Implementing Customer Exit Reporting Variables as Methods

Applies to:

SAP BI 7.0

For more information, visit the Business Intelligence homepage.

Summary

This article describes how we can implement customer exit reporting variables as methods of a global class. Instead of changing the customer exit for populating the values for reporting variable, every time a new customer exit variable is created, a new method of the same name as the variable is created.

Author: Praveen M R

Company: IBM

Created on: 15 February 2009

Author Bio

Praveen is a SAP Certified Consultant working for IBM India. His areas of expertise are BI and ABAP.

Table of Contents

Introduction

Global Reporting Variables of processing type Customer Exit allows you to use a function module exit to set default values for variables. This is the function module *EXIT_SAPLRRS0_001.* The enhancement RSR00001 (*BI: Enhancements for Global Variables in Reporting*; transaction SMOD; component or function module EXIT_SAPLRRS0_001) is called several times during the execution of the report. The I_STEP parameter specifies when the enhancement is called.

When the number of Customer Exit processing type variables increases, the current standard solution of implementing the customer exit in the include, becomes unmanageable and difficult to control and co-ordinate between developers in a multi team environment. In scenarios of parallel development between different developers, there is a high chance of untested code getting transported across the landscape. This could lead to unavailability of BI reports in the system. Here, we look into a solution to avoid all those issues.

Solution

In the proposed solution, the code for populating the reporting variable will be implemented as a class method of a global class. This solution has two parts.

- The Global Class **ZC_BW_GLOBVAR**
- The Include **ZXRSRU01**

The class ZC_BW_GLOBVAR needs to be created in class builder SE24. Whenever a new customer exit variable is created, a new method of same name is added to the class ZC_BW_GLOBVAR and the code for populating the customer exit variable needs to be written inside the method. If a variable ZVBPFYPD is created in the variable maintenance screen, a new method ZVBPFYPD will be added to the class.

Include ZXRSRU01

Create a project in the CMOD transaction, select SAP enhancement **RSR00001** and assign it to the enhancement project. Activate the project. In the main include ZXRSRU01**,** the below code needs to be added.

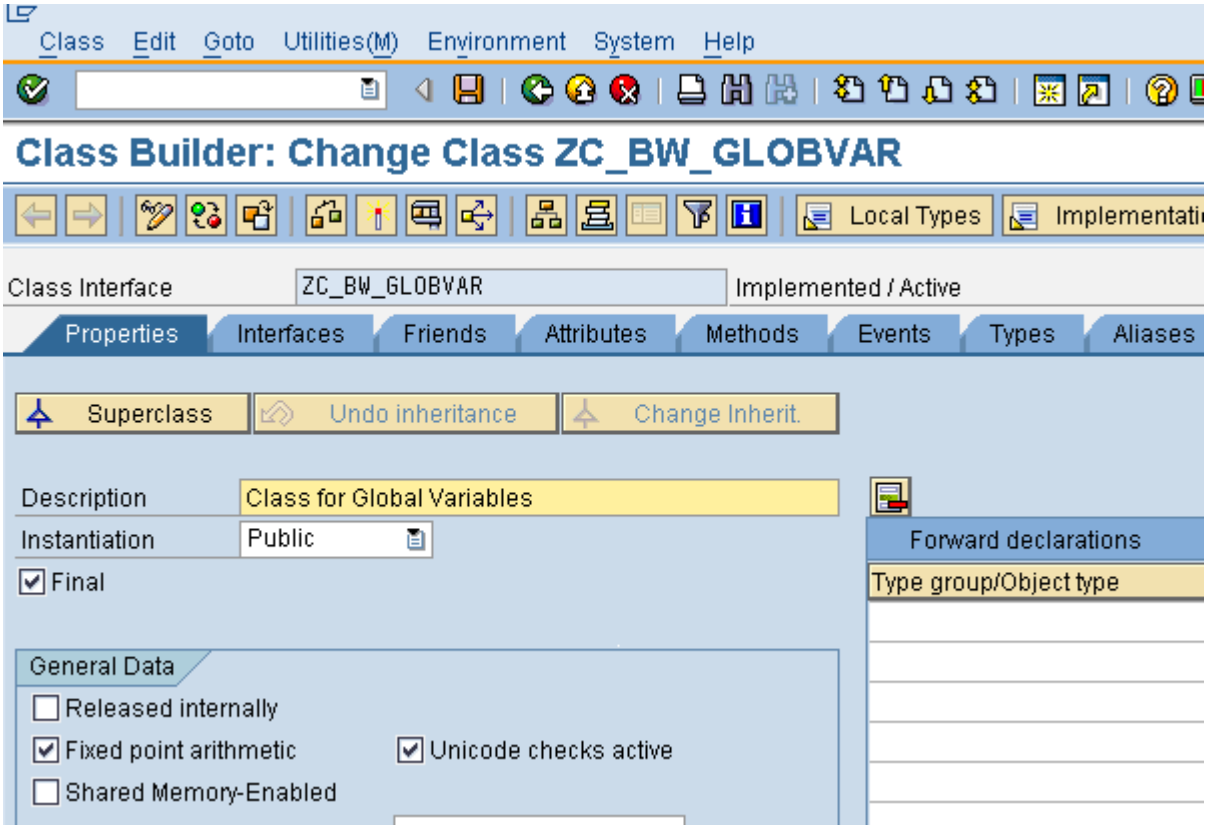
```
TRY.
IF I_STEP <> 3 .
  CALL METHOD ZC_BW_GLOBVAR=>(I_VNAM)
    EXPORTING
      I_VNAM        = I_VNAM
      I_VARTYP      = I_VARTYP
      I_IOBJNM      = I_IOBJNM
      I_S_COB_PRO   = I_S_COB_PRO
      I_S_RKB1D     = I_S_RKB1D
      I_PERIV       = I_PERIV
      I_T_VAR_RANGE = I_T_VAR_RANGE
      I_STEP        = I_STEP
    IMPORTING
      E_T_RANGE     = E_T_RANGE
      E_MEEHT       = E_MEEHT
      E_MEFAC       = E_MEFAC
      E_WAERS       = E_WAERS
      E_WHFAC       = E_WHFAC
    CHANGING
      C_S_CUSTOMER  = C_S_CUSTOMER
      .
ELSE.

  CALL METHOD ZC_BW_GLOBVAR=>STEP3
    EXPORTING
      I_VNAM        = I_VNAM
```

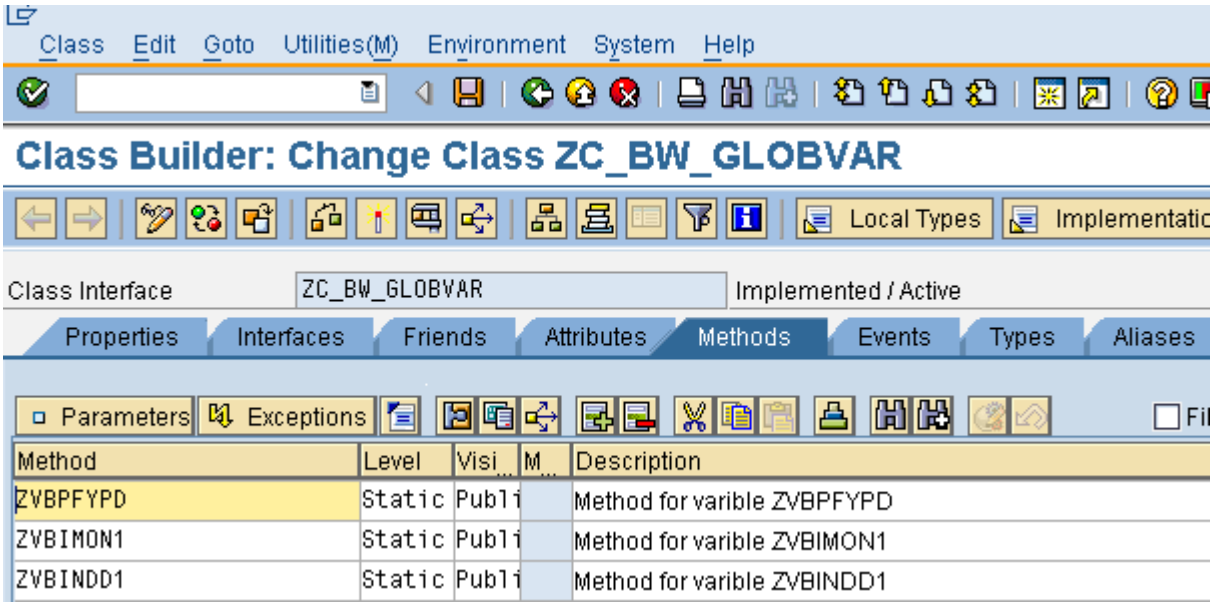
```
        I_VARTYP      = I_VARTYP
        I_IOBJNM      = I_IOBJNM
        I_S_COB_PRO   = I_S_COB_PRO
        I_S_RKB1D     = I_S_RKB1D
        I_PERIV       = I_PERIV
        I_T_VAR_RANGE = I_T_VAR_RANGE
        I_STEP        = I_STEP
      IMPORTING
        E_T_RANGE     = E_T_RANGE
        E_MEEHT       = E_MEEHT
        E_MEFAC       = E_MEFAC
        E_WAERS       = E_WAERS
        E_WHFAC       = E_WHFAC
      CHANGING
        C_S_CUSTOMER  = C_S_CUSTOMER
        .
  ENDIF.
    IF SY-SUBRC = 0 .
       EXIT.
    ENDIF.
  CATCH CX_SY_DYN_CALL_ERROR.
  ENDTRY.
```

Class ZC_BW_GLOBVAR

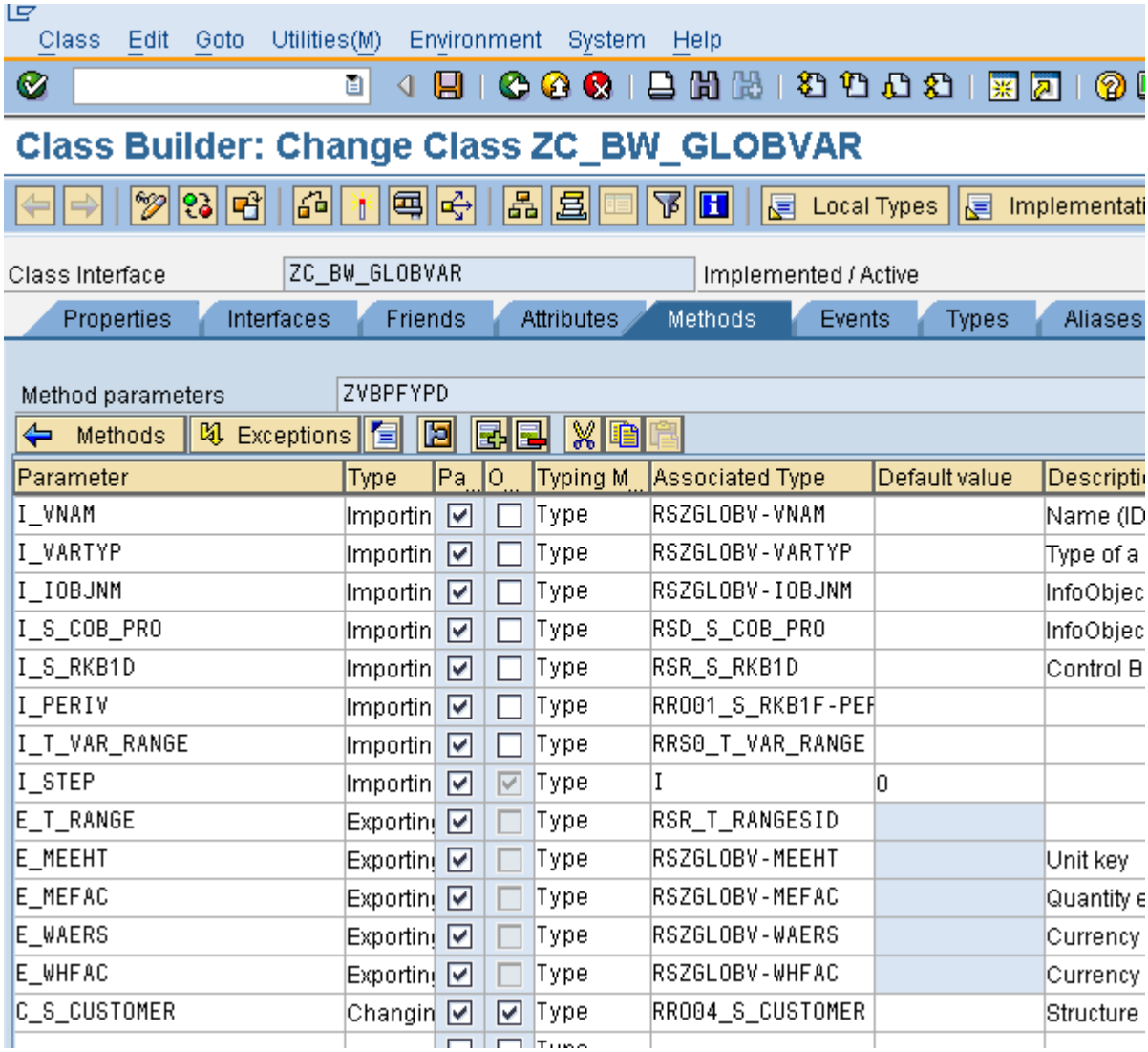
In the class builder (SE24) create class ZCBW_GLOBVAR.



For each reporting variable create a public static method of same name.

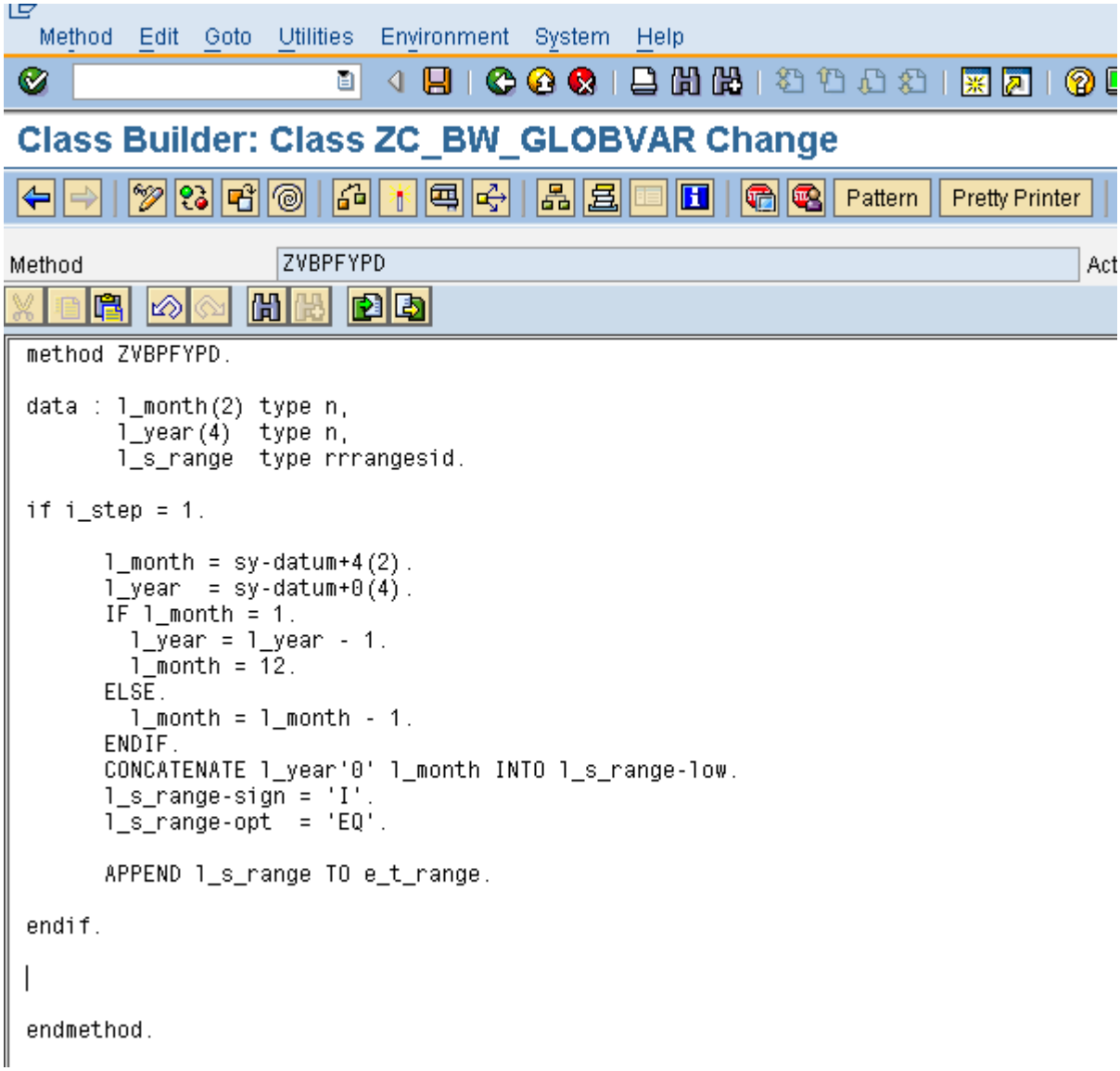


The parameter interface for the methods should be.



Parameter	Type	Pa...	O...	Typing M...	Associated Type	Default value	Descripti
I_VNAM	Importin	☑	☐	Type	RSZGLOBV-VNAM		Name (ID
I_VARTYP	Importin	☑	☐	Type	RSZGLOBV-VARTYP		Type of a
I_IOBJNM	Importin	☑	☐	Type	RSZGLOBV-IOBJNM		InfoObjec
I_S_COB_PRO	Importin	☑	☐	Type	RSD_S_COB_PRO		InfoObjec
I_S_RKB1D	Importin	☑	☐	Type	RSR_S_RKB1D		Control B
I_PERIV	Importin	☑	☐	Type	RRO01_S_RKB1F-PER		
I_T_VAR_RANGE	Importin	☑	☐	Type	RRS0_T_VAR_RANGE		
I_STEP	Importin	☑	☑	Type	I	0	
E_T_RANGE	Exportin	☑	☐	Type	RSR_T_RANGESID		
E_MEEHT	Exportin	☑	☐	Type	RSZGLOBV-MEEHT		Unit key
E_MEFAC	Exportin	☑	☐	Type	RSZGLOBV-MEFAC		Quantity e
E_WAERS	Exportin	☑	☐	Type	RSZGLOBV-WAERS		Currency
E_WHFAC	Exportin	☑	☐	Type	RSZGLOBV-WHFAC		Currency
C_S_CUSTOMER	Changin	☑	☑	Type	RRO04_S_CUSTOMER		Structure

Now we can write the code for populating the report variable in the method that we created.




```
method ZVBPFYPD.

data : l_month(2) type n,
       l_year(4)  type n,
       l_s_range  type rrrangesid.

if i_step = 1.

      l_month = sy-datum+4(2).
      l_year  = sy-datum+0(4).
      IF l_month = 1.
        l_year = l_year - 1.
        l_month = 12.
      ELSE.
        l_month = l_month - 1.
      ENDIF.
      CONCATENATE l_year'0' l_month INTO l_s_range-low.
      l_s_range-sign = 'I'.
      l_s_range-opt  = 'EQ'.

      APPEND l_s_range TO e_t_range.

endif.

endmethod.
```

Validating Reporting Variables

The customer exit EXIT_SAPLRRS0_001 is also used to check the values of the reporting variables, this is done when I_STEP = 3 and when as exception is triggered the variable screen appears again. Creating a public static method STEP3 in class ZC_BW_GLOBVAR and writing the code inside the method, we can implement this validation.

Advantages

This approach prevents the situation where the main program SAPLXRSR having post transport syntax errors leading to unavailability of all the BI reports in the system. Hence eliminating the effort and time required to correct these post transport syntax errors and the down time for BI reports.

One time change to the user exit include. Enables parallel development of reporting variables efficiently.

Related Content

SAP Library - BI Suite - Reporting Variables -Customer Exit

SAP Library - Customer Exits

SAP Library - Class Builder

For more information, visit the Business Intelligence homepage.

Disclaimer and Liability Notice

This document may discuss sample coding or other information that does not include SAP official interfaces and therefore is not supported by SAP. Changes made based on this information are not supported and can be overwritten during an upgrade.

SAP will not be held liable for any damages caused by using or misusing the information, code or methods suggested in this document, and anyone using these methods does so at his/her own risk.

SAP offers no guarantees and assumes no responsibility or liability of any type with respect to the content of this technical article or code sample, including any liability resulting from incompatibility between the content within this document and the materials and services offered by SAP. You agree that you will not hold, or seek to hold, SAP responsible or liable with respect to the content of this document.