委托函数

namespace DelegateTest

{

class Program

{

static void cat(int num) {

Console.WriteLine("I am a cat who own " + num + " fish!");

}

static void dog(int num) {

Console.WriteLine("I am a dog who own " + num + " bone!");

}

delegate void animal(int num);

static void f(animal a,int num) {

a(num);

}

static void Main(string[] args)

{

f(dog,2);

f(cat, 3);

}

}

}

输出结果：

I am a dog who own 2 bone!

I am a cat who own 3 fish!

请按任意键继续. . .

1，delegate定义一个函数类型，delegate 函数类型 函数类型名（形参）；

delete void animal（int num）；

2，【函数类型名】和【函数形参】【一起】作为主调函数的参数。

void f（animal a; int num） {

a(num);

}

多播委托

animal a1 = new animal（dog）；

a1 += new animal(cat);

f(a1,num);

一次调用两个函数，但是两个被调函数的形参都是num。

输出：

I am a dog who own num bone!

I am a cat who own num fish!

请按任意键继续. . .

多播委托的函数必须是无返回值类型的函数。

static void cat(int num) {

Console.WriteLine("I am a cat who own " + num + " fish!");

}

static void dog(int num) {

Console.WriteLine("I am a dog who own " + num + " bone!");

}

delegate void animal(int num);

static void f(animal a, animal b, int num1,int num2) {

a(num1);

b(num2);

}

static void Main(string[] args)

{

f(dog,cat,1,2);

}

输出：

I am a dog who own 1 bone!

I am a cat who own 2 fish!

请按任意键继续. . .