Enhancement Framework - Function group and Function module enhancement - Concept and Simple Scenario

Enhancement Framework  Function group (Function module) enhancements

The enhancement framework for function modules allows us to enhance the standard application function module parameters with the new business functionality.

New parameters can be added based on the functionality required to be extended for a standard SAP object.

⚠️ Note: The parameters which are enhanced must be 'optional' in nature as being mandatory will require the changes to be registered for all the calls made by. This can be a real time consuming and tedious process.

⚠️ Caution! - This functional ability can be enabled or achieved using the 'COPY' or popularly called as 'CLONING' technique which will serve the purpose, but will be a great tough time working during an 'UPGRADE'.

As a SAP developer the best utilization would be the combined 'Benefit' of 'Function-group (function module)' enhancement with 'Source code enhancements (Source code plug-ins)'.

Here, to demonstrate the powerful technique of 'Function-group' enhancement we consider a basic level scenario which will make things simple.

⚠️ Note: This scenario deals with a 'Z' function module but practically what would be the approach to enhance the standards can be well seen.

A few conventions followed in this process and terminologies,

'DV_KUNNR' - Customer Number
'DV_CCODE' - Country Code
'DV_NAME' - Customer Name
'DV_STRAS' - Street or House Number
'DV_LIFNR' - Vendor

Consider a customer has a function module (say) 'ZDAVE_ENHC_INTERFACE'.

There are two standard (say) parameters as 'Importing' (DV_KUNNR and DV_CCODE)

<table>
<thead>
<tr>
<th>Parameter Name</th>
<th>Type</th>
<th>Associated Type</th>
<th>Default value</th>
<th>Opti.</th>
<th>Pas.</th>
<th>Short text</th>
</tr>
</thead>
<tbody>
<tr>
<td>DV_KUNNR</td>
<td>TYPE</td>
<td>KUNNR</td>
<td></td>
<td></td>
<td></td>
<td>Customer Number</td>
</tr>
<tr>
<td>DV_CCODE</td>
<td>TYPE</td>
<td>LAND1</td>
<td></td>
<td></td>
<td></td>
<td>Country Key</td>
</tr>
</tbody>
</table>

'DV_NAME' as 'Exporting'
Check out the ‘Global Data’ declarations for structures and internal tables in the function group.

Originally the structure ‘FS_KNA1’ and internal table ‘T_KNA1’ are present.
Referring back to the function module 'Source code' which was originally written.
Test the function module to check its original functionality.

Customer

Country code

Test Function Module: Initial Screen

Test for function group: ZDAVE_ENHC_INTERFACE_FGP
Function module: ZDAVE_ENHC_INTERFACE
Uppercase/Lowercase: 

<table>
<thead>
<tr>
<th>Import parameters</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>DV_KUNNR</td>
<td>1000C</td>
</tr>
<tr>
<td>DV_CCODE</td>
<td>IN</td>
</tr>
</tbody>
</table>

Observe the output, which gives the 'Name' of the customer on input of 'Customer' number and 'Country code'.
Now, the time has come to enhance the original business functionality. Follow the path 'Function module  Enhance Interface'.

Fill the 'Create' implementation details.
The customer (say) wants to add two new parameters to the original `export` parameter ‘DV_NAME’ with ‘DV_STRAS’ and ‘DV_LIFNR’.

Observe the difference.

Next step would be to enhance the `Source code` in order to link the new parameters and perform the required operations.

So create an ‘Implicit enhancement’.

Remember! : We can add our code implicitly which makes it little flexible as ‘Explicit’ needs ‘ENHANCEMENT-POINT’ or an ‘ENHANCEMENT-T-SECTION’ which are not present everywhere but only at some locations.

As it is understood that ‘Implicit’ can be done at certain predefined places in the source code, here we enhance the original structure ‘FS_KNA1’ in the `function group global data`. 
Fill in the details to enhance the original structure 'T_KNA1' with fields 'STRAS' and 'LIFNR' respectively.

Activate the enhancement and observe.
Create another implicit enhancement for after the complete original code ends.
⚠️ **Note:** Fill in the 'Create' implementation dialog for the implicit enhancement which will allow creating a new internal table which will hold the original with the additional data required and finally the data will be transferred to the original table in order to make the functionality work.
After enhancing the function group, make sure all the enhancements are active and return to the function module source code to enhance the functionality.
Choose 'CODE' for 'dynamic enhancement implementation'.

Fill in the details and possibly choose a different name in order to easily manage the implementations.
Observe the code very carefully and then activate the enhancement.

Finally, the time has come to confirm the functionality of the enhanced object.

Test and observe the difference.
Test Function Module: Result Screen

Test for function group: ZDAVE_ENHC_INTERFACE_FOP
Function module: ZDAVE_ENHC_INTERFACE

Runtime: 786 Microseconds

<table>
<thead>
<tr>
<th>Import parameters</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>DV_KUNNR</td>
<td>1000</td>
</tr>
<tr>
<td>DV_CCODE</td>
<td>IN</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Export parameters</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>DV_NAME</td>
<td>Harsh Dave</td>
</tr>
<tr>
<td>DV_STRAS</td>
<td>Genoisis Spire</td>
</tr>
<tr>
<td>DV_LIFNR</td>
<td>ZDAVE</td>
</tr>
</tbody>
</table>