Partner for your success

w5Capi

USER GUIDE

v1.1

Approval

Completion of this approval section signifies the approvers have read, understood, and agreed with the contents of this document.

Approver Name	Role	Date(dd-mmm-yyyy)	Comment
Satya Panigrahi	R&D Manager	14/Nov/2014	Initial Version for W5CAPI V1.1

Preface

About this document

This document explains about W5CAPI API. It describes the APIs that developers can use to integrate and implement real-time communications services such as Voice, Video and Chat in their web applications. It also explains how to run the demo and the source code implemented to achieve real-time communication.

Audience

Software Developers of Product and Service companies

Freelancers

All vertical companies who want to enable real-time communication in their applications.

Document conventions

Information elements	**Style convention used in the document**
UI labels and menu options	**Bold**
Code fragments	*`Courier New, Italics, Blue color`*

Assumptions

The users of this product have knowhow on how to develop applications that run in the browser. If the same application to be run on desktop and other OS, you are recommended to contact W5RTC Sales team.

Acronyms

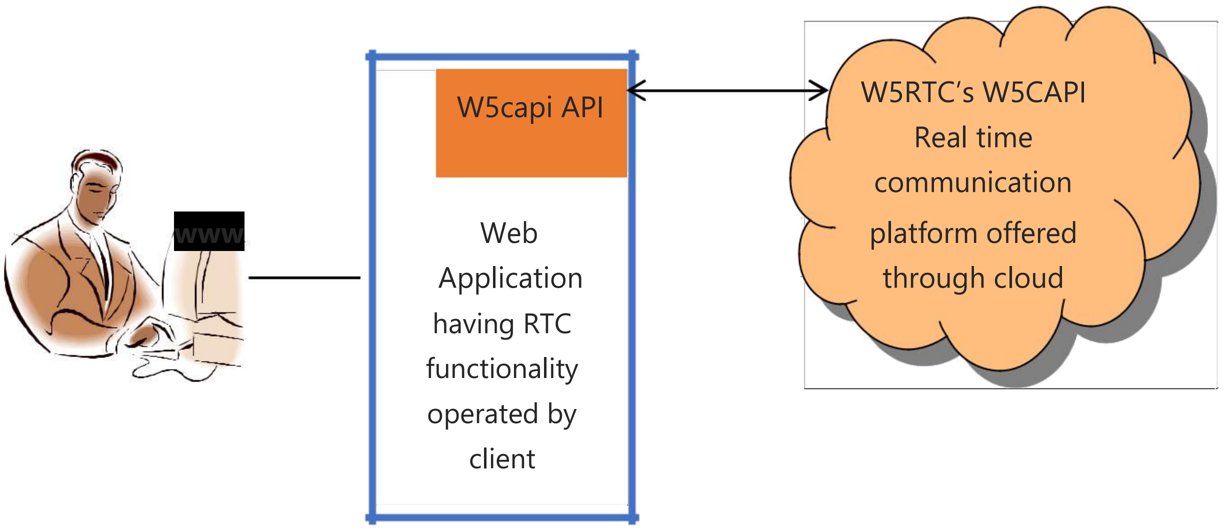
Acronym	Description
W5CAPI	W5RTC platform that comes with Cloud based Application programmable Interface
API	Application programmable interface

Contents

1. Executive Summary

W5CAPI is a real time communication PaaS (Platform as a Service) product that has all the hardware infrastructure and servers in the cloud that enables real-time communication between 2 or more users. The platform is designed with high-level of security, scalability and quality. In order to interface with the platform, W5CAPI provides API to the software developers that can be used in their application to enable real-time communication in their application. When integrated in their application, they will able to perform real time communication, such as instant messaging, voice call, video call, file sharing, and screen sharing, right through their applications.

1.1 Usage work flow


www
W5capi API
Web Application having RTC functionality operated by client
W5RTC's W5CAPI Real time communication platform offered through cloud

1. Application Software developers use W5CAPI API and build applications.
2. They include License Key in the application to access the W5RTC's W5CAPI platform
3. W5CAPI platform enables real-time communication between the users accessing via the applications built by Application software developer.

1.2 Benefits of using W5CAPI

There are many benefits using W5CAPI. As an example W5RTC had built W5CALL application (https://w5rtc.in/w5call.php) using W5CAPI that can be used to integrate in any website to enable real-time communication between website visitors to website agents/owner. This W5CALL application can help companies to Increases sales by improving customer collaboration and engagement by enabling real-time voice, video and data communication directly through websites.

W5CALL is just an application. Similarly one can build any application for any verticals such as education, health etc., using W5CAPI platform that supports rich real-time communication features such as chat, voice and video call. It can bring in many benefits but not limited to:

Reduces travel costs of your customers that could occur for face to face communication requirements such as interviews, trainings, conferences.

Brings staff together for training and conferences through the applications. That allows multiple remote parties to meet at one place online.

Reduces hardware software cost by eliminating the need for audio-video (AV) hardware and software.

Creates scope for desktop, web and mobile application to innovate themselves or scale-up to capitalize on real time communications services and more.

1.3 Technical details

W5CAPI is built using in-house proprietary signaling technology for connecting the calls between 2 parties and uses WebRTC technology for media communication such as Voice and Video. W5CAPI provides simple, intuitive, easy-to-integrate, and easy-to-use API for building high quality and secure applications based on real time communications services.

The W5CAPI framework consists of a client or browser side JavaScript library that is downloaded in to the browser dynamically when the application uses the W5CAPI API and sends the valid license to the W5CAPI CDN servers. This Client side JavaScript library has the state machine that is executed by the browser. When executed it does many functions such as:

Accesses local camera and microphone

Establishes communication with NAT servers to resolve the Public IP address to reach the device having Private IP

Establishes connection to signaling server that connects multiple users

Connects the media (Voice, Video and Data) between 2 users

Supports various in-call actions such as mute, un-mute, enable Video etc. Disconnects the calls

Connects media streams to video tags

All these functions can be accessed directly from the browser and does not require an add-in browser plug-in.

In addition to all, W5CAPI also provides cross browser support. The W5CAPI Client API hides the programming differences between the browsers such as Chrome, Firefox and Opera. All this helps the application developers to concentrate on their applications logic and the rest of complex bank-end functionality is hidden and taken care of by W5RTCs' W5CAPI platform. .

As a PaaS platform, W5CAPI provides all the functionalities explained above in their cloud servers that can be consumed by the customer as a service with nominal monthly fees.

1.4 Terminologies

Callback: A JavaScript function supplied to a library. The library calls it when a particular event occurs.

Media Stream: An object encapsulating video or audio tracks.

Peer Connection: A connection between two different browsers that enables peer to peer communication.

Server: The Node.js server plus some JavaScript code provides the server side of the W5CAPI framework.

2. Browser Support

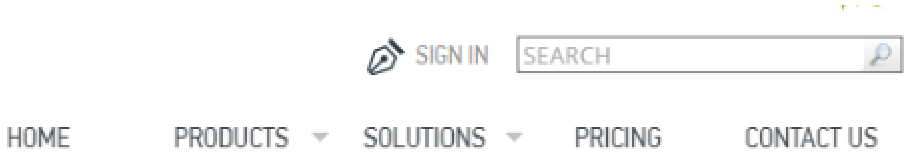
W5CAPI platform supports features in the browsers as shown in table below. We are actively working on to support many additional features in the roadmap that not only support real-time communication in all browsers but also to support in all operating systems such as MAC, IOS and Android as a standalone application.

	Chrome	Firefox	Internet Explorer	Opera	Safari
Audio	✓	✓	✓	✓	✓
Video	✓	✓	✓	✓	✓
Chat	✓	✓	✓	✓	✓
Screen Sharing	✓	✗	✗	✗	✗
File Sharing	✓	✓	✓	✓	✓

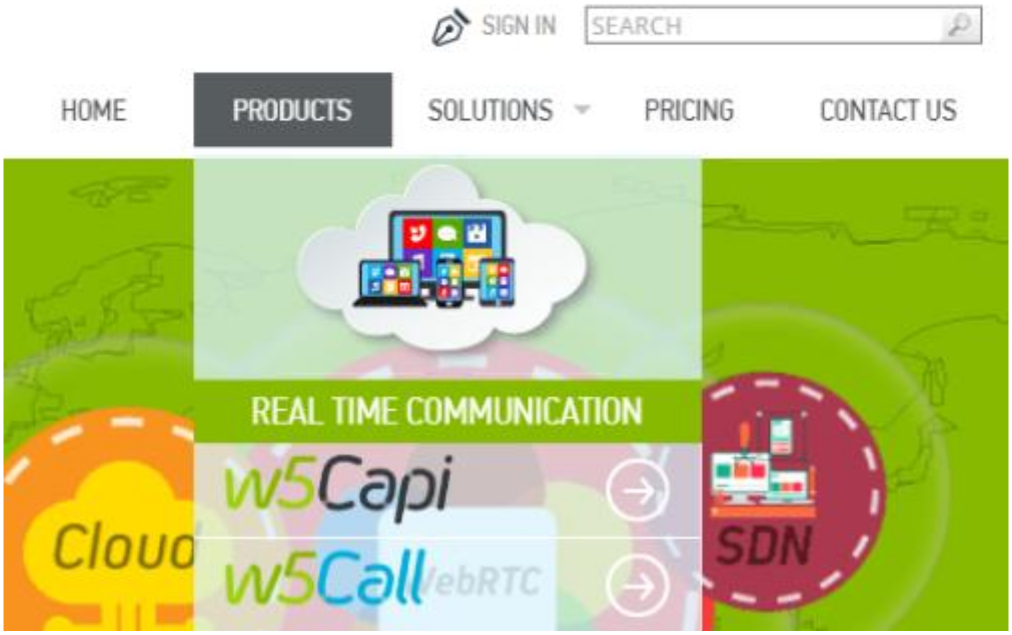
3. Getting the license key

If you have access to this document implies that you have already bought this product from W5RTC. If the new users require getting the W5CAPI product and license below are the steps that must be followed.

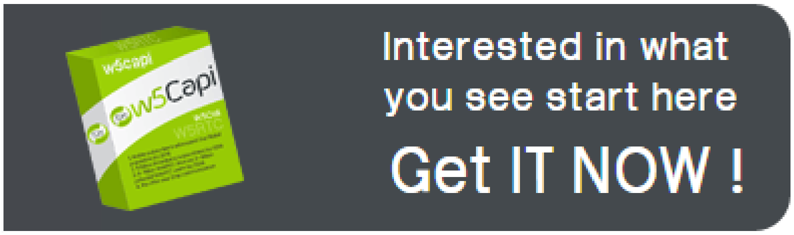
1. Go to w5rtc.com or w5rtc.in
2. Signup for the W5RTC account clicking SIGN IN button



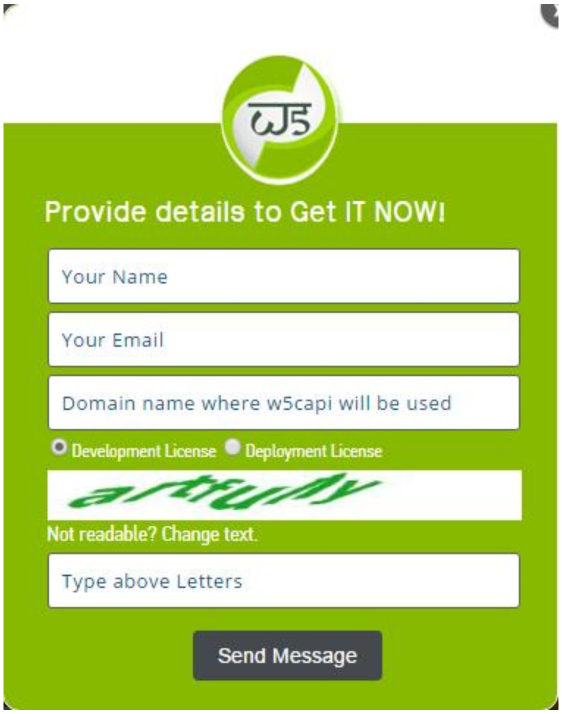
3. On the menu bar, click **Products** → **W5CAPI**.



4. The W5CAPI product page appears. On the right side of the page, click **Get it now!**



5. The “Provide details to get it now” form appears to fill-in you details.



Provide details to Get IT NOW!

Your Name

Your Email

Domain name where w5capi will be used

● Development License ○ Deployment License

Not readable? Change text.

Type above Letters

6. In **Your Name**, type your name.
7. In **Your Email**, type your email address.
8. In **Domain name where W5CAPI will be used**, type the domain name.
9. Select a license option:

   If you want to use the W5CAPI APIs for development purpose, select the **Development License**. This is the trail license provided to the customer to develop their application for maximum of 1 month

   After development, when you are ready to deploy or go live, you must get the **Deployment License**. This is the permanent license given for one year and renewed each year.

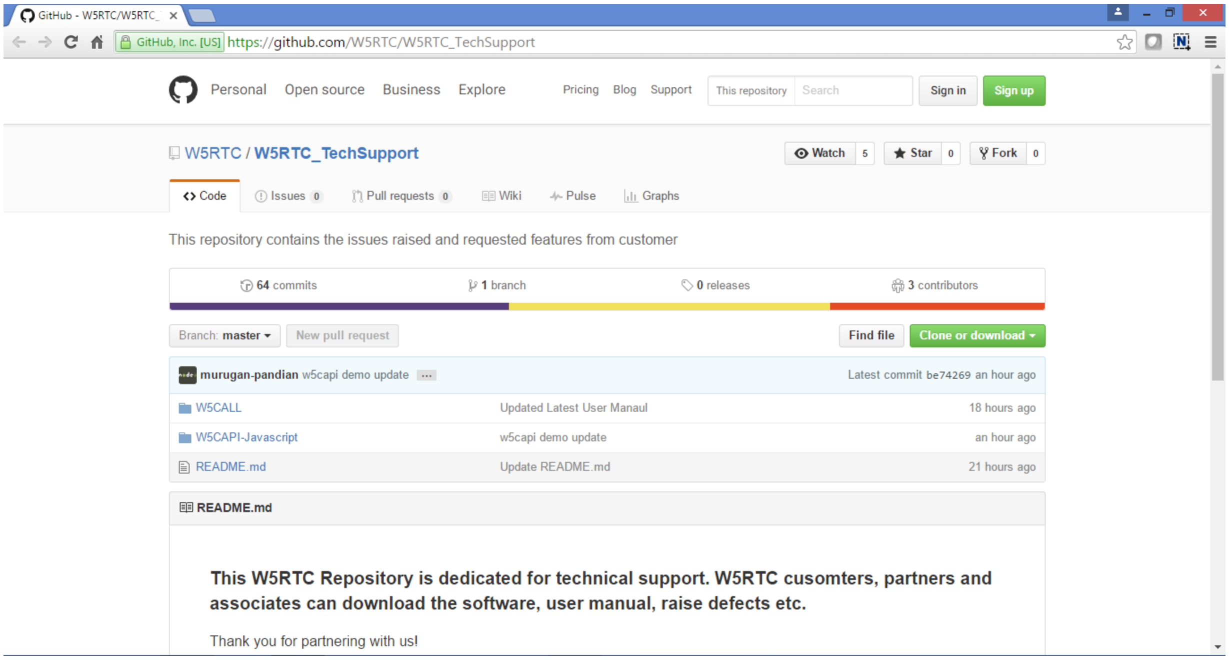
10. To confirm that that you are filling up the form manually by yourself, type the alphabets displayed in the image. If you can't read the text in the image, click **Change Text** to change the text.
11. Click **Send Message**.

W5RTC will contact you and provide you complete package along with the W5CAPI license. Please note the W5CAPI license Key is private, don't share it with anyone.

4. W5CAPI package

Open follow github repositories https://github.com/W5RTC/W5RTC_TechSupport

.



This repository contains the issues raised and requested features from customer

murugan-pandian w5capi demo update … Latest commit be74269 an hour ago

W5CALL	Updated Latest User Manaul	18 hours ago
W5CAPI-Javascript	w5capi demo update	an hour ago
README.md	Update README.md	21 hours ago

README.md

This W5RTC Repository is dedicated for technical support. W5RTC cusomters, partners and associates can download the software, user manual, raise defects etc.

Thank you for partnering with us!

The repositories contain the following files in it::

User manual

W5CAPI-Demo – The demo source code for easy learning and integration

license.txt – The W5CAPI License required accessing W5CAPI Cloud Platform and extesionID

W5CAPI-Extension-The extension source code for easy learning and integration

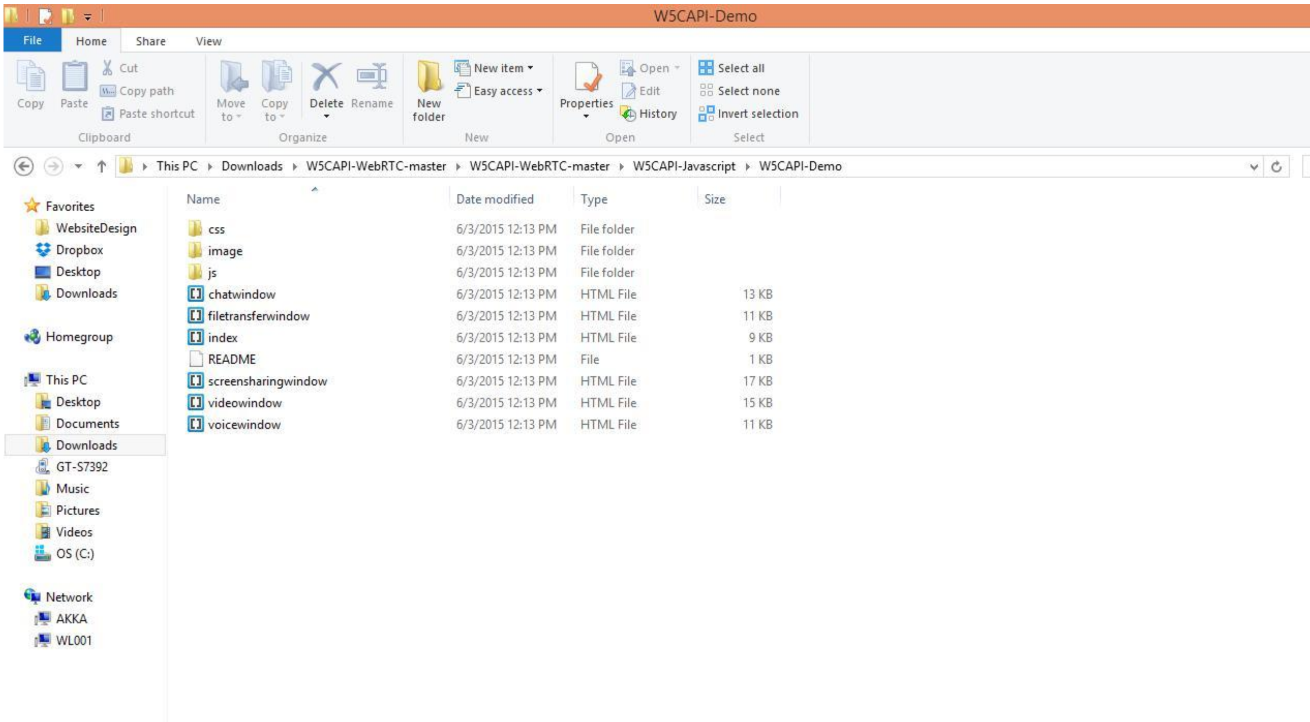
5. Running the demo application

This section explains how to setup and execute the demo. This will allow the user to understand how W5CAPI can be used to implement various features. By just cut and paste the code from demo source code provided, once can easily build the real-time communication in their own application.

5.1 Setting up the demo

To set up the demo:

1. Download W5CAPI- W5RTC_TechSupport repositories and open W5CAPI-Demo directory.



Name	Date modified	Type	Size
css	6/3/2015 12:13 PM	File folder	
image	6/3/2015 12:13 PM	File folder	
js	6/3/2015 12:13 PM	File folder	
chatwindow	6/3/2015 12:13 PM	HTML File	13 KB
filetransferwindow	6/3/2015 12:13 PM	HTML File	11 KB
index	6/3/2015 12:13 PM	HTML File	9 KB
README	6/3/2015 12:13 PM	File	1 KB
screensharingwindow	6/3/2015 12:13 PM	HTML File	17 KB
videowindow	6/3/2015 12:13 PM	HTML File	15 KB
voicewindow	6/3/2015 12:13 PM	HTML File	11 KB

The W5CAPI-Demo directory contains the following subdirectories:

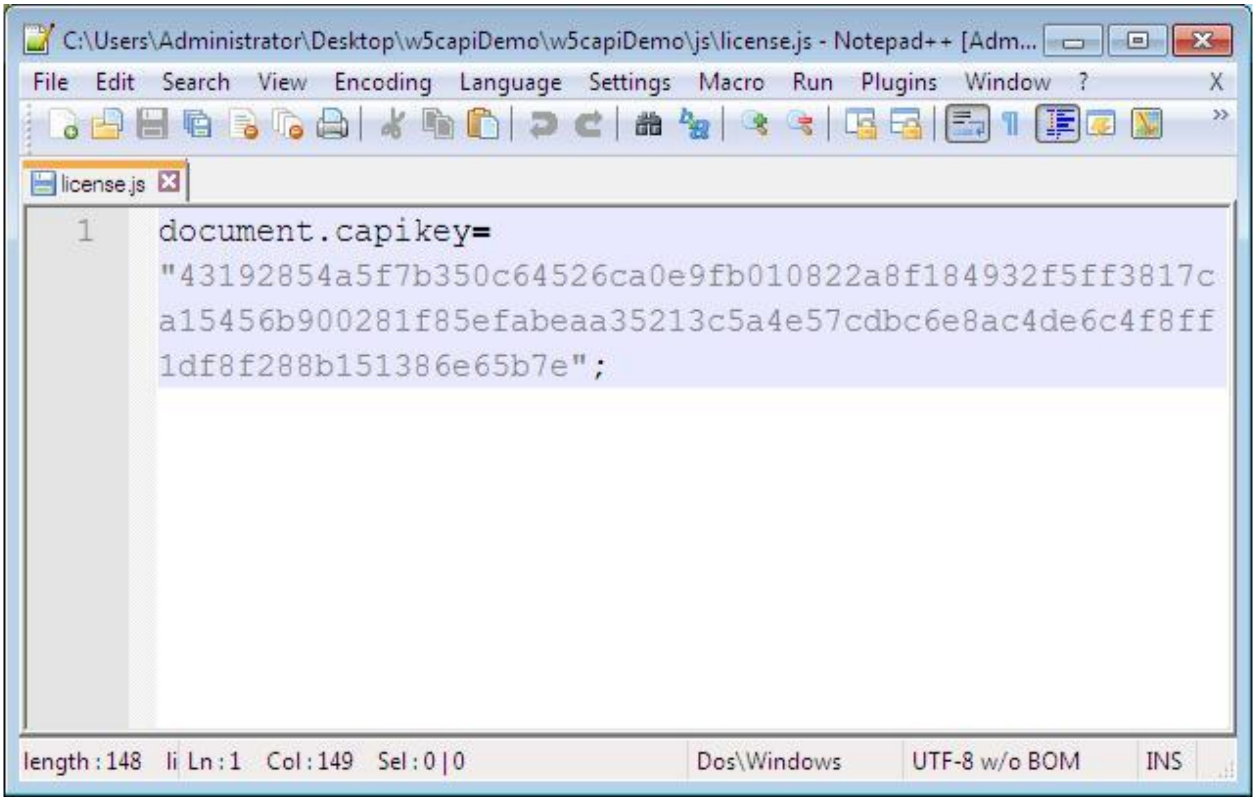
Image: Contains the images required for the demo

Css: Contains the style sheet required for the demo

Js: Contains the java scripts for license activation, notification, and RTC application logic.

2. Open the license.js file from the following path:

   /W5CAPI-Demo/js/license.js


C:\Users\Administrator\Desktop\w5capiDemo\w5capiDemo\js\license.js - Notepad++ [Adm...
File Edit Search View Encoding Language Settings Macro Run Plugins Window ? X
license.js
length : 148 li Ln : 1 Col : 149 Sel : 0 | 0 Dos\Windows UTF-8 w/o BOM INS

```
document.capikey=
"43192854a5f7b350c64526ca0e9fb010822a8f184932f5ff3817c
a15456b900281f85efabeaa35213c5a4e57cdbc6e8ac4de6c4f8ff
1df8f288b151386e65b7e";
```

3. Paste your license key in the file, within the quotation marks “”. The license is provided to you in “License.txt” file by W5RTC along with the package sent to you.

4. Copy the W5CAPI-Demo directory to your hosting space.

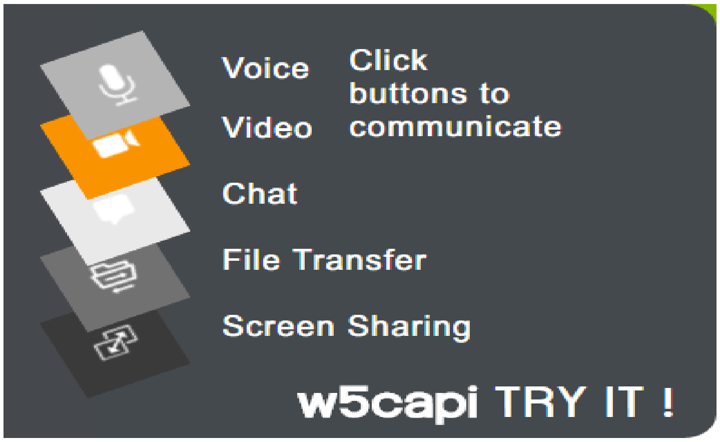
   Now you have successfully set up the W5CAPI-Demo application. You can run the RTC calls, as explained in the subsections further.

5.2 Running audio call

To run the audio call:

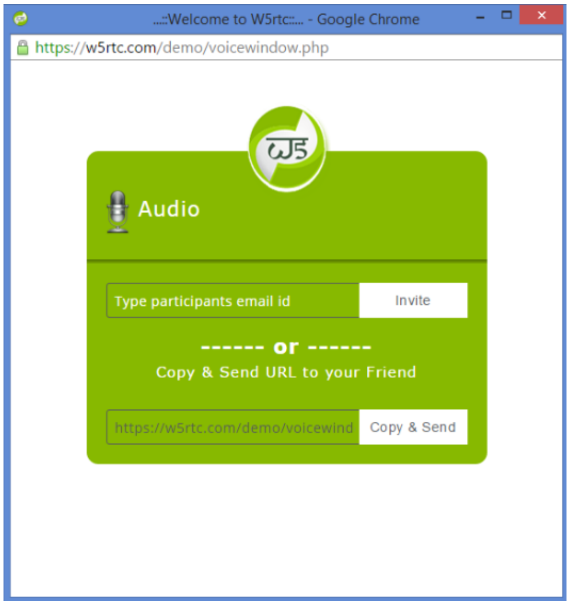
1. Open a browser and open the index file of the hosted directory. *http://<domainName>/W5CAPI-Demo/index.html*

The RTC Demo pane appears with the following list of features: Audio, Video, Chat, File Transfer, and Screen Sharing.

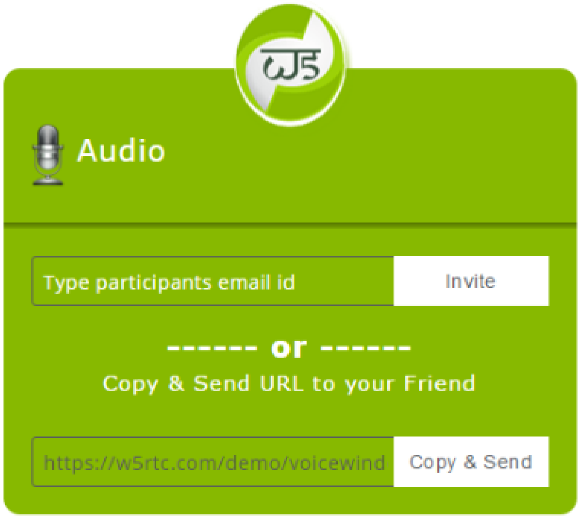


2. Click **Audio**.

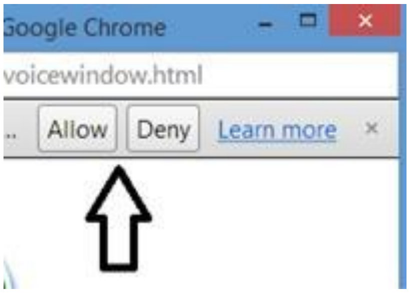
   The w5 Audio window appears.


...::Welcome to W5rtc::... - Google Chrome
https://w5rtc.com/demo/voicewindow.php
Audio
Type participants email id
Invite
------ or ------
Copy & Send URL to your Friend
https://w5rtc.com/demo/voicewind
Copy & Send

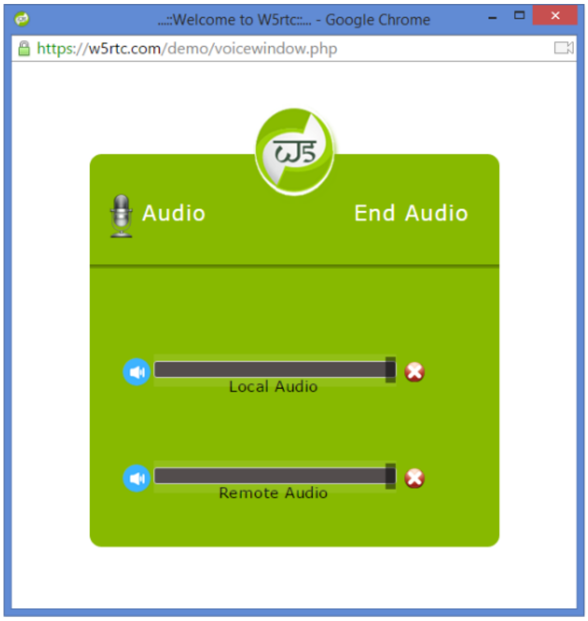
3. Under **Copy & Send URL to your Friend**, copy the URL, and send it to your friend (Callee) to initiate the call or type your friends email address and click Invite, this will open your email client with URL in the email body.
4. When the Callee opens the URL in browser, he/she finds the same window as yours.


Audio
Type participants email id
Invite
------ or ------
Copy & Send URL to your Friend
https://w5rtc.com/demo/voicewind
Copy & Send

5. After starting the audio call, remember to watch out for browser pop up to allow access to you system's camera and micro phone in both Caller and Callee window. This is the extra security provides to ask permission to use your camera and microphone.



6. Now both Caller and Callee are engaged in the audio call.


..::Welcome to W5rtc::.. - Google Chrome
https://w5rtc.com/demo/voicewindow.php
Audio
End Audio
Local Audio
Remote Audio

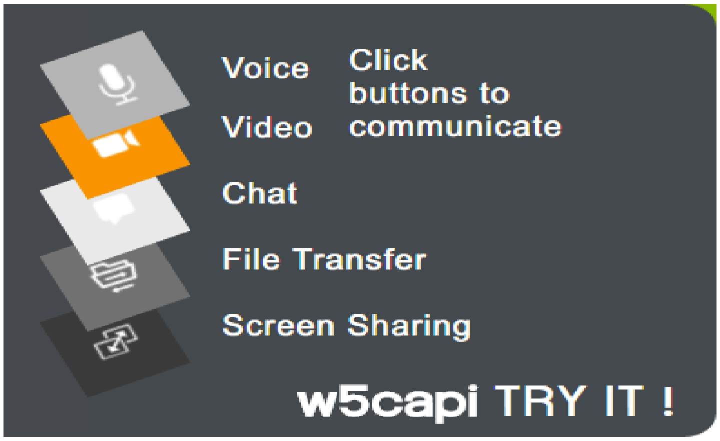
7. To disconnect the call click **End Audio**.

5.3 Running video call

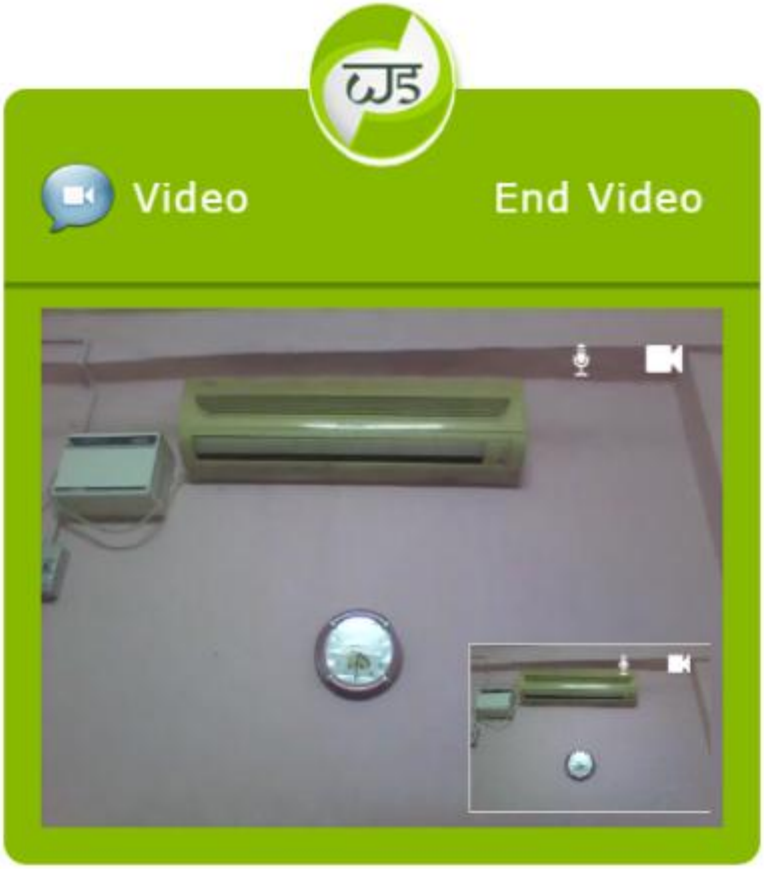
To run the video call:

1. Open a browser and open the index file of the hosted directory. *http://<domainName>/W5CAPI-Demo/index.html*

   The RTC Demo pane appears with the following list of features: Audio, Video, Chat, File Transfer, and Screen Sharing.



2. Click **Video**. The steps here after is same as explained in audio call Step 2 to Step 7 except the Video communication window looks differently as shown in figure below.



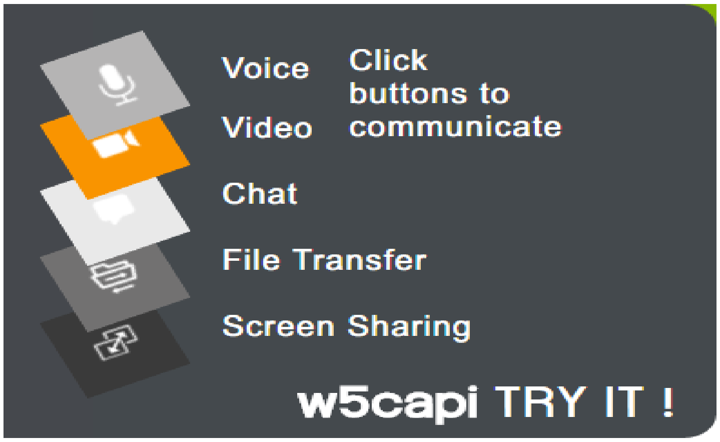
5.4 Running chat call

To run chat:

1. Open a browser and open the index file of the hosted directory.

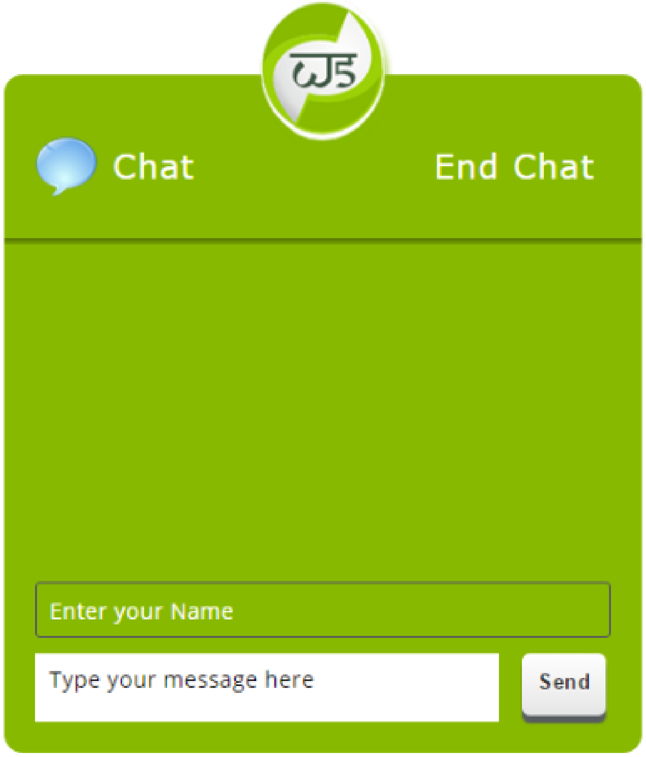
   <Hosting space directory>/W5CAPI-Demo/index.html

   The RTC Demo pane appears with the following list of features: Audio, Video, Chat, File Transfer, and Screen Sharing.

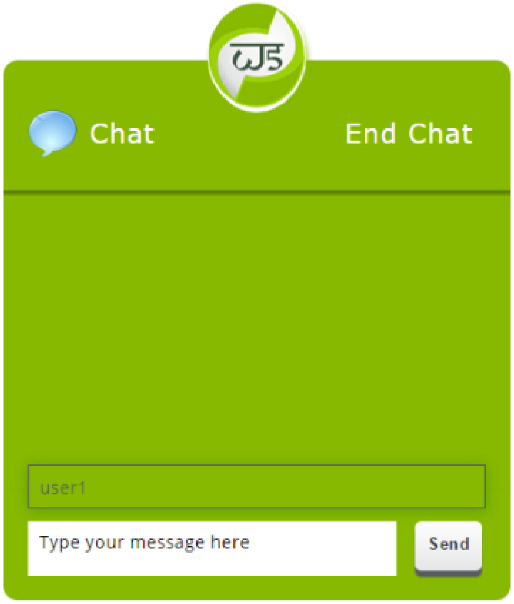


2. Click **Chat**. The steps here after is same as explained in audio call Step 2 to Step 7 except the Chat communication window looks differently as depicted in the following steps.

3. When the Callee/Caller opens the URL in browser, he/she finds the chat window.

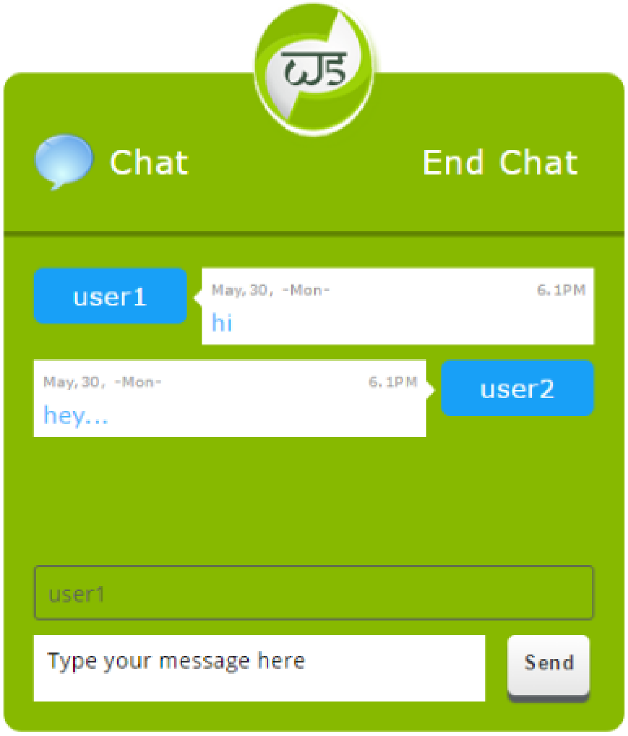


4. Both Caller and Callee enter their respective names inside the text box **Enter Your Name** as shown below.



user1

5. Now both Caller and Callee can chat with each other.



user1 May, 30, -Mon- 6.1PM
hi

May, 30, -Mon- 6.1PM user2
hey...

user1

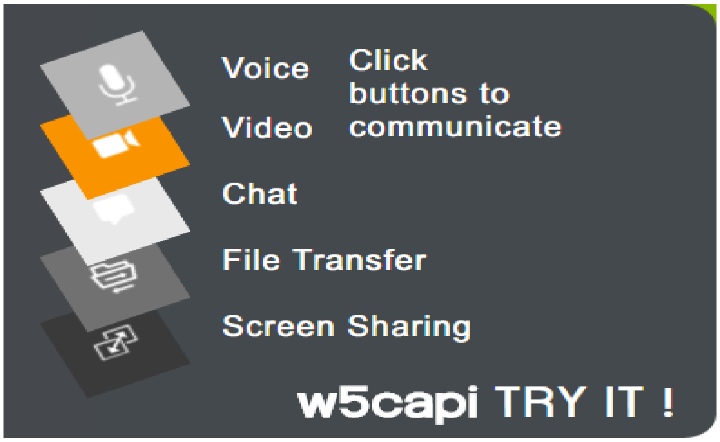
5.5 Running file transfer

To run file transfer:

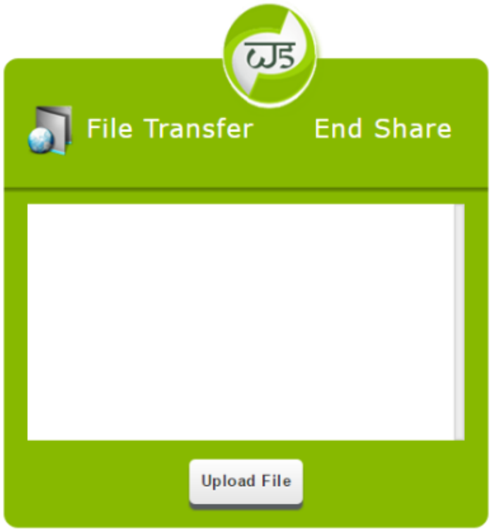
1. Open a browser and open the index file of the hosted directory.

   <hosting space directory>/W5CAPI-Demo/index.html

   The RTC Demo pane appears with the following list of features: Audio, Video, Chat, File Transfer, and Screen Sharing.

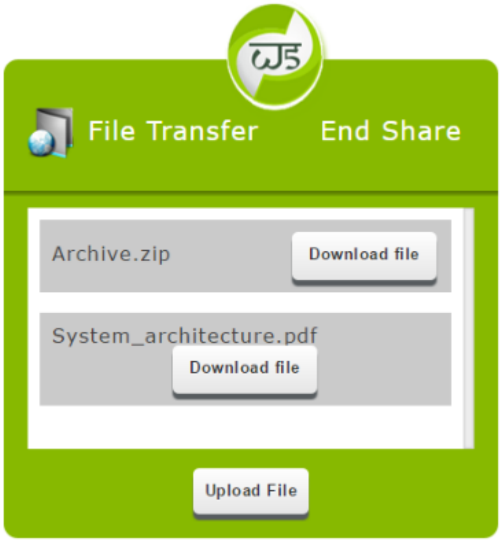


2. Click **File Transfer**. The initial steps are similar to the earlier explained steps with the difference of the screen as depicted below for file transfer.



3. Click **Upload File**, and browse and select the file that you want to transfer.

   The Callee can see the file in his/her file transfer window available for download. The Callee can download the file clicking **Download file button**.



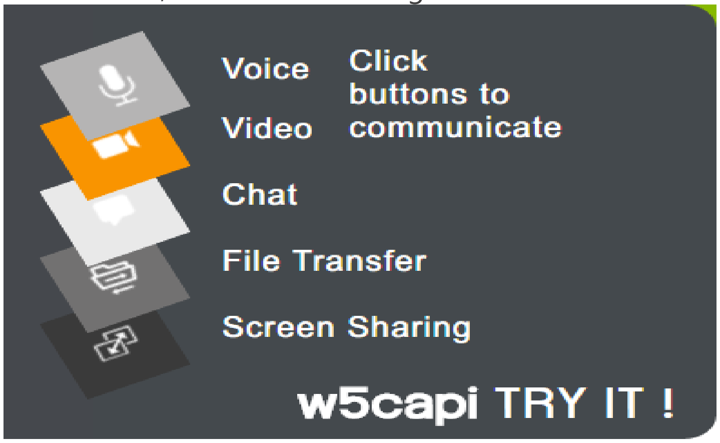
5.6 Running screen sharing

To run the screen sharing:

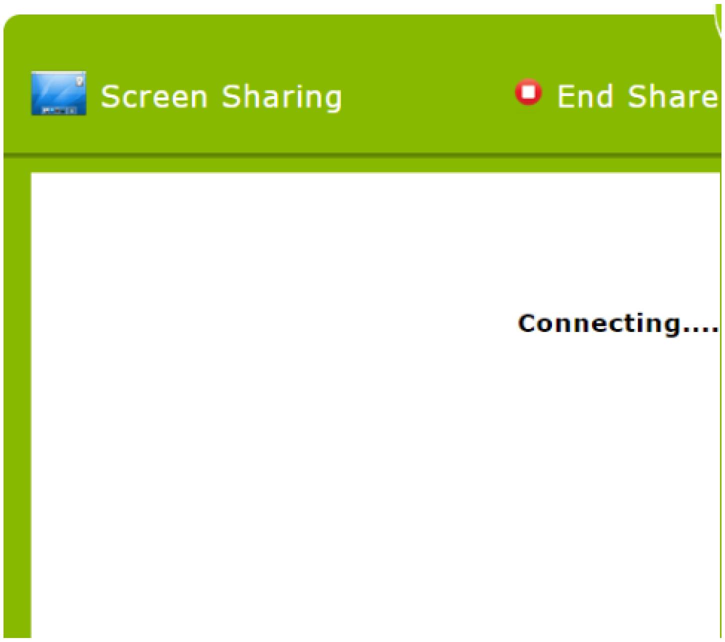
1. Open a browser and open the index file of the hosted directory.

   <hosting space directory>/W5CAPI-Demo/index.html

   The RTC Demo pane appears with the following list of features: Audio, Video, Chat, File Transfer, and Screen Sharing.



2. Click **Screen Sharing**. The initial steps are similar to the earlier explained steps with the difference of the screen as depicted below for screen sharing.



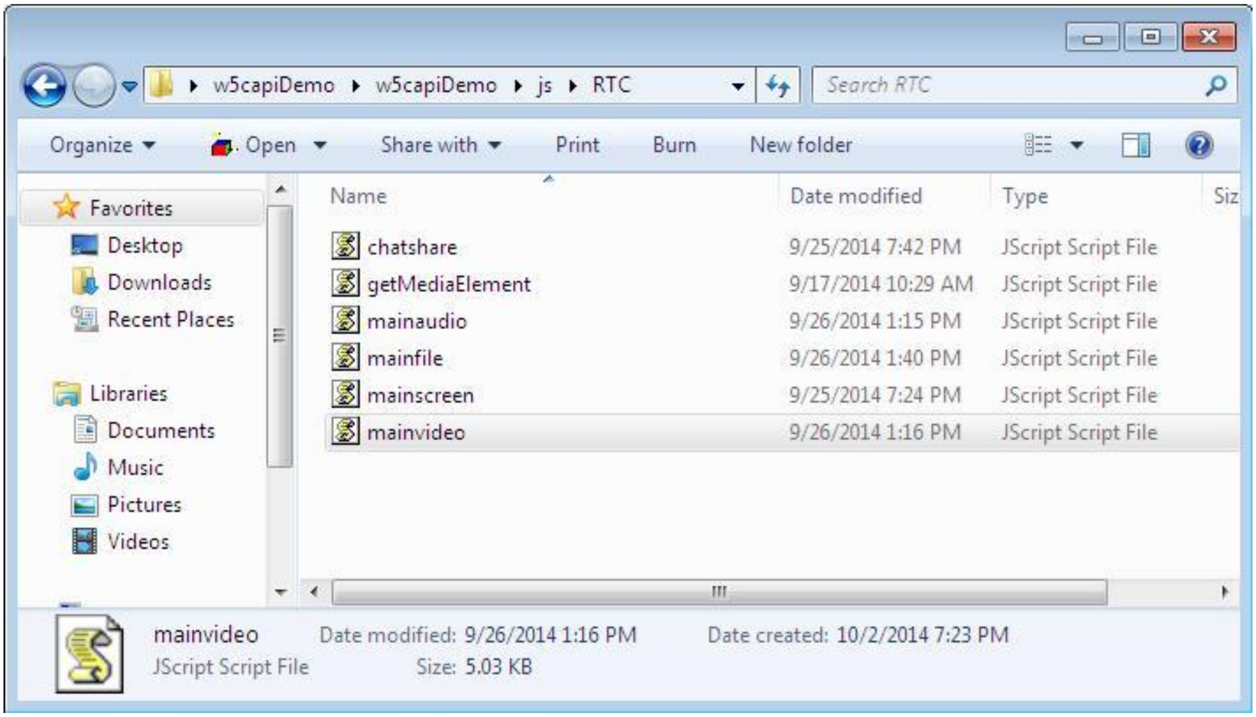
Connecting....

Note: Screen sharing can only work with SSL service. That is the URL must start with https:// and not http://.

6. Demo application Implementation

The W5CAPI-Demo provides .js files for each of the audio, video, chat, file transfer and screen sharing services. These files are located at the *//W5CAPI-Demo/js/RTC*directory.

mainaudio.js: To implement audio communication
mainvideo.js: To implement video communication
chatshare.js: To implement chat communication
mainfile.js: To implement file sharing communication
mainscreen.js: To implement screen sharing communication



Name	Date modified	Type	Siz
chatshare	9/25/2014 7:42 PM	JScript Script File	
getMediaElement	9/17/2014 10:29 AM	JScript Script File	
mainaudio	9/26/2014 1:15 PM	JScript Script File	
mainfile	9/26/2014 1:40 PM	JScript Script File	
mainscreen	9/25/2014 7:24 PM	JScript Script File	
mainvideo	9/26/2014 1:16 PM	JScript Script File	

mainvideo Date modified: 9/26/2014 1:16 PM Date created: 10/2/2014 7:23 PM
JScript Script File Size: 5.03 KB

The *W5CAPI-Demo/index.html* file links to the individual .js files developed for RTC. Each .js file has an application logic attached to it.

Note: API key is required to implement the demo application. Refer the Sec 3 “Getting the license key” to obtain the key.

6.1 Implementing audio call

The audio call is implemented in the mainaudio.js file located at the *//W5CAPI-Demo/js/RTC*directory.

6.1.1 Initializing W5CAPI

This process constructs the w5peer object and initializes the basic functionality of rtc service with default parameters.

Syntax:

```
var W5CAPI = new  W5Peer({key:document.capikey});
```

How we implemented:

```
var W5CAPI = new  W5Peer({key:document.capikey});
```

6.1.2 Assigning userid

A user-id is a unique number assigned to a user. Both the Caller and the Callee are assigned unique user ids during the registration.

Syntax:

```
W5CAPI. registerUserWithId (user-id);
```

How we implemented:

```
window.params.caller=window.params.caller || W5CAPI.userid;
W5CAPI. registerUserWithId (window.params.caller);
window.params.callee=window.params.callee ||uid1;
```

6.1.3 Initialize Stream Events

Media stream refers to the stream of data that are call backed from the media sources i.e. microphone and camera. There are of two types of streams—local and remote. The media streams needs to be captured and displayed in media player. The media streams are attached with event to notify the application about the stream type and intended action.

How we implemented: (Local stream)

```
// Append the local media stream
   W5CAPI.onLocalStream (function(stream) {
        showvoicebox();
        appendMediaElement(stream, 'local');
        localStream=stream;
    });
```

How we implemented: (Remote stream)

```
// Append remote media stream
    W5CAPI.onRemoteStream(function(stream) {

         appendMediaElement(stream, 'remote');
    });
```

6.1.4 Opening a signaling channel

This process opens signaling channel between the client and the W5RTC Signaling Server.

How we implemented:

```
W5CAPI.onSuccess({function(data) {
    setUp.disabled = false;
    setUp.className='textaligncenterdiv';
         }
});
```

6.1.5 Placing a call

This process places a call to the remote user.

How we implemented:

```
W5CAPI.call(window.params.callee,{audio:true});
```

6.1.6 Ending a Call

This process ends an existing call. When a user ends a call W5CAPI fires an event to notify the peers with user-left-all event.

How we Implemented:

```
Var callclose = document.getElementById('closecall');
    callclose.onclick = function(){
         W5CAPI.stopCall();
         location.reload(true);
         location.href = location.href.replace(location.search, '');
    }
```

```
// Notification if the other user has disconnected.
W5CAPI.on('user-left-all',function(id){
    if(id==window.params.callee){
          alert('The user whose id is:'+ id +' has
disconnected...!!!');
          location.href = location.href.replace(location.search, '');
    }
});
```

6.1.7 Full Audio code extract

```
------------------------------------------
// Audio Call DEMO functionality
function audioCall()
{
    var params = {},
    r = /([^&=]+)=?([^&]*)/g;
    function d(s){
          return decodeURIComponent(s.replace(/\+/g, ' '));
    }
    var match, search = window.location.search.toLowerCase();
    while (match = r.exec(search.substring(1)))
          params[d(match[1])] = d(match[2]);

    //User is creating instance of W5peer class for acessing the
required methods/functions defined in that particular class.
    var W5CAPI = new W5Peer({key:document.capikey});

    // Generate Random string
    function getRandomString() {
          return (Math.random() * new
Date().getTime()).toString(36).replace( /\./g , '');
    }
    var uid= getRandomString();
    window.params = params;
    window.params.caller= window.params.caller ||
    W5CAPI.userid; window.params.callee= window.params.callee
    ||uid; W5CAPI.registerUserWithId( window.params.caller);

    // Check if browser is Internet
    Explorer function isIE () {
var myNav = navigator.userAgent.toLowerCase();
return (myNav.indexOf('msie') != -1) ?
parseInt(myNav.split('msie')[1]) : false;
       }
```

```
document.getElementById('user_name1').value =
location.href.replace(location.search, '') + '?caller='
+window.params.callee + '&callee=' +window.params.caller;

    // Describes the bandwidth of the media
    W5CAPI.bandwidth = {audio: 50};        // 50kbs

    //  Some      Global      variables
    declared var localStream;
    var removeElement;
    var removeRemoteElement;

    // Append the local media stream
    W5CAPI. onLocalStream (function(stream) {
          showvoicebox();
          appendMediaElement(stream, 'local');
          localStream=stream;
     });

    // Append remote media stream
    W5CAPI. onRemoteStream (function(stream) {

          appendMediaElement(stream, 'remote');
     });

    // Describes the Media Element Controls
    function appendMediaElement(stream, type)
    {
var mediaElement = getMediaElement(stream, {
buttons: ['stop', 'volume-slider', 'full-screen', 'mute-audio',
'mute-video'],
width: innerWidth / 2 - 20,
showOnMouseEnter: false
        });
         document.getElementById('audio').appendChild(mediaElement);
mediaType = type;
        removeMediaElement = mediaElement;
        var isIE = /*@cc_on!@*/ false || !! document.documentMode;
        var isSafari =
Object.prototype.toString.call(window.HTMLElement).indexOf('Construc
tor') > 0;

        if (isIE || isSafari) {
            setTimeout(function() {
                var audiotag = document.getElementById(id);
                attachMediaStream(audiotag, stream);
            }, 400);
        } else {

        }
         var callback = {
   mediaElement: mediaElement,
   stream: stream
        };
         stream.onended = function(){
   W5CAPI.emit('stream-removed', callback);
         }

   // Functionality for Remote Media Element
         if(type=='remote'){
               removeRemoteElement=mediaElement;
               var h2=document.createElement('h2');
               h2.innerHTML='Remote Audio';
               mediaElement.appendChild(h2);

   document.getElementById('crossbutton').onclick=function(){
                     mediaElement.style.opacity = 0;
                     setTimeout(function(){
                           if (mediaElement.parentNode) {

   mediaElement.parentNode.removeChild(mediaElement);
                           }
                     }, 800);
               }
         }

   // Functionality for Local Media Element
if(type == 'local') {
               removeElement=mediaElement;
               var h2=document.createElement('h2');
               h2.innerHTML='Local Audio';
               mediaElement.appendChild(h2);
            mediaElement.media.muted = true;
   mediaElement.media.volume = 0;
               document.querySelector('.stop').id='crossButton1';

   document.getElementById('crossButton1').onclick=function(){
                     mediaElement.style.opacity = 0;
setTimeout(function() {
         if (mediaElement.parentNode) {
               mediaElement.parentNode.removeChild(mediaElement)
               ;
         }
         }, 800);
                     stream.stop();
w5capi.sendCustomMessage({ removeMediaElement1:
                     true,
                     to: window.params.callee+document.capikey
```

```
           });
         }
       }
     }
var setUp = document.getElementById('starttalkdtndiv');

   // Setup Socket Connection by calling the openSignalingChannel
function.
W5CAPI.onSuccess(function(data) {
              setUp.disabled = false;
         }
    });
   w5capi.sendCustomMessage({ enableButton:
        true,
        to: window.params.callee+document.capikey
    });
    //........custom-signaling-message function defined for handling
user's some custom message...............
    w5capi.on('custom-signaling-message', function(message) {
        console.log('custom signaling message', message);
        if (message.to != w5capi.userid) return;

        // Start the Call
        setUp.onclick = function() {
            showvoicebox();
            this.disabled = true;
            var div2 = document.createElement('div');
            div2.id = 'connectid';
            div2.style.width = '400px';
            div2.style.height = '30px';
            div2.style.position = 'absolute';
            div2.style.marginTop = '80px';
            div2.style.marginLeft = '100px';
            div2.style.fontSize = '18px';
            div2.innerHTML = '<font color="white">' + '<b>' +
"Connecting..." + '</b>' + '</font>';
            document.getElementById('writeText').appendChild(div2)
            ; connectLocalNotyDiv = div2;
            //when link sender initiated call request to link
reciever
            w5capi.sendCustomMessage({
                to: window.params.callee+document.capikey,
                doYouWantToRecieveCall: true
            });
        }
        if (message.enableButton) {
            w5capi.sendCustomMessage({
                enableButton1: true,
                to: message.userid
            });
        }
        //Remote client got msg from call initiator
        else if (message.enableButton1) {
            showvoicebox();
            var div = document.createElement('div');
            div.id = 'nameid';
            div.style.width = '300px';
            div.style.height = '30px';
            div.style.position = 'absolute';
            div.style.marginTop = '80px';
            div.style.marginLeft = '80px';
            div.style.fontSize = '16px';
            div.innerHTML = '<font color="white">' + '<b>' + "Thank
You for joining...!" + '</b>' + '</font>';
            document.getElementById('writeText').appendChild(div);
            var div1 = document.createElement('div');
            div1.id = 'starttalkdtndiv';
            div1.innerHTML = '<div class="btn signinbtndivone">' +
"Start Audio" + '</div>';
            div1.style.marginTop = '120px';
            div1.style.marginLeft = '130px';
            document.getElementById('writeText').appendChild(div1)
            ; div2 = document.createElement('div');
            div2.id = 'connectid';
            div2.style.width = '400px';
            div2.style.height = '30px';
            div2.style.position = 'absolute';
            div2.style.marginTop = '80px';
            div2.style.marginLeft = '100px';
            div2.style.fontSize = '18px';
            div2.style.display = 'none';
            div2.innerHTML = '<font color="white">' + '<b>' +
"Connecting....." + '</b>' + '</font>';
            document.getElementById('writeText').appendChild(div2)
            ; thankYouNotyDiv = div;
            buttonDiv = div1;
            connectNotyDiv = div2;

            buttonDiv.onclick = function() {
                connectNotyDiv.style.display =
                'block'; buttonDiv.style.display =
                'none'; thankYouNotyDiv.style.display
                = 'none'; w5capi.sendCustomMessage({
                    to: window.params.callee+document.capikey,
                    pleaseRecieveTheCall: true
                });
            }
        } else {
            setTimeout(function() {
                if (removeMediaElement &&
removeMediaElement.parentNode && !message.iWantToRecieve
&& !message.yesRecieving) {
                    try {

removeMediaElement.parentNode.removeChild(removeMediaElement);
                    } catch (e) {}
                }
            }, 800);
        }

        //link reciever got incoming call request from link sender
        if (message.doYouWantToRecieveCall) {
            buttonDiv.style.display = 'none';
            thankYouNotyDiv.style.display = 'none';
            document.getElementById('incomingCall').style.display =
'block';
            incomingCall = document.getElementById('incomingCall');
            // if link reciever accepting incoming call
            document.getElementById('right').onclick = function() {

document.getElementById('incomingCall').style.display =
                'none'; thankYouNotyDiv.style.display =
                'none'; buttonDiv.style.display = 'none';
                w5capi.sendCustomMessage({
                    to: message.userid,
                    iWantToRecieve: true
                });
            }
            //if link reciever rejecting the incoming call
            document.getElementById('worng').onclick = function() {

document.getElementById('incomingCall').style.display =
                'none'; thankYouNotyDiv.style.display =
                'none'; buttonDiv.style.display = 'none';
                w5capi.sendCustomMessage({
                    to: message.userid,
                    rejectingTheCall: true
                })
            }
        }
        //when link sender got confirmation that other user is ready
to join his call
        //initiated call.
        if (message.iWantToRecieve) {
            connectLocalNotyDiv.style.display =
            'none'; /*
            Old
            Syntax */
            //w5capi.call(window.params.callee);
            /*
             New
            Syntax */
            console.error('executed once');
            //w5capi.call(window.params.callee,{audio:true});
        }
        //link sneder got call request from link
        reciever. if (message.pleaseRecieveTheCall) {
            showvoicebox();
            document.getElementById('incomingCall').style.display =
'block';
            incomingCall1 = document.getElementById('incomingCall');
            //if agree to recieve the call request.
            document.getElementById('right').onclick = function() {

document.getElementById('incomingCall').style.display =
                'none'; w5capi.sendCustomMessage({
                    to: message.userid,
                    yesRecieving: true
                });
            }
            //if rejecting the call request.
            document.getElementById('worng').onclick = function() {

document.getElementById('incomingCall').style.display =
                'none'; w5capi.sendCustomMessage({
                    to: message.userid,
                    iAmRejecting: true
                });
            }
        }
        //The link reciever is intiating call to link
        sender if (message.yesRecieving) {
            connectNotyDiv.style.display =
             'none'; /*
            Old
            Syntax */
            //w5capi.call(window.params.callee);
            /*
             New
            Syntax */
            console.error('executed once');
            w5capi.call(window.params.callee,{audio:true});
        }

        //link reciever got message that other user has rejected the
call.
        if (message.iAmRejecting) {
            var str = connectNotyDiv.innerHTML;
```

```
              var res = str.replace("Connecting.....", "Call
Rejected....");
              connectNotyDiv.innerHTML =
              res; setTimeout(function() {
                  connectNotyDiv.style.display =
              'none'; }, 4000);
          }
          //link sender got message that link reciever has
rejected the call.
          if (message.rejectingTheCall) {
              var str = connectLocalNotyDiv.innerHTML; var
              res = str.replace("Connecting...", "Call
Rejected....");
              connectLocalNotyDiv.innerHTML =
              res; setTimeout(function() {
                  connectLocalNotyDiv.style.display =
              'none'; }, 4000);
          }
      })
    // Media Stream Removed
    W5CAPI.on('stream-removed', function(e) {
if (e.mediaElement && e.mediaElement.parentNode) {
e.mediaElement.parentNode.removeChild(e.mediaElement);
          }
      });

    // End the call
var callclose = document.getElementById('closecall');
     callclose.onclick = function(){
          location.reload(true);
          location.href = location.href.replace(location.search, '');

     }

    // Notify when user has disconnected the
    call W5CAPI.on('user-left-all',function(id){
          if(id==window.params.callee){
          try{
          localStream.stop();
                     if(removeElement && removeElement.parentNode){

    removeElement.parentNode.removeChild(removeElement);
                     }
                }catch(e){
                console.log(e);
                }
                var n = noty({
                  text: 'The user whose id is:' + id + ' has
disconnected...!!!',
                  type: 'information',
                dismissQueue: true,
                layout: 'topCenter',
                theme: 'defaultTheme',
                buttons: [{
                    addClass: 'btn btn-primary',
                    text: 'Ok',
                    onClick: function($noty)
                        { $noty.close();
                        if (thankYouNotyDiv && buttonDiv &&
incomingCall) {
                            thankYouNotyDiv.style.display = 'none';
                            buttonDiv.style.display = 'none';
                            incomingCall.style.display = 'none';
                        }
                        if (connectLocalNotyDiv) {
                            connectLocalNotyDiv.style.display =
'none';
                        }
                        if (connectNotyDiv) {
                            connectNotyDiv.style.display = 'none';
                        }
                        if (incomingCall1) {
                            incomingCall1.style.display = 'none';
                        }
              location.href = location.href.replace(location.search,
'');

         }
   });
}
---------------------------------------------
```

6.2 Implementing Video call

The video call is implemented in the mainvideo.js file located at the *//W5CAPI-Demo/js/RTC*directory.

6.2.1 Initializing W5CAPI

This process constructs the w5peer object and initializes the basic functionality of rtc service with default parameters.

How we implemented:

```
var W5CAPI = new  W5Peer({key:document.capikey});
```

6.2.1 Assigning userid

A user-id is a unique number assigned to a user. Both the Caller and the Callee are assigned unique user ids during the registration.

Syntax:

```
W5CAPI. registerUserWithId (user-id);;
```

How we implemented:

```
window.params = params;
window.params.caller=window.params.caller || W5CAPI.userid;
W5CAPI. registerUserWithId (window.params.caller);
window.params.callee=window.params.callee ||uid1;
```

6.2.2 Initialize Stream Events

Media stream refers to the stream of data that are call backed from the media sources i.e. microphone and camera. There are of two types of streams—local and remote. The media streams needs to be captured and displayed in media player. The media streams are attached with event to notify the application about the stream type and intended action.

How we implemented: (Local stream)

```
W5CAPI. onLocalStream (function(stream)
    { if(origNotificationDiv&&buttonDiv){
          origNotificationDiv.style.display='none';
          destNotificationDiv.style.display = 'none';
          buttonDiv.style.display='none';
    }
    showvideobox();
appendMediaElement(stream, 'local');
});
```

How we implemented: (Remote stream)

```
W5CAPI. onRemoteStream (function(stream) {
    appendMediaElement(stream, ‘remote’);
});
```

6.2.3 Opening a signaling channel

This process opens signaling channel between the client and the W5RTC Signaling Server.

How we implemented:

```
W5CAPI.onSuccess(function(data) {
    setUp.disabled = false;
    setUp.className='textaligncenterdiv';
          }
});
```

6.2.4 Placing a call

This process places a call to the remote user.

How we implemented:

```
W5CAPI.call(window.params.callee,{audio:true,video:true});
```

6.2.5 Ending a Call

This process ends an existing call. When a user ends a call W5CAPI fires an event to notify the peers with user-left-all event.

How we Implemented:

```
varcallclose = document.getElementById('closecall');
    callclose.onclick = function(){
          W5CAPI.stopCall();
          location.reload(true);
          location.href = location.href.replace(location.search, '');
    }

// Notification if the other user has disconnected.
W5CAPI.on('user-left-left',function(id){
    if(id==window.params.callee){
          alert('The user whose id is:'+ id +' has
disconnected...!!!');
          location.href = location.href.replace(location.search, '');
    }
});
```

6.2.6 Full Video code extract

```
----------------------------------------------------
//Video Call DEMO functionality
function videoCall()
{
    var params = {},
     r = /([^&=]+)=?([^&]*)/g;
    function d(s){
          return decodeURIComponent(s.replace(/\+/g, ' '));
    }
    var match, search = window.location.search.toLowerCase();
    while (match = r.exec(search.substring(1)))
params[d(match[1])] = d(match[2]);

    //User is creating instance of W5peer class for acessing the
required methods/functions defined in that particular class.
    var W5CAPI = new  W5Peer({key:document.capikey});

    //Generate Random string
    function getRandomString(){
          return (Math.random() * new
Date().getTime()).toString(36).replace( /\./g , '');
    }


    //Assign user id
    var uid1= getRandomString();
    window.params = params;
    window.params.caller=window.params.caller || W5CAPI.userid;
    W5CAPI. registerUserWithId( window.params.caller);
    window.params.callee=window.params.callee ||uid1;

w5capiReady(function(){
        // alert(window.params.caller);
         w5capi.registerUserWithId(window.params.caller);
     });

      /*
     @ New
     Syntax */
     //accept incomming call
     w5capi.onCall(function(data) {

         w5capi.accept(data);
     });


    document.getElementById('user_name2').value
=location.href.replace(location.search, '') + '?caller=' +
window.params.callee + '&callee=' +window.params.caller ;

    // Describes the bandwidth of the
    media W5CAPI.bandwidth = {
          audio: 50,
          video: 256
     };

    //  Some      Global      variables
    declared var removeMediaElement;
    var  localMediaStream;

    // Append the local media stream
    W5CAPI. onLocalStream (function(stream) {
          showvideobox();
          appendMediaElement(stream, 'local');
     });

    // Append remote media stream
    W5CAPI. onRemoteStream (function(stream) {
          appendMediaElement(stream, 'remote');
     });

     // Describes the Media Element Controls
function appendMediaElement(stream, type) {
var mediaElement = getMediaElement(stream, {
buttons: ['take-snapshot', 'volume-slider', 'full-screen', 'mute-
audio', 'mute-video'],
width: innerWidth / 2 - 20,
showOnMouseEnter: false
         });
   // removeMediaElement=mediaElement;
          stream.onended = function() {
                W5CAPI.emit('stream-removed', callback);
          }
          var callback = {
                mediaElement: mediaElement,
                stream: stream
         };
mediaElement.media.play();

    // Append the Remote Media Element in the specific container
if(type == 'remote'){
                document.querySelector('.big
-video').appendChild(mediaElement);
                forRemoteGui();
         }

     // Append the Local Media Element in the specific
          container if(type == 'local') {
                localMediaStream=stream;
                removeMediaElement=mediaElement;
                document.querySelector('.small-
video').appendChild(mediaElement);
                forLocalGui();
          }
    }
    w5capi.on('stream-removed', function(stream) {
        var mediaElement = document.getElementById(stream.uuid);
        if (mediaElement) {
            mediaElement.parentNode.removeChild(mediaElement);
        }
     });

 W5CAPI.on('stream-removed', function(e) {
         if (e.mediaElement && e.mediaElement.parentNode) {
e.mediaElement.parentNode.removeChild(e.mediaElement);
        }
     });
var setUp = document.getElementById('starttalkdtndiv');

    // Setup Socket Connection by declaring openSignalingChannel
function.
    W5CAPI.onSuccess(function(data){
              setUp.disabled = false;
              setUp.className='textaligncenterdiv';
w5capi.sendCustomMessage ({
         enableButton:true,
         to:window.params.callee+document.capikey
        }
    }); w5capi.on('custom-signaling-
    message',function(message){ console.log('custom
    signaling message',message);
    if(message.to!=w5capi.userid)return;
    if(message.enableButton){
                setUp.disabled = false;
               // Start the Call
             setUp.onclick = function () {
               window.action1=null;
               showvideobox();
               var div2=document.createElement('div');
               div2.id='connectid'; div2.style.width='400px';
               div2.style.height='30px';
               div2.style.position='absolute';
               div2.style.marginTop='80px';
               div2.style.marginLeft='130px';
               div2.style.fontSize ='18px';
               div2.innerHTML='<font color="black">'+'<b>'+

"Connecting..."+'</b>'+'</font>';

    document.getElementById('writeText').appendChild(div2)
               ; connectLocalNotyDiv=div2;
                    //when link sender initiated call request to link
reciever
               w5capi.sendCustomMessage({
                    to:window.params.callee+document.capikey,
                    readyForVideoCall:true
              });
         }
         w5capi.sendCustomMessage({
              enableButton1:true,
              to:message.userid
         });
   }
         //Remote client got msg from call initiator
         else if(message.enableButton1){
         showvideobox();
          var div=document.createElement('div');
         div.id='nameid';
         div.style.width='300px';
         div.style.height='30px';
         div.style.position='absolute';
         div.style.marginTop='20px';
         div.style.marginLeft='70px';
         div.style.fontSize='16px';
          div.innerHTML='<font color="black">'+'<b>'+"Thank You for
joining...!"+'</b>'+'</font>';
          document.getElementById('writeText').appendChild(div);
          thankYouNotyDiv=div;
         var div1=document.createElement('div');
        div1.innerHTML='<div class="btn signinbtndivone">'+"Start
Video"+'</div>';
        div1.style.marginTop='50px';
        div1.style.marginLeft='130px';
        document.getElementById('writeText').appendChild(div1)
        ; div2=document.createElement('div');
                    div2.id='connectid';
                    div2.style.width='400px';
               div2.style.height='30px';
               div2.style.position='absolute';
               div2.style.marginTop='80px';
               div2.style.marginLeft='130px';
                 div2.style.fontSize ='18px';
                 div2.style.display='none';
                    div2.innerHTML='<font
color="black">'+'<b>'+"Connecting..."+'</b>'+'</font>'
;

   document.getElementById('writeText').appendChild(div2)
         ; buttonDiv=div1;
         connectNotyDiv=div2;
         buttonDiv.onclick=function(){
         connectNotyDiv.style.display = 'block';
           buttonDiv.style.display='none';
          thankYouNotyDiv.style.display='none';
           w5capi.sendCustomMessage({
                to:window.params.callee+document.capikey,
                itsOneVideoCall:true
           });
         }
    }

           //link reciever got incoming call request from link
sender
    if(message.readyForVideoCall)
    {

    document.getElementById('incomingVideo').style.display='block';
         incomingCall=document.getElementById('incomingVideo');
         thankYouNotyDiv.style.display='none';
         buttonDiv.style.display='none';
               // if link reciever accepting incoming call
         document.getElementById('right').onclick=function(){

    document.getElementById('incomingVideo').style.display='none';
              w5capi.sendCustomMessage({
                    to:message.userid,
                    yesIamReady:true
              });
         }
              //if link reciever rejecting the incoming call
        document.getElementById('wrong').onclick=function(){

document.getElementById('incomingVideo').style.display='none';
              w5capi.sendCustomMessage({
                    to:message.userid,
                    rejectingTheCall:true
              })
        }
    }
         //when link sender got confirmation that other user
is ready to join his call
      //initiated call.
    if(message.yesIamReady){
         connectLocalNotyDiv.style.display='none';
         /*
            Old
            Syntax */
            //w5capi.call(window.params.callee);
            /*
             New
            Syntax */

w5capi.call(window.params.callee,{audio:true,video:true});
    }
         //link sneder got call request from link
    reciever. if(message.itsOneVideoCall){
```

```
             showvideobox();

     document.getElementById('incomingVideo').style.display='block';

     incomingVideoCall1=document.getElementById('incomingVideo')
                  ; //if agree to recieve the call request.
             document.getElementById('right').onclick=function(){

     document.getElementById('incomingVideo').style.display='none';
                  w5capi.sendCustomMessage({
                          to:message.userid,
                          yesItsVideo:true
                  });
             }
                      //if rejecting the call request.
            document.getElementById('wrong').onclick=function(){

document.getElementById('incomingVideo').style.display='none';
                  w5capi.sendCustomMessage({
                          to:message.userid,
                          iAmRejecting:true
                  });
            }
     }
             //The link reciever is intiating call to link
     sender if(message.yesItsVideo){
             connectNotyDiv.style.display='none';
             /*
                  Old
                  Syntax */
                  //w5capi.call(window.params.callee);
                  /*
                   New
                  Syntax */

w5capi.call(window.params.callee,{audio:true,video:true});
     }

         //link reciever got message that other user has rejected the
call.
     if(message.iAmRejecting){
             console.log('call rejected'); var
             str=connectNotyDiv.innerHTML;
             console.log('str',str);
             var res=str.replace("Connecting...","Call
             Rejected...."); connectNotyDiv.innerHTML=res;
             setTimeout(function(){
                     connectNotyDiv.style.display='none'; },
                     4000);
     }
```

```
          //link sender got message that link reciever has rejected
the call.
    if(message.rejectingTheCall){
          var str=connectLocalNotyDiv.innerHTML;
          var res=str.replace("Connecting...","Call
          Rejected...."); connectLocalNotyDiv.innerHTML=res;
          setTimeout(function(){
                connectLocalNotyDiv.style.display='none'; },
                4000);

    }
    })
    // End the call
    var callclose = document.getElementById('closecall');
    callclose.onclick = function(){
          W5CAPI.stopCall();
          location.reload(true);
          location.href = location.href.replace(location.search, '');
    }

    // Notify when user left the call
    W5CAPI.on('user-left-all',function(id){
          if(id==window.params.callee){
                try{
                      localMediaStream.stop();
                      if(removeMediaElement &&
removeMediaElement.parentNode){

    removeMediaElement.parentNode.removeChild(removeMediaElement);
                      }
                }catch(e){
                      console.log(e);
                }
                alert('The user whose id is:'+ id +' has
disconnected...!!!');
                location.href = location.href.replace(location.search,
'');
          }
    });
}
----------------------------------------------------
```

6.3 Implementing chat

The chat call is implemented in the chatshare.js file located at the *//W5CAPI-Demo/js/RTC*directory.

6.3.1 Initializing W5CAPI

This process constructs the w5peer object and initializes the basic functionality of rtc service with default parameters.

How we implemented:

```
var W5CAPI = new  W5Peer({key:document.capikey});
```

6.3.2 Assigning userid

A user-id is a unique number assigned to a user. Both the Caller and the Callee are assigned unique user ids during the registration.

How we implemented:

```
window.params = params;
window.params.caller=window.params.caller || W5CAPI.userid;
W5CAPI. registerUserWithId( window.params.caller);
window.params.callee=window.params.callee ||uid1;
```

6.3.3 Opening a signaling channel

This process opens signaling channel between the client and the W5RTC Signaling Server.

How we implemented:

```
W5CAPI.onSuccess(function(data) {
    setUp.disabled = false;
    setUp.className='textaligncenterdiv';
        }
});
```

6.3.4 Initiate a chat session

This process places a call to the remote user to establish a chat session.

How we implemented:

```
W5CAPI.call(window.params.callee, {data:true});
```

6.3.5 Handling data channel events

When there is activity (opening data channel or receive text or receive file or file progress, closing data channel, etc) in data channel there is an callback fired for each of these events. Here for chat we handle the *open, close, error* event when data channel open, peer receives message on data channel and data channel closes respectively.

How we implemented:

```
//This event will be handled when user will get messages through
data channel.
W5CAPI.onMessage(function(data){
         appendRemoteMessages(data.message, data.fullName);
});

//This callback handle datachannel event .
W5CAPI.onDatachannel(function(data) {
If(data.event==’open’)
{
         showchatbox(); document.getElementById('text-
         message').disabled = false;
}
});

//This event is handling when data channel is closed.
W5CAPI. onDatachannel (function(event){
If(data.event==’close’)
{
W5CAPI.channels.value=''; document.getElementById('text-
         message').disabled = true;

}

});
```

6.3.6 Sending Text to Peer

W5CAPI provides simple API to send text messages to the peer; user needs to use the API to send text.

How we implemented:

```
W5CAPI.send({
         fullName: document.getElementById('full-name').value,
         message: this.value
         });
```

6.3.7 Handling call end event

When a user leaves the chat session W5CAPI fires an event to notify the peers with user-left-all event about the end of a session.

How we implemented:

```
// Notification if the other user has disconnected.
W5CAPI.on('user-left-all',function(id){
    if(id==window.params.callee+apikey){ alert('The
         user whose id is:'+ id +' has
disconnected...!!!');
         location.href = location.href.replace(location.search, '');
    }
});
```

6.3.8 Full Chat code extract

```
-------------------------------------
// Text Chat DEMO functionality
function chatshare()
{
    var params = {},
    r = /([^&=]+)=?([^&]*)/g;
    function d(s) {
         return decodeURIComponent(s.replace(/\+/g, ' '));
    }
    var match, search = window.location.search.toLowerCase();
    while (match = r.exec(search.substring(1)))
         params[d(match[1])] = d(match[2]);

    //User is creating instance of W5peer class for acessing the
required methods/functions defined in that particular class.
    var W5CAPI= new  W5Peer({key:document.capikey});
    //For generating random string
    function getRandomString()
    {
          return (Math.random() * new
Date().getTime()).toString(36).replace( /\./g , '');
    }
    var uid4=getRandomString();

    //get the caller and calee
    ids window.params = params;
    window.params.caller= window.params.caller || W5CAPI.userid;
    window.params.callee= window.params.callee ||uid4;
    W5CAPI. registerUserWithId (window.params.caller);

    w5capiReady(function(){

     });
    console.log(window.params.caller);
 w5capi.registerUserWithId(window.params.caller);

    //For generating URL for remote side
    document.getElementById('user_name3').value =
location.href.replace(location.search, '') + '?caller=' +
window.params.caller + '&callee=' +window.params.caller;
    var setUp = document.getElementById('starttalkdtndiv');
    //Open socket connection by calling openSignalingChannel
    function W5CAPI.onSuccess(function(data) {
                setUp.disabled = false;
                setUp.className='textaligncenterdiv';
w5capi.sendCustomMessage({ enableButton:
         true,
         to: window.params.callee+document.capikey
          }
    });

w5capi.onCall(function(data){

              w5capi.accept(data);
         });



     //.......some global variables declared...............
     var thankYouNotyDiv;
     var buttonDiv;
     var connectNotyDiv;
     var connectLocalNotyDiv;
     var incomingCall;
     var incomingCall1;
     //........................end........................
     //........custom-signaling-message function defined for handling
 user's some custom message...............
     w5capi.on('custom-signaling-message', function(message) {
         console.log('custom signaling message', message);
         if (message.to != w5capi.userid)
         return; if (message.enableButton) {
             //when link sender initiated call request to link
 reciever
             setUp.onclick = function()
                 { showchatbox();
                 this.disabled = true;
                 var div2 = document.createElement('div');
                 div2.id = 'connectid';
                 div2.style.width = '400px';
                 div2.style.height = '30px';
                 div2.style.position = 'absolute';
                 div2.style.fontSize = '18px';
                 div2.style.marginTop = '80px';
                 div2.style.marginLeft = '100px';
                 div2.innerHTML = '<font color="white">' + '<b>' +
 "Connecting....." + '</b>' + '</font>';

 document.getElementById('writeText').appendChild(div2);
                 connectLocalNotyDiv = div2;
                 w5capi.sendCustomMessage({
                     to: window.params.callee+document.capikey,
                     isThisChat: true
                 });
             }
             w5capi.sendCustomMessage({ enableButton1:

                 true, to: message.userid

             });
         }
         //Remote client got msg from call initiator
         if (message.enableButton1) {
             showchatbox();
             var div = document.createElement('div');
             div.id = 'nameid';
             div.style.width = '300px';
             div.style.height = '30px';
             div.style.position = 'absolute';
             div.style.marginTop = '20px';
             div.style.marginLeft = '80px';
             div.style.fontSize = '16px';
             div.innerHTML = '<font color="white">' + '<b>' + "Thank
 You for joining...!" + '</b>' + '</font>';
             document.getElementById('writeText').appendChild(div);
             thankYouNotyDiv = div;
             var div1 = document.createElement('div'); div1.innerHTML
             = '<div class="btn signinbtndivone">' +
 "Start Chat" + '</div>'; div1.style.marginTop
             = '50px'; div1.style.marginLeft =
             '140px';
             document.getElementById('writeText').appendChild(div1)
             ; var div2 = document.createElement('div');
             div2.id = 'connectid';
             div2.style.width = '400px';
             div2.style.height = '30px';
             div2.style.position = 'absolute';
             div2.style.fontSize = '18px';
             div2.style.marginTop = '80px';
             div2.style.marginLeft = '100px';
             div2.style.display = 'none';
             div2.innerHTML = '<font color="white">' + '<b>' +
 "Connecting....." + '</b>' + '</font>';
             document.getElementById('writeText').appendChild(div2)
             ; buttonDiv = div1;
             connectNotyDiv = div2;
             buttonDiv.onclick = function() {
                 connectNotyDiv.style.display =
                 'block'; thankYouNotyDiv.style.display
                 = 'none'; buttonDiv.style.display =
                 'none'; w5capi.sendCustomMessage({
                     to: window.params.callee+document.capikey,
                     readyForChat: true
                 });
             }
         }
         //link reciever got incoming call request from link sender
         if (message.isThisChat) {
             thankYouNotyDiv.style.display = 'none';
             buttonDiv.style.display = 'none';
             document.getElementById('incomingChat').style.display =
 'block';
             incomingCall = document.getElementById('incomingChat');
             // if link reciever accepting incoming call
             document.getElementById('right').onclick = function() {

 document.getElementById('incomingChat').style.display =
                 'none'; w5capi.sendCustomMessage({
                     to: message.userid,
                     gotChatResponse: true
                 });
             }
             //if link reciever rejecting the incoming call
             document.getElementById('worng').onclick = function() {

 document.getElementById('incomingChat').style.display = 'none';
```

```
                thankYouNotyDiv.style.display =
                'none'; buttonDiv.style.display =
                'none'; textMessage.disabled = true;
                w5capi.sendCustomMessage({
                    to: message.userid,
                    rejectingTheChat: true
                });
            }
        }
        //when link sender got confirmation that other user is ready
to join his call
        //initiated call.
        if (message.gotChatResponse) {
            connectLocalNotyDiv.style.display =
            'none'; /*
             @ Old
            Syntax */
            //w5capi.call(window.params.callee);

            /*
             @ new
            Syntax */
            w5capi.call(window.params.callee,{data:true});
        }
        //link sneder got chat request from link
        reciever. if (message.readyForChat) {
            showchatbox();
            document.getElementById('incomingChat').style.display =
'block';
            incomingCall1 = document.getElementById('incomingChat');
            document.getElementById('right').onclick = function() {
                //if agree to recieve the chat request.

document.getElementById('incomingChat').style.display =
                'none'; w5capi.sendCustomMessage({
                    to: message.userid,
                    yesReadyForChat: true
                });
            }
            //if rejecting the chat request.
            document.getElementById('worng').onclick = function() {

document.getElementById('incomingChat').style.display =
                'none'; textMessage.disabled = true;
                w5capi.sendCustomMessage({
                    to: message.userid,
                    chatRejected: true
                });
            }
        }
        //The link reciever is intiating call to link
        sender if (message.yesReadyForChat) {
            connectNotyDiv.style.display =
            'none'; /*
             @ Old
            Syntax */
            //w5capi.call(window.params.callee);

            /*
             @ new
            Syntax */
            w5capi.call(window.params.callee,{data:true});
        }
        //link reciever got message that other user has rejected the
call.
        if (message.chatRejected) {
            var str = connectNotyDiv.innerHTML;
            var res = str.replace("Connecting.....", "Call
Rejected....");
            connectNotyDiv.innerHTML =
            res; setTimeout(function() {
                connectNotyDiv.style.display =
            'none'; }, 4000);
            textMessage.disabled = true;
        }
        //link sender got message that link reciever has
rejected the call.
        if (message.rejectingTheChat) {
            textMessage.disabled = true;
            var str = connectLocalNotyDiv.innerHTML;
            var res = str.replace("Connecting.....", "Call
Rejected....");
            connectLocalNotyDiv.innerHTML =
            res; setTimeout(function() {
                connectLocalNotyDiv.style.display =
            'none'; }, 4000);
        }
    })
```

```
       W5CAPI.datachannel = true;

 //This event will be handled when user will get messages
through data channel.
   W5CAPI.onMessage(function(data){
         appendDIV3(data.msg.message, data.msg.fullName);
   });
```

```
   //This event is handling when data channel is closed.
   W5CAPI.onDatachannel(function(data){
      If(data.event==’open’)
               {
            if (thankYouNotyDiv && buttonDiv) {
            thankYouNotyDiv.style.display = 'none';
            buttonDiv.style.display = 'none';
        }
        showchatbox(); document.getElementById('text-
        message').disabled = false; }else
        if(data.event==='close')
        {
            w5capi.channels.value = '';
        document.getElementById('text-message').disabled = true;
        }
     });
    //.......creating object for text area and message box.
    var textMessage = document.getElementById('text-message');
    var messagesBox = document.getElementById('messages-box1');

    //When user will type messages on the
    textarea. textMessage.onkeyup = function(e) {
         if (e.keyCode != 13) return;
         if (this.value.length==1)
         {
              var n = noty({
                text: 'There are no text to send, please type your
message..',
                type: 'information',
                dismissQueue: true,
                layout: 'topCenter',
                theme: 'defaultTheme',
                buttons: [{
                    addClass: 'btn btn-primary',
                    text: 'Ok',
                    onClick: function($noty)
                        { $noty.close();
                    }
                }, ]
            });
              this.value='';
         }else
{


              W5CAPI.send({
                   fullName: document.getElementById('full-
name').value,
                   message: this.value
              });
              appendDIV4(this.value);
              this.value = '';
              this.focus();
         }
    };

    //Appending local user's messages on the message-
    box function appendDIV4(text, fullName) {
         fullName= fullName || document.getElementById('full-
name').value;
         var namErr=/^[A-Za-z0-
         9_]{1,15}$/; if(fullName==""){
              var n = noty({
                text: 'Please enter your name and
                send..', type: 'information',
                dismissQueue: true,
                layout: 'topCenter',
                theme: 'defaultTheme',
                buttons: [{
                    addClass: 'btn btn-primary',
                    text: 'Ok',n($noty) {
                        $noty.close();
                    }
                }, ]
            });
              return ;
         }
         if(!namErr.test(fullName)){
              ar n = noty({
                text: 'Name must contain characters(a-z)/(A-Z)&
maximum upto 15...!',
                type: 'information',
                dismissQueue: true,
                layout: 'topCenter',
                theme: 'defaultTheme',
                buttons: [{
                    addClass: 'btn btn-primary',
                    text: 'Ok',
                    onClick: function($noty)
                        { $noty.close();
                    }
                }, ]
            });
              return ;
         }
         var currentTime=new w5desi();
         var y=currentTime.time;
         document.getElementById('innerdiv').innerHTML =
document.getElementById('innerdiv').innerHTML +'<div
class="bpaddingdiv1 wordbrealaddl"><div class="chatwindowname
fl"><div class="rpaddingdiv1"><div class="newallpaddingdiv1
textaligncenterdiv colorwhite fontssizediv5 bgcolordeepblue
```

```
borderradiusdiv1">'+fullName+'</div></div></div><div
class="chatwindowleft fr"><div id="demoright"
class="smallallpaddingdiv1 fontssizediv6 colorlightblueone"><div
class="smallbpaddingdiv1"><div class="fl smallfontssizediv
colorgray">'+currentDay+'</div><div class="fr smallfontssizediv
colorgray">'+y+'</div><div
class="clear"></div></div>'+text+'</div></div><div
class="clear"></div></div>';

    //For Autoscroll
          var
scroltop=parseInt(document.getElementById('innerdiv').clientHeight)-
document.getElementById('messages-box1').clientHeight;
          if(document.getElementById('innerdiv').clientHeight >
document.getElementById('messages-box1').clientHeight) {
                document.getElementById('messages-box1').scrollTop
=scroltop;
          }
    }

    //Appending remote user's message in the
    messagebox. function appendDIV3(text ,fullName) {
console.log('fullname first',
fullName); var fullName1= fullName;
          var namErr=/^[A-Za-z0-
          9_]{1,15}$/; if(fullName1==''){
                return;
          }
          if(!namErr.test(fullName1)){
                return ;
          }
          var currentTime=new w5desi();
          var y=currentTime.time;

          document.getElementById('innerdiv').innerHTML =
document.getElementById('innerdiv').innerHTML + '<div
class="bpaddingdiv1 wordbrealaddl"><div class="chatwindowleft
fl"><div id="demoleft" class="smallallpaddingdiv1 fontssizediv6
colorlightblueone"><div class="smallbpaddingdiv1"><div class="fl
smallfontssizediv colorgray">'+currentDay+'</div><div class="fr
smallfontssizediv colorgray">'+y+'</div><div
class="clear"></div></div>'+text+'</div></div><div
class="chatwindowname fr"><div class="lpaddingdiv1"><div
class="newallpaddingdiv1 textaligncenterdiv colorwhite fontssizediv5
bgcolordeepblue
borderradiusdiv1">'+fullName1+'</div></div></div><div
class="clear"></div></div>';
          var
scroltop=parseInt(document.getElementById('innerdiv').clientHeight)-
document.getElementById('messages-box1').clientHeight;
```

```
              if(document.getElementById('innerdiv').clientHeight >
document.getElementById('messages-box1').clientHeight ){
                    document.getElementById('messages-
box1').scrollTop =scroltop;
              }
              delete fullName1;
    }

   //Providing the current time and date when user send
   msgs function startDate() {
         var today = new Date();
         var d = today.getDate();
         var month=new Array();
         month[0]="January";
         month[1]="February";
         month[2]="March";
         month[3]="April";
         month[4]="May";
         month[5]="June";
         month[6]="July";
         month[7]="August";
         month[8]="September";
         month[9]="October";
         month[10]="November";
         month[11]="December";
         var m=month[today.getMonth()];
         var st=m.slice(0,3);
         var weekday=new Array(7);
         weekday[0]="Sunday";
         weekday[1]="Monday";
         weekday[2]="Tuesday";
         weekday[3]="Wednesday";
         weekday[4]="Thursday";
         weekday[5]="Friday";
         weekday[6]="Saturday";
         var day =weekday[today.getDay()];
         var day1=day.slice(0,3);
         var x=st + "," + d + ", -" + day1+"-";
         t = setTimeout(function () {
              startDate() },
         500);
         return x;
   }

   var currentDay= startDate();
   function w5desi(){
         this.time=getTime1();
         function getTime1() {
              var dTime = new Date();
              var hours = dTime.getHours();
```

```
               var minute = dTime.getMinutes();
               var period = "AM";
               if (hours > 12) {
                     period = "PM"
               }else{
                     period = "AM";
               }
               hours = ((hours > 12) ? hours - 12 : hours)
               var x= hours + "." + minute + period
               return x;
         }
    }

   //Notification when user click on send button for sending
messages
   document.getElementById('btn').onclick=function()
         { var value=textMessage.value;
         if(value.length==0)
         {
               var n = noty({
                 text: 'There are no text in the text-field to send,
please type your message..',
                 type: 'information',
                 dismissQueue: true,
                 layout: 'topCenter',
                 theme: 'defaultTheme',
                 buttons: [{
                     addClass: 'btn btn-primary',
                     text: 'Ok',
                     onClick: function($noty)
                          { $noty.close();
                     }
                 }, ]
             });
             this.value =
         ''; }else{
               W5CAPI.send({
                     fullName: document.getElementById('full-
name').value,
                     message: value
               });
               appendLocalMessages(textMessage.value);
               textMessage.value = '';
               textMessage.focus();
         }
   }

   // End the chat call
   var callclose = document.getElementById('closecall');
    callclose.onclick = function(){
```

```
location.reload(true);
        location.href = location.href.replace(location.search, '');
    }

    // Notification if the other user has disconnected.
    W5CAPI.on('user-left-all',function(id){
         if(id==window.params.callee+document.capikey){
              var n = noty({
                text: 'The user whose id is:' + id + ' has
disconnected...!!!',
                type: 'information',
                dismissQueue: true,
                layout: 'topCenter',
                theme: 'defaultTheme',
                buttons: [{
                    addClass: 'btn btn-primary',
                    text: 'Ok',
                    onClick: function($noty)
                        { $noty.close();
                        if (incomingCall1) {
                            incomingCall1.style.display = 'none';
                        }
                        if (incomingCall && thankYouNotyDiv &&
buttonDiv) {
                            incomingCall.style.display = 'none';
                            thankYouNotyDiv.style.display = 'none';
                            buttonDiv.style.display = 'none';
                        }
                        if (messagesBox) {

document.getElementById('innerdiv').innerHTML = "";
                        }
                        if (connectNotyDiv) {
                            connectNotyDiv.style.display = 'none';
                        }
                        if (connectLocalNotyDiv) {
                            connectLocalNotyDiv.style.display =
'none';
                        }
              location.href  =  location.href.replace(location.search,
'');
         }
    })
}
----------------------------------------
```

6.4 Implementing file transfer

The file transfer call is implemented in the mainfile.js file located at the *//W5CAPI-Demo/js/RTC*directory.

6.4.1 Initializing W5CAPI

This process constructs the w5peer object and initializes the basic functionality of rtc service with default parameters.

How we implemented:

```
var W5CAPI = new  W5Peer({key:document.capikey});
```

6.4.2 Assigning userid

A user-id is a unique number assigned to a user. Both the Caller and the Callee are assigned unique user ids during the registration.

How we implemented:

```
window.params = params;
window.params.caller=window.params.caller || W5CAPI.userid;
W5CAPI. registerUserWithId (window.params.caller);
window.params.callee=window.params.callee ||uid1;
```

6.4.3 Opening a signaling channel

This process opens signaling channel between the client and the W5RTC Signaling Server.

How we implemented:

```
W5CAPI.onSuccess(function(data) {
    setUp.disabled = false;
    setUp.className='textaligncenterdiv';
          }
});
```

6.4.4 Initiate a File transfer session

This process places a call to the remote user to establish a file transfer session.

How we implemented:

```
W5CAPI.call(window.params.callee, {data:true});
```

6.4.5 Initializing data channel

To send chat messages or transfer files on webrtc data channels we need to enable the data channel. It can work without media-streaming. It is used when users want to chat and share files or screen.

How we implemented:

```
W5CAPI.datachannel = true;
```

6.4.6 Handling file events

W5CAPI notifies the user with events on when file is received ,when file is process.

How we implemented:

```
//When user uploads a file this event is fired.
W5CAPI.onFile(function (file){
   If(file.event==’file-start’)
     {
      }
         // Add an inner html to display the shared file.
});

// When you sent a file; or when you receive a file, file-end is
fired.
W5CAPI.onFile(function (file){ if
(!progressHelper[file.uuid]) {
         console.error('No such progress-helper element exists.',
file);
         return;
}
if (file.type.indexOf('image') != -1)
      {
progressHelper[file.uuid].div.innerHTML ='<div class="bgcolorgray
allpaddingdiv1 colorlightblack headdingfonts fontssizediv4">' +
file.name + '</div><div class="bgcolorlightgray allpaddingdiv1"><img
src="' + file.url + '" title="' + file.name + '"></div>' + '<br/>';
}else{
   //Downloading the file
progressHelper[file.uuid].div.innerHTML = '<div class="bgcolorgray
allpaddingdiv1 colorlightblack headdingfonts
fontssizediv4">'+file.name+'<a href="' + file.url + '" download="' +
file.name + '"><div class="btn signinbtndivone download">'+"Download
file"+'</div></a></div>'+'<br/>';
}
});

// File transmission is on the way
W5CAPI.onFile(function (file) {
    var helper = progressHelper[file.uuid];
    if (!helper) return;
    helper.progress.value = file.currentPosition || file.maxChunks ||
helper.progress.max;
    updateLabel(helper.progress, helper.label);
});

//This event will handled when data channel will be open.
W5CAPI.onDatachannel(function(data) {
    showfilesharingbox();
    document.getElementById('uplod').style.display='block'
    ; document.getElementById('file').disabled=false;
});

//This event is handling when data channel is
    closed. W5CAPI.onDatachannel(function(event){
          document.getElementById('uplod').style.display='none';
     });
```

6.4.7 Sending file to Peer

W5CAPI provides simple API to send file to the peer; user needs to use the API to send file.

How we implemented:

```
//When user will select the file and send.
document.getElementById('file').onchange = function (){
          W5CAPI.sendFile(this.files[0]);
};
```

6.4.8 Handling call end event

When a user leaves the file transfer session W5CAPI fires an event to notify the peers with user-left-all event about the end of a session.

How we implemented:

```
// Notification if the other user has disconnected.
W5CAPI.on('user-left-all',function(id){
    if(id==window.params.callee+document.capikey){
          alert('The user whose id is:'+ id +' has
disconnected...!!!');
```

```
            location.href = location.href.replace(location.search, '');
    }
});
})
```

6.4.9 Full File Transfer code extract

```
-------------------------------------
function fileShare()
{
    var params = {},
    r = /([^&=]+)=?([^&]*)/g;

    function d(s) {
           return decodeURIComponent(s.replace(/\+/g, ' '));
    }
    var match, search = window.location.search.toLowerCase();
    while (match = r.exec(search.substring(1)))
    params[d(match[1])] = d(match[2]);
    // User is creating instance of W5peer class for accessing the
required methods/functions defined in that particular class.
    var W5CAPI = new  W5Peer({key:document.capikey});

     //Generate random string
function getRandomString() {
           return (Math.random() * new
Date().getTime()).toString(36).replace( /\./g , '');
     }
var uid3=getRandomString();

     //Get the caller and callee
     ids window.params = params;
     window.params.caller= window.params.caller || W5CAPI.userid;
     window.params.callee= window.params.callee ||uid3;
     W5CAPI. registerUserWithId (window.params.caller);

    w5capiReady(function(){
         w5capi.registerUserWithId(window.params.caller);
     });

    //For generating url for remote side
    document.getElementById('user_name4').value =
location.href.replace(location.search, '') + '?caller=' +
window.params.callee + '&callee=' +window.params.caller;

     var setUp = document.getElementById('starttalkdtndiv');
 w5capi.onCall(function(data) {
        w5capi.accept(data);
    });


   //Open socket connection by calling openSignalingChannel
   function W5CAPI.onSuccess(function(data) {
              setUp.disabled = false;
              setUp.className='textaligncenterdiv';
         }
   });

w5capi.sendCustomMessage({
         enableButton:true,
         to:window.params.callee+document.capikey
   });
//.......some global variables declared...............
var thankYouDiv;
var buttonDiv;
var connectNotyDiv;
var connectLocalNotyDiv;
var incomingCall1;
var incomingCall;
//........................end........................
//........custom-signaling-message function defined for handling
user's some custom message...............
w5capi.on('custom-signaling-message',function(message)
          {
   console.log('custom signaling message',message);
   if(message.to!=w5capi.userid)return;
   if(message.enableButton){
              //when link sender initiated call to link
            reciever setUp.onclick = function ()
            {
            showfilesharingbox();
            this.disabled = true;
            var div2=document.createElement('div');
                       div2.id='connectid';
                       div2.style.width='400px';
                       div2.style.height='30px';
                       div2.style.position='absolute';
                       div2.style.fontSize ='18px';
                       div2.style.marginTop='80px';
                       div2.style.marginLeft='100px';
                       div2.innerHTML='<font
color="black">'+'<b>'+"Connecting....."+'</b>'+'</font>';

document.getElementById('writeText').appendChild(div2);
                       connectLocalNotyDiv=div2;

document.getElementById('uplod').style.display='none';
```

```
             w5capi.sendCustomMessage({
               to:window.params.callee+document.capikey,
               startFileShare:true
           });
           }
         w5capi.sendCustomMessage({
              enableButton1:true,
              to:message.userid
         });
   }
   //Remote client got msg from call
   initiator if(message.enableButton1)
        {
         showfilesharingbox();
          var div=document.createElement('div');
         div.id='nameid';
         div.style.width='300px';
         div.style.height='30px';
         div.style.position='absolute';
         div.style.marginTop='20px';
         div.style.marginLeft='70px';
         div.style.fontSize='16px';
          div.innerHTML='<font color="black">'+'<b>'+"Thank You for
joining...!"+'</b>'+'</font>';
          document.getElementById('writeText').appendChild(div);
          thankYouDiv=div;
         var div1=document.createElement('div');
        div1.innerHTML='<div class="btn signinbtndivone">'+"Start
Transfer"+'</div>';
        div1.style.marginTop='50px';
        div1.style.marginLeft='120px';
        document.getElementById('writeText').appendChild(div1)
        ;
        document.getElementById('uplod').style.display='none';
        var div2=document.createElement('div');
                    div2.id='connectid';
                    div2.style.width='400px';
               div2.style.height='30px';
               div2.style.position='absolute';
               div2.style.marginTop='80px';
               div2.style.marginLeft='100px';
                 div2.style.fontSize ='18px';
                 div2.style.display='none';
                    div2.innerHTML='<font
color="black">'+'<b>'+"Connecting....."+'</b>'+'</font>';

   document.getElementById('writeText').appendChild(div2)
         ; buttonDiv=div1;
         connectNotyDiv=div2;
            buttonDiv.onclick=function()
                {
```

```
              connectNotyDiv.style.display = 'block';
              thankYouDiv.style.display='none';
              buttonDiv.style.display='none';
              document.getElementById('uplod').style.display='none';
              w5capi.sendCustomMessage({
                    to:window.params.callee+document.capikey,
                    readyToRecieveFile:true
              });
         }
   }
   //link reciever got incoming call request from link
   sender if(message.startFileShare)
                {
         thankYouDiv.style.display='none';
         buttonDiv.style.display='none';

   document.getElementById('incomingFile').style.display='block';
          incomingCall=document.getElementById('incomingFile');
                            // if link reciever accepting incoming
call
         document.getElementById('right').onclick=function(){
         document.getElementById('uplod').style.display='block'
         ;

   document.getElementById('incomingFile').style.display='none';
              w5capi.sendCustomMessage({
                    to:message.userid,
                    gotFileInfo:true
              });
         }
               //if link reciever rejecting the incoming call
                document.getElementById('wrong').onclick=function()
                    {

   document.getElementById('incomingFile').style.display='none';
              w5capi.sendCustomMessage({
                    to:message.userid,
                    fileRejected:true
              });
         }

   }
   //when link sender got confirmation that other user is ready to
join his call
   //initiated call.
   if(message.gotFileInfo)
    {
      console.error('start file');
         document.getElementById('uplod').style.display='block'
         ; connectLocalNotyDiv.style.display='none';
        /**
        @ Old
        Syntax */
        //w5capi.call(window.params.callee);
        /**
        @ New Syntax
        */
         w5capi.call(window.params.callee,{data:true});
   }
    //link sneder got request from link reciever.
   if(message.readyToRecieveFile)
        {
         showfilesharingbox();

   document.getElementById('incomingFile').style.display='block';
         incomingCall1=document.getElementById('incomingFile');
         document.getElementById('uplod').style.display='none';
                   //if agree to recieve the request.
         document.getElementById('right').onclick=function(){
         document.getElementById('uplod').style.display='block'
         ;

   document.getElementById('incomingFile').style.display='none';
             w5capi.sendCustomMessage({
                   to:message.userid,
                   yesIamReady:true
             });
         }
          //if rejecting the request.
         document.getElementById('wrong').onclick=function()
        {

   document.getElementById('incomingFile').style.display='none';
             w5capi.sendCustomMessage({
                   to:message.userid,
                   wantToRejectedFileSharing:true
             });
         }
   }
  //The link reciever is intiating call to link sender
   if(message.yesIamReady)
        {
         document.getElementById('uplod').style.display='block'
         ; connectNotyDiv.style.display='none';

/**
        @ Old
        Syntax */
        //w5capi.call(window.params.callee);
        /**
        @ New Syntax
        */
```

```
           w5capi.call(window.params.callee,{data:true});
    }
    //link reciever got message that other user has rejected the call.
    if(message.wantToRejectedFileSharing)
          {
           var str=connectNotyDiv.innerHTML;
           var res=str.replace("Connecting.....","Call
           Rejected...."); connectNotyDiv.innerHTML=res;
           setTimeout(function(){
                  connectNotyDiv.style.display='none'; },
                  4000);
    }
     //link sender got message that link reciever has rejected the
call.
    if(message.fileRejected)
          {
           var str=connectLocalNotyDiv.innerHTML;
           var res=str.replace("Connecting.....","Call
           Rejected...."); connectLocalNotyDiv.innerHTML=res;
           setTimeout(function(){
                  connectLocalNotyDiv.style.display='none'; },
                  4000);
    }
     })

     // To  customize  chunk-size;  defualt  is
    15k W5CAPI.chunkSize = 40*1000; // 12k

     // To  customize  interval  size;  defualt  is
    500ms W5CAPI.chunkInterval = 100;

    //This event will handled when datachannel will be open.
    W5CAPI. onDatachannel (function(data) {
if(data.event==='open')
           document.getElementById('file').disabled=false;
showfilesharingbox();
    });

    var progressHelper =
    {}; var storeDiv;
    var messagesBox = document.getElementById('messages-
    box2'); w5capi.onFile(function(data){

           if(data.event==='file-start')
           {
                var file=data.file;
                var div = document.createElement('div');
          div.title = file.name;
```

```
        div.innerHTML = '<div class="bgcolorgray allpaddingdiv1
colorlightblack headdingfonts fontssizediv4">' + file.name +
'</div><div class="bgcolorlightgray
allpaddingdiv1"><label>0%</label> <progress></progress></div>';
        messagesBox.appendChild(div);
                    progressHelper[file.uuid] =
                                {
            div: div,
            progress: div.querySelector('progress'),
            label: div.querySelector('label')
        };
        progressHelper[file.uuid].progress.max =
         file.maxChunks; }else if(data.event==='file-progress')
        {
            console.error('file progress');

                  var file=data;
                  var helper = progressHelper[file.uuid];
            console.log(file);
        if (!helper) return;
        helper.progress.value = file.currentPosition
|| file.maxChunks || helper.progress.max;
        updateLabel(helper.progress, helper.label);
        }else if(data.event==='file-end')
     {
         var file=data.file;
         if (!progressHelper[file.uuid]) {
            console.error('No such progress-helper element exists.',
file);
            return;
        }
if (file.type.indexOf('image') != -1)
            {
                   progressHelper[file.uuid].div.innerHTML ='<div
class="bgcolorgray allpaddingdiv1 colorlightblack headdingfonts
fontssizediv4">' + file.name + '</div><div class="bgcolorlightgray
allpaddingdiv1"><img src="' + file.url + '" title="' + file.name +
'"></div>' + '<br/>';
}else{
   //Downloading the file
                   progressHelper[file.uuid].div.innerHTML = '<div
class="bgcolorgray allpaddingdiv1 colorlightblack headdingfonts
fontssizediv4">'+file.name+'<a href="' + file.url + '" download="' +
file.name + '"><div class="btn signinbtndivone download">'+"Download
file"+'</div></a></div>'+'<br/>';
              }
    });
```

```
   // To display percentage for sending or receiving instance
function updateLabel(progress, label) {
```

```
         if (progress.position == -1) return;
         var position = +progress.position.toFixed(2).split('.')[1]
|| 100;
         label.innerHTML = position + '%';
    }

    //When user will select the file and send.
    document.getElementById('file').onchange = function (){
         W5CAPI.sendFile(this.files[0]);
    };

   //This event is handling when datachannel is closed.
    W5CAPI.on(‘datachannel-close’,function(event){
         document.getElementById('uplod').style.display='none';
     });

  //When user will click on end share button
    var callclose = document.getElementById('closecall');
    callclose.onclick = function(){
location.reload(true);
        location.href = location.href.replace(location.search, '');
    }

    //When other user has disconnected.
    W5CAPI.on('user-left-all',function(id){
         if(id==window.params.callee+document.capikey){

         var n = noty({
            text        : 'The user whose id is:'+ id +' has
disconnected...!!!',
            type            : 'information',
            dismissQueue: true,
            layout      : 'topCenter',
            theme       : 'defaultTheme',
            buttons     : [
                    {addClass: 'btn btn-primary', text: 'Ok',
                            onClick: function ($noty) {
                    $noty.close();
                        if(incomingCall1)
                        {
                               incomingCall1.style.display='none';
                         }
                        if(incomingCall && thankYouDiv && buttonDiv)
                        {
                               incomingCall.style.display='none';
                               thankYouDiv.style.display='none';
                               buttonDiv.style.display='none';
                         }
                        if(connectNotyDiv)
                        {
                               connectNotyDiv.style.display='none';
                          }
                         if(connectLocalNotyDiv)
                         {

   connectLocalNotyDiv.style.display='none';
                          }
location.href = location.href.replace(location.search, '');
         }
   })
}
----------------------------------
```

6.5 Implementing screen share

The screen share call is implemented in the mainscreen.js file located at the *//W5CAPI-Demo/js/RTC*directory.

6.5.1 Initializing W5CAPI

This process constructs the w5peer object and initializes the basic functionality of rtc service with default parameters.

How we implemented:

```
var W5CAPI = new  W5Peer({key:document.capikey});
```

6.5.2 Assigning userid

A user-id is a unique number assigned to a user. Both the Caller and the Callee are assigned unique user ids during the registration.

How we implemented:

```
window.params = params;
window.params.caller=window.params.caller || W5CAPI.userid;
W5CAPI. registerUserWithId(window.params.caller);
window.params.callee=window.params.callee ||uid1;
```

6.5.3 Setting media type.

The media type parameter is set to screen; this enables the application to share the screen across the peers. If user needs audio, video and screen together he/she can enable all the three parameters. Apart from media type we have also adjusted the resolution and bandwidth to maintain the quality of the screen display.

How we implemented:

```
// Describe the screen resolution needed for Screen
    Sharing W5CAPI.resolutions = {
minWidth:1280,
minHeight:720,
maxWidth:1280,
maxHeight:720,
minAspectRatio:1.77
, minFrameRate: 30
   };

// Describes the bandwidth of the media
W5CAPI.bandwidth = {video: 300};   // 300kbs is needed for screen

// Mention the direction, since only one user can share screen at a
time.
    W5CAPI.direction = 'one-way';
```

6.5.4 Initialize Stream Events

Media stream refers to the stream of data that are call backed from the media sources i.e. microphone and camera. There are of two types of streams—local and remote. The media streams needs to be captured and displayed in media player. The media streams are attached with event to notify the application about the stream type and intended action.

How we implemented:

```
// Append the local media stream
W5CAPI. onLocalStream (function(stream) {
   appendMediaElement(stream, 'local');
});


// Append remote media stream
W5CAPI. onRemoteStream (function(stream)
   { showscreensharingbox();
   appendMediaElement(stream, 'remote');
});;
```

6.5.5 Opening a signaling channel

This process opens signaling channel between the client and the W5RTC Signaling Server.

How we implemented:

```
W5CAPI. onSuccess (function(data) {
    setUp.disabled = false;
    setUp.className='textaligncenterdiv';
          }
});
```

6.5.6 Placing a call

This process places a call to the remote user to start a screen sharing session.

How we implemented:

```
W5CAPI.call(window.params.callee, {screen:true,direction:'one-way'});
```

6.5.7 Handling call end event

When a user leaves the session W5CAPI fires an event to notify the peers with user-left-all event about the end of a session.

How we implemented:

```
// Notification if the other user has disconnected.
W5CAPI.on('user-left-all',function(id){
    if(id==window.params.callee){
          alert('The user whose id is:'+ id +' has 
disconnected...!!!');
          location.href = location.href.replace(location.search, '');
    }
});
```

6.5.8 Full Screen Share code extract

```
---------------------------------
// Screen Share DEMO functionality
function shareScreen()
{
    var params = {},
     r = /([^&=]+)=?([^&]*)/g;
    function d(s) {
        return decodeURIComponent(s.replace(/\+/g, ' '));
    }
    var match, search = window.location.search.toLowerCase();
    while (match = r.exec(search.substring(1)))
          params[d(match[1])] = d(match[2]);
    //User is creating instance of W5peer class for acessing the
required methods/functions defined in that particular class.
    var W5CAPI = new  W5Peer({key:document.capikey});

    // Generate Random string
    function getRandomString() {
    return (Math.random() * new
Date().getTime()).toString(36).replace( /\./g , '');
    }
    var uid1= getRandomString();

    // Check if browser is Mozilla Firefox
    var isFirefox = !!
    navigator.mozGetUserMedia; if(isFirefox){
          alert('Screen sharing in WEBRTC is not supported in
firefox,please run it in google chrome')
          return;
    }
    window.params = params;
    window.params.caller= window.params.caller || W5CAPI.userid;
    window.params.callee= window.params.callee ||uid1;
    W5CAPI. registerUserWithId(window.params.caller);

 w5capiReady(function(){
         w5capi.registerUserWithId(window.params.caller);
     });


     //set extention ID
   w5capi.extensionID('nlnfogneblaieieknkhaabhhendjdanf');
   /*
     @ New
     Syntax */
     //accept incomming call
     w5capi.onCall(function(data) {

         w5capi.accept(data);
     });

    // Check if browser is Internet
    Explorer function isIE () {
          var myNav = navigator.userAgent.toLowerCase();
          return (myNav.indexOf('msie') != -1) ?
parseInt(myNav.split('msie')[1]) : false;
```

```
   // If internet explorer then notify the user that some features
will not work
   if (isIE()) {
        var n = noty({
        text : 'Internet explorer does not support
Webrtc...please test this W5CAPI Screensharing demo only in
chrome!!!',
        type : 'information',
        dismissQueue: true,
        layout : 'topCenter', theme
        : 'defaultTheme', buttons :
        [{
                        addClass: 'btn btn-primary', text:
                        'Ok', onClick: function ($noty) {
                        $noty.close();
                        }
                   }]
        });
        return;
   }
   // Check if Browser is Safari then notify the user that some
features will not work
   var isSafari=navigator.userAgent.search("Safari") >= 0 &&
navigator.userAgent.search("Chrome") < 0;
   if (isSafari)
   {
        alert("You are using Safari browser");
    var n = noty({
           text         : 'Safari does not support Webrtc..please
test this w5capi Screensharing demo only in chrome!!!',
           type            : 'information',
           dismissQueue: true,
           layout       : 'topCenter',
           theme        : 'defaultTheme',
             buttons      : [{addClass: 'btn btn-primary', text:
'Ok',
                                    onClick: function ($noty) {
                    $noty.close();
                 }
                              }]
       });        return;
   }

   document.getElementById('user_name6').value
=location.href.replace(location.search, '') + '?caller=' +
window.params.callee + '&callee=' +window.params.caller;


   // Describe the screen resolution needed for Screen Sharing
```

```
    W5CAPI.resolutions = {
minWidth:1280,
minHeight:720,
maxWidth:1280,
maxHeight:720,
         minAspectRatio:1.77
         , minFrameRate: 30
    };

   // Describes the bandwidth of the media
      W5CAPI.bandwidth = {video: 300};   // 300kbs is needed for screen

    // Mention the direction
    //W5CAPI.direction = 'one-way';

   // Append the local media stream
W5CAPI. onLocalStream (function(stream)
{ appendMediaElement(stream, 'local');
    });

   // Append remote media stream
W5CAPI. onRemoteStream (function(stream)
{ appendMediaElement(stream, 'remote');
    });

    // Describes the Media Element Controls
function appendMediaElement(stream) {
var mediaElement =
document.createElement(stream.getVideoTracks().length ? 'video'
: 'audio');
        mediaElement.src = URL.createObjectURL(stream);
        mediaElement.controls =false;
mediaElement.play();
document.querySelector('#writeText').appendChild(mediaElement);
   }
var setUp = document.getElementById('starttalkdtndiv');

   // Setup Socket Connection
   W5CAPI.onSuccess(function(data) {
            setUp.disabled = false;
              setUp.className='textaligncenterdiv';

   w5capi.sendCustomMessage({
        enableButton:true,
        to:window.params.callee+document.capikey
   });
        });


      //  Some Global variables declared
```

```
         var thankYouNotyDiv;
         var buttonDiv;
         var connectNotyDiv;
         var connectLocalNotyDiv;
         var incomingCall1;
         var incomingCall;
      //........custom-signaling-message function defined for
handling user's some custom message...............
   w5capi.on('custom-signaling-message',function(message){
   console.log('custom signaling message',message);
   if(message.to!=w5capi.userid)return;
   if(message.enableButton){
         w5capi.sendCustomMessage({
              enableButton1:true,
              to:message.userid
         });
   }
              //Remote client got msg from call initiator
   if(message.enableButton1){
         showscreensharingbox();
          var div=document.createElement('div');
          div.id='nameid';
              div.style.width='100%';
          div.style.height='30px';
          div.style.position='absolute';
         div.style.marginTop='100px';
              div.style.textAlign='center';
          div.style.fontSize='16px';
          div.innerHTML='<font color="black">'+'<b>'+"Thank You for
joining...!"+'</b>'+'</font>';
          document.getElementById('writeText').appendChild(div);
          thankYouNotyDiv=div;
         var div1=document.createElement('div');
        div1.innerHTML='<div class="btn signinbtndivone">'+"Screen
share"+'</div>';
        div1.style.marginTop='120px';
              div1.style.textAlign='center';
            div1.style.width='100%';
            div1.style.position='absolute';
        document.getElementById('writeText').appendChild(div1)
        ; var div2=document.createElement('div');
                   div2.id='connectid';
                   div2.style.width='400px';
               div2.style.height='30px';
               div2.style.position='absolute';
               div2.style.marginTop='80px';
              div2.style.marginLeft='350px';
                 div2.style.fontSize ='18px';
                 div2.style.display='none';
```

```
                        div2.innerHTML='<font
color="black">'+'<b>'+"Connecting....."+'</b>'+'</font>';

    document.getElementById('writeText').appendChild(div2)
           ; buttonDiv=div1;
           connectNotyDiv=div2;
           buttonDiv.onclick=function(){
                  connectNotyDiv.style.display = 'block';
                  thankYouNotyDiv.style.display='none';
                  buttonDiv.style.display='none';

                  if (sessionStorage.w5rtc_capiext)
            { w5capi.sendCustomMessage({
                         to:window.params.callee+document.capikey,
                         readyForScreenShare:true
                  });
}else
{
    console.error('poda');
var installing=true;
chrome.webstore.install('https://chrome.google.com/webstore/detail/n
lnfogneblaieieknkhaabhhendjdanf',
                         function (arg) {
                                setTimeout(function(){
                                  location.reload();
                                   w5capi.sendCustomMessage({
                                to:window.params.callee+document.capikey,
                                  readyForScreenShare:true
                               });
                                  },4000);
                         }, function (err) {
                              console.error(err)
                         });
}
           }
    }

            //link reciever got incoming call request from link
    sender if(message.isThisScreen){
            thankYouNotyDiv.style.display='none';
            buttonDiv.style.display='none';

    document.getElementById('incomingScreen').style.display='block';
           incomingCall=document.getElementById('incomingScreen');
                   // if link reciever accepting incoming call
           document.getElementById('right').onclick=function(){

    document.getElementById('incomingScreen').style.display='none';
                  w5capi.sendCustomMessage({
                         to:message.userid,
                    yesThisIsScreen:true
              });
         }
              //if link reciever rejecting the incoming call
         document.getElementById('wrong').onclick=function(){

   document.getElementById('incomingScreen').style.display='none';
              w5capi.sendCustomMessage({
                    to:message.userid,
                    rejectingScreensharing:true
              });
         }
   }
         //when link sender got confirmation that other user
is ready to join his call
      //initiated call.
   if(message.yesThisIsScreen){
         connectLocalNotyDiv.style.display='none';
        /**
         @Old
         Syntax */
         //w5capi.call(window.params.callee);
        /**
         @ New
         Syntax */

   w5capi.call(window.params.callee,{screen:true,direction:'one-
way'});
   }
              //link sneder got call request from link
   reciever. if(message.readyForScreenShare){
         showscreensharingbox();

   document.getElementById('incomingScreen').style.display='block'
       ; incomingCall1=document.getElementById('incomingScreen');
              //if agree to recieve the call request.
         document.getElementById('right').onclick=function(){

   document.getElementById('incomingScreen').style.display='none';
              w5capi.sendCustomMessage({
                    to:message.userid,
                    yesIamReady:true
              });
         }
                 //if rejecting the call request.
         document.getElementById('wrong').onclick=function(){

   document.getElementById('incomingScreen').style.display='none';
              w5capi.sendCustomMessage({
                    to:message.userid,
       wantToRejectScreen:true
              });
         }
    }
         //The link reciever is intiating call to link
    sender if(message.yesIamReady){
         connectNotyDiv.style.display='none';
          /**
         @Old
         Syntax */
         //w5capi.call(window.params.callee);
        /**
         @ New
         Syntax */

    w5capi.call(window.params.callee,{screen:true,direction:'one-
way'});
    }

         //link reciever got message that other user has rejected
the call.
    if(message.wantToRejectScreen){
         var str=connectNotyDiv.innerHTML;
         var res=str.replace("Connecting.....","Call
         Rejected...."); connectNotyDiv.innerHTML=res;
         setTimeout(function(){
              connectNotyDiv.style.display='none'; },
              4000);
    }
          //link sender got message that link reciever has rejected
the call.
    if(message.rejectingScreensharing)
    {
         var str=connectLocalNotyDiv.innerHTML;
         var res=str.replace("Connecting.....","Call
         Rejected...."); connectLocalNotyDiv.innerHTML=res;
         setTimeout(function(){
              connectLocalNotyDiv.style.display='none'; },
              4000);
    }
    })
    // Start the Call
         setUp.onclick = function ()
    { showscreensharingbox();
    this.disabled = true;
    var div2=document.createElement('div');
                   div2.id='connectid';
                   div2.style.width='400px'
                   ;
               div2.style.height='30px';
               div2.style.position='absolute';
               div2.style.marginTop='80px';
```

```
                 div2.style.fontSize ='18px';
              div2.style.marginLeft='350px';
                 div2.innerHTML='<font
color="black">'+'<b>'+"Connecting....."+'</b>'+'</font>';

   document.getElementById('writeText').appendChild(div2)
                    ; connectLocalNotyDiv=div2;
                    //when link sender initiated call request to link
reciever
  w5capi.sendCustomMessage({
     to:window.params.callee+document.capikey,
     isThisScreen:true
  })
         }


   // End the call
   var endShare=document.getElementById('callClose');
   endShare.onclick=function(){
         location.reload(true);
         location.href = location.href.replace(location.search, '');
   }
   W5CAPI.on('user-left-all',function(id){
         if(id==window.params.callee+document.capikey){
              var n = noty({
            text         : 'The user whose id is:'+ id +' has
disconnected...!!!',
            type             : 'information',
            dismissQueue: true,
            layout       : 'topCenter',
            theme        : 'defaultTheme',
                    buttons     : [{addClass: 'btn btn-primary',
text: 'Ok',
                                    onClick: function ($noty) {
                    $noty.close();
                         if(incomingCall1){
                              incomingCall1.style.display='none';
                         }
                         if(incomingCall && thankYouNotyDiv &&
buttonDiv){
                              incomingCall.style.display='none';
                              thankYouNotyDiv.style.display='none';
                              buttonDiv.style.display='none';
                         }
                         if(connectNotyDiv){
                              connectNotyDiv.style.display='none';
                         }
                         if(connectLocalNotyDiv){

   connectLocalNotyDiv.style.display='none';
```

```
                         }

              location.href = location.href.replace(location.search,
'');
         }
   });

  }
--------------------------------------
```

7. APIs Details for advanced developers

W5Peer()	Constructor

Method to construct a w5rtc user object. It takes the API-Key as an argument, which is validated by the server. On successful validation a user object is constructed. It contains the functions and members to establish an RTC communication.

Return

W5Peer object.

Argument

API-Key

Usage

Var W5CAPIuser = new W5Peer({key:'22f55f4a79aaafca72248c7f8f7d3adb80e07d66eb091f3f});

Note: On successful user object construction, user can use the objects and attributes to change the RTC parameters.

onSuccess(callback)	API

This API initializes the signaling gateway and opens a socket connection between the web client application and the server application. The signaling parameters or messages are sent on this channel to the peer. W5CAPI uses proprietary signaling message to establish the real time communication.

This API takes a callback function as an argument. The users can implement their own application logic and add callback.

Return

W5Peer object.

Argument

API-Key

Usage

Var W5CAPIuser = new W5Peer(API-Key); //construct W5Peer object

w5rtcUser.onSuccess(function() {callButton.disabled = false;}});

call('target-userid') API

This API initiates a call to another user. It takes the target-user-id as an argument to initiate the call. The target user id is the user id assigned to the w5rtcUser object.

Note: Once the signaling channel is established, user is able to make call to the desired peer.

Return

Null

Argument

Null

Usage

Var W5CAPIuser = new W5Peer(API-Key); //construct W5Peer object

W5CAPIuser.registerUserWithId(caller)

w5rtcUser.onSuccess(function() {callButton.disabled = false;}});

w5rtcUser.call('callee');

stopCall() API

Disconnects the call.

Return

Null

Argument

Null

Usage

Var W5CAPIuser = new W5Peer(API-Key); //construct W5Peer object

W5CAPIuser.registerUserWithId('caller');

W5CAPIuser.onSuccess(function() {callButton.disabled = false;}});

W5CAPIuser.call('callee');

W5CAPIuser.stopCall();

muteAudio () API

Disables the local microphone.

Return

Null

Argument

Null

Usage

W5CAPIuser. muteAudio (); // Mute the Audio track

unmuteAudio () API

Enables the local microphone.

Return

Null

Argument

Null

Usage

W5CAPIuser. un*muteAudio ();*

pauseVideo ()	API

Mute video track of the stream. Video is muted on local and remotes sides.

Return

Null

Argument

Null

Usage

W5CAPIuser. pauseVideo *();*

playVideo ()	API

Play video track of the stream. Video is played on local and remotes sides.

Return

Null

Argument

Null

Usage

W5CAPIuser. playVideo *();*

setResolutions(width,height); API

This method overrides resolutions object to set custom video resolutions. The supported values are: 1920:1080, 1280:720, 960:720, 640:360, 640:480, 320:240, and 320:180

Return

Null

Argument

width: numeric,

height: numeric,

Usage

W5CAPIuser.setResolutions(1280, 720);

signal(string); API

This API is used to send data directly to the peers using the signaling gateway.

Return

Null

Argument

string: string data, send message to the peer through signaling gateway.

Usage

```
W5CAPIuser.signal({
            to: message.from,
            peerid: message.peerid,
            ready: true
            });
```

send(data); API

This API allows user to send longest possible string to the peer. This method also allows you to send any JavaScript object.

This object will be the "data" passed over "***datachannel-message***" event.

The ***datachannel-message*** event id fired as soon as peer receives data.

Return

Null

Argument

data: longest-possible-string

Usage

```
W5CAPIuser.send({
    message: '----',
    sender: 'userid',
    sendingTime: '----',
    sendingLanguage: 'en-US'
});
```

sendFile(file); API

This API allows user to send file to the peer. User is allowed to send more than one file concurrently.

The send file method is associated with file progress helper events ***file-start***, ***file-end*** and ***file-progress.*** These events are used to notify the file sharing/sending progress.

Return

Null

Argument

file: selected file that user needs to send to the peer.

Usage

```
document.querySelector('input[type=file]').onchange = function() {
    W5CAPIuser.sendFile(this.files[0]);
};
```

send(var); API

"send" method provides a common style to send both file and text messages. This API provides the flexibility to use single API instead of sendText and sendFile.

Return

Null

Argument

var: string/file/object.

Usage

```
W5CAPIuser.send('longest-possible-string');
W5CAPIuser.send(file);
W5CAPIuser.send(data);
```

setStatus(status) API

This API allows user to send status to other user's.

The setStatus method is associated with status helper events ***user-status***, ***online-users*** and ***offline-user.*** These events are used to notify the user status progress.

on(event, callback) API

This API is used to register a callback function for an event. The function is executed when the event is fired by the application.

Return

Null

Argument

Event: W5CAPI events.

callback: function that should perform certain task on receiving the event

Usage

```
W5CAPIuser. onLocalStream (, function (event) {
   // event.stream ---- MediaStream object
   // event.mediaElement ---- HTMLAudioElement or HTMLVideoElement
});
```

saveToDisk(fileurl, filename) API

Developers can use this API to save DataURL or fileURL to disk using "save-as" dialog.

Return

Null

Argument

fileurl : string.

filename : string.

Usage

onLocalStream Event

This event is always fired while capturing local media sources.

Usage

```
W5CAPIuser. onLocalStream ( function (event) {
    // event.stream ---- MediaStream object
    // event.mediaElement ---- HTMLAudioElement or HTMLVideoElement
});
```

onRemotestream Event

This event is always fired as soon as remote stream is delivered to remote peer.

Usage

```
W5CAPIuser.onRemotestream( function (event) {
    // event.stream ---- MediaStream object
    // event.mediaElement ---- HTMLAudioElement or HTMLVideoElement
});
```

datachannel-open Event

This event is fired as soon as data connection is opened and makes the user ready to send and receive data.

This event is fired repeatedly for each peer. So if you're interconnecting 10 users, this event will be fired 9-times for each user.

Usage

```
W5CAPIuser. onDatachannel ( function (event) {
      If(event.event=='open')
{
console.log("data channel is opend")
}


});
```

datachannel-message Event

This event is fired when a peer receives data (string/object/file) from other peer.

Usage

```
W5CAPIuser. onMessage ( function(data) {
   console.log(‘ received data’ , data);
});
```

datachannel-close Event

This event is fired as soon as data connection is closed

Usage

```
W5CAPIuser.onDatachannel( function(data) {
   If(event.event==’close’)
{
console.log(“data channel is close”)
}
});
```

file-start Event

This event is fired when user selects a file to be uploaded and uploads the file .

Usage

```
W5CAPIuser.onFile( function (file){
If(file.event==’file-start’
// Add an inner html to display the shared file.
        }
        });
```

file-end Event

This event is fired when a file is sent form peer end or when a file is receives and a peers end.

Usage

```
 W5CAPIuser.onFile( function (file){
If(file.event=='file-end')
{
//Download the file
}
}
```

file-progress Event

This event is fired to notify that the File transmission is on progress.

Usage

```
W5CAPIuser.onFile( function (file) {
If(file.event=='file-progress')
{
//Display the file transmission progress.
}
}
```

user-left-all Event

This event is fired when a user leaves a session (Audio/Video/Data/Screen)

Usage

```
W5CAPIuser.on('user-left-all', function (id) {
        console.log ('The user whose id is:'+ id +' has disconnected...!!!');
}
```

stream-removed Event

This event is fired when media stream is stopped.

Usage

```
W5CAPIuser.on('stream-removed', function (e) {
        //console.log ('media stream stopped need to remove the stream');
}
```

user-status Event

This event is fired when user change status.

Usage

```
W5CAPIuser.on('user-status',function(data){
//console.log(‘Username:’+data.from+’Status:’+data.status);
});
```

online-users Event

This event is fired when user come online.

Usage

```
W5CAPIuser.on('online-users',function(data){
//console.log(‘Online user’s list’+data.id);});
```

offline-user Event

This event is fired when user goes offline.

Usage

```
W5CAPIuser.on('offline-user',function(data){

//console.log('offline user id'+data.from);

});
```

peer-stats Event

This event is fired when peer connection established and this event emit peer connection stats.

Usage

```
W5CAPIuser.on('peer-stats',function(data){

     //console.log('bandwidth'+data.bandwidth);

    // console.log('bandwidthused'+data.useBandwidth);

   //console.log('localAddress'+data.localAddress);

   //console.log('remoteAddress'+data.remoteAddress);

    });
```

userid String Attribute

Sets user-id for the user. The id (Alpha Numeric value) is unique within the Application or API-key domain. By default, W5CAPI auto generates a random number for the id. However, you can override it to use custom values.

Return

Null

Argument

Null

Usage

Var W5CAPIuser = new W5Peer(API-Key);

W5CAPIuser.registerUserWithId('caller1234');

mediatype	Array Attribute

Set audio-video transmission by the local user in any subsequent calls. This array contains three values audio, video and screen. When audio is true the application transmits audio stream only, similar works when the video is true. When both the attributes are true, the application streams both audio and video packets. When screen is true the application shares the screen among the peers.

Return

Null

Argument

Null

Usage

var w5rtcUser = new W5Peer(API-Key);

w5rtcUser.call('callee',{audio:true,video:true});

Note: This is override parameter, when user constructs the W5Per object, the default value set for audio and video is true.

bandwidth Array Attribute

Set bandwidth for the data transmission in any subsequent calls. This array contains two values audio, video. Assigning the value overrides the default values set for the bandwidth. The unit taken by the values is ***kbps.***

Return

Null

Argument

Null

Usage

```
var w5rtcUser = new W5Peer(API-Key);
w5rtcUser.bandwidth = {audio: 50,video: 256};
```

Note: This works only on Chrome browser; auto-skipped for Firefox. The default value set for audio & video are 50kbps and 256 kbps respectively.

direction Array Attribute

This attribute sets the type of communication; weather the user wants one-way/two-way communication.

Return

Null

Argument

Null

Usage

```
var w5rtcUser = new W5Peer(API-Key);
w5rtcUser.call('callee',{audio:true,direction:'two-way'});
```

datachannel Boolean Attribute

Sets the webrtc data channels to send chat messages or transfer files. It can work without media-streaming. It is used when users want to chat and share files or screen.

Return

Null

Argument

Null

Usage

var w5rtcUser = new W5Peer(API-Key);

w5rtcUser. `registerUserWithId`(caller);

w5rtcUser.datachannel = true;

chunkSize Attribute

This attribute chunks the file into a given size in bytes. Currently 64k bytes is the maximum limit for Chrome-to-Chrome connection and 16k bytes is maximum receiving limit for Firefox.

Return

Null

Argument

Null

Usage

var w5rtcUser = new W5Peer(API-Key);

w5rtcUser.chunkSize = 16 * 1000;

chunkInterval Attribute

This attribute sets the time interval between two chunk packets. The unit is in milliseconds.

Return

Null

Argument

Null

Usage

var w5rtcUser = new W5Peer(API-Key);

w5rtcUser.chunkInterval = 200;

Resolutions	Array Object

This parameter sets the video resolution. You can override this object to set custom video resolutions.

Return

Null

Argument

Null

Elements

{minWidth: value, minHeight: value, maxWidth: value, maxHeight: value}

Usage

var w5rtcUser = new W5Peer(API-Key);

w5rtc.resolutions = {minWidth: 320, minHeight: 180, maxWidth: 1280, maxHeight: 720};

peers	Array Object

This array object holds the list of all users (W5Peer object). You can access a user object (W5Peer object) from the peers list using the user id.

Return

W5Peer object

Argument

Null

Usage

```
var w5rtcUser = new W5Peer(API-Key);
var user1 = w5rtcUser.peers[7564321]
```

8. Troubleshooting

This section explains the common trouble shooting examples. Please contact W5RTC technical support for additional help.

8.1 Self-video not working

Solution:

1. Test camera in another application (such as Skype or WebEx).
2. Completely close the browser, and then reopen it.
3. Shutdown or reset camera and computer.
4. Ensure no other program is using the camera.
5. No hardware / driver support mismatch.

8.2 Peer not able to see video

Solution:

Firewall plays a main role in blocking the video streams.

1. Ensure that both the parties are in same subnet network.
2. Check whether you are connected to the signaling server.
3. Even with TURN, sometimes the firewall blocks the media traffic.

8.3 Poor Video and Audio Quality

Solution:

Check your video camera. Higher the pixels, better the quality.You can modify the default video resolution by using following attribute:

```
w5rtc.setResolutions(width,height)

w5rtc.bandwidth = {

audio:
```

```
video:

}
```

Additional steps:

1. Reduce video resolution.
2. Try using quality headset.
3. Use a conference microphone/speaker with built-in noise cancellation.
4. In W5CAPI we take care of this automatically, the above steps is for testing purpose only.

8.4 Delays in Audio & Video

Solution:

1. Turn server adds some delays to the stream.
2. Often, due to hardware issues, media frames get stuck in the buffer, reducing the resolution. Fewer streams often reduce the effect. Please contact W5RTC if you face such issues.

9. Limitations of W5CAPI

9.1 WebRTC screen sharing has the following issues

It's not supported by Firefox.

It can only be accessed from a page hosted on an SSL server. It

captures all your displays as a single wide video stream.

You may need to enable a chrome flag on the sending browser.

The video is usually down sampled by 2 in each dimension before transmission to

another computer.

It's resource hungry.

Some people can't seem to get it to work on their machine.

Automatic installation screen sharing chrome extension in w5capi in latest w5capi release.

9.2 Browser limitations

We support the real-time features only in Chrome, Firefox,Opera,IE and Safari browsers. We are working in the roadmap to support OS such as Android and IOS etc.

9.3 Other limitations

Please contact W5RTC if you encounter any other issues or to clarify any questions.

10. Reporting bugs or Enhancement

To report bugs or enhancement, please visit w5rtc.com or w5rtc.in website.

1. Click Products -> w5capi

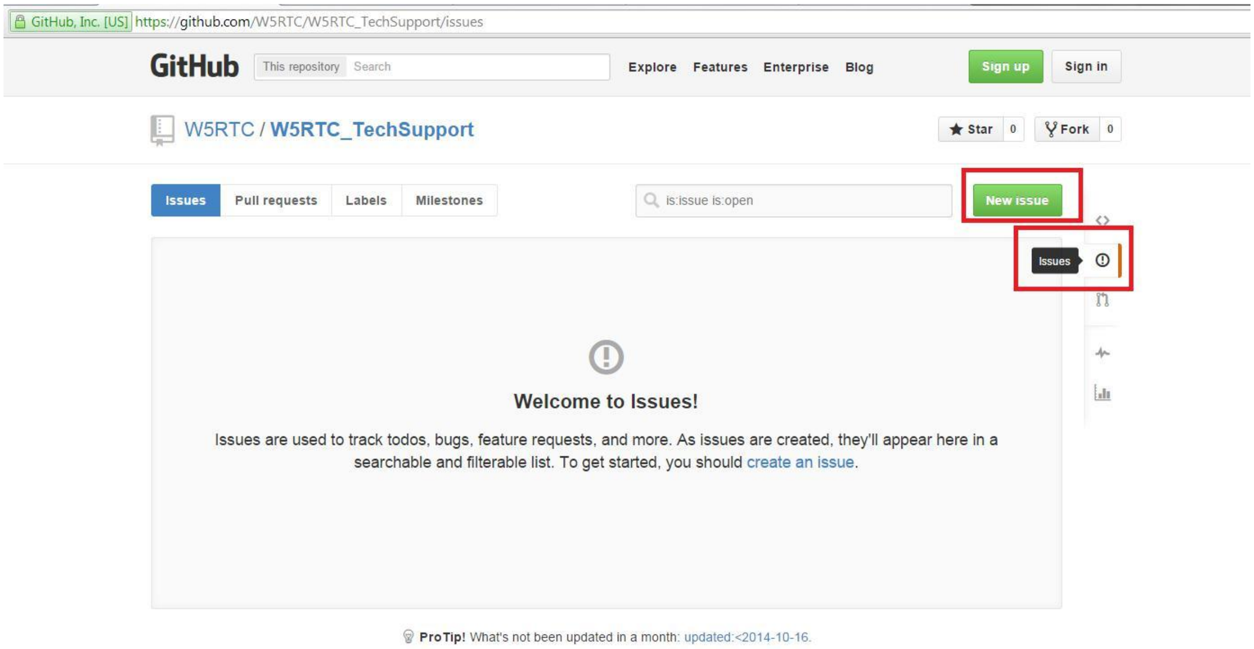
In the w5rtc website page locate below image in the right side of the w5capi page:

SUPPORT
Technical
Support
Write TO US !

2. Click the above image; it will open the github repository link as below or you can directly open this link

https://github.com/W5RTC/W5RTC_TechSupport/issues

3. To report New bug or enhancement, Click on the "New Issue" Button in the Right hand side as depicted below:

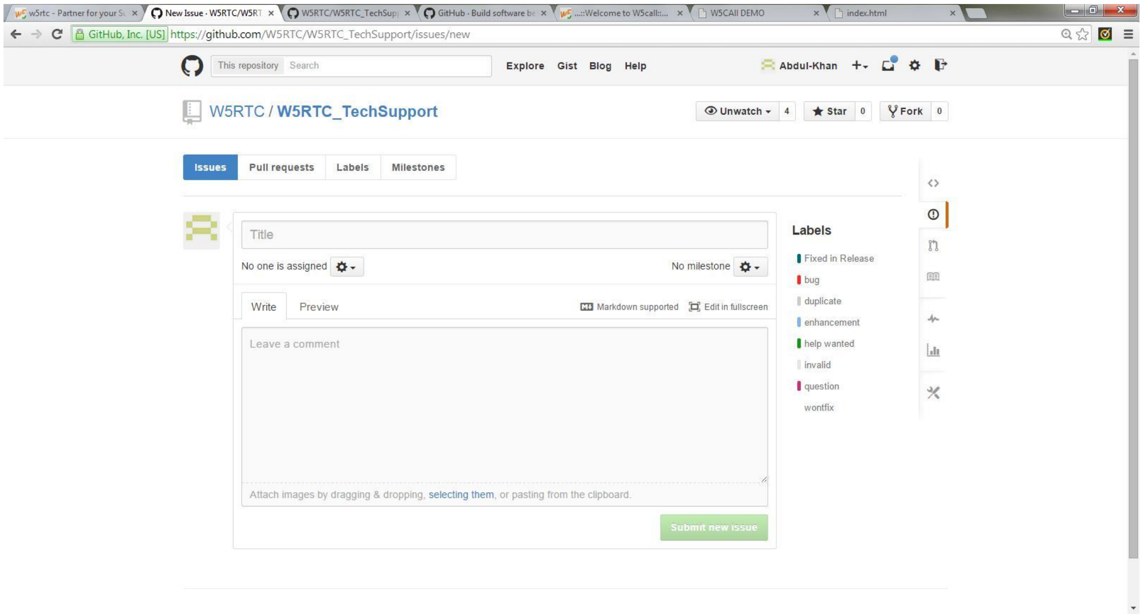


Welcome to Issues!

Issues are used to track todos, bugs, feature requests, and more. As issues are created, they'll appear here in a searchable and filterable list. To get started, you should create an issue.

ProTip! What's not been updated in a month: updated:<2014-10-16.

4. It will ask you to login. So you must have Login and password. We recommend you to create login and password in your organization name directly in the github for easy management.
5. After Logging in you will be able to raise bug or enhancement in the below screen.



Attach images by dragging & dropping, selecting them, or pasting from the clipboard.

6. Ensure to provide enough details so it is easy to work on the defect and deliver the fix faster. We strongly recommend you to provide the following details:

    W5RTC Product Name

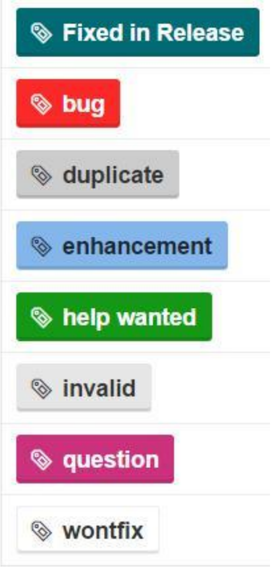
    W5RTC Product Version

    Problem Description

    Steps to Reproduce

    Other Details

Based on the details provided, we will categorize the defect (as below) as applicable and will respond to your query. Also once the fix is available, we will update the release the fix is available and will send the release download and installation instruction to you.



11. FAQ

11.1 What is W5CAPI?

W5CAPI is a real time communication PaaS platform. Using this product, Application software developers can develop the application that supports real-time communication. .

11.2 Why Do I need W5CAPI?

W5CAPI takes care of complex platform and state machine logic that is required to establish real-time communication. This allows the Application software developers/companies to focus on their application logic and tasks that are actually relevant to them.

11.3 How can I use W5CAPI?

You can develop RTC applications for any business that require real time communication services. (i.e.) Call Centers, video conferencing firms, health domains, e-commerce portals, and so on.

You can easily integrate our API in to your application to communicate (video/audio/data) with your customers in real time.

11.4 What is data channel?

It is a live communication session path established across browsers for text messaging, file transfer and file sharing.

11.5 What is media channel?

It is a live communication session path established across browsers to transmit user Voice and Video.

11.6 What is signalling channel?

It is a live communication session path established across browsers for connecting 2 or more users. In the signalling channel, the signalling messages are exchanges between 2 or more users that consist of media capabilities and NAT/ICE IP candidates. .

11.7 What is NAT? How is it related to /ICE/STUN/TURN?

NAT (Network Address Translator) is responsible for converting private IP to public IP so the device with Private IP is able to access as an example the Internet. However to establish the communication between each other, both parties must know the public address of each other. This means they must know the NAT public address. The protocols and technology ICE/STUN/TURN helps to find that public address. Most of the time users are connected peer-to-peer. In case of not possible to establish peer-to-peer, the media is passed through TURN server..

Revision history

Date	Version	Revised by	Comment
Satya Panigrahi	V1.1	Satya Panigrahi	Initial Version

Thank You for partnering with us for Success! Enjoy W5CAPI !

Revision history

Date	Version	Revised by	Comment
Satya Panigrahi	V1.1	Satya Panigrahi	Initial Version

w5rtc

Partner for your success