namespace ConsoleApplication2

{

class Cat

{

private static string home; //静态成员变量

private string name;

/\*静态构造器，初始化静态成员变量\*/

static Cat()

{

Cat.Home = "MiMi's HOME";

}

public Cat()

{

this.name = "LiuLangMao";

}

public Cat(string name)

{

this.name = name;

}

/\*访问静态成员变量\*/

public static string Home

{

get

{

return home;

}

set

{

Cat.home = value;

}

}

public string Name

{

get

{

return name;

}

set

{

name = value;

}

}

}

/\*派生类无静态变量时\*/

class BoSiCat : Cat

{

static BoSiCat()

{

//Cat.Home = "MiaoMiao's HOME";

}

}

/\*派生类有静态变量时\*/

class BoSiCat : Cat

{

public string BoSiCatHome;

static BoSiCat()

{

BoSiCat.BoSiCatHomeHome = Cat.Home;

}

}

class Program

{

static void Main(string[] args)

{

BoSiCat bs = new BoSiCat();

Console.WriteLine(Cat.Home);

Console.WriteLine(Cat.Home);

}

}

}

1，所有的实例成员变量和static变量，都声明成private

2，定义实例成员变量和static变量时，不用=初始化任何变量

3，为所有的实例成员变量和static变量提供属性

4，static变量用static构造函数初始化，实例成员变量用构造函数初始化，构造函数必须调用属性来赋值

5，static构造函数只初始化static变量，构造函数只初始化实例成员变量，互不牵涉

6，无参构造函数，不需要传参，在函数内直接初始化成员变量

7，static构造函数，禁止有参数和修饰符，直接初始化static变量

8，派生类的无参实例构造函数，函数内部只初始化派生类自己定义的成员变量，系统会自动调用基类的无参构造函数

9，派生类的构造函数，函数的形参包含自己定义的变量的形参，以及基类的变量的形参，同时显示调用基类的有参构造函数

public PaiSheng( 派生类的形参，基类的形参) : base(基类的形参) {派生类成员变量 = 派生类形参}

11，派生类自己没有新增static变量，就不要提供static构造函数，即使提供，不要在static构造函数内操作基类的static成员变量。

12，可认为派生类不会继承和触碰到基类的static变量，如果派生类需要基类那样的static变量，可以自己重新定义，在static构造函数中，派生类.静态变量 = 基类.静态变量。

静态成员变量理论上都要声明成public，除非为了封装一下，提供public的set和get方法，才能声明成private。

派生类会继承基类的static成员，但是派生类和基类同样是共享一份。

static函数没有“虚函数”一说。因为static函数实际上是“加上了访问控制的全局函数”，全局函数哪来的什么虚函数？

静态构造函数是用来初始化静态成员变量的。

静态构造函数会在第一次使用静态变量或第一次New该类的实例的时候被调用，初始化静态成员变量。 任何类的静态构造函数只可能运行一次。

派生类的静态构造函数会先运行基类的静态构造函数，再运行自己的。

静态构造函数不能被显示调用，不能有形参，不能被重载。派生类的静态构造函数不能调用基类的静态构造函数。

namespace ConsoleApplication3

{

class Person

{

private string name;

public Person()

{

this.Name = "无名";

}

public Person(string name)

{

this.Name = name;

}

public string Name

{

get

{

return this.name;

}

set

{

this.name = value;

}

}

}

class Student : Person

{

private string major;

public Student()

{

this.Major = "未选专业";

}

public Student(string major)

{

this.Major = major;

}

public Student(string major,string name) : base(name)

{

this.Major = major;

}

public string Major

{

get

{

return this.major;

}

set

{

this.major = value;

}

}

}

class Program

{

static void Main(string[] args)

{

Student student1 = new Student();

Console.WriteLine(student1.Name);

Console.WriteLine(student1.Major);

Student student2 = new Student("电子信息工程");

Console.WriteLine(student2.Name);

Console.WriteLine(student2.Major);

Student student3 = new Student("电子信息工程", "李留威");

Console.WriteLine(student3.Name);

Console.WriteLine(student3.Major);

}

}

}

打印：

无名

未选专业

无名

电子信息工程

李留威

电子信息工程

请按任意键继续. . .