Creating and Saving Projects

Objectives

Learning Goals

- To learn how to create and save a *Visual Basic* program
- To learn the relevant parts of the *Visual Studio Development Environment*
- To learn the difference between *Design* and *Run* mode

Notes

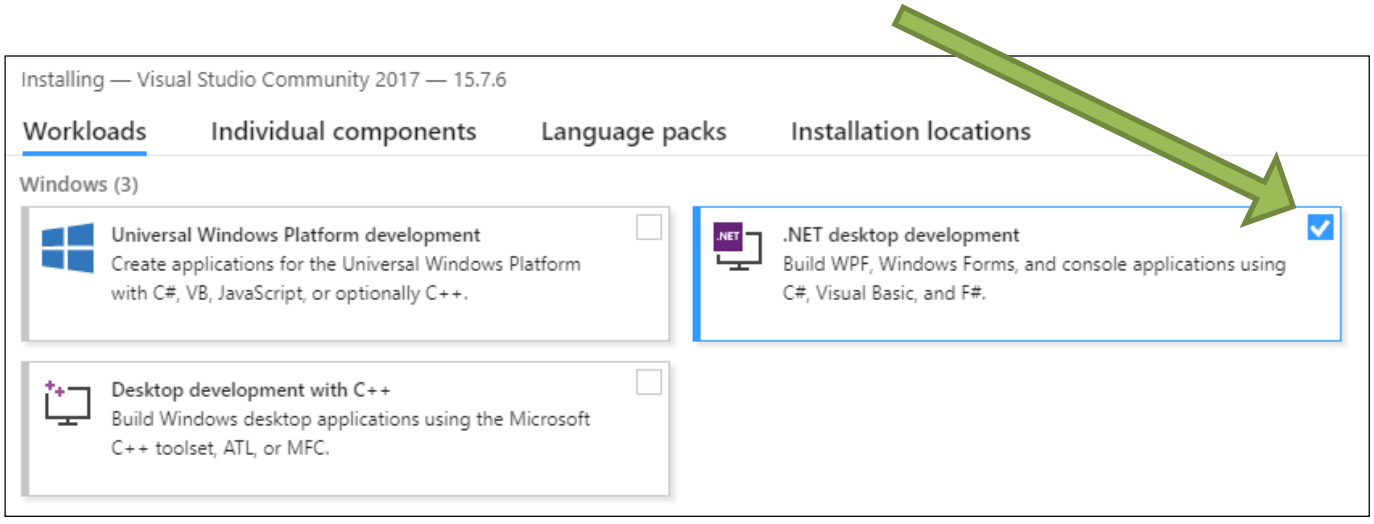
It is advised that students understand how to save solution, project and form files in the Visual Studio environment before starting the first programming example. This will avoid confusion later.

The platform appears complex at first, but it's an extremely powerful and well-designed programming environment.

Download the Free Software (Visual Studio Community Edition)

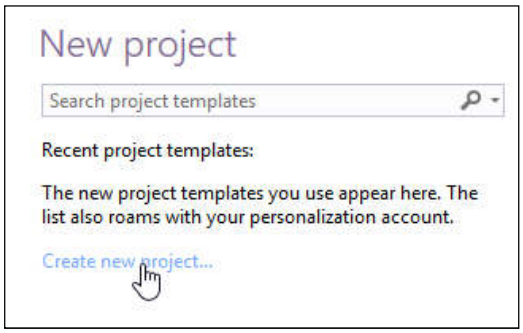
If you have a Windows computer or emulator, you may download the free version of Visual Studio Community and work on these projects at home. There is also a version of Visual Studio Community for the Mac but this has not been tested with the projects (there will likely be differences in the solutions). You will need a Microsoft account to log in, or you may set one up along the way.

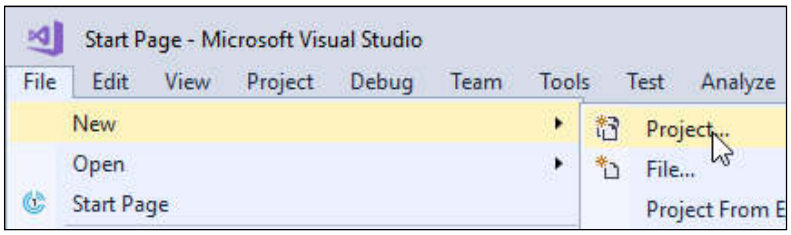
1. Go to the website www.visualstudio.com.
2. Download the Community 2017 version of the software.
3. Follow the prompts.
4. When given the option to include optional components, you must include the *.NET desktop development* package from the *Workloads* menu. This is essential.
5. Complete the installation. You will probably need to restart your computer before using the application.



Starting a New Project

Open Visual Studio and click 'Create new project' on the start page. Alternatively, from within the application, click '**File / New / Project**'.





Select 'Visual Basic' from the menu on the left then 'Windows Forms App (.NET Framework)' from the main window (see the screenshot below).

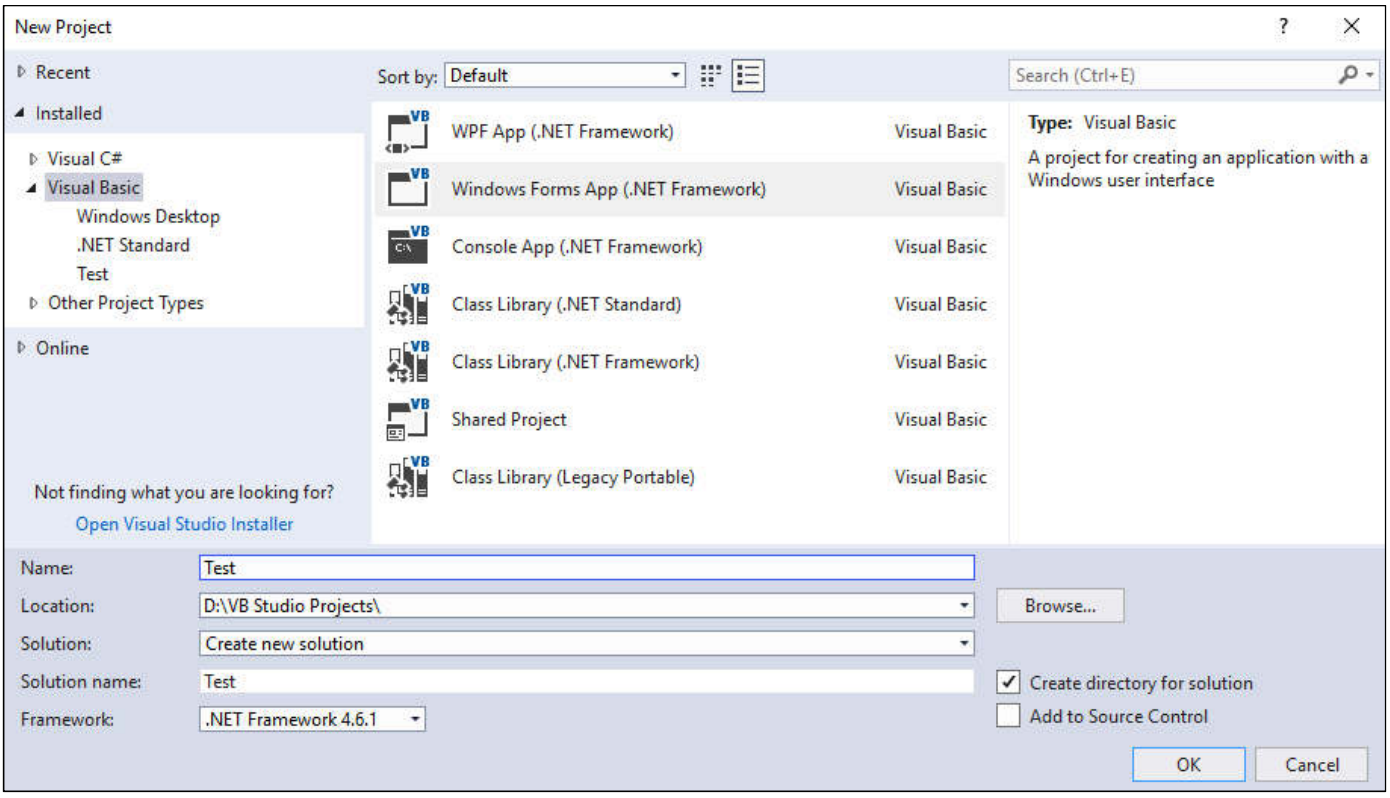
Give the project a meaningful name. This first project might simply be called 'Test' but the names of the actual projects should reflect their titles e.g. '01. Hello Universe 1' or '25. Prime Numbers' etc.

Choose a location for the files. The location should be selected carefully as it will need to be present each time you open the project. A space on your school's network drive usually works well, whereas the computer's hard drive is only useful if you use the same machine on each occasion. You could copy files to a USB if you would like to back them up or transfer them.

You should create a new folder for each project and keep all the associated files within. If the 'Create directory for solution' box is checked, then a folder will be created based on the project name. It's a good idea to number the projects so that your resource folders are kept in order (as in the example names above).

The project name will be copied automatically to the *Solution name* box.

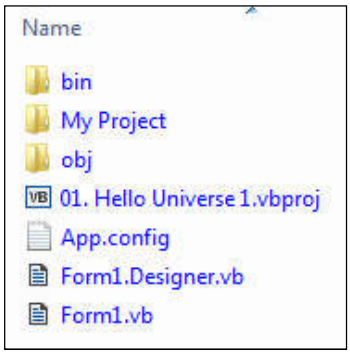
Click 'OK' when you are ready.



Project Files

When you create a project, various files and folders are added automatically. Each file has a file extension which is normally visible in the Windows folder. If yours are not visible, then it may be because of a setting in the *Control Panel: Folder Options* (look under the 'View' tab, where 'Hide extensions for known file types' will probably be checked). Some of the files and folders created are listed below. The images show the results for a project named '01. Hello Universe 1'.

.sln file	The solution file. Click on this to open the project.
.vbproj file	Contains all the information about the project.
.vb files	Forms are saved as separate files with the .vb extension.
.resx file	Created if icons or images are used on forms.
bin folder **obj** folder	Created automatically, containing binary, debug and executable files. These must be left in place.
.exe file	The executable file. This is your finished product.

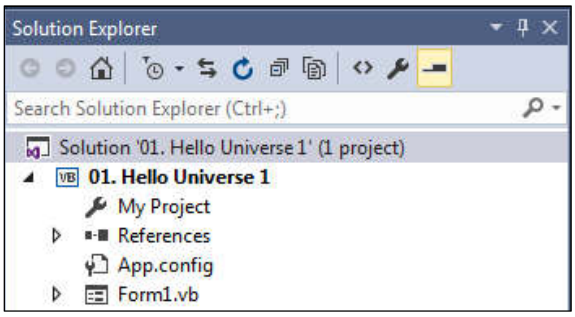


Moving or deleting any project files can result in the project failing to open correctly. This is because the application cannot find the various component files it needs. The best way to relocate or backup a project is to copy the entire folder after it has been saved and the project closed.

The Solution Explorer

This shows the various files in the *Development Environment*.

The keyboard shortcut used to display the *Solution Explorer* is 'Ctrl + Alt + L'.



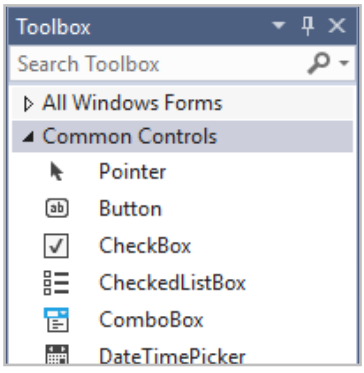
Designing the Interface

The Toolbox

A program interface is created by selecting *Objects* (or *Controls*) from the toolbox and drawing them on the *form*. Double-clicking an object in the toolbox will automatically place an instance of the object on the form.

The toolbox is found on the left side of the screen but can also be accessed from the 'View' menu. Use the pin in the top-right to turn off 'Auto Hide' and keep the toolbox on display.

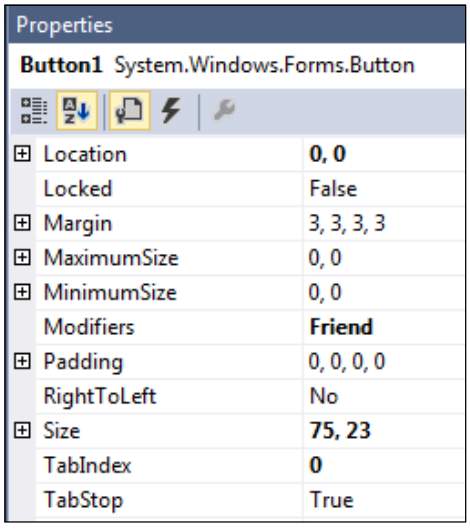
The keyboard shortcut used to display the toolbox is 'Ctrl + Alt + X'.



The Properties Window

Properties for each object are set using the *Properties* window. It is strongly recommended that you set the name of each object before editing any other properties and code.

The keyboard shortcut used to display the *Properties* window is 'F4'.

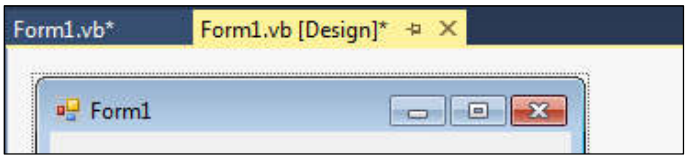


Designing the Interface (continued)

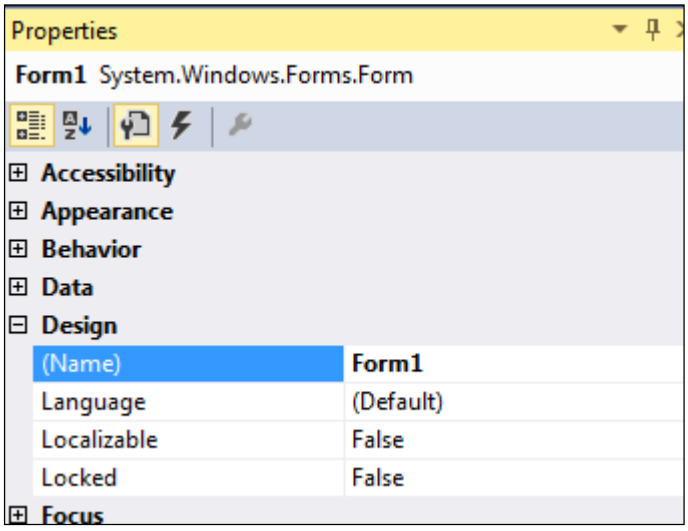
Names of Objects

Type of Object	Number	Names of Objects
Form	1	Form1
TextBox	1	txtOut
Button	4	btnHello, btnGoodbye, btnClear, btnExit

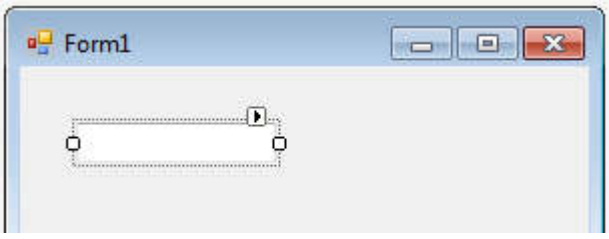
The form will be added automatically when the project is created. Double-click on the form (or any object place upon it) to open the code window, then use the tabs to switch between *Design View* and the code.



If necessary, you can name the form using the *Properties* window in the bottom-right (you might need to close a few sub-menus before you will see the *Design* section).

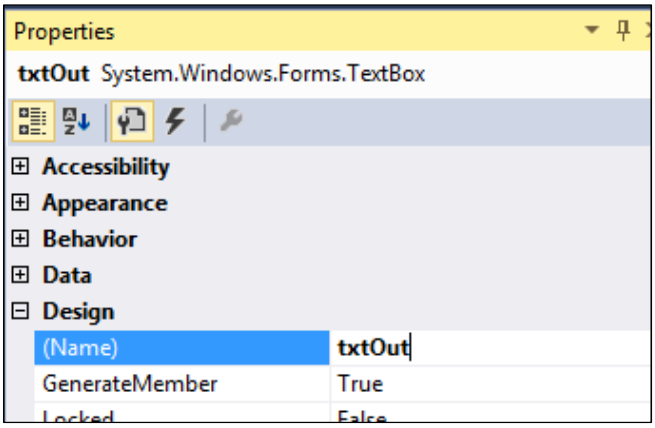


Double-click on an object in the toolbar to add it to your form and then drag it into place.



Again, name the object using the *Properties* window.

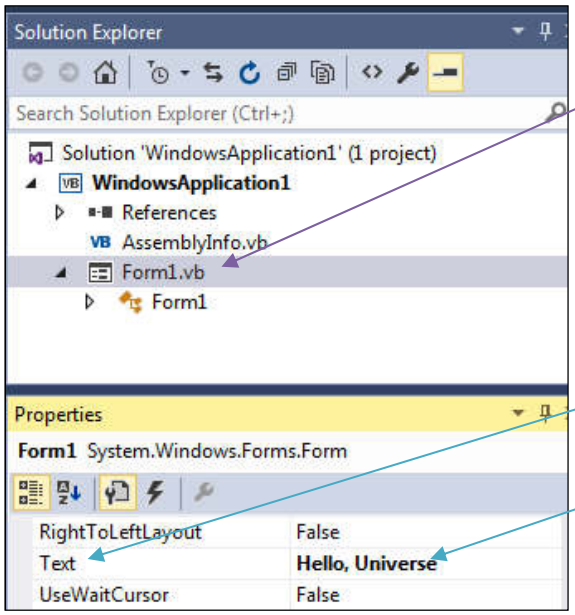
Note: *Make sure the correct object has been selected in your form before naming it.*



Initial Properties of Objects

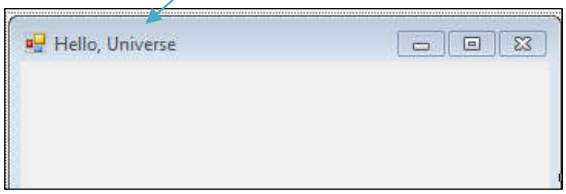
If the Solution *Explorer* is hidden at any time, open it by selecting '**View / Solution Explorer**' from the menu or using the shortcut 'Ctrl + Alt + L'.

Form Title

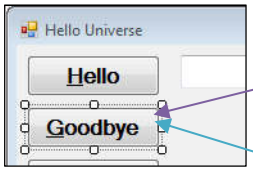


If necessary, double click on 'Form1.vb' to open the form in Design View. The *Text* control is under the *Appearance* sub-menu in the *Properties* window.

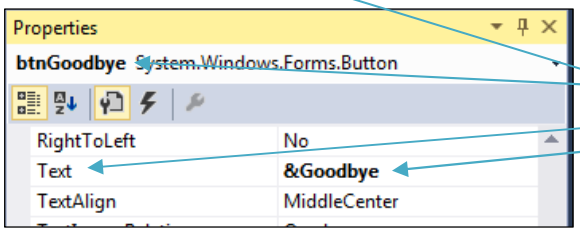
Object	Property	Initial Value
Form1	Text	Hello, Universe
...	...	...



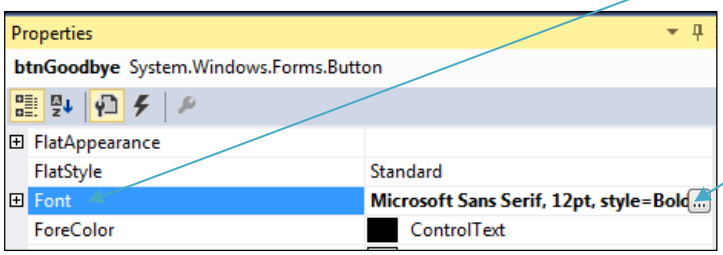
Button Text



Select the button first



Object	Property	Initial Value
btnGoodbye	Text	&Goodbye
	Font	Bold, 12
...	...	...



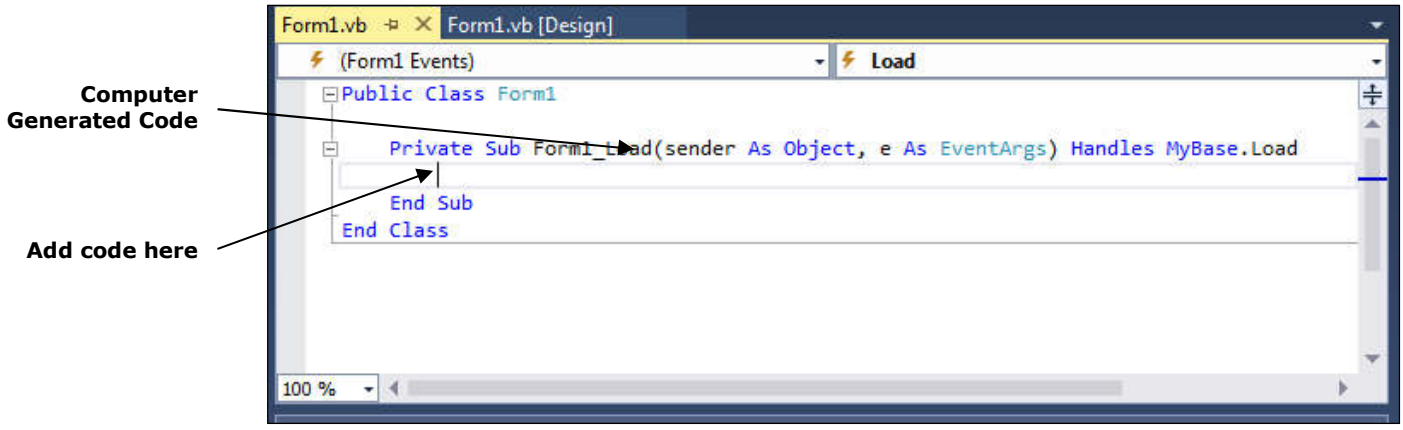
Designing the Interface (continued)

The Code Window

Visual Studio is considered an *Object-Oriented Programming Environment*. Objects have *Methods* and *Classes* associated with them. Programming code is triggered by events associated with these objects.

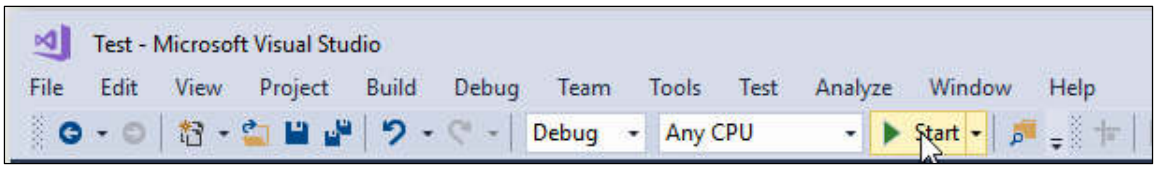
The *Code Window* can be accessed by double-clicking on any object. Alternatively, you may navigate with the tabs or use the shortcut key combination 'Ctrl + Alt + 0 (zero)'.

Visual Studio events contain a lot of computer generated code. For introductory programming, most of this can be ignored.



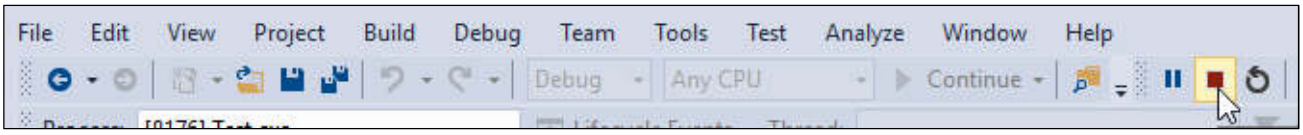
Testing and Building a Program

To test a program it must be *Run*, *Built* or *Started*. This puts the program into *Run Mode*. During this time, changes to objects cannot be made. Some code, however, can be modified.



Programs can be started by clicking on the *Start* button or pressing the 'F5' key.

To return to *Design Mode*, simply click the *Stop* button or exit from within the program.



To return to Design Mode, click the *Stop* button or exit the program.

Selecting 'Build / Build Solution' will create an exe file for your project. This can be found in the bin folder.

Program 1. Hello Universe 1

Introduction

These task sheets are designed to help **you** create solutions to visual basic projects, rather than provide solutions for you. Use the information given, test everything frequently and don't be afraid to make you own adaptations. As with all computer programming, however, small changes in one place can have a devastating effect in others. Make sure that you fully understand what you are doing before altering the suggestions made here. Always save a working project before making experimental changes.

Remember, when programming computers, you have to be **accurate**. Any misspelt object names or bits of code will cause errors. Having said this, it is very normal for programmers to spend much of their time trying to find out why things are not working properly. It's all part of the fun.

Objectives

Program Purpose

- Display 2 messages
- Clear the messages
- Exit the program

Learning Goals

- To create a simple visual basic program
- To display text in a textbox
- To clear text from a textbox
- To use buttons

Design Notes

When creating the buttons, set the font for the first one then copy and paste it.
It is helpful to set common properties such as *FontSize* and *Bold* before copying and pasting.

Interface

The picture on the right shows an example interface for this task. You could copy this one exactly or you may prefer to design your own.

Names of Objects

Type of Object	Number	Names of Objects
Form	1	Form1
TextBox	1	txtOut
Button	4	btnHello, btnGoodbye, btnClear, btnExit

Initial Properties of Objects

Object	Property	Initial Value
Form1	Text	Hello, Universe
	StartPosition	CenterScreen
txtOut	Text	Leave blank
	Font	Bold, 12
btnHello	Text	&Hello
	Font	Bold, 12
btnGoodbye	Text	&Goodbye
	Font	Bold, 12
btnClear	Text	&Clear
	Font	Bold, 12
btnExit	Text	E&xit
	Font	Bold, 12

The ampersand symbol (&) creates a keyboard shortcut for the user. The letter after the '&' will be underlined after the user presses the 'Alt' key. They can then use 'Alt + underlined letter' when in run mode instead of clicking the button.

Events – Code

Private Sub btnHello_Click(ByVal sender As ...)

txtOut.Text = "Hello, Universe"

End Sub

Private Sub btnGoodbye_Click(ByVal sender As ...)

txtOut.Text = "Goodbye, Universe"

End Sub

Private Sub btnClear_Click(ByVal sender As ...)

txtOut.Clear()

End Sub

Private Sub btnExit_Click(ByVal sender As ...)

End

End Sub

```
Private Sub btnHello_Click(ByVal sender As
    txtOut.Text = "Hello, Universe"
End Sub

Private Sub btnGoodbye_Click(ByVal sender
    txtOut.Text = "Goodbye, Universe"
End Sub

Private Sub btnClear_Click(ByVal sender As
    txtOut.Clear()
End Sub

Private Sub btnExit_Click(ByVal sender As
    End
End Sub
```

Suggestions for Consolidation and Extension

1. Modify the program to display additional messages by adding additional textboxes and buttons.

2. Experiment with the following properties of the textbox:
   - TextAlign
   - Multiline
   - BackColor, ForeColor
   - BorderStyle
   - Font

3. Experiment with the following properties of the buttons:
   - Image, BackgroundImage, ImageAlign
   - BackColor, ForeColor
   - Font
   - Cursor

Further Information

1. Naming Objects

In Visual Basic, a naming convention applies to all objects. A three-letter prefix is used to denote the type of object. This is followed by a name that identifies its purpose. The designer of the program must make up this name. Naming is very important when creating and testing code.

For example: **txtDisplay** *a textbox that displays output.*

btnExit *a button that causes the program to end.*

2. Object.Property Syntax

When writing Visual Basic code, properties are referred to using the *Object.Property* syntax. This is where the object is named, followed by a dot, followed by the property.

For example: **txtDisplay.text** **btnExit.text** **Form1.backcolor**

Whilst the program is running, most of the object properties can be changed using simple code.

For example: **txtDisplay.text = "Hello, Universe!"**

Questions

1. In the program *Hello Universe 1*, which three different types of objects were used?

2. Name two of the properties that these three objects have in common.

3. Name a property that is unique to each object.

4. Which key can you press to run a program?

5. Which key can you press to display the *Properties* window?

6. Describe the difference between the two tabs – *Form1.vb* and *Form1.vb (Design).*

7. Write a line of code that displays the text "I like studying Visual Basic" in a textbox named *txtDisplay*.

8. Write a line of code that clears the textbox named *txtDisplay*.

9. Name the Visual Basic code word that ends a program.

10. Describe the difference between *Design Mode* and *Run Mode*.

11. Which keyboard shortcut can you use to display the *Project* window?

12. What is the file extension for a Visual Basic solution file?

13. What is the file extension for a Visual Basic form file?

14. Identify some of the other files and folders that the application creates automatically.

Further Extension Activities

Design a program with 5 buttons and 3 text boxes. The program should use 3 of the buttons to display different messages in each text box. Another button should clear the 3 textboxes. The last button should exit the program. Set the font to Arial, 14, italic. Remember to name all objects appropriately.