CHAPTER 9

Using ActiveX Controls

ActiveX® controls extend the functionality of Microsoft® Access. You can use Microsoft Access-compatible ActiveX controls to enhance your applications. This chapter describes ActiveX controls and explains how to add them to your application.

Chapter Contents

Overview of ActiveX Controls

Registering an ActiveX Control

Adding an ActiveX Control to a Form

Manipulating ActiveX Controls

Overview of ActiveX Controls

In addition to the standard built-in controls that appear in the toolbox, Access supports ActiveX controls, formerly called OLE controls or custom controls. An ActiveX control, like a built-in control, is an object that you place on a form to display data or perform an action. However, unlike a built-in control, the code that supports the ActiveX control is stored in a separate file or files which you must install in order to use the control.

The following ActiveX controls are available for you to use with Access:

- The Calendar control, which makes it easy to display and update a monthly calendar on a form. You can choose to install this control when you install Access.

There are more than 100 other ActiveX controls available in Microsoft Office 2000 Developer, and any number of controls are available from third-party vendors.

Some ActiveX controls, including the Calendar control, are automatically registered with the system when you install the control files. Others must be registered manually. After the control has been registered, you can add it to a form that is open in Design view.

You can use the methods, events, and properties associated with a control to manipulate the control's appearance and behavior. In addition, many ActiveX controls can be data-bound—they can store or display data from a field in the form's underlying table or query.

Note If you distribute an application that uses ActiveX controls, you must make sure that the controls are installed on each computer that runs your application.

Support for ActiveX Controls

Other applications that support ActiveX technologies, such as Microsoft Word, Microsoft Excel, or Microsoft Visual Basic® for Applications (VBA), can also support ActiveX controls just like Access. However, each application may support a different subset of ActiveX technologies. For this reason, controls that work in some applications may not work in others.
For example, if you’ve installed VBA or Office and the ActiveX controls they include, you may have several ActiveX controls that don’t work in Access. When you add a control to a form, you may see controls in the list of ActiveX controls that aren’t compatible with Access. Using a control that hasn’t been certified for use with Access can cause unpredictable results.

To determine whether a specific ActiveX control is compatible with Access, see the documentation provided with the control or contact the vendor of the control. In addition, if you insert an ActiveX control on an Access form and get the “No Object in this Control” error, you most likely have selected a control that is not supported.

**See Also** For more information about which ActiveX controls are compatible with Access, type **ActiveX controls** in the Office Assistant or on the **Answer Wizard** tab in the Help window, and then click **Search**.

### Types of ActiveX Controls

There are two types of ActiveX controls. The first type is visible both at design time and at run time. An example of this type of ActiveX control is the Calendar control included with Access. The Calendar control displays a calendar for a particular month in a particular year, as shown in the following illustration.

![Calendar Control Illustration](image)

After you install the required files, you can place this control on your form. You can use the methods, events, and properties of the Calendar control to:

- Display a calendar for any month of any given year.
- Store a date selected by the user in the current record.
- Trigger an action from an event based on the current date value.

The second type of ActiveX control is visible only at design time. Typically, this type of control is provided to simplify interaction between an Access application and one or more dynamic-link libraries (DLLs) or executable files. The Microsoft ImageList control and the Microsoft Common Dialogs control included in Office 2000 Developer are examples of this type of ActiveX control.

### Registering an ActiveX Control

Some ActiveX controls are automatically registered with the operating system when the ActiveX control (.ocx) files are installed. If the ActiveX control is not automatically registered, then you need to register it before you can use it on a form.

To determine whether an ActiveX control must be registered, open a form in Design view and click **ActiveX Control** on the **Insert** menu to open the **Insert ActiveX Control** dialog box. If the control you want to use is included in the list, you can add it to a form. If it’s not listed, you must register it first.
To register an ActiveX control

1. On the Tools menu, click ActiveX Controls.
   The ActiveX Controls dialog box appears.
2. Click Register.
3. In the Add ActiveX Control dialog box, click the corresponding ActiveX control (.ocx) file, and then click Open.

Adding an ActiveX Control to a Form

After an ActiveX control has been registered, you can add it to a form.

To add an ActiveX control to a form

1. Open the form in Design view.
2. Click the More Controls tool in the toolbox, or click ActiveX Control on the Insert menu.
   A menu appears that lists all of the registered ActiveX controls in your system.
3. In the list, click the ActiveX control you want to add.
   Note Some of the controls that appear in the list of ActiveX controls may not be compatible with Access. Before adding a control to a form, be sure to verify that it is compatible. For more information, see “Support for ActiveX Controls” earlier in this chapter.
4. If you used the More Controls tool in the toolbox, click on the form where you want to place the control. If you used the ActiveX Control command on the Insert menu, the control is automatically placed on the form for you.
   Access creates the control.
   Tip If you plan to use the control frequently, you can add it to the toolbox or to any toolbar. For more information about customizing toolbars, see Chapter 1, “Creating an Application.”

Manipulating ActiveX Controls

An ActiveX control is an object with three distinct categories of attributes: methods, events, and properties.

A property is a control characteristic such as color, width, height, or font. You can set the properties for an ActiveX control in VBA code or in the control’s property sheet. If an ActiveX control provides its own custom properties dialog box, you can also set the control’s custom properties there. To open the custom properties dialog box for an ActiveX control, click the Custom property box in the control’s property sheet, and then click the Build button next to the Custom property box, as shown in the following illustration.
Using Methods, Events, and Properties in Code

As a developer, you can use the methods associated with an ActiveX control to manipulate that control. For example, you can use the **Refresh** method of the Calendar control to repaint the calendar.

```vba
Forms!EntryForm!Calendar.Refresh
```

You can use the events associated with a control to update the control or to synchronize the control with a record, section, or environment. For example, you can use the Click event of the Calendar control to start a procedure that retrieves the specified date and sets the value of a control on a different, synchronized form.

You can use the properties associated with a control to display or retrieve data. For example, you can use the **Value** property of the Calendar control to retrieve the date selected by the user.

The following example uses the Calendar control’s Click event and **Value** property to assign a chosen date from the Calendar control to a control on another form. The Click event occurs when the user clicks a date in the control. The code then uses the **Value** property to retrieve the selected date.

```vba
Private Sub Calendar_Click()
    Forms!SyncForm!DateVal = Forms!EntryForm!Calendar.Value
End Sub
```
To limit the dates a user can select with the Calendar control, you can use the BeforeUpdate event to examine the requested date and then either accept or reject the requested date, depending on whether it falls within the acceptable range.

Private Sub Calendar_BeforeUpdate (Cancel As Integer)
    Dim dteFirstDate As Date               ' Starting date.
    Dim dteLastDate As Date                  ' Ending date.
    Dim dteReqDate As Date                  ' User's requested date.
    On Error GoTo ErrorHandler
    dteFirstDate = #1/1/90#                  ' Assign starting date.
    dteLastDate = #1/1/99#                  ' Assign ending date.
    ' Retrieve requested date.
    dteReqDate = Forms!EntryForm!Calendar.Value
    If dteReqDate < dteFirstDate Then      ' If user requests invalid date.
        MsgBox "That date is invalid.", vbInformation, "Date Check"
        Cancel = True                          ' Cancel the update.
    ElseIf dteReqDate > dteLastDate Then
        MsgBox "That date is invalid.", vbInformation, "Date Check"
        Cancel = True
    End If
    Exit Sub
ErrorHandler:
    MsgBox "Error #: " & Err.Number & vbCrLf & Err.Description
End Sub

See Also  For information about the methods, events, and properties of the Calendar control, type calendar control in the Office Assistant or on the Answer Wizard tab in the Help window.

You can access the events for an ActiveX control through the Procedure box in the Code window. In form Design view, click Code on the View menu, then click the ActiveX control in the Object box. The Procedure box lists the event procedures for the ActiveX control, as shown in the following illustration.
**Note**  In some cases, the events for an ActiveX control have names that are identical to Access events. In this situation, the Access event is assigned the original event name, and the ActiveX control event is assigned the event name concatenated with the word “Object.” For example, if the control supports an Enter event, then the Procedure box for the control contains an Enter event and an EnterObject event.

**Data Binding Support**

Some ActiveX controls can be bound to a single column in a form’s underlying recordset. For example, the Calendar control can be bound to a Date/Time field. By setting the control’s **ControlSource** property to the name of the field you want to bind the control to, you can display, edit, and update the value in a field of the table or query that is named in the form’s **RecordSource** property.

**Note**  Your form must be in Single Form view to use data-bound ActiveX controls. This means that you can’t set a form’s **DefaultView** property to Continuous Forms if that form contains a data-bound ActiveX control. Additionally, you can’t access ActiveX controls at all in Datasheet view.

---

Send feedback to MSDN

© 2002 Microsoft Corporation. All rights reserved.