Overview

To develop web applications for streaming video and calls, use Web SDK. This is a set of scripts and examples to work with the WCS server.

Download

Download Web SDK bundle: Release notes

Link to the current JavaScript implementation: https://flashphoner.com/downloads/builds/flashphoner_client/wcs_api-2.0/current/flashphoner.js

API documentation: http://flashphoner.com/docs/api/WCS5/client/web-sdk/latest

Working with the source code of the examples on your Web server

To work with demo examples on your own Apache, Nginx, Tomcat, IIS, or any other Web server, use the latest available build of the web client

The bundle archive includes the following items:

- doc - JavaScript API documentation
- examples – demo examples
- flashphoner.js – the main API file you should add to your web page, it enables all supported technologies.
- flashphoner-webrtc-only.js - the alternative API file if you plan to use WebRTC only
- flashphoner-no-flash.js – the alternative API file if you don’t plan using Flash
- flashphoner-no-webrtc.js - the alternative API file if you don’t plan using WebRTC
- flashphoner-no-wsplayer.js - the alternative API file if you don’t plan using the Websocket player
- media-provider.swf – the file to support Flash operation

We will review individual examples below.

Working with the source code of the examples directly on the WCS server

If you installed Web Call Server, you can work with the source code of the demo examples directly.

The code of the examples is located at:

/usr/local/FlashphonerWebCallServer/client2/examples/demo

To test an individual example, open the corresponding page in the browser, for example:

```
https://host:8888/client2/examples/demo/streaming/player/player.html
```

Here, host is the address of your WCS server.

Therefore, you can make necessary changes to the scripts and test the modified demo example directly on the WCS server.

Source code of API and examples on Github

https://github.com/flashphoner/flashphoner_client/tree/wcs_api-2.0

Usually, you only need builds from the above links. Most likely you will not need the source code. It is a last resort to make a quick fix on the API level.

In this documentation we use the source code to comment how examples work. For example, this line 3 is a reference to the third line of the source code of package.json with the hash of 0b891b8.

Dependencies

WebSDK is build with webrtc/adapter library version not lower than 7.2.6. In this regard, direct use of this library together with WebSDK should be avoided.

Known issues

1. WebRTC publishing may not work in WKWebView on iOS old versions

When web application is opened via WKWebView on iOS 11 and higher, stream playback works only via WSPlayer, and stream publishing and playback via WebRTC do not work

Symptoms

When Two-Way Streaming example is opened in an iOS application using WKWebView to view web links (e.g., Telegram) both stream publishing and playback do not work; in Player example stream playback works via WSPlayer.

The same way will work a web page opened from home screen, if its code contains

```
<meta name="apple-mobile-web-app-capable" content="yes" />
```

Solution

Use WKWebView only for web applications designed for playing streams without calling getUserMedia() function, which is not supported in WKWebView.

In case WebRTC is required, remove <meta name="apple-mobile-web-app-capable" content="yes"> from page code to open it in Safari.