Data Types, Variables

VB data types can be categorized into five groups by the type of data managed.

Boolean Data (Boolean)
This single data type provides a single bit of data, either True or False.

Character Data (Char, String)
Visual Basic includes data types that manage either single characters or long strings of characters.

Date and Time Data (Date)
A single data type manages both date and time values.

Floating Point Data (Decimal, Double, Single)
The various floating point data types each manage a subset of rational numbers. Some of these data types provide more mathematical accuracy than others.

Integer Data (Byte, Integer, Long, Short)
The integer data types store integer values between a data type-defined minimum and maximum value. Some of these data types support negative numbers.

A variable is declared with the statement: `Dim variablename as datatype`. To declare a variable x as an integer, the statement is: `dim x as integer`.

When first created, the variables that are strings are set to zero-length strings and the rest to value zero.

Important concepts that relate to variables: Variables defined in VB have a scope. Essentially all variables are defined in the memory and will be lost once the application is complete. Depending on where and how you define the variable, it exists only for a period of time. The scope of a variable identifies where the variable can be used. The scope of a variable depends on where the variable is defined. A variable can be defined in three locations:

1. event procedure of an object,
2. declarations section of a form and
3. declarations section of a code module.

A variable defined in the event procedure of an object is called a local variable and is available only for that event. Once the event is executed the local variable is removed from the memory. A variable defined in the declarations section of a form or a module is called a form-level variable and is available for all events and procedures in the form. A form level variable remains in the memory until the application ends. A variable defined in the declarations section of a code module is called a global variable and is available for all the forms defined in the application. A global variable remains in memory until the application ends.

How to declare variables
1. A local variable is defined with the Dim statement.
2. A form-level variable is defined with the Dim or Private statement.
3. A global variable is defined with the Public statement.
Using Multiple Forms
Create a new application. Design an interface as follows.

Now we will add another form to the application. From the menu choose Project, Add Windows Form to add a new form. Keep the default name of Form2 and design its interface as follows. Write the appropriate code behind the Exit command button to close both forms and end the program.

When an application has more than one form, and data may need to flow between forms, you will need to create a module. Global variables are created in the module. To add a module, right-click on the project in the solution explorer window, select the Add option and add a module. You can also add the module by choosing Project, Add Class from the top menu.
We define global variables in the code module’s declaration section. A code module is used to write code independent of control events in the form. A global variable is defined using the word Public (instead of Dim). To define \( y \) as a global variable of “String” type, we use the statement \( \text{Public } y \text{ as string} \).

To display and hide forms we use the following code.

\[
\text{Formname.show} \quad \text{Displays a form on the screen. If a form is not loaded it automatically loads it.}
\]

\[
\text{Formname.hide} \quad \text{Hides the form from view but remains in memory}
\]

Note that form 1 opens when you start the application. When the user clicks on the Click Here When Done button in form1, we want to hide form1 and show form2. The following code does this.

```vbnet
Private Sub Button1_Click(sender As Object, e As EventArgs) Handles Button1.Click
    Me.Hide()
    Form2.Show()
End Sub
```

### Radio Buttons, Check Boxes, and Group Boxes

We will now explore radio buttons and check boxes in VB. The radio buttons and check boxes provide a user-friendly environment for a user. Create two radio buttons, name them radicecream and radyogurt, create appropriate text and place them on the form. Run the application and you will notice that only one radio button can be selected at a time. Add a button called calculate price and place it in the form. Add a label with the text “The total price is” and a label with the name lblprice. The interface should be as follows.

```
Now let us create the code that will calculate the price based on $1.00 for ice cream and $.80 for yogurt. Note that the checked property of the radio button will indicate whether the radio button is selected. The function formatcurrency can be used to display the price is currency format.
Make sure the program works correctly. Now let us add three more radio buttons (name them rad1, rad2 and rad3) with the labels: one topping, two toppings and three toppings. The additional costs with the toppings are $0, $0.1 and $0.15, respectively. Notice that we want to treat these radio buttons as a separate group from the initial ones i.e. a user should be able to select one from ice cream/yogurt and one from one/two/three toppings. However, the way we set up the form currently, does not permit this. We can use a GroupBox control in this instance. A GroupBox acts as a container, which contains a number of radio buttons. By default, the form is defined as the container from which only one radio button can be chosen. GroupBox can be used to define groups, with one button from each group selected simultaneously. You will need to cut the three radio buttons and paste them onto the GroupBox selected, or draw new radio buttons within the GroupBox. Create the appropriate code to display the right price.

We would also like to provide options for users to add whipped cream and/or nuts. Whipped cream costs an extra $0.15 and the nuts $0.20. We could use check boxes in this case (name them chkcream and chknuts).

The interface should look as follows:

Now write code to hide the total price when a user makes a different selection.

*The Select Case Statement*

The select case statement is used for complicated if statements. The following is the syntax for the select case statement

```vbnet
Select Case testexpression
    Case expressionlist1
        instructions
    Case expressionlist2
        instructions
    Case Else
        instructions
End Select
```

To specify a range in the expressionlist, you can use To and Is. For example, Is > 10 or 1 to 10.