В браузере по MSE

Описание

Media Source Extensions (MSE) — это API браузера, позволяющее играть аудио и видео через соответствующие HTML5 тэги<audio>и<video>. Если WebRTC предназначен как для воспроизведения, так и для трансляции потоков в реальном времени, то MSE - только для воспроизведения. Таким образом, технология MSE может быть использована, если необходимо только проигрывать поток на странице, и при этом нет жестких требований к задержкам.

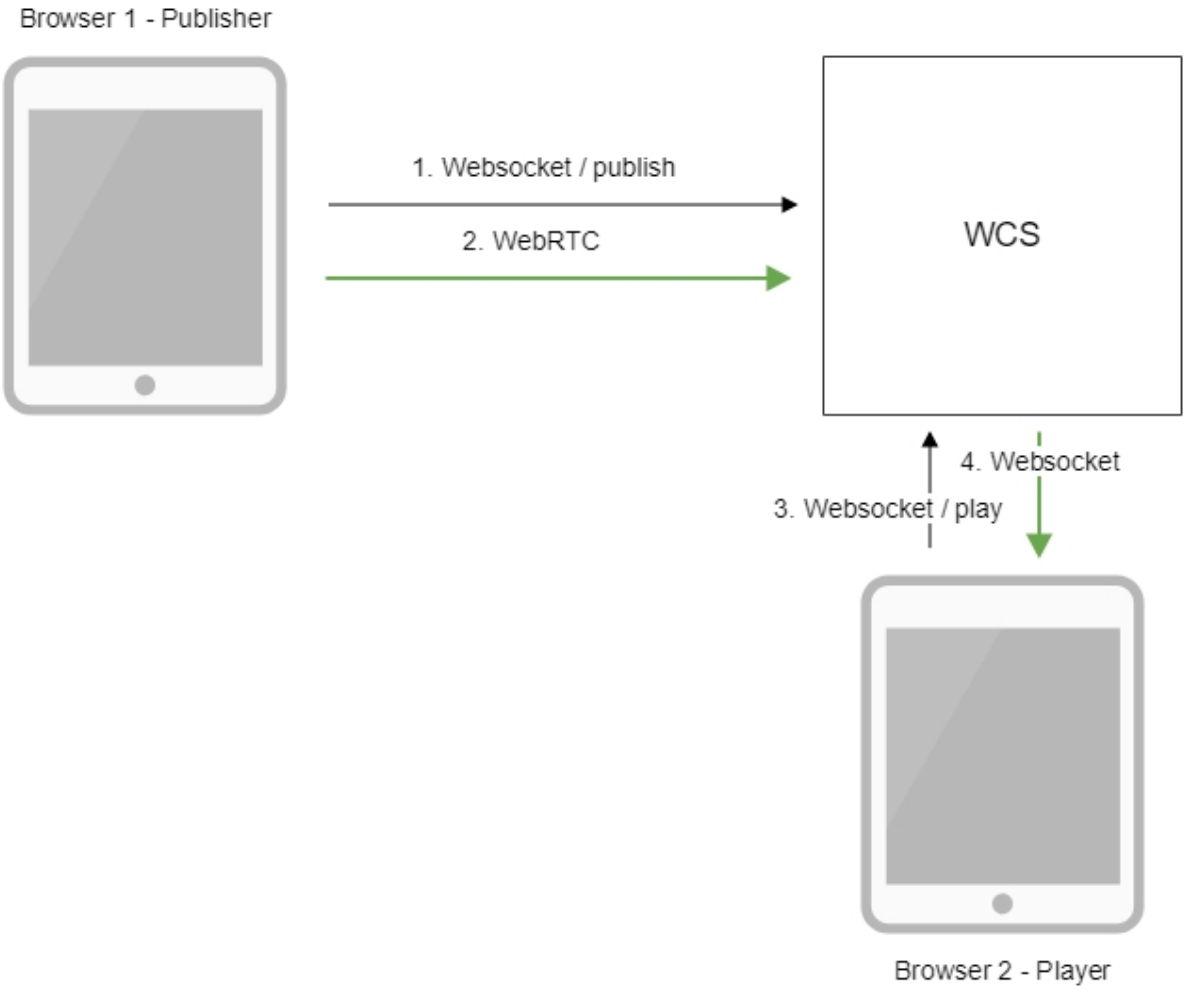
Поддерживаемые платформы и браузеры

	Chrome	Firefox	Safari 11	Edge
Windows	+	+		+
Mac OS	+	+	+	
Android	+	+		
iPadOS	-	-	+	

Поддерживаемые кодеки

- Видео: H.264
- Аудио: AAC

Схема работы



1. Браузер соединяется с сервером по протоколу Websocket и отправляет команду publish.
2. Браузер захватывает микрофон и камеру и отправляет WebRTC поток на сервер.
3. Второй браузер устанавливает соединение также по Websocket и отправляет команду play.
4. Второй браузер получает H.264 + AAC поток по Websocket и воспроизводит этот поток при помощи MSE на странице.

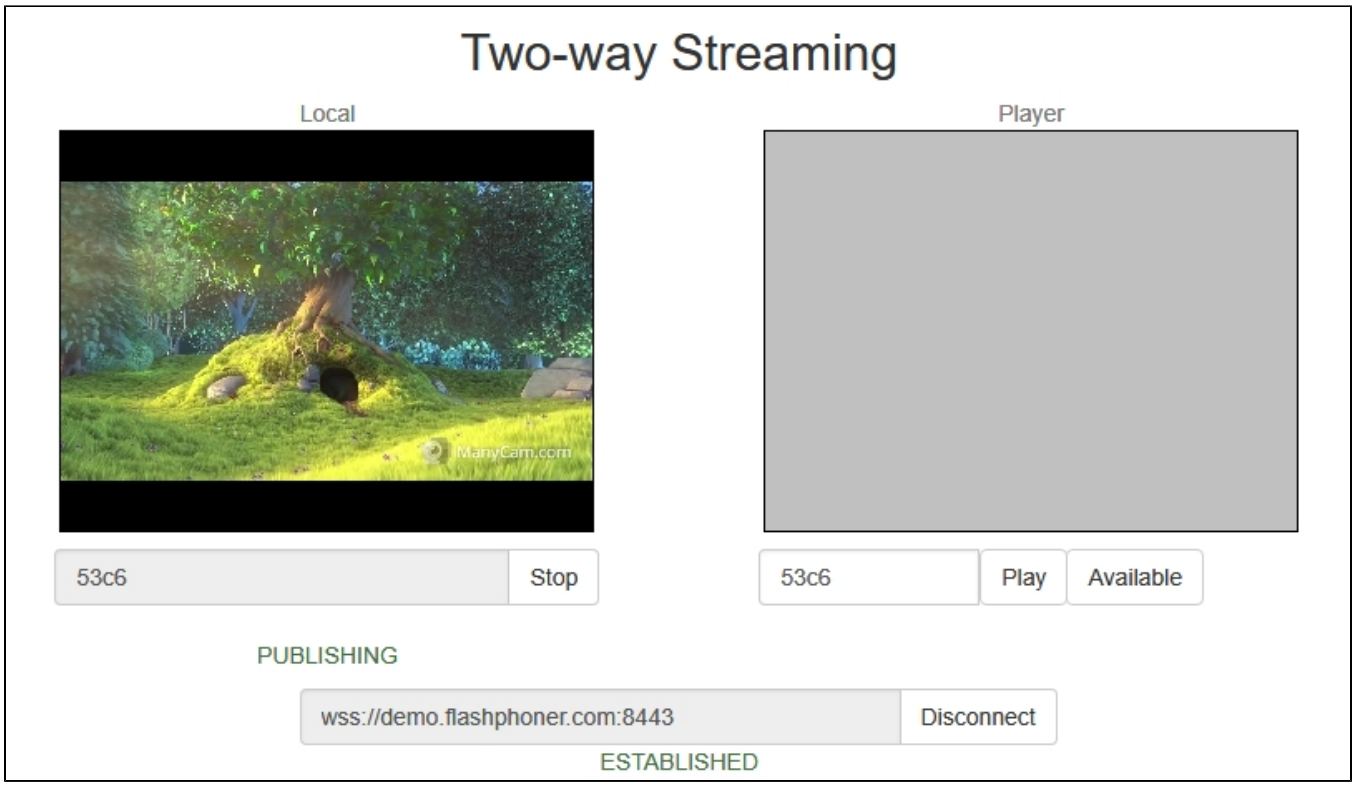
Краткое руководство по тестированию

Трансляция видеопотока на сервер и воспроизведение его по MSE в браузере

1. Для теста используем:

- демо-сервер demo.flashphoner.com;
- веб-приложениеTwo Way Streaming
- для публикации потока
- веб-приложениеPlayerдля воспроизведения потока по MSE

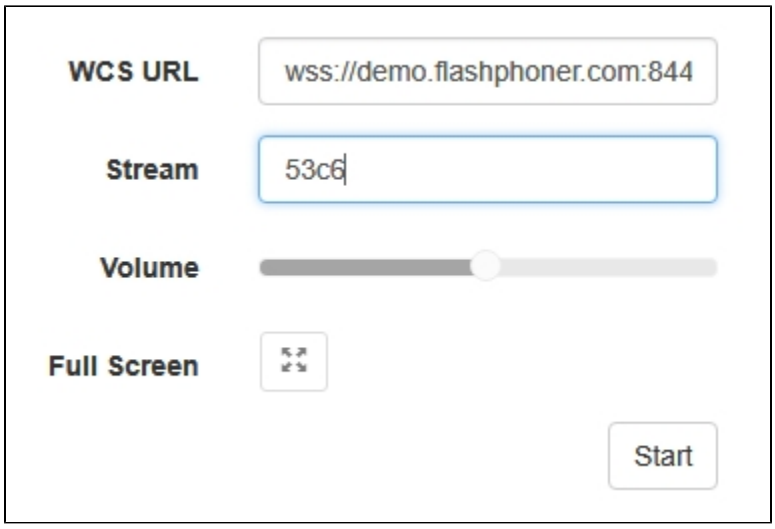
2. Откройте веб-приложение Two Way Streaming. Нажмите Connect, затем Publish. Скопируйте идентификатор потока:



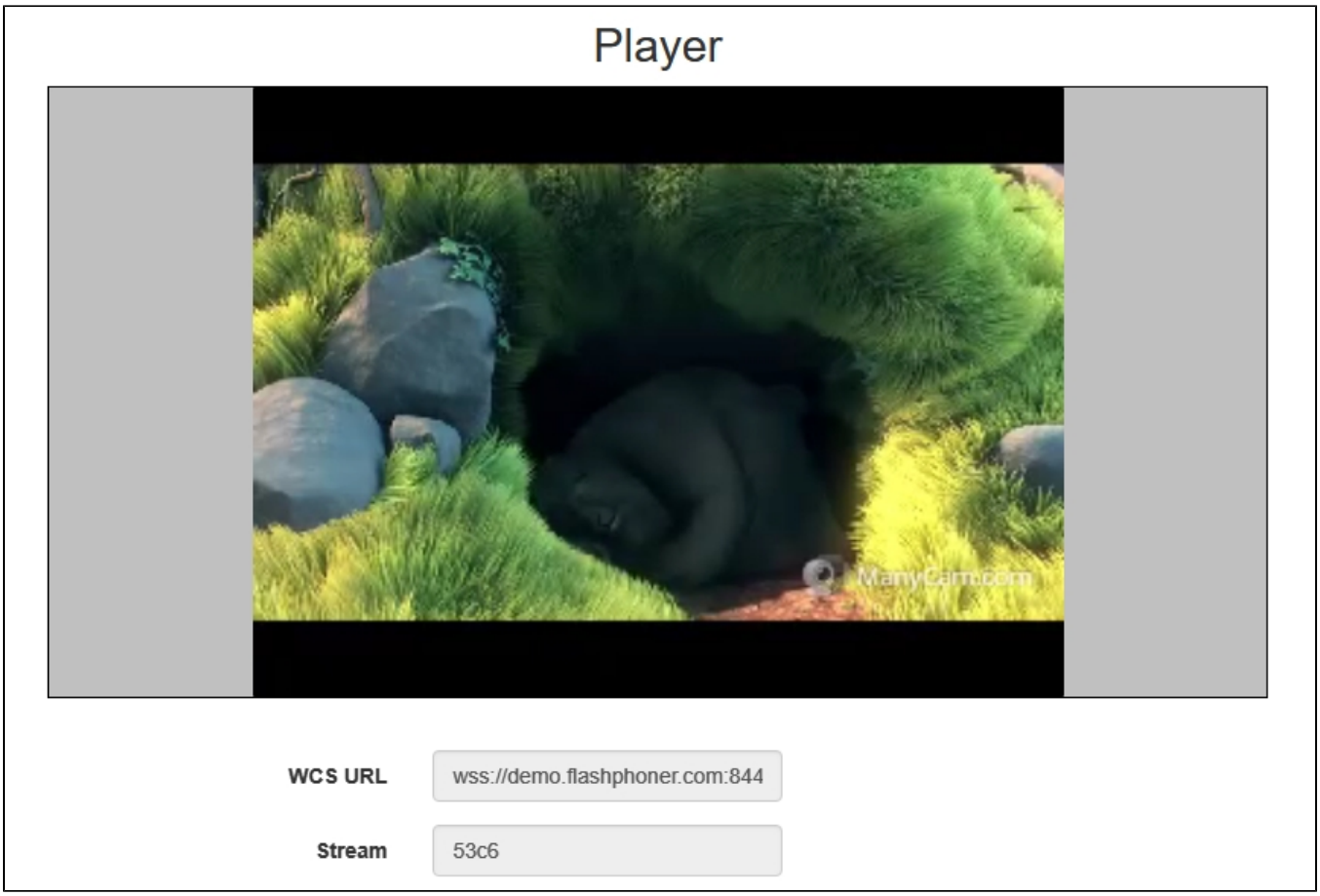
3. Откройте веб-приложение Player, указав в параметрах URL технологию MSE

https://demo.flashphoner.com/client2/examples/demo/streaming/player/player.html?mediaProvider=MSE

4. Укажите в поле Stream идентификатор потока:



5. Нажмите кнопку Start. Начнется воспроизведение потока:

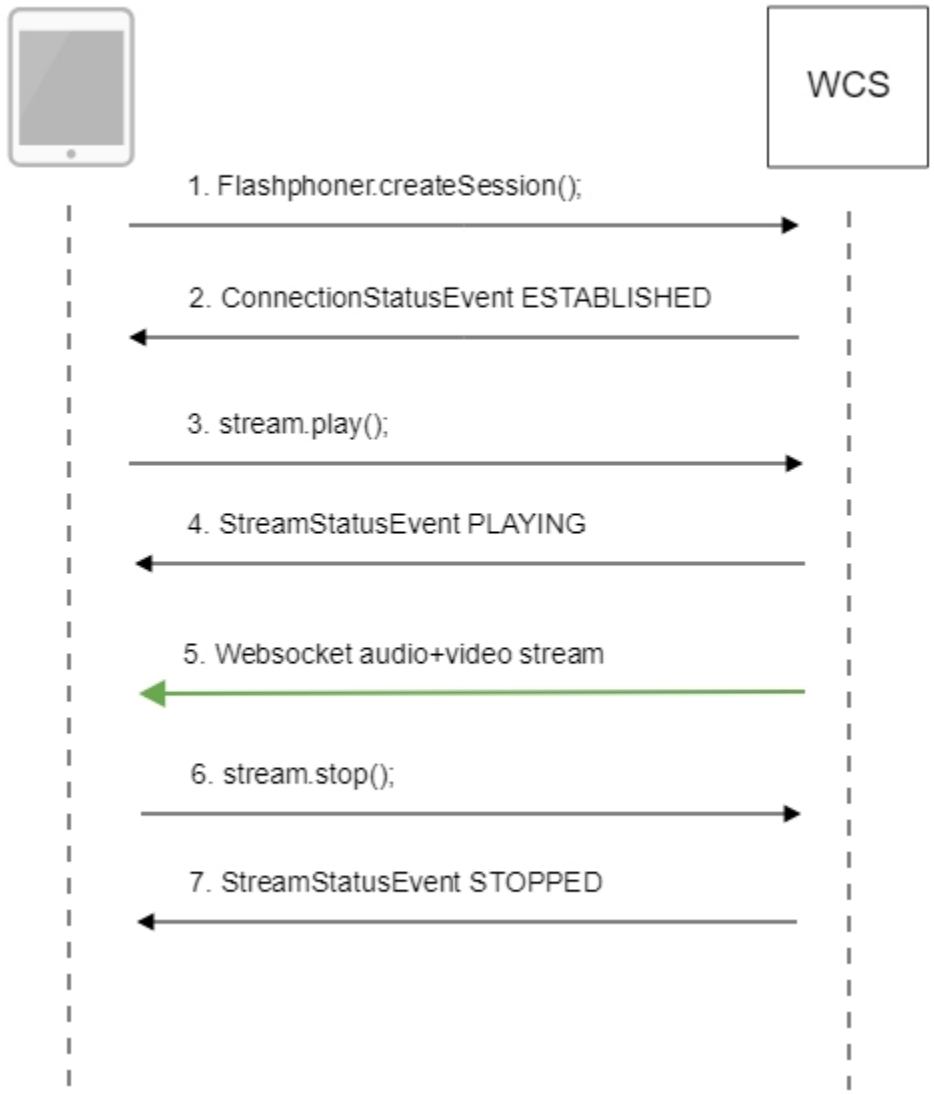


Последовательность выполнения операций (Call flow)

Ниже описана последовательность вызовов при использовании примера Player для воспроизведения потока по MSE

player.html

player.js



1. Установка соединения с сервером.

Flashphoner.createSession();code

```
    Flashphoner.createSession({urlServer: url}).on(SESSION_STATUS.ESTABLISHED, function(session){
        setStatus(session.status());
        //session connected, start playback
        playStream(session);
    }).on(SESSION_STATUS.DISCONNECTED, function(){
        setStatus(SESSION_STATUS.DISCONNECTED);
        onStopped();
    }).on(SESSION_STATUS.FAILED, function(){
        setStatus(SESSION_STATUS.FAILED);
        onStopped();
    });
```

2. Получение от сервера события, подтверждающего успешное соединение.

ConnectionStatusEvent ESTABLISHEDcode

```
    Flashphoner.createSession({urlServer: url}).on(SESSION_STATUS.ESTABLISHED, function(session){
        setStatus(session.status());
        //session connected, start playback
        playStream(session);
    }).on(SESSION_STATUS.DISCONNECTED, function(){
        ...
    }).on(SESSION_STATUS.FAILED, function(){
        ...
    });
```

3. Воспроизведение потока.

stream.play();code

```
    if (Flashphoner.getMediaProviders()[0] === "MSE" && mseCutByIFrameOnly) {
        options.mediaConnectionConstraints = {
            cutByIFrameOnly: mseCutByIFrameOnly
        }
    }
    ...
    stream = session.createStream(options).on(STREAM_STATUS.PENDING, function(stream) {
        ...
    });
    stream.play();
```

4. Получение от сервера события, подтверждающего успешное воспроизведение потока.

StreamStatusEvent, статус PLAYINGcode

```
    stream = session.createStream(options).on(STREAM_STATUS.PENDING, function(stream) {
        ...
    }).on(STREAM_STATUS.PLAYING, function(stream) {
        $("#preloader").show();
        setStatus(stream.status());
        onStarted(stream);
    }).on(STREAM_STATUS.STOPPED, function() {
        ...
    }).on(STREAM_STATUS.FAILED, function(stream) {
        ...
    }).on(STREAM_STATUS.NOT_ENOUGH_BANDWIDTH, function(stream){
        ...
    });
    stream.play();
```

5. Прием аудио-видео потока по Websocket и воспроизведение по MSE

6. Остановка воспроизведения потока.

stream.stop();code

```
function onStarted(stream) {
    $("#playBtn").text("Stop").off('click').click(function(){
        $(this).prop('disabled', true);
        stream.stop();
    }).prop('disabled', false);
    ...
}
```

7. Получение от сервера события, подтверждающего остановку воспроизведения потока.

StreamStatusEvent, статус STOPPEDcode

```
    stream = session.createStream(options).on(STREAM_STATUS.PENDING, function(stream) {
        ...
    }).on(STREAM_STATUS.PLAYING, function(stream) {
        ...
    }).on(STREAM_STATUS.STOPPED, function() {
        setStatus(STREAM_STATUS.STOPPED);
        onStopped();
    }).on(STREAM_STATUS.FAILED, function(stream) {
        ...
    }).on(STREAM_STATUS.NOT_ENOUGH_BANDWIDTH, function(stream){
        ...
    });
    stream.play();
```

Буферизация MSE

При большом количестве подписчиков, играющих потоки с помощью MSE, повышается средняя загрузка процессора на сервере. Для снижения нагрузки, в версии5.2.360добавлена возможность буферизации кадров, отправляемых MSE-подписчику. Количество кадров, передаваемых в одном пакете, определяется настройкой в файле flashphoner.properties

```
avcc_buffer_wait_frames_count=5
```

По умолчанию, в одном пакете передается 5 кадров

Размер буфера для отправляемых пакетов задается в байтах настройкой

```
avcc_send_buffer_size=500000
```

По умолчанию, размер буфера составляет 500 кбайт. Если пакет не помещается в буфер, он сразу отправляется подписчику с выводом в лог сообщения об ошибке

```
12:00:50,555 ERROR       AvccSendBuffer - VideoProcessor-db2da9a0-ddb6-11e9-9fc2-cf9284f3bdd0 Failed to buffer
frame
```

Для снижения средней загрузки процессора количество кадров в пакете и размер буфер рекомендуется увеличить. Отметим, что чем больше буферизация, тем больше вносимая ею задержка.

При необходимости, буферизация может быть отключена при помощи изменения параметра `msePacketizationVersion` в исходных текстах WebSDK

```
        wsConnection.onopen = function () {
            onSessionStatusChange(SESSION_STATUS.CONNECTED);
            cConfig = {
                appKey: appKey,
                mediaProviders: Object.keys(MediaProvider),
                keepAlive: keepAlive,
                authToken:authToken,
                clientVersion: "0.5.28",
                clientOSVersion: window.navigator.appVersion,
                clientBrowserVersion: window.navigator.userAgent,
                msePacketizationVersion: 2,
                custom: options.custom
            };
```

на

```
                msePacketizationVersion: 1,
```

В этом случае настройки буферизации работать не будут, кадры будут отправляться непосредственно MSE-подписчикам.

Известные проблемы

1. При воспроизведении видеопотока, опубликованного из Flash клиента с низким FPS, по MSE с установленной настройкой mseCutByIFrameOnly=true и включенным транскодингомв браузерах MS Edge и Internet Explorer 11 возможны фризы.

Симптомы: при воспроизведении видео, опубликованного из Flash клиента, в приложении Player с явно указанным разрешением и выставленной настройкой mseCutByIFrameOnly=true, напримерhttps://server:8888/client2/examples/demo/streaming/player/player.html?resolution=320x240&mediaProvider=MSE&mseCutByIFrameOnly=trueв браузере MS Edge или Internet Explorer 11 наблюдаются частые фризы.

Решение:

а) при публикации потока из Flash клиента FPS должен быть не ниже 25, также желательно избегать транскодинга;

б) если увеличить FPS невозможно, необходимо уменьшать следующий параметр в файле flashphoner.properties, например

```
video_encoder_h264_gop=30
```

2. MSE не поддерживается в iOS Safari на iPhone.

Симптомы: воспроизведение потока по MSE на iPhone с iOS 12 и выше не запускается, в примере Embed Player при этом отображается сообщение "None of preferred media providers available"

Решение:

а) использовать WebRTC на iPhone с iOS 12 и выше

б) если необходимо однопортовое соединение, использовать WSPlayer или TURN сервер

3. Нельзя воспроизвести два потока по MSE через одно Websocket соединение на одной странице

Симптомы: в примере 2Players не играют два потока при подключении по HTTP в основных браузерах (Chrome, Firefox, Safari)

Решение: использовать отдельное Websocket соединение для каждого потока на одной странице при воспроизведении по MSE