**.NET 5.0, VS2019 Preview and C# 9.0 for ASP .NET Core developers**

By Shahed C on April 6, 2020

[3 Replies](https://wakeupandcode.com/net-5-0-vs2019-preview-and-c-9-0-for-asp-net-core-developers/#comments)



This is the fourteenth of a new [series of posts](https://wakeupandcode.com/aspnetcore/#aspnetcore2020) on ASP .NET Core 3.1 for 2020. In this series, we’ll cover 26 topics over a span of 26 weeks from January through June 2020, titled **ASP .NET Core A-Z!** To differentiate from the [2019 series](https://wakeupandcode.com/aspnetcore/#aspnetcore2019), the 2020 series will mostly focus on a growing single codebase ([NetLearner!](https://wakeupandcode.com/netlearner-on-asp-net-core-3-1/)) instead of new unrelated code snippets week.

Previous post:

* [Middleware in ASP .NET Core 3.1](https://wakeupandcode.com/middleware-in-asp-net-core-3-1/)

**NetLearner on GitHub**:

* Repository: <https://github.com/shahedc/NetLearnerApp>
* v0.14-alpha release: <https://github.com/shahedc/NetLearnerApp/releases/tag/v0.14-alpha>

**NOTE**: The NetLearner suite of apps won’t be updated to .NET 5.0 at this time, so you can check out the new template-generated projects in an experimental subfolder:

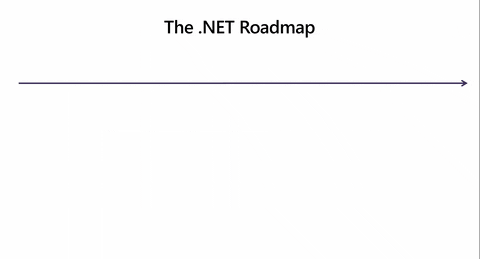
* Experimental projects with .NET 5: <https://github.com/shahedc/NetLearnerApp/tree/main/experimental/NetLearner.DotNet5>

**In this Article:**

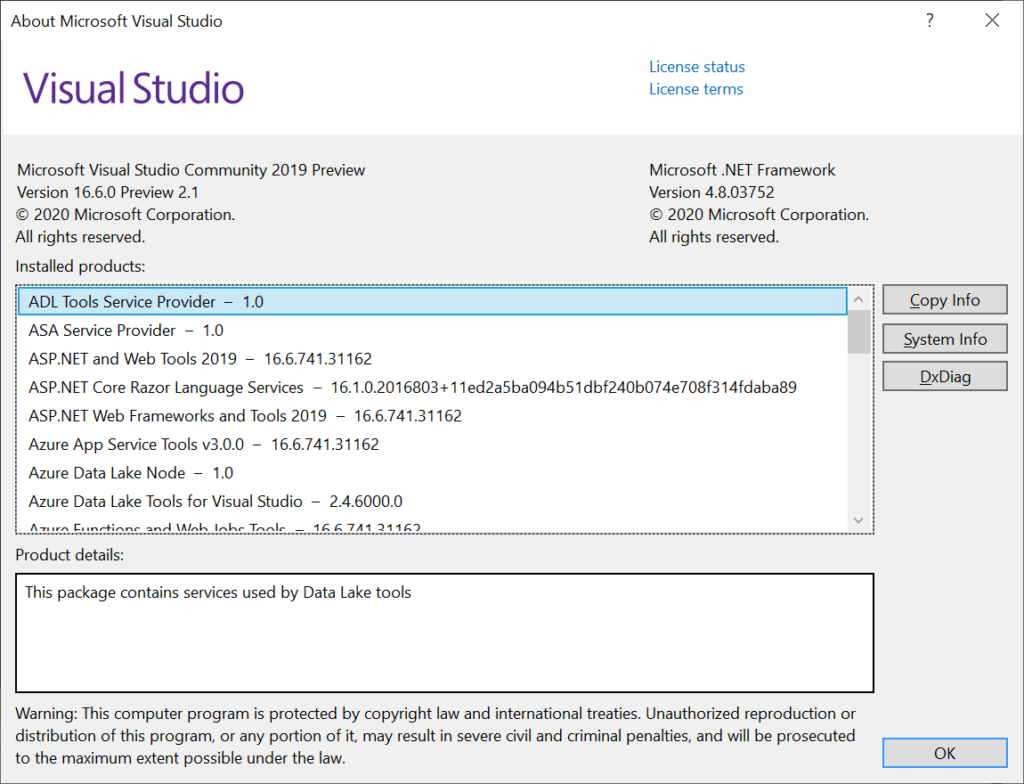
* N is for .NET 5.0, VS2019 Preview and C# 9.0
* Visual Studio 2019 Preview
* .NET 5.0 Preview 2
* C# 9.0 features
* ASP .NET Core 5.0 Project Types
* Migrating from 3.1 to 5.0
* References

**N is for .NET 5.0, VS2019 Preview and C# 9.0**

As we have crossed the halfway point in this series, now is a good time to take a step back and look forward at .NET 5.0. This is the next iteration of .NET Core, which skips over the number 4, and unifies all flavors of .NET into a single .NET (i.e. .NET Core, .NET Framework and Mono).



In order to use .NET 5.0 (Preview 2 as of this writing), we need to install Visual Studio 2019 v16.6.0 Preview 2.1. This will give you access to all the latest (preview) project templates for ASP .NET Core 5.0. Since C# 9.0 is still early in development, we will just briefly touch on upcoming/proposed language features.



**Visual Studio 2019 Preview**

Visual Studio 2019 has been out since early 2019, so you can grab the latest stable version from the main download page. Visit any of the URLs below to select the edition you need:

* VS2019 Downloads: <https://visualstudio.microsoft.com/downloads/>
* [Community Edition](https://visualstudio.microsoft.com/thank-you-downloading-visual-studio/?sku=Community&amp;rel=16)
* [Professional Edition](https://visualstudio.microsoft.com/thank-you-downloading-visual-studio/?sku=Professional&amp;rel=16)
* [Enterprise Edition](https://visualstudio.microsoft.com/thank-you-downloading-visual-studio/?sku=Enterprise&amp;rel=16)

In order to get the latest *Preview* version, you must visit the special download page for Previews:

* VS2019 Preview Downloads: <https://visualstudio.microsoft.com/vs/preview/>
* [Community Edition](https://visualstudio.microsoft.com/thank-you-downloading-visual-studio/?sku=community&amp;ch=pre&amp;rel=16)
* [Professional Edition](https://visualstudio.microsoft.com/thank-you-downloading-visual-studio/?sku=professional&amp;ch=pre&amp;rel=16)
* [Enterprise Edition](https://visualstudio.microsoft.com/thank-you-downloading-visual-studio/?sku=enterprise&amp;ch=pre&amp;rel=16)

As before, the Community Edition (comparable to Pro) is free for students, open-source contributors and individuals. The Pro and Enterprise editions add additional products and services from small teams to enterprise companies.

*But wait!* What if you can’t stay online for the length of the installation or need to reinstall quickly at a later date? If you need an offline installer, check out the instructions on the following page:

* Create an offline installation: <https://docs.microsoft.com/en-us/visualstudio/install/create-an-offline-installation-of-visual-studio?view=vs-2019>

What are some cool new and improved features to be aware of? There are so many that I stitched together a series of tweets from [Amanda Silver](https://twitter.com/amandaksilver) (CVP of Product for Developer Tools) and created the following thread in April 2019:

* Twitter thread: <https://twitter.com/shahedC/status/1113177299652837376>

The aforementioned thread highlights the following features. Click each hyperlink in the list below for more info on each.

* [Live Share](https://visualstudio.microsoft.com/services/live-share/): Available as an extension in VS Code, Live Share is installed by default with VS2019. Easily collaborate with other developers while coding in real-time!
* [Intellicode](https://docs.microsoft.com/en-us/visualstudio/intellicode/): Use AI to write better code. Choose to share what you want with others or keep things private.
* [Git-first workflows](https://devblogs.microsoft.com/visualstudio/get-to-code-how-we-designed-the-new-visual-studio-start-window/): Choose to create a new project from a source code repo or use a template. The new start window provides more options up front.
* [Debug search](https://devblogs.microsoft.com/visualstudio/enhanced-in-visual-studio-2019-search-for-objects-and-properties-in-the-watch-autos-and-locals-windows/): Search while debugging. Type in search filters in the Watch, Locals, and Autos panels.
* [Snapshot debugging](https://docs.microsoft.com/en-us/visualstudio/debugger/debug-live-azure-applications?view=vs-2019): Available in the Enterprise Edition, snapshot debugging allows you to get a snapshot of your app’s execution after deployment. This includes cloud deployments, Azure VMs and Kubernetes containers.
* [VS Search](https://twitter.com/amandaksilver/status/1113171409482596352): Dynamic search results include commands, menus, components and templates. Note that this was formerly know as *Quick Launch*.
* [App Service Debugging](https://twitter.com/amandaksilver/status/1113171680422236162): Attach the debugger to your app running in Azure App Service!
* [App Service Connectivity](https://twitter.com/amandaksilver/status/1113171890955202560): Connect your web app to Azure App Service with ease, including App Insights monitoring.
* [Azure Monitor](https://twitter.com/amandaksilver/status/1113172181163290624): Use Azure Monitor to get additional insight on your deployed app!

If you prefer to sit back and relax and just watch a recap of the launch announcements, I put together a handy list of YouTube videos from the VS2019 launch event. This playlist kicks off with the 50-minute keynote, is followed by a string of videos and ends with a cumulative 7-hour video if you prefer to watch all at once.

* VS2019 Launch event playlist: [https://www.youtube.com/watch?v=DANLUUIUrcM&list=PLLPmfQOWse5Gr4pMcyFmwHW-hDUWmbT6B](https://www.youtube.com/watch?v=DANLUUIUrcM&amp;list=PLLPmfQOWse5Gr4pMcyFmwHW-hDUWmbT6B)

So, what’s new in the latest version of Visual Studio 2019 v16.6 Preview 2?

* **Version Control**: improved Git functionality, including quicker GitHub hosting with a single-click
* **Snapshot Debugging**: improved Snapshot Debugger experience, without requiring a restart when using Snapshot Debugger with Azure App Service
* **.NET Async:** new .NET Async tool (included with Performance Profiler) to help with better understanding and also optimizing your async/await code
* **JavaScript/TypeScript Debugging**: improved JS/TS debugger, with support for debugging service workers, web workers, iframes and in-page JS.
* **.NET Productivity**: multiple features for .NET developers including help for the following:
* add an explicit cast (when an implicit cast isn’t possible),
* add a “file banner” across one or more code files,
* refactor/simply conditional expressions,
* convert regular string literals to verbatim strings

The list of features above is just a summary of the official announcement. For details on each of these features, check out the official announcement at:

* VS 2019 v16.6 Preview 2: <https://devblogs.microsoft.com/visualstudio/visual-studio-2019-version-16-6-preview-2/>

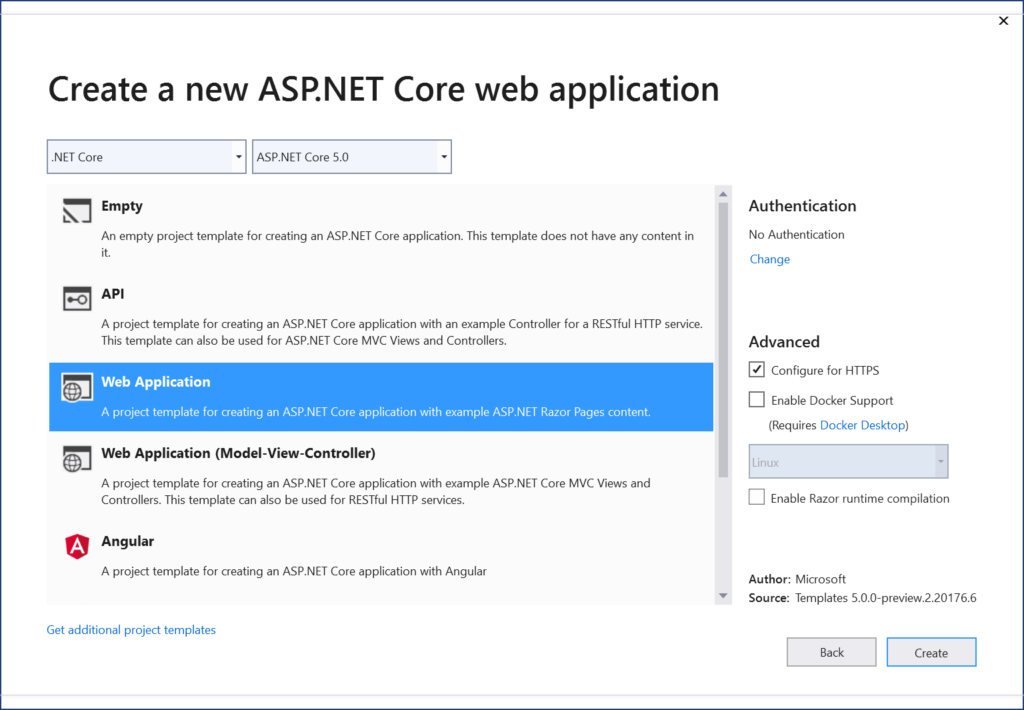
**.NET 5.0 Preview 2**

To get started, start with the latest version of .NET 5.0:

* Download .NET 5.0: <https://dotnet.microsoft.com/download/dotnet-core/5.0>

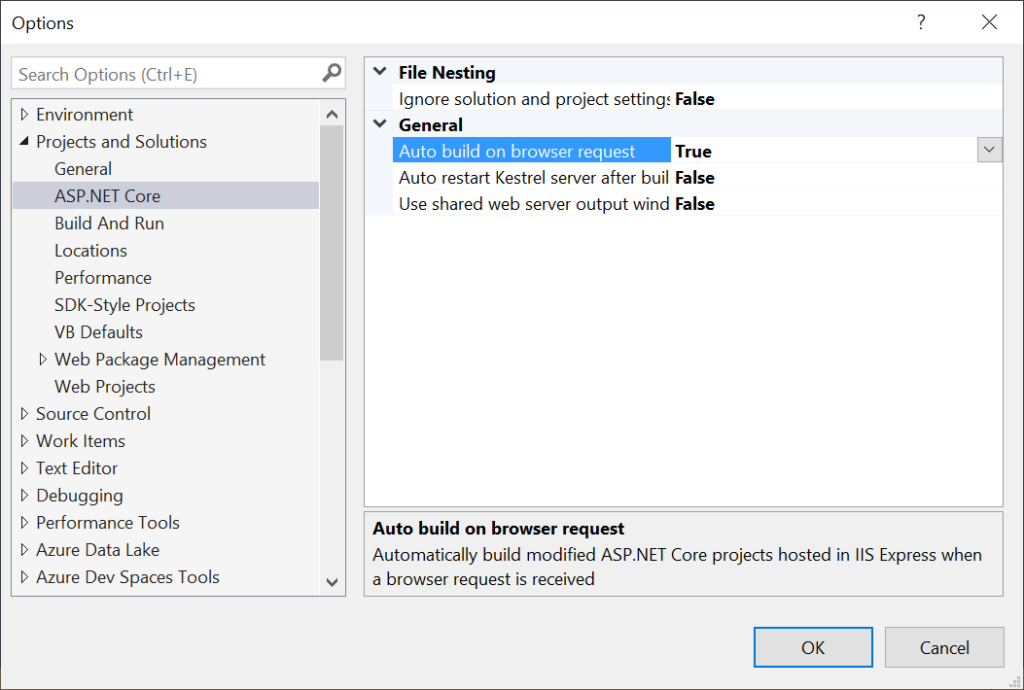
If you’ve already installed Visual Studio 2019, you may have followed the following steps:

1. Download Visual Studio 2019 Preview
2. Click **File** | **New** | **Project**(or create new from splash screen)
3. Create a new ASP .NET Core Web App
4. Select .NET Core and then ASP .NET Core 5.0 as the project type/platform



**NOTE**: For previous versions of .NET Core, you may not have seen an option for ASP .NET Core 3.0 in the list of project templates after installing VS2019. When ASP .NET Core 3.0 was still in preview as of April 2019 (even after the release of Visual Studio 2019), it was not automatically available for selection. In order to create ASP .NET Core 3.0 projects with VS2019, you would have had to install .NET Core 3.0 and enable previews of the .NET Core SDK in Visual Studio’s Options panel.

In Visual Studio 2019 Preview, this additional step isn’t necessary for ASP .NET Core 5.0. In fact, the Options panel includes settings for ASP .NET Core, without any mention of enabling preview versions.

VS2019 Options panel

**C# 9.0 Features**

To see what’s to come in C# 9.0, check out the official list of milestones on GitHub, specifically Milestone 15 for C# 9.0:

* C# Language Milestones: <https://github.com/dotnet/csharplang/milestones>
* C# 9.0 Milestone 15: <https://github.com/dotnet/csharplang/milestone/15>

There are already dozens of feature candidates for the C# 9.0 release. The list includes Records and Pattern-Based “With” Expressions. From the [official proposal](https://github.com/dotnet/csharplang/blob/master/proposals/records.md), “*Records are a new, simplified declaration form for C# class and struct types that combine the benefits of a number of simpler features*“.

This new feature allows you to omit an argument for its corresponding *optional* parameter, when invoking a function member. In this case, the value of the receiver’s member is implicitly passed. This is accomplished in by the use of a “with-expression” and the use of this.*SomeIdentifier* to set a default argument.

Here is an example of what that could look like:

class TwitterUser  
{  
 public readonly int Followers;  
 public readonly int Following;  
 public TwitterUser With(  
 int followers = this.Followers,   
 int following = this.Following)   
 => new TwitterUser(followers, following);  
  
}

How would you use this so-called “caller-receiver default argument?

TwitterUser twitterUser1 = new TwitterUser(1000, 2000)  
TwitterUser twitterUser2 = twitterUser1.With(followers: 0);

The above code creates a new TwitterUser, using the existing object to copy its values but changes the number of followers. For more information about the Records feature, check out the official proposal with detailed notes and code samples:

* C# Records Proposal: <https://github.com/dotnet/csharplang/blob/master/proposals/records.md>

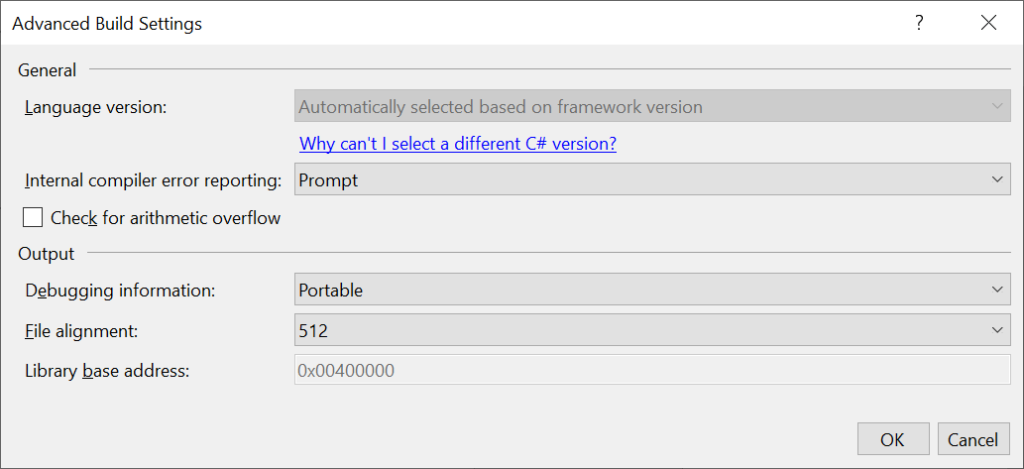
**NOTE**: To ensure that C# 9.0 preview features are available when a preview version becomes available, you can set the **LangVersion** property explicitly for your project. This setting is buried deep inside your Advanced settings within your project’s Build tab.

To update the language setting (after a future preview release):

1. Right-click your project in **Solution Explorer**.
2. Select **Properties** to view your project properties.
3. Click the **Build** tab within your project properties.
4. Click the **Advanced** button on the lower right.
5. Select the appropriate **Language version**, e.g. C# 9.0 (beta)
6. ***Optional***: you may select “**unsupported preview**…” instead

If you try the above steps with VS2019 v16.6.0 Preview 2 (as of this writing), the language version selection is not available at this time. For more information on C# language versioning, check out the official documentation at:

* C# Language Versioning: <https://docs.microsoft.com/en-us/dotnet/csharp/language-reference/configure-language-version>

C# Language Version Selection

The above screenshot shows the aforementioned setting in the Visual Studio UI. If you wish to update your .csproj file directly, you may view/edit the <**LangVersion**> value. A few samples are shown below:

For a .NET Core 3.0 console app set to use C# *preview* versions, the value of <**LangVersion**> is set to the value “preview”. This may be subject to change for C# 9 when it is released.

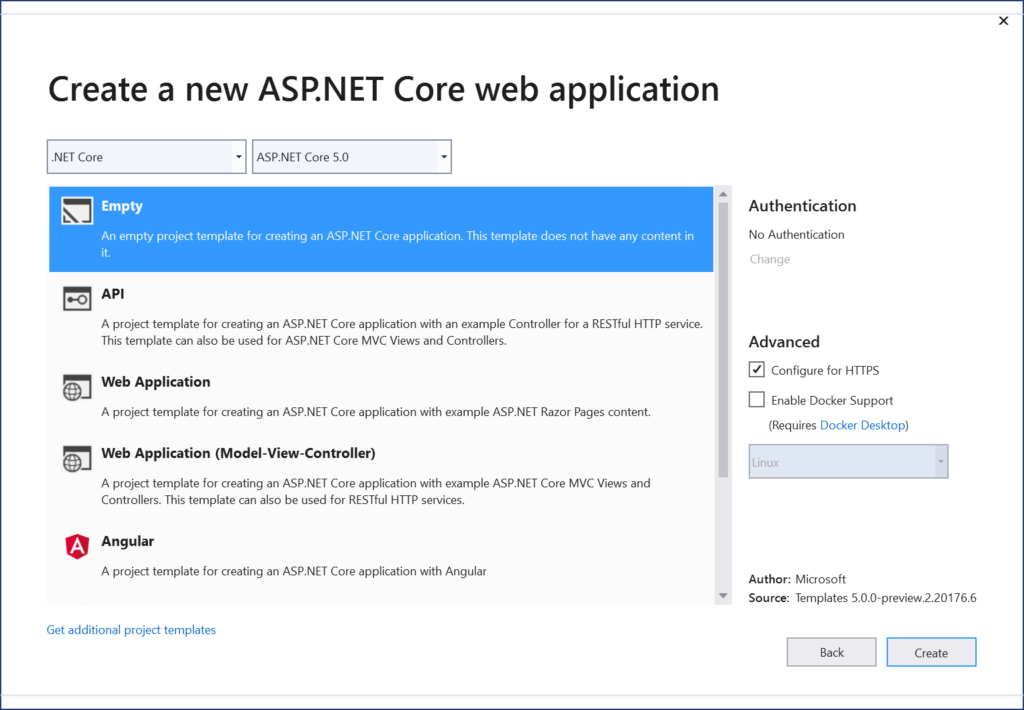
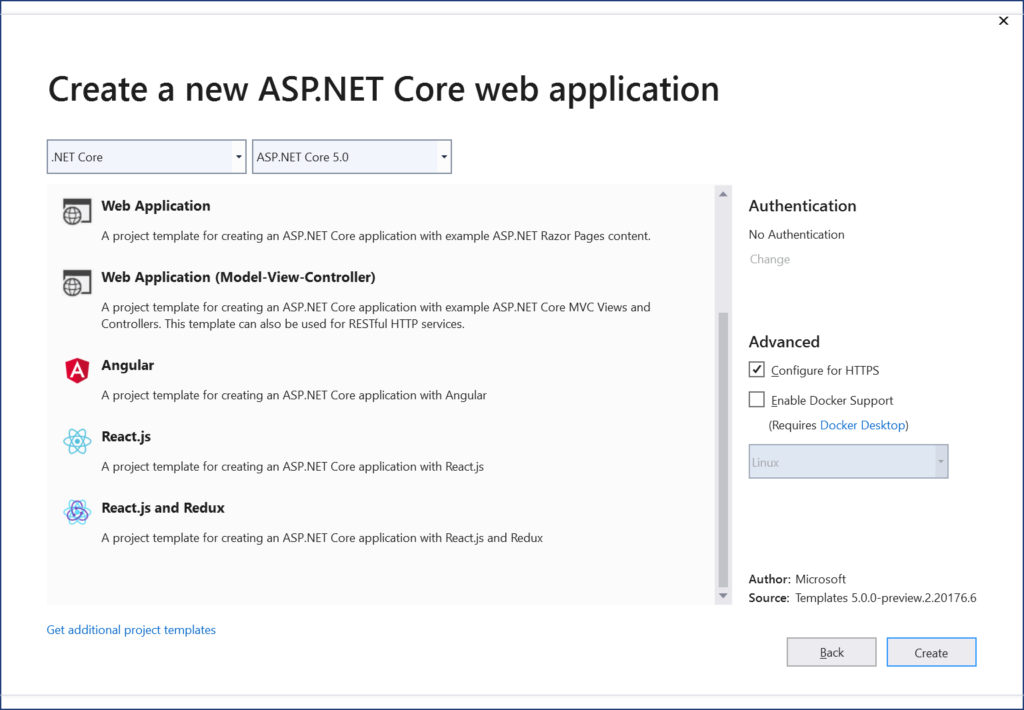
<Project Sdk="Microsoft.NET.Sdk">  
 <PropertyGroup>  
 <OutputType>Exe</OutputType>  
 <TargetFramework>netcoreapp3.0</TargetFramework>  
 <LangVersion>preview</LangVersion>  
 </PropertyGroup>  
</Project>

For a .NET Core 3.0 console app set to use C# 8.0 explicitly before its release, the value of <**LangVersion**> was set to the value “8.0”. Once again, this may be subject to change for C# 9.

<Project Sdk="Microsoft.NET.Sdk">  
 <PropertyGroup>  
 <OutputType>Exe</OutputType>  
 <TargetFramework>netcoreapp3.0</TargetFramework>  
 <LangVersion>8.0</LangVersion>  
 </PropertyGroup>  
</Project>

**ASP .NET Core 5.0 Project Types**

When you create a new .NET 5.0 web project with Visual Studio 2019, you’ll see some familiar project types. You will also see some *new* project types. These are shown in the 2 screenshots below:

List of ASP .NET Core 5.0 projects  List of ASP .NET Core 5.0 projects (continued)

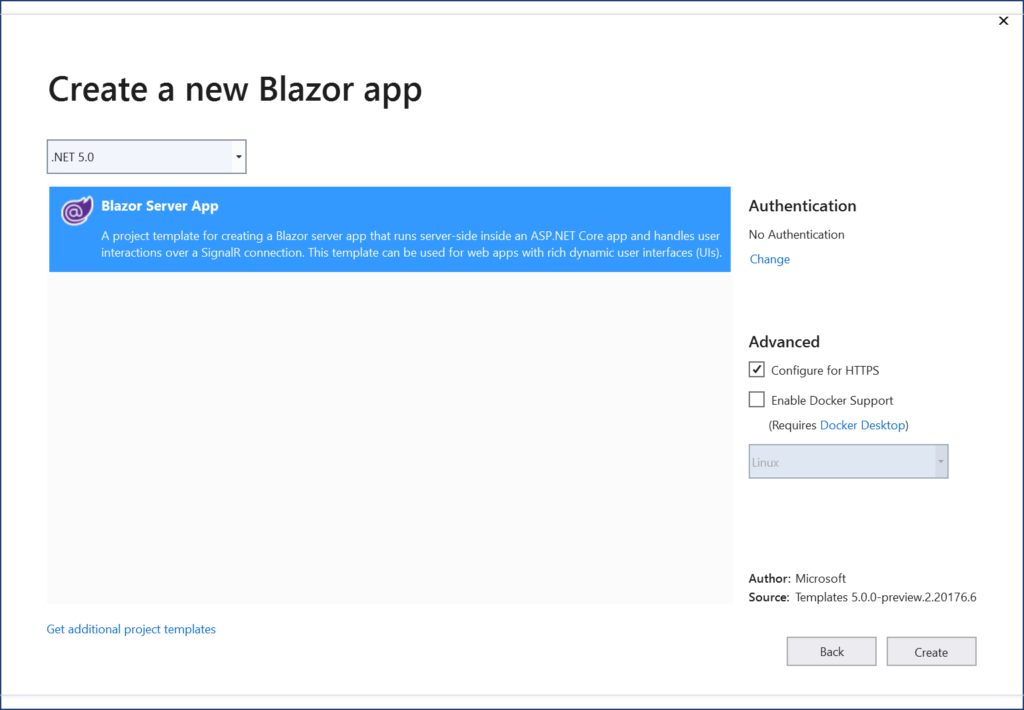
The above project types are described below:

1. **Empty**: familiar empty project that just writes out Hello World to the HTTP Response, *without* the use of MVC or Razor Pages
2. **API**: familiar project type for creating Web APIs and RESTful services. Can be mixed and matched with Razor Pages or MVC components.
3. **Web Application**: familiar project type for creating Web Apps with Razor Pages. Can be mixed and matched with Web API and/or MVC components.
4. **Web Application (MVC)**: familiar project type for creating Web Apps with MVC application structure. Can be mixed and matched with Razor Pages and/or Web API.
5. **Angular, React.js, React.js and Redux**: familiar web projects for web developers who wish to build a JavaScript front-end, typically with a Web API backend.

FYI, the following project templates have moved out of their original location (inside ASP .NET Core web apps):

* **gRPC Service**: allows creation of a new gRPC service to make use of Google’s high-performance Remote Procedure Call (RPC) framework
* **Worker Service**: allows creation of background processes, e.g. Windows services or Linux daemons.
* **Razor Class Library**: allows creation of reusable UI Class Libraries with Razor Pages. See previous post on [Razor Class Libraries](https://wakeupandcode.com/your-first-razor-ui-library-with-asp-net-core/).

Blazor’s server-side project template can also be found at the root-level of the template list, under Blazor App.

 List of ASP .NET Core 5.0 projects (Blazor Server)

Well, what about *client-side Blazor*? You may have noticed that server-side Blazor (aka Razor Components are mentioned, but there is no sign of client-side Blazor. As of March 2020, client-side Blazor running in the browser with WebAssembly is still in preview ([v3.2.0 Preview 3](https://devblogs.microsoft.com/aspnet/blazor-webassembly-3-2-0-preview-3-release-now-available/)).

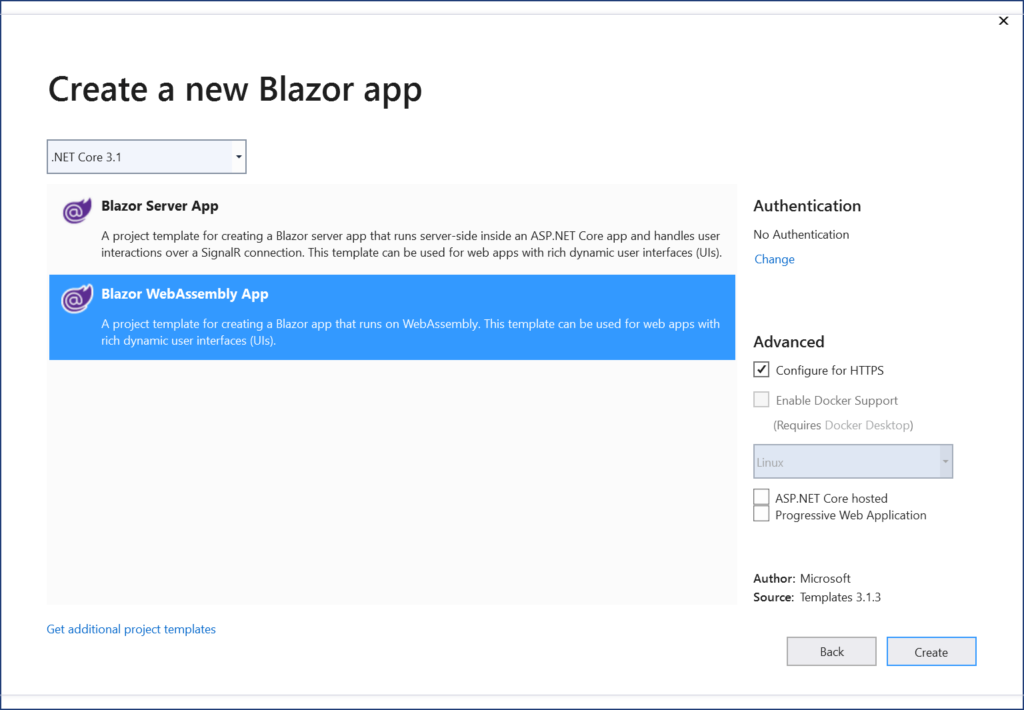
If you were to install the latest .NET Core 3.1 SDK, you would also have to download the latest Blazor WebAssembly separately.

* Install Blazor WebAssembly template: <https://www.nuget.org/packages/Microsoft.AspNetCore.Components.WebAssembly.Templates/>

In order to install the client-side template from a command line, run the following command:

dotnet new -i Microsoft.AspNetCore.Components.WebAssembly.Templates::3.2.0-preview2.20160.5

**NOTE**: Installing VS2019 v16.6 Preview 2 *automatically* installs an updated version of the .NET Core 3.1 SDK. This version includes the latest Blazor WebAssembly template. This means that a manual installation of client-side Blazor is not required when using this preview version of Visual Studio. However, you would have to select .NET Core 3.1 after selecting Blazor, if you wish to select the latest client-side Blazor WebAssembly template in VS2019 v16.6 Preview 2.

Blazor templates in VS2019 Preview

On a related note, check out my previous post on *server-side* Blazor:

* Blazor Full-Stack Web Dev in ASP .NET Core 3.1:  <https://wakeupandcode.com/blazor-full-stack-web-dev-in-asp-net-core-3-1/>

**Migrating from 3.1 to 5.0**

With all this goodness on the roadmap, you may be wondering how you can start migrating your ASP .NET Core 3.1 projects to .NET 5.0. Fear not, the official guide (a work in progress) is now available:

* Migrate from ASP .NET Core 3.1 to 5.0: <https://docs.microsoft.com/en-us/aspnet/core/migration/31-to-50>

This documented migration process involves the following steps:

1. Update .NET Core SDK version in global.json (from “3.1.200” to “5.0.100-preview.2.20176.6”)
2. Update the target framework (from netcoreapp3.1 to netcoreapp5.0)
3. Update package references (e.g. from “3.1.2” to “5.0.0-preview.2.20167.3”)
4. Update Docker images to include a base image that includes ASP .NET Core 5.0 (e.g. by using the docker pull command followed by mcr.microsoft.com/dotnet/core/aspnet:5.0)
5. Review breaking changes: refer to <https://docs.microsoft.com/en-us/dotnet/core/compatibility/3.1-5.0>

To deploy a .NET 5 project via Azure DevOps, try the following YAML

steps:  
- task: UseDotNet@2  
 inputs:  
 packageType: sdk  
 version: 5.0.100-preview.2.20176.6  
 installationPath: $(Agent.ToolsDirectory)/dotnet

The above YAML snippet is a suggestion from .NET developer [Neville Nazerane](https://www.facebook.com/i.am.neville) via the unofficial ASP .NET Core Facebook group:

* YAML snippet source: <https://www.facebook.com/groups/about.asp.net.core/permalink/1538257476352234/?comment_id=1539064492938199>

**NOTE**: ASP .NET Core in .NET 5.0 Preview 2 only includes minor bug fixes (but not any major new features).

**References**

* [May 6, 2019] Introducing .NET 5: <https://devblogs.microsoft.com/dotnet/introducing-net-5/>
* [March 16, 2020] Announcing .NET 5.0 Preview 1: <https://devblogs.microsoft.com/dotnet/announcing-net-5-0-preview-1/>
* [April 2, 2020] Announcing .NET 5.0 Preview 2: <https://devblogs.microsoft.com/dotnet/announcing-net-5-0-preview-2/>
* ASP .NET Core Updates in .NET 5 Preview 2: <https://devblogs.microsoft.com/aspnet/asp-net-core-updates-in-net-5-preview-2/>
* VS 2019 v16.6 Preview 2: <https://devblogs.microsoft.com/visualstudio/visual-studio-2019-version-16-6-preview-2/>
* EF Core 5.0 Preview 2: <https://devblogs.microsoft.com/dotnet/announcing-entity-framework-core-5-0-preview-2/>
* Migrate from ASP .NET Core 3.1 to 5.0: <https://docs.microsoft.com/en-us/aspnet/core/migration/31-to-50>
* C# 9 Language Features: <https://github.com/dotnet/csharplang/milestone/15>
* Blazor WebAssembly 3.2.0 Preview 3: <https://devblogs.microsoft.com/aspnet/blazor-webassembly-3-2-0-preview-3-release-now-available/>
* Blazor WebAssembly Template on NuGet: <https://www.nuget.org/packages/Microsoft.AspNetCore.Components.WebAssembly.Templates/>