Χρήση τεχνολογίας **Web Real Time Communication(WebRTC)** για επικοινωνία μέσω βίντεο κλήσης

Η δημιουργία εφαρμογής βίντεο κλήσης με τη βοήθεια της τεχνολογίας Web Real Time Communication(WebRTC) είναι αρκετά απλή με τη χρήση των διαθέσιμων modules που μας δίνονται απο το JavaScript API του WebRTC.

Παρακάτω θα γίνει αναφορά σε ορισμένα κομμάτια κώδικα της εφαρμογής, βλέποντας παράλληλα την λειτουργία της.

Η εφαρμογή είναι διαθέσιμη για Peer-to-Peer επικοινωνία στο https://tha07.github.io/.

Ανάπτυξη εφαρμογής

Για τις ανάγκες της εφαρμογής, έγινε χρήση της της συνάρτησης επικοινωνία χρήστη-με-χρήστη (Peer-to-Peer Communicaton).

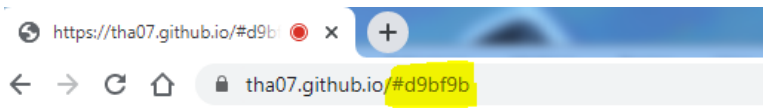
1. Όπως αναφέρθηκε, για την επικοινωνία δύο χρηστών είναι απαραίτητο να υπάρχει ένας signaling server, καθώς δεν περιλαμβάνεται στην επικοινωνία με WebRTC. Για το παράδειγμα αυτό θα γίνει χρήση ενός έτοιμου signaling server(Scaledrone).

Για τη λειτουργία του προγράμματος, είναι απαραίτητη η δημιουργία ξεχωριστού δωματίου(όσο αυτό είναι δυνατό) για κάθε βίντεοκλήση, με τη βοήθεια hascode.

```
1  // Δημιουργία hashcode για το link του νέου χρήστη
2  if (!location.hash) {
3    location.hash = Math.floor(Math.random() * 0xFFFFFF).toString(16);
4  }
5  const roomHash = location.hash.substring(1);
6
```

Δημιουργία Hash Code για τον μοναδικό σύνδεσμο

Ο σύνδεσμος που θα δημιουργηθεί μόλις μπούμε στο δωμάτιο, πρέπει να αποσταλεί με εξωτερικό τρόπο, στον χρήστη με τον οποίο επιθυμούμε να γίνει η επικοινωνία.



Σύνδεσμος στον Browser

2. Στη συνέχεια, γίνεται δημιουργία ενός αντικειμένου "configuration" στο οποίο αναφέρονται οι σύνδεσμοι των Session Traversal Utilities for NAT (STUN) και Traversal Using Relays around NAT (TURN) server που θα χρειαστούν.

```
const configuration={iceServers: [
    {
      url: 'turn:numb.viagenie.ca',
        credential: 'muazkh',
        username: 'webrtc@live.com'
    },
    {
      urls: "stun: stun1.l.google.com:19302"
    }
  ]};
```

TURN και STUN servers

Για το πρόγραμμα γίνεται χρήση δωρεάν διαθέσιμων STUN και TURN server, με τον STUN Server της Google.

3. Το πρόγραμμα αρχικά δημιουργεί το δωμάτιο στον signaling server του scaledrone, και επειτα διαβάζει τον αριθμό των χρήστών συνδεδεμένων στο server. Αν ο αριθμός των χρηστών είναι δύο, τότε ξεκινάει η WebRTC επικοινωνία.

```
let room;
let pc;


function onSuccess() {};
function onError(error) {
  console.error(error);
};

drone.on('open', error => {
  if (error) {
    return console.error(error);
  }
  room = drone.subscribe(roomName);
  room.on('open', error => {
    if (error) {
      onError(error);
    }
  });
  // We're connected to the room and received an array of 'members'
  // connected to the room (including us). Signaling server is ready.
  room.on('members', members => {
    console.log('MEMBERS', members);
    // If we are the second user to connect to the room we will be creating the offer
    const isOfferer = members.length === 2;
    startWebRTC(isOfferer);
  });
```

Βασικός κορμός προγράμματος

4. Η βασική επικοινωνία WebRTC γίνεται στη συνάρτηση startWebRTC που βρίσκεται στο τέλος του κύριου προγράμματος.

4.1. Δημιουργία νέας RTCPeerConnection[WEBRTC] (με τα στοιχεία των STUN και TURN servers) στη μεταβλητή pc.

```
69  function startWebRTC(isOfferer) {
70    pc = new RTCPeerConnection(configuration);
71
```

4.2. Χρήση του event handler onicecandidate[ONICE], έτσι ώστε να γίνεται γνωστό πότε ένας Interactive Connectivity Establishment (ICE) χρήστης είναι έτοιμος να στείλει το μήνυμα στον άλλο χρήστη μέσω του signaling server.

```
74  pc.onicecandidate = event => {
75    if (event.candidate) {
76      sendMessage({'candidate': event.candidate});
77    }
78  };
```

4.3. Χρήση του event listener onnegotiationneeded[ONNEG], έτσι ώστε ο χρήστης να μπορεί να κάνει αλλαγή στο session.

```
81  if (isOfferer) {
82    pc.onnegotiationneeded = () => {
83      pc.createOffer().then(localDescCreated).catch(onError);
84    }
85  }
86
```

4.4. Όταν φτάσει κάποιο stream δεδομένων, διοχετεύεται στο HTML στοιχείο remoteVideo.

```
88  pc.onaddstream = event => {
89    remoteVideo.srcObject = event.stream;
90  };
91
```

4.5. Εμφάνιση τοπικής ροής βίντεο στο HTML στοιχείο localVideo.

```
92  navigator.mediaDevices.getUserMedia({
93    audio: true,
94    video: true,
95  }).then(stream => {
96    // Display your local video in #localVideo element
97    localVideo.srcObject = stream;
98    // Add your stream to be sent to the conneting peer
99    pc.addStream(stream);
100 }, onError);
101
```

4.6. Επικοινωνία με τον signaling server, και παίρνουμε τα διαθέσιμα δεδομένα.

```
103   room.on('data', (message, client) => {
104
105     if (client.id === drone.clientId) {
106       return;
107     }
108
```

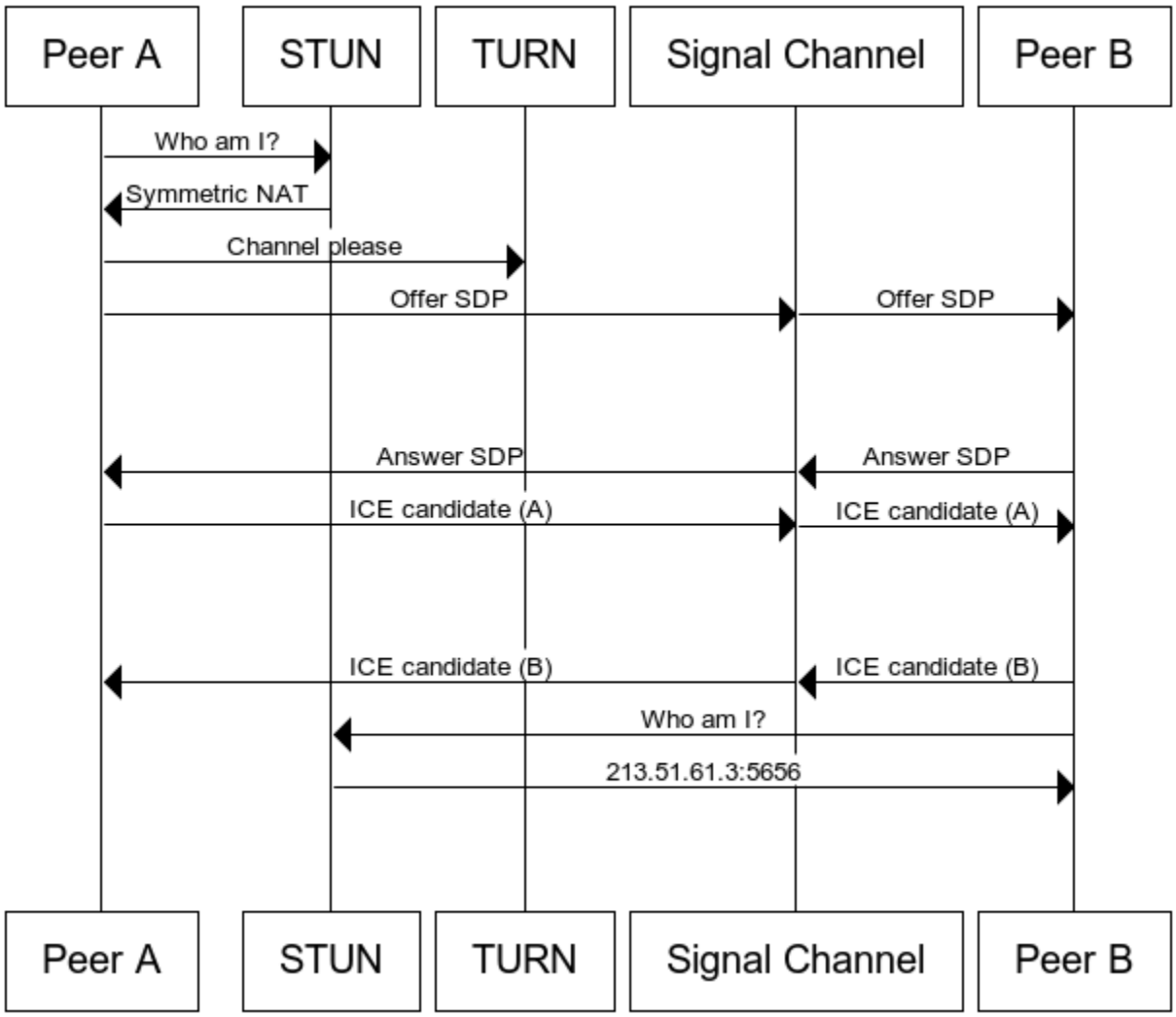
4.7. Σε περίπτωση που έρθει προσφορά απο ήδη διαθέσιμο χρήστη, λαμβάνεται το Session Description Protocol(SDP), και αποστέλεται αντίστοιχα το δικό μας. Διαφορετικά, προσθέτουμε τον νέο ICE υποψήφιο σώζοντας τα δικά του στοιχεία SDP.

```
109     if (message.sdp) {
110       pc.setRemoteDescription(new RTCSessionDescription(message.sdp), () => {
111         if (pc.remoteDescription.type === 'offer') {
112           pc.createAnswer().then(localDescCreated).catch(onError);
113         }
114       }, onError);
115     } else if (message.candidate) {
116       pc.addIceCandidate(
117         new RTCIceCandidate(message.candidate), onSuccess, onError
118       );
119     }
120   });
121 }
```

5. Συνοπτικά ο HTML κώδικας:

```
1  <!DOCTYPE html>
2  <html>
3  <head>
4    <script src='https://cdn.scaledrone.com/scaledrone.min.js'></script>
5    <meta charset="utf-8">
6    <meta name="viewport" content="width=device-width">
7    <style>
51 </head>
52 <body>
53   <video id="localVideo" autoplay muted></video>
54   <video id="remoteVideo" autoplay></video>
55   <div class="copy">Στείλε το URL ένα φίλο για να αρχίσετε τη συνομιλία!</div>
56    <div class="copy2">Δώσε πρόσβαση κάμερας και ήχου στον επιλεγμένο Browser</div>
57    <div class="titlos">Σύνδεση τύπου Peer-to-Peer με χρήση WebRTC</div>
58   <script src="script.js"></script>
59 </body>
60 </html>
```

Η όλη συναλλαγή σε διάγραμμα



Συναλλαγή WebRTC [Diag1]

Αναφορές

[ONICE]: https://developer.mozilla.org/en-US/docs/Web/API/RTCPeerConnection/onicecandidate

[ONNEG]: https://developer.mozilla.org/en-US/docs/Web/API/RTCPeerConnection/onnegotiationneeded

[Diag1]: https://developer.mozilla.org/en-US/docs/Web/API/WebRTC_API/Connectivity

[WEBRTC]: https://webrtc.org/getting-started/peer-connections