



- SAP NetWeaver 2004s
- Version 63A
- Material number: 50093223

Copyright 2008 SAP AG. All rights reserved.

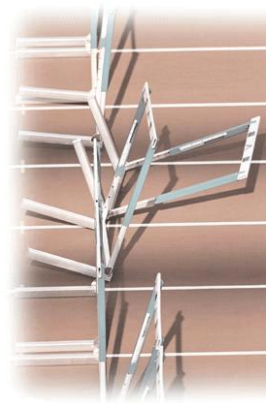
No part of this publication may be reproduced or transmitted in any form or for any purpose without the express permission of SAP AG. The information contained herein may be changed without prior notice.

© SAP AG 2008

- Some software products marketed by SAP AG and its distributors contain proprietary software components of other software vendors.
- Microsoft, Windows, Excel, Outlook, and PowerPoint are registered trademarks of Microsoft Corporation.
- IBM, DB2, DB2 Universal Database, OS/2, Parallel Sysplex, MVS/ESA, AIX, S/390, AS/400, OS/390, OS/400, iSeries, pSeries, xSeries, zSeries, System i, System i5, System p, System p5, System x, System z, System z9, z/OS, AFP, Intelligent Miner, WebSphere, Netfinity, Tivoli, Informix, i5/OS, POWER, POWER5, POWER5+, OpenPower and PowerPC are trademarks or registered trademarks of IBM Corporation.
- Adobe, the Adobe logo, Acrobat, PostScript, and Reader are either trademarks or registered trademarks of Adobe Systems Incorporated in the United States and/or other countries.
- Oracle is a registered trademark of Oracle Corporation.
- UNIX, X/Open, OSF/1, and Motif are registered trademarks of the Open Group.
- Citrix, ICA, Program Neighborhood, MetaFrame, WinFrame, VideoFrame, and MultiWin are trademarks or registered trademarks of Citrix Systems, Inc.
- HTML, XML, XHTML and W3C 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.
- MaxDB is a trademark of MySQL AB, Sweden.
- SAP, R/3, mySAP, mySAP.com, xApps, xApp, SAP NetWeaver, and other SAP products and services mentioned herein as well as their respective logos are trademarks or registered trademarks of SAP AG in Germany and in several other countries all over the world. All other product and service names mentioned are the trademarks of their respective companies. Data contained in this document serves informational purposes only. National product specifications may vary.

- The information in this document is proprietary to SAP. No part of this document may be reproduced, copied, or transmitted in any form or for any purpose without the express prior written permission of SAP AG.
- This document is a preliminary version and not subject to your license agreement or any other agreement with SAP. This document contains only intended strategies, developments, and functionalities of the SAP® product and is not intended to be binding upon SAP to any particular course of business, product strategy, and/or development. Please note that this document is subject to change and may be changed by SAP at any time without notice.
- SAP assumes no responsibility for errors or omissions in this document. SAP does not warrant the accuracy or completeness of the information, text, graphics, links, or other items contained within this material. This document is provided without a warranty of any kind, either express or implied, including but not limited to the implied warranties of merchantability, fitness for a particular purpose, or non-infringement.
- SAP shall have no liability for damages of any kind including without limitation direct, special, indirect, or consequential damages that may result from the use of these materials. This limitation shall not apply in cases of intent or gross negligence.
- The statutory liability for personal injury and defective products is not affected. SAP has no control over the information that you may access through the use of hot links contained in these materials and does not endorse your use of third-party Web pages nor provide any warranty whatsoever relating to third-party Web pages.

- To take part in this course, knowledge from the following training course is necessary:
 - BC 400 ABAP Workbench Basics



- **Participants:**
 - **Project team**
 - **Employees responsible for data processing**
 - **Employees responsible for organization**
- **Duration: 3 days**



© SAP AG 2003

Hints to the user:

- These training documents are not meant to be used **without corresponding classroom instruction. They are only complete when used in tandem with the comments of an SAP trainer.** Space is provided for you in the course documents to note this additional information.

- **Course goal**
- **Objectives**
- **Course content**
- **Course overview diagram**
- **Main business scenario**
- **Course introduction**

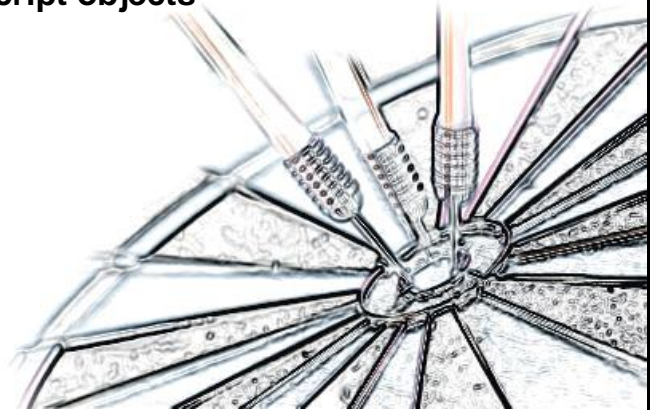


© SAP AG 1999

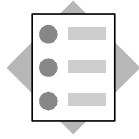


At the conclusion of this course, you will be able to:

- **Print forms from the SAP system**
- **Make changes to SAP standard forms and SAP print programs**
- **Transport SAPscript objects**



© SAP AG 2006



- To maintain forms using SAPscript tools
- To use SAPscript control statements and icons
- To control print output using print programs
- To be able to change print programs and forms in the SAP system
- To maintain fonts in the SAP system

© SAP AG 2006

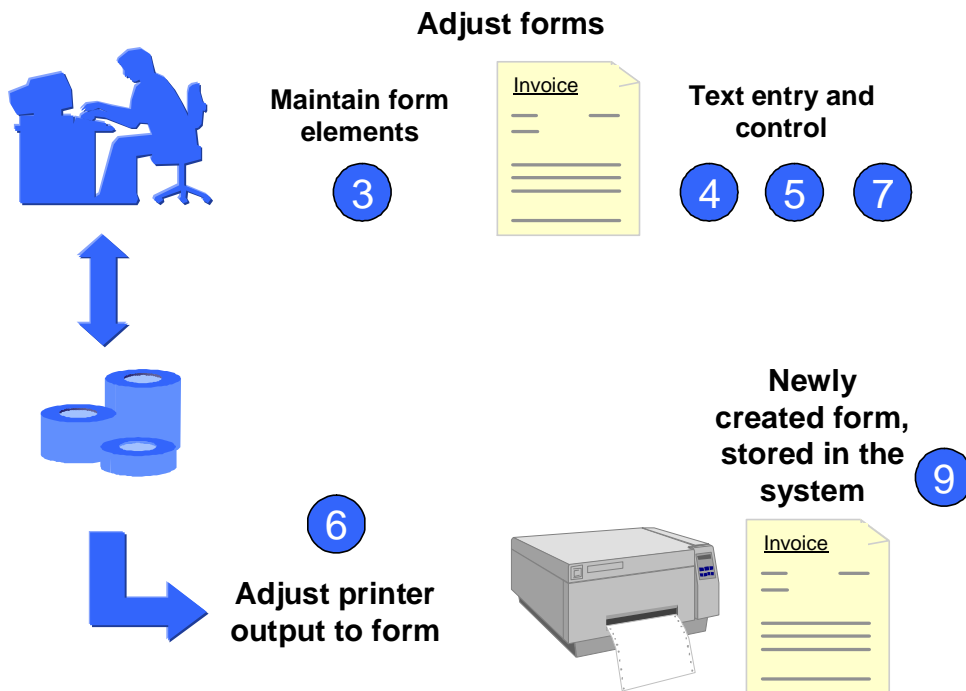
Preface

Unit 1	Introduction	Unit 6	Print Program
Unit 2	SAPscript Overview	Unit 7	Styles
Unit 3	Form Elements	Unit 8	The Next Generation
Unit 4	Graphical Editor and Line Editor	Unit 9	Modifications
Unit 5	Symbols and Control Statements	Unit 10	Font Maintenance

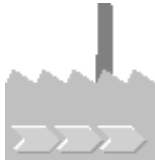
Exercises and their solutions can be found at the end of each chapter

Appendix

© SAP AG 2008



© SAP AG 2006



- **All course participants will perform the following print control actions on a form that they have changed:**
 - Changing the layout of a form
 - Changing the text in a form
 - Creating a print program and controlling the text output sequence
 - Reproducing a typical change scenario in the SAP system

© SAP AG 2006

Contents:

- **SAPscript in the SAP System**
- **The Purpose of SAPscript**
- **SAPscript Components**
- **Forms and Print Programs**
- **Process Flow**

© SAP AG 2006

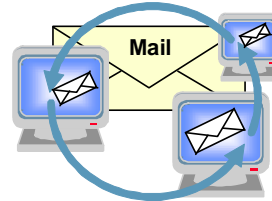


After completing this unit, you will be able to:

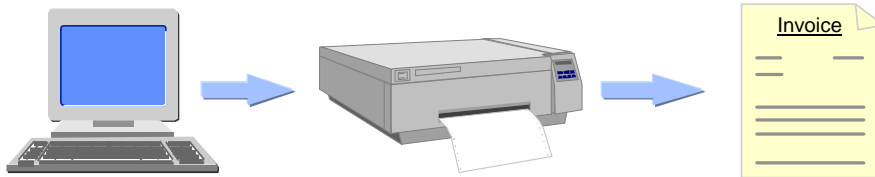
- **Understand how SAPscript fits into the overall structure**
- **Identify the integration of SAPscript in the SAP system**
- **Name the SAPscript components**
- **Understand how the components interact, in particular print programs and forms**

© SAP AG 2006

- Entering text

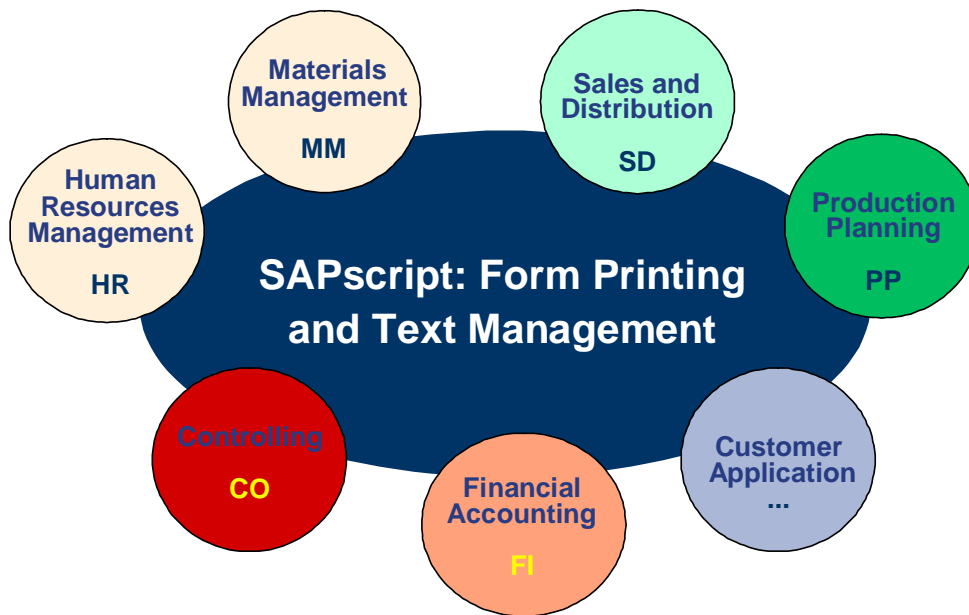


- Printing using forms



© SAP AG 2002

- Every company needs to routinely output documents with a uniformly defined layout (for example, invoices, delivery notes) all the time.
- These documents are often created automatically.
- The basic layout of the document is pre-defined, but, in many cases, other data has to be merged with it, such as address data or purchase order items. This data might be entered manually by an employee or retrieved from a database table.
- Large quantities of these documents may have to be produced. Form printing is usually a matter of large print runs of documents such as pay slips, checks, order confirmations, reminders, and so forth.



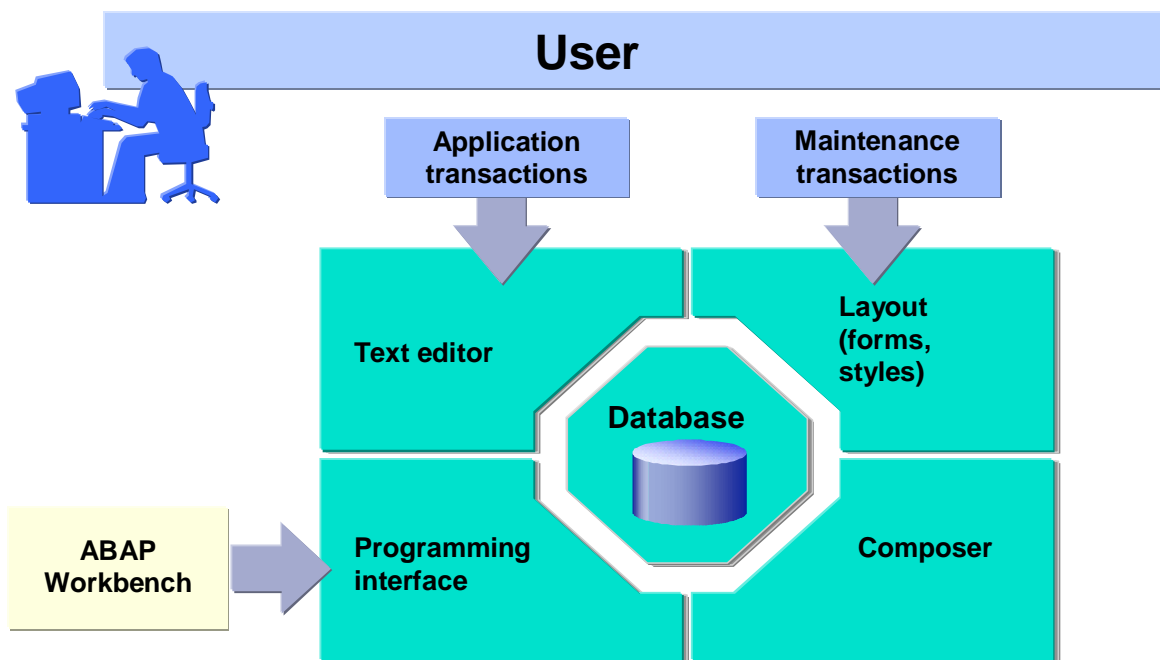
© SAP AG 2006

- SAPscript was developed to meet these requirements.
- SAPscript is an integrated tool for text entry and form printing in many SAP applications (Financial Accounting, Sales and Distribution, Materials Management, online documentation, and so on).

- **Maintenance in multiple languages**
- **High performance for mass printing**
- **Supports corporate identity**
- **Word processing scalability**
- **Interface to SAP transport system and translation systems**
- **Platform independence**

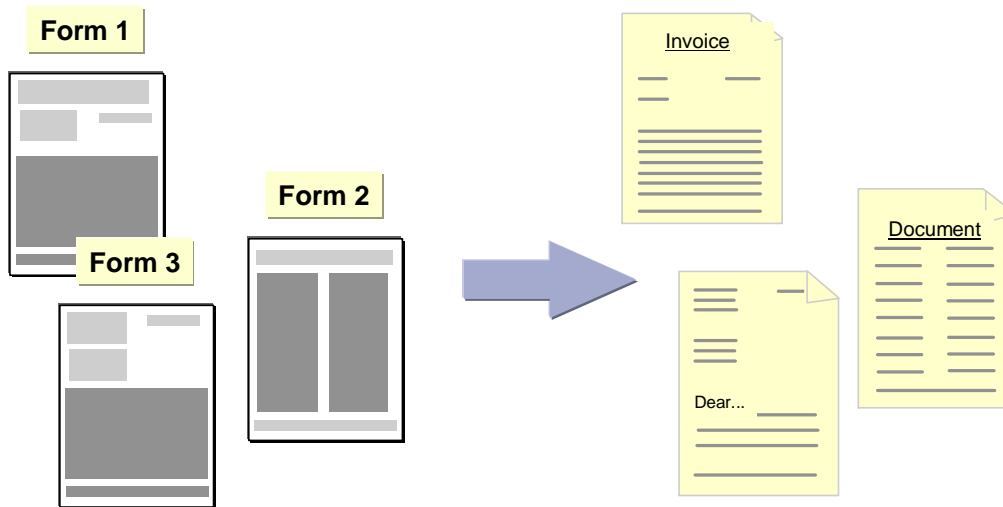
© SAP AG 2006

- The PC editor also supports Asian languages. Form printing allows for different address layouts based on country and for simultaneous use of multiple currencies.
- SAPscript supports a unified design in all documents.
- The editor interfaces provide you with the relevant functions in each application.
- SAPscript can be run on all frontends supported by SAP.



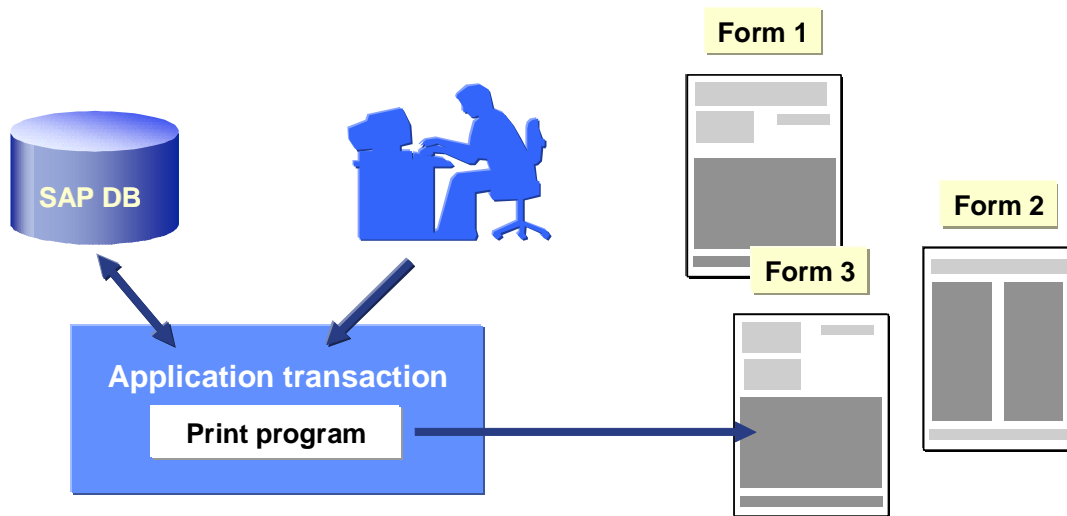
© SAP AG 2002

- A document's layout is defined in a form.
- Text modules are entered using the SAPscript Editor. These can be stored centrally in the database.
- Most SAPscript users only come in contact with the Editor. The Editor is used in various applications for entering text such as letters.
- You can generate documents automatically from an ABAP program using the SAPscript programming interface.



© SAP AG 2002

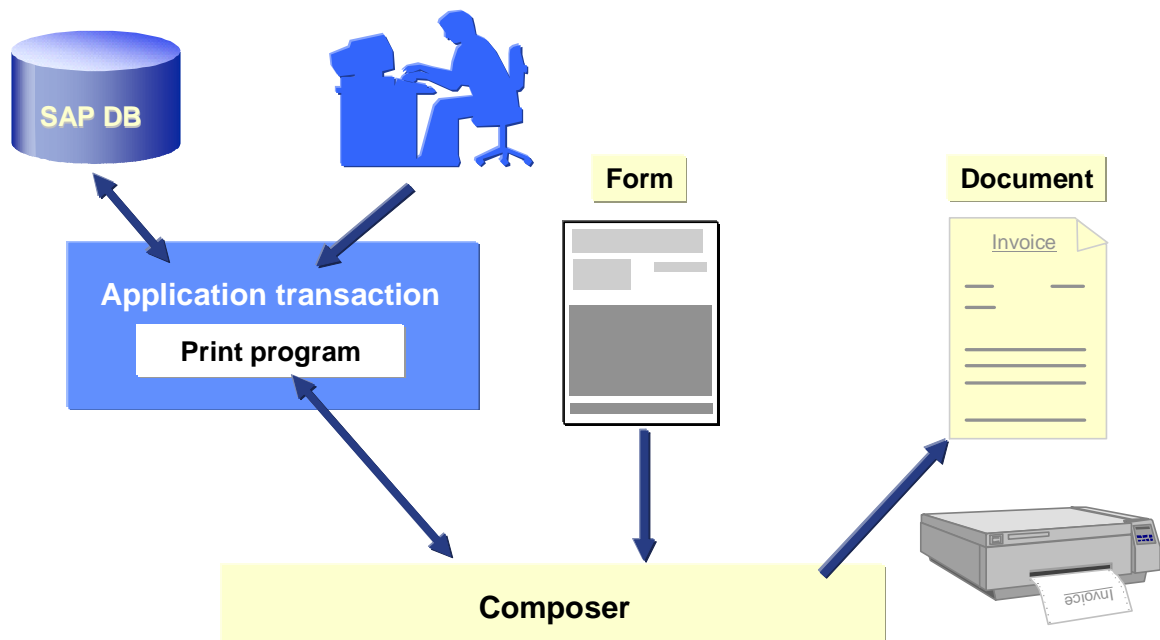
- A form specifies the appearance and structure of a document.
- Every SAPscript document uses a form.
- Forms contain predefined text modules with space reserved for variable data. You can use these text modules for different applications.
- To make changes to your documents, such as moving a piece of text, or changing fonts, paragraph formats, and tabs, you only need to change the form.
- To edit forms, choose *Tools* → *SAPscript* → *Form* from the main R/3 menu.
- To create your own forms, you can copy and customize existing forms.



© SAP AG 2002

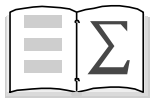
■ The print program:

- Controls the actual text output to the printer, screen or fax
- Selects data from the database or from user input
- Selects a form and controls which text should be printed, their sequence, and their frequency
- Determines the printer attributes such as output device, number of copies, whether to print immediately



© SAP AG 1999

- The final appearance of your document depends on the interaction between your print program and its form.
- The SAPscript print program initializes the printing process. Every command entered using the SAPscript programming interface is transferred to the composer.
- The composer receives layout information from the form specified in the print program. The documents are formatted according to this layout information.
- If the documents contain variables, the composer replaces these variables with data from the R/3 system, such as the current date, or with the user data selected by the print program.
- The print program controls the completion of the form. Once this is done, the composer places the completed document in the spool.



You are now able to:

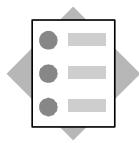
- **Understand how SAPscript fits into the overall structure**
- **Identify the integration of SAPscript in the SAP system**
- **Name the SAPscript components**
- **Understand how the components interact, in particular print programs and forms**

© SAP AG 2006

Contents:

- Windows and pages
- Paragraph formats
- Character formats
- Layout of forms

© SAP AG 2009



At the conclusion of this unit, you will be able to:

- **Describe the elements SAPscript offers to lay out a form**
- **Create and change form elements in R/3**

Example: Booking Confirmation

SAP

Fly & Smile 11/22/2007 Date Depart Price
4 Truckee Way
Durango, CO 85650

1/04/08 6:07 398.80 USD
1/16/08 15:06 990.00 USD

Turnaround Inc.
145 Apple Valley Ln.
Ithaca, NY 14850

Dear Sir/Madam,

Thank you for your order. We confirm
your bookings as follows:

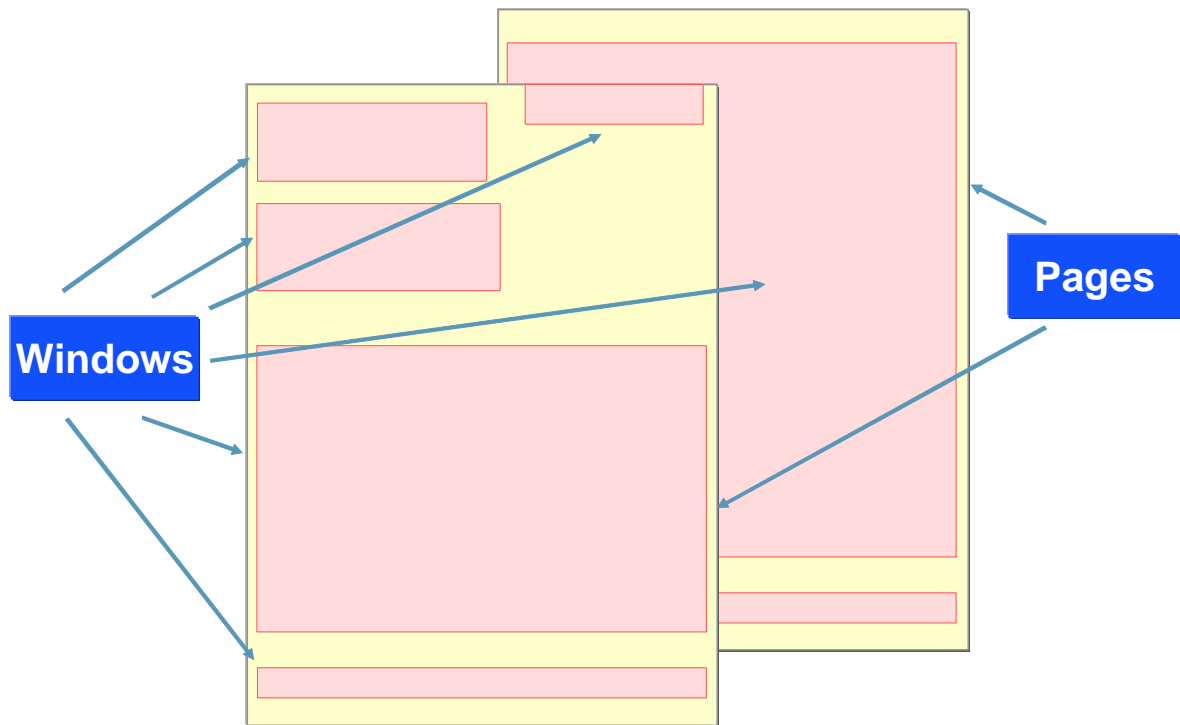
Flight	Date	Depart	Price
AA 0017	12/20/07	13:15	799.00 USD
AZ 2017	12/28/07	21:55	799.00 USD
LH 0400	12/21/07	6:07	398.80 EUR
LH 0400	12/28/07	6:07	398.80 EUR

Page 1

Page 2

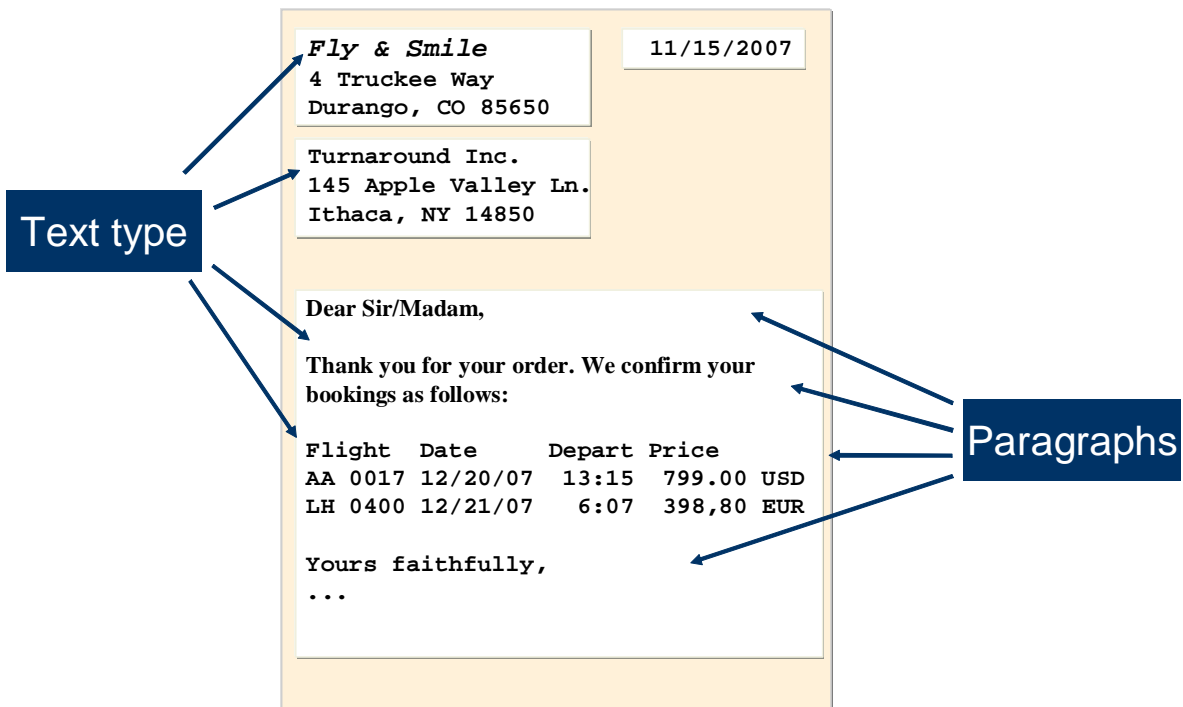
© SAP AG 2006

- A typical form contains the following details:
 - Address
 - Sender
 - Body text
 - Business transaction data (for example, booking data)
- Text and data often run over several pages in a form.



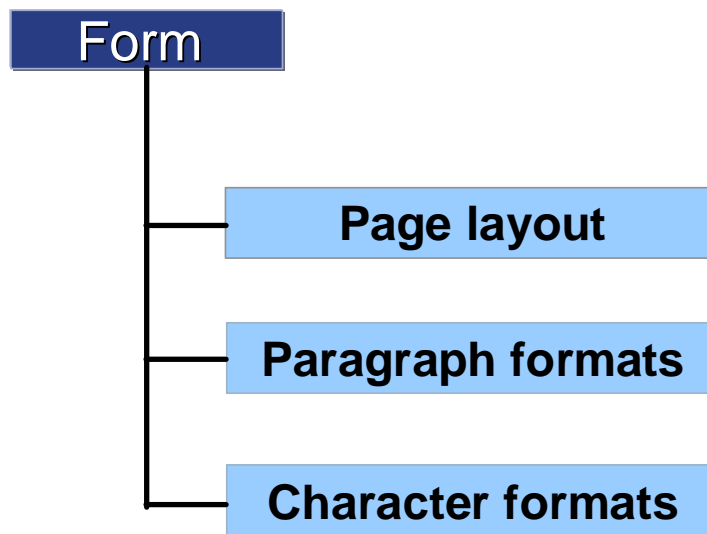
© SAP AG 1999

- The various output areas found on a form page are called **windows**.
- You can organize texts within your windows using **text elements** and **paragraphs**.



© SAP AG 2006

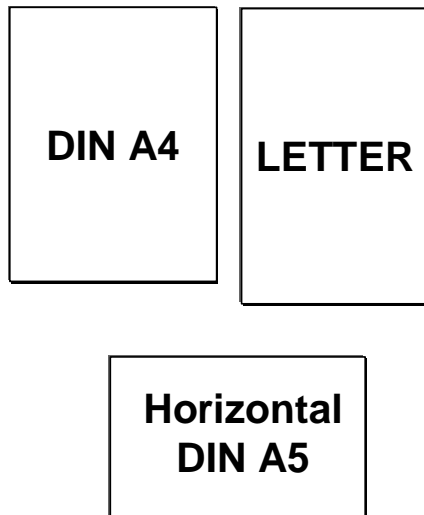
- Various paragraph and character formats provide you with additional ways to structure the texts that you create.



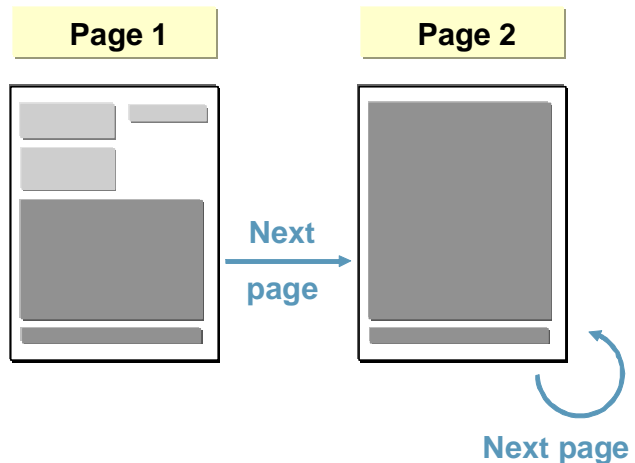
© SAP AG 1999

- The following sub-objects allow you to structure forms in SAPscript:
 - Page layout
 - Paragraph formats
 - Character formats
- Use the Form Painter to edit the individual sub-objects of a form.

Page format



Static page sequence



© SAP AG 2002

- A form can have one or more pages. You determine the page sequence of a document by simply entering the page that follows the one you are currently on. You must always enter a starting page in a form's header data.
- A specific page format (for example, LETTER or DIN A4) is defined for each form.
- SAPscript allows for automatic page numbering.

Template

Fly & Smile
4 Truckee Way
Durango, CO 85650

11/15/2007

Turnaround Inc.
145 Apple Valley Ln.
Ithaca, NY 14850

Dear Sir/Madam,

Thank you for your order.
We are pleased to confirm the
following bookings:

Flight	Date	Departure	Price
AA 0017	12/20/07	13:15	799.00 USD
AA 2017	12/28/07	21:55	799.00 USD
LH 0400	12/21/07	06:07	387.80 EUR
LH 0400	12/28/07	06:07	398.80 EUR

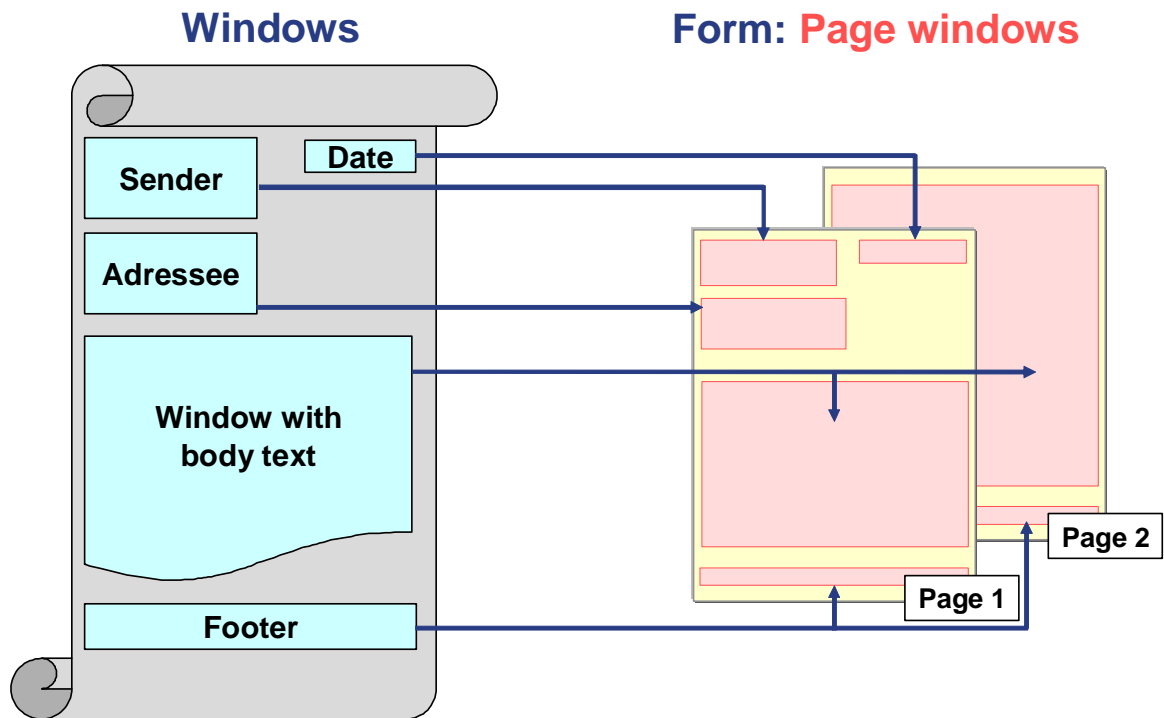
Page 1

© SAP AG 2006

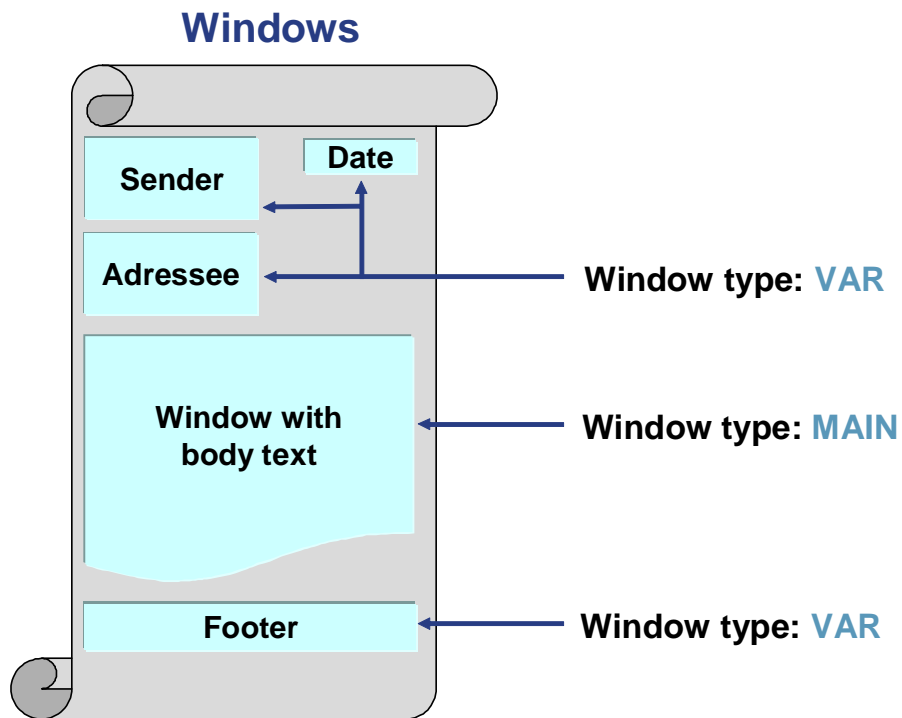
Form: Window

The diagram illustrates a form page structure with several rectangular windows. The page has a light yellow background. There are four distinct windows: a small one at the top right, a medium one at the top left, a large one in the center, and a horizontal one at the bottom. All windows are outlined in red.

- The different areas on a form page that might, for example, contain an address, sender, or body text, are called windows.



- Windows can appear on numerous pages within a single form. Therefore, SAPscript differentiates between logical windows and physical page windows.
- The user determines the placement of page windows on each page by entering their size and position. A window can have a different size and position on different pages.
- Texts that appear in page windows are defined for an entire form using that form's corresponding logical window.



- There are two types of windows in SAPscript:
 - MAIN
 - VAR
- Windows of type MAIN are used to display text that goes on for multiple pages.
- Windows of type VAR may vary in size and position for each page that they appear on. The text entered in their corresponding logical windows will, however, always be displayed on each page where the window occurs. Any text that cannot be fully displayed due to the size of the window is lost.
- Up to Release 4.0, windows of type CONST still exist. These windows behave in the same way as type VAR windows.

Fly & Smile
4 Truckee Way
Durango, CO 85650

11/15.2007

Turnaround Inc.
145 Apple Valley Ln
Ithaca, NY 14850

Dear Sir/Madam,

Thank you for your order. We confirm your bookings as follows:

Flight	Date	Depart	Price
AA 0017	12/20/07	13:15	799.00 USD
AZ 2017	12/28/07	21:55	799.00 USD
LH 0400	12/21/07	6:07	398.80 EUR
LH 0400	12/28/07	6:07	398.80 EUR

Page 1

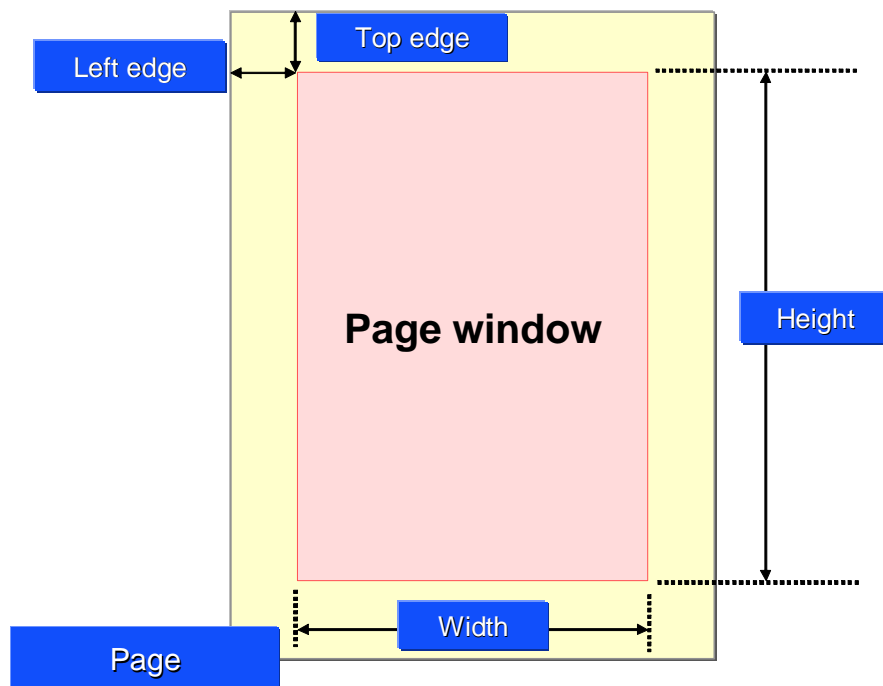
Flight	Date	Depart	Price
LH 0400	01/04/08	6:07	398.80 EUR
LH 0003	01/16/08	15:06	990.00 EUR

Yours faithfully,
...

Page 2

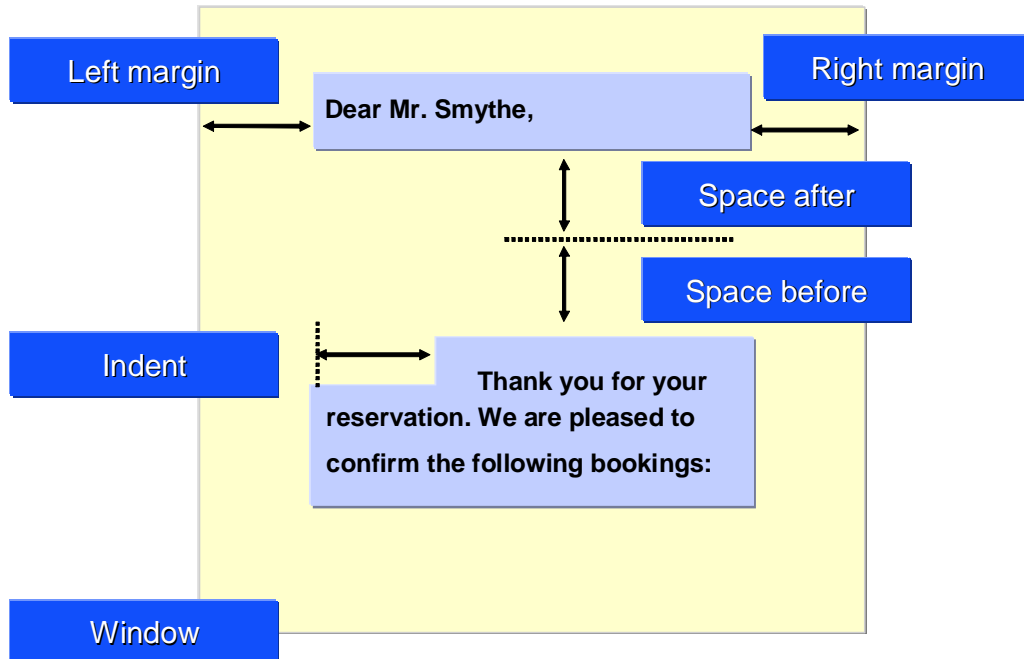
© SAP AG 2006

- In the main window, texts of variable length are displayed, for example all of a customer's bookings.
- Whenever the output area of one page is filled, SAPscript continues outputting body text in the main window on the next page.
- Main windows can be at a different position and level on the pages on which they appear. However, they must always have the same width.



© SAP AG 2002

- All forms have pre-determined page formats (for example LETTER or DIN A4).
- The position of a page window is indicated by its distance from the top and the left edges of the page.
- A page window's size is determined by its height and width.
- Both position and size can be given in various units of measure (for example, in lines or in characters).



© SAP AG 2002

- You can modify the appearance of text in SAPscript by changing its paragraph and character formats.
- Paragraph formats define:
 - Line spacing
 - Vertical spacing: before and after
 - Left margin/right margin
 - Paragraph alignment (such as left or centered)
 - First line indentation
 - Tabs
 - Fonts
 - Outline options

Bold, italic,
16 point

Arial,
14 point, bold

Fly & Smile
4 Truckee Way
Durango, CO 85650

11/15.2007

Turnaround Inc.
145 Apple Valley Ln.
Ithaca, NY 14850

Dear Sir/Madam,

Thank you for **your order**. We confirm your bookings as follows:

Flight	Date	Depart	Price
AA 0017	12/20/07	13:15	799.00 USD
AA 2017	12/28/07	21:55	799.00 USD
LH 0400	12/21/07	6:07	398.80 EUR
LH 0400	12/28/07	6:07	398.80 EUR

Page 1

© SAP AG 2006

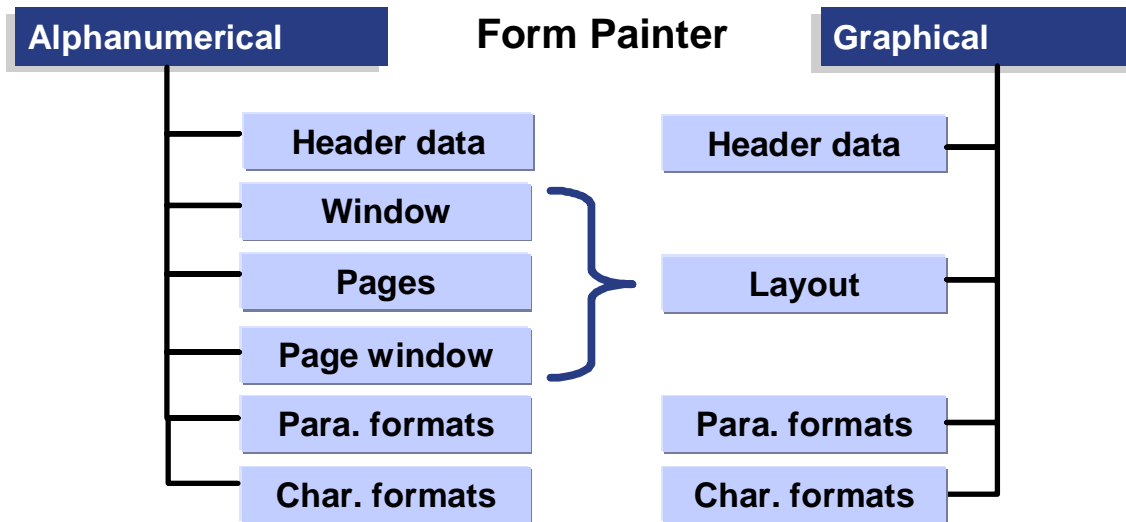
■ Similarly to paragraphs, you can define font attributes for character strings:

- Font
- Font size
- Bold/italics
- Underlining

The screenshot shows the SAP Form Painter interface. At the top is a menu bar with options: Form, Edit, Goto, Utilities(M), Settings, Environment, System, and Help. Below the menu is a toolbar with various icons for editing and viewing. The main window is titled 'Form Painter: Request'. It contains a 'Form' field with the value 'ZBC460_DEMO' and a 'Language' field with the value 'EN'. There is a 'Create' button next to the 'Form' field. Below these fields is a 'Subobjects' section with a list of radio buttons: Header (selected), Page Layout, Paragraph Formats, Character Formats, and Documentation. At the bottom of the 'Subobjects' section are 'Display' and 'Change' buttons.

© SAP AG 2006

- You can create and maintain forms by choosing Tools -> Form Printout -> SAPscript -> Form.
- Input help (F4) is available for the form name.
- You can edit the individual parts of a form directly:
 - Header
 - Page Layout
 - Paragraph Formats
 - Character Formats
 - Documentation

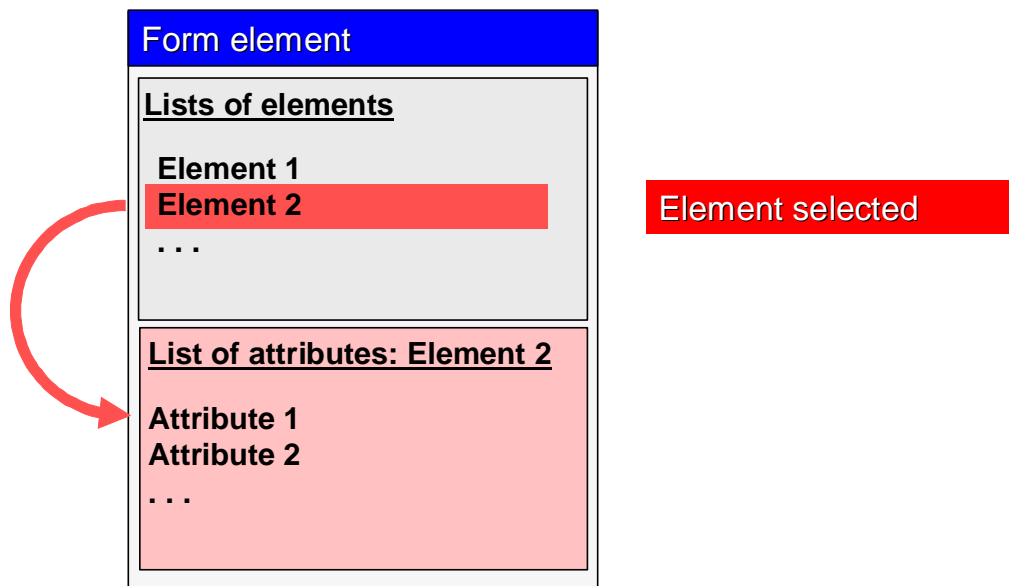


© SAP AG 2006

- As of Release 4.0, a new Form Painter with a graphical interface enables you to create and adjust forms faster and more intuitively.
- You can also continue to use the alphanumeric Form Painter if you prefer. All of the functions for designing forms are still available in the alphanumeric Form Painter.
- In the form maintenance transaction of the graphical Form Painter, the 'Windows', 'Pages', and 'Page windows' components from the alphanumeric Form Painter have been consolidated into a single component called 'Layout'.

Note:

The graphical Form Painter is available as of Release 4.0 for frontends with either a Windows NT 4.0 or Windows 95/98 operating system. Otherwise, work with the alphanumeric Form Painter.



© SAP AG 2006

- The Form Painter maintenance screen is organized as follows:
 - Form element for example, a page
 - Lists of elements for example, a list of the windows for this page
 - Lists of attributes for example, a list of attributes for each window such as position, size, and window type



Header Data

Page Layout

Paragraph Formats

Character Formats

Documentation and Printing Test

© SAP AG 2006

Header data

Administrative data

Administration information

Language attributes

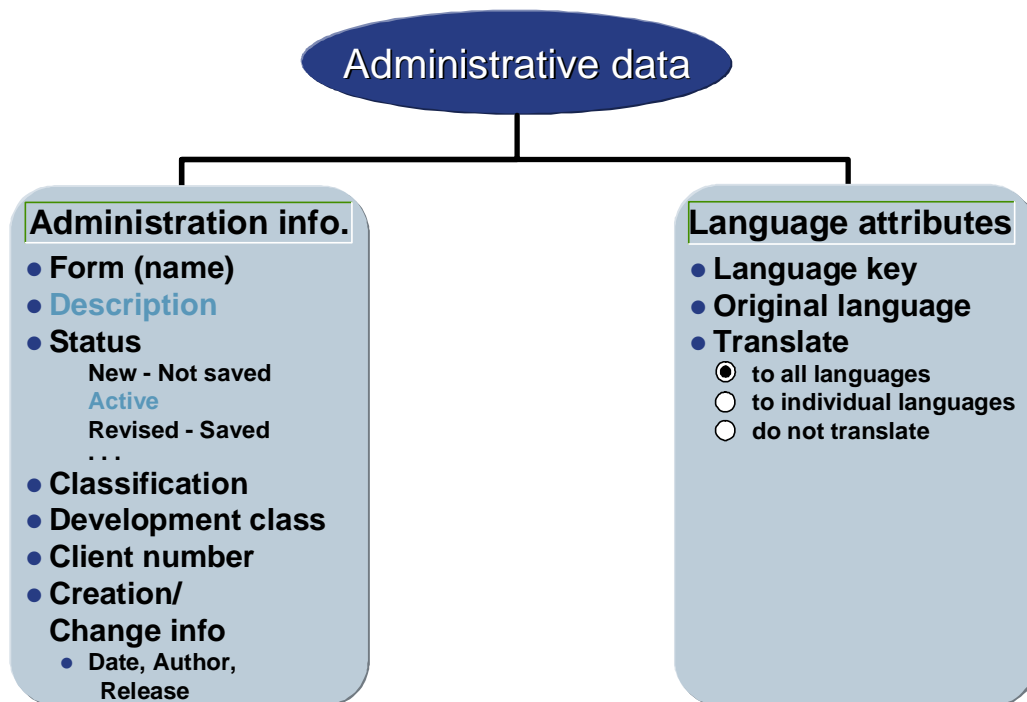
Basic settings

Set up page

Default values for text
formatting

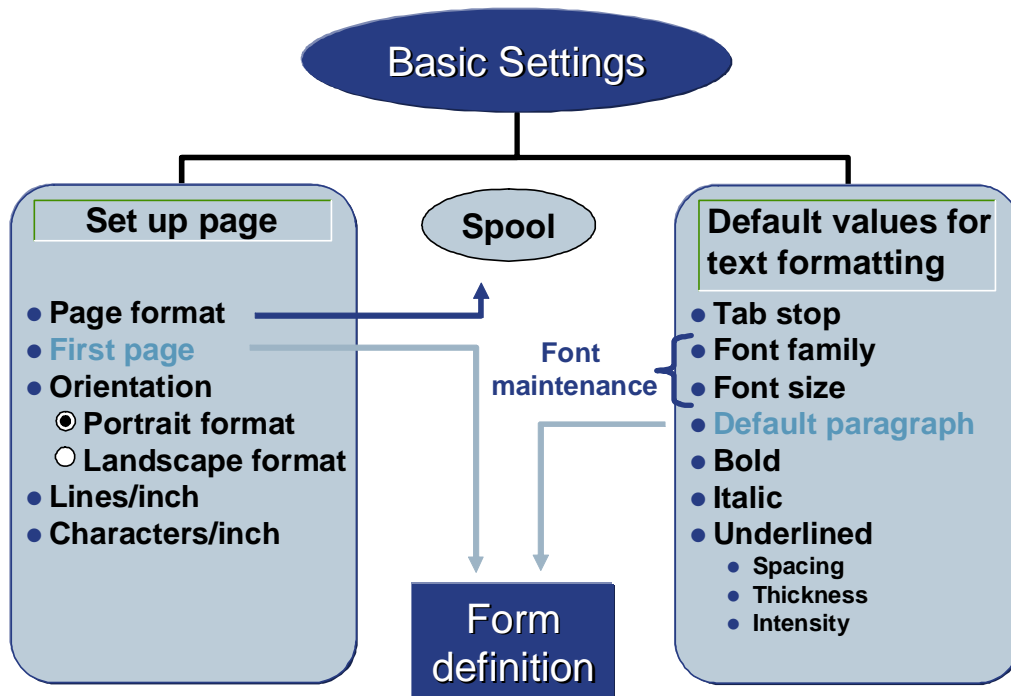
© SAP AG 2002

- You must enter the following items in each form's header data:
 - A short text describing the form
 - A default paragraph format (the format's attributes can be edited using the *Paragraph formats* function)
 - A default value for tabs
 - A first page (page attributes are defined using the *Layout* function in the graphical Form Painter and the *Pages* function in the alphanumeric Form Painter)
 - Page size (page format and orientation)
 - Line spacing and character spacing:
Form attributes such as tab stops or line spacing can be defined using more than one kind of unit. If you use the units CH (character) or LN (line), measurements are automatically converted to lines per inch (LPI) or characters per inch (CPI).
 - Font attributes (such as font, font size, bold, italics, or underlining)
- When creating a new form, you should always assign a description of what your form is to be used for.



© SAP AG 2002

- Administrative information, such as a form's name, client, and language, or the date when it was last changed, is displayed under **Administrative Data**.
- A form can have the following statuses:
 - **New:** The form has just been created and is not yet ready to be printed.
 - **Active:** The form is ready to be printed. To set the status of a form to active, choose *Form* → *Activate*.
 - **Revised:** A form that is revised and saved is given the status *revised*. However, whenever you print, R/3 continues to use the **Active** version of the form.
- To undo any unwanted changes you have made to a form, choose *Utilities* → *Reset*.



© SAP AG 2002

- Standard settings such as first page, page format, default paragraph, tab stops, font family, and font size can be made under **Basic Settings**.
- Both the first page setting and the default paragraph setting are mandatory and must be maintained by the user.
- Header data settings take effect only if no other settings have been made for that component.
Example: You want to use a default paragraph in a particular window. If no default paragraph has been specified in the window's attributes, the system automatically uses the default paragraph set in the header data.

Header Data

Page Layout

Paragraph Formats

Character Formats

Documentation and Printing Test

© SAP AG 2006

Graphical Form Painter: Page Layout

SAP

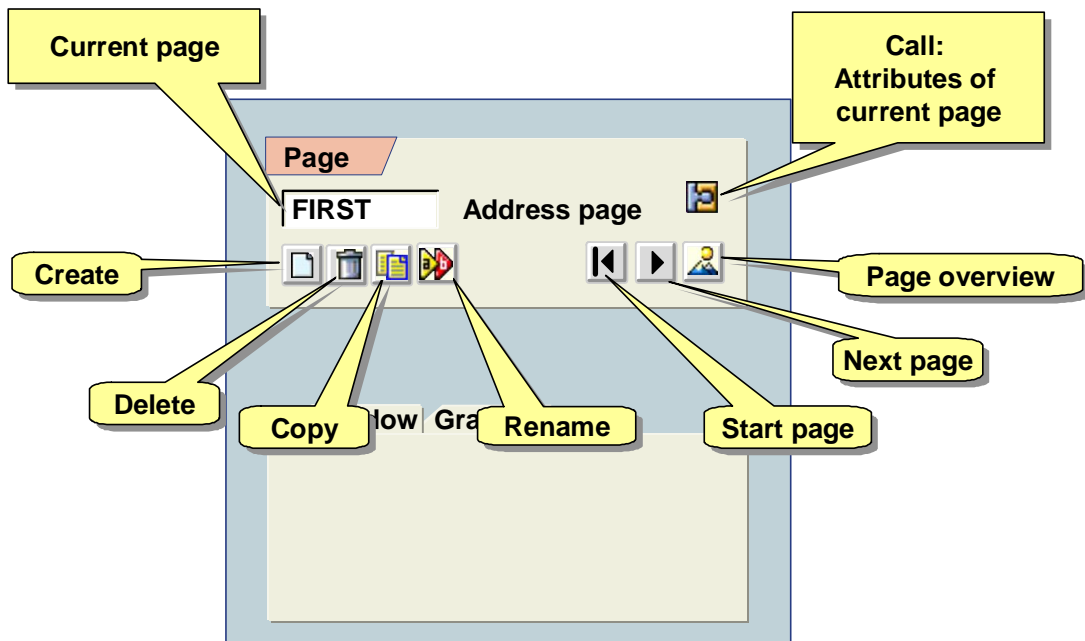
The screenshot displays the SAP Graphical Form Painter interface. On the left is the 'Page' window, which contains the following elements:

- Page Header:** A red tab labeled 'Page'. Below it, a text field contains 'FIRST', followed by 'Address Page' and a small icon.
- Navigation Icons:** A row of icons for file operations (copy, paste, delete, etc.) and navigation (back, forward, search).
- Window/Graphic Tabs:** Two tabs, 'Window' and 'Graphic', are visible. The 'Window' tab is active.
- Window Properties:**
 - Window:** A text field containing 'ADDRESS'.
 - Description:** A text field containing 'Address'.
 - Main window:** A checkbox that is currently unchecked.
 - Default paragraph:** A text field.
 - Margins and Dimensions:** A table with four rows: 'Left margin' (1.60 CM), 'Upper margin' (1.60 CM), 'Window width' (9.00 CM), and 'Window height' (6.00 CM).
- Bottom Buttons:** A 'Design / Text' button and an 'Options' button.

On the right is the 'Form ...' window, which shows a grid-based design area. It contains three windows: 'WINDOW1*', 'WINDOW2*', and 'MAIN*'. The 'MAIN*' window is outlined with a dashed border. A blue box labeled 'Administrative window' points to the 'Page' window, and another blue box labeled 'Design window' points to the 'Form ...' window.

© SAP AG 2006

- The page layout can be edited in both the administrative window and the graphic design window.
- In the design window, you can determine the size and position of display areas using drag and drop functionality.
- The most important attributes of the current page and its windows are displayed in the administrative window.



© SAP AG 2002

- Forms can have multiple pages defined for them. A document's page sequence is controlled statically by the **First page** value entered in its header data and by the **Next page** value entered in its Page attributes.
- If no next page is specified, form printing ends with the current page.
- Use the *Layout* function from the initial Form Painter screen to call the graphical Form Painter. From the **Page layout** screen, you can now define the pages you need. You can select standard functions, such as *Create page*, *Copy page*, and *Delete page*, using the *Edit* → *Page...* menu or by selecting the appropriate pushbutton on the screen.
- Additional attributes can be entered for all those pages found in the list box display.

Pages: Attributes

SAP

The screenshot shows the 'Attributes Page' dialog box in SAP. It contains several sections: 'Page' (FIRST), 'Description' (Address page), 'Next page' (NEXT), 'Page counter' (with Mode: Initialize counter, Increase counter, Keep counter unchanged), 'Numbering' (with Numbering type, Output length, and an Uppercase checkbox), and 'Print attributes' (with Resource name and Print mode). Callouts provide additional information: 'NUMBERING TYPES' lists ARABIC (Arabic numerals), CHAR (Fixed character), LETTER (Letters), and ROMAN (Roman numerals); 'For example: Output length 4 -> Page 1: 0001'; 'Paper tray choice' points to the Print mode field; and 'One-sided or double-sided printing default' lists S (Simplex mode), D (Duplex mode), and T (Tumble mode). The bottom of the dialog has a status bar with a green checkmark, a blue 'i' icon, and a red 'X' icon, along with the copyright notice '© SAP AG 2002'.

NUMBERING TYPES

ARABIC	Arabic numerals
CHAR	Fixed character
LETTER	Letters
ROMAN	Roman numerals

For example:
Output length 4
-> Page 1: 0001

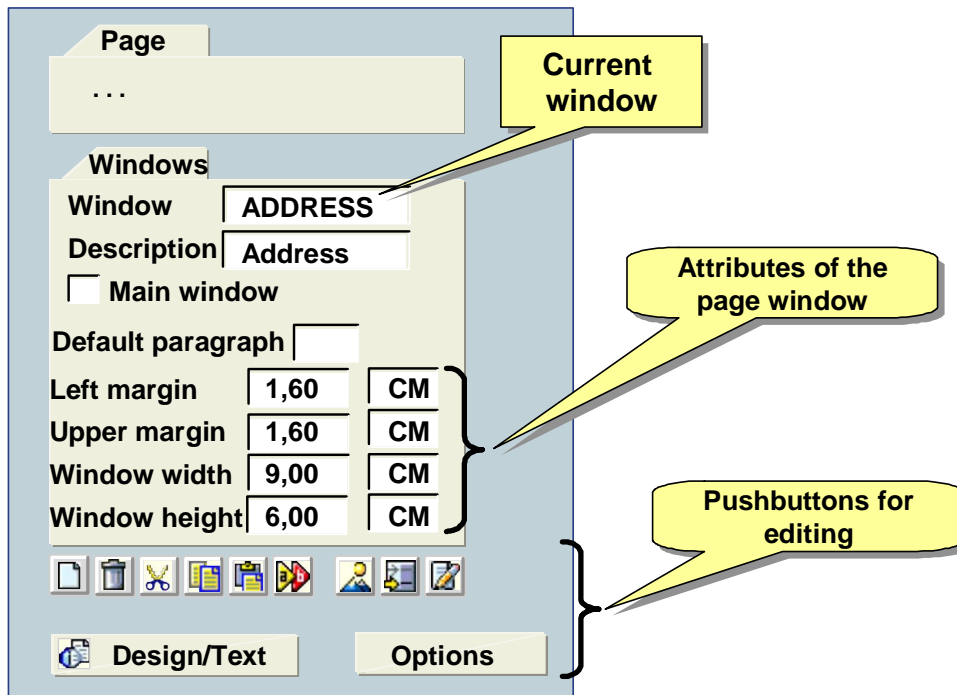
Paper tray choice

One-sided or double-sided printing default

S	Simplex mode
D	Duplex mode
T	Tumble mode

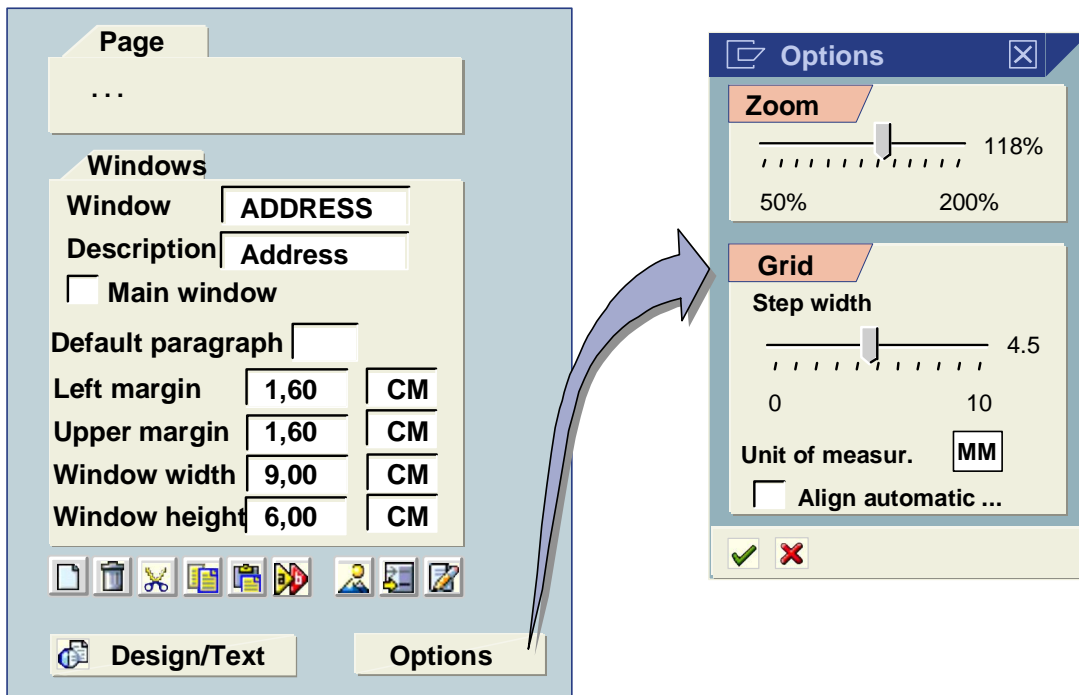
© SAP AG 2002

- Page attributes include *Next page*, *Page counter*, and *Print attributes*.
- The counter settings determine whether page numbers should increase, remain the same, or revert to their initial value.
- The print attributes allow you to choose which paper tray you want to use and whether you want the printout to be single- or double-sided.



© SAP AG 2002

- The Window list box displays a list of all windows found on the current page.
- For each window, important attributes such as position on the page, window size, window type, and default paragraph are displayed.
- A number of standard window editing functions are also available on this screen including *Create window*, *Copy window*, *Delete window* and so forth.
- In addition, the *Design/Text* pushbutton allows you to switch between design mode and text mode in the design window.
- In design mode, you can use your mouse to work with windows; text mode displays the individual window texts.

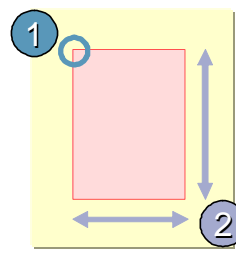
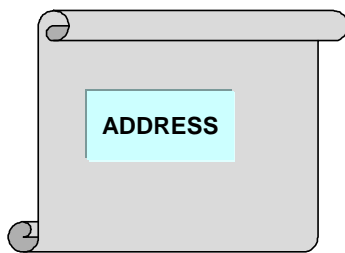


© SAP AG 2001

- The *Options* pushbutton allows you to zoom in on a specific section of your page.
- In addition, you can choose to have your window aligned automatically to the grid.
- You can also choose what size the individual units of your design window grid should be in a unit of measure of your choice.

Attributes: Windows	
Window	ADDRESS
Description	Address
Window type	VAR
Default paragraph	BF

Attributes: Page Windows		
Window	ADDRESS	
Description	Address	
Window type	VAR	
Left margin	1.60	CM
Upper margin	1.60	CM
Window width	9.00	CM
Window height	6.00	CM



Units of measurement:	
CH	Characters
CM	Centimeters
MM	Millimeters
IN	Inches
PT	Points
TW	Twips

© SAP AG 2002

- Windows and page windows are dealt with separately in the alphanumeric Form Painter.
- You can define logical windows in a form by specifying the name and type of window. To specify the main window, which can contain text extending across more than one page, use MAIN.
- To specify the other windows of a form, use VAR or, prior to Release 4.0, CONST:
 - CONST: The window has the same dimensions on all pages where it occurs.
 - VAR: The height and width of the window can vary.
- To create text for individual windows of the form, choose *Text elements*. The text you create is divided into text elements that are stored together with the form. You will find out about the Text Editor in the next chapter.
- You can set a default paragraph for any window, which then applies to all text elements within it.
- You can assign logical windows to any pages of your form. You define the distribution of page windows on a page by specifying the size and position of the windows.
- To print labels, or to print documents with more than one column, you can define more than one main window on any one page by choosing *Edit → Windows → Create → Main window*.

Header Data

Page Layout



Paragraph Formats

- Standard attributes
- Font attributes
- Tabs
- Outline attributes

Character Formats

Documentation and Printing Test

© SAP AG 2006

Paragraph Formats: Standard Attributes

SAP

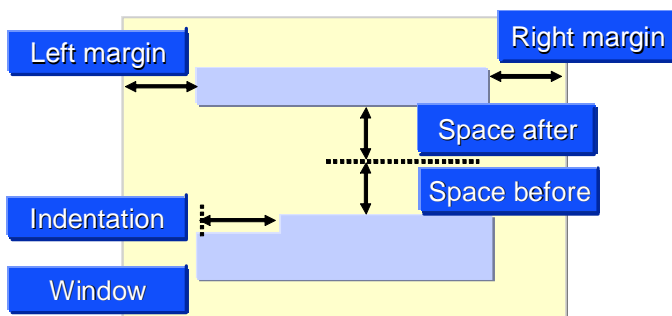
Standard attributes			
Paragraph	HD	Description	Header 1
Left margin	1.00	CM	Alignment LEFT
Right margin	1.50	CM	Line spacing 1.00 LN
Indent 1st line	2.00	CM	<input type="checkbox"/> No blank lines
Space before	1.50	CM	<input type="checkbox"/> Page protection
Space after			<input type="checkbox"/> Next paragraph same page

Alignment:

BLOCK justified
 CENTER centered
 LEFT aligned left
 RIGHT aligned right

Units of measurement:

LN Lines
 CM Centimeters
 MM Millimeters
 IN Inches
 PT Points
 TW Twips



© SAP AG 2002

- You can specify the following standard paragraph attributes:
 - Left/right margin
 - Paragraph indentation
 - Line spacing
 - Vertical spacing: space before/space after
 - Paragraph alignment: left, right, centered, or justified
 - Page protection: no automatic page break within a paragraph
- The characteristics of a paragraph are described by different attributes. Choose *Standard*, *Font*, *Tabs*, or *Outline* to switch between the different groups of attributes.

Font attributes			
Paragraph	HD	Description	Header 1
Family	COURIER	Bold	<input checked="" type="radio"/> On <input type="radio"/> Off
Font size	12,0 Point	Italic	<input type="radio"/> On <input type="radio"/> Off
		Underlined	<input type="radio"/> On <input type="radio"/> Off
			<input type="radio"/> Retain

SAPscript font maintenance values

Override header data values

Adopt values from header data

© SAP AG 2002

- Font attributes for paragraphs include:
 - Font family, such as Courier, Helvetica, or Times Roman
 - Font size (in tenths of a point)
 - Bold
 - Italics
 - Underlining, including the following additional options: spacing, thickness, and intensity
- You can take a particular font attribute from the general form attributes and use it in the current paragraph by changing your setting values to the following: Font family = space, font size = 0, bold, italics, and underlining all set to *Retain*.

Tabs

Paragraph **HD** Descript. **Header 1**

Number	Tab position	Alignment
1	1.25 CM	LEFT
2	2.50 CM	LEFT
3		

Units of measurement:
 CH Characters
 CM Centimeters
 MM Millimeters
 IN Inches
 PT Points
 TW Twips

Alignment:
 CENTER centered
 DECIMAL aligned to decimal character
 LEFT aligned left
 RIGHT aligned right
 SIGN +/- sign, aligned right

Scroll pushbuttons

© SAP AG 2002

- Tab attributes control the tab positions in a paragraph.
- You can define as many tab positions as you wish. To align a paragraph, use the following tab positions: LEFT, RIGHT, CENTER, SIGN, or DECIMAL. The standard tabs are available in addition to the tabs of a paragraph format.
- To define the position of numbers in a paragraph, select SIGN or DECIMAL.
- SIGN lets you define numbers right-aligned at the tab position. This leaves room for a minus sign or implied blank space at the end of the number.
- DECIMAL lets you align decimal points with any tab position.

Outline attributes

Paragraph	Description	Numbering 1
Outline	N1	
Outline level	01	
Number margin		
Left delimiter	(
Right delimiter)	
<input type="checkbox"/> Number chaining		

Numbering type

LETTER

Fixed character ☐

Output length

☐ Uppercase

Character string

Numbering types
ARABIC arabic numerals
CHAR fixed character
LETTER letters
ROMAN roman numerals

Example:
Output length 4 ->
Page 1: 0001

Display:
(a) First of all ...
(b) ...

Characters are formatted with the format entered

© SAP AG 2002

- You can use the outline attributes provided by SAPscript to divide text into units such as chapters, sub-chapters, and sections.
- To number paragraphs automatically, select one of the following numbering types:
 - Arabic numerals: **1, 2, 3...**
 - Roman numerals: **I, II, III...**
 - Letters **a, b,c...**
 - Fixed characters: ***, *, *...**
- You can also number text units by creating consecutive numbering levels :
 1. First level
 - 1.1. Second level, linked to first level
 - 1.2. Second level, linked to first level
 2. First level

[...]

Header Data

Page Layout

Paragraph Formats



Character Formats

- Standard attributes
- Font attributes

Documentation and Printing Test

© SAP AG 2006

Standard attributes			
String	<input type="text" value="BF"/>	Description	<input type="text" value="Boldface"/>
<input type="checkbox"/> Marker		Protected	<input type="radio"/> On <input type="radio"/> Off <input checked="" type="radio"/> Retain
Bar code	<input type="text"/>	Hidden	<input type="radio"/>
		Superscript	<input type="radio"/>
		Subscript	<input type="radio"/>

Symbolic name of a bar code from SAPscript font maintenance

Override header data or paragraph format settings

Adopt header data or paragraph format settings

© SAP AG 2002

■ Standard character format attributes include:

- Protected: The character string cannot be broken up and the entire string will be printed on one line.
- Hidden: The character string is not printed.
- Superscript: The character string is printed one half line space above the line.
- Subscript: The character string is printed one half line space below the line.
- Barcode: The character string is encrypted and printed as a barcode.

The screenshot shows the 'Font Attributes' dialog box in SAP. It contains the following fields and options:

- Char. Format:** BF
- Description:** Bold
- Family:** COURIER (circled in blue)
- Font Size Pt:** 12.0
- Style Options:**
 - Bold:** On (selected)
 - Italics:** Off
 - Underlined:** Off
- Retain:** Off

Three yellow callout boxes provide additional context:

- Values from SAPscript font maintenance:** Points to the 'Family' and 'Font Size Pt' fields.
- Override settings from header data or paragraph format:** Points to the 'On' radio button for 'Bold'.
- Use settings from header data or paragraph format:** Points to the 'Retain' radio button.

© SAP AG 2006

- Similar to paragraph formatting, you can define font attributes by specifying the font and font size, as well as the styles bold, italics, or underlining.
- By explicitly selecting these attributes (using the 'on' and 'off' radio buttons), you can override the paragraph formatting or form header data settings for text display.
- If you want the font attributes you have selected for a specific paragraph to apply to the remainder of the form, choose 'Retain' or leave the values for 'Family' and 'Font Size' blank.

Header Data

Page Layout

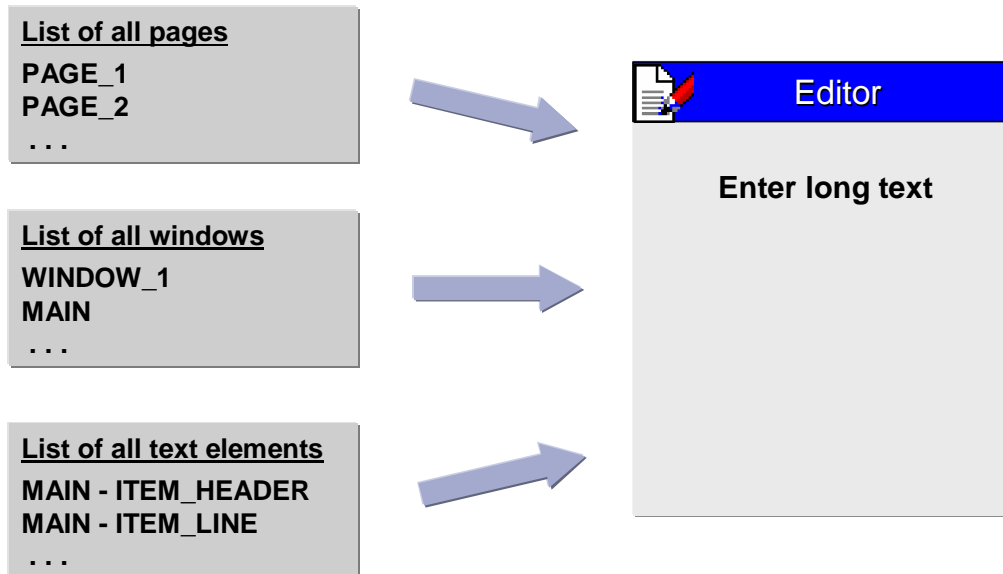
Paragraph Formats

Character Formats



Documentation and Printing Test

© SAP AG 2006

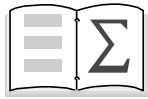


© SAP AG 2002

- Any changes that you make to a form should be documented accordingly.
- You can maintain form documentation using the *Form documentation* pushbutton.
- When you choose this pushbutton, a list of form components (pages, windows, text elements, and so forth) is displayed.
- On this screen, you can enter long text documenting the individual components.

**Print preview of the
form maintenance screen
(without print program)**

- You always use a print program when printing a form.
- If you want to preview your document before printing it without actually starting the print program, choose *Utilities* → *Test print* in the form maintenance transaction.
- The print preview that you get using this function is, however, not an exact representation of what your form will look like when it is output to a printer.
- Symbols that are filled by a program at runtime are represented by x's, for example, and the sequence of text elements with names is not the same during a test print as in a real printing. Also, control commands are displayed in a test print.
- Starting with 4.6C, graphics are displayed in the preview.



You can now:

- **Name the elements that SAPscript provides for structuring your form.**
- **Create these elements in the system and change them.**

© SAP AG 1999

Exercise



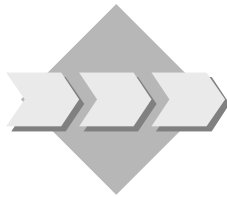
Unit: Form Elements

Topic: Simple Adjustments to Forms



At the end of these exercises, you will be able to:

- Make simple adjustments to forms



A given form must be adjusted to meet simple requirements.

Part 1: Copying a Form, and Simple Operations with Pages and Windows

- 1-1 First ensure that the graphical Form Painter is being used (initial screen of Form Painter → *Settings*). Create a new form called Z_BC460_## (## = your two-digit group number).
 - 1-1-1 Copy the existing form SAPBC460_VORLAGE to this form.
 - 1-1-2 Choose DINA4 as the page format.
- 1-2 Set the zoom for the Form Painter to 70%.
- 1-3 Create a new window with the name INFO on the page called FIRST. It should be at the same height as the existing ADDRESS window. (In the alphanumeric Form Painter, you first have to create a window and then the page window).
- 1-4 Create an additional page with the name NEXT.
 - 1-4-1 NEXT should be the static next page of FIRST and of NEXT itself.
 - 1-4-2 The NEXT page should contain the main window at a height of 25cm as well as the PAGENR window. (Both windows have already been created on the FIRST page.)
 - 1-4-3 Something to puzzle over:** Page numbering should begin with II (in Roman numerals) on the NEXT page.
- 1-5 Save the form either as a local object or in package ZBC460_## (## stands for your two-digit group number).
Activate your form.

Part 2: Paragraph and character format selection

- 2-1 Create at least the following formats for your form with name Z_BC460_##:
 - a paragraph format of your choice (if you like, create several with different outline levels)
 - Character format KL (choose a small font size)
- 2-2 In the INFO window, enter "Person responsible: Mr. Sample", and choose character format KL. If you wish, switch to the graphical editor (= WYSIWYG) (menu path: Goto → Change Editor).
- 2-3 Document your form.
- 2-4 Activate the form.
- 2-5 View the result in the test printout print preview. (Your course instructor will tell you the printer to use as the output device.)



Unit: Form Elements

Topic: Simple Adjustments to Forms

1-1 On the initial screen of the Form Painter, choose *Settings -> Form Painter....*
Proceed as described. Ask your course instructor for help if you experience difficulties.

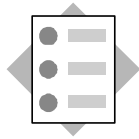
1-2 In the layout screen, choose *Utilities -> Options....*

From 1-3 onward: Proceed as described. Ask your course instructor for help if you have any questions.

Contents:

- **How the graphical editor and the line editor work**
- **Entering Texts**
- **Outputting text to the screen or a printer**
- **Simple text operations**
- **Text layout**

© SAP AG 2009



After completing this unit, you will be able to:

- Understand the structure of the graphical editor and the line editor
- Enter texts in the graphical editor and line editor
- List the text formatting options that are available
- Use text elements

© SAP AG 2008



The SAPscript Editor

Form Element: Text Element

© SAP AG 2006

Standard Text: Request

Line editor

Graphical PC editor

© SAP AG 2009

- Two different editors are available in SAPscript for entering text: the graphical editor and the line editor.
- In order to use the graphical editor, your operating system must meet certain requirements. Your frontend must have either Windows NT 4.0 or at least Windows 95 installed on it. The required components are installed via the SAP GUI. The user is not required to carry out any particular activities.
- From the 'Standard Text: Request' screen you can choose which editor you want to work with. Choose *Goto -> Change Editor* from within one editor to change to the other.

Graphical Editor

The graphical editor is the Microsoft Word editor

Paragraph and character format selection using list box

Functions can be reached using mouse or pushbutton

Special text structures displayed separately

Integrated syntax check for SAPscript control statements

Line editor

Paragraph formats in format column using F4

Character format selection using menu

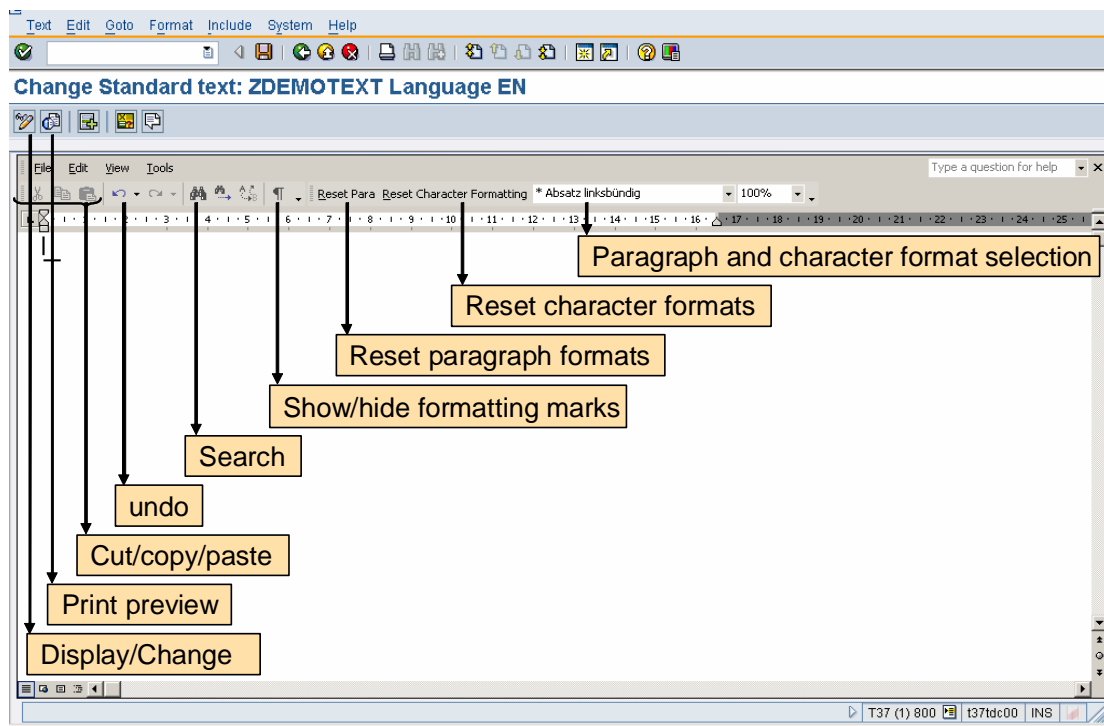
Text area selection using pushbutton or by double-clicking

Special text structures, such as SAPscript control statements are identified in the format column

Integrated syntax check for SAPscript control statements

© SAP AG 2009

- Den grafischen Editor und Zeileneditor können Sie zur Erstellung von Texten nutzen.
- The types of texts you will need to create include long texts in applications (for example, material texts), standard texts, and texts in forms.
- The major advantage of the graphical editor is that all its functions can be reached using either the mouse or pushbuttons. You can choose the paragraph and character formats you want to use with pushbuttons. The action is displayed immediately on the screen. SAPscript control statements are tested using an integrated syntax check.
- In the line editor, you assign paragraph formats in the format column (sometimes referred to as tag column). Character formats can be found in the menu. You can display the attributes of the individual formats using the menu. Text can be selected by double-clicking on it or by using a special pushbutton. Special text structures are identified in the format column.
- As of Release 4.0A, the integrated syntax check is also available in the line editor.

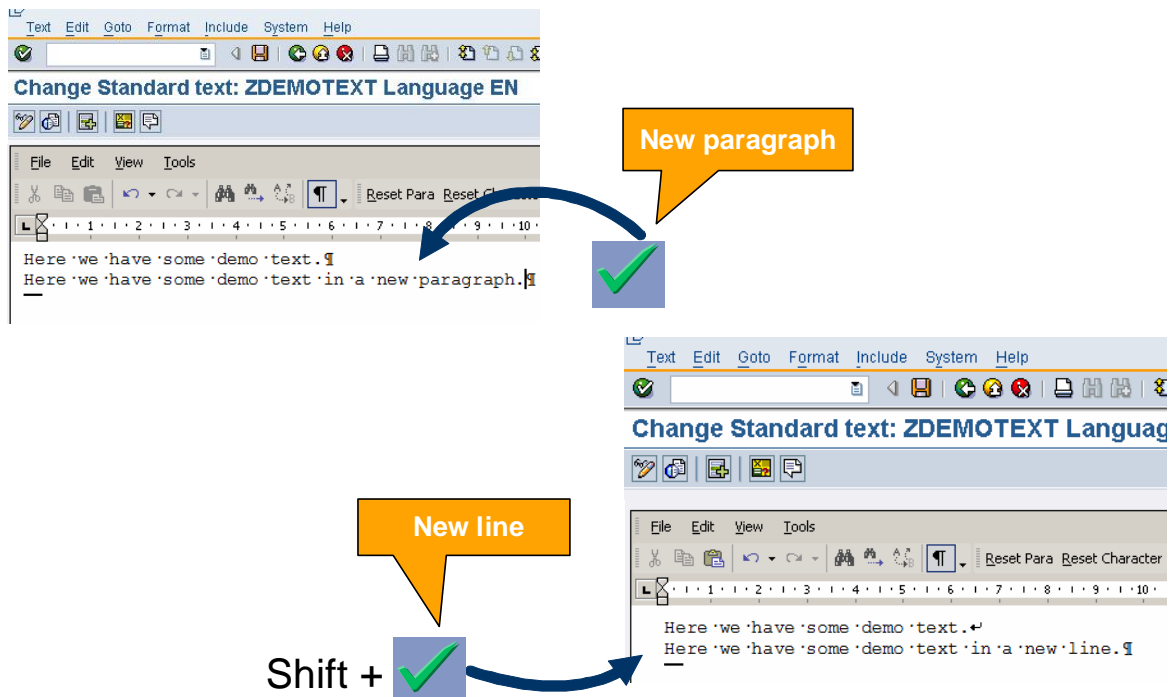


© SAP AG 2009

- The Microsoft Word editor is available as the graphical editor. The functions result from the WYSIWYG display of text (**What You See Is What You Get**).
- In contrast to the line editor, you do not control the graphical editor with line commands and the format column, but rather using pushbuttons and menu functions.
- You can use the paragraph format and character format list box to assign paragraph and character formats to the text.

Entering Texts in the Graphical Editor

SAP

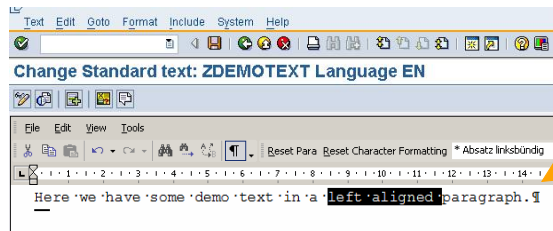


© SAP AG 2009

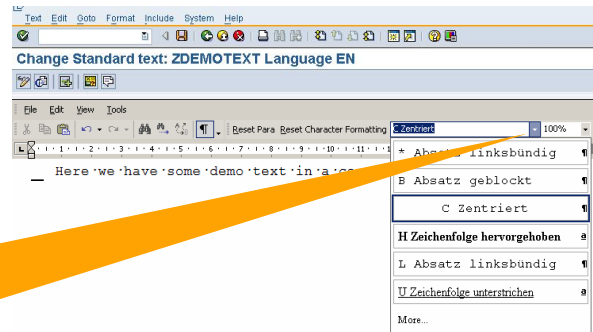
- The editor provides automatic word wraparound.
- To create a new paragraph, position the cursor and press ENTER. If no paragraph format has been used up to this point, the new paragraph is formatted according to the default paragraph setting. Otherwise the new paragraph assumes the format of the one immediately preceding it. To insert a line break within a paragraph, use SHIFT + ENTER.
- Within a paragraph, your text automatically wraps when it reaches the right edge of your window.

Assigning Paragraph and Character Formats

SAP



Select text
with the
mouse

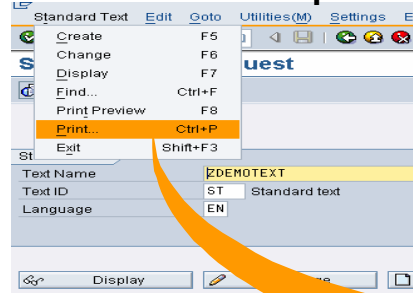


Assign new paragraph
or
character format

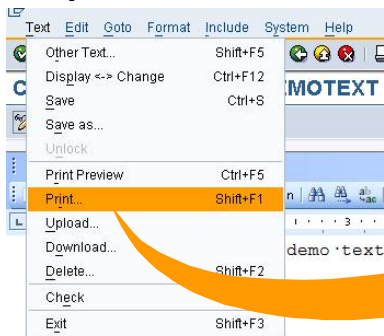
© SAP AG 2009

- You can use the graphic editor to format paragraphs. You can choose the paragraph and character formats that are available using the list box on the screen in the graphical editor.
- Use the mouse to select the text that you want to assign a new paragraph or character format to. Hold down the left mouse button and drag the cursor over the text to be selected. Expand the list box and choose the relevant paragraph or character format.
- If you want to reset the format, choose *Reset Paragraph Format* or *Reset Character Format*.

Standard Text: Request



Graphical Editor



Print Standard text ZDEMOTEXT

Output Device: P280

Number of copies: 1

Page selection: ☐ In Editor Format

Spool Request

Name: SCRIPT P280 BENTINK

Title:

Authorization:

Spool Control

☐ Print immediately

☒ Delete After Output

☒ New Spool Request

☐ Close Spool Request

Spool retention pd: 8 Day(s)

Storage Mode: Print only

Cover Page Setting

SAP cover page

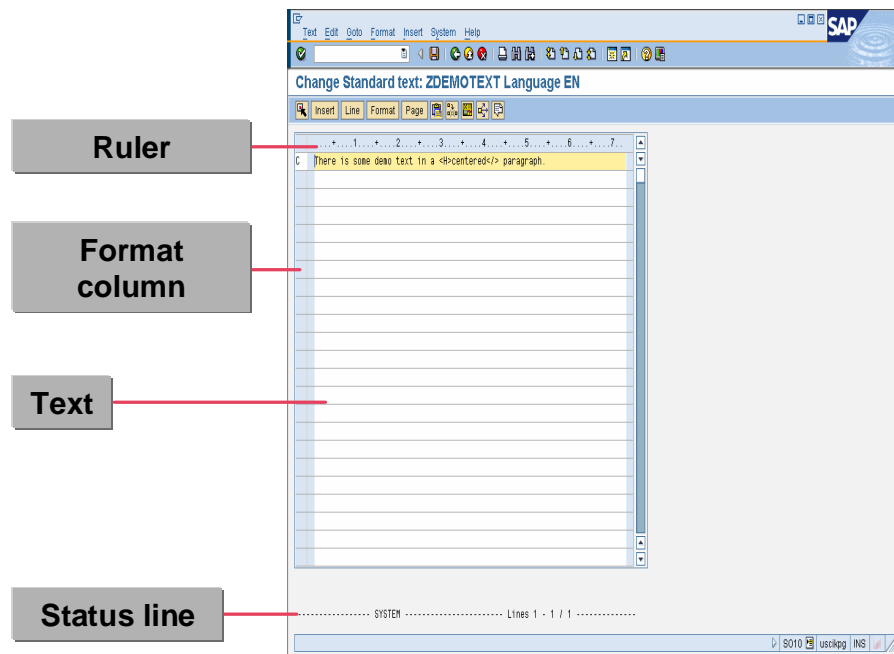
Recipient:

Department:

Print Preview Print

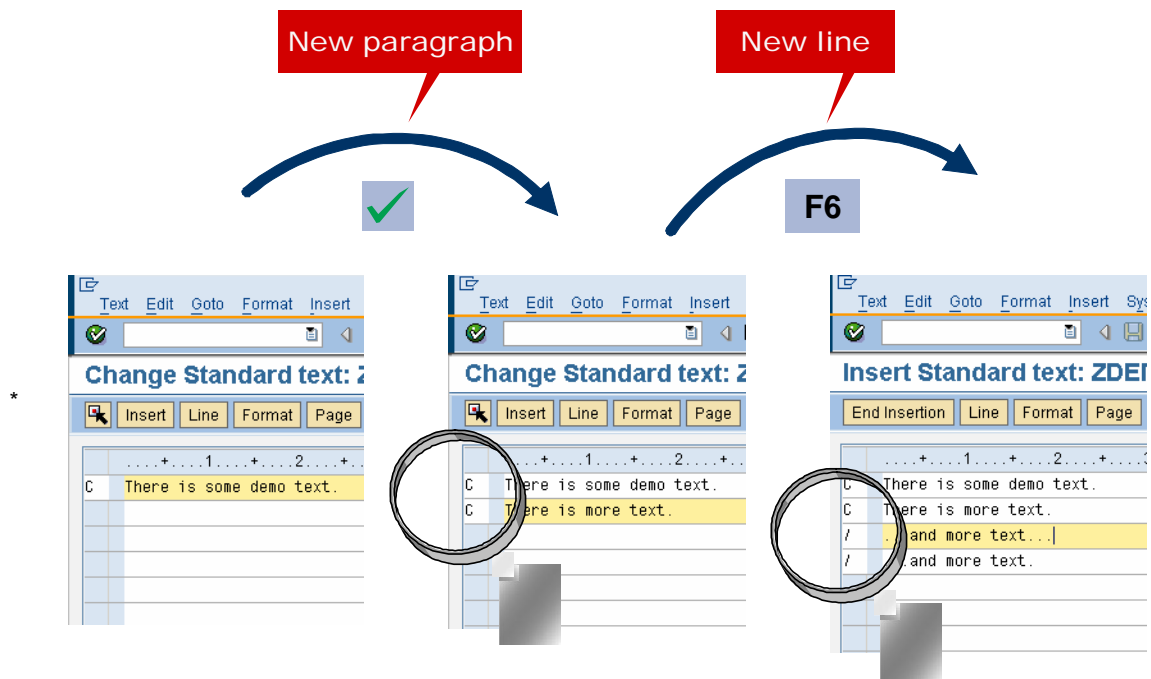
© SAP AG 2009

- To output a text, choose *Text* -> *Print* from the editor or *Standard Text* -> *Print* from the *Standard Text: Request* screen.
- In the editor menu, you can choose between print and print preview.
- In the dialog box that follows you can specify different print parameters. For instance, you can specify the output device, the number of copies to be printed, or which pages you would like to print, as well as spool control settings, such as print immediately.
- To preview the text output, choose the *Print Preview* pushbutton. You can trigger printing on the selected printer from this screen by choosing the *Print* pushbutton.



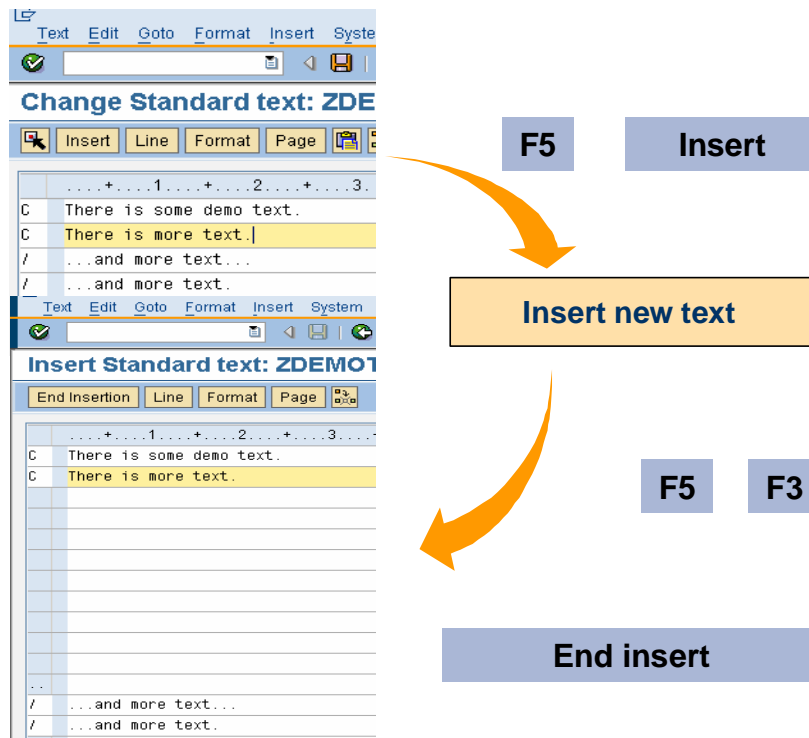
© SAP AG 2006

- The title bar in the line editor displays the name of the current editing action and of the text being edited.
- The menu bar offers users various application functions such as 'Save' or 'Print'.
- The format column contains the format ID or 'tag' which determines how the text will be formatted for output. Enter desired paragraph formats here.
- The status bar provides status information about various editing functions. This information might include:
 - The form currently being edited
 - The line section (screen lines/total number of lines)
 - The text passages currently selected
 - The last paragraph and character formats used



© SAP AG 2006

- To create a new paragraph, position the cursor and press ENTER. An asterisk (*) in the tag column indicates the default paragraph format. If you have used a different paragraph format, the new paragraph is marked with the previous format key.
You can use the F6 function key to create a line break within a paragraph. A slash (/) in the tag column indicates this line break.
- The editor provides automatic word wraparound when you select a function key, select a menu function, or press ENTER. Words that were split at the end of a line are joined back together. You can also choose the Format pushbutton to achieve this result.
- If you overwrite a paragraph marker in the tag column with blank spaces and then choose the Format pushbutton, text that was previously divided into two paragraphs becomes one paragraph.
- If you overwrite a new line format marker (/) with blank spaces and then choose the Format pushbutton, blank lines are deleted and lines of text that used to be split are joined together.

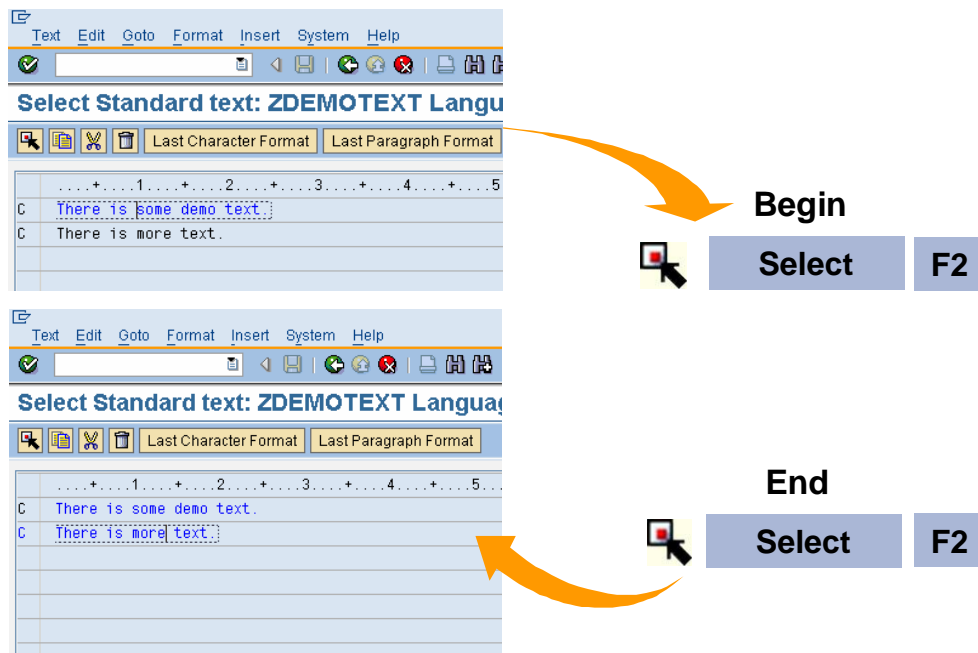


© SAP AG 2006

- The *Insert* pushbutton allows you to insert as much new text as you like at the current cursor position.
- This prompts the editor to switch into insert mode and provide a number of empty lines. When insert mode is active, this is displayed in the title bar.
- The newly inserted text is formatted to match the paragraph format at the cursor position. You can change it as required.
- Use the *End Insert* pushbutton or the F3 function key to switch back to the original mode and insert the new text where the cursor is positioned.

Selecting Text in the Line Editor

SAP

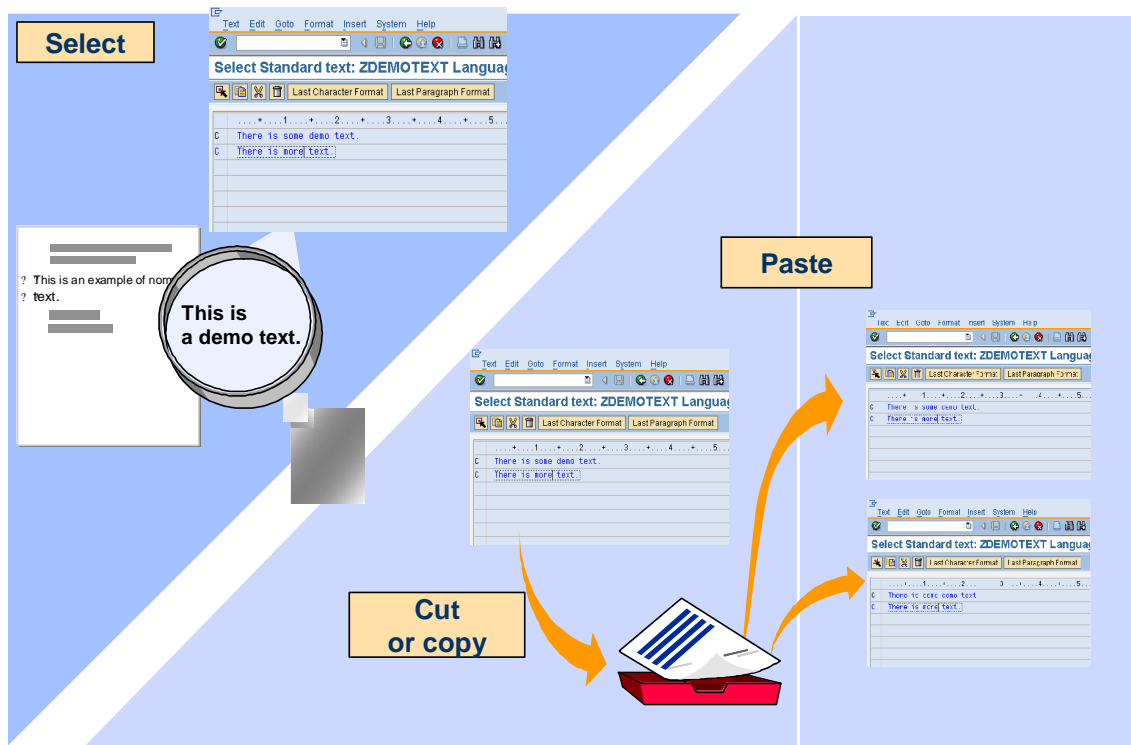


© SAP AG 2006

- You can select text to be edited using either the *Select Text* pushbutton or by double-clicking on it.
- Place the cursor at the start of the text you want to select and choose the pushbutton or double-click on it, and then place the cursor at the end of selection and choose the pushbutton or double-click on it.
- The system automatically switches to select mode after you have set the first select marker with your cursor. Selected text is highlighted in a different color. The area you have selected is also displayed in the status bar.
- You can also select entire lines by double-clicking in the format column. By repeatedly double-clicking in the format column you can extend the selected area.

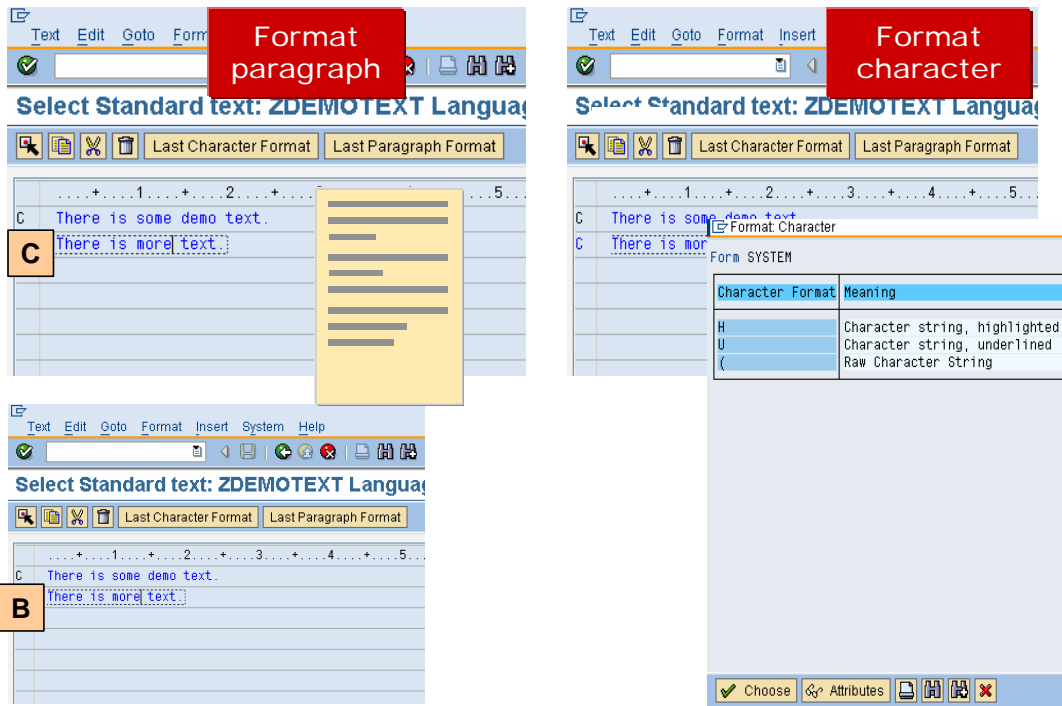
Selecting, Cutting, and Copying

SAP



© SAP AG 2006

- You need the select function to carry out actions such as deleting or copying areas of text.
- When you choose the *Cut* or *Copy* pushbuttons, the text you have selected is copied to the clipboard. The cut function causes the selected text to be removed from the editor; the copy function does not. In both cases, the previous content of the clipboard is deleted.
- To insert the contents of the clipboard, position the cursor and select *Paste*.
- It is possible to copy text between different editor sessions or screens using the clipboard.
- Under *Goto -> User Clipboard* there are five separate user clipboards which you can use to store and save texts of your choice.



© SAP AG 2006

- You can organize text using all the paragraph and character formats that are available in the current form.
- To apply paragraph and character formats, choose *Format -> Paragraph* or *Format -> Character*. The dialog box displays the format key, its description, and its attributes. Within the dialog box, you can display the attributes of the individual formats.
- You enter format keys in the format column. You can change or delete format keys that have already been entered. To display a list of possible paragraph formats, select F4.
- To format a character string, enter <NN>...</>. NN stands for a one-digit or two-digit character key.
- When you select a character string, the word on which the cursor was positioned is formatted. When you select an entire section, ensure there are no blank spaces between the last word to be selected and the cursor.
- To reapply the format last used when in selection mode, choose the pushbuttons *Last Character Format* or *Last Paragraph Format*.

<NN>...</>	Character string
„	Tab character
<(>...<)>	Hidden special characters
N...N&	Symbol

Paragraph	
NN	
Neue line	/
Raw line	(
Line feed and raw line	/(
Default paragraph	*
Comment line	/*
Long line	=
Long line with line break	/=
Command line	/:

© SAP AG 2006

- To suppress special character recognition, use <(>...<)>.
- To move a piece of text to the next tab position, set a double comma (,,) at the beginning of the text you want to move. You can define tab positions in paragraphs. If you do not define tabs, the default tab positions of the underlying form are used.
- To add a comment line which is ignored when the text is formatted, enter /* in the tag column.
- To protect a long line from the previously defined line format, enter =. This function also suppresses the implicit blank space in the editor at the end of the previous line.
- To format symbols, enter &N...N&. In SAPscript, symbols are variables that are only filled with values when the text is formatted for printing. This is explained in more detail in unit 4.
- A line marked with /: allows you to use a SAPscript control statement. Lines that contain control statements are not affected by editor formatting. The statements are only interpreted by the composer. Unit 4 contains further details about this topic.

The SAPscript Editor



Form Element: Text Element

© SAP AG 2006

Template

Fly & Smile
4 Truckee Way,
Durango, CO 85650

07/15/2007

Turnaround Inc.
145 Apple Valley Ln.
Ithaca, NY 14850

Dear Sir/Madam,

Thank you for your order.
We are pleased to confirm
the following bookings:

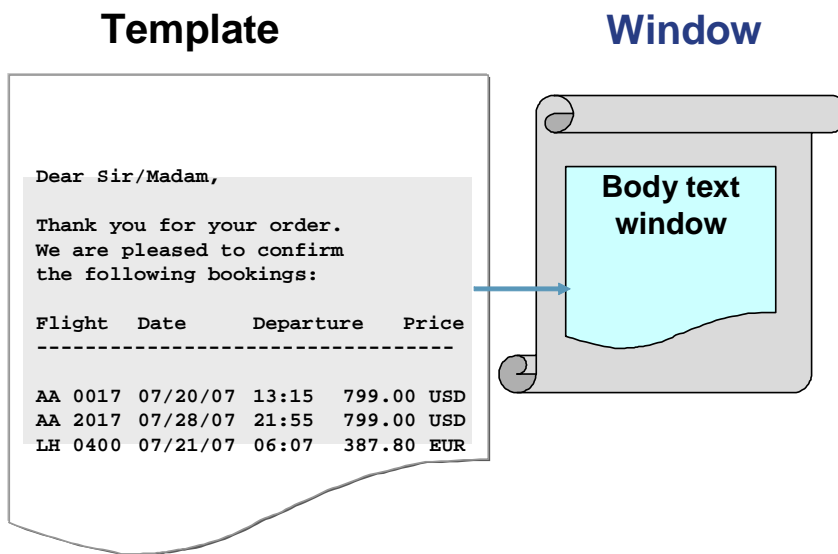
Flight	Date	Departure	Price
AA 0017	07/20/07	13:15	799.00 USD
AA 2017	07/28/07	21:55	799.00 USD
LH 0400	07/21/07	06:07	387.80 EUR
LH 0400	07/28/07	06:07	398.80 EUR

Page 1

© SAP AG 2006

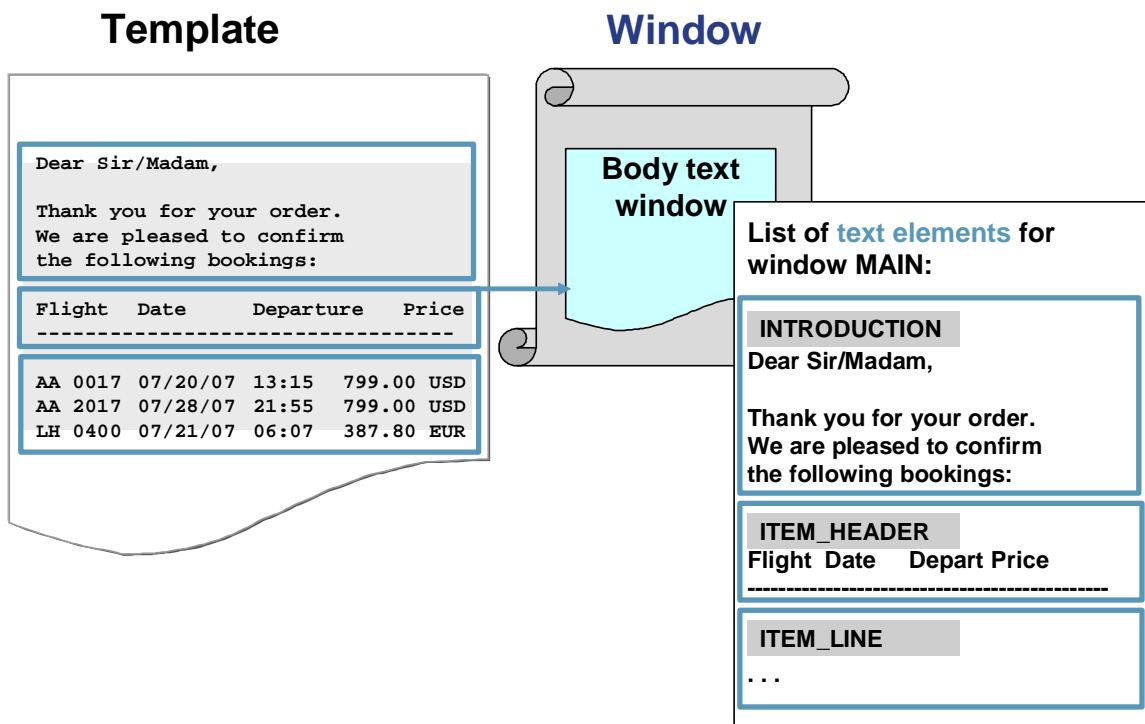
Form: Page window

- The various output areas that appear on form pages are called page windows.
- Page window texts can be structured using text elements.



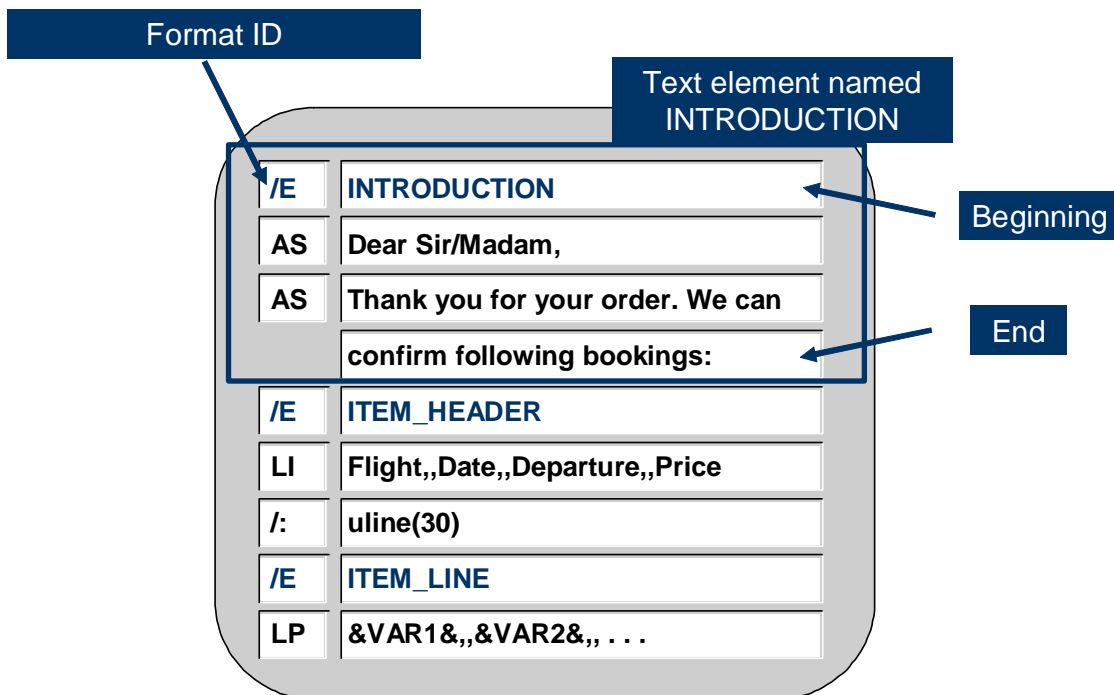
© SAP AG 2006

- The main window of a form should always output body text as a default.
- This body text contains the opening form of address and subsequent text, table header, and item entries.



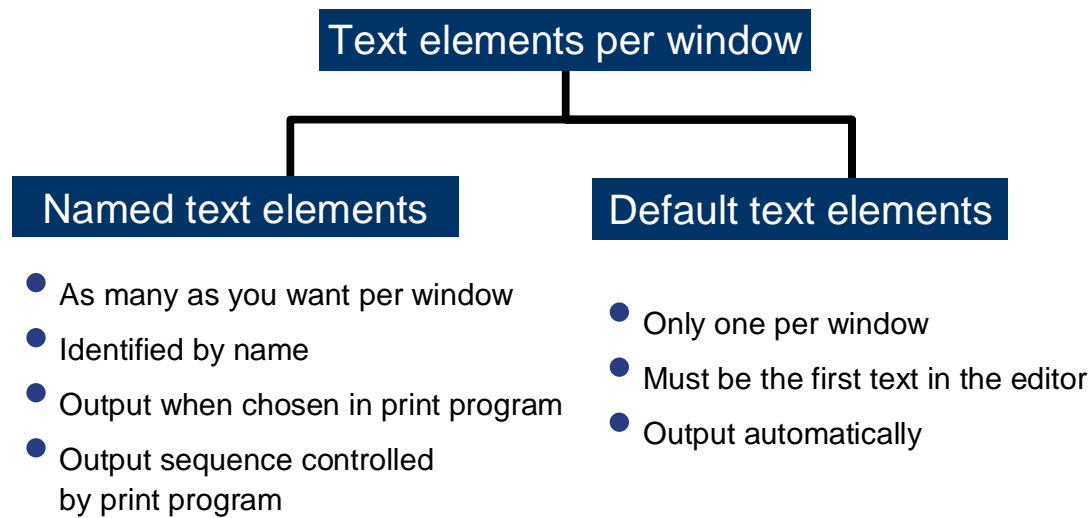
© SAP AG 2006

- The text in the example above has been divided into three different text elements, one for the form of address and subsequent text, one for the table header, and one for the individual item data.
- The sequence in which these elements are output is controlled by the print program. In addition, texts within the main window such as the table header can be displayed multiple times in different areas within the window on different pages. For more information about these functions, refer to the print program unit.
- You can also use the print program to determine if you want certain texts to be displayed at all.



© SAP AG 2006

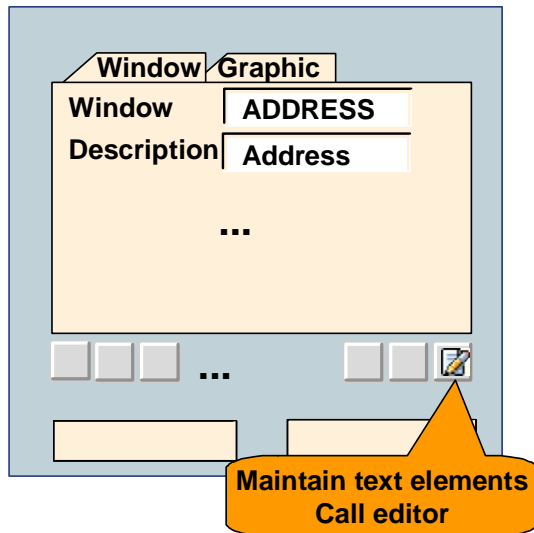
- Use the format ID /E in the format column of the line editor to denote the beginning of a text element. The name by which the text element is identified is highlighted in red in the line editor. The end of a text element comes when a new text element is started using /E in the format column.
- In the PC editor, a named text element's name is displayed in bold text. Additionally, the line appears gray.



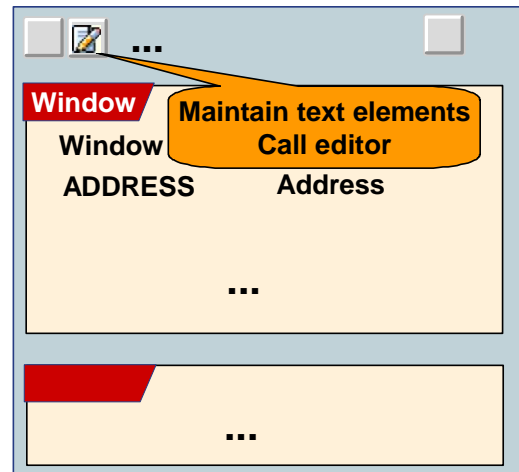
© SAP AG 2006

- SAPscript differentiates between default text elements and text elements with names.
- You can tell that you are dealing with a default text element whenever you have body text at the beginning of a window (for example, an address in a window other than a main window or a page number), but no /E text element is defined in the tag column. In contrast to named text elements, default text elements are output automatically, which means that no print control functions are available.

Graphical Form Painter

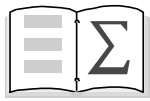


Alphanumeric Form Painter



© SAP AG 2006

- You can use the Form Painter to maintain all text elements in any given window.
- Use either the dialog box *Form: Change Page Layout* (graphical Form Painter) or the dialog box *Form: Change Windows* (alphanumeric Form Painter) to maintain these elements. Choose either *Edit -> Text Elements* or the corresponding pushbutton to branch to the editor for that text element.

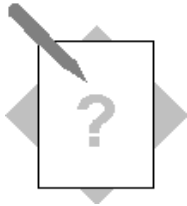


You are now able to:

- **Understand the structure of the graphical editor and the line editor**
- **Enter texts in the graphical editor and line editor**
- **List the text formatting options that are available**
- **Use text elements**

© SAP AG 2008

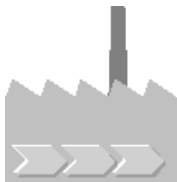
Exercises



Unit: Graphical Editor and Line Editor



- Changes to SAP standard forms
- Layout options using the graphical Form Painter
- Text changes
- Print preview using the print program



In most cases, standard forms are altered to reflect the needs of individual customers.

Activities-{}-	Object	Name / Description
	of the print program to be used	SAPBC460D_01
	of the form to be used	SAPBC460D_FM_03
	of the form to be created	Z_BC460_EX3_##
	Package	\$TMP or ZBC460_##
	## is your group number	

1 Make a copy of form SAPBC460D_FM_03 and review the output on your screen.

1-1 Create a new form with the name Z_BC460_EX3_##. Copy form SAPBC460D_FM_03 using *Form -> Copy from....*

1-2 Save and activate the form.

Choose *Save*.

In the *Create Object Directory Entry* window, choose the *Local Object* function or save in your package called ZBC460_##.

Choose *Activate*.

- 1-3 Run the print program SAPBC460D_01 for your form name.

Choose *Screen Display*.

In the next dialog box, choose *Print Preview*.

- 2 Make changes to your form. Test your form as described in 1.3.

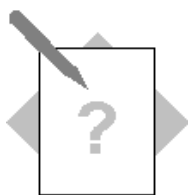
- 2-1 Make changes to the text of the letter (in the MAIN window). **Ensure that you use both the line editor and the graphical editor.** Make a change in one of the editors and observe the change in the other. You can switch between the two editors by choosing *Goto -> Change Editor*.

Use the following functions in the editors:

- *Insert*
- *Select*
- *Copy*
- *Cut*
- *Reinsert*

Note that you can work with the graphical editor in a very similar way to other text editors.

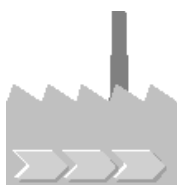
- 2-2 Change the address in your form.
- 2-3 Hide the page number on the first page.
- 2-4 Move the DATE window to the same height as the ADDRESS window. Note: The graphical Form Painter allows you to easily move a window to a different place on a page by dragging and dropping.
- 2-5 Create a paragraph format to write text with justified and bold formatting. Add some text using this new paragraph type.
- 2-6 In the graphical editor, use bold formatting for the flight number mentioned in the letter. Then observe the results in the line editor.
- 2-7 Add a new column called "Discount." Define new tabs for the corresponding paragraph formats. Enter the discounts in this column.



Unit: Graphical Editor and Line Editor



- Changes to SAP standard forms
- Layout options using the graphical Form Painter
- Text changes
- Print preview using the print program



In most cases, standard forms are altered to reflect the needs of individual customers.

- 1 Make a copy of form SAPBC460D_FM_03 and review the output on your screen.
 - 1-1 Create a new form with the name Z_BC460_EX3_##.
 - 1-1-1 Use *Copy from...* to copy form SAPBC460D_FM_03.
 - 1-1-2 Choose *Tools -> SAPscript -> Form*.
 - 1-1-3 Enter the name of the new form: Z_BC460_EX3_##.
 - 1-1-4 Choose *Create*.
 - 1-1-5 Enter a *description*.
 - 1-1-6 Copy the form using *Form -> Copy from...*
 - 1-2 Save and activate the form.
 - 1-2-1 Choose *Save*.
 - 1-2-2 In the *Create Object Catalog Entry* dialog box, choose the *Local Object* function or enter your package name, which is ZBC460_##, and choose *Save*.
 - 1-2-3 Activate your form using *Form -> Activate*.
 - 1-3 Run the print program SAPBC460D_01 for your form name.
 - 1-3-1 Choose *System -> Services -> Reporting*.
 - 1-3-2 Enter the program name and choose *Execute*.
 - 1-3-3 Enter the name of the form. Choose *Screen Display*.
 - 1-3-4 In the next dialog box, choose *Print Preview*.

2 Make changes to your form. Test your form as described in 1-3.

- 2-1 Make changes to the text of the letter (in the MAIN window). **Ensure that you use both the line editor and the graphical editor.** Make a change in one of the editors and observe the change in the other. You can switch between the two editors by choosing *Goto -> Change Editor*.

2-1-1 Ensure that you are using the graphical Form Painter.
In the form maintenance transaction (*Tools -> SAPscript -> Form*), choose the graphical Form Painter via *Settings -> Form Painter*.

2-1-2 Return to the form in change mode.

2-1-3 Choose *Layout*.

2-1-4 Use the mouse to select the MAIN window in the dialog box.

2-1-5 Choose *Text*.

2-1-6 Use the following functions in the editors:

- *Insert*
- t*
- *Select*
- *Copy*
- *Cut*
- *Reinsert*

Note that you can work with the graphical editor in a very similar way to other text editors.

2-1-7 Activate the form via *Form -> Activate* and test it via *Form -> Check*.

2-2 Change the address in your form.

2-2-1 Switch to the ADDRESS window.

2-2-2 Choose *Text*.

2-2-3 Make changes.

2-2-4 Activate the form via *Form -> Activate* and test it via *Form -> Check*.

2-3 Hide the page number on the first page.

2-3-1 Return to the graphical Form Painter.

2-3-2 Ensure that you are on the FIRST page.

2-3-3 Switch to the PAGENR window.

2-3-4 Choose the *Delete* function on the *Form -> Change Page Layout* screen.

- 2-4 Move the DATE window to the same height as the ADDRESS window.
Note: The graphical Form Painter allows you to move windows easily with drag and drop functionality.
- 2-4-1 Return to the graphical Form Painter.
 - 2-4-2 Ensure that you are on the FIRST page.
 - 2-4-3 Use the mouse to choose the DATE window.
 - 2-4-4 Hold down the left mouse button and move the DATE window to the same height as the ADDRESS window.
Note: You could assign the same coordinates to the DATE window as the ADDRESS window.
 - 2-4-5 Activate the form via *Form -> Activate* and test it via *Form -> Check*.
- 2-5 Create a paragraph format to write text with justified and bold formatting. Add some text using this new paragraph format.
- 2-5-1 Switch to paragraph view by choosing *Goto -> Paragraph Formats*.
 - 2-5-2 Create a new paragraph using *Edit -> Create Element*.
 - 2-5-3 Enter a two-digit key and a description.
 - 2-5-4 Set the alignment to *JUSTIFIED*.
 - 2-5-5 Choose *Font* to activate bold formatting and to choose a font family.
 - 2-5-6 To return to the layout view, choose *Goto -> Layout*.
 - 2-5-7 Edit the text elements of the MAIN window as before.
 - 2-5-8 Insert a new paragraph.
 - 2-5-9 Position the cursor on the paragraph and use the mouse to select the newly created paragraph format.
 - 2-5-10 Chose *Goto -> Editor* to switch to the line editor.
Look at the paragraph tag.
 - 2-5-11 Activate the form via *Form -> Activate* and test it via *Form -> Check*.
- 2-6 In the graphical editor, use bold formatting for the flight number mentioned in the letter. Then observe the results in the line editor.
- 2-6-1 Edit the text elements of the MAIN window as before.
 - 2-6-2 Select the flight number in each line by double-clicking on it.
 - 2-6-3 Select character format *Bold*.
 - 2-6-4 Switch to the line editor.. Observe how the character format is displayed in the line editor.

- 2-7 Add a new column called "Discount." Define new tabs for the corresponding paragraph formats. Enter the discounts in this column.
- 2-7-1 Edit the text elements of the MAIN window as before.
- 2-7-2 In the header line after price, add a tab followed by "Discount".
- 2-7-3 Switch to the line editor.. The tab is shown there as a double comma (,,).
- 2-7-4 Switch back to the graphical editor.
- 2-7-5 Add a tab and a discount amount for some of the item lines.
- 2-7-6 Choose *Back*.
- 2-7-7 Choose *Goto -> Paragraph Formats* to switch to paragraph maintenance.
- 2-7-8 Choose the IH paragraph.
- 2-7-9 Change the tabs by choosing *Tabs* and entering your own alignment and position for the discount column.
- 2-7-10 Proceed in exactly the same way for the IL paragraph.
- Activate the form via *Form -> Activate* and test it via *Form -> Check*.

Contents:

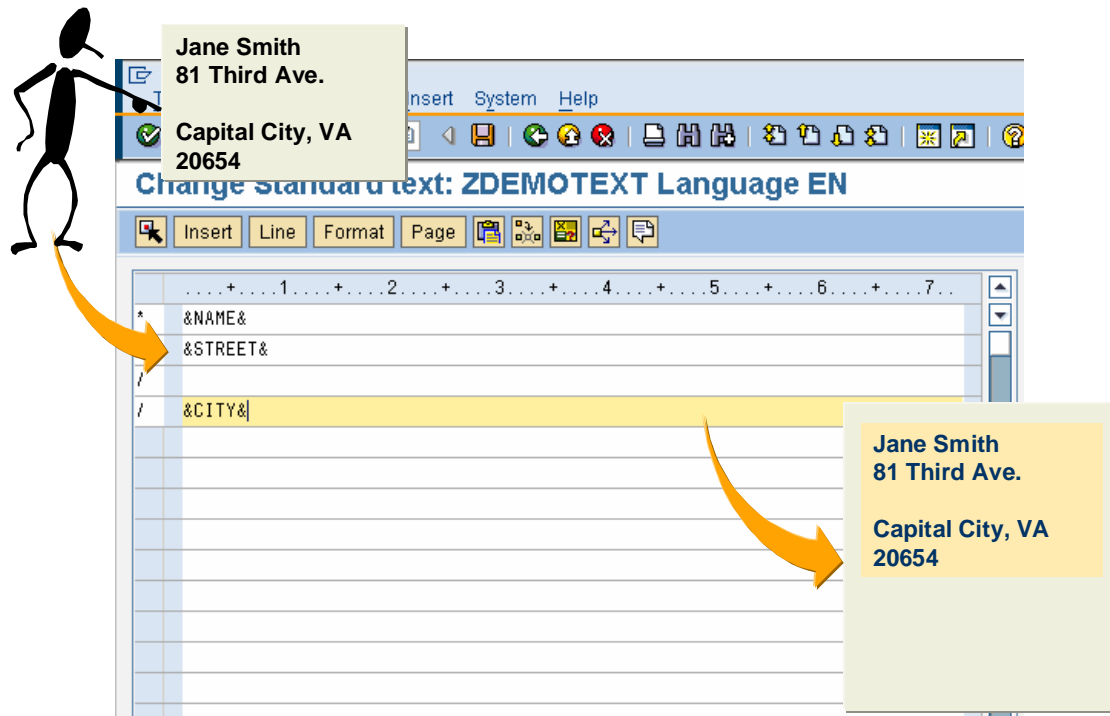
- Using symbols
- Types of symbols
- Formatting options with symbols
- Control commands in the SAPscript Editor

© SAP AG 1999



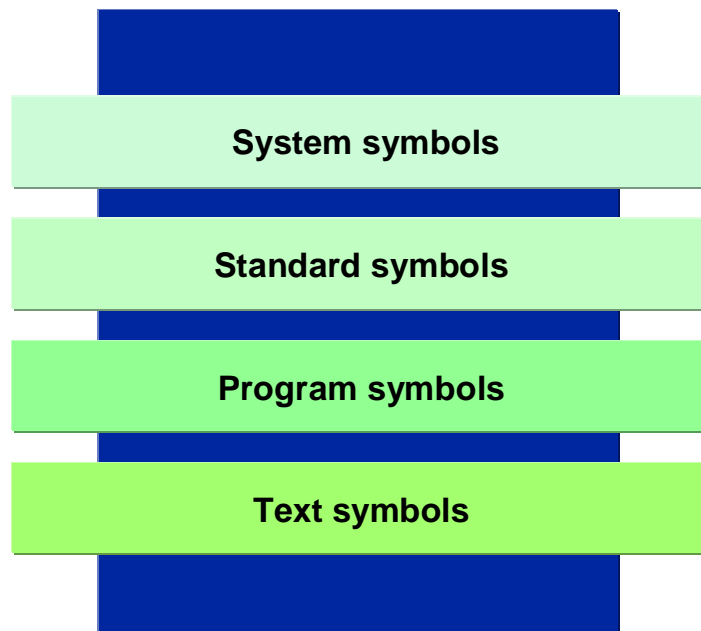
At the conclusion of this unit, you will be able to:

- **Use SAPscript symbols**
- **Describe the difference between the types of symbols**
- **Include the various formatting options**
- **Control the output of a SAPscript text using control commands in the editor**



© SAP AG 2006

- Symbols are placeholders for values that are inserted during print formatting.
- The use of symbols allows you to output the most current values from SAP tables in the form of text modules.
- Symbols are identified by their name, which appears between two '&' symbols. No distinction is made between upper case and lower case.
- Symbol names cannot contain blank spaces.
- Symbols must always completely fit on a single text line; they are not allowed to wrap to the next line of text.



© SAP AG 2002

- SAPscript uses four different types of symbol:
 - System symbols
 - Standard symbols
 - Program symbols
 - Text symbols
- These symbol types differ in the way in which values are assigned to them.
- You can select symbols by choosing *Include* → *Symbols*.

DATE	Date
DAY	Day
NAME_OF_DAY	Name of day
MONTH	Month
NAME_OF_MONTH	Name of month
YEAR	Year
TIME	Time
HOURS	Hours
MINUTES	Minutes
SECONDS	Seconds
PAGE	Page number
NEXTPAGE	Number of next page
DEVICE	Output device
SPACE	Blank space
ULINE	Underline
VLIN	Vertical line

© SAP AG 2006

- SAPscript provides standard system symbols that are automatically replaced with data from the SAP System when a document is printed.
- You can use system symbols in any document.
- To choose a system symbol, choose *Include* → *Symbols* → *System*.
- From Release 4.5 A, the table TTXSY, which contains system symbols, is also available to users.

Table TTDTG

Symbols are user-defined

Symbols are language-dependent

Name: max. 22 characters

Value: max. 60 characters

© SAP AG 2002

- Standard symbols are application-defined. They are maintained centrally in table TTDTG.
- You can display or change standard symbols either by choosing *System → Services → Tablemaintenance → Extended table maint.*, or by choosing *Tools → SAPscript → Administration → Settings*.
- You can use standard symbols in any document.
- Standard symbols are language-specific.
- Examples of standard symbols are:
 - &SGDH& for the opening salutation: "Dear Sir/Madam:"
 - &MFG& for the closing salutation: "Yours faithfully"

Value is defined in the text module

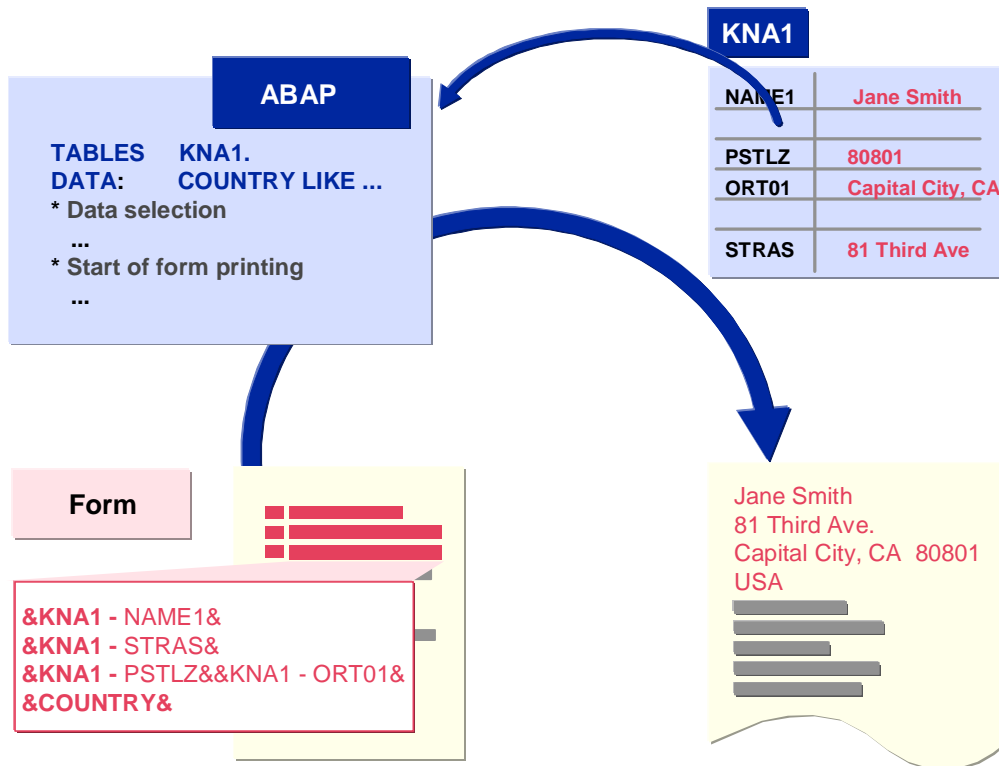
DEFINE &symbol& = 'value'

Name: max. 32 characters

Value: max. 60 characters

© SAP AG 2002

- You can define a text symbol for any text module. This symbol is valid only in the text module for which you have defined it.
- You can assign values to a symbol in the following two ways:
 - Use DEFINE.
 - The value assigned to the symbol is saved when you save the text module.
 - To assign several different values to the same symbol, use DEFINE every time you assign a new value.
 - Use a standard text symbol. To do so, choose *Include* → *Symbols* → *Text symbols* ...
 - The system displays all available text symbols in the current text and/or -form.
 - You can assign any value to the symbols.
 - The value assigned to a text symbol is only temporary. This value is not saved together with the text.



© SAP AG 2002

- Program symbols are substitute symbols for values, which originate from the ABAP program.
- Program symbols are for contents of data base fields or global program symbols. The necessary data base fields must be defined in a table or a structure in the ABAP Dictionary and be defined in the print program with a TABLES statement. Global program symbols can be defined in the print program over DATA, SELECT OPTION, and so forth.
- The data objects must be filled with values by the print program. The editing of the values effected via SAPscript in accordance with the ABAP Dictionary or in the program defined characteristics.

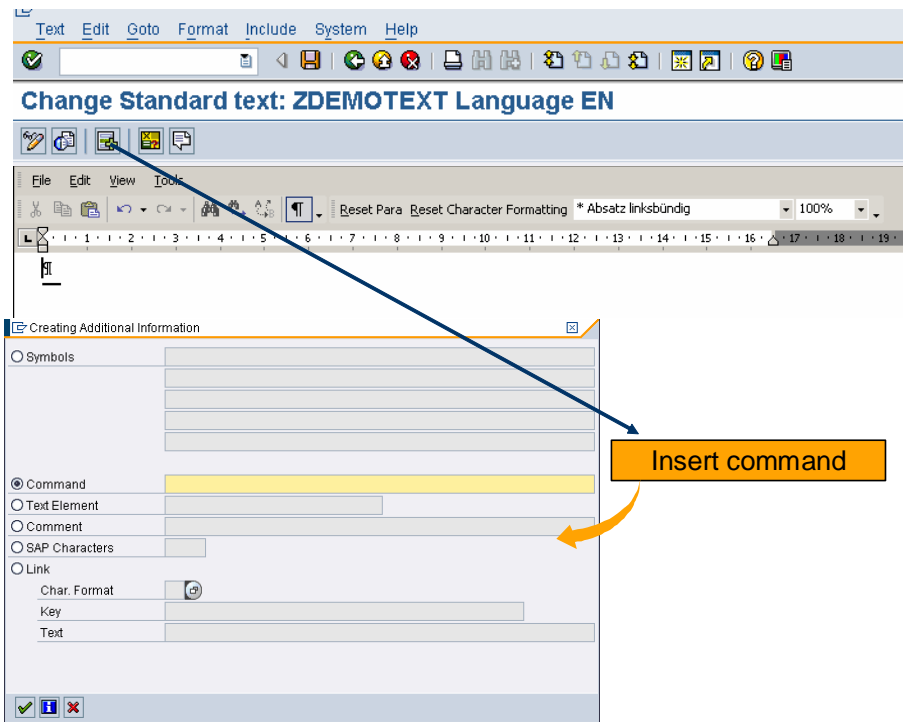


-
- 5-9

&SYMBOL&	
&SYMBOL+4&	Offset
&SYMBOL(5)&	Length
&SYMBOL(I)&	Suppress initial value
&SYMBOL(Z)&	Suppress leading zeros
&SYMBOL(C)&	Compress blank spaces
&SYMBOL(R)&	Right-aligned output
&SYMBOL(S)&	Suppress operators
&SYMBOL(*)&	Dictionary length
&SYMBOL(8.2)&	Decimal format
&'text1'SYMBOL'text2'&	Text before and after

© SAP AG 2002

- The value of a symbol is displayed in full. However, blank spaces at the end of the value are cut off. To change the symbol display, use the following additional options:
 - Offset Output begins here. Offset always refers to the formatted value.
 - Length Data of a specified length is output
 - I If the field has initial value, nothing is output
 - ZLeading zeros are suppressed
 - CSeveral consecutive blank spaces are compressed into a single blank space. Leading blank spaces are suppressed.
 - RThe output is right-justified
 - SThe sign is hidden
 - (x,y) Decimal notation: the data has x length with y decimal places
- Text can also be inserted before or after a symbol, for example: **&'text1'SYMBOL'text2'&**.
- You can also combine formatting options.
For more formatting options, refer to the online documentation.



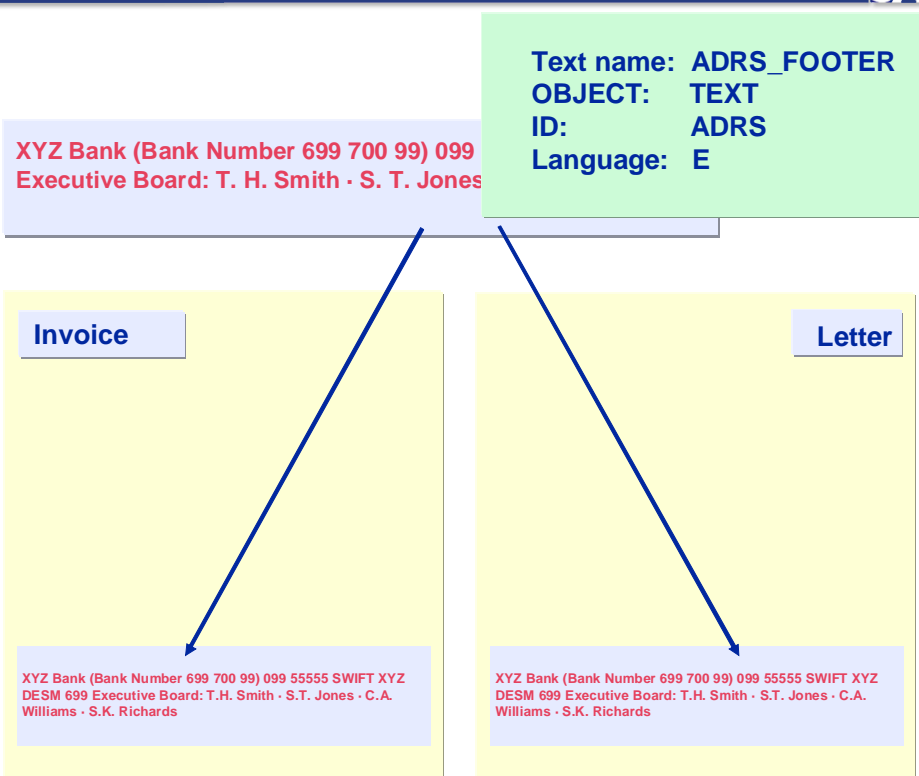
© SAP AG 2009

- You can insert commands, symbols, and text elements in the graphical editor using *Edit -> Command -> Insert Command*.
- The structures you insert appear in fields on the screen.
- You can delete or change the inserted structures by placing your cursor on the appropriate line and then entering the required changes in the additional information.
- Commands and symbols inserted using the dialog box undergo a syntax check before they are included in your text.
- You can also check syntax manually using *Text -> Check*.
- As of Release 4.0C you can also check unknown symbols in form texts.

```
/: INCLUDE  
/: DEFINE  
/: ADDRESS ... ENDADDRESS  
/: PROTECT ... ENDPROTECT  
/: NEW-PAGE  
/: IF ... ENDIF  
/: CASE ... ENDCASE  
...
```

© SAP AG 2006

- You can use control statements to further influence SAPscript text output. Functions provided by these statements include:
 - Including texts
 - Assigning a value to text symbols
 - Starting or suppressing a page break
 - Resetting outline paragraphs
 - Evaluating conditions
- To enter a control statement, use the format key **/:** in the format column. eingeleitet.
- The length of a control statement must not exceed one line.
- SAPscript has a SAPscript debugger that you activate in the initial form maintenance screen (SE71) via *Utilities -> Activate Debugger*. This is then always started when you print forms. To deactivate the debugger you have to explicitly end SAPscript debugging.



© SAP AG 2002

- A form often contains predefined text modules such as materials lists, customer descriptions, or a company footer, which includes bank account numbers, the names of the executive board members, and so on.
- You enter and store these text modules separately.
 - You can combine and display these modules in a form
 - You can use a text module in more than one form
- To identify a text, specify the client, name, language, text object and text ID.
- The text object and text ID are used to classify texts.
- A text object describes the context of a document, such as Item Text: Purchasing Document, Materials Text, or Standard Text. Such text can be used for more than one application.
- Text objects are divided into text IDs to permit a more precise classification of text types. For example, the category, Sales and Distribution Texts is a subcategory of Materials Texts.
- To work on any standard document - that is, all texts allocated to Object TEXT - choose *Tools* → *SAPscript* → *Standard text*.

```
/: INCLUDE ADRS_FOOTER OBJECT TEXT ID ADRS
```

...

Form: RVINVOICE01

Window : FOOTER

```
/: INCLUDE &T001G-TXTFU& OBJECT TEXT ID ADRS
```

...

INCLUDE *name* [*parameter*]

parameter:

- OBJECT
- ID
- LANGUAGE
- PARAGRAPH

© SAP AG 2002

- You can include text modules from the current client in any window in a form. You can also include text modules in other text modules. In either case, use INCLUDE.
- You must specify the name of the text you want to include.
- You can specify additional key fields for the text using the OBJECT, ID, and LANGUAGE parameters.
- If you use the PARAGRAPH parameter to specify a paragraph format in a form, this will be used as the default format in the inserted text.
- Use *Insert* → *Text* → *Standard* ... to include standard text. Enter the name of the text that you want to include in the dialog box that appears.

Form: RVINVOICE01

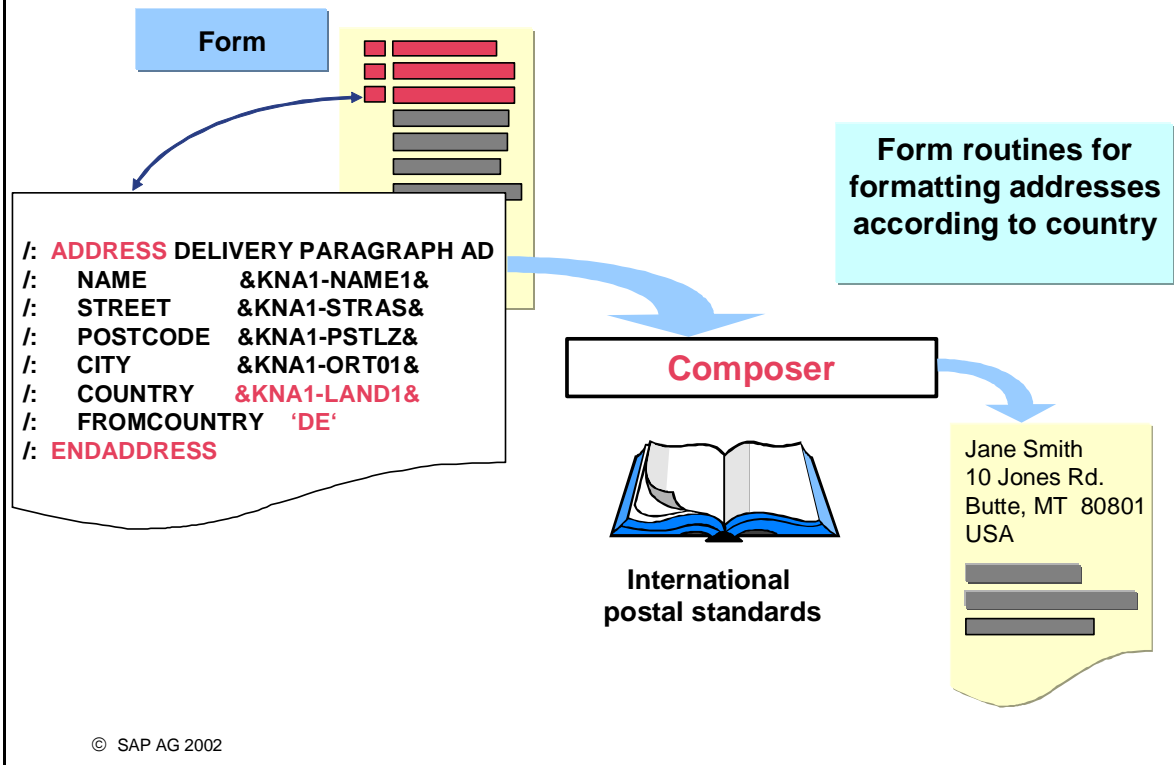
Window: FOOTER

```
/ : DEFINE &SYMBOL& = 'String1 String2 String3'
```

...

© SAP AG 1999

- You must specify values for text symbols explicitly. The DEFINE command allows you to anchor the value you assign in a text.
- To assign values to text symbols, use DEFINE before the text symbol is displayed for the first time.
- Text symbols are retained in the document. This means that the text symbol remains in the text module the next time you call it.
- In addition, you can easily change the value of a text symbol that appears repeatedly in your text.



- The ADDRESS-ENDADDRESS command formats addresses according to the postal norms of the recipient's country as defined in the COUNTRY parameter. The corresponding references fields are described in structure ADRS. You can assign both literals and symbols to the parameter.
- The composer calls the function module ADDRESS_INTRO_PRINTFORM when formatting addresses and transfers the appropriate format parameter from the text to the function module.
- For additional information, refer to the online documentation for function module ADDRESS_INTRO_PRINTFORM.
- You should always include the FROMCOUNTRY parameter when formatting addresses; otherwise the sender's address will be formatted according to the COUNTRY parameter. For example, with postal codes, if COUNTRY = Germany and no FROMCOUNTRY parameter is included, the sender's postal code will be output as D-80801.


```
/: SET TIME MASK = '.....'
```

```
/: SET DATE MASK = '.....'
```

```
/: SET COUNTRY '.....'
```

Examples:

```
/: SET TIME MASK = 'HH : MM'
```

```
/: SET DATE MASK = 'DD.MMMM YYYY'
```

```
/: SET COUNTRY 'US'
```

© SAP AG 2006

- Time, date, and decimal formats are user-specific.
- To influence the format of program and system symbols of this kind, use the following control statements:
 - SET TIME MASK
 - controls the format of time fields
 - SET DATE MASK
 - controls the format of date fields. For example, enter:
/: SET DATE MASK = 'DDDD, the DD MMMM YYYY', and the date field is shown in the output as follows: Monday, the 26 November 2007. If you enter DDD or MMM, the result is an abbreviated form: Mon and Nov.
 - SET COUNTRY
 - provides a choice of country-specific date or decimal formats that differ from the user master data
- The formatting masks must be defined as command lines. All fields are then output according to the format specified.

Start new form page:

/: NEW-PAGE [page]

Protection against page break:

/: PROTECT

...

/: ENDPROTECT

© SAP AG 1999

- SAPscript automatically inserts a page break when the main window (MAIN) is full.
- To override the automatic page break in the main window and insert a manual page break, use the control command NEW-PAGE.
- You can specify the subsequent page explicitly. If you do not, the page defined in the form is used as the next page.
- To avoid unwanted page breaks, use PROTECT...ENDPROTECT. All text lines enclosed by these commands are printed on one page.
- If the text lines fit onto the current page, they are printed on that page as if the PROTECT command had not been used. If there is not enough room on the page, the PROTECT command acts as a NEW-PAGE command and inserts a page break.

Conditional text output:

Case distinction:

```

/ : IF condition
:
/ : ELSEIF condition
:
/ : ELSE
:
/ : ENDIF

/ : CASE symbol
/ : WHEN value
:
/ : WHEN OTHERS
:
/ : ENDCASE
    
```

© SAP AG 2006

- You can use the IF control statement to specify that text lines are only output under certain conditions.
- If the logical expression specified in the IF control statement is fulfilled, text lines that occur between IF and ENDIF are output. If not, they are ignored.
- The following comparison operators are permissible in the condition:

=	EQ	equal to	>=	GE	greater than or equal to
<	LT	less than	<>	NE	not equal
>	GT	greater than	NOT, AND, OR comparison operators		
<=	LE	less than or equal to			
- These comparisons are text comparisons only. **Important:** If numerical values are being compared, always also specify the lengths of the variables. Otherwise, this might produce a different result. Example: /: DEFINE &var1& = '2' and /: DEFINE &var2& = '10'. /: IF &var1(2)& > &var2(2)& -> condition is not met. If you did not specify the length in brackets, the condition would be met.
- To distinguish between dual-level and multilevel cases, use ELSE and ELSEIF within IF...ENDIF.
- CASE is a special type of multilevel case distinction using IF control statements. For the various cases, only one symbol can be tested for equivalence with different values.



- 5-20

Form

```

/: PERFORM <form> IN PROGRAM <prog>
/: USING &invar1&
/: USING &invar2&
.....
/: CHANGING &outvar1&
.....
/: ENDPERFORM
    
```

IN_TAB
structure ITCSY

NAME	VALUE
invar1	...
invar2	...

ABAP
Dictionary

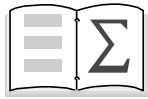
```

REPORT <prog>.
FORM <form> TABLES in_tab  STRUCTURE itcsy
                   out_tab STRUCTURE itcsy.
. . .
ENDFORM.
    
```

ABAP
report

© SAP AG 2002

- Use the PERFORM command to transfer SAPscript data to programs and program data into SAPscript without having to change your print program.
 - Example: You want to use data from a SAPscript database table that cannot be read by the print program assigned to it. Have a program in customer namespace Z read the data and then use the PERFORM command in SAPscript to call the program.
- The form routine called during this procedure has to use an internal table based on the structure ITCSY.
- The structure ITCSY is composed of two fields, NAME and VALUE. Text fields are transferred into the NAME field and text field contents into the VALUE field.
- The variables listed under the PERFORM command are symbols.
- All symbols (parameter name and contents) that are transferred from SAPscript to the program being called are listed in the PERFORM command using USING.
- Parameters transferred back to SAPscript from internal tables in programs are received by the PERFORM command as text symbols using the keyword CHANGING.

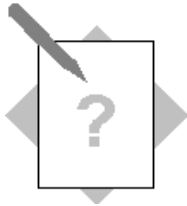


You are now able to:

- **Include SAPscript symbols**
- **Differentiate between the various symbol types**
- **Include several different formatting options**
- **Use control commands in the editor to prepare a SAPscript text for output**

© SAP AG 2001

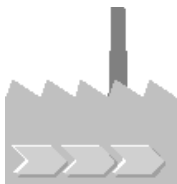
Exercises



Unit: Symbols and Control Statements




- Changes to SAP standard forms
- Layout options using the graphical Form Painter and the alphanumeric Form Painter
- Text changes
- Print preview using the print program



In most cases, standard forms are altered to reflect the needs of individual customers.

There are two alternative exercises for this unit

<i>Activities</i> 	Object	Name / Description
	Name of the form to be created	Z_BC460_EX4_##
	Name of the form to be used	SAPBC460T_FM_04
	Name of the print program to be used	SAPBC460D_01
	Name of the program to be created	Z_BC460_EX4_##
	Package	\$TMP or ZBC460_##
	Note: ## is your group number	

- 1 Create a new form with the name Z_BC460_EX4_## for this exercise.
 - 1-1 To do this, use form SAPBC460T_FM_04 as a template by choosing *Form – Copy from...* (see PC editor and line editor exercise).
- 2 In the INTRODUCTION text element in the MAIN window, change the form of address to print the name of the customer. Use a text symbol that is filled from your own program. The print program is not able to access the name of customer.
 - 2-1 Assign the value '00000002' to a new symbol called &CUST& in the MAIN window using the DEFINE command.

- 2-2 Read the name of this customer from the SCUSTOM table in a form routine of an ABAP program. Use the PERFORM command in SAPscript to get the data from your own program and insert it into the form. (You create the program executed here later.) Assign this value to the new symbol &NAME&.
- 2-3 Change the form of address to the &NAME& symbol.
- 2-4 Now write the ABAP program Z_BC460_EX4_##. Note that the program must have the following structure:

```
REPORT Z_BC460_EX4_##.
FORM GET_NAME TABLES INTTAB STRUCTURE ITCSY
                        OUTTAB STRUCTURE ITCSY.
```

- * Read entry from the imported internal table
- * Select the required data record from table SCUSTOM using the customer ID in the inttab value (8) field.
- * Write the selected data record back to the exporting internal table

```
ENDFORM.
```

- 2-5 Test your changes using the program SAPBC460D_01.
 - 2-6 You may wish to use the DEBUG mode to check the values of INTTAB and OUTTAB.
- 3 Place a box around the SENDER window. This should be a little larger than the window so that it does not cover any of the text in the window.
- 3-1 Choose the SENDER window and choose *Text*.
 - 3-2 Use the POSITION command to set the x and y origin to -0.2 cm. This positions the upper-left corner of the box 0.2 cm left and 0.2 cm up from the upper-left corner of the window.
 - 3-3 Use the SIZE command to set the width and height to +0.4 cm. This makes up for the upper position and adds another 0.2 cm to the window width and height.
 - 3-4 Use the BOXFRAME command to draw the box with a line width of 10 TW.
 - 3-5 Test your changes using the program SAPBC460D_01.
- 4 Create a standard text with the name Z_BC460_EX4_## and text ID SDVD. Copy this text to your form. The text should be printed in the center.
- 4-1 Create a standard text via *Tools -> SAPscript -> Standard Text*. Change the text ID to SDVD (standard text for SD forms).
 - 4-2 Enter a few lines of text and save. **Note:** Since you want to control the paragraph formatting from within the form, do not choose a paragraph format.
 - 4-3 Use the INCLUDE command to insert the standard text that you want to be printed in your form. **Note:** You can insert your newly created text module easily by choosing *Insert -> Text -> Standard*.
 - 4-4 Add the parameter PARAGRAPH to select the paragraph tag that will make the text be printed in the center.
 - 4-5 Test your changes using the program SAPBC460D_01.

Alternative to Previous Exercise

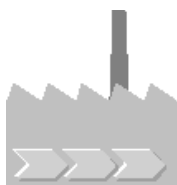
- 1-1 Copy form SAPBC460T_FM_04 to Z_BC460_EX5_##. Save the new form either as a local object or in package ZBC460_## (## stands for your two-digit group number).
If you want to test your results straight away, use print program SAPBC460D_01 (after every change, you have to activate the form and completely exit the print program).
- 1-2 In transaction SE75, create the new **standard symbol** zform##. Enter the value as "Dear Customer". Delete the previous form of address in the form and instead use the new standard symbol.
Note the following: Other participants want to access table TTDTG as well - do not stay in table maintenance too long.
- 1-3 Use transaction SO10 to create a **standard text** (text module) and include it in your form. To do so, follow the activities described in task 4 in the first of the two alternative exercises here.
- 1-4 Set the **date format** to "Walldorf, the 1 November 2007."
- 1-5 Enter the page number in the form "Page X of Y".
- 1-6 When the system user is called BC460-##, the following text should appear after the closing greeting form: "sgd. Smith". Get the suitable system variable in an IF condition and output the text if you wish.
- 1-7 Draw a **box** around the SENDER window. To do so, follow the activities described in task 3 in the first of the two alternative exercises here.
- 1-8 Read customer discount
Your form is always printed for customer number 2 (Andreas Lotz). At the bottom of the MAIN window, provide a PS containing the percentage of discount that this customer has received. The value can be found in table SCUSTOM, field DISCOUNT. Unfortunately, the print program is unable to access this field. Therefore, you have to read it using an external PERFORM from the SAPscript form.
- 1-8-1 Use the DEFINE command in the main window to assign the customer number (2) to the new symbol &CUSTOMER&.
- 1-8-2 Use this symbol in a PERFORM call from within your form. Use the &RABATT& symbol as a return field. The ABAP supporting program should be called ZBC460_##, and its subroutine GET_DISCOUNT.
- 1-8-3 After the external PERFORM, write a PS as follows: "You have received &RABATT& per cent discount."
- 1-8-4 Now write the ABAP supporting program ZBC460_## with subroutine GET_DISCOUNT. Assign the correct type to the interface.
- 1-8-5 The three required steps in the subroutine are:
a) Which customer number was transferred? (1. Evaluate itab)
b) How much discount does this customer have? (Evaluate database table SCUSTOM)
c) Write back read value (to second itab)
- 1-8-6 Activate the form and supporting program, and test the result with report SAPBC460D_01.



Unit: Symbols and Control Statements



- Changes to SAP standard forms
- Layout options using the graphical Form Painter and the alphanumeric Form Painter
- Text changes
- Print preview using the print program



In most cases, standard forms are altered to reflect the needs of individual customers.

- 1 Create a new form with the name Z_BC460_EX4_## for this exercise.
 - 1-1 To do this, use form SAPBC460T_FM_04 as a template by choosing *Form* → *Copy from...* (see PC editor and line editor exercise).
 - 1-1-1 The solution form is called SAPBC460S_FM_04.
- 2 In the INTRODUCTION text element in the MAIN window, change the form of address to print the name of the customer. Use a text symbol that is filled from your own program. The print program is not able to access the name of customer.
 - 2-1 Assign the value '00000002' to a new symbol called &CUST& in the MAIN window using the DEFINE command.
 - 2-1-1 In the PC editor in the MAIN window under the INTRODUCTION text element, choose *Edit* → *Command* → *Insert Command*. Choose *Command* and enter the following in the corresponding input area:

DEFINE &CUST& = '00000002'
 - 2-1-2 Alternatively, enter the following command in the line editor:

/: DEFINE &CUST& = '00000002'

- 2-2 Read the name of this customer from the SCUSTOM table in a form routine of an ABAP program. Use the PERFORM command in SAPscript to get the data from your own program and insert it into the form. (You create the program executed here later.) Assign this value to the new symbol &NAME&.

2-2-1 Then add the following command lines to the command DEFINE:

```
/: PERFORM GET_NAME IN PROGRAM Z_BC460_EX4_##  
/: USING &CUST&  
/: CHANGING &NAME&  
/: ENDPERFORM
```

- 2-3 Change the form of address to the &NAME& symbol.

2-3-1 Switch to the PC editor and the MAIN window under the INTRODUCTION text element.

2-3-2 Remove the text to be replaced.

2-3-3 Place the cursor on the position in the line at which you want the name to appear.

2-3-4 Choose *Edit -> Command -> Insert Command*.

2-3-5 Choose *Symbols* and enter the following in the corresponding input area:

&NAME&

2-3-6 Alternatively, enter the following text in the line editor:

* Dear Mr. &NAME&,

- 2-4 Now write the ABAP program Z_BC460_EX4_##. Note that the program must have the following structure:

```
REPORT Z_BC460_EX4_##.  
FORM GET_NAME TABLES INTTAB STRUCTURE ITCSY  
                        OUTTAB STRUCTURE ITCSY.
```

- * Read entry from the imported internal table
- * Select the required data record from table SCUSTOM using the customer ID in the inttab value (8) field.
- * Write the selected data record back to the exporting internal table

```
ENDFORM.
```

2-4-1 The program for the solution is called SAPBC460S_04.

The code of this program is as follows:

```
REPORT sapbc460s_04 .

TABLES: scustom.

*&-----*
*&      Form  GET_NAME
*&-----*

FORM get_name TABLES inttab STRUCTURE itcsy
outtab STRUCTURE itcsy.
* read first line of inttab
READ TABLE inttab INDEX 1.
* select from scustom and modify outtab with new
data
SELECT SINGLE * FROM scustom
      WHERE id = inttab-value(8).
IF sy-subrc = 0.
  READ TABLE outtab INDEX 1.
  MOVE scustom-name TO outtab-value.
  MODIFY outtab INDEX sy-tabix.
ELSE.
  READ TABLE outtab INDEX 1.
  MOVE 'no name' TO outtab-value.
  MODIFY outtab INDEX sy-tabix.
ENDIF.

ENDFORM.                " GET_NAME
```

2-4-2 Perform a syntax check on the changes to the text by choosing *Form -> Check -> Texts*. If required, correct the syntax by placing the cursor on the incorrect text and choosing *Change Command*.

2-5 Test your changes using the program SAPBC460D_01.

- 2-6 You may wish to use the DEBUG mode to check the values of INTTAB and OUTTAB.
 - 2-6-1 The ABAP debugger is not the subject of this course and is therefore not explained here.
 - 2-6-2 Activate the SAPscript Debugger in transaction SE71 (Tools → *SAPscript* → *Form*) by choosing *Utilities* → *Activate Debugger*.
 - 2-6-3 Execute the program and go through the form in the debugger.
 - 2-6-4 The debugger can only be deactivated during the debugger run. Choose *Debugger* → *Exit* to deactivate the debugger.
- 3 Place a box around the SENDER window. This should be a little larger than the window so that it does not cover any of the text in the window.
 - 3-1 Choose the SENDER window and choose *Text*.
 - 3-2 Use the POSITION command to set the x and y origin to -0.2 cm. This positions the upper-left corner of the box 0.2 cm left and 0.2 cm up from the upper-left corner of the window.
 - 3-2-1 Use the following command:

```
/: POSITION XORIGIN '-0.2' CM YORIGIN '-0.2' CM
```
 - 3-3 Use the SIZE command to set the width and height to +0.4 cm. This makes up for the upper position and adds another 0.2 cm to the window width and height.
 - 3-3-1 Insert the following command:

```
/: SIZE WIDTH '+0.4' CM HEIGHT '+0.4' CM
```
 - 3-4 Use the BOXFRAME command to draw the box with a line width of 10 TW.
 - 3-4-1 Insert the following command:

```
/: BOX FRAME 10 TW
```
 - 3-5 Test your changes using the program SAPBC460D_01.
- 4 Create a standard text with the name Z_BC460_EX4_## and text ID SDVD. Copy this text to your form. The text should be printed in the center.
 - 4-1 Create a standard text via *Tools* → *SAPscript* → *Standard Text*. Change the text ID to SDVD (standard text for SD forms).
 - 4-1-1 Text name: Z_BC460_EX4_##
 - 4-1-2 Text ID: SDVD
 - 4-2 Enter a few lines of text and save. **Note:** Since you want to control the paragraph formatting from within the form, do not choose a paragraph format.
 - 4-2-1 The paragraph tag should be *, meaning that you would like to use the default value. This can be overridden from within the form.

- 4-3 Use the INCLUDE command to insert the text that you want to be printed in your form. **Note:** You can insert your newly created standard text easily by choosing *Insert -> Text -> Standard*.

4-3-1 Note that you can search for your text using *Search (F4)*.

4-3-2 After the INCLUDE has been inserted via *Insert -> Text -> Standard*, the command is as follows:

```
/: INCLUDE Z_BC460_EX4_## OBJECT TEXT ID SDVD LANGUAGE  
EN
```

- 4-4 Add the parameter PARAGRAPH to select the paragraph tag that will make the text be printed in the center.

4-4-1 Choose *Change Command* to add the parameter PARAGRAPH. Enter PARAGRAPH C after LANGUAGE EN.

- 4-5 Test your changes using the program SAPBC460D_01.

SOLUTION TO ALTERNATIVE EXERCISE

- 1-1 Copy as before: Create a new form with the name Z_BC460_EX5_## in SE71 and copy the template SAPBC460T_FM_04.
- 1-2 Enter transaction SE75. Choose "Standard Symbols" then "Change". Confirm the security advice that the table is client-independent. Choose "New Entries" and then enter the following:
Language: EN
Symbol Name: zform##
Symbol Value: Dear Customer,

To include the new standard symbol in your form, go to the correct position in the editor and type it in (between two & symbols). Alternatively, you can select it from the list of standard symbols: Menu *Include -> Symbols -> Standard Symbols*.

- 1-3 See the solution description for task 4 in the first of the two alternative exercises.

- 1-4 Enter the following command:

```
/: SET DATE MASK = ' Walldorf, the DD MMMM YYYY'
```

After this, you can output the date yourself. Since you want to suppress the first zero of the date, you have to create the symbol in the following form: &DATE(Z)&

- 1-5 Enter the following as normal text:
Page &PAGE& of &SAPSCRIPT-FORMPAGES&
(If you do not know the names of the symbols by heart, you can select them from the list: Menu *Include -> Symbols -> Standard Symbols*.)
- 1-6 The Coding is: (new line) `/: IF &SY-UNAME& = 'BC460-##' (new line) * "sgd. Smith". (new line) /: ENDIF`

1-7 The coding is:
/: POSITION XORIGIN '-0.2' CM YORIGIN '-0.2' CM
/: SIZE WIDTH '+0.4' CM HEIGHT '+0.4' CM
/: BOX FRAME 1 PT

1-8 The SAPscript coding is:
/: DEFINE &CUSTOMER& = 2
/: PERFORM GET_DISCOUNT IN PROGRAM ZBC460_##
/: USING &CUSTOMER&
/: CHANGING &RABATT&
/: ENDPERFORM
* You have received &RABATT(C)& per cent discount.

The ABAP coding is:

```
FORM get_discount
  TABLES intab STRUCTURE itcsy
          outtab STRUCTURE itcsy.

  DATA: discount TYPE scustom-discount.

  * 1. What was transferred?
  * READ TABLE intab INDEX 1.
  READ TABLE intab WITH KEY name = 'CUSTOMER'.

  * 2. Evaluation (read data)
  SELECT SINGLE discount
    FROM scustom
    INTO discount
    WHERE id = intab-value.

  * Eliminate first zeros
  * (since outtab-value is a character variable,
  * the SAPscript formatting option (Z) does not work.)

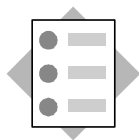
  READ TABLE outtab INDEX 1.
  WRITE discount TO outtab-value LEFT-JUSTIFIED NO-ZERO.

  * 3. Write back result
  MODIFY outtab INDEX 1.
ENDFORM.
```


Contents:

- Tasks of the print program and the composer
- Structure of the print program
- Important function modules
- Printing text elements
- Processing headings
- Composer procedure

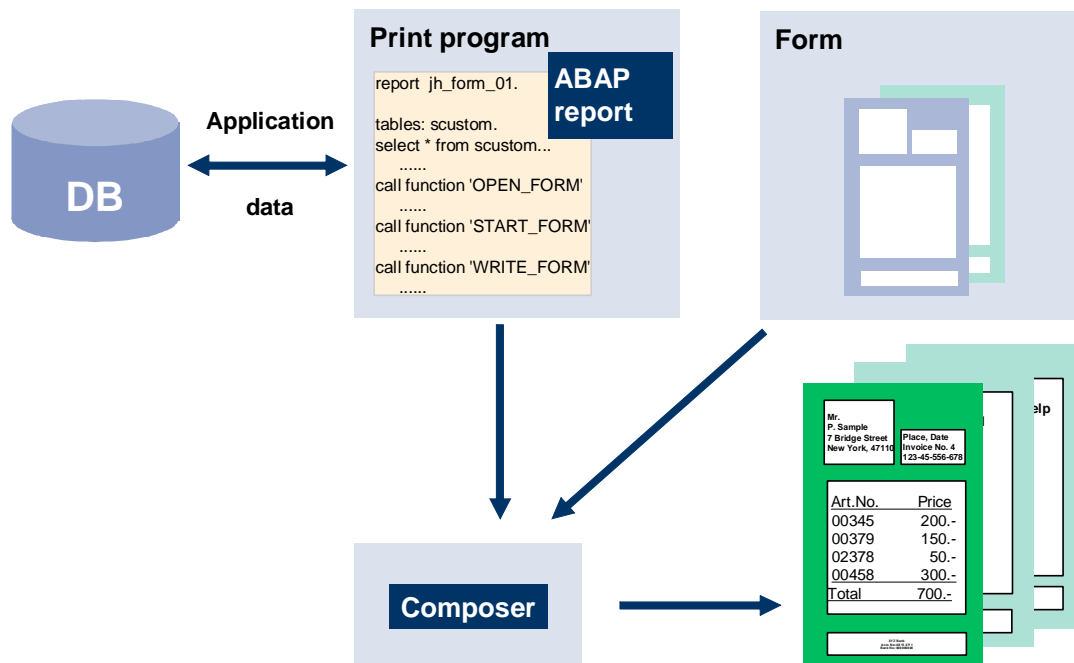
© SAP AG 1999



At the conclusion of this unit, you will be able to:

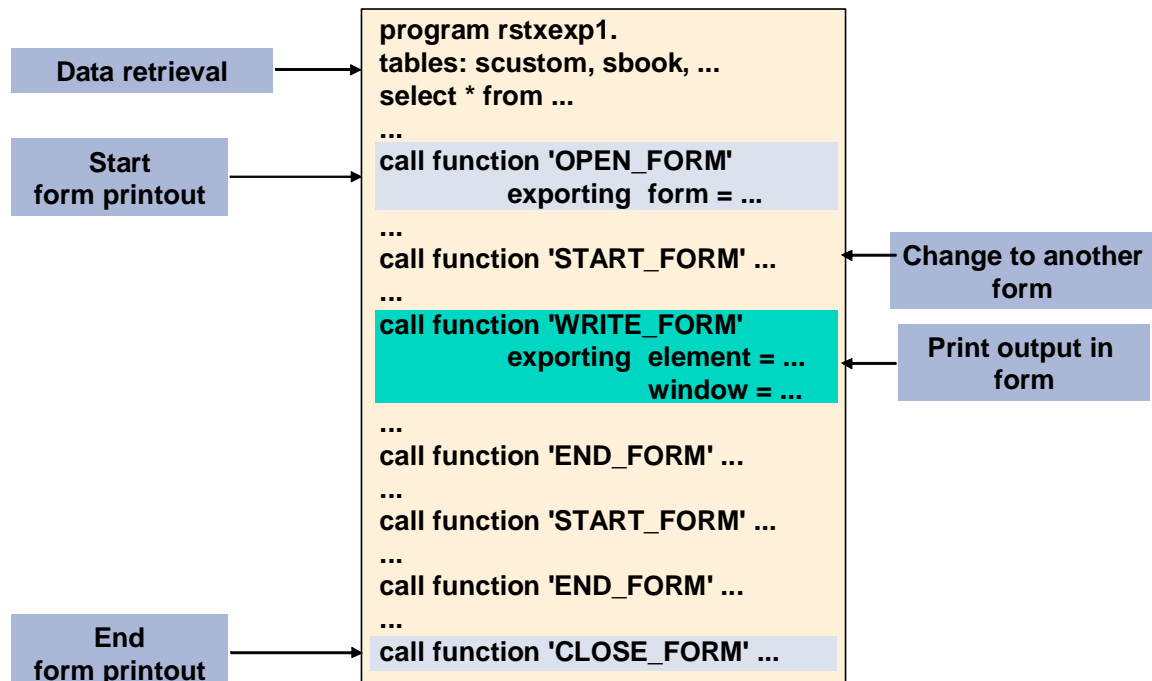
- **Explain the structure of print programs in SAPscript**
- **Explain how print programs and forms work together**
- **Name the most important SAPscript function modules**
- **Write text element output in forms**
- **Describe how the SAPscript composer works**

© SAP AG 2001



© SAP AG 2006

- In the SAP system, both the print program and the form template are needed to print documents. The print program is either an ABAP report or a module pool.
- The print program:
 - retrieves SAP application data from the database
 - defines the form processing logic (the order and repetition of text elements)
 - chooses a form template for printing
 - selects the output device, such as printer, monitor, or fax, e-mail, as well as special print properties, such as immediate output, number of copies, pages to be printed
- The SAPscript composer is responsible for the actual print output, and in particular controls:
 - line and page breaks in the form
 - formatting of SAPscript control statements and symbol variables
 - the inclusion of long texts
 - the creation of the page description for the spool request (Final Format or OTF).



© SAP AG 2006

- SAP applications such as FI, CO, and MM are supplied with standard forms and print programs that can be customized to meet the needs of the customer.
- Within the print program you must distinguish between:
 - data retrieval from the SAP system
 - starting and ending form printing
 - changing to different forms
 - printing out text into the form
- Every time you print a form, you must use the paired function modules OPEN_FORM and CLOSE_FORM. A print program can also print several forms which are either administered individually as separate spool requests or combined into a single spool request. To combine forms into a single spool request, use the START_FORM and END_FORM function modules.
- To print text elements in particular form windows, use the WRITE_FORM function module.
- To transfer control statements to a form, use the CONTROL_FORM function module.

```
CALL FUNCTION 'OPEN_FORM'
  EXPORTING
    FORM          = ...
    LANGUAGE      = ...
    DEVICE        = ...
    OPTIONS       = ...
    DIALOG        = ...
  IMPORTING
    LANGUAGE      = ...
  EXCEPTIONS ... = ...
```

```
CALL FUNCTION 'CLOSE_FORM'
  IMPORTING
    RESULT        = .
  ..
  EXCEPTIONS ... = .
  ..
```

© SAP AG 2002

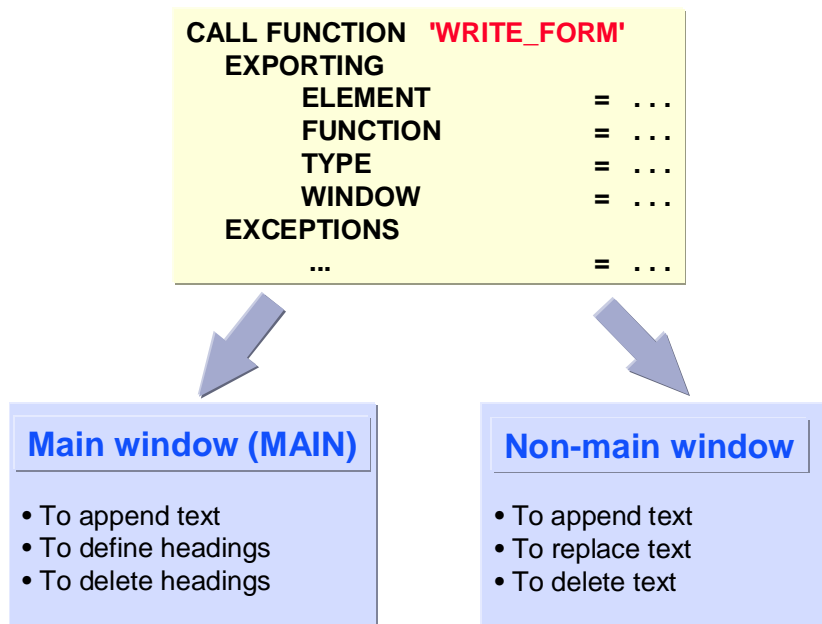
- The function module OPEN_FORM initializes form printing. You must call OPEN_FORM before you can work with any of the other form function modules.
- To specify the form and the desired language, use the FORM and LANGUAGE parameters.
- To control the output channel, use the DEVICE parameter with:
 - PRINTER for print output using spool
 - TELEFAX for fax output using spool or SAPconnect
 - SCREEN for monitor output using GUI. You cannot print the text if you use this value.
- To specify attributes for printing or faxing - such as the number of copies, immediate output, and so on - use the OPTIONS parameter (structure ITCPO).
- To suppress the dialog box for output device parameters, such as the name of the device in DEVICE = PRINTER or DEVICE = TELEFAX, use the DIALOG parameter.
- To end form printing, use the CLOSE_FORM function module.
- Once the form printing has finished, you can obtain status information and the print and fax parameters from the RESULT parameter (structure ITCPP).

```
CALL FUNCTION 'START_FORM'
  EXPORTING
    FORM          = ...
    LANGUAGE      = ...
    STARTPAGE     = ...
  IMPORTING
    LANGUAGE      = ...
  EXCEPTIONS ... = ...

CALL FUNCTION 'END_FORM'
  IMPORTING
    RESULT        = ...
  EXCEPTIONS ... = ...
```

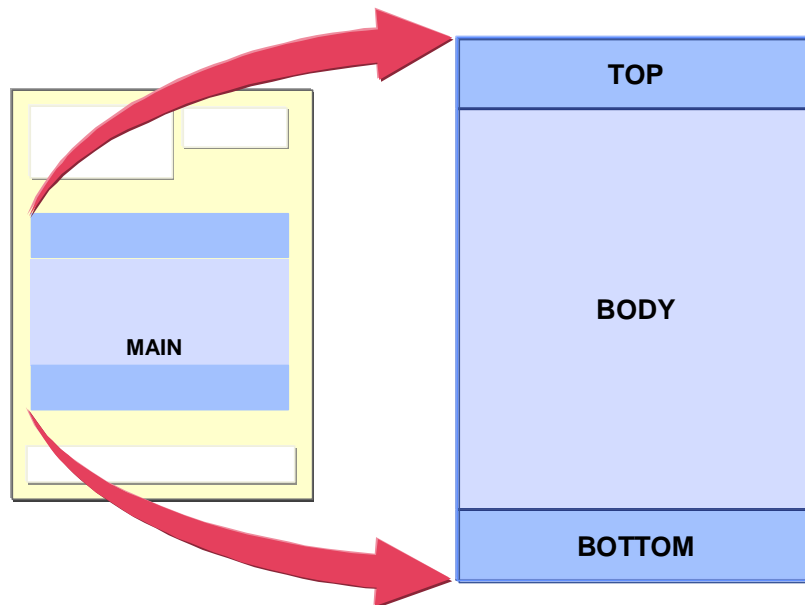
© SAP AG 2002

- To print several identical forms containing different data within a single spool request, begin each form using `START_FORM` and end it using `END_FORM`.
- Before using `START_FORM` for the first time, you must call `OPEN_FORM`.
- The function modules for each form are called between `START_FORM` and `END_FORM`.
- `END_FORM` does not end the printing process. Use either `START_FORM` or `CLOSE_FORM` after `END_FORM`.
Note: `END_FORM` cannot replace `CLOSE_FORM`.
- To specify the form under `START_FORM`, use the `FORM` and `LANGUAGE` parameters.
- To choose a start page other than the default start page, use the `STARTPAGE` parameter.



© SAP AG 1999

- To output text elements in the window of a form, use the function module WRITE_FORM.
- The order in which text elements are printed corresponds to the order in which WRITE_FORM is called. In other words, it is determined by the print program.
- You must specify the desired text element using the ELEMENT parameter.
- The WINDOW parameter specifies the window in which the text element is output.
- The TYPE parameter specifies the output area of the main window.
- The FUNCTION parameter specifies whether text is to be appended, replaced or deleted.



© SAP AG 2002

- You can output text in the main window (type MAIN) in one of three areas: TOP, BODY, or BOTTOM.
- The size of the top and bottom areas depends on the size of the text contained in the field. The amount of space needed for this text is deducted from the BODY area.
- The top and bottom areas of the main window allow you to create headers (titles) and footers (similar to creating page headers and footers in ABAP during print list generation).
- You can enter and change the contents of the TOP, BODY, and BOTTOM areas dynamically during form output. SAPscript automatically outputs the current contents of these areas on every page that contains a main window.

CALL FUNCTION 'WRITE_FORM'
EXPORTING
ELEMENT = ...
EXCEPTIONS ... = ...

/E ITEM_HEADER	
Art.No.,	Price
/E ITEM_LINE	
&vbdpa-matnr&,,	&vbdpa-netpr&

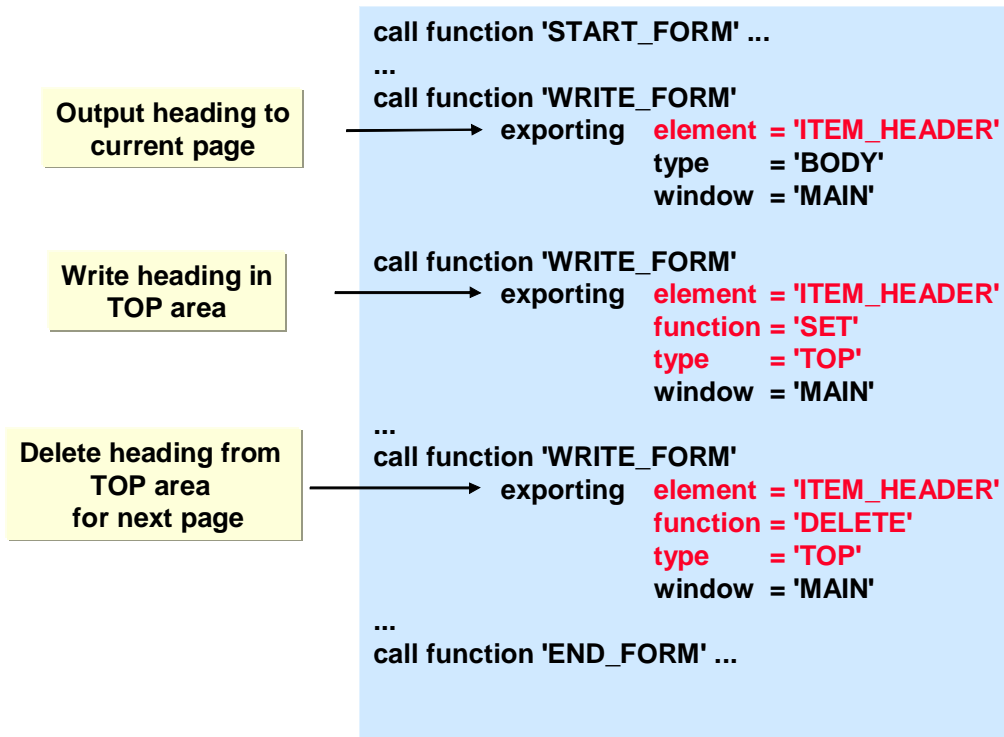
Art.No.	Price
00345	200.-
MAIN	

WRITE_FORM
element = 'ITEM_LINE'
window = 'MAIN'

Art.No.	Price
00345	200.-
00379	150.-
MAIN	

© SAP AG 2006

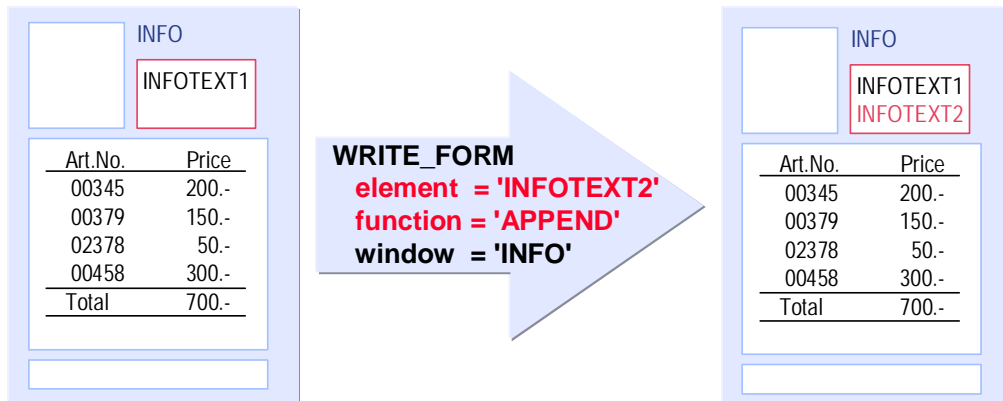
- To print text elements in the main window of a form, use WRITE_FORM.
- The order in which text elements are printed corresponds to the order in which WRITE_FORM is used in the print program.
- You must specify the desired text element using the ELEMENT parameter.
- The WINDOW parameter must be set to MAIN. This is the default value.
- The TYPE parameter must be set to BODY. This is the default value.
- The FUNCTION parameter is not relevant here.
- The output of text elements in the main window triggers a page break as soon as the current page window is full.
- You can also use the WRITE_FORM_LINES function module. To do so, go directly to the LINES parameter and specify the text to be printed.



© SAP AG 2002

- You can add headers to main windows using: WRITE_FORM, TYPE = TOP, FUNCTION = SET.
- If no text has been entered in the BODY area, the header is output to the current page.
- If text **has** been entered in the BODY area, the header is output for the first time at the beginning of the next page.
- After a header has been created, the text elements that are to be output underneath it (for example, a line item from an invoice) are output to the main window using: WRITE_FORM, TYPE = BODY. This output can be several pages in length
- After these text elements have been output in their entirety, the header is deleted by calling: WRITE_FORM, TYPE = 'TOP', FUNCTION = 'DELETE'.
- This deletion first takes effect on the following page. In other words, if there is already text present in that BODY, no header can be subsequently deleted from the TOP area.
- With many application forms, a form of address and short text appear on the first page before the headers are output. This is why, in the example above, the header is first output in the BODY area and then output in the TOP area on subsequent pages.

```
CALL FUNCTION 'WRITE_FORM'
  EXPORTING
    ELEMENT          = ...
    FUNCTION          = 'APPEND'
    WINDOW           = 'INFO'
    EXCEPTIONS ...   = ...
```



© SAP AG 2006

- To add a text element to a non-main window, use WRITE_FORM with FUNCTION = 'APPEND'.
- Unlike text inserted into the main window, any text that does not fit into the current page window is lost.
- As with the main window, the order in which text elements are printed corresponds to the order in which WRITE_FORM is used in the print program.
- You must specify the name of the text element in the ELEMENT parameter.
- You must specify the name of the non-main window in the WINDOW parameter.
- The TYPE parameter is not relevant here.
- The FUNCTION parameter must have the value 'APPEND'.

```
CALL FUNCTION 'WRITE_FORM'
EXPORTING
  ELEMENT      = ...
  FUNCTION     = 'SET'
  WINDOW      = 'INFO'
EXCEPTIONS ...
```

INFO

INFOTEXT1
INFOTEXT2

Art.No.	Price
00345	200.-
00379	150.-
02378	50.-
00458	300.-
Total	700.-

WRITE_FORM
element = 'INFOTEXT3'
function = 'SET'
window = 'INFO'

INFO

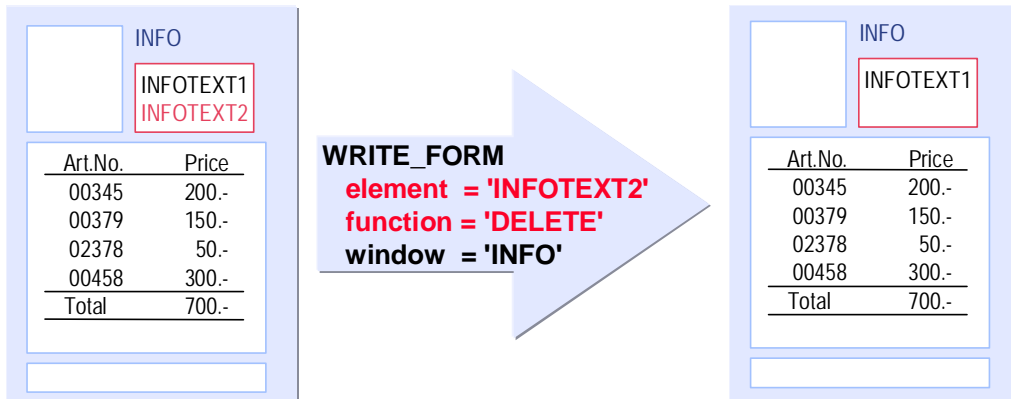
INFOTEXT3

Art.No.	Price
00345	200.-
00379	150.-
02378	50.-
00458	300.-
Total	700.-

© SAP AG 2006

- To replace the current content of a non-main window with a text element, use: WRITE_FORM with FUNCTION = 'SET'.
- You must specify the name of the text element in the ELEMENT parameter.
- You must specify the name of the window in the WINDOW parameter.
- The TYPE parameter is not relevant here.
- The FUNCTION parameter must have the value: 'SET' (which is its default value).

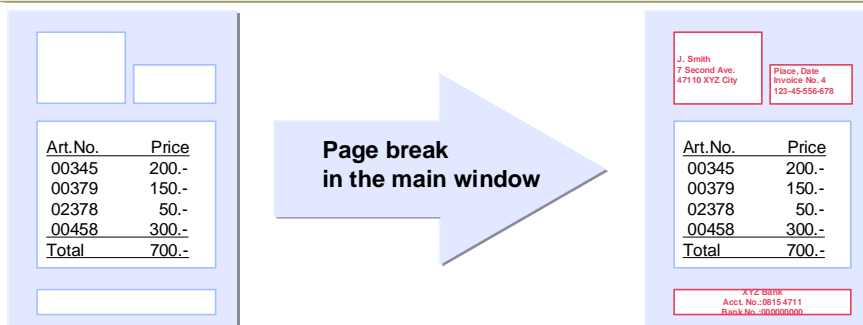
```
CALL FUNCTION 'WRITE_FORM'
  EXPORTING
    ELEMENT      = ...
    FUNCTION     = 'DELETE'
    WINDOW      = 'INFO'
  EXCEPTIONS ...
```



© SAP AG 2006

- To delete a text element from a non-main window, use: WRITE_FORM with FUNCTION = 'DELETE'.
- You must specify the text element in the ELEMENT parameter.
- You must specify the window in the WINDOW parameter.
- The TYPE parameter is not relevant here.
- The FUNCTION parameter must have the value: 'DELETE'.

- The main window controls page breaks.
- Text elements in non-main windows are processed at page breaks or at the end of the layout set.
- Symbols in the BODY area of the main window are replaced with data immediately.
- Symbols in the TOP or BOTTOM area are replaced with data whenever changes are made to that area.
- Symbols in non-main windows are replaced when a page break occurs



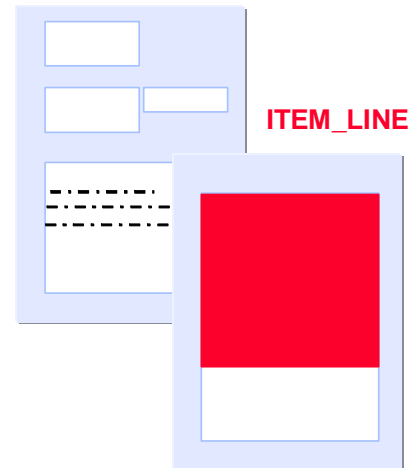
© SAP AG 2002

- The main window controls the page break. The text elements of non-main windows are not processed until a page break.
 - A page break can be triggered by:
 - an overflow of the main window (automatic)
- a NEW-PAGE command in the main window (manual)
- The main window of a form is divided into the TOP, BODY and BOTTOM areas.
- To print text elements in the TOP, BODY or BOTTOM area of the main window, or in non-main windows, use WRITE_FORM.
- A default text element at the beginning of a non-main window is printed once in the window as soon as the window is processed. However, following a page break, default text elements in the main window do not appear in the main window of the subsequent page.

ABAP
report

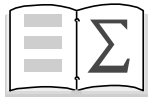
```
CALL FUNCTION 'CONTROL_FORM'
  EXPORTING
    COMMAND      = ...
  EXCEPTIONS ... = ...
```

```
...
CALL FUNCTION 'CONTROL_FORM'
  EXPORTING
    COMMAND = 'PROTECT'.
CALL FUNCTION 'WRITE_FORM'
  EXPORTING
    ELEMENT = 'ITEM_LINE'.
...
CALL FUNCTION 'CONTROL_FORM'
  EXPORTING
    COMMAND = 'ENDPROTECT'.
...
```



© SAP AG 2006

- The function module `CONTROL_FORM` allows you to create SAPscript control statements from within an ABAP program.
- Enter the SAPscript control statement you want to use to control output in the `COMMAND` parameter of the function module without the `/'` addition.
- In the above example, the text element `ITEM_LINE` is protected from an automatic page break. The contents of the text element `ITEM_LINE` will always appear together on one page.
- Normally, you would execute the SAPscript statement `PROTECT ... ENDPROTECT` in SAPscript (see the unit on symbols and control statements).



You are now able to:

- **Describe the structure of print programs in SAPscript.**
- **Understand how the print program and the form interact.**
- **Name the most important SAPscript function modules.**
- **Output text elements in forms.**
- **Understand how the SAPscript composer works.**

© SAP AG 2001

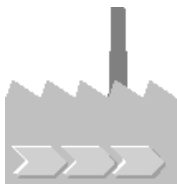
Exercises



Unit: The Print Program




- Calls of function modules for form printing
- Print control via print program
- Enhancements to the print program in conjunction with the related form
- Print preview using the print program



In most cases, standard forms are altered to reflect the needs of individual customers. In some cases, it is then necessary to adjust the print program to meet these needs.

There are two alternative exercises and an optional exercise for this unit

<div>Activities</div> 	Object	Name / Description
	Name of the form to be used	SAPBC460D_FM_03
	Name of the print program to be created	Z_BC460_EX5A_##
	Package	\$TMP or ZBC460_##
	Note: ## is your group number	

1. Exercise

1. Write a print program in which form SAPBC460D_FM_03 is output. Take a close look at the form in the Form Painter so that you know the sequence in which the form's text elements have to be output in the print program.
2. Allow users to specify which page to begin printing by including a selection screen on which they can enter a page number.

NOTE: Choose *Pattern* to adopt all function modules.

2. Exercise (Alternative to Previous Exercise)

- 1-1 Write a print program for form SAPBC460_PRINT.
To do so, copy the template program SAPBC460T_PRINT_TEMPLATE to ZBC460_##. It already contains a selection screen and the required data retrieval. The places where you are required to insert coding are shown by a row of asterisks.

The program should print form SAPBC460_PRINT (booking confirmation) for all customers that the user enters on the selection screen. It is important to remember that each customer may have more than one booking.

You need:

Function modules for the start and end of form printing

Function modules for outputting text elements. Take a look at the form to see which text elements are required.


Function modules for changing between forms (all customers require their own form).

For the sake of simplicity, you can ignore error handling (sy-subrc-Abfrage) here.

(The coding for the following two tasks is *italicized* in the solution.)

- 1-2 Enhance your print program to set column headers for the items and then delete them again at the end.
- 1-3 Your course instructor will tell you the printer to define as the output device.

Optional Exercise

<div>Activities</div> 	Object	Name / Description
	Name of the form to be created	Z_BC460_EX5_##
	Name of the form to be used	SAPBC460D_FM_05
	Name of the print program to be used as a template for copying	SAPBC460T_05
	Name of the print program to be created	Z_BC460_EX5_##
	Package	\$TMP or ZBC460_##
	Note: ## is your group number	

- 1 Copy print program SAPBC460T_05 and complete the report by inserting the function module calls to open the form, to start, and to write data to it.

1-1 Copy program SAPBC460T_05.

1-1-1 To do so, switch to the ABAP editor by choosing *Tools -> ABAP Workbench -> Development -> ABAP Editor*.

1-1-2 Enter the program name SAPBC460T_05 and choose *Program -> Copy*.

1-1-3 Enter Z_BC460_EX5_## and choose *Copy* to copy coding and text elements.

1-1-4 In the *Create Object Directory Entry* window, choose the *Local Object* function or save in your package called ZBC460_##.

- 1-2 Edit the source text of report Z_BC460_EX5_##. This report contains a print program in which the function module calls for SAPscript form output are shown by asterisks. Complete the report by inserting the function module calls. The sections for which you have to insert coding are shown as follows:

***CALL FUNCTION**

*** fill in missing statements**

NOTE: Choose *Pattern* to adopt all function modules.

If you experience difficulties in defining the correct order of the function calls, you can find information in the underlying structure of a print program in the course materials. It may also be useful to check the MAIN window of form SAPBC460D_FM_5. You can determine the order of the text elements by checking the content.

- 1-3 Test your finished program.

Execute your program by choosing *Program* → *Execute*.

The FORM parameter must be filled with the name of the SAPscript form that is to be used for printing.

In this example, do not change the default value SAPBC460D_FM_05.

To check the output of your form, choose *Print Preview*. If you have a printer at your disposal, you can also print the output.

- 2 Create a new form with the name Z_BC460_EX5_## that is to be used by your program.

- 2-1 Copy form SAPBC460D_FM_05 by choosing *Form* → *Copy from...*
(see exercise on the PC editor and line editor)

- 3 Add your program to the list of print programs by choosing *Program Symbols*.

- 3-1 Choose *Include* → *Symbols* → *Program Symbols*.

- 3-2 Choose *Append Print Program*.

- 3-3 Enter the name of your print program.

- 4 Extend the item line (text element ITEM_LINE) in form Z_BC460_EX5_## to include a field for the destination airport.

- 4-1 Insert a tab and the destination city at the end of the item line. To position the field, define a new tab position. Insert the program symbol for the destination city from table SPFLI.

If there is not enough space available at the end of the item line, restrict the output length using a formatting option.

- 4-2 Additionally, add the new column in the text element for the item header (ITEM_HEADER).

- 4-3 Activate the form and execute print program Z_BC460_EX5_## again with your new form.

Note: It is not sufficient to simply return to the selection screen.

- 5 If the destination city is NEW YORK, you want the following message to be output in bold after the item line:

"Please be aware of possible departure delays at New York airport due to potential American air traffic controller strikes."

- 5-1 Print this text in a new line after the item line. Use the command IF ... ELSE ... ENDIF.

- 5-2 To avoid a page break occurring between the item line and the message, use the command PROTECT ... ENDPROTECT.

- 6 Include the country in the customer address in case the sender's country does not match the customer's country.
- 6-1 Include COUNTRY as a subcommand of the control statement ADDRESS in the ADDRESS window. Specify the program symbol for the customer's country under COUNTRY. You can find the corresponding program symbol in the table SCUSTOM.
 - 6-2 Include FROMCOUNTRY as a subcommand of the control statement ADDRESS. For this exercise you can take 'DE' as the country of origin.

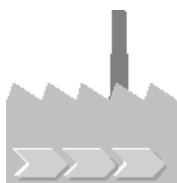
When you carry out a test, you should notice that the address is output differently. If the customer's country is US, USA should appear in the last line of the address. If the customer's country is DE, no country should appear.



Unit: The Print Program



- Calls of function modules for form printing
- Print control via print program
- Enhancements to the print program in conjunction with the related form
- Print preview using the print program



In most cases, standard forms are altered to reflect the needs of individual customers. In some cases, it is then necessary to adjust the print program to meet these needs.

1 The solution program for the first task is called SAPBC460S_05A.

```
*&-----*
*& Report  SAPBC460S_05A                               *
*-----*
& Example print program for training course BC460, chapter 5 *
&-----*

REPORT  sapbc460s_05a                                     .
PARAMETERS: stpage(10) DEFAULT 'FIRST'.
-----*

* Open form printing
CALL FUNCTION 'OPEN_FORM'
  EXPORTING
    form              = 'SAPBC460D_FM_03'
  EXCEPTIONS
    OTHERS             = 1.

IF sy-subrc <> 0.
  WRITE: 'Error in OPEN_FORM'(001).
ENDIF.
```

```

* Choose start page
CALL FUNCTION 'START_FORM'
  EXPORTING
    start page      = stpage
  EXCEPTIONS
    OTHERS          = 1.
IF sy-subrc <> 0.
  WRITE: / 'Error in START_FORM'(002).
ENDIF.

```

```

* Output text element INTRODUCTION

```

```

CALL FUNCTION 'WRITE_FORM'
  EXPORTING
    element          = 'INTRODUCTION '
  EXCEPTIONS
    OTHERS           = 1.

```

```

IF sy-subrc <> 0.
  WRITE: / 'Error in WRITE_FORM, element INTRODUCTION'(003).
ENDIF.

```

```

* Output text element ITEMS

```

```

CALL FUNCTION 'WRITE_FORM'
  EXPORTING
    element          = 'ITEMS'
  EXCEPTIONS
    OTHERS           = 1.

```

```

IF sy-subrc <> 0.
  WRITE: / 'Error in WRITE_FORM, element ITEMS'(004).
ENDIF.

```



```

* Output CLOSING_REMARK text element
CALL FUNCTION 'WRITE_FORM'
  EXPORTING
    element                = 'CLOSING_REMARK'
  EXCEPTIONS
    OTHERS                  = 1.
IF sy-subrc <> 0.
  WRITE: / 'Error in WRITE_FORM, element CLOSING_REMARK'(005).
ENDIF.

```

```

* End form output
CALL FUNCTION 'END_FORM'
  EXCEPTIONS
    unopened                = 1
    bad_pageformat_for_print = 2
    OTHERS                   = 3.
IF sy-subrc <> 0.
  WRITE: / 'Error in END_FORM'(006).
ENDIF.

```

```

* Close form printing
CALL FUNCTION 'CLOSE_FORM'
  EXCEPTIONS
    unopened                = 1
    bad_pageformat_for_print = 2
    OTHERS                   = 3.
IF sy-subrc <> 0.
  WRITE: / 'Error in CLOSE_FORM'(007).
ENDIF.

```

The solution program for the ALTERNATIVE EXERCISE is called
SAPBC460S_PRINT_SOL:

REPORT sapbc460s_print_sol.

TABLES: sbook, scustom.

DATA:

it_sbook TYPE TABLE OF sbook,
it_scustom TYPE TABLE OF scustom,

** Print parameters for form printing*
options TYPE itcpo.

** Selection screen*

SELECT-OPTIONS:

so_kunde FOR scustom-id DEFAULT 1 TO 3 OBLIGATORY,
so_car FOR sbook-carrid DEFAULT 'AA' TO 'LH'.

PARAMETERS:

pa_form TYPE thead-tdform DEFAULT 'SAPBC460_PRINT' OBLIGATORY.

** Preassign print parameters*
options-tddest = 'P280'.

** Data retrieval: Customer information*

SELECT *
FROM scustom
INTO TABLE it_scustom
WHERE id IN so_kunde.

** Was at least one customer found?*
CHECK NOT sy-dbcnt = 0.

```

* Start of form printing
CALL FUNCTION 'OPEN_FORM'
  EXPORTING
    options = options
  EXCEPTIONS
    OTHERS  = 11.

IF sy-subrc <> 0.
  WRITE: / 'Error in function module OPEN_FORM'(e01),
    scustom-id.
ENDIF.

* Evaluate table of selected customers
LOOP AT it_scustom
  INTO scustom.

* Data retrieval: Booking information for current customer
SELECT *
  FROM sbook
  INTO TABLE it_sbook
  WHERE carrid IN so_car AND
    customid = scustom-id.

* The form must be executed again for each customer.
CALL FUNCTION 'START_FORM'
  EXPORTING
    form = pa_form
  EXCEPTIONS
    OTHERS = 7.

IF sy-subrc <> 0.
  WRITE: / 'Error in function module START_FORM'(e02),
    scustom-id.
ENDIF.

```

```

* Output all text elements
CALL FUNCTION 'WRITE_FORM'
  EXPORTING
    element = 'INTRODUCTION '
  EXCEPTIONS
    OTHERS = 9.

* Set header on current page
CALL FUNCTION 'WRITE_FORM'
  EXPORTING
    element = 'ITEM_HEADER'
  EXCEPTIONS
    OTHERS = 9.

* Set header for subsequent pages
CALL FUNCTION 'WRITE_FORM'
  EXPORTING
    element = 'ITEM_HEADER'
    type    = 'TOP'
  EXCEPTIONS
    OTHERS = 9.

* Create list of booking items:
LOOP AT it_sbook
  INTO sbook.

* Call suitable text element for each item
CALL FUNCTION 'WRITE_FORM'
  EXPORTING
    element = 'ITEM_LINE'
  EXCEPTIONS
    OTHERS = 9.

ENDLOOP.

```

** Delete header for subsequent pages*

CALL FUNCTION 'WRITE_FORM'

EXPORTING

element = 'ITEM_HEADER'

function = 'DELETE'

type = 'TOP'

EXCEPTIONS

OTHERS = 9.

CALL FUNCTION 'WRITE_FORM'

EXPORTING

element = 'CLOSING_REMARK'

EXCEPTIONS

OTHERS = 9.

** Close form of current customer*

CALL FUNCTION 'END_FORM'

EXCEPTIONS

OTHERS = 4.

IF sy-subrc <> 0.

WRITE: / 'Error in function module END_FORM'(e03),
scustom-id.

ENDIF.

ENDLOOP.

** End of form printing*

CALL FUNCTION 'CLOSE_FORM'

EXCEPTIONS

OTHERS = 5.

IF sy-subrc <> 0.

WRITE: / 'Error in function module CLOSE_FORM'(e04),
scustom-id.

ENDIF.

2 The solution program for the optional task is called SAPBC460S_05B. The added source code is marked bold.

```
*&-----*
*& Report   SAPBC460S_05B                               *
*&-----*
*& Example print program for training course BC460, chapter 5
*&-----*
REPORT sapbc460s_05b .
```

TABLES: scustom, sbook, spfli.

SELECT-OPTIONS: s_id FOR scustom-id DEFAULT 200 TO 200,
 s_fli FOR sbook-carrid DEFAULT 'LH' TO 'LH'.
PARAMETERS: form LIKE thead-tdform DEFAULT 'SAPBC460D_FM_05'.

DATA customers LIKE scustom OCCURS 100
 WITH HEADER LINE

DATA bookings LIKE sbook OCCURS 1000
 WITH HEADER LINE

DATA connections LIKE spfli OCCURS 1000
 WITH HEADER LINE

DATA: BEGIN OF sums OCCURS 10,
 forcuram LIKE sbook-forcuram,
 forcurkey LIKE sbook-forcurkey,
 END OF sums.

DATA BEGIN OF options.
 INCLUDE STRUCTURE itcpo.

DATA END OF options.

DATA BEGIN OF result.
 INCLUDE STRUCTURE itcpp.

DATA END OF result.

* Get data

SELECT * FROM scustom INTO TABLE customers
 WHERE id IN s_id
 ORDER BY PRIMARY KEY.

SELECT * FROM sbook INTO TABLE bookings
 WHERE customid IN s_id AND carrid IN s_fli

```

        AND    forcurkey NE space
        ORDER BY PRIMARY KEY.
SELECT * FROM  spfli    INTO TABLE connections
        FOR ALL ENTRIES IN bookings
        WHERE carrid = bookings-carrid
        AND    connid = bookings-connid
        ORDER BY PRIMARY KEY.

```

```

* Open print job
options-tddest = '*'.
options-tdimmed = '*'.
options-tddelete = '*'.
options-tdnewid = 'X'.

```

CALL FUNCTION 'OPEN_FORM'

EXPORTING

```

        device           = 'PRINTER'
        dialog           = 'X'
        options           = options

```

EXCEPTIONS

```

        canceled         = 1
        device           = 2
        form             = 3
        options           = 4
        unclosed         = 5
        OTHERS           = 6.

```

```

IF sy-subrc <> 0.
    WRITE 'Error in OPEN_FORM'(001).
    EXIT.
ENDIF.

```

```

* Print form for all customers
LOOP AT customers.
* Set customer address
    scustom = customers.
* Open form of respective customer
    CALL FUNCTION 'START_FORM'

```

EXPORTING

form = form

EXCEPTIONS

OTHERS = 1.

IF sy-subrc <> 0.

WRITE 'Error in START_FORM'(002).

EXIT.

ENDIF.

* Output introduction text

CALL FUNCTION 'WRITE_FORM'

EXPORTING

element = 'INTRODUCTION'

EXCEPTIONS

OTHERS = 1.

IF sy-subrc <> 0.

WRITE 'Error in WRITE_FORM, element INTRODUCTION'(003).

EXIT.

ENDIF.

* Output column headings of main window

CALL FUNCTION 'WRITE_FORM'

EXPORTING

element = 'ITEM_HEADER'

EXCEPTIONS

OTHERS = 1.

IF sy-subrc <> 0.

WRITE 'Error in WRITE_FORM, element ITEM_HEADER'(004).

EXIT.

ENDIF.

* Set column headings into TOP area of main window for

* subsequent pages

CALL FUNCTION 'WRITE_FORM'

EXPORTING

element = 'ITEM_HEADER'

function = 'SET'

type = 'TOP'

EXCEPTIONS

OTHERS = 1.

IF sy-subrc <> 0.

WRITE 'Error in WRITE_FORM, top element
ITEM_HEADER'(005).

EXIT.

ENDIF.

* Customer bookings

CLEAR sums. REFRESH sums.

LOOP AT bookings WHERE customid = customers-id.

sbook = bookings.

* Get departure time

READ TABLE connections WITH KEY carrid = bookings-carrid
connid = bookings-connid.

IF sy-subrc = 0.

spfli = connections.

ELSE.

CLEAR spfli.

ENDIF.

* Print item

CALL FUNCTION 'WRITE_FORM'

EXPORTING

element = 'ITEM_LINE'

EXCEPTIONS

OTHERS = 1.

IF sy-subrc <> 0.

WRITE 'Error in WRITE_FORM, element ITEM_LINE'(006).

EXIT.

ENDIF.

```

* Add current position to corresponding entry in table sums
  MOVE-CORRESPONDING sbook TO sums.
  COLLECT sums.
ENDLOOP.          " at bookings

```

```

* Delete column headings from TOP area of main window

```

```

  CALL FUNCTION 'WRITE_FORM'

```

```

    EXPORTING

```

```

      element      = 'ITEM_HEADER'

```

```

      function     = 'DELETE'

```

```

      type        = 'TOP'

```

```

    EXCEPTIONS

```

```

      OTHERS       = 1.

```

```

  IF sy-subrc <> 0.

```

```

    WRITE 'Error in WRITE_FORM, delete element
          ITEM_HEADER'(007).

```

```

    EXIT.

```

```

  ENDIF.

```

```

* Print final remark

```

```

  CALL FUNCTION 'WRITE_FORM'

```

```

    EXPORTING

```

```

      element      = 'CLOSING_REMARK'

```

```

    EXCEPTIONS

```

```

      OTHERS       = 1.

```

```

  IF sy-subrc <> 0.

```

```

    WRITE 'Error in WRITE_FORM, element CLOSING_REMARK'(008).

```

```

    EXIT.

```

```

  ENDIF.

```

```

* Print sum

```

```

  LOOP AT sums.

```

```

    MOVE-CORRESPONDING sums TO sbook.

```

```

    CALL FUNCTION 'WRITE_FORM'

```

```

      EXPORTING

```

```

        element    = 'SUM'

```

```

      EXCEPTIONS

```

```

        OTHERS     = 1.

```

```
IF sy-subrc <> 0.
    WRITE 'Error in WRITE_FORM, element SUM'(009).
    EXIT.
ENDIF.
ENDLOOP.          " at sums

* Close customer form
CALL FUNCTION 'END_FORM'
    EXCEPTIONS
        OTHERS    = 1.
IF sy-subrc <> 0.
    WRITE 'Error in END_FORM'(010).
    EXIT.
ENDIF.
ENDLOOP.          " at customers
* close print job
CALL FUNCTION 'CLOSE_FORM'
    IMPORTING
        result    = result
    EXCEPTIONS
        OTHERS    = 1.
IF sy-subrc <> 0.
    WRITE 'Error in CLOSE_FORM'(011).
    EXIT.
ENDIF.
```

The solution form is called SAPBC460S_FM_05.

- 1 Copy paragraph IL to IM and insert a sixth tab position: 1 5 CM, LEFT.
- 2 The text element ITEM_LINE is defined as follows:

```
/E  ITEM_LINE
IM  &SBOOK-CARRID&,,&SBOOK-CONNID&,,&SBOOK-FLDATE&,,
=   &SPFLI-DEPTIME&,,&SBOOK-FORCURAM&,,&SBOOK-FORCURKEY&,,
=   &SPFLI-CITYTO(8)&
```

- 1 Copy paragraph IH to IJ and insert a fourth tab position: 15CM, LEFT.
- 2 The text element ITEM_HEADER is defined as follows:

```
/E  ITEM_HEADER
IJ  Flight,,Date,,Departure,,Price,,Destination
```

- 1 The text element ITEM_LINE is defined as follows:

```
/: IF &SPFLI-CITYTO& = 'NEW YORK'
/: PROTECT
IM  &SBOOK-CARRID&,,&SBOOK-CONNID&,,&SBOOK-FLDATE&,,&SPFLI-
DEPTIME& ,,
=   &SBOOK-FORCURAM&,,&SBOOK-FORCURKEY&,,&SPFLI-CITYTO(8)&
/   <B>Please be aware of possible departure delays at New York airport due to potential
American air traffic controller strikes.</>
/: ENDPROTECT
/: ELSE
IM  &SBOOK-CARRID&,,&SBOOK-CONNID&,,&SBOOK-FLDATE&,,&SPFLI-
DEPTIME&    =,,&SBOOK-FORCURAM&,,&SBOOK-FORCURKEY&,,&SPFLI-
CITYTO(8)&
/: ENDIF
```

Activate the form and execute Z_BC460_EX_## with your new form.

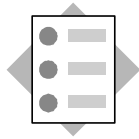
- 2 The ADDRESS window is defined as follows:

```
/: ADDRESS PARAGRAPH AS
/:  TITLE    &SCUSTOM-FORM&
/:  NAME     &SCUSTOM-NAME&
/:  STREET   &SCUSTOM-STREET&
/:  POSTCODE &SCUSTOM-POSTCODE&
/:  CITY     &SCUSTOM-CITY&
/:  COUNTRY  &SCUSTOM-COUNTRY&
/:  FROMCOUNTRY 'DE'
/: ENDADDRESS
```

Contents:

- Defining and using styles

© SAP AG 1999



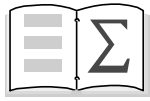
At the conclusion of this unit, you will be able to:

- **Create and maintain styles**
- **Use styles**

The screenshot shows the SAP Form Designer interface. A text box contains the text "You have assigned a STYLE to this text." with a red arrow pointing to it from a yellow callout box. Below this, another text box contains "This is an outline:" with a red arrow pointing to it from the same yellow callout box. A third text box contains "First level" with a red arrow pointing to it from the same yellow callout box. A fourth text box contains "Second level" with a red arrow pointing to it from the same yellow callout box. A fifth text box contains "Third level" with a red arrow pointing to it from the same yellow callout box. A red arrow points from the text "F4" to the "Format Column(1) 6 A..." dialog box. The dialog box shows a list of styles: G3 Outline level 3, AS Standard paragraph, B Bold, CU Italic, underlined, G1 Outline level 1, G2 Outline level 2, and G3 Outline level 3. The yellow callout box contains the text: "You have assigned a style to this text.", "This is an outline:", "=> First level", "=> Second level", and "=> Third level".

© SAP AG 2006

- You define paragraph and character formats in styles which are then available independently of the existing forms. Besides the definitions in the form, this is an additional method to format text.
- The SAP system comes with a number of predefined styles, such as those for electronic mail messages or online documentation.
- You can assign a style to any text. To do so, choose *Format -> Change Style...* in the editor. To display all formats defined in a style, choose *Format -> Character* and *Format -> Paragraph*.
- Using the control statement `/: STYLE` you can quickly change the style within a text.
- When you assign a style, the paragraph and character format definitions used in this style override those of the current form. The format definitions in the form have no effect then.
- You use style maintenance to create new styles, change styles, find existing styles, and display all available styles.
- To access the style maintenance transaction, choose *Tools -> Form Printout -> SAPscript -> Style*. To access it from within form maintenance or standard text maintenance, choose *Environment -> Style*.



You can now:

- **Create and maintain styles**
- **Use styles**

© SAP AG 1999

Exercise



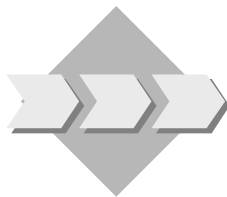
Unit: Styles

Topic: Create and Assign Styles



At the end of these exercises you will be able to:

- Create and maintain SAPscript styles
- Assign SAPscript styles to texts



You would like to use standard paragraph and character formats independently of forms.

- 1-1 Create a style called Z_BC_## that contains different paragraph and character formats.
stands for your two-digit group/monitor number.
- 1-2 Enhance your form Z_BC460_EX5_## (from the exercise on symbols and control statements) as follows:
Set up the style in a text element of your choice using the appropriate SAPscript statement. Format parts of this text element with formats from the style.
- 1-3 Assign (in transaction SO10) your new style to the standard text that you have included in your form. Format parts of this standard text by making a selection from the format list (that is dependent on the chosen style).



Unit: Styles

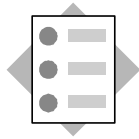
Topic: Create and Assign Styles

- 1-1 Enter transaction SE72 and create a style. You arrive at the header data first. Enter a description. You cannot yet define the default paragraph (that is always selected in a text when you format a paragraph with *).
Choose the *Paragraph Formats* pushbutton and create at least one paragraph format. Proceed exactly as you would to create paragraph formats within a form. Choose the *Character Formats* pushbutton and create at least one character format. Proceed exactly as you would to create character formats within a form. Now return to the header data. Enter one of the paragraph formats you have just created as the default paragraph. (You can use the F4 help to do this.) Now activate the style. You are prompted to enter a package. As before, choose ZBC460_## or *Local Object*. If you opt for the package, you have to enter the transport request number as usual (on the next screen). Confirm that a message confirming that the style has been activated successfully is displayed in the status bar.
- 1-2 Go to the maintenance of your form (SE71) and then to the editor of a window. The SAPscript statement that you use to set up a style is:
/: STYLE Z_BC_##
The text that follows this statement will be formatted according to the style that has been set up. Note that the format selection (for example, using the dropdown list) still relates to the form's formats, not the style. As a consequence, you have to enter your required formats manually:
<Format>Text</>.
If you want the format of text that appears later the same text element to return to the form defaults, you can deactivate the style as follows:
/: STYLE *
- 1-3 You assign a style to a standard text (transaction SO01) as follows:
Menu *Format* → *Change Style*. Choose your style from the selection list by double-clicking on it. You will notice that the character and paragraph format selection lists now only contain the style's formats.
To check which style is assigned to the text, you can take a look at the status bar of the line editor; it is shown on the left.

Contents:

- Objectives and advantages of SAP Smart Forms
- SAP Interactive Forms with Adobe

© SAP AG 2006

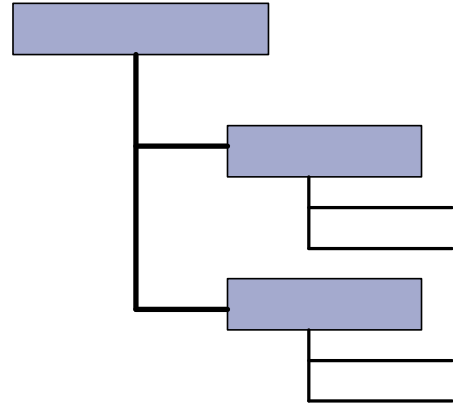


After completing this unit, you will be able to:

- **List the objectives and advantages of SAP Smart Forms**
- **Describe SAP Interactive Forms with Adobe**

© SAP AG 2006

- **Clear representation of the form logic**
- **Reduced maintenance effort (by approximately 50%)**
- **Automatic generation of ABAP code**

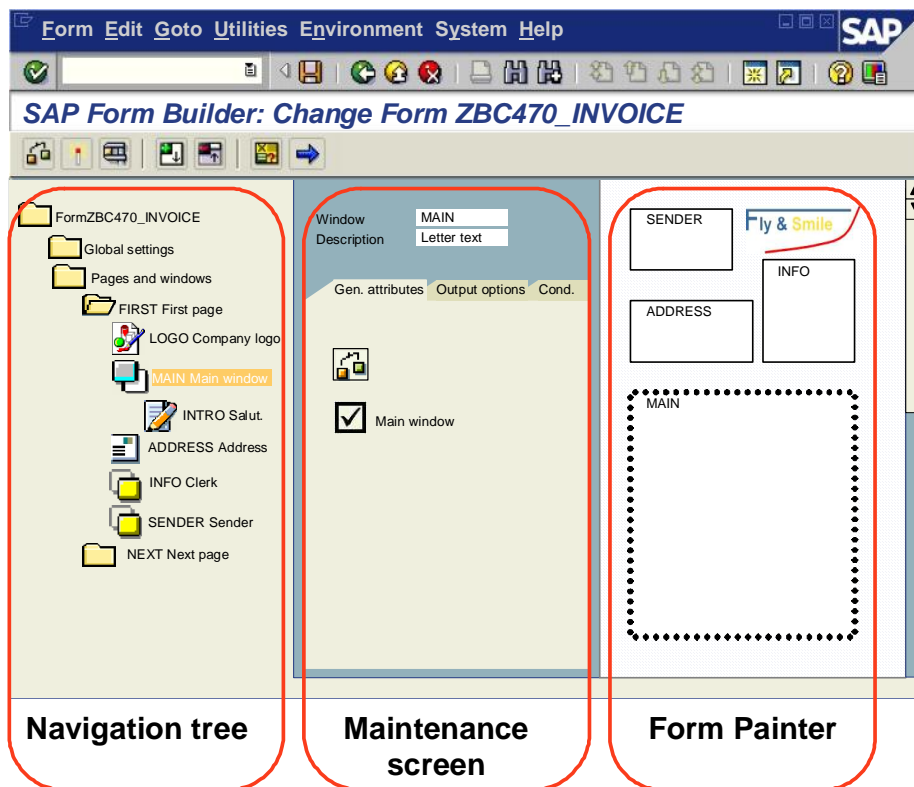


© SAP AG 2006

- SAP Smart Forms are maintained in the SAP Form Builder. The Form Builder gives you an overview of all components used in the form (see next slide).
- When you activate your SAP Smart Form, the system automatically generates the associated ABAP code in the form of a function module. You integrate this module with your data retrieval program. This eliminates the maintenance effort in the print program.

Areas of the SAP Form Builder

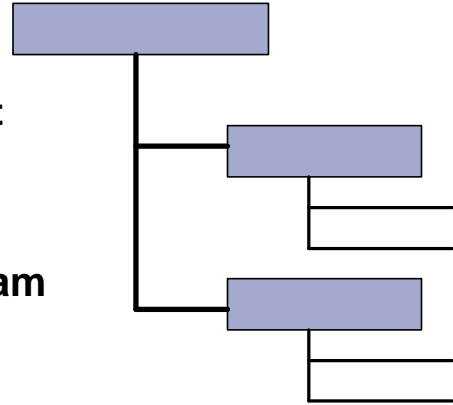
SAP



© SAP AG 2002

- To edit forms using SAP Smart Forms, use the graphical SAP Form Builder.
- The SAP Form Builder is divided into three areas:
 - On the left-hand side: The navigation tree. This tree graphically displays the hierarchy of the SAP Smart Form. The individual form elements (such as pages or graphics) are represented by nodes. You can additionally display the field list with variables below the navigation tree.
 - In the middle: The maintenance screen. This screen has several tabs on which you set and change the attributes of the node currently selected. You can also enter text with the editor or use the Table Painter to determine the layout of a table.
 - On the right-hand side: The Form Painter. The Form Painter is used to define the layout of a page, such as the position and size of text windows and graphics. You can hide the Form Painter if you wish: *Utilities* → *Form Painter on/off*.
- You can select nodes to edit by double-clicking them in the navigation tree or the Form Painter.

- **No control commands required**
- **Table processing using SAP Smart Forms**
- **Clearly defined interface between the form and the application program**



© SAP AG 2002

- The technology of the SAPscript control commands is replaced with a new concept in SAP Smart Forms. This new concept is more intuitive and has become necessary because the function module is generated automatically.
- In SAPscript, displaying items in complex tables requires a large number of tasks to be performed in the form and print program. In SAP Smart Forms, this is automated using nodes. For example, you can use SAP Smart Forms to display the items with a dynamic table grid.
- SAP Smart Forms also support the output of documents in XML format.

Table

Your bookings:

Flight	Date	Price
AA017	12/16/2000	1,200.00 USD
AA017	12/31/2000	1,200.00 USD
Sum for AA		2,400.00 USD
LH400	11/17/2000	581.00 DEM
LH402	11/17/2000	669.00 DEM
LH403	12/12/2000	610.00 DEM
Sum for LH		1,860.00 DEM
Total		2,400.00 USD
		1,860.00 DEM

Template

Name of passenger (not transferrable) YILMAZ/E MS					Issued 6NOV00	
To	Carr.	Flight	Cl.	Date	Time	
FRANKFURT	LH	2362	L	27NOV	1840	
BERLIN TXL	LH	2351	L	28NOV	1910	
Flight price DEM 350.00			Form and serial number 3344563125667 Please do not write on or stamp this field.			
Tax DEM 52.59						
Total DEM 402.59						

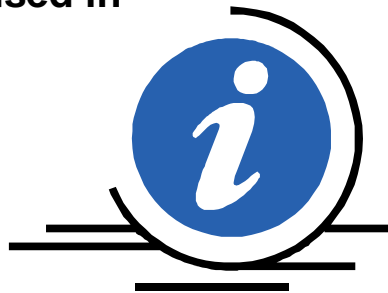
- Layout
 - Size
- } Only at runtime

- Layout fixed
- Size fixed

© SAP AG 2002

- Tables and templates have several things in common. For example, they are both designed with the Table Painter tool, and they use different line types.
- The most important difference between them is how their layout is determined:
 - The precise layout and the length of tables can only be determined at runtime, depending on the type and the number of records read by the application program from the database.
 - Templates, however, are completely defined in the Form Painter. This means that the type and the number of their cells cannot be modified at application program runtime. You, therefore, use templates primarily for external forms.

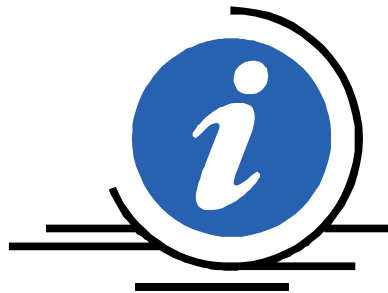
- **SAPscript forms are still supported**
- **SAPscript forms can still be used**
- **SAPscript text modules can be used in SAP Smart Forms**



© SAP AG 2006

- You can use SAPscript forms simultaneously with SAP Smart Forms. SAP still supports SAPscript forms.
- However, SAP does not plan to further develop SAPscript. In the long term, you should convert your SAPscript forms into SAP Smart Forms.

- **SAP Smart Forms are available as of Release 4.6C**
- **SAPscript forms and SAPscript styles can be migrated into SAP Smart Forms or Smart Styles (using migration tool, some adjustments may be required)**

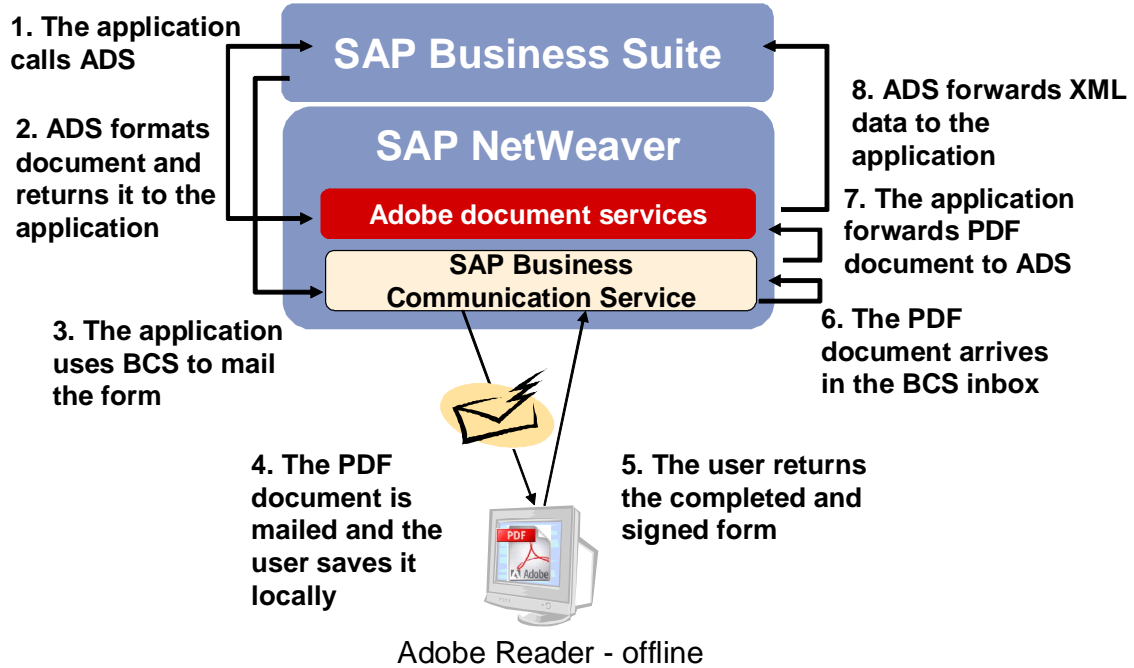


© SAP AG 2006

- You can find the function to convert SAPscript forms into SAP Smart Forms in the *Utilities* menu of the SAP Smart Forms maintenance transaction as of Release 4.6C.
- Since, for example, the display of tables was primarily achieved in SAPscript using the print program, you have to adjust such functions to the new SAP Smart Forms technology after migrating the form.

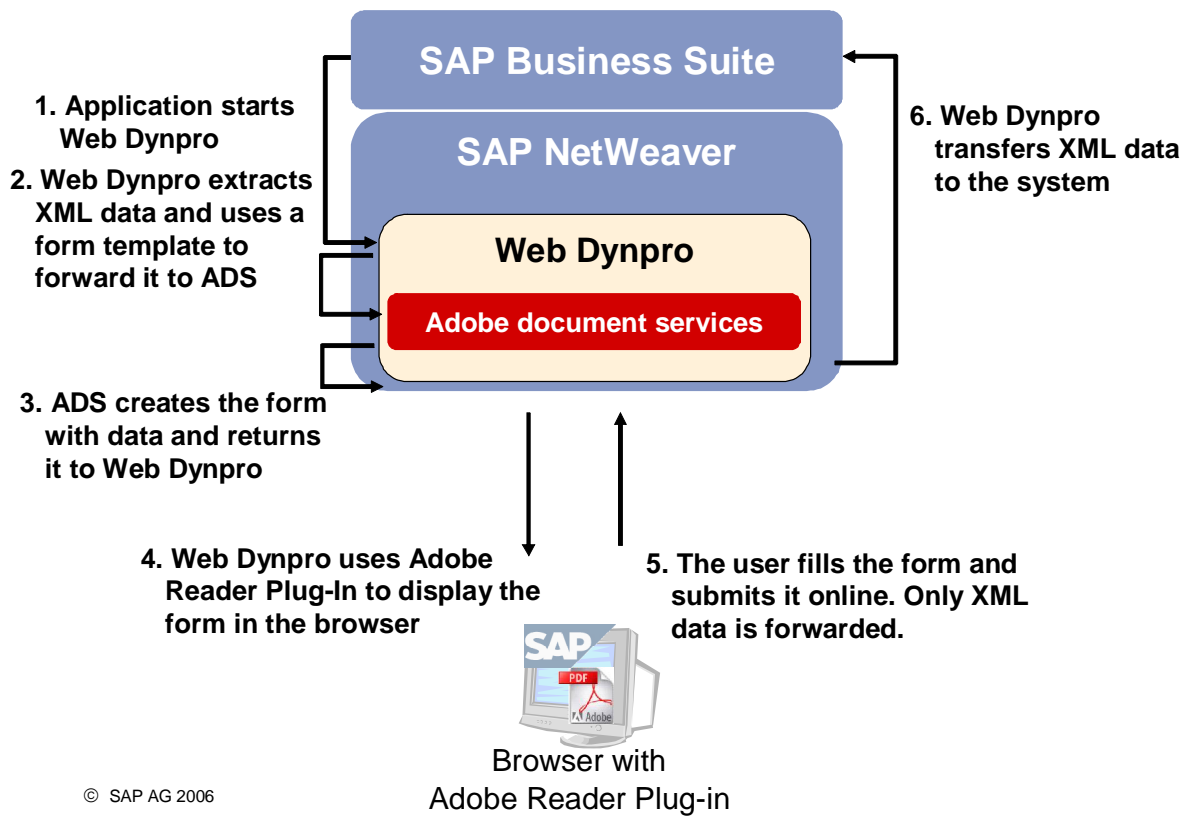
- Overview

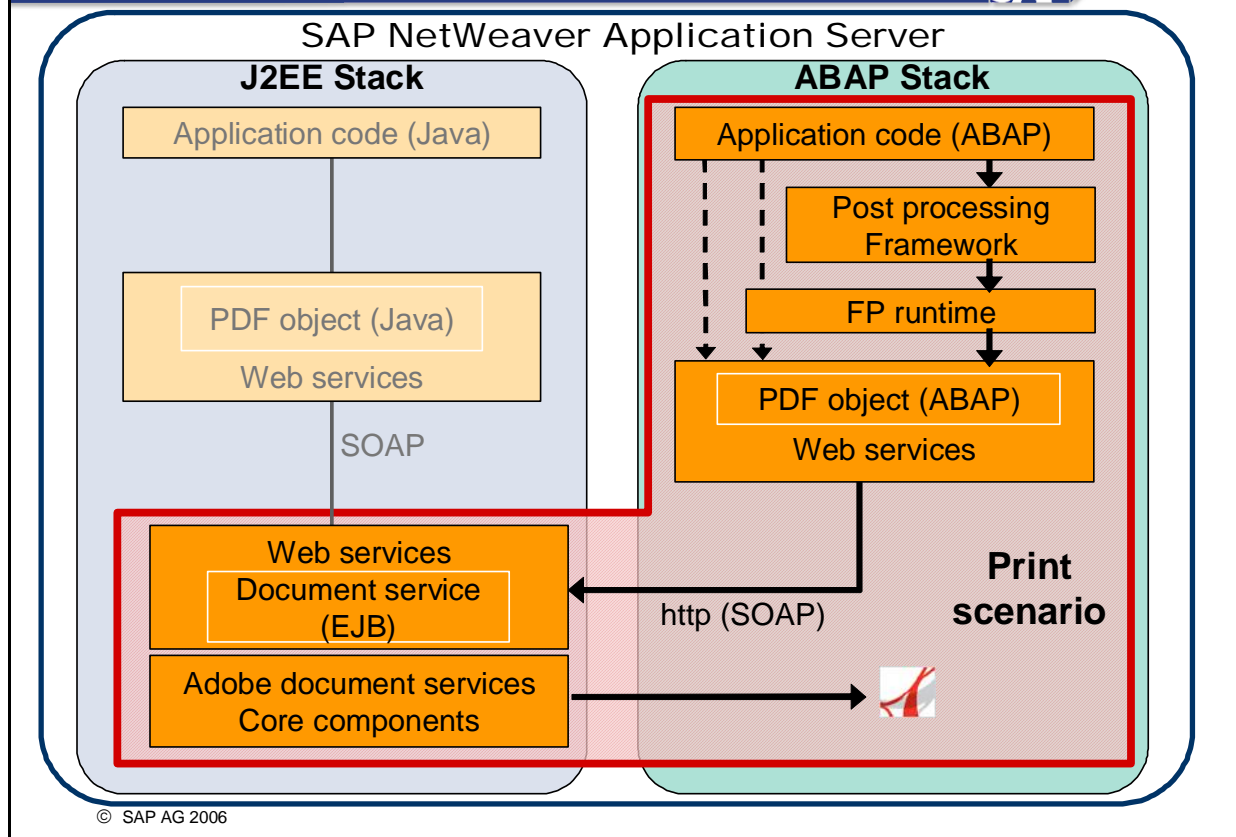
© SAP AG 2006



© SAP AG 2006

- Since SAP NetWeaver 2004, SAP Interactive Forms by Adobe provides another form tool that is used in some applications instead of or in addition to SAPscript/SAP Smart Forms.
- In broad terms, there are three different scenarios:
 - In the **offline scenario**, a PDF document is created with reader rights that allow user entries. This completed form can subsequently be returned to the system at any time and the system will extract the data.
 - In the **online scenario**, the form that is ready for input is integrated into a Web application (for example, Web Dynpro). The user enters data and then sends it, which means that the data is transferred to the system.
 - The **print scenario** corresponds to the print scenario of SAP Smart Forms as regards its idea, but not its architecture.

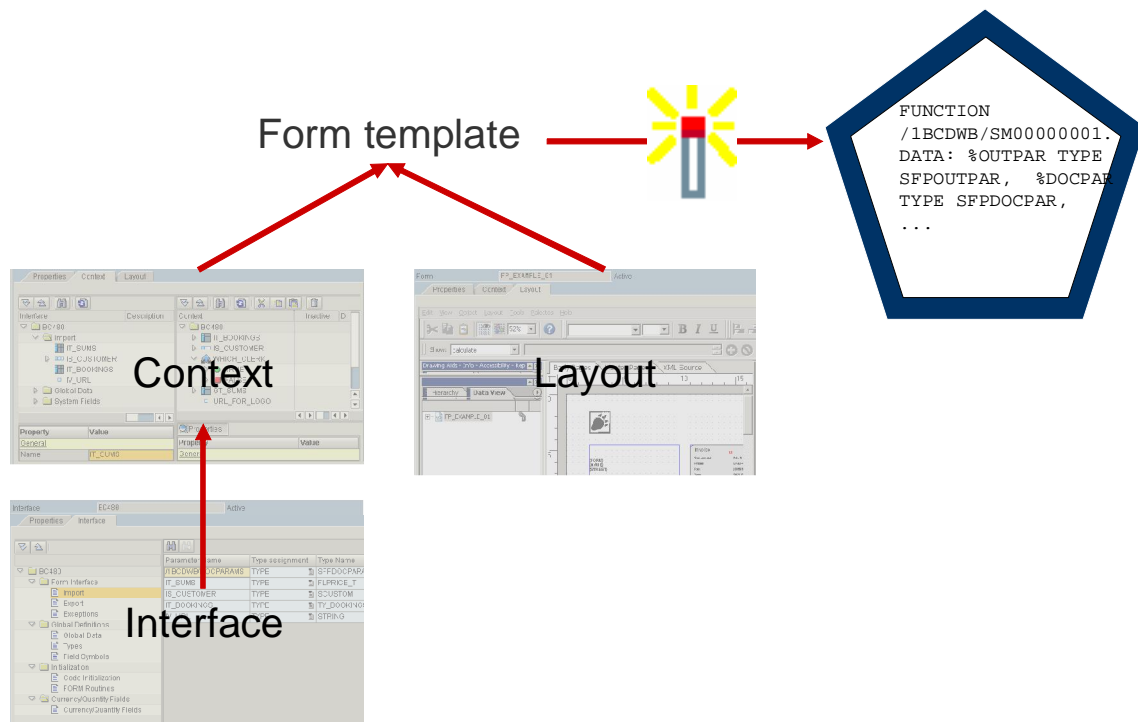




- A short overview of the runtime architecture of a print scenario
- At runtime, an ABAP application program uses the Post Processing Framework (PPF) to determine whether an output is required and, if so, which output. The Post Processing Framework provides SAP applications with a uniform interface for condition-dependent generation of actions (for example, printing delivery notes, faxing order confirmations, or triggering approval procedures). The actions are generated if specific conditions occur for an application document. They are then either processed immediately or at a later time. PPF is the successor to Output Control.
- The form processing runtime provides functions such as opening and closing a spool job for PDF-based forms. It will then call a Web service, which will communicate with Adobe document services via the XML standard Simple Object Access Protocol (SOAP) using the http protocol. Adobe document services are responsible for form rendering, including filling in fields, page breaks, or applying layout settings.
- Adobe document services must be deployed on the J2EE engine. They encompass the Web Service Adobe document services as well as other services that can be installed with the SAP NetWeaver Application Server.

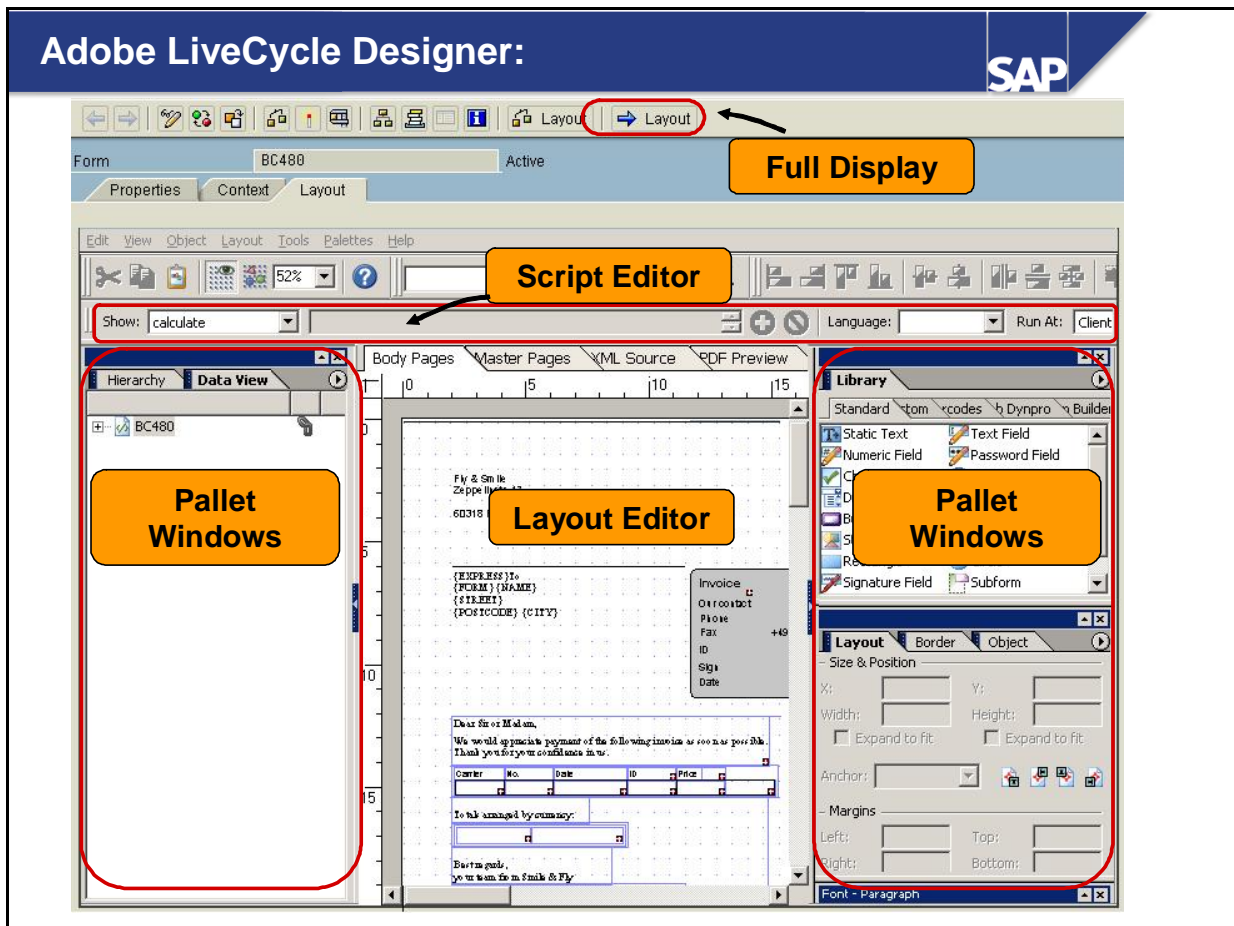
Print Scenario: Tools Involved (Design Time)

SAP

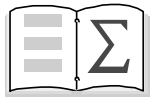


© SAP AG 2006

- To create a print form, you must carry out the following steps:
- **Interface** (transaction SE80 or SFP): The interface is used to define which data a program can pass on to a form. It also contains global data and initialization coding that can be used in a form.
- **Form design** (transaction SE80 or SFP): A print form design normally consists of the context and the layout.
- In the context, you define which parts of the interface you actually want to use in that particular form. You can also add elements like text modules or images.
- The layout usually consists of static and dynamic elements. The layout is defined in Adobe LiveCycle Designer. It is displayed in a special XML format.
- Before a form template can be used, it must be activated. When a form template is activated, the system generates a function module. The generated function module encapsulates all properties of a form and is called whenever a program triggers form processing.



- Form processing takes place in transaction SFP or SE80. Adobe LiveCycle Designer is integrated there to determine the form design.
- The Designer workspace consists of four main areas.
- In the top area, the Script Editor can be displayed. It allows you to enter scripts for calculations. You can choose between JavaScript and Adobe's FormCalc.
- The subdivisions of the left and right areas are called pallet windows with further subdivisions of pallets. It is up to you to decide which pallet windows you want to display in which size.



You are now able to:

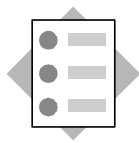
- **List the objectives and advantages of SAP Smart Forms**
- **Describe SAP Interactive Forms with Adobe**

© SAP AG 2006

Contents:

- Assigning print programs to forms
- Change procedure
- Transporting SAPscript objects

© SAP AG 1999



At the conclusion of this unit, you will be able to:

- **Assign print programs to forms during Customizing**
- **Explain what changes made to the SAP Standard mean to the end user**
- **List considerations when you correct and transport SAPscript objects**

Create purchase order

Transaction
SAPMV45A
VA01

Order confirmation

RVADOR01
Print program

RVORDER01
Form

Printer

Assignment stored
in Customizing

Art.	Medium	Program	Form
BA00	1	RVADOR01	RVORDER01
BA00	2	RVADOR01	RVORDER01

© SAP AG 2002

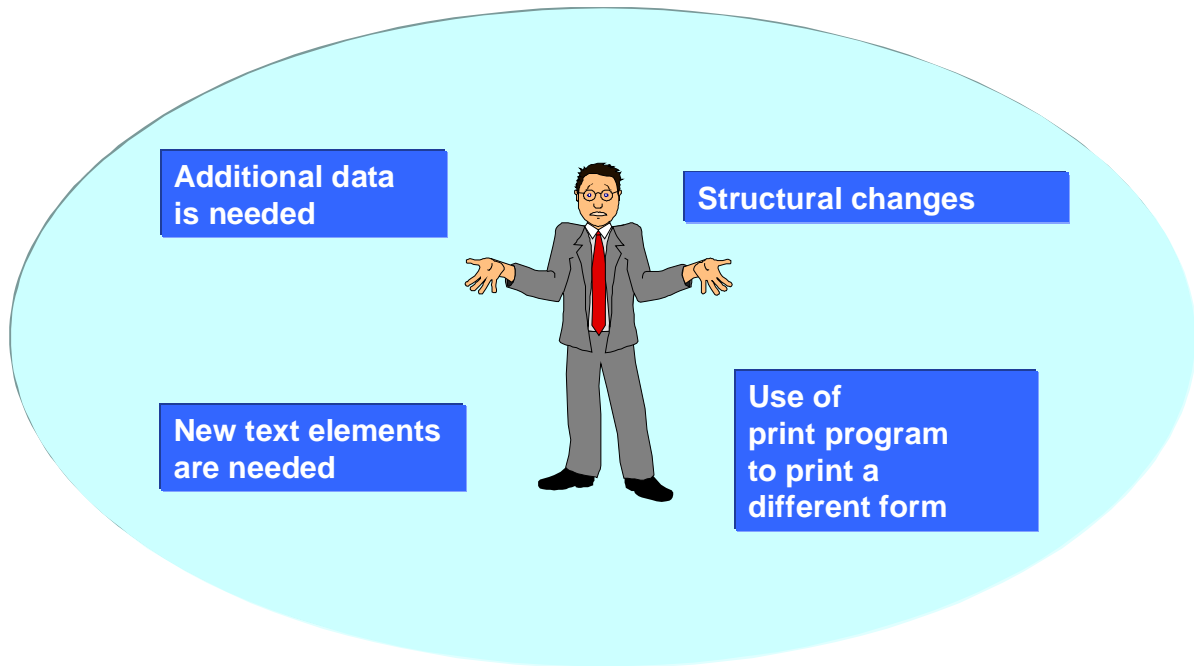
- Forms can be printed automatically from a customer request, such as a purchase order.
- The above example is a simplified representation of just such a process.
- The customer submits an order. The system creates a purchase order. Order confirmation is generated when the purchase order is saved and this is sent to the customer to confirm that the order is being processed.
- The name of the form that is created, together with its print program, is stored in the corresponding application component during Customizing.
- If you want to use a different form or print program for printing, you should store it in your application component during Customizing.

- **SAP delivers standard forms in all applications**
- **SAP also delivers corresponding print programs for these forms**
- **Sometimes forms must be altered to satisfy customer requirements**
- **Often forms can be changed to meet customer needs without having to alter the corresponding print program**



© SAP AG 2006

- SAP delivers standard forms for all applications as well as the corresponding print programs for these forms.
- SAP standard forms can be altered to meet individual customer needs.
- In most cases, forms can be adapted to meet customer needs without having to change the corresponding print program.



© SAP AG 2006

- Whenever SAP objects are altered, this is called a modification.
- Modified objects are not protected from changes made during upgrades. For this reason, use a copy of the print program delivered by SAP when making changes to it.
- Only make changes to an SAP print program after having seriously considered whether the modifications are absolutely necessary.
- Possible reasons to change an SAP print program:

Requirement	Changes to SAP Print Program
Structural changes	Change necessary
New text elements are needed	Change necessary
Print program to be used to print additional forms	Change may be performed if desired
Additional data needed	Change may be performed if desired



Which new fields and tables are needed

Are the required fields in the existing tables or are new fields and tables required?

Which program and which form are affected?

© SAP AG 2006

- You might require more data than is provided in the standard shipment.
- Before making any changes, try to get a clear overview of exactly which additional fields you need. Are these fields already available in the tables already being used or do you need additional tables? Have the table fields that you need already been supplied with data?
- Open up your application component in Customizing and note the forms and programs you are currently using to fulfill your requirements.



Does the print program have to be changed or only the form?

Can the data you need be delivered from another program using PERFORM?

Do SAP enhancements exist in the form of customer exits?

© SAP AG 2006

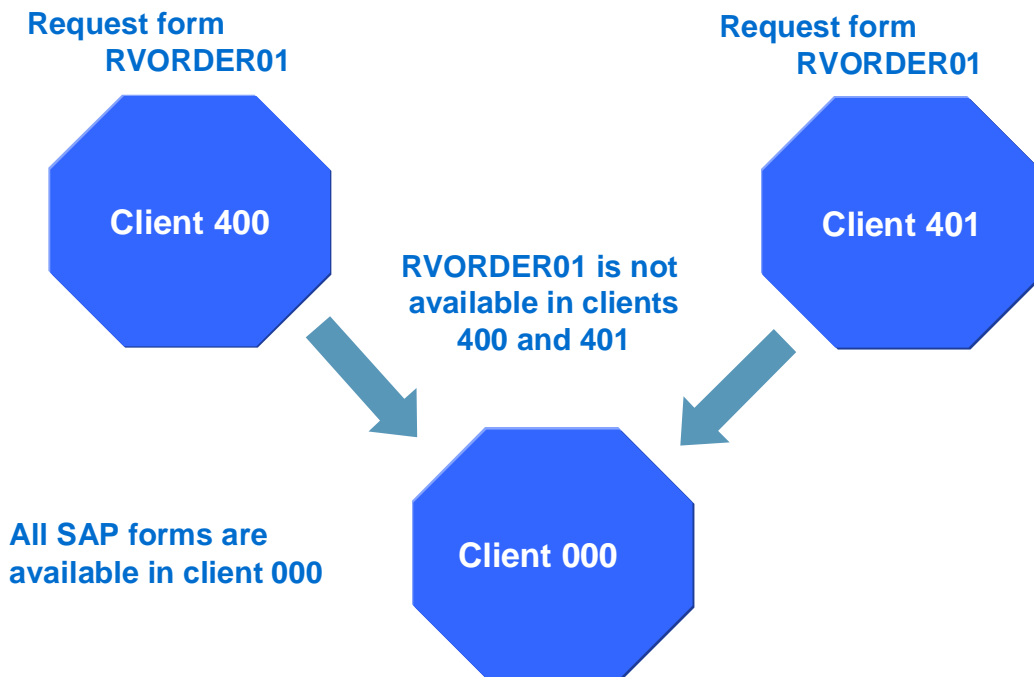
- Consider carefully whether it is necessary to change the print program, or if it would be sufficient to simply alter your form.
- Be aware that additional data can be retrieved using the PERFORM command in SAPscript.
- Execute the PERFORM command in your form and write the subroutine called in a separate program in the customer namespace. The subroutine then retrieves the additional data you want.
- In some cases, includes already exist that have been created by SAP application developers and that you can use to include your enhancements. These customer exits are not affected by any upgrades that are imported. Subsequent adjustments are, therefore, not necessary.



- **Copy SAP standard form and, if required, print program to customer namespace Z or Y**
- **Make alterations to copies**
- **Store new forms and print programs in Customizing**
- **Test alterations**

© SAP AG 2006

- Avoid making modifications if at all possible. Use the SAP standard shipment as a model for enhancements.
- When making changes to an SAP standard form, make a copy of it in one of the customer namespaces (Z or Y) and then make your changes to the copy.
- Whenever a form's structure has been altered, its print program requires changes as well.
- To do this, copy the print program to one of the customer namespaces (Z or Y) and then make your changes to the copy.
- Store your newly created forms and print programs in Customizing.
- Test your changes.



© SAP AG 2006

- Forms are client-dependent. This means that, except for the SAP standard forms stored in client 000, forms are only available in the client in which they were created.
- Forms in client 000 have a special status. Whenever a form is not available in the current client, the system checks to see if the form exists in client 000. If the form exists in client 000, it is then made available in the current client.
- SAP standard forms are, as a matter of course, all stored in client 000.
- You can benefit from client access by storing all your forms in client 000. That way, if a local version is not available in the client you are working in, the version stored in client 000 is accessed automatically.

Program Edit Goto System Help

Copy Forms Between Clients

Form Name: RVORDER01

Source Client: 000

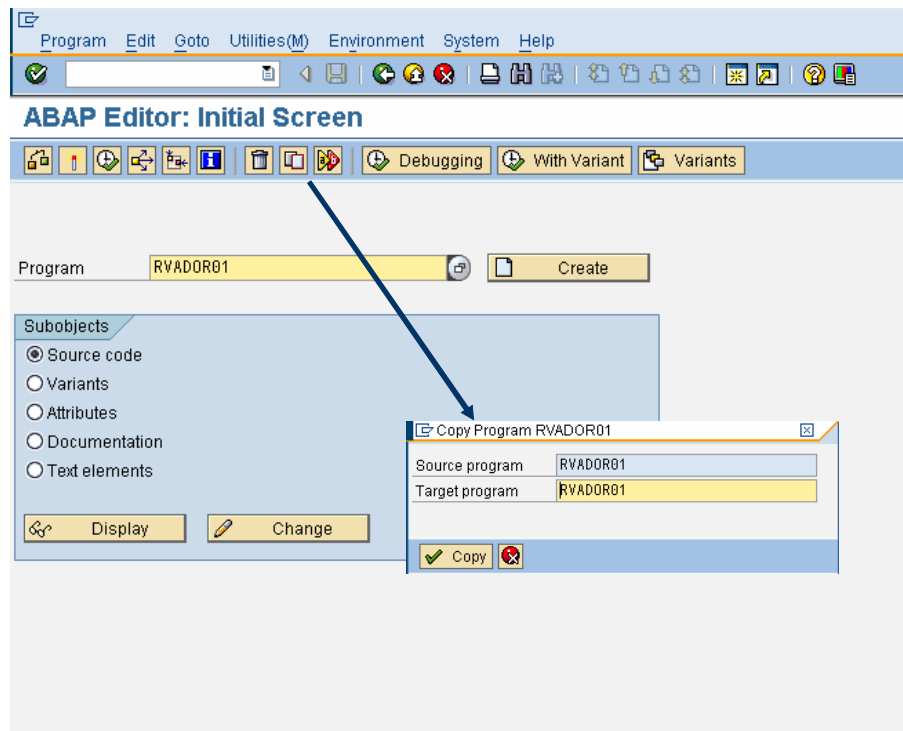
Target Form: ZRVORDER01

☐ Original Language Only

☒ Flow Trace

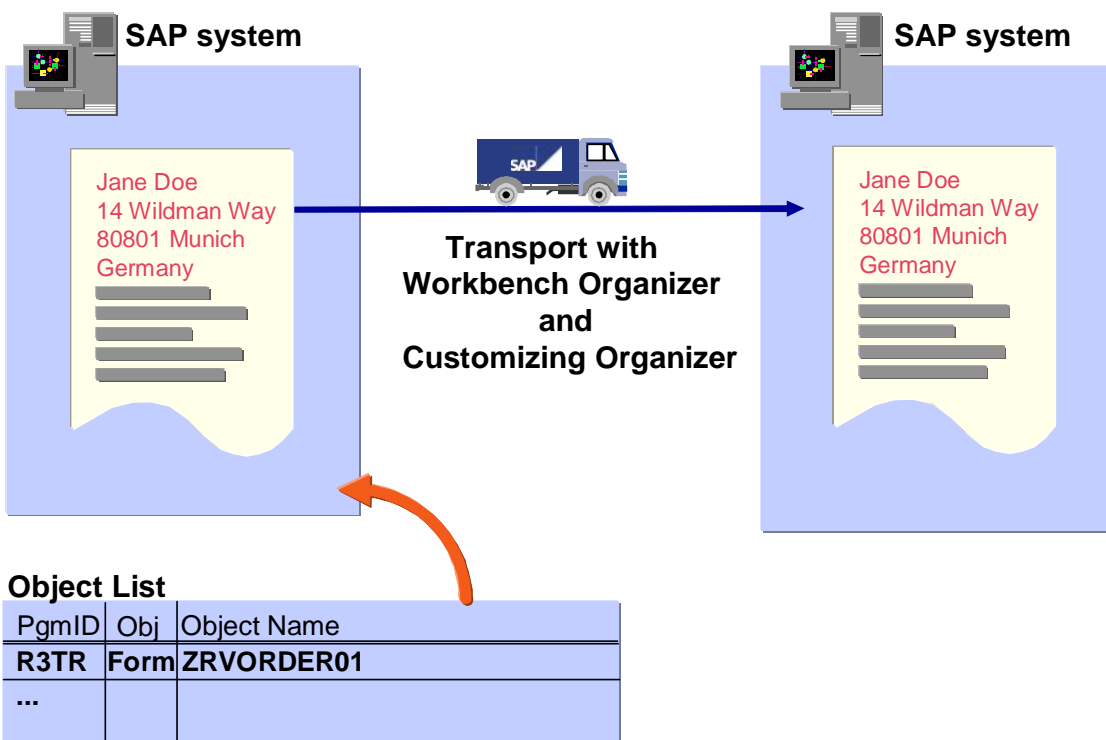
© SAP AG 2006

- SAP standard forms are client-dependent. In order to copy an SAP standard form, it must be available in the current client.
- Call the form maintenance transaction (SE71) and choose *Utilities -> Copy from Client*.
- Enter the name of the form to be copied and choose a name for the target form in the current client. Client 000 is entered as the default source client.



© SAP AG 2006

- To copy print programs, call the ABAP Editor from the ABAP Workbench.
- Enter the name of the program you want to copy and choose the *Copy ... icon*.
- In the dialog box, enter the name of the copied program in customer namespace Z or Y and choose the *Copy* pushbutton.
- Now you can edit the program you have copied. After you have made the changes you want, insert the new print program at the appropriate point in Customizing.



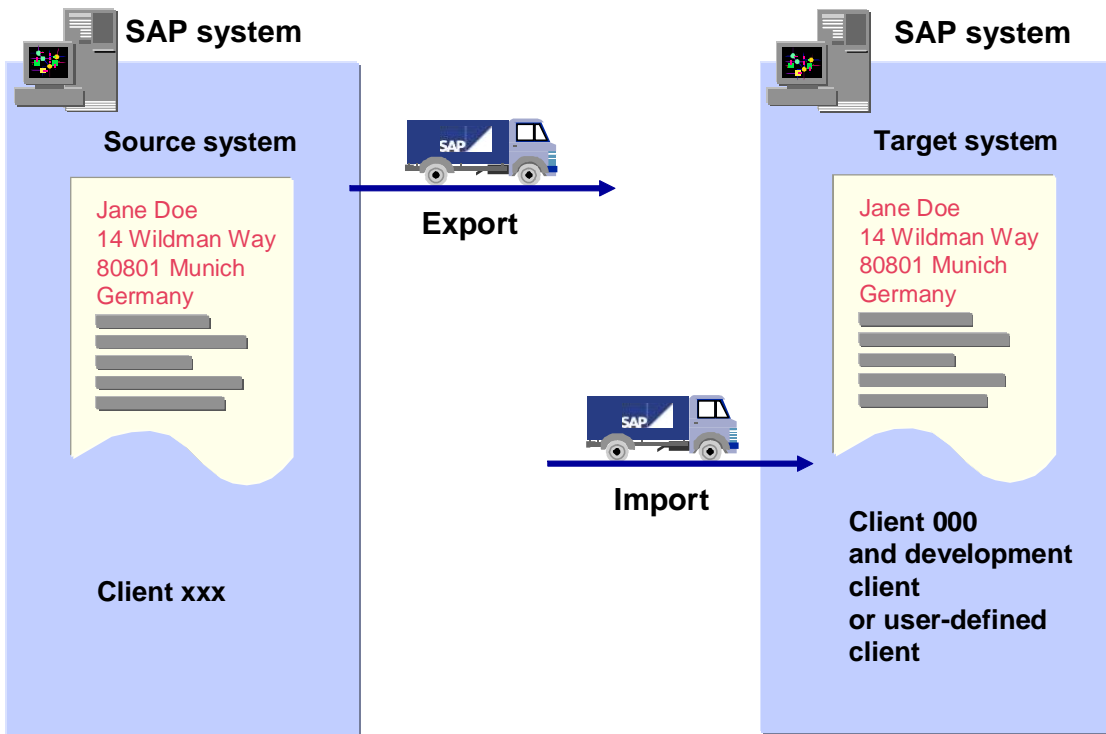
© SAP AG 2006

- As of Release 3.0A, the SAP transport system is available in full for transporting forms and styles. The first time an object is saved, a dialog box connected to the Workbench Organizer (WBO) is displayed.
- Workbench Organizer functionality allows you to transport objects out of development clients and into other clients without difficulty. Client 000 is the default target client for such transport. By transferring your forms and styles to this client, you make them available to all clients.
- As of Release 4.0, texts administered by SAPscript (standard texts, for example) can be transported using the Customizing Organizer. Prior to Release 4.0, use the Workbench Organizer for such transports.
- If you want to transport documents of this type, they must first be manually attached to a transport request in the Customizing Organizer. Enter the following after double-clicking on the task number:

R3TR TEXT <OBJECT>,<NAME>,<ID>,<L>

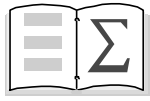
(OBJECT = text object, NAME = text name, ID = text ID, L = text language).

After attaching your document to a transport request, all standard transport functions of the Customizing Organizer are available.



© SAP AG 2006

- SAPscript texts, styles, and forms are client-dependent.
- When exporting a transport request, make sure that your SAPscript objects are available in the client in which the transport request is released.
- As of Release 3.0, export takes place from the source system client in which the transport request is released. Prior to Release 3.0, export takes place from source system client 000.
- As of Release 4.0, all transports are made into target system client 000 as well as into an additional client that can be either a development client or another client explicitly specified during the import process (using transport control program TP).
- In Release 3.0x/3.1x, all transports are made into target system client 000 or into another client explicitly specified during the import process (using transport control program TP).



You are now able to:

- **Understand the significance of assigning a form to a print program in Customizing**
- **Deal with alterations to the SAP standard shipment**
- **Comprehend the factors that must be taken into account when correcting and transporting SAPscript objects**

© SAP AG 2006

Exercises



Unit: Modifications



- Example of possible changes that could be made to the SAP standard
- Illustration of how forms are assigned a print programming in Customizing



In most cases, standard forms are altered to reflect the needs of individual customers. In some cases, the print program must also be altered to reflect these needs.

The purpose of this exercise is to familiarize you with a real SAP print program and SAP form.

- 1 Find in Customizing configuration area that assigns forms and print programs for SD Order output documents.
 - 1-1 See Appendix A.
Choose *Tools* → *Business Engineer* → *Customizing* → *Execute project* → *SAP Reference IMG*
 - 1-2 Follow the following path within the IMG
 - ☐ Sales and Distribution
 - ☐ Basic Functions
 - ☐ Output Control
 - ☐ Output Determination
 - ☐ Output Determination Using the Condition Technique
 - ☐ Maintain Output Determination for Sales Documents
 - ☐ Maintain Output TypesThen select Output type BA00 and a processing routine.
 - 1-3 What are the names of the program and of the form for the BA00 Order Confirmation?

- 2 Display the sales confirmation for an order on your screen. Note the PO number.
- 2-1 Choose *Logistics* → *Sales and Distribution* → *Sales Order* → *Display*
 - 2-2 Enter the order number. Your instructor will provide a sales order number for you.
 - 2-3 Print the screen output using *Sales document* → *Issue output to* → *Screen*

Select output type BA00 and choose *Execute*.

When the *Issue Output* screen displays, select *Execute* again.
 - 2-4 What is the PO Number? This is also known as the reference number.
- 3 What is the screen field name for the PO number?
- 3-1 On the *Display Sales Order* screen choose *Enter*; use the same sales order as in 2.
 - 3-2 Find the PO number on the overview screen
 - 3-3 Position your cursor on the field. Choose F1 and then *Technical information*.
What is the screen field name?
- 4 Display the form for the sales confirmation. Find the program symbol that is used to display the PO number. To do this, switch to the Form Painter.
- 4-1 Which window on the FIRST page corresponds to the area on the printout that contains the PO number?
 - 4-2 What is the field name?
Note that the table name is different from the screen name in exercise 3; however, the field name is the same. This is common, but not a rule.
- 5 Display the print program for the sales confirmation. Review the structure. To do this, switch to the ABAP Editor.
- 5-1 In the coding start at FORM PROCESSING and observe the following structure of the program:
 - Get the data for the form.
 - Open the form.
 - Print header information
 - Print item information
 - Print ending information
 - Close the form.
 - 5-2 Observe the following structure of ITEM_PRINT:
 - Print Item header information
 - Loop record-wise through an internal table (tvbdpa)
 - Create print condition of various items for the form.
 - Endloop

- 6 Observe that the transparent tables for the Sales Order are not available to the print program.
- 6-1 Some of the sales order tables are VBAK (header), VBAP (item), and VBKD (business data). Are they available in the program?
 - 6-2 Is the transparent table that contains the PO number in 3 available to the program?
 - 6-3 Can you find the structure table that the form uses to display the PO number?
- 7 The business unit would like the sales confirmation to include the telephone number of the person that placed the order. It should be placed in the same window as the PO number. Develop a strategy for accomplishing this.
- 7-1 This field can be found on the order. On the *Display Sales Order Overview* screen choose *Goto* → *Header* → *Purchase order data*.
 - 7-2 What is the field name on the screen?
 - 7-3 What would the symbol name be in the form?
 - 7-4 What other things would you have to consider when adding this field to the window?



Unit: Modifications



- Become familiar with an SAP print program
- Become familiar with an SAP form
- Learn how an SAP print program and an SAP form are related using an example



In most cases, standard forms are altered to reflect the needs of individual customers. In some cases, the print program must be adjusted as well.

The program and the form for the BA00 order confirmation are as follows:

Program: **RVADOR01**
Form: **RVORDER01**

The order number depends on the sales order.

The screen field name for the purchase order number is as follows:

VBKD-BSTKD

The window on the FIRST page containing the order number is called:

INFO (see print preview).

The field name in the INFO window is called:

VBDKA-BSTKD

The sales document tables are **NOT** available in the program.

Transparent table **VBKD** is **NOT** available in the program.

Table **VBDKA** is available in the program.

The field name on the screen for the sold-to party's telephone number is:

VBAK-TELF1

In the form, the symbol is called:

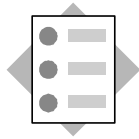
VBDKA-TELF1

Ensure that there is enough space available in the window to add an additional line. If not, you have to increase the size of the window. If you only have to add one line, you do not need to change the size of the other windows.

Contents:

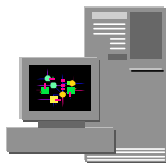
- System fonts and printer fonts
- System barcodes and printer barcodes

© SAP AG 1999



After completing this unit, you will be able to:

- **Understand how system fonts and printer fonts are related**
- **Comprehend the factors that you must consider when maintaining system fonts and printer fonts**



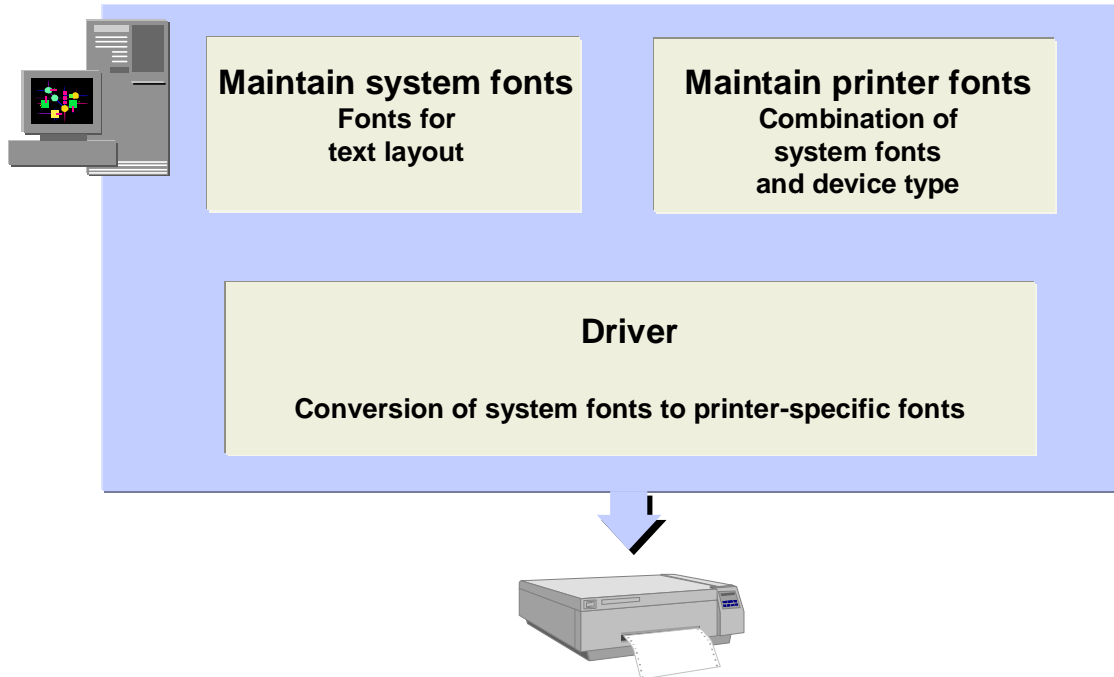
Enter text
with the
fonts used
by SAPscript

SAPscript font maintenance

- Font families
- System fonts
- Printer fonts/AFM metrics
- System bar codes
- Printer bar codes

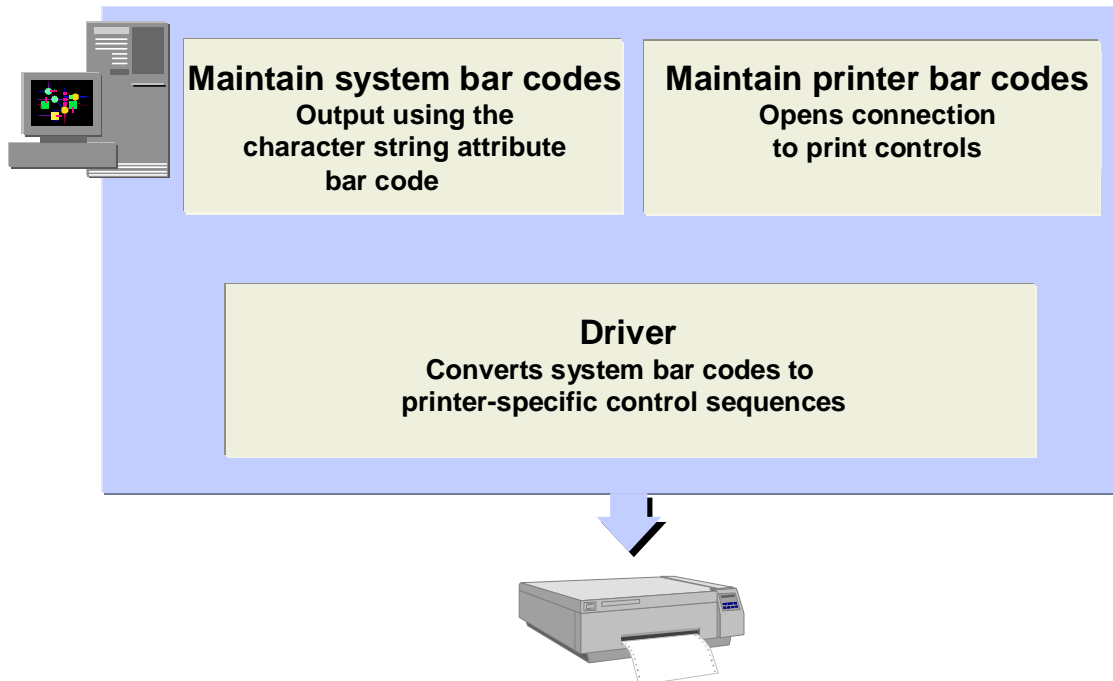
© SAP AG 2006

- All fonts and bar codes used in SAPscript are administered in the SAPscript font maintenance transaction. To access this transaction, choose *Tools -> Form Printout -> Administration -> Font*.
- The usable font names for system and printer fonts are entered in font families along with an ID to show whether the fonts are proportional or non-proportional.
- System fonts represent the font selections that can be used for styles and forms. They are independent of available output devices.
- Printer fonts are the fonts that are actually available on the various device types (printers, screens, telex machines and so on) that SAPscript uses for its output.



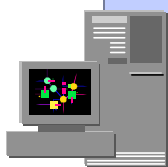
© SAP AG 2006

- SAPscript provides for automatic conversion between system fonts appearing in a given document and printer fonts installed on a particular printer. To support this conversion, you can maintain replacement fonts for each font family.
- Conversions between system fonts and printer fonts can be displayed in font maintenance. You can choose to display the conversion of a given system font for all printer types, or the conversion of all system fonts for a single printer type.
- The printers are controlled by various printer drivers which convert the internal SAPscript output format (OTF, output text format) into control sequences which can be interpreted by printers. The control sequences are mostly defined using print controls. You assign drivers to device types in the *Spool Administration* transaction.



© SAP AG 2006

- The *bar code* character string attribute allows you to output bar codes using SAPscript. You maintain the various bar code types independently of any device as system bar codes.
- For printer types which support bar code printing, the printer bar code establishes a connection with the device-specific control sequence known as a print control. You maintain print controls in the *Spool Administration* transaction. To access this transaction, choose *Tools -> CCMS -> Print -> Spool Administration*.
- When a bar code is printed, a print control called the bar code prefix is first sent to the printer. This is followed by the bar code data (for example, an 8-digit number). Finally, another print control, called the bar code suffix, is sent.



Font maintenance

Printer fonts

HPLJ4	Courier	120	X	_	10	CPI	SF003	SF003
-------	---------	-----	---	---	----	-----	-------	-------

Printer bar codes

HPLJ4	EAN8	SBP01	SBS01
-------	------	-------	-------

Spool administration

HPLJ4	SBP01	xxxxxxxx
HPJL4	SBS01	yyyyyyyy

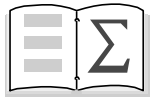
Printer command for activating
and deactivating bar codes

HPLJ4	SF003	zzzzzzzz
-------	-------	----------

Printer command for
font conversion

© SAP AG 2006

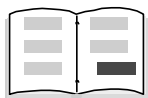
- A print control is a control sequence for an output device which is addressed using a symbolic name. You use the spool administration transaction to maintain print controls for individual device types.
- Print controls are responsible for controlling printer fonts and printer bar codes. In the font maintenance transaction, only the symbolic name of the print control is entered for printer fonts and printer bar codes.



You are now able to:

- **Understand the differences between system fonts and printer fonts.**
- **Maintain system and printer fonts.**

© SAP AG 2001



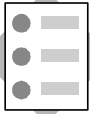
- **This unit contains additional information you can reference.**
- **This information is not part of the standard training course, therefore, parts or all of it may not be covered during the course.**

© SAP AG 2001

Contents:

- SAPscript programming interface functions
- Initializing and changing text
- Searching for text
- Updating text
- Saving text directly

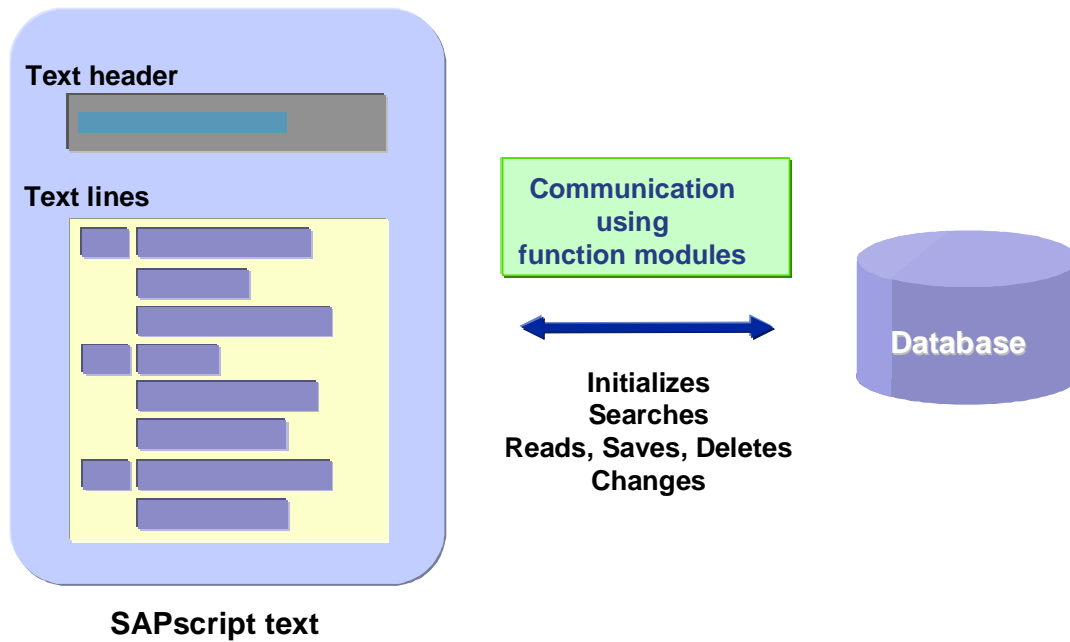
© SAP AG 1999



At the conclusion of this unit, you will be able to:

- **Name the functions and purposes of the most important SAPscript function modules (read, change, save, delete) in document administration.**
- **Understand how to save SAPscript texts using the update procedure.**
- **Describe how application transactions work that use SAPscript texts.**

© SAP AG 2001



© SAP AG 2002

- The programming interface consists of a range of function modules used to create, change, and administer SAPscript text.
- Typical functions include:
 - Initializing and changing text using the Editor
 - Generic searching for text
 - Reading, saving, and deleting text
 - Updating text in the database

Work areas for SAPscript text in the print program:

HEADER: Text header (field string with Dictionary structure THEAD)

LINES: Line table (internal table with Dictionary structure TLINE)

INIT_TEXT: Initializes work areas in the program required to create a new text.
The HEADER is filled with administration data.
The LINES are deleted.

EDIT_TEXT: Calls the Editor, which allows you to enter data into a new or existing text.

The HEADER and LINES serve as transfer parameters.

© SAP AG 2001

- You should prepare a text header and line table for every document to be edited by an applications program.
- The text header contains all administration data belonging to a document.
- The line table contains the text lines.
- When you create a new text, initialize the work areas using INIT_TEXT. The command CLEAR HEADER is **not** sufficient.
- To call the long text Editor, use EDIT_TEXT. You can control numerous functions, such as the display or change mode, using input parameters.
- When you choose *Text* → *Save* while using EDIT_TEXT, the function module SAVE_TEXT is automatically called. To suppress this function, use the parameter SAVE = SPACE.
- To transfer the text to be edited to the function module EDIT_TEXT, use HEADER and LINES.

SELECT_TEXT: **Selects text on the basis of the search criteria given by the user.
Displays a text catalog**

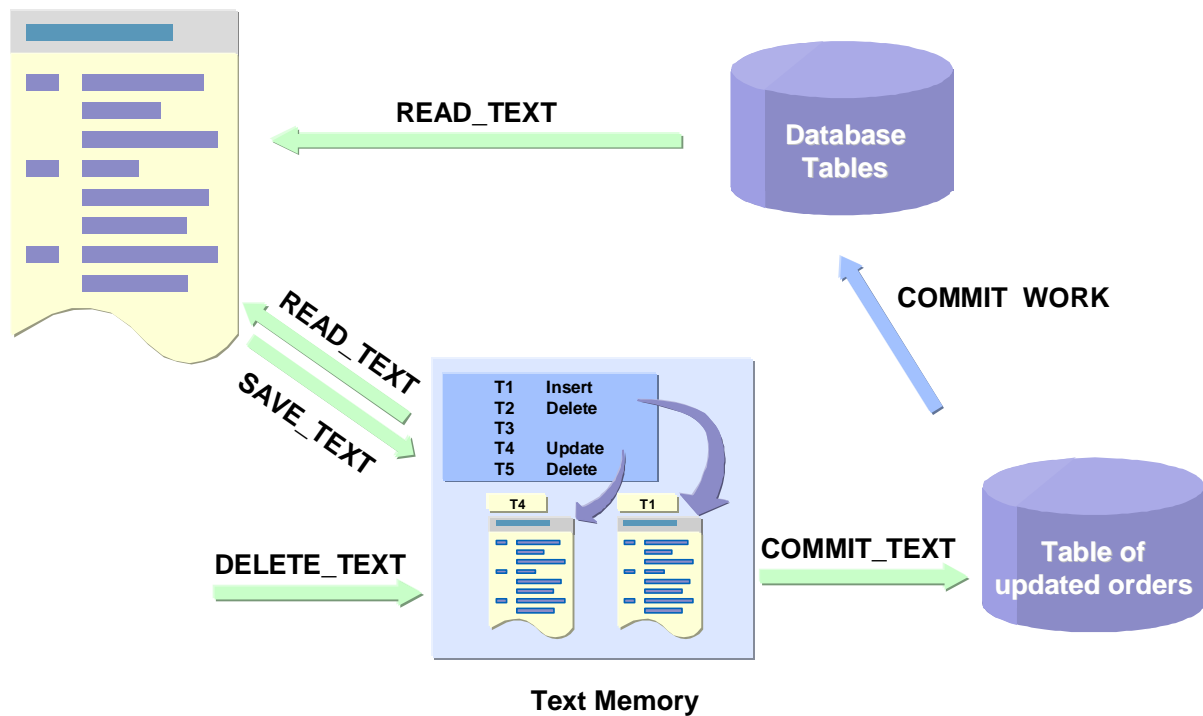
Entry: **Text key (generic entries are permitted)**
Output: **Table with header information on all texts found**

© SAP AG 2002

- The function module SELECT_TEXT generates a table with the text headers of all text modules corresponding to the search key.
- SAPscript text is client-specific.
- The text key contains four components:
 - OBJECT - Text object
 - ID - Text ID
 - NAME - Text name
 - LANGUAGE- Text language
- You can select text generically with SELECT_TEXT by using an asterisk (*).

The Text Update Process

SAP



© SAP AG 2002

- Most SAPscript documents in the application programs are saved via the update procedure.
- The R/3 Update concept is explained fully in the R/3 Documentation.
- The backup mode (V=update, D=direct without update) is an attribute of an entire document class, that is, of all documents allocated to a particular text object. The developer defines this mode in a separate transaction (*Tools -> SAPscript -> Administration -> Settings*).
- When you save text via the update procedure, all function modules work with the text memory.
- The text memory contains all of the text changes made in the applications program.
- To store changes to text in the database:
 - Use COMMIT_TEXT
 - Use the ABAP/4 command COMMIT WORK
- If you fail to follow the two steps above, all text changes are lost and the update is terminated.

READ_TEXT: Reads a single text from the text memory or database in the work area

Entry: Text key

Output: HEADER and LINES

SAVE_TEXT: Stores a single text from the work area in the text memory or database

Entry: HEADER and LINES

DELETE_TEXT: Deletes one or more texts in the text memory or database

Entry: Text key (can be generic)

© SAP AG 2002

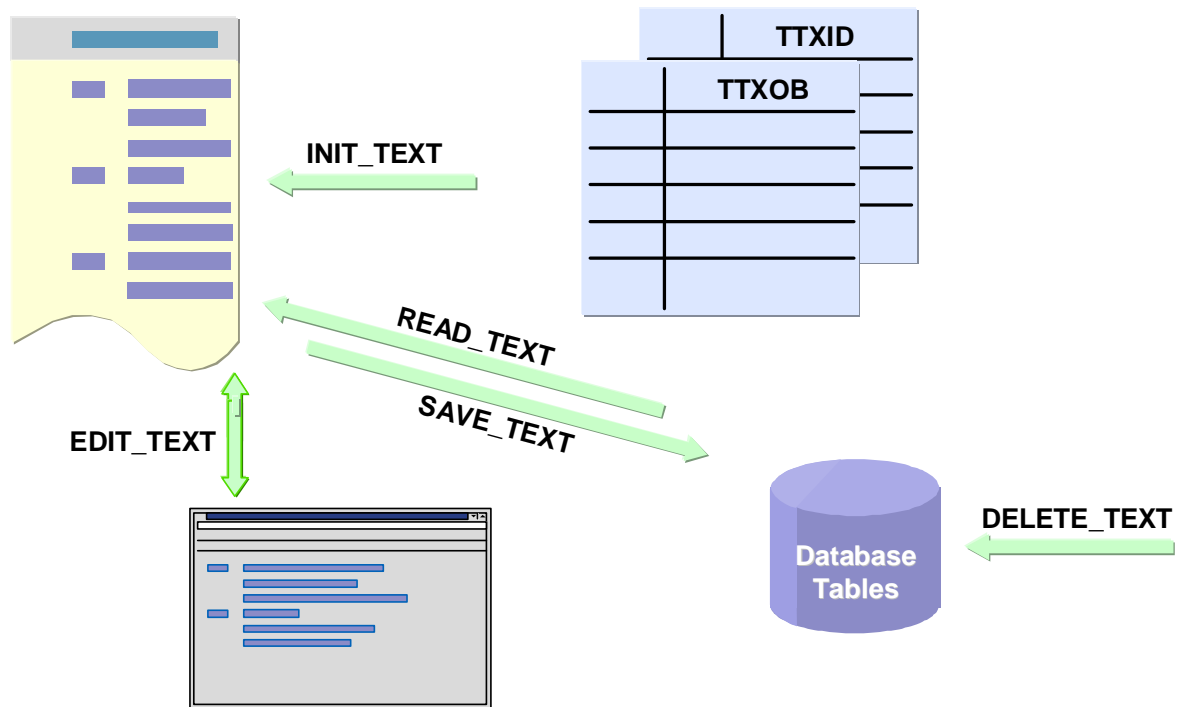
- The function module READ_TEXT reads text from the text memory or database into the internal work areas HEADER and LINES, which are parts of the print program.
- To define the text key you must specify:
 - Object
 - Name
 - ID
 - LanguageGeneric entries are not permitted.
- To store text module in the text memory or database, use SAVE_TEXT.
- The parameter FUNCTION returns one of the following:
 - I = New text
 - U = Changed text
 - D = Deleted text
- To delete text from the text memory or database, use DELETE_TEXT.
Generic entries for the text key are permitted.
- Text is either deleted immediately or recorded in the text memory, depending on the text object.

COMMIT_TEXT: Updates text from the text memory in the update file
Text memory is empty after successful call

COMMIT WORK: Call from within an application program (ABAP command)
Necessary for storing texts in the database

© SAP AG 2002

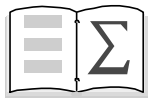
- When the R/3 System saves text (or data in general) via update, this takes place asynchronously in a separate process, namely the update task.
- To store text changes for the text update procedure:
 - Use COMMIT_TEXT
 - Use the ABAP/4 command COMMIT WORK
- COMMIT_TEXT **does not** replace the ABAP/4 command COMMIT WORK.
- You can call COMMIT_TEXT and COMMIT WORK only once per logical unit of work (LUW), and you must do so at the end of the procedure.
- If you fail to execute either COMMIT_TEXT or COMMIT WORK, all text changes are lost.
- After COMMIT_TEXT has been executed, the text memory is empty.



© SAP AG 2002

SAPscript Editor

- As an alternative to the update file, you can save text directly in the database. This applies to all text, the text object of which has the backup mode D.
- The backup mode (V=update, D=direct without update) is an attribute of an entire text class, that is, of all text allocated to a particular text object. The developer defines this mode in a separate transaction (*Tools -> SAPscript -> Administration -> Settings*).
- For mass processing of text (such as transport, client copies, or company reorganization), you can specify the backup mode temporarily as D. To make use of this temporary mode under **SAVE_TEXT** or **DELETE_TEXT**, use the parameter **SAVEMODE_DIRECT**.
- When you define **SAVEMODE_DIRECT = 'X'**, the function modules **SAVE_TEXT** and **DELETE_TEXT** proceed as shown above.
- **SAVEMODE_DIRECT** works only once, for a single call of the corresponding function module.
- **NOTE:** Do not mix normal updates with **SAVEMODE_DIRECT** under any circumstances.



You are now able to:

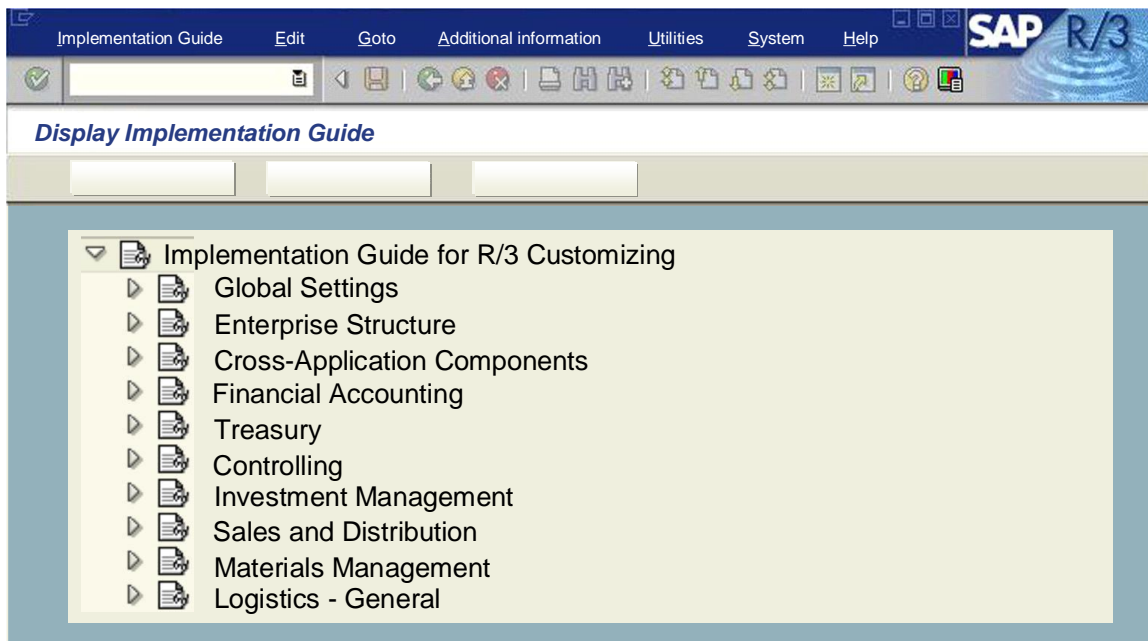
- **Name the functions and purposes of the most important SAPscript function modules (read, change, save, delete) for text administration.**
- **Understand how to save SAPscript texts using the update procedure.**
- **Describe how application transactions work that use SAPscript texts.**

© SAP AG 2001

Contents:

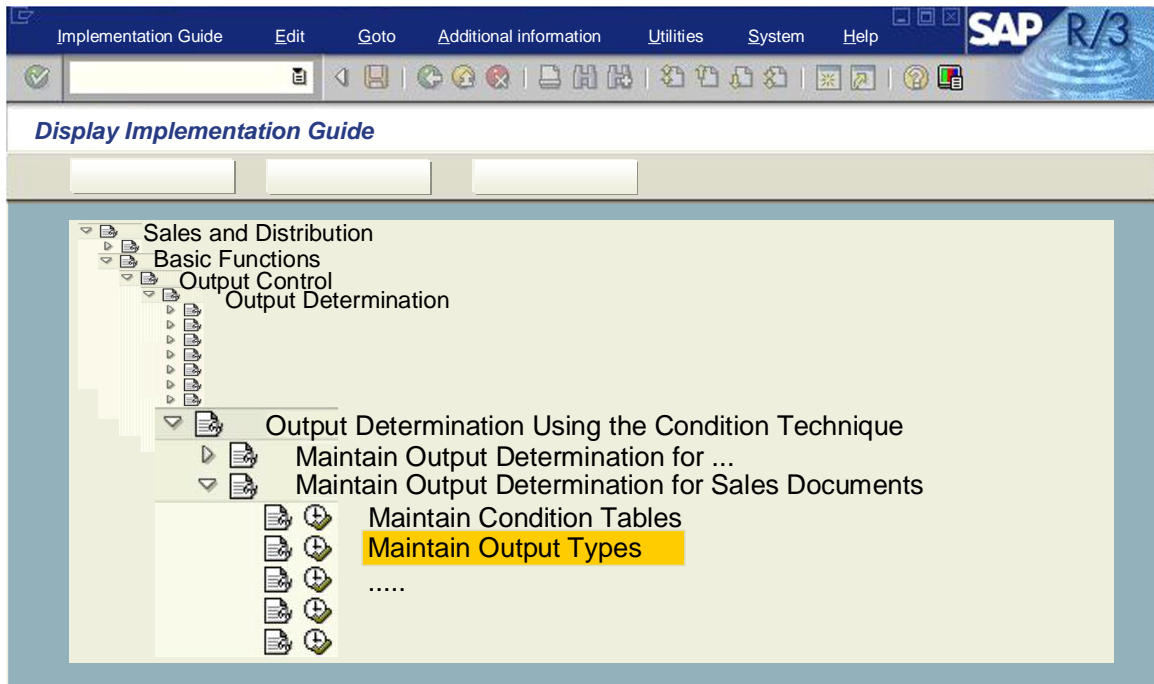
- Customizing Form Overview
- Invoice form (SD)
- Order form (MM)
- Account statement (FI)

© SAP AG 1999



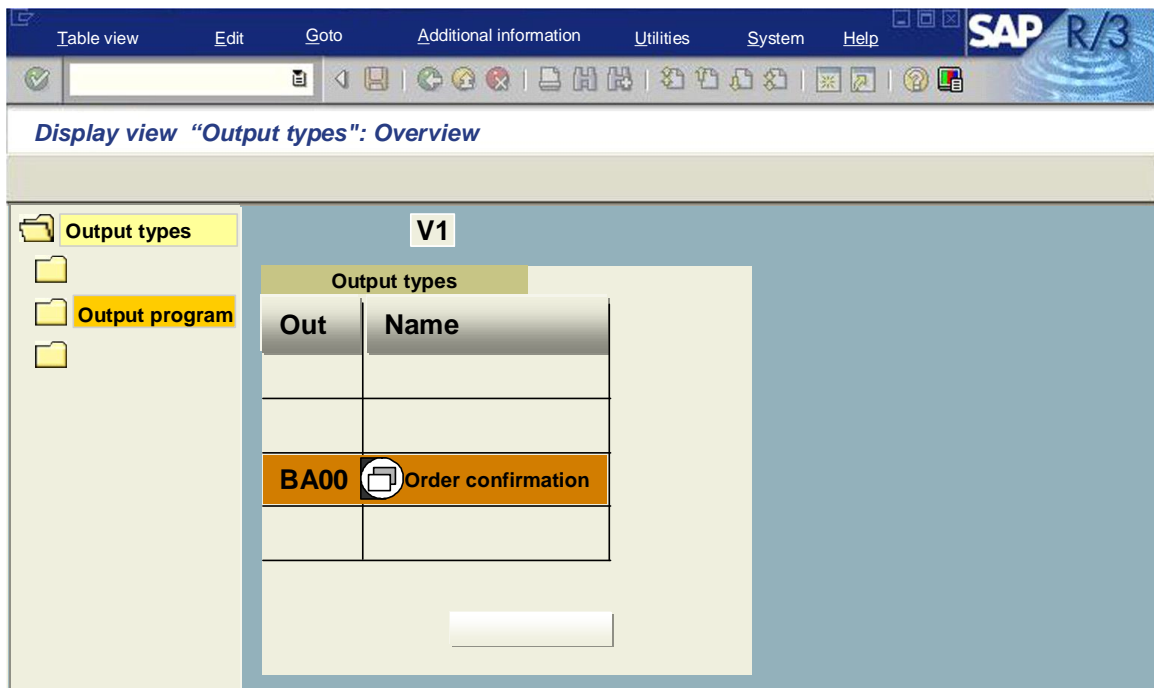
© SAP AG 2002

- To access the R/3 Customizing Implementation Guide, choose *Tools* → *Business Engineer* → *Customizing*, then *SAP Reference IMG*.
- The Enterprise Implementation Guide (IMG) is a subset of the SAP Reference IMG (from which it is generated).
- From this menu, you can branch to the appropriate application (such as Financial Accounting (FI), Logistics General (LO), or Sales and Distribution (SD)).



© SAP AG 2001

- To assign print programs to forms within SD, choose *Sales and Distribution* → *Basic Functions* → *Output Control* → *Output Determination* → *Output Determination Using the Condition Technique* → *Maintain Output Determination for Sales Documents* → *Maintain Output Types*.



- Select an output type and double-click on *Output programs*.

Table view Edit Goto Additional information Utilities System Help SAP R/3

Display view "Output programs": Overview

Output types

Output type BA00

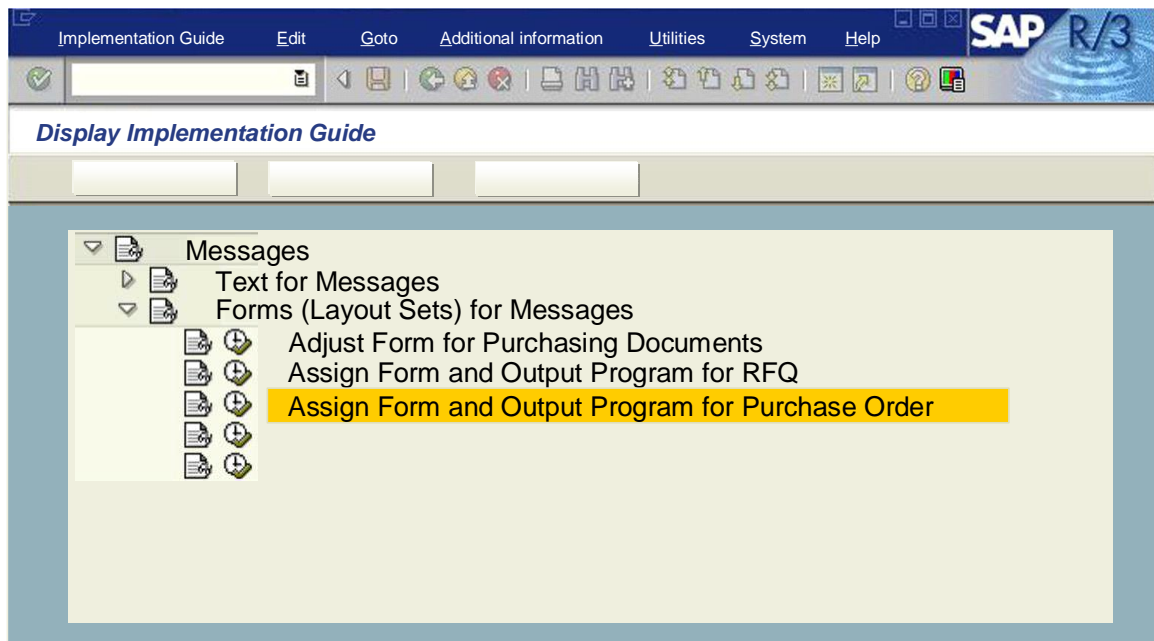
Application V1

Output programs

Med	Name	Prog	Form	Form
1	Print output	RVADOR01	ENTRY	RVORDER01
2	Telefax	RVADOR01	ENTRY	RVORDER01
6	...			

© SAP AG 2002

- Here, the assignment of print program and form is determined, according to the message type.
- The message type indicates the type of form to be used.
- The medium indicates how the message is to be sent.
- Column 3 contains the entries for the standard print programs.
- If you want to change the print program or the form, you should enter your own object (or a modified object) at this point.



© SAP AG 2001

- To assign print programs to forms within Materials Management (MM), select *Materials Management* → *Purchasing* → *Messages* → *Forms for Messages* → *Assign Form and Output Program for Purchase Order*.

Implementation Guide Edit Goto Additional information Utilities System Help

SAP R/3

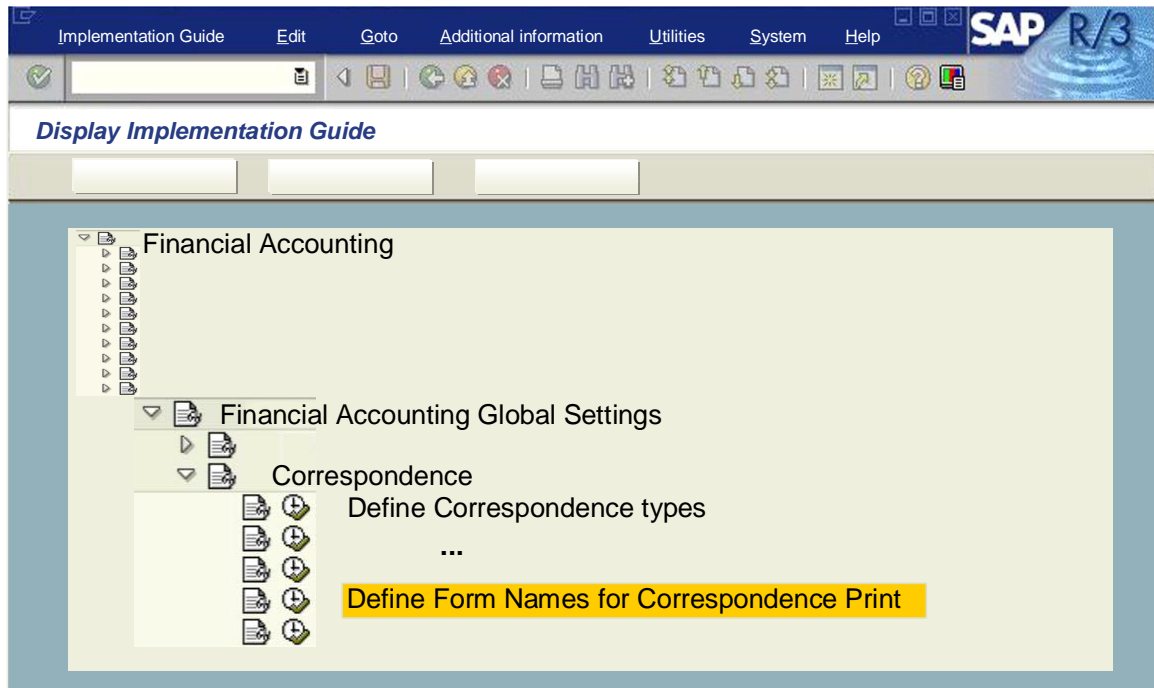
Change view "Output: Output programs": Overview

Out...	Name	Med	Prog	Form	Form
AUFB	Sales conf. dunn.		SAPFM06P	ENTRY_AUFB	MEDRUCK
MAHN	Dunning		SAPFM06P	ENTRY_MAHN	MEDRUCK
NEU	Sales order		SAPFM06P	ENTRY_NEU	MEDRUCK

Text Text

© SAP AG 1999

- Depending on the output type, you use this view to assign a print program to a form.
- The overview contains the entries of the standard programs.
- If you want to change the print program or the form, enter your own object on this screen.



- To assign print programs to correspondence within the FI component (for example, an account statement), choose *Financial Accounting* → *Financial Accounting Global Settings* → *Correspondence* → *Define Form Names for Correspondence Print*.

Implementation Guide Edit Goto Additional information Utilities System Help

SAP R/3

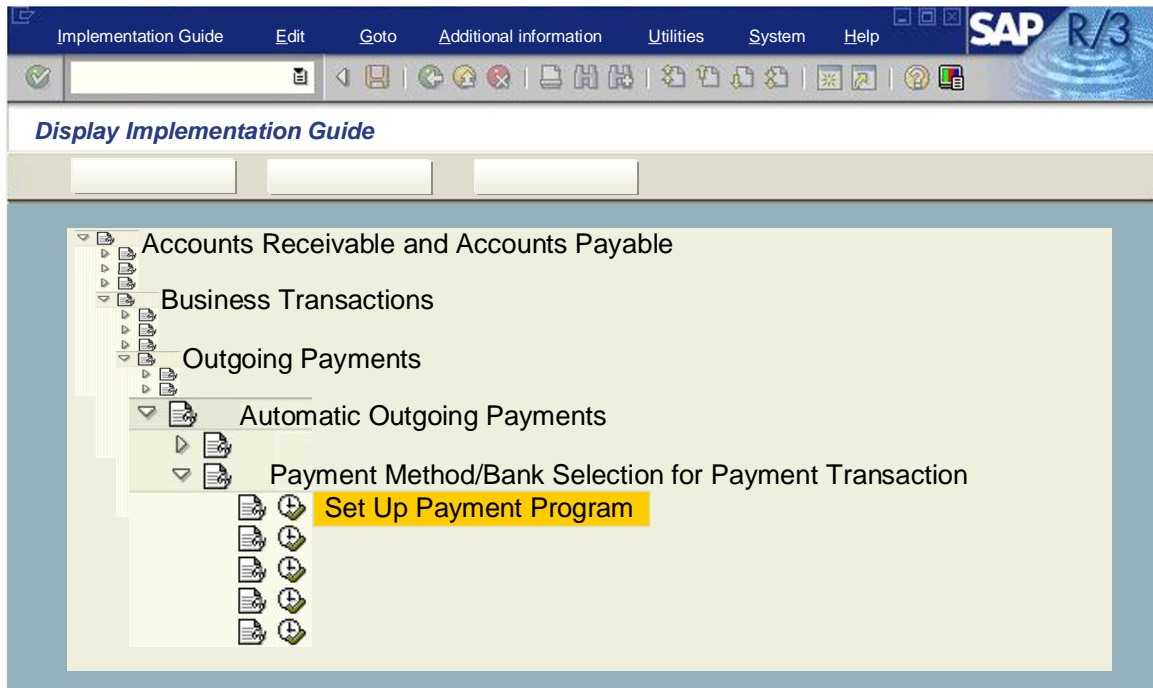
Change view "Forms for correspondence": Overview

	Prog	Prog	FoID	Form
	RFKORD00	Print program: Payment confirmation	50	F140_PAY_CON_050
	RFKORD00	Print program: Payment confirmation	51	F140_PAY_CON_051
	RFKORD00	Print program: Payment confirmation	52	F140_PAY_CON_052
	RFKORD10	Print program: Account statement	VB	F140_ACC_STAT_03

Position

© SAP AG 2002

- Use this screen to assign print programs to forms.
- The entries of the standard programs are recorded.



© SAP AG 1999

- To change the entry of the print program and the form to outgoing payments within FI, choose *Financial Accounting* → *Accounts Receivable and Accounts Payable* → *Business Transactions* → *Outgoing Payments* → *Automatic Outgoing Payments* → *Payment Method/Bank Selection for Payment Transaction* → *Set Up Payment Program*.

SAP R/3

CtryPaymentMeth.

Maintain Payment Program Config.: Countries

Company codes

Country	Name
AR	Argentina
AT	Austria
AU	Australia
BE	Belgium
BR	Brasil

© SAP AG 2002

- From here on out, a different menu path must be chosen to change print programs and forms:
- To change the print program, choose *Ctry Payment Methods*.
- The subsequent dialog window displays the available countries.
- Double-click to select the desired country.

Maintain Payment Program Config.: Country Pmnt Methods - List

Country **DE** Germany

Method	Name (in national language)
1	TRLO: individual payment
A	Bank direct debit
C	check (with checkmanagement)
E	Automatic debit

© SAP AG 1999

- Double-click to select the payment method from those displayed.
- In our example above, payment method C (check) has been selected.

Outgoing Payments FI (4)

SAP

Payment method Edit Goto Utilities System Help

SAP R/3

Maintain Payment Program Config.: Country Payment Methods - Details

Country key **DE** Germany

Payment method **C** Check

Payment method classification

<input checked="" type="checkbox"/> Check will be created	<input type="checkbox"/> Check/bill of exchange
<input type="checkbox"/> ...	<input type="checkbox"/> ...
<input type="checkbox"/> ...	<input type="checkbox"/> ...
<input checked="" type="checkbox"/> Allowed for personnel payments	<input type="checkbox"/> Bill/exchange ...

Required master record specifications

<input type="checkbox"/> Street, P.O. box ...
<input type="checkbox"/> ...
<input type="checkbox"/> ...

Update specifications

Doc. type for payment	<input type="checkbox"/>
...	<input type="checkbox"/>
...	<input type="checkbox"/>
...	<input type="checkbox"/>

Form printout

Name of print program	RFFOUS_C	Key in code line	01
Name of print dataset	LIST1S	...	

© SAP AG 1999

- Enter changes to your print program here.

SAP R/3

Paying CCs

Maintain Payment Program Config.: Company Codes

Countries

Country	Company name	City
0001	SAP AG	Walldorf
AR01	Template AR	Argentina
AT01	Template AT	Austria
AU01	Template AU	Australia

© SAP AG 1999

- To change your form, choose *Paying company codes*.
- The dialog window displays the company codes. Double-click to select a company code.
- The same payment method is displayed as in the print programs. Choose the appropriate entries. In the example above, the payment method is *Check*.


Maintain Payment Program Config.: CC Pmnt Methods-General Data

Form data

Maintain Payment Program Config.: CC Pmnt Methods-Form data

Paying co.code **DE01** Country Template
Payment method **C** Check (with checkmanagement)

Forms

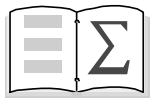
Form for payment transfer **F110_PRENUM_CHK**  Display form
Next form
Correspondence sort variant

Paid items printout

Line items per form **99**
☐ Pmnt adv. if form is full
☐ Extra form if form is full
Line items sort variant

© SAP AG 1999

- To go to the entry for the payment medium, choose *Form data*.



You are now able to:

- **Navigate in Customizing to assign forms to print programs for SD, MM, and FI**

© SAP AG 1999

Contents:

- Inserting a company logo using program RSTXLDMC
- Inserting bitmaps

© SAP AG 2001

Access TIFF graphic file on frontend

**Upload PC file as text module using
ABAP report RSTXLDMC**

Define new window in form

**Incorporate existing text element
in a new window**

© SAP AG 2002

- To insert a graphical element into a form using program RSTXLDMC (before Release 4.5), note the following:
 - The graphic to be inserted must be available on the front-end in TIFF 6.0 format (Tag Image File format).
 - In the form, define a new window into which you insert the graphic as text module (after uploading it into a standard text).
- To upload the TIFF file into a standard text, use the ABAP report RSTXLDMC.
- As of Release 4.5 you can include graphic administration into the page layout of the form maintenance. Choose *Edit -> Graphic -> Create graphic*. Your graphic must be a BMP file. Choose *Import* to import your graphic into R/3.
You can upload BMP files directly from the graphic administration into a standard text. In the form maintenance initial screen choose *Environment -> Administration -> Graphic*.

RSTXLDMC: Creating Graphical Elements

SAP

Uploading TIFF Graphics into SAPscript Texts

File name and parameter for TIFF conversion

File name: /tmp/file.tif

Type (BMON=b/w, BCOL=color): BMON

Resolution for graphic (dpi):

☐ Graphic resident in printer

Parameters for standard text

Text name: ZHEX-MACRO-*

Text language: ST

Text ID: ...

Text title:

Line width for text: 132

Parameters for positioning the TIFF graphic

☐ Absolute positioning

Absolute x position:

Absolute y position:

...

© SAP AG 2002

■ Procedure:

1. Use RSTXLDMC to save the TIFF file on your frontend as text module.
2. The system generates a suggested name for the text: ZHEX-MACRO-*. The asterisk denotes the file type. In our example the file is BMON, so the resulting name is ZHEX-MACRO-BMON. Use the ID ST and the logon language.
3. For more information on input parameters, see the documentation on the report RSTXLDMC.
4. You can specify the exact positioning of the graphic when uploading. If you make no specifications, the system prints the graphic in the current page window.
5. To check the resulting change to the form, choose *Utilities* → *Test print*. Print the spool request created with the test print on a printer that is capable of printing graphics.

Page

...

Graphic

Name

Description

Left margin

Upper margin

Window width 0.00 CM

Window height 0.00 CM

Design / Text **Options**

Include Graphic

Stored on ...

Name

Object ID

☒ Black and White Bitmap Image

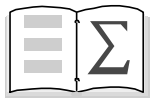
☐ Color Bitmap Image

Technical Attributes

Resolution DPI

© SAP AG 2006

- As of Release 4.6 you can import graphics and include them in your form using the *Graphic* tab in the Form Painter. To do this, choose *Graphic -> Create* followed by *Import*. Enter the file path and the name the graphic should have in the SAP system. Import the graphic and position the graphic window at the desired position in the form.
- If you want to resize the graphic in the form, recreate the graphic in the form and specify a lower resolution. You do not have to reimport the graphic into the SAP system.
- Using the SAPscript control statement BITMAP you can also insert graphics dynamically. For more information, see SAP Note number 307414.
- All graphics imported are managed in SAPscript graphics administration. To start graphics administration, choose *Tools -> Form Printout -> Administration -> Graphic*.
- As of Release 4.6C, graphics are displayed in the print preview.



You are now able to:

- **Insert graphics using program RSTXLDMC**
- **Insert bitmaps**

© SAP AG 2006

Exercise



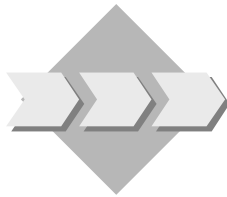
Unit: Appendix - Graphics

Topic: Include Graphics in Forms



At the end of these exercises you will be able to:

- Import a graphic into the SAP system and include it in a form



You would like to enhance your form by adding your company logo and also work with graphics in windows.

- 1-1 Import a bitmap graphic into the SAP system. Use transaction SE78. The name of the graphic in the system: demo_###. Choose a random bmp file on your PC. (Tip: Use the F4 help in SE78 to find a suitable graphic, for example in the folder C:\WINDOWS.)
- 1-2 Create a graphics window in one of the forms you have worked with already by inserting the graphic demo_###. Move the graphic to an appropriate position within the layout window.
Note: The graphic will always appear in the same position, irrespective of other content in the window.
- 1-3 Take a look at the syntax of the bitmap command you just created in the editor. Use this syntax to embed the graphic in your form again, but this time in a normal window.
Note: The position of the graphic will depend on the content of the window it is placed in.



Unit: Appendix - Graphics

Topic: Include Graphics in Forms

- 1-1 Enter the graphics administration transaction (transaction code: SE78). Choose function **Graphic -> Import** and import a bmp file of your choice.
- 1-2 Switch to form maintenance via menu path **Environment -> Form**. Create the graphic in the form as you would normally. Ask your course instructor for help if you have any questions.
- 1-3 In the graphics window, choose the *Edit Text* function. Copy the instruction in the text editor and paste it into a window of your choice in your form. Activate and execute the form.

