Tutorial 2 - Database lookup

This tutorial will show you how to add a voice system to an existing or new database. We will use the same database used by the Visual Basic sample program BIBLIO.MDB, which is a database of authors, book titles and ISBN numbers, with some additional information added to link to the voice files describing the books. This example is found in your VBV directory.

Specifications

This system will provide a voice interface to allow callers to search for books by ISBN number, and by author name. When searching for an ISBN, we will search the ISBN matching the number supplied. When searching by author, we must translate the supplied digits into letters before searching.

Databases

First step is to plan the databases.

We will need three databases, one which allows us to search for an author from a list of authors, another that can access the book names by a selected author, and one which indexes into a list of ISBN codes, and can retrieve the name of the corresponding book.

Voice Menus

Next step is to plan the voice menus.

The main menu should say:

"Good morning (afternoon, evening). Press 1 to search by ISBN or 2 to list books by author or 8 to exit"

The voice system should then wait for a digit.

If it receives a 1, it must request an ISBN.

"Please enter the ISBN code, and press # when finished".

We should use # to allow the caller to terminate digit collection immediately.

Then it should search the ISBN list for ISBN's that match the entered number. If it finds one, it should say the full ISBN of the book.

"The ISBN of this book is <number>. The author is <author>. Press 1 if this is correct or 2 to search by another ISBN".

If the caller enters 1, the system says:

"The title is <title>."

Entering 2, returns the user to the ISBN prompt.

If the caller enters 2 at the main menu, the system says:

"Please enter the name of the author, and press # when finished"

The system should then search for authors starting with the digits specified.

"Author <author name>. If this is correct, press 1, otherwise press 2. Press 8 to exit"

If the caller presses 1, the system should allow the caller to query which books have been written by this author.

Creating the databases

Next we need to create the databases. In this case, we will be using the databases provided in the VBVoice directory called biblio.mdb.

Creating voice phrases

We will need to create voice phrases containing the author's names and book titles, and find a way of finding the right phrase to play. There are several ways of doing this:

1. Create a new VOX file for each book title and author, and then add a new field to each database table containing the filename for each author/book title. The VBVoice controls can then extract the filenames out of the database using a DataGet control and play them using the System Phrase *FileSpec<>.*
2. An easier but less efficient way to do this is to create two phrase files, one for

author names and one for book titles, and set the name of each phrase to be equal to the book title or author. We can then use the database contents (book title, author name) directly to find the phrase in the phrase file using the System Phrase *Phrase by Name<>.* One disadvantage of this method is that if the database is updated, for instance the spelling of an author's name is corrected, the phrase name in the voice file will become invalid and will have to be updated. This method may also be slightly less efficient since VBVoice has to read the file index and search for the script in the phrase file for each phrase - unless the file is pre-opened at startup. (See Setup Menu - Startup Files page I-22).

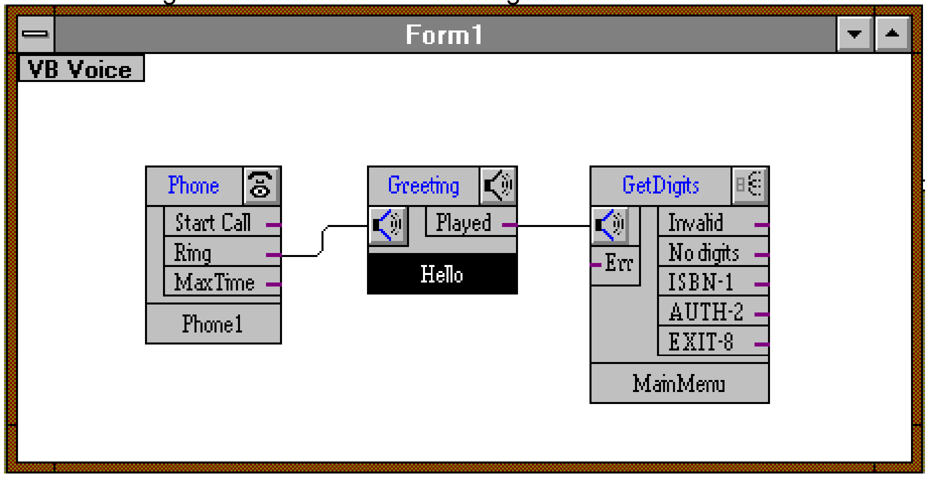
3. The best way is to use the AU_ID field to identify the author. Each author can have a voice file with their name recorded, in the form AUTHn.VOX, where n is the AU_ID.

Adding the controls

1. Create a new project and call it BOOKS. Add the VBVOICE.VBX file (in your Windows System directory), and add a new form. To have VBVOICE.VBX added to your default new project, open AUTOLOAD.MAK in your VB directory and add VBVOICE.VBX to it.
2. Add a Phone control on the left hand side. Every system starts with one or more Phone controls. You may add as many phone controls as lines you have operating (and for which you are licensed).
3. Click on the setup button (upper right corner of phone control) to view the dialog. The VoiceLine text box should be set to 1 by default. The Allocate Line and Answer Call options should be checked since we want this system to answer the phone automatically.
4. Add a Greeting control to play the initial greeting. Greeting controls are used if you want to play a prompt without getting any digits from the caller. Set the name of this control to Hello by double-clicking on the name and typing in the new name.
5. To add a greeting that automatically says, "Good Morning, Good Afternoon, or Good Evening," click on the greeting button (left side of Greeting control) to get the Create Greeting dialog. Then click the Add System Phrase button. In the System Phrase dialog, scroll down the Phase Types list to find the phrase "Initial Greeting" and then select it. Press OK to return to the greeting dialog, and press OK to save your selection. This will be the only phrase we need in this greeting.
6. Now, add a GetDigits control to play the main menu. Callers will re-enter here when they complete a search. Name this control MainMenu by double-clicking on the name and entering the new name. At this point, choose Save Project from the File menu to save your work.
7. In the GetDigits dialog (MainMenu), we need to add the greeting:
   "Press 1 to search by ISBN, 2 to list books by author or 8 to exit."
8. First we need to create a file to hold our phrases. We must load Announce to initially create our phrase file. Minimize the Visual Basic window and load Announce by clicking the Announce Icon in the Windows Program Manager.
9. Choose New from the File menu. From the New dialog, choose New List to create a new phrase file.
10. From the File menu, choose Save As... and save the current file as BIBLIO.VAP in the VBV directory. From the File menu, choose New again. Now, you should be in the New dialog with the New Phrase highlighted. Type in the entire script above (at step #7) and click New Phrase. From the File menu, choose Save to save your work.
11. Minimize Announce and return to the Visual Basic window. Click the Setup

button (upper right corner) on the MainMenu control. We will now enter our 3 exit conditions using 1,2 and 8. The GetDigits control comes with several conditions predefined. We are not using the first option, digit '0', so we can remove that by selecting it and pressing Delete. The next two conditions are valid, but let's change the names to something more descriptive. Double-click on condition '1' to show the Digit Match Condition setup dialog, type in a new Condition Name "ISBN - 1", and press OK. For the condition '2', use the name "AUTH - 2". Change the next condition '3' to the name "Exit - 8" and the DigitMask to "8". The remaining "#" condition can be deleted. Choose OK to exit the GetDigits setup dialog.

12. We now have an initial greeting and a main menu. We need to connect these controls together to make the system do something. Click in the Ring output of the Phone1 control and drag to the speaker icon in the Hello control (the cursor should change to a square when it is in position), and release. You should see a line connecting the 2 points. Repeat this for the Played output of the Hello control and the speaker icon of the MainMenu control.
13. Your design should now look something like this:



1. To gain some confidence before continuing the design, let's record the greeting and test the system so far. Click the greeting dialog from the MainMenu control. In the Create Greeting dialog, click Add VAP Phrase. From the Add VAP Phrase dialog, click the selector arrow under VAP File and select the phrase file that was created earlier, BIBLIO.VAP. From the Add VAP Phrase dialog, click Edit Phrase. Announce! should now start up in the phrase file BIBLIO.VAP. Close the initial box (labeled Press 1...) that appears so that only the box labeled biblio.vap is present, and then press the Record button to get to the Record dialog.
2. Announce! will show you the script you must record. Press Record to begin your recording.
3. When you are finished, choose OK and click the Play button to hear your prompt. If you don't like it, record it again.
4. When you are finished, close Announce! You will be prompted to save your file and you will then return to Visual Basic. Choose OK from the Add VAP Phrase dialog and then OK from the Create Greeting dialog. You are now ready to test the system, such as it is.
5. Choose Test from the VBVoice Status menu to get the Test dialog. If the Phone1 control is not highlighted, click on it to make it the active control. (This tells it where to start the test.)
6. Choose Start. The Ring or Call dialog will appear. Choose Ring. If all goes

well, you should now hear the greeting "Good morning" (or afternoon / evening depending on what time it is). Next, the system should play your newly recorded prompt. Press the digit 1 - either on the Test dialog or on the phone depending upon your setup. The system will attempt to exit via the output "ISBN -1". At this point, the system will stop since we haven't connected that output to anything yet. If necessary, close the Test Log dialog so you can see your controls. The ISBN output should be highlighted.

7. Re-test the system again, but this time let's skip the Phone1 and Hello controls (we know they work) by making MainMenu the active control. This time, press 2 to test the "AUTH -2" output, and then re-test again to check the error handling by pressing an incorrect digit, and then by pressing no digits at all. (Remember, close the Test   Log dialog to see your controls after each test is completed).
8. Save the project up to this point.

Checking for errors

Use the Check Control, Check Form and Check System to check your system. Even if you have only one form in your system, Check System performs some extra checks over Check Form. If you get errors, double-click on the error message to highlight the control in error. To get help on the error, select the error line in the list box and press the F1 key (or select help from the test log menu).

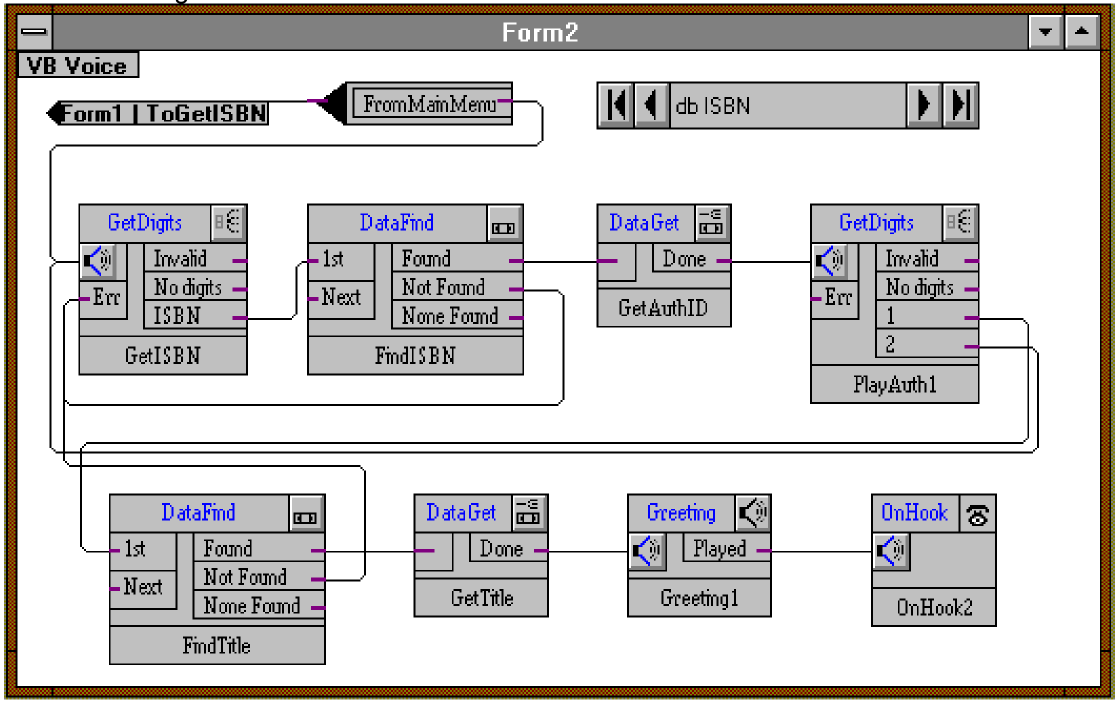
Adding controls for ISBN access

1. Now we will add the subsystem that gets the ISBN number and finds the book of that number. Start by adding another form (form2).
2. Add another GetDigits control, call it GetISBN. Add a new phrase to the greeting with a script of ‘Please enter the ISBN. Press # when finished’. To do this, choose the greeting button and from the Create Greeting dialog, press Add VAP Phrase. Choose BIBLIO.VAP file from the pull-down list and press EditPhrase. Announce! will again be loaded with the first phrase you recorded. Close this window and choose New from the File menu. Type the phrase above into the New Phrase text box and press New Phrase. Using the same procedure as the first phrase, record this phrase and return to Visual Basic. From Add VAP Phrase dialog, press the down-arrow selector under VAP File to re-load and update the BIBLIO.VAP file. Both phrases will now be listed in the selection box. Double-click on the new phrase to return to the Create Greeting dialog and press OK.
3. Set up the GetISBN control to have one condition only,which is a wild character $. This will allow the control to collect all the digits. Set the ‘Terminate on ..’ digit to # to allow the caller to terminate digit collection without waiting for a timeout. Press OK.
4. Now add a VB data control and call it dbISBN. To change the name of standard controls, you must use the Properties window and select the Name property. Set the Caption to the same name also. Then set the DatabaseName property to the C:\VBV\BIBLIO.MDB file and set the RecordSource to ‘Titles’. The Titles table contains a list of book titles and their ISBN codes, and the Author Id number for each book. Since the data control is not graphically connected, it can be tucked away in a corner, out of the way.
   Note: To view the contents of the database, and view names and ID numbers, use the DataMgr program provided with Visual Basic and open c:\vbv\biblio.mdb
5. Add a DataFind control (call it FindISBN) to reference the dbISBN data control. Use the setup dialog to set the *Search in data control* property to dbISBN. To do this, check the ‘Records containing:’ button, and set to %GetISBN%. Click the Find Control button at the bottom of the dialog to get

a list of available controls. You can cut and paste into the text box. Set the 'Search in field' to ISBN.

6. Set the Match Fields to 'Exactly', and the Match On to 'All Characters'. This will make the DataFind control search for the entry that matches with the digits supplied.
7. This control will read the full ISBN number from the database. We then have to say the full ISBN number back to the caller for verification.
8. Add a DataGet control and call it GetAuthID. This control will read the author's id number from the database. We can use this author id to select the file to play for the author's name. Set the 'Get data from field' to AU_ID. We do not need any data matching conditions since we are only retrieving data and not testing it, so we can delete all of the Test Conditions.
9. Now add a GetDigits control and call it PlayAuth1. This will say the ISBN, Authors ID number and name, and ask the caller if this is correct to continue the search for a title. We need to add 3 phrases to the greeting. Two will be VAP phrases in BIBLIO.VAP that you must record; example: "The author is", plus "Press one to select this Author, or two to search by another ISBN". The other is a System Phrase. that will say the author ID number. Click Add System Phrase from the Create Greeting dialog. In the Phrase Types scroll box, scroll down to Number<>. In the Number to say box, type %GetAuthID%. Click OK to save the information in the System Phrase dialog.
10. If you wish to add the author's name as well, you will have to record an individual file for each author containing the spoken name of the author. Each file should be named AUTHn.VOX (ie. c:\vbv\auth21.vox), where n is the authors ID number as assigned in the database. To play these files, use another System Phrase, of type FileSpec<>, with FilePath set to AUTH%GetAuth%.VOX.
11. When the phrases are complete, click OK again to exit the Create Greeting dialog.
12. Now we need to set the GetDigits control to look for the digits 1 and 2, in the same way as before. The '2' output should connect back to the GetISBN input to start a new search by ISBN. Reminder: To view the contents of the database, and view names and ID numbers, use the DataMgr program provided with Visual Basic and open c:\vbv\biblio.mdb.
13. To complete this section of the system, connect the 'Done' output of the 'GetAuthID' control to the input of PlayAuth1. Depending on how you have laid out the controls, you may want to use a named connection for clarity rather than a line connection.
14. In order to continue, and list the book titles, you will need another DataFind control. Call this one FindTitle. Set the 'Search in data control' to dbISBN, the 'Records containing:' to %GetISBN%, and the 'in field' to ISBN. Match fields 'exactly' on 'all characters'. You can now connect the 1 output of PlayAuth to the input of FindTitle.
15. Add another DataGet and call it GetTitle. Set the 'Get data from field' to Title. Compare data exactly, and match on all characters. Delete all test conditions.
16. You will now need to record your Titles. This is done the same way you created your Biblio.vap file list. (1) open announce, (2) click new list, (3) click new phrase, (4) type in the title of the book exactly as it appears in the datamanager (matching case and spaces - even a trailing space could cause the application to fail to recognize the title by exact match), (4) click record, and record your voice phrase, (5) repeat steps 3 and 4 until all titles are recorded, (6) save as c:\vbv\Titles.vap
17. Add a greeting control, and click the left greeting icon on the control. Chose Add system phrase. Select 'VAP Phrase by Name', enter 'VAP File' Titles.vap, 'Phrase script' %GetTitle%.

18. Finish this form by adding an OnHook control and connecting each control in the same sequence they were presented to you. You should also connect the Not found output of FindTitle to the Err output of GetISBN.
19. You will need to connect MainMenu to GetISBN. Add an OutConnector (will show up as OutConn1) to form 1, and connect it to ISBN-1. Change the name to something meaningful. Add an InConn to Form2 and input to GetISBN. Again change the name to something meaningful and unique from all other controls. Click and drag from the OutConn to white space to set the specifics.
Your design should now look like this:



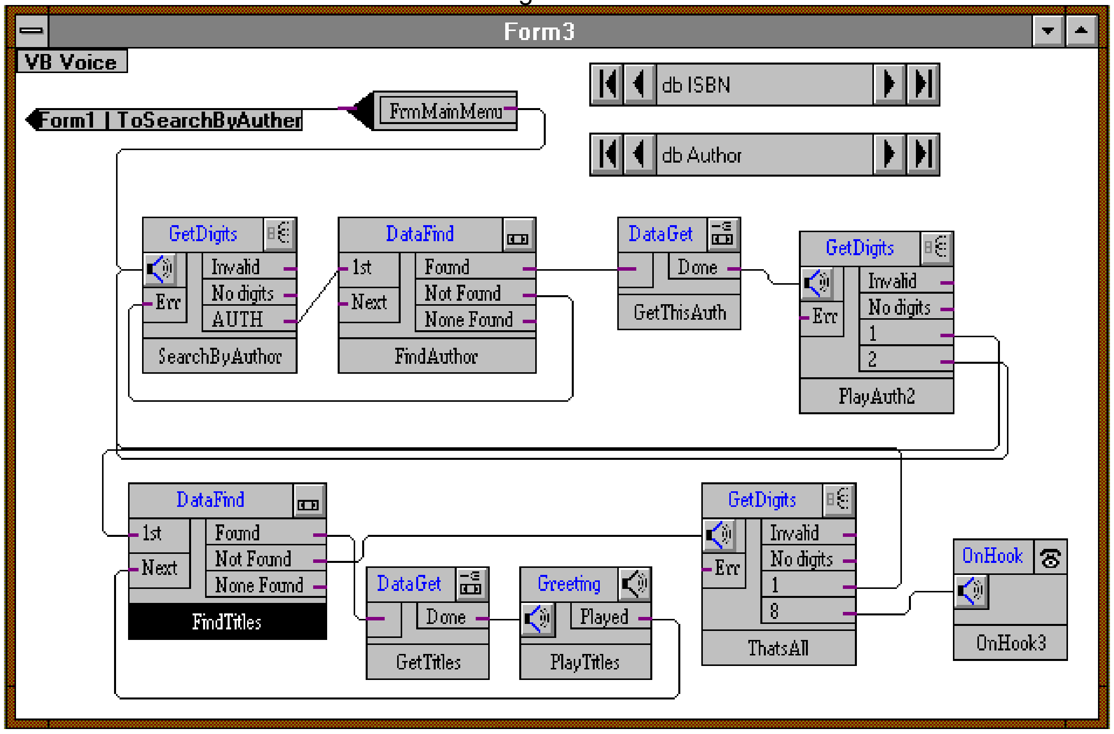
1. To make the databases available to VBVoice, Visual Basic must be started using the Run / Start command. You can now start Test mode once again, and test your database access. You should be able to enter a valid ISBN number and have the system play the author's ID, Name, and book title.
2. Another situation you may want to handle is when there are no records found at all, and play a message, “there were no records found”. To do this, connect the ‘None Found’ output to a greeting control to play the message. After playing the message, take the caller back to the main menu.

Adding the search by Author-name

1. Add a New Form (form3). This part of the design is similar to the ISBN search, except that: the GetDigits control (call it SearchByAuthor) must say: “Please enter the first few letters of the Author's last name, using the letters on the keypad. Use the digit 1 for Q and Z.. Press # when finished"
2. A new data control (call it dbAuthors) must have the RecordSource set to Authors. Remember: this is set in the properties window.
3. A DataFind control (call it FindAuthor) must search in the ‘Author’ field, for records containing %SearchByAuthor% and reference your dbAuthors data control. The Use Digit Translation box should be set, so that the digits are translated into the keyboard equivalents before matching (i.e. 2 is A, B, or C, 3 is D, E or F etc.).
4. The DataGet control (let's call it GetThisAuth) should access the AU_ID field,

so the system can retrieve the author's name file.

5. The GetDigits (call it PlayAuth2) control should reference the GetThisAuth control to select which Author file and number to play.
6. Add another Datafind control (call it FindTitles), this one referencing the data control dbISBN (copy it over from form2), set ‘Records containing’ to %GetThisAuth%, and ‘In Field’ to AU_ID. This will retrieve the authors ID number to locate your earlier recorded AUTHn.VOX files.
7. Connect the Found port of FindTitles to A DataGet (call it GetTitles), have it “Get data from field’ Titles. Again, delete all test conditions. Connect your GetTitles to a new Greeting control (call it PlayTitles), and ‘Add System Phrase’, select ‘VAP Phrase by Name<>’, in ‘Vap file’ Titles.vap, for ‘Phrase Script’ %GetTitles%. Connect the ‘Played’ port back to the ‘Next’ port of FindTitles. This will allow the search to continue listing books by that author.
8. Now connect the ‘Not Found’ output of FindTitles to a new GetDigits (call it ThatsAll). Record a phrase to say “That is all the titles by the selected author. Chose 1 to select another Author or 8 to exit”. Connect the 1 port back to SearchByAuthor. Connect the 8 port to a new OnHook control.
9. Note: Both data controls are required on this form (dbISBN and dbAuthors). Form3 should look like something like this:

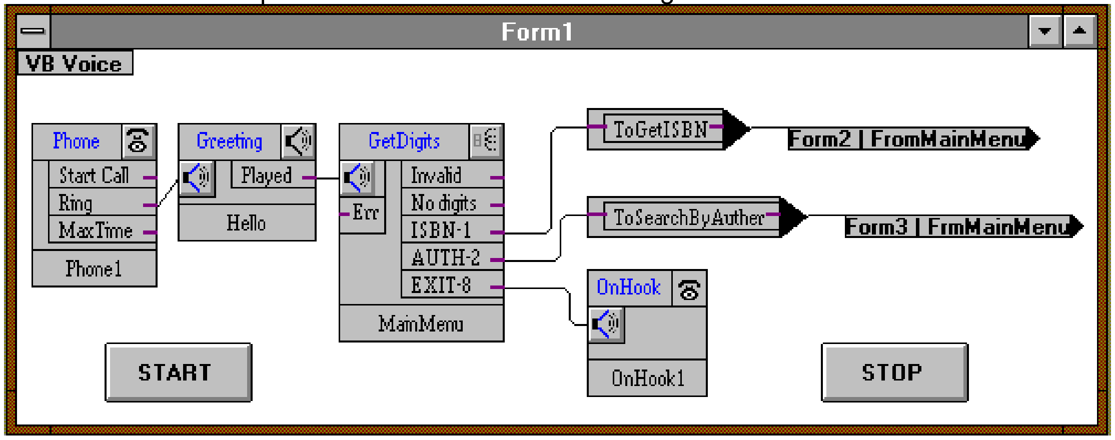


To Run The Program.

1. Double click on the white space in Form1 and Type:
   form1.show
   form2.show
   form3.show
2. Close the window and save.
3. Add two Visual Basic button controls to form 1. Change the Caption and Name (in properties window) of one to START and the other to STOP.
4. Double click the START button and type in the code window for the Click event:
   Dim i As Integer i = vbv_start_system()
5. Double click on the STOP button and type:

Dim i As Integer i = vbv_stop_system

6. Select Add file... from the file menu and select VBVOICE.BAS from your VBV directory. This adds the declarations for the VBVoice DLL functions.
7. Save your project, then chose Run, Test, Start.
   Your complete Form1 should look something like this



Placing an order.

We have now designed two systems to allow a caller to choose a book, either by ISBN number or by Author selection. We now want to add a section which will allow the caller to order the book. We will assume the caller has an account number for billing purposes.

All we have to do is ask for the number, validate it, and add the entry into a database.

1. First we will need 2 new database tables, Accounts and Orders. The Accounts table contain a list of valid account numbers. In a real system, this table would also contain billing information. The Orders table will contain 2 fields, Account and ISBN. Each record corresponds to an order taken.
2. Add 2 data controls, one to reference each table in the database.
3. Use a GetDigits control to request the number. Since we know that account numbers are all 7 digits, use the digit mask nnnnnnn.
4. Use a DataFind control to search in the account database to verify the account number.
5. Use a DataNew control to add a new record to the Orders database
6. Use a DataChg control to add the data to the new record. A DataChg control contains a list of database fields to update, and the new data for each field. We will want to set the Account field to the account number received from the caller, and the ISBN field to the ISBN of the book selected by the caller. (See note below)

Note that there are two ways that we could have obtained the ISBN - either from the ISBN selection path or the path that does look-up by Author. We would like to use common code to take the order, regardless of how the book was selected. However the ISBN will reside in a different location depending on how the book selection was made. One way to do this is to pick just one of the GetDigits controls to reference the ISBN number. If the book is selected via another path which does not use this GetDigits, then it's value can be set by code to the ISBN value, using the Digits property