BAPI

Business Application Programming Interface

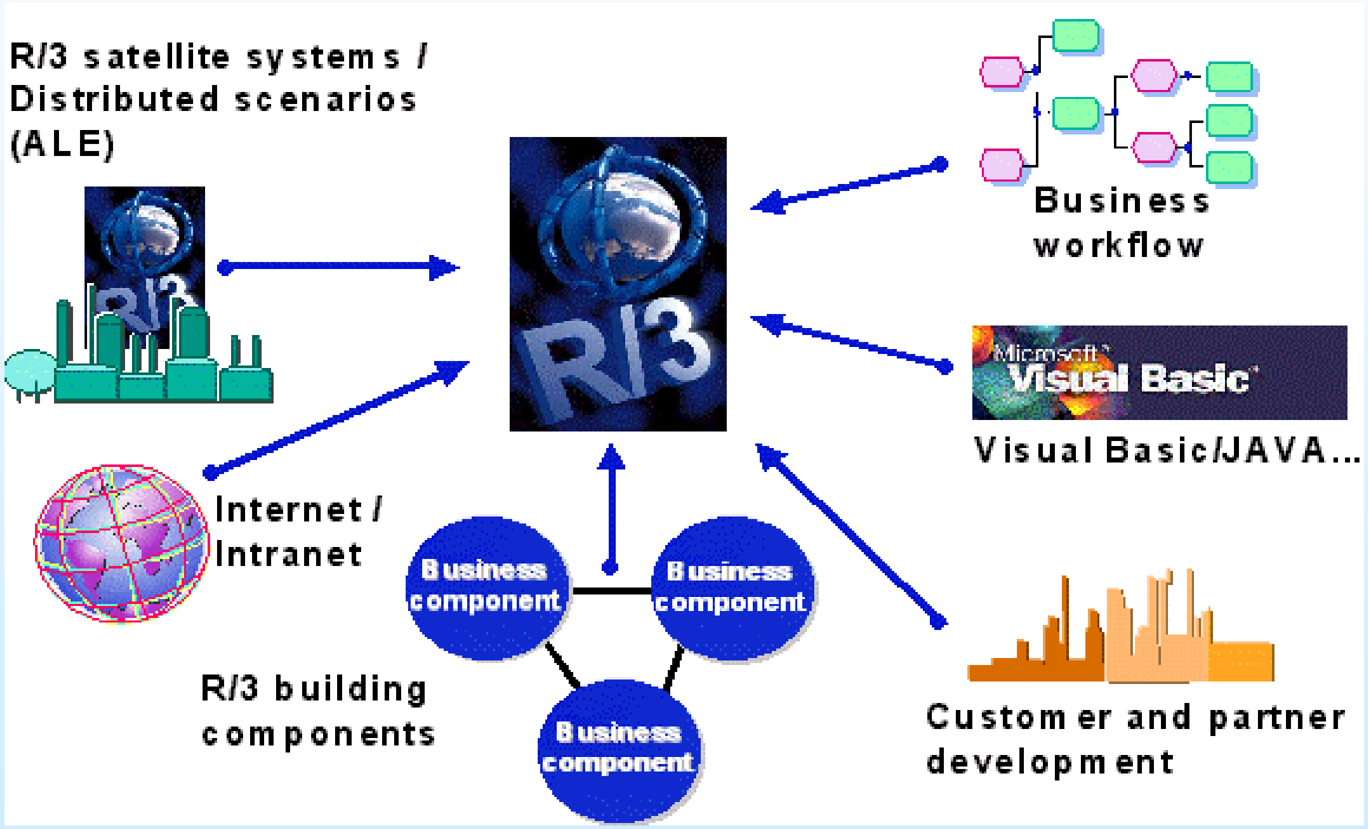
What is BAPI

A Business Application Programming Interface is a precisely defined interface providing access process and data in Business Applications Systems Such as SAP R/3

Benefits of BAPI

- Can be used in diverse languages / Development Environments (ABAP, Visual Basic, Java, C++, etc.)
- Can be called from diverse platforms (COM, CORBA, Unix)
- Reduced development cost
- Reduced maintenance cost
- “Best-of-both-worlds” approach
  - Rich functionality of the R/3 system
  - User-specific front-ends

Where BAPIs can be used



Return Code Information

- Usually a structure, sometimes a table
- Data dictionary structures used
  - BAPIRETURN
  - BAPIRETURN1
  - BAPIRET1
  - BAPIRET2

BAPI Return Structure

- Type Message type
  - blank or "S"=Success
  - "E"=Error
  - "W"=Warning
  - "I"=Information
  - "A"=Abort

Message	Message text
Log_No	Application Log Number
Log_Msg_No	Application Log Message Serial Number
Message_V1 - V4	Message variables

SAP transactions

- BAPI Business Object Browser (BAPIs only)
- SWO1 Business Object Builder (all objects)
- SWO2 Business Object Browser (all objects)
- SE11 Data Dictionary
- SE37 Function Builder

JCO Overview

- **High-performance JNI-based middleware**
- **Support R/3 3.1H and higher.**
- **Supports inbound and outbound calls.**
- **Supports client pooling.**
- **Supports desktop and web/application server applications.**
- **Multi-platform**
- **Complete and correct code page handling**
- **Easy to install and deploy**

Installation and Deployment

- Required files in \WINNT\system32:
  - librfc32.dll (at least 46D, build 263)
  - jRFC11.dll (JDK 1.1)
  - jRFC12.dll (JDK 1.2 and 1.3)
- Required files in Java class path:
  - jCO.jar

BAPI step by step procedure

STEP 1 - Define Structure For The BAPI

STEP 2 - Write Function Module

STEP 3 - Create the API Method Using The BAPI WIZARD

STEP 4 – Final Steps

About the Example

About the Example:

Front End : Java Servlets (Web Application)

Web Server : Apache Tomcat

The Servlet takes Vendor number and passes it to the BAPI which in turn fetches the Vendor information from the LFA1 table and returns it in BAPIRET2 format to the servlet, the servlet fetches the data from return structure and displays it.

Step 1 : Define a Structure for BAPI

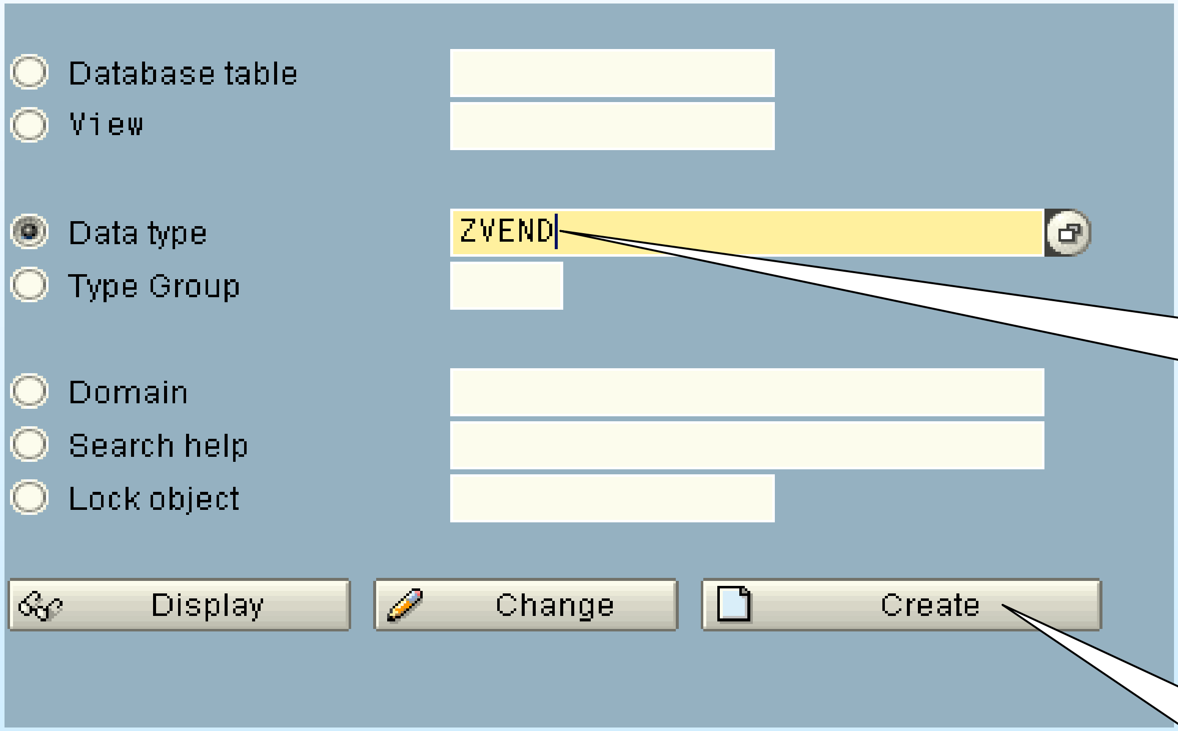
In this step structures for the parameters and tables of the function module used for the BAPI are defined.

USE TCODE : **SE11** then *Data type -> Structure*

Define the structure Name : Ex: **ZVEND**

Important note: You will have to define a structure for <u>every</u> parameter in the BAPI. You cannot use the same structures used in existing applications because BAPI structures are frozen when BAPIs are released and then there are restrictions on changing them.

Creating a Structure

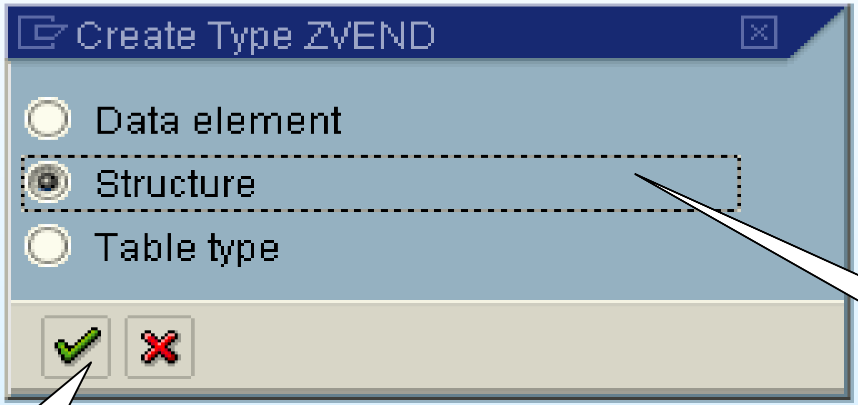


○ Database table	
○ View	
● Data type	ZVEND
○ Type Group	
○ Domain	
○ Search help	
○ Lock object	

Enter the Structure name

Click on Create Button

Creating a Structure


Create Type ZVEND

Data element

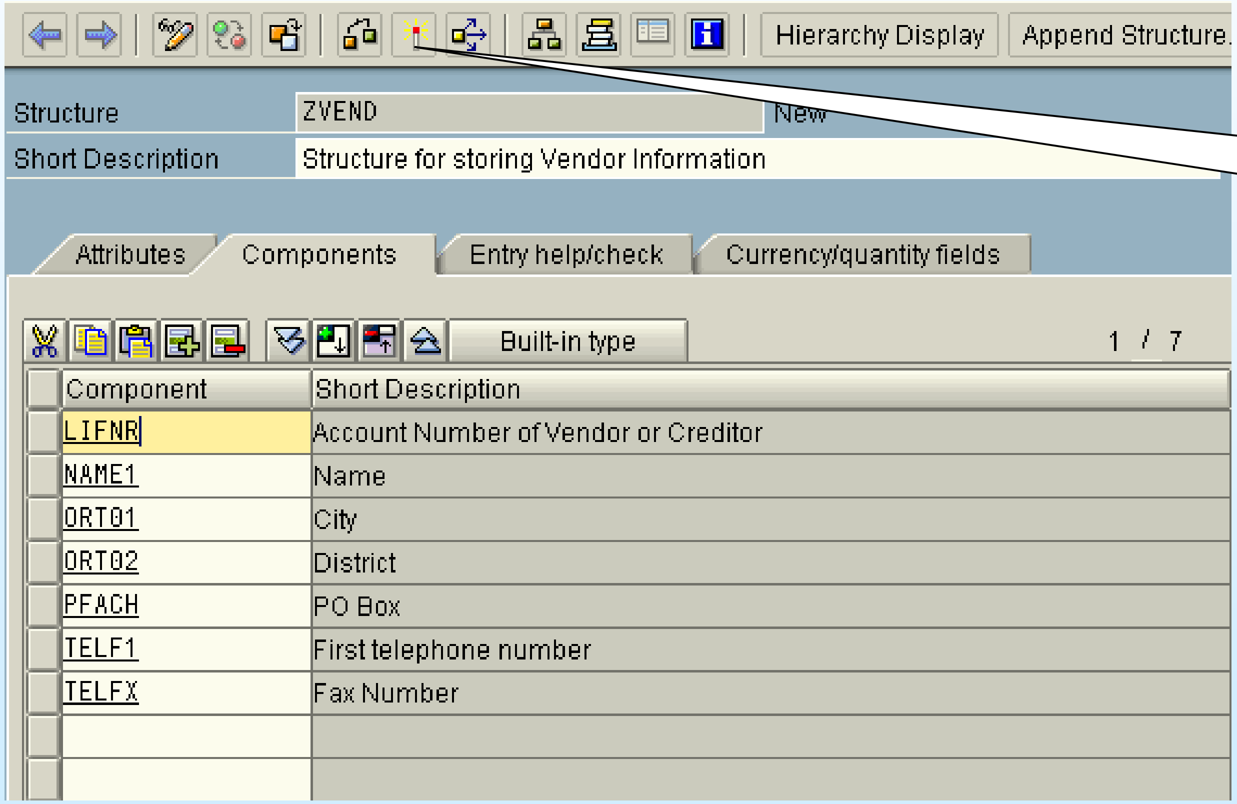
[✓ Structure]

Table type

Select Structure

Click on Check Button

Creating a Structure



Structure	ZVEND New
Short Description	Structure for storing Vendor Information

1 / 7

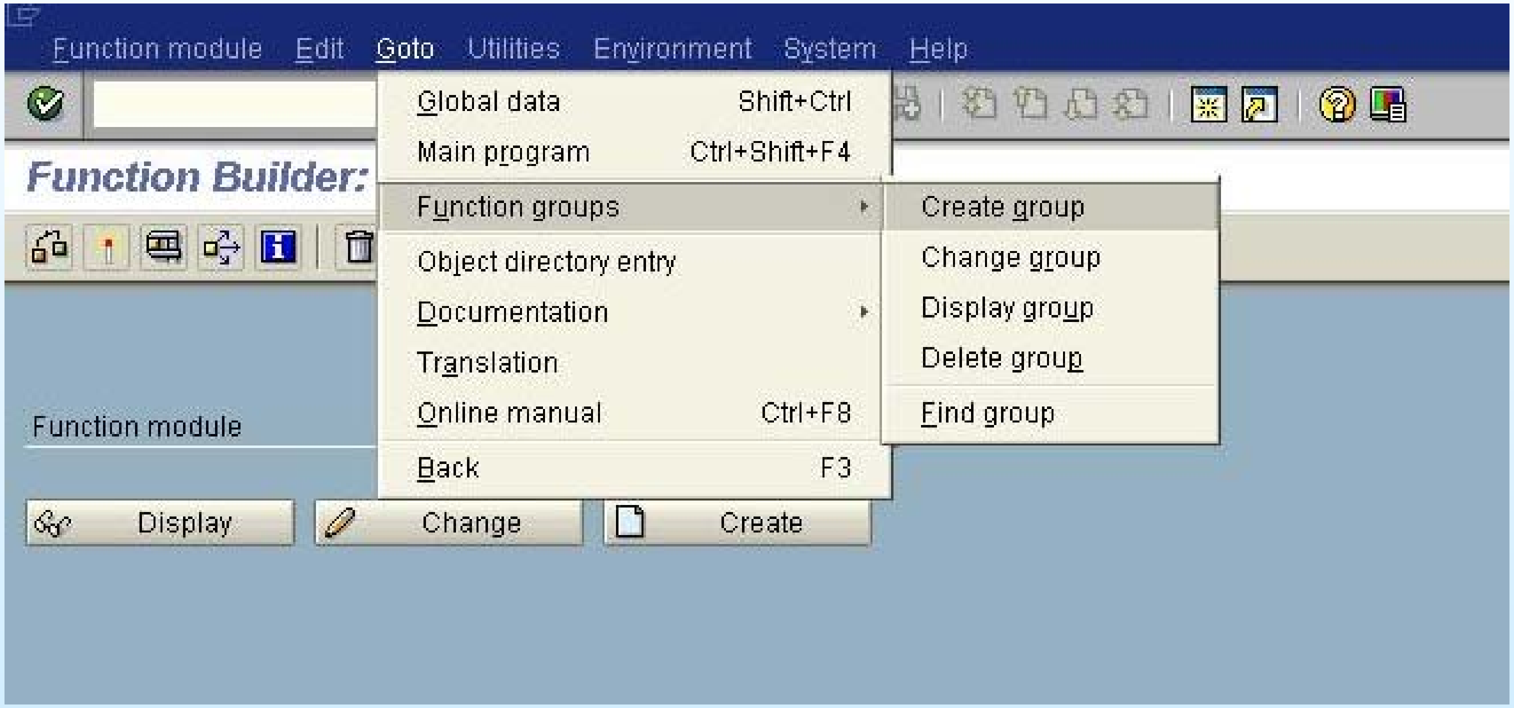
Component	Short Description
LIFNR	Account Number of Vendor or Creditor
NAME1	Name
ORT01	City
ORT02	District
PFACH	PO Box
TELF1	First telephone number
TELFX	Fax Number

Activate the Structure

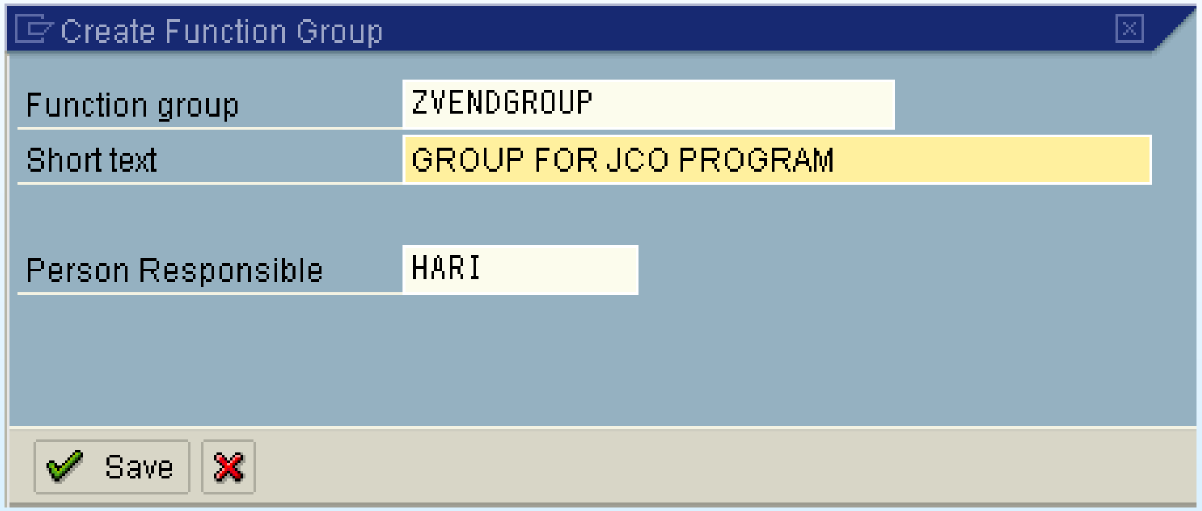
Step 2 : Write Function Module

- Each BAPI must have its own function group.
- Under the attributes tab remember to select Processing Type *Remote Enabled module*, otherwise the function module cannot be invoked via RFC and used as a BAPI
- Import/Export parameters can only be BY VALUE for an RFC enabled function module

Creating Function group



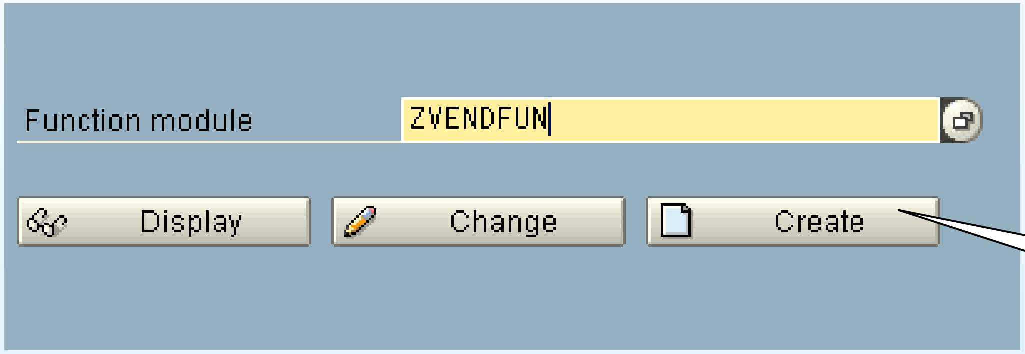
Creating Function group



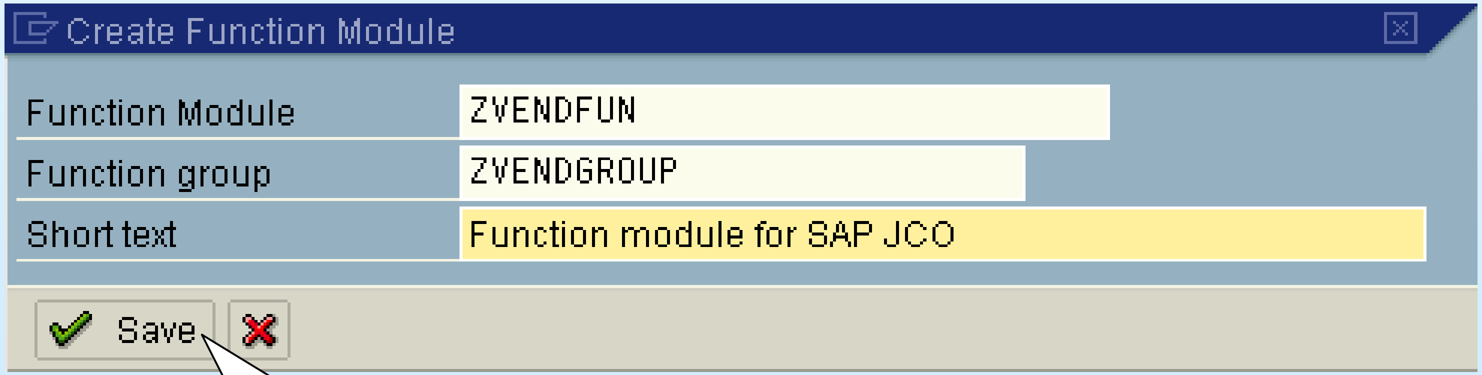
Create Function Group

Function group	ZVENDGROUP
Short text	GROUP FOR JCO PROGRAM
Person Responsible	HARI

Creating Function module


Function module: ZVENDFUN

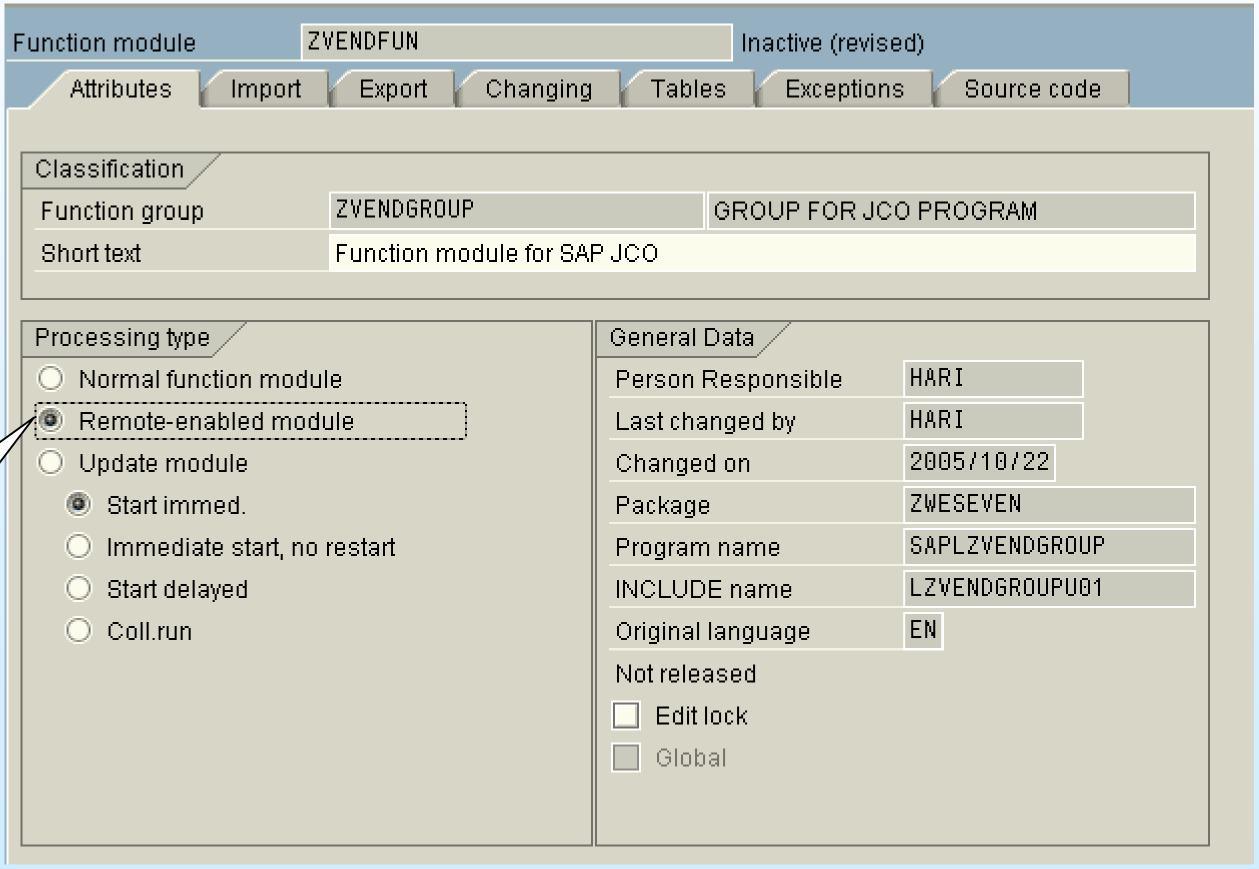
Click on Create



Function Module	ZVENDFUN
Function group	ZVENDGROUP
Short text	Function module for SAP JCO

Click on Save

Creating Function module



Function module	ZVENDFUN	Inactive (revised)

Classification

Function group	ZVENDGROUP	GROUP FOR JCO PROGRAM
Short text	Function module for SAP JCO	

Processing type

- ○ Normal function module
- ● Remote-enabled module
- ○ Update module
  - ● Start immed.
  - ○ Immediate start, no restart
  - ○ Start delayed
  - ○ Coll.run

General Data

Person Responsible	HARI
Last changed by	HARI
Changed on	2005/10/22
Package	ZWESEVEN
Program name	SAPLZVENDGROUP
INCLUDE name	LZVENDGROUPU01
Original language	EN

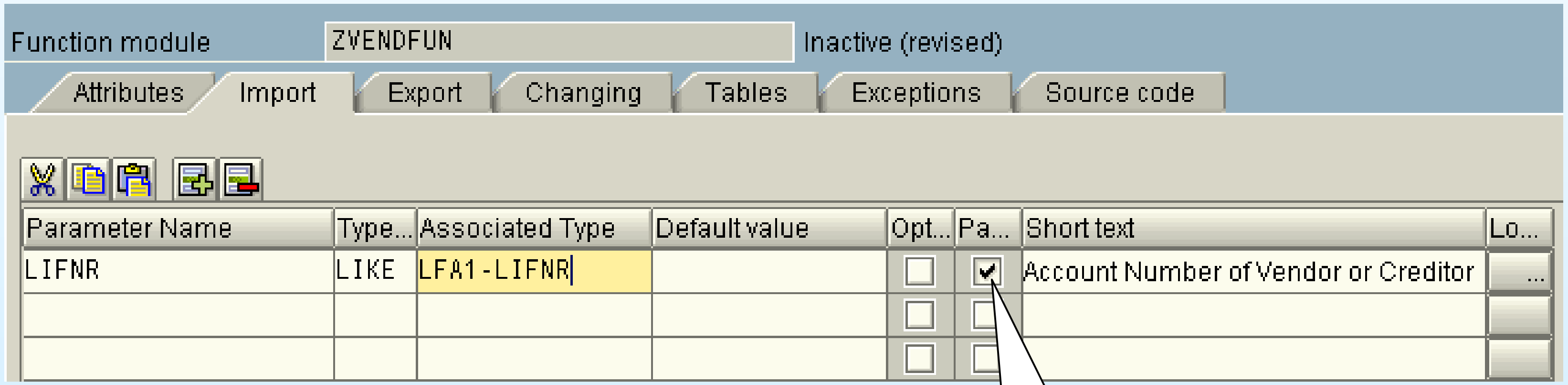
Not released

- ☐ Edit lock
- ☐ Global

Make the function Remote Enabled

Creating Function module

Import Parameters



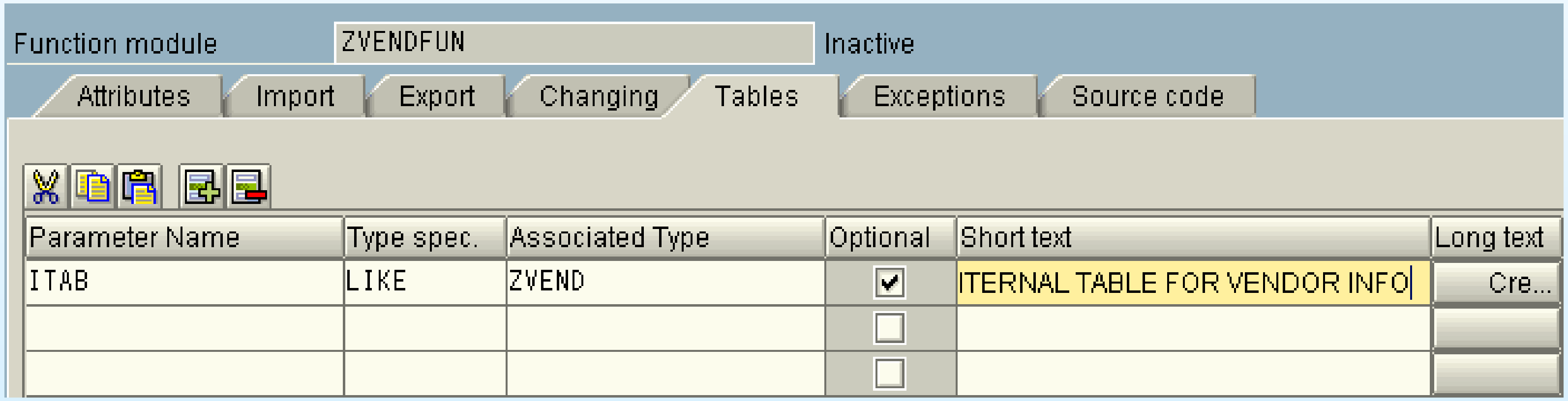
Function module ZVENDFUN Inactive (revised)

Parameter Name	Type...	Associated Type	Default value	Opt...	Pa...	Short text	Lo...
LIFNR	LIKE	LFA1-LIFNR		☐	✓	Account Number of Vendor or Creditor	...
				☐	☐		
				☐	☐		

Check
“ Pass Value”

Creating Function module

Tables

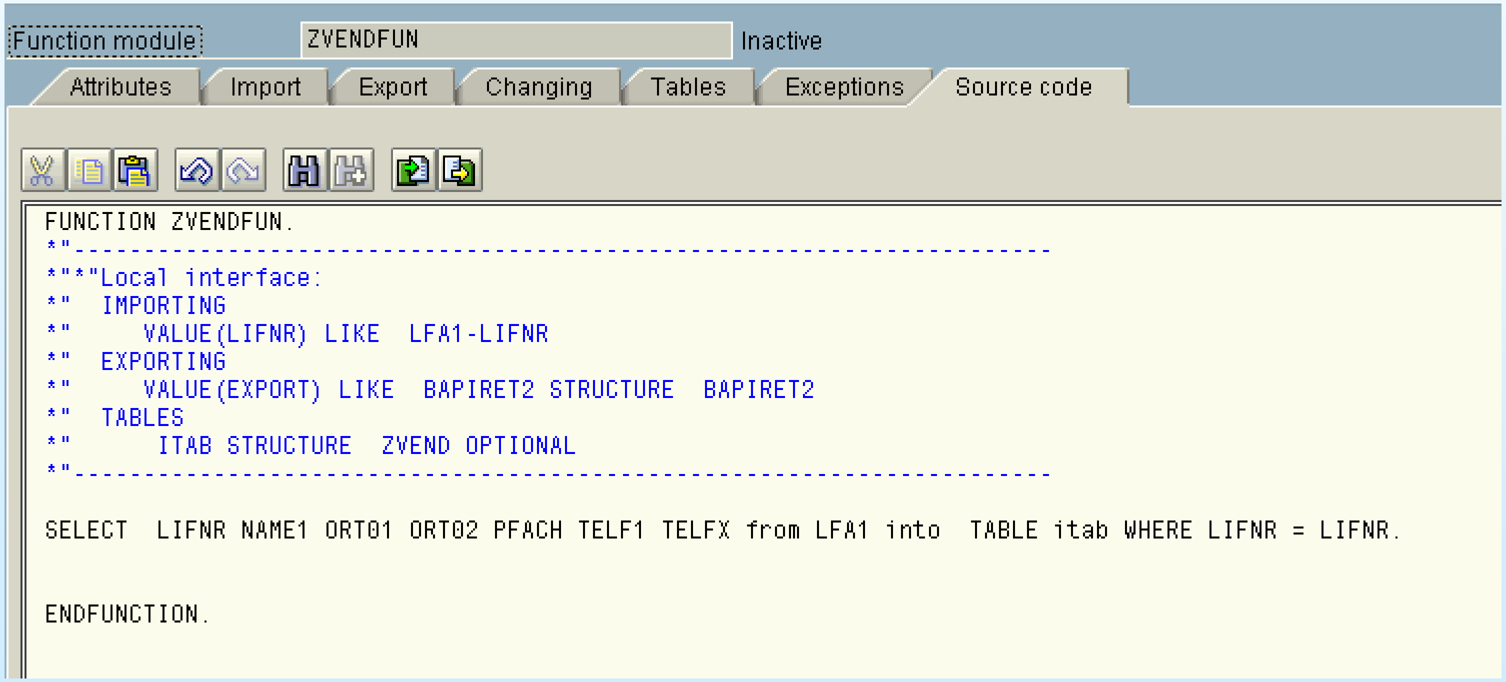


Function module ZVENDFUN Inactive

Parameter Name	Type spec.	Associated Type	Optional	Short text	Long text
ITAB	LIKE	ZVEND	✓	ITERNAL TABLE FOR VENDOR INFO	Cre...
			☐		
			☐		

Creating Function module

Source Code



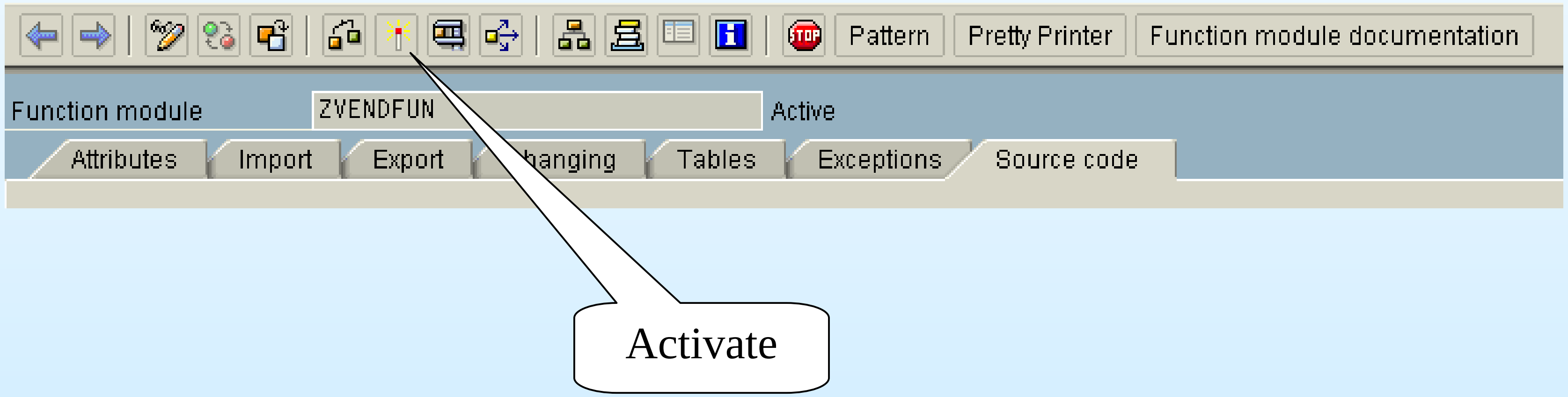
```
FUNCTION ZVENDFUN.
*"----------------------------------------------------------------------
*"*"Local interface:
*"  IMPORTING
*"     VALUE(LIFNR) LIKE  LFA1-LIFNR
*"  EXPORTING
*"     VALUE(EXPORT) LIKE  BAPIRET2 STRUCTURE  BAPIRET2
*"  TABLES
*"      ITAB STRUCTURE  ZVEND OPTIONAL
*"----------------------------------------------------------------------

SELECT  LIFNR NAME1 ORT01 ORT02 PFACH TELF1 TELFX from LFA1 into  TABLE itab WHERE LIFNR = LIFNR.


ENDFUNCTION.
```

Creating Function module

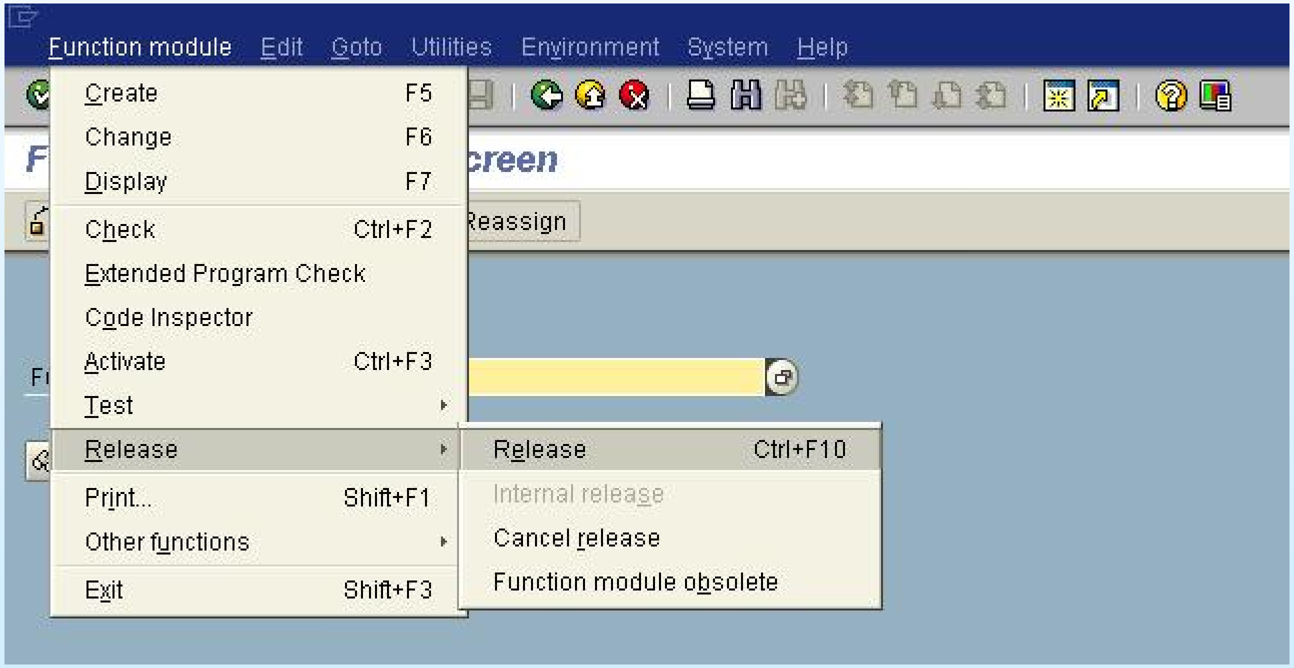
Activate Function Module



Function module ZVENDFUN Active

Releasing Function module

Release the Function Module



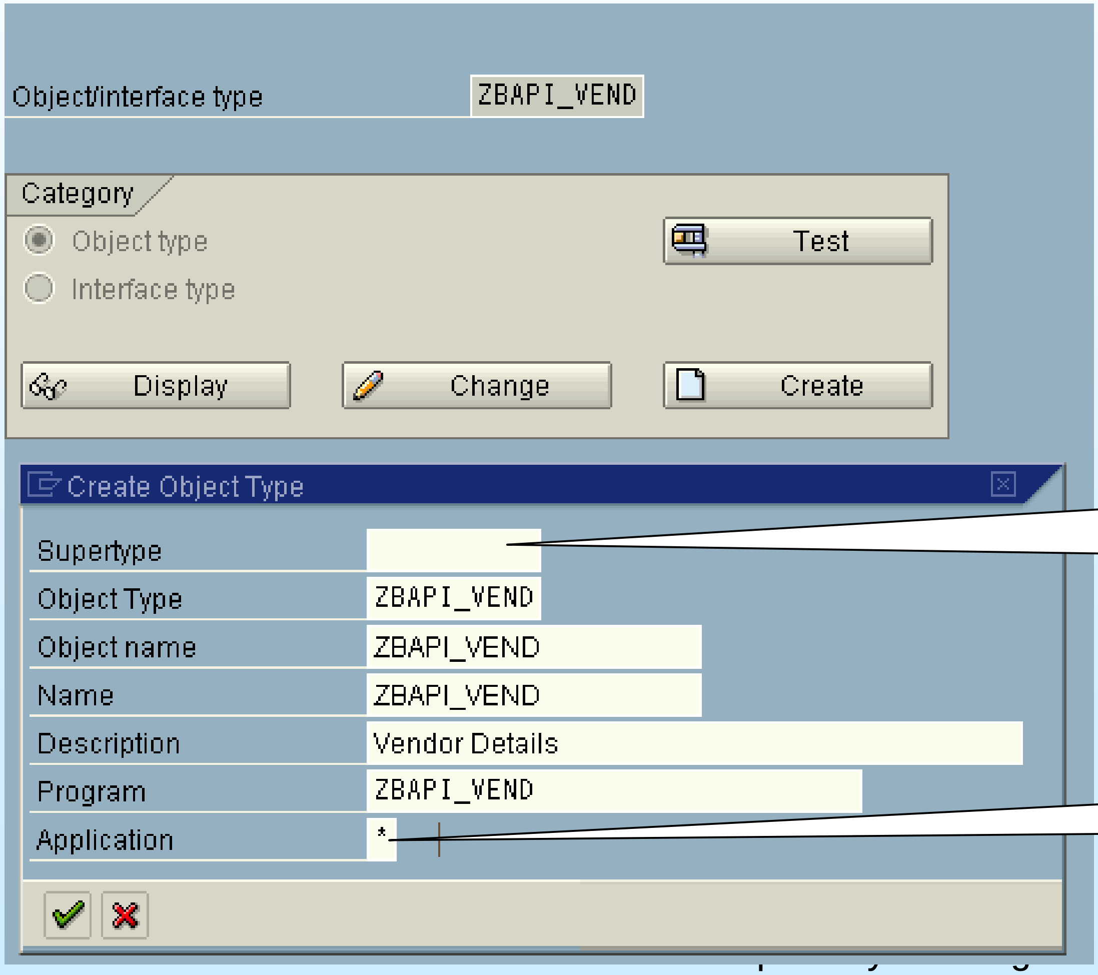
Step 2 : Create the API Method Using The BAPI WIZARD

- BAPI wizard is used to expose the remote function module as a BAPI
- Wizard will generate some additional code, so the function module is a valid method of the BOR. This allows the BAPI to be called as a workflow method in addition to be called by an outside program.
- Each function module corresponds to a method in the BOR

Go to the Business Object Builder SWO1.

You can either create the new Object type as a subtype of an existing business object or create a new business object from scratch..

Create new BAPI Object



Object/interface type ZBAPI_VEND

Category
- [✓ Object type]
- Interface type

Supertype	
Object Type	ZBAPI_VEND
Object name	ZBAPI_VEND
Name	ZBAPI_VEND
Description	Vendor Details
Program	ZBAPI_VEND
Application	*

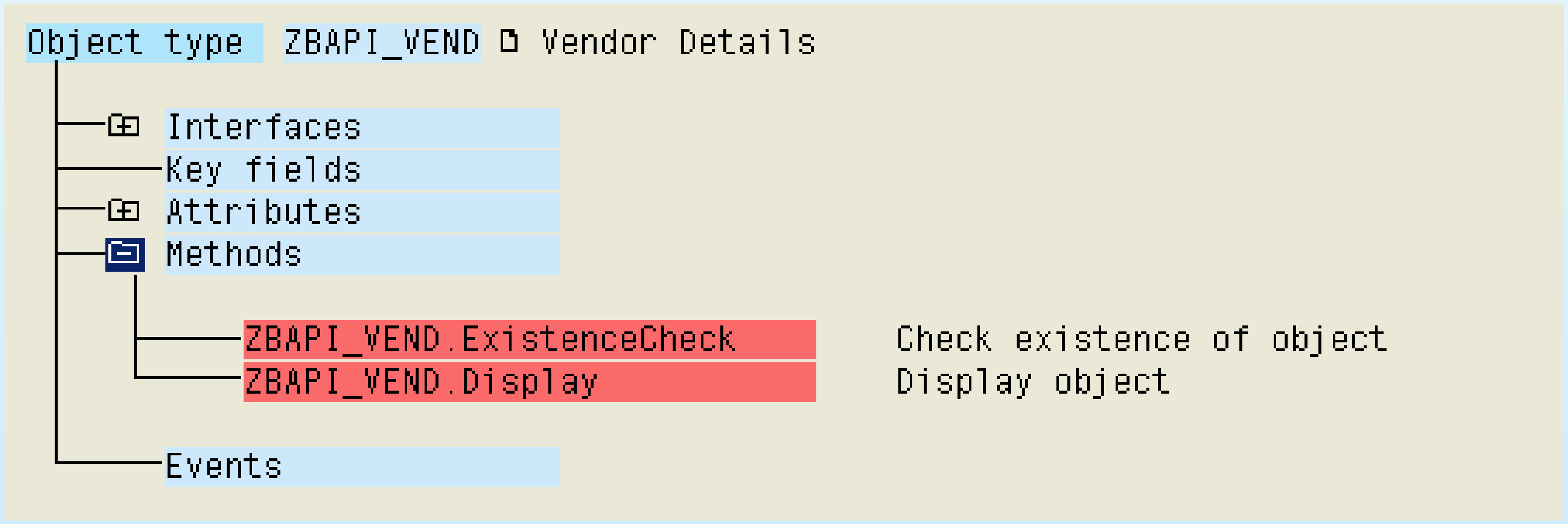
USE TCODE SWO1

Supertype not required as we are creating a new Object

* for Cross Apps

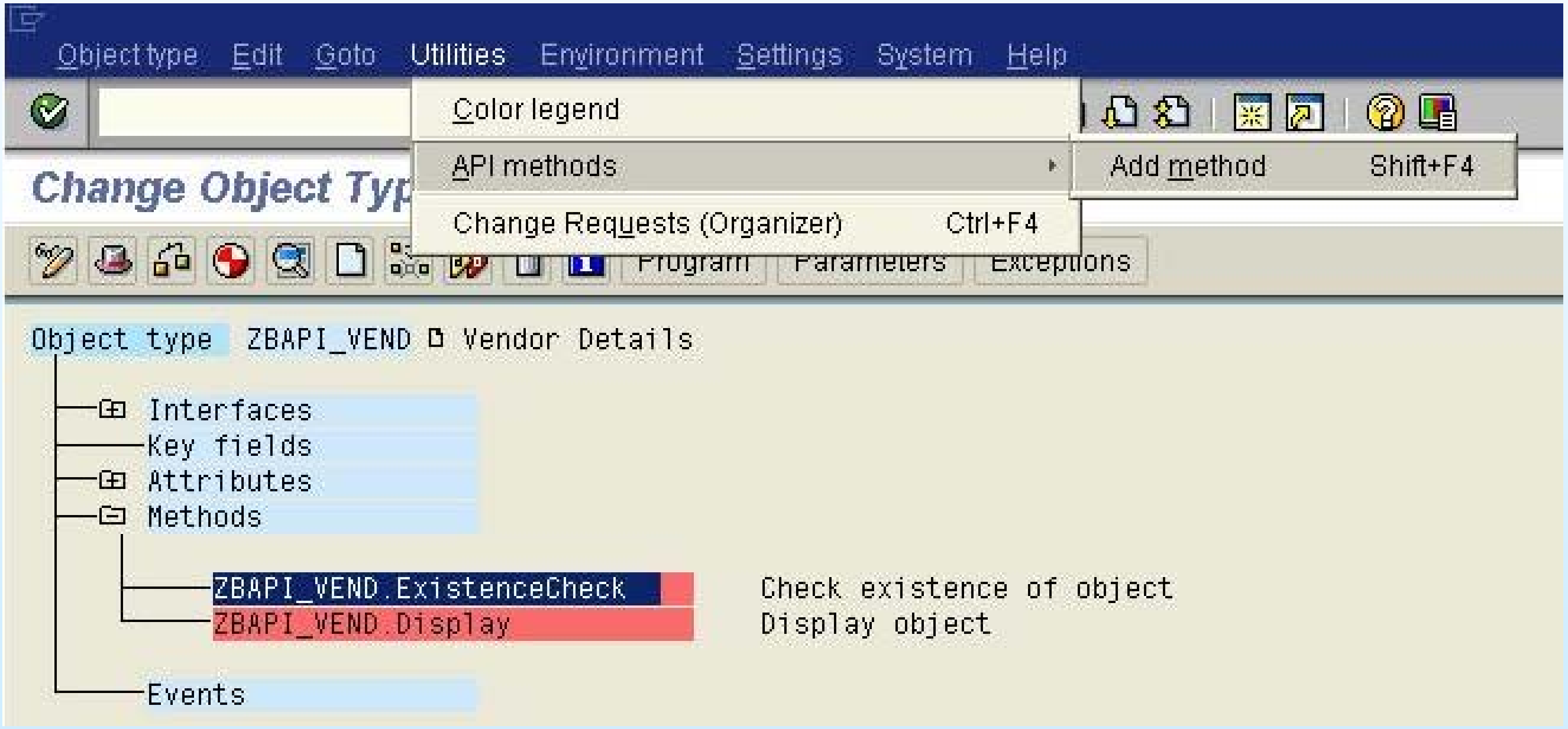
Create new BAPI Object

Note that when you create the business object a standard interface, an attribute ObjectType and the methods ExistenceCheck and Display are automatically generated. These cannot be changed !



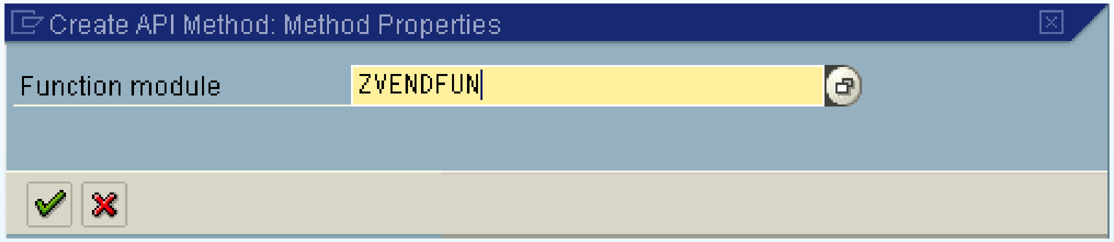
```
Object type  ZBAPI_VEND  Vendor Details
  Interfaces
  Key fields
  Attributes
  Methods
      ZBAPI_VEND.ExistenceCheck      Check existence of object
      ZBAPI_VEND.Display             Display object
  Events
```

Adding API method


Object type Edit Goto Utilities Environment Settings System Help
Color legend
API methods
Add method Shift+F4
Change Requests (Organizer) Ctrl+F4
Change Object Typ
Program Parameters Exceptions

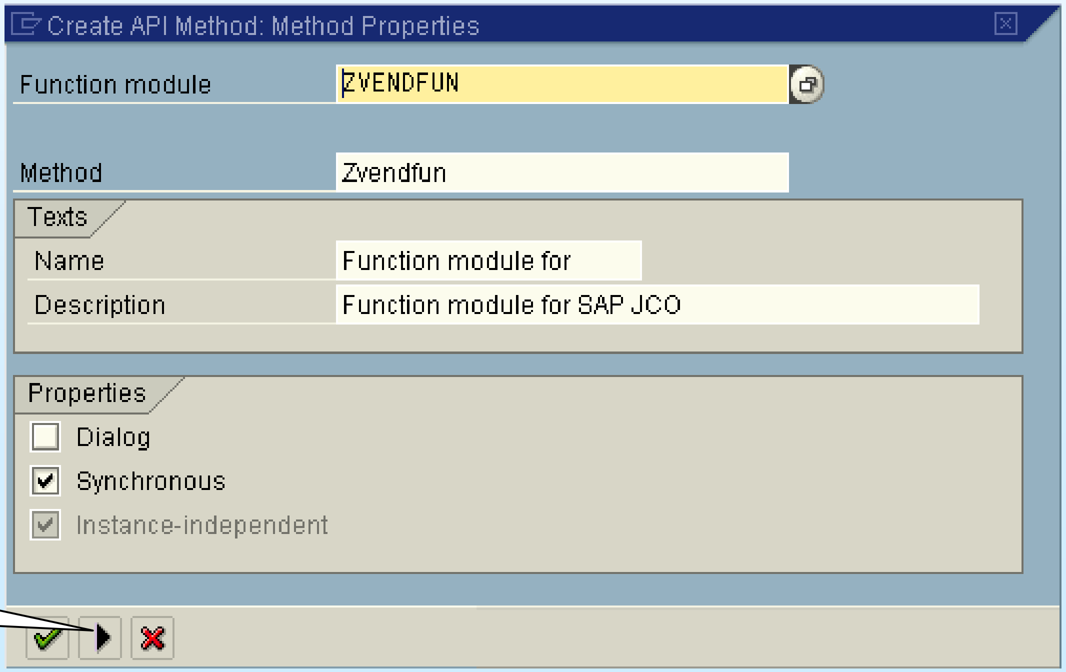
```
Object type  ZBAPI_VEND  Vendor Details
    Interfaces
    Key fields
    Attributes
    Methods
          ZBAPI_VEND.ExistenceCheck      Check existence of object
          ZBAPI_VEND.Display             Display object
    Events
```

Adding API method



Create API Method: Method Properties

Function module	ZVENDFUN



Create API Method: Method Properties

Function module	ZVENDFUN
Method	Zvendfun

Texts

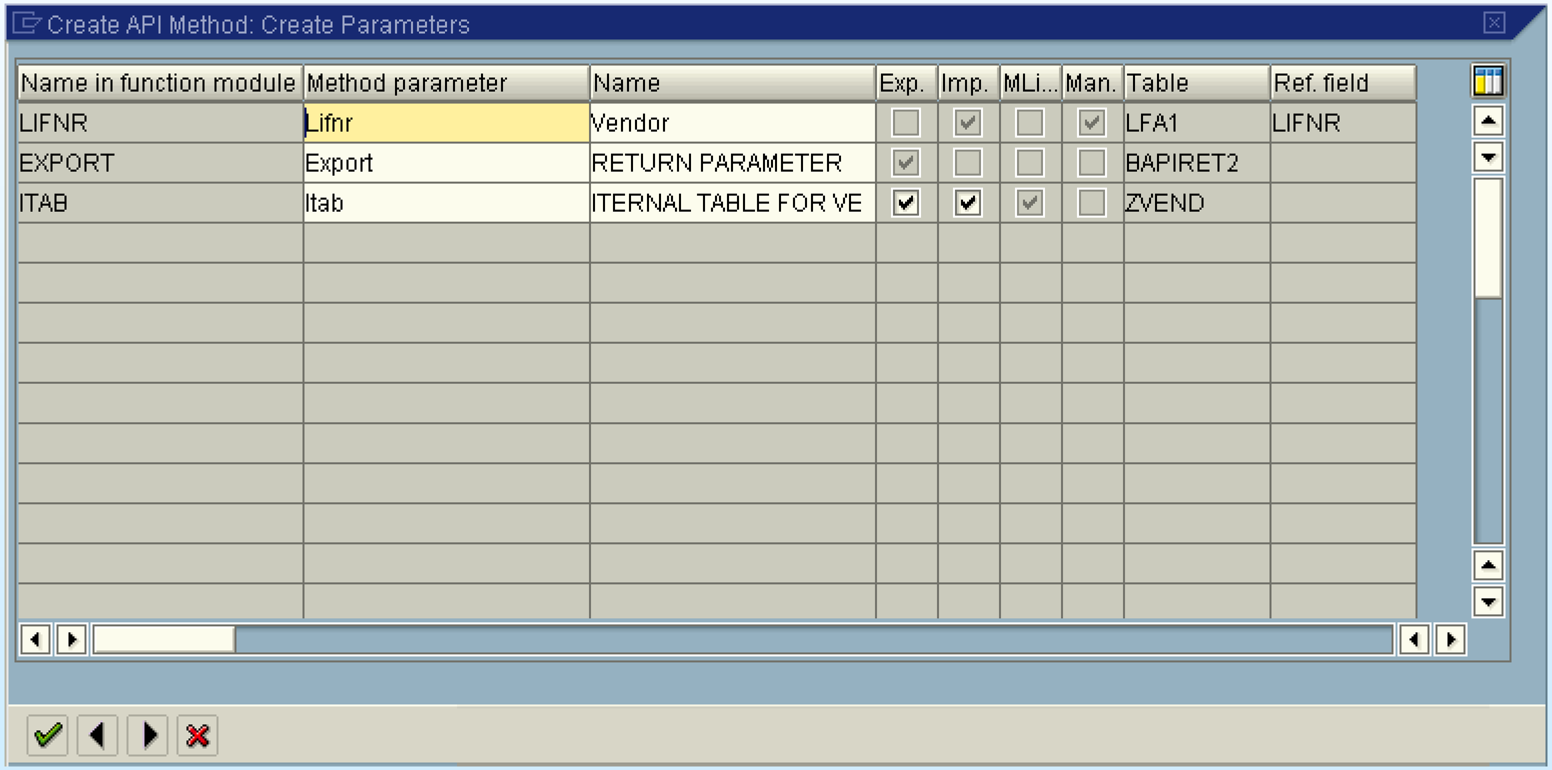
Name	Function module for
Description	Function module for SAP JCO

Properties

- ☐ Dialog
- ☑ Synchronous
- ☑ Instance-independent

Click here

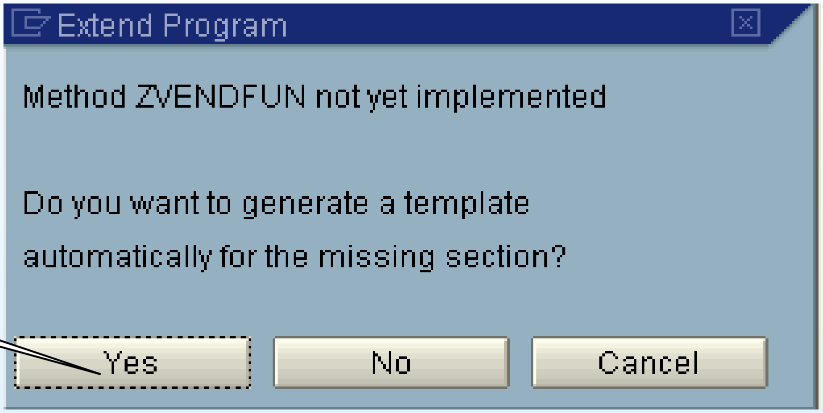
Adding API method



Create API Method: Create Parameters

Name in function module	Method parameter	Name	Exp.	Imp.	MLi...	Man.	Table	Ref. field
LIFNR	Lifnr	Vendor		✓		✓	LFA1	LIFNR
EXPORT	Export	RETURN PARAMETER	✓				BAPIRET2	
ITAB	Itab	ITERNAL TABLE FOR VE	✓	✓	✓		ZVEND	

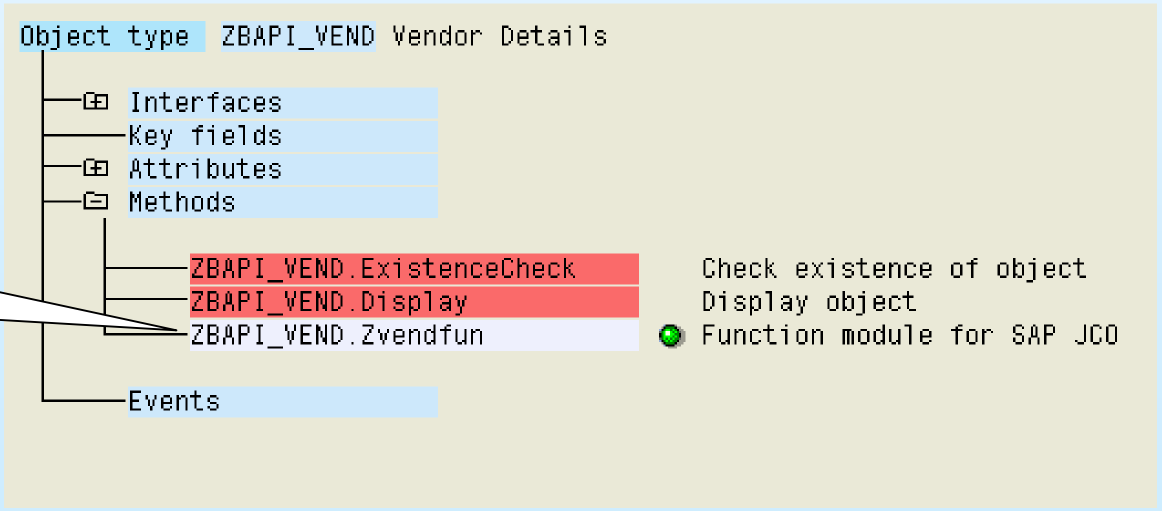
Adding API method



Method ZVENDFUN not yet implemented

Do you want to generate a template automatically for the missing section?

Click Yes

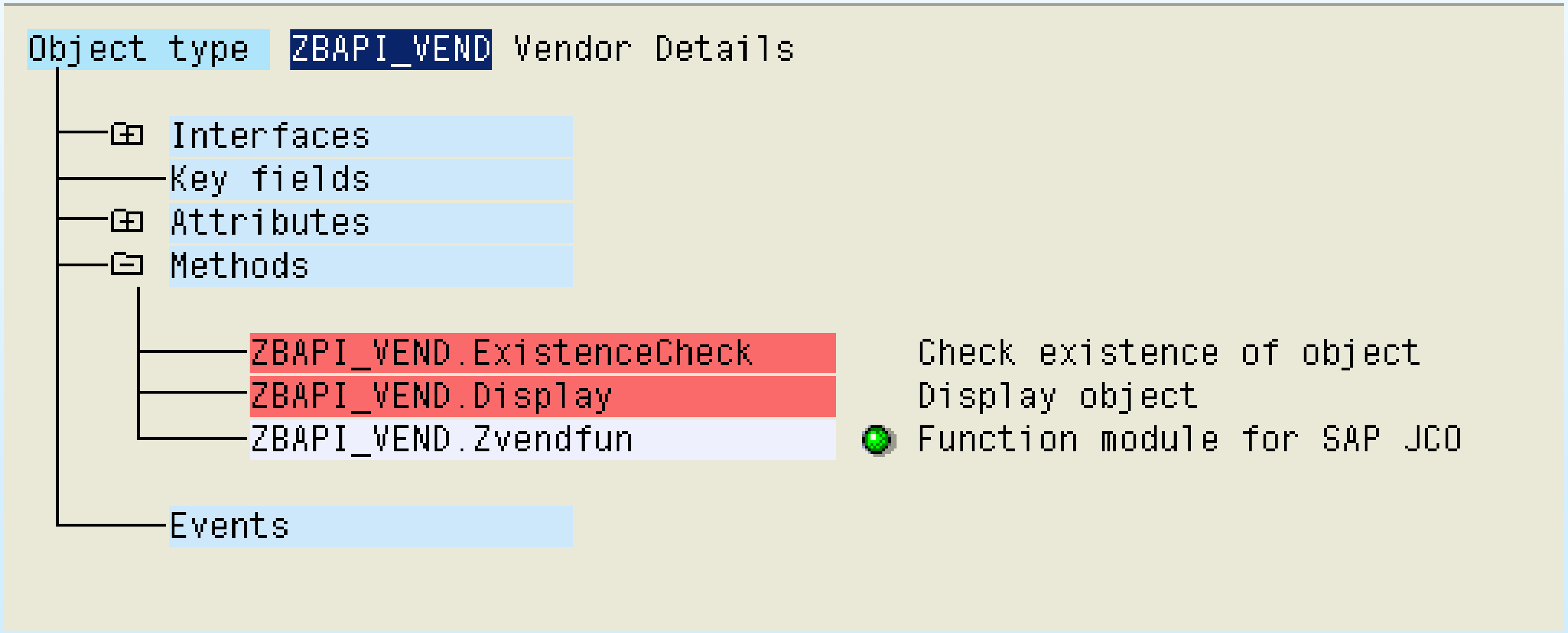


```
Object type  ZBAPI_VEND Vendor Details
  Interfaces
  Key fields
  Attributes
  Methods
    ZBAPI_VEND.ExistenceCheck     Check existence of object
    ZBAPI_VEND.Display            Display object
    ZBAPI_VEND.Zvendfun         ● Function module for SAP JCO
  Events
```

API method added

Implementing BAPI Object

Select the BAPI object

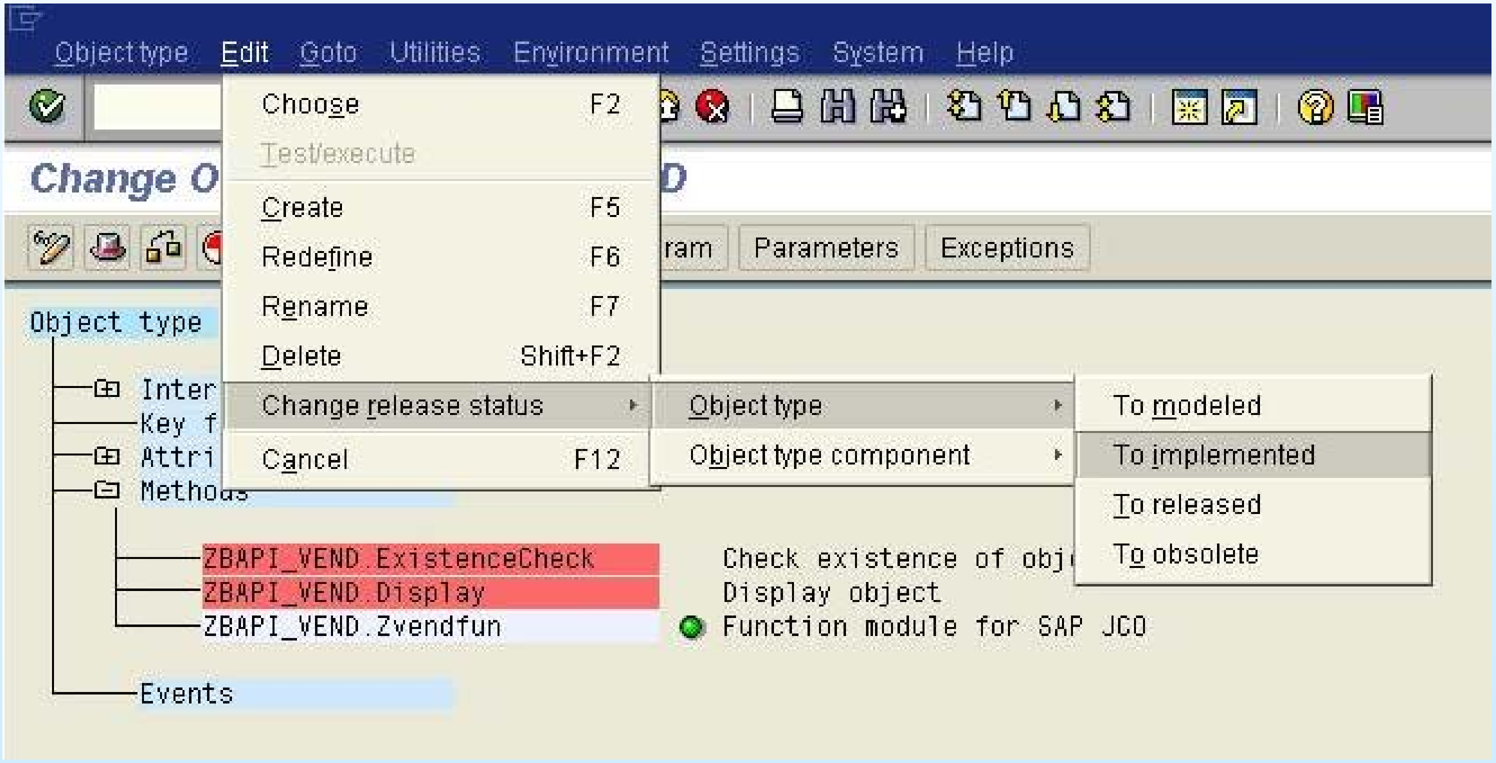


```
Object type  ZBAPI_VEND  Vendor Details

  ├─⊞ Interfaces
  ├──Key fields
  ├─⊞ Attributes
  ├─⊟ Methods
  │    ├──ZBAPI_VEND.ExistenceCheck        Check existence of object
  │    ├──ZBAPI_VEND.Display               Display object
  │    └──ZBAPI_VEND.Zvendfun           ●  Function module for SAP JCO
  └──Events
```

Implementing BAPI Object

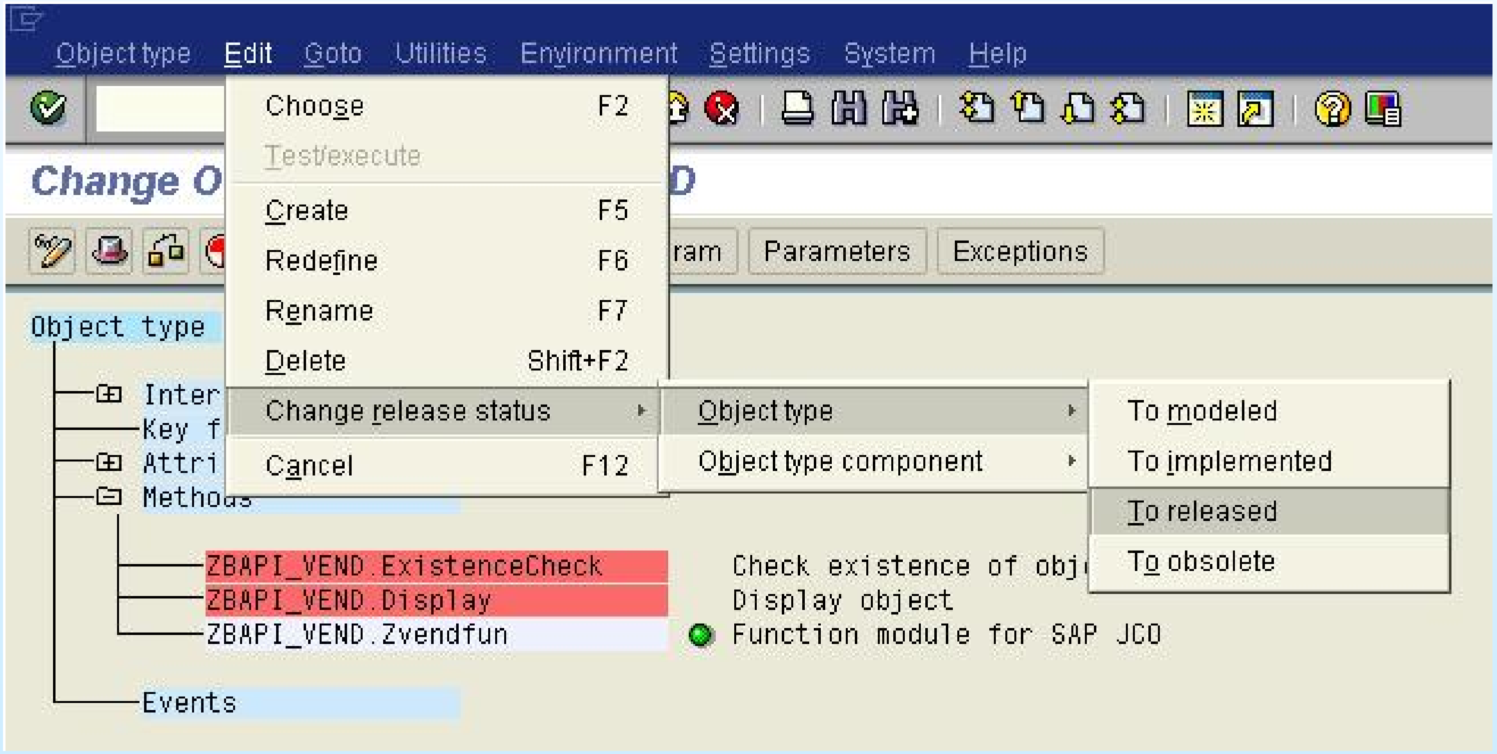
Change release status To implemented



```
ZBAPI_VEND.ExistenceCheck    Check existence of obj
ZBAPI_VEND.Display           Display object
ZBAPI_VEND.Zvendfun          Function module for SAP JCO
```

Releasing BAPI Object

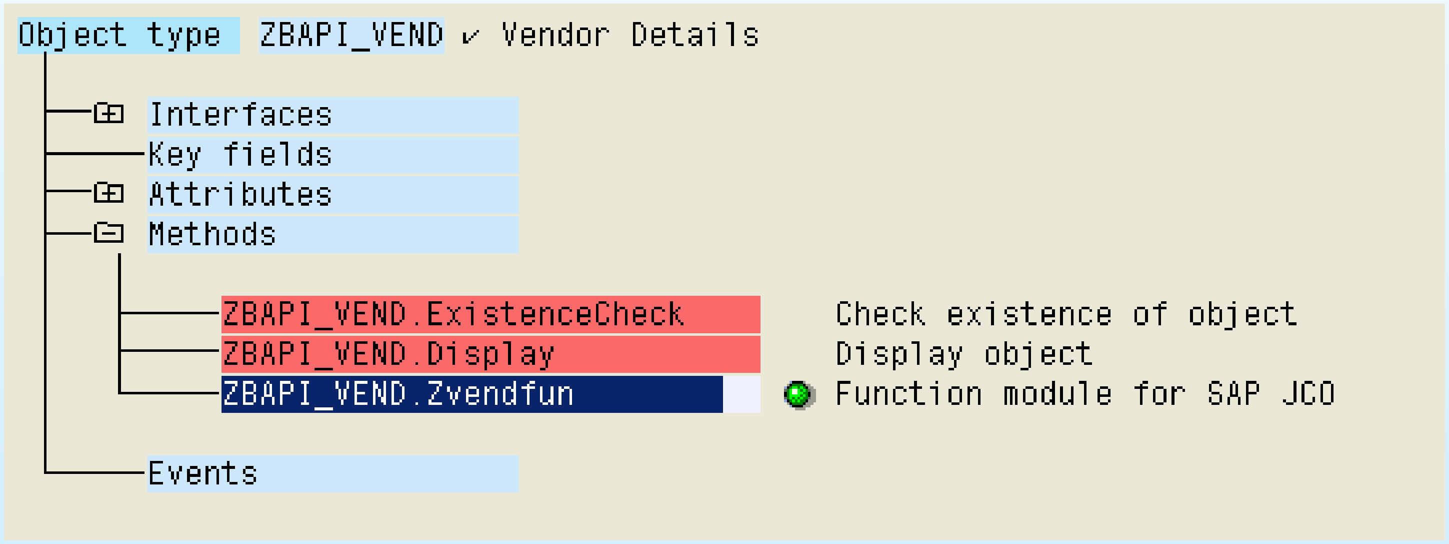
Change release status To released



```
ZBAPI_VEND.ExistenceCheck      Check existence of obj
ZBAPI_VEND.Display             Display object
ZBAPI_VEND.Zvendfun            Function module for SAP JCO
```

Implementing API Method

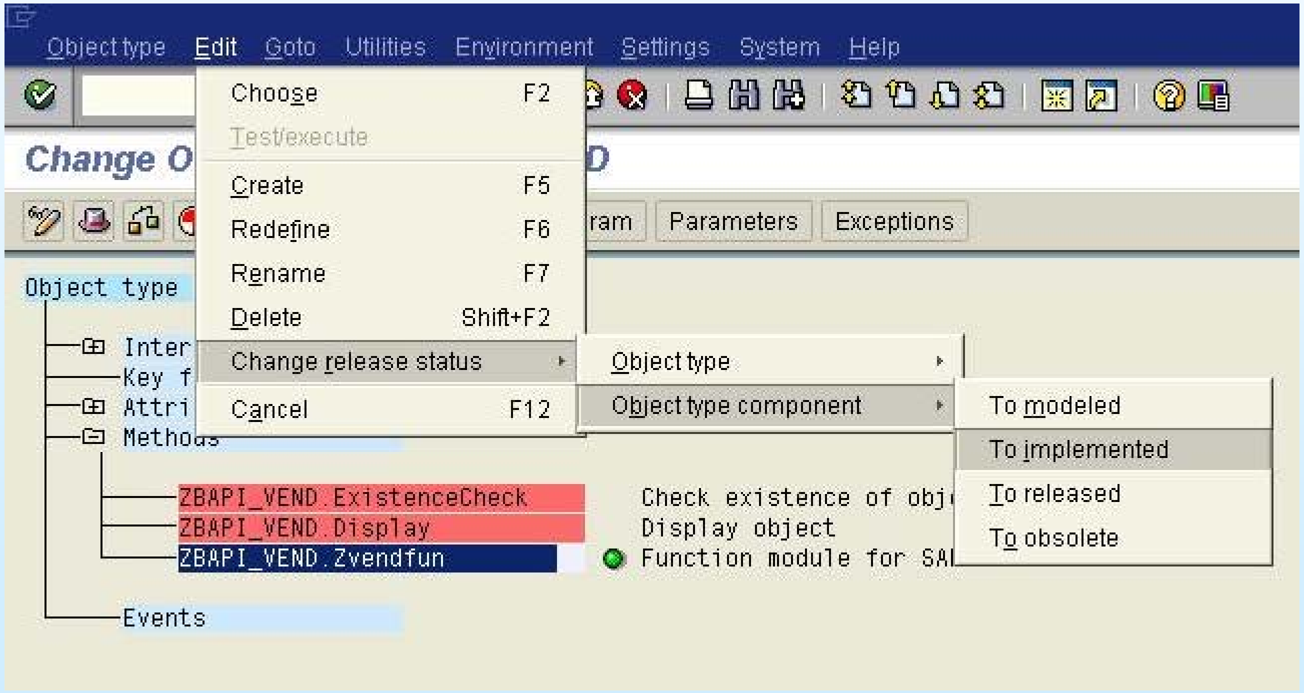
Select the API Method



```
Object type  ZBAPI_VEND ✓ Vendor Details
  ⊞ Interfaces
  Key fields
  ⊞ Attributes
  ⊟ Methods
      ZBAPI_VEND.ExistenceCheck     Check existence of object
      ZBAPI_VEND.Display            Display object
      ZBAPI_VEND.Zvendfun        ●  Function module for SAP JCO
  Events
```

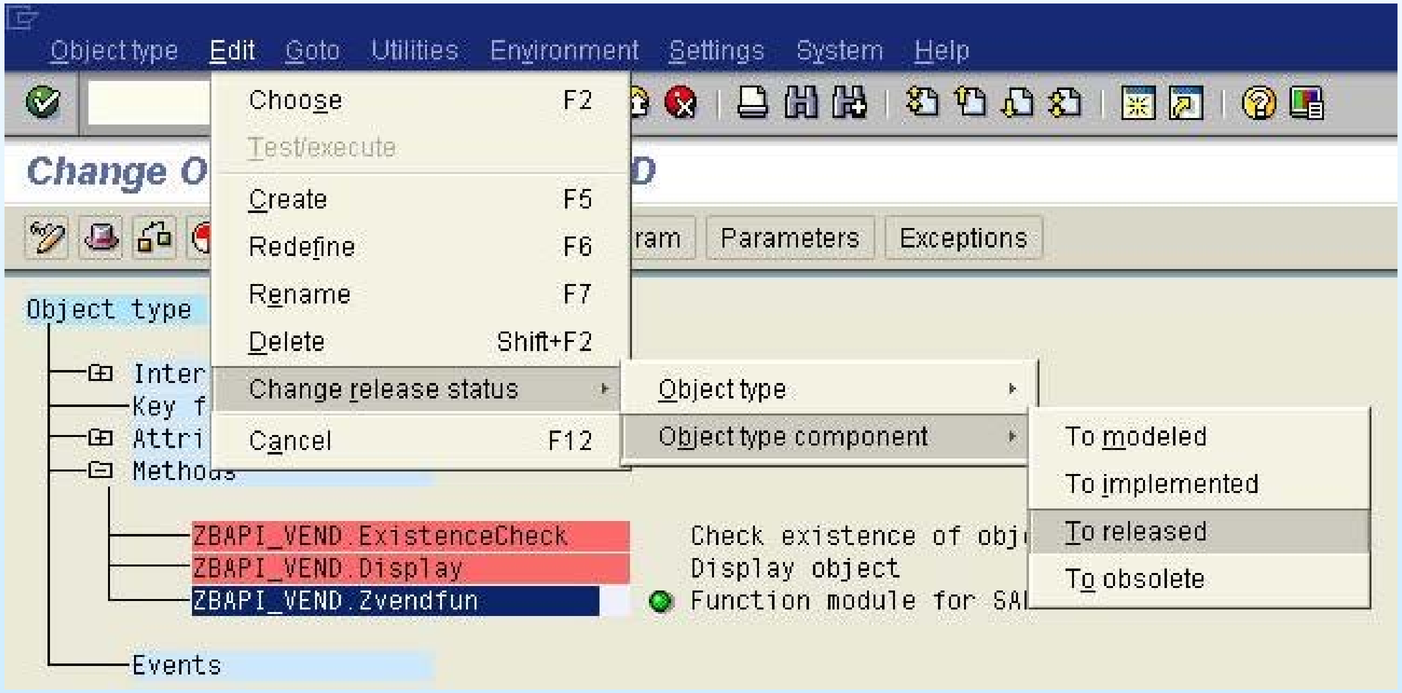
Implementing API Method

Change release status To implemented



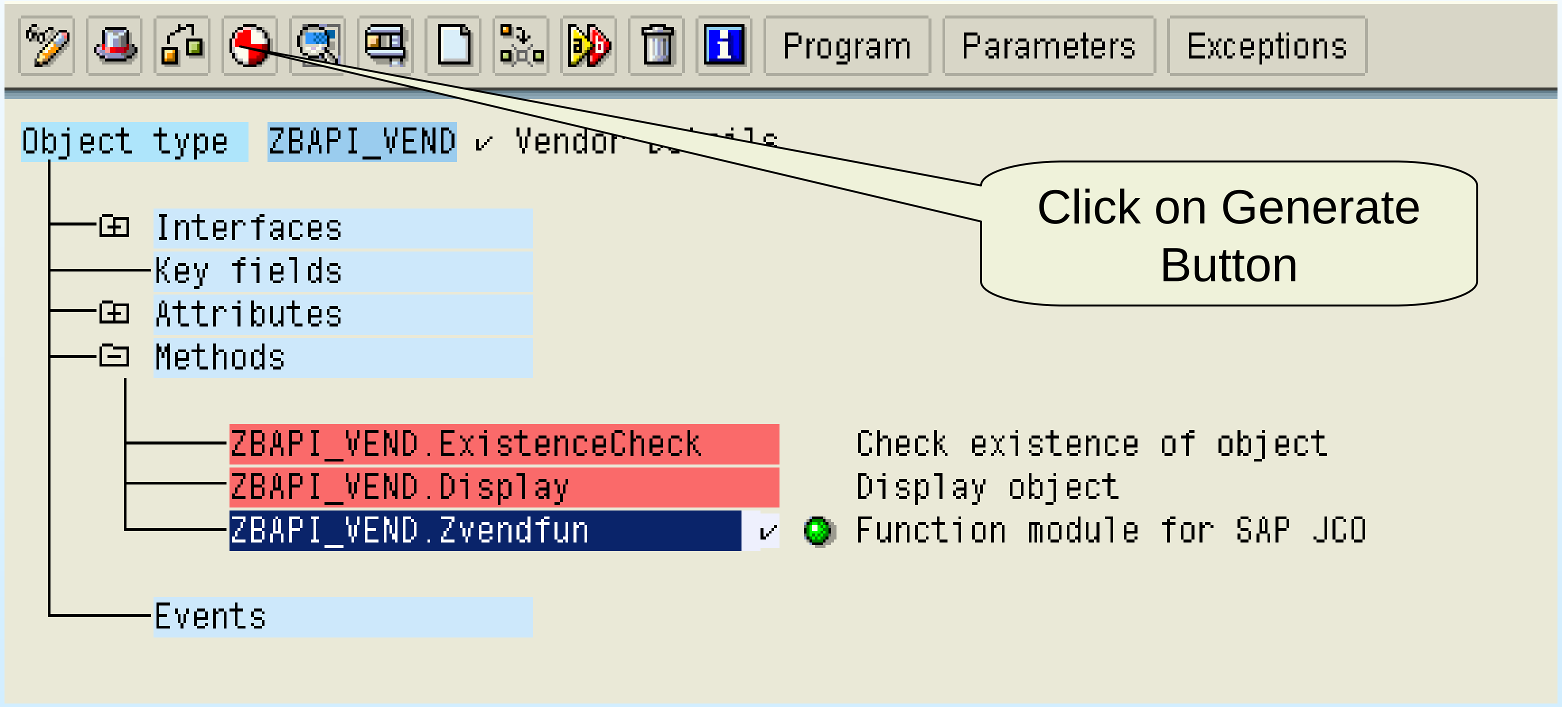
Releasing API Method

Change release status To released



```
ZBAPI_VEND.ExistenceCheck    Check existence of obj
ZBAPI_VEND.Display           Display object
ZBAPI_VEND.Zvendfun          Function module for SA
```

Generating API Method



```
Object type  ZBAPI_VEND ✓ Vendor Details

  Interfaces
  Key fields
  Attributes
  Methods
      ZBAPI_VEND.ExistenceCheck      Check existence of object
      ZBAPI_VEND.Display             Display object
      ZBAPI_VEND.Zvendfun        ✓   Function module for SAP JCO
  Events
```

Configuring Apache Tomcat

Directory Structure

```
Jakarta-tomcat-4.1.31
  └──► Webapps
         └──► <User Folder>  (Vendor)
                 │
                 ├──► WEB-INF
                 │      ├──► classes
                 │      ├──► lib
                 │      └──► Web.xml
                 └──► Index.html
```

Configuring Apache Tomcat

classes

This folder contains all the class files created for successful execution of the servlet.

lib

This folder contains all the library files required i.e
sapjco.jar
servlet.jar

Note: While compiling the java code make sure that the Classpath is set to the above to .jar files

Configuring Apache Tomcat

Web.xml

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<web-app xmlns="http://java.sun.com/xml/ns/j2ee“
          xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
          xsi:schemaLocation="http://java.sun.com/xml/ns/j2ee/web-app_2_4.xsd“ version="2.4">
     <servlet>
          <servlet-name>Some internal name</servlet-name>
         <servlet-class>display_vend</servlet-class>
     </servlet>
     <servlet-mapping>
          <servlet-name>Some internal name</servlet-name>
          <url-pattern>/NameSeenByUser.do</url-pattern>
      </servlet-mapping>
</web-app>
```

Servlet name

Servlet Program

import statements required

- import javax.servlet.*;
- import javax.servlet.http.*;
- import java.io.*;
- import com.sap.mw.jco.*;

Servlet Program

```
public class display_vend extends HttpServlet
  {
   PrintWriter pw;
   public void doPost(HttpServletRequest req, HttpServletResponse res)
     { int num = Integer.parseInt(req.getParameter("rand"));
       String no,name,city,district,po,tele,fax;
       String SID = "R"+num;
       String vendno = req.getParameter("vendno");
       IRepository repository; // The repository we will be using
        try {
           // Add a connection pool to the specified system
                JCO.addClientPool(SID, 100, "800", "hari", "sapnjoy", "EN", "sapides", "00” );
         // Alias for this pool , Max. number of connections , SAP client  , userid
         //  password  , language , host name
```

Unique name for connection pool each time connection is established random number is generated in the index.html i.e starting page and value is passed to servlet

Servlet Program

```
 repository = JCO.createRepository("MYRepository", SID); // Create a new repository
 // Get a function template from the repository
 IFunctionTemplate ftemplate = repository.getFunctionTemplate("ZVENDFUN");
// Create a function from the template
 JCO.Function function = new JCO.Function(ftemplate);
JCO.Client client = JCO.getClient(SID); // Get a client from the pool
JCO.ParameterList input = function.getImportParameterList(); // Fill in input parameters
 input.setValue(vendno, "LIFNR"   );
 client.execute(function); // Call the remote system
JCO.Structure ret = function.getExportParameterList().getStructure("RETURN");
 pw = res.getWriter();
 pw.println("<html><body bgcolor=#eeeff8><center><hr><h1>Customer Details</h1><hr>");
// Get table containing the data
JCO.Table vend = function.getTableParameterList().getTable("ITAB");
```

Servlet Program

```
for (int i = 0; i < vend.getNumRows(); i++)
   {
     vend.setRow(i);
     no = vend.getString("LIFNR");
    name = vend.getString("NAME1");
    city = vend.getString("ORT01") ;
    district = vend.getString("ORT02") ;
   po = vend.getString("PFACH") ;
   tele = vend.getString("TELF1") ;
   fax = vend.getString("TELFX") ;
   // Fetching data from SAP database and storing in local  variables
```

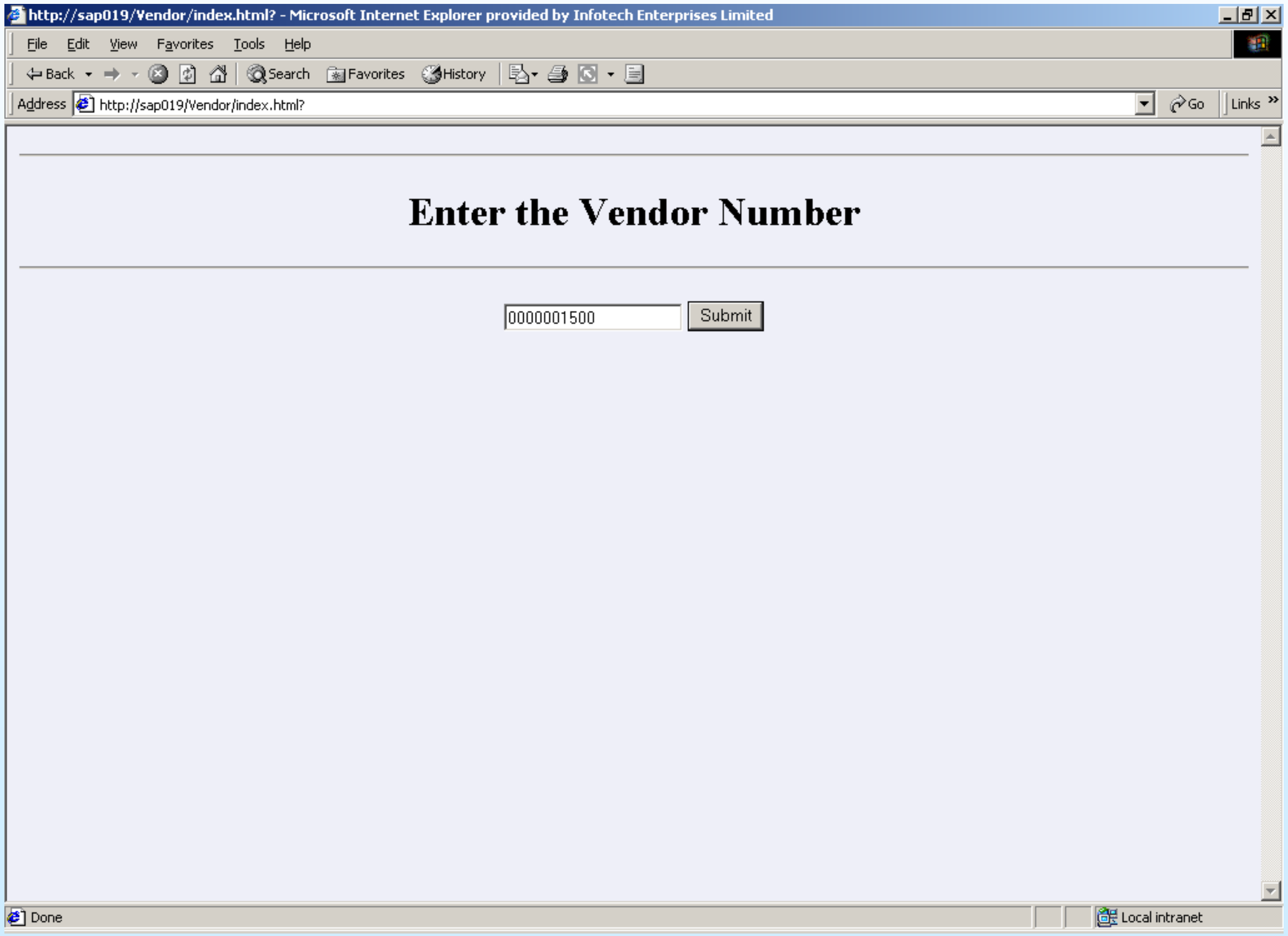
Servlet Program

```
pw.println("<table border=1><tr><td><B>Vendor Number</B></td><td>"+no+ "</td></tr><tr><td>" +
                "<B>Customer Name</B></td><td>"+name+ "</td></tr><tr><td>" +
                "<B>Customer Address</B></td><td></tr>"+
                "<tr><td>        </td><td><B>City</B></td><td>"+city+"</td></tr>" +
                "<tr><td>        </td><td><B>District</B></td><td>"+district+"</td></tr>"+
                "<tr><td>        </td><td><B>PO Box</B></td><td>"+po+"</td></tr>"+
                "<tr><td><B>Telephone</B></td><td>"+tele+"</td></tr>"+
                "<tr><td><B>TeleFax</B></td><td>"+fax+"</td></tr></table>"   );
pw.println("<form name=form1 action='index.html' method=get><input type=submit
value='Back'></form></center></body></html>");     } }
 catch (Exception E)
             {
               System.out.println(E);
             }
    }  }
```

Index.html

```
<html>
<head><script language="JavaScript">
function randomnumber() {
    var r=Math.floor(Math.random()*1111)
    if (r!=0) document.form1.rand.value=r; }
</script>
</head>
<body bgcolor=#eeeff8 onLoad="javascript:randomnumber();">
<center><hr><h1>Enter the Vendor Number</h1><hr>
</center><form name=form1 action="NameSeenByUser.do" method=post>
<center><input type=text name=vendno>
<input type=submit value="Submit">
<input type=hidden name="rand"></center>
</form></body></html>
```

Output

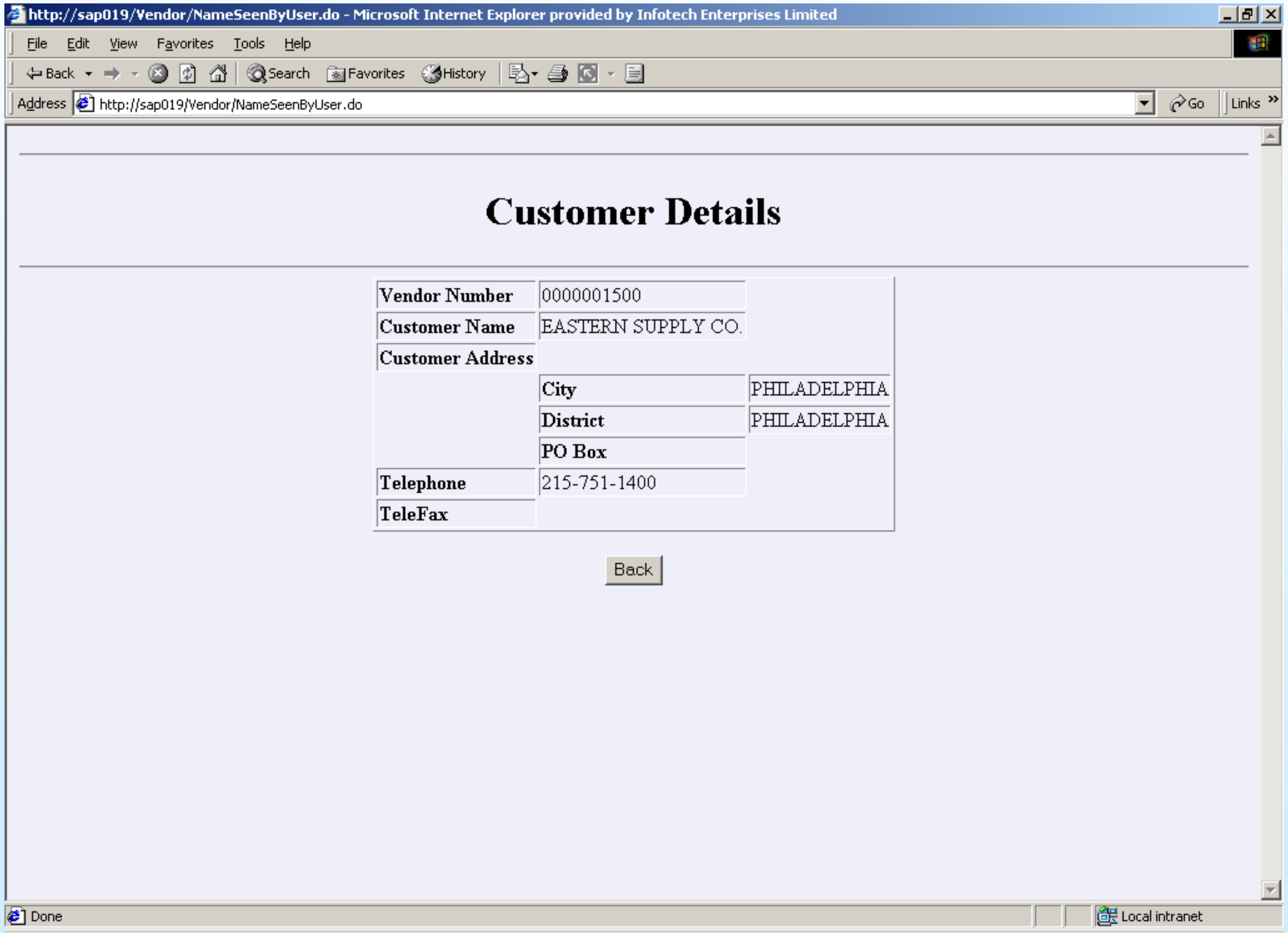


http://sap019/Vendor/index.html? - Microsoft Internet Explorer provided by Infotech Enterprises Limited

Enter the Vendor Number

0000001500

Output


http://sap019/Vendor/NameSeenByUser.do - Microsoft Internet Explorer provided by Infotech Enterprises Limited
File Edit View Favorites Tools Help
Back Search Favorites History
Address http://sap019/Vendor/NameSeenByUser.do
Go Links
Done
Local intranet

Customer Details

Vendor Number	0000001500	
Customer Name	EASTERN SUPPLY CO.	
Customer Address		
	City	PHILADELPHIA
	District	PHILADELPHIA
	PO Box	
Telephone	215-751-1400	
TeleFax		

Back