

Implementing An ADO Data Control With VB6

Contributed by Larry Gomez
Monday, 19 March 2007
Last Updated Saturday, 19 January 2008

The ADO data control can save Visual Basic developers hours of time. In this article Susan shows us exactly how to go about implementing an ADO control. You might say that the data control is a Visual Basic orphan; some developers refuse to even use it.

{mosgoogle}

Why? They find it limiting. On the other hand, I believe in using the least effort possible. If a simple control will do the trick, I'm all for it. In this series of articles we're going to review database programming from the ground up. The data control is a good place to start.

Once you know its assets and limitations, you can decide for yourself whether the data control is for you. Ask a group of VB developers how they feel about the data control, and you'll get a variety of responses. Some use the control quite a bit while others never use it at all.

Why the discrepancy between users? Those who don't use it tell us that the control isn't all that flexible. It seems the control has a hard time growing with an application. Frankly, the majority of my applications never come back to haunt me like this.

When they do, it's for added functionality. Rarely do I actually have to rewrite existing features. I'm told I'm part of a small minority. For what it's worth, I like the data control--especially now that version 6.0 supports ADO. In this article we're going to work with the ADO data control in Visual Basic 6. We will learn about ADO, how to create a bound form, and more.

The ADO data control presents a fast and easy way to create a bound form by providing built-in functions that define, navigate, display, add to, and update the records in a recordset.

The resulting form makes it easy for a user (non-developer) to maintain records. Although the data control defines the recordset, you'll still need a control for each field to automatically display the bound records.

The image above displays a simple form with several bound controls. The data control at the bottom of the form contains four navigation buttons.

These buttons, from left to right, allow you to move to the first, previous, next, and last records in the form's recordset. If you update a record, VB will save that change when you move to another record. If you close the form before moving to another record, VB won't save changes you've made to the current record.

Creating A Bound Form

Now, let's create the form shown in the previous image. First, launch a new standard .exe project, insert a form, and change that form's name to frmDataControl. Then, save the project as DataControl.

[Read more](#)

