.NET Core: Developing Cross-Platform Web Apps with ASP.NET Core – Workshop*PLUS*

< Engineer Name >

Customer Engineer

v3.1

Module 1: .NET Overview

Module Overview

Module 1: Overview

Section 1: .NET Platform

Lesson: Overview

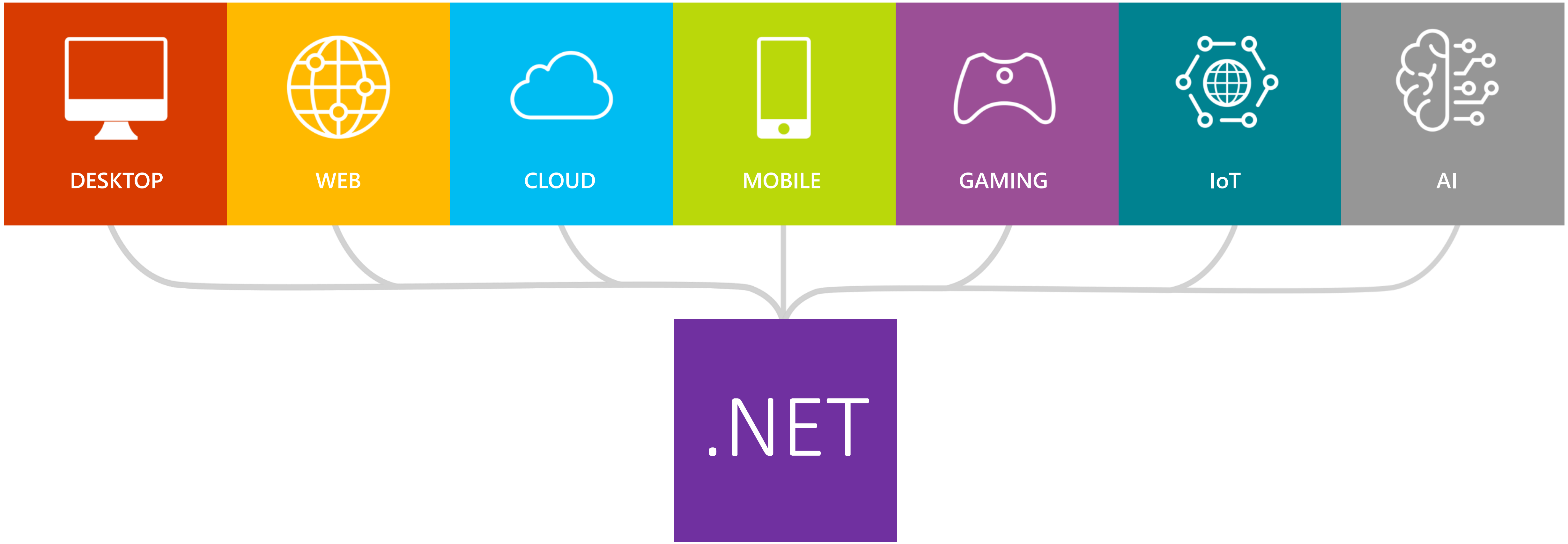
.NET

Free. Cross-platform. Open source.
A developer platform for building all your apps.

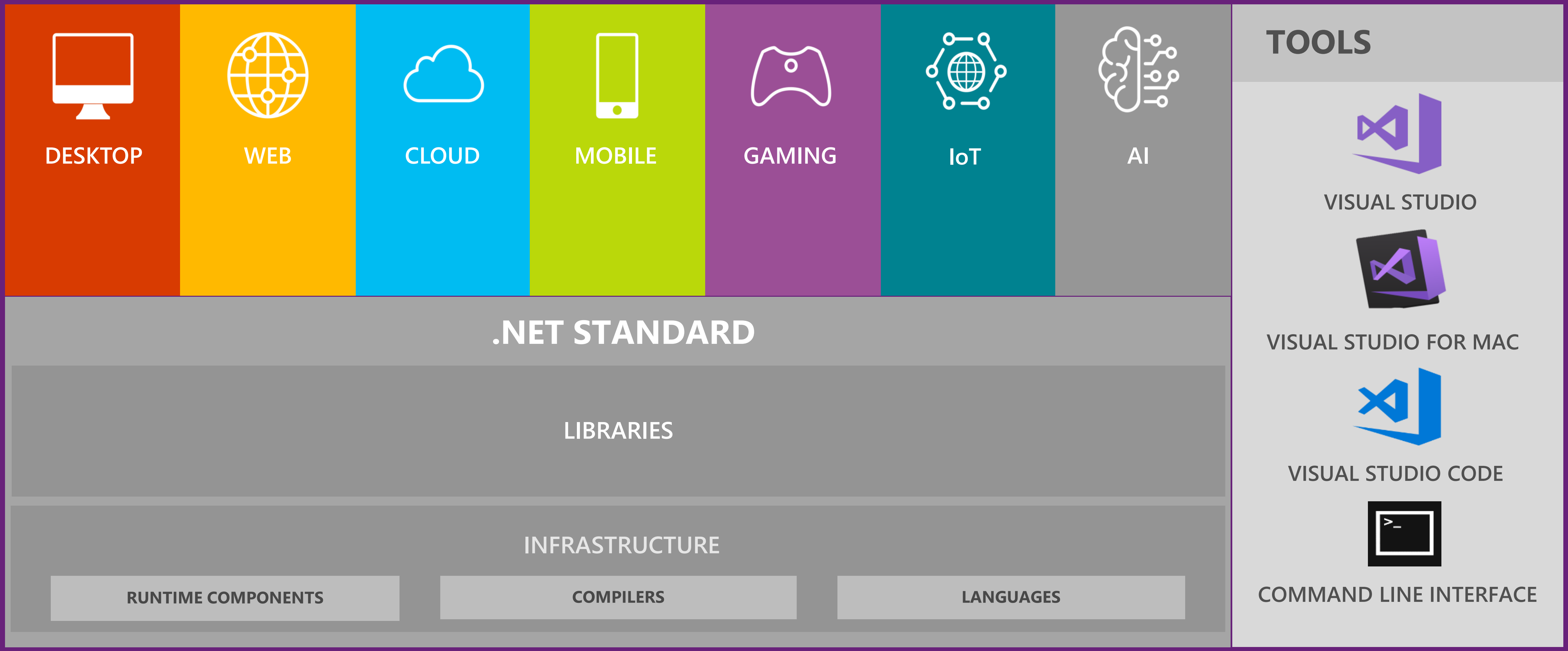
www.dot.net

What is .NET?

Your platform for building **anything**



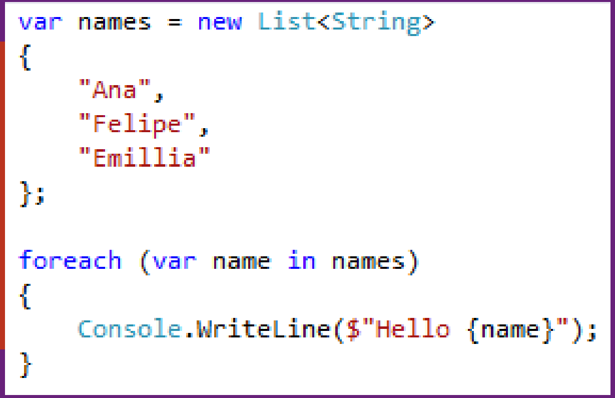
.NET is a software development platform



You can write code with many .NET languages

C# (c-sharp)

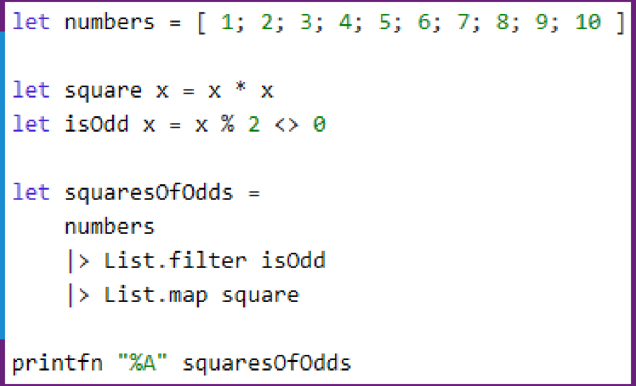
- C# is a simple, modern, object-oriented, and type-safe programming language
- Its roots in the C family of languages makes C# immediately familiar to C, C++, Java, and JavaScript programmers



```csharp
var names = new List<String>
{
    "Ana",
    "Felipe",
    "Emillia"
};

foreach (var name in names)
{
    Console.WriteLine($"Hello {name}");
}
```

F# (f-sharp)

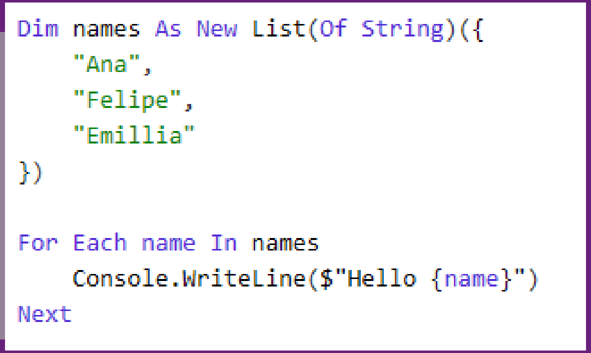
- F# is a cross-platform, open-source, functional programming language for .NET
- It also includes object-oriented and imperative programming



```fsharp
let numbers = [ 1; 2; 3; 4; 5; 6; 7; 8; 9; 10 ]

let square x = x * x
let isOdd x = x % 2 <> 0

let squaresOfOdds =
    numbers
    |> List.filter isOdd
    |> List.map square

printfn "%A" squaresOfOdds
```

Visual Basic

- Visual Basic is an approachable language with a simple syntax for building type-safe, object-oriented apps

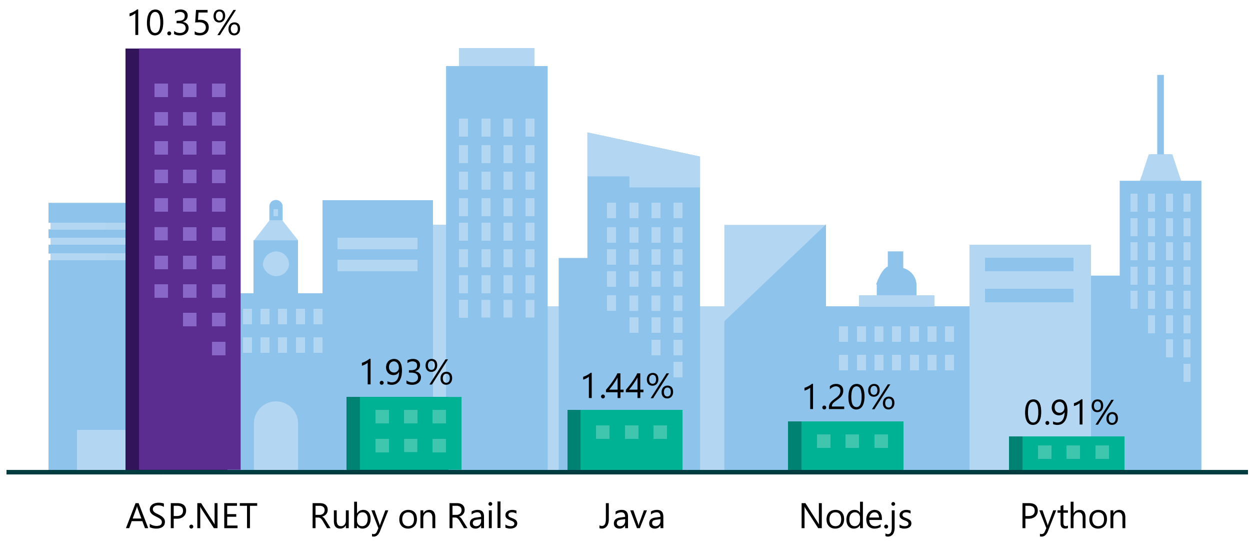


```vb
Dim names As New List(Of String)({
    "Ana",
    "Felipe",
    "Emillia"
})

For Each name In names
    Console.WriteLine($"Hello {name}")
Next
```

.NET is the platform of choice for the top 100K websites

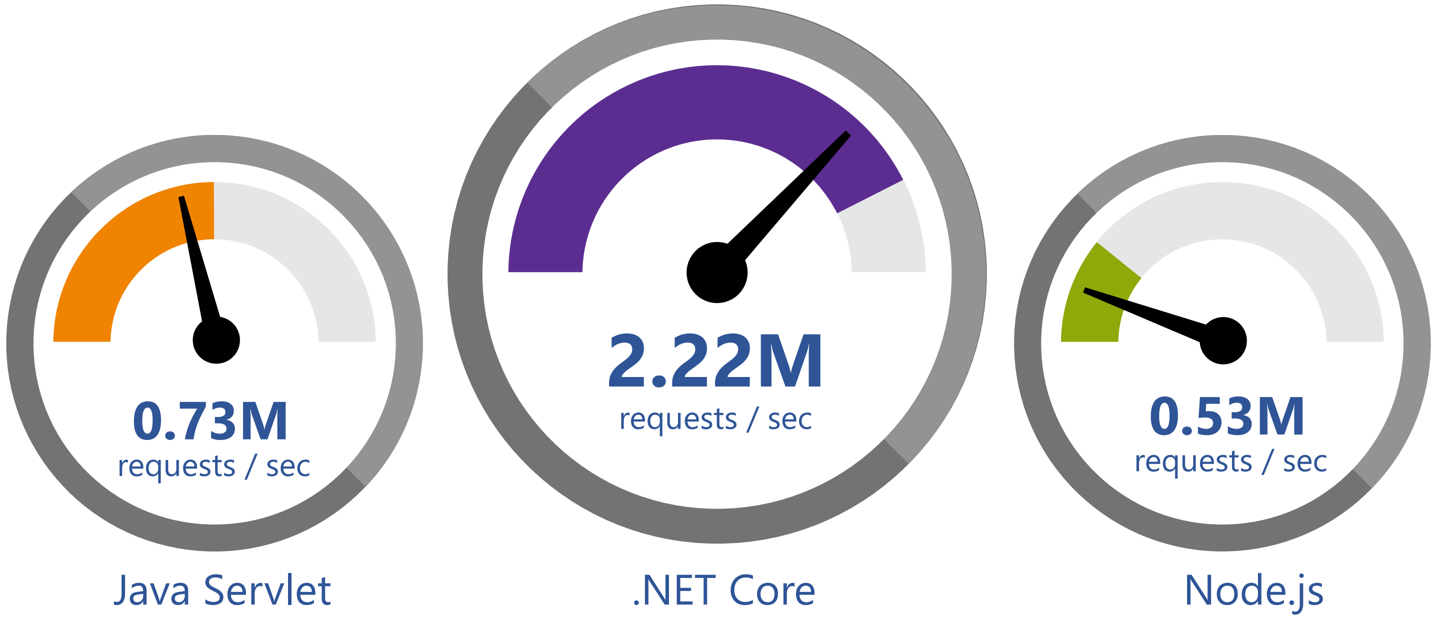
More websites have been developed with ASP.NET than Ruby, Java, Python, Node.js, and Go combined.

Companies like Raygun, GoDaddy, and Tencent choose .NET for better performance, increased flexibility, and higher compatibility.



Data sourced from SimilarTech

.NET is Fast. Really Fast!

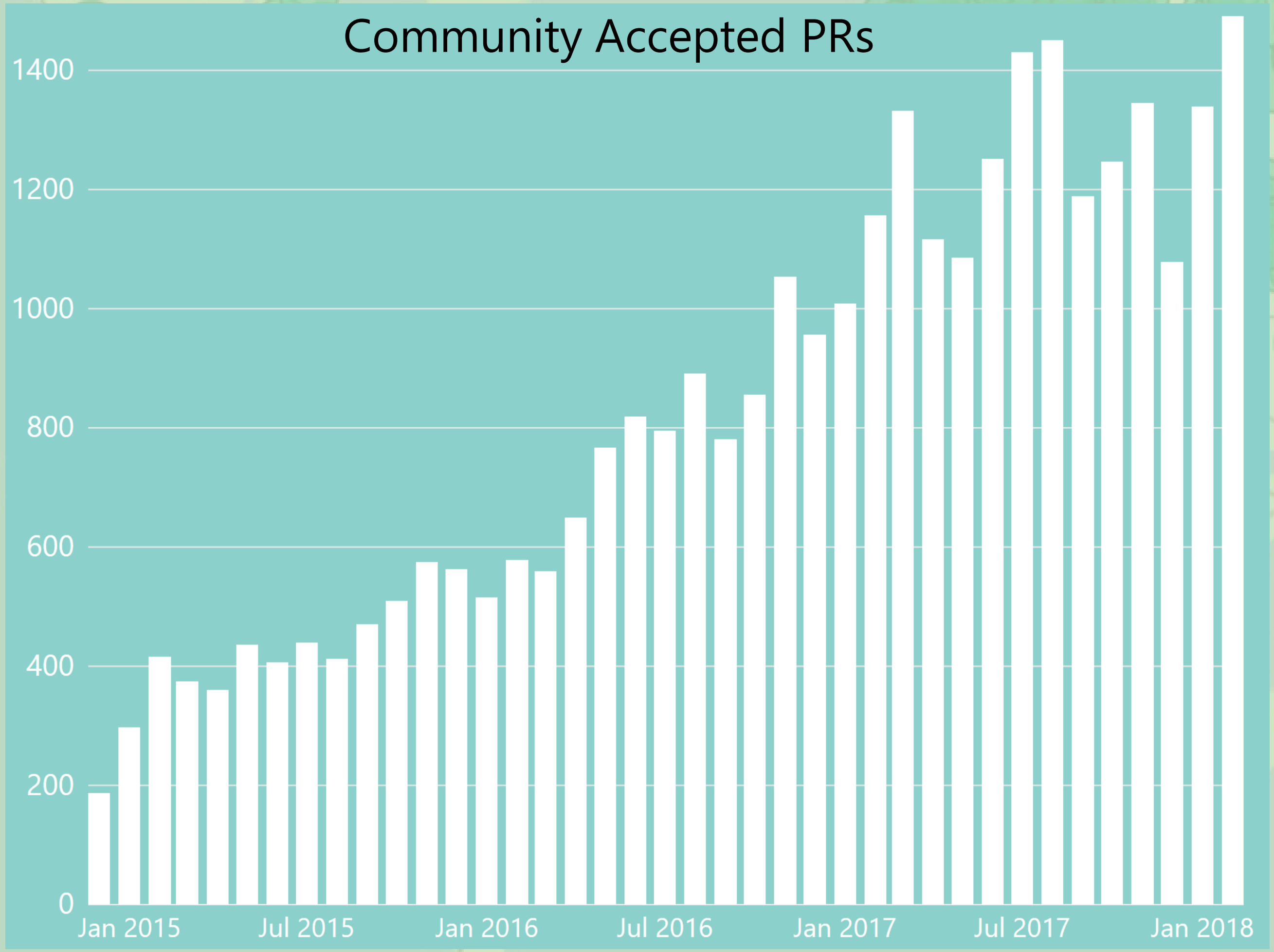


Data sourced from official tests available at TechEmpower Round 15.

"Using the same-size server, we were able to go from 1,000 requests per second per node with Node.js to 20,000 requests per second with .NET Core."
— Raygun

https://www.microsoft.com/net/customers

.NET is Open Source



“Samsung is embracing .NET because it is a completely open source project.” — Samsung

".NET is open source; that allows us to contribute back to it if we have any performance issues which Microsoft review and together we make a better product.“ — Illyriad Games

16,000+ Community Contributions from 3000+ Companies outside Microsoft

Steeltoe

Microsoft Web Protection Library

.NET Core

MSBuild

DotNetNuke

ASP.NET AJAX Control Toolkit

ASP.NET Core

SourceLink

Microsoft Azure WebJobs SDK

MVVM Light Toolkit

nUnit

Salesforce Toolkits for .NET

Mailkit

IronPython

WorldWide Telescope

Open XML SDK

Nancy

MvvmCross

MEF

.NET SDK for Hadoop

Mimekit

ASP.NET Web Pages

xUnit

NuGet

IdentityManager

Couchbase Lite for .NET

WCF

Entity Framework

ILMerge

Xamarin.Mobile

ASP.NET SignalR

.NET Micro Framework

Open Live Writer

Mono

Xamarin.Auth

Xamarin SDK

Orchard CMS

OWIN Authentication Middleware

ProtoBuild

IdentityServer

Microsoft Azure SDK for .NET

.NET Compiler Platform ("Roslyn")

Prism

ASP.NET Web API

Cake

Orleans

LLILC

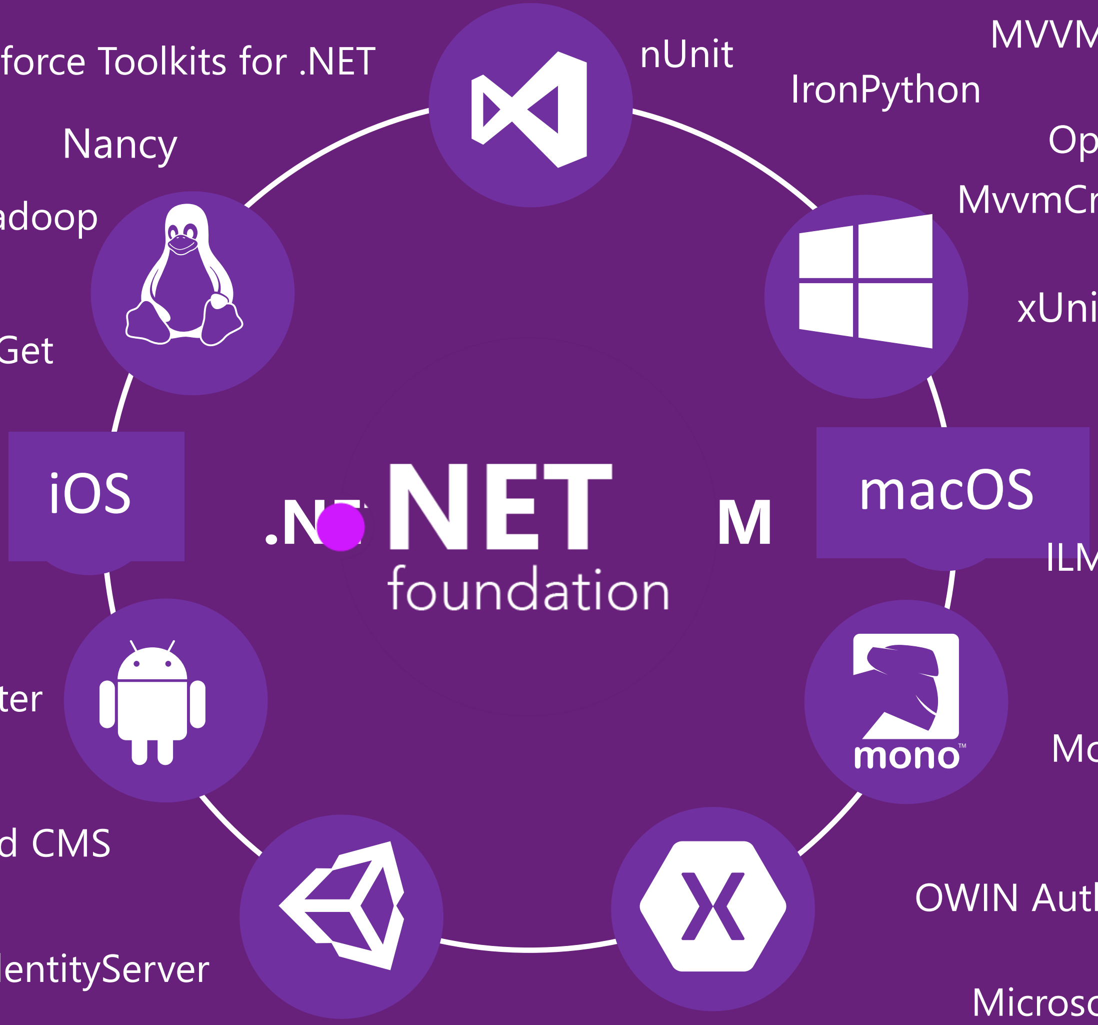
Kudu

Umbraco

Cecil

System.Drawing

ASP.NET MVC



.NET Core

	.NET Core			ASP.NET Core	EF Core
Source License	MIT			Apache 2	Apache 2
Binary License	Microsoft EULA			Microsoft EULA	Microsoft EULA
Acquisition	Installer	Package-Manager	NuGet	NuGet	NuGet
OSes	Windows	macOS	Linux	Same	Same
App Deployment	Runtime-dependent	Self-contained	Docker	Same	
Side-by-side installs	Yes!			Yes!	Yes!

.NET Framework or .NET Core 🤔

Which one to choose for building server apps?

.NET Framework	.NET Core
Current application runs on .NET framework. Recommended to extend it instead of migrating.	Cross-platform needs
Need 3rd party libraries not available on .NET Core	Targeting microservices
Need .NET technologies not available on .NET Core	Using Docker containers
Need a platform not supported by .NET Core	Need high performance & scalable systems
	Side-by-side .NET versions by application
	Fully open-source

Demo:

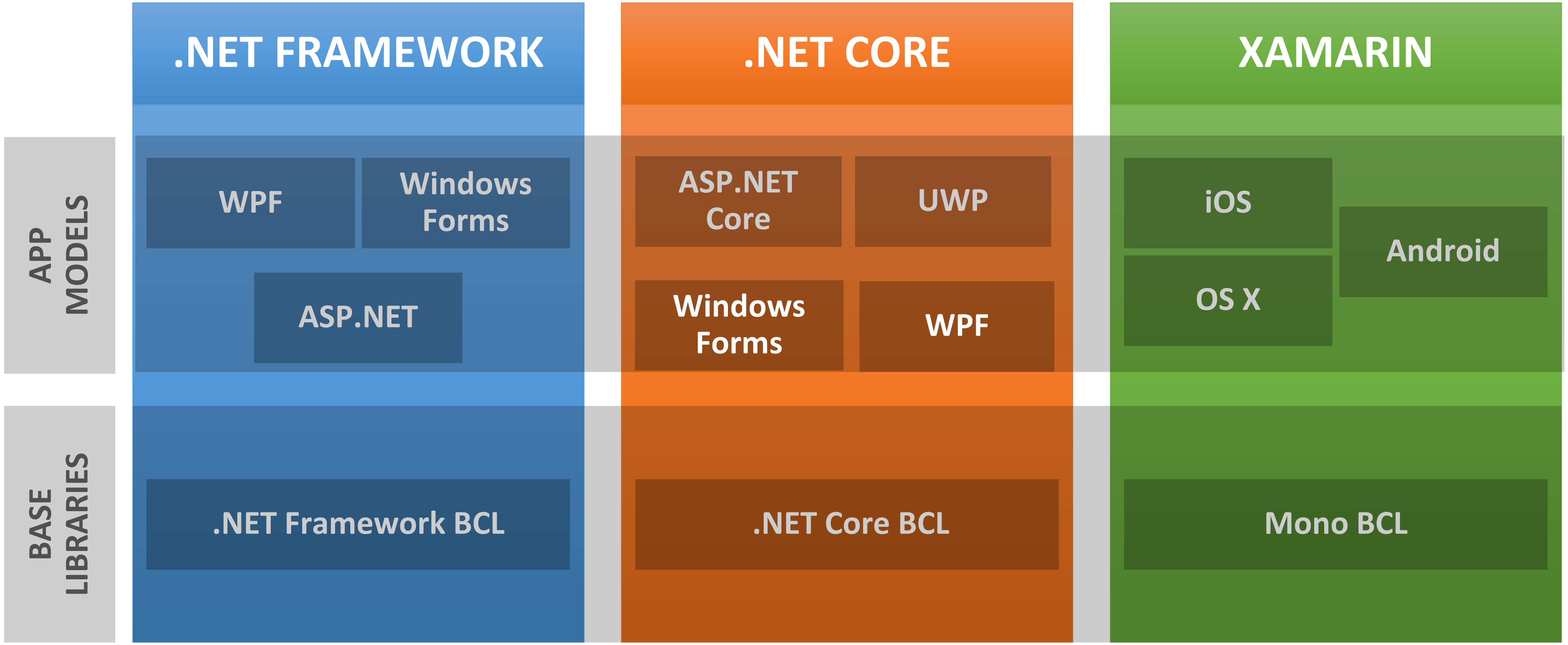
Create Console App

Module 1: Overview

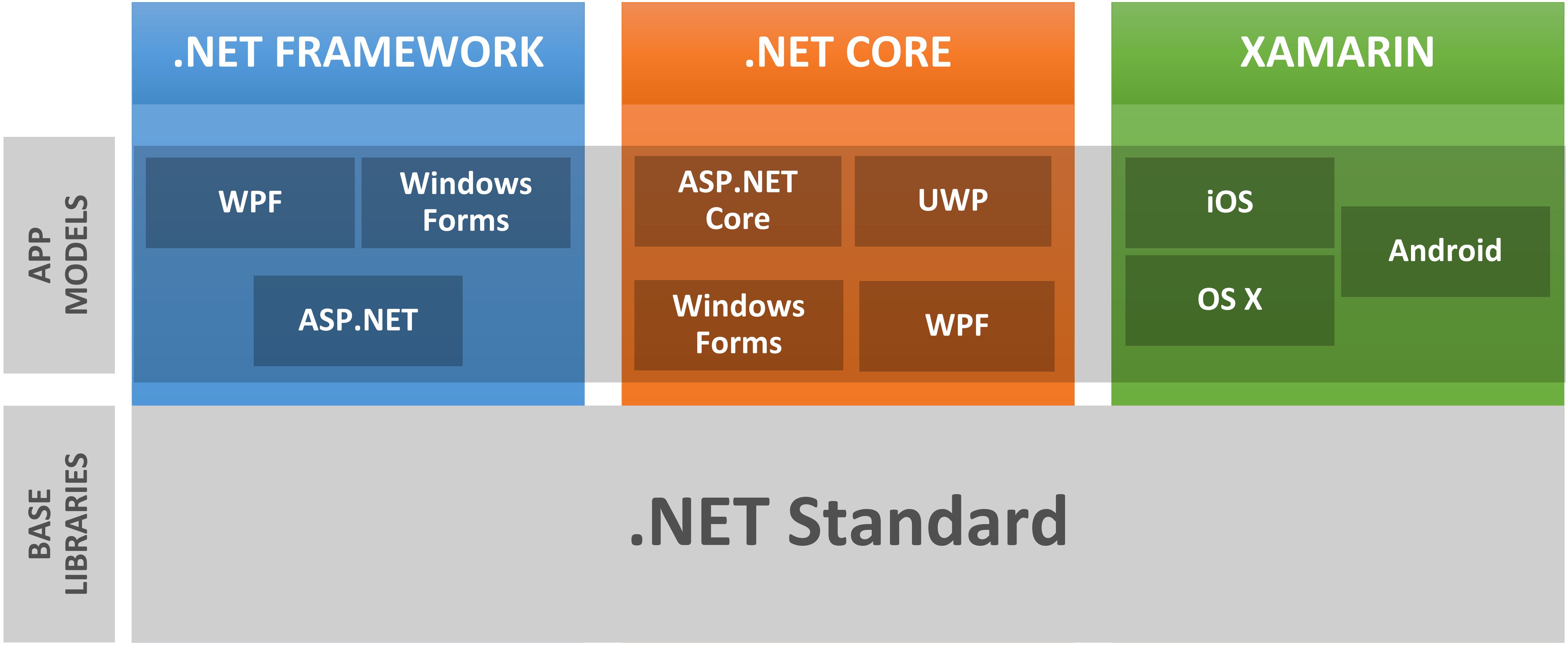
Section 1: .NET Platform

Lesson: .NET Standard

.NET Before – Reuse Code!



.NET Now


APP MODELS
BASE LIBRARIES
.NET FRAMEWORK
WPF
Windows Forms
ASP.NET
.NET CORE
ASP.NET Core
UWP
Windows Forms
WPF
XAMARIN
iOS
OS X
Android
.NET Standard

.NET Standard Library

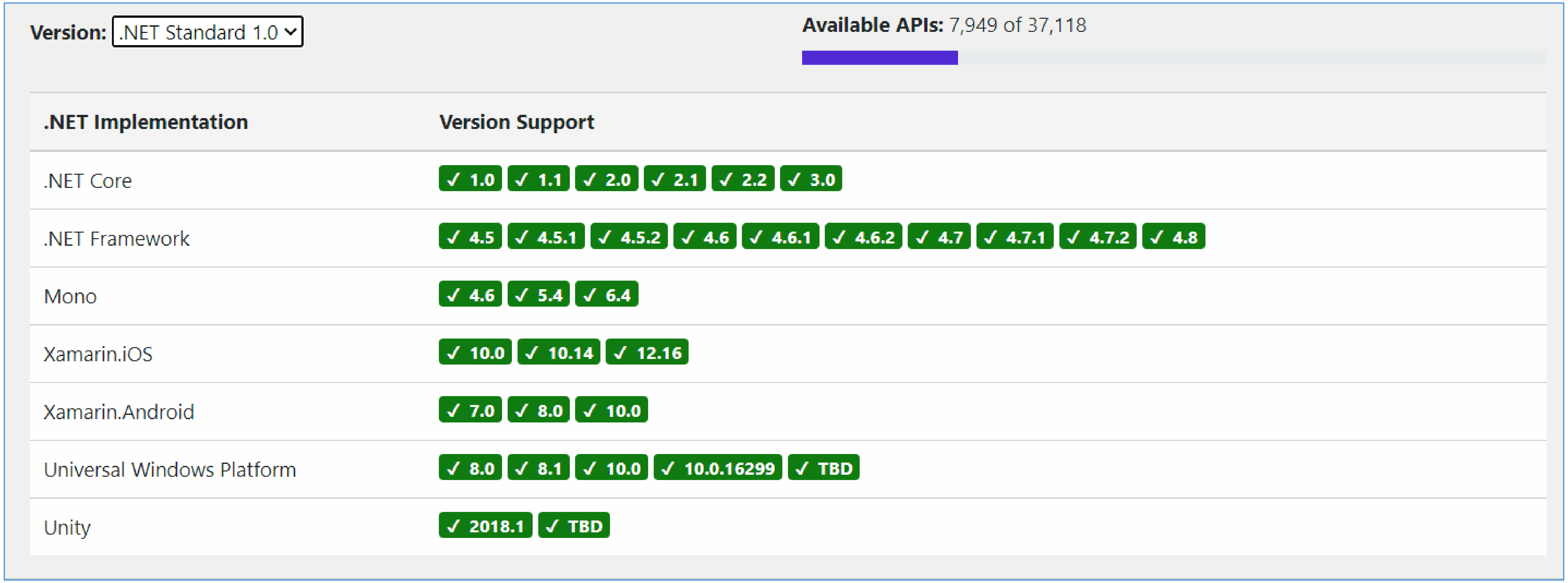
- Goal: Establish greater uniformity in the .NET ecosystem
- A set of APIs that all .NET platforms have to implement
- Unifies the .NET platform and prevents future fragmentation
- .NET Standard will replace Portable Class Libraries (PCLs)
- Addresses three main scenarios:
    - Defines uniform set of BCL APIs for all .NET platforms to implement, independent of workload
    - Enables developers to produce portable libraries that are usable across .NET runtimes, using the same set of APIs
    - Reduces and hopefully eliminates conditional compilation of shared source due to .NET APIs

.NET Standard Versions

.NET Standard	1.0	1.1	1.2	1.3	1.4	1.5	1.6	2.0	2.1
.NET Core	1.0	1.0	1.0	1.0	1.0	1.0	1.0	2.0	3.0
.NET Framework [1]	4.5	4.5	4.5.1	4.6	4.6.1	4.6.1 [2]	4.6.1 [2]	4.6.1 [2]	N/A [3]
Mono	4.6	4.6	4.6	4.6	4.6	4.6	4.6	5.4	6.4
Xamarin.iOS	10.0	10.0	10.0	10.0	10.0	10.0	10.0	10.14	12.16
Xamarin.Mac	3.0	3.0	3.0	3.0	3.0	3.0	3.0	3.8	5.16
Xamarin.Android	7.0	7.0	7.0	7.0	7.0	7.0	7.0	8.0	10.0
Universal Windows Platform	10.0	10.0	10.0	10.0	10.0	10.0.16299	10.0.16299	10.0.16299	TBD
Unity	2018.1	2018.1	2018.1	2018.1	2018.1	2018.1	2018.1	2018.1	TBD

.NET Framework won't support .NET Standard 2.1 or later versions.

.NET Standard Versions Illustration



Version: .NET Standard 1.0

Available APIs: 7,949 of 37,118

.NET Implementation	Version Support
.NET Core	✓ 1.0 ✓ 1.1 ✓ 2.0 ✓ 2.1 ✓ 2.2 ✓ 3.0
.NET Framework	✓ 4.5 ✓ 4.5.1 ✓ 4.5.2 ✓ 4.6 ✓ 4.6.1 ✓ 4.6.2 ✓ 4.7 ✓ 4.7.1 ✓ 4.7.2 ✓ 4.8
Mono	✓ 4.6 ✓ 5.4 ✓ 6.4
Xamarin.iOS	✓ 10.0 ✓ 10.14 ✓ 12.16
Xamarin.Android	✓ 7.0 ✓ 8.0 ✓ 10.0
Universal Windows Platform	✓ 8.0 ✓ 8.1 ✓ 10.0 ✓ 10.0.16299 ✓ TBD
Unity	✓ 2018.1 ✓ TBD

Source: https://dotnet.microsoft.com/platform/dotnet-standard

What version should you target?

- The **higher the version**, the **more APIs** you have
- The **lower the version**, the **more platforms** supp

Lo **ersion**

Mo **More APIs**

Target the lowest version you can get away with!

APIs in .NET Standard 2.0

XML	XLinq • XML Document • XPath • Schema • XSL
SERIALIZATION	BinaryFormatter • Data Contract • XML
NETWORKING	Sockets • HTTP • Mail • WebSockets
IO	Files • Compression • MMF
THREADING	Threads • Thread Pool • Tasks
CORE	Primitives • Collections • Reflection • Interop • Linq

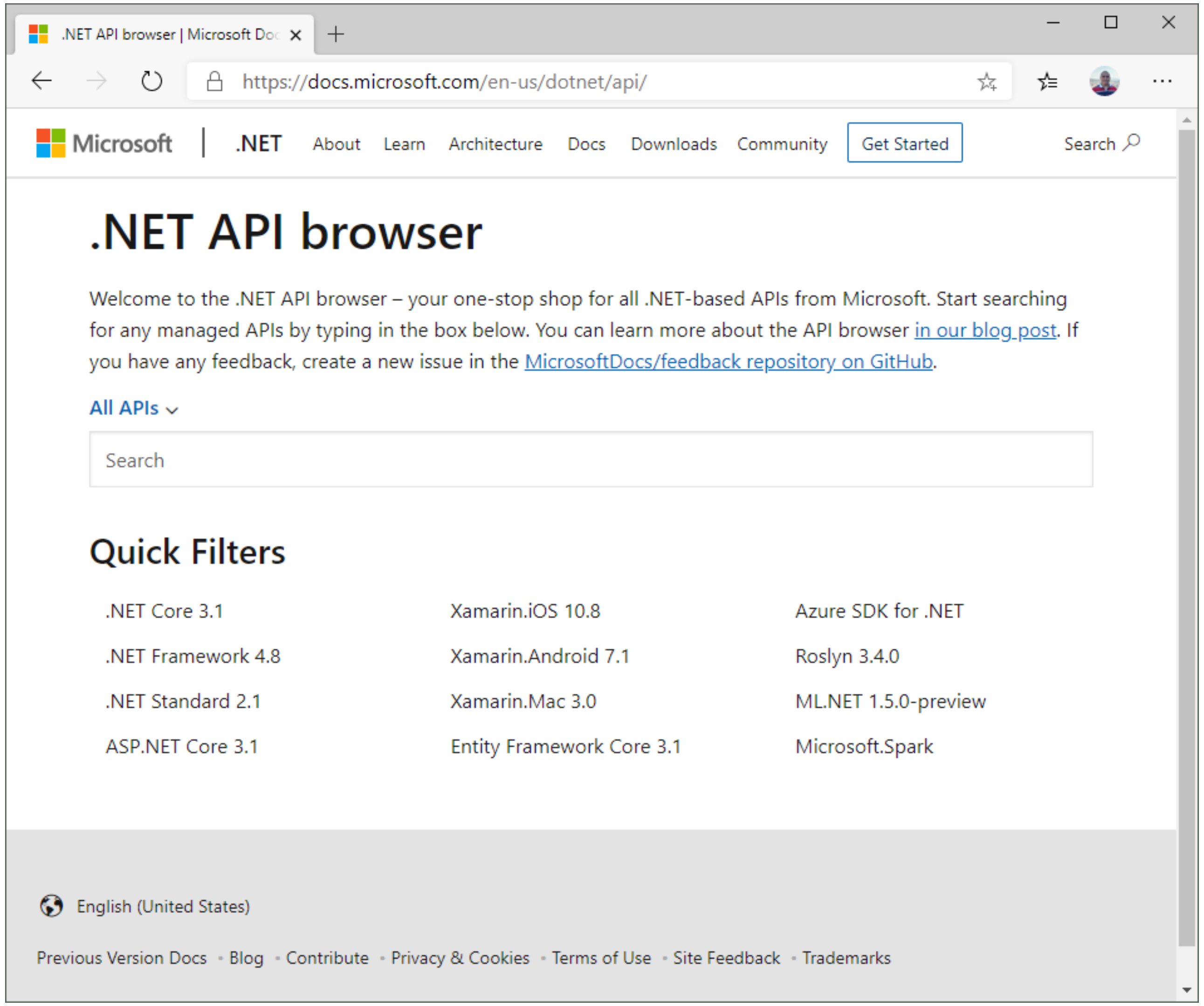
APIs in .NET Standard 2.1

- **Span<T>**
- **Foundational-APIs working with spans**
- **Reflection emit**
- **SIMD**
- **ValueTask and ValueTask<T>**
- **DbProviderFactories**
- **General Goodness**

.NET API Browser

Is one-stop shop for all .NET-based APIs from Microsoft. You can search for any managed APIs in it.

https://docs.microsoft.com/en-us/dotnet/api/


.NET API browser | Microsoft Doc
https://docs.microsoft.com/en-us/dotnet/api/
Microsoft
.NET
About
Learn
Architecture
Docs
Downloads
Community
Get Started
Search

.NET API browser

Welcome to the .NET API browser – your one-stop shop for all .NET-based APIs from Microsoft. Start searching for any managed APIs by typing in the box below. You can learn more about the API browser in our blog post. If you have any feedback, create a new issue in the MicrosoftDocs/feedback repository on GitHub.

All APIs

Search

Quick Filters

.NET Core 3.1	Xamarin.iOS 10.8	Azure SDK for .NET
.NET Framework 4.8	Xamarin.Android 7.1	Roslyn 3.4.0
.NET Standard 2.1	Xamarin.Mac 3.0	ML.NET 1.5.0-preview
ASP.NET Core 3.1	Entity Framework Core 3.1	Microsoft.Spark

English (United States)
Previous Version Docs · Blog · Contribute · Privacy & Cookies · Terms of Use · Site Feedback · Trademarks

Demo:

1. Open-Source ASP.NET
2. .NET Core on NuGet

Demo:

.NET Standard 2.0

Module 1: Overview

Section 2: .NET Core

Lesson: Command Line Interface (CLI)

.NET Core Command Line Interface (CLI)

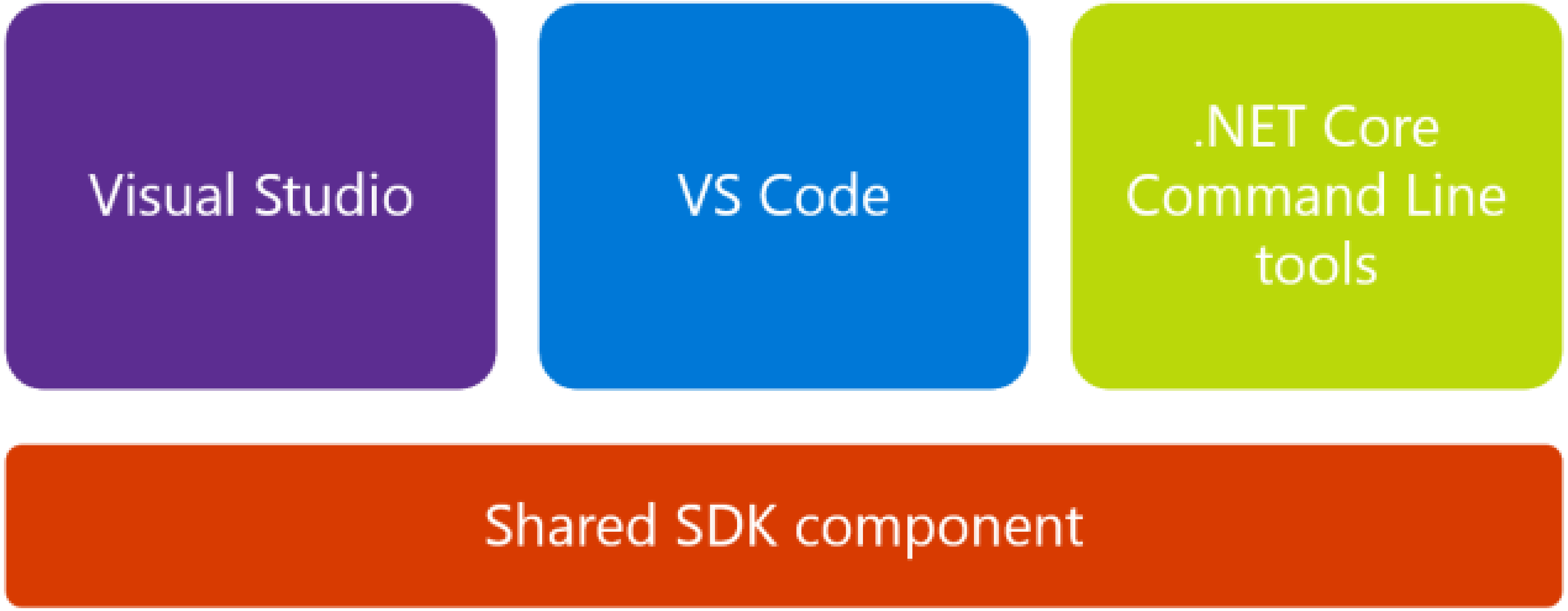
- Cross-platform toolchain for developing .NET Core applications
- Primary layer built upon by Visual Studio, editors, build orchestrators, etc.
- Cross-platform with same surface area for supported platforms
- Language agnostic
- Target agnostic

```
dotnet new
dotnet restore
dotnet build --output /stuff
dotnet run
dotnet stuff/new.dll
```

CLI Command Examples

`dotnet restore`	Uses NuGet to restore dependencies as well as project-specific tools that are specified in the project file in parallel.
`dotnet build`	Restores any dependencies then builds the project and its dependencies into a set of binaries. The binaries include the project's code in Intermediate Language (IL) files with a .dll extension and symbol files used for debugging with a .pdb extension.
`dotnet run`	It allows you to run your application from the source code with one command. It's useful for fast iterative development from the command line. The command depends on the dotnet build command to build the code. Any requirements for the build, such as that the project must be restored first.
`dotnet clean`	Cleans the output of the previous build.
`dotnet new web`	Create a new Empty web application then restores the dependencies/packages for it.
`dotnet watch`	A file watcher to listen when a file change can to trigger compilation, test execution, or deployment

.NET Core Tooling



CLI dotnet new templates

.NET Core have many templates from the CLI

Example:

```
dotnet new razor -lang c#
```

This will install an ASP.NET Core Web Application which using Angular and C# language

To get more template you can check https://dotnetnew.azurewebsites.net/

Template description	Template name	Languages
Console application	console	[C#], F#, VB
Class library	classlib	[C#], F#, VB
Unit test project	mstest	[C#], F#, VB
xUnit test project	xunit	[C#], F#, VB
Razor page	page	[C#]
MVC ViewImports	viewimports	[C#]
MVC ViewStart	viewstart	[C#]
ASP.NET Core empty	web	[C#], F#
ASP.NET Core Web App (Model-View-Controller)	mvc	[C#], F#
ASP.NET Core Web App	razor	[C#]
ASP.NET Core with Angular	angular	[C#]
ASP.NET Core with React.js	react	[C#]
ASP.NET Core with React.js and Redux	reactredux	[C#]
ASP.NET Core Web API	webapi	[C#], F#
Razor class library	razorclasslib	[C#]
global.json file	globaljson	
NuGet config	nugetconfig	
Web config	webconfig	
Solution file	sln	

.NET Core CLI Extensibility

- .NET Core is built for extensibility, you extend the CLI with your own custom commands and tooling
- The CLI tools can be extended in three main ways:
  1. Via NuGet packages on a per-project basis
     Per-project tools are contained within the project's context, but they allow easy installation through restoration.
  2. Via NuGet packages with custom targets
     Custom targets allow you to easily extend the build process with custom tasks.
  3. Via the system's PATH
     PATH-based tools are good for general, cross-project tools that are usable on a single machine.

Example of extensibility is the EF Core commands

Module 1: Overview

Section 2: .NET Core

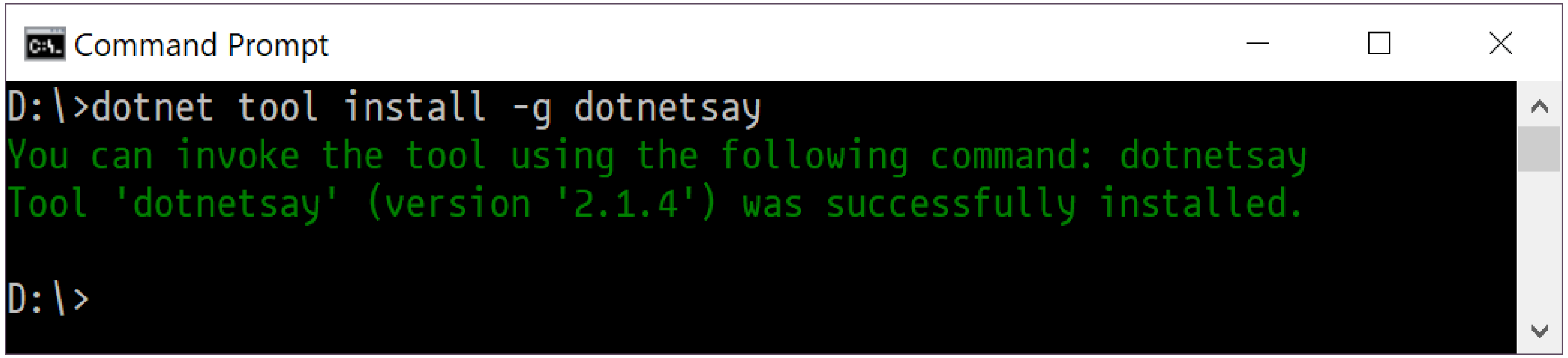
Lesson: Global Tools

.NET Core Global Tools

- .NET Core Global Tool is a special NuGet package that contains a console application.
- A Global Tool can be installed on your machine on a default location that is included in the PATH environment variable or on a custom location.

To install a global tool use the following format:

```
dotnet tool install -g <package-name> --version <version-number>
```



```
D:\>dotnet tool install -g dotnetsay
You can invoke the tool using the following command: dotnetsay
Tool 'dotnetsay' (version '2.1.4') was successfully installed.

D:\>
```

You can run the tool by typing its name like: `D:\> dotnetsay`

.NET Global Tools Examples

- Code generation tool for creating controllers, views, and models in ASP.NET Core projects.
  - `dotnet tool install -g dotnet-aspnet-codegenerator`

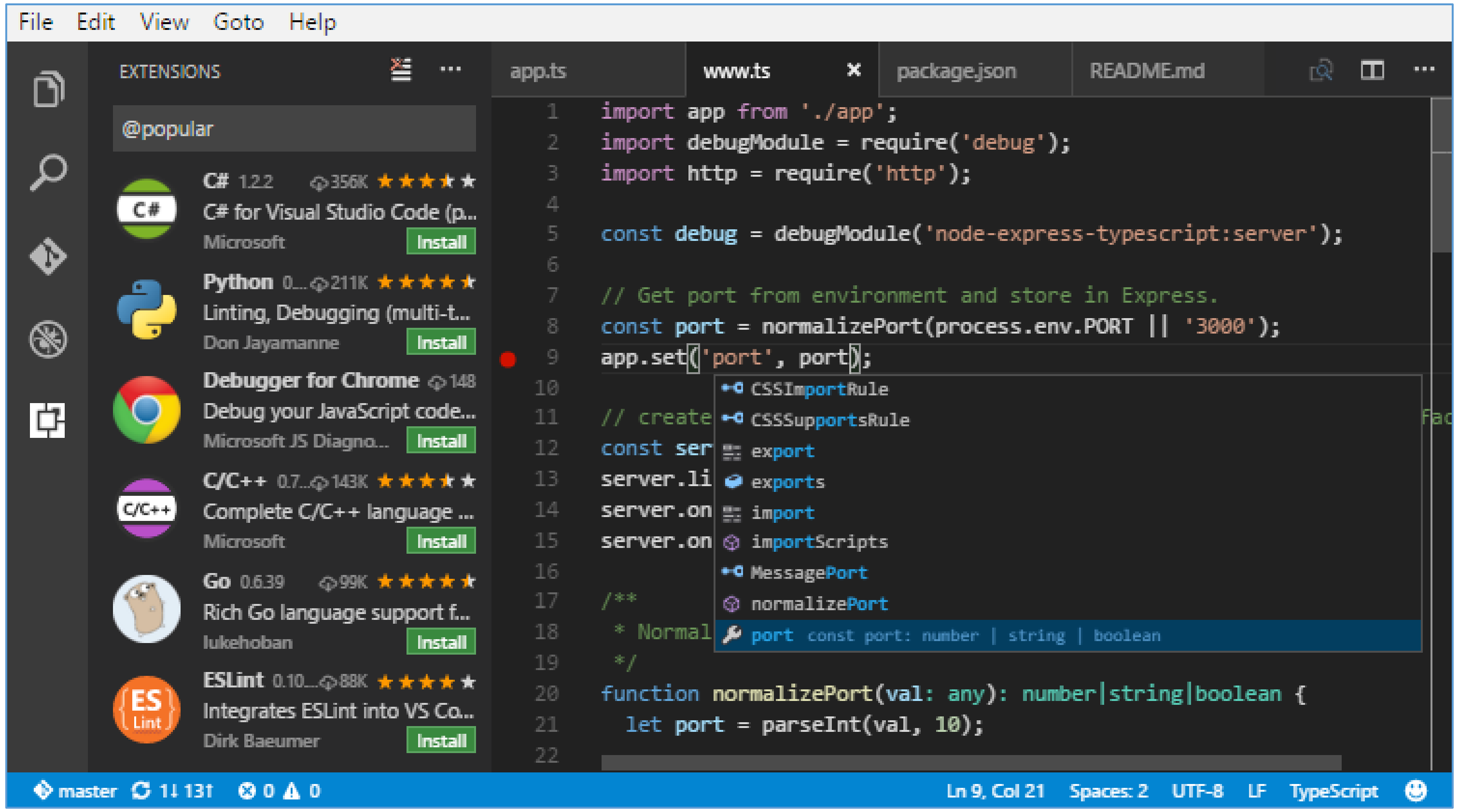
- A tool to run cross platform Cake build scripts.
  - `dotnet tool install -g Cake.Tool`

- dotnet-format is a code formatter for dotnet that applies style preferences to a project or solution.
  - `dotnet tool install -g dotnet-format`

- You find more at https://github.com/natemcmaster/dotnet-tools

Visual Studio Code

- Visual Studio Code is a lightweight but powerful source code editor which runs on your desktop.
- It combines the simplicity of a source code editor with powerful developer tooling, like IntelliSense code completion and debugging.
- It supports macOS, Linux, and Windows - so you can hit the ground running, no matter the platform.



C# 1.2.2 356K ★★★★★	C# for Visual Studio Code (p...	Microsoft Install
Python 0... 211K ★★★★★	Linting, Debugging (multi-t...	Don Jayamanne Install
Debugger for Chrome 148	Debug your JavaScript code...	Microsoft JS Diagno... Install
C/C++ 0.7... 143K ★★★★★	Complete C/C++ language ...	Microsoft Install
Go 0.6.39 99K ★★★★★	Rich Go language support f...	lukehoban Install
ESLint 0.10... 88K ★★★★★	Integrates ESLint into VS Co...	Dirk Baeumer Install

```
import app from './app';
import debugModule = require('debug');
import http = require('http');

const debug = debugModule('node-express-typescript:server');

// Get port from environment and store in Express.
const port = normalizePort(process.env.PORT || '3000');
app.set('port', port);

// create
const ser
server.li
server.on
server.on

/**
 * Normal
 */
function normalizePort(val: any): number|string|boolean {
  let port = parseInt(val, 10);
```

```
CSSImportRule
CSSSupportsRule
export
exports
import
importScripts
MessagePort
normalizePort
port  const port: number | string | boolean
```

Visual Studio Code

- VS Code includes enriched built-in support for Node.js development with JavaScript and TypeScript, powered by the same underlying technologies that drive Visual Studio. VS Code also includes great tooling for web technologies such as JSX/React, HTML, CSS, SCSS, Less, and JSON.

- Where to get it?
    - You can download for FREE for any platform from https://code.visualstudio.com/

Demo: .NET Core CLI & Visual Studio Code

Module 1: Overview

Section 2: .NET Core

Lesson: .NET Runtime

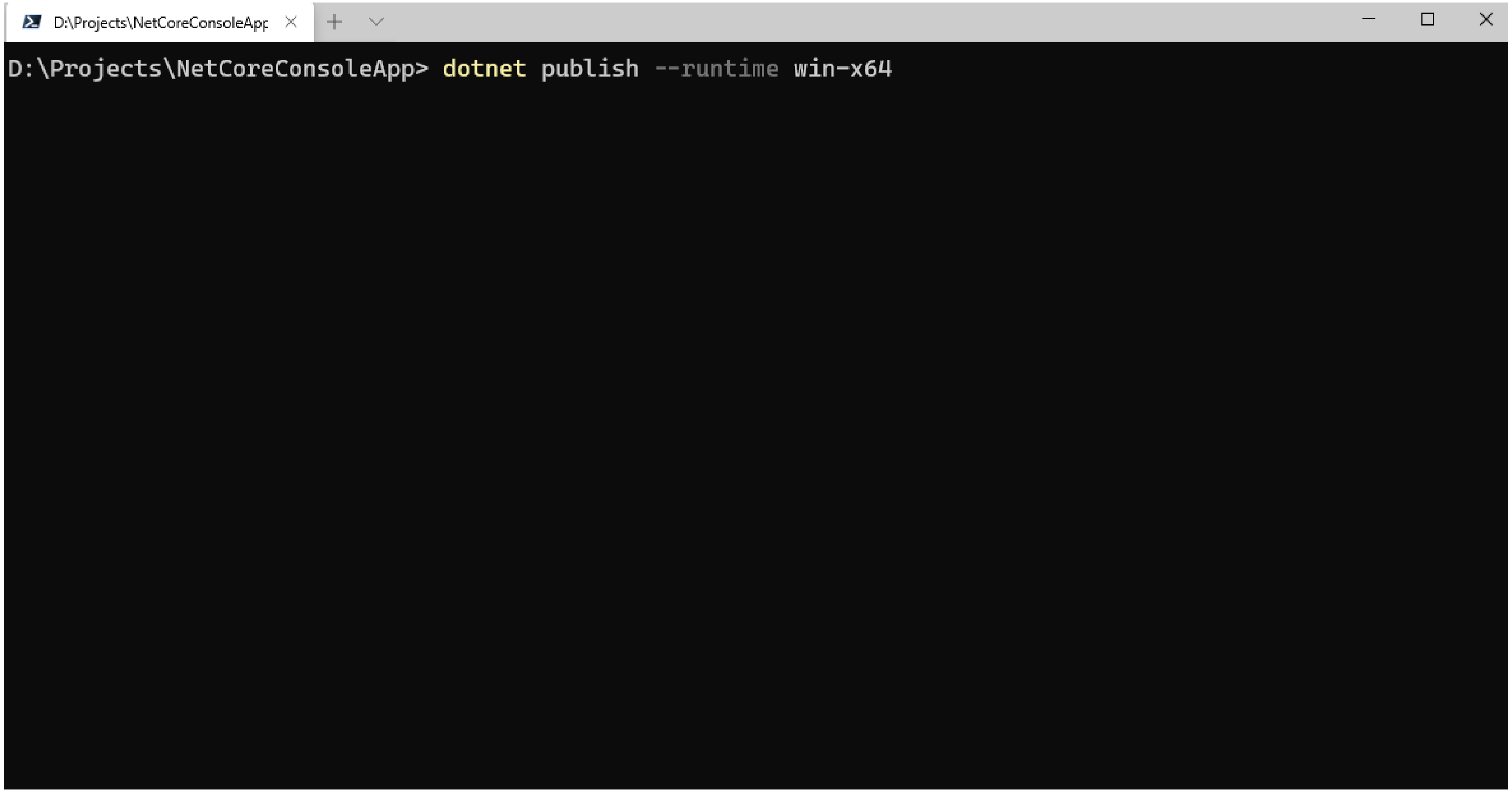
.NET Runtimes

- You can build you ASP.NET Core for a target different platforms using RIDs
- Runtime IDs are values to identify the target platform which the app will run on.
- RIDs can be portable (not coupled to the version of the OS) like:
  - win-x64
  - ubuntu-x64

- RIDs can be coupled to the OS like:
  - win8-x86
  - osx.10.13-x64
  - sles.12.2-x64

- RIDs that represent concrete operating systems usually follow this pattern:
  [os].[version]-[architecture]-[additional qualifiers]

.NET Runtimes Build Command



```
D:\Projects\NetCoreConsoleApp> dotnet publish --runtime win-x64
```

Demo: build on windows for different runtime

Module 1: Overview

Section 2: .NET Core

Lesson: .NET Core Support Lifecycle

.NET Core Release Types

Long Term Support (LTS)

More Stable ✓

- Designed for long-term support. They included features and components that have been stabilized, requiring few updates over a longer support release lifetime. These releases are a good choice for hosting applications that you do not intend to update.
- LTS releases are supported for **three years** after the **initial release**.

Current

More Features 👍

- Includes new features that may undergo future change based on feedback. These releases are a good choice for applications in active development, giving you access to the latest features and improvements. You need to upgrade to later .NET Core releases more often to stay in support.
- Current releases are supported for **three months** after a **subsequent Current or LTS release**.

.NET Core Release Lifecycles

Version	Original Release Date	Latest Patch Version	Patch Release Date	Support Level	End of Support
.NET Core 3.1	03-Dec-2019	3.1.2	18-Feb-2020	LTS	03-Dec-2022
.NET Core 3.0	23-Sep-2019	3.0.3	18-Feb-2020	EOL	03-Mar-2020
.NET Core 2.2	04-Dec-2018	2.2.8	19-Nov-2019	EOL	23-Dec-2019
.NET Core 2.1	30-May-2018	2.1.16	18-Feb-2020	LTS	21-Aug-2021
.NET Core 2.0	14-Aug-2017	2.0.9	10-Jul-2018	EOL	01-Oct-2018
.NET Core 1.1	16-Nov-2016	1.1.13	14-May-2019	EOL	27-Jun-2019
.NET Core 1.0	27-Jun-2016	1.0.16	14-May-2019	EOL	27-Jun-2019

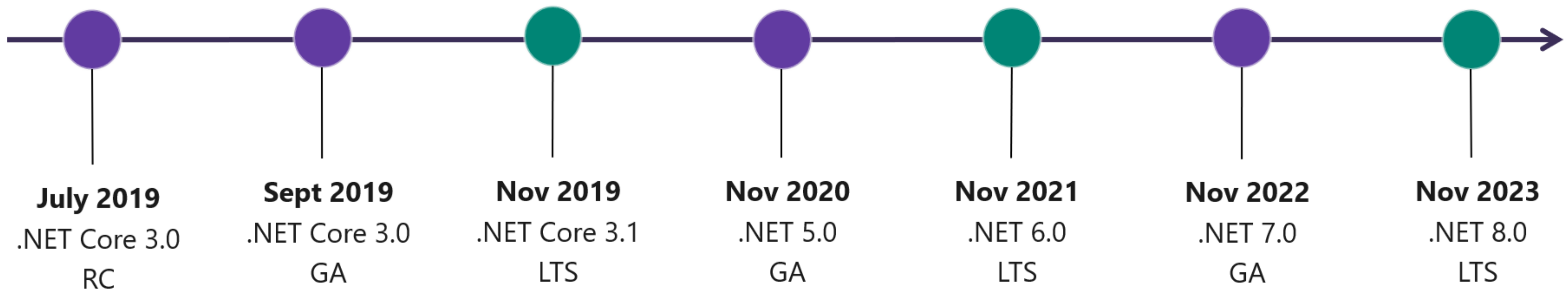
* These dates are taken in March 2020

* EOL: End of Life: End of support refers to the date when Microsoft no longer provides fixes, updates, or online technical assistance.

* Highlighted in RED are out of support

* Source: https://dotnet.microsoft.com/platform/support/policy/dotnet-core

.NET Roadmap



Milestone	Release Date
.NET Core 2.1.x (servicing)	LTS (Long Term Support) release. Approximately every 1-2 months or as needed.
.NET Core 3.1.x (servicing)	LTS (Long Term Support) release. Approximately every 1-2 months or as needed.
.NET 5.0	Release scheduled for November 2020
.NET 6.0	LTS (Long Term Support) release, scheduled for November 2021
.NET 7.0	Release scheduled for November 2022
.NET 8.0	LTS (Long Term Support) release, scheduled for November 2023

.NET Core Repositories

- https://github.com/dotnet/runtime
  Contains the code to build the .NET Core runtime, libraries and shared host (dotnet) installers for all supported platforms, as well as the sources to .NET Core runtime and libraries.

- https://github.com/dotnet/sdk
  Contains core functionality needed to create .NET Core projects, that is shared between Visual Studio and CLI.

- https://github.com/dotnet/aspnetcore
  Contains the code for asp.net core plus the following:
    - Documentation - documentation sources for https://docs.microsoft.com/aspnet/core/
    - Entity Framework Core - data access technology
    - Extensions - Logging, configuration, dependency injection, and more.

Other .NET Repositories

- https://github.com/aspnet/AspNetWebStack
  Code for ASP.NET MVC 5.x, Web API 2.x, and Web Pages 3.x. For ASP.NET Core MVC

- https://github.com/dotnet/winforms
  Code for WinForms, Windows Forms is a UI framework for building Windows desktop applications.

- https://github.com/dotnet/ef6
  codebase for Entity Framework 6 (previously maintained at https://entityframework.codeplex.com).

Additional Repos

There are more repos can be found under the following organizations

- https://github.com/dotnet
- https://github.com/aspnet
- https://github.com/microsoft
- https://github.com/azure
- https://github.com/sharepoint

Module Summary

- In this module, you learnt the following:
  - .NET Platform Overview
  - .NET Standard
  - .NET Core CLI
  - .NET Core Global Tools
  - .NET Runtimes
  - .NET Core Support Lifecyle
  - .NET Roadmap
  - .NET Repositories

Microsoft