

Adding New Menu Buttons to UI Builder for Microsoft Access

Table of Contents

Introduction	1
Alternatives.....	2
UI Builder's Form Directory.....	2
UI Builder's Report/Query Viewer	2
UI Builder - Enterprise's "Alternate Roles" functionality.....	2
Instructions for Starter and Business Editions.....	3
Instructions for the Enterprise Edition	5

Introduction

In the past several years, some users have requested help adding more menu buttons to UI Builder. This document outlines the steps to do so.

- You will need to be familiar with Microsoft Access tables, forms, and Visual Basic to complete the changes described.
- We strongly recommend making a backup copy of your working database.
- Making these modifications will mean upgrading to the next edition of UI Builder requires you to "retrofit" these changes into the newer version.

The most common question we hear is, why are there so many steps to add more menu buttons? In brief, UI Builder was designed to provide a more usable alternative to the Microsoft Access switchboard, and other custom menus. To maintain usability and provide a solution that can be used by a majority of screen resolutions, eight menu buttons were selected. The design of UI Builder is centered around making it easy to configure and maintain an Access database menu, without overwhelming users with buttons. Moreover, since each main menu button can have five submenu buttons, a total of 40 menu buttons can be presented to the user based on their selections.

Alternatives

There are several alternatives to making changes to UI Builder's menu that can help you offer Access to more forms and reports.

UI Builder's Form Directory

UI Builder's Form Directory is available in all editions. The Form Directory is a way to present end-users with an unlimited list of forms they can open. You are able to add forms to the directory, provide a descriptive name, and indicate how the form should be opened. You can then add the Form Directory as a menu option, which gives your users access to as many forms as needed. Typically the Form Directory will be most useful where you have many forms for less critical activities, such as maintaining lookup values. The Enterprise Edition of UI Builder even supports the ability to limit visibility to specific forms in the Form Directory based upon the user's assigned role. Additional information on setting up the Form Directory is available in the UI Builder Application Guide.

UI Builder's Report/Query Viewer

UI Builder's Report/Query Viewer is available in all editions. The Report/Query Viewer is a way to present end-users with an unlimited list of reports and forms they can open. You are able to add queries and reports to the list, provide a descriptive name, and long description. You can then add the Report/Query Viewer as a menu option, which gives your users access to as many reports and queries as needed. The Enterprise Edition of UI Builder even supports the ability to limit visibility to specific reports and queries in the Report/Query Viewer based upon the user's assigned role. Additional information on setting up the Report/Query Viewer is available in the UI Builder Application Guide.

UI Builder - Enterprise's "Alternate Roles" functionality

UI Builder Enterprise allows you to configure different menus for each role you create. In addition, when you assign a role to a user, you can assign more than one role to them. For example, you can choose to assign "Administrator" and "Sales Rep" roles to a single user. When they open the database, they will see a small popup form with dropdown box where they can "toggle" between the different role menus they have access to. In this way, you can offer users a wide array of menu buttons, grouped based on the functional role they need to take on. Additional information on setting up alternate roles is available in the UI Builder Application Guide.

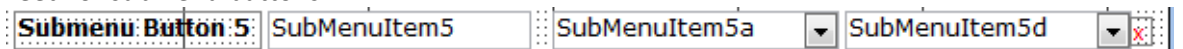
Instructions for Starter and Business Editions

Note Wherever the number (#) sign is used, replace that with the number of the new menu item you are adding, beginning with 9. Repeat these steps for as many items you need to add, replacing # with the next number.

1. Add new buttons to the form frmMain, and any other layout you want to use (all of the main menu layout forms begin with **frmMain_***)
 - a. Rename the Exit button to btn#+1 (if you will have a total of 10 buttons, then you would name it btn11)
 - b. Make sure to name each new button btn#.
 - c. Copy the subroutine **btn8_Click** and create **btn#_Click** (replacing any reference to 9 in the Sub to #)
 - d. Copy the subroutine **btn8_MouseMove** and create **btn#_MouseMove** (replacing any reference to 9 in the Sub to #)
 - e. Change the subroutine ClearMenu to reference #+1 instead of 9
2. Add new submenu buttons to the form frmMain, and any other layout you want to use
 - a. Make sure to name each new button sbtn#.
 - b. Copy the subroutine **sbtn5_Click** and create **sbtn#_Click** (replacing any reference to 5 in the Sub to #)
 - c. Copy the subroutine **sbtn5_MouseMove** and create **sbtn#_MouseMove** (replacing any reference to 5 in the Sub to #)
 - d. Change the subroutine ClearMenu to reference #+1 instead of 5
3. Add new columns to the table tblAppInfo
 - a. Menu-Submenu-#
 - b. Menu-Action-#d
 - c. Menu-Action-#
 - d. Menu-Item-#
(recommend you copy existing field into new row, or duplicate existing field definition)
4. Add new columns to the table tblSubmenu
 - a. SubMenuItem#
 - b. SubMenuItem#a
 - c. SubMenuItem#d
(recommend you copy existing field into new row, or duplicate existing field definition)
5. Add new rows to tblSubMenu, adding # to the MenuItem column, and 0 to the RoleID column.
6. In the form frmAdministrator:
 - a. In the Default Menu tab, select, copy, and paste an entire row.
 - a. Copy the naming convention of each control in the new row.
 - b. Change the control sources to map to the new row #s.
 - c. Make the following VB Changes

- Copy the subroutine **Menu_Action_8_Change** and create new **Menu_Action_#_Change**
- Add reference to **Menu_Action_#_Change** in Form_Load
- Add reference to **Menu_Action_#.RowSource = Menu_Action_1.RowSource** in Form_Load
- Change **Do Until i = 9** to **Do Until i = 11** (where 11 is # + 1)
- If Form_Timer, change **Do Until i = 9** to **Do Until i = 11** (where 11 is # + 1)
- Copy the subroutine **Menu_Action_8_Change** and create **Menu_Action_#_Change**
- Copy the subroutine **Menu_Action_8d_Change** and create **Menu_Action_#d_Change**
- Copy the subroutine **Menu_Action_8_LostFocus** and create **Menu_Action_#_LostFocus**

7. In the form frmAdminSubform add the number of desired submenu buttons.
 - a. Copy and paste the entire row Submenu Button 5 to make as many new rows as you need for submenu buttons.



- b. Copy the naming convention of each control in the new row.
 - c. Change the control sources to map to the new row #s.
 - d. Make the following VB Changes
 - Copy the subroutine **SubMenuItem5d_Change()** and create new **SubMenuItem#d_Change()**
 - Copy the subroutine **SubMenuItem5a_Change()** and create new **SubMenuItem#a_Change()**
 - Copy the subroutine **SubMenuItem5a_LostFocus ()** and create new **SubMenuItem#a_LostFocus ()**
 - Copy the subroutine **btnClear5_MouseMove ()** and create new **btnClear#_MouseMove ()**
 - Copy the subroutine **btnClear5_Click ()** and create new **btnClear#_Click ()**
 - Add reference to **SubMenuItem#a_Change** in Form_Load
8. In the function fLoadMenu (located in the module modMenu):
 - a. Change all occurrences of **Do Until i = 9** to **Do Until i = #+1**
9. In the function fLoadSubMenu (located in the module modMenu):
 - a. Change all occurrences of **Do Until iCounter = 6** to **Do Until i = #+1 (for # of submenu buttons)**
10. In the function fSetColors:
 - a. Change Do Until i = 10 to **Do Until i = #+1 (for # of main menu buttons)**
 - b. Change Do Until i = 6 to **Do Until i = #+1 (for # of submenu buttons)**
11. In the function fGetCurrentMenu:
 - a. Change **Do Until i = 9** to **Do Until i = #+1**
12. In the function fShowSubMenu2
 - a. Add new lines: **Forms!frmMain!sbtn#.Visible = False**

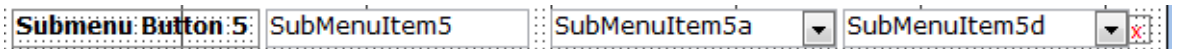
Instructions for the Enterprise Edition

Note Wherever the number (#) sign is used, replace that with the number of the new menu item you are adding, beginning with 9. Repeat these steps for as many items you need to add, replacing # with the next number.

1. Add new buttons to the form frmMain, and any other layout you want to use (all of the main menu layout forms begin with **frmMain_***)
 - a. Rename the Exit button to btn#+1 (if you will have a total of 10 buttons, then you would name it btn11)
 - b. Make sure to name each new button btn#.
 - c. Copy the subroutine **btn8_Click** and create **btn#_Click** (replacing any reference to 8 in the Sub to #)
 - d. Copy the subroutine **btn8_MouseMove** and create **btn#_MouseMove** (replacing any reference to 8 in the Sub to #)
 - e. Change the subroutine ClearMenu to reference #+1 instead of 9
2. Add new submenu buttons to the form frmMain, and any other layout you want to use
 - a. Make sure to name each new button sbtn#.
 - b. Copy the subroutine **sbtn5_Click** and create **sbtn#_Click** (replacing any reference to 5 in the Sub to #)
 - c. Copy the subroutine **sbtn5_MouseMove** and create **sbtn#_MouseMove** (replacing any reference to 5 in the Sub to #)
 - d. Change the subroutine ClearMenu to reference #+1 instead of 5
3. Add new columns to the table tblAppInfo
 - a. Menu-Submenu-#
 - b. Menu-Action-#d
 - c. Menu-Action-#
 - d. Menu-Item-#
(recommend you copy existing field into new row, or duplicate existing field definition)
4. Add new columns to the table tblRoleMenu (necessary for Enterprise edition only)
 - a. Menu-Submenu-#
 - b. Menu-Action-#d
 - c. Menu-Action-#
 - d. Menu-Item-#
(recommend you copy existing field into new row, or duplicate existing field definition)
5. Add new columns to the table tblSubmenu
 - a. SubMenuItem#
 - b. SubMenuItem#a
 - c. SubMenuItem#d
(recommend you copy existing field into new row, or duplicate existing field definition)
6. Add new rows to tblSubMenu, adding # to the MenuItem column, and 0 to the RoleID column.

7. In the form frmAdministrator:
 - b. In the Default Menu tab, select, copy, and paste an entire row.
 - a. Copy the naming convention of each control in the new row.
 - b. Change the control sources to map to the new row #s.
 - c. Make the following VB Changes
 - Copy the subroutine **Menu_Action_8_Change** and create new **Menu_Action_#_Change**
 - Add reference to **Menu_Action_#_Change** in Form_Load
 - Add reference to **Menu_Action_#.RowSource = Menu_Action_1.RowSource** in Form_Load
 - Change **Do Until i = 9** to **Do Until i = 11** (where 11 is # + 1)
 - If Form_Timer, change **Do Until i = 9** to **Do Until i = 11** (where 11 is # + 1)
 - Copy the subroutine **Menu_Action_8_Change** and create **Menu_Action_#_Change**
 - Copy the subroutine **Menu_Action_8d_Change** and create **Menu_Action_#d_Change**
 - Copy the subroutine **Menu_Action_8_LostFocus** and create **Menu_Action_#_LostFocus**

7. In the form frmAdminSubform add the number of desired submenu buttons.
 - a. Copy and paste the entire row Submenu Button 5 to make as many new rows as you need for submenu buttons.



- b. Copy the naming convention of each control in the new row.
 - c. Change the control sources to map to the new row #s.
 - d. Make the following VB Changes
 - Copy the subroutine **SubMenuItem5d_Change()** and create new **SubMenuItem#d_Change()**
 - Copy the subroutine **SubMenuItem5a_Change()** and create new **SubMenuItem#a_Change()**
 - Copy the subroutine **SubMenuItem5a_LostFocus ()** and create new **SubMenuItem#a_LostFocus ()**
 - Copy the subroutine **btnClear5_MouseMove ()** and create new **btnClear#_MouseMove ()**
 - Copy the subroutine **btnClear5_Click ()** and create new **btnClear#_Click ()**
 - Add reference to **SubMenuItem#a_Change** in Form_Load

8. In the form frmAdminRoleMenu (necessary for Enterprise edition only)
 - a. In the Default Menu tab, select, copy, and paste an entire row.
 - b. Copy the naming convention of each control in the new row.
 - c. Change the control sources to map to the new row #s.
 - d. Make the following VB Changes
 - Copy the subroutine **Menu_Action_8_Change** and create new **Menu_Action_#_Change**
 - Add reference to **Menu_Action_#_Change** in Form_Load
 - Add reference to **Menu_Action_#.RowSource = Menu_Action_1.RowSource** in Form_Load
 - Change **Do Until i = 9** to **Do Until i = 11** (where 11 is # + 1)

- If Form_Timer, change **Do Until i = 9** to **Do Until i = 11** (where 11 is # + 1)
 - Copy the subroutine **Menu_Action_8_Change** and create **Menu_Action_#_Change**
 - Copy the subroutine **Menu_Action_8d_Change** and create **Menu_Action_#d_Change**
 - Copy the subroutine **Menu_Action_8_LostFocus** and create **Menu_Action_#_LostFocus**
9. In the function fLoadMenu (located in the module modMenu):
- a. Change all occurrences of **Do Until i = 9** to **Do Until i = #+1**
10. In the function fLoadSubMenu (located in the module modMenu):
- a. Change all occurrences of **Do Until iCounter = 6** to **Do Until i = #+1 (for # of submenu buttons)**
11. In the function fSetColors:
- a. Change Do Until i = 10 to **Do Until i = #+1 (for # of main menu buttons)**
 - b. Change Do Until i = 6 to **Do Until i = #+1 (for # of submenu buttons)**
 - c. Change **Do Until i = 9** to **Do Until i = #+1**
12. In the function fGetCurrentMenu:
13. In the function fShowSubMenu2
- a. Add new lines: **Forms!frmMain!sbtn#.Visible = False**