

```
public class HelloWorld {
    #include<iostream>
    using std::cout;
    using std::endl;
    public static void main(String args[]) {
        System.out.println("Hello World!");
    }
}
DSEG SEGMENT
STR DB "Hello World!"
DSEG ENDS
CSEG SEGMENT
ASSUME DS:DSEG,CSEG
START: MOV AX,DSEG
MOV DS,AX
MOV DX,OFFSET STR
MOV AH,09H
INT 21H
MOV AH,4CH
INT 21H
CSEG ENDS
END START
```

Hello World!

```
#include<iostream>
using std::cout;
using std::endl;
void main() {
    cout<<"Hello World!"<<endl;
    return 0;
}
printf("Hello World!");
using System.Collections.Generic;
using System.Linq;
using System.Text;
namespace hello
{
    class Program
    {
        static void Main(string[] arg
        {
            Console.WriteLine("Hello
        }
    }
}
```

.NET Core 的 Hello World 之旅

# 1 .NET core项目的创建与编译

北京理工大学计算机学院  
金旭亮

# 创建项目文件夹



打开命令提示符窗口，创建一个空的文件夹.....

```
命令提示符

D:\>md myApp

D:\>cd myApp

D:\myApp>dir
驱动器 D 中的卷是 Working
卷的序列号是 FA4A-1772

D:\myApp 的目录

2019/07/19  08:39    <DIR>          .
2019/07/19  08:39    <DIR>          ..
                0 个文件              0 字节
                2 个目录 119,123,361,792 可用字节

D:\myApp>_
```

# 创建项目



使用“**dotnet new**”命令创建一个.NET core控制台项目：

```
命令提示符
D:\myApp>dotnet new console
The template "Console Application" was created successfully.

Processing post-creation actions...
Running 'dotnet restore' on D:\myApp\myApp.csproj...
  D:\myApp\myApp.csproj 的还原在 75.83 ms 内完成。

Restore succeeded.

D:\myApp>
```



网络下载组件的默认存储文件夹：C:\Users\用户名\.nuget\packages

# .NET core 命令行 (CLI) 工具命令的格式

要执行的命令

`dotnet new console`

“the driver”  
命令行工具中的“驱动”程序

传给要执行命令的参数

执行“**dotnet --help**”，  
可以查看可用的 .NET  
Core CLI 内置命令的清单。

# 查看可用的 .NET Core 项目模板清单

项目

文件

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#], VB	Common/WPF
Windows Forms (WinForms) Application	winforms	[C#], VB	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 Component	razorcomponent	[C#]	Web/ASP.NET
Razor Page	page	[C#]	Web/ASP.NET
MVC ViewImports	viewimports	[C#]	Web/ASP.NET
MVC ViewStart	viewstart	[C#]	Web/ASP.NET
Blazor (server-side)	blazorserverside	[C#]	Web/Blazor
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 API	webapi	[C#], F#	Web/WebAPI
ASP.NET Core gRPC Service	grpc	[C#]	Web/gRPC
dotnet gitignore file	gitignore		Config
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
Protocol Buffer File	proto		Web/gRPC

**dotnet new --help**

# 多了解一点.NET Core项目的构建知识



(共享的) .NET Core (CLI) SDK 组件



MSBuild



类似于

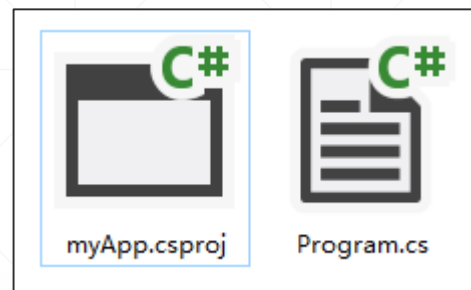
# dotnet new 命令生成的控制台项目文件

```
命令提示符
D:\myApp>dir
驱动器 D 中的卷是 Working
卷的序列号是 FA4A-1772

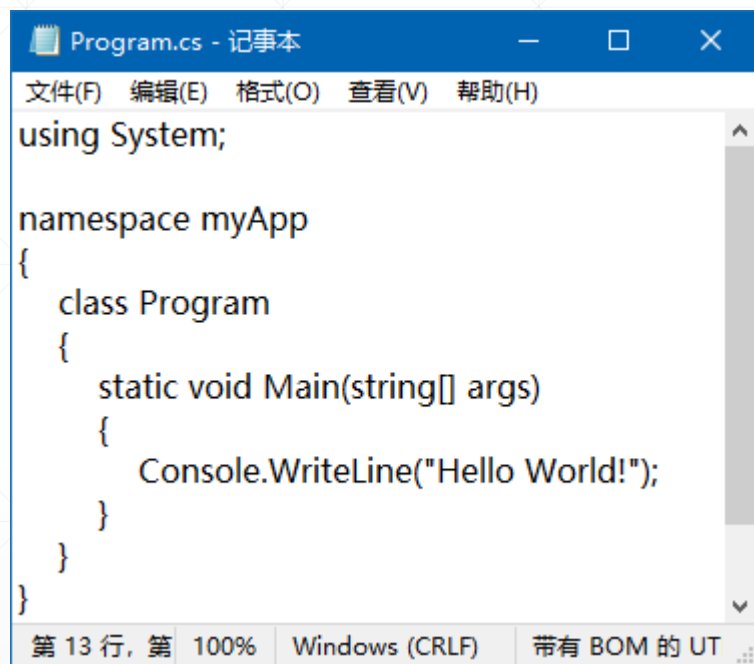
D:\myApp 的目录

2019/07/19  08:48    <DIR>          .
2019/07/19  08:48    <DIR>          ..
2019/07/19  08:48             178 myApp.csproj
2019/07/19  08:48    <DIR>          obj
2019/07/19  08:48             187 Program.cs
                2 个文件             365 字节
                3 个目录 119,123,001,344 可用字节

D:\myApp>
```



# dotnet new console创建的两个重要文件



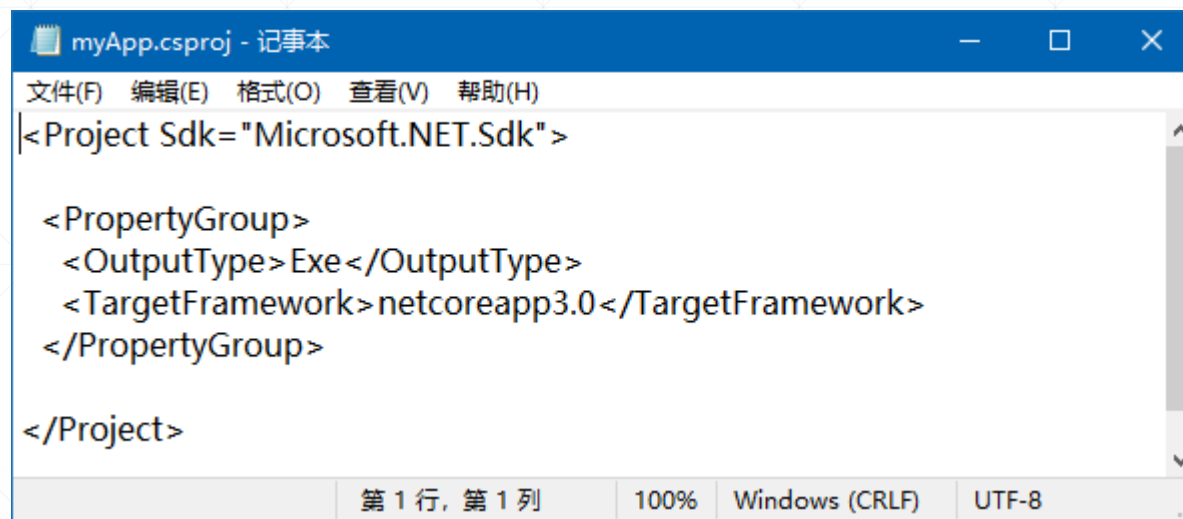
```
Program.cs - 记事本
文件(F) 编辑(E) 格式(O) 查看(V) 帮助(H)
using System;

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

第 13 行, 第 100% Windows (CRLF) 带有 BOM 的 UT



Program.cs  
(程序入口点)



```
myApp.csproj - 记事本
文件(F) 编辑(E) 格式(O) 查看(V) 帮助(H)
<Project Sdk="Microsoft.NET.Sdk">

  <PropertyGroup>
    <OutputType>Exe</OutputType>
    <TargetFramework>netcoreapp3.0</TargetFramework>
  </PropertyGroup>

</Project>
```

第 1 行, 第 1 列 100% Windows (CRLF) UTF-8

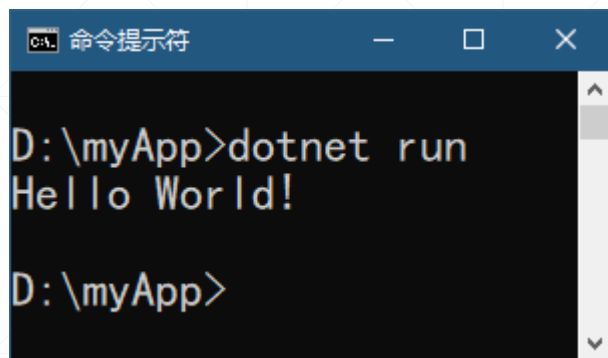


myApp.csproj  
(包容构建项目所需的信息)



# 编译并运行项目

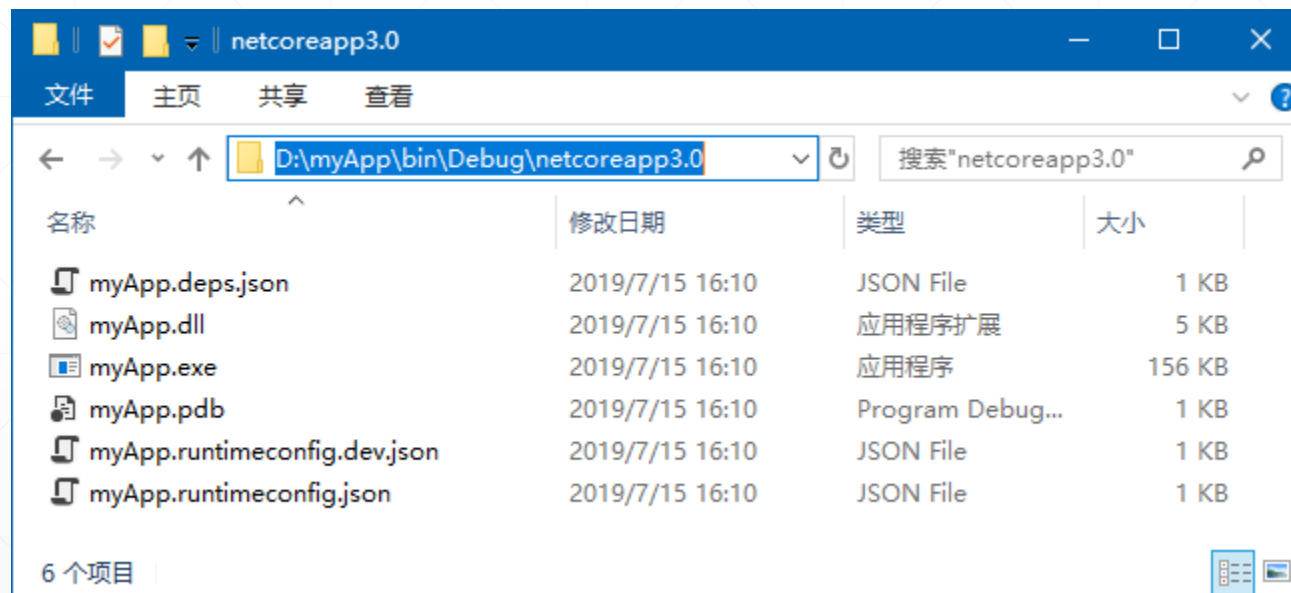
输入 “**dotnet run**” 编译并运行



```
C:\> 命令提示符

D:\myApp>dotnet run
Hello World!

D:\myApp>
```



编译生成的结果

# 执行.NET Core程序的另外两种方式

```
命令提示符
D:\myApp\bin\Debug\netcoreapp3.0>dir
驱动器 D 中的卷是 Working
卷的序列号是 FA4A-1772

D:\myApp\bin\Debug\netcoreapp3.0 的目录

2019/07/19  10:27    <DIR>          .
2019/07/19  10:27    <DIR>          ..
2019/07/19  10:27             407 myApp.deps.json
2019/07/19  10:27          4,608 myApp.dll
2019/07/19  10:27        159,744 myApp.exe
2019/07/19  10:27          408 myApp.pdb
2019/07/19  10:27          286 myApp.runtimeconfig.dev.json
2019/07/19  10:27          172 myApp.runtimeconfig.json
                6 个文件          165,625 字节
                2 个目录 119,122,100,224 可用字节

D:\myApp\bin\Debug\netcoreapp3.0>dotnet myApp.dll
Hello World!

D:\myApp\bin\Debug\netcoreapp3.0>myApp
Hello World!

D:\myApp\bin\Debug\netcoreapp3.0>_
```

先进入到可执行程序所在的文件夹，之后：



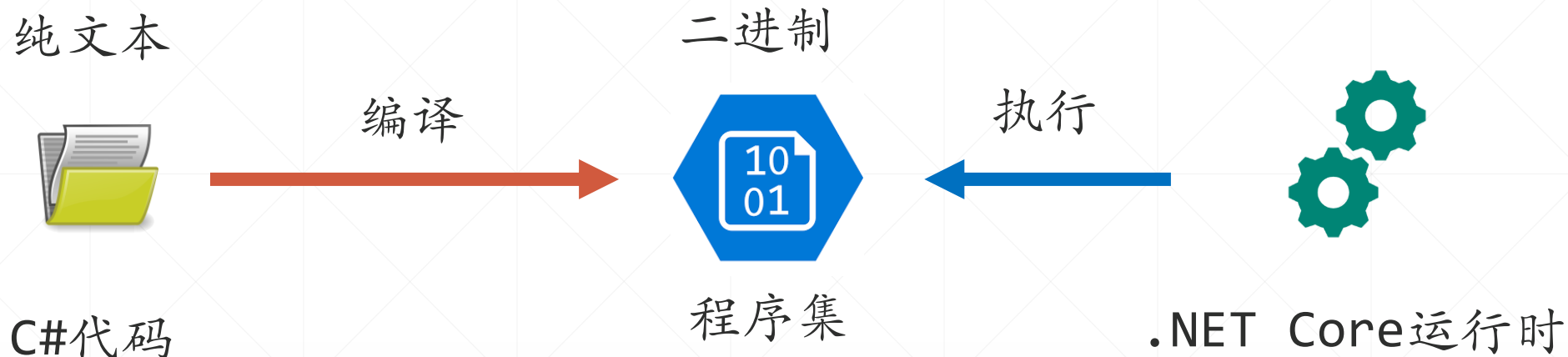
dotnet dll文件名

-----



exe文件名

# 发生了什么？



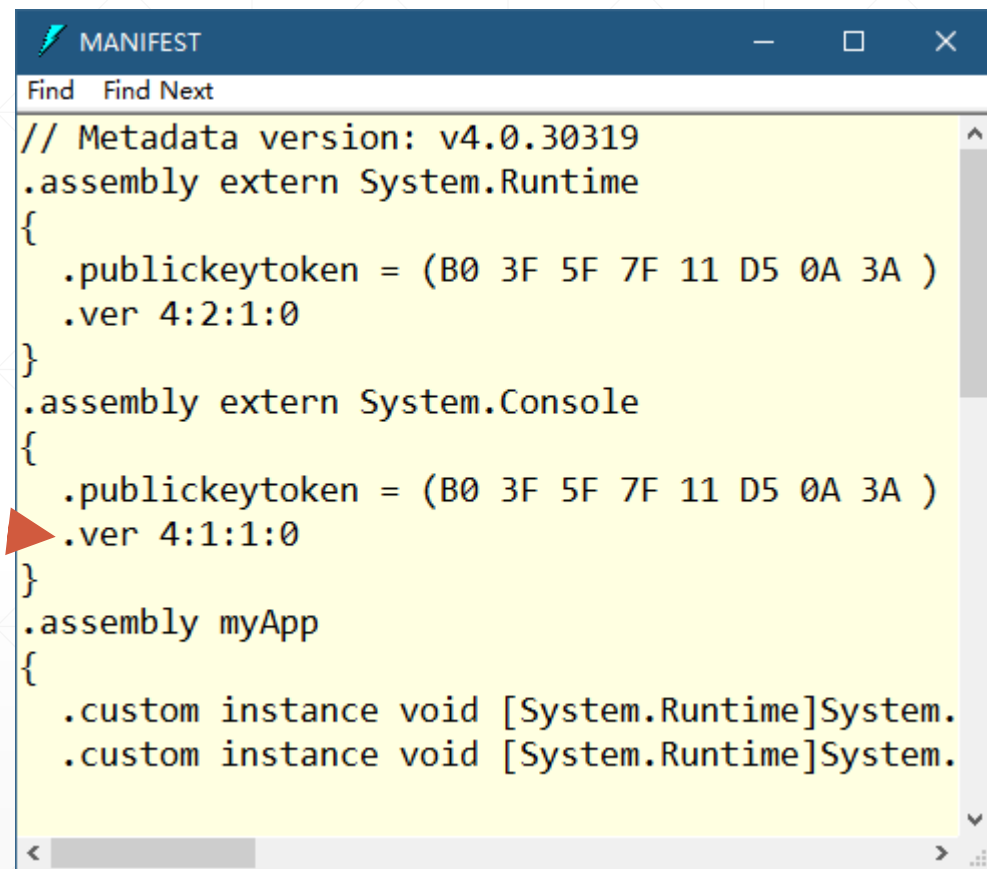
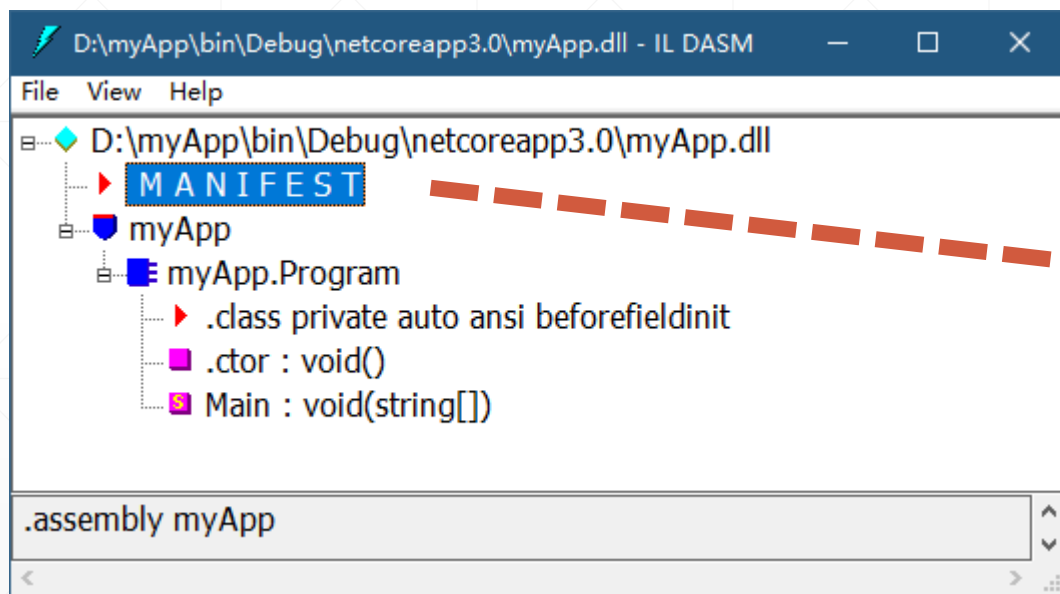
示例编译生成的dll/exe文件，称为“**程序集(Assembly)**”，程序集是构建.NET Core应用的基本构造块，也是部署的最小单元。



可执行程序（exe）文件只能在Windows平台上运行，而dll文件，则具有跨平台特性。

# 程序集中有什么？

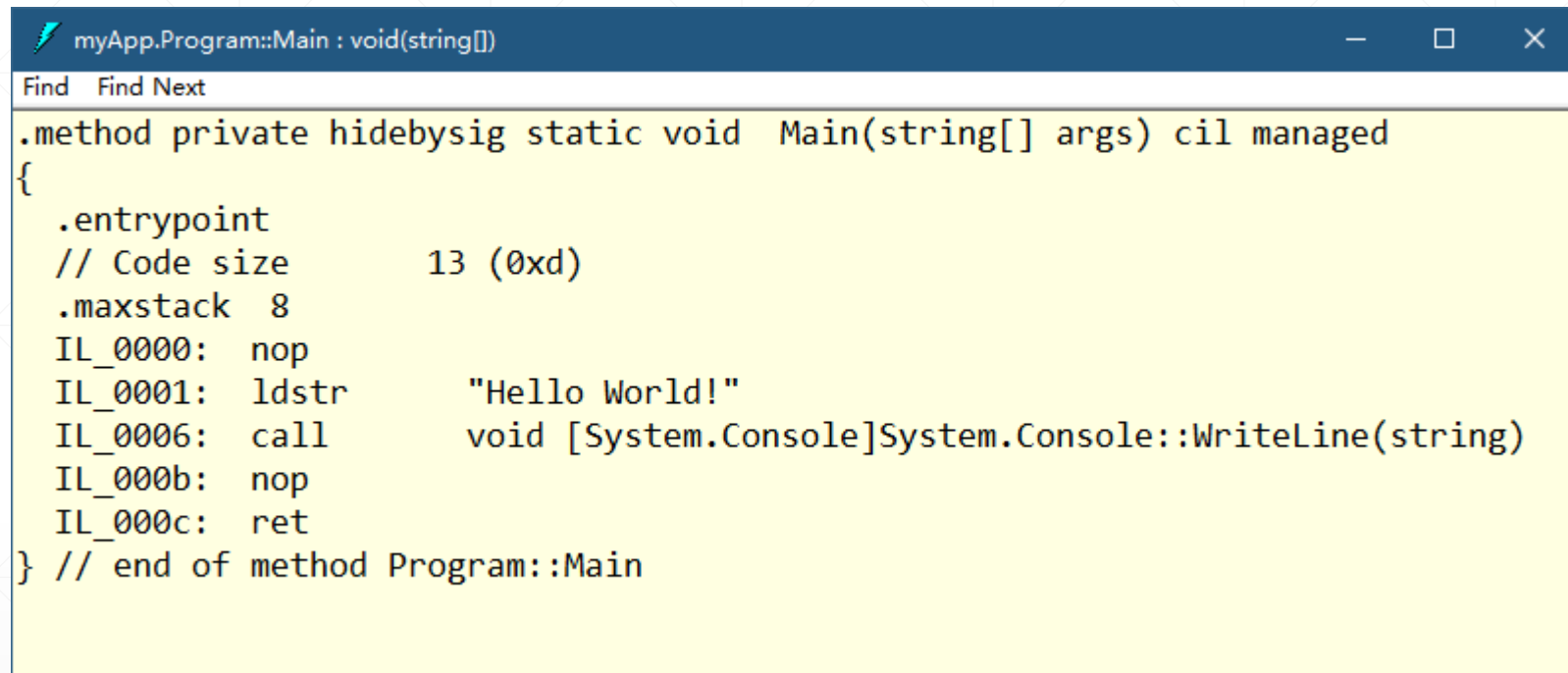
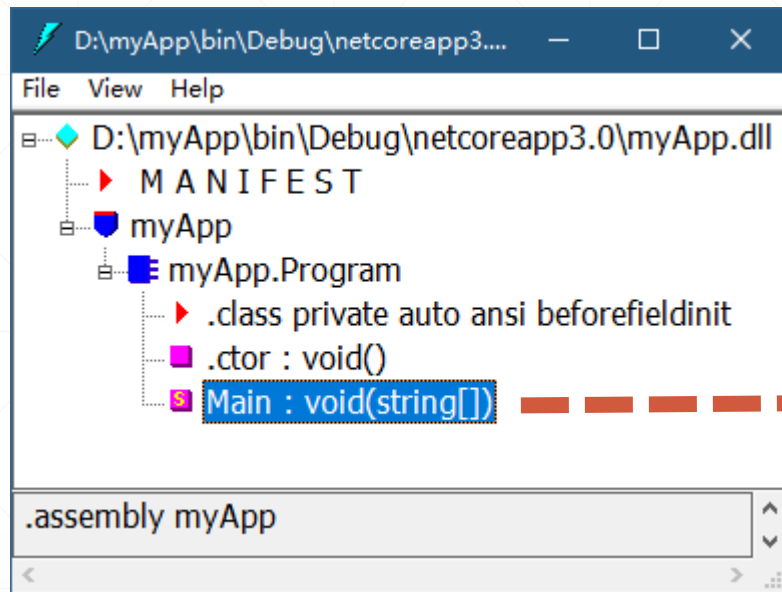
使用 .NET Framework 所提供的 ildasm 工具，可以查看程序集中的内容：



C:\Program Files (x86)\Microsoft  
SDKs\Windows\v10.0A\bin\NETFX 4.7.2 Tools

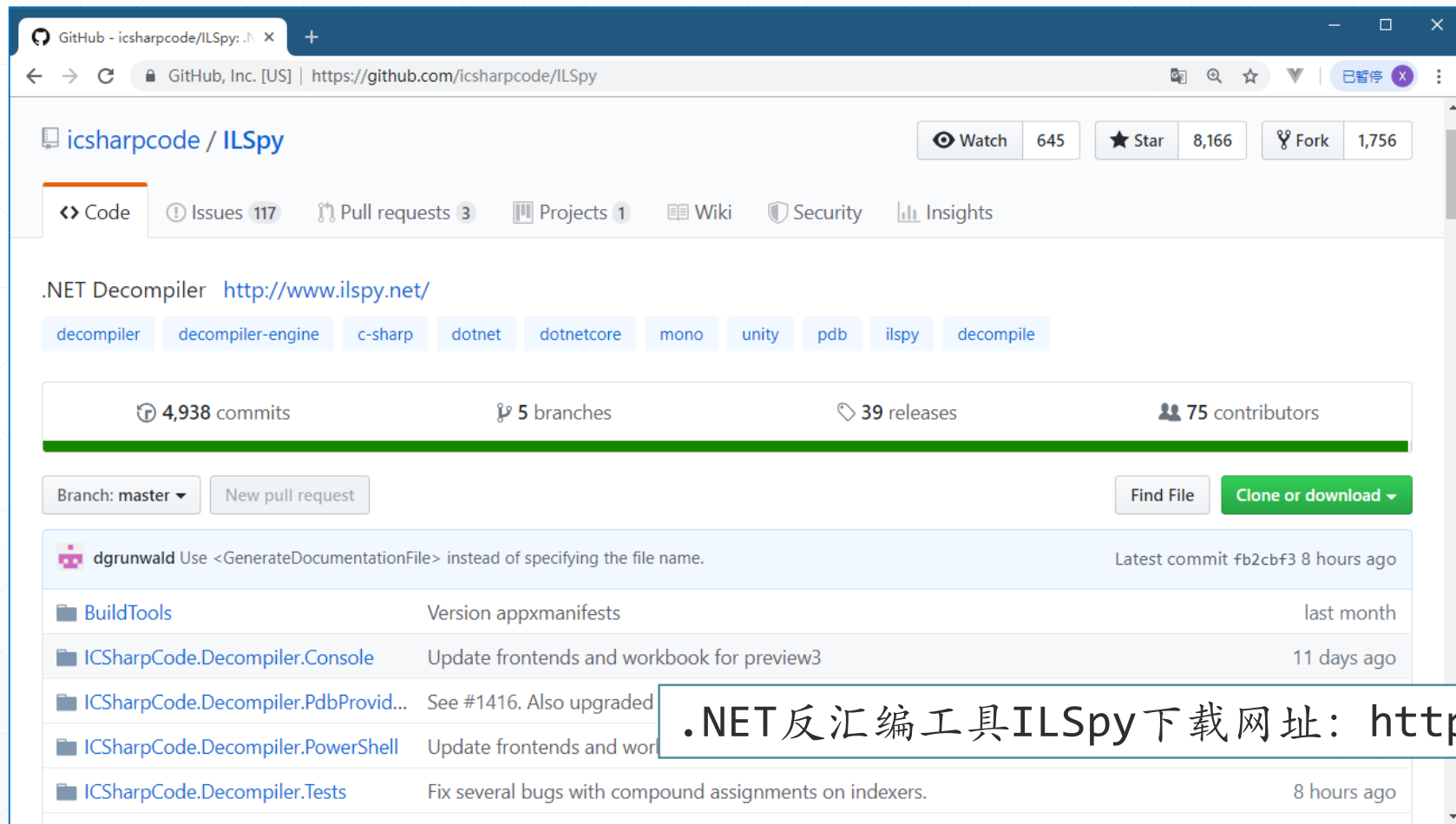
清单 (Manifest) 描述了程序集中  
包容有哪些内容，被称为是程序集  
的“元数据 (Metadata)”。

# C#代码被转换为IL指令



C#写的Program.Main方法被转换为使用“**IL (Intermediate language, 中间语言)**”指令代码编写的“汇编”程序。

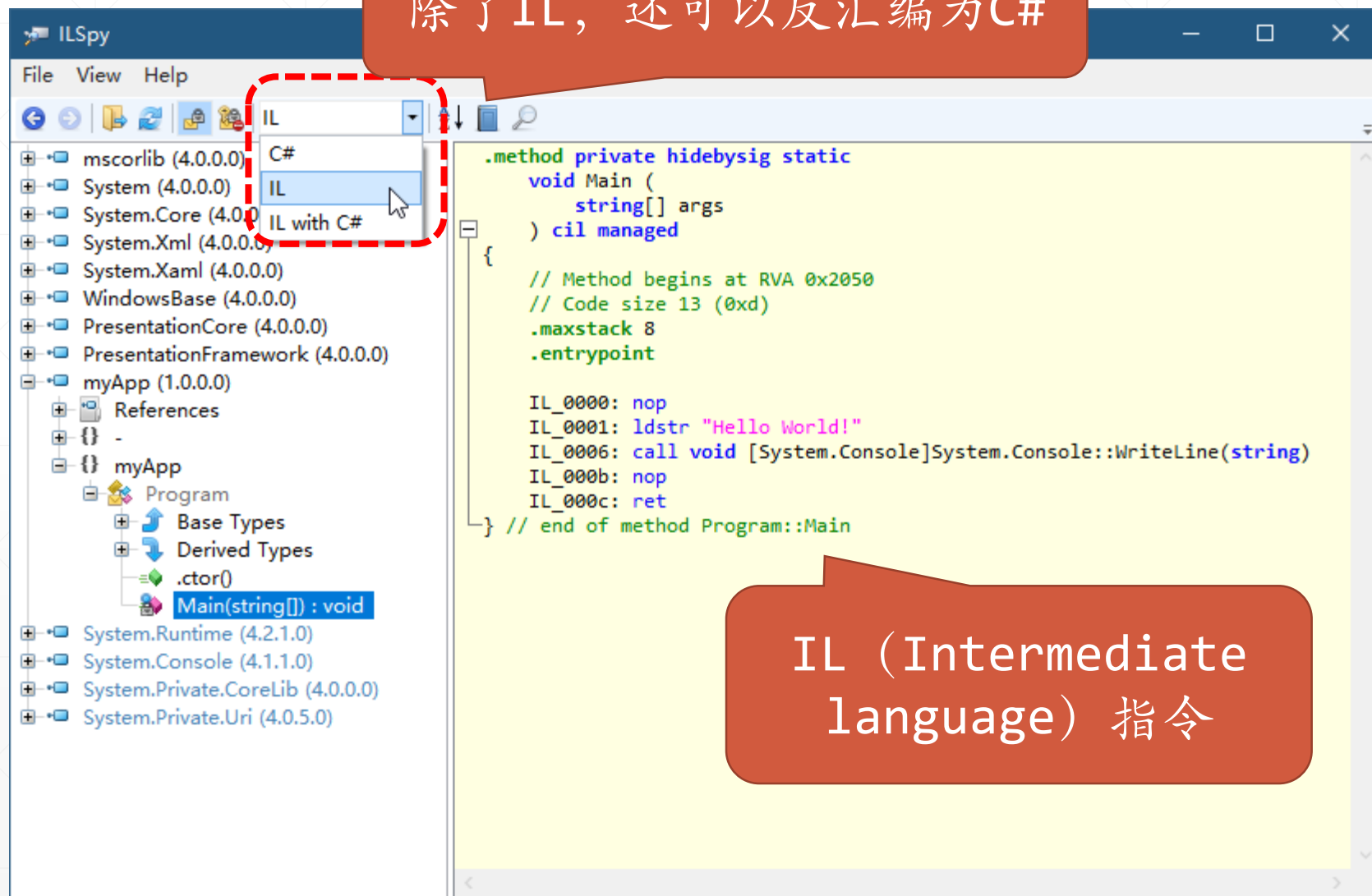
# 另一个强大的程序集探查工具



.NET反汇编工具ILSpy下载网址: <https://ilspy.net/>

# 使用ILSpy反汇编.NET Core程序集

除了IL，还可以反汇编为C#



IL (Intermediate language) 指令

## 小结：程序集中有什么？

