第一章

.NET 的构成

.NET = 新平台 + 标准协议 + 统一开发工具。

- 公共语言运行时(CLR)

  > 整个.NET Framework 的构建基础。负责整个.NET 执行时的代码管理，内存，线程，安全，强制安全类型检查管理。

- 服务框架(Services Framework)

  > 基类库 数据访问类库，开发工具使用的类

- 两类应用模板

  > 传统的WIndows 应用程序模板 Win Forms
  > 基于ASP.NET的面向Web的应用程序模板

ASP.NET 建立在 公共语言运行时之上。是.NET框架的一部分

编译器和集成开发环境

- 集成开发环境: Visual Studio
- 编译器csc.exe： 语法 csc 3.cs

跨语言的操作性

ASP.NET 支持多种编程语言

- C#
- VB
- J#
- 带有托管功能的C++
- JS.NET

类库

在服务框架里

托管代码

.NET 跨平台

垃圾回收 GC

CLR

.NET 使用中间语言的作用

- ASP.NET 将C# 代码翻译成 Microsoft中间语言 (MSIL) 的，与编程语言和CPU 无关的表现形式，并储存在多个程序集中(.dll 文件)。
- MSIL 将运行在.NET 框架的上下文中，CLR的 JIT(Just In Time)编译器会将MSIL翻译成CPU特定的指令
- 服务器会缓存编译的结果

第二章

动态网页编程技术

- 静态网页 后缀 html htm xml shtml
- 动态网页 asp aspx jsp php perl cgi

> 使用语言 HTML + ASP/PHP/JSP

- 超文本标记语言(HTML)

> 版本信息
>
> > XML HTML
> > 说明性标题: title meta data
> > 文档主体

- XHTML

> 标记和属性必须小写 特殊符号要用编码表示 属性都要赋值，没有就重复本身

- web 表单
  ``
  ``
- ASP.NET 开发模式

> web forms
>
> > 整合了 HTML 服务器控件和代码
> > 在服务器上编译和运行，再生成HTML发送给客户端

> MVC
>
> > Model 实现数据逻辑操作
> > View 用于显示应用程序的用户界面

> Controller 作为中间件，获取Model 数据给 View

> Core
>
> > 跨平台

- 指令

  > @page 定义ASP.NET 的页特定属性，只用于 .aspx
  > @Import 将命名空见导入

- 服务器元素必须加 runat="server" 如 form

IIS 的虚拟目录

- / 服务器根目录
- ./ 当前目录
- 当前目录
- ~/ 根目录
- ../ 上一级目录

C#

- 无类库 使用.NET 的类库
- 类型检查和异常处理也是使用CLR 处理
- 有GC

C# 的基类 数据类型的基类

C# 的继承

- 构造函数没有类型
- 继承时，子类的构造函数也要同时构造父类

> a(int x,int y):base(x,y){}

- base 用于从派生类访问基类成员

C# 的命名空间 作用

- 解决类名冲突的问题
- using name

引用类型和值类型

- 值类型

> 简单类型
>
> > 整数
> > 布尔 只能为true false
> > 字符
> > 实数 decimal float double

> 结构类型 不支持继承

> 枚举类型 enum 默认为整数元素 第一个为0

- 引用类型 储存变量的地址

类类型

数组类型

> 一维数组 int[] a = new int[]
> {1,2,3}/int[2]/int[3]{1,2,3}
> 二维数组 int[,] b={{1,2},{3,4}}
> 二维数组 int[,] b= new int[3,4]

委托类型

> 定义一种变量来指代一个函数或者一个方法
> delegate int a(); a b= new a(c);
> 事件的声明通过委托实现 先定义委托再委托
> 定义事件。触发时教案的过程实际是调用委托
> delegate void myevent (object
> sender,EventArgs)
> event myevent a

接口类型

> 不能定义数据只能定义方法，属性和事件
> 不定义具体实现，而是在接口的继承类中实现

二位数组定义

- 二维数组 int[,] b={{1,2},{3,4}}
- 二维数组 int[,] b= new int[3,4]
- foreach(int i in a){}

try catch finally 执行

- try 运行语句
- catch(Exception e) 捕捉异常

> OverflowException

> ArgumentNUllException

- finally 最终执行
- throw(e) 抛出异常

装箱 拆箱

- 给Object 对象赋值的过程叫装箱，反之拆箱
- 使用强制类型转换 拆箱 int j = (int) o

类型强制转换

- TO.String() 转换为string 类型 23.ToString()
- Parse() : Int32.Parse("1234")
- Convert.ToString(DateTime.Now)

引用类型转换为string

其他

- static 使用 类名.成员名访问 也可以使用静态方法访问
- readonly 运行时只读，内容在运行时指定
- abstract 不能创建实例
- virtual 修饰方法 可以有实现
- const 必须编译时指明

- 重载 名称一样，但是参数不一样。返回值要一样
- 多态 基类为接口
- 类方法前加上 virtual 修饰符 就不能加别的修饰符号
- 基类的普通函数在派生类重新定义要加new，虚函数加override
- 将派生类赋值给基类时， 调用普通方法时，总调用基类。调用虚方法时，调用派生类
- 抽象类不能直接实例化只能继承。继承类必须重写抽象方法

第四章

获取UID 方法

首次加载 page_load 的方法

- 执行顺序

> Page_PreInit

> Page_Init

> Page_Load

> 控件事件

- 可以通过Page.IsPostBack 来判断是否执行
- 可以在事件发给服务器之前做处理，类似click事件

回发 ispostback autopostback

- 在Page_PreInit 中。
- 第一次浏览网页 返回false 否则 true

- autopostback 默认为false, 当为true 时按Tab自动回发到服务器

属性绑定字段

- ToolTip 鼠标悬停文本
- Text 基本的文本
- OnTextChanged() 文本改变使用
- OnClick() 点击事件
- NavigateUrl URl 字符串转跳的界面

Dropdownlist

- <asp:Listitem>
-

checkboxlist

-
- OnCheckedChanged() 点击时触发
- Checked 是否被选中
-

用户自定义控件

- 扩展名为 .ascx
- @page 换成 @Control
- html 元素都删除

注册语句

```
<%@ Register TagPrefix="myCo" TagName="myTxt" Src="4-33.ascx"%>
```

验证控件

服务端和用户端区别

- 服务端 指将用户输入的信息发送到web服务器验证
- 客户端 是利用JS 脚本验证
- 页面生成前会自动检测是否支持JavaScript 支持就在客户端验证

验证时机

- XXX Validator
- EnableClientScript 是否使用客户端验证
- 默认 为True 当界面改变属性ControlToValidate指定控件的值并将焦点移出时，就会产生客户端验证
- 如果为 false 只有当页面有往返时，才会实现验证

可以放几个控件

其他

- 事件定义

> protected void Page_Load(object sender,EventArgs e)

- 动态数据绑定

`<%# databinding expression %>`

第五章

Respond

- Write() / = 输出数据
- End() 将缓冲区的信息输出，并终止执行
- WriteFile() 输出文件内容
- Redirect() 转向其他URL
- Clear() 清除缓冲区 前提 Buffer 为True
- Flush() 缓冲区信息输出 前提 ~

Request 值的获取方法

- Form 获取 Post 数据
- QueryString 获取Get 数据
- Request() 都可以
- ContentEncoding.EncodingName 获得编码方式
- QueryString()

> 使用? 与URL 间隔 不同变量使用&间隔
>
> ```
> Request["user"]
> ```

- Request.ServerVariables["环境变量名"]
- Request.Browser["浏览器特性名"]
-

Application 放的东西

多用户共享对象，生命周期到网站关闭

- Application["a"]="b"
- 要加锁来避免 竞争。

> Application.Lock() 和 Application.Unlock()

Cookie Session

- Session["a"] = "b"

> Session.Timeout 有效期
> Session.Abandon()失效
>
> HttpCookie a = new HttpCookie("user")
> a.Value = "b"
> Response.Cookie.Add(a)
> Request.Cookie["user"].Value

- Session 保存在服务器
- Cookie 保存在客户端浏览器

Server

重定向

- Redirect() 在服务器执行但重定向发生在客户端，可以在浏览器地址栏看到地址变化 可以定向到其他网站，使用查询字符串传递数据
- Execute()和Transfer 重定向发生在服务端，地址看不到变化只能定向到同一个网站的不同页
- Redirect()和Transfer()不会返回原网页,Execute()会

MapPath

- 转化为服务器上的物理地址

html uml 编解码

- HtmlEncode("") 将字符串转换为HTML 编码

- UrlEncode() 将特殊字符转化为URL编码，处理连接地址

配置文件

global

- 可选的
- 主要提供开始和清除代码以及设置整体的参数
- 网站开始时自动调用

web.config

- 和machine 一个目录 继承它所有设置