# Передача события, привязанного к потоку

## Описание

В сборке 5.2.935 добавлена возможность отправки с клиента события, привязанного к публикуемому потоку, и передачи этого события всем подписчикам. В настоящее время эта возможность используется для оповещения подписчиков о том, заглушено ли аудио/видео на публикующей стороне.

## Отправка события с публикующего клиента

### Отправка оповещения о статусе аудио/видео в потоке: заглушено/не заглушено

Оповещения об изменении состояния аудио отсылаются следующим образом, при вызове функций `Stream.muteAudio()` и `Stream.unmuteAudio()`:

```
var muteAudio = function muteAudio() {
    if (mediaConnection) {
    mediaConnection.muteAudio();
    sendStreamEvent(STREAM_EVENT_TYPE.AUDIO_MUTED);
    }
};
...
var unmuteAudio = function unmuteAudio() {
    if (mediaConnection) {
    mediaConnection.unmuteAudio();
    sendStreamEvent(STREAM_EVENT_TYPE.AUDIO_UNMUTED);
    }
};
```

Аналогично, при вызовах `Stream.muteVideo()` и `Stream.unmuteVideo()` отсылаются события об изменении состояния видео:

```
var muteVideo = function muteVideo() {
    if (mediaConnection) {
    mediaConnection.muteVideo();
    sendStreamEvent(STREAM_EVENT_TYPE.VIDEO_MUTED);
    }
};
...
var unmuteVideo = function unmuteVideo() {
    if (mediaConnection) {
    mediaConnection.unmuteVideo();
    sendStreamEvent(STREAM_EVENT_TYPE.VIDEO_UNMUTED);
    }
};
```

### Отправка данных всем подписчикам потока

В сборке WCS 5.2.942 и сборке WebSDK 2.0.168 добавлена возможность отправки любых данных с публикующего клиента в формате JSON всем подписчикам опубликованного потока. Для этого необходимо вызвать метод `Stream.sendData()`, например

```
stream.sendData({"number":33,"string":"hello",boolean:true});
```

## Отправка события подписчикам потока с сервера В сборке WCS [5.2.944] (https://flashphoner.com/downloads/builds/WCS/5.2/FlashphonerWebCallServer-5.2.944.tar.gz) добавлена возможность отправки события всем подписчикам потока с сервера по REST API.  REST-запрос должен быть HTTP/HTTPS POST запросом в таком виде: - HTTP: `http://test.flashphoner.com:8081/rest-api/stream/event/send` - HTTPS: `https://test.flashphoner.com:8444/rest-api/stream/event/send` Здесь: - `test.flashphoner.com` - адрес WCS-сервера - `8081` - стандартный REST / HTTP порт WCS-сервера - `8444` - стандартный HTTPS порт - `rest-api` - обязательная часть URL - `/stream/event/send` - используемый REST-метод
### REST-методы и статусы ответа
#### /stream/event/send Отправить данные всем подписчикам потока ##### Request example

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

{
    "streamName":"test",
    "payload":{
        "number":33,
```

```
        "string":"hello",
        "boolean":true
    }
}
```

##### 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 |
### Параметры

| Параметр | Описание | Пример |
|---|---|---|
| streamName | Имя потока | `test` |
| payload | Данные в формате JSON | <pre>{<br>"number":33,<br>"string":"hello",<br>"boolean":true<br>}</pre> |

Если поток опубликован на сервере, но не имеет ни одного подписчика, запрос вернет `200 OK`, но событие никому не будет отослано
## Получение события на стороне подписчика При передаче события, сигнализирующего об изменении состояния потока, подписчик получает событие `STREAM_EVENT` [code]
(https://github.com/flashphoner/flashphoner_client/blob/3cc75f99f8c83206abe6efb0719dc85b99c0dded/examples/demo/streaming/media_

```
previewStream = session.createStream({
    name: streamName,
    display: remoteVideo,
    ...
}).on(STREAM_EVENT, function(streamEvent) {
    switch (streamEvent.type) {
        case STREAM_EVENT_TYPE.AUDIO_MUTED:
            $("#audioMuted").text(true);
            break;
        case STREAM_EVENT_TYPE.AUDIO_UNMUTED:
            $("#audioMuted").text(false);
            break;
        case STREAM_EVENT_TYPE.VIDEO_MUTED:
            $("#videoMuted").text(true);
            break;
        case STREAM_EVENT_TYPE.VIDEO_UNMUTED:
            $("#videoMuted").text(false);
            break;

    }
    console.log("Received streamEvent ", streamEvent.type);
});
```

В сборке WCS [5.2.942](https://flashphoner.com/downloads/builds/WCS/5.2/FlashphonerWebCallServer-5.2.942.tar.gz) и сборке WebSDK [2.0.168](https://flashphoner.com/downloads/builds/flashphoner_client/wcs_api-2.0/flashphoner-api-2.0.168-a8c61f9ba0f76ff2f159a4ca4cf1f355c27f59e1.tar.gz) добавлен тип `STREAM_EVENT`_TYPE.DATA` для получения данных в формате JSON, отосланных функцией `Stream.sendData()` или REST запросом `/stream/send/event`

```
session.createStream({
    name: streamName,
    display: remoteVideo
    ...
}).on(STREAM_EVENT, function(streamEvent) {
    switch (streamEvent.type) {
        case STREAM_EVENT_TYPE.DATA:
            console.log(JSON.stringify(streamEvent.payload));
            break;
    }
}).play();
```

### Получение события, связанного со входящим потоком микшера Начиная со сборки [5.2.966]
(https://flashphoner.com/downloads/builds/WCS/5.2/FlashphonerWebCallServer-5.2.966.tar.gz), подписчик, играющий выходной поток [микшера](../Stream_mixer_functions/Stream_mixer.ru.md), получает события, связанные с одним из входящих потоков микшера. При этом в объект `payload` добавляется поле `streamName`, чтобы показать, к какому именно потоку относится событие

```
session.createStream({
    name: streamName,
    display: remoteVideo,
    ...
}).on(STREAM_EVENT, function(streamEvent) {
    let mutedName="";
    if(streamEvent.payload !== undefined) {
        mutedName=streamEvent.payload.streamName;
    }
    switch (streamEvent.type) {
        case STREAM_EVENT_TYPE.AUDIO_MUTED:
            $("#audioMuted").text(true + " " + mutedName);
            break;
        case STREAM_EVENT_TYPE.AUDIO_UNMUTED:
            $("#audioMuted").text(false + " " + mutedName);
            break;
        case STREAM_EVENT_TYPE.VIDEO_MUTED:
            $("#videoMuted").text(true + " " + mutedName);
            break;
        case STREAM_EVENT_TYPE.VIDEO_UNMUTED:
            $("#videoMuted").text(false + " " + mutedName);
            break;

    }
    console.log("Received streamEvent ", streamEvent.type);
}).play();
```

## Определение статуса потока при подключении к потоку При подключении к потоку, подписчик может определить, заглушена аудио/видео дорожка в потоке или нет, при помощи методов `Stream.getAudioState()` и `Stream.getVideoState()` в обработчике события `STREAM_STATUS.PLAYING`:

```
session.createStream({
    name: streamName,
    display: remoteVideo,
    ...
}).on(STREAM_STATUS.PLAYING, function (stream) {
    if (stream.getAudioState()) {
        $("#audioMuted").text(stream.getAudioState().muted);
    }
    if (stream.getVideoState()) {
        $("#videoMuted").text(stream.getVideoState().muted);
    }
    ...
}).play;
```

### Определение статусов входящих потоков при подключении к выходному потоку микшера Начиная со сборки WCS [5.2.1011] (https://flashphoner.com/downloads/builds/WCS/5.2/FlashphonerWebCallServer-5.2.1011.tar.gz), при подключении подписчика к выходному потоку микшера, он получает набор [событий `STREAM_EVENT`](#receiving-mixer-incoming-stream-event) на каждый входящий поток микшера, если хотя бы в одном из них аудио или видео было заглушено. При этом порядок получения этих событий не гарантируется и не зависит от порядка добавления потоков в микшер.
## Обработка события на бэкенде Для того, чтобы обработать событие об изменении состояния потока на бэкенде, к [REST hook приложению](../REST_Hooks/Controlling_REST_methods.ru.md) должны быть [добавлены] (../Working_with_the_server/Command_line_interface/Applications_management.ru.md) методы `sendStreamEvent` и `StreamEvent`

```
add app-rest-method MyAppKey sendStreamEvent
add app-rest-method MyAppKey StreamEvent
```

### Оповещение о заглушенном аудио/видео Если аудио или видео было заглушено, бэкенд-сервер получит событие `sendStreamEvent`

```
URL:http://localhost:8081/apps/EchoApp/sendStreamEvent
OBJECT:
{
  "nodeId" : "qg4BeHzYSAtkhUkXgnSMEUZpsshaLPL5@192.168.0.39",
  "appKey" : "defaultApp",
  "sessionId" : "/192.168.0.83:64573/192.168.0.39:8443-a98bb891-aeaf-46a8-8fba-772e07ac035b",
  "mediaSessionId" : "9906b2b0-9c28-11eb-8d20-75f877676678",
  "type" : "audioMuted",
  "origin" : "https://wcs:8888"
}
```

Также бэкенд-сервер получит событие `StreamEvent` для каждого подписчика`этого потока

```
URL:http://localhost:8081/apps/EchoApp/StreamEvent
OBJECT:
{
  "nodeId" : "qg4BeHzYSAtkhUkXgnSMEUZpsshaLPL5@192.168.0.39",
  "appKey" : "defaultApp",
  "sessionId" : "/192.168.0.83:64573/192.168.0.39:8443-a98bb891-aeaf-46a8-8fba-772e07ac035b",
```

```
  "mediaSessionId" : "9fed5c50-9c28-11eb-8d20-75f877676678",
  "type" : "audioMuted"
}
```

### Оповещение о данных, переданных подписчикам Если подписчикам потока были отправлены данные со стороны публикующего клиента или с сервера, бэкенд-сервер получит событие `sendStreamEvent`

```
URL:http://localhost:8081/apps/EchoApp/sendStreamEvent
OBJECT:
{
  "nodeId" : "qg4BeHzYSAtkhUkXgnSMEUZpsshaLPL5@192.168.0.39",
  "appKey" : "defaultApp",
  "sessionId" : "/192.168.0.83:64573/192.168.0.39:8443-a98bb891-aeaf-46a8-8fba-772e07ac035b",
  "mediaSessionId" : "9906b2b0-9c28-11eb-8d20-75f877676678",
  "type" : "data",
  "payload" : {
    "count" : 23
  },
  "origin" : "https://wcs:8888"
}
```

Также бэкенд-сервер получит событие `StreamEvent` для каждого подписчика этого потока

```
URL:http://localhost:8081/apps/EchoApp/StreamEvent
OBJECT:
{
  "nodeId" : "qg4BeHzYSAtkhUkXgnSMEUZpsshaLPL5@192.168.0.39",
  "appKey" : "defaultApp",
  "sessionId" : "/192.168.0.83:64573/192.168.0.39:8443-a98bb891-aeaf-46a8-8fba-772e07ac035b",
  "mediaSessionId" : "9fed5c50-9c28-11eb-8d20-75f877676678",
  "type" : "data",
  "payload" : {
    "count" : 23
  }
}
```