A step by step guide for beginners on user defined BAPI creation

Introduction

A Business Application Programming Interface (BAPI) is a precisely defined interface providing access to processes and data in business application systems such as R/3. BAPIs are defined as API methods of SAP business object types. These business object types and their BAPIs are described and stored in the Business Object Repository (BOR). A BAPI is implemented as a function module, that is stored and described in the Function Builder.

BAPIs can be called within the R/3 System from external application systems and other programs. BAPIs are the communication standard for business applications. BAPI interface technology forms the basis for the following developments:

- Connecting:
- New R/3 components, for example, Advanced Planner and Optimizer (APO) and Business Information Warehouse (BW).
- Non-SAP software
- Legacy systems
- Isolating components within the R/3 System in the context of Business Framework
- Distributed R/3 scenarios with asynchronous connections using Application Link Enabling (ALE)
- Connecting R/3 Systems to the Internet using Internet Application Components (IACs)
- PC programs as frontends to the R/3 System, for example, Visual Basic (Microsoft) or Visual Age for Java (IBM).
- Workflow applications that extend beyond system boundaries
- Customers' and partners' own developments

BAPI is basically a RFC enabled function module. The difference between RFC enabled function module and BAPI is business objects. You create business objects and those are then registered in your BOR (Business Object Repository) which can be accessed outside the SAP system by using some other applications (Non-SAP) such as VB or JAVA. In this case you only specify the business object and its method from external
system. In BAPI there is no direct system call, while RFC are direct system call. Some BAPIs provide basic functions and can be used for most SAP business object types. These BAPIs should be implemented the same for all business object types. In short BAPI is a RFC enabled function module on the wrapper of Business object.

In this guide I have started with creating a structure in dictionary. Then I am creating a simple remote enabled function module. The function module fetches some details from Table T001 based on the input company code (BUKRS). Table T001 contains the Company Codes. The next step is creating a business object using SWO1 and we are adding the above function module as a method of that business object. After completing the steps in SWO1 we will find our created BAPI in BAPI Explorer. Then we will do the testing of BAPI by inputting some values.

Stage 1: Creating a structure in SE11

**Step 1:** Go to transaction SE11 and create a structure as shown or as per your requirement.

![Dictionary: Maintain Structure](image1.png)

Do not forget to save it in a package. Check the structure (ctrl + F2) and activate (ctrl + F3) the structure.

Stage 2: Creating the function module in SE37

**Step 2:** Go to transaction SE37 where you create function modules.

Click on create after you enter the name of the function module. Enter the function group and short text. Now click on save button.
Note: If function group is not already made, you could create it from Goto > Function group > Create Group from the initial screen of SE37.

Step 3: A message will be displayed ‘Function module name is reserved for SAP’. Just click continue.
Step 4: A screen will be displayed as below. In the attributes tab, make the processing type as ‘Remote-Enabled Module’.
Step 5: Now go to the Import tab and enter the details as follows or as per your requirement. Here I am giving the associated type as ZBAPISTRT001-BUKRS that we have created in previous stage.

Step 6: In the export tab enter the details as below.
Step 7: In the Tables tab enter the details as below.
Step 8: Now go to the source code tab and enter the code as shown below.
Step 9: Next we have to release the function module by Function Module>Release>Release.
Stage 3: Creating the business object in SWO1

Step 10: Go to transaction SWO1 and enter the name of the BAPI you would like to create and click create. Here I have given the name ‘ZBAPI_ABY’. In the coming screen Enter the details as below or as per requirement and click continue. That will ask you to save the object. Save it in a package.
Step 11: In the resulting screen click on the methods to drop down and see what methods are provided by default. There would be two methods, showing in red color which comes by default while creating the BAPI. Now select ‘Add Method’ from Utilities > API Methods > Add Method.
Step 12: On the screen that follows, provide the function module name that we just created in the previous stage and click on the continue icon. In the coming pop-up, click the next step icon. We observe that the information is predefined in the fields.
Step 13: This is the next screen where you would just click on the "next" icon.
Step 14: A popup will be displayed as below.
Click on Yes. You can see an information message reading “ZBAPIT001” inserted.

**Step 15:** Now save after you add the method. Select & Double click on the API method.

Go to Tab: ABAP Check ‘API Function’ as shown below.
Select the Radio button reading “API Function” as already said above.

**Step 16:** Now select the Object “ZBAP1_ABY” as shown below.
Step 17: Now Go to : Edit > Change Release Status > Object type > To Modeled.
Step 18: The below shown screen will be displayed. Click on yes.
The message shows, The object type status set to modeled. ( or already modeled)

Step 19: Now go to: Edit > Change Release Status > Object type > To Implemented. You can see a message reading “Object type status set to implemented”.

Step 20: Now, go to: Edit > Change Release Status > Object > To Released. There would be two pop ups coming up. Click continue on the Pop Ups.

Step 21: Keep the cursor on the 'Method'. Now go to: Edit > Change Release Status > Object type component > To Modeled. (Shown as below).
Step 22: Now, go to: Edit > Change Release Status > Object type component > To Implemented. You can see the message reading “status for method ‘zbapifmt001’ set to implemented”.

Step 23: Now go to: Edit > Change Releasse Status > Object type component > To Released. You can see the message reading “status for method ‘zbapifmt001’ set to Released”.

Step 24: Click on ‘Generate’ from Object Type as shown below.
After clicking on the generate button, you can see the message reading “Object type ‘ZBAPI_ABY’ generated successfully”. Now we are done with the creation of a BAPI.

**Stage 4: Viewing the created BAPI in BAPI Explorer**

**Step 25:** Go to BAPI Explorer (Transaction code is ‘BAPI’) there we can find the BAPI (our BAPI). You can click on the Alphabetical tab so that you can browse the BAPI’s in an alphabetical order. Find your BAPI as shown.
Stage 5: Test your BAPI

Step 26: Enter the name of your BAPI in the transaction SWO1 and click on ‘Test’.
Step 27: The below screen is displayed. Click on the Execute icon (circled in red) against the BAPI as shown.
Step 28: The below screen will be displayed where you would require entering the data against the empty input fields.
Step 29: You could use the input help as shown below; here I have selected ‘0001’. And click the execute button.
Step 30: The resulting screen is displayed below, where you can see ‘1 Entry’ (circled in red) against ITEMTAB. Double click on ‘1 Entry’.
The results as per our input are as shown below.
By this, we would get it confirmed that our BAPI is working properly. We can even check it by passing different values again. Come back to the input and execution screen.

We are now done with the creation and successful execution of a BAPI.