This handout very briefly describes how to create a Windows application using Visual Basic 2005 (VB). In KC-267, VB is part of Visual Studio 2005. A hobbyist version, Visual Basic 2005 Express, also is available from Microsoft (see the link on the course webpage).

To use VB to create an application, do the following steps:


2. Every application must be part of a project. To create a project, use File -> New -> Project. Choose VB as the language, if it is not already the default language, by double-clicking on Other Languages and selecting VB. Then select the Windows Application template. You can give the project a name if you wish, or leave it as WindowsApplication1. (The name of the resulting application will be the same as the project name.) Click on OK.

3. The instructor prefers to have the ToolBox and the Properties windows pinned open. This can be done using View -> ToolBox to open the ToolBox window, and View -> Other Windows -> Properties Window to open the Properties window.

VB programs consist of two parts: a form design that determines what the application will look like when it is run, and a set of handler procedures written in Visual Basic programming language code that are executed when actions like mouse clicks or key presses are performed. These are explained below.

To run a VB program from the VB IDE, use Debug -> Start without Debugging. To stop a VB program, just click on the Windows close window button. Note that the IDE will not recompile a program when there is a running instance of the previous version.

Form Design

The form design is presented as the GUI will appear. There is actual code behind the form, but very rarely does it need to be seen by programmers. All VB programs start with a main form named Form1.

The form design is created by dragging and dropping items from the ToolBox onto Form1. Each item has a set of properties that may be set in the Properties Window for that item. For example, almost every item has Color, Font, and Background properties. The Properties Window changes to the appropriate property list each time a different item is selected in the form design. (Note: changing the Text property of Form1 will change the text of the titlebar of the resulting application.)

Here is a list of the most common, simpler-to-use GUI items and their most commonly set or used properties:

- **Label** - text usually used as a prompt for an input item or to print out results. Properties include Text (displayed as the label).
- **TextBox** - allow users to input from keyboard. Properties include Text (can be set to default value or instructions), Multiline (to allow multiple lines of input), Scrollbars.
- **Button** - causes an action to be perform when clicked. Properties include Text (the button label).
• RadioButton - allows users to select one option from a group of choices when grouped together. By default, all radio buttons on a form are grouped together unless surrounded by a GroupBox. Radio buttons may be added or removed from a GroupBox by dragging them in or out of the GroupBox. RadioButton properties include Text (the button label), Checked (true if selected, false otherwise).

• CheckBox - allows users to select or clear the associated option. Properties include Text (the checkbox label), Checked.

• ListBox - a list of selectable entries. Properties include Items (the list of entries), SelectMode (whether more than one item can be selected)

• CheckedListBox - a list of items with checkboxes. Properties include Items.

• ComboBox - textbox with a drop-down list of possible entries. Properties include Items.

Visual Basic Programming Language Code

The handlers that are executed when an action like a mouse-click occurs are written in the Visual Basic programming language. Each GUI item has a default handler whose code shell is created when a double-click is performed on it in the form design. For completeness, Form1 is a class that contains all of these handlers, and the handlers are procedures. We will not discuss classes in this course, but we will talk about procedures in the next handout.

This section will follow the ordering of the class textbook and assumes it has been read for terminology. This handout just gives the specific VB syntax for the concepts presented in the textbook. Reserved words are shown in **bold**. Syntactic categories are represented using `<category>`. Actual code is shown in **Courier** font.

Variables and Data Types

Variables are declared with the following syntax:

```
Dim <identifier> As <data type>
```

Data types in VB include Integer, Double (for real numbers), Boolean, Char (for characters) and String.

Arrays

Homogeneous arrays are declared with the following syntax:

```
Dim <identifier> () As <data type>
```

The () is used to indicate this identifier is a one-dimensional array.

Constants and Literals

Constants are declared with the following syntax:

```
Const <identifier> As <data type> = <literal>
```

Assignment Statements

Assignment statements have the following syntax:

```
<identifier> = <expression>
```

VB defines the following operators:
<table>
<thead>
<tr>
<th>Operation</th>
<th>Operator</th>
</tr>
</thead>
<tbody>
<tr>
<td>Unary Negation</td>
<td>-</td>
</tr>
<tr>
<td>Addition</td>
<td>+</td>
</tr>
<tr>
<td>Subtraction</td>
<td>-</td>
</tr>
<tr>
<td>Multiplication</td>
<td>*</td>
</tr>
<tr>
<td>Division</td>
<td>/</td>
</tr>
<tr>
<td>Remainder</td>
<td>Mod</td>
</tr>
<tr>
<td>Equality</td>
<td>=</td>
</tr>
<tr>
<td>Inequality</td>
<td>&lt;&gt;</td>
</tr>
<tr>
<td>Less than</td>
<td>&lt;</td>
</tr>
<tr>
<td>Less than or equal</td>
<td>&lt;=</td>
</tr>
<tr>
<td>Greater than</td>
<td>&gt;</td>
</tr>
<tr>
<td>Greater than or equal</td>
<td>&gt;=</td>
</tr>
<tr>
<td>Logical negation</td>
<td>Not</td>
</tr>
<tr>
<td>Logical conjunction</td>
<td>And</td>
</tr>
<tr>
<td>Logical disjunction</td>
<td>Or</td>
</tr>
<tr>
<td>String concatenation</td>
<td>&amp;</td>
</tr>
</tbody>
</table>

The mathematical operations have usual precedence and associativity, and are of higher precedence than the relational and logical operators. Parentheses may be used to change the order of evaluation.

**Control Statements**

VB has an if-statement with the following syntax:

```vbnet
If <condition> Then
  <body>
EndIf
```

The if-else-statement has the following syntax: VB has an if-statement with the following syntax:

```vbnet
If <condition> Then
  <if-body>
Else
  <else-body>
EndIf
```

The while statement has the following syntax:

```vbnet
While <condition>
  <body>
End While
```

**Comments**

Comments in VB start with a single quote (' ') and end at the end of a line of text.
Example Program

Here is the handler code for the example program designed in class on 3/21/07 that computes the divisors of an integer. In this code, Button1 is the button that is clicked to make this handler run, TextBox1 is a textbox containing the user input and Label3 is a label where the resulting output is to be placed.

Public Class Form1

    Private Sub Button1_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles Button1.Click

        Dim n As Integer  'Integer input by user
        Dim k As Integer  'Loop counter

        'Initialization statements
        Label3.Text = "The divisors are:
        n = TextBox1.Text
        k = 0

        'Count until k is equal to n
        While k < n
            'Increment k
            k = k + 1

            'Test if k divides into n evenly
            If n Mod k = 0 Then
                'It does, so add it to the end of the result text
                Label3.Text = Label3.Text & " " & k
            End If
        End While

    End Sub

End Class