We have got the company!

- **Fruit flies** were the first living creatures to be sent into space. Lucky one!
- A **bee**’s wings beat 190 times a second, that’s 11,400 times a minute. OMG!
- **Caterpillars** have 12 eyes!
- One **dung beetle** can drag 1,141 times its weight – that’s like a human pulling six double-decker buses!
- **Grasshoppers** existed before dinosaurs!

DotNetCore

Prepared for Vth semester DDU-CE students 2022-23 WAD

Apurva A Mehta

.NET Core

- .NET Core is a *software development framework* which is used to create different types of applications.
- There are many frameworks which are written on top of .NET Core for creating various applications.
- .NET Core is a cross-platform, high-performance, open-source framework for building modern, cloud-based, internet-connected applications.

Benefits and Features

- Cross Platform
- Unified programming model for MVC and Web API
- Dependency Injection
- Testability
- Open Source
- Modular
- Command line tool support

Cross Platform

- ASP.NET 4.x applications → windows platform
- .NET Core applications can be **developed** and **run** across different platforms like Windows, macOS, or Linux.
- ASP.NET 4.x applications can be hosted only on IIS.
- .NET Core applications can be hosted on IIS, Apache, Docker, or even self-host in your own process.
- From a development standpoint, you can either use Visual Studio or Visual Studio Code, Sublime, Bracket, Vim, Etc… for building .NET Core applications.

Unified programming model

- With ASP.NET core, we use the same unified programming model to create MVC style web applications and ASP.NET Web API's.

Dependency Injection

- ASP.NET Core has built-in support for dependency injection.

Testability

- With built-in dependency injection and the unified programming model for creating Web Applications and Web API's, unit testing ASP.NET Core applications is straight forward.

Open-source and community-focused

- https://github.com/dotnet/core
- ASP.NET Core is fully open source and is being actively developed by the .NET team in collaboration with a vast community of open source developers.
- ASP.NET core is continually evolving as the vast community behind it is suggesting ways to improve it and help fix bugs and problems.
- This means we have a more secure and better quality software.
- MIT Licence (Private and Commercial use)

Modular HTTP Request Pipeline

- ASP.NET Core Provides Modularity with Middleware Components in ASP.NET Core
- We compose the request and response pipeline using the middleware components.
- It includes a rich set of built-in middleware components.
- We can also write our own custom middleware components.

Command line tool support

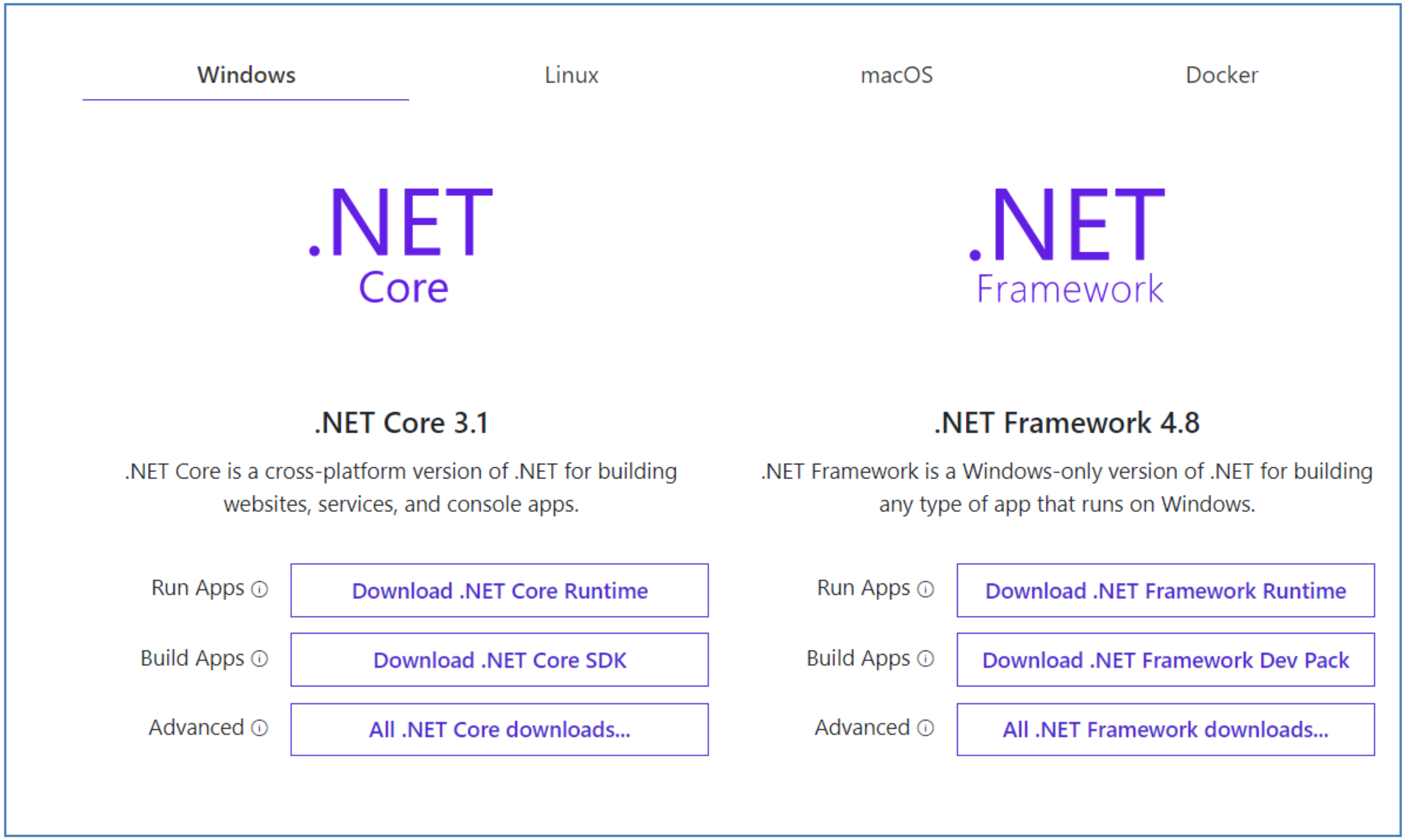
- .NET Core fully supports command line tool which is useful in complete cycle of development.
  - Create new project
  - Add package
  - Build
  - Run
  - Test
  - Deploy
  - Etc...

.NET Core CLI

- .NET CLI helps us to perform almost all the tasks which are required in order to work with .NET Core application.
- .NET CLI works with the command and these commands are applicable on all types of application of .NET Core.
- .NET CLI is a cross platform tool for developing .NET applications.

Download .NET

- https://dotnet.microsoft.com/download



.NET Core 3.1

.NET Core is a cross-platform version of .NET for building websites, services, and console apps.

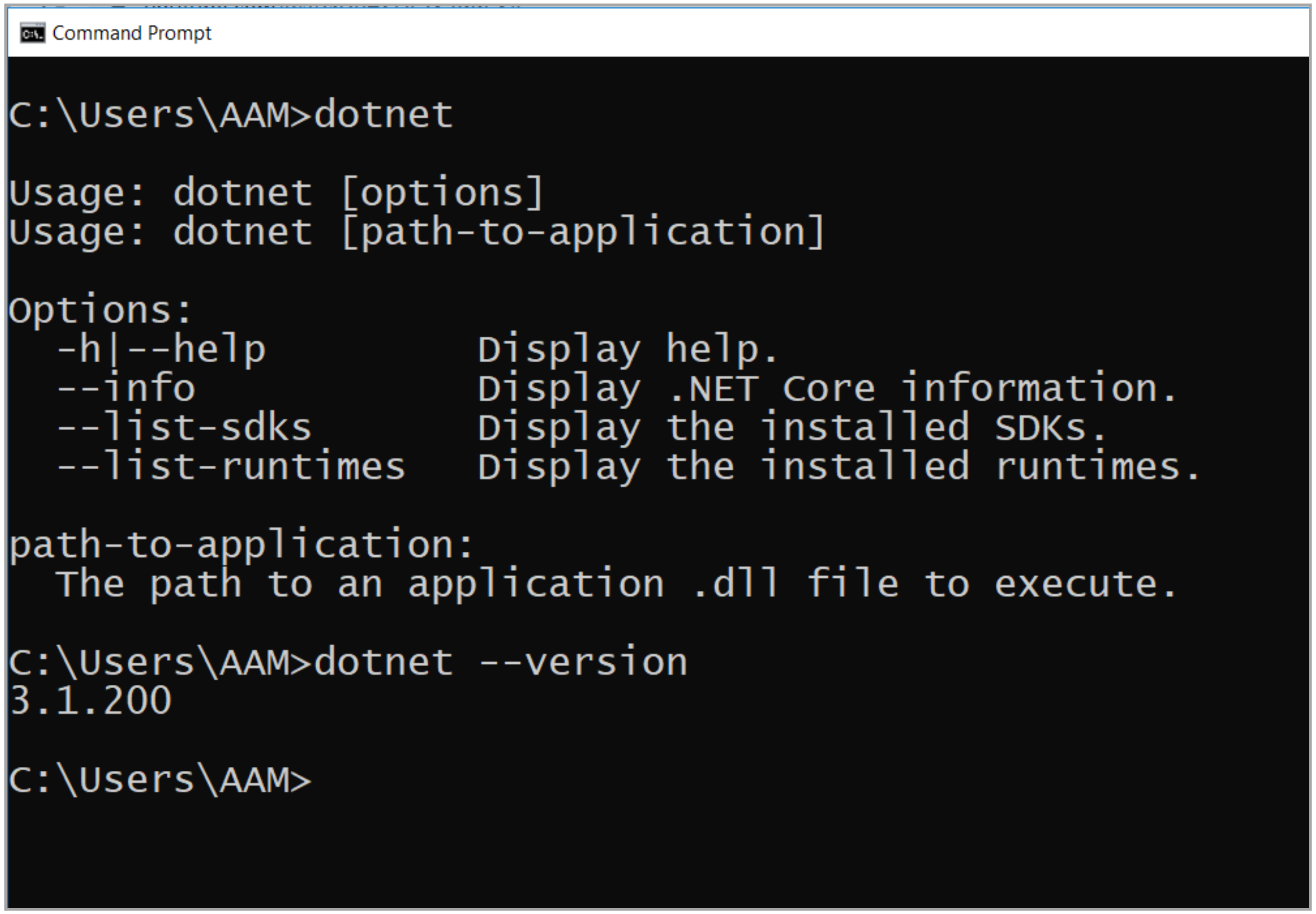
Run Apps ⓘ	Download .NET Core Runtime
Build Apps ⓘ	Download .NET Core SDK
Advanced ⓘ	All .NET Core downloads...

.NET Framework 4.8

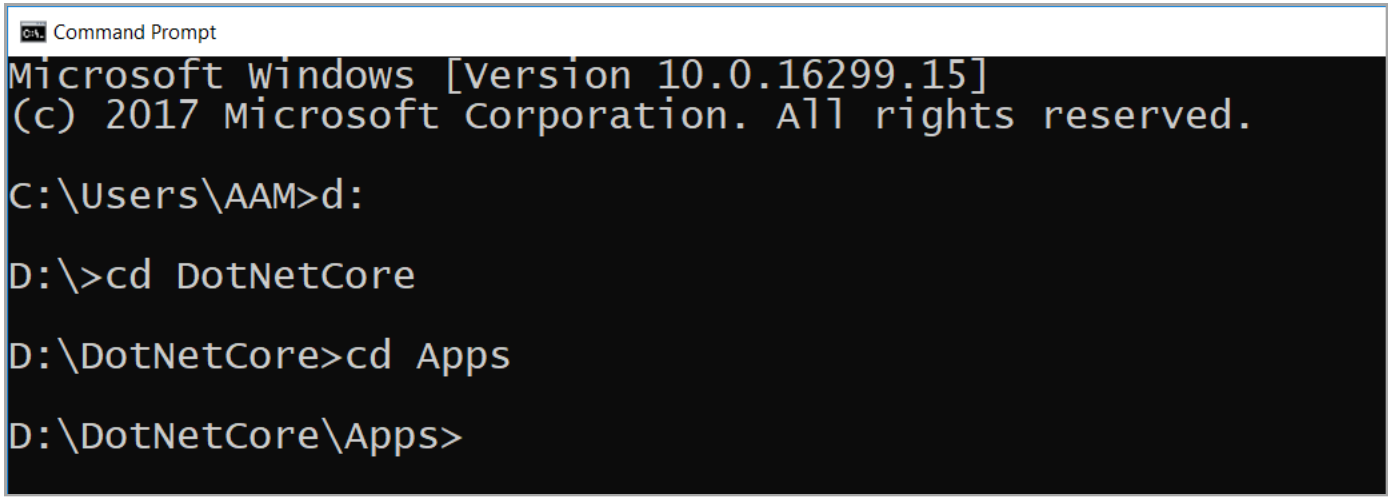
.NET Framework is a Windows-only version of .NET for building any type of app that runs on Windows.

Run Apps ⓘ	Download .NET Framework Runtime
Build Apps ⓘ	Download .NET Framework Dev Pack
Advanced ⓘ	All .NET Framework downloads...

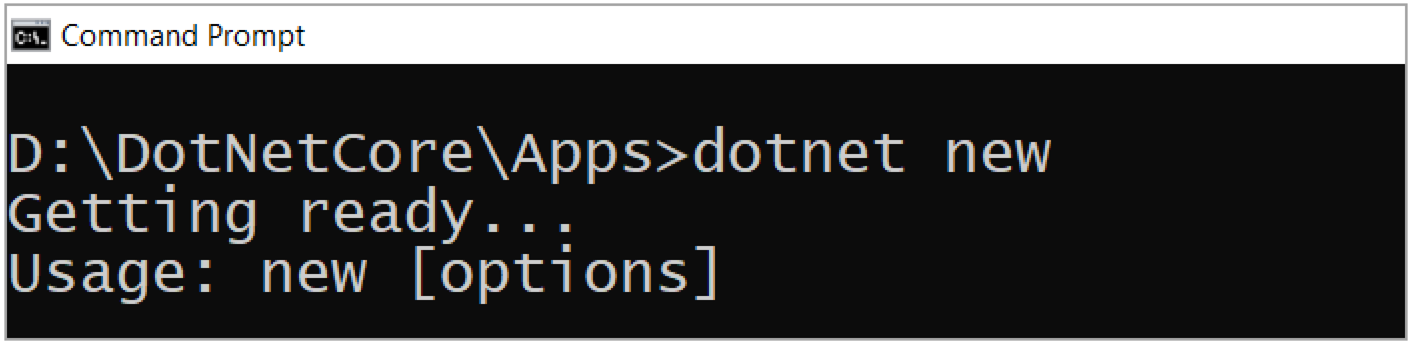
Let’s Rain…



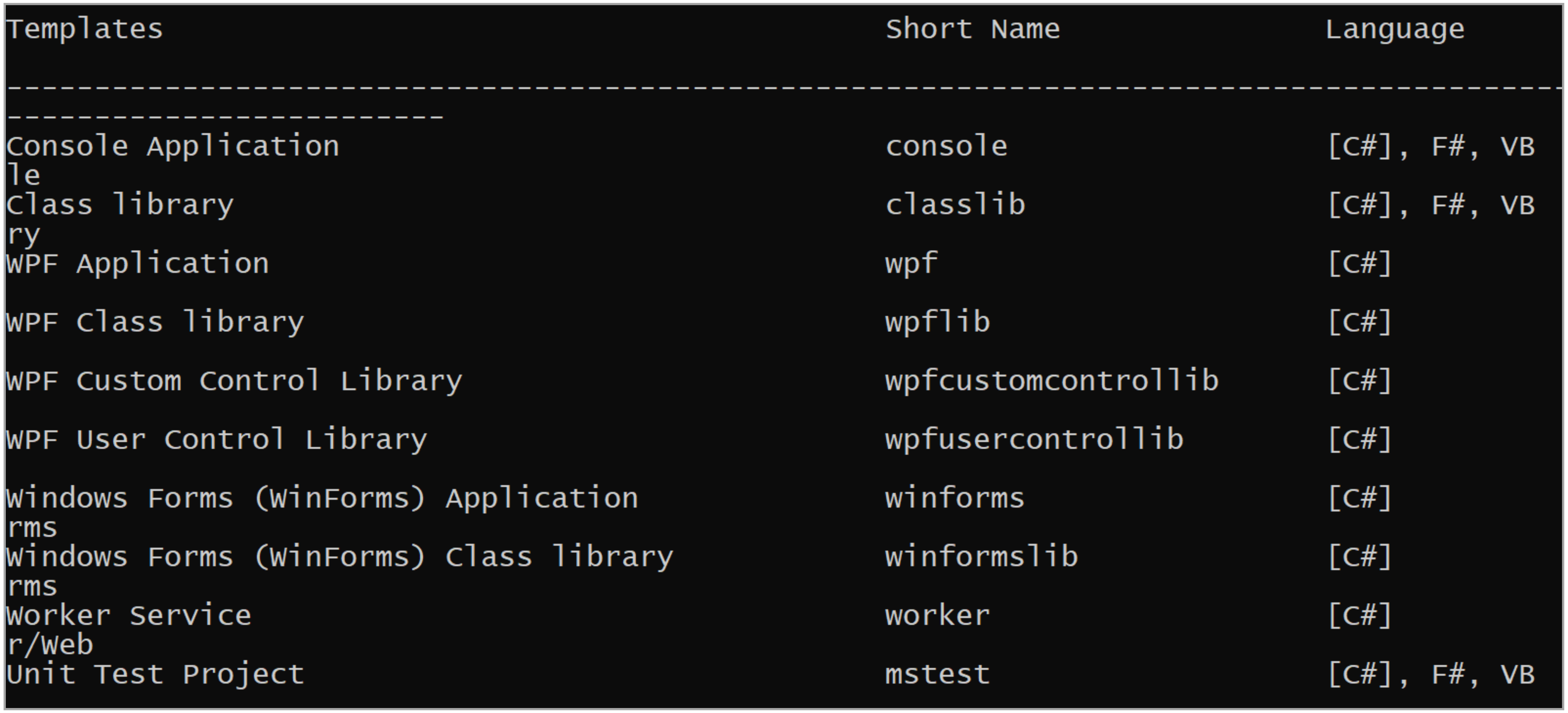
```
C:\Users\AAM>dotnet

Usage: dotnet [options]
Usage: dotnet [path-to-application]

Options:
  -h|--help           Display help.
  --info              Display .NET Core information.
  --list-sdks         Display the installed SDKs.
  --list-runtimes     Display the installed runtimes.

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

C:\Users\AAM>dotnet --version
3.1.200

C:\Users\AAM>
```



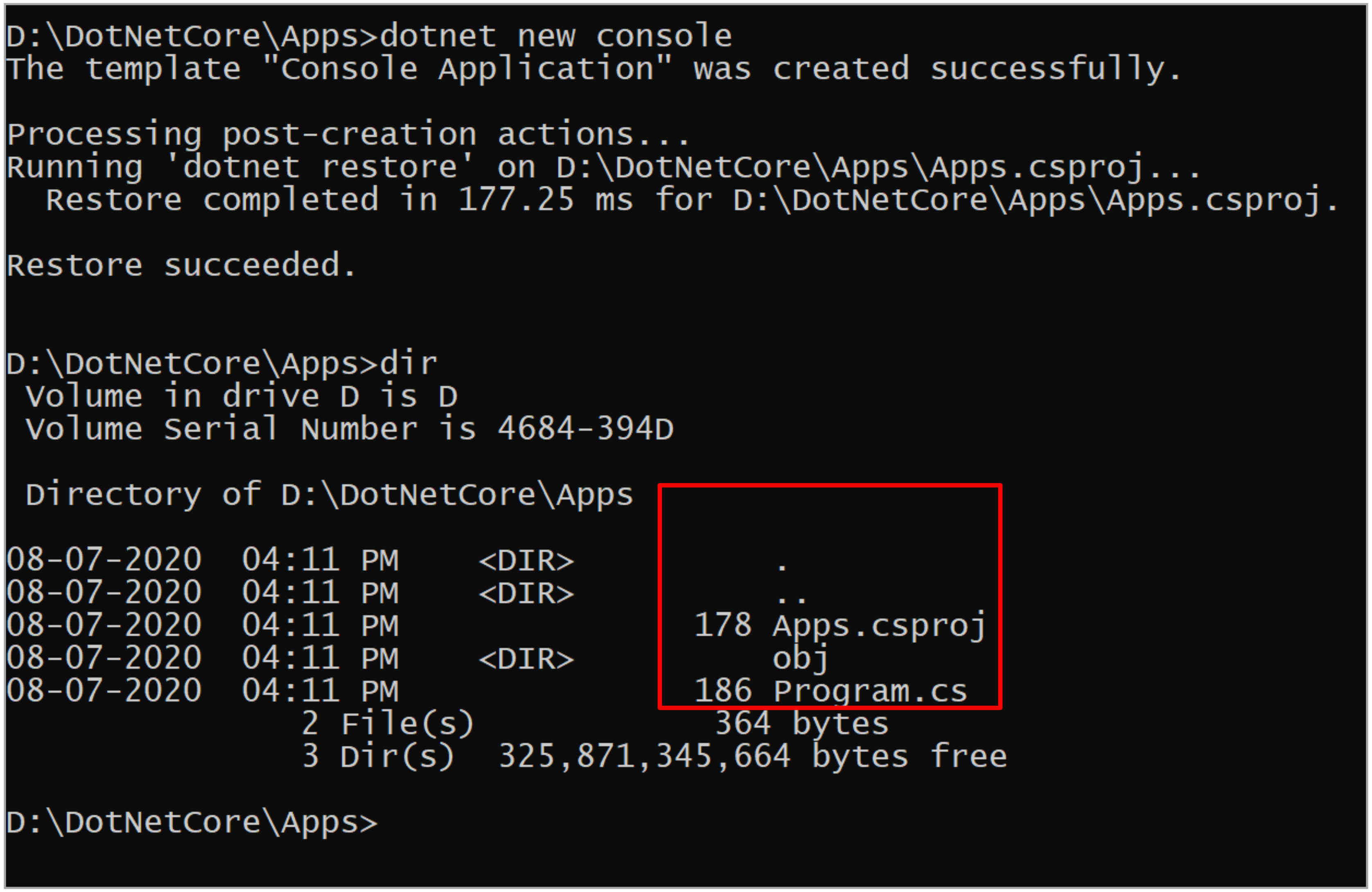
```
Microsoft Windows [Version 10.0.16299.15]
(c) 2017 Microsoft Corporation. All rights reserved.

C:\Users\AAM>d:

D:\>cd DotNetCore

D:\DotNetCore>cd Apps

D:\DotNetCore\Apps>
```



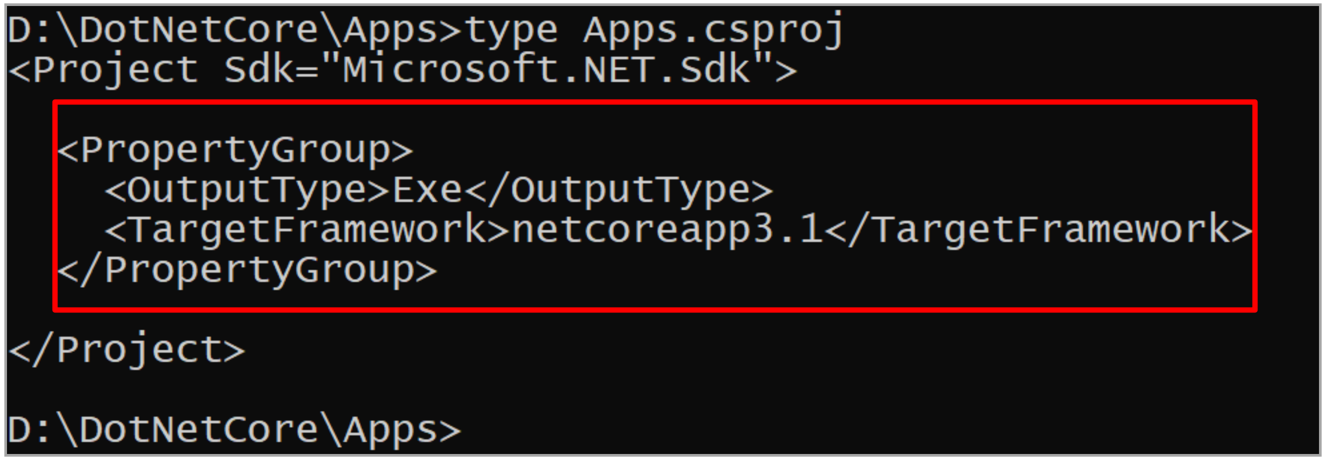
```
D:\DotNetCore\Apps>dotnet new
Getting ready...
Usage: new [options]
```



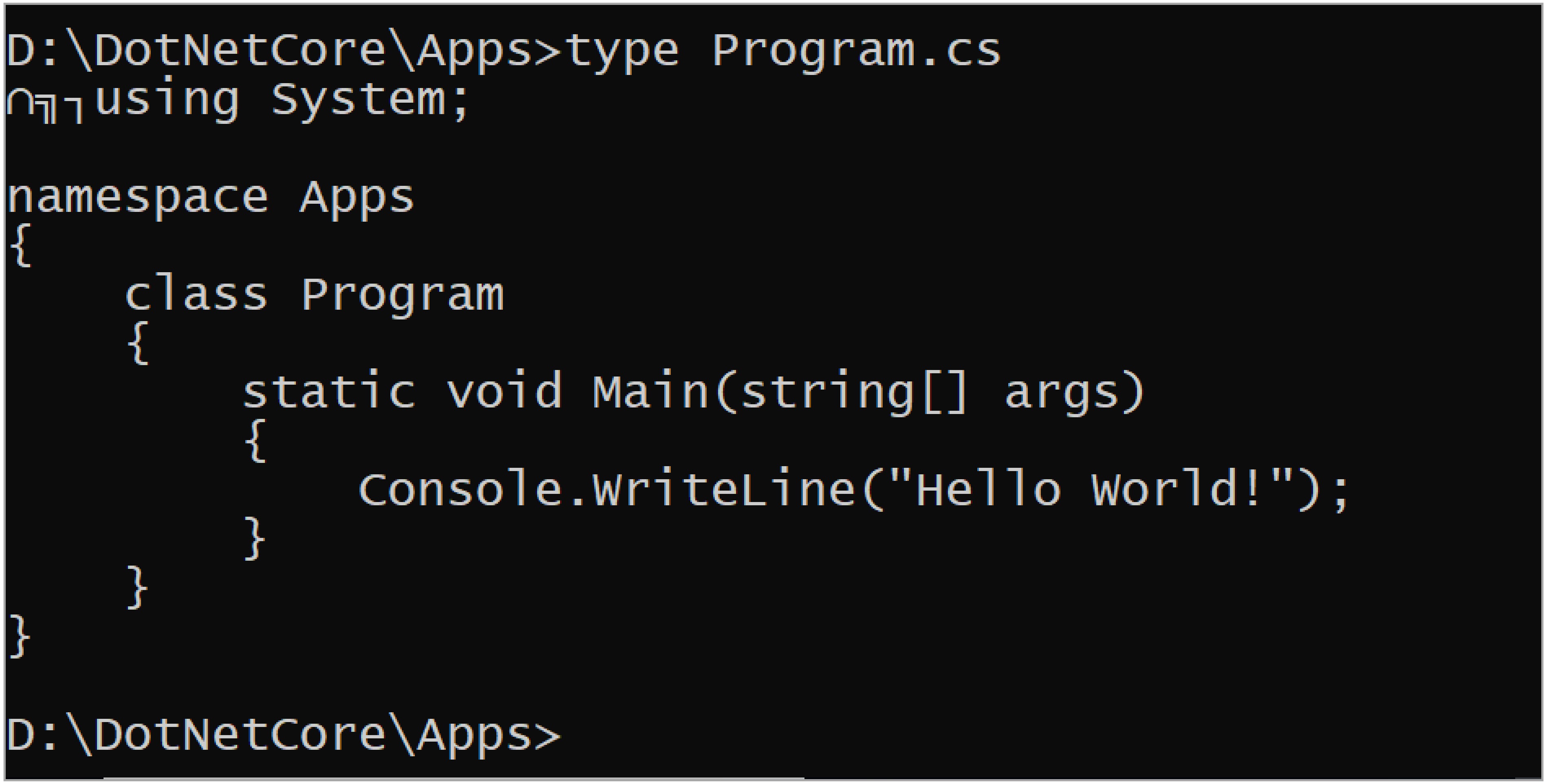
```
Templates                                         Short Name               Language
                                                                                               
-----------------------------------------------------------------------------------------
-------------------------
Console Application                               console                  [C#], F#, VB
le
Class library                                     classlib                 [C#], F#, VB
ry
WPF Application                                   wpf                      [C#]

WPF Class library                                 wpflib                   [C#]

WPF Custom Control Library                        wpfcustomcontrollib      [C#]

WPF User Control Library                          wpfusercontrollib        [C#]

Windows Forms (WinForms) Application              winforms                 [C#]
rms
Windows Forms (WinForms) Class library            winformslib              [C#]
rms
Worker Service                                    worker                   [C#]
r/Web
Unit Test Project                                 mstest                   [C#], F#, VB
```



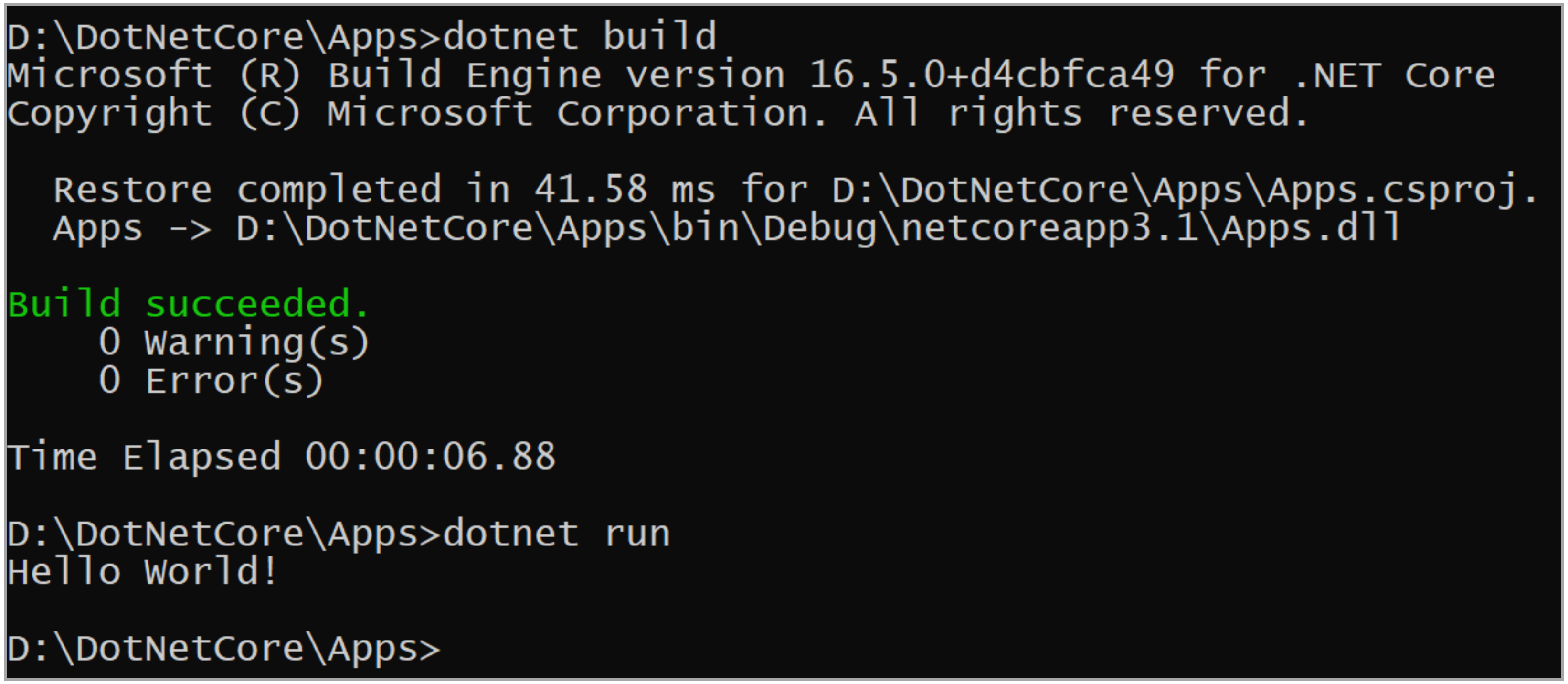
```
D:\DotNetCore\Apps>dotnet new console
The template "Console Application" was created successfully.

Processing post-creation actions...
Running 'dotnet restore' on D:\DotNetCore\Apps\Apps.csproj...
  Restore completed in 177.25 ms for D:\DotNetCore\Apps\Apps.csproj.

Restore succeeded.


D:\DotNetCore\Apps>dir
 Volume in drive D is D
 Volume Serial Number is 4684-394D

 Directory of D:\DotNetCore\Apps

08-07-2020  04:11 PM    <DIR>          .
08-07-2020  04:11 PM    <DIR>          ..
08-07-2020  04:11 PM               178 Apps.csproj
08-07-2020  04:11 PM    <DIR>          obj
08-07-2020  04:11 PM               186 Program.cs
               2 File(s)            364 bytes
               3 Dir(s)  325,871,345,664 bytes free

D:\DotNetCore\Apps>
```



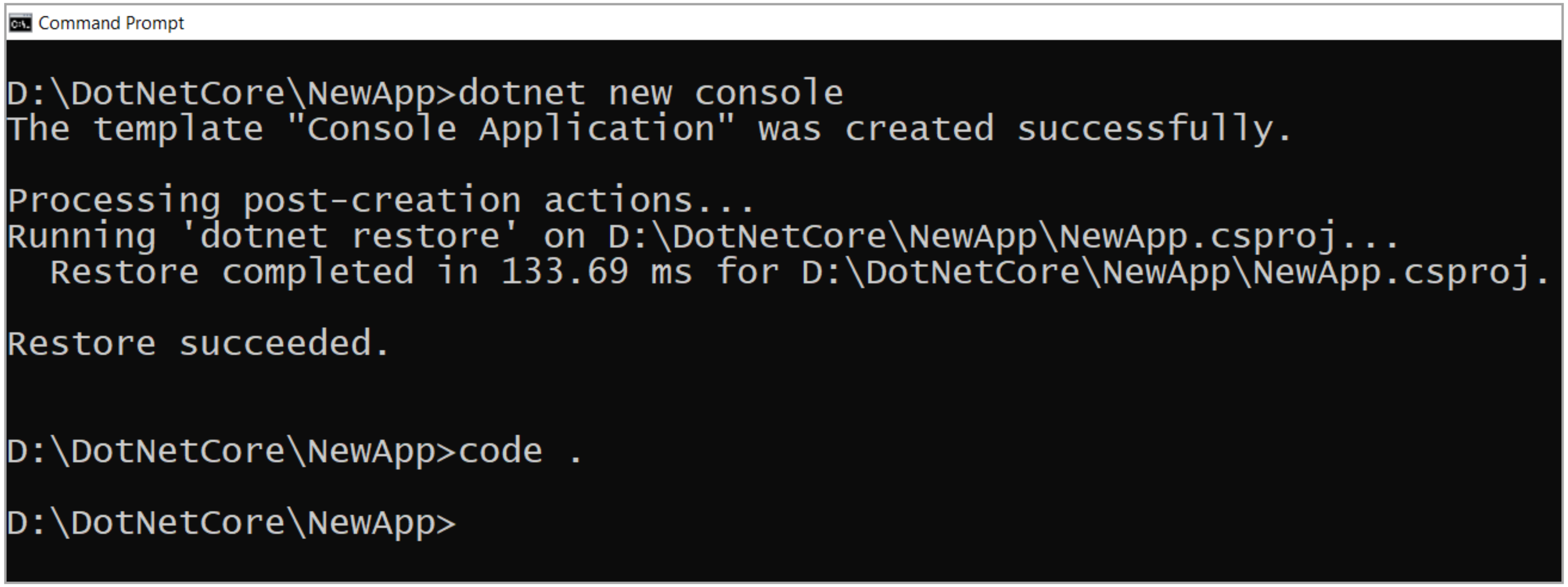
```
D:\DotNetCore\Apps>type Apps.csproj
<Project Sdk="Microsoft.NET.Sdk">

  <PropertyGroup>
    <OutputType>Exe</OutputType>
    <TargetFramework>netcoreapp3.1</TargetFramework>
  </PropertyGroup>

</Project>

D:\DotNetCore\Apps>
```



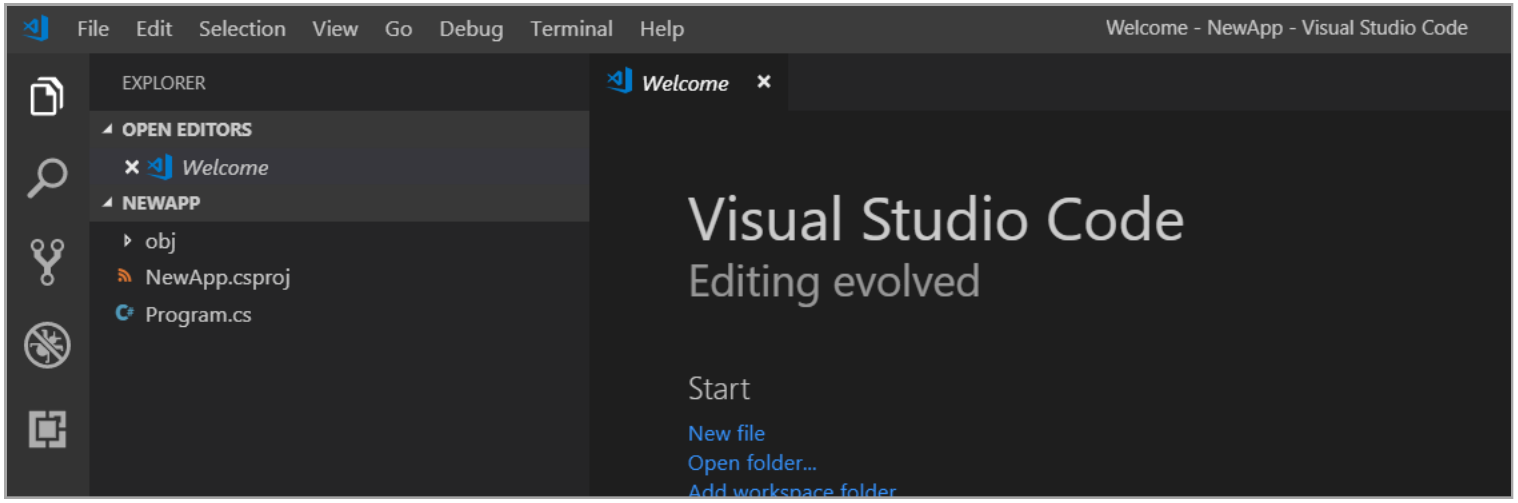
```
D:\DotNetCore\Apps>type Program.cs
∩╗┐using System;

namespace Apps
{
    class Program
    {
        static void Main(string[] args)
        {
            Console.WriteLine("Hello World!");
        }
    }
}

D:\DotNetCore\Apps>
```

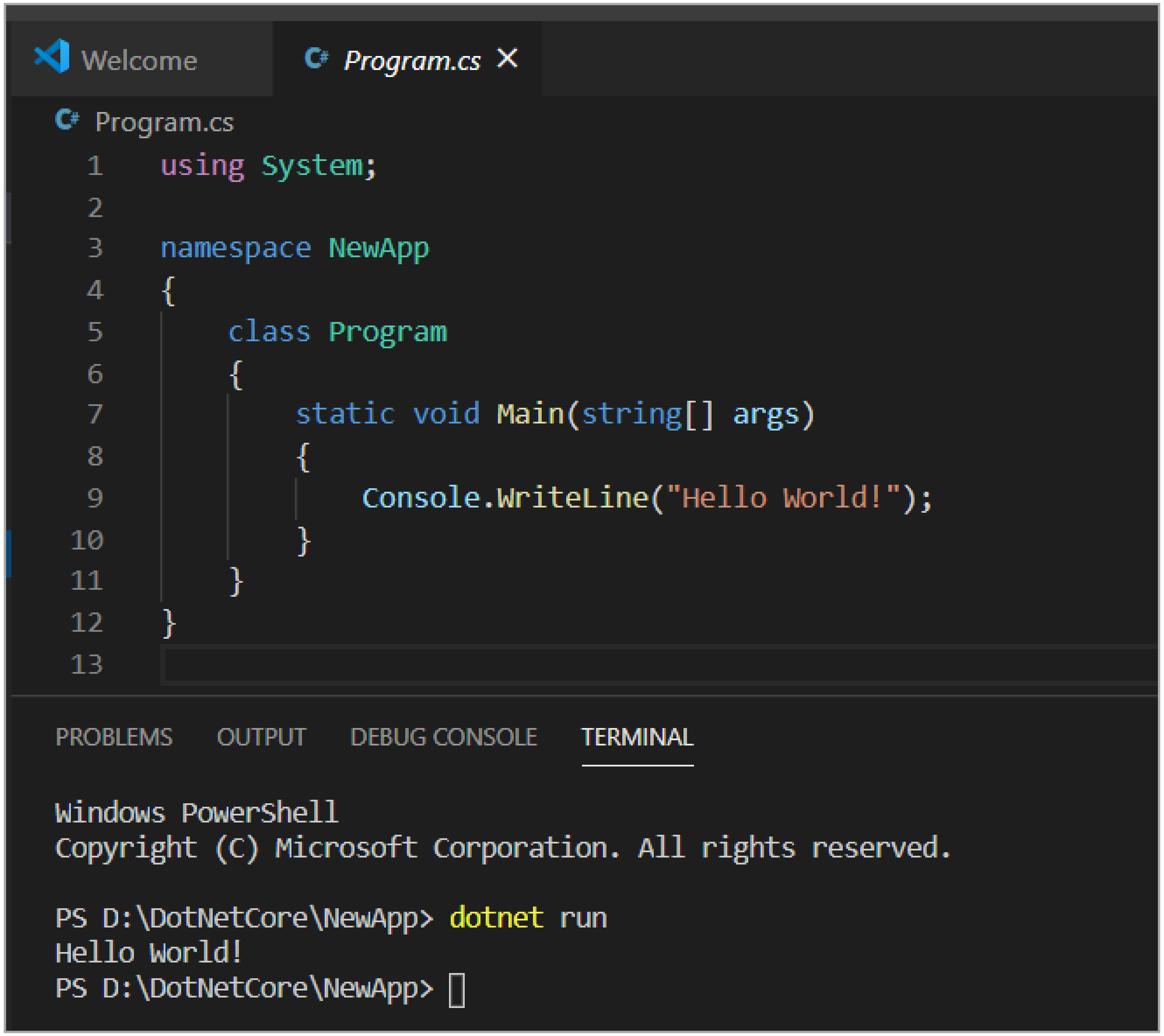


```
D:\DotNetCore\Apps>dotnet build
Microsoft (R) Build Engine version 16.5.0+d4cbfca49 for .NET Core
Copyright (C) Microsoft Corporation. All rights reserved.

  Restore completed in 41.58 ms for D:\DotNetCore\Apps\Apps.csproj.
  Apps -> D:\DotNetCore\Apps\bin\Debug\netcoreapp3.1\Apps.dll

Build succeeded.
    0 Warning(s)
    0 Error(s)

Time Elapsed 00:00:06.88

D:\DotNetCore\Apps>dotnet run
Hello World!

D:\DotNetCore\Apps>
```



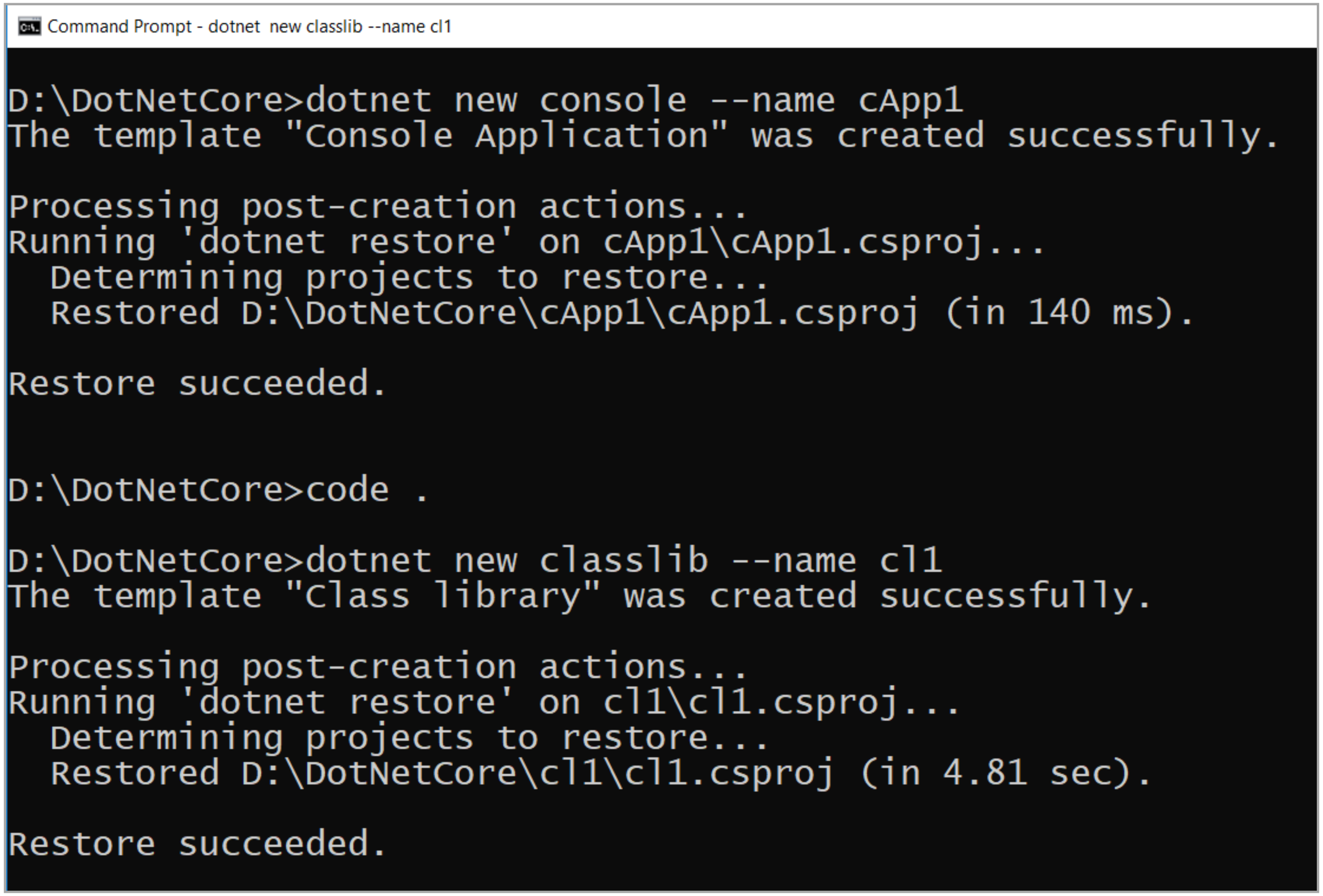
```
Command Prompt

D:\DotNetCore\NewApp>dotnet new console
The template "Console Application" was created successfully.

Processing post-creation actions...
Running 'dotnet restore' on D:\DotNetCore\NewApp\NewApp.csproj...
  Restore completed in 133.69 ms for D:\DotNetCore\NewApp\NewApp.csproj.

Restore succeeded.


D:\DotNetCore\NewApp>code .

D:\DotNetCore\NewApp>
```



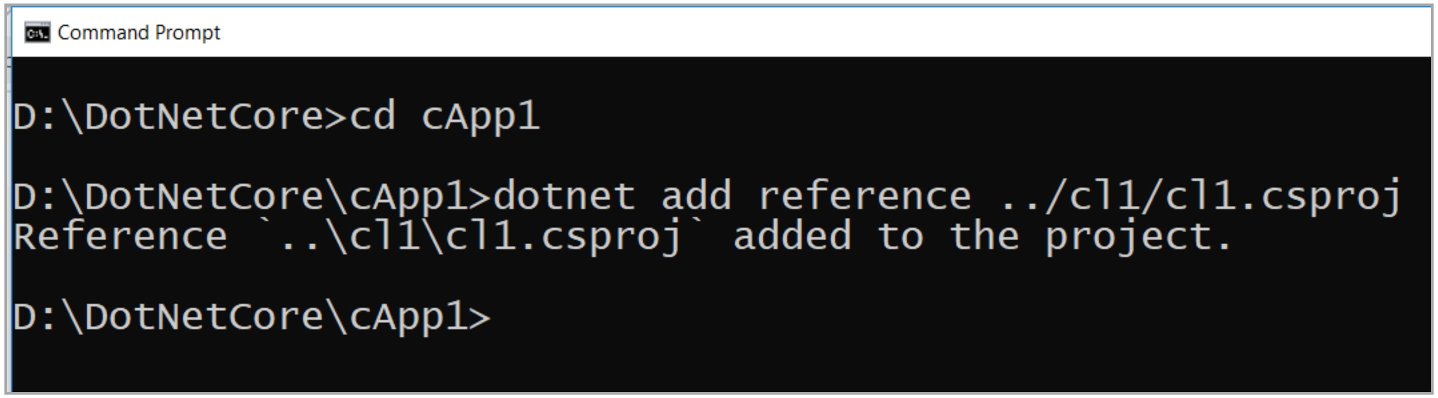


```csharp
using System;

namespace NewApp
{
    class Program
    {
        static void Main(string[] args)
        {
            Console.WriteLine("Hello World!");
        }
    }
}

```

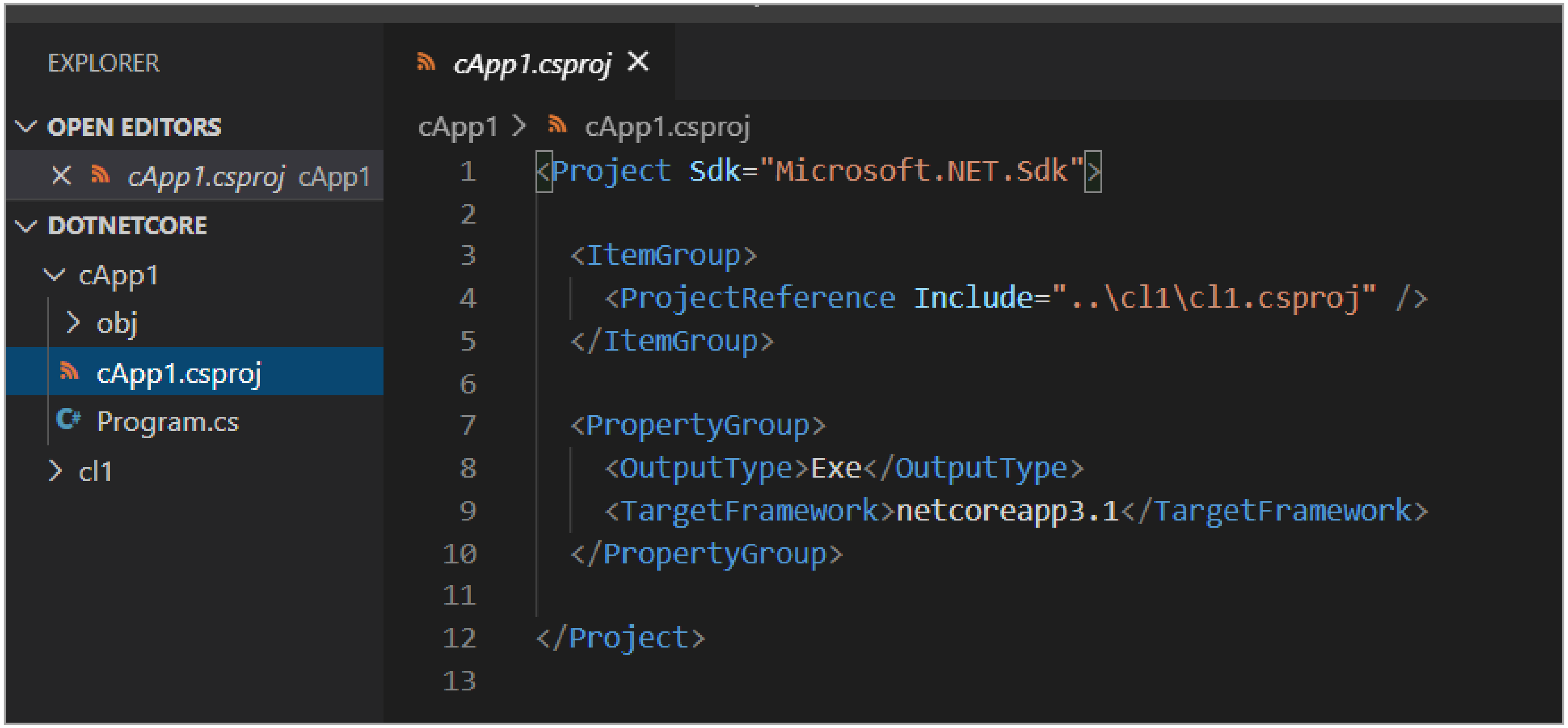
```
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

PS D:\DotNetCore\NewApp> dotnet run
Hello World!
PS D:\DotNetCore\NewApp>
```



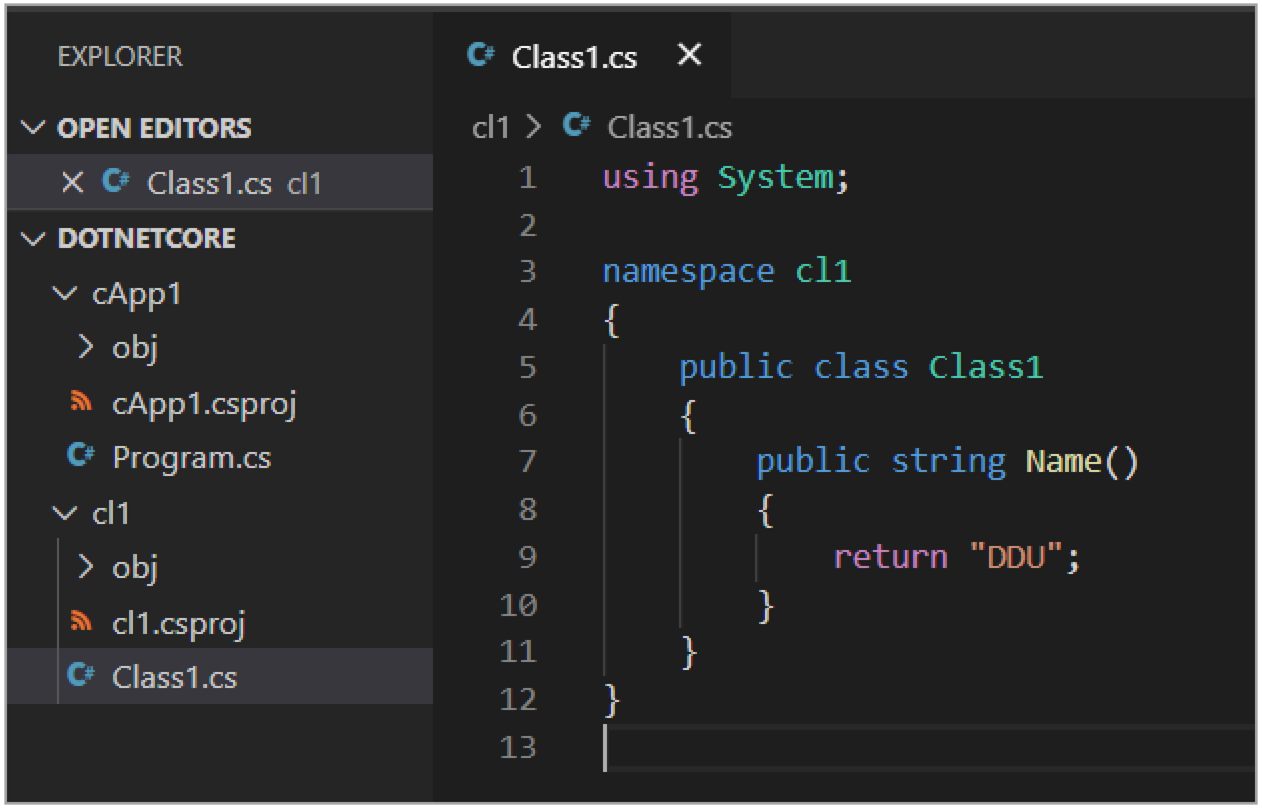
```
D:\DotNetCore>dotnet new console --name cApp1
The template "Console Application" was created successfully.

Processing post-creation actions...
Running 'dotnet restore' on cApp1\cApp1.csproj...
  Determining projects to restore...
  Restored D:\DotNetCore\cApp1\cApp1.csproj (in 140 ms).

Restore succeeded.


D:\DotNetCore>code .

D:\DotNetCore>dotnet new classlib --name cl1
The template "Class library" was created successfully.

Processing post-creation actions...
Running 'dotnet restore' on cl1\cl1.csproj...
  Determining projects to restore...
  Restored D:\DotNetCore\cl1\cl1.csproj (in 4.81 sec).

Restore succeeded.
```



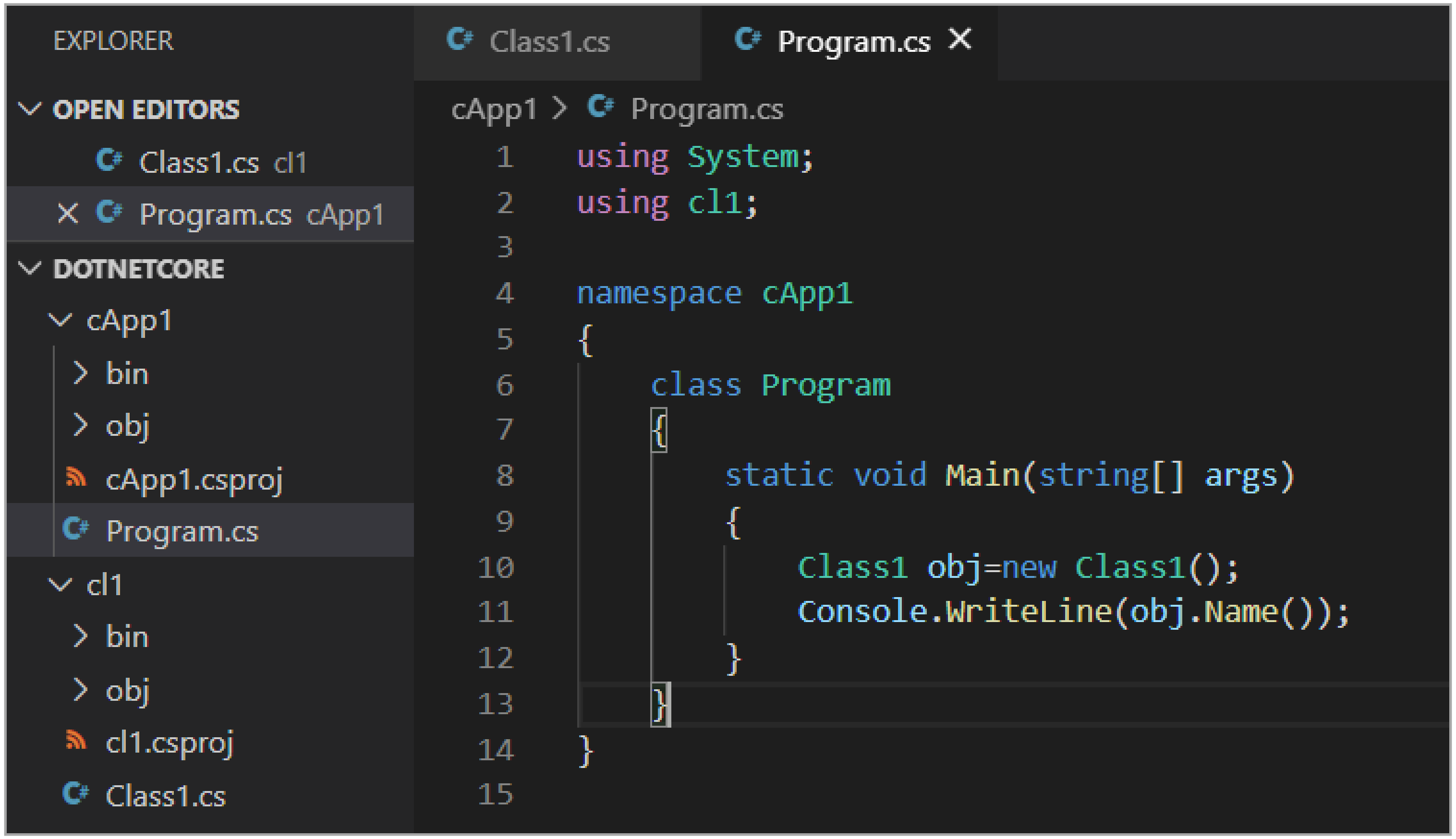
```
D:\DotNetCore>cd cApp1

D:\DotNetCore\cApp1>dotnet add reference ../cl1/cl1.csproj
Reference `..\cl1\cl1.csproj` added to the project.

D:\DotNetCore\cApp1>
```



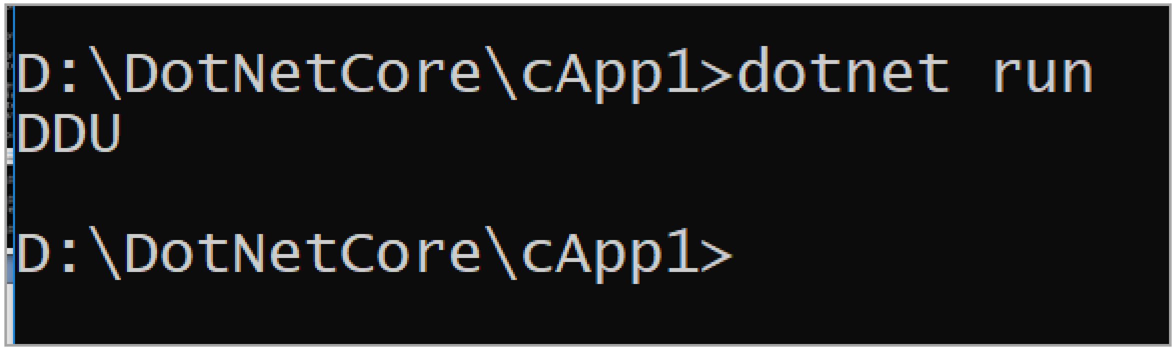
```xml
<Project Sdk="Microsoft.NET.Sdk">

  <ItemGroup>
    <ProjectReference Include="..\cl1\cl1.csproj" />
  </ItemGroup>

  <PropertyGroup>
    <OutputType>Exe</OutputType>
    <TargetFramework>netcoreapp3.1</TargetFramework>
  </PropertyGroup>

</Project>
```



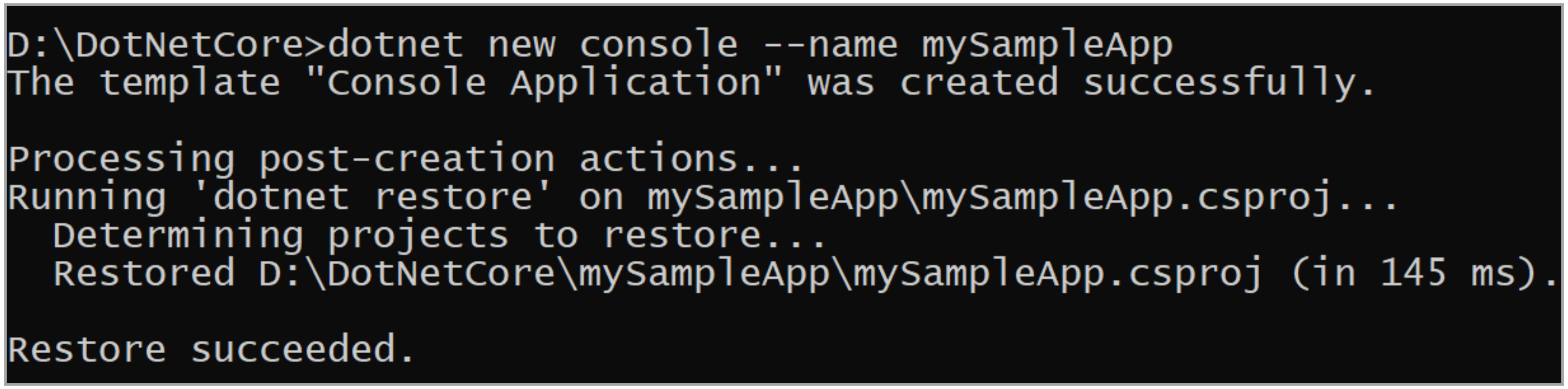
```csharp
using System;

namespace cl1
{
    public class Class1
    {
        public string Name()
        {
            return "DDU";
        }
    }
}

```



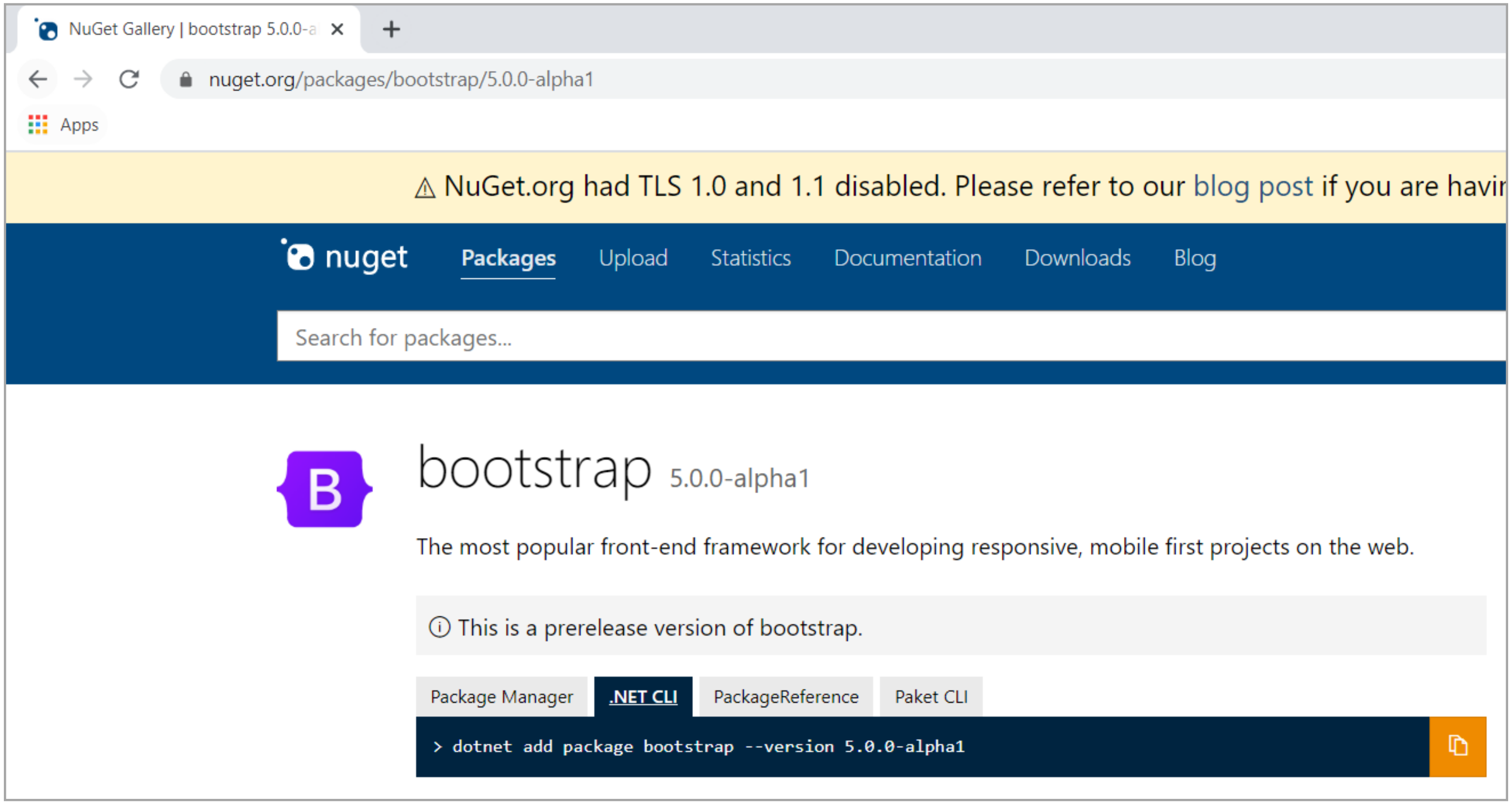
```csharp
using System;
using cl1;

namespace cApp1
{
    class Program
    {
        static void Main(string[] args)
        {
            Class1 obj=new Class1();
            Console.WriteLine(obj.Name());
        }
    }
}
```



```
D:\DotNetCore\cApp1>dotnet run
DDU

D:\DotNetCore\cApp1>
```



```
D:\DotNetCore>dotnet new console --name mySampleApp
The template "Console Application" was created successfully.

Processing post-creation actions...
Running 'dotnet restore' on mySampleApp\mySampleApp.csproj...
  Determining projects to restore...
  Restored D:\DotNetCore\mySampleApp\mySampleApp.csproj (in 145 ms).

Restore succeeded.
```



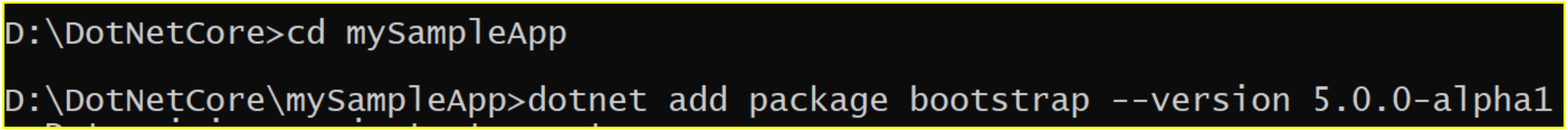
⚠ NuGet.org had TLS 1.0 and 1.1 disabled. Please refer to our blog post if you are havi

bootstrap 5.0.0-alpha1

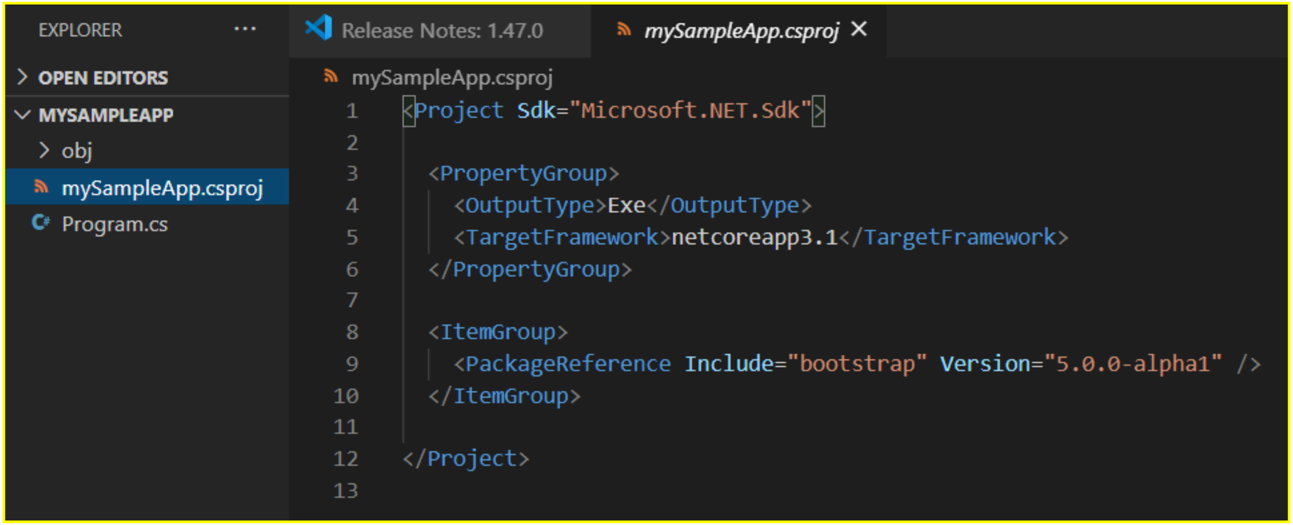
The most popular front-end framework for developing responsive, mobile first projects on the web.

ⓘ This is a prerelease version of bootstrap.

```
> dotnet add package bootstrap --version 5.0.0-alpha1
```



```
D:\DotNetCore>cd mySampleApp

D:\DotNetCore\mySampleApp>dotnet add package bootstrap --version 5.0.0-alpha1
```



```
<Project Sdk="Microsoft.NET.Sdk">

  <PropertyGroup>
    <OutputType>Exe</OutputType>
    <TargetFramework>netcoreapp3.1</TargetFramework>
  </PropertyGroup>

  <ItemGroup>
    <PackageReference Include="bootstrap" Version="5.0.0-alpha1" />
  </ItemGroup>

</Project>

```

.NET Core vs .NET Framework server apps

- There are two supported .NET implementations for building server-side apps
  - .NET Framework
  - .NET Core.
  - Both share many of the same components and you can share code across the two.
  - However, there are fundamental differences between the two and your choice depends on what you want to accomplish.

.NET Core$^{\text{When}}$

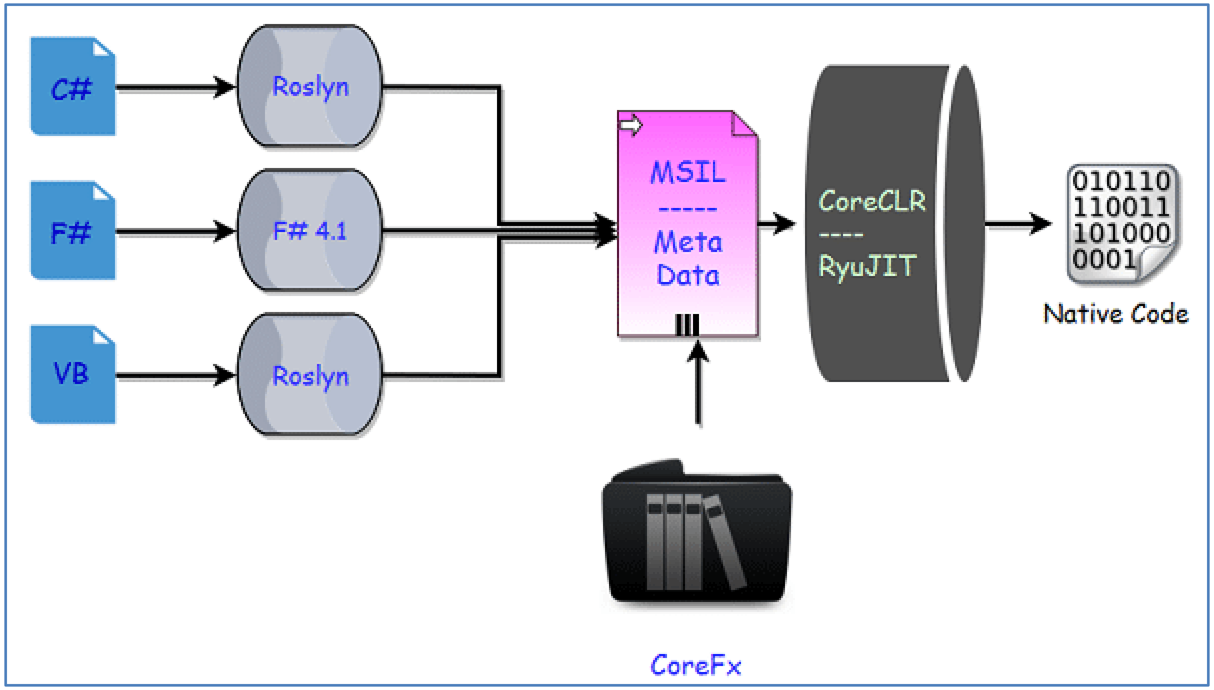
- You have cross-platform needs.
- You're targeting microservices.
- You're using Docker containers.
- You need high-performance and scalable systems.
- You need side-by-side .NET versions per application.

.NET Framework When

- Your app currently uses .NET Framework (recommendation is to extend instead of migrating).
- Your app uses third-party .NET libraries or NuGet packages not available for .NET Core.
- Your app uses .NET technologies that aren't available for .NET Core.
- Your app uses a platform that doesn’t support .NET Core.
  - Windows, macOS, and Linux support .NET Core.

What is the difference between SDK and Runtime in .NET Core?

What is CoreCLR?


C#
F#
VB
Roslyn
F# 4.1
Roslyn
MSIL

Meta
Data
CoreCLR

RyuJIT
010110
110011
101000
0001
Native Code
CoreFx