

Customization Technology

This document describes the customization of BX Mobile Warehouse's screens through examples. The customization examples consist of 3 major steps:

- Configure and run the BX Mobile Warehouse client program and check the screen to be customized.
- Create the SAP user query needed.
- Run the BX Mobile Warehouse client program again to check the results of the customization (If needed, user query modifications, fine tuning can be done here).

You will see that most of the screens of BX Mobile Warehouse are customizable.

The Customization can be: Validation (eg. checking if the entered values are correct), this happens on validate or button events. Setting some field data (eg. filling the Quantity on a screen). Other customization options are adding some fields on the screen, but that will be discussed in another document if needed.

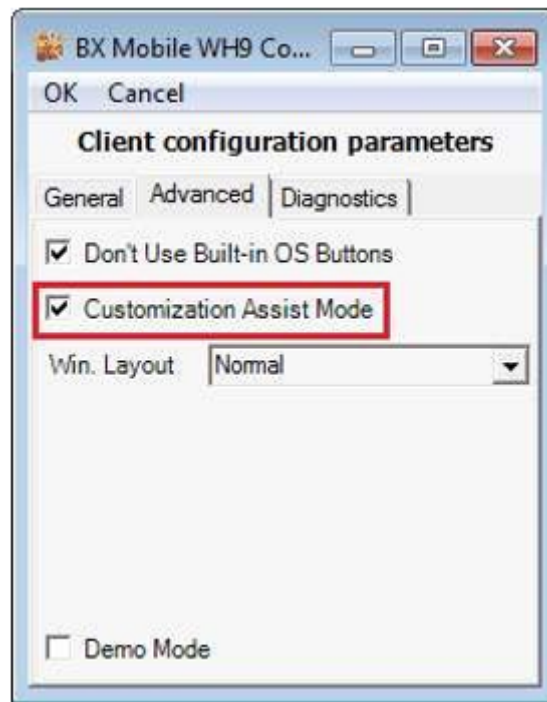
Customization uses SAP Business One's usual User Queries, which have to be named specifically: eg. 'BXMobileWH9_GoodsReceiptPOScreen_Load': this is a user query that runs when the 'GR PO' screen has been loaded and displayed. The query names, and parameters can be found with the help of the 'Customization Assist' mode in the Warehouse client. The query result column names indicate which fields to set.

1. Customization Steps

1.1. Configure and run warehouse client

To customize the Complete Job screen, you should change the BX Mobile Warehouse client program configuration.

Start the BX Mobile Warehouse Configuration, and go to the Appearance tab to make enable the customization assistance.



This mode is only needed when you want to have information about customization possibilities on the screen. It is not needed during normal Warehouse client operations .

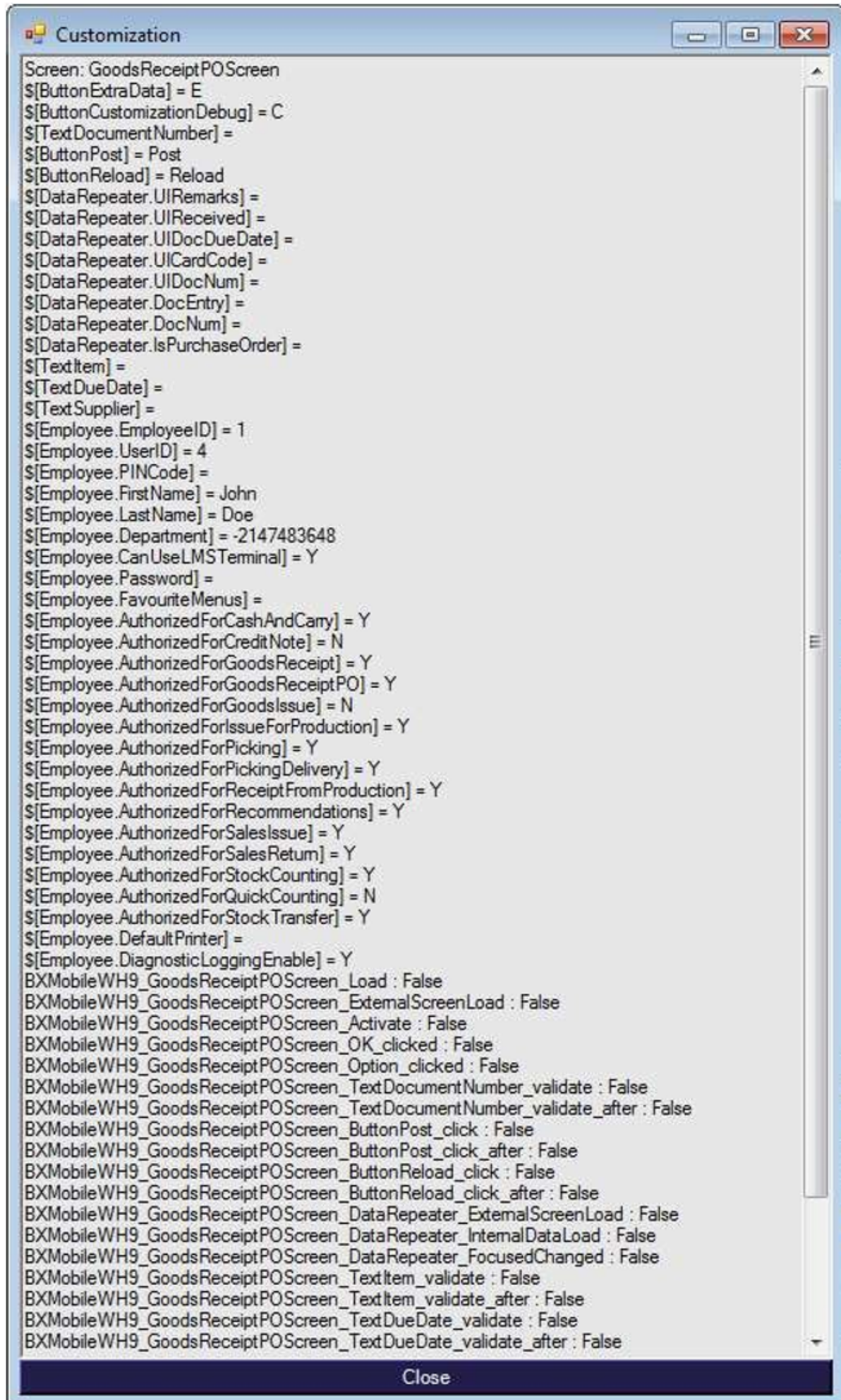
After that the Assist Mode has been activated, you can login in the BX Mobile Warehouse client program and select the form to be customized.

In the following example, we will work with GR/PO form.



To display the Customization screen, you have to press the Customize button (C) on the top of the

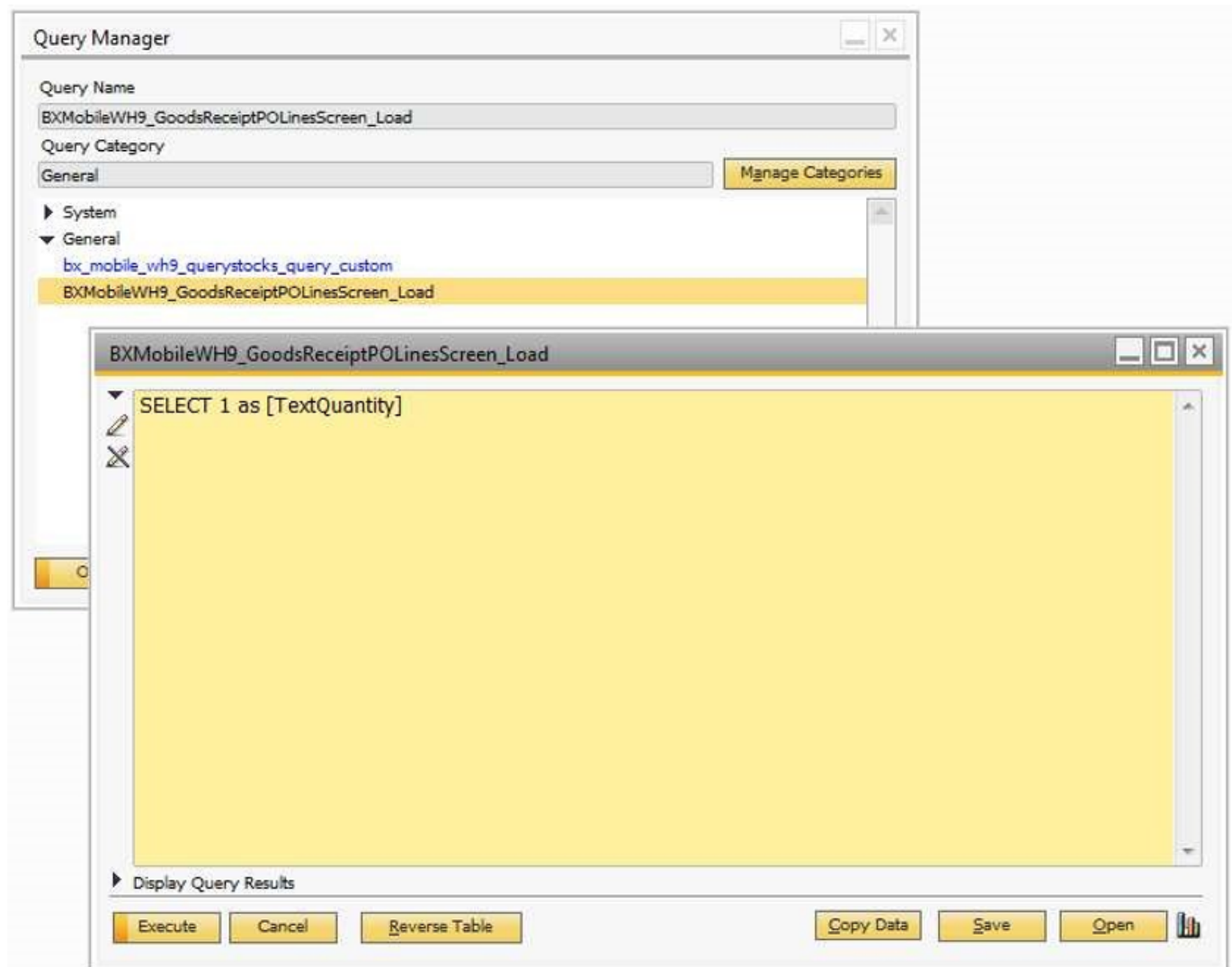
screen:



Here you can see the possible parameters for customization user queries, and the events that can be customized with user queries (event name = user query name).
In our example, we want to fill Quantity with default value: 1

1.2. Create the SAP user query needed

Open SAP Business One, Tools → Queries → Query Manager (or Query Generator).



Then, you have to create a user query named: *BXMobilWH9_GoodsReceiptPOLinesScreen_Load*

The query contents should be:

```
SELECT 1 as [TextQuantity]
```

This query runs when the “GR PO” > “Receive” screen loads. It selects the quantity 1 into the Quantity field.

1.3. Run the WH9 client program again to check the results of the customization

To refresh the customization, we must restart the Warehouse (client). Every time a new query is created, you have to restart the program. If you modify a user query, it is not needed. So it's fast to fine tune the query.

Now let's open the WH9 and go to the "GR PO" > "Receive" screen



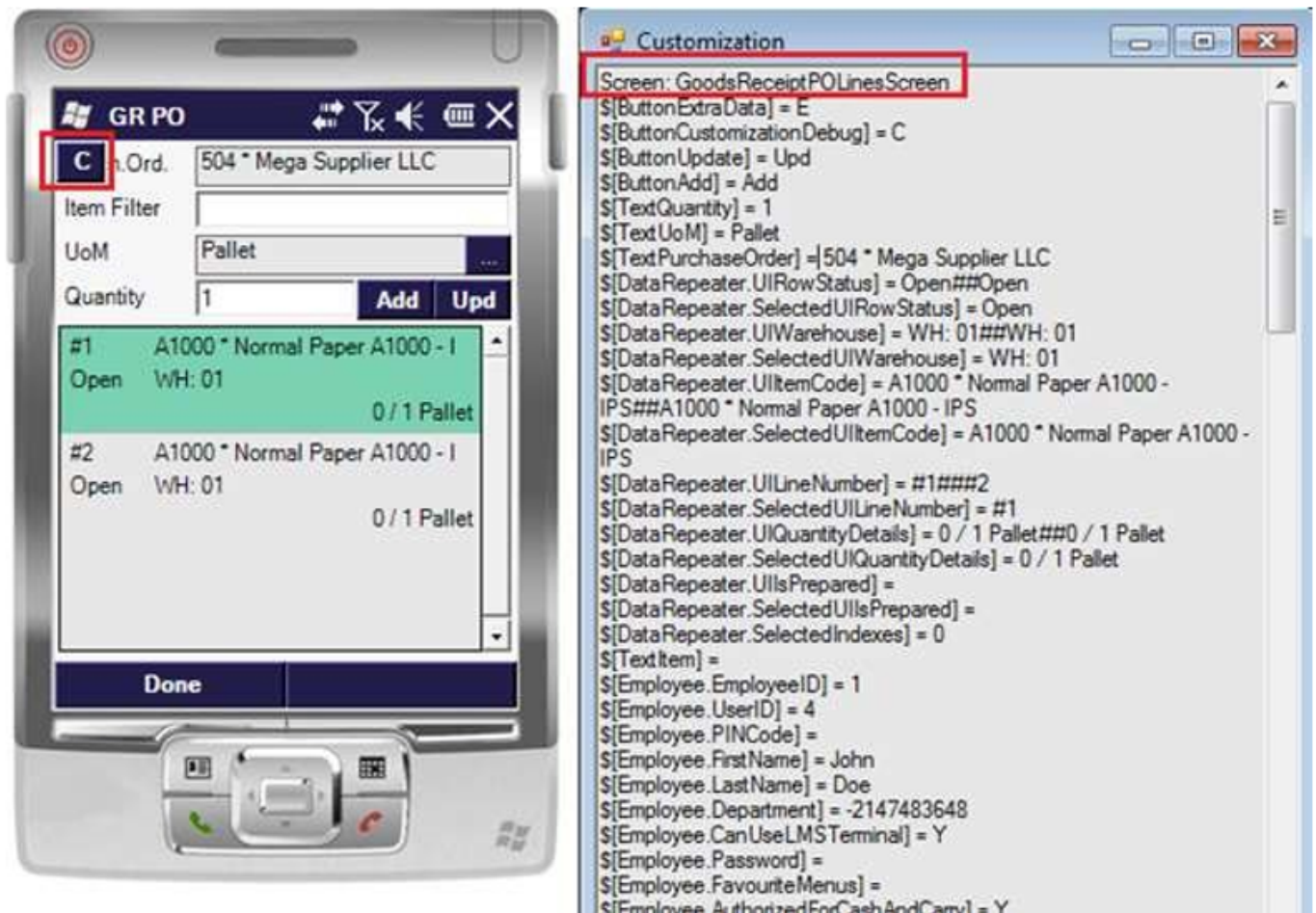
2. Customization Examples

2.1. Goods Receipt PO

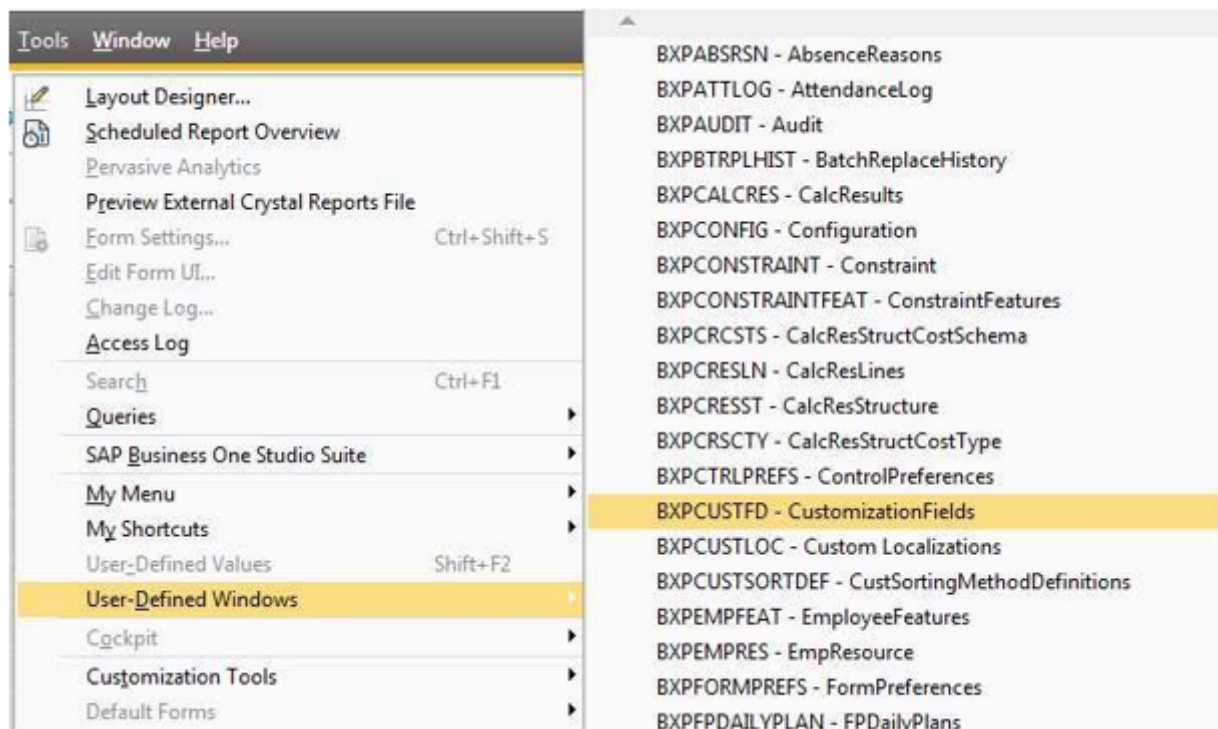
2.1.1. Set (user) fields in GR PO

The customization makes possible to set additional fields (SAP or user fields) in the Goods Receipt PO document from.

Go to GR PO Lines screen, and open Customization window. Here you can see the name of this screen: GoodsReceiptPOLinesScreen



To add new fields, open the CustomizationFields User-Defined window in SAP.

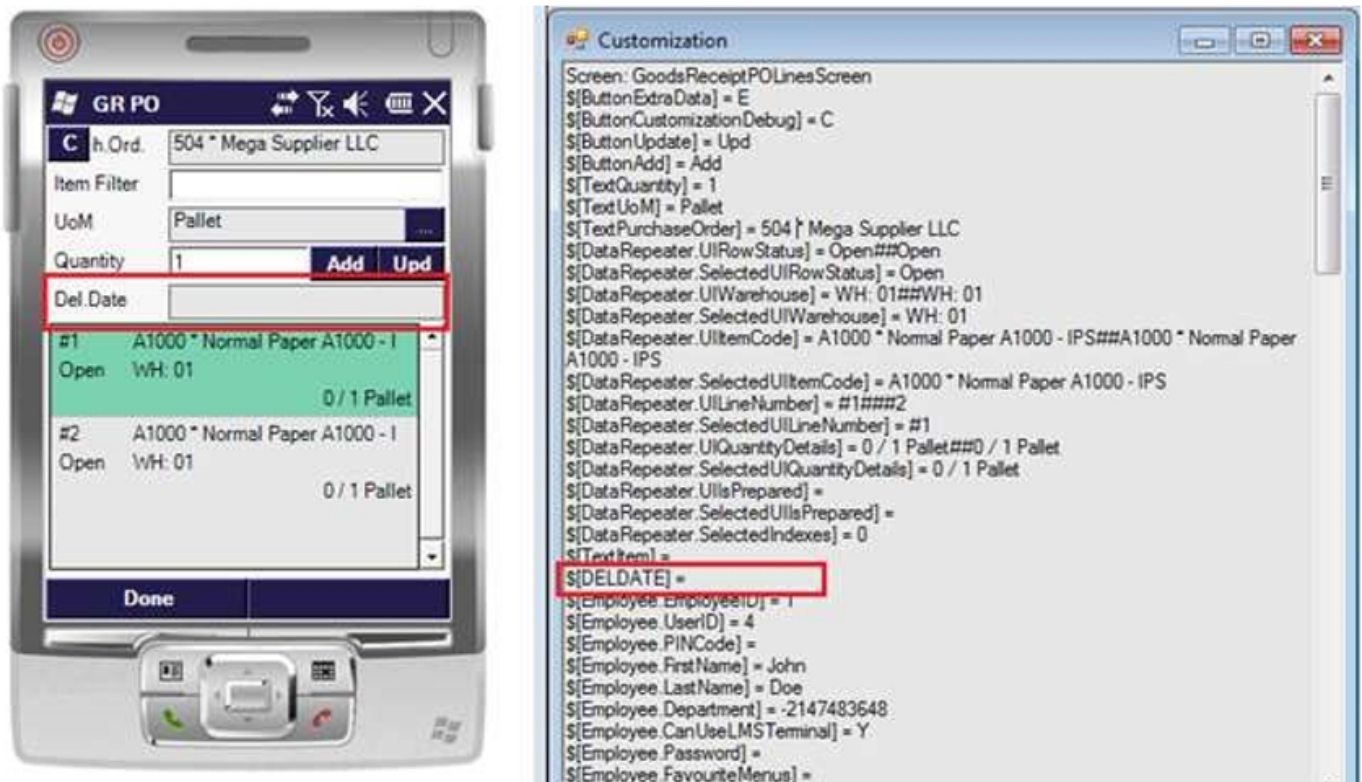


Use the recommended parameters as below:

Screen: GoodsReceiptPOLinesScreen this is the name of the screen (first line on Customization screen)
 Module: BXMobilWH9
 Label: displayed text
 Field Name: DELDATE you can use this name in custom queries.

CustomizationFields											
#	Code	Name	Field Display Type	Field Type	Folder Name	Label	Module	On External Form	Read Only	Visible	Last Modif.
1	1	1	None	String		DelDate	BXMobilWH9	No	Yes	Yes	

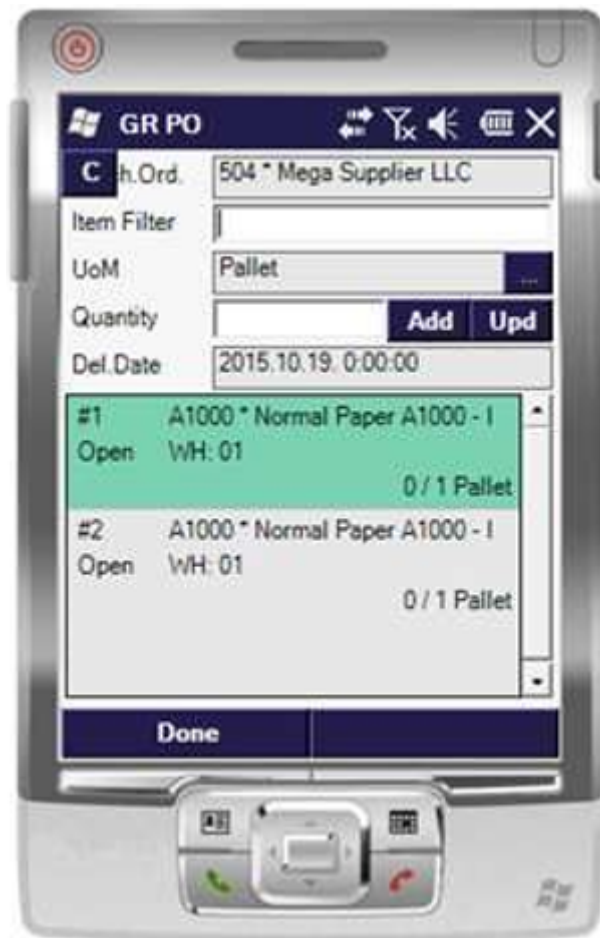
Now you are able to load data into this new field. To do it, you have to use the BXMobilWH9_GoodsReceiptPOLinesScreen_Load user query.



If you want to select the delivery date of the purchase order, you will need the purchase order number. You can check the field name on customization screen:
`$(TextPurchaseOrder) = 504 * Mega Supplier LLC`

You have to use `$(TextPurchaseOrder)` field and you have to split the string in the query. If you want to read the data from the field, then you have to use `$` character.

```
SELECT
    DocDueDate as [DELDATE]
FROM OPOR WHERE
    DocNum = SUBSTRING( $(TextPurchaseOrder), 0, CHARINDEX('*',
$(TextPurchaseOrder), 1) - 1)
```



2.1.2. Set Freight

If is possible to set the Freight costs/lines in the created Goods Receipt PO with user query: See : 3.2.1 for the example query.

In this page, you can play around and test anything.

UserQuery: bx_mobile_wh9_document_additionalexpenses (Category: BXMobilWH9)	
Parameters [%1] - employeeID [%2] - TerminalID [%3] - Head Code from Mobile Transaction table (@BXPLMSMOBTHD.Code) [%4] - grouped Head Codes (string, list), separated with # characters, only applicable for grouped Purchase Order/APReserve invoice → Goods Receipt PO	Results Zero, one or more rows for the Freight charges to be created. The result columns can have the names: * BO_LineTotal = the LineTotal field, amount of money (mandatory) * BO_ExpenseCode = Expense Code (integer) (mandatory)

2.1.3. Batch number generation

In the Goods Receipt PO process, the system can automatically generate batch numbers when entering the received quantities and bin locations. The batch number generation logic needs to be defined in a user query.

UserQuery: bx_mobile_wh9_get_new_batchnumber (Category: BXMobileWH9)	
Parameters [%1]: employee ID (int) [%2]: terminal ID (nvarchar) [%3]: base document type (int) [%4]: base document entry (int) [%5]: base document line (int) [%6]: item code (nvarchar)	Results The user query should return the batch number in a column called BXBATNUM

For example, the following query will generate a batch number combining the supplier code and the current date:

```
SELECT T0.[CardCode] + '-' + CONVERT(varchar, GETDATE(), 112) AS 'BXBATNUM'
FROM OPOR T0 WHERE T0.[DocEntry] = [%4]
```

2.1.4. Serial number generation

In the Goods Receipt PO process, the system can automatically generate serial numbers when entering the received stocks. The serial number generation logic needs to be defined in a user query.

UserQuery: bx_mobile_wh9_get_new_serialnumber (Category: BXMobileWH9)	
Parameters [%1]: employee ID (int) [%2]: terminal ID (nvarchar) [%3]: base document type (int) [%4]: base document entry (int) [%5]: base document line (int) [%6]: item code (nvarchar)	Results The user query should return the serial numbers (multiple rows possible) in a column called BXSERNUM

2.1.5. Expiration date at GRPO/Inventory

New fix fields like ExpirationDate and ManufactruingDate have been add to to BX Mobile Warehouse. If you are working with GS1-128 they will be included after upgrade to version 16.02.24062. They also can be add by customizing at CustomizationField table (Tools>UserDefined Windows>BXPCUSTFD) as it is shown below.

These fields can be used at BatchScreen or SerialScreen.

```
BATCH_EXPIRATION_DATE = "#ExpirationDate";
BATCH_MANUFACTURING_DATE = "#ManufacturingDate";
BATCH_ATTRIBUTE1 = "#Attribute1";
BATCH_ATTRIBUTE2 = "#Attribute2";
BATCH_DETAILS = "#Details";
```

```
SERIAL_EXPIRATION_DATE = "#SerialExpirationDate";
SERIAL_MANUFACTURING_DATE = "#SerialManufacturingDate";
SERIAL_ATTRIBUTE1 = "#SerialAttribute1";
SERIAL_ATTRIBUTE2 = "#SerialAttribute2";
SERIAL_DETAILS = "#SerialDetails";
```

Below you have an example for Batches

Field Display Type	Field Name	Field Type	Folder Name	Label	Language Code	Module
None	#ExpirationDate	String		MHD		BXMobileWH9
None	#ManufacturingDate	String		Herst.Datum		BXMobileWH9
None	#ExpirationDate	String		MHD		BXMobileWH9

Module	On External Form	Position Data	Read Only	Screen	Sort Order	Table N
BXMobileWH9	No		No	ReceiptFromProductionQuantitiesBatchScreen		
BXMobileWH9	No		No	ReceiptFromProductionQuantitiesBatchScreen		
BXMobileWH9	No		No	GoodsReceiptPOQuantitiesBatchScreen		

2.2. Sales Orders

2.2.1. Set Freight in Delivery document

By default the Freight in the created Delivery document is 0. It is possible to customize WH9 so freight lines are added with a custom freight calculation algorithm.

This algorithm receives a parameter which allows to query the items, quantities, base documents which will be used to create the new Delivery Document.

UserQuery: bx_mobile_wh9_document_additionalexpenses (Category: BXMobileWH9)	
Parameters [%1] - employeeID [%2] - TerminalID [%3] - Head Code from Mobile Transaction table (@BXPLMSMOBTHD.Code) [%4] - grouped Head Codes (string, list), separated with # characters, only applicable for grouped Purchase Order/APReserve invoice → Goods Receipt PO	Results Zero, one or more rows for the Freight charges to be created. The result columns can have the names: BO_LineTotal = the LineTotal field, amount of money (mandatory) BO_ExpenseCode = Expense Code (integer) (mandatory)

This example user query looks at the Mobile Transaction (lines) table, filters for Base Document Type = Sales Order, finds the base Sales Order Document DocEntry. It then retrieves the freight expense records related to the Sales Order (from table RDR3), and extracts Expense Code and Line Total, so it essentially copies all the freight charges from the Sales Order without any calculation.

A combined example query to copy expenses from Sales Orders AND Purchase Orders: Query name: bx_mobile_wh9_document_additionalexpenses

```
declare @salesOrderDocEntry int
SELECT @salesOrderDocEntry = U_BXPBsDcE FROM [@BXPLMSMOBTLN] WHERE U_BXPHdCd
=
[%3] AND U_BXPBsDcT = 17 AND U_BXPDocTy = 15
-- 15-delivery, 17-sales order
IF @salesOrderDocEntry > 0 BEGIN
SELECT ExpnsCode as B0_ExpenseCode, LineTotal as B0_LineTotal, 17 as
B0_BaseDocType, @salesOrderDocEntry as B0_BaseDocEntry, LineNum as
B0_BaseDocLine FROM RDR3 WHERE DocEntry = @salesOrderDocEntry AND Status <>
'C' END
declare @purchaseOrderDocEntry int
SELECT @purchaseOrderDocEntry = U_BXPBsDcE FROM [@BXPLMSMOBTLN] WHERE
U_BXPHdCd
= [%3] AND U_BXPBsDcT = 22 AND U_BXPDocTy = 20
-- 22-Purchase order, 20-Goods Receipt P0
IF @purchaseOrderDocEntry > 0 BEGIN
SELECT ExpnsCode as B0_ExpenseCode, LineTotal as B0_LineTotal, 22 as
B0_BaseDocType, @purchaseOrderDocEntry as B0_BaseDocEntry, LineNum as
B0_BaseDocLine FROM POR3 WHERE DocEntry = @purchaseOrderDocEntry AND Status
<>
'C'
END
```

2.3. Pick Lists

2.3.1. Set Freight in Delivery

See: 2.2.1 Set Freight in Delivery document

2.3.2. Set (user) fields in Delivery

It is possible to set the values of the Delivery Document which are created from the WH9 Pick List screen.

To set a field, you have to add a customization field to the Pick List Lines screen. The customization field name must be:

- BO_U_XXField1 to set Delivery Document U_XXField1 custom field
- BO_FieldName to set Delivery Document SAP internal field by API name. In this case, the field name must match the DI API name, see the DI API SDK documentation, Documents object.
- Example BO_JournalMemo

When pressing the Delivery button on the Pick List lines screen, these field values will be used in the newly created Delivery Document.

CustomizationFields													
#	Code	Name	Field Display Type	Field Name	Field Type	Folder Name	Label	Language Code	Module	On External Form	Position Data	Read Only	Screen
6	6	6	None	BO_U_Test1	String		Test 1		BXMobileWH9	No		No	PickingLinesScreen
7	7	7	None	BO_U_Test2	String		Test 2		BXMobileWH9	No		No	PickingLinesScreen

In the Customization Fields (BXPCUSTFD) user table, it is important that the Field Name is BO_U_* and it must match the UDF name (with a U_Prefix), eg. BO_U_Test1 → Test1 userdefined field. The Label can be anything.

2.3.3. Pick List screen - customize list

The Pick List screen allows to optionally override the original program logic that displays the pick lists to the employee. By default, all open pick lists are displayed that have not been started by anyone.

UserQuery: bx_mobile_wh9_picklists_query_custom (Category: BXMobileWH9)	
Parameters \$[AbsEntry] \$[CardCode] \$[EmployeeNo] - logged in employeeID \$[ItemCode] \$[PickDate] \$[WarehouseCode]	Results A table with multiple rows with a single column (integer) with PickList AbsEntry values. (OPKL.AbsEntry)

Example:

We want to use the Pick List's Picker (picker name) field to assign Pick Lists to Employees. We have to ensure that this field is in the 'FirstName LastName' or 'FirstName MiddleName LastName' format for

the user query to work.

The example user query name: bx_mobile_wh9_picklists_query_custom

```
IF $[AbsEntry] IS NOT NULL BEGIN
SELECT AbsEntry FROM OPKL WHERE AbsEntry = $[AbsEntry] AND Status <> 'C' END
ELSE BEGIN
SELECT AbsEntry FROM OPKL WHERE
Name = (SELECT firstName + ' ' + ISNULL(middleName + ' ', '') + lastName
FROM OHEM WHERE empID = $[EmployeeNo]) AND Status <> 'C'
ORDER BY AbsEntry
END
```

Note: this simple example only filters for employee or PickListNo, but doesn't respect other filters like Customer, Item, DueDate, Warehouse. It also allows the employee to enter a PickListNo and allows him to select that pick list even if he's not assigned for it.

2.4. Query Stocks

2.4.1. Override list

It is possible to override the list of items on the Query Stocks screen. Note: this is the same screen that can be opened from Pick List and from other modules in WH9 when pressing the Find Stocks button, so the custom logic is also relevant for those cases.

UserQuery: bx_mobile_wh9_querystocks_query_custom (Category: BXMobileWH9)	
Parameters	Results
\$[Warehouse]	A table with multiple rows with specific column names:
\$[BinLocation]	- Warehouse
\$[ItemCode]	- BinLocation
\$[BatchNumber]	- ItemCode
(\$[..] - other user fields from screen)	- ItemName
	- ManagedBy (Batch: 10000044, Serial: 10000045, None: -1)
	- OnHandQuantity (in inventory UoM)

Example custom query which returns items sorted by quantity (descending). Note: this query doesn't filter by batch input parameter, only Warehouse, ItemCode and BinLocation.

```
-- bx_mobile_wh9_querystocks_query_custom
-- return maximum 20(+20) matches by filters, ordered by quantity descending
-- first select is for bin-activated warehouses
SELECT TOP 20 OIBQ.WhsCode as Warehouse, OIBQ.BinCode as BinLocation,
OIBQ.ItemCode, OITM.ItemName,
CASE WHEN OITM.ManSerNum = 'Y' THEN 10000045
  WHEN OITM.ManBtchNum = 'Y' THEN 10000044
  ELSE -1 END as ManagedBy, OIBQ.OnHandQty as OnHandQuantity
FROM OIBQ
JOIN OIBIN ON (OIBIN.AbsEntry = OIBQ.BinAbs)
JOIN OITM ON (OITM.ItemCode = OIBQ.ItemCode)
WHERE
(OIBQ.ItemCode = ${ItemCode} OR ${ItemCode} = '') AND
(OIBQ.WhsCode = ${Warehouse} OR ${Warehouse} = '') AND
(OIBIN.BinCode = ${BinLocation} OR ${BinLocation} = '')
AND OnHandQty > 0
UNION
-- this second select is for non-bin warehouses
SELECT TOP 20 OITW.WhsCode as Warehouse, '' as BinLocation, OITW.ItemCode,
OITM.ItemName,
CASE WHEN OITM.ManSerNum = 'Y' THEN 10000045 WHEN OITM.ManBtchNum = 'Y' THEN
10000044 ELSE -1 END as ManagedBy, OITW.OnHand as OnHandQuantity FROM OITW
JOIN OITM ON (OITM.ItemCode = OITW.ItemCode)
JOIN OWHS ON (OWHS.WhsCode = OITW.WhsCode)
WHERE
OWHS.BinActivat = 'N' AND
(OITW.ItemCode = ${ItemCode} OR ${ItemCode} = '') AND
(OITW.WhsCode = ${Warehouse} OR ${Warehouse} = '')
-- order by quantity descending
ORDER BY OnHandQuantity DESC
```

3. General for multiple processes

3.1. Creating documents as drafts

For goods receipts, deliveries and stock transfers, it's possible to control whether the documents should be created as drafts or as real documents, when posted from BX Mobile WH 9. The logic controlling this needs to be defined in a user query.

UserQuery: bx_mobile_wh9_document_creation_type (Category: BXMobileWH9)	
Parameters [%1]: employee ID (int) [%2]: terminal ID (nvarchar) [%3]: mobile transaction head code (nvarchar)	Results The user query should return the result in a column called BXDOCTYP. The result must be an integer, and the following values are supported: - 0: real document - 1: draft

For example, with the following logic, all goods receipt PO documents (doc. type = 20) will be created as drafts, while the other documents are created as real documents:

```
SELECT CASE T0.[U_BXPDocTy] WHEN 20 THEN 1 ELSE 0 END as 'BXDOCTYP' FROM
[dbo].[@BXPLMSMOBTHD] T0 WHERE T0.[Code] = [%3]
```

3.2. Manipulating DataRepeater (List)

You will need a splitter function, to separate data. If it does not exist, then you can create them with these queries:

```
CREATE FUNCTION [dbo].[SplitStringForDataRepeater]
(
    @string NVARCHAR(MAX)
)
RETURNS @output TABLE(splitdata NVARCHAR(MAX)
)
BEGIN
    DECLARE @start INT, @end INT
    SELECT @start = 1, @end = CHARINDEX('##', @string)
    WHILE @start < LEN(@string) + 1 BEGIN
        IF @end = 0
            SET @end = LEN(@string) + 2
        INSERT INTO @output (splitdata)
        VALUES(SUBSTRING(@string, @start, @end - @start))
        SET @start = @end + 2
        SET @end = CHARINDEX('##', @string, @start)
    END
    RETURN
END
```

```
CREATE FUNCTION [dbo].[SplitStringForDataRepeater2]
( @string NVARCHAR(MAX), @string2 NVARCHAR(MAX) )
RETURNS @output TABLE(splitdata NVARCHAR(MAX) , splitdata2 NVARCHAR(MAX))
BEGIN
    DECLARE @start INT, @end INT , @start2 INT, @end2 INT
    SELECT @start = 1, @end = CHARINDEX('##', @string) , @start2 = 1, @end2
=CHARINDEX('##', @string2)
    WHILE @start < LEN(@string) + 1 BEGIN
        IF @end = 0
            BEGIN
                SET @end = LEN(@string) + 2
                SET @end2 = LEN(@string2) + 2
            END
        INSERT INTO @output (splitdata, splitdata2)
        VALUES(SUBSTRING(@string, @start, @end - @start),SUBSTRING(@string2,
@start2, @end2 - @start2))
        SET @start = @end + 2
        SET @end = CHARINDEX('##', @string, @start)
```

```
SET @start2 = @end2 + 2
SET @end2 = CHARINDEX('##', @string2, @start2)
END
RETURN END
```

3.2.1. Add a string

This customization will add a string to the Warehouse field on Sales Order Issue Lines screen.

Query name

BXMobileWH9_SalesIssueLinesScreen_DataRepeater_InternalDataLoad

Query content

```
SELECT
    S.splitdata + ' Additional Info' AS [DataRepeater.UIWarehouse] from
    dbo.SplitStringForDataRepeater($[DataRepeater.UIWarehouse]) S
```

3.2.2. Add information from RDR1

This customization will add information from RDR1 table by lineNumber.

We can load DocNumber from \$[TextSalesOrder] field, and we can get the exact number with SUBSTRING. SplitStringForDataRepeater2 function splits UIWarehouse UILineNumber from the datarepeater, and then we can use them to connect to SAP table, and we can use the splitted values in the query.

Query name

BXMobileWH9_SalesIssueLinesScreen_DataRepeater_InternalDataLoad

Query content

```
SELECT
    S.splitdata + ' - ' + ItemCode AS [DataRepeater.UIWarehouse] FROM
    dbo.SplitStringForDataRepeater2(
        $[DataRepeater.UIWarehouse] ,
        $[DataRepeater.UILineNumber] ) S
LEFT JOIN
    (SELECT
        '#' + cast(LineNum + 1 as varchar(10)) as ln,
        ItemCode
    FROM
        RDR1 left join ORDR on RDR1.DocEntry = ORDR.DocEntry
    WHERE
        ORDR.DocNum = SUBSTRING($[TextSalesOrder] ,0,CHARINDEX('*',
        $[TextSalesOrder] )) ) as RDR1Line
    ON S.splitdata2 = RDR1Line.ln
```

4. Special customization

You can use this special customization to manipulate the loaded data

4.1.1. bx_mobile_wh9_querystocks_query_custom

Parameters which can be used in the query: ItemCode, Warehouse

Query fields: Warehouse, BinLocation, ItemCode, ItemName, ManagedBy, OnHandQuantity

This example will order the result by batch number.

```
SELECT TOP 20
    OIBQ.WhsCode as Warehouse,
    OIBN.BinCode as BinLocation,
    OIBQ.ItemCode,      OITM.ItemName,
    CASE WHEN OITM.ManSerNum = 'Y' THEN 10000045 WHEN OITM.ManBtchNum = 'Y'
THEN
10000044 ELSE -1 END as ManagedBy,
    OIBQ.OnHandQty as OnHandQuantity
FROM OIBQ
    JOIN OIBN ON (OIBN.AbsEntry = OIBQ.BinAbs)
    JOIN OITM ON (OITM.ItemCode = OIBQ.ItemCode)
    JOIN OBBQ ON (OBBQ.BinAbs = OIBQ.BinAbs)
    JOIN OBTN ON (OBBQ.SnBMDAbs = OBTN.AbsEntry)
WHERE
    (OIBQ.ItemCode = ${ItemCode} OR ${ItemCode} = '') AND
    (OIBQ.WhsCode = ${Warehouse} OR ${Warehouse} = '') AND
    (OBBQ.ItemCode = ${ItemCode} OR ${ItemCode} = '') AND
    (OBBQ.WhsCode = ${Warehouse} OR ${Warehouse} = '') AND
    OIBQ.OnHandQty > 0
    AND      OBBQ.OnHandQty > 0
ORDER BY
    OBTN.DistNumber
```

4.1.2. bx_mobile_wh9_picklists_query_custom

Parameters which can be used in the query: AbsEntry, CardCode, EmployeeNo, ItemCode, PickDate, WarehouseCode, BinLocationCode

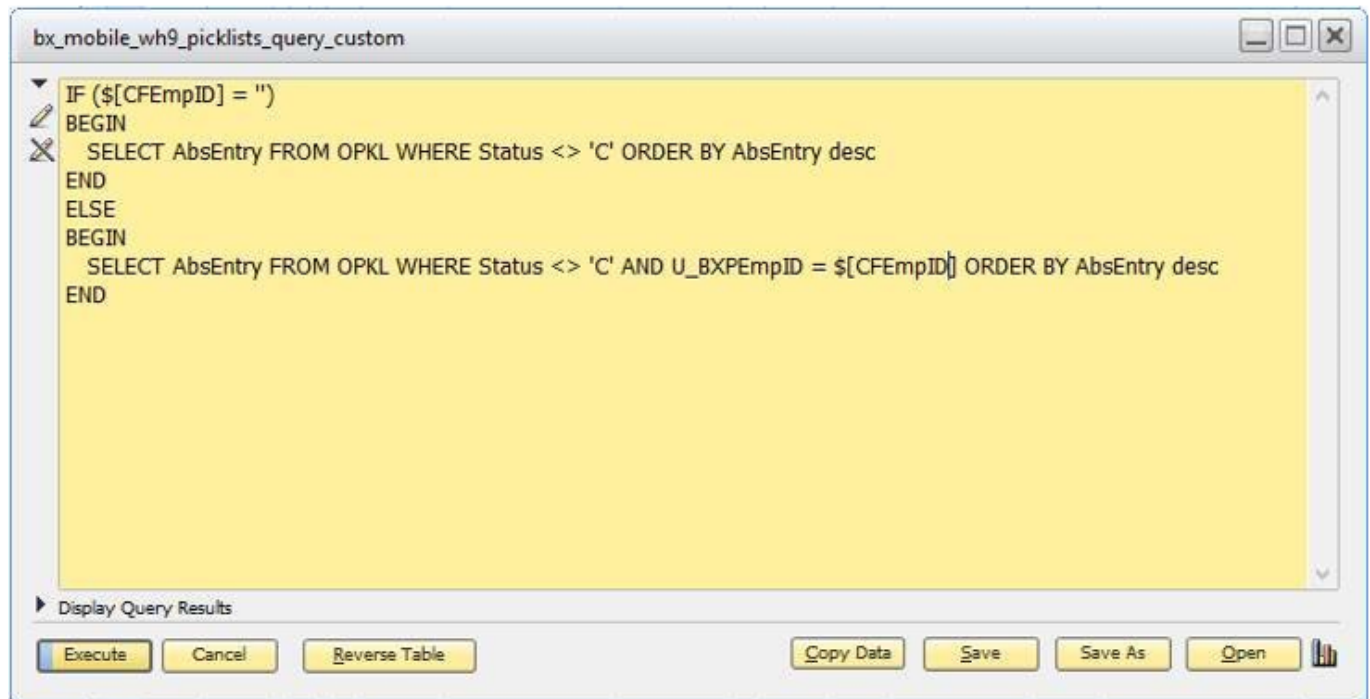
Query fields: AbsEntry

On the pick list screen, you can use custom fields in the custom query.

For example you can add a new field for employee input (CFEmpID)

CustomizationFields																
#	Code	Name	I	E	Field D...	Field Name	Field Type	F...	Label	La...	Module	On E...	Position Data	Rea...	Screen	Sor...
1	11001	11001			None	CFEmpID	String		EmpID		EXMobileWH9	No		No	PickingScreen	
2					None		String					No		No		

This filed can be used in the user query



4.1.3. bx_mobile_wh9_salesorderlines_query_custom

Parameters which can be used in the query: DocEntry
Query fields: LineNum, BinCode

4.2. Button customization

If you want to click on Reload button after scanning a document number, then you have create a user query with validate after event on the field.

For example on GR PO screen, if you want to click on Reload automatically after scannig, then you have to create a custom query:

```
BXMobileWH9_GoodsReceiptPOScreen_TextDocumentNumber_validate_after
```

And if the document field is not empty, then you can click on Reload button:

```
IF ${TextDocumentNumber} <> '' BEGIN

SELECT 'ButtonReload' as 'Click$'
END
```

4.3. Custom message

You can send a message to employee is a custom event.

```
SELECT 'Information' as 'Message$', 'I' as 'MessageType$'
```

Messages can be one of these types:

- I : Information
- E : Error

From:

<http://wiki.produmex.name/> -

Permanent link:

<http://wiki.produmex.name/doku.php?id=implementation:bxmobwh:customizationtechnology> 

Last update: **2017/02/22 14:09**