

WebRTC as RTMP re-publishing

- [Пример ре-публикации WebRTC аудио / видео потока как RTMP](#)
- [Код примера](#)
- [Работа с кодом примера](#)

Пример ре-публикации WebRTC аудио / видео потока как RTMP

Данный пример показывает как из браузера можно отправить аудио и видео на сервер по технологии WebRTC и далее перенаправить полученный трафик на этот же или на другой сервер по протоколу RTMP.

На скриншоте ниже браузер установил подключение к WCS-серверу и отправляет аудио и видео на сервер, где происходит ре-публикация на localhost по протоколу RTMP.



Чтобы воспроизвести перенаправленный поток, необходимо скопировать ссылку в сторонний RTMP плеер, например, в VLC.

Код примера

Код данного примера находится на WCS-сервере по следующему пути:

`/usr/local/FlashphonerWebCallServer/client2/examples/demo/streaming/webrtc-as-rtmp-republishing`

webrtc-as-rtmp-republishing.css - файл стилей
webrtc-as-rtmp-republishing.html - страница примера
webrtc-as-rtmp-republishing.js - скрипт

Тестировать данный пример можно по следующему адресу:

`https://host:8888/client2/examples/demo/streaming/conference/webrtc-as-rtmp-republishing.html`

Здесь host - адрес WCS-сервера.

Работа с кодом примера

Для разбора кода возьмем версию файла webrtc-as-rtmp-republishing.js с хешем ecbadc3, которая находится [здесь](#) и доступна для скачивания в соответствующей сборке [2.0.212](#).

Скрипт создает подключение к WCS-серверу и управляет публикацией WebRTC потока.

1. Инициализация API.

Flashphoner.init() [code](#)

```
Flashphoner.init();
```

2. Подключение к серверу.

Flashphoner.createSession() [code](#)

```
Flashphoner.createSession({urlServer: url}).on(SESSION_STATUS.ESTABLISHED, function(session){
    ...
});
```

3. Получение от сервера события, подтверждающего успешное соединение.

ConnectionStatusEvent ESTABLISHED [code](#)

```
Flashphoner.createSession({urlServer: url}).on(SESSION_STATUS.ESTABLISHED, function(session){
    //session connected, start streaming
    startStreaming(session);
}).on(SESSION_STATUS.DISCONNECTED, function(){
    ...
}).on(SESSION_STATUS.FAILED, function(){
    ...
});
```

4. Публикация WebRTC видеопотока с републикацией на RTMP-сервер.

session.createStream(), stream.publish() [code](#)

При создании потока методу createStream() передается параметр rtmpUrl, который содержит адрес RTMP-сервера для ре-публикации этого потока.

Имя ре-публикуемого RTMP потока формируется из стандартного префикса, указанного в опции rtmp_transponder_stream_name_prefix в файле [f lashphoner.properties](#), и параметра streamName

```
session.createStream({
    name: streamName,
    display: localVideo,
    cacheLocalResources: true,
    receiveVideo: false,
    receiveAudio: false,
    rtmpUrl: rtmpUrl
    ...
}).publish();
```

5. Получение от сервера события, подтверждающего успешную публикацию потока.

StreamStatusEvent PUBLISHING [code](#)

```
session.createStream({
    ...
}).on(STREAM_STATUS.PUBLISHING, function(publishStream){
    setStatus(STREAM_STATUS.PUBLISHING);
    onStart(publishStream);
    sendDataToPlayer();
}).on(STREAM_STATUS.UNPUBLISHED, function(){
    ...
}).on(STREAM_STATUS.FAILED, function(){
    ...
}).publish();
```

6. Формирование URL RTMP-потока для отображения на странице и копирования во внешний плеер

sendDataToPlayer() [code](#)

```
function sendDataToPlayer() {  
    var player = document.getElementById("player");  
    var host = field("rtmpUrl")  
        .replace("localhost", window.location.hostname)  
        .replace("127.0.0.1", window.location.hostname);  
  
    var rtmpStreamPrefix = "rtmp_";  
    var url = host + "/" + rtmpStreamPrefix + field("streamName");  
    player.setURLtoFlash(url);  
}
```