Create Simple Transformation for XML in ABAP

Indeed, even in its most early frame, this less known wizard of basic change, could be truly valuable for ABAP designers. I think, ABAP designers would love to have a graphical utility to create basic change/XSLT code. I was writing some glad-code to consume an external web service from ABAP. As the web service was returning the result data in the form of an XML string, I wound up composing a Simple Transformation to get the qualities in ABAP inner table.

Consuming webservice from ABAP is Xn SE80->Repository Browser->Enterprise Services->[ right click to bring the context menu ] -->Create Proxy [ use WSDL URL e.g. http://www.webservicex.net/uklocation.asmx?WSDL to generate proxy through wizard ]. Xn LPCONFIG to configure the logical port.

I detected this enchantment stick in Simple Transformation editorial manager [ Transaction XSLT_TOOL ] and attempted to make it work. Be that as it may, couldn't unravel and composed the ST physically. Strangely, I couldn't discover any data about this alleged "Alter Simple Transformation Graphically" include. With all due respect, I looked on SAP OSS, SAP Help and SDN. There you go ...Well hunt include on SAP Sites.. is awesome in any case :)

After a year, I coincidentally was venerating ( designers do this regularly ) my old code and it helped me to remember this utility. I gave figuring a shot by and by and figured out how to find something worth sharing. Learn SAP ABAP here.

Service proxy call provides data in the form of XML result. Code for proxy call is as below:

```
CREATE OBJECT LO_UK_LOCATION. TRY. 
  lv_input-POST_CODE = p_post. 
  CALL METHOD 
  LO_UK_LOCATION->GET_UKLOCATION_BY_POST_CODE 
  EXPORTING INPUT = lv_input 
  IMPORTING OUTPUT = lv_output. 
  CATCH CX_AI_SYSTEM_FAULT . 
  WRITE : 'CX_AI_SYSTEM_FAULT'. 
  CATCH CX_AI_APPLICATION_FAULT . 
  WRITE : 'CX_AI_APPLICATION_FAULT'. 
ENDTRY.
```

Now the result variable `lv_output` will be having the data in the form of an XML string as below.

```
<NewDataSet>
  <Table>
    <Town>Balthangie</Town>
    <County>Aberdeenshire</County>
    <PostCode>AB53</PostCode>
  </Table>
  <Table>
    <Town>Bankhead</Town>
    <County>Aberdeenshire</County>
    <PostCode>AB51</PostCode>
  </Table>
</NewDataSet>
```

In order to transform the XML to ABAP internal table, I decided to use Simple Transformation. Now I will generate the simple transformation as below:

1. Create Table Type ZNEWDATASET in transaction SE11. The hierarchy structure should be compatible with the XML hierarchy.

   ![SE11_abap_structure.bmp](http://4.bp.blogspot.com/_agslXecgV60/SYm-dD_ALjI/AAAAAAAAAOk/ZYNsyZCY9rc/s400/se11_abap_structure.bmp)

2. Now create the ST program in transaction XSLT_TOOL. Spot the magic stick in the editor menu.

   ![Edit_ABAP_ST_Graphically.bmp](http://2.bp.blogspot.com/_agslXecgV60/SYnCVvZo-PI/AAAAAAAAAOs/6Ls41Ow9MHk/s400/Edit_ABAP_ST_Graphically.bmp)

3. Click on 'Edit Simple Transformation Graphically' button (magic stick) and you will get the editor as below. Create a new root (right click - context menu ) in "Data Roots" section. I've named it NEWDATASET but the important thing is to enter the correct type, created earlier.

   ![Data_Roots.bmp](http://1.bp.blogspot.com/_agslXecgV60/SYnDkmN__yI/AAAAAAAAAO0/03omYgw0RHk/s400/Data_Roots.bmp)

4. Now you can see the data root hierarchy as below. Drag and Drop the NEWDATASET root on right panel ( Simple Transformation ). It automatically generates the ST nodes corresponding to the ABAP structure. However, you need to adjust the names as per the XML element names.

   ![drag_drop_structure_to_st.bmp](http://1.bp.blogspot.com/_agslXecgV60/SYnE0KQ1teI/AAAAAAAAAO8/_qiDiPCZi7M/s400/drag_drop_structure_to_st.bmp)

5. Now adjust the names in ST panel as per actual XML element names.

   ![adjust_name_of_st_elements.bmp](http://1.bp.blogspot.com/_agslXecgV60/SYNFnErN-I/AAAAAAAAAPE/RWSmxJXbRQ/s400/adjust_name_of_st_elements.bmp)

6. Save and Activate. The ST program (Z_TEST), generated by the utility, is as below:

```
<NewDataSet>
  
  <Table>
    <Town>Balthangie</Town>
    <County>Aberdeenshire</County>
    <PostCode>AB53</PostCode>
  </Table>
  
  <Table>
    <Town>Bankhead</Town>
    <County>Aberdeenshire</County>
    <PostCode>AB51</PostCode>
  </Table>

</NewDataSet>
```
7. Now use the code below to get data into ABAP internal table.

data : result type ZNEWDATABASE, "itab of table type ZNEWDATABASE wa_result like line of result.
call transformation Z_TEST source xml lv_output-GET_UKLOCATION_BY_POST_CODE_R result newdataset = result[].
LOOP AT result into wa_result. WRITE :/ wa_result-TOWN, wa_result-COUNTY, wa_result-POST_CODE. ENDLOOP.

Source: SAPBlog