Exercise 3-1: Build a DataGridView application

In this exercise, you'll build the application shown in figure 3-9 in your textbook. That will show you how to build a simple application with data sources, a dataset, and a **DataGridView** control.

Build the form and test it with valid data

1. Start a new application named **TermsMaintenance** in your chapter 3 directory, and use the techniques in figures 3-1 through 3-8 to create the data source and drag it onto the form. Then, adjust the size of the form and the **DataGridView** control as needed, but don't change anything else.
2. Test the application with valid data in three phases. First, sort the rows by clicking on a column header, and size one of the columns by dragging its column separator. Second, change the data in one column of a row, and move to another row to see that the data is changed in the dataset. Third, add a new row with valid data in all columns, and move to another row to see that the row has been added. At this point, the changes have been made to the dataset only, not the database. Now, click the Save button in the toolbar to save the changes to the database.

Test the form with invalid data and provide exception handling

3. Test the application with invalid data by deleting the data in the **Description** column of a row and moving to another row. This should cause a `NoNullAllowedException` that's automatically handled by the **DataGridView** control so the application doesn't crash.
4. Add an event handler for the `DataError` event of the `DataGridView` control as shown in figure 3-15. To start the code for that handler, click on the control, click on the **Events** button in the Properties window, and double-click on the DataError event. Then, write the code for the event, and redo the testing of step 3 to see how your code works.
5. When you're through experimenting, end the application and close the project

Exercise 3-2: Build an application with text boxes

In this exercise, you'll build the application shown in figure 3-12 in your textbook. That will show you how to use data sources with controls like text boxes.

Build the form and test it with valid data

1. Start a new application named **VendorMaintenance** in your chapter 3 directory, and create a data source for the fields in the **Vendor** table that are used by the form in figure 3-12. Then, use the techniques in figures 3-10 and 3-11 to drag the data source fields onto the form as text boxes. At this point, the form should look like the one in figure 3-11.
2. Test the application with valid data in three phases. First, use the toolbar to navigate through the rows. Second, change the data in one column of a row, move to another row, and return to the first row to see that the data has been changed in the dataset. Third, add a new row with valid data in all columns, move to another row, and return to the added row to see that the row has been added to the dataset. Now, click the Save button to save the dataset to the database.

Test the form with invalid data and provide exception handling

3. Add a new row to the dataset, but don't enter anything into the **City** field. Then, click on the **Save** button. This should cause a `NoNullAllowedException`, since **City** is a required field.
4. Add exception handling code for an ADO.NET `DataException` as shown in figure 314 to catch this type of error. This code replaces the default code in the `VendorsBindingNavigatorSaveItem_Click` event handler. But note that you also need to add a using statement for `System.Data.SqlClient` to the form. Then, run the application and redo the testing of step 3 to see how this error is handled now.
5. Delete the data in the **Name** column of a row, which means that the column contains an empty string. Next, move to another row, and return to the first row to see that the row has been accepted into the dataset. Then, click on the **Save** button and discover that this doesn't throw an exception because an empty string isn't the same as a null value. This indicates that data validation is required because an empty string isn't an acceptable value in the database. In the next chapter, you'll learn one way to provide data validation.
6. Adjust the controls on the form and any related properties so the form looks like the one in figure 3-12. This should take just a minute or two.

Use the Dataset Designer

7. Use one of the techniques in figure 3-16 to view the schema for the dataset in the Dataset Designer.

8. Click on the table adapter in the Dataset Designer and review its properties in the Properties window. Then, look at the `Select` statement that's used for getting the data into the dataset. To do that, click on the arrow in front of `SelectCommand`, and click on the ellipsis button for `CommandText`. This opens up the Query Builder, which you'll learn about in chapter 5, and there you can see the Select statement that's used for getting the data into the dataset. Then, close the Query Builder.
9. Right-click on the query in the Dataset Designer, and preview the data that will be retrieved by that query as shown in figure 3-17.
10. When you're through experimenting, close the project