|  |
| --- |
| using System;  using System.Collections.Generic;  using System.Linq;  using System.Text;  using System.Threading.Tasks;    namespace AppConsole  {        /// <summary>      /// 普通用户      /// </summary>      public class User      {          /// <summary>          /// 全局变量          /// </summary>          Dictionary<string, object> dictInfo = null;            /// <summary>          /// 构造器          /// </summary>          public User()          {              dictInfo = new Dictionary<string, object>();          }            /// <summary>          /// 构造函数重载          /// </summary>          /// <param name="UserID"></param>          /// <param name="UserName"></param>          public User(int UserID, string UserName)          {              this.UserName = UserName;              this.ID = UserID;          }            /// <summary>          /// this，第【1】种用法，索引器          /// </summary>          /// <param name="name"></param>          /// <returns></returns>          public object this[string name]          {              get { return dictInfo[name]; }              set { dictInfo[name] = value; }          }              /// <summary>          /// 编号          /// </summary>          public int ID { get; set; }              /// <summary>          /// 用户名          /// </summary>          public string UserName { get; set; }            /// <summary>          /// this第【2】种用法，当做参数传递          /// </summary>          public void Said()          {              new VIP().Say(this);          }        }        /// <summary>      /// 会员      /// </summary>      public class VIP : User      {              /// <summary>          /// 积分          /// </summary>          public int integral { get; set; }            /// <summary>          /// 构造函数          /// </summary>          public VIP()          {              ID = 520;              integral = 1000;          }            /// <summary>          /// this第【3】种用法，通过this()调用无参构造函数          /// </summary>          /// <param name="UserName"></param>          public VIP(string UserName)              : this()          {              this.UserName = UserName;          }            /// <summary>          /// 构造函数重载          /// </summary>          /// <param name="UserID"></param>          /// <param name="UserName"></param>          public VIP(int UserID, string UserName)              : base(UserID, UserName)          {            }            /// <summary>          ///Say方法          /// </summary>          /// <param name="user"></param>          public void Say([LCQAttribute] User user)          {              Console.WriteLine(string.Format("嗨，大家好！我的编号是{0}，大家可以叫我{1}！", user.ID, user.UserName));          }        }        /// <summary>      /// 静态类，来扩展User类      /// </summary>      public static class Helper      {            /// <summary>          /// 第【4】种用法： this扩展User类          /// </summary>          /// <param name="user"></param>          public static void Sing(this User user)          {              Console.WriteLine(string.Format("嗨，大家好！我的编号是{0}，大家可以叫我{1}！", user.ID, user.UserName));          }      }        /// <summary>      /// 特性类：指定特性仅适用于方法和方法的参数      /// </summary>      [System.AttributeUsage(AttributeTargets.Method | AttributeTargets.Parameter)]      public class LCQAttribute : System.Attribute      {        }  } |

### 2. [代码]**this第【1】种用法，索引器**

[?](https://www.oschina.net/code/snippet_584165_48374)

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15 | /// <summary>  /// 全局变量  /// </summary>  Dictionary<string, object> dictInfo = null;    /// <summary>  /// this，第【1】种用法，索引器  /// </summary>  /// <param name="name"></param>  /// <returns></returns>  public object this[string name]  {      get { return dictInfo[name]; }      set { dictInfo[name] = value; }  } |

### 3. [代码]**this第【2】种用法，当做参数传递**

[?](https://www.oschina.net/code/snippet_584165_48374)

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7 | /// <summary>  /// this第【2】种用法，当做参数传递  /// </summary>  public void Said()  {      new VIP().Say(this);  } |

### 4. [代码]**this第【3】种用法，通过this()调用无参构造函数**

[?](https://www.oschina.net/code/snippet_584165_48374)

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9 | /// <summary>  /// this第【3】种用法，通过this()调用无参构造函数  /// </summary>  /// <param name="UserName"></param>  public VIP(string UserName)      : this()  {      this.UserName = UserName;  } |

### 5. [代码]**this第【4】种用法：扩展User类**

[?](https://www.oschina.net/code/snippet_584165_48374)

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15 | /// <summary>  /// 静态类，来扩展User类  /// </summary>  public static class Helper  {        /// <summary>      /// 第【4】种用法： this扩展User类      /// </summary>      /// <param name="user"></param>      public static void Sing(this User user)      {          Console.WriteLine(string.Format("嗨，大家好！我的编号是{0}，大家可以叫我{1}！", user.ID, user.UserName));      }  } |

### 6. [代码]**最后，控制台测试**

[?](https://www.oschina.net/code/snippet_584165_48374)

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33 | /// <summary>  /// 主程序入口  /// </summary>  /// <param name="args"></param>  static void Main(string[] args)  {        //0>声明实体      User user = new User();      user.ID = 1;      user.UserName = "lichaoqiang";        //第【一】种用法:this用作索引器 public object this[string name]{……}      user["UserID"] = 1;      Console.WriteLine("第【一】种用法:this用作索引器");        //第【二】种用法:this用作参数传递 user.Say(this);      Console.WriteLine("第【二】种用法:this用作参数传递");      user.Said();        //第【三】种用法:this() public VIP:this(){   }      VIP vip = new VIP("yezi");      vip.Said();      Console.WriteLine("第【三】种用法:this()");        //第【四】种用法： this扩展VIP类 public static Sing(this User user){……}      Console.WriteLine("第【四】种用法： this扩展VIP类");      user.Sing();          Console.Read();    } |