

Getting Started with .NET Core

Jeremy Clark

www.jeremybytes.com

@jeremybytes



What is .NET Core?

- The new face of .NET
- Quickly evolving & open source
- Cross-platform (Windows, macOS, Linux)

What I like about .NET Core

- .NET Core is easy to get installed and configured
- Self-hosted web applications with Kestrel
- Vastly simplified ASP.NET templates
- Built-in Dependency Injection works for many scenarios

What I like about .NET Core

- Command-line interface (CLI)
- Command-line help is actually helpful
- The same tool chain can be used across platforms (CLI, Visual Studio Code)
- Cross-platform actually works! (great for containers)

Why .NET Core?

- .NET Core is the future
- New C# 8 language features will *NOT* come to .NET Framework
- .NET Standard 2.1 will *NOT* support .NET Framework
- .NET Core 3.0 support WinForms & WPF (for real)
Don't get too excited, this is not cross-platform, but it's still important for moving forward

What you need to build applications with .NET Core

- .NET Core SDK

<https://dotnet.microsoft.com/download>

- Visual Studio Code (or Visual Studio)

<https://code.visualstudio.com/download>

- C# Extension for Visual Studio Code

<https://marketplace.visualstudio.com/items?itemName=ms-vscode.csharp>

CLI: Command-line Interface

- `dotnet`
- `dotnet new`
- `dotnet add package`
- `dotnet build`
- `dotnet run`

dotnet -h

```
c:\sdd2019>dotnet -h
.NET Core SDK (3.0.100-preview3-010431)
Usage: dotnet [runtime-options] [path-to-application] [arguments]

Execute a .NET Core application.

runtime-options:
  --additionalprobingpath <path>          Path containing probing policy and assemblies
  --additional-deps <path>                Path to additional deps.json file.
  --fx-version <version>                 Version of the installed Shared Framework to u
  --roll-forward-on-no-candidate-fx <n>  Roll forward on no candidate framework (0=off,

path-to-application:
  The path to an application .dll file to execute.

Usage: dotnet [sdk-options] [command] [command-options] [arguments]
```


dotnet -h

SDK commands:

add	Add a package or reference to a .NET project.
build	Build a .NET project.
build-server	Interact with servers started by a build.
clean	Clean build outputs of a .NET project.
help	Show command line help.
list	List project references of a .NET project.
migrate	Migrate a project.json project to an MSBuild project.
msbuild	Run Microsoft Build Engine (MSBuild) commands.
new	Create a new .NET project or file.
nuget	Provides additional NuGet commands.
pack	Create a NuGet package.
publish	Publish a .NET project for deployment.
remove	Remove a package or reference from a .NET project.
restore	Restore dependencies specified in a .NET project.
run	Build and run a .NET project output.
sln	Modify Visual Studio solution files.
store	Store the specified assemblies in the runtime package store.
test	Run unit tests using the test runner specified in a .NET project.
tool	Install or manage tools that extend the .NET experience.
vstest	Run Microsoft Test Engine (VSTest) commands.

dotnet new

Templates	Short Name	Language	Tags
Console Application	console	[C#], F#, VB	Common/Console
Class library	classlib	[C#], F#, VB	Common/Library
WPF Application	wpf	[C#]	Common/WPF
Windows Forms Application	winforms	[C#]	Common/WinForms
Worker Service	worker	[C#]	Common/Worker/Web
Unit Test Project	mstest	[C#], F#, VB	Test/MSTest
NUnit 3 Test Project	nunit	[C#], F#, VB	Test/NUnit
NUnit 3 Test Item	nunit-test	[C#], F#, VB	Test/NUnit
xUnit Test Project	xunit	[C#], F#, VB	Test/xUnit
Razor Page	page	[C#]	Web/ASP.NET
MVC ViewImports	viewimports	[C#]	Web/ASP.NET
MVC ViewStart	viewstart	[C#]	Web/ASP.NET
ASP.NET Core Empty	web	[C#], F#	Web/Empty
ASP.NET Core Web App (Model-View-Controller)	mvc	[C#], F#	Web/MVC
ASP.NET Core Web App	webapp	[C#]	Web/MVC/Razor Pages
ASP.NET Core with Angular	angular	[C#]	Web/MVC/SPA
ASP.NET Core with React.js	react	[C#]	Web/MVC/SPA
ASP.NET Core with React.js and Redux	reactredux	[C#]	Web/MVC/SPA
Razor Class Library	razorclasslib	[C#]	Web/Razor/Library/Razor Class Library
ASP.NET Core Web App (Razor Components)	razorcomponents	[C#]	Web/RazorComponents
ASP.NET Core Web API	webapi	[C#], F#	Web/WebAPI
ASP.NET Core gRPC Service	grpc	[C#]	Web/gRPC
global.json file	globaljson		Config
NuGet Config	nugetconfig		Config
Dotnet local tool manifest file	tool-manifest		Config
Web Config	webconfig		Config
Solution File	sln		Solution

Creating & Running

- New Projects

- `dotnet new webapi`

- `dotnet new console`

- Build / Run

- `dotnet build`

- `dotnet run`

Adding Packages / References

- NuGet Packages

```
dotnet add package Newtonsoft.Json
```

- Add Reference

```
dotnet add reference ../folder/my-library.csproj
```

Solutions

- Solution Files

```
dotnet new sln
```

```
dotnet sln add ./folder1/my-console.csproj
```

```
dotnet sln add ./folder2/my-library.csproj
```

Async

- "async" Main in a console application

- Open .proj

- Add to Properties group section

<LangVersion>latest</LangVersion>

No Longer Needed

ASP.NET Dependency Injection

- Auto-inject dependencies as parameters in Controller constructors
- In Startup.cs

```
services.AddSingleton<IPeopleProvider, StaticPeopleProvider>()
```
- Lifetimes: Singleton, Scoped, Transient



Thank You!

Jeremy Clark

- <http://www.jeremybytes.com>
- jeremy@jeremybytes.com
- @jeremybytes