2. Customization Examples

2.1. Goods Receipt PO

Helpful Tips and Resources

Click the link below to visit our Article site, where you will find examples and useful information. We are continuously adding new articles featuring the most common customizations.

Produmex Scan Articles: Customization

2.1.1. Set (user) fields in GR PO

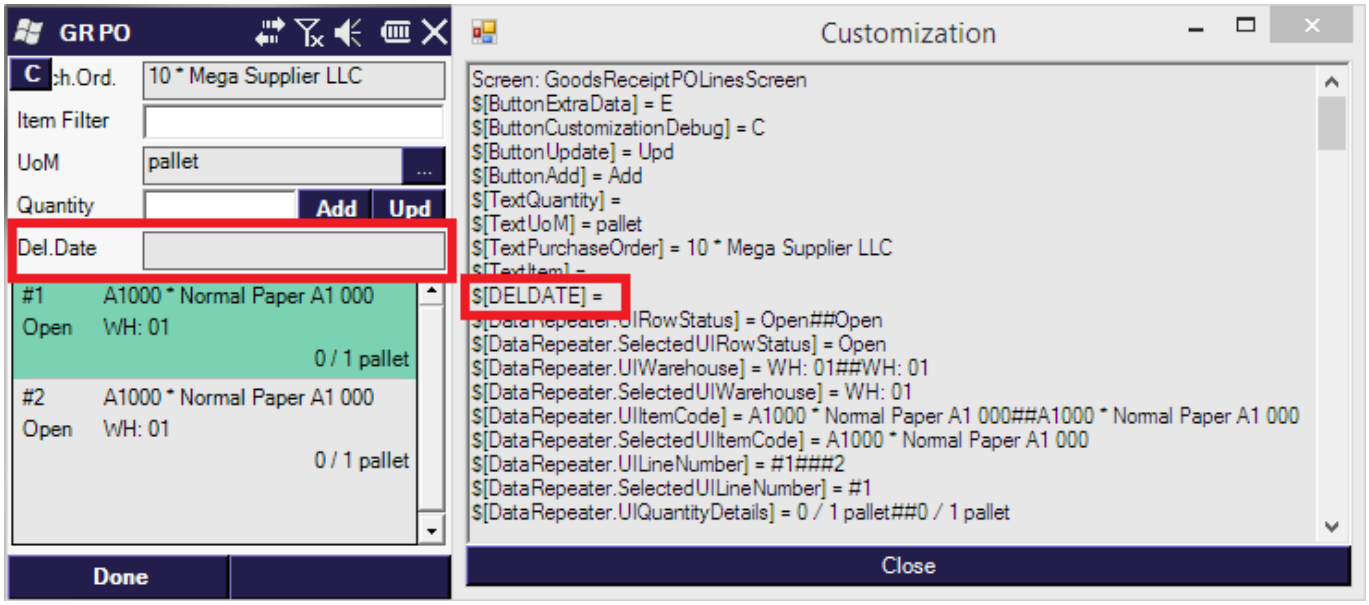
The customization makes possible to set additional fields (SAP or user fields) in the Goods Receipt PO document from.

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 Business One. Use the recommended parameters as below:

Screen	Module	Label	Field Name	Read Only
GoodsReceiptPOLinesScreen	BXMobileWH9	Del.Date	DELDATE	YES

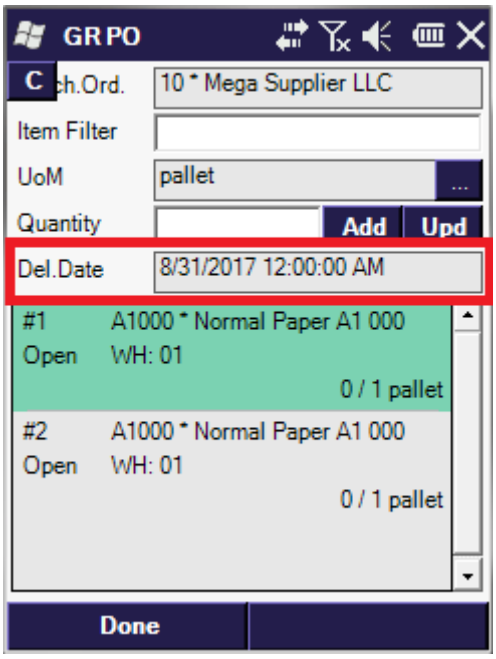
Now you are able to load data into this new field. To do it, you have to use the *BXMobileWH9_GoodsReceiptPOLinesScreen_Load* user query.



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.
In the example: $[TextPurchaseOrder] = 10 * 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

It is possible to set the Freight costs/lines in the created Goods Receipt PO with user query. Please see

2.1 Set Freight in Delivery document for the example query.

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)

2.1.3. Batch number generation

On some screens (e.g. Goods Receipt PO, Receipt from Production) the system can automatically generate batch numbers when entering the received quantities and bin locations. The batch number generation logic can 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 (type of the field should be NVARCHAR)

Note: The standard *click* and *click_after* events cannot use the parameters of the bx_mobile_wh9_get_new_batchnumber query.

Example:
The following query generates 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

The following 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”;

These fields can be added on the CustomizationFields table.

Example for batches:

Field Name	Label	Screen
#ExpirationDate	BBD	ReceiptFromProductionQuantitiesBatchScreen
#ManufacturingDate	ManufacturingDate	ReceiptFromProductionQuantitiesBatchScreen
#ExpirationDate	BBD	GoodsReceiptPOQuantitiesBatchScreen

2.1.6. Generate GRPO from Draft

> **Note:** This customization is **compatible from SAP Business One version 2405 or later versions.** It will not function on earlier versions!

With this customization GRPOs can be created from draft. When starting the Goods Receipt Purchase Order function on the scanner and there is a draft GRPO registered in SBO for the given warehouse, the draft document is listed among the Purchase Orders.

The 'Post' button is immediately active for draft documents when selected and the 'Receive' button is hidden/not available. When the 'Post' button is pressed while a GRPO draft is selected, the GRPO is created from the GRPO draft with the exact same stock parameters as the GRPO draft.

Note: DR = draft GRPO (new), PO = Purchase Order, RI = A/P Reserve Invoice

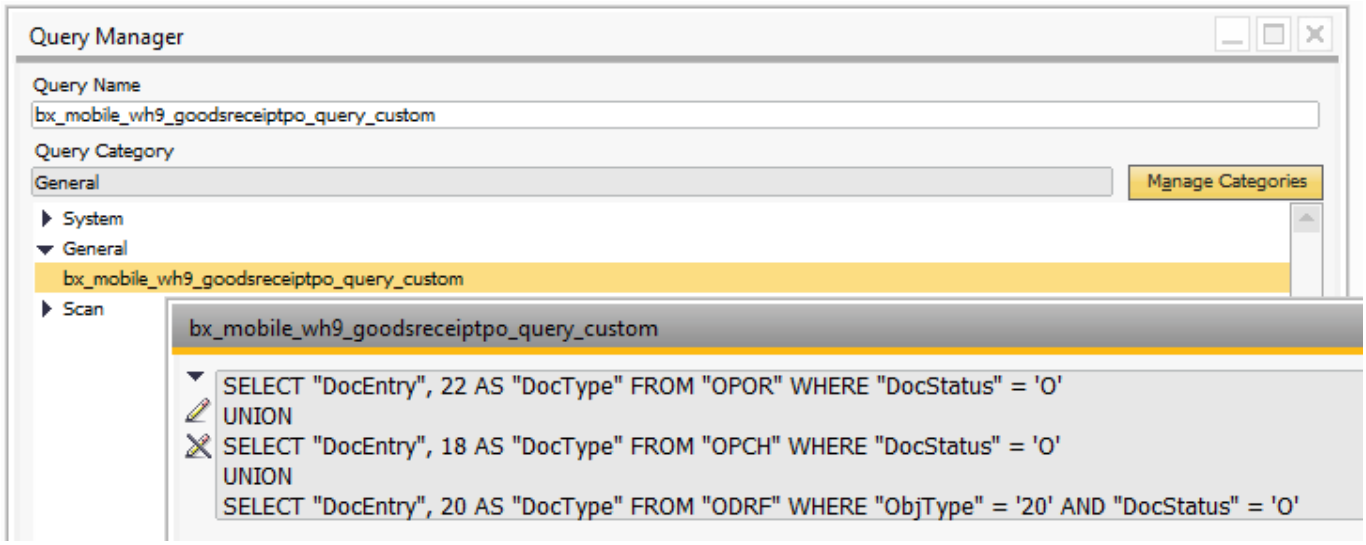
UserQuery: bx_mobile_wh9_goodsreceiptpo_query_custom

```
SELECT
    Distinct "DocEntry", "DocType", ItemCode
FROM (
```

```
    SELECT "OPOR"."DocEntry", 22 as "DocType", "CardCode", "DocNum",
"DocDueDate", "POR1"."ItemCode" FROM "POR1" LEFT JOIN "OPOR" ON
"POR1"."DocEntry" = "OPOR"."DocEntry" WHERE "DocStatus" = 'O'
    UNION
    SELECT "OPCH"."DocEntry", 18 as "DocType", "CardCode", "DocNum",
"DocDueDate", "PCH1"."ItemCode" FROM "PCH1" LEFT JOIN "OPCH" ON
"PCH1"."DocEntry" = "OPCH"."DocEntry" WHERE "DocStatus" = 'O'
    UNION
    SELECT "ODRF"."DocEntry", 20 as "DocType", "CardCode", "DocNum",
"DocDueDate", "DRF1"."ItemCode" FROM "DRF1" LEFT JOIN "ODRF" ON
"DRF1"."DocEntry" = "ODRF"."DocEntry" WHERE "ODRF"."ObjType" = '20' AND
"DocStatus" = 'O' AND "ODRF"."WddStatus" in ('Y','')
    ) as DOCUMEND
WHERE
    (CASE WHEN isnull($[ItemCode], '') = '' THEN 1 ELSE CASE WHEN
"DOCUMEND"."ItemCode" = $[ItemCode] THEN 1 ELSE 0 END END) = 1
    AND (CASE WHEN isnull($[DocNum], '') = '' THEN 1 ELSE CASE WHEN
"DOCUMEND"."DocNum" = $[DocNum] THEN 1 ELSE 0 END END) = 1
    AND (CASE WHEN isnull($[CardCode], '') = '' THEN 1 ELSE CASE WHEN
"DOCUMEND"."CardCode" = $[CardCode] THEN 1 ELSE 0 END END) = 1
    AND (CASE WHEN isnull($[DueDate], '') = '' THEN 1 ELSE CASE WHEN
"DOCUMEND"."DocDueDate" = $[DueDate] THEN 1 ELSE 0 END END) = 1
```

Implementation:

Add the query in the Query Manager in SBO



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 Produmex Scan 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 [%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)

Example:
A combined 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 BO_ExpenseCode,
    LineTotal as BO_LineTotal,
    17 as BO_BaseDocType,
    @salesOrderDocEntry as BO_BaseDocEntry,
    LineNum as BO_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 PO
IF @purchaseOrderDocEntry > 0
BEGIN
SELECT
    ExpnsCode as BO_ExpenseCode,
    LineTotal as BO_LineTotal,
    22 as BO_BaseDocType,
    @purchaseOrderDocEntry as BO_BaseDocEntry,
    LineNum as BO_BaseDocLine
FROM POR3
WHERE DocEntry = @purchaseOrderDocEntry AND Status <> 'C'
END
```

HANA version

```
CREATE PROCEDURE PMXSCAN_ADDITIONALEXPENSES (
    IN
    pHeadCode NVARCHAR(5000)
)
LANGUAGE SQLSCRIPT
AS
BEGIN
    declare purchaseOrderDocEntry int;
    declare salesOrderDocEntry int;
    SELECT SUM("U_BXPBsDcE") into salesOrderDocEntry FROM
    (
        SELECT 0 "U_BXPBsDcE" FROM DUMMY
        UNION
        SELECT "U_BXPBsDcE"
        FROM "@BXPLMSMOBTLN"
        WHERE "U_BXPHdCd" = pHeadCode AND "U_BXPBsDcT" = 17 AND "U_BXPDocTy"
= 15
    );

    -- 15-delivery, 17-sales order
    IF salesOrderDocEntry > 0 THEN
        SELECT
            "ExpnsCode" as "BO_ExpenseCode",
            "LineTotal" as "BO_LineTotal",
            17 as "BO_BaseDocType",
            salesOrderDocEntry as "BO_BaseDocEntry",
            "LineNum" as "BO_BaseDocLine"
        FROM "RDR3"
        WHERE "DocEntry" = salesOrderDocEntry AND "Status" <> 'C';
    END IF;

    SELECT SUM("U_BXPBsDcE") into purchaseOrderDocEntry FROM
    (
        SELECT 0 "U_BXPBsDcE" FROM DUMMY
        UNION
        SELECT "U_BXPBsDcE"
        FROM "@BXPLMSMOBTLN"
        WHERE "U_BXPHdCd" = pHeadCode AND "U_BXPBsDcT" = 22 AND "U_BXPDocTy"
= 20
    );
    -- 22-Purchase order, 20-Goods Receipt PO
    IF purchaseOrderDocEntry > 0 THEN
        SELECT
            "ExpnsCode" as "BO_ExpenseCode",
            "LineTotal" as "BO_LineTotal",
            22 as "BO_BaseDocType",
            purchaseOrderDocEntry as "BO_BaseDocEntry",
            "LineNum" as "BO_BaseDocLine"
        FROM "POR3"
        WHERE "DocEntry" = purchaseOrderDocEntry AND "Status" <> 'C';
    END IF;
```

```
END;


SAP Query name:
bx_mobile_wh9_document_additionalexpenses

content:
call "PMXSCAN_ADDITIONALEXPENSES"( [%3] );
```

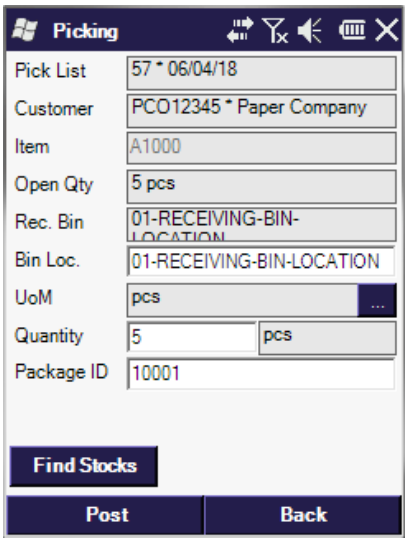
This example user query looks at the Mobile Transaction (lines) table, filters for Base Document Type = Sales Order, finds the base Sales Order Document DocEntry. It then retrieves the freight expense records related to the Sales Order (from table RDR3), and extracts Expense Code and Line Total, so it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

2.2.2. Packing ID

It is possible to record packaging ID during picking. The package ID screen can be added to the picking lines screen.

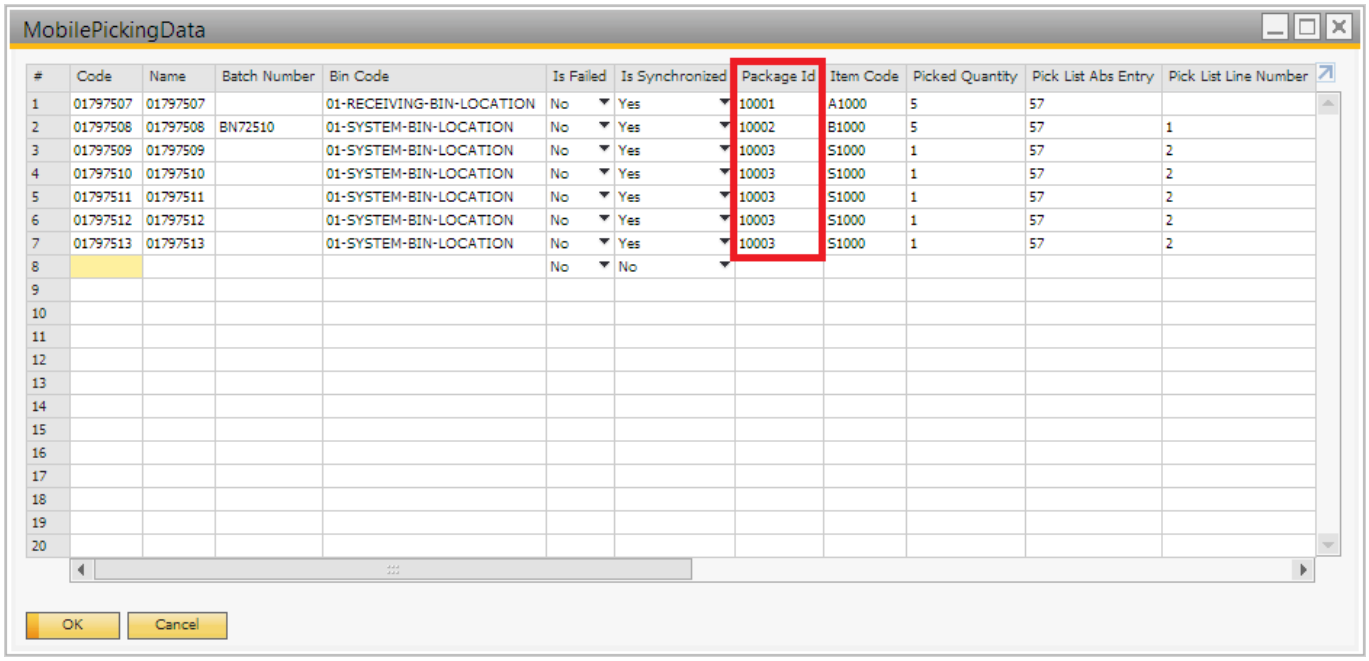
Example:

Field Name	Label	Screen
#PackageId	Package ID	PickingLinesPickNormalScreen
#PackageId	Package ID	PickingLinesPickBatchScreen
#PackageId	Package ID	PickingLinesPickSerialScreen



The Package ID is automatically filled with the last Package ID value that was added for the pick list. Both numbers and letters are supported in this field.

The package ID is stored on the MobilePickingData (BXPLMSMOBPICK) user table.



#	Code	Name	Batch Number	Bin Code	Is Failed	Is Synchronized	Package Id	Item Code	Picked Quantity	Pick List Abs Entry	Pick List Line Number
1	01797507	01797507		01-RECEIVING-BIN-LOCATION	No	Yes	10001	A1000	5	57	
2	01797508	01797508	BN72510	01-SYSTEM-BIN-LOCATION	No	Yes	10002	B1000	5	57	1
3	01797509	01797509		01-SYSTEM-BIN-LOCATION	No	Yes	10003	S1000	1	57	2
4	01797510	01797510		01-SYSTEM-BIN-LOCATION	No	Yes	10003	S1000	1	57	2
5	01797511	01797511		01-SYSTEM-BIN-LOCATION	No	Yes	10003	S1000	1	57	2
6	01797512	01797512		01-SYSTEM-BIN-LOCATION	No	Yes	10003	S1000	1	57	2
7	01797513	01797513		01-SYSTEM-BIN-LOCATION	No	Yes	10003	S1000	1	57	2
8					No	No					
9											
10											
11											
12											
13											
14											
15											
16											
17											
18											
19											
20											

2.3. Pick Lists

2.3.1. Set Freight in Delivery

Please see: 2.1 Set Freight in Delivery document

2.3.2. Set (user) fields in Delivery

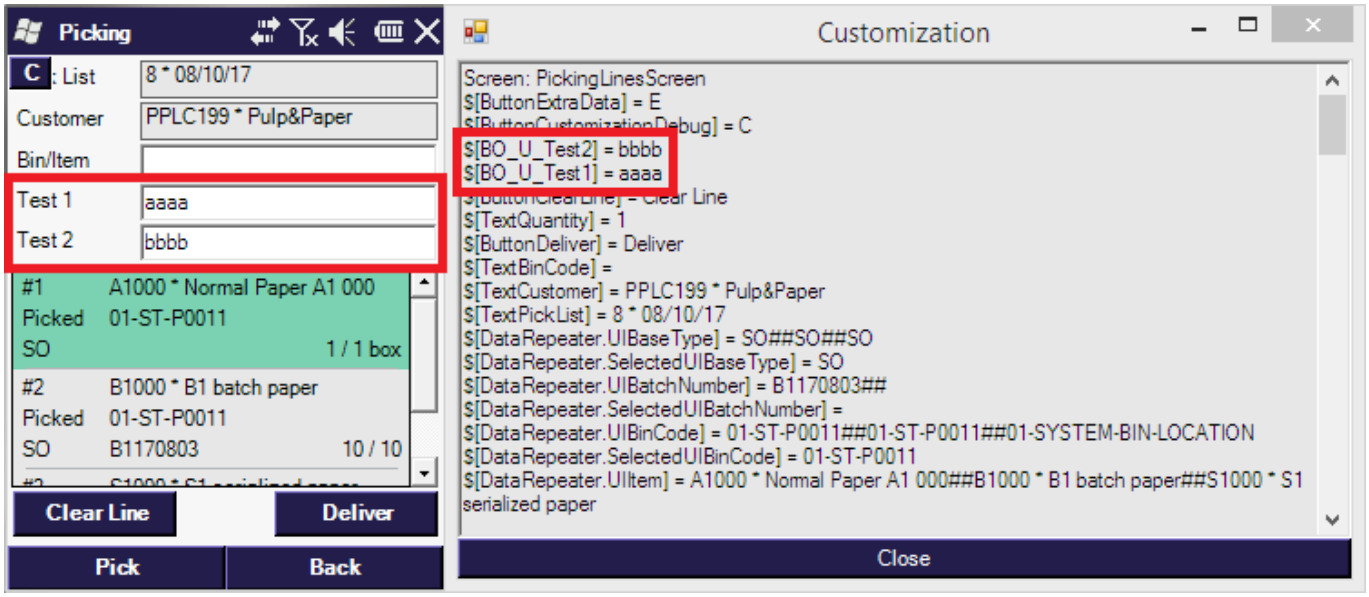
It is possible to set the values of the Delivery Document which are created from the Produmex Scan Pick List screen.
To set a field, you have to add a customization field to the Pick List Lines screen. The customization field name must be:

- **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. Example: BO_JournalMemo

In the Customization Fields 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.

Example:

Screen	Module	Label	Field Name
PickingLinesScreen	BXMobileWH9	Test 1	BO_U_Test1
PickingLinesScreen	BXMobileWH9	Test 2	BO_U_Test2



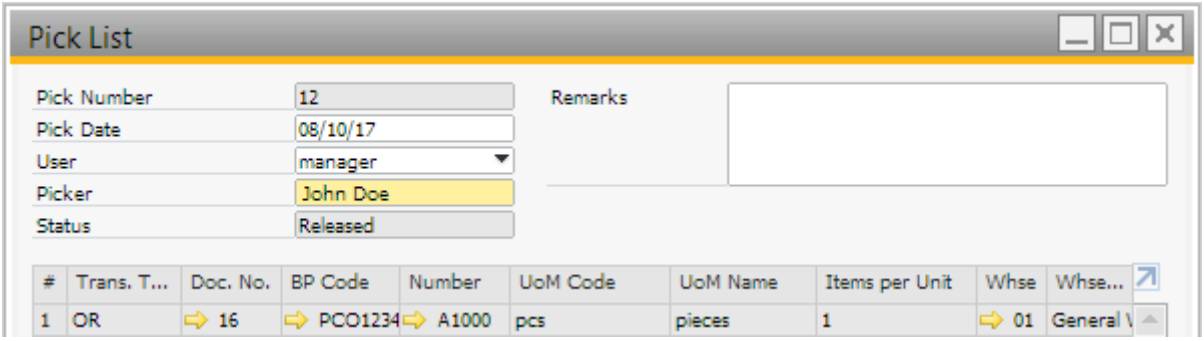
When pressing the Delivery button on the Pick List lines screen, these field values will be used in the newly created Delivery Document.

2.3.3. Pick List screen - customize list

With the default settings, only open pick lists that have not been started by anyone are displayed, but the original program logic can be overridden by customization.

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:
With the example user query the list of picklists is filtered down to picklist assigned to the employee. We used the 'Picker' field on the Pick list to assign the employee to the pick list.
Please note: The Picker field must be in the 'FirstName LastName' or 'FirstName MiddleName LastName' format for the example 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
```

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.

2.3.4. Auto fill batch with recommended batch

To automatically populate the Batch field with the recommended batch, add the following query:

Query name: *BXMobileWH9_PickingLinesPickBatchScreen_Activate*

```
SELECT $[TextRecBatch] AS [TextBatch]
```

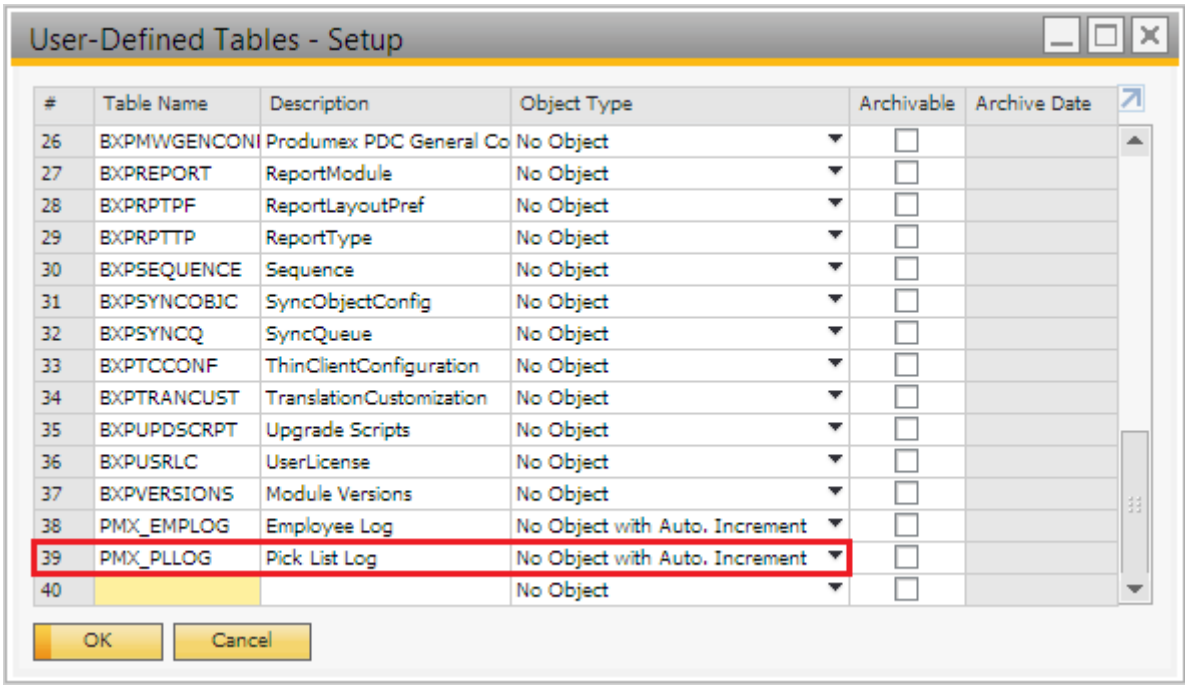
2.3.5. Capture pick list selection events into a separate table

It is possible to capture the date and time when an employee selects a pick list into a separate table.

Create the user table

First create the user table for the pick list selection events. Example: PMX_PLLOG user table

Set the object type to ’No object with Auto.Increment’.



Add the user defined fields to the table. In the example we will add the following fields:

Title	**Description**	**Type**
Date	Date	Date/Time
Time	Time	Numeric
EmpID	Employee ID	Alphanumeric
EmpName	Employee Name	Alphanumeric
PLN	Pick List	Numeric

Create the user query

The user query name is: *BXMobileWH9_PickingScreen_OK_clicked*

SQL

```
INSERT INTO "@PMX_PLLOG" ("Name", "U_Date", "U_Time", "U_EmpID",
"U_EmpName", "U_PLN")
values ('Pick List Entry', cast(getdate() as date),
cast(substring(CONVERT(VARCHAR,GETDATE(),108),1,2) * 100 +
substring(CONVERT(VARCHAR,GETDATE(),108),4,2) as int),
$[Employee.EmployeeID], $[Employee.FirstName]+ ' '+  $[Employee.LastName],
SUBSTRING($[DataRepeater.SelectedUIPickListNo], CHARINDEX('#',
$[DataRepeater.SelectedUIPickListNo]) + 1,
LEN($[DataRepeater.SelectedUIPickListNo]) - CHARINDEX('#',
$[DataRepeater.SelectedUIPickListNo])))
```

HANA

```
INSERT INTO "@PMX_PLLOG" ("Name", "U_Date", "U_Time", "U_EmpID",
"U_EmpName", "U_PLN")
```

```
values ('Pick List Entry', cast(getdate() as date),
cast(substring(CONVERT(VARCHAR,GETDATE(),108),1,2) * 100 +
substring(CONVERT(VARCHAR,GETDATE(),108),4,2) as int),
$[Employee.EmployeeID], $[Employee.FirstName]+ ' '+  $[Employee.LastName],
SUBSTRING($[DataRepeater.SelectedUIPickListNo],
LOCATE($[DataRepeater.SelectedUIPickListNo], '#') + 1) FROM DUMMY´)
```

When the 'Pick' button is pressed, this query inserts the current date to the Date column, the current time to the Time column, the employee ID to the Employee ID column, the first and last name of the employee to the Employee Name column and the pick list number to the Pick List column.

2.4. Query Stocks

2.4.1. Override list

It is possible to override the list of items on the Query Stocks screen.

> This is the same screen that can be opened from Pick List and from other modules in Produmex Scan 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 $[Warehouse] $[BinLocation] $[ItemCode] $[BatchNumber] ($[..] - other user fields from screen)	**Results** A table with multiple rows with specific column names: - Warehouse - BinLocation - ItemCode - ItemName - ManagedBy (Batch: 10000044, Serial: 10000045, None: -1) - OnHandQuantity (in inventory UoM)

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

Query name: *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, OBIN.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 OBIN ON (OBIN.AbsEntry = OIBQ.BinAbs)
JOIN OITM ON (OITM.ItemCode = OIBQ.ItemCode)
WHERE (OIBQ.ItemCode = $[ItemCode] OR $[ItemCode] = '')
    AND (OIBQ.WhsCode = $[Warehouse] OR $[Warehouse] = '')
    AND (OBIN.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
```

2.5. General for multiple processes

2.5.1. Creating documents as drafts

It is possible to control whether the documents should be created as drafts or as real documents when posted from Produmex Scan for the following documents:

Document	**Doc. type**
Delivery	15
Sales Order (by *BN Create SO* function)	17
Goods Receipt PO	20
Goods Issue	60

The controlling logic must be defined with a user query.

UserQuery: bx_mobile_wh9_document_creation_type (Category: BXMobileWH9)	
Parameters [%1]: employee ID (int) [%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:

MS SQL

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

HANA

```
SELECT
case
T0."U_BXPDocTy" when 20 then 1
else 0
end as "BXDOCTYP"
FROM "@BXPLMSMOBTHD" T0 WHERE T0."Code" = [%3]
```

2.5.2.Special field for series numbering

A user field for series numbering can be added to screens where a Post event starts.
Field name: BO_Series

Example: Add a user field for numbering series to the GR PO screen
Add the following record to the Customization Fields user table:

Screen	Module	Label	Field Name	Read Only
GoodsReceiptPOPostSelectionScreen	BXMobileWH9	Series numbering	BO_Series	No

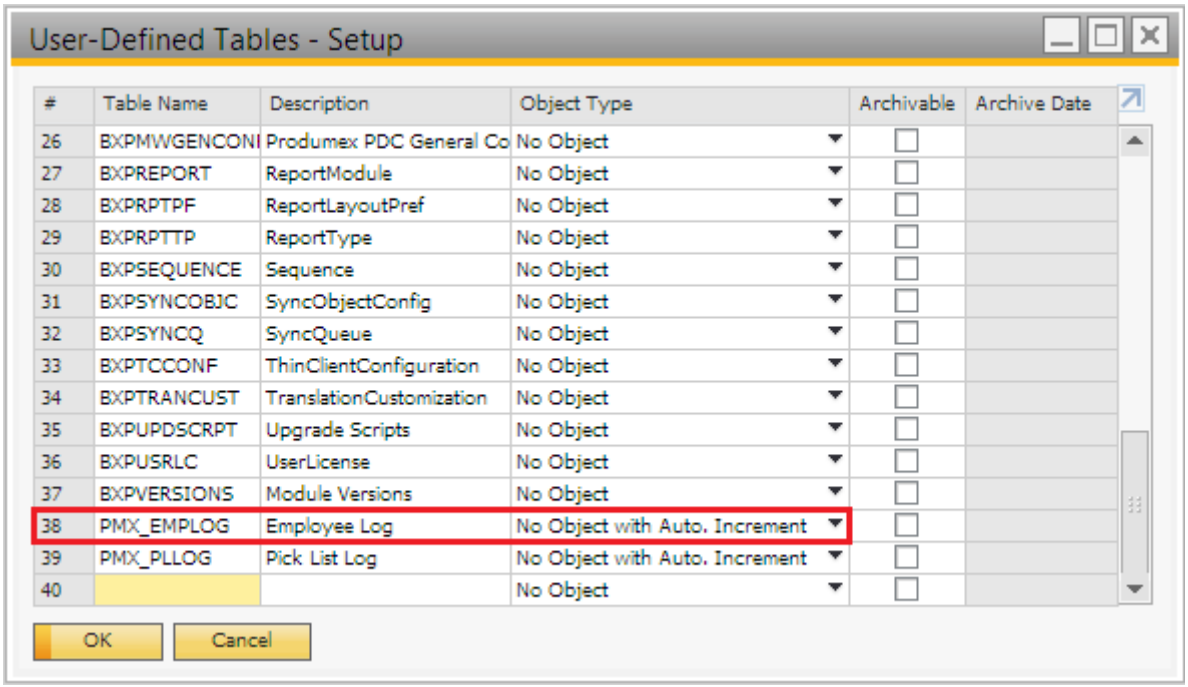
2.5.3.Capture login events into a separate table

It is possible to capture the date and time of the employee login and logout events in a separate user table.

Create the user table

First create the user table for the log in and log out events. Example: PMX_EMPLOG user table

Set the object type to ’No object with Auto.Increment’.



Add the user defined fields to the table. In the example we will add the following fields:

Title	**Description**	**Type**
Date	Date	Date/Time
Time	Time	Numeric
EmpID	Employee ID	Alphanumeric
EmpName	Employee Name	Alphanumeric

Create the user query

The user query name for the log in is: *BXMobileWH9_LoginScreen_OK_clicked*

SQL

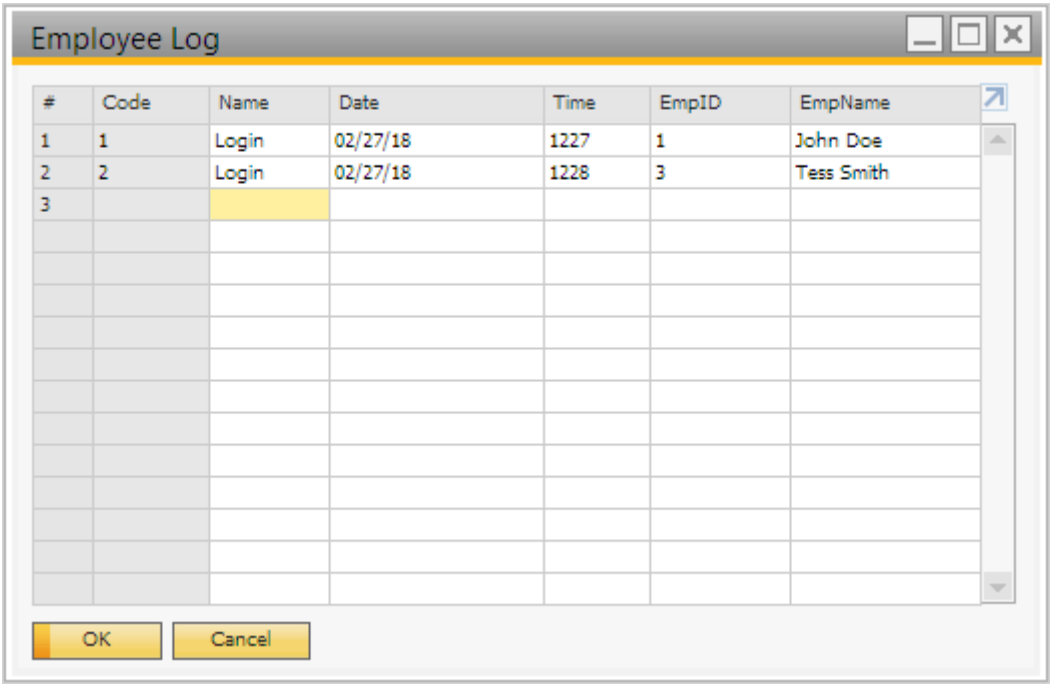
```
INSERT INTO "@PMX_EMPLOG" ("Name", "U_Date", "U_Time", "U_EmpID",
"U_EmpName")
values ('Login', cast(getdate() as date),
cast(substring(CONVERT(VARCHAR,GETDATE(),108),1,2) * 100 +
substring(CONVERT(VARCHAR,GETDATE(),108),4,2) as int), $[TextUser],
$[TextUserName])
```

HANA

```
INSERT INTO "@PMX_EMPLOG" ("Name", "U_Date", "U_Time", "U_EmpID",
"U_EmpName")
values ('Login', cast(current_timestamp as date), TO_INT( TO_VARCHAR(
CURRENT_TIMESTAMP, 'HH24MI' ) ), $[TextUser], $[TextUserName])
```

This query inserts the current date to the Date column, the current time to the Time column, the employee ID to the Employee ID column, the first and last name of the employee to the Employee

Name column and the event name in the Name column.



2.6. Special customization

You can use this special customization to manipulate the loaded data.

2.6.1. Special customization queries

2.6.1.1. Query Stocks

User query: bx_mobile_wh9_querystocks_query_custom	
Parameters $[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, OBIN.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 OBIN ON (OBIN.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
```

2.6.1.2. Picklist

User query: bx_mobile_wh9_picklists_query_custom	
Parameters $[AbsEntry], $[CardCode], $[EmployeeNo], $[ItemCode], $[PickDate], $[WarehouseCode], $[BinLocationCode]	**Query fields** AbsEntry

On the pick list screen, you can use custom fields in the custom query.
Example:
Add a new custom field 'CFEmpID' for employee input on the CustomizationFields table.

Field Name	**Label**	**Screen**
CFEmpID	EmpID	PickingScreen

After the field is added it can be used in a custom query. Query name:
bx_mobile_wh9_picklist_query_custom

```
IF ($[CFEmpID]='')
BEGIN
    SELECT AbsEntry FROM OPKL WHERE Status <> 'C'
    ORDER BY AbsEntry desc
END
ELSE
BEGIN
    SELECT AbsEntry FROM OPKL WHERE Status <> 'C' AND U_BXPEmpID=$[CFEmpID]
    ORDER BY AbsEntry desc
END
```

2.6.1.3. Sales order lines

User query: bx_mobile_wh9_salesorderlines_query_custom	
Parameters $[DocEntry]	**Query fields** LineNum, BinCode

2.6.1.4. Sales issue

User query: bx_mobile_wh9_salesissue_query_custom	
Parameters $[DocNum], $[CardCode], $[DueDate], $[ItemCode], $[EmployeeNo]	**Query fields** DocEntry, DocType (17:Sales Order; 13:A/R Reserve invoice)

2.6.1.5. Goods Receipt PO

User query: bx_mobile_wh9_goodsreceiptpo_query_custom	
Parameters $[DocNum], $[CardCode], $[DueDate], $[ItemCode], $[EmployeeNo]	**Query fields** DocEntry, DocType (18:A/P Reservere invoice; 22: Purchase Order)

2.6.1.6. Stock Transfer Request

User query: bx_mobile_wh9_stocktransferrequest_query_custom	
Parameters $[DocNum] $[WarehouseFrom] $[WarehouseTo]	**Query fields** DocEntry

2.6.1.7. Stock Transfer Request Lines

User query: bx_mobile_wh9_stocktransferrequestlines_query_custom	
Parameters $[DocEntry]	**Query fields** LineNum IsAvailable

2.6.1.8. Customize The order of the pick list lines

This query is designed to work with both **Speed Picking** and **Sequential Picking** processes. In both cases, the system will place the picked items at the end of the list without reloading the entire item list. This behavior is intentional and should remain the standard for these picking logics to maintain consistency and efficiency.

Important Note: The *“PickEntry”* field and the *“Order”* alias are mandatory and must be used!

User query: bx_mobile_wh9_picklistlines_query_custom	
Parameters $[AbsEntry]	**Query fields** PickEntry AS “Order”

```
IF $[AbsEntry] IS NOT NULL
BEGIN
    SELECT PickEntry AS "Order"
    FROM PKL1 WHERE AbsEntry = $[AbsEntry]
    ORDER BY RelQtty DESC, PickEntry DESC
END
```

2.6.2. Button customization

Example:
Automatically click on the ‘Reload’ button after scanning on the GR PO screen.

Query name: *BXMobileWH9_GoodsReceiptPOScreen_TextDocumentNumber_validate_after*

```
IF $[TextDocumentNumber] <> ''
BEGIN
SELECT 'ButtonReload' as 'Click$'
END
```

The name of the button can be found in customization assist. A custom button also can be pressed. The two buttons at the bottom of the screen are called 'OK' and 'Option'

2.6.3. Custom message

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

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

HANA version:

```
SELECT 'Information' as "Message$", 'I' as "MessageType$" FROM DUMMY
```

See the supported message types here: Supported message types

2.7. Other Examples

2.7.1. Populate Bin Location field with default item locaion in GR PO

You have to add the same query content for all item types

Query names:

BXMobileWH9_GoodsReceiptPOQuantitiesNormalScreen_Load
BXMobileWH9_GoodsReceiptPOQuantitiesBatchScreen_Load

BXMobileWH9_GoodsReceiptPOQuantitiesSerialScreen_Load

```
select
    "OBIN"."BinCode" as "TextBinLocation"
from
    "OITW"
    LEFT JOIN "OBIN" on "OITW"."DftBinAbs" = "OBIN"."AbsEntry"
where
    "OITW"."ItemCode" = $[SelectedPurchaseOrderLine.ItemCode]
    and "OITW"."WhsCode" = $[SelectedPurchaseOrderLine.WarehouseCode]
```

2.7.2. Delete the document number in screen load event in GR PO Screen

You have to add an SAP query with the name of the screen load event Query names:

BXMobileWH9_GoodsReceiptPOScreen_Load

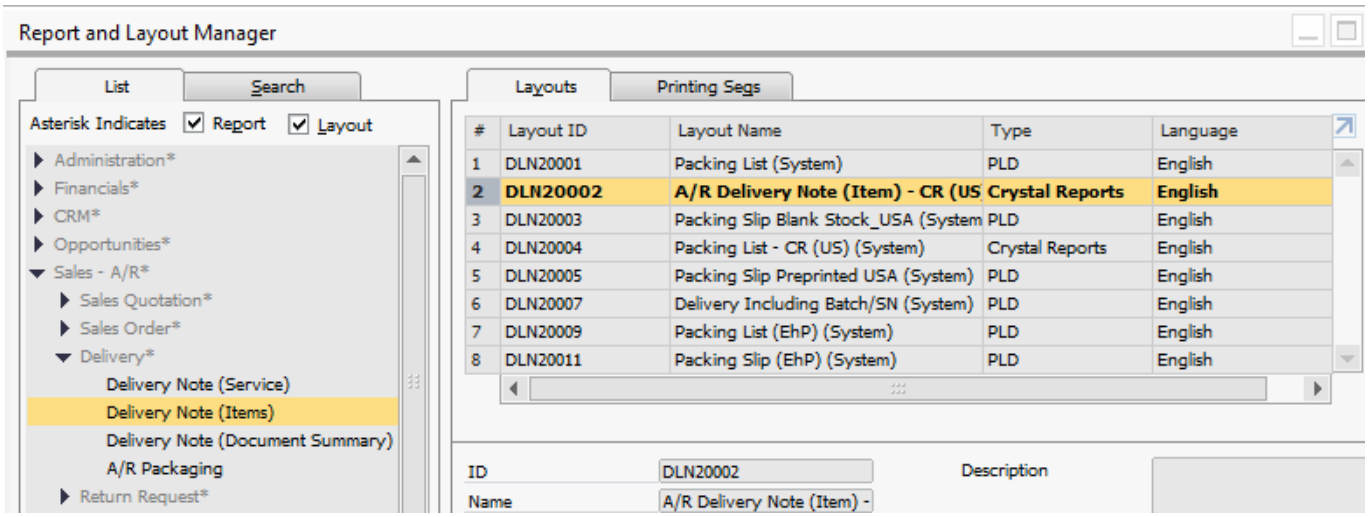
```
select '' as "TextDocumentNumber"
```

2.7.3. Auto print Delivery in Sales Issue process

It is possible to use the ButtonPost_click_after event to trigger the printing. The SAP document is already created in this event. The number of the new Delivery is not available in the customization, so we can only print the last Delivery document that was created by the logged in employee.

You can see the configuration steps below.

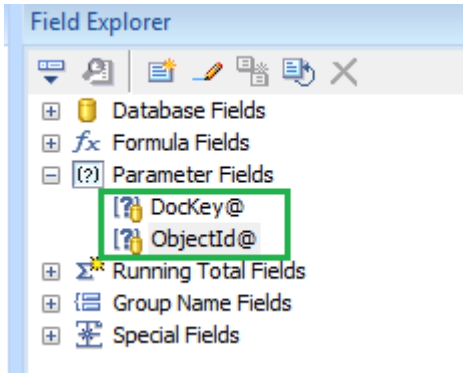
- Locate the Delivery note from the “Report and Layout Manager”



- Open it in Crystal Reports Designer.

- Set the database configuration for the report file and save a copy from the report.

- Check the paramters of the report file, we need them in the custom query.



-import the report file into SBO by function “Report and Layout Manager”

- Create the custom query

- Use the SBO Report ID in column: “PrintLayout$”
- Set all properties in a new column by addind “Print_” prefix
- Set the printer name in column: “PrintPrinter$”
- Send a message about the printing in columns “Message$” and “MessageType$”

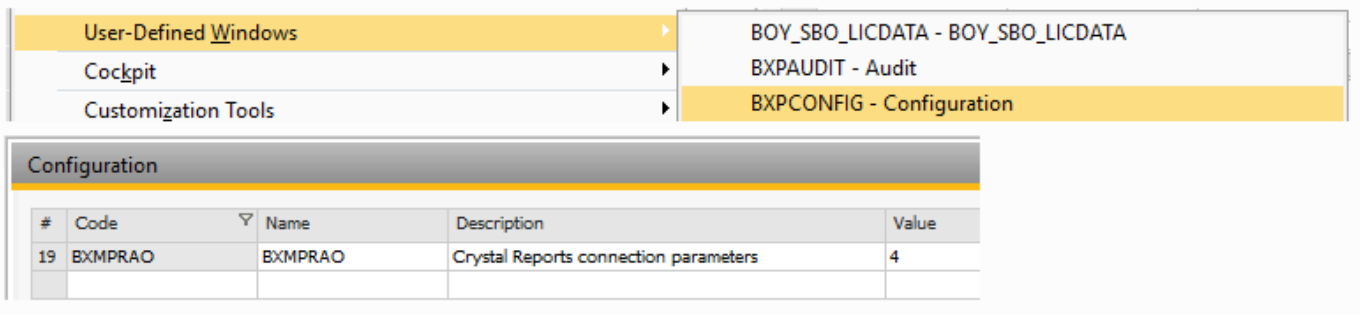
```
SELECT TOP 1
    'RCRI0013' as "PrintLayout$",
    "DocEntry" as "Print_DocKey@",
    15 as "Print_ObjectId@",
    'Bullzip PDF Printer' as "PrintPrinter$",
    'Delivey is Printed'  "Message$", 'I' "MessageType$"
FROM
    "ODLN"
WHERE
    "U_BXPEmpID" = $[Employee.EmployeeID]
ORDER BY "DocEntry" DESC
```

- The name od the query must be: BXMobileWH9_SalesIssueScreen_ButtonPost_click_after

- Produmex Scan application must be restarted after adding the new customization

- Since the report file contains sub reports you may enable it in the setting below. You can find more information about this setting on the url below

Advanced printing configurations



In the Picking process you can use the query name below:

BXMobileWH9_PickingLinesScreen_ButtonDeliver_click_after

The PickingLineScreen will be closed after createing the Delivery and the event won't be able to use objects from the PickingLineScreen anymore. It is necessary to check the setting below in order to prevent closing the screen.

“Don't close screen after picking delivery”