**Video 1 – Introduction**

**Videos 2 to 7 show how to declare different variable types, assign values to them and display them.**

**Video 2 – Book Details - Working with a String Variable**

As before, create a folder for Week02 inside Code. Create a folder BookDetails inside Week02.

Open the folder inside VISUAL STUDIO CODE and create a new Terminal.

Create, build and run this application as before using: **dotnet new console** and **dotnet build** and **dotnet run** and similar folders and files will be created as well as the default message.

Edit the code as follows:

A screen shot of a computer code

Description automatically generated

The code in lines 4 and 5 can be frustrating when you are developing an application. So these can be commented until the final application is complete.

Continue to add a new variable called title, assign a value to it and display text and the value of it.

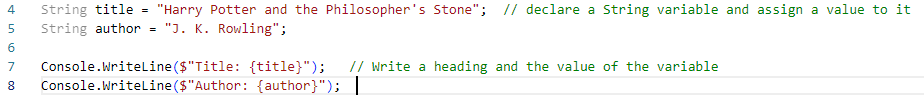
A close-up of a computer screen

Description automatically generated

Build and re-run the application again.

**Video 3 – Book Details - Working with Multiple Strings**

Add another String variable called author, assign a value and display it.



**Video 4 – Book Details - Working with Integers**

An integer is another type of variable which can hold whole numbers.

Int32 is a common one and is a 32 bit integer so can use 32 bits to hold a value. It has a huge range and can hold any whole number between -2 billion and + 2 billion. Numerical variables do not need quotes when assigning values to them.

There are other types like bit64 but this would be at a cost of the application using more memory.

Add an integer variable called pages as follows:

A white background with colorful text

Description automatically generated

**Video 5 – Book Details - Working with Floating Point Numbers**

A Floating-point number is another type of variable and is used to store a number with a decimal part such as 10.45.

Double is the type for a double precision floating point number.

Add a floating-point variable called price as follows:

A screenshot of a computer

Description automatically generated

Can also use price = 10.50d; to infer a floating-point number

When you run it, the value is displayed as 10.5. You can format the price similar to the date in week 1 by using a colon after the variable where the value is to be displayed and use format specifiers as follows:

Edit the code as follows:

Run the application again and the price should be formatted accordingly.

**Video 6 – Book Details - Working with Date/Time Variables**

The DateTime variable was also used in week 1 to display a date and/or time.

Add a DateTime variable called pubDate as follows:





Build and re-run the application.

Modify the code to display the date in a specific format as follows:



Run the application again and note the different format.

**Video 7 – Book Details - Working with Boolean Variable**

A Boolean variable can only hold 1 or 2 values, either True or False.

Add a Boolean variable called inStock as follows:





Build and re-run the application.

The full program should be similar to the one below:

A screenshot of a computer code

Description automatically generated

Note the use of comments throughout the code.

All these examples demonstrate how to create different types of variables and then assign hard coded values to them. Videos 8 to 12 show how to get the client to enter values for these variables which produces dynamic results.

**Video 8 – Book Details - Inputting Strings**

Modify the code as follows:

A computer code with black and yellow text

Description automatically generated

A screenshot of a computer program

Description automatically generated

Build and re-run the application. A warning will be displayed connected to the red wavy line below some of the variables. This is because these variables have been declared but are not being used. But this will change in the next few examples.

**Video 9 – Book Details - Inputting Integers**

Modify the code as follows:





The variable pages was declared as Int32 which means it can hold a whole number as a value.

When a client enters a value, the expectation is that the value entered is a String type. This is not the case in this example so a red line appears to show an error, which will also be highlighted if you try and build it. You need to convert what the client enters to an integer so that the integer can be assigned to the Integer type variable called pages. This is done by using the Int32.Parse command as follows:



Run the application again and it should be error free.

C# is strongly typed so you cannot assign one set of data to a variable that has a different type.

**Video 10 – Book Details - Inputting Floating Point Numbers**

The Int32.Parse command can be applied in the same way to assign non string values entered by a client to be parsed to another type of variable.

Add the code as follows:





Build and re-run the application again.

**Video 11 – Book Details - Inputting Dates**

The DateTime.Parse command can be applied in the same way to assign non string values entered by a client to be parsed to another type of variable.

Add the code as follows:





Build and re-run the application again.

**Video 12 – Book Details - Inputting Booleans**

The Boolean.Parse command can be applied in the same way to assign non string values entered by a client to be parsed to another type of variable.

Add the code as follows:





Build and re-run the application again.

**Video 13 – Book Details - Running the Final Application**

Comment out the last two lines and run the final application. The code should be similar to below:

A screenshot of a computer program

Description automatically generated

The previous videos focus on simple variables which can only hold a single value, so a named value pair. Sometimes you will want a variable to hold multiple values. This can be using Arrays and Lists.

Videos 14 to 18 use an application BoysNames and use an array.

Videos 19 to 24 use an application GirlsNames and use a list.

**Video 14 Boys Names - Declaring a String Array**

As before, create a folder for Week02 inside Code. Create a folder BoysNames inside Week02.

Open the folder inside VISUAL STUDIO CODE and create a new Terminal.

Create, build and run this application as before using: **dotnet new console** and **dotnet build** and **dotnet run** and similar folders and files will be created as well as the default message.

Edit the code as follows:

A screenshot of a computer code

Description automatically generated

We want to create an array to hold multiple boys names. Arrays in C# are not dynamic which means that you need to specify how many values the array will hold.

Add the following code above line 4.



Notice the use of square brackets.

**Video 15 Boys Names - Assigning Values to an Array**

The array boysNames can hold 5 values. Index numbers are used to distinguish between these values, starting at zero.

After the array has been created, add the code as follows:

A white background with black text

Description automatically generated

This code assigns values to each of the index numbers. Build and re-run the application.

**Video 16 Boys Names - Displaying the Values Assigned to an Array**

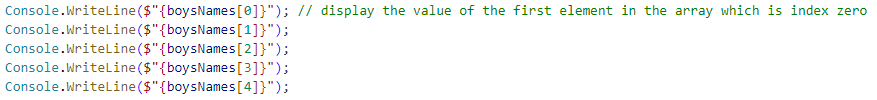
Add the following code to display the value assigned to the first element in the array.



Notice the code uses the dollar symbol for interpolation and curly brackets to display value of a variable or an array element.

Build and re-run the application again. Check the first name is displayed and then copy the Console statements to display all the boys’ names, changing the index number for each one.

Finally, the code should be like below:



**Video 17 Boys Names - Making a Literal Array Declaration**

The previous example uses 6 lines of code to create and populate a String array. 1 line to declare an array of 5 elements and 5 lines to populate it.

This can be done in 1 line of code using a literal declaration. Comment the previous code. This can be done as a block by using Edit, Toggle Block Comment.

Edit the code as follows:



This code creates an array using a literal declaration on one line which creates the String array and populates the values of all the elements inside curly brackets.

It would also work by omitting the 5 because the number of values lets C# know that there are 5 values so we don’t need to specify the size of the array.

It can also be simplified further as follows:



We don’t need to create a new instance of the String array because the code at the start of the line defines a String array.

Make each of the changes above, building and running the applications after each change.

There are many resources, web sites, books etc on creating arrays and some may use different methods, so it is useful to know alternative ones and shortcuts.

Comment out the last 2 lines and run the final application. Finally, the code should be like below:

A screenshot of a computer

Description automatically generated

**Video 18 Boys Names - Running the Final Application**

Run the application from Windows Explorer using the executable file BoysNames.exe

**Video 19 Girls Names - Importing the System Collection Generic Namespace**

As before, create a folder for Week02 inside Code. Create a folder GirlsNames inside Week02.

Open the folder inside VISUAL STUDIO CODE and create a new Terminal.

Create, build and run this application as before using: **dotnet new console** and **dotnet build** and **dotnet run.** Edit the default code to display the message “Girls Names”.

A problem with Arrays in C# is they are fixed size so if you want to add another value then you need to create a new array.

A better way is to use a List which is dynamic so you can add and remove elements as required.

The System namespace has a whole library of code that comes with the .NET SDK framework.

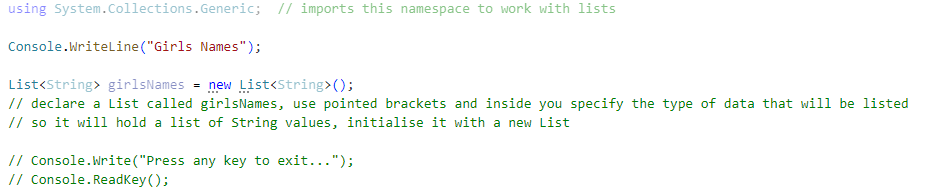
Lets us work with variables, format them, declare arrays, all the common stuff you do in programming gets included in this namespace

If we want to use a List then we need to use another namespace as it is not part of the System namespace. It is still part of the .NET framework but not available by default so we need to import it because it is less commonly used. We install it by adding the code below at the top of the file:



**Video 20 Girls Names - Declaring a List of Strings**

Modify the code as follows:



This creates a List to hold values of girls’ names. Build and run the application.

In previous versions, without the code, using System.Collections.Generic; you would get an error. This is no longer the case with .NET8 so you don’t actually need to import the namespace.

**Video 21 Girls Names - Populating a List of Strings**

We want to assign values to the list as follows:

A white background with text

Description automatically generated

This code adds the names to the list created in the previous step.

Unlike the array before, we can add or remove names as required. Use add and remove methods.

**Video 22 Girls Names - Displaying the Values added to a List**

We want to display the values in the list as follows:



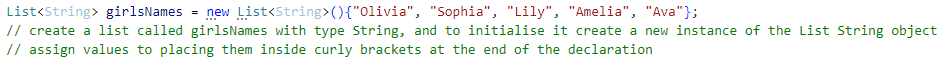
Copy the code and edit the index number to display all the names in the list.

Build and run the application. This is not the most efficient way as it relies on you knowing the index number.

**Video 23 Girls Names - Making a Literal List Declaration**

Comment out the lines of code which declares the List and added names to it.

Similar to making a literal declaration with an array, it can be done with a list as follows:



Comment out the final two lines. Build and run the application again to display the names from the Literal List.

**Video 24 Girls Names - Running the Final Application**

Run the application from Windows Explorer using the executable file GirlsNames.exe

**Video 25 Simple Calculator – Inputting an Integer**

As before, create a folder for Week02 inside Code. Create a folder SimpleCalculator inside Week02.

Open the folder inside VISUAL STUDIO CODE and create a new console application.

**A computer code with text

Description automatically generated**

Build and run the application. The client should be prompted to enter an integer which will be displayed.

**Video 26 Calculator – Inputting a Second Integer**

Modify the code as follows:

A screenshot of a computer code

Description automatically generated

Build and run the application.

**Video 27 Simple Calculator – Performing Addition**

Add the code as follows:



Build and run the application multiple times with different numbers. The result should change accordingly.

**Video 28 Simple Calculator – Performing Other Arithmetic Operations**

Add the code as follows to perform other basic arithmetic operations:

A group of numbers with red and blue letters

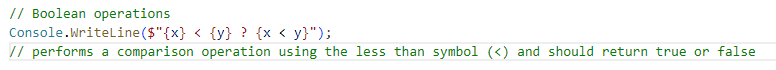
Description automatically generated with medium confidence

Build and run the application.

**Video 29 Simple Calculator – Performing A Less Than Operation**

This looks at Boolean operations which compares values together to produce a Boolean value which is either true or false.

Add the code as follows to perform a Less than operations:



Build and run the application and the result should display True or False.

**Video 30 Simple Calculator – Performing Other Comparison Operations**

Add the code as follows to perform other Boolean operations:

A screenshot of a computer

Description automatically generated

Comment out the last 2 lines, build and run the final application from VISUAL STUDIO CODE.

**Video 31 Simple Calculator - Running the Final Application**

Run the application from Windows Explorer using the executable file SimpleCalculator.exe

**Video 32 Custom Greeting – Display the Current Date and Time**

The last project was based on Arithmetic operations so we will move onto Branching structures.

As before, create a folder for Week02 inside Code. Create a folder CustomGreeting

Open the folder inside VISUAL STUDIO CODE and create a new console application.

Add the following code to display the current date and time:

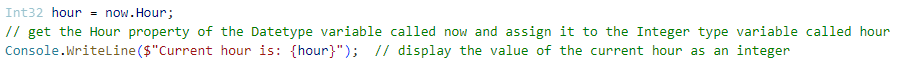
A white background with black and red text

Description automatically generated

Build and run the application. The current date and time from the system should be displayed in full.

**Video 33 Custom Greeting – Display the Current Hour**

If you want to get a part of the Date and/or Time variable, you can use the appropriate property. Add the following code to display the current hour property:



Build and run the application. The current hour from the system should be displayed.

**Video 34 Custom Greeting – Implementing an If statement**

We will use an If statement to produce a custom message dependant on the value of the hour variable.

An If statement will execute a code block if a Boolean expression is true, otherwise it will be ignored. It uses the format: **if (boolean expression) {code block}**

Add the following code to display the custom message:

A computer screen shot of a white background

Description automatically generated

Build and run the application. The custom message will only be displayed if the condition is true.

Int32 hour = 7;  // can hard code a number for testing purposes

**Video 35 Custom Greeting – Implementing an Else If statement**

We can add additional expressions using Else If clauses to display different custom messages.

So if the first statement is false, move on and try again.

Add the following code to display the custom message:

A screenshot of a computer program

Description automatically generated

This code can display different custom messages, depending on the value of hour.

Int32 hour = ?;  // can hard code a different number to see different clauses being executed for testing purposes try 11, 14, 20, 23

**Video 36 Custom Greeting – Implementing an Else statement**

The previous example will display a custom message if the value of the hour is between 5am and 10pm. This means that nothing will happen if the value is outwith this range. So we can add an Else clause to execute if none of the other conditions are true.

Add the following code to display the custom message:

A close up of a text

Description automatically generated

Comment out the last 2 lines. Build and run the final application from Visual Studio Code.

**Video 37 Custom Greeting – Running the Final Application**

As before, run the final application from Windows Explorer.

**Video 38 Custom Hello World – Generating a Random Number**

As before, create a folder for Week02 inside Code. Create a folder CustomHelloWorld

Open the folder inside VISUAL STUDIO CODE and create a new console application. Build and run the application.

It is useful to be able to generate a random number for testing purposes and C# has a random number generator which can be used as follows:

A screenshot of a computer code

Description automatically generated

Run the application and a random number will be displayed. Run it a few times and hopefully the random number will change.

**Video 39 Custom Hello World – Generating a Random String**

Add the following code to create and display a random language from an array with 4 values:

A screenshot of a computer code

Description automatically generated

Noice the use of language (singular) and languages (plural). This is a common practice using plural for a collection and singular for a single instance.

Build and run the application and repeat.

**Video 40 Custom Hello World – Implementing a Switch Statement**

A Switch statement is an alternative to an If statement to control the program flow. Add the following code:

A white screen with black text

Description automatically generated

Build and run the application. If the random number is 0, 1 or 2 then the relevant greetings should be displayed. If the number is 3 then nothing will happen.

In an If statement the Else clause can be used to specify what to do in all other instances ie where none of the conditions are true. This can be done using the default keyword at the end of a Switch statement as follows:

A white background with black text

Description automatically generated

Run the application a few times and the last Console.WriteLine command will be executed when the random number generated was 3.

**Video 41 Custom Hello World – If versus Switch Statement**

When do you use an If statement or a Switch statement. Either can be used, but as a general rule of thumb:

If you have a variable that can take on a range of values eg hour can be any integer number between 0 and 23, and different things happen depending on different ranged, then use the If statement. Eg messages, Good Morning, Good Afternoon, Good Evening, Good Night

If you have a variable that can only take on discreet values eg language can be 1 of 4 languages then you should use the Switch statement.

**Video 42 Custom Hello World – Running the Final Application**

As before, comment out the last 2 lines, build and run the final application from Visual Studio Code.

**Video 43 Squares – Implementing a For Loop**

As before, create a folder for Week02 inside Code. Create a folder called Squares

Open the folder inside VISUAL STUDIO CODE and create a new console application. Build and run the application.

A For loop has the following format:

**for (condition to go around the loop) {code block to be executed}**

The condition contains 3 parts - when to start counting, when to stop counting and how to increment.

Syntax example: (Int32 i = 1; I <= 10; i++) where an Integer type variable called i is declared, which starts with a value of 1, ends with a value of 10, and increments by 1 each time it goes around the loop.

Add the following code:

A computer code with text

Description automatically generated with medium confidence

Build and run the application. It should execute the code inside the curly brackets 10 times, when i has a value of 1 up to 10.

**Video 44 Squares – Displaying the Squares of Integers**

Modify the code as follows to square the value of i and display the results:



Comment out the last 2 lines, build and run the final application from Visual Studio Code.

**Video 45 Squares – Running the Final Application**

Run the final application from Windows Explorer.

**Video 46 Boys Names – Looping through an Array with a Foreach Loop**

A foreach loop is a variation of the for loop and is used to loop through a collection.

A previous example used an array to create a collection of boys names. Open the folder BoysNames. Previously we used 5 Console.WriteLine statements to display the 5 names in the boysNames array. This is manageable because the number is low. If the number was much bigger then it best to use a for each loop.

Comment out the 5 Console.WriteLine statements and modify the code as follows:

A screenshot of a computer code

Description automatically generated

Notice the use of boysName (singular as it refers to one of the items in the collection) and boysNames (the collection so plural).

Build and run the application. Make sure that you have changed to the terminal for BoysNames.

**Video 47 Girls Names – Looping through a List with a Foreach Loop**

A previous example used a list to create a collection of girls names. Open the folder GirlsNames. Previously we used 5 Console.WriteLine statements to display the 5 names in the boysNames array. .

Comment out the 5 Console.WriteLine statements and modify the code as follows:

**A computer screen shot of a computer code

Description automatically generated**

Build and run the application.

**Video 48 Multiplication Table – Inputting an Integer**

As before, create a folder for Week02 inside Code. Create a folder called MultiplicationTable

Open the folder inside VISUAL STUDIO CODE and create a new console application. Build and run the application.

Add the following code to prompt the client to enter an integer, assign it to the variable x and display it:

A screenshot of a computer code

Description automatically generated

Build and run the application.

**Video 49 Multiplication Table – Implementing a While Loop**

A while loop will only execute if the condition is true. Normally there would be a counting variable before the loop.

A While loop has the following format:

**while (condition to go around the loop) {code block to be executed}**

Add the following code:

A math equation with colorful text

Description automatically generated with medium confidence

This code block will be repeated until i has a value of 11 and the condition is not true.

Build, run the application a few times and enter different numbers.

**Video 50 Multiplication Table – Implementing a While Loop**

A variation of the while loop is the Do while loop where the condition appears at the end of the Do while statement. This means that the code block will always get executed at least one.

Modify the code as follows:

A computer code with text

Description automatically generated with medium confidence

Note the keyword do has been added, the condition has been moved to the end and a semi-colon has been added.

Comment out the last 2 lines, build and run the final application from Visual Studio Code.

**Video 51 Multiplication Table – Running the Final Application**

As before, run the final application from Windows Explorer.