[**C# 访问修饰符internal的访问范围误区释疑**](http://www.cnblogs.com/liuww/p/5335003.html)

**一、前言**

MSDN关于访问修饰符的访问级别解释：

访问修饰符是一些关键字，用于指定声明的成员或类型的可访问性。本节介绍四个访问修饰符：

* [public](https://msdn.microsoft.com/zh-cn/library/yzh058ae.aspx)
* [protected](https://msdn.microsoft.com/zh-cn/library/bcd5672a.aspx)
* [internal](https://msdn.microsoft.com/zh-cn/library/7c5ka91b.aspx)
* [private](https://msdn.microsoft.com/zh-cn/library/st6sy9xe.aspx)

使用这些访问修饰符可指定下列五个可访问性级别：

**public**：访问不受限制。

**protected**：访问仅限于包含类或从包含类派生的类型。

**Internal**：访问仅限于当前程序集。

[protected internal](https://msdn.microsoft.com/zh-cn/library/ms173121.aspx):访问限制到当前程序集或从包含派生的类型的类别。

**private**：访问仅限于包含类型。

**二、认知误区**

　　internal，英文含义是“内部的”，这时候基础不扎实的同学，可能就混淆了这个“内部的”的确切含义，到底是指“同一命名空间”的内部，还是“同一程序集”的内部，本人就是突然被问到这个问题的时候，真的就犹豫了，而且曾经一度以为就是“同一命名空间”的内部（话外：我们太多时候都自以为了，其实只要自己稍微MSDN查一下，就很清楚了）。

　　第一部分已经摘抄了微软MSDN官方的解释，其实这个内部就是“同一程序集”的内部，也就是说，internal修饰的方法或者属性，只要是在同一个程序集的中的其他类都可以访问，如果二者不在同一命名空间，只要使用using引用上相应的命名空间即可，这里，从另外一个方面也间接看出命名空间并不是界定访问级别的，而是保证全局的类唯一性的，下面就从现实生活中解释下internal的实际作用。

**三、释疑**

　　都说艺术源于生活，编程也是一门艺术，所以一样也是可以成生活中找到相应的场景，下面我们以实际生活场景来描述internal修饰符的作用。

　　某公司的某技术中心，发文规定即日起，中心内的“打印机”仅限本中心的各个部门使用，其他中心的人员不能使用。这个现实生活的场景中，技术中心其实就是个程序集，而每个部门相当于不同的类，当然部门可以有相应的标签，相当于不同的命名空间，其实就是进行逻辑划分，职责不同的部门属于不同的命名空间就可以理解了。而我们的“主角”打印机的旁边就会被贴上公告“本中心的打印机只能本中心使用。。。。”，此时就相当于给打印机打上了internal修饰符了。其他中心的同学们，再也用不上本中心的打印机了。

　　下面我们用代码来描述上面的场景：

1. 我们先在一个公共的类库中（程序集）定义一个打印机的类
   1. [复制代码](javascript:void(0);)
   2. 1 namespace CommonAsset
   3. 2 {
   4. 3 public class Printer
   5. 4 {
   6. 5 private bool \_isBad;
   7. 6 /// <summary>
   8. 7 /// 打印机是否坏了
   9. 8 /// </summary>
   10. 9 public bool IsBad
   11. 10 {
   12. 11 get { return \_isBad; }
   13. 12 set { \_isBad = value; }
   14. 13 }
   15. 14
   16. 15 public void Print()
   17. 16 {
   18. 17 Console.WriteLine("开始打印!");
   19. 18 }
   20. 19 }

20 }

[复制代码](javascript:void(0);)

1. 然后我们在创建一个中心A(程序集)，分别定义各个部门，且属于不同的命名空间

[复制代码](javascript:void(0);)

1 namespace CenterA.Administration

2 {

3 public class DepartmentB

4 {

5 internal Printer PrinterB { get; set; }

6 public DepartmentB()

7 {

8 PrinterB = new Printer();

9 }

10 }

11 }

12

13

14 namespace CenterA.Technology

15 {

16 public class DepartmentA

17 {

18 internal Printer PrinterA { get; set; }

19 public DepartmentA()

20 {

21 PrinterA = new Printer();

22 }

23

24 public void PrintSomething()

25 {

26 if (PrinterA.IsBad)

27 {

28 CenterA.Administration.DepartmentB departmentB = new CenterA.Administration.DepartmentB();

29 departmentB.PrinterB.Print();

30 }

31 else

32 {

33 PrinterA.Print();

34 }

35 }

36 }

37 }

[复制代码](javascript:void(0);)

1. 上面可以看见，DepartmentB和DepartmentA虽然不在同一个命名空间，但由于同属于CenterA这个程序集，DepartmentA是可以访问DepartmentB中的

Printer 的。