C# 属性（ Property ） C# 委托（ Delegate ）

C# 索引器（ Indexer ）

索引器（ Indexer ） 允许一个对象可以像数组一样被索引。当您为类定义一个索引器时，该类的行为就会像一个 **虚拟数组（ virtual array ）** 一样。您可以使用数组访问运算符（ [ ] ）来访问该类的实例。

语法

一维索引器的语法如下：

```
element-type this[int index]
{
   // get 访问器
   get
   {
      // 返回 index 指定的值
   }

   // set 访问器
   set
   {
      // 设置 index 指定的值
   }
}
```

索引器（ Indexer ）的用途

索引器的行为的声明在某种程度上类似于属性（ property ）。就像属性（ property ），您可使用 **get** 和 **set** 访问器来定义索引器。但是，属性返回或设置一个特定的数据成员，而索引器返回或设置对象实例的一个特定值。换句话说，它把实例数据分为更小的部分，并索引每个部分，获取或设置每个部分。

定义一个属性（ property ）包括提供属性名称。索引器定义的时候不带有名称，但带有 **this** 关键字，它指向对象实例。下面的实例演示了这个概念：

实例

```
using System;
namespace IndexerApplication
{
   class IndexedNames
   {
      private string[] namelist = new string[size];
      static public int size = 10;
      public IndexedNames()
      {
         for (int i = 0; i < size; i++)
         namelist[i] = "N. A.";
      }
```

```
      public string this[int index]
      {
         get
         {
            string tmp;

            if( index >= 0 && index <= size-1 )
            {
               tmp = namelist[index];
            }
            else
            {
               tmp = "";
            }

            return ( tmp );
         }
         set
         {
            if( index >= 0 && index <= size-1 )
            {
               namelist[index] = value;
            }
         }
      }

      static void Main(string[] args)
      {
         IndexedNames names = new IndexedNames();
         names[0] = "Zara";
         names[1] = "Riz";
         names[2] = "Nuha";
         names[3] = "Asif";
         names[4] = "Davinder";
         names[5] = "Sunil";
         names[6] = "Rubic";
         for ( int i = 0; i < IndexedNames.size; i++ )
         {
            Console.WriteLine(names[i]);
         }
         Console.ReadKey();
      }
   }
}
```

当上面的代码被编译和执行时，它会产生下列结果：

```
Zara
Riz
Nuha
Asif
Davinder
Sunil
```

```
Rubic
N. A.
N. A.
N. A.
```

重载索引器（Indexer）

索引器（Indexer）可被重载。索引器声明的时候也可带有多个参数，且每个参数可以是不同的类型。没有必要让索引器必须是整型的。C# 允许索引器可以是其他类型，例如，字符串类型。

下面的实例演示了重载索引器：

实例

```
using System;
namespace IndexerApplication
{
   class IndexedNames
   {
      private string[] namelist = new string[size];
      static public int size = 10;
      public IndexedNames()
      {
         for (int i = 0; i < size; i++)
         {
          namelist[i] = "N. A.";
         }
      }
      public string this[int index]
      {
         get
         {
            string tmp;

            if( index >= 0 && index <= size-1 )
            {
               tmp = namelist[index];
            }
            else
            {
               tmp = "";
            }

            return ( tmp );
         }
         set
         {
            if( index >= 0 && index <= size-1 )
            {
               namelist[index] = value;
            }
         }
      }
```

```
      public int this[string name]
      {
         get
         {
            int index = 0;
            while(index < size)
            {
               if (namelist[index] == name)
               {
                return index;
               }
               index++;
            }
            return index;
         }

      }

      static void Main(string[] args)
      {
         IndexedNames names = new IndexedNames();
         names[0] = "Zara";
         names[1] = "Riz";
         names[2] = "Nuha";
         names[3] = "Asif";
         names[4] = "Davinder";
         names[5] = "Sunil";
         names[6] = "Rubic";
         // 使用带有 int 参数的第一个索引器
         for (int i = 0; i < IndexedNames.size; i++)
         {
            Console.WriteLine(names[i]);
         }
         // 使用带有 string 参数的第二个索引器
         Console.WriteLine(names["Nuha"]);
         Console.ReadKey();
      }
   }
}
```

当上面的代码被编译和执行时，它会产生下列结果：

```
Zara
Riz
Nuha
Asif
Davinder
Sunil
Rubic
N. A.
N. A.
```

```
N. A.
2
```

C# 属性（ Property ） C# 委托（ Delegate ）

2 篇笔记

写笔记

也可在 class 实例化时自定义索引器 size。

实例：

```
public class IndexedNames
{
  private string[] nameList;
  public int size;
  public IndexedNames(int size)
  {
    this.size = size;
    nameList = new string[size];
    for(int i = 0;i < size; i++)
    {
      nameList[i]="N. A.";
    }
  }
  publis string this[int index]
  {
    get
    {
      string temp;
      if(index>=0 && index<size)
      {
        temp = nameList[index];
      }
      else
      {
        temp = "";
      }
      return temp;
    }
    set
    {
      if(index>=0 && index<size)
      {
        nameList[index] = value;
      }
    }
```

```
  }
}
static void Main(string[] args)
{
   IndexedNames index = new IndexedNames(5);
   index[0] = "Zara";
   index[1] = "Riz";
   index[2] = "Nuha";
   index[3] = "Asif";
   index[4] = "Davinder";
   for (int i = 0; i < index.size; i++)
   {
      Console.WriteLine(index[i]);
   }
   Console.ReadKey();
}
```

Yorkove 8个月前 (07-24)

```
using System;

namespace IndexerApplication{
    //1.类成员变量是数组的情况
    // class IndexedNames{

        // private string[] namelist = new string[size];
        // static public int size = 10;

        // public IndexedNames(){

            // for(int i = 0; i < size; i++){
                // namelist[i] = "N.A.";
            // }
        // }

        // public string this[int index]{
            // get{

                // string tmp;
                // if(index >=0 && index <=size-1){
                    // tmp = namelist[index];
                // }else{
                    // tmp="";
                // }
                // return tmp;
            // }
            // set{
                // if(index >=0 && index <=size-1){
```

```
                    // namelist[index] = value;
                // }
            // }
        // }

        // static void Main(string[] args){

            // IndexedNames names = new IndexedNames();
            // names[0] = "Zara";
            // names[1] = "Riz";
            // names[2] = "Nuha";
            // names[3] = "Asif";
            // names[4] = "Davinder";
            // names[5] = "Sunil";
            // names[6] = "Rubic";

            // for(int i = 0; i < IndexedNames.size; i++){
                // Console.WriteLine(names[i]);
            // }
        // }
    // }

        //2.类成员变量是多个基本类型
    class Employee{
        public string firstName;
        public string middleName;
        public string lastName;

        public string this[string index]{
            set{
                switch(index){
                    case "a":firstName = value;
                        break;
                    case "b":middleName = value;
                        break;
                    case "c":lastName = value;
                        break;
                    default: throw new ArgumentOutOfRangeException("index");
                }
            }
            get{
                switch(index){
                    case "a":return firstName;
                    case "b":return middleName;
                    case "c":return lastName;
                    default: throw new ArgumentOutOfRangeException("index");
                }
            }
        }
```

```
        static void Main(string[] args){
            Employee ee = new Employee();

            ee.firstName = "胡";
            ee.middleName = "大";
            ee.lastName = "阳";

            Console.WriteLine("我的名字叫: {0}{1}{2}",ee["a"],ee["b"],ee["c"]);
        }
    }
}
```

二狗子 3个月前 (12-06)