Windows and .NET Framework Catalog

Priority Guide for a C# Terminal API Browser and Object Workbench

Organized source list, framework descriptions, cataloging roles, and suggested ingestion order

Working definition
This guide treats a framework source as any authoritative API surface your application can index: .NET reference assemblies, Windows SDK metadata, WinRT metadata, native Win32 metadata, COM type libraries, installed assemblies, SDK packages, and selected third-party NuGet packages.

1. Executive summary

- The most valuable first milestone is a managed .NET object workbench: index .NET core libraries, WPF, Windows Forms, Windows App SDK/WinUI, Windows Runtime projections, PowerShell, Roslyn, MSBuild, NuGet, and ASP.NET Core.
- The Windows SDK and Win32 metadata are irreplaceable for Windows coverage, but they require a separate native/interop catalog model. Many Win32 entries are functions, structs, handles, callbacks, and COM interfaces rather than normal C# classes.
- Reference packs such as Microsoft.NETCore.App.Ref and Microsoft.WindowsDesktop.App.Ref are not user-facing frameworks. They are scanner inputs that describe the public API surface for a framework version.
- Your UI should sort by official platform/product first, then hide package-level metadata names inside a Source Details panel.
- Secondary libraries should be prioritized only when they add useful managed wrappers, better ergonomics, or real automation value beyond the official metadata.

2. Ranking scale

Rank	Meaning	Use
S	Irreplaceable foundation	Without this, the app cannot be a serious .NET/Windows API workbench.
A	High-value core	Strongly expands usefulness for terminal automation, Windows app work, and real code generation.

Rank	Meaning	Use
B	Important expansion	Useful after the core catalog is stable; often specialized or workflow-dependent.
C	Later or niche	Valuable for completeness, but not essential for the first production-quality version.

3. The corrected hierarchy

> **User-facing names vs scanner names**
> Do not expose reference-pack IDs as primary UI categories. Show users official platforms and framework names. Store the exact package, target framework, assembly, and metadata source behind each API record.

Layer	Examples	Catalog role
Official platform/product	.NET 10, Windows SDK, Windows App SDK, PowerShell 7	Top-level user-facing grouping.
API family/framework	Core libraries, WPF, WinUI 3, Win32, WinRT, DirectX	What the user probably thinks they are browsing.
Metadata source	Microsoft.NETCore.App.Ref, Windows.winmd, Win32Metadata	What your indexer reads.
Assembly or metadata file	System.Runtime.dll, PresentationFramework.dll, Windows.winmd	Physical unit that contains type/member metadata.
Namespace	System.IO, Microsoft.UI.Xaml, Windows.Storage	Logical organization inside assemblies/metadata.
Type/member	FileStream, Window, StorageFile, CreateFileW	The searchable API item.

4. Master priority list

#	Framework/source	Rank	Description	Indexer/execution note
1	Modern .NET core libraries	S	Primary managed object surface: strings, collections, I/O, networking, reflection, threading, JSON, diagnostics, tasks, security, and runtime primitives.	Microsoft.NETCore.App.Ref for cataloging; installed Microsoft.NETCore.App runtime for execution.

#	Framework/source	Rank	Description	Indexer/execution note
2	Windows SDK - Win32, COM, DirectX, system APIs	S	Official Windows platform API surface: process, file, registry, shell, security, networking, devices, UI, graphics, audio, media, diagnostics, and COM.	Windows SDK installation plus Win32Metadata; use generated interop or wrappers for C# execution.
3	Windows Desktop .NET - WPF and Windows Forms	S	Managed desktop UI frameworks with many instantiable controls, windows, drawing, input, automation, and document APIs.	Microsoft.WindowsDesktop.App.Ref and corresponding runtime/shared framework.
4	Windows App SDK and WinUI 3	S	Modern Windows desktop app platform: WinUI 3 controls, app lifecycle, windowing, notifications, deployment, resources, and modern Windows features.	Microsoft.WindowsAppSDK NuGet assets and WinUI assemblies.
5	Windows Runtime APIs for .NET	S	Managed projection over WinRT APIs such as storage, devices, sensors, media, launcher, notifications, and app model capabilities.	Microsoft.Windows.SDK.NET.Ref and Windows SDK metadata.
6	.NET Framework reference assemblies	A	Legacy Windows-only .NET surface: System.Web, WCF, Workflow, Remoting, EnterpriseServices, legacy WPF/WinForms, and compatibility APIs.	Versioned .NET Framework targeting/reference packs.
7	PowerShell SDK	A	Runspaces, pipelines, cmdlet development, providers, AST/parsing, remoting, streams, and shell automation APIs.	System.Management.Automation and installed PowerShell assemblies.
8	Roslyn compiler platform	A	C# parsing, compilation, syntax trees, semantic models, analyzers, scripting, code generation, and dynamic code evaluation.	Microsoft.CodeAnalysis packages and assemblies.
9	MSBuild and NuGet Client SDK	A	Project evaluation, build execution, package restore, dependency resolution, target framework analysis, and NuGet package ingestion.	Microsoft.Build.* and NuGet.* packages.
10	Microsoft.Extensions and ASP.NET Core	A	Dependency injection, hosting, configuration, logging, options, middleware, Kestrel, HTTP abstractions, SignalR, Razor, and Blazor.	Microsoft.AspNetCore.App.Ref plus Microsoft.Extensions packages.

#	Framework/source	Rank	Description	Indexer/execution note
11	WebView2	A	Embed Chromium/Edge web content in WPF, WinForms, WinUI, and native apps; useful for terminal side panels and app preview surfaces.	Microsoft.Web.WebView2 SDK.
12	Entity Framework Core	B	Official Microsoft data access framework; useful for database automation, schema inspection, and object querying.	Microsoft.EntityFrameworkCore packages and provider packages.
13	.NET MAUI - Windows target	B	Cross-platform UI framework that uses WinUI/Windows App SDK on Windows. Useful for completeness, less central to a Windows-only terminal app.	Microsoft.Maui packages and Windows target assets.
14	UWP and WinUI 2	B	Important for existing/legacy Store and UWP projects. Less central for new desktop apps than Windows App SDK/WinUI 3.	Windows SDK contracts and WinUI 2 packages.
15	Windows Driver Kit	C	Driver development headers, libraries, metadata, templates, tools, verification, signing, deployment, KMDF/UMDF/WDM models.	WDK installation and WDK metadata; mostly code generation and docs rather than CLR instantiation.
16	Visual Studio SDK	C	Visual Studio extension, tooling, shell integration, project system, and design-time automation APIs.	Visual Studio SDK packages and installed assemblies.

5. Official Microsoft and Windows catalog sources

5.1 Core managed runtime and compatibility

Item	Rank	Brief description	Primary metadata source	Why it matters
Modern .NET	S	The current Microsoft-managed runtime family for C#, F#, and VB. Provides the Base Class Library: collections, I/O, networking, reflection, threading, tasks, JSON, diagnostics, security, and runtime services.	Microsoft.NETCore.App.Ref, Microsoft.NETCore.App runtime, SDK packs	Primary starting point for object browsing and invocation.

Item	Rank	Brief description	Primary metadata source	Why it matters
.NET Standard	B	API compatibility contract used by many libraries, especially older packages targeting netstandard2.0. It is a contract, not a runtime.	NETStandard.Library.Ref	Useful for compatibility queries and package targeting, but overlaps with modern .NET.
.NET Framework	A	Original Windows-only .NET runtime family. Includes APIs absent or different in modern .NET: System.Web, WCF, Workflow, Remoting, EnterpriseServices, legacy WinForms/WPF.	Versioned .NET Framework reference assemblies	Important if your terminal indexes legacy Windows applications.
Microsoft.Extensions	A	Common app infrastructure used across console, desktop, web, and service apps: dependency injection, hosting, logging, configuration, options, caching, HTTP client factory.	Microsoft.Extensions.* packages	High practical value for automation recipes and object graph construction.

5.2 Windows desktop UI and app platforms

Item	Rank	Brief description	Primary metadata source	Why it matters
Windows App SDK	S	Current Windows desktop application platform delivered independently of the OS. Includes WinUI 3 and app platform capabilities such as lifecycle, windowing, notifications, deployment, resources, and modern Windows features.	Microsoft.WindowsAppSDK NuGet assets	Essential for modern Windows desktop development.
WinUI 3	S	Modern native Windows UI framework in the Windows App SDK. Provides XAML controls, styling, layout, composition integration, and Fluent design-aligned UI.	Microsoft.UI.Xaml and related Windows App SDK assemblies	Top choice for new Windows desktop UI catalog coverage.
WPF	S	Mature XAML desktop framework with windows, controls, data binding, styles, templates, documents, graphics, animation, and rich composition.	Microsoft.WindowsDesktop.App.Ref; PresentationFramework, PresentationCore, WindowsBase	Extremely useful for object inspection, UI experiments, and desktop automation.
Windows Forms	S	Classic managed desktop UI framework with forms, controls, events, designers, dialogs, drawing, and simple windowed app development.	Microsoft.WindowsDesktop.App.Ref; System.Windows.Forms	Highly instantiable and easy to demonstrate in an object workbench.

Item	Rank	Brief description	Primary metadata source	Why it matters
UWP	B	Universal Windows Platform app model and WinRT-centered UI/application framework. Still relevant for existing Store/UWP apps but not the main path for new desktop apps.	Windows SDK contracts; UWP/WinUI 2 metadata	Include for compatibility and legacy project coverage.
WinUI 2	B	WinUI version for UWP apps, distinct from WinUI 3. Provides controls/styles for UWP projects.	Microsoft.UI.Xaml UWP packages	Separate from WinUI 3 in the catalog.
.NET MAUI - Windows	B	Cross-platform .NET UI framework; on Windows it uses WinUI and Windows App SDK underneath.	Microsoft.Maui.* packages	Useful if the app indexes cross-platform .NET UI projects.
WebView2	A	Microsoft Edge/Chromium embedding SDK for WPF, WinForms, WinUI, and native desktop apps.	Microsoft.Web.WebView2	Very valuable for terminal preview panes, docs panels, HTML output, and hybrid UI.

5.3 Windows SDK API families

Important model change

Most Windows SDK APIs are not CLR classes. Catalog them as native functions, structs, enums, COM interfaces, handles, callbacks, WinRT classes, tools, or build assets. Execution usually requires WinRT projection, COM interop, generated P/Invoke, or a wrapper library.

API family	Rank	Brief description	Metadata/source	Catalog value
Win32 core APIs	S	Classic native Windows API surface for process, thread, memory, file, registry, service, console, DLL loading, synchronization, and OS information.	Windows SDK headers/libs; Win32Metadata	Irreplaceable for Windows automation and low-level system operations.
COM and OLE	S	Binary component object model used throughout Windows. Includes interfaces, coclasses, type libraries, activation, marshaling, automation, and shell integration.	Type libraries, registry, Win32Metadata, Windows SDK	Critical for Office, Shell, UI Automation, legacy Windows, and app interoperability.
Windows Runtime - WinRT	S	Modern Windows API model with metadata-described classes, interfaces, structs, enums, events, and async operations.	Windows.winmd, Microsoft.Windows.SDK.NET.Ref	Closest native Windows API family to a managed object model.
DirectX family	A	Graphics, compute, display, text, imaging, audio, input, and GPU APIs: Direct3D, Direct2D, DirectWrite, DXGI, DirectML, XAudio2, DirectStorage.	Windows SDK headers/libs; DirectX metadata/wrappers	Essential for graphics/media tooling and specialized hardware acceleration.

API family	Rank	Brief description	Metadata/source	Catalog value
Shell and desktop integration	A	Taskbar, Start/menu integration, file associations, property system, known folders, shell items, thumbnails, drag/drop, clipboard, notifications.	Windows SDK, COM, WinRT, Windows App SDK	High-value for a terminal that interacts deeply with Windows desktop.
UI Automation and accessibility	A	Automation tree, controls, patterns, events, accessibility, input automation, and app testing integration.	UIAutomation assemblies, COM interfaces, Windows SDK	Useful for testing, automation, and app inspection.
Networking APIs	A	Winsock, WinHTTP, WinINet, HTTP Server API, IP Helper, DNS, WLAN, Bluetooth, WebSockets, network list/location APIs.	Windows SDK headers/libs; .NET network APIs	Important for terminal diagnostics and network automation.
Security and identity	A	Cryptography, certificate stores, DPAPI, SSPI, ACLs, tokens, credentials, smart cards, Windows Hello, authorization.	Windows SDK; CNG/CryptoAPI; .NET security APIs	Required for serious system utilities and secure automation.
Storage and file-system APIs	A	Volumes, files, NTFS features, VSS, Cloud Files, ProjFS, virtual disks, storage management, packaging, BITS.	Windows SDK, COM, WinRT	Useful for file-system browser features and automation.
Diagnostics and management	A	ETW, TraceLogging, Event Log, WMI/CIM, performance counters, dump/debug APIs, Windows Error Reporting.	Windows SDK, WMI metadata, .NET diagnostics APIs	Excellent fit for a terminal framework library.
Devices and hardware	B	SetupAPI, Configuration Manager, HID, USB, sensors, location, printing, display, power, portable devices, Bluetooth.	Windows SDK; WDK for some areas	Useful, but many calls are native and device-specific.
Media and audio	B	Media Foundation, Core Audio/WASAPI, MIDI, speech, capture, encoding, playback, session control.	Windows SDK; WinRT media APIs	Important for media tools but specialized.

5.4 Developer platforms and automation

Item	Rank	Brief description	Primary metadata source	Why it matters
PowerShell SDK	A	Managed API for runspaces, pipelines, cmdlets, providers, remoting, streams, jobs, parsing, AST inspection, and session state.	System.Management.Automation and installed PowerShell assemblies	Directly relevant to a custom terminal.
Roslyn	A	Compiler platform for C# and VB: syntax trees, semantic models, analyzers, refactoring, scripting, code generation, and compilation.	Microsoft.CodeAnalysis packages	Enables generated-code preview, inline C# scripting, and type-aware assistance.

Item	Rank	Brief description	Primary metadata source	Why it matters
MSBuild	A	Build engine and project evaluation API. Reads/evaluates projects, items, properties, targets, loggers, and build execution.	Microsoft.Build.* assemblies	Needed to understand project references and resolve real assemblies.
NuGet Client SDK	A	Package discovery, download, dependency resolution, framework compatibility, package metadata, and restore primitives.	NuGet.Protocol, NuGet.Packaging, NuGet.Frameworks, NuGet.ProjectModel	Needed for secondary-source expansion.
Visual Studio SDK	C	Extension and IDE automation APIs for Visual Studio shell, packages, project system, tool windows, commands, and design-time services.	Visual Studio SDK packages and installed assemblies	Valuable only if integrating with Visual Studio.
System.CommandLine	B	Microsoft command-line parser and invocation model for .NET console applications.	System.CommandLine package	Useful for command generation, wrappers, and internal CLI tooling.

5.5 Advanced and specialized Microsoft sources

Item	Rank	Brief description	Primary source	Catalog value
Windows Driver Kit - WDK	C	Toolchain and API set for building, testing, signing, and deploying drivers. Includes KMDF, UMDF, WDM, device-class technologies, headers, libs, tools, templates, and metadata.	WDK installation and WDK metadata	Mostly a code-generation/reference section; not ordinary CLR object instantiation.
ML.NET	B	Microsoft machine-learning framework for .NET: data pipelines, transforms, trainers, model loading, prediction engines.	Microsoft.ML packages	Useful for data tooling but not core to Windows API coverage.
ONNX Runtime	B	High-performance inference runtime with CPU/GPU providers; useful for local AI inference and Windows ML-adjacent workflows.	Microsoft.ML.OnnxRuntime packages	Useful if the terminal includes AI/vision/media experiments.
.NET Aspire	B	Cloud-native application orchestration, service defaults, dashboards, resource modeling, and local distributed app development.	Aspire packages and templates	Helpful for modern service development but not Windows-specific.
Orleans	C	Distributed virtual actor framework for scalable services.	Microsoft.Orleans packages	Powerful but outside the main Windows/object-workbench goal.

Item	Rank	Brief description	Primary source	Catalog value
Windows Community Toolkit	B	Community/Microsoft-supported helpers, controls, animations, converters, and extensions for Windows app development.	CommunityToolkit.WinUI and related packages	Useful secondary layer over WinUI/Windows App SDK.

6. Secondary sources sorted by usefulness

Selection rule
Secondary sources should either expose a Windows capability more safely than raw P/Invoke, make UI/terminal automation easier, provide broad utility APIs likely to be used interactively, or help your app index/build/run other projects.

6.1 Tier S/A secondary sources

Item	Rank	Brief description	Category	How to use in catalog
Vanara	S	Broad .NET wrappers around Windows P/Invoke functions, COM interfaces, structures, constants, and helper classes. Much easier to browse and use than raw generated signatures.	Windows interop/wrappers	Add after official Win32Metadata; link wrapper APIs to underlying Win32 functions.
TerraFX.Interop.Windows	S	Low-level Windows and DirectX SDK bindings with close-to-native signatures. Excellent for comprehensive and modern native coverage.	Windows interop/bindings	Best for low-level fidelity; less friendly than Vanara.
Vortice.Windows	A	Managed bindings around DirectX, DXGI, WIC, Direct2D, DirectWrite, Direct3D, XAudio, DirectML, DirectStorage, and related APIs.	Graphics/media interop	Essential if graphics/GPU/media APIs matter.
Microsoft.CsWin32	S	Microsoft source generator that creates C# P/Invoke bindings from Win32 metadata.	Official generator	Not a framework to browse by itself, but invaluable execution glue for Win32 calls.
Spectre.Console	A	Rich terminal UI output: tables, trees, panels, prompts, progress, command apps, markup, and formatting.	Terminal UI	Highly relevant to your command library/terminal experience.
Terminal.Gui	A	Full-screen terminal user-interface toolkit for .NET: windows, dialogs, menus, controls, and keyboard-driven UI.	Terminal UI	Useful if your terminal supports TUI panels or embedded tools.

Item	Rank	Brief description	Category	How to use in catalog
FlaUI	A	Managed wrapper over Windows UI Automation for inspecting and controlling desktop applications.	Automation/testing	Strong fit for a Windows automation terminal.
Avalonia	A	Cross-platform XAML-based .NET UI framework with its own rendering engine and desktop/mobile/web targets.	UI framework	Useful if your app or generated tools need cross-platform UI.
Uno Platform	B	Cross-platform UI platform that follows WinUI/XAML-style APIs across desktop, mobile, and web targets.	UI framework	Relevant for WinUI-style cross-platform projects.
ReactiveUI	B	Reactive MVVM framework usable with WPF, WinUI, Avalonia, MAUI, and others.	Application architecture	Useful for cataloging MVVM/event-stream patterns.
Prism	B	Modular application framework with MVVM, navigation, commands, regions, and event aggregation.	Application architecture	Valuable for enterprise desktop app patterns.
SSH.NET	A	SSH, SFTP, SCP, tunneling, shell sessions, and remote command execution.	Remote automation	Very useful in a terminal automation app.
RestSharp	B	HTTP/REST client abstraction with request/response helpers and serialization support.	Networking/API client	Useful for interacting with web APIs quickly.
Refit	B	Type-safe REST client generation from C# interfaces.	Networking/API client	Good generated-code target for API automation.

6.2 Tier B/C secondary sources

Item	Rank	Brief description	Category	How to use in catalog
Dapper	B	Lightweight object mapper for SQL. Simple extension-method-heavy API over IDbConnection.	Data access	Good for database automation recipes.
Newtonsoft.Json	B	Mature JSON serialization, deserialization, dynamic tokens, LINQ-to-JSON, converters, and schema-adjacent utilities.	Serialization	Still common in existing .NET apps.
Serilog	B	Structured logging framework with sinks, enrichers, context, message templates, and configuration support.	Logging	Useful for generated tools and services.
Polly	B	Resilience library: retry, timeout, circuit breaker, fallback, hedging, and rate-limiting strategies.	Resilience	Useful around network and service automation.

Item	Rank	Brief description	Category	How to use in catalog
FluentValidation	B	Strongly typed validation rules and validators for .NET models.	Validation	Helpful for object-workbench input validation patterns.
AutoMapper	C	Object-to-object mapping configuration and projection helpers.	Mapping	Useful in app catalogs but less essential.
MediatR	B	In-process request, command, query, notification, and pipeline behavior pattern.	Application architecture	Common in modern C# projects.
CsvHelper	B	CSV parsing/writing, mapping, culture handling, and streaming-oriented data import/export.	Data files	High utility for terminal data workflows.
SkiaSharp	B	2D graphics drawing, image surfaces, paths, text, and GPU/CPU rendering backed by Skia.	Graphics	Useful for visualizers and generated diagrams.
ImageSharp	B	Managed image loading, processing, transformations, and encoding.	Imaging	Useful for file tools and media utilities.
OpenCvSharp	C	OpenCV bindings for image processing, computer vision, camera frames, and matrix operations.	Computer vision	Powerful but large/specialized.
NAudio	B	Audio playback, recording, MIDI, waveform processing, codecs, and Windows audio API helpers.	Audio/media	Useful for Windows media/audio tooling.
ClosedXML	B	Spreadsheet manipulation over OpenXML with a friendly Excel workbook/worksheet model.	Documents/data	Useful for reports and automation.
QuestPDF	B	Fluent .NET PDF generation library for reports and documents.	Documents/output	Good generated-output target.
Html Agility Pack / AngleSharp	C	HTML parsing, querying, DOM traversal, and web content manipulation.	Parsing/web	Useful later for scraping/local HTML workflows.

7. Recommended ingestion order

Phase	Scope	Implementation focus
Phase 1	Modern .NET core libraries	Index Microsoft.NETCore.App.Ref; build CLR type/member search; support constructor invocation, static methods, properties, async return values, IDisposable tracking.

Phase	Scope	Implementation focus
Phase 2	Windows Desktop .NET	Index Microsoft.WindowsDesktop.App.Ref; add WPF and Windows Forms categories; support STA-thread warnings for UI objects.
Phase 3	Windows App SDK / WinUI 3	Index Windows App SDK assemblies; model package/version/source details; add UI-thread and packaging capability notes.
Phase 4	Windows Runtime projection	Index Microsoft.Windows.SDK.NET.Ref and Windows SDK WinRT metadata; support WinRT async patterns and platform availability.
Phase 5	.NET Framework compatibility	Index versioned reference assemblies; store runtime family separately from modern .NET even when type names match.
Phase 6	PowerShell, Roslyn, MSBuild, NuGet	Add developer automation surfaces that help the terminal inspect projects, run commands, compile code, and ingest packages.
Phase 7	ASP.NET Core and Microsoft.Extensions	Add hosting, dependency injection, logging, configuration, HTTP, middleware, and web-framework APIs.
Phase 8	Win32Metadata, COM, and CsWin32	Add native API catalog branch. Treat functions, structs, handles, callbacks, and COM separately from CLR classes.
Phase 9	Interop wrapper libraries	Add Vanara, TerraFX, Vortice. Link wrapper methods to underlying Win32/DirectX concepts where possible.
Phase 10	High-use secondary frameworks	Add Spectre.Console, Terminal.Gui, FlaUI, SSH.NET, RestSharp, Refit, Dapper, Serilog, Polly, CsvHelper, SkiaSharp/ImageSharp/NAudio.
Phase 11	WDK and specialized systems	Add as advanced reference/code-generation section. Avoid pretending WDF handles are CLR objects.

8. Suggested UI categories

Top-level UI category	Included items	Browsable object model
Microsoft > .NET	Modern .NET, .NET Standard, Microsoft.Extensions, ASP.NET Core, EF Core, MAUI, ML.NET, Aspire, Orleans	Managed classes, structs, interfaces, delegates, attributes, extension methods, generic types.
Microsoft > Windows	Windows SDK, Windows Runtime, Win32, COM, DirectX, Media, Shell, Devices, Storage, Security, Diagnostics	Mixed catalog: WinRT classes, native functions, COM interfaces, structs, handles, constants.

Top-level UI category	Included items	Browsable object model
Microsoft > Windows Desktop UI	WinUI 3, WPF, Windows Forms, UWP, WinUI 2, WebView2	Mostly managed/WinRT UI objects, controls, windows, views, dialogs, events.
Microsoft > Developer Tools	PowerShell, Roslyn, MSBuild, NuGet Client SDK, Visual Studio SDK	Automation APIs for building, scripting, compiling, analyzing, and packaging.
Interop Wrappers	Vanara, TerraFX, Vortice, CsWin32-generated outputs	Alternative projections over native Windows capability.
Community Frameworks	Avalonia, Uno, ReactiveUI, Prism, Spectre.Console, Terminal.Gui, FlaUI	High-value app frameworks and automation/UI utilities.
Utility Libraries	Dapper, Newtonsoft.Json, Serilog, Polly, CsvHelper, SkiaSharp, ImageSharp, NAudio, ClosedXML, QuestPDF	Common libraries useful for generated tools, examples, and terminal workflows.

9. Database/source model

Preserve source identity
Do not collapse APIs just because they have the same name. System.IO.FileStream in .NET Framework, .NET 10, and a compatibility facade may share names but differ in availability, implementation, platform support, and member surface.

Record	Examples	Purpose
Product	Windows, .NET, Windows App SDK, PowerShell	Top-level user-facing owner.
ProductVersion	Windows SDK 10.0.x, .NET 10, PowerShell 7.x	Version-aware search and availability.
ApiFamily	Win32, WinRT, WPF, ASP.NET Core, Roslyn	User-facing framework grouping.
SourceAsset	NuGet package, SDK folder, reference pack, .winmd file, installed assembly	Where the metadata was obtained.
AssemblyOrMetadataFile	System.Runtime.dll, Windows.winmd, Win32.winmd	Physical metadata unit.
Namespace	System.IO, Microsoft.UI.Xaml, Windows.Storage	Logical code grouping.
TypeOrNativeSymbol	HttpClient, Window, StorageFile, CreateFileW, WDFDEVICE	API entity.
MemberOrSignature	Constructor, method, property, field, event, function signature, callback	Invocable or inspectable item.
ExecutionProjection	CLR reflection, WinRT projection, COM interop, P/Invoke, tool execution	How the terminal can actually use it.

10. Practical notes for your app

- Keep managed CLR APIs, WinRT APIs, native functions, COM interfaces, and WDK driver APIs as separate action families. A single tree can display them, but invocation and object-lifetime rules differ.
- Give every cataloged entry an availability record: product, version, target framework, runtime identifier, OS contract, assembly identity, package version, and source hash.
- Treat reference assemblies and metadata packages as read-only catalog inputs. Use installed runtime assemblies or generated bindings only when execution is requested.
- Rank APIs by constructibility: static-only, concrete class, struct, abstract class, interface, open generic, COM coclass, WinRT class, native function, WDF handle, tool command.
- Add recipes for hard-to-construct types. Many important classes require factories, DI containers, OS handles, STA threads, message loops, configuration, native dependencies, or platform capabilities.
- Represent native Windows APIs as code-generation and interop workflows first. Do not force them into a normal C# class/constructor model.

11. Source-name translation cheat sheet

Technical source name	User-facing label	Brief explanation
Microsoft.NETCore.App.Ref	Modern .NET core libraries	Reference pack describing the public API surface of the corresponding .NET shared framework.
Microsoft.WindowsDesktop.App.Ref	WPF and Windows Forms	Reference pack describing managed Windows desktop APIs in the Windows Desktop shared framework.
Microsoft.AspNetCore.App.Ref	ASP.NET Core shared framework	Reference pack for Microsoft-supported ASP.NET Core assemblies.
NETStandard.Library.Ref	.NET Standard compatibility contract	Reference pack for API compatibility specifications, not a runtime.
Microsoft.WindowsAppSDK	Windows App SDK / WinUI 3	NuGet-distributed Windows desktop platform and UI SDK.
Microsoft.Windows.SDK.NET.Ref	Windows Runtime APIs projected to .NET	Targeting pack exposing Windows SDK/WinRT APIs to modern .NET projects.
Microsoft.Windows.SDK.Win32Metadata	Win32 native API metadata	WinMD metadata for native Win32 functions, structures, constants, handles, and COM declarations.
Microsoft.Windows.SDK.Contracts	Legacy/compatibility WinRT contracts	WinRT API contracts mainly relevant for older desktop project models.
Windows SDK installation	Official Windows platform SDK	Headers, libraries, WinRT metadata, tools, samples, signing and build assets.

Technical source name	User-facing label	Brief explanation
WDK installation / WDK metadata	Windows Driver Kit	Driver-specific SDK: frameworks, headers, libraries, metadata, templates, analysis, signing, testing, and deployment tools.

12. References and useful entry points

- Microsoft: .NET distribution packaging - reference packs and shared frameworks - https://learn.microsoft.com/en-us/dotnet/core/distribution-packaging
- Microsoft: Windows SDK overview - https://learn.microsoft.com/en-us/windows/apps/windows-sdk/
- Microsoft: Windows App SDK overview - https://learn.microsoft.com/en-us/windows/apps/windows-app-sdk/
- Microsoft: WinUI 3 documentation - https://learn.microsoft.com/en-us/windows/apps/winui/winui3/
- Microsoft: Windows desktop app API reference - https://learn.microsoft.com/en-us/windows/apps/api-reference/
- Microsoft: Win32 API reference - https://learn.microsoft.com/en-us/windows/win32/api/
- Microsoft: .NET Standard - https://learn.microsoft.com/en-us/dotnet/standard/net-standard
- Microsoft: .NET Framework reference assemblies - https://learn.microsoft.com/en-us/dotnet/framework/migration-guide/reference-assemblies
- Microsoft: ASP.NET Core shared framework/metapackage - https://learn.microsoft.com/en-us/aspnet/core/fundamentals/metapackage-app
- Microsoft GitHub: Win32 metadata - https://github.com/microsoft/win32metadata
- Vanara - https://github.com/dahall/Vanara
- TerraFX.Interop.Windows - https://github.com/terrafx/terrafx.interop.windows
- Vortice.Windows - https://github.com/amerkoleci/Vortice.Windows
- Avalonia UI - https://avaloniaui.net/
- Spectre.Console - https://spectreconsole.net/
- RestSharp - https://restsharp.dev/

13. One-page implementation checklist

Area	Task
Catalog v1	Index .NET core, WPF, WinForms, Windows App SDK, WinRT, .NET Framework, PowerShell, Roslyn, MSBuild, NuGet, ASP.NET Core.
Search model	Support type/member/package/source queries, signature search, inherited vs declared members, extension methods, availability filters.
Invocation model	Separate CLR reflection, WinRT, COM, P/Invoke, WDK/tool execution, and generated-code workflows.
Object shelf	Store live objects as session handles with runtime type, creation recipe, disposal state, owning worker, and dependencies.

Area	Task
Native model	For Win32, catalog functions/structs/COM; use CsWin32/Vanara/TerraFX/Vortice for execution and generated code.
Safety model	Warn for destructive APIs, OS/platform restrictions, UI-thread requirements, network/file/process operations, native calls, driver APIs, and elevation.
Expansion path	After the Microsoft foundation is stable, add high-value secondary libraries based on user workflows and NuGet popularity.

Prepared as a shareable planning document for a C# terminal API browser and object workbench.