传智播客.NET 视频学习课件

访问.NET 网站了解更多课程详情

http://net.itcast.cn

（小提示：为什么本书中超链接打不开？）

观看须知

1.Pdf 是什么？ 我怎么看不了？

pdf 是一种电子书格式，本套课件需要安装了 pdf 的软件才可以观看。推荐 "Foxit reader"(资源里有安装软件，直接安装就可以。已经用卡巴斯基 2010 杀毒，请放心安装)

2.在观看过程点击链接报安全提示？ 没法打开电子书中的链接？

A 用 Foxit Reader 观看解决办法：

请找到软件中 工具----选项-----左侧信任管理器---右边“启用安全阅读模式”去掉“钩”即可。

B 用 Adobe Reader 观看解决办法：

一般直接点“允许”就可以了

3.这个资源是什么？

这是传智播客.NET 2010 年培训班现场教学所用的课件，是货真价实的现场授课内容。

4.我如何下载传智播客.NET 教学视频?

目前您可以通过 3 种方式，都是完全免费的！（小提示：为什么本书中超链接打不开？）

电骡（verycd）： http://www.verycd.com/topics/2857178/ （推荐）

传智播客.net 官方网站下载： http://net.itcast.cn/ViewArticle-103.aspx

如鹏网.net 教学专区： http://www.rupeng.com/forum/forum-54-1.html

5.我想了解一下.net 精品就业班的情况？如何预约报名？

请您访问传智播客.NET 官方网站（http://net.itcast.cn）

找首页的客服了解具体报名情况。（小提示：为什么本书中超链接打不开？）

6.传智播客.Net 精品就业班讲师介绍

杨中科，专注于企业级系统开发，撰写了《自己动手写开发工具》、《程序员的 SQL 学习笔记》等技术图书。CSDN 学生大本营 2009 年度十佳老师。曾任职于微软中国、金蝶软件等知名 IT 企业，主导了金蝶 EAS 湖南烟草局 SCM 系统、字符终端图形库 AHA3 及开发工具 AHAIDE、上海浦东发展银行图形前端等项目的开发，并且在中国工商银行批量平台、集中监控运维系统（部署于中国工商银行、中国农业银行、交通银行、北京银行、深圳发展银行等大中型银行）、力诺集团呼叫中心、新广源集团呼叫中心、大连地区环境发展预测等项目中担任主力开发人员。 杨中科发起的如鹏网（

http://www.rupeng.com）是专门为计算机初学者提供学习指导的网站，运营两年多来，撰写的大量的学习方法的文章和《C 语言也能干大事》等视频教程帮助无数的计算机初学者走出迷茫走入正确、快速发展的通道。

7.如何利用本课件学习？

配合传智播客.NET 视频教程学习（http://www.verycd.com/topics/2857178/），我们推荐您按如下顺序学习，效果更佳！

第 1 季 C#编程基础

第 2 季 C#面向对象基础

第 3 季 WinForm 基础

第 4 季 SQL 从入门到提高

第 5 季 ADO.Net

第 6 季 HTML

第 7 季 JavaScript

第 8 季 Dom

第 9 季 JQuery

第 10 季 asp.net 基础

第 11 季 asp.net 中级

第 12 季 asp.net 高级

第 13 季 ajax

传智播客会不断将现场教学中的教学内容以视频教程等方式整理发布出来，包括项目课程、其他.Net 高级技术等，了解最新视频教程及传智播客.Net 教学的全部课程内容请访问传智播客.Net 培训网站 http://net.itcast.cn

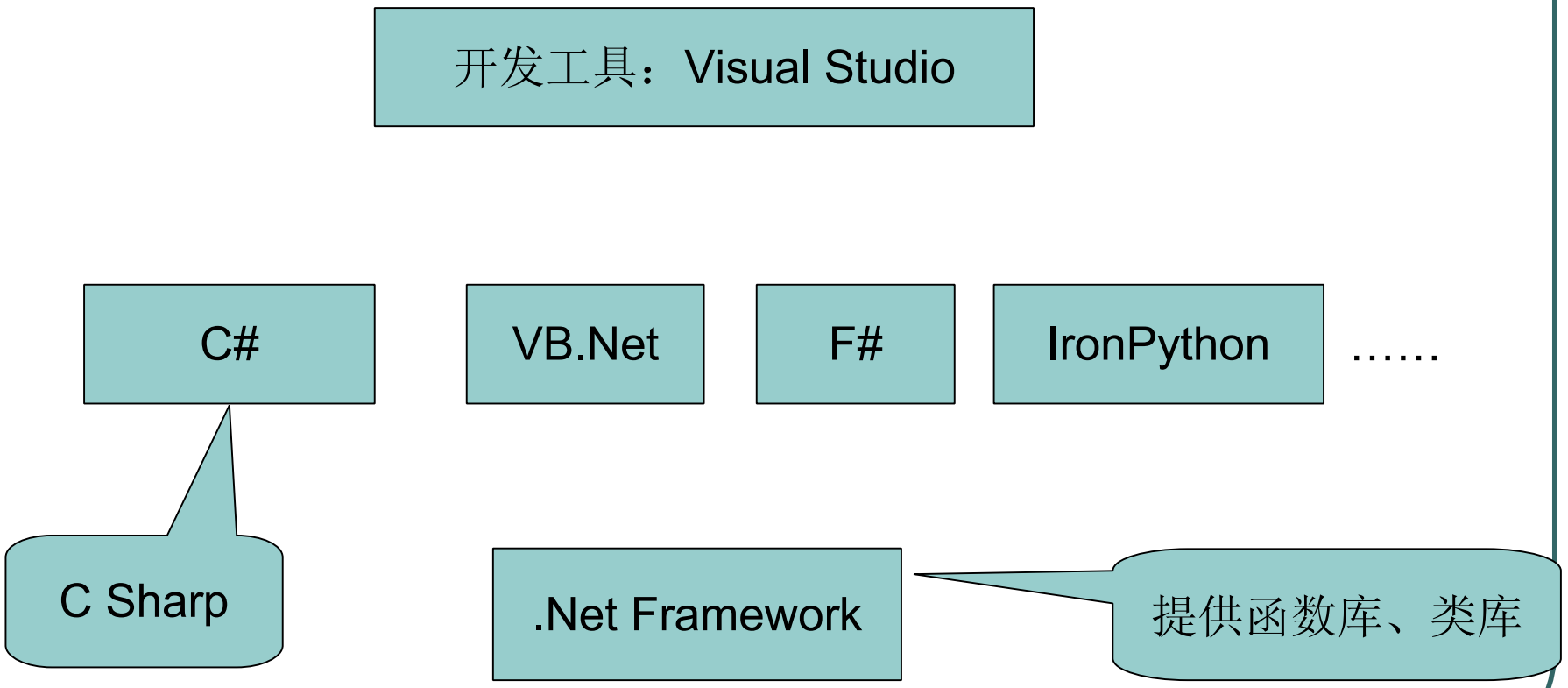
传智播客 www.itcast.cn —高级软件人才实作培训专家

C#编程基础

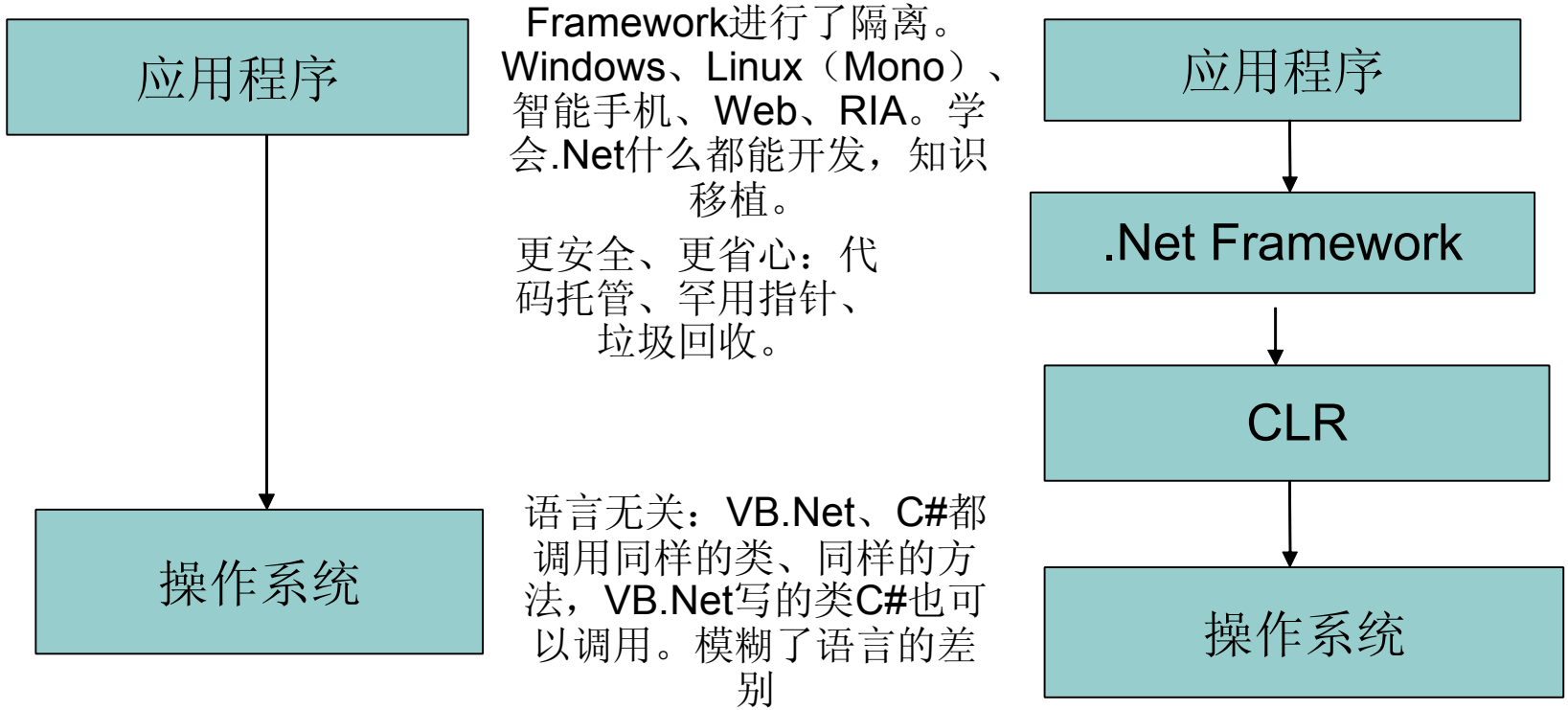
讲师：杨中科

.Net开发环境(*了解)

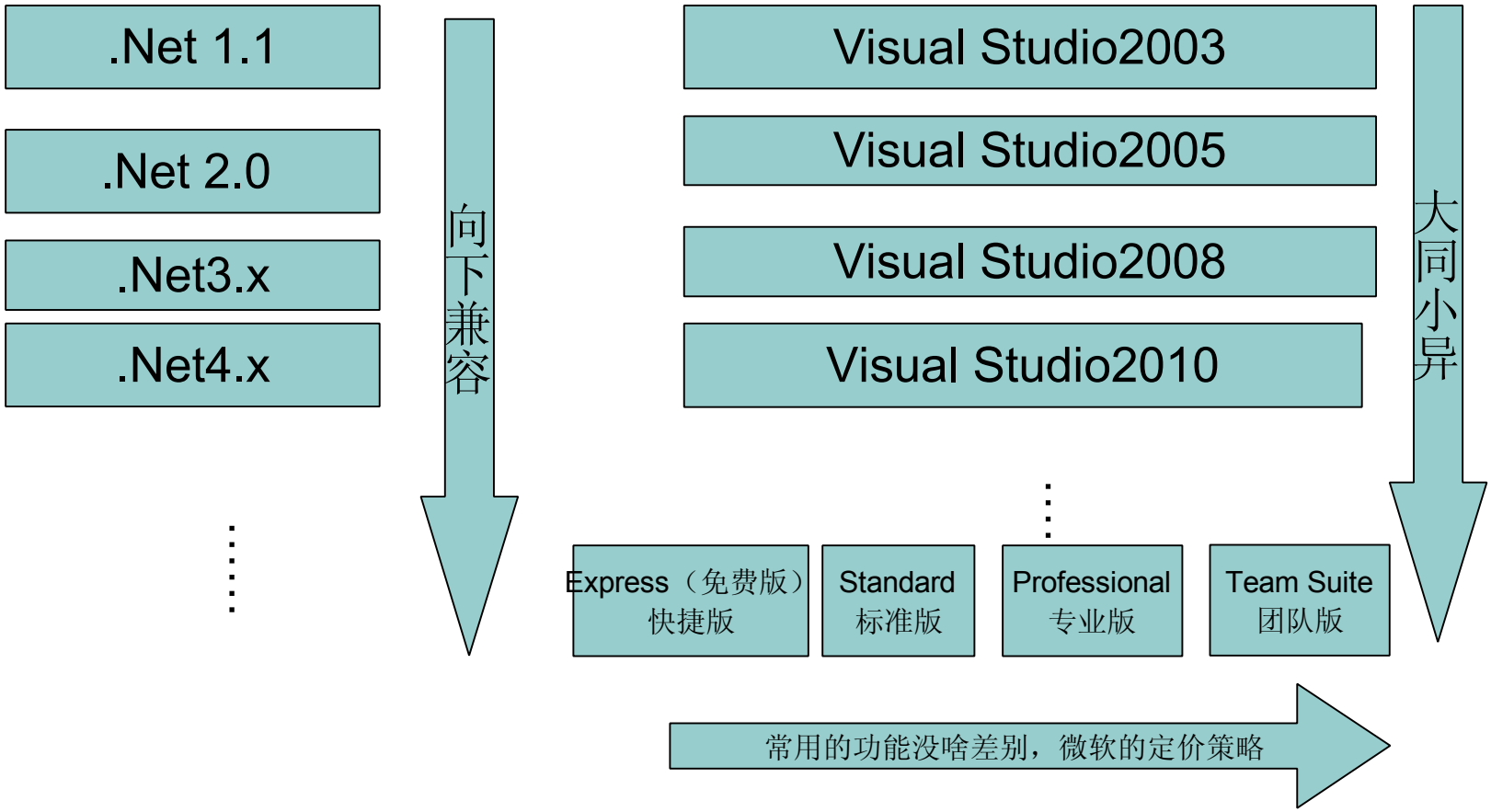
点Net、Dot Net



.Net简介（*了解）


应用程序
操作系统
平台无关：CLR、.Net Framework进行了隔离。Windows、Linux（Mono）、智能手机、Web、RIA。学会.Net什么都能开发，知识移植。
更安全、更省心：代码托管、罕用指针、垃圾回收。
语言无关：VB.Net、C#都调用同样的类、同样的方法，VB.Net写的类C#也可以调用。模糊了语言的差别
应用程序
.Net Framework
CLR
操作系统

.Net 的版本（*了解）


.Net 1.1
.Net 2.0
.Net3.x
.Net4.x
向下兼容
Visual Studio2003
Visual Studio2005
Visual Studio2008
Visual Studio2010
大同小异
Express（免费版）
快捷版
Standard
标准版
Professional
专业版
Team Suite
团队版
常用的功能没啥差别，微软的定价策略

第一个C#程序

- 创建第一个控制台程序并且调试运行。
- 为什么要从控制台程序开始？不要把精力放到表现层，而是把精力放到写代码上。无论控制台还是Winform还是ASP.Net最难的是写代码，而不是拖控件。
- 唯一需要学的三条控制台指令：Console.WriteLine：打印输出、Console.ReadLine：从控制台读入一行、Console.ReadKey：控制台暂停。至于Console.是什么意思不用管。1+1=2的例子
- 占位符是个什么玩意儿？{0}的个数必须和参数的个数一样，是一一对应的关系。

VS开发环境介绍

- 解决方案管理器：解决方案、工程、文件
- C#源文件一般以cs结尾
- C#程序的入口是Main函数，在Main中写代码就行，暂时不用关心其他部分是什么意思
- 错误列表。见到编译错误不用慌。错误排除演示。常见错误：结尾没有分号，大小写错误。
- 显示代码行号。工具→选项→文本编辑器→C#→显示→行号

C#语法

- 从上到下一条条的依次执行。
- 大小写敏感
- 函数参数用()包围
- 两句代码之间用分号（;）分割（易错：全角问题）
- 注释：单行注释、多行注释。

变量

- 把变量看成放数据的容器。
- 定义变量的方式：类型 变量名；int i3;
- 变量的类型：不同类型的容器放不同的东西，铁罐不能放硫酸。不能在int类型的变量中放字符串。
- 变量不能放和变量类型不兼容的数据。
- 变量的名字不能重名（不严格）
- 问题
  - int i=10;Console.WriteLine("i");

常用基本数据类型

- string、int 、char 、bool 、decimal （了解） 、byte（了解）、double（了解）、long（了解）、float（了解）等。
- bool的取值：true、false。
- 为什么输出"要用转义符"\""，因为编译器默认是遇到"开始字符串，再遇到"是结束字符串，但是如果遇到前面有\的"就不把它当成有字符串起始意义的"。
- string："\"ab\""、"ab\nb"、"c:\\a.txt"、@"c:\a.txt"（推荐）。@表示字符串中的\不当成转义符
- @不是万能的，不能解决字符串中有双引号的问题，如果有双引号还是用转义符
- ‘a’是char，"a"是string
- string s = Console.ReadLine();
- 问题：
  - String str=Console.ReadLine();
  - Console.WriteLine(str);
  - 用户输入a\nb，输出什么？转义符只针对在代码中直接写出的字符串，对于程序中读取出来没有这个问题。
- 简单的类型转换：Convert.ToString()、ToString()、Convert.ToInt32() 。
- 输入一个数，打印这个数的平方。

补充

- 问题：int i=Convert.ToInt32(Console.ReadLine())
- string s1 = Console.ReadLine();
- int i = Convert.ToInt32(s1);
- @可以表示多行字符串。

变量的命名

- 命名规则：第一个字符必须是字母或者下划线（_），其后的字符可以是任意个数字、字母、下划线。不能全部使用C#的关键字，比如class、namespace、new、void等。判断方式：VS中亮蓝色的就是关键字。
- 这也是类、函数等的命名规则。
- 中文变量名（类名、函数名）。
- C#中建议变量的开头用小写。
- 下列哪个是正确的变量名？_a、a、a1、a_a、1_a、a1、1a、a3_、a$b、int、int1、a b、A1、INT。
- 变量的声明：int i;int x,y;int i=3;

运算符、表达式

- +、-、*（乘）、/（除）、%（求余）
- +可以用作字符串连接，其他不可以。将string和其他类型+，会自动tostring();
- 易错：string s1 = "hello"+yang;
- ++（自增）、--（自减）
- (*)运算符优先级：i1+i2*i3、(i1+i2)*i3。不要变态，括号是王道。内层的()先计算。
- 变量可以和字面量混合运算。
- 案例：让用户输入两个数，打印出两个数的和。

赋值

- 赋值运算符=，让左边变量的值等于右边的计算结果。这就能解释令人不解的i=i+1；
- （*）+=、-=、*=、/=。
- i2=i1++;i2=++i1；(了解)。--同理。
- 习题：int a=10;a++;a=a+a;Console.WriteLine("{0}",a);执行结果是什么？
- 易错：int i=10;int j=i;i=5;j=?。
- int x=10;x+y=80;Console.WriteLine(y); x + y = 80; //左边的必须是变量！！！不能是常量，不能是表达式
- 习题：交换两个变量的值

布尔运算

- 相等判断：==，不要和=混淆。WriteLine("{0}",i==1);WriteLine("{0}",i=1);的区别。Console.WriteLine("{0}",i=1);//C#中赋值表达式也有值，它的值表示为赋值后变量的值
- 不等判断：!=
- 大小比较：<、>、<=、>=
- 取反：!
- 组合运算：&&（并且）、||（或者）。
  - && 并且：只有两边都为true的时候，表达式的值才为true，否则是false；
  - ||或者：两边只要有一个为true的时候，表达式的值就是true，否则是false；

if

```
if(i>1)
{
}
```

```
if(i>1)
{
}
else
{
}
```

单句的时候大括号可以省略，但是不建议省略，演示说明why

```
if(i>100||i<10)
{
}
```

```
if(i>100&&j>10)
{
}
```

```
if(i>100)
{
}
else if(i<10)
{
}
else
{
}
```

if语句的嵌套

```
if((i>0&&j>0)||(i<0&&j<0))
{
}
```

易错代码：

```
if(i>10);
{
    Console.WriteLine("是");
}
```

课上练习

- 学编程不是看书，不是听老师讲，而是自己动手写。
- 作业1：提示用户输入密码，如果密码是“888888”则提示正确，否则提示错误。
- 作业2：提示用户输入密码，如果密码是“888888”则提示正确，否则要求再输入一次，如果密码是“888888”则提示正确，否则提示错误。
- 作业3：提示用户输入用户名，然后再提示输入密码，如果用户名是“admin”并且密码是“888888”，则提示正确，否则提示错误，如果用户名不是admin还提示用户用户名不存在。
- 作业4：提示用户输入年龄，如果大于等于18，则告知用户可以查看，如果小于10岁，则告知不允许查看，如果大于等于10岁，则提示用户是否继续查看（yes、no），如果输入的是yes则提示用户可以查看，否则提示不可以查看。（测试边界条件，-1,8888888888888888888888888，aaaa。微软如狼似虎的Tester）
- 作业5：依次提示用户输入两个整数（假设i1、i2）。如果i1、i2都是正数，则将i1的值递增一个数，然后打印i1+i2的值；如果i1、i2都是负数，则将i1的值递减10个数，然后打印i1*i2的值；如果i1、i2中任一个为0，则提示数据有错误；否则计算i1*i2的绝对值。
- 注意：变量命名要用有意义的变量名。

switch case

被判定的值进入满足条件的分支执行

```
switch(i)
case 1:
//
break;
case 2:
//
break;
```

```
switch(i)
case 1:
//
break;
case 2:
//
break;
default:
break;
```

```
switch(i)
case 1:
case 2:
//
break;
```

case中的值必须是常量，不能是变量、表达式。

```
switch(i)
case a:
//
break;
```

- 类似于if...else...else if...else，但是是离散值的判断。
- switch一般都可以用if重写，但是if不一定能用switch重写
- 不要忘了break。C#中的break不写是不行的，除了合并case的情况

while循环

```
while(i<100)
{
  //打印i
  i++;
}
```

只要while后小括号中的表达式为true，就不断执行大括号中的代码

```
while(不是终点)
{
    跑
}
```

```
do
{
  //打印i
  i++;
}
while(i<100)
```

先做……如果满足则再来一次，直至while表达式为false。至少被执行一次。（*）

练习1：用while计算1到100之间整数的和；（有更好的数学解法）

练习2：要求用户输入用户名和密码，只要不是admin、888888就一直提示要求重新输入。

练习3：不断要求用户输入一个数字，然后打印这个数字的二倍，当用户输入q的时候程序退出（return）。

练习4：不断要求用户输入一个数字（假定用户输入的都是正整数），当用户输入end的时候显示刚才输入的数字中的最大值。设一个变量int max，初始值为0，用户每输入一次就把用户输入的和max比较一下，如果输入的比max大，则让max等于用户输入。

易错：(1)string s = Console.ReadLine()，读取s的时候并不会再次要求输入；（2）while(Console.ReadLine()!="q")

循环的中断

- break：立即终止整个循环。
- continue：立即终止当前循环步骤，进行下一次循环步骤。
- 练习1：用 while continue实现计算1到100之间的除了能被7整除之外所有整数的和。
- 练习2：用while break实现要求用户输入用户名和密码，只要不是admin、888888就一直提示要求重新输入。
- 练习3：编写聊天机器人，如果问“今天天气怎么样？”则回答天气，如果问……，如果说“88”，则“再见”。

for循环

- for(code1;code2;code3)。code1：循环的初始化代码，只在循环开始之前运行一次；code2，bool类型的表达式，每次循环完前都判断一下是否为true，只有为true才会进行本次循环；code3在每次循环之后执行一次。各段之间用回车换行，设置断点查看执行过程。
- for的三段都可以省略，但是不能丢了“;”。
- break、continue同样可以应用于for。
- for和while代码之间都可以互相转换，究竟用哪种方式则取决于一些惯用用法和个人习惯，不用记，代码写多了自然就有感觉

```
for(int i=0;i<10;i++)
{
}
```

```
for(int i=100;i>0;i--)
{
}
```

如果是遍历数组，遍历1..100推荐直接在第一段中声明变量。

for中终止条件注意是<还是<=

用 for实现计算1到100之间所有整数的和。用 for实现计算200到300之间所有整数的和。

类型转换Cast（*）

- 把源类型赋值给目标类型，两个类型不一致的时候会发生类型转换。a=b，b是源，a是目标
- 隐式转换、显式转换。当目标类型一定能满足源类型转换过去后的要求的话就是隐式转换；如果当目标类型不一定能满足源类型转换过去后的要求的话就需要显式转换（程序员自己负责）。
- 把中国人转换为人是隐式转换，把人转换为中国人则是显式转换（强制转换）
- 内存中的数据没有变化，只是不同的视角而已。
- int、byte的表示范围：int.MaxValue、int.MinValue。
- byte b=1;int i=1;i = b;b = (byte)i;
- 为什么要显式转换？编译器不能保证一定正确，转换结果自己负责。要人、给一个中国人当前没问题，但是要中国人、给一个人则可能有问题，编译器不承担这个责任，由开发人员来保证。
- 只有在内存存储上存在交集的类型之间才能进行Cast，否则则不可以，比如不可以int i;string s=(string)i;反之也不可以。这种情况必须用Convert类提供的方法。

类型转换Convert

- Convert不再是内存级别的转换，而是考虑数据意义的转换。可以把姓名Convert成人。Convert是一个加工、改造的过程
- Convert.ToInt32、Convert.ToString、每种类型都还有ToString方法（类型的ToString和Convert.ToString略微的差别暂时不用关心）

枚举

确定数量、确定值的几个取值：东西南北、男女、上中下。

```
enum Gender{male,female}
enum QQStatus{online,offline,hidden}
```

为什么有枚举，如果用string来表示四季：

- //1、需要在每个地方都进行数据合法性的校验
- //2、给函数传递参数的时候，只有看文档才只能哪些值合法

枚举的用法，QQStatus status = QQStatus.online;

和用字符串比起来，用枚举的好处就是限定了变量的取值范围，程序处理起来更方便。

为什么不能：QQStatus status = online;

数组

- 保存多个值。任意类型都可以声明数组。

```
int[] nums = {5,3,8}
```

```
int[] nums = new int[3]
```

```
int[] nums = new int[3]{5,3,8}//个数和声明个数必须一致
int[] nums = new int[5]{5,3,8}//错误
```

使用索引器访问指定编号位置的元素，访问数组元素：nums[0]、nums[1]。索引从0开始。取到的元素的类型就是数组元素的类型。还可以对数组元素进行赋值

数组的长度一旦声明就无法改变

练习1：从一个整数数组中取出最大的整数

练习3：将一个字符串数组输出为|分割的形式，比如“小月月|大月月|老月月”

练习4：将两个int类型数组连接为一个string类型数组。

练习2：计算一个整数数组的所有元素的和。

练习5：有一个整数数组，请声明一个字符串数组，将整数数组中的每一个元素的值转换为字符串保存到字符串数组中。string[] strs = new string[values.Length];

练习6：将一个字符串数组的元素的顺序进行反转。{"3","a","8","haha"} {"haha","8","a","3"}。第i个和第length-i-1个进行交换。

练习7：将一个int数组向左滚动平移1次。{3,5,8,7}→{5,8,7,3}

数组2

foreach循环

```
string[] names= {"tom","jerry","lily"};
foreach(string name in names)
{
   Console.WriteLine("我的名字{0}",name);
}
```

和for的区别：for可以不逐个遍历，比如每隔一个遍历一个，或者可以从后向前遍历

只能（只应该）在foreach对集合进行读，而不应该写。

函数

- 函数就是将一堆代码进行重用的一种机制。函数就是一段代码，这段代码可能有输入的值（参数），可能会返回值。一个函数就像一个专门做这件事的人，我们调用它来做一些事情，它可能需要我们提供一些数据给它，它执行完成后可能会有一些执行结果给我们。要求的数据就叫参数，返回的执行结果就是返回值。
- Console.ReadLine就是一个有返回结果的函数；Console.WriteLine("hello")就是一个有执行参数的函数，只有告诉WriteLine被打印的数据它才知道如何打印；int i= Convert.ToInt32("22")则是一个既有参数又有返回值的函数。
- 有了函数写代码就像拼积木，C#中的各种各样的技术其实就是通过for、if等这些基础的语法将不同的函数按照一定的逻辑组织起来。

自己写函数

1 读取输入的整数，定义成函数，多次调用

```
static int ReadInt()
{
   String s = Console.ReadLine();
   return Convert.ToInt32(s);
}
```

写程序测试，程序调用到函数的时候是进入函数内部执行的，执行完毕再继续向下执行。

2、函数如果没有返回值则标记返回值类型为void

3、return语句

导致函数立即返回。在返回值为void的函数中return，在返回值非void的函数中"return 数值"。

return不是结束程序，只是结束当前函数，控制台程序中Environment.Exit(0)退出程序。

函数返回值易错点

一个函数如果“答应”返回一个非void类型的值，则函数的所有路径都要有返回值。比如将对输入年龄转换为年龄段描述的函数。

函数参数

- 计算两个整数中的最大值：int Max(int i1,int i2)
- 计算输入数组的和：int Sum(int[] values)
- 确定函数参数的原则：自己能确定的数据自己内部解决，自己确定不了的数据通过参数传递。
- 练习：string[] strs={"aa","333","ccc"};
- 返回给我一个字符串，然后字符串使用我指定的分隔符来进行分割，比如我指定用"|"分割，那么返回给我"aa|333|ccc"。
- string Join(string[] strs,string seperator)
- 问题：函数名开头大写，参数名开头小写，参数名、变量名要有意义

可变参数

- 参数数组：int sum(params int[] values)

int sum(string name,params int[] values)

可变参数数组必须是最后一个

- 参数默认值（.Net4.0）：void SayHello(string name,int age=20)(*了解)

函数重载（函数重名）

函数的重名：错误

```
static void SayHello(string name)
{
    Console.WriteLine("我是{0}",name);
}
static void SayHello(string name)
{
    Console.WriteLine("I am{0}",name);
}
```

```
static void SayHello(string name)
{
    Console.WriteLine("我是{0}",name);
}
static void SayHello(int age)
{
    Console.WriteLine("我的年龄{0}",age);
}
```

```
static void SayHello(string name) 错误
{
    Console.WriteLine("我是{0}",name);
}
static int SayHello(string name)
{
    return 10;
}
```

```
static void SayHello(string name)
{
    Console.WriteLine("我是{0}",name);
}
static void SayHello(string name,string nickname)
{
    Console.WriteLine("我是{0}，昵称是
{1}",name,nickname);
}
```

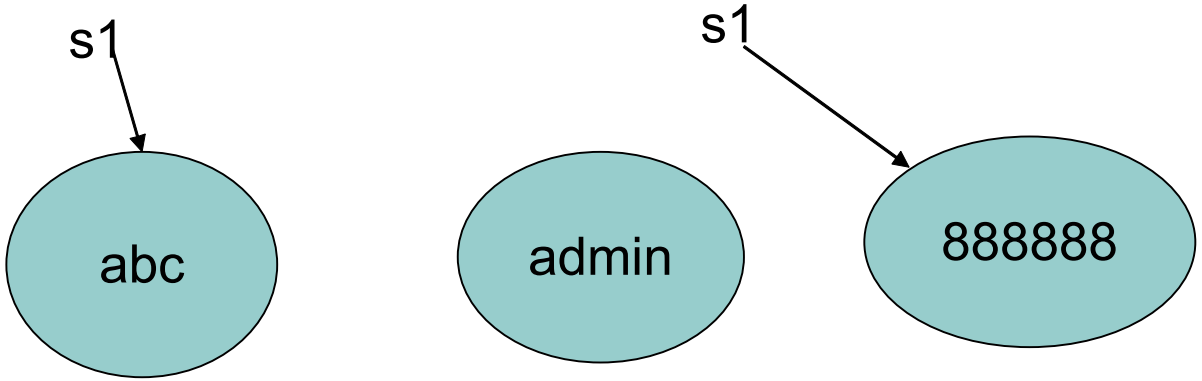
构成重载的条件：参数类型不同或者参数个数不同（不严谨的），与返回值无关。

字符串的处理

- C#中单个字符用单引号包含就是char类型，（'a'），单引号中放且只能放一个字符。
- 单个字符也可以表示为字符串，还可以有长度为0的字符串
- 使用s.Length属性来获得字符串中的字符个数
- string可以看做是char的只读数组。char c = s[1];。例子：遍历输出string中的每个元素。
- C#中字符串有一个重要的特性：不可变性，字符串一旦声明就不再可以改变。所以只能通过索引来读取指定位置的char，不能对指定位置的char进行修改。
- 如果要对char进行修改，那么就必须创建一个新的字符串，用s.ToCharArray()方法得到字符串的char数组，对数组进行修改后，调用new string(char[])这个构造函数（暂时不用细研究）来创建char数组的字符串。一旦字符串被创建，那么char数组的修改也不会造成字符串的变化。例子：将字符串中的A替换为a。

常见疑问

- 谁说字符串不可变？string s = "abc";s="123"，s这不是变了吗
- 要区分变量名和变量指向的值的区别。程序中可以有很多字符串，然后由字符串变量指向他们，变量可以指向其他的字符串，但是字符串本身没有变化。字符串不可变性指的是内存中的字符串不可变，而不是变量不变。
- string s10 = s1;//s10指向s1指向的字符串，而不是s10指向s1，哪怕s1以后指向了其他内存，那么s10还是指向"hello"



String类常用函数

- ToLower()：得到字符串的小写形式。
- 注意字符串是不可变的，所以这些函数都不会直接改变字符串的内容，而是把修改后的字符串的值通过函数返回值的形式返回。s.ToLower()与s=s.ToLower()
- ToUpper()：得到字符串的大写形式； Trim()去掉字符串两端的空白。
- s1.Equals(s2, StringComparison.OrdinalIgnoreCase)，两个字符串进行比区分大小写的比较。

字符串的分割

- string[] Split(params char[] separator)：将字符串按照指定的分割符分割为字符串数组；
- string[] Split(char[] separator, StringSplitOptions options)将字符串按照指定的char分割符分割为字符串数组（ options 取RemoveEmptyEntries的时候移除结果中的空白字符串）；
- string[] Split(string[] separator, StringSplitOptions options)将字符串按照指定的string分割符分割为字符串数组。
- 例子1：从日期字符串（"2008-08-08"）中分析出年、月、日；2008年08月08日。
- 例子2：从一个记录了学生成绩的文本文档，每个学生成绩是一行，每行是用|分割的数据，用|分割的域分别是姓名、年龄、成绩，写程序取出成绩最高学生的姓名和成绩。参考：使用string[] lines = System.IO.File.ReadAllLines(@"c:\root.ini", Encoding.Default);从文本文件读取数据，返回值为string数组，每个元素是一行。

字符串函数详解

- 字符串替换：string Replace(string oldValue, string newValue)将字符串中的出现oldValue的地方替换为newValue。例子：名字替换。
- 取子字符串：string Substring(int startIndex)，取从位置startIndex开始一直到最后的子字符串；
- string Substring(int startIndex, int length)，取从位置startIndex开始长度为length的子字符串，如果子字符串的长度不足length则报错。
- bool Contains(string value)判断字符串中是否含有子串value
- bool StartsWith(string value)判断字符串是否以子串value开始；
- bool EndsWith (string value)判断字符串是否以子串value结束；
- int IndexOf(string value)：取子串value第一次出现的位置。

字符串的处理练习

- 课上练习1：接收用户输入的字符串，将其中的字符以与输入相反的顺序输出。"abc"→"cba"
- 课上练习2：接收用户输入的一句英文，将其中的单词以反序输出。"hello c sharp"→"sharp c hello"
- 课上练习3：从Email中提取出用户名和域名：abc@163.com。IndexOf找到@的位置。SubString。
- 课上练习4：文本文件中存储了多个文章标题、作者，标题和作者之间用若干空格（数量不定）隔开，每行一个，标题有的长有的短，输出到控制台的时候最多标题长度20，如果超过20，则截取长度17的子串并且最后添加“...”，加一个竖线后输出作者的名字。
- 练习5：自己动手写聊天机器人，能够回答不同城市的天气、回答感情问题、问的次数过多还会饿死，提问可以模糊提问（包含指定关键词）。问“天气：北京”就回复“北京的天气是晴”。

函数的ref、out参数（*）

函数参数默认是值传递的，也就是“复制一份”，例子：

```
int age=20;
IncAge(age);
Console.WriteLine("age={0}",age);
```

ref必须先初始化，因为是引用，所以必须先“有”，才能引用，而out则是内部为外部赋值，所以不需要初始化，而且外部初始化也没用。

ref应用场景内部对外部的值进行改变，out则是内部为外部变量赋值，out一般用在函数有多个返回值的场所。

案例：两个变量的交换；int.TryParse。

面向对象概念(*)

- 面向对象不是取代面向过程的。
- 类、对象。“人”是类，“张三”是“人”这个类的对象。类是抽象的，对象是具体的。按钮就是类，某个按钮就是对象。对象可以叫做类的实例（Instance）。类就像int，对象就像10。字段Field（和某个对象相关的变量），字段就是类的状态。人这个类有姓名、年龄、身高等字段。类不占内存，对象才占内存。
- 方法Method，方法就是类能够执行的动作，比如问好、吃饭等。
- 类的继承，类之间可以有继承关系，比如“电脑”类可以从“电器”类继承，这样的好处是“电脑”类只需要定义自己特有的字段、方法就可以，也就是只要定义内存大小、CPU型号这些字段或者弹出光驱等方法就可以。父类(Parent)、基类(Base，基业，祖宗十八代传下来的)。电脑类是电器类的子类(ChildClass)。重用。
- 面向对象的三个特性：封装、继承、多态。
- 没有面向对象的世界中的难题。

定义类

- class Person{ }

定义字段：

```
class Person
{
   public string Name;
    public int Age;
    public void SayHello()
    {
        Console.WriteLine("你好,我是{0}",Name);
    }
}
```

初始化对象，调用方法：

```
Person tom = new Person();
tom.Name="tom";
tom.SayHello();
Console.WiteLine("{0}",tome.Name)
```

定义方法（类的函数）：

```
class Person
{
    public void SayHello()
    {
        Console.WriteLine("你好,我是{0}",this.Name);
        //不写this.也可以，写上更清晰
    }
}
```

初始化对象，调用方法：

```
Person tom = new Person();
tom.SayHello();
```

对象必须调用构造函数初始化“new Person()”后才能用，不能Person tom这样声明;就调用。

一个类可有有多个实例。

类就是把一系列相关的变量（状态）、行为定义为一个整体。字段记录的就是这个对象相关的数据。

成员访问级别

- 字段、方法、属性（后面讲）都可以叫做类的成员Member，它们都需要定义访问级别。访问级别的用处在于控制成员在哪些地方可以被访问，这样达到面向对象中“封装”的目的。
- 几个访问级别：public（任何地方都可以访问）；private（默认级别。只能由本类中的成员访问）。还有internal、protected两个级别，以后会讲。

```
class Person
{
    private string Name;

    public void GiveName(string name)
{
    Name = name;//拒绝不好听的名字
}
     public void SayHello()
     {
         Console.WriteLine("你好,我是
{0}",Name);
     }
}
```

```
Person tom = new Person();
tom.GiveName("tom");
 //tom.Name="tom";
 tom.SayHello();
```

永远不要把字段public

属性

惯用法：属性开头字母大写，字段开头字母小写

```
class Person
{
    private int age;
    public int Age
    {
        get{return age;}
        set{age=value;}
    }
    public void SayHello()
    {
        Console.WriteLine("我的年龄是{0}",Age);
    }
}
```

- 只用set或者只用get就可以定义只读或者只写属性（只写的不常见）
- 可以为set、get设置访问级别
- 例子，限制非法值的设置
- （.Net3.x）简化set、get：public int Age{get;set;}。适合于set、get中没有特殊逻辑代码的情况。允许外部访问的值一定要声明为属性。
- 字段和属性的区别是什么？属性看似字段、不是字段，可以进行非法值控制，可以设置只读。
- set、get块内部其实就是get_***、set_***方法。

课上练习

- 面向对象版聊天机器人
- 机器人有不同的名字、维护自己的FullLevel，可以SayHello（我叫***），可以喂食（Eat(int foodCount)），可以对它说话（Speak），对异常情况（错误的喂饭数字，喂的太多撑死了）进行处理，有两个机器人供选择，一开始通过1、2数字选择聊天机器人。

构造函数

- 构造函数用来创建对象，并且可以在构造函数中对对象进行初始化。
- 构造函数是用来创建对象的特殊函数，函数名和类名一样，没有返回值，连void都不用。
- 构造函数可以有参数，new对象的时候传递函数参数即可
- 构造函数可以重载，也就是有多个参数不同的构造函数。
- 如果不指定构造函数，则类有一个默认的无参构造函数。如果指定了构造函数，则不再有默认的无参构造函数，如果需要无参构造函数，则需要自己来写。

对象的引用(非常重要)

- i1=3;i2=i1;i1++;//i2是3
- p1=new Person();p1.i=3;p2=p1;p1.i++;//p2.i是4
- int、string、decimal、bool、byte等基础类型（值类型）是传递拷贝；对象（引用类型）则是传递引用。因为基础类型不怎么占内存，而对象则比较占内存。
- 函数间传递对象。
- 为对象变量重新赋值。p2=p1是让p2指向p1指向的对象。

继承

```
public class Chinese:Person
{
    public string HuKou{get;set;}
    public void KongFu(){....}
}
Chinese p1 = new Chinese();
p1.Name="李雷";
p1.HuKou="北京市朝阳区";
p1.SayHello();
p1.KongFu();
```

定义类的时候不指定父类，则父类是Object类。Object类是任何类的直接或者间接父类。

对象的隐式转换和显式转换

```
Chinese ch= new  Chinese();
//隐式转换，把子类变量赋值给父类变量
Person p = ch;
//显式转换，把父类变量赋值给子类变量
Person p = new Chinese();
Chinese ch = (Chinese)p;
//如果对象不在同一个继承树路径上
//则不能强制类型转换
Dog g = new Dog();
Chinese ch = (Chinese)g;//错误
//is运算
if(p is Chinese)
{
   Console.WriteLine("中国人");
}
else if(p is Korean)
{
   Console.WriteLine("韩国人");
}
```

```
//as 运算符
Chinese ch = p as Chinese ;
if(ch!=null)
{
   Console.WriteLine("中国人");
}
Korean ch = p as Korean ;
if(ch!=null)
{
   Console.WriteLine("韩国人");
}
```

()转换和as 转换的区别：如果转换失败()会报异常，而as则会返回null。

异常与异常处理

- 传统的错误表示方式：错误码。举例。需要知道不同错误码的含义，如果不处理错误码，则程序可能陷入不可以预置的错误。陈摄影师以为文件已经被删除造成的麻烦。
- 错误码的缺点：不处理则很难发现，每次处理则很麻烦；难以看出错误的原因；容易使得程序进入不确定状态。
- try catch。Exception ex 异常也是对象。
- Exception 类主要属性：Message、StackTrace
- 发生异常后程序默认就退出了，try代码块中的后续代码不会被执行。catch以后的代码则会继续执行。
- 不要吃掉异常，一般情况下不需要处理异常。
- 扔出自己的异常，扔：throw，抓住：catch

常量与静态成员

- const常量。常量名要大写。一定不会变化的值才能声明为常量。
- 全局变量。static类变量。
- 不用new就能用的方法：static方法，static方法其实就是普通函数
- 在static方法中可以调用其他static成员，但是不能调用非static成员。在非static方法中可以调用static成员。
- 静态类，不能被new的类就是静态类。静态类一般用来实现一些函数库。***Helper，SqlHelper，PageHelper。

命名空间

- namespace（命名空间），用于解决类重名问题，可以看做“类的文件夹”。
- 在代码中使用其他类的时候需要using类所在的namespace。System.Collections.ArrayList，快速引入的方法，右键→解析（Ctrl+.）。
- 为什么使用Convert、Console等类不需要自己写using？
- 如果代码和被使用的类在一个namespace则不需要using。
- 可以修改默认的namespace，因此不要认为在相同文件夹下就不用using，不在相同文件夹下就需要using。
- 类内部定义的类的引用：namespace+外部类名+内部类名

索引器

- C#中提供了按照索引器进行访问的方法
- 定义索引器的方式：string this[int index]{get { return ""; }set { }}，string为索引器的类型，[]中是参数列表。进行索引器写操作就是调用set代码块，在set内部使用value得到用户设置的值；进行读操作就执行get代码块。
- 索引器参数可以不止一个，类型也不限于int，几乎可以是任意类型。

第一个Windows程序

- WinForm：Windows Form，.Net中用来开发Windows窗口程序的技术，无论是之前学的控制台程序，还是后面要学的ASP.Net都是调用.Net框架，因此所有知识点都是一样的。
- 新建一个Windows项目：Windows→Windows窗体应用程序
- 控件：窗口上很多元素都是相似的，因此将这些元素抽象为一些类，这些类就叫做控件。识别PowerPoint中的控件，按钮（Button）、文本框（TextBox）、标签（Label）、单选按钮（RadioButton）、复选框（CheckBox）。
- 添加、删除、移动、缩放控件

WinForm概念

- 拖放控件，输入姓名，点击按钮，窗口标题显示问好。设置窗口对象的属性就可以改变窗口的外观。
- 点击按钮调用TextBox的Hide方法，调用控件的方法就可以使得控件发生动作。
- 当用户点击按钮的时候Button1_click方法被调用，这个方法不是程序员调用的，而是程序员把方法写好，并且说明“当用户点击按钮的时候执行Button1_click方法中的代码”，这一点和控制台程序不同。这被称为“好莱坞原则”：不要找我，我会找你（Don't call me,I will call you ）
- 控件、属性、事件、事件处理方法、控件的名字（控件的实例；对象名）

简单的WinForm程序

- 简单的加法计算器，用户在文本框1、2中输入两个数，点击按钮，在文本框3中显示两个数的和。如果1或者2为错误的数据格式，则弹出对话框提示错误。int.TryParse、MessageBox.Show。 string s = string.Format("{0}你好", textBox1.Text);//推荐
- 练习1：输入宽和高，输出面积。
- 练习2：输入Email地址，输出用户名和域名。
- 练习3：用户在文本框1、2中输入两个数，点击按钮，在文本框3中显示从文本框1中的数字到文本框2中数字之间的累加和。如果1或者2为错误的数据格式，则弹出对话框提示错误。如果文本框1中的数字比文本框2中数字大，则提示错误。
- 练习4：页面上有一张图片(PictureBox，在Image属性中加载图片)，默认是隐藏的（Visible=False），用户在文本框中输入身份证号（131226198105223452），点击按钮，如果年龄大于18岁则显示图片（Visible=True），否则提示年龄太小。取当前年份：DateTime.Now.Year。
- 练习5：页面上有一个文本框，文本框左侧和右侧各有一个按钮，点击左侧按钮文本框中的文字向左循环滚动一次，点击右侧按钮文本框中的文字向右循环滚动一次。

TextBox

文本框的几种模式：Multiline（多行）、PasswordChar（密码）

将文本框的PasswordChar设为*就是密码框效果，将MultiLine属性设置为true并且将高度拉到合适的大小就是多行效果，textBox4.AppendText("hello"+"\n")就是附加一行。

控件名要有含义、控件名前缀的“潜规则”。按钮Button：btn；文本框TextBox：txt；复选框CheckBox：cb。控件的名字要有意义。

案例1：登录界面。登录错误三次退出程序，假设用户名、密码是admin、888888，不区分大小写。（易错点：局部变量与类变量）

退出程序this.Close()或者Application.Exit()

案例2：修改密码。界面上有旧密码、新密码、重复新密码，假设旧密码为888888，两次输入的新密码必须和旧密码不一样，并且两次输入的新密码必须一致。

案例3：在多行文本框中输入多行“姓名=成绩”格式的数据，要求输出成绩最高的学生的姓名和成绩。

ComboBox

- SelectedIndex ：选中项的序号。没有任何选中的时候是-1，否则是选中的序号（0开始）
- 练习1：简单的四则运算器
- 如何禁止用户编辑？三种风格。
- 响应选择改变事件
- 练习2：省市选择器
- 练习3：日月选择器，假设2月份总是28天。1、3、5、7、8、10、12月份是31天，其他是30天。表驱动。

ListBox

- ListBox：SelectedIndex、Items、SelectedIndexChanged事件。
- 多选的方式，设置SelectionMode为MultiExtend
- ListBox：SelectedIndices、SelectedItem、SelectedItems、SelectionMode；
- 案例：人员选择，左边选择到右边、右边退回左边、批量添加。遇到的问题，删除顺序，计算机的思维。
- 问题：删除自动生成的事件代码。先在事件视图中删除方法名，再到代码中删除；先删除form.cs中的，再去手动删除designer.cs中。

计时器Timer

- Timer。每隔一段时间触发一个事件。不可视控件。Inerval、Enabled。Tick事件。
- 计量单位：ms（毫秒）。 1秒=1000毫秒
- DateTime
- 案例：小时钟。取当前时间DateTime.Now.ToString();
- 案例：走马灯
- 案例：QQ消息窗口。如何显示其他窗口。

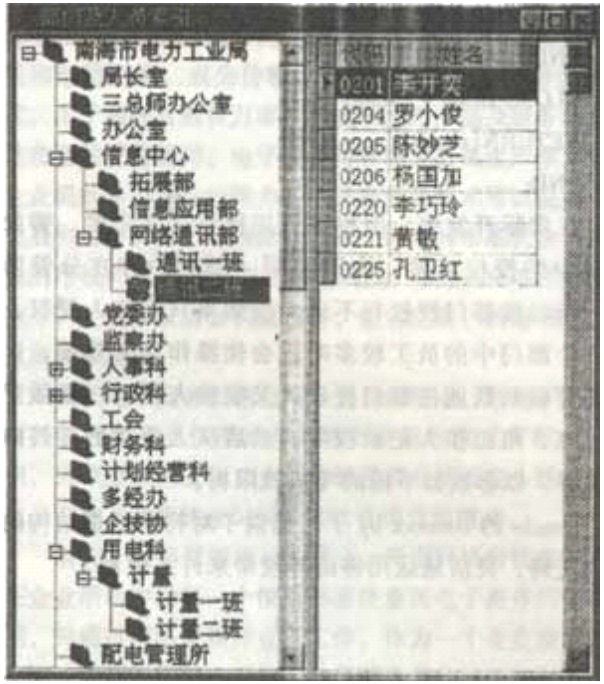
右下角提示信息窗口

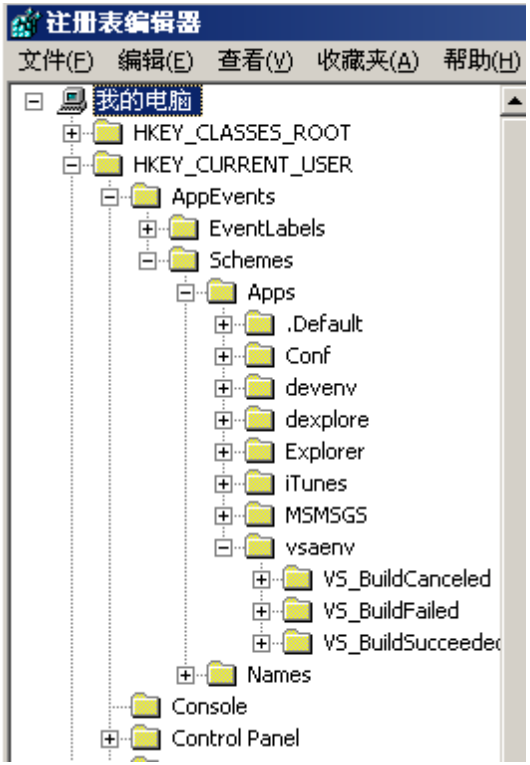
- Windows坐标系：左上角为原点；窗口的Left值为窗口左上角的横坐标，Top值为窗口左上角的纵坐标；窗口的Width为宽度，Width为高度。
- 取得屏幕工作区 Screen.GetWorkingArea(this)
- 将窗口显示在右下角，简单的数学运算。

滑动提示窗口

- 创建一个Timer，每隔一段时间修改一下Top的值，当全部显示出来的时候禁用Timer。
- 对外提供提供SlideShow(String caption,String msg)方法
- 超时自动关闭。
- 滑动隐藏。Timer接力。
- 鼠标点击则取消“超时自动隐藏”。

树状结构数据

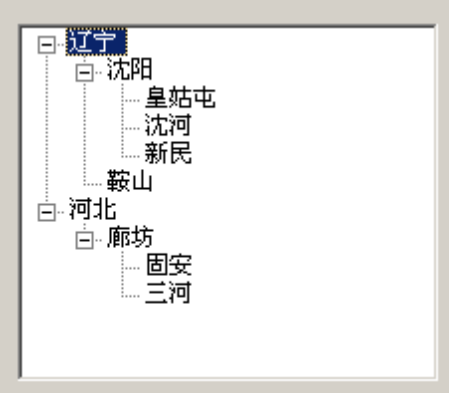

南海市电力工业局
局长室
三总师办公室
办公室
信息中心
拓展部
信息应用部
网络通讯部
通讯一班
党委办
监察办
人事科
行政科
工会
财务科
计划经营科
多经办
企技协
用电科
计量
计量一班
计量二班
配电管理所
0201
0204 罗小俊
0205 陈妙芝
0206 杨国加
0220 李巧玲
0221 黄敏
0225 孔卫红


注册表编辑器
文件(F) 编辑(E) 查看(V) 收藏夹(A) 帮助(H)
我的电脑
HKEY_CLASSES_ROOT
HKEY_CURRENT_USER
AppEvents
EventLabels
Schemes
Apps
.Default
Conf
devenv
dexplore
Explorer
iTunes
MSMSGS
vsaenv
VS_BuildCanceled
VS_BuildFailed
VS_BuildSucceeded
Names
Console
Control Panel


插入(I) 格式(O) 工具(T) 形状(S) 窗口
新建页(N)...
字段(F)...
符号(S)...
形状屏幕提示(T)...
注释(M)
图片(P)
CAD 绘图(D)...
控件(C)...
对象(O)...
超链接(H)... Ctrl+K
剪贴画(C)...
来自文件(F)...
来自扫描仪或
图表(H)
公式(E)...

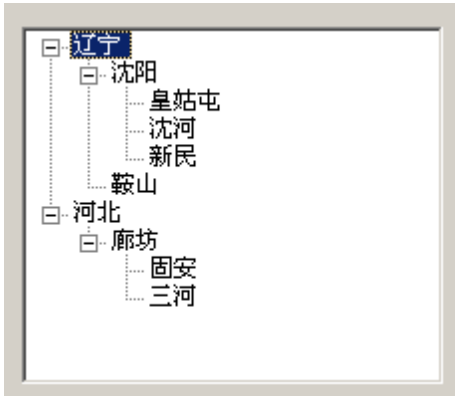
TreeView控件

- 树的基本概念：父节点、子节点、兄弟节点、根节点
- 编辑节点，节点名字Name，节点文字Text
- 动态添加节点（VS编辑界面中有增加根节点、增加子节点两个按钮）：
  - 向根节点下增加子节点：treeView.Nodes.Add，它的返回值就是创建的节点对象。treeView.Nodes代表根节点的所有子节点。
  - 向TreeNode增加子节点，treeNode.Nodes.Add，它的返回值就是创建的节点对象。treeNode.Nodes代表节点的所有子节点。



练习：构建省市县三级树

- 首先treeView.Nodes.Add添加省节点。并且把添加的节点TreeNode对象放到treeNodeLiaoning等变量中
- 向treeNodeLiaoning等省级节点下添加市节点，并且把添加的节点TreeNode对象放到treeNodeShenYang等变量中
- 向treeNodeShenYang等市节点下增加区县节点。



TreeView深入

- 选择项发生变化的时候在TextBox中显示当前选择项的值。AfterSelect事件。
- TreeNode的Tag，TreeNode中Text只能盛放显示的值，可以用Tag属性给节点关联一个对象。 Tag属性是Object类型的，因此可以关联任何的对象。案例：省市县显示当前项的邮编
- 练习：选择节点的时候显示市长、人口（int）类型。

容器控件

- GroupBox
- Panel
- TabControl（增删Tab页，设定激活的页）

其他常用控件

- RadioButton：Checked。CheckedChanged和Click事件的区别
- CheckBox
- CheckedListBox
- DateTimePicker。
- PictureBox
- 作业：开发注册页面。数据合法性判断：Email、密码强度实时显示。带协议、同意协议CheckBox。

文件对话框

- 打开文件对话框 OpenFileDialog
- 两种创建方式：可视化拖放、new
- 显示对话框的方式：ShowDialog
- ShowDialog的返回值DialogResult类型
- 通过FileName属性得到选择的文件名
- 属性：Title、InitialDirectory
- 文件过滤器 Filter：文本文件|*.txt|All|*.*
- SaveFileDialog。OpenFileDialog相同的成员：Title、ShowDialog、FileName等

列表(非范型)

- 数组的局限性：无法对数据中的数据进行动态的增删。
- ArrayList（列表）。Add、Clear、Contains、Count、IndexOf、Insert、Remove、RemoveAt、Reverse、Sort、索引器。
- 作业1：输入整数，然后反序输出
- 作业2：两个列表的连接、合并。
- 作业3：分拣奇偶数，奇数在左，偶数在右，面试题

Hashtable

- 简繁体翻译，为什么效率这么高
- HashSet

范型

- 用ArrayList的麻烦的地方：数据放进去就不知道是什么类型的了；不能防止非法类型数据的放入；将ArrayList返回给其他函数，会令调用者很困惑。
- List<int> 除此除外并无不同。
- IEnumerable<int>、ICollection<int>
- 除了List类外，还有很多类支持范型，比如Dictionary<K,V>等，待定的类型还可以不止一个。简繁体翻译器。

Dictionary

- Key-value Pair 键值对
- 作业：Dictionary性能测试
- Dictionary<K,V>:
  - Add：添加，如果重复，则报错
  - 索引器：可以重复设置，即使不存在也没关系，如果重复则保留最后一次的值
  - ContainsKey，判断是否存在这个Key

对话框、字符串、数组综合案例

选择成绩文件。计算平均分，查找某个学生的成绩。File.ReadAllLines

菜单、工具栏

- 主菜单（MenuStrip）：分割线、快捷键、图标、子菜单、事件响应。
- 右键菜单（ContextMenuStrip）
- 工具栏ToolStrip
- 状态栏 StatusStrip

菜单工具栏案例

- 案例：自己动手写记事本。打开、保存、另存为、退出、剪切、复制、粘贴
- 带工具栏
- 练习：标题显示文件名
- 练习：状态栏中显示当前时间

调试（*理解）Debug

- 暂停（不知道程序现在运行到哪里了）
- 断点、设置断点 breakpoint
- 条件断点
- 命中次数断点
- 查看变量的值，查看表达式的值。红色、黑色

变量作用域

```
        static void Main(string[] args)
        {
            string name = "tom";
            SayHello();
        }
        static void SayHello()
        {
            Console.WriteLine("我的姓名是：
{0}",name);
        }
```

```
        static void Main(string[] args)
        {
            string name = "tom";
            SayHello();
        }
        static void SayHello()
        {
            string name = "jerry";
            Console.WriteLine("我的姓名是：
{0}",name);
        }
```

变量名：变量名就像试验实例容器的编号，在一个实验室中，编号不能重复。如果去了更高一级的实验室，就不能和下属的实验室的容易编号重复。只要不出实验室，不同实验室之间的容器编号可以重复。

变量作用域2

```
string name = "tom";
if (name == "tom")
{
    string name = "jerry";
}
```

```
for (int i = 0; i < 10; i++)
{
    string name =
i.ToString();
}

Console.WriteLine("{0}",name);
```

```
for (int i = 0; i < 10; i++)
{
    string name =
i.ToString();
}
Console.WriteLine("{0}",i);
```

```
int i;
for (i = 0; i < 10; i++)
{
    string name = i.ToString();
}
Console.WriteLine("{0}",i);
```

```
static void SayHello(string name)
{
    string name = "lily";
    Console.WriteLine("我是{0}",name);
}
```

```
int i = 3;
if (i > 3)
{
    if (i < 100)
    {
        int i = 10;
    }
}
```

```
if (s=="a")
{
    int i = 0;
}
if (s=="b")
{
    int i = 0;
}
```

在同一个括号内变量定义不能重名，括号可以嵌套，但是通过函数的调用并不在这个范围内

传智播客 www.itcast.cn —高级软件人才实作培训专家！

C#面向对象基础

讲师：杨中科

面向对象概念 OOP

- 面向对象不是取代面向过程的。
- 类、对象。“人”是类，“张三”是“人”这个类的对象。类是抽象的，对象是具体的。按钮就是类，某个按钮就是对象。对象可以叫做类的实例。类就像int，对象就像10。字段（和类相关的变量），字段就是类的状态。人这个类有姓名、年龄、身高等字段。类不占内存，对象才占内存。
- 方法，方法就是类能够执行的动作，比如问好、吃饭等。
- 类的继承，类之间可以有继承关系，比如“电脑”类可以从“电器”类继承，这样的好处是“电脑”类只需要定义自己特有的字段、方法就可以，也就是只要定义内存大小、CPU型号这些字段或者弹出光驱等方法就可以。父类、基类。电脑类是电器类的子类。重用。
- 面向对象的三个特性：封装、继承、多态。
- 没有面向对象的世界中的难题。

定义类

- class Person{ }

定义字段：

```
class Person
{
   public string Name;
    public int Age;
    public void SayHello()
    {
        Console.WriteLine("你好,我是{0}",Name);
    }
}
初始化对象，调用方法：
        Person tom = new Person();
        tom.Name="tom";
        tom.SayHello();
Console.WiteLine("{0}",tome.Name)
```

定义方法（类的函数）：

```
class Person
{
    public void SayHello()
    {
        Console.WriteLine("你好,我是{0}",Name);
    }
}
初始化对象，调用方法：
        Person tom = new Person();
        tom.SayHello();
```

对象必须调用构造函数初始化“new Person()”后才能用，不能Person tom这样声明;就调用。

一个类可有有多个实例。

类就是把一系列相关的变量（状态）、行为定义为一个整体。

课上练习

- 聊天机器人
- 练习要求：自己把我的程序重新写出来，性别的时候说中文，对异常情况（错误的喂饭数字，喂的太多撑死了），一开始通过1、2数字选择多个聊天机器人。

成员访问级别

- 字段、方法、属性（后面讲）都可以叫做类的成员，他们都需要定义访问级别。访问级别的用处在于控制成员在哪些地方可以被访问，这样达到面向对象中“封装”的目的。
- 几个访问级别：public（任何地方都可以访问）；private（默认级别。只能由本类中的成员访问）。还有internal、protected两个级别，以后会讲。

```
class Person
{
   private string Name;

   public void GiveName(string name)
{
   Name = name;//拒绝不好听的名字
}
    public void SayHello()
    {
        Console.WriteLine("你好,我是
{0}",Name);
    }
}
```

```
Person tom = new Person();
tom.GiveName("tom");
 //tom.Name="tom";
 tom.SayHello();
```

永远不要把字段public

属性

惯用法：属性开头字母大写，字段开头字母小写

```
class Person
{
    private int age;
    public int Age
    {
        get{return age;}
        set{age=value;}
    }
    public void SayHello()
    {
        Console.WriteLine("我的年龄是{0}",Age);
    }
}
```

- 只用set或者只用get就可以定义只读或者只写属性（只写的不常见）例子
- 可以为set、get设置访问级别
- 例子，限制非法值的设置
- （.Net3.x）简化set、get：public int Age{get;set;}。适合于set、get中没有特殊逻辑代码的情况。
- 看似字段、不是字段，可以进行非法值检验。

构造函数

- 默认构造函数
- 重载构造函数
- //重载构造函数之间互相调用

继承

```
public class Chinese:Person
{
    public string HuKou{get;set;}
    public void KongFu(){....}
}
Chinese p1 = new Chinese();
p1.Name="李雷";
p1.HuKou="北京市朝阳区";
p1.SayHello();
p1.KongFu();
```

定义类的时候不指定父类，则是Object类。Object类是任何类的直接或者间接父类。（不严谨）

虚方法

```
public class Person
{
    public virtual void SayHello()
    {
        Console.WriteLine("我的年龄是{0}",Age);
    }
}
public class Chinese : Person
{
    public string HuKou { get; set; }
    public override void SayHello()
    {
        Console.WriteLine("小生虚度光阴{0}载", Age);
    }
} Korean类。
Chinese  p1 = new Chinese();
p1.SayHello();
```

案例，覆盖Object类的ToString方法来自定义显示的字符串。

在子类方法中调用父类的方法 base.SayHello();

子类中的构造函数；

多态（*理解）

```
Person p =  new Korean();
p.SayHello();

Object obj = new Chinese();
string s = obj.ToString();
obj = p;
s = obj.ToString();
```

```
隐藏父类方法（尽量避免，不推荐这么做）
public class Chinese : Person
   {
      public string HuKou { get; set; }
      public void SayHello()
      {
         Console.WriteLine("小生虚度光阴{0}载", Age);
      }
   }
Chinese p1 = new Chinese();
p1.SayHello();
Person p2 = new Chinese();
p2.SayHello();
```

多态的好处（*理解）

```
public Person
    GiveMeAPerson()
{
    return new Chinese();
}
Person p =
    GiveMeAPerson();
p.SayHello();
```

对象的引用

i1=3;i2=i1;i1++;与p1=new Person();p2=p1;p1.i++;

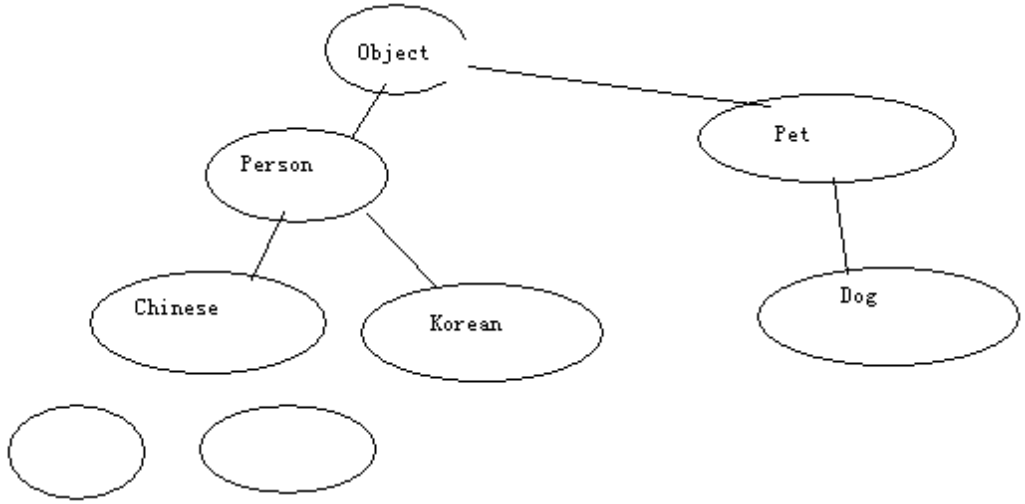
- int、string、decimal、bool、byte等基础类型是传递拷贝；对象则是传递引用。因为基础类型不怎么占内存，而对象则比较占内存。
- 函数间传递对象。
- 为对象变量重新赋值

对象的隐式转换和显式转换

```
//隐式转换，把子类变量赋值给父类变量
Chinese ch= new  Chinese();
Person p = ch;
//显式转换，把父类变量赋值给子类变量
Person p = new Chinese();
Chinese ch = (Chinese)p;
//如果对象不在同一个继承树路径上
//则不能强制类型转换
Dog g = new Dog();
Chinese ch = (Dog)g;//错误
//is运算
if(p is Chinese)
{
   Console.WriteLine("中国人");
}
else if(p is Korean)
{
   Console.WriteLine("韩国人");
}
```

```
//as 运算符
Chinese ch = p as Chinese ;
if(ch!=null)
{
   Console.WriteLine("中国人");
}
Korean ch = p as Korean ;
if(ch!=null)
{
   Console.WriteLine("韩国人");
}
//可以解决强制类型转换的不在继承树路
      径上的问题。C#中不可以
Dog g = new Dog();
Chinese ch = g as Chinese ;
```

协变和逆变


Object
Person
Pet
Chinese
Korean
Dog

抽象类、接口

```
public abstract class Person
{
    public abstract void SayHello();
}
```

一旦类中定义了一个抽象方法，那么这个类必须声明为抽象类

抽象类不能用new初始化实例。abstract 方法不能定义方法体。

```
public interface  Flyable
{
    void Flay();
}
```

Walkable

接口方法不要public。

1、一个类只能继承自一个类，但是可以实现多个接口

2、一个抽象类中可以定义实现代码，但是接口不能定义实现代码。

3、接口的多态特性、类型转换、is、as 和类基本一样。举例

异常与异常处理

- 传统的错误表示方式：错误码。举例。
- 错误码的缺点：不处理则很难发现，每次处理则很麻烦；难以看出错误的原因；容易使得程序进入不确定状态。
- try catch。Exception ex 异常也是对象。
- Exception 类主要属性：Message、StackTrace
- 发生异常后程序默认就退出了，后续代码不会被执行。catch以后的代码则会继续执行。
- 不要吃掉异常
- 扔出自己的异常

常量与静态成员

- const常量。常量名要大写。
- 全局变量。static类变量。
- 不用new就能用的方法：static方法，static方法其实就是普通函数
- 在static方法中可以调用其他static方法、字段、属性，但是不能调用非static方法、字段、属性。在非static方法中可以调用static的方法、字段。

其他oop问题

- namespace，解决类重名问题，就是文件夹。using，引入其他类。System.Collections.ArrayList，快速引入的方法，右键→解析。
- 索引器。public string this[int index]{get { return ""; }set { }}

列表(非范型)

- 数组的局限性：无法对数据中的数据进行动态的增删。
- ArrayList（列表，排队）。Add、Clear、Contains、Count、IndexOf、Insert、Remove、RemoveAt、Reverse、Sort、ToArray（再没关系）、索引器。
- 作业1：输入整数，排序，然后输出
- 作业2：两个列表的连接、合并。
- 作业3：分拣奇偶数，MS面试题

集合接口

- ArrayList实现了IList, ICollection, IEnumerable接口。
- foreach 遍历ArrayList，实现了IEnumerable接口的都可以用foreach进行遍历。
- 不止ArrayList实现了这些接口。

范型

- 用ArrayList的麻烦的地方：数据放进去就不知道是什么类型的了；不能防止非法类型数据的放入；将ArrayList返回给其他函数，会令调用者很困惑。
- List<int> 除此除外并无不同。
- IEnumerable<int>、ICollection<int>
- 除了List类外，还有很多类支持范型，比如Dictionary<K,V>等，待定的类型还可以不止一个。简繁体翻译器。

Dictionary

- Key-value Pair 键值对
- 作业：Dictionary性能测试
- Dictionary<K,V>:
  - Add：添加，如果重复，则报错
  - 索引器：可以重复设置，即使不存在也没关系，如果重复则保留最后一次的值
  - ContainsKey，判断是否存在这个Key

传智播客 www.itcast.cn —高级软件人才实作培训专家！

Windows编程基础

讲师：杨中科

第一个Windows程序

- WinForm
- 新建一个Windows项目
- 输入姓名窗口标题显示问好
- 添加、删除、移动、缩放控件
- 控件、属性、事件、事件处理方法、控件的名字（控件的实例；对象名）

加法计算器

- 简单的加法计算器

int.TryParse

MessageBox.Show

- 练习：加法计算器，考虑异常情况，聚焦。

TextBox

TextBox：Multiline、PasswordChar

控件名要有含义、控件名前缀的“潜规则”。按钮Button：btn；文本框TextBox：txt；复选框CheckBox：cb。

案例1：登录界面。登录错误三次退出程序。（易错点：局部变量与类变量）

案例2：修改密码

IDictionary与火星文翻译器

ComboBox

- ComBox
- SelectedIndex：没有任何选中的时候是-1，否则是序号（0开始）
- 练习1：简单的四则运算器
- 三种风格。string s = string.Format("{0}你好", textBox1.Text);//推荐
- 响应选择改变事件
- 练习2：省份选择

ListBox

- ListBox：SelectedIndex、Items、SelectedIndexChanged事件。
- 案例1：增删查ListBox
- ListBox：SelectedIndices、SelectedItem、SelectedItems、SelectionMode；
- 案例2：人员选择。补充问题：删除自动生成的事件代码。先在事件视图中删除方法名，再到代码中删除；先删除form.cs中的，再去手动删除designer.cs中。多选的方式，设置SelectionMode为MultiExtend

计时器Timer

- Timer。每隔一段时间触发一个事件。不可视控件。Inerval、Enabled。Tick事件。
- 计量单位：ms 1秒=1000毫秒
- DateTime
- 案例：小时钟。DateTime.Now.ToString();
- 案例：走马灯
- 案例：QQ消息窗口

容器控件

- GroupBox
- Panel
- TabControl（增删Tab页，设定激活的页）

其他常用控件

- RadioButton：Checked。CheckedChanged和Click事件的区别
- CheckBox
- CheckedListBox
- DateTimePicker。
- PictureBox
- 作业：开发注册页面。数据合法性判断：Email、密码强度实时显示。带协议、同意协议CheckBox。

文件对话框

- 打开文件对话框 OpenFileDialog
- 两种创建方式：可视化拖放、new
- 显示对话框的方式：ShowDialog
- ShowDialog的返回值DialogResult类型
- 通过FileName属性得到选择的文件名
- 属性：Title、InitialDirectory
- 文件过滤器 Filter：文本文件|*.txt|All|*.*
- SaveFileDialog。OpenFileDialog相同的成员：Title、ShowDialog、FileName等

对话框、字符串、数组综合案例

选择成绩文件。计算平均分，查找某个学生的成绩。File.ReadAllLines

数据库开发及ADO.Net

讲师：杨中科

数据库概述

- 用自定义文件格式保存数据的劣势
- DBMS（DataBase Management System，数据库管理系统）和数据库。平时谈到“数据库”可能有两种含义：MSSQLServer、Oracle等某种DBMS；存放一堆数据表的一个分类（ Catalog ）。
- 不同品牌的DBMS有自己的不同的特点：MYSQL、MSSQLServer、DB2、Oracle、Access、Sybase等。对于开发人员来讲，大同小异
- SQL<>SQLServer<>MSSQLServer。最常见的错误。
- 除了Access、SQLServerCE等文件型数据库之外，大部分数据库都需要数据库服务器才能运行。学习、开发时是连接本机的数据库，上线运行时是数据库运行在单独的服务器。

数据库中的概念

- Catalog（分类）（又叫数据库DataBase、表空间TableSpace），不同类的数据应该放到不同的数据库中
  - 便于对各个Catalog进行个性化管理
  - 避免命名冲突
  - 安全性更高
- Table（表）：书都放到书架上，碗都放到橱柜中，不同类型的资料放到不同的“格子”中，将这种区域叫做“表”（Table）。不同的表根据放的数据不同进行空间的优化，找起来也方便。
- 列（Column）、字段（Field）

2003年5月入职，是产品开发部的，姓名马小虎

王二小，技术支持部，入职是2005年7月

姓名	马小虎
部门	开发部
入职时间	2008.06.06

姓名	部门	入职时间

主键（PrimaryKey）

工号	姓名	部门	入职时间
001	风姐	员工培训部	2010年7月5日
002	瘦瘦	公关部	2010年8月2日
003	憨憨	开发部	2009年3月5日

主键就是数据行的唯一标识。不会重复的列才能当主键。一个表可以没有主键，但是会非常难以处理，因此没有特殊理由表都要设定主键

主键有两种选用策略：**业务主键和逻辑主键**。业务主键是使用有业务意义的字段做主键，比如身份证号、银行账号等；逻辑主键是使用没有任何业务意义的字段做主键，完全给程序看的，业务人员不会看的数据。因为很难保证业务主键不会重复（身份证号重复）、不会变化（帐号升位），因此推荐用逻辑主键。

表间关联、外键（ForeignKey）

商品名	价格	生产厂家	厂家地址	厂家电话
大大香瓜子	5.00	大大食品厂	恰恰大街300号	010-123456
大大开心果	15.00	大大食品厂	恰恰大街300号	010-123456
苦咖啡	2	伊利食品厂	内蒙古伊利路1号	400400400
随变	3	伊利食品厂	内蒙古伊利路1号	400400400
冰工厂	1	伊利食品厂	内蒙古伊利路1号	400400400

商品名	价格	厂家编号
大大香瓜子	5.00	001
大大开心果	15.00	001
苦咖啡	2	002
随变	3	002
冰工厂	1	002

编号	名称	地址	电话
001	大大食品厂	恰恰大街300号	010-123456
002	伊利食品厂	内蒙古伊利路1号	400400400

SQLServer的管理

- 需要安装SQLServer2005或者SQLServer2008，若要使用SQLServer管理工具进行开发还要安装SQL Server Management Studio，还可以使用VisualStudio进行管理
- 使用免费的SQLServerExpress版本，Express版本的服务器名称.\SQLEXPRESS，对于开发人员来讲和其他版本没有区别。
- SQLServer的两种验证方式：用户名验证和Windows验证，开发时用Windows验证就行。
- 开发人员关注点在开发上，而不是配置、备份等之上，那是DBA做的事情。
- 创建数据库，创建表，设置主键
- SQLServer2008ManagementStudio中：编辑200行。05中：打开表。
- 常用字段类型：bit(可选值0、1)、datetime、int、varchar、nvarchar（可能含有中文用nvarchar）
- Nvarchar(50)、Nvarchar(MAX)
- varchar、nvarchar 和char(n)的区别： char(n)不足长度n的部分用空格填充。Var：Variable，可变的。

SQL语句入门

- SQL语句是和DBMS“交谈”专用的语句，不同DBMS都认SQL语法。
- SQL语句中字符串用单引号。
- SQL语句是大小写不敏感的，不敏感指的是SQL关键字，字符串值还是大小写敏感的
- 创建表、删除表不仅可以手工完成，还可以执行SQL语句完成，在自动化部署、数据导入中用的很多，CREATE TABLE T_Person(Id int NOT NULL,Name nvarchar(50),Age int NULL)、Drop table T_Person1
- 简单的Insert语句。INSERT INTO T_Person(Id,Name,Age) VALUES(1,'Jim',20)
- （*） SQL主要分DDL（数据定义语言）和DML（数据操作语言）两类。Create Table、Drop Table、Alter Table等属于DDL，Select、Insert、Update、Delete等属于DML

主键选择

- SQLServer中两种常用的主键数据类型：int（或bigint）+标识列（又称自动增长字段）；uniqueidentifier（又称Guid、UUID）
- 用标识列实现字段自增可以避免并发等问题，不要开发人员控制自增。用标识列的字段在Insert的时候不用指定主键的值。将字段的“是标识列”设置为“是”，一个表只能有一个标识列。
- Guid算法是一种可以产生唯一标识的高效算法，它使用网卡MAC、地址、纳秒级时间、芯片ID码等算出来的，这样保证每次生成的GUID永远不会重复，无论是同一个计算机上还是不同的计算机。在公元3400年以前产生的GUID与任何其他产生过的GUID都不相同。SQLServer中生成GUID的函数newid()，.Net中生成Guid的方法：Guid.NewGuid()，返回是Guid类型。
- （*）Int自增字段的优点：占用空间小、无需开发人员干预、易读；缺点：效率低；数据导入导出的时候很痛苦。
- （*）Guid的优点：效率高、数据导入导出方便；缺点占用空间大、不易读。
- 业界主流倾向于使用Guid。

数据插入

- Insert语句可以省略表名后的列名，但是不推荐。
- 如果插入的行中有些字段的值不确定，那么Insert的时候不指定那些列即可。
- 给可以给字段默认值，如果Guid类型主键的默认值设定为newid()就会自动生成，很少这么干。
- 主键：insert into Person3(Name,Age) values('lily',38);insert into Person4(Id,Name,Age) values(newid(),'tom',30);

数据更新

- 更新一个列：UPDATE T_Person Set Age=30
- 更新多个列： UPDATE T_Person Set Age=30,Name='tom'
- 更新一部分数据： UPDATE T_Person Set Age=30 where Name='tom'，用where语句表示只更新Name是'tom'的行，注意SQL中等于判断用单个=，而不是==。
- Where中还可以使用复杂的逻辑判断UPDATE T_Person Set Age=30 where Name='tom' or Age<25，or相当于C#中的||（或者）
- update Person1 set NickName=N'二十岁'
- where (Age>20 and Age<30) or(Age=80)
- Where中可以使用的其他逻辑运算符：or、and、not、<、>、>=、<=、!=（或<>）等

数据删除

- 删除表中全部数据：DELETE FROM T_Person。
- Delete只是删除数据，表还在，和Drop Table不同。
- Delete 也可以带where子句来删除一部分数据：DELETE FROM T_Person WHERE FAge > 20

数据检索

- 执行备注中的代码创建测试数据表。
- 简单的数据检索 ： SELECT * FROM T_Employee
- 只检索需要的列 ： SELECT FNumber FROM T_Employee 、SELECT FName,FAge FROM T_Employee
- 列别名：SELECT FNumber AS 编号,FName AS 姓名 ,FAge AS Age111 FROM T_Employee
- 使用where检索符合条件的数据：SELECT FName FROM T_Employee WHERE FSalary<5000。故事：新员工的数据检索噩梦。
- 还可以检索不与任何表关联的数据：select 1+1;select newid();select getdate();

数据汇总

- SQL聚合函数：MAX（最大值）、MIN（最小值）、AVG（平均值）、SUM （和）、COUNT（数量）
- 大于25岁的员工的最高工资 ：SELECT MAX(FSalary) FROM T_Employee WHERE FAge>25
- 最低工资和最高工资：SELECT MIN(FSalary),MAX(FSalary) FROM T_Employee

数据排序

- ORDER BY子句位于SELECT语句的末尾，它允许指定按照一个列或者多个列进行排序，还可以指定排序方式是升序（从小到大排列，ASC）还是降序（从大到小排列，DESC）。
- 按照年龄升序排序所有员工信息的列表：SELECT * FROM T_Employee ORDER BY FAge ASC
- 按照年龄从大到小排序，如果年龄相同则按照工资从大到小排序：SELECT * FROM T_Employee ORDER BY FAge DESC,FSalary DESC
- ORDER BY子句要放到WHERE子句之后：SELECT * FROM T_Employee WHERE FAge>23 ORDER BY FAge DESC,FSalary DESC

通配符过滤

- 通配符过滤使用LIKE 。
- 单字符匹配的通配符为半角下划线“_”，它匹配单个出现的字符。以任意字符开头，剩余部分为“erry”：SELECT * FROM T_Employee WHERE FName LIKE '_erry'
- 多字符匹配的通配符为半角百分号“%”，它匹配任意次数（零或多个）出现的任意字符。“k%”匹配以“k”开头、任意长度的字符串。检索姓名中包含字母“n”的员工信息：SELECT * FROM T_Employee WHERE FName LIKE '%n%'

空值处理

- 数据库中，一个列如果没有指定值，那么值就为null，这个null和C#中的null，数据库中的null表示“不知道”，而不是表示没有。因此select null+1结果是null，因为“不知道”加1的结果还是“不知道”。
- SELECT * FROM T_Employee WHERE FNAME=null ；SELECT * FROM T_Employee WHERE FNAME!=null ；都没有任何返回结果，因为数据库也“不知道”。
- SQL中使用is null、is not null来进行空值判断： SELECT * FROM T_Employee WHERE FNAME is null ； SELECT * FROM T_Employee WHERE FNAME is not null ；

多值匹配

- SELECT FAge,FNumber,FName FROM T_Employee WHERE FAge IN (23,25,28)
- 范围值：SELECT * FROM T_Employee WHERE FAGE>=23 AND FAGE <=27 ； SELECT * FROM T_Employee WHERE FAGE BETWEEN 23 AND 27

数据分组

- 按照年龄进行分组统计各个年龄段的人数：SELECT FAge,Count(*) FROM T_Employee GROUP BY Fage
- GROUP BY子句必须放到WHERE语句的之后
- 没有出现在GROUP BY子句中的列是不能放到SELECT语句后的列名列表中的（聚合函数中除外）
  - 错误：SELECT FAge,FSalary FROM T_Employee GROUP BY FAge
  - 正确：SELECT FAge,AVG(FSalary) FROM T_Employee GROUP BY FAge

Having语句

- 在Where中不能使用聚合函数，必须使用Having，Having要位于Group By之后，SELECT FAge,COUNT(*) AS 人数 FROM T_Employee
- GROUP BY FAge
- HAVING COUNT(*)>1
- 注意Having中不能使用未参与分组的列，Having不能替代where。作用不一样，Having是对组进行过滤。

限制结果集行数

- **select** top 5 * **from** T_Employee **order by** FSalary **Desc**
- （*）检索按照工资从高到低排序检索从第六名开始一共三个人的信息： **SELECT** top 3 * **FROM** T_Employee **WHERE** FNumber **NOT IN** (**SELECT** TOP 5 FNumber **FROM** T_Employee **ORDER BY** FSalary **DESC**) **ORDER BY** FSalary **DESC**
- SQLServer2005后增加了Row_Number函数简化实现，后面会讲。

去掉数据重复

- 执行备注中的**SQL**语句，**Alter**和**Insert**单独执行。
- **SELECT** FDepartment **FROM** T_Employee →**SELECT DISTINCT** FDepartment **FROM** T_Employee
- **DISTINCT**是对整个结果集进行数据重复处理的，而不是针对每一个列，因此下面的语句并不会只保留Fdepartment进行重复值处理：
  SELECT DISTINCT FDepartment,FSubCompany **FROM** T_Employee

常见问题

- 1、SQLServer2008 Management Studio中点击【执行】按钮，而不是绿色箭头的调试按钮。
- 2、如果机器上安装了VisualStudio2010或者SQLServer2008，需要安装SQLServer2005 Management Studio
- 3、 SQLServer2008 Management Studio中是“修改前200条”
- 4、Alter增加字段之后要关闭窗口重新打开才能看到新增加的列。

联合结果集

- 执行备注中的代码
- 简单的结果集联合：
  - SELECT FNumber,FName,FAge FROM T_Employee UNION SELECT FIdCardNumber,FName,FAge FROM T_TempEmployee
- 基本的原则：每个结果集必须有相同的列数；每个结果集的列必须类型相容。
- **SELECT** FNumber,FName,FAge,FDepartment **FROM** T_Employee **UNION SELECT** FIdCardNumber,FName,FAge,'临时工，无部门' **FROM** T_TempEmployee

Union all

SELECT FName **FROM** T_Employee **UNION**
SELECT FName **FROM** T_TempEmployee。
UNION合并两个查询结果集，并且将其中完全重复的数据行合并为一条

SELECT FName **FROM** T_Employee
UNION ALL
SELECT FName **FROM** T_TempEmployee

Union因为要进行重复值扫描，所以效率低，因此如果不是确定要合并重复行，那么就用**UNION ALL**

案例1

- 要求查询员工的最低年龄和最高年龄，临时工和正式员工要分别查询
- **SELECT** '正式员工最高年龄',**MAX**(FAge) **FROM** T_Employee
- **UNION ALL**
- **SELECT** '正式员工最低年龄',**MIN**(FAge) **FROM** T_Employee
- **UNION ALL**
- **SELECT** '临时工最高年龄',**MAX**(FAge) **FROM** T_TempEmployee
- **UNION ALL**
- **SELECT** '临时工最低年龄',**MIN**(FAge) **FROM** T_TempEmployee

案例2

- 查询每位正式员工的信息，包括工号、工资，并且在最后一行加上所有员工工资额合计。
- **SELECT** FNumber,FSalary **FROM** T_Employee
- **UNION ALL**
- **SELECT** '工资合计',**SUM**(FSalary) **FROM** T_Employee

数字函数（*）

- 执行备注中的代码
- ABS()：求绝对值。
- CEILING()：舍入到最大整数 。3.33将被舍入为4、2.89将被舍入为3、-3.61将被舍入为-3。Ceiling→天花板
- FLOOR()：舍入到最小整数。3.33将被舍入为3、2.89将被舍入为2、-3.61将被舍入为-4。Floor→地板。
- ROUND()：四舍五入。舍入到“离我半径最近的数”。Round→“半径”。Round(3.1425,2)。

字符串函数（*）

- LEN()：计算字符串长度
- LOWER()、UPPER ()：转小写、大写
- LTRIM()：字符串左侧的空格去掉
- **RTRIM** ()：字符串右侧的空格去掉
- LTRIM(RTRIM('　　bb　　'))
- SUBSTRING(string,start_position,length)

参数string为主字符串，start_position为子字符串在主字符串中的起始位置，length为子字符串的最大长度。SELECT SUBSTRING('abcdef111',2,3)

日期函数

- GETDATE()：取得当前日期时间
- DATEADD (datepart , number, date)，计算增加以后的日期。参数date为待计算的日期；参数number为增量；参数datepart为计量单位，可选值见备注。DATEADD(DAY, 3,date)为计算日期date的3天后的日期，而DATEADD(MONTH ,-8,date)为计算日期date的8个月之前的日期
- DATEDIFF (datepart , startdate , enddate)：计算两个日期之间的差额。datepart 为计量单位，可取值参考DateAdd。
- 统计不同工龄的员工的个数：select DateDiff(year,FInDate,getdate()),count(*) from T_Employee
- group by DateDiff(year,FInDate,getdate())
- DATEPART (datepart,date)：返回一个日期的特定部分
- 统计员工的入职年份个数：select DatePart(year,FInDate),count(*)
- from T_Employee
- group by DatePart(year,FInDate)

类型转换函数

- CAST (expression AS data_type)
- CONVERT (data_type, expression)
- **SELECT** FldNumber,
- **RIGHT**(FldNumber,3) **as** 后三位,
- **CAST**(**RIGHT**(FldNumber,3) **AS INTEGER**) **as** 后三位的整数形式,
- **CAST**(**RIGHT**(FldNumber,3) **AS INTEGER**)+1 **as** 后三位加1,
- **CONVERT**(**INTEGER**,**RIGHT**(FldNumber,3))/2 **as** 后三位除以2
- **FROM** T_Person

空值处理函数

- 执行备注中的代码
- ISNULL(expression,value)：如果expression不为空则返回expression，否则返回value。SELECT ISNULL(FName,'佚名') as 姓名 FROM T_Employee

CASE函数用法1

- 单值判断，相当于switch case
- CASE expression
- WHEN value1 THEN returnvalue1
- WHEN value2 THEN returnvalue2
- WHEN value3 THEN returnvalue3
- ELSE defaultreturnvalue
- END
- 例子SELECT
- SELECT
- FName,
- (CASE FLevel WHEN 1 THEN 'VIP客户'
- WHEN 2 THEN '高级客户'
- WHEN 3 THEN '普通客户'
- ELSE '客户类型错误'
- END) as FLevelName
- FROM T_Customer

CASE函数用法2

- 测试数据在备注中

```
CASE
WHEN condition1 THEN returnvalue1
WHEN condition 2 THEN returnvalue2
WHEN condition 3 THEN returnvalue3
ELSE defaultreturnvalue
END
```

相当于if…else…else….

例子：

```
SELECT
  FName,
  FWeight,
 (CASE
  WHEN FWeight<40 THEN ‘瘦瘦'
  WHEN FWeight>50 THEN ‘肥肥'
  ELSE 'ok'
  END) as isnormal
FROM T_Person
```

练习1

- 表中有A B C三列,用SQL语句实现：当A列大于B列时选择A列否则选择B列，当B列大于C列时选择B列否则选择C列。
- select (case when a>b then a else b end),(case when b>c then b else c end) from t

练习2

单号	金额
Rk1	10
Rk2	20
Rk3	-30
Rk4	-10

将上面的表输出为如下的格式：

单号	收入	支出
Rk1	10	0
Rk2	20	0
Rk3	0	30
Rk4	0	10

练习3

- 有一张表T_Scores，记录比赛成绩
- Date　　Name　Score
- 2008-8-8　　拜仁　　胜
- 2008-8-9　　奇才　　胜
- 2008-8-9　　湖人　　胜
- 2008-8-10　　拜仁　　负
- 2008-8-8　　拜仁　　负
- 2008-8-12　　奇才　　胜
- 要求输出下面的格式：
- Name　胜　　负
- 拜仁　1　　2
- 湖人　1　　0
- 奇才　2　　0
- 数据和参考答案见备注
- 注意：在中文字符串前加N，比如N'胜'

练习

- 创建一张表，记录电话呼叫员的工作流水，记录呼叫员编号、对方号码、通话开始时间、通话结束时间。建表、插数据等最后都自己写SQL语句。
- 要求：
  - 输出所有数据中通话时间最长的5条记录。orderby datediff
  - 输出所有数据中拨打长途号码（对方号码以0开头）的总时长。like、sum
  - 输出本月通话总时长最多的前三个呼叫员的编号。
    datediff(month....),sum,order by
  - 输出本月拨打电话次数最多的前三个呼叫员的编号.group by,count(*)
  - 输出所有数据的拨号流水，并且在最后一行添加总呼叫时长
    - 呼叫员编号、对方号码、通话时长
    - .....
    - 汇总 [市内号码总时长][长途号码总时长]

索引Index

- 全表扫描：对数据进行检索（select）效率最差的是全表扫描，就是一条条的找。
- 如果没有目录，查汉语字典就要一页页的翻，而有了目录只要查询目录即可。为了提高检索的速度，可以为经常进行检索的列添加索引，相当于创建目录。
- 创建索引的方式，在表设计器中点击右键，选择“索引/键”→添加→在列中选择索引包含的列。
- 使用索引能提高查询效率，但是索引也是占据空间的，而且添加、更新、删除数据的时候也需要同步更新索引，因此会降低Insert、Update、Delete的速度。只在经常检索的字段上(Where)创建索引。
- （*）即使创建了索引，仍然有可能全表扫描，比如like、函数、类型转换等。

表连接Join

- 有客户表（T_Customers）和订单表（T_Orders）两个表，客户表字段为：Id、Name、Age，订单表字段为：Id、BillNo、CustomerId，订单表通过CustomerId关联客户表。测试数据见备注。
- select o.BillNo,c.Name,c.Age
- from T_Orders as o
- join T_Customers as c on o.CustomerId=c.Id
- join是和哪个表连接，on后是连接的关系是什么。
- 要求显示所有年龄大于15岁的顾客购买的订单号、客户姓名、客户年龄。
- 要求显示年龄大于平均年龄的顾客购买的订单
- （*）Inner Join、Left Join、Right Join

子查询

- 将一个查询语句做为一个结果集供其他SQL语句使用，就像使用普通的表一样，被当作结果集的查询语句被称为子查询。所有可以使用表的地方几乎都可以使用子查询来代替。SELECT * FROM(SELECT * FROM T2 where FAge<30)
- 执行备注中的SQL。
- 单值做为子查询：SELECT 1 AS f1,2,(SELECT MIN(FYearPublished) FROM T_Book),(SELECT MAX(FYearPublished) FROM T_Book) AS f4
- 只有返回且仅返回一行、一列数据的子查询才能当成单值子查询。下面的是错误的：SELECT 1 AS f1,2,(SELECT FYearPublished FROM T_Book)
- SELECT * FROM T_ReaderFavorite WHERE FCategoryId=(SELECT FId FROM T_Category WHERE FName='Story')

子查询

- 如果子查询是多行**单列**的子查询，这样的子查询的结果集其实是一个集合。

```
SELECT * FROM T_Reader
WHERE FYearOfJoin IN
(
select FYearPublished FROM T_Book
)
```

- 限制结果集。返回第3行到第5行的数据（ ROW_NUMBER 不能用在where子句中，所以将带行号的执行结果作为子查询，就可以将结果当成表一样用了）：

```
SELECT * FROM
(
SELECT ROW_NUMBER() OVER(ORDER BY FSalary DESC) AS rownum,
FNumber,FName,FSalary,FAge FROM T_Employee
) AS a
WHERE a.rownum>=3 AND a.rownum<=5
```

ADO.Net基础

- 程序要和数据库交互要通过ADO.Net 进行，通过ADO.Net就能在程序中执行SQL了。ADO.Net中提供了对各种不同数据库的统一操作接口。
- 直接在项目中内嵌mdf文件的方式使用SQLServer数据库（新建→数据→基于服务的数据库）。mdf文件随着项目走，用起来方便，和在数据库服务器上创建数据库没什么区别，运行的时候会自动附加（Attach）
- 双击mdf文件会在“服务器资源管理器”中打开，管理方式和在Management Studio没有什么本质不同。要拷贝mdf文件需要关闭所有指向mdf文件的连接。
- 正式生产运行的时候附加到SQLServer上、修改连接字符串即可，除此之外没有任何的区别，在“数据库”节点上点右键“附加”；在数据库节点上→任务→分离就可以得到可以拷来拷去mdf文件。
- 用的时候要在控制台、WinForm项目中在Main函数最开始的位置加入备注中的代码。ASP.Net项目中不需要。

面试题

- http://www.nowamagic.net/database/db_EmployeeDepartmentSQL.php

可能遇到的错误

- 1、由于启动用户实例的进程时出错，导致无法生成 SQL Server 的用户实例
- http://wenwen.soso.com/z/q15616823.htm
- 2、版本太低，只支持2005及以下数据库。解决：安装VisualStudio 2008 SP1
- 3、启动超时。多试几次

连接SQLServer

- 连接字符串：程序通过连接字符串 指定要连哪台服务器上的、哪个实例的哪个数据库、用什么用户名密码等。
- 项目内嵌mdf文件形式的连接字符串"Data Source=.\SQLEXPRESS;AttachDBFilename=|DataDirectory|\Database1.mdf;Integrated Security=True;User Instance=True"。“.\SQLEXPRESS”表示“本机上的SQLEXPRESS实例”，如果数据库实例名不是SQLEXPRESS，则需要修改。“Database1.mdf”为mdf的文件名。
- ADO.Net中通过SqlConnection类创建到SQLServer的连接，SqlConnection代表一个数据库连接，ADO.Net中的连接等资源都实现了IDisposable接口，可以使用using进行资源管理。执行备注中的代码如果成功了就ok。

执行简单的Insert语句

- SqlCommand表示向服务器提交的一个命令（SQL语句等），CommandText属性为要执行的SQL语句，ExecuteNonQuery方法执行一个非查询语句（Update、Insert、Delete等）

```
using (SqlCommand cmd = conn.CreateCommand())
        {
            cmd.CommandText = "Insert into
    T_Users(UserName,Password) values('admin','888888')";
            cmd.ExecuteNonQuery();
        }
```

- ExecuteNonQuery返回值是执行的影响行数
- 常犯错：

```
string username='test';
....
cmd.CommandText = "Insert into T_Users(UserName,Password)
    values(username,'888888')";
```

ExecuteScalar

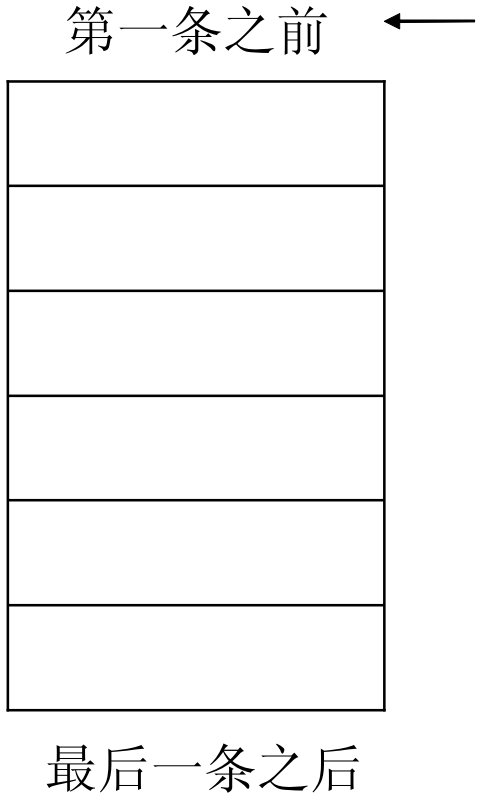
- SqlCommand的ExecuteScalar方法用于执行查询，并返回查询所返回的结果集中第一行的第一列，因为不能确定返回值的类型，所以返回值是object类型。
  - cmd.CommandText = "select count(*) from T_Users";int i = Convert.ToInt32(cmd.ExecuteScalar())
  - cmd.CommandText = "select getdate()"; DateTime dt = Convert.ToDateTime(cmd.ExecuteScalar());
- 得到自动增长字段的主键值，在values关键词前加上output inserted.Id，其中Id为主键字段名。执行结果就试插入的主键值，用ExecuteScalar执行最方便。
  - cmd.CommandText = "Insert into T_Users(UserName,Password) output inserted.Id values('admin','888888')";
  - int i = Convert.ToInt32(cmd.ExecuteScalar());

执行查询

- 执行有多行结果集的用ExecuteReader

```
SqlDataReader reader = cmd.ExecuteReader();...
while (reader.Read())
{     Console.WriteLine(reader.GetString(1));
}
```

- reader的GetString、GetInt32等方法只接受整数参数，也就是序号，用GetOrdinal方法根据列名动态得到序号
- 练习：控制台登陆程序。
- 为什么用using。Close：关闭以后还能打开。Dispose：直接销毁，不能再次使用。using在出了作用域以后调用Dispose，SqlConnection、FileStream等的Dispose内部都会做这样的判断：判断有没有close，如果没有Close就先Close再Dispose。



SQL注入漏洞攻击

- 登录判断：select * from T_Users where UserName=... and Password=...，将参数拼到SQL语句中。
- 构造恶意的Password：' or '1'='1

```
if (reader.Read())
{
    Console.WriteLine("登录成功");
}
else
 {
    Console.WriteLine("登录失败");
}
```

- 防范注入漏洞攻击的方法：不使用SQL语句拼接，通过参数赋值

查询参数

- SQL语句使用@UserName表示“此处用参数代替”，向SqlCommand的Parameters中添加参数
  - cmd.CommandText = "select * from T_Users where UserName=@UserName and Password=@Password";
  - cmd.Parameters.Add(new SqlParameter("UserName","admin"));
  - cmd.Parameters.Add(new SqlParameter("Password",password));
- 参数在SQLServer内部不是简单的字符串替换，SQLServer直接用添加的值进行数据比较，因此不会有注入漏洞攻击。

案例

- 用户界面中进行登录判断。输错三次禁止登陆，用数据库记录ErrorTimes。
- 数据导入：从文本文件导入用户信息。易错点：Parameter的重复添加
- 数据导出：将用户信息导出到文本文件。

案例

- 省市选择程序，数据全部来自于数据库：http://www.programfan.com/blog/article.asp?id=28128把createtable中的varchar改为nvarchar，在Insert语句的汉字前面加上N（查找“,'”替换为“,N'”）
  - ComboBox的显示值：Items.Add的参数是Object类型，也就是可以放任意数据类型的数据，可以设置DisplayMember属性设定显示的属性，通过SelectedItem属性取得到就是选择的条目对应的对象。例子。疑问：取出来的是Object，怎么能转换为对应的类型？变量名只是“标签”。显示的值和实际的对象不一样，在ASP.Net中也有相同的东西
  - 创建一个ProvinceItem类，将数据填充在这个对象中添加到ComboBox中。
  - 将连接字符串写在代码中的缺点：多次重复，违反了DRY（Don't Repeat Yourself）原则；如果要修改连接字符串就要修改代码。将连接字符串写在App.Config中：
    - 添加App.config文件（文件名不能改）：添加→新建项→常规→应用程序配置文件。App.config是.Net的通用配置文件，在ASP.Net中也能同样使用。
    - 在App.config中添加connectionStrings段，添加一个add项，用name属性起一个名字（比如DbConnStr），connectionString属性指定连接字符串。
    - 在“引用”节点上点右键“添加引用”，找到System.configuration。不是所有.Net中的类都能直接调用，类所在的Assembly要被添加到项目的引用中才可以。
    - ConfigurationManager.ConnectionStrings[" DbConnStr "].ConnectionString得到连接字符串。
    - 如何在部署的程序中修改配置。

案例

- IP地址归属地查询。查询结果：山东移动[菏泽]。打开文件
- 文件夹选择对话框FolderBrowserDialog；
- 按照通配符搜索目录下的文件string[] Directory.GetFiles(string path, string searchPattern, SearchOption searchOption)
- Path.GetFileNameWithoutExtension(filename)，得到文件的文件名（不要扩展名）；Path.Combine(string path1, string path2)，将两个路径合并； Path. GetExtension(string path)，得到文件的后缀；Path. GetFileName(string path)，得到文件的文件名；Path. GetFullPath(string path)：得到文件的全路径。
- 导入前先清除旧数据。如果看不到数据可能是没放Main中的代码
- 到58.com上找手机号测试。

DataSet

- 每次读取数据都创建连接、执行Command得到SqlDataReader太麻烦，让我们封装一个方法吧！
- SqlDataReader是连接相关的， SqlDataReader中的查询结果并不是放到程序中的，而是放在数据库服务器中，SqlDataReader只是相当于放了一个指针（游标），只能读取当前游标指向的行，一旦连接断开就不能再读取。这样做的好处就是无论查询结果有多少条，对程序占用的内存都几乎没有影响。
- SqlDataReader对于小数据量的数据来说带来的只有麻烦，优点可以忽略不计。ADO.Net中提供了数据集的机制，将查询结果填充到本地内存中，这样连接断开、服务器断开都不影响数据的读取。
- DataSet dataset = new DataSet(); SqlDataAdapter adapter = new SqlDataAdapter(cmd); adapter.Fill(dataset);
- SqlDataAdapter是DataSet和数据库之间沟通的桥梁。数据集DataSet包含若干表DataTable，DataTable包含若干行DataRow。foreach (DataRow row in dataset.Tables[0].Rows) row["Name"]。

SQLHelper

- 封装一个SQLHelper类方便使用，提供ExecuteDataTable(string sql,params SqlParameter[] parameters)、ExecuteNonQuery(string sql,params SqlParameter[] parameters)、ExecuteScalar(string sql,params SqlParameter[] parameters)等方法。 网上有微软提供的最全的SQLHelper类，是Enterprise Library中的一部分。
- 用SQLHelper重写登录程序
- 练习：用SQLHelper重写省市选择程序、IP地址归属地查询
- new SqlParameter("e",0)的陷阱
- sqlconnection在程序中一直保持它open可以吗？对于数据库来说，连接是非常宝贵的资源，一定要用完了就close、dispose。

DataSet的更新

- 可以更新行row["Name"] = "yzk"、删除行datatable.Rows.Remove()、新增行datatable. NewRow()。这一切都是修改的内存中的DataSet，没有修改数据库。
- 可以调用SqlDataAdapter的Update方法将对DataSet的修改提交到数据库，Update方法有很多重载方法，可以提交整个DataSet、DataTable或者若干DataRow。但是需要为SqlDataAdapter提供DeleteCommand、UpdateCommand、InsertCommand它才知道如何将对DataSet的修改提交到数据库，由于这几个Command要求的格式非常苛刻，因此开发人员自己写非常困难，可以用SqlCommandBuilder自动生成这几个Command，用法很简单：new SqlCommandBuilder(adapter)。查看生成的Command（没有直接赋值给SqlDataAdapter，看SqlCommandBuilder的）。SqlCommandBuilder要求表必须有主键。
- (*)通过DataRow的RowState可以获得行的状态（删除、修改、新增等）；调用DataSet的GetChanges()方法得到变化的结果集，降低传递的资源占用。

可空数据类型

- C#中值类型（int、Guid、bool等）是不可以为空的，int i=null是错误的，因此int、bool等这些类型不能表示数据库中的“Null”。因此C#提供了“可空类型”这种语法，只要在类型后加?就构成了可空的数据类型，比如int?、bool?，这样int? i=null 就可以了。解决数据库中int可以为null，而C#中int不能为null的问题。
- 判断可空类型是否为空，i==null或者i.HasValue；得到可空变量的值，int i1=(int)i.Value或者int i1=i.Value。
- 类型转换：不可空类型赋值给可空类型无需显式转换（一定成功），可空类型赋值给不可空类型则需显式转换（不一定成功）。

弱类型DataSet的缺点

- 只能通过列名引用，dataset.Tables[0].Rows[0]["Age"]，如果写错了列名编译时不会发现错误，因此开发时必须要记着列名。
- int age = Convert.ToInt32(dataset.Rows[0]["Age"])，取到的字段的值是object类型，必须小心翼翼的进行类型转换，不仅麻烦，而且容易出错。
- 将DataSet传递给其他使用者，使用者很难识别出有哪些列可以供使用
- 运行时才能知道所有列名，数据绑定麻烦，无法使用Winform、ASP.Net的快速开发功能。
- 自己动手写强类型DataSet(类型化DataSet，TypedDataSet)，创建继承自DataSet的PersonDataSet类，封装出int? Age等属性和bool IsAgeNull等方法，向PersonDataSet中填充。

VS自动生成强类型DataSet

- 添加→新建项→数据集
- 将表从服务器资源管理器拖放到DataSet中。注意拖放过程是自动根据表结构生成强类型DataSet等类，没有把数据也拖过来，程序还是连的那个数据库，自动将数据库连接字符串写在了App.Config中。
- 代码中使用DataSet示例：CC_RecordTableAdapter adapter = new CC_RecordTableAdapter(); 如何得知Adapter的类名？选中DataSet中下半部分的Adapter，Name属性就是类名。需要右键点击类名→解析
- 取得所有的数据：adapter.GetData()，例子程序：遍历显示所有数据，i<adapter.GetData().Count;adapter.GetData()[i].Age。
- 常见问题：类名敲不对，表名+TableAdapter，表名+DataTable，表名+Row，然后用“解析”来填充类名，别照着我的代码敲。
- 常见问题：类的内部定义的类要通过包含namespace的全名来引用，不能省略。类的内部定义的类就能避免同一个namespace下类不能重名的问题。
- 强类型DataSet其实就是一种代码生成器的实现机制（DataSetPersons.Designer.cs），调用的***TableAdapter等类都是VS自动生成的，可以看到的，不要手动改生成的类代码，改xsd即可。
- GetData和Fill的区别。

强类型强在哪？

- 像使用类的属性一样使用列名，dsPerson[0].Age，可以使用VS的自动提示功能，绝对不会写错列名，写错了编译通不过。
- 将强类型DataSet传递给其他人，使用者可以轻松确定有哪些列
- int age = dsPerson[0].Age，列名的类型是明确的，避免类型转换的麻烦。
- 编译时就可以确定
- 名词：强类型DataSet（类型化DataSet），英文：Typed DataSet。
- DataSet包含DataTable、DataTable包含DataRow，强类型DataSet同样如此。查看源代码看看VS帮我们做了什么
- GetData返回是什么类型？每一行是什么类型？看类型定义即可得知。一般规律：表类型名：表名+DataTable，行类型名：表名+row，忘了也没关系：“转到定义”。

更新DataSet

- 调用Adapter的Update方法就可以将DataSet的改变保存到数据库。adapter.Update(datatable)
- 要调用Update方法更新必须设置数据库主键，后面的Delete也是如此。
- 常见错误：“当传递具有已修改行的 DataRow 集合时，更新要求有效的 UpdateCommand”，要为表设置主键。“谁都变了，唯有主键不会变”，程序要通过主键来定位要更新的行。忘了设主键怎么办？先到数据库中设置主键，然后在DataSet的对应DataTable上点右键，选择“配置”，在对话框中点击【完成】。好习惯：所有表都要设置主键！！！看看为什么会自动帮我们GetData、Update、Delete。

其他问题

- 插入新行，调用Insert方法。
- 增加字段怎么办？DataSet设计器中点【配置】，对话框中点【查询生成器】，勾选新增加的字段即可。删除字段同样如此。如果是高手也可以直接手改SQL语句。
- 要修改字段就要重新配置生成，这就是强类型DataSet的弱点，因此强类型DataSet不一定真的就是“强”，还是叫“类型化DataSet”(Typed DataSet)吧
- 常见错误：报错：数据为空。判断列的值为空的方法：Is**Null
- 为什么Select方法会填充、Update方法会更新，Insert方法会插入？没有多么神奇，看看Adapter的SelectCommand等属性，是那些SQL语句在起作用，如果有需要完全可以手工调整。

增加新的SQL语句

- 设计器的Adapter中点右键，选择“添加查询”→“使用SQL语句”，就可以添加多种类型的SQL语句。如果是“SELECT（返回行）”则SQL语句的列必须是对应DataSet类的父集合，生成两个方法：FillBy*和GetBy*，方法名根据查询语句的意义定，比如FillByAge，FillBy是将结果填充到现有DataSet，GetBy是将结果以DataSet方式返回，建议两个都生成，方便以后用。看看默认生成的GetData就明白了
- GetDataById、IncAge
- “SELECT（返回单个值）”就是ExecuteScalar
- 对于增加的SQL语句在代码中是以方法的形式使用的。方法的参数类型、顺序就是VS猜测的，如果不正确或者需要调整只要选中对应的语句，然后在【属性】窗口中修改Parameters属性即可
- 增加新的SQL语句本质论，探寻源码：不能并发调用。
- 像使用普通类的方法一样使用Adapter。SQL语句不用再写在界面代码中。这就是一种数据访问层（DAL：Data Access Layer）

TypedDataSet练习

- 用类型化DataSet重写登录、数据导入、手机号码归属地查询、省市选择等程序
- 补充问题：
  - 看mdf中数据没有改变？把那段代码放到Main中
  - 登录错误三次被锁定的问题，输入对了也不让登陆，这是错误码？就应该如此，防止暴力破解。
  - 执行UpdateQuery以后本地DataSet并没有更新。

强类型DataSet其他

- 通过查看生成的源代码的值，生成的强类型TableAdapter默认每次调用方法都是打开连接、执行、关闭连接，而如果操作之前连接已经打开则不会自动帮我们连接、关闭，因此如果想批量操作提高效率可以操作之前先自己Open，操作完毕再Close。经测试：插入三千条数据，不优化用了45秒，优化后只用一两秒。回答面试问题：如何优化访问数据库的效率。
- 常见错误：DataSet ds = new DataSet()；ds = GetData();变量名和对象。

数据绑定

- DataGridView绑定。拖放TableAdapter、DataSet、bindingSource，将bindingSource的DataSource设定为DataSet，设定DataMember属性，然后DataGridView绑定到bindingSource。在Load的时候调用TableAdapter的Fill方法将数据填充到DataSet。绑定：双方能同步感知对方的变化。
- DataGridView绑定到BindingSource， BindingSource绑定到DataSet，所以DataGridView显示的是DataSet中的数据。
- 修改列标题。
- 将保存提交到数据库，在DataGridView中修改会同步反应到DataSet中，这样只要将DataSet Update到数据库就是“保存修改”，Update，保存前要dataGridView1.EndEdit();dataGridView1.CommitEdit(DataGridViewDataErrorContexts.Commit);bindingsource1.EndEdit()已提交正在编辑的修改。
- 删除当前选择行：cCRecordBindingSource.RemoveCurrent()，只是删除DataSet中的数据，需要Update才能提交到数据库。
- 绑定单独控件，在控件属性的DataBindings中将属性绑定到BindingSource 的指定字段，这样控件中的值就会显示这个字段的值了
- 不会讲太多WinForm特有的东西。

补充

- 拖过来的控件是什么？控件就是控件类的对象，Winform中从Component类继承的类都可以拖到窗口中以控件的形式出来，本质上和new出来的对象没区别。控件的id就是变量名。
- 新建的强类型DataSet只有“生成”以后才会在工具箱中出现
- 并不是控件的所有属性都能绑定，只有显示在DataBindings节点下的属性才能绑定（*）只有标记了[Bindable(true)]的属性才能绑定。
- 只有移开焦点才会同步，并不是实时同步。
- 刷新查询窗口中的数据“执行SQL”

探究（常考）

- BindingSource是做什么的？维持当前项。这就是为什么详细控件和DataGridView会联动。试试控件绑定到不同的BindingSource。
- Adapter的作用是负责DataSet和数据库之间的数据传递。
- 绑定到ComboBox。给Person增加一个TypeId字段（表示是黄种人、白种人、黑种人还是其他人种）。ComboBox的绑定分为显示数据项的绑定、选中值的绑定两个，DataSource属性设定要数据项绑定的数据源，DisplayMember属性为显示的属性、ValueMember为值（通过SelectedValue取得）的属性；然后绑定SelectedValue属性到表的字段。
- DataGridView中的ComboBox列：设定列的ColumnType为DataGridViewComboBoxColumn为，然后其他绑定和普通ComboBox一样，由于BindingSource是维持当前项，所以记住“专BindingSource专用”

练习

- 开发**管理系统：
  http://image.baidu.com/i?ct=503316480&z=&tn=baiduimagedetail&word=%B9%DC%C0%ED%CF%B5%CD%B3+%BD%E7%C3%E6&in=19618&cl=2&lm=-1&pn=134&rn=1&di=264416700&ln=1&fr=&ic=&s=&se=&sme=0
- 数据刷新、数据检索、数据导出

传智播客 www.itcast.cn —高级软件人才实作培训专家

HTML基础加强

讲师：杨中科

课前说明

- 内容：HTML、CSS
- 目标：掌握手写HTML实现一般难度的Web页面的能力（如网站注册表单），为ASP.Net学习打基础。坚持手写HTML，可视化设计只是一种自学的手段。
- 参考书：张孝祥《JavaScript网页开发——体验式学习教程》

什么是浏览器？

- 浏览器就是接收浏览者的操作（打开一个网址、点击一个链接、点击一个按钮），然后帮浏览者去Web服务器请求网页内容(HTML格式返回)，然后展现成人眼能够看得懂的可视化页面的软件。
- IE==浏览器？IE是浏览器的一种，还有FireFox、Opera、Chrome等，注意遨游（Maxthon）、世界之窗、搜狗浏览器、360浏览器等并不是一种独立于IE的浏览器，其内核还是IE的内核，只不过换了一个外壳而已，所以用遨游的不能嘲笑用IE的，否则就露怯了。试着用WebBrowser控件自己开发一个浏览器
- 所谓的Trident 引擎就是IE的WebBrowser控件。现在很多非IE核心的浏览器用的是WebKit引擎，比如遨游3或搜狗的双核、Chrome、Safari。Ge

HTML

- HTML就是描述网页长什么样子、有什么内容的一个文本。查看网页的描述内容（HTML）的方式：使用IE浏览器的话，在网页上点击右键，选择“查看源文件”
- 浏览器兼容性问题：描述文件是一个统一的，但是就像口语翻译一样，不同的翻译翻译出来的东西也是不一样的。不同浏览器品牌对HTML的支持是有差异的，所以同一个网页在IE上和FireFox上看起来可能长得不一样，最明显的就是以前QQ空间上的页面在FireFox上显示就有问题，甚至有的页面在IE6、IE7、IE8上长的也不一样。因此Web开发过程中的一个重要的也是最头疼的问题就是浏览器的兼容。测试FireFox（简称FF）、Chrome等浏览器安装各自的软件就可以，测试不同版本的IE可以用IETester

静态页面、动态页面(*)

- 网站页面分为静态页面和动态页面两种
  - 静态页面：有一个html页面文件保存在服务器上，浏览器要这个页面的时候服务器就把这个页面文件发给浏览器；
  - 动态页面：服务器上没有浏览者要看的页面，而是服务器动态生成的HTML页面发给浏览器，动态语言的服务器端可以用C#、VB.Net、PHP、Java、C等编写。
- 编写普通的HTML页面是和任何后台语言无关的，可以使用Dreamweaver、Expression Web(FrontPage的改头换面版)等工具写，这些工具是给页面美工用的，开发人员用VisualStudio写html就够了。不要把精力放到怎么把界面做好看上，正规公司都有专门的页面美工，不正规公司都是偷别人的美工页面，无论是偷别人的页面，还是使用公司美工开发出来的页面，对于开发人员要做的“填模板”工作都是一样的。

第一个网页

- 新建Web项目（新建→ASP.Net Web应用程序），新建html页面（添加→新建项→Web→HTML页）
- 查看页面的方式：
  - 切换到“设计”视图，可以在这里查看初步的预览效果，不是很准，可以在“设计”视图从工具箱中拖放控件可视化的设计，设计复杂页面的时候很少直接可视化设计。
  - 在编辑器上点右键，选择“在浏览器中查看”。无法进行调试。
  - 将要查看的页面设为起始页（在文件上点击右键“设为起始页”），然后点击“启动调试”。可以调试。
- 学没有JS、C#代码的时候用“在浏览器中查看”。修改页面不用关闭浏览器再打开，刷新就可以。

HTML页结构说明

- 所有内容都在<html></html>标签之内；<head></head>内放的是头部信息，是对页面的描述，不会直接显示在页面中，<head>内的<title>中设置的是页面的标题，<title>只能放在<head>中；<body>是页面的主体，大部分显示内容都定义在这里。
- 所有页面都应该至少包含这些部分，由于浏览器容错性强，所以即使不包含也能正常显示，但是最好还是写完全了。

颜色体系

- body标签的bgcolor属性可以设定网页的背景颜色，<body bgcolor="#006699">
- #006699这就是HTML中表示颜色的方式，每两个是一组，三组分别就表示R、G、B的值，是16进制表示。关于RGB见备注
- 可以使用VS内置的颜色选择对话框生成RGB值，也可以用取色器（比如DebugBar内置的取色器：打开IE，打开DebugBar工具栏，点击吸管图标）；HTML还预定义了一些颜色：red、black、white等，比如bgcolor="black"。
- 配色不是一个专业开发人员考虑的，是美工的事情，所以对于颜色的取值不用太操心，知道有这么一回事就行了。

HTML和XML的联系、区别

- XHTML
- 格式标签： 创建段落；
回车，也可以写成
，在HTML中有一些标签可以不关闭，
就是一个，这是和XML不同的地方（常考），但是为了遵循XHTML规范，推荐像XML一样严格关闭。

- 属性值：HTML中属性值即可以用单引号括起来、也可以用双引号括起来、甚至不用引号都可以（不推荐），单双要配对。
- 注释：HTML使用和XML一样的<!--注释内容-->来做注释。
- 特殊字符：HTML中<、>是有特殊含义的、空格是不会被显示的（输入一个带空格的字符串看看），所以需要特殊符号，相当于C#中的'\n'转义符。<（小于号，less than）；>（大于号，greater than）； （空格，no-break space）。使用工具、免除记忆。为什么特殊字符？见备注。

文字格式

-
只是回车，<p>是分段。<p>前后会有比较大的空白，而
则没有。
- 传智播客 居中显示
- h标签，HTML定义了<h1></h1>到<h6></h6>六个h标签，分别表示不同大小的字体。
- a粗体。
- 字体标签，红色 红色

URL、超链接

- URL：URL表示资源在网络中的地址，比如http://127.0.0.1/a.htm、ftp://192.168.88.128/b.zip。还有URI的概念，比URL大，有的类中使用URI这种说法，可以暂时看成和URL一样就行。
- 超级链接：如鹏网。
- <a>中还可以嵌套图片，这样就是点击图片打开连接

超链接深入

- 相对URL：相对URL表示相对于当前文档的资源，“/”表示网站根目录，“../”表示父目录，“../../”表示父目录的父目录，“./”或者不写任何斜线表示相对于当前路径的目录。站内引用最好用相对URL，这样域名改变了、目录改变了都不受影响。
  -
  -
- 将<a>的target属性设定为"_blank"就可以在新窗口中打开超链接。国情：国内的网站很多都是默认在新窗口中打开。
- 用name属性为<a> 起名字：这里是最后。这样可以通过转到平台来跳转到超链接的部分。
- 案例：去往评论、回到正文。多敲几个回车

图片

- 注意图片是链接的，不是插入的，所以如果Src指向的文件不存在了，就看不了了。alt属性为图片无法显示时的显示文本，鼠标方式去也会有悬浮提示“点击查看大图”；border属性指定边框，border="0"不显示边框；width、height属性指定图片的显示大小，如果不指定则是图片的原始大小。
- 最好指定width、height，哪怕是原始尺寸大小，因为如果不指定大小，图片会不占位置，图片下载后才调整大小，会造成页面很乱。如果指定了width、height哪怕图片没有加载完成，也会先把位置占上。
- 如果网页上要显示小图（比如缩略图），不要仅仅是把大图设定一下width、height来缩小，因为仍然会下载大图，会使得加载速度很慢。
- 易错，不要以为把bmp后缀改为jpg就是改文件格式了！

列表、表格

- 列表：
灌水区
版务区
原创贴图。unordered list。
- （*）还有有序的列表
，用的很少。ordered list
- 表格：<table></table>为表格，在内部通过<tr>创建行，<tr>内部通过<td> 创建单元格。可以将table的border属性设为0来隐藏表格线。
  - <tr>的属性：align，水平对齐，可选值left、right、center；valign，垂直对齐，可选值top、middle、bottom。
  - <td>也有align和valign。<tr align="right"><td>tom</td><td align="left">20</td><td> 男</td></tr>：子标签默认继承父标签的属性，如果自己单独指定了属性，则会覆盖父标签的属性。
  - （*）还可以使用rowspan、colspan进行单元格的合并，用VS可视化的功能来做就行。
  - （*）表头的td可以用th代替，这样就会表头粗体、居中显示。
  - 建议将表头用<thead>代替<tr>

表单

- 网站表单与填表
- <form>标签为表单标签。如果要把数据提交到服务器，则需要将<input>、<textarea>、<select>等表单元素放到form中。
- <input>是主要的表单元素，type的可选值：submit（提交按钮）、button（普通按钮）、checkbox（复选框）、file（文件选择框）、hidden（隐藏字段）、image（图片按钮）、password（密码框）、radio（单选按钮）、reset（重置按钮）、text（文本框）。<input type="file" />

input表单详解

- submit：点击submit按钮表单就会被提交给服务器，中文IE下默认按钮文本为“提交查询”，可以设置value属性修改按钮的显示文本
- text：size属性为宽度，value为值，maxlength为可以输入的最大长度，readonly只读。<input type="text" readonly/>（只写属性名，不写属性值）或者<input type="text" readonly="readonly" />（推荐）
- checkbox：checked属性表示是否被选中，<input type="checkbox" checked />或者<input type="checkbox" checked="checked" />(推荐)checked、readonly等这种只有一个可选值的属性都可以省略属性值。
- radio：相同name属性的为一组，不同radio设定不同的value值，这样通过取指定name的值就可以知道谁被选中了，不用单独的判断。
- file：使用file，则form的enctype必须设置为multipart/form-data、method属性为POST（*）
- image：使用src属性指定图片的地址，用来实现美化的“登录按钮”。

<select>标签

- 用来创建类似于WinForm中的ComboBox或者ListBox
- 如果size属性大于1就是ListBox（size的值为显示出来的列表数量），否则就是ComboBox。<select multiple>或者<select multiple="multiple">（推荐），那么就是可以多选的ListBox。
- select中的项是<option>，<option>北京</option>还可以设定项的值<option value="1">北京</option>。
- 将一个option设置为选中：<option selected>333</option>或者<option selected="selected">333</option>(推荐)就可以将这个项设定为选择项
- 如何实现“不选择”，添加一个<option value="-1">--不选择--<option>，然后编程判断select选中的值如果是-1就认为是不选择。
- select分组选项，可以使用optgroup对数据进行分组，分组本身不会被选择，无论对于下拉列表还是列表框都适用。备注

其他标签

- <textarea>多行文本（也是表单元素）：<textarea>文本</textarea>，cols、rows属性表示行数和列数。
- <label>：在<input type="text">前可以写普通的文本来修饰，但是单击修饰文本的时候input并不会得到焦点，而用label则可以，for属性指定要修饰的控件的id，<label for="txt1" >asdfad</label>
  - 为被修饰的控件设置一个唯一的id。
  - <label for="ma">婚否</label> <input id="ma" type="checkbox" />
- fieldset：GroupBox效果，将控件划分一个区域，看起来更规整
  - <fieldset>
  - <legend>常用</legend>
  - <input type="text" />
  - </fieldset>

练习

- 练习1：实现登录界面，有用户名、密码、验证码（使用普通图片代替）、“记住密码”复选框、登录按钮。使用Table进行布局。使用label来写修饰文本。
- 练习2：实现注册页面，分为两个页面，第一个页面是协议显示页面，点击“我同意”超链接进入第二个注册页面，填写内容：用户名、密码、重复密码、省份（下拉列表）、性别（男、女、保密三个Radio）、职业（学生、公司职员、其他三个Radio）、爱好（登山、篮球、足球、读书、游泳五个CheckBox）。使用label来写修饰文本。将爱好几个CheckBox放到一个GroupBox中
- 自动提示快速完成页面，结束标签<自动补全，Ctrl+J自动提示。

头部标签

- <meta>标签，<meta>有指定name和指定http-equiv两种用法，<meta name="名字" content="值" />、<meta http-equiv="名字" content="值" />两种用法。
  - <meta http-equiv="Content-Type" content="text/html;charset=utf-8" />指定网页编码
  - <meta http-equiv="Refresh" content="3" /> 三秒钟后刷新此网页。
  - <meta http-equiv="Refresh" content="3;url=http://www.rupeng.com" /> 三秒钟后重定向到新网页。发帖成功后提示“发帖成功，即将转向帖子查看页面”。
  - <meta http-equiv="Cache-Control" content="no-cache" /> 禁止浏览器缓存页面。

层（Div）、块（Span）

- 层： 将内容放到层中，就以将这些内容当成一个整体进行处理，比如整体隐藏、整体移动等。div非常强大和常用。类似于WinForm的Panel。
- span：div是将内容放到一个矩形的区块中，会影响布局，而span只是把一段内容定义成一个整体进行操作，但不应该布局、显示。
- 层内文字连续英文不换行的问题，wordbreak
- 溢出处理。overflow：scroll，overflow：hide

样式表、CSS

- CSS（层叠样式表）是用来美化页面用的，可以对页面元素进行更精细的设置，样式主要描述元素的字体颜色、背景颜色、边框等。CSS主要有元素内联、页面嵌入和外部引用三种使用方式。CSS是描述元素的皮肤！
  - 元素内联，直接将样式写入元素的style属性中，<input type="text" readonly="readonly" style="background-color: #FF00FF" />，适用于样式没有可复用性的场合。
  - 页面嵌入：在head中加入
    - <style type="text/css">
    - input{border-color:Yellow;color:Red;}
    - </style>
    - 表示页面中所有input都是采用指定的样式。适合于样式复用，减小页面体积
  - 外部引用，将css内容写入css后缀的文件
    - textarea{background:yellow}
    - 然后在页面中引用，在head中加入
    - <link type="text/css" rel="Stylesheet" href="s1.css" />
    - 适合于多个页面共享css。

- 推荐把尽可能多的样式写到单独的css文件中，这样可以复用，美工人员和开发人员很好的分工。
- 只有页面特有的样式才写到<style>中
- 只有元素特有的样式才写到元素的style属性中。
- 如果不同级别的样式有冲突，详细级别、子元素的会覆盖更高级别、父元素的。

常见样式

- css计量单位：css中表示宽度、距离时有多种计量单位：px（像素）、30%（百分比）、em（相对单位）等。width:20px。
- background-color:Red;背景颜色；color：文本颜色
- border-style:solid;边框风格，实线（默认是没有），还有dotted(点)等值；border-color：边框颜色；border-width：边框宽度(默认是0)。例子：style="border-color:Red;border-width:1px;border-style:dotted;"
- display：元素是否显示，可选值none（不显示）、block （显示为块级元素，此元素前后会带有换行符。）、inline（显示为内联元素，元素前后没有换行符 ）等。
- cursor，鼠标在元素上时显示的光标图标，可选值：cursor（默认光标）、pointer（超链接上的手）、text（输入Bean）、wait（忙沙漏）、help（帮助）等。还可以通过cursor:url(dinosau2.ani)使用ani、cur格式的自定义光标图片。
- LI不显示圆点：LIST-STYLE-TYPE: none;一般设在li或者ul上
- 应用：图片：不显示边框，见备注

样式选择器

- 对于非元素内联的样式需要定义样式选择器，通俗的说就是这个样式适合于哪些元素，三种：标签选择器、class选择器和id选择器。
- 标签选择器 input{border-color:Yellow;color:Red;}，对于指定的标签采用统一的样式
- class选择器，以定义一个命名的样式，然后在用到它的时候设定元素的class属性为样式的名称，还可以同时设定多个class，名称之间加空格
  - 样式名称开头加“.”
  - .warning{background:Yellow;}
  - .highlight{font-size:xx-large;cursor:help;}
  - <table><tr><td class="highlight">aaa</td><td class="warning">bb</td><td class="highlight warning">ccc</td></tr></table>

标签+class选择器

- class选择器也可以针对不同的标签，实现同样的样式名对于不同的标签有不同的样式，只要在样式名前加标签名即可。
  - input.accountno{text-align:right;color:Red;}
  - label.accountno{font-style:italic;}
  - <input class="accountno" type="text" value="111111111111111" />
  - <label class="accountno">3333333333333333</label>

id选择器

- 为指定id的元素设定样式，id前加#
- #username
- {
- font-size:xx-large;
- }
- <input id="username" type="text" value="aaaaaaaaaaaa" />

- style、class可以同时组合使用
- <input id="username" class="accountno" style="font-size:xx-large" type="text" value="aaaaaaaaaaaa" />

更多选择器(*)

- 关联选择器：
  - P strong{ background-color:Yellow}
  - 表示P标签内的strong标签内的内容使用的样式
  - fadsfasdfads
  - adfasfd
- 组合选择器，同时为多个标签设定一个样式
  - H1,H2,input{background-color:Green}
  - <h1>nihao</h1>
  - <input type="text" value="test" />

伪选择器

- 伪选择器：为标签的不同状态设定不同的样式：
- A:visited：超链接点击过的样式；A:active：选中超链接时的样式；A:link：超链接未被访问时的状态；A:hover：鼠标移到超链接时的状态。
- A:visited {TEXT-DECORATION: none}
- A:active {TEXT-DECORATION: none}
- A:link {TEXT-DECORATION: none}
- A:hover {TEXT-DECORATION: underline}

- 说明：TEXT-DECORATION: none表示超链接不显示下划线。

Div+CSS布局

- 网页布局就是“这块内容显示在左边，那两块内容并排显示，那块内容漂浮在页面上”。
- 不要使用<table>进行布局，因为：table可能会在所有tr、td加载完成以后才显示，所以加载完成之前界面是一片空白；用table布局会将布局方式写在html中，违反了“语义性”原则；用table会影响搜索引擎的抓取，不利于SEO。因此Table用来表达真是表格状数据的东西，布局用Div(层)+Css来做,Div用来圈定元素，CSS用来定义元素的位置。
- Div+CSS就是将要布局的内容用 切成块，然后使用css描述每个块的大小、位置等。
- 布局最重要的一个属性就是float，查看此文档的描述。
- 初学不用研究太深，能读懂、会用就可以，备注有很多案例。

传智播客 www.itcast.cn —高级软件人才实作培训专家

JavaScript语言基础

讲师：杨中科

课前说明

- 内容：JavaScript语言。JavaScript语法很多和C#、Java、C等语言类似，因此本课只讲JavaScript特有的语法，不会再讲解编程的基本原理。
- 目标：掌握JavaScript语法，为Dom编程和JQuery打基础。
- 参考书：张孝祥《JavaScript网页开发——体验式学习教程》

什么是JavaScript

- HTML只是描述网页长相的标记语言，没有计算、判断能力，如果所有计算、判断（比如判断文本框是否为空、判断两次密码是否输入一致）都放到服务器端执行的话网页的话页面会非常慢、用起来也很难用，对服务器的压力也很大，因此要求能在浏览器中执行一些简单的运算、判断。JavaScript就是一种在浏览器端执行的语言。
- JavaScript的Java没直接的关系，唯一的关系就是JavaScript原名LiveScript，后来吸收了Java的一些特性，升级为JavaScript。JavaScript有时被简称为JS。
- JavaScript是解释型语言，无需编译就可以随时运行，这样哪怕语法有错误，没有语法错误的部分还是能正确运行。

JS的开发环境

- VS中JavaScript、JQuery的自动完成功能：在VS2010中直接有，VS2008需要安装VisualStudio 2008SP1（http://www.microsoft.com/downloads/details.aspx?displaylang=zh-cn&familyid=27673c47-b3b5-4c67-bd99-84e525b5ce61 ）和VS90SP1-KB958502-x86（http://code.msdn.microsoft.com/KB958502/Release/ProjectReleases.aspx?ReleaseId=1736）补丁会更强更好用。如果实在“.”不出来也没关系，不影响运行。注意：先安装2008SP1，再安装VS90SP1-KB958502-x86。
- JS是非常灵活的动态语言，不像C#等静态语言那样严谨，开发工具中的JS完成功能只是一个辅助、建议，“.”出来的成员调用可能不能用，“.”不出来的成员也许也能调用，因此不要因为“点儿不出来”而担心代码有问题。
- VS2008的HTML编辑器中触发JavaScript自动完成：Ctrl+J。

JS入门

- <script type="text/javascript">
- alert(new Date().toLocaleDateString());
- </script>
- <script language="....>已经不推荐使用。
- JavaScript代码放到<script>标签中，script可以放到<head>、<body>等任意位置，而且可以有不止一个<script>标签。alert函数是弹出消息窗口，new Date()是创建一个Date类的对象，默认值就是当前时间。JS是大小写敏感的。
- 放到<head>中的<script>在body加载之前就已经运行了。写在body中的<script>是随着页面的加载而一个个执行的。
- 除了可以在页面中声明JavaScript以外，还可以将JavaScript写到单独的js文件中，然后在页面中引入：<script src="test.js" type="text/javascript"></script>。声明到单独的js文件的好处是多页面也可以共享、减小网络流量。js文件的CDN(*)
- 注意：不要写成<script src="test.js" type="text/javascript"/>否则会有问题，这是一个比较特殊的地方。

事件

- 在超链接的点击里执行JavaScript：发发
- JavaScript中也有事件的概念，当按钮被点击的时候也可以执行JavaScript：
  - <input type="button" onclick="alert(99)" value="久久"/>
  - 只有超链接的href中的JavaScript中才需要加“"javascript:”，因为它不是事件，而是把“"javascript:”看成像“http:”、“ftp:”、“thunder://”、“ed2k://”、"mailto:"一样的网络协议，交由js解析引擎处理。只有href中这是这是一个特例。

JS的变量

- JavaScript中即可以使用双引号声明字符串，也可以使用单引号声明字符串。主要是为了方便和html集成，避免转义符的麻烦。
- JavaScript中有null、undefined两种，null表示变量的值为空，undefined则表示变量还没有指向任何的对象，未初始化。两者的区别参考资料。
- JavaScript是弱类型，声明变量的时候无法：int i=0；只能通过var i=0;声明变量，和C#中的var不一样，不是C#中那样的类型推断。
- JavaScript中也可以不用var声明变量，直接用，这样的变量是“全局变量”，因此除非确实想用全局变量，否则使用的时候最好加上var。
- JS是动态类型的，因此var i=0;i="abc";是合法的。

JS除错与调试

- 如果JavaScript中的代码有语法错误，浏览器会弹出报错信息，查看报错信息就能帮助排查错误。
- JavaScript的调试，使用VS可以很方便的进行JavaScript的调试，调试时需要注意几点：
  - IE6的调试选项要打开，Internet选项→高级，去掉“禁用脚本调试”前的勾选。
  - 以调试方式运行网页。
  - 设置断点、监视变量等操作和C#一样。
- 案例：用循环语句的方法计算1到100之间整数的和

判断变量初始化

JavaScript中判断变量、参数是否初始化的三种方法：

```
var x;
if (x == null) {
    alert("null");
}
if (typeof (x) == "undefined") {
    alert('undefined');
}
if (!x) {alert('不x');}
if(x){}//变量被初始化了或者变量不为空或者变量不为0.
```

推荐用最后一种方法。

函数的声明

- JavaScript中声明函数的方式：

```
function add(i1, i2) {
        return i1 + i2;
    }
int add(int i1,int i2)//C#写法
```

- 不需要声明返回值类型、参数类型。函数定义以function开头。

```
var r = add(1, 2);
alert(r);
r = add("你好", "tom");
alert(r);
```

- JavaScript中不像C#中那样要求所有路径都有返回值，没有返回值就是undefined。
- 易错：自定义函数名不要和js内置、dom内置方法重名，比如selectall、focus等函数名不要用。

匿名函数

```
var f1 = function(i1, i2) {
        return i1 + i2;
    }
    alert(f1(1,2));
```

- 类似于C#中的匿名函数。
- 这种匿名函数的用法在JQuery中的非常多
- alert(function(i1, i2) { return i1 + i2; }(10,10));//直接声明一个匿名函数，立即使用。用匿名函数省得定义一个用一次就不用的函数，而且免了命名冲突的问题，js中没有命名空间的概念，因此很容易函数名字冲突。通过例子发现一旦命名冲突以最后声明的为准
- 必须<script src="my1.js" type="text/javascript"></script>不能：<script src="my1.js" type="text/javascript"/>

JS面向对象基础（*）

- JavaScript中没有类的语法，是用函数闭包（closure）模拟出来的，下面讲解的时候还是用C#中的类、构造函数的概念，JavaScript中String、Date等“类”都被叫做“对象”，挺怪，方便初学者理解，不严谨。JavaScript中声明类（类不是类，是对象）：

```
function Person(name,age) {
    this.name = name;
    this.age =age;
    this.SayHello=function(){
      alert("你好，我是"+this.name+"，我"+this.age+"岁了");
    }
}
var p1 = new Person("tom",20);
p1.SayHello();
```

- 必须要声明类名，function Person(name,age)可以看做是声明构造函数，Name、Age这些属性也是使用者动态添加了。var p1 = Person("tom", 30);//不要丢了new，否则就变成调用函数了，p1为undefined。new 相当于创建了函数的一个实例

String对象(*)

- length属性；
- charAt方法；
- indexOf；
- match、replace、search方法，正则表达式相关
- split
- substr、substring

Array对象

- JavaScript中的Array对象就是数组，首先是一个动态数组，而且是一个像C#中数组、ArrayList、Hashtable等的超强综合体。

```
var names = new Array();
names[0] = "tom";
names[1] = "jerry";
names[2] = "lily";
for (var i = 0; i < names.length; i++) {
    alert(names[i]);
}
```

- 无需预先制定大小，动态。

练习

- 求一个数组中的最大值。定义成函数。
- 将一个字符串数组输出为|分割的形式，比如“梅西|卡卡|郑大世”。不要使用JavaScript中的Join函数。arr1.join("|")将数组用分隔符连接成一个字符串。
- 将一个字符串数组的元素的顺序进行反转。{"3","a","8","haha"} {"haha","8","a","3"}。不要使用JavaScript中的反转函数。提示：第i个和第length-i-1个进行交换。定义成函数。myreverse
- 交换两个变量，数组同样是传递引用，js出错很麻烦。

JS中的Dictionary

- JS中的Array是一个宝贝，不仅是一个数组，还是一个Dictionary，还是一个Stack。
- var pinyins = new Array();
- pinyins["人"] = "ren";
- pinyins["口"] = "kou";
- pinyins["手"] = "shou";
- alert(pinyins["人"]);
- alert(pinyins.人);
- 像Hashtable、Dictionary那样用，而且像它们一样效率高。
- 课下练习：网页版的火星文翻译。

Array的简化声明

- Array还可以有简化的创建方式
  - var arr = [3, 5, 6, 8, 9]; 普通数组初始化
  - 这种数组可以看做是pinyins["人"] = "ren";的特例，也就是key为0、1、2……
- 字典风格的简化创建方式：
  - var arr = {"tom":30,"jim":20};

数组、for及其他

- 对于数组风格的Array来说，可以使用join方法拼接为字符串

```
var arr = ["tom","jim","lily"];
alert(arr.join(","));//JS中join是array的方法，不像.Net中是string的方法
```

for循环可以像C#中的foreach一样用

- for循环还可以获得一个对象所有的成员，类似于.Net中的反射

```
for (var e in document) {
        alert(e);
    }
```

有了它没有文档也可以进行开发。

扩展方法（*）

- 通过类对象的prototype设置扩展方法，下面为String对象增加quote（两边加字符）方法
- ```
  String.prototype.quote = function(quotestr) {
      if (!quotestr) {
          quotestr = "\"";
      }
      return quotestr + this + quotestr;
  };
  alert("abc".quote());      alert("abc".quote("|"));
  ```
- 扩展方法的声明要在使用扩展方法之前执行。JS的函数没有专门的函数默认值的语法，但是可以不给参数传值，不传值的参数值就是undefined，自己做判断来给默认值。
  ```

传智播客 www.itcast.cn —高级软件人才实作培训专家

DOM编程

讲师：杨中科

课前说明

- 内容：使用JavaScript操作Dom进行DHTML开发。
- 目标：能够使用JavaScript操作Dom实现常见的DHTML效果。
- 参考书：张孝祥《JavaScript网页开发——体验式学习教程》

DOM入门

- DOM就是HTML页面的模型，将每个标签都做为一个对象，JavaScript通过调用DOM中的属性、方法就可以对网页中的文本框、层等元素进行编程控制。比如通过操作文本框的DOM对象，就可以读取文本框中的值、设置文本框中的值。JavaScript→Dom就是C#→.Net Framwork。没有.net，C#只能for、while，连WriteLine、MessageBox都不行。Dom就是一些让JavaScript能操作HTML页面控件的类、函数。
- DOM也像WinForm一样，通过事件、属性、方法进行编程。
- CSS+JavaScript+DOM=DHTML
- 学习阶段只考虑IE。用IE Collection安装IE所有版本，学习使用IE6（要调试必须使用本机安装的版本）。

事件

- 事件：<body onmousedown="alert('哈哈')">当点击鼠标的时候执行onmousedown中的代码。有时间事件响应的代码太多，就放到单独的函数中：

```
<script type="text/javascript">
    function bodymousedown() {
        alert("网页被点坏了，赔吧！ ");
        alert("逗你玩的！ ");
    }
</script>
<body onmousedown="bodymousedown()">
```

bodymousedown后的括号不能丢（onmousedown="bodymousedown" ），因为表示onmousedown事件发生时调用bodymousedown函数，而不是onmousedown事件的响应函数是bodymousedown。

动态设置事件

可以在代码中动态设置事件响应函数，就像.Net中btn.Click+=一样

```
function f1() {
    alert("1");
}
function f2(){
    alert("2");
}
```

```
<input type="button" onclick="document.ondblclick=f1" value="关联事件1" />//注意f1不要加括号。如果加上括号就变成了执行f1函数，并且将函数的返回值复制给document.ondblclick
<input type="button" onclick="document.ondblclick=f2" value="关联事件2" />
```

window对象1

window对象代表当前浏览器窗口，使用window对象的属性、方法的时候可以省略window，比如window.alert('a')可以省略成alert('aa')。

(1)alert方法，弹出消息对话框

(2)confirm方法，显示“确定”、“取消”对话框，如果按了【确定】按钮，就返回true，否则就false

```
if (confirm("是否继续？")) {
    alert("确定");
}
else {
    alert("取消");
}
```

window对象2

- (3)重新导航到指定的地址：navigate("http://www.rupeng.com");
- (4)setInterval每隔一段时间执行指定的代码，第一个参数为代码的字符串，第二个参数为间隔时间（单位毫秒），返回值为定时器的标识

```
setInterval("alert('hello')", 5000);
```

- (5)clearInterval取消setInterval的定时执行，相当于Timer中的Enabled=False。因为setInterval可以设定多个定时，所以clearInterval要指定清除那个定时器的标识，即setInterval的返回值。

```
var intervalId = setInterval("alert('hello')", 5000);
clearInterval(intervalId);
```

window对象3

- (6)setTimeout也是定时执行，但是不像setInterval那样是重复的定时执行，只执行一次，clearTimeout也是清除定时。很好区分：Interval：间隔；timeout：超时。

  ```
  var timeoutId = setTimeout("alert('hello')", 2000);
  ```

- 案例：实现标题栏走马灯的效果，也就是浏览器的标题文字每隔500ms向右滚动一下。提示：标题为document.title属性。实现代码参考备注。
- 练习：刚进入的时候还是向左滚动，点击【向左】按钮就向左连续滚动，点击【向右】按钮就向右连续滚动。
  - 思路1、全局变量，标志当前的滚动方向，当点击向左的时候dir="left",向右dir="right"。
  - 思路2、scrollleft scroolright,向右滚的时候将scrollleft的Interval clear掉，然后setInterval启动scrollright

window对象4

- (7)showModalDialog弹出模态对话框，注意showModalDialog必须在onClick等用户手动触发的事件中才会执行，否则可能会被最新版本的浏览器当成广告弹窗而拦截。
  - 第一个参数为弹出模态窗口的页面地址。
  - 在弹出的页面中调用window.close()（不能省略window.close()中的window.）关闭窗口，只有在对话框中调用window.close()才会自动关闭窗口，否则浏览器会提示用户进行确认。
- (8)showModelessDialog弹出非模态窗口，参数等和showModalDialog一样。
- 如何在弹出窗口和主窗口之间进行参数传递、如何控制弹出窗口的大小等后面再讲。

body、document对象的事件

- （1）onload：网页加载完毕时触发，浏览器是一边下载文档、一边解析执行，可能会出现JavaScript执行时需要操作某个元素，这个元素还没有加载，如果这样就要把操作的代码放到body的onload事件中，或者可以把JavaScript放到元素之后。元素的onload事件是元素自己加载完毕时触发，body onload才是全部加载完成
- （2）onunload：网页关闭（或者离开）后触发。
- （3）onbeforeunload：在网页准备关闭（或者离开）后触发。在事件中为"window.event.returnValue赋值（要显示的警告消息），这样窗口离开（比如前进、后退、关闭）就会弹出确认消息
  - <body onbeforeunload="window.event.returnValue=' 真的要放弃发帖退出吗？'">。显示的文字随浏览器版本而有差异。

其他事件

- 除了有特有的属性之外，当然还有通用的HTML元素的事件：onclick（单击）、ondblclick（双击）、onkeydown（按键按下）、onkeypress（点击按键）、onkeyup（按键释放）、onmousedown（鼠标按下）、onmousemove（鼠标移动）、onmouseout（鼠标离开元素范围）、onmouseover（鼠标移动到元素范围）、onmouseup（鼠标按键释放）等。

window对象的属性1

- window.location.href='http://www.itcast.cn'，重新导向新的地址，和navigate方法效果一样。window.location.reload() 刷新页面
- window.event是非常重要的属性，用来获得发生事件时的信息，事件不局限于window对象的事件，所有元素的事件都可以通过event属性取到相关信息。类似于winForm中的e(EventArg).
  - altKey属性，bool类型，表示发生事件时alt键是否被按下，类似的还有ctrlKey、shiftKey属性，例子 <input type="button" value="点击" onclick="if(event.altKey){alert('Alt点击')}else{alert('普通点击')}" /> ；
  - clientX、clientY 发生事件时鼠标在客户区的坐标；screenX、screenY 发生事件时鼠标在屏幕上的坐标； offsetX、offsetY 发生事件时鼠标相对于事件源（比如点击按钮时触发onclick）的坐标。
  - returnValue属性，如果将returnValue设置为false，就会取消默认事件的处理。在超链接的onclick里面禁止访问href的页面。在表单校验的时候禁止提交表单到服务器，防止错误数据提交给服务器、防止页面刷新。
  - srcElement，获得事件源对象。几个事件共享一个事件响应函数用。
  - keyCode，发生事件时的按键值。
  - button，发生事件时鼠标按键，1为左键，2为右键，3为左右键同时按。<body onmousedown="if(event.button==2){alert('禁止复制');}">

window对象的属性2

- （*）screen对象，屏幕的信息

```
alert("分辨率：" + screen.width + "*" + screen.height);
        if (screen.width < 1024 || screen.height < 768) {
            alert("分辨率太低！");
        }
```

- clipboardData对象，对粘贴板的操作。clearData("Text")清空粘贴板；getData("Text")读取粘贴板的值，返回值为粘贴板中的内容；setData("Text",val)，设置粘贴板中的值。
  - 案例：复制地址给友好。见备注。
  - 当复制的时候body的oncopy方法被触发，直接return false就是禁止复制。<body oncopy="alert('禁止复制！');return false;"
  - 很多元素也有oncopy、onpaste事件：
  - 案例：禁止粘贴帐号。见备注。

window对象的属性3

在网站中复制文章的时候，为了防止那些拷贝党不添加文章来源，自动在复制的内容后添加版权声明。

```
function modifyClipboard() {
        clipboardData.setData('Text', clipboardData.getData('Text')
    + '本文来自传智播客技术专区，转载请注明来源。' +
    location.href);
      }
```

oncopy="setTimeout('modifyClipboard()',100)"。用户复制动作发生0.1秒以后再去改粘贴板中的内容。100ms只是一个经常取值，写1000、10、50、200……都行。不能直接在oncopy里修改粘贴板。

不能直接在oncopy中执行对粘贴板的操作，因此设定定时器，0.1秒以后执行，这样就不再oncopy的执行调用栈上了。

window对象的属性4

- history操作历史记录
  - window.history.back()后退；window.history.forward()前进。也可以用window.history.go(-1)、window.history.go(1)前进
- document属性。是最复杂的属性之一。后面讲解详细使用。

document属性1

- document是window对象的一个属性，因为使用window对象成员的时候可以省略window.，所以一般直接写document
- document的方法：
  - （1）write：向文档中写入内容。writeln，和write差不多，只不过最后添加一个回车
  - <input type="button" value="点击" onclick="document.write('你好')" />
  - 在onclick等事件中写的代码会冲掉页面中的内容，只有在页面加载过程中write才会与原有内容融合在一起
  - <script type="text/javascript">
  - document.write('你好');
  - </script>
  - write经常在广告代码、整合资源代码中被使用。见备注

内容联盟、广告代码、cnzz，不需要被主页面的站长去维护内容，只要被嵌入的js内容提供商修改内容，显示的内容就变了。

document方法

- getElementById方法（非常常用），根据元素的Id获得对象，网页中id不能重复。也可以直接通过元素的id来引用元素，但是有有效范围、form1.textbox1之类的问题，因此不建议直接通过id操作元素，而是通过getElementById
- （*）getElementsByName，根据元素的name获得对象，由于页面中元素的name可以重复，比如多个RadioButton的name一样，因此getElementsByName返回值是对象数组。
- （*）getElementsByTagName，获得指定标签名称的元素数组，比如getElementsByTagName("p")可以获得所有的<p>标签。
- 案例：点击一个按钮，被点击的按钮显示“呜呜”，其他按钮显示“哈哈”。
- 案例：十秒钟后协议文本框下的注册按钮才能点击，时钟倒数。(btn.disabled = true)
- 练习：加法计算器。两个文本框中输入数字，点击【=】按钮将相加的结果放到第三个文本框中。
- 练习：美女时钟。

DOM的动态创建

- document.write只能在页面加载过程中才能动态创建。
- 可以调用document的createElement方法来创建具有指定标签的DOM对象，然后通过调用某个元素的appendChild方法将新创建元素添加到相应的元素下

```
function showit() {
    var divMain = document.getElementById("divMain");
    var btn = document.createElement("input");
    btn.type = "button";
    btn.value = "我是动态的！ ";
    divMain.appendChild(btn);
}
<div id="divMain"></div>
<input type="button" value="ok" onclick="showit()" />
```

innerText、innerHTML

- 几乎所有DOM元素都有innerText、innerHTML属性（注意大小写），分别是元素标签内内容的文本表示形式和HTML源代码，这两个属性是可读可写的。
  - 传智播客
  - <input type="button" value="inner*" onclick="alert(document.getElementById('link1').innerText);alert(document.getElementById('link1').innerHTML);" />
- 用innerHTML也可以替代createElement，属于简单、粗放型、后果自负的创建
  - function createlink() {
  - var divMain = document.getElementById("divMain");
  - divMain.innerHTML = "如鹏网";
  - }
  - 的innerHTML和的innerHTML不一样。

练习

- 练习：点击按钮增加一个网站的超链接
- 练习：点击按钮动态增加网站列表，分两列，第一列为网站的名字，第二列为带网站超链接的网站名。增加三行常见网站。浏览器兼容性问题，见备注。
- 动态产生的元素，查看源代码是看不到的。通过DebugBar→Dom→文档→HTML可以看到。
- 练习：无刷新评论。
- <table>
- <tr><td>猫猫：</td><td>沙发耶！</td></tr>
- </table>
- 昵称：<input type="text" />
- <textarea></textarea>
- <input type="button" value="评论" />

浏览器兼容性问题

- 浏览器兼容性的例子：ie6，ie7对table.appendChild("tr")的支持和IE8不一样，用insertRow、insertCell来代替或者为表格添加tbody，然后向tbody中添加tr。FF不支持InnerText。
- 所以动态加载网站列表的程序修改为：
- var tr = tableLinks.insertRow(-1);//FF必须加-1这个参数
- var td1 = tr.insertCell(-1);
- td1.innerText = key;
- var td2 = tr.insertCell(-1);
- td2.innerHTML = "" + value + "";
- 或者：<table id="tableLinks">
- <tbody></tbody>
- </table>，然后tableLinks. tBodies[0].appendChild(tr);

事件冒泡

- 事件冒泡：如果元素A嵌套在元素B中，那么A被点击不仅A的onclick事件会被触发，B的onclick也会被触发。触发的顺序是“由内而外”。验证：在页面上添加一个table、table里有tr、tr里有td，td里放一个p，在p、td、tr、table中添加onclick事件响应，见备注。
- 练习：KevinButton

其他

- 事件中的this。除了可以使用event.srcElement在事件响应函数中，this表示发生事件的控件。只有在事件响应函数才能使用this获得发生事件的控件，在事件响应函数调用的函数中不能使用，如果要使用则要将this传递给函数或者使用event.srcElement。(*)this和event.srcElement的语义是不一样的，this就是表示当前监听事件的这个对象，event.srcElement是引发事件的对象：事件冒泡。
- 易错：修改元素的样式不是设置class属性，而是className属性。案例：网页开关灯的效果。
- 修改元素的样式不能this.style="background-color:Red"。
- 易错：单独修改样式的属性使用“style.属性名”。注意在css中属性名在JavaScript中操作的时候属性名可能不一样，主要集中在那些属性名中含有-的属性，因为JavaScript中-是不能做属性、类名的。所以CSS中背景颜色是background-color，而JavaScript则是style.background；元素样式名是class，在JavaScript中是className属性；font-size→style.fontSize；margin-top→style.marginTop
- 单独修改控件的样式<input type="button" value="AAA" onclick="this.style.color='red'" />。技巧，没有文档的情况下的值属性名，随便给一个元素设定id，然后在js中就能id.style.出来能用的属性。

案例练习

- 案例1：创建三个输入文本框，当光标离开文本框的时候如果文本框为空，则将文本框背景色设置为红色，如果不为空则为白色。提示：焦点进入控件的事件是onfocus，焦点离开控件的事件是onblur。
- 案例2：评分控件V1，用一个单行5列的Table做评分控件，监听td的click事件，点击一个td的时候，将这个td及之前的td背景变为红色，之后的td背景变为白色。鼠标在评分控件上的时候显示超链接形式的鼠标图标。演示JQuery版。
- 练习1：超链接的单选效果。页面上若干个超链接，点击一个超链接的时候被点击的超链接变为红色背景，其他超链接背景还原为白色。参考：点击变“呜呜”，没有点击变“哈哈”。window.event.returnValue=false;。难点“this”
- 练习2：点击按钮，表格隔行变色：偶数行为黄色背景，奇数行为默认颜色。通过table的getElementsByTagName取得所有的tr，依次遍历，如果是偶数就……。为什么？防止看串了行。
- 练习3：放若干文本框，获得焦点的文本框黄色背景，其他控件背景颜色是白色
  - 思路1：监听所有input的onfocus事件→将背景设置为黄色，监听所有input的onblur事件→将背景设置为白色。思路2：只监听onfocus和练习1一样。
- 练习4：点击表格行，被点击的行高亮显示（背景是黄色），其他行白色背景。监听每个tr的onclick事件，将点击的背景设置为黄色，其他的设置为白色背景。

控制层的显示

- 修改style.display，例子：切换层的显示

```
function toggledivO {
    var div1 = document.getElementById('div1');
    if (div1.style.display == '') {
        div1.style.display = 'none';//不显示
    }
    else {
        div1.style.display = '';//显示
    }
}
```

IE中body的事件范围

- IE中如果在body上添加onclick、onmousemove等事件响应，那么如果页面没有满，则“body 中最后一个元素以下（横向不限制）”的部分是无法响应事件的，必须使用代码在document上监听那些事件，比如document.onmousemove = MovePic
- FF中也差不多。

元素的位置、大小单位

- 通过dom读取元素的top、left、width、height等取到的值不是数字，而是“10px”这样的字符串；为这些属性设值的时候IE可以是80、90这样的数字，FF必须是“80px”、“90%”等这样的字符串形式，为了兼容统一用字符串形式。
- 易错：不要写成div1.style.width=80px，而是div1.style.width='80px'
- 如果要修改元素的大小（宽度加10），则首先要取出元素的宽度，然后用parseInt将宽度转换为数字（parseInt可以将"20px"这样数字开头的包含其他内容的字符串解析为20，parseInt('22px',10)，也就是解析尽可能多的部分）；然后加上一个值，再加上px赋值回去。

层的操作

- 元素的position 样式值：static（无定位，显示在默认位置）、absolute（绝对定位）、fixed（相对于窗口的固定定位，位置不会随着浏览器的滚动而变化，IE6不支持）、relative（相对元素默认位置的定位）。如果要通过代码修改元素的坐标则一般使用absolute，然后修改元素的top（上边缘距离）、left（左边缘距离）两个样式值。left、top都是指的层的左上角的坐标
- 案例：跟着鼠标飞的图片。提示：鼠标移动的事件是onmousemove（一边移动事件一边触发，而不是移动开始或者移动完成才触发），通过window.event的clientX、clientY属性获得鼠标的位置。
- 案例：鼠标放到一个超链接的时候，在鼠标的位置显示一个黄色背景，带图片的悬浮提示，鼠标离开就消失。提示：鼠标进入控件的事件是onmouseover，离开的事件是onmouseout。
- 案例：点击按钮层动态变大。提示：英文字母连续单词不会在中间自动换行的陷阱

问题

- 易错：不要写成div1.style.width=80px，而是div1.style.width='80px'
- 修改元素的样式不能this.style="background-color:Red"，哪怕可以的话也是把以前所有样式都冲掉了。单独修改控件的样式this.style. background='red'，只修改要修改的样式。技巧，没有文档的情况下的值属性名，随便给一个元素设定id，然后在js中就能id.style.出来能用的属性。
- createElement的两种用法，注意innerText的问题
  - var input = document.createElement("<input type='button' value='hello'/>")快速创建元素，并且赋值，但是注意设置的inner部分不会被设置var link = document.createElement("百度")
- label.setAttribute("for", "username"); //设定一些Dom元素属性名特殊的属性,label.for = "username"会有问题。label.setAttribute("xuehao","33333")

案例练习

- 练习：点击【登录】按钮，弹出一个显示用户名、密码等的层。将用户名、密码等写到一个层中，层默认是隐藏的，点击【登录】超链接以后将层显示出来，如果点击层中的关闭按钮，则隐藏层。绝对定位，显示到中间位置。
- 练习：一幅图片。点击小图，弹出一个层在点击的位置显示小图对应的大图，并且显示姓名、身高等信息，点击层中的关闭按钮关闭层。进阶：元素的额外属性。动画效果的显示出来。两种：静态；动态载入数据。
- 评分控件V2。用一个单行5列的Table，td中默认都是starEmpty.jpg这个图片。监听td的mouseover事件，鼠标在一个td的时候将这个td及之前的td的内容换成starFill.jpg这个图片。鼠标在评分控件上的时候显示超链接形式的鼠标图标。案例：注册页面，点击“高级”CheckBox，则显示高级选项，否则隐藏

案例练习

- 练习：界面上有几个球队名字的列表，将鼠标放到球队名字上就变为红色背景，其他球队背景颜色为白色，点击一个球队的时候就将点击的球队变为fontSize=30字体。
- 练习：显示数字时钟，时间显示到一个div中。思路：。
- 练习：有一个搜索文本框，焦点不在文本框中的时候，如果文本框没有值，则文本框中显示灰色文本（Gray）的“输入搜索关键词”，否则显示用户输入的值；焦点在文本框中时如果之前显示“输入搜索关键词”则清空文本框的值，并且将文本修改为黑色。onfocus的时候如果文本框中的值为“输入搜索关键词”，则清空文本框，并且恢复文本框的颜色为Black；onblur的时候如果文本框中没有值，则将文本框的值设置为“输入搜索关键词”并且文本框中显示灰色文本（Gray）style.color='Gray'。（五分钟）

form对象

- document.getElementById('btn1').click()
- form对象是表单的Dom对象。
- 方法：submit()提交表单，但是不会触发onsubmit事件。
- 实现autopost，也就是焦点离开控件以后页面立即提交，而不是只有提交submit按钮以后才提交，当光标离开的时候触发onblur事件，在onblur中调用form的submit方法。代码见备注。
- 在点击submit后form的onsubmit事件被触发，在onsubmit中可以进行数据校验，数据数据有问题，返回false即可取消提交
- <form name="form1" action="a.aspx" method="get" onsubmit="if(document.getElementById('txtname').value.length <=0){alert('姓名必填');return false;}">

不同浏览器的差异（*）

- 面试题：说说开发项目的时候不同浏览器的不同点，你是怎么解决的？appendChild,insertCell,px
- 不同浏览器中对DOM支持的方法不一样
  - 获取网页中那个元素触发了事件：在IE里使用srcElement ；在FireFox里使用target
  - 使用Dom获取和更改网页标签元素内文本：在IE里使用innerText ；在FireFox里使用textContent
  - 动态为网页或元素绑定事件：在IE中绑定事件的方法是attachEvent ；在FireFox中绑定事件的方法是addEventListener
  - 更多http://www.360doc.com/content/09/0319/12/16915_2855107.shtml
- 不同浏览器中对CSS的支持不一样，所以出现在IE中显示正常的网页，在FF下全部乱掉了。哀悼网页使用的CSS只有IE支持，FF都不支持。
- JQuery之类的框架进行了封装，将不同浏览器的差异帮开发人员处理了，开发人员只要调用JQuery的方法，JQuery会帮助在不同浏览器中进行翻译。用JQuery就可以解决不同浏览器上Dom的不同。对于CSS的不同是美工的事，IETester、FF、Chrome。

弹出对话框的处理

- 复习，使用window.showModalDialog('dialog.htm')弹出模态对话框
- 给对话框传递参数，使用showModalDialog的第二个参数传递参数，在对话框中用window.dialogArguments获得传递的参数值；对话框中给window.parent.returnValue设定返回值，这样在父窗口中就可以通过showModalDialog返回值读取设置的返回值了。例子：弹出对话框询问用户姓名，向用户问好；弹出含有“是”、“否”、“取消”三个按钮的模态窗口，点击按钮的时候窗口关闭，然后主窗口显示用户点击的按钮。
- 传递多个参数，将参数包装到数组中，然后仍然是通过第二个参数传递，返回多个返回值也可以返回数组：var arr = new Array();arr[0]=30;arr[1]="tom";
- 练习（面试题），弹出一个含有确定、取消、重试三个按钮的对话框，并且得知用户的选择。

JS中的正则表达式

- 复习C#正则表达式
- JavaScript中创建正则表达式类的方法：
  - var regex = new RegExp("\\d{5}") 或者 var regex = /\d{5}/
  - /表达式/是JavaScript中专门为简化正则表达式编写而提供的语法，写在//中的正则表达式就不用管转义符了。
- RegExp对象的方法：
  - （1）test(str)判断字符串str是否匹配正则表达式，相当于IsMatch
  - var regex = /.+@.+/;
  - alert(regex.test("a@b.com"));
  - alert(regex.test("ab.com"));
  - （2）exec(str)进行搜索匹配，返回值为匹配结果(*)
  - （3）compile编译表达式，提高运行速度。 (*)

string的正则表达式方法

- String对象中提供了一些与正则表达式相关的方法，相当于对于RegExp类的包装，简化调用：

  match(regexp)，相当于调用exec

```
var s = "aaa@163.com";
var regex = /(.+)@(.+)/;
var match = s.match(regex);
alert(RegExp.$1 + "，服务器：" + RegExp.$2);
```

  练习：光标离开Email地址框的时候用正则表达式校验是否是合法的Email地址，如果不是的话Email地址框变红，并且注册按钮禁用，否则Email地址框颜色为白色，启用注册按钮。

HTML、JS的压缩

- HTML、JavaScript的压缩和混淆。去掉空格、缩短变量名，让js、html尺寸更小，提高下载速度。
- HTML、JS压缩、混淆有动态和静态两种方案。HTML压缩器，比如HTML Compress，JavaScript压缩工具：Google Closure Compiler、YUI Compressor 等。
- 很多js库都提供了.min.js、compress.js的压缩版本。

案例

- 案例1：回车实现Tab跳转。响应文本框的onKeyDown事件，window.event.keyCode获得用户点击的keyCode。（*）keyCode和ASCII不是完全一致，主键盘的1和小键盘的1的ASCII一样，但是keyCode不一样。详见备注。回车的keyCode为13，Tab的keyCode为9。if(window.event.keyCode == 13){window.event.keyCode = 9;}
- <body onkeydown="if(window.event.keyCode==13){window.event.keyCode=9;}">
- 只有少数的键才能被替换，大部分是不行的，有权限问题。

案例：金额文本框

- 财务相关系统中涉及到金额的文本框有如下要求：
  - 进入金额文本文本框不使用中文输入法
  - 不能输入非数字
  - 焦点在文本框中时文本框左对齐；焦点离开文本框时文本框右对齐，显示千分位
- 禁用输入法：style="ime-mode:disabled"
- 禁止键入非法值，只有这些才能被键入(k == 9) || (k == 13) || (k==46)||(k==8)||(k==189)||(k==109)||(k==190)||(k==110)|| (k>=48 && k<=57)||(k>=96 && k<=105)||(k>=37 && k<=40)。 onkeydown="return numonKeyDown()" 不要写成onkeydown="numonKeyDown()" 区分事件响应函数和事件响应函数调用的函数。
- 禁止粘贴(伟大的Tester)，<input onpaste="return false;" ，太暴力，应该只是禁止粘贴非法值。在onpaste中通过clipboardData.getData('Text')取到粘贴板中的值，然后遍历每个字符，看是否是合法的值，如果全部是合法值才允许粘贴，只要有一个非法值就禁止粘贴。charAt、charCodeAt
- 添加千分位的方法，见备注
- 焦点在的时候左对齐没有千分位，焦点不在时右对齐千分位。this.style.textAlign='right'

练习1

- 案例代码阅读，模拟对话框。见备注(*)先创建一个满浏览器的层，设定透明度，有遮挡的效果，然后再创建一个层（ZIndex>遮挡层的ZIndex）显示对话框内容。
- 案例：实现省市选择界面。请选择省的处理，从后向前删。
- 练习：歌曲列表 （CheckBox+Label）全选、全不选、反选，只针对一个层中，div.getElementsByTagName("input")，再判断type='checkbox'的项，checked="checked"。
- if(cb.checked=="checked"){//用调试，期望的和实际的。
- 练习：权限选择页面，选择、撤回、全部选择、全部撤回。代码参考“实现省市选择界面”，因为可能多选，判断选择项和单选的会有不同。不用写四个方法，两个方法就够了。
- 善用调试，遇到问题多调试！

传智播客 www.itcast.cn —高级软件人才实作培训专家

JQuery编程

讲师：杨中科

课前说明

- 内容：掌握JQuery编程思想，使用JQuery进行常见网页效果开发。
- 目标：能够使用JQuery开发常见网页效果。
- 参考书：《锋利的JQuery》

JQuery简介

- 普通JavaScript的缺点：每种控件的操作方式不统一，不同浏览器下有区别，要编写跨浏览器的程序非常麻烦。因此出现了很多对JavaScript的封装库，比如Prototype、Dojo、ExtJS、JQuery等，这些库对JavaScript进行了封装，简化了开发。这些库是对JavaScript的封装，也就是咱们调用JQuery的一句函数，JQuery内部这句函数帮我们调用JavaScript中的代码几十句，因为JQuery就是JavaScript语法写的一些函数类，内部仍然是调用JavaScript实现的，所以并不是代替JavaScript的。使用JQuery的代码、编写JQuery的扩展插件等仍然需要JavaScript的技术，Jquery本身就是一堆JavaScript函数。
- Jquery是最火的JavaScript库，已经被集成到VS2010了，得到了MS的支持，MS的Ajax toolkit和JQuery结合也是最方便，JQuery的扩展插件也是非常多。

Jquery简介

- JQuery能做什么。
- JQuery的优点：尺寸小、使用简单方便（Write Less, Do More，吃得少干得多。链式编程（$("#div1").draggble().show().hide().fly()）、隐式迭代（自动对于多个元素进行迭代方法调用））、屏蔽浏览器差异跨浏览器兼容性好（IE 6.0+, FF 2+, Safari 3.0+, Opera 9.0+, Chrome）、插件丰富、开源、免费。
- VS中JavaScript、JQuery的自动完成功能：在VS2010中直接有，VS008需要安装VisualStudio 和VS90SP1-KB958502-x86补丁会更强更好用, 下载地址见备注。然后引用jquery-1.4.1.js，jquery-1.4.1-vsdoc.js放到同目录下，不需要在页面引用。
- vsdoc是vs2008sp1以后增加的一个技术，将js文件对应的vsdoc（相当于js库提供的方法的说明库）放到和js一起，就有会这个第三方js的自动提示的功能

简单的JQuery

- $(document).ready(function() {
- alert("加载完毕！");
- });//注册事件的函数，和普通的dom不一样，不需要在元素上标记on**这样的事件。
- 当页面Dom元素加载完毕时执行代码，可以简写为：
- $(function() {
- alert("加载完毕！");
- });
- 和onload类似，但是onload只能注册一次(window.onload=function...)（没有C#中的+=机制），后注册的取代先注册的，而ready则可以多次注册都会被执行。
- JQuery的ready和Dom 的onload的区别：onload是所有Dom元素创建完毕、图片、Css等都加载完毕后才被触发，而ready则是Dom元素创建完毕后就被触发，这样可以提高网页的响应速度。在jQuery中也可以用$(window).load()来实现onload那种事件调用的时机。

JQuery提供的函数

- $.map(array,fn)对数组array中每个元素调用fn函数逐个进行处理，fn函数将处理返回，最后得到一个新数组

例子，得到一个元素值是原数组值二倍的新数组

```
var arr = [3, 5, 9];
var arr2 = $.map(arr, function(item) { return item * 2; });//联想C#委托的例子。函数式编程。
```

$.map不能处理Dictionary风格的数组。

- $.each(array,fn)对数组arry每个元素调用fn函数进行处理，没有返回值

```
var arr = { "tom": "汤姆", "jerry": "杰瑞", "lily": "莉莉" };
$.each(arr, function(key, value) { alert(key+"="+value); });
```

如果是普通风格的数组，则key的值是序号。

- 还可以省略function的参数，这时候用this可以得到遍历的当前元素：

```
var arr = [3, 6, 9];
$.each(arr, function() { alert(this); });//能读懂。
```

普通数组推荐用无参，用dict风格的就用key、value。

jQuery对象、Dom对象

- jQuery对象就是通过jQuery包装Dom对象后产生的对象：alert($('#div1').html())。Dom对象要想通过jQuery进行操作，先要转换为JQuery对象。
- $('#div1').html()等价于：document.getElementById("div1").innerHTML;
- $('#div1')得到的就是jQuery对象，jQuery对象只能调用jQuery对象封装的方法，不能调用Dom对象的方法，Dom对象也不能调用jQuery对象的方法，所以alert($('#div1').innerHTML是错的，因为innerHTML是DOM对象的属性。
- Array是JS语言本身的对象，不是Dom对象，因此不需要转换为Jquery对象才能用
- （*）将Dom对象转换为JQuery对象的方法，$(dom对象)；当调用jQuery没有封装的方法的时候必须用Dom对象，转换方法：var domobj = jqobj[0]或者var domobj=jqobj.get(0)
- jQuery修改样式：$("#div1").css("background", "red");获得样式$("#div1").css("background")；修改value：$("#un").val("abc")，获得value：$("#un").val()，类似的获得、设置innerText、innerHTML用text()和html()。val、html、text等是方法，不是属性，jQuery中很少有属性的用法，因为属性写法很难“链式编程”。

JQuery选择器

- JQuery选择器用于查找满足条件的元素，比如可以用$("#控件Id")来根据控件id获得控件的jQuery对象，相当于getElementById：
  - $("#div1").html("hello")
- $("TagName")来获取所有指定标签名的jQuery对象，相当于getElementsByTagName：
  - $(function() {
  - $("#btnClick").click(function() {
  - $("p").html("我们都是P");
  - });
  - });

JQuery选择器2

- CSS选择器，同时选择拥有样式的多个元素：

```
<style type="text/css">
    .test{ background-color:Red}
</style>
<script type="text/javascript">
    $(function() {
    $(".test").click(function() {
            alert($(this).text());
        });
    });
</script>
<p class="test">test1</p>
<p class="test">test2</p>
<p class="test">test3</p>
```

JQuery选择器3

- 多条件选择器：$("p,div,span.menuitem")，同时选择p标签、div标签和拥有menuitem样式的span标签元素
- 注意选择器表达式中的空格不能多不能少。易错！
- 层次选择器：
  - （1）$("div li")获取div下的所有li元素（后代，子、子的子……）
  - （2）$("div > li")获取div下的直接li子元素
  - （3）$(".menuitem + div")获取样式名为menuitem之后的第一个div元素（不常用）
  - （4）$(".menuitem ~ div")获取样式名为menuitem之后所有的div元素（不常用）

JQuery的迭代

- 如何判断对象是否存在，jQuery选择器返回的是一个对象数组，调用text()、html()、click()之类方法的时候其实是对数组中每个元素迭代调用每个方法，因此即使通过id选择的元素不存在也不会报错，如果需要判断指定的id是否存在，应该写：
- if ($("#btn1").length <= 0) {
- alert("id为btn1的元素不存在！");
- }

节点遍历

- next()方法用于获取节点之后的挨着的第一个同辈元素，$(".menuitem").next("div")、nextAll()方法用于获取节点之后的所有同辈元素，$(".menuitem").nextAll("div")
- prev、prevAll之前的元素。
- siblings()方法用于获取所有同辈元素，$(".menuitem").siblings("li")。siblings、next等所有能传递选择器的地方能够使用的语法都和$()语法一样。
- 案例：选中的p变色 $(this).css();$(this).siblings().css()
- 案例：评分控件。prevAll,this,nextAll

链式编程

- 高亮选中项：给所有有menuitem这个样式的元素添加click监听事件，当click的时候，向被点击的元素添加highlight这个样式，然后从其兄弟节点（siblings）中移除highlight风格。“.”的时候是针对的上一步的返回值的节点集合的操作。

```
<style type="text/css">
    .menuitem{background-color:Yellow; }
    .highlight { background-color: Red;}
</style>
$(function() {
        $(".menuitem").click(function() {
            $(this).addClass("highlight").siblings().removeClass("highlight");
        });
    });
<p class="menuitem">111111</p><br />
<p class="menuitem">111111</p><br />
<p class="menuitem">111111</p><br />
```

基本过滤选择器

- :first 选取第一个元素。$("div:first")选取第一个
- :last 选取最后一个元素。$("div:last")选取最后一个
- :not(选择器) 选取不满足“选择器”条件的元素，$("input:not(.myClass)")选取样式名不是myClass的<input>
- :even、:odd，选取索引是奇数、偶数的元素：$("input:even")选取索引是奇数的<input>
- :eq(索引序号)、:gt(索引序号)、:lt(索引序号) 选取索引等于、大于、小于索引序号的元素，比如$("input:lt(5)")选取索引小于5的<input>
- $(":header")选取所有的h1……h6元素（*）
- $("div:animated")选取正在执行动画的 元素。 （*）

案例

- 第一行是表头，所以显示大字体（fontSize=30），最后一行是汇总，所以显示红色字体。正文的前三行是前三名，所以显示傻大的字体（28）表格的奇数行是黄色背景。
- 用Dom实现；用JQuery实现。对比差异！

案例

- 案例：表格隔行变色
- 案例：前三名粗体显示
- 不仅可以使用选择器进行进行绝对定位，还可以进行相对定位，只要在$()指定第二个参数，第二个参数为相对的元素. $("ul", $(this)).css("background", "red");
- 案例：修改点击行的所有td的背景色
- 练习：图片版评分控件
- 练习：单选超链接

过滤器

- 属性过滤选择器：
  - $("div[id]")选取有id属性的
  - $("div[title=test]")选取title属性为“test”的<div>，JQuery中没有对getElementsByName进行封装，用$("input[name=abc]")
  - $("div[title!=test]")选取title属性不为“test”的
  - 还可以选择开头、结束、包含等，条件还可以复合。（*）
- 表单对象选择器（过滤器）：
  - $("#form1:enabled")选取id为form1的表单内所有启用的元素
  - $("#form1:disabled")选取id为form1的表单内所有禁用的元素
  - $("input:checked")选取所有选中的元素（Radio、CheckBox）
  - $("select:selected")选取所有选中的选项元素（下拉列表）

元素的each

- jQuery元素的也可以调用each方法，只是对$.each的简化调用。

显示选中的复选框信息

```
$(function() {
    $("input[name=names]").click(function() {
        var names = $("input[name=names]:checked");
        var arr = new Array();
        names.each(function(key, value) { arr[key] = $(value).val(); });
        $("#msgNames").text("共选中"+names.length+"条： "+arr.join(","));
    });
});
<input type="checkbox" name="names" value="tom" />tom
<input type="checkbox" name="names" value="jim" />jim
<input type="checkbox" name="names" value="lily" />lily
<p id="msgNames"></p>
```

表单选择器

- $(":input")选取所有<input>、<textarea>、<select>和<button>元素。和$("input")不一样， $("input")只获得<input>
- $(":text")选取所有单行文本框，等价于$("input[type=text]")
- $(":password")选取所有密码框。同理还有:radio、:checkbox、:submit、:image、:reset、:button、:file、:hidden。

JQuery的Dom操作

- 1、使用html()方法读取或者设置元素的innerHTML：
- alert($("a:first").html());
- $("a:first").html("hello");
- 2、使用text()方法读取或者设置元素的innerText：
- alert($("a:first").text());
- $("a:first").text("hello");
- 3、使用attr()方法读取或者设置元素的属性，对于JQuery没有封装的属性（所有浏览器没有差异的属性）用attr进行操作。
- alert($("a:first").attr("href"));
- $("a:first").attr("href", "http://www.rupeng.com");
- 4、使用removeAttr删除属性。删除的属性在源代码中看不到，这是和清空属性的区别。

动态创建Dom节点

- 使用$(html字符串)来创建Dom节点，并且返回一个jQuery对象，然后调用append等方法将新创建的节点添加到Dom中：
- var link = $("百度");
- $("div:first").append(link);
- $()创建的就是一个jQuery对象，可以完全进行操作
  - var link = $("百度");
  - link.text("百毒");
  - $("div:first").append(link);。
- append方法用来在元素的末尾追加元素。//$("#select1 option:selected").remove().appendTo($("#select2")) ;

$("#select1 option:selected").appendTo($("#select2")) ;

- prepend，在元素的开始添加元素。
- after，在元素之后添加元素（添加兄弟）
- before：在元素之前添加元素（添加兄弟）

删除节点

- （1）remove()删除选择的节点
  - 案例：清空ul中的项，代码见备注。$("ul li.testitem").remove(); 删除ul下li中有testitem样式的元素。
- remove方法的返回值是被删除的节点对象，还可以继续使用被删除的节点。比如重新添加到其他节点下
- var lis = $("#ulSite li").remove();
- $("#ulSite2").append(lis);
- 权限选择：var items = $("#select1 option:selected").remove(); $("#select2").append(items); 更狠的：$("#select1 option:selected").appendTo($("#select2"))
- （2）empty()是将节点清空，不像remove那样还可以添加到其他元素中。
- 案例：选择球队

补充

- 写代码的好习惯，{、(写完开始就写结束，省得忘了。，在JQuery中这样写就不容易写错了。
- 如果报错“例外被抛出”等，很可能是选择器表达式有问题，比如单词拼写错误、加了不必要的空格等。val是方法不是属性。jQuery是完全按照JavaScript的语法写出来的JavaScript函数库，没有任何的魔法，任何看似怪异的写法都是很合法的JavaScript语法。JQuery就是一堆写好的JavaScript函数库而已，没有什么特殊的，你也可以写出来，因此完全可以和普通JS代码混着用。最好不要dom、jQuery方式混着用。
- 晕了一天重新站在JavaScript、Dom角度重新审视JQuery这个小弟。

练习

- 加法计算器。两个文本框中输入数字，点击【=】按钮将相加的结果放到第三个文本框中。
- 十秒钟后协议文本框下的注册按钮才能点击，时钟倒数。设置可用性等JQuery未封装方法：attr("")
- 无刷新评论。
- 案例1：创建若干个输入文本框，当光标离开文本框的时候如果文本框为空，则将文本框背景色设置为红色，如果不为空则为白色。提示：焦点进入控件的事件是focus，焦点离开控件的事件是blur。
- 案例：选择球队。被悬浮行高亮显示（背景是红色），点击球队将它放到另一个的球队列表。

节点操作

- 替换节点：
- $("br").replaceWith("<hr/>");
- 将
替换为<hr/>

- 包裹节点
- wrap()方法用来将所有元素逐个用指定标签包裹：
- $("b").wrap("") 将所有粗体字红色显示

样式操作

- 获取样式 attr("class")，设置样式attr("class","myclass")，追加样式addClass("myclass")(不影响其他样式)，移除样式removeClass("myclass")，切换样式（如果存在样式则去掉样式，如果没有样式则添加样式）toggleClass("myclass")，判断是否存在样式：hasClass("myclass")
- 案例：网页开关灯的效果
- 练习：给body设置body{ filter:Gray; } 这个style就可以让网页变为黑白显示，做切换黑白效果的按钮。
- 点击表格行，被点击的行高亮显示（背景是黄色），其他行白色背景。监听每个tr的click事件，将点击的背景设置为黄色，其他的设置为白色背景。颜色定义为class样式。
- 练习：聚焦控件的高亮显示。颜色定义为class样式。 $("body *")，选择器*表示所有类型的控件。
- 练习：搜索框效果。焦点放入控件，如果文本框中的值是“请输入关键词”，那么将文本清空，并且颜色设置为黑色。如果焦点离开控件，如果文本框中是空值，那么将文本框填充为“请输入关键词”，颜色设置为灰色（Gray）。颜色定义为class样式。

RadioButton操作

- 取得RadioButton的选中值
  - $("input[name=gender]:checked").val()
  - <input id="Radio2" checked="checked" name="gender" type="radio" value="男" />男<input
  - id="Radio1" checked="checked" name="gender" type="radio" value="女" />女<input id="Radio3"
  - checked="checked" name="gender" type="radio" value="未知" />未知</p>
- 设置RadioButton的选中值：
  - $("input[name=gender]").val(["女"]);
  - 或者
  - $(":radio[name=gender]").val(["女"]);
  - 注意val中参数的[]不能省略

RadioButton操作2

- 对RadioButton的选择技巧对于CheckBox和Select列表框也适用
- 除了可以使用val批量设置RadioButton、CheckBox等的选中以外，还可以设定checked属性来单独设置控件的选中状态
- $("#btn1").attr("checked",true)
- 练习：权限选择框
- 练习：CheckBox的全选、全不选、反选

事件

- JQuery中的事件绑定：$("#btn").bind("click",function(){})，每次都这么调用太麻烦，所以jQuery可以用$("#btn").click(function(){})来进行简化
- 合成事件hover，hover(enterfn,leavefn)，当鼠标放在元素上时调用enterfn方法，当鼠标离开元素的时候调用leavefn方法。
- 事件冒泡：JQuery中也像JavaScript一样是事件冒泡
- 如果想获得事件相关的信息，只要给响应的匿名函数增加一个参数：e，e就是事件对象。调用事件对象的stopPropagation()方法终止冒泡。e. stopPropagation();
- $("tr").click(function(e) { alert("tr被点击"); e.stopPropagation(); });//注意函数的参数是e
- 阻止默认行为：有的元素有默认行为，比如超链接点击后会转向新链接、提交按钮默认会提交表单，如果想阻止默认行为只要调用事件对象的preventDefault()方法和window.event.returnValue=false效果一样。
- $("a").click(function(e) { alert("所有超链接暂时全部禁止点击"); e.preventDefault(); });

事件其他（*）

- 属性：pageX、pageY、target获得触发事件的元素(冒泡的起始，和this不一样)、which如果是鼠标事件获得按键（1左键，2中键，3右键）。altKey、shiftKey、ctrlKey获得alt、shift、ctrl是否按下，为bool值。keyCode、charCode属性发生事件时的keyCode（键盘码，小键盘的1和主键盘的keyCode不一样）、charCode（ASC码）。
- 移除事件绑定：bind()方法即可移除元素上所有绑定的事件，如果unbind("click")则只移除click事件的绑定。bind:+=；unbind:-=
- 一次性事件：如果绑定的事件只想执行一次随后立即unbind可以使用one()方法进行事件绑定：

鼠标

- 获得发生事件时鼠标的位置
- $(document).mousemove(function(e) {
- document.title = e.pageX + "," + e.pageY;
- });
- 在mousemove、click等事件的匿名响应函数中如果指定一个参数e，那么就可以从e读取发生事件时的一些信息，比如对mousemove等鼠标事件来说，就可以读取e.pageX、e.pageY来获得发生事件时鼠标在页面的坐标。
- 练习：跟着鼠标走的图片
- 练习：Tooltips效果
- 练习：美女图片详细解析层。

动画

- show()、hide()方法会显示、隐藏元素。用toggle()方法在显示、隐藏之间切换
- $(":button[value=show]").click(function() { $("div").show(); });
- $(":button[value=hide]").click(function() { $("div").hide(); });
- 如果show、hide方法不带参数则是立即显示、立即隐藏，如果指定速度参数则会用指定时间进行动态显示、隐藏，单位为毫秒，也可以使用三个内置的速度：fast（200毫秒）、normal（400毫秒）、slow（600毫秒），jQuery动画函数中需要速度的地方一般也可以使用这个三个值。
- 案例：QQTab效果
- 练习：优酷视频右边视频列表、视频详细信息效果
- 练习：大旗try.daqi.com的页面顶部的效果。

- 很多Dom做的功能用ASP.net服务端代码也能完成，但是那样会页面频繁刷新，性能、可用性非常差。能用Dom操作就不要用ASP.net服务端代码。先学HTML、JS、Dom，不要一上来就学asp.net，因为那样容易被ASP.Net好用所迷惑。
- 自己动手写WebOS，图标之间是浮动的图标，图标内是两个left 100%的一个图片和一个文字，在浮动图标的div上onclick的时候图标变背景色。ondblclick用JQueryUI实现点击图标显示对话框

难点

- id和jquery对象的区别。动态创建出来的对象在append之前是不能通过$("#id")来引用的。
- js中单引号与双引号

JQuery插件： JQuery cookie

- 什么是cookie：Cookie就是保存在浏览器上的内容，用户在这次浏览页面的时候向Cookie中保存文本内容，下次再访问页面的时候就可以取出来上次保存的内容，这样就可以得到上次“记忆”的内容。Cookie不是JQuery特有的概念，只不过JQueryCookie把它简化的更好用而已。Cookie就是存储在浏览器里的一些数据。
- Cookie需要浏览器的支持，浏览器的Cookie是可以禁用的，如果禁用了Cookie就不能使用了，不过一般不用考虑禁用Cookie的情况。Cookie的几个特征：Cookie是与域名相关的，所以163.com不能读取baidu.com记录的Cookie，正因为如此读取、设置Cookie的时候不用担心不同域名cookie的冲突；一个域名能写入的Cookie总尺寸是有限制的，一般是是几千字节，能写入的Cookie总条数一般是几十条，超过以后浏览器自己会根据自己的策略移除一些Cookie；Cookie不是写入以后一定下次能读出来，浏览器可能会定期清除、用户也可能会手动清除。写到Cookie中的数据一定是可有可无的数据，像防止投票作弊就不能用Cookie。

JQuery Cookie使用

- 使用方法，Cookie保存的是键值对
  - 1、添加对jquery.cookie.js
  - 2、设置值，$.cookie('名字', '值')。cookie中保存的值都是文本。
  - 3、读取值，var v=$.cookie('名字')
  - alert($.cookie("用户名"));
  - $.cookie("用户名","tom");在同域名的另外一个页面中也能读取到。
- 案例：点击登录以后保存用户名，再登录的时候读取上次保存的用户名，帮用户填上
- 设置值的时候还可以指定第三个参数，$.cookie('名字', '值', { expires: 7, path: '/', domain: 'itcast.cn', secure: true })，expires表示要求浏览器保留Cookie几天，这个值只是给浏览器的建议，可能没到时间就已经被清除了。可以实现“勾选【记录我的用户名10天】”的功能。如果不设定expires在浏览器关闭以后就清除，options参数用哪个设置哪个。
- 练习：实现“上次访问时间”功能；
- 练习：允许用户点击不同颜色的色块设置网页的背景颜色，然后记住用户的选择，下次进入的时候是用用户上次设置的背景色。todo：换CSS文件实现换肤。

常用JQuery插件

- JQuery官方的UI控件 JQueryUI
  - http://jqueryui.com/ 下发包
  - 演示常用方法，分析代码。
  - 表现和内容分离，语义化。
- JQuery.validate 表单验证插件
- Form,用于为表单提供直接的Ajax能力
- 所有插件列表http://plugins.jquery.com/

网页分析工具

- DebugBar→IE的
- IE8内置了开发人员工具（工具→开发人员工具），IE6/7需要安装Internet Explorer Developer Toolbar（下载地址见备注）。IECollection也带了安装包。可以禁用cookie、禁用JavaScript、清理Cookie、禁止缓存、调试JS等。
- Firebug→FireFox下的
- 多版本IE工具：IECollection，比IETester更强大。

传智播客 www.itcast.cn —高级软件人才实作培训专家

ASP.Net

讲师：杨中科

课前说明

- 内容：掌握基于ASP.Net的Web开发，ASP.Net内部原理、状态管理（Cookie、Session、ViewState等）、普通ASP.Net控件、数据验证、母版、ListView/Repeater等数据绑定控件、AJAX 、缓存、Membership、导航、自定义控件等。
- 目标：能够使用ASP.Net开发常见的动态网站功能，并且和Dom、JQuery等客户端技术结合进行网站的开发。
- 参考书：《ASP.NET 3.5 揭秘》
- 注意：原理先行，对于ASP.Net来说弄明白了原理才能学得更快。会讲一些原理性的非常规用法代码，用※标识，不要学这种写法。
- Java班先讲Servelet再讲JSP，.Net也是先讲HttpHandler再讲WebForm。

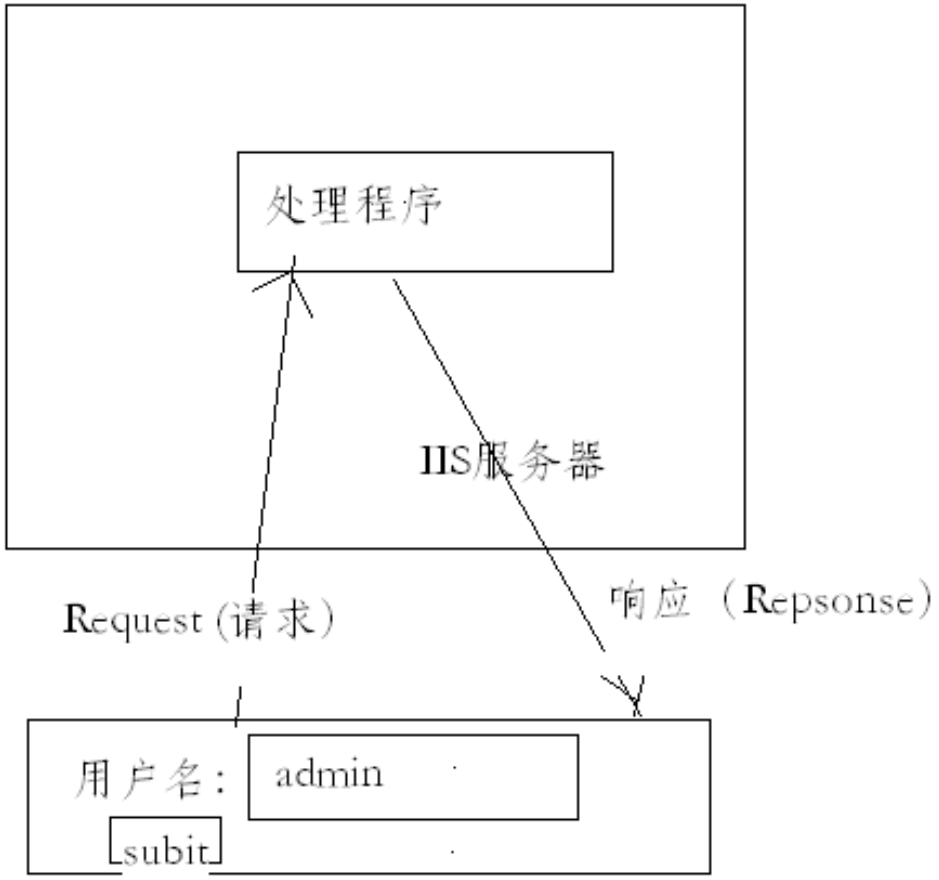
什么是ASP.Net

- ASP.Net是一种动态网页技术，在服务器端运行.Net代码，动态生成HTML。可以使用JavaScript、Dom在浏览器端完成很多工作，但是有很多工作无法在浏览器端完成，比如存储数据、访问数据库、复杂的业务逻辑运算、安全性要求高的逻辑运算等。
- 演示第一个ASP.Net页面：加法计算器。新建Web应用程序。 ※
- 服务端控件和HTML控件的生成关系：在aspx页面中可以使用服务端控件，简化开发，浏览器只认html，因此服务端控件会渲染到浏览器成html，TextBox→<input type="text"/>。
- 服务器控件不是新的控件，在浏览器端仍然是生成html标签。服务端控件虽然好用，但是也有缺点，并不是什么地方用服务器端控件都好，具体后面讲。

Web应用程序和网站

- WebApplication（Web应用程序）和WebSite（网站）的区别，WebSite是为了兼容从ASP转过来的开发人员的习惯而存在的，用起来简单，比如不需要创建命名空间、CS代码修改以后不需要重启就能看到变化（无论是WebSite还是WebApplication，修改aspx都不需要重启。 WebApplication每次修改以后点击【生成解决方案】也能立即看到修改效果），但是不利于工程化开发，比如代码出错不容易发现，代码不分命名空间。开发技术上没有任何区别，只是开发、调试习惯不同而已。
- 方便开发不用每次调试都设定起始页，在项目的选项中设定【Web】→启动操作→当前页面，这样当前激活的页就是起始页。
- 讲简单基础知识时用WebSite；讲高级技术和做项目的时候用WebApplication。

请求→处理→响应



自己动手写动态网站

- 入门1.html
  - <form action="入门1.ashx">
  - <input type="text" name="username" /><input type="submit" />
  - </form>
- 新建一个【一般处理程序】入门1.ashx， ProcessRequest中写
  - context.Response.ContentType = "text/html";
  - string username = context.Request["username"];
  - context.Response.Write(username + "你好"+Guid.NewGuid());
- 每当用户请求访问ashx页面的时候，ProcessRequest方法就会被调用，在这里通过访问context.Request获得访问者的请求参数等。然后在ProcessRequest中通过context.Response向浏览器发回数据给浏览器。ProcessRequest结束之时就是服务器为这个浏览者本次访问服务完成之时
- 浏览器向服务器端提交数据，被提交数据的表单（input、select、textarea等）放到form中，form中通过action属性设定表单被提交给哪个页面，为了在服务端取出表单项的值，需要在HTML中为表单元素设定name属性，注意id是给JS操作Dom用的，name才是提交给服务器用的。在服务器端用context.Request["username"]来根据表单项的name来获得提交的属性值。通过context.Response.Write向浏览器输出处理后的显示HTML内容。

“返回”提交页面

- 如果还想返回提交页面，那么需要自己绘制。
  - context.Response.Write(@"
  - <form action='入门1.ashx'>
  - <input type='text' name='username' value="+username+@" /><input type='text' /><input type='submit' name='sb' />
  - </form>");
- 并没有真的返回提交页面，只是看起来像罢了。

“返回”提交页面改进版

- 为了请求、返回的内容一样，将页面保存为一个htm模板文本，模板中有一些待填值的占位符，第一次进入页面的时候就直接访问ashx，读取htm模板，将待填值占位符设置为空，然后输出到浏览器。
- 为了区分是第一次直接进入页面还是点击提交以后重新进入ashx，在form中增加一个隐藏字段：<input type="hidden" name="ispostback" value="true" />，如果能够从Request中读取到ispostback=true就说明是点击提交以后重新进入ashx，否则就是第一次进入ashx。ispostback就是一个标志位。
- ASP.Net中将Web虚拟路径（/images/1.jpg）转换为磁盘全路径（d:/wwww/mysite/images/1.jpg）的方法是HttpContext.Current.Request.MapPath("/1/入门2.htm")。
- 实现思路：在ProcessRequest中首先从Request中读取ispostback，如果读取到true，说明是提交进入的，就加载模板，并且进行占位符用计算后的值替换，否则就将模板中的占位符清空直接输出给浏览者。代码见备注※
- 刚进入hello2.ashx的时候是直接向浏览器输出内容，用户在输出的内容中填入数值，再点击提交，服务器就知道“提交回来了”（PostBack）
- Http是请求、响应的模型，服务器不会来读取浏览器的网页，能够得到的就是客户端网页提交过来的数据。
- 如果访问Hello2.ashx，多次点击刷新，都是“直接进入”
- 如果提交表单，再多次点击刷新，都是“提交进入”
- 所有表单都是提交的以name为key，以value为值的内容，其他属性是不会被提交到服务器的。
- 文本框上次输入的值在提交表单后又显示出来并不是理所当然的，是开发人员帮着读取提交上来的值然后渲染上去的。这就是ASP.Net中ASPX和CS的关系。用aspx重写这个程序，使用IsPostBack等属性，对比。IsPostBack是依赖于ViewState的，所以禁用了页面的ViewState，IsPostBack永远为False。

Get与Post

- 还可以设定form的method属性指定表单提交方式，get（默认值）是通过URL传递表单值，post传递的表单值是隐藏到http报文中，url中看到不。
- 点击type=submit的按钮会自动提交表单。
- get和post的区别(常考)：get是通过url传递表单值，post通过url看不到表单域的值；get传递的数据量是有限的，如果要传递大数据量不能用get，比如type="file"上传文章、type="password"传递密码或者<textarea>发表大段文章，post则没有这个限制；post会有浏览器提示重新提交表单的问题，get则没有(加分的回答)。对于Post的表单重新敲地址栏再刷新就不会提示重新提交了，因为重新敲地址就没有偷偷提交的数据了
- Get方式URL数据格式。服务端文件名后跟着“?”，由于客户端可能向服务器端提交多个键值对，键值对之间用“&”进行分割，如果URL中有汉字、特殊符号等，则需要对URL进行编码。
- 表单域只有设定了name的才会被提交给服务器（用get方式看的清楚）。如果给submit按钮设定name，那么按钮的value也会被提交给服务器

数值自增※

- 实现input的自增：点击按钮input中的值自动增加，代码见备注。点击刷新就一直是值不变，只有点击提交才会变，分析原理。用开发人员工具篡改文本框的值自增就从新的值开始了。
- 使用aspx重写input的自增。
- 练习：加法计算器。常见错误：把htm设置成了起始页。type=submit才会自动提交表单，type=button不会自动提交。从ashx启动和从htm启动的区别
- 为什么单使用div在服务器取不出来值呢？因为不是服务器来读取客户的网页，而是浏览器收集客户再表单中输入的字段，然后形成请求参数发给服务器处理程序，由于没有把div当前的innerText发给服务器，所以服务器无法得知当前的值。也不要幻想有办法能将div的innerText提交给服务器，因为只有设定了name的input、textarea、select的value属性值才会被提交给服务器。
- 实现div内文本的自增。因为服务器不记得上次给浏览器的值是什么，而且不像input那样会将上次的值重新提交回来，因此浏览器需要用一个隐藏字段将上一次的值保存下来。代码见备注
- 非表单元素无法将客户端的元素值传递给服务器端，即使是表单元素也只能传递value值，对于其他属性值比如背景颜色、大小等也是无法传递的，因此对于这些值都要存在隐藏字段中。这就是ASP.Net中ViewState的实现原理。
- 其实div中的只是起到显示作用而已，使用IE的“开发人员工具”修改div的innerText，然后点击按钮发现值并没有按照我们修改以后的递增。而修改input版本的则有效果。

ViewState初探（重点，常考）

- Label版本的值存到了ViewState中，TextBox版本的不用存，因为TextBox就是input，自己就会提交给服务器，不需要隐藏字段
- 用asp.net重写Div文本自增（还要同时递增Label的宽度，注意Width的单位是Unit类型，不是简单的int）
  - Label1.Text = (Convert.ToInt32(Label1.Text)+1).ToString();
  - Label1.Width = new Unit(Label1.Width.Value + 10);
- 查看生成的源代码，ASP.Net将所有隐藏内容统一放到了名字为__VIEWSTATE的隐藏字段中，使用序列化算法将所有隐藏内容放到一个字符串中。点击几次在使用ViewStateDecoder这个工具查看ViewState内容，发现了确实将这些改变的内容放到了ViewState中。存储非表单域、非value值的容器。
- 禁用ViewState的方法，禁用单个控件的ViewState设定enableviewstate=false，禁用ViewState以后TextBox版本不受影响，Div版本受影响，因为input的value不依靠ViewState。禁用整个页面的，在aspx的Page指令区加上EnableViewState="false" 。内网系统、互联网的后台可以尽情的用ViewState。
- 回答ViewState原理的时候：说Input版本（TextBox）自增和Div版本（Label）的不同。（完美！！！）

无状态Http

- Http协议是无状态的，不会记得上次和网页“发生了什么”（故事：24小时记忆）。服务器不记的上次给了浏览器什么，浏览器需要记住这些值（input就是记到value中，对于其他的值就要放到隐藏字段中，比如ViewState），下次再提交服务器的时候（请在我的宽度基础上增加10，）就要把上次的值提交给服务器，让他想起来。如果要知道上一次的状态，一个方法是在对浏览器响应结束之前将状态信息保存到页面表单中，下次页面再向服务器发出请求的时候带上这些状态信息，这样服务器就能根据这些状态信息还原上次的状态了，类似于去看病的病历本。
- 状态信息保存在隐藏字段中的缺点：加大网站的流量、降低访问速度、机密数据放到表单中会有数据欺骗等安全性问题。故事：自行打印存折，因为余额不是写到存折这个隐藏字段中的，唯一的关联就是卡号。要把机密数据放到服务器，并且区别不同的访问者的私密区域，那么就要一个唯一的标识。

Cookie

- 表单是和页面相关的，只有浏览器端提交了这些数据服务器端才能得到。而有时候希望在服务端任意的地方存取一些和访问者相关的信息，这时候就不方便将这些信息保存到表单中了，因为如果那样的话必须随时注意在所有页面表单中都保存这些信息。Cookie是和站点相关的，并且每次向服务器请求的时候除了发送表单参数外，还会将和站点相关的所有Cookie都提交给服务器，是强制性的。Cookie也是保存在浏览器端的，而且浏览器会在每次请求的时候都会把和这个站点的相关的Cookie提交到服务器，并且将服务端返回的Cookie更新回数据库，因此可以将信息保存在Cookie中，然后在服务器端读取、修改。服务器返回数据除了普通的htmlm数据以外，还会返回修改的Cookie，浏览器把拿到的Cookie值更新本地浏览器的Cookie就可以。
- 哪怕请求jpg、js、css这种文件也会带着Cookie，因为服务器端可能要进行Session的操作，比如判断是否登录。互联网优化的案例：图片服务器和主站域名不一样，降低Cookie流量的传输。面试时聊网站调优
- 案例，一个页面设置Cookie，一个页面读取Cookie
  - 设置值的页面：Response.SetCookie(new HttpCookie("UserName", TextBox1.Text));

Session原理

- private int i = 0;//每次请求来了都会new一个新的实现了IHttpHandler接口的类“变量1”的实例进行处理，用完了就GC掉，所以不会保持上次的值。（有时会考）
- Cookie不能存储过多信息。如果想保存大量的数据，可以保存一个Guid到Cookie中，然后在服务器中建立一个以Guid为Key，复杂数据为Value全局Dictionary。static字段对于不同用户也只有一份，因此用static实现多用户共享数据。代码见备注※。
- ASP.Net已经内置了Session机制，把上面的例子用ASP.NetSession重写。不要放太多的对象到Session，Session会有超时销毁的机制，发帖（服务器不可能知道浏览器是否在开着，什么时候关闭），发帖计时，在线时间统计，靠请求来判断是否活着。Cookie是存在客户端，Session是存在服务器端，目的是一样的：保存和当前客户端相关的数据（当前网站的任何一个页面都能取到Session、Cookie）。不能放太大的数据，放的数据是object。
- 可以看到Session机制并不是Http协议规定的，是ASP.net实现的，现在PHP、JSP等大部分服务端技术都实现了Session，原理都差不多。
- 案例：用Session实现验证码。HttpHandler要能够操作Session，要实现IRequiresSessionState接口。为什么每次点击值都变化？很正常，因此每次页面点击页面都会刷新，就向ashx重新请求图片，ashx的ProcessRequest都会执行。正常网站登录成功就进入主页面，没机会让你看到变化，但是一旦输入错误也是变化。
- 点击图片产生新的验证码。 每次点击都生成一个新的地址，让浏览器去请求。在AJAX中还会用。

http协议简介

- Web开发是和Http协议打交道的，必须了解Http协议。Http协议版本：Http/0.9、Http/1.0、Http/1.1，现在主流的是Http/1.1版本。
- Http协议分析工具：
  - 1、DebugBar，Http(S)标签的内容。免费的。只能分析当前浏览器中的内容。
  - 2、httpwatch，收费的，只能分析当前浏览器中的内容。推荐使用
  - 3、HttpAnalyzer，收费的，能分析计算机上所有的Http请求数据。
- Http协议的几个概念：
  - 1.连接(Connection)：浏览器和服务器之间传输数据的通道。 一般请求完毕就关闭，不会保持连接。
  - 2.请求(Request)：浏览器向服务器发送的“我要***”的消息，包含请求的类型、请求的数据、浏览器的信息（语言、浏览器版本等）。
  - 3.响应(Response)：服务器对浏览器请求的返回的数据，包含是否成功、错误码等。

http请求报文

- 用httpwatch查看访问一个网站（用DiscuzNT测试环境）的响应情况。敲入一个网址后，浏览器向服务器发出请求。页面中的图片、js、css在单独的请求中。
- GET / HTTP/1.1表示向服务器用GET方式请求首页，使用HTTP/1.1协议
- Accept-Encoding gzip, deflate表示浏览器支持gzip、deflate两种压缩算法
- Accept-Language zh-cn 表示浏览器支持的语言，很多进入后自动就是中文界面的国际网站就是通过读取这个头的值实现的。
- Connection Keep-Alive。一般情况下，一旦Web服务器向浏览器发送了请求数据，它就要关闭TCP连接，然后如果浏览器或者服务器在其头信息加入了Connection:keep-alive，则TCP连接在发送后将仍然保持打开状态，于是，浏览器可以继续通过相同的连接发送请求。保持连接节省了为每个请求建立新连接所需的时间，还节约了网络带宽。
- Cookie是浏览器向服务器发送和当前网站关联的Cookie，这样在服务器端也能读取浏览器端的Cookie了。
- User-Agent为浏览器的版本信息。通过这个信息可以读取浏览器是IE还是FireFox、支持的插件、.Net版本等。

http响应码

- 浏览器向服务器发出请求，服务器处理可能是成功、可能是失败、可能没有权限访问等原因，服务器会通过响应码来告诉浏览器处理结果。
  - "200" : OK
  - 301" : Moved Permanently 永久转移
  - "302" : Found 暂时转移
  - "307" : Temporary Redirect
  - "400" : Bad Request 错误请求，发出错误的不符合Http协议的请求
  - "401" : Unauthorized 未认证。一般需要用户名、密码才能登陆。
  - "403" : Forbidden 禁止
  - "404" : Not Found 未找到。演示访问一个不存在的页面看报文
  - "500" : Internal Server Error 服务器内部错误。演示页面抛出异常。
  - "503" : Service Unavailable。一般是访问人数过多。
- 200段是成功；300段需要对请求做进一步的处理；400段表示客户端请求错误；500段是服务器的错误。

服务器返回的报文

- Server: Cassini/3.5.0.5 表示服务器的类型
- Content-Type: text/html; charset=utf-8 表示返回数据的类型
- 服务器通过Content-Type告诉客户端响应的数据的类型，这样浏览器就根据返回数据的类型来进行不同的处理，如果是图片类型就显示，如果是文本类型就直接显示内容，如果用html类型就用浏览器显示内容，如果是下载类型就弹出下载工具等。
- 常用Content-Type：text/HTML、image/GIF、image/JPEG、text/plain、text/javascript、application/x-excel 、application/octet-stream（二进制文件）
- Content-Length: 19944表示后续数据消息体的长度，报文头只是描述，返回的具体数据（比如HTML文本、图片数据等）在两个回车之后的内容中。

Http其他

- http是无状态的，不会记得“上个请求***”，所以哪怕是同一个页面中的js、css、jpg也都要重复的提交Accept-Language、Accept-Encoding、Cookie等。
- 网页中如果有图片、css、js等外部文件的话图片、css、js都在单独的请求中，也就是并不是页面的所有内容都在一个请求中完成，而是每个资源一个请求。
- 一般情况下，只有浏览器请求服务器端，服务器端才有给浏览器响应数据，不会主动向浏览器推送数据，这样是安全考虑，也是提高服务器的性能考虑。如果要服务器向浏览器推送数据，则需要使用ServerPush等额外的技术。
- Http是“请求—响应”的工作方式，因此页面会不断刷新，如果不希望页面刷新则要使用AJAX等技术。
- 断点续传的原理。多线程下载基于断点续传。(*)

请求响应模型的例子(重点)

- 按钮实现表格行删除效果；使用超链接进行删除，代码见备注。
- 这就是asp.net中数据绑定控件中行按钮和行超链接实现方式的不同。ListView中Button、HyperLink两种行删除方式。按钮方式是将行的id通过表单提交到服务器，行超链接的方式是通过超链接的url通过get的方式提交给处理页面，超链接的方式由于没有提交所有的表单信息，因此很多服务端控件的高级用法用不了。
- 用aspx重写，超链接的因为没有向服务器提交ViewState等隐藏字段，所以处理时IsPostBack是False，而按钮的则是提交了表单，所以IsPostBack=True。可以在超链接的href中写表单提交的JavaScript，这样就是WebForm中LinkButton的原理。
- 客户端、服务端由于在两台计算机中，所以无法做到两边变量的互相读取或者两边函数的互相调用，所以如果想看起来好像做到，那么必须通过提交的方式来将客户端变量值做为一个表单字段提交到服务器、或者服务端将服务端变量打印到客户端代码中。

Web开发的一些基本原则

- 最小权限原则。只允许用户做***，而不是“不允许用户做***”
- 浏览器查看的是服务端代码的执行输出的文本，除非服务器有漏洞，否则浏览者无法查看服务端的aspx、cs代码，目标另存为也是保存的aspx的执行结果，而看不到aspx的源代码。js、html是被输出到浏览器上执行的，因此无法禁止浏览者查看js、html。
- C#代码是运行在服务器端的，JS代码是运行在浏览器客户端的
- 能在浏览器端完成的事情，就不要到服务端去做。
- 客户端是不可信的。

原则1

- C#代码是运行在服务器端的，JS代码是运行在浏览器客户端的
- 按钮确认提交的实现在asp:Button的OnClientClick中写。
  - <input type="submit" name="delete" value="删除" onclick="return confirm('真的要删除吗？')" />。OnClientClick和OnClick的区别：鸠占鹊巢的OnClick
  - 代码是运行在浏览器端的，和服务器端没有关系。
- 在服务器端“弹出消息窗口”
  - context.Response.Write("<script type='text/javascript'>alert('删除成功')</script>");明白为什么即可。
  - 并不是真的是在服务器端运行的，只是生成了JavaScript代码到浏览器端，浏览器会在解析文档的时候运行alert，不推荐用这种写法，读懂即可，推荐用后面讲的RegisterClientStartupScript。只是渲染到浏览器端，所以并不会得到对话框关闭服务端的代码才会执行下去（在context.Response.Write("<script type='text/javascript'>alert('删除成功')</script>")后设置断点）
  - 对于服务器端的代码来说，生成一堆HTML代码就是一堆字符串，没有任何意义，只有到了浏览器端执行才有意义。

原则1

- 案例1、在项目中添加对System.Windows.Forms的引用，然后MessageBox.Show("Hello");，用CassiniDev.exe启动测试程序让学生们远程测试。证明C#代码是运行在服务器端的。
- 127.0.0.1是回环地址（LoopBack），就是表示通过回环地址访问本机，哪怕是本机外网地址也访问不了。localhost就是127.0.0.1别名。是无法在外部访问。
- 0.0.0.0 任意IP（Any IP），不用写死绑定的IP了，通过任何一块网卡都可以访问网络程序。
- 案例2、伟大的ASP.Net，可以在访问者磁盘中创建木马文件
- File.WriteAllText("c:/muma.exe", "木马(){葵花点穴手();降龙十八掌();熊猫烧香();}");
- 用CassiniDev.exe启动测试程序让学生们远程测试（VS内置的服务器不能远程访问）。exe生成到了服务器的磁盘中，而不是访问者的磁盘中，因为C#代码是运行在服务器中的，而不是浏览器中的，浏览器得到的只有返回的HTML内容。
- 案例3、开两个页面分别访问点击自增1的界面，互不影响。因为状态是保存在页面的ViewState中的。

原则2

- 能在浏览器端完成的事情，就不要到服务端去做。
- 按钮隐藏一个控件就不要写服务端代码，在客户端用JavaScript、dom来操作就可以。比如要操作数据库，显然是在浏览器端做不到的，这时候就要写服务端代码。校验用户名、密码这样的操作可以放到浏览器端（用户名、密码是写死的），技术上可以，但是这样安全性太差，因此必须放到服务器端。

原则3

- 客户端是不可信的。
  - 客户端验证不能代替服务端验证
  - 不要把敏感数据、算法写在客户端
  - 不要把机密信息在html中隐藏的方式来保证安全
  - 应该是在机密页面打开之前做权限校验，而不是在一个页面中做校验，如果正确就倒向机密页面，不正确就不导向。
  - 不要轻信用户提交上来的数据

客户端验证不能代替服务端验证

- 设置取款金额不能高于100元
  - 客户端：<form id="form1" runat="server" onsubmit="if(parseInt(document.getElementById('TextBox1').value,10)>100){alert('最多只能取款100元');return false;}">
  - 服务器端：Label1.Text = "取款成功，金额："+TextBox1.Text;
  - <asp:Button来讲，onclick是服务端事件，OnClientClick是最终生成到客户端浏览器中的onclick代码。
- 如果禁用JavaScript（Internet选项→安全→自定义级别→脚本→活动脚本→禁用，可以用“开发人员工具”），那么客户端JavaScript校验就被禁用了，就可以取款多于100元了。
- 在服务端也要进行数据校验，代码见备注。
- 客户端校验是为了很好的客户端体验，服务器端校验是最后一次把关，防止恶意请求。后面要讲的ASP.Net Validation就是ASP.Net内置的数据校验技术，会在客户端和服务器端同时校验

不要把敏感数据、算法写在浏览器端

```
$("#btnLogin").click(function() {
        if ($("#username").val() == "admin" &&
    $("#password").val() == "123456") {
            alert("登录成功！ ");
        }
        else {
            alert("登录失败！ ");
        }
    });
```

- 用户在浏览器中查看源代码就可以看到用户名、密码是什么。在动态网站空间还是很稀有的那个年代，很多个人主页都是用这种方法进行数据的“保密”。

不要把机密信息隐藏在html中

- 只有密码输对了才显示下载地址，实现代码见备注
- 应该在服务端控制密码不对则Visible=False，服务端控件的HyperLink1.Visible = false是根本不输出到客户端。在和JQuery等结合的时候是无法用$("#控件id").show()来显示Visible=False的控件，因为控件根本没有渲染到HTML中。
- 应该是在机密页面打开之前做权限校验，而不是在一个页面中做校验，如果正确就倒向机密页面，不正确就不导向。

XSS漏洞

- 不要轻信用户提交上来的数据
- alert消息太难看，因此开发一个ashx页面用来统一展示消息 ShowMessage.ashx
  - context.Response.Write(" "+context.Request["Msg"]+" ");
  - context.Response.Write("返回上一页");
  - 系统内部需要弹出消息的时候只要将用户Redirect到 ShowMessage.ashx?Msg=消息就可以，比如
  - Response.Redirect("ShowMessage.ashx?Msg=用户名不能为空！");
- 利用漏洞1：送奖品的消息框。
- 利用漏洞2：收集帐号、密码
- 上面的这两个例子就是XSS（跨站脚本，Cross-site scripting）

XSS漏洞2

- 用户发帖时也存在XSS的问题。将发帖内容保存到一个文本文件中，不用数据库，降低问题复杂度。代码见备注。
- 我们可以对请求的数据做检测，如果请求数据中有<等就认为是恶意请求，禁止提交。aspx默认就是采用这种策略，这样做的缺点是如果做的是一个程序员论坛，程序员就无法发表HTML代码的帖子了，因此更好的处理策略是将用户发表的内容按照原样显示出来，而不是以HTML的方式显示出来。使用HttpUtility.HtmlEncode就可以将字符串中的<、>等特殊字符转换为HTML显示的字符，也就是不把<script>当成定义脚本的标签，而是当成“<script>”这样可以在页面上直接显示出来的内容。
- 修改看帖代码，将context.Response.Write(line + "<hr/>");修改为context.Response.Write(HttpUtility.HtmlEncode(line)+"<hr/>");即可。

XSS漏洞3

- aspx中默认对请求的数据进行了校验，如果数据中有<、>等有潜在XSS攻击的字符，则会报错。对于一些CMS系统等确实需要提交HTML内容的地方要关闭它，在页面顶部的Page中加入 ValidateRequest="false" 这个属性(VS2010中还要按照报错信息修改Web.config)。
- 在显示的时候如果需要对内容在显示之前进行HTMLEncode，除了可以使用HttpUtility.HtmlEncode进行手动编码的话，还可以使用Literal控件显示，如果修改Literal的Mode属性为Encode那么就会自动进行HtmlEncode然后显示。

特殊路径标识“~”

- 和“/表示网站根目录(域名)、../表示上级目录、./表示当前目录”等Http标准定位不一样，~是ASP.Net定义的特殊符号，是ASP.Net内部进行定义推荐的用法，推荐资源定位都使用~从应用根目录开始定义。应用根目录和网站根目录的区别在于：如果将一个应用部署到http://www.rupeng.com/search这个目录下，应用的根目录是“http://www.rupeng.com/search”，网站的根目录是“http://www.rupeng.com/”（创建WebSite进行演示，因为不同的WebSite都是在同一个网站根目录下的），因此最好用“~”，“~”并不会被浏览器认，因此ASP.Net会将这个路径转换为相对于网站的根目录的全路径再输出到浏览器。
- 举例，用~，而不是/的好处。WebSite。

编程处理“~”

- 如果在服务端控件中（使用runat=server的控件）会自动将“~”进行转换，如果在HTML控件或者需要在代码中转换的话可以使用VirtualPathUtility类中静态方法进行虚拟路径、全路径等的转换，比如VirtualPathUtility.ToAbsolute("~/a/b1.aspx")就是将虚拟路径转换为相对于应用根的全路径，也就是/WebSite4/a/b1.aspx
- （*）VirtualPathUtility类主要方法：string AppendTrailingSlash(string virtualPath)：如果路径virtualPath最后没有“/”则添加；string Combine(string basePath, string relativePath)，将两个路径进行合并；string GetDirectory(string virtualPath)，返回虚拟路径的目录部分；string MakeRelative(string fromPath, string toPath)，计算两个虚拟路径的相对路径；ToAbsolute，转换为绝对路径

Request对象

- 1、（*） Request.AppRelativeCurrentExecutionFilePath，获取当前执行请求相对于应用根目录的虚拟路径，以~开头，比如“~/Handler.ashx”，
- 2、（*） Request.PhysicalApplicationPath，获取当前应用的物理路径，比如D:\我的文档\Visual Studio 2008\WebSites\WebSite4\
- 3、（*） Request.PhysicalPath，获取当前请求的物理路径，比如D:\我的文档\Visual Studio 2008\WebSites\WebSite4\Handler.ashx
- 4、（*） Request.RawUrl获得原始请求URL、Request.Url获得请求的URL，区别涉及到URL重写的问题
- 5、Request.UrlReferrer 网页的来源，可以根据这个判断从百度搜的哪个关键词、防下载盗链、防图片盗链，可以伪造(比如迅雷)。"本图片仅供如鹏网内部交流使用"，在DZ中测试。全局防盗链用Globals.asax
- 6、Request.UserHostAddress获得访问者的IP地址
- 7、（*） Request.UserLanguages获得访问者浏览器支持的语言，可以通过这个实现对不同语言的人显示不同语言的页面。
- 8、Request.Cookies 获取浏览器发过来的浏览器端的Cookie，从它里面读取Cookie值，比如context.Request.Cookies["mysessionid"]，使用Request.Cookies 的时候一般只是读取，将Cookie写回浏览器要用Response.Cookies
- 9、Request.MapPath(virtulPath)将虚拟路径转换为磁盘上的物理路径，Request.MapPath("~/a/b.aspx")就会得到D:\2008\WebSites\WebSite4\a\b.aspx

Response对象

- 响应的缓冲输出：为了提高服务器的性能，ASP.Net向浏览器Write的时候默认并不会每Write一次都会立即输出到浏览器，而是会缓存数据，到合适的时机或者响应结束才会将缓冲区中的数据一起发送到浏览器。
- Response对象的主要成员：
  - 1、Response.Buffer、Response.BufferOutput：经过Reflector反编译，发现两个属性是一样的，Buffer内部就是调用的BufferOutput。这个属性用来控制是否采用响应缓存，默认是true。
  - 2、Response.Flush()将缓冲区中的数据发送给浏览器。这在需要将Write出来的内容立即输出到浏览器的场合非常适用。案例：大批量数据的导入，显示正在导入第*条数据，用Thread.Sleep模拟耗时。
  - 3、Response.Clear()清空缓存区中的数据，这样在缓存区中的没有发送到浏览器端的数据被清空，不会被发送到浏览器。在用aspx输出非html的例子中经常看到用clear来输出httpmodule等给附加的内容（不推荐，推荐用ashx）
  - 4、Response.ContentEncoding输出流的编码。
  - 5、Response.ContentType 输出流的内容类型，比如是html（text/html）还是普通文本（text/plain）还是JPEG图片（image/JPEG）。

Response对象2

- 6、Response.Cookies 返回给浏览器的Cookie的集合，可以通过它设置Cookie
- 7、Response.OutputStream 输出流，在输出图片、Excel文件等非文本内容的时候要使用它
- 8、Response.End() 终止响应，将之前缓存中的数据发给浏览器，End()之后的代码不会被继续执行。在终止一些非法请求的时候，比如盗链等可以用End()立即终止请求。
- 9、Response.Redirect(url) 重定向浏览器到新的网址。即可以重定向到站外网址也可以重定向到站内网址。Response.Redirect("http://www.rupeng.com")、Response.Redirect("a.htm")。Redirect是向浏览器发回302重定向，是通知浏览器“请重新访问url这个网址”，这个过程经历了服务器通知浏览器“请重新访问url这个网址”和浏览器接到命令访问新网址的过程。使用HttpWatch查看整个响应过程的Http报文。用Redirect因为是浏览器自己去重新访问新网址的，所以在地址栏中是可以看到网址的变化的。后面会用来防止刷新浏览器时提示“重试”。
- 10、Response.SetCookie(HttpCookie cookie)，向输出流中更新写到浏览器中的Cookie，如果Cookie存在就更新不存在就增加。是对Response.Cookies的简化调用。
- 11、Response.Write()向浏览器输出内容。
- 12、(*)Response.WriteFile(filename)向浏览器输出文件。比如Response.WriteFile("c:/boot.ini")

Server对象

- Server是context的一个属性，是HttpServerUtility类的一个对象
- Server.HtmlDecode()、Server.HtmlEncode() Server.UrlEncode()、Server.UrlDecode()是对HttpUtility类中相应方法的一个代理调用。个人推荐总是使用HttpUtility，因为有的地方很难拿到Server对象。别把HtmlEncode、UrlEncode混了，UrlEncode是处理超链接的， HtmlEncode是处理html代码的。
- Server.Transfer(path) 内部重定向请求，Server.Transfer("JieBanRen.aspx")将用户的请求重定向给JieBanRen.aspx处理，是服务器内部的接管，浏览器是意识不到这个接管的，不是象Response.Redirect那样经历“通知浏览器‘请重新访问url这个网址’和浏览器接到命令访问新网址的过程”，是一次http请求，因此浏览器地址栏不会变化(联想，呼叫中心坐席告诉客户一个号码和帮客户转接的区别)。因为是内部接管，所以在被重定向到的页面中是可以访问到Request、Cookies等这些来源页面接受的参数的，就像这些参数是传递给他的，而Redirect则不行，因为是让浏览器去访问的。注意Transfer是内部接管，因此不能像Redirect那样重定向到外部网站。 (常考)
- 使用Server.Transfer不能直接重定向到ashx，否则会报错“执行子请求出错”。
- 有的时候不能拿到HttpContext对象，比如在Global.asax中(后面讲)，可以通过HttpContext.Current拿到当前的HttpContext，进而拿到Response、Request、Server等。
- Server.MapPath。如果拿不到Server则使用HostingEnvironment.MapPath()方法

HttpHandler的无状态

- 一个浏览者发出的请求都是由实现了IHttpHandler接口的对象进行响应，由于下次访问不一定还是上次那个对象进行响应，上次响应完毕对象可能已经被销毁了，写的类变量值早就不存在了，因此不要将状态信息保存到类变量中。
- 编写一个ashx
  - private int i;
  - public void ProcessRequest(HttpContext context)
  - {
  - context.Response.ContentType = "text/plain";
  - context.Response.Write(i++);
  - }
- 多次刷新我们发现，变量根本不会记忆上次的值。
- 如果每次响应都创建一个对象的话会非常消耗资源，因此服务器可能会对于多个请求重用一个对象，这样如果实现IHttpHandler的IsReusable方法返回true，那么就表示这个HttpHandler允许对象重用。一般返回true就可以。

HttpHandler1

- HttpHandler是对请求的响应，可以输出普通的html内容，也可以输出图片、也可以输出一个文件（下载）
- 输出一幅动态创建的图片（能看懂就可以）
- 案例1：图片中显示访问者信息
- 案例2：填入朋友的姓名就能生成恶搞的图片链接
- 网上看到的注册、登录时候的验证码也是动态生成的图片、55.la也是这样实现的原理。

HttpHandler实现文件下载

- 如果HttpHandler输出的是html、txt、jpeg等类型的信息，那么浏览器会直接显示，如果希望弹出保存对话框，则需要添加Header：string encodeFileName = HttpUtility.UrlEncode("过滤词.txt"); Response.AddHeader("Content-Disposition", string.Format("attachment;filename=\"{0}\"", encodeFileName));其中filename后为编码后的文件名。filename段为建议的保存文件名
- 动态输出用处，不用再把资源保存到磁盘上再输出(不会有文件重名的问题，文件不生成在服务器端)。案例：点击链接弹出图片下载对话框。Web的原则：能直接将生成的内容以流的形式输出给浏览器，就不要生成临时文件。
- 用NPOI动态生成一个Excel表然后弹出对话框让用户下载，文件名是“用户列表.xls”。
- 练习：从数据库用户表导出数据到Excel文件，让用户下载。mdf放到App_Data下，asp.Net不用那段设置DataDirectory的代码，用DataReader的方式读取数据
- 练习：用户表增加一个级别字段。只有登录用户才能下载images下的图片文件（Session中标识是否登录），如果用户没有登录则首先重定向到登录界面让用户登录，用户登录成功则跳转到下载列表页面，下载链接固定写好即可。如果登录用户是普通用户则在图片左上角加上“免费用户试用”的字样。“安全退出”。画页面流程。
- 练习：给上面的程序加上登录验证码。

补充

- WebApplication每次修改以后点击【生成解决方案】也能立即看到修改效果，不需要重启浏览器。原理：生成以后才将变化的部分生成到dll，而WebSite则每次访问页面的时候会检查cs是否变了，变了则自动重新编译，所以每次修改以后立即有效果。
- 方便开发不用每次调试都设定起始页，在项目的选项中设定【Web】→启动操作→当前页面，这样当前激活的页就是起始页。
- VS2008一个Bug的发现：路径中有#的问题，发现问题的过程重现。

WebForm1

- 如果每次输出网页都直接用HttpHandler的话太痛苦了，所以一般生成html的时候都直接创建aspx（Web窗体，WebForm）。
- WebForm分为两个文件aspx和aspx.cs，aspx是页面模板，是页面描述文件，就是html的内容，和aspx结合的更好，不用像一开始那样程序员自己去填充模板，控件都是定义在aspx中，内联的JavaScript、CSS也是写在aspx中的，，服务端的C#代码是定义在aspx.cs中。aspx控件页面长相，cs控制程序逻辑，这种“前aspx后cs”的方式就被称为CodeBehind。aspx就是模板引擎，不需要再去寻找第三方的模板引擎。

WebForm2

- cs可以调用aspx中的控件，aspx中也可以访问cs中定义的字段、函数，还可以编写复杂的C#代码， for等所有C#代码都可以写在aspx中（不推荐）
- 前面 <%=UserName %> <%SayHello(); %> <%if (UserName == "aaa") { UserName = "bbb"; } %> 后面
- 在当前位置输出表达式的值的时候使用<%=UserName %>，不要丢了=，相当于在当前位置调用Response.Write(UserName)
- 使用的函数、代码相当于在这个位置调用函数、执行代码。注意aspx中调用cs的成员级别必须是protected或者public，不能是private的。
- <%%>中的代码是运行在服务器端的，是C#语法，其他部分是运行在浏览器端的，是html、JavaScript语法。

aspx、cs、dll之间的关系（*）

- 在WebForm的页面中执行下面的代码
  - Response.Write(this.GetType() + "
");
  - Response.Write(this.GetType().Assembly.Location + "
");
  - Response.Write(this.GetType().BaseType + "
");
  - Response.Write(this.GetType().BaseType.Assembly.Location + "
");
- 发现当前执行页面的类名是ASP.webform1_aspx这样的类名，父类才是ASPNETTest1.WebForm1
- 使用Reflector打开这个临时dll，反编译这两个类，发现ASPNETTest1.WebForm1是在VS中编写的aspx.cs类，而ASP.webform1_aspx则是一个继承自ASPNETTest1.WebForm1的子类，ASP.webform1_aspx代码是根据aspx内容动态生成的构建网页内容的类。综上，aspx最终也会生成一个类，这个类是继承自aspx.cs中的类。查看反编译以后的代码，可以看到就是编译生成了普通的.Net 代码。因为aspx生成的代码是cs类的子类，所以就明白了为什么“aspx中调用cs的成员级别必须是protected或者public，不能是private的。”

Page类成员

- 1、Request、Response、Server属性：对context.Request、context.Response、context.Server的简化调用。
- 2、AppRelativeVirtualPath属性：获得页面相对于应用根路径的路径，比如~/Default2.aspx
- 3、FindControl(ctrlId)，根据控件的id找到控件。一般情况下直接在代码中写控件id引用控件就可以了，但是对于有些场合：使用ListView等控件的模板、编写自定义控件等则需要使用FindControl来引用控件，FindControl返回值是Control，一般需要显式转换为相应的控件：TextBox txtBox = (TextBox)FindControl("TextBox1");txtBox.Text = "aaa";
- 4、IsPostBack、 Session
- 5、ResolveClientUrl(url)将虚拟路径转换为客户端访问时的路径，比如ResolveClientUrl("~/a/b.aspx")结果是a/b.aspx，这通常在ListView等控件的模板中输出HTML使用。基本就是对VirtualPathUtitlity.ToAbsolute简化调用。考虑当前页面的相对路径，生成的路径短。最常用。
- 6、ResolveUrl(url) 将虚拟路径转换为相对于网站根目录的路径，比如ResolveUrl("~/a/b.aspx")的结果是/WebSite4/a/b.aspx 。不考虑当前页面，VirtualPathUtitlity.ToAbsolute直接转换为一个全路径。

ASP.Net服务端基本控件介绍

- ASP.Net服务端控件是ASP.Net对HTML的封装，在C#代码中就可以用txt1.Text='abc'这种方式来修改input的值，ASP.Net会将服务端控件渲染成HTML代码输出给浏览器。服务端控件是ASP.Net非常吸引初学者、非常容易上手的东西，也是最被人诟病的东西。物尽其用，服务端控件在内网系统、互联网系统的后台部分等访问频率不高的地方用的还是很适合的。
- 所有的ASP.Net大部分都是从Control、WebControl类继承的，几乎都有的成员有：
  - （1）ClientID，控件在客户端的Id，控件在服务端的Id不一定等于客户端HTML中的Id，比如说在ListView等控件的模板中。因此如果要在客户端通过JavaScript Dom、JQuery的getElementById、$("#id")来操作控件的话最好不要直接写服务端Id，而是$('#<%=txt1.ClientID%>')。用JQuery事件设置鼠标移到控件上和从控件移开的不同样式。在用户控件中就可以看到ClientID和id的不同。
  - （2）Visible 属性， 控件是否可见，如果Visible=False是不会渲染到HTML中的，这和在HTML中给元素style.display='none'效果是不一样的。
  - （3）CssClass 属性，控件的样式名，就是HTML中控件的class属性。也可以单独修改BackColor、BorderStyle等属性，但是不建议这么做，因为会生成很多的内联样式，生成html尺寸大，不便于统一的修改。
  - （4）Attributes，用来设置获取控件的额外属性。和Dom中的setAttribute()、getAttribute()是一样的。Button1.Attributes["a1"] = "2.jpg";
    Button1.Attributes["onmouseover"] = "alert('hello')";

服务端控件的额外属性

- 说明：所有的服务端控件不仅可以使用控件定义的属性，还可以使用额外的属性，这些属性包括控件没有封装的HTML属性（比如onmouseover等浏览器端事件页当作属性），ASP.Net会将它不识别的属性原封不动的渲染到客户端。在代码中也可以通过Attributes属性设置额外属性：
  CheckBox1.Attributes["onmouseover"] = "alert('hello')";

ASP.Net服务端基本控件1

- 1、Label控件。Text属性为显示文本。AssociatedControlID属性用来关联一个控件，如果为空的话会展示为一个Span，如果指定为一个控件的id，则会展示为一个HTML中的<Label>并且将for属性设置为被关联控件的ClientId。
- 2、Literal控件也是展示一段文本，但是Literal控件不会渲染任何额外的标签，就是将Text属性的值展示出来而已。
- 3、TextBox控件，文本框控件。TextMode属性取值SingleLine、MultiLine、Password，分别渲染为input(type=text)、textarea和input(type=password)。当AutoPostBack属性为true的时候，用户焦点离开TextBox就会造成页面Post，实现原理就是讲ASP.Net原理时的AutoPostBack。TextChanged事件(需要AutoPostBack=true才会修改后触发，否则是页面提交才触发)，文本发生变化的时候事件触发。ASP.Net中要提交表单的时候最好调用__doPostBack方法。
- 4、RadioButton控件，渲染为input(type=radio)，通过GroupName属性进行分组

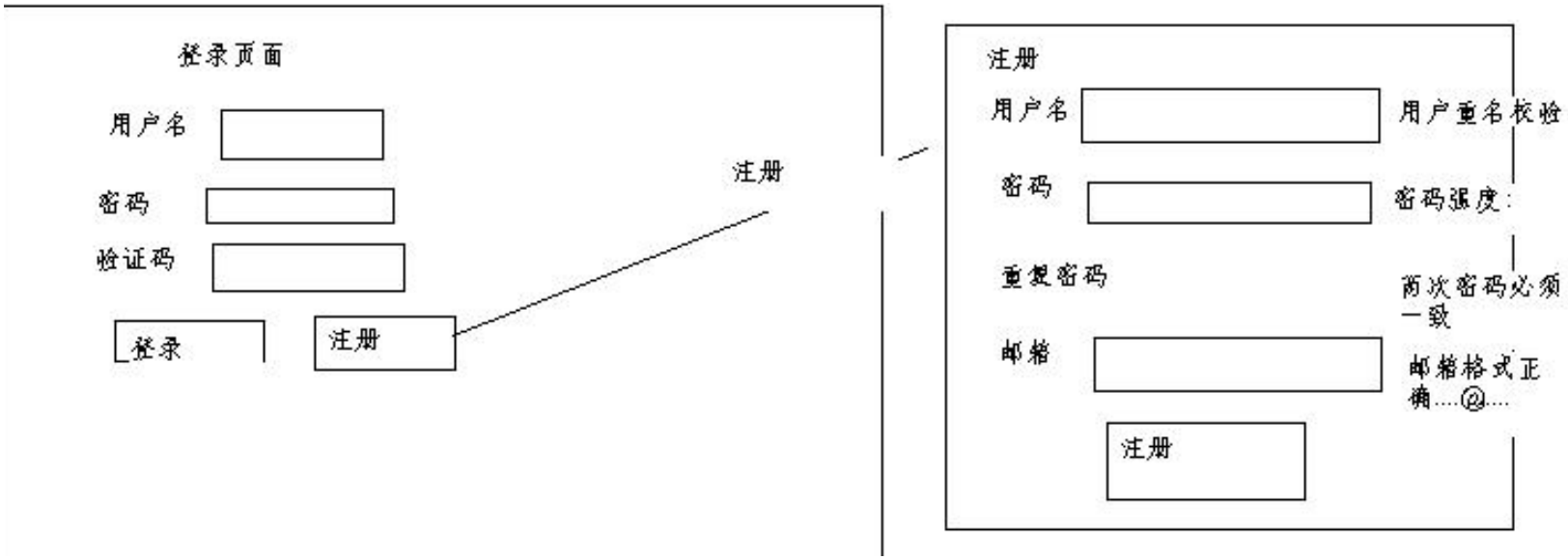
ASP.Net服务端基本控件 Button

- 5、Button控件。OnClientClick属性，当用户点击按钮的时候在浏览器端执行的代码，注意OnClientClick是字符串属性，写的代码是JavaScript代码，运行在浏览器端。<asp:Button ID="btnDel" runat="server" onclientclick="return confirm('真的要删除吗？')" Text="删除" />
- 6、LinkButton，用法和Button差不多，区别就是Button控件渲染为按钮，而LinkButton渲染为超链接。不要用LinkButton来实现普通的超链接，因为LinkButton的href为一段javascript代码，进行的是表单的Post，无法“在新窗口中打开连接”。和讲“行删除”那个例子中href为javascript的超链接原理一样。
- 7、ImageButton控件也和Button差不多，只不过是显示为图片，渲染为input(type=image)
- 8、Button、LinkButton、ImageButton等控件都有CommandName、CommandArgument两个属性和Command事件，可以让多个按钮控件共享一个Command事件处理函数，通过读取事件对象e的CommandName、CommandArgument两个属性读取被点击按钮上设置的这两个参数来执行不同的操作。例子：编辑、删除多行数据。这种用法在ListView等控件中用的最多。

ASP.Net服务端基本控件3

- 9、Panel控件用来盛放一些控件。如果设定GroupingText属性那么就渲染为含有<fieldset>的div标签，也就是GroupBox效果，否则渲染为
- 10、HyperLink控件，超链接。和LinkButton不一样(常考)，不会向服务器端Post，就是一个超链接。NavigateURL：链接地址；Text：显示文本。如果设定ImageUrl属性则会显示图片超链接。如果就是指向新浪网这种外部页面完全不用HyperLink， HyperLink主要的优势在于会自动将虚拟路径转换为客户端路径。
- 11、FileUpload控件，文件上传控件。渲染成input(type=file)。属性：FileContent以流形式获得上传的文件；FileName 上传文件名；HasFile Bool值，表示用户是否选择文件，SaveAs方法用于将文件保存到磁盘的指定位置。漏洞：文件上传漏洞（上传一个下载源代码的aspx、Process.Start启动格式化，创建管理员、开启远程桌面）。解决方法：只允许上传指定类型文件，上传文件夹不给执行权限。WebShell。
- 练习：用户注册（非空检查、Email合法性检查、用户名存在性检查），密码修改，用户登录，用户上传头像。页面流程见备注。加上防暴力破解。
- 练习：开发一个在线Excel阅读软件。用户上传一个Excel文件，服务端读取文件，手写拼table、tr、td标签，注意input(type=file)要将form的enctype="multipart/form-data"，支持多sheet，用JQuery来实现Sheet的切换。

用户注册登录


登录页面
用户名
密码
验证码
登录
注册
注册
用户名
用户重名校验
密码
密码强度:
重复密码
两次密码必须一致
邮箱
邮箱格式正确....@....
注册

三种控件

- HTML控件，ASP.Net把HTML控件当成普通字符串渲染到浏览器端，不去检查正确性、无法在服务器端进行处理。
- ASP.Net服务端控件，经过ASP.Net高度封装的控件，使用简单，运行在服务器端，可以在服务端使用C#代码进行操作，会渲染到客户端为HTML控件。
- runat=server的HTML控件。在HTML控件的基础上添加runat="server"，也是运行在服务器端的，也可以服务端使用C#代码进行操作，也会渲染到客户端，不像ASP.Net服务端控件那样高度封装，暴露的属性大部分是普通HTML属性。和ASP.Net服务端控件相比的好处是：当需要在服务器端要对控件进行操作的时候，如果控件没有被ASP.Net服务端控件封装的时候，用runat=server的HTML控件很方便，runat=server的HTML控件也会对虚拟路径、id→ClientID进行处理，所以在使用虚拟路径、UserControl中也可能会用到，试验在WebUserControl中使用id
  onmouseover="document.getElementById('<%=TextBox1.ClientID%>').value='哈哈';"
- 直接在属性中有问题，会把<%直接输出到浏览器端，因此不要在控件的属性值中写<%%>。

服务端HTML控件的类型

- a→HtmlAnchor；form→HtmlForm；head→HtmlHead；input→HtmlInputButton、HtmlInputCheckBox 、HtmlInputText等；meta→HtmlMeta；table→HtmlTable；tr→HtmlTableRow；td→HtmlTableCell；title→HtmlTitle。未单独封装的标签（比如div）或者自定义的标签（比如mmm）对应类型为HtmlGenericControl。使用Attributes属性操作未封装的属性。
- 不用单独记忆，忘了的话，在aspx中弄一个标签试验一下就行。
- 服务端HTML控件不像ASP.Net控件那样封装的高级，比如ASP.Net控件的BgColor属性为Color类型，而HTML控件的BgColor属性则为字符串类型，需要开发人员设置合法的值。

验证控件

- 必须要对用户输入的数据进行合法性校验，这些校验逻辑很多是重复的，比如字段不能为空、必须为日期格式、数字不能大于100等，而且要同时在客户端和服务器端校验，客户端校验提高可用性，服务端校验防止恶意攻击。ASP.Net验证控件就是为了简化这些问题而提供的。
- ASP.Net提供了如下的控件：
  - RequiredFieldValidator：字段必填；
  - RangeValidator：值在给定的最大值、最小值之间；
  - CompareValidator：用于比较两个值的关系是否满足要求或者是否是指定类型的数据；
  - RegularExpressionValidator：校验数据满足正则表达式；
  - CustomValidator：自定义验证。

RequiredFieldValidator

- ControlToValidate设定要验证的控件，Text属性为当被验证的控件属性为空的时候显示的错误信息，Text不仅可以写普通文本，可以写任何HTML内容。
- 有时候控件如果是默认值也认为是空值，比如下拉列表的选中值为“--选择性别--”、文本框的值为“填入搜索关键词”，只要将RequiredFieldValidator的InitialValue属性设定为“--选择性别--”、“填入搜索关键词”就可以。

Validator共性

- 页面中的IsValid属性用来判断页面中的所有Validator是否都校验通过，只有都校验通过才为True，即使页面中的Validator服务端校验报错（禁用JavaScript跳过客户端校验），在服务端方法中的业务代码（比如btn1_Click）也会被执行，因此如果代码在数据校验不通过的不能执行则需要判断IsValid的值。
- 所有验证控件都有Display属性，用来决定如何显示错误信息。三个值：Static：没有错误信息的时候控件的visiblity样式为hidden来实现隐藏；Dynamic：没有错误信息的时候控件的display样式为none来实现隐藏。这两者的区别是display:none和visiblity:hidden隐藏的区别是visiblity:hidden隐藏控件仍然会占据空间，而display:none隐藏则不会占空间。Display属性还可以设置为None（用来配合后面讲的ValidationSummary）
- 几乎所有控件都有ControlToValidate、Text属性，不再额外说。所有控件都几乎在客户端和服务器端都要进行校验。

Validator共性

- 如果在一个页面中同时放置注册和登录表单，那么他们的验证就会同时进行，这样虽然只是登录，但是注册的验证也会触发，可以使用验证组来解决这个问题，将同一组的控件（表单、提交按钮、Validator等）的ValidationGroup设为相同的值就可以，这样的话和Button等触发事件的控件的ValidationGroup 相同的控件才会校验。
- 如果将按钮控件（Button、ImageButton、LinkButton）的CausesValidation属性设置为false，则这个按钮的点击不触发校验。

RangeValidator

- RangeValidator：MinimumValue、MaximumValue为最大、最小值，Type属性为数据类型（String、Integer、Double、Date、Currency等）。例子：年龄、毕业日期在合理范围内。
- RangeValidator、CompareValidator、RegularExpressionValidator等都不会对非空值进行校验，所以如果字段不允许为空则需要再使用RequiredFieldValidator控件。
- MinimumValue、MaximumValue等当然也可以在运行时动态设置，比如设置最大日期为当前日期，DateTime.Now.ToShortDateString()。

CompareValidator

- CompareValidator：Type属性同RangeValidator。Operator属性，比较操作符，可选值DataTypeCheck、Equal、GreaterThan、GreaterThanEqul、LessThan、LessThanEqual、NotEqual。ValueToCompare，所比较的运算符右边的值。ControlToCompare，设定与另外一个控件进行比较。
- 例子，校验工资必须为整数，转正日期必须不能早于于入职日期
- 校验数据类型，不进行范围的比较：ControlToValidate设定要校验的控件，Type设置要检验的数据类型，Operator设定为DataTypeCheck。
- 要进行范围的比较：ControlToValidate设定要校验的控件，Type设置要检验的数据类型，Operator设定为GreaterThan，ValueToCompare为要比较的值。也可以编程的时候动态设置。
- 与另外一个控件的值进行比较，ControlToValidate设定要校验的控件，Type设置要检验的数据类型，Operator设定为GreaterThan，ControlToCompare设置为要比较的控件（运算符右边的控件）。
- 注意验证控件描述的是“正确条件”，不是“错误条件”。控件id一定要明显

高级Validator

- RegularExpressionValidator：ValidationExpression属性为正则表达式，VS的可视化编辑提供了几个内置的正则表达式，也可以自己写。案例，校验Email地址、身份证号码、QQ号码（5位至10位的数字“\d{5,10}”）、个人说明必须在10到50字之间
- CustomValidator，自定义验证控件。当ASP.Net内置的验证控件无法满足要求的时候可以使用CustomValidator。
- ServerValidate事件为服务端的校验代码，在事件处理函数中读取args.Value来获得待校验的值，如果值合法则将args.IsValid设置为true，否则设置为false。如果为ClientValidationFunction设定一个函数名，那么会首先在客户端调用指定的JavaScript函数进行客户端校验，否则将只做服务端校验。客户端校验函数的签名为“函数名(src,args)”，args的属性以及意义和服务端的一样。
- 练习：实现注册页面（写校验逻辑就行）：用户名（不能为空，长度必须在3—10之间），密码（不能为空，长度在3—9之间），再次输入密码（必须和密码一样），邮箱（选填，必须符合邮箱格式），出生日期，毕业日期（毕业日期要大于出生日期，小于当前日期），性别（“请选择性别”、男、女、保密，必填），月份、日（校验日期中日的数值在合法范围内，{1，3，5，7，8，10，12}是31天，{4，6，9，11}是30天，2月假设最大固定是29天不考虑闰年平年。，CustomValidator）。

汇总错误消息

- 使用Validator错误消息会显示在放置的位置，这样有两个可能的问题：如果表单非常大，用户看不到全部的错误消息，希望把错误消息集中显示在提交按钮旁边；如果错误信息非常多错误信息会散落各地，希望能集中显示。
- ValidationSummary控件用来集中显示错误消息。每个验证控件都有ErrorMessage、Text两个属性，ErrorMessage是用来显示到ValidationSummary中的值，Text是用来显示到Validator位置的值，如果Text为空，则ErrorMessage会同时显示到ValidationSummary和控件的位置。一般Text的值要简短（比如“必填”、“*”），因为直接显示到控件的位置，能知道指的是哪个控件；ErrorMessage要详细一些（比如“用户名不能为空”，不能是“这里不能为空”等），这样才能从ValidationSummary每条错误信息中读取值出来。
- ValidationSummary的属性：DiplayMode，显示模式，ShowMessageBox用户同时显示警告对话框。

母版（MasterPage）

- 网站的布局通常是统一的，上面是Logo、菜单条、下面是公司地址、版权声明等。如果每个页面都重复做这些功能的话：重复性劳动、一旦修改那么每个页面都要修改。可以使用FrameSet（框架集）技术来解决，但是FrameSet技术不灵活，而且很难进行SEO，所以只有部分内网系统还在用FrameSet，.Net中一般用母版（MasterPage）技术来解决这个问题。
- MasterPage是这样一种技术，把页面布局画好，在变化的内容部分“留空”，留空的部分由子页面填充内容，这样子页面只要填空就行，不用重复设计页面结构，一旦要修改页面结构修改母版页就可以，这样所有页面都会变化。母版页“挖坑”，具体页面“填坑”。母版页、具体页面中几乎可以使用所有的普通WebForm页面能够使用的技术。

母版2

- 添加一个“母版页”，使用<asp:ContentPlaceHolder>挖坑，新建的母版页已经自动设置了两个ContentPlaceHolder，还可以添加更多的ContentPlaceHolder。在id="head"前面后面都加入一些script，在id="ContentPlaceHolder1"前后也加入一些内容。
- 创建使用母版页的具体页面，WebSite是新建“Web窗体”的时候勾选“选择模板页”，WebApplication是新建“Web内容窗体”，然后选择页面使用的母版页（一个项目内可以创建多个母版页，比如新闻频道用一个母版页，视频频道用另外一个母版页）。
- 使用母版的具体页面和普通aspx页面的不同是：@Page区域用MasterPageFile属性表示页面使用哪个母版页，页面不包含html等内容，只定义了<asp:Content这些填坑的内容。<asp:Content>就是用来在具体页面中对母版页进行填坑的，ContentPlaceHolderID为这个Content要填母版页中的哪个坑，对应母版页中ContentPlaceHolder的Id

母版3

- 具体页面可以对母版页填坑也可以不填坑，如果不填坑则显示<asp:ContentPlaceHolder>中定义的默认内容。
- 案例：实现上导航菜单、下版权声明，左侧功能面板，右侧为具体的内容的模板，然后分别实现关于我们和登录界面。
- 在母版页中使用超链接、图片地址等的时候需要注意路径问题，在母版页中的runat=server控件的链接地址、图片地址等会被解析为相对于母版页的地址，而客户端HTML控件ASP.Net引擎是直接输出的，因此是解析为相对于具体页面的地址。建议使用服务端控件，如果不能使用服务器控件，那么可以在aspx页面中调用ResolveClientUrl 、ResolveUrl进行虚拟路径的转换
- 每个具体页设置不同的标题，只要在具体页面的@page中设置Title属性即可。可以在具体页中通过Master.FindControl来定位母版页中的控件然后对母版页中的控件进行操作，比如在一个具体页面中将母版页中的一个控件隐藏。

数据绑定控件简介

- 从数据绑定开始用WebApplication，否则会有很多麻烦。
- 数据绑定分为数据源和数据绑定控件两部分，数据绑定控件通过数据源来获得数据，通过数据源来隔离数据提供者和数据使用者，数据绑定控件通过数据源来对数据进行修改，数据源有SqlDataSource、AccessDataSource、ObjectDataSource、LinqDataSource、EntityDataSource、XmlDataSource等（SiteMapDataSource是SiteMap专用数据源），由于大部分项目都不会页面直连数据库（因为违反最基本的分层原则），所以SqlDataSource、AccessDataSource不会使用，LinqDataSource、EntityDataSource也是只有在很极端的采用Linq、EF的项目中才会用，XmlDataSource是处理XML数据是才可能会用。ObjectDataSource是Web开发中应用最广的数据源，也能很容易的进行数据库切换。
- 数据绑定控件有列表数据绑定控件（DropDownList、RadioButtonList、ListBox、CheckBoxList、BulletedList等）和复杂控件（DataGrid、GridView、DetailsView、FormView、ListView、Repeater、DataList等，GridView等都是ListView子集）。复杂控件中DataGrid已经不推荐使用，Repeater是最轻量级的组件，在互联网的前台用的最多，ListView是ASP.Net 3.5中新增的控件，ListView是GridView、DetailsView、FormView、Repeater、DataList等这些控件的大一统者，那些控件的优点ListView全都有，会了ListView那些控件也就会用了，因此数据绑定控件主要讲列表数据绑定控件、Repeater和ListView，项目中会用到FormView 、GridView。

ObjectDataSource

- ObjectDataSource用来将一个类做为数据源，TypeName属性为数据源类的全名，有DeleteMethod、InsertMethod、SelectMethod、UpdateMethod等几个属性，分别为类中删除、插入、查询、更新数据的方法名，这些方法可能有参数，参数的值是通过DeleteParameters、UpdateParameters、InsertParameters等嵌套节点设置的。
- 手工编写ObjectDataSource太麻烦，使用可视化界面来完成。将ObjectDataSource拖放到界面上，在右上角的智能标志上选择“配置数据源”即可进行配置。数据源类一般用数据集就可以，新建一个数据集，将表拖进来生成DataTable、Adapter等，生成完成后，在ObjectDataSource的“配置数据源”中就可以看到Adapter类，选中类，选择【下一步】，分别选择对应的获得、删除、更新、插入数据的方法。

列表绑定控件

- DropDownList显示来自于ObjectDataSource的数据，选择数据源（DataSourceID属性）为刚才的ObjectDataSource，并且设定显示字段（DataTextField）和值字段（DataValueField）即可。RadioButtonList、ListBox、CheckBoxList、BulletedList等也都是这么用。
- 手工设定绑定，除了可以给控件的DataSourceID属性设置一个数据源的方式进行数据绑定（推荐），还可以在代码中通过代码设置绑定（旧版本的ASP.Net只能这样绑定，新版本中不推荐）。
  - 用代码绑定可以将任何实现了IEnumerable接口的对象绑定到数据绑定控件。ListBox2.DataSource = new object[]{3,5,6};ListBox2.DataBind();
  - 由于数据绑定控件默认会将数据保存在ViewState中，因此不会每次刷新页面都会重新加载数据，只有第一次需要加载（!IsPostBack）
  - 由于代码绑定在禁用ViewState的情况下有很多麻烦事，因此推荐用DataSourceID的方式，控件会自己来判断是否应该重新取得数据。
- DropDownList原有“请选择性别”和数据绑定项的共存：AppendDataBoundItems="true"

复杂数据绑定控件

- 除了显示Text、Value这样简单的列表数据绑定控件之外，还有更复杂的数据绑定控件的要求，比如要将人员信息显示在界面上，包含姓名、年龄、照片等。这时候就要使用Repeater、ListView等控件。
- 学HTML的时候是手写表格，但是项目中很多数据不是固定的，而是动态的。可以用Dom动态增加表格行，但是数据仍然是固定的，我们需要从数据库等地方取得动态的数据来显示。比如网站的友情链接列表就不是固定的，而是从数据库中动态读取动态生成的。

Repeater

- Repeater(foreach)用于对绑定数据源中的数据进行遍历显示，每条数据以什么格式显示是由Repeater的<ItemTemplate>来决定的，模板会多次显示<ItemTemplate>姓名：<%#Eval("Name")%>年龄：<%#Eval("Age")%>
</ItemTemplate>
- <%#Eval("Name")%>表示在这个位置显示当前行的Name属性，注意调用Eval、Bind这些数据绑定方法的时候要用#。
- 因为Eval就是将属性显示到指定的位置，因此也可以显示到文本框中<ItemTemplate>姓名：<input type="text" value='<%#Eval("Name") %>' /></ItemTemplate>
- 注意不要写成value="<%#Eval('Name') %>" 因为<%%>中的是C#代码，"是字符，而不是字符串
- 还可以用在服务器控件中<asp:TextBox Text='<%#Eval("Name") %>' runat="server"></asp:TextBox>

Repeater其他模板

- 1、<AlternatingItemTemplate>，设置隔行的不同显示风格，如果设定<AlternatingItemTemplate>，则奇数行用<ItemTemplate>模板，偶数行用<AlternatingItemTemplate>模板
- <AlternatingItemTemplate><asp:TextBox BackColor="Red" ID="TextBox2" Text='<%#Eval("Name") %>' runat="server"/></AlternatingItemTemplate> 。设置隔行变色是为了防止数据太多看串行。
- 2、HeaderTemplate、FooterTemplate：头部、尾部的模板，分别显示在所有数据的前面和后面。
- 3、SeparatorTemplate ：两项数据之间的分隔符，比如换行符

案例

- 案例：显示人员信息，姓名、年龄、照片(网站中都是存的图片路径
- 案例：鼠标放到图片上，图片动态放大，其他还原。元素动画渐变：animate({ "width": "300", "height": "200" })
- 问题：在服务端控件中不能写'~/images/<%#Eval("PicPath")%>'，在页面中增加FormatImgURL方法，参数为object类型。对于服务端控件，只能ImageUrl='<%#Eval("PicPath") %>'，不能ImageUrl='images/<%#Eval("PicPath") %>'。能不用服务端控件就不用
- 练习：Table显示网站列表，字段：网站名、创立时间、创始人、网址。显示三列：网站名、创立时间、创始人，网站名展现为超链接
- 练习：显示友情链接列表，字段：网站名、超链接、友情链接类型（文本、图片）、Logo图片地址。 只显示文本友情链接。每行3个友情链接。div+css来布局，float:left;width:33%。li
- 在页面中分文本友情链接区和图片友情链接区两块。如何在ObjectDataSource中使用自定义的查询方法。

ItemDataBound

- 对于每行数据显示的时候都会调用ItemDataBound事件，在这个事件中可以对当前行进行处理，事件对象主要成员：
  - 1、e.Item.ItemType为当前行的类型，Item为ItemTemplate行、AlternatingItem为AlternatingItemTemplate行，还有Header、Footer等取值。
  - 2、ItemIndex当前行的序号
  - 3、DataItem当前行绑定的对象。
- 如果要在ItemDataBound事件中对ItemTemplate模板中的控件做处理，则必须使用runat=server的ASP.Net控件或者HTML控件，为控件设置Id，然后用FindControl根据Id来取得控件。注意在ASP.Net的模板中不能直接通过控件的Id来操作控件，必须用FindControl找到控件才能操作。案例：年龄大于30行的文本框变为红色；年龄大于30行的行变为红色。

ItemCommand

- 可以在模板中放置Button控件（Button、LinkButton、ImageButton），模板中的按钮一般不写OnClick事件响应，而是响应Repeater的ItemDataBound事件。
- 为Button控件设定CommandName、CommandArgument属性，然后在ItemDataBound事件读取e的CommandName、CommandArgument属性就可以获得发生事件的命令和行参数了。如果对数据进行了操作，则需要Repeater1.DataBind()来重新绑定，从数据库中刷新最新的数据。
- 案例：涨一岁，给被点击的行的年龄增加1。
- 练习：人员管理程序(Id、用户名、启用状态)，增加【禁用】当前行按钮，点击【禁用】点击的时候提示是否真的要禁用，如果确实要禁用则将“启用状态”字段设置为“禁用”。禁用字段显示为红色文字。字段变为禁用后，行显示【启用】按钮，点击【启用】按钮将“启用状态”字段设置为“启用”。即使没有设置AlternatingItemTemplate偶数行也被识别为ItemType=AlternatingItem
- 鼠标移动到一个人员行上的时候行背景颜色为黄色高亮显示。

ListView

- Repeater一般只用来展示数据，如果要增删改查则用ListView更方便。使用向导（强类型数据）来使用ListView会自动生成很多模板，免去手写模板代码的麻烦，再进行手工调整即可。
- 首先设定数据源，然后点击智能提示中的“配置ListView”，选择一种布局和样式，然后根据需要勾选“启用编辑”、“启用删除”、“启用插入”、“启用分页”，就会自动生成常用的模板。注意这只是提高开发效率的一个快捷方式，不是唯一的途径。
- LayoutTemplate为布局模板，布局模板中必须有一个ID为itemPlaceholder的服务端控件（4.0以后不需要），什么类型无所谓，不会被显示，itemPlaceholder前面就是相当于Repeater中的HeaderTemplate，itemPlaceholder后面就是相当于Repeater中的FooterTemplate，因此ListView中没有这两个模板。
- ItemTemplate是每一项的模板，AlternatingItemTemplate是隔行显示的模板，和Repeater一样。EmptyDataTemplate为数据源没有数据的时候显示的内容（Insert也算数据），这样的话可以实现“没有查找结果”、“对不起，找不到您要找的数据”等，InsertItemTemplate为插入数据界面的模板，EditItemTemplate为编辑数据的模板，InsertItemTemplate，为插入数据的模板，SelectedItemTemplate为标记为Selected的行的模板。
- 1、生成的样式要提到style中，不要内联样式。
- 2、ItemTemplate里面一般也没必要用<asp:Label展示只读数据，所以直接输出<%# Eval("Name") %>
- 3、LayoutTemplate中必须有一个id为itemPlaceholder的服务端控件，之上为头，之下为尾。
- 4、LayoutTemplate表头内容要汉化，所有Template中的不需要显示的字段，比如Id，都要删掉。

ListView

- EditItemTemplate、InsertItemTemplate中控件的绑定表达式为Text='<%# Bind("Age")，因为Eval只是计算表达式的值输出，而Bind不仅可以计算表达式的值输出，还可以将用户填入的值更新到数据中，因此Eval是单向绑定，Bind是双向绑定。
- 通过每行的Insert、Delete、Edit、Cancel等Command进行增删改，这几个CommandName被ListView内部处理，不需要开发人员处理，因此自定义的CommandName不要和他们重复。ListView中可以像Repeater那样为行增加Command按钮，处理方法和Repeater一样，ListView也支持Repeater那样的ItemDataBound事件。和Repeater的不同点：
  - 1、判断数据行的类型e.Item.ItemType == ListViewItemType.DataItem（编辑模板也放在这里）
  - 2、取得行对应的DataRowView：ListViewDataItem lvDataItem = (ListViewDataItem)e.Item;DataRowView rowView = (DataRowView)lvDataItem.DataItem;
  - 3、在FindControl的时候注意AlternatingItemTemplate的问题。
- ListView中可以使用Validator，只要将Validator放入相应的模板中，将Validator手动设为要验证的控件的Id，然后设定相应按钮、控件、Validator为同样的ValidationGroup，防止不同模板中的Validator互相干扰。将Cancel按钮的CausesValidation="false"

ListView

- 新增数据行的默认值：响应ListView的ItemCreated事件（每一行的在页面上的创建都会触发这个事件），当e.Item.ItemType为InsertItem的时候通过FindControl找到控件然后初始化。比如给年龄默认值。
- ObjectDataSource绑定Id为Guid类型的表的时候会生成一个“DataObjectTypeName="System.Guid" ”，有问题，删掉就行。
- 插入数据的初始化：注意和“新增数据行”不同，“插入数据的初始化”是在用户点击“插入按钮”之后执行。比如如果主键为Guid，则需要在数据插入数据库之前为主键赋值。响应ListView的ItemInserting事件（将一些插入数据库之前的对数据进行调整的代码），e.Values为所有字段的键值对，可以读取插入的值，也可以向字段中写值，这样就可以为Id赋值e.Values["Id"]=Guid.NewGuid()。在这个事件中对数据进行校验，可以通过e.Cancel = true来取消非法数据插入。
- 更新之前的处理：就像数据插入前可以在ItemInserting事件中处理一样，可以在ItemUpdating事件中对更新过程进行处理，e.ItemIndex可以取到当前更新行的行号，e.OldValues可以取到更新前的值，e.NewValues可以取到更新后的值，可以通过e.Cancel = true来取消非法数据插入。同理是ItemDeleting

DropDownList的绑定

- 补充，有的模板中可以绑定
- ListView中是无法像TextBox等控件那样将DropDownList的选中值绑定到数据的字段的，必须编程处理。例子，人员的性别（男、女、保密），三个值固定写在DropDownList中。
  - 1）在显示数据的时候DropDownList显示数据的值。在ItemTemplate中加入DropDownList，设定DropDownList Enabled="false"，这样就是只读的。在ItemDataBound事件中e.Item.FindControl()来找到DropDownList控件，然后ListViewDataItem lvDataItem = (ListViewDataItem)e.Item;
    DataRowView rowView = (DataRowView)lvDataItem.DataItem;取到DataRowView进而取到DataRow，读取数据的值，然后赋值给DropDownList的SelectedValue属性。也会同步的处理EditTemplete的展示，由于ListView的ItemType处理的不明确，因此需要判断绑定的DataItem数据、FindControl的值是否为空。
  - 2）在插入数据的时候设定DropDownList对应的字段的值，响应ItemInserting事件，通过e.Item.FindControl找到DropDownList控件，然后通过e.Values设定值
  - 3）在数据更新的时候设定DropDownList对应的字段的值，响应ItemUpdating事件，通过ListViewDataItem dataItem

案例：友情链接管理。

- 需求：后台提供友情链接增删改的页面。友情链接字段：序号、网站名、友情链接类型（文本超链接、图片超链接）、Logo地址、链接地址。要进行数据的非空校验等非法值校验：序号必填必须为整数，网站名必填、链接地址必填；当友情链接类型为文本超链接的时候Logo地址隐藏、当友情链接为图片超链接的时候Logo地址显示。
- 展示页面：在首页底部提供文本友情链接和Logo友情链接两个区域，没有Logo地址的显示在文本友情链接区，有Logo的显示在图片友情链接区；每行最多5个友情链接条目，超过则折行。

案例：入库单管理

- 需求：提供入库单明细的增删改查页面。字段：Id（Guid类型）、类型（可选值：采购入库、盘盈入库、退货入库，值来自于另外一张入库单类型表）、入库日期（默认为当天,ItemCreated，使用JQueryUI的datepicker控件）、单价、数量、金额。所有字段都不能为空，当用户输入单价或者数量之后自动计算金额（金额=单价*数量），考虑折扣因此金额可以不等于数量*单价，用户还可以修改计算以后的金额。如果数量为负值（红单），则此行显示为红色背景。不允许修改、删除。

行命令按钮

- ListView的行按钮和Repeater一样，不同的是取当前行数据的方式。int index = ((ListViewDataItem)e.Item).DisplayIndex取出操作行的行号，ListView1.DataKeys[index].Value取出主键的值，如果对数据进行了操作，最后要对ListView执行DataBind刷新数据。由ListView的DataKeyNames属性决定存储哪些字段值为主键，可以多个主键（和数据库主键没有直接关系），所以有Values。
- 排序：将LayoutTemplate中的表头用<asp:LinkButton runat="server" CommandName="Sort" Text="Id" CommandArgument="Id" />代替，其中CommandArgument的值为排序字段。只要是CommandName、CommandArgument对就行，展现成什么、显示在哪儿都可以。
- 案例：“涨一岁”按钮

综合练习：用户管理

- 字段：主键、用户名、用户类型（超级用户、系统管理员、用户）、创建日期、密码、是否禁用。密码以MD5值保存。密码不显示、密码不允许管理员修改，管理员创建的时候给初始密码888888（和主键为Guid类型的一样，在Inserting的时候e.Values["Password"]=GetMd5("888888")），创建日期默认为当前时间，可以修改。
- 增加【禁用】当前行按钮，点击【禁用】点击的时候提示是否真的要禁用，如果确实要禁用则将“禁用”字段设置为true。禁用字段显示为红色文字。
- 字段变为禁用后，行显示【启用】按钮，点击【启用】按钮将“禁用”字段设置为false。
- 增加重置密码按钮，点击重置密码按钮将用户的密码重置为888888。

DataPager

- ListView搭配DataPager控件实现分页。有两种使用方式，一种是将DataPager声明到ListView中；一种是DataPager、ListView没有嵌套关系，然后将DataPager的PagedControlID设定为要分页的ListView。没有什么区别，一般用“配置ListView”自动生成的内置方式即可。 DataPager的PageSize属性为一页的条数。
- （*）实现IPageableItemContainer 接口的控件都可以使用DataPager进行分页，但是ASP.Net内置的控件目前只有ListView实现了这个接口。
- DataPager中按钮显示风格由Fields中的字段设置，可以放置多个字段，分为“NextPreviousPagerField”（下一页、下一页、首页、末页等）、“NumericPagerField”（数字页号）、“TemplatePagerField”用模板自定义。代码中选择相应的Field，在属性视图中就可以快速修改它们的属性。

DataPager调整

- 相关单词：First：第一；Last：最后；Next：下一个；Previous：上一个。
- NextPreviousPagerField主要属性：ButtonCssClass：按钮的样式；ButtonType，按钮渲染成什么（Button按钮、Link超链接、Image图片）；FirstPageImageUrl，【第一页】按钮图片地址；FirstPageText，【第一页】按钮文本，这样可以实现上一页显示为“<”，最后一页显示为“>>|”这样的效果；ShowFirstButton，是否显示【第一页】，其他按钮也有对应属性。
- NumericPagerField主要属性：ButtonCount，最多数字的个数；按钮渲染成什么（Button、Link、Image）；CurrentPageLabelCssClass当前页文本的样式；NumericButtonCssClass数字按钮的样式。
- 如何实现|<</</页数/>/>>|这样的效果。顺序添加NextPreviousPagerField、NumericPagerField、NextPreviousPagerField，将第一个NextPreviousPagerField的First、 Previous设置为可见， Last、 Next设置为不可见，将最后一个NextPreviousPagerField的First、 Previous设置为不可见，Last、 Next设置为可见。
- （*）实现输入页面编号点击跳转或者在下拉列表中选择页数要用TemplatePagerField

高效率分页

- ListView默认的分页是先从数据源取得所有数据，然后再截取当前页面的部分，在数据量非常大的情况下效率非常低，因此默认分页基本不能用。应该是只从数据源取得要显示的数据。
- 复习：SQL中语句中取得分页数据。SQL语句中获得每一行序号的方法： SELECT Id, SiteName, LogoURL,Row_Number() over(order by Id) rownum FROM T_Links，Row_Number()函数是SQL2005之后提供的一个计算结果集行号的函数（不是表的行号），over中指定排序规则， Row_Number()从1开始。取得第11条至20数据（条数从0开始）的方法，使用子查询用行号进行再次处理：
- Select * from (SELECT Id, SiteName, LogoURL,Row_Number() over(order by Id) rownum FROM T_Links) t where t.rownum>11 and t.rownum<=20。

高效率分页

- 在强类型DataSet中增加取得所有数据条数的方法QueryCount，增加取得指定行数范围数据的方法GetPagedData：select * from
- (SELECT ......,Row_Number() over(order by ...) rownum FROM T_Links) t where t.rownum>@startRowIndex and t.rownum<=@startRowIndex+@maximumRows。
- 由于数据集编辑器不支持（不是运行时不支持，只是设计器不会自动帮我们生成一些东西） Row_Number()，所以创建完成后需要手动在GetPagedData属性的Parameters中增加两个参数：startRowIndex、maximumRows（参数名必须是这两个，这是由ObjectDataSource的StartRowIndexParamterName、MaximumRowsParamterName确定的，一般不需要改。），都是Int32类型。

高效率分页

- ObjectDataSource中EnablePaging属性设置为true，SelectCountMethod设置为QueryCount，SelectMethod设置为GetPagedData。
- 如果出错的话看看是不是没有放置内置的DataPager或者外置的DataPager的PagedControlID没有指向ListView。
- 先按照正常的流程配置ObjectDataSource，让ListView自动生成Template，再修改ObjectDataSource的EnablePaging="True"，SelectCountMethod设置为取得行数的方法。
- DataPager默认是用PostBack机制，显示不到地址中，不利于网友间共享，只要指定QueryStringField属性（比如pagenum）就可以实现超链接形式的分页链接。

- ListViewEdit.aspx?Id=22&action=edit
- 编辑(action=edit)、新增(action=addnew)、查看(action=view)。
- 字段非常多，ListView只能展示关键性字段，其他字段也在ListViewEdit.aspx来查看。
- 新增
- <tr><td>编辑查看</td><td>tom</td></tr>
- InsertItemPosition="None"不显示插入模板，这样也能显示出来“EmptyTemplate”中的值。
- FormView用来进行单条的编辑、查看、新增，有编辑、查看、新增三个模板。
- 使用 FormView1.ChangeMode(FormViewMode.Insert)切换显示模式
- 强类型DataSet中增加GetDataById方法，然后配置ObjectDataSource，让Select方法使用GetDataById方法，传递参数传递的是Id参数的值，参数源“QueryString”，QueryStringField：id。

单独页面编辑

- ListView的在位编辑只适合字段比较少、比较简单的场合，复杂数据的编辑、插入、查看等要在单独页面中。
- 创建一个单独的页面Edit***.aspx，然后在ListView页面中的编辑放一个编辑的超链接，向Edit***.aspx传递?id=1&action=edit。页面顶端增加一个Edit***.aspx?action=addnew的超链接。
- 使用FormView控件进行单条数据的编辑，在Page_Load中判断action，然后使用FormView1.ChangeMode方法切换FormView的模式。
- 强类型DataSet中增加一个GetDataById方法，在ObjectDataSource中选择这个方法为Select参数，参数源为QueryString，QueryStringField为id。
- 在元素插入、修改完成（Inserted、Updated事件）后重定向到列表页面。

- 1、删掉Id，需要在Inserting为Id赋值(e.Values["Id"] = Guid.NewGuid())
- 2、插入完成、更新完成以后重定向到ListUI。ItemUpdated、ItemInserted中Redirect
- FormView中的DropDownList的绑定，思路和Listview一样。把InType改造成DropDownList。
- JQueryUI的js 在JQuery之后引入
- 响应FormView的ItemCreated事件，用FormView.CurrentMode判断当前渲染的模板（因为FormView同时只能渲染一个模板，所以不需要像ListView那样e.Item.ItemType）。然后用FormView.FindControl找到控件（也不像Listview中用e.Item.FindControl）

CKEditor 集成

- CKEditor原名FckEditor，著名的HTML编辑器，可以在线编辑HTML内容，演示一下。打开。自己人用CKEditor，网友用UBBEditor。
- 配置参考文档，主要将ckeditor中的（adapters、images、lang、plugins、skins、themes、ckeditor.js、config.js、contents.css）解压到js目录，然后“显示所有文件”，将ckeditor的目录“包含在项目中”，在发帖页面引用ckeditor.js，然后设置多行文本框的class="ckeditor"（CSS强大）(服务端控件CssClas=" ckeditor "，客户端控件要设定cols、rows属性，一般不直接用html控件)，代码中仍然可以通过TextBox控件的Text属性来访问编辑器内容。
- 由于页面提交的时候asp.net会把富文本编辑器中的html内容当成攻击内容，因此需要在aspx中的Page标签中设置 ValidateRequest="false" 来禁用攻击检测（2010中还要根据报错信息修改WebConfig来禁用XSS检测）。

CKFinder集成

- CKFinder是一个CKEditor插件，用来为CKEditor提供文件的上传的功能。将bin\Release下的CKFinder.dll添加到项目的引用；将core、ckfinder.js、ckfinder.html、config.ascx解压到CKFinder自己的目录。按照文档修改CKEditor的config.js，将上传的处理程序设定为CKFinder，注意路径的问题。
- 使用测试，在插入超链接、插入图片、插入文件中都有“上传”
- 因为上传文件是非常危险的动作，因此在文件上传的时候会进行权限校验。在config.ascx的CheckAuthentication方法中校验是否有权限上传，返回true表示有权限，否则没有权限，一般修改成判断用户是否登录，并且登录用户是有上传权限的用户，可以用Session或者Membership来做。思考：如何实现只有指定IP地址的用户才能上传？
- 在SetConfig函数中设置上传文件夹的位置BaseUrl、缩略图的位置，每种类型数据的上传路径、允许上传的文件类型AllowedExtensions等。

练习点评：入库单管理

- 需求：提供入库单明细的增删改查页面。字段：Id（Guid类型）、类型（可选值：采购入库、盘盈入库、退货入库，值来自于另外一张入库单类型表）、入库日期（默认为当天,ItemCreated，使用JQueryUI的datepicker控件）、单价、数量、金额。所有字段都不能为空，当用户输入单价或者数量之后自动计算金额（金额=单价*数量），考虑折扣因此金额可以不等于数量*单价，用户还可以修改计算以后的金额。如果数量为负值（红单），则此行显示为红色背景。不允许修改、删除。
- ListView在2010的Bug。用myid的方法解决。
- 3.5中插入有问题。

练习点评：用户管理

- 字段：主键、用户名、用户类型（超级用户、系统管理员、用户）、创建日期、密码、是否禁用。密码以MD5值保存。密码不显示、密码不允许管理员修改，管理员创建的时候给初始密码888888（和主键为Guid类型的一样，在Inserting的时候e.Values["Password"]=GetMd5("888888")），创建日期默认为当前时间，可以修改。
- 增加【禁用】当前行按钮，点击【禁用】点击的时候提示是否真的要禁用，如果确实要禁用则将“禁用”字段设置为true。禁用字段显示为红色文字。
- 字段变为禁用后，行显示【启用】按钮，点击【启用】按钮将“禁用”字段设置为false。
- 增加重置密码按钮，点击重置密码按钮将用户的密码重置为888888。

综合案例：企业网站

- 母版：上导航（鼠标放到的导航按钮动态变大）、下版权声明
- 分为前台和后台两块，后台包括管理员管理及各个模块的管理，用户管理自己写。Insert、Update完成后Redirect到列表界面。
- 新闻：发布日期（JQueryUI的datepicker控件）、标题、内容。列表要分页（用DataPager）。使用CKEditor做编辑器。在单独的编辑界面编辑新闻。
- 看新闻功能：页面名ViewNews.aspx，帖子Id通过URL传递过来 Id=3。传入参数的错误处理（帖子不存在等）。禁用ViewState。查看列表也要页面分页。
- 招聘信息：岗位名称、发布日期、截止日期、需求人数、岗位要求。在单独的编辑界面编辑产品。展示页面的分页有页码跳转。
- 产品展示页面（鼠标高亮显示当前行，点击产品图片在层中动态显示大图。字段：产品名、产品图片、产品类别（干果、果汁、保健品等））。产品类别管理。不同类别分别展示。在单独的编辑界面编辑产品。WebUserControl，传递一个CatId来决定显示不同类型中的数据，在控件的Page_Load中用代码进行数据绑定。
- 友情链接
- 后台管理界面左侧是QQTab风格的导航，两组内容、系统，系统下设修改密码、用户管理。后台登陆要验证码（用这个代码）。

禁用ViewState

- 默认情况下ASP.Net是启用ViewState的，这样在页面中会生成冗长的隐藏字段，ViewState对于需要PostBack处理的页面才可能有用，对于新闻展示页面不需要交互完全没必要用ViewState
- 禁用ViewState的方式：
  - 页面整体禁用ViewState：在顶部Page中EnableViewState="False"
  - 指定控件禁用ViewState，在控件上EnableViewState="False"
- 页面禁用ViewState以后并没有完全去掉ViewState，只要ViewState不是很大就可以了。如果要求一点儿ViewState都不能有，那么则页面中不能有runat=server的form，如果页面中没有表单元素，把form完全去掉就可以。如果Button等服务端控件没有放到runat=server的form中，那么则是不可用的。

缓存（Cache）

- 如果每次进入页面的时候都查询数据库生成页面内容的话，如果访问量非常大，则网站性能会非常差。而如果只有第一次访问的时候才查询数据库生成页面内容，以后都直接输出内容，则能提高系统性能。这样无论有多少人访问都只访问一次数据库，数据库压力不变。
- 缓存是一种用空间换取时间的技术，存在于计算机中很多地方，用来将一些慢速设备中的常用数据保存在快速设备中，取数据的时候直接从快速设备中取。比如CPU二级缓存、windows文件读取缓存。
- 缓存存在失效的问题：为了保证从缓存中读取数据和慢速数据中数据一致，则需要在慢速数据中对应的数据发生变化的时候，清除缓存中相应的数据。
- 缓存是改进网站性能的第一个手段，就像索引是改进数据库性能的第一个手段一样。ASP.net缓存主要分为：页面缓存、数据源缓存、数据缓存这三种主要类型。

页面缓存

- 给页面添加<%@ OutputCache Duration="15" VaryByParam="none"%>标签就可以启用页面缓存，这样整个页面的内容都会被缓存，页面中的ASP.Net代码、数据源在缓存期间都不会被运行，而是直接输出缓存的页面内容。 Duration表示缓存时间，以秒为单位，超过这个时间则缓存失效，再次生成以后会再缓存15秒，以此类推。在Page_Load处设置断点、修改数据库数据测试。
- 缓存是针对所有这个页面的访问者。这样1个访问者和1万个访问者、一次访问和100万次访问对数据库的压力是一样的。
- 对于看新闻页面来讲，如果如上设置的话，则会缓存在第一个看到的新闻，因为?id=2、?id=3只是页面的不同参数而已，为了能让不同的新闻各自缓存，因此可以设置VaryByParam="id"，表示对于不同的id参数进行单独缓存。如果有多个确定缓存的参数，则将参数名用分号隔开即可，比如VaryByParam="id,number"。测试。
- 如果想让任何不同的查询字符串都创建不同的缓存，则设置VaryByParam="*"，一般情况下设置“*”就足够。
- 在WebUserControl中也可以像页面缓存一样设置控件的缓存。

数据源缓存

- 设定ObjectDataSource的CacheDuration（缓存时间：秒），EnableCaching=true。这样每隔CacheDuration指定的时间段才调用SelectMethod指定的方法来执行数据库查询，其他时候都是直接返回缓存的数据。
- 缓存固定的时间适用于首页、文章列表等访问频繁的页面，对于看贴页面则不适合，假设有100万个帖子，如果每个帖子都是固定缓存1小时的话，假设一小时之内有10万个帖子被看了，那么就要缓存十万个帖子，非常占用内存，因为“百年一看”的“坟帖”偶然被访问一次也缓存一个小时，占用内存。这时候可以采用“滑动窗口（sliding）”策略，比如帖子缓存10分钟，如果10分钟之内又访问了，则缓存的失效时间修改为从被访问这一刻起的10分钟之后，以此类推。这样经常访问的帖子就可以“长期缓存”，而不经常访问的帖子也不会因为偶然访问而长期占用缓存。设置方法，数据源：CacheExpirationPolicy="Sliding"。面试可聊。todo: 貌似滑动有问题。不是问题，Sliding只是策略，服务器会参考。

缓存其他

- 页面缓存、数据源缓存等内部都是使用HttpRuntime.Cache来实现缓存的，在一些页面缓存、数据源缓存完成不了的特殊的缓存要求中，可以直接调用HttpRuntime.Cache进行缓存。在如鹏网项目中会讲到。
- （*）ASP.Net缓存默认是保存在内存中的，还可以配置保存到数据库中。大型网站还会配合使用Memcached等技术。
- 清除缓存。在缓存还未失效的时候可能需要立即清空缓存，让数据库的修改立即反映到界面中。ASP.Net没有提供现成的方法，可以使用Hack级别的代码，打开。

错误页

- 当页面发生错误的时候，ASP.Net会将错误信息展示出来，这样一来不好看，二来会泄露网站的内部实现信息，给网站带来安全隐患，因此需要定制错误页，发生错误时显示开发人员定制的页面。404页面放点广告也是好的嘛。
- 配置web.config，配置customErrors区域：
  - <customErrors mode="On" defaultRedirect="MyErrorPage.aspx">
  - <error statusCode="403" redirect="NoAccess.htm" />
  - <error statusCode="404" redirect="FileNotFound.htm" />
  - </customErrors>
- mode三个可选值：On：总是显示定制错误页面；Off：不显示定制错误界面，直接显示调用堆栈等异常信息；remoteonly：对于本机的访问显示调用堆栈等异常信息，对于外部用户的显示定制错误页面。一般设置为RemoteOnly，这样发生错误的话，管理员可以在服务器的浏览器中看详细错误信息，普通用户看不到。学习演示的时候mode设置为On，否则看不到定制页。可以在定义错误页中判断Request.UserHostAddress来设置某些ip看到异常信息，可以读取Session如果是管理员则可以看异常信息。

错误页2

- error子元素设定对于不同的状态码使用不同的错误页，很多网站都把404做一个特殊的错误页。没有单独设置的状态码错误则显示defaultRedirect中指定的页面。
- 错误页即可以使用htm页面，也可以使用aspx页面。在aspx页面中可以用HttpContext.Current.Server.GetLastError()拿到异常对象。一般不要把异常信息显示给用户，而是使用后面讲的Log4Net等将异常记录到系统日志。如果要在错误页面中拿到异常对象，比如customErrors中设置redirectMode="ResponseRewrite"，因为默认是客户端重定向，在错误页面中就拿不到异常对象了。

AJAX简介

- 没有AJAX会怎么样？普通的ASP.Net每次执行服务端方法的时候都要刷新当前页面，比如实现显示服务器的时间。每次都要刷新页面的坏处：页面刷新打断用户操作、速度慢、增加服务器的流量压力。如果没有AJAX，在youku看视频的过程中如果点击了“顶、踩”、评论、评论翻页，页面就会刷新，视频就会被打断。
- AJAX（Asynchronous JavaScript and XML，异步JavaScript和XML）是一种进行页面局部异步刷新的技术。用AJAX向服务器发送请求和获得服务器返回的数据并且更新到界面中，不是整个页面刷新，而是在HTML页面中使用JavaScript创建XMLHTTPRequest对象来向服务器发出请求以及获得返回的数据，就像JavaScript版的WebClient一样，在页面中由XMLHTTPRequest来发出Http请求和获得服务器的返回数据，这样页面就不会刷新了。 XMLHTTPRequest是AJAX的核心对象

XMLHTTPRequest

- 开发一个AJAX功能需要开发服务端和客户端两块程序。以一个显示服务端时间为例。首先开发一个GetDate1.ashx，输出当前时间。在HTML页面中放一个按钮，在按钮的onclick中创建XMLHTTP向GetDate1.ashx发送请求，获得返回的数据并且显示到界面上。代码见备注。面试常考：不使用UpdatePanel、JQuery等AJAX库编写一个AJAX程序。
- 也可以在xmlhttp.open中向服务器传递参数：xmlhttp.open("POST", "GetDate1.ashx?id=1", false)，如果传递给服务器的请求里有中文，则需要使用Javascript函数encodeURI来进行URL编码。
- 案例：用AJAX实现汇率转换，页面放一个文本框（输入人民币）、一个下拉列表（美元，日元，港币）、一个按钮，点击按钮在一个span中显示转换后的金额。汇率计算在服务器端完成，假设汇率（人民币/外币）：7、0.1、0.9。

JQuery AJAX

- new ActiveXObject("Microsoft.XMLHTTP")是IE中创建XMLHttpRequest对象的方法。非IE浏览器中创建方法是new XmlHttpRequest()。为了兼容不同的浏览器需要编写很多代码，见备注。
- 回调函数中data参数为服务器返回的数据， textStatus为服务器返回状态码， textStatus为"success"表示成功。
- JQuery中提供了简化ajax使用的方法。$.ajax()函数是JQuery中提供的ajax访问函数，$.post()是对$.ajax()的post方式提交ajax查询的封装，$.get()是对$.ajax()的get方式提交ajax查询的封装。推荐用post方式，因为post方式没有缓存的问题，演示，get方式中缓存处理的处理方法。代码见备注。todo：好像服务端异常404、500等回调方法并不会被调用。
- 如果有请求参数则在第二个参数用字典方式来设置。$.post("GetDate1.ashx", {"id":"2"},function(data, textStatus) {。JQuery帮我们进行了URL编码，因此参数中有中文也不用担心。
- 案例：JQuery AJAX版的汇率转换。注意status指的是通讯是否成功

案例练习

- 案例：新闻的无刷新评论，防止打断视频播放。刚进入界面的时候评论也是页面显示后才加载，“正在加载评论”。界面上有一个表格，逐条显示评论，页面刚进来的时候显示已有评论。用户在评论文本框中输入文本，点击评论按钮，向服务器发出ajax请求，将用户的评论内容发给服务器，如果服务器返回“插入成功”的消息则将用户的评论动态添加到现有表格中，如果用户评论中含有“TMD”、“去死”等不良信息则提示用户“请文明用语”。
- 练习：完成上述内容，并且实现评论“看到时”才加载，参考
- 练习1：AJAX校验用户名是否存在，焦点离开用户名、点击【检查用户名】的校验。分别用XMLHttp和JQueryAJAX实现。
- 练习2：填入商品名称，AJAX自动带出价格填充到价格框。

Json

- AJAX传递复杂数据如果自己进行格式定义的话会经历组装、解析的过程，因此AJAX中有一个事实上的数据传输标准JSon。Json将复杂对象序列化为一个字符串，在浏览器端再将字符串反序列化为JavaScript可以读取的对象。看一下Json的格式。Json被几乎所有语言支持。
- C#中将.Net对象序列化为Json字符串的方法：JavaScriptSerializer().Serialize(p)， JavaScriptSerializer在System.Web.Extensions.dll中，是.Net3.x 中新增的类，如果在.Net2.0中则需要用第三方的组件。
- JQuery AJAX得到的data是Json格式数据，用$.parseJSON(data)方法将JSon格式数据解析为JavaScript对象
- 可以在post函数最后一个函数传递"json"则data就是反序列化以后的对象，免去了parseJSON

案例练习

- 案例练习：无刷新分页。AJAX比较难的应用。
- 案例练习：无刷新删除行，配合JQueryUI中的Effect实现动态删除效果，要考虑删除失败的情况。
- 练习：鼠标放到图片上，AJAX方式去后台取图片相关信息并且展示出来。
- 练习1：新闻的无刷新评论用方式Json实现，要分页。评论实现“看到才加载”。
- 练习2：实现一个看帖页面，页面中有“赞成”、“反对”两个按钮，点击“赞成”按钮则增加赞成数，并且在“赞成”按钮上显示最新的赞成个数，“反对”按钮同样如此。（*）同一个IP在24小时之内只能投票一次。讨论方案。select * from ** where ip="....." and datediff(hour,投票时间,getdate())<24 如果条数大于0就说明24小时之内投过票了。
- 练习3：每隔十秒钟刷新一次数据库中的最新评论。

微软AJAX解决方案（*）

- ASP.Net中内置的简化AJAX开发的控件UpdatePanel
  - 放入ScriptManager，将要实现AJAX效果的控件放到UpdatePanel中即可。
  - UpdatePanel原理探秘，用HttpWatch看。
  - 只把需要无刷新更新的部分放到UpdatePanel中。
  - UpdatePanel用来实现一些对性能要求不高的需求非常方便。
- Timer实现定时AJAX效果，原理分析。
- UpdateProgress显示“正在加载数据”。
- AJAX Toolkit简介。

WCF简化AJAX（*）

- 开发步骤：
  - 添加一个Web项目，在Web项目中新建“新建项”→“Web”→“启用了AJAX的WCF服务”（2008中在“C#”根类别下）
  - 页面上拖放ScriptManager控件，ScriptManager的Services属性中新增一项，Path属性设置为服务路径，比如“~/Service1.svc”
  - 调用服务端方法的时候 Service1.DoWork(OnDoWorkSucceed,OnDoWorkFailed), Service1为服务类名，DoWork为方法名，OnDoWorkSucceed是调用成功时被回调的函数（Javascript中的委托），OnDoWorkFailed是调用失败时被回调的函数。两个函数都是有一个参数result的，成功函数的result值为函数返回值，失败函数的result值为错误消息。调用都是异步的。 注意这是Javascript代码！
- 注意“~Service1.svc"要加在ScriptManager的Services属性中，而不是Scripts属性中。如果写Javascript的时候没有自动提示，把aspx关掉再打开就行。如果还不行的话手动写。
- 服务端还可以返回复杂对象，JS端可以直接从result读取复杂对象的字段值。

全局文件

- 添加Web→全局应用程序类，注意文件名不要改。
  - 全局文件是对Web应用声明周期的一个事件响应的地方
  - 将Web应用启动时初始化的一些代码写到Application_Start中，比如后面讲的Log4Net的初始化等。应用关闭的时候Application_End调用。
  - 当一个Session启动的时候Session_Start被调用，Session结束（用户主动退出或者超时结束）Session_End被调用。
  - 当一个用户请求来的时候Application_BeginRequest方法被调用
  - 当应用中出现未捕获异常，Application_Error被调用（常考，ASP.Net中的错误处理机制），用HttpContext.Current.Server.GetLastError()获得异常信息，然后用Log4Net记录到日志中。
- 案例练习：实现图片的防盗链，讨论。
- 案例练习：屏蔽指定的IP地址。

URL重写(UrlRewrite)

- 为什么要URL重写？1、有利于SEO，带参数的URL权重较低；2、地址看起来更正规，推广uid。看看如鹏网的URL重写。
- 伪静态：看起来像普通页面，而非动态生成的页面。
- 原理：在Global.asax的Application_BeginRequest 中读取Request.Url 得到请求的URL（View-3.aspx），然后用HttpContext.Current.RewritePath(ReWriteUrl)进行重写（也就是交由另外一个页面处理这个请求）(View.aspx?tid=3格式)
  http://www.cnblogs.com/hd/archive/2005/06/20/177633.html
- 也可以使用微软的URLRewrite，只要修改配置文件就可以进行URL重写。照着文档自学配置。见备注

IIS配置及防黑

- 安装IIS。部署网站（发布或者拷贝都可以）。修改连接字符串，compilation设为false，删掉cs代码
- 上传文件夹不给执行权限： 在iis管理器中找到上传文件夹，选择属性--执行权限，设置为“无”。这样哪怕利用漏洞上传了可执行代码到上传文件夹，也无法执行。
- 取消所有文件夹的浏览权限，防止用户查看网站的文件列表，在iis管理器中找到主站节点→属性→主目录→取消“目录浏览”。
- 后台文件夹只允许管理员的IP访问，文件夹→属性→
- IIS管理中，Web服务器扩展，只允许asp.net那几个，其他的CGI、ASP等全部禁止。

嵌入Web资源的方法（*）

- 可以将js、图片、css等嵌入Assembly中，这样就不用将文件在aspx中写了，特别适合做自定义控件的时候将控件用到的资源打包。
  - 将文件放到项目的合适路径，比如jpg文件所在路径的namespace为RuPengSite.Ctrls.Test， jpeg文件名为My. jpg
  - 在属性视图中设置My. jpg文件的“生成操作”为“嵌入的资源”（会编译到）
  - AssemblyInfo.cs或者项目中任意一个类的namespace上方加入：[assembly: System.Web.UI.WebResource(" RuPengSite.Ctrls.Test. My. jpg ", "image/jpeg")] 。
  - 取得资源路径的方法是ClientScriptManager.GetWebResourceUrl即可，第一个参数为资源所在Assembly的一个类的类名，第二个为RuPengSite.Ctrls.Test. My. jpg 。
- 如果在页面要需要引用嵌入的js，可以ClientScript.RegisterClientScriptResource(typeof(MyClass), " RuPengSite.Ctrls.Test. My.js"); 即可。
- 嵌入Web资源本质论。axd，就是运行时将dll中的资源显示出来

- json
- setInterval和AJAX实现ServerPush
- AJAX的后退按钮
- AJAX的跨域问题。JSonP（JQuery第191页）