FROM FLOWGORITHM TO C#.

Poggie

02.01.2024

// TODO.

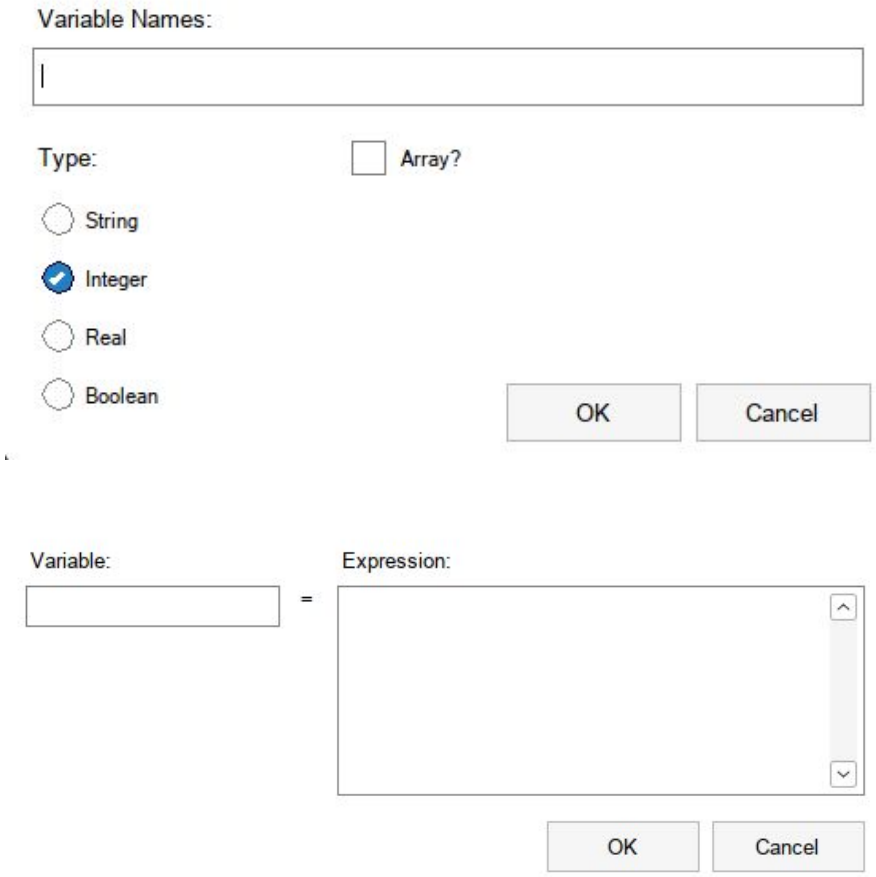
Quick Recap

Project Setup

Simple C# Programs

More on Variables & Expressions

Exercises



Quick recap.

Input / Output

Variables and primitive types

Expressions

PROJECT SETUP.

Visual Studio 2022

Open recent

Search recent (Alt+S)

[redacted]

Get started

Clone a repository
Get code from an online repository like GitHub or Azure DevOps

Open a project or solution
Open a local Visual Studio project or .sln file

Open a local folder
Navigate and edit code within any folder

Create a new project
Choose a project template with code scaffolding to get started

Continue without code →

Create a new project

console app

Clear all

Recent project templates

- ASP.NET Core Web App (Model-View-Controller) — C#
- Console App — C#
- ASP.NET Core Web API — C#
- Class Library — C#
- MSTest Test Project — C#
- Console App (.NET Framework) — C#
- Blank Node.js Console Application — JavaScript

C# | All platforms | Console

Console App
A project for creating a command-line application that can run on .NET on Windows, Linux and macOS
C# Linux macOS Windows Console

Console App (.NET Framework)
A project for creating a command-line application
C# Windows Console

Other results based on your search

Console App
A project for creating a command-line application that can run on .NET on Windows, Linux and macOS
Visual Basic Linux macOS Windows Console

Console App (.NET Framework)
A project for creating a command-line application

Additional information

Console App

C# Linux macOS Windows Console

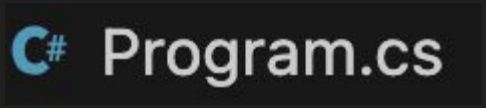
Framework

.NET 6.0 (Long Term Support)

- [x] Do not use top-level statements

GETTING IN THE CODE.

Where we write our C# code.



```
using System;

namespace Example {
    0 references
    class Program {
        0 references
        public static void Main(string[] args) {
            Console.WriteLine("Hello, World!");
        }
    }
}
```

C# uses curly braces to group things together

This denotes a ‘block’ of code

They help us keep track of what parts of the code are related

A statement is a line of code that performs a basic operation.

```
using System;

namespace Example {
    0 references
    class Program {
        0 references
        public static void Main(string[] args) {
            Console.WriteLine("Hello, World!");
        }
    }
}
```

All statements must end in a semi-colon !!

The statement in this program is a print statement, it displays a message on your screen.

```
using System;

namespace Example {
    0 references
    class Program {
        0 references
        public static void Main(string[] args) {
            Console.WriteLine("Hello, World!");
        }
    }
}
```

You can print a phrase on your screen using the Console.WriteLine() command. The phrase you would like to see should be put inside the round parenthesis.

NOTE: Console ≠ console ≠ CONSOLE

```
using System;

namespace Example {
    0 references
    class Program {
        0 references
        public static void Main(string[] args) {
            Console.WriteLine("Hello, World!");
        }
    }
}
```

Phrases that appear in quotation marks are called “strings”

All strings must start and end with double quotations

```
using System;

namespace Example {
    0 references
    class Program {
        0 references
        public static void Main(string[] args) {
            Console.WriteLine("Hello, World!");
        }
    }
}
```

This method is the entry point of all C# console applications

The execution of a program always starts from the first statement in the main method and ends when it finishes the last statement.

```
using System;

namespace Example {
    0 references
    class Program {
        0 references
        public static void Main(string[] args) {
            Console.WriteLine("Hello, World!");
        }
    }
}
```

What is a method? ???? ?

```
using System;

namespace Example {
    0 references
    class Program {
        0 references
        public static void Main(string[] args) {
            Console.WriteLine("Hello, World!");
        }
    }
}
```

Why is this method public static and void ???

(don’t worry about it for now lol)

```
using System;

namespace Example {
    0 references
    class Program {
        0 references
        public static void Main(string[] args) {
            Console.WriteLine("Hello, World!");
        }
    }
}
```

Tells the compiler that our program uses the System namespace

```
using System;

namespace Example {
    0 references
    class Program {
        0 references
        public static void Main(string[] args) {
            Console.WriteLine("Hello, World!");
        }
    }
}
```

What is a namespace ? ???? ?

```
namespace Demo {
    1 reference
    class Program {
        // do stuff
    }
    1 reference
    class Program {
        // do stuff
    }
}
```

```
namespace Demo1 {
    0 references
    class Program {
        // do stuff
    }
}

namespace Demo2 {
    0 references
    class Program {
        // do stuff
    }
}
```

using System;

Access to primitive types

Console Input/Output Operations

… and more!

```
using System;
```

EXAMPLES.

```
namespace Demo1 {
    0 references
    class Program {
        Console.WriteLine("Will this run?");
    }
}
```

```
namespace De
    0 references
    class Progra
        Console.W        Will this run?");
    }
}
```

```
namespace Demo2 {
    0 references
    class Program {
        0 references
        public static void Main(string[] args) {
            Console.WriteLine("What about this?");
        }
    }
}
```

```
namespace Demo2 {
    0 references
    class Program {
        0 references
        public static void Main(string[] args) {
            Console.WriteLine("What about this?");
        }
    }
}
```

```
Console.WriteLine("Will this work ???????");
```

```
Console.WriteLine("Wi      is work ???????");
```

Modern C# Hello World

Facebook Twitter LinkedIn

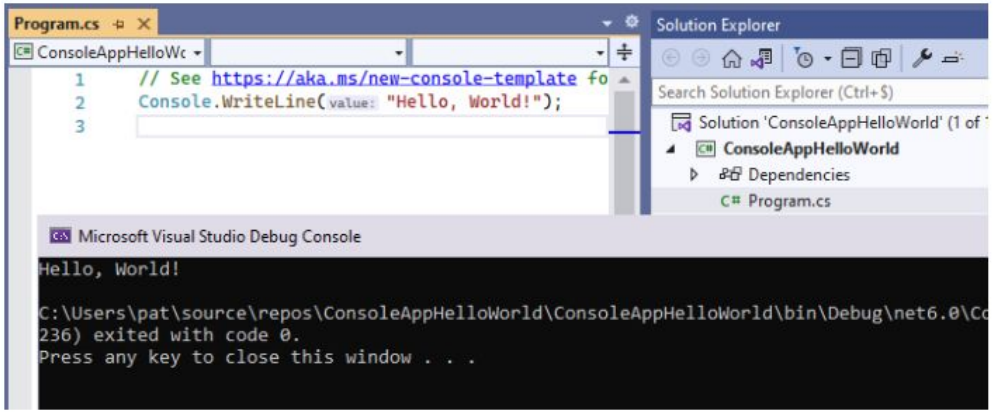
November 23, 2021 6 minutes read

```
1 Console.WriteLine("Modern C# Hello World!");
```

With Visual Studio 2022 when you create a new console project based on .NET 6, the *Hello World* source code generated is now as simple as that:

```
1 Console.WriteLine("Hello, World!");
```

Nice and concise isn't it? Here is what running this program looks like:



In Visual Studio 2019, the *Hello World* source code proposed when creating a new console project used to be much more verbose with the definition of a namespace, a class and a `Main()` method.

```
using System;

namespace Demo3 {
    0 references
    class Program {
        0 references
        public static void Main(string[] args) {
            Console.WriteLine("this?");
        }

}
```

```
using Syst

namespace Demo
    0 references
    class Program
        0 references
        public stat                (string[] args) {
            Conso                  is?");
        }

}
```

```
namespace Demo4 {
    0 references
    class Program {
        0 references
        public static void Main(string[] args) {
            Console.WriteLine("What");
            Console.WriteLine("about");
            Console.WriteLine("this");
            Console.WriteLine("time")
            Console.WriteLine("lol");
        }
    }
}
```

```
namespace D
    0 referen
    class P
        0 refere
        public s id ing[] args) {
            Consol hat");
            Console.W 'about");
            Console this");
            Conso e")
            Co iteLin
        }
    }
}
```

```
namespace Demo5 {
    0 references
    class Program {
        0 references
        public static void Main(string[] args) {
            Console.WriteLine("?");;;;;;;;;;;;
        }
    }
}
```

```
namespace Demo5 {
    0 references
    class Program {
        0 references
        public static v      ain(string[] args) {
            Console.Wr      ine("?");;;;;;;;;;;;
        }
    }
}
```

```
namespace Demo6 {
    1 reference
    class Program {
        0 references
        publicstaticvoidMain(string[] args) {
            Console.WriteLine("will this work?")
        }
    }
}
```

```
namespace D
    1 reference
    class Progr
        0 references
        publicstatic        tring[] args) {
            Console        "will this work?")
        }
    }
}
```

```
0 references
namespace Demo7 { class Program { public static void // wowie
0 references
Main(string[] args) { Console.WriteLine(":^)"); } } }
```

0 references

```
namespace Demo7 { class Prograu    ublic static void // wowie
```

0 references

```
Main(string[] args) { Conso    iteLine(":^)"); } } }
```

When do I press enter?

```
// in general, when writing code,
// you should go to a new line each time ...

namespace Demo8 {
    0 references
    class Program { // ... you type an open curly bracket
        0 references
        public static void Main(string[] args) {
            Console.WriteLine(":)"); // ... you type a ;
        }
    } // ... or you type a close curly bracket
}
```

What about when I indent?

```
// when we start a new "block" of instructions (e.g. open curly bracket)
// we indent (press tab)

// when we end the "block" of instructions (e.g. closed curly bracket)
// we un-indent the subsequent lines of code
namespace Demo9 {
    0 references
    class Program {
        0 references
        public static void Main(string[] args) {
            // I am the first statement of this block of code
            Console.WriteLine(">:)");
            // I am the last statement of this block of code
        }
    }
}
```

EXERCISES.

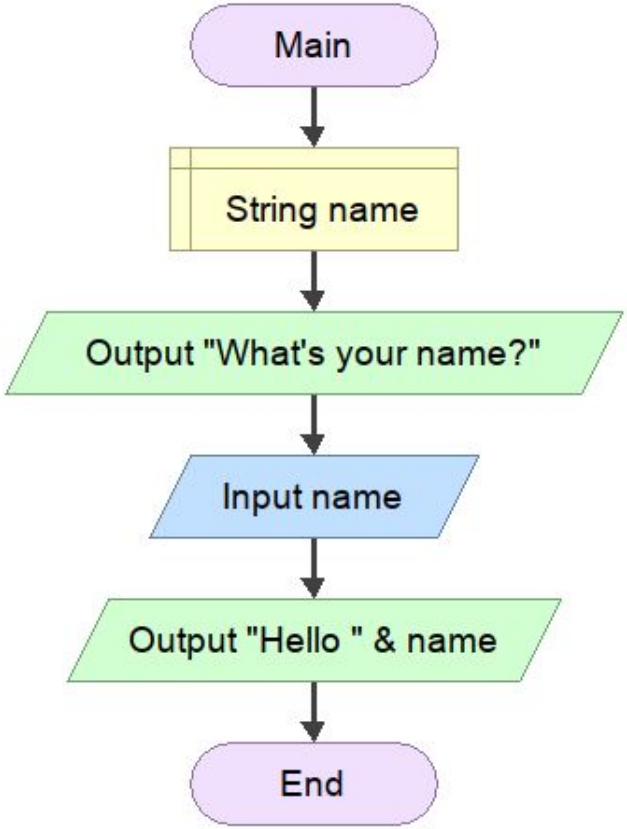
Exercise 1.

Create a Project that prints “Hello, World!”

Compress to a .zip file

Submit it on Lea

```
Console.WriteLine("Hello, World!");
```



Exercise 2.

Turn our exercise from last class into a C# Program

```
name = Console.ReadLine();
```