

# Platformy programistyczne: .NET i Java

---

WYKŁAD 4: WPROWADZENIE DO PLATFORMY .NET C.D.

# W poprzednim odcinku

---

Część 1:

- Agile Manifesto
- Software Craftsmanship
- Keep Calm(s) and DevOps

Część 2:

- DevOps w praktyce

Część 3:

- [zakres nieznany i niezapowiedziany]



# Agenda

---

Część 1:

- .NET na żywo
- Powiew świeżości w świecie .NET-a
- Technologie .NET

Część 2:

- Live coding ASP.NET Core

Część 3:

- [zakres nieznany i niezapowiedziany]





Demo Time!



Powiew świeżości...

Co to takięgo?

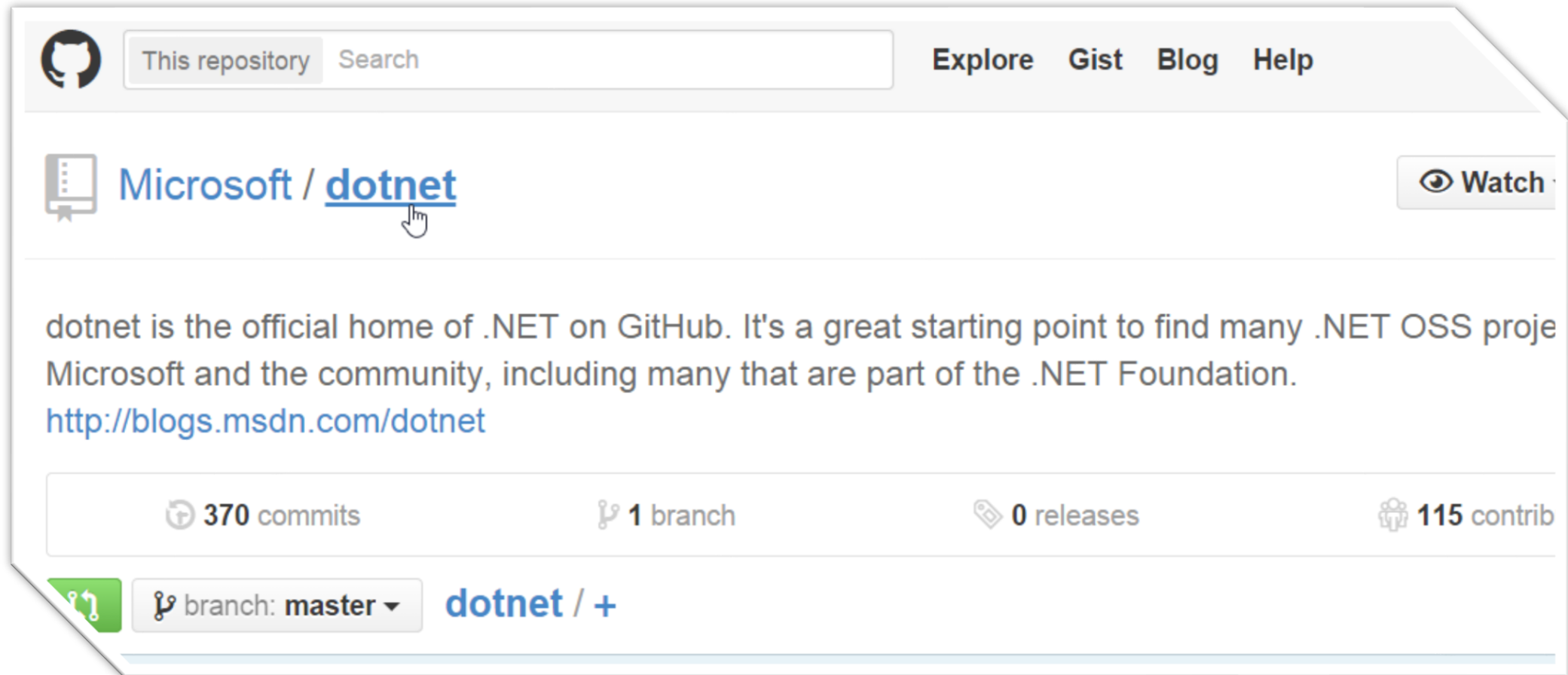
---

Cross Platform Open  
Source Development...

**.NET**



# Coś się zmieniło...



The screenshot shows the GitHub repository page for Microsoft/dotnet. At the top, there is a search bar with the text "This repository" and a search icon. To the right of the search bar are navigation links: "Explore", "Gist", "Blog", and "Help". Below the search bar, the repository name "Microsoft / dotnet" is displayed, with a hand cursor pointing to the "dotnet" link. To the right of the repository name is a "Watch" button. The main content area contains a description: "dotnet is the official home of .NET on GitHub. It's a great starting point to find many .NET OSS projects from Microsoft and the community, including many that are part of the .NET Foundation." Below the description is a link to the official blog: "http://blogs.msdn.com/dotnet". At the bottom of the repository page, there are statistics: "370 commits", "1 branch", "0 releases", and "115 contributors". In the bottom left corner, there is a green button with a plus sign and a dropdown menu showing "branch: master". To the right of the dropdown menu is the text "dotnet / +".

This repository Search

Explore Gist Blog Help

Microsoft / dotnet Watch

dotnet is the official home of .NET on GitHub. It's a great starting point to find many .NET OSS projects from Microsoft and the community, including many that are part of the .NET Foundation.

<http://blogs.msdn.com/dotnet>

370 commits 1 branch 0 releases 115 contributors

branch: master dotnet / +

„There are only two hard things in  
Computer Science: cache invalidation and  
naming things“

---

*PHIL KARLTON*



Łamiąca wiadomość...



# Scott Hanselman, 19 stycznia 2016:

---

„So we're changing the name and picking a better version number.

REINTRODUCING ASP.NET CORE 1.0 AND .NET CORE 1.0

- ASP.NET 5 is now ASP.NET Core 1.0.
- .NET Core 5 is now .NET Core 1.0.
- Entity Framework 7 is now Entity Framework Core 1.0 or EF Core 1.0 colloquially.

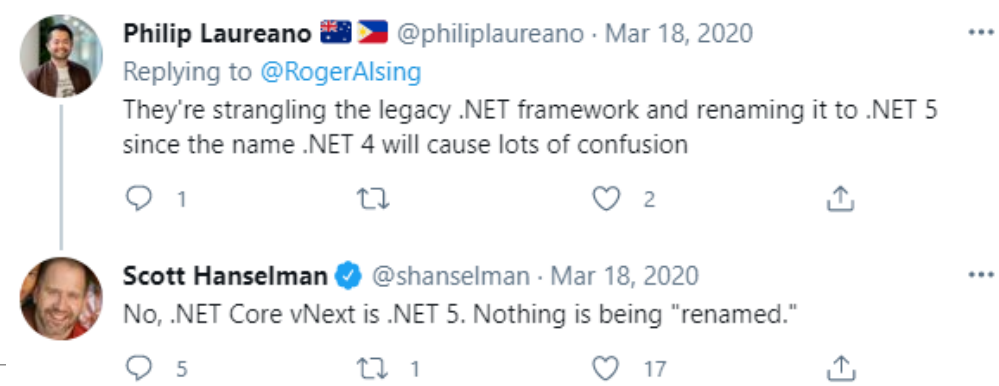
Why 1.0? Because these are new. The whole .NET Core concept is new.”

Łamiąca wiadomość 2...



# Scott Hanselman, 18 marca 2020

---

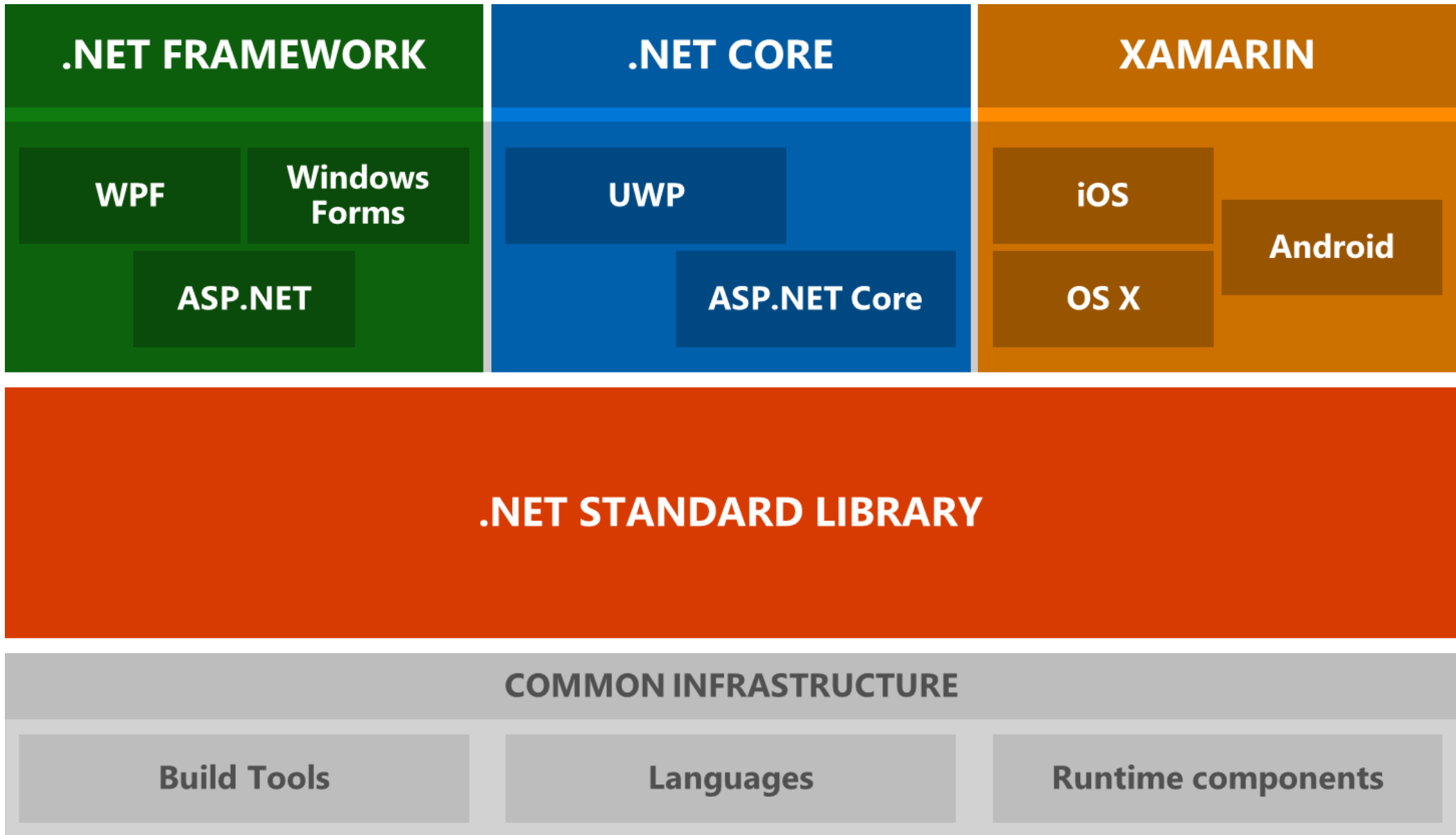


„.NET 5.0 is the next major release of .NET Core following 3.1. We named this new release .NET 5.0 instead of .NET Core 4.0 for two reasons:

We skipped version numbers 4.x to avoid confusion with .NET Framework 4.x.

We dropped "Core" from the name to emphasize that this is the main implementation of .NET going forward. .NET 5.0 supports more types of apps and more platforms than .NET Core or .NET Framework.

ASP.NET Core 5.0 is based on .NET 5.0 but retains the name "Core" to avoid confusing it with ASP.NET MVC 5. Likewise, Entity Framework Core 5.0 retains the name "Core" to avoid confusing it with Entity Framework 5 and 6.”

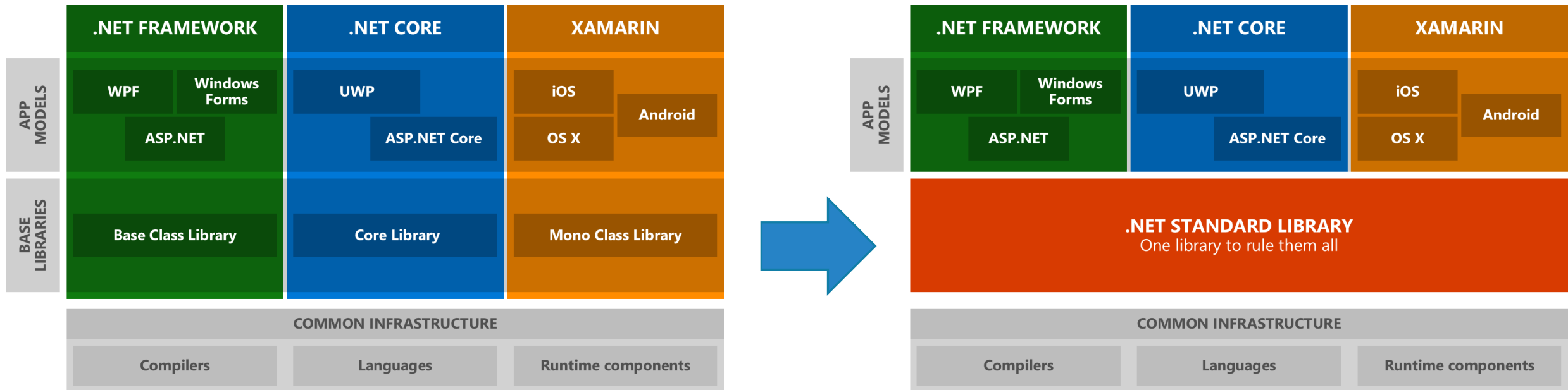


# Środowiska uruchomieniowe

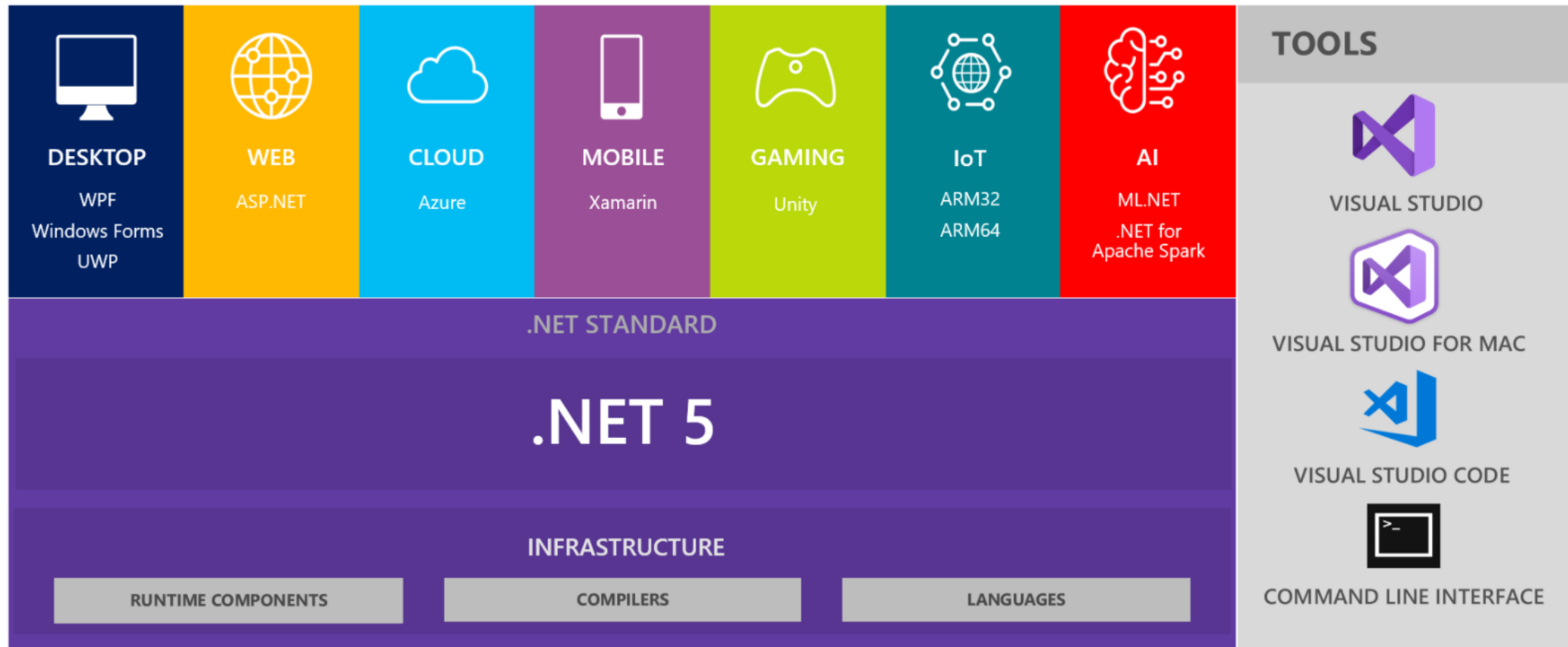
---

- **.NET Framework** – pełna wersja środowiska, kompatybilna z poprzednimi, zawarta w Windows 10. Rozwijana od 2002 roku. Implementuje .NET Standard Library. Zawiera dodatkowe API specyficzne dla systemu Windows – przez to obsługuje WPF i Windows Forms.
- **.NET Core** – nowy framework, open source, modularny, docelowo wspierany na różnych platformach (Windows, Linux, Mac OSX). Wykorzystywany przez aplikacje ASP.NET Core oraz UWP. Implementuje .NET Standard Library
- **Mono for Xamarin** – środowisko wykorzystywane przez aplikacje Xamarin. Historycznie powstało jako open source'owa wersja standardowego .NET Framework. Implementuje .NET Standard Library. Zoptymalizowana pod kątem aplikacji na środowiska iOS i Android.

# .NET Standard Library



# .NET Unified platform





# .NET Standard

- Zestaw API, które będą implementowane przez wszystkie platformy
- Usystematyzowanie i ułatwienie pracy deweloperom: ma na celu rozwiązanie problemu współdzielenia bibliotek/kodu pomiędzy różnymi platformami
- .NET Standard 2.0 zastąpi PCL (Portable Class Libraries) jako narzędzie do budowania cross-platformowych bibliotek

<b>.NET Standard</b>	<b>1.0</b>	<b>1.1</b>	<b>1.2</b>	<b>1.3</b>	<b>1.4</b>	<b>1.5</b>	<b>1.6</b>	<b>2.0</b>	<b>2.1</b>
.NET Core and .NET 5	1.0	1.0	1.0	1.0	1.0	1.0	1.0	2.0	3.0
.NET Framework <sup>1</sup>	4.5	4.5	4.5.1	4.6	4.6.1	4.6.1 <sup>2</sup>	4.6.1 <sup>2</sup>	4.6.1 <sup>2</sup>	N/A <sup>3</sup>
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

# Wsparcie narzędziowe

---

- **Kompilatory**
  - **Roslyn** – kompilator do kodu pośredniego (IL), open source, wykorzystywany zarówno w przypadku .NET Framework oraz .NET Core
  - **RyuJIT** – nowy kompilator Just-in-Time dla środowiska .NET x64, zoptymalizowany ze względu na szybkość uruchamiania i działania, wspiera instrukcje SIMD (single instructions, multiple data)
  - **.NET Native** – kompiluje kod C# do natywnego kodu maszynowego uruchamianego przy minimalistycznej wersji CLR
- .NET project system ("csproj", "vbproj", "fsproj")
- MSBuild – platformowa budowania projektów
- NuGet – Microsoftowy, open-sourcowy manager pakietów
- .NET CLI

# RyuJIT

---



りゅう

---

Japanese [\[edit\]](#)

---

**Noun** [\[edit\]](#)

りゅう (*romaji ryū*)

1. 流: style
2. 竜, 龍: dragon

§ **Proper noun** [\[edit\]](#)

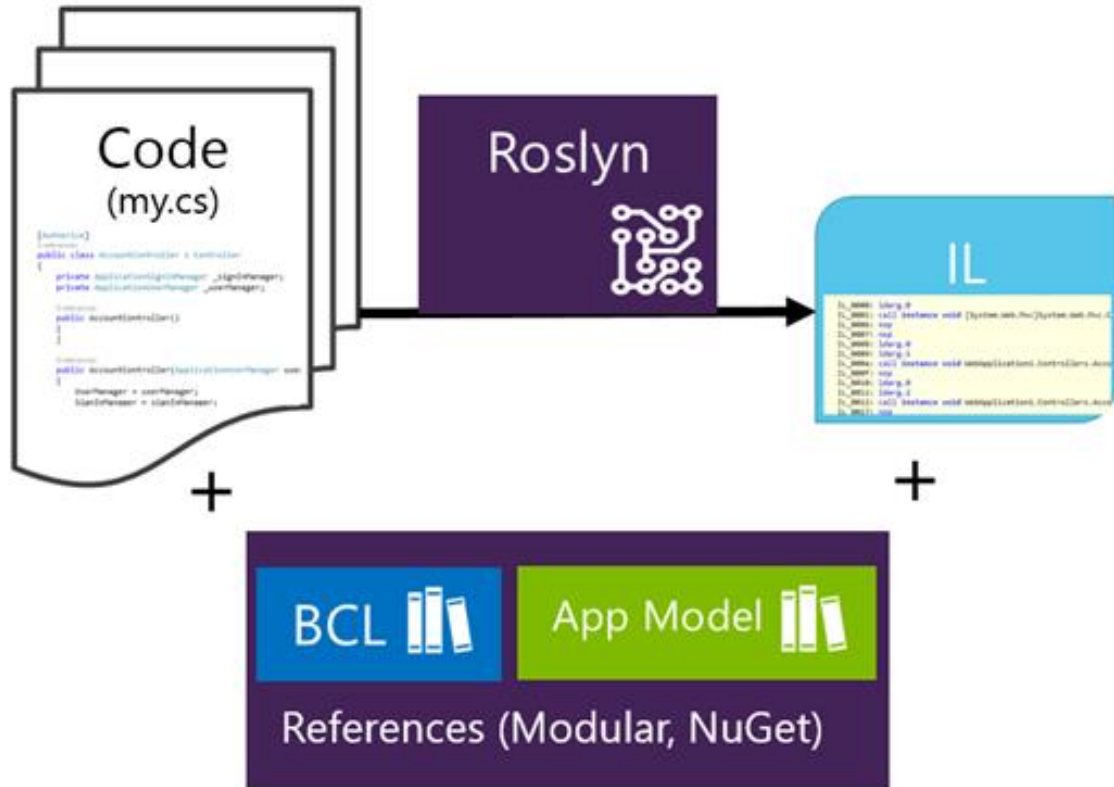
りゅう (*romaji Ryū*)

1. 竜: A male *given name*
  2. 龍: A male *given name*
  3. 隆: A male or female *given name*
  4. 龍: A *surname*.
-

# Code / Build / Debug

# Deploy & Run

Roslyn takes your code and compiles it to IL. You have very modular references to the BCL and App Model you're targeting.



References are built with your app into one native dll deployed locally with runtime

WinStore  
ASP.NET



References & CoreCLR are deployed with app locally, JIT compilation on start up

# Nowe, nowe, nowe...

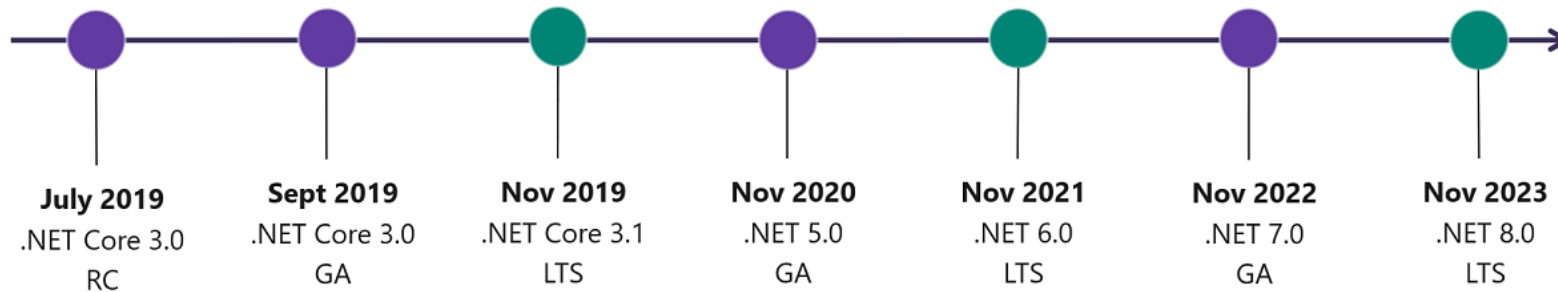
---

- **Visual Studio 2019**
  - Debugger improvements, One-click code cleanup, Visual Studio Live Share, Visual Studio IntelliCode
- **.NET Core 3.0**
  - C# 8.0 support: Ranges and indices, Async streams, Using Declarations, Switch Expressions
  - IEEE Floating-point improvements
  - Local dotnet tools
  - Windows desktop
  - Fast built-in JSON support
  - TLS 1.3 & OpenSSL 1.1.1
  - Cryptography
  - GPIO Support for Raspberry Pi
- **Operating system support**
  - Windows Client: 7, 8.1, 10 (1607+)
  - Windows Server: 2012 R2 SP1+
  - macOS: 10.12+
  - RHEL: 6+, Fedora: 26+, Ubuntu: 16.04+, Debian: 9+, SLES: 12+, openSUSE: 42.3+, Alpine: 3.8+
- **Chip support**
  - x64 on Windows, macOS, and Linux
  - x86 on Windows
  - ARM32 on Windows and Linux
  - ARM64 on Linux

# The future

---

## .NET Schedule



- .NET Core 3.0 release in September
- .NET Core 3.1 = Long Term Support (LTS)
- .NET 5.0 release in November 2020
- Major releases every year, LTS for even numbered releases
- Predictable schedule, minor releases if needed



# Technologie w .NET Framework

### Services

- ASP.NET Web API (Cloud optimized)
- ASP.NET SignalR (Cloud optimized)
- WCF
- WCF Data Services
- LightSwitch OData Services
- OData Lib
- Service Bus for Windows Server Lib
- Service Bus for Microsoft Azure Lib

### Windows Store Apps

- Universal Windows Apps
- .NET Native
- WinRT XAML/.NET Windows Store SDK
- p&p Prism for Windows Runtime
- p&p Unity for Windows Store Apps

### Windows Phone Store Apps

- Universal Windows Apps
- .NET Native
- Windows Phone SDK

### Cloud Apps

- Microsoft Azure .NET SDKs
- Microsoft Azure Storage Lib
- Microsoft Azure Configuration Manager Lib
- Microsoft Azure Media Services .NET SDK
- Microsoft Azure Mobile Services
- p&p Autoscaling App Block
- p&p Transient Fault Handling App Block

### Core

Get the .NET technology guide

- .NET Runtimes
- .NET Compiler Platform ("Roslyn")
- Languages (C#, VB, F#)
- Base Class Library

### Partners Cross Device Apps

- Xamarin
- ITR-Mobility IFactr
- Citrix Mobile SDK for Windows Apps

### Internet of Things (IoT)

- .NET Micro Framework
- .NET Compact Framework

### Web Apps

- ASP.NET MVC (Cloud optimized)
- ASP.NET Web Pages (Cloud optimized)
- ASP.NET Web Forms
- LightSwitch HTML5 Client

### Desktop Apps

- Windows Presentation Foundation
- Windows Forms
- p&p Prism for WPF

### .NET Extension Libs

- Async
- HttpClient
- Immutable Collections
- TPL Dataflow
- Rx (Reactive Extensions)
- Ix (Interactive Extensions; Ix-Async)
- Fsharp Core
- WF Activities Extensions
- p&p Semantic Logging App Block
- Portable Class Libraries
- p&p EntLib - Validation App Block
- p&p EntLib - Exception Handling App Block
- p&p EntLib - Logging App Block
- Compression

### Data Access

- Entity Framework
- ADO.NET
- ASP.NET Universal Providers
- .NET Map Reduce API for Hadoop
- .NET API for Hadoop WebClient
- Linq to Hive
- Linq to Sql
- p&p Data Access App Block

### DI and IoC Containers

- Composition (MEF2)
- MEF (Managed Extensibility Framework)
- p&p Unity

### Caching

- Microsoft Azure Caching
- Windows Server AppFabric Caching
- Microsoft Azure Caching Memcache Shim
- ASP.NET Cache

### Security

- ASP.NET Identity
- DotNetOpenAuth
- Windows Identity Foundation
- Authorization Manager (AzMan)
- Web Protection Library
- OWIN Authentication Middleware
- Microsoft Azure AD

### Emerging Application Patterns

### Established Application Patterns

### Cross-Cutting Patterns

- NuGet Package
- Open Source
- Support (MS Official)

Like it? Get it.





# Poster dziś...

**DISCOVER what's new**  
Visual Studio 2017

**Advanced toolchain**  
Build apps with the new toolchain, production quality, faster build and supporting new C++ and C# code you need for advanced in the code you are working for.

**Test code navigation**  
Navigate your code to quickly find tests and run. The ability to navigate code is a key feature of Visual Studio 2017.

**Use code analysis**  
Use the new code analysis tool to find errors in your code. You can get more detailed feedback on code quality in your IDE. It's now possible to analyze code in real-time, allowing you to catch errors before they happen. You can also see code quality and style tips.

**Source Profiler**  
Source Profiler is a tool for profiling your code. It helps you understand how your code is performing. It can help you find bottlenecks and optimize your code. It's now possible to profile code in real-time, allowing you to see the performance of your code as you work.

**New Start page**  
The new Start page is a central hub for your development workflow. It provides a quick overview of your projects, recent files, and other useful information. It's now possible to customize the Start page to suit your needs.

**Open folder**  
Open folders and projects with a single click. This makes it easier to navigate through your workspace and get started with your work.

**Live Unit Testing**  
Run your unit tests as you code. This allows you to see the results of your tests as you write your code, helping you catch errors early and improve the quality of your code.

**NET Core CLI reference**  
Access the .NET Core CLI reference directly from Visual Studio. This makes it easier to find the commands you need to build and run your .NET Core applications.

**Extensibility**  
Make Visual Studio your own. Visual Studio 2017 is designed to be extended. You can use the Visual Studio API to create custom tools, extensions, and integrations that enhance your development experience.

**Workloads**  
Select a workload for streamlined options, some, none, and make your own. Visual Studio 2017 offers several workloads that are tailored to different development scenarios. You can also create your own workload to suit your needs.

**Debugging and support**  
Better debugging? Yes, please! Visual Studio 2017 has several new debugging features that make it easier to find and fix bugs in your code. It also has a new support system that makes it easier to get help when you need it.

**Workloads you can install and configure**  
Visual Studio 2017 offers several workloads that you can install and configure to suit your needs. These include:

- ASP.NET and web development
- C++/C# desktop development
- Cloud and mobile development
- Game development
- IoT development
- Machine Learning
- Mobile development
- Python development
- R development
- SQL Server development
- System Center development
- Web development

**Learn more**  
Visit [visualstudio.com](http://visualstudio.com) for more information on the new features and workloads in Visual Studio 2017.

**Language improvements**

**NET Standard**  
Get in the merge lane: all your .NET APIs, together. The .NET Standard is a set of APIs that are shared across all .NET implementations. This makes it easier to write code that can run on any .NET implementation, reducing the need for platform-specific code.

**Visual Studio Mobile Center**  
Monitor, manage, and update your mobile apps. Visual Studio Mobile Center is a cloud-based service that allows you to monitor the performance of your mobile apps, manage updates, and troubleshoot issues. It's now possible to manage your mobile apps directly from Visual Studio.

**Visual Studio Team Services**  
Share code, track work, and ship software for any language. Visual Studio Team Services is a cloud-based service that allows you to share code, track work, and ship software for any language. It's now possible to use Visual Studio Team Services to manage your development workflow.

**Git commands**  
A collection of Git commands and their usage. This poster provides a quick reference for common Git commands, including `git init`, `git add`, `git commit`, `git push`, and `git pull`.

**TypeScript**  
Developed by Microsoft, TypeScript is a typed superset of JavaScript that compiles to plain JavaScript. This poster provides a quick overview of the TypeScript language and its features.

**Visual Basic**  
Visual Basic is a multi-paradigm programming language that is easy to learn and use. This poster provides a quick overview of the Visual Basic language and its features.

**Azure 2.0 CLI**  
The Azure 2.0 CLI is a command-line interface for managing Azure resources. This poster provides a quick overview of the Azure 2.0 CLI and its features.

**C# 7**  
C# 7 introduces several new features, including tuple literals, pattern matching, and the `await` keyword. This poster provides a quick overview of the C# 7 language and its features.

**F#**  
F# is a functional programming language that is designed to be easy to learn and use. This poster provides a quick overview of the F# language and its features.

**C++**  
C++ is a general-purpose programming language that is designed to be easy to learn and use. This poster provides a quick overview of the C++ language and its features.

**Default keyboard shortcuts**  
A list of default keyboard shortcuts for Visual Studio 2017. This poster provides a quick reference for common keyboard shortcuts, including `Ctrl+C`, `Ctrl+V`, and `Ctrl+S`.

**Give us your feedback**  
Got a suggestion, bug report, or comment? This is how you can help us improve Visual Studio. Visit [visualstudio.com/feedback](http://visualstudio.com/feedback) to provide your feedback.

# UI - Windows Forms

---

- Najstarsza i jeszcze popularna technologia tworzenia aplikacji typu rich-client w .NET Framework
- Pozwala na szybkie i proste tworzenie aplikacji desktopowych
- Wyświetlanie niestandardowych kontrolek jest oparte na GDI+, co skutkuje niską wydajnością
- Technologia nie jest przystosowana do dynamicznej zmiany layoutu
- Aktualnie wypierana przez WPF, który będzie (być może) wypierany przez UWP

# UI - Silverlight

---

- Technicznie rzecz biorąc Silverlight jest osobnym frameworkiem, a nie podzbiór kluczowych elementów
- Aplikacja może być uruchamiana jako plugin przeglądarki
- Model graficzny jest podzbiorem WPF (Windows Presentation Foundation)
- Technologia na wymarciu z punktu widzenia tworzenia aplikacji



# UI – WPF

---

- Windows Presentation Foundation – wprowadzone w .NET Framework 3.0
- Posiada wbudowane mechanizmy do bardziej wyrafinowanych operacji graficznych, transformacji, renderingu 3D, obsługi przezroczystości
- O wiele większe wsparcie (w porównaniu do Windows Forms) przy tworzeniu dynamicznych layoutów (bardzo ważne przy internacjonalizacji aplikacji)
- Wykorzystuje sprzętowe wsparcie do renderowania zawartości (DirectX). Jednak bez niego wymaga dużych zasobów i może działać wolniej
- Interfejs użytkownika jest definiowanych w XAML-u

# UI - Universal Windows Platform

„Windows 10 introduces the Universal Windows Platform (UWP), which provides a common app platform available on every device that runs Windows 10. The UWP provides a guaranteed core API across devices.”



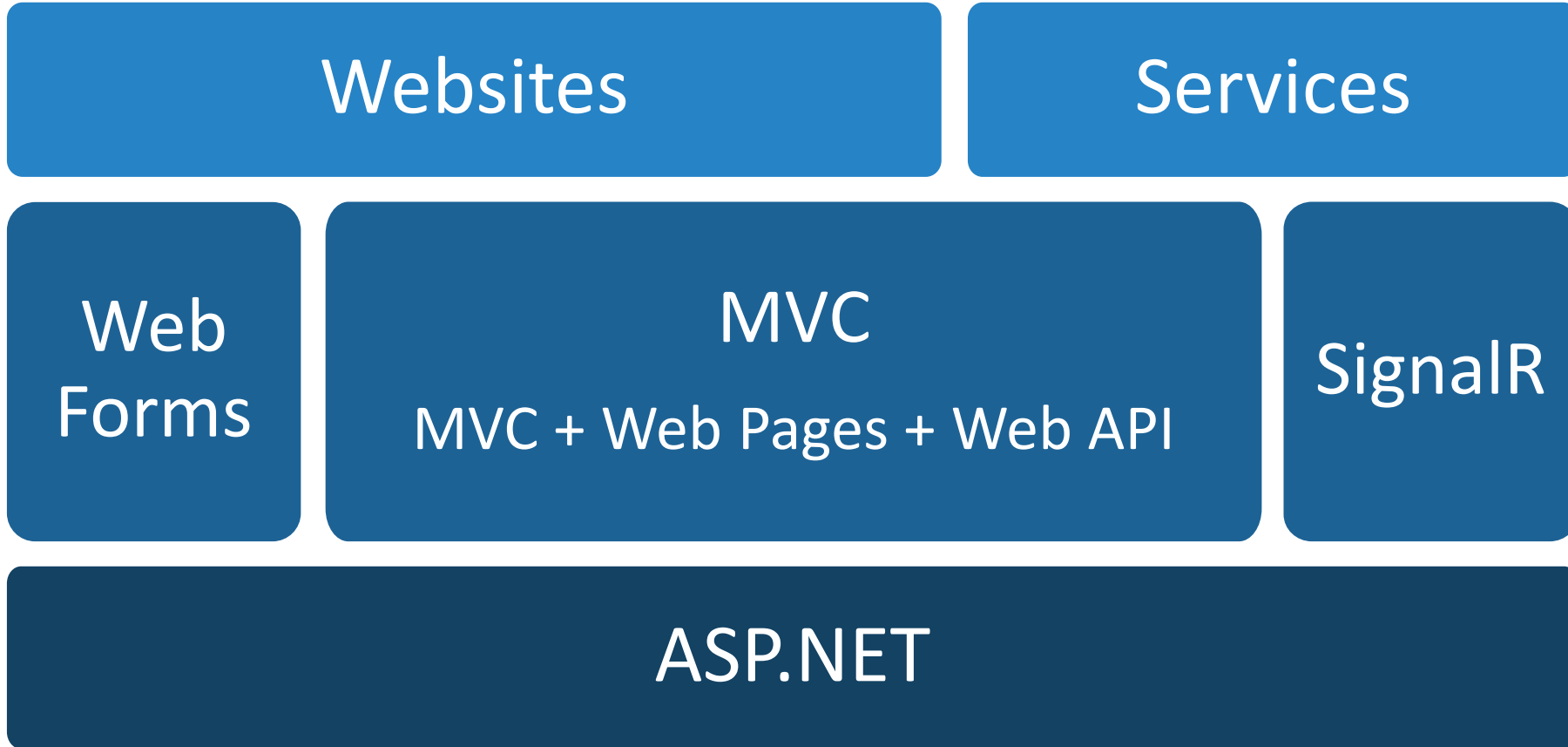
# UI - ASP.NET

---

- Aplikacja uruchamiana może być w wielu środowiskach np. Windows IIS (Internet Information Services), przy użyciu wbudowanego serwera Kestrel (ASP.NET Core)
- Aplikacja jest dostępna przez dowolną przeglądarkę WWW
- W porównaniu do aplikacji typu rich-client:
  - Użytkownicy nie muszą nic dodatkowo instalować
  - Aplikacja dostępna jest na dowolnej platformie
  - Wszelkie zmiany muszą być aktualizowane tylko po stronie serwera
- Infrastruktura ASP.NET przewiduje kilka sposobów tworzenia aplikacji webowych

# UI - ASP.NET

---



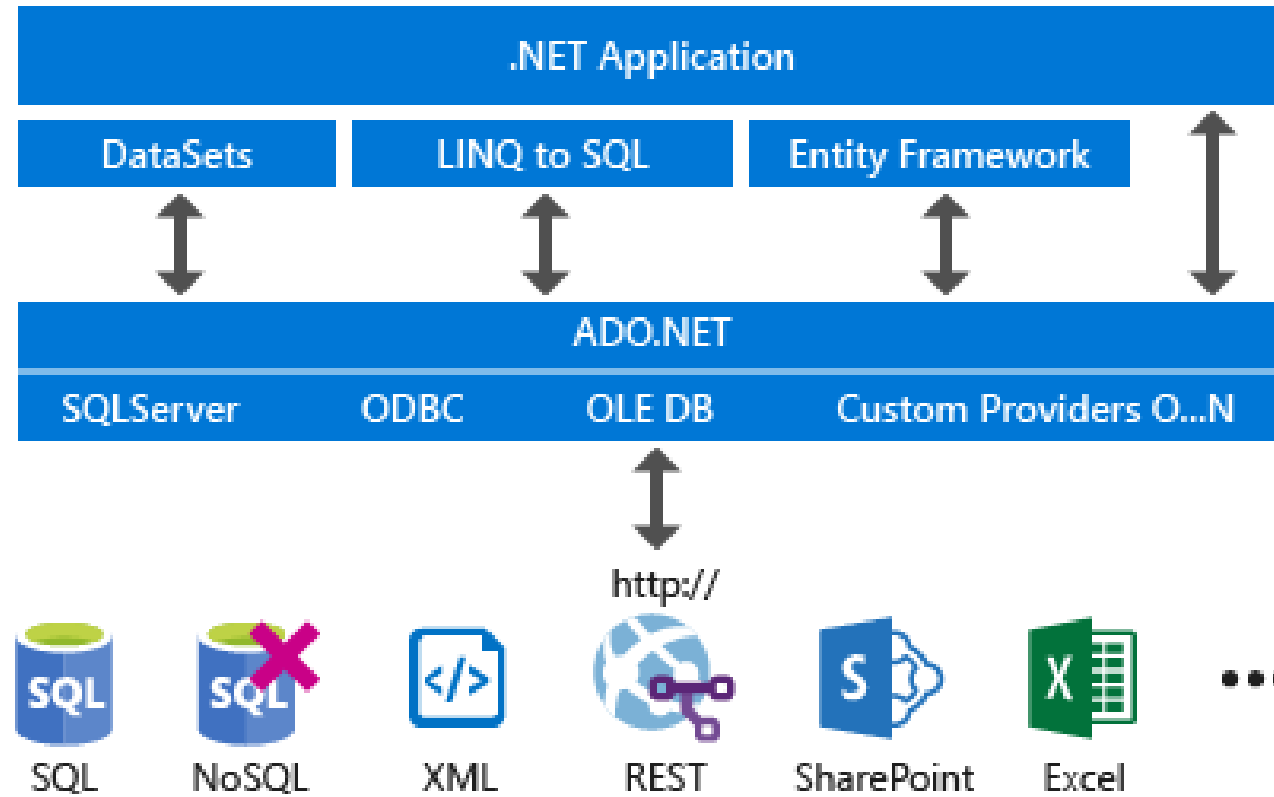
# Backend – ADO.NET

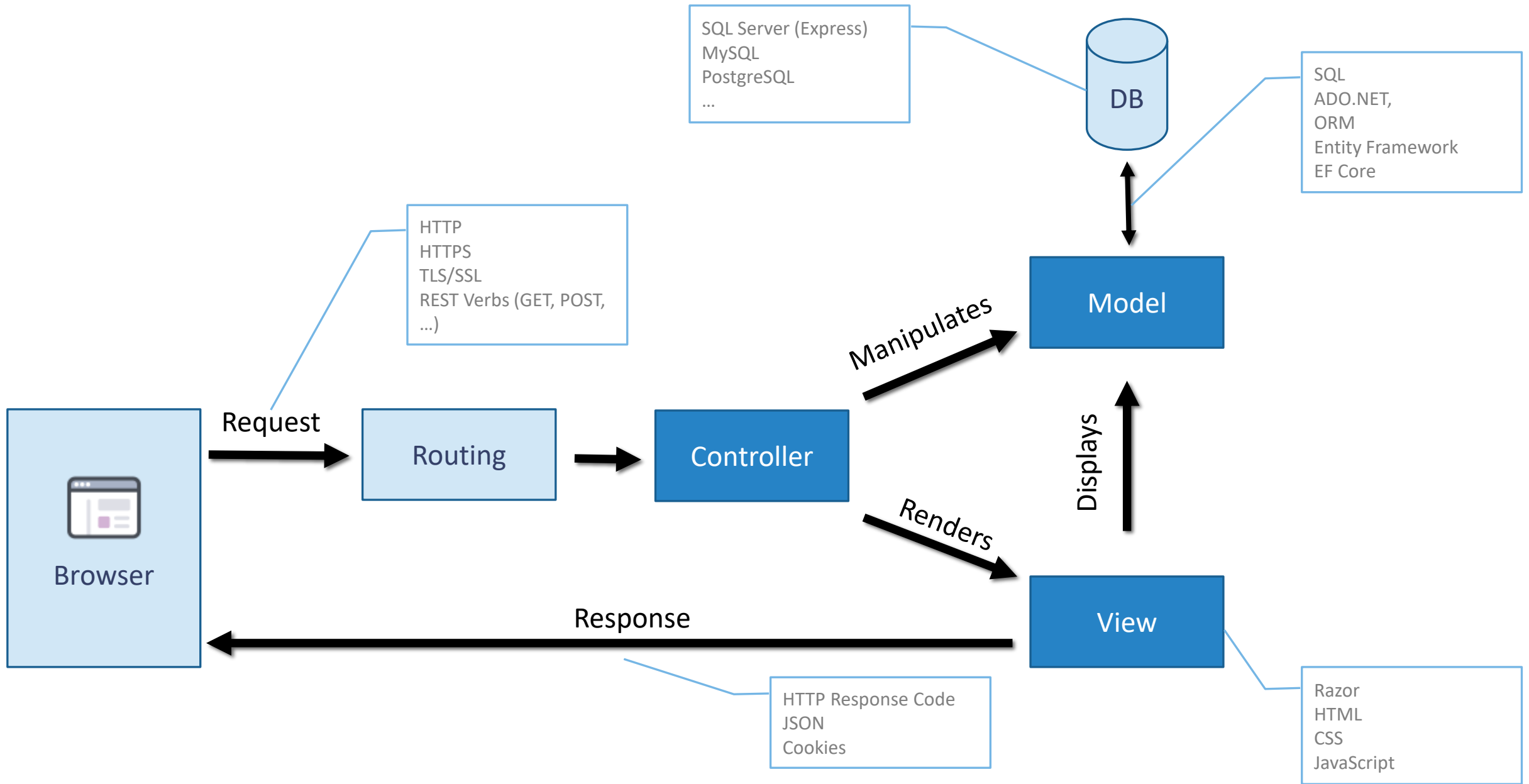
---

- Zunifikowany sposób dostępu do danych w .NETcie. Następca ADO, lecz zaimplementowany od podstaw
- Wyróżniane są dwie warstwy:
  - Provider level – mechanizmy dostępu do bazy danych, natywne wsparcie dla MS SQL Server, Oracle, OLE-DB, ODBC
  - Model DataSet – mechanizm utrzymywania kopii danych w pamięci, co pozwala zmniejszyć liczbę odwołań do serwera bazy, zwiększyć skalowalność oraz wydajność aplikacji typu rich-client
- Dodatkowe mechanizmy dostępu zawierające ORM (Object/relational mapper – pozwala na automatyczne mapowanie obiektów do wierszy w bazie danych):
  - LINQ to SQL- prostszy, szybszy, już coraz rzadziej używany
  - Entity Framework – bardziej elastyczny, nadrobił braki i teraz to podstawowy ORM



# Backend – ADO.NET







Demo Time!

Czas na...



niezapowiedzianą kartkówkę...