

CodeSmith 代码自动生成利器

文 曲贵川 (沈阳理工大学)

摘要: 为了满足大型数据库应用系统开发中快速开发的要求, 缩短软件开发周期, 提高代码可维护性, 在软件系统开发的过程中, 采用层次化的软件设计方法, 根据接口规范, 可以设计出代码生成工具。文章讨论了生成工具所需要的数据库信息的获取与处理、代码文档模型和控件模型等关键性技术和架构, 并介绍了在工程实践中使用代码生成工具 CodeSmith 进行系统开发所达到的快速开发和良好的适应性、扩展性的效果。

关键词: 代码生成工具; CodeSmith 数据库应用; 次化; 软件设计快速软件开发

一、引言

所谓代码生成技术, 即在开发应用程序之前, 先设计一个程序, 用它来自动生成目标程序的代码。随着应用程序开发环境的日益复杂, 代码生成技术在专业应用程序开发过程中的地位越来越重要了。使用代码生成技术, 都可以极大地加快应用程序的编程速度, 降低错误发生率, 从而提高应用程序的生产效率。

二、代码生成技术的优点

现代企业应用程序一般采用三层或多层模式, 从下向上依次为数据库访问层、应用逻辑层和用户界面层。一个典型的在 Microsoft .NET 平台上开发的应用软件的模式为: 数据库访问层: 使用 MS SQL Server 2000 作为数据库管理系统, 库中包含有由数十到数百个表组成的数据库。与数据有关的底层处理由对应的一组 Transaction SQL 存储过程 (Stored Procedure) 完成。应用逻辑层: 由一组 .NET 组件 (.dll) 构成, 是实现应用程序的业务逻辑部分。应用逻辑层通过 ADO.NET 与数据库层交换数据。如果用用户界面层使用在客户端运行的 UI 程序, 那么应用逻辑层通常用 Web Service 实现。用户界面层: .NET 平台上的用户界面可以采用两种方式: Web Form 和 Windows Form。前者在服务器上运行, 用户通过自己计算机上的浏览器访问; 后者是运行于客户终端计算机上的 Windows 应用程序, 通常是通过 Web Service 与服务层上的应用逻辑层数据库交换数据。对于开发一个有 100 - 200 个数据库表的中型数据库应用软件开发来说, 主要的编程工作量集中

在以下几个方面:

(一) 对各数据库表的基本操作如添加、删除、修改和查询编制相应的存储过程;

(二) 对每一个数据库表建立相应的组件, 编写相应的类 (Class) 代码以封装对数据库的操作。在类代码中, 通过 ADO.NET 调用数据库的存储过程来实现相应的数据库操作;

(三) 应用逻辑的实现;

(四) Web Service 代码;

(五) 用户界面的实现, 其中又可分为对数据库的维护部分和应用逻辑的用户接口部分;

实际上, 上述 1, 2, 4 和 5 的部分工

作基本上是机械重复的, 有标准方法可循。例如, 对所有的表来说, 插入、删除和修改一个记录的算法都是一样的, 所不同的只是每个具体表的字段参数。在这种情况下, 根据精心设计的模板自动生成相应的程序模块不但可以提高编程效率, 而且还大大降低了调试工作量。如果在调试中发现了某个错误, 这时只需修改相应的模板后再次生成相应的代码即可。即使对于上述的业务逻辑模块以及用户界面设计而言, 自动代码生成也是有价值的。商业代码生成软件通常可提供大量程序模板供选择, 在代码生成之后, 应用程序开发人员可以根据实际情况对其进行修改, 这比用手方式编码能够节约大量的时间。

基于上面的种种优点, Codesmith 便是你不二的选择。Codesmith 是专门针对 .NET 开发人员开发的一款基于模板的代码生成工具, 它允许你产生任何继续 ASCII 的代码, 除了可以生成 C#, VB.NET, 你还可以生成 java, 甚至存储过程。代码生成可以通过属性来控制, 属性可以是任何 .NET 对象, 也可以是自定义对象如 CodeSmith 内置的 TableSchema 对象, 当然对象的情况是可以拥有自己的属性设计器, 总而言之。CodeSmith 是允许用户自定义扩展的工具。CodeSmith 的模板是以 ".cst" 为后缀名的代码文件, 其风格非常类似于 ASP.NET。模板中除了大量固定的代码以外, 还有遍布于各处的元数据 (metadata)。固定代码就是会被 CodeSmith 原封不动地输出的代码, 而元数据则可以被看作是模板中的变量。那么现在我们就通过一个简单的模板文件来了解 CodeSmith 的工作原理:

```
1<%@ Template Language="C#"
TargetLanguage="C#"
Description="Demonstrates the most basic
template." %>
2< % @      Property
Name="SampleBooleanProperty"
Type="System.Boolean" Default="True"
Category="Options" %>
3< % @      Property
Name="SampleStringProperty"
Type="System.String"
Default="SampleString"
Category="Options"%>
Public class SimpleTemplate
{
    Public SimpleTemplate:base() { };
    <% if(SimpleBooleaTemplate) { %>
    {
        Private void
        <%=SampleStringSample%> ()
        {
            //do something
        }
    }
    %> }
```

如果您了解 ASP.NET 的语法, 那么就很好理解 CodeSmith 模板文件了。第一

条语句: 是模板文件的声明语句, 每个模板文件都必须以此作为开头, 它包含了必要的声明信息, 如模板语言、目标语言以及模板描述。除必需的 "Language" 属性以外, CodeSmith 还提供可供模板声明使用的属性:

Description: 作为模板的描述性信息并可以在该模板的 ToolTip 中显示;

Inherits: 尽管默认情况下所有模板都继承自 CodeSmith.Engine.CodeTemplate, 但仍然可以由您来指定模板继承自其它类 (只要该类直接或间接地继承自 CodeSmith.Engine.CodeTemplate);

Src: 该属性使您能够将其它类的功能添加进当前模板, 您能够用这个属性来实现 "代码隐藏" 模型;

Debug: 当该属性被设置为 True 时, 您就可以使用 System.Diagnostics.Debugger.Break 方法来为您的模板设置断点;

OutputType: 该属性用于指定以 "Normal"、"Trace" 或 "None" 方式输出模板;

LinePragmas: 当该属性被设置为 True 时, 模板错误将指向源代码; 设置为 False 时, 模板错误将指向编译后的代码;

模板中的代码可以分为三个部分: 静态、系统生成和动态的。静态代码就是系统将原封不动地输出的代码, 例如前面代码示例中的: public SimpleTemplate(): base()

CodeSmith 可以自动生成部分代码, 例如示例中的: // This class generated by CodeSmith on <%= DateTime.Now.ToLongDateString() %>. CodeSmith 将 <%= %> 标记中间的部分作为计算表达式, 在运行时计算该表达式并将结果输出到代码中相应的位置。CodeSmith 中的属性用以实现用户自定义的输出内容, 例如示例中的: < % @ Property Name="SampleStringProperty" Type="System.String" Default="SampleString" Category="Options" Description="This is a sample string property." %> 属性具有名称和类型, 这是必需的内容, 另外还可以给属性设置一个默认值, 添加一些描述性说明以便于用户在使用该属性时能够迅速地解其用途和用法。使用属性时仍然会用到 <%= %> 标签, 在该标签中插入需要的属性名即可, 例如: private void <%= SampleStringProperty %>() 尽管模板文件的内容似乎要比所需要的源代码复杂得多, 但这是一劳久逸的工作, 因为您可以用模板生成无数代码, 当初编辑模板时花费在编写动态内容及属性上的辛苦付出将在日后使用模板时得到丰厚的汇报!

目前 CodeSmith V4.1 已经很好地集成进 Visual Studio 2005, 这样一来, 程序员就能够更加方便地使用它了。那么, 就让我们来看看, CodeSmith 还为我们带来了哪些振奋人心的新特性吧:

CodeSmith Project: CSP (下转 181 页)

asp.net 网站开发中常用的加密技术

文 王丽娟 吴东明 (焦作师范高等专科学校)

摘要: 本文基于 asp.net 开发环境, 详细介绍了在网站开发过程中常用的对网页中的数据进行加密的技术。

关键词: asp.net; RSA; javascript

进入信息和网络化的时代以来, 网络正在我们的工作和生活中扮演着日益重要的角色。越来越多的用户通过网络来获取信息、处理信息。通过网页获取自己需要的信息已成为广大网络用户的主要手段。为了确保网站的信息安全, 避免自己写出来的客户端 IE 运行的代码被别人轻易的拷贝以及更好的对网站进行维护管理。灵活的使用网页中的数据加密技术对网站的内容进行保护, 也越来越受青睐。本文就 asp.net web 应用过程中常用的数据加密技术进行探讨。

一、我们可以使用 javascript 脚本对网页进行加密

使用 javascript 加密最简单的结果就是让浏览者不能使用鼠标右键或者是禁止用户查看源文件。可以通过以下方法对网页进行简单保护

(1) 利用弹出窗口封锁鼠标右键。将下面这段代码放在网页 HTML 代码的 < head > < /head > 标志中, 就可以实现封锁右键给网页加密。

```
<script language="javascript">
function click() {if (event.button==2)
{alert('本站不准使用右键')}}
document.onmousedown=click < /script >
```

(2) 禁止查看源文件。将下面这段代码放在网页的 HTML 代码的 < head > < /head > 标志中则可以实现禁止利用 IE 浏览器查看源文件。

```
<script language="JavaScript">
document.onmousedown=click
function click() {
if ( event.button==2) {alert('不准查看源文件')}
if ( event.button==3) {alert('不准查看源文件')}
}< /script >
```

二、我们可以使用 .Net Framework 中的类提供的加密服务来保证数据安全
可以使用 RSA 算法进行加密。

在 .Net Framework 中与 RSA 加密算法相关的类主要有两个: RSA 类和

(上接 180 页) 的出现使得您无论在 Visual Studio 2005、MSBuild、Windows Explorer、命令行、批处理文件或是 CodeSmith 自身环境下都可以以轻松、一致的方式自动地生成代码;

ActiveSnippets: 比 Visual Studio 2005 的 Snippet 更加强大, 可以通过执行逻辑处理或是访问复杂的原数据来控制代码片段的输出;

CodeSmith Maps: 创建字典式的映射, 例如 SQL 与 C# 的数据类型映射;

.netTiers 2.0: 增强的 .netTiers 模板;

Extended Property Management: 可以在 CodeSmith Studio 中编辑或者添加模

RSACryptoServiceProvider 类。RSA 类是“表示 RSA 算法的所有实现均从中继承的基类”, 而 RSACryptoServiceProvider 类是“使用加密服务提供程序 (CSP) 提供的 RSA 算法的实现执行不对称加密和解密”。另外, “表示 RSA 算法的标准参数”的 RSAPParameters 结构也是很重要的, 它保存了 RSA 算法的参数。下面着重介绍一下如何在 ASP .Net 中实现 RSA 加密。

RSA 参数的产生: RSA 参数的类型就是上面提到的 RSAPParameters 结构, 包含了八个字段。加密时只需要 Exponent 和 Modulus 两个值, 可看成公钥。解密时所有字段都需要, 可看成私钥。下面这段程序显示了如何产生 RSA 两个参数:

```
RSACryptoServiceProvider rsa = new
RSACryptoServiceProvider();
```

```
RSAPParameters
rsaParamsExcludePrivate=rsa.ExportParameters
(false);
```

```
RSAPParameters
rsaParamsIncludePrivate=rsa.ExportParameters
(true);
```

RSACryptoServiceProvider 类的 ExportParameters(bool) 方法用于导出 RSA 参数, true 表示导出上述八个字段的“私钥”, false 表示导出“公钥”

使用 RSA 参数进行加密: 这一步需要把上面两个参数导入到 RSACryptoServiceProvider 类对象中, 再用它对数据进行加密。如下面的代码所示, 我们可以写一个函数来完成加密过程:

```
Public byte [] RSAEncrypt ( byte [] b)
{RSACryptoServiceProvider rsa = new
RSACryptoServiceProvider();
rsa.ImportParameters
(rsaParamsExcludePrivate); // 导入公钥
byte [] EncryptedData=rsa.Encrypt
(DataToEncrypt, false);
return EncryptedData;}
```

三、使用 MD5 和 SHA1 两种加密方法

MD5 的全称是 Message-Digest Algorithm 5 (信息 - 摘要算法), 它的作用是让大容量信息在用数字签名软件签署私人密钥前被“压缩”成一种保密的格式 (把一个任意长度的字节串变换成一定长的大整数)。需要获

式扩展属性;

Property Persistence: 现在 CodeSmith 可以记住您最近一次执行模板时的属性值;

CodeSmith Studio: 改进了智能感应并全面提升性能;

如果您对 CodeSmith 提供的模板感到不太满意, 或者您需要根据自己的需求来制作相应的模板, 则可以使用 CodeSmith Studio 来新建或编辑现有代码模板

其实有很多代码都是不需要我们自己去从头到尾亲自写一遍的, 尤其是在工作中, 为什么不能用更少的时间完成更多的事情呢? 程序员从事的是脑力劳动, 我们

得一个随机长度的信息并产生一个 128 位的信息摘要。MD5 使用到的相关类:

```
System.Security.Cryptography.MD5
System.Security.Cryptography.
MD5CryptoServiceProvider() System.Web.
Security.FormsAuthentication.
HashPasswordForStoringInConfigFile(strSource,
"MD5")
```

SHA1 的全称是 Secure Hash Algorithm (安全哈希算法), 加密哈希函数将任意长度的二进制字符串映射为固定长度的小型二进制字符串。加密哈希函数有这样一个属性: 在计算上不大可能找到散列为相同的值的两个不同的输入; 也就是说, 两组数据的哈希值仅在对应的数据也匹配时才会匹配。数据的少量更改会在哈希值中产生不可预知的大量更改。所以您很难从加密后的文字中找到蛛丝马迹。SHA1 相关类:

```
System.Security.Cryptography.SHA1
System.Security.Cryptography.
SHA1CryptoServiceProvider()
System.Web.Security.
FormsAuthentication.
HashPasswordForStoringInConfigFile
(strSource, "SHA1")
```

下面方法直接使用

HashPasswordForStoringInConfigFile 生成:

```
public string Get_MD5_Method3(string
strSource)
{return System.Web.Security.
FormsAuthentication.
HashPasswordForStoringInConfigFile
(strSource, "MD5");}
```

结束语: 无论什么时候只要我们使用数据库开发网站, 我们就必须保护用户资料, 这非常必要。通过以上介绍的几种方法可以在网站开发过程中实现对网页以及网页中的用户信息进行加密保护, 以避免代码、用户重要资料外泄。

参考文献:

[1] 石志国 《信息安全学概论》清华大学出版社 2007

[2] 译: 良忠 《拒绝黑客: asp.net web 应用程序安全性剖析》电子工业出版社 2005

应该充分运用我们的头脑来找到解决问题的更快更好的办法, 只有当我们能够很好地解决自己工作中的问题的时候, 我们才能够为我们的用户提供能很好解决他们面临的问题的解决方案。

参考文献:

[1] 冯玉林 黄涛 面向对象的软件构造 [J] 软件学报 2001.3

[2] Jack Prisot Easy Persist ORM Framework Guide [M] Complex 2007.8

[3] 万军民 基于 net 的代码自动生成与实现 [J] 计算机工程 2005.12