В браузере по WebRTC

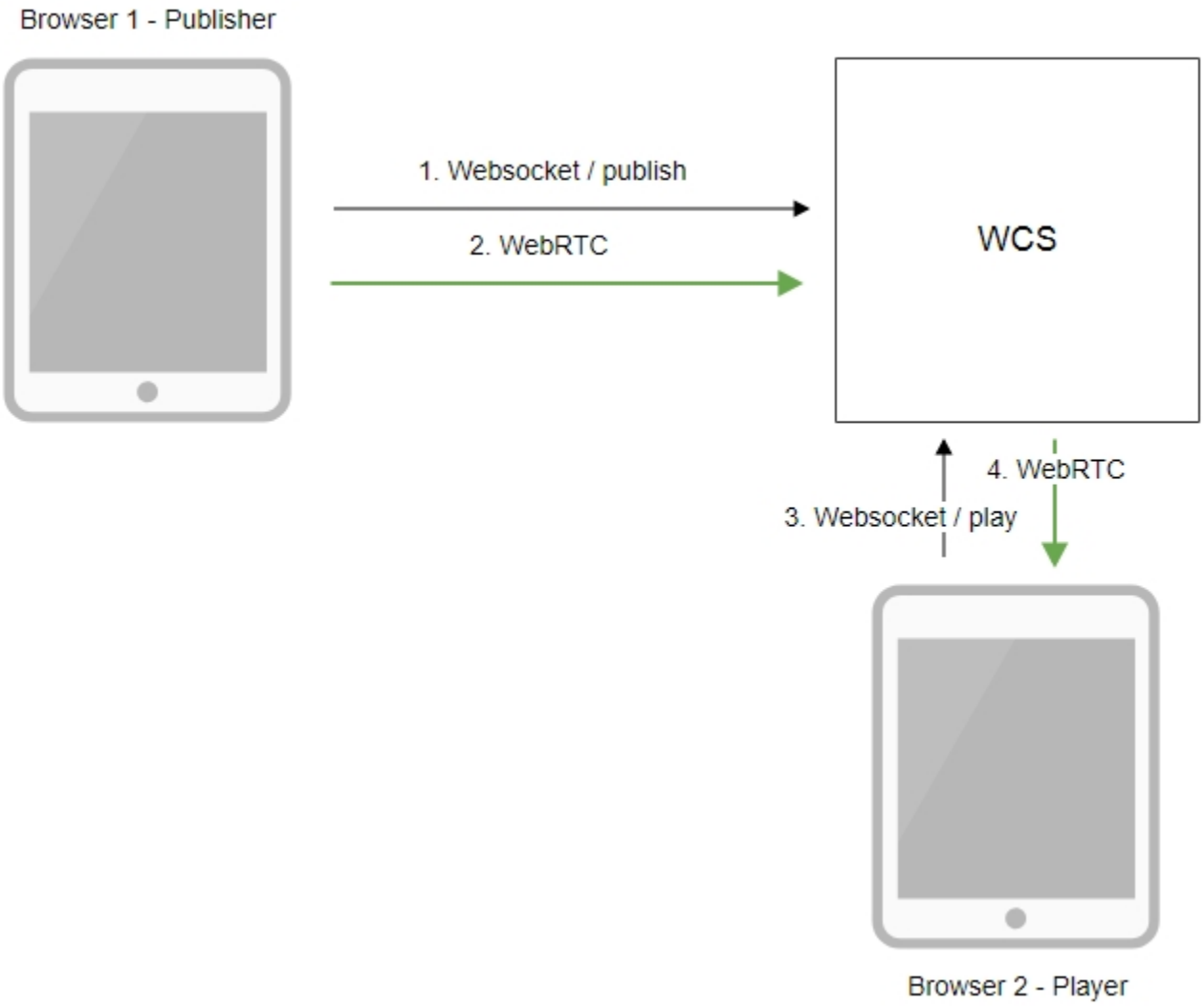
- Описание
  - Поддерживаемые платформы и браузеры
  - Схема работы
- Краткое руководство по тестированию
  - Трансляция видеопотока на сервер и воспроизведение его по WebRTC в браузере
- Последовательность выполнения операций (Call flow)
- Воспроизведение двух и более потоков на одной странице
  - Максимальное количество потоков, которые можно играть на одной странице
- Автозапуск воспроизведения
  - Особенности автозапуска воспроизведения в браузерах
    - Chrome
    - Firefox и MacOS Safari
    - iOS Safari
- Тонкая настройка воспроизведения звука в iOS Safari
- Воспроизведение стерео звука в браузере
  - Браузеры на основе Chrome
    - С использованием Web SDK
    - С использованием Websocket API
- Дополнительная задержка при воспроизведении видеопотока
- Известные проблемы

Описание

Поддерживаемые платформы и браузеры

	Chrome	Firefox	Safari 11	Edge
Windows	+	+		+
Mac OS	+	+	+	
Android	+	+		
iOS	+ (iOS 14.4)	+(iOS 14.4)	+	

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 WebRTC поток и воспроизводит этот поток на странице.

Краткое руководство по тестированию

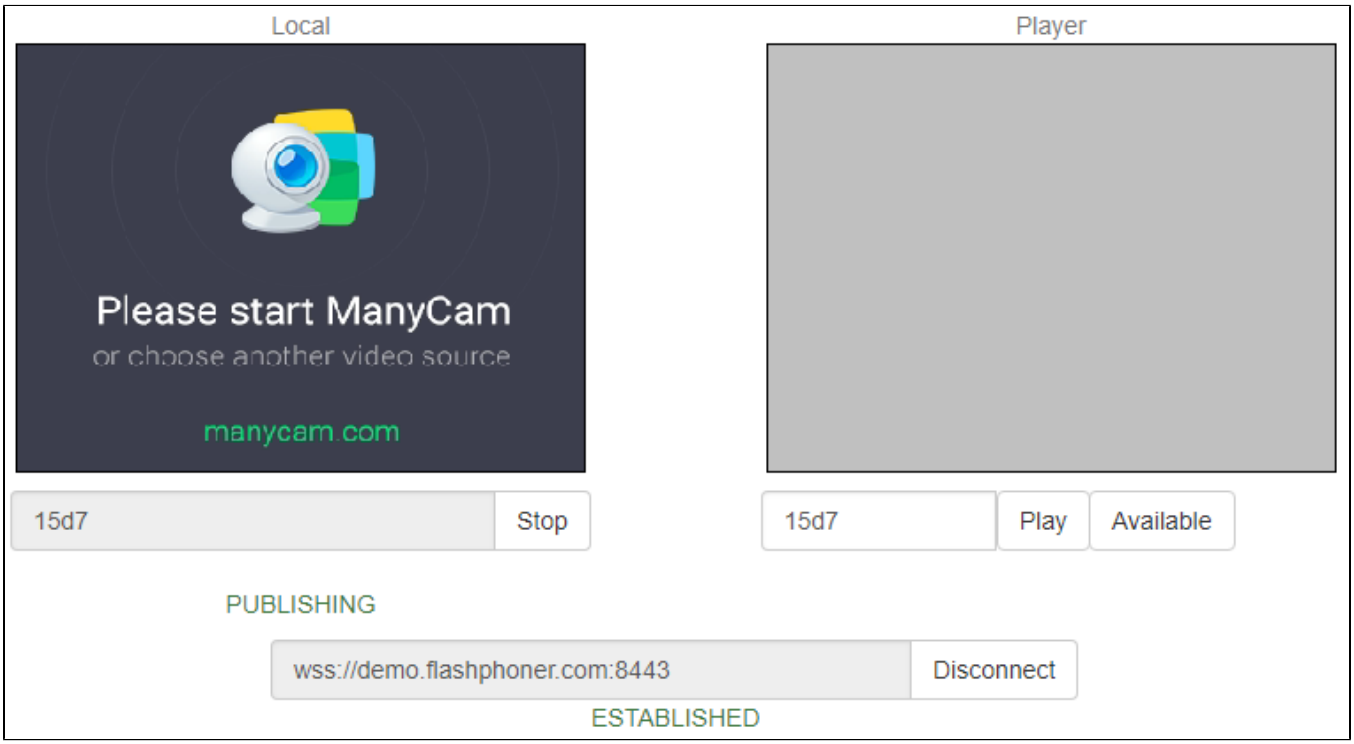
Трансляция видеопотока на сервер и воспроизведение его по WebRTC в браузере

1. Для теста используем демо-сервер demo.flashphoner.com и веб-приложение Two Way Streaming

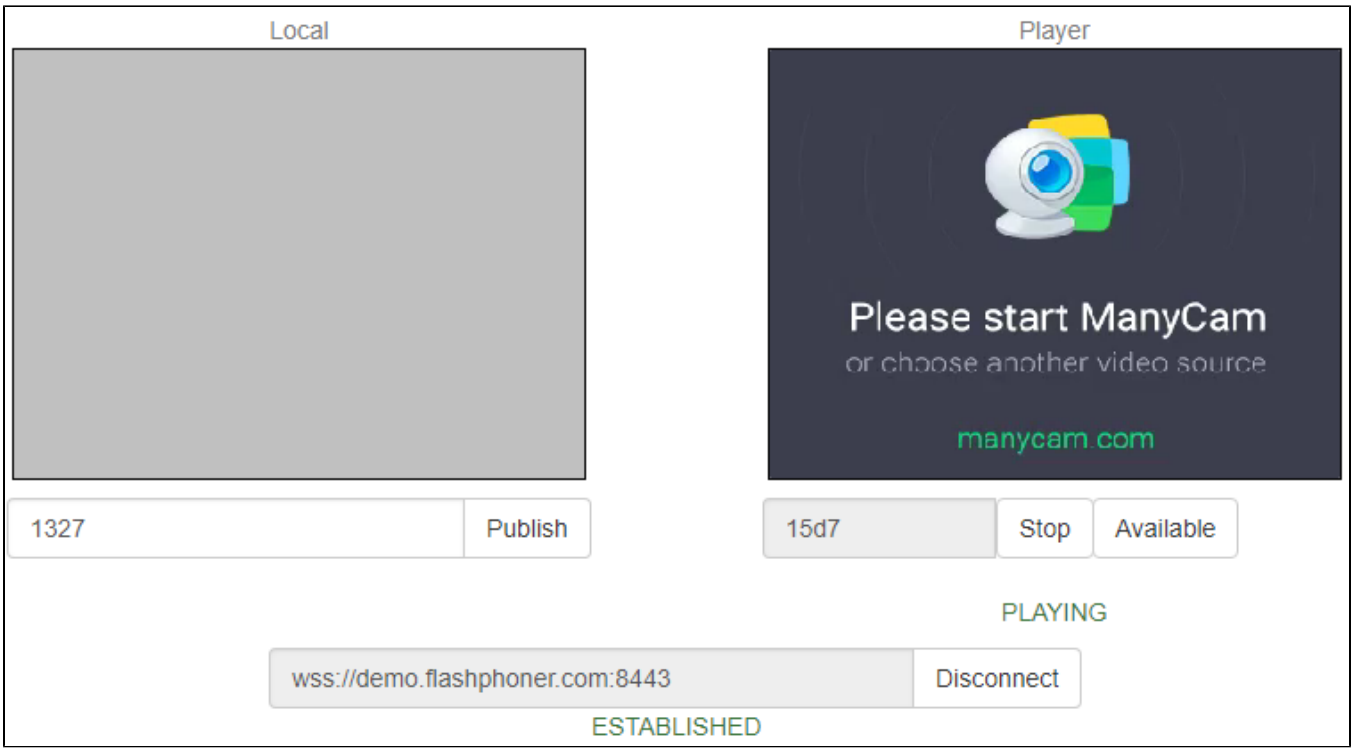
https://demo.flashphoner.com/client2/examples/demo/streaming/two_way_streaming/two_way_streaming.html

2. Установите соединение с сервером по кнопке Connect

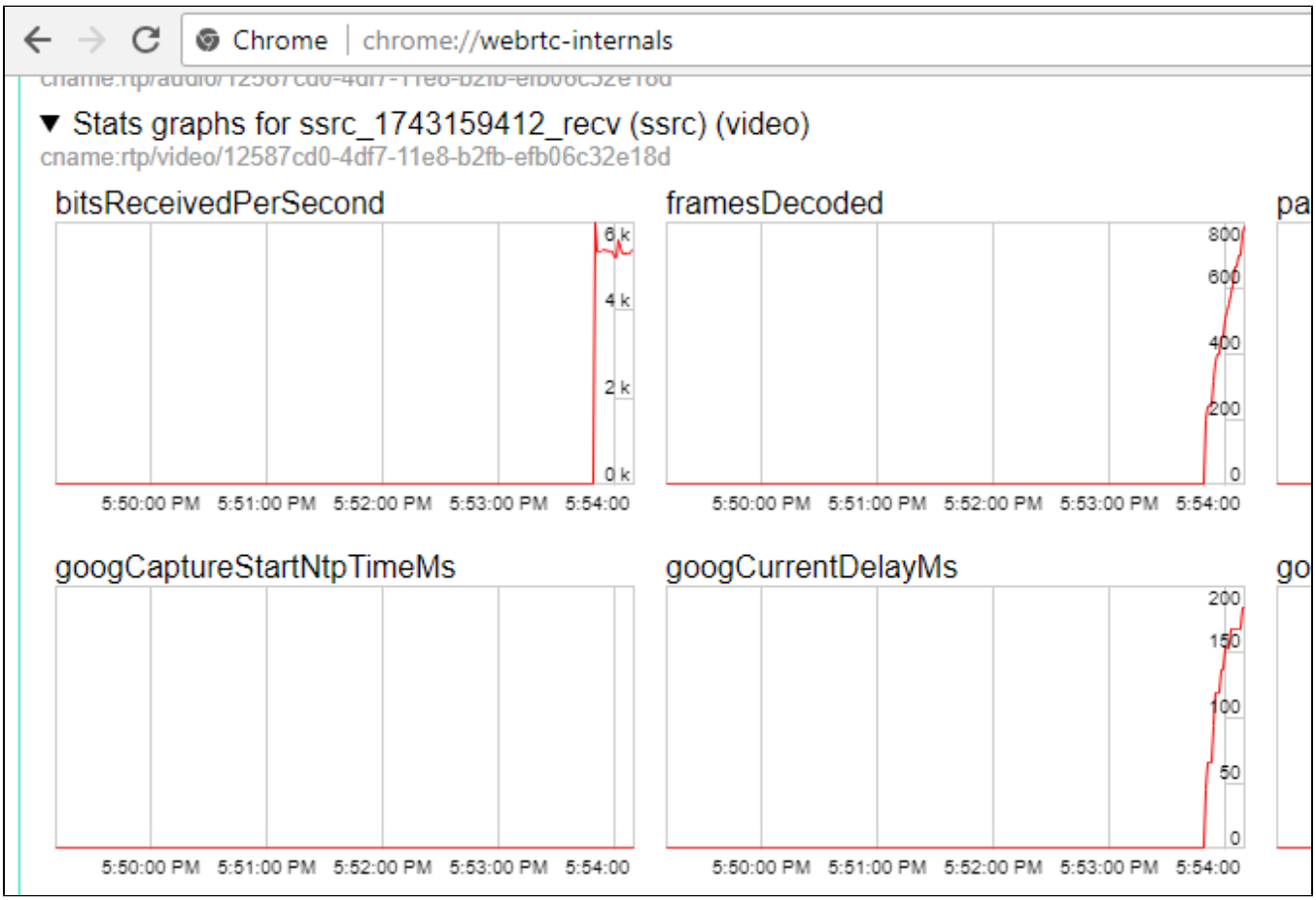
3. Нажмите Publish. Браузер захватывает камеру и отправляет поток на сервер.



4. Откройте Two Way Streaming в отдельном окне, нажмите Connect и укажите идентификатор потока, затем нажмите Play.



5. Графики воспроизведения chrome://webrtc-internals



Последовательность выполнения операций (Call flow)

Ниже описана последовательность вызовов при использовании примера Two Way Streaming для воспроизведения потока

two_way_streaming.html

two_way_streaming.js

1. Установка соединения с сервером.

Flashphoner.createSession();code

```
    Flashphoner.createSession({urlServer: url}).on(SESSION_STATUS.ESTABLISHED, function (session) {
        setStatus("#connectStatus", session.status());
        onConnected(session);
    }).on(SESSION_STATUS.DISCONNECTED, function () {
        setStatus("#connectStatus", SESSION_STATUS.DISCONNECTED);
        onDisconnected();
    }).on(SESSION_STATUS.FAILED, function () {
        setStatus("#connectStatus", SESSION_STATUS.FAILED);
        onDisconnected();
    });
```

2. Получение от сервера события, подтверждающего успешное соединение.

ConnectionStatusEvent ESTABLISHEDcode

```
    Flashphoner.createSession({urlServer: url}).on(SESSION_STATUS.ESTABLISHED, function (session) {
        setStatus("#connectStatus", session.status());
        onConnected(session);
    }).on(SESSION_STATUS.DISCONNECTED, function () {
        ...
    }).on(SESSION_STATUS.FAILED, function () {
        ...
    });
```

3. Воспроизведение потока.

stream.play();code

```
    session.createStream({
        name: streamName,
        display: remoteVideo
        ...
    }).play();
```

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

StreamStatusEvent, статус PLAYINGcode

```
    session.createStream({
        name: streamName,
        display: remoteVideo
    }).on(STREAM_STATUS.PENDING, function(stream) {
        ...
    }).on(STREAM_STATUS.PLAYING, function (stream) {
        setStatus("#playStatus", stream.status());
        onPlaying(stream);
    }).on(STREAM_STATUS.STOPPED, function () {
        ...
    }).on(STREAM_STATUS.FAILED, function (stream) {
        ...
    }).play();
```

5. Прием аудио-видео потока по WebRTC

6. Остановка воспроизведения потока.

stream.stop();code

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

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

StreamStatusEvent, статус STOPPEDcode

```
    session.createStream({
        name: streamName,
        display: remoteVideo
    }).on(STREAM_STATUS.PENDING, function(stream) {
        ...
    }).on(STREAM_STATUS.PLAYING, function (stream) {
        ...
    }).on(STREAM_STATUS.STOPPED, function () {
        setStatus("#playStatus", STREAM_STATUS.STOPPED);
        onStopped();
    }).on(STREAM_STATUS.FAILED, function (stream) {
        ...
    }).play();
```

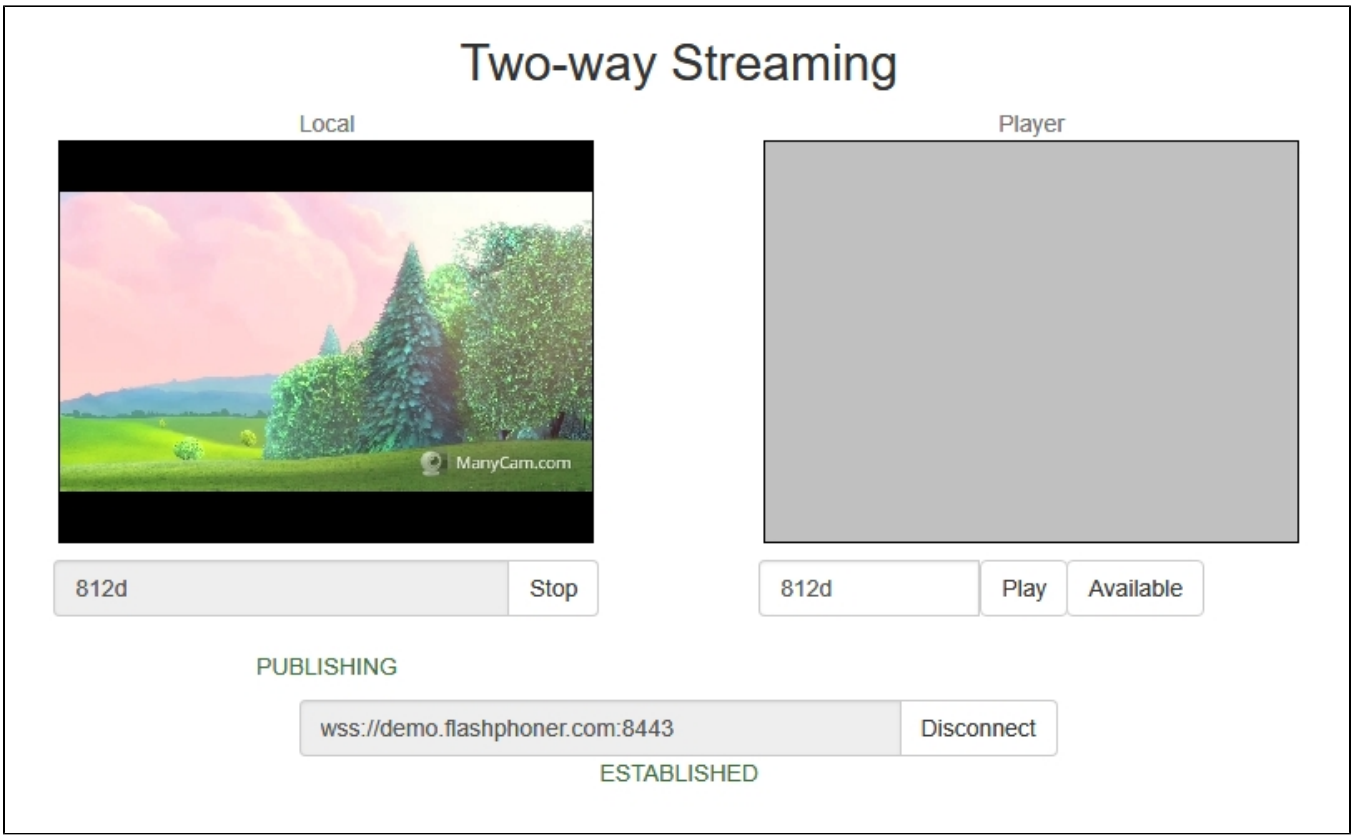
Воспроизведение двух и более потоков на одной странице

WCS предоставляет возможность воспроизведения двух и более потоков на одной странице. С точки зрениясхемы работыипоследовательности выполнения операцийвоспроизведение любого числа потоков не отличается от воспроизведения одного.

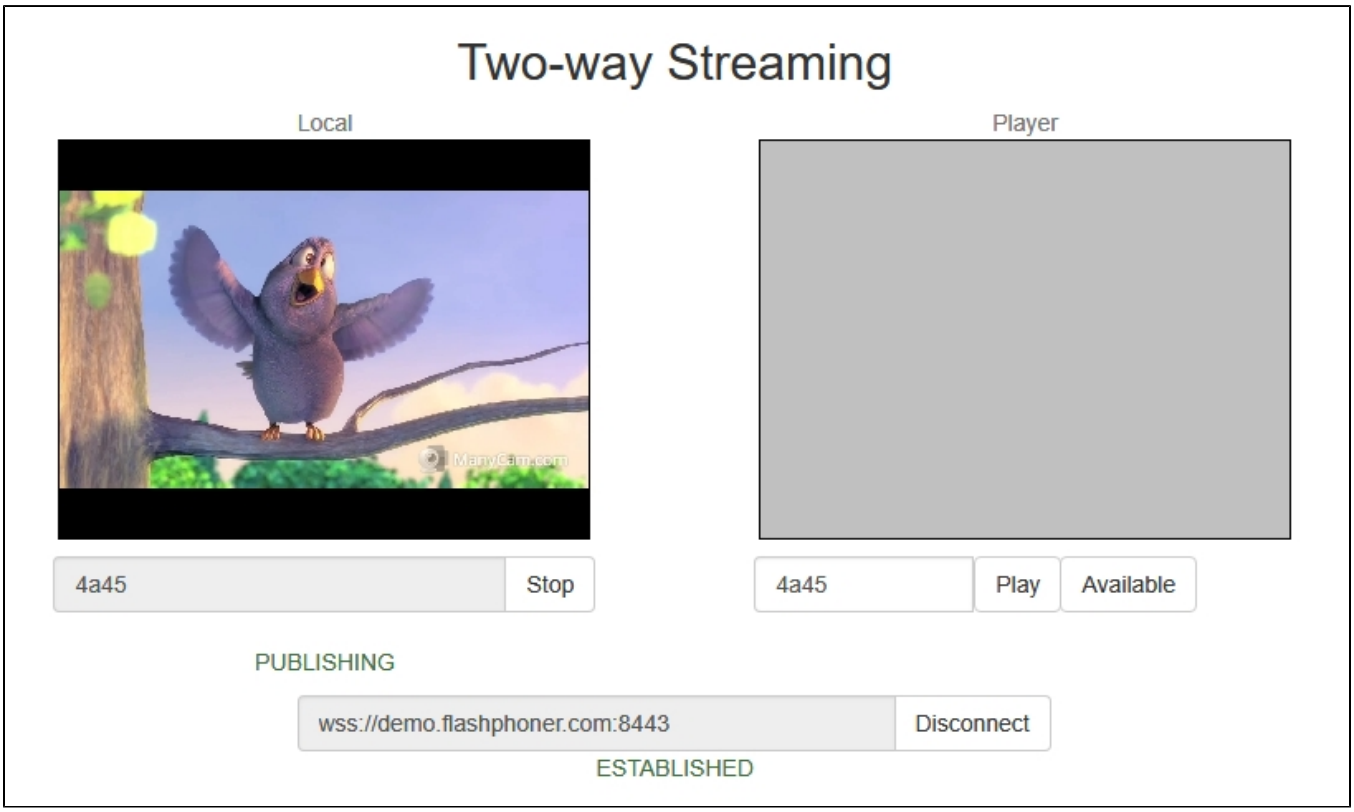
1. Для теста используем:

- демо-сервер demo.flashphoner.com;
- веб-приложениеTwo Way Streamingдля публикации потоков
- веб-приложение2 Playersдля воспроизведения потоков

2. Откройте веб-приложение Two Way Streaming, нажмите Connect, затем Publish. Скопируйте идентификатор первого потока из окна Play:

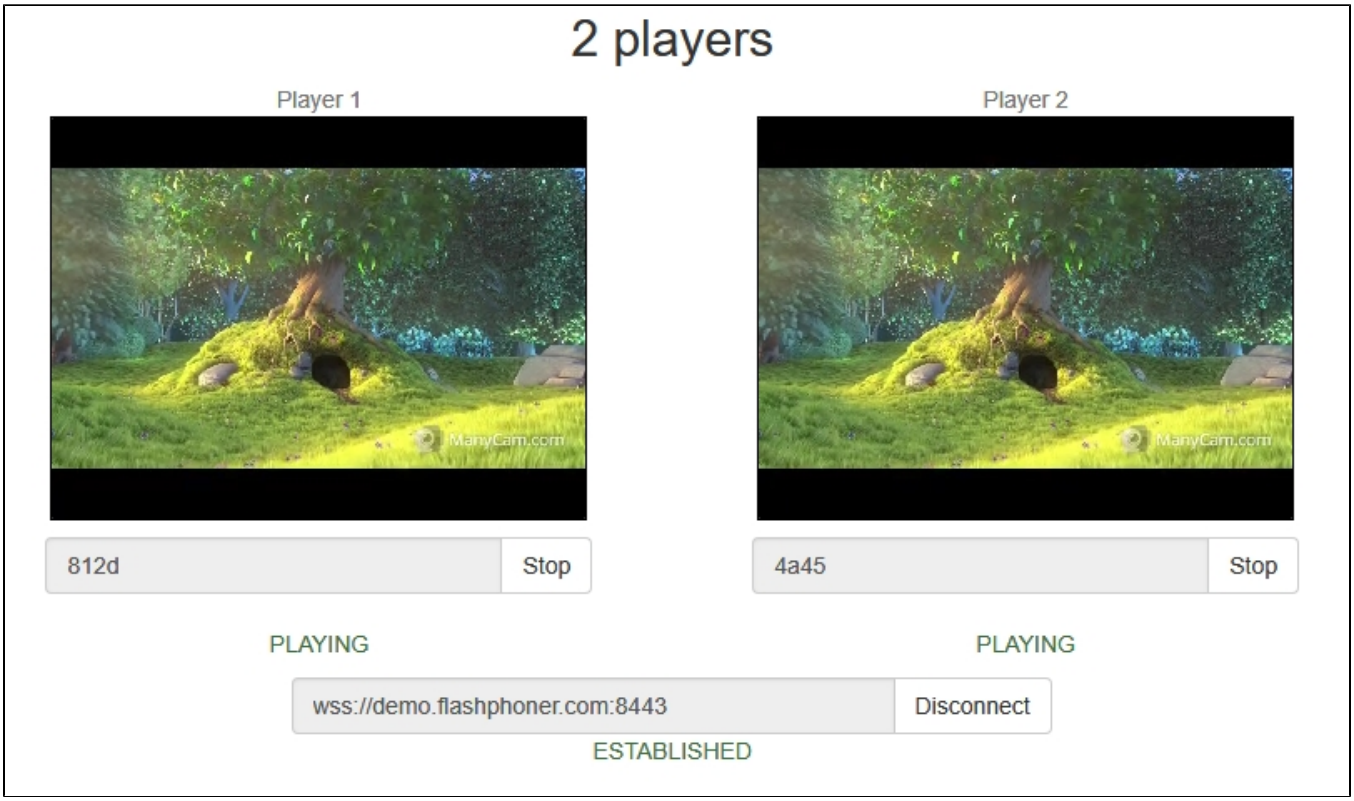


3. В другой вкладке откройте веб-приложение Two Way Streaming, нажмите Connect, затем Publish. Скопируйте идентификатор второго потока из окна Play:

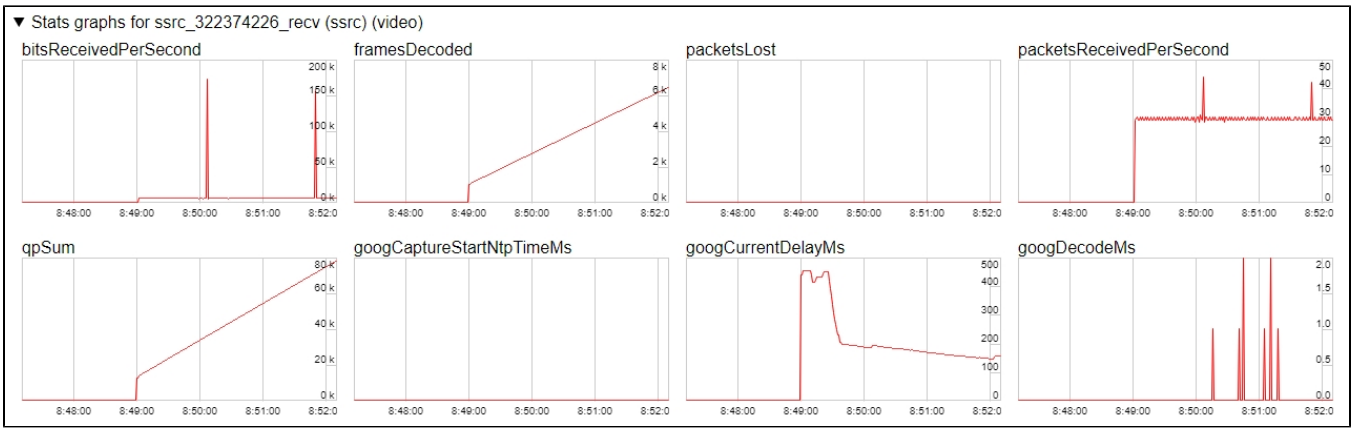


4. Откройте веб-приложение 2 Players, укажите идентификаторы первого (слева) и второго (справа) потоков:

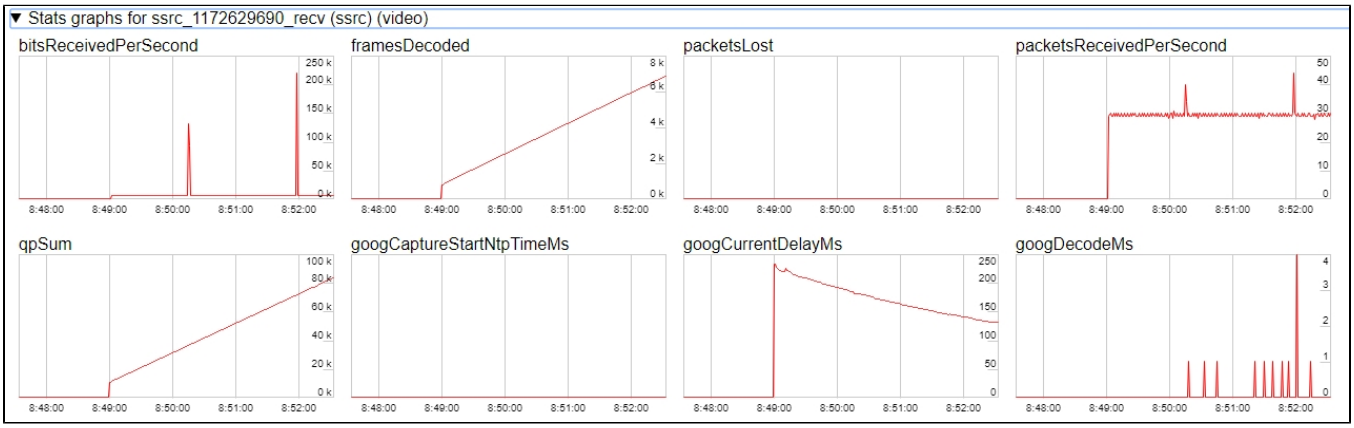
5. Нажмите Play под правым и левым окнами плеера:



6. Графики chrome://webrtc-internals для первого потока:



7. Графики chrome://webrtc-internals для второго потока:



Максимальное количество потоков, которые можно играть на одной странице

Максимальное количество потоков, которое можно воспроизвести на одной странице с приемлемым качеством, зависит от следующих параметров:

- параметры одного потока (разрешение и битрейт)
- пропускная способность канала от сервера до клиента
- используемый транспорт (UDP или TCP)
- производительность клиентского устройства

Например, для потока 1920x1080 с битрейтом 2 Мбит/с, с использованием TCP транспорта для канала пропускной способностью 30-35 Мбит/с экспериментально получены следующие максимально возможные значения:

- ПК на базе Intel Core i5 8 gen и новее, от 8 Гб RAM: до 15 потоков аудио+видео, или 6 потоков аудио+видео и 14 потоков только с аудио
- Флагманское Android/iOS устройство 2018 года и новее (Samsung S серии, Apple iPhone Pro): до 15 потоков аудио+видео, или 6 потоков аудио+видео и 14 потоков только с аудио
- Устройство среднего и ниже класса или устаревшее Android/iOS устройство (Nokia 5, Apple iPhone 7): до 6 потоков аудио+видео, или только с аудио

Таким образом, для потока 1920x1080 с битрейтом 2 Мбит/с оптимальным будет проигрывание не более чем 6 потоков на одной странице, чтобы воспроизведение работало у любых клиентов.

Рассмотрим случай вебинара: один поток 1920x1080 с битрейтом 2 Мбит/с и несколько потоков 640x360 с битрейтом 500 кбит/с. В тех же условиях:

- ПК на базе Intel Core i5 8 gen и новее, от 8 Гб RAM: до 25 потоков аудио+видео, или 6 потоков аудио+видео и 25 потоков только с аудио
- Флагманское Android/iOS устройство 2018 года и новее (Samsung S серии, Apple iPhone Pro): до 20 потоков аудио+видео, или 6 потоков аудио+видео и 25 потоков только с аудио
- Устройство среднего и ниже класса или устаревшее Android/iOS устройство 2017 года и новее: до 10 потоков аудио+видео, или 6 потоков аудио+видео и 15 потоков только с аудио

Таким образом, для вебинара с трансляцией рабочего стола и нескольких веб камер оптимальным будет проигрывание не более чем 10 потоков на одной странице, чтобы воспроизведение работало у любых клиентов.

Автозапуск воспроизведения

Примеры Player и Embed Player поддерживают автозапуск воспроизведения при помощи параметра

```
autoplay=true
```

например

```
https://hostname:8888/embed_player?urlServer=wss://hostname:
8443&streamName=stream1&autoplay=true&mediaProviders=WebRTC
```

Здесь

- hostname - имя WCS-сервера
- stream1 - имя потока на сервере

Особенности автозапуска воспроизведения в браузерах

Chrome

В последних версиях браузера Chrome (71 и выше) была изменена политика автозапуска воспроизведения контента на веб-страницах. Теперь для запуска воспроизведения видео необходимо, чтобы пользователь совершил какое-либо действие, например, нажатие на кнопку.

Это изменение влияет на создание аудиоконтекста, который необходим, чтобы работала регулировка громкости. Согласно новой политике, создание аудиоконтекста также требует действия от пользователя.

В связи с этим, в Chrome 71, а также в других браузерах на базе Chromium, поддерживающих изменение политики автозапуска, видео при автозапуске может проигрываться без звука. Для того, чтобы включить звук, пользователь должен переместить регулятор громкости в окне Embed Player.

Firefox и MacOS Safari

Как и в Chrome, автозапуск воспроизведения работает без звука, для включения звука требуется действие пользователя.

iOS Safari

Автозапуск воспроизведения работает, начиная с iOS 12.2. При этом политика автозапуска требует, чтобы пользователь переместил регулятор громкости для воспроизведения звука.

В версиях iOS 12.2-12.3 звук может не начать воспроизводиться и при движении регулятора громкости. В таких случаях необходимо повторно запустить воспроизведение видео, не обновляя страницу.

При включенном Low Power Mode автозапуск воспроизведения в iOS Safari не работает.

Тонкая настройка воспроизведения звука в iOS Safari

В случае воспроизведения и последующей публикации видео на одной странице (например, видеочат) в iOS Safari уровень звука для проигрываемого потока может меняться. Избежать этого можно двумя способами:

1. Запрашивать доступ к медиаустройствам при создании сессии перед проигрыванием

```
    Flashphoner.createSession({urlServer: url}).on(SESSION_STATUS.ESTABLISHED, function (session) {
        ...
        if (Browser.isSafariWebRTC() && Browser.isiOS() && Flashphoner.getMediaProviders()[0] === "WebRTC") {
            Flashphoner.playFirstVideo(localVideo, true, PRELOADER_URL).then(function () {
                Flashphoner.getMediaAccess(null, localVideo).then(function (disp) {
                });
            });
        }
        ...
    });
```

2. Через 1-1.5 секунды после получения статуса потока PLAYING, отключить и снова включить звук и/или видео

```
    session.createStream({
        name: streamName,
        display: remoteVideo
    }).on(STREAM_STATUS.PENDING, function (stream) {
        ...
    }).on(STREAM_STATUS.PLAYING, function (stream) {
        setStatus("#playStatus", stream.status());
        onPlaying(stream);
        if (Browser.isSafariWebRTC() && Browser.isiOS() && Flashphoner.getMediaProviders()[0] === "WebRTC") {
            setTimeout(function () {
                stream.muteRemoteAudio();
                stream.unmuteRemoteAudio();
            }, 1500);
        }
        ...
    }).play();
```

Воспроизведение стерео звука в браузере

Как и для публикации стерео звука в кодеке Opus, для проигрывания в браузере необходимо установить параметры кодека на стороне сервера

```
opus_formats = maxaveragebitrate=64000;stereo=1;sprop-stereo=1;
```

В этом случае браузер Firefox играет стерео звук без дополнительных настроек.

При проигрывании в браузере потока, захваченного из RTMP, RTSP или VOD источника, как правило. звук транскодируется в кодек Opus. По умолчанию, кодировщик Opus настроен на передачу речи и монофонического аудио. Для проигрывания стерео звука в браузере, необходимо повысить битрейт кодирования Opus до 60 кбит/с или выше

```
opus.encoder.bitrate=64000
```

Браузеры на основе Chrome

По умолчанию, из-за бага в движке браузер Chrome играет WebRTC поток со стерео звуком в кодеке Opus как моно. В связи с этим необходимы дополнительные настройки на стороне клиента, в зависимости от реализации.

С использованием Web SDK

В сборке Web SDK 0.5.28.2753.151 добавлена настройка для проигрывания стерео звука

```
constraints.audio.stereo=true
```

например

```
    session.createStream({
        name: streamName,
        display: remoteVideo,
        constraints: {
            audio: {
                stereo: true
            }
        }
        ...
    }).play();
```

С использованием Websocket API

Если в проекте используется только Websocket API, необходимо изменить параметры кодека Opus в исходящем (offer) SDP, непосредственно после его создания

```
        var connection = new RTCPeerConnection(connectionConfig, connectionConstraints);
        ...
        connection.createOffer(constraints).then(function (offer) {
          offer.sdp = offer.sdp.replace('minptime=10', 'minptime=10;stereo=1;sprop-stereo=1');
          connection.setLocalDescription(offer).then(function () {
              ...
          });
        });
```

Дополнительная задержка при воспроизведении видеопотока

В некоторых случаев при воспроизведении видеопотока необходимо добавить заданную задержку относительно трансляции. В сборке WebSDK 0.5.28.2753.142, поставляемой вместе со сборкой WCS5.2.708, с этой целью добавлена опция playoutDelay:

```
    session.createStream({
        name: streamName,
        display: remoteVideo,
        playoutDelay: 10
    }).on(STREAM_STATUS.PENDING, function (stream) {
        ...
    }).play();
```

Задержка задается в секундах.

Данная опция поддерживается только в браузерах на основе Chromium,которые поддерживают атрибут

```
partial interface RTCRtpReceiver {
  attribute double? playoutDelayHint;
};
```

Задержка не применяется к аудио дорожкам в потоке, а также к аудио потокам без видео.

Известные проблемы

1. Возможный баг в браузере Safari на iOS приводит к фризам при воспроизведении WebRTC

Симптомы: останавливается воспроизведение видео, звуковая дорожка при этом может продолжать играть, для восстановления требуется перезагрузка страницы либо перезапуск браузера.

Решение:
а) включить транскодер на сервере, указав в файлеflashphoner.properties

```
disable_streaming_proxy=true
```

б) при воспроизведении потока с iOS Safari явно указать ширину и высоту, например:

```
session.createStream({constraints:{audio:true,video:{width:320,height:240}}}).play();
```

2. При публикации потока по RTMP и воспроизведении в браузере по WebRTC вместо аудиокодека Opus используется PCMU

Симптомы: в chrome://webrtc-internals отображается кодек PCMU

Решение: отключить алгоритм избегания транскодинга (Avoid Transcoding Alhorithm) при помощи опции в файлеflashphoner.properties

```
disable_rtc_avoid_transcoding_alg=true
```

3.При публикации потока при помощи Flash Streaming, воспроизведении этого потока в iOS Safari по WebRTC и одновременной публикации потока по WebRTC из Safari перестает воспроизводиться звук.

Симптомы:

а) Публикация потока stream1 из приложения Flash Streaming в браузере Chrome под Windows
б) Воспроизведение потока stream1 на iOS Safari в приложении Two Way Streaming. Звук и видео воспроизводятся нормально.
в) Публикация потока из iOS Safari в приложении Two Way Streaming. Воспроизведение звука пропадает.
г) Остановка публикации в iOS Safari. Воспроизведение звука восстанавливается.

Решение: отключить алгоритм избегания транскодинга (Avoid Transcoding Alhorithm) на сервере при помощи опции в файлеflashphoner.properties

```
disable_rtc_avoid_transcoding_alg=true
```

4. При публикации потока по RTMP и отключении Keep Alive для всех протоколов воспроизведение в браузере по WebRTC останавливается по истечению WebSocket-таймаута

Симптомы: при публикации потока по RTMP воспроизведение в браузере по WebRTC останавливается без явного указания ошибки

Решение: если Keep Alive отключен для всех протоколов при помощи настройкив файлеflashphoner.properties

```
keep_alive.algorithm=NONE
```

необходимо также отключить таймаут на чтение WebSocket настройкой

```
ws_read_socket_timeout=false
```

5. Кодек G722 не работает в браузере Edge

Симптомы: поток со звуком G722 не воспроизводится в браузере Edge или воспроизводится без звука, с фризами

Решение: использовать другой кодек или другой браузер. В случае, если использование другого браузера невозможно, исключить кодек G722 при помощи настройки

```
codecs_exclude_streaming=g722,telephone-event
```

6. Некоторые браузеры, основанные на Chromium, например Opera, Yandex, в зависимости от версии и ОС не поддерживают кодек H264

Симптомы: не работает публикация, не работает воспроизведение частично (только звук) или полностью при трансляции WebRTC потока H264

Решение: разрешить поддержку vp8 на стороне сервера

```
codecs=opus,...,h264,vp8,...
```

исключить H264 для трансляции или воспроизведения на стороне клиента

```
publishStream = session.createStream({
    ...
    stripCodecs: "h264,H264"
}).on(STREAM_STATUS.PUBLISHING, function (publishStream) {
    ...
});
publishStream.publish();
```

Необходимо отметить, что при трансляции H264 потока и воспроизведении его как VP8 на сервере включаетсятранскодинг.

7. Если в настройках браузера Chrome 71 и выше для сайта разрешен Flash, при воспроизведении по WebRTC в консоль браузера может выводиться ошибка "Cross-origin content must have visible size large than 400 x 300 pixels, or it will be blocked"

Симптомы: при воспроизведении по WebRTC в консоль браузера Chrome выводится сообщение "Cross-origin content must have visible size large than 400 x 300 pixels, or it will be blocked", при этом воспроизведение работает

Решение: использовать WebSDK без поддержки Flash

```
flashphoner-no-flash.js
```

8. При большом количестве подписчиков наблюдаются задержки в воспроизводимом потоке

Симптомы: при большом количестве подписчиков (более 200 для потока 720p) наблюдаются задержки и фризы видео, аудио продолжает играть

Решение: включить многопоточную отсылку кадров подписчикам

```
streaming_distributor_video_proxy_pool_enabled=true
```

Эта настройка действует только на потоки, которые не транскодируются на данном сервере

9. При проигрывании потока в iOS Safari по умолчанию звук идет в голосовой динамик

Симптомы: при проигрывании WebRTC потока, например, при входе в чат-комнату с iOS устройства тихий звук

Решение: отключить и снова включить звук при старте проигрывания потока, например

```
    stream = session.createStream(options).on(STREAM_STATUS.PLAYING, function (stream) {
        stream.muteRemoteAudio();
        stream.unmuteRemoteAudio();
    }).play();
```

10. При проигрывании потока в iOS Safari резко возрастает нагрузка на сервер, при условии использования JDK 11

Симптомы: при подключении подписчика с iOS Safari резко возрастает нагрузка на процессор сервера

Решение: обновить JDK до одной из рекомендованных версий: 8, 12, 14.

11. При воспроизведении двух и более потоков на одной странице в браузере Chrome на некоторых устройствах Xiaomi с MIUI 12 картинка в первом потоке подергивается

Симптомы: при воспроизведении двух потоков на одной странице в примере2 Players,картинка первого потока подергивается, мелькает изображение второго потока

Решение:

a) использовать на устройстве Xioami MIUI 11

b) использовать микшер для проигрывания двух и более потоков на одной странице

12. В Safari 16 переключение в полноэкранный режим работает только в случае, если для HTML5 видео элемента включены стандартные элементы управления

Симптомы: при переключении в полноэкранный режим видео не отображается, аудио продолжает играть, при нескольких переключениях подряд страница может зависнуть

Решение: обновить Web SDK до сборки 2.0.224 и включить стандартные элементы управления в плеере при помощи опции `useControls` :

```
function playStream(session) {
    var streamName = $('#streamName').val();
    var options = {
        name: streamName,
        display: remoteVideo,
        useControls: true
    };
    ...
    stream = session.createStream(options).on(STREAM_STATUS.PENDING, function (stream) {
        ...
    });
    stream.play();
}
```