ASP应用程序开发规范

版本号：1.02.0007

吴　　咏　　炜

E-mail: adah@sh163.net

二〇〇一年八月十二日

目　录

总论

本规范的目的是为 ASP 应用程序的开发提供一个公用的框架和基础，以利于开发出高性能、易于维护、易于重用的 Web 应用程序。

本规范既是规范，又是一种编程的参考，请大家在遇到问题时务必先在其中查找是否已涉及、解决类似问题。只有在大家充分利用本规范、不重复犯同样的错误时，这一规范才有存在的意义。

如有任何修正、补充意见，请及时与我联系，我会及时更新本规范和相关程序、文档。

服务器设置和站点结构

- Windows NT 4.0 服务器上应按先后顺序安装以下这些软件：NT Service Pack 3（或更高版本），Internet Explorer 5.01，NT Option Pack，MDAC 2.0，SQL Server 7.0 及其 SP2（如果需要的话），NT Service Pack 6a（或更高版本），（先删除 odbcinst.hlp 和 odbcinst.cnt 而后安装）MDAC 2.1 SP2。
- 检查安全性列表，对服务器作必要的调整，特别是：一、安装合适的补丁程序，如 Index Server 的 Hit highlighting 功能补丁（如果使用该功能的话；否则参见第二条）；二、在站点的“属性—主目录—配置”中删除不必要的映射以避免可能有的安全性漏洞，例如，不使用 Web password reset 应删除 htr，不使用 Hit highlighting 应删除 htw。
- 试着在注册表中设置 [HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\W3SVC\ASP\Parameters] "**ProcessorThreadMax**" = 14H (DWord)，并观察网站性能。
- WWW 服务的日志格式一般选为“W3C Extended Log File Format”，并在扩展属性中加选“日期”、“URI 查询”、“用户代理”和“引用站点”。
- 由于我们通常将数据库的连接信息放在 INC 文件中，这是一个潜在的不安全因素。因此我们应该使客户端无法查看 INC 文件。可在服务器中添加一个映射，让服务器调用示例中的 inc.exe 来查看 INC 文件，输出一个固定的字符串而不是 INC 文件的内容。
- 建议在根目录下开设 data 目录，去除其读、脚本和执行权限，也不索引该目录。然后，把应用程序中用到的不允许用户直接读取的数据（如 Access 数据库等）放在其中，以免用户私自下载该类数据。在 ASP 程序中，可用 `Server.MapPath("/data")` 来得到该目录的真实路径。

  注意：在 `Session.OnEnd` 和 `Application.OnEnd` 中不能使用 `Server.MapPath`！请参看示例中的 global.asa 和 connect.inc。
- 建议在根目录下开设 admin 目录，存放对网站数据库进行管理的程序模块。如果条件允许，可对此目录下的页面访问使用 NT 挑战/应答方式进行用户验证，以保证网站安全。
- 建议在根目录下开设 common 目录，用于存放公用的程序模块和样式表之类的信息。

- 图片应放在相应 HTML 文件（图片专属于某一页面的话）或根目录（如图片为共用图片）的 images 子目录下，图片的原始文件放在相应 images 目录的 img_src 子目录下（但不应上传到服务器上），Flash、AVI、QuickTime 等多媒体文件放在根目录下的 media 子目录中。
- 如果使用 Personal Web Server，建议增加“启动 PWS”和“停止 PWS”两个快捷方式，以方便调试需要重新启动 PWS 才能有效更改的模块，如 global.asa 和服务器端控件（个人 Web 管理器并不能完全停止 PWS 服务）。假设 Windows 安装在 C:\WINDOWS 目录下，则两个快捷方式应分别为“C:\WINDOWS\SYSTEM\inetsrv\pws.exe /start”和“C:\WINDOWS\SYSTEM\inetsrv\pws.exe /stop”。

基本要求

- 在每天早晨或在要对程序做大规模改动时，应当在工作用机本地备份服务器上的程序。
- 除非有特殊情况，文件、目录名称全部使用小写字母、数字和下划线的组合，其中不得包含汉字、空格或特殊字符（除文件名中允许带一个圆点）。纯 HTML 文件的后缀名统一为 htm。一个目录下的缺省文件为 default.asp 和 default.htm。
- 网页之间的超链接一般应全部为相对 URL。
- 建议的 HTML 文件结构为：

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0 Transitional//EN">
<html>
<head>
<title>文档标题（必填）</title>
<meta http-equiv="Content-Type" ...>
其他 meta 标记
样式表链接
样式表定义
客户端 JavaScript 初始化操作及函数定义
</head>
<body...>
. . .
</body>
</html>
```

- 为保证与最新 Web 标准的兼容，所有 HTML 标签中的属性值都应该使用双引号引起。即：我们应该写 `<a href="URL">`，而不是 `<a href=URL>` 或 `<a href='URL'>`。
- 包含汉字的页面字符集应设为 GB2312（在 `<head>` 段中含有 `<meta http-equiv="Content-Type" content="text/html; charset=gb2312">`，源码中的汉字之间不要添加空格（汉字与西文字符间可根据排版需要增加一个空格，但不应加在避头的符号之前或避尾的符号之后）。为发挥浏览器自动排版的功能并适应用户不同桌面大小的需要，在一段正文文字的内部尽量不要采用 `<br>` 标记来人工干预分行。
- 允许全文检索的页面（为与 Index Server 配合，或为了使 Internet 上的搜索引擎能有效检索）应加入 Keywords 元标记和 Description 元标记。
- 为保证 Index Server 能正常工作，如果一个网站中使用了不同语言的话，应在每一页中加入 MS.Locale 元标记。简体中文的话应使用 `<meta name="MS.Locale" content="ZH-CN">`，英文应使用 `<meta name="MS.Locale" content="EN-US">`（美

国英语）或 <meta name="MS.Locale" content="EN-GB">（英国英语）。注意服务器上 Index Server 需要安装相应的语言资源。

- 如无特殊原因，插入图形时在 <img> 标签中应当完整地提供 width、height 和 alt 属性。
- 不省略 <p>、<option>、<td> 等标记相应的结束标记。
- 在源代码中不应看到连续一个以上的 ——应使用表格、层、全角空格（在中文输入法中用 Shift-Space 切换成全角状态再键入空格）来达到类似排版效果
- 字号设定应使用级联样式表来实现，正文文字中（标题除外）禁止使用 <font size> 来定义字号大小。
- 汉字间的标点符号应使用全角符号。带括号的数字应使用半角括号，后面应该跟一个空格。英文字母周围的括号应使用半角括号。

程序风格

- 为方便以后他人阅读、理解和维护程序，应**规范使用**保留字、子程序、函数、变量的名称，大小写、命名方法等尽量向本规范的示例及微软的文档、示例和已有程序靠拢。
- 程序**缩进**的单位为两个空格。
- 程序开头应包含**版本**信息，形式如下：

```
<%
' Copyright (C) 2000 Wu Yongwei. All rights reserved.
'
' Version:              0.01.0001
' Original Author:      Wu Yongwei
' Creation Date:        24 Apr 2000
' Last Update:
' Modified By:
' Modification Date:
'

Option Explicit
. . .
```

  版权年份和原始作者名按实际情况填写。版本号从 0.01.0001（到程序基本定型时再将版本号升为 1.00.0000）开始编，每次修改后都应根据改动量大小增加版本号的相应各位。所有日期统一按 dd Mon yyyy 方式表示，以方便今后直接用计算机进行处理。
- 建议的**程序结构**为：版本信息，Option Explicit，环境设置（如 Response.Buffer = True），变量声明，Include 文件，子程序和函数定义，无输出的数据处理部分，HTML 及有显示输出的代码部分。
- 一般在使用 <**SCRIPT** LANGUAGE="*Language*" **RUNAT="Server"**> 方式嵌入到页面里的服务器端脚本中只包含变量声明及子程序和函数的定义（在其中包含的立即执行代码将在页面中所有以 <% . . . %> 形式包含的代码执行完毕后才被执行）。
- 使用 **Option Explicit**，并在每个变量使用前进行声明。
- **使用常数**以保持代码清晰易读。
- **变量命名**应以明确为本，不可为贪图省力而使用模棱两可的变量名。

- 一般使用**变量名** `Conn` 表示 Connection 对象，`RS` 表示 Recordset 对象，`SQL` 表示 SQL 查询字串。
- 数据库对象在使用后应尽早**显式关闭**；调用 `Connect` 的页面不再使用数据库时应调用 `Disconnect`（`Connect` 和 `Disconnect` 在 connect.inc 中定义）。
- 在 ASP 中返回字串值显示在 HTML 页面中时，如果不认为其中含有 HTML 格式，一般应使用 **`Server.HTMLEncode`** 函数进行编码或使用 `Text` 子程序(在 common.inc 中定义）进行显示，以免出现意料之外的结果。

  注意：如果要显示的字符串中含有不属于当前字符集的字符（例如，在当前字符集为简体中文时，从 Access 2000 的 TEXT 字段或 SQL Server 7.0 的 nvarchar 字段中读取像“Rømer”这样的欧洲人名输出到网页中），必须对要输出的字符串使用 `Server.HTMLEncode` 进行编码转换；同时，应当选用恰当的字体，保证输出的字符能够正常显示。另外，到 Communicator 4 为止的各个 Netscape 版本不支持在某一个 HTML 页面中输出该页面字符集中不包含的字符（当然，当页面字符集设为 Unicode 时可以支持所有字符的显示）。
- 在使用诸如“`xxx.asp?param=...`”的 URL 形式传参数到其他页面时，如果所传的参数可能含有除字母和数字之外的特殊字符，应使用 **`Server.URLEncode`** 函数进行编码。
- 使用 `INSERT`、`SELECT`、`DELETE` 等 SQL 语句时，如果所带数据中含有**单引号**，应当使用函数 `SQEncode`（在 common.inc 中定义）把一个单引号转成两个。
- Application、Session、Form 等**对象的名字**一般全部使用小写字母；Cookie 的名字一般使用大写字母，如果其中包含子键的话则子键名也全部使用大写字母。
- 有下列情况之一，表单必须在 **Action** 中明确指定递交目标文件名：一、表单所在的 HTML 或脚本文件是当前目录下的默认文档；二、表单所在的脚本文件既可能接受 GET 方式传送的数据，也可能接受 POST 方式传送的数据。
- 连接 ODBC 数据库的**连接字串**中不要增加无效的空格（特别是分号前后）；在某些情况下这会引起数据库驱动程序工作不正常。
- 如果情况允许，尽可能直接使用 **OLE DB** 而不是 ODBC 来连接数据库（参见“附录一：使用 OLE DB”）。
- 不推荐使用 **Browser Capabilities** 组件来判断浏览器类型，请使用客户端 JavaScript 脚本（参见示例中的 browser.js）或其他更好的 ASP 组件。

效率指南

网上有相当多关于如何对 ASP 应用程序进行精调的文章，请大家平时多加留意。本节对一些常见情况做一说明。本节参考了 http://www.asptoday.com/articles/20000113.htm（*Enhancing Performance in ASP – Part 1*）和 http://www.asptoday.com/articles/20000426.htm（*Enhancing Performance in ASP – Part 2*）和其他许多网上文献。

- 不使用不必要的 `<%@ LANGUAGE=VBScript %>`。
- 对于不需要会话支持的页面，在开头加入 `<%@ ENABLESESSIONSTATE=False %>`。
- 除非绝对必要，不使用事务处理（不写 `<%@ TRANSACTION=Required %>`）。
- 记住使用包含文件会降低脚本性能，因而不要在不必要时使用。要使用 adovbs.inc 中定义的 ADO 常量，可考虑在 global.asa 中声明 ADO 类型库，随后即可在脚本中对这些常量自由引用（参考示例）。

- 不必担心写注释会影响脚本性能。
- 在 ASP 脚本开头加入 `Response.Buffer = True`，或在服务器端设置对所有页面进行缓冲（IIS 5.0 默认打开缓冲，所以不需要在脚本开头加入此行；相反，在调试代码时使用 `Response.Buffer = False` 反而可以使出错信息显示得更清楚）。如果页面输出内容很多，可考虑在其中插入 `Response.Flush` 进行分段输出，并使用 `Response.IsClientConnected` 来探测用户是否已断开连接，如该属性为 `True` 则页面剩余部分无需再做处理。
- 不过多使用嵌入式 ASP 输出（`<%= ... %>`）。如果不影响程序结构性，尽可能把连续的输出用算符 `&` 连在一起用 `Response.Write` 发送到客户端。
- 使用字符串变量来缓冲输出（`strOutput = strOutput & "..."`）会降低脚本性能（重分配变量空间是一项极耗资源的操作）。
- 对重复使用的脚本代码进行封装，把代码和相关的局部变量放入子程序或函数块。
- 如果一个集合中的某个数值（像 `Request.Form("userid")`、`RS.Field(0)`）读取使用超过一次，应把它赋到一个本地变量中然后直接读取该变量。
- 如无特殊情况，使用 Request 对象中的某一集合时应清楚指明，即，应使用类似于 `Request.Form("userid")` 或 `Request.QueryString("userid")` 等形式而不是为贪方便而使用 `Request("userid")`。在不明确指明时，IIS 会按 QueryString、Form、Cookies、ClientCertificate、ServerVariables 的顺序搜索所有五个集合直至找到第一个匹配项。
- 尽量少使用 Session 变量——可使用 Session 变量来存储在一次会话中一直用到的数据，但不要用它来传递数据。
- 对于服务器端对象，尽可能晚创建、早释放。
- 记住使用 `INSERT`、`UPDATE`、`DELETE` 等 SQL 语句或使用存储过程来更新数据库效率要比使用记录集对象高，特别是在对多个记录进行操作时。
- 如非必要，在 ADO 中打开记录集时不使用复杂游标类型：对于读操作，缺省的服务器端前向游标性能最高，客户端游标（此时游标类型只能是 `adOpenStatic`）性能次之，其他类型的服务器端游标（键集、动态和静态）性能都要更低。
- 批量更新数据可考虑使用 `UpdateBatch`；此时游标位置应为 `adUseClient`，锁定类型为 `adLockBatchOptimistic`（参见示例中的 admin/news/publish.asp）。
- 除非使用 `UpdateBatch` 进行批量更新数据，使用客户端游标时只应使用缺省的 `adLockReadOnly` 锁定类型；读取数据后可考虑使用 `Set RS.ActiveConnection = Nothing` 断开连接以进一步提高性能（参见示例中的 admin/news/search.asp）。
- 在 `SELECT` 语句中只返回需要使用的字段，并使用数字而不是字段名来对其进行引用。
- 考虑使用以下这些方法来提高记录集的访问性能：使用 `GetString`（速度最快，但输出格式不够灵活）；使用 `GetRows`（灵活、高效，特别是对于较大的记录集）；把记录集的各个字段绑定到本地变量上（方便、灵活，在记录集较小时性能高于 `GetRows`）。
- 在读取多个记录时适时使用 `CacheSize` 属性，特别当使用服务器端游标时。
- 如使用 Command 对象应手工添加参数而不是使用 `Refresh` 方法来自动获取。

公用程序模块

adovbs.inc

微软的 ADO 常量定义。该文件应主要用作参考，不推荐在应用程序中直接包含此文件。可考虑使用 ADO 的类型库以直接使用这些常量（参见示例中的 global.asa）。

chklogin.inc

在用户未登录或被强制注销时转移到登录页面（参见示例）。

common.inc

在该文件中定义了一些最常用的常量、子程序和函数。下面列表给出简要描述（具体使用请参看源代码）：

名称	类型	描述
Quot	常量	引号（不过请尽可能使用“""”以提高程序执行效率）
Text	子程序	在页面中显示文本（可包含特殊符号和回车换行）
Redirect	子程序	产生在客户端重定向的代码
IIf	函数	模拟 VB 中的该函数
BinFromHex	函数	将十六进制表示的内码转换成所对应的二进制流
BinToHex	函数	将二进制流转换成相应的十六进制表示的内码
BinToHexEx	函数	同 BinToHex，但允许在字节之间插入分隔符
Rot13	函数	实现 ROT13 编码（A↔N, B↔O,…, M↔Z）
SQEncode	函数	在 SQL 命令执行前把一个单引号变成两个
SDate	函数	把一个日期转换成为执行 SQL 语句所需的形式
FormatDate	函数	返回以四位年份形式表示的日期字符串
Check	函数	根据参数值是否相等返回 " checked" 或空串
IsCSym	函数	判断一个字符是否为字母、数字、下划线之一
IsValidID	函数	判断一个字符串是否非空且仅包含字母、数字和下划线

在只使用该文件中的一小部分常量、子程序或函数时，可以把该部分代码复制到 ASP 源程序中直接使用以利提高程序执行效率。不过，如果使用了有可能做进一步改动的函数如 SDate 时则该建议不适用。

connect.inc

定义数据库连接和断开的子程序并声明变量 Conn（参见示例）。

constant.inc

从数据库中读入系统常量（参见示例中的 global.asa）。

counter.inc

定义子程序 `IncrementCounter`，可将 data 目录下的 counter.txt 文件中的计数加一并存回该文件中（如果该文件不存在则自动创建该文件）。

ftemplate.inc

在 ASP 中实现 Fast Template 模板。

login.inc

处理用户登录（参见示例中的 login.asp）。

logout.inc

处理用户注销（参见示例中的 login.asp 和 global.asa）。

pjmd5.inc

由 Paul Johnston 实现的 MD5 散列值计算程序，略有改动。可利用其实现了不明文传送口令的安全登录（参见示例中的 pjmd5.js、login.asp 和 login.inc）。

pupload.inc

纯 ASP 上载的包含文件。在英文 Windows 中，只使用这个文件就可以实现上载文件的存盘和存入数据库；在中文 Windows 中，也可以使用该文件把上载文件存入数据库，但如果要将上载文件直接存盘，必须安装 SoftArtisans 的 SA-FileManager 控件并设 `Application("use_safilemanager")` 为 `True`，或者安装 MDAC 2.5（Windows 2000 中已包含该组件）及更新版本并设 `Application("use_adostream")`为 `True`。

rc4.inc

实现了 RSA 的 RC4 加密算法。在许多情况下可能需要配合 `BinFromHex` 和 `BinToHex` 函数（在 common.inc 中定义）一起使用。

tearhttp.inc

利用 w3 Sockets 组件实现 HTTP 1.0 中的 GET 和 POST 方法。

timethis.inc

用于测试脚本运行时间的辅助包含文件，其中定义了 `GetTime`（获取以毫秒为单位的时间值）函数和 `TimeToString`（把毫秒值转变成含时、分、秒、百分之一秒数值的字符串）函数。

style.css

定义大多数页面都需要的通用样式。

工具指南

EditPlus 2

EditPlus 2 非常小巧、易用，具有极优秀的代码着色效果，是我们推荐使用的编辑器。使用时请注意（对于版本 2.1，其他版本的菜单位置可能有所不同）

- 在 Tools – Preferences – General 中选中 HTML tag in lower case、Working directory follows active document 和 HOME key goes to first non-space character。
- 在 Tools – Preferences – Files 中设定一个 Backup file directory，其下的 Settings & syntax 中 HTML 文件类型的后缀增加一个“inc”，Tab/Indent 中设 Tab 和 Indent 值为 2、并选中 Insert spaces instead of tab。
- 在 General – Fonts 下建议把 Custom 1 定义为宋体 9 磅字，并在编辑中文文档时选用此字体。
- 重定制 HTML 模板以提高工作效率。参考模板如下所示：

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0 Transitional//EN">
<HTML>
<HEAD>
<TITLE> New Document </TITLE>
<META HTTP-EQUIV="Content-Type" CONTENT="text/html; charset=gb2312">
<META NAME="Author" CONTENT="YourName">
</HEAD>

<BODY BGCOLOR="#FFFFFF">
^!
</BODY>
</HTML>
```

- 在具体项目中可定制项目模板。参考模板如下所示（请注意其中大小写变化，

<body> 中删去了 bgcolor 属性，<head> 中加入了样式表定义）：

```
<%
' Copyright (C) 2000 Wu Yongwei. All rights reserved.
'
' Version:            0.01.0001
' Original Author:    Wu Yongwei
' Creation Date:      ^!
' Last Update:
' Modified By:
' Modification Date:
'

Option Explicit
Response.Buffer = True
%>
<!-- #include virtual="/common/common.inc" -->
<!doctype html public "-//W3C//DTD HTML 4.0 Transitional//EN">
<html>
<head>
<title> New Document </title>
<meta http-equiv="Content-Type" content="text/html; charset=gb2312">
<!-- <meta http-equiv="Pragma" content="no-cache"> -->
<meta name="Author" content="Wu Yongwei">
<link rel="stylesheet" type="text/css" href="/common/style.css">
</head>

<body>

<%

%>

</body>
</html>
```

Visual InterDev 6.0

Visual InterDev 的最大优点是它的对象感知技术，能够非常方便地给出对象的属性和方法。为和其他编辑器产生的代码保持一致，请在 Tools – Options – Text Editor – Tabs – HTML 中把 Tab size 和 Indent size 改为 2，下面把 Keep tabs 改为 Insert spaces。

常见问题

HTML/ASP

1. 如何防止一个页面被浏览器缓存？

✓ 如果某一页内容不希望浏览器直接从缓存中读取，有下列三种方法可以使用。

i. 可在 <head> 段中加入 <meta http-equiv="Pragma" content="no-cache">（立即失效）。不过，要注意：一个使用 POST 方法接收数据的 ASP 页面如果其中定义了页面字符集（charset=*gb2312*）则不能使用该方法，而应考虑下面的方法 ii（使用 Response.Expires = 1），否则 Netscape 浏览器在页面字符集设定和选定的字符集设定不同时可能会陷入一个循环而无法显示该页面；在表单递交的目标文件中一般也不必定义禁止缓存，浏览器会访问服务器来获取表单递交的结果。如果在一个含有表单的页面使用了禁止缓存，有一个副作用是发送表单后，用户选“返回”或“Back”回到前一页可能会丢失原先填在表单里的内容。

ii. 在 ASP 代码开头加入 Response.Expires = *m* 设置过期分钟数（不能为小数；如希望更精确可考虑使用 ExpiresAbsolute）进行控制。请仔细考虑该页面的过期时间，尽量使浏览器能有效利用缓存较快地显示网页内容。

iii. 为了防止共用代理服务器错误使用缓存，可把对页面 "nocache.asp" 的调用改为 "nocache.asp?nocacheurl=" & Server.URLEncode(Now)。如可能对代理服务器进行设置，应使其对服务器发来的页面中 HTTP 标题 Cache-control 为 private 的页面不予缓存——这也是 HTTP 1.1 规范对共用缓存的要求。WinGate 似乎无法通过这种方法对缓存内容进行控制，可令其对 URL 中含有“.asp”的页面不予缓存。

2. 为什么有时候 Response.IsClientConnected 似乎工作不正常？

✓ 在 IIS 4.0 中，要用 Response.IsClientConnected 来得到正确的结果，必须在此之前向客户端输出内容；IsClientConnected 探测的是最近这次输出时客户端是否仍然连通。如果使用了 Response.Buffer = True 的话，还需要在使用 IsClientConnected 之前使用 Response.Flush！

IIS 5.0 中没有该限制，允许随时使用 Response.IsClientConnected 来探测浏览器是否连通。

3. ASP 中 CreateObject 和 Server.CreateObject 有何区别？

✓ VBScript 内建的 CreateObject（或 JScript 中的 ActiveXObject）函数相当于通过脚本引擎调用 CoCreateInstance 创建自动化对象，而 Server.CreateObject 则是通过 MTS 调用 ObjectContext 对象的 CreateInstance 方法创建对象。因此：

i. CreateObject 函数产生的对象不能参与事务处理；

ii. CreateObject 函数产生的对象不能访问 ASP 的内建对象；

iii. 使用 CreateObject 函数 ASP 不会调用 OnStartPage/OnEndPage 页面方法；

iv. ASP 不知道 CreateObject 函数所创建对象的线程模型；

v. IIS 4.0 下 CreateObject 的速度要比 Server.CreateObject 快。

我们的测试表明，在 IIS 4.0 中使用 Server.CreateObject 创建对象所需的时间大致是 CreateObject 的 2 至 4 倍。但在 IIS 5.0 中两者性能相同。另外，数据库测试（使用 MDAC 2.1 和 2.5）的结果表明，如果不显式创建 Connection 对象而直接把连接字串传给 Recordset，那么只有在直接使用 CreateObject 时 SQL Server 才会启用连接池。我们建议，在对数据库进行操作时，除非只打开一个 Recordset 对象对其作简单操作，一般还是应该使用 Server.CreateObject（请参见示例中 CreateObject 和 Server.CreateObject 的使用）。

VBScript

4. 调用子程序（Sub）和函数（Function）时到底是否需要使用括号？

✓ 一般而言，在 VBScript 中调用子程序和函数时，是否使用括号的形式可以有很多种：

```
SubOrFuncNoParam                    SubOrFuncNoParam()
r = SubOrFuncNoParam                r = SubOrFuncNoParam()
Call SubOrFuncNoParam               Call SubOrFuncNoParam()
SubOrFuncOneParam Param             SubOrFuncOneParam(Param)
r = SubOrFuncOneParam(Param)        Call SubOrFuncOneParam(Param)
SubOrFuncTwoParam Param1, Param2
r = SubOrFuncTwoParam(Param1, Param2)
Call SubOrFuncTwoParam(Param1, Param2)
. . .
```

但不允许

```
SubOrFuncTwoParam(Param1, Param2)
r = SubOrFuncTwoParam Param1, Param2
. . .
```

使用 `Call` 关键字时传递参数必须使用括号。如果不使用 `Call` 关键字，一般建议在不使用返回值（Sub，或 Function 放弃返回值）时不要使用括号（此时，使用括号传递单个参数还会有一个副作用：所有变量将强制为按传值方式而不是按引用方式传送）；反之，使用返回值（Function）时则使用括号。对于下列情况应予特别注意：

- 在 VBScript 中调用 JScript 的函数（function）返回数值时，即使该函数没有参数也**必须**使用括号。

数据库

5. 为什么应当显式释放对象？

✓ 一、显式释放对象可以及早释放服务器资源，把连接返回到连接池；二、缺省的资源回收机制并不能真正完善地收回资源。对于第二点，网上有相关文章谈及：

> You must close recordset, set to nothing, close connections and set to nothing in that sequence. The standard garbage collection is incomplete and unreliable.
>
> DataReturn [an ASP webhosting company] has many sites that fail horribly if they let IIS do automatic garbage collection. Adding Close/Set Nothing makes the sites work like champs again. It is mandatory in any high volume site.
>
> (http://www.4guysfromrolla.com/webtech/060999-2.shtml)

6. 应当使用系统 DSN 还是文件 DSN？

✓ 在同时访问人数较大时，文件 DSN 由于每次打开数据库连接时都需要读写文件，效率比系统 DSN 要低。所以在这两者之间我们推荐使用系统 DSN。

不过，效率更高的是使用无 DSN 连接。也就是说，在连接字串中直接填入驱动程序、服务器名称、数据库名称等信息，像 `strConn = "Driver={Microsoft Access Driver`

(*.mdb)};DBQ=C:\Inetpub\wwwroot\data\test.mdb"。Web 上的测试表明：

> DSN-less connections were slightly faster than System DSN connections. The increase in performance was nothing monumental; the greatest performance boost was a mere 13% faster with 64 concurrent requests. For one, two, or four concurrent requests, there was virtually no performance improvement. In fact, no noticeable improvement is seen in a DSN-less connection over a System DSN until there are 10 or more concurrent connections.
>
> (http://www.4guysfromrolla.com/webtech/070399-1.shtml)

效率最高的方式是使用 OLE DB。请参见“附录一：使用 OLE DB”和网上相关文章：如 http://www.4guysfromrolla.com/webtech/063099-1.shtml。使用 OLE DB 需要安装 MDAC 2.0 或以上版本。

7. 在 Application 或者 Session 变量中存放 COM 对象（如 Connection 或 Recordset）是否可以提高服务器性能？

✓ 不。如果把 COM 对象、特别是“单元”模型的 COM 对象存放在 Session 变量中的话，由于线程和资源方面的原因，反而会严重影响 Web 服务器的性能。可参见相关文章，如：http://www.learnasp.com/learn/nosessionobjects.asp。

8. 我使用 ADOX.Catalog 对象来创建 Access 97 数据库，但在要使用新创建的数据库时，系统报错“文件正在使用中”，如何解决该问题？

✓ 如果想要在创建 Access 97 数据库的同一页面中使用该数据库，应先断开 Catalog 对象与该数据库的活动连接。代码如下所示：

```
strDBPath = Server.MapPath("/data") & "\test.mdb"
strConn = "Provider=Microsoft.Jet.OLEDB.3.51; Data Source=" & strDBPath
Set objCat = Server.CreateObject("ADOX.Catalog")
objCat.Create strConn
Set objCat.ActiveConnection = Nothing
Set objCat = Nothing
Set Conn = Server.CreateObject("ADODB.Connection")
Conn.Open strConn
. . .
```

如使用 Access 2000 数据库则该问题不存在。

附录一：使用 OLE DB

虽说OLE DB听起来有些陌生，可实际上在使用ADO的同时我们就已经在使用OLE DB了——ADO使用的数据库组件模型就是OLE DB。对于OLE DB来说，不同的数据库类型就对应着不同的数据提供者（*data provider*），而我们通常用的ODBC只是这些数据提供者中的一种，即Microsoft OLE DB provider for ODBC Drivers（*MSDASQL*），同时，微软也对一些常见的数据源在OLE DB层提供了直接支持，其中包括了Oracle、SQL Server和Access数据库（要注意MDAC 2.0中所带的Jet 3.51的引擎只支持Access 97数据库，而MDAC 2.1中所带的Jet 4.0引擎虽然同时支持Access 97和Access 2000，但用于对Access 97进行操作时效率相当低；一般建议先安装MDAC 2.0 SP1后安装MDAC 2.1 SP2）。要从使用ODBC改为使用OLE DB相当简单：比如，原先调用Access 97数据库使用的连接字串为“`Driver={Microsoft Access Driver (*.mdb)};DBQ=C:\Sample.mdb`”，现在只需改为“`Provider=Microsoft.Jet.OLEDB.3.51; Data Source=C:\Sample.mdb`”程序执行效率就会大幅度提高——非常简单。SQL Server的连接字串形如“`Provider=SQLOLEDB; Data Source=Server; Initial Catalog=pubs; User ID=sa; Password=passwd`”。

我们在使用Access数据库的情况下对两种连接方式进行了测试，结果列在下面表中以作参考。测试用机的硬件为赛扬466MHz/128MB内存，操作系统为Windows NT 4.0（SP6），Web服务器为IIS 4.0，安装了MDAC 2.0 SP1和MDAC 2.1 SP2；测试脚本是对一个数据库反复做添加和删除操作，测试过程中只对连接字串做了修改，每次测试前对数据库进行了恢复。测试软件为微软的Web Application Stress Tool，表中括号外的数值为TTLB（Time To Last Byte），单位为毫秒，括号中的值为该项时间值与基准时间值相比的比率。

同时连接数：1

	Access 97	Access 2000
Jet OLE DB 3.51	124.22 (**100%**)	N/A
Jet OLE DB 4.0	187.72 (151%)	140.96 (113%)
ODBC DSN-less	235.61 (190%)	179.58 (145%)
ODBC System DSN	219.30 (177%)	164.41 (132%)
ODBC File DSN	257.10 (207%)	205.40 (165%)

同时连接数：2

	Access 97	Access 2000
Jet OLE DB 3.51	255.93 (**100%**)	N/A
Jet OLE DB 4.0	511.60 (200%)	275.16 (108%)
ODBC DSN-less	2267.59 (886%)	2066.04 (807%)
ODBC System DSN	2151.02 (840%)	1751.65 (684%)
ODBC File DSN	2643.86 (1033%)	2297.17 (898%)

从测试结果中可以看出：性能最高的是使用Microsoft Jet 3.51 OLE DB Provider连接Access 97数据库，其次是使用Jet 4.0连接Access 2000数据库；无DSN ODBC连接在本次测试中略慢于系统DSN，而文件DSN则速度最低。由于ODBC使用Jet 4.0驱动程序，所以在使用ODBC连接时Access 2000的测试成绩全部高于Access 97。另外，可以明显注意到的是，在同时连接数由1上升到2时，OLE DB的性能线性降低，服务器的吞吐量基本不变；而与此同时ODBC的性能则急剧下降，负荷能力相比之下差远了。

附录二：CursorType 和 LockType 的组合

并不是每一种 CursorType 和 LockType 的组合都能同样被支持——这取决于许多情况。下面以表格方式列出 Access 97 和 SQL Server 7.0 的 OLE DB 提供者支持的组合类型（MDAC 2.1）：

Access 97/UseServer

	ReadOnly	Pessimistic	Optimistic	BatchOptimistic
ForwardOnly	**F \ R**	K \ P	K \ O	K \ B
Keyset	**K \ R**	**K \ P**	**K \ O**	**K \ B**
Dynamic	S \ R	K \ P	K \ O	K \ B
Static	**S \ R**	K \ P	K \ O	K \ B

Access 97/UseClient

	ReadOnly	Pessimistic	Optimistic	BatchOptimistic
ForwardOnly	S \ R	S \ B	S \ O	S \ B
Keyset	S \ R	S \ B	S \ O	S \ B
Dynamic	S \ R	S \ B	S \ O	S \ B
Static	**S \ R**	S \ B	**S \ O**	**S \ B**

MS SQL/UseServer

	ReadOnly	Pessimistic	Optimistic	BatchOptimistic
ForwardOnly	**F \ R**	**F \ P**	**F \ O**	**F \ B**
Keyset	**K \ R**	**K \ P**	**K \ O**	**K \ B**
Dynamic	**D \ R**	**D \ P**	**D \ O**	**D \ B**
Static	**S \ R**	K \ P	K \ O	K \ B

MS SQL/UseClient

	ReadOnly	Pessimistic	Optimistic	BatchOptimistic
ForwardOnly	S \ R	S \ B	S \ O	S \ B
Keyset	S \ R	S \ B	S \ O	S \ B
Dynamic	S \ R	S \ B	S \ O	S \ B
Static	**S \ R**	S \ B	**S \ O**	**S \ B**