**Summarizing Build 2019 + SignalR Service for ASP .NET (Core) Developers**

By Shahed C on May 14, 2019

[1 Reply](https://wakeupandcode.com/summarizing-build-2019-signalr-service/#comments)

This is the **nineteenth**of a [series of posts](https://wakeupandcode.com/aspnetcore/#aspnetcore2019) on ASP .NET Core in 2019. In this series, we’ll cover 26 topics over a span of 26 weeks from January through June 2019, titled **A-Z of ASP .NET Core!**

** A – Z of ASP .NET Core!**

**In this Article:**

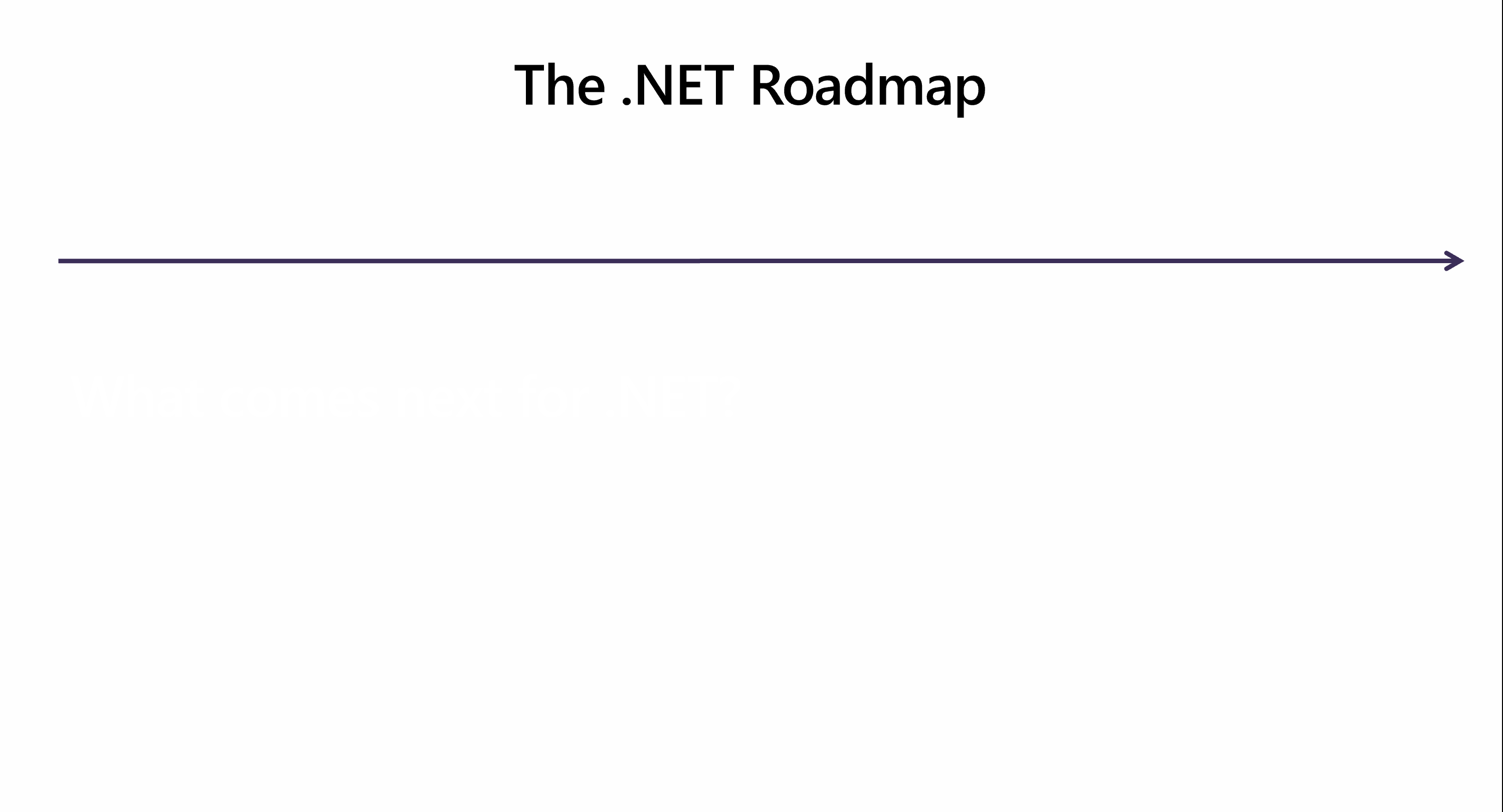
* S is for Summarizing Build 2019 (and SignalR Service!)
* Build 2019 for .NET Developers
* What’s New in .NET Core 3.0 (Preview 5)
* What’s New in ASP.NET Core 3.0 (Preview 5)
* What’s Next for .NET Core (.NET Core vNext = .NET 5!)
* EF 6.3 for .NET Core, SqlClient & Diagnostics
* SignalR Service (Sneak Peek)
* References for Build 2019
* References for SignalR Service

**S is for Summarizing Build 2019 (and SignalR Service!)**

For the letter S, I was originally thinking of an article dedicated to SignalR Service. However, Microsoft’s annual Build Conference just happened at the time of this writing. So this week’s post will focus on Summarizing Build 2019 for ASP .NET (Core) Developers, followed by a sneak peek of SignalR Service at the end.

The biggest news for .NET Developers is that .NET Core is the future of .NET, going forward. Furthermore, .NET Core vNext will be named .NET 5, a unified platform for all .NET developers (.NET Framework, Xamarin/Mono and .NET Core).

The GIF below was generated from the [.NET Overview session at Build 2019](https://youtu.be/ZlO1utbB2GQ) to illustrate this future:

[](https://wakeupandcode.com/wp-content/uploads/2019/05/DotNet5-Next.gif)

.NET Roadmap, from 2014 through 2016, then 2019 and Beyond

**Build 2019 for .NET Developers**

The quickest to way to catch up on Build 2019 content is to watch all the relevant videos. But how do you know which ones to watch? Well, if you’re a .NET developer, I’ve put together a handy video playlist specifically for .NET Developers (including ASP .NET Core Developers).

* Build 2019 for .NET Devs: <https://aka.ms/DotNet-Build2019>

Your ***key takeaways*** from new announcements should include:

* **.NET Core is the future of .NET**: If you’ve already started working with .NET Core, that’s great! If you’re starting a new project, you should consider .NET Core.
* **.NET Framework will continue to be supported**: If you have any existing applications on .NET Framework (Windows-only), you can keep those on .NET Framework.
* **.NET Releases will become more predictable**: Starting with .NET 5.0, there will be 1 major release every year,  after which each even-numbered release (6.0, 8.0, etc) will come with LTS (Long-Term Support).

In 2019, the expected **schedule for .NET Core 3.x** is as follows:

* July 2019: .NET Core 3.0 RC (Release Candidate)
* September 2019: .NET Core 3.0 (includes ASP .NET Core 3.0)
* November 2019: .NET Core 3.1 (LTS)

In 2020 and beyond, the expected **schedule for .NET Core 5+** is shown below:

* Early to mid 2020: .NET 5.0 Preview 1
* November 2020: .NET 5.0
* November 2021: .NET 6.0 (LTS)
* November 2022: .NET 7.0
* November 2023: .NET 8.0 (LTS)

Minor releases (e.g. 5.1, etc) will be considered only if necessary. According to the [official announcement](https://devblogs.microsoft.com/dotnet/introducing-net-5/), the first preview of .NET 5.0 should be available within the first half of 2020.

**NOTE**: The upcoming .NET 5.0 should not be confused with the so-called “[ASP .NET 5](https://www.hanselman.com/blog/ASPNET5IsDeadIntroducingASPNETCore10AndNETCore10.aspx)” which was the pre-release name for ASP .NET Core 1.0 before the product was first released in 2016. Going forward, the name of the unified framework is simply .NET 5, without the need for a trailing “Core” in the name.

**What’s New in .NET Core 3.0 (Preview 5)**

As of May 2019, .NET Core 3.0 is in Preview 5, is expected to be in RC in July 2019, to be followed by a full release in September 2019. This includes ASP .NET Core 3.0 for web development (and more!). For my first look at .NET Core 3.0, you may browse through this earlier post in this series:

* .NET Core 3.0, VS2019 and C# 8.0 for ASP .NET Core developers: <https://wakeupandcode.com/net-core-3-vs2019-and-csharp-8/>

The primary themes of .NET Core 3.0 are:

1. **Windows desktop apps**: while this is usually not a concern for ASP .NET Core web application developers, it’s good to know that Windows developers can start using .NET Core right away.
2. **Full-stack web dev**: Blazor is no longer experimental and its latest preview allows developers to use C# for full-stack web development. More info at: <https://blazor.net>
3. **AI & ML**: Not just buzzwords, Artificial Intelligence and Machine Learning are everywhere. ML.NET 1.0 is now available for C# developers to join this exciting new area of software development. More info at: [dot.net/ml](https://dot.net/ml)
4. **Big Data**: .NET for Apache Spark is now in Preview, available on Azure Databricks and Azure HDInsight. More info at: [dot.net/spark](https://dot.net/spark)

For more information on Blazor, you may browse through  this earlier post in this series:

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

A lot has changed with Blazor in recent months, so the above post will be updated after Core 3.0 is released. In the meantime, check out the official Blazor session from Build 2019.

* **Blazor** @ Build 2019: <https://youtu.be/y7LAbdoNBJA>

**What’s New in ASP.NET Core 3.0 (Preview 5)**

What about ASP .NET Core 3.0 in Preview 5? In the [Preview 5 announcement](https://devblogs.microsoft.com/aspnet/asp-net-core-updates-in-net-core-3-0-preview-5/), you can see a handful of updates for this specific release. This includes the following:

* **Easy Upgrade**: to upgrade from an earlier preview, update the package reference in your .csproj project file to the latest version (3.0.0-preview5-19227-01)
* **JSON Seralization**: now includes support for reading/writing JSON using [System.Text.Json](https://docs.microsoft.com/en-us/dotnet/api/system.text.json?view=netcore-3.0). This is part of the [built-in JSON support](https://docs.microsoft.com/en-us/dotnet/core/whats-new/dotnet-core-3-0#fast-built-in-json-support) mentioned in the official writeup for .NET Core 3.0 Preview 5.
* **Integration with SignalR**: Starting with this release, SignalR clients and servers will now use the aforementioned System.Text.Json as the default Hub protocol. You can read more about this in the [SignalR section](https://docs.microsoft.com/en-us/aspnet/core/migration/22-to-30?view=aspnetcore-2.2&amp;tabs=visual-studio#update-signalr-code) of the Migration guide for 2.x to 3.0.
* **Continued NewtonsoftJson support**: In case you need to switch back to NewtonSoft.Json (previously the default option for the SignalR Hub), the instructions are provided in the aforementioned Migration guide and the announcement. Note that NewtonSoft.Json needs to be installed as a [NuGet package](https://www.nuget.org/packages/Microsoft.AspNetCore.SignalR.Protocols.NewtonsoftJson/).

There has been a lot of development in ASP .NET Core 3.0 in previous Preview releases, so you can refer to my ***earlier posts in the series***for more info:

* [Dec 2018] Exploring .NET Core 3.0 and the Future of C#: <https://wakeupandcode.com/exploring-net-core-3-0-and-the-future-of-csharp/>
* [April 2019] .NET Core 3.0, VS2019 and C# 8.0 for ASP .NET Core developers: <https://wakeupandcode.com/net-core-3-vs2019-and-csharp-8/>

Here are ***links to all preview notes*** if you need a refresher on what was new in each Preview:

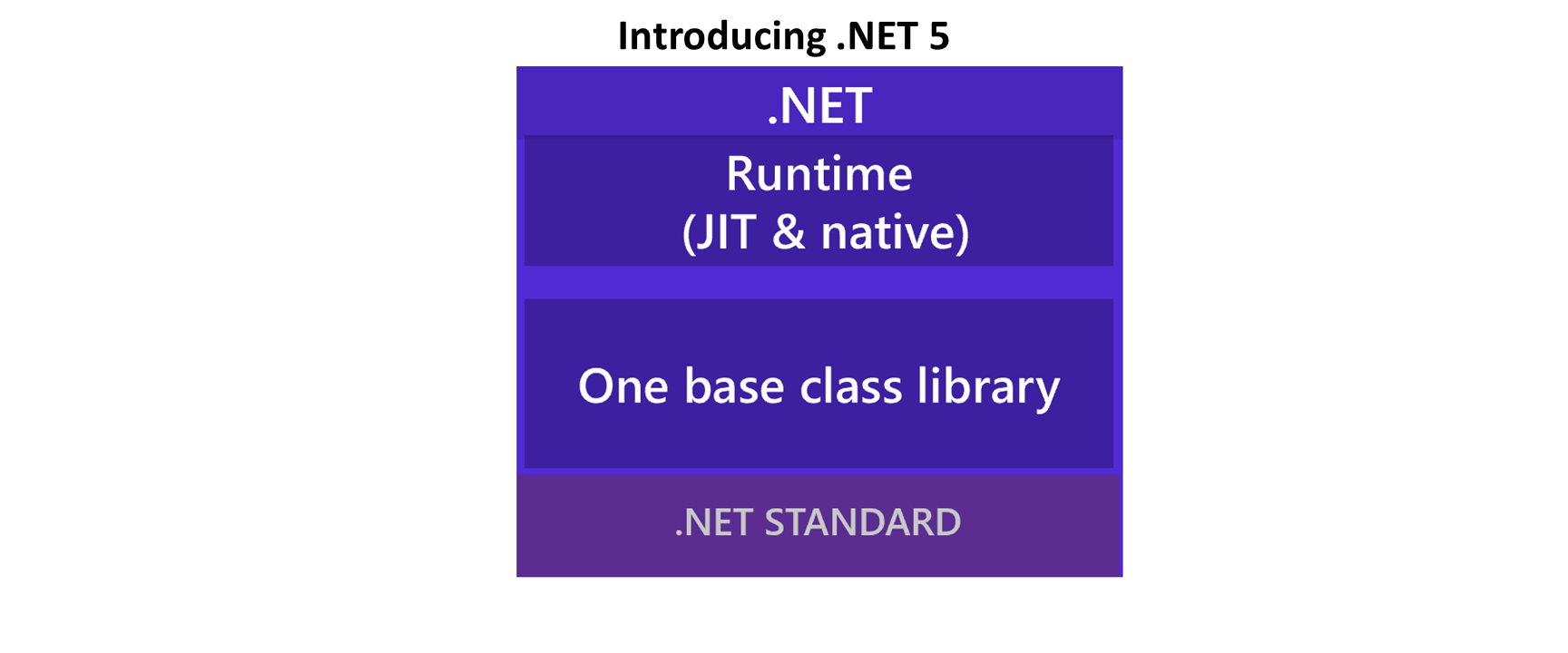
* [Jan 29, 2019] Preview 2: <https://devblogs.microsoft.com/aspnet/aspnet-core-3-preview-2/>
* [March 6, 2019] Preview 3: <https://devblogs.microsoft.com/aspnet/asp-net-core-updates-in-net-core-3-0-preview-3/>
* [April 18, 2019] Preview 4: <https://devblogs.microsoft.com/aspnet/asp-net-core-updates-in-net-core-3-0-preview-4/>
* [May 6, 2019] Preview 5: <https://devblogs.microsoft.com/aspnet/asp-net-core-updates-in-net-core-3-0-preview-5/>

**NOTE**: Changes from each preview to the next is usually cumulative. *However,* please note that **Blazor** went [from experimental to preview in April 2019,](https://devblogs.microsoft.com/aspnet/blazor-now-in-official-preview/) and is *now called Blazor on both the client and server*. Previously, server-side Blazor was temporarily renamed to Razor Components, but then it was changed back to server-side Blazor.

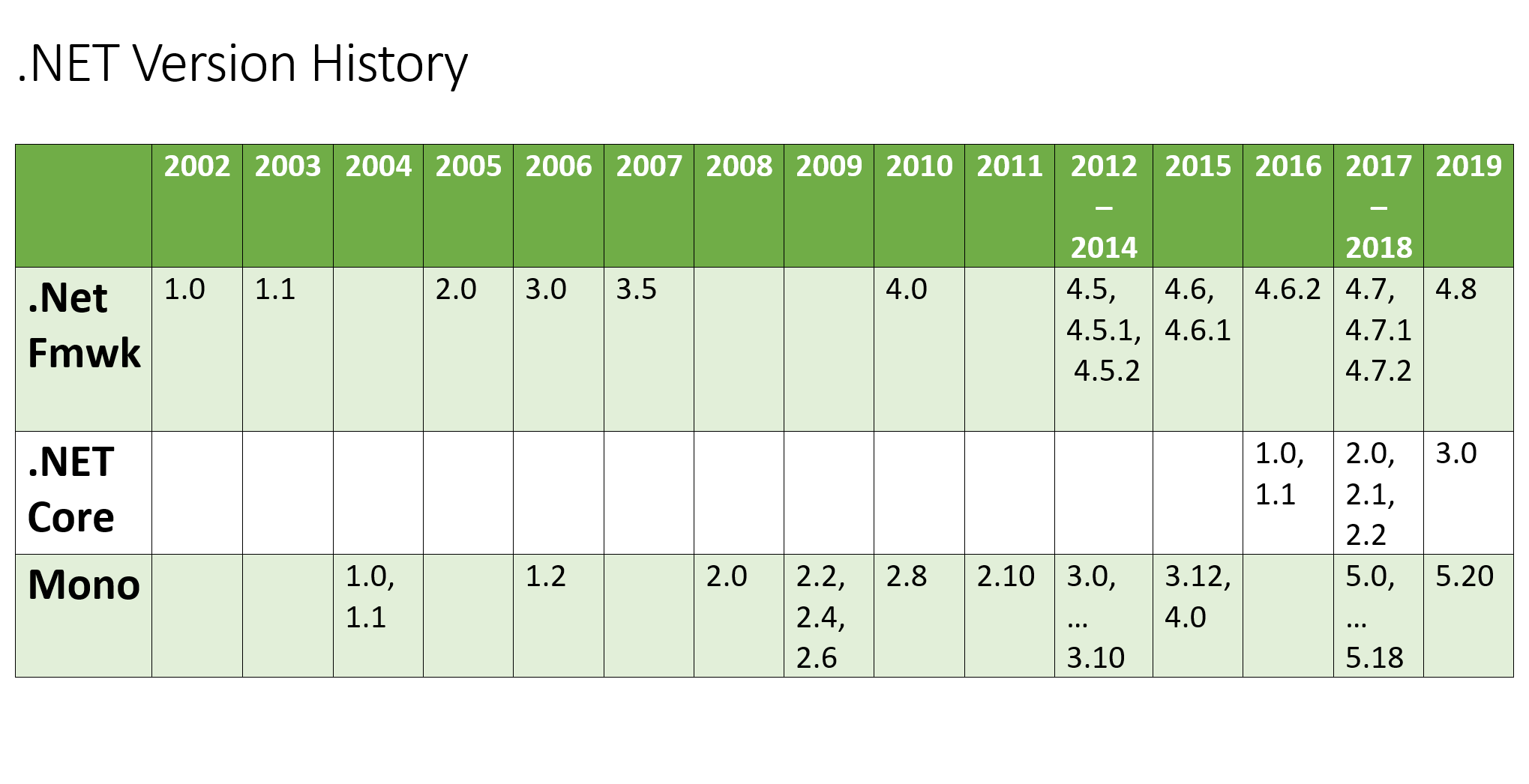
**What’s Next for .NET Core (.NET Core vNext = .NET 5!)**

So, what’s next for .NET Core? First of all, the next major version won’t be called .NET Core, as we now know. With the upcoming release of .NET 5, you can now rest assured that all your investment into .NET Core will carry over into the future unified release.

Imagine taking the cross-platform .NET Core and bringing in the best of Mono for a single BCL (Base Class Library implementation). You may recall that .NET Standard was introduced when there were multiple versions of .NET (Framework, Mono/Xamarin and Core). Going forward, .NET Standard will continue to exist and .NET 5 will adhere to .NET Standard.

[](https://wakeupandcode.com/wp-content/uploads/2019/05/DotNet5-Layers-Heading.png)

Compare the unified diagram with the various versions of .NET Framework, .NET Core and Mono over the years:

[](https://wakeupandcode.com/wp-content/uploads/2019/05/DotNet-VersionHistory.png)

Note that not everything in the world of .NET today will make it into .NET 5. Not to worry though, as there are various recommended alternatives for all .NET developers. Take the following technologies, for example:

* **Web Forms**: As you may have noticed, ASP .NET Web Forms have not been ported to ASP .NET Core. Instead of Web Forms, developers may consider Blazor as their choice of web application development.
* **WCF**: Although Web API has been included in ASP .NET Core, there is no option for WCF. Going forward, you may use gRPC as an alternative.

Migration Guides for the above scenarios will be provided at a later date.

**EF 6.3 for .NET Core, SqlClient & Diagnostics**

In addition to ASP .NET Core itself, there are other tools and technologies that may be useful for ASP .NET Core developers. That may include (but is not limited to) the following):

* **Entity Framework 6.3**: In addition to the EF Core running on .NET Core, EF 6.x was known to run on the Windows-only .NET Framework 4.x but not on .NET Core. Going forward, [EF 6.3 will run on .NET Core 3.0](https://devblogs.microsoft.com/dotnet/announcing-entity-framework-6-3-preview-with-net-core-support/) across platforms.
* **SqlClient**: Instead of replacing the existing System.Data.SqlClient package directly, the new [Microsoft.Data.SqlClient](https://www.nuget.org/packages/Microsoft.Data.SqlClient/) (available on NuGet) will support both .NET Core and .NET Framework.
* **.NET Core Diagnostic Tools**: Making use of [.NET Core Global Tools](https://docs.microsoft.com/en-us/dotnet/core/tools/global-tools), a new suite of tools will help developers with diagnostics and troubleshooting ofperf issues.

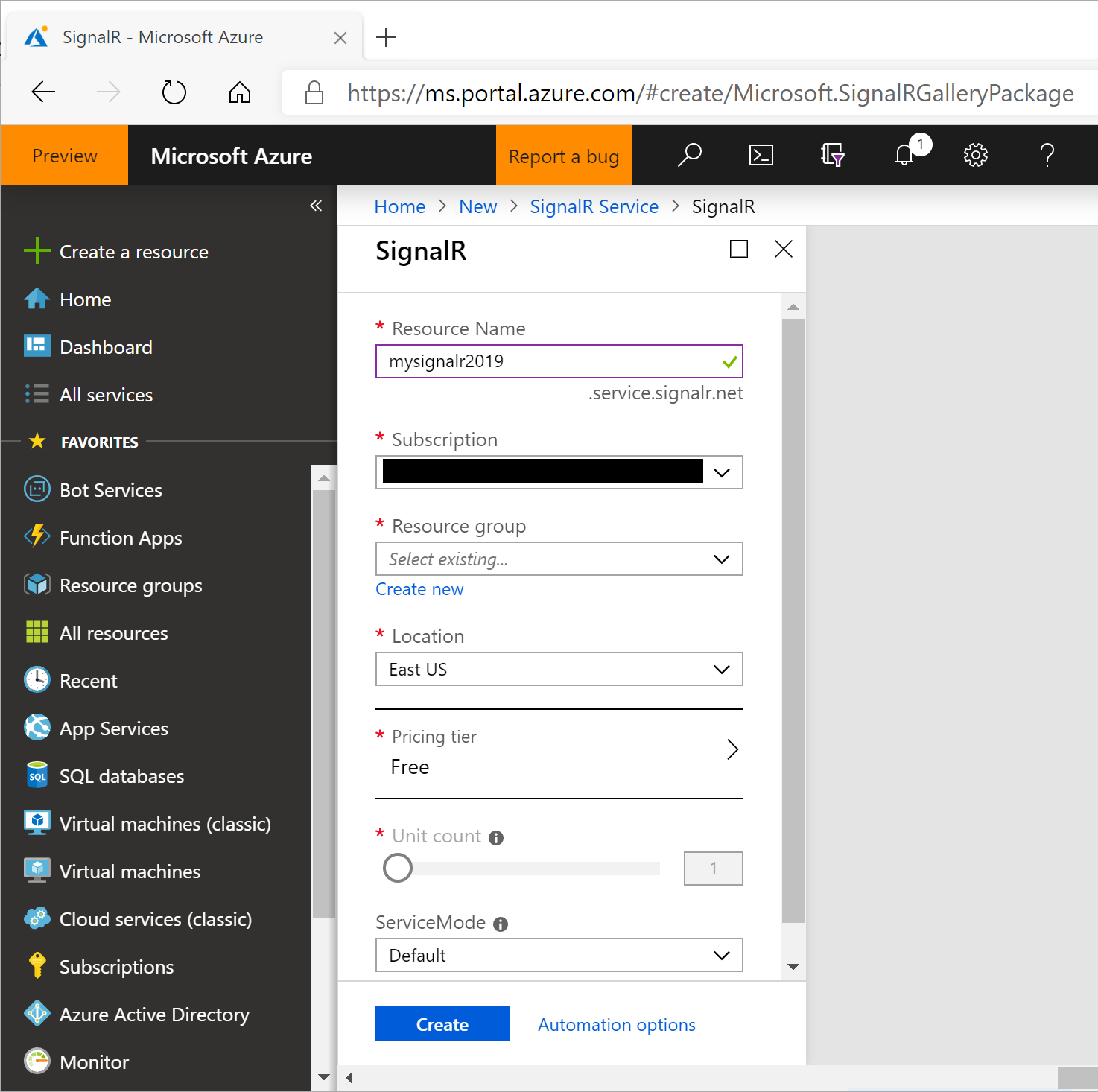
From the [tools](https://github.com/dotnet/diagnostics)‘ GitHub page, the following tools are currently available, with the following descriptions:

* **dotnet-dump**: *“Dump collection and analysis utility.”*
* **dotnet-trace**: *“Enable the collection of events for a running .NET Core Application to a local trace file.”*
* **dotnet-counters**: *“Monitor performance counters of a .NET Core application in real time.”*

**SignalR Service (Sneak Peek)**

Finally, let’s take a quick peek at the all-new SignalR Service.

* **What**: SignalR Service is a cloud-based service available in Azure, to help developers add real-time features in web applications and more.
* **When:** SignalR Service has been available since Build 2018
* **Where:**Available in the [Azure Portal](https://portal.azure.com)
* **How:**Create a new “SignalR Service” Resource
* Direct link: <https://ms.portal.azure.com/#create/Microsoft.SignalRGalleryPackage>

[](https://wakeupandcode.com/wp-content/uploads/2019/05/SignalR-Service-Portal.png)

* **Who** can use this: Web developers who want to build real-time features can get started with a variety of [official Quickstart guides](https://docs.microsoft.com/en-us/azure/azure-signalr/): ASP .NET Core, JavaScript, C#, Java and REST API

If you’re already familiar with using SignalR, switching to using Azure SignalR Service is as easy as 1-2-3.

1. Append a call to .**AddAzureSignalR**() to **AddSignalR**() in the **ConfigureServices**() method of your Startup class.
2. Replace the call to **UseSignalR()** with a call to **UseAzureSignalR**() in your **Configure**() method
3. Ensure that your environment’s connection string is set correctly for the key “**Azure:SignalR:ConnectionString**“.

In the **ConfigureServices**() method, this is what your code should look like:

public void ConfigureServices(IServiceCollection services)  
{  
 // ...  
 services.AddMvc();  
 services.AddSignalR().AddAzureSignalR();  
}

In the **Configure**() method, this is what your code should look like:

app.UseAzureSignalR(routes =>  
{  
 routes.MapHub<HubClassName>("/HubRouteName");  
});

Your connection string (in your environment or User Secrets) may look like the following:

"Azure:SignalR:ConnectionString": "Endpoint=<yourendpoint>;AccessKey=<yourkey>;"

For a detailed tutorial for ASP .NET Core developers, try this official guide:

* Quickstart to learn how to use Azure SignalR Service: <https://docs.microsoft.com/en-us/azure/azure-signalr/signalr-quickstart-dotnet-core>

After the A-Z weekly series is complete, stay tuned for monthly blog posts about cool things .NET developers can do in Azure. This will include a more in-depth look at SignalR Service in a future writeup, including guidance for both web and mobile developers.

**References for Build 2019 Announcements**

* Build 2019: <https://mybuild.techcommunity.microsoft.com/sessions>
* YouTube Playlist: <https://aka.ms/DotNet-Build2019>
* Announcing .NET Core 3.0 Preview 5: <https://devblogs.microsoft.com/dotnet/announcing-net-core-3-0-preview-5/>
* .NET Core is the Future of .NET: <https://devblogs.microsoft.com/dotnet/net-core-is-the-future-of-net/>
* Introducing .NET 5: <https://devblogs.microsoft.com/dotnet/introducing-net-5/>
* ASP.NET Core updates in .NET Core 3.0 Preview 5: <https://devblogs.microsoft.com/aspnet/asp-net-core-updates-in-net-core-3-0-preview-5/>
* Announcing Entity Framework 6.3 Preview with .NET Core Support: <https://devblogs.microsoft.com/dotnet/announcing-entity-framework-6-3-preview-with-net-core-support/>
* Introducing diagnostics improvements in .NET Core 3.0: <https://devblogs.microsoft.com/dotnet/introducing-diagnostics-improvements-in-net-core-3-0/>
* dotnet/diagnostics: <https://github.com/dotnet/diagnostics>
* Introducing the new Microsoft.Data.SqlClient: <https://devblogs.microsoft.com/dotnet/introducing-the-new-microsoftdatasqlclient/>

**References for SignalR Service**

* Azure SignalR Service now supports ASP.NET!: <https://devblogs.microsoft.com/aspnet/azure-signalr-service-now-supports-asp-net/>
* Quickstart to learn how to use Azure SignalR Service: <https://docs.microsoft.com/en-us/azure/azure-signalr/signalr-quickstart-dotnet-core>
* Tutorial: Azure SignalR Service authentication with Azure Functions: <https://docs.microsoft.com/en-us/azure/azure-signalr/signalr-tutorial-authenticate-azure-functions>
* Guide for authenticating Azure SignalR Service clients: <https://docs.microsoft.com/en-us/azure/azure-signalr/signalr-concept-authenticate-oauth>
* Azure SignalR Service serverless quickstart: <https://docs.microsoft.com/en-us/azure/azure-signalr/signalr-quickstart-azure-functions-csharp>