TOOL DEVELOPMENT

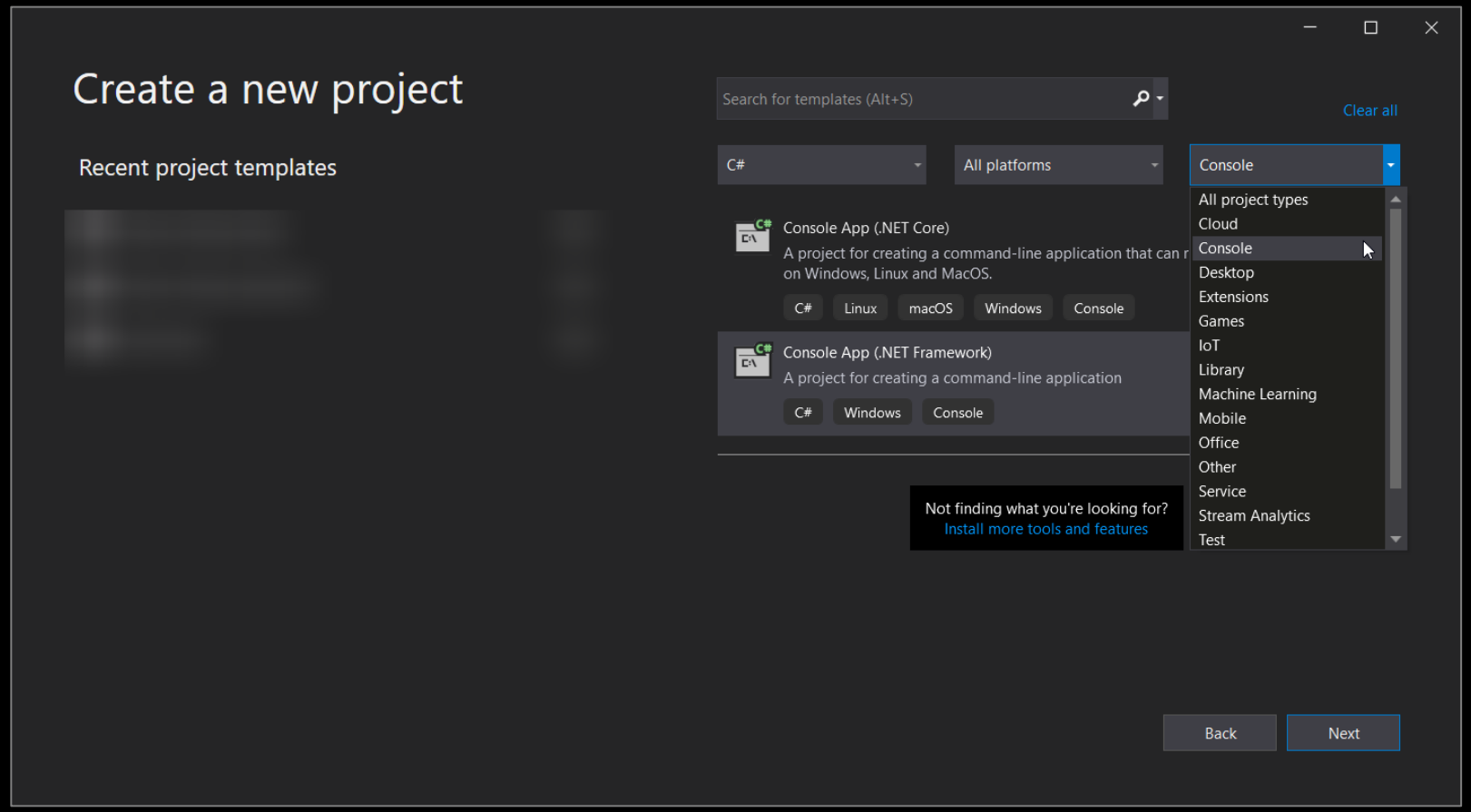
C# INTRODUCTION

C# - ‘HELLO WORLD’

FIRST PROJECT SETUP + EXPLORATION

PROJECT CREATION

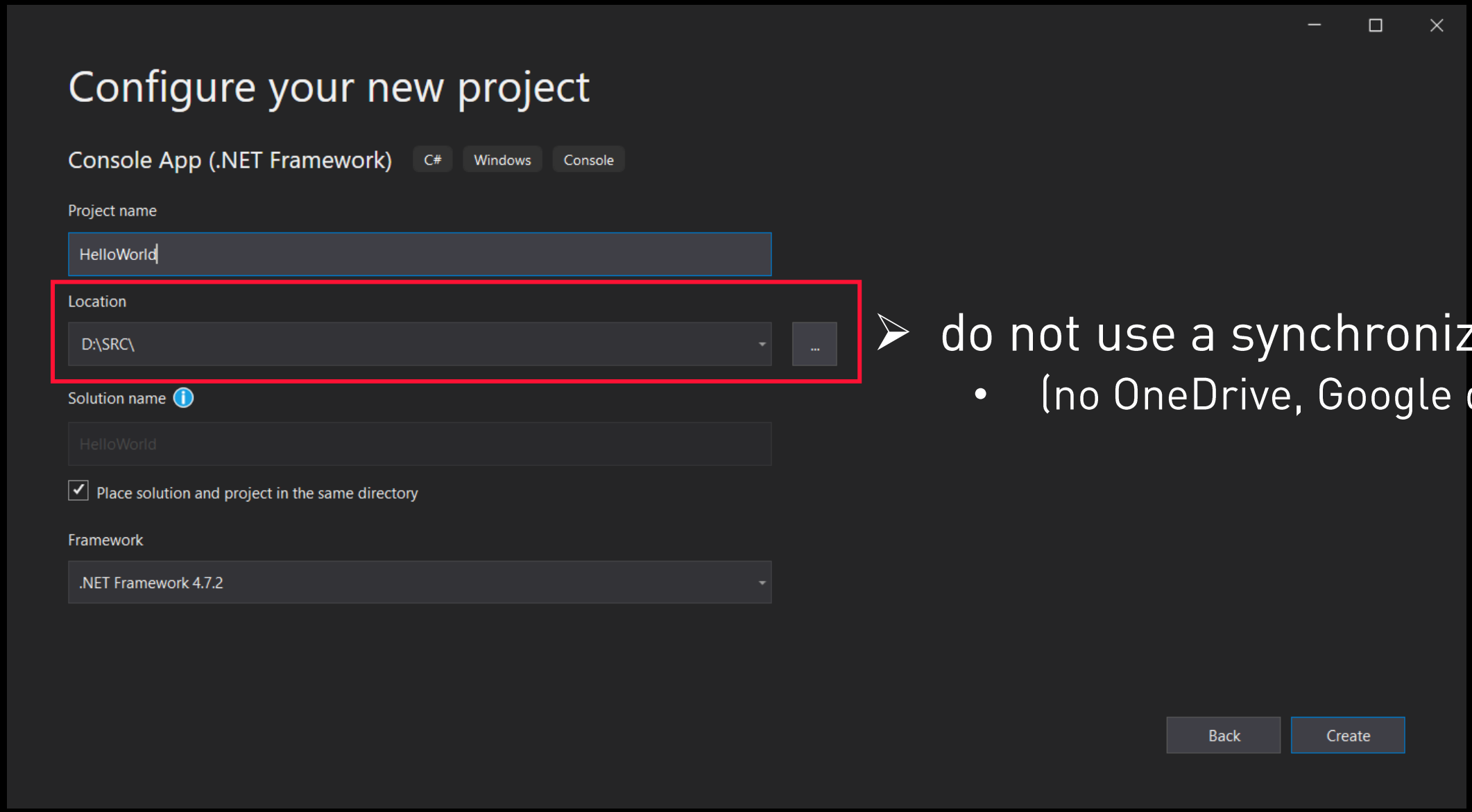
- Project Creation
  - Visual Studio → File → New → Project…



- Different C# Project Types
  - WPF Application (UI Project)
  - Console Application (Console)
  - Class Library (Library/DLL)
  - …
- For now: Console Application

first c# project

PROJECT CONFIGURATION

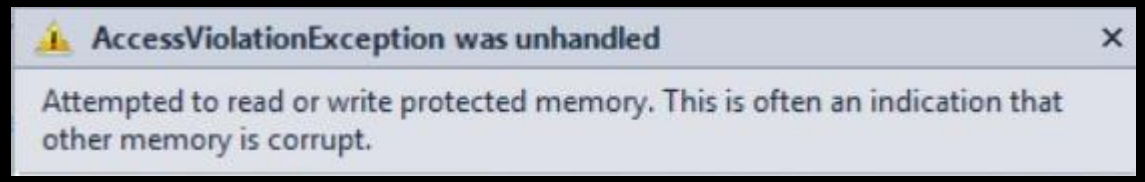


- do not use a synchronized folder
  - (no OneDrive, Google drive,...)

first c# project

NO SYNCHRONIZED FOLDER??

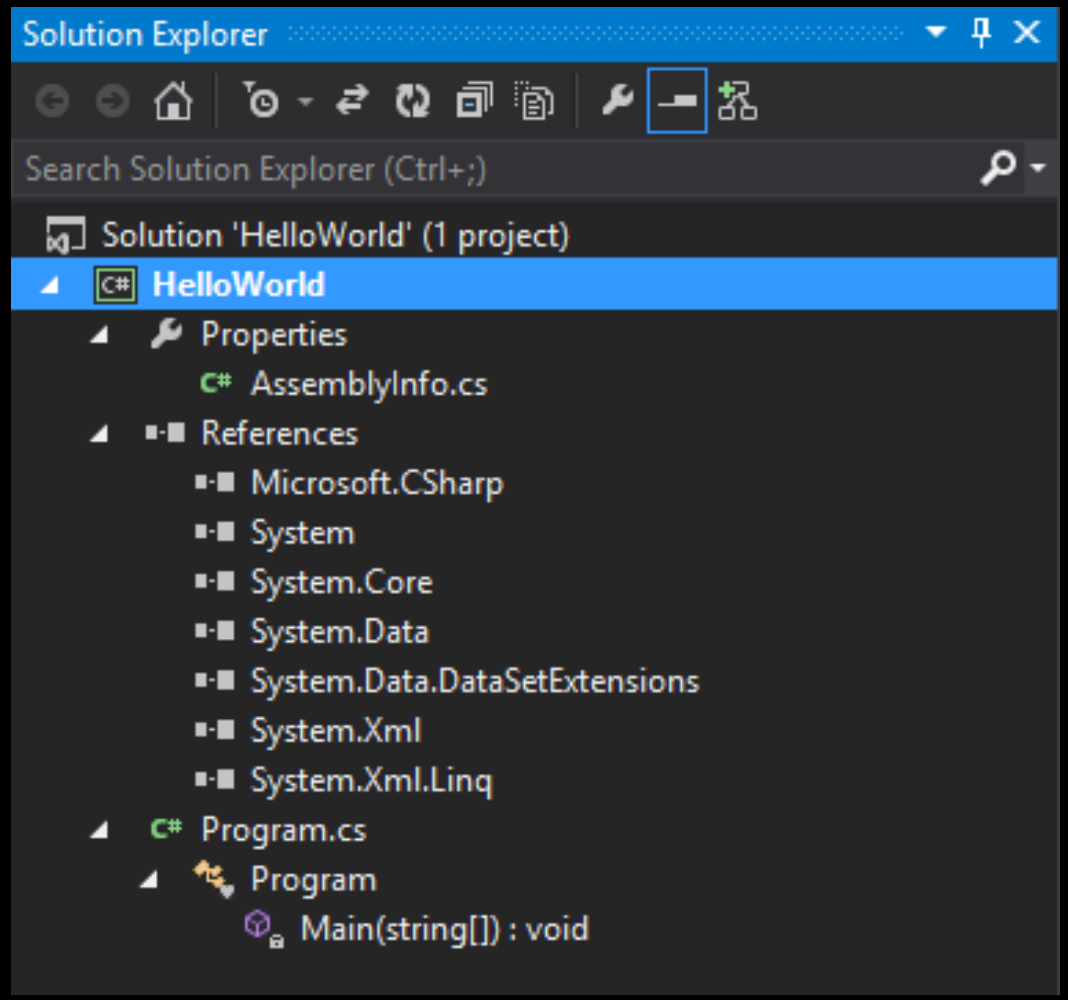
- syncing sometimes freezes files



- extra issues when sharing code
- solution:

first c# project

SOLUTION EXPLORER



- Properties
- References
- Code Files

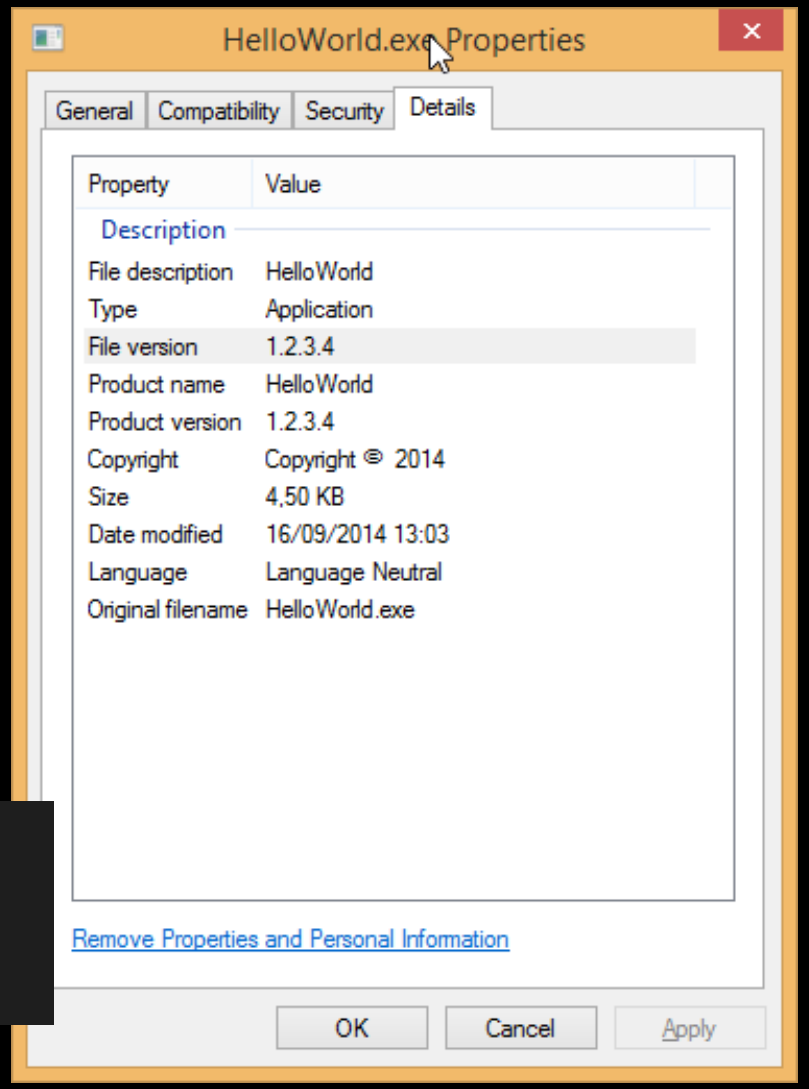
PROJECT PROPERTIES

EXPLORE

PROPERTIES

- AssemblyInfo.cs [cs → Source Code File for C#]
  - Extra information about the assembly (application/library)
    - Title
    - Description
    - Copyright
    - Version
    - ...
  - 'Baked' into the application/library

```
// [assembly: AssemblyVersion("1.0.*")]
[assembly: AssemblyVersion("1.0.0.0")]
[assembly: AssemblyFileVersion("1.2.3.4")]
```



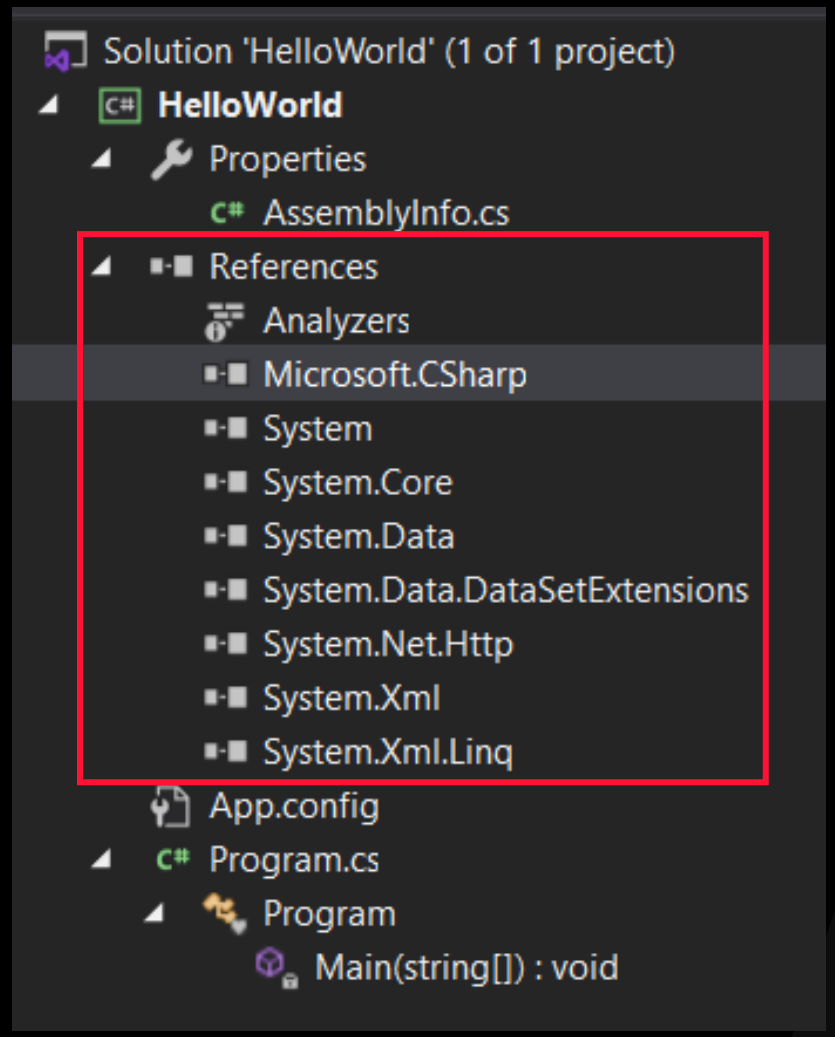
REFERENCES

WORKING WITH LIBRARIES

PROJECT REFERENCES

- ✓ Default references
- ✓ Possibility to add references

- ➢ references to libraries

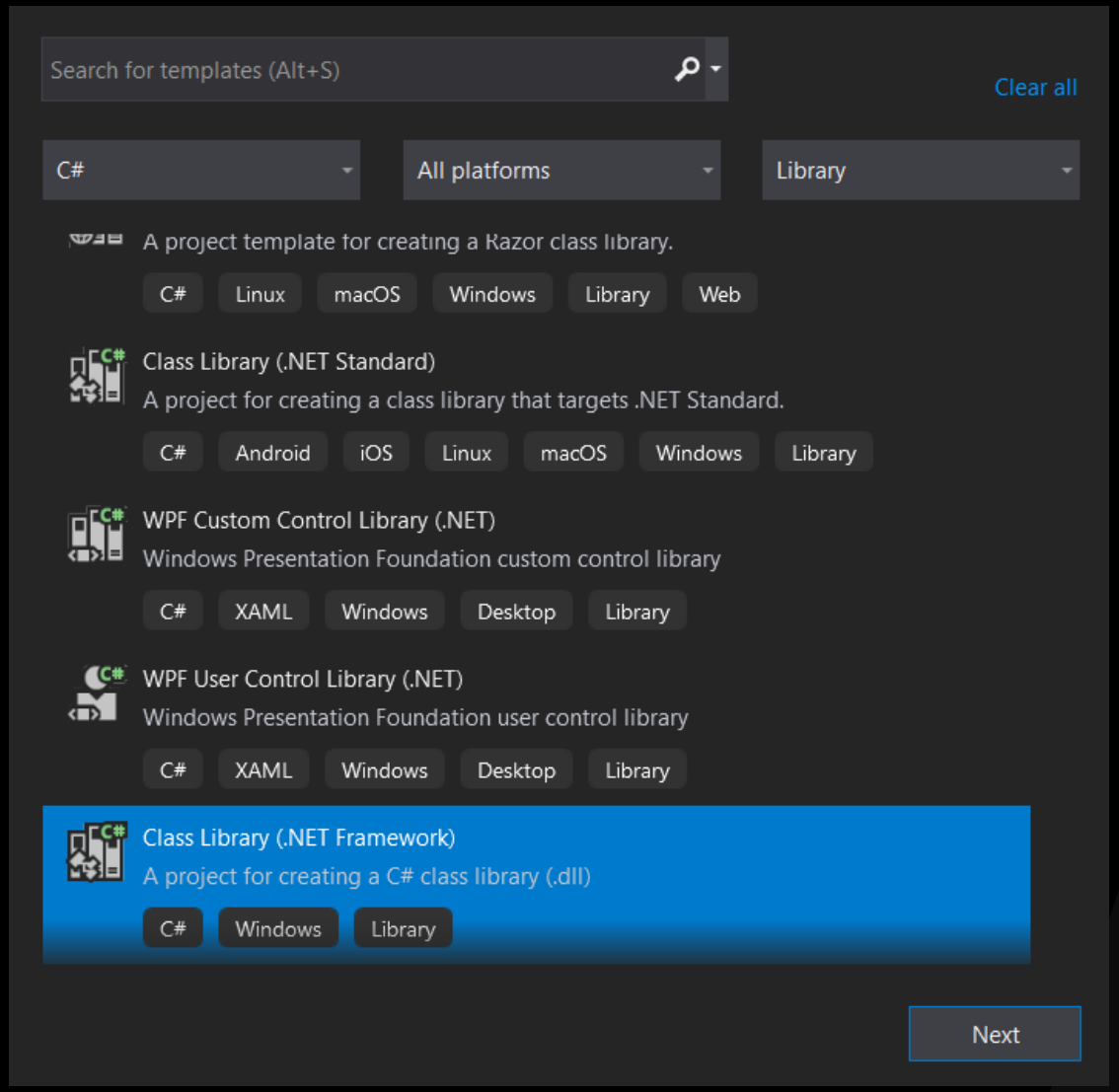


LIBRARIES

- Managed DLL
- 'Reusable' objects and functionality packed into an assembly (= library)

- .NET Framework = Huge set of libraries
  - System.Xml → Contains XML functionality
  - System.Net → Contains Networking functionality
  - …

- Third Party Libraries
  (Don't try to write everything yourself ☺)
  - SharpDX (Managed DirectX)
  - Json.NET (Managed Json Parser)
  - …



REFERENCES

- Access / use in code:
  1. add the reference
  2. with the using keyword, reference a namespace

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

namespace HelloWorld
{
    0 references
    class Program
    {
        0 references
        static void Main(string[] args)
        {
        }
    }
}
```

NAMESPACES

- Recognized by the 'using' directive

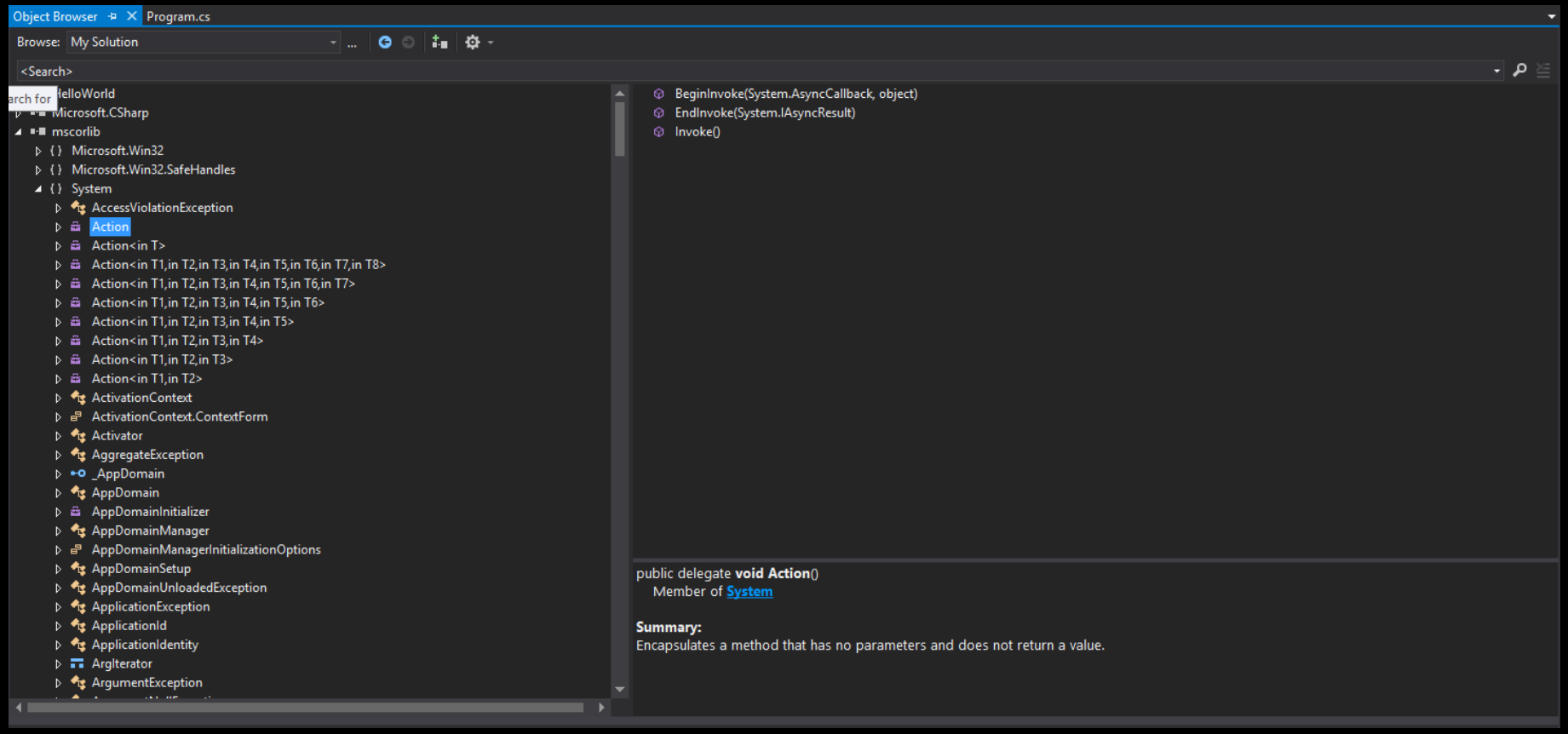
- using System;
  - Gives you access to the exposed objects of the System library

- namespace HelloWorld
  - Namespace identifier for your own objects
  - (ex. When writing your own library)

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

namespace HelloWorld
{
    0 references
    class Program
    {
        0 references
        static void Main(string[] args)
        {
        }
    }
}
```

EXPLORING A LIBRARY

- Object Browser
  - Useful tool of Visual Studio to browse the content of a library
  - (Right-Click on Reference → View in Object Browser)



WHAT DOES A LIBRARY CONTAIN?

- .NET/C# Object/Directives/Statements Documentation
  - Select an object in your code, press F1

Microsoft | Docs Documentation

Microsoft | Docs Documentation

Sign in

.NET Languages Workloads APIs Resources

Download .NET

Docs / .NET / .NET API browser / System.Collections.Generic

Bookmark Edit

Version

.NET Framework 4.7.2

Search

System.Collections.Generic

- Comparer<T>
- Dictionary<TKey,TValue>.Enumerator
- Dictionary<TKey,TValue>.KeyCollection.Enumerator
- Dictionary<TKey,TValue>.KeyCollection
- Dictionary<TKey,TValue>.ValueCollection.Enumerator
- Dictionary<TKey,TValue>.ValueCollection
- Dictionary<TKey,TValue>
- EqualityComparer<T>
- HashSet<T>.Enumerator
- HashSet<T>
- ICollection<T>

System.Collections.Generic Namespace

Contains interfaces and classes that define generic collections, which allow users to create strongly typed collections that provide better type safety and performance than non-generic strongly typed collections.

Classes

Comparer<T>	Provides a base class for implementations of the IComparer<T> generic interface.
Dictionary<TKey,TValue>.KeyCollection	Represents the collection of keys in a Dictionary<TKey,TValue>. This class cannot be inherited.
Dictionary<TKey,TValue>.ValueCollection	Represents the collection of values in a Dictionary<TKey,TValue>. This class cannot be inherited.
Dictionary<TKey,TValue>	Represents a collection of keys and values.
EqualityComparer<T>	Provides a base class for implementations of the IEqualityComparer<T> generic interface.
HashSet<T>	Represents a set of values.

Is this page helpful?

Yes No

In this article

- Classes
- Structs
- Interfaces
- Remarks
- See also

ww.microsoft.com

FLASH QUIZ

NAMESPACES

FLASH QUIZ

Which of the following namespaces does **not** exist in the .net framework class library?

A. System.Reflection
B. System.Threading
C. System.Net
D. System.Error
E. System.Dynamic

FLASH QUIZ

Which of the following namespaces does **not** exist in the .net framework class library?

A. System.Reflection
B. System.Threading
C. System.Net
D. System.Error
E. System.Dynamic

CODE FILES

.CS FILES

CODE FILES

- A project can contain multiple code files (.cs files)

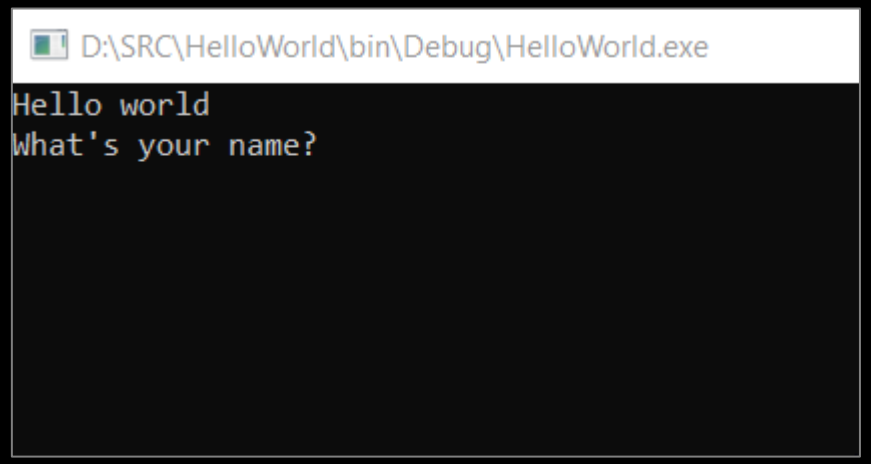
- Console Application contains one code file by default
  - Program.cs
    - Static void Main(string[] args) → Entry Point

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

namespace HelloWorld
{
    0 references
    class Program
    {
        0 references
        static void Main(string[] args)
        {

        }
    }
}
```

EXERCISE: Hello World Console Application

- Find out how to write to the 'Console'
  - There is an Object inside the System namespace specially to interact with the 'Console'
    - Find it in the documentation or object browser
    - Inspect its methods and find out how to:
      - Write a line to the console
      - Read a line from the console

- Program Execution Order:
  - Write 'Hello World!'
  - Read a line (this prevents the application from exiting)



LOCAL VARIABLES

- Explicitly typed:
  - Convention: lowerCamelCase
  -
```
string message = "hello world";
Hero myHero = new Hero();
```

- Implicitly typed:
  - Convention: lowerCamelCase
  - **must** be initialized (compiler figures out type)

```
var msg = "hello world";
var yourHero = new Hero();
```

(local variable) string msg

(local variable) Hero yourHero