Микширование потоков

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.

> **Warning**
>
> При микшировании потоков на сервере включается транскодинг. Рекомендуемая конфигурация сервера: 2 ядра CPU на 1 микшер.

Поддерживаемые протоколы входных потоков

- WebRTC
- RTMP
- RTSP
- MPEG-TS

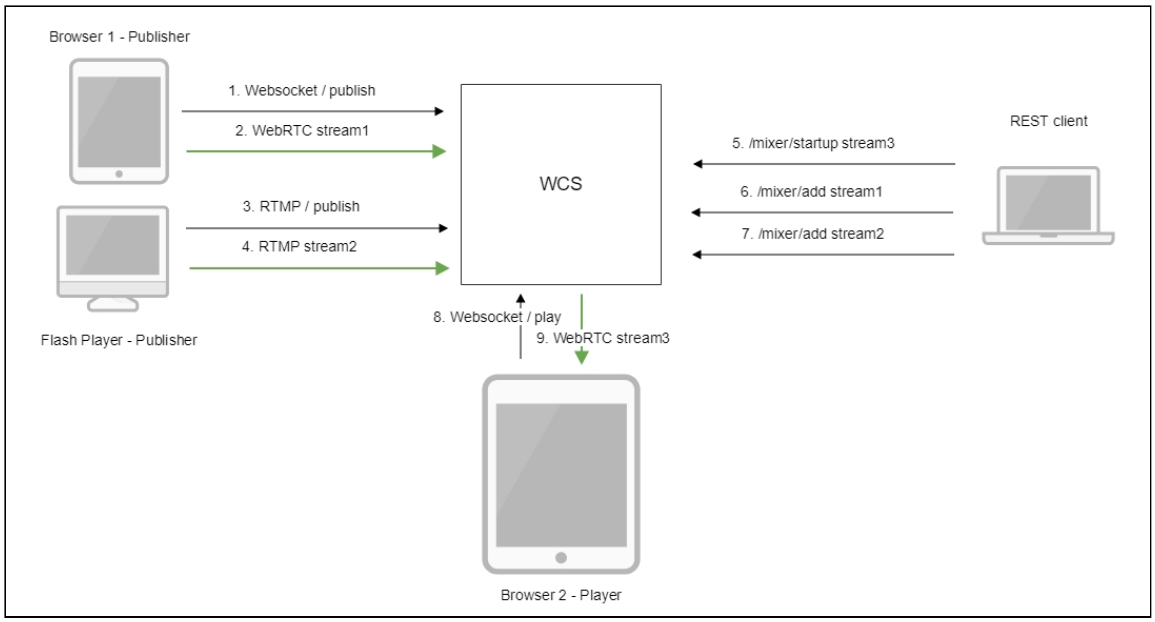
Возможности управления выходным потоком

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 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 добавлен в этот микшер. В имени потока может быть указано и ключевое слово демонстрации экрана, например `user1_desktop#room1`

Схема работы



1. Браузер соединяется с сервером по протоколу Websocket и отправляет команду `publishStream`.
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 `playStream`.
9. Второй браузер получает WebRTC `stream3` и воспроизводит этот поток на странице.

REST API

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

/mixer/startup

Создать микшер

REQUEST EXAMPLE

```
POST /rest-api/mixer/startup HTTP/1.1
Host: localhost:8081
Content-Type: application/json

{
    "uri": "mixer://mixer1",
    "localStreamName": "stream3",
    "hasVideo": true,
    "hasAudio": true,
    "watermark": "watermark.png",
    "background": "background.png",
    "mixerLayoutClass": "com.flashphoner.mixerlayout.TestLayout"
}
```

RESPONSE EXAMPLE

```
HTTP/1.1 200 OK
Access-Control-Allow-Origin: *
Content-Type: application/json
```

RETURN CODES

Code	Reason
200	OK
400	Bad request
409	Conflict
500	Internal error

/mixer/add

Добавить поток в микшер

REQUEST EXAMPLE

```
POST /rest-api/mixer/add HTTP/1.1
Host: localhost:8081
Content-Type: application/json

{
    "uri": "mixer://mixer1",
    "remoteStreamName": "stream1",
    "hasVideo": "true",
    "hasAudio": "true",
    "streamLabel": "John Doe",
    "videoPositionId": "speaker"
}
```

RESPONSE EXAMPLE

```
HTTP/1.1 200 OK
Access-Control-Allow-Origin: *
Content-Type: application/json
```

RETURN CODES

Code	Reason
200	OK
404	Not found
409	Conflict
500	Internal error

/mixer/remove

Удалить поток из микшера

REQUEST EXAMPLE

```
POST /rest-api/mixer/remove HTTP/1.1
Host: localhost:8081
Content-Type: application/json

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

RESPONSE EXAMPLE

```
HTTP/1.1 200 OK
Access-Control-Allow-Origin: *
Content-Type: application/json
```

RETURN CODES

Code	Reason
200	OK
404	Not found
500	Internal error

/mixer/find_all

Найти все микшеры

REQUEST EXAMPLE

```
POST /rest-api/mixer/find_all HTTP/1.1
Host: localhost:8081
Content-Type: application/json
```

RESPONSE EXAMPLE

```
HTTP/1.1 200 OK
Access-Control-Allow-Origin: *
Content-Type: application/json

[
    {
        "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"
        ],
        ...
    }
]
```

RETURN CODES

Code	Reason
200	OK
404	Not found
500	Internal error

/mixer/terminate

Завершить работу микшера

REQUEST EXAMPLE

```
POST /rest-api/mixer/terminate HTTP/1.1
Host: localhost:8081
Content-Type: application/json

{
    "uri": "mixer://mixer1"
}
```

RESPONSE EXAMPLE

```
HTTP/1.1 200 OK
Access-Control-Allow-Origin: *
Content-Type: application/json
```

RETURN CODES

Code	Reason
200	OK
404	Not found
500	Internal error

/mixer/setAudioVideo

Заглушить/возобновить видео или изменить уровень громкости аудио входного потока микшера

REQUEST EXAMPLE

```
POST /rest-api/mixer/setAudioVideo HTTP/1.1
Host: localhost:8081
Content-Type: application/json

{
    "uri": "mixer://m1",
    "streams": "^stream.*",
    "audioLevel": 0,
    "videoMuted": true
}
```

RESPONSE EXAMPLE

```
HTTP/1.1 200 OK
Access-Control-Allow-Origin: *
```

```
Content-Type: application/json
```

RETURN CODES

Code	Reason
200	OK
400	Bad request
404	Not found
500	Internal error

/mixer/set_body_watermark

Добавить водяной знак к картинке выходного потока микшера

REQUEST EXAMPLE

```
POST /rest-api/mixer/set_body_watermark HTTP/1.1
Host: localhost:8081
Content-Type: application/json

{
    "uri":"mixer://m1",
    "watermark":"/opt/media/logo.png",
    "x":10,
    "y":10,
    "marginTop":5,
    "marginLeft":5,
    "marginBottom":5,
    "marginRight":5
}
```

RESPONSE EXAMPLE

```
HTTP/1.1 200 OK
Access-Control-Allow-Origin: *
Content-Type: application/json
```

RETURN CODES

Code	Reason
200	OK
400	Bad request
404	Not found

/mixer/set_stream_watermark

Добавить водяной знак к картинке одного из входящих потоков микшера

REQUEST EXAMPLE

```
POST /rest-api/mixer/set_stream_watermark HTTP/1.1
Host: localhost:8081
Content-Type: application/json

{
    "uri":"mixer://m1",
    "watermark":"/opt/media/logo.png",
    "mediaSessionId": "030bb470-185c-11ed-9fad-918e05233ae9",
    "x":10,
    "y":10,
    "marginTop":5,
    "marginLeft":5,
    "marginBottom":5,
    "marginRight":5
}
```

RESPONSE EXAMPLE

```
HTTP/1.1 200 OK
Access-Control-Allow-Origin: *
Content-Type: application/json
```

RETURN CODES

Code	Reason
200	OK
400	Bad request
404	Not found

/mixer/set_stream_label

Установить или изменить отображаемое имя участника в микшере

REQUEST EXAMPLE

```
POST /rest-api/mixer/set_stream_label HTTP/1.1
Host: localhost:8081
Content-Type: application/json

{
    "uri":"mixer://m1",
    "remoteStreamName":"stream1",
    "streamLabel": "Mr. John Doe"
}
```

RESPONSE EXAMPLE

```
HTTP/1.1 200 OK
Access-Control-Allow-Origin: *
Content-Type: application/json
```

RETURN CODES

Code	Reason
200	OK
404	Not found

/mixer/set_parameter

Изменить параметр микшера

REQUEST EXAMPLE

```
POST /rest-api/mixer/set_parameter HTTP/1.1
Host: localhost:8081
Content-Type: application/json

{
    "uri":"mixer://m1",
    "mixerLayoutDir":"/opt/GridLayout"
}
```

RESPONSE EXAMPLE

```
HTTP/1.1 200 OK
Access-Control-Allow-Origin: *
Content-Type: application/json
```

RETURN CODES

Code	Reason
200	OK
400	Bad request
404	Not found

/mixer/set_stream_avatar

Установить картинку аватара на аудио поток в микшере

> **Warning**
>
> Данная возможность поддерживается только на системах от Ubuntu 20.04 и других с glibc 2.31 и новее!

REQUEST EXAMPLE

```
POST /rest-api/mixer/set_stream_avatar HTTP/1.1
Host: localhost:8081
Content-Type: application/json

{
    "uri":"mixer://m1",
    "remoteStreamName":"stream1",
    "avatar": "https://mystorage/storage/avatar.png"
}
```

RESPONSE EXAMPLE

```
HTTP/1.1 200 OK
Access-Control-Allow-Origin: *
Content-Type: application/json
```

RETURN CODES

Code	Reason
200	OK
400	Bad request
404	Not found
500	Internal server error

/mixer/remove_stream_avatar

Убрать картинку аватара с аудио потока в микшере

> **Warning**
>
> Данная возможность поддерживается только на системах от Ubuntu 20.04 и других с glibc 2.31 и новее!

REQUEST EXAMPLE

```
POST /rest-api/mixer/remove_stream_avatar HTTP/1.1
Host: localhost:8081
```

```
Content-Type: application/json

{
    "uri":"mixer://m1",
    "remoteStreamName":"stream1"
}
```

RESPONSE EXAMPLE

```
HTTP/1.1 200 OK
Access-Control-Allow-Origin: *
Content-Type: application/json
```

RETURN CODES

Code	Reason
200	OK
400	Bad request
404	Not found
500	Internal server error

/mixer/set_position

Переместить картинку потока в указанную позицию (только для собственных вариантов размещения картинок)

REQUEST EXAMPLE

```
POST /rest-api/mixer/set_position HTTP/1.1
Host: localhost:8081
Content-Type: application/json

{
    "uri":"mixer://m1",
    "remoteStreamName":"stream1",
    "videoPositionId":"speaker"
}
```

RESPONSE EXAMPLE

```
HTTP/1.1 200 OK
Access-Control-Allow-Origin: *
Content-Type: application/json
```

RETURN CODES

Code	Reason

Code	Reason
200	OK
400	Bad request
404	Not found
500	Internal server error

Параметры

Имя параметра	Описание	Пример
uri	Уникальный идентификатор микшера	`mixer://mixer1`
localStreamName	Имя выходного потока микшера	`stream3`
hasVideo	Микшировать видео	`true`
hasAudio	Микшировать аудио	`true`
remoteStreamName	Имя потока, добавляемого в микшер	`stream1 rtmp://rtmp.flashphoner.com:1935/live/rtmp_stream1`
mediaSessionId	Идентификатор медиасессии на сервере	`ce92b134-2468-4460-8d06-1ea3c5aabace`
status	Статус потока	`PROCESSED_LOCAL`
background	Фоновое изображение	`background.png`
watermark	Водяной знак	`watermark.png`
mixerLayoutClass	Класс размещения картинок в потоке	`com.flashphoner.mixerlayout.TestLayout`
streams	Список потоков или регулярное выражение для поиска	`^stream.* ["stream1", "stream2"]`
audioLevel	Уровень громкости аудио для входного потока	`0`
videoMuted	Заглушить видео	`true`
streamLabel	Отображаемое имя участника в микшере	`John Doe`
avatar	URI картинки в форматах PNG, JPG, BMP	`https://mystorage.com/storage/avatar.jpg`

Имя параметра	Описание	Пример
videoPositionId	Идентификатор позиции в описании варианта размещения картинок	`speaker`

Настройка микшера при создании по REST API

В сборке 5.2.872 добавлена возможность передать параметры, аналогичные настройкам микшера в файле flashphoner.properties, при создании микшера по запросу `/mixer/startup`. В этом случае параметры будут применены только к этому экземпляру микшера. Например, запрос

```
{
 "uri": "mixer://mixer1",
 "localStreamName": "stream3",
 "mixerVideoWidth": 640,
 "mixerVideoHeight": 360,
 "mixerVideoFps": 24,
 "mixerVideoBitrateKbps": 500
}
```

создаст микшер с выходным потоком 640x360, fps 24 и битрейтом 500 кбит/с, независимо от того, какие настройки указаны в файле flashphoner.properties.

Полный список параметров микшера можно получить в ответе на запрос `/mixer/find_all`, например

Mixer parameters full list

Изменение параметров микшера в процессе его работы по REST API

Начиная со сборки 5.2.1480, следующие параметры можно менять для активного микшера при помощи запроса `/mixer/set_parameter`:

```
[
  {
    ...
    "mixerLayoutClass":
"com.flashphoner.media.mixer.video.presentation.GridLayout",
    "mixerAutoScaleDesktop": true,
    "mixerDesktopAlign": "TOP",
    "mixerDisplayStreamName": true,
    "mixerFontSize": 20,
    "mixerFontSizeAudioOnly": 40,
    "mixerMinimalFontSize": 1,
    "mixerShowSeparateAudioFrame": true,
    "mixerTextAutoscale": true,
    "mixerTextColour": "0xFFFFFF",
    "mixerTextBulkWriteWithBuffer": true,
```

```
    "mixerTextBulkWrite": true,
    "mixerTextBackgroundOpacity": 100,
    "mixerTextBackgroundColour": "0x2B2A2B",
    "mixerFrameBackgroundColour": "0x2B2A2B",
    "mixerTextPaddingLeft": 5,
    "mixerVoiceActivitySwitchDelay": 0,
    "mixerVoiceActivityFrameThickness": 6,
    "mixerVoiceActivityFramePositionInner": false,
    "mixerVoiceActivityColour": "0x00CC66",
    "mixerVoiceActivity": true,
    "mixerVideoLayoutDesktopKeyWord": "desktop",
    "mixerVideoGridLayoutPadding": 30,
    "mixerVideoGridLayoutMiddlePadding": 10,
    "mixerVideoDesktopLayoutPadding": 30,
    "mixerVideoDesktopLayoutInlinePadding": 10,
    "mixerTextPaddingTop": 5,
    "mixerTextPaddingRight": 4,
    "mixerTextFont": "Serif",
    "mixerTextPaddingBottom": 5,
    "mixerTextDisplayRoom": true,
    "mixerTextCutTop": 3,
    "mixerTextAlign": "BOTTOM_LEFT",
    "mixerVideoDesktopFullscreen": false,
    "mixerLayoutDir": ""
  }
]
```


Если запрос содержит параметры, которые не могут быть изменены для активного микшера, такой запрос вернет `400 Bad request` с указанием неподдерживаемого параметра.

Отправка REST-запроса к WCS-серверу

Для отправки REST-запроса к WCS-серверу необходимо использовать REST-клиент, `curl` для отправки запроса из скрипта, либо подходящую функцию языка разработки приложения.

Ограничения

При создании микшера при помощи REST API необходимо указать имя его выходного потока (параметр `localStreamName`). Если этого не сделать, сервер вернет `400 Bad request` с сообщением `No localStreamName given`.

Настройка

Настройка автоматического создания микшера при публикации потока

Для того, чтобы включить возможность автоматического создания микшера для потоков, содержащих в имени символ `#`, необходимо, чтобы REST hook 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 flash
```

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.

В сборке 5.2.689 добавлена возможность включить или отключить микширование аудио при создании микшера

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

или в настройках сервера для всех потоков

```
mixer_audio_enabled=false
```

Публикация в аудио-видео микшер потока без одной из составляющих

Если в аудио-видео микшер публикуется поток без видео или без аудио дорожки, необходимо отключать контроль активности аудио и видео

```
rtp_activity_audio=false
rtp_activity_video=false
```

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 настройки 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 кодеком видео выходного потока микшера

В сборке 5.2.1075 добавлена возможность указать кодек видео выходного потока микшера. По умолчанию, используется кодек H264

```
video_mixer_output_codec=h264
```

Кодек VP8 можно указать при помощи настройки

```
video_mixer_output_codec=vp8
```

Отметим, что в этом случае ключевые фреймы в потоке идут реже по сравнению с H264, поскольку размер ключевого фрейма существенно больше. Это может привести к тому, что первый кадр видео при проигрывании видео в браузере может отобразиться с задержкой относительно потока H264 с такими же параметрами.

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.

> **Warning**
>
> 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

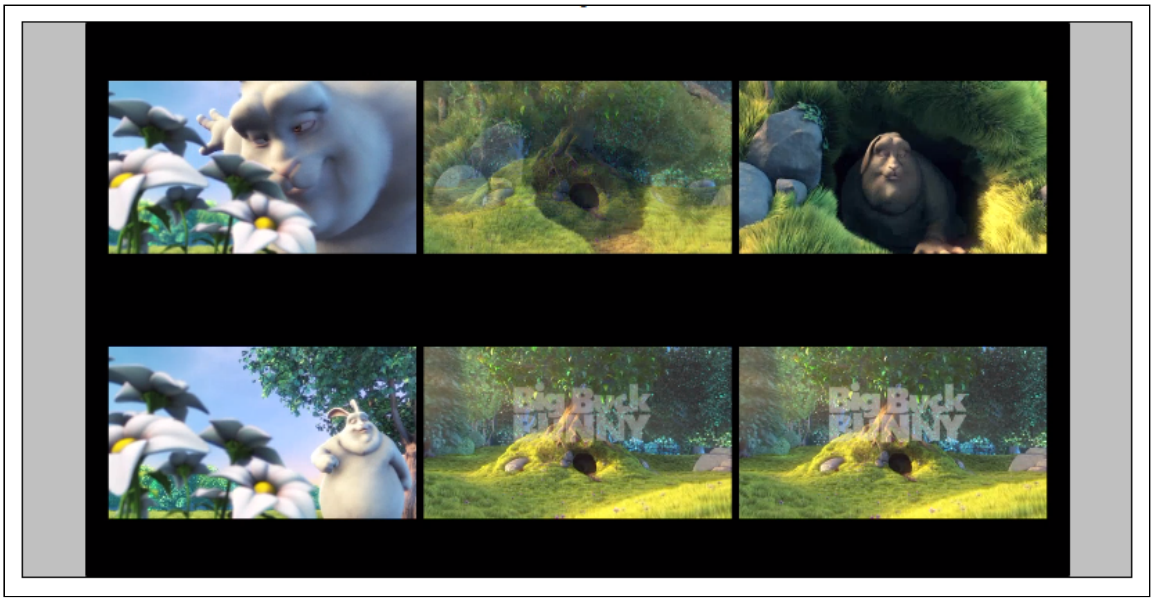
```
mixer_idle_timeout=60000
```

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:

Картинки сеткой



Этот вариант включается настройкой в файле flashphoner.properties

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

Картинки сеткой с минимальным расстоянием между ними

Этот вариант включается настройкой

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

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

Картинки сеткой с минимальным расстоянием между ними и обрезкой картинок вокруг центра

В сборке 5.2.842 добавлена возможность обрезки картинок с тем, чтобы в микшере был виден только центр картинки. Это может быть полезно при видеоконференции, когда лицо участника находится в центре картинки.

Этот вариант включается настройкой

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

Демонстрация экрана (screen sharing)

Этот вариант включается, если на вход микшеру подается поток с именем, содержащим ключевое слово, указанное в настройке

```
mixer_video_layout_desktop_key_word=desktop
```

По умолчанию, для потока демонстрации экрана имя должно включать слово `desktop` , например `user1_desktop`

В сборке 5.2.710 добавлена возможность настройки размещения картинки экрана (screen sharing):

```
mixer_desktop_align=TOP
```

По умолчанию, картинка экрана размещается над остальными картинками

Поддерживаются следующие варианты размещения:

Вариант	Описание
`TOP`	Экран вверху
`LEFT`	Экран слева
`RIGHT`	Экран справа
`BOTTOM`	Экран внизу
`CENTER`	Экран в центре (картинки потоков в округ)

Например, при настройке

```
mixer_desktop_align=RIGHT
```

экран будет помещен справа от остальных картинок

Если опубликовано несколько потоков с экрана, главным потоком будет выбран тот, который опубликован первым. Если затем этот поток будет завершен, его место займет поток, следующий по алфавиту.

Картинка в картинке

Этот вариант размещения добавлен в сборке 5.2.852. В этом случае поток с ключевым слово в имени, указанным в настройке

```
mixer_video_layout_desktop_key_word=desktop
```

например, `user1_desktop`, будет фоном для картинок остальных потоков. Такое размещение картинок включается при помощи настройки

```
mixer_video_desktop_fullscreen=true
```

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

Для того, чтобы собственный класс поддерживал размещение текста над или под картинкой потока, начиная со сборки 5.2.878 необходимо использовать класс `Box` для расчета расположения картинок, например

TestLayoutWithBox.java

Основные методы класса `Box`:

Box methods available

Возможные позиции для размещения объекта класса `Box`:

BoxPosition

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/wcs-core.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 настройке в файле 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:

ОБРЕЗКА КАРТИНОК ВОКРУГ ЦЕНТРА ПРИ СОБСТВЕННОМ ВАРИАНТЕ РАЗМЕЩЕНИЯ КАРТИНОК

Для обрезки картинок с выделением центральной части, аналогично `CropNoPaddingGridLayout`, при создании собственного варианта размещения картинок, необходимо вместо метода `Box.fillParent()` использовать

`Box.fillParentNoScale()`. Например, вариант размещения картинок по имени с обрезкой вокруг центра:

SideBySideLayout.java

ОТДЕЛЬНЫЙ КАТАЛОГ ДЛЯ СОБСТВЕННЫХ JAVA БИБЛИОТЕК

Начиная со сборки 5.2.1512, Java библиотеки (jar файлы) должны помещаться в каталог `/usr/local/FlashphonerWebCallServer/lib/custom`

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

Этот каталог сохраняется при дальнейших обновлениях сервера к более новым сборкам. Таким образом, нет необходимости снова копировать jar файлы после установки обновления.

Управление размещением картинок при создании микшера

В сборке 5.2.693 появилась возможность указать размещение картинок при создании микшера по REST API, например

```
{
 "uri": "mixer://mixer1",
 "localStreamName": "mixer1",
 "mixerLayoutClass": "com.flashphoner.mixerlayout.TestLayout"
}
```

Таким образом, размещением картинок можно управлять для каждого микшера отдельно

Управление профилем кодирования выходного потока микшера

Некоторые браузеры не поддерживают воспроизведение H264 потока, закодированного с определенным профилем. В связи с этим, в сборке 5.2.414 добавлена настройка для указания профиля кодирования выходного потока микшера

```
mixer_video_profile_level=42c02a
```

По умолчанию, задан профиль constrainted baseline level 4.2.

Управление фоном микшера и добавление водяного знака при создании микшера

В сборке 5.2.693 появилась возможность указать фон микшера и добавить водяной знак при создании микшера по REST API, например

```
{
 "uri": "mixer://mixer1",
 "localStreamName": "mixer1",
 "watermark": "watermark.png",
 "background": "background.png"
}
```

По умолчанию, файлы должны располагаться в каталоге `/usr/local/FlashphonerWebCallServer/conf`. Можно указать и полный путь к файлам, например

```
{
 "uri": "mixer://mixer1",
 "localStreamName": "mixer1",
 "watermark": "/opt/media/watermark.png",
 "background": "/opt/media/background.png"
}
```

Динамическое добавление и изменение водяного знака

В сборке 5.2.1349 добавлена возможность динамически добавлять и изменять водяной знак, не останавливая микшер и не публикуя поток заново. Водяной знак может быть добавлен, изменен или перемещен в соответствии с указанными координатами при помощи REST API запросов:

- `/mixer/set_body_watermark` - к картинке всего выходного потока микшера

  ```
  {
   "uri":"mixer://m1",
   "watermark":"/opt/media/logo.png",
   "x":0,
   "y":0,
   "marginTop":0,
   "marginLeft":0,
   "marginBottom":0,
   "marginRight":0
  }
  ```

- `/mixer/set_stream_watermark` - к картинке одного из входных потоков микшера

```
{
 "uri":"mixer://m1",
 "watermark":"/opt/media/logo.png",
 "mediaSessionId": "030bb470-185c-11ed-9fad-918e05233ae9",
 "x":0,
 "y":0,
 "marginTop":0,
 "marginLeft":0,
 "marginBottom":0,
 "marginRight":0
}
```

Здесь

- `watermark` - имя файла водяного знака
- `x`, `y` - координаты верхнего левого угла водяного знака на картинке потока
- `marginTop`, `marginLeft`, `marginBottom`, `marginRignt` - отступы от границ картинки потока

Если координаты выходят за границы картинки потока, водяной знак будет вписан в эти границы с учетом отступов.

Для того, чтобы переместить водяной знак в другое место на картинке, необходимо отправить запрос с тем же файлом, но новыми координатами. Чтобы убрать водяной знак с картинки, необходимо отправить запрос с пустым полем `watermark`

```
{
 "uri":"mixer://m1",
 "watermark":""
}
```

Стерео звук в выходном потоке микшера

По умолчанию, микшер преобразует стерео аудио из входящих потоков в моно, чтобы снизить количество обрабатываемых данных. Это позволяет уменьшить возможные задержки при использовании микшеров реального времени для видеоконференций.

При необходимости, микшер может быть переключен в режим обработки стерео звука, например, в случае онлайн-радиостанции для трансляции музыки. Для этого в сборке 5.2.922 добавлена настройка, позволяющая задать количество аудио каналов микшера

```
audio_mixer_output_channels=2
```

Декодирование символов в имени входящего потока

В сборке 5.2.1802 добавлена возможность декодирования символов в имени потока, закодированных на стороне клиента при помощи encodeURIComponent(). Эта возможность включается настройкой

```
mixer_decode_stream_name=true
```

или добавлением параметра при создании микшера запросом `/mixer/startup`

```
{
  "uri": "mixer://mixer1",
  "localStreamName": "mixer1",
  ...,
  "mixerDecodeStreamName": true
}
```

В этом случае в изображении будут отображаться декодированные символы, если такие символы есть в используемом шрифте, или близкие к ним по начертанию.

Поддержка MCU

Поддержка MCU в микшере включается для аудио настройкой

```
mixer_mcu_audio=true
```

В этом случае для трех входящих потоков `stream1`, `stream2`, `stream3` на выходе микшера `mixer1` будут следующие потоки:

Output stream name	stream1		stream2		stream3	
	audio	video	audio	video	audio	video
mixer1	✅	✅	✅	✅	✅	✅
mixer1-stream1	❌	❌	✅	❌	✅	❌
mixer1-stream2	✅	❌	❌	❌	✅	❌
mixer1-stream3	✅	❌	✅	❌	❌	❌

Таким образом, каждый из дополнительных потоков содержит аудио всех потоков в микшере, кроме одного. Это позволяет, например, устранить эхо для участников конференции.

Аналогичная возможность для видео включается настройкой

```
mixer_mcu_video=true
```

В этом случае для трех входящих потоков `stream1`, `stream2`, `stream3` на выходе микшера `mixer1` будут следующие потоки:

Output stream name	stream1		stream2		stream3	
	audio	video	audio	video	audio	video
mixer1	✅	✅	✅	✅	✅	✅
mixer1-stream1	❌	❌	✅	✅	✅	✅
mixer1-stream2	✅	✅	❌	❌	✅	✅
mixer1-stream3	✅	✅	✅	✅	❌	❌

Таким образом, каждый из дополнительных потоков содержит аудио и видео всех потоков в микшере, кроме одного. Это позволяет организовать полноценную чат-комнату на базе микшера.

При этом, если включена запись микшера при помощи настройки

```
record_mixer_streams=true
```

будет записываться только основной выходной поток микшера (`mixer1` в вышеприведенном примере).

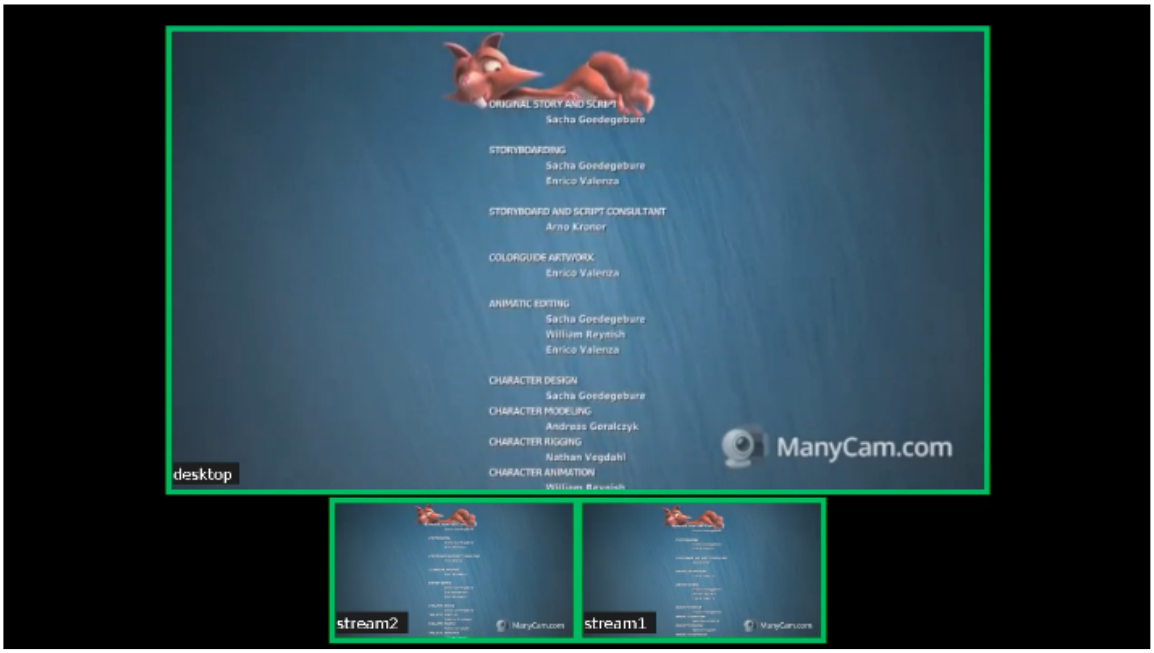
Управление аудио и видео входного потока

В сборке 5.2.835 добавлена возможность управлять уровнем аудио и заглушать видео для входных потоков в микшере. Оригинальный поток при этом не изменяется. Видео может быть заглушено (черный экран) и затем восстановлено. Для аудио может быть указан уровень громкости, либо звук может быть заглушен снижением уровня до 0.

Входные потоки управляются при помощи REST API запроса `/mixer/setAudioVideo`.

Например, создаем микшер и добавляем туда три потока: два потока`участников и один поток ведущего

```
curl -H "Content-Type: application/json" -X POST http://localhost:8081/rest-
api/mixer/startup -d '{"uri": "mixer://m1", "localStreamName":"m1"}'
curl -H "Content-Type: application/json" -X POST http://localhost:8081/rest-
api/mixer/add -d '{"uri": "mixer://m1", "remoteStreamName": "stream1"}'
curl -H "Content-Type: application/json" -X POST http://localhost:8081/rest-
api/mixer/add -d '{"uri": "mixer://m1", "remoteStreamName": "stream2"}'
curl -H "Content-Type: application/json" -X POST http://localhost:8081/rest-
api/mixer/add -d '{"uri": "mixer://m1", "remoteStreamName": "desktop"}'
```



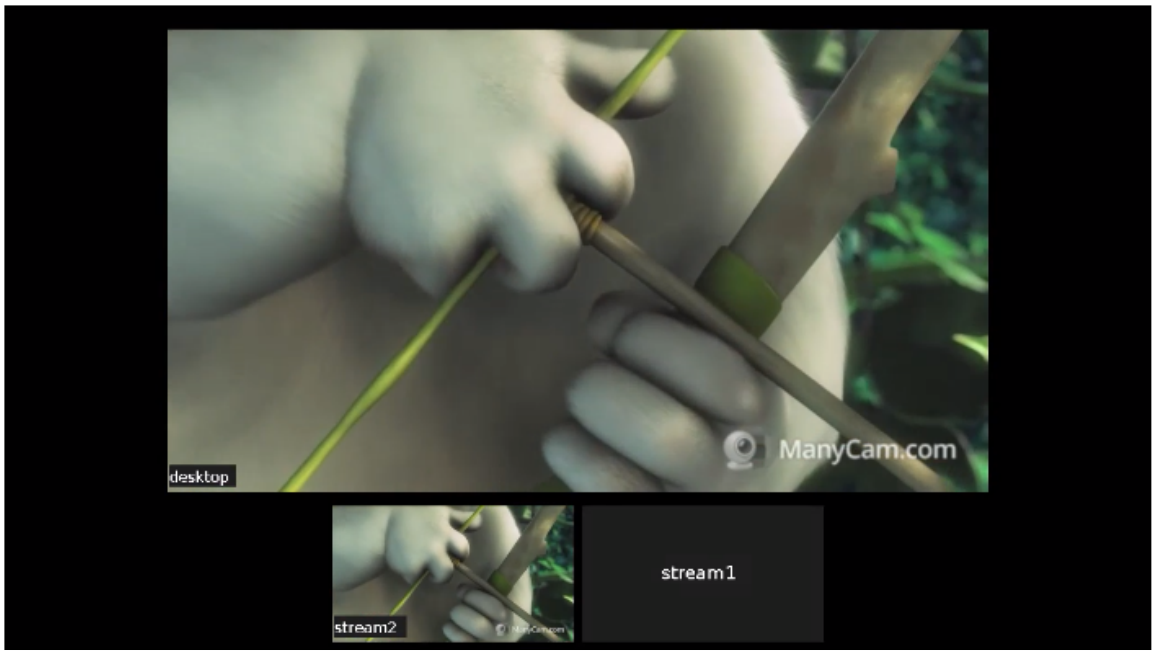
Уберем звук у всех участников, кроме ведущего

```
POST /rest-api/mixer/setAudioVideo HTTP/1.1
User-Agent: curl/7.29.0
Host: localhost:8081
Content-Type: application/json
Content-Length: 62

{
 "uri": "mixer://m1",
 "streams": "^stream.*",
 "audioLevel": 0
}
```

Параметр `streams` может быть задан либо как регулярное выражение для поиска потоков по имени, либо как список потоков. Синтаксис регулярных выражении соответствует поддерживаемому в языке Java, примеры приведены здесь.

Уберем видео у потока `stream1`

```
POST /rest-api/mixer/setAudioVideo HTTP/1.1
User-Agent: curl/7.29.0
Host: localhost:8081
Content-Type: application/json
Content-Length: 65

{
 "uri": "mixer://m1",
 "streams": ["stream1"],
 "videoMuted": true
}
```



Проверим состояние потоков запросом `/mixer/find_all`

Request

```
POST /rest-api/mixer/find_all HTTP/1.1
User-Agent: curl/7.29.0
Host: localhost:8081
Content-Type: application/json
```

Response

```
HTTP/1.1 200 OK
Access-Control-Allow-Origin: *
Access-Control-Allow-Credentials: true
Content-Type: application/json
Content-Length: 574

[
  {
    "localMediaSessionId": "e2fa5c8b-16f3-4917-9d5f-557dde75db07",
    "localStreamName": "m1",
    "uri": "mixer://m1",
    "status": "PROCESSED_LOCAL",
    "hasAudio": true,
    "hasVideo": true,
    "record": false,
    "mediaSessions": [
      {
        "localMediaSessionId": "3dd763b0-2ae7-11eb-aa72-37b2cbcbf6b9",
        "audioLevel": 0,
        "videoMuted": true,
        "localStreamName": "stream1"
      },
      {
        "localMediaSessionId": "8af64760-2ae7-11eb-b086-cdf035231b9d",
```

```
        "audioLevel": 100,
        "videoMuted": false,
        "localStreamName": "desktop"
      },
      {
        "localMediaSessionId": "7cc4b410-2ae7-11eb-b34c-a5240fe9f151",
        "audioLevel": 0,
        "videoMuted": false,
        "localStreamName": "stream2"
      }
    ]
  }
]
```

Управление аудио и видео входного потока при добавлении его в микшер

В сборке 5.2.982 добавлена возможность заглушить или изменить уровень аудио и заглушить видео при добавлении потока в микшер. Для этого в запросе `/mixer/add` необходимо указать дополнительные параметры аналогично запросу `/mixer/setAudioVideo`

```
POST /rest-api/mixer/add HTTP/1.1
User-Agent: curl/7.29.0
Host: localhost:8081
Content-Type: application/json
Content-Length: 85

{
 "uri": "mixer://m1",
 "remoteStreamName": "stream1",
 "audioLevel": 0,
 "videoMuted": false
}
```

Отметим, что, если поток в микшер был добавлен с параметром `hasVideo: false`

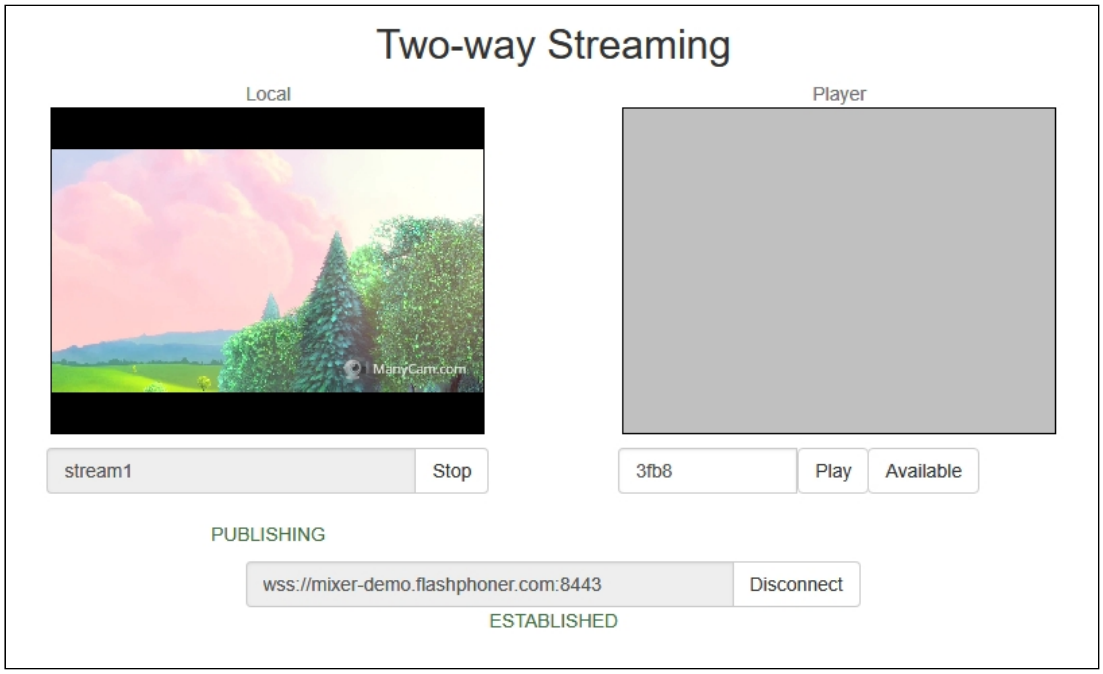
```
{
 "uri": "mixer://m1",
 "remoteStreamName": "stream1",
 "hasVideo": false,
 "hasAudio": true
}
```

то параметр `videoMuted` будет игнорироваться, такой поток в микшере останется только с аудио.

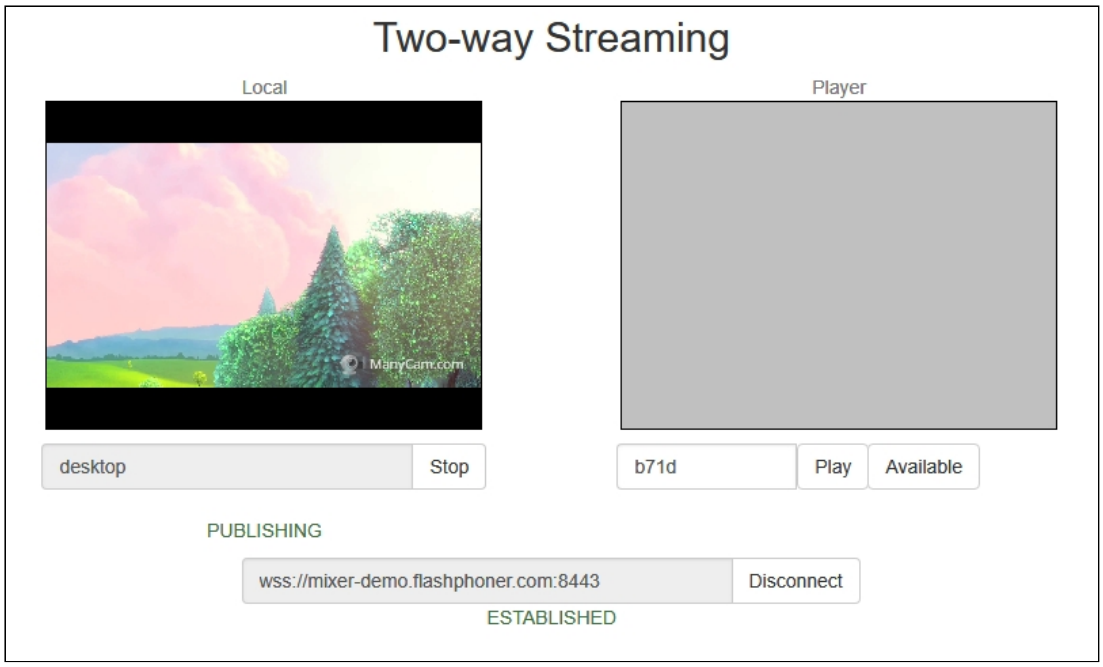
Краткое руководство по тестированию

1. Для теста используем:

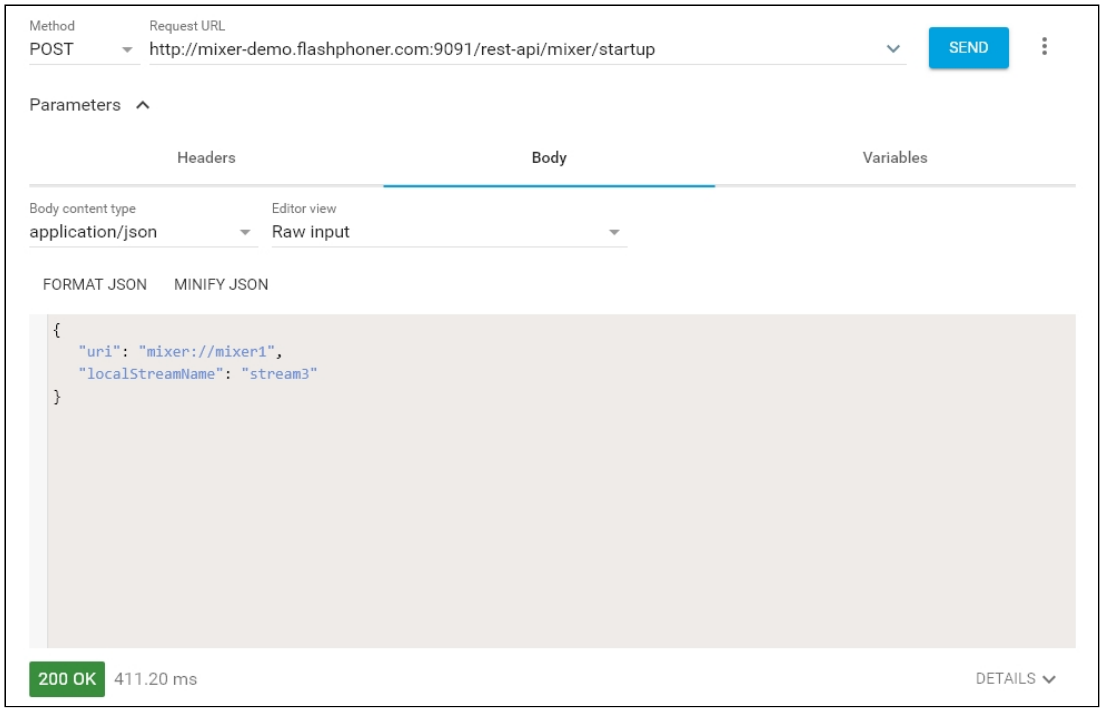
2. демо-сервер demo.flashphoner.com;
3. браузер Chrome и REST-клиент для отправки запросов на сервер;
4. веб-приложение Two Way Streaming для публикации входных потоков микшера;
5. веб-приложение Player для воспроизведения выходного потока микшера.
6. Откройте страницу веб-приложения Two Way Streaming. Опубликуйте поток с именем `stream1`:



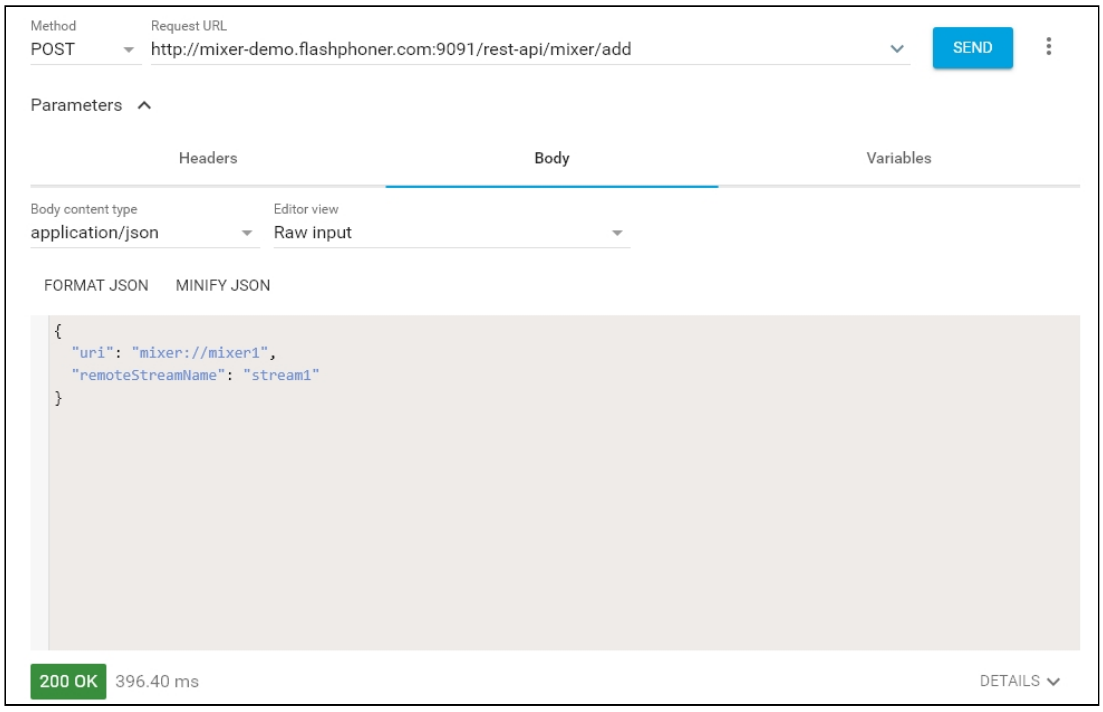
7. В другой вкладке откройте страницу веб-приложения Two Way Streaming. Опубликуйте поток с именем `desktop`:



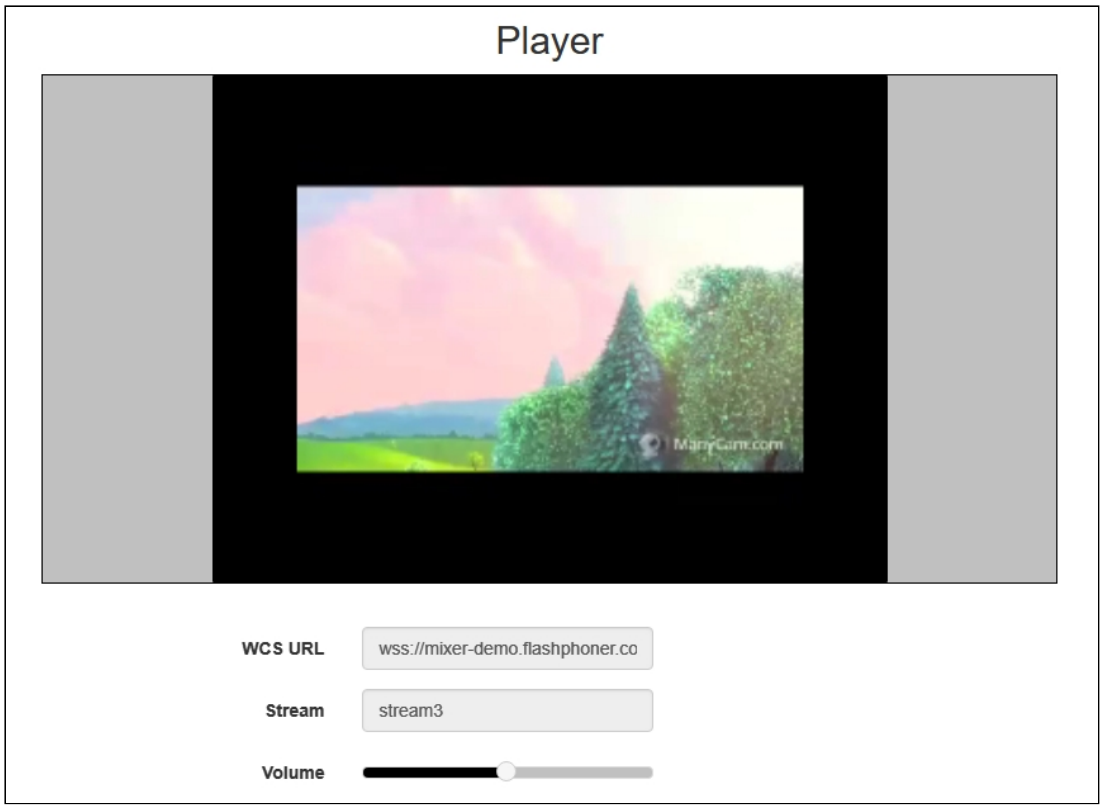
8. Откройте REST-клиент. Отправьте запрос `/mixer/startup`, указав в параметрах URI микшера `mixer://mixer1` и имя выходного потока `stream3`:



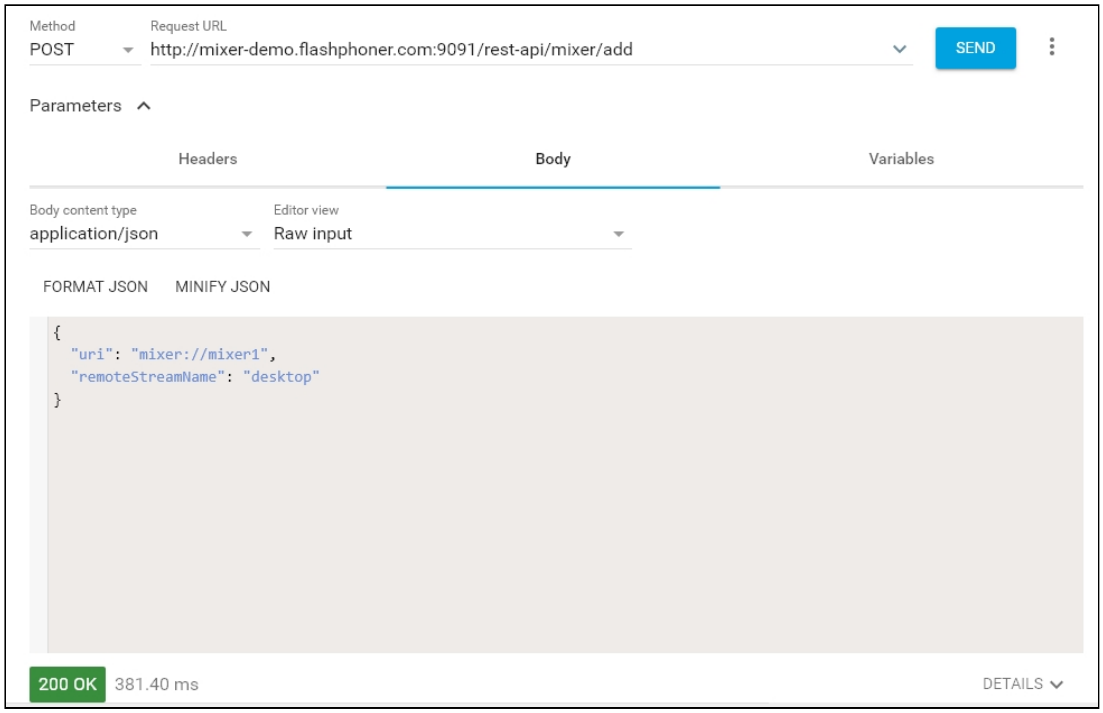
9. Отправьте запрос `/mixer/add`, указав в параметрах URI микшера `mixer://mixer1` и имя входного потока `stream1`:



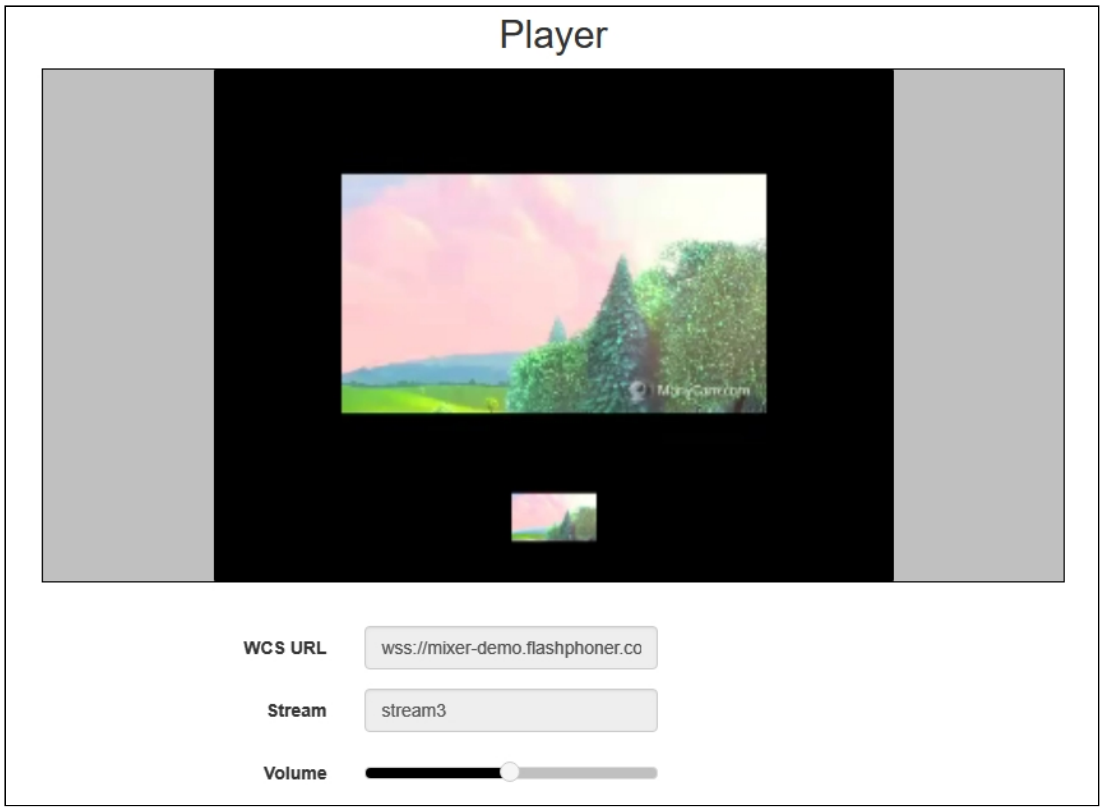
10. Откройте веб-приложение Player, укажите в поле `Stream` имя выходного потока микшера `stream3` и нажмите `Start`:



11. Отправьте запрос `/mixer/add`, указав в параметрах URI микшера `mixer://mixer1` и имя входного потока `desktop`:

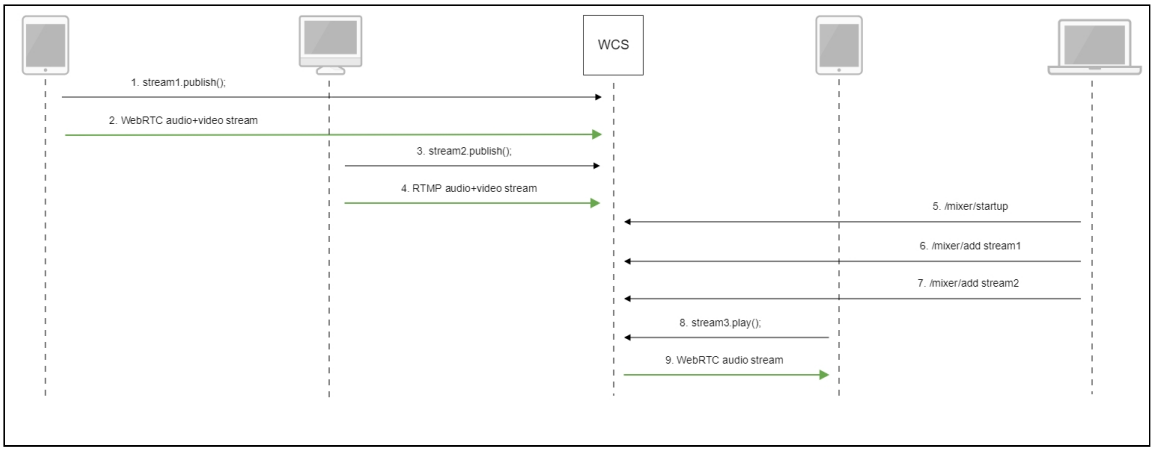


12. В выходном потоке микшера отобразится поток `desktop`, имитирующий демонстрацию экрана, и поток `stream1`:



Последовательность выполнения операций

Ниже описана последовательность вызовов при использовании микшера.



1. Публикация WebRTC-потока `stream1`
2. Отправка WebRTC 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-потока клиенту

REST hooks для микшера

По умолчанию, для выходного потока микшера могут быть обработаны все стандартные REST методы. При этом, REST метод `/connect` приходит на бэкенд сервер при создании микшера.

При необходимости, для микшера может быть создано отдельное REST-приложение. Чтобы направить вызовы от микшера к этому приложению, в сборке 5.2.634 добавлена настройка

```
mixer_app_name=defaultApp
```

По умолчанию, вызовы от микшера обрабатывает приложение `defaultApp`, как и для остальных потоков.

Известные проблемы

1. Имя микшера не может содержать символы, недопустимые в URI

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 автоматичекого создания микшера. Например, имя `user_1#my_room` использовать нельзя. Если микшируются потоки чат-комнат, в именах комнат также нельзя использовать недопустимые символы.

2. Один поток не может быть добавлен в два микшера, если не используется микшер реального времени

Один поток не может быть добавлен в два микшера одновременно, если не используется микшер реального времени

Симптомы

При добавлении потока во второй микшер выходной поток первого микшера останавливается

Решение

Не использовать один поток более чем в одном микшере при `mixer_realite=false`

3. Настройка качества кодирования не применяется при использовании OpenH264

Симптомы

Качество картинки не изменяется при различных значениях настройки `mixer_video_quality`, например

```
mixer_video_quality=5
```

не отличается от

```
mixer_video_quality=20
```

Решение

Не использовать кодирование на базе OpenH264, поскольку управление CRF в нем не поддерживается

```
encoder_priority=FF
```

4. Если файл водяного знака поврежден, отображается черный экран

При добавлении водяного знака или фона, если PNG файл поврежден, либо это не PNG файл, используется водяной знак по умолчанию (черная картинка)

Симптомы

a) При добавлении водяного знака в выходном потоке черный экран, в серверном логе сообщение

```
Wrong watermark file format. Should be PNG
```

b) при добавлении фона в выходном потоке фон остается черным, в серверном логе сообщение

```
Custom mixer background file wrong format. Should be PNG
```

Решение

Использовать только PNG файл с корректной структурой для добавления водяного знака или фона.