Introduction to C#

1

OUTLINE

- What is .NET?
- .NET compilation
- What is C#?
- Hello World in .NET 6 and .NET 5
- Hello World explained
- Namespaces
- Visual Studio
- Keyboard shortcuts

2

2

WHAT IS .NET?

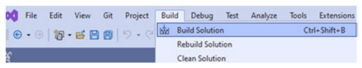
- .NET is a free, cross-platform, open source developer platform
- With .NET, you can build different applications such as web, mobile, desktop, games, and IoT
- You can write .NET apps in C#, F#, or Visual Basic
- Different .NET implementations target different operating systems:
  - .NET is a cross-platform implementation for websites, servers, and console apps on Windows, Linux, and macOS
  - .NET Framework supports websites, services, desktop apps, and more on the Windows operating system
  - Xamarin/Mono is an implementation for running apps on all major mobile operating systems
- .NET Standard is a base set of APIs (Application Programming Interfaces) that are common to all .NET implementations
- NuGet is a package manager that stores tens of thousands of packages that can extend the base functionality of .NET

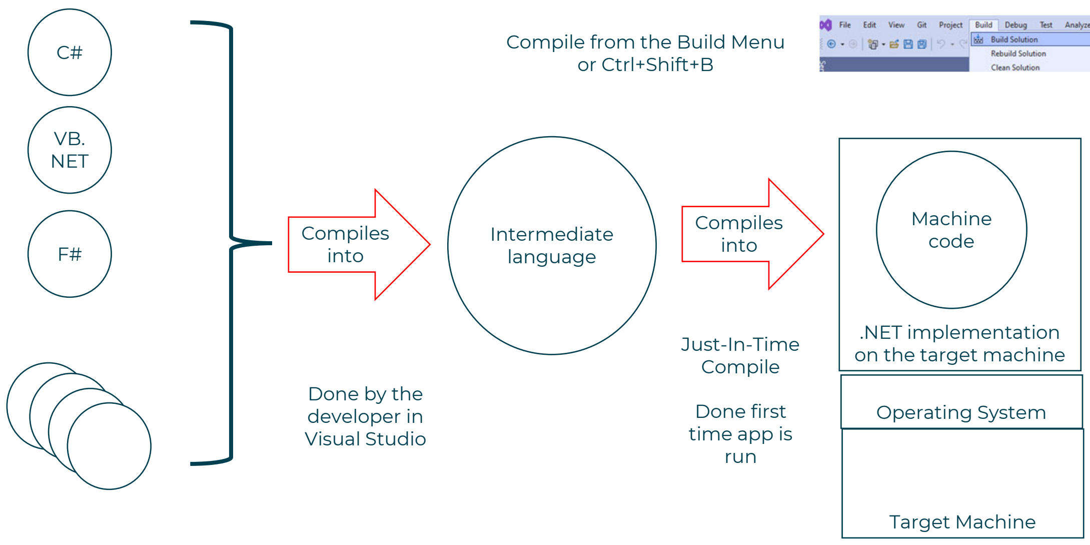
3

3

.NET Compilation

Compile from the Build Menu
or Ctrl+Shift+B





4

4

WHAT IS C#?

- C# is an object-oriented and type-safe programming language
- Type safety means C# declares data with a specific datatype and it checks either at compile time or runtime that the correct type of data is being used for the operations that are being performed
- C# is case sensitive

- NOTE: This course uses C# 10.0 and .NET 6

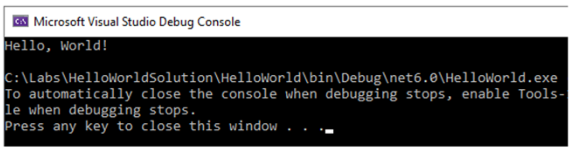
5

5

HELLO WORLD .NET 6

With .NET 6, a Console Application contains the following code only:

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



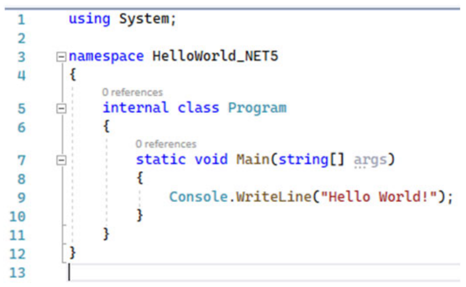
```
Hello, World!

C:\Labs\HelloWorldSolution\HelloWorld\bin\Debug\net6.0\HelloWorld.exe
To automatically close the console when debugging stops, enable Tools-
le when debugging stops.
Press any key to close this window . . .
```

6

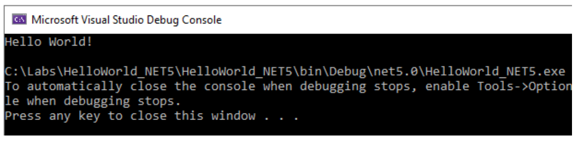
6

HELLO WORLD .NET 5

With .NET 5, a console application contains the following code:



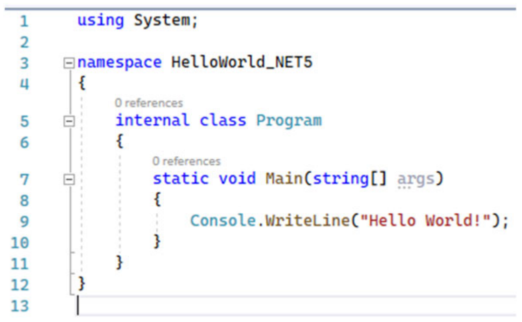
```
using System;

namespace HelloWorld_NET5
{
    0 references
    internal class Program
    {
        0 references
        static void Main(string[] args)
        {
            Console.WriteLine("Hello World!");
        }
    }
}
```



Microsoft Visual Studio Debug Console

```
Hello World!

C:\Labs\HelloWorld_NET5\HelloWorld_NET5\bin\Debug\net5.0\HelloWorld_NET5.exe
To automatically close the console when debugging stops, enable Tools->Option
le when debugging stops.
Press any key to close this window . . .
```

Hello World Explained



```
using System;

namespace HelloWorld_NET5
{
    0 references
    internal class Program
    {
        0 references
        static void Main(string[] args)
        {
            Console.WriteLine("Hello World!");
        }
    }
}
```

using System

Use the System library to access useful functions such as the Console class's WriteLine method, without having to use its fully-qualified name

namespace HelloWorld_NET5

A namespace is used to logically arrange items such as classes and control the scope of their names in large projects

internal class Program

A class defines a type of object

static void Main(string[] args)

Main is a method which is a code block that contains a series of statements

In C#, every executed instruction is performed in the context of a method

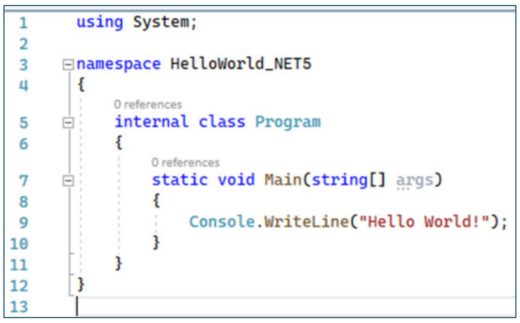
Main returns nothing (**void**) and accepts an array (collection) of text strings as input

Top-level Statements

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

When you use top-level statements such as the .NET 6 console application above, the .NET compiler synthesises a **Program** class with a **Main** method and places all your top level statements in that Main method.

The compiler effectively converts the above code to code equivalent to that on the right-hand side.



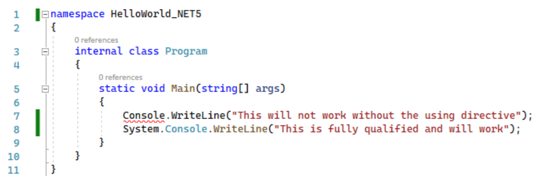
```
 using System;

namespace HelloWorld_NET5
{
    0 references
    internal class Program
    {
        0 references
        static void Main(string[] args)
        {
            Console.WriteLine("Hello World!");
        }
    }
}
```

9

9

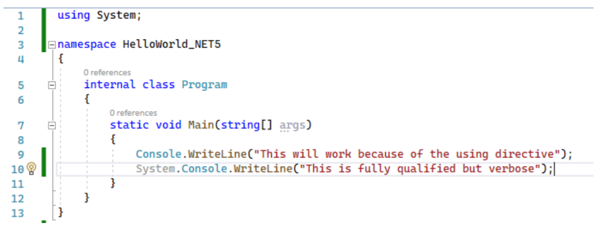
REFERRING TO NAMESPACES

Option A: Use the fully-qualified name



```
namespace HelloWorld_NET5
{
    0 references
    internal class Program
    {
        0 references
        static void Main(string[] args)
        {
            Console.WriteLine("This will not work without the using directive");
            System.Console.WriteLine("This is fully qualified and will work");
        }
    }
}
```

Option B: Issue a ‘using’ directive



```
using System;

namespace HelloWorld_NET5
{
    0 references
    internal class Program
    {
        0 references
        static void Main(string[] args)
        {
            Console.WriteLine("This will work because of the using directive");
            System.Console.WriteLine("This is fully qualified but verbose");
        }
    }
}
```

10

10

CREATING NAMESPACES

The intent of a Namespace is to uniquely identify the items it contains.

Below there are two classes, both called '*Car*'.

They can be disambiguated using their fully-qualified names of **Volkswagen.Car** and **Ferrari.Car**

```
namespace Volkswagen
{
    0 references
    public class Car
    {
    }
}
```

```
namespace Ferrari
{
    0 references
    public class Car
    {
    }
}
```

If only one class is required, you can issue a 'using' directive to import that namespace or use an alias.

```
using Volkswagen;
```

```
using v = Volkswagen;
```

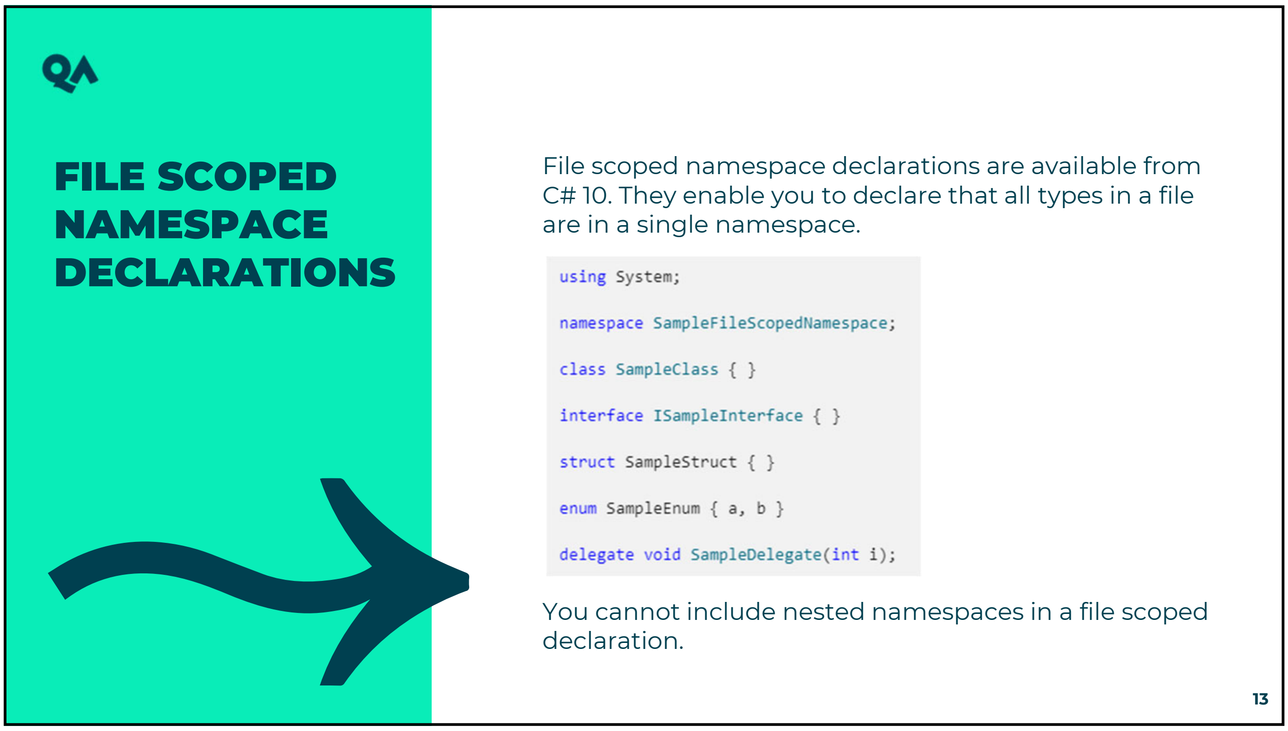
NAMESPACE ALIASES

Namespaces can be nested.

```
namespace PC
{
    namespace MyCompany
    {
        namespace Project
        {
            1 reference
            public class MyClass { }
        }
    }
}
```

Use an alias to shorten a long namespace.

```
using Project = PC.MyCompany.Project;
```



FILE SCOPED NAMESPACE DECLARATIONS

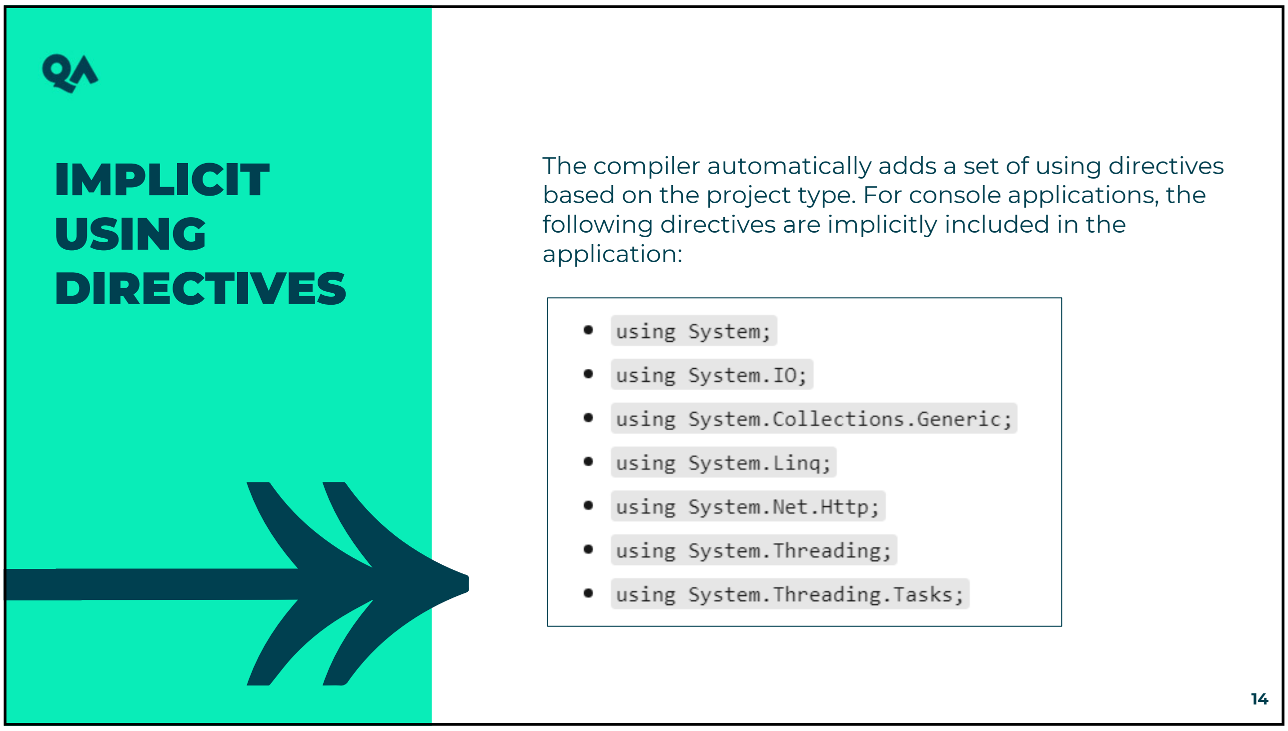
File scoped namespace declarations are available from C# 10. They enable you to declare that all types in a file are in a single namespace.

```
using System;

namespace SampleFileScopedNamespace;

class SampleClass { }

interface ISampleInterface { }

struct SampleStruct { }

enum SampleEnum { a, b }

delegate void SampleDelegate(int i);
```

You cannot include nested namespaces in a file scoped declaration.

13

13



IMPLICIT USING DIRECTIVES

The compiler automatically adds a set of using directives based on the project type. For console applications, the following directives are implicitly included in the application:

- `using System;`
- `using System.IO;`
- `using System.Collections.Generic;`
- `using System.Linq;`
- `using System.Net.Http;`
- `using System.Threading;`
- `using System.Threading.Tasks;`

14

14

GLOBAL USING DIRECTIVES

A global using directive imports a namespace for your whole application instead of a single file.

Either add a global using directive to the top of one of your code files:

```csharp
global using PC.MyCompany.Project;
```

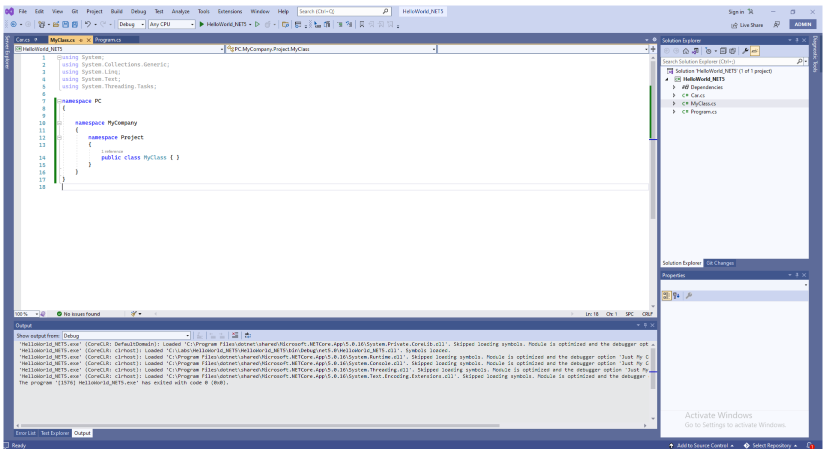
or, add a **<Using>** item to your project file:

```xml
<Project Sdk="Microsoft.NET.Sdk">

  <PropertyGroup>
    <OutputType>Exe</OutputType>
    <TargetFramework>net6.0</TargetFramework>
    <ImplicitUsings>enable</ImplicitUsings>
    <Nullable>enable</Nullable>
  </PropertyGroup>

    <ItemGroup>
        <Using Include="PC.MyCompany.Project" />
    </ItemGroup>

</Project>
```

15

15

Visual Studio



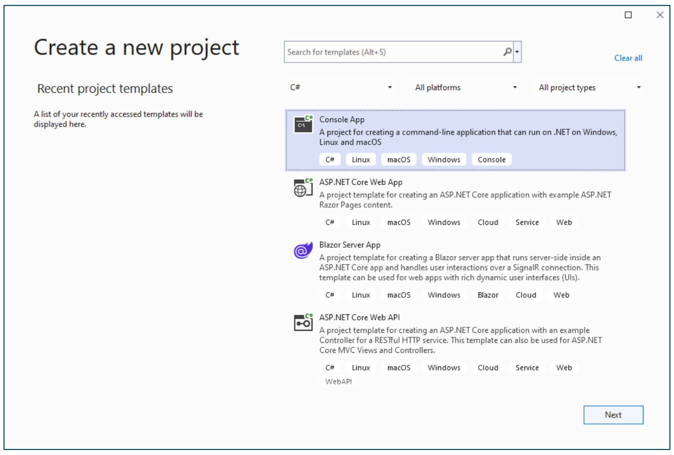
```csharp
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace PC
{

    namespace MyCompany
    {
        namespace Project
        {
            public class MyClass { }
        }
    }
}
```

```
'HelloWorld_NET5.exe' (CoreCLR: DefaultDomain): Loaded 'C:\Program Files\dotnet\shared\Microsoft.NETCore.App\5.0.16\System.Private.CoreLib.dll'. Skipped loading symbols. Module is optimized and the debugger opt
'HelloWorld_NET5.exe' (CoreCLR: clrhost): Loaded 'C:\Labs\HelloWorld_NET5\HelloWorld_NET5\bin\Debug\net5.0\HelloWorld_NET5.dll'. Symbols loaded.
'HelloWorld_NET5.exe' (CoreCLR: clrhost): Loaded 'C:\Program Files\dotnet\shared\Microsoft.NETCore.App\5.0.16\System.Runtime.dll'. Skipped loading symbols. Module is optimized and the debugger option 'Just My C
'HelloWorld_NET5.exe' (CoreCLR: clrhost): Loaded 'C:\Program Files\dotnet\shared\Microsoft.NETCore.App\5.0.16\System.Console.dll'. Skipped loading symbols. Module is optimized and the debugger option 'Just My C
'HelloWorld_NET5.exe' (CoreCLR: clrhost): Loaded 'C:\Program Files\dotnet\shared\Microsoft.NETCore.App\5.0.16\System.Threading.dll'. Skipped loading symbols. Module is optimized and the debugger option 'Just My
'HelloWorld_NET5.exe' (CoreCLR: clrhost): Loaded 'C:\Program Files\dotnet\shared\Microsoft.NETCore.App\5.0.16\System.Text.Encoding.Extensions.dll'. Skipped loading symbols. Module is optimized and the debugger
The program '[1576] HelloWorld_NET5.exe' has exited with code 0 (0x0).
```

16

16

Visual Studio Project Templates


Create a new project

Search for templates (Alt+S)

Clear all

Recent project templates

A list of your recently accessed templates will be displayed here.

C# | All platforms | All project types

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

ASP.NET Core Web App
A project template for creating an ASP.NET Core application with example ASP.NET Razor Pages content.
C# Linux macOS Windows Cloud Service Web

Blazor Server App
A project template for creating a Blazor server app that runs server-side inside an ASP.NET Core app and handles user interactions over a SignalR connection. This template can be used for web apps with rich dynamic user interfaces (UIs).
C# Linux macOS Windows Blazor Cloud Web

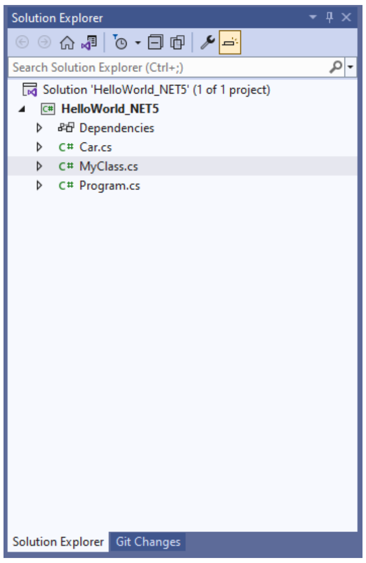
ASP.NET Core Web API
A project template for creating an ASP.NET Core application with an example Controller for a RESTful HTTP service. This template can also be used for ASP.NET Core MVC Views and Controllers.
C# Linux macOS Windows Cloud Service Web WebAPI

Next

17

17

Visual Studio: Solution Explorer


Solution Explorer

Search Solution Explorer (Ctrl+;)

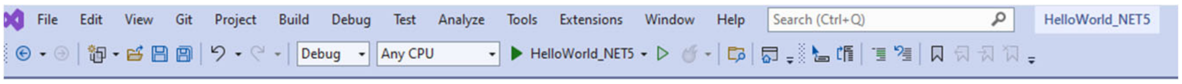
```
Solution 'HelloWorld_NET5' (1 of 1 project)
◢ HelloWorld_NET5
    ▷ Dependencies
    ▷ Car.cs
    ▷ MyClass.cs
    ▷ Program.cs
```

Solution Explorer | Git Changes

18

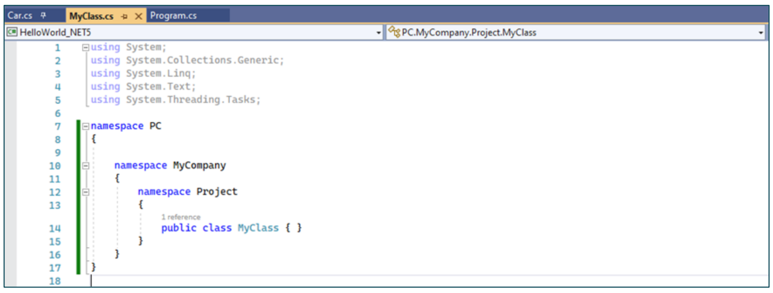
18

Visual Studio: Menu and Toolbar



19

Visual Studio: Code Editors

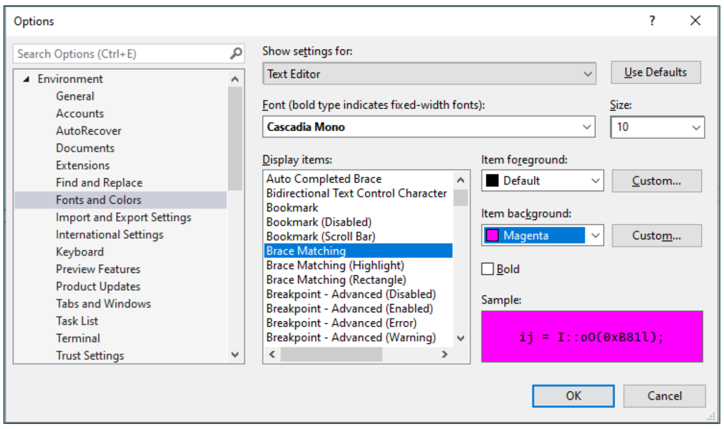


```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace PC
{

    namespace MyCompany
    {
        namespace Project
        {
            public class MyClass { }
        }
    }
}
```

20

Brace Matching

Tools -> Options -> Fonts and Colors



Options

Search Options (Ctrl+E)

- Environment
  - General
  - Accounts
  - AutoRecover
  - Documents
  - Extensions
  - Find and Replace
  - Fonts and Colors
  - Import and Export Settings
  - International Settings
  - Keyboard
  - Preview Features
  - Product Updates
  - Tabs and Windows
  - Task List
  - Terminal
  - Trust Settings

Show settings for: Text Editor

Font (bold type indicates fixed-width fonts): Cascadia Mono

Size: 10

Display items:
- Auto Completed Brace
- Bidirectional Text Control Character
- Bookmark
- Bookmark (Disabled)
- Bookmark (Scroll Bar)
- Brace Matching
- Brace Matching (Highlight)
- Brace Matching (Rectangle)
- Breakpoint - Advanced (Disabled)
- Breakpoint - Advanced (Enabled)
- Breakpoint - Advanced (Error)
- Breakpoint - Advanced (Warning)

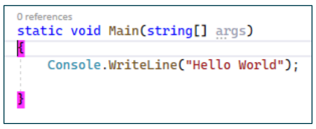
Item foreground: Default

Item background: Magenta

☐ Bold

Sample:

```
ij = I::oO(0xB81l);
```

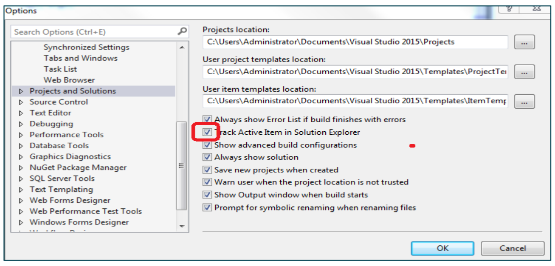


```
0 references
static void Main(string[] args)
{
    Console.WriteLine("Hello World");
}
```

21

Tracking and Line Numbers

Tools -> Options -> Projects and Solutions



Options

Search Options (Ctrl+E)

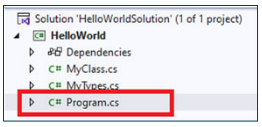
- Synchronized Settings
- Tabs and Windows
- Task List
- Web Browser
- ▷ Projects and Solutions
- ▷ Source Control
- ▷ Text Editor
- ▷ Debugging
- ▷ Performance Tools
- ▷ Database Tools
- ▷ Graphics Diagnostics
- ▷ NuGet Package Manager
- ▷ SQL Server Tools
- ▷ Text Templating
- ▷ Web Forms Designer
- ▷ Web Performance Test Tools
- ▷ Windows Forms Designer

Projects location: C:\Users\Administrator\Documents\Visual Studio 2015\Projects

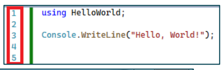
User project templates location: C:\Users\Administrator\Documents\Visual Studio 2015\Templates\ProjectTe

User item templates location: C:\Users\Administrator\Documents\Visual Studio 2015\Templates\ItemTemp

- ✓ Always show Error List if build finishes with errors
- ✓ Track Active Item in Solution Explorer
- ✓ Show advanced build configurations
- ✓ Always show solution
- ✓ Save new projects when created
- ✓ Warn user when the project location is not trusted
- ✓ Show Output window when build starts
- ✓ Prompt for symbolic renaming when renaming files

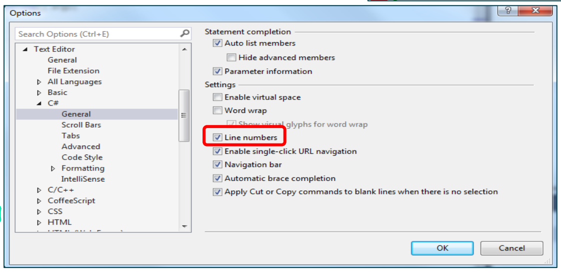


Solution 'HelloWorldSolution' (1 of 1 project)
- HelloWorld
  - Dependencies
  - MyClass.cs
  - MyTypes.cs
  - Program.cs



```
1  using HelloWorld;
2
3  Console.WriteLine("Hello, World!");
4
5
```

Tools -> Options -> Text Editor -> C#



Options

Search Options (Ctrl+E)

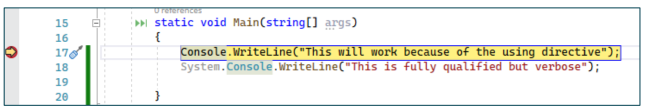
- Text Editor
  - General
  - File Extension
  - ▷ All Languages
  - ▷ Basic
  - C#
    - General
    - Scroll Bars
    - Tabs
    - Advanced
    - Code Style
    - ▷ Formatting
    - IntelliSense
  - ▷ C/C++
  - ▷ CoffeeScript
  - ▷ CSS
  - ▷ HTML

Statement completion
- ✓ Auto list members
  - ☐ Hide advanced members
- ✓ Parameter information

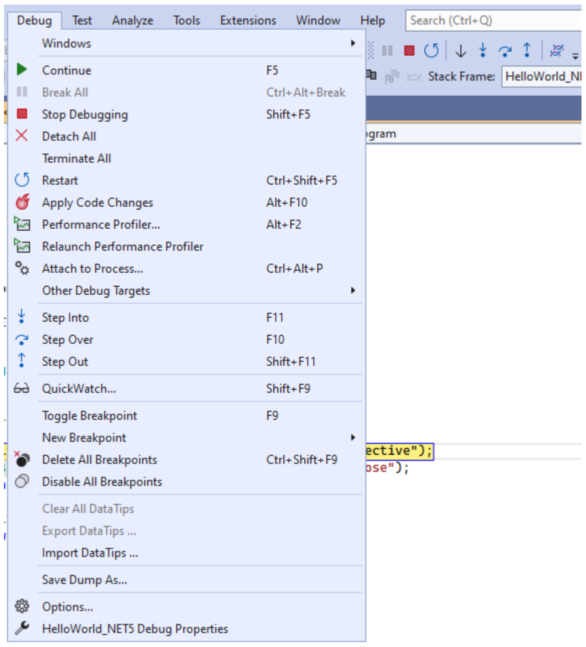
Settings
- ☐ Enable virtual space
- ☐ Word wrap
  - ✓ Show visual glyphs for word wrap
- ✓ Line numbers
- ✓ Enable single-click URL navigation
- ✓ Navigation bar
- ✓ Automatic brace completion
- ✓ Apply Cut or Copy commands to blank lines when there is no selection

22

Debugging



```
0 references
static void Main(string[] args)
{
    Console.WriteLine("This will work because of the using directive");
    System.Console.WriteLine("This is fully qualified but verbose");

}
```

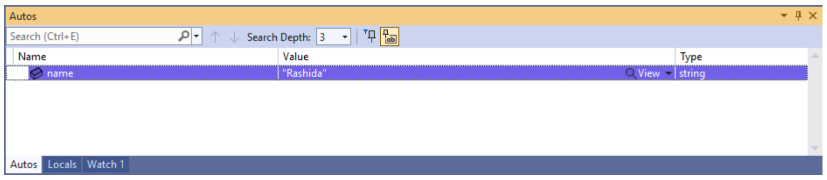


Windows	
Continue	F5
Break All	Ctrl+Alt+Break
Stop Debugging	Shift+F5
Detach All	
Terminate All	
Restart	Ctrl+Shift+F5
Apply Code Changes	Alt+F10
Performance Profiler...	Alt+F2
Relaunch Performance Profiler	
Attach to Process...	Ctrl+Alt+P
Other Debug Targets	
Step Into	F11
Step Over	F10
Step Out	Shift+F11
QuickWatch...	Shift+F9
Toggle Breakpoint	F9
New Breakpoint	
Delete All Breakpoints	Ctrl+Shift+F9
Disable All Breakpoints	
Clear All DataTips	
Export DataTips ...	
Import DataTips ...	
Save Dump As...	
Options...	
HelloWorld_NET5 Debug Properties	

23

23

Debugging Windows

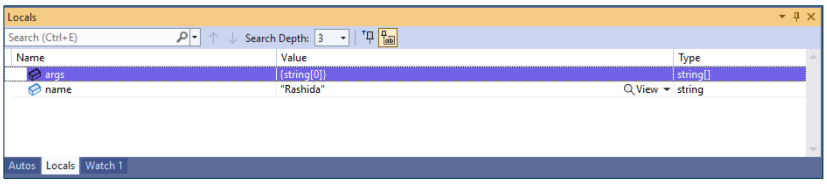


Autos

Search (Ctrl+E) Search Depth: 3

Name	Value	Type
name	"Rashida"	string

Autos Locals Watch 1

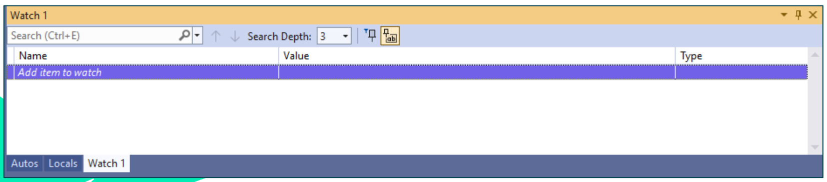


Locals

Search (Ctrl+E) Search Depth: 3

Name	Value	Type
args	{string[0]}	string[]
name	"Rashida"	string

Autos Locals Watch 1



Watch 1

Search (Ctrl+E) Search Depth: 3

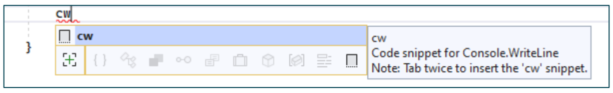
Name	Value	Type
Add item to watch		

Autos Locals Watch 1

24

24

IntelliSense

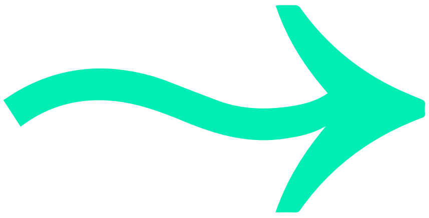


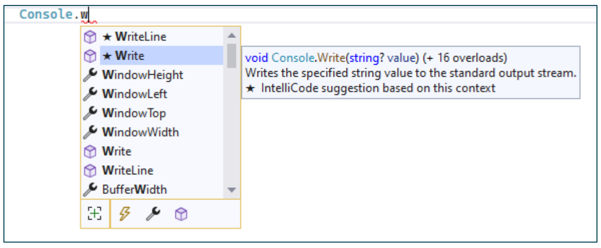
```
cw
}
```

cw

cw
Code snippet for Console.WriteLine
Note: Tab twice to insert the 'cw' snippet.

Code snippets





```
Console.w
```

★ WriteLine
★ Write
WindowHeight
WindowLeft
WindowTop
WindowWidth
Write
WriteLine
BufferWidth

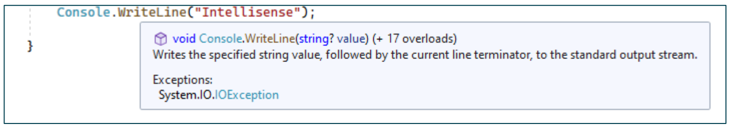
void Console.Write(string? value) (+ 16 overloads)
Writes the specified string value to the standard output stream.
★ IntelliCode suggestion based on this context

List members

25

25

IntelliSense



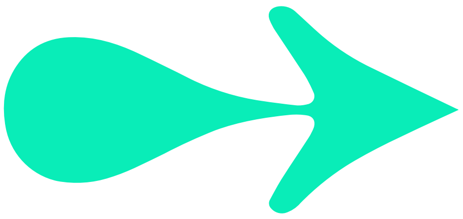
```
Console.WriteLine("Intellisense");
}
```

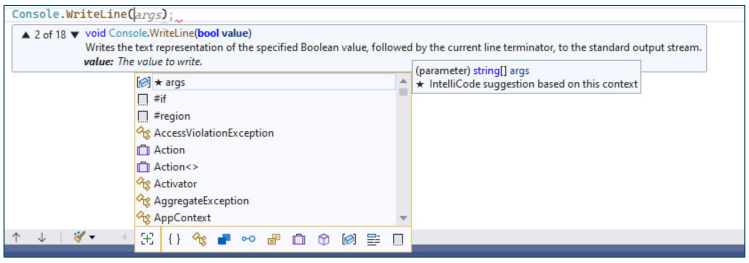
void Console.WriteLine(string? value) (+ 17 overloads)
Writes the specified string value, followed by the current line terminator, to the standard output stream.

Exceptions:
System.IO.IOException

Quick info

Parameter info





```
Console.WriteLine(args);
```

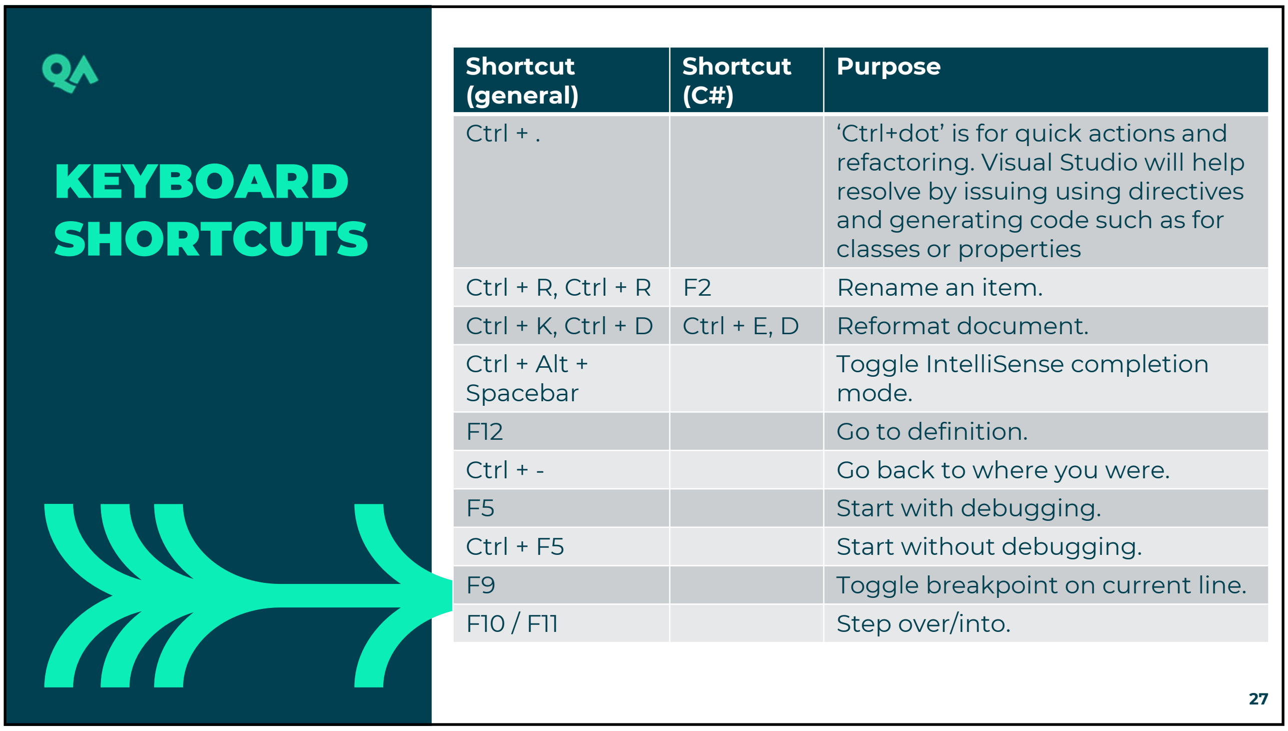
▲ 2 of 18 ▼ void Console.WriteLine(bool value)
Writes the text representation of the specified Boolean value, followed by the current line terminator, to the standard output stream.
value: *The value to write.*

(parameter) string[] args
★ IntelliCode suggestion based on this context

★ args
#if
#region
AccessViolationException
Action
Action<>
Activator
AggregateException
AppContext

26

26

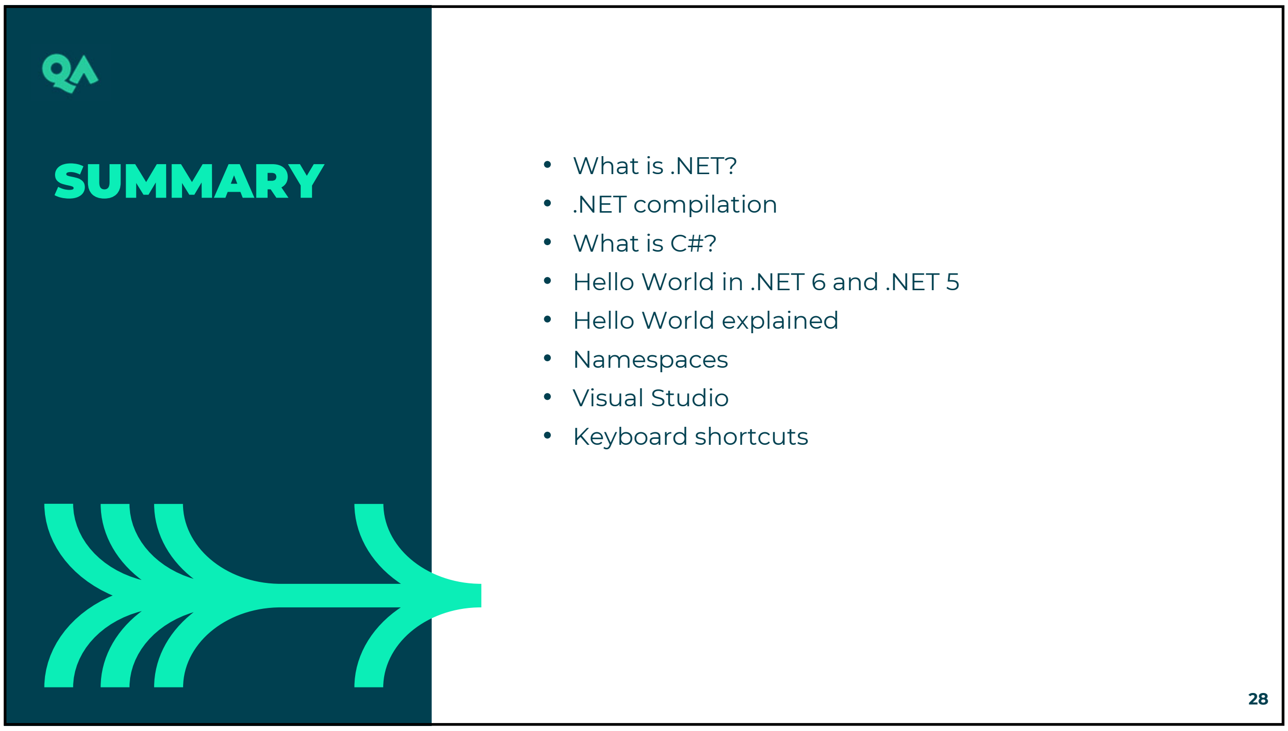


KEYBOARD SHORTCUTS

Shortcut (general)	Shortcut (C#)	Purpose
Ctrl + .		'Ctrl+dot' is for quick actions and refactoring. Visual Studio will help resolve by issuing using directives and generating code such as for classes or properties
Ctrl + R, Ctrl + R	F2	Rename an item.
Ctrl + K, Ctrl + D	Ctrl + E, D	Reformat document.
Ctrl + Alt + Spacebar		Toggle IntelliSense completion mode.
F12		Go to definition.
Ctrl + -		Go back to where you were.
F5		Start with debugging.
Ctrl + F5		Start without debugging.
F9		Toggle breakpoint on current line.
F10 / F11		Step over/into.

27

27



SUMMARY

- What is .NET?
- .NET compilation
- What is C#?
- Hello World in .NET 6 and .NET 5
- Hello World explained
- Namespaces
- Visual Studio
- Keyboard shortcuts

28

28

ACTIVITY:
Exercise 2

29