

# Introduction to ASP.NET Core

04/17/2020 • 8 minutes to read •  +1

## In this article

[Why choose ASP.NET Core?](#)

[Build web APIs and web UI using ASP.NET Core MVC](#)

[Client-side development](#)

[ASP.NET Core target frameworks](#)

[Recommended learning path](#)

[Migrate from .NET Framework](#)

[How to download a sample](#)

[Next steps](#)

By [Daniel Roth](#), [Rick Anderson](#), and [Shaun Luttin](#)

ASP.NET Core is a cross-platform, high-performance, [open-source](#) framework for building modern, cloud-enabled, Internet-connected apps. With ASP.NET Core, you can:

- Build web apps and services, [Internet of Things \(IoT\)](#) apps, and mobile backends.
- Use your favorite development tools on Windows, macOS, and Linux.

- Deploy to the cloud or on-premises.
- Run on [.NET Core](#).

## Why choose ASP.NET Core?

Millions of developers use or have used [ASP.NET 4.x](#) to create web apps. ASP.NET Core is a redesign of ASP.NET 4.x, including architectural changes that result in a leaner, more modular framework.

ASP.NET Core provides the following benefits:

- A unified story for building web UI and web APIs.
- Architected for testability.
- [Razor Pages](#) makes coding page-focused scenarios easier and more productive.
- [Blazor](#) lets you use C# in the browser alongside JavaScript. Share server-side and client-side app logic all written with .NET.
- Ability to develop and run on Windows, macOS, and Linux.
- Open-source and [community-focused](#).
- Integration of [modern, client-side frameworks](#) and development workflows.
- Support for hosting Remote Procedure Call (RPC) services using [gRPC](#).
- A cloud-ready, environment-based [configuration system](#).
- Built-in [dependency injection](#).
- A lightweight, [high-performance](#), and modular HTTP request pipeline.
- Ability to host on the following:
  - [Kestrel](#)
  - [IIS](#)
  - [HTTP.sys](#)
  - [Nginx](#)
  - [Apache](#)
  - [Docker](#)
- [Side-by-side versioning](#).

- Tooling that simplifies modern web development.

## Build web APIs and web UI using ASP.NET Core MVC

ASP.NET Core MVC provides features to build [web APIs](#) and [web apps](#):

- The [Model-View-Controller \(MVC\) pattern](#) helps make your web APIs and web apps testable.
- [Razor Pages](#) is a page-based programming model that makes building web UI easier and more productive.
- [Razor markup](#) provides a productive syntax for [Razor Pages](#) and [MVC views](#).
- [Tag Helpers](#) enable server-side code to participate in creating and rendering HTML elements in Razor files.
- Built-in support for [multiple data formats and content negotiation](#) lets your web APIs reach a broad range of clients, including browsers and mobile devices.
- [Model binding](#) automatically maps data from HTTP requests to action method parameters.
- [Model validation](#) automatically performs client-side and server-side validation.

## Client-side development

ASP.NET Core integrates seamlessly with popular client-side frameworks and libraries, including [Blazor](#), [Angular](#), [React](#), and [Bootstrap](#). For more information, see [Introduction to ASP.NET Core Blazor](#) and related topics under *Client-side development*.

## ASP.NET Core target frameworks

ASP.NET Core 3.x and later can only target .NET Core. Generally, ASP.NET Core is composed of [.NET Standard](#) libraries. Libraries written with .NET Standard 2.0 run on any [.NET platform that implements .NET Standard 2.0](#).

There are several advantages to targeting .NET Core, and these advantages increase with each release. Some advantages of .NET Core over .NET Framework include:

- [Cross-platform](#) Runs on Windows, macOS, and Linux

- Cross platform: runs on Windows, macOS, and Linux.
- Improved performance
- [Side-by-side versioning](#)
- New APIs
- Open source

## Recommended learning path

We recommend the following sequence of tutorials for an introduction to developing ASP.NET Core apps:

1. Follow a tutorial for the app type you want to develop or maintain.

App type	Scenario	Tutorial
Web app	New server-side web UI development	<a href="#">Get started with Razor Pages</a>
Web app	Maintaining an MVC app	<a href="#">Get started with MVC</a>
Web app	Client-side web UI development	<a href="#">Get started with Blazor</a>
Web API	RESTful HTTP services	<a href="#">Create a web API<sup>+</sup></a>
Remote Procedure Call app	Contract-first services using Protocol Buffers	<a href="#">Get started with a gRPC service</a>
Real-time app	Bidirectional communication between servers and connected clients	<a href="#">Get started with SignalR</a>

2. Follow a tutorial that shows how to do basic data access.

Scenario	Tutorial
New development	<a href="#">Razor Pages with Entity Framework Core</a>

Maintaining an MVC app

[MVC with Entity Framework Core](#)

3. Read an overview of ASP.NET Core [fundamentals](#) that apply to all app types.
4. Browse the table of contents for other topics of interest.

†There's also an [interactive web API tutorial](#). No local installation of development tools is required. The code runs in an [Azure Cloud Shell](#) in your browser, and [curl](#) is used for testing.

## Migrate from .NET Framework

For a reference guide to migrating ASP.NET 4.x apps to ASP.NET Core, see [Migrate from ASP.NET to ASP.NET Core](#).

## How to download a sample


Many of the articles and tutorials include links to sample code.

1. [Download the ASP.NET repository zip file](#).
2. Unzip the *Docs-master.zip* file.
3. Use the URL in the sample link to help you navigate to the sample directory.

## Preprocessor directives in sample code

To demonstrate multiple scenarios, sample apps use the `#define` and `#if-#else/#elif-#endif` preprocessor directives to selectively compile and run different sections of sample code. For those samples that make use of this approach, set the `#define` directive at the top of the C# files to define the symbol associated with the scenario that you want to run. Some samples require defining the symbol at the top of multiple files in order to run a scenario.

For example, the following `#define` symbol list indicates that four scenarios are available (one scenario per symbol). The current sample configuration runs the `TemplateCode` scenario:

C#	
<pre>#define TemplateCode // or LogFromMain or ExpandDefault or FilterInCode</pre>	

To change the sample to run the `ExpandDefault` scenario, define the `ExpandDefault` symbol and leave the remaining symbols commented-out:


C#	
<pre>#define ExpandDefault // TemplateCode or LogFromMain or FilterInCode</pre>	

For more information on using [C# preprocessor directives](#) to selectively compile sections of code, see [#define \(C# Reference\)](#) and [#if \(C# Reference\)](#).

## Regions in sample code

Some sample apps contain sections of code surrounded by `#region` and `#endregion` C# directives. The documentation build system injects these regions into the rendered documentation topics.

Region names usually contain the word "snippet." The following example shows a region named `snippet_WebHostDefaults`:

C#	
<pre>#region snippet_WebHostDefaults Host.CreateDefaultBuilder(args)     .ConfigureWebHostDefaults(webBuilder =&gt;     {         webBuilder.UseStartup&lt;Startup&gt;();     } }</pre>	

```
});  
#endregion
```

The preceding C# code snippet is referenced in the topic's markdown file with the following line:

Markdown

 Copy

```
[!code-csharp](sample/SampleApp/Program.cs?name=snippet_WebHostDefaults)]
```

You may safely ignore (or remove) the `#region` and `#endregion` directives that surround the code. Don't alter the code within these directives if you plan to run the sample scenarios described in the topic. Feel free to alter the code when experimenting with other scenarios.

For more information, see [Contribute to the ASP.NET documentation: Code snippets](#).

## Next steps

For more information, see the following resources:

- [Get started with ASP.NET Core](#)
- [Publish an ASP.NET Core app to Azure with Visual Studio](#)
- [ASP.NET Core fundamentals](#)
- [The weekly ASP.NET community standup](#) covers the team's progress and plans. It features new blogs and third-party software.

---

Is this page helpful?

 Yes  No

---