ent	Typing Method	Component Type	Data Type	Length	Deci...	Short Description
ID	Types	SIBFBORIID	CHAR	70	0	Instance Ident. in BOR Compat. Persistent Object Referen
TYPE	Types	SIBFTYPEID	CHAR	32	0	Type of Objects in Persistent Object References
CAT	Types	SIBFCATID	CHAR	2	0	Category of Objects in Persistent Object References
NT_ID	Types	DOCUMENTID	CHAR	36	0	Document ID
AME	Types	BITM_FILENAME	CHAR	255	0	File Name of Browser Item (Display Attribute)
YPE	Types	FPGRAPHICMIME	CHAR	64	0	MIME Type of Graphic
R	Types	SO_CRO_NAM	CHAR	12	0	Creator Name
R_NAME	Types	SGS_CRNAME	CHAR	60	0	Creator Name
O_ON	Types	/SAPTRX/TIMESTA...	CHAR	19	0	Formatted Date/Time

'Attachments' – Attach any File (PDF/DOC/JPG/XLS/TXT..) to BO using GOS from SAP Gateway-UI.

Introduction:

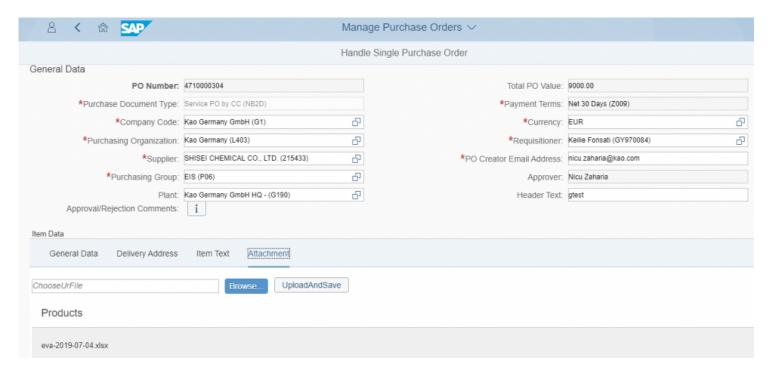This blog post will describe in detail how to Attach File to BO using GOS where

- User can upload the files
- User can see the list of attachments
- User can delete the attachment
- User can display content of attachment

Learn More: **SAP ABAP 7.4 Certification Preparation Guide (http://www.sapspot.com/sap-abap-7-4-certification/)**
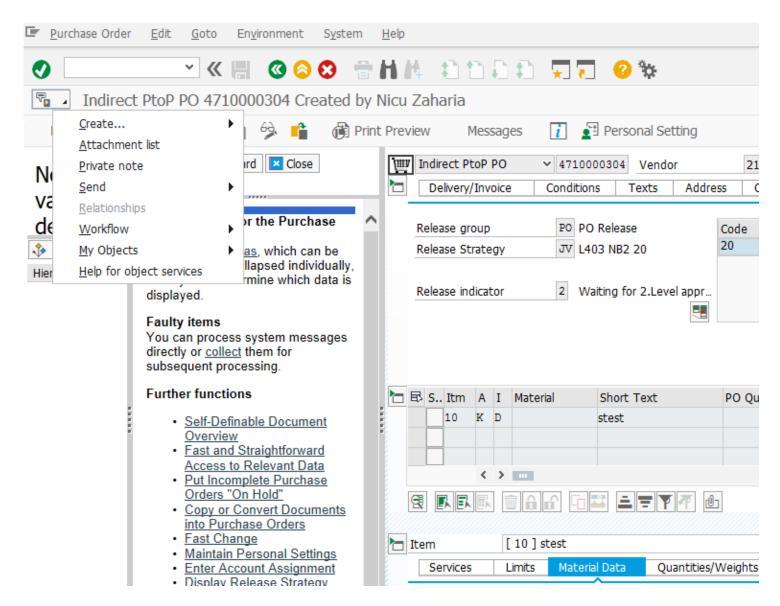
In this blog post, we will take the real-time example of Attach any File (PDF/DOC/JPG/XLS..) to object in ME23N from SAP Gateway.
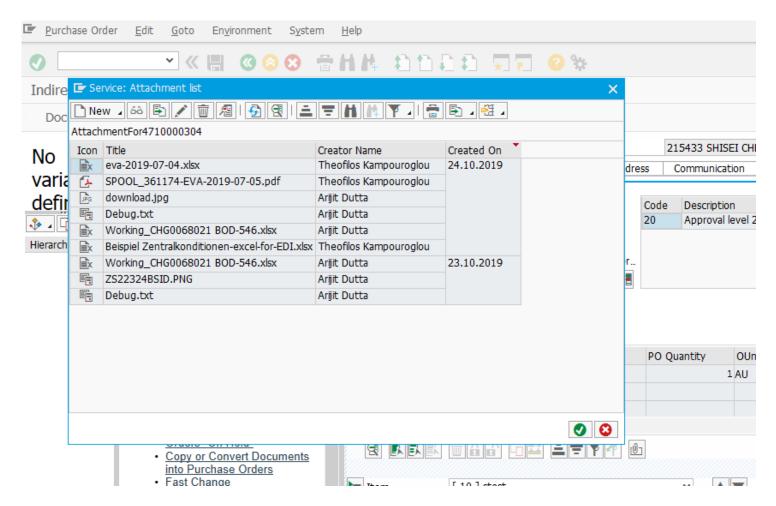
Requirement:

Create/DisplayContent/Display List of Attachments/Delete Attachments in PO (ME23N) using Net Weaver Gateway
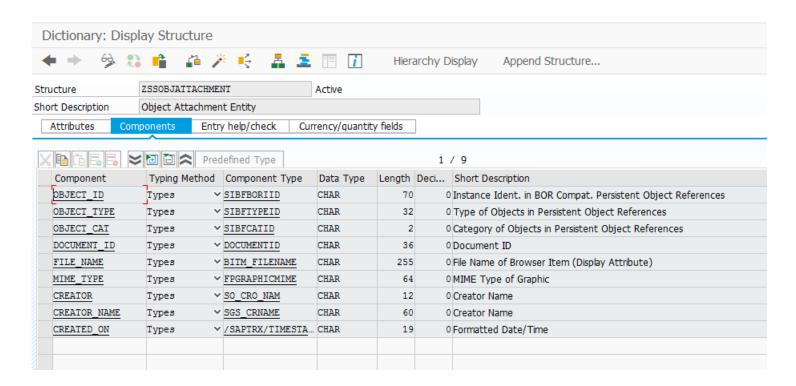
UI:

ME23N:

List of attachments for PO 4710000304

Steps to be followed:

Dictionary Structure

An attachment is identified with a composite key of Object ID, Object Type, Object Category and Document ID.

Dictionary: Display Structure

Hierarchy Display   Append Structure...

Structure	ZSSOBJATTACHMENT	Active
Short Description	Object Attachment Entity	

Attributes | Components | Entry help/check | Currency/quantity fields

Predefined Type   1 / 9

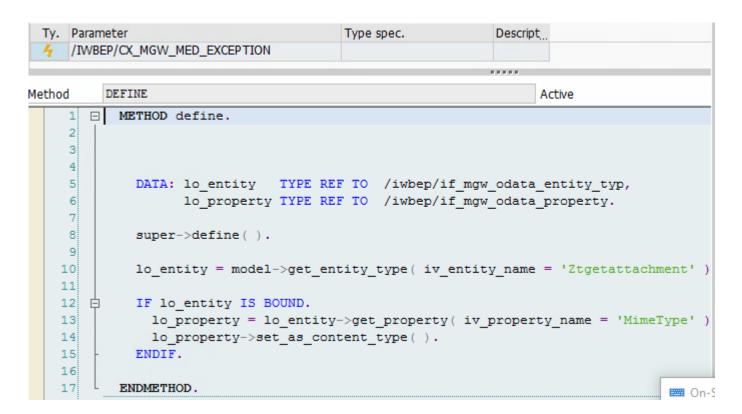| Component	Typing Method	Component Type	Data Type	Length	Deci...	Short Description
OBJECT_ID	Types	SIBFBORIID	CHAR	70	0	Instance Ident. in BOR Compat. Persistent Object References
OBJECT_TYPE	Types	SIBFTYPEID	CHAR	32	0	Type of Objects in Persistent Object References
OBJECT_CAT	Types	SIBFCATID	CHAR	2	0	Category of Objects in Persistent Object References
DOCUMENT_ID	Types	DOCUMENTID	CHAR	36	0	Document ID
FILE_NAME	Types	BITM_FILENAME	CHAR	255	0	File Name of Browser Item (Display Attribute)
MIME_TYPE	Types	FPGRAPHICMIME	CHAR	64	0	MIME Type of Graphic
CREATOR	Types	SO_CRO_NAM	CHAR	12	0	Creator Name
CREATOR_NAME	Types	SGS_CRNAME	CHAR	60	0	Creator Name
CREATED_ON	Types	/SAPTRX/TIMESTA...	CHAR	19	0	Formatted Date/Time

Meta Data

The entity for attachments is to be marked as 'Media Type'

The model provider extension class should redefine DEFINE method to handle this Media Type entity.

Ty.	Parameter	Type spec.	Descript...
⚡	/IWBEP/CX_MGW_MED_EXCEPTION		

Method   DEFINE   Active

```abap
 1 □  METHOD define.
 2
 3
 4
 5        DATA: lo_entity   TYPE REF TO  /iwbep/if_mgw_odata_entity_typ,
 6              lo_property TYPE REF TO  /iwbep/if_mgw_odata_property.
 7
 8        super->define( ).
 9
10        lo_entity = model->get_entity_type( iv_entity_name = 'Ztgetattachment' )
11
12 □      IF lo_entity IS BOUND.
13          lo_property = lo_entity->get_property( iv_property_name = 'MimeType' )
14          lo_property->set_as_content_type( ).
15        ENDIF.
16
17     ENDMETHOD.
```

⌨ On-S

Data Provider

The four main methods GET_ENTITYSET (attachment list), GET_STREAM, CREATE_STREAM and DELETE_STREAM are implemented in a utility class which is expected to be reused from each service requiring object services attachments.

GET_ENTITYSET

This method would retrieve the list of attachments for a business object.

```abap
ABAP code:

METHOD ztgetattachments_get_entityset.
*—————————————-
* Data declaration
*—————————————-

DATA object_id TYPE zif_zodata_get_po_attachment=>sibfboriid.
DATA object_type TYPE zif_zodata_get_po_attachment=>sibftypeid.
DATA object_cat TYPE zif_zodata_get_po_attachment=>sibfcatid.
DATA document_id TYPE zif_zodata_get_po_attachment=>documentid.
DATA ztgetattachment TYPE zif_zodata_get_po_attachment=>ztgetattachment.
DATA ls_ztgetattachment TYPE LINE OF zif_zodata_get_po_attachment=>ztgetattachment.
DATA lv_rfc_name TYPE tfdir-funcname.
DATA lv_destination TYPE rfcdest.
DATA lv_subrc TYPE syst-subrc.
DATA lv_exc_msg TYPE /iwbep/mgw_bop_rfc_excep_text.
DATA lx_root TYPE REF TO cx_root.
DATA lo_filter TYPE REF TO /iwbep/if_mgw_req_filter.
DATA lt_filter_select_options TYPE /iwbep/t_mgw_select_option.
DATA lv_filter_str TYPE string.
DATA ls_paging TYPE /iwbep/s_mgw_paging.
DATA ls_converted_keys LIKE LINE OF et_entityset.
DATA ls_filter TYPE /iwbep/s_mgw_select_option.
DATA ls_filter_range TYPE /iwbep/s_cod_select_option.
DATA lr_object_id LIKE RANGE OF ls_converted_keys-object_id.
DATA ls_object_id LIKE LINE OF lr_object_id.
DATA lr_object_type LIKE RANGE OF ls_converted_keys-object_type.
DATA ls_object_type LIKE LINE OF lr_object_type.
DATA lr_object_cat LIKE RANGE OF ls_converted_keys-object_cat.
DATA ls_object_cat LIKE LINE OF lr_object_cat.
DATA lr_document_id LIKE RANGE OF ls_converted_keys-document_id.
DATA ls_document_id LIKE LINE OF lr_document_id.
DATA lo_dp_facade TYPE REF TO /iwbep/if_mgw_dp_facade.
DATA ls_gw_ztgetattachment LIKE LINE OF et_entityset.
DATA lv_skip TYPE int4.
DATA lv_top TYPE int4.
*—————————————-
* Map the runtime request to the RFC – Only mapped attributes
*—————————————-
```

```abap
* Get all input information from the technical request context object
* Since DPC works with internal property names and runtime API interface holds external property names
* the process needs to get the all needed input information from the technical request context object
* Get filter or select option information
lo_filter = io_tech_request_context->get_filter( ).
lt_filter_select_options = lo_filter->get_filter_select_options( ).
lv_filter_str = lo_filter->get_filter_string( ).

* * Check if the supplied filter is supported by standard gateway runtime process
IF lv_filter_str IS NOT INITIAL
AND lt_filter_select_options IS INITIAL.
” If the string of the Filter System Query Option is not automatically converted into
” filter option table (lt_filter_select_options), then the filtering combination is not supported
” Log message in the application log
me->/iwbep/if_sb_dpc_comm_services~log_message(
EXPORTING
iv_msg_type = ‘E’
iv_msg_id = ‘/IWBEP/MC_SB_DPC_ADM’
iv_msg_number = 025 ).
” Raise Exception
RAISE EXCEPTION TYPE /iwbep/cx_mgw_tech_exception
EXPORTING
textid = /iwbep/cx_mgw_tech_exception=>internal_error.
ENDIF.

* * Get key table information
io_tech_request_context->get_converted_source_keys(
IMPORTING
es_key_values = ls_converted_keys ).

ls_paging-top = io_tech_request_context->get_top( ).
ls_paging-skip = io_tech_request_context->get_skip( ).

* * Maps filter table lines to function module parameters
LOOP AT lt_filter_select_options INTO ls_filter.

CASE ls_filter-property.
WHEN ‘OBJECT_ID’.
lo_filter->convert_select_option(
EXPORTING
is_select_option = ls_filter
```

```abap
        IMPORTING
        et_select_option = lr_object_id ).

        READ TABLE lr_object_id INTO ls_object_id INDEX 1.
        IF sy-subrc = 0.
        object_id = ls_object_id-low.
        ENDIF.
        WHEN 'OBJECT_TYPE'.
        lo_filter->convert_select_option(
        EXPORTING
        is_select_option = ls_filter
        IMPORTING
        et_select_option = lr_object_type ).

        READ TABLE lr_object_type INTO ls_object_type INDEX 1.
        IF sy-subrc = 0.
        object_type = ls_object_type-low.
        ENDIF.
        WHEN 'OBJECT_CAT'.
        lo_filter->convert_select_option(
        EXPORTING
        is_select_option = ls_filter
        IMPORTING
        et_select_option = lr_object_cat ).

        READ TABLE lr_object_cat INTO ls_object_cat INDEX 1.
        IF sy-subrc = 0.
        object_cat = ls_object_cat-low.
        ENDIF.
        WHEN 'DOCUMENT_ID'.
        lo_filter->convert_select_option(
        EXPORTING
        is_select_option = ls_filter
        IMPORTING
        et_select_option = lr_document_id ).

        READ TABLE lr_document_id INTO ls_document_id INDEX 1.
        IF sy-subrc = 0.
        document_id = ls_document_id-low.
        ENDIF.
```

```abap
WHEN OTHERS.
” Log message in the application log
me->/iwbep/if_sb_dpc_comm_services~log_message(
EXPORTING
iv_msg_type = ‘E’
iv_msg_id = ‘/IWBEP/MC_SB_DPC_ADM’
iv_msg_number = 020
iv_msg_v1 = ls_filter-property ).
” Raise Exception
RAISE EXCEPTION TYPE /iwbep/cx_mgw_tech_exception
EXPORTING
textid = /iwbep/cx_mgw_tech_exception=>internal_error.
ENDCASE.

ENDLOOP.
* Get RFC destination
lo_dp_facade = /iwbep/if_mgw_conv_srv_runtime~get_dp_facade( ).
lv_destination = /iwbep/cl_sb_gen_dpc_rt_util=>get_rfc_destination( io_dp_facade = lo_dp_facade ).
*———————————-
* Call RFC function module
*———————————-
lv_rfc_name = ‘ZODATA_GET_PO_ATTACHMENT’.

IF lv_destination IS INITIAL OR lv_destination EQ ‘NONE’.

TRY.
CALL FUNCTION lv_rfc_name
EXPORTING
object_id = object_id
object_type = object_type
object_cat = object_cat
document_id = document_id
TABLES
ztgetattachment = ztgetattachment
EXCEPTIONS
system_failure = 1000 message lv_exc_msg
OTHERS = 1002.

lv_subrc = sy-subrc.
*in case of co-deployment the exception is raised and needs to be caught
CATCH cx_root INTO lx_root.
```

```abap
          lv_subrc = 1001.
          lv_exc_msg = lx_root->if_message~get_text( ).
        ENDTRY.

      ELSE.

        CALL FUNCTION lv_rfc_name DESTINATION lv_destination
          EXPORTING
            object_id = object_id
            object_type = object_type
            object_cat = object_cat
            document_id = document_id
          IMPORTING
            ztgetattachment = ztgetattachment
          EXCEPTIONS
            system_failure = 1000 MESSAGE lv_exc_msg
            communication_failure = 1001 MESSAGE lv_exc_msg
            OTHERS = 1002.

        lv_subrc = sy-subrc.

      ENDIF.

      LOOP AT ztgetattachment INTO ls_ztgetattachment.
        ls_gw_ztgetattachment-object_id = ls_ztgetattachment-object_id.
        ls_gw_ztgetattachment-object_type = ls_ztgetattachment-object_type.
        ls_gw_ztgetattachment-object_cat = ls_ztgetattachment-object_cat.
        ls_gw_ztgetattachment-document_id = ls_ztgetattachment-document_id.
        ls_gw_ztgetattachment-file_name = ls_ztgetattachment-file_name.
        ls_gw_ztgetattachment-mime_type = ls_ztgetattachment-mime_type.
        ls_gw_ztgetattachment-creator = ls_ztgetattachment-creator.
        ls_gw_ztgetattachment-creator_name = ls_ztgetattachment-creator_name.
        ls_gw_ztgetattachment-created_on = ls_ztgetattachment-created_on.
        APPEND ls_gw_ztgetattachment TO et_entityset.
        CLEAR ls_gw_ztgetattachment.
      ENDLOOP.
    ENDMETHOD.
```

GET_STREAM

This method will display the content of attachment.

```abap
Abap Code:

METHOD /iwbep/if_mgw_appl_srv_runtime~get_stream.

TYPES: BEGIN OF ty_stream,
mime_type TYPE string,
value TYPE xstringval,
END OF ty_stream.

DATA: ls_key_tab LIKE LINE OF it_key_tab,
lv_doc_id TYPE sofolenti1-doc_id.

DATA: lt_obj_header TYPE STANDARD TABLE OF solisti1,
ls_objhead TYPE solisti1,
lt_obj_cont TYPE STANDARD TABLE OF solisti1,
lt_attachment TYPE STANDARD TABLE OF soattlsti1,
lt_cont_hex TYPE STANDARD TABLE OF solix,
ls_doc_data TYPE sofolenti1.

* DATA it_content TYPE STANDARD TABLE OF soli.
DATA lv_xstring TYPE xstring.
DATA lv_mimetype TYPE mimetypes-type.
DATA: ls_stream TYPE ty_stream.
DATA lv_file TYPE char255.
DATA ls_lheader TYPE ihttpnvp.

READ TABLE it_key_tab INTO ls_key_tab WITH KEY name = 'DocumentId'.
IF sy-subrc EQ 0.
lv_doc_id = ls_key_tab-value.
ENDIF.

* FM to Read Link/Attachment.
CONCATENATE '&SO_FILENAME=' lv_file INTO ls_objhead-line.
* ls_objhead-line = '&SO_FILENAME='.
APPEND ls_objhead TO lt_obj_header.

ls_objhead-line = '&SO_FORMAT=BIN'.
APPEND ls_objhead TO lt_obj_header.
CALL FUNCTION 'SO_DOCUMENT_READ_API1'
EXPORTING
```

```abap
    document_id = lv_doc_id
  IMPORTING
    document_data = ls_doc_data
  TABLES
    object_header = lt_obj_header
    object_content = lt_obj_cont
    attachment_list = lt_attachment
    contents_hex = lt_cont_hex
  EXCEPTIONS
    document_id_not_exist = 1
    operation_no_authorization = 2
    x_error = 3
    OTHERS = 4.

TRY.
  CALL METHOD cl_bcs_convert=>xtab_to_xstring
    EXPORTING
      it_xtab = lt_cont_hex
    RECEIVING
      rv_xstring = lv_xstring.
  CATCH cx_bcs .
ENDTRY.

CALL FUNCTION 'SDOK_MIMETYPE_GET'
  EXPORTING
    extension = ls_doc_data-obj_type
  IMPORTING
    mimetype = lv_mimetype.

*******************
**************

ls_stream-value = lv_xstring.
* ls_stream-value = bindata.
ls_stream-mime_type = lv_mimetype.

******
ls_lheader-name = 'Content-Disposition'.
ls_lheader-value = |inline; filename="{ ls_doc_data-obj_descr }"|.
set_header( is_header = ls_lheader ).
*******
```

```
copy_data_to_ref( EXPORTING is_data = ls_stream
CHANGING cr_data = er_stream ).

ENDMETHOD.
```

CREATE_STREAM

This method will upload the any types of files attachment.

Abap Code:

```
METHOD /iwbep/if_mgw_appl_srv_runtime~create_stream.
*——————————————————-
* Constants
*——————————————————-
CONSTANTS : c_bus2012 TYPE swo_objtyp VALUE 'BUS2012', " bus NUMBER FOR PO
c_ext(3) TYPE c VALUE 'EXT',
c_atta(4) TYPE c VALUE 'ATTA',
c_b(1) TYPE c VALUE 'B',
c_x(1) TYPE c VALUE 'X',
c_o(1) TYPE c VALUE 'O'.


*——————————————————-
* Data declaration
*——————————————————-

DATA: it_content TYPE solix_tab, " content OF FILE STORAGE
it_objhead TYPE STANDARD TABLE OF soli,
ls_objhead TYPE soli,
wa_folmem_k TYPE sofmk, "folder content DATA
wa_note TYPE borident, " bor OBJECT IDENTIFIER
wa_object TYPE borident,
wa_obj_id TYPE soodk, "DEFINITION OF an OBJECT (key part)
wa_fol_id TYPE soodk,
wa_obj_data TYPE sood1, " OBJECT DEFINITION AND CHANGE ATTRIBUTES
lv_ep_note TYPE borident-objkey, "bor OBJECT KEY
lv_extension TYPE c LENGTH 4, "FILE EXTENSION ONLY
lv_so_num TYPE ebeln, "PO NUMBER
lv_file_des TYPE so_obj_des, " file name
lv_tmp_fn TYPE string.
```

```abap
DATA object_id TYPE zif_zodata_get_po_attachment=>sibfboriid.
DATA object_type TYPE zif_zodata_get_po_attachment=>sibftypeid.
DATA object_cat TYPE zif_zodata_get_po_attachment=>sibfcatid.
DATA document_id TYPE zif_zodata_get_po_attachment=>documentid.
DATA ls_converted_keys LIKE er_entity.
DATA lv_source_entity_set_name TYPE string.
DATA lo_facade TYPE REF TO /iwbep/if_mgw_dp_facade.
DATA lt_request_header TYPE tihttpnvp.
DATA ls_request_header TYPE ihttpnvp.
DATA: ls_key_tab LIKE LINE OF it_key_tab.
TYPES: BEGIN OF ty_filemetadata,
filename TYPE string,
mimetype TYPE string,
END OF ty_filemetadata.

DATA: ls_filemetadata TYPE ty_filemetadata.

DATA lv_extensions TYPE char10.
DATA lv_file TYPE char255.
DATA lv_po_num TYPE ebeln.

*/Refresh data
REFRESH: it_content[], it_objhead[].

*/Get folder id
CALL FUNCTION 'SO_FOLDER_ROOT_ID_GET'
EXPORTING
region = c_b
IMPORTING
folder_id = wa_fol_id
EXCEPTIONS
communication_failure = 1
owner_not_exist = 2
system_failure = 3
x_error = 4
OTHERS = 5.
***********************************************
*To get the extension from the file name
IF iv_slug IS NOT INITIAL.
SPLIT iv_slug AT '/' INTO lv_po_num lv_file.
CALL FUNCTION 'TRINT_FILE_GET_EXTENSION'
```

```abap
        EXPORTING
        filename = lv_file
        uppercase = 'X'
        IMPORTING
        extension = lv_extensions.
        ENDIF.

*****************************************************88
* To convert the file content to binary
DATA: lv_filesize TYPE i,
lv_base64 TYPE string, "Base64
lv_string TYPE string, "Base64
lv_binary TYPE xstring.

CALL FUNCTION 'SCMS_BASE64_ENCODE_STR'
EXPORTING
input = is_media_resource-value
IMPORTING
output = lv_base64.
* *Decode Base64 String to String
CALL METHOD cl_http_utility=>if_http_utility~decode_base64
EXPORTING
encoded = lv_base64
RECEIVING
decoded = lv_string.

CALL FUNCTION 'SSFC_BASE64_DECODE'
EXPORTING
b64data = lv_string
* B64LENG =
* B_CHECK =
IMPORTING
bindata = lv_binary
EXCEPTIONS
ssf_krn_error = 1
ssf_krn_noop = 2
ssf_krn_nomemory = 3
ssf_krn_opinv = 4
ssf_krn_input_data_error = 5
ssf_krn_invalid_par = 6
ssf_krn_invalid_parlen = 7
```

```abap
    OTHERS = 8.

    CALL FUNCTION 'SCMS_XSTRING_TO_BINARY'
    EXPORTING
    buffer = lv_binary "is_media_resource-value "xstring
    append_to_table = c_x
    TABLES
    binary_tab = it_content. "binary.

    **************************************************
    * to convert binary (SOLIXTAB) to SOLITAB
    DATA lt_data TYPE soli_tab.
    CALL FUNCTION 'SO_SOLIXTAB_TO_SOLITAB'
    EXPORTING
    ip_solixtab = it_content
    IMPORTING
    ep_solitab = lt_data.
    *****************************************************
    * To create the document object
    */ Assigning Valuse to the Standard Strucuture Fields
    wa_object-objkey = lv_po_num. " PO number
    wa_object-objtype = c_bus2012. " bus number
    *_
    wa_obj_data-objsns = c_o. " sensitivity of object (o-standard)
    wa_obj_data-objla = sy-langu. " language
    wa_obj_data-objdes = lv_file. "iv_slug. " slug value - description
    wa_obj_data-file_ext = lv_extensions. " file extension
    * wa_obj_data-objlen = lines( it_content ) * 255.
    wa_obj_data-objlen = lines( lt_data ) * 255.

    CONCATENATE '&SO_FILENAME=' lv_file INTO ls_objhead-line.
    APPEND ls_objhead TO it_objhead.

    ls_objhead-line = '&SO_FORMAT=BIN'.
    APPEND ls_objhead TO it_objhead.

    */ Insert data
    CALL FUNCTION 'SO_OBJECT_INSERT'
    EXPORTING
    folder_id = wa_fol_id
    object_type = c_ext
```

```
object_hd_change = wa_obj_data
IMPORTING
object_id = wa_obj_id
TABLES
objhead = it_objhead
objcont = lt_data
EXCEPTIONS
active_user_not_exist = 1
communication_failure = 2
component_not_available = 3
dl_name_exist = 4
folder_not_exist = 5
folder_no_authorization = 6
object_type_not_exist = 7
operation_no_authorization = 8
owner_not_exist = 9
parameter_error = 10
substitute_not_active = 11
substitute_not_defined = 12
system_failure = 13
x_error = 14
OTHERS = 15.

*************************************************
* To link the document attachment and business object
IF sy-subrc = 0 AND wa_object-objkey IS NOT INITIAL.
wa_folmem_k-foltp = wa_fol_id-objtp.
wa_folmem_k-folyr = wa_fol_id-objyr.
wa_folmem_k-folno = wa_fol_id-objno.
*/Please note: wa_fol_id and wa_obj_id are different work areas
wa_folmem_k-doctp = wa_obj_id-objtp.
wa_folmem_k-docyr = wa_obj_id-objyr.
wa_folmem_k-docno = wa_obj_id-objno.
lv_ep_note = wa_folmem_k.
wa_note-objtype = 'MESSAGE'.
wa_note-objkey = lv_ep_note.

*/Link it
CALL FUNCTION 'BINARY_RELATION_CREATE_COMMIT'
EXPORTING
obj_rolea = wa_object
```

```abap
obj_roleb = wa_note
relationtype = c_atta
EXCEPTIONS
no_model = 1
internal_error = 2
unknown = 3
OTHERS = 4.
IF sy-subrc EQ 0.
* Commit it
COMMIT WORK.
WRITE:/ 'Attached Successfully'.
ENDIF.

ENDIF.

* Fill the export parameter er_entity accordingly
* ls_filemetadata-filename = iv_slug.
ls_filemetadata-filename = lv_file.
ls_filemetadata-mimetype = is_media_resource-mime_type.

copy_data_to_ref(
EXPORTING
is_data = ls_filemetadata
CHANGING
cr_data = er_entity ).
ENDMETHOD.
```

DELETE_STREAM

This method will delete the attachment

Abap Code:

```abap
METHOD /iwbep/if_mgw_appl_srv_runtime~delete_stream.

DATA i_objectd TYPE borident.
DATA: ls_key_tab LIKE LINE OF it_key_tab,
lv_doc_id TYPE borident-objkey. "sofolenti1-doc_id.
DATA: i_borident TYPE borident,
i_attsrv TYPE REF TO cl_gos_document_service.
```

```
i_objectd-objtype = 'BUS2012'.

READ TABLE it_key_tab INTO ls_key_tab WITH KEY name = 'ObjectId'.
IF sy-subrc EQ 0.
i_objectd-objkey = ls_key_tab-value.
ENDIF.

READ TABLE it_key_tab INTO ls_key_tab WITH KEY name = 'DocumentId'.
IF sy-subrc EQ 0.
lv_doc_id = ls_key_tab-value.
ENDIF.

CREATE OBJECT i_attsrv.
CALL METHOD i_attsrv->delete_attachment
EXPORTING
is_object = i_objectd
ip_attachment = lv_doc_id.
COMMIT WORK.

ENDMETHOD.
```

SAP ABAP (http://www.sapspot.com/category/erp/sap-abap/), SAP Fiori (http://www.sapspot.com/category/sap-fiori/) NW ABAP Gateway (OData) (http://www.sapspot.com/tag/nw-abap-gateway-odata/), sap fiori (http://www.sapspot.com/tag/sap-fiori/). permalink (http://www.sapspot.com/attachments-attach-any-file-pdf-doc-jpg-xls-txt-to-bo-using-gos-from-sap-gateway-ui/).

Leave a Reply

Your email address will not be published. Required fields are marked *

Comment

Name *

Email *

Website

Post Comment

Search

Search...   🔍

🖼 (http://bit.ly/erpprep)

Ads

^

POPULAR   RECENT   💬

Basics of SAP HANA (http://www.sapspot.com/basics-sap-hana/)

Jan 9, 2017

Seamless Big Data tiering with HANA, Hadoop and Vora…with a little help from DLM – Part 1 (http://www.sapspot.com/seamless-big-data-tiering-hana-hadoop-vorawith-little-help-dlm-part-1/)

Jun 24, 2016

Business Partner Setup in SAP S/4HANA (http://www.sapspot.com/business-partner-setup-in-sap-s-4hana/)

Jul 2, 2018

Inter-company STO with SD Delivery, Billing & LIV (http://www.sapspot.com/inter-company-sto-sd-delivery-billing-liv/)

Aug 19, 2017

Employee Central – SAP ERP HCM prepackaged Integration: how to change the standard field mapping (http://www.sapspot.com/employee-central-sap-erp-hcm-prepackaged-integration-change-standard-field-mapping/)

Jan 22, 2018

Archives

˄

📅 March 2020 (http://www.sapspot.com/2020/03/) (5)

📅 February 2020 (http://www.sapspot.com/2020/02/) (25)

📅 January 2020 (http://www.sapspot.com/2020/01/) (23)

📅 December 2019 (http://www.sapspot.com/2019/12/) (30)

📅 November 2019 (http://www.sapspot.com/2019/11/) (22)

📅 October 2019 (http://www.sapspot.com/2019/10/) (26)

📅 September 2019 (http://www.sapspot.com/2019/09/) (21)

📅 August 2019 (http://www.sapspot.com/2019/08/) (13)

📅 July 2019 (http://www.sapspot.com/2019/07/) (18)

📅 June 2019 (http://www.sapspot.com/2019/06/) (22)

📅 May 2019 (http://www.sapspot.com/2019/05/) (24)

📅 April 2019 (http://www.sapspot.com/2019/04/) (17)

📅 March 2019 (http://www.sapspot.com/2019/03/) (23)

📅 February 2019 (http://www.sapspot.com/2019/02/) (6)

📅 January 2019 (http://www.sapspot.com/2019/01/) (18)

📅 December 2018 (http://www.sapspot.com/2018/12/) (15)

📅 November 2018 (http://www.sapspot.com/2018/11/) (15)

Contact Us (http://www.sapspot.com/contact-us/)   SAP Certification (http://www.sapspot.com/sap-certification/)