

# Описание

- [Скачать](#)
- [Работа с кодом примеров на вашем Web-сервере](#)
- [Работа с кодом примеров прямо на WCS сервере](#)
- [Исходный код API и примеров на Github](#)
- [Зависимости](#)
- [Известные проблемы](#)

Для web-разработки приложений потокового видео и звонков используется Web SDK. Это набор скриптов и примеров для работы с WCS-сервером.

## Скачать

Скачать Web SDK

<https://docs.flashphoner.com/display/WEBSDK2RU/Web+SDK+release+notes>

Скачать модуль JavaScript

[https://flashphoner.com/downloads/builds/flashphoner\\_client/wcs\\_api-2.0/current/flashphoner.js](https://flashphoner.com/downloads/builds/flashphoner_client/wcs_api-2.0/current/flashphoner.js)

API документация

<http://flashphoner.com/docs/api/WCS5/client/web-sdk/latest>

## Работа с кодом примеров на вашем Web-сервере

Чтобы работать с демо - примерами на вашем web-сервере, используйте последнюю доступную сборку web-клиента, входящую в комплект поставки WCS, а также доступную здесь

[Web SDK release notes](#)

Сборка выглядит следующим образом:

doc - JavaScript API документация

examples - демо примеры

flashphoner.js - основной файл API, который нужно будет добавить на вашу web-страницу, включает все поддерживаемые технологии.

flashphoner-webrtc-only.js - альтернативный файл API, если Вы планируете использовать только WebRTC

flashphoner-no-flash.js - альтернативный файл API, если вы не планируете использовать Flash

flashphoner-no-webrtc.js - альтернативный файл API, если вы не планируете использовать WebRTC

flashphoner-no-wsplayer.js - альтернативный файл API, если вы не планируете использовать Websocket плеер

media-provider.swf - файл для поддержки работы с Flash

## Работа с кодом примеров прямо на WCS сервере

Если у вас есть установленный Web Call Server, вы можете работать с кодом демо-примеров напрямую.

Код примеров находится по следующему пути:

`/usr/local/FlashphonerWebCallServer/client2/examples/demo`

Для тестирования отдельного примера откройте в браузере соответствующую страницу, например:

`https://host:8888/client2/examples/demo/streaming/player/player.html`

Здесь host - это адрес вашего WCS-сервера.

Таким образом, вы можете вносить необходимые изменения в скрипты и тестировать измененный демо-пример прямо на WCS сервере.

## Исходный код API и примеров на Github

[https://github.com/flashphoner/flashphoner\\_client/tree/wcs\\_api-2.0](https://github.com/flashphoner/flashphoner_client/tree/wcs_api-2.0)

Как правило, для разработки достаточно сборок, которые доступны по ссылкам выше.

Исходный код скорее всего не потребуется и может использоваться в крайних случаях, когда нужен быстрый фикс на уровне API.

В данной документации мы будем использовать код для пояснения работы примеров.

Например так: [line 3](#) можно сослаться на третью строку исходного кода файла package.json с хэшем ревизии 0b891b8.

## ЗАВИСИМОСТИ

WebSDK собирается с использованием библиотеки [webrtc/adapter](#) версии не ниже 7.2.6. В связи с этим, следует избегать прямого использования данной библиотеки совместно с WebSDK.

## Известные проблемы

1. При открытии веб-приложения в WKWebView на iOS 11 и выше работает только воспроизведение по WSPlayer, и не работают публикация и воспроизведение потока по WebRTC

Симптомы: при открытии примера [Two-Way Streaming](#) в iOS-приложении, использующем WKWebView для просмотра ссылок на страницы (например, Telegram) не работают ни воспроизведение, ни публикация; в примере [Player](#) воспроизведение работает по WSPlayer.

Таким же образом будет работать при открытии веб-страницы с домашнего экрана, если код страницы содержит

```
<meta name="apple-mobile-web-app-capable" content="yes" />
```

Решение: использовать WKWebView только для веб-приложений, предназначенных для воспроизведения видеопотока, без вызова функции `getUserMedia()`, которая не поддерживается в WKWebView.

При необходимости использования WebRTC удалить из кода страницы `<meta name="apple-mobile-web-app-capable" content="yes">`, чтобы для ее открытия использовался Safari.