ACCESS USER GROUP - EUROPE APRIL 3, 2024

Access Shortcut Tool Demo

Simple Audit Log

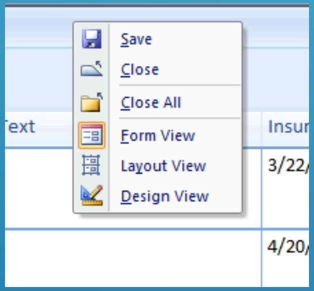
PRESENTER:

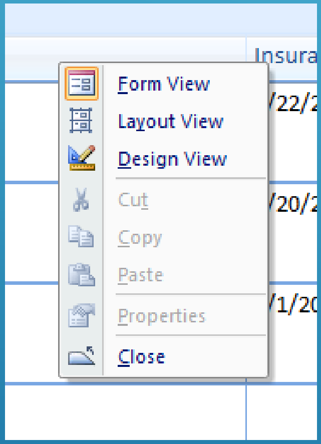
- Dale Fye
- Owner: Dev-Soln, LLC (Developing Solutions)
- Microsoft Access MVP (2013-2016)
- www.dev-soln.com

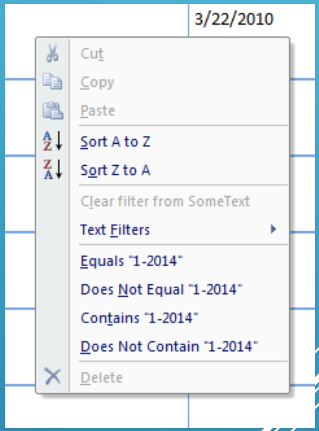
- Retired Army Officer
- Access developer since 1994
- Contribute on Experts Exchange

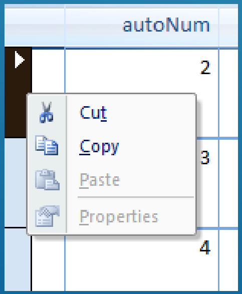
WHAT IS A SHORTCUT MENU

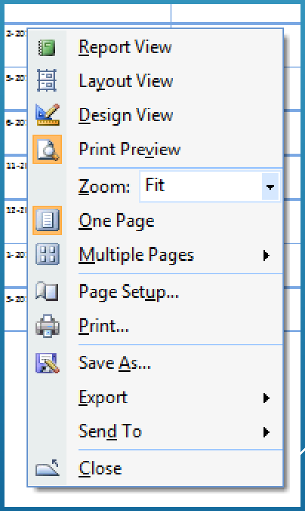
- Technically, it is a member of the CommandBars collection











Commandbar History

- Shortcut menus are an integral part of all Windows applications.
- Each windows application has its own set of shortcut menus.
- Through A2003 – Access had a built in tool for creating menu bars and shortcut menus.
- Feature (built-in tool) deprecated with A2007
  - still able to import menu bars and shortcut menus from earlier versions.
- A2010+
  - "Menu" commandbars were replaced by Ribbon
  - still able to import shortcut menus
- Shortcut menus still available:
  - build shortcut menus using code
  - Import from previous versions
- Runtime applications do not use the default shortcut menus but you can use your own.

ACCESS COMMANDBARS

How many are there?

```
?application.commandbars.count
```

What are they?

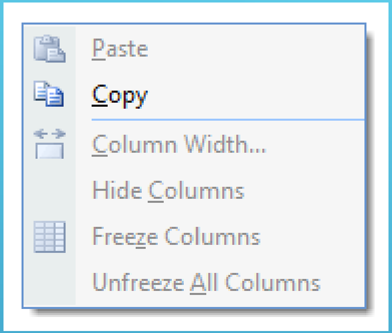
```
For each cbr in application.commandbars
    debug.print cbr.name
Next
```

What controls do they contain?

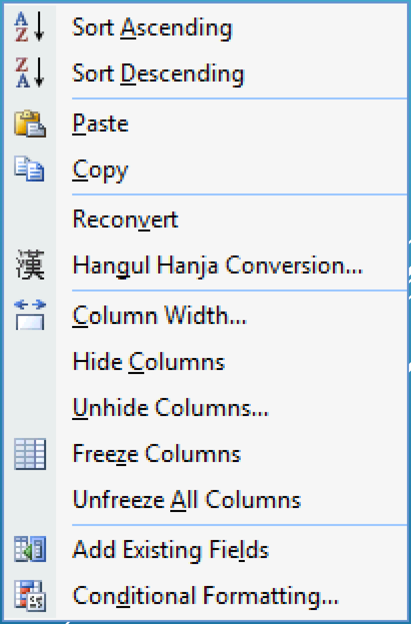
```
With commandbars("Form Datasheet Column")
    for intloop = 1 to .Controls.Count
        debug.Print  .Controls(intLoop).Caption, .Controls(intLoop).ID
    next
End with
```

DISPLAYING ELEMENTS OF A COMMANDBAR

Commandbars("Form Datasheet Column").ShowPopup



Actual list of all controls in the commandbar.



Note:
Built-in shortcut menus are context sensitive. Some controls will be hidden others will be enabled/disabled depending on the context the menu is used in.

BUILT-IN MENUS

- There are over 200 built in menus
- Hard to determine exactly which commandbar is visible
- Determining values of various properties of a commandbar and its controls is difficult
  - Commandbars have a “Name” property
  - Controls have a “Caption” property but not “Name”
  - Commandbar.Index property is not consistent:
    - changes after adding or deleting a commandbar
  - Commandbar controls have an index property which is consistent
  - FaceID (for images) of the built-in controls

USING BUILT-IN COMMANDBARS

- I frequently disable the built-in commandbars on forms and reports.
- Built-in commandbars are disabled in runtime applications
- Difficult to determine which commandbar you are seeing, especially with datasheets (21 separate cbars)
- You can manipulate built-in commandbars in your applications
  - **Be really careful when doing this.**
  - Recommend making these temporary

CREATING YOUR OWN SHORTCUT MENU

- Early Binding:
  - Microsoft Office XX.0 Object Library
- Declare variables

```
Dim cbr as CommandBar , ctrl as CommandBarControl
Dim cbrButton As CommandBarButton
Dim cbrcombo As CommandBarComboBox
```

- Define commandbar

```
set cbr = CommandBars.Add(Name, [Position], [Menubar], [Temporary])
Name       := name used to refer to the commandbar
Position   := Must be 5 for shortcut menus
Menubar    := [True/False] Replace the active menu bar – default is False
Temporary  := False if you want the cbar to remain in the application
permanently
```

ADDING CONTROLS

- Defining Controls:

  Set ctrl = cbr.Controls.Add([Type], [ID], [Parameter], [Before], [Temp])

Type := msoConrolButton,
msoControlEdit,
msoControlDropdown,
msoControlPopup

ID := value of the ID from any of the built-in controls

Parameter := any value you want to pass to the procedure which will be executed when the option is selected

Before := where you want the control in the commandbar. If greater than the number of controls in the commandbar, it will raise an error

Temp := False if you want the control to remain between sessions

```
Set ctrl0 = .Add(Type:=1, Temporary:= -1)
With ctrl0
    .Caption = "&Print“
    .OnAction = "=fnReportPrint()“
    .FaceID = 4
    .Visible = -1
    .Enabled = -1
    .BeginGroup = 0
    .ToolTipText = "&Print“
    .Width = 177
End With
```

CONTROL PROPERTIES

.Caption	:= same as form caption
.OnAction	:= "=fnReportPrint()“
	- this refers to the procedure which will run when the option is selected
.FaceID	:= if using one of images associated with built-in options
.Visible	:= self explanatory
.Enabled	:= self explanatory
.BeginGroup	:= insert a line separator between items
.ToolTipText	:= self explanatory
.State	:= Checked or unchecked

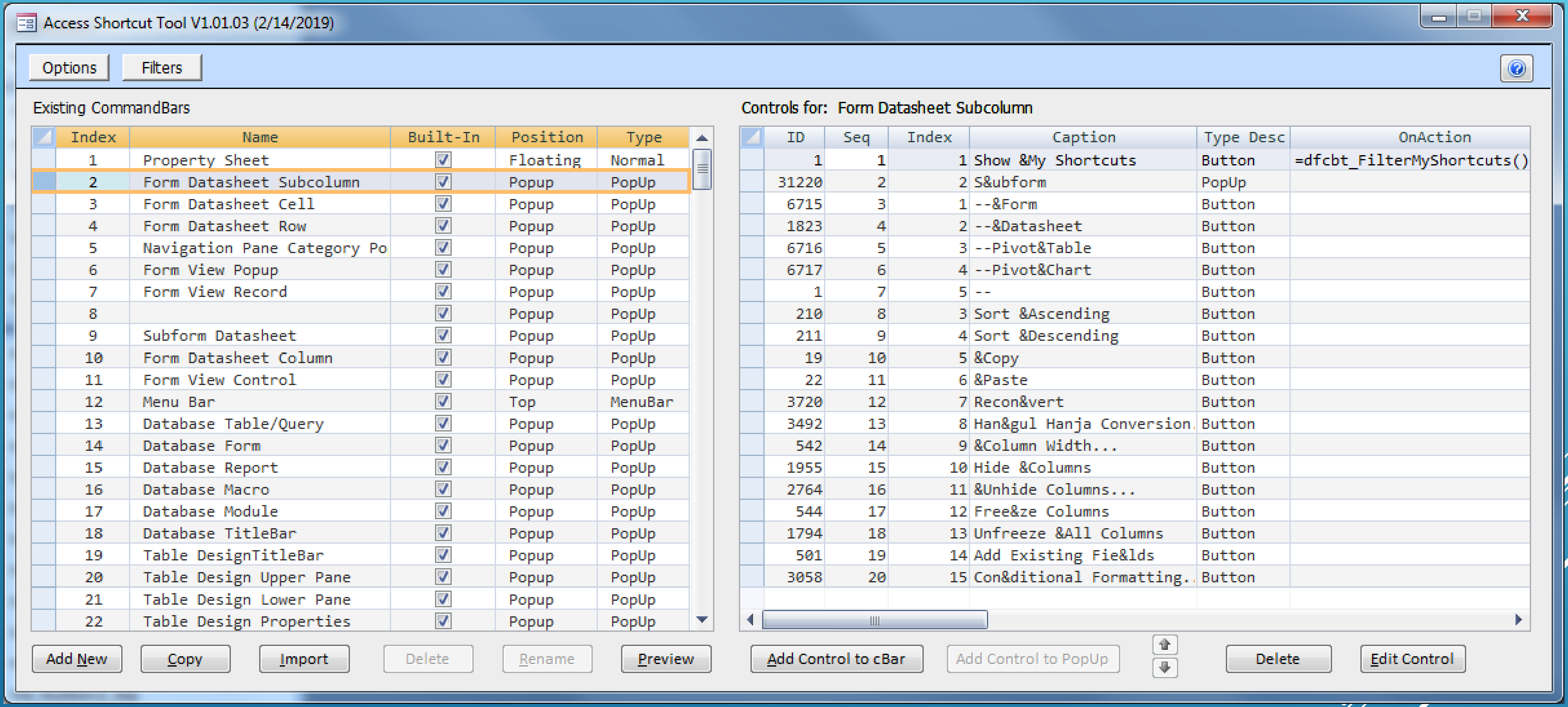
RUNTIME SHORTCUT MENUS

- Built-in menus are disabled
- You can copy and use the built-in menus but takes a bit more work.
- Article on using commandbars in run-time environment

ACCESS SHORTCUT TOOL

- Shortcut Tool contains
  - What's This Menu option
  - Add New
  - Copy commandbar
  - Import – both toolbar and code
- After selecting a commandbar, you can see all of the properties of each of the controls

ACCESS SHORTCUT TOOL



Existing CommandBars

Index	Name	Built-In	Position	Type
1	Property Sheet	✓	Floating	Normal
2	Form Datasheet Subcolumn	✓	Popup	PopUp
3	Form Datasheet Cell	✓	Popup	PopUp
4	Form Datasheet Row	✓	Popup	PopUp
5	Navigation Pane Category Po	✓	Popup	PopUp
6	Form View Popup	✓	Popup	PopUp
7	Form View Record	✓	Popup	PopUp
8		✓	Popup	PopUp
9	Subform Datasheet	✓	Popup	PopUp
10	Form Datasheet Column	✓	Popup	PopUp
11	Form View Control	✓	Popup	PopUp
12	Menu Bar	✓	Top	MenuBar
13	Database Table/Query	✓	Popup	PopUp
14	Database Form	✓	Popup	PopUp
15	Database Report	✓	Popup	PopUp
16	Database Macro	✓	Popup	PopUp
17	Database Module	✓	Popup	PopUp
18	Database TitleBar	✓	Popup	PopUp
19	Table DesignTitleBar	✓	Popup	PopUp
20	Table Design Upper Pane	✓	Popup	PopUp
21	Table Design Lower Pane	✓	Popup	PopUp
22	Table Design Properties	✓	Popup	PopUp

Controls for: Form Datasheet Subcolumn

ID	Seq	Index	Caption	Type Desc	OnAction
1	1	1	Show &My Shortcuts	Button	=dfcbt_FilterMyShortcuts()
31220	2	2	S&ubform	PopUp	
6715	3	1	--&Form	Button	
1823	4	2	--&Datasheet	Button	
6716	5	3	--Pivot&Table	Button	
6717	6	4	--Pivot&Chart	Button	
1	7	5	--	Button	
210	8	3	Sort &Ascending	Button	
211	9	4	Sort &Descending	Button	
19	10	5	&Copy	Button	
22	11	6	&Paste	Button	
3720	12	7	Recon&vert	Button	
3492	13	8	Han&gul Hanja Conversion.	Button	
542	14	9	&Column Width...	Button	
1955	15	10	Hide &Columns	Button	
2764	16	11	&Unhide Columns...	Button	
544	17	12	Free&ze Columns	Button	
1794	18	13	Unfreeze &All Columns	Button	
501	19	14	Add Existing Fie&lds	Button	
3058	20	15	Con&ditional Formatting..	Button	

ARTICLES AND OTHER RESOURCES

- Articles
  - Understanding and using CommandBars Part 1
  - Understanding and using CommandBars Part 2
  - Using built-in Shortcut Menus
  - Using Shortcut menus in Access run-time
- Access Shortcut Tool

SIMPLE AUDIT LOG

- Used to keep track of changes to data values
- Common uses:
  - Financial Systems
  - Public records
  - Personal records

GENERAL TECHNIQUE

- Create a separate “audit” table for each table you want to track changes
- Use the Form before and after update events to write entire record to the “audit” table
- Issues:
  - Database bloat
  - Hard to determine what has changed between records

MY TECHNIQUE

- Single “Audit” table with fields for:
  - Table name
  - Action (Insert, Edit, Delete)
  - Action By: who made the change
  - Action_DT: when action was taken
  - Record_ID: PK for the record inserted, edited or deleted
  - Field name: name of the field that was changed
  - Field value: value that the field was changed to

PROCESSES

- Uses the following events
  - Form_BeforeUpdate
  - Form_AfterUpdate
  - Form_Delete
  - Form_AfterDelConfirm
- One line of code in each event
- BeforeUpdate and Delete events save data to [Audit_Log_Temp] table
- AfterUpdate and AfterDeleteConfirm move data from "temp" table to "Audit_Log" table

RESOURCES

- Articles
  - Simple Audit Log