

Звонки в браузере с поддержкой WebRTC

- Описание
 - Поддерживаемые платформы и браузеры
 - Поддерживаемые протоколы
 - Поддерживаемые кодеки
 - Поддерживаемые SIP функции
 - Схема работы
- Последовательность выполнения операций (Call Flow)
- Исходящий звонок из браузера на SIP-устройство
- Прием входящего звонка с SIP-устройства в браузере
- Управление камерой, микрофоном и устройствами вывода звука
 - Выбор и переключение устройств ввода и вывода
 - Установка размера видео
- Совершение звонка без микрофона и камеры
- Отображение WebRTC-статистики
- Настройка используемых кодеков
- Передача дополнительных параметров в SDP в запросе INVITE и ответе 200 OK
- Звонки с использованием SIP TLS сигналинга
 - Звонки через сервер с самоподписанным сертификатом
- Подключение к существующей сессии
- Известные проблемы

Web Call Server поддерживает аудио и видеозвонки из браузера на SIP устройства, PBX серверы, SIP-GSM-шлюзы, VoIP конференции и другие устройства с поддержкой протокола SIP. Таким образом, веб-приложение в браузере может работать, как программный телефон с поддержкой протокола SIP, принимать и инициировать голосовые и видеозвонки.

Описание

Поддерживаемые платформы и браузеры

	Chrome	Firefox	Safari 11	Edge
Windows	+	+		+
Mac OS	+	+	+	
Android	+	+		
iOS	-	-	+	

Поддерживаемые протоколы

- WebRTC
- RTP
- SIP

Поддерживаемые кодеки

- H.264
- VP8
- G.711
- Speex
- G.729
- Opus

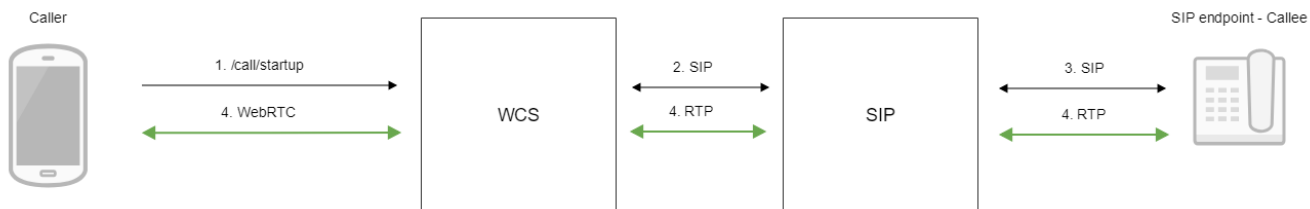
Поддерживаемые SIP функции

- DTMF
- Удержание звонка
- Перевод звонка

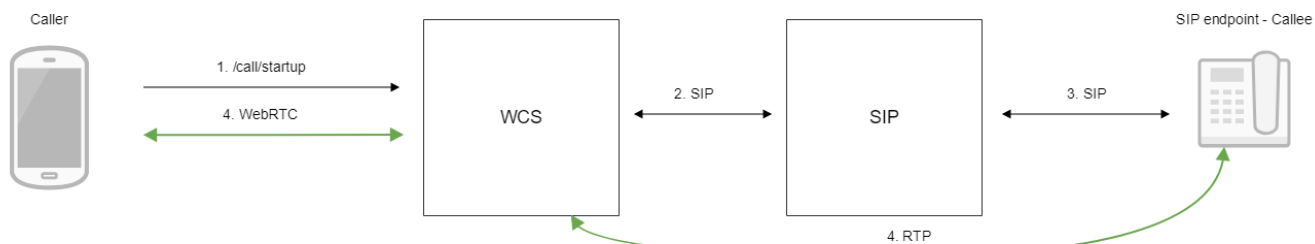
SIP функции управляются при помощи REST API.

Схема работы

1: SIP-сервер как прокси-сервер для передачи вызовов и RTP медиа



2: SIP-сервер только как сервер для передачи вызовов



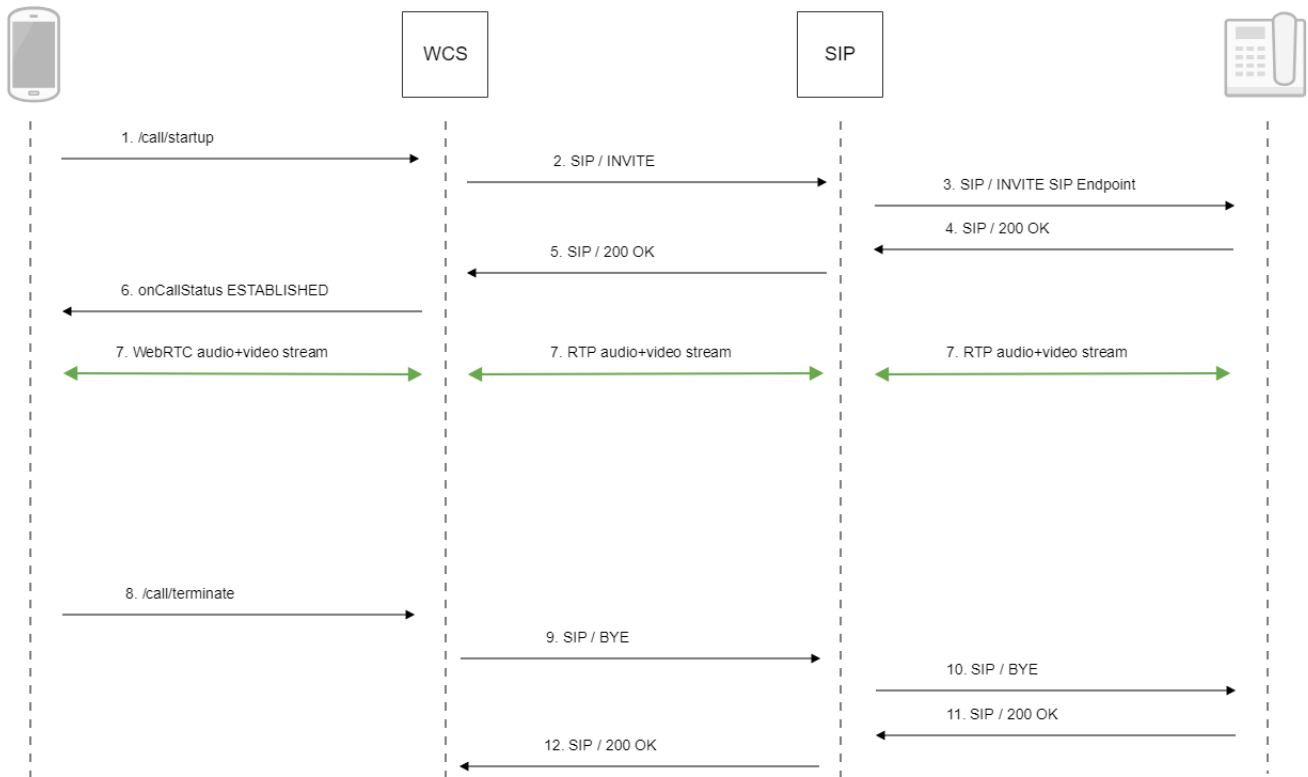
1. Браузер начинает звонок с помощью REST-вызова /call/startup
2. WCS соединяется с SIP-сервером
3. SIP-сервер соединяется с SIP-устройством, принимающим звонок
4. Браузер и SIP-устройство обмениваются аудио- и видеопотоками

Последовательность выполнения операций (Call Flow)

Ниже описана последовательность вызовов при использовании примера Phone для создания звонка

[phone.html](#)

[phone.js](#)



1. Отправка REST-запроса /call/startup при помощи JavaScript API:

session.createCall(), call.call()[code](#)

```

var outCall = session.createCall({
  callee: $("#callee").val(),
  visibleName: $("#sipLogin").val(),
  localVideoDisplay: localDisplay,
  remoteVideoDisplay: remoteDisplay,
  constraints: constraints,
  receiveAudio: true,
  receiveVideo: false
  ...
});

outCall.call();
  
```

2. Установка соединения с SIP-сервером

3. Установка соединения с адресатом

4. Получение подтверждения от SIP-устройства

5. Получение подтверждения от SIP-сервера

6. Получение от сервера события, подтверждающего успешное соединение.

CallStatusEvent ESTABLISHED[code](#)

```

var outCall = session.createCall({
    callee: $("#callee").val(),
    visibleName: $("#sipLogin").val(),
    localVideoDisplay: localDisplay,
    remoteVideoDisplay: remoteDisplay,
    constraints: constraints,
    receiveAudio: true,
    receiveVideo: false
}).on(CALL_STATUS.RING, function(){
    ...
}).on(CALL_STATUS.ESTABLISHED, function(){
    setStatus("#callStatus", CALL_STATUS.ESTABLISHED);
    $("#holdBtn").prop('disabled', false);
    onAnswerOutgoing();
}).on(CALL_STATUS.HOLD, function() {
    ...
}).on(CALL_STATUS.FINISH, function(){
    ...
}).on(CALL_STATUS.FAILED, function(){
    ...
});

outCall.call();

```

7. Стороны звонка обмениваются аудио- и видеопотоками

8. Завершение звонка

call.hangup()[code](#)

```

function onConnected(session) {
    $("#connectBtn, #connectTokenBtn").text("Disconnect").off('click').click(function(){
        $(this).prop('disabled', true);
        if (currentCall) {
            showOutgoing();
            disableOutgoing(true);
            setStatus("#callStatus", "");
            currentCall.hangup();
        }
        session.disconnect();
    }).prop('disabled', false);
}

```

9. Отправка команды на SIP-сервер

10. Отправка команды на SIP-устройство

11. Получение подтверждения от SIP-устройства

12. Получение подтверждения от SIP-сервера

Исходящий звонок из браузера на SIP-устройство

1. Для тестирования используем:

- два SIP-аккаунта;
- веб-приложение [Phone Video](#) для совершения звонка;
- программный телефон для ответа на звонок.

2. Откройте веб-приложение Phone Video. Введите данные SIP-аккаунта, звонящего из браузера:

Connection

WCS URL

wss://p11.flashphoner.com:8443

SIP Login

10006

SIP Auth Name

10006

SIP Password

.....

SIP Domain

yoursip.domain

**SIP Outbound
Proxy**

yoursip.domain

SIP Port

5060

**Register
required**



Connect

3. Запустите программный телефон, введите данные SIP-аккаунта, принимающего звонок:

Учетная запись Голосовая почта Топология Присутствие Транспорт Дополнительно

Имя учетной записи: Account 2

Протокол: SIP

Разрешить использование этой учетной записи для

- ☒ Вызов
- ☒ Чат / присутствие

Сведения о пользователе

* Идентификатор пользователя: 10005

* Домен: yoursip.domain

Пароль: ●●●●●

Отображаемое имя: 10005

Имя авторизации: 10005

Прокси-сервер домена

☒ Зарегистрироваться в домене и принимать вызовы

Отправлять исходящие через:

☒ Домен

☐ Прокси-сервер Адрес:

4. Нажмите в браузере кнопку Connect, будет установлено соединение с сервером. Затем введите идентификатор SIP-аккаунта, принимающего звонок, и нажмите кнопку Call:

SIP Port

5060

Register
required

☒

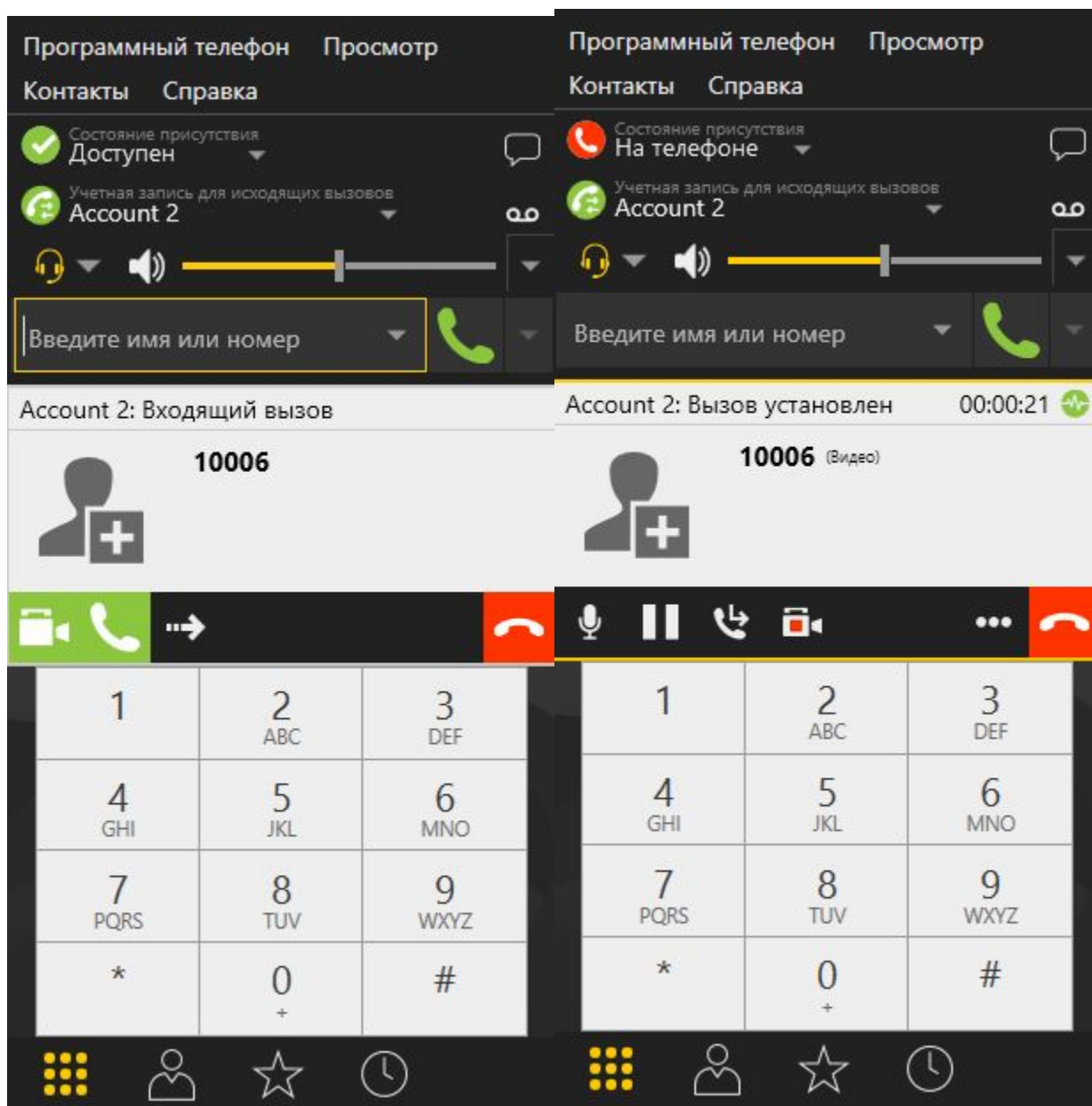
REGISTERED

Disconnect

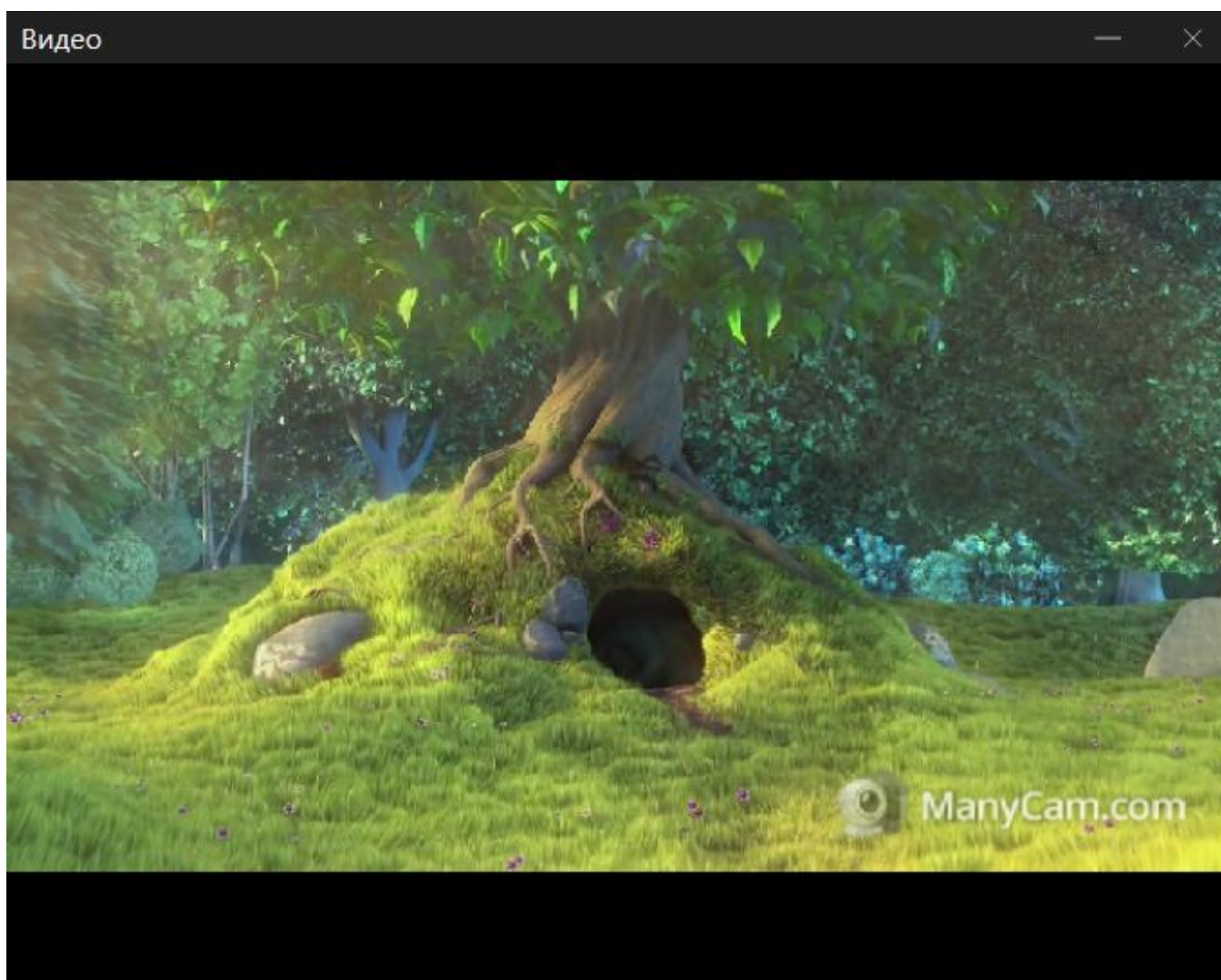
10005

Call

5. Примите звонок в программном телефоне, нажав кнопку ответа на звонок с использованием видео:



В отдельном окне отобразится видео, транслируемое из браузера:



6. В браузере также отобразится видео:



7. Для завершения звонка нажмите кнопку Hangup в браузере, либо кнопку завершения звонка в программном телефоне.

Прием входящего звонка с SIP-устройства в браузере

1. Для тестирования используем:

- два SIP-аккаунта;
- программный телефон для совершения звонка;
- веб-приложение [Phone Video](#) для ответа на звонок.

2. Откройте веб-приложение Phone Video. Введите данные SIP-аккаунта, принимающего звонок в браузере:

Connection

WCS URL

wss://p11.flashphoner.com:8443

SIP Login

10006

SIP Auth Name

10006

SIP Password

.....

SIP Domain

yoursip.domain

**SIP Outbound
Proxy**

yoursip.domain

SIP Port

5060

**Register
required**



Connect

3. Запустите программный телефон, введите данные звонящего SIP-аккаунта:

Учетная запись Голосовая почта Топология Присутствие Транспорт Дополнительно

Имя учетной записи: Account 2

Протокол: SIP

Разрешить использование этой учетной записи для

- ☒ Вызов
- ☒ Чат / присутствие

Сведения о пользователе

* Идентификатор пользователя: 10005

* Домен: yoursip.domain

Пароль: ●●●●●

Отображаемое имя: 10005

Имя авторизации: 10005

Прокси-сервер домена

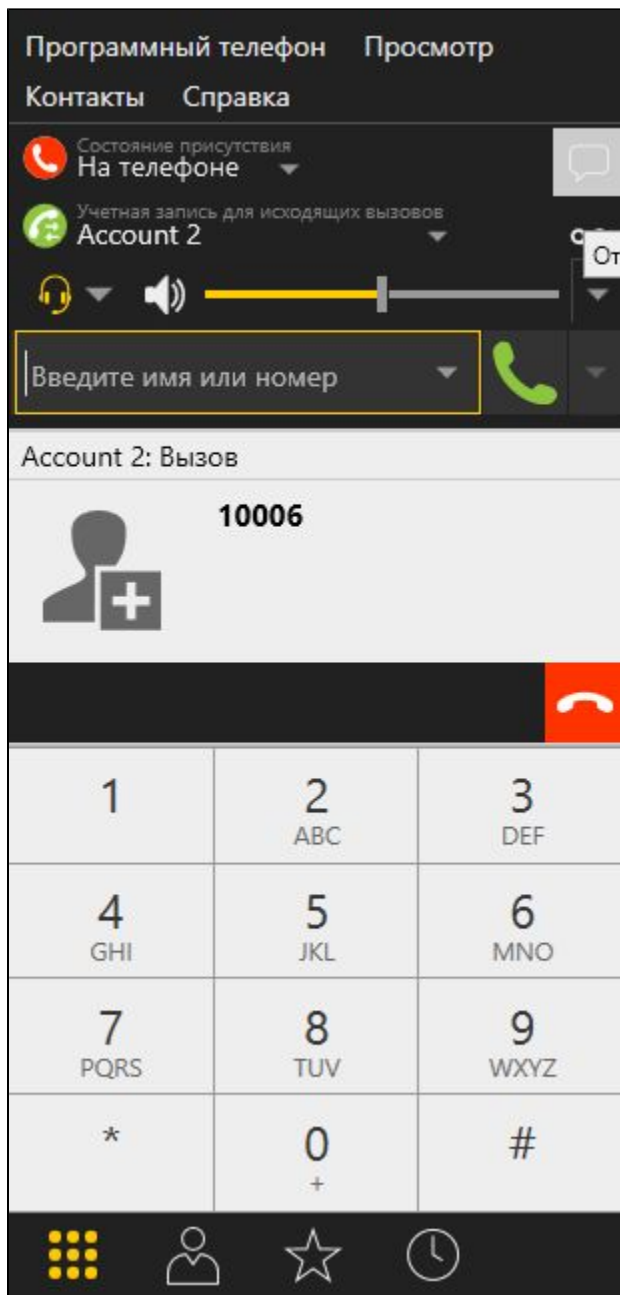
☒ Зарегистрироваться в домене и принимать вызовы

Отправлять исходящие через:

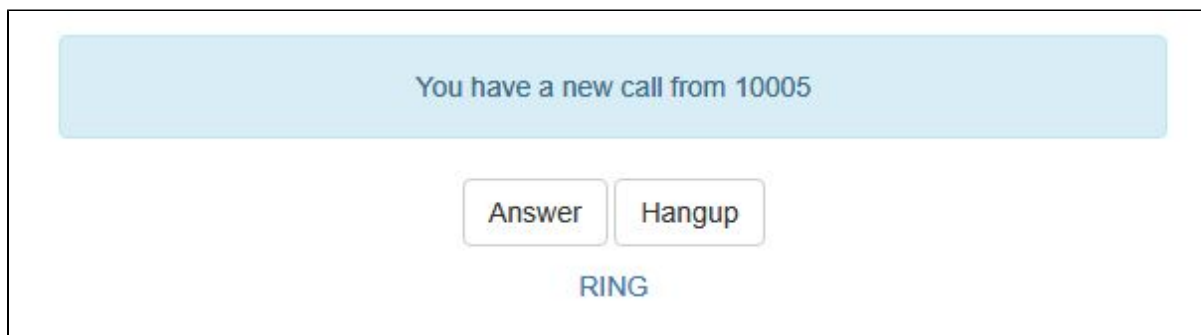
☒ Домен

☐ Прокси-сервер Адрес:

4. Нажмите в браузере кнопку Connect, будет установлено соединение с сервером. В программном телефоне введите идентификатор SIP-аккаунта, принимающего звонок, и нажмите кнопку вызова:



5. Примите звонок в браузере, нажав кнопку Answer:

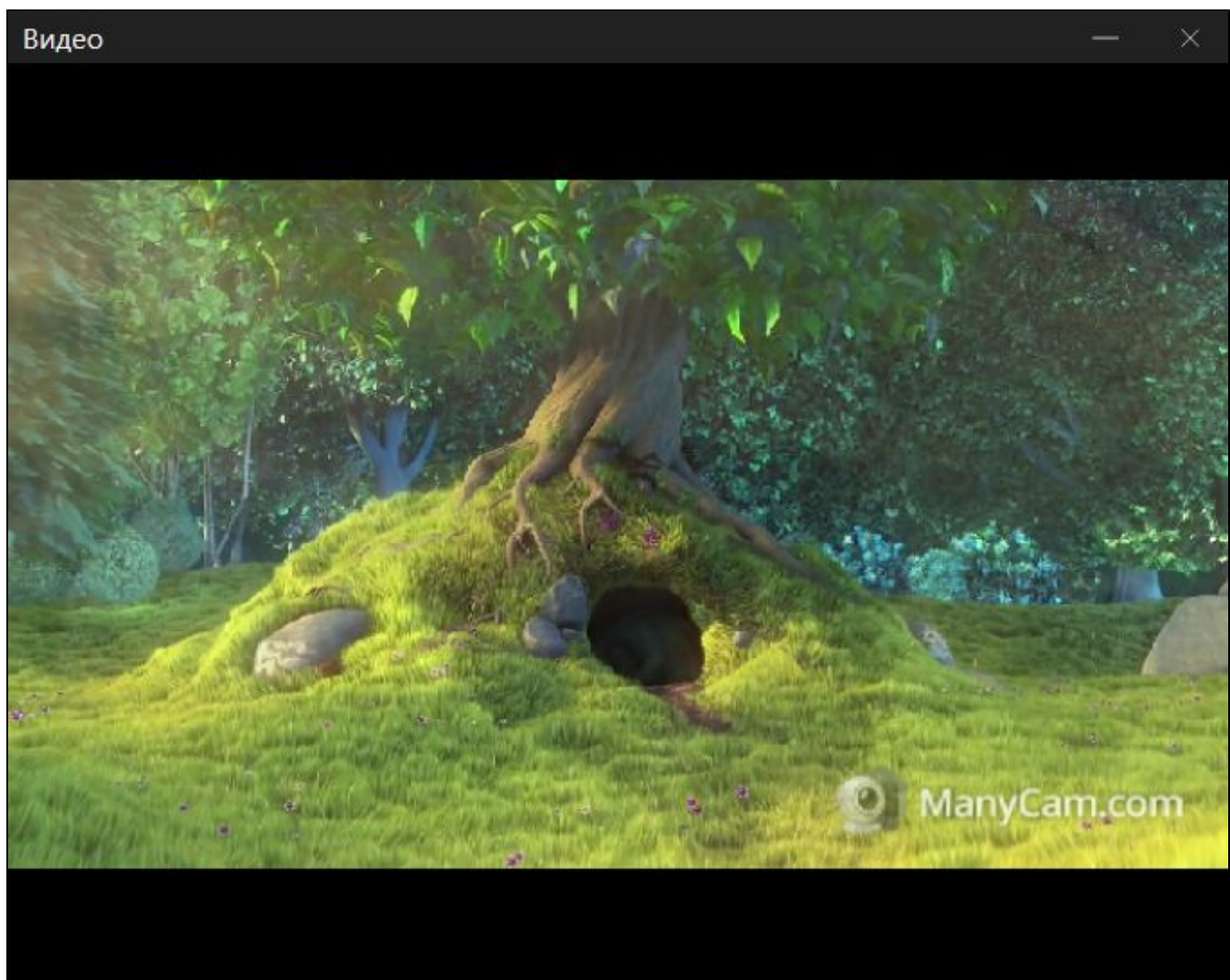




6. В браузере отобразится видео:



7. В отдельном окне программного телефона также отобразится видео, транслируемое из браузера:



8. Для завершения звонка нажмите кнопку Hangup в браузере, либо кнопку завершения звонка в программном телефоне

Управление камерой, микрофоном и устройствами вывода звука

Выбор и переключение устройств ввода и вывода

Как и [при захвате видеопотока](#), при совершении звонка из браузера можно выбрать камеру, микрофон и (только в браузере Chrome) устройство вывода звука. Кроме того, устройства можно переключать во время звонка.

Callee SIP username		Call
Camera	ManyCam Virtual Webcam ▼	Next
Mic	Microphone (ManyCam Virtual Microphone) ▼	Next
Speaker	Speakers (Realtek High Definition Audio) ▼	

1.Выбор камеры, микрофона, устройства вывода звукаcode:

```
Flashphoner.getMediaDevices(null, true, MEDIA_DEVICE_KIND.ALL).then(function (list) {
    for (var type in list) {
        if (list.hasOwnProperty(type)) {
            list[type].forEach(function(device) {
                if (device.type == "mic") {
                    ...
                } else if (device.type == "speaker") {
                    ...
                } else if (device.type == "camera") {
                    ...
                }
            });
        }
    }
    ...
}).catch(function (error) {
    $("#notifyFlash").text("Failed to get media devices "+error);
});
```

2. Переключение устройства вывода звука во время звонкаcode:

```
$( "#speakerList" ).change(function() {
    if (currentCall) {
        currentCall.setAudioOutputId($(this).val());
    }
});
```

3. Переключение микрофона во время звонкаcode:

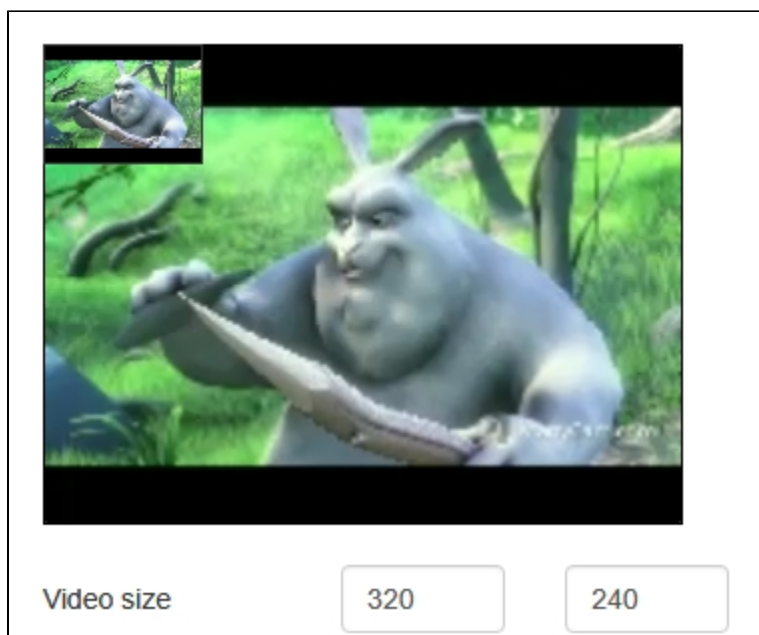
```
$("#switchMicBtn").click(function() {
    if (currentCall) {
        currentCall.switchMic().then(function(id) {
            $('#micList option:selected').prop('selected', false);
            $('#micList option[value="'+ id +'"]').prop('selected', true);
        }).catch(function(e) {
            console.log("Error " + e);
        });
    }
}).prop('disabled', true);
```

4. Переключение камеры во время звонкаcode:

```
$("#switchCamBtn").click(function() {
    if (currentCall) {
        currentCall.switchCam().then(function(id) {
            $('#cameraList option:selected').prop('selected', false);
            $('#cameraList option[value="'+ id +'"]').prop('selected', true);
        }).catch(function(e) {
            console.log("Error " + e);
        });
    }
}).prop('disabled', true);
```

Установка размера видео

При создании звонка, может быть указан размер исходящего видео



code:

```
function getConstraints() {
  var constraints = {
    ...
    video: {
      deviceId: {exact: $('#cameraList').find(":selected").val()},
      width: parseInt($('#sendWidth').val()),
      height: parseInt($('#sendHeight').val())
    }
  };
  if (Browser.isSafariWebRTC() && Browser.isiOS() && Flashphoner.getMediaProviders()[0] === "WebRTC") {
    constraints.video.width = {min: parseInt($('#sendWidth').val()), max: 640};
    constraints.video.height = {min: parseInt($('#sendHeight').val()), max: 480};
  }
  return constraints;
}
```

Совершение звонка без микрофона и камеры

В некоторых случаях, когда звонок не предполагает двухсторонней коммуникации, например, при звонке на голосовое меню, можно позвонить, не используя микрофон и камеру.

Для этого необходимо отключить таймер активности RTP настройкой в файле [flashphoner.properties](#)

```
rtp_activity_detecting=false
```

и отключить аудио и видео в настройке граничных параметров исходящего звонка в браузерах Chrome, Safari, MS Edge:

```

var constraints = {
  audio: false,
  video: false
};

var outCall = session.createCall({
  callee: $("#callee").val(),
  visibleName: $("#sipLogin").val(),
  constraints: constraints,
  ...
})

```

В браузере Firefox необходимо создать пустой аудиопоток:

```

var constraints = {
  audio: false,
  video: false
};
if(Browser.isFirefox()) {
  var audioContext = new AudioContext();
  var emptyAudioStream = audioContext.createMediaStreamDestination().stream;
  constraints.customStream = emptyAudioStream;
}
var outCall = session.createCall({
  callee: $("#callee").val(),
  visibleName: $("#sipLogin").val(),
  constraints: constraints,
  ...
})

```

Отображение WebRTC-статистики

Во время SIP-звонка клиентское приложение получает WebRTC-статистику в соответствии со [стандартом](#). Эта статистика может быть отображена в браузере, например:



Video size

Mute Audio ☐ off

Mute Video ☐ off

Statistics Video

Bytes sent	41485
Packets sent	535
Frames encoded	517

Statistics Audio

Bytes sent	154972
Packets sent	901

Отметим, что в браузере Safari отображается только статистика аудио.

1. Отображение статистики во время звонка

call.getStats()[code](#):

```
currentCall.getStats(function (stats) {
  if (stats && stats.outboundStream) {
    if (stats.outboundStream.videoStats) {
      $('#videoStatBytesSent').text(stats.outboundStream.videoStats.bytesSent);
      $('#videoStatPacketsSent').text(stats.outboundStream.videoStats.packetsSent);
      $('#videoStatFramesEncoded').text(stats.outboundStream.videoStats.framesEncoded);
    } else {
      ...
    }

    if (stats.outboundStream.audioStats) {
      $('#audioStatBytesSent').text(stats.outboundStream.audioStats.bytesSent);
      $('#audioStatPacketsSent').text(stats.outboundStream.audioStats.packetsSent);
    } else {
      ...
    }
  }
});
```

Настройка используемых кодеков

WCS указывает поддерживаемые кодеки в INVITE SDP согласно следующим параметрам в файле [flashphoner.properties](#).

1. В INVITE SDP включаются кодеки, указанные параметром `codecs`, по умолчанию

```
codecs=opus, alaw, ulaw, g729, speex16, g722, mpeg4-generic, telephone-event, h264, vp8, flv, mpv
```

2. Из INVITE SDP исключаются кодеки, указанные параметром `codecs_exclude_sip`, по умолчанию

```
codecs_exclude_sip=mpeg4-generic, flv, mpv
```

3. Из INVITE SDP исключаются кодеки, указанные браузером, если установлен параметр

```
allow_outside_codecs=false
```

4. Из INVITE SDP исключаются кодеки, указанные параметром `stripCodecs` в клиентском приложении, например:

```
var outCall = session.createCall({
  callee: $("#callee").val(),
  ...
  stripCodecs: "SILK, G722"
  ...
});

outCall.call();
```

Передача дополнительных параметров в SDP в запросе INVITE и ответе 200 OK

При создании звонка при помощи JavaScript API могут быть определены дополнительные параметры для управления пропускной способностью канала через SDP для исходящих (в запросе INVITE)

```
var sdpAttributes = ["b=AS:3000", "b=TIAS:2500000", "b=RS:1000", "b=RR:3000"];
var outCall = session.createCall({
    sipSDP: sdpAttributes,
    ...
});
```

и входящих звонков (в ответе 200 OK)

```
var sdpAttributes = ["b=AS:3000", "b=TIAS:2500000", "b=RS:1000", "b=RR:3000"];
inCall.answer({
    sipSDP: sdpAttributes,
    ...
});
```

Эти параметры проставляются в SDP после connection information ("c=IN IP4 <WCS IP>") и до time description ("t=0 0"):

```
v=0
o=Flashphoner 0 1541068898263 IN IP4 192.168.1.5
s=Flashphoner/1.0
c=IN IP4 192.168.1.5
b=AS:3000
b=TIAS:2500000
b=RS:1000
b=RR:3000
t=0 0
m=audio
```

Звонки с использованием SIP TLS сигналинга

Использование SIP TLS сигналинга включается при помощи настройки

```
sip_use_tls=true
```

При этом сертификат SIP сервера проверяется с использованием системного хранилища сертификатов. Поэтому для использования SIP TLS на SIP сервере должен быть установлен действительный SSL сертификат, выданный известным удостоверяющим центром.

Звонки через сервер с самоподписанным сертификатом

Для того, чтобы совершать звонки через SIP сервер с самоподписанным SSL сертификатом, необходимо этот сертификат добавить в локальное хранилище сертификатов сервера, на который установлен WCS:

1. Получите самоподписанный сертификат с SIP сервера

```
openssl s_client -showcerts -connect 192.168.0.153:5061
```

Здесь

- 192.168.0.153 - IP адрес SIP сервера
- 5061 - порт SIP сигналинга

2. Скопируйте сертификаты из ответа сервера

```
Certificate chain
0 s:/CN=pbx.mycompany.com/O=My Super Company
 i:/CN=Asterisk Private CA/O=My Super Company
-----BEGIN CERTIFICATE-----
... SIP server certificate goes here
-----END CERTIFICATE-----
1 s:/CN=Asterisk Private CA/O=My Super Company
 i:/CN=Asterisk Private CA/O=My Super Company
-----BEGIN CERTIFICATE-----
... SIP server CA certificate goes here
-----END CERTIFICATE-----
```

и добавьте их в файл pbx.crt. Содержимое файла должно быть таким:

```
-----BEGIN CERTIFICATE-----
... SIP server certificate goes here
-----END CERTIFICATE-----
-----BEGIN CERTIFICATE-----
... SIP server CA certificate goes here
-----END CERTIFICATE-----
```

3. Определите каталог установки Java

```
readlink -f $(which java)
```

Например, если ответ был таким

```
/usr/java/jdk1.8.0_181/bin/java
```

то Java установлена в каталоге

```
/usr/java/jdk1.8.0_181/
```

4. Найдите файл хранилища сертификатов Java, например

```
find /usr/java/jdk1.8.0_181/jre/lib/security/cacerts
```

5. Импортируйте сертификаты, полученные на шаге 2, в хранилище сертификатов Java

```
keytool -importcert -keystore /usr/java/jdk1.8.0_181/jre/lib/security/cacerts -storepass changeit -file pbx.crt
-alias "pbx"
```

6. Перезапустите WCS.

Подключение к существующей сессии

В некоторых случаях необходимо подключиться в браузере к уже существующей сессии и принять входящий звонок. Как правило, это актуально на мобильных устройствах, т.к. при уходе браузера в фон websocket сессия автоматически закрывается через некоторое время, и приложению остаются доступными только push уведомления. Чтобы сохранить сессию активной при отключении, необходимо при создании сессии указать опцию keepAlive

```

var connectionOptions = {
    urlServer: url,
    keepAlive: true,
    sipOptions: sipOptions
};
...
Flashphoner.createSession(connectionOptions).on(SESSION_STATUS.ESTABLISHED, function(session, connection){
    ...
});

```

В этом случае сессия останется активной до истечения интервала в миллисекундах (по умолчанию 3600 секунд, или 1 час)

```
client_timeout=3600000
```

Периодичность проверки этого интервала задается настройкой в миллисекундах (по умолчанию 300 секунд, или 5 минут)

```
client_timeout_check_interval=300000
```

Чтобы подключиться к этой сессии заново, необходимо запомнить токен сессии

```

Flashphoner.createSession(connectionOptions).on(SESSION_STATUS.ESTABLISHED, function(session, connection){
    authToken = connection.authToken;
    ...
});

```

Затем к этой сессии можно подключиться с помощью этого токена (например, при получении push уведомления о входящем звонке):

```

var connectionOptions = {
    urlServer: url,
    keepAlive: true
};

if (authToken) {
    connectionOptions.authToken = authToken;
} else {
    connectionOptions.sipOptions = sipOptions;
}

Flashphoner.createSession(connectionOptions).on(SESSION_STATUS.ESTABLISHED, function(session, connection){
    ...
});

```

Известные проблемы

1. Невозможно совершить SIP-звонок при некорректно заданных параметрах звонка SIP Login, SIP Authentication name

Симптомы: звонок не совершается, зависает в статусе PENDING

Решение: согласно [стандарту](#), SIP Login и SIP Authentication name должны содержать неэкранированных пробелов, спецсимволов и не должны заключаться в угловые скобки '<>'.
 Например, такое заполнение полей не соответствует стандарту

```

sipLogin='Ralf C12441@host.com'
sipAuthenticationName='Ralf C'
sipPassword='demo'
sipVisibleName='null'

```

а такое соответствует

```
sipLogin='Ralf_C12441'  
sipAuthenticationName='Ralf_C'  
sipPassword='demo'  
sipVisibleName='Ralf C'
```

2. Проблемы со звуком при звонках из браузера Edge.

Симптомы:

- а) исходящий звук периодически то резко приглушается, то идет нормально.
- б) входящий звук слышен, только если говорить в микрофон.

Решение:

Отключить для браузера Edge использование кодеков SILK и G.722 в SIP звонках при помощи опции stripCodecs:

```
var outCall = session.createCall({  
  callee: $("#callee").val(),  
  visibleName: $("#sipLogin").val(),  
  localVideoDisplay: localDisplay,  
  remoteVideoDisplay: remoteDisplay,  
  constraints: constraints,  
  receiveAudio: true,  
  receiveVideo: false,  
  stripCodecs: "SILK,G722"  
  ...  
});  
  
outCall.call();
```

или при помощи настройки

```
codecs_exclude_sip=g722,mpg4-generic,flv,mpv
```

3. Не работает переключение микрофона в браузере Safari.

Симптомы: не переключается микрофон при помощи метода switchMic() WCS WebSDK.

Решение: использовать другой браузер, поскольку Safari всегда использует микрофон sound input, выбранный в настройках звука системы sound menu (для входа необходимо нажать клавишу Option (Alt) и щелкнуть по иконке звука в меню). После выбора другого микрофона в sound menu требуется перезагрузка Mac.

Если не работает микрофон Logitech USB camera (когда выбран в sound input), может помочь изменение format / sample rate в Audio MIDI Setup и перезагрузка.

4. Не устанавливается исходящий видеозвонок из браузера, если размер INVITE SDP превышает размер MTU

Симптомы: при попытке установить исходящий видеозвонок SIP-сторона возвращает 408 Request timeout, аудиозвонки при этом устанавливаются успешно

Решение: уменьшить количество кодеков в INVITE SDP таким образом, чтобы SDP укладывалось в размер пакета, определенный MTU (как правило, 1500 байт), при помощи настроек

```
codecs_exclude_sip=mpg4-generic,flv,mpv,opus,ulaw,h264,g722,g729  
allow_outside_codecs=false
```

Следует оставить только те кодеки, которые поддерживаются обеими сторонами звонка, в данном случае это VP8 и PCMA (alaw).

5. При аудио+видео звонке, если вызываемый абонент отвечает только с аудио, нет звука в браузере

Симптомы: при исходящем аудио+видео звонке (допустим, из примера Phone Video), если вызываемый абонент принимает звонок только с аудио (например, звонок на IVR), в браузере на вызывающей стороне нет звука

Решение: обновить WCS до сборки [5.2.1672](#) и включить генератор видео кадров

```
generate_av_for_ua=all
```

6. При аудио+видео звонке на IVR сообщение слышно не с самого начала

Симптомы: при исходящем аудио+видео звонке (допустим, из примера Phone Video), звук сообщения IVR появляется не сразу

Решение: обновить WCS до сборки [5.2.1755](#) и уменьшить интервал ожидания перед запуском генератора видео кадров

```
generate_av_for_ua=all  
rtp_generator_start_timeout=100
```

7. При звонках между браузерами включается излишний транскодинг в VP8

Симптомы: со стороны SIP сервера приходит видео в кодеке H264, но в браузере играет VP8

Решение:

a) добавить на стороне сервера настройку

```
profiles=42e01f,640028
```

b) если не помогает, добавить настройку

```
proxy_use_h264_packetization_mode_1_only=false
```