



.NET Visual Studio Solution

.NET

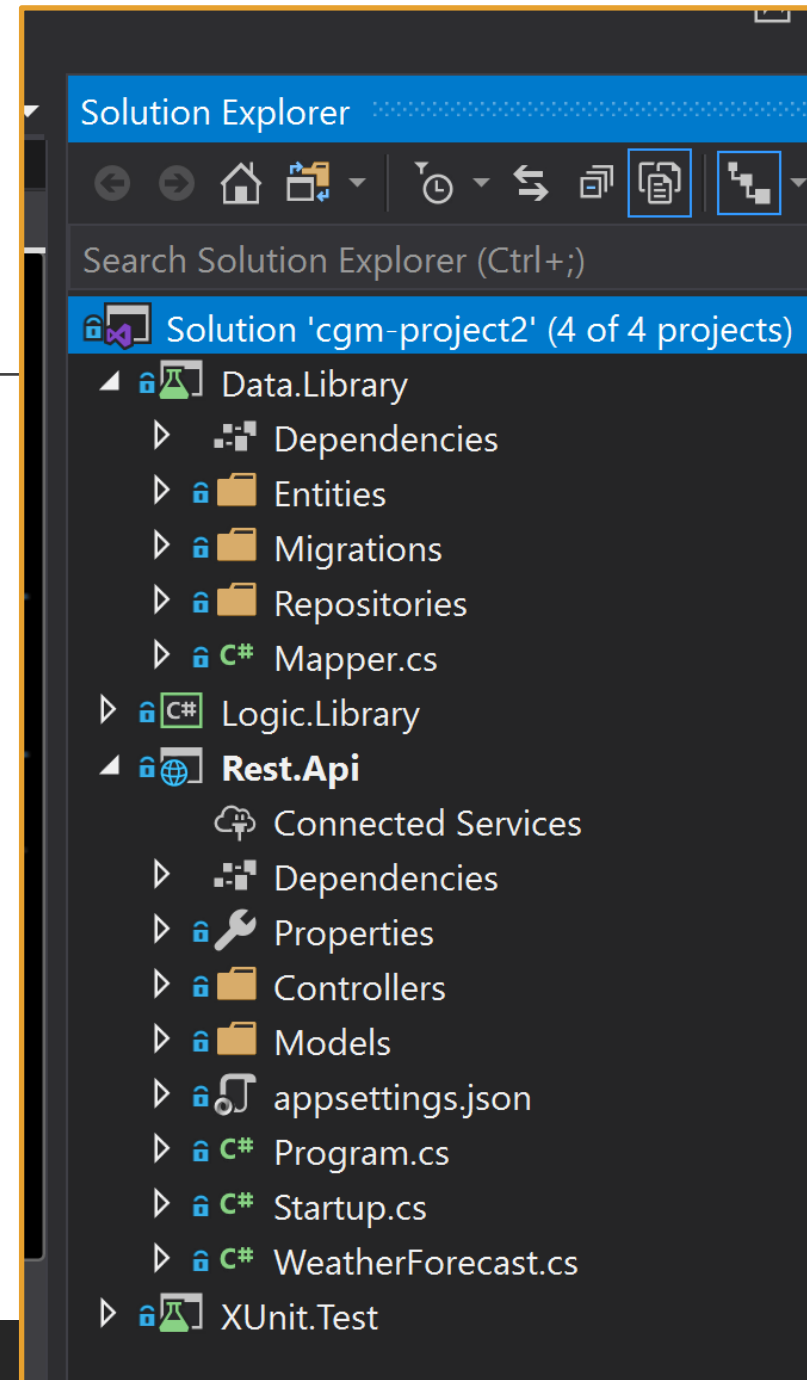
*.NET **projects** are contained within a **solution**. A **solution** is a container for one or more related **projects**.*

.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 one or more related projects along with their build information, Visual Studio window settings, and any miscellaneous files that aren't associated with one particular project.

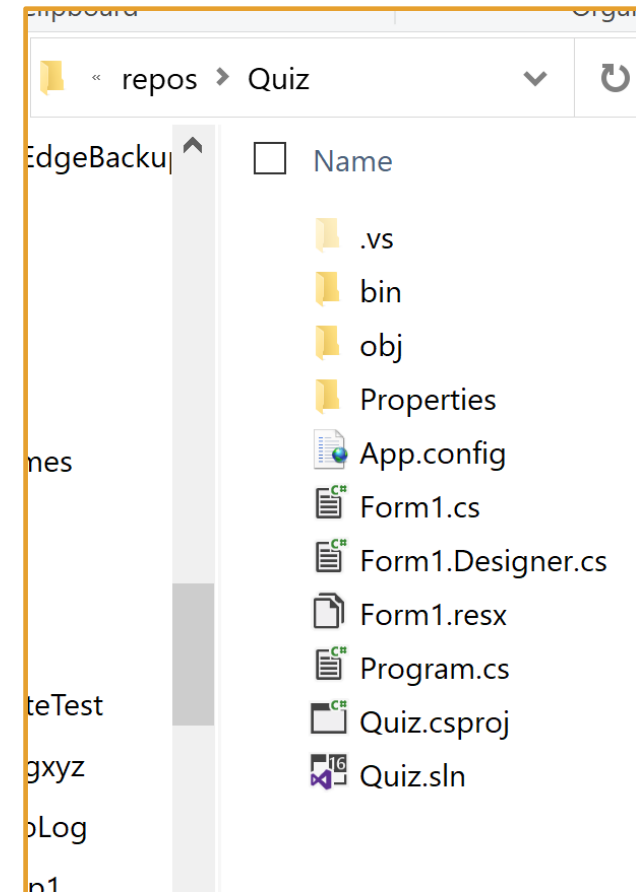
A ***solution*** is described by a text file (extension .sln) in XML format. It is not intended to be edited by hand.



.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>

The image shows a Visual Studio interface with three main components:

- Left Panel (Solution Explorer):** Displays the project structure for 'Rest.Api'. It includes folders like 'bin', 'Controllers', 'Models', 'obj', and 'Properties', along with files such as 'appsettings.Development.json', 'appsettings.json', 'Program.cs', 'Rest.Api.csproj' (selected), 'Rest.Api.csproj.user', 'Startup.cs', and 'WeatherForecast.cs'.
- Center Panel (Code Editor):** Shows the content of 'Rest.Api.csproj'. The XML structure defines the project as a .NET Core Web application (TargetFramework: netcoreapp3.0) with various package references including Microsoft.AspNetCore.Authentication.JwtBearer, Microsoft.EntityFrameworkCore.Design, Microsoft.EntityFrameworkCore.SqlServer, Microsoft.EntityFrameworkCore.Tools, Microsoft.Extensions.Logging.Debug, and Microsoft.VisualStudio.Web.CodeGeneration.Design. It also includes project references to 'Data.Library' and 'Logic.Library'.
- Right Panel (Solution Explorer):** Provides an overview of the entire solution 'cgm-project2' (4 of 4 projects). It lists the 'Data.Library' project and its files (Dependencies, Entities, Migrations, Repositories, Mapper.cs), the 'Logic.Library' project, and the 'Rest.Api' project with its associated files (Connected Services, Dependencies, Properties, Controllers, Models, appsettings.json, Program.cs, Startup.cs, WeatherForecast.cs, XUnit.Test).

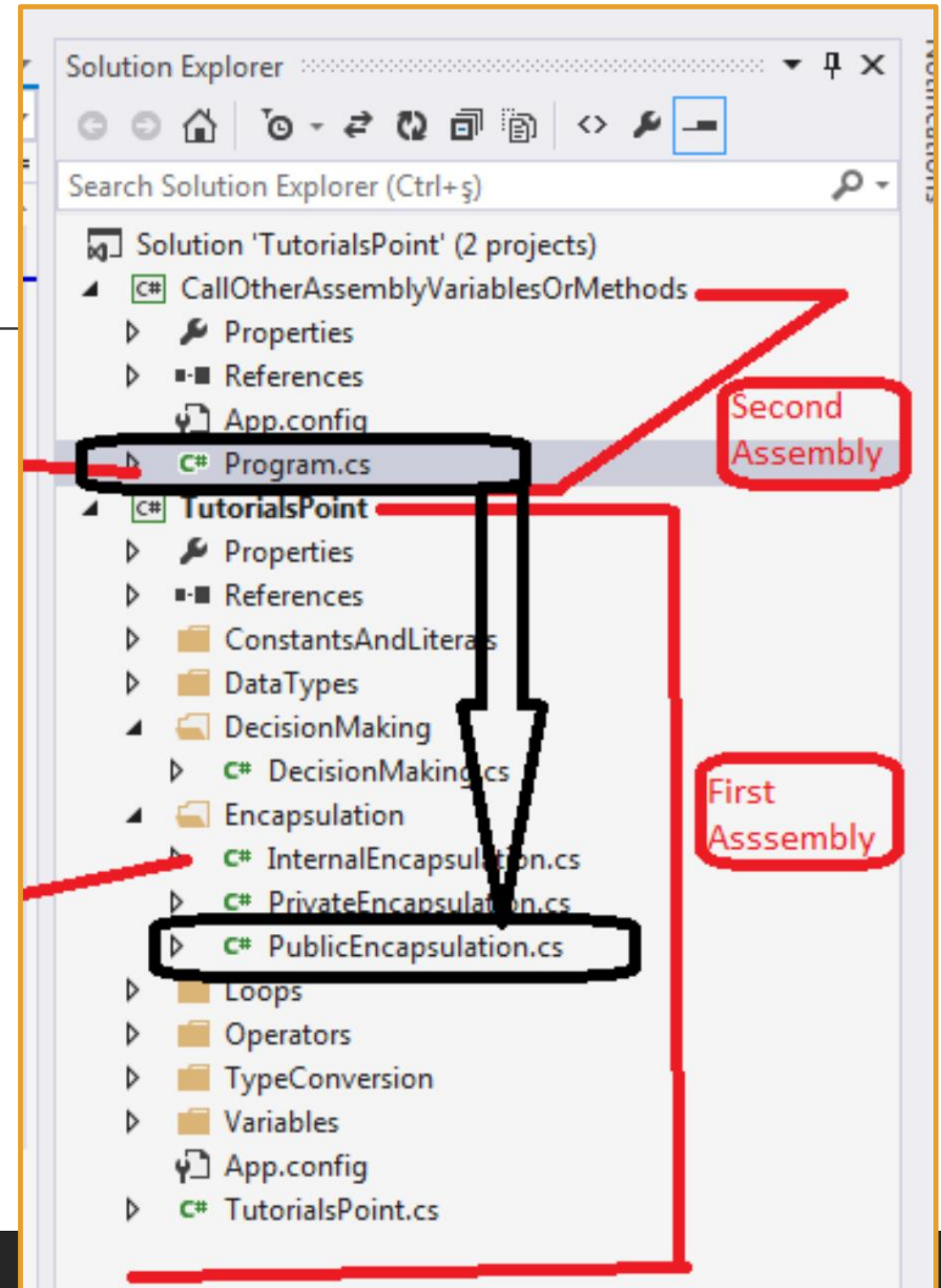
.NET Solution - Assembly

<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 project's files are compiled (combined) into one **.dll** or **.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).
- Security boundary. 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.
- Deployment unit. 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**.

