow Along

http://goo.gl/v3aHL9

What are we going to cover?

WebRTC

Respoke

Code Samples

</QA>

#boisecodecamp

@geekgonenomad

Who Is This Guy?

Kyle Tyacke

Flash / Web Dev

Developer Evangelist Respoke

Beer Geek

Who Are You?

What the Heck Is WebRTC?

WebRTC

“is a free, open project that enables web browsers with Real-Time Communications (RTC) capabilities via simple JavaScript APIs.”

WebRTC

“is a free, open project that enables web browsers with Real-Time Communications (RTC) capabilities via simple JavaScript APIs.”

WebRTC

"is a free, open project that enables web browsers with Real-Time Communications (RTC) capabilities via simple JavaScript APIs."

WebRTC

"is a free, open project that enables web browsers with Real-Time Communications (RTC) capabilities via simple JavaScript APIs."

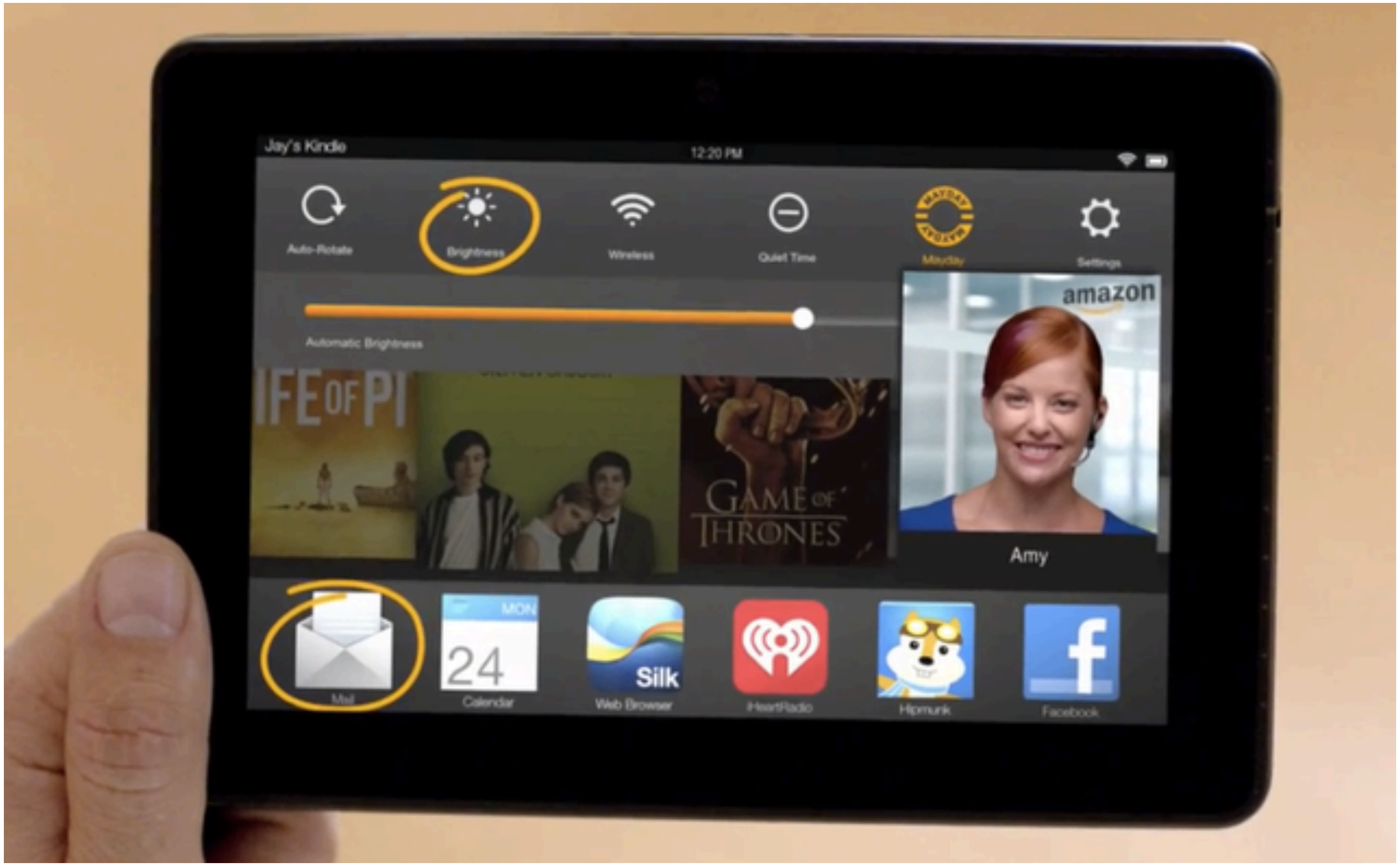
WebRTC

"is a free, open project that enables web browsers with Real-Time Communications (RTC) capabilities via simple JavaScript APIs."

Contributors

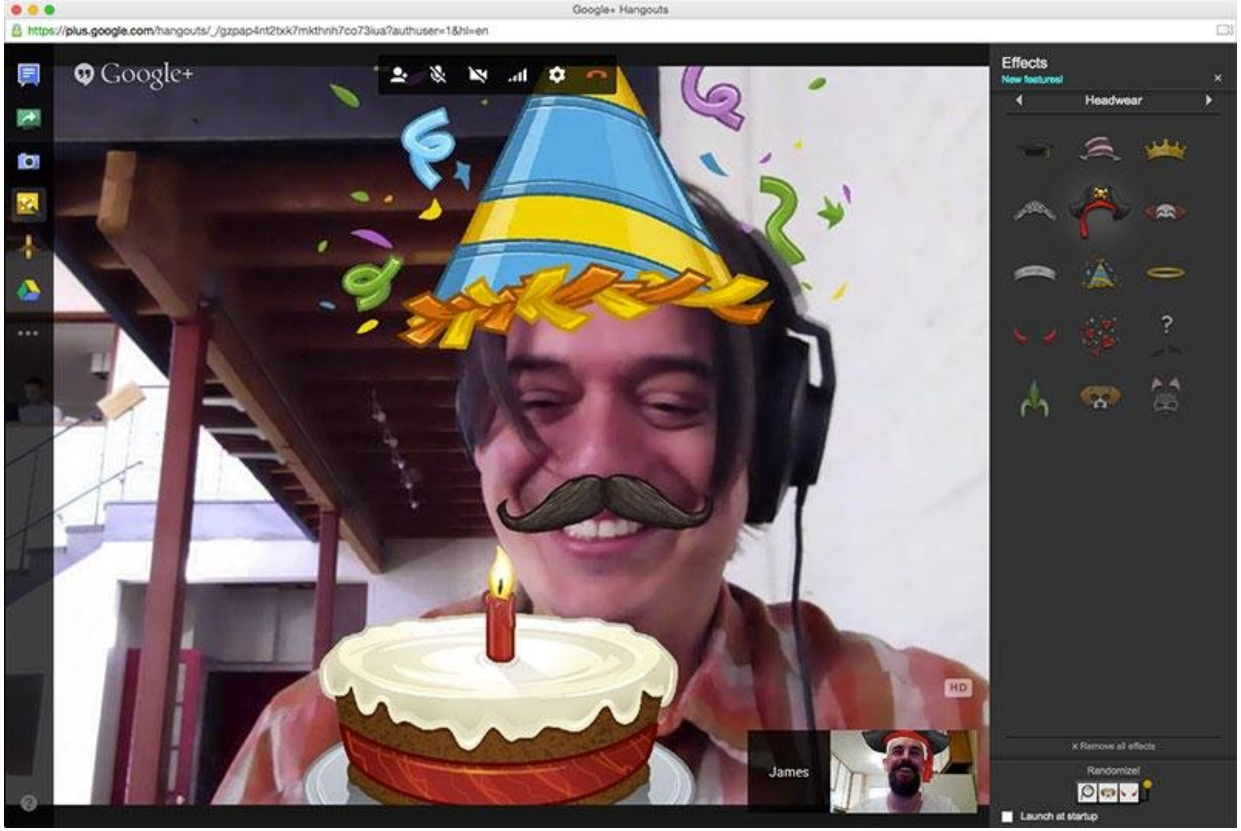
WebRTC in the Wild

Amazon Mayday



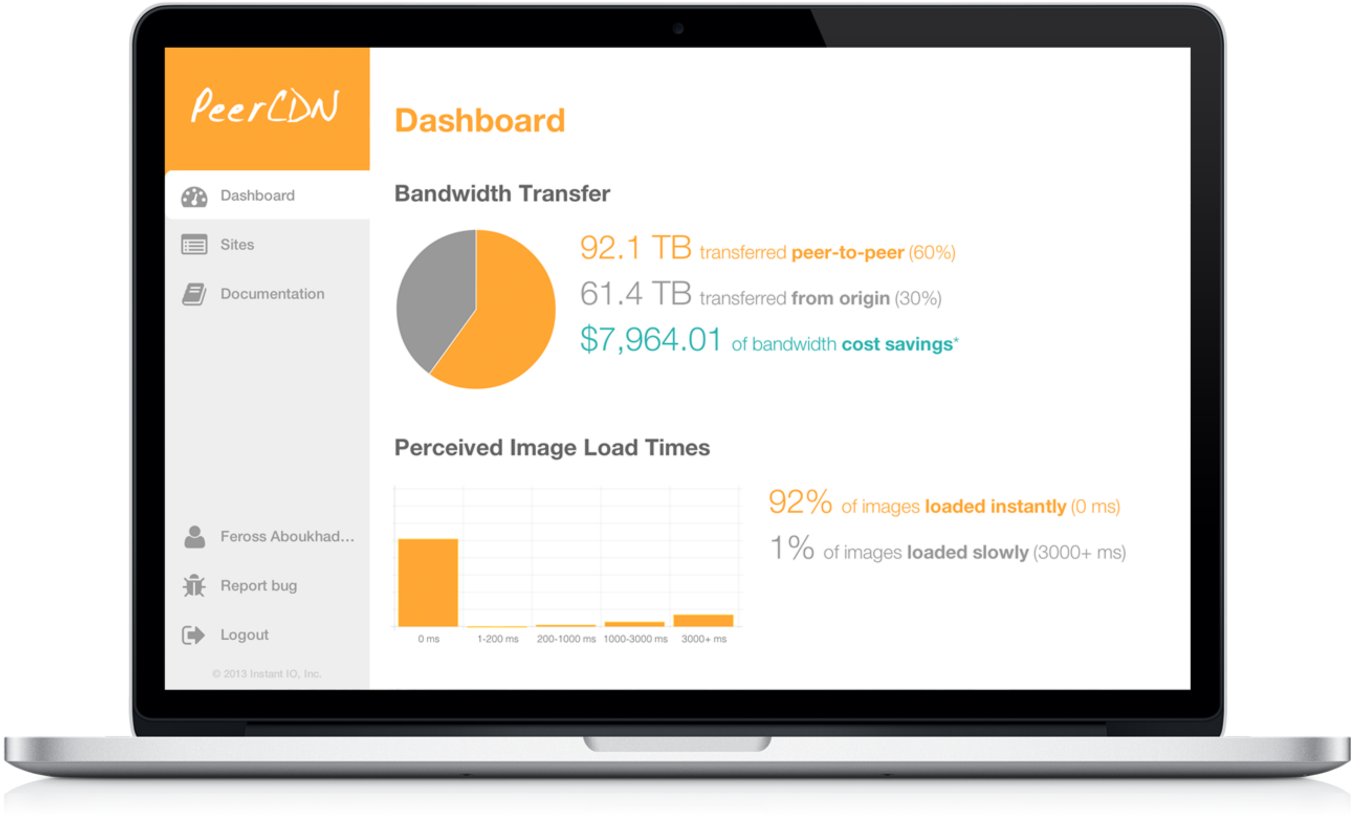
GoogleHangouts

https://plus.google.com/hangouts



PeerCDN

https://peercdn.com/



Dashboard

Bandwidth Transfer

92.1 TB transferred **peer-to-peer** (60%)

61.4 TB transferred **from origin** (30%)

$7,964.01 of bandwidth **cost savings***

Perceived Image Load Times

92% of images **loaded instantly** (0 ms)

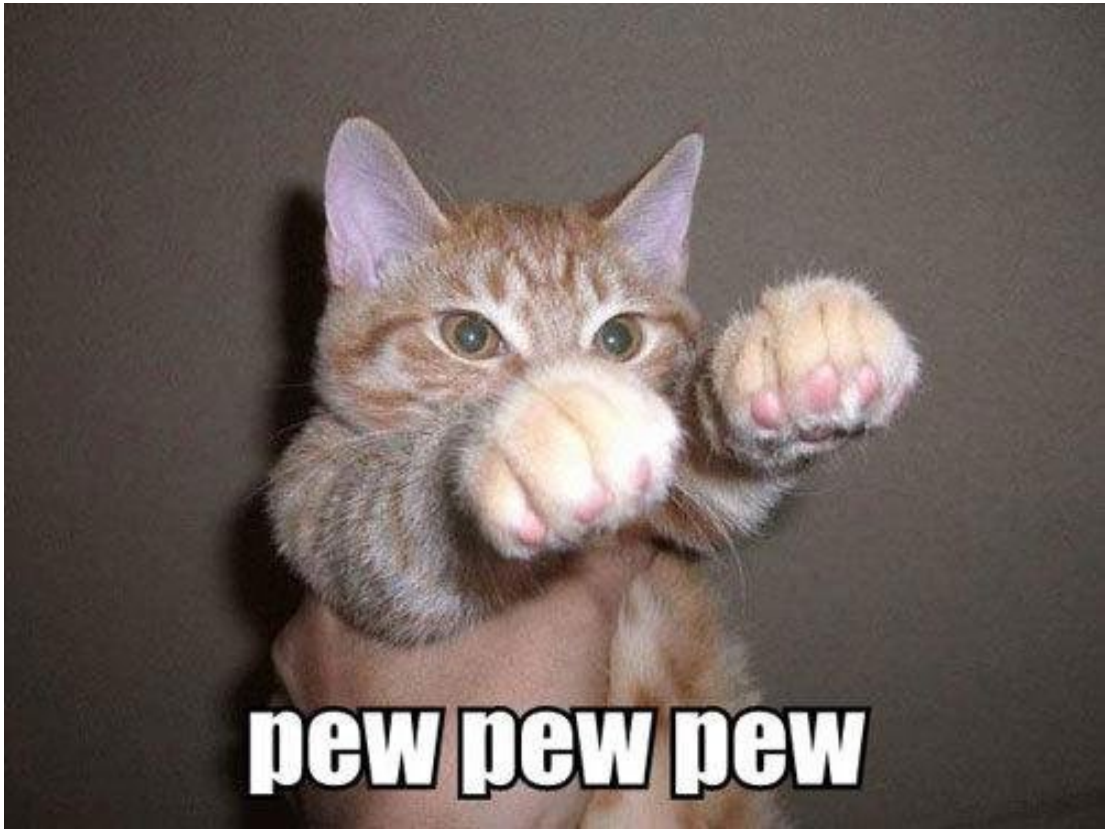
1% of images **loaded slowly** (3000+ ms)

#boisecodecamp

@geekgonenomad

Sharefest

https://sharefest.me



Cubeslam

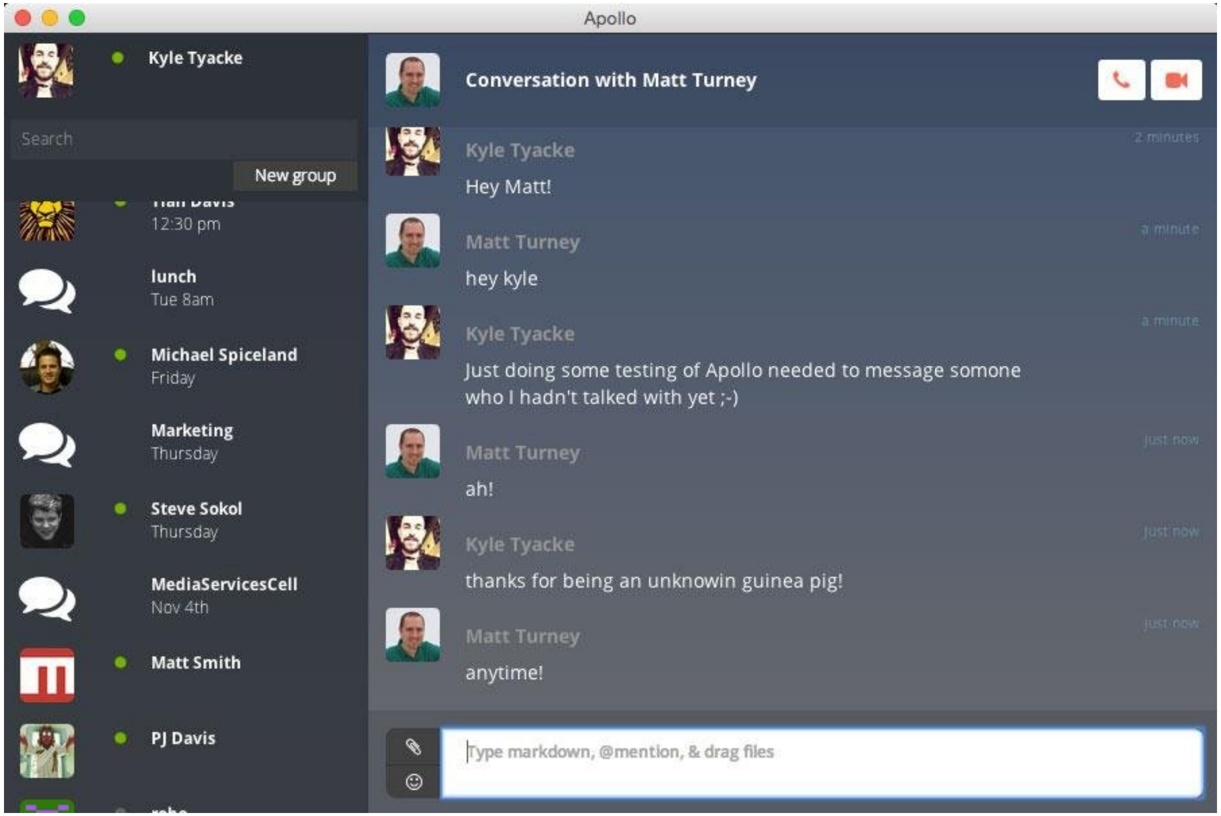
https://www.cubeslam.com

WebRTC

Cube Slam is built using WebRTC, an open web technology that lets you video chat with your friends right in the browser, no plug-ins necessary. The getUserMedia API provides access to your webcam and microphone, RTCPeerConnection sends the audio and video to your friends, and RTCDataChannel exchanges all the bits and pieces that keep the game in sync. When there are no firewalls in the way, game data can be sent directly peer-to-peer, greatly reducing server costs.

Apollo

https://github.com/respoke/apollo



Conversation with Matt Turney

Kyle Tyacke 2 minutes
Hey Matt!

Matt Turney a minute
hey kyle

Kyle Tyacke a minute
Just doing some testing of Apollo needed to message somone who I hadn't talked with yet ;-)

Matt Turney just now
ah!

Kyle Tyacke just now
thanks for being an unknowin guinea pig!

Matt Turney just now
anytime!

Type markdown, @mention, & drag files

How Does WebRTC Work?

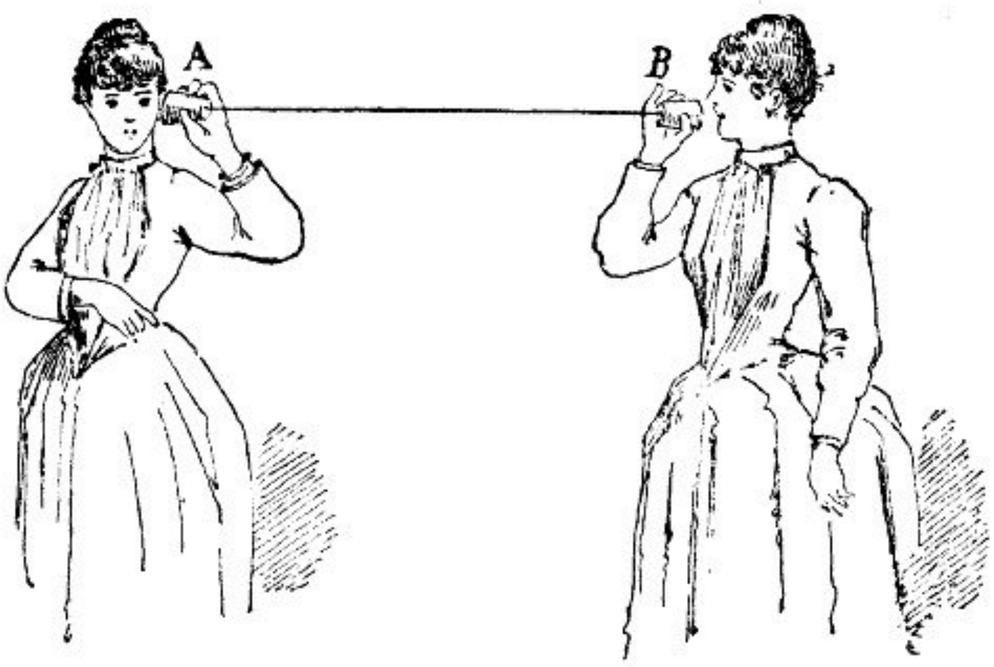
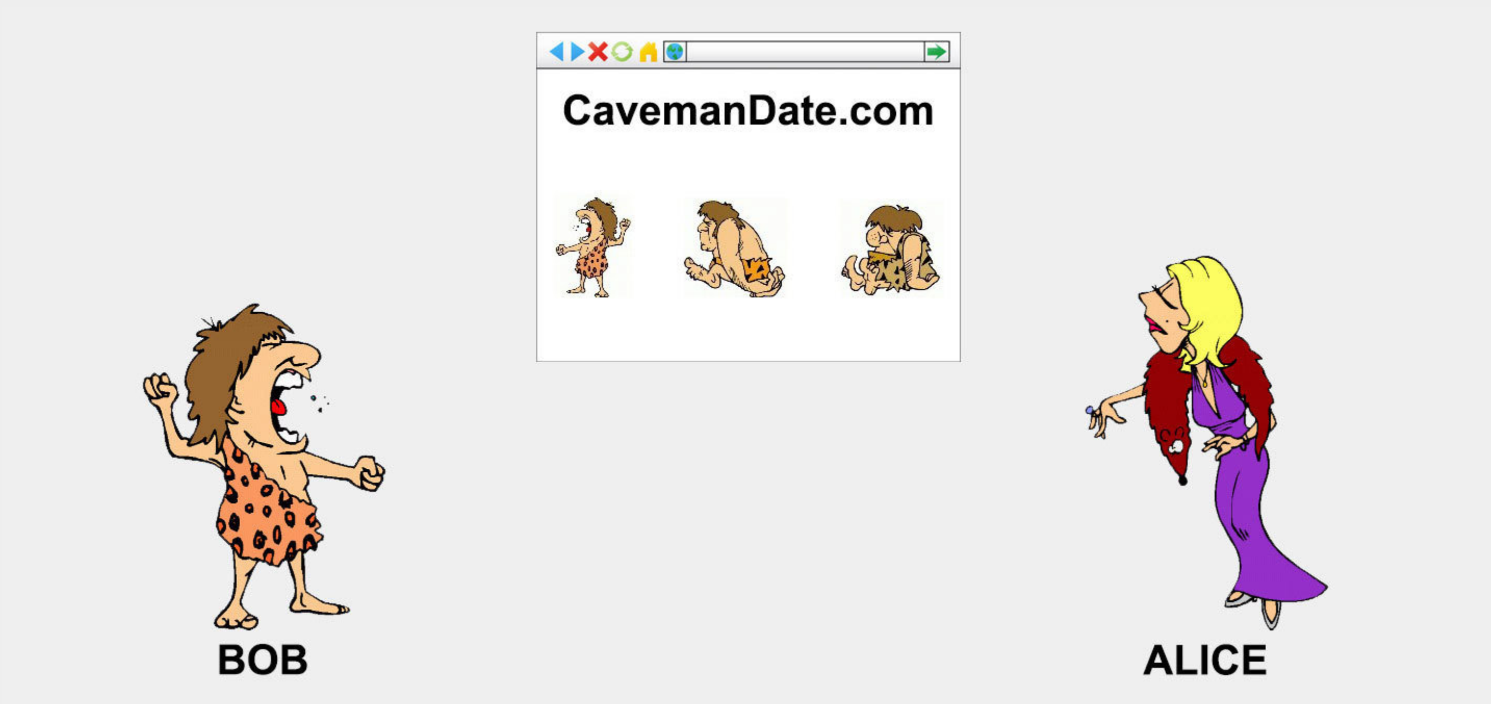
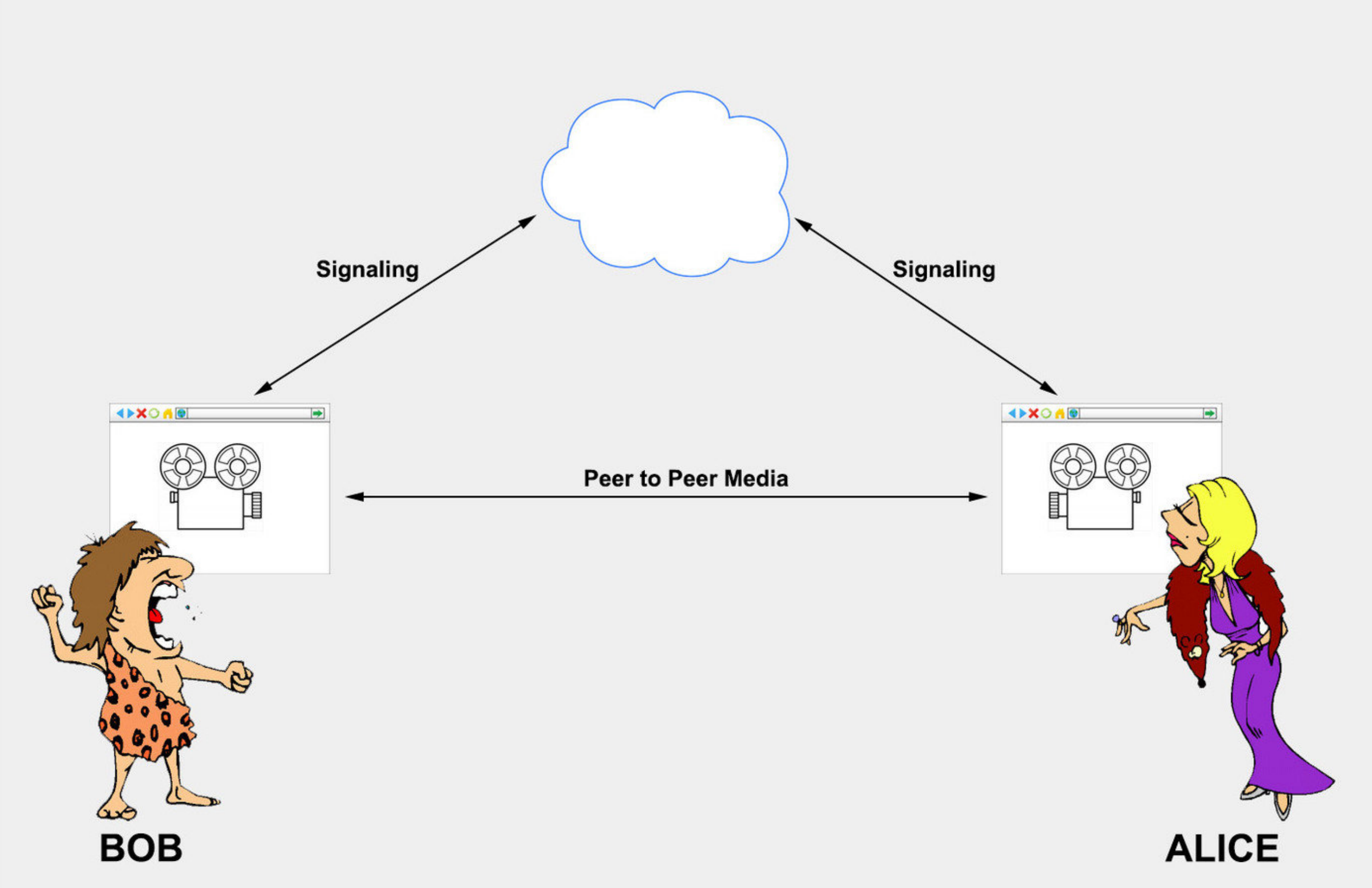


FIG. 76. Trådtelefon.

When Bob Met Alice



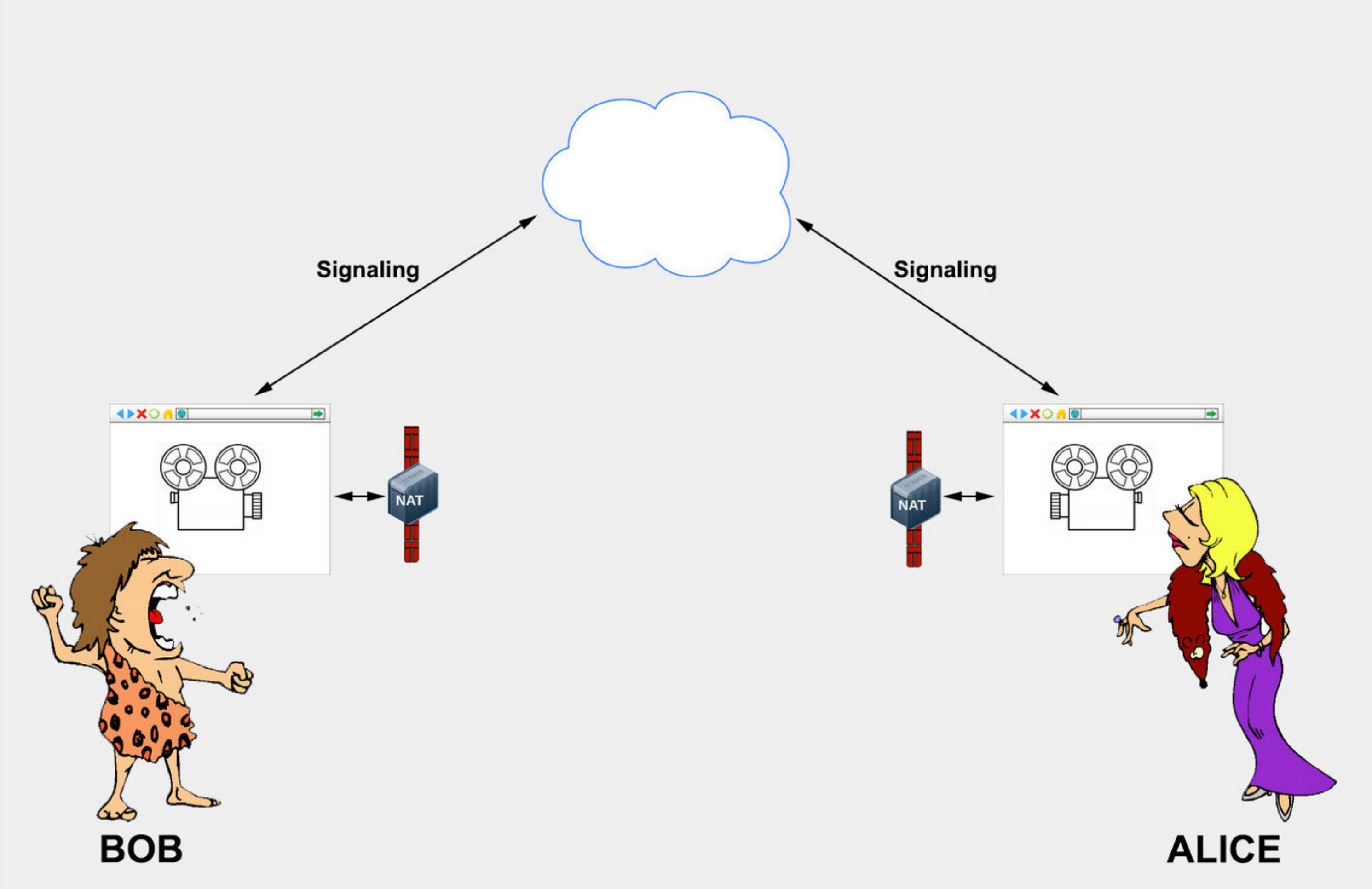
An Ideal World!



#boisecodecamp

@geekgonenomad

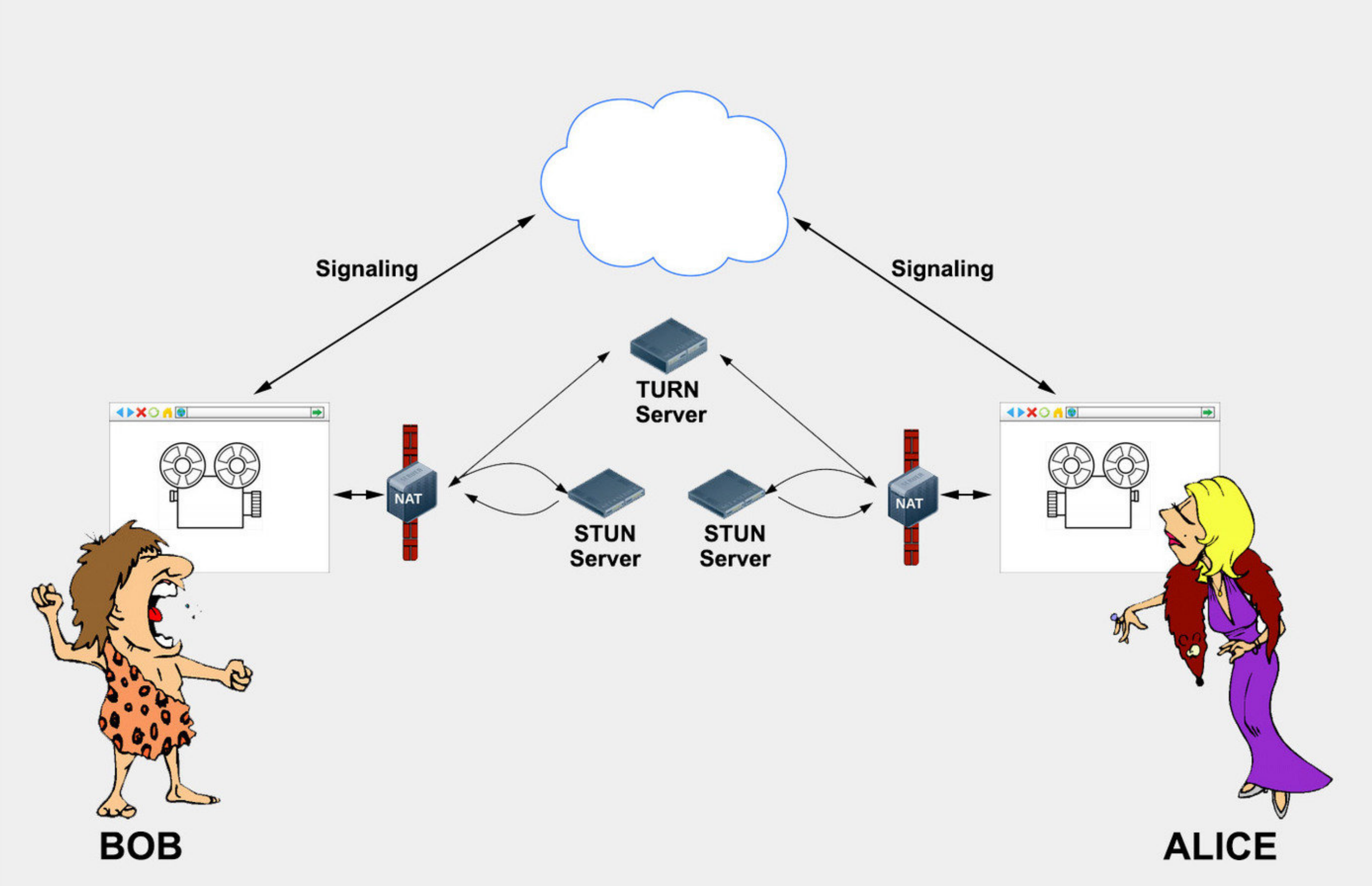
The Real World...



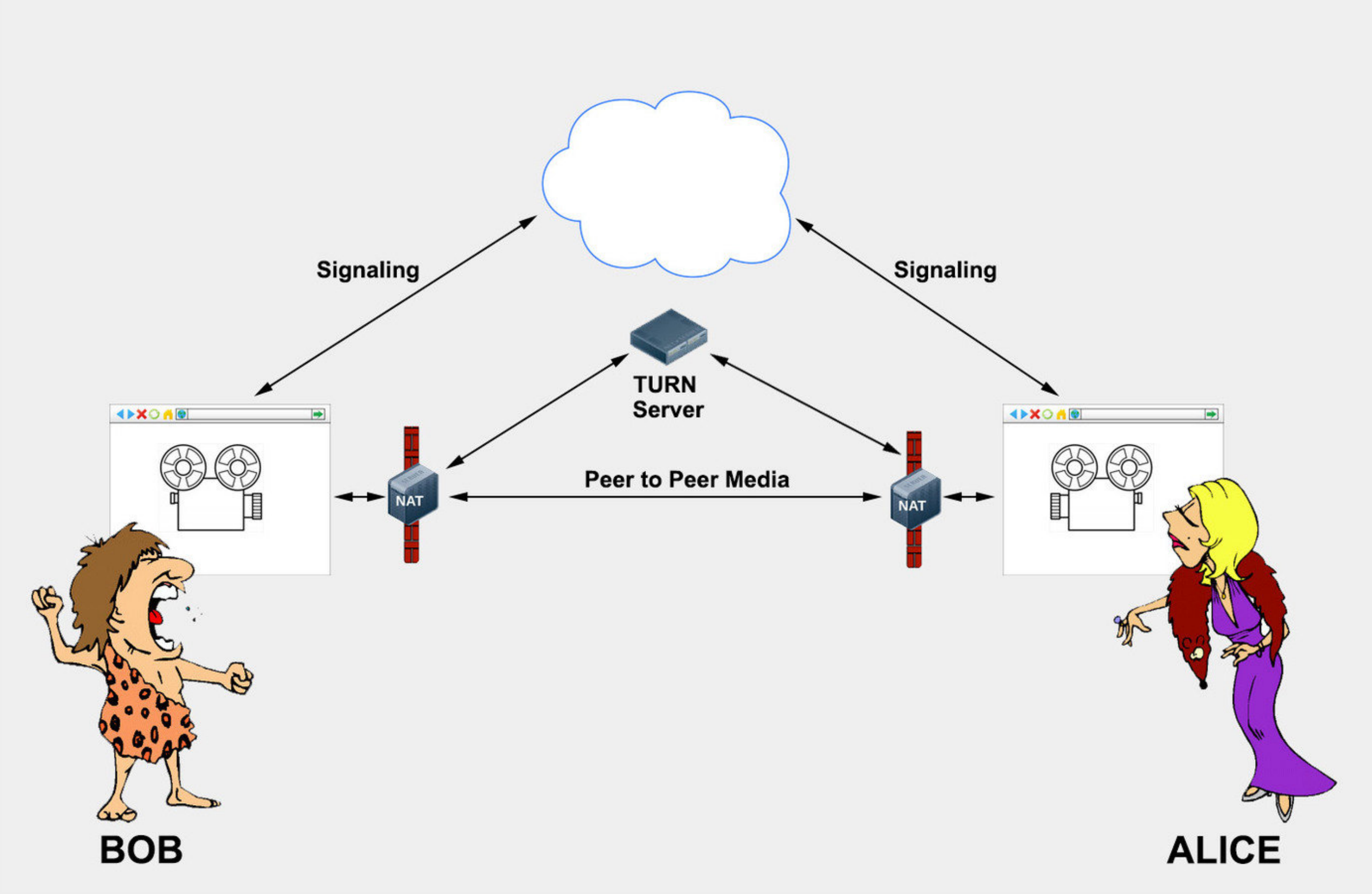
#boisecodecamp

@geekgonenomad

ICE, ICE, Baby


Signaling
Signaling
TURN Server
NAT
NAT
STUN Server
STUN Server
BOB
ALICE

The Complete Connection



#boisecodecamp

@geekgonenomad

Simple Right?

respoke

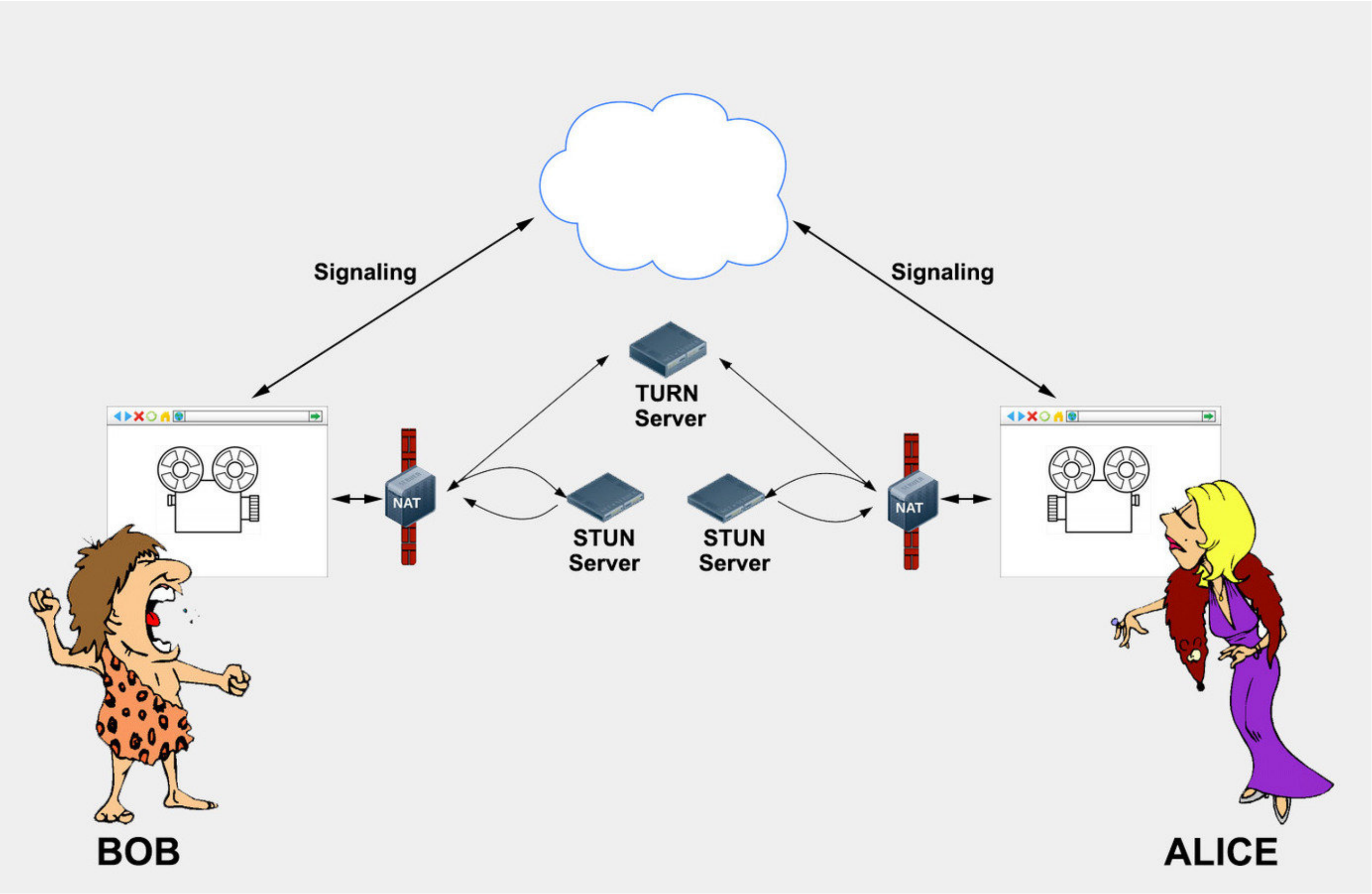
Respoke?

A set of API's that gives developers the ability to easily add real-time communication to their applications.

Like WebRTC?

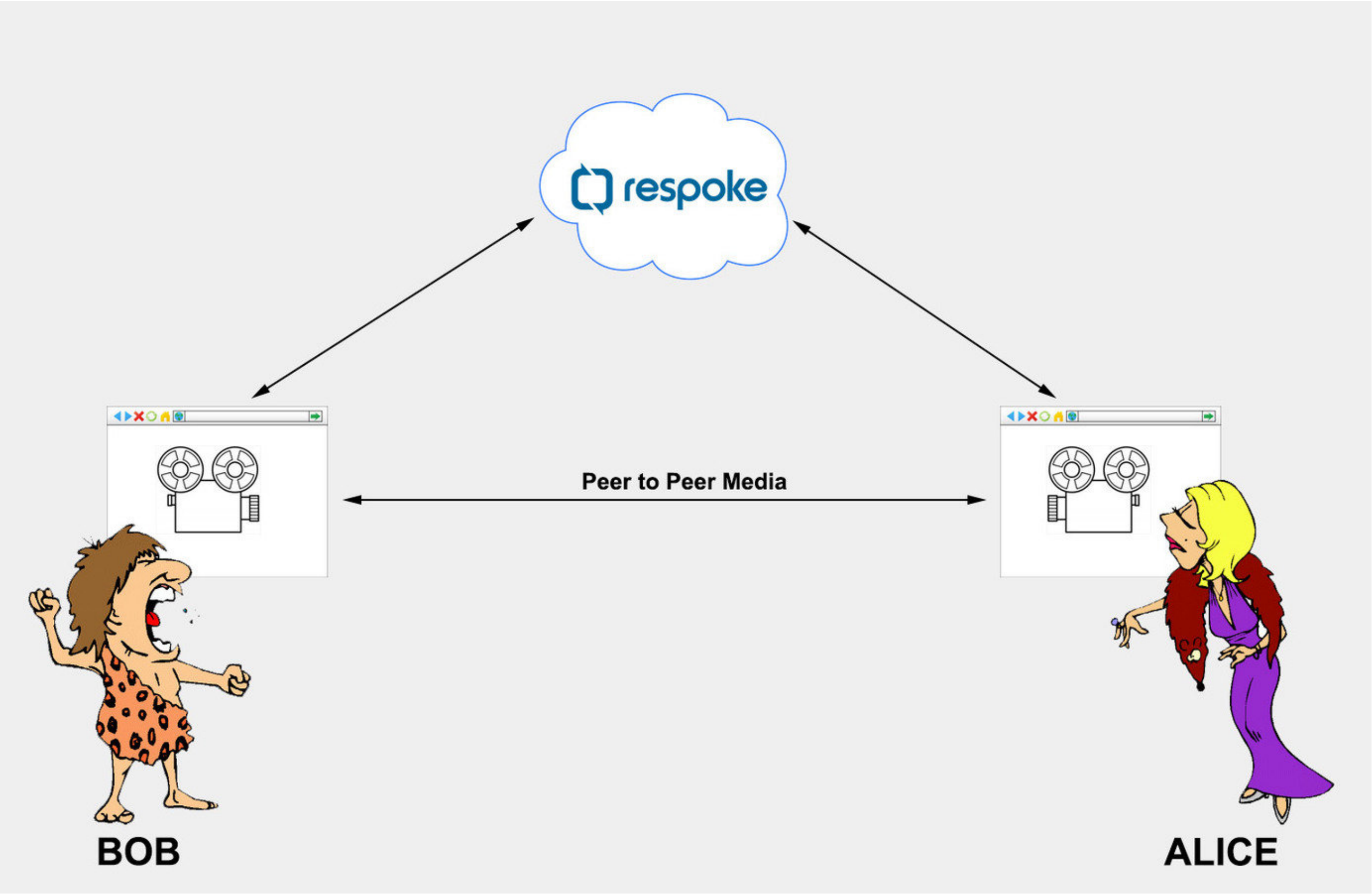
Sort of...

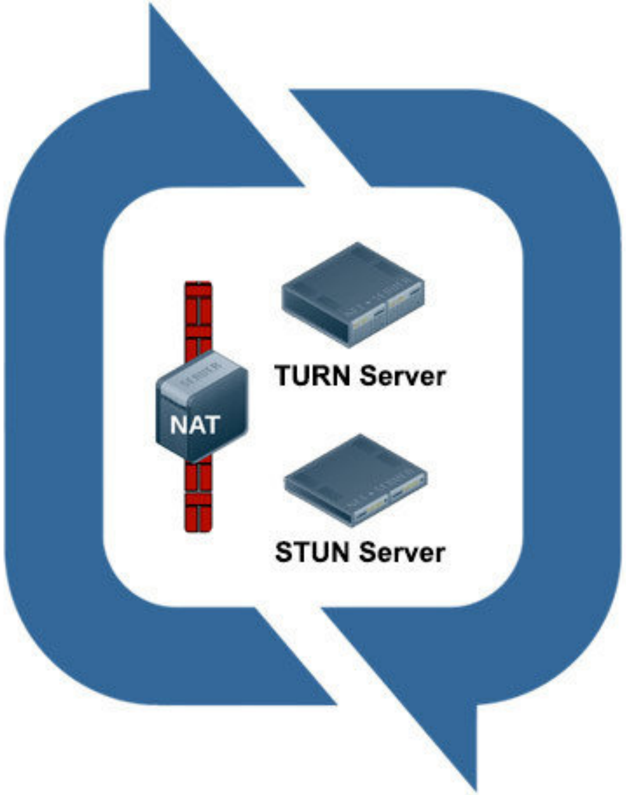
Remember This?



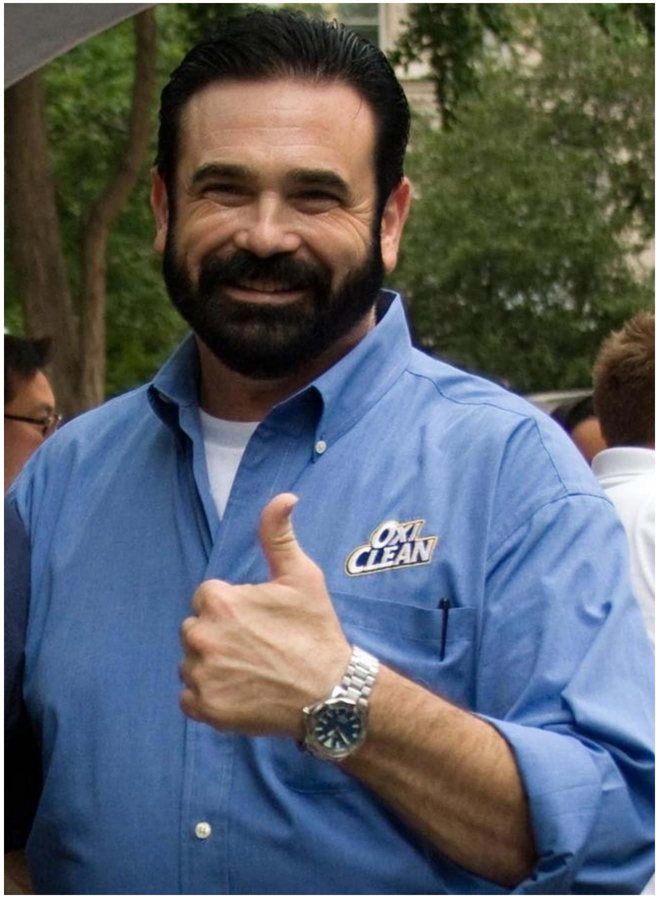
#boisecodecamp

@geekgonenomad


respoke
Peer to Peer Media
BOB
ALICE


NAT
TURN Server
STUN Server

But Wait! There's More...



Open Source

Cross-browser Support

Endpoint Discovery

Endpoint Messaging

Groups

Presence

Phone System Support

Mobile SDKs

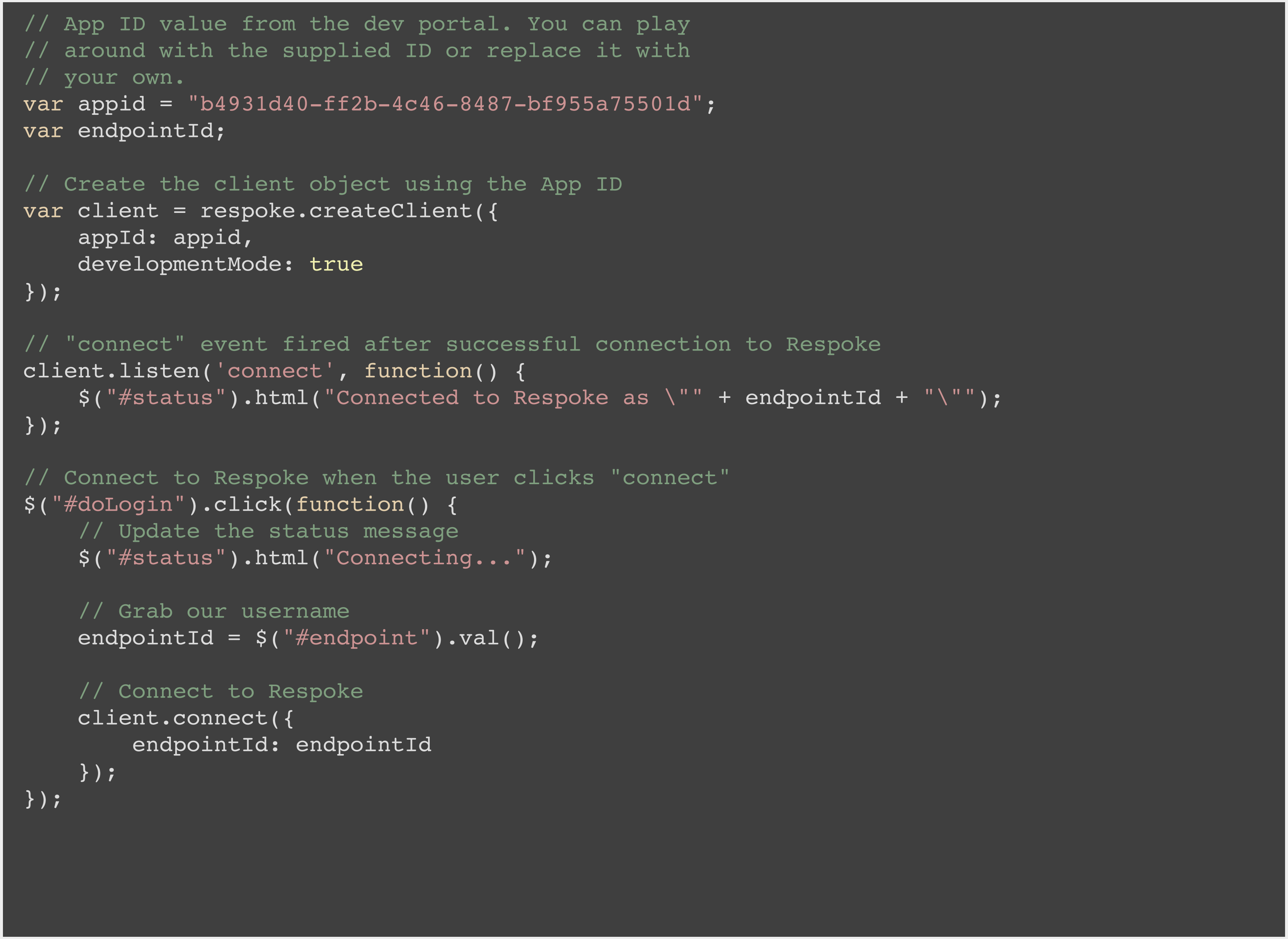
<CODE/>

Create a Respoke Account

www.respoke.io

Connecting to Respoke

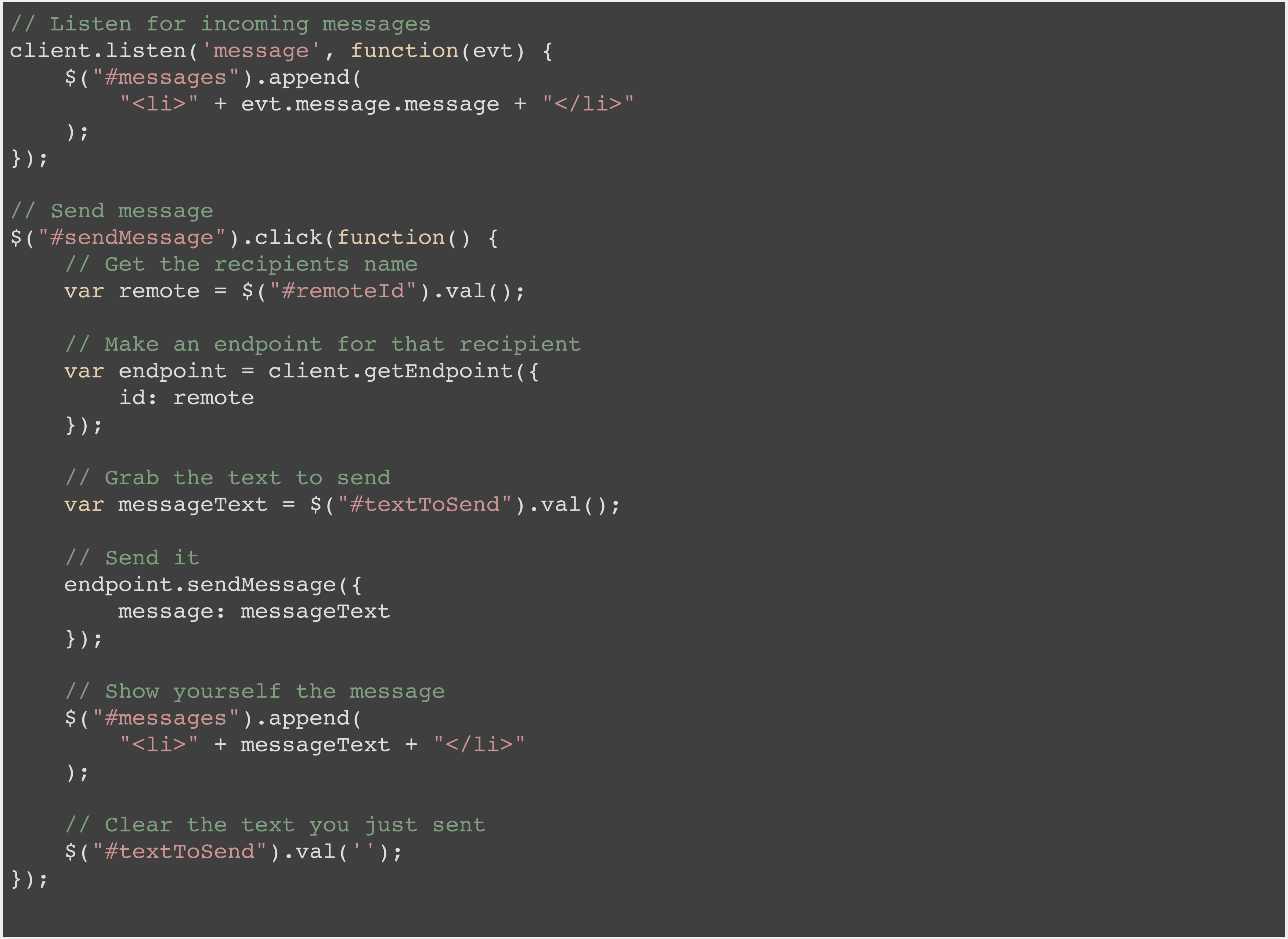
http://goo.gl/fajmx8



```
// App ID value from the dev portal. You can play
// around with the supplied ID or replace it with
// your own.
var appid = "b4931d40-ff2b-4c46-8487-bf955a75501d";
var endpointId;

// Create the client object using the App ID
var client = respoke.createClient({
    appId: appid,
    developmentMode: true
});

// "connect" event fired after successful connection to Respoke
client.listen('connect', function() {
    $("#status").html("Connected to Respoke as \"" + endpointId + "\"");
});

// Connect to Respoke when the user clicks "connect"
$("#doLogin").click(function() {
    // Update the status message
    $("#status").html("Connecting...");

    // Grab our username
    endpointId = $("#endpoint").val();

    // Connect to Respoke
    client.connect({
        endpointId: endpointId
    });
});
```

Messaging

http://goo.gl/vq8XCy



```
// Listen for incoming messages
client.listen('message', function(evt) {
    $("#messages").append(
        "<li>" + evt.message.message + "</li>"
    );
});

// Send message
$("#sendMessage").click(function() {
    // Get the recipients name
    var remote = $("#remoteId").val();

    // Make an endpoint for that recipient
    var endpoint = client.getEndpoint({
        id: remote
    });

    // Grab the text to send
    var messageText = $("#textToSend").val();

    // Send it
    endpoint.sendMessage({
        message: messageText
    });

    // Show yourself the message
    $("#messages").append(
        "<li>" + messageText + "</li>"
    );

    // Clear the text you just sent
    $("#textToSend").val('');
});
```

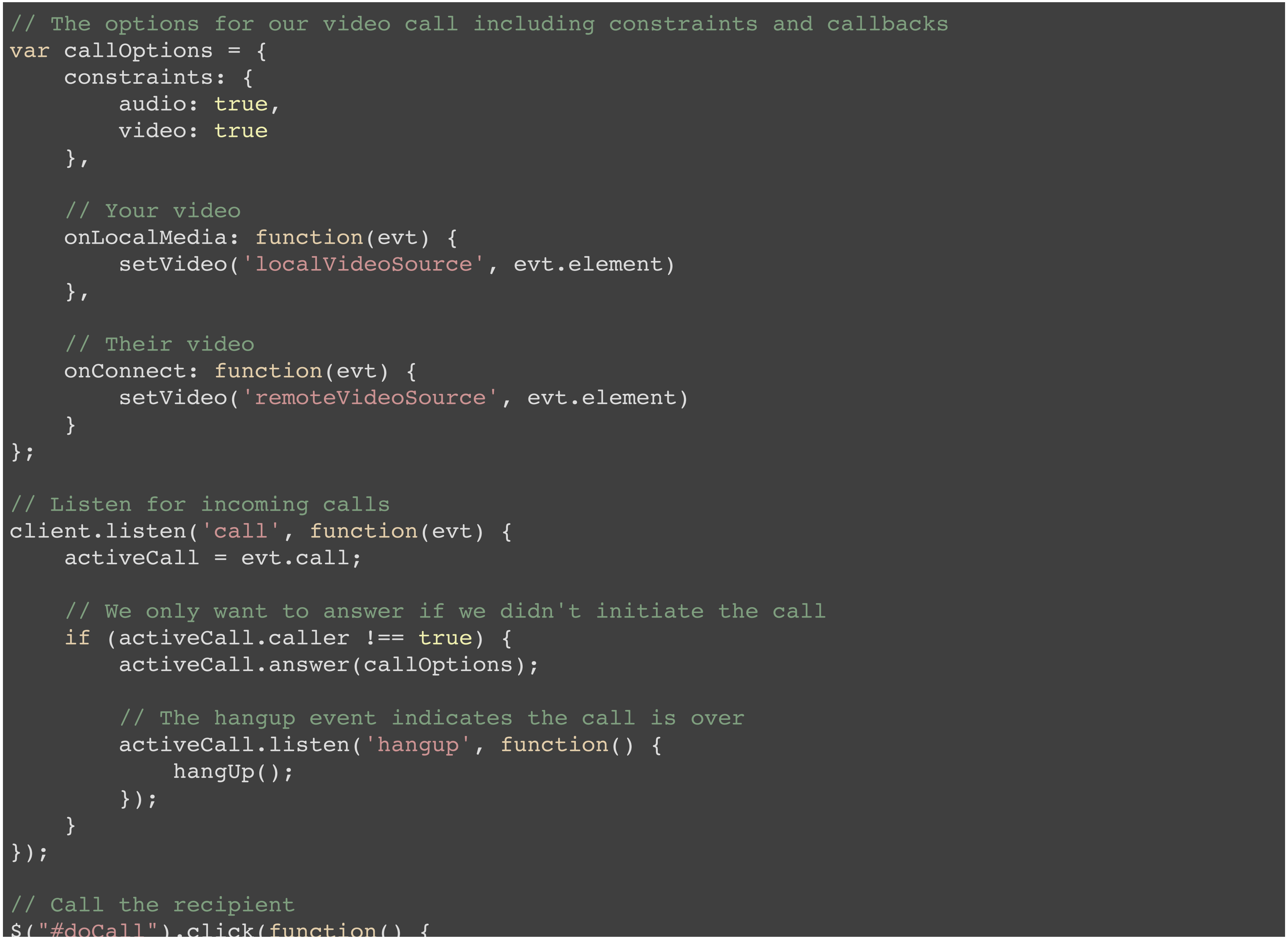
Group Messaging

http://goo.gl/DNxVuc

```
<!DOCTYPE html>

<head>
    <title>Respoke - Group Messaging Example</title>

    <!-- Respoke client library -->
    <script src="https://cdn.respoke.io/respoke.min.js"></script>

    <!-- jQuery, for this example -->
    <script src="http://code.jquery.com/jquery-1.11.0.min.js"></script>

    <!-- Some simple styles to make things perty -->
    <link rel="stylesheet" type="text/css" href="style.css">
    </style>
</head>

<body>
    <h3 id="status">Not Connected to Respoke</h3>
    <h3 id="groupStatus">Not Connected to Group</h3>
    <div id="login">
        <input id="endpoint" placeholder="Username" type="text" />
        <button id="doLogin">Connect</button>
    </div>

    <div id="messaging">
        <ul id="messages"></ul>
        <br />
        <textarea id="groupMsg" placeholder="Message to Send" rows="4"></textarea>
        <br/>
        <button id='sendMessage'>Message Group</button>
        <button id='leaveGroup'>Leave Group</button>
    </div>


    <script type="text/javascript">
```

Video Calling

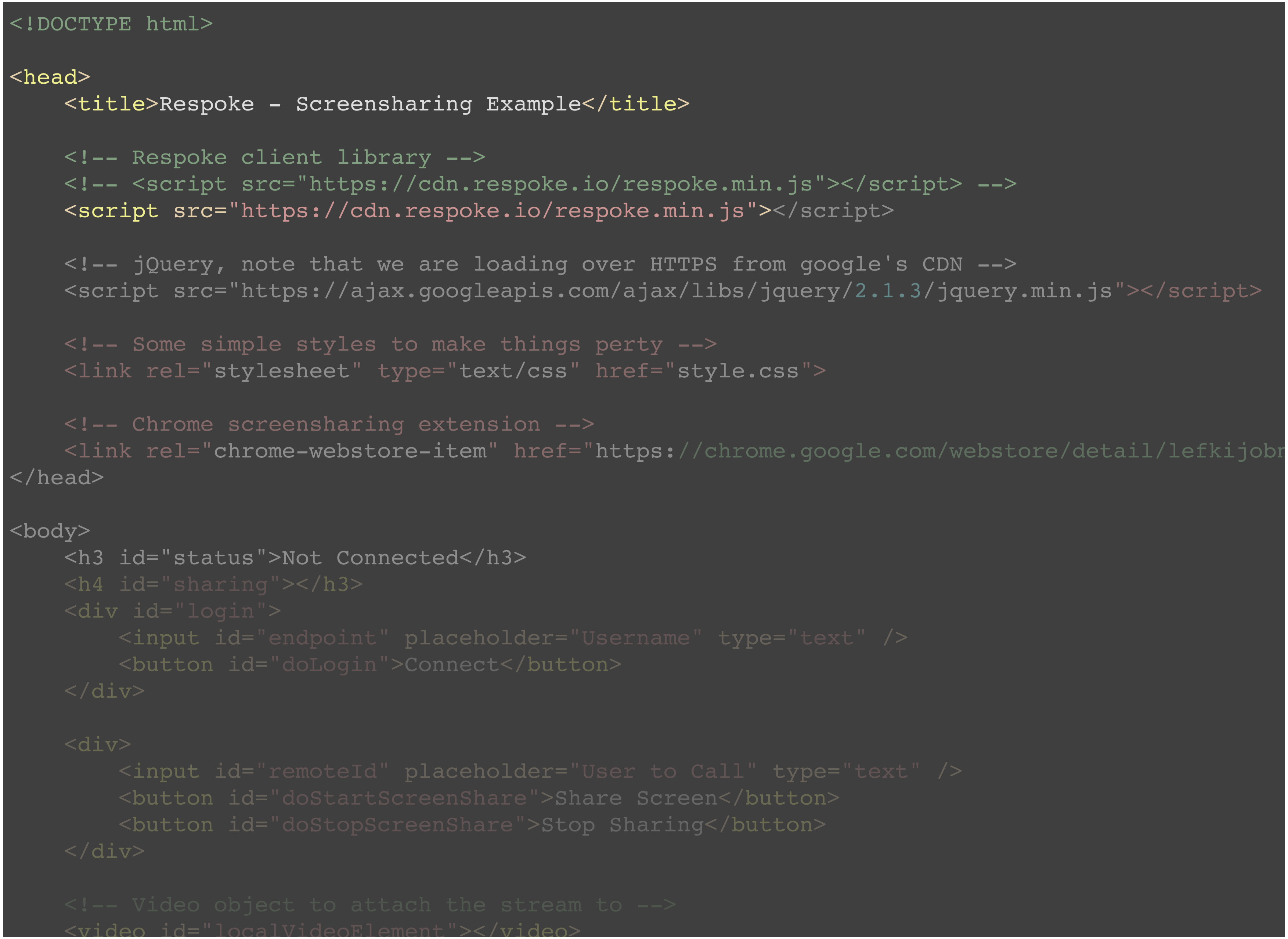
http://goo.gl/m4tZwX



```
// The options for our video call including constraints and callbacks
var callOptions = {
    constraints: {
        audio: true,
        video: true
    },

    // Your video
    onLocalMedia: function(evt) {
        setVideo('localVideoSource', evt.element)
    },

    // Their video
    onConnect: function(evt) {
        setVideo('remoteVideoSource', evt.element)
    }
};

// Listen for incoming calls
client.listen('call', function(evt) {
    activeCall = evt.call;

    // We only want to answer if we didn't initiate the call
    if (activeCall.caller !== true) {
        activeCall.answer(callOptions);

        // The hangup event indicates the call is over
        activeCall.listen('hangup', function() {
            hangUp();
        });
    }
});

// Call the recipient
$("#doCall").click(function() {
```

Screen Sharing

http://goo.gl/2pOJ3A



```
<!DOCTYPE html>

<head>
    <title>Respoke - Screensharing Example</title>

    <!-- Respoke client library -->
    <!-- <script src="https://cdn.respoke.io/respoke.min.js"></script> -->
    <script src="https://cdn.respoke.io/respoke.min.js"></script>

    <!-- jQuery, note that we are loading over HTTPS from google's CDN -->
    <script src="https://ajax.googleapis.com/ajax/libs/jquery/2.1.3/jquery.min.js"></script>

    <!-- Some simple styles to make things perty -->
    <link rel="stylesheet" type="text/css" href="style.css">

    <!-- Chrome screensharing extension -->
    <link rel="chrome-webstore-item" href="https://chrome.google.com/webstore/detail/lefkijobr
</head>

<body>
    <h3 id="status">Not Connected</h3>
    <h4 id="sharing"></h3>
    <div id="login">
        <input id="endpoint" placeholder="Username" type="text" />
        <button id="doLogin">Connect</button>
    </div>

    <div>
        <input id="remoteId" placeholder="User to Call" type="text" />
        <button id="doStartScreenShare">Share Screen</button>
        <button id="doStopScreenShare">Stop Sharing</button>
    </div>

    <!-- Video object to attach the stream to -->
    <video id="localVideoElement"></video>
```

Docs and Additional Examples

https://docs.respoke.io/

What's It All Mean?

Faster Development

Better User Experience

Happy Bosses

Happy ~~Bosses~~ Users!

#boisecodecamp

@geekgonenomad

Communication should be a *feature*, not a *product*…

Kyle Tyacke

✉ ktyacke@respoke.io

@geekgonenomad

www.respoke.io

Examples and Source

http://ktyacke.github.io/respoke-webrtc-preso/