



# .NET Solution

---

.NET

A **project** is contained within a **solution**. Despite its name, a **solution** is not an "answer". It's simply a container for one or more related **projects** (along with build information, Visual Studio window settings, and any miscellaneous files that aren't associated with a particular **project**). A **solution** is described by a text file (extension .sln) with its own unique format; it's not intended to be edited by hand.

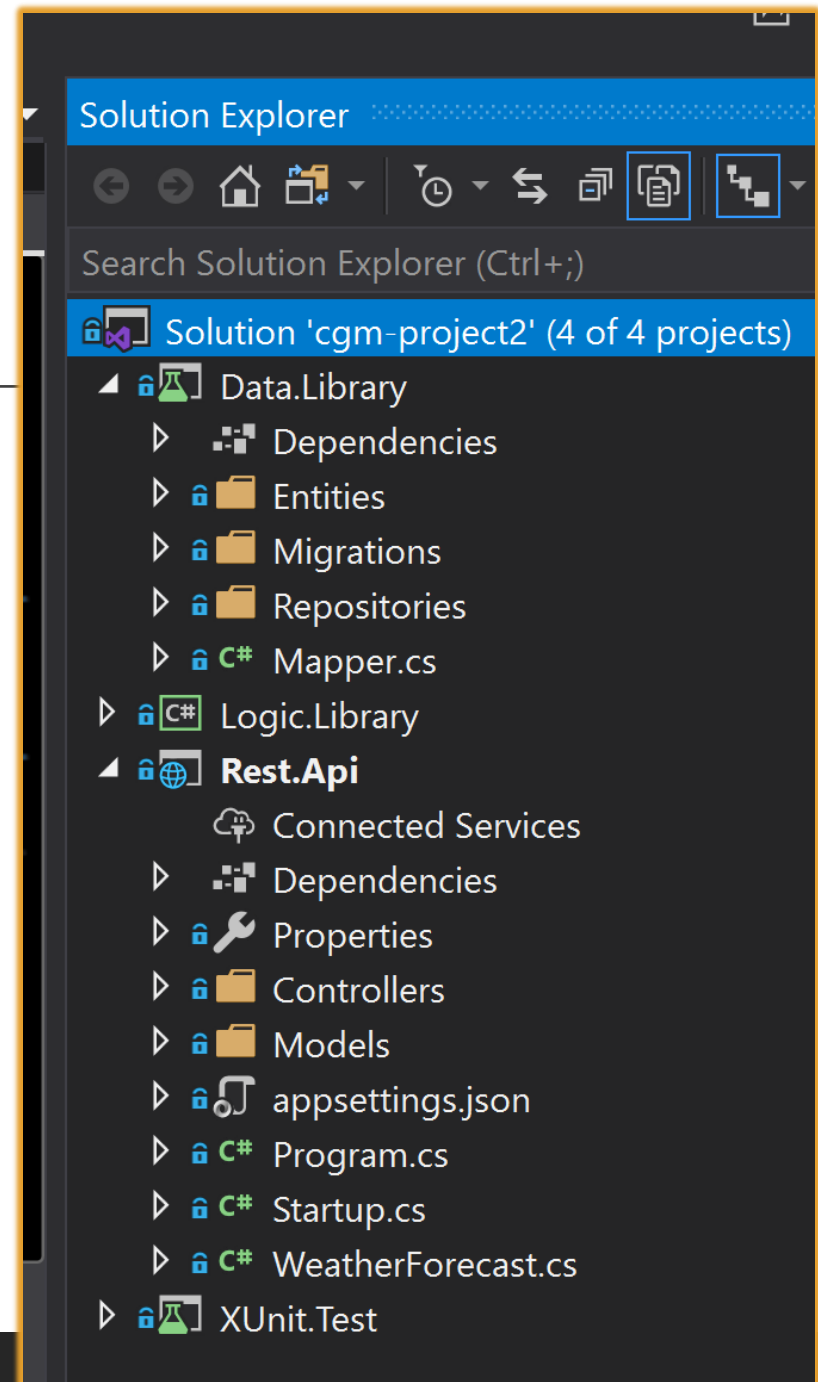
[HTTPS://DOCS.MICROSOFT.COM/EN-US/VISUALSTUDIO/IDE/SOLUTIONS-AND-PROJECTS-IN-VISUAL-STUDIO?VIEW=VS-2019](https://docs.microsoft.com/en-us/visualstudio/ide/solutions-and-projects-in-visual-studio?view=vs-2019)

# .NET Solution

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

Projects are contained within solutions. A Solution is a container for one or more related projects along with build information, Visual Studio window settings, and any miscellaneous files that aren't associated with a particular project.

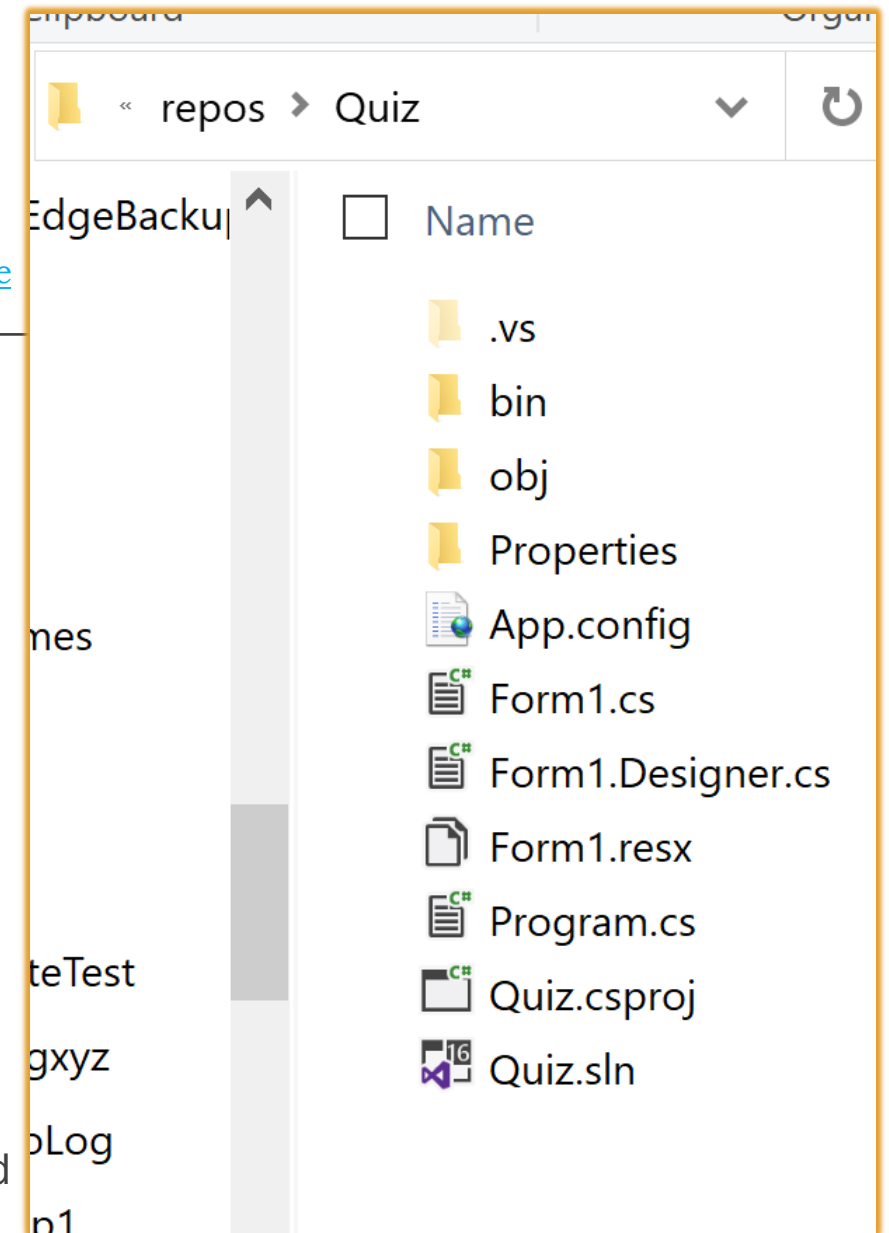
A solution is described by a text file (extension .sln) with its own unique format; it's not intended to be edited by hand.



# .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 or website in Visual Studio starts with a project. A project contains all files that are compiled into an executable, library, or website.
- Those files can include source code, icons, images, data files, etc.
- A project also contains compiler settings and other configuration files that might be needed by various services or components that your program communicates with.
- 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 your project, including the content, platform requirements, versioning information, web server or database server settings, and the tasks to perform.

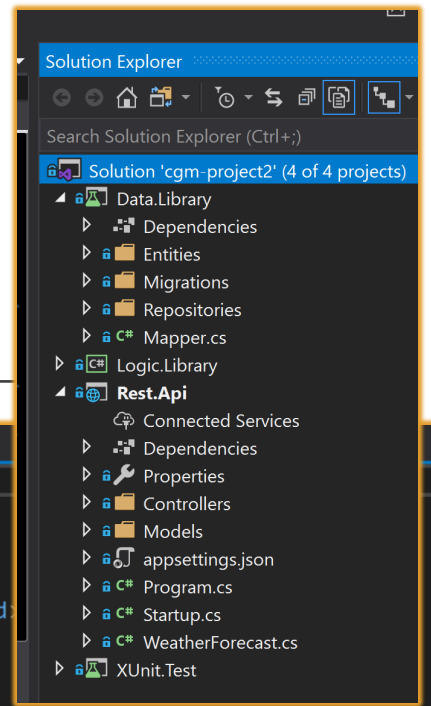
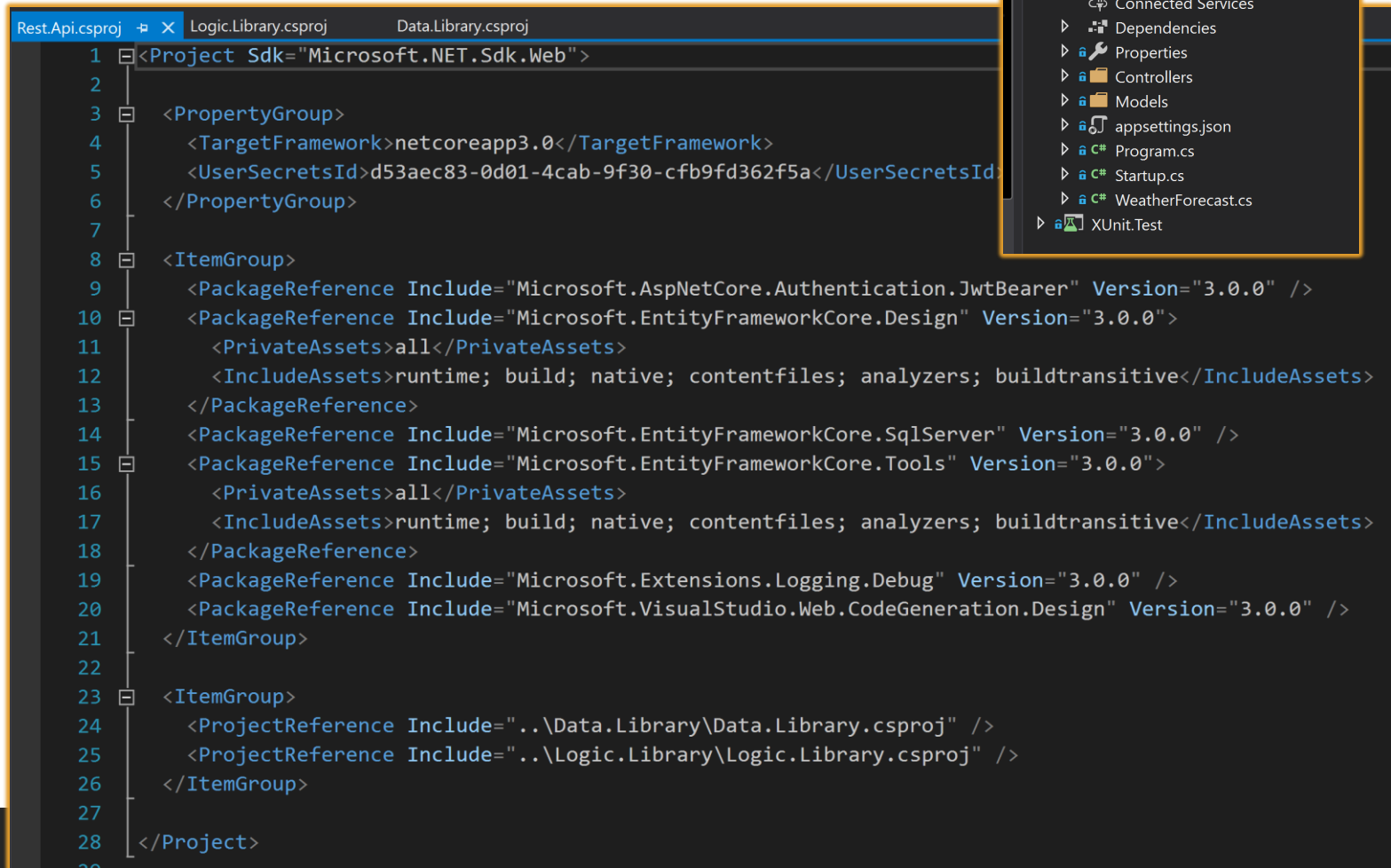
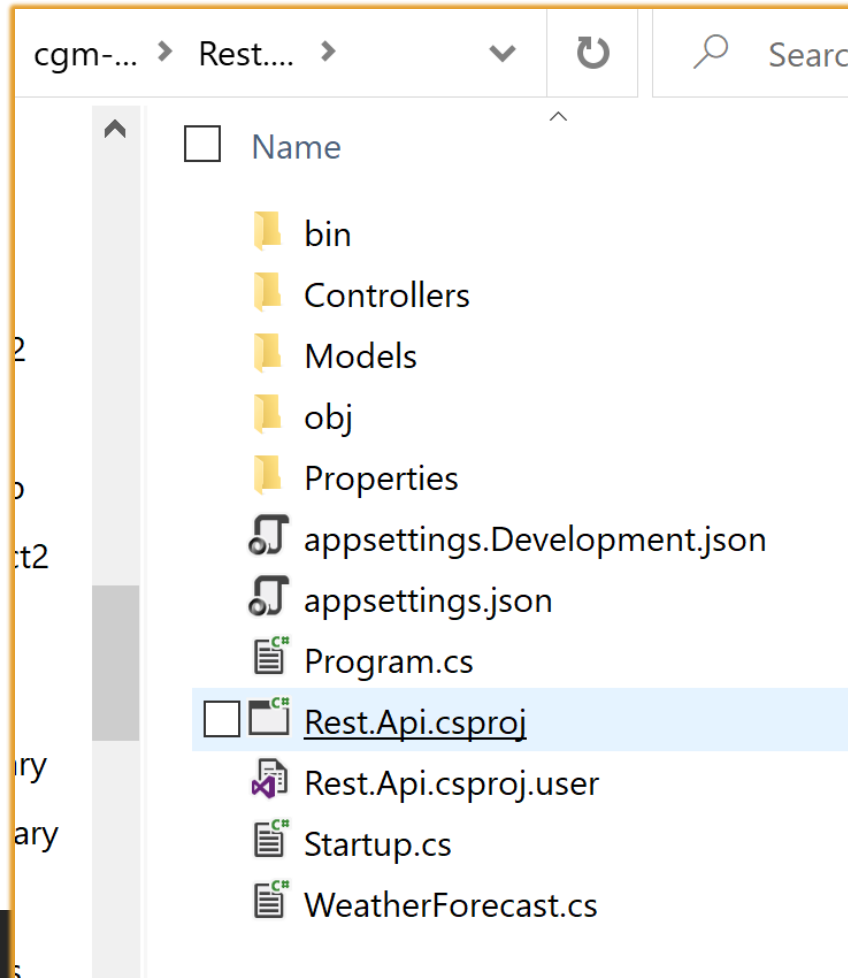




# .NET Solution - Projects

<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>



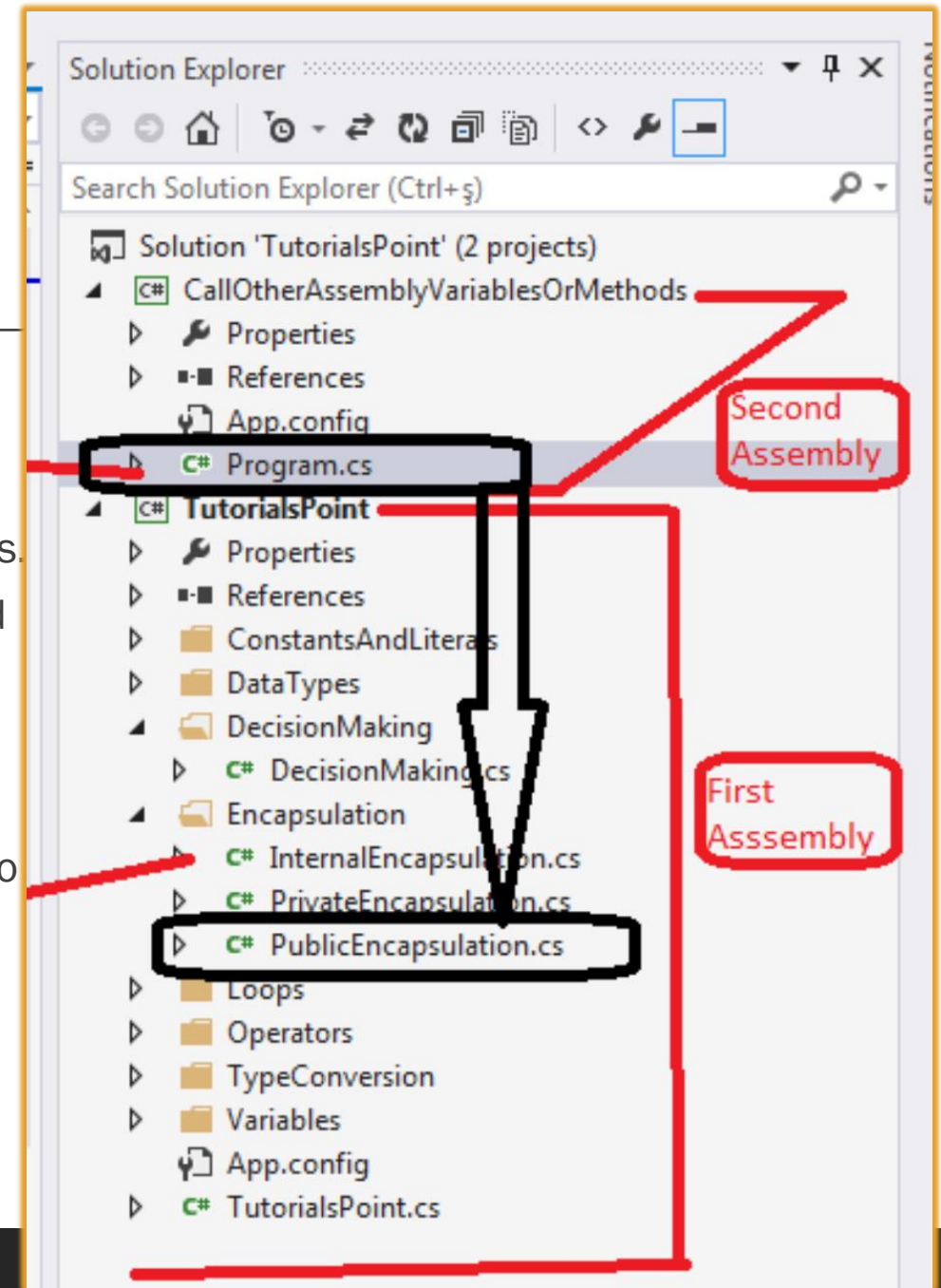
# .NET Solution - Assembly

<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.
- is a collection of types and resources that are built to work together and form a logical unit of functionality.
- take the form of **executable (.exe)** or **dynamic link library (.dll)** files.
- are the building blocks of **.NET** applications.
- provide the **Common Language Runtime** with the information it needs to be aware of **type** implementations.

In **.NET Core** and **.NET Framework**, you can build an assembly from one or more source code files. Each Projects files are compiled (combined) into one big .dll 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.

