Asp.net MVC 学习

1. C#知识点提要

1.1 特性（类似于java的注解）

例如 当使用nunit 进行测试的时候，测试的类上要用

[Testfixture]

Public class test{

//测试方法上要使用[Test]

[Test]

Public void test(){}

}

1.2 自动属性

在java中写bean的时候，写get,set方法要具体实现，而在c#中，可以直接通过

Public string Name{get;set;} 直接完成对属性的读写方法。

但是，记住，不能只写get或者set方法，必须要一起写，且get,set前面可以加权限，如 private set

Public class userInfo｛

public string name{set;get;}

public string password{set;get;}

｝

1.3 对象和集合的初始化器

对于上面那个model，我们采用对象初始化器写法

UserInfo user=new UserInfo{

Name=”zk”;

Password=”123”

}; //创建并初始化了一个实例

List<UserInfo> list=new List<UserInfo> {

new UserInfo{name=”zz”,password=”123”}

new UserInfo{name=”zzk”,password=”123”}

}

1.4 c#中委托的介绍

委托是一个类，它定义了方法的类型，使得可以将方法当作另一个方法的参数来进行传递。

委托的特点：

\* 委托类似于c++中的函数指针，但是类型安全的

\* 委托允许将方法作为参数传递。

委托的声明：

\* delegate: 常用的一种声明

Delegate至少0个参数，至多32个参数，可以无返回值，也可以指定返回值类型

　　Func可以接受0个至16个传入参数，必须具有返回值

　　Action可以接受0个至16个传入参数，无返回值

　　Predicate只能接受一个传入参数，返回值为bool类型

1.5 Lambda 表达式

Lambda表达式的格式如下：

（参数列表）=> 表达式或者语句块

示例代码：

1. (x, y) => x \* y         //多参数，隐式类型=> 表达式
2. x => x \* 5              //单参数， 隐式类型=>表达式
3. x => { return x \* 5; }      //单参数，隐式类型=>语句块
4. (int x) => x \* 5            //单参数，显式类型=>表达式
5. (int x) => { return x \* 5; }      //单参数，显式类型=>语句块
6. () => Console.WriteLine()   //无参数

1.6 linq 语言集成查询

类似于 流运算 和sql

二． controller 和 view的关系。

2.1 controller 控制器

