

# Вставка одного потока в другой

## Описание

В сборке [5.2.841](#) добавлена возможность вставки одного опубликованного на сервере потока в другой. Эту функцию можно использовать, например, для добавления рекламы в поток. При этом содержимое одного потока полностью заменяется другим, либо до окончания публикации второго потока, либо до прекращения вставки.

## Поддерживаемые кодеки

Видео:

- H264
- VP8

Аудио:

- Opus
- AAC
- G711

## Ограничения

1. Оба потока, к которым применяется вставка, должны быть закодированы одинаковыми аудио и видео кодеками.
2. Для аудио, должна быть одинаковая частота дискретизации и одинаковое количество каналов.
3. Вставка не применяется к потокам звонков. Для звонков используются собственные технологии вставки [аудио](#) и [видео](#).
4. В один поток может быть вставлен только один поток одновременно, но один и тот же поток может быть вставлен в несколько потоков.
5. Циклическая вставка не поддерживается. Невозможно вставить `stream1` в `stream2`, а затем `stream2` в `stream1` без остановки предыдущей вставки.

Реализация вставки в сборках до [5.2.1618](#)

## REST API

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

- HTTP: `http://test.flashphoner.com:8081/rest-api/stream/inject/startup`
- HTTPS: `https://test.flashphoner.com:8444/rest-api/stream/inject/startup`

Здесь:

- `test.flashphoner.com` - адрес WCS-сервера
- `8081` - стандартный REST / HTTP порт WCS-сервера
- `8444` - стандартный HTTPS порт
- `rest-api` - обязательная часть URL
- `/stream/inject/startup` - используемый REST-метод

## REST методы и ответы

### **`/stream/inject/startup`**

Вставить поток stream2 в stream1

#### REQUEST EXAMPLE

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

{
  "localStreamName": "stream1",
  "remoteStreamName": "stream2"
}
```

#### 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
409	Conflict

Code	Reason
500	Internal error

### **/stream/inject/find\_all**

Найти все вставки на сервере

#### REQUEST EXAMPLE

```
POST /rest-api/stream/inject/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

[
  {
    "localStreamName": "stream1",
    "remoteStreamName": "stream2"
  }
]
```

#### RETURN CODES

Code	Reason
200	OK
404	Not found

### **/stream/inject/terminate**

Остановить вставку в поток stream1

#### REQUEST EXAMPLE

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

{
  "localStreamName": "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 error

#### Parameters

Name	Description	Example
localStreamName	Имя потока, в который производится вставка	<code>stream1</code>
remoteStreamName	Имя потока, который будет вставлен	<code>stream2</code>

#### Вставка VOD потока из файла

В сборке [5.2.1535](#) добавлена возможность вставить VOD поток непосредственно из файла при отправке запроса `/stream/inject/startup`:

```
{
  "localStreamName": "host",
  "remoteStreamName": "vod-live://advertising.mp4"
}
```

При этом вставляемый файл начинает проигрываться без пауз, с первого ключевого кадра. Если этот же файл вставить в другой поток, в том потоке файл также начнет проигрываться с начала.

Эта возможность полезна, например, при вставке рекламных роликов в поток, который смотрят зрители.

#### Настройка

В сборке [5.2.1235](#) добавлена настройка, которая определяет, в течение какого времени в миллисекундах необходимо ждать ключевого кадра во вставляемом

потоке

```
inject_wait_keyframe_ms=1000
```

По умолчанию, интервал составляет 1000 миллисекунд. Если ключевой кадр во вставляемом потоке за это время не был получен, сервер начинает генерировать черный фон (по умолчанию), либо кадр с изображением из файла, заданного в настройке `custom_watermark_filename`. Это поведение можно отключить настройкой

```
inject_wait_keyframe_ms=-1
```

В этом случае будет продолжаться проигрывание потока, в который производится вставка, до момента получения ключевого кадра во вставляемом потоке.

## Реализация вставки в сборке 5.2.1618 и новее

### Настройка

В сборке 5.2.1618 добавлена новая реализация вставки одного потока в другой, позволяющая выбрать, какую именно составляющую заменить: аудио, видео или обе. Эта возможность включается настройкой

```
use_new_injector=true
```

### REST API

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

- HTTP: `http://test.flashphoner.com:8081/rest-api/stream/inject2/startup`
- HTTPS: `https://test.flashphoner.com:8444/rest-api/stream/inject2/startup`

Здесь:

- `test.flashphoner.com` - адрес WCS-сервера
- `8081` - стандартный REST / HTTP порт WCS-сервера
- `8444` - стандартный HTTPS порт
- `rest-api` - обязательная часть URL
- `/stream/inject2/startup` - используемый REST-метод

### REST методы и ответы

`/stream/inject2/startup`

Вставить поток stream2 в поток stream1

#### REQUEST EXAMPLE

```
POST /rest-api/stream/inject2/startup HTTP/1.1
Host: localhost:8081
Content-Type: application/json

{
  "localStreamName": "stream1",
  "remoteStreamName": "stream2",
  "video": true,
  "audio": true,
  "muteIfAbsent": 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
409	Conflict
500	Internal error

### **/stream/inject2/find\_all**

Найти все вставки на сервере

#### REQUEST EXAMPLE

```
POST /rest-api/stream/inject2/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

[
```

```

{
  "streamName": "test",
  "videoInjectorInfo": {
    "targetStreamName": "test2",
    "rootStreamName": "test2",
    "startTime": 1683344295099
  },
  "audioInjectorInfo": {
    "targetStreamName": "test2",
    "rootStreamName": "test2",
    "startTime": 1683344295056
  }
}
]

```

#### RETURN CODES

Code	Reason
200	OK
404	Not found

#### **/stream/inject2/terminate**

Остановить вставку в поток stream1

#### REQUEST EXAMPLE

```

POST /rest-api/stream/inject2/terminate HTTP/1.1
Host: localhost:8081
Content-Type: application/json

{
  "localStreamName": "stream1",
  "video": true,
  "audio": 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

Code	Reason
404	Not found
500	Internal error

## Parameters

Name	Description	Example
localStreamName	Имя потока, в который производится вставка	<code>stream1</code>
remoteStreamName	Имя потока, который будет вставлен	<code>stream2</code>
video	Заменять видео составляющую при вставке потока	<code>true</code>
audio	Заменять аудио составляющую при вставке потока	<code>true</code>
mutelfAbsent	Заменять составляющую, которой нет в исходном потоке, на темноту или тишину	<code>true</code>
videoInjectorInfo	Информация о видео и з вставленного потока	<pre>{   "targetStreamName": "stream2",   "rootStreamName": "stream2",   "startTime": 1683344295099 }</pre>



Name	Description	Example
audioInjectorInfo	Информация об аудио из вставленного потока	<pre>{    "targetStreamName"   : "stream2",    "rootStreamName":   "stream2",   "startTime":   1683344295056 }</pre>

## Вставка VOD потока из файла

В сборке [5.2.1719](#) добавлена возможность вставить VOD поток непосредственно из файла при отправке запроса `/stream/inject2/startup:`

```
{  
  "localStreamName": "host",  
  "remoteStreamName": "vod-live://advertising.mp4",  
  "video": true,  
  "audio": true  
}
```

При этом вставляемый файл начинает проигрываться без пауз, с первого ключевого кадра. Если этот же файл вставить в другой поток, в том потоке файл также начнет проигрываться с начала

Эта возможность полезна, например, при вставке рекламных роликов в поток, который смотрят зрители.

## Краткое руководство по тестированию

1. Для тестирования используем:

- WCS-сервер;
- Веб-приложение Media Devices для публикации потоков;
- Две веб-камеры, либо два различных ПК для публикации потоков;
- Веб-приложение Player для воспроизведения потока;
- браузер Chrome и [REST-клиент](#) для отправки запросов на сервер

2. Откройте приложение Media Devices, опубликуйте поток `test` разрешением 640x360

**Send Video**

**Cam**  ▼

**Screen share**

**Size**

**FPS**

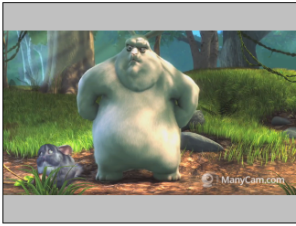
### Media Devices

**Video stats**  
Codec: H264  
Codec Rate: 90000  
Fir Count: 0  
Pll Count: 1  
Nack Count: 0  
Packets Sent: 361  
Bytes Sent: 270747  
Height: 360  
Width: 640  
Bitrate: 336728

**Audio stats**  
Codec: opus  
Codec Rate: 48000  
Packets Sent: 398  
Bytes Sent: 32444  
Bitrate: 32496


**Connection**

Local



640x360

Player



**Video stats**  
**Audio stats**  
**Connection**

PUBLISHING

**Timeout**

ESTABLISHED

3. Проиграйте поток `test` в примере Player

The screenshot shows a video player interface. At the top, the word "Player" is centered. Below it, a dark blue background displays white text for credits, organized into categories: COLORGUIDE ARTWORK (Enrico Valenza), ANIMATIC EDITING (Sacha Goedegebure, William Reynish, Enrico Valenza), CHARACTER DESIGN (Sacha Goedegebure), CHARACTER MODELING (Andreas Goralczyk), CHARACTER RIGGING (Nathan Vegdahl), CHARACTER ANIMATION (William Reynish, Nathan Vegdahl, Sacha Goedegebure, Andreas Goralczyk, Enrico Valenza), CHARACTER ANIMATION, SECOND UNIT (Nathan Dunlap, Daniel M. Lara, Bassam Kurdali, Claudio Andaur, Lee Salvemini), and MATTE PAINTING. A "ManyCam.com" logo is visible in the bottom right of the video area. Below the video, there are playback controls: "WCS URL" with a text box containing "wss://test1.flashphoner.com:844...", "Stream" with a text box containing "test", "Volume" with a slider, "Full Screen" with a button icon, and a "PLAYING" indicator with a "Stop" button.

4. Опубликуйте поток `adv` в примере Media Devices, используя другую вкладку браузера, другую камеру или другой ПК

**Send Video**

**Cam** OBS Virtual Camera ▼

Switch

**Screen share** off

**Size** 640 360

**FPS** 30

### Media Devices

**Video stats**

Codec: H264  
 Codec Rate: 90000  
 Fir Count: 0  
 Pli Count: 3  
 Nack Count: 0  
 Packets Sent: 781  
 Bytes Sent: 417431  
 Height: 360  
 Width: 640  
 Bitrate: 232864

**Audio stats**

Codec: opus  
 Codec Rate: 48000  
 Packets Sent: 905  
 Bytes Sent: 68422  
 Bitrate: 31760

**Connection**

Local

1:20

640x360

adv Stop

Player

5172

Play

PUBLISHING

wss://test1.flashphoner.com:8443 Disconnect

Timeout 1000 msec

ESTABLISHED

Video stats

Audio stats

Connection

5. Откройте REST-клиент, отправьте запрос `/stream/inject/startup`

Method: **POST** URL: `http://test1.flashphoner.com:8081/rest-api/stream/inject/startup` SEND

**HEADERS** **BODY** AUTHORIZATION VARIABLES

```

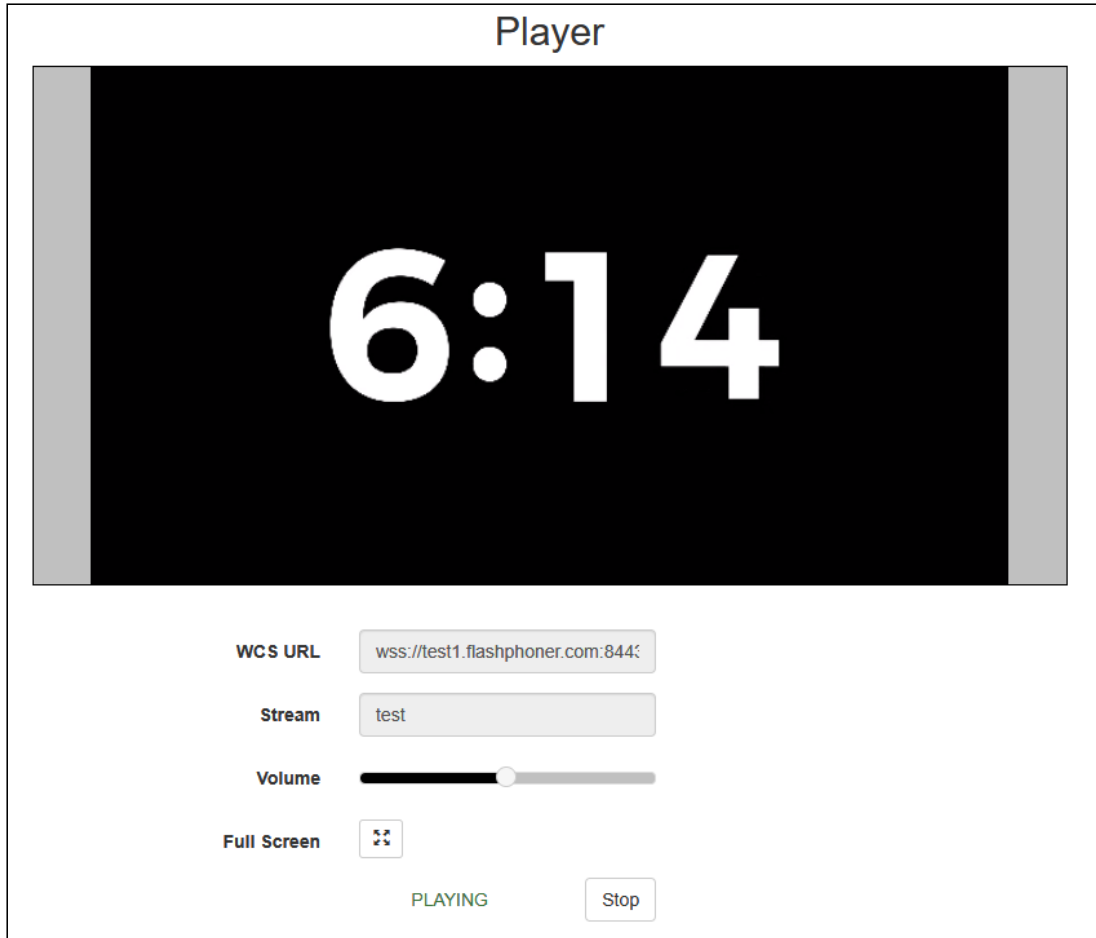
1 {
2   "localStreamName": "test",
3   "remoteStreamName": "adv"
4 }
```

**Response** 200 OK 91 B 69 ms

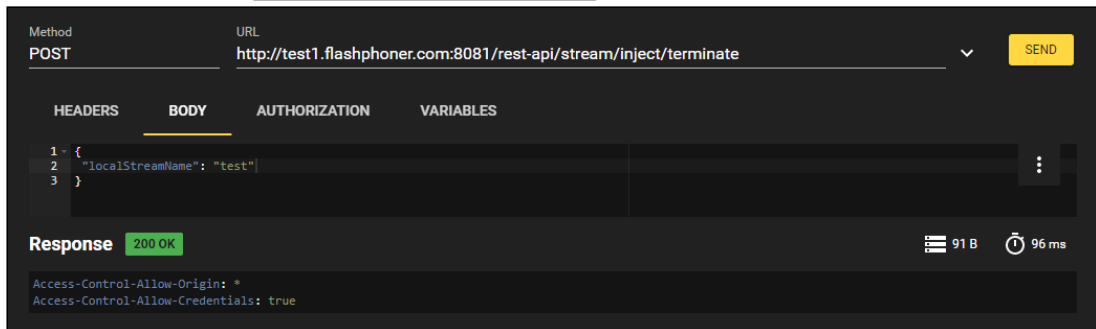
```

Access-Control-Allow-Origin: *
Access-Control-Allow-Credentials: true
```

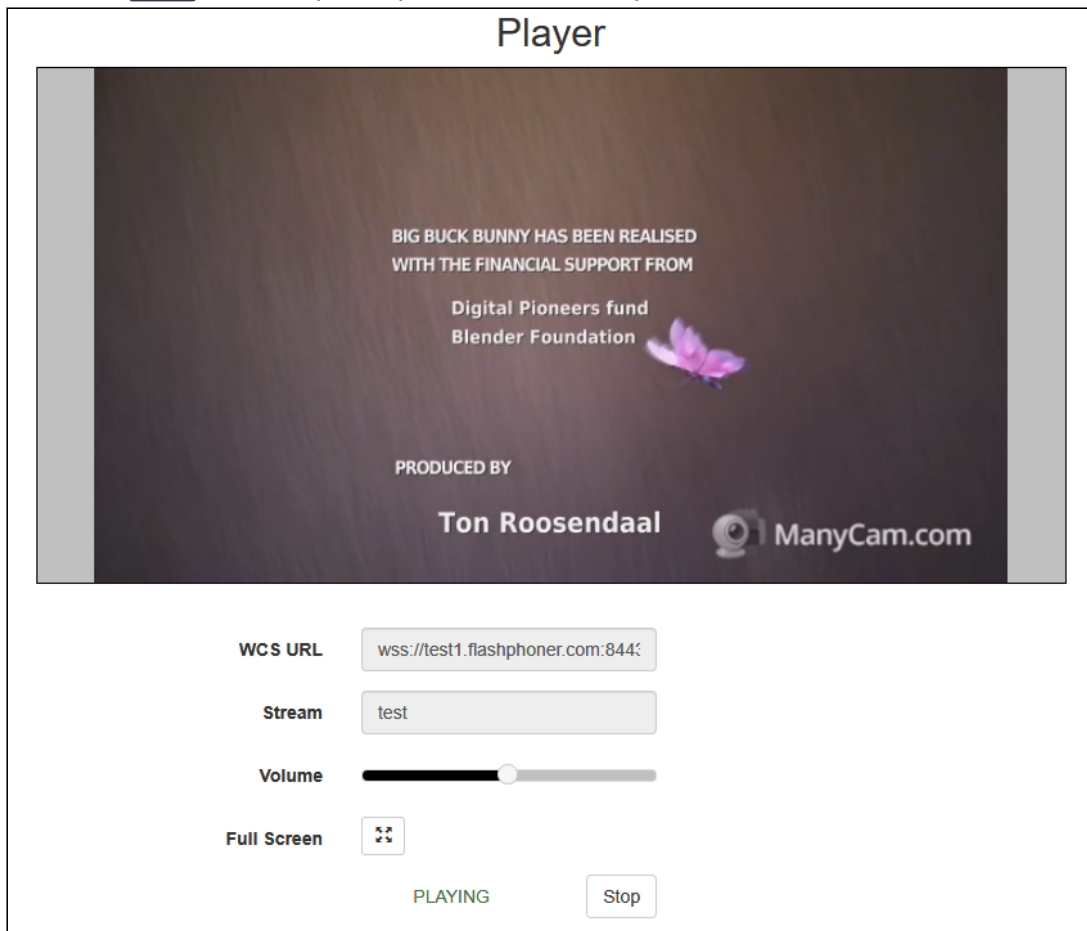
6. В потоке test воспроизводится содержимое потока `adv`



7. Отправьте запрос `/stream/inject/terminate`



8. В потоке `test` вновь играет оригинальное содержимое



## Известные проблемы

1. По окончании вставки одного RTMP потока в другой может теряться синхронизация между аудио и видео а оригинальном потоке

### Симптомы

При вставке RTMP потока в другой RTMP поток, по окончании вставки оригинальный поток играет с рассинхронизацией аудио и видео

### Решение

Включить буферизацию входящих RTMP потоков

```
rtmp_in_buffer_enabled=true
```