



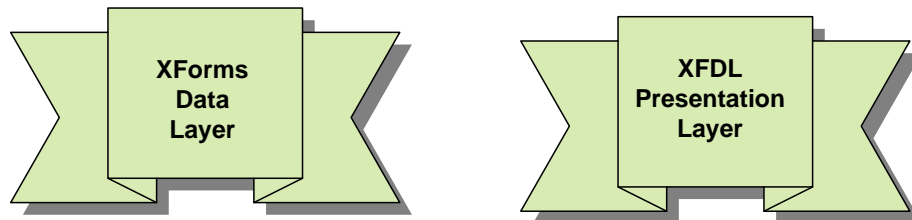
Which takes precedence: XForms binds or XFDL options?



© 2008 IBM Corporation

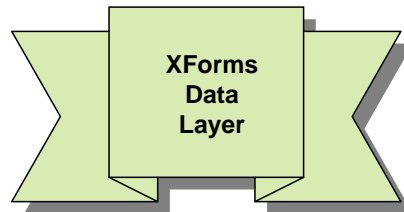
This module describes how to determine whether XForms binds or XFDL options take precedence when working with a form object.

Data and presentation layer



When discussing the relevance between XForms model item properties and XFDL options, it helps to think of them as data layer and presentation layer where XForms is the data layer and XFDL is the presentation layer.

Data layer



```
<xforms:bind nodeset="instance('data')/fieldText1" readonly="true()"></xforms:bind>
```

In this `xforms:bind` the XForms data layer is locked by the `"true()"` value of the `readonly` model item property.

Presentation layer

```
<field sid="FIELD1">  
  <xforms:input ref="instance('data')/fieldText1">  
    <xforms:label>FIELD1: Read/Write - [readonly MIP = true()],  
[readonly tag = off]</xforms:label>  
  </xforms:input>  
  <readonly>off</readonly>  
</field>
```

Setting the readonly option of the XFDL presentation layer to “off” does not override the XForms readonly model item property. The XFDL presentation layer cannot make changes to the XForms data layer.

Example forms

- Technote # [1255009](#)
- Example 1: readOnly.xfdl
- Example 2: mandatoryOptionalPrecedence.xfdl
- Example 3: hiddenVisiblePrecedence.xfdl
- Example 4: nonRelevantExample.xfdl

Example forms discussed here can be found in technote # 1255009 located at the listed link.

- Example 1 - readOnly.xfdl
- Example 2 – mandatoryOptionalPrecedence.xfdl
- Example 3 – hiddenVisiblePrecedence.xfdl
- Example 4 – NonRelevantExample.xfdl

Example 1: readOnly.xfdl

Data model

```
<xforms:instance id="data" xmlns="">
  <fieldText1>test Info 1</fieldText1>
  <fieldText2>test Info 2</fieldText2>
  <fieldText3>test Info 3</fieldText3>
  <fieldText4>test Info 4</fieldText4>
  <fieldText5>test Info 5</fieldText5>
  <fieldText6>test Info 6</fieldText6>
  <data>
    <fieldText1>test Info 1</fieldText1>
  </data>
</xforms:instance>
```

Nodeset bind

```
<xforms:bind nodeset="instance('data')/fieldText1" readonly="true()"></xforms:bind>
```

To demonstrate readonly, you start with the data model of an XForms form. To create an indirect link between the field in the form and a data element in the XForms model, use a nodeset bind. Within the bind the readonly model item property (MIP) is set to 'true()'.

Readonly field

```
<field sid="FIELD1">  
  <xforms:input ref="instance('data')/fieldText1">  
    <xforms:label>FIELD1: Read/Write - [readonly MIP = true()], [readonly tag =  
    off]</xforms:label>  
  </xforms:input>  
  <readonly>off</readonly>  
</field>
```

In the form “FIELD1”, the readonly option is set to “off”. Because the model item property is set to “true()” within the XForms bind, the data layer is locked. Once the data layer is locked by the readonly model item property, the presentation layer XFDL field option cannot make changes to the data model.

Solution for readonly fields



```
<xforms:bind nodeset="instance('data')/fieldText1"
  readonly="true()"></xforms:bind>
```

```
<field sid="FIELD1">
  <xforms:input ref="instance('data')/fieldText1">
    <xforms:label>FIELD1: Read/Write - [readonly MIP = true()], [readonly tag =
    off]</xforms:label>
  </xforms:input>
  <readonly>off</readonly> // Remove readonly option
</field>
```

Here Field1 appears to be duplicating the readonly option by setting the readonly option to “on”. Because the XForms bind has set the model item property to “true()”, this sets this item to readonly and locks the datalayer. The XFDL <readonly> is disregarded because the data layer is now locked. When creating a form with XForms data driven principles in mind, the correct way to make the field readonly is to set the model item property to “readonly=true()” and remove the XFDL readonly option from the field. It is not necessary to use both. A rule of thumb is that when using XForms, you should use the xforms bind. And when using XFD, use the element options.

Example 2: mandatoryOptionalPrecedence

```
<xforms:bind id="Field2" nodeset="instance('INSTANCE')/PAGE1/FIELD2"
  required="false()">
</xforms:bind>
```

► Field option:

```
<constraints>
  <mandatory>on</mandatory>
</constraints>
```

9

© 2008 IBM Corporation

By default the data layer XForms model item property (MIP) "required" is set to "false()", therefore, stating required=false() is not necessary. Because the data layer MIP is "not required", the data layer is not locked.

```
<xforms:bind id="Field2" nodeset="instance('INSTANCE')/PAGE1/FIELD2"
required="false()"></xforms:bind>
```

Opening the sample form and viewing "FIELD2", it appears that the XFDL presentation layer has overridden the data layer. The XFDL presentation layer cannot override the XForms model item property. Instead what happens is that, according to the XForms model, the data for this field is not mandatory. However, the XFDL mandatory constraint changes the presentation indicating that the data is mandatory.

```
<constraints>
  <mandatory>on</mandatory>
</constraints>
```

In this example these properties are in direct conflict of each other. It is not a best practice to mix the XForms model item property required and XFDL constraint mandatory. Ideally the form creator should use only the XForms model item property and remove the XFDL constraints from the field.

Example 3: hiddenVisiblePrecedence.xfdl

Presentation layer

```
<visible>on</visible> //remove from the field presentation layer
```

Data layer

```
<xforms:bind id="field2"
  nodeset="instance('INSTANCE')/PAGE1/FIELD2"
  relevant="false()"></xforms:bind> //set the
  relevant="true"
```

10

© 2008 IBM Corporation

In the "hiddenVisiblePrecedence" example, the 'FIELD2' XFDL <visible> is set to "on" and appears to have overridden the data layer.

<visible>on</visible>

The model item property (MIP) relevant, by default, is set to "false()" and therefore does not lock the data layer allowing the XFDL to make the change.

```
<xforms:bind id="field2" nodeset="instance('INSTANCE')/PAGE1/FIELD2"
  relevant="false()"></xforms:bind>
```

To correct this form and set visibility to "on", the form creator must set the model item property relevant to "true()" and remove the XFDL visible option from the field.

To further discuss the model item property "relevant":

XForms Relevant - defines an XPath formula with a Boolean result that helps determine whether a node is relevant to processing. By default, the GUI widgets bound to non-relevant nodes are invisible and inactive (unless overridden by the appropriate XFDL options), and they are not submitted.

Example 4: nonRelevantExample.xfdl

```
<xforms:instance xmlns="" id="Visible">  
  <xforms:instance xmlns="" id="Active">
```

Button 1:

```
<xforms:setvalue ref="instance('Visible')/visible"  
value="if(..visible='true', 'false', 'true')"></xforms:setvalue>
```

Button 2:

```
<xforms:setvalue ref="instance('Active')/active"  
value="if(..active='true', 'false', 'true')"></xforms:setvalue>  
<xforms:setvalue ref="instance('Active')/data11"  
value="if(..active='false', "", "")"></xforms:setvalue>
```

The final example (NonRelevantExample.xfdl) demonstrates how the non-relevant nodes are overridden by the XFDL options. There are two instances: Visible and Active. In the first button, the xforms:setvalue value is changed by an “if” statement that when <visible> is true() then make it false(), otherwise it is true(). The second button uses the same concept to set visible and active, and clears the field when the field is not active.

Conclusion

- Use model item properties rather than XFDL options
- Exception “relevant”

When using XForms data driven forms, it is recommended to use only the XForms model item properties. By adding the XFDL options to the form along with the model item properties, the form designer risks a result that might not be expected. Proper form design dictates that when using XFDL, then use only the element options. When using XForms, use only XForms model item properties with the exception of relevant, where if set to false() or non-relevant, the form creator can change the visible or active state.

Feedback

Your feedback is valuable

You can help improve the quality of IBM Education Assistant content to better meet your needs by providing feedback.

- Did you find this module useful?
- Did it help you solve a problem or answer a question?
- Do you have suggestions for improvements?

Click to send e-mail feedback:

mailto:iea@us.ibm.com?subject=Feedback_about_1255009.ppt

This module is also available in PDF format at: [./1255009.pdf](http://1255009.pdf)

You can help improve the quality of IBM Education Assistant content by providing feedback.