关于Web.config 文件：

using System.Web.Configuration;

using System.Configuration;

读取：WebConfigurationManager

――――――――――――――――――――――――――――――――――――――

两个主要的属性：也就是 Section: AppSettings ; ConnectionStrings

两个内建的直接使用的Section:

WebConfigurationManager.AppSettings

WebConfigurationManager.ConnectionStrings

s += string.Format("<br>Appsetings:{0}-{1}",

WebConfigurationManager.AppSettings.Keys[0],

WebConfigurationManager.AppSettings[0]);

s += string.Format("<br>Connect:{0}-{1}-{2}",

WebConfigurationManager.ConnectionStrings[0].Name,

WebConfigurationManager.ConnectionStrings[0].ConnectionString,

WebConfigurationManager.ConnectionStrings[0].ProviderName);

// 只读，不能如此赋值， 将出错：

WebConfigurationManager.ConnectionStrings[0].Name = "vivian";

WebConfigurationManager.ConnectionStrings[0].ConnectionString = "connect string";

<appSettings>

<add key="fname" value="Vancouver William" />

<add key="lname" value="Lilian Liu" />

<add key="apname1" value="Van William" />

<add key="appName2" value="App Namespace 2" />

</appSettings>

<connectionStrings>

<remove name="LocalSqlServer" />

<add name="goodName"

connectionString="mydoc/myaccess.mdx"

providerName="ms access" />

<add name="ApplicationServices" connectionString="data source=.\SQLEXPRESS;Integrated Security=SSPI;AttachDBFilename=|DataDirectory|\aspnetdb.mdf;User Instance=true"

providerName="System.Data.SqlClient" />

<add name="lwhProvider" connectionString="server=local;uid=sa;pwd=sa123"

providerName="mysql" />

</connectionStrings>

读取其他的Section: 必须是Web.config 认知的。而且要显式转化类型才可以使用。

方法： WebConfigurationManager.GetSection（path ）

―――――――――――――――――――――――――――――――――――――――――――

<system.web>

<compilation debug="true" targetFramework="4.0" />

<membership>

<providers>

<clear/>

<add name="AspNetSqlMembershipProvider"

type="System.Web.Security.SqlMembershipProvider"

connectionStringName="ApplicationServices"

enablePasswordRetrieval="false"

enablePasswordReset="true"

requiresQuestionAndAnswer="false"

requiresUniqueEmail="false"

maxInvalidPasswordAttempts="5"

minRequiredPasswordLength="6"

minRequiredNonalphanumericCharacters="0"

passwordAttemptWindow="10"

applicationName="/"

/>

</providers>

</membership>

</system.web>

MembershipSection ms = (MembershipSection)

WebConfigurationManager.GetSection("system.web/membership");

s += string.Format("<br>Provider:{0} val:{1}",

ms.Providers[0].Name,

ms.Providers[0].Parameters["passwordAttemptWindow"]);

CompilationSection cps = (CompilationSection)

WebConfigurationManager.GetSection("system.web/compilation");

s += string.Format("<br>Compilation:{0}-{1}", cps.TargetFramework, cps.MaxBatchSize);

注意： 1） 类的名称都有约定， 即首字母大写， 后面加 XxxxSection

2） 不同的类有不同的属性：

如何编程改写web.config：

―――――――――――――――――――――――――――――――――――――――

<appSettings>

<add key="fname" value="Women" />

<add key="lname" value="Lilian Liu" />

<add key="apname1" value="Van William" />

<add key="appName2" value="App Namespace 2" />

</appSettings>

<connectionStrings>

<add name="ApplicationServices"

connectionString="data source=.\SQLEXPRESS;Integrated Security=SSPI;AttachDBFilename=|DataDirectory|\aspnetdb.mdf;User Instance=true"

providerName="System.Data.SqlClient" />

</connectionStrings>

Configuration ob = WebConfigurationManager.OpenWebConfiguration(Request.ApplicationPath);

//修改

ob.AppSettings.Settings["fname"].Value = "Vancouver William";

// 修改，变成了新增

ob.ConnectionStrings.ConnectionStrings[0].ProviderName = "ms access";

ob.ConnectionStrings.ConnectionStrings[0].Name = "goodName";

ob.ConnectionStrings.ConnectionStrings[0].ConnectionString= "mydoc/myaccess.mdx";

//新增

ConnectionStringSettings css = new ConnectionStringSettings();

css.Name = "lwhProvider";

css.ProviderName = "mysql";

css.ConnectionString = "server=local;uid=sa;pwd=sa123";

ob.ConnectionStrings.ConnectionStrings.Add(css);

ob.Save(ConfigurationSaveMode.Minimal);

<appSettings>

<add key="fname" value="Vancouver William" />

</appSettings>

<connectionStrings>

<remove name="LocalSqlServer" />

<add name="goodName" connectionString="mydoc/myaccess.mdx" providerName="ms access" />

<add name="ApplicationServices" connectionString="data source=.\SQLEXPRESS;Integrated Security=SSPI;AttachDBFilename=|DataDirectory|\aspnetdb.mdf;User Instance=true"

providerName="System.Data.SqlClient" />

<add name="lwhProvider" connectionString="server=local;uid=sa;pwd=sa123"

providerName="mysql" />

</connectionStrings>

如何创建自定义的Section:

――――――――――――――――――――――――――――――――――――――――――――――――

第一步： 定义类， 继承自：ConfigurationSection 嵌套成员继承自：ConfigurationElement

using System.Web.Configuration;

using System.Configuration;

namespace webapp1.classme

{

public class myclass : ConfigurationSection

{

[ConfigurationProperty("book",IsRequired=true, IsKey=true)]

public string book

{

get

{

return (string)base["book"];

}

set

{

base["book"] = value;

}

}

[ConfigurationProperty("price",IsRequired=false)]

public decimal price

{

get { return (decimal)base["price"]; }

set { base["price"] = value; }

}

[ConfigurationProperty("auts")]

public aut auts {

get{

return (aut)base["auts"];

}

set {

base["auts"] = value;

}

}

}

public class aut : ConfigurationElement

{

[ConfigurationProperty("aname")]

public string aname

{

get { return (string)base["aname"]; }

set { base["aname"] = value; }

}

}

}

第二步： web.config 里注册类：

<configSections>

<section name="myCust" type="**webapp1.classme.myclass**" /> //指定 Type

</configSections>

<myCust book="Disney Land" price="922.99">

<auts aname="william liu" />

</myCust>

第三步： 程序中如何使用：

myclass me = (myclass)WebConfigurationManager.GetSection("myCust");

s += string.Format("<br>mycust:{0}-{1}-{2}<br>", me.book, me.price, me.auts.aname);

注意：名称的映射关系

[ConfigurationProperty("book", IsRequired=true, IsKey=true)]

public string book1

{

get { return (string)base["book"]; }

set { base["book"] = value; }

}

book1 对应编程时的对象： me.book1

myclass me = (myclass)WebConfigurationManager.GetSection("myCust");

[ConfigurationProperty("book")] = base["book"] = web.config attributes

如何创建集合：

―――――――――――――――――――――――――――――――――――――――――――――――――

第一步： 集合项目的类：ConfigurationElement

public class subj : ConfigurationElement

{

[ConfigurationProperty("sub", IsRequired = true, IsKey=true)]

public string sub

{

get { return (string)base["sub"]; }

set { base["sub"] = value; }

}

[ConfigurationProperty("score")]

public decimal score

{

get { return (decimal)base["score"]; }

set { base["score"] = value; }

}

}

第二步： 创建集合：ConfigurationElementCollection

public class subjCollection : ConfigurationElementCollection

{

public subj this[int idx]

{

get {

return base.BaseGet(idx) as subj;

}

set {

if (base.BaseGet(idx) != null)

{

base.BaseRemoveAt(idx);

}

base.BaseAdd(idx, value);

}

}

protected override ConfigurationElement CreateNewElement()

{

return new subj();

}

protected override object GetElementKey(ConfigurationElement element)

{

return ((subj)element).sub;

}

}

注意： 必须实现两个抽象方法：

CreateNewElement 和 GetElementKey

第三步： 在ConfigurationSection 里定义：public subjCollection subject

public class myclass : ConfigurationSection

{

……其他同上例子…………………………………..

[ConfigurationProperty("subject")]

public subjCollection subject

{

get { return this["subject"] as subjCollection; }

set { this["subject"] = value; }

}

}

第四步： web.config 里定义和添加项目：

<configuration>

<configSections>

<section name="myCust" type="webapp1.classme.myclass" />

</configSections>

<myCust book="Disney Land" price="922.99">

<auts aname="william liu" />

<subject>

<add sub="math" score="98" />

<add sub="Art" score="100" />

<add sub="English" score="99" />

</subject>

</myCust>

</configuration>

注意： subject 集合： 需要使用 <add> 来添加集合项目：

第五步： 程序里如何使用：

string s = null;

myclass me = (myclass)WebConfigurationManager.GetSection("myCust");

s += string.Format("<br>mycust:{0}-{1}-{2}<br>", me.book1, me.price1, me.auts.aname);

s += string.Format("<br>Subject:{0}", me.subject.Count);

foreach (subj el in me.subject) // 遍历集合元素

{

s += string.Format("<br>Item: {0}-{1}-{2}", el.sub, el.score, el.score);

}

输出如下：

mycust: Disney Land - 922.99 - william liu  
  
Subject:3  
Item: math-98-98  
Item: Art-100-100  
Item: English-99-99

web.config 的加密和解密：

――――――――――――――――――――――――――――――――――――――――――――――

Configuration conf = WebConfigurationManager.OpenWebConfiguration(Request.ApplicationPath);

ConfigurationSection confs = conf.GetSection("myCust");

if (!confs.SectionInformation.IsProtected)

{

confs.SectionInformation.ProtectSection("DataProtectionConfigurationProvider");

}

else

{

confs.SectionInformation.UnprotectSection();

}

conf.Save();

//不管加密过的，还是没有加密的， 读取没有任何问题：

string s = null;

myclass me = (myclass)WebConfigurationManager.GetSection("myCust");

s += string.Format("<br>mycust:{0}-{1}-{2}<br>", me.book1, me.price1, me.auts.aname);

s += string.Format("<br>Subject:{0}", me.subject.Count);

foreach (subj el in me.subject)

{

s += string.Format("<br>Item: {0}-{1}-{2}", el.sub, el.score, el.score);

}

非常简单：

1. 获取可以读写操作的 Configuration 对象
2. 获取需要加密或者解密的 Section: GetSection
3. 判断是否是已经加密： SectionInformation.IsProtected
4. 加密： SectionInformation.ProtectSection("DataProtectionConfigurationProvider");
5. 解密： SectionInformation.UnprotectSection();

可以对现有已经加密的web.config 再对其他未加密的Section 加密：

Configuration conf = WebConfigurationManager.OpenWebConfiguration(Request.ApplicationPath);

ConfigurationSection confs = conf.GetSection("appSettings");

if (!confs.SectionInformation.IsProtected)

{

confs.SectionInformation.ProtectSection("DataProtectionConfigurationProvider");

}

else

{

confs.SectionInformation.UnprotectSection();

}

conf.Save();

<myCust configProtectionProvider="DataProtectionConfigurationProvider">

<EncryptedData>

<CipherData>

<CipherValue></CipherValue>

</CipherData>

</EncryptedData>

</myCust>

<appSettings configProtectionProvider="DataProtectionConfigurationProvider">

<EncryptedData>

<CipherData>

<CipherValue></CipherValue>

</CipherData>

</EncryptedData>

</appSettings>

加密后的web.config