C# 运算符重载 C# 命名空间（Namespace）

C# 接口（Interface）

接口定义了所有类继承接口时应遵循的语法合同。接口定义了语法合同 **"是什么"** 部分，派生类定义了语法合同 **"怎么做"** 部分。

接口定义了属性、方法和事件，这些都是接口的成员。接口只包含了成员的声明。成员的定义是派生类的责任。接口提供了派生类应遵循的标准结构。

接口使得实现接口的类或结构在形式上保持一致。

抽象类在某种程度上与接口类似，但是，它们大多只是用在当只有少数方法由基类声明由派生类实现时。

定义接口: MyInterface.cs

接口使用 **interface** 关键字声明，它与类的声明类似。接口声明默认是 public 的。下面是一个接口声明的实例：

```
interface IMyInterface
{
    void MethodToImplement();
}
```

以上代码定义了接口 IMyInterface。通常接口命令以 `I` 字母开头，这个接口只有一个方法 MethodToImplement()，没有参数和返回值，当然我们可以按照需求设置参数和返回值。

值得注意的是，该方法并没有具体的实现。

接下来我们来实现以上接口：InterfaceImplementer.cs

实例

```
using System;

interface IMyInterface
{
        // 接口成员
    void MethodToImplement();
}

class InterfaceImplementer : IMyInterface
{
    static void Main()
    {
        InterfaceImplementer iImp = new InterfaceImplementer();
        iImp.MethodToImplement();
    }

    public void MethodToImplement()
    {
        Console.WriteLine("MethodToImplement() called.");
```

```
    }
}
```

InterfaceImplementer 类实现了 IMyInterface 接口，接口的实现与类的继承语法格式类似：

```
class InterfaceImplementer : IMyInterface
```

继承接口后，我们需要实现接口的方法 MethodToImplement()，方法名必须与接口定义的方法名一致。

接口继承: InterfaceInheritance.cs

以下实例定义了两个接口 IMyInterface 和 IParentInterface。

如果一个接口继承其他接口，那么实现类或结构就需要实现所有接口的成员。

以下实例 IMyInterface 继承了 IParentInterface 接口，因此接口实现类必须实现 MethodToImplement() 和 ParentInterfaceMethod() 方法：

实例

```
using System;

interface IParentInterface
{
    void ParentInterfaceMethod();
}

interface IMyInterface : IParentInterface
{
    void MethodToImplement();
}

class InterfaceImplementer : IMyInterface
{
    static void Main()
    {
        InterfaceImplementer iImp = new InterfaceImplementer();
        iImp.MethodToImplement();
        iImp.ParentInterfaceMethod();
    }

    public void MethodToImplement()
    {
        Console.WriteLine("MethodToImplement() called.");
    }

    public void ParentInterfaceMethod()
    {
        Console.WriteLine("ParentInterfaceMethod() called.");
    }
}
```

实例输出结果为：

```
MethodToImplement() called.
ParentInterfaceMethod() called.
```

C# 运算符重载 C# 命名空间（Namespace）

2 篇笔记

写笔记

接口注意的几点：

1. 接口方法不能用public abstract等修饰。接口内不能有字段变量，构造函数。
2. 接口内可以定义属性（有get和set的方法）。如string color { get ; set ; }这种。
3. 实现接口时，必须和接口的格式一致。
4. 必须实现接口的所有方法。

樱花树下约定 9个月前 (06-07)

1.接口的特点

接口的定义是指定一组函数成员而不实现成员的引用类型，其它类型和接口可以继承接口。定义还是很好理解的，但是没有反映特点，接口主要有以下特点：

- (1)通过接口可以实现多重继承，C# 接口的成员不能有 public、protected、internal、private 等修饰符。原因很简单，接口里面的方法都需要由外面接口实现去实现方法体，那么其修饰符必然是 public。C# 接口中的成员默认是 public 的，java 中是可以加 public 的。
- (2)接口成员不能有 new、static、abstract、override、virtual 修饰符。有一点要注意，当一个接口实现一个接口，这2个接口中有相同的方法时，可用 new 关键字隐藏父接口中的方法。
- (3)接口中只包含成员的签名，接口没有构造函数，所有不能直接使用 new 对接口进行实例化。接口中只能包含方法、属性、事件和索引的组合。接口一旦被实现，实现类必须实现接口中的所有成员，除非实现类本身是抽象类。
- (4)C# 是单继承，接口是解决 C# 里面类可以同时继承多个基类的问题。

2.接口的简单使用

```
class Program
{
    static void Main(string[] args)
    {
        IWorker james1 = new James1();
        IWorker james2 = new James2();
        james1.work("设计");
        james2.work("编程");
        //从这个例子我体会到了有接口的好处，可以想象如果又来了新的员工。
        //如果不采用接口，而是每个员工都有一个单独的类，这样就会容易出错。
        //如果有接口这种协议约束的话，那么只要实现了接口就肯定有接口里声明的方法，我们只需拿来调
用。
    }
```

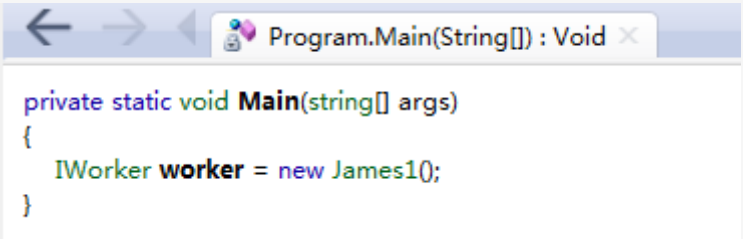
```
}
public interface IWorker{ void work(string s); }
class James1 : IWorker
{
    public void work(string s)
    {
        Console.WriteLine("我的名字是James1，我的工作是" +s);
    }
}
class James2 : IWorker
{
    public void work(string s)
    {
        Console.WriteLine("我的名字是James2，我的工作是"+s);
    }
}
```

3.一个可以实例化接口的特例

```
class Program
{
    static void Main(string[] args)
    {
        //C#中COM接口是可以实例化的，但其实这种写法是使接口“映射”到某一个类上，实际上创建的是这个类的实例。
        IWorker worker = new IWorker();
    }
}
[ComImport, CoClass(typeof(James1))]
[Guid("d60908eb-fd5a-4d3c-9392-8646fcd1edce")]
public interface IWorker{ void work(string s); }
//ComImport特性发生在tlbimp.exe导入COM类型类库的时候，生成的托管类型会标记有ComImport特性
//Guid特性是一个GUID标识，COM类型是用GUID来标识的。
```

利用 .NET Reflector 查看时可以很明显的看到 Main 方法里面的代码是：

```
IWorker worker=new James1();
```



4.接口和抽象类的区别

接口用于规范，抽象类用于共性。抽象类是类，所以只能被单继承，但是接口却可以一次实现多个。

接口中只能声明方法，属性，事件，索引器。而抽象类中可以有方法的实现，也可以定义非静态的类变量。

抽象类可以提供某些方法的部分实现，接口不可以。抽象类的实例是它的子类给出的。接口的实例是实现接口的类给出的。

在抽象类中加入一个方法，那么它的子类就同时有了这个方法。而在接口中加入新的方法，那么实现它的类就要重新编写（这就是为什么说接口是一个类的规范了）。

接口成员被定义为公共的，但抽象类的成员也可以是私有的、受保护的、内部的或受保护的内部成员（其中受保护的内部成员只能在应用程序的代码或派生类中访问）。

此外接口不能包含字段、构造函数、析构函数、静态成员或常量。

还有一点，我们在VS中实现接口时会发现有2个选项，一个是实现接口，一个是显示实现接口。实现接口就是我们平常理解的实现接口，而显示实现接口的话，实现的方法是属于接口的，而不是属于实现类的。

Lost丶先生 8个月前 (07-20)