How To…

Pass dynamic parameters to script logic

SAP Product Name: SAP Business Planning and Consolidation 7.0, version for SAP NetWeaver
Applicable Product Versions: 7.0 SP1 & above
Document Version: 1.1
Last Updated: Jan 2009

Enterprise Performance Management

www.sdn.sap.com/irj/sdn/bpx-epm
1 Scenario

In this scenario, we will try to pass runtime parameters (called “prompts”) from data manager to script logic.

2 Introduction

There will be multiple scenarios where we want to get the user input and then, use that value in the script logic calculation. For Example, users may want to pass the percentage value of the budget increase when they execute the script logic, not hard-coded inside the script logic. As of Business Planning and Consolidation 7.0 Version for NetWeaver Support Package 1 (scheduled to be released on Dec 12 2008), we are supporting dynamic replacements inside script logic, which means values can be provided from data manager and will be replaced dynamically during runtime of your script file.

This how to guide walks through a step-by-step example in the ApShell Planning cube. Please note you must have appropriate NetWeaver BI and BPC authorizations to perform the required steps.
3 The Step By Step Solution

We will first create a custom process chain which will be used in the data manager as template. Then, we will build the script & finally, configure the data package to execute the script using the custom process chain created.

If you already have a process chain where you want to add this enhancement, just add new parameters from 22nd step in sub-section 3.1. Then, follow 3.2, 3.3 and 3.4.

3.1 Create BPC Process Chain

1. Log in to ABAP layer of your BPC through SAP Log on pad.

2. Go to transaction RSPC.
3. Click New.

4. Provide Process Chain tech name and description. (In typical NetWeaver BI implementations, custom objects usually begin with Z, but just follow your organization naming standards.) Click OK.

   In this example, we are calling the process chain “ZBPC_PROT_EXP” and the description “ZBPC: Prompt Example”.

5. System will prompt start variant. Create new start variant by clicking on new button.

In this example, we’re calling the process variant “ZBPC_PROMP_EXP_START” and the long description “ZBPC: Prompt Example Start”.

7. In the next screen, click on “Change Selection” to maintain the start time.

8. Click on “Immediate” button and press save button (in the bottom).
9. Press save and then go back.

10. Click OK.

11. Click save. In the whole process chain creation, remember to save your work for every step. Then, change the right hand side view to process type by clicking on the chain button.
12. Then, expand “BPC – Process Types”. Drag “BPC: Modify Dynamically” and drop in the right hand side area.

13. System will prompt to choose a variant. Create a new variant by clicking on the new button.

14. Provide Tech name and description and click OK.
15. Click save in the next screen and go back.

16. Click OK

17. System would have added the new variant in the right hand side.
18. Now, choose the first variant by clicking on it and then, drag & drop to the second variant to link them.

19. After successful link, system will show an arrow.

20. Now, drag and drop the “BPC: Run Logic” process.
21. Create new Run Logic step when prompted.

22. Provide Tech name and description for Run Logic step.

   In this example, specify the process variant as "ZBPC_PROMPT_EXP_RUN_LOGIC" and the long description as "ZBPC: Prompt Example Run Logic"

23. In the next screen, enter below fields:

   TAB
   SUSER
   SELECTION
   SAPPSET
   SAPP
   REPLACEPARAM
   LOGICFILENAME
   EQU

   (TAB, REPLACEPARAM and EQU are the new fields to support dynamic replacements of prompts)
24. Click Ok. Then, connect this process variant with previous step (i.e. “Dynamic Modify”).


26. Expand General Services in the right hand side and choose “OR” process.
27. Create a new OR process variant and click OK.

28. Then, connect this OR process with the “Dynamic Modify” step.

29. Choose Errors action this time.
30. Connect the same OR step with Run Logic step. When the popup box comes up, choose “always” as the Action.

31. Choose this process from left hand side and drop.

32. Create new variant.
33. Provide tech name and description.

34. In the next screen, add CLOSE_MODE as a field and give 0 as value. Press save and go back.

35. Then, connect this step with previous OR steps.
36. Save and activate the process chain by clicking on the candle button.

37. Now, we will be re-assigning this process chain to the right component so that it will be visible in BPC data manager. Click on “Display Components”

38. Choose BPC Examples component.
39. Save and activate again.

You don’t have to create new process chain for each script file you want to execute. You can pass the script file dynamically either in the data manager dynamic script (explained in section 3.3) of the package or prompt the user for the script file (explained in Appendix section).

Please note that currently you can create only one Data Manager package per process chain.
3.2 Develop the required script
In this step, we will develop the script with parameters which will replaced by data manager prompts.

1. Login to the BPC Administration.

2. Navigate to the AppSet/Application where you want to create the script Logic.
   In this case, we're working with a copy of ApShell.

1. In the action pane, click "Create New Logic".
2. Name the Logic File "IncreasePercentage" and click OK.

3. Enter the script logic code here. (Sample code is provided in the appendix). $WS_PERCT$ and $EXP_PERCT$ will be replaced at run time with prompt value. Use $ character before and after name to be replaced.

4. Save the script logic.
3.3 Develop the required script
In this step, we will create the data package to execute the script having the process chain created as template.

1. Login to the BPC Excel client.

2. After logging in, click on “Manage Data” in the Action pane.
3. Click on “Maintain data management” in the next menu.

4. Then, click on “Manage packages (organize list)”. 

[Image of a SAP interface]
5. Choose the desired team and click on “Create package” to add a new package.

6. Click on open folder button to select a process chain for this package.

7. Choose the process chain which you modified in the earlier step.
8. Enter Package name and Description (Avoid using special characters for naming the package, and choose a group as well. Make sure to select the desired task type as well. Then, click Save.

   In this case, we’re calling the package “Increase Exp and WS”

9. Next screen, click save to store the package in the system.

10. You will get a message that package list successfully updated.
11. Go back to “Manage packages (organize list)” to modify the package.

12. Click “view package” (button next to open folder on the same line as process chain).

13. Click on “Advanced” button.
14. Here, maintain the dynamic script (sample code is provided in the appendix). Then, click ok.

Note: There are two INFO statements here. The parameter “EQU” is used as the splitter between parameter and value (e.g. in A=B, EQU is just “=”). The parameter “TAB” is used to separate whole parameter & value sets (e.g. for A=B;C=D, the TAB is “;”). You can see how these two variables are used in the final TASK line of this dynamic script file.

15. Press Save in the next screen.

16. Press save again
17. Press save again. Now, we created the package.
3.4 Run the package and verify data

We will now execute the package and validate the data.

1. Log in to the excel client and pull in the data for CE0001000 account parent for 2006.SEP. You may also need to adjust other dimension filters as shown here in the current view.

2. In excel client, go back in the action pane to choose “Run a data management package”.

---

<table>
<thead>
<tr>
<th>App</th>
<th>CATEGORY</th>
<th>ENTITY</th>
<th>MEASURES</th>
<th>P_ACTIVITY</th>
<th>P_DATASRC</th>
<th>RPTCURREN LC</th>
<th>TIME</th>
</tr>
</thead>
<tbody>
<tr>
<td>PLANNING</td>
<td>ACTUAL</td>
<td>C9000</td>
<td>PERIODIC</td>
<td>ALL.ACT</td>
<td>UPLOAD</td>
<td>2006.SEP</td>
<td></td>
</tr>
<tr>
<td>ACTUAL</td>
<td>PC4You Holdings Inc</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Periodic Costs</td>
<td>Personal Costs</td>
<td>All activities / Drivers</td>
<td>Uploaded Data</td>
<td>Local Currency</td>
<td>2006.SEP</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

---

<table>
<thead>
<tr>
<th>Current View:</th>
</tr>
</thead>
<tbody>
<tr>
<td>Application: PLANNING</td>
</tr>
<tr>
<td>Category: ACTUAL</td>
</tr>
<tr>
<td>Entity: C9000</td>
</tr>
<tr>
<td>P_ACCT: CE000000</td>
</tr>
<tr>
<td>P_Activity: ALL.ACT</td>
</tr>
<tr>
<td>P_DataSrc: UPLOAD</td>
</tr>
<tr>
<td>RptCurrency: LC</td>
</tr>
<tr>
<td>Time: 2006.SEP</td>
</tr>
<tr>
<td>MEASURES: PERIODIC</td>
</tr>
</tbody>
</table>

---

<table>
<thead>
<tr>
<th>Session Information</th>
</tr>
</thead>
<tbody>
<tr>
<td>Logon: 1922146 - APSHELL_V</td>
</tr>
<tr>
<td>CV: PLANNING- ACTUAL- H1 -</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Manage Data Options</th>
</tr>
</thead>
<tbody>
<tr>
<td>Upload data file</td>
</tr>
<tr>
<td>Download data file</td>
</tr>
<tr>
<td>Preview data file</td>
</tr>
<tr>
<td>Run a data management package</td>
</tr>
<tr>
<td>View package status</td>
</tr>
<tr>
<td>View schedule status</td>
</tr>
<tr>
<td>Maintain conversions</td>
</tr>
<tr>
<td>Maintain transformations</td>
</tr>
<tr>
<td>Maintain data management</td>
</tr>
</tbody>
</table>

---

<table>
<thead>
<tr>
<th>Available Interfaces</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
</tr>
</tbody>
</table>
3. Place the cursor on the package created in the earlier step and click “Run”.

4. Entered desired values

5. Fill in the data ranges as necessary and choose when you want to run the package. Then, click Next.
6. Confirm the selection and click Finish.

7. Click OK on the message and verify the logs.

8. Now, click on the View Status check the log of the script.
9. You can see your job running with green arrow.

10. Once successfully completed, changes to blue color icon with completed as Progress. Double click on this line to see the details of the log.

11. Here you can view the whole log.
12. Close the data manager to refresh the EVDRE query. Now, you can see the changed data.

Note that this dynamic replacement can also be used to pass multiple member values to the script (you would pass one long string with all the values as you would type in the script logic). Furthermore, you could even combine these parameters with the use of the START BADI keyword in script logic, and pass parameters all the way from the user input through to your custom written BAdI.
4 Appendix

4.1 Sample Source Code for Script Logic file:

```plaintext
*XDIM_MEMBERSET P_ACCT= CE0004020, CE0004010
*XDIM_MEMBERSET CATEGORY = ACTUAL
*XDIM_MEMBERSET TIME=2006.SEP
*XDIM_MEMBERSET ENTITY= C9000
*XDIM_MEMBERSET P_ACTIVITY=NONE
*XDIM_MEMBERSET P_DATASRC=UPLOAD
*XDIM_MEMBERSET RPTCURRENCY = LC

//Increase Wage and Salary & Personnel Exp. by entered percentage
[P_ACCT],[#CE0004020] = [P_ACCT],[CE0004020] * ( 1 + $WS_PERCT% / 100)
[P_ACCT],[#CE0004010] = [P_ACCT],[CE0004010] * ( 1 + $EXP_PERCT% / 100)
*COMMIT
```

Note the $ sign at the beginning and end of the variables.

4.2 Sample Source Code for dynamic script in the package:

```plaintext
PROMPT(SELECTINPUT,...,%ENTITY_DIM%,%CATEGORY_DIM%,%CURRENCY_DIM%,%TIME_DIM%)
PROMPT(TEXT,%WS_PERCT%,"Input W/S Percent in decimals");
PROMPT(TEXT,%EXP_PERCT%,"Input Exp. Percent in decimals");
INFO(%EQU%,=)
INFO(%TAB%,;)
TASK(ZBPC_PROMPT_EXP_RUN_LOGIC,TAB,%TAB%)
TASK(ZBPC_PROMPT_EXP_RUN_LOGIC,EQU,%EQU%)
TASK(ZBPC_PROMPT_EXP_RUN_LOGIC,SUSER,%USER%)
TASK(ZBPC_PROMPT_EXP_RUN_LOGIC,SAPPSET,%APPSET%)
TASK(ZBPC_PROMPT_EXP_RUN_LOGIC,SAPP,%APP%)
TASK(ZBPC_PROMPT_EXP_RUN_LOGIC,SELECTION,%SELECTION%)
TASK(ZBPC_PROMPT_EXP_RUN_LOGIC,LOGICFILENAME, INCREASEPERCENTAGE.LGF)
TASK(ZBPC_PROMPT_EXP_RUN_LOGIC,REPLACEPARAM,WS_PERCT%EQU%%WS_PERCT%%TAB%EXP_PERCT%EQU%%EXP_PERCT%)
```

REPLACEPARAM in the last TASK basically builds a look up table with variable name & its value entered by the users. During execution of the script logic, wherever the variables appear in the script, they will first replaced by the user entered value, before execution of the script.

4.3 Data Manager dynamic script to prompt for Script Logic file:

```plaintext
PROMPT(SELECTINPUT,...,%ENTITY_DIM%,%CATEGORY_DIM%,%CURRENCY_DIM%,%TIME_DIM%)
PROMPT(TEXT,%SCRIPT_FILE%,"Choose Script Logic File");
PROMPT(TEXT,%WS_PERCT%,"Input W/S Percent in decimals");
PROMPT(TEXT,%EXP_PERCT%,"Input Exp. Percent in decimals");
INFO(%EQU%,=)
INFO(%TAB%,;)
TASK(ZBPC_PROMPT_EXP_RUN_LOGIC,TAB,%TAB%)
TASK(ZBPC_PROMPT_EXP_RUN_LOGIC,EQU,%EQU%)
TASK(ZBPC_PROMPT_EXP_RUN_LOGIC,SUSER,%USER%)
TASK(ZBPC_PROMPT_EXP_RUN_LOGIC,SAPPSET,%APPSET%)
TASK(ZBPC_PROMPT_EXP_RUN_LOGIC,SAPP,%APP%)
TASK(ZBPC_PROMPT_EXP_RUN_LOGIC,SELECTION,%SELECTION%)
TASK(ZBPC_PROMPT_EXP_RUN_LOGIC,LOGICFILENAME,%SCRIPT_FILE%)
TASK(ZBPC_PROMPT_EXP_RUN_LOGIC,REPLACEPARAM,WS_PERCT%EQU%%WS_PERCT%TAB%EXP_PERCT%EQU%%EXP_PERCT%)
```

Adding Prompt for script logic file (shown in bold) will prompt the users to choose a script logic file which they want to execute.