



# Stream Diagnostic

## Пример публикации и воспроизведения потока с выводом отладочной информации

Пример демонстрирует возможности получения отладочной информации и вывода ее на страницу. Отладочный лог и соответствующее событие в сессии можно получить только при условии, что параметр `sessionDebugEnabled` в файле настроек `wcs-core.properties` установлен в `true`, при этом требуется перезапуск сервера.

Пример вывода диагностической информации при публикации потока

### Stream Diagnostic

➔


LocalPreview

PUBLISHING

```
Create new session with url ws://localhost:8080/test
Debug session STARTED
Stream test PUBLISHING
Stream test PLAYING
```

и при завершении публикации

### Stream Diagnostic



Local

➔

Preview

UNPUBLISHED [Download debug logs](#)

```
Create new session with url ws://localhost:8080/test
Debug session STARTED
Stream test PUBLISHING
Stream test PLAYING
Stream test STOPPED
Stream test UNPUBLISHED
Debug session STOPPED
Debug session PASSED
```

## Код примера

Код данного примера находится на WCS-сервере по следующему пути:

*/usr/local/FlashphonerWebCallServer/client2/examples/demo/streaming/stream-diagnostic*

- stream-diagnostic.css - файл стилей
- stream-diagnostic.html - страница примера
- stream-diagnostic.js - скрипт, обеспечивающий работу примера

Тестировать данный пример можно по следующему адресу:

*https://host:8888/client2/examples/demo/streaming/stream-diagnostic/stream-diagnostic.html*

Здесь host - адрес WCS-сервера.

## Работа с кодом примера

Для разбора кода возьмем версию файла `stream-diagnostic.js` с хешем `ecbadc3`, которая находится [здесь](#) и доступна для скачивания в соответствующей сборке [2.0.212](#).

## 1. Инициализация API

`Flashphoner.init()` [code](#)

```
Flashphoner.init({createMicGainNode: false});
```

## 2. Подключение к серверу

`Flashphoner.createSession()` [code](#)

Действие выводится в специальный элемент на странице при помощи функции логирования `log()`

```
var url = field('url');
log("Create new session with url " + url);
$('#url').prop('disabled', true);
session = Flashphoner.createSession({urlServer:
url}).on(SESSION_STATUS.ESTABLISHED, function(session){
    ...
});
```

## 3. Получение от сервера события, подтверждающего успешное соединение

`ConnectionStatusEvent ESTABLISHED` [code](#)

```
session = Flashphoner.createSession({urlServer:
url}).on(SESSION_STATUS.ESTABLISHED, function(session){
    //session connected, start streaming
    startStreaming(session);
}).on(SESSION_STATUS.DISCONNECTED, function(){
    ...
}).on(SESSION_STATUS.FAILED, function(){
    ...
}).on(SESSION_STATUS.DEBUG, function(event){
    ...
});
```

## 4. Запуск отладочного вывода сессии и публикация видеопотока

`Session.startDebug()`, `Session.createStream()`, `Stream.publish()` [code](#)

При создании передаются:

- `streamName` - имя видеопотока
- `localVideo` - `div` элемент, в котором будет отображаться видео с камеры

```

session.startDebug();
session.createStream({
  name: streamName,
  display: localVideo,
  cacheLocalResources: true,
  receiveVideo: false,
  receiveAudio: false
  ...
}).publish();

```

## 5. Получение от сервера события, подтверждающего успешную публикацию потока

`StreamStatusEvent PUBLISHING` [code](#)

При получении данного события создается превью-видеопоток при помощи `Session.createStream()` и вызывается `Stream.play()` для его воспроизведения.

```

session.createStream({
  ...
}).on(STREAM_STATUS.PUBLISHING, function(publishStream){
  log("Stream " + streamName + " " + STREAM_STATUS.PUBLISHING);
  setStatus(STREAM_STATUS.PUBLISHING);
  //play preview
  session.createStream({
    name: streamName,
    display: remoteVideo
    ...
  }).play();
}).on(STREAM_STATUS.UNPUBLISHED, function(){
  ...
}).on(STREAM_STATUS.FAILED, function(stream){
  ...
}).publish();

```

## 6. Остановка воспроизведения видеопотока

`Stream.stop()` [code](#)

```

function onStarted(publishStream, previewStream) {
  $("#publishBtn").text("Stop").off('click').click(function(){
    $(this).prop('disabled', true);
    previewStream.stop();
  }).prop('disabled', false);
  $("#downloadDiv").hide();
}

```

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

`StreamStatusEvent STOPPED` [code](#)

```
session.createStream({
  name: streamName,
  display: remoteVideo
}).on(STREAM_STATUS.PLAYING, function(previewStream){
  ...
}).on(STREAM_STATUS.STOPPED, function(){
  log("Stream " + streamName + " " + STREAM_STATUS.STOPPED);
  publishStream.stop();
}).on(STREAM_STATUS.FAILED, function(stream){
  ...
}).play();
```

## 8. Остановка публикации видеопотока после остановки воспроизведения превью-потока

`Stream.stop()` [code](#)

```
session.createStream({
  ...
}).on(STREAM_STATUS.PLAYING, function(previewStream){
  ...
}).on(STREAM_STATUS.STOPPED, function(){
  log("Stream " + streamName + " " + STREAM_STATUS.STOPPED);
  publishStream.stop();
}).on(STREAM_STATUS.FAILED, function(stream){
  ...
}).play();
```

## 9. Получение от сервера события, подтверждающего остановку публикации потока

`StreamStatusEvent UNPUBLISHED` [code](#)

```
session.createStream({
  ...
}).on(STREAM_STATUS.PUBLISHING, function(publishStream){
  ...
}).on(STREAM_STATUS.UNPUBLISHED, function(){
  setStatus(STREAM_STATUS.UNPUBLISHED);
  log("Stream " + streamName + " " + STREAM_STATUS.UNPUBLISHED);
  //enable start button
  onStoped();
}).on(STREAM_STATUS.FAILED, function(stream){
  ...
}).publish();
```

## 10. Остановка отладочного вывода сессии после остановки публикации

`Session.stopDebug()` [code](#)

```
function onStopped() {  
    ...  
    if (session)  
        session.stopDebug();  
}
```

## 11. Получение от сервера события, сигнализирующего о завершении отладочного вывода сессии.

`ConnectionStatusEvent.DEBUG` [code](#)

При получении данного события формируется ссылка на файл лога сессии

```
session = Flashphoner.createSession({urlServer:  
url}).on(SESSION_STATUS.ESTABLISHED, function(session){  
    ...  
}).on(SESSION_STATUS.DISCONNECTED, function(){  
    ...  
}).on(SESSION_STATUS.FAILED, function(){  
    ...  
}).on(SESSION_STATUS.DEBUG, function(event){  
    log("Debug session " + event.status);  
    if (event.file) {  
        var link = window.location.protocol + "://" + window.location.host +  
"/" + event.file;  
        $("#link").attr("href", link);  
        $("#downloadDiv").show();  
    }  
});
```

## 12. Вывод отладочной информации на страницу

[code](#)

```
function log(string) {  
    document.getElementById("debug").innerHTML += string + '<br>';  
}
```