I'm not robot

reCAPTCHA

Continue

I'm not robot

reCAPTCHA

Continue

Event driven programming in vb

not work at all with Access 2000, even with the VB Service Pack. If you want to use it you will have to convert the database. The Data Control To begin the application, we will first create a new form for Projects maintenance: ProjMaint. The first control we will place on the form, once we've set the basic form properties and saved it, is called the Data Control. It is the object which links a form to a database and allows you to access the fields in the tables making up the database. It's called Data in the Toolbox. VB provides other objects that allow you to link to databases. ADO (ActiveX Data Objects) are certainly more powerful and more efficient than the Data Control. However, they do require a lot more coding and are more difficult to implement. Also, they are not available in the Standard Edition of VB, only in the Professional and Enterprise Editions. In simple applications, the Data Control, slow as it is, gives you a tool that is easy to implement and will provide most of the functionality you need. 81 | P a g e The arrow buttons on the control are used to navigate through the database records: First record and Previous Next and Last record The buttons correspond to 4 methods of the DC which you can use when you have to navigate using code. They are: MoveFirst MovePrevious MoveNext MoveLast Let's look at the important properties of the Data Control: Name: the name to use in code - Data1 is default - eventually we'll have several data controls on the form - we'll call this one dta_proj. Connect: the kind of database - in this case it's Access - could be Foxpro, dBaseIV, etc. DatabaseName: the name and path of the database the control is connected to. RecordSource: the name of the database table being used. 82 | P a g e BOFAction and EOFAction: action to take when trying to read before the beginning of file or past the end of file - we'll look at those later. Recordset: this is a run time property, and it's an important one - it represents the result of the query executed to the database - it contains all the records required by this Data Control - when you navigate through the database, you are actually navigating through the recordset, which is then mapped back to the database - that is why the methods of the Data Control refer to the Recordset property. Next we add the controls needed to look at the fields in the records. In many instances we will need to make changes to the data. Therefore, we'll use a TextBox for each of the fields so that we can both diaplay and enter data as needed. Each TextBox will be a bound control, meaning that it is bound or tied to a specific field from the database. When we navigate through the database using the arrow buttons the content of each TextBox will always reflect the content of the current field. To bind the control to the database field we use its Data properties: DataSource is the name of the Data Control - remember that the DC specifies the name of the database to use and the name of the table to access - tip: enter this one before the DataField. DataField is the name of the field to bind - that field is selected from the content of the table. Top 85 | P a g e validation. If there is an error, you evoke the SetFocus method to put focus back to the control with the error. Finding a specific record When you navigate with the arrow buttons or the Move... methods you are necessarily moving one record at a time. Very often there is a need to access a specific record in the database. For example, it might be to change the ending-date for the project called "XYZ Corp. Payroll System". In this example we assume that the search will be conducted on Project title. It could be on Number or End-date if necessary and it would just involve minor changes to the code. We also assume that the user does not want to enter the full project title and will only input the first few characters; we will therefore make use of the "Like" operator to match the recordset to the search string. First, we create a new TextBox, called txt_findTitle, to enter the search string. We will give this TextBox the TabIndex 0 because we want it to be the first control read when we look at a record. As soon as we move off the TextBox, the LostFocus event is triggered and checks whether the user has entered a search string or not. If there is no input into the search string, the user can work with the current record in the form. If there is a search string specified, the appropriate record will be loaded into the form. The FindFirst method of the DC will locate the first occurence in the recordset matching the "content" parameter. If there are more than one records that match, the user then navigates forward using the arrows. The format of the FindFirst method is: 86 | P a g e DataControl.Recordset.FindFirst "fieldname = 'searchstring'" If the fieldname contains a string value, you have to use single quotes to name the searchstring; you can use the other comparison operators in place of the =. This technique can be adapted to search any field in the recordset for a specific record. There are also FindNext, FindPrevious and FindLast methods for the Data Control recordset. LESSON 10 - Working with a database ...part 2 Tuesday, August 02, 2011 87 | P a g e Using multiple tables Our ProjectMgt application contains an Employee table and a Department table linked through the employee's department number. Now, if we create a form for Employee maintenance using the same technique we used in the previous lesson, we can access all fields in the Employee table using one data control, Data1. This is what the basic form will look like, before we get to put in the usual improvements: 90 | P a g e Now to get the list right. First, we delete the department_number TextBox. Then we add a DBList. Now we specify the DBList1 properties. Careful! This is where most people hit a snag! The Data properties: these specify where the data entered will be stored. We are in the Employee table. That's Data1. So, the data entered will go into DataSource: Data1 and the field into which it is going is DataField: e-Dept. The List properties: these tell the control where to get the information to show in the list. Since we want it from the Department table, we specify RowSource: Data2. What will appear in the list is the Department name so we choose ListField: d_Name. Finally, there has to be a link between Data2 and Data1. That is always the field which is the primary key in the list table and that is the BoundColumn: d_Number. 91 | P a g e And once everything is cleaned-up, the Data2 control is hidden, we get the final result: Pare Turse Systems Analysis Network support G3000 3120 Employee table 92| Page