**委托和Lambda表达式总结**

C#委托，将**方法作为对象**封装起来，允许在运行时间接地绑定一个方法调用。

委托是一个应用类型，不必使用new来实例化，直接传递名称，而不是显示的实例化——委托推断

1. 声明委托数据类型：

|  |
| --- |
| public delegate bool ComparisonHandler(int first, int second); |

1. 声明一个嵌套的委托数据类型：

|  |
| --- |
| class DelegateSample  {  public delegate bool ComparisonHandler(int first, int second);  } |

上面声明的委托数据类型是DelegateSample.ComparisonHandler，被定义成嵌套在DelegateSample中的一个类型。

1. 委托实例化

|  |
| --- |
| public static bool GreaterThan(int first, int second)  {  return first > second;  } |

1. 将委托实例作为参数传递

|  |
| --- |
| public static void BubbleSort(int[] items, ComparisonHandler comparisonMethod)  { …… }  BubbleSort(items, GreaterThan);//C#2.0以上调用方式  BubbleSort(items, new ComparisonHandler(GreaterThan));//C#2.0之前 |

要改变BubbleSort的排序方式只需要改变委托实例化方法

1. 匿名方法

匿名方法就是没有实际方法声明的委托实例，或者说定义是直接内嵌在代码中。

|  |
| --- |
| ComparisonHandler comparisonMethod;//使用的嵌套委托  comparisonMethod =delegate(int first, int second)  {  return first < second;  }; |
| BubbleSort(items, comparisonMethod); |

1. 使用匿名方法而不声明变量，去掉委托声明，直接在函数体中使用delegate关键字

|  |
| --- |
| BubbleSort(items,  delegate(int first, int second)  {  return first < second;  }  ); |

1. 高级主题：无参数的匿名方法

|  |
| --- |
| delegate { return Console.ReaLine()!=”” } |

委托时可以改变的，目前还不需要研究

Lambda表达式，C#3.0中引入Lambda表达式是比匿名方法更加简洁的一种**匿名函数语法**。这里的匿名函数泛指Lambda表达式和匿名方法。Lambda表达式本身划分为两种类型：语句Lambda和表达式Lambda。（总体来说Lambda表达式是委托-匿名方法的升级版）

1. **语句Lambda**。是C#3.0为匿名方法提供的一种简化的语法。这种与语法不包含delegate关键字，但添加了运算符=>。

代码清单对比：（左侧使用匿名方法，右侧使用语句Lambda）

|  |  |
| --- | --- |
| BubbleSort(items,  delegate(int first, int second)  {  return first < second;  }  ); | BubbleSort(items,  (int first, int second) =>  {  return first < second;  }  ); |

1).语句Labmda中省略参数类型（编译器能推断输出参数类型，或者能将参数类型隐式转换成期望的参数类型）。

|  |
| --- |
| BubbleSort(items,  (first, second) =>  {  return first < second;  }  ); |

2).无参数语句Lambda

|  |
| --- |
| Func<string> getUserInput =  () =>  {  string input;  do  {  input = Console.ReadLine();  } while (input.Trim().Length == 0);  return input;  }; |

当编译器能够推断出数据类型，而且只有一个输入参数的时候，语句Lambda可以不带圆括号()

3).只有一个输入参数的语句Lambda

1. **表达式Lambda。**语句Lambda含有一个语句块，可以包含零个或者更多的语句，表达式Lambda只有一个表达式，没有语句块。

|  |
| --- |
| BubbleSort( items, (first, second) => first < second ); |

语句Lambda与表达式Lambda的区别：后者没有return语句及大括号（没有表达式）。

注意：语句Lambda和表达式Lambda是一个断言（返回一个bool型的值）。

Lambda运算符理解成“满足……条件”（更多适用于表达式Lambda），上述例子可以理解成“first和second满足first小于second的条件”。

不可对匿名方法使用typeof()运算符。只有将一个匿名方法赋给委托变量或者转换成一个委托变量之后，才可以调用GetType()

1. **Linq语句。**System.Linq.Enumerable()中的Where()函数

|  |
| --- |
| names.Where( name => name.Contains(“ ”)) ; |

List names中包含空格的name