

# .NET Visual Studio Solution

.NET

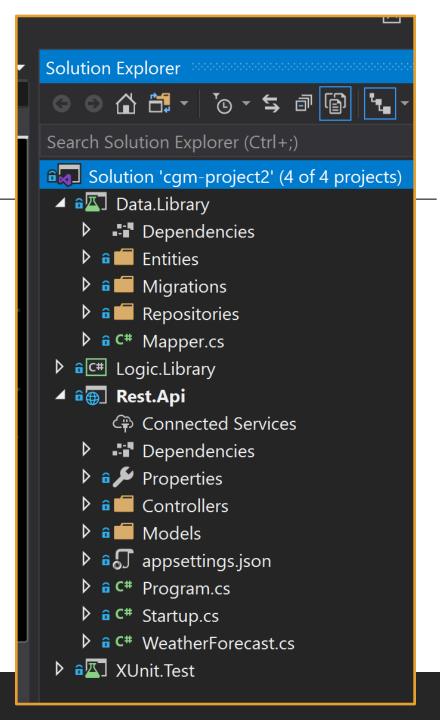
NET projects are contained within a solution. A solution is a container for one or more related projects.

## .NET Solution

https://docs.microsoft.com/en-us/visualstudio/ide/solutions-and-projects-in-visual-studio?view=vs-2019#solutions

A **Solution** is a container for <u>one or more</u> related projects along with their build information, Visual Studio window settings, and any miscellaneous files that aren't associated with one particular project.

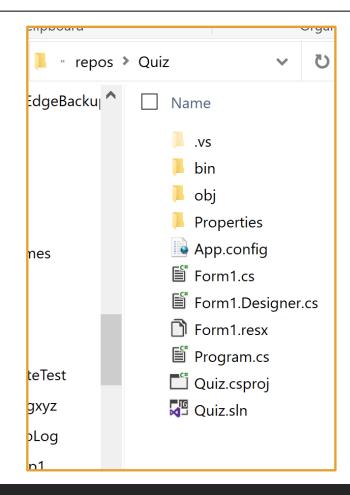
A **solution** is described by a text file (extension .sln) in XML format. It is not intended to be edited by hand.



## .NET Solution - Projects

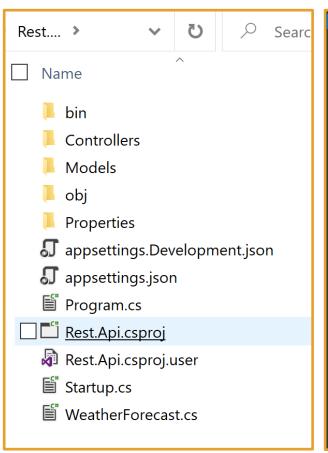
https://docs.microsoft.com/en-us/visualstudio/ide/solutions-and-projects-in-visual-studio?view=vs-2019 https://docs.microsoft.com/en-us/dotnet/csharp/tour-of-csharp/program-structure

- An app in Visual Studio starts with a project. A project contains all files that, when compiled, are placed into an executable (.exe) or a library (.dll).
- Files can include source code, icons, images, data files, etc.
- A project contains compiler settings and other configuration files that might be needed by various services or components.
- Visual Studio uses MSBuild to build each project in a solution, and each project contains an MSBuild project file.
- The file extension for a C# project is .csproj.
- The project file is an XML document that contains all the information and instructions that MSBuild needs in order to build a project including the content, platform requirements, versioning information, web server or database server settings, and the tasks to perform.



# .NET Solution - Projects

https://docs.microsoft.com/en-us/visualstudio/ide/solutions-and-projects-in-visual-studio?view=vs-2019 https://docs.microsoft.com/en-us/dotnet/csharp/tour-of-csharp/program-structure



```
▲ ⊕ Rest.Api
                                                                                                       Connected Services
                                                                                                    Dependencies
Rest.Api.csproj - X Logic.Library.csproj
                              Data.Library.csproj
                                                                                                     Properties
     ▶ a ■ Models
     3 崫
          <PropertyGroup>
                                                                                                    ▶ • ∏ appsettings.json
                                                                                                     ▶ a C# Program.cs
             <TargetFramework>netcoreapp3.0</TargetFramework>
                                                                                                     ▶ a C# Startup.cs
             <UserSecretsId>d53aec83-0d01-4cab-9f30-cfb9fd362f5a/UserSecretsId>
                                                                                                     ▶ a C# WeatherForecast.cs
           </PropertyGroup>
                                                                                                   ▶ a XUnit.Test
     <PackageReference Include="Microsoft.AspNetCore.Authentication.JwtBearer" Version="3.0.0" />
    10 🚊
             <PackageReference Include="Microsoft.EntityFrameworkCore.Design" Version="3.0.0">
               <PrivateAssets>all</PrivateAssets>
               <IncludeAssets>runtime; build; native; contentfiles; analyzers; buildtransitive</IncludeAssets>
             </PackageReference>
             <PackageReference Include="Microsoft.EntityFrameworkCore.SqlServer" Version="3.0.0" />
    15 
             <PackageReference Include="Microsoft.EntityFrameworkCore.Tools" Version="3.0.0">
               <PrivateAssets>all</PrivateAssets>
               <IncludeAssets>runtime; build; native; contentfiles; analyzers; buildtransitive</IncludeAssets>
             </PackageReference>
             <PackageReference Include="Microsoft.Extensions.Logging.Debug" Version="3.0.0" />
             <PackageReference Include="Microsoft.VisualStudio.Web.CodeGeneration.Design" Version="3.0.0" />
           </ItemGroup>
          <ItemGroup>
             <ProjectReference Include="..\Data.Library\Data.Library.csproj" />
             <ProjectReference Include="..\Logic.Library\Logic.Library.csproj" />
           </ItemGroup>
```

Solution Explorer

☐ Data.Library
Data.Library
Dependencies
Definition

▶ a Migrations

Solution 'cgm-project2' (4 of 4 projects)

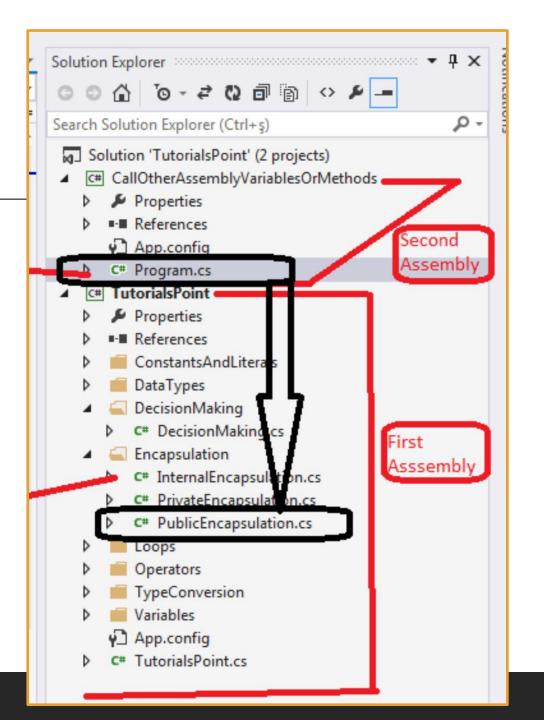
### .NET Solution - Assembly

https://docs.microsoft.com/en-us/dotnet/standard/assembly/

#### Assemblies:

- form the fundamental units of deployment, version control, reuse, activation scoping, and security permissions for *.NET*-based applications.
- are a collection of types and resources that work together and form a logical unit of functionality.
- take the form of **executable** (.exe) or **dynamic link library** (.dll) files.
- provide the *Common Language Runtime* with the information it needs to be aware of *type* implementations.

In .NET, you can build an assembly from one or more source code files. Each projects' files are compiled (combined) into one .dll of .exe file called an Assembly.



### .NET Solution - Assembly

https://docs.microsoft.com/en-us/dotnet/standard/assembly/

#### An **assembly** is:

- Code that the CLR executes. Each assembly can have only one entry point (Main).
- <u>Security boundary</u>. An **assembly** is the unit at which permissions are requested and granted.
- Version boundary. The assembly is the smallest versionable unit in the CLR. All types and resources in the same assembly are versioned as a unit.
- <u>Deployment unit</u>. When an application starts, only the *assemblies* that the application initially calls must be present. Other *assemblies* are retrieved on demand. This is called *Just-In-Time* (*JIT*) *compiling*.

