Building Fabulous NuGet Packages

Jason Bock
Staff Software Engineer
Rocket Mortgage

Personal Info

- https://mstdn.social/@jasonbock
- https://bsky.app/profile/jasonbock.bsky.social
- https://github.com/jasonbock
- https://youtube.com/JasonBock
- https://www.linkedin.com/in/jasonrbock/
- jason.r.bock@outlook.com

Downloads

https://github.com/JasonBock/Collatz
https://github.com/JasonBock/UsingUsings
https://github.com/JasonBock/Presentations

Overview

- History
- Concepts
- Packages
- Tools
- Call to Action

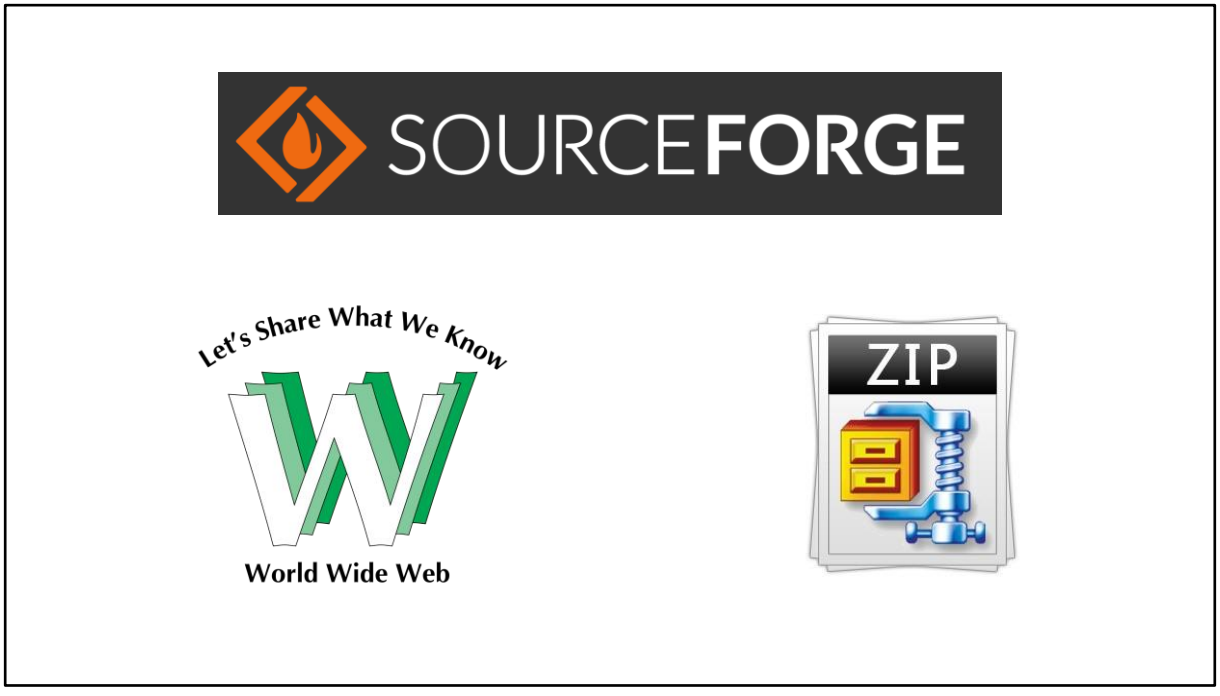
Remember…
https://github.com/JasonBock/Collatz
https://github.com/JasonBock/UsingUsings
https://github.com/JasonBock/Presentations

History

Building Fabulous NuGet Packages

If you’re a .NET dev, you’ve heard of NuGet. But most .NET devs typically only know it at a cursory level. It’s worth spending time going over how we got NuGet and why it’s useful.

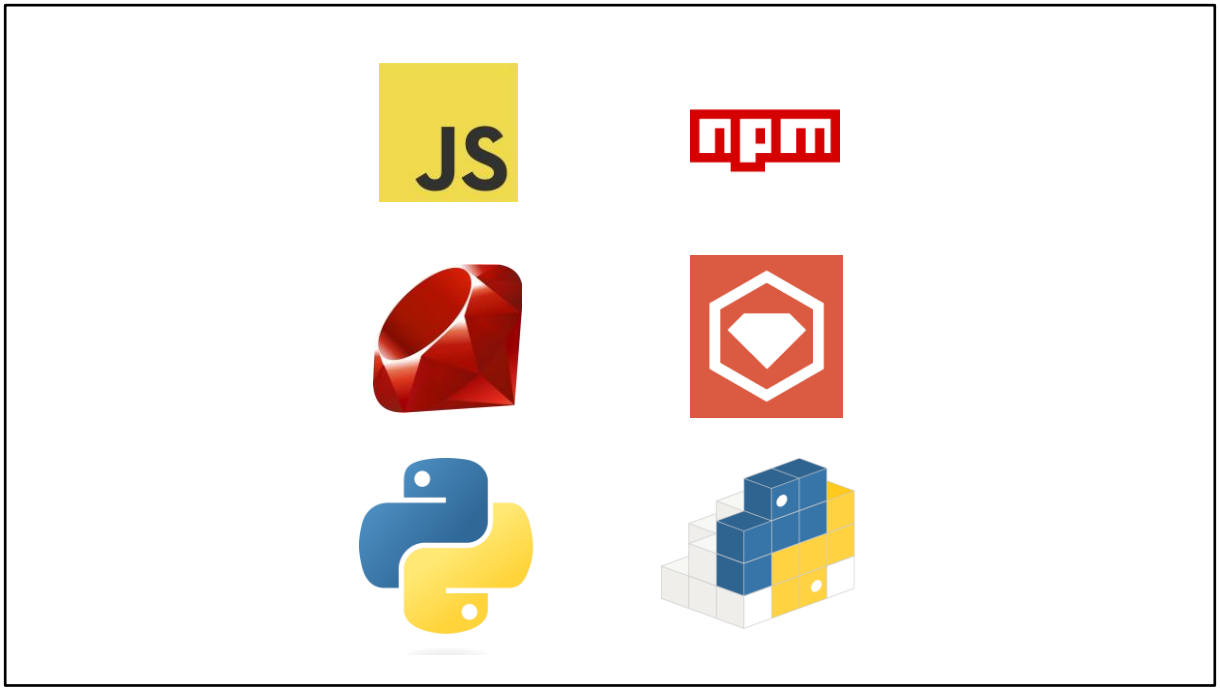
http://brandonvaughn.com/wp-content/uploads/2016/02/hear-1024x681.jpg



When .NET was first introduced, there wasn't a big emphasis on sharing code, though .NET devs were definitely using whatever was available to share tools and libraries. It was a very fast-and-loose environment, where nefarious behaviors could be introduced with ease. I remember hearing about NUnit, and I can't remember for the life of me how I was actually able to get everything.

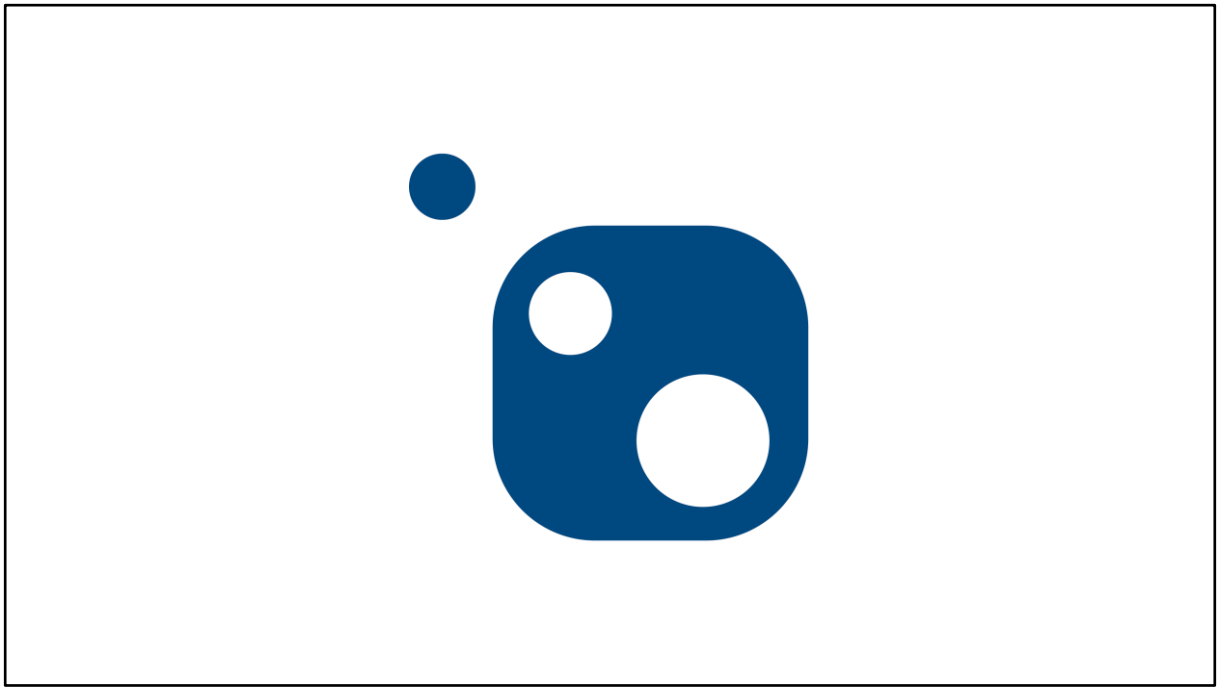
Over time, it became clear that .NET needed a “good” way to share code

https://unsplash.com/photos/qgHGDbbSNm8



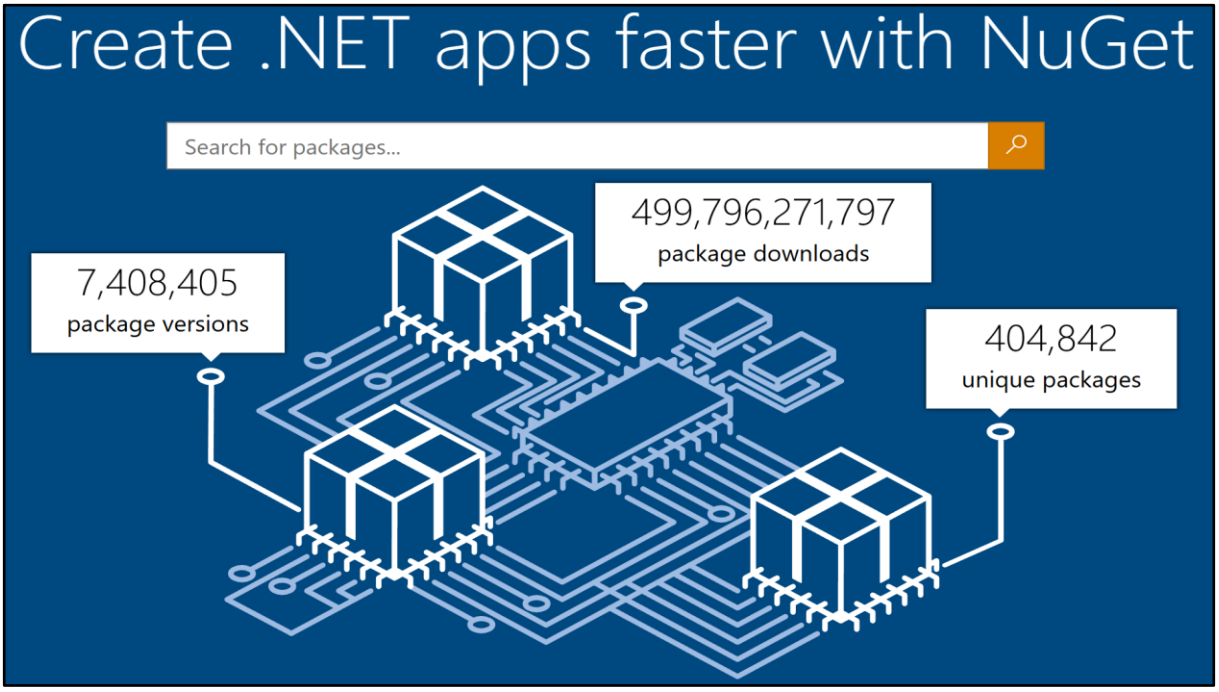
Lots of other languages have package systems as well. For example,

JavaScript => npm
Ruby => Ruby Gems
Python => Python Package Index



NuGet was introduced as .NET's package format in 2010. You can create packages and upload them to nuget.org.

https://nuget.org



These statistics were generated on 2024/6/2

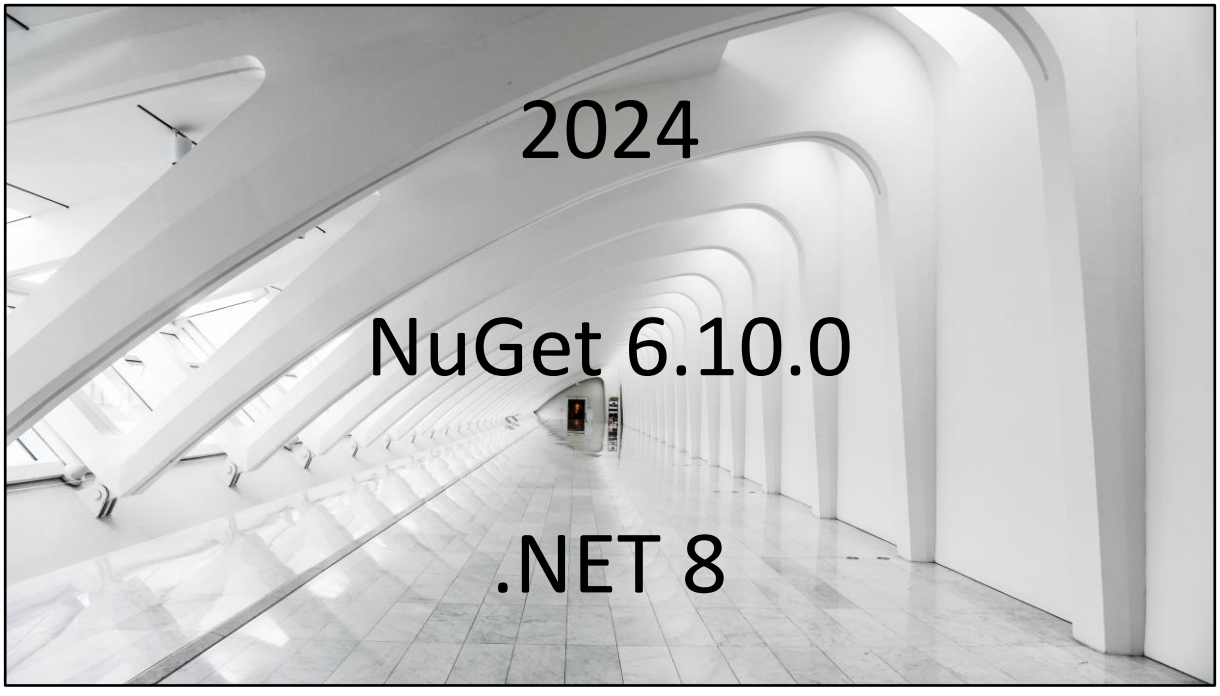
https://www.nuget.org/

Package Downloads

Name	Downloads
	142,178,354
	49,119,017
	34,502,391
	28,028,861
	26,503,545
	26,471,060

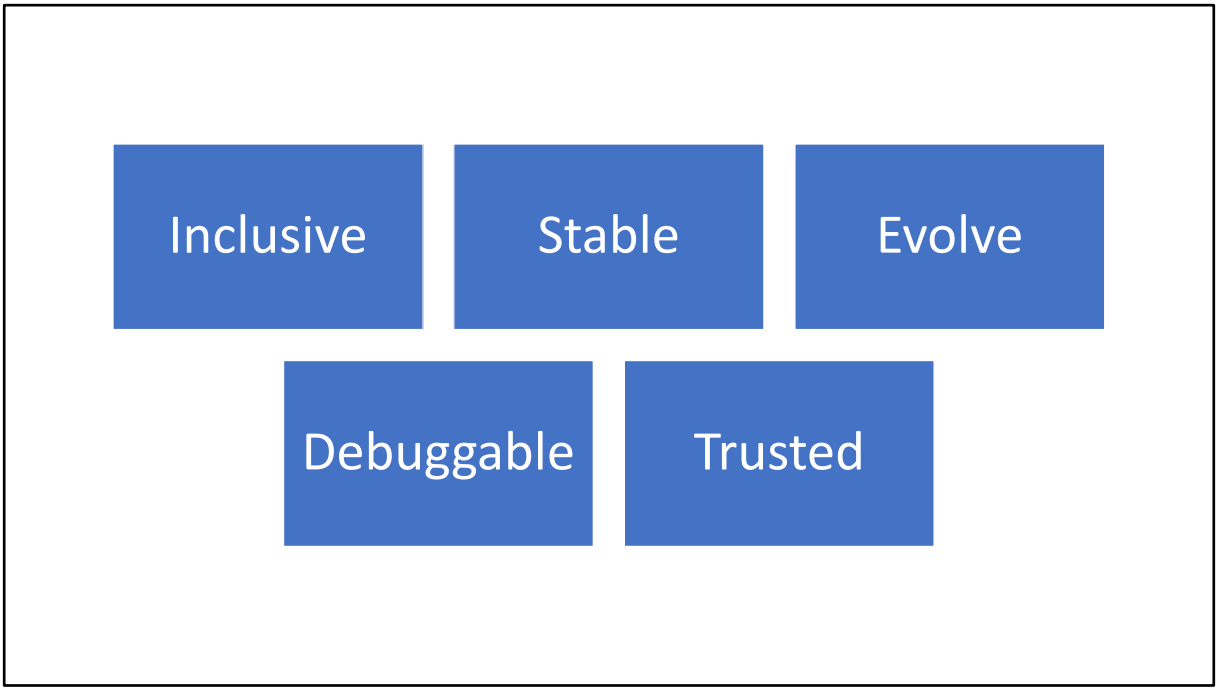
As of 2/25/2023, can you guess which package is at the top of the list? ☺

https://www.nuget.org/stats



Keep in mind in this talk, I'm talking about modern NuGet concepts. In 2024, the current version is NuGet 6.10.0 (as of 2024/6/2). If you've used NuGet in the past, things weren't as straightforward. With SDK-style .cspoj files, a cross-platform .NET runtime, there are more features and arguably "cleaner" ways to create and manage packages

https://www.pexels.com/photo/long-exposure-photography-white-dome-building-interior-911758/



NuGet packages are (or at least should be)

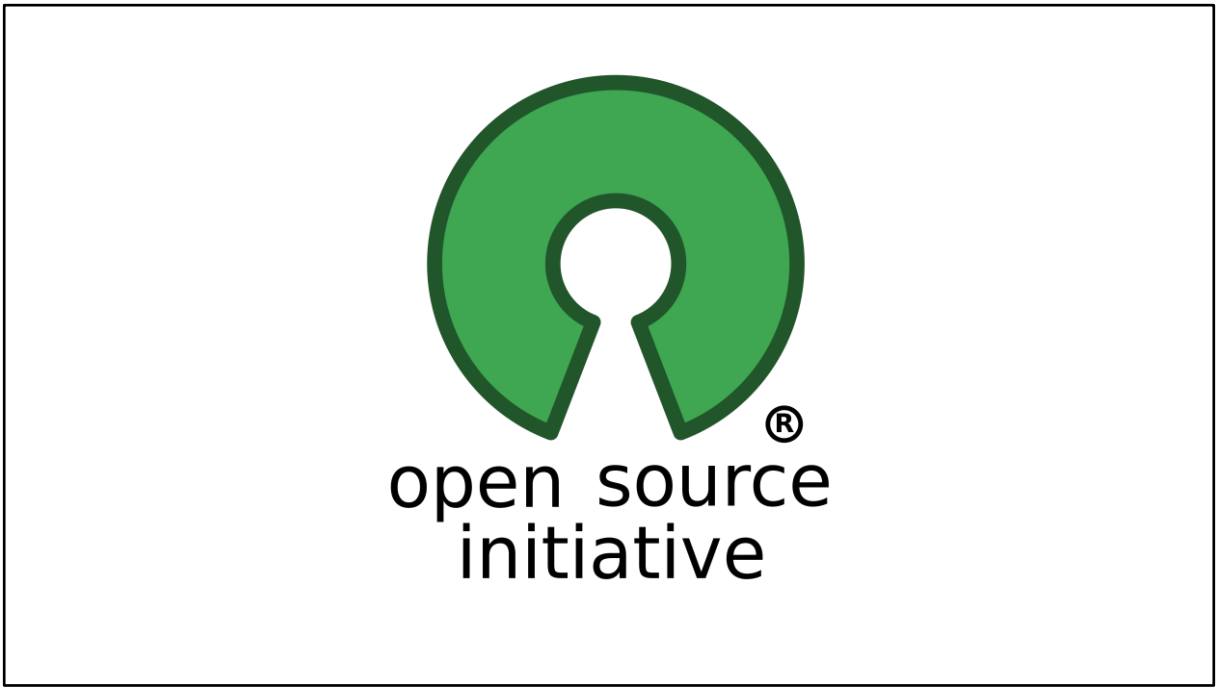
Inclusive - Good .NET libraries strive to support many platforms, programming languages, and applications.
Stable - Good .NET libraries coexist in the .NET ecosystem, running in applications built with many libraries.
Designed to evolve - .NET libraries should improve and evolve over time, while supporting existing users.
Debuggable - .NET libraries should use the latest tools to create a great debugging experience for users.
Trusted - .NET libraries have developers' trust by publishing to NuGet using security best practices.

Having these qualities cultivates an environment where people can safely share their ideas.

https://learn.microsoft.com/en-us/dotnet/standard/library-guidance/

Interestingly enough, throughout my career, I've noticed that people tend to consume packages, rather than creating them. This is why .NET developers tend to know that NuGet exists, but mostly that it's the place where to get free stuff.

https://unsplash.com/photos/CAhjZmVk5H4



This is something that people should consider, and not just from an OSS perspective.

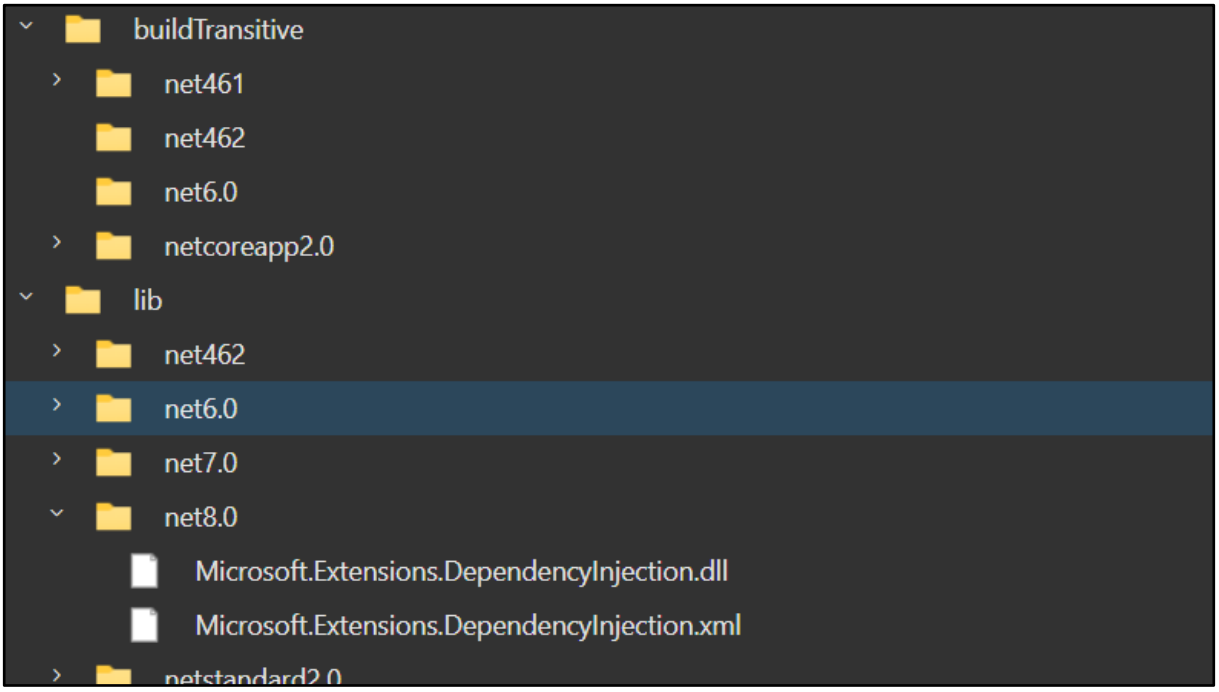
https://opensource.org/

Inner sourcing encourages companies to follow OSS practices internally to share functionality across teams. Building NuGet packages for internal consumption is a part of that.

https://www.pexels.com/photo/photo-of-people-doing-handshakes-3183197/

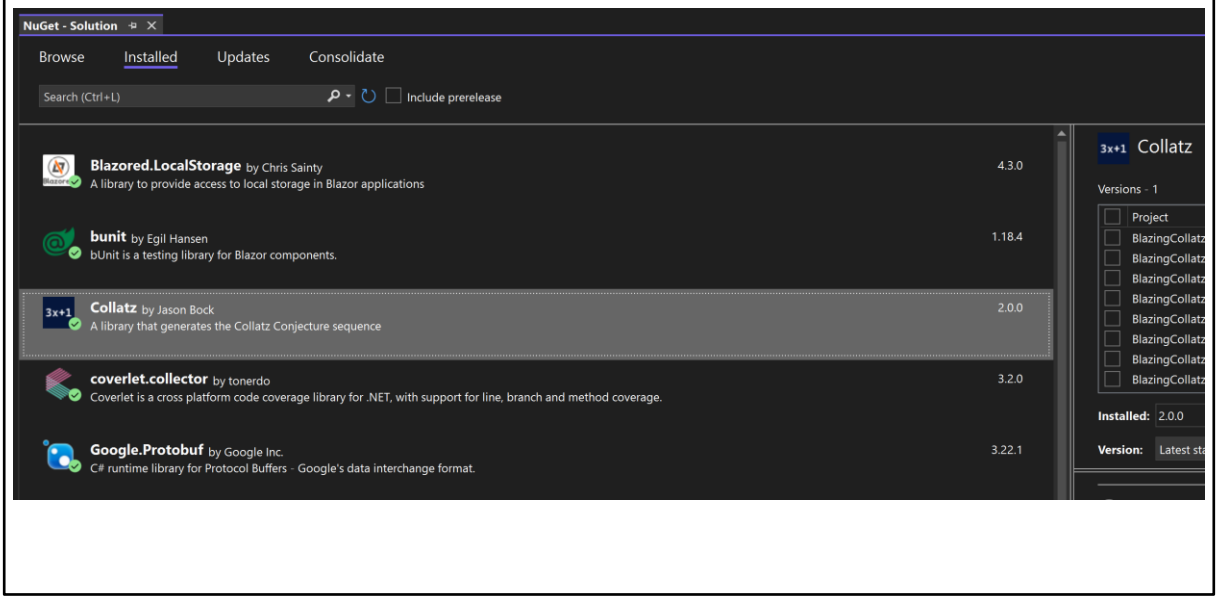
Concepts

Building Fabulous NuGet Packages



A NuGet (or .nupkg) file is just a bunch of content in well-known folder locations. This is what Microsoft.Extensions.DependencyInjection looks like

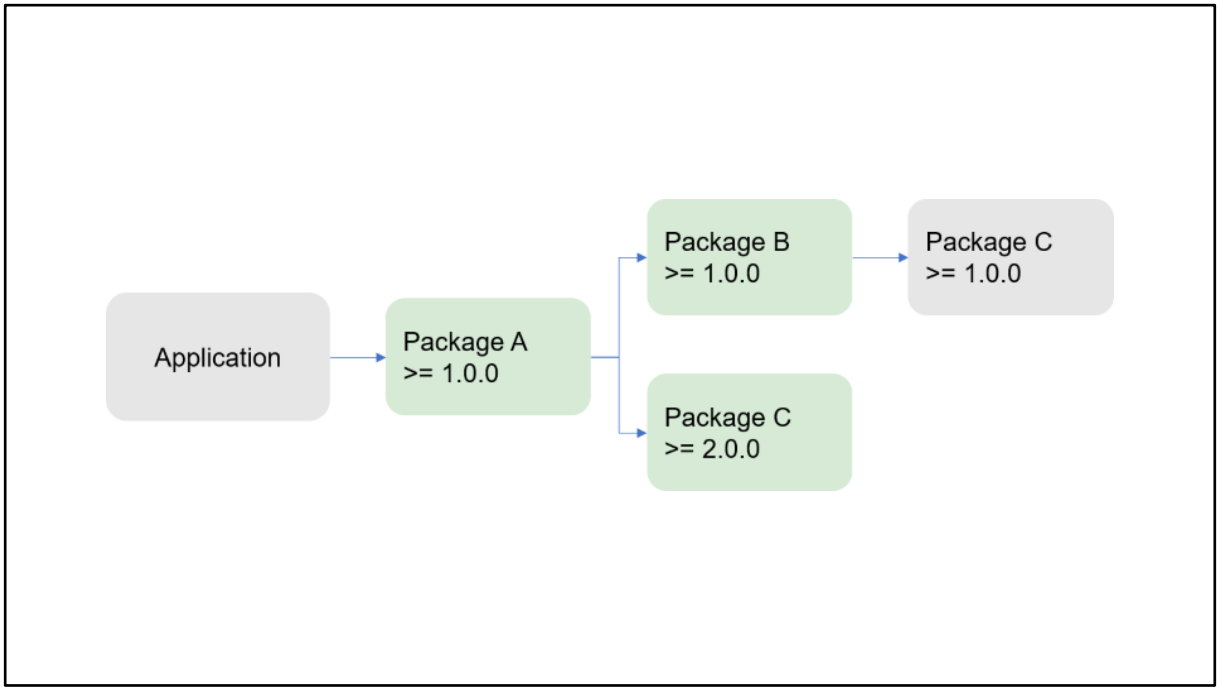
https://nuget.info/packages/Microsoft.Extensions.DependencyInjection/8.0.0



In VS, you can use the Manage NuGet Packages window to find, reference, update, and consolidate packages.

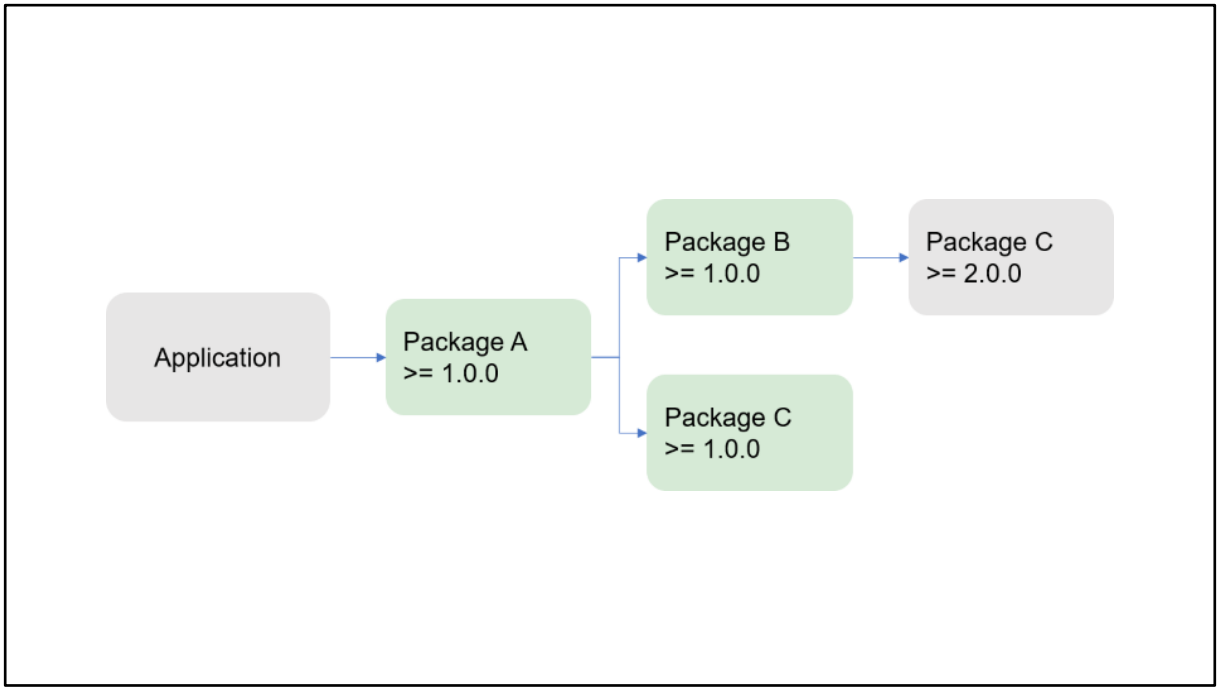
While your project may reference a handful of packages (and if you're referencing 20+ packages, you may want to rethink that), those packages reference other packages. These are transitive references. This matters because, depending on your tree, packages may reference different versions of other packages, which may require you to pick which version to reference.

https://unsplash.com/photos/S297j2CsdlM



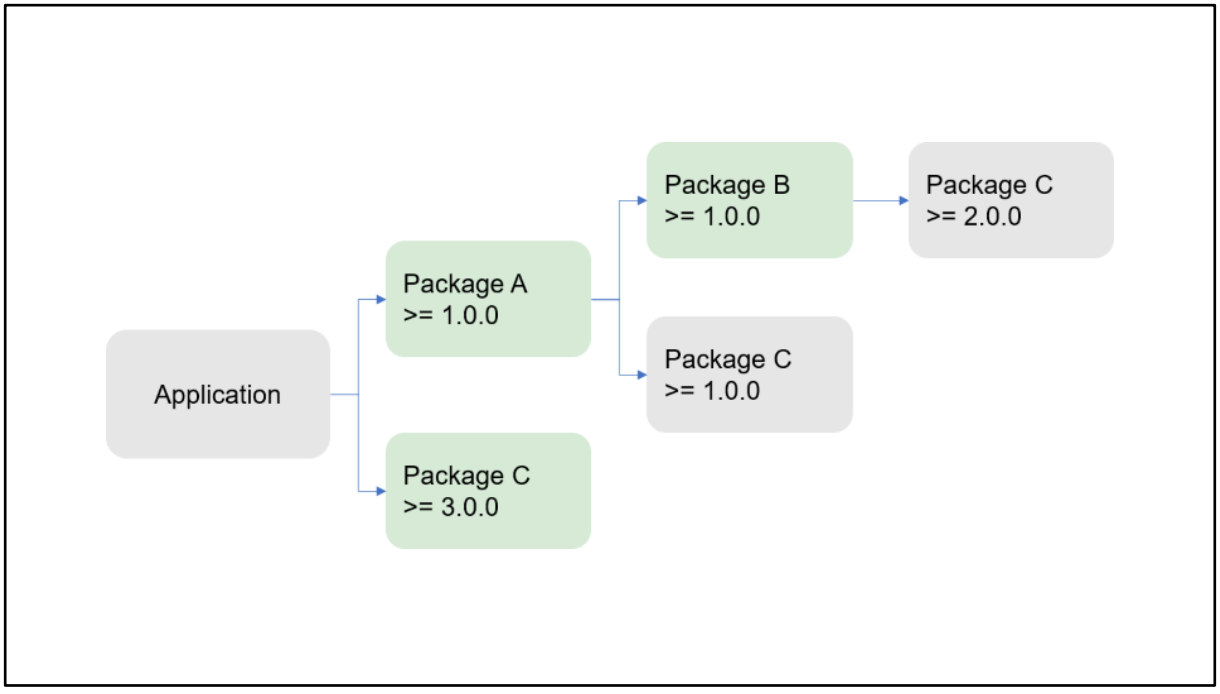
Let’s spend a bit of time on how NuGet handles dependency resolution. Essentially, NuGet takes a “direct dependency wins” approach by default. That means in this case, Package C 2.0.0 will be the one that “wins”.

https://learn.microsoft.com/en-us/nuget/concepts/dependency-resolution



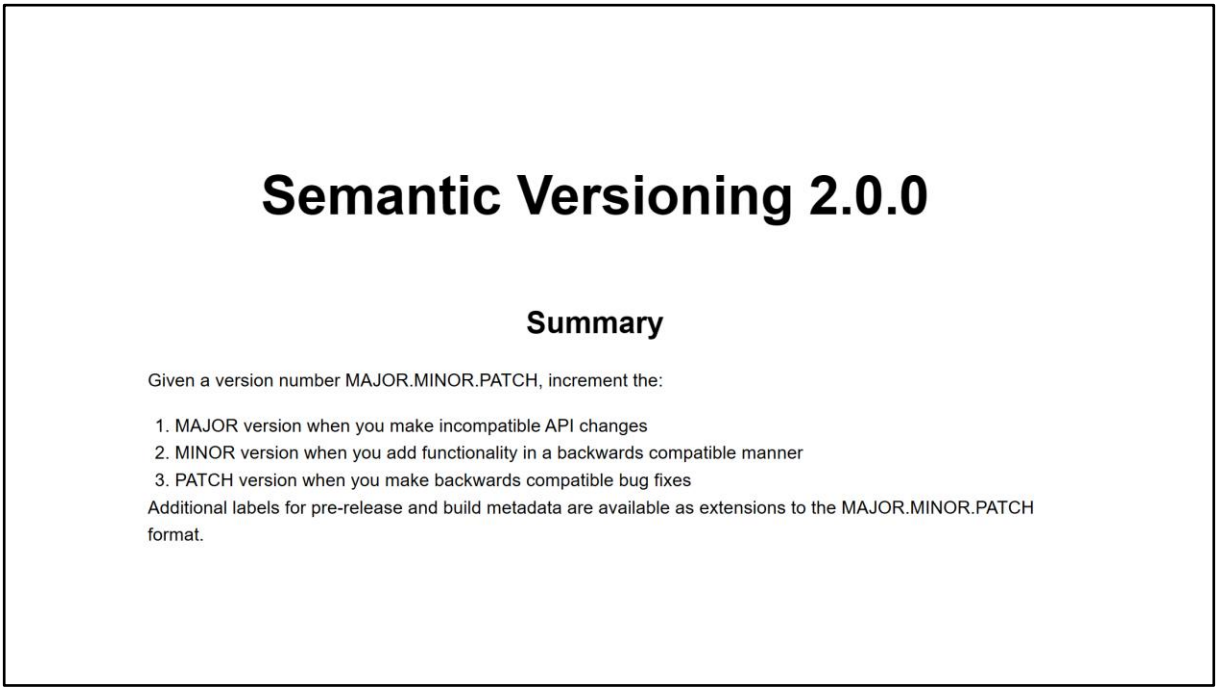
Note that in this case, Package C 1.0.0 will “win” because of the rule.

https://learn.microsoft.com/en-us/nuget/concepts/dependency-resolution



You can take a direct dependency on a version of Package C, which, by the “DDW” rule, 3.0.0 will be the one loaded.

https://learn.microsoft.com/en-us/nuget/concepts/dependency-resolution



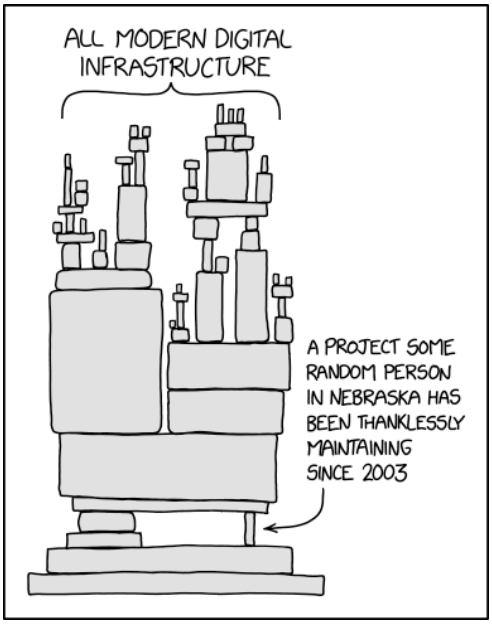
As you can tell, you should use SemVer in your NuGet packages. There's a new feature that will help with version changes – I'll show that later.

https://semver.org/

We want to know what our references are, because, while NuGet will do some verification steps, there's always a chance a package will do nefarious things.

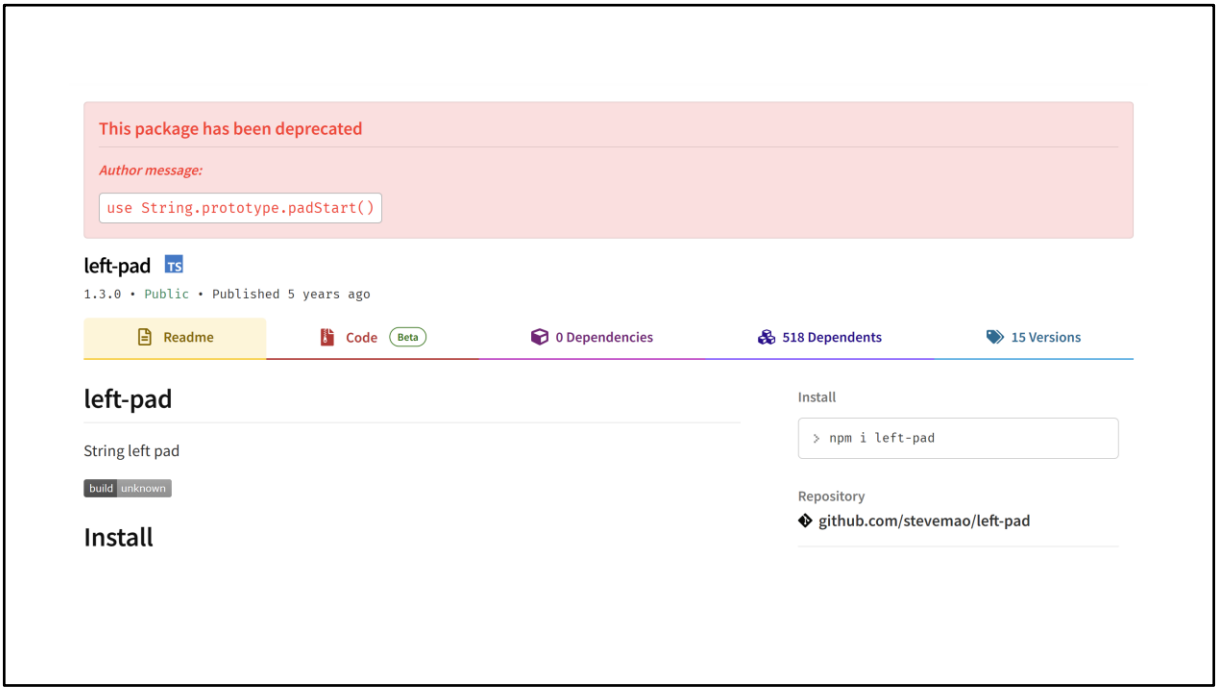
```
* Software Bill of Materials - SBOMs, security, licensing
  * https://www.cisa.gov/sbom
  * https://www.rezilion.com/blog/software-bill-of-materials-guide/
```

https://unsplash.com/photos/padlock-on-black-metal-fence-Dd6n63H9szw



Not only can dependencies conflict or contain security issues, you may end up with a dependency that can bring everything down if someone decides to stop doing all work.

https://xkcd.com/2347/



If you never heard of the left-pad incident, this is a great example of this kind of fragile dependency.

https://www.npmjs.com/package/left-pad
https://www.davidhaney.io/npm-left-pad-have-we-forgotten-how-to-program/

A more recent example of a complicated, years-long effort to add a backdoor to Linux via the xz utility

https://en.wikipedia.org/wiki/XZ_Utils_backdoor

Packages

Building Fabulous NuGet Packages

So, how do we actually create a NuGet package? What are all the options? What are the different types of packages? Designing a “good” package makes it appealing for others to try.

https://unsplash.com/photos/PxM8aeJbzvk

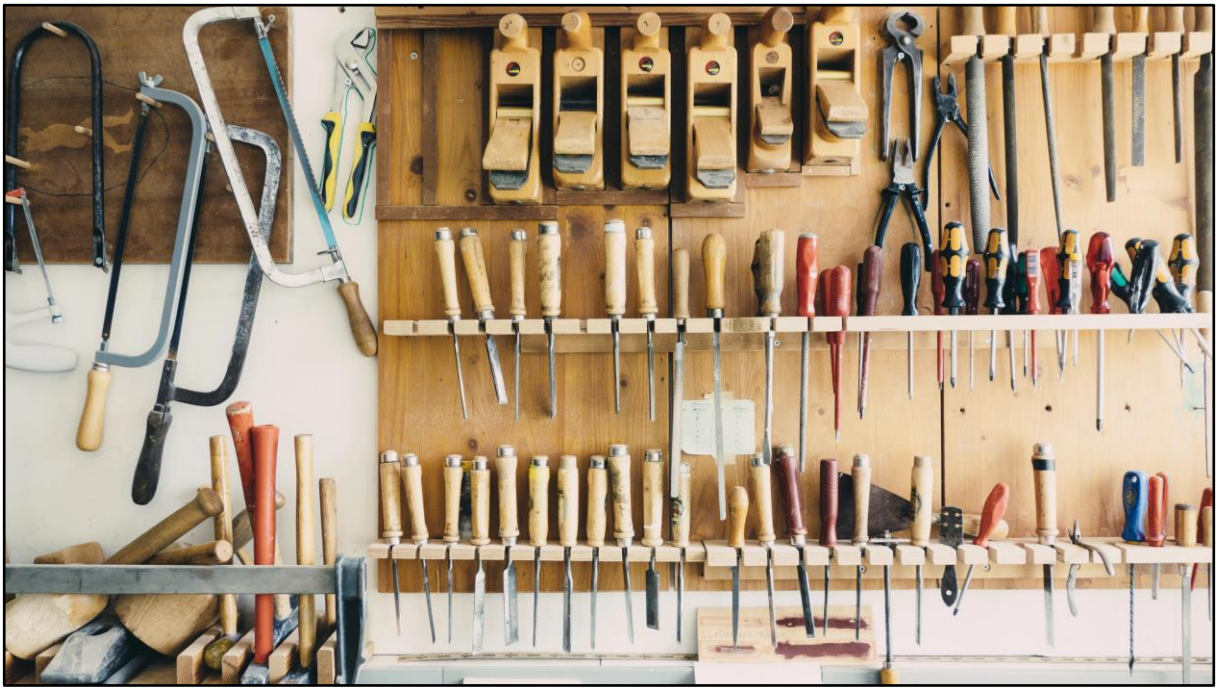
Demo: Building Packages

Building Fabulous NuGet Packages

Let’s go through my Collatz and UsingUsings packages. File formats (rename to .zip), metadata to add, changelogs, readme, icons, snupkgs, referencing, debugging, versioning, etc.

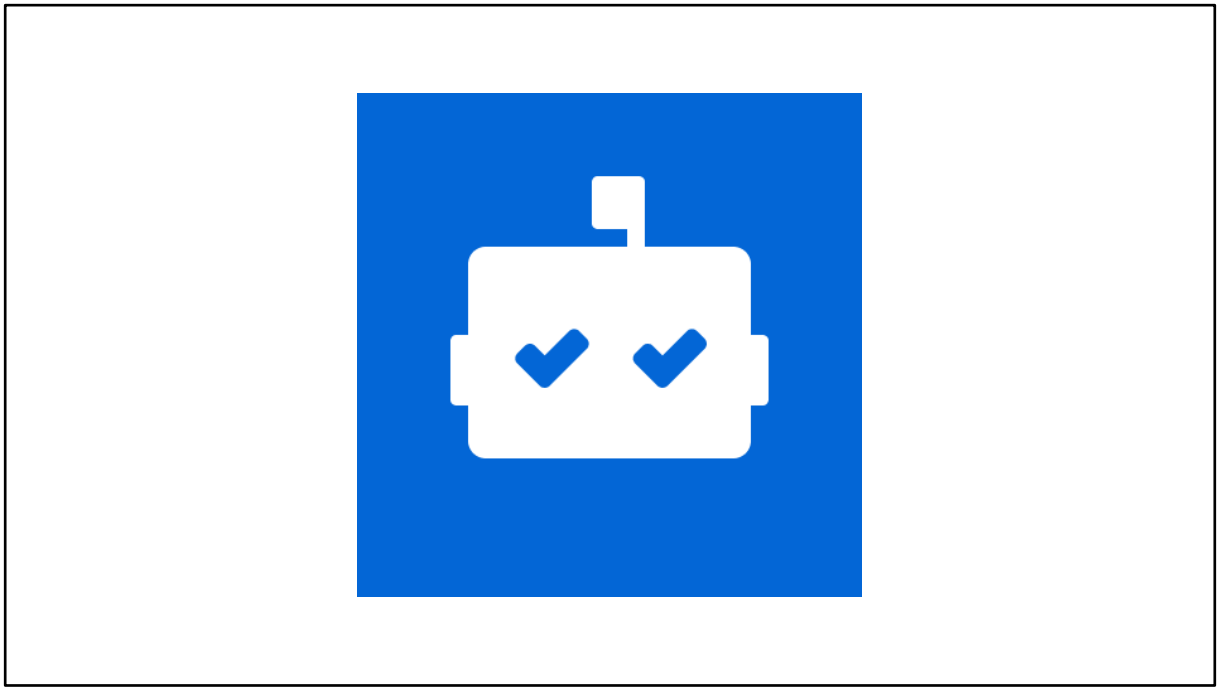
Tools

Building Fabulous NuGet Packages



There are a lot of tools available that can help you with NuGet packages.

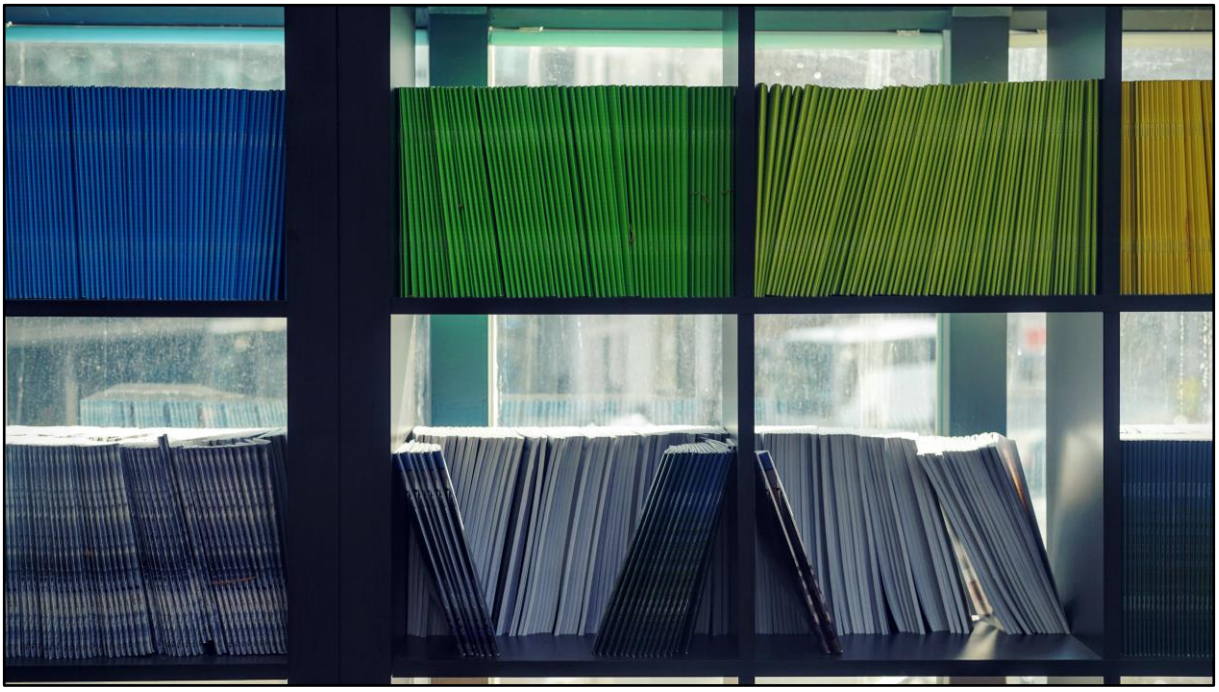
https://unsplash.com/photos/t5YUoHW6zRo



One that I won’t demo, but you should be using not just for NuGet packages, but really any repo, is Dependabot. It’ll inform you with PRs when dependencies have changed, which, as a NuGet author, is great to know, especially when security vulnerabilities arise.

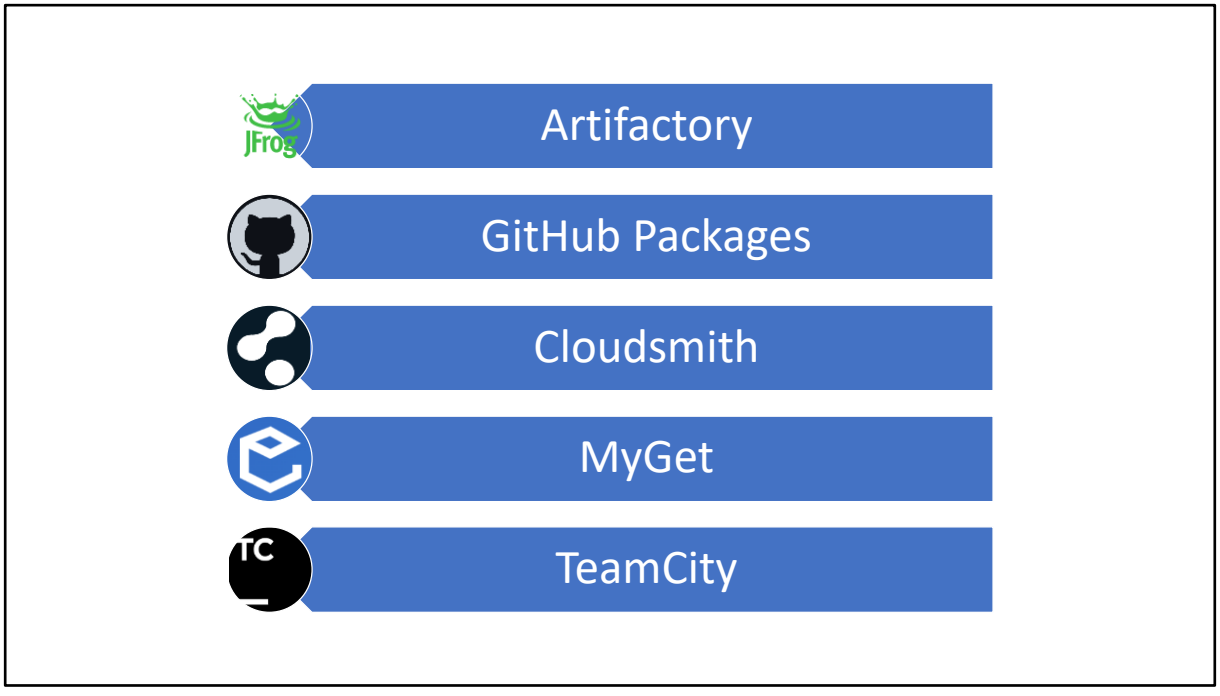
There are also other ways to serve NuGet packages. It’s not just nuget.org.

https://unsplash.com/photos/hEC6zxdFF0M



You can always host a NuGet server simply off of a local feed using the file system. While this is simple, you'll probably only want to use this for testing purposes.

https://learn.microsoft.com/en-us/nuget/hosting-packages/local-feeds
https://unsplash.com/photos/8EzNkvLQosk



Here are some other NuGet servers you can use. No advertising, just a list of choices:

- https://jfrog.com/artifactory/
- https://docs.github.com/en/packages/working-with-a-github-packages-registry/working-with-the-nuget-registry
- https://cloudsmith.com/nuget-feed/
- https://myget.org/
- https://www.jetbrains.com/teamcity/

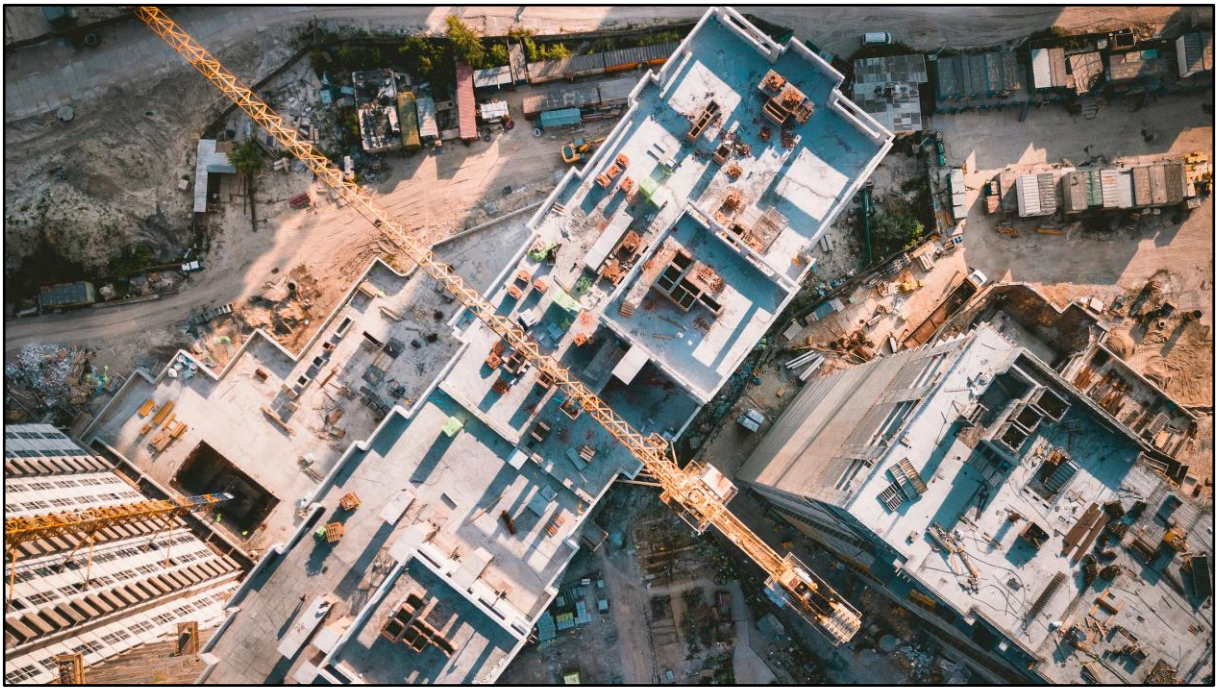
Demo: Using Tools

Building Fabulous NuGet Packages

Let’s look at different tools, Fuget, NuGet package explorer (both desktop and nuget.info), Dependabot

Call to Action

Building Fabulous NuGet Packages



Start building your own packages! And look at how you can use modern NuGet features to simplify your project setups.

https://unsplash.com/photos/0-no6ywKMPY

Building Fabulous NuGet Packages

Jason Bock

Staff Software Engineer

Rocket Mortgage

Remember…

- https://github.com/JasonBock/WhatsNewInCSharp11
- https://github.com/JasonBock/Presentations
- References in the notes on this slide

References

* [C# language standard](https://github.com/dotnet/csharpstandard)
* [Features Added in C# Language Versions](https://github.com/dotnet/csharplang/blob/master/Language-Version-History.md)
* [Language Feature Status](https://github.com/dotnet/roslyn/blob/master/docs/Language%20Feature%20Status.md)
* [C# Guide](https://docs.microsoft.com/en-us/dotnet/csharp/)
* What's the strangest corner case you've seen in C# or .NET?](https://stackoverflow.com/questions/194484/whats-the-strangest-corner-case-youve-seen-in-c-sharp-or-net)
* [Eliminating Nulls in C#](https://magenic.com/thinking/eliminating-nulls-in-c)
* [Language Version Planning](https://github.com/dotnet/csharplang/projects/4)
* [Language Design Meetings](https://github.com/dotnet/csharplang/tree/master/meetings)

C# 11
* [Regular Expression Improvements in .NET 7](https://devblogs.microsoft.com/dotnet/regular-expression-improvements-in-dotnet-7/)
* [What's new in C# 11](https://docs.microsoft.com/en-us/dotnet/csharp/whats-new/csharp-11)
* [Early peek at C# 11 features](https://devblogs.microsoft.com/dotnet/early-peek-at-csharp-11-features)
* [.NET 7 Preview 5 – Generic Math](https://devblogs.microsoft.com/dotnet/dotnet-7-generic-math/)

C# 10
* [C# 10 - `record struct` Deep Dive & Performance Implications](https://nietras.com/2021/06/14/csharp-10-record-struct/)

C# 9
* [C# 9 - Improving performance using the SkipLocalsInit attribute](https://www.meziantou.net/csharp-9-improve-performance-using-skiplocalsinit.htm)
* [JsonSrcGen + CoreRT = Pure Magic](https://trampster.blogspot.com/2020/09/jsonsrcgen-corert-pure-magic-in-my.html)
* [Reduce mental energy with C# 9](https://daveabrock.com/2020/06/18/reduce-mental-energy-with-c-sharp)
* Source Generator
    * [Introducing C# Source Generators - https://devblogs.microsoft.com/dotnet/introducing-c-source-generators/
    * Source Generators Cookbook - https://github.com/dotnet/roslyn/blob/master/docs/features/source-generators.cookbook.md
    * Demos - https://github.com/dotnet/roslyn-sdk/tree/master/samples/CSharp/SourceGenerators
    * INotifyPropertyChanged with C# 9.0 Source Generators - https://jaylee.org/archive/2020/04/29/notify-property-changed-with-rosyln-generators.html
    * C# Source Generators: Less Boilerplate Code, More Productivity - https://dontcodetired.com/blog/post/C-Source-Generators-Less-Boilerplate-Code-More-Productivity
    * Using C# Source Generators with Microsoft Feature Management Feature Flags - https://dontcodetired.com/blog/post/Using-C-Source-Generators-with-Microsoft-Feature-Management-Feature-Flags

* StackOnlyJsonParser - https://github.com/TomaszRewak/C-sharp-stack-only-json-parser
* First Look: C# Source Generators - https://daveabrock.com/2020/05/08/first-look-c-sharp-generators
* Source Generators in C#](https://channel9.msdn.com/Shows/Visual-Studio-Toolbox/Source-Generators-in-CSharp

C#8
* Nullable reference types
* https://github.com/dotnet/roslyn/blob/master/docs/features/nullable-reference-types.md
https://github.com/dotnet/roslyn/blob/master/docs/features/nullable-metadata.md
* https://docs.microsoft.com/en-us/dotnet/csharp/language-reference/proposals/csharp-8.0/nullable-reference-types-specification
* .NET Docs - https://docs.microsoft.com/en-us/dotnet/csharp/nullable-references
* Embracing nullable reference types - https://devblogs.microsoft.com/dotnet/embracing-nullable-reference-types/
* Reserved attributes contribute to the compiler's null state static analysis - https://docs.microsoft.com/en-us/dotnet/csharp/language-reference/attributes/nullable-analysis
* Ian Griffiths series - https://endjin.com/blog/2020/04/dotnet-csharp-8-nullable-references-non-nullable-is-the-new-default
* C# 8 Interfaces
* Static Members - https://jeremybytes.blogspot.com/2019/12/c-8-interfaces-static-members.html
* Public, Private, and Protected Members - https://jeremybytes.blogspot.com/2019/11/c-8-interfaces-public-private-and.html
* Dangerous Assumptions in Default Implementation - https://jeremybytes.blogspot.com/2019/09/c-8-interfaces-dangerous-assumptions-in.html
* A Closer Look at C# 8 Interfaces - https://jeremybytes.blogspot.com/2019/09/a-closer-look-at-c-8-interfaces.html
* Interfaces in C# 8 are a Bit of a Mess - https://jeremybytes.blogspot.com/2019/09/interfaces-in-c-8-are-bit-of-mess.html
Essential C# 8.0 - https://www.codemag.com/article/1911112

C#, Span and async - https://blogs.endjin.com/2019/10/c-span-and-async/
C# 8.0 and .NET Standard 2.0 - Doing Unsupported Things - https://stu.dev/csharp8-doing-unsupported-things/
"deferring" in C#8 - https://twitter.com/reubenbond/status/1184135702851678208
What's New in C#8 - https://docs.microsoft.com/en-us/dotnet/csharp/whats-new/csharp-8
This is how you get rid of null reference exceptions forever - https://channel9.msdn.com/Shows/On-NET/This-is-how-you-get-rid-of-null-reference-exceptions-forever
Alignment with C# 8 Switch Expressions - https://csharp.christiannagel.com/2019/09/17/switchexpressionalign/
Async Streams – A Look at New Language Features in C# 8 - https://blog.jetbrains.com/dotnet/2019/09/16/async-streams-look-new-language-features-c-8/
Eliminating Nulls in C# - https://magenic.com/thinking/eliminating-nulls-in-c
DIM
Default Interface Members and Inheritance - https://daveaglick.com/posts/default-interface-members-and-inheritance
Default Interface Members, What Are They Good For? - https://daveaglick.com/posts/default-interface-members-what-are-they-good-for
Update libraries to use nullable reference types and communicate nullable rules to callers - https://docs.microsoft.com/en-us/dotnet/csharp/nullable-attributes
Try out Nullable Reference Types - https://devblogs.microsoft.com/dotnet/try-out-nullable-reference-types/
C# 8 Pattern Matching Enhancements - https://www.infoq.com/news/2019/06/CSharp-8-Pattern-Matching/
Tutorial: Using pattern matching features to extend data types - https://docs.microsoft.com/en-us/dotnet/csharp/tutorials/pattern-matching
Default implementations in interfaces - https://devblogs.microsoft.com/dotnet/default-implementations-in-interfaces/
C#8, .NET Framework and .NET Core Divergence - https://twitter.com/dotMorten/status/1105343335164596224?s=09
Async Streams with C#8 - https://csharp.christiannagel.com/2019/03/20/asyncstreams/
.NET Core 3.0 vs. .NET Framework - https://twitter.com/dotMorten/status/1105343335164596224?s=09
Much Ado about Nothing: A C# play in two acts.
Part 1 - https://www.youtube.com/watch?v=GusJQNjj_2g
Part 2 - https://www.youtube.com/watch?v=lVDYwA-BcwE
Adapting Projects to Use C# 8 and Nullable Reference Types -

https://www.infoq.com/articles/csharp-nullable-reference-case-study
NullableAttribute and C#8 -
https://codeblog.jonskeet.uk/2019/02/10/nullableattribute-and-c-8/
Update on IAsyncDisposable and IAsyncEnumerator -
https://www.infoq.com/news/2019/01/IAsyncDisposable-IAsyncEnume
Do more with patterns in C# 8.0 -
https://blogs.msdn.microsoft.com/dotnet/2019/01/24/do-more-with-patterns-in-c-8-0/
Take C# 8.0 for a spin -
https://blogs.msdn.microsoft.com/dotnet/2018/12/05/take-c-8-0-for-a-spin/
Containing Null with C# 8 Nullable References -
https://praeclarum.org/2018/12/17/nullable-reference-types.html
C# 8: Pattern Matching Extended -
https://csharp.christiannagel.com/2018/07/03/patternmatchingcs8/
Writing operators - https://github.com/akarnokd/async-enumerable-dotnet/wiki/Writing-operators
Optimizing C# Struct Equality with IEquatable and ValueTuples -
https://montemagno.com/optimizing-c-struct-equality-with-iequatable/
C# 8 - Jon Skeet and Mads Torgersen -
https://www.youtube.com/watch?v=gGUYUJmssYM
The future of C# (Build 2018) -
https://channel9.msdn.com/Events/Build/2018/BRK2155
First steps with nullable reference types -
https://codeblog.jonskeet.uk/2018/04/21/first-steps-with-nullable-reference-types/
A Preview of C# 8 with Mads Torgersen -
https://channel9.msdn.com/Blogs/Seth-Juarez/A-Preview-of-C-8-with-Mads-Torgersen
Introducing Nullable Reference Types in C# -
https://blogs.msdn.microsoft.com/dotnet/2017/11/15/nullable-reference-types-in-csharp/
Herding Nulls and Other C# Stories From the Future -
https://www.infoq.com/presentations/c-sharp-future

Before C#8
How Microsoft rewrote its C# compiler in C# and made it open source -
https://medium.com/microsoft-open-source-stories/how-microsoft-rewrote-its-c-compiler-in-c-and-made-it-open-source-4ebed5646f98
Tuples, deconstruction, and dictionaries -
https://twitter.com/davidpine7/status/1032706884569059328
Interesting uses of tuple deconstruction -

https://compiledexperience.com/blog/posts/abusing-tuples
Performance traps of ref locals and ref returns in C# - https://blogs.msdn.microsoft.com/seteplia/2018/04/11/performance-traps-of-ref-locals-and-ref-returns-in-c/
Memory Management, C# 7.2 and Span<T> - https://speakerdeck.com/slang25/memory-management-c-number-7-dot-2-and-span
Using In Parameter Modifier - C# 7.2 - https://codewala.net/2018/03/27/using-in-parameter-modifier-c-7-2/
Essential .NET - C# 8.0 and Nullable Reference Types - https://msdn.microsoft.com/en-us/magazine/mt829270.aspx
C# 7 Series, Part 6: Read-only structs - https://blogs.msdn.microsoft.com/mazhou/2017/11/21/c-7-series-part-6-read-only-structs/
What's new in C# 7.2 - What's new in C# 7.2
What's new in C# 7.2 - https://docs.microsoft.com/en-us/dotnet/csharp/whats-new/csharp-7-2
C# 7.2: Understanding Span - https://channel9.msdn.com/Events/Connect/2017/T125
C# Language Internals - https://www.youtube.com/watch?v=1lnwO63LhRI
Dissecting the local functions in C# 7 - https://blogs.msdn.microsoft.com/seteplia/2017/10/03/dissecting-the-local-functions-in-c-7/
C# 7.1 - Everything You Need To Know - https://www.danielcrabtree.com/blog/329/c-sharp-7-1-everything-you-need-to-know
C# 7, 8 and Beyond: Language Features from Design to Release to IDE Support - https://www.infoq.com/presentations/csharp-7-8-language-features
Perusing C# 7.1 - http://davidpine.net/blog/csharp-seven-dot-one/
Exploring C# 7 - http://davidpine.net/blog/exploring-csharp-seven/
Tuples and Generics - https://magenic.com/thinking/tuples-and-generics-in-c7
Tuples and Deconstruction in C#7 - https://magenic.com/thinking/tuples-and-deconstruction-in-c7
C#7 Series
Value Tuples: https://blogs.msdn.microsoft.com/mazhou/2017/05/26/c-7-series-part-1-value-tuples/
Async Main: https://blogs.msdn.microsoft.com/mazhou/2017/05/30/c-7-series-part-2-async-main/
Default Literals:

https://blogs.msdn.microsoft.com/mazhou/2017/06/06/c-7-series-part-3-default-literals/

Discards:
https://blogs.msdn.microsoft.com/mazhou/2017/06/27/c-7-series-part-4-discards/

Private Protected:
https://blogs.msdn.microsoft.com/mazhou/2017/10/05/c-7-series-part-5-private-protected/

Read-Only Structs:
https://blogs.msdn.microsoft.com/mazhou/2017/11/21/c-7-series-part-6-read-only-structs/

Ref Returns:
https://blogs.msdn.microsoft.com/mazhou/2017/12/12/c-7-series-part-7-ref-returns/

"in" Parameters:
https://blogs.msdn.microsoft.com/mazhou/2018/01/08/c-7-series-part-8-in-parameters/

Ref Structs:
https://blogs.msdn.microsoft.com/mazhou/2018/03/02/c-7-series-part-9-ref-structs/

Span<T> and universal memory management:
https://blogs.msdn.microsoft.com/mazhou/2018/03/25/c-7-series-part-10-spant-and-universal-memory-management/

Expression – Bodied Members in C# 7.0 - http://dailydotnettips.com/2017/07/17/expression-bodied-members-in-c-7-0/

Abolishing Switch-Case Statement and Pattern Matching in C# 7.0: https://rubikscode.net/2017/06/18/abolishing-switch-case-and-pattern-matching-in-c-7-0/

C# 7 – Simplify and Improve Your Code in 2017: https://channel9.msdn.com/Events/DEVintersection/DEVintersection-2017-Orlando/DEV005

C# 7.x and 8.0: Uncertainty and Awesomeness: https://www.erikheemskerk.nl/c-sharp-7-2-and-8-0-uncertainty-awesomeness/

C# 7.1 and Beyond: Polishing Usability - https://www.erikheemskerk.nl/c-sharp-7-1-polishing-usability/

C# 7.2 and 8.0 Roadmap - https://www.infoq.com/news/2017/06/CSharp-7.2

An Early Look at C# 7.1: Part 1 - https://www.infoq.com/news/2017/06/CSharp-7.1-a

An Early Look at C# 7.1: Part 2 - https://www.infoq.com/news/2017/06/CSharp-7.1-b

The Future of C# - https://channel9.msdn.com/Events/Build/2017/B8104

What's new in C# 7 - https://docs.microsoft.com/en-us/dotnet/articles/csharp/whats-new/csharp-7
Patterns and Practices in C# 7 - https://www.infoq.com/articles/Patterns-Practices-CSharp-7
New Features in C# 7.0 - https://blogs.msdn.microsoft.com/dotnet/2017/03/09/new-features-in-c-7-0/
C#7 Tools (see other related articles from this author) - http://blog.somewhatabstract.com/2017/02/20/c7-tools/
.NET Framework - What's New in C# 7.0 - https://msdn.microsoft.com/en-us/magazine/mt790184.aspx
Nested Functions: https://en.wikipedia.org/wiki/Nested_function (think function to delete files, needs recusion, local functions makes this elegant, but is it performant?)
C# 7 Work List of Features - https://github.com/dotnet/roslyn/issues/2136
What's New in C#7 - https://blogs.msdn.microsoft.com/dotnet/2016/08/24/whats-new-in-csharp-7-0/
Visual Studio "15" Preview 5 - https://www.visualstudio.com/news/releasenotes/vs15-relnotes#cshappvb