How-To Guide: Implement Value Helps in GenIL Components

SAP CRM 7.0

Target Audience
- System administrators
- Technology consultants

Document version: 1.2 – December 2008
Terms for Included Open Source Software

This SAP software contains also the third party open source software products listed below. Please note that for these third party products the following special terms and conditions shall apply.

1. This software was developed using ANTLR.

2. gSOAP
Part of the software embedded in this product is gSOAP software. Portions created by gSOAP are Copyright (C) 2001-2004 Robert A. van Engelen, Genivia inc. All Rights Reserved.

THE SOFTWARE IN THIS PRODUCT WAS IN PART PROVIDED BY GENIVIA INC AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE AUTHOR BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

3. SAP License Agreement for STLport
SAP License Agreement for STLPort between
SAP Aktiengesellschaft
Systems, Applications, Products in Data Processing
Neurottstrasse 16
69190 Walldorf, Germany
(hereinafter: SAP)

(a) Subject Matter of the Agreement
A) SAP grants Customer a non-exclusive, non-transferrable, royalty-free license to use the STLport.org C++ library (STLport) and its documentation without fee.
B) By downloading, using, or copying STLport or any portion thereof Customer agrees to abide by the intellectual property laws, and to all of the terms and conditions of this Agreement.
C) The Customer may distribute binaries compiled with STLport (whether original or modified) without any royalties or restrictions.

D) Customer shall maintain the following copyright and permissions notices on STLport sources and its documentation unchanged:

Copyright 2001 SAP AG

E) The Customer may distribute original or modified STLport sources, provided that:
o The conditions indicated in the above permissions notice are met;
o The following copyright notices are retained when present, and conditions provided in accompanying permission notices are met:

Copyright 1994 Hewlett-Packard Company
Copyright 1996,97 Silicon Graphics Computer Systems Inc.
Copyright 1997 Moscow Center for SPARC Technology.
Copyright 1999,2000 Boris Fomitchev

Permission to use, copy, modify, distribute and sell this software and its documentation for any purposes is hereby granted without fee, provided that the above copyright notice appear in all copies and that both that copyright notice and this permission notice appear in supporting documentation. Hewlett-Packard Company makes no representations about the suitability of this software for any purpose. It is provided "as is" without express or implied warranty.
Permission to use, copy, modify, distribute and sell this software and its documentation for any purpose is hereby granted without fee, provided that the above copyright notice appear in all copies and that both that copyright notice and this permission notice appear in supporting documentation. Silicon Graphics makes no representations about the suitability of this software for any purpose. It is provided “as is” without express or implied warranty.

Permission to use, copy, modify, distribute and sell this software and its documentation for any purposes is hereby granted without fee, provided that the above copyright notice and this permission notice appear in all copies and that both that copyright notice and this permission notice appear in supporting documentation. Moscow Center for SPARC makes no representations about the suitability of this software for any purpose. It is provided “as is” without express or implied warranty. Boris Fomitchev makes no representations about the suitability of this software for any purpose. This material is provided “as is”, with absolutely no warranty expressed or implied. Any use is at your own risk. Permission to use or copy this software for any purpose is hereby granted without fee, provided the above notices are retained on all copies. Permission to modify the code and to distribute modified code is granted, provided the above notices are retained, and a notice that the code was modified is included with the above copyright notice.

Permission to use, copy, modify, distribute and sell this software and its documentation for any purposes is hereby granted without fee, provided that the above copyright notice appear in all copies and that both that copyright notice and this permission notice appear in supporting documentation. SAP makes no representations about the suitability of this software for any purpose. It is provided with a limited warranty and liability as set forth in the License Agreement distributed with this copy. SAP offers this liability and warranty obligations only towards its customers and only referring to its modifications.

b) Support and Maintenance
SAP does not provide software maintenance for the STLport. Software maintenance of the STLport therefore shall be not included. All other services shall be charged according to the rates for services quoted in the SAP List of Prices and Conditions and shall be subject to a separate contract.

c) Exclusion of warranty
As the STLport is transferred to the Customer on a loan basis and free of charge, SAP cannot guarantee that the STLport is error-free, without material defects or suitable for a specific application under third-party rights. Technical data, sales brochures, advertising text and quality descriptions produced by SAP do not indicate any assurance of particular attributes.

d) Limited Liability
A) Irrespective of the legal reasons, SAP shall only be liable for damage, including unauthorized operation, if this (i) can be compensated under the Product Liability Act or (ii) if caused due to gross negligence or intent by SAP or (iii) if based on the failure of a guaranteed attribute.

B) If SAP is liable for gross negligence or intent caused by employees who are neither agents or managerial employees of SAP, the total liability for such damage and a maximum limit on the scope of any such damage shall depend on the extent to which its occurrence ought to have anticipated by SAP when concluding the contract, due to the circumstances known to it at that point in time representing a typical transfer of the software.

C) In the case of Art. 4.2 above, SAP shall not be liable for indirect damage, consequential damage caused by a defect or lost profit.
D) SAP and the Customer agree that the typical foreseeable extent of damage shall under no circumstances exceed EUR 5,000.
E) The Customer shall take adequate measures for the protection of data and programs, in particular by making backup copies at the minimum intervals recommended by SAP. SAP shall not be liable for the loss of data and its recovery, notwithstanding the other limitations of the present Art. 4 if this loss could have been avoided by observing this obligation.

F) The exclusion or the limitation of claims in accordance with the present Art. 4 includes claims against employees or agents of SAP.

4. Adobe Document Services
Adobe, the Adobe logo, Acrobat, PostScript, and Reader are either registered trademarks or trademarks of Adobe Systems Incorporated in the United States and / or other countries. For information on Third Party software delivered with Adobe document services and Adobe LiveCycle Designer, see SAP Note 854621.
## Typographic Conventions

<table>
<thead>
<tr>
<th>Type Style</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Example Text</td>
<td>Words or characters quoted from the screen. These include field names, screen titles, pushbuttons labels, menu names, menu paths, and menu options. Cross-references to other documentation.</td>
</tr>
<tr>
<td>Example text</td>
<td>Emphasized words or phrases in body text, graphic titles, and table titles</td>
</tr>
<tr>
<td>EXAMPLE TEXT</td>
<td>Technical names of system objects. These include report names, program names, transaction codes, table names, and key concepts of a programming language when they are surrounded by body text, for example, SELECT and INCLUDE.</td>
</tr>
<tr>
<td>Example text</td>
<td>Output on the screen. This includes file and directory names and their paths, messages, names of variables and parameters, source text, and names of installation, upgrade and database tools.</td>
</tr>
<tr>
<td>Example text</td>
<td>Exact user entry. These are words or characters that you enter in the system exactly as they appear in the documentation.</td>
</tr>
<tr>
<td>&lt;Example text&gt;</td>
<td>Variable user entry. Angle brackets indicate that you replace these words and characters with appropriate entries to make entries in the system.</td>
</tr>
<tr>
<td>EXAMPLE TEXT</td>
<td>Keys on the keyboard, for example, F2 or ENTER.</td>
</tr>
</tbody>
</table>

## Icons

<table>
<thead>
<tr>
<th>Icon</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>☢</td>
<td>Caution</td>
</tr>
<tr>
<td>🏠</td>
<td>Example</td>
</tr>
<tr>
<td>🔄</td>
<td>Note</td>
</tr>
<tr>
<td>💬</td>
<td>Recommendation</td>
</tr>
<tr>
<td>📦</td>
<td>Syntax</td>
</tr>
</tbody>
</table>

Additional icons are used in SAP Library documentation to help you identify different types of information at a glance. For more information, see Help on Help → General Information Classes and Information Classes for Business Information Warehouse on the first page of any version of SAP Library.
Contents

1 Introduction ......................................................................................................................... 8
2 Procedures .......................................................................................................................... 9
  2.1 Value Help for Input Fields in Freestyle Mode .......................................................... 9
  2.2 Value Help for Context Node Attributes ...................................................................... 9
  2.3 Implementing the New Method...................................................................................... 9
    2.3.1 Mapping Strategies ................................................................................................ 10
    2.3.2 Examples for Search ID Types ............................................................................... 12
  2.4 Example for Custom Search Help ............................................................................... 15
  2.5 Value Help for an Advanced Search ........................................................................... 18
1 Introduction

A common task for a Customer Relationship Management (CRM) developer or consultant is to provide a value help box for a business object attribute displayed in a view on the CRM WebClient UI.

Note that this document only describes the general process. The screenshots and codelines are only used as examples. Ensure that you always use a customer namespace for enhancements and do not modify SAP components. For information on how to enhance SAP components, see the How-To Guide: Framework Enhancements, which is part of the consultant cookbook.

The following are not supported by the current value help implementation:

- Complex search help
- Collective search help
- Events in search exits other than select and display
2 Procedures

2.1 Value Help for Input Fields in Freestyle Mode

To create a value help for an input field that is in freestyle mode in a view, do the following:

1. Open the component workbench (transaction BSP_WD_CMPWB) and navigate to the component view containing the relevant attribute.

2. Open the view layout and use the THTMLB: INPUTFIELD tag where needed.

   The value attribute is not mandatory. The attribute ID is mandatory only if the value attribute is not assigned to a context node attribute or the name attribute is not used. If the value attribute is bound to the context node attribute and the ID attribute is not assigned, the ID is set to the name of the node attribute. The output mapping of the example would then be HELPOUTPUTFIELDS = "STRUCT.NAMCOUNTRYISO =PARTNER".

   Syntax

   <thtmlb:inputField id               = "MEMBER_ID"
    helpId           = "CRMSH_PRIL_BUPAP"
    helpInputFields  = "MEMBER_ID=PARTNER"
    helpOutputFields = "MEMBER_ID=PARTNER"
    value     = "//PARTNER/STRUCT.NAMCOUNTRYISO"/>

   If the value attribute is not transferred properly, make sure that the output mapping points to a valid input field ID.

2.2 Value Help for Context Node Attributes

To create a value help for a context node attribute in a view with a configuration tag, do the following:

1. Open the component workbench (transaction BSP_WD_CMPWB) and navigate to the component view containing the attribute for which you want to offer value help.

2. Go to the configuration tab and open the field properties for the desired attribute. Make sure that the attribute is an input field.

3. Go to the view structure tab and open the implementation class below the context node used for display.

4. Use the method _GET_V_XYZ as a template by choosing Copy from its context menu.

5. Rename the copied value help getter to reflect the appropriate attribute (for example, _GET_V_COUNTRYORIGIN).

2.3 Implementing the New Method

To implement the method that you created in the previous section (for example, _GET_V_COUNTRYORIGIN), you must provide the following information within the method:

- The name of the search help, data element, and custom value help to use for the current value help
- A list of input mapping that can be used as search criteria
• A list of output mapping that defines to which fields the selected values are transferred
• The type of search help that you are using

Only the name and the output mapping are required. A utility class has been created to simplify the definition of value help: CL_BSP_WD_VALUEHELP_F4DESCR:

- ls_map type: IF_BSP_WD_VALUEHELP_F4DESCR=>GTYPE_PARAM_MAPPING
- lt_inmap type: IF_BSP_WD_VALUEHELP_F4DESCR=>GTYPE_PARAM_MAPPING_TAB
- lt_outmap type: IF_BSP_WD_VALUEHELP_F4DESCR=>GTYPE_PARAM_MAPPING_TAB

The class supports the following search ID types:

<table>
<thead>
<tr>
<th>Search ID Type</th>
<th>Technical Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>Reference Structure Field</td>
<td>HELP_ID_KIND_COMP</td>
</tr>
<tr>
<td>DDIC Name</td>
<td>HELP_ID_KIND_NAME</td>
</tr>
<tr>
<td>Reference Data Element</td>
<td>HELP_ID_KIND_DTEL</td>
</tr>
</tbody>
</table>

The following sections provide detailed information on the implantation of the new method.

### 2.3.1 Mapping Strategies

#### 2.3.1.1 Regular Mapping

The mapping of input and output is different depending on the context in which you are implementing value help.

In the first scenario, the value help is located directly in a view.

The input and output context attributes must be defined using the STRUCT. prefix. In the following example, both input and output mapping use the context node attribute STRUCT.COUNTRYORIGIN and are associated with the search parameter PRODUCT_ID.

**Syntax**

```java
data:
  ls_map  type IF_BSP_WD_VALUEHELP_F4DESCR=>GTYPE_PARAM_MAPPING,
  lt_inmap type IF_BSP_WD_VALUEHELP_F4DESCR=>GTYPE_PARAM_MAPPING_TAB,
  lt_outmap type IF_BSP_WD_VALUEHELP_F4DESCR=>GTYPE_PARAM_MAPPING_TAB.
```

```java
ls_map-context_attr = 'struct.prd_id'.
ls_map-f4_attr      = 'PRODUCT_ID'.
append ls_map to: lt_inmap, lt_outmap.
```

If the value help is within a table view, the syntax is different. The TABLE[] prefix must be used and the index of the row where the value help is located must also be provided. The next example creates two output mappings to the LANGU and LANGU_TEXT attributes. The IV_INDEX is transferred to the v-method by the framework.
2 Procedures

Syntax

DATA:
    lv_str_index TYPE string,
    ls_map    TYPE if_bsp_wd_valuehelp_f4descr=>gtype_param_mapping,
    lt_inmap  TYPE
    if_bsp_wd_valuehelp_f4descr=>gtype_param_mapping_tab,
    lt_outmap TYPE
    if_bsp_wd_valuehelp_f4descr=>gtype_param_mapping_tab.

    lv_str_index = iv_index.
    CONCATENATE 'table[' lv_str_index '].langu' INTO ls_map-context_attr.
    ls_map-f4_attr      = 'KEY'.
    APPEND ls_map TO: lt_outmap.

    lv_str_index = iv_index.
    CONCATENATE 'table[' lv_str_index '].langu_text' INTO ls_map-context_attr.
    ls_map-f4_attr      = 'VALUE'.
    APPEND ls_map TO: lt_outmap.

If the value help is used within a tree, the syntax is different. Include the TABLE[] prefix, the index of the row, and the name of the tree.

The next example creates one output mapping to the LANGU attributes using a tree. The IV_INDEX is passed to the v-method by the framework.

Syntax

DATA:
    lv_str_index TYPE string,
    ls_map    TYPE if_bsp_wd_valuehelp_f4descr=>gtype_param_mapping,
    lt_inmap  TYPE
    if_bsp_wd_valuehelp_f4descr=>gtype_param_mapping_tab,
    lt_outmap TYPE
    if_bsp_wd_valuehelp_f4descr=>gtype_param_mapping_tab.

    lv_str_index = iv_index.
    CONCATENATE 'firsttree_table[' lv_str_index '].langu' INTO ls_map-context_attr.
    ls_map-f4_attr      = 'KEY'.
    APPEND ls_map TO: lt_outmap.

Sometimes values not present on the UI have to be sent, for example, to refine a search. You can do this by using a constant with input mapping as follows:

Syntax

ls_map-context_attr = '\"321654987\"'.
ls_map-f4_attr = 'NUMBER'.

2.3.1.2 Context-Sensitive Mapping

When using a search help, it is possible to pass initial values from the UI to the value help and to limit the number of records sent back by the value help. For example, consider the search help BUPAP, displayed in the figure below.

You can see in its definition that there are seven search help parameters. The PARTNER parameter has the IMP column checked. This means PARTNER can be used as an input
mapping and is used as an initial search parameter. Multiple input mappings can also be provided.

If the dialog type is defined as Dialog with value restriction, the value help displays the search criteria with the initial values automatically added. If the dialog type is Display values immediately, the help values are automatically displayed using the input mapping to limit the result.

When the search help has multiple IMP parameters, you can create the same number of input mappings in the v-method of the context node attribute. For example, if the BUPAP search help has PARTNER, MC_NAME1, and MC_NAME2 marked as IMP parameters, the following mappings can be created:

Syntax

```plaintext
ls_map-context_attr = 'STRUCT.PARTNER'.
ls_map-f4_attr      = 'PARTNER'.
append ls_map to: lt_inmap.

ls_map-context_attr = 'STRUCT.NAME1'.
ls_map-f4_attr      = MC_NAME1'.
append ls_map to: lt_inmap.

ls_map-context_attr = '\\W\\'.
ls_map-f4_attr      = MC_NAME2'.
append ls_map to: lt_inmap.
```

2.3.2 Examples for Search ID Types

2.3.2.1 Reference Structure Field
For the *Reference Structure Field* search ID type, you can map a check table to either the structure or a DDIC search help. If you are using the check table, you receive all of the records for your value help without the possibility of restricting the selection.

To use the new method for your value help, you must adjust the following variables in it:

- **LS_MAPPING_CONTEXT_ATTR** (context attribute)
- **LS_MAPPING_F4_ATTR** (help parameter)
  
  In the following example, this is the ‘KEY’ of the used check table.
- **IV_HELP_ID** (structure field reference)

### Syntax

```plaintext
data:
  ls_map    type IF_BSP_WD_VALUEHELP_F4DESCR=>GTYPE_PARAM_MAPPING,
  lt_inmap  type IF_BSP_WD_VALUEHELP_F4DESCR=>GTYPE_PARAM_MAPPING_TAB,
  lt_outmap type IF_BSP_WD_VALUEHELP_F4DESCR=>GTYPE_PARAM_MAPPING_TAB.

  ls_map-context_attr = 'struct.countryorigin'.
  ls_map-f4_attr      = 'KEY'.
  append ls_map to: lt_inmap, lt_outmap.

  create object rv_valuehelp_descriptor type CL_BSP_WD_VALUEHELP_F4DESCR
    exporting
      iv_help_id        = 'CRMST_ADDRESS_BUIL-COUNTRY'
      iv_help_id_kind   = IF_BSP_WD_VALUEHELP_F4DESCR=>HELP_ID_KIND_COMP
      iv_input_mapping  = lt_inmap
      iv_output_mapping = lt_outmap.
```

### 2.3.2.2 DDIC Name (Using a Specific Search Help)

This example uses the DDIC search help H_T005_LAND, which has the import parameter LAND1. This parameter is used (as a search help parameter) in the method you implement in the next step. All fields defined in the search help can be used as output mapping, even if the column is not visible on the UI. This is the case for all search parameters where *LPos* equals 0.
To use the new method for your value help, you must adjust the following variables in it:

- **LS_MAPPING_CONTEXT_ATTR** (context attribute)
- **LS_MAPPING_F4_ATTR** (search help parameter)
- **IV_HELP_ID** (name of the value help in the DDIC)

### Syntax

```data
ls_map    type IF_BSP_WD_VALUEHELP_F4DESCR=>GTYPE_PARAM_MAPPING,
lt_inmap  type IF_BSP_WD_VALUEHELP_F4DESCR=>GTYPE_PARAM_MAPPING_TAB,
lt_outmap type IF_BSP_WD_VALUEHELP_F4DESCR=>GTYPE_PARAM_MAPPING_TAB.

ls_map-context_attr = 'struct.countryorigin'.
ls_map-f4_attr      = 'LAND1'.
append ls_map to: lt_inmap, lt_outmap.

create object rv_valuehelp_descriptor type CL_BSP_WD_VALUEHELP_F4DESCR
exporting
    iv_help_id        = 'H_T005_LAND'
    iv_help_id_kind   = IF_BSP_WD_VALUEHELP_F4DESCR=>HELP_ID_KIND_NAME
    iv_input_mapping  = lt_inmap
```
iv_output_mapping = lt_outmap.
endmethod.

2.3.2.3 Reference Data Element

You can also use a data element with a DDIC value help assigned to it as a search ID type.

To use the new method for your value help, you must adjust the following variables in it:

- LS_MAPPING_CONTEXT_ATTR (context attribute)
- LS_MAPPING_F4_ATTR (search help parameter)
- IV_HELP_ID (name of the data element)

Syntax

data:
   ls_map    type IF_BSP_WD_VALUEHELP_F4DESCR=>GTYPE_PARAM_MAPPING,
   lt_inmap  type IF_BSP_WD_VALUEHELP_F4DESCR=>GTYPE_PARAM_MAPPING_TAB,
   lt_outmap type IF_BSP_WD_VALUEHELP_F4DESCR=>GTYPE_PARAM_MAPPING_TAB.

   ls_map-context_attr = 'struct.countryorigin'.
   ls_map-f4_attr      = 'LAND1'.
   append ls_map to: lt_inmap, lt_outmap.

   create object rv_valuehelp_descriptor type
   CL_BSP_WD_VALUEHELP_F4DESCR
      exporting
         iv_help_id        = 'ZCOUNTRY'
         iv_help_id_kind   = IF_BSP_WD_VALUEHELP_F4DESCR=>HELP_ID_KIND_DTEL
         iv_input_mapping  = lt_inmap
         iv_output_mapping = lt_outmap.
endmethod.

2.4 Example for Custom Search Help

The framework allows you to create your own custom search help. The examples described here can be found in the following locations:

<table>
<thead>
<tr>
<th>Component</th>
<th>CRM_THTMLB_COMP</th>
</tr>
</thead>
<tbody>
<tr>
<td>View</td>
<td>InputFieldTagDisplay</td>
</tr>
<tr>
<td>V-method redefinition</td>
<td>CL_CRM_THTM_INPUTFIELDTAG_CN00-&gt;GET_V_BP_DESCRIPTION</td>
</tr>
<tr>
<td>Adding a reference</td>
<td>CL_CRM_THTM_INPUTFIELDTAG_CTXT-&gt;CREATE_PARTNERS</td>
</tr>
</tbody>
</table>
1. Create a class that implements the IF_BSP_WD_CUSTOM_F4_CALLBACK interface. The following classes are used by the F4 controller to retrieve your value:
   - CT_RESULTS_TAB
     This is an internal table containing the values to display in the value help. Only KEY/VALUE list is supported.
   - IR_CUSTOM_REF
     This references an object that can be used to retrieve the data. For example, you could decide to use a custom controller to retrieve data.
   - IS_SEARCH_HELP
     You can use the search description to retrieve the search parameters.

**Syntax**

```
METHOD IF_BSP_WD_CUSTOM_F4_CALLBACK~RETRIEVE_CUSTOM_VALUES.
  DATA: lt_scarr_tab TYPE scarr_tab,
       ls_scarr TYPE scarr,
       ls_key_value TYPE shsvalstr.

  SELECT * FROM scarr INTO TABLE lt_scarr_tab.

  LOOP AT lt_scarr_tab INTO ls_scarr.
    ls_key_value-key = ls_scarr-carrid.
    ls_key_value-value = ls_scarr-carrname.
    APPEND ls_key_value to ct_results_tab.
  ENDLOOP.
ENDMETHOD.
```

2. Redefine the V-method.

Input and output mappings are defined normally. You can add input and output mapping to fields in the UI or pass constants as input mapping as described in *Mapping Strategies* [page 10].

The main differences are the definition of the search help ID and the possibility of adding an extra input mapping. The extra input mapping is a constant used to retrieve a reference to an object that can be passed to the custom search help implementation class.

**Search help ID**: Custom search help IDs are defined by placing the name of your custom class in between parentheses ((...)). For example, ‘(CL_SEARCH_HELP_NAME)’.

**Object reference**: It is also possible to pass an extra input mapping parameter to advise the F4 controller that an external object has to be used by the callback. This object is defined by adding the input mapping F4_ATTR = “KEY_REF” in the v-method. Only one object can be passed to a callback. In addition to defining an input mapping for a reference, the actual reference to the object has to be stored. The class CL_BSP_WD_REFERENCE_TOOL stores all the references used by the F4 controller. In the following example, a reference to a custom controller is registered in
CL_BSP_WD_REFERENCE_TOOL using the key ‘CRM_THTMLB_COMP/CUPARTNER’. The same key is reused in the v-method and sent to the F4 controller by an input mapping.

Using the v-method below, the system generates and executes the JavaScript when a user clicks the value help icon. The first highlighted parameter defines the input field ID from where the value help was called and the third parameter identifies the search help to use (in this case, the custom search help CL_CUSTOM_F4). You can also see that the F4 controller receives only one input mapping parameter (KEY_REF) and sets the column UNIT of the selected F4 row in the ‘C1_W1_V6_PARTNERS_STRUCT.BP_DESCRIPTION’ input field:

```
ICWCShowF4HelpExternNew(null,null,null,document.getElementById('C1_W1_V6_partners_struct.bp_description'),null,'(CL_CUSTOM_F4)',null,'X','C1_W1_V6','C1_W1_V6_partners.crm_thtmlb_comp/cupartner'=KEY_REF,'C1_W1_V6_partners_struct.bp_description=UNIT')
```

To add an object reference to be reused by the custom search help, you can use the following code:

**Syntax**

```plaintext
DATA: lr_cc_ref TYPE REF TO object.

lr_cc_ref = owner->get_custom_controller( controller_id = 'CRM_THTMLB_COMP/CUPARTNER' ).
cl_bsp_wd_reference_tool=>set_reference( iv_controller_key = 'CRM_THTMLB_COMP/CUPARTNER' iv_controller_ref = lr_cc_ref ).
```

Below is the v-method associated with the input field using the custom value help. Two mappings are defined: the first one is an output mapping to the STRUCT.BP_DESCRIPTION field and the second is an input mapping indicating that the custom search uses an object reference that has been previously saved using the CRM_THTMLB_COMP/CUPARTNER key and CL_BSP_WD_REFERENCE_TOOL class.

**Syntax**

```plaintext
METHOD get_v_bp_description.

DATA:
  ls_map TYPE
  if_bsp_wd_valuehelp_f4descr=>gtype_param_mapping,
  lt_inmap TYPE
  if_bsp_wd_valuehelp_f4descr=>gtype_param_mapping_tab,
  lt_outmap TYPE
  if_bsp_wd_valuehelp_f4descr=>gtype_param_mapping_tab.

  ls_map-context_attr = 'STRUCT.BP_DESCRIPTION'.
  ls_map-f4_attr = 'UNIT'.
  APPEND ls_map TO lt_outmap.

  ls_map-context_attr = '\"CRM_THTMLB_COMP/CUPARTNER\"'.
  ls_map-f4_attr = IF_THTMLB_CUSTOM_F4_CALLBACK=>GC_KEY_REF.
  APPEND ls_map TO lt_inmap.
```
CREATE OBJECT rv_valuehelp_descriptor TYPE cl_bsp_wd_valuehelp_f4descr
EXPORTING
  iv_help_id = '(CL_CUSTOM_F4)'
  iv_help_id_kind = if_bsp_wd_valuehelp_f4descr=>help_id_kind_comp
  iv_input_mapping = lt_inmap
  iv_output_mapping = lt_outmap.
ENDMETHOD.

2.5 Value Help for an Advanced Search

For the advanced search, you do not have to create a value help getter. Instead, you must redefine method GET_DQUERY_DEFINITIONS in the implementation class of the view controller. Table RT_RESULT can be filled in with information relating to the value help.

For example, see structure CRMS_THTMLB_SEARCH_FIELD_INFO and attributes HELP_ID, INPUT_MAPPING, and OUTPUT_MAPPING in the following code:

Syntax

method GET_DQUERY_DEFINITIONS.
  data:
    ls_ddlb type CRMS_THTMLB_SEARCH_DDLB_NVP.

type-context: <rt_result> type CRMS_THTMLB_SEARCH_FIELD_INFO.

  CALL METHOD SUPER->GET_DQUERY_DEFINITIONS
    RECEIVING
      RT_RESULT = rt_result.
  * F4-help for field 'ORDER_NUMBER
    read table rt_result with key field = 'ORDER_NUMBER' assigning
    <rt_result>.
    if sy-subrc = 0.
      <rt_result>-help_id-help_id = 'CRM_ORDER_BY_OBJECT_ID'.
      <rt_result>-input_mapping-CONTEXT_ATTR = 'ORDER_NUMBER'.
      <rt_result>-input_mapping-F4_ATTR = 'OBJECT_ID'.
      <rt_result>-output_mapping-CONTEXT_ATTR = 'ORDER_NUMBER'.
      <rt_result>-output_mapping-F4_ATTR = 'OBJECT_ID'.
    endif.
  endmethod.