Atitit netcore vscode 环境搭建

搜索的时候，推荐使用 OmniSharp 关键字，因为这可以得到唯一的结果，你不会弄混淆。如果你使用 C# 作为关键字，那需要小心，你得找到名字只有 C#，点开之后是 C# for Visual Studio Code 的那款插件。因为可能装错，所以我不推荐这么做。

对于新版的 Visual Studio Code，装完会自动启用，所以你不用担心。我们可以后续步骤了。

准备一个空的文件夹，这个文件夹将会成为我们解决方案所在的文件夹，也就是 sln 文件所在的文件夹。在这个空的文件夹中打开 VSCode，然后打开 VSCode 的终端。

在 VSCode 中的终端中输入：

> dotnet new console -o Walterlv.Demo

这样会在当前的文件夹中创建一个 Walterlv.Demo 的子文件夹，并且在此文件夹中新建一个名为 Walterlv.Demo 的控制台项目

如果你观察我们刚刚创建的项目，你会发现里面有一个 csproj 文件和一个 Program.cs 文件。csproj 文件是 Sdk 风格的项目文件，而 Program.cs 里面包含最简单的 Hello World 代码：

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

## 运行与调试成调试所需的 launch.json 和 tasks.json 文件：

理论上，你按下 F5，选择 .NET Core 后就能自动生成调试所需的 launch.json 和 tasks.json 文件：

让你的 VSCode 具备调试 C# 语言 .NET Core 程序的能力

如果不能生成所需的文件，你可以使用以下博客中的方法，手动添加这两个文件：

手工编辑 tasks.json 和 launch.json，让你的 VSCode 具备调试 .NET Core 程序的能力

# 问题解决

Vscode 找不到环境变量xx.exe解释器

vscode 启动的时候，只读取一次环境变量。

你在启动vscode之后加的Path,  vscode不知道。所以重启vscode就可以

## 请指定项目或解决方案文件。当前工作目录中未包含项目或解决方案文件。

可以命令行生产csprj文件，然后复制过来dotnet new console -o Walterlv.Demo

### **Visual Studio Code中vscode-solution-explorer解决方案管理器插件的使用**

可能很多.neter朋友们刚开始使用Visual Studio Code的时候很不适应各种命令行dotnet命令来创建项目以及解决方案。幸运的是，Visual Studio Code扩展中提供了类似于Visual Studio的解决防范资源管理的插件来解决这个问题。下面我们一步一步的看下如何使用此插件吧！

打开Visual Studio Code扩展，然后输入vscode-solution-explorer，然后如下图所示进行安装。

(9+条消息)使用 dotnet 命令行配合 vscode 完成一个完整 .NET 解决方案的编写和调试 - dotNET跨平台 - CSDN博客.mhtml

(9+条消息)使用 dotnet 命令行配合 vscode 完成一个完整 .NET 解决方案的编写和调试 - dotNET跨平台 - CSDN博客