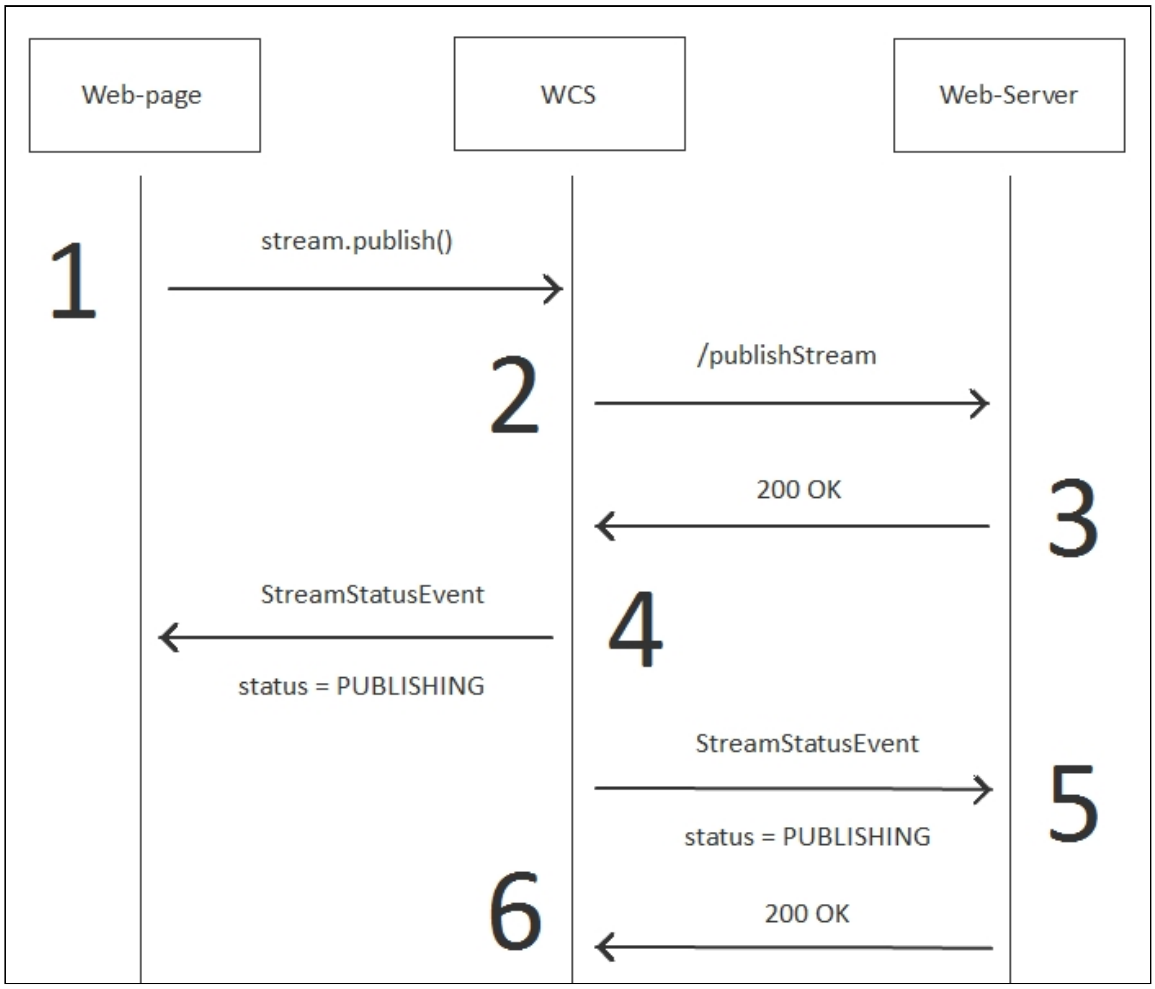
Тип 3 - событие

Описание работы события на примере `StreamStatusEvent`

REST-метод `StreamStatusEvent` относится к событиям. События происходят внутри WCS-сервера и используются для передачи статусов тех или иных операций.

Например, событие `StreamStatusEvent` используется для передачи статусов операций, связанных с видеопотоками, такими как `Stream.play()` и `Stream.publish()`. Действительно, если мы публикуем, воспроизводим видеопоток или останавливаем его, мы должны знать его статус для того, чтобы управлять таким потоком.

Бэкенд сервер не может аутентифицировать (запретить или разрешить) событие и просто принимает его, например для сохранения информации о потоке в базе данных.



Из этой диаграммы видно, что событие `StreamStatusEvent` идет по двумbнаправлениям:

1. На клиента - шаг 4
2. На web-сервер - шаг 5

Пример:

REST hook

```
POST /rest/my_api/StreamStatusEvent HTTP/1.1
Accept: application/json
Content-Type: application/json;charset=UTF-8
User-Agent: Java/1.8.0_111
Host: 192.168.1.101
Connection: keep-alive
Content-Length: 3614

{
    "nodeId":"Hw47CFMBEchVOpBMDr29IxjudnJ1sjOY@192.168.1.101",
    "appKey":"defaultApp",
    "sessionId":"/192.168.1.102:4388/192.168.1.101:8443",
    "mediaSessionId":"56141d10-fddc-11e6-ac3a-4d67d5b3360d",
    "name":"b4e7",
    "published":true,
    "hasVideo":true,
    "hasAudio":true,
    "status":"PUBLISHING",
    "sdp":".....",
    "record":false,
    "width":0,
    "height":0,
    "bitrate":0,
    "quality":0,
    "mediaProvider":"WebRTC"
}
```

Backend response

```
HTTP/1.1 200 OK
Date: Tue, 28 Feb 2017 17:35:44 GMT
Server: Apache/2.2.15 (CentOS)
X-Powered-By: PHP/5.3.3
Content-Length: 3656
Connection: close
Content-Type: application/json

{
   "nodeId":"Hw47CFMBEchVOpBMDr29IxjudnJ1sjOY@192.168.1.101",
    "appKey":"defaultApp",
    "sessionId":"/192.168.1.102:4388/192.168.1.101:8443",
    "mediaSessionId":"56141d10-fddc-11e6-ac3a-4d67d5b3360d",
    "name":"b4e7",
    "published":true,
    "hasVideo":true,
```

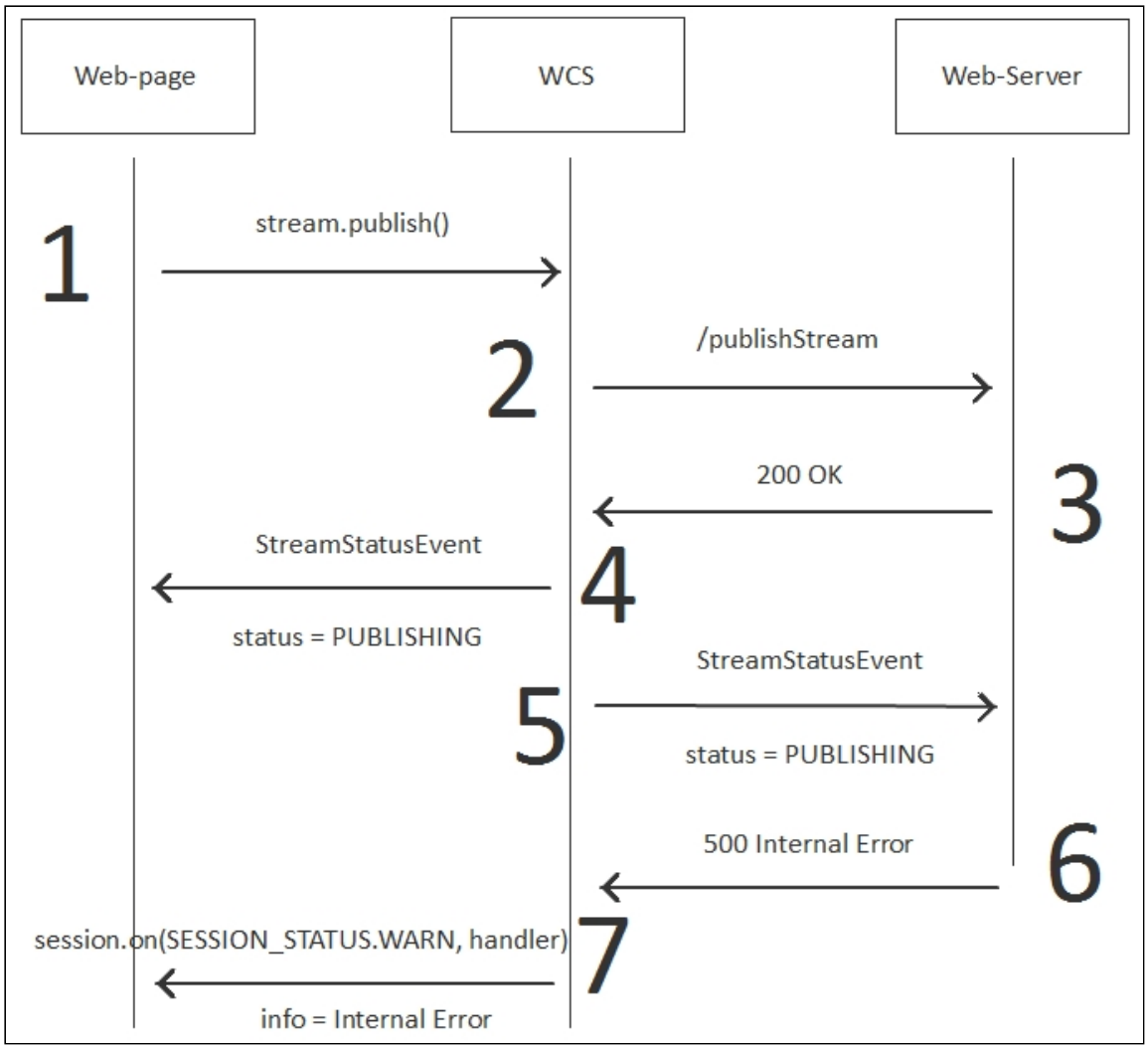
```
    "hasAudio":true,
    "status":"PUBLISHING",
    "sdp":".....",
    "record":false,
    "width":0,
    "height":0,
    "bitrate":0,
    "quality":0,
    "mediaProvider":"WebRTC"
}
```

Обработка ошибок

По-умолчанию, WCS не проверяет статус ответа на вызов REST-метода `/StreamStatusEvent`. Т.е. если web-сервер вернет HTTP статус ошибки 403 или 500 или любой другой, то WCS это роигнорирует.

Чтобы WCS прореагировал на такую ошибку, нужно выставить `restOnError: FAIL` в настройках метода `StreamStatusEvent` в объекте restClientConfig при создании подключения.

В этом случае на клиента будет отправлено отдельное событие `ErrorEvent`, и клиент будет оповещен об ошибке.



На шаге 6 web-сервер возвращает HTTP статус `500 Internal Error` в ответ на вызов метода `/StreamStatusEvent`. WCS сервер на шаге 7 уведомляет клиента о возникшей ошибке.

Пример:

REST hook

```
POST /rest/my_api/StreamStatusEvent HTTP/1.1
Accept: application/json
Content-Type: application/json;charset=UTF-8
User-Agent: Java/1.8.0_111
Host: 192.168.1.101
Connection: keep-alive
Content-Length: 3624

{
    "nodeId":"Hw47CFMBEchVOpBMDr29IxjudnJ1sjOY@192.168.1.101",
    "appKey":"defaultApp",
    "sessionId":"/192.168.1.102:25301/192.168.1.101:8443",
    "mediaSessionId":"e9c002d0-fde2-11e6-a2bf-c99492323844",
    "name":"dc6a",
```

```
    "published":true,
    "hasVideo":true,
    "hasAudio":true,
    "status":"PUBLISHING",
    "sdp":".....",
    "record":false,
    "width":0,
    "height":0,
    "bitrate":0,
    "quality":0,
    "mediaProvider":"WebRTC"
}
```

Backend response

```
HTTP/1.1 500 Internal Server Error
Date: Tue, 28 Feb 2017 18:22:49 GMT
Server: Apache/2.2.15 (CentOS)
X-Powered-By: PHP/5.3.3
Content-Length: 0
Connection: close
Content-Type: text/html; charset=UTF-8
```