匿名类型

匿名类型关键字var
var关键字，可以把它理解为类型占位符，它并不是C#中的一种基本类型。它能在你编写程序的时候，自动计算出等式右边的类型，而且支持智能感知。

```
//可以是整型
var vInt = 123;
//可以是字符串
var vString = "Hello Var!";
//可以是日期类型
var vDateTime = new DateTime();
//甚至可以是一个数组
var vIntArray = new int[] { 1, 2, 3, 4, 5, 6 };
```

但需要注意的是，一旦指明类型之后，不能再更改它的类型了，否则将报错。

```
//将它指定为整型
var vInt = 123;
//这样就不行了，第一次指定类型之后，vInt就被指定为int类型了，而不能将string值为它赋值
vInt = "string";
```

注意事项
匿名类型必须初始化，且不能被初始化为null。
但是却可以这样初始化null:

```
var vString = (String)null;
```

其实等价于下面这段代码：

```
String vString = null;
```

匿名类型可以用于简单类型，也可以用于复杂类型。定义复合匿名类型时需要有成员声明。

```
//Name和Age就是一个成员声明
var student = new { Name = "James", Age = 27 };
```

匿名类型不能定义为类成员变量。
方法可以返回匿名类型。

```
static object getStudent()
{
    var student = new { Name = "James", Age = 27 };
    return student;
}
```

但是调用起来就比较麻烦了

```
static void Main(String[] args)
{
     Object student = getStudent();
     Console.WriteLine("Name:{0}\nAge:{1}",
         student.GetType().GetProperty("Name").GetValue(student,null),
         student.GetType().GetProperty("Age").GetValue(student, null));
}
```

一些匿名类型的使用技巧

为匿名类型增加一个方法

我们不能直接在匿名类型中这样做，但可以通过一些间接的手段。

```
static void Main(String[] args)
{
    Action<String> printName = delegate(String name)
    {
        Console.WriteLine("Welcome {0}",name);
    };
     var customer = new { Name = "James", WelcomeInfo = printName };
     //智能感知已经发现了这个方法，我们可以像平常那样去调用了
    customer.WelcomeInfo(customer.Name);
}
```

匿名类型也能进行数据绑定

```
private void Form1_Load(object sender, EventArgs e)
{
    UpdateDate();
}
private void UpdateDate()
{
    var student1 = new { Name = "Liu", Age = 14 };
    var student2 = new { Name = "Guan", Age = 12 };
    var student3 = new { Name = "Zhang", Age = 13 };
    var students = new List<object> { student1, student2, student3 };
    dataGridView1.DataSource = students;
}
```

匿名类型相等性测试

```
static void Main(String[] args)
{
    var student1 = new { Name = "Liu", Age = 14 };
    var student2 = new { Name = "Liu", Age = 14 };
    var student3 = new { Name = "Zhang", Age = 14 };
    var student4 = new { Age = 14, Name = "Liu" };
    Console.WriteLine(student1.Equals(student2));
    Console.WriteLine(student1.Equals(student3));
    Console.WriteLine(student1.Equals(student4));
    //输出结果 true , false , false
}
```

使用匿名方法递归

```
static void Main(String[] args)
{
    Func<long, long> factorial = delegate(long n)
    {
        return n > 1 ? n * (long)(new StackTrace().GetFrame(0).GetMethod()
                        .Invoke(null, new object[] { n - 1 })) : n;
    };

    Console.WriteLine(factorial(5));
    Console.ReadKey();
}
```