DataGridView provides a visual interface to data. It is an excellent way to display and allow editing for your data. It is accessed with VB.NET code. Data edited in the DataGridView can then be persisted in the database.

Example. First, you should add a DataGridView control to your Windows Forms application by double-clicking on the control name in the Visual Studio designer panel. After you add the control, you can add the Load event on the form.

Load:
You can create the Load event on the Form's event pane in Visual Studio. We use Load in this example.

Here:
We use an empty DataTable on the DataGridView control. We assign the DataSource property.

VB.NET program that uses DataGridView

```
Public Class Form1

    Private Sub Form1_Load(ByVal sender As System.Object, _
                           ByVal e As System.EventArgs) Handles MyBase.Load
        '
        ' Fill in the data grid on form load.
        '
        DataGridView1.DataSource = GetDataTable()
    End Sub

    Private Function GetDataTable() As DataTable
        '
        ' This Function needs to build the data table.
        '
        Return New DataTable()
    End Function
End Class
```

This event handler is executed when the program starts up and when the DataGridView control is displayed. The Form1_Load sub calls into the GetDataTable function, which would return a DataTable from your database in SQL Server.

Assigning the DataSource property on DataGridView copies no data. It allows the DataGridView to read in the DataTable and display all its contents on the screen in grid form. This is an efficient way to populate DataGridView.

DataTable

Lightning bolt. When using the DataGridView control in Windows Forms, you should use the lightning bolt panel. This allows you to manipulate the events on the control. DataGridView has many events, and this article doesn't describe them all.

> **However:**
> You will often want the CellClick, SelectionChanged, CellDoubleClick, and KeyPress events, depending on your requirements.

Objects. You can use an object collection, such as a List(Of String), in your DataGridView using the Visual Basic language. The object collection will be read. Its properties (get accessors) will be used to display the values on the screen.

List

> **Tip:**
> This is the easiest way to get started with DataGridView. But it may be less effective than more complex approaches.

Class used in program: VB.NET

```
''' <summary>
''' This class contains two properties.
''' </summary>
Public Class Test

    Public Sub New(ByVal name As String, ByVal cost As String)
        _name = name
        _cost = cost
    End Sub

    Private _name As String
    Public Property Name() As String
        Get
            Return _name
        End Get
        Set(ByVal value As String)
```

```
            _name = value
        End Set
    End Property

    Private _cost As String
    Public Property Cost() As String
        Get
            Return _cost
        End Get
        Set(ByVal value As String)
            _cost = value
        End Set
    End Property

End Class
```

VB.NET program that uses DataGridView with class

```
Public Class Form1

    Private Sub Form1_Load(ByVal sender As System.Object, _
                           ByVal e As System.EventArgs) Handles MyBase.Load
        '
        ' Fill in the data grid with a List
        '
        Dim list = New List(Of Test)
        list.Add(New Test("Mac", 2200))
        list.Add(New Test("PC", 1100))
        DataGridView1.DataSource = list
    End Sub
End Class
```

The program includes the Public Class Test definition, which encapsulates two properties with backing stores. The names of these properties are Name and Cost. These could be used for an inventory of merchandise.

Note:
It is important to declare the members as properties so the metadata can be used by the DataGridView.

After the Dim List is allocated, two new Test objects are added to its contents. These two objects are reflected in the DataGridView output to the screen. You can see the four cells from the four values in the example in the screenshot.

Hide row headers. You can hide the row headers, which are the boxes on the left of the DataGridView control, from appearing on the screen. The screenshot shows what the row headers look like on DataGridView controls.

Tip:
Go to the Visual Studio designer and change the value of

RowHeadersVisible to false. This hides row headers.

Improve tabbing. It is possible to improve the behavior for tabbing into and out of the DataGridView control in your Windows Forms program. You can change the StandardTab property on the control in the Visual Studio designer.

So:

When StandardTab is enabled, the focus will move out of the DataGridView control when the user presses tab.

Add rows. We can manually add rows to the DataGridView control. The Rows.Add function will return the index of the newly-added row. After calling Rows.Add, you can use the Item accessor and the column index to assign the Value properly.

Here:

The example adds a row with two strings to the DataGridView on load. These are displayed to the screen.

VB.NET program that adds rows

```
Public Class Form1
    Private Sub Form1_Load(ByVal sender As System.Object, _
                           ByVal e As System.EventArgs) Handles MyBase.Load
        ' Add row using the Add subroutine.
        Dim n As Integer = DataGridView1.Rows.Add()
        DataGridView1.Rows.Item(n).Cells(0).Value = "First"
        DataGridView1.Rows.Item(n).Cells(1).Value = "Second"
    End Sub
End Class
```

Adding columns in designer first. The example throws an exception if you do not have at least two columns in the DataGridView. To add columns to the control, you can use the Columns collection in the designer and use the Add button.

Then:

After you add the columns, compile and run your program and no exception will be thrown.

Edit Columns

Get current cell. We can obtain the location of the current cell in VB.NET code. One way you can do this is add the SelectionChanged event. As a reminder, you can add events easily in

Visual Studio by using the lightning bolt panel.

Next, in the DataGridView1_SelectionChanged event, you can access the DataGridView1.CurrentCellAddress property. This is a System.Drawing.Point type that has two instance properties, X and Y.

Tip:
You can use the Point itself or just access its properties. Its properties are of type Integer.

VB.NET program that gets current cell

```
Public Class Form1

    Private Sub Form1_Load(ByVal sender As System.Object, _
                           ByVal e As System.EventArgs) Handles MyBase.Load
        ' Uses Test class from above.
        Dim list = New List(Of Test)
        list.Add(New Test("Mac", 2200))
        list.Add(New Test("PC", 1100))
        DataGridView1.DataSource = list
    End Sub

    Private Sub DataGridView1_SelectionChanged(ByVal sender As System.Object, _
                                               ByVal e As System.EventArgs) _
                                               Handles DataGridView1.SelectionChanged
        ' Get the current cell location.
        Dim y As Integer = DataGridView1.CurrentCellAddress.Y
        Dim x As Integer = DataGridView1.CurrentCellAddress.X

        ' Write coordinates to console.
        Console.WriteLine(y.ToString + " " + x.ToString)
    End Sub
End Class
```

The example shows the SelectionChanged event handler. SelectionChanged signals when the user changes the current cell or clicks on any cell. It is ideal to use for accessing the cell address and changing the UI based on some criteria.

Expand cells. It is possible to make the cells in a DataGridView expand horizontally. You can set the AutoSizeColumnsMode to fill. Also, you can weight certain columns to have larger widths than other columns using the Edit Columns dialog box.

Double-clicking. You can handle the events that occur when the user double-clicks on a cell. You can open the Visual Studio designer and add the CellDoubleClick event in the pane with the lightning bolt.

Note:

When the CellDoubleClick event occurs and the header was clicked on, the RowIndex of the DataGridViewCellEventArgs parameter is -1.

VB.NET program that uses double-clicking

```
Public Class Form1
    Private Sub Form1_Load(ByVal sender As System.Object, _
                           ByVal e As System.EventArgs) Handles MyBase.Load
        ' Load here.
    End Sub

    Private Sub DataGridView1_CellDoubleClick(ByVal sender As System.Object, _
        ByVal e As System.Windows.Forms.DataGridViewCellEventArgs) _
        Handles DataGridView1.CellDoubleClick
        If e.RowIndex = -1 Then
            Return
        End If
        ' Double-click logic.
    End Sub
End Class
```

Summary. We looked at the DataGridView control in the VB.NET language targeting the .NET Framework. This control is ideal for rendering data to the screen in programs that use collections of objects or databases and DataTable instances.

And:

We used the DataSource property, looked at events, tabbing, row headers, and cells in the DataGridView control.