Introduction

What is Programming?

Programming can seem daunting but if you think about it as another form of writing with its own grammar, style and layout then that's a good way to approach it. From the simplest sample to the most advanced application the concepts are the same, there's just more to write - when reading or writing you have words and paragraphs and programming has its own way being written and breaking up functionality however unlike text you can do many different things with the same piece of functionality if you write it in a way. There are two main rules to help keep programming from being too complicated that are KISS and DRY - which is Keep it Short and Simple and Don't Repeat Yourself with those two things in mind that will help you along the way.

Programming is essentially a set of tasks for a computer to perform and rules for it to follow, this essentially can be a self-contained sequence of actions to be performed - these are known as algorithms and form the basic concept of programming. Think of a sequence of actions like how to make a cup of tea, you can even write this down and that's an algorithm there will be steps to follow, some which need to be done in an order, also maybe different choices such as taking milk or sugar - things that vary in programming are known as variables such as how many lumps of sugar to add when making tea.

When creating an algorithm, it's important to have all the steps explained - each part of this can be a block of functionality such as fill kettle or pour water - if there's a lot of similar steps together you can group them together, much like a paragraph - in programming these are known as functions but may also be referred to as methods. These functions may be used to perform a small sub-set of tasks and can contain their own variables but it's also possible to supply these functions with values - these are known as parameters and a function can use these parameters to use for its task and some functions can even return a value, for example you can have a function to add two numbers together and return the answer.

Just like in writing there's words you need to use to make it make sense such as verbs and nouns and programming has its own which are often known as key or reserved words - these can dictate what a variable might be, can it just be whole numbers which are known as integers or a piece of text which is known as a string. Key words can also set how functions may be used in programming - in a lot of languages there is a concept known as a class, this is where all the code - which is what makes up programming is grouped together to represent something or some common functionality.

There will be functions that can be used only inside one of these classes - those are known as private, and there are those that can be used both inside and outside the class - these are known as public, both functions and variables can be private and public. There's also a special kind of variable called a property, where you can get a value from them or set their value - these are most useful for parts of a class that you want to use such as if a class were to represent a Car you could have a property for number of doors or one for colour. You can also use classes to help organise code but also in many programming languages the minimum amount of code needed can itself be a class and will often have a main method which will be the start of the code to be used.

When you want to write something you'll use a word processing application or something similar, in programming this is also the case as all programming languages have their preferred programming environment - these are often called Integrated Development Environments or IDEs and these allow you to write your code and also make it easier to read with colourisation of keywords and line numbers to help you find parts you have written, there may be other features to help you find out what things you can type in to access additional functionality or related actions. To perform the actions that have been defined in the code it is compiled, this means it is turned into instructions the computer can understand, some programming languages have an additional layer for this but essentially down at the "bottom" there's code the computer uses to run or execute your actions.

The best way to start programming is to write small examples that use the various features of a programming language and perform familiar actions such as addition or subtraction using the programming language to see how it works, programming languages are very good at doing these kind of actions so they're a good way to get started, then you can build up from there - you probably did something similar when learning to read and write where smaller examples can be used to get started and programming should be no different. The best thing to do is as things get more complicated then break them down - you'll have a few lines of code grouped together in functions just like sentences in paragraphs and don't put too many together so that if there's a mistake you can spot it, just like proof-reading you may need to just need to concentrate on small sections at a time to help find anything wrong.

Also don't worry if something doesn't work the first time around, maybe like the making tea example you forgot to put in a step for filling the kettle, sometimes you can learn a lot more from making a mistake to see what you might need to look out for, in programming you might have a variable you expect to be larger than ten but you never set it to anything larger than nine instead might be the kind of problems you encounter, most mistakes in programming are down to assuming something will be in a given state at a particular time, when you perform actions in your code it will go through various states, by breaking up your code into smaller sections you can examine these states when the code is being executed, often called debugging, to see if things are the way they should be, or not in programming!

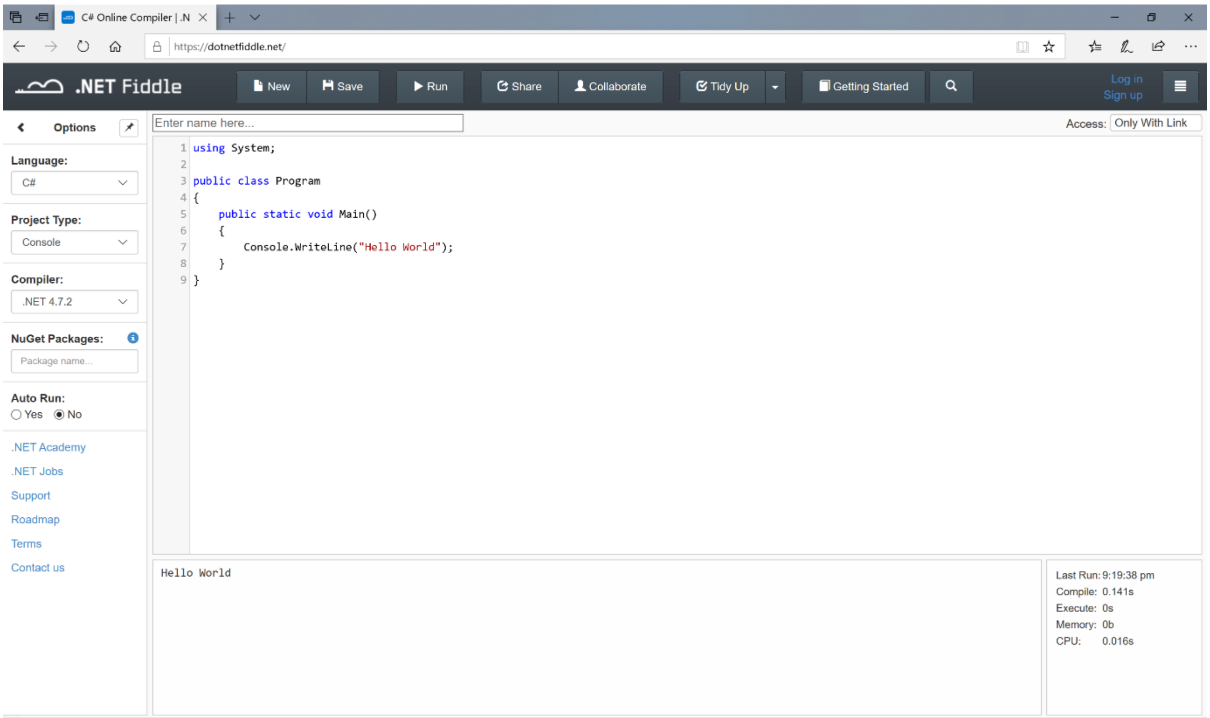
What is C#?

C# is what you'll be learning to write code in - pronounced "C-sharp" like the musical note, this language is like many others although they might have slightly different keywords and other minor differences but belongs to a well-established family of programming languages. C# began as part of the Microsoft .NET Framework - this is one of those layers mentioned previously which executes your code and issues the actual instructions your computer will perform when it runs, it also has all the additional functionality you can take advantage of as part of the framework which contains commonly used and needed functionality so generally you only need to write what you need that's unique to your programme and if you need to do something like open a file then the libraries that make up the .NET Framework are there plus it's possible to use libraries written by other people or companies in your applications but as a beginner you'll just be starting out with the basics but it's nice to know you can take advantage of more functionality should you need it.

C# supports many features common to modern programming languages and has its own structure with blocks of code between curly braces being the most obvious which helps you clearly see where a function or a class starts and ends, it has a variety of keywords you can store the information you need with variables to store the type of information be that integers for whole numbers like 1 to 10 and strings to store text like "Hello World!", each piece of functionality you can use in C# will be introduced and explained along the way but you see many of the things mentioned that are common to all programming which you can apply to other programming languages - but also learn many useful things that you can do in C# to make things easier or to do the actions you need.

What is dotnetfiddle.net?

To run any C# code during this workshop all you need is to be able to access dotnetfiddle.net in your browser, there are two main parts a larger upper text area which will contain the code you need to type in and a lower text area which will show any output:



Access: Only With Link

Language: C#

Project Type: Console

Compiler: .NET 4.7.2

NuGet Packages: Package name...

Auto Run: ○ Yes ● No

```
using System;

public class Program
{
    public static void Main()
    {
        Console.WriteLine("Hello World");
    }
}
```

```
Hello World
```

Last Run: 9:19:38 pm
Compile: 0.141s
Execute: 0s
Memory: 0b
CPU: 0.016s

It will only be necessary to type in any code into the larger upper text area which by default when you first visit dotnetfiddle.net will sow the following:

```
using System;

public class Program
{
        public static void Main()
        {
                Console.WriteLine("Hello World");
        }
}
```

Any output that your code will produce in dotnetfiddle.net will appear in the lower text area, the default example will show the following:

```
Hello World
```

Anything you type will be "run" as you enter it however you can change the "Auto Run" to No if you prefer to select "Run" from the top instead.

Overview

In this workshop we'll start with a small amount of code you'll need to write to get something to happen which is to output something on the screen - this is a good way of seeing the steps of getting something written and working without too much code to go through and work out what it does, anything new will be explained along the way plus various concepts will be repeated and used in different ways just like it's possible to use language in different ways in writing, programming is the same but it's okay to take time to understand something that's new in programming just as it was when learning to read and write.

Also you can always stop and go back over an example and there'll be other concepts you don't see in normal writing such as being able to repeat some actions - called looping or iteration, create decisions or branches and dealing with input and output but always using something that you've learned in a previous part of the workshop should help, you'll not only learn more about programming in general but also how to use the tools and features of a programming language used by millions of people all around the world used in portable devices, to complex business processes, major websites and software applications many of which you've probably seen or used.

C# will be introduced in this workshop in phases with many of the core features you'll need to create small or complex applications introduced, so anything new will be explained along the way so you can always go back later if you're not sure of anything - everyone has been a beginner at some point and it's never too late or too early to learn to code and C# opens many options to take skills to mobile, desktop, cloud or even for games and new devices such as HoloLens but the smallest and best application is an old concept known as a console application, this allows you to see all your code in once place and easily see the output - don't worry if you're not familiar with the console these days but many years ago it was the only option for an application but still be able to take advantage of the latest features.

To make things even easier you don't need to use any programming tools or IDE such as Visual Studio for C# but will instead use a website called dotnetfiddle.net. There you can type in your Code and see what happens when it runs and there's nothing extra to install. However you can of course install “Visual Studio 2019 Community Edition” on your PC, then “Create a new project” and select “Console App (.NET Core)” and then input any code into the “Program.cs” file as this is the same code from that point this workshop will cover if you want to go through the exercises using the full tools available, but to keep things simple at first it's recommended you use something like dotnetfiddle.net for now.