# [Содержание](#Содержание)

# [Создание oData сервиса. Часть 1.](#Создание_oData_сервиса_Часть_1)

[*Step 1* – *Создание сервиса*](#S_1_1_Создание_сервиса)

[*Step 2* – Активация сервиса](#S1_2_Активация_сервиса)

[*Step 3* – Активация SAP *Netweaver Gateway*](#S1_3_Активация_SAP_Netweaver_Gateway)

[*Step 4* – *Тестирование сервиса*](#S1_4_Тестирование_сервиса)

# REST Service Integration in SAP BW – 28.04.2021

# <https://www.dahlbeer.com/post/rest-service-integration-in-sap-bw>

# How to Create an OData Service and Exposing CDS Views as an OData Service

# <https://blog.sap-press.com/how-to-create-an-odata-service-and-exposing-cds-views-as-an-odata-service>

# OData API in SAP Cloud Integration for Beginners

# <https://blogs.sap.com/2019/08/12/odata-service-in-cpi-for-beginners/>

# [HTTPS POST-запрос от Java-приложения к OData-Service в системе S/4HANA с действительным токеном CSR](#POST_запрос_от_Java_к_OData_Service)

# Bw data source on web service

# Configure a webservice datasource in #BW – 17/01/2014

# <https://blogs.sap.com/2014/01/17/configure-a-webservice-datasource-in-bw/>

# REST Service Integration in SAP BW

# <https://www.dahlbeer.com/post/rest-service-integration-in-sap-bw>

# Передача данных, используя Web Services в BW – 2/02/2018

[Creating InfoPackages for Real-Time Data Acquisition](https://help.sap.com/saphelp_nw74/helpdata/en/4a/2c6be7c16d47dbe10000000a42189c/content.htm)

[Data Transfer Using Web Services](https://help.sap.com/saphelp_nw74/helpdata/en/4a/1dfbbe5c171b40e10000000a42189c/content.htm)

[Transferring Data from SLT Using Web Service](https://help.sap.com/saphelp_nw74/helpdata/en/c5/80b8ba211c4d5993adf04eb2070b4d/content.htm)

[Transferring Transaction Data Using Web Services](https://help.sap.com/saphelp_nw74/helpdata/en/4a/2c6d68c16d47dbe10000000a42189c/frameset.htm)

[How to configure a Service Provider](https://wiki.scn.sap.com/wiki/display/ABAPConn/How+to+configure+a+Service+Provider)

[Настройка Web сервиса между системами SAP, на англ. раз.](https://wiki.scn.sap.com/wiki/display/ABAP/CREATING+OF+WEBSERVICES+AND+CONSUMING+IN+SAP)

[Пошаговое руководство по созданию Web сервиса, на англ. два.](https://blogs.sap.com/2014/05/20/step-by-step-to-create-consume-and-trace-web-service-in-abap-system/)

##### Передача данных

###### Использование

=ReportFilter([Volume for max data])

=[Объем] Where ([Календарный день] = [max\_data attr])

Web Services используется, если вы хотите, что бы загрузка данных контролировалась из внешней системы. В таком случае данные пушатся/проталкиваются – push.

При использовании Web сервисов для передачи данных в BW, возможно использовать экстракцию данных в режиме реального времени. Альтернативный вариант — использовать стандартный DTP.

1. Если данные нужны часто, то используется сбор данных в режиме реального времени – а) данные сначала записываются в PSA и затем б) с помощью демона или фонового процесса через определённые промежутки времени - записываются в DSO.
2. Если нет необходимости в обновлении данных через определённые короткие интервалы времени, то используется стандартное обновление.

# Передача данных, используя Web Services в BW

<https://helpbw.wordpress.com/2018/02/02/%D0%BF%D0%B5%D1%80%D0%B5%D0%B4%D0%B0%D1%87%D0%B0-%D0%B4%D0%B0%D0%BD%D0%BD%D1%8B%D1%85-%D0%B8%D1%81%D0%BF%D0%BE%D0%BB%D1%8C%D0%B7%D1%83%D1%8F-web-services-%D0%B2-bw/>

Web Service - загрузка данных контролируется из внешней системы - данные пушатся/проталкиваются – push. As of 7.3 this can also be a pull datasource meaning bw can initiate the data retrieve.

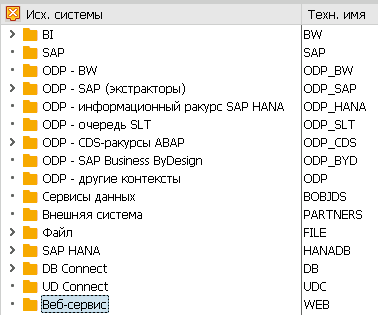
Режим передачи данных в BW

1. В режиме *реального времени* /если данные нужны часто/ - данные записываются в PSA и затем с помощью демона или фонового процесса через определённые промежутки времени - записываются в ADSO.
2. Использование *стандартного DTP*.

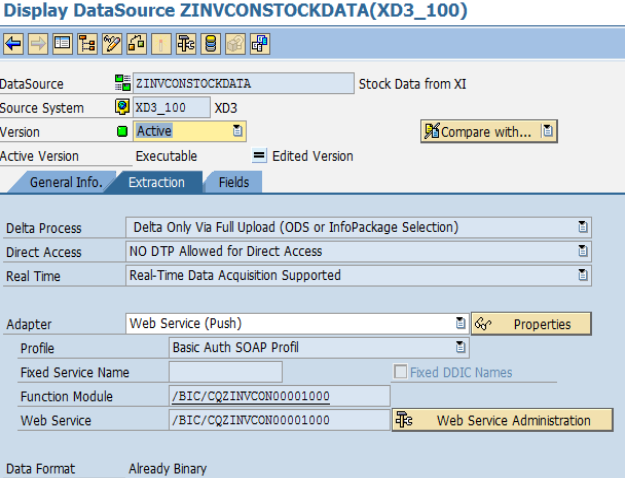
**Configure a webservice datasource in #BW**

<https://blogs.sap.com/2014/01/17/configure-a-webservice-datasource-in-bw/>

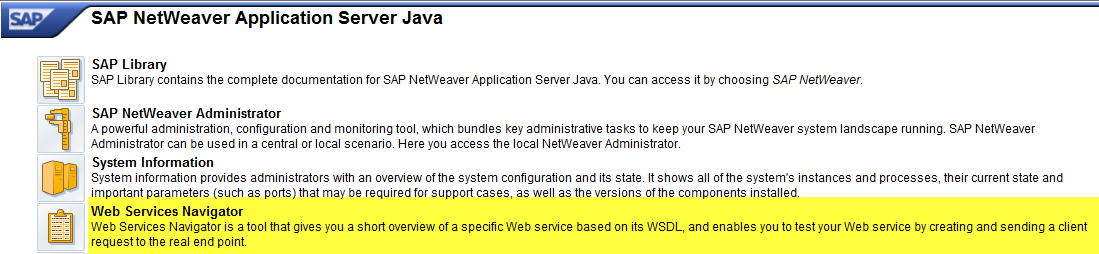
1. To receive *soap* documents the SOAP service needs to be active in ***SICF*** only then data can be sent to bw through webservices.
2. Before the datasource can be configured a ***sources system*** of the type *webservice* needs to be created.



1. Maintain the necessary fields in the datasource and create an ***infopackage***.



1. On the web services navigator test the WSDL service interface which can be access by going through the the Netweaver Java Application Server.



1. To retrieve the URL with the XML definition go through ***WSADMIN*** /the transaction is obsolete from SAP NetWeaver AS 7.00, SP14/ use ***SOAMANAGER***.
2. Make sure that the ***port*** is setup properly in *SAP Netweaver Administrator –> SOA tab –> SOA Middleware Global Settings –> HTTP Proxy tab*.
3. Take the ***URL*** and provide it in the Web Service Navigator to test your webservice - the webservice shows the defined fields provided in the definition of the datasource. After the import *manage PSA* will show the new created entry.



The XML-Schema defines the rules how the XML document has to be structured. The document used to exchange data needs to be *well-formed* and *valid*. Meaning it has to contain all rules according to W3C.

# Создание oData сервиса. Часть 1.

<https://sapcode.ru/%D1%81%D0%BE%D0%B7%D0%B4%D0%B0%D0%BD%D0%B8%D0%B5-odata-%D1%81%D0%B5%D1%80%D0%B2%D0%B8%D1%81%D0%B0-%D1%87%D0%B0%D1%81%D1%82%D1%8C-1/?ysclid=lagddqgc2a705146351>

<https://sapcode.ru/%D1%81%D0%BE%D0%B7%D0%B4%D0%B0%D0%BD%D0%B8%D0%B5-odata-%D1%81%D0%B5%D1%80%D0%B2%D0%B8%D1%81%D0%B0-%D1%87%D0%B0%D1%81%D1%82%D1%8C-1/> - 21.04.2022

# [Содержание](#Содержание)

***oData*** /Open Data Protocol/ – открытый веб-протокол передачи данных. Общение по протоколу осуществляется по HTTP путем обмена *json* и *xml*.

Пример - Создание сервиса для работы с данными таблиц *SPFLI* и *SFLIGHT*.

ICF-узел = *ODATA*

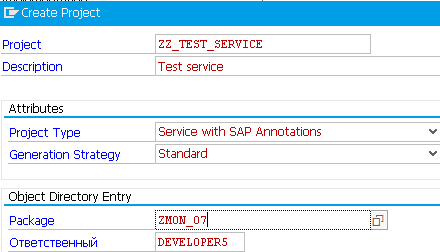
SAP System Alias = *LOCAL*

BEP = *ZZ\_TEST\_SERVICE\_SRV*

***Step 1*** – *Создание сервиса*

# [Содержание](#Содержание)

Ведение сервисов осуществляется в транзакции ***SEGW*** - *SAP Gateway Service Builder*.

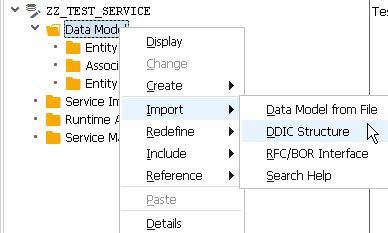


После подтверждения открывается *новый пустой сервис* с заранее определенными каталогами. Необходимо создать сущности с требуемыми параметрами. Для начала попробуем вывести таблицу ***SPFLI***.

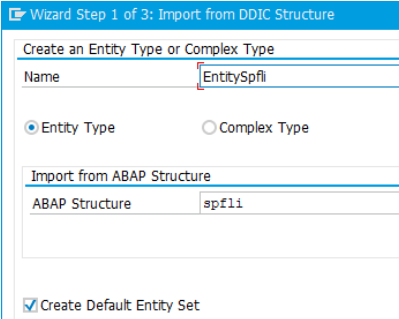
Существует несколько вариантов создания сервиса

* *создать* самим построчно;
* *импортировать* из структуры словаря /ABAP Structure/;
* [*импортировать сервис* из другой системы](http://sapcode.ru/%d1%8d%d0%ba%d1%81%d0%bf%d0%be%d1%80%d1%82%d0%b8%d1%80%d0%be%d0%b2%d0%b0%d1%82%d1%8c%d0%b8%d0%bc%d0%bf%d0%be%d1%80%d1%82%d0%b8%d1%80%d0%be%d0%b2%d0%b0%d1%82%d1%8c-odata-%d1%81%d0%b5%d1%80%d0%b2/).

В данном случае целесообразно импортировать из словаря.



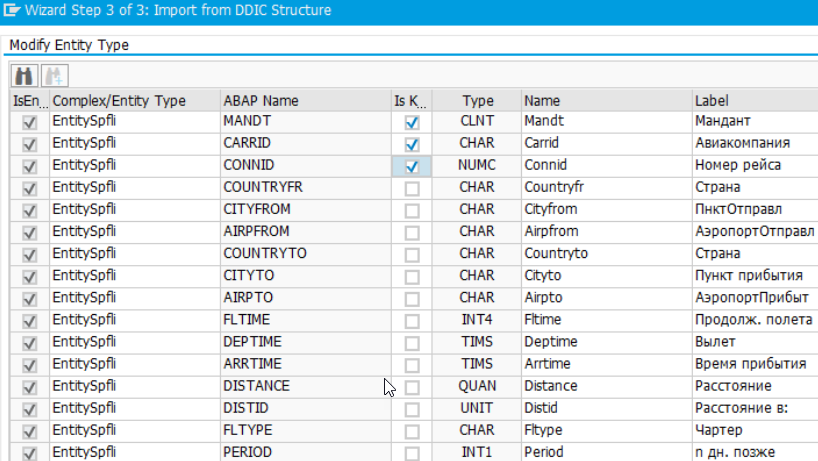
Далее вводим *имя сервиса*, а также выбираем *структуру для импорта*.



*Rem*

* В отличие от ABAP тут есть *регистрозависимость*сущностей и атрибутов.
* Галка Create Default Entity Set означает, что в добавок к сущности для одной записи /выходной структуре/, будет создана сущность для набора записей /выходная таблица/.

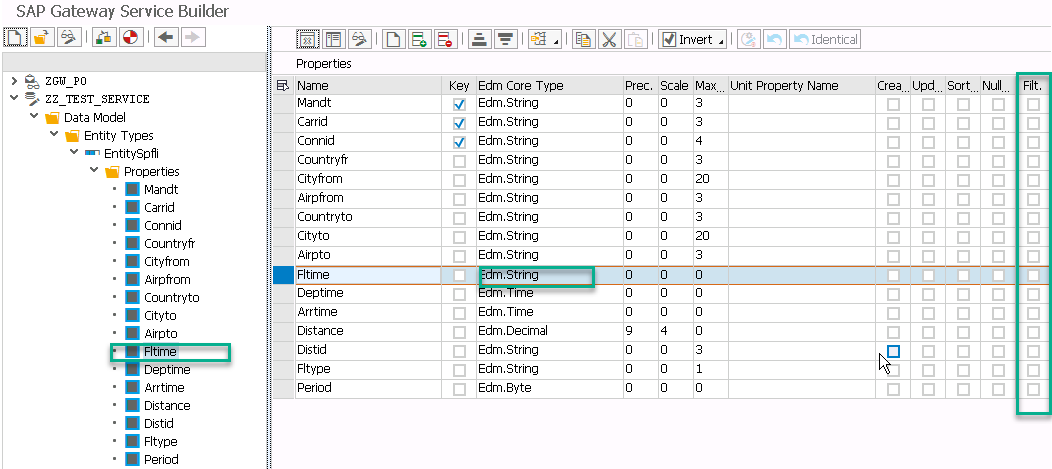
В нашем случае выбираем все поля. Также выбираем *ключевые поля* для сущности - их можно указать, когда захотим указать условия чтения при запуске нашего сервиса



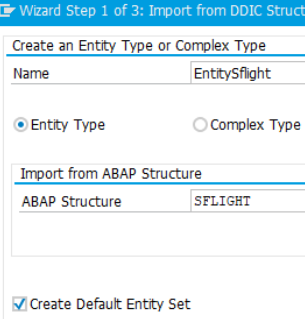
В полученном окне появятся созданные свойства функции. Здесь их можно изменить при необходимости.

В нашем случае стоит заменить тип у поля *Fltime* с INT4 на *Edm.String* /со строками меньше проблем/.

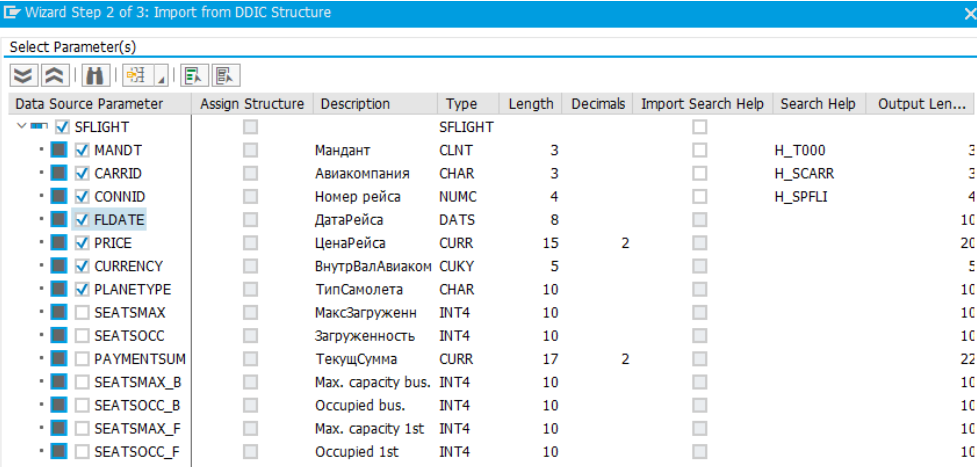
Также можно указать некоторые поля как *поля фильтра*, установив галку в поле ***Filt*** - такие поля можно будет использовать для ограничения выборки при попытке получить Set.

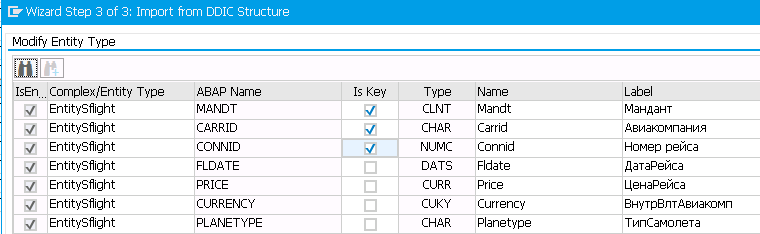


Таким же образом добавим еще одну сущность в выборку. Теперь с таблицей ***SFLIGHT***

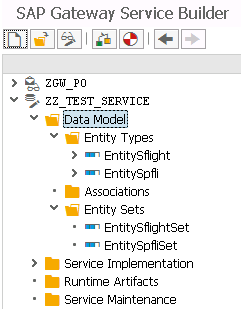


Из общего списка выберем только некоторые поля

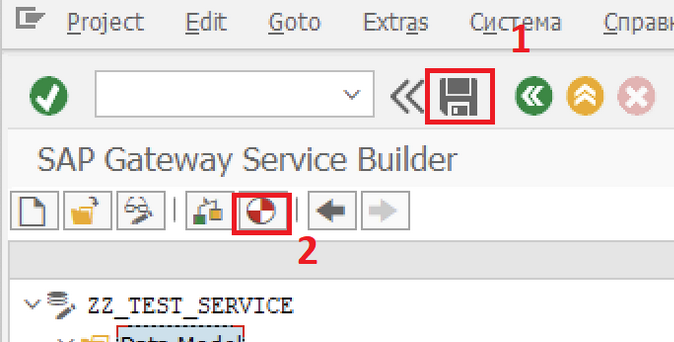




Структура сервиса теперь выглядит так

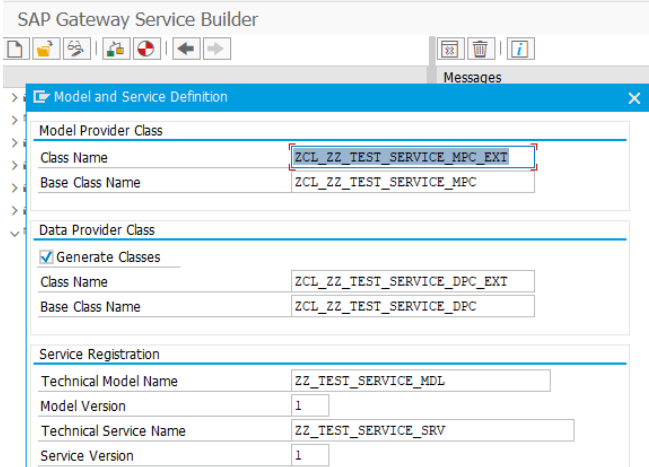


Сохраняем сервис и генерируем к нему объекты.



Rem

Если в будущем мы будем изменять сервис, каждый раз надо генерировать его заново, а также [*сбросить кэши*](http://sapcode.ru/odata-%d0%be%d1%87%d0%b8%d1%81%d1%82%d0%ba%d0%b0-%d0%ba%d1%8d%d1%88%d0%b0-%d0%bc%d0%be%d0%b4%d0%b5%d0%bb%d0%b8/)



После генерации появится ряд объектов, которые можно модифицировать под свои нужды, - это классы \*\_MPC\_EXT и \*\_DPC\_EXT.

Rem

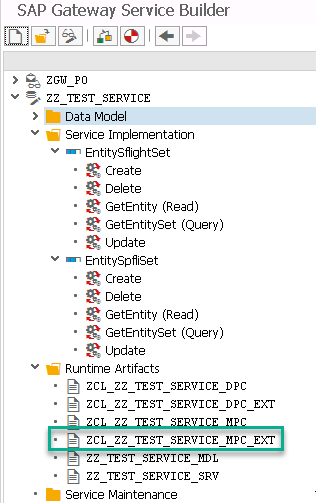
Классы \*\_MPC и \*\_DPC не меняются разработчиком вручную.

Для работы с ***сущностями*** нужен *\*\_DPC\_EXT* - в нем для каждой из сущностей после генерации будет созданы методы

* \*\_CREATE\_ENTITY – создание записей,
* \*\_UPDATE\_ENTITY – обновление записей,
* \*\_DELETE\_ENTITY – удаление записей,
* \*\_GET\_ENTITY – для получения одной записи,
* \*\_GET\_ENTITY\_SET – для получения списка записей.

Класс *\*\_MPC\_EXT* – служит для изменения ***глобальных параметров сервиса***.

Проваливаемся в класс ZCL\_ZZ\_TEST\_SERVICE\_MPC\_EXT и реализуем методы.



Метод ***\_GET\_ENTITY*** *получает записи по ключу* ⇒ на входе имеет таблицу со значениями ключей. Ключами являются поля, которые мы указали как ключи в соответствующей сущности.

METHOD ***entityspfliset\_get\_entity***.

DATA(lv\_mandt) = it\_key\_tab[ name = 'Mandt' ]-value.

DATA(lv\_carrid) = it\_key\_tab[ name = 'Carrid' ]-value.

DATA(lv\_connid) = it\_key\_tab[ name = 'Connid' ]-value.

SELECT SINGLE \* FROM spfli INTO *er\_entity* WHERE carrid = *lv\_carrid* and connid = *lv\_connid*.

ENDMETHOD.

METHOD ***entitysflightset\_get\_entity***.

DATA(lv\_mandt) = it\_key\_tab[ name = 'Mandt' ]-value.

DATA(lv\_carrid) = it\_key\_tab[ name = 'Carrid' ]-value.

DATA(lv\_connid) = it\_key\_tab[ name = 'Connid' ]-value.

SELECT SINGLE \* FROM sflight INTO *er\_entity* WHERE carrid = *lv\_carrid* and connid = *lv\_connid*.

ENDMETHOD.

Метод ***\_GET\_ENTITY\_SET*** *выдает список записей* и на входе у него - список значений для фильтра.

METHOD ***entitysflightset\_get\_entityset***.

DATA(lr\_mandt) = *VALUE #*( it\_filter\_select\_options[ property = 'Mandt' ]-select\_options

OPTIONAL ).

DATA(lr\_carrid) = VALUE #( it\_filter\_select\_options[ property = 'Carrid' ]-select\_options

OPTIONAL ).

DATA(lr\_connid) = VALUE #( it\_filter\_select\_options[ property = 'Connid' ]-select\_options

OPTIONAL ).

SELECT \* FROM sflight INTO TABLE *et\_entityset* WHERE carrid IN *lr\_carrid* AND

connid IN lr\_connid.

ENDMETHOD.

METHOD ***entityspfliset\_get\_entityset***.

DATA(lr\_mandt) = VALUE #( it\_filter\_select\_options[ property = 'Mandt' ]-select\_options

OPTIONAL ).

DATA(lr\_carrid) = VALUE #( it\_filter\_select\_options[ property = 'Carrid' ]-select\_options

OPTIONAL ).

DATA(lr\_connid) = VALUE #( it\_filter\_select\_options[ property = 'Connid' ]-select\_options

OPTIONAL ).

SELECT \* FROM spfli INTO TABLE *et\_entityset* WHERE carrid IN *lr\_carrid* AND

connid IN *lr\_connid*.

ENDMETHOD.

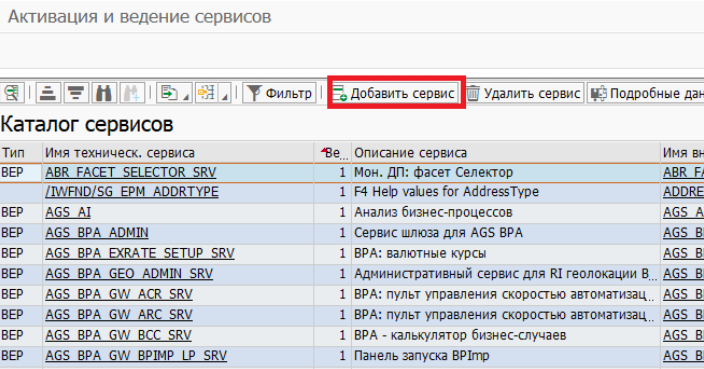
***Step 2*** – Активация сервиса

# [Содержание](#Содержание)

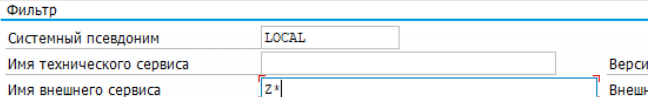
Для полноценной работы сервиса требуется его ***регистрация*** - перейдем в транзакцию **/IWFND/MAINT\_SERVICE** и добавим сервис.

*Rem*

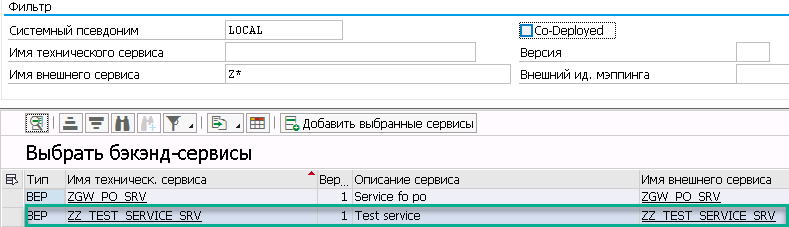
If the t-code **/IWFND/MAINT\_SERVICE** does not work try adding /N at the front of it - ***/n/IWFND/MAINT\_SERVICE***. For some reason this can help with transactions that start with a ***/***.



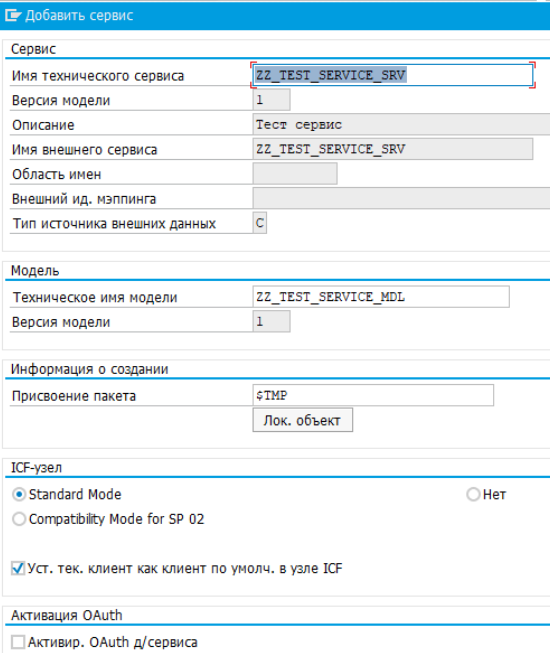
Введем ***Системный псевдоним*** /в общем случае - LOCAL; если планируются вызывать сервис из другой системы, то имя даем соответствующее/. [Существуют более сложные варианты](http://sapcode.ru/odata-aliases-%d1%80%d0%b0%d0%b1%d0%be%d1%82%d0%b0-%d1%81-%d0%bd%d0%b5%d1%81%d0%ba%d0%be%d0%bb%d1%8c%d0%ba%d0%b8%d0%bc%d0%b8-%d1%81%d0%b8%d1%81%d1%82%d0%b5%d0%bc%d0%b0%d0%bc%d0%b8/)



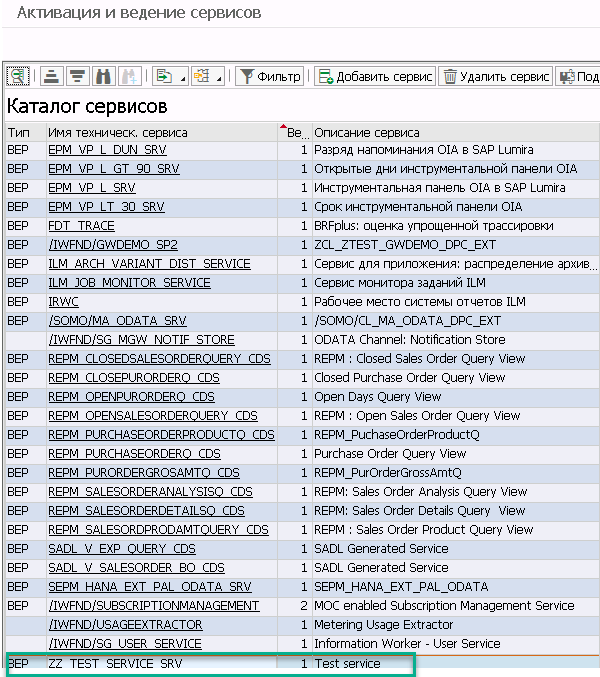
Жмем ENTER и в открывшемся списке выбираем нужный сервис.



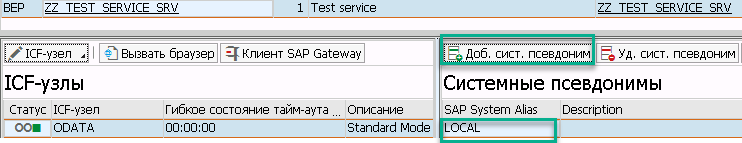
Указываем пакет



Возвращаемся в транзакцию /n**/IWFND/MAINT\_SERVICE** - тут можем найти новый сервис <oData name>\_SRV.



Дополнительно надо *положить запись ALIAS в настроенный запрос* /иначе есть риск, что при переносе запись потеряется/

.

***Step 3*** – Активация SAP *Netweaver Gateway* /если не активен/

# [Содержание](#Содержание)

*/n****/IWFND/IWF\_ACTIVATE*** – активация Netweaver Gateway /see <https://answers.sap.com/questions/10298396/sap-netweaver-gateway-has-been-deactivated.html/>.

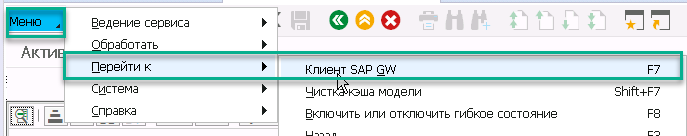
***Step 4*** – *Тестирование сервиса*

# [Содержание](#Содержание)

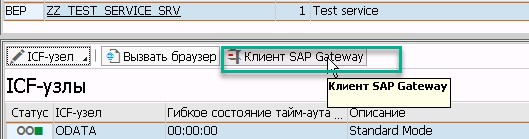
Теперь можно перейти в Gateway Client

1. Из каталога сервисов

Способ 1



Способ 2



1. Через транзакцию /N***/IWFND/GW\_CLIENT***

И протестировать

/sap/opu/odata/sap/ZZ\_TEST\_SERVICE\_SRV/?$format=xml

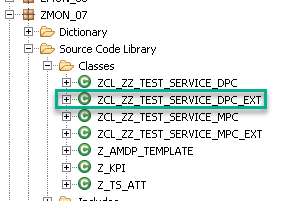
/sap/opu/odata/sap/ZZ\_TEST\_SERVICE\_SRV/$metadata

/sap/opu/odata/sap/ZZ\_TEST\_SERVICE\_SRV/EntitySpfliSet(Mandt='100',Carrid='AA', Connid='0017')

*Error*

<message xml:lang="ru">Method 'ENTITYSPFLISET\_GET\_ENTITY' not implemented in data provider class</message>

− Make sure that you have redefined **'EMPLOYEESET\_GET\_ENTITYSET' method ... in your \_DPC\_EXT class and you have to write the logic there /from** <https://answers.sap.com/questions/11978417/entityset-not-implemented-in-data-provider-class.html>/



See <https://blogs.sap.com/2016/12/05/tutorial-build-your-own-sap-fiori-approve-purchase-order-app-part-12/>

# Создание oData сервиса. Часть 2. Function import

<https://sapcode.ru/%D1%81%D0%BE%D0%B7%D0%B4%D0%B0%D0%BD%D0%B8%D0%B5-odata-%D1%81%D0%B5%D1%80%D0%B2%D0%B8%D1%81%D0%B0-%D1%87%D0%B0%D1%81%D1%82%D1%8C-2-function-import/?ysclid=lagdf8lyyz30111915>

# PI REST Adapter: Connect to BW/4HANA using X-CSRF-Token

<https://blogs.sap.com/2019/09/05/pi-rest-adapter-connect-to-bw4hana-using-x-csrf-token/>

# HTTPS POST-запрос от Java-приложения к OData-Service в системе S/4HANA с действительным токеном CSR

# [Содержание](#Содержание)

Ситуация следующая.

С одной стороны - *OData-Service*, который должен создавать запись при получении POST-запроса. Сервис создается в системе S/4HANA и доступен через SAP-Gateway.

С другой стороны - *Java-приложение* /OpenJDK 11/, которое выполняет цикл и в каждом цикле должно выдавать POST-запрос к OData-Service.

*Сеанс пользователя* идентифицируется файлами cookie, передаваемыми в HTTPS-заголовке.

Поскольку *CSRF-токен* действителен только для определенного ***сеанса*** пользователя - необходимо сделать следующее /см. <https://blogs.sap.com/2021/06/04/how-does-csrf-token-work-sap-gateway/>

1. Открыть сеанс, отправив GET запрос c указанием заголовка для получения CSRF-токена и сеансовых файлов cookie.

Поля заголовков - *x-csrf-token* = "fetch" и *set-cookie* = "fetch".

1. Сохранить полученные CSRF-токен и сеансовые файлы cookie, поскольку они нужны для POST-запроса.
2. Выполнить POST-запрос, установив файлы cookie сеанса и CSRF-токен из сохраненных значений.

Также важно отметить, что CSRF-токен, а также файлы cookie сеанса истекают после предоставленного или скорректированного периода времени или каких-либо изменений в сеансе, и оба должны быть получены заново /см. [https://blogs.sap.com/2021 /06/04/how-does-csrf-token-work-sap-gateway/#comment-575524](https://question-it.com/%20%20/blogs.sap.com/2021/06/04/how-does-csrf-token-work-sap-gateway/#comment-575524)/.

Пример рабочего кода

import javax.net.ssl.HttpsURLConnection;

import java.io.\*;

import java.net.Authenticator;

import java.net.PasswordAuthentication;

import java.net.URL;

public class **ODataLogger** {

private String sessionCookies;

private String csrfToken;

public ODataLogger() {}

public void ***logOdata*** (String user, String pass, String jsonBody) throws IOException {

this.setDefaultAuthenticator(user, pass);

fetchSessionHeaderFields();

postRequest(jsonBody);

}

private void ***setDefaultAuthenticator*** (String user, String pass) {

Authenticator auth = new *Authenticator*() {

@Override

protected PasswordAuthentication ***getPasswordAuthentication***() {

return new *PasswordAuthentication*(user, pass.toCharArray());

}

};

Authenticator.*setDefault*(auth);

}

private void ***fetchSessionHeaderFields***() throws IOException {

URL url = new URL("<my-service-link>");

HttpsURLConnection conn = (HttpsURLConnection) url.*openConnection*();

conn.*setRequestMethod*("GET");

conn.*setRequestProperty*("Content-Type", "application/json");

conn.*setRequestProperty*("*x-csrf-token*", "fetch");

conn.*setRequestProperty*("*set-cookie*","fetch");

*// Reading Response*

int status = con.*getResponseCode*();

Reader streamReader = null;

if (status < 299) {

StringBuffer sb = new *StringBuffer*(conn.*getHeaderFields*().*get*("set-cookie").toString());

*// Delete trailing and leading character*

sb.*deleteCharAt*(this.sessionCookies.length()-1);

sb.*deleteCharAt*(0);

this.sessionCookies = sb.toString();

this.csrfToken = conn.*getHeaderField*("x-csrf-token");

return;

}

}

private void ***postRequest***(String jsonBody) throws IOException {

*// Creating the connection*

URL url = new URL("<my-service-link>");

HttpsURLConnection conn = (HttpsURLConnection) url.openConnection();

conn.setRequestMethod("POST");

conn.setRequestProperty("Content-Type", "application/json");

conn.setRequestProperty("x-csrf-token", *this.csrfToken*);

conn.setRequestProperty("Cookie", *this.sessionCookies*);

conn.setRequestProperty("Accept", "application/json");

*// Setting JSON Body*

conn.*setDoOutput*(true);

try(OutputStream os = conn.*getOutputStream*()) {

byte[] input = jsonBody.*getBytes*("utf-8");

os.*write*(input, 0, input.length);

}

*// Reading response*

int status = con.*getResponseCode*();

Reader streamReader = null;

if (status > 299) {

streamReader = new InputStreamReader(conn.getErrorStream());

} else {

streamReader = new *InputStreamReader*(conn.*getInputStream*());

}

BufferedReader in = new *BufferedReader*(streamReader);

String inputLine;

StringBuffer content = new *StringBuffer*();

while ((inputLine = in.*readLine*()) != null) {

content.*append*(inputLine);

}

in.close();

conn.disconnect();

# [SAPUI5 - Gateway Errors on update method](https://stackoverflow.com/questions/51657340/sapui5-gateway-errors-on-update-method)

# <https://stackoverflow.com/questions/51657340/sapui5-gateway-errors-on-update-method>

I can't seem to get update methods working in ***SAPUI5***.

Here's the example

oModel.update("/JobOfflineSet('" + self.jobId + "')", oEntry, {

success: function () {

var oRouter = sap.ui.core.UIComponent.getRouterFor(self);

oRouter.navTo("main", {});

dialog.close();

},

error: function (oError) {}

});

And I'm receiving the following error.

There's no error shown in the SAP Gateway and we aren't even getting into ABAP to debug.

{"error":{"code":"/IWBEP/CM\_MGW\_RT/022","message":{"lang":"en","value":"The system cannot return your search. Please try again."},"innererror":{"application":{"component\_id":"","service\_namespace":"/SAP/","service\_id":"ZSV\_SURVEY\_SRV","service\_version":"0001"},"transactionid":"306596E88F59F1CD80C7005056BEAC32","timestamp":"","Error\_Resolution":{"SAP\_Transaction":"","SAP\_Note":"See SAP Note 1797736 for error analysis (https://service.sap.com/sap/support/notes/1797736)","Batch\_SAP\_Note":"See SAP Note 1869434 for details about working with $batch (https://service.sap.com/sap/support/notes/1869434)"},"errordetails":[{"code":"/IWBEP/CX\_MGW\_BUSI\_EXCEPTION","message":"The system cannot return your search. Please try again.","propertyref":"","severity":"error","target":""}]}}}