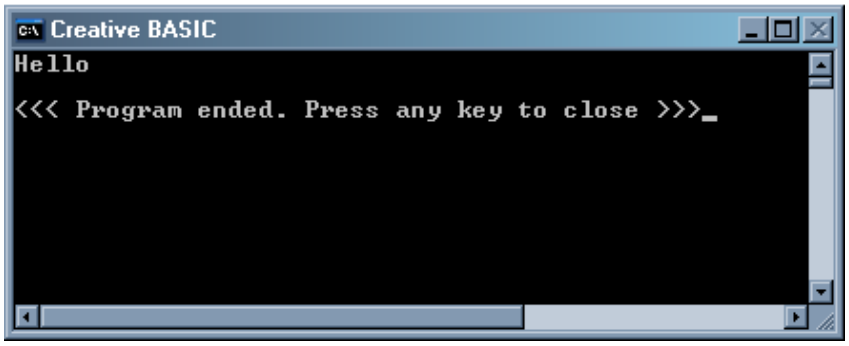
# Using Windows

Here are some window types, or styles, that you might consider using.

## A Console Window



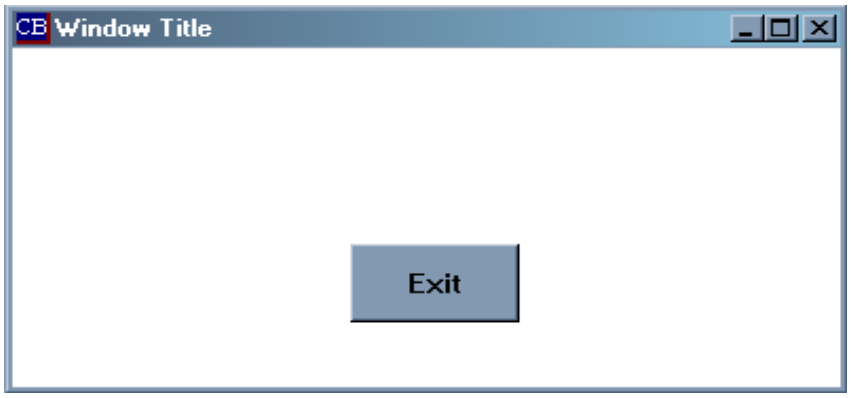

This has a limited user interface.

Input and output is limited to characters A - Z, 0 - 9, and the ASCII characters 0 - 255.

The screen is entirely character based - only simple graphics built from the ASCII character set are possible. Colours can be used.

Capable of considerable computation provided input and output to screen are kept to a few values.

## A Normal Window




The workhorse of Windows programming, it can display graphics and text in full colour.

All the Windows controls can be used within it, giving an excellent user interface. Limited to what can be done using one window at a time -

## Frame and Child


Sometimes you will need an unchanging surround, but with several 'child' windows contained within it.

The user selects which child window he needs by use of a menu (as in the example shown), or by using 'forward and back' buttons.

This example shows a menu at the top of the screen, and a status bar at the bottom.

The information shown on each child window can be maintained as the user moves back and forth between windows. Any change on one window can update the displays on all the others.

Complex applications can be built using multiple windows in this way.

## Window with No Caption


This type of window has no Caption bar and no Minimize, Maximize, or Close buttons.

The example is a small display window showing the time. It can be moved to any desired position on screen, and is closed using the button on the right.

Since it has no caption bar, which is normally required to drag the window around the screen, a trick is required to make dragging possible. We will look at this later..

Of course this style of caption-less window can be made any size you wish. Just don't forget to give it a 'close' button, or you will have no way of getting rid of it.

## A Dialog Box


A dialog box prompts the user to supply information needed by the program before continuing with the application.

It remains open and blocks all other Windows until

## A Skeleton Console Window

Open a new program using the **'File - New - Source File'** menu. You now have a new, blank editor window. Type in the following instructions:

```
openconsole


do:until inkey$ <> ""
closeconsole
end
```

Click on the green Run button, or use the **'Build – Run'** menu, to start the program.

This small program has created a **'Console window'**.

This type of window has limited graphical ability, but it will be useful for testing a number of programming commands.

To close the window, simply press any key.

What do the program instructions mean?

The 'open' and 'close' console commands are obvious. The first opens the window, and the second closes it again.

The command just before 'closeconsole', will be examined later. For now, note that it waits for any key on the keyboard to be pressed.

The ‘End’ command is present in all programs as the last instruction in the main program.

It closes all open windows and files, and terminates the program.

Save the program in a working directory somewhere, using the **‘File – Save As’** menu.

I usually have a directory called **Cwork** in which I keep all useful programs for future use.

Give it a name such as ‘**cbcons.iba**’, so that you can load it again whenever you wish to open a skeleton Console window program.

This simple skeleton will be useful when testing a number of programming examples.

All that will be required, will be to insert test statements in the blank lines between the ‘openconsole’ statement and the closing three statements.

Let us test it out ..

Click on one of the blank lines following the ‘openconsole’ command and type this instruction:

```
Print "My name is John Smith"
```

Click on the green Run button, and Hey Presto, we have something useful.

We have used the **‘print’** command to place text on the screen. As you see, the required text is enclosed in quotes.

Now try this small change:

```
Print "My name is "
Print "John Smith"
```

Run the program again, and you will see the text is now on two lines.

Finally, try this:

```
Print "My name is "
Print
Print "John Smith"
```

Easy isn't it.

If you wish, you could save the test program, using the **‘File - Save As’** menu, and give it a new name, so that it doesn't overwrite your original skeleton program ‘cbcons.iba’.