



EasyInput

a more effective user work due to integration of SAP and MS Excel

Batch Input (BDC) Technique and Transaction Scripts

- EasyInput is one of the products from the **BCC EXTRA offering**.
- BCC EXTRA is a set of unique products allowing you to boost the effectiveness of the use of SAP systems at low cost. These are functionalities developed in-house by BCC, enhancing the SAP system standard, and the tools facilitating the SAP system implementation and development.
- We build EXTRA products based on a many-year implementation experience gained while working for different customers from Poland and abroad.

More BCC EXTRA products at: www.bccgroup.eu/extra

- **Batch Input** (called also **BDC** = Batch Data Communication) is a technique for mass input of data by simulating user inputs in screens of existing transactions. Screens are not displayed. Batch Input can be run in the background. Important: With Batch Input one do not bypass any of the standard SAP consistency checks, authorizations, etc.
- Batch Input was a first way of uploading data into SAP in a pre-BAPI era, now SAP is making available BAPIs for most of the functionalities. It is recommended to first look out for BAPI and if it is not available then go for Batch Input.

- Batch Input technique was originally meant to be used inside ABAP programs via instruction „Call Transaction Using“ or in Batch Input Sessions. **The same Batch Input technique can be used in EasyInput transactions scripts.**
- Although BAPI function modules used in functional scripts are less prone to cause problems with upgrades, transaction scripts can be simpler to create for business users that know the transactions they use very well.



- The **Transaction Recorder (SHDB transaction)** is used to record any SAP transaction. The recording has the form of **Batch Input Script**.
- The recording can be then used within EasyInput workbook (imported to EI_SCRIPT_TR worksheet and linked to data worksheet).
- Important:
 - F1, F4 helps are not recorded
 - In ScreenPainter screens, movements in the scrollbar are not recorded. Use the function keys or tab for positioning.

- Usually the default parameters of transaction recording with Transaction Recorder (SHDB transaction) can be used (Standard, Batch Input mode).
- In EasyInput there are two different script modes that can be used to record a transaction:
 - **Standard (Batch Input)** – This is the most commonly used mode. This mode operates in BDC mode within SAP so screens might look different from manual entry. Standard mode can only upload data into SAP. This mode does not need e-CATT to be working in the SAP environment and it's performance is better as the extended mode.
 - **Extended (Non-Batch Input, e-CATT)** – This mode also operates in BDC mode within SAP but allows you to read data from SAP in addition to uploading data. Additionally in this mode the Batch Input marker is not set, so that the screen are usually identical to manual entry.

Important! Reading data from transaction has negative influence on performance.

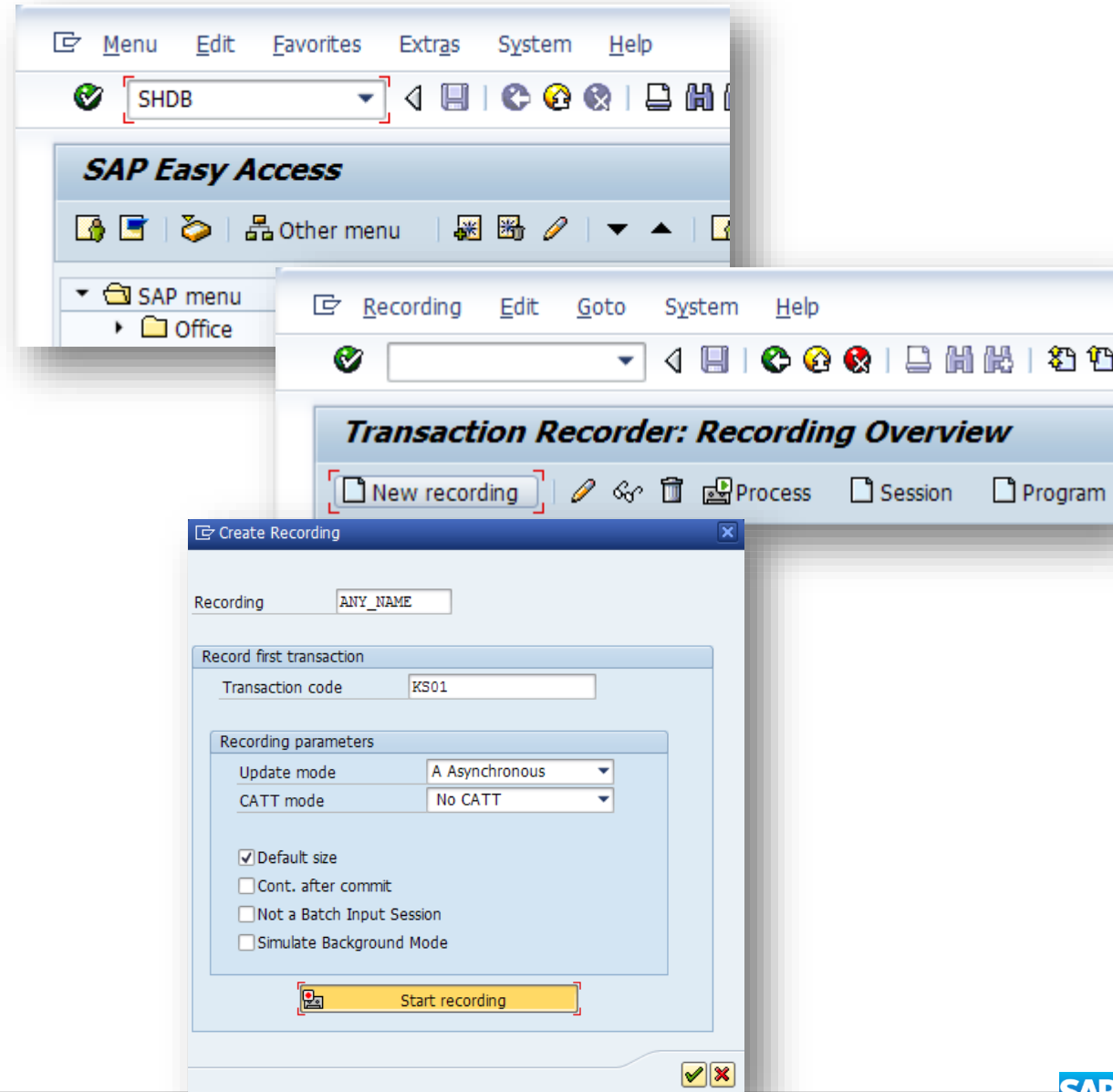
Important! E-CATT requires additional authorizations and has to be switched on on the mandant level. The users of the full version can use restricted BCC e-CATT access not requiring additional authorizations and mandant level switch.

- When recording a transaction script with the Transaction Recorder (SHDB transaction) one should choose:
 - **For Standard mode – the default setting should be left**
 - **For Extended mode – e-CATT with individual screen control + No Batch Input + Cont. after commit settings should be chosen**
- Which mode should one choose?
 - In most cases the Standard mode should be chosen (as this mode is usually more efficient for data processing)
 - In some cases (e.g. HR transactions) the Extended mode will be the only option (example of such transactions: PA30, PA40, SU01, SU10, PFCG, CSOI)
 - In some cases (e.g. some MM transactions) the Standard mode will be the only option (example of such transactions: ME21N, ME22N, ME52N, ME52N, CA01, MIGO)
 - When reading is needed one should use the Extended mode (as only the extended mode supports reading)

How to Start a Recording with Transaction Recorder (SHDB)




- In SAP GUI in the command field type **SHDB** and press Enter to start the Transaction Recorder
- In the Transaction Recorder click the **New Recording** button (the Create Recording popup window will appear)
- On the Create Recording popup window put the recording parameters:
 - Recording – any recording ID
 - Transaction Code – ID of transaction to be recorded
 - Other parameters should be left as default for Standard mode or can be changed for Extended mode
- Click Start recording button to start



Tips For Recording with Transaction Recorder (SHDB)



- **F1 and F4 help actions will not be recorded, so they can be freely used during recording**
- **The recording should be done without errors. If you encounter an error during recording, you'd better start the recording once more.**
- **The screen fields, one enter data to, during recording will be saved in the script. Be sure to enter or change data in all the screen fields that should be recorded in the script.**
- The recording should be done twice (once for the **actual mode** and the second time for the **test mode** with Cancel icon  pressed at the end of the transaction).
It is assumed however that most important part of the actual recording will be reused for test mode. Thus, in fact in the test mode recording only the last screens recorded are important.
- In Screen Painter screens, movements in the scrollbar are not recorded. Use the function keys or **tab** for positioning!

BDC Script Elements 1/2



Recorded transaction ID at the beginning of the script

Line	Program	Screen	S...	Field name	Field value
1			T	KS01	
2	SAPLKMA1	0260	X		
3				BDC_CURSOR	CSK5Z-DATAB_ANFO
4				BDC_OKCODE	/00
5				CSK5Z-KOKRS	0001
6				CSK5Z-KOSTL	ZZZZ
7				CSK5Z-DATAB_ANFO	08.10.2014
8				CSK5Z-DATBI_A	31.12.9999
9	SAPLKMA1	0299	X		
10				BDC_OKCODE	
11				BDC_SUBSCR	0300SUBSCR
12				BDC_CURSOR	CSK5Z-KTIE
13				CSK5Z-KTIE	Name
14				CSK5Z-LTIE	Long Name
15				CSK5Z-VERSER	SIWIECK
16				CSK5Z-VERSER	Krzysztof Siwiec
17				CSK5Z-AN	IT
18				CSK5Z-AN	E
19				CSK5Z-AN	0001-1
20				CSK5Z-AN	0001
21				CSK5Z-AN	0001
22				CSK5Z-AN	EUR
23				CSK5Z-AN	1310

Program/Screen pair begins each screen block of script. The lines below, until next program/screen pair, describe that screen.

BDC_CURSOR means cursor positioning. For standard scripts these script rows are irrelevant (they can be used in extended scripts for reading data)

BDC_OKCODE depicts command executed when leaving the screen. There is always one BDC_OKCODE per screen block. Most common values are:


- /00 – Enter
- =BU – Save
- /ECNC – Cancel
- =YES – pressing YES button
- =NO – pressing NO button

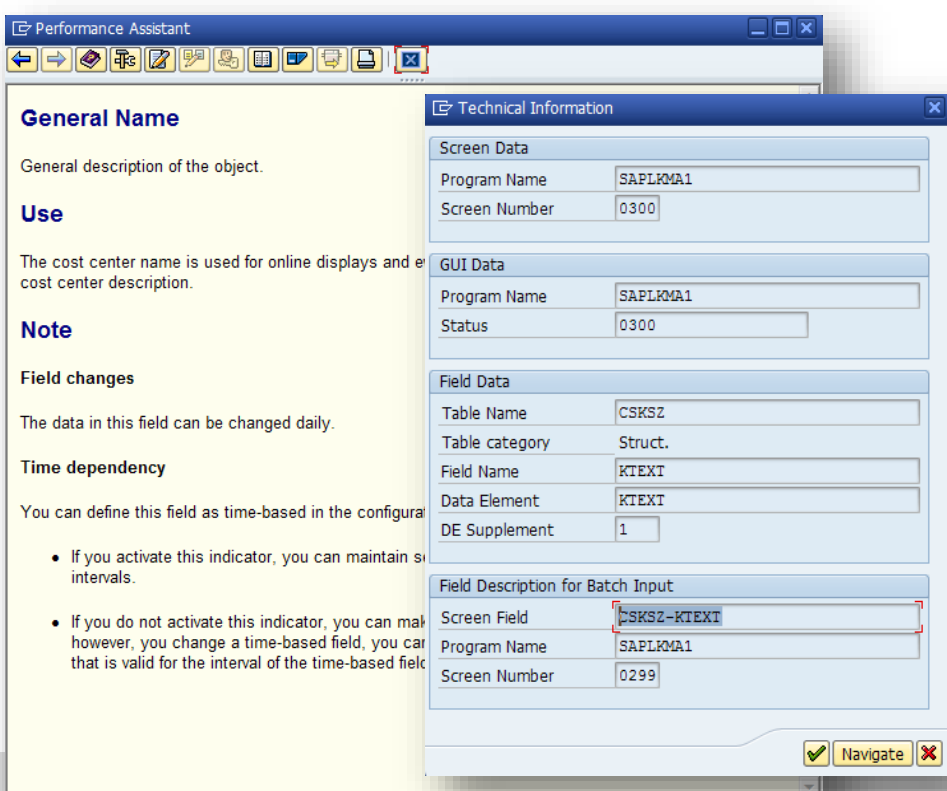
BDC_SUBSCR means subscreens of the screen. These rows of the script should not be changed.

Screen field references are used to pass values to screen (pass data to SAP). They have usually form **Table-Field**.

BDC Script Elements 2/2



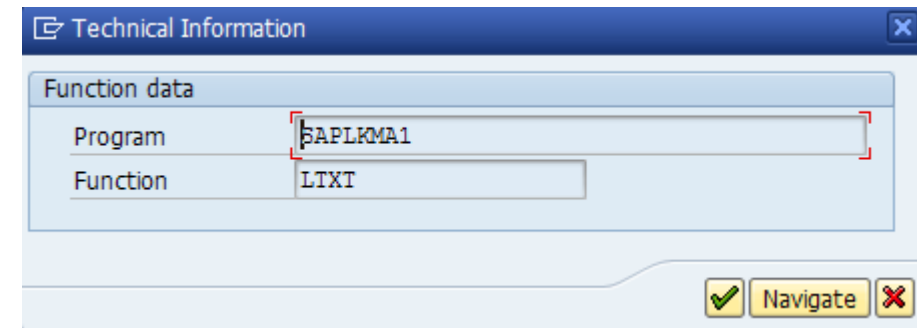
- One can check screen field TABLE-ID for each transaction (by selecting the field, pressing F1 and then looking at technical information )
- One can check command (function) attached to screen buttons (by selecting the button and pressing F1)



The screenshot shows the SAP Performance Assistant interface. On the left, there is a 'General Name' and 'Use' section. The main area displays the 'Technical Information' dialog box for a screen field. The dialog is divided into several sections:

- Screen Data:** Program Name: SAPLRMA1, Screen Number: 0300
- GUI Data:** Program Name: SAPLRMA1, Status: 0300
- Field Data:** Table Name: CSKSZ, Table category: Struct., Field Name: KTEXT, Data Element: KTEXT, DE Supplement: 1
- Field Description for Batch Input:** Screen Field: CSKSZ-KTEXT, Program Name: SAPLRMA1, Screen Number: 0299

At the bottom right of the dialog, there are 'Navigate' and 'Close' buttons.



The screenshot shows the 'Technical Information' dialog box, specifically the 'Function data' section. It displays the following information:

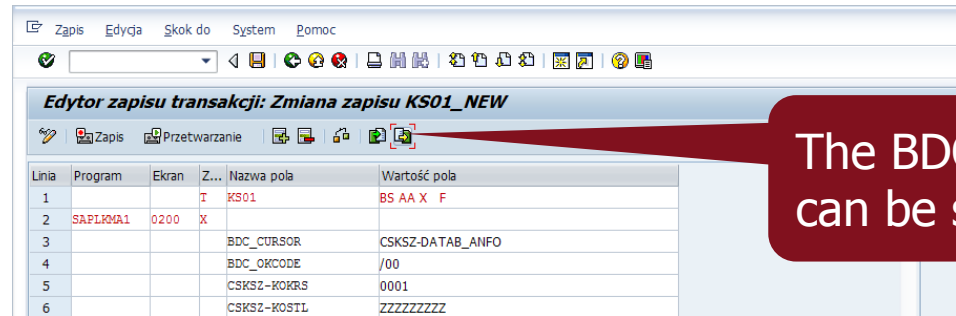
Function data	
Program	SAPLRMA1
Function	LTXI

At the bottom right of the dialog, there are 'Navigate' and 'Close' buttons.

Saving Recording to the Text File and Importing it to EasyInput

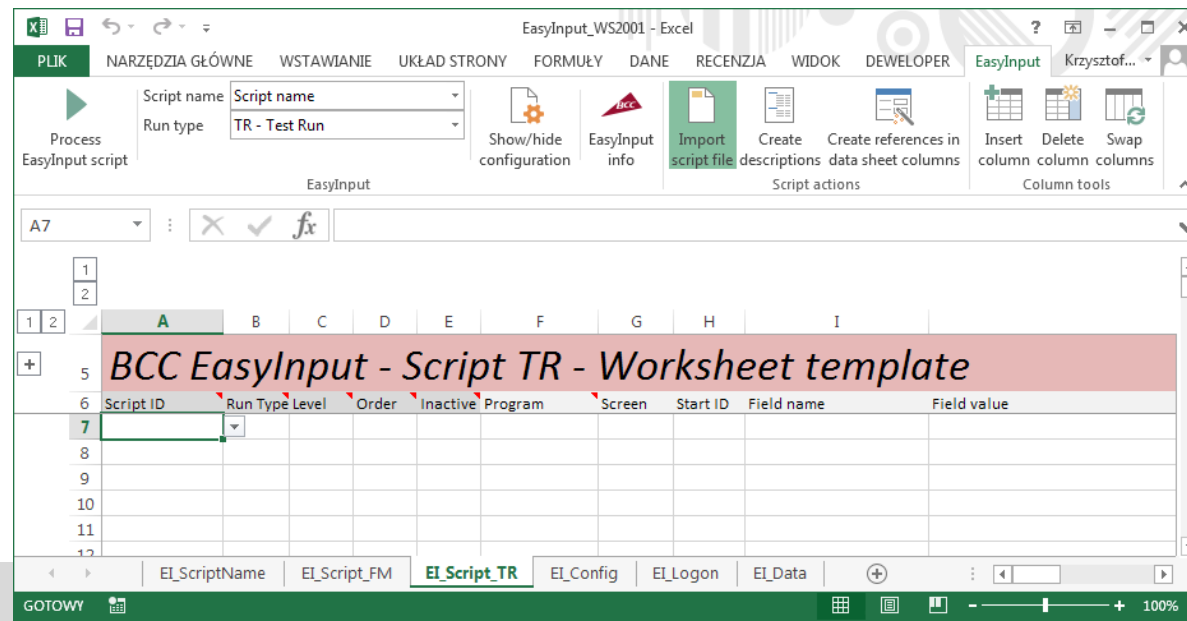


- After recording is finished the screen showing the script is presented. The user can save it to the text file.



The BDC (transaction) script can be saved to file.

- The text file can be then imported in EasyInput workbook by pressing the *Import script file* icon.



- In order to be able to run a transactional EasyInput script in actual and test mode one has to record respective script lines.
- On the EI_Script_TR in column **Run Type** one can assign to each script row whether:
 - It is used only for actual run - A
 - It is used only for test run - T
 - It is used for both test and actual run - C
- **In order to facilitate the maintenance it is recommended to make most of the script common (assign it to Run Type C).** Only the lines that has to be different should be assigned either to Actual (A) or Test (T) Run Types.
- In most scripts only the last screens need to be different (the screen ending with save function =BU for actual run and another ending with cancel /ECNC for test run)

Actual and Test Run Recordings 2/3



- In most cases the ending of the script should be similar to that shown on the picture below.
- In order not to have any fields transferred only in actual run (not tested in the test run) there should be no fields transfer on the ending screen used only for actual run. To achieve this one should **press Enter before clicking the Save button**. Pressing Enter will generate additional program/screen block in the recording. This program/screen block will differ only with BDC_OKCODE value from the last screen of the recording (/00 instead of =BU value).
- Since the last actual screen will be a copy of the screen before one can remove (inactivate – X in column E) any TABLE-FIELD lines from it.

Script ID	Run Type (A/T/C)	Level	Order	Inact	Program	Screen	Start ID	Field name	Field value
61	FB50	C - All run types	1					ACGL_ITEM-BEWAR (01)	
62	FB50	C - All run types	1					ACGL_ITEM-SGTXT (01)	
63	FB50	C - All run types	1					ACGL_ITEM-MARKSP (01)	X
64	FB50	A - Actual run			SAPMF05A	1001	X	BDC_OKCODE	=BU
65	FB50	A - Actual run	0					BDC_SUBSCR	SAPMF05A 1010HEAD
66	FB50	A - Actual run	0					BDC_SUBSCR	SAPMF05A 1300APPL_SUB_I
67	FB50	A - Actual run	0					BDC_SUBSCR	SAPLSEXM 0200APPL_SUB
68	FB50	A - Actual run	0					BDC_SUBSCR	SAPLFSKB 0100ITEMS
69	FB50	A - Actual run	0					BDC_CURSOR	ACGL_ITEM-HKONT (01)
70	FB50	A - Actual run	0	X					
71	FB50	T - Test run	0		SAPMF05A	1001	X		
72	FB50	T - Test run	0					BDC_OKCODE	/ECNC
73	FB50	T - Test run	0		SAPLSP01	0200	X		
74	FB50	T - Test run	0					BDC_OKCODE	=YES

Actual and Test Run Recordings 3/3



FB50	C - All run types	1		SAPMF05A	1001	X													
FB50	C - All run types	1					BDC_OKCODE	=0005											
FB50	C - All run types	1					BDC_SUBSCR	SAPMF05A		1010HEAD									
FB50	C - All run types	1					BDC_SUBSCR	SAPMF05A		1300APPL_SUB_T									
FB50	C - All run types	1					BDC_SUBSCR	SAPLSEXM		0200APPL_SUB									
FB50	C - All run types	1					BDC_SUBSCR	SAPLFSKB		0100ITEMS									
FB50	C - All run types	1					ACGL_ITEM-HKONT(01)	400000				G/L							
FB50	C - All run types	1					ACGL_ITEM-SHKZG(01)	S				Debit/Credit	WriteData						
FB50	C - All run types	1					ACGL_ITEM-WRBTR(01)	10000				Amount	WriteData						
FB50	C - All run types	1					ACGL_ITEM-MWSKZ(01)	V0				Tax code	WriteData						
FB50	C - All run types	1					ACGL_ITEM-KOSTL(01)	CO01				Cost Center	WriteData						
FB50	C - All run types	1					ACGL_ITEM-AUFNR(01)					Order	WriteData						BM
FB50	C - All run types	1					ACGL_ITEM-MARKSP(01)	X											
FB50	C - All run types	1					BDC_CURSOR	ACGL_ITEM-HKONT(01)				G/L							
FB50	A - Actual run	0		SAPMF05A	1001	X													
FB50	A - Actual run	0					BDC_OKCODE	=BU											
FB50	A - Actual run	0					BDC_SUBSCR	SAPMF05A		1010HEAD									
FB50	A - Actual run	0					BDC_SUBSCR	SAPMF05A		1300APPL_SUB_T									
FB50	A - Actual run	0					BDC_SUBSCR	SAPLSEXM		0200APPL_SUB									
FB50	A - Actual run	0					BDC_SUBSCR	SAPLFSKB		0100ITEMS									
FB50	A - Actual run	0		X			ACGL_ITEM-HKONT(01)	400000				G/L							
FB50	A - Actual run	0		X			ACGL_ITEM-SHKZG(01)	S				Debit/Credit							
FB50	A - Actual run	0		X			ACGL_ITEM-WRBTR(01)	10000				Amount							
FB50	A - Actual run	0		X			ACGL_ITEM-MWSKZ(01)	V0				Tax code							
FB50	A - Actual run	0		X			ACGL_ITEM-KOSTL(01)	CO01				Cost Center							
FB50	A - Actual run	0		X			ACGL_ITEM-AUFNR(01)												
FB50	A - Actual run	0		X			ACGL_ITEM-MARKSP(01)	X											
FB50	A - Actual run	0		X			BDC_CURSOR	ACGL_ITEM-HKONT(01)				G/L							
FB50	T - Test run	0		SAPMF05A	1001	X													
FB50	T - Test run	0					BDC_OKCODE	/ECNC											
FB50	T - Test run	0		SAPLSPO1	0200	X													
FB50	T - Test run	0					BDC_OKCODE	=YES											

The screen before the last actual saving screen is a copy of the last screen. Usually forced by pressing Enter before saving (in this case by pressing insert line button =0005)

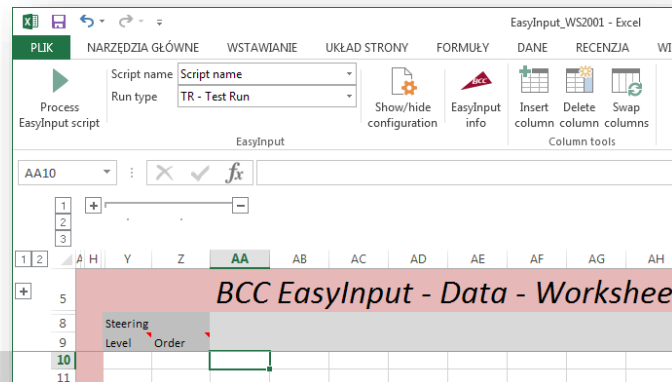
Since the last actual screen will almost identical to the screen before (apart from BDC_OKCODE command), one should remove (inactivate – X in column E) any TABLE-FIELD lines from it. Otherwise they would overwrite the values passed earlier.

- Level/Order columns may be left empty for simple scripts, where no header/ item/ subitem dependencies exist.
- **Level column serves looping** over document/ master data subentries (e.g. line items). Rows assigned to level 0 (or empty) will be taken once in the resulting script (e.g. header lines). Rows with level 1 will be used as many times as there is lines of assigned type (e.g. line items) on the data worksheet. Thus when recording one should take into account recording only one document line item in such a way, that this part of script could be repeated. Depending on the transaction it requires various techniques. For many transactions inserting always blank row and than entering line item always in the first row of the document is a good solution.

Level/ Order Columns in EasyInput Script



- If a document has further sublevel for each line item (e.g. conditions in SD for each line item) next level number can be used (2 in this example)
- **Order can be used if two different sublevel exists** (e.g. Roles and Profiles for a user in transaction SU01). Than one sublevel can be assigned level/order pair 1/a and the second 1/b.
- The level/order pair defined on the script is usually automatically assigned to data rows on EI_Data worksheet (Y/Z columns). The system use linking data column with script (defined in script) to detect the right level/order before processing the data. For some scripts however, where the data columns from various level/order overlap EasyInput may not be able to assign data row to a given level/order. Then manual assignments in columns Y/Z of data row is needed.



Line Activity Formulas 1/4



- **Line activity formulas** is the name of an additional functionality contained in EasyInput, that is configured in columns P-Z and sometimes used in column J (field value) of the EI_Script_XX worksheet.

Field value	Field desc	Transfer type	Read/Write Column	Omitting Screen	Col. 1	Col. 2	Col. 3	Col. 4	Condition (True/False)	Value 1	Value 2	Value 3	Value 4	Read value	Loop index
/00															2
21.04.2014	Document Date	WriteData	AE												2
SA	Document Type	WriteData	AC												2
0001	Company Code	WriteData	AB												2
21.04.2014	Posting Date	WriteData	AD												2
eur	Currency	WriteData	AF												2
	Exchange rate	WriteData	AG												2
	Document type														2
40	Posting key	WriteData	AH												2
400000	Account	WriteData	AI												2
SAPMFO5A	1300APPL_SUB_T														2
SAPLSEXM	0200APPL_SUB														2
															2
															2
BSEG-MWSKZ	Tax code														2
/00															2
10000	Amount	WriteData	AJ												2
00	Tax code	WriteData	AK												2
															2
															2

- Line activity formulas are used to:
 - **Make a certain script line conditional (T)**
 - **Calculate a value of the field value column (J)**
- Line activity formulas can be based on:
 - Data contained in the data worksheet (by default EI_Data)
 - Technical data contained in column Z - Loop Index
 - Data read in the script (ReadData option, Y)

■ Referencing data contained in the data worksheet

Each script line is assigned to a certain level/order. When processing a Data line assigned to the a level/order all script lines with the same level/order are processed from top to bottom in the active script. For each script line if columns P/Q/R/S are filled with references to columns in Data sheet (e.g. AA, AB,...), relevant data sheet values are copied to columns U/V/W/X. Thus one can create an Excel formula in the line based on data in columns U/V/W/X, as these columns are filled with data in runtime. Remember that, the script is processed from the top down, so if one reads a value in the first line of a certain level/order one can reuse it in formulas in the lines below with the same level/order or with higher level (without filling the references in columns P/Q/R/S in these lines).

Important: From version 2.17 B1009 on it is possible to read data row number. Just put string ROW in columns P/Q/R/S.

■ Referencing the technical data contained in column Z - Loop Index

Sometimes in the formula one has to make a distinction between the first document line and the others. For this purpose Loop Index value is filled in column Z of the script worksheet in the runtime. Loop Index column contains for lines on level 0 the number of consecutive transaction processed, and for the lines on higher levels, the number of a consecutive data line (e.g. Line item number) processed within a transaction.

■ Referencing data read in the script (ReadData option, Y)

If a script contains several transactions/ function modules then the result read in one can be used in the next. Apart from copying the data read with the ReadData statement to the Data worksheet, the data is also copied to column Y of the script worksheet to the relevant line. Important! Since reading data is performed **after** the transaction/ function module is called, its result from column Y cannot be used when a script contain only one transaction/function module!

■ Make a certain script line conditional (T)

Apart from static inactivation in column E of the script, a dynamic inactivation of the line can be performed in the column T. If the script is not inactivated in column E, then:

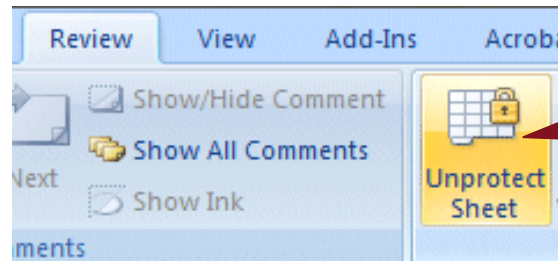
- If there is no formula in column T, the script line is active
- If there is formula returning true in column T, the script line is active
- If there is formula returning false in column T, the script line is inactive
- If the script is inactivated in column E, then the line is inactive and no other checks are performed.

■ Calculate a value of the field value column (J)

Value passed to script line can be passed as:

- A constant put into J column
- A variable read from the Data worksheet with the statement WriteData and the reference to the data column (L/M)
- A formula put into J column

In the third case one has to take off the script worksheet protection and change the formatting of the cell to general, so that Excel treats the formula as the formula and not as a string. In the formula one can reference the fields filled in the runtime in columns U/V/W/X and in the special cases in column Y.

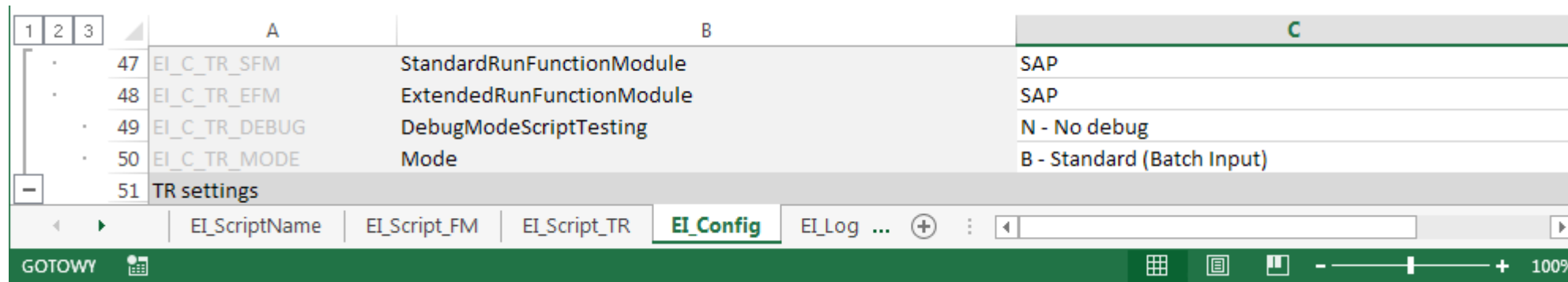


Use standard MS Excel ribbon menu to unprotect a worksheet.

Transaction Script Testing



- Once the transaction script is created, it is possible to test it. By testing, one can check the parameters passed to the Batch Input Call and even see the subsequent SAP GUI screens called by the script. In order to switch on the testing mode set the „DebugModeScriptTesting“ configuration switch on the EI_Config worksheet.
- In test mode during script run the additional worksheet EI_Debug is created and all the parameters passed to the Batch Input are shown there. Additionally either all or only the erroneous SAP GUI screens are presented helping in script debugging.



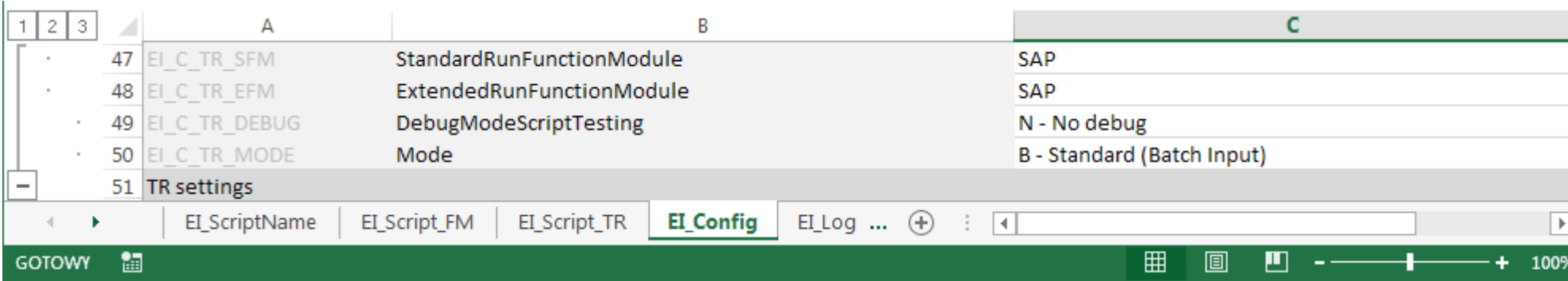
The screenshot shows a table with three columns: A, B, and C. The table contains the following data:

	A	B	C
·	47 EI_C_TR_SFM	StandardRunFunctionModule	SAP
·	48 EI_C_TR_EFM	ExtendedRunFunctionModule	SAP
·	49 EI_C_TR_DEBUG	DebugModeScriptTesting	N - No debug
·	50 EI_C_TR_MODE	Mode	B - Standard (Batch Input)
-	51 TR settings		

The screenshot also shows a navigation bar with tabs: EI_ScriptName, EI_Script_FM, EI_Script_TR, EI_Config (selected), EI_Log ...

Transaction Script Default Mode

- In EasyInput there are two different script modes that can be used to record a transaction:
 - Standard (Batch Input) – This is the most commonly used mode. This mode operates in BDC mode within SAP so screens might look different from manual entry. Standard mode can only upload data into SAP. This mode does not need e-CATT to be working in the SAP environment and it's performance is better as the extended mode.
 - Extended (Non-Batch Input, e-CATT) – This mode also operates in BDC mode within SAP but allows you to read data from SAP in addition to uploading data. Additionally in this mode the Batch Input marker is not set, so that the screen are usually identical to manual entry. Important! Reading data from transaction has negative influence on performance.
- The default mode is set in the TR configuration settings (*Mode*)
It can be also changed on the Script level (on the EI_ScriptName worksheet).



Standard/ Extended Mode Function Modules (full version only)



- In configuration on EI_Config worksheet in the TR settings part the full version user can select whether to use SAP standard function modules or these provided by BCC with the full version. The modules provided by BCC can be better in case SAP changes the interface of their own modules and do not require e-CATT switch on the mandant level.
- In case of full version of EasyInput the change from "SAP" to "BCC" is possible for *StandardRunFunctionModule* and *ExtendedRunFunctionModule* settings.

The screenshot shows the Excel interface for the 'EI_Config' worksheet. The 'TR settings' row (row 51) is highlighted, showing the following configuration:

	A	B	C
47	EI_C_TR_SFM	StandardRunFunctionModule	SAP
48	EI_C_TR_EFM	ExtendedRunFunctionModule	SAP
49	EI_C_TR_DEBUG	DebugModeScriptTesting	N - No debug
50	EI_C_TR_MODE	Mode	B - Standard (Batch Input)
51	TR settings		

The 'EasyInput' ribbon is visible at the top, showing options for 'Script name', 'Run type' (set to 'TR - Test Run'), 'Show/hide configuration', and 'EasyInput info'. The 'Worksheet template' dropdown is set to 'C21'.

www.bccgroup.eu/extra

Free trial version
Order online
Other BCC EXTRA products