

BC SAPscript Raw Data Interface



HELP.BCSRVSCRRDI

Release 4.6C



Copyright

© Copyright 2001 SAP AG. All rights reserved.

No part of this publication may be reproduced or transmitted in any form or for any purpose without the express permission of SAP AG. The information contained herein may be changed without prior notice.

Some software products marketed by SAP AG and its distributors contain proprietary software components of other software vendors.

Microsoft[®], WINDOWS[®], NT[®], EXCEL[®], Word[®], PowerPoint[®] and SQL Server[®] are registered trademarks of Microsoft Corporation.

IBM[®], DB2[®], OS/2[®], DB2/6000[®], Parallel Sysplex[®], MVS/ESA[®], RS/6000[®], AIX[®], S/390[®], AS/400[®], OS/390[®], and OS/400[®] are registered trademarks of IBM Corporation.

ORACLE[®] is a registered trademark of ORACLE Corporation.

INFORMIX[®]-OnLine for SAP and Informix[®] Dynamic Server[™] are registered trademarks of Informix Software Incorporated.

UNIX[®], X/Open[®], OSF/1[®], and Motif[®] are registered trademarks of the Open Group.






HTML, DHTML, XML, XHTML are trademarks or registered trademarks of W3C[®], World Wide Web Consortium, Massachusetts Institute of Technology.

JAVA[®] is a registered trademark of Sun Microsystems, Inc.

JAVASCRIPT[®] is a registered trademark of Sun Microsystems, Inc., used under license for technology invented and implemented by Netscape.

SAP, SAP Logo, R/2, RIVA, R/3, ABAP, SAP ArchiveLink, SAP Business Workflow, WebFlow, SAP EarlyWatch, BAPI, SAPPHIRE, Management Cockpit, mySAP.com Logo and mySAP.com are trademarks or registered trademarks of SAP AG in Germany and in several other countries all over the world. All other products mentioned are trademarks or registered trademarks of their respective companies.

Icons

Icon	Meaning
	Caution
	Example
	Note
	Recommendation
	Syntax

Inhalt

BC SAPscript Raw Data Interface	5
SAPscript Raw Data Interface	6
Concept and Usage	7
Prerequisites and Basic Conditions	8
Programming Interface for Printing	9
Output Stream (Output Mode Spool).....	10
Header Record Structure	11
Sort Record Structure	12
Data Record Structure	13
Control Record Structure	14
Output Stream (Output Mode IDOC).....	15
SAPscript Control Statement ADDRESS	16
Filling the Internal Sort Fields.....	18

BC SAPscript Raw Data Interface

SAPscript Raw Data Interface

Concept and Purpose

[Definitions of the SAP Online Documentation \[Extern\]](#)

[Concept and Usage \[Seite 7\]](#)

[Prerequisites and Basic Conditions \[Seite 8\]](#)

[Programming Interface for Printing \[Seite 9\]](#)

[Output Stream \(Output Mode Spool\) \[Seite 10\]](#)

[Output Stream \(Output Mode IDOC\) \[Seite 15\]](#)

[SAPscript Control Statement ADDRESS \[Seite 16\]](#)

[Filling the Internal Sort Fields \[Seite 18\]](#)

Concept and Usage

Connecting External Text Management Systems

Using the SAPscript Raw Data Interface (RDI), you can connect external text management systems to master individual requests, for example, optimizing postage expense. The interface contains all data of the R/3 forms, but no layout information such as font or page size. The external system alone is responsible for layout and administration of the document data.

The RDI is of special importance for automatic addressing and postaging: The external system sorts the data from the interface and forwards the documents accordingly.

No Integration in the R/3 System

The Raw Data Interface is a certified interface and allows easy connection of external systems. However, this extra functionality demands its price: You lose the tight integration within the R/3 System. For example, there is no way to find out from within an R/3 application whether the external system successfully printed and sent the documents. In addition, each adaptation of standard forms causes extra expenses, since the system has to adapt an internal as well as an external form. And the external tool is no integral part of the ABAP Workbench, so that field information from the ABAP Dictionary (field type, output length, and so on) is not available.

Prerequisites and Basic Conditions

- **R/3 Customizing for printing via RDI**

In the R/3 System, you can flag a form for "external printing". If a form is not flagged explicitly, application-dependent customizing settings apply.
- **Output device for external printing**

You must create at least one "output device" for the device type for external printing delivered by SAP.
- **Defining the external forms and allocating them to the R/3 forms**

For each R/3 form you want to print externally, you must enter form definitions in the external text system, particularly define the appropriate variables.
- **External program for reading the data**

The external program reads the data according to the specification ([Output Stream \[Seite 10\]](#)) and processes and prints the documents.
- **Status administration for RDI data**

For all documents you want to print externally, you should be able to display the current print status. In addition, you need an overview of all unsuccessful print requests.
- **Optional: archiving**

If required, the external system should be able to deviate the print output into an archive.

Programming Interface for Printing

Initializing the Form Printout

To initialize the printing of forms via the Raw Data Interface, use function modules (OPEN_FORM und CLOSE_FORM) and a flag in the form (administration data). For this purpose, the function modules include the additional optional import parameter RAW_DATA_INTERFACE.

The Import Parameter RAW_DATA_INTERFACE

The values allowed for RAW_DATA_INTERFACE are 'X' (RDI), 'Space' (SAPscript form printout), 'I' (IDOC), and '*' (default value). By default, the parameter is set to '*'. In this case, the flag in the form (form maintenance transaction SE71) determines, whether to print via the RDI or not. With OPEN_FORM, you can change the option for the current printout. If you set the parameter RAW_DATA_INTERFACE to 'X', the system prints via the Raw Data Interface, if you set it to 'Space', it doesn't.

If you do not specify a form with OPEN_FORM, you must at a later time call START_FORM. In this case, the form specified with START_FORM determines where to print, if for OPEN_FORM as well as for START_FORM the parameter RAW_DATA_INTERFACE is set to '*'. If you use 'X' or 'Space' with START_FORM, you can change the value set with the form, as described for OPEN_FORM. As with SAPscript form printing, you must always call the function module END_FORM after using START_FORM.

Switching Between RDI and SAPscript Form Printing

Since all documents printed after OPEN_FORM and before CLOSE_FORM are stored in one print request, you cannot switch between RDI and SAPscript form printing before reaching CLOSE_FORM. If you try to switch at START_FORM the system terminates processing and sends an A-type error message.

Compatibility

Downward compatibility is guaranteed, since the default values for the function modules let the form parameter determine whether to use the RDI or not. If the parameter in the form is not set, the system automatically triggers the "normal" SAPscript form printing.

You can display the output in the spool (transaction SP01). Each document starts with a header record, followed by any number of data records. The structure of these data records is described below.

Instead of using the spool, you can also transfer data as IDOC. However, only the print program can initiate this. It must set the parameter RAW_DATA_INTERFACE to 'I' ([Output Mode IDOC \[Seite 15\]](#))

Output Stream (Output Mode Spool)

Output Stream (Output Mode Spool)

The output stream consists of header record, data records, sort records, and control records. Each record starts with a flag (1 byte) that determines the record type: header [H], data [D], sort [S], and control [C].

[Header Record Structure \[Seite 11\]](#)

[Sort Record Structure \[Seite 12\]](#)

[Data Record Structure \[Seite 13\]](#)

[Control Record Structure \[Seite 14\]](#)

Header Record Structure

Each document starts with the 'H' flag that introduces the header (structure STXRDIH).

- RDI version (length 6) (the current version is 040A01)
- client (length 3)
- document number (length 10) (also return value of function module CLOSE_FORM: RESULT-TDSPOOLID)
- language (length 1)
- form name (length 16)
- device type (length 8), for example, PRINTER, SCREEN,...
- name of the computer from which the processing was started (length 64)
- background processing mode (length 1), values 'X' or Space

Then follow the data of structure ITCPO (SAPscript output interface).



Note the length specifications of the different fields. For integer types (INT1, INT2, INT4), it must not necessarily be the length defined in the ABAP Dictionary, but can also be the output length specified in the domain.

Sort Record Structure

Sort Record Structure

After the header record, the sort fields follow (structures ITCSNDSORT, ITCRCVSORT), initiated by 'S'. There are ten internal and five external sort fields of length 32.

[Filling the Internal Sort Fields \[Seite 18\]](#)

Data Record Structure

After the sort record, the actual data records follow, started by 'D'.

- name of the form window (length 8)
- flag for the start of a new main window (length 1), values X or Space
- flag for the start of a text element (length 1), values X or Space
- name of the text element (length 30)
- name of the symbol (length 130)
- succession flag (length 1), values X or Space
- occupied length (length 3)
- value of the symbol (length up to 255)



At the start of a text element, the corresponding flag is set. Since usually a text element covers several data records, this flag is required for a unique allocation, if WRITE_FORM is used to print text elements several times in direct succession. This is of special importance when using the SAPscript control statement NEW-WINDOW.



When printing via RDI, a page break must be triggered explicitly either by the form (NEW-PAGE statement) or by the print program (function module CONTROL_FORM). The external tool recognizes a page break by means of a control record (as described below).

Symbol Name - Symbol

The *symbol name* identifies a symbol within the text element. It can be one of the symbols supported by SAPscript. For normal text, the *symbol name* is empty (Space). After the *symbol name*, the *succession flag* follows. If it is set (X), then the value of the subsequent data record belongs to the symbol. The *occupied length* indicates the number of characters that belong to the symbol. It includes trailing blanks that are part of the symbol. If the value of a symbol covers several data records, the *succession flag* is set and the *occupied length* always contains 255 (maximum length of a data record). Then follows the *value of the symbol* up to the length specified in *occupied length*. This dynamic output allows to use only as many characters as are actually needed, instead of outputting a given number even though only few characters are occupied.



The *value of a symbol* is output according to the specified formatting options. If pre-text or post-text exist, they are included into the *value of the symbol*. Symbols occurring in the pre-text or post-text are replaced, their values being allocated to the main symbol.

Control Record Structure

Control Record Structure

The data records can be interrupted by a control record. A control record contains important information for interpreting the data. It starts with the flag 'C', followed by the control information. This information is stored as character string in the form KEY WORD VALUE.

At present, the control record is used for the following information:

Codepage and Language

After header and sort record and before the first data record, always a control record of this type follows. If you use the SAPscript statement INCLUDE to include a text in a different language, the system writes the control record containing the corresponding codepage and language before writing the data record containing the include text. At the end of the included text, a reset may be necessary, which is also indicated in the control record.

The key words are CODEPAGE and LANGUAGE.



Control record for a German text with codepage 1100:

```
CCODEPAGE 1100 LANGUAGE DE
```

The first 'C' introduces the control record.

Page Name

At the beginning of a new page, specify its name using the key word PAGENAME.



Control record for page FIRST:

```
CPAGENAME FIRST
```

The first 'C' introduces the control record.

Output Stream (Output Mode IDOC)

Data can also be printed as IDOC (see [Programming Interface for Printing \[Seite 9\]](#)). The name of the corresponding basis type is SAPRDI01. It consists of the segments E1RDIH (header), E1RDI_BODY (dummy segment, see below), E1RDIC (control record), E1RDIS (sort record), and E1RDID (data record). The logical message type is SAPRDI, the partner type is LS. In addition, you must maintain in table T000 the logical system for the appropriate client.

Record Structure

The structures of header, control, sort, and data records are the same as described in [Output Stream \(Output Mode Spool\) \[Seite 10\]](#). However, the flags 'H', 'C', 'S', and 'D' are missing. For internal reasons, the segment E1RDI_BODY encapsulates control, sort, and data segments. When interpreting the data, the system can ignore it. This dummy segment appears in front of each control record.

For output mode IDOC, the header (E1RDIH) does not contain the document number. The return structure RDI_RESULT of function module CLOSE_FORM returns this number (RDI_RESULT-DOCNUM).

Using START_FORM / END_FORM

If you use START_FORM / END_FORM to print several documents in one request, the system creates only one IDOC and writes a separate header for each document (as for output mode spool).

Status

The status of IDOC is set to 30, status text „IDOC is ready for dispatching (ALE service)“. Internally, the constant ALE_READY_FOR_DISPATCHING represents this status.

Processing Selected Intermediate Documents

Use program RSEOUT00 to start processing all selected intermediate documents.

SAPscript Control Statement ADDRESS

SAPscript Control Statement ADDRESS

When using the ADDRESS control statement, the processing order is a little different. First, the system outputs general information (ABAP Dictionary structure STXADDRESS) that may be needed to actually format the address:

- STXADDRESS-TDPARAGRAPH (option PARAPGRAPH of the ADDRESS statement)
- STXADDRESS-TYPE (option TYPE of the ADDRESS statement), values 1, 2, or 3
- STXADDRESS-NUMBER (number of the address)
- STXADDRESS-FROM_COUNT (country key of the sender country)
- STXADDRESS-RECEIVER_L (country key of the recipient country)
- STXADDRESS-PRIORITY (priority rule, option PRIORITY of the ADDRESS statement)
- STXADDRESS-DELIVERY (flag whether the option DELIVERY was set for the ADDRESS statement ('X') or not (Space))
- STXADDRESS-ANZZL (option LINES of the ADDRESS statement)
- STXADDRESS-PERSONNUMB (person number)
- STXADDRESS-COUNT_IRL (flag whether country is in recipient language)
- STXADDRESS-LANG_COUNT (language for country)
- STXADDRESS-NO_UPPER (flag whether no uppercase for city)

Then immediately follows the structure of the actual address, depending on the type.

Type1	Type 2	Type 3
ADRS1-TITLE_TEXT	ADRS2-TITLE_PERS	ADRS3-TITLE_COMP
ADRS1-NAME1	ADRS2-NAME_PERS	ADRS3-NAME1
ADRS1-NAME2		ADRS3-NAME2
ADRS1-NAME3		ADRS3-NAME3
ADRS1-NAME4		ADRS3-TITLE_PERS
		ADRS3-NAME_PERS
		ADRS3-DEPARTMENT

This is followed by

ADRSx-NAME_CO
 ADRSx-STREET
 ADRSx-HOUSE_NUM1
 ADRSx-STR_SUPPL1
 ADRSx-STR_SUPPL2
 ADRSx-CITY1
 ADRSx-CITY2
 ADRSx-POST_CODE1
 ADRSx-POST_CODE2
 ADRSx-POST_CODE3

ADRSx-PO_BOX
ADRSx-PO_BOX_LOC
ADRSx-LOCATION
ADRSx-REGION
ADRSx-LAND1

where x equals 1, 2, or 3



If symbols are used within the ADDRESS statement, the system replaces them with their values. In the corresponding data record, as symbol name only the appropriate data field appears (for example, ADRS1-NAME1).

Filling the Internal Sort Fields

Filling the Internal Sort Fields

To fill the ten internal sort fields in the sort record, use the SAPscript statement `PERFORM`. Make sure to place the statement into the default text element of the main window before outputting any text. Only very few statements such as `DEFINE` may precede the `PERFORM` statement.

Pass the up to ten sort fields as `USING` parameters to the subroutine `RDI_FILL_INTERNAL_SORTFIELDS` in program `RSTXSORT`.

Example:

```
/: DEFINE &SORT01& = 'SAP'  
/: DEFINE &SORT02& = 'FI'  
/: PERFORM RDI_FILL_INTERNAL_SORTFIELDS IN PROGRAM RSTXSORT  
/: USING &SORT01&  
/: USING &SORT02&  
/: ENDPERFORM
```

This program fragment enters the values 'SAP' and 'FI' into the sort fields `SORT01` and `SORT02`. Each sort field can be up to 32 characters long.