

第05期-启动调试器和断点跟踪

演示代码

```
public class TestClass
{
    public int Count { get; set; }

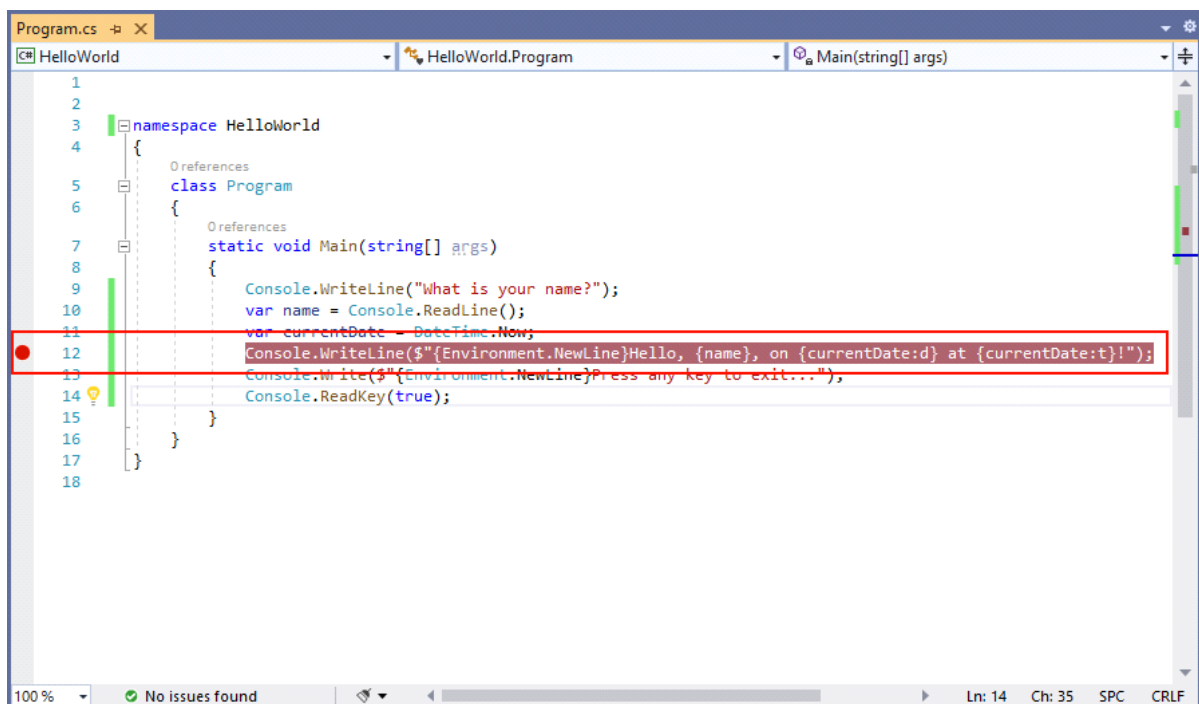
    public int Sum(int a)
    {
        Count++;

        int result = 0;

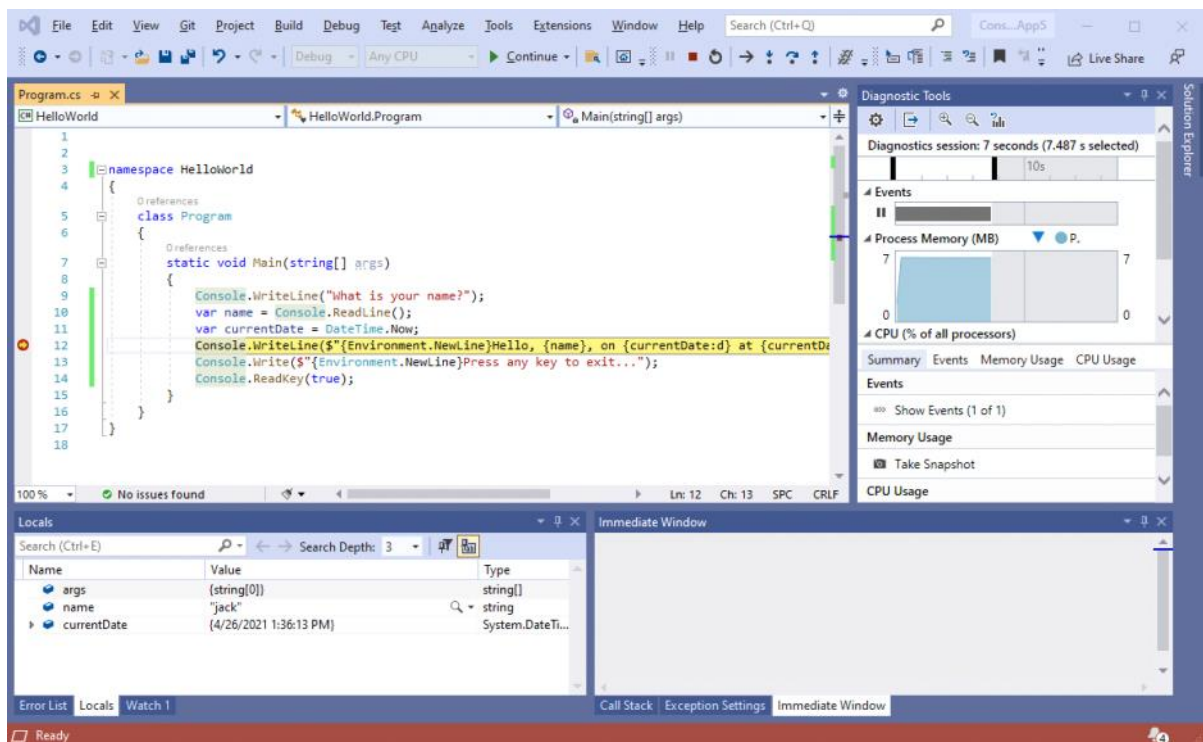
        for (int i = 0; i < a; i++)
        {
            result += i;
        }

        return result;
    }
}
```

设置断点



开始调试 F5 运行



跟踪方式

- F11单步执行：每次一条语句
- F10单步跳过：跳过函数或者方法调用
- Shift+F11: 跳出当前函数或方法
- 鼠标绿色箭头快速运行到某个点
- 运行到光标处
- Ctrl+Shift+F5：快速重启应用
- 编写和调试正在运行的代码

条件断点

位置: Program.cs, 行: 2, 字符: 1, 必须匹配源

☒ 条件

条件表达式: 为 true 示例: $x == 5$

添加条件

☐ 操作

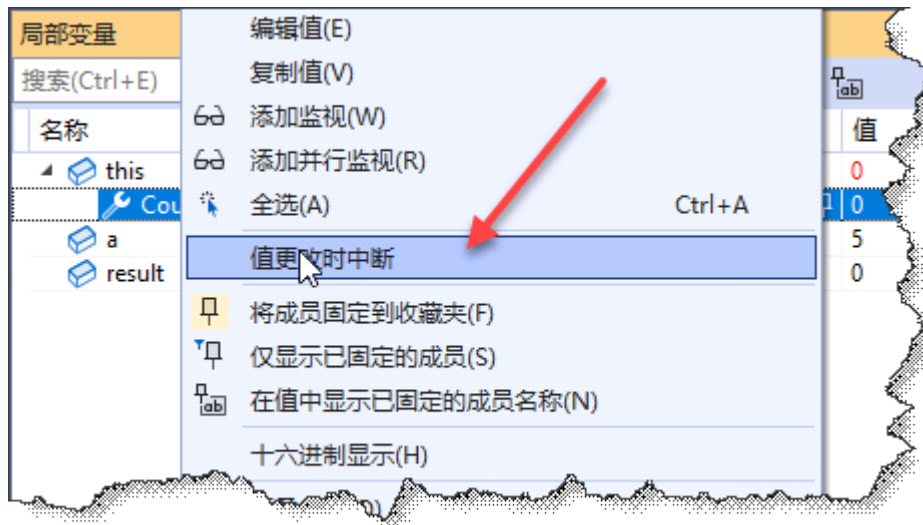
☐ 点击后移除断点

☐ 仅当命中以下断点时启用:

关闭

- 创建对象 ID 简化使用

- 筛选条件支持: MachineName, ProcessName, ThreadId , ThreadName
- 设置函数断点
- 数据断点, 支持 .NET 5 + 以上。

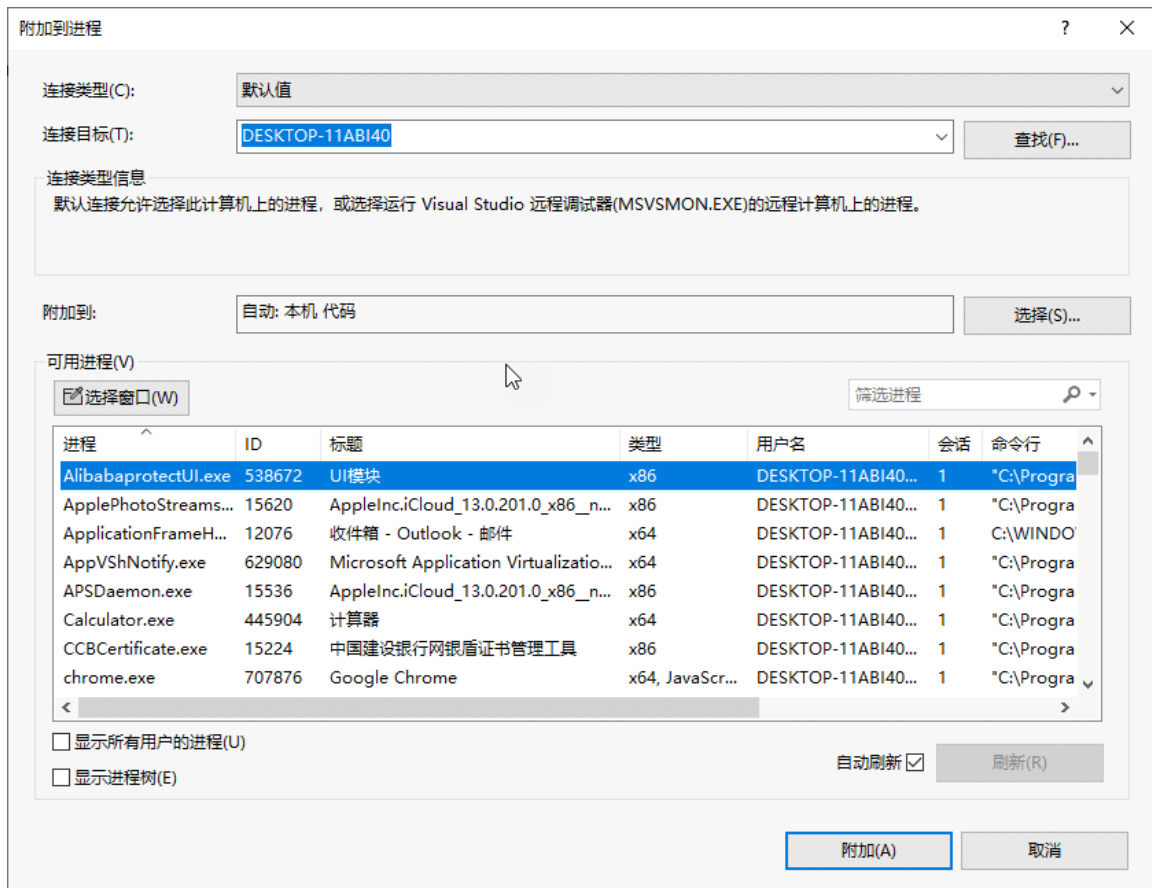


- 设置依赖断点
- 设置临时断点
- 断点标签
- 导入和导出断点
- "调用堆栈" 和 "反汇编" 调试器窗口设置断点
- 在 "断点" 窗口中管理断点

断点故障排除

- 红色实心 and 白色圆点
- 尚未为此文档加载任何符号
- 当前源代码与 中内置的版本不同
- 删除了断点, 启动继续命中该断点

附加到进程



在程序中启动调试器

`System.Diagnostics.Debugger.Launch();`

