Access 2003 Macros

Stephen Moffat, The Mouse Training Company

Microsoft

Microsoft Office Access 2003

Stephen Moffat, The Mouse Training Company

Access 2003 Macros

Access 2003 Macros

ISBN 978-87-403-0036-9

Contents

Course Objectives

This manual is designed to be used in a classroom environment. There are eight sections, each with their own set of objectives

The manual provides a step by step guide for each new topic with a brief introduction. There are often extra tips and information shown with reference tables which students may use after the course.

It is recommended that you have undertaken the Access 2002 Advanced course and have had adequate practice of the activities covered in the course OR have a good knowledge of the Advanced Skills of Access.

The Macros course is designed for those who are looking to design and develop 'front end' applications in Access using Macros.

These are the overall objectives for the Access Macros Course:

- Looking at Macro Concepts
- Creating Macros
- Running Macros
- Using Auto Keys
- Conditional Macros
- Event Procedures
- Splash Screens
- Main Switchboard

1 Intro to Macros

Section Objectives

- What is a Macro?
- Why are Macros used?
- How Macros work with other Database Objects

1.1 What is a Macro?

A macro helps you perform routine tasks by automating them, for example, instead of clicking the Reports tab in the database window, finding and opening a specific report, printing it and then closing it, you could create a macro to print the report with the click of a single button.

In some programs such as Microsoft Excel, it is possible to record a macro using the Macro Recorder.

A Macro is a way of programming Access to perform repetitive tasks automatically. Macros can be used to open forms, maximise them and produce welcome messages or can perform complex calculations and controls on selected data.

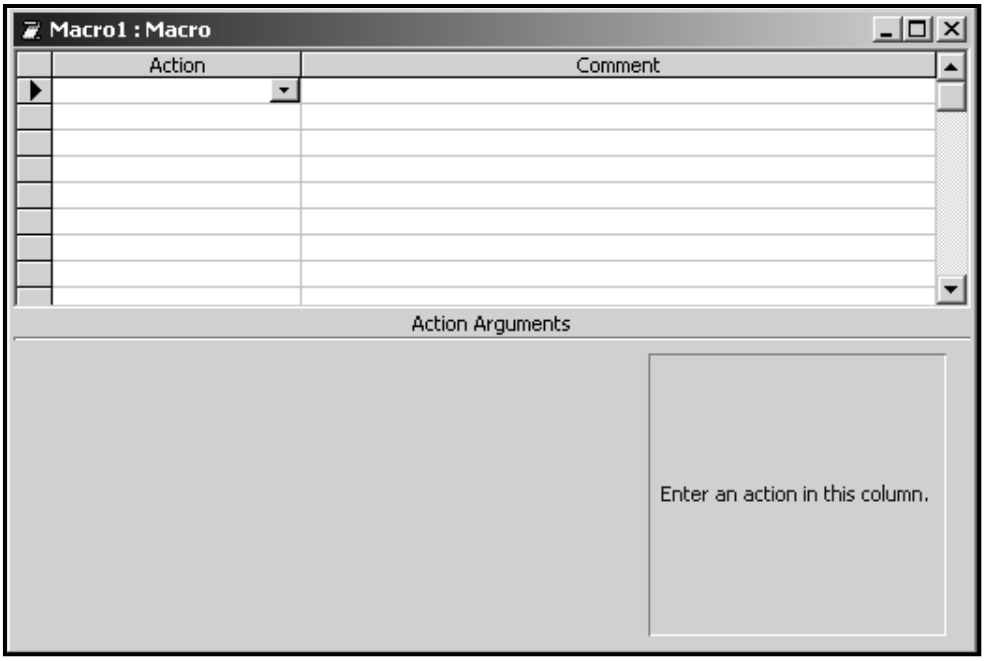
Most of the time, you will want to attach a macro to a form or form object to control the way the form or object work – often to insert standard text, work out conditions e.g. If the town is Edinburgh then the district must be Lothian, and so on.

1.2 Creating a Macro

A macro is a list of actions which are run (or "executed") in sequence. A macro may contain a single action, or it may have many.

Each task that you require the macro to do, is known as an action. When you run the macro, Access carries out the actions in the sequence you have created them. For example, one action may be to open a form, the second action may be to maximise the form.

In Access, the sequence of actions making up a macro are not recorded. The macro is designed via a graphical interface which, in its simplest for looks like the diagram below:



When you create a macro, you design it in the Macro window. The upper part of the Macro window is used to add actions and the lower part is used to define the arguments.

Action Pane

In each action cell, an action can be chosen from the combo box's drop down list, or by typing in the first few letters of the action name.

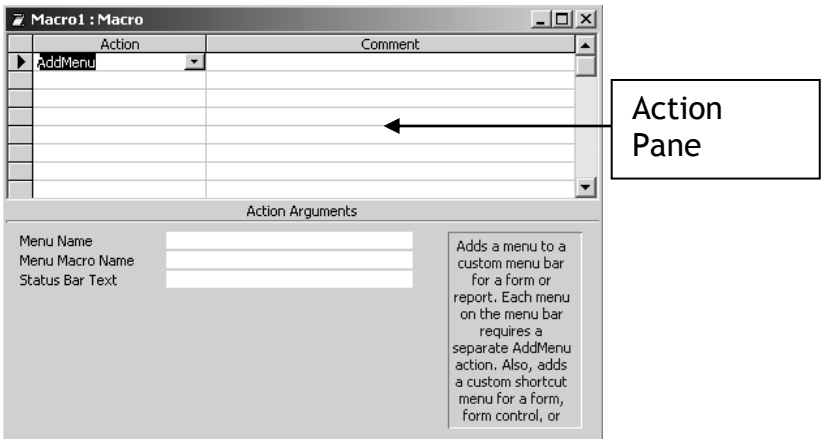
The only valid actions are those which appear in the list. Note, also that you can run a macro from within a macro by using the RunMacro action.

Each macro can have one or more actions. You add individual actions in the Action column. Description for each action can be added in the Comment column

You can also create comments with each action which is not part of the macro command and will be ignored when it is run, but is useful to the programmer to explain the reasoning behind each action.

Enter as many actions as you require in the design window. When the macro runs, the actions will be executed from top to bottom.

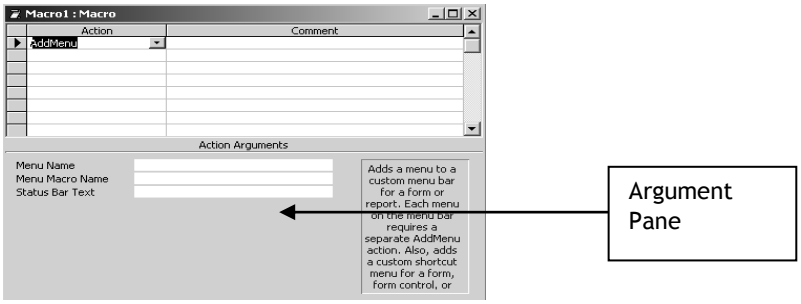
The Macro will ignore blank lines. You can, therefore, safely add blank lines and use the spacing to help readability.



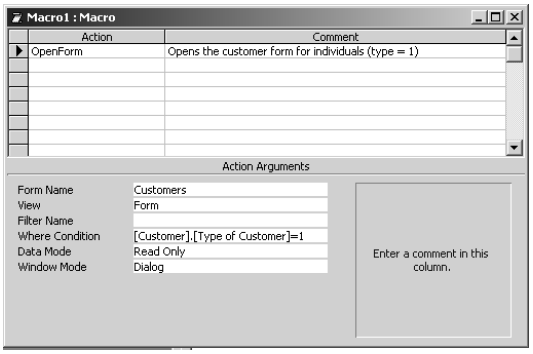
Argument Pane

Once an action has been chosen, relevant Action Arguments appear in the bottom half of the design window. This is how you further specify what the action will do. Some actions have no arguments, some have many. Some arguments are required, and others are optional. When the insertion point is in an argument cell, an explanation appears to the right of the arguments.

After you add an action to a macro, you set the arguments for the action in the lower portion of the Macro window. These arguments give additional information on how to carry out the action.



The figure below shows the action arguments for an action called OpenForm, which opens a specific form and has six different arguments that can be specified



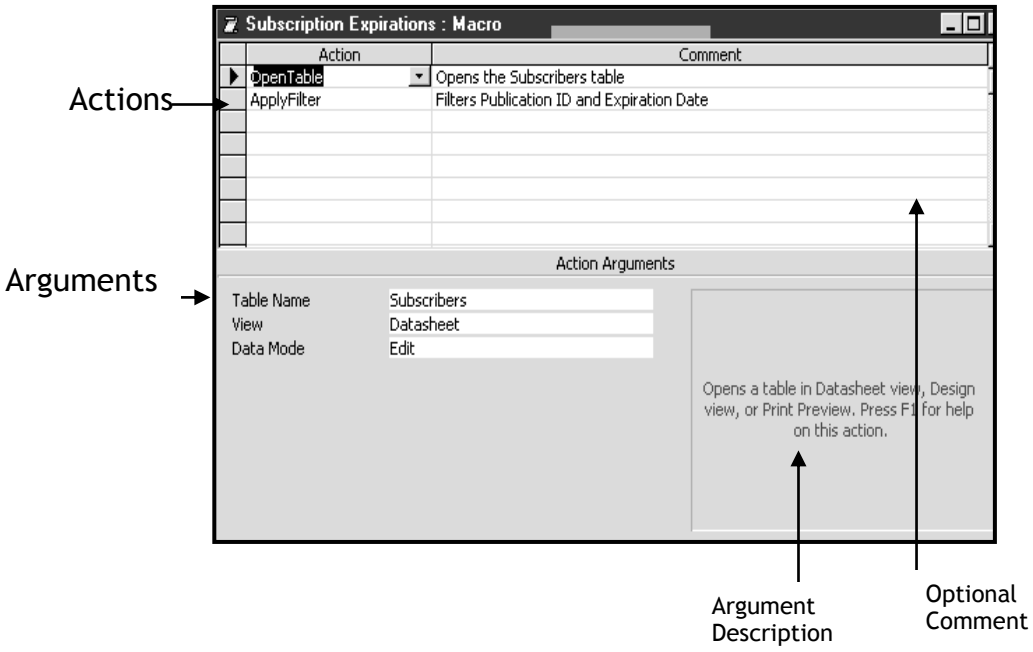
Arguments	Description
Form Name	Specifies the form for Access to open
View	Specifies the view; form, Design, Print Preview or Datasheet
Filter Name	Applies any specified filter or query
Where condition	Limits the number of records Displayed
DataMode	Specifies a Data Entry mode, Add, Edit or Read Only
Window Mode	Specifies a window mode: Normal, Hidden, Icon, Dialog

The following are useful guidelines which can be used:

1. In general, it's a good idea to set action arguments in the order they're listed, because choices for one argument may determine those for arguments that follow.
2. If an action has an argument that calls for the name of a database object, you can set the argument and the corresponding object type argument automatically by dragging the object from the Database window to the argument box.
3. If you add an action to your macro by dragging a database object from the Database window, Microsoft Access automatically sets appropriate arguments for that action.
4. You can type a value in an argument box, or in many cases you can select a setting from a list.
5. You can use an expression preceded by an equal sign (=) to set any action's arguments. You can't use an expression for the following arguments.

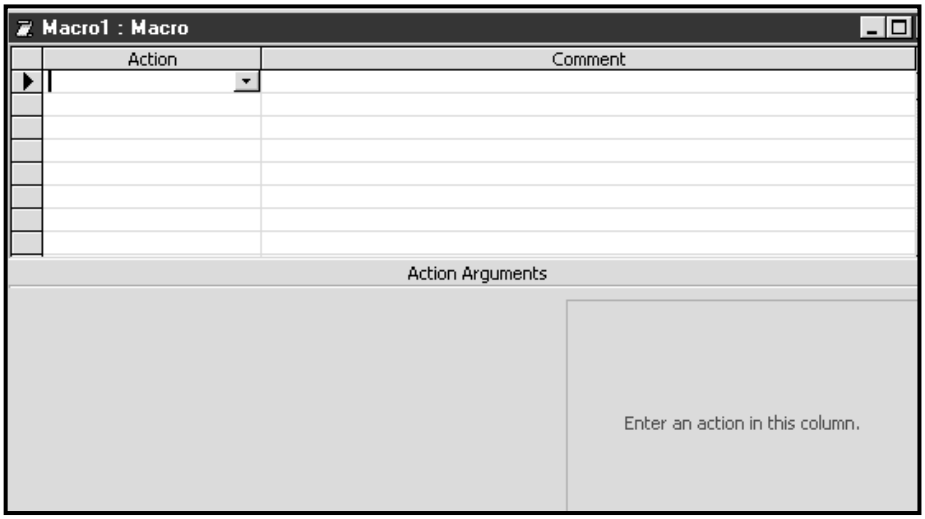
. Press F6 to move between the Actions in the top part of the window and the Argument boxes at the bottom of the window.

The following diagram shows a macro that opens a table "Subscribers" and applies a filter to extract the publications and expiration dates.



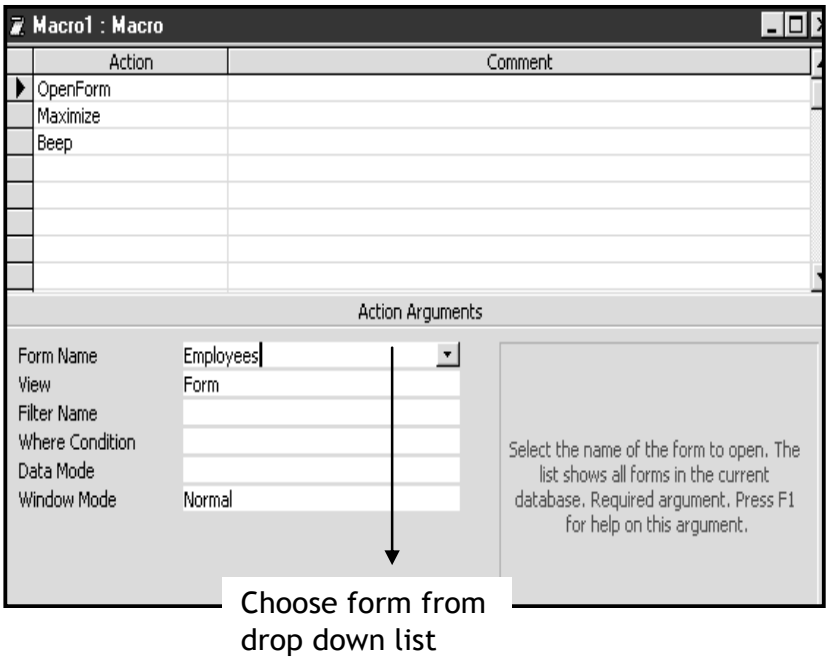
To create a new macro:

- Click onto the Macro tab from the Database window.
- Select New to show the new macro box:



- In the Action column either type in the macro action, or click on the drop down list and select the action from there. Press Tab and add a comment if necessary.
- You can add optional comments in the Comments column to assist in describing what the macro does or to assist later analysis.
- Click into the argument section on the bottom of the box and add any appropriate arguments. For example, if the action was Open Form, the argument would be the name of the form.
- Click back into the Action column on the second line and type or select the second action you want to perform. Add any appropriate arguments. Repeat to add additional steps as necessary.
- Close and save the macro.

The following diagram shows a macro that opens the form employees, maximises it, and then bleeps.



If you want to set an Action Argument for an Object Name, you can click and drag the object form the Database window to the action's Object Name Argument box in the Macro window. For example, if your action is SelectObject, you can drag a Form from the Database window and drop it on the Object Name argument to specify both the Object type and Object Name.

To Select Actions by dragging and dropping Objects

- Create a new Blank Macro
- Choose Windows, Tile Vertically from the Menu. Access places the Macro and the Database container windows side by side.
- Click on the Forms Tab to display all forms in your Database
- Click and Drag the required form from the Database Container in to any empty action cell of the Macro Window

1.3 Editing a macro

Some Microsoft Access tasks require several steps. For example a particular task might require you to (1) open a form (2) select a specific record, (3) select a specific field in that record. Macros can contain as many actions as necessary to automate even the most complicated tasks.

To edit a macro

- In the Database window, click the Macros tab.
- Click the name of the macro you want to edit.
- Click Design.
- Make the changes you want to make.

To add an action

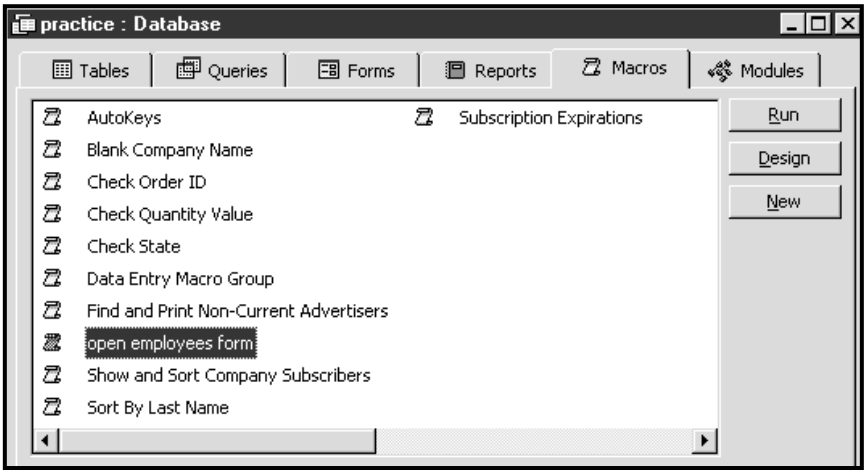
- In the Macro window, click the first empty row in the Action column.

If you want to insert an action between two action rows, click the selector for the action row just below the row where you want to insert the new action, and then click Insert Row button.

- In the Action column, click the arrow to display the action list.
- Click the action you want to use.
- Type a comment for the action. Comments are optional.
- In the lower part of the window, specify arguments for the action, if any are required. For tips on setting action arguments, click.

1.4 Using the Macro

Once the macro is saved, you can play it back doing one of the following:



To run a macro from the Macro window, click Run on the Macro Design Toolbar.

To run a macro from the Database window, click the Macros tab and then double-click a macro name.

To run a macro from form Design view or report Design view, on the Tools menu, point to Macro, and click Run Macro.

To run a macro from anywhere else in Microsoft Access, on the Tools menu, click Run Macro. Then click a macro in the Macro Name box.

. You normally run a macro directly only to test it. You can then attach the macro to a form, report, or control so that it runs in response to an event, or you can create a custom menu command that runs the macro.

Use this Page for Notes

Use this Page for Notes

2 Macro Groups

Section Objectives:

- Creating a Macro Group
- Running a Macro Group

2.1 Creating a Macro Group

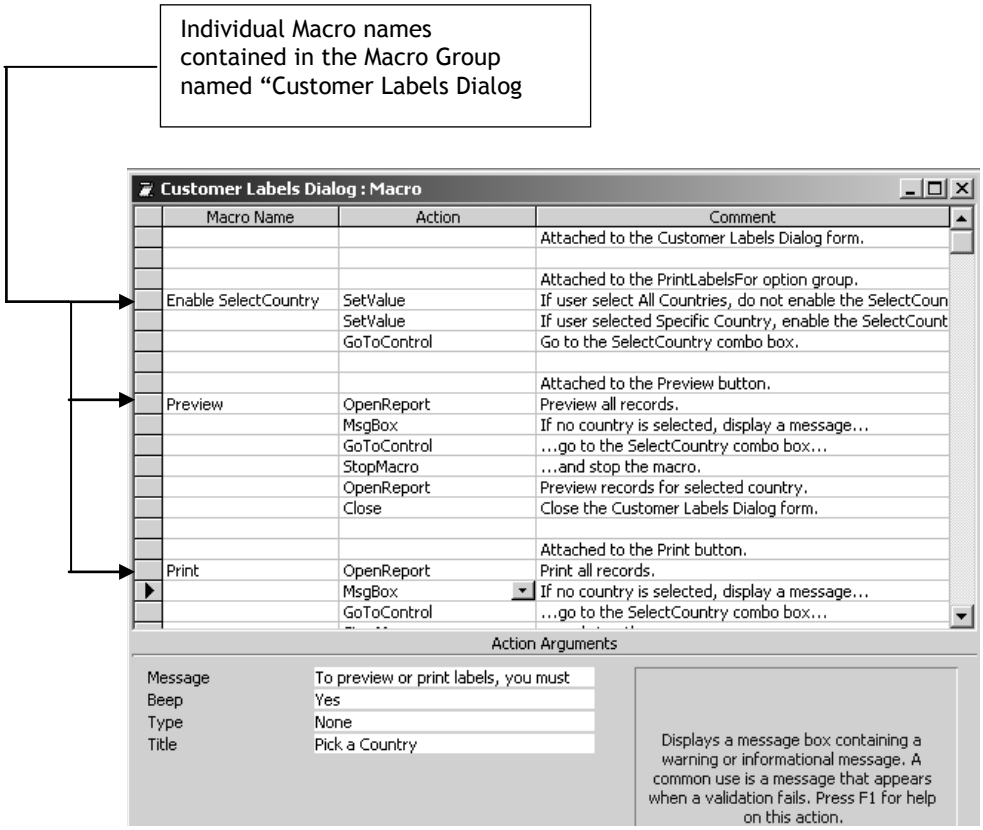
A Macro Group is a group of individual macros created in one macro design window. Each macro within the group still runs independently of the other.

Grouping your macros makes editing the macros a lot easier at a later date.

The individual macros' name will not appear in the database window, only the name of the macro group. This means that if a group contained six macros, the database window wouldn't be cluttered with six individual names; it would only contain the one macro group name.

The following is an example of a Macro Group.

To create a Macro Group



- Select the Macro tab in the Database window
- Click on the New button
- Click on the Macro Names button or choose View, Macro Names.
- Type the Macro Name for the first macro and add the Actions for the macro.
- Repeat the above step for each macro
- Choose File, Save
- Enter a Name for the group, then click on OK
- Choose File, Close

Running a Macro within a Group

A macro in a group can be run from the same areas as an individual macro. However, the group name must be included when entering the macro name. A full stop is used to separate the group and macro names.

GroupName.MacroName

If you are viewing macro names from an area other than the Database window, Access will display the Group name followed by the macro name for each macro in the database.

. If you wish, you can choose to run a macro group as an ordinary macro, e.g. by double clicking on the macro group name in the database window. When the macro group is run this way, only the first macro in the group will run.

Use this page for notes

3 Auto Keys

Section Objectives:

- Assigning an Action to a Key
- Key Combinations

3.1 Auto Keys

Assign an action or set of actions to a key

To make an Access application more user friendly to an end user, it is possible to assign frequently used actions to specific key combinations. For example, you can create a global keyboard shortcut to have Access close the current form when the user presses the F5 key.

You can assign an action or set of actions to specific key combinations by creating a macro group called AutoKeys. When you press the key or key combination, Microsoft Access carries out the action.

. If you assign a set of actions to a key combination that has already been assigned a set of actions by Microsoft Access, e.g. CTRL + A to select all, the actions that you assign to this key combination will replace the Microsoft Access key assignment.

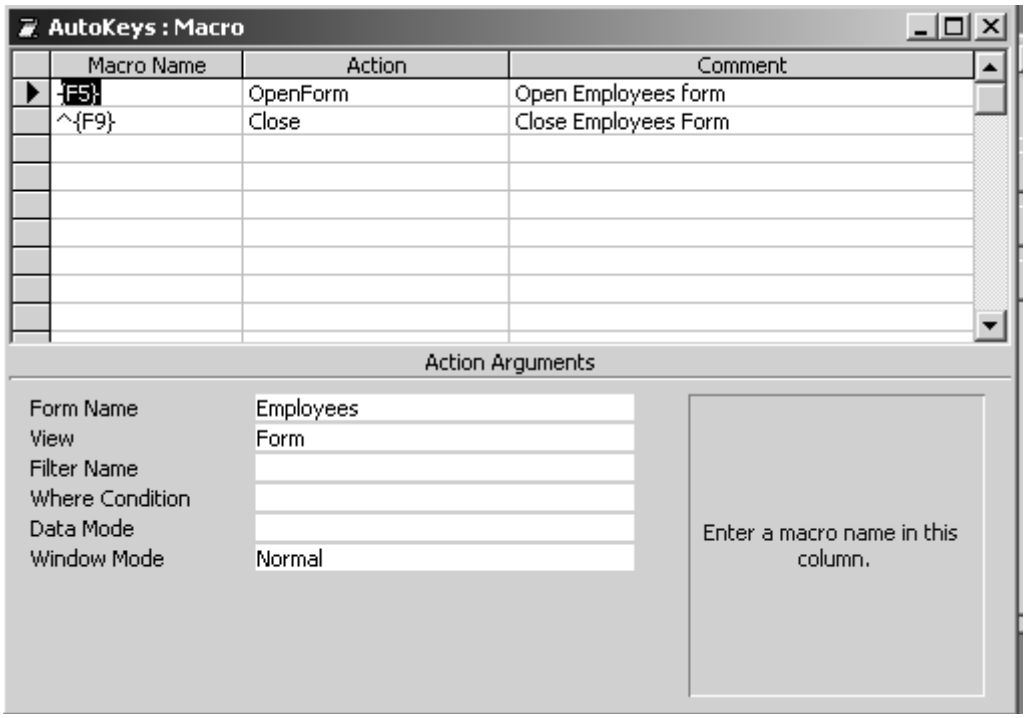
The table below lists all the key combinations you can use to make key assignments. Remember just as you use the ^ sign to represent the CTRL key, you may use the + sign to represent the SHIFT key

AutoKey syntax	**Key combination**
^A or ^4	CTRL + Any letter or number key
{F1}	Any function key
^{F1}	CTRL + Any function key
+{F1}	SHIFT + Any function key
{INSERT}	INS
^{INSERT}	CTRL + INS
+{INSERT}	SHIFT + INS
{DELETE} or {DEL}	DEL
^{DELETE} or ^{DEL}	CTRL + DEL
+{DELETE} or +{DEL}	SHIFT + DEL

To create an AutoKey Macro

- Click the Macros tab in the Database window,.
- Click New.
- Click Macro Names  on the toolbar.
- In the Macro Name column, type the key or key combination to which you want to assign the action or set of actions. See Above Table
- Add the action or set of actions you want the key or key combination to carry out. For example, you could add a RunMacro action that runs the Print Current Record macro when CTRL+P is pressed.
- Repeat for any other key assignments you want to make.
- Save the macro group with the name AutoKeys.

The new key assignments are in effect as soon as you save the macro group and each time you open the database.



Use this page for notes

4 Event Procedures

Section Objectives:

- What are Events
- Attaching Macros To Events

4.1 What is an event?

An event is a specific action that occurs on or with a certain object. Microsoft Access can respond to a variety of events: mouse clicks, changes in data, forms opening or closing, and many others. Events are usually the result of user action.

Access identifies certain actions that occur on a form or report as an Event. For example, moving from one record to another, selecting or exiting from a field, double clicking a control, etc. Each form/report and control event has an Event Property. By setting the appropriate event property to a Macro Name, you can regulate how the control is going to respond to the event. When the Event occurs, the Macro will run. For example, when you open a form, Access initiates an Open event

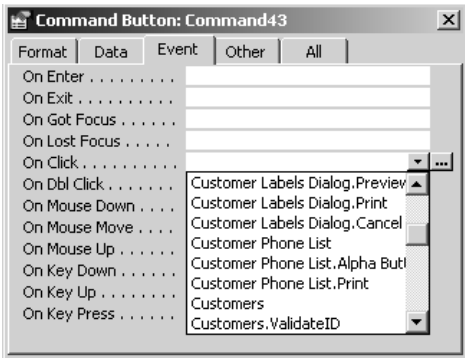
If you attach a macro to the Open event property of the form, then the macro will run every time the form is opened.

Attaching Macros to Events

Macros that are attached to an event are useful for opening specific forms, printing reports using specific queries, etc

To Attach a Macro to an Event:

- Open the Form or Report in the Design View.
- Select the object (i.e. Form or Report) or the control within that object that you wish to attach the macro to.
- Choose View, Properties or
- Click on the Properties  button
- Click on the Event Properties tab
- Select the Event that you are going to attach the macro to from the Properties sheet
- Click on the drop down arrow  at the end of the event box then select the Macro



4.2 Different Types of Events

The tables below provide a complete list and explanation of all events founding MS Access 2002 grouped by the following main categories:

Data Events

Event	It occurs
AfterDelConfirm	After you confirm record deletions and the records are actually deleted, or after the deletions are cancelled.
AfterInsert	After a new record is added to the database.
AfterUpdate	After a control or record is updated with changed data. This event occurs when the control or record loses the focus, or you click Save Record on the Records menu. This event occurs for new and existing records.
BeforeDelConfirm	After one or more records are deleted, but before Microsoft Access displays a dialog box asking you to confirm or cancel the deletion. This event occurs after the Delete event.
BeforeInsert	When you type the first character in a new record, but before the record is added to the database.
BeforeUpdate	Before a control or record is updated with changed data. This event occurs when the control or record loses the focus, or you click Save Record on the Records menu. This event occurs for new and existing records.
Change	When the contents of a text box or the text box portion of a combo box changes; for example, when you type a character in the control or change the Text property of the control by using a macro or Visual Basic.
Current	When the focus moves to a record, making it the current record, or when you requery a form's source of data. This event occurs when a form is first opened, and whenever the focus leaves one record and moves to another. It also occurs when you requery the source of the data for a form; for example, when you click Remove Filter/Sort on the Records menu, or use the ShowAllRecords action or the Requery action.

Delete	When a record is deleted, but before the deletion is confirmed and actually performed.
NotInList	When a value is entered in a combo box that isn't in the combo box list.
Updated	When an OLE object's data has been modified.

Error and Timing Events

Event property	It occurs
OnError (forms, reports)	When a run-time error is produced in Microsoft Access while you are in the form or report. This includes Microsoft Jet Database Engine errors, but not run-time errors in Visual Basic. (Since macros can't determine what error has occurred, you normally use Visual Basic event procedures with this event.)
OnTimer (forms)	When a specified time interval passes, as specified by the TimerInterval property of the form. You can use the Timer event to keep data synchronized in a multi user environment by requerying or refreshing data at specified intervals

Filter Events

Event property	It occurs
OnApplyFilter (forms)	When you click Apply Filter on the Records menu, or click the Apply Filter button on the command bar. This applies the most recently created filter (created using either the Filter by Form feature or the Advanced Filter/Sort window).When you click Filter By Selection after pointing to Filter on the Records menu, or click the Filter By Selection button on the command bar. This applies a filter based on the current selection in the form. When you click Remove Filter/Sort on the Records menu, or click the Remove Filter button on the command bar. This removes any filter (or sort) currently applied to the form. When you close the Advanced Filter/Sort window or the Filter by Form window.

OnFilter (forms)	When you click Filter By Form after pointing to Filter on the Records menu, or click the Filter By Form button on the command bar. This opens the Filter by Form window, where you can quickly create a filter based on the fields in the form. When you click Advanced Filter/Sort after pointing to Filter on the Records menu. This opens the Advanced Filter/Sort window, where you can create complex filters for the form.

Focus Events

Event property	It occurs
OnActivate (forms, reports)	When a form or report becomes the active window.
OnDeactivate (forms, reports)	When a different Microsoft Access window becomes the active window, but before the window becomes the active window. The Deactivate event does not occur when the focus moves to another application's window, a dialog box, or a pop-up form.
OnEnter (controls)	Before a control actually receives the focus, either from a control on the same form or when the form opens. This event occurs before the GotFocus event.
OnExit (controls)	Just before a control loses the focus to another control on the same form. This event occurs before the LostFocus event.
OnGotFocus (forms, controls)	When a control, or a form with no active or enabled controls, receives the focus. A form can get the focus only if all visible controls on a form are disabled, or there are no controls on the form.
OnLostFocus (forms, controls)	When a form or control loses the focus. A form can have the focus only if all visible controls on a form are disabled, or there are no controls on the form.

Mouse Events

Event property	It occurs
OnClick (forms, controls)	For a control, this event occurs when you press and then release (click) the left mouse button on a control. For a form, this event occurs when you click a record selector or an area outside a section or control.
OnDblClick (forms, controls)	When you press and release (click) the left mouse button twice on a control or its label. For a form, this event occurs when you double-click on a blank area or record selector on the form.
OnMouseDown (forms, controls)	When you press a mouse button while the pointer is on a form or control. Canceling the MouseDown event using the CancelEvent action in a macro for a form or control prevents the shortcut menu from being displayed when you right-click the form or control.
OnMouseMove (forms, controls)	When you move the mouse pointer over a form, form section, or control.
OnMouseUp (forms, controls)	When you release a pressed mouse button while the pointer is on a form or control.

Window Events

Event property	It occurs
OnClose (forms, reports)	When a form or report is closed and is removed from the screen.
OnLoad (forms)	When a form is opened and its records are displayed. This event occurs before the Current event, but after the Open event.
OnOpen (forms, reports)	When a form is opened but before the first record is displayed. When a report is opened but before it prints.
OnResize (forms)	When the size of a form changes. This event also occurs when a form is first displayed.
OnUnload (forms)	When a form is closed and its records are unloaded, but before it's removed from the screen. This event occurs before the Close event.

4.3 Working with command Buttons

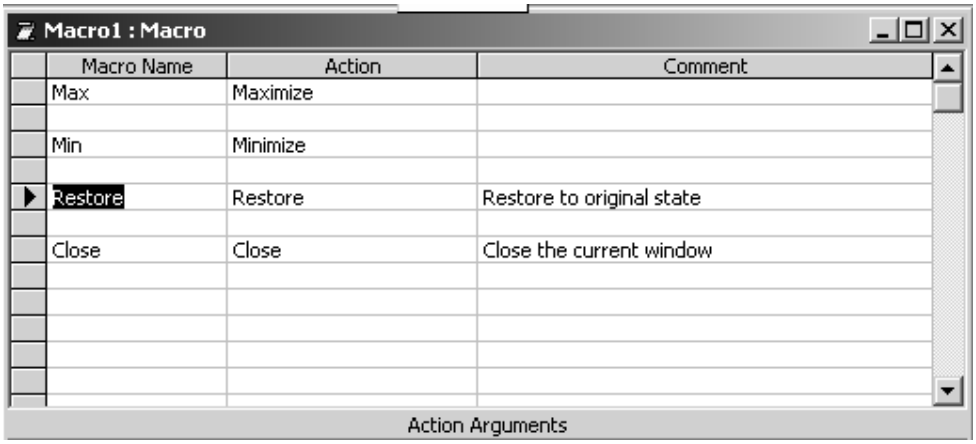
Command buttons are the type of form control that are used to run macros. You use a command button on a form to start an action or a set of actions. For example, you could create a command button that opens another form. To make a command button do something, you write a macro and attach it to the button's OnClick property.

To design Macros to maximise, minimise, restore and forms

This is a two step process, firstly we must create the macros and then attach the macros to command buttons within the form

To Create the Macro Group to maximise, minimise, restore and forms

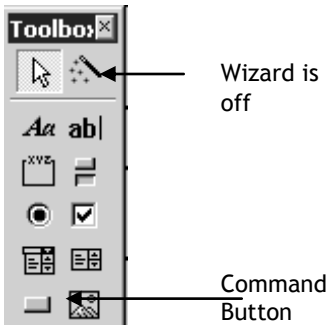
- Create a new macro
- Add the Macro Name column by clicking on the Macro Names  button
- Add Macros to the Macro Group as shown below: (There are no action arguments)



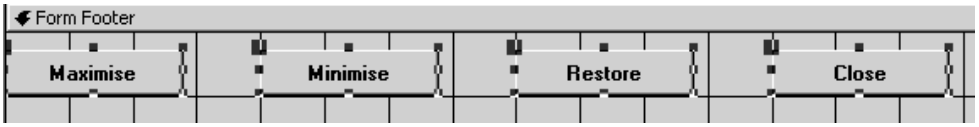
- Save the Macro Group as mcr_Form_Operations

To attach the macros to the form:

- Open the form you wish to add the macros to in Design view.
- Extend the Form Footer section to around 1 inch



- Create a command button, by clicking on the Command button on the Toolbox making sure that the wizard is deactivated and then clicking on the Form Footer
- Copy the button and create three more command buttons in the Form Footer
- Change the Caption Properties to Maximise, Minimise, Restore and Close, using the following as a guide:\

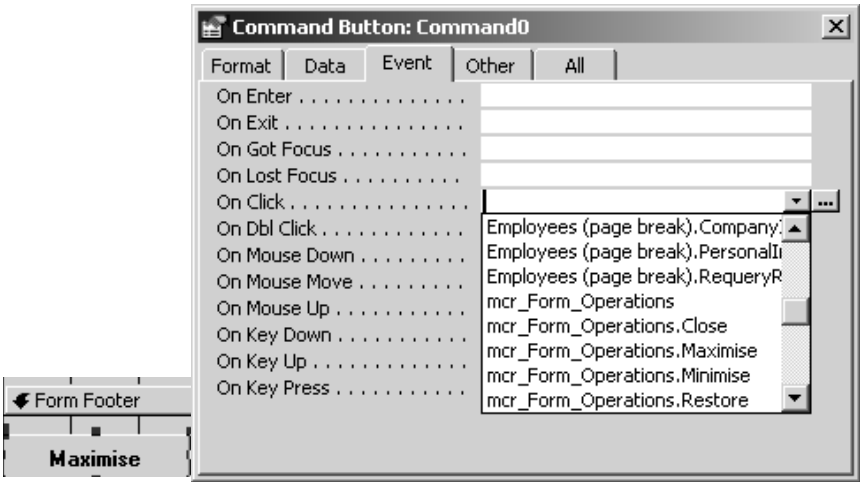


- Attach the macros to the command buttons:

Show the Properties Sheet for the Maximise Command, by right mouse clicking on the command button and choosing properties

Select the Event Properties and click into the On Click property cell.

Choose mcr_Form_Operations.Max from the drop down list of macros.



Repeat for all three remaining command buttons

. Test the Operation of the Command buttons using the Close command button last!!

. You can create over 30 different types of command buttons with the Command Button Wizard. When you use the Command Button Wizard, Microsoft Access creates the button and the event procedure for you.

. You can display text on a command button by setting its Caption property, or you can display a picture by setting its Picture property.

Use this page for Note

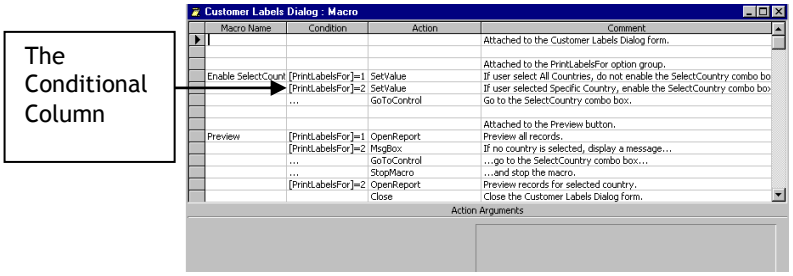
5 Conditional Macros

Section Objectives:

- Conditional Macros
- Message Box Action
- Set Value Action
- Understanding Form Events

5.1 Conditional Macros

The conditional column within the Macro design window is used to write logical expressions which act as a condition for that macro line. When the line in the macro containing the condition is reached, the action will be carried out only if the condition is true. For example, you may want to test a data field on an invoice to see if the date is before or after the due date; if the date if before, execute the macro action, otherwise do nothing.



Conditions

To test the condition, express the condition as a logical equation using one or more of the following operators:

Operator	Description
=	Equal To
>	Greater than
<	Less than
<>	Not equal to
>=	Greater than or Equal to
<=	Less than or Equal to

The “AND” and “OR” operators can also be used within the conditional expression to test for more than one condition.

AND The condition must meet both criteria.

OR The condition must meet one criteria.

Referring to Control Names in Expressions

When working with Expressions in Conditional Macros, you may need to refer to a form or report. To refer to a form or a report use the following syntax:

If a space occurs within the name of a form, or report or control, you must enclose the name in square brackets. For example, Forms![Employee Details]!Salary refers to the Salary control on the currently open form called Employee Details

The following are examples of logical conditions:

Form![Sales]![January]<100000

If the amount in the January control, on the Sales form, is less than the 100000, then……..

[January]=150000

If the amount in the January control on the current form is equal to the 150000, then…..

Form![Sales]![January]<100000 OR

Form![Sales]![January]>200000

If the amount in the January control on the Sales form is less than 100000 or greater than 200000, them……

True Results

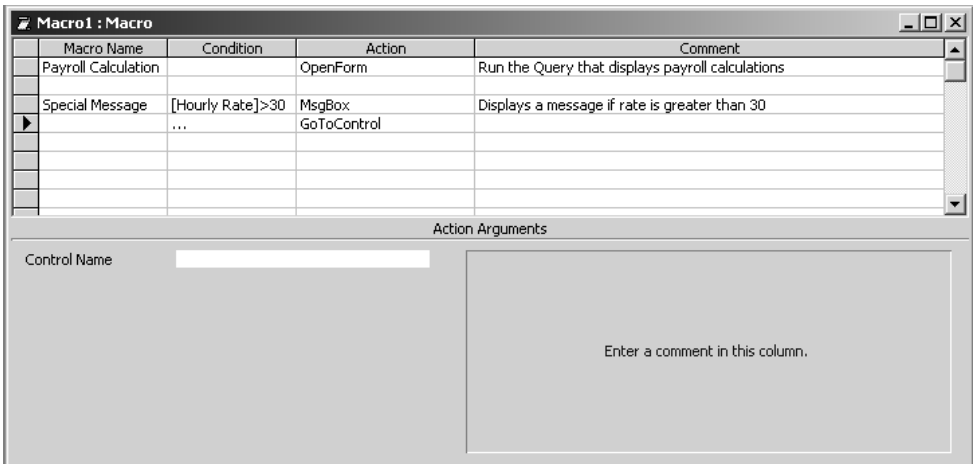
If the condition test returns a true result, Access will run the Action on the same row as the conditional expression. To run more than one Action, type an ellipsis "…" (three full stops) in the row directly below the conditional expression and then another Action on the same row as the ellipsis. Completer the same process for each additional Action.

False Results

If the conditional test returns a false result, Access will run the first Action that doesn't have an ellipsis in the conditional column.

If Access reached a blank cell in the Conditional column, the Action on the same row will be completed without being tested for a condition. If Access reached another conditional expression, it will evaluate the new condition and then continue depending on the result.

The following is an example of a conditional macro that will test the control Hourly Rate to see if it is greater than 30, if the condition is true the Message Box will be displayed and the GoToControl command will be activated.



There are further examples of expressions which can be used in Conditional Macros on page 80 at the end of this manual

To create a conditional Macro

- Open the macro in the Design View
- Choose View, Conditions or
- Click on the Conditions button
- Type in the conditional expression that you want to use when the condition is True in the Action column.
- To add another Action based on the same condition, type an ellipsis (…) on the next row, in the condition column.

5.2 The Expression Builder

The Expression Builder can be used to build the conditional expression in the conditional column.

To use the Expression Builder,

- Click into the Conditional column
- Click on the Build button in the Macro Design Toolbar
- Or
- Right-click where you want to insert the expression.
- On the shortcut menu, click Build.

. If the Condition column where you start the Expression Builder already contains a value, that value is automatically copied into the expression box.

Validating Data

Verifying that data has been entered into a control correctly can be performed through the ValidationRule property for the control or by setting record or field validation rules in the table to which the control is bound.

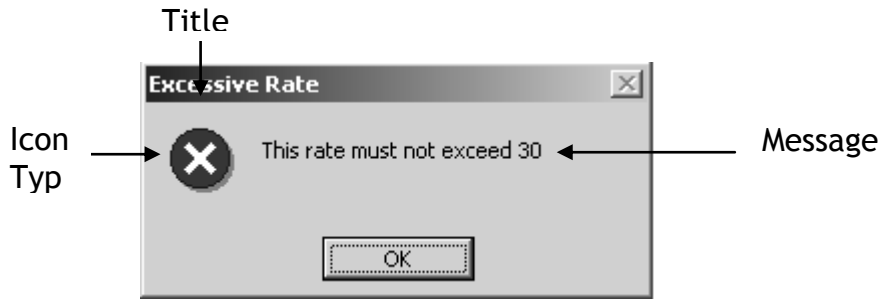
Macros provide you with additional power and can be more flexible for complex validations. Macros can be used for validation when a validation rule involves conditions for more than one value on a form, when you want to display different error messages for different types or errors in the one field or to override your validation rule etc.

Displaying Messages and Cancelling Events

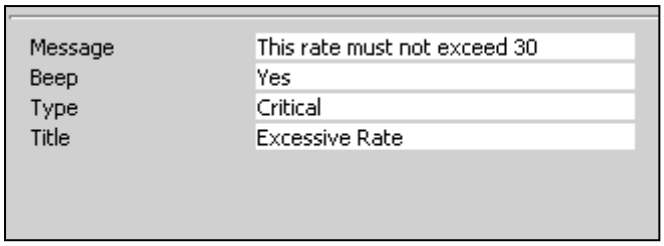
When validating data, you can use a message box to display a message that explains what has gone wrong and then cancel the event that caused the validation macro to run.

Displaying the Message

The MsgBox action is used to display box with a message. The following diagram is an example of a Message box:



The following four arguments determine how the message box will look when it is displayed:



Message The Message that you want to appear in the message box. It can contain up to 255 characters of text or an expression that begins with an equals sign.

Beep Specify whether the computer beeps when the message is displayed. Select Yes to activate the beep or No to deactivate the beep

Type Select the type of Icon to display in the message box using one of the following:

NONE	DISPLAYS NO ICON
Critical	
Warning!	
Warning?	
Information	

Title The text that you want to appear in the Title Bar of the Message box. If the title is left blank, "Microsoft Access" will be displayed

Cancelling the Event

The CancelEvent action is used to cancel the event that caused the macro to run. This action does not have any arguments, it simply cancels the event. For example, if you use the Cancel Event action with the BeforeUpdate property, it will stop Access updating the data in the table when data does not meet the condition in the validation macro.

The following table lists all the events that can be cancelled be the CancelEvent action:

Before Update	DblClick
BeforeInsert	KeyPress
BeforeDelConfirm	MouseDown
Delete	Format
Exit	Print
Open	Unload

The SetValue Action

You can use the SetValue action to set the value of a Microsoft Access field, control, or property on a form, a form datasheet, or a report.

The SetValue action has the following arguments.

Item — The name of the field, control, or property whose value you want to set. Enter the field, control, or property name in the Item box in the Action Arguments section of the Macro window. You must use the full syntax to refer to this item i.e. Forms!Employee!Bonus

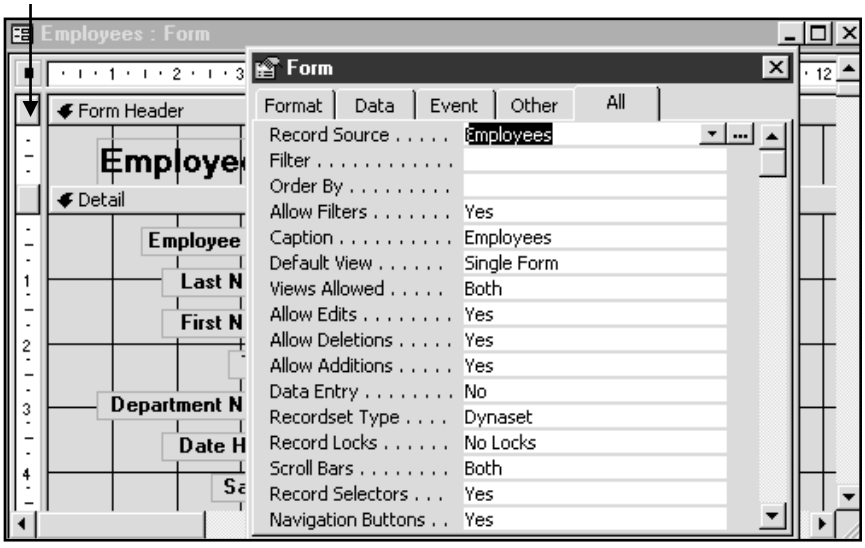
Expression — The expression Microsoft Access uses to set the value for this item. You must always use the full syntax to refer to any objects in the expression. For example, to increase the value in a Salary control on an Employees form by 10 percent, use Forms!Employees!Salary*1.1.

5.3 Understanding Form Events

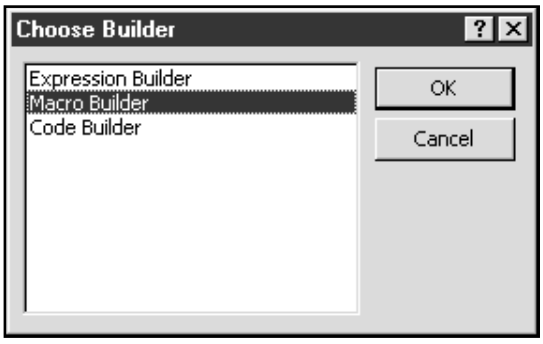
To add a Macro to a Form Event

- Open the form you wish to add the macro to in Design view.
- Right click onto the little square on the left of the ruler to select the form and select properties.

Right click here

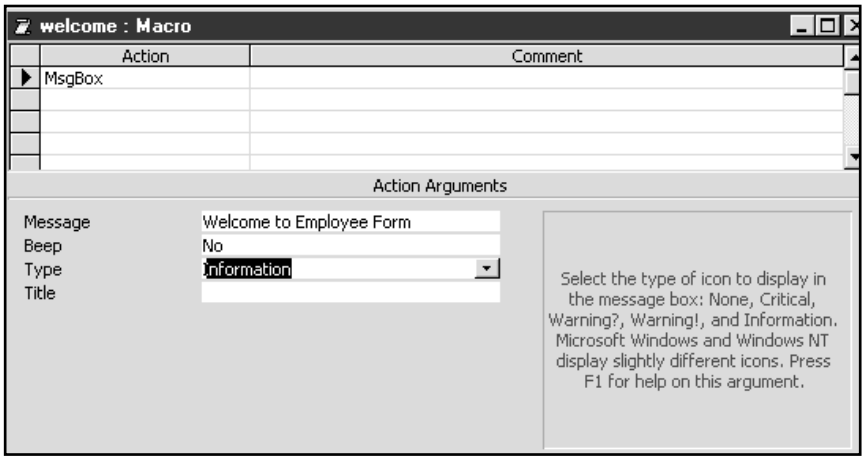


- Scroll down the form properties until you come to the property On Open… (near the bottom).
- Click onto the three dots …. Next to the On Open property and select the Macro Builder option from the following screen and press OK.



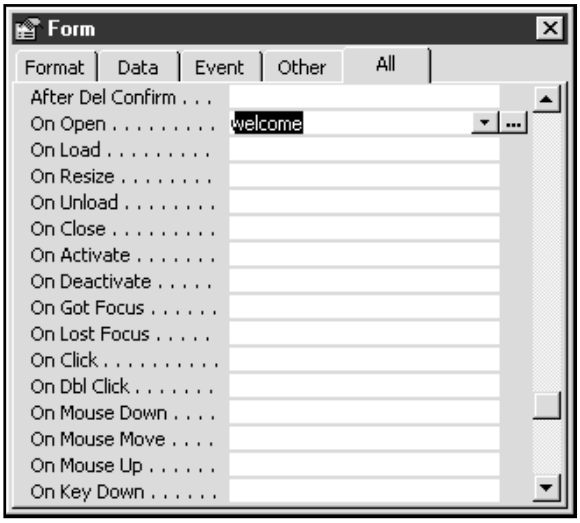
- From here, you name then create your macro as normal. In the example below, we want to create a macro that produces a welcome message every time we open the form. We have used the msgbox macro action at

the top, with the message we want displayed (and information icon we want displayed) in the arguments section at the bottom.



- Close and save the macro.

The macro name is now displayed in the On Open property event. (Notice the drop down arrow, allowing you to select a different macro +to apply to this event).



- Close the property box, switch back to form view and your message should be displayed!

. You can now choose to create a macro in its own window, or as an event in a form. The form properties allow you to create, change or add macros as often as you like.

To set a form, report, or control property by using a macro

- In a macro, add a SetValue action.
- Set the Item action argument of the SetValue action to an expression that refers to the property you want to set:
- To set a form or report property, use the syntax Forms!formname.propertyname or Reports!reportname. propertyname. For example, the following expression refers to the Visible property of the Customers form:

Forms!Customers.Visible

- Set the Expression action argument of the SetValue action to the value you want to set the property to. If the setting is a string, be sure to enclose it in double (“) quotation marks. For example, to set the Caption property of a form to Orders, you would enter “Orders” in the Expression argument.

Use this page for Notes

6 Start Up Options

Section Objectives:

- Creating a SwitchBoard
- Working with Start Up Options
- AutoExec Macro
- Splashboard

6.1 Using a Switchboard

Up to now, forms have been used to add data to populate tables, to show data from multiple tables (sub forms). There is another type of form which contains no data, only buttons that allow you to jump to other locations on a form.

This type of form is usually called a Switchboard and it usually is the first form or screen a user will see when they open the database file.

A Switchboard's primary use is as an application interface menu. The switchboard shown below is the application interface for the NorthWind Database. This Switchboard contains several command buttons. When the user clicks on any switchboard button a macro is triggered that performs some action or a series of actions.

Notice the command buttons that allow you to open other forms, queries and reports.

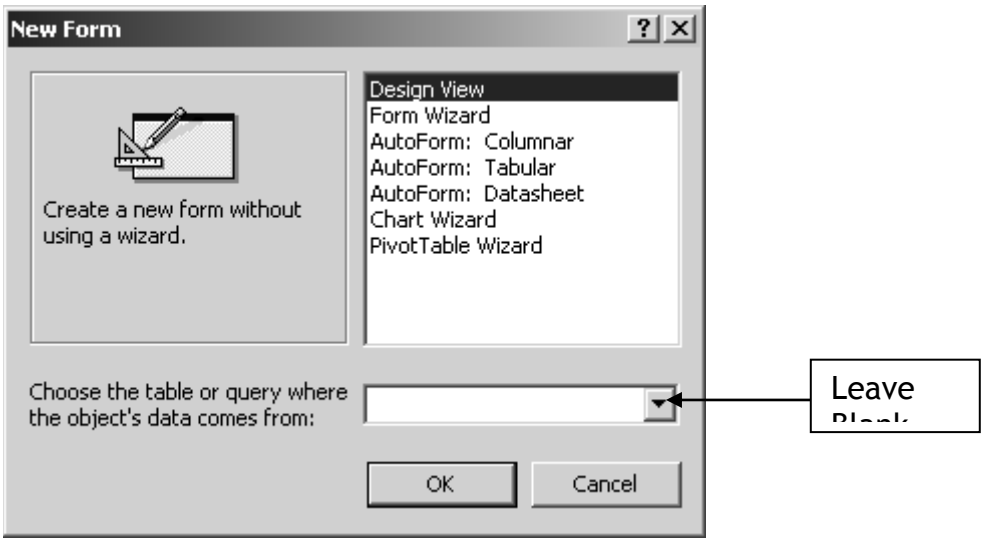
Creating a form for a Switchboard

Typically a switchboard will contain command buttons, labels, OLE objects such as pictures, lines and rectangles but will lack the other types of controls you normally see on forms such as text boxes, combo boxes etc.

To create the Switchboard, you firstly need to create an Unbound form and then add the various components to it.

To create a Switchboard Form

- Click on the Forms Tab
- Choose New
- Choose Design View and do not select any table or query to base the form on. (This is an Unbound Form)



- Resize your form
- Save your form as for example Main Menu

. We will enhance the appearance of this form later

- Click Size To Fit Form on the Window menu to size your form to fit it's contents.
- Click Save on the toolbar to save the size of the form.

Creating Command Buttons

A command buttons main purpose is to run a macro. You use a command button on a form to start an action or a set of actions. For example, you could create a command button that opens another form. To make a command button do something, you write a macro and attach it to the button's OnClick property.

Command buttons can be added to any form to:

- Open, close, print another form
- Navigate through records
- Open up and print reports
- Run a query
- Open other applications such as Word and Excel
- Automatically close down Access

There are three ways to create a command button

1. Click the Command button icon on the toolbox with the Wizard

Or

2. Click the Command button icon on the toolbox without the Wizard

Or

3. Drag a macro name from the Database container to the form

The Command Button wizard

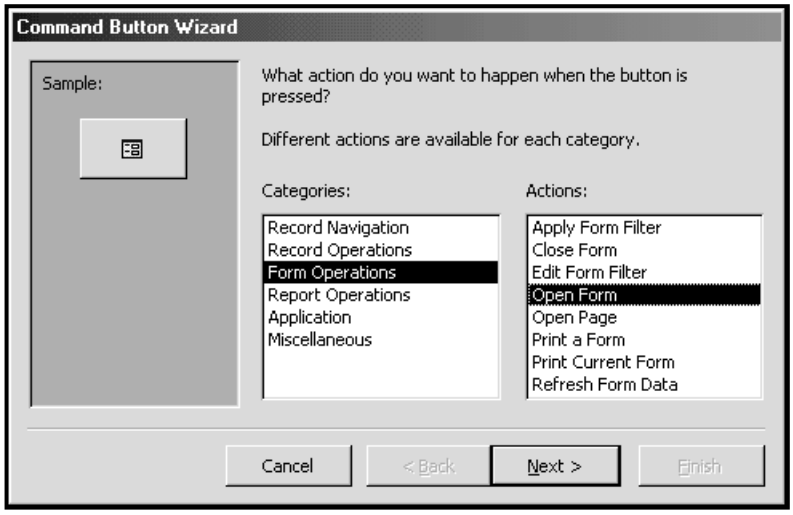
Essentially, when you create a command button with the wizard you are creating a macro instead of going into the macro window in Design View.

To Create a Command button using the wizard

Open the form you wish to add the command button to. If you are designing a new switchboard, create a form in Design view.

Ensure you are in Design View and the Wizard is turned on in the Toolbox (it should be pale grey).

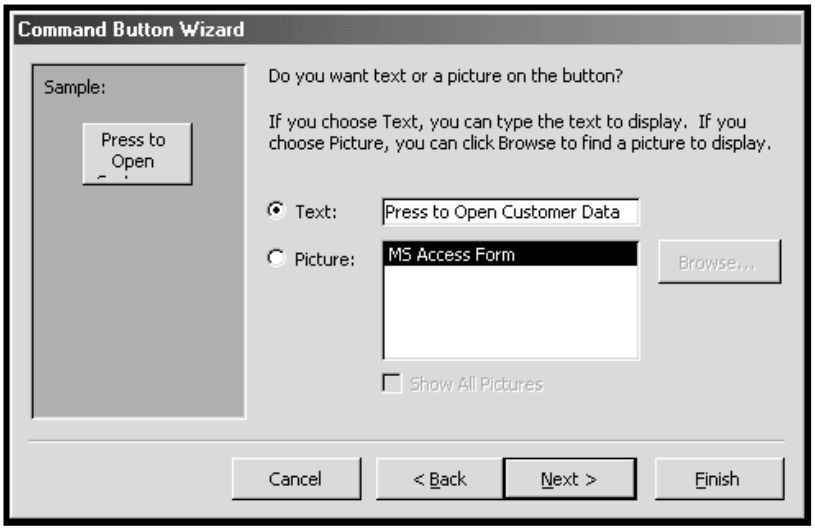
Select the command button from the toolbox and draw a small box in the place you want the button to go. The macro kicks in and displays the following:



To create a button that jumps to another form, select Form Operations from the categories on the left and Open Form from the Actions on the right. Press Next>.

Select the form you wish the button to move to and press Next>. Select the option to show all records on the form and press Next>

Type in the text you want to appear on the button or select a picture for the button. Press Next>



Assign a meaningful name to the button and press Finish.

Press the button to ensure it works!

Linking a command button to a Macro.

As soon as you create a command button in the Design window, it is already active. You can click on it, although it doesn't perform any action (unless you created it with the Wizard), it does become dimmed or sunken

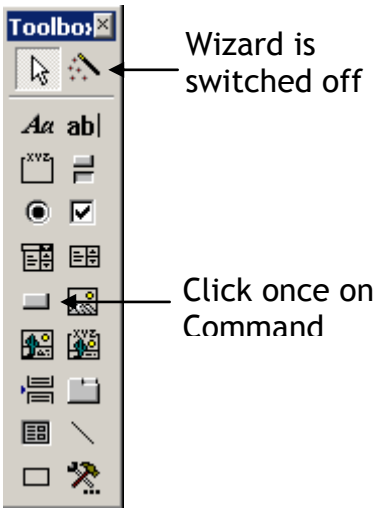
To Link the Command Button to a Macro

Firstly, create the Command Button

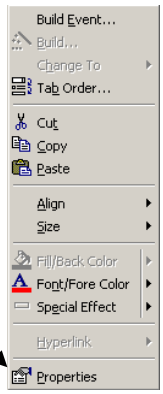
- Click on your Forms Tab
- Select the Main Menu Switchboard form and choose Design
- Ensure the Toolbox is displayed.

If it is not displayed, then click on the Toolbox button to display

- Create a command button, by clicking on the Command button on the Toolbox making sure that the wizard is deactivated and then clicking on the Form

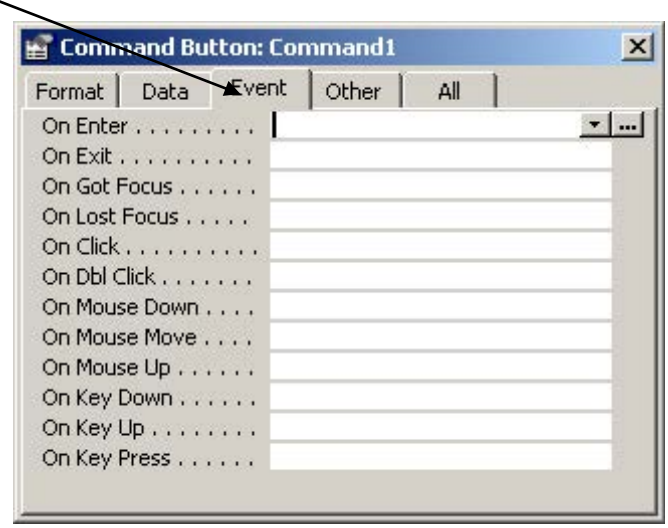


- Select the properties of the Command button by either right mouse clicking on the button and choosing Properties from the Shortcut Menu



Or

Click on the Properties button  on the toolbar. Click onto the Event Tab. The following Properties Sheet is displayed



The Property most commonly used to link a command button on a macro is the On Click. This property runs a macro whenever the user clicks on the button. To associate the macro with the On Click Event of this button follow these steps

- Click in to On Click property
- Then, click on to the drop down arrow which appears.
- Select the macro that you which to link to the Command button

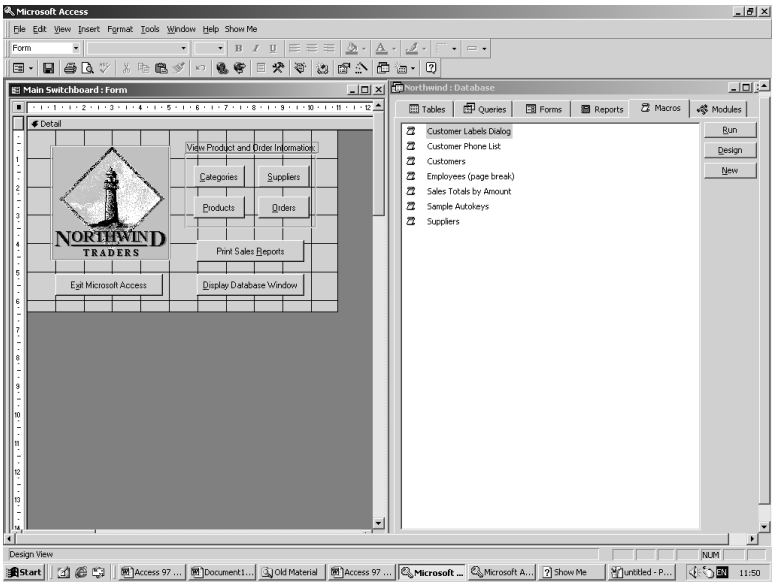
. When you enter a macro name, the macro does not have to exist. You can enter the name of the macro you create later. In this way, you can create your Switchboard first; add all of your command buttons and; your macros later. If the macros name you type for the On Click event does not exist, when you open the form and click the button the user will get an error message.

Dragging a macro name to the Switchboard

Another way to create a command button is by dragging and dropping the a macro name from the macro Database window to a position on the Switchboard

- Click on the Form tab
- Select the Switchboard form and choose Design
- Choose Windows, Tile Vertically
- Click on to the Macros Tab

Your window should look as follows:



- Now, click and drag the required macro onto the form
- Click on the button name and change it(This is the same as changing the Caption Property of the command button)

. You can also use this method with Group Macros. However, you need to the On Click Event property and chose the individual macro which you want from that group as by default only the Group name will be displayed.

6.2 Adding a Picture to a Command Button

In addition to having a command button display a caption, you can also have a Command button display any picture

› To change a command button to a picture

- Type the name of the Bitmap image into the Picture property of the Command button as below across

Or

- Click on to the ellipsis ... and chose a picture form the list or browse to a custom picture

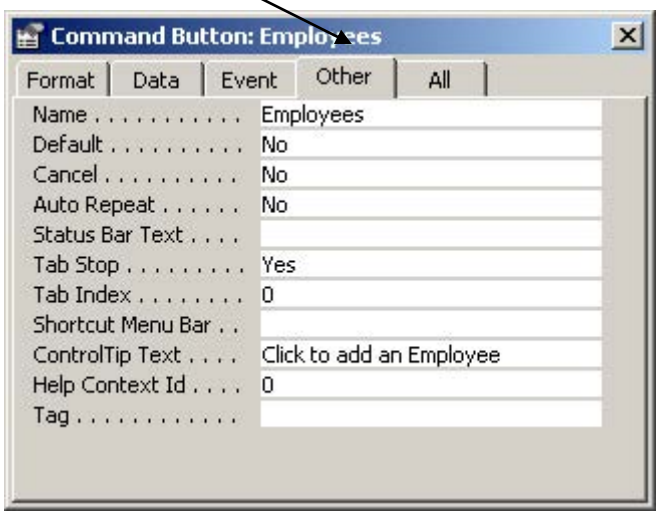
Control Tip Text

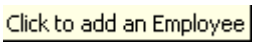
You can use the ControlTipText property to specify the text that appears in a ScreenTip when you hold the mouse pointer over a control. The ControlTipText property provides an easy way to provide helpful information about controls on a form.

› To add Control Tips to your command buttons

- Select the command button you wish to add a control tip to
- Click on to the properties button  on the toolbar

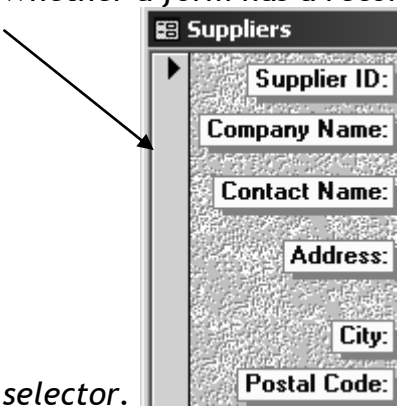
- Click on to the **Other** tab



- Click into the ControlTipText property and type e.g. Click to add as Employee
- When you are in form view and allow your mouse to hover over the command button, the text will appear as follows 

6.3 Changing the Form Properties

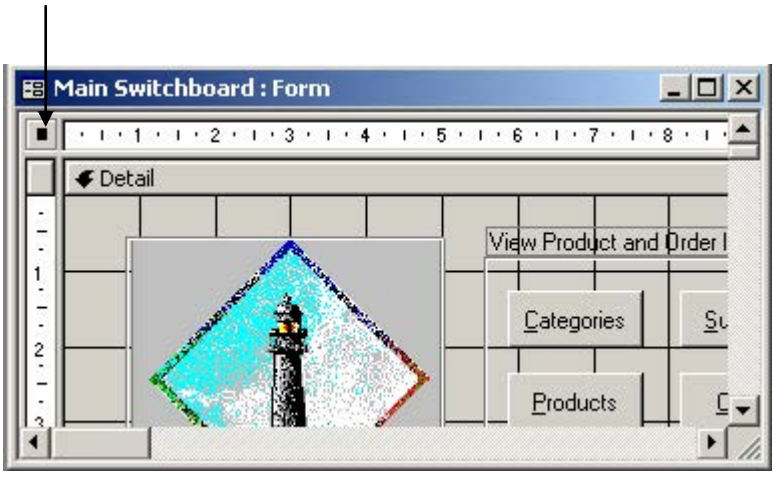
To enhance the appearance of the form, there are several form properties which can be changed. The properties and their explanations are listed in the following table

Property	**Value**	***Descriptions***
Default View	Single Form	*Whether a form is displayed in Datasheet view, as a single form (one record), or as a continuous form (multiple records).*
Views Allowed	Form	*Whether you can switch between Form view and Datasheet view.*
Scroll Bar	Neither	*Whether a form has scroll bars.*
Navigation Buttons	No	*Whether a form has navigation buttons.* 
Record Selectors	No	*Whether a form has a record selector.* 
Auto Resize	Yes	*Whether a Form opens automatically sized to display complete records*
Auto Centre	Yes	*Whether a form is centred automatically in the application window when the form is opened*
Borders Style	Dialog	*The type of border and border elements (title bar, Close button, Control menu, Maximize and Minimize buttons) to use for the form. It also determines whether the form is sizable.*
*Modal**	*Yes*	*Whether a form opens as a modal form*

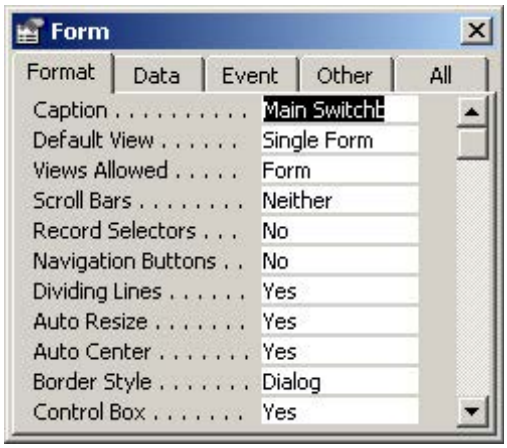
* When a form opens as a modal form, you must close the form before you can move the focus to another object. It allows the form to be displayed on top of other windows.

6.4 To change the properties of the form

- Click on the intersection of the two ruler bars in the Design View of the form



The properties of the form will now displayed



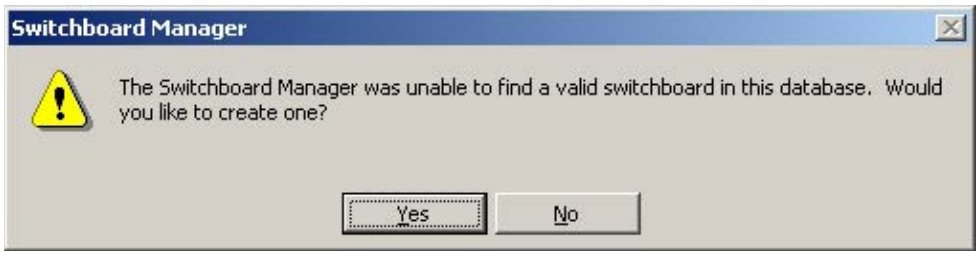
. The above properties listed in the table can be found under the Format and Other tabs.

6.5 The Switchboard Manager

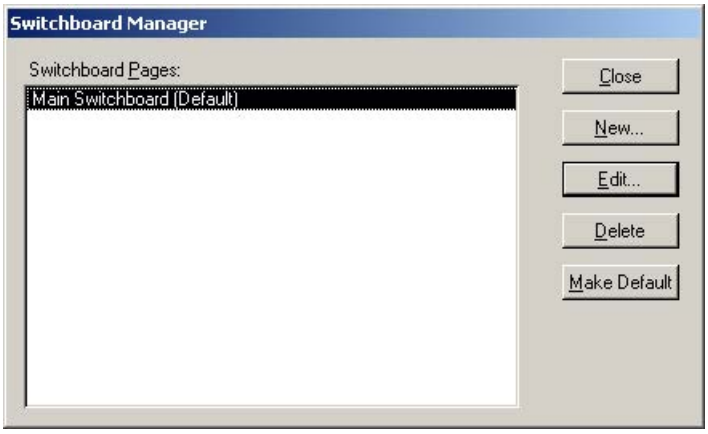
It is also possible to create a Switchboard using the Switchboard Manager

Create a switchboard form using the Switchboard Manager

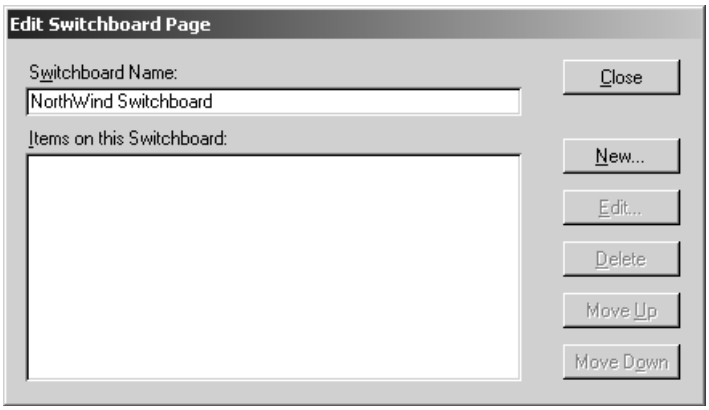
- On the Tools menu, choose Add-ins, and then click Switchboard Manager.
- If Microsoft Access asks if you'd like to create a switchboard, click Yes.



- In the Switchboard Manager dialog box, click Edit.

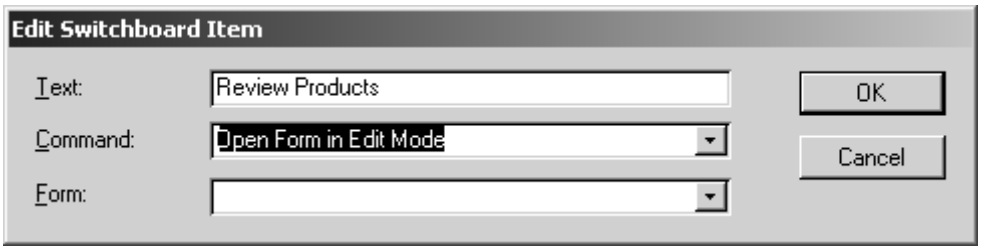


- In the Edit Switchboard Page dialog box, type a name for the switchboard in the Switchboard Name box, and then click New.



- In the Edit Switchboard Item dialog box, type the text for the first switchboard button in the Text box, and then click a command in the Command box. For example, type Review Products in the Text box, and then click Open Form in Edit Mode in the Command box.

Depending on which command you click, Microsoft Access displays another box below the Command box. Click an item in this box, if necessary. For example, if you clicked Open Form in Edit Mode in the Command box in step 5, click the name of the form you want to open in the Form box, such as Review Products, and then click OK.



- Repeat until you've added all the items to the switchboard. If you want to edit or delete an item, click the item in the Items On This Switchboard box, and then click Edit or Delete. If you want to rearrange items, click the item in the box, and then click Move Up or Move Down.
- Click Close.

Sub Switchboards

You can use the Switchboard Manager to create a switchboard that branches to other switchboards.

+To create a sub switchboard

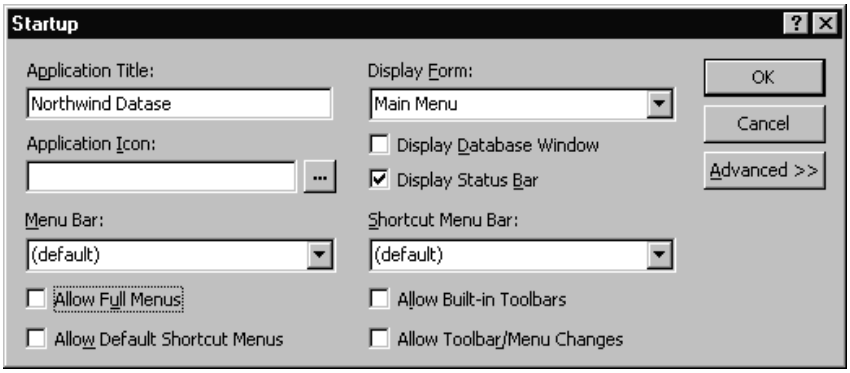
- Use the above procedure to create one or more switchboards.
- To have a switchboard branch to another switchboard, choose the Go To Switchboard command in the Command box in above procedure, and then specify the switchboard you want to go to.

Access 2002 gives you two means of controlling what happens when you start access and open a database

1. Startup options
2. AutoExec macros

Startup options

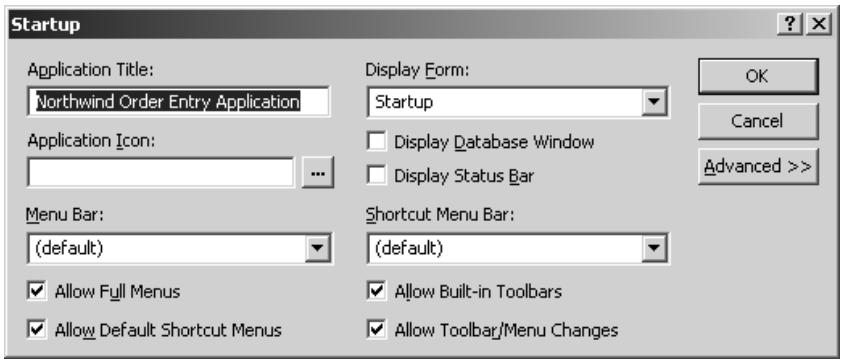
Startup options enable you to customise the application title bar, menus, toolbars, and startup form. The startup options apply only to the current database or application and are set by choosing Tools, Startup. When you define settings in the Startup Dialog box, you are setting the database properties



The menu Bar and Shortcut Menu Bar options in the Startup dialog box do not override the property settings for individual forms and reports that have a custom menu bar or shortcut menu. Therefore you can set global menu options in the Startup dialog box and override them in individual forms and reports.

To display a Switchboard automatically when you open your database

- Choose Tools, Startup.



- In the Display Form box, select your Main Menu Switchboard.
- If you don't want users to see or use the Database window, which appears behind the form, clear the Display Database Window check box.

. Changes to these settings in the Startup dialog box won't take effect until the next time the database or application is opened.

. To make a switchboard the switchboard that's automatically opened when you open the database, click the switchboard name in the Switchboard Manager dialog box, and then click Make Default

Bypass settings that determine how a database or application starts

If you used the Startup dialog box on the Tools menu or created an AutoExec macro to specify what happens when a database or application starts, you can bypass those settings to regain full access to the database or application.

To bypass startup settings

- Hold down the Bypass key (the SHIFT key) while you open the database.

The AutoExec Macro

The AutoExec can also be used to control the way a database behaves when it is opened. An AutoExec macro runs after the Startup options have taken effect. It is a macro which automatically executes or runs every time you start up your database.

Often people create an autoexec macro, with actions to open the main switchboard form, hide the toolbars and menus, maximise the form and show a welcome message.

> Note: you can use the Tools\Startup menu to automatically load the switchboard form and hide the menus. However, you cannot ensure the form will be maximised or provide a user prompt with this method.

- To create an AutoExec macro:
  - Create a new macro from the Macro tab in the main database window.
  - Set the appropriate actions as normal. For example, open the switchboard, maximise and display a messagebox.
  - Save the macro giving it the name Autoexec (all one word).
  - Close the database file down then open it back up again.

You will see the macro runs automatically when you open the database!

> You can use the Startup dialog box instead of or in addition to an AutoExec macro. An AutoExec macro runs after the Startup options have taken effect; therefore, you should avoid any actions in an AutoExec macro that change the effect of the Startup option settings. For example, if you specify a form in the Display Form box in the Startup dialog box, and you also use the OpenForm action in an AutoExec macro, Microsoft Access first displays the form specified in the Startup dialog box, then immediately displays the form specified in the OpenForm action.

To open the database and bypass the autoexec macro, you must hold the Shft key on the keyboard down whilst clicking on the open button. Use the same method to open a database and bypass the startup options you set in the Tools\ startup options menu.

6.6 Splashboard

A Splashboard is a form that appears for a specified time interval when the Access database file is opened initially. It displays some general information regarding the application, such as when the application was created, who created it etc.

The Splashboard itself is a form, which is set to automatically open by creating a macro to open the Splashboard form. This macro is called Autoexec to ensure that it is automatically opened when the application is opened.

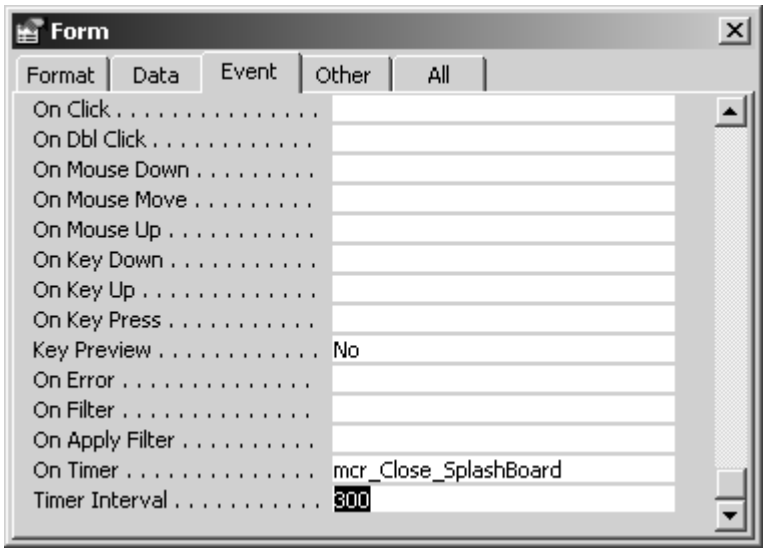
It is also necessary to create a macro to close this form after a certain time interval. We achieve this by using the On Timer and the Interval Events

To Create a Splashboard

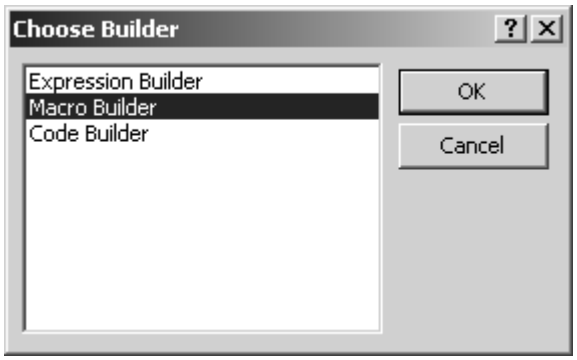
- Create a blank form with no control source.
- Add the required information to your Splashboard, using the Toolbox icons, Label Aa and Text Box ab|
- Format the form as you might a Start Up form
- Save the form

To cause the Splashboard to display for a timed period on opening the application

- Display the Properties of the form



- Click on the Event Tab
- Click in the On Timer, click on to the … and select Macro Builder and select OK



- Create a macro which will close the Splashboard form
- Save and close the macro
- Now, select the Timer Interval event and specify the time period that you would like to display the Splashboard form for.

The TimerInterval property setting of the form specifies the interval, in milliseconds, between Timer events. It is measured in milliseconds and can be between a and 65,535

. Don’t forget to create the macro to Open the Splashboard form and save t

Use this page for notes

Appendix A

Category	Task	Action
Data in forms and reports	Restrict data	ApplyFilter
Move through data	FindNext, FindRecord, GoToControl, GoToPage, GoToRecord	
Execution	Carry out a command	RunCommand
Exit Microsoft Access	Quit	
Run a macro, procedure, or query	OpenQuery, RunCode, RunMacro, RunSQL	
Run another application	RunApp	
Stop execution	CancelEvent, Quit, StopAllMacros, StopMacro	
Import/export	Send Microsoft Access objects to other applications	OutputTo, SendObject
Transfer data between Microsoft Access and other data formats	TransferDatabase, TransferSpreadsheet, TransferText	
Object manipulation	Copy, rename, or save an object	CopyObject, Rename, Save
Delete an object	DeleteObject	
Move or resize a window	Maximize, Minimize, MoveSize, Restore	
Open or close an object	Close, OpenForm, OpenModule, OpenQuery, OpenReport, OpenTable	
Print an object	OpenForm, OpenQuery, OpenReport, PrintOut	
Select an object	SelectObject	
Set the value of a field, control, or property	SetValue	

Appendix A (cont....)

Update data or the screen	RepaintObject, Requery, ShowAllRecords	
Miscellaneous	Create a custom menu bar, a custom shortcut menu, global menu bar, or global shortcut menu	AddMenu
Set the state of menu items on a custom menu bar or global menu bar	SetMenuItem	
Display information on the screen	Echo, Hourglass, MsgBox, SetWarnings	
Generate keystrokes	SendKeys	
Display or hide the built-in or custom command bar	ShowToolbar	
Sound a beep	Beep	

Appendix B

Action	Description
AddMenu	Adds a menu to a custom menu bar for a form or report. Each menu on the bar requires a separate AddMenu action
ApplyFilter	Applies a filter or query to a table, form, or report.
Beep	Causes the computer to beep
CancelEvent	Cancels the event that caused the macro to run
Close	Closes the specified window or the active window if none is specified
CopyObject	Copies the specified database object to a different Microsoft Access database or the same database with a new name
Delete Object	Deletes the specified object or the object selected in the Database window if no object if specified
Echo	Hides or shows the results of a macro while it runs
FindNext	Finds the next record that meets the criteria specified with the most recent FindRecord action or Find dialog box. Use to move successively through records that meet the same criteria
FindRecord	Finds the first or next record that meets the specified criteria. Records can be found n the active form or Datasheet.
GoToControl	Selects the specified field on the active Datasheet or form
GoToPage	Selects the first control on the specified page of the active form

Appendix B (cont...)

GoToRecord	Makes the specified record the current record in a table, form, or query. Use to move to the first, last, next, or previous record
HourGlass	Changes the mouse pointer to an hourglass while the macro runs
Maximise	Maximises the active window
Minimise	Minimises the active window
MoveSize	Moves and/or changes the size of the active window.
MsgBox	Displays a message box containing a warning or informational message
OpenForm	Opens a form in Form view, Design view, Print Preview, or Datasheet view.
OpenModule	Opens the specified Visual Basic module in Design view
OpenQuery	Opens a query in Datasheet view, Design view, or Print Preview
OpenReport	Opens a report in Design view, Print Preview, or prints the report immediately
OpenTable	Opens a table in Datasheet view, Design view, or Print Preview
OutputTo	Exports the specified database object to a Microsoft File (.xls), rich-text file (.rtf), text file (.txt), or HTML file (.html)
Printout	Prints the active datasheet object. You can print datasheets, reports, forms, and modules
Quit	Quits Microsoft Access
Rename	Renames the specified object
RepaintObject	Completes any pending screen updates or pending recalculations of controls on the specified object, or on the active object if none is specified

Appendix B (cont....)

Requery	Forces a requery of a specific control on the active database object, or the object if no control is specified
Restore	Restores a maximised or minimised window to its previous size
RunApp	Starts another program, such as Microsoft Excel or Word.
RunCode	Runs a Visual Basic Function procedure
RunCommand	Runs a command from Microsoft Access's menus – for example File – Save
RunMacro	Run a macro
RunSQL	Runs the specified SQL statement for an action query
Save	Saves the specified object , or the active object if none is specified
SelectObject	Selects a specified database object. You can then run an action that applies to that object
SetMenuItem	Sets the state or menu items (enabled or disabled, checked or unchecked) on custom menus. Works only on custom menus created using menu bar macros.
SetValue	Set a value for control field, or a property on a form or report
SetWarnings	Turns all system messages on or off. This has the same effect as clicking OK or Yes in each message box
ShowAllRecords	Removes any applied filter from the active table, query or form.
ShowToolbar	Shows or hides a built-in toolbar or a custom toolbar
StopAllMacros	Stop all currently running macros

Appendix B (cont…)

StopMacro	Stops the currently running macro. Use to stop a macro when a certain condition is met
TransferDatabase	Imports or exports data to or from the current database to or from another database
TransferSpreadsheet	Imports data from a spreadsheet file into the current database or exports data from the current database into a spreadsheet file.
TransferText	Imports data from a text file into the current database or exports data from the current database into a text file

Appendix C

Examples of macro conditions

Use this expression	To carry out the action if
[City]="Paris"	Paris is the City value in the field on the form from which the macro was run.
DCount("[OrderID]", "Orders")>35	There are more than 35 entries in the OrderID field of the Orders table.
DCount("*", "Order Details", "[OrderID]=Forms![Orders]![OrderID]")>3	There are more than three entries in the Order Details table for which the OrderID field of the table matches the OrderID field on the Orders form.
[ShippedDate] Between #2-Feb-1995# And #2-Mar-1995#	The value of the ShippedDate field on the form from which the macro is run is no earlier than
2-Feb-1995 and no later than	
2-Mar-1995.	
Forms![Products]![UnitsInStock] <5	The value of the UnitsInStock field on the Products form is less than five.
IsNull([FirstName])	The FirstName value on the form from which the macro is run is Null (has no value). This expression is equivalent to [FirstName] Is Null.
[Country]="UK" And Forms![SalesTotals]![TotalOrds] >100	The value in the Country field on the form from which the macro is run is UK, and the value of the TotalOrds field on the SalesTotals form is greater than 100.
[Country] In ("France", "Italy", "Spain") And Len([PostalCode])<>5	The value in the Country field on the form from which the macro is run is France, Italy, or Spain, and the postal code isn't five characters long.
MsgBox("Confirm changes?",1)=1	You click OK in a dialog box that the MsgBox function displays. If you click Cancel in the dialog box, Microsoft Access ignores the action.