

Send, Receive, and Process Interactive Forms via Email in AS ABAP.

Jeff Gebo

NetWeaver RIG, SAP Labs LLC

Introduction

Creating Interactive Forms in AS ABAP

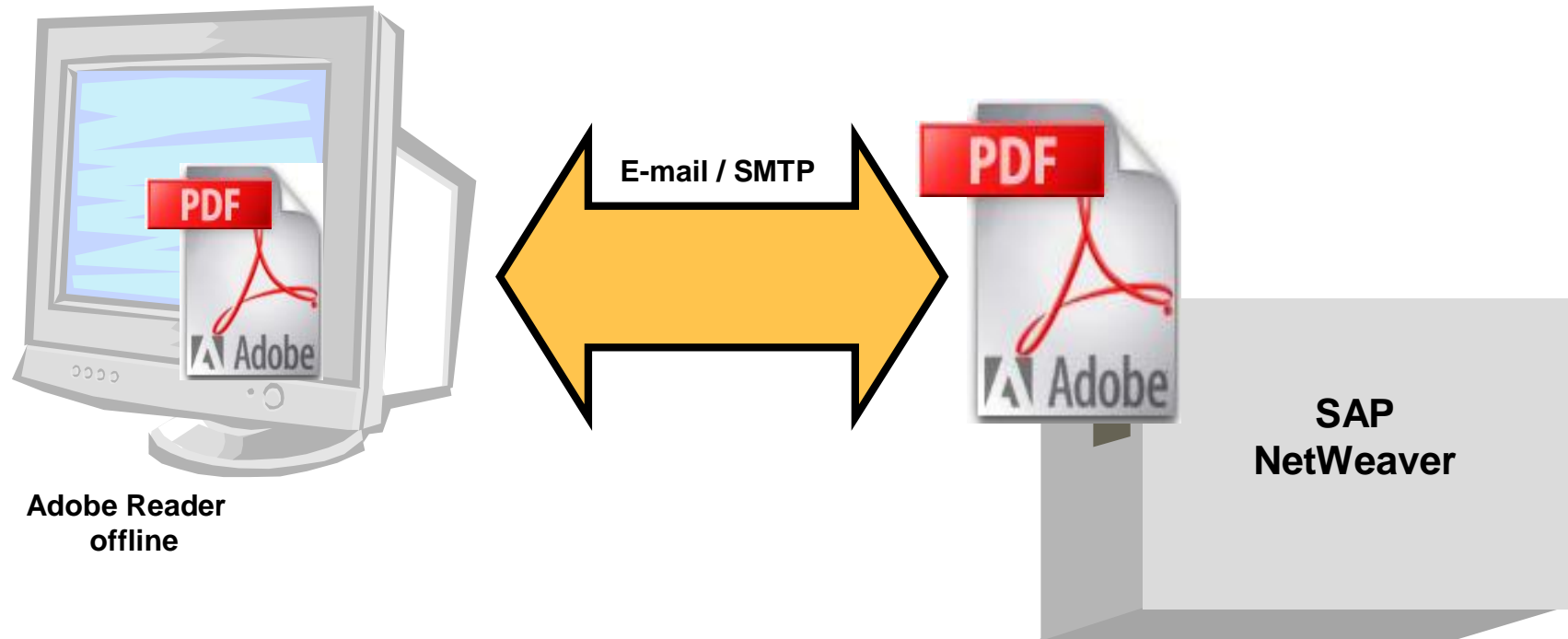
SMTP in AS ABAP

Sending Email with attached Interactive Forms

Processing Inbound Emails

Processing Interactive Forms

Interactive Form Offline Scenario



- User fills out form offline, submits form via email when online.
- No direct SAP access needed.
 - Client email program needed to send and receive emails with attachments.

Introduction

Creating Interactive Forms in AS ABAP

SMTP in AS ABAP

Sending Email with attached Interactive Forms

Processing Inbound Emails

Processing Interactive Forms

To create an Interactive Form you must create the form interface and then the form itself in transaction SFP.

Form Interface

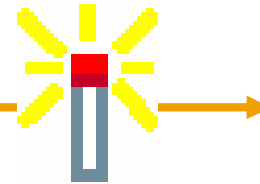
- **Represents the data structures you wish to display or have the user enter on the Interactive Form.**
- **Can be simple types, structures or tables.**

Form

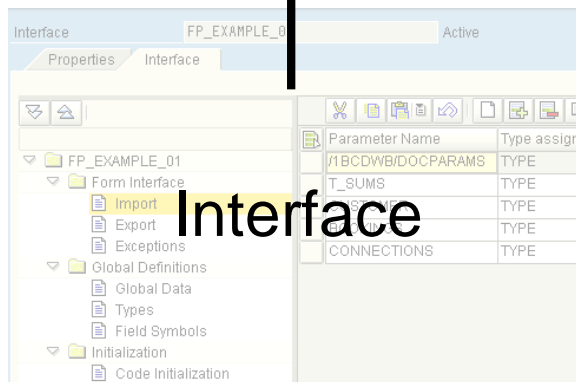
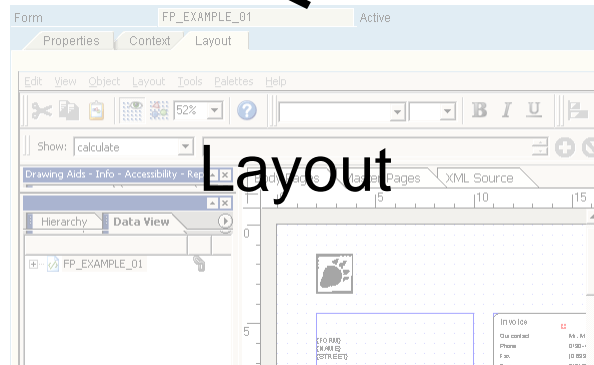
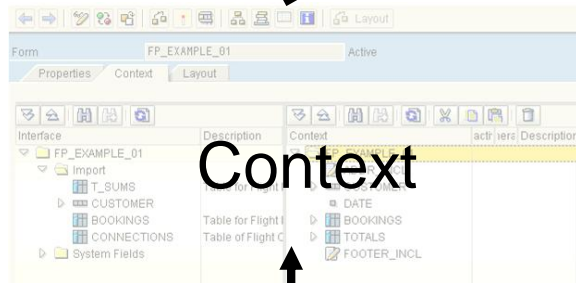
- **Create a form and assign the form interface to it. Data elements in the form interface can then be used in the form.**
- **LiveCycle Designer is displayed in the “Layout” mode of the form.**

The Tools Involved (Design Time)

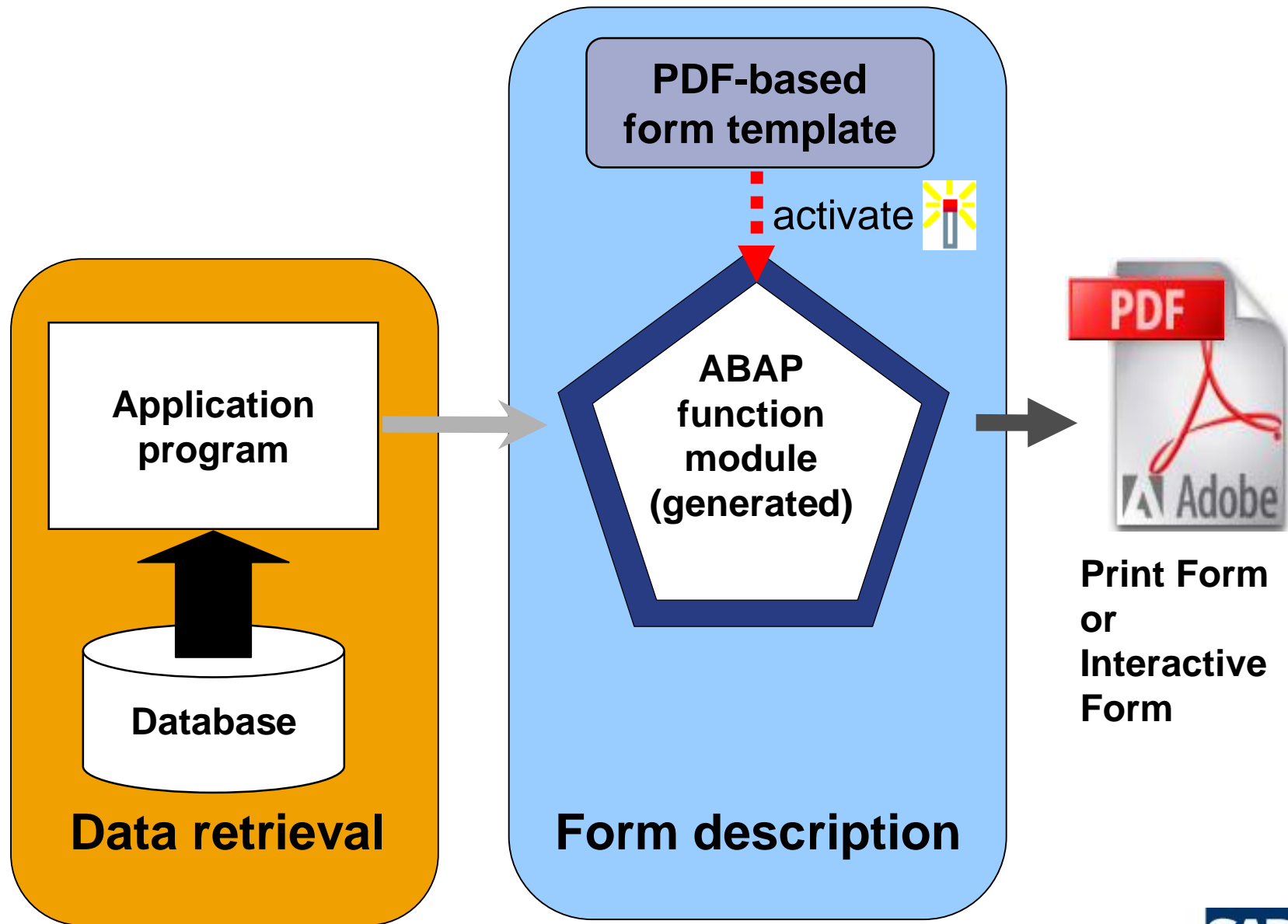
Form Template



```
FUNCTION  
/1BCDWB/SM00000001.  
DATA: %OUTPAR TYPE  
SFPOUTPAR,  
%DOCPAR TYPE  
SFPDOCPAR,  
...
```



What Happens at Run Time



Form Interface Example

Form Builder: Display Interface Z_CUST_BOOK_FLIGHT_GEBO

The screenshot shows the SAP Form Builder interface for the display interface Z_CUST_BOOK_FLIGHT_GEBO. The interface is active. The main workspace is divided into two panes. The left pane shows a tree view with the following structure:

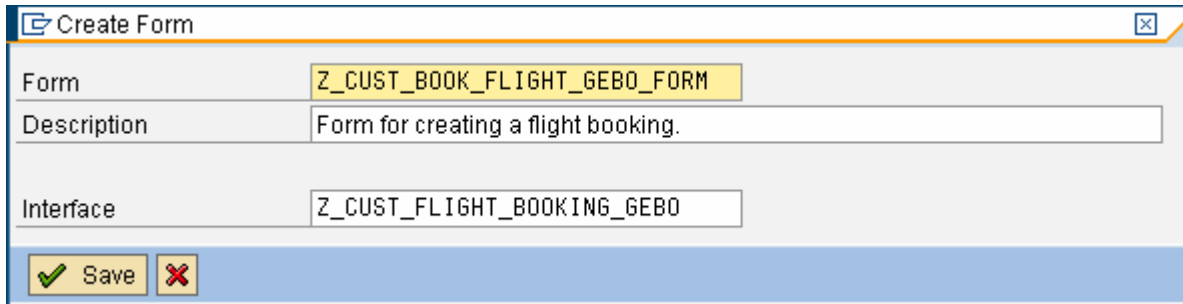
- Z_CUST_BOOK_FLIGHT_GEBO
 - Form Interface
 - Import
 - Export
 - Exceptions

The right pane shows a table of parameters:

Parameter Name	Type assignment	Type Name
/1BCDWB/DOCPARAMS	TYPE	SFPDOCPARAMS
RESERVED	TYPE	BAPISBODAT-RESERVED
BOOKING_DATA	TYPE	BAPISBONEW

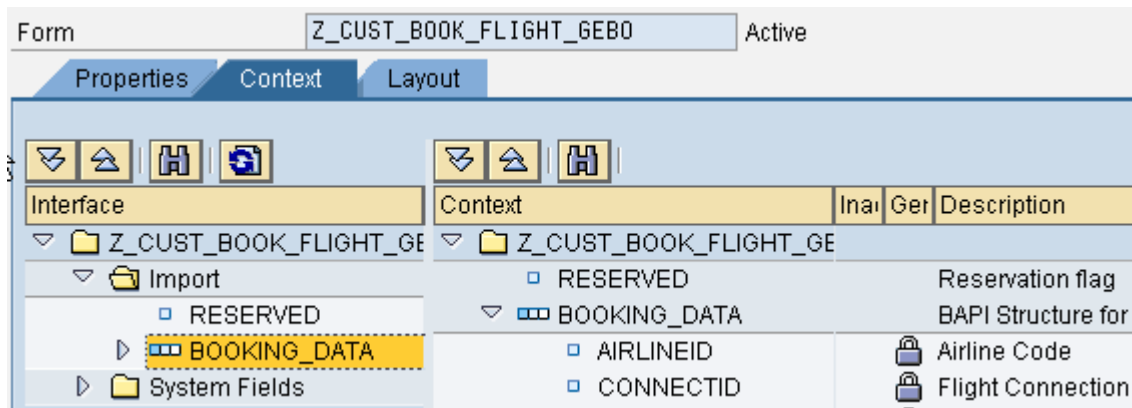
- Use transaction SFP to create/modify/view Form Interfaces.
- Add your data elements to the Import parameter list.

Creating Forms



Form: Z_CUST_BOOK_FLIGHT_GEBO_FORM
Description: Form for creating a flight booking.
Interface: Z_CUST_FLIGHT_BOOKING_GEBO

■ When creating a form you must assign it an interface.



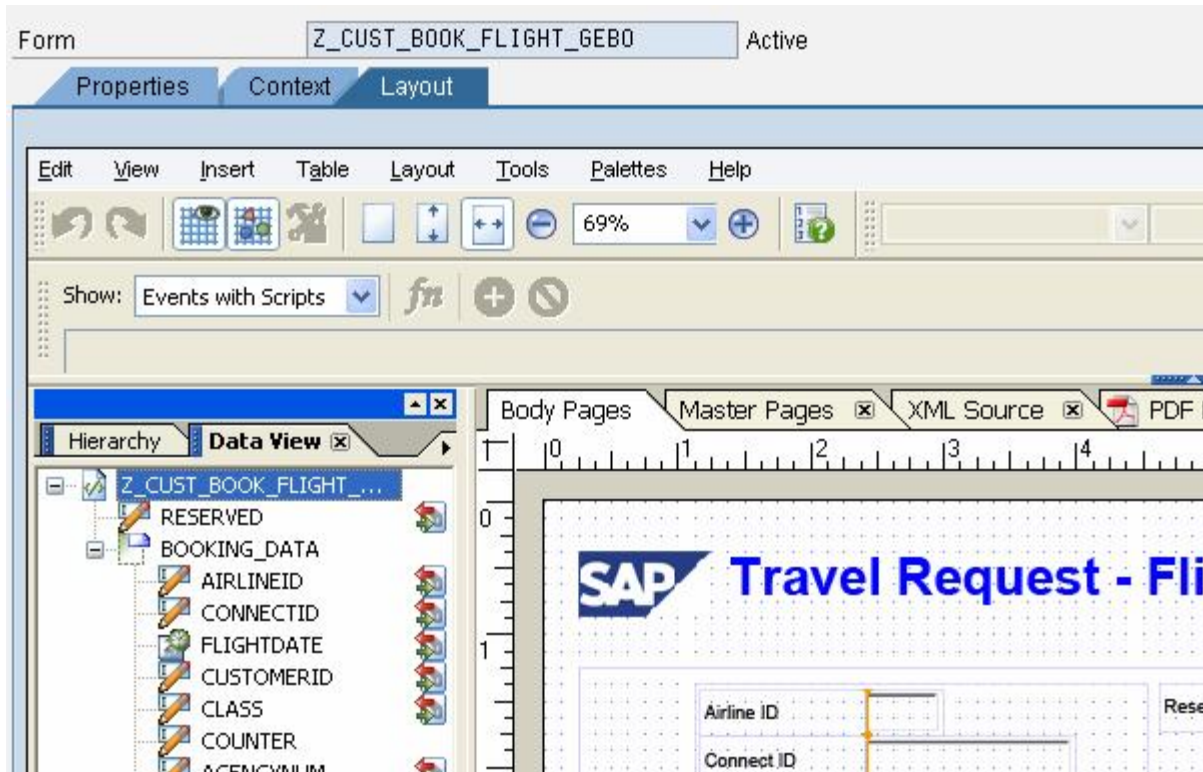
Interface	Context	Ina	Get	Description
Z_CUST_BOOK_FLIGHT_GE	Z_CUST_BOOK_FLIGHT_GE			
Import				
RESERVED	RESERVED			Reservation flag
BOOKING_DATA	BOOKING_DATA			BAPI Structure for
System Fields	AIRLINEID			Airline Code
	CONNECTID			Flight Connection

■ Create the forms context by dragging and dropping the interface's import elements to the forms context.

■ These context elements can then be bound to input fields on the form.

Designing Forms

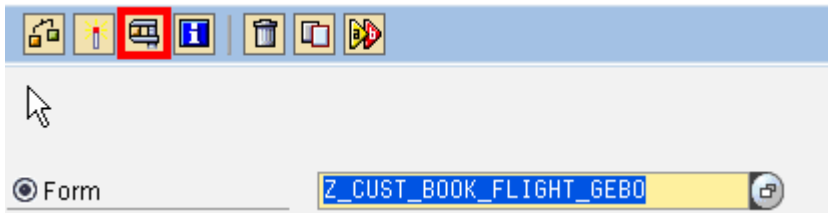
Select the Layout tab of the Form Builder, this will start the LiveCycle Designer.



Design your form as you would a standard print form. Just make sure not to set the input fields to read only!

Testing Forms

You can test your form by clicking the test icon.



This will display a function module that contains the same input parameters as the form's interface – just enter data into the data elements.

Test for function group	/1BCDWB/SM00000038
Function module	/1BCDWB/SM00000038
Uppercase/Lowercase	<input type="checkbox"/>
Import parameters	Value
/1BCDWB/DOCPARAMS	<Initial>
RESERVED	
BOOKING_DATA	00000000-00-0000000000 000000000000

Sections of the Application Program

* (1) Data retrieval and processing

```
SELECT ... FROM ...
```

```
...
```

* (2) Find out name of generated function module

```
CALL FUNCTION 'FP_FUNCTION_MODULE_NAME'...
```

* (3) Start form processing

```
CALL FUNCTION 'FP_JOB_OPEN'...
```

```
LOOP AT ...
```

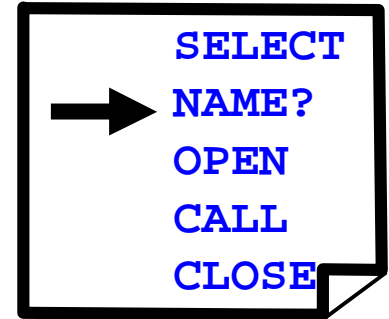
* (4) Call function module dynamically

```
CALL FUNCTION <generated function module> ...
```

```
ENDLOOP.
```

* (5) End form processing

```
CALL FUNCTION 'FP_JOB_CLOSE'...
```



DATA:

```
form          TYPE fpwbformname,  
fm_name      TYPE funcname.
```

*** (2) Find out name of generated function module**

```
CALL FUNCTION 'FP_FUNCTION_MODULE_NAME'  
  EXPORTING i_name      = form  
  IMPORTING  
    e_funcname      = fm_name.
```

Starting and Ending Form Processing

DATA:

```
fp_outputparams    TYPE sfpoutputparams  
fp_outputparams-getpdf = 'X'.  
...
```

* (3) Start form processing

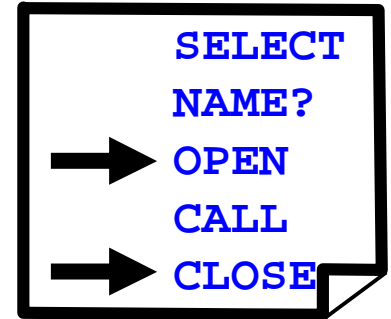
```
CALL FUNCTION 'FP_JOB_OPEN'...
```

* set output parameters like printer, preview...

```
CHANGING ie_outputparams = fp_outputparams ...  
  
...
```

* (5) End form processing

```
CALL FUNCTION 'FP_JOB_CLOSE'...
```



Calling the Generated Function Module

DATA:

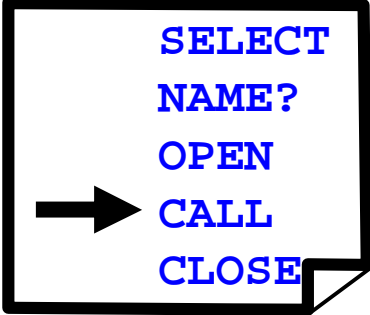
```
fp_docparams    TYPE sfpdocparams,  
fm_name         TYPE funcname,  
fp_result       TYPE TYPE fpformoutput, ...
```

```
fp_docparams-fillable = 'X'.  
fp_docparams-langu = customer_language.
```

* (4) Call function module dynamically

```
CALL FUNCTION fm_name  
  EXPORTING  
    /1bcdwb/docparams = fp_docparams  
    bookings = it_bookings  
  IMPORTING  
    /1bcdwb/formoutput = fp_result  
  EXCEPTIONS  
    OTHERS = 1.
```

```
data: pdf type fpformoutput-pdf.  
pdf = fp_result-pdf.
```



SELECT
NAME?
OPEN
CALL
CLOSE

Introduction

Creating Interactive Forms in AS ABAP

SMTP in AS ABAP

Sending Email with attached Interactive Forms

Processing Inbound Emails

Processing Interactive Forms

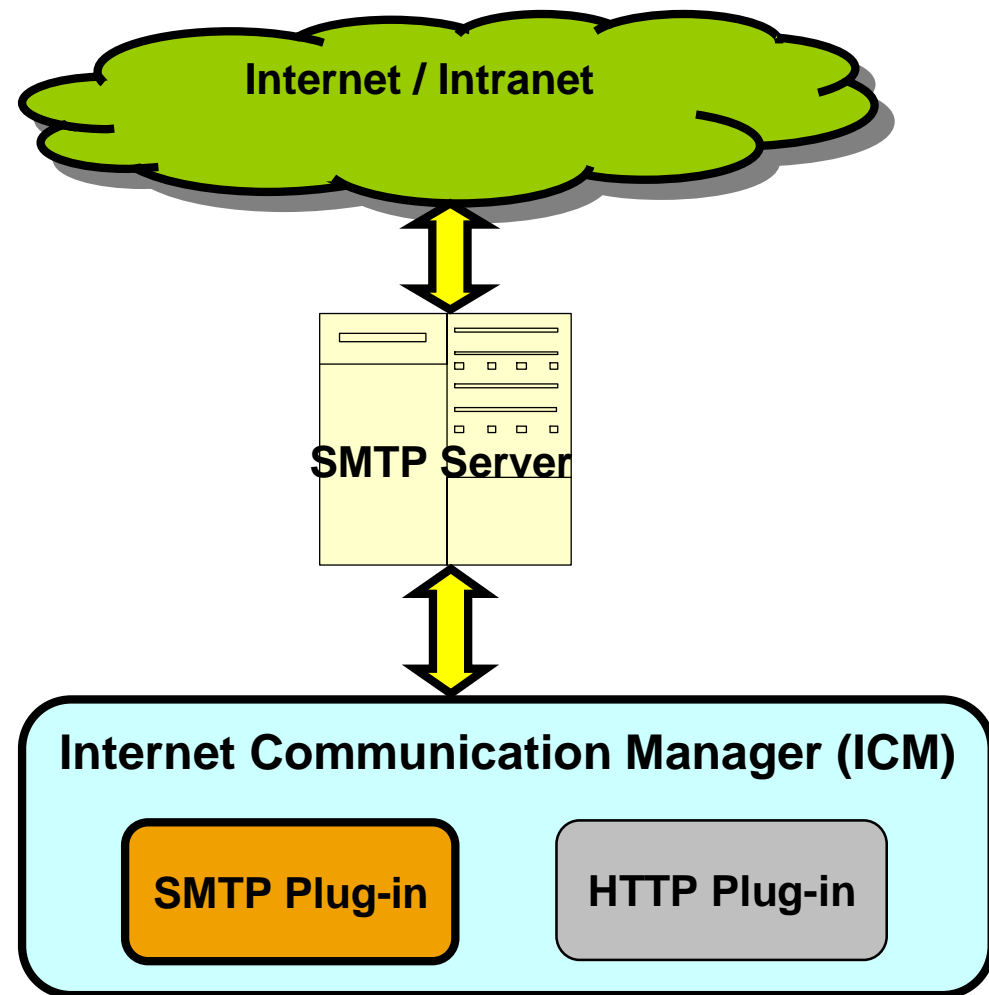
ICM handles communication protocols via Plug-Ins.

- Currently there are plug-ins for SMTP and HTTP. This can be easily extended in the future for new protocols.

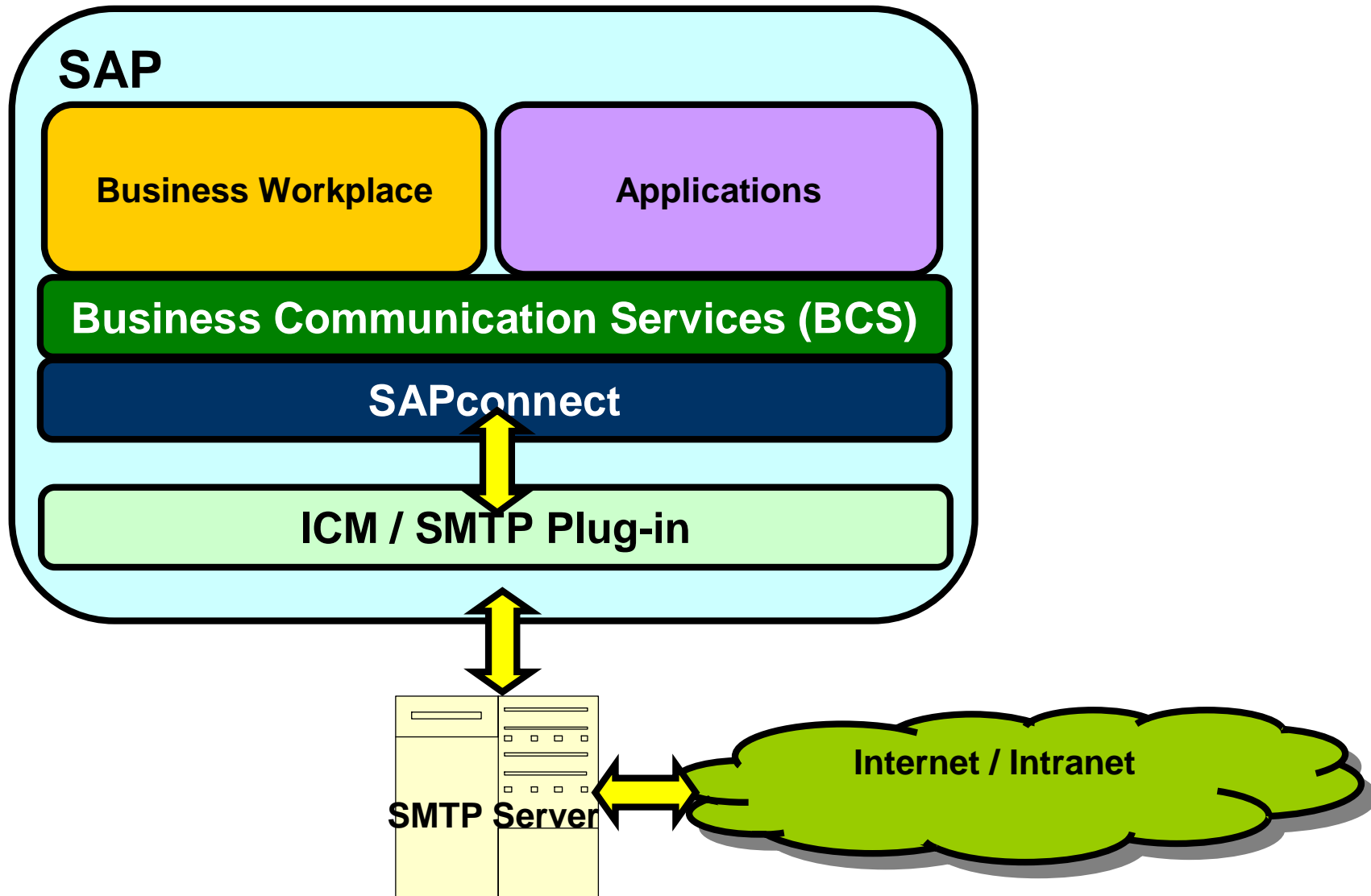
- Configure SMTP plug-in with profile parameters:

```
icm/server_port_<*> =  
PROT=SMTP,PORT=<port>
```

```
is/SMTP/virt_host_0 = *.*;
```



SAPconnect – Connecting SAP to the Outside World



SAPconnect – Configuring SMTP

To configure SAPconnect for receiving and sending emails, you must have a 3rd Party SMTP server already installed (example: Microsoft Exchange).

■ Transaction SCOT is where you can configure and monitor SAPconnect's SMTP connection to the SMTP server.

The screenshot displays the SAPconnect configuration interface. On the left, a tree view shows the node structure for 'BWL (100)'. The nodes listed are FAX (Telefax), INT (Internet), SMTP, X.400 (X.400), RML (Remote Ma), PAG, and PRT. The SMTP node is highlighted in green. To the right, a detailed configuration window titled 'SAPconnect: General node data' is open for the SMTP node. The window is divided into several sections:

- General information:** Node: SMTP, Description: Mail Server, Maximum waiting time for repeat send attempt procedure: Hours/minutes: / 30, Node in use.
- SMTP Connection:** Mail Host: cdph1819.phl.sap.corp, Mail Port: 25, Code Page: 0 No Conversion into Other Character Set.
- Supported address types:** Fax (Set), Internet (Set), Pager (SMS) (Set).
- Last changed by:** GEBO on 2006-06-22.

At the bottom of the window, there are icons for a checkmark, a help icon, and a close icon.

Business Communication Services are tools for interacting with email and workflow.

- **The Business Workplace can be accessed to view / send email and workflow items.**

- **More importantly for this class, there is a number of BCS classes that make it easy for use to Send email, and process received emails.**

- **These classes include CL_BCS and CL_DOCUMENT_BCS which are heavily used to send and receive email.**

Introduction

Creating Interactive Forms in AS ABAP

SMTP in AS ABAP

Sending Email with attached Interactive Forms

Processing Inbound Emails

Processing Interactive Forms

Business Communication Services Classes

The **Business Communication Services** includes classes that can be used to create and send emails with attachments.

Classes that will be used:

CL_BCS – class that actually send the email.

CL_DOCUMENT_BCS – class that represents the email document, and it's attachments.

CL_CAM_ADDRESS_BCS – used to create an email internet address of type **IF_RECIPIENT_BCS**.

CL_SAPUSER_BCS – class used to get the email address of an SAP user.

- **Note: All SAP users that will be used to send and receive email via SMTP must be given an email address.**

Mobile Phone		
Fax		Extension
E-Mail	travel.request@pfd.sap.corp	
Comm. Meth		

Programming Steps for Sending Email with Attachment

- * (1) Create email subject and body
- ...
- * (2) Create the email document object
- ...
- * (3) Add Interactive Form/PDF attachment.
- ...
- * (4) Create CL_BCS object used to send the email
- ...
- * (5) Set the sender and recipient on the CL_BCS object
- ...
- * (6) Send the email
- ...

* (1) Create email subject and body

* (1) Create email subject and body

```
data: l_subject type so_obj_des,  
      lt_bodytext type BCSY_TEXT,  
      l_bodytext_row type soli.
```

```
l_subject = 'Trip Creation Form'.
```

```
concatenate 'Please fill out the form and return it to:'  
           'travel.request@pdf.sap.corp'  
           into l_bodytext_row.
```

```
append l_bodytext_row to lt_bodytext.
```


* (2) Create the email document object

* (2) Create the email document object

```
data: document TYPE REF TO cl_document_bcs,  
      num_rows type i,  
      textlength type so_obj_len.
```

```
describe table l_mailtext lines num_rows.
```

```
num_rows = num_rows * 255.
```

```
move num_rows to textlength.
```

```
document = cl_document_bcs=>create_document(  
            i_type      = 'RAW'  
            i_text      = l_mailtext  
            i_length    = textlength  
            i_subject   = l_subject ).
```

* (3) Add Interactive Form/PDF attachment.

* Add attachment

```
data: attdoctype type soodk-obj tp,  
      atttitle   type S00D-0BJDES,  
      attsize    type S00D-0BJLEN,  
      pdftab     type SOLIX_TAB.
```

```
attdoctype = 'pdf' .
```

```
atttitle = 'CreateFlight' .
```

```
attsize = strlen( pdf ).
```

```
pdftab = cl_document_bcs=>xstring_to_solix(  
          ip_xstring = pdf ).
```

```
document->add_attachment( exporting I_ATTACHMENT_TYPE = attdoctype  
                           I_ATTACHMENT_SUBJECT = atttitle  
                           I_ATTACHMENT_SIZE = attsize  
                           I_ATTACHMENT_LANGUAGE = sy-langu  
                           I_ATT_CONTENT_HEX = pdftab ).
```

* (4) Create CL_BCS object used to send the email

* Create persistent send request

```
data: send_request TYPE REF TO cl_bcs.
```

```
send_request = cl_bcs=>create_persistent( ).
```

* Add document to send request

```
send_request->set_document( document ).
```

* (5) Set the sender and recipient on the CL_BCS object

* Get sender object

```
data: sender TYPE REF TO cl_sapuser_bcs.  
sender = cl_sapuser_bcs=>create( 'TRAVELREQ' ).
```

* Add sender

```
CALL METHOD send_request->set_sender  
EXPORTING i_sender = sender.
```

* Create recipient.

```
data: recipient TYPE REF TO if_recipient_bcs.  
recipient = cl_cam_address_bcs=>create_internet_address(  
p_addr ).
```

* Add recipient with its respective attributes to send request

```
send_request->add_recipient( exporting i_recipient = recipient ).
```

* (6) Send the email

* Set send immediately

```
send_request->set_send_immediately( 'X' );
```

* Send document

```
send_request->send( );
```

COMMIT WORK.

Catch the BCS Exceptions

The BCS classes throw exceptions, make sure place the BCS class interaction in a try/catch block:

* Create and send the email.

```
data: bcs_exception type ref to cx_bcs.
```

```
try.
```

* Enter your email code here!

```
catch cx_bcs into bcs_exception.
```

```
data: ex_msg type string.
```

```
ex_msg = bcs_exception->get_text( ).
```

```
write: 'Caught exception.', ex_msg.
```

```
endtry.
```

Introduction

Creating Interactive Forms in AS ABAP

SMTP in AS ABAP

Sending Email with attached Interactive Forms

Processing Inbound Emails

Processing Interactive Forms

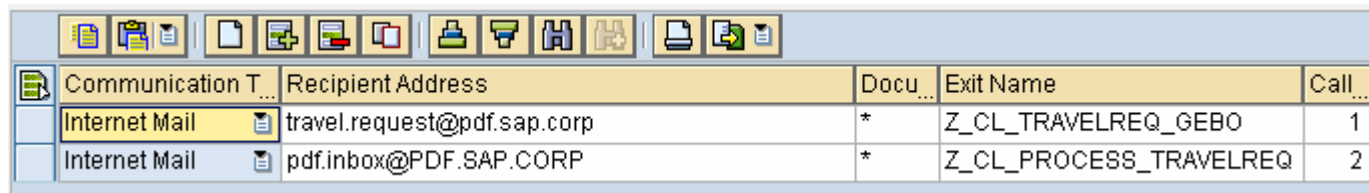
SAPconnect allows the configuration of Inbound Processing rules.

■ **If a rule is true for an incoming email, an Inbound Exit class can be executed to process the email and any attachments it may have.**

■ **Rules are made up of communication type, recipient address, and document type.**

■ **Multiple rules can be configured, each rule must be given a call sequence (a number). Inbound emails are checked against the lowest call sequence first (1, then 2, etc...).**

■ **To create/modify/view rules, go to transaction SCOT, then menu path Settings > Inbound Processing.**



The screenshot shows the SAP SCOT transaction interface. At the top, there is a toolbar with various icons for file operations. Below the toolbar is a table with the following columns: Communication T..., Recipient Address, Docu..., Exit Name, and Call... The table contains two rows of data:

Communication T...	Recipient Address	Docu...	Exit Name	Call...
Internet Mail	travel.request@pdf.sap.corp	*	Z_CL_TRAVELREQ_GEBO	1
Internet Mail	pdf.inbox@PDF.SAP.CORP	*	Z_CL_PROCESS_TRAVELREQ	2

Inbound Exit classes are singleton classes that must implement the IF_INBOUND_EXIT_BCS interface.

- **Must implement the two methods defined on this interface.**
 - **CREATE_INSTANCE** – contains code to create the singleton instance.
 - ◆ **Must set the returning parameter RO_REF to the instance of the class.**
 - **PROCESS_INBOUND** – contains code to process the inbound email.
 - ◆ **Import parameter: IO_SREQ type ref to CL_SEND_REQUEST_BCS**
 - Contains inbound email document class CL_DOCUMENT_BCS
 - Can be used to create reply emails.
 - ◆ **Import parameter: IT_RECIPIENTS type BCSY_SMTPA – list of SMTP recipients.**
 - ◆ **Import parameter: IT_DOCTYPES type BCSY_SODOC – list of documents types make up the email.**
 - ◆ **Exporting parameter: E_RETCODE – used to control processing of other Inbound Exit processes in the call sequence.**
 - ◆ **Exporting parameter: ES_T100MSG – used to store T100 messages that can be written to a log.**

CREATE_INSTANCE Implementation

The CREATE_INSTANCE method is used to get a reference to the singleton instance of the class – if it does not exist it should create it:

- * Check if the singleton instance has already been created.

```
IF instance IS INITIAL.  
    CREATE OBJECT instance.  
ENDIF.
```

- * Return the instance.
ro_ref = instance.

PROCESS_INBOUND Implementation

The PROCESS_INBOUND method is executed for every inbound email. You must place code here to do the email processing that is desired:

- * Get the email document that was sent.

data: document type ref to if_document_bcs.

```
document = io_sreq->get_document( ).
```

- * Get the interactive form attachment.

data: pdf_table type BCSS_DBPC.

```
pdf_table = document->get_body_part_content( 2 ).
```

Converting attachment type BCSS_DBPC to xstring

The attachment is returned from the CL_DOCUMENT_BCS object as a table of type BCSS_DBPC – to process an Interactive Form attachment it must be converted into an xstring type:

* Convert the pdf table into an xstring.

```
data: pdf_xstring type xstring,
```

```
      pdf_line type solix.
```

```
clear pdf_xstring.
```

```
loop at pdf_table-cont_hex into pdf_line.
```

```
  concatenate pdf_xstring pdf_line-line into pdf_xstring  
  in byte mode.
```

```
endloop.
```

Introduction

Creating Interactive Forms in AS ABAP

SMTP in AS ABAP

Sending Email with attached Interactive Forms

Processing Inbound Emails

Processing Interactive Forms

The processing of Interactive Forms is made up of two parts:

- 1. Get the form data from the Interactive Form PDF document using the CL_FP_PDF_OBJECT class.**
 - **The form data is returned as an XML document.**
- 2. Parsing the XML document using the iXML classes.**
 - **Once the form data is parsed from the XML document it can be used in your business processing (call a BAPI to create a flight booking for instance).**

Retrieving Form Data from Interactive Form (1)

- * Get a reference to the form processing class.

data: l_fp type ref to if_fp.

```
l_fp = cl_fp=>get_reference( ).
```

- * Get a reference to the PDF Object class.

data: l_pdfobj type ref to if_fp_pdf_object.

```
l_pdfobj = l_fp->create_pdf_object( ).
```

- * Set the pdf in the PDF Object.

```
l_pdfobj ->set_document( pdfdata = pdf_xstring ).
```

- * Set the PDF Object to extract data the Form data.

```
l_pdfobj ->set_extractdata( ).
```

- * Execute call to ADS

```
l_pdfobj ->execute( ).
```

Retrieving Form Data from Interactive Form (2)

- * Get the PDF Form data.

data: pdf_form_data type xstring.

```
l_pdfobj ->get_data( importing formdata =  
    pdf_form_data ).
```

- * Convert the xstring form data to string so it can be processed using the iXML classes.

data: converter type ref to cl_abap_conv_in_ce.

```
converter = CL_ABAP_CONV_IN_CE=>CREATE( input =  
    pdf_form_data ).
```

```
converter->READ( importing data = formxml ).
```


Parsing the XML Document Using the iXML classes(1)

- * Pull in the iXML type group.

```
type-pools: ixml.
```

- * Get a reference to iXML object.

```
data: l_ixml type ref to IF_IXML.
```

```
l_ixml = cl_ixml =>create( ).
```

- * Get iStream object from StreamFactory

```
data: streamFactory type ref to if_ixml_stream_factory.
```

```
data: iStream type ref to if_ixml_iStream.
```

```
streamFactory = l_ixml ->create_stream_factory( ).
```

```
iStream =
```

```
    streamFactory->create_iStream_string( formxml ).
```

Parsing the XML Document Using the iXML classes(2)

- * Create an XML Document class that will be used to process the XML

data: document type ref to if_ixml_document.

```
document = I_ixml ->create_document( ).
```

- * Create the Parser class

data: parser type ref to if_ixml_parser.

```
parser = I_ixml ->create_parser( stream_factory = streamFactory  
                                istream         = iStream  
                                document        = document ).
```

- * Parse the XML

```
parser->parse( ).
```

Parsing the XML Document Using the iXML classes(3)

- * Define XML Node type object

```
data: node type ref to if_ixml_node.
```

- * Get the RESERVED Data Node and value.

```
data: strChecked type string.
```

```
node = document->find_from_name( name = 'RESERVED' ).
```

```
strChecked = node->get_value( ).
```

- * Look up the rest of the nodes and values.

```
data: custbook type BAPI_SBONEW.
```

```
node = document->find_from_name( name = 'AIRLINEID' ).
```

```
custbook-airlineid = node->get_value( ).
```

```
...
```

- No part of this publication may be reproduced or transmitted in any form or for any purpose without the express permission of SAP AG. The information contained herein may be changed without prior notice.
 - Some software products marketed by SAP AG and its distributors contain proprietary software components of other software vendors.
 - Microsoft, Windows, 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, z/OS, AFP, Intelligent Miner, WebSphere, Netfinity, Tivoli, and Informix are trademarks or registered trademarks of IBM Corporation 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.

- Weitergabe und Vervielfältigung dieser Publikation oder von Teilen daraus sind, zu welchem Zweck und in welcher Form auch immer, ohne die ausdrückliche schriftliche Genehmigung durch SAP AG nicht gestattet. In dieser Publikation enthaltene Informationen können ohne vorherige Ankündigung geändert werden.
 - Die von SAP AG oder deren Vertriebsfirmen angebotenen Softwareprodukte können Softwarekomponenten auch anderer Softwarehersteller enthalten.
 - Microsoft, Windows, Outlook, und PowerPoint sind eingetragene Marken der 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, z/OS, AFP, Intelligent Miner, WebSphere, Netfinity, Tivoli, und Informix sind Marken oder eingetragene Marken der IBM Corporation in den USA und/oder anderen Ländern.
 - Oracle ist eine eingetragene Marke der Oracle Corporation.
 - UNIX, X/Open, OSF/1, und Motif sind eingetragene Marken der Open Group.
 - Citrix, ICA, Program Neighborhood, MetaFrame, WinFrame, VideoFrame, und MultiWin sind Marken oder eingetragene Marken von Citrix Systems, Inc.
 - HTML, XML, XHTML und W3C sind Marken oder eingetragene Marken des W3C®, World Wide Web Consortium, Massachusetts Institute of Technology.
 - Java ist eine eingetragene Marke von Sun Microsystems, Inc.
 - JavaScript ist eine eingetragene Marke der Sun Microsystems, Inc., verwendet unter der Lizenz der von Netscape entwickelten und implementierten Technologie.
 - MaxDB ist eine Marke von MySQL AB, Schweden.
 - SAP, R/3, mySAP, mySAP.com, xApps, xApp, SAP NetWeaver und weitere im Text erwähnte SAP-Produkte und -Dienstleistungen sowie die entsprechenden Logos sind Marken oder eingetragene Marken der SAP AG in Deutschland und anderen Ländern weltweit. Alle anderen Namen von Produkten und Dienstleistungen sind Marken der jeweiligen Firmen. Die Angaben im Text sind unverbindlich und dienen lediglich zu Informationszwecken. Produkte können länderspezifische Unterschiede aufweisen.
-
- Die in dieser Publikation enthaltene Information ist Eigentum der SAP. Weitergabe und Vervielfältigung dieser Publikation oder von Teilen daraus sind, zu welchem Zweck und in welcher Form auch immer, nur mit ausdrücklicher schriftlicher Genehmigung durch SAP AG gestattet.
 - Bei dieser Publikation handelt es sich um eine vorläufige Version, die nicht Ihrem gültigen Lizenzvertrag oder anderen Vereinbarungen mit SAP unterliegt. Diese Publikation enthält nur vorgesehene Strategien, Entwicklungen und Funktionen des SAP®-Produkts. SAP entsteht aus dieser Publikation keine Verpflichtung zu einer bestimmten Geschäfts- oder Produktstrategie und/oder bestimmten Entwicklungen. Diese Publikation kann von SAP jederzeit ohne vorherige Ankündigung geändert werden.
 - SAP übernimmt keine Haftung für Fehler oder Auslassungen in dieser Publikation. Des Weiteren übernimmt SAP keine Garantie für die Exaktheit oder Vollständigkeit der Informationen, Texte, Grafiken, Links und sonstigen in dieser Publikation enthaltenen Elementen. Diese Publikation wird ohne jegliche Gewähr, weder ausdrücklich noch stillschweigend, bereitgestellt. Dies gilt u. a., aber nicht ausschließlich, hinsichtlich der Gewährleistung der Marktgängigkeit und der Eignung für einen bestimmten Zweck sowie für die Gewährleistung der Nichtverletzung geltenden Rechts.
 - SAP haftet nicht für entstandene Schäden. Dies gilt u. a. und uneingeschränkt für konkrete, besondere und mittelbare Schäden oder Folgeschäden, die aus der Nutzung dieser Materialien entstehen können. Diese Einschränkung gilt nicht bei Vorsatz oder grober Fahrlässigkeit.
 - Die gesetzliche Haftung bei Personenschäden oder Produkthaftung bleibt unberührt. Die Informationen, auf die Sie möglicherweise über die in diesem Material enthaltenen Hotlinks zugreifen, unterliegen nicht dem Einfluss von SAP, und SAP unterstützt nicht die Nutzung von Internetseiten Dritter durch Sie und gibt keinerlei Gewährleistungen oder Zusagen über Internetseiten Dritter ab.

Colors

Remove page for slide show



R	0	239	77	191
G	51	159	77	191
B	102	0	77	191



R	124	87	190	80	33
G	30	129	102	34	87
B	31	174	0	71	71

Links to SAP PowerPoint Samples & Guidelines

Remove page for slide show

In order to access the links below, please view this slide in Slide Show mode

[SAP PPT Guidelines](#)

Other than Title Slides and Divider Pages - The SAP PPT Guidelines include theory and examples of:

- Colors, Fonts & Bullets
- Image Usage & Quotes
- Graphics, Charts, Tables & Processes

[SAP PPT Samples](#)

SAP PPT Sample pages include live ppt slides that were discussed in the SAP PPT Guidelines. Users can select from a library of slides including title slides, divider pages, bulleted slides, charts and graphs

For questions regarding the information and materials contained in these guidelines, please email branding@SAP.com