

Список методов и используемых параметров

Таблица содержит полный список REST методов и используемых параметров.

Серым цветом выделены параметры, для которых описание дано выше или ниже по таблице.

В зависимости от направления вызова и его назначения, могут использоваться разные подмножества параметров для одного и того же вызова. Например в случае вызова `ConnectionStatusEvent` будут передаваться параметры `sipLogin`, `sipPassword`, и т.д. В случае ошибки, это же событие `ConnectionStatusEvent` будет иметь всего два параметра: `status` и `info` при передаче на клиента и `status`, `info`, `nodeId`, `sessionId`, `appKey` при передаче на бэкэнд сервер.

connect	Установить соединение с WCS сервером
urlServer	Этот параметр будет использован WCS JavaScript API для соединения с сервером
appKey	По этому параметру WCS получит REST URL для данного приложения. Для просмотра и добавления приложений воспользуйтесь Интерфейсом Командной Строки (CLI).
sipRegisterRequired	Если этот параметр <code>true</code> , будет произведена регистрация на SIP сервере путем вызова <code>SIP REGISTER</code> . Если параметр <code>false</code> , регистрация на SIP-сервере произведена не будет. В этом случае web-страница не сможет принимать входящие звонки с SIP, но по прежнему сможет осуществлять исходящие вызовы, если SIP сервер позволяет исходящие звонки без SIP-регистрации
sipLogin	SIP имя пользователя
sipAuthenticationName	SIP имя пользователя, которое используется для SIP аутентификации. Может отличаться от <code>sipLogin</code>
sipPassword	SIP пароль. Используется для SIP аутентификации
sipVisibleName	SIP имя пользователя, которое будет отображаться абонентам, получающ

	им от этого пользователя входящий звонок
sipDomain	SIP домен: FQDN или IP-адрес
sipOutboundProxy	IP прокси сервер: FQDN или IP адрес. Может отличаться от <code>sipDomain</code>
sipPort	SIP порт, который используется на SIP сервере для обработки SIP трафика.
sipContactParams	Строка пользовательских произвольных параметров, которая будет добавлена в <code>SIP Contact</code> заголовок <code>REGISTER</code> запроса
status	
mediaProviders	Массив с доступными типами медиа на WCS JavaScript API: <code>['WebRTC', 'Flash']</code>
restClientConfig	JSON - объект, описывающий конфигурацию управления взаимодействием с бэкенд сервером. Если объект не был передан, используются значения по умолчанию. См <code>[restClientConfig](restClientConfig_object_description.ru.md)</code>
width	Максимальная ширина видео, в пикселях
height	Максимальная высота видео, в пикселях
custom	Объект с произвольным содержимым для идентификации клиента на бэкенд сервере
ConnectionStatusEvent	Изменение статуса соединения
sipRegisterRequired	
sipLogin	
sipPassword	
sipVisibleName	
sipDomain	
sipOutboundProxy	
sipPort	
sipContactParams	

status	Статус соединения с WCS сервером: <code>PENDING</code> , <code>ESTABLISHED</code> , <code>FAILED</code> , <code>DISCONNECTED</code>
info	В это поле может быть добавлена дополнительная информация. Например если <code>status: "FAILED"</code> , в <code>info</code> будет передано описание причины
authToken	Ключ использующийся WCS JavaScript API для подключения к активной SIP сессии
mediaProviders	
nodeId	
sessionId	
appKey	
custom	Объект с произвольным содержимым для идентификации клиента на бэкенд сервере, переданный в <code>/connect</code>
RegistrationStatusEvent	Изменение статуса SIP регистрации
status	Статусы регистрации: <code>REGISTERED</code> , <code>UNREGISTERED</code> , <code>FAILED</code>
info	
sipMessageRaw	Исходное SIP-сообщение с заголовками. SIP ответ на запрос <code>REGISTER</code>
nodeId	
sessionId	
appKey	
call	Исходящий звонок
callId	Уникальный идентификатор звонка
callee	Вызываемый абонент в формате SIP URI или телефонный номер
caller	Вызывающий абонент в формате SIP URI
visibleName	Имя, которое будет отображаться у вызываемого абонента
hasVideo	Если <code>true</code> , то это видео звонок
inviteParameters	Параметры, <u>которые будут</u> добавлен

	ы в запрос <code>SIP INVITE</code>
isMsrp	Если <code>true</code> , то это не голосовой вызов, а установление MSRP-соединения для передачи данных
status	
incoming	Если <code>true</code> , то это звонок, поступивший с SIP стороны
mediaProvider	Используемая технология передачи медиа на WCS JavaScript API, возможные значения: <code>WebRTC</code> , <code>Flash</code>
sdp	SDP, сформированное WCS JavaScript API, присутствует при <code>mediaProvider: "WebRTC"</code>
OnCallEvent	Входящий звонок
callId	
callee	
caller	
visibleName	
hasVideo	
inviteParameters	
sipMessageRaw	<code>SIP INVITE</code> сообщение, на основе которого создается событие входящего звонка
incoming	
status	
mediaProvider	
sdp	
nodeId	
sessionId	
appKey	
CallStatusEvent	Изменение статуса звонка
callId	
incoming	Если <code>true</code> , то звонок входящий
status	Статус _____

	звонка: TRYING, RING, SESSION_PROGRESS, BUSY, ESTABLISHED, HOLD, FINISH, FAILED
info	
sipMessageRaw	Исходное сообщение, соответствующее передаваемому событию. Например, в случае TRYING, это будет ответ SIP 100 TRYING. В случае ESTABLISHED будет ответ SIP 200 OK, в случае HOLD будет ответ SIP 200 OK на re-INVITE, и т.д.
sipStatus	Статус, полученный от SIP стороны
caller	
callee	
hasVideo	
visibleName	
mediaProvider	
nodeId	
sessionId	
appKey	
answer	Ответить на входящий звонок
callId	
incoming	
sipStatus	
caller	
callee	
hasVideo	
visibleName	
mediaProvider	
sdp	
status	
nodeId	
sessionId	

appKey	
hangup	Завершить звонок
callId	
hasVideo	
nodeId	
sessionId	
appKey	
hold	Поставить звонок на удержание
callId	
hasVideo	
nodeId	
sessionId	
appKey	
unhold	Снять звонок с удержания
callId	
hasVideo	
nodeId	
sessionId	
appKey	
transfer	Перевести звонок
callId	
target	Номер или SIP URI абонента, которому будет переведен звонок
nodeId	
sessionId	
appKey	
TransferStatusEvent	Изменение статуса передачи звонка
callId	
incoming	Если <code>true</code> , то передача была иници

	ирована другой стороной
status	Статус передачи звонка: <code>ACCEPTED</code> , <code>TRYING</code> , <code>COMPLETED</code> , <code>FAILED</code> . Если статус не распознан, в событии будет передан статус, полученный от SIP стороны
info	
sipMessageRaw	
hasVideo	
nodeId	
sessionId	
appKey	
sendDTMF	Отправить DTMF сигнал
callId	
dtmf	Символ в текстовом виде, который передается в DTMF: 0-9, *, #
type	Тип DTMF сигнала: <code>INFO</code> , <code>INFO_RELAY</code> , <code>RFC2833</code>
nodeId	
sessionId	
appKey	
sendMessage	Отправить сообщение
id	Уникальный идентификатор сообщения
from	Номер или SIP URI отправителя сообщения
to	Номер, имя или SIP URI получателя сообщения
body	Текст сообщения
contentType	<code>text/plain</code> - сообщение будет отправлено как <code>SIP MESSAGE</code> с заголовком <code>Content-Type : text/plain</code> и с текстом в теле сообщения <code>message/cpim</code> - сообщение будет отправлено как <code>SIP MESSAGE</code> с заголовком <code>Content-Type:message/cpim</code> и сообщением в формате <code>text/plain</code> в теле CPIM-сообщения

	<p><code>multipart/mixed</code> - сообщение будет отправлено как <code>SIP MESSAGE</code> с заголовком <code>Content-Type: multipart/mixed</code> и CPIM сообщениями в теле, каждое из которых в своем теле имеет <code>text/plain</code> сообщение</p>
<code>isImdnRequired</code>	<p>Если флаг установлен в <code>true</code>, для типов <code>message/cpim</code> и <code>multipart/mixed</code> в тело CPIM сообщения будет добавлена информация, запрашивающая уведомление о доставке через IMDN</p>
<code>recipients</code>	<p>Список получателей сообщения, разделенный запятой. Через запятую должны быть перечислены SIP URI, телефоны или SIP логины получателей. Поле используется только в том случае, если задан <code>ContentType multipart/mixed</code>. Это поле будет корректно работать только тогда, когда SIP-сервер поддерживает отправку сообщений нескольким абонентам на основе <code>multipart/mixed</code>. WCS отправит SIP серверу <code>multipart/mixed</code> сообщение с несколькими адресатами. Если ваш SIP-сервер не поддерживает такую отправку, оставьте это поле пустым и попытайтесь отправить несколько отдельных сообщений</p>
<code>nodeId</code>	
<code>sessionId</code>	
<code>appKey</code>	
OnMessageEvent	Входящее сообщение
<code>id</code>	
<code>from</code>	
<code>to</code>	
<code>body</code>	
<code>contentType</code>	
<code>isImdnRequired</code>	<p>Если входящее сообщение имеет такой флаг, то будет высылаться уведомление о доставке IMDN</p>
<code>sipMessageRaw</code>	<p><code>SIP MESSAGE</code> сообщение, соответствующее событию входящего сообщения <code>OnMessageEvent</code></p>
<code>status</code>	

nodeId	
sessionId	
appKey	
MessageStatusEvent	Изменение статуса сообщения
id	
from	
to	
contentType	
isImdnRequired	
body	
status	Статусы сообщений: RECEIVED, ACCEPTED, FAILED, IMDN_DELIVERED, IMDN_FAILED, IMDN_NOTIFICATION_SENT
info	
sipMessageRaw	SIP сообщение, соответствующее статусу: - RECEIVED - Получено входящее сообщение в SIP INVITE - ACCEPTED - SIP 200 OK или 202 ACCEPTED ответ на запрос SIP MESSAGE - FAILED - SIP 4xx ответ от SIP-сервера - DELIVERED - Получено входящее уведомление о доставке SIP MESSAGE со статусом Delivered - DELIVERY_FAILED - Получено уведомление о доставке со статусом Delivery Failed
nodeId	
sessionId	
appKey	
sendIMDN	Отправить уведомление о доставке
messageId	
nodeId	
sessionId	
appKey	
subscribe	SIP subscribe - подписка на уведомления SIP сервера: RFC3265

event	Тип события: reg
expires	Интервал в секундах. В течение этого интервала WCS-сервер будет делать re-SUBSCRIBE.
terminate	
nodeId	
sessionId	
appKey	
SubscriptionStatusEvent	Изменение статуса SIP-подписки
event	
expires	
terminate	Если true то подписка должна быть деактивирована
requestBody	Поле, содержащее пришедший XML с SIP стороны
status	Статус подписки: <code>Active</code> , <code>Terminated</code>
info	
sipMessageRaw	SIP сообщение, изменяющее статус подписки: - <code>Active - SIP 200 OK</code> ответ на запрос <code>SUBSCRIBE</code> - <code>Terminated - SIP 200 OK</code> ответ на запрос <code>SUBSCRIBE</code> с <code>expires:0</code> - <code>Terminated - SIP NOTIFY</code> запрос с статусом <code>terminated</code> внутри тела сообщения <code>NOTIFY</code>
nodeId	
sessionId	
appKey	
sendXcapRequest	Отправка XCAP запроса
url	URL для XCAP запроса
nodeId	
sessionId	
appKey	
XcapStatusEvent	Получение XCAP ответа

url	
xcapResponse	Тело XCAP-ответа
publishStream	Публикация потока на сервер
name	Имя публикуемого потока. Должно быть уникальным. Если поток с таким именем уже опубликован, публикация данного потока будет запрещена.
mediaSessionId	Идентификатор медиа сессии
published	Если true, то поток является опубликованным
hasVideo	Сигнализирует присутствие видео в потоке
status	
sdp	SDP, полученное от клиента
nodeId	
sessionId	
appKey	
record	Если <code>true</code> , то опубликованный поток записывается
custom	Объект с произвольным содержимым для идентификации клиента на бэкенд сервере
unPublishStream	Остановка публикации потока
name	
mediaSessionId	
published	
hasVideo	
status	
sdp	
nodeId	
sessionId	
appKey	
record	

custom	Объект с произвольным содержимым для идентификации клиента на бэкенд сервере, переданный в <code>/publishStream</code>
playStream	Воспроизведение потока
name	Имя воспроизводимого потока
mediaSession	
published	
hasVideo	
status	
sdp	
nodeId	
sessionId	
appKey	
custom	Объект с произвольным содержимым для идентификации клиента на бэкенд сервере
playHLS	Воспроизведение HLS потока с сервера
name	
mediaSessionId	
mediaProvider	
nodeId	
sessionId	
appKey	
token	Токен для аутентификации клиента
playRTSP	Воспроизведение RTSP потока с сервера
name	
mediaSessionId	
mediaProvider	
nodeId	

sessionId	
appKey	
rtspUrl	URL RTSP потока
User-Agent	User agent клиента
stopStream	Остановка воспроизведения потока
name	
mediaSessionId	
published	
hasVideo	
status	
sdp	
nodeId	
sessionId	
appKey	
custom	Объект с произвольным содержимым для идентификации клиента на бэкенд сервере, переданный в <code>/playStream</code>
StreamStatusEvent	Изменение статуса потока
name	
status	Статус потока: <code>PUBLISHING</code> , <code>UNPUBLISHED</code> , <code>PUBLISHING</code> , <code>STOPPED</code>
mediaSessionId	
published	
hasVideo	
sdp	
info	
nodeId	
sessionId	
appKey	

record	
custom	Объект с произвольным содержимым для идентификации клиента на бэкенд сервере, переданный в <code>/publishStream</code> или <code>/playStream</code>
StreamKeepAliveEvent	Keep-alive запрос для потока
nodeId	
appKey	
sessionId	
mediaSessionId	
name	
published	
hasVideo	
status	Статус потока: <code>PLAYING</code> , <code>PUBLISHING</code>
info	
mediaProvider	Используемая технология передачи медиа на WCS JavaScript API, возможные значения: <code>WebRTC</code> , <code>Flash</code>
record	
sendData	Отправка данных
operationId	Уникальный идентификатор для отправляемых данных
payload	JSON объект с данными
nodeId	
sessionId	
appKey	
OnDataEvent	Получение входящих данных
operationId	
payload	
nodeId	
sessionId	
appKey	

DataStatusEvent	Изменение статуса отправленных данных
operationId	
status	ACCEPTED, FAILED
info	
nodeId	
sessionId	
appKey	
ErrorEvent	Неклассифицированная ошибка
info	Дополнительная информация об ошибке
sendBugReport	Отправка отчета об ошибке для сохранения на сервере
text	Отправка отчета об ошибке для сохранения на сервере
type	Если тип <code>no_media</code> , сервер включит дамп трафика перед созданием этого баг-репорта, чтобы убедиться в том, что трафик идет правильно для этого пользователя. Отправка баг-репорта такого типа может быть полезна при диагностике проблемы, когда звук передается только в одну сторону
nodeId	
sessionId	
appKey	
BugReportStatusEvent	Подтверждение отправки отчета об ошибке с выводом имени сохраненного файла
filename	Имя файла на сервере, под которым был сохранен баг-репорт
nodeId	
sessionId	
appKey	
sendStreamEvent	Событие, сигнализирующее изменение состояния публикуемого потока
info	Дополнительная информация

type	Тип события: <code>audioMuted</code> , <code>videoMuted</code> , <code>audioUnmuted</code> , <code>videoUnmuted</code>
mediaSessionId	Идентификатор публикующей медиа сессии
nodeId	
sessionId	
appKey	
StreamEvent	Событие, сигнализирующее изменение состояния потока для подписчиков
info	Дополнительная информация
type	Тип события: <code>audioMuted</code> , <code>videoMuted</code> , <code>audioUnmuted</code> , <code>videoUnmuted</code>
mediaSessionId	Идентификатор медиасессии подписчика
nodeId	
sessionId	
appKey	
Context Parameters	Параметры контекста. Используются для всех вызовов от WCS к бэкенд серверу
nodeId	Уникальный идентификатор инстанса WCS сервера
sessionId	Уникальный идентификатор подключения клиента на этом инстансе
appKey	Идентификатор приложения на WCS сервере, с которым пользователь установил соединение