Andre Fischer

September 30, 2013 | 16 minute read

How to Implement Basic Delta Query Support in SAP NetWeaver Gateway

Updates

- 04.03.2014 –> Added a link to document that describes how to consume a service that desribes the OData Delta Query mechanism

Objectives

In this how-to guide, you will learn how to implement Delta Token capabilities with SAP NetWeaver Gateway to enable offline functionality. This how-to guide assumes that you have already worked your way through the following how-to guides:

1. “How to Develop a Gateway Service using Code based Implementation” https://scn.sap.com/docs/DOC-43030 and

2. “How to Develop Query Options for an OData Service Using Code-Based Implementation” http://scn.sap.com/docs/DOC-43000

These guides describe the basic implementation steps of the service that we are now going to enhance. The sample service uses demo data from the Enterprise Procurement Model, which is part of every SAP NetWeaver ABAP server as of 7.02.
After completing these steps, you will be able to:
• Enhance the GET_ENTITYSET and GET_ENTITYSET_DELTA methods of an entity set to support the delta query functionality.
Also check out the following document How To Enable Delta Queries using Syclo Exchange Framework and SAP NetWeaver Gateway that explains how delta queries can leverage the Syclo Exchange Framework.

Business Example

You want to build an application that shows a list of products. The application shall be able to retrieve only those products that have been newly created, changed or deleted after having retrieved a complete list of products as an initial load.

Example of a consumer Application

New: Check out the following document: Going Offline With Kapsel Now

Short Introduction to Delta Token Functionality

OData exposes collections of records as EntitySets. These EntitySets support various options like sorting, filtering or paging to minimize the amount of data that is sent accross the wire.
Paging is a very powerful option to reduce the size of the payload and improve performance. To use paging on client side, the OData request needs to specify the $skip and $top tokens. For server-driven paging you have to use the $skiptoken. Some applications, however, prefer to use client side caching rather than paging. In such cases, SAP NetWeaver Gateway provides Delta Tokens that help to reduce resource consumption on the server and the wire when the same request is sent several times from the same client. The OData Delta Token enables clients to use caching when retrieving entity sets. Consequently, the client gets only the changed records after an initial retrieval of all records. Therefore, the initial request will return all records and a Delta Token. The subsequent request (with exactly the same query options) sends the Delta Token from the first request. The response will then contain only the changed records. If no records have been changed since the first request then the response will not contain any records at all.
You can implement Delta Token in several ways and these options can be grouped into two main approaches:

1. The first approach, which is based on Syclo Exchange framework, calculates deltas **at modification time**: The ABAP system tracks relevant changes when they occur. At request time, the deltas are already prepared and thus available. On the one hand this approach requires more development. On the other hand it's more scalable and has an optimized overall performance.
2. The second approach is based on delta determination **at request time** where the system compares old and new state to find out which records have been changed/deleted. The implementation effort is rather small but it does not optimize the performance of the backend. That means, the more records you have in the full collection, the longer the response time of the request.

With both approaches, the payload of the response is reduced but as described only the first approach is also able to optimize the performance in the backend.
In this guide we will look at the basic implementation, using the Delta Request Log Component. The Delta Request Log Component is based on the second approach. Therefore the implementation effort is rather small but the performance of the backend is not optimized.

Prerequisites

You have implemented an OData service as described in the two how-to guides mentioned above.
If you are using NetWeaver 7.02, then you need at least software component IW_BEP Support Package 7 and software component IW_FND Support Package 7.
If you are on NetWeaver 7.40, then you need at least software component SAP_GWFND Support Package 4.
You have to implement following SAP notes: 1926946, 1932247 and 1970229.

Implementation Overview

In this how-to guide, you will create a new Entity Set Products_DQ that shall support delta queries in the existing sample service that you have created earlier. You will redefine the methods GET_ENTITYSET and GET_ENTITYSET_DELTA to support the delta query functionality by calling method /IWBEP/CL_QUERY_RESULT_LOG-> CREATE_UPDATE_LOG_ENTRY_HASH.

Task 1: Create an additional Entity Set Products_DQ

We will enhance the service you created earlier to support delta queries for product data. For this we create a new entity set called Products_DQ where we implement the delta query specific coding while leaving the original Products Entity Set unchanged.

Task 2: Enhance method ZCL_GW_PRODUCT_DPC_EXT->PRODUCTS_DQ_GET_ENTITYSET to support the delta query functionality

In this task you will enhance the existing GET_ENTITYSET method of the newly created entity set Products_DQ. Like the existing GET_ENTITYSET method of the Products entity set it should retrieve the application data containing product entries.
First you need to create a local data provider facade which will be needed later by the Delta Token functionality.
In the next step you will need to call method /IWBEP/CL_QUERY_RESULT_LOG-> CREATE_UPDATE_LOG_ENTRY_HASH for the delta query functionality by passing the local table containing the product entries and the local data provider facade. The method is the central method for the Delta Token functionality. It calculates a hash value for each product and stores the hash values in some application tables, so the entries can be later checked for changes. The method will also return a delta token which you need to export.

Step 1: Get the data provider facade (you need to pass the facade later to the Delta Token method)

```
* get the data provider facade
  TRY.
      lo_dp_facade = /iwbep/if_mgw_conv_srv_runtime~get_dp_facade( ).
    CATCH /iwbep/cx_mgw_tech_exception.
      RAISE EXCEPTION TYPE /iwbep/cx_mgw_tech_exception.
  ENDTRY.
```

Step 2: Call method /IWBEP/CL_QUERY_RESULT_LOG-> CREATE_UPDATE_LOG_ENTRY_HASH for the delta query functionality by passing local table et_entityset containing the product entries

```
* call the delta token functionality
  TRY.
   CALL METHOD /iwbep/cl_query_result_log=>create_update_log_entry_hash
        EXPORTING
          io_tech_request_context  = io_tech_request_context
          io_dp_facade             = lo_dp_facade
          ir_service_document_name = mr_service_document_name
          ir_service_version       = mr_service_version
          it_entityset             = et_entityset
        CHANGING
          ev_delta_token           = lv_delta_token.
    CATCH /iwbep/cx_qrl_locked.
      RAISE EXCEPTION TYPE /iwbep/cx_qrl_locked.
    CATCH /iwbep/cx_qrl_delta_unavailabl.
      RAISE EXCEPTION TYPE /iwbep/cx_qrl_delta_unavailabl.
  ENDTRY.
```

Step 3: Export the delta token which was calculated by the method in step (2)

```
  es_response_context-deltatoken = lv_delta_token.
```

Task 2: Re-define method ZCL_GW_PRODUCT_DPC_EXT->

/IWBEP/IF_MGW_APPL_SRV_RUNTIME~GET_ENTITYSET_DELTA to support the delta query functionality

In this task you will re-define method GET_ENTITYSET_DELTA. First, you need to retrieve the application data containing product entries, just like in GET_ENTITYSET method.
Then you need to create a local data provider facade which will be needed later by the Delta Token functionality.
In the next step you will need to call method /IWBEP/CL_QUERY_RESULT_LOG-> CREATE_UPDATE_LOG_ENTRY_HASH for the delta query functionality by passing the local table containing the product entries and the local data provider facade.
It calculates a hash value for each product. Those hash values are then compared with the hash values of the prior call (related to the same delta token ID) to determine which values have changed in the meantime. The method will return a new delta token which you need to export. The method will also return any entries that were changed in the meantime (please store these in local table lt_product) and any entries that ware deleted (please store these in local table lt_deleted_product).

Step 1: Re-define method /IWBEP/IF_MGW_APPL_SRV_RUNTIME~GET_ENTITYSET_DELTA

Step 2: Retrieve all relevant entries for this entity set (like in GET_ENTITYSET above)

Step 3: Get the data provider facade (you need to pass the facade later to the Delta Token method)

```
* get the data provider facade
  TRY.
      lo_dp_facade = /iwbep/if_mgw_conv_srv_runtime~get_dp_facade( ).
    CATCH /iwbep/cx_mgw_tech_exception.
      RAISE EXCEPTION TYPE /iwbep/cx_mgw_tech_exception.
  ENDTRY.
```

Step 4: Call method /IWBEP/CL_QUERY_RESULT_LOG-> CREATE_UPDATE_LOG_ENTRY_HASH

Call method /IWBEP/CL_QUERY_RESULT_LOG-> CREATE_UPDATE_LOG_ENTRY_HASH for the delta query functionality by passing local table et_entityset containing the product entries – just like in task 1, step 2 but this time take over the changed objects (and store them in local table lt_product) as well as the deleted objects (and store them in local table lt_deleted_product):

```
*  call the delta token functionality
  TRY.
     CALL METHOD /iwbep/cl_query_result_log=>create_update_log_entry_hash
          EXPORTING
            io_tech_request_context  = io_tech_request_context
            io_dp_facade             = lo_dp_facade
```

```
            ir_service_document_name = mr_service_document_name
            ir_service_version       = mr_service_version
            it_entityset             = lt_product
          IMPORTING
            et_deleted_entityset     = lt_deleted_product
            et_entityset             = lt_product
          CHANGING
            ev_delta_token           = lv_delta_token.
  CATCH /iwbep/cx_qrl_locked.
        RAISE EXCEPTION TYPE /iwbep/cx_qrl_locked.
  CATCH /iwbep/cx_qrl_delta_unavailabl.
        RAISE EXCEPTION TYPE /iwbep/cx_qrl_delta_unavailabl.
ENDTRY.
```

Step 5: Export the delta token which was calculated by the method in step (4)

```
  es_response_context-deltatoken = lv_delta_token.
```

Step 6: Export the deleted entity set

Export the deleted entity set – add deleted records (from table lt_deleted_product) to the deleted entries list er_deleted_entityset

```
*   export the deleted entity set
    copy_data_to_ref( EXPORTING
                       is_data = lt_deleted_product
                      CHANGING
                       cr_data = er_deleted_entityset ).
```

Step 7: Export the changed entity set – add changed records from lt_product to er_entityset

```
*   export the changed entity set
    copy_data_to_ref( EXPORTING
                       is_data = lt_product
                      CHANGING
                       cr_data = er_entityset ).
```

Implementation

ZCL_GW_PRODUCT_DPC_EXT->PRODUCTS_DQ_GET_ENTITYSET

The coding of the method:

```
DATA:
        lt_return          TYPE TABLE OF bapiret2,
        lv_delta_token     TYPE string,
        lo_dp_facade       TYPE REF TO /iwbep/if_mgw_dp_facade.

* retrieve all relevant entries for this entityset
  CALL FUNCTION 'BAPI_EPM_PRODUCT_GET_LIST'
    TABLES
      headerdata = et_entityset
      return     = lt_return.
* get the data provider facade
  TRY.
      lo_dp_facade = /iwbep/if_mgw_conv_srv_runtime~get_dp_facade( ).
    CATCH /iwbep/cx_mgw_tech_exception.
      RAISE EXCEPTION TYPE /iwbep/cx_mgw_tech_exception.
  ENDTRY.
* call the delta token functionality
  TRY.
      CALL METHOD /iwbep/cl_query_result_log=>create_update_log_entry_hash
        EXPORTING
          io_tech_request_context  = io_tech_request_context
          io_dp_facade             = lo_dp_facade
          ir_service_document_name = mr_service_document_name
          ir_service_version       = mr_service_version
          it_entityset             = et_entityset
        CHANGING
          ev_delta_token           = lv_delta_token.
    CATCH /iwbep/cx_qrl_locked.
      RAISE EXCEPTION TYPE /iwbep/cx_qrl_locked.
    CATCH /iwbep/cx_qrl_delta_unavailabl.
      RAISE EXCEPTION TYPE /iwbep/cx_qrl_delta_unavailabl.
  ENDTRY.

* export the delta token
  es_response_context-deltatoken = lv_delta_token.
```

ZCL_GW_PRODUCT_DPC_EXT-> /IWBEP/IF_MGW_APPL_SRV_RUNTIME~GET_ENTITYSET_DELTA

The coding of the method:

```
DATA:
        lt_return             TYPE TABLE OF bapiret2,
        lv_entity_set_name    TYPE string,
        lt_product            TYPE zcl_gw_product_mpc=>tt_product,
        ls_product            TYPE zcl_gw_product_mpc=>ts_product,
        lt_deleted_product    TYPE zcl_gw_product_mpc=>tt_product,
        ls_deleted_product    TYPE zcl_gw_product_mpc=>ts_product,
        lv_delta_token        TYPE string,
        lo_dp_facade          TYPE REF TO /iwbep/if_mgw_dp_facade.

  FIELD-SYMBOLS:
        &lt;lt_deleted_product>  TYPE zcl_gw_product_mpc=>tt_product,
        &lt;lt_product>          TYPE zcl_gw_product_mpc=>tt_product.

  lv_entity_set_name = io_tech_request_context->get_entity_set_name( ).

  IF lv_entity_set_name = 'Products_DQ'.

*   retrieve all relevant entries for this entityset
    CALL FUNCTION 'BAPI_EPM_PRODUCT_GET_LIST'
      TABLES
        headerdata = lt_product
        return     = lt_return.
*   get the data provider facade
    TRY.
        lo_dp_facade = /iwbep/if_mgw_conv_srv_runtime~get_dp_facade( ).
      CATCH /iwbep/cx_mgw_tech_exception.
        RAISE EXCEPTION TYPE /iwbep/cx_mgw_tech_exception.
    ENDTRY.
*   call the delta token functionality
    TRY.
```

```
        CALL METHOD /iwbep/cl_query_result_log=>create_update_log_entry_hash
           EXPORTING
             io_tech_request_context  = io_tech_request_context
             io_dp_facade             = lo_dp_facade
             ir_service_document_name = mr_service_document_name
             ir_service_version       = mr_service_version
             it_entityset             = lt_product
           IMPORTING
             et_deleted_entityset     = lt_deleted_product
             et_entityset             = lt_product
           CHANGING
             ev_delta_token           = lv_delta_token.
       CATCH /iwbep/cx_qrl_locked.
         RAISE EXCEPTION TYPE /iwbep/cx_qrl_locked.
       CATCH /iwbep/cx_qrl_delta_unavailabl.
         RAISE EXCEPTION TYPE /iwbep/cx_qrl_delta_unavailabl.
     ENDTRY.
*    export the delta token
     es_response_context-deltatoken = lv_delta_token.
*    export the deleted entity set
     copy_data_to_ref( EXPORTING
                        is_data = lt_deleted_product
                       CHANGING
                        cr_data = er_deleted_entityset ).

*    export the changed entity set
     copy_data_to_ref( EXPORTING
                        is_data = lt_product
                       CHANGING
                        cr_data = er_entityset ).

   ENDIF.
```

Test the service

1. Start the Gateway Client by calling transaction /IWFND/GW_CLIENT

2. Enter the following URI to test your implementation:
   /sap/opu/odata/sap/ZGW_PRODUCT_SRV/Products_DQ
3. You should get a response containing all products. Almost at the end of the XML document, you will get your delta token (of course in your response you will get a different ID)
   <link rel="**delta**" href="**Products_DQ?!deltatoken= '005056A2025C1EE2BFE687AFDC2FAAF4_20130807073741'**" />
4. Now enter the URI from above again, but this time add your delta token (copy and paste it from your response). You URI should look something like this (but will have a different ID):
   /sap/opu/odata/sap/ZGW_PRODUCT_SRV/Products_DQ?!deltatoken='005056A2025C1EE2BFE687AFDC2FAAF4_20130807073741'
5. After you click on execute, you will notice that the response does not contain any products at all. This is because the Delta Token functionality returns only changed records and no products were changed in the meantime.
6. You could now change a product and execute exactly the same URI as in step (4). The response should contain exactly one product.

Technical Restrictions

Key Fields of Entity Type

There are strict restrictions regarding the key fields of Entity Types that are used with the basic delta query functionality:

- The maximum legth (output length) of a key field must not exceed 200 characters
- If several key fields were defined for one Entity Type then the combined legth (output length) of those key fields must not exceed 200 characters
- The key field type must have a fixed legth; therefore types of variable legth (e.g. type String) are not permitted

JSON Format

Currently the JSON format is not supported in combination with the delta query functionality.

Expand System Query Option ($expand)

Currently the Expand System Query Option ($expand) is not supported in combination with the delta query functionality.

Paging

Both the server-driven paging and the client-driven paging are not supported in combination with the delta query functionality. As a result, the following query options are not supported in combination with the delta query functionality:

- Skip Token System Query Option ($skiptoken)
- Top System Query Option ($top)
- Skip System Query Option ($skip)

Further Notes

Clean-Up Report

The Delta Token functionality based on Delta Request Log Component uses two database tables to store hash values: /IWBEP/D_QRL_HDR and /IWBEP/D_QRL_ITM. Once this offline functionality is used productively the size of these tables will increase with each request. We offer areport that can be run to clean up outdated entries: /WBEP/R_CLEAN_UP_QRL. We suggest that you schedule a job for this report so that it is executed periodically, for instance on a daily basis.

Assigned tags

SAP Gateway

former jive document

Similar Blog Posts

#3 – How To Implement Lightening Fast OData Services with Exchange Table

By Kenichi Unnai Apr 03, 2015

Deploying SAP NetWeaver Gateway in mixed (7.40 and earlier) environments

By Andre Fischer Nov 27, 2013

HowTo Guides - Subscription & Notification with SAP NetWeaver Gateway - Series

By Former Member May 07, 2012

Related Questions

SAP MCFU - heavy load of requests managed by queues?

By Lakshman Balanagu Jan 04, 2018

Server side caching

By Former Member Jan 07, 2016

Server side caching

By Former Member Jan 07, 2016

26 Comments

You must be Logged on to comment or reply to a post.

Aliaksandr Shchurko

January 6, 2014 at 9:36 am

Thanks!

Like 0 | Share

Jibin Joy

January 17, 2014 at 11:33 am

Thanks ....

Like 0 | Share

Derek Wichmann

January 20, 2014 at 9:54 pm

Thanks for the write up. I'm happy to see there is a low effort generic method for performing deltas. I have a couple questions regarding performance of this method:

** Regarding the modification time (syclo exchange tables) vs. request time delta calculation, can you provide actual numbers on the performance difference? Perhaps number of seconds required to perform calculation on X number of Products?

** I assume the cost to calculate the delta at request time (`create_update_log_entry_hash`) depends on the complexity of the business object. Can you tell me for how many records (say for this example, Product) this basic delta calculation is no longer performant?

Like 0 | Share

Andre Fischer | Blog Post Author

March 4, 2014 at 3:31 pm

Check out the following document: Going Offline With Kapsel Now that describes the development of a consumer app.

Like 0 | Share

Sreenivas Pachva

May 3, 2014 at 9:50 am

Hi Andre,

Nice Document and Thanks for sharing!!!!!!

Thanks

Pachva

Like 0 | Share

Former Member

July 28, 2014 at 8:40 pm

Thanks Andre for informative blog,

As we are implementing entity set method.. can we use filter option along with Delta query. When i tired to GET data using filter option - I received the delta token but it was not working when I used delta token to fetch changed data..

Can we use filter option along with Deltatoken in URI

Regards,

Rajesh

Like 0 | Share

Former Member

January 6, 2015 at 11:15 am

Hi Rajesh,

Did you get any answer for your question?

If yes, please respond with the answer you got.

Best Regards,

C Vinish

Like 0 | Share

Former Member

March 13, 2015 at 8:56 pm

Nice Blog...

Do anybody know when delta query functionality support JSON format?

Best Regards,

Jan

Like 0 | Share

Former Member

April 29, 2015 at 3:24 am

We seriously need Delta Token support for JSON data, It doesn't make sense to keep it waiting for more than 2 years now.

We do a lot of data interchange and XML is causing us Huge performance problems.

Anybody has an update ?

Thanks

Pradeep Gudipati

Like 0 | Share

Former Member

April 30, 2015 at 8:13 am

We have same problem, our customers need performance...

I asked SAP on Webinar and somebody told me that is in roadmap with Odata v4, but without release date!

Actual Roadmap:

https://websmp207.sap-ag.de/~sapidb/011000358700000091542012D.pdf

Best Regards,

Jan

Like 0 | Share

Former Member

April 1, 2016 at 7:19 pm

Currently we have the same problem and we would need JSON support, our API is developed under JSON and delta token in XML is difficult to change now..

Any workaround in hybrid apps?

Like 0 | Share

Manfred Scheiner

April 27, 2016 at 3:46 pm

We have the same problem. For me that's the biggest drawback of SAP Gateway today, which shouldn't be left that way by SAP.

Like 0 | Share

Vishal VK

January 25, 2018 at 2:50 am

We are also facing this issue.

Does delta query now support JSON format ?

Like 0 | Share

Former Member

April 1, 2015 at 11:49 am

Hi Andre,

Is it possible to use $filter option with delta token URI.

Best Regards,

C Vinish

Like 0 | Share

Former Member

April 2, 2015 at 9:46 am

Hello Vinish,

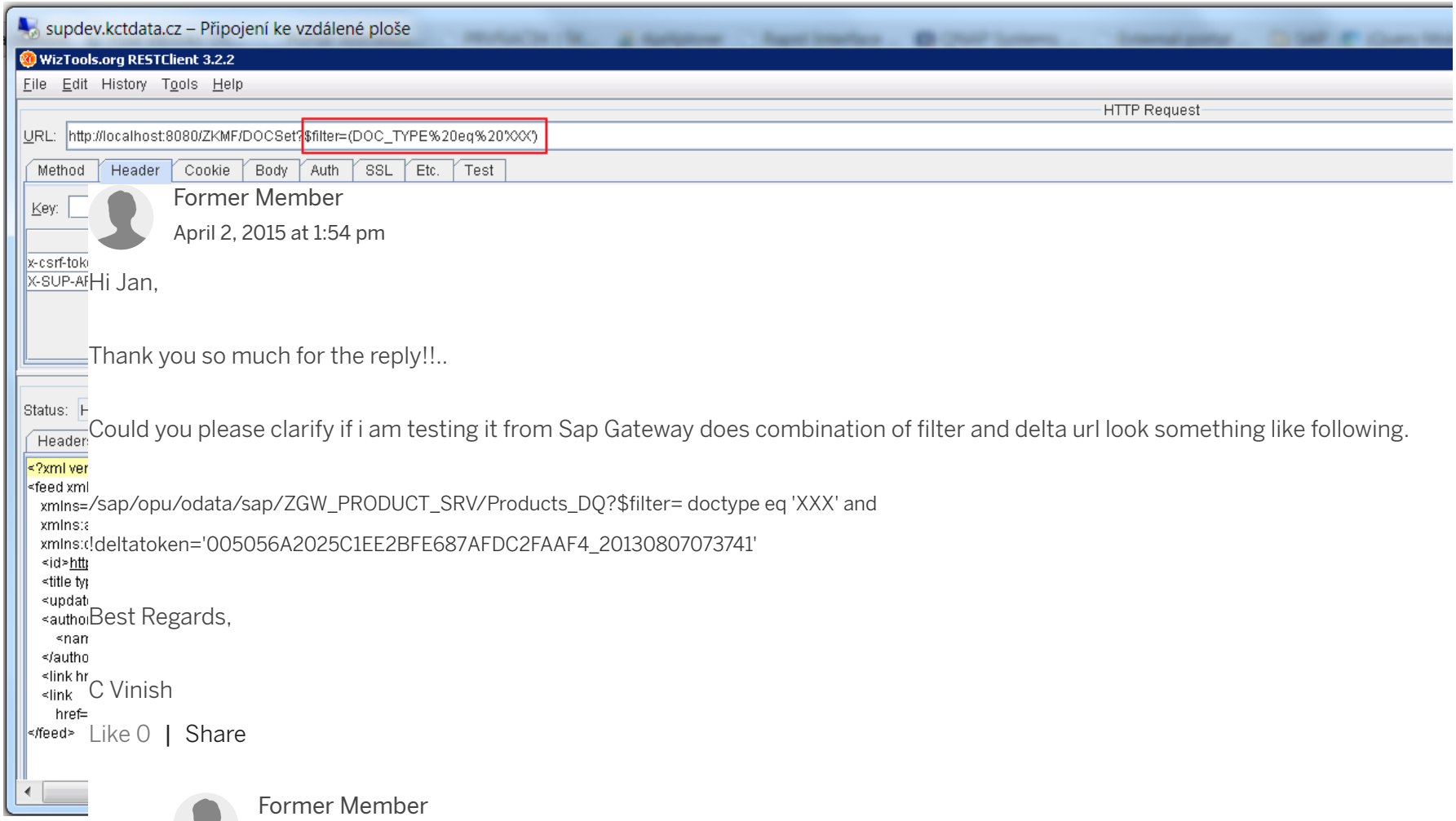
Yes it's possible, data are filtered correctly. Filter atribut and other Odata URL attributes are filled to get that use class attribut **mr_request_details** in method **GET_ENTITYSET_DELTA.** Like this see code:

```
FIELD-SYMBOLS <fls_request_data> TYPE /iwbep/if_mgw_core_srv_runtime=>ty_s_mgw_request_context.
ASSIGN me->mr_request_details->* TO <fls_request_data>.
IF sy-subrc EQ 0 AND <fls_request_data> IS ASSIGNED.
  et_filter[] = <fls_request_data>-filter_select_options[].
  et_key[] = <fls_request_data>-key_tab[].
  MOVE-CORRESPONDING <fls_request_data>-paging TO es_paging.
  et_navigation_path[] = <fls_request_data>-navigation_path[].
  et_order[] = <fls_request_data>-order[].
ENDIF.
```

Remark: But we found bug that filter is missing in response url tag see screen when you comunicate throw SAP Mobile Platform. This error I reported to SAP month ago and SAP still fighting with fix that...

Best Regards,

Jan



Former Member

April 2, 2015 at 1:54 pm

Hi Jan,

Thank you so much for the reply!!..

Could you please clarify if i am testing it from Sap Gateway does combination of filter and delta url look something like following.

/sap/opu/odata/sap/ZGW_PRODUCT_SRV/Products_DQ?$filter= doctype eq 'XXX' and

!deltatoken='005056A2025C1EE2BFE687AFDC2FAAF4_20130807073741'

Best Regards,

C Vinish

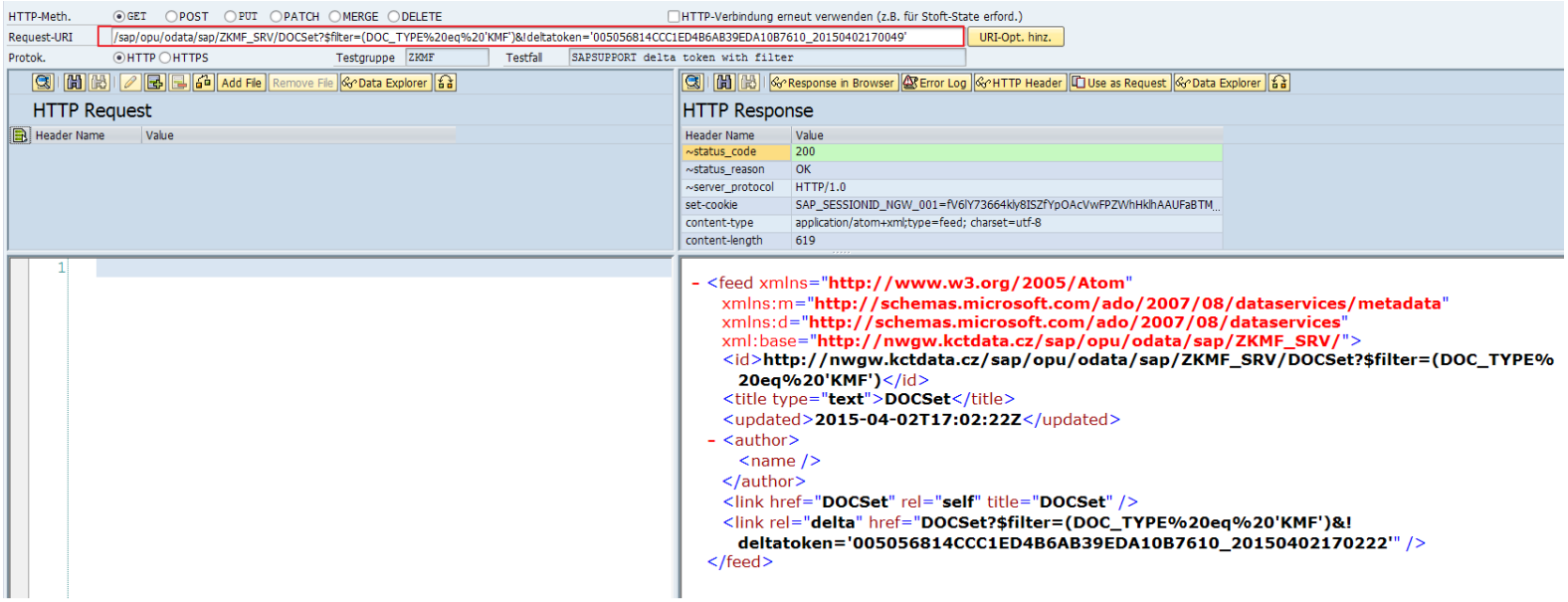
Like 0 | Share

Former Member

April 2, 2015 at 5:06 pm

Hi Vinish,

see URL in screenshot from transaction /IWFND/GW_CLIENT



Best Regards,

Jan

Like 0 | Share

Former Member

June 23, 2015 at 10:58 am

Hello,

We have implemented Basic Delta Query Support in SAP NetWeaver Gateway, following the guidelines explained in http://scn.sap.com/docs/DOC-47043

But there is a problem with $filter in a Delta Query scenario, as you can see below:

1st Request:

/sap/opu/odata/SAP/ZCONTRACTS_SRV/HierarchySet()?$filter=IvUserwin%20eq%20'MUNUEAL'&$format=xml

In response:

<link rel="delta" href="HierarchySet()?
$filter=IvUserwin%20eq%20'MUNUEAL'&$format=xml?!deltatoken='005056A326E01EE586B29DDCF362D39E_20150623104524'"/>

!deltatoken parameter is included, but with "?" operator, the correct one is "&"

If I manually change it with & it works ok.

Is there any known issue about it?

Product version:

SAP_GWFND 740 0010 SAPK-74010INSAPGWFND SAP Gateway Foundation 7.40

Thank you,

Former Member
June 23, 2015 at 4:11 pm

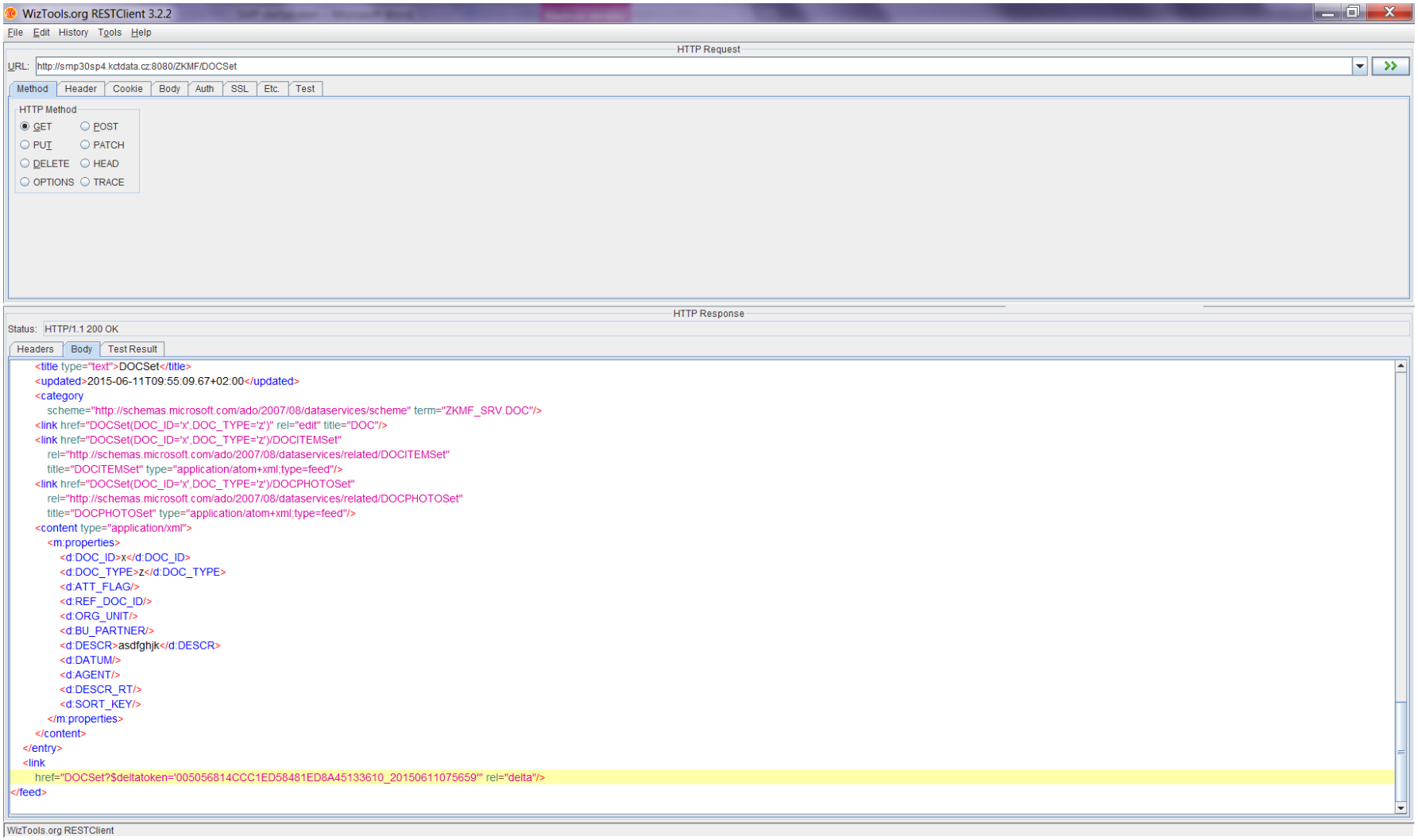
Hello Antoni,

I tried with previous version

SAP_GWFND 740 0009 SAPK-74009INSAPGWFND SAP Gateway Foundation

and it works delta token is separated by "&". Create OSS message and "fight" with SAP or downgrade Gateway.

Remark: btw I reported similar OSS message, when I communicate with Gateway by SMP 3.0 SP 7 PL 2 (3.0.7.2) and SAP replaces key word !deltatoken => $deltatoken !!! see attachment



Best Regards,

Jan

Like 0 | Share

Hemendra Sabharwal

July 30, 2015 at 4:20 am

Very nice blog, thanks for sharing Andre Fischer.

But I am stuck. Could you please suggest how to implement "Delta Token" when the WCF service from Microsoft SQL server is exposed as OData service and consumed through Integration Gateway/SMP 3.0 SP05 in Android Native Application.

Thanks,

Warm Regards

Hemendra

Like 0 | Share

Hemendra Sabharwal

July 30, 2015 at 6:49 am

I guess by using CTP for WCF, as suggested in below blog:

http://blogs.msdn.com/b/odatateam/archive/2011/04/20/using-microsoft-wcf-data-services-reference-data-caching-extensions-ctp.aspx

Thanks,

Warm Regards

Hemendra

Like 0 | Share

' Pavan ' Golesar
November 17, 2015 at 6:51 am

Hi,

Nice Blog,

Would be trying soon.

Thanks

--Pavan G

Like 0 | Share

Mike Doyle
June 19, 2017 at 10:50 am

Thanks Andre for a great blog. Is JSON format now supported with delta queries? From what release?

Like 0 | Share

Andre Fischer | Blog Post Author
June 19, 2017 at 11:04 am

Hi Mike,

it is planned to support delta queries with our implementation of OData V4.

For OData V2 delta queries were not part of the OData specification. Therefore we added support for the OData V2 protocol by using a custom query option "!deltatoken" rather than "$deltatoken".

Here we only support ATOM/XML because only in ATOM/XML we were able to use the concept of tombstones.

Regards,

Andre

Like 0 | Share

vivek gaurav
August 31, 2017 at 12:34 pm

Thanks Andre for very good blog.

I implemented it for our gateway service and it is working fine with delta token. But when i am integrating gateway service with SMP for kapsel based offline app. I am facing problem while passing delta token. because refresh does take any other parameter.So that we can get delta data from backend.

Do you know any blog with respect to passing delta token from kapsel.

I raised SCN ticket as well :-

https://answers.sap.com/questions/289958/pass-delta-token-to-kapsel-based-offline-sevice.html

Thanks in Advance

Like 0 | Share

Former Member

November 9, 2017 at 10:49 am

Very Nice Blog...Simple and step by step explanation is really helpful

Like 0 | Share

Find us on

Privacy	Terms of Use
Legal Disclosure	Copyright
Trademark	Cookie Preferences
Newsletter	Support