

# Обмен данными - OnDataEvent

- Отправка данных на серверметодом `sendData / OnDataEvent`
  - Обработка ошибок
- Отправка данных подключенному клиентуметодом `/data/send`

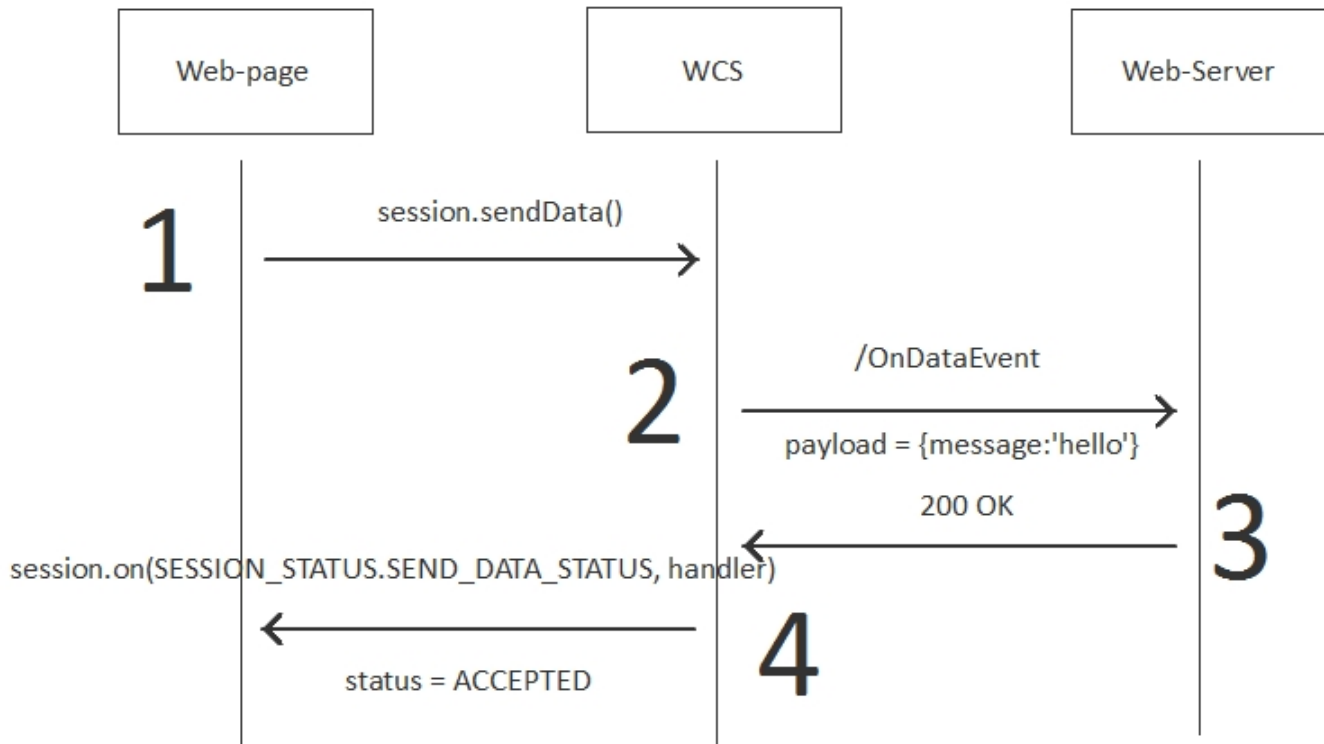
С помощью `sendData` можно организовать произвольный обмен данными между любыми клиентами, подключенными к WCS-серверу.

Зная `sessionId` подключенного клиента, можно направить ему произвольный объект с данными.

## Отправка данных на серверметодом `sendData / OnDataEvent`

Клиент может отправить произвольный объект с помощью вызова прямого метода `sendData()` - шаг 1.

На web-сервере будет вызван REST-метод `/OnDataEvent` - шаг 2. В результате, клиент получит статус `SESSION_STATUS.SEND_DATA_STATUS ACCEPTED` - шаг 4.



Пример:

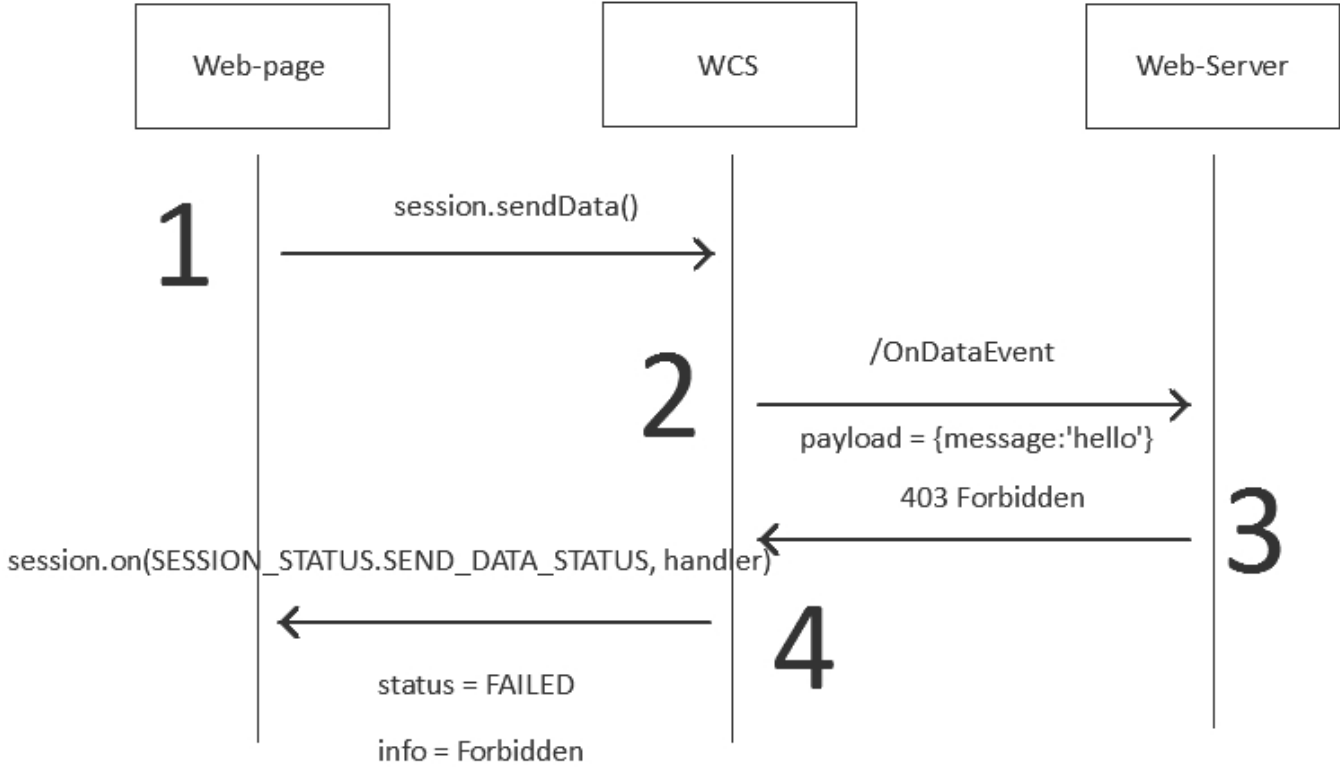
2	3
---	---

<pre> POST /rest/my_api/OnDataEvent HTTP/1.1 Accept: application/json, 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"   } } </pre>	<pre> 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"   } } </pre>
---	--

### Обработка ошибок

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

Клиент может отправить произвольный объект с помощью вызова прямого метода `sendData()` - шаг 1. На web-сервере будет вызван REST-метод `/OnDataEvent` - шаг 2. В результате, клиент получит статус `SESSION_STATUS.SEND_DATA_STATUS FAILED` - шаг 4.



Пример:

2	3
---	---

```
POST /rest/my_api/OnDataEvent HTTP/1.1
Accept: application/json, 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"
  }
}
```

```
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 кастомный объект, в который вы можете поместить любые данные, например `{"test": "test"}`, как в примере выше.

Клиент получит событие `SESSION_STATUS.APP_DATA`

