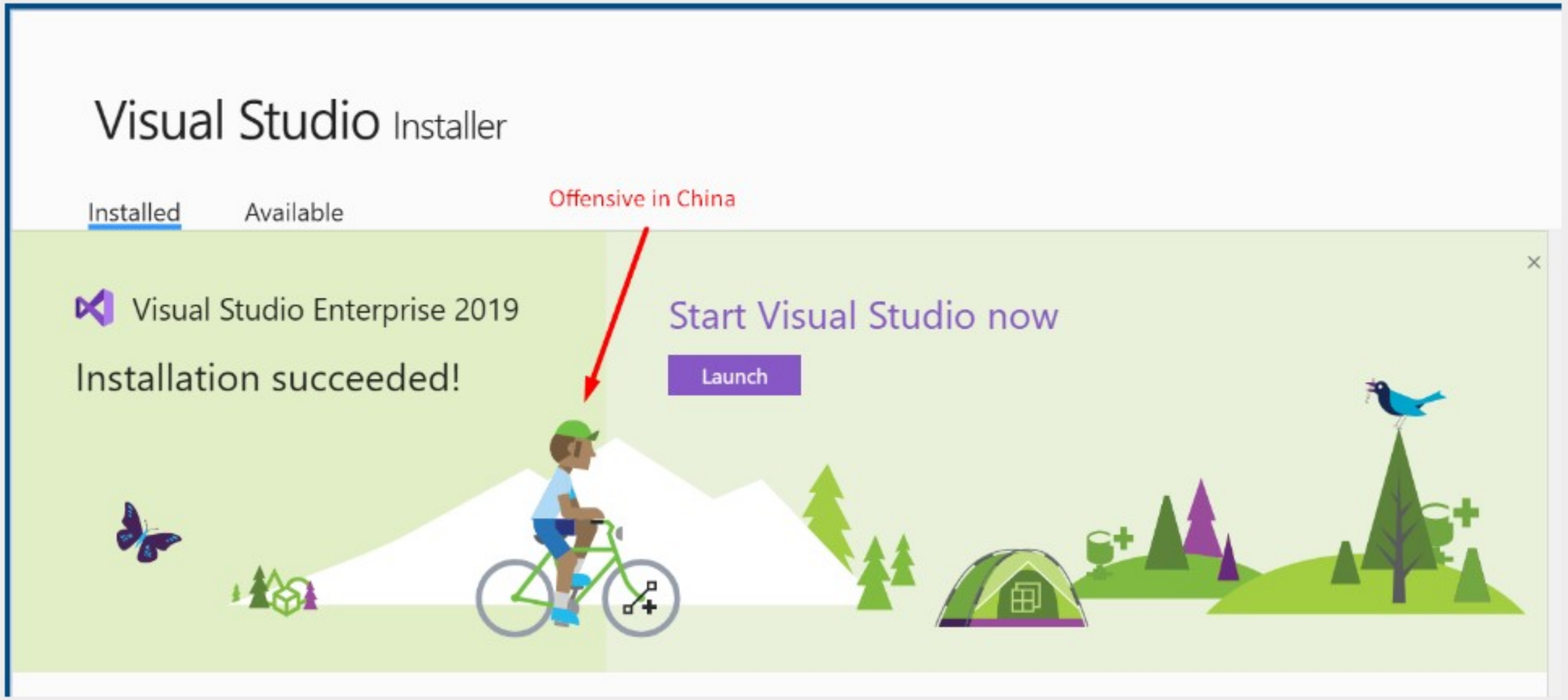
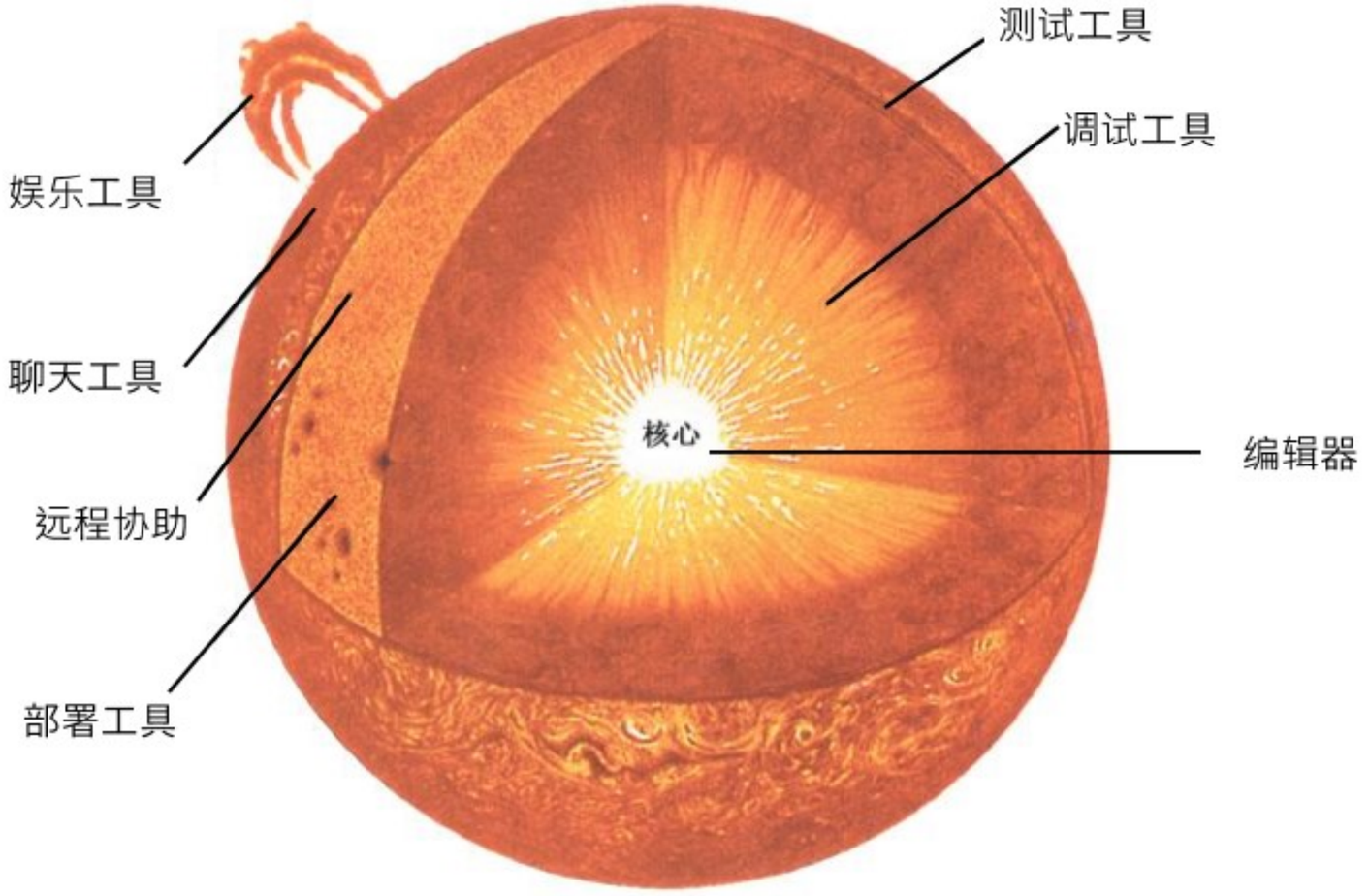
VisualStudio 2019 和 C# 8.0

林德熙

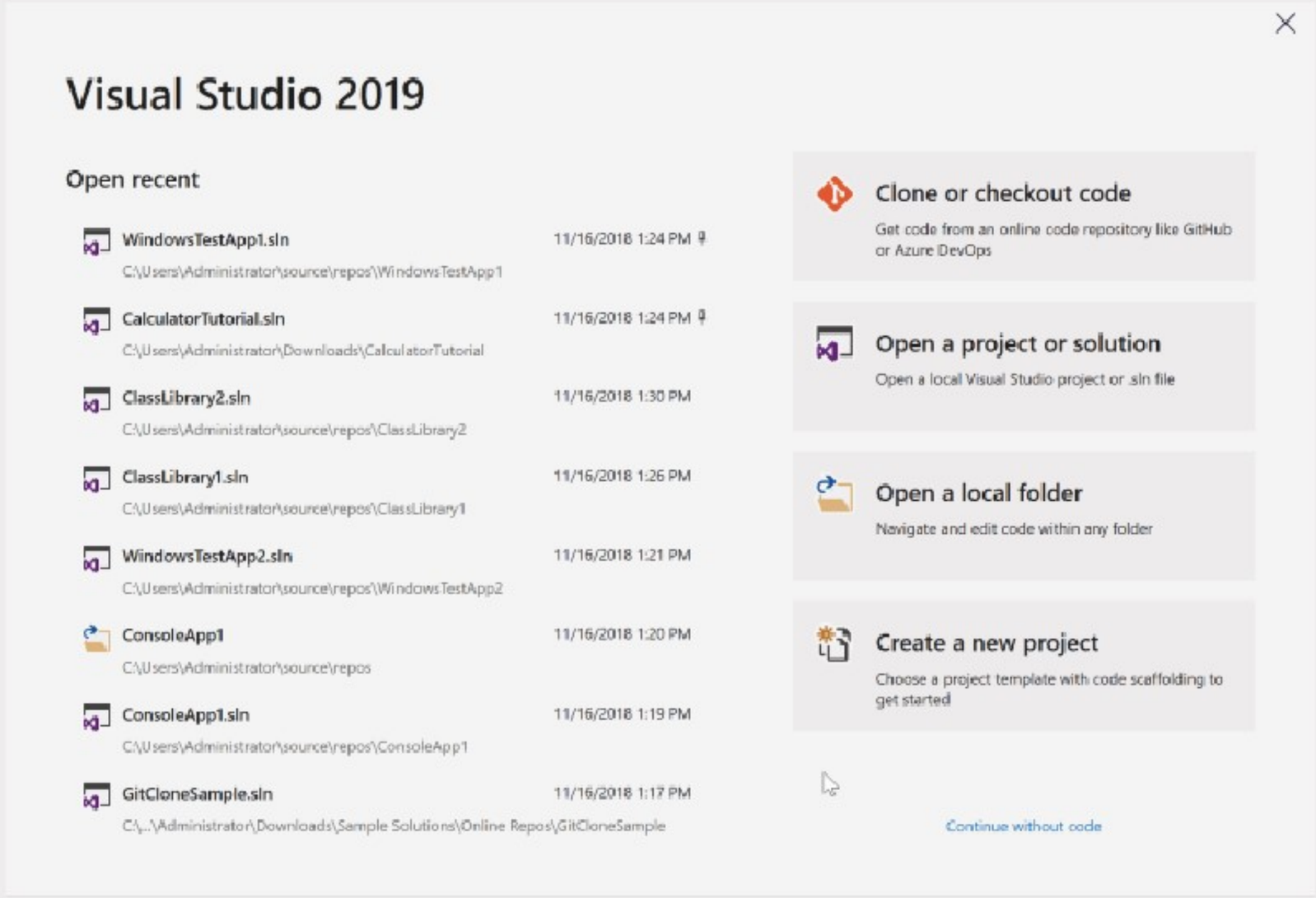

Visual Studio Installer
Installed
Available
Offensive in China
Visual Studio Enterprise 2019
Installation succeeded!
Start Visual Studio now
Launch


测试工具
调试工具
娱乐工具
聊天工具
核心
编辑器
远程协助
部署工具

CVTE
Dream·Future

打开 VisualStudio 玩一下 

全新欢迎界面


Visual Studio 2019
Open recent
WindowsTestApp1.sln
11/16/2018 1:24 PM
C:\Users\Administrator\source\repos\WindowsTestApp1
CalculatorTutorial.sln
11/16/2018 1:24 PM
C:\Users\Administrator\Downloads\CalculatorTutorial
ClassLibrary2.sln
11/16/2018 1:30 PM
C:\Users\Administrator\source\repos\ClassLibrary2
ClassLibrary1.sln
11/16/2018 1:26 PM
C:\Users\Administrator\source\repos\ClassLibrary1
WindowsTestApp2.sln
11/16/2018 1:21 PM
C:\Users\Administrator\source\repos\WindowsTestApp2
ConsoleApp1
11/16/2018 1:20 PM
C:\Users\Administrator\source\repos
ConsoleApp1.sln
11/16/2018 1:19 PM
C:\Users\Administrator\source\repos\ConsoleApp1
GitCloneSample.sln
11/16/2018 1:17 PM
C:\...\Administrator\Downloads\Sample Solutions\Online Repos\GitCloneSample
Clone or checkout code
Get code from an online code repository like GitHub or Azure DevOps
Open a project or solution
Open a local Visual Studio project or .sln file
Open a local folder
Navigate and edit code within any folder
Create a new project
Choose a project template with code scaffolding to get started
Continue without code

启动慢

格式化卡

打开项目慢

对VS固有印象是什么？

复制粘贴卡

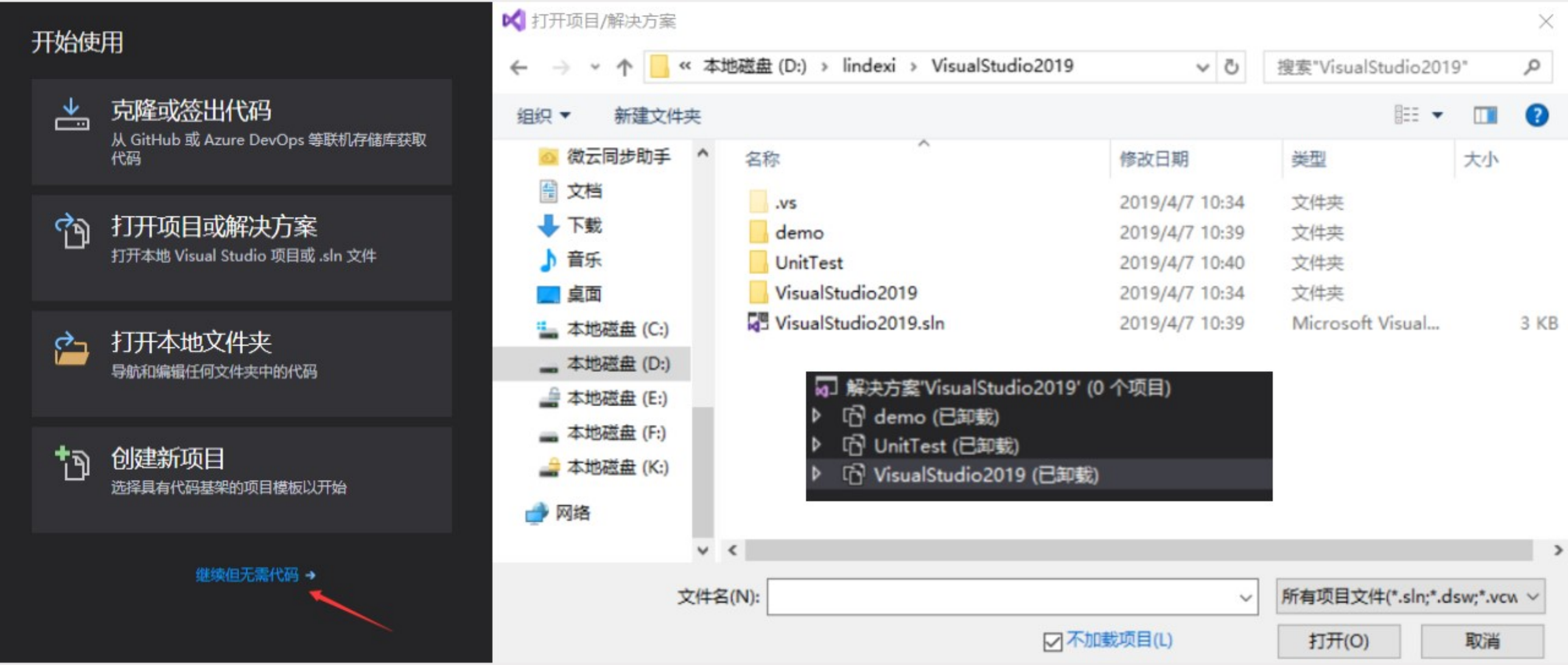
分析项目慢

输入卡

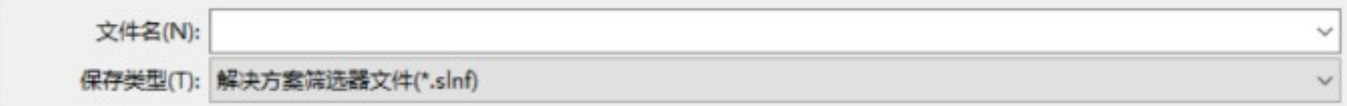
CVTE
Dream·Future

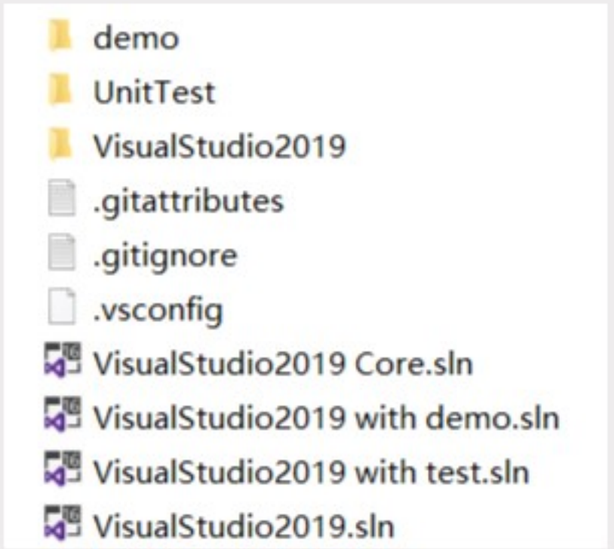
又卡又慢？ 

不加载大项目


开始使用
克隆或签出代码
从 GitHub 或 Azure DevOps 等联机存储库获取代码
打开项目或解决方案
打开本地 Visual Studio 项目或 .sln 文件
打开本地文件夹
导航和编辑任何文件夹中的代码
创建新项目
选择具有代码基架的项目模板以开始
继续但无需代码 →
打开项目/解决方案
« 本地磁盘 (D:) › lindexi › VisualStudio2019
搜索"VisualStudio2019"
组织
新建文件夹
微云同步助手
文档
下载
音乐
桌面
本地磁盘 (C:)
本地磁盘 (D:)
本地磁盘 (E:)
本地磁盘 (F:)
本地磁盘 (K:)
网络
名称
修改日期
类型
大小
.vs 2019/4/7 10:34 文件夹
demo 2019/4/7 10:39 文件夹
UnitTest 2019/4/7 10:40 文件夹
VisualStudio2019 2019/4/7 10:34 文件夹
VisualStudio2019.sln 2019/4/7 10:39 Microsoft Visual... 3 KB
解决方案'VisualStudio2019' (0 个项目)
demo (已卸载)
UnitTest (已卸载)
VisualStudio2019 (已卸载)
文件名(N):
所有项目文件(*.sln;*.dsw;*.vcw
不加载项目(L)
打开(O)
取消

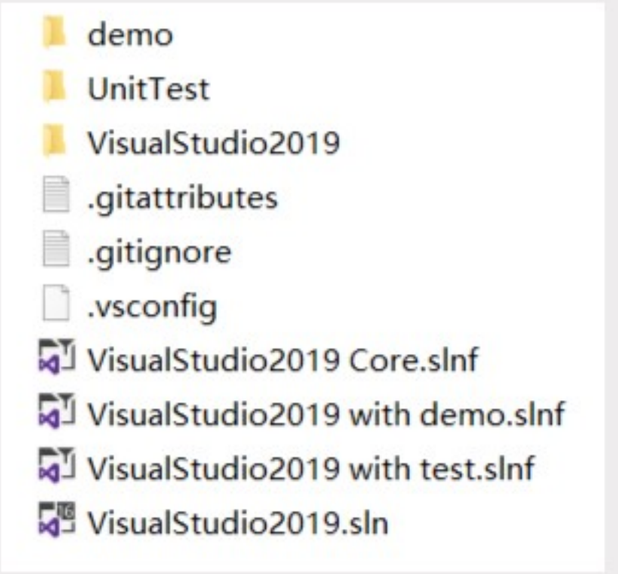
slnf 格式，保存大项目里面默认不加载项目





只带核心项目的
只带demo的
只带单元测试的

添加一个项目需要修改所有工程

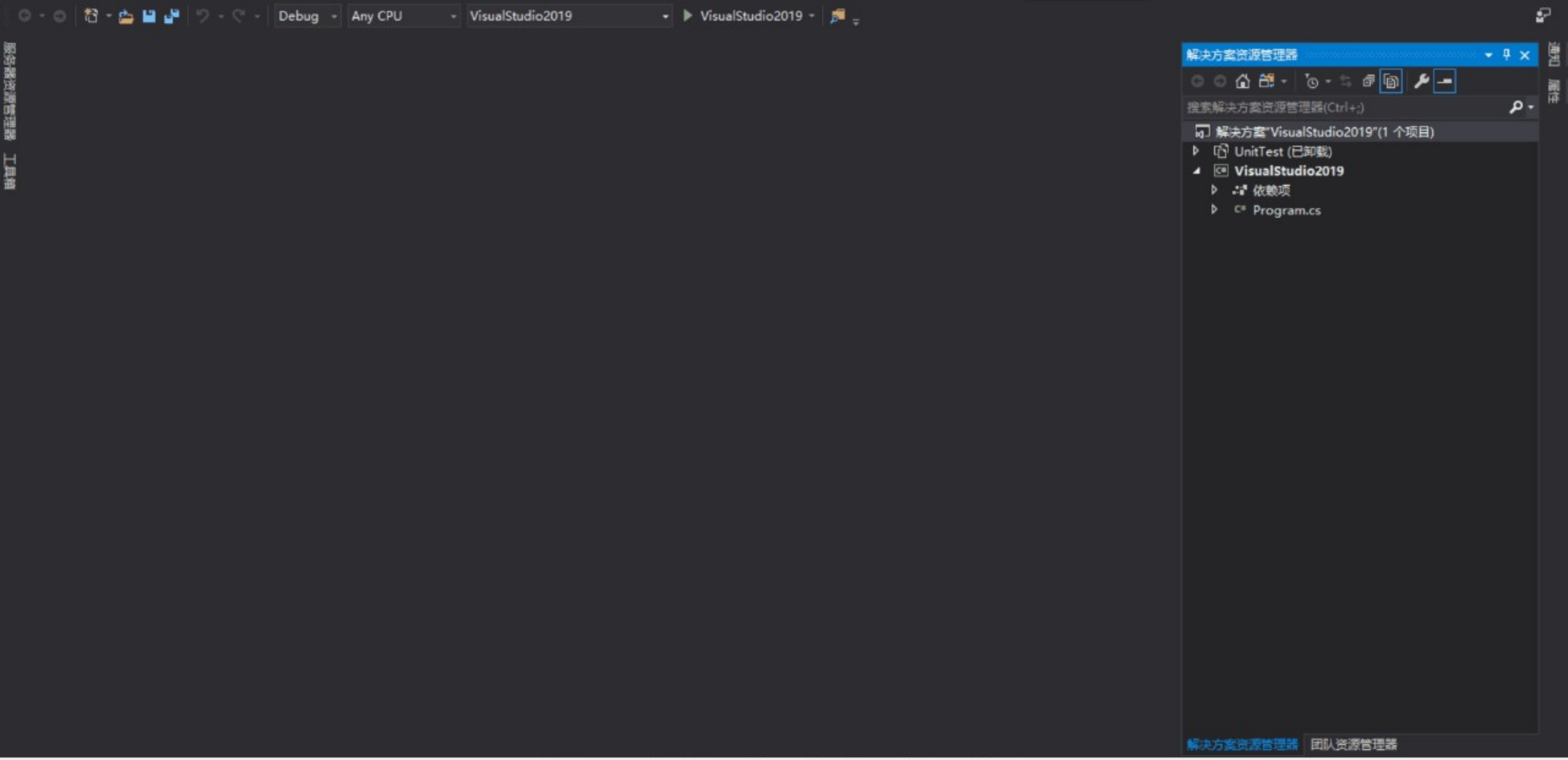


slnf=sln+filter
一次更改，所有项目生效

```
{
  "solution": {
    "path": "VisualStudio2019.sln",
    "projects": [
      "VisualStudio2019\\VisualStudio2019.csproj"
    ]
  }
}
```

依赖于sln文件，只指定需要加载的项目

VisualStudio 解决方案筛选器 slnf 文件



单个sln依赖，按需加载，按需编译

CVTE
Dream·Future

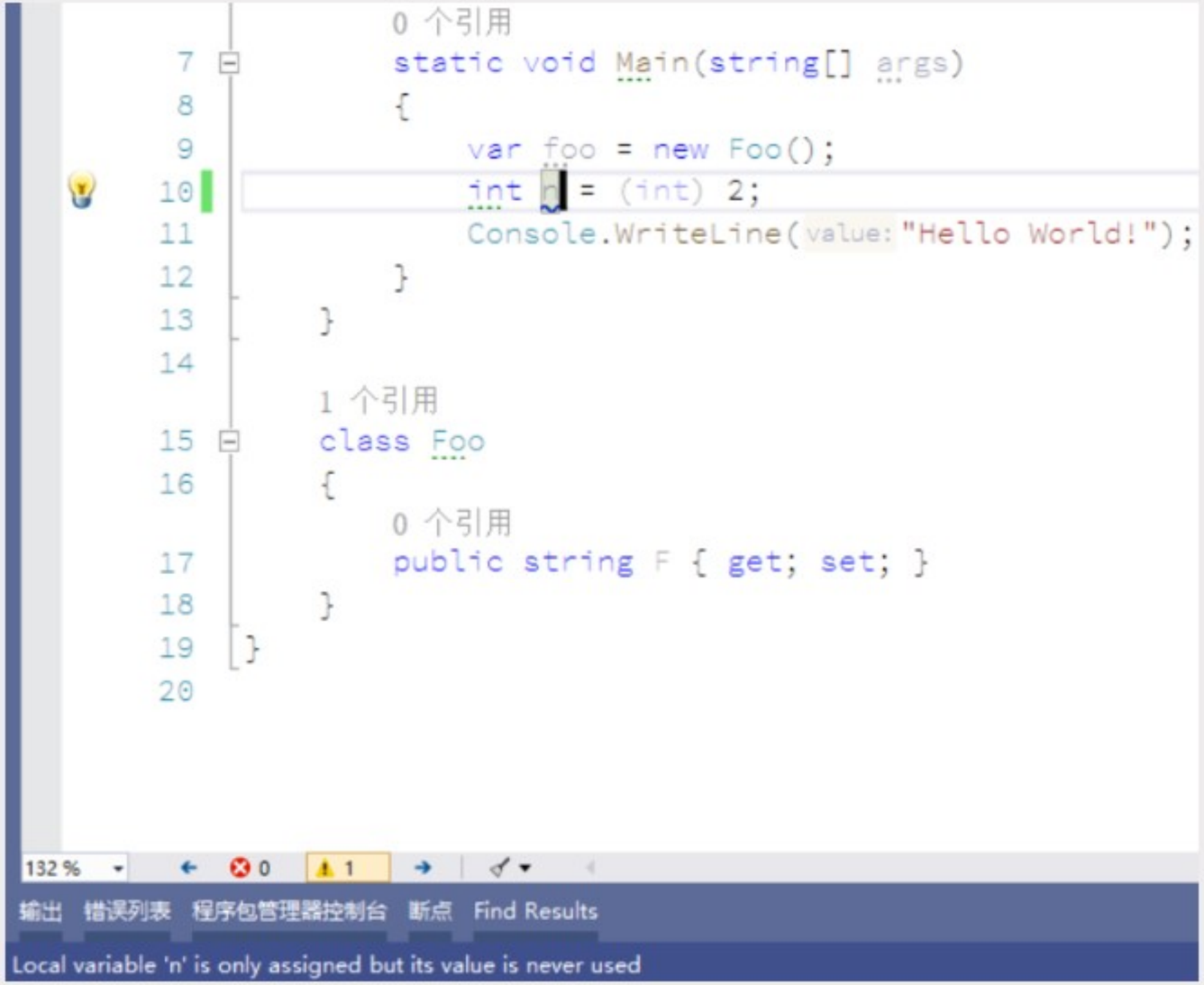
解决卡的问题

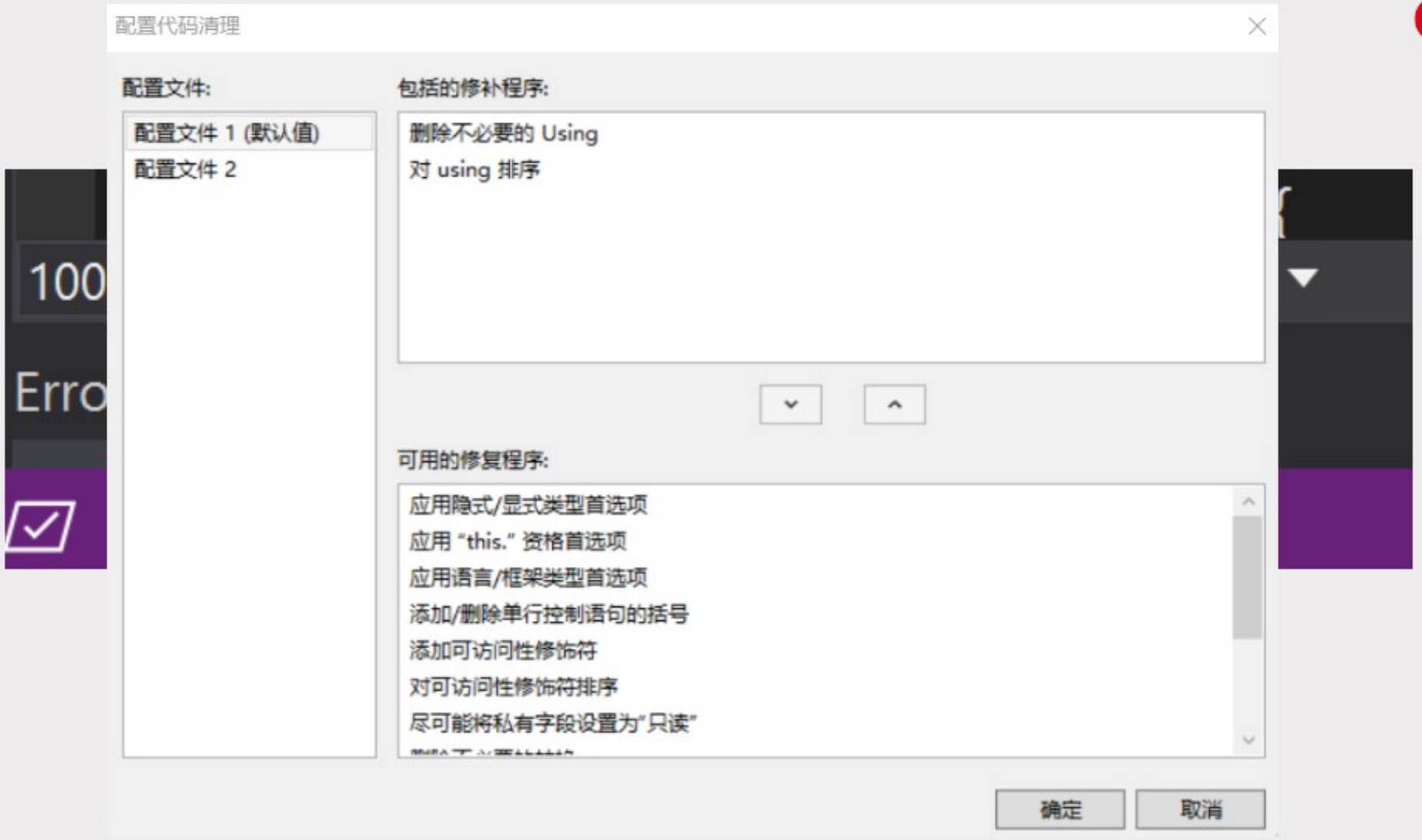
编辑性能优化


差异显示模式(D):
整行
显示差异概述边距
在复制/剪切时复制格式文本
最大长度(L):
10240
使用准确分类
键入时自动取消长时间运行的辅助操作
自动调整允许的最大键入延迟
允许的最大键入延迟(毫秒):

编辑工具 


namespace BirinugeGurwainaynanu
{
0 个引用
class Program
{
0 个引用
static void Main(string[] args)
{
var foo = new Foo();
int n = (int) 2
Console.WriteLine(value:"Hello World!");
}
}
1 个引用
class Foo
{
0 个引用
public string F { get; set; }
}
}
1
1
导航到下一个错误或警告(Alt+PgDn)
Find Results
1
1


0 个引用
static void Main(string[] args)
{
var foo = new Foo();
int n = (int) 2;
Console.WriteLine(value:"Hello World!");
}
}
1 个引用
class Foo
{
0 个引用
public string F { get; set; }
}
}
132 %
0
1
输出
错误列表
程序包管理器控制台
断点
Find Results
Local variable 'n' is only assigned but its value is never used


配置代码清理
配置文件:
配置文件 1 (默认值)
配置文件 2
包括的修补程序:
删除不必要的 Using
对 using 排序
可用的修复程序:
应用隐式/显式类型首选项
应用 “this.” 资格首选项
应用语言/框架类型首选项
添加/删除单行控制语句的括号
添加可访问性修饰符
对可访问性修饰符排序
尽可能将私有字段设置为“只读”
确定
取消

社区化

统一的格式化工具

Dotnet Format

- You can now apply code style preferences from the command-line with the dotnet format global tool. To install, you will need .NET Core 2.1 SDK or later. Run the following command in your favorite terminal: `dotnet tool install -g dotnet-format --version 3.0.0-beta4-19105-10`

dotnet-format

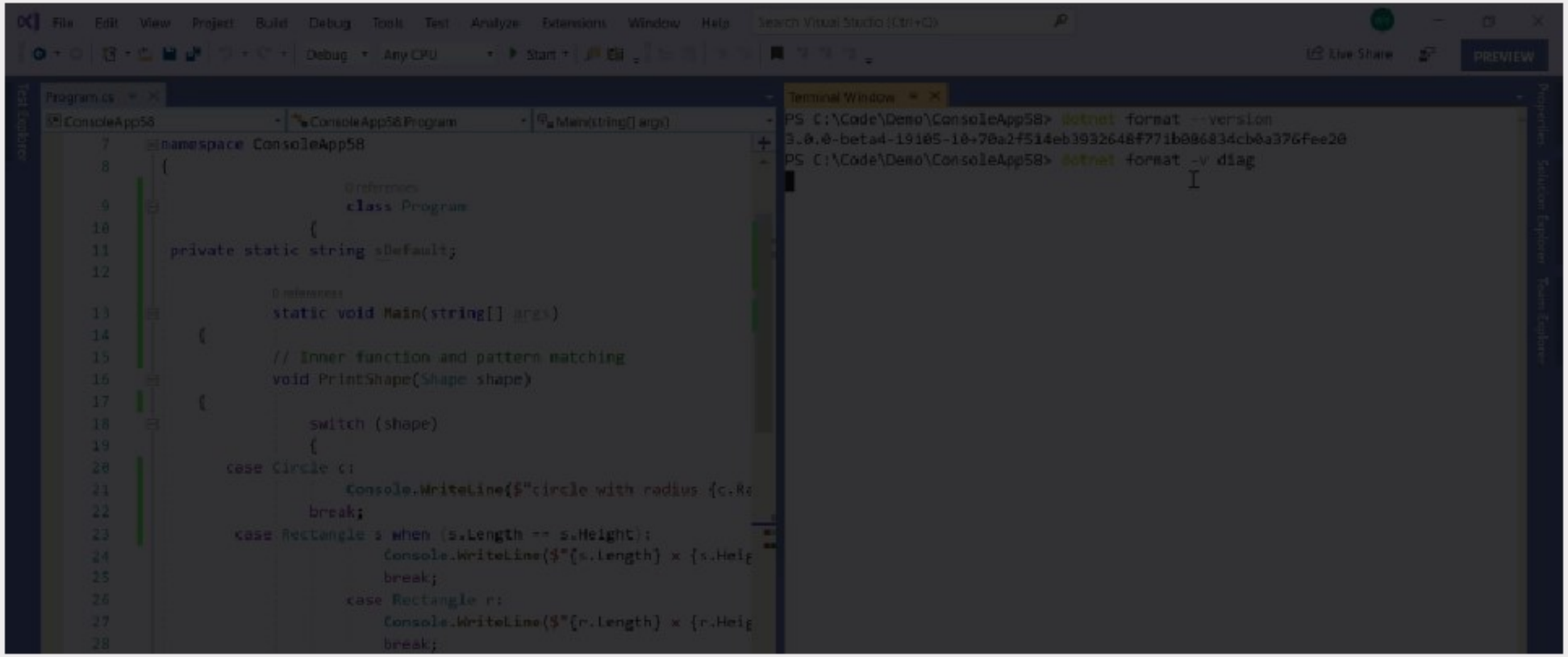
nuget v3.0.2

myget v3.0.4-prerelease.19179.2

Branch	Windows (Debug)	Windows (Release)	Linux (Debug)	Linux (Release)	Localization (Debug)	Localization (Release)
master	build succeeded	build succeeded	build succeeded	build failed	build succeeded	build succeeded

`dotnet-format` is a code formatter for `dotnet` that applies style preferences to a project or solution. Preferences will be read from an `.editorconfig` file, if present, otherwise a default set of preferences will be used. At this time `dotnet-format` is able to format C# and Visual Basic projects with a subset of supported .editorconfig options.

https://github.com/dotnet/format

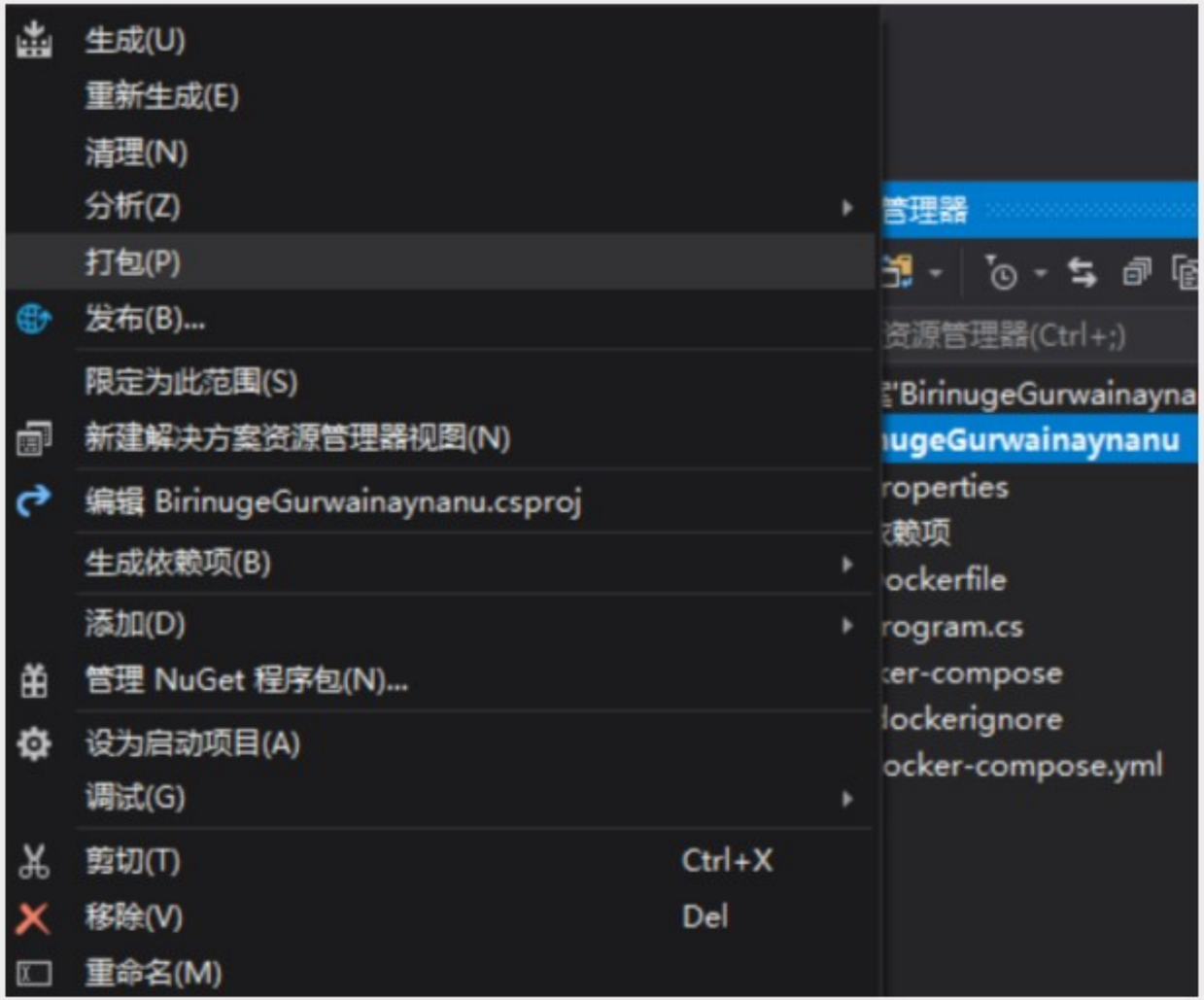


命令行可集成CI

支持私有 editor config 格式

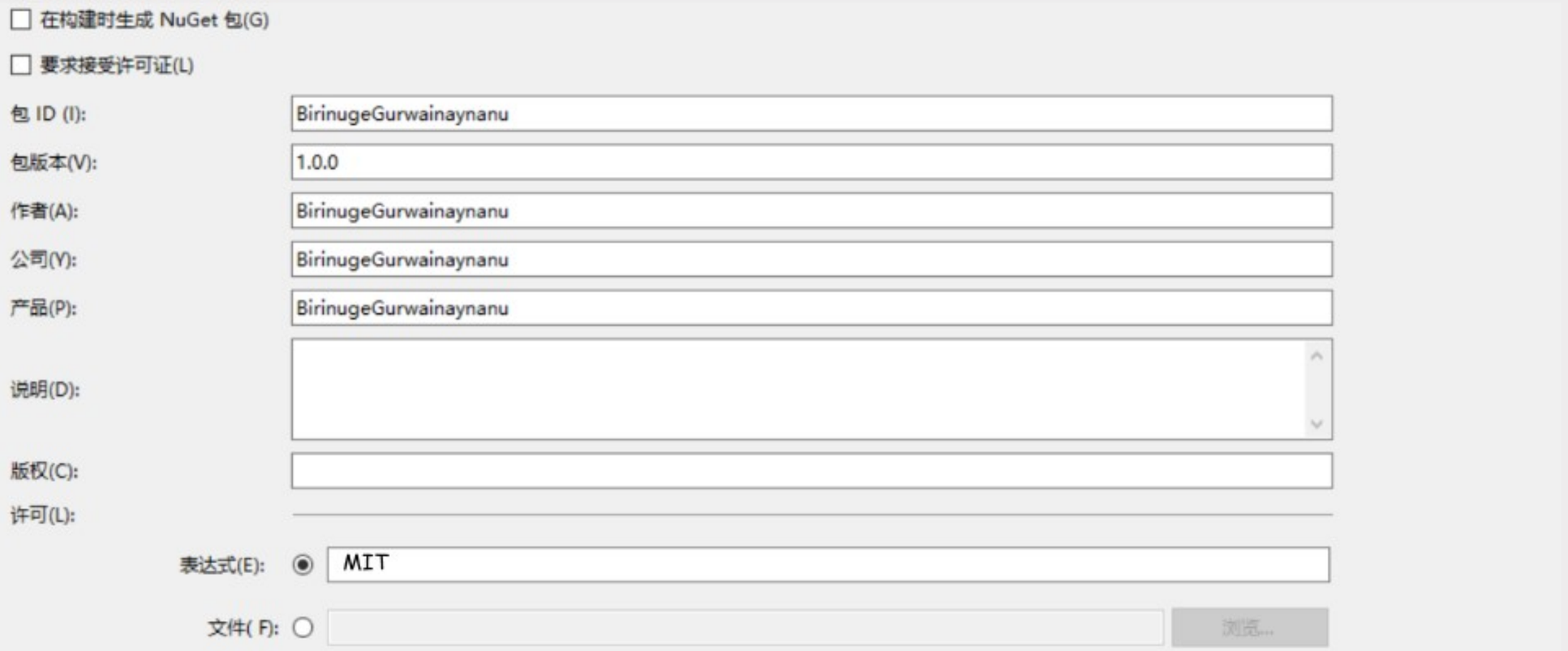
支持 VSC 格式化

更靠近社区开发的功能



右击一键打包 NuGet

许可证支持表达式或自定义的文件



开发效率

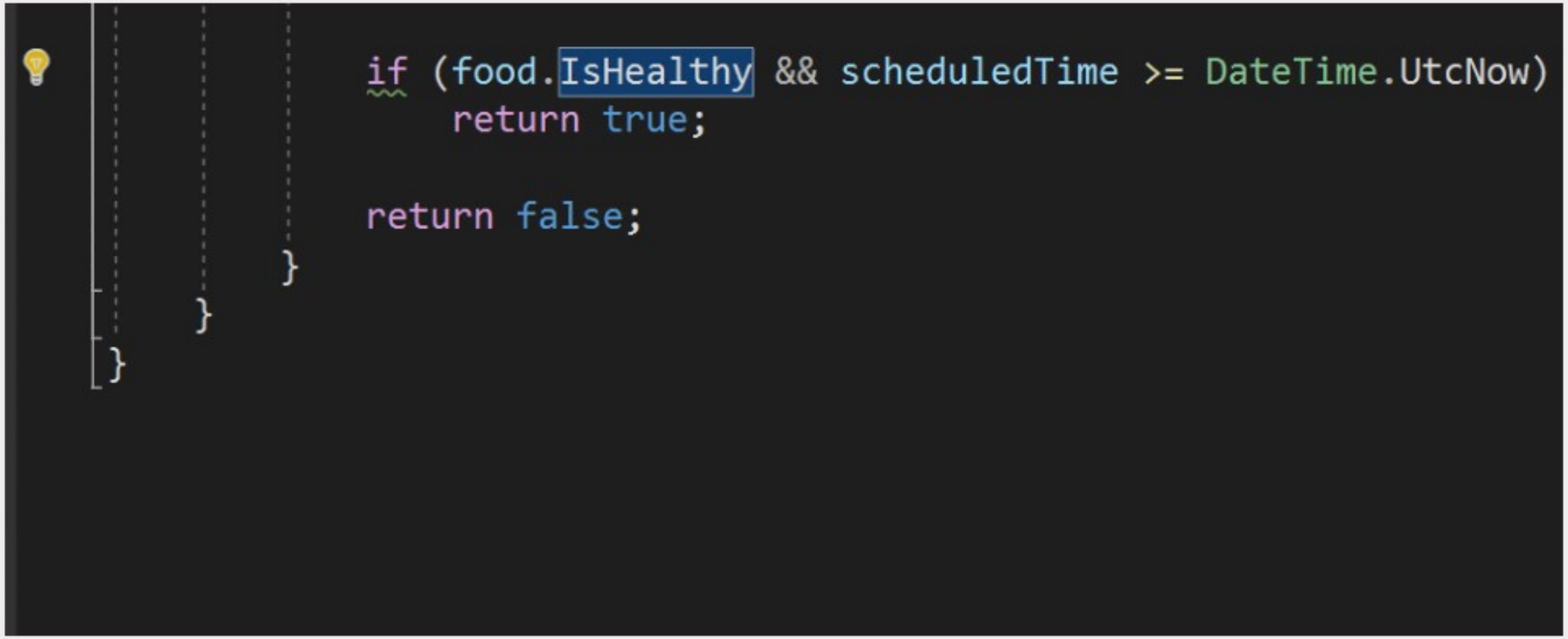
优化自动格式化和重构

CVTE
Dream·Future

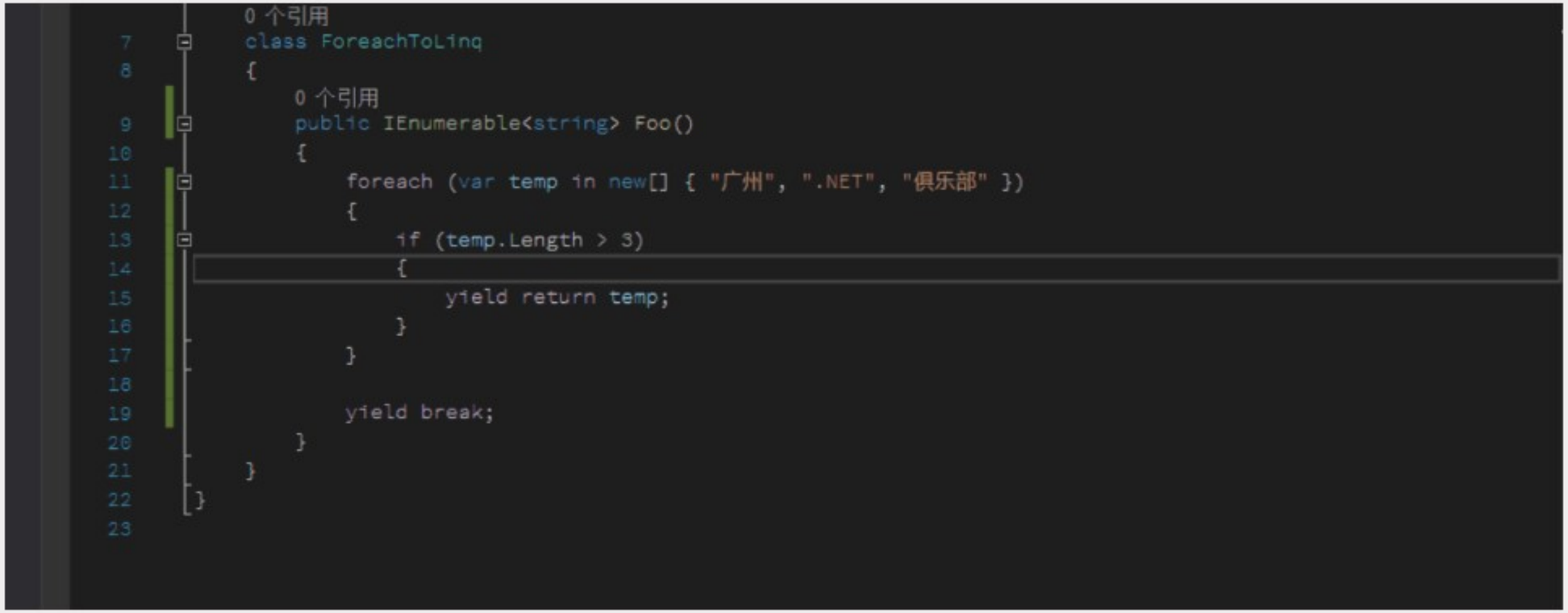
自动格式化和重构

- 让命名空间和文件所在文件夹保持一致
- 对类里面成员抽出接口
- 换行对齐参数
- 将匿名对象转换为 ValueTuple
- 转换代码为 Lambda 代码
- 反转if判断
- 自动使用 C# 8.0 的 range 功能
- 自动替换@$"为$@"

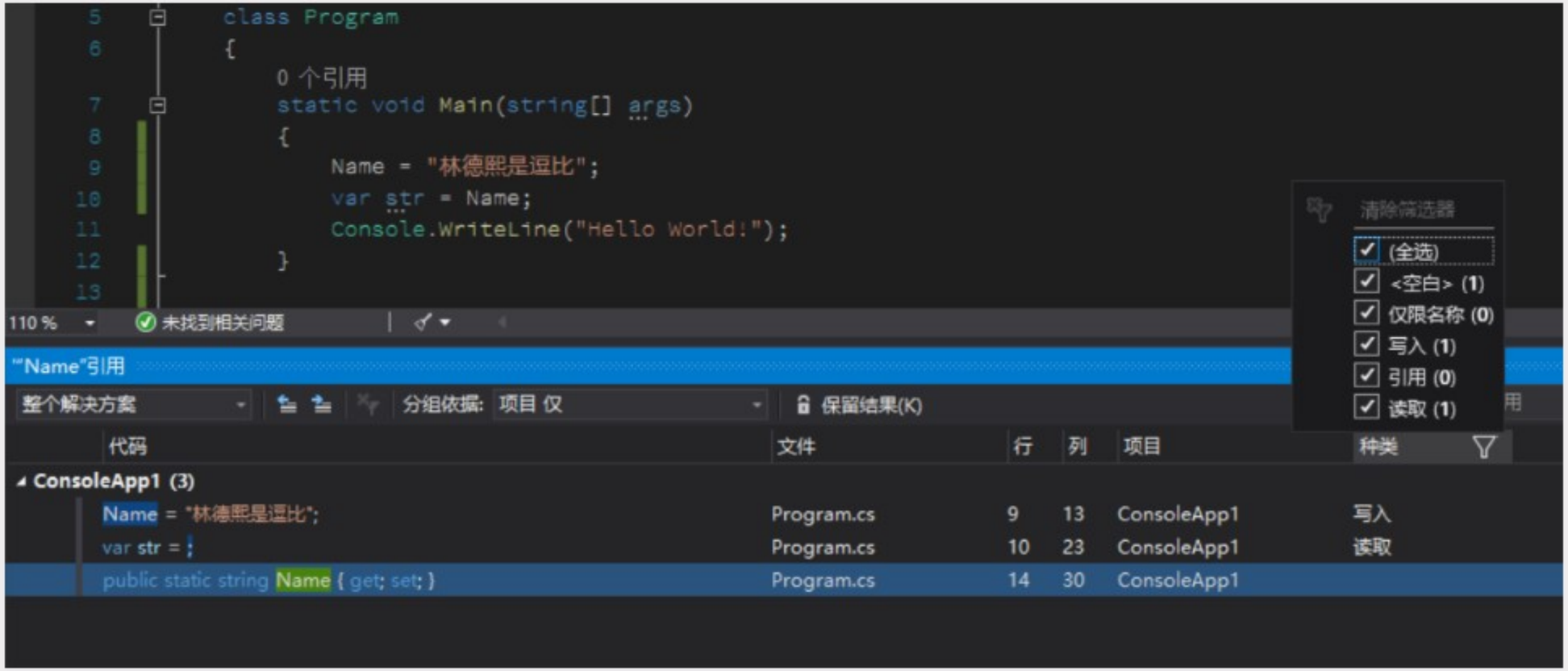
- 虽然还有很多功能，但是 Resharper 都有


if (food.IsHealthy && scheduledTime >= DateTime.UtcNow)
return true;
return false;
}
}
}

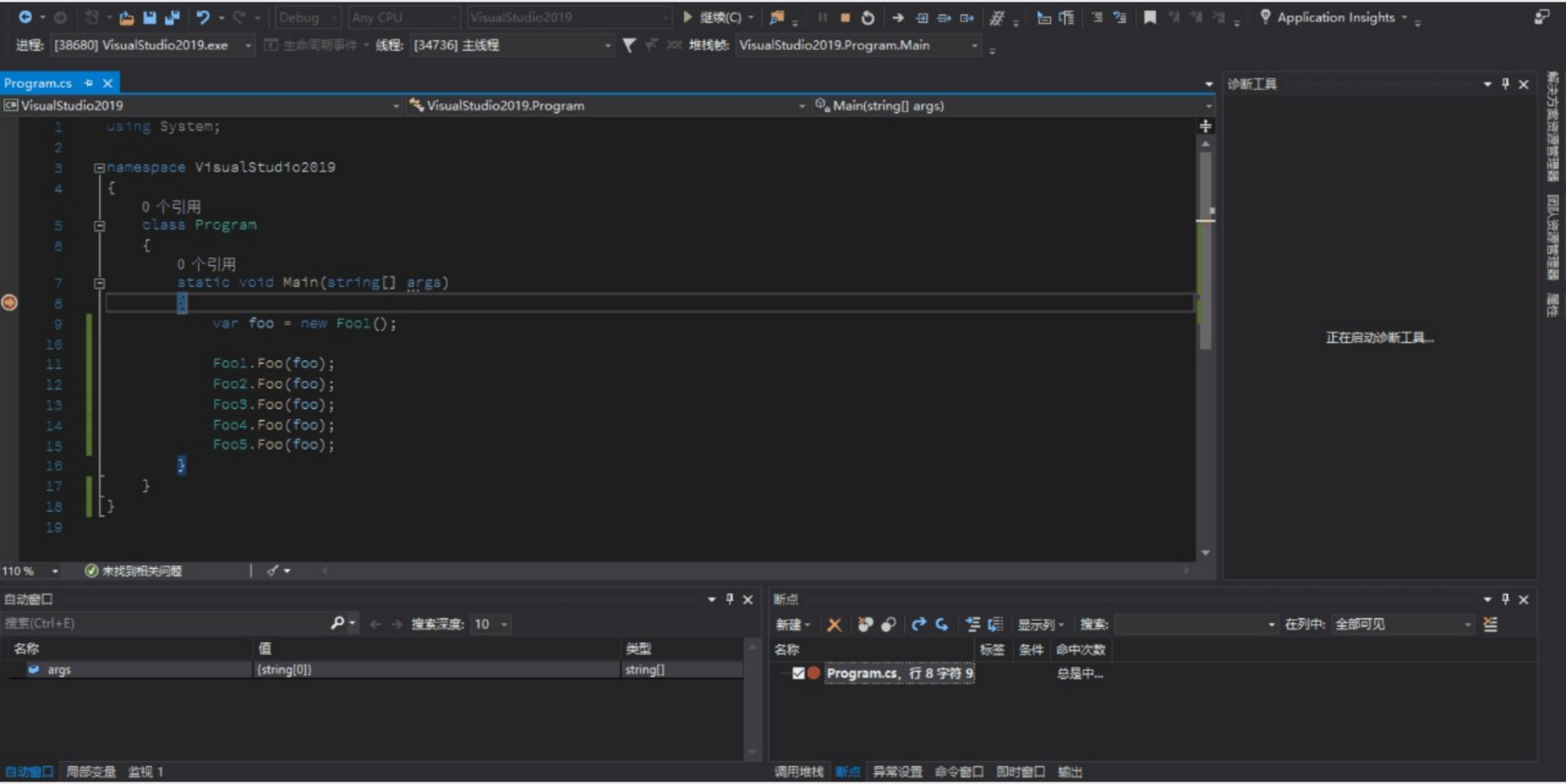
```
0 references
public void EatBreakfast()
{
    var banana = new Banana();

}

0 references
public bool Eat(IFood food, DateTime scheduledTime)
{
```

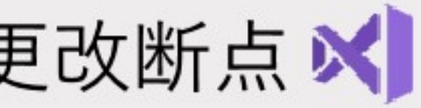

0 个引用
class ForeachToLinq
{
0 个引用
public IEnumerable<string> Foo()
{
foreach (var temp in new[] { "广州", ".NET", "俱乐部" })
{
if (temp.Length > 3)
{
yield return temp;
}
}
yield break;
}
}
}

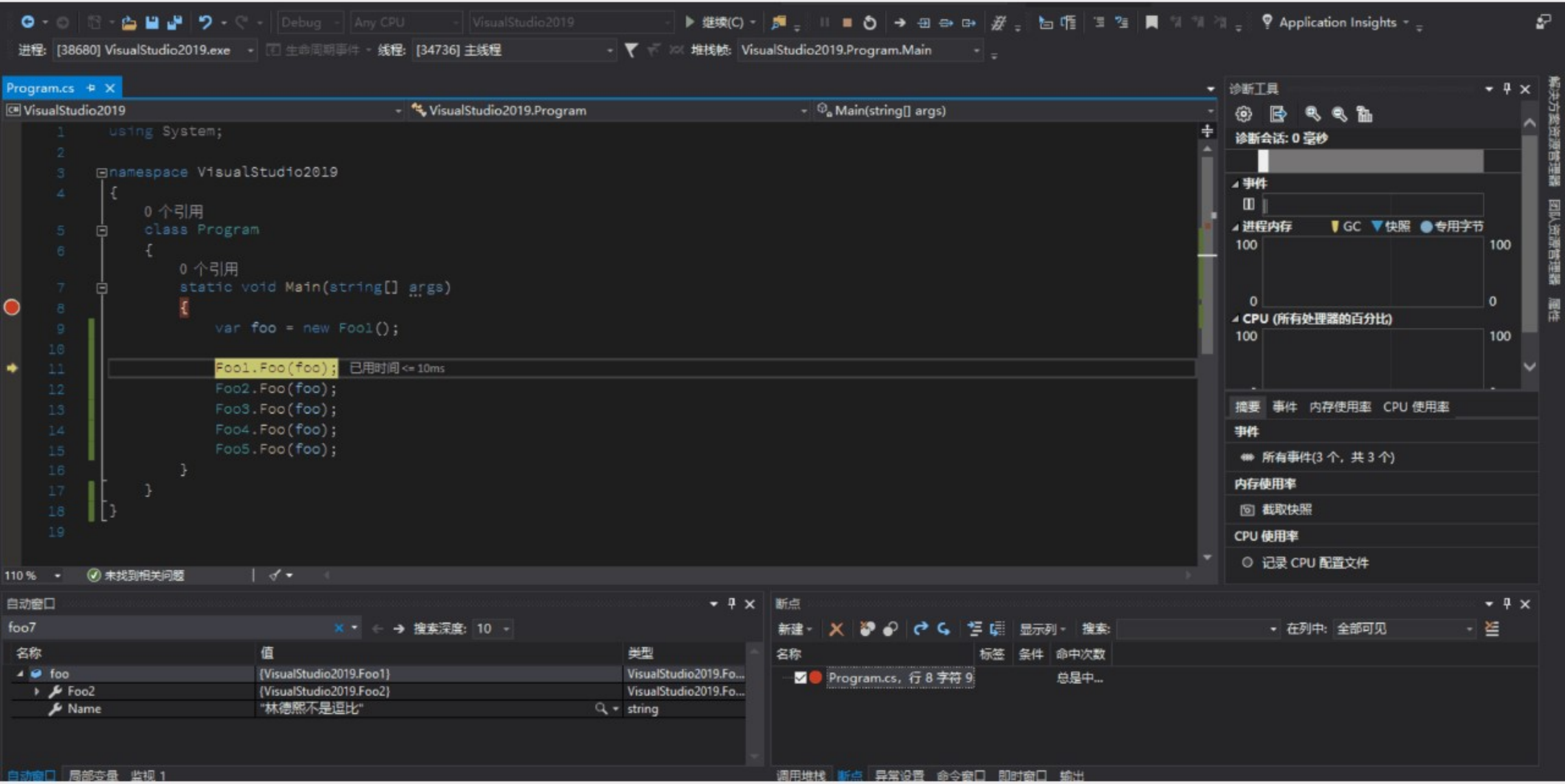
过滤引用


5 class Program
6 {
0 个引用
7 static void Main(string[] args)
8 {
9 Name = "林德熙是逗比";
10 var str = Name;
11 Console.WriteLine("Hello World!");
12 }
13
110 %
未找到相关问题
"Name"引用
整个解决方案
分组依据: 项目 仅
保留结果(K)
清除筛选器
(全选)
<空白> (1)
仅限名称 (0)
写入 (1)
引用 (0)
读取 (1)
代码
文件
行
列
项目
种类
ConsoleApp1 (3)
Name = "林德熙是逗比";
Program.cs 9 13 ConsoleApp1 写入
var str = ;
Program.cs 10 23 ConsoleApp1 读取
public static string Name { get; set; }
Program.cs 14 30 ConsoleApp1

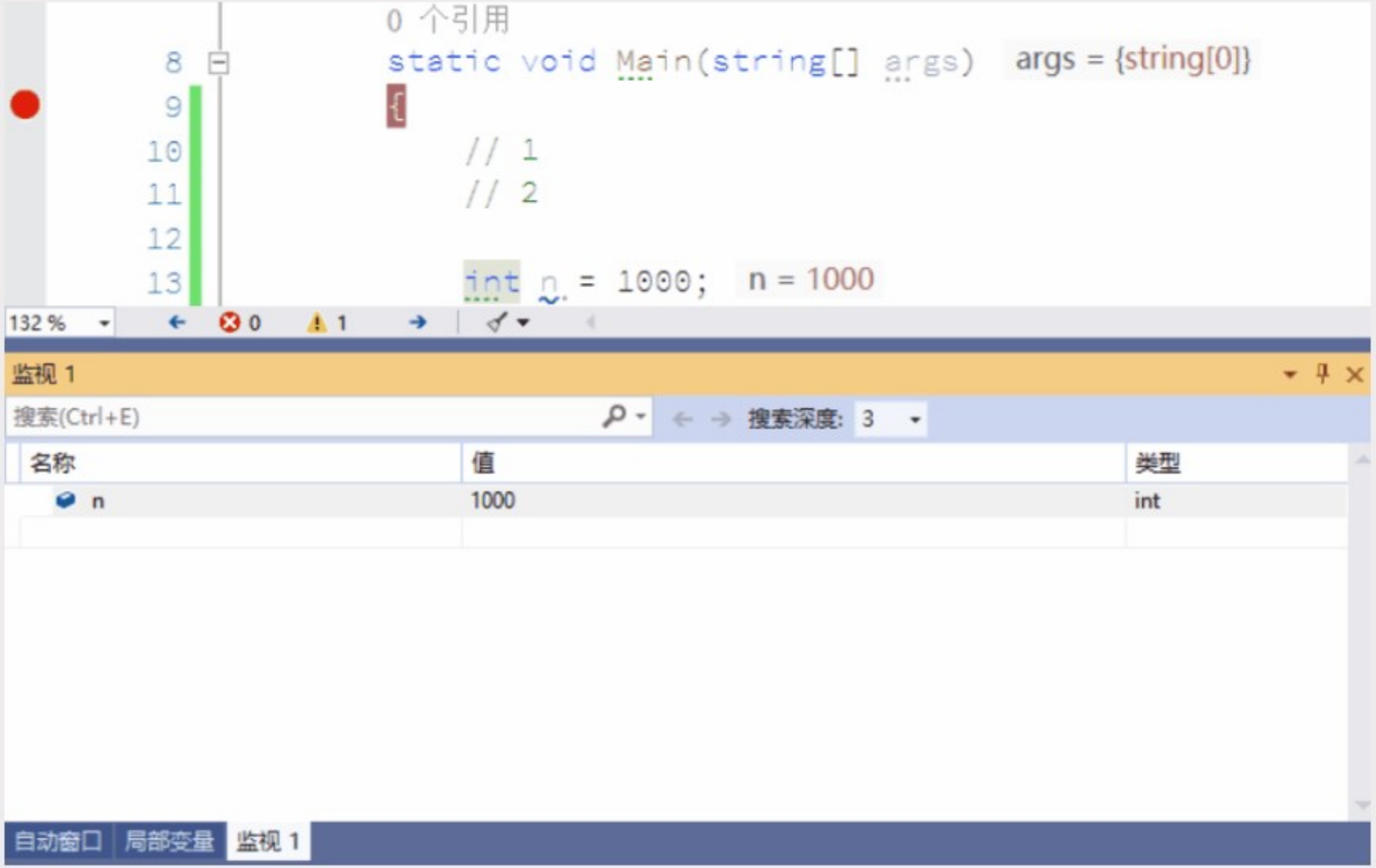
调试工具 


Debug
Any CPU
VisualStudio2019
继续(C)
Application Insights
进程: [38680] VisualStudio2019.exe
线程: [34736] 主线程
堆栈帧: VisualStudio2019.Program.Main
Program.cs
VisualStudio2019
VisualStudio2019.Program
Main(string[] args)
using System;
namespace VisualStudio2019
{
0 个引用
class Program
{
0 个引用
static void Main(string[] args)
{
var foo = new Foo1();
Foo1.Foo(foo);
Foo2.Foo(foo);
Foo3.Foo(foo);
Foo4.Foo(foo);
Foo5.Foo(foo);
}
}
}
诊断工具
正在启动诊断工具...
110 %
未找到相关问题
自动窗口
搜索(Ctrl+E)
搜索深度: 10
名称
值
类型
args
{string[0]}
string[]
自动窗口
局部变量
监视 1
断点
新建
显示列
搜索:
在列中: 全部可见
名称
标签
条件
命中次数
Program.cs, 行 8 字符 9
总是中...
调用堆栈
断点
异常设置
命令窗口
即时窗口
输出

值更改断点 


Debug
Any CPU
VisualStudio2019
继续(C)
Application Insights
进程: [38680] VisualStudio2019.exe
线程: [34736] 主线程
堆栈帧: VisualStudio2019.Program.Main
Program.cs
VisualStudio2019
VisualStudio2019.Program
Main(string[] args)
using System;
namespace VisualStudio2019
{
0 个引用
class Program
{
0 个引用
static void Main(string[] args)
{
var foo = new Foo1();
Foo1.Foo(foo); 已用时间 <= 10ms
Foo2.Foo(foo);
Foo3.Foo(foo);
Foo4.Foo(foo);
Foo5.Foo(foo);
}
}
}
110 %
未找到相关问题
诊断工具
诊断会话: 0 毫秒
事件
进程内存
GC
快照
专用字节
100
0
CPU (所有处理器的百分比)
摘要
事件
内存使用率
CPU 使用率
所有事件(3 个，共 3 个)
截取快照
记录 CPU 配置文件
自动窗口
foo7
搜索深度: 10
名称
值
类型
foo
{VisualStudio2019.Foo1}
VisualStudio2019.Fo...
Foo2
{VisualStudio2019.Foo2}
VisualStudio2019.Fo...
Name
"林德熙不是逗比"
string
自动窗口
局部变量
监视 1
断点
新建
显示列
搜索:
在列中:
全部可见
名称
标签
条件
命中次数
Program.cs，行 8 字符 9
总是中...
调用堆栈
断点
异常设置
命令窗口
即时窗口
输出

格式化显示内容提示


0 个引用
8 static void Main(string[] args) args = {string[0]}
9 {
10 // 1
11 // 2
12
13 int n = 1000; n = 1000
132 %
0
1
监视 1
搜索(Ctrl+E)
搜索深度: 3
名称
值
类型
n
1000
int
自动窗口
局部变量
监视 1

性能调试工具

细分的功能


分析目标
更改目标
启动项目
BirinugeGurwainaynanu
可用工具
.NET 对象分配跟踪
查看 .NET 对象分配的位置以及 GC 收回它们的时间。
检测
检测应用程序以调查确切的调用数和调用时间
应用程序时间线
检查应用程序中所用时间的分布情况。这对像低帧率之类的问题进行故障排除时很有用
CPU 使用率
查看 CPU 执行代码时的时间耗费情况。当 CPU 遇到性能瓶颈时很有用
内存使用率
检查应用程序内存以查找内存泄漏等问题

启动项目

BirinugeGurwainaynanu

可用工具

☑ .NET 对象分配跟踪
查看 .NET 对象分配的位置以及 GC 收回它们的时间。

☐ CPU 使用率
查看 CPU 执行代码时的时间耗费情况。当 CPU 遇到性能瓶颈时很有用

☐ 检测
检测应用程序以调查确切的调用数和调用时间

☐ 内存使用率
检查应用程序内存以查找内存泄漏等问题

☐ 应用程序时间线
检查应用程序中所用时间的分布情况。这对像低帧率之类的问题进行故障排除时很有用

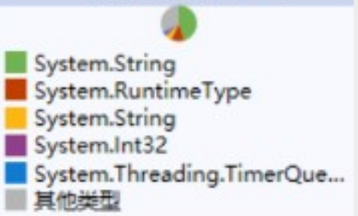
集合 分配 调用树 函数

显示其他类型

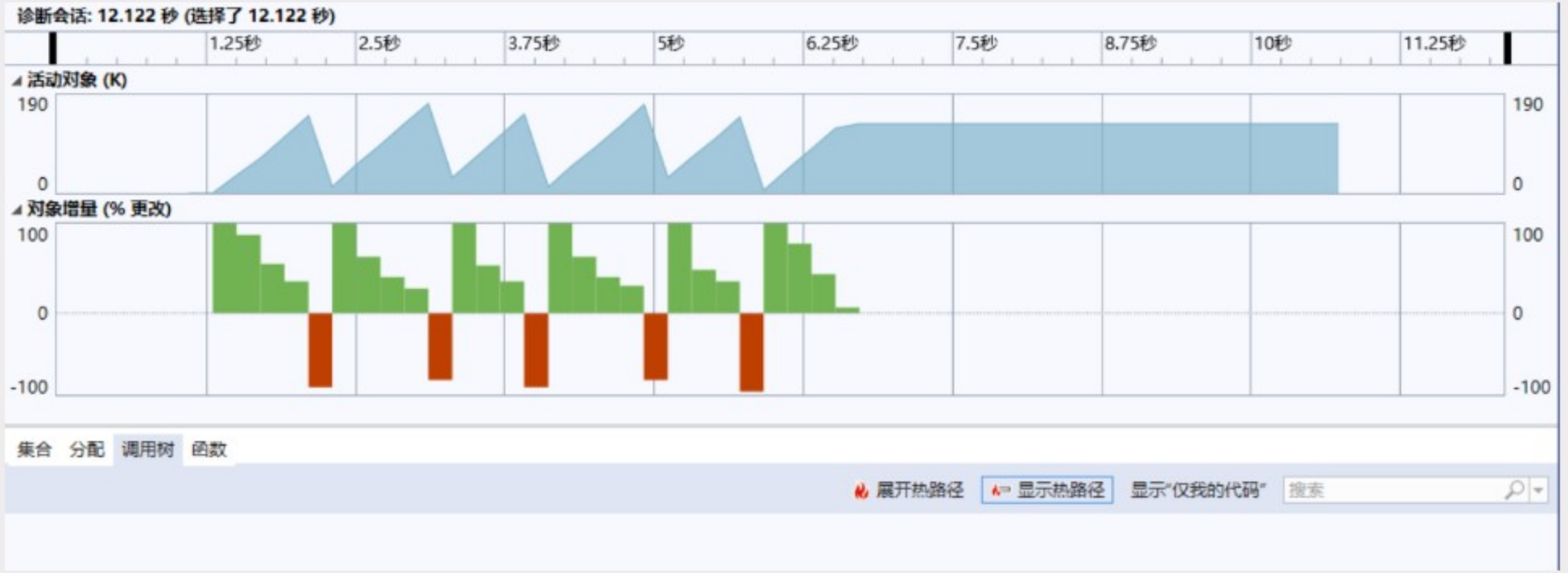
GC	收集的	已保留
5	174763	369
4	174763	369
3	174763	369
2	174763	369
1	167951	369

排在前列的已收集类型

排在前列的存活类型



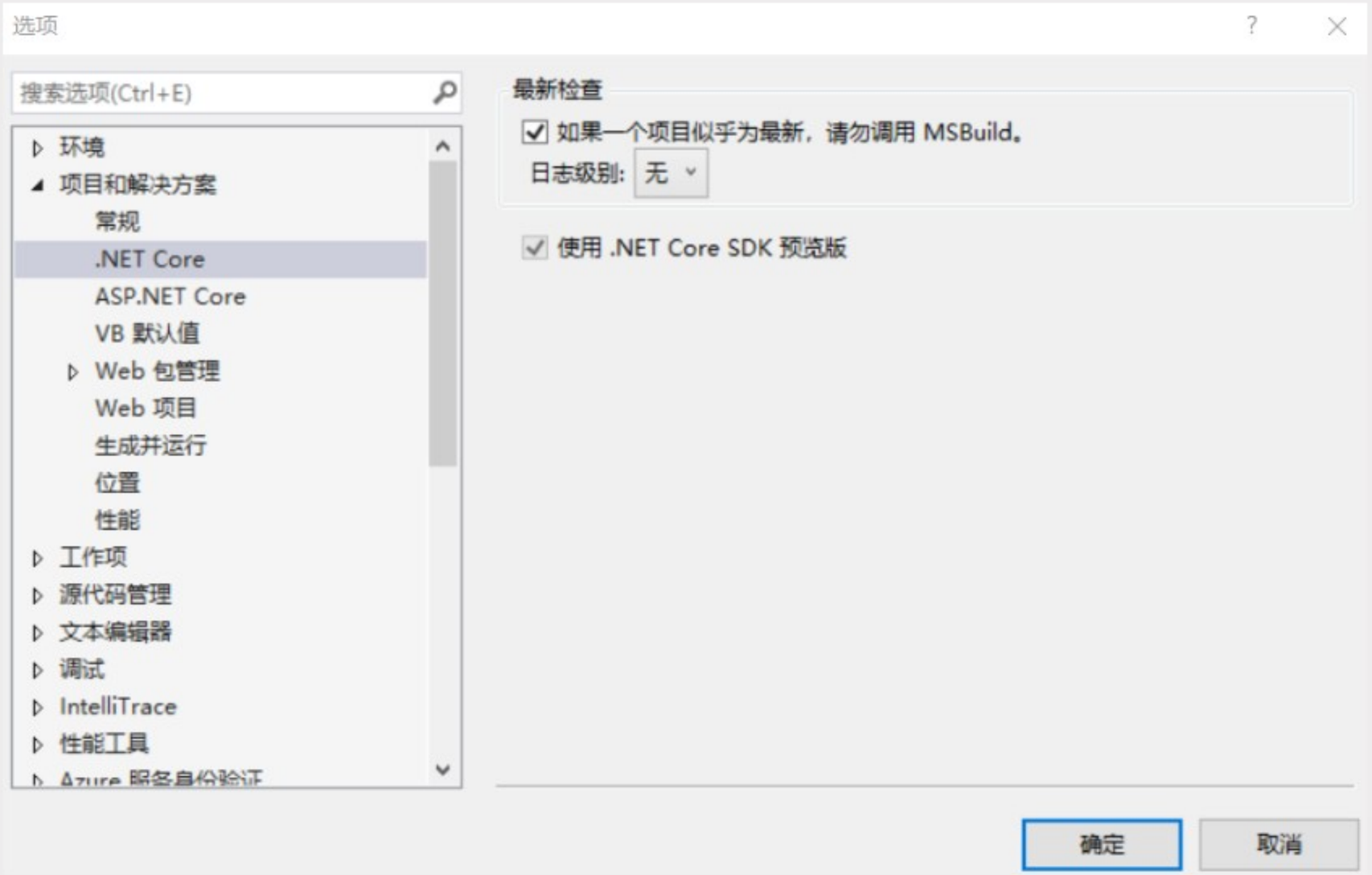
一键显示热路径

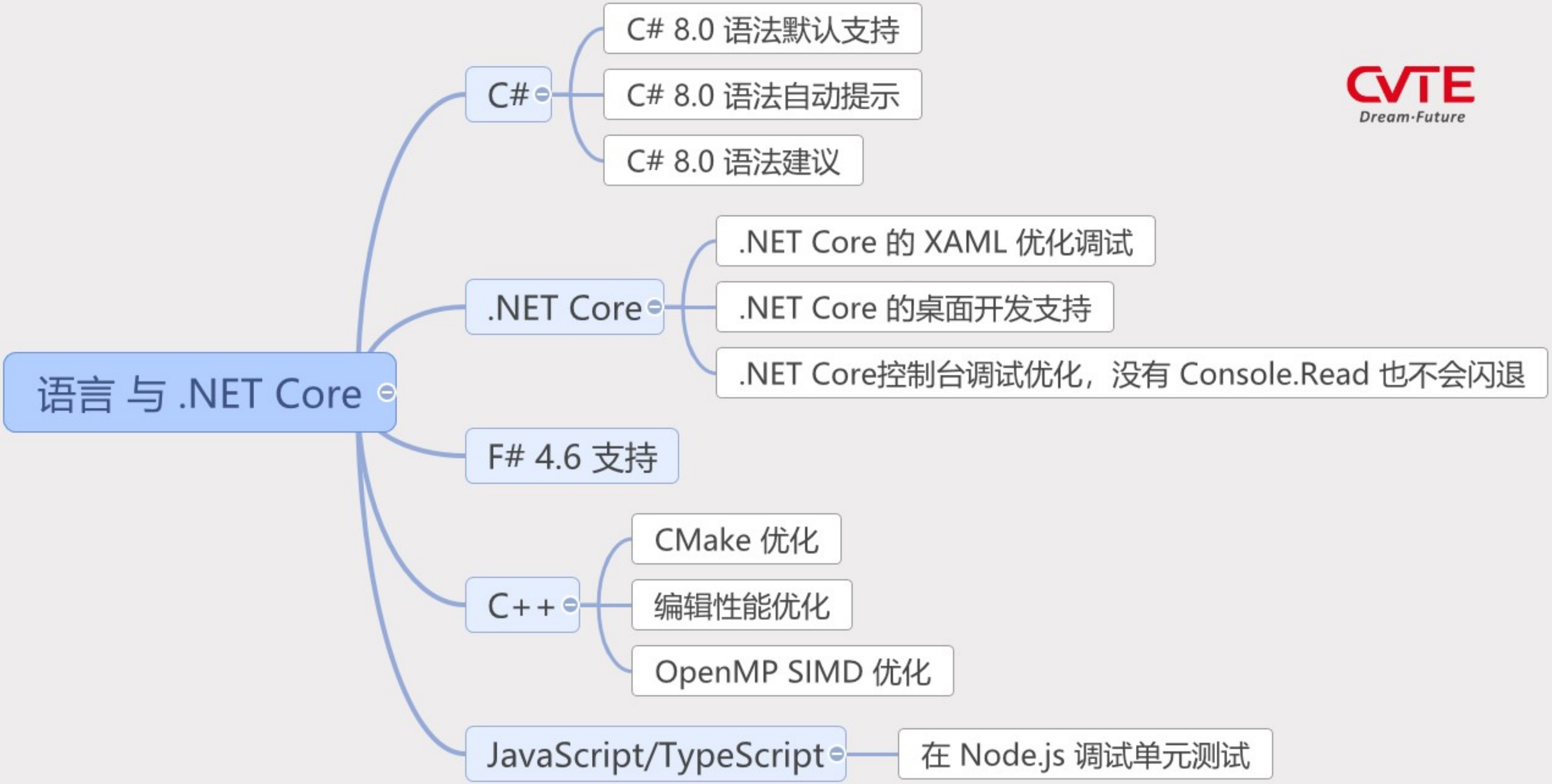

诊断会话: 12.122 秒 (选择了 12.122 秒)
1.25秒
2.5秒
3.75秒
5秒
6.25秒
7.5秒
8.75秒
10秒
11.25秒
活动对象 (K)
190
0
对象增量 (% 更改)
100
0
-100
集合
分配
调用树
函数
展开热路径
显示热路径
显示"仅我的代码"
搜索


集合 分配 调用树 函数
展开热路径
显示热路径
显示"仅我的代码"
搜索
函数名称
总计(分配)
自身(分配)
模块名
System.Private.CoreLib.dll!0x007ffc181... 132 0 System.Private.CoreLib.dll
System.Private.CoreLib.dll!0x007ffc1... 129 0 System.Private.CoreLib.dll
System.Private.CoreLib.dll!0x007ffc... 114 0 System.Private.CoreLib.dll
System.Private.CoreLib.dll!0x007... 113 0 System.Private.CoreLib.dll
[本机] coreclr.dll 113 0 coreclr.dll
System.Private.CoreLib.dll!... 113 0 System.Private.CoreLib.dll
[本机] coreclr.dll 113 0 coreclr.dll
System.Private.CoreLi... 112 0 System.Private.CoreLib.dll
System.Private.Core... 86 0 System.Private.CoreLib.dll
System.Private.Co... 55 0 System.Private.CoreLib.dll
System.Private.... 55 0 System.Private.CoreLib.dll
System.Privat... 30 0 System.Private.CoreLib.dll
System.Privat... 25 0 System.Private.CoreLib.dll

更多平台支持优化

默认使用预览版（如果是预览版的VS）




语言 与 .NET Core
C#
C# 8.0 语法默认支持
C# 8.0 语法自动提示
C# 8.0 语法建议
.NET Core
.NET Core 的 XAML 优化调试
.NET Core 的桌面开发支持
.NET Core控制台调试优化，没有 Console.Read 也不会闪退
F# 4.6 支持
C++
CMake 优化
编辑性能优化
OpenMP SIMD 优化
JavaScript/TypeScript
在 Node.js 调试单元测试

CVTE
Dream·Future

C# 8.0

CVTE
Dream·Future

switch

switch expressions

```
public enum Rainbow
{
    Red,
    Orange,
    Yellow,
    Green,
    Blue,
}
```

```
public static string FromRainbow(Rainbow colorBand) =>
    colorBand switch
    {
        Rainbow.Red    => "红",
        Rainbow.Orange => "橙",
        Rainbow.Yellow => "黄",
        Rainbow.Green  => "绿",
        Rainbow.Blue   => "蓝",
        _              => throw new
ArgumentException(message: "invalid enum value", paramName:
nameof(colorBand)),
    };
```


switch
将 case 和 : 替换为 =>
将 break 替换为逗号
将默认替换为下划线

property pattern

```
internal class Foo
{
    public string F { get; set; }
}


private static string LeafalljurearjehuNerawaljeeyinaryem(Foo foo)
{
    var str = foo switch
    {
        { F: "林德熙" } => "林德熙是逗比",
        { F: "逗比" } => "逗比",
    };

    return str;
}
```

Tuple patterns

```
private static string Foo(string first, string second)
{
    var str = (first, second) switch
    {
        ("lin", "dexi") => "林德熙是逗比",
        (_, "dexi") => "没错，这就是逗比",
        (_, _) => "不认识",
    };

    return str;
}

    Console.WriteLine(Foo("lin", "dexi"));
    Console.WriteLine(Foo("逗比", "dexi"));
    Console.WriteLine(Foo(null, null));
```

```
林德熙是逗比
没错，这就是逗比
不认识
```

CVTE
Dream·Future

Deconstruct

逐渐 C# 和 .NET 分离

```
public class Point
{
    public int X { get; }
    public int Y { get; }

    public Point(int x, int y) => (X, Y) = (x, y);

    public void Deconstruct(out int x, out int y) =>
        (x, y) = (X, Y);
}

private static string Foo(Point point)
{
    var (x, y) = point;
    return $"{x},{y}";
}
```

Positional patterns

```
private static string Foo(Point point) =>
    point switch
{
    var (x, y) when x > 0 && y > 0 => "两个值都大于零",
    var (x, y) when x < 0 && y < 0 => "两个值都小于零",
    var (_, _) => "不合法的值",
    _ => throw new ArgumentException("point"),
};

    Console.WriteLine(Foo(new Point(1, 1)));
    Console.WriteLine(Foo(new Point(1, 0)));
    Console.WriteLine(Foo(null));
```

```
两个值都大于零
不合法的值
System.ArgumentException:"point"
```

using declarations

```
static void WriteLinesToFile(string str)
{
    using var file = new System.IO.StreamWriter("lindexi.txt");

    file.WriteLine(str);
    // file 会在这里释放
}
```

Static local functions

```
private static int Foo()
{
    int y = 5;
    int x = 7;
    return Add(x, y);

    static int Add(int left, int right) => left + right;
}
```

```
private static int Foo()
{
    int y = 5;
    int x = 7;
    return Add(x, y);

    static int Add(int left, int right) => x + right;
}
```

静态本地函数不能包含对“x”的引用

CVTE
Dream·Future

Disposable ref structs

```
public ref struct Point
{
    public int X { get; }
    public int Y { get; }

    public Point(int x, int y) => (X, Y) = (x, y);

    public void Dispose()
    {
    }
}
```

```
private static void Foo()
{
    Point point = new Point();
    using (point)
    {
    }
}
```

C# 和 .NET 关系不紧密

Asynchronous streams

```
private static async void Foo()
{
    await foreach (var number in GenerateSequence())
    {
        Console.WriteLine(number);
    }
}

private static async System.Collections.Generic.IAsyncEnumerable<int>
GenerateSequence()
{
    for (int i = 0; i < 20; i++)
    {
        await Task.Delay(100);
        yield return i;
    }
}
```

Indices and ranges

Span<int>

int[]

```
var n = new[]
{
    0, 1, 2, 3, 4, 5, 6, 7, 8, 9
};
// 倒数第1个
var foo = n[^1]; // 9
// 倒数第5个到倒数第1个
var f1 = n[^5..^1];
// 从第3个开始到最后一个
var f2 = n[3..];
// 从第3个到第6个
var f3 = n[3..6];
// 获取全部
var all = n[..];
```

~~List<int>~~

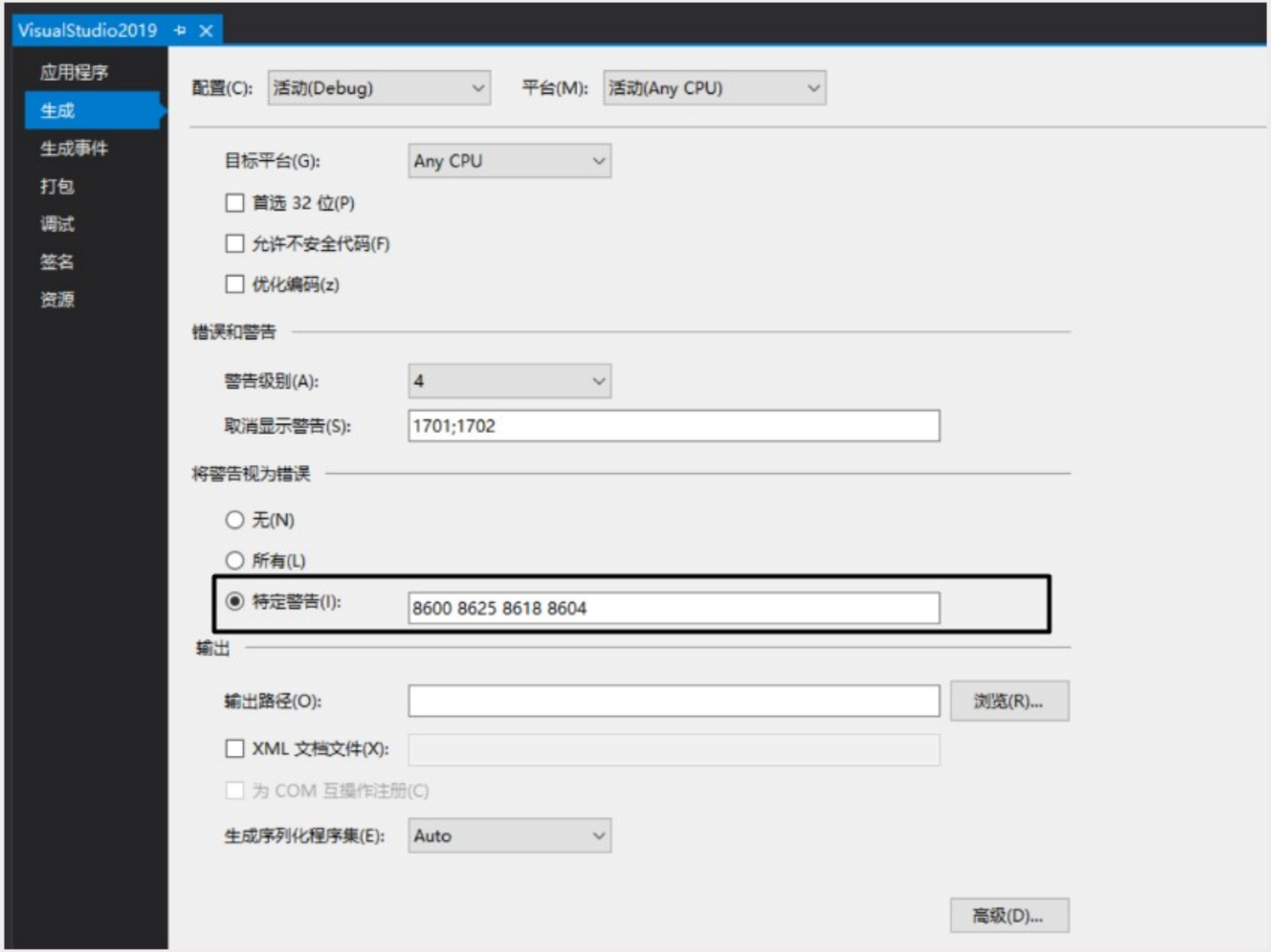
~~int*~~

```
Range range = 3..;
```

```
        range = ^1..^3;

        var intCollection = new IntCollection();
        foreach(var temp in intCollection[range])
        {
            // 99 98
        }



2 个引用
public class IntCollection
{
    1 个引用
    public IntCollection()...

    1 个引用
    public System.Collections.Generic.IEnumerable<int> this[System.Range range]...

    private int[] _intCollection;
}
```

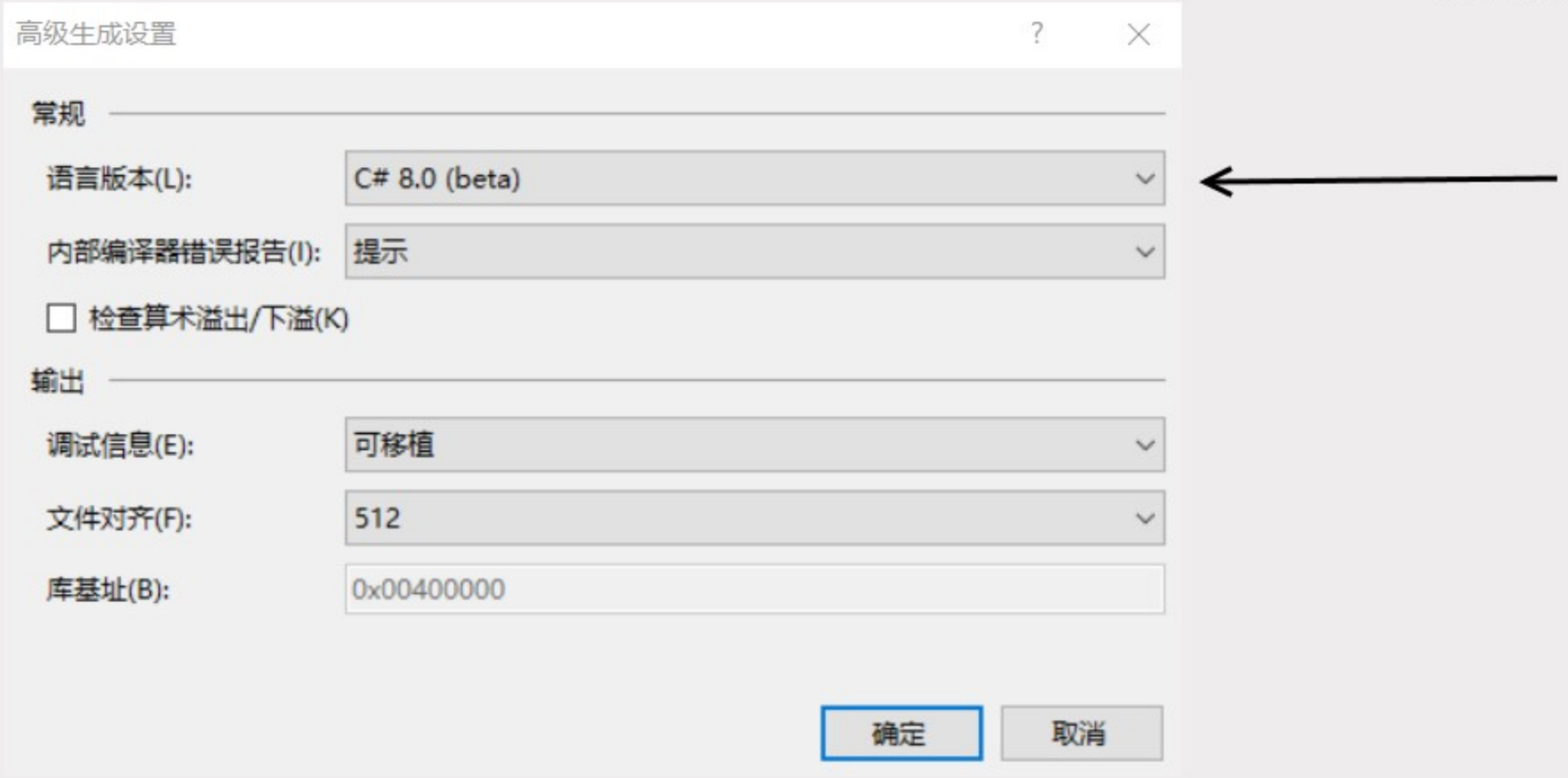
Nullable reference type

所有值不可空，除非显式指定



CVTE Dream·Future

自来喵的野


高级生成设置
常规
语言版本(L):
C# 8.0 (beta)
内部编译器错误报告(I):
提示
检查算术溢出/下溢(K)
输出
调试信息(E):
可移植
文件对齐(F):
512
库基址(B):
0x00400000
确定
取消

很多不需要关心，但很有用的功能

Visual Studio 2019

适用于 Android、iOS、Windows、Web 和云的功能完备型集成开发环境 (IDE)

版本：16。
发行说明

比较版本
如何离线安装

社区

功能强大的 IDE，免费供学生、开放源代码参与者和个人使用

免费下载

下载预览版 ›

Professional

最适合小型团队的专业 IDE

免费试用

下载预览版 ›

企业版

适用于任何规模团队的可缩放端到端解决方案

免费试用

下载预览版 ›

CVTE
Dream·Future

VisualStudio 2019 新特性