

User's Guide

QuickForm Version 2.0

Form Builder for Visual Basic

Martin Massé

Introduction

Welcome to QuickForm 2.0, the quickest and easiest way to create forms for Microsoft® Visual Basic 3.0™ . QuickForm allow you to create forms and code from Access Tables or Queries (Version 1.0, 1.1, 2.0). using a template file.

QuickForm makes you more productive by providing tools to help you generate forms, controls, and the code to make them work. You build your template file using VisualBasic. You set properties on objects you put on the form or the form itself. Then you make this form work by creating the code in the same template file using VisualBasic or a text editor. Using QuickForm, you can create powerful, full-featured forms, using your code architecture and custom control you own.

Contents:

- Setting Up
- Getting Started

Setting Up

You install QuickForm on your computer by decompressing the ZIP file in a directory of your choice. The Zip file contains the following files:

Files	Directory
QKFORM.EXE	
QKFORM.HLP	[NOT IMPLEMENTED YET]
QKFORM.INI	Same as QKFORM.EXE
QFGUIDE.DOC	
DEFAULT.QFT	
MLIST.VBX	Same as QKFORM.EXE or System Directory
MCMBOX.VBX	Same as QKFORM.EXE or System Directory
VSVBX.VBX	Same as QKFORM.EXE or System Directory

Getting Started

Form and Controls Template

First, create a new project or open an existing project in Visual Basic.

Create a new form.

Add controls you want to use for generating the form. (Ex. TextBox, CheckBox, VSElastic, SSPanel3D or any control you have). Include only one control of each class. If you include more than one control in a class, the second one in the form definition will be ignored.

WARNING!!! You have to put at least one TextBox, one Label, one CommandButton and one DataControl. If you don't, the form will not be generated properly.

You can now set properties for all the objects. (Ex. Font properties, size, 3D, or any properties your controls have). The Name property of each object can be use as prefix for the controls generated by QuickForm. (Ex. If you set the name property of the TextBox to *txt*, and if you choose the Use Name as Prefix option in the Options form, a TextBox generated by QuickForm will have this name property *txtCustomerID*. The same technique is applicable for forms.)

Code Template

Next, you can write code to be added to your form and control. There are the techniques used to do this. See DEFAULT.QFT for an example.

This code has to be added in the same template file as the form and controls template, like a real VisualBasic form file.

[Declaration]

The declaration section of a form is found immediately after the form definition. If you want to add code in the declaration, you just have to add it here.

[Sub/Function at form level]

Add your Function and Sub anywhere after the declaration section. If there is no declaration section, write it after the form definition. If you have to perform an action for all fields in a Sub or Function, you can use the QF Script Language. Example, you want to clear all the fields using a Sub call *ClearAllFields*. Below is the code needed to do this:

Add this script to your template file:

```
Sub ClearAllFields()  
    DO_FOR TextBox  
        %CONTROLNAME% = ""
```

```

_LOOP_
DO_FOR CheckBox
%CONTROLNAME% = FALSE
_LOOP_
End Sub

```

Suppose that you have three fields, two TextBox named *txtCustomerID* and *txtName*, and a CheckBox named *chkYes*. The code added to the form will look like this:

```

Sub ClearAllFields()
    txtCustomerID = ""
    txtName = ""
    chkYes = False
End Sub

```

Suppose you want to update this field with the value in the current record, you have to write something like this:

```

Sub UpdateFields()
DO_FOR TextBox
%CONTROLNAME% = Data1.Recordset("%FIELDNAME%")
_LOOP_
End Sub

```

This code result in:

```

Sub UpdateFields()
txtCustomerID = Data1.Recordset("CustomerID")
txtName = Data1.Recordset("Name")
End Sub

```

You know have two **variables**,

%CONTROLNAME% = Name of the control in the form
 %FIELDNAME% = Corresponding field in the table or query

and a script statement DO_FOR *ControlClass* _LOOP_

The *ControlClass* argument can be any control class of control you generate in the form except control class used as labels and buttons.

WARNING!!! You cannot use DO_FOR inside an other DO_FOR statement.

[Code at control level]

To add code to control level, you are going to use the same techniques used previously. Suppose

you want TextBox to toggle font bold when they got focus and to get back FontBold = False when a LostFocus event occurs:

DO_FOR *TextBox*

Sub %CONTROLNAME%_GotFocus()

Me.%CONTROLNAME%.FontBold = True

End Sub

Sub %CONTROLNAME%_LostFocus()

Me.%CONTROLNAME%.FontBold = False

End Sub

LOOP

You can use this technique for all class of control except class uses as buttons, class uses as label and form class. To add code to buttons, do the following.

[Buttons]

First, you have to determine if you used **Name As Prefix** options, and in this case , what is the prefix used.

Next, write the code using the right name. Example, to add code to the New button.

Sub *prefix*New_Click()

Data1.Recordset.AddNew

End Sub

WARNING!!!

QuickForm does not detect errors in script statement. Be sure to write it without mistake.

Now, you can save the form as **Name.QFT (QuickForm Template)**. Your template for the form and control is now completed.

Options

Open QuickForm.

Press the Options button to open the Options window.

You can change the **Default Template File** you want to use. In this case we suppose that you want to use the Template file you have just created. Press the Select button and select it.

The **Use Name As Prefix** option is checked by default, unchecked it if you don't want to use this option. This option tell QuickForm to use the Name property as a prefix for object. (Ex. Txt for TextBox.)

The **Bound DataFields** option is unchecked by default, checked it if you want to use it. This option tell to QuickForm to include the name of the table or query you have selected, in the DataSource properties and the corresponding field's name in the DataField properties.(Bound to DataControl)

The **Generate Buttons** option is checked by default, uncheck it if you don't want buttons to be generated. The Generate Buttons option tell QuickForm to generate these buttons: New, Delete, Save, Cancel, Search, Filter, and Help.

Now, you have to select **Default class for** objects. When reading the table or query you have selected, QuickForm determines the datatype for each field in the table. When QuickForm encountered a logical field, the class found in the default object class for True/False value is used for control type, otherwise , the control type of the control will be the class found in the default object class for Text, Number, Etc... The default class for labels will be used for generating labels for controls that don't have a Caption property. The default class for buttons will be used for the buttons generated if the **Generate Buttons** option is selected.

The option **After Caption Text** works as follow. If a field is named CustomerID, then the default caption property set by QuickForm will be "CustomerID". If you put a value in this field, like ":", then the default caption will be "CustomerID:"

Database

Now, you have to open a database and select a table or a query. After selecting the table or query, the list above is filled with the table or query information.

The first row contains the form header.

The second row contains the form and file information :

FormName: Name property of the form.

FileName: *.FRM file to be generated.

Caption: Caption property of the form.

The third row contains the field header..

The fourth row and following ones contain following fields information:

PrimaryKey: The field is part of the primary key if the word PrimaryKey appeared in this column.

FieldSource: Name of the fields in the table or query.

FieldName: Name of the control in the form.

Caption: Label of the control in the form.

Class: Object class used when generating this control.

DataType: DataType of the field in the table or query.

All these properties can be changed by selecting the item and choosing the Edit menu or by right clicking on it, the Edit menu will appear.

Generating the form

Select the destination directory in the Destination tab. The default directory is the same directory as the database file previously selected. Press the Select button if you want to change it.

Ready? Press **GO!!!**

Your form is now completed and ready to use. Add it to your project, and that's all.