**Nuget**

Nuget is a tool for which developers can create, share, and consume useful code as bundled packages.

Put simply, a NuGet package is a single ZIP file with the .nupkg extension that contains compiled code (DLLs), other files related to that code, and a descriptive manifest that includes information like the package's version number. Developers with code to share create packages and publish them to a public or private host. Package consumers obtain those packages from suitable hosts, add them to their projects, and then call a package's functionality in their project code. NuGet itself then handles all of the intermediate details.

We tell the project what dependency to use (“package”), which version to use, and what location to get it from (“feed”) – the **source** of the file

Diagram

Description automatically generated

The dependency tells the project how to configure **itself** (either by the packages folder structure, or with a .props/.targets file)

Diagram

Description automatically generated

1. **Creator**:

* The source of the package
* Probably a visual studio project, .NET project, C++, source code, etc.

1. **Build/pack**

* All contents of the package are gathered up
  + .dll, .lib, source code, .pdb, additional configs (.props/.targets)
  + Metadata about the package (name, version number) 🡪 Represented as either .nuspec file (short xml) or for .NET projects can be a part of the .csproj file
* Packing will be run by one of the tools: DotNet CLI, Nuget CLI, MsBuild (depends on your needs)

1. **Package**

* Output of pack command: .nupkg file
* This is just a zip file, and will hold all the contents you packed
* Unzipping this files to inspect the contents is a useful way to diagnose issues

1. **Publish**:

* Publish packages to the Host, using DotNet CLI, Nuget CLI, MsBuild

1. **Host**

* Host can be public or private, hosted over https, networked drive file system, or local file system
* E.g, nuget.org, rocscience.visualstudio
* The NuGet package manager will try to look for the nugget package from all sources, which is specified in Tools > Nuget Package Manager > Package Sources

1. **Browsing/Install - Restore**

* Typically will be done from visual studio or by looking/searching the feeds UI
* Can be done with Visual Studio Package Manager Console or dotnet/nuget CLI (dotnet restore/nuget restore)

1. **Consumer**

* Download/install 🡪 Set up configurations to actually use the packages to our project