# ****using****

****1. using 关键字有两个主要用途：****

(一).作为指令，用于为命名空间创建别名或导入其他命名空间中定义的类型。

(二).作为语句，用于定义一个范围，在此范围的末尾将释放对象。

****2. using指令****

①允许在命名空间中使用类型，这样，您就不必在该命名空间中限定某个类型的使用：

using System.Text;

using PC.Company;

②为命名空间或类型创建别名。

using MyCompany = PC.Company; //命名空间的别名。

using Project = PC.Company.Project; //类型的别名

**using引入命名空间，并不等于编译器编译时加载该命名空间所在的程序集**，程序集的加载决定于程序中对该程序集是否存在调用操作，如果代码中不存在任何调用操作则编译器将不会加载using引入命名空间所在程序集。因此，在源文件开头，引入多个命名空间，并非加载多个程序集，不会造成“过度引用”的弊端。

创建别名的另一个重要的原因在于同一文件中引入的不同命名空间中包括了相同名称的类型，如SharpMap.Geometries.Point与System.Drawing.Point。为了避免出现名称冲突，可以通过设定别名来解决：

using SGPoint = SharpMap.Geometries.Point;

using SDPoint = System.Drawing.Point;

尽管我们可以通过类型全名称来加以区分，但是这显然不是最佳的解决方案。用using指令创建别名，有效的解决了这种可能的命名冲突，才是最佳的解决方案。

****3. using语句****

using 语句允许程序员指定使用资源的对象应当何时释放资源。using 语句中使用的对象（对象指是在using括号之中）必须实现 IDisposable 接口。此接口提供了 Dispose 方法，该方法将释放此对象的资源。

①可以在 using 语句之前声明对象。

　　 Font font2 = new Font("Arial", 10.0f);

　　 using (font2)

　　 {

　　 // use font2

　　 }

②可以在 using 语句之中声明对象。

　　 using (Font font2 = new Font("Arial", 10.0f))

　　 {

　　 // use font2

　　 }

③可以有多个对象与 using 语句一起使用，但是必须在 using 语句内部声明这些对象。

using (Font font3=new Font("Arial",10.0f), font4=new Font("Arial",10.0f))

　　 {

　　 // Use font3 and font4.

　　 }

4. 使用规则

①using只能用于实现了IDisposable接口的类型，禁止为不支持IDisposable接口的类型使用using语句，否则会出现编译错误；

　　②using语句适用于清理单个非托管资源的情况，而多个非托管对象的清理最好以try-finnaly来实现，因为嵌套的using语句可能存在隐藏的Bug。内层using块引发异常时，将不能释放外层using块的对象资源；

　　③using语句支持初始化多个变量，但前提是这些变量的**类型必须相同**，例如：

using(Pen p1 = new Pen(Brushes.Black), p2 = new Pen(Brushes.Blue))

　　 {

　　 //

　　 }

④针对初始化对个不同类型的变量时，可以都声明为IDisposable类型，例如：

using (IDisposable font = new Font("Verdana", 12), pen = new Pen(Brushes.Black))

　　 {

　　 float size = (font as Font).Size;

　　 Brush brush = (pen as Pen).Brush;

　　 }

5. using实质

在程序编译阶段，编译器会自动将using语句生成为try-finally语句，并在finally块中调用对象的Dispose方法，来清理资源。所以，**using语句等效于try-finally语句**，例如：

Font f2 = new Font("Arial", 10, FontStyle.Bold);

　　try

　　{

　　 //执行文本绘制操作

　　}

　　finally

　　{

　　 if (f2 != null) ((IDisposable)f2).Dispose();

　　}