如何定义一个接口：

使用interface关键字，然后指定名称和实例方法的签名，接口里的方法不可以是静态的不可以有修饰符

例子：

Ps：接口使用大写的“i”开头便于辨别

文本

中度可信度描述已自动生成

接口是“缩水版”的多继承，C#不支持多继承，通过接口提供类似的方案。CLR只允许显示实现接口，而C#语言允许在只继承一个接口时使用隐式实现，而继承多个接口时必须显示实现每一个方法。

能使用基类实例的地方一定能使用其派生类（因为派生类继承了基类的所有特征，也就是“多态”），接口类型和实现了该接口类型的类之间的关系也是如此

下面是例子：

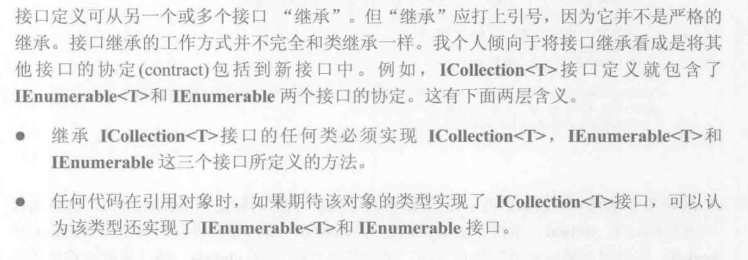
图形用户界面, 文本, 应用程序, 电子邮件

描述已自动生成

CLR认为接口定义就是类型定义，同样会为接口类型定义内部数据结构（类型对象指针和同步块索引）

接口无法继承类，只可以继承接口

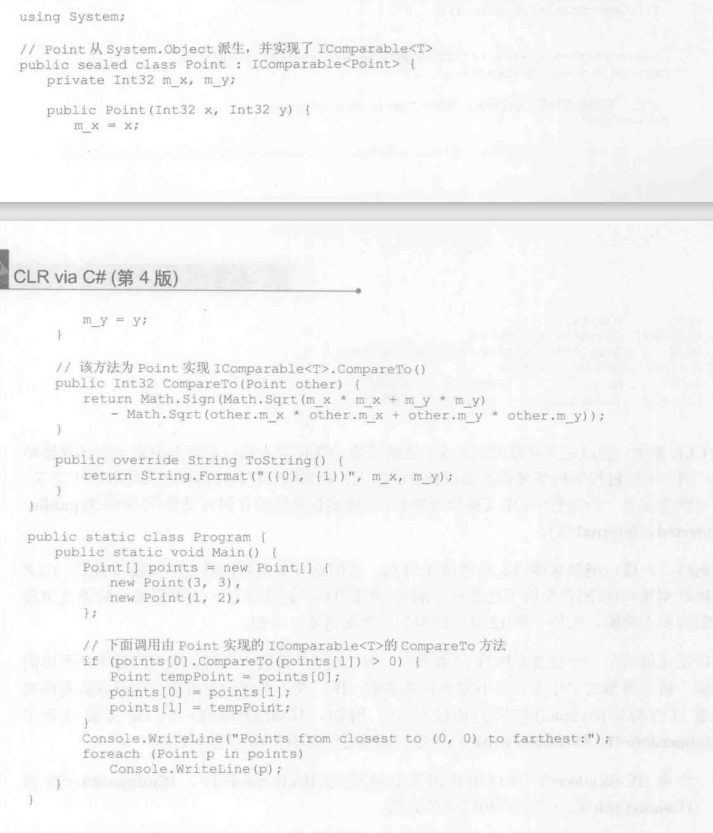
关于接口继承接口的例子（P261）



ICollection<T>继承了两个接口，继承了ICollection<T>接口的类型必须实现这三个接口里的所有方法

如何实现接口

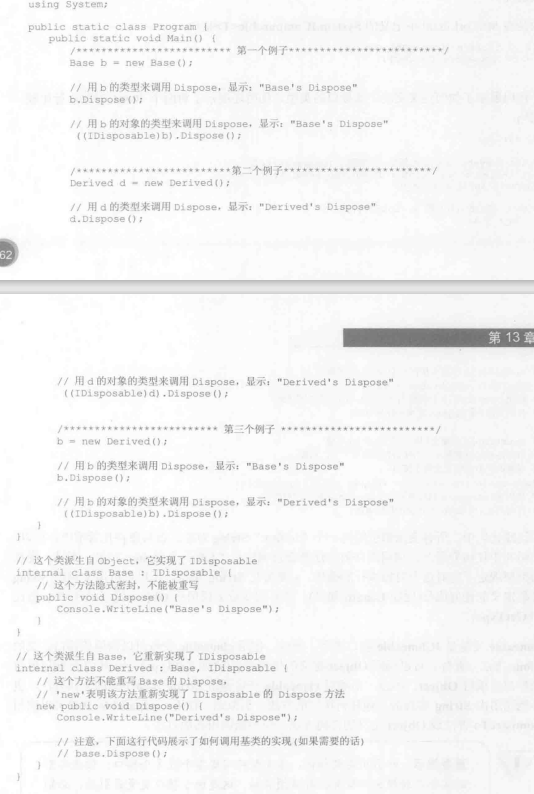
例子（P262）



Ps：实现接口的法要标记为public，CLR规定接口方法需要标记为virtual。方法可以标记为virtual，这样这个类的派生类可以重写这个接口方法；如果没有显示标记为virtual，编译器在编译时会加上sealed和virtual，这样子类就无法重写接口方法。

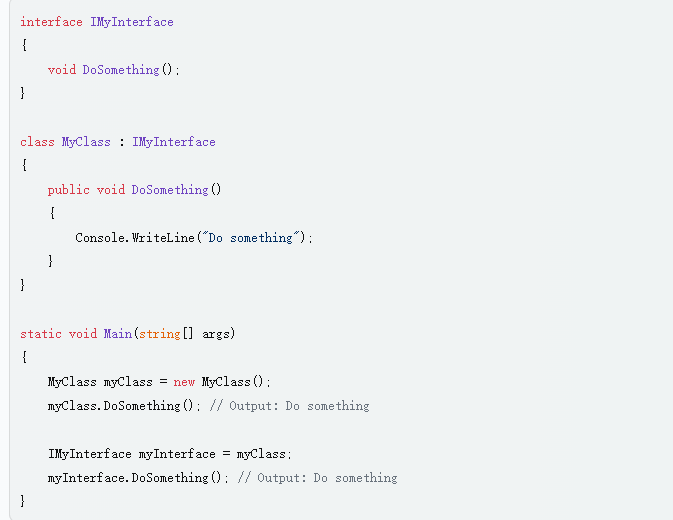
接口类型的额外用法：可以定义接口类型的字段、参数、局部变量

例子：（P263-264）



显式接口实现和隐式接口实现：隐式实现接口成员时，类中的成员方法与接口中定义的方法具有相同的名称和签名。对于隐式实现的成员，既可以通过类对象实例来访问，也可以通过接口来访问。

例子：



显式接口实现：显式实现接口成员时，类中的成员方法名称前面需要加上接口名称并用点号分隔。对于显式实现的成员，只能通过接口来访问，不能通过类对象实例来访问。

图形用户界面, 文本, 应用程序, 电子邮件

描述已自动生成

接口以及继承接口的类在元数据表方法表中的呈现：

P264：

对于这个类

文本

描述已自动生成

继承了IDisposable接口并且这个接口只包含Dispose方法

方法表中包含了哪些方法以及其类型相关的信息

文本, 电子邮件

描述已自动生成

使用泛型接口的好处：

1.泛型具有类型安全性

2.泛型可以有效避免装箱拆箱

3.类可以实现同一个接口若干次，而只要每次使用不同的类型参数（原本一个类只能实现一个接口一次）

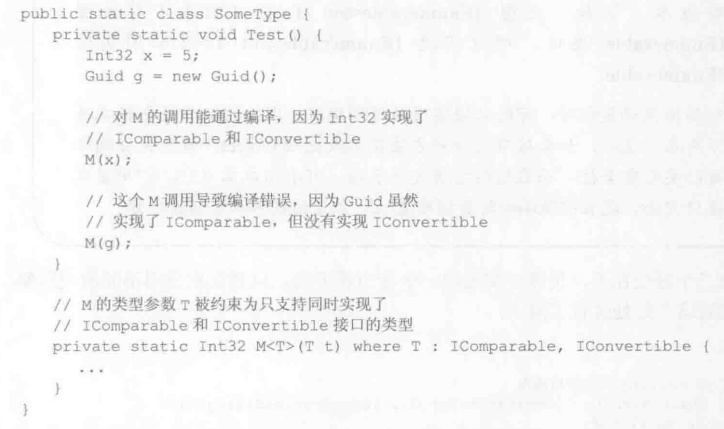
4.接口的泛型参数可标记为逆变和协变，为泛型接口的使用提供更大的灵活性

泛型类型参数约束为接口的好处：

1.可将泛型参数约束为多个接口。因此传递的类型参数的类型必须实现全部接口约束

2.减少装箱

例子：



如果一个类继承的接口中包含相同签名的方法，那么这些方法必须使用显式接口方法实现，调用也只能使用接口类型调用

例子：

文本

低可信度描述已自动生成

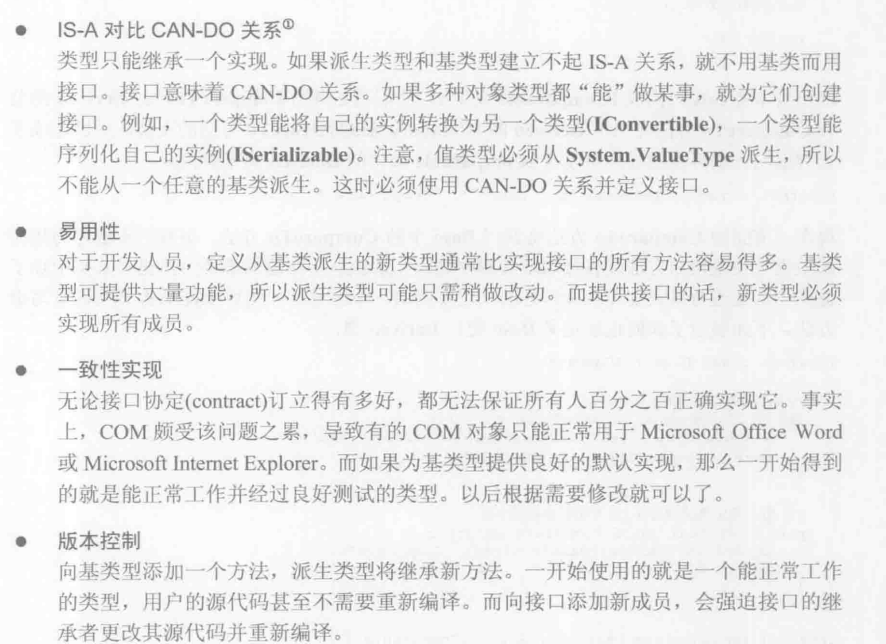
在不存在接口的泛型版本时，可以通过显式实现接口来确保编译时的类型安全性，同时避免装箱。但调用实现的方法时又必须使用接口类型变量，这样就又会出现装箱的情况同时失去了类型安全性（存在转换失败抛出异常的情况）。

显式实现接口的弊端：

信件

中度可信度描述已自动生成

如何选择，使用接口还是基类？



1.如果类型间共享的代码少，实现功能相差很大则使用接口。

2.当类型以及提供了较多的方法实现时则使用它作为基类，这样在后续添加代码时改动较小影响较小。（此时如果使用接口，那么需要重新实现很多次一样的功能）