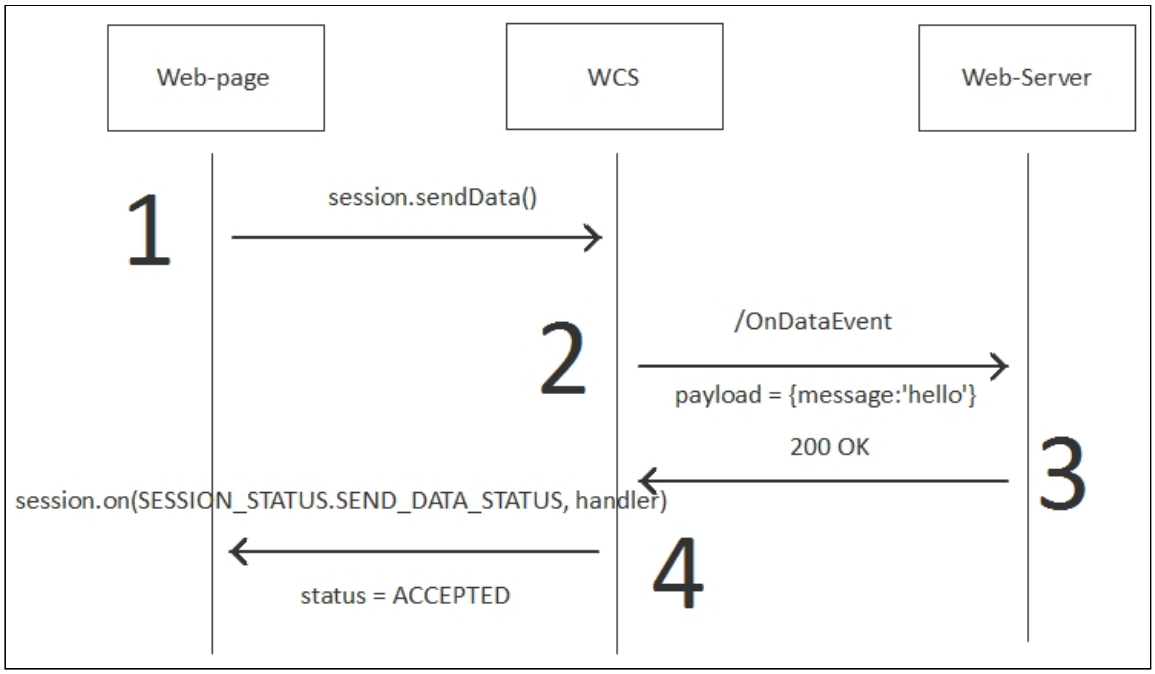
Обмен данными - OnDataEvent

С помощью клиентского вызова `Session.sendData()` и REST метода `OnDataEvent` можно организовать произвольный обмен данными между любыми клиентами, подключенными к WCS-серверу.Зная `sessionId` подключенного клиента, можно направить ему произвольный объект с данными при помощи REST API запроса `/rest-api/data/send`.

Отправка данных на сервер



1. Клиент отправляет произвольные данные с помощью функции `Session.sendData()`
2. WCS отправляет `/OnDataEvent` REST запрос на бэкенд сервер
3. Бэкенд сервер возвращает `200 OK`
4. WCS отправляет событие `SESSION_STATUS.SEND_DATA_STATUS` со значением `ACCEPTED` клиенту-отправителю

Пример:

REST hook

```
POST /rest/my_api/OnDataEvent 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: 218

{
    "nodeId":"Hw47CFMBEchVOpBMDr29IxjudnJ1sjOY@192.168.1.101",
    "appKey":"defaultApp",
    "sessionId":"/192.168.1.102:55839/192.168.1.101:8443",
    "operationId":"d1999750-fde9-11e6-9f1b-913210792931",
    "payload":{
        "message":"hello"
    }
}
```

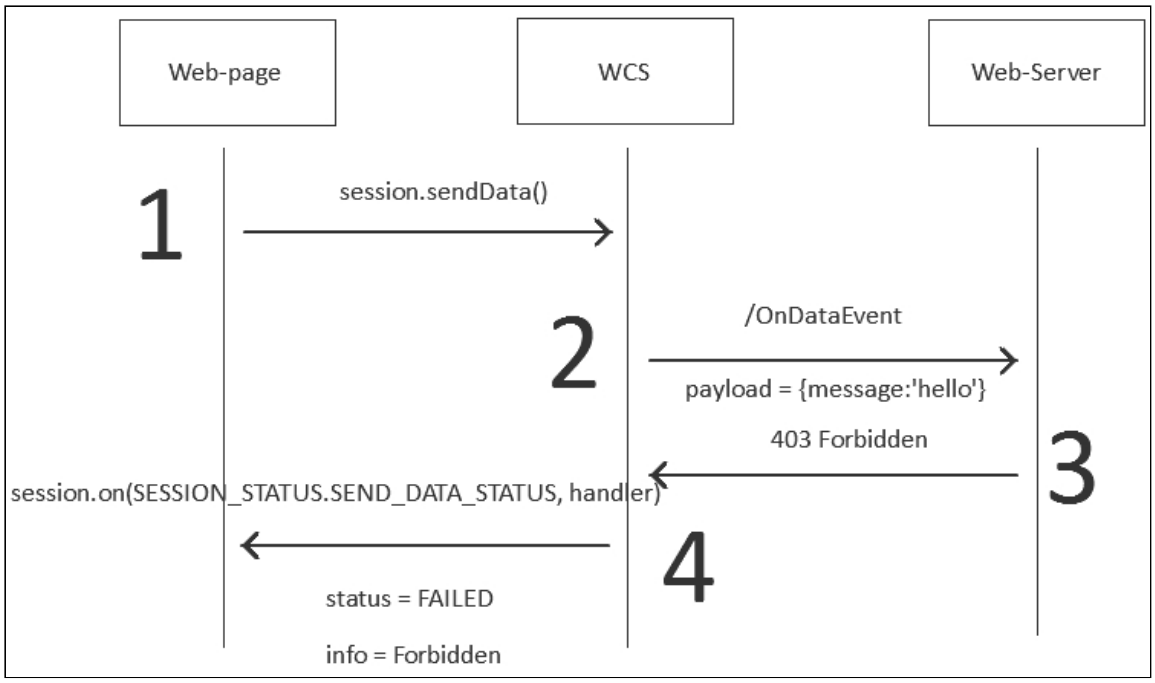
Backend response

```
HTTP/1.1 200 OK
Date: Tue, 28 Feb 2017 19:12:14 GMT
Server: Apache/2.2.15 (CentOS)
X-Powered-By: PHP/5.3.3
Content-Length: 220
Connection: close
Content-Type: application/json

{
    "nodeId":"Hw47CFMBEchVOpBMDr29IxjudnJ1sjOY@192.168.1.101",
    "appKey":"defaultApp",
    "sessionId":"/192.168.1.102:55839/192.168.1.101:8443",
    "operationId":"d1999750-fde9-11e6-9f1b-913210792931",
    "payload":{
        "message":"hello"
    }
}
```

Обработка ошибок

Для включения такого поведения, нужно передать `restOnError: FAIL` в restClientConfig для метода `OnDataEvent` при установке соединения методом `connect`.



1. Клиент отправляет произвольные данные с помощью функции `Session.sendData()`
2. WCS отправляет `/OnDataEvent` REST запрос на бэкенд сервер
3. Бэкенд сервер возвращает `403 Forbidden`
4. WCS отправляет событие `SESSION_STATUS.SEND_DATA_STATUS` со значением `FAILED` клиенту-отправителю

Пример:

REST hook

```
POST /rest/my_api/OnDataEvent 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: 218

{
    "nodeId":"Hw47CFMBEchVOpBMDr29IxjudnJ1sjOY@192.168.1.101",
    "appKey":"defaultApp",
    "sessionId":"/192.168.1.102:25789/192.168.1.101:8443",
    "operationId":"d0149310-fdeb-11e6-9b58-9509528e5d66",
    "payload":{
        "message":"hello"
    }
}
```

Backend response

```
HTTP/1.1 403 Forbidden
Date: Tue, 28 Feb 2017 19:26:30 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
```

Отправка данных подключенному клиенту методом /data/send

Можно отправить подключившемуся клиенту прямое сообщение REST API вызовом `http://host:8081/rest-api/data/send`

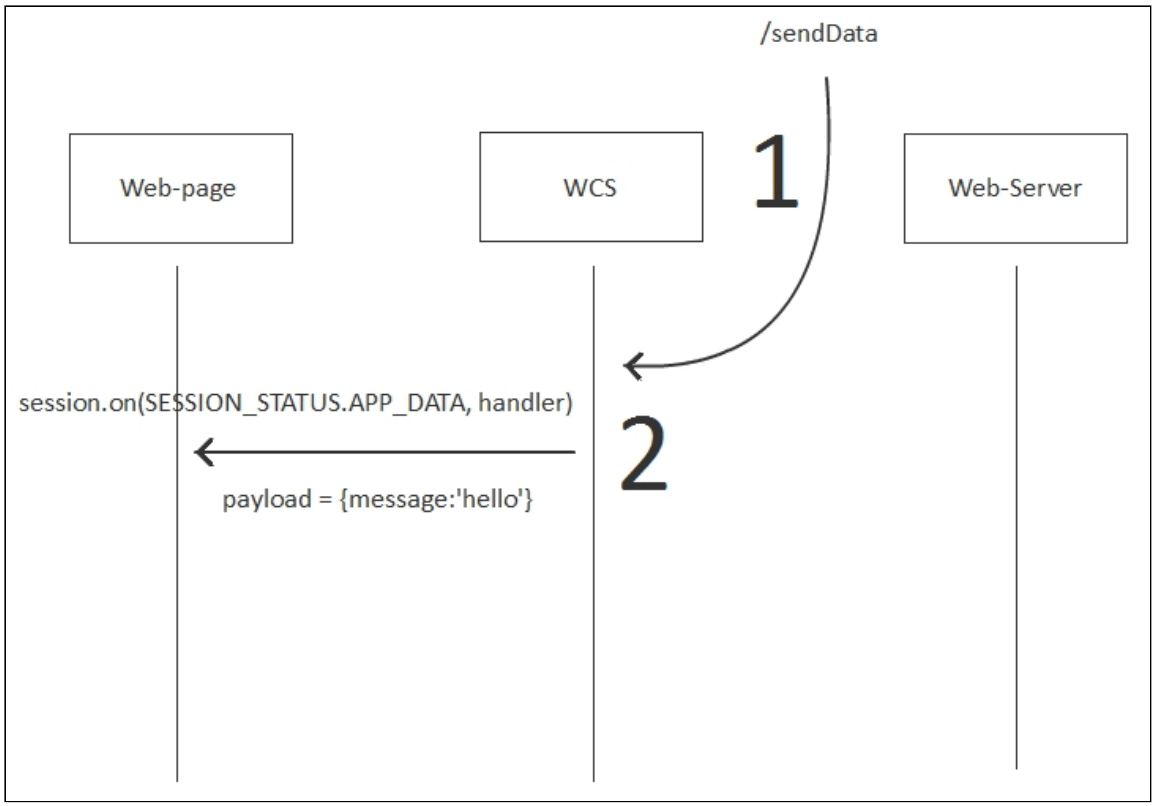
Для этого нужно передать следующий JSON-объект

```
{
   "nodeId":"",
   "sessionId":"/192.168.1.102:25789/192.168.1.101:8443",
   "operationId":"",
   "payload":{
      "test":"test"
   }
}
```

Здесь `sessionId` - идентификатор сессии подключенного клиента, может быть получен на бэкенд-сервере при обработке REST метода `/connect`.

В этом случае, подключенный клиент получит объект `custom`, в который вы можете поместить любые данные, например `{"test":"test"}`, как в примере выше.

Клиент получит событие `SESSION_STATUS.APP_DATA`


/sendData
Web-page
WCS
1
Web-Server
session.on(SESSION_STATUS.APP_DATA, handler)
2
payload = {message:'hello'}