Setting up your development environment

Integrated Development Environment

- Visual Studio 2022 for Windows
- Visual Studio 2022 for Mac
- Visual Studio Code for Windows, Mac, or Linux

.NET Framework

You can see which SDKs and runtimes are currently installed using the following commands:

- `dotnet --list-sdks`
- `dotnet --list-runtimes`

Building Blocks of .NET Platform

DotNet Framework

- .NET can be understood as a runtime environment and a comprehensive

CLR

- dotnet new --list **list all project templates**
- dotnet new sln
- dotnet new console --framework net6.0
- dotnet new classlib --framework net6.0
- dotnet build
- dotnet run
- dotnet sln add <path of csproj file>
- dotnet add <path to the main project> reference <path to referenced library>

Base Class Library

Defining Custom Namespaces

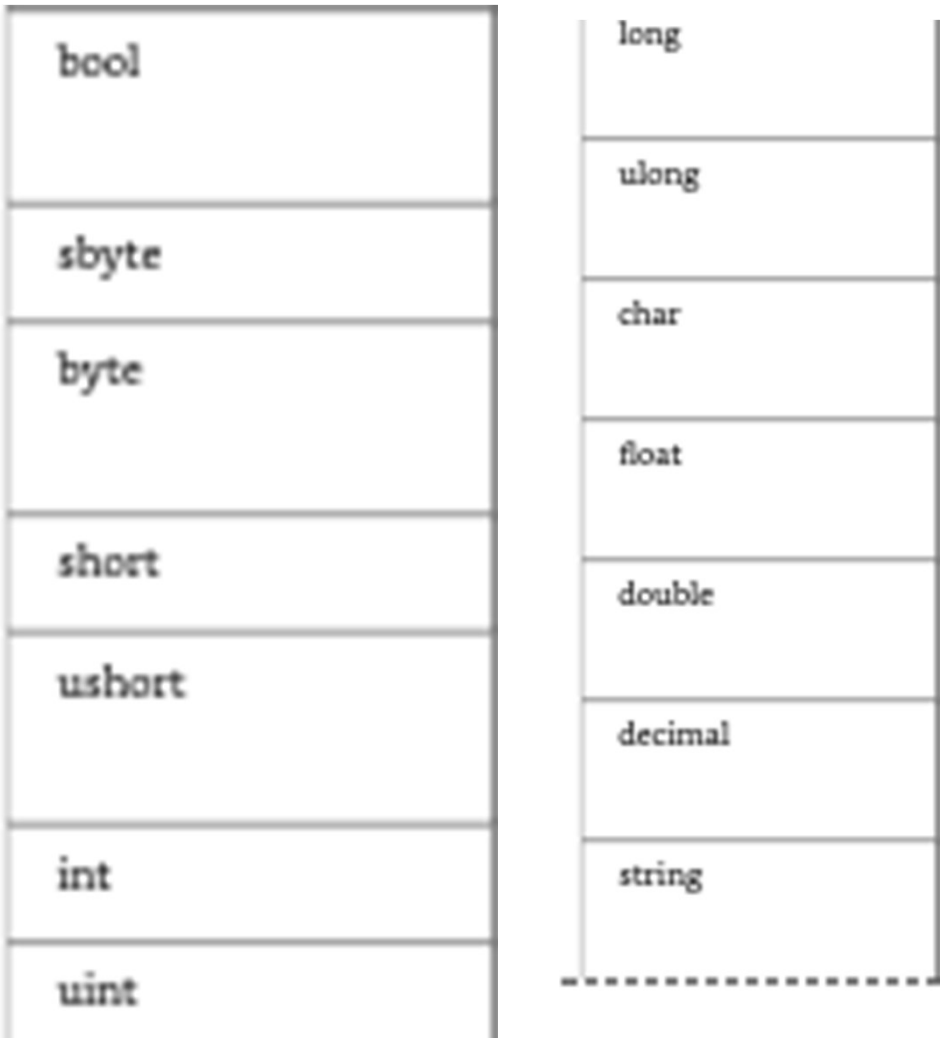
- when you build larger applications with many types, it can be helpful to group your related types into custom namespaces.
- In C#, this is accomplished using the namespace keyword. **
- Name spaces can be nested. Inside one namespace can come another namespace

using / fully qualified type name

- Types whenever defined can be referred fully qualified or
- Add using *name-space-name* statement at the top of the file where we are using it.

C# Programming Language

Data Types


bool
sbyte
byte
short
ushort
int
uint
long
ulong
char
float
double
decimal
string

C# type/keyword	Range	Size
`sbyte`	-128 to 127	Signed 8-bit integer
`byte`	0 to 255	Unsigned 8-bit integer
`short`	-32,768 to 32,767	Signed 16-bit integer
`ushort`	0 to 65,535	Unsigned 16-bit integer
`int`	-2,147,483,648 to 2,147,483,647	Signed 32-bit integer
`uint`	0 to 4,294,967,295	Unsigned 32-bit integer
`long`	-9,223,372,036,854,775,808 to 9,223,372,036,854,775,807	Signed 64-bit integer
`ulong`	0 to 18,446,744,073,709,551,615	Unsigned 64-bit integer

C# type/keyword	Approximate range	Precision	Size
`float`	$\pm 1.5 \times 10^{-45}$ to $\pm 3.4 \times 10^{38}$	~6-9 digits	4 bytes
`double`	$\pm 5.0 \times 10^{-324}$ to $\pm 1.7 \times 10^{308}$	~15-17 digits	8 bytes
`decimal`	$\pm 1.0 \times 10^{-28}$ to $\pm 7.9228 \times 10^{28}$	28-29 digits	16 bytes

The default Literal

- The default literal assigns a variable the default value for its data type

```
int myInt = default;
```

Parsing Values from String Data

- Parse() method
- TryParse method

Methods

Optional Parameters

- Parameters of a method can be converted as optional by assigning a default value for them in function declaration

- # Named Parameters and Positional Parameters

- arguments. Named arguments allow you to invoke a method by specifying parameter values in any order you choose.

the params Modifier

- The params keyword allows you to pass into a method a variable number of identically typed parameters as a single logical parameter.

Object Oriented Programming with C#

Method Overloading

when you define a set of identically named methods that differ by the number (or type) of parameters, the method in is said to be overloaded.

Constructors

Constructors

- Whenever object of a class or struct is created, its constructor is called.
- A class or struct may have multiple constructors that take different arguments.
- Constructors enable the programmer to initialize data members

Properties

Properties

- A property is a member that provides a flexible mechanism to read, write, or compute the value of a private field.
- Properties can be used as if they're public data members, but they're special methods called accessors.
- The get accessor returns the value of the private field, and the set accessor may perform some data validation before assigning a value to the private field

Auto-implemented properties

- Are used in cases if you don’t want to implement any extra logic but simply set or return the value.