# gRPC介绍

## gRPC是什么？

现在系统都服务化，.net core 实现服务化的方式有很多，我们通过grpc实现客户端、服务端通信。

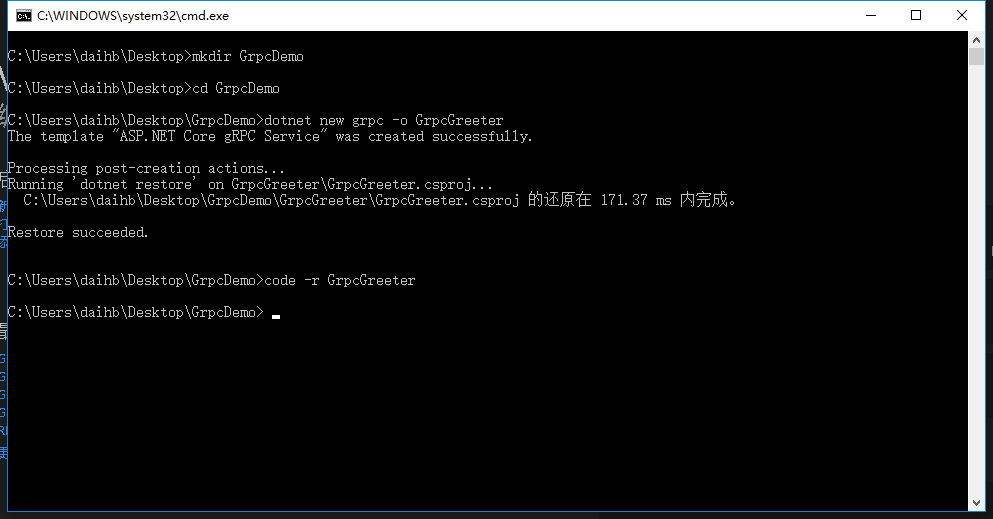
grpc(https://grpc.io/)是google发布的一个开源、高性能、通用RPC（Remote Procedure Call）框架，使用HTTP/2协议，支持多路复用，并用ProtoBuf作为序列化工具，提供跨语言、跨平台支持。下面以.net core3版本演示如何使用grpc框架实现通信。

## 2.开发gRPC

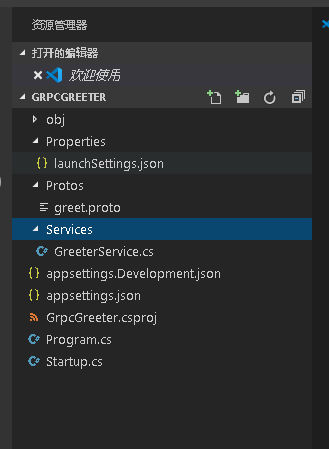
下面演示一下如何使用grpc框架，由于是基于3.0的需要使用vs2019才能进行开发，由于我没有装vs2019，下面就用vs code来进行开发吧，可能会比较麻烦。

1. 先使用命令行在目录GrpcDemo创建一个名为GrpcGreeter的.net core 3的grpc模板项目。

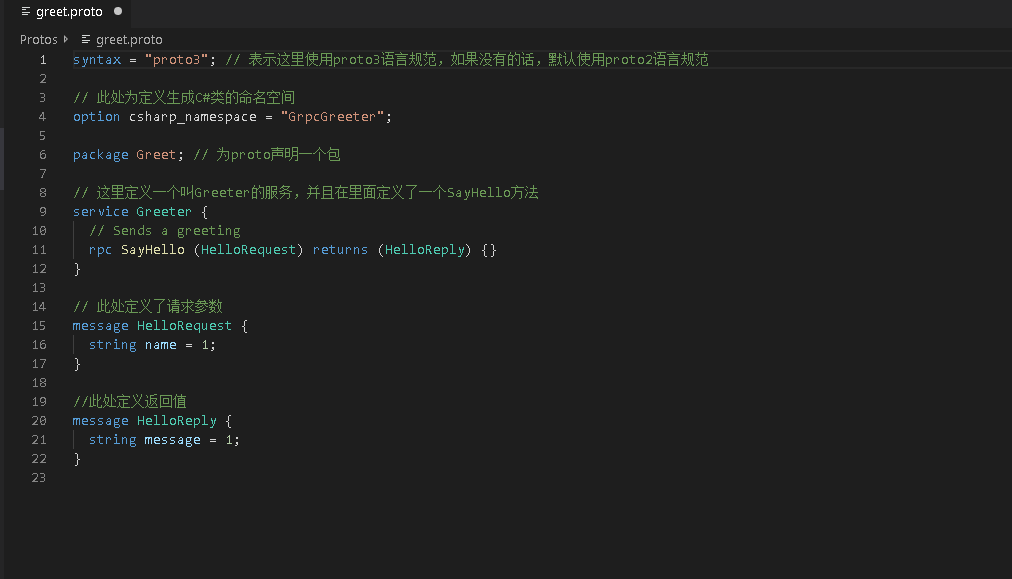
并使用code -r GrpcGreeter命令来启用vscode打开这个项目。



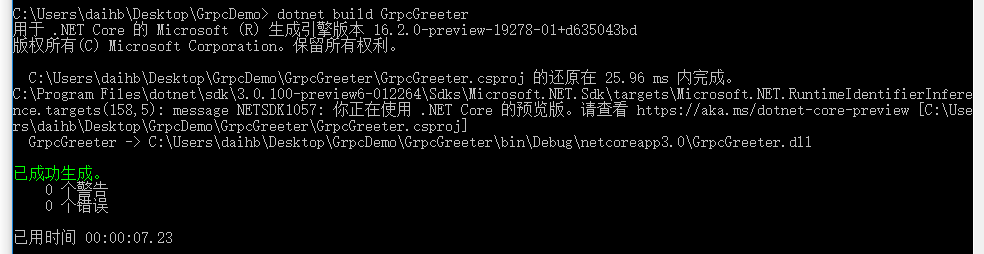
项目的目录结构：



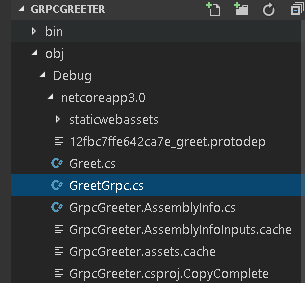
Protos目录下的greet.proto文件为我们的grpc服务定义。现在打开greet.proto文件，在其中编写基于proto3语言的协议代码，用于自动生成到各语言协议，如果需要更深入的学习proto3语言可以打开该网站[Proto3语言指南](https://developers.google.com/protocol-buffers/docs/proto3)。greet.proto 代码如下：



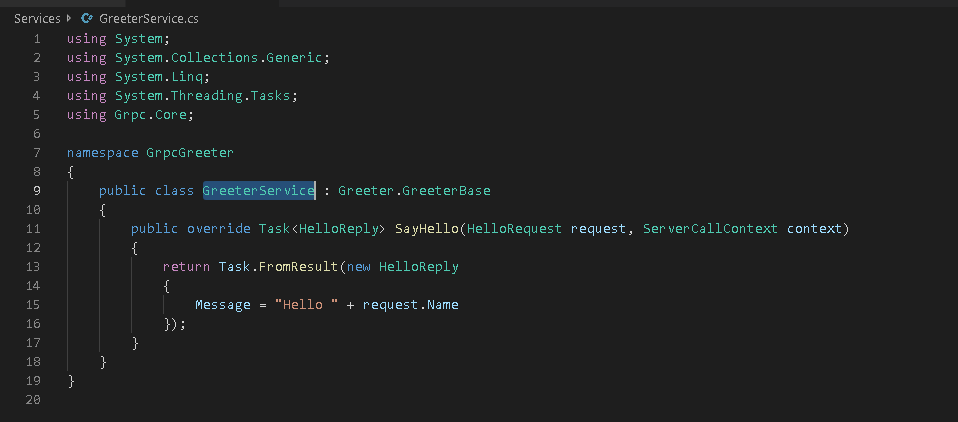
现在我们用dotnet buildGrpcGreeter编译一下项目，然后看下会发生什么。



可以看到obj/Debug/netcoreapp3.0目录生成了Greet.cs和GreetGrpc.cs两个类文件，这就是根据上面定义的proto文件生成的类文件。

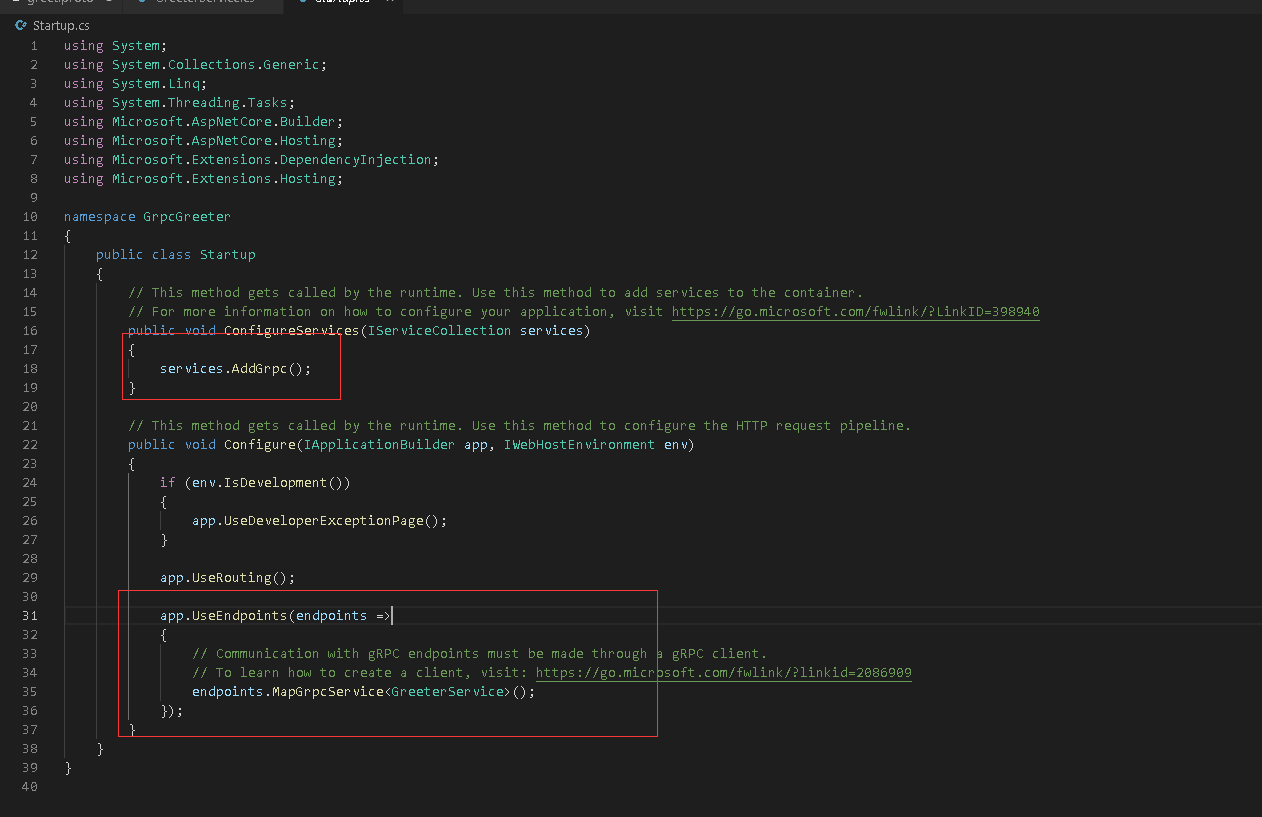


再看下services/GreeterService.cs的文件，这里就实现了grpc服务。



可以看到这个服务继承了上面编译项目所生成的类，然后实现了之前在proto文件里定义的方法。并且返回了向客户端打招呼的消息。

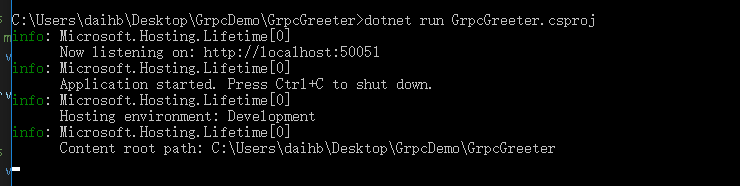
下面我们再看下StratUp文件，重点看红线标出的部分。



services.AddGrpc();方法则添加了Grpc服务的依赖注入。

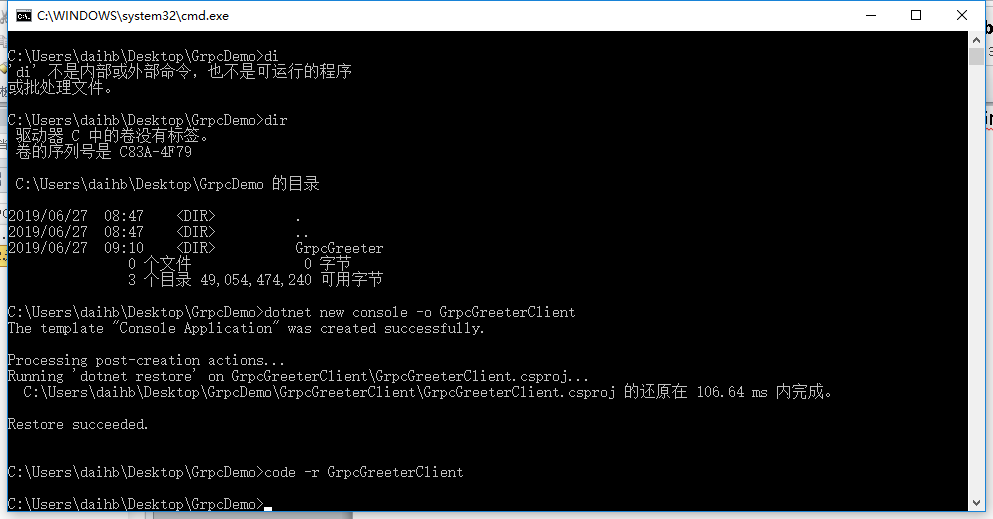
然后使用app.UseEndpoints方法注册服务，在lambda表达式使用IEndpointRouteBuilder的MapGrpcService方法来注册了上面所写的GreeterService服务。

现在让我们使用dotnet run GrpcGreeter命令来运行一下项目：



可以看到已经成功运行了项目，并监听了本机的50051端口。

1. 创建客户端项目来调用grpc服务。使用dotnet new console -o GrpcGreeterClient命令创建项目，并使用code -r GrpcGreeterClient启用vscode打开项目。

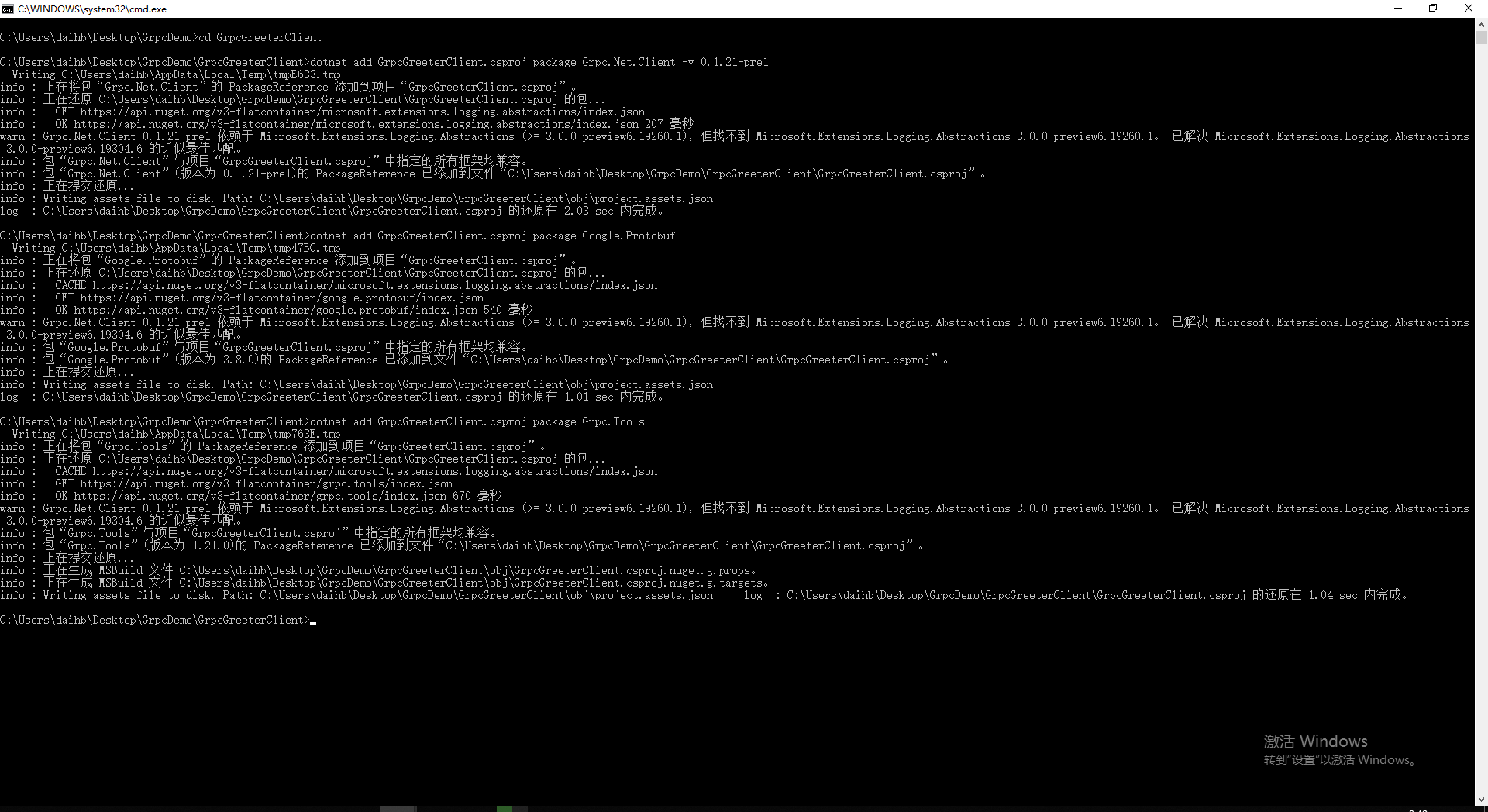


安装一些grpc的依赖包，使用下面的命令到项目根目录下安装依赖包。其中Grpc.Net.Client需特别指定一下版本。

dotnet add GrpcGreeterClient.csproj package Grpc.Net.Client –version 0.1.21-pre1

dotnet add GrpcGreeterClient.csproj package Google.Protobuf

dotnet add GrpcGreeterClient.csproj package Grpc.Tools



可以看到，现在已经成功的安装包了。

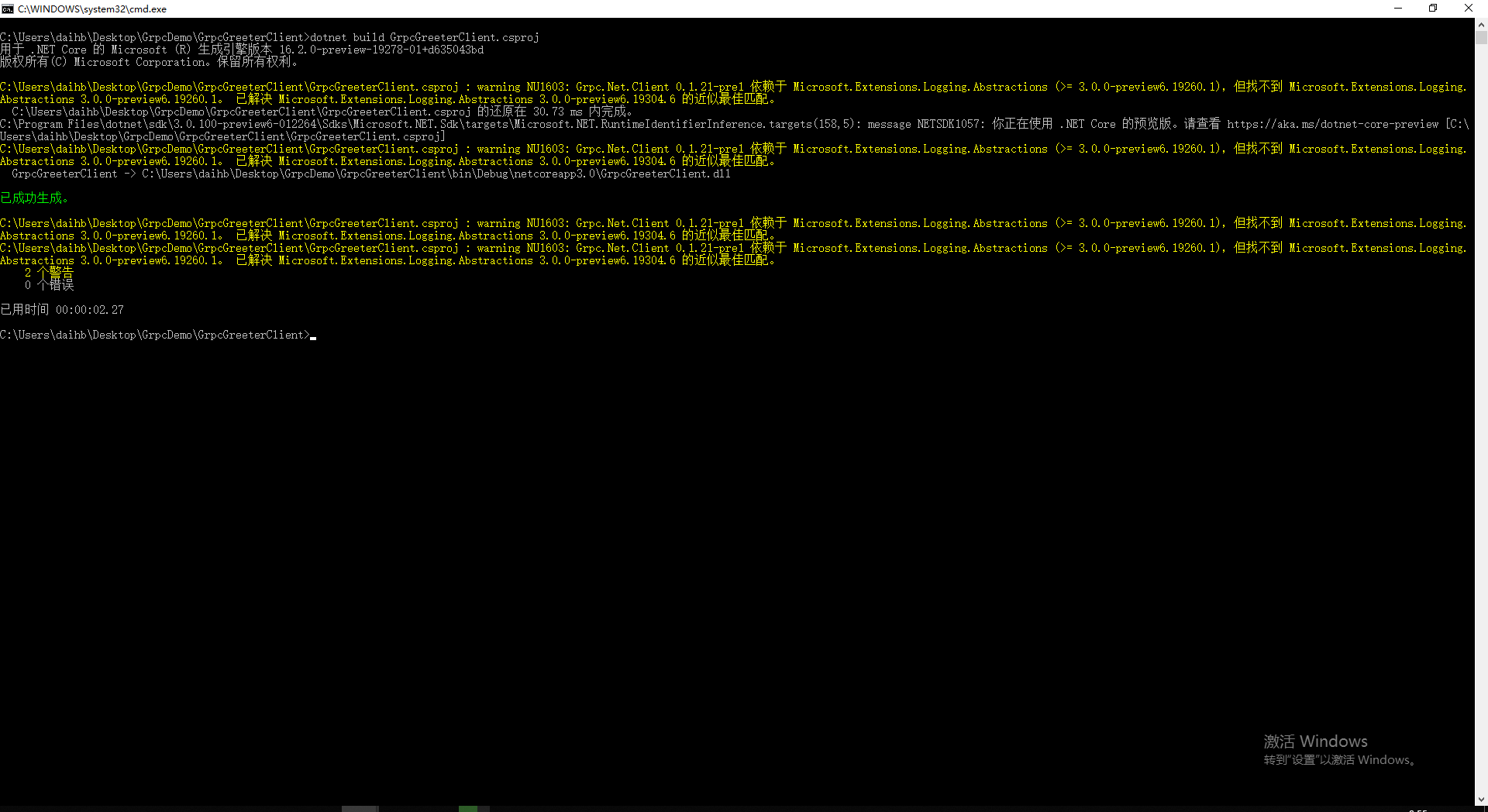
现在把GrpcGreeter项目的protos文件夹复制到现在的项目文件夹。并在GrpcGreeterClient.csproj 文件的Project节点下添加以下内容来包括文件夹到项目中。

<ItemGroup>

<Protobuf Include="Protos\greet.proto" GrpcServices="Client" />

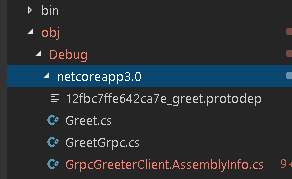
</ItemGroup>

随后，我们编译一下项目

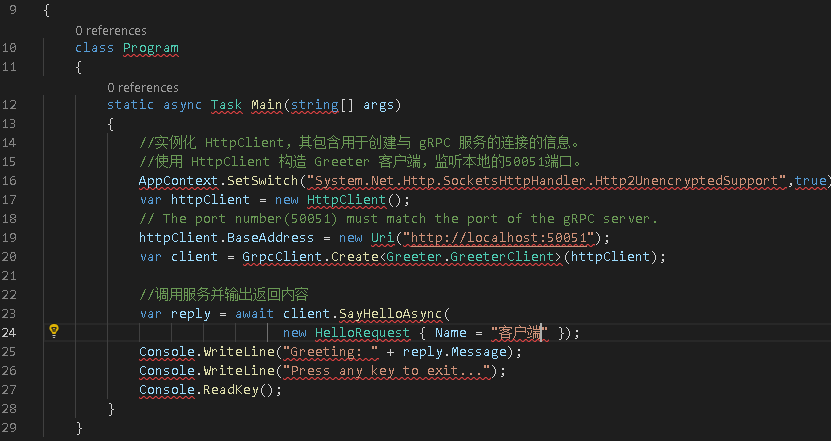


可以看到编译成功了，但是产生警告了，这些警告不会有影响，先不予理会。

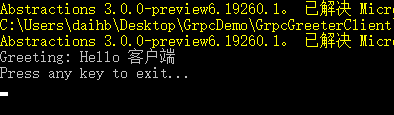
然后可以看到obj\Debug\netcoreapp3.0目录下有生成Greet.cs和GreetGrpc两个类文件。



下面我们可以编写代码进行调用服务端的服务了，在Program的Main方法里进行调用。由于是vscode，不太好用，代码提示有些问题。



下面可以运行一下项目，看下能否成功调用服务。



可以看到已经成功运行，并输出返回内容了，它已经成功的向我们打招呼了。

这是第二种调用方式，相比第一种较为简便，可以达到同样的效果。

