



QDABRA DATABASE ACCELERATOR V2.3

How to create a button that saves but does not close the form

In the [Getting Started](#) document (also included with every installation of DBXL v2.3), we illustrate how to create a button that submits to DBXL. However, that button is configured to close the form upon submit.

How can you configure your form to save and NOT close the form? This is what some customers might call “save your progress”, before submitting. Let’s test by using the form called SampleForm.xsn, included in this download. You can open it in Design mode to analyze the data connection and its two buttons. Using the XSN, create a new document type in the DBXL Administration Tool, click **Open**, then:

1. Enter data and click **Save and Continue**.
2. Make some changes and click **Save and Continue**.
3. Make some changes and click **Submit and close**.

Our result might surprise you:

Documents							Refresh
		Delete		Delete All		Reshred	
						Reshred All	
Doc. ID	Ref. ID	Version #	Name	Author	Description	New	
383	5	1	1	1	1	Open	
384	6	1	2	2	2	Open	
385	7	1	3	3	3	Open	

We end up with three documents! Are we doing something wrong?

When we submit an xml document, it is saved to DBXL, but the form (open on your machine) does not receive the Qdabra Processing Instructions (which will identify the xml as already-existing and thus prompt an overwrite instead of a duplicate).

When the form is closed after submit (as in the Getting Started document), the user is forced to re-open the document. When the xml is retrieved from DBXL, it comes with the Processing Instructions, so further submits overwrite the existing document. This is why the scenario in the Getting Started document works.

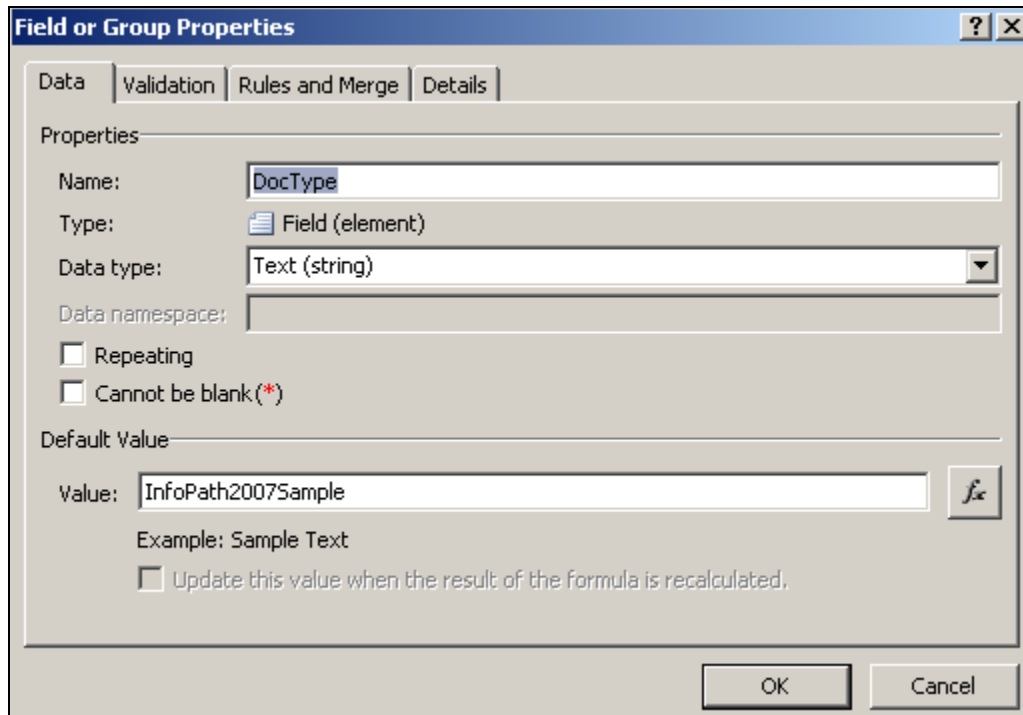
So the question remains, how do we create a button to “save and leave the form open”?

Unfortunately, in DBXL v2.3 this requires code. Included in this download you’ll find code samples for InfoPath 2003 and InfoPath2007. When using this code, you’ll need to watch out for two things:

- If you want to use these code samples in a browser-enabled for, you’ll need to remove the portions of the code that call message boxes, which are incompatible with IPFS.
- To use your own document type name, you’ll need to change the hardcoded value in two places:



1. You'll need a DocType node and you'll need to set its value to the name of your Document Type in the DBXL Administration Tool.



The image shows a screenshot of the 'Field or Group Properties' dialog box. The 'Data' tab is selected. The 'Name' field is set to 'DocType'. The 'Type' is 'Field (element)'. The 'Data type' is 'Text (string)'. The 'Data namespace' is empty. The 'Repeating' checkbox is unchecked. The 'Cannot be blank (*)' checkbox is unchecked. The 'Default Value' section shows a 'Value' of 'InfoPath2007Sample' and an 'Example' of 'Sample Text'. There is a checkbox for 'Update this value when the result of the formula is recalculated.' which is unchecked. The 'OK' and 'Cancel' buttons are at the bottom right.

2. The value of the docType variable in the code must also hold the same value as the doctypename in the DBXL Administration Tool:

***//doctype should be the name of the document type configuration from DAT.
string docType = "InfoPath2007Sample";***