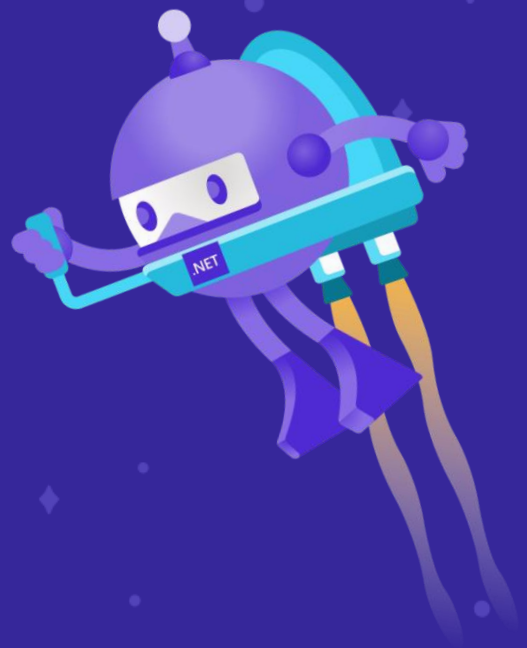
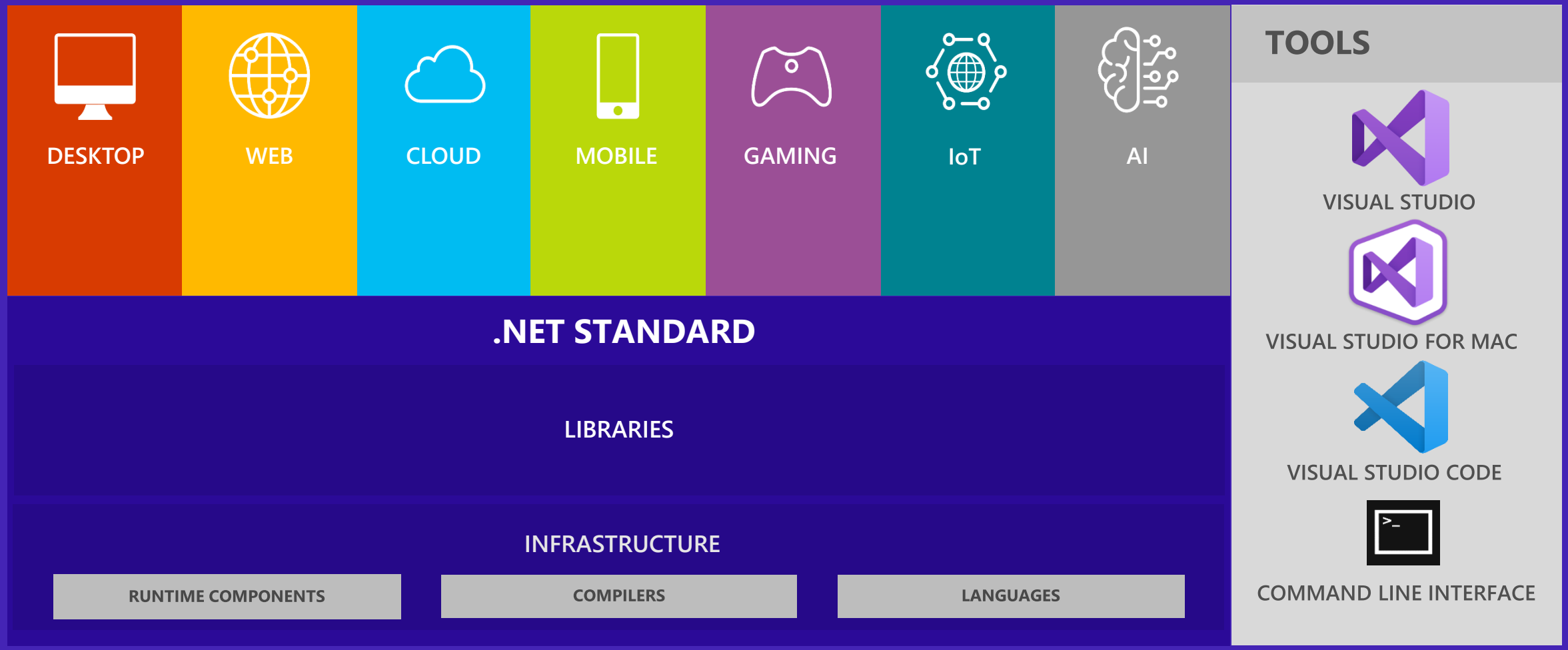
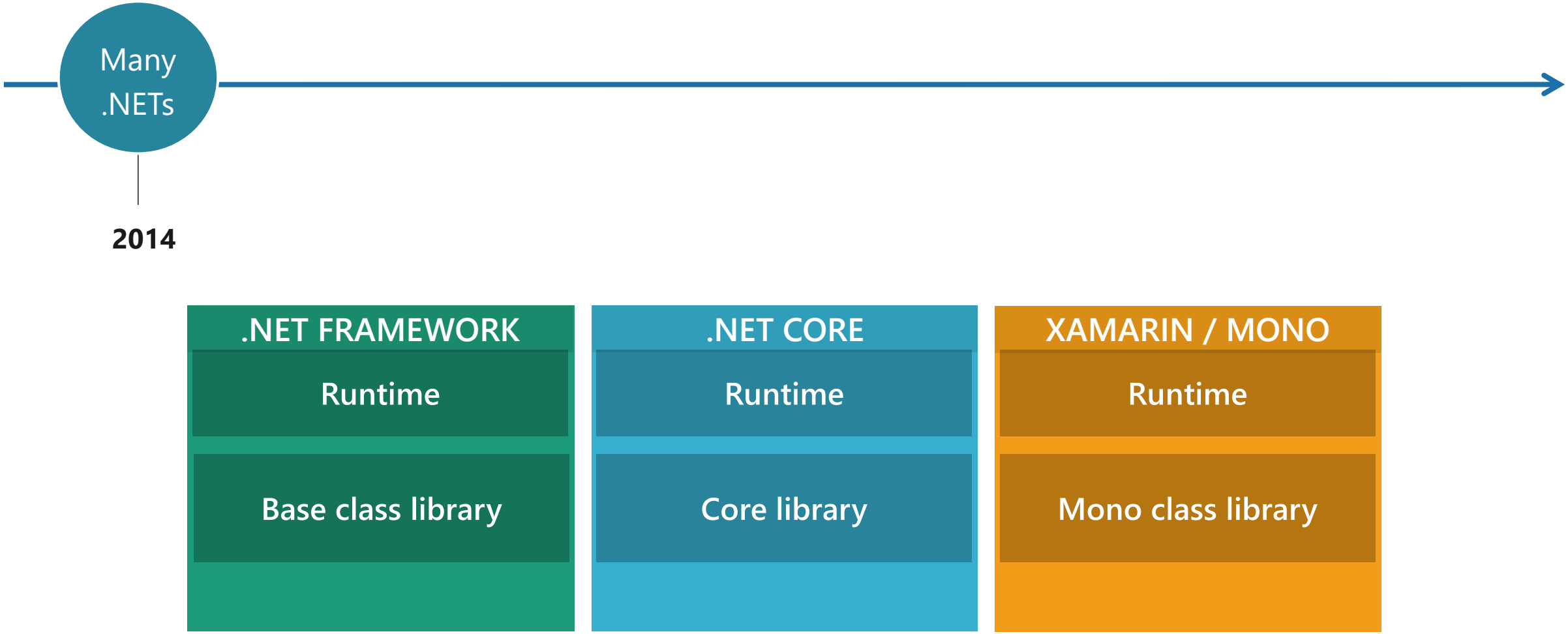
Day 6 : ASP.NET Core



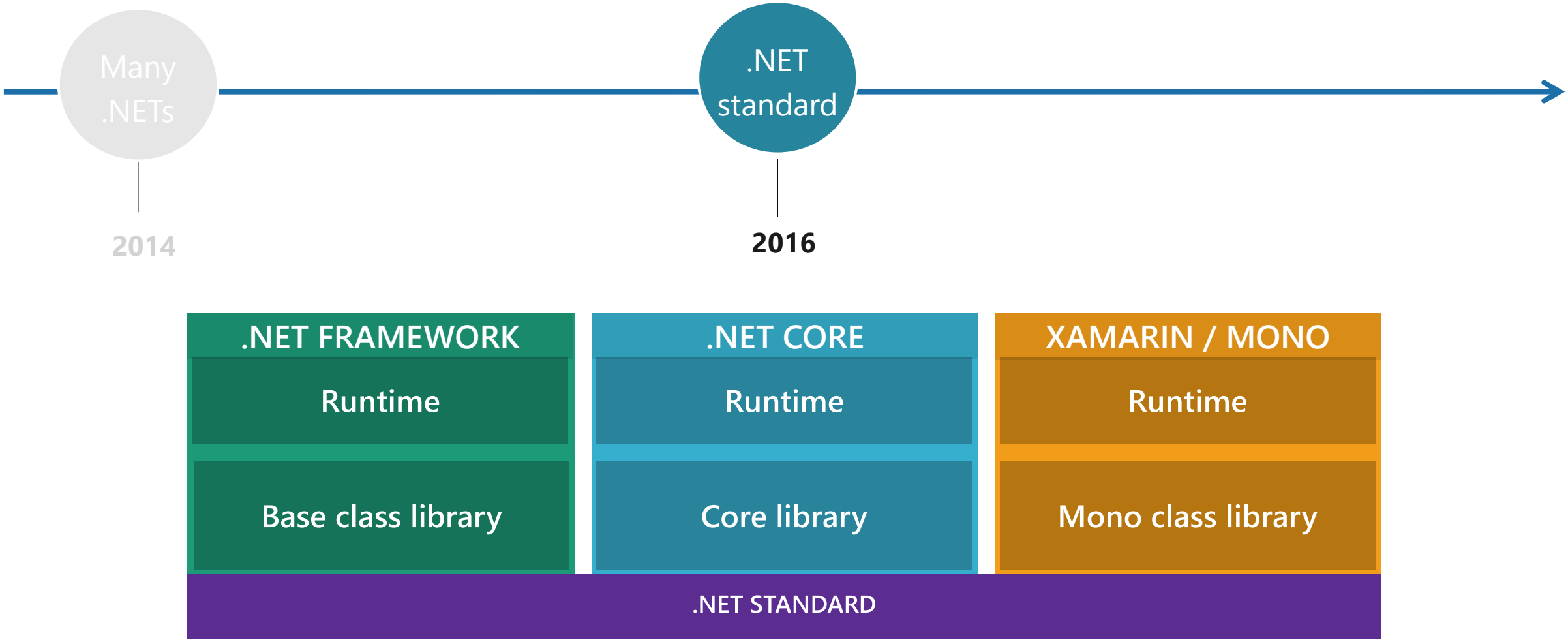
.NET is a software development platform



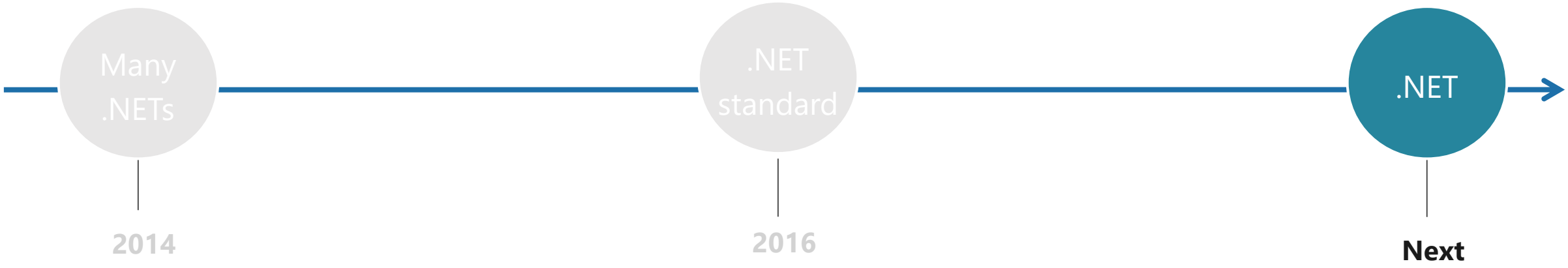
The .NET Roadmap

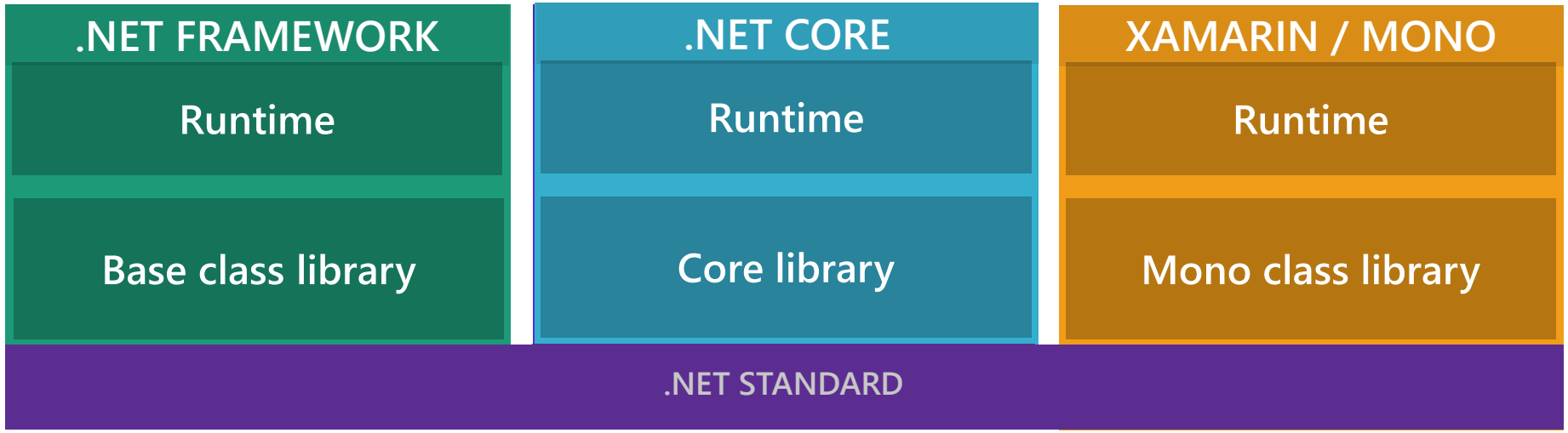


The .NET Roadmap

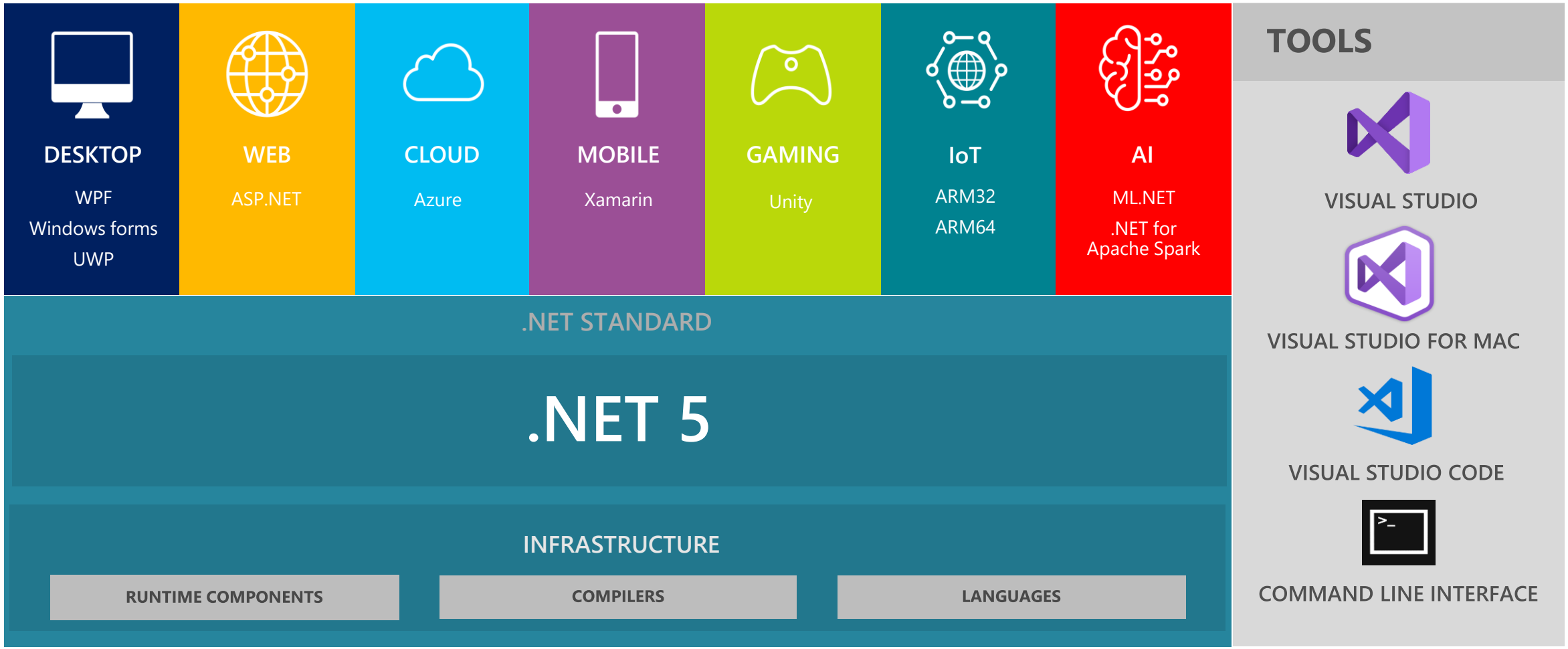


The .NET Roadmap





.NET—A unified platform



RELEASED

.NET 5 General Availability

- Single file applications & smaller container images
- Web and cloud investments
- Windows desktop development enhancements
- Windows ARM64 support
- Continued performance improvements
- New C# 9.0, F# 5.0 language features

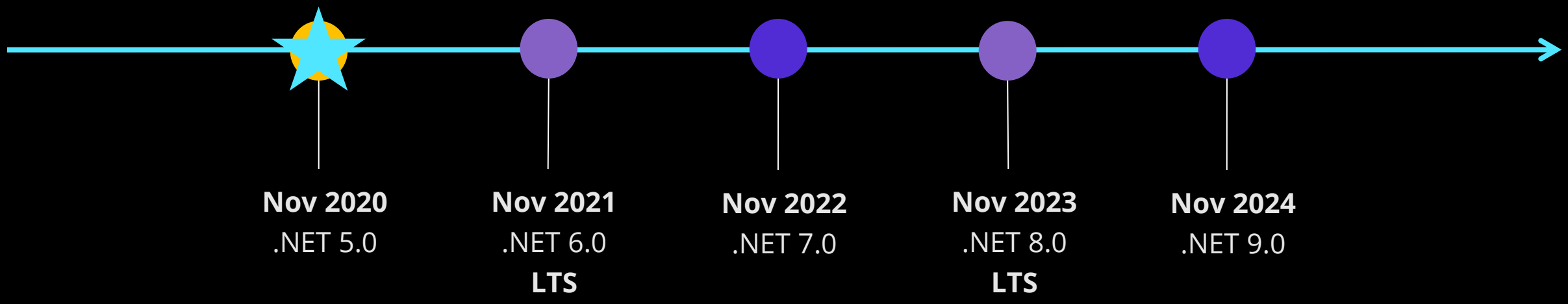
dot.net/get-dotnet5

One .NET vision – .NET 5 to 6 "wave"

.NET Framework	Single SDK, one BCL, unified toolchain
	Cross-platform native UI
.NET	Cross-platform web UI
	Cloud native investments
Mono / Xamarin	Continue improvements in speed, size, diagnostics, Azure services

.NET has the best of breed solutions for all modern workloads

.NET Schedule



- .NET 5.0 released on Nov 2020!
- Major releases every year in November
- LTS for even numbered releases
- Predictable schedule, minor releases as needed

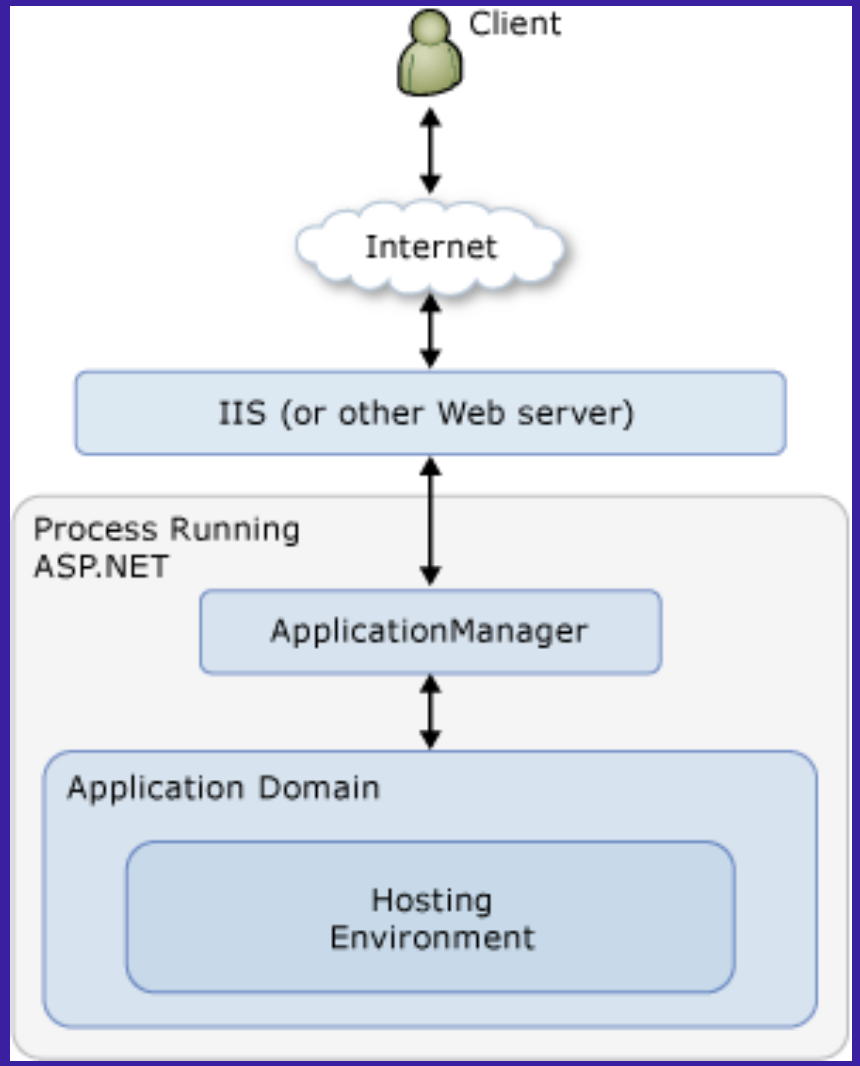
ASP.NET Core

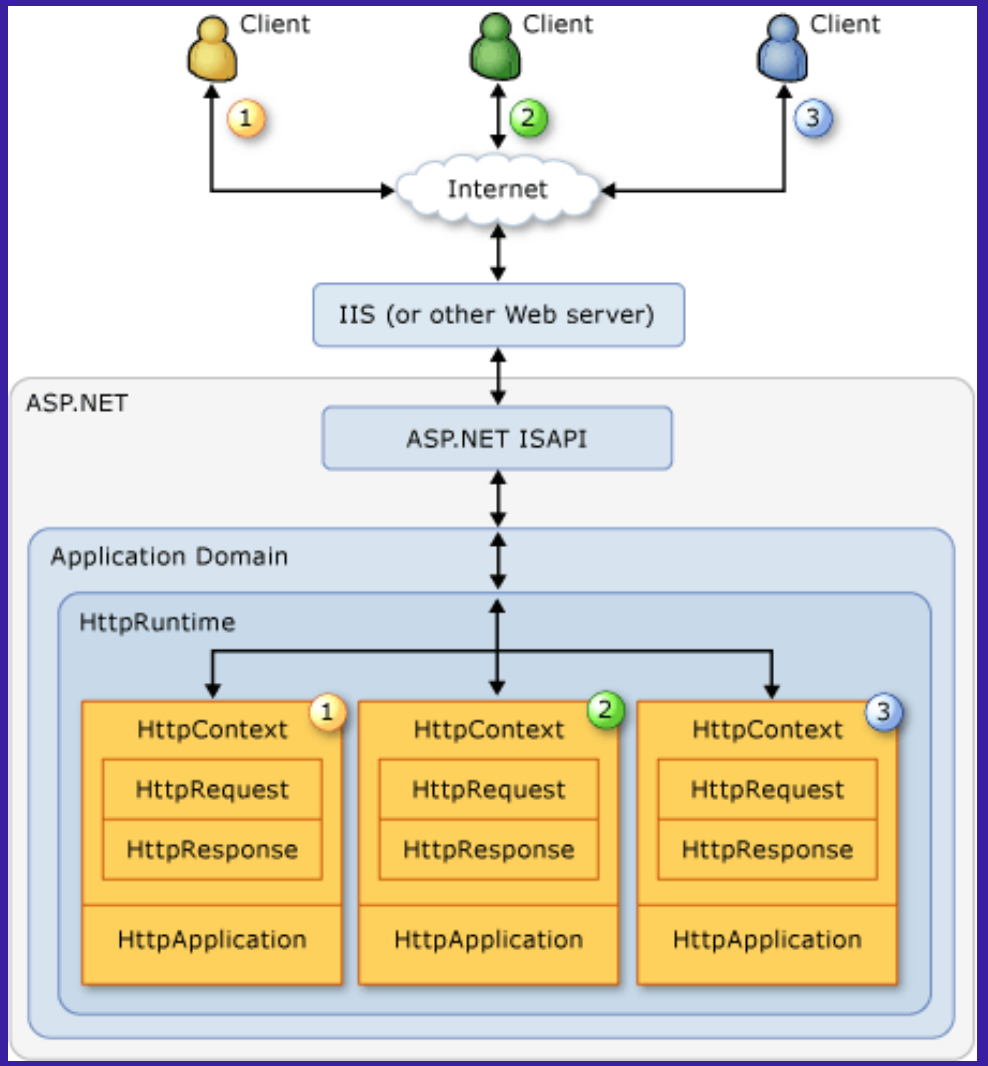
- A cross-platform, high-performance, open-source framework for building modern, cloud-enabled, Internet-connected apps.
- Is a redesign of ASP.NET 4.x, including architectural changes that result in a leaner, more modular framework
- Ability to develop and run on Windows, macOS, and Linux
- A lightweight, high-performance, and modular HTTP request pipeline.

Framework selection

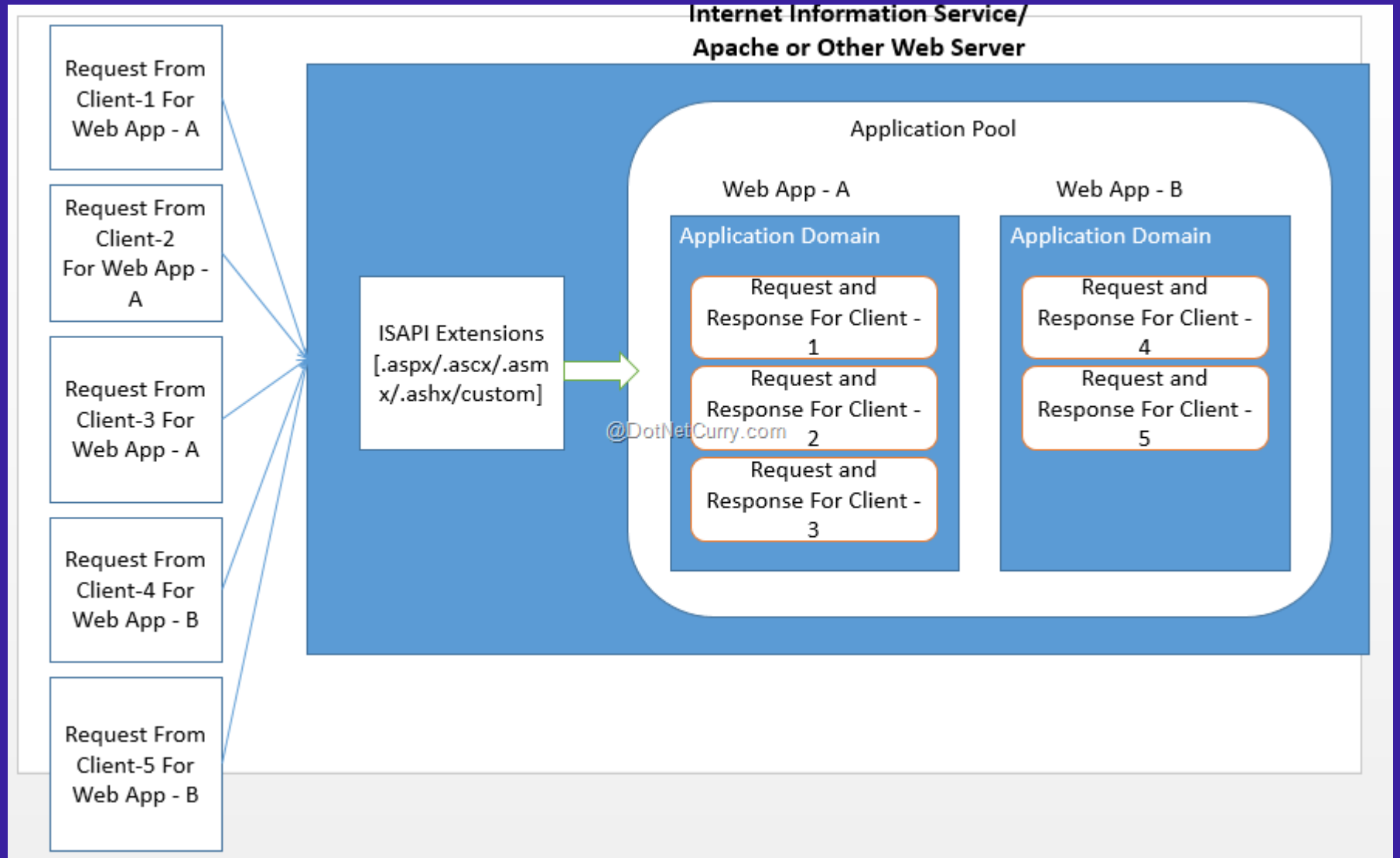
ASP.NET Core	ASP.NET 4.x
Build for Windows, macOS, or Linux	Build for Windows
Razor Pages is the recommended approach to create a Web UI as of ASP.NET Core 2.x. See also MVC, Web API, and SignalR.	Use Web Forms, SignalR, MVC, Web API, WebHooks, or Web Pages
Multiple versions per machine	One version per machine
Develop with Visual Studio, Visual Studio for Mac, or Visual Studio Code using C# or F#	Develop with Visual Studio using C#, VB, or F#
Higher performance than ASP.NET 4.x	Good performance
Use .NET Core runtime	Use .NET Framework runtime

Hosting in Windows



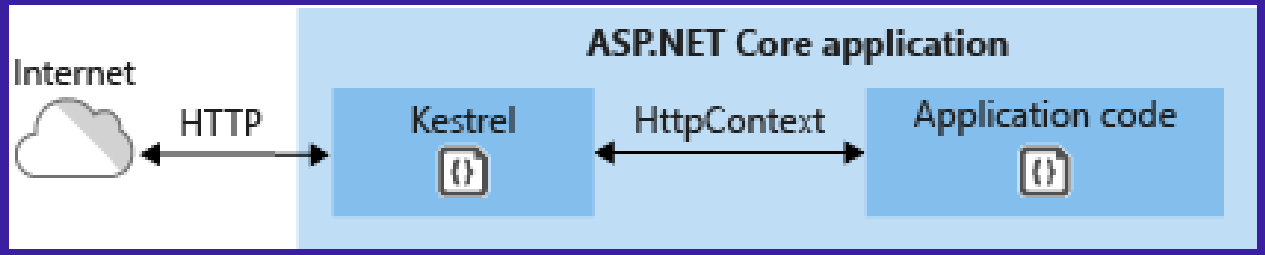


IIS Internals



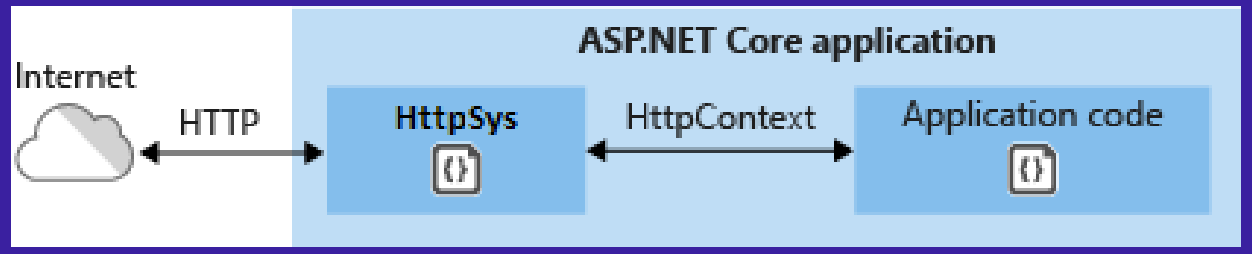
Web server implementations - ASP.NET Core

- Kestrel
  - default, cross-platform HTTP server implementation
  - provides the best performance and memory utilization
  - doesn't have some of the advanced features in HTTP.sys
  - Agility, it's developed and patched independent of the OS.



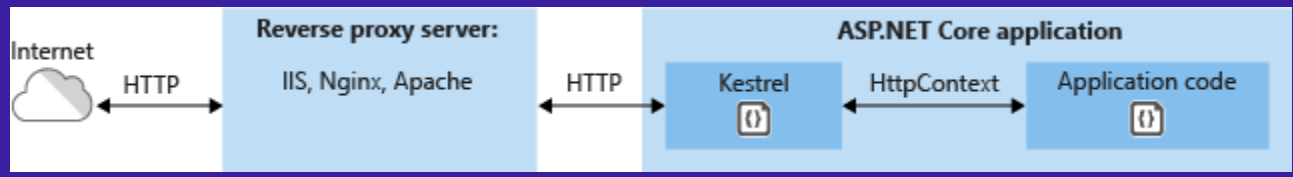
Web server implementations - ASP.NET Core

- HTTP.sys
  - web server for ASP.NET core that runs only on Windows
  - alternative to Kestrel and also provides some extra features
  - Supports windows authentication, port sharing, web sockets
  - Agility, it's developed and patched independent of the OS.



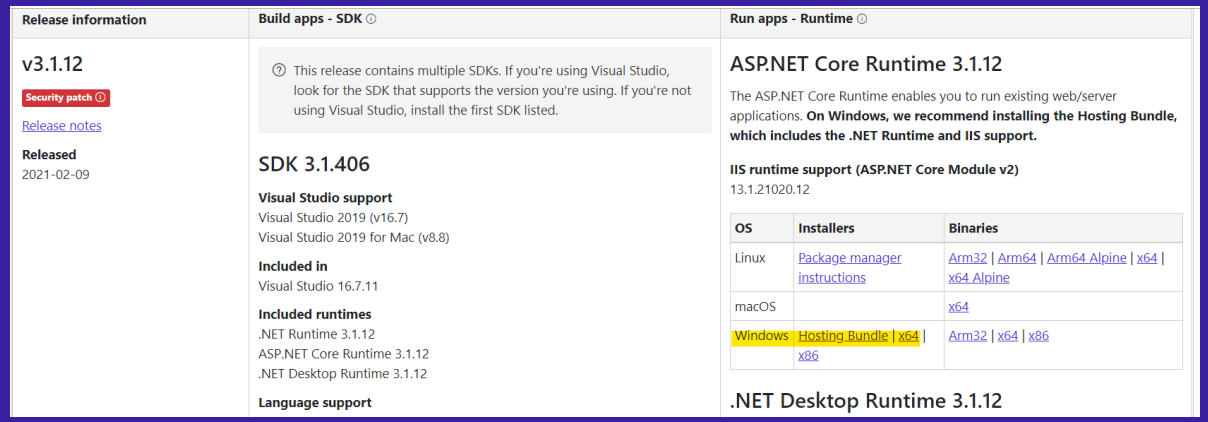
Web Server implementations - ASP.NET Core

- IIS
  - A flexible, secure and manageable Web Server for hosting web apps, including ASP.NET Core.
  - Supports Windows Only
  - Needs a module called ASP.NET Core Hosting Bundle to be installed
  - Supports In Process as well as out-of-process hosting



ASP.NET Core Hosting Bundle

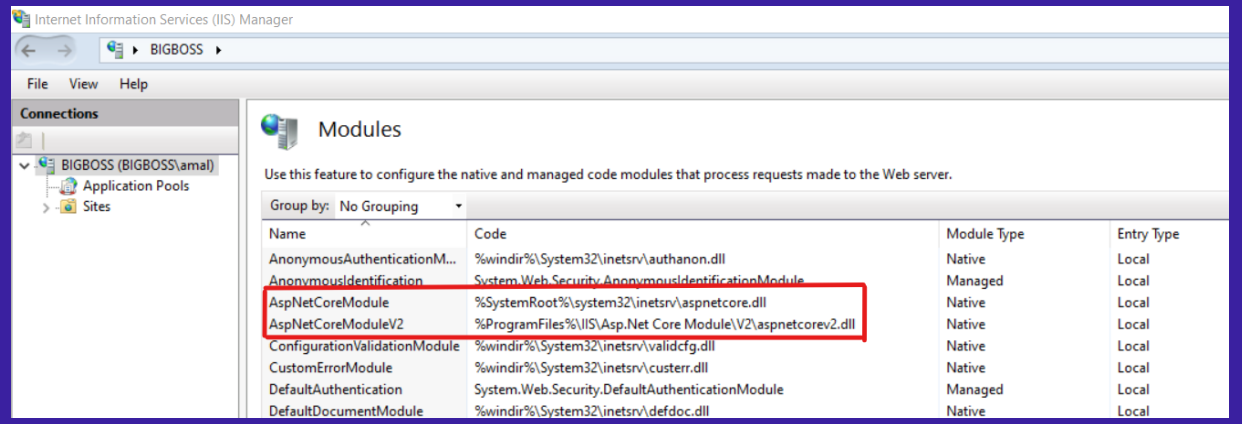
- An installer for the .NET Core Runtime and the ASP.NET Core Module



Release information	Build apps - SDK	Run apps - Runtime
v3.1.12 Security patch Release notes **Released** 2021-02-09	This release contains multiple SDKs. If you're using Visual Studio, look for the SDK that supports the version you're using. If you're not using Visual Studio, install the first SDK listed. **SDK 3.1.406** **Visual Studio support** Visual Studio 2019 (v16.7) Visual Studio 2019 for Mac (v8.8) **Included in** Visual Studio 16.7.11 **Included runtimes** .NET Runtime 3.1.12 ASP.NET Core Runtime 3.1.12 .NET Desktop Runtime 3.1.12 **Language support**	**ASP.NET Core Runtime 3.1.12** The ASP.NET Core Runtime enables you to run existing web/server applications. **On Windows, we recommend installing the Hosting Bundle, which includes the .NET Runtime and IIS support.** **IIS runtime support (ASP.NET Core Module v2)** 13.1.21020.12

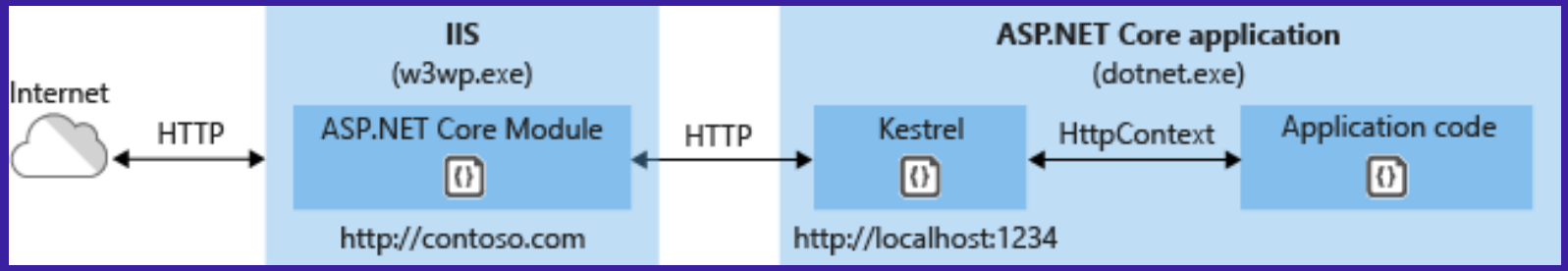
OS	Installers	Binaries
Linux	Package manager instructions	Arm32 | Arm64 | Arm64 Alpine | x64 | x64 Alpine
macOS		x64
Windows	Hosting Bundle | x64 | x86	Arm32 | x64 | x86

.NET Desktop Runtime 3.1.12



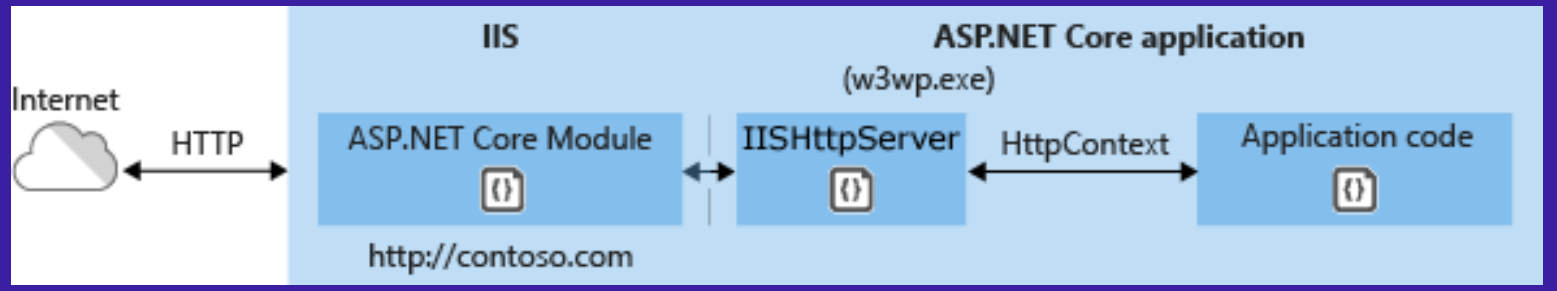
IIS – Out-of-Process Hosting

- Runs an ASP.NET Core app as a diff process
- Performance hit is there



IIS – In-Process Hosting

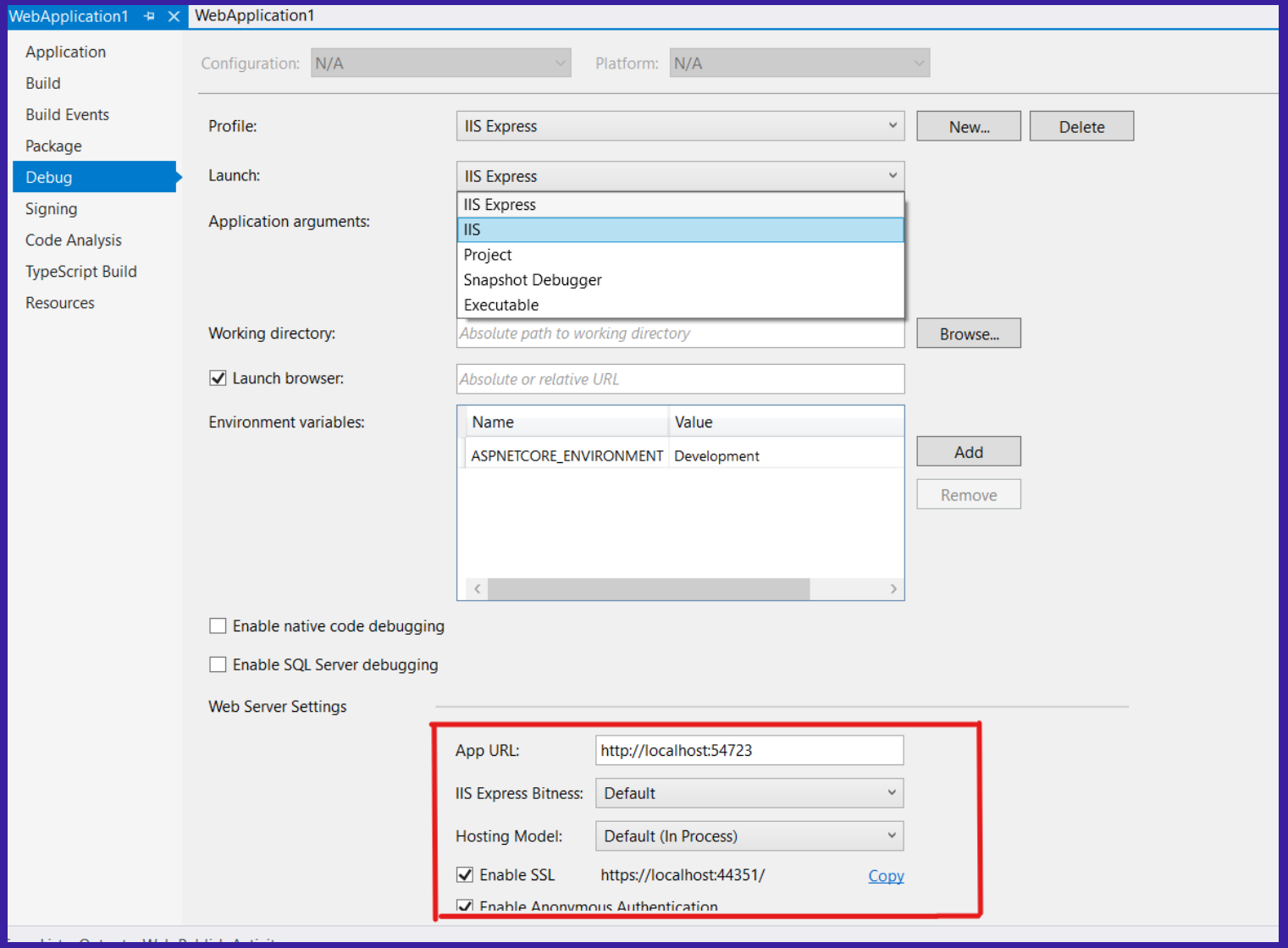
- Runs an ASP.NET Core app in the same process as its IIS worker process
- Provides improved performance because requests aren't proxied



Configuration File

```
<?xml version="1.0" encoding="utf-8"?>
<configuration>
  <location path="." inheritInChildApplications="false">
    <system.webServer>
      <handlers>
        <add name="aspNetCore" path="*" verb="*" modules="AspNetCoreModuleV2" resourceType="Unspecified" />
      </handlers>
      <aspNetCore processPath="dotnet"
                  arguments=".\MyApp.dll"
                  stdoutLogEnabled="false"
                  stdoutLogFile=".\logs\stdout"
                  hostingModel="inprocess" />
    </system.webServer>
  </location>
</configuration>
```

Visual Studio Integration



Program.cs

```
namespace WebApplication1
{
    0 references
    public class Program
    {
        0 references
        public static void Main(string[] args)
        {
            CreateHostBuilder(args).Build().Run();
        }

        1 reference
        public static IHostBuilder CreateHostBuilder(string[] args) =>
            Host.CreateDefaultBuilder(args)
                .ConfigureWebHostDefaults(webBuilder =>
                {
                    webBuilder.UseStartup<Startup>();
                });
    }
}
```

Startup.cs

```
namespace WebApplication1
{
    1 reference
    public class Startup
    {
        // This method gets called by the runtime. Use this method to add services to the container.
        // For more information on how to configure your application, visit https://go.microsoft.com/fwlink/?LinkID=398940
        0 references
        public void ConfigureServices(IServiceCollection services)
        {
        }

        // This method gets called by the runtime. Use this method to configure the HTTP request pipeline.
        0 references
        public void Configure(IApplicationBuilder app, IWebHostEnvironment env)
        {
            if (env.IsDevelopment())
            {
                app.UseDeveloperExceptionPage();
            }

            app.UseRouting();

            app.UseEndpoints(endpoints =>
            {
                endpoints.MapGet("/", async context =>
                {
                    await context.Response.WriteAsync("Hello World!");
                });
            });
        }
    }
}
```

Thanks for joining!

Ask questions on Twitter using #dotNETConf