Visual Composer Modeling: UI Modeling with Clustered Data

Applies to:

Visual Composer for SAP NetWeaver Composition Environnent (CE) 7.1.1. For more information, visit the Portal and Collaboration homepage.

Summary

This document attempts to explain how the data model in Visual Composer works, and how to properly connect your visual elements to data

Author: Eyal Allweil

Company: SAP

Created on: 23 July 2009

Author Bio

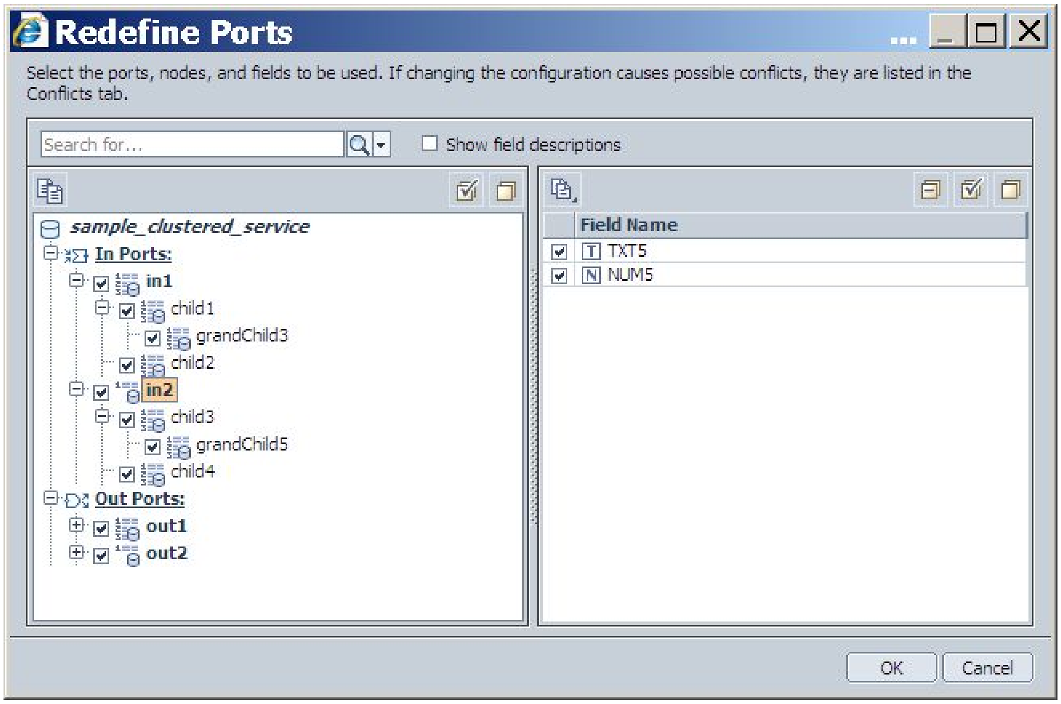
Eyal Allweil is a developer working in the Visual Composer Development group.

Table of Contents

Introduction

This document attempts to explain how the data model in Visual Composer works, and how to properly connect your visual elements to data. It is meant as a companion document to How to Model a Clustered and Complex Data Service in Visual Composer by Netanel Slomianksi. In this document, I occasionally refer to differences between Enhancement Pack 1 and Enhancement Pack 2, but all the screenshots and instructions refer to Enhancement Pack 1.

For the purpose of simplicity, this document uses a different service than the one described in Netanel's document. The ports used are very similar to one another; however, *in1* has a record set as root, whereas *in2* has a record. In addition, *in1* and *out1* have the same structure, as do *in2* and *out2*. Here is a picture of the service as seen in the Redefine Ports dialog.



Redefine Ports

Select the ports, nodes, and fields to be used. If changing the configuration causes possible conflicts, they are listed in the Conflicts tab.

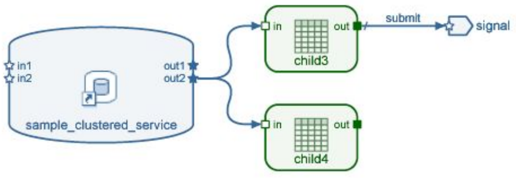
Search for... ☐ Show field descriptions

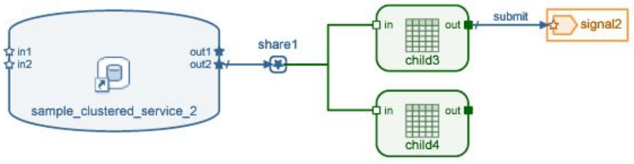
- *sample_clustered_service*
  - In Ports:
    - ☑ in1
      - ☑ child1
        - ☑ grandChild3
      - ☑ child2
    - ☑ in2
      - ☑ child3
        - ☑ grandChild5
      - ☑ child4
  - Out Ports:
    - ☑ out1
    - ☑ out2

	Field Name
☑	T TXT5
☑	N NUM5

Flow, Bind and Map Links

The first thing we will do is to understand the differences between links. Examine the following pictures. What is the difference between the left and right?

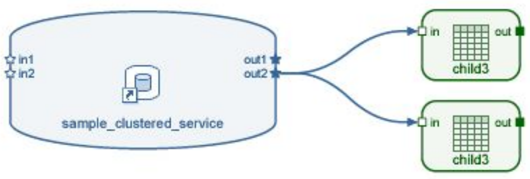


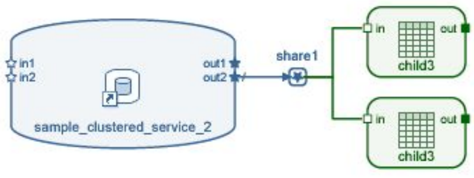


On the left, the views are connected to the service by flow links. This means that the data in *out2* is copied twice; once to *child3* and once to *child4*. If we trigger the *submit* event to *signal*, only the changes that have been made in *child3* will be sent. Those in *child4* will be lost.

On the right side, *child3* and *child4* are connected to *share1* by bind links, so no additional copying is done. When the *submit* event is triggered, the changes made in both *child3* and *child4* will be sent. Note also that *share1* is connected to the service by a map link. This means that *share1* may have different metadata than *out2*.

Whether or not views have their own copy of data also affects the selected row. In the left side of the picture below, the tables for *child3* can have the different rows selected, whereas on the right side, changing the selection in one table will change it in the other.





Another difference between links is their *select mode* property. This property allows a link to transfer only the selected row(s) instead of all the rows in their source.

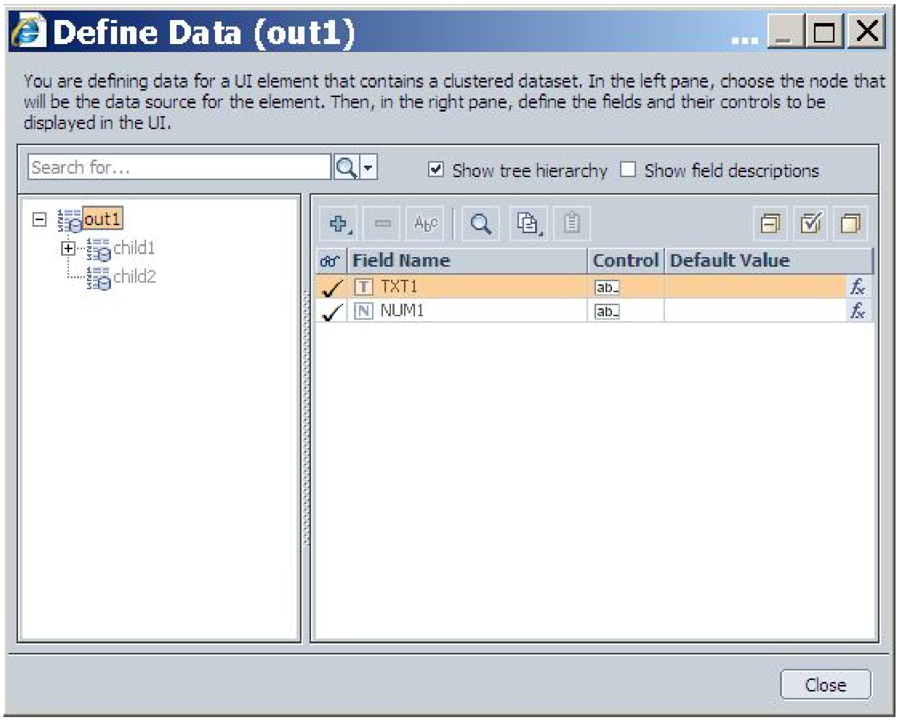
The following table sums up the differences between links:

Link Name	Can Change Structure	Duplicates Data in Runtime	Has Select Mode	Shared Cursor
Data Map	Yes	Yes	Yes	No
Data Flow	No	Yes	Yes	No
Data Bind	No	No	No	Yes

Setting a View's Context Node

A view bound to clustered data must have a context node, which determines (among other things) the data node that the view will eventually display in runtime. The context node can be seen in the configure panel (labeled *source node*) and in the Define Data dialog. In Enhancement Pack 1, the context node appears in the Define Data dialog as the highlighted node in the hierarchy tree. In Enhancement Pack 2, the context node is indicated by a small icon in the hierarchy tree.

If we create a table by dragging a line out of *out1*, the Define Data dialog will be opened, as seen below:



Define Data (out1)

You are defining data for a UI element that contains a clustered dataset. In the left pane, choose the node that will be the data source for the element. Then, in the right pane, define the fields and their controls to be displayed in the UI.

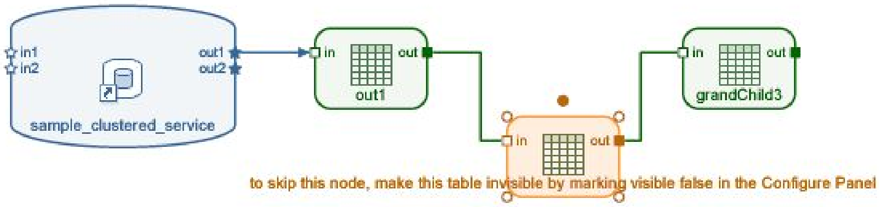
Search for...

☑ Show tree hierarchy ☐ Show field descriptions

	Field Name	Control	Default Value
✓	TXT1	ab_	
✓	NUM1	ab_	

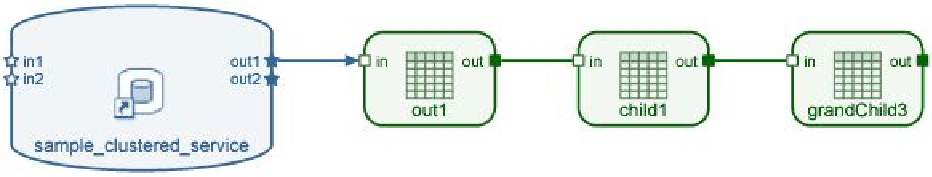
In Enhancement Pack 1, you change the context node by clicking on a different node in the tree (in Enhancement Pack 2, this has been changed; instead, you right-click and choose *Select as Context).* It is only possible to descend one level in the hierarchy at a time, which is why the nodes *child1* and *child2* are disabled. However, if we had dragged a table out of *out2* we would have been able to skip the root node, because it is a record and not a record set.

Typically, it doesn't make sense to skip record sets. A nested table belongs to a certain row in its parent table; it does not exist independently of it. For example, displaying a record set of phone numbers is only meaningful if you know whose phones they are. However, sometimes a service will return a node of type record set, but the modeler knows that in runtime this node will only have one row. In such a case, it may make sense to skip this node. In Enhancement Pack 2, you can skip nodes freely in the Define Data dialog. In Enhancement Pack 1, we need to use an invisible table, as in the following picture.



to skip this node, make this table invisible by marking visible false in the Configure Panel

One restriction present in both versions of Visual Composer is that you cannot go up in the node hierarchy. For example, in the following diagram, it would not be possible to change the context node of *grandChild3* to be *out1*, because *out1* is higher up in the hierarchy than *child1*.



Binding Controls to Fields

Typically, the controls in a view are bound to fields in that view's context node. But sometimes we want to bind a control to a field in a different node. In order to do this, we need to select the field from the Layout board and double-click the *value* property in its Configure task panel. This will open the Dynamic Expression dialog, which allows us to pick the field we want from a different element. Whenever you pick a field from a different element in this manner, the value of the field used (in runtime) is that of the selected row.

In Enhancement Pack 2, it is possible to browse other nodes in the Define Data dialog and create/bind controls to them exactly as you would on the context node.

Conclusions

In this article, I have tried to show how Visual Composer's data model works in runtime and some of the differences between Enhancement Packs 1 and 2. I have also explained how to work around some of the restrictions in Enhancement Pack 1.

Related Content

How to Model a Clustered and Complex Data Service in Visual Composer

Visual Composer 7.1 SDN Forum

For more information, visit the Portal and Collaboration homepage.

Copyright

These materials are subject to change without notice. These materials are provided by SAP AG and its affiliated companies ("SAP Group") for informational purposes only, without representation or warranty of any kind, and SAP Group shall not be liable for errors or omissions with respect to the materials. The only warranties for SAP Group products and services are those that are set forth in the express warranty statements accompanying such products and services, if any. Nothing herein should be construed as constituting an additional warranty.