

C# 匿名方法

我们已经提到过，委托是用于引用与其具有相同标签的方法。换句话说，您可以使用委托对象调用可由委托引用的方法。

匿名方法（Anonymous methods）提供了一种传递代码块作为委托参数的技术。匿名方法是没有名称只有主体的方法。

在匿名方法中您不需要指定返回类型，它是从方法主体内的 `return` 语句推断的。

编写匿名方法的语法

匿名方法是通过使用 `delegate` 关键字创建委托实例来声明的。例如：

```
delegate void NumberChanger(int n);  
...  
NumberChanger nc = delegate(int x)  
{  
    Console.WriteLine("Anonymous Method: {0}", x);  
};
```

代码块 `Console.WriteLine("Anonymous Method: {0}", x);` 是匿名方法的主体。

委托可以通过匿名方法调用，也可以通过命名方法调用，即，通过向委托对象传递方法参数。

例如：

```
nc(10);
```

实例

```
class TTT {  
    public delegate void  
TestDelegate1();  
    public delegate void  
TestDelegate2(int a);  
    public delegate int  
TestDelegate(int a, int b);
```

```

void TEE()
{
    TestDelegate testA = (int a,
int b) =>
    {
        return a + b;
    };

    //简化： 如果一个Lambda的方法中
    只有一个返回值，那么大括号的return可以省略
    testA = (int a, int b) => a +
b;

    //简化2： 在Lambda的参数列表中
    参数类型可以省略
    testA = (a, b) => a + b;
    TestDelegate1 test2 = () => {
};

}

```

下面的实例演示了匿名方法的概念：

实例

```
using System;

delegate void NumberChanger(int n);
namespace DelegateAppl
{
    class TestDelegate
    {
        static int num = 10;
        public static void AddNum(int p)
        {
            num += p;
            Console.WriteLine("Named Method: {0}", num);
        }

        public static void MultNum(int q)
        {
            num *= q;
            Console.WriteLine("Named Method: {0}", num);
        }

        static void Main(string[] args)
        {
            // 使用匿名方法创建委托实例
            NumberChanger nc = delegate(int x)
            {
                Console.WriteLine("Anonymous Method: {0}", x);
            };

            // 使用匿名方法调用委托
            nc(10);

            // 使用命名方法实例化委托
            nc = new NumberChanger(AddNum);

            // 使用命名方法调用委托
            nc(5);
        }
    }
}
```

```
// 使用另一个命名方法实例化委托
nc = new NumberChanger(MultNum);

// 使用命名方法调用委托
nc(2);
Console.ReadKey();
}
}
}
```

当上面的代码被编译和执行时，它会产生下列结果：

```
Anonymous Method: 10
Named Method: 15
Named Method: 30
```