

# Микширование потоков

- [Описание](#)
  - Поддерживаемые протоколы входных потоков
  - Возможности управления выходным потоком
  - Автоматическое создание микшера при публикации потока
  - Схема работы
- [REST-вызовы](#)
  - REST-вызовы и статусы ответа
  - Параметры
  - Отправка REST-запроса к WCS-серверу
- [Настройка](#)
  - Настройка автоматического создания микшера при публикации потока
  - Настройка микширования аудио и видео
  - Буферизация выходного потока микшера
  - Управление битрейтом выходного потока микшера
  - Управление звуковой дорожкой выходного потока микшера
  - Собственный lossless видеопроцессор для входящих потоков
  - Управление размещением картинок в выходном потоке микшера
    - Реализация собственного варианта размещения картинок
- [Краткое руководство по тестированию](#)
- [Последовательность выполнения операций \(Call flow\)](#)
- [Известные проблемы](#)

## Описание

WCS позволяет микшировать потоки активных трансляций. Выходной поток микшера может быть записан, [воспроизведен](#) или [ретранслирован](#) по любой из технологий, поддерживаемых WCS.

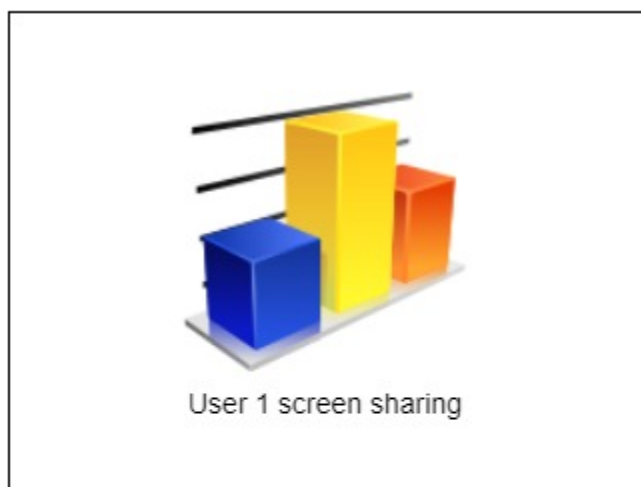
Микширование управляется при помощи [настроек](#) и [REST API](#).

## Поддерживаемые протоколы входных потоков

- WebRTC
- RTMP
- RTSP

## Возможности управления выходным потоком

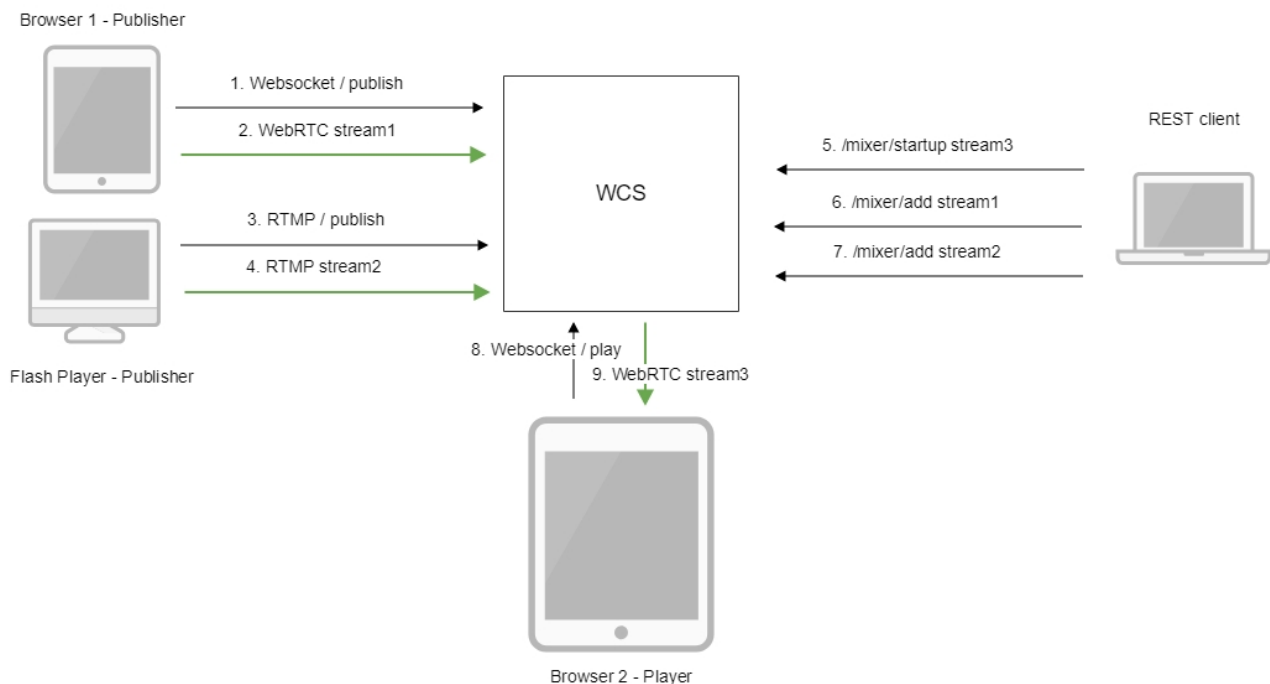
Микшер позволяет задать размещение видеопотоков в выходном кадре. Поток с определенным именем (по умолчанию desktop) рассматривается, как демонстрация экрана (screen sharing), и размещается в центре кадра:



## Автоматическое создание микшера при публикации потока

Если в имени публикуемого RTMP-потока есть символ '#', сервер рассматривает все, что после данного символа, как имя микшера, который будет создан автоматически при публикации потока. Например, для потока user1#room1 будет создан микшер room1, и поток будет в этот микшер добавлен. В имени потока может быть указано и ключевое слово демонстрации экрана, например user1#room1#desktop

## Схема работы



1. Браузер соединяется с сервером по протоколу Websocket и отправляет команду publish.
2. Браузер отправляет WebRTC поток stream1 на сервер.
3. Flash Player устанавливает соединение по RTMP и отправляет команду publish.
4. Flash Player отправляет RTMP поток stream2 на сервер.
5. REST-клиент создает микшер с выходным потоком stream3 запросом /mixer/startup
6. REST-клиент добавляет к микшеру поток stream1
7. REST-клиент добавляет к микшеру поток stream2
8. Второй браузер устанавливает соединение по Websocket и отправляет команду play.
9. Второй браузер получает WebRTC аудиопоток stream3 и воспроизводит этот поток на странице.

## REST-вызовы

REST-запрос должен быть HTTP/HTTPS POST запросом в таком виде:

- HTTP: <http://streaming.flashphoner.com:8081/rest-api/mixer/startup>
- HTTPS: <https://streaming.flashphoner.com:8444/rest-api/mixer/startup>

Здесь:

- streaming.flashphoner.com- адрес WCS-сервера
- 8081 - стандартный REST / HTTP порт WCS-сервера
- 8444- стандартный HTTPS порт
- rest-api- обязательный префикс
- mixer/startup- используемый REST-вызов

## REST-вызовы и статусы ответа

| REST-метод         | Пример тела REST-запроса                                                                          | Пример ответа | Статусы ответа                                     | Описание                                                          |
|--------------------|---------------------------------------------------------------------------------------------------|---------------|----------------------------------------------------|-------------------------------------------------------------------|
| /mixer<br>/startup | <pre>{   "uri": "mixer://mixer1",   "localStreamName":   "stream3",   "hasVideo": "false" }</pre> |               | 200 - OK<br>409 - Conflict<br>500 - Internal error | Создает микшер, для которого публикуется поток с указанным именем |

|                        |                                                                             |                                                                                                                                                                                                                                                                                                                                                        |                                                                                                 |                                                 |
|------------------------|-----------------------------------------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-------------------------------------------------------------------------------------------------|-------------------------------------------------|
| /mixer/add             | <pre>{   "uri": "mixer://mixer1",   "remoteStreamName":   "stream1" }</pre> |                                                                                                                                                                                                                                                                                                                                                        | 200 - OK<br><br>404 - Mixer not found<br><br>404 - Stream not found<br><br>500 - Internal error | Добавить RTMP-поток в микшер                    |
| /mixer/remove          | <pre>{   "uri": "mixer://mixer1",   "remoteStreamName":   "stream1" }</pre> |                                                                                                                                                                                                                                                                                                                                                        | 200 - OK<br><br>404 - Mixer not found<br><br>404 - Stream not found<br><br>500 - Internal error | Убрать RTMP-поток из микшера                    |
| /mixer/find_all        |                                                                             | <pre>{   "localMediaSessionId":   "ce92b134-2468-4460-8d06-1ea3c5aabace",   "remoteMediaSessionId": null,   "localStreamName": "mixer1",   "remoteStreamName": null,   "uri": "mixer://mixer1",   "status": "PROCESSED_LOCAL",   "mediaSessions": [     "95bf2be8-f459-4f62-9a7f-c588f33e0ad3",     "693781de-cada-4589-abe1-c3ee55c66901"   ] }</pre> | 200 - OK<br><br>404 - Not found<br><br>500 - Internal error                                     | Найти все микшеры                               |
| /mixer/terminate       | <pre>{   "uri": "mixer://mixer1" }</pre>                                    |                                                                                                                                                                                                                                                                                                                                                        | 200 - OK<br><br>404 - Not found<br><br>500 - Internal error                                     | Завершить работу микшера                        |
| /stream/startRecording | <pre>{   "mediaSessionId":   "23d07fa1-3c74-4d6f-a0de-9b4bf83ce665" }</pre> |                                                                                                                                                                                                                                                                                                                                                        | 200 - OK<br><br>404 - Not found<br><br>500 - Internal error                                     | Начать запись потока в указанной медиасессии    |
| /stream/stopRecording  | <pre>{   "mediaSessionId":   "23d07fa1-3c74-4d6f-a0de-9b4bf83ce665" }</pre> |                                                                                                                                                                                                                                                                                                                                                        | 200 - OK<br><br>404 - Not found<br><br>500 - Internal error                                     | Завершить запись потока в указанной медиасессии |

Параметры

| Имя параметра | Описание                         | Пример         |
|---------------|----------------------------------|----------------|
| uri           | Уникальный идентификатор микшера | mixer://mixer1 |

|                  |                                                   |                                                               |
|------------------|---------------------------------------------------|---------------------------------------------------------------|
| localStreamName  | Имя выходного потока микшера                      | stream3                                                       |
| hasVideo         | Микшировать видео (true) или только аудио (false) | false                                                         |
| remoteStreamName | Имя потока, добавляемого в микшер                 | stream1<br>rtmp://rtmp.flashphoner.com:1935/live/rtmp_stream1 |
| mediaSessionId   | Идентификатор медиасессии                         | ce92b134-2468-4460-8d06-1ea3c5aabace                          |
| status           | Статус потока                                     | PROCESSED_LOCAL                                               |

## Отправка REST-запроса к WCS-серверу

Для отправки REST-запроса к WCS-серверу необходимо использовать [REST-клиент](#).

## Настройка

Основные настройки микширования задаются при помощи следующих параметров файла настроек [flashphoner.properties](#)

| Параметр                                  | Значение по умолчанию | Описание                                                                                        |
|-------------------------------------------|-----------------------|-------------------------------------------------------------------------------------------------|
| mixer_video_desktop_layout_inline_padding | 10                    | Расстояние между окнами видеопотоков в нижней строке (под окном демонстрации экрана)            |
| mixer_video_desktop_layout_padding        | 30                    | Расстояние между окном видеопотока демонстрации экрана и нижней строкой (остальные видеопотоки) |
| mixer_video_enabled                       | true                  | Включает (по умолчанию) или отключает микширование видео                                        |
| mixer_video_grid_layout_middle_padding    | 10                    | Расстояние между окнами видеопотоков в одной строке (при отсутствии демонстрации экрана)        |
| mixer_video_grid_layout_padding           | 30                    | Расстояние между строками видеопотоков (при отсутствии демонстрации экрана)                     |
| mixer_video_height                        | 720                   | Высота изображения выходного потока микшера                                                     |
| mixer_video_layout_desktop_key_word       | desktop               | Ключевое слово для потока демонстрации экрана (screen sharing)                                  |
| mixer_video_width                         | 1280                  | Ширина изображения выходного потока микшера                                                     |
| record_mixer_streams                      | false                 | Включает или отключает (по умолчанию) запись всех выходных потоков микшера                      |

## Настройка автоматического создания микшера при публикации потока

Для того, чтобы включить возможность автоматического создания микшера для потоков, содержащих в имени символ '#', необходимо, чтобы приложение, которое обрабатывает входящие потоки, зарегистрировало обработчик 'com.flashphoner.server.client.handler.wcs4.FlashRoomRecordingStreamingHandler'. Регистрация обработчика производится при помощи [интерфейса командной строки](#). Например, для приложения flashStreamingApp, используемого для публикации входящих RTMP потоков, это делается командой

```
update app -m com.flashphoner.server.client.handler.wcs4.FlashRoomRecordingStreamingHandler -c com.flashphoner.server.client.handler.wcs4.FlashStreamingCallbackHandler flashStreamingApp
```

Подробнее об управлении приложениями из командной строки WCS-сервера можно узнать [здесь](#).

## Настройка микширования аудио и видео

По умолчанию микшируются видео и аудиопотоки. Если необходимо микшировать только аудио, необходимо указать это при создании микшера

```
{
  "uri": "mixer://mixer1",
  "localStreamName": "stream3",
  "hasVideo": "false"
}
```

Если необходимо отключить микширование видео для всех потоков, необходимо указать параметр в файле [flashphoner.properties](#)

```
mixer_video_enabled=false
```

При этом видео может быть включено для определенного микшера при его создании.

## Буферизация выходного потока микшера

В некоторых случаях необходима буферизация выходного потока микшера. Эта функция включается настройкой в файле [flashphoner.properties](#)

```
mixer_out_buffer_enabled=true
```

Размер буфера задается в миллисекундах настройкой

```
mixer_out_buffer_start_size=400
```

В данном случае размер буфера составит 400 мс.

Период отправки данных потока из буфера в миллисекундах указывается настройкой

```
mixer_out_buffer_polling_time=20
```

В данном случае период составляет 20 мс.

## Управление битрейтом выходного потока микшера

Если в качестве транскодера используется кодек OpenH264, появляется возможность управлять битрейтом выходного потока микшера при помощи настройкой в файле [flashphoner.properties](#)

```
mixer_video_bitrate_kbps=2000
```

По умолчанию, битрейт выходного потока установлен в 2 Мбит/с. При недостаточной пропускной способности канала между сервером и зрителем битрейт может быть понижен, например

```
encoder_priority=OPENH264  
mixer_video_bitrate_kbps=1500
```

Если качество картинки с битрейтом по умолчанию низкое, присутствуют искажения, рекомендуется увеличить битрейт выходного потока микшера до 3-5 Мбит/с

```
encoder_priority=OPENH264  
mixer_video_bitrate_kbps=5000
```

## Управление звуковой дорожкой выходного потока микшера

По умолчанию, звуковая дорожка в выходном потоке микшера кодируется в Opus с частотой дискретизации 48 кГц. Эти параметры могут быть изменены при помощи настроек в файле [flashphoner.properties](#). Например, для использования выходного потока в SIP, можно установить следующие значения:

```
audio_mixer_output_codec=pcma  
audio_mixer_output_sample_rate=8000
```

В этом случае звук будет кодироваться в PCMA (alaw) с частотой дискретизации 8 кГц.

## Собственный lossless видеопроцессор для входящих потоков

Для обработки входящих потоков микшера, например, если необходима дополнительная буферизация или синхронизация аудио и видео дорожек, может быть использован собственный lossless видеопроцессор. Этот видеопроцессор включается настройкой в файле [flashphoner.properties](#)

```
mixer_lossless_video_processor_enabled=true
```

Максимальный размер буфера микшера в миллисекундах устанавливается настройкой

```
mixer_lossless_video_processor_max_mixer_buffer_size_ms=200
```

По умолчанию, размер буфера микшера составляет 200 мс. При его заполнении, видеопроцессор использует свой собственный буфер и ожидает освобождения буфера микшера. Периодичность проверки буфера микшера устанавливается в миллисекундах настройкой

```
mixer_lossless_video_processor_wait_time_ms=20
```

По умолчанию, периодичность проверки составляет 20 мс.

Необходимо отметить, что использование lossless видеопроцессора может вносить задержку в трансляции в реальном времени.

При использовании lossless видеопроцессора, чтобы освободить ресурсы, занятые микшером, нужно принудительно остановить микшер при помощи REST запроса `/mixer/terminate`, либо остановить все входящие в микшер потоки, в этом случае микшер остановится по истечении времени, заданного в миллисекундах настройкой

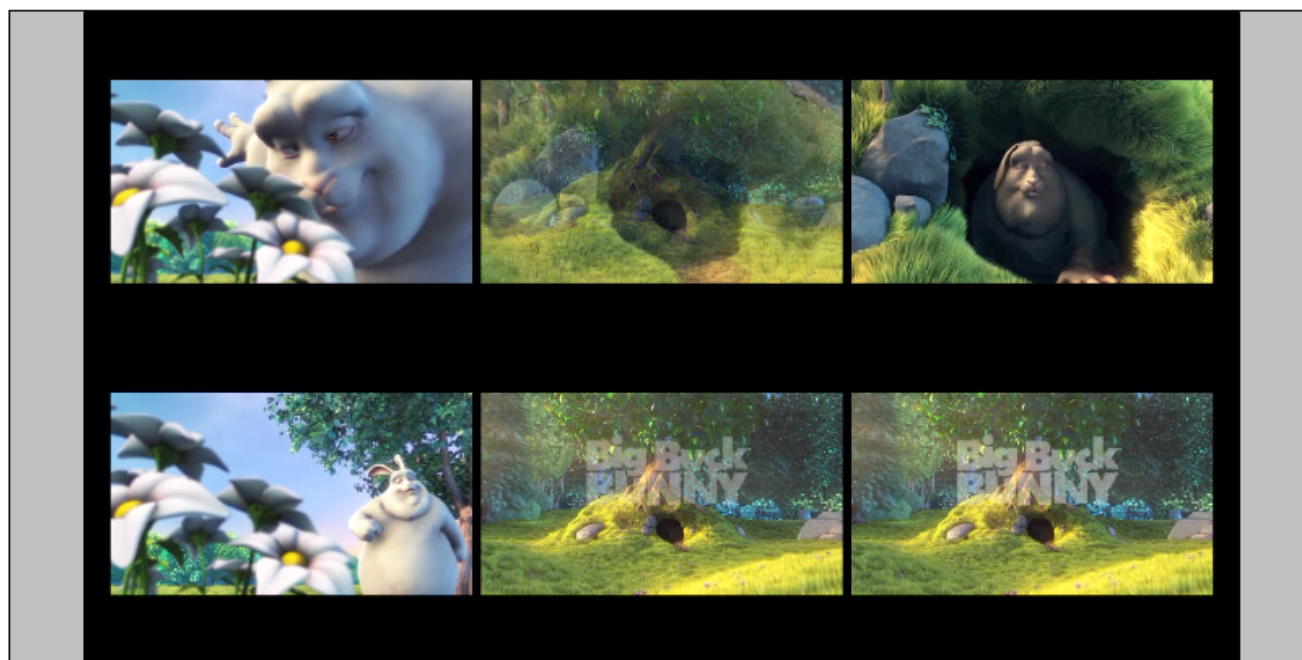
```
mixer_idle_timeout=60000
```

По умолчанию, при отсутствии входящих потоков, микшер останавливается через 60 секунд.

## Управление размещением картинок в выходном потоке микшера

По умолчанию, микшер предусматривает три варианта размещения картинок входных потоков в выходном потоке:

### 1. Картинки сеткой



Этот вариант включается настройкой в файле [flashphoner.properties](#)

```
mixer_layout_class=com.flashphoner.media.mixer.video.presentation.GridLayout
```

### 2. Картинки сеткой с минимальным расстоянием между ними

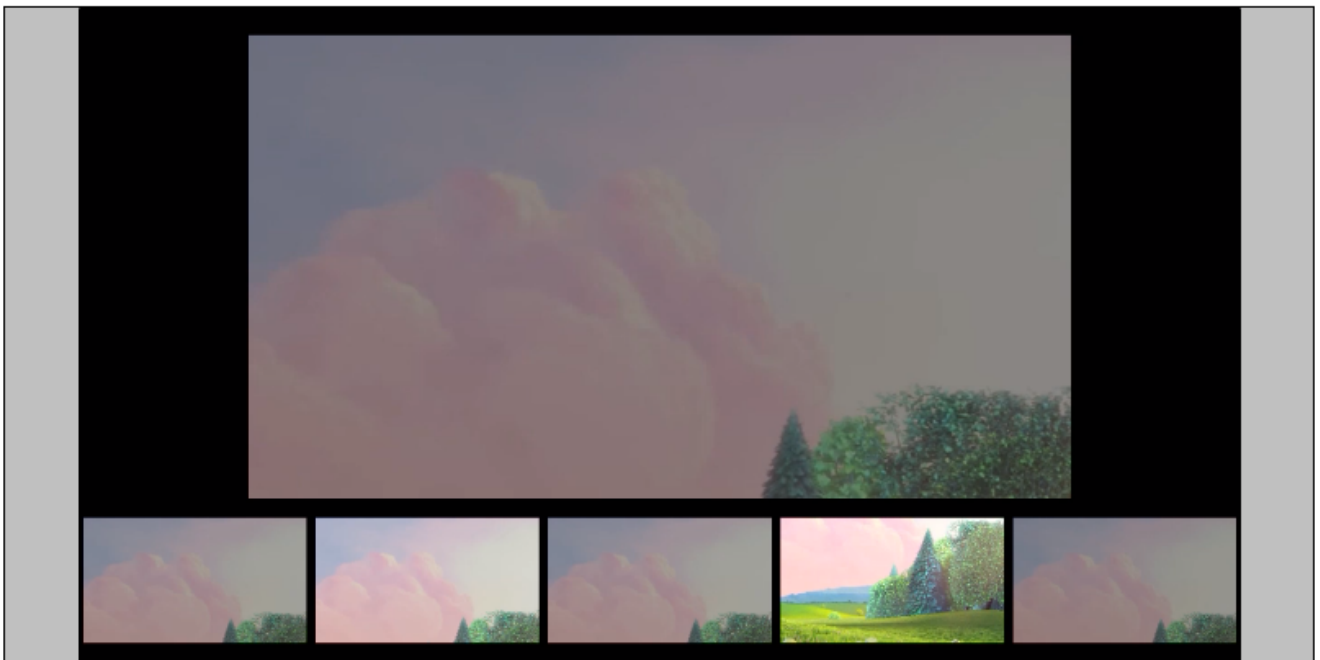


Этот вариант включается настройкой

```
mixer_layout_class=com.flashphoner.media.mixer.video.presentation.CenterNoPaddingGridLayout
```

и обеспечивается только для входных потоков одинакового разрешения, с одинаковым соотношением сторон

### 3. Демонстрация экрана (screen sharing) в центре кадра



Этот вариант включается, если на вход микшеру подается поток с именем, указанным в настройке

```
mixer_video_layout_desktop_key_word=desktop
```



По умолчанию, для потока демонстрации экрана используется имя desktop.

## Реализация собственного варианта размещения картинок

Для более тонкой настройки размещения картинок в выходном потоке микшера необходимо разработать класс на языке Java, реализующий интерфейс `IVideoMixerLayout`, например

### TestLayout.java

```
package com.flashphoner.mixerlayout;

import com.flashphoner.sdk.media.IVideoMixerLayout;
import com.flashphoner.sdk.media.YUVFrame;
import java.awt.*;
import java.util.ArrayList;

public class TestLayout implements IVideoMixerLayout {

    private static final int PADDING = 5;

    @Override
    public Layout[] computeLayout(YUVFrame[] yuvFrames, String[] strings, int canvasWidth, int canvasHeight) {
        ArrayList<IVideoMixerLayout.Layout> layout = new ArrayList<>();
        for (int c = 0; c < yuvFrames.length; c++) {
            Point prevPoint = new Point();
            Dimension prevDimension = new Dimension(canvasWidth, canvasHeight);
            if (layout.size() > 0) {
                prevPoint.setLocation(layout.get(c-1).getPoint());
                prevDimension.setSize(layout.get(c-1).getDimension());
            }
            Point currentPoint = new Point((int) (prevPoint.getX()+prevDimension.getWidth()+PADDING),
                                           (int) (prevPoint.getY()+prevDimension.getHeight()));
            layout.add(new IVideoMixerLayout.Layout(currentPoint, new Dimension(canvasWidth/yuvFrames.length,
                                                                                canvasHeight/yuvFrames.length), yuvFrames[c]));
        }
        return layout.toArray(new IVideoMixerLayout.Layout[layout.size()]);
    }
}
```

Затем следует скомпилировать класс в байт-код. Для этого создаем дерево каталогов, соответствующее названию пакета написанного класса

```
mkdir -p com/flashphoner/mixerlayout
```

и выполняем команду

```
javac -cp /usr/local/FlashphonerWebCallServer/lib/tbs-flashphoner.jar ./com/flashphoner/mixerlayout/TestLayout.java
```

Теперь упакуем скомпилированный код в jar-файл

```
jar -cf testlayout.jar ./com/flashphoner/mixerlayout/TestLayout.class
```

и скопируем его в каталог, где размещены библиотеки WCS сервера

```
cp testlayout.jar /usr/local/FlashphonerWebCallServer/lib
```

Для того, чтобы использовать разработанный класс, необходимо указать его в настройках файла [flashphoner.properties](#)

```
mixer_layout_class=com.flashphoner.mixerlayout.TestLayout
```

и перезапустить WCS.

Выходной поток микшера для приведенного примера и трех входящих потоков будет выглядеть так:



## Краткое руководство по тестированию

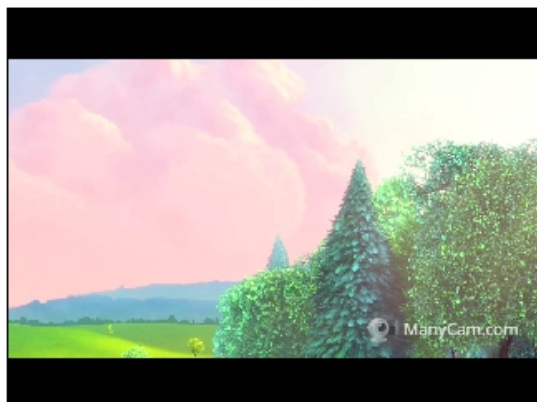
1. Для теста используем:

- демо-сервер [demo.flashphoner.com](http://demo.flashphoner.com);
- браузер Chrome и [REST-клиент](#) для отправки запросов на сервер;
- веб-приложение [Two Way Streaming](#) для публикации входных потоков микшера;
- веб-приложение [Player](#) для воспроизведения выходного потока микшера.

2. Откройте страницу веб-приложения Two Way Streaming. Опубликуйте поток с именем stream1:

## Two-way Streaming

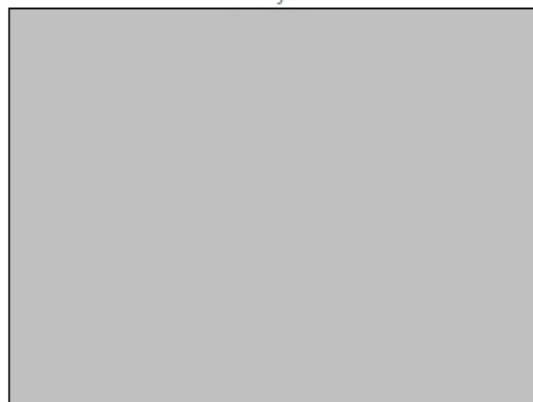
Local



stream1

Stop

Player



3fb8

Play

Available

PUBLISHING

wss://mixer-demo.flashphoner.com:8443

Disconnect

ESTABLISHED

3. В другой вкладке откройте страницу веб-приложения Two Way Streaming. Опубликуйте поток с именемdesktop:

## Two-way Streaming

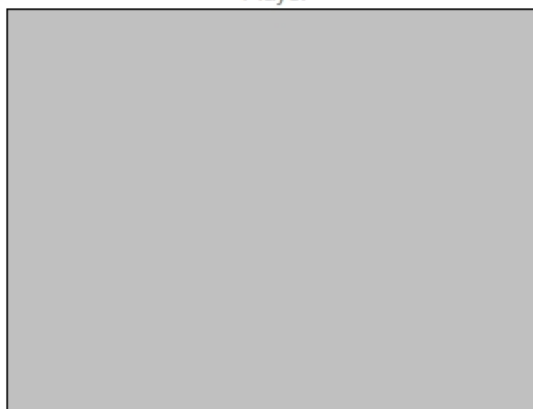
Local



desktop

Stop

Player



b71d

Play

Available

PUBLISHING

wss://mixer-demo.flashphoner.com:8443

Disconnect

ESTABLISHED

4. Откройте REST-клиент. Отправьте запрос /mixer/startup, указав в параметрах URI микшер mixer://mixer1 и имя выходного потока stream3:

Method

POST

Request URL

http://mixer-demo.flashphoner.com:9091/rest-api/mixer/startup

SEND

Parameters

Headers

Body

Variables

Body content type

application/json

Editor view

Raw input

FORMAT JSON

MINIFY JSON

```
{
  "uri": "mixer://mixer1",
  "localStreamName": "stream3"
}
```

200 OK

411.20 ms

DETAILS

5. Отправьте запрос /mixer/add, указав в параметрах URI микшер mixer://mixer1 и имя входного потока stream1:

Method

POST

Request URL

http://mixer-demo.flashphoner.com:9091/rest-api/mixer/add

SEND

Parameters ^

Headers

Body

Variables

Body content type

application/json

Editor view

Raw input

FORMAT JSONMINIFY JSON

```
{
  "uri": "mixer://mixer1",
  "remoteStreamName": "stream1"
}
```

200 OK396.40 ms

DETAILS v

6. Откройте веб-приложение Player, укажите в поле Stream имя выходного потока микшерastream3и нажмите Start:

## Player



**WCS URL**

wss://mixer-demo.flashphoner.co

**Stream**

stream3

**Volume**



7. Отправьте запрос /mixer/add, указав в параметрах URI микшератmixer://mixer1и имя входного потокаdesktop:

Method

POST

Request URL

http://mixer-demo.flashphoner.com:9091/rest-api/mixer/add

SEND

Parameters ^

Headers

Body

Variables

Body content type

application/json

Editor view

Raw input

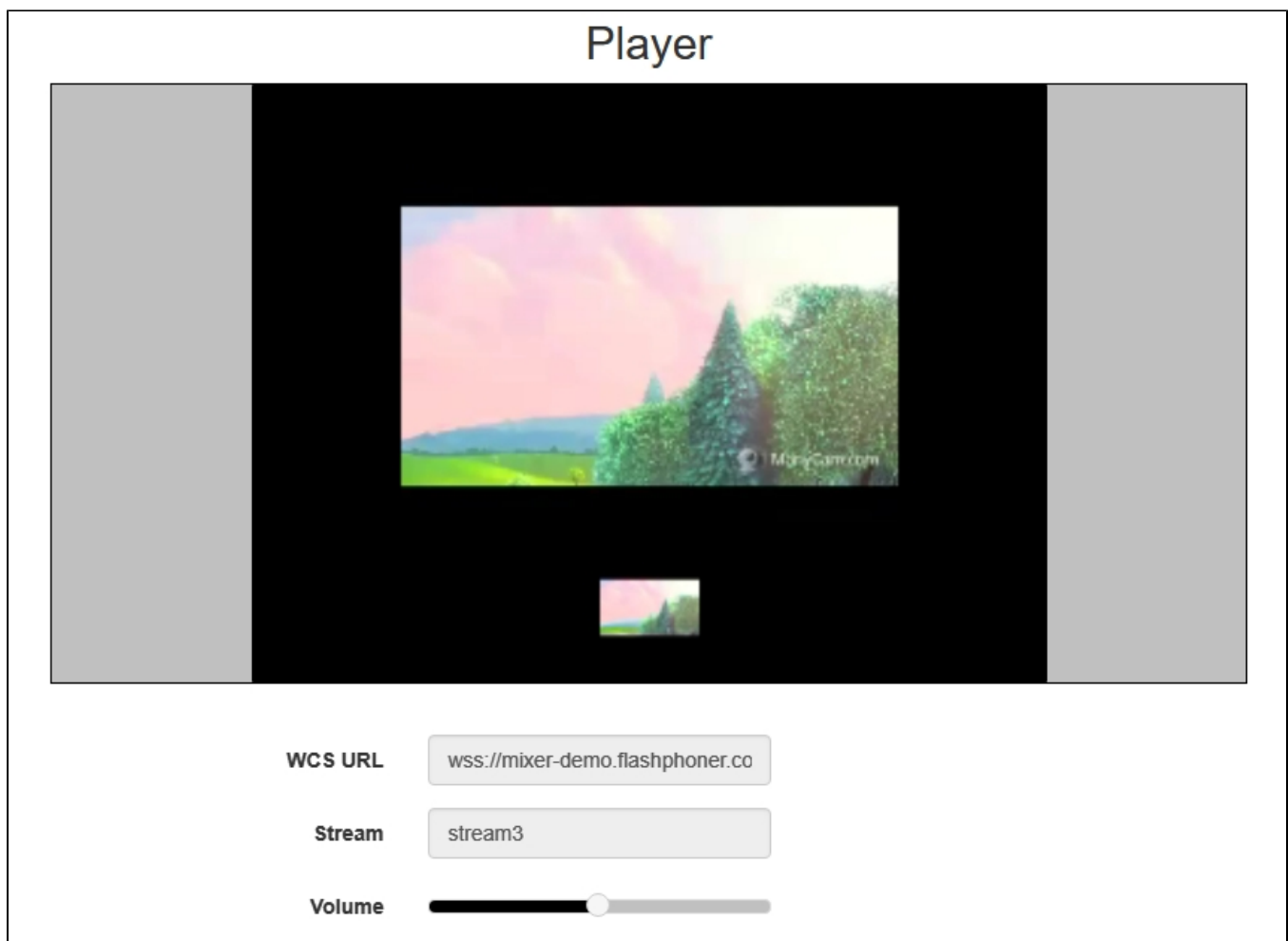
FORMAT JSONMINIFY JSON

```
{
  "uri": "mixer://mixer1",
  "remoteStreamName": "desktop"
}
```

200 OK381.40 ms

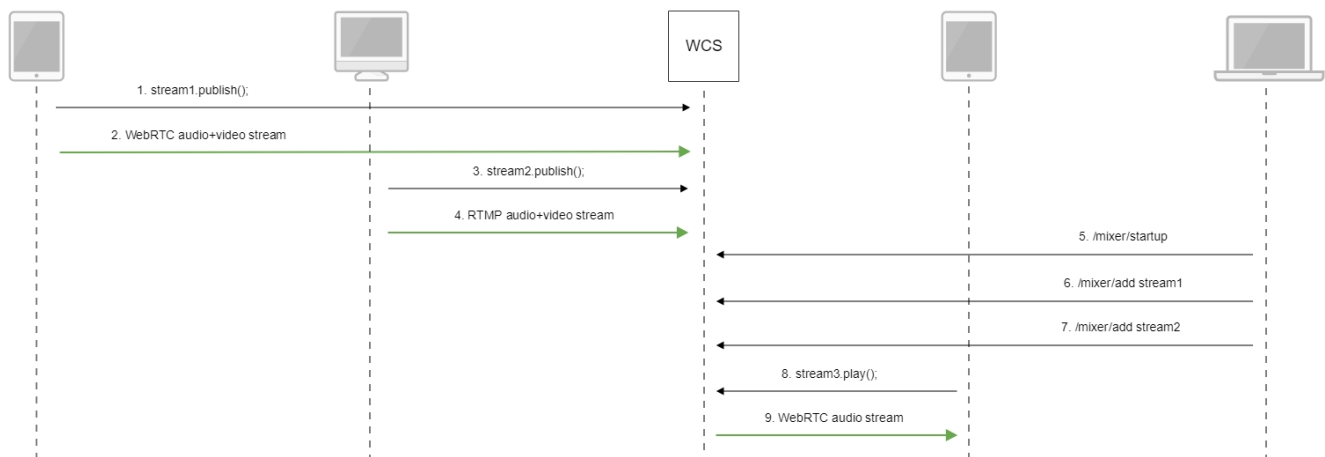
DETAILS v

8. В выходном потоке микшера отобразится потокdesktop, имитирующий демонстрацию экрана, и потокstream1:



## Последовательность выполнения операций (Call flow)

Ниже описана последовательность вызовов при использовании микшера.



1. Публикация **WebRTC-потока** stream1
2. Отправка **RTMP-потока** на сервер
3. Публикация **RTMP-потока** stream2



4. Отправка RTMP-потока на сервер

5. Отправка запроса /mixer/startup на создание микшера mixer://stream3 с выходным потоком stream3

```
http://demo.flashphoner.com:9091/rest-api/mixer/startup
{
  "uri": "mixer://stream3",
  "localStreamName": "stream3"
}
```

6. Отправка запроса /mixer/add на добавление к микшеру mixer://stream3 потока stream1

```
http://demo.flashphoner.com:9091/rest-api/mixer/add
{
  "uri": "mixer://stream3",
  "localStreamName": "stream3"
  "remoteStreamName": "stream1"
}
```

7. Отправка запроса /mixer/add на добавление к микшеру mixer://stream3 потока stream2

```
http://demo.flashphoner.com:9091/rest-api/mixer/add
{
  "uri": "mixer://stream3",
  "localStreamName": "stream3"
  "remoteStreamName": "stream2"
}
```

8. Воспроизведение WebRTC-потока stream3

9. Отправка WebRTC-аудиопотока клиенту

## Известные проблемы

1. Микшер не создается, если имя микшера или имя выходного потока содержит символы, недопустимые для указания в URI

Симптомы: не создается микшер с именем вида test\_mixer.

Решение: не использовать в имени микшера или имени выходного потока недопустимые символы, в особенности, если включена возможность автоматического создания микшера. Например, имя

```
user_1#my_room
```

использовать нельзя.

Если микшируются потоки чат-комнат, в именах комнат также нельзя использовать недопустимые символы.

2. Выходной поток микшера будет пустым, если на сервере включен транскодинг только по требованию.

Симптомы: микшер видеопотоков успешно создается, но в выходном потоке черный экран.

Решение: для работы микшера на сервере должен быть включен транскодинг при помощи следующего параметра в файле [flashphoner.properties](#)

```
streaming_video_decoder_fast_start=true
```