Guia de Consulta Rápida

Windows Script Host

Roberto G. A. Veiga

Novatec Editora
www.**novateceditora**.com.br

Guia de Consulta Rápida **Windows Script Host** de Roberto G. A. Veiga.

ISBN: 85-85184-82-5

Novatec Editora Ltda.
Rua Cons. Moreira de Barros 1084 Conj. 01
02018-012 São Paulo - SP Brasil
Tel.: (0xx11) 6959-6529
Fax: (0xx11) 6950-8869
E-mail: novatec@novateceditora.com.br
Site: www.novateceditora.com.br

Marcas registradas

Introdução

O Windows Script Host, ou simplesmente WSH, é um ambiente de scripting para a plataforma Windows de 32 bits que veio substituir a linguagem batch herdada do MS-DOS. Permite a desenvolvedores, administradores de sistemas e consultores de informática desenvolverem scripts para a realização de várias tarefas dentro do Windows, desde as mais simples, como exibir texto numa caixa de mensagem, até as mais complexas, como manipular os objetos do Active Directory via ADSI.

A principal característica do WSH é o fato de ser independente de linguagem de programação, o que significa que qualquer linguagem, desde que disponha de um mecanismo de scripting (interpretador) compatível com a tecnologia ActiveX, pode ser utilizada no desenvolvimento de scripts do WSH. A Microsoft oferece dois mecanismos de scripting - o do VBScript e o do JavaScript -, mas é possível encontrar outros mecanismos de scripting para linguagens populares, como Perl, Tcl ou Python, fornecidos por outros fabricantes.

Abaixo, um exemplo de script do WSH - o qual utiliza o VBScript como linguagem de programação - que exibe a frase "Alô, mundo!" numa caixa de mensagem do Windows:

Exemplo

```
Rem Exibe a frase "Alô, mundo!". Para testá-lo, salve como um arquivo com
    extensão .vbs e depois execute-o:
Wscript.Echo "Alô, mundo!"
```

Ao executar o arquivo .vbs, será exibida a seguinte caixa de mensagem:

Para os programadores ASP, o WSH parecerá bem familiar. De fato, o ASP é um ambiente de scripting, ou "host", para o Microsoft Internet Information Server, assim como o WSH, como dito anteriormente, é um ambiente de scripting para a plataforma Windows. Ambos são compatíveis com a tecnologia ActiveX da Microsoft.

Este guia abordará a versão mais recente do WSH, a 2.0, que traz consigo a versão 5.1 do Windows Script Engine e acrescenta a possibilidade de utilização da linguagem XML no desenvolvimento de scripts.

Muita informação adicional sobre o WSH e outras tecnologias de scripting da Microsoft pode ser obtida no site http://msdn.microsoft.com/scripting. É esse site que você deve acessar em busca de novidades sobre o WSH, incluindo novas versões e patches de atualização.

Existem outros sites que também oferecem informações sobre o WSH. Três que eu sugiro são:

- http://wsh.superexpert.com,
- http://www.devguru.com e
- http://wsh.glazier.co.nz.

Por que usar o WSH?

- Um script WSH pode ser executado tanto a partir do Windows quanto a partir do prompt da linha de comando.
- WSH é independente de linguagem de programação, o que dá ao desenvolvedor de scripts liberdade para utilizar a sua linguagem favorita, desde que ela suporte a tecnologia ActiveX da Microsoft.
- A partir da versão 2.0 do WSH, é possível utilizar a XML, inclusive reutilizando antigos arquivos de script .vbs e .js dentro de um único arquivo .wsf.
- Baixa utilização de memória do sistema.
- Mapeamento automático de extensões de arquivo, o que siginifca que o WSH entregará a execução de arquivos com extensão .vbs diretamente ao interpretador VBScript e de arquivos .js ao interpretador JScript.
- Não é preciso nenhum programa especial para gerar scripts do WSH, pois não são nada além de arquivos de texto. O bom e velho Notepad dá conta do recado.

Instalação

O WSH vem instalado por padrão no Windows 98 (versão 1.0) e no Windows 2000 (versão 2.0). No Windows 95 e no Windows NT ele deve ser instalado como um componente adicional. O arquivo de instalação, chamado **ste51en.exe** para a versão 5.1 em inglês do pacote Windows Script Engine, pode ser baixado gratuitamente, via download, do site http://msdn.microsoft.com/scripting.

A instalação do WSH no Windows 95 ou no NT, como a de qualquer componente extra do Windows, é extremamente rápida e descomplicada. Basta executar o arquivo de instalação e nenhuma outra ação será requerida do usuário que não a de responder SIM quando for solicitado e clicar no botão OK no final do processo. No Windows 98 e no 2000, a decisão de ter o WSH instalado ou não, pode ser tomada na instalação do Sistema Operacional, quando surge a janela **Componentes Opcionais**. O WSH pode ser encontrado dentro de **Acessórios** ao clicar no botão **Detalhes**. Posteriormente, o WSH pode ser instalado/desinstalado em **Painel de Controle ® Adicionar/Remover Programas ® Instalação do Windows ® Acessórios**.

À época em que estava escrevendo este guia, a Microsoft estava liberando a versão 5.5 do Windows Script Engine, que também utiliza a versão 2.0 do WSH mas, por outro lado, traz melhorias para os mecanismos de scripting do VBScript e do JScript.

Executando um script WSH

Existem duas maneiras de executar um script WSH:

Utilizando o WScript

O WScript.exe permite que os arquivos do WSH sejam executados a partir do Windows. Para executar um arquivo do WSH utilizando o WScript.exe, você pode:

- Dar duplo clique no arquivo.
- Entrar o caminho completo e o nome do arquivo em **Iniciar** ® **Executar** e clicar em **OK**.
- Entrar Wscript.exe seguido do caminho completo e do nome do arquivo em **Iniciar** ® **Executar** e clicar em **OK**.

Utilizando o CScript

O CScript.exe, por outro lado, permite que os arquivos do WSH sejam executados a partir do prompt da linha de comando. Para usar o CScript, utilize a seguinte sintaxe:

cscript [*opções*] [*script*] [*parâmetros*]

Argumento	Descrição
opções	
//I	Habilita o modo interativo.
//B	Desabilita o modo interativo; o script é executado em segundo plano.
//T:*nn*	Habilita time-out. O padrão é nolimit. Quando algum valor for especificado, o script só poderá ser executado até que ele seja atingido.
//logo	Exibe uma mensagem sobre o WSH.
//nologo	Não exibe a mensagem sobre o WSH.
//H:*valor*	Valor tanto pode ser CScript quanto WScript. Esse parâmetro registra um ou outro como a aplicação padrão para executar scripts. Se omitido, WScript.exe será assumido.
//S	Salva as atuais configurações da linha de comando.
//?	Mostra a utilização dos comandos do CScript.exe.
//E:*engine*	Executa o script com o mecanismo de scripting especificado em engine.
//D	Habilita o depurador.
//X	Carrega o programa no depurador.
//Job:<JobID>	Executa o *JobID* especificado em um arquivo .wsf.
script	Nome do arquivo script.
parâmetros	Parâmetros passados ao script. Devem ser precedidos por uma barra (/).

Exemplo

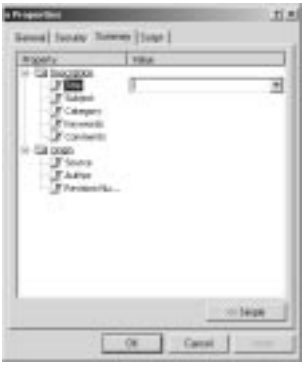
Executa o script chamado "x.vbs" com o parâmetro "//nologo" e habilitando o depurador:
C:\>cscript //nologo //D x.vbs

Arquivos .WSH

Os arquivos .wsh são arquivos de texto onde são armazenadas configurações específicas de um script. Ele é criado automaticamente quando são alteradas as propriedades do script conforme demonstrado abaixo.

Para criar um arquivo .wsh:

1. Dê um clique com o botão direito do mouse sobre um arquivo de script.
2. No menu de contexto escolha **Propriedades**. Abaixo, a página de propriedades do arquivo "x.vbs" no ambiente Windows 2000:



3. Altere as configurações que quiser na página de propriedades.
4. Clique no botão **OK**.

O conteúdo do arquivo .wsh criado seria semelhante a este:

```
[ScriptFile]
Path=C:\x.vbs
[Options]
Timeout=0
DisplayLogo=1
BatchMode=0
```

Quando for executar o arquivo .wsh através da linha de comando, ou da caixa de diálogo Executar, dê duplo clique no arquivo, que o WScript.exe ou o CScript.exe lerá o arquivo em busca das configurações definidas para o script ao qual o arquivo .wsh estiver associado e, só então, o script será executado. É importante ressaltar que a criação do arquivo .wsh não elimina a necessidade da existência do arquivo de script.

Utilização de Elementos XML

A eXtensible Markup Language (XML) é, em alguns pontos, semelhante à HTML, pois também é uma linguagem de formatação de conteúdo para exibição de documentos, principalmente na Internet. A grande e fundamental diferença é que a XML não está restrita a tags predefinidas como a HTML, o que oferece muita flexibilidade ao desenvolvedor. A XML é um padrão aberto e é de se esperar que sua utilização só aumente nos próximos anos.

Se você não tiver muita familiaridade com a XML, infelizmente este guia não deverá ser o seu ponto de partida. Sugiro que você adquira o Guia de Consulta Rápida da Novatec sobre a XML, ou então procure informações na Internet, em sites como o http://www.microsoft.com/workshop/xml ou o http://msdn.microsoft.com/xml.

Para poder usar a XML é necessário que haja um analisador da linguagem instalado na máquina. O Windows Script Host 2.0 vem com esse analisador. Lembre-se que scripts do WSH que utilizem a XML deverão ter a extensão de arquivo .wsf.

Arquivos .WSF

Os arquivos .wsf são justamente arquivos de script que utilizam tags XML e que podem, dentre outras coisas, executar simultaneamente código em VBScript e em JScript. Portanto, eles não estão atrelados diretamente a nenhum mecanismo de scripting e são capazes de utilizar mais de um ao mesmo tempo, desde que os mecanismos de scripting sejam compatíveis com a tecnologia ActiveX. Abaixo, um exemplo de arquivo .wsf:

```
<?XML version="1.0" ?>
<job>
<script language="VBScript">
<![CDATA[
Wscript.Echo "Windows Script Host eh quente!"
]]>
</script>
</job>
```

Elementos da XML

Na tabela abaixo estão listados os elementos da linguagem XML que poderão ser usados em scripts do WSH:

Elementos	Descrição
<?XML ?>	Indica que um arquivo deve ser analisado por um analisador XML.
<?job ?>	Indica se o WSH deverá exibir mensagens de erro durante a execução do script e se deverá acionar o depurador quando da ocorrência de um erro
<job>	Define um *job* dentro de um arquivo .wsf.
<object>	Cria um objeto.
<package>	Permite que mais de um *job* seja definido dentro de um arquivo .wsf.
<reference>	Usada para incluir uma referência a uma biblioteca externa ao script.
<resource>	Isola dados dentro de um arquivo de script.
<script>	Insere um bloco de código escrito em VBScript ou JScript dentro de um arquivo .wsf.
<![CDATA[...]]>	Evita que o analisador XML falhe ao encontrar palavras reservadas da linguagem em código de script escrito em VBScript ou JScript.

<?XML ?>

Indica que um arquivo deve ser analisado por um analisador XML. Deve ser inserido no início do arquivo.

<?XML version=*"versão"* **>**

Parâmetro	Descrição
version	Versão da XML sendo executada.

<?job ?>

Especifica atributos para tratamento de erros, informando ao WSH se deve ou não ser exibida uma mensagem de erro durante a execução do script e, caso ocorra algum erro, se o depurador deverá ou não ser acionado.

<?job error=*"varboolean"* **debug=***"varboolean"* **?>**

Parâmetro	Descrição
error	Booleano. Habilita (True) ou não (False) a exibição de mensagens de erro durante a execução do script.
debug	Booleano. Aciona (True) ou não (False) o depurador quando ocorrer um erro na execução do script.

Exemplo

```
<?XML version="1.0" ?>
<job>
<?job error="true" debug="true"?>
<script language="VBScript">
WScript.Echo "Olá mundo!"
</script>
</job>
```

<job>

Define um *job* dentro de um arquivo .wsf. Um *job* é uma tarefa a ser executada quando o script for chamado.

<job [**id**="*JobID*"]**>**

Código a ser executado

</job>

Parâmetro	Descrição
id	Nome pelo qual será feita referência ao job dentro do script e através do qual poderá ser executado a partir da linha de comando.

Exemplo

Utilização da tag <job > para definir mais de um *job* dentro de um arquivo .wsf:

```
<?XML version="1.0" ?>
<package>
<job id="Olá">
<script language="VBScript">
WScript.Echo "Olá!"
</script>
</job>
<job id="Adeus">
<script language="VBScript">
WScript.Echo "Adeus!"
</script>
</job>
</package>
```

<object>

Cria um novo objeto dentro do arquivo .wsf. Semelhante à tag de mesmo nome do HTML. A tag *object* define um objeto global dentro do código do script, que pode ser acessado por código escrito em VBScript ou JScript.

<object id=*"objID"* [**progid**="*progID*" |
classid="**clsid**:*GUID*"] **/>**

Parâmetro	Descrição
id	Nome pelo qual o objeto será conhecido no script.
progid	Nome da classe a partir da qual poderá ser definido um objeto.
classid	Código da classe a partir da qual poderá ser definido um objeto.

Exemplo

```
<?XML version="1.0" ?>
<job>
<object id="WshShell" progid="WScript.Shell"/>
<script language="VBScript">
<![CDATA[
WshShell.Popup "WshShell eh um objeto declarado através da tag <object >."
]]>
</script>
</job>
```

<package>

Permite que haja múltiplos *jobs* dentro de um mesmo arquivo .wsf. A utilização de *package* é opcional quando há apenas um *job* no script.

<package>

Jobs

</package>

Exemplo

Utilização da tag <package > em um arquivo .wsf com vários *jobs*:

```
<?XML version="1.0" ?>
 <package>
<job id="Um">
<script language="VBScript">
WScript.Echo "Job Um."
</script>
</job>
<job id="Dois">
<script language="VBScript">
WScript.Echo "Job Dois."
</script>
</job>
<job id="Três">
<script language="VBScript">
WScript.Echo "Job Três."
</script>
</job>
</package>
```

<reference>

Esta tag é utilizada quando se quer fazer uso de objetos existentes em uma biblioteca externa ao script.

<reference [**object=***"progID"* **| guid=***"typelibGUID"*]
[**version=***"version_info"*] **/>**

Parâmetro	Descrição
object	Nome da classe a partir da qual um objeto de uma biblioteca externa pode ser instanciado.
guid	O *guid* da biblioteca à qual está sendo feita referência.
version	Versão da biblioteca a qual está sendo feita referência. Se nenhum valor for fornecido, será assumido 1.0.

Exemplo

Utilização da tag <reference > para acessar o objeto Recordset do ADO:

```
<?XML version="1.0" ?>
<package>
<job>
<reference object="ADODB.Recordset"/>
<script language="VBScript">
WScript.Echo "adUseClient = " + CStr(adUseClient)
</script>
</job>
</package>
```

<resource>

Armazena dados de texto ou numéricos para posterior utilização dentro do script.

<resource id=*"resourceID"***>**

dado

</resource>

Exemplo

```
Utilização da tag <resource> para armazenar um valor:
<?XML version="1.0" ?>
<job>
<resource id="mensagem">
WSH eh legal!
</resource>
<script language="VBScript">
strMensagem = getResource("mensagem")
WScript.Echo strMensagem
</script>
</job>
```

Parâmetro	Descrição
id	Nome pelo qual será feita referência ao recurso dentro do script. Deverá ser fornecido como argumento à função getResource() para que se obtenha o valor dos dados isolados pela tag.

<script>

Insere um bloco de código dentro do script, especificando também a linguagem que está sendo utilizada. Semelhante à tag de mesmo nome encontrada no HTML.

<script language=*"linguagem"* **[src=***"arquivo"***]>**

Código

</script>

Parâmetro	Descrição
language	Especifica a linguagem a ser utilizada. Atualmente são suportadas as linguagens VBScript e JScript.
src	A localização de um arquivo externo, contendo código de script, a ser incluído no arquivo .wsf.

<![CDATA[...]]>

<![CDATA[...]]> não é propriamente uma instrução da XML. Sua única função é impedir que o analisador XML falhe ao analisar um arquivo .wsf que contenha código em VBScript ou JScript onde se encontrem palavras ou caracteres reservados da XML.

<![CDATA[

Código

]]>

Exemplos

O exemplo a seguir mostra o uso de <![CDATA[...]]> para impedir que o analisador XML falhe ao analisar um arquivo .wsf:

```
<?XML version="1.0" ?>
<job>
<script language="VBScript">
<![CDATA[
Rem & eh um caractere reservado da linguagem XML
WScript.Echo "Olá " & "Mundo!"
]]>
</script>
</job>
```

Objetos do WSH

Um script do WSH pode fazer uso de qualquer objeto COM registrado na máquina em que estiver sendo executado, sendo que ele próprio fornece alguns objetos intrínsecos que oferecem funcionalidades interessantes. São eles:

Objeto Wscript

O objeto Wscript é o principal objeto fornecido pelo WSH. Através dele podem ser instanciados todos os demais objetos do WSH (*WshShell*, *WshNetwork*, *WshShortcut*, etc.) utilizando o método *CreateObject()*. Além disso, o objeto Wscript fornece métodos e propriedades relacionados ao próprio script.

Palavra-Chave	
ProgID	N/A.
CLSID	60254CA2-953b-11CF-8C96-00AA00B8708C.
Propriedades	
Application	Fornece a interface IDispatch para o objeto Wscript.
Arguments	Argumentos passados na linha de comando.
BuildVersion	Retorna a "build version" do arquivo executável do WSH.
FullName	Caminho completo para o executável WSH.
Interactive	Se True, o script será executado em modo interativo, caso False, o script será executado em segundo plano.
Name	Nome amigável do objeto Wscript (propriedade padrão).
Path	Nome da pasta onde reside o executável do WSH.
ScriptFullName	Caminho completo para o script que está sendo executado pelo WSH.
ScriptName	Nome do arquivo de script que está sendo executado pelo WSH.
StdErr	Retorna o fluxo de erro padrão.
StdIn	Retorna o fluxo de entrada padrão.
StdOut	Retorna o fluxo de saída padrão.
TimeOut	Especifica o tempo máximo em que o script poderá ser executado.
Version	Uma string contendo a versão do WSH.
Métodos	
CreateObject	Cria e estabelece uma conexão com um objeto.
DisconnectObject	Disconecta um objeto previamente conectado.
Echo	Exibe uma caixa de mensagem.
GetObject	Retorna um objeto de automação de arquivo.
Quit	Termina a execução do script com um código de erro específico.
Sleep	Interrompe a execução do script durante um período de tempo determinado.

Propriedades do objeto Wscript

Application

Fornece a interface IDispatch para o objeto Wscript. Utilizado para se fazer referência a uma instância do objeto Wscript.

Set *objWscript* **= Wscript.Application**

Arguments

Fornece uma coleção de objetos WshArguments; permite recuperar argumentos passados para um script do WSH.

Set *objArgs* **= Wscript.Arguments**

Exemplo

```
Rem Exibe todos os argumentos da linha de comando:
Set objArgs = Wscript.Arguments
For I = 0 to objArgs.Count - 1
    Wscript.Echo objArgs(I)
Next
```

BuildVersion

Retorna a "build version" do arquivo executável do WSH (cscript.exe ou wscript.exe) utilizado para executar o script.

lngBuildVersion **= Wscript.BuildVersion**

Exemplo

```
Rem Exibe a "build version" do arquivo wscript.exe:
lngBuildVersion = Wscript.BuildVersion
Wscript.Echo lngBuildVersion
```

FullName

Fornece uma string contendo o caminho completo para o executável do WSH.

strFullName **= Wscript.FullName**

Exemplo

```
Rem Exibe o caminho completo para o Wscript.exe:
strFullName = Wscript.FullName
Wscript.Echo strFullName
```

Interactive

Especifica se o script será executado em modo interativo (True), permitindo interação com o usuário, ou em segundo plano (False). Em segundo plano, métodos como Echo, InputBox ou MsgBox não funcionarão.

Wscript.Interactive = *True|False*

Exemplo

```
Rem Exibirá a frase "Modo Interativo"  numa caixa de mensagem, pois a
propriedade Interactive foi definida como True:
Wscript.Interactive = True
Wscript.Echo "Modo Interativo"
```

Name

Fornece uma string contendo o nome amigável do objeto WScript. Essa é a propriedade padrão.

strName **= Wscript.Name**

Exemplo

```
Rem Exibe o nome amigável do objeto Wscript:
strName = Wscript.Name
Wscript.Echo strName
```

Path

Fornece uma string contendo o nome da pasta onde o executável do WSH reside.

strPath **= Wscript.Path**

Exemplo

```
strPath = Wscript.Path
Wscript.Echo strPath
```

ScriptFullName

Fornece o caminho completo para o script que está sendo executado pelo WSH.

strScriptFullName **= Wscript.ScriptFullName**

Exemplo

```
strScriptFullName = Wscript.ScriptFullName
Wscript.Echo strScriptFullName
```

ScriptName

Fornece o nome do arquivo de script que está sendo executado pelo WSH.

strScriptName **= Wscript.ScriptName**

Exemplo

```
Rem Exibe o nome do script:
strScriptName = Wscript.ScriptName
Wscript.Echo strScriptName
```

StdErr

Retorna um objeto TextStream, que permite escrever no fluxo de erro padrão. Pode ser utilizado apenas quando o executável do WSH é o cscript.exe.

Set *objStdErr* **= Wscript.StdErr**

StdIn

Retorna um objeto TextStream, que permite ler texto a partir do fluxo de entrada padrão. Pode ser utilizado apenas quando o executável do WSH é o cscript.exe.

Set *objStdIn* **= Wscript.StdIn**

StdErr

Retorna um objeto TextStream, que permite escrever no fluxo de saída padrão. Pode ser utilizado apenas quando o executável do WSH é o cscript.exe.

Set *objStdOut* **= Wscript.StdOut**

TimeOut

Especifica o tempo máximo (em segundos) que um script terá para ser executado. Quando esse limite de tempo for atingido, o script suspenderá automaticamente sua execução.

Wscript.TimeOut = *lngSegundos*

Version

Fornece uma string com a versão do Windows Script que está sendo executado.

strVersão **= Wscript.Version**

Métodos do objeto Wscript

ConnectObject

Relaciona um evento de um objeto a uma rotina dentro do script que será executada quando da ocorrência desse evento.

Wscript.ConnectObject *obj* [,*prefixo*])

Parâmetro	Descrição
obj	Nome do objeto, definido previamente pela utilização do método CreateObject().
prefixo	Prefixo que, concatenado ao caractere "_" e ao nome do evento gerado pelo objeto, forma o nome de um procedimento dentro do script que será chamado quando o evento ocorrer.

Exemplo

```
Rem Cria uma instância do objeto MeuObjeto e relaciona o evento OnBegin à
    rotina Evento_OnBegin:
Set obj = Wscript.CreateObject("MeuObjeto")
Wscript.ConnectObject obj, "Evento"
Sub Evento_OnBegin
        Wscript.Echo "Um objeto MeuObjeto foi inicializado!"
End Sub
```

CreateObject

Define um objeto e estabelece uma conexão com ele. Muitos objetos (como o *WshShell*, o *WshNetwork*, etc.) dependem diretamente da sua utilização para serem instanciados. Substitui a tag <*object*> da XML.

Set *obj* **= Wscript.CreateObject**(*progID* [,*prefixo*])

Parâmetro	Descrição
progID	Tipo do objeto a ser definido.
prefixo	Chama um procedimento dentro do script cujo nome é formado por *prefixo* mais o caractere "_", mais o nome de um evento gerado pelo objeto.

Exemplos

```
Rem Criando um objeto chamado WshShell:
Dim WshShell
Set WshShell = Wscript.CreateObject("Wscript.Shell")
```

DisconnectObject

Disconecta um objeto previamente conectado.

Wscript.DisconnectObject *obj*

Echo

Exibe uma caixa de mensagem informativa na tela.

Wscript.Echo *msg*

Exemplo

```
Rem Exibe uma caixa de mensagem com os dizeres "WSH é quente!":
Wscript.Echo "WSH é quente!"
```

GetObject

Retorna um objeto de automação de um arquivo. Pode ser usado para ler, alterar ou inserir dados em arquivos que aceitem a automação OLE, tais como documentos do Word, planilhas do Excel ou bancos de dados do Access. Nestes casos, o método *GetObject()* primeiro chama o aplicativo associado ao formato do arquivo.

Set *obj* **= GetObject(***arquivo,* [, *progID*][, *prefixo*])

Parâmetro	Descrição
arquivo	Arquivo de onde será retornado o objeto de automação.
progID	String contendo o tipo do objeto a ser retornado.
prefix	Semelhante ao parâmetro de mesmo nome do método *CreateObject()*.

Exemplo

```
Rem Abre um arquivo do Microsoft Excel chamado exemplo.xls:
dim obj
Set obj = GetObject("C:\exemplo.xls")
```

Quit

Encerra a execução do script com o código de erro correspondente.

Wscript.Quit [*intCodigoDeErro*]

Parâmetro	Descrição
intCodigoDeErro	Valor será retornado no final da execução do processo. Caso omitido, *Quit* retornará 0.

Sleep

Interrompe a execução do script durante um determinado período de tempo.

Wscript.Sleep *período*

Parâmetro	Descrição
período	Período de tempo (em milissegundos) em que a execução do script estará suspensa.

Exemplo

```
Rem A execução do script será suspensa por dez segundos:
Wscript.Echo "Script inicializado!"
Wscript.Sleep 10000
Wscript.Echo "Script finalizado!"
```

Objeto WshArguments

O objeto *WshArguments* retorna uma coleção contendo todos os argumentos passados para um script do WSH. Não é explicitamente acessado, sendo instanciado quando se faz uso da propriedade *Arguments* dos objetos *Wscript* ou *WshShortcut*.

Palavra-Chave

ProgID	N/A.
CLSID	60254CA4-953b-11CF-8C96-00AA00B8708C.

Propriedades

Item	Matriz contendo os argumentos passados na linha de comando para o script.
Count	Número de argumentos da linha de comando.
lenght	Número de argumentos da linha de comando (JScript).

Propriedades do objeto WshArguments

Item

A propriedade *Item* é a propriedade padrão do objeto *WshArguments*. É uma matriz que contém os argumentos passados na linha de comando para o script, indexados a partir de 0 (ou seja, o primeiro item tem índice 0, o segundo tem índice 1 e assim por diante).

Set *objArgs* **= Wscript.Arguments**

strArgs **=** *objArgs(índice)*

Exemplo

```
Rem Recupera o segundo argumento passado para o script na linha de comando:
Set objArgs = Wscript.Arguments
strArgs = objArgs(1)
Wscript.Echo strArgs
```

Count

Retorna o número de argumentos passados para o script na linha de comando.

Set *objArgs* **= Wscript.Arguments**

intArgs **=** *objArgs*.**Count**

lenght

A propriedade *lenght* é em tudo semelhante à propriedade *Count*. Usada para efeitos de compatibilidade com a linguagem JScript.

Objeto WshShell

O objeto *WshShell* permite que os scripts do WSH interajam diretamente com o sistema operacional.

Palavra-Chave	
ProgID	Wscript.Shell.
CLSID	F935DC22-1CF0-11d0-ADB9-00C04FD58A0B.
Propriedades	
Environment	Retorna um objeto *WshEnvironment*.
SpecialFolders	Fornece acesso às pastas especiais do Windows.
Métodos	
AppActivate	Ativa a janela de um aplicativo.
CreateShortcut	Cria e retorna um objeto *WshShortcut* ou *WshURLShortcut*.
ExpandEnvironmentStrings	Expande uma variável de ambiente do tipo PROCESS e retorna a string correspondente.
LogEvent	Salva informações no log do Windows NT ou no arquivo WSH.log.
Popup	Exibe uma caixa de mensagem.
RegDelete	Exclui uma chave ou valor do registro do Windows.
RegRead	Lê uma chave ou valor do registro do Windows.
RegWrite	Escreve chaves ou valores no registro do Windows.
Run	Cria um novo processo que executa um programa do DOS ou do Windows.
SendKeys	Envia uma tecla ou conjunto de teclas para o sistema operacional.

Propriedades do objeto WshShell

Environment

Retorna um objeto *WshEnvironment*. O tipo do objeto retornado, no ambiente NT, pode ser SYSTEM, USER, VOLATILE ou PROCESS. No ambiente Windows 9x, apenas o tipo PROCESS é suportado.

Set *WshShell* = **Wscript.CreateObject("Wscript.Shell")**

Set *objEnvironment* **=** *WshShell*.**Environment(***[strTipo]***)**

strVar **=** *objEnvironment(strVariavelDoAmbiente)*

Parâmetro	Descrição
strTipo	Tipo do objeto retornado. Pode ser SYSTEM, USER, VOLATILE ou PROCESS no Windows NT e apenas PROCESS no Windows 9x. Retornará SYSTEM no Windows NT e PROCESS no Windows 9x, caso não tenha sido especificado.

As principais variáveis de ambiente são:

Nome	Significado
NUMBER_OF_PROCESSORS	Número de processadores.
PROCESSOR_ARCHITECTURE	Tipo do processador.
PROCESSOR_IDENTIFIER	ID do processador da máquina.
PROCESSOR_LEVEL	Nível do processador da máquina.
PROCESSOR_REVISION	Versão do processador.
OS	Sistema operacional.
COMSPEC	Executável para a linha de comando.
HOMEDRIVE	Drive local primário; na maioria das vezes é C:\.
HOMEPATH	Diretório padrão para os usuários.
PATH	Variável PATH (a mesma encontrada no arquivo Autoexec.bat).
PATHEXT	Extensões de arquivos executáveis (.com, .exe, .bat, etc.).
PROMPT	Prompt da linha de comando.
SYSTEMDRIVE	Drive local onde se encontra a pasta do sistema.
SYSTEMROOT	Pasta do sistema.
WINDIR	Semelhante a SYSTEMROOT.
TMP	Pasta onde são armazenados os arquivos temporários.
TEMP	Semelhante a TMP.

Exemplo

```
Rem Obtém o nome do Sistema Operacional:
Set WshShell = Wscript.CreateObject("Wscript.Shell")
Set objEnvironment = WshShell.Environment("SYSTEM")
strVar = objEnvironment("OS")
Wscript.Echo strVar
```

SpecialFolders

Fornece um objeto, o *WshSpecialFolders*, através do qual é possível obter-se a localização de pastas especiais do Windows, como a pasta Desktop, por exemplo.

Set *WshShell =* **Wscript.CreateObject("Wscript.Shell")**

strVar **=** *WshShell.***SpecialFolders(***strNomeDaPasta***)**

Parâmetro	Descrição
strNomeDaPasta	Nome de uma das pastas especiais do Windows.

Pastas Especiais do Windows:

AllUsersDesktop
AllUsersStartMenu
AllUsersPrograms
AllUsersStartup
Desktop
Favorites
Fonts
MyDocuments
NetHood
PrintHood
Programs
Recent
SentTo
StartMenu
Startup
Templates

Exemplo

```
Rem Retorna o caminho completo para a pasta Meus Documentos do usuário atual:
Set WshShell = Wscript.CreateObject("Wscript.Shell")
strVar = WshShell.SpecialFolders("MyDocuments")
Wscript.Echo strVar
```

Métodos do objeto WshShell

AppActivate

Ativa a janela de um aplicativo em execução.

Set *WshShell =* **Wscript.CreateObject("Wscript.Shell")**

*WshShell.***AppActivate** *strTítulo*

Parâmetro	Descrição
strTítulo	Nome que aparece na barra de título do aplicativo sendo executado.

Exemplo

```
Rem Escreve "Olá Mundo!" no Notepad:
Set WshShell = CreateObject("WScript.Shell")
WshShell.Run "notepad"
WScript.Sleep 200
WshShell.AppActivate "Notepad"
WScript.Sleep 200
WshShell.SendKeys "Olá Mundo!"
WScript.Sleep 2000
```

CreateShortcut

Retorna dois tipos de objeto: um objeto *WshShortcut* para um atalho comum ou um objeto *WshURLShortcut* para um atalho que aponta para uma URL.

Set *WshShell* = **Wscript.CreateObject("Wscript.Shell")**

Set *objShortcut* **=**
WshShell.**CreateShortcut(***strNomeDoAtalho***)**

Parâmetro	Descrição
strNomeDoAtalho	Nome do atalho a ser criado na área de trabalho do usuário ou em outra localização. Se a extensão for .lnk, será criado um objeto *WshShortcut*. Se for .url, será criado um objeto *WshURLShortcut*.

ExpandEnvironmentStrings

Expande uma variável de ambiente do tipo PROCESS.

Set *WshShell* = **Wscript.CreateObject("Wscript.Shell")**

strVar **=** *WshShell*.**ExpandEnvironmentStrings(***string***)**

Parâmetro	Descrição
string	Uma string representando uma variável de ambiente contida entre dois caracteres de percentagem (%).

Exemplo

```
Rem Exibe o nome do sistema operacional:
Set WshShell = Wscript.CreateObject("Wscript.Shell")
strVar = WshShell.ExpandEnvironmentStrings("%OS%")
Wscript.Echo strVar
```

LogEvent

Salva informações no log do Windows NT ou no arquivo WSH.log (quando o script for executado no Windows 9x).

Set *WshShell* = **Wscript.CreateObject("Wscript.Shell")**

WshShell.**LogEvent** *intTipo, strMsg [,strAlvo]*

Parâmetro	Descrição
intTipo	Um inteiro representando o tipo (status) do evento ocorrido.
0	SUCESS
1	ERROR
2	WARNING
4	INFORMATION
8	AUDIT_SUCESS
16	AUDIT_FAILURE
strMsg	Uma mensagem descrevendo o evento ocorrido.
strAlvo	Nome do sistema onde o evento será salvo. Se nenhum valor for especificado, o WSH assumirá o sistema local. Válido apenas para o Windows NT.

Popup

Mostra uma caixa de mensagem na tela do computador. Bastante semelhante à função *MsgBox* do *VBscript*.

Set *WshShell* = **Wscript.CreateObject("Wscript.Shell")**

intBotão **=** *WshShell.***Popup(***strMsg [,intSegundos] [,strTítulo] [,intTipo]* **)**

Parâmetro	Descrição
strMsg	Mensagem a ser apresentada na caixa de mensagem.
intSegundos	Tempo em que a caixa de mensagem ficará na tela antes de ser automaticamente fechada.
strTítulo	Texto da barra de título da caixa de mensagem. Se omitido, será exibido "Windows Script Host".
intTipo	Valor obtido pela combinação (soma) dos valores de constantes referentes aos tipos de botões e de ícones e que determinam a aparência da caixa de mensagem.
Botões	
0	Botão OK.
1	Botões OK e Cancelar.
2	Botões Abortar, Repetir e Ignorar.
3	Botões Sim, Não e Cancelar.
4	Botões Sim e Não.
5	Botões Repetir e Cancelar.
Ícones	
16	Exibe um "X" dentro de um círculo vermelho. Significa a ocorrência de um erro.
32	Exibe um ponto de interrogação. Normalmente usado para perguntas.
48	Exibe um ponto de exclamação. Normalmente usado para advertências.
64	Exibe a letra "I". Usado quando a caixa de mensagem traz informações para o usuário.

O método *Popup* sempre retorna um inteiro, representando o botão que foi clicado. Abaixo, uma lista dos valores que podem ser retornados pelo método:

Valores	Botões
1	OK.
2	Cancelar.
3	Abortar.
4	Repetir.
5	Ignorar.
6	Sim.
7	Não.

RegDelete

Exclui uma chave ou valor do registro do Windows.

Set *WshShell* = **Wscript.CreateObject("Wscript.Shell")**

WshShell.**RegDelete** *strVar*

Parâmetro	Descrição
strVar	O valor ou chave do registro a ser excluído. O método *RegDelete* entende que se o parâmetro passado terminar com um caractere "\", deverá excluir uma chave, caso contrário excluirá um valor.

Exemplo

```
Rem Exclui o valor Y da chave X e depois a própria chave X de
HKEY_LOCAL_MACHINE\Software:
Set WshShell = Wscript.CreateObject("Wscript.Shell")
WshShell.RegDelete "HKLM\Software\X\Y"
WshShell.RegDelete "HKLM\Software\X\"
```

Abaixo, as denominações abreviadas e completas das chaves-raiz do registro do Windows:

Abreviação	Chaves-Raiz
HKCU	HKEY_CURRENT_USER.
HKLM	HKEY_LOCAL_MACHINE.
HKCR	HKEY_CLASSES_ROOT.
	HKEY_USERS
	HKEY_CURRENT_CONFIG

Esses valores das chaves-raiz servem também para os métodos *RegRead* e *RegWrite*, que veremos logo a seguir.

RegRead

Lê dados de uma chave ou valor do registro do Windows. Tem uma limitação: só é capaz de ler dados dos tipos REG_SZ, REG_EXPAND_SZ, REG_DWORD, REG_BINARY e REG_MULTI_SZ. Se for outro o tipo do dado a ser lido, *RegRead* retornará DISP_E_TYPEMISMATCH.

Set *WshShell* = **Wscript.CreateObject("Wscript.Shell")**

WshShell.**RegRead** *strVar*

Parâmetro	Descrição
strVar	O valor ou chave do registro a ser lido. O método *RegRead* entende que se o parâmetro passado terminar com um caractere "\", deverá ler o dado de uma chave, caso contrário lerá um valor.

Exemplo

```
Rem Lê o valor Y da chave X e depois lê a própria chave X de
HKEY_LOCAL_MACHINE\Software:
Set WshShell = Wscript.CreateObject("Wscript.Shell")
WshShell.RegRead "HKLM\Software\X\Y"
WshShell.RegRead "HKLM\Software\X\"
```

RegWrite

Com o método *RegWrite*, é possível para um script do WSH escrever dados em chaves ou valores do registro.

Set *WshShell* = **Wscript.CreateObject("Wscript.Shell")**

WshShell.**RegWrite** *strVar, strDado [,strTipo]*

Parâmetro	Descrição
strVar	O valor ou chave do registro onde se vai escrever um dado. O método *RegWrite*, como os anteriores, entende que se o parâmetro passado terminar com um caractere "\", deverá escrever em uma chave, caso contrário irá escrever um valor.
strDado	Dado a ser escrito.
strTipo	Tipo do dado a ser escrito. Pode ser REG_SZ, REG_EXPAND_SZ, REG_DWORD e REG_BINARY. Se tentar utilizar outro tipo, será retornado E_INVALIDARG.

Exemplo

```
Rem Escreve "WSH" no valor Y da chave X e depois na própria chave X de
HKEY_LOCAL_MACHINE\Software:
Set WshShell = Wscript.CreateObject("Wscript.Shell")
WshShell.RegWrite "HKLM\Software\X\Y", "WSH"
WshShell.RegWrite "HKLM\Software\X\", "WSH"
```

Run

Cria um novo processo que executa um arquivo executável (.exe, .com, .bat, etc.).

Set *WshShell* = **Wscript.CreateObject("Wscript.Shell")**

WshShell.**Run** *strComando [,intEstiloDeJanela] [,strAguardaRetorno]*

Parâmetro	Descrição
strComando	Caminho para o executável.
intEstiloDeJanela	Estilo da janela do executável durante sua execução pelo script.
0	Esconde a janela e ativa uma outra janela.
1	Ativa e exibe uma janela. Se a janela for minimizada ou maximizada, o sistema restaurará seu tamanho e posição originais. Uma aplicação deve utilizar essa opção quando estiver exibindo uma janela pela primeira vez.
2	Exibe a janela minimizada.
3	Exibe a janela maximizada.
4	Exibe a janela em sua posição e tamanho mais recentemente definidos.
5	Exibe a janela em sua posição e tamanho atuais.
6	Minimiza a janela especificada e ativa a próxima janela.

7	Exibe a janela minimizada, mantendo-a como janela ativa.
8	Exibe a janela em seu estado atual, mantendo-a como janela ativa.
9	Ativa e exibe uma janela. Se a janela for minimizada ou maximizada, o sistema restaurará seu tamanho e posição originais. Uma aplicação deve utilizar essa opção quando estiver restaurando uma janela minimizada.
10	Configura a exibição da janela de acordo com o programa que iniciou a aplicação.
strAguardaRetorno	Se não especificado ou *False*, retornará imediatamente ao script com um código de erro zero. Caso *True*, retornará qualquer código de erro da aplicação que estiver sendo executada.

Exemplo

```
Rem Inicia o Notepad:
Set WshShell = Wscript.CreateObject("WScript.Shell")
WshShell.Run "notepad"
```

SendKeys

Envia um caractere ou uma seqüência de caracteres (string) para o Windows, como se as teclas tivessem sido pressionadas no teclado.

Set *WshShell* = **Wscript.CreateObject("Wscript.Shell")**

WshShell.**SendKeys** *string*

Parâmetro	Descrição
string	Caractere ou seqüência de caracteres que devem ser enviados para o Windows. Para repetir uma seqüência várias vezes, utilize a forma {string *número de repetições*}. Por exemplo, {R 10} enviaria o caractere "R" dez vezes.

Alguns caracteres especiais, como a tecla de espaço ou a tecla Enter, por exemplo, que não são exibidos quando suas teclas correspondentes são pressionadas, necessitam de uma maneira especial para poderem ser utilizados pelo método *SendKeys*. Essa forma especial, na maioria dos casos, consiste em colocar o código delas entre chaves ({}).

Caracteres Especiais	Código
BACKSPACE	{BACKSPACE}, {BS} ou {BKSP}
BREAK	{BREAK}
CAPS LOCK	{CAPSLOCK}
DELETE	{DELETE} ou {DEL}
END	{END}
ENTER	{ENTER} ou ~
ESC	{ESC}
HELP	{HELP}
HOME	{HOME}

INSERT	{INSERT} ou {INS}
NUM LOCK	{NUMLOCK}
PAGE DOWN	{PGDN}
PAGE UP	{PGUP}
PRINT SCREEN	{PRTSC}
SCROLL LOCK	{SCROLLLOCK}
SETA P/ BAIXO	{DOWN}
SETA P/ DIREITA	{RIGHT}
SETA P/ ESQUERDA	{LEFT}
SETA P/ CIMA	{UP}
TAB	{TAB}
F1	{F1}
F2	{F2}
F3	{F3}
F4	{F4}
F5	{F5}
F6	{F6}
F7	{F7}
F8	{F8}
F9	{F9}
F10	{F10}
F11	{F11}
F12	{F12}
SHIFT	+
CTRL	^
ALT	%

Os caracteres de soma (+), percentagem (%), acento circunflexo (^), parênteses (( )) e til (~) também precisam estar entre chaves para poderem ser utilizados pelo método *SendKeys*.

Exemplo

```
Rem Escreve "Olá Mundo!" no Notepad:
Set WshShell = CreateObject("WScript.Shell")
WshShell.Run "notepad"
WScript.Sleep 200
WshShell.AppActivate "Notepad"
WScript.Sleep 200
WshShell.SendKeys "Olá Mundo!"
WScript.Sleep 2000
```

Objeto WshNetwork

O objeto *WshNetwork* permite aos scripts do WSH acessarem os recursos de rede do Windows.

Palavra-Chave

ProgID	Wscript.Network.
CLSID	F935DC26-1CF0-11d0-ADB9-00C04FD58A0B.

Propriedades

ComputerName	Uma string contendo o nome do computador.
UserDomain	Uma string contendo o nome do domínio do usuário.
UserName	Uma string contendo o nome do usuário.

Métodos

AddPrinterConnection	Mapeia uma impressora remota para um nome de recurso local.
AddWindowsPrinterConnection	Adiciona uma conexão de impressora ao Windows.
EnumNetworkDrives	Enumera os drives de rede atualmente mapeados.
EnumPrinterConnections	Enumera os mapeamentos de impressoras de rede atuais.
MapNetworkDrive	Mapeia um compartilhamento com um recurso local.
RemoveNetworkDrive	Remove um mapeamento de rede.
RemovePrinterConnection	Remove um mapeamento para uma impressora.
SetDefaultPrinter	Determina uma impressora como sendo a impressora padrão.

Propriedades do objeto WshNetwork

ComputerName

Fornece uma string contendo o nome do computador onde o script está sendo executado.

Set *WshNetwork* **=**
Wscript.CreateObject("Wscript.Network")

strNomeComputador **=** *WshNetwork*.**ComputerName**

Exemplo

```
Rem Obtém o nome do computador:
Set WshNetwork = Wscript.CreateObject("Wscript.Network")
strComputerName = WshNetwork.ComputerName
Wscript.Echo strComputerName
```

UserDomain

Fornece uma string contendo o nome do domínio onde está definida a conta do usuário atualmente conectado.

Set *WshNetwork* **=**
Wscript.CreateObject("Wscript.Network")

*strUserDomain = WshNetwork.***UserDomain**

Exemplo

```
Rem Obtém o nome do domínio:
Set WshNetwork = Wscript.CreateObject("Wscript.Network")
strUserDomain = WshNetwork.UserDomain
Wscript.Echo strUserDomain
```

UserName

Fornece uma string contendo o nome do usuário atualmente conectado à máquina.

Set *WshNetwork* **=**
Wscript.CreateObject("Wscript.Network")

*strUserName = WshNetwork.***UserName**

Exemplo

```
Rem Obtém o nome do usuário atual:
Set WshNetwork = Wscript.CreateObject("Wscript.Network")
strUserName = WshNetwork.UserName
Wscript.Echo strUserName
```

Métodos do objeto WshNetwork

AddPrinterConnection

Mapeia uma impressora remota para um recurso local do tipo "LPT1:".

Set *WshNetwork* **=**
Wscript.CreateObject("Wscript.Network")

*WshNetwork.***AddPrinterConnection** *strRecursoLocal, strImpressoraRemota*

Parâmetro	Descrição
strRecursoLocal	Nome do recurso local para o qual será mapeada a impressora. Exemplo: "LPT1:".
strImpressoraRemota	Caminho para a impressora remota, do tipo "\\nomedoservidor\nomedaimpressoracompartilhada".

Exemplo

```
Rem Mapeia a impressora Y no servidor X para o recurso local LPT3:
Set WshNetwork = Wscript.CreateObject("Wscript.Network")
WshNetwork.AddPrinterConnection "LPT3:", "\\X\Y"
```

AddWindowsPrinterConnection

Adiciona uma conexão de impressora ao Windows.

Set *WshNetwork* **=**
Wscript.CreateObject("Wscript.Network")

*WshNetwork.***AddWindowsPrinterConnection**
strImpressora, strDriver [,strPortaDeImpressora]

Parâmetro	Descrição
strImpressora	Caminho para a impressora remota, do tipo "\\nomedoservidor\nomedaimpressoracompartilhada".
strDriver	Driver utilizado pela impressora. Só poderá ser usado quando o script for executado no Windows 9x. No Windows NT/2000 será ignorado pelo WSH.
strPortaDeImpressora	Porta a ser utilizada pela impressora. A porta padrão é a "LPT1:". Também só pode ser utilizado no Windows 9x, sendo ignorado no Windows NT/2000.

Exemplo

```
Rem Adiciona a impressora Y localizada no servidor X ao Windows:
Set WshNetwork = Wscript.CreateObject("Wscript.Network")
WshNetwork..AddWindowsPrinterConnection "\\X\Y"
```

EnumNetworkDrives

Enumera todos os mapeamentos de rede da máquina em uma matriz, instanciando um objeto *WshCollection*.

Set *WshNetwork* **=**
Wscript.CreateObject("Wscript.Network")

Set *objDrives* **=** *WshNetwork.***EnumNetworkDrives**

EnumPrinterConnections

Enumera os mapeamentos de impressoras da máquina em uma matriz, instanciando um objeto *WshCollection*.

Set *WshNetwork* **=**
Wscript.CreateObject("Wscript.Network")

Set *objImpressoras* **=**
*WshNetwork.***EnumPrinterConnections**

MapNetworkDrive

Mapeia um compartilhamento em um servidor remoto para um recurso local, por exemplo, "D:".

Set *WshNetwork* **=**
Wscript.CreateObject("Wscript.Network")

*WshNetwork.***MapNetworkDrive** *strRecursoLocal, strCompartilhamento [,fAtualizaPerfil] [,strUsuário] [,strSenha]*

Parâmetro	Descrição
strRecursoLocal	Recurso local (unidade de disco de rede) para o qual o compartilhamento será mapeado.
strCompartilhamento	Nome do compartilhamento, do tipo "\\nomedoservidor\pastacompartilhada", a ser mapeado.
fAtualizaPerfil	Se *True,* salva o mapeamento no perfil do usuário.
strUsuário	Nome do usuário que se quer utilizar para fazer o mapeamento (caso não seja o usuário atualmente conectado).
strSenha	Senha do usuário que se quer utilizar para fazer o mapeamento (caso não seja o usuário atualmente conectado).

Exemplo

```
Rem Faz o mapeamento do compartilhamento Y no servidor X para a unidade de
disco de rede "Z:":
Set WshNetwork = Wscript.CreateObject("Wscript.Network")
WshNetwork.MapNetworkDrive "Z:", "\\X\Y"
```

RemoveNetworkDrive

Remove um mapeamento.

Set *WshNetwork* **=**
Wscript.CreateObject("Wscript.Network")

WshNetwork.**RemoveNetworkDrive** *strUnidadeDeDisco [,fForçado]*

Parâmetro	Descrição
strUnidadeDeDisco	Unidade de disco de rede utilizada pelo mapeamento que se pretende desfazer.
fForçado	Se *True*, desfaz o mapeamento independentemente dele estar ou não sendo utilizado no momento.

Exemplo

```
Rem Desfaz o mapeamento de rede do exemplo anterior:
Set WshNetwork = Wscript.CreateObject("Wscript.Network")
WshNetwork.RemoveNetworkDrive "Z:"
```

RemovePrinterConnection

Semelhante ao método *RemoveNetworkDrive*, o método *RemovePrinterConnection* remove um mapeamento feito para uma impressora remota.

Set *WshNetwork* **=**
Wscript.CreateObject("Wscript.Network")

WshNetwork.**RemovePrinterConnection** *strImpressora*

Parâmetro	Descrição
strImpressora	Impressora cujo mapeamento se pretende desfazer.

Exemplo

```
Rem Desfaz o mapeamento para a impressora em "LPT1:":
Set WshNetwork = Wscript.CreateObject("Wscript.Network")
WshNetwork.RemovePrinterConnection "LPT1:"
```

SetDefaultPrinter

Define uma impressora como sendo a impressora padrão do Windows.

Set *WshNetwork* **=**
Wscript.CreateObject("Wscript.Network")

WshNetwork.**SetDefaultPrinter** *strImpressora*

Parâmetro	Descrição
strImpressora	Nome da impressora que se quer definir como a impressora padrão do Windows. Note que apenas nomes de impressora, não nomes de recursos locais tais como "LPT1:", são aceitos como argumento.

Exemplo

```
Rem Define a impressora \\X\Y como a impressora padrão do Windows:
Set WshNetwork = Wscript.CreateObject("Wscript.Network")
WshNetwork.SetDefaultPrinter "\\X\Y"
```

Objeto WshCollection

O objeto *WshCollection* não é acessado explicitamente. É instanciado quando da utilização dos métodos *EnumNetworkDrives* e *EnumPrinterConnections* do objeto *WshNetwork*.

Palavra-Chave

ProgID	N/A.
CLSID	F935DC24-1CF0-11d0-ADB9-00C04FD58A0B.

Propriedades

Item	Matriz contendo os elementos enumerados.
Count	O número de elementos enumerados.
lenght	O número de elementos enumerados (JScript).

Propriedades do objeto WshCollection

Item

É a propriedade padrão do objeto *WshCollection*. É uma matriz que contém os elementos enumerados, indexados a partir de 0 (ou seja, o primeiro item tem índice 0, o segundo tem índice 1 e assim por diante).

Set *WshNetwork* **=**
Wscript.CreateObject("Wscript.Network")

Set *objEnum* **=** *WshNetwork*.**EnumNetworkDrives**

ou

Set *objEnum* **=** *WshNetwork*.**EnumPrinterConnections**

strItem **=** *objEnum (índice)*

Count

Fornece o número de elementos enumerados.

Set *WshNetwork* **=**
Wscript.CreateObject("Wscript.Network")

Set *objEnum* **=** *WshNetwork*.**EnumNetworkDrives**

ou

Set *objEnum* **=** *WshNetwork*.**EnumPrinterConnections**

intItens **=** *objEnum*.**Count**

lenght

A propriedade *lenght* é em tudo semelhante à propriedade *Count*. Usada para efeitos de compatibilidade com a linguagem JScript.

Objeto WshEnvironment

O objeto *WshEnvironment*, assim como o *WshCollection*, não é explicitamente acessado. Para usar suas três propriedades e seu único método, em primeiro lugar é preciso instanciá-lo utilizando a propriedade *Environment* do objeto *WshShell*.

Palavra-Chave	Descrição
ProgID	N/A.
CLSID	N/A.
Propriedades	
Item	Atribui ou obtém o valor de uma variável de ambiente.
Count	O número de elementos enumerados.
lenght	O número de elementos enumerados (JScript).
Método	
Remove	Remove uma variável de ambiente.

Propriedades do objeto WshEnvironment

Item

É a propriedade padrão do objeto *WshEnvironment*. Através dela é possível atribuir ou obter o valor de uma variável de ambiente. O uso da palavra *Item* é opcional.

Set *WshShell* = **Wscript.CreateObject("Wscript.Shell")**

Set *objEnvironment* **=** *WshShell*.**Environment(***[strTipo]***)**

strVar **=** *objEnvironment*.**Item(***strVariavelDoAmbiente***)**

Parâmetro	Descrição
strTipo	Tipo do objeto retornado. Pode ser SYSTEM, USER, VOLATILE ou PROCESS no Windows NT e apenas PROCESS no Windows 9x. Retornará SYSTEM no Windows NT e PROCESS no Windows 9x, caso não seja especificado.

As principais variáveis de ambiente são:

Nome	Significado
NUMBER_OF_PROCESSORS	Número de processadores.
PROCESSOR_ARCHITECTURE	Tipo do processador.
PROCESSOR_IDENTIFIER	ID do processador da máquina.
PROCESSOR_LEVEL	Nível do processador da máquina.
PROCESSOR_REVISION	Versão do processador.
OS	Sistema operacional.
COMSPEC	Executável para a linha de comando.
HOMEDRIVE	Drive local primário; normalmente é C:\.
HOMEPATH	Diretório padrão para os usuários.
PATH	Variável PATH (a mesma encontrada no arquivo Autoexec.bat).
PATHEXT	Extensões de arquivos executáveis (.com, .exe, .bat, etc.).
PROMPT	Prompt da linha de comando.
SYSTEMDRIVE	Drive local onde encontra-se a pasta de sistema.
SYSTEMROOT	Pasta do sistema.
WINDIR	Semelhante a SYSTEMROOT.
TMP	Pasta onde são armazenados os arquivos temporários.
TEMP	Semelhante a TMP.

Exemplo

```
Rem Obtém o nome do Sistema Operacional:
Set WshShell = Wscript.CreateObject("Wscript.Shell")
Set objEnvironment = WshShell.Environment("SYSTEM")
strVar = objEnvironment.Item("OS")
Wscript.echo strVar
```

Count

Fornece o número de elementos enumerados.

Set *WshShell* = **Wscript.CreateObject("Wscript.Shell")**

Set *objEnvironment* **=** *WshShell*.**Environment(***[strTipo]***)**

intVar **=** *objEnvironment*.**Count**

lenght

A propriedade *lenght* é em tudo semelhante à propriedade *Count*. Usada para compatibilidade com a linguagem JScript.

Método do WshEnvironment

Remove

Exclui uma variável de ambiente.

Set *WshShell* = **Wscript.CreateObject("Wscript.Shell")**

Set *objEnvironment* **=**
WshShell.**Environment(***"VOLATILE"***)**

objEnvironment.**Remove** *strVariávelDeAmbiente*

Objeto WshShortcut

O objeto *WshShortcut* não é explicitamente acessado. Para poder trabalhar com ele, é necessário instanciá-lo usando o método *CreateShortcut()* do objeto *WshShell*. O objeto *WshShortcut* permite a criação e a manipulação de atalhos na área de trabalho do usuário.

Palavra-Chave	
ProgID	N/A.
CLSID	F935DC28-1CF0-11d0-ADB9-00C04FD58A0B.
Propriedades	
Arguments	Argumentos passados para o atalho.
Description	Uma descrição do atalho.
FullName	Uma string contendo o caminho completo para o atalho.
HotKey	Uma tecla ou combinação de teclas que aciona o atalho.
IconLocation	A localização do ícone de um atalho.
TargetPath	O arquivo alvo do atalho.
WindowsStyle	O estilo da janela do atalho.
WorkingDirectory	O diretório de trabalho de um atalho.
Métodos	
Save	Salva um atalho.

Propriedades do objeto WshShortcut

Arguments

Retorna um objeto *WshArguments*, contendo uma coleção de argumentos passados para o atalho.

Set *WshShell* = **Wscript.CreateObject("Wscript.Shell")**

Set *objShortcut* **=**
WshShell.**CreateShortcut(***strNomeDoAtalho***)**

Set *objArgs* **=** *objShortcut*.**Arguments**

Exemplo

```
Rem Exibe todos os argumentos passados para o atalho criado para o Notepad:
Set WshShell = Wscript.CreateObject("Wscript.Shell")
Set objShortcut = WshShell.CreateShortcut("c::\notepad.lnk")
Set objArgs = objShortcut..Arguments
For I = 0 to objArgs.Count - 1
    Wscript.Echo objArgs(I)
Next
```

Description

Fornece uma descrição do atalho.

Set *WshShell* = **Wscript.CreateObject("Wscript.Shell")**

Set *objShortcut* **=**
WshShell.**CreateShortcut(***strNomeDoAtalho***)**

objShortcut.**Description** = *string*

FullName

Fornece uma string contendo o caminho completo para o atalho.

Set *WshShell* = **Wscript.CreateObject("Wscript.Shell")**

Set *objShortcut* **=**
WshShell.**CreateShortcut(***strNomeDoAtalho***)**

strFullName **=** *objShortcut*.**FullName**

HotKey

Associa uma tecla ou combinação de teclas para acionar um atalho.

Set *WshShell* = **Wscript.CreateObject("Wscript.Shell")**

Set *objShortcut* **=**
WshShell.**CreateShortcut(***strNomeDoAtalho***)**

objShortcut.**HotKey** = *strHotKey*

Exemplo

```
Rem Associa as teclas "Ctrl", "Alt" e "N" ao atalho criado para o Notepad:
Set WshShell = Wscript.CreateObject("Wscript.Shell")
Set objShortcut = WshShell.CreateShortcut("c::\notepad.lnk")
objShortcut.Hotkey = "ALT + CTRL + N"
```

IconLocation

Fornece a localização do ícone de um atalho.

Set *WshShell* = **Wscript.CreateObject("Wscript.Shell")**

Set *objShortcut* **=**
WshShell.**CreateShortcut(***strNomeDoAtalho***)**

objShortcut.**IconLocation** = *strCaminhoDoìcone*

TargetPath

Fornece o caminho para o arquivo alvo do atalho.

Set *WshShell* = **Wscript.CreateObject("Wscript.Shell")**

Set *objShortcut* **=**
WshShell.**CreateShortcut(***strNomeDoAtalho***)**

objShortcut.**TargetPath** = *strAlvo*

Exemplo

```
Rem Define o caminho para o executável do Notepad:
Set WshShell = Wscript.CreateObject("Wscript.Shell")
Set objShortcut = WshShell.CreateShortcut("c::\notepad.lnk")
objShortcut.TargetPath = "c:\winnt\notepad.exe"
```

WindowStyle

Fornece o estilo de janela de um atalho.

Set *WshShell* = **Wscript.CreateObject("Wscript.Shell")**

Set *objShortcut* **=**
WshShell.**CreateShortcut(***strNomeDoAtalho***)**

objShortcut.**WindowStyle** = *intEstiloDaJanela*

Os valores possíveis para *intEstiloDaJanela* são:

Estilos de Janela	Descrição
1	Ativa e exibe a janela como do tipo *Normal*.
3	Ativa e exibe a janela maximizada.
1	Minimiza a janela e exibe a próxima janela.

Exemplo

```
Rem Determina que o Notepad seja iniciado com a janela minimizada:
Set WshShell = Wscript.CreateObject("Wscript.Shell")
Set objShortcut = WshShell.CreateShortcut("c::\notepad.lnk")
objShortcut.WindowStyle = 7
```

WorkingDirectory

Fornece a pasta de trabalho do atalho.

Set *WshShell* = **Wscript.CreateObject("Wscript.Shell')**

Set *objShortcut* **=**
WshShell.**CreateShortcut(***strNomeDoAtalho***)**

objShortcut.**WorkingDirectory** = *strPastaDeTrabalho*

Método do objeto WshShortcut

Save

Salva o atalho na localização especificada no argumento do método *CreateShortcut()* do objeto *WshShell*.

Set *WshShell* = **Wscript.CreateObject("Wscript.Shell')**

Set *objShortcut* **=** *WshShell*.**CreateShortcut(***strNomeAtalho***)**

objShortcut.**Save**

Exemplo

```
Rem Salva um atalho para o Notepad na unidade C:
Set WshShell = Wscript.CreateObject("Wscript.Shell")
Set objShortcut = WshShell.CreateShortcut("c::\notepad.lnk")
objShortcut.TargetPath = "c:\winnt\notepad.exe"
objShortcut.WindowStyle = 1
objShortcut.Description = "Atalho para o Notepad!"
objShortcut.WorkingDirectory = "c:\winnt"
objShortcut.Save
```

Objeto WshSpecialFolders

O objeto *WshSpecialFolders* permite obter o caminho para as pastas especiais do Windows. É instanciado quando da utilização da propriedade *SpecialFolders* do objeto *WshShell*.

Palavra-Chave

ProgID	N/A.
CLSID	N/A.

Propriedades

Item	Matriz de elementos contendo o caminho completo para as pastas especiais do Windows.
Count	O número de elementos enumerados.
lenght	O número de elementos enumerados (JScript).

Propriedades do objeto WshSpecialFolders

Item

A propriedade *Item* é a propriedade padrão do objeto *WshSpecialFolders*. É uma matriz de elementos que contém o caminho completo para as pastas especiais do Windows. Se a pasta não existir, NULL será retornado. O uso da palavra reservada *Item* é opcional, o que significa dizer desnecessário.

Set *WshShell* = **Wscript.CreateObject("Wscript.Shell")**

strVar **=** *WshShell*.**SpecialFolders.Item(***strNomeDaPasta***)**

Parâmetro	Descrição
strNomeDaPasta	Especifica o nome de uma das pastas especiais do Windows.

Count

Fornece o número de elementos enumerados.

Set *WshShell* = **Wscript.CreateObject("Wscript.Shell")**

intVar **=** *WshShell*.**SpecialFolders.Count**

lenght

A propriedade *lenght* é em tudo semelhante à propriedade *Count*. Usada para efeitos de compatibilidade com a linguagem JScript.

Objeto WshURLShortcut

O objeto *WshURLShortcut* também é instanciado a partir da utilização do método *CreateShortcut()* do objeto *WshShell*. Permite criar e manipular atalhos para URLs.

Palavra-Chave	Descrição
ProgID	N/A.
CLSID	N/A.
Propriedades	
FullName	Uma string contendo o caminho completo para o atalho.
TargetPath	A URL alvo do atalho.
Métodos	
Save	Salva um atalho.

Propriedades do objeto WshURLShortcut

FullName

Fornece uma string contendo o caminho completo para o atalho.

```
Set WshShell = Wscript.CreateObject("Wscript.Shell")
Set objURLShortcut =
    WshShell.CreateShortcut(strNomeDoAtalho)
strFullName = objURLShortcut.FullName
```

TargetPath

Fornece o caminho para a URL alvo do atalho.

```
Set WshShell = Wscript.CreateObject("Wscript.Shell")
Set objURLShortcut =
    WshShell.CreateShortcut(strNomeDoAtalho)
objURLShortcut.TargetPath = strAlvo
```

Método do objeto WshURLShortcut

Save

Salva o atalho na localização especificada no argumento do método *CreateShortcut()* do objeto *WshShell*.

```
Set WshShell = Wscript.CreateObject("Wscript.Shell")
Set objURLShortcut =
    WshShell.CreateShortcut(strNomeDoAtalho)
objURLShortcut.Save
```

Objeto Dictionary

O objeto **Dictionary** não é um objeto intrínseco ao WSH, mas pode ser bastante útil em diversas situações. O objeto **Dictionary** permite armazenar pares nome/valor (referenciados como chave e item, respectivamente) em uma matriz. Cada item é associado a uma única chave.

Sintaxe para a instanciação do objeto:

Set ***obj*** **=** CreateObject**("Scripting.Dictionary")**

No exemplo abaixo é criado um objeto **Dictionary** chamado "frutas", ao qual adiciona-se chaves e valores. Ao final, o script exibirá uma caixa de mensagem com o item contido na chave "d".

Exemplo

```
Dim frutas
Set frutas = CreateObject("Scripting.Dictionary")
frutas.Add "a", "Laranja"
frutas.Add "b", "Maçã"
frutas.Add "c", "Morango"
frutas.Add "d", "Pêssego"
Wscript.Echo "A fruta contida na chave 'd' é: " & frutas.item("d")
```

Propriedade

CompareMode	Define ou retorna o modo utilizado na comparação de strings.
Count	Retorna o número de itens em um objeto Dictionary.
Item	Define ou retorna um item para uma chave especificada.
Key	Define ou retorna o valor de uma chave especificada.

Métodos

Add	Adiciona um par chave/item ao objeto Dictionary.
Exists	Booleano. Retornará True se a chave especificada existir.
Items	Retorna uma matriz contendo todos os itens de um objeto Dictionary.
Keys	Retorna uma matriz contendo todas as chaves de um objeto Dictionary.
Remove	Remove um par chave/item de um objeto Dictionary.
RemoveAll	Remove todos os pares chave/item de um objeto Dictionary.

CompareMode

A propriedade **CompareMode** define ou retorna o modo utilizado na comparação de strings em um objeto **Dictionary**. Disponível somente no VBScript.

Set ***obj*** **=** CreateObject**("Scripting.Dictionary")**

obj.CompareMode **[=*compara*]**

Argumento	Descrição
compara	Modo de comparação usado por funções como **StrComp**.
vbBinaryCompare=0	Comparação binária.
vbTextCompare=1	Comparação textual.

Count

A propriedade **Count** retorna o número de itens em um objeto **Dictionary**.

Set *obj* = CreateObject("Scripting.Dictionary")
obj.Count

Item

A propriedade **Item** define ou retorna um item para uma chave especificada em um objeto **Dictionary**.

Set ***obj*** **=** CreateObject**("Scripting.Dictionary")**

***obj*.**Item**(*chave*)[=*novoitem*]**

Argumento	Descrição
chave	Chave associada ao item que estiver sendo lido ou adicionado.
novoitem	Novo valor associado à chave. Usado somente com objeto **Dictionary**.

Key

A propriedade **Key** define ou retorna o valor de uma chave em um objeto **Dictionary**.

Set ***obj*** **=** CreateObject**("Scripting.Dictionary")**

obj.Key (***chave***) = ***novachave***

Argumento	Descrição
chave	Chave existente.
novachave	Novo valor da chave.

Add

O método **Add** adiciona um par chave/item a um objeto **Dictionary**.

Set ***obj*** **=** CreateObject**("Scripting.Dictionary")**

obj.Add ***chave*, *item***

Argumento	Descrição
chave	Chave associada ao item que estiver sendo adicionado.
item	Item a ser adicionado.

Exists

O método **Exists** retornará **True** se a chave especificada existir no objeto **Dictionary**; caso contrário, retornará **False**.

Set ***obj*** **=** CreateObject**(“Scripting.Dictionary”)**

obj.Exists **(*chave*)**

Argumento	Descrição
chave	Chave a ser pesquisada no objeto **Dictionary**.

Items

O método **Items** retorna uma matriz contendo todos os itens de um objeto **Dictionary**.

Set ***obj*** **=** CreateObject**(“Scripting.Dictionary”)**

obj.Items

Keys

O método **Keys** retorna uma matriz contendo todas as chaves de um objeto **Dictionary**.

Set ***obj*** **=** CreateObject**(“Scripting.Dictionary”)**

obj.Keys

Remove

O método **Remove** remove um par chave/item de um objeto **Dictionary**.

Set ***obj*** **=** CreateObject**(“Scripting.Dictionary”)**

obj.Remove **(*chave*)**

Argumento	Descrição
chave	Chave a ser removida do objeto **Dictionary**.

RemoveAll

O método **RemoveAll** remove todos os pares chave/item de um objeto **Dictionary**.

Set ***obj*** **=** CreateObject**(“Scripting.Dictionary”)**

obj.RemoveAll

FileSystemObject

O *FileSystemObject*, ou simplesmente FSO, está contido na biblioteca scrrun.dll. O FSO permite ao desenvolvedor interagir com o sistema de arquivos do Windows, inclusive fornecendo outros objetos e coleções para tarefas específicas, tais como listar os drives da máquina ou copiar o conteúdo de uma pasta para outra localização.

Objetos	
FileSystemObject	Objeto principal do FSO. Tem de ser instanciado antes de qualquer outro.
Drive	Fornece acesso às propriedades dos drives da máquina.
File	Fornece propriedades e métodos para manipulação de arquivos.
Folder	Fornece propriedades e métodos para manipulação de pastas.
TextStream	Permite acessar (criar, ler e escrever) arquivos de texto.
Coleções	
Drives	Uma lista de todos os drives da máquina.
Files	Uma lista dos arquivos contidos em uma pasta.
Folders	Uma lista das pastas contidas em outra pasta ou na raiz de um disco.

Com o FSO você poderá:

- desenvolver páginas ASP e
- criar scripts do WSH.

O primeiro passo antes de utilizar as propriedades, métodos, objetos e coleções fornecidos pelo FSO é instanciar o objeto principal, o *FileSystemObject*, da seguinte forma:

Set *fso* = CreateObject("Scripting.FileSystemObject")

Vamos ver então o que o objeto principal, o *FileSytemObject*, tem a nos oferecer.

Propriedade	
Drives	Retorna uma coleção Drives.
Métodos	
BuildPath	Anexa um nome a um caminho existente.
CopyFile	Copia um arquivo para um local especificado.
CopyFolder	Copia uma pasta para um local especificado.
CreateFolder	Cria uma pasta.
CreateTextFile	Cria um arquivo de texto.
DeleteFile	Exclui um arquivo.
DeleteFolder	Exclui uma pasta.
DriveExists	Retornará *True* se o drive especificado existir.
FileExists	Retornará *True* se o arquivo especificado existir.

FolderExists	Retornará *True* se a pasta especificada existir.
GetAbsolutePathName	Retorna o caminho completo para a pasta atual.
GetBaseName	Retorna o nome de um arquivo sem a extensão.
GetDrive	Retorna um objeto Drive.
GetDriveName	Retorna o nome de um drive.
GetExtensionName	Retorna a extensão de um arquivo.
GetFile	Retorna um objeto File.
GetFileName	Retorna o nome do arquivo em um caminho especificado.
GetFolder	Retorna um objeto Folder.
GetParentFolderName	Retorna a pasta onde está contida a última pasta ou arquivo de um caminho.
GetSpecialFolder	Retorna o caminho para uma pasta especial do Windows.
GetTempName	Retorna um nome de arquivo temporário.
MoveFile	Move um arquivo para outra localização.
MoveFolder	Move uma pasta e seu conteúdo para outra localização.
OpenTextFile	Abre um arquivo de texto.

Drives

Retorna uma coleção *Drives* onde estão contidos todos os drives da máquina.

Set *fso* **=CreateObject("Scripting.FileSystemObject")**

Set *myDrives* **=** *fso*.**Drives**

Exemplo

```
Rem Exibe todos os drives da máquina:
Set fso = CreateObject("Scripting.FileSystemObject")
Set myDrives  = fso.Drives
For Each drive in myDrives
        S = S + drive + "\" + chr(10)
Next
Wscript.Echo S
```

BuildPath

Anexa um nome a um caminho existente.

Set *fso* **=CreateObject("Scripting.FileSystemObject")**

novocaminho **=** *fso*.**BuildPath(***caminho, nome***)**

Parâmetro	Descrição
caminho	Caminho ao final do qual se quer adicionar um nome.
nome	Nome que se quer adicionar ao caminho existente.

Exemplo

```
Rem Adiciona o nome de arquivo "x.txt" ao caminho "c:\windows\":
Set fso = CreateObject("Scripting.FileSystemObject")
novocaminho = fso.BuildPath("c:\windows", "x.txt")
Wscript.Echo novocaminho
```

CopyFile

Copia um ou mais arquivos para um local especificado.

Set *fso* **= CreateObject("Scripting.FileSystemObject")**

fso.**CopyFile** *origem, destino [,fSubstituir]*

Parâmetro	Descrição
origem	Caminho completo para o(s) arquivo(s) a ser(em) copiado(s). Caracteres curinga (*) podem ser usados no lugar do nome dos arquivos.
destino	Nova localização do(s) arquivo(s).
fSubstituir	Se *True*, substituirá os arquivos existentes na nova localização que tenham o mesmo nome dos arquivos que estão sendo copiados.

Exemplo

```
Rem Copia o arquivo "c:\x.txt" para a pasta "c:\windows\":
Set fso = CreateObject("Scripting.FileSystemObject")
fso.CopyFile "c:\x.txt", "c:\windows\"
```

CopyFolder

Copia uma pasta para um local especificado.

Set *fso* **= CreateObject("Scripting.FileSystemObject")**

fso.**CopyFolder** *origem, destino [,fSubstituir]*

Parâmetro	Descrição
origem	Caminho completo para a pasta a ser copiada. Caracteres curinga (*) podem ser usados no lugar do nome da pasta.
destino	Nova localização da pasta.
fSubstituir	Se *True*, substituirá os arquivos homônimos caso haja uma pasta na nova localização com o mesmo nome da pasta a ser copiada.

Exemplo

```
Rem Copia a pasta "c:\teste\" para a pasta "c:\windows\":
Set fso = CreateObject("Scripting.FileSystemObject")
fso.CopyFolder "c:\teste", "c:\windows\"
```

CreateFolder

Cria uma pasta na localização especificada, retornando um objeto *Folder*.

Set *fso* **= CreateObject("Scripting.FileSystemObject")**

Set *f* **=** *fso*.**CreateFolder(***strNovaPasta***)**

Parâmetro	Descrição
strNovaPasta	Caminho completo para a nova pasta a ser criada. Será retornado um erro caso a pasta já exista.

Exemplo

```
Rem Cria a pasta "novapasta" em "c:\":
Set fso = CreateObject("Scripting.FileSystemObject")
Set f = fso.CreateFolder("c:\novapasta")
```

CreateTextFile

Cria um novo arquivo de texto na localização especificada, retornando um objeto *TextStream*.

Set *fso* **= CreateObject("Scripting.FileSystemObject")**

Set *f* **=** *fso*.**CreateTextFile(***strArquivoTexto [,fSubstituir] [,unicode]***)**

Parâmetro	Descrição
strArquivoTexto	Caminho completo para o arquivo de texto a ser criado.
fSubstituir	Se *True*, substituirá um arquivo homônimo pelo arquivo que estiver sendo criado. Se *False*, o arquivo original será mantido.
unicode	Se *True*, um arquivo Unicode será criado. Se *False* ou omitido, será criado um arquivo ASCII.

Exemplo

```
Rem Cria um arquivo chamado "x.txt" em "c:\":
Set fso = CreateObject("Scripting.FileSystemObject")
Set f = fso.CreateTextFile("c:\x.txt", true)
```

DeleteFile

Exclui um arquivo.

Set *fso* **= CreateObject("Scripting.FileSystemObject")**

fso.**DeleteFile** *strArquivo [,fForçado]*

Parâmetro	Descrição
strArquivo	Caminho para o arquivo a ser excluído.
fForçado	Booleano. Esse parâmetro terá de ser *True* se o arquivo a ser excluído for somente de leitura.

Exemplo

```
Rem Exclui o arquivo "x.txt" criado anteriormente:
Set fso = CreateObject("Scripting.FileSystemObject")
fso.DeleteFile "c:\x.txt", true
```

DeleteFolder

Exclui uma pasta.

Set *fso* **= CreateObject("Scripting.FileSystemObject")**

fso.**DeleteFolder** *strPasta [,fForçado]*

Parâmetro	Descrição
strPasta	Caminho para a pasta a ser excluída.
fForçado	Booleano. Esse parâmetro terá de ser *True* se a pasta a ser excluída for somente de leitura.

Exemplo

```
Rem Exclui a pasta "novapasta" criada anteriormente:
Set fso = CreateObject("Scripting.FileSystemObject")
fso.DeleteFolder "c:\novapasta"
```

DriveExists

Retornará *True* se o drive especificado existir.

Set *fso* **= CreateObject("Scripting.FileSystemObject")**

fRetorno **=** *fso*.**DriveExists(***strDrive***)**

Parâmetro	Descrição
strDrive	Nome do drive.

Exemplo

```
Rem Verifica se o drive "z:\" existe:
Set fso = CreateObject("Scripting.FileSystemObject")
fRetorno = fso.DriveExists("z:")
if fRetorno = True then
        Wscript.Echo "O drive Z:\ existe!"
else
        Wscript.Echo "O drive Z:\ não existe!"
end if
```

FileExists

Similar ao método *DriveExists()*, o método *FileExists()* também retornará *True* se o arquivo especificado existir.

Set *fso* **= CreateObject("Scripting.FileSystemObject")**

fRetorno **=** *fso*.**FileExists(***strArquivo***)**

Parâmetro	Descrição
strArquivo	Caminho completo para o arquivo cuja existência se pretende averiguar.

Exemplo

```
Rem Verifica se o arquivo "x.txt" existe:
Set fso = CreateObject("Scripting.FileSystemObject")
fRetorno = fso.FileExists("c:\x.txt")
if fRetorno = True then
        Wscript.Echo "O arquivo existe!"
else
        Wscript.Echo "O arquivo não existe!"
end if
```

FolderExists

Retornará *True* se a pasta especificada existir.

Set *fso* **= CreateObject("Scripting.FileSystemObject")**

fRetorno **=** *fso*.**FileExists(***strArquivo***)**

Parâmetro	Descrição
strArquivo	Caminho completo para a pasta cuja existência se pretende averiguar.

Exemplo

```
Rem Verifica a pasta Windows existe:
Set fso = CreateObject("Scripting.FileSystemObject")
fRetorno = fso.FolderExists("c:\windows")
if fRetorno = True then
        Wscript.Echo "A pasta Windows existe!"
else
        Wscript.Echo "A pasta Windows não existe! Provavelmente você instalou o
    sistema em outra máquina."
end if
```

GetAbsolutePathName

Retorna o caminho completo da pasta atual em relação a outra pasta.

Set *fso* **= CreateObject("Scripting.FileSystemObject")**

strCaminho **=**
fso.**GetAbsolutePathName(***strEspecificação***)**

Parâmetro	Descrição
strEspecificação	Um pouco difícil de explicar com palavras. Observe o exemplo abaixo.

Exemplo

```
Rem Partindo do princípio que a pasta atual é "Meus Documentos", observe o
comportamento do método GetAbsolutePathName:
Set fso = CreateObject("Scripting.FileSystemObject")
strCaminho = fso.GetAbsolutePathName("c:")
Rem Vai exibir "c:\meus documentos"
Wscript.Echo strCaminho
strCaminho = fso.GetAbsolutePathName("roberto")
Wscript.Echo strCaminho
Rem Vai exibir "c:\meus documentos\roberto"
strCaminho = fso.GetAbsolutePathName("c:\\\")
Wscript.Echo strCaminho
Rem Vai exibir "c:\"
```

GetBaseName

Retorna o nome de um arquivo sem a extensão.

Set *fso* **= CreateObject("Scripting.FileSystemObject")**

strNomeDoArquivo **=** *fso*.**GetBaseName(***strCaminho***)**

Parâmetro	Descrição
strCaminho	Caminho completo para o arquivo.

Exemplo

```
Rem Retorna o nome do arquivo "Notepad.exe":
Set fso = CreateObject("Scripting.FileSystemObject")
strNomeDoArquivo = fso.GetBaseName("c:\windows\notepad.exe")
Wscrtpt.Echo strNomeDoArquivo
```

GetDrive

Retorna um objeto *Drive*.

Set *fso* **= CreateObject("Scripting.FileSystemObject")**

Set *objDrive* **=** *fso*.**GetDrive(***strDrive***)**

Parâmetro	Descrição
strDrive	Uma denominação aceitável para o drive. Pode ser uma letra, uma letra seguida de dois pontos, uma letra seguida de dois pontos e de uma barra invertida, ou um nome de compartilhamento.

GetDriveName

Retorna o nome do drive contido no caminho fornecido.

Set *fso* **= CreateObject("Scripting.FileSystemObject")**

strDrive **=** *fso*.**GetDriveName(***strCaminho***)**

Parâmetro	Descrição
strCaminho	Um caminho na forma "unidadededisco\pasta\arquivo". O método retornará justamente a "unidadededisco".

GetExtensionName

Similar ao método *GetBaseName()*, só que ele retorna justamente a extensão de um arquivo.

Set *fso* **= CreateObject("Scripting.FileSystemObject")**

strExtDoArquivo **=** *fso*.**GetExtensionName(***strCaminho***)**

Exemplo

```
Rem Retorna a extensão do arquivo "Notepad.exe":
Set fso = CreateObject("Scripting.FileSystemObject")
strExtDoArquivo = fso.GetExtensionName("c:\windows\notepad.exe")
Wscript.Echo strExtDoArquivo
```

GetFile

Retorna um objeto *File* para um arquivo existente.

Set *fso* **= CreateObject("Scripting.FileSystemObject")**

Set *objFile* **=** *fso*.**GetFile(***strArquivo***)**

Parâmetro	Descrição
strArquivo	Caminho completo para o arquivo a ser retornado.

GetFileName

Retorna o nome do arquivo em um dado caminho.

Set *fso* **= CreateObject("Scripting.FileSystemObject")**

strNomeDoArquivo **=** *fso*.**GetFileName(***strCaminho***)**

Parâmetro	Descrição
strCaminho	Caminho completo para o arquivo cujo nome se pretende obter.

GetFolder

Retorna um objeto *Folder* para uma pasta existente.

Set *fso* **= CreateObject("Scripting.FileSystemObject")**

Set *f* **=** *fso.***GetFolder(***strPasta***)**

Parâmetro	Descrição
strPasta	Caminho completo para a pasta a ser retornada.

GetParentFolderName

Retorna a pasta onde está contida a última pasta ou arquivo de um caminho.

Set *fso* **= CreateObject("Scripting.FileSystemObject")**

strParentFolder **=** *fso.***GetParentFolderName(***strCaminho***)**

Parâmetro	Descrição
strCaminho	Caminho completo para uma pasta ou arquivo.

Exemplo

```
Rem Retorna a pasta onde está contida a pasta "System" do Windows:
Set fso = CreateObject("Scripting.FileSystemObject")
strParentFolder = fso.GetParentFolderName("c:\windows\system")
Wscript.Echo strParentFolder
```

GetSpecialFolder

Retorna o nome de uma pasta especial do Windows.

Set *fso* **= CreateObject("Scripting.FileSystemObject")**

strSpecialFolder **=** *fso.***GetSpecialFolder(***intTipo***)**

Parâmetro	Descrição
intTipo	
0	Pasta onde estão instalados os arquivos do Windows (normalmente, "C:\Windows" ou "C:\Winnt").
1	Pasta onde estão os arquivos de sistema do Windows (normalmente, "C:\Windows\System" ou "C:\Winnt\System").
2	Pasta onde são armazenados os arquivos temporários.

GetTempName

Gera um nome aleatório de arquivo temporário que pode ser usado pelo método *CreateTextFile()* em tarefas que exijam a utilização de arquivos temporários.

Set *fso* **= CreateObject("Scripting.FileSystemObject")**

*ArquivoTemporário = fso.***GetTempName**

Exemplo

```
Rem Gera um nome aleatório de arquivo temporário para ser utilizado pelo
método CreateTextFile():
Set fso = CreateObject("Scripting.FileSystemObject")
ArquivoTemporário = fso.GetTempName
Set objFile = fso.CreateTextFile(ArquivoTemporário)
Wscript.Echo fso.GetFileName(ArquivoTemporário)
```

MoveFile

Move um ou vários arquivos para outra localização. Semelhante a *CopyFile*, no entanto o(s) arquivo(s) deixa(m) de existir na origem. Os arquivos só serão movidos de um volume para o outro se o sistema operacional permitir.

Set *fso* **= CreateObject("Scripting.FileSystemObject")**

fso.**MoveFile** *origem, destino*

Parâmetro	Descrição
origem	Caminho completo para o(s) arquivo(s) a ser(em) movidos(s). Caracteres curinga (*) podem ser usados no lugar do nome dos arquivos.
destino	Nova localização do(s) arquivo(s).

Exemplo

```
Rem Move o arquivo "c:\x.txt" para a pasta "c:\windows\":
Set fso = CreateObject("Scripting.FileSystemObject")
fso.MoveFile "c:\x.txt", "c:\windows\"
```

MoveFolder

Move uma pasta para outra localização. Similar ao método *CopyFolder*, no entanto a pasta deixa de existir na origem.

Set *fso* **= CreateObject("Scripting.FileSystemObject")**

fso.**MoveFolder** *origem, destino*

Parâmetro	Descrição
origem	Caminho completo para a pasta a ser movida. Caracteres curinga (*) podem ser usados no lugar do nome da pasta.
destino	Nova localização da pasta.

Exemplo

```
Rem Move a pasta "c:\teste\" para a pasta "c:\windows\":
Set fso = CreateObject("Scripting.FileSystemObject")
fso.MoveFolder "c:\teste", "c:\windows\"
```

OpenTextFile

Abre um arquivo de texto e retorna um objeto *TextStream*.

Set *fso* **= CreateObject("Scripting.FileSystemObject")**

Const *ForReading* **= 1,** *ForWriting* **= 2,** *ForAppending* = 8

Set *f = fso*.**OpenTextFile(***strArquivo [,intModoDeAbertura] [,fCriarNovo] [,intFormato]***)**

Parâmetro	Descrição
strArquivo	Caminho completo para o arquivo de texto a ser aberto.

intMododeAbertura	Indica o que poderá ser feito com o arquivo depois de aberto.
ForReading	Somente leitura.
ForWriting	Escrita.
ForAppending	Dados serão anexados ao fim do arquivo.
fCriarNovo	Booleano. Indica se deverá ser criado um novo arquivo caso *strArquivo* não exista. O padrão é *False*.
intFormato	Um dos três valores *TriState* usados para definir o formato de abertura do arquivo.
-2	TriStateUseDefault. Usa o padrão do sistema para abrir o arquivo.
-1	TriStateTrue. Abre o arquivo como Unicode.
0	TriStateFalse. Abre o arquivo como ASCII.

Exemplo

```
Rem Abre o arquivo "c:\x.txt" para escrita:
Set fso = CreateObject("Scripting.FileSystemObject")
Const ForReading = 1, ForWriting = 2, ForAppending = 8
Set f = fso.OpenTextFile("c:\x.txt", ForWriting, True)
```

Objeto Drive

O objeto *Drive* é instanciado através da utilização do método *GetDrive()* do objeto *FileSystemObject*.

Propriedades	Descrição
AvailableSpace	A quantidade de espaço disponível em um drive para o usuário.
DriveLetter	A letra que representa o drive.
DriveType	O tipo do drive.
FileSystem	O sistema de arquivos em uso.
FreeSpace	A quantidade de espaço livre no drive.
IsReady	Retornará *True* se o drive estiver pronto para ser usado.
Path	Caminho para um drive.
RootFolder	Pasta raiz do drive.
SerialNumber	Número de identificação exclusiva do volume.
ShareName	Retorna o compartilhamento de rede para um drive especificado.
TotalSize	A quantidade de espaço em um drive.
VolumeName	Define ou retorna o nome do volume.

Propriedades do objeto Drive

AvailableSpace

Retorna a quantidade de espaço disponível para o usuário no drive especificado. Normalmente esse valor será igual àquele retornado pela propriedade *FreeSpace*.

Set *fso* **= CreateObject("Scripting.FileSystemObject")**

Set *objDrive* **=** *fso.***GetDrive(***strDrive***)**

intEspaço **=** *objDrive.***AvailableSpace**

DriveLetter

Retorna a letra que representa um drive.

Set *fso* **= CreateObject("Scripting.FileSystemObject")**

Set *objDrive* **=** *fso*.**GetDrive(***strDrive***)**

strLetra **=** *objDrive*.**DriveLetter**

DriveType

Retorna um inteiro indicando o tipo do drive.

Set *fso* **= CreateObject("Scripting.FileSystemObject")**

Set *objDrive* **=** *fso*.**GetDrive(***strDrive***)**

intTipoDoDrive **=** *objDrive*.**DriveType**

Tipo	Descrição
0	Desconhecido.
1	Removível (disquete).
2	Fixo.
3	Drive de Rede.
4	Drive de CD-ROM.
5	Disco de RAM.

FileSystem

Retorna o nome do sistema de arquivo em uso no volume. Os valores de retorno podem ser, tipicamente, "FAT", "NTFS" e "CDFS".

Set *fso* **= CreateObject("Scripting.FileSystemObject")**

Set *objDrive* **=** *fso*.**GetDrive(***strDrive***)**

strSistemaDeArquivo **=** *objDrive*.**FileSystem**

FreeSpace

Retorna a quantidade de espaço livre num drive.

Set *fso* **= CreateObject("Scripting.FileSystemObject")**

Set *objDrive* **=** *fso*.**GetDrive(***strDrive***)**

intEspaço **=** *objDrive*.**FreeSpace**

IsReady

Retornará *True* se o drive estiver pronto para operações de entrada/saída. Caso contrário retornará *False*. Especialmente útil para trabalhar com disquetes e CD-ROM.

Set *fso* **= CreateObject("Scripting.FileSystemObject")**

Set *objDrive* **=** *fso*.**GetDrive(***strDrive***)**

fPronto **=** *objDrive*.**IsReady**

Path

Retorna o caminho para um drive.

Set *fso* **= CreateObject("Scripting.FileSystemObject")**

Set *objDrive* **=** *fso.***GetDrive(***strDrive***)**

strCaminho **=** *objDrive.***Path**

RootFolder

Especifica a pasta raiz de um drive.

Set *fso* **= CreateObject("Scripting.FileSystemObject")**

Set *objDrive* **=** *fso.***GetDrive(***strDrive***)**

strPastaRaiz **=** *objDrive.***RootFolder**

SerialNumber

Retorna um identificador decimal exclusivo para o volume.

Set *fso* **= CreateObject("Scripting.FileSystemObject")**

Set *objDrive* **=** *fso.***GetDrive(***strDrive***)**

intID **=** *objDrive.***SerialNumber**

ShareName

Retorna o compartilhamento de rede para um drive.

Set *fso* **= CreateObject("Scripting.FileSystemObject")**

Set *objDrive* **=** *fso.***GetDrive(***strDrive***)**

strCompartilhamento **=** *objDrive.***ShareName**

TotalSize

Retorna a quantidade total de espaço em um drive, em bytes.

Set *fso* **= CreateObject("Scripting.FileSystemObject")**

Set *objDrive* **=** *fso.***GetDrive(***strDrive***)**

intEspaçoTotal **=** *objDrive.***TotalSize**

VolumeName

Retorna ou atribui um nome a um volume.

Set *fso* **= CreateObject("Scripting.FileSystemObject")**

Set *objDrive* **=** *fso.***GetDrive(***strDrive***)**

strLetra **=** *objDrive.***VolumeName**

ou

*objDrive.***VolumeName =** *strLetra*

Objeto File

O objeto *File* é instanciado através da utilização do método *GetFile()* do objeto *FileSystemObject*.

Propriedades	Descrição
Attributes	Obtém ou define atributos de um arquivo.
DateCreated	Data da criação de um arquivo.
DateLastAcessed	Data em que um arquivo foi acessado pela última vez.
DateLastModified	Data em que um arquivo foi modificado pela última vez.
Drive	Retorna a letra do drive onde o arquivo reside.
Name	Define ou retorna o nome do arquivo.
ParentFolder	Pasta onde o arquivo reside.
Path	Caminho completo para o arquivo.
ShortName	Nome do arquivo usando a convenção de nomenclatura do MS-DOS (8.3).
ShortPath	Caminho completo para o arquivo usando a convenção de nomenclatura do MS-DOS.
Size	Tamanho do arquivo em bytes.
Type	Retorna informações sobre o tipo do arquivo.

Métodos	Descrição
Copy	Copia um arquivo para outra localização.
Delete	Exclui um arquivo.
Move	Move um arquivo para outra localização.
OpenAsTextStream	Abre um arquivo como sendo um "fluxo de texto".

Propriedades do objeto File

Attributes

Retorna ou define os atributos de um arquivo.

Set *fso* **= CreateObject("Scripting.FileSystemObject")**

Set *objFile* **=** *fso*.**GetFile(***strArquivo***)**

intAtributos **=** *objFile*.**Attributes**

ou

objFile.**Attributes =** *intAtributos*

Parâmetro	Descrição
0	Arquivo normal. Nenhum atributo definido.
1	Arquivo somente leitura.
2	Arquivo oculto.
4	Arquivo de sistema.
8	Drive de disco.
16	Pasta.
32	Arquivo foi alterado desde o último backup.
64	Atalho.
128	Arquivo comprimido.

DateCreated

Retorna a data e a hora da criação de um arquivo.

Set *fso* **= CreateObject("Scripting.FileSystemObject")**

Set *objFile* **=** *fso*.**GetFile(***strArquivo***)**

strData **=** *objFile*.**DateCreated**

DateLastAcessed

Retorna a data e a hora em que um arquivo foi acessado pela última vez.

Set *fso* **= CreateObject("Scripting.FileSystemObject")**

Set *objFile* **=** *fso*.**GetFile(***strArquivo***)**

strData **=** *objFile*.**DateLastAcessed**

DateLastModified

Retorna a data e a hora em que um arquivo foi modificado pela última vez.

Set *fso* **= CreateObject("Scripting.FileSystemObject")**

Set *objFile* **=** *fso*.**GetFile(***strArquivo***)**

strData **=** *objFile*.**DateLastModified**

Drive

Retorna a letra do drive onde o arquivo reside.

Set *fso* **= CreateObject("Scripting.FileSystemObject")**

Set *objFile* **=** *fso*.**GetFile(***strArquivo***)**

strLetraDoDrive **=** *objFile*.**Drive**

Name

Define ou obtém o nome de um arquivo.

Set *fso* **= CreateObject("Scripting.FileSystemObject")**

Set *objFile* **=** *fso*.**GetFile(***strArquivo***)**

strNome **=** *objFile*.**Name**

objFile.**Name =** *strNome*

ParentFolder

Retorna a pasta onde o arquivo reside.

Set *fso* **= CreateObject("Scripting.FileSystemObject")**

Set *objFile* **=** *fso*.**GetFile(***strArquivo***)**

strPasta **=** *objFile***.ParentFolder**

Path

Retorna o caminho completo para o arquivo.

Set *fso* **= CreateObject("Scripting.FileSystemObject")**

Set *objFile* **=** *fso*.**GetFile(***strArquivo***)**

strCaminho **=** *objFile*.**Path**

ShortName

Fornece o nome do arquivo usando a convenção de nomenclatura do MS-DOS (8.3).

Set *fso* **= CreateObject("Scripting.FileSystemObject")**

Set *objFile* **=** *fso*.**GetFile(***strArquivo***)**

strNomeCurto **=** *objFile*.**ShortName**

ShortPath

Fornece o caminho completo para o arquivo usando a convenção de nomenclatura do MS-DOS (8.3).

Set *fso* **= CreateObject("Scripting.FileSystemObject")**

Set *objFile* **=** *fso*.**GetFile(***strArquivo***)**

strNomeCurto **=** *objFile*.**ShortPath**

Size

Retorna o tamanho de um arquivo em bytes. Se quiser converter para megabytes, basta dividir o valor retornado por 1024.

Set *fso* **= CreateObject("Scripting.FileSystemObject")**

Set *objFile* **=** *fso*.**GetFile(***strArquivo***)**

intTamanhoDoArquivo **=** *objFile*.**Size**

Type

Retorna informações sobre o tipo do arquivo. Por exemplo, para um arquivo ".txt", retornaria "Text Document". Acredito que a propriedade obtenha essas associações em HKEY_CLASSES_ROOT.

Set *fso* **= CreateObject("Scripting.FileSystemObject")**

Set *objFile* **=** *fso*.**GetFile(***strArquivo***)**

strTipoDoArquivo **=** *objFile*.**Type**

Métodos do objeto File

Copy

Copia um arquivo para outra localização.

Set *fso* **= CreateObject("Scripting.FileSystemObject")**

Set *objFile* **=** *fso*.**GetFile(***strArquivo***)**

objFile.**Copy** *destino [,fSubstituir]*

Parâmetro	Descrição
destino	Nova localização do arquivo.
fSubstituir	Se *True*, substituirá um arquivo existente na nova localização que tenha o mesmo nome do arquivo que estiver sendo copiado.

Exemplo

```
Rem Copia o arquivo "c:\x.txt" para a pasta "c:\windows\":
Set fso = CreateObject("Scripting.FileSystemObject")
Set objFile = fso.GetFile("c:\x.txt")
objFile.Copy "c:\windows\"
```

Delete

Exclui um arquivo.

Set *fso* **= CreateObject("Scripting.FileSystemObject")**

Set *objFile* **=** *fso*.**GetFile(***strArquivo***)**

objFile.**Delete** *[fForçado]*

Parâmetro	Descrição
fForçado	Booleano. Deverá ser *True* caso o arquivo a ser excluído seja somente leitura.

Exemplo

```
Rem Exclui o arquivo "c:\x.txt":
Set fso = CreateObject("Scripting.FileSystemObject")
Set objFile = fso.GetFile("c:\x.txt")
objFile.Delete
```

Move

Move um arquivo para outra localização.

Set *fso* **= CreateObject("Scripting.FileSystemObject")**

Set *objFile* **=** *fso*.**GetFile(***strArquivo***)**

objFile.**Move** *destino*

Parâmetro	Descrição
destino	Nova localização do arquivo.

Exemplo

```
Rem Move o arquivo "c:\x.txt" para a pasta "c:\windows\":
Set fso = CreateObject("Scripting.FileSystemObject")
Set objFile = fso.GetFile("c:\x.txt")
objFile.Move "c:\windows\"
```

OpenAsTextStream

Abre um arquivo de texto e retorna um objeto *TextStream*.

Set *fso* **= CreateObject("Scripting.FileSystemObject")**

Set *objFile* **=** *fso*.**GetFile(***strArquivo***)**

Const *ForReading* **= 1**, *ForWriting* **= 2**, *ForAppending* **= 8**

Set *FluxoDetexto* =
objFile.**OpenAsTextStream(***[intModoDeAbertura] [,intFormato]***)**

Parâmetro	Descrição
intModoDeAbertura	Indica o que poderá ser feito com o arquivo depois de aberto.
ForReading	Somente leitura.
ForWriting	Escrita.
ForAppending	Dados serão anexados no fim do arquivo.
intFormato	Um dos três valores *TriState* usados para definir o formato de abertura do arquivo.
-2	TriStateUseDefault. Usa o padrão do sistema para abrir o arquivo.
-1	TriStateTrue. Abre o arquivo como Unicode.
0	TriStateFalse. Abre o arquivo como ASCII.

Exemplo

```
Rem Abre o arquivo "c:\x.txt" para escrita:
Set fso = CreateObject("Scripting.FileSystemObject")
Set objFile = fso.GetFile("c:\x.txt")
Const ForReading = 1, ForWriting = 2, ForAppending = 8
Set FluxoDeTexto = objFile.OpenAsTextStream(ForWriting)
```

Objeto Folder

O objeto *Folder* é instanciado quando são utilizados os métodos *CreateFolder()* ou *GetFolder()* do objeto *FileSystemObject*.

Propriedades	Descrição
Attributes	Obtém ou define atributos de uma pasta.
DateCreated	Data da criação de uma pasta.
DateLastAcessed	Data em que a pasta foi acessada pela última vez.
DateLastModified	Data em que a pasta foi modificada pela última vez.
Drive	Retorna a letra do drive onde a pasta reside.
Files	Retorna uma coleção de objetos *File*.
IsRootFolder	Booleano. Determina se a pasta é a pasta raiz do volume.
Name	Define ou retorna o nome da pasta.
ParentFolder	Pasta dentro da qual a pasta atual reside.
Path	Caminho completo para a pasta.
ShortName	Nome da pasta usando a convenção de nomenclatura do MS-DOS (8.3).
ShortPath	Caminho completo para a pasta usando a convenção de nomenclatura do MS-DOS.
Size	Tamanho do conjunto de todos os arquivos e subpastas contidos dentro da pasta, em bytes.
SubFolders	Retorna uma coleção de objetos *Folder* contidos na pasta.
Type	Retorna informações sobre o tipo da pasta.

Métodos	Descrição
Copy	Copia uma pasta para outra localização.
Delete	Exclui uma pasta.
Move	Move uma pasta para outra localização.
CreateTextFile	Cria um arquivo de texto e retorna um objeto *TextStream*.

Propriedades do objeto Folder

Attributes

Define ou retorna os atributos de uma pasta. Semelhante à propriedade de mesmo nome do objeto *File*. Aliás, os objetos *File* e *Folder* compartilham muitas propriedades e métodos.

Set *fso* **= CreateObject("Scripting.FileSystemObject")**

Set *f* **=** *fso.***CreateFolder(***strNovaPasta***)**

ou

Set *f* **=** *fso.***GetFolder(***strPasta***)**

intAtributos **=** *f.***Attributes**

ou

*f.***Attributes =** *intAtributos*

Parâmetro	Descrição
0	Arquivo normal. Nenhum atributo definido.
1	Arquivo somente leitura.
2	Arquivo oculto.
4	Arquivo de sistema.
8	Drive de disco.
16	Pasta.
32	Arquivo foi alterado desde o último backup.
64	Atalho.
128	Arquivo comprimido.

DateCreated

Retorna a data e a hora da criação de uma pasta.

Set *fso* **= CreateObject("Scripting.FileSystemObject")**

Set *f* **=** *fso.***CreateFolder(***strNovaPasta***)**

ou

Set *f* **=** *fso.***GetFolder(***strPasta***)**

strData **=** *f.***DateCreated**

DateLastAcessed

Retorna a data e a hora em que uma pasta foi acessada pela última vez.

Set *fso* **= CreateObject("Scripting.FileSystemObject")**

Set *f* **=** *fso.***CreateFolder(***strNovaPasta***)**

ou

Set *f* **=** *fso.***GetFolder(***strPasta***)**

strData **=** *f.***DateLastAcessed**

DateLastModified

Retorna a data e a hora em que uma pasta foi modificada pela última vez.

Set *fso* **= CreateObject("Scripting.FileSystemObject")**

Set *f* **=** *fso*.**CreateFolder(***strNovaPasta***)**

ou

Set *f* **=** *fso*.**GetFolder(***strPasta***)**

strData **=** *f*.**DateLastModified**

Drive

Retorna a letra do drive onde a pasta reside.

Set *fso* **= CreateObject("Scripting.FileSystemObject")**

Set *f* **=** *fso*.**CreateFolder(***strNovaPasta***)**

ou

Set *f* **=** *fso*.**GetFolder(***strPasta***)**

strLetraDoDrive **=** *f*.**Drive**

Files

Retorna a coleção dos objetos *File* contidos na pasta.

Set *fso* **= CreateObject("Scripting.FileSystemObject")**

Set *f* **=** *fso*.**CreateFolder(***strNovaPasta***)**

ou

Set *f* **=** *fso*.**GetFolder(***strPasta***)**

Set *myFiles* **=** *f*.**Files**

IsRootFolder

Retornará *True* se a pasta em questão for a pasta raiz de um volume; caso contrário, retornará *False*.

Set *fso* **= CreateObject("Scripting.FileSystemObject")**

Set *f* **=** *fso*.**CreateFolder(***strNovaPasta***)**

ou

Set *f* **=** *fso*.**GetFolder(***strPasta***)**

fRetorno **=** *f*.**IsRootFolder**

Name

Define ou obtém o nome de uma pasta.

Set *fso* **= CreateObject("Scripting.FileSystemObject")**

Set *f* **=** *fso*.**CreateFolder(***strNovaPasta***)**

ou

Set *f* **=** *fso*.**GetFolder(***strPasta***)**

strNome **=** *f*.**Name**

ou

f.**Name =** *strNome*

ParentFolder

Retorna a pasta dentro da qual a pasta atual reside.

Set *fso* **= CreateObject("Scripting.FileSystemObject")**

Set *f* **=** *fso.***CreateFolder(***strNovaPasta***)**

ou

Set *f* **=** *fso.***GetFolder(***strPasta***)**

strPasta **=** *f***.ParentFolder**

Path

Retorna o caminho completo para a pasta.

Set *fso* **= CreateObject("Scripting.FileSystemObject")**

Set *f* **=** *fso.***CreateFolder(***strNovaPasta***)**

ou

Set *f* **=** *fso.***GetFolder(***strPasta***)**

strCaminho **=** *f.***Path**

ShortName

Fornece o nome da pasta usando a convenção de nomenclatura do MS-DOS (8.3).

Set *fso* **= CreateObject("Scripting.FileSystemObject")**

Set *f* **=** *fso.***CreateFolder(***strNovaPasta***)**

ou

Set *f* **=** *fso.***GetFolder(***strPasta***)**

strNomeCurto **=** *f***.ShortName**

ShortPath

Fornece o caminho completo para a pasta usando a convenção de nomenclatura do MS-DOS (8.3).

Set *fso* **= CreateObject("Scripting.FileSystemObject")**

Set *f* **=** *fso.***CreateFolder(***strNovaPasta***)**

ou

Set *f* **=** *fso.***GetFolder(***strPasta***)**

strNomeCurto **=** *f***.ShortPath**

Size

Retorna o tamanho total, em bytes, do conjunto de arquivos e subpastas contidos na pasta. Como no caso da propriedade de mesmo nome do objeto *File*, se quiser converter para megabytes, basta dividir o valor retornado por 1024.

Set *fso* **= CreateObject("Scripting.FileSystemObject")**

Set *f* **=** *fso.***CreateFolder(***strNovaPasta***)**

Set *f* **=** *fso.***GetFolder(***strPasta***)**

intTamanhoDaPasta **=** *f***.Size**

SubFolders

Retorna a coleção *Folders* com todas as subpastas contidas na pasta.

Set *fso* **= CreateObject("Scripting.FileSystemObject")**

Set *f* **=** *fso.***CreateFolder(***strNovaPasta***)**

ou

Set *f* **=** *fso.***GetFolder(***strPasta***)**

Set *mySubFolders* **=** *f.***SubFolders**

Type

Retorna informações sobre o tipo da pasta.

Set *fso* **= CreateObject("Scripting.FileSystemObject")**

Set *f* **=** *fso.***CreateFolder(***strNovaPasta***)**

ou

Set *f* **=** *fso.***GetFolder(***strPasta***)**

strTipoDaPasta **=** *f.***Type**

Métodos do objeto Folder

Copy

Copia uma pasta para outra localização.

Set *fso* **= CreateObject("Scripting.FileSystemObject")**

Set *f* **=** *fso.***CreateFolder(***strNovaPasta***)**

ou

Set *f* **=** *fso.***GetFolder(***strPasta***)**

*f.***Copy** *destino [,fSubstituir]*

Parâmetro	Descrição
destino	Nova localização da pasta.
fSubstituir	Se *True*, substituirá todos os arquivos homônimos numa pasta na nova localização que tenha o mesmo nome da pasta que estiver sendo copiada.

Delete

Exclui uma pasta.

Set *fso* **= CreateObject("Scripting.FileSystemObject")**

Set *f* **=** *fso.***CreateFolder(***strNovaPasta***)**

ou

Set *f* **=** *fso.***GetFolder(***strPasta***)**

*f.***Delete** *[fForçado]*

Parâmetro	Descrição
fForçado	Booleano. Deverá ser *True* caso a pasta a ser excluída seja somente leitura.

Move

Move uma pasta para outra localização.

Set *fso* **= CreateObject("Scripting.FileSystemObject")**

Set *f* **=** *fso*.**CreateFolder(***strNovaPasta***)**

ou

Set *f* **=** *fso*.**GetFolder(***strPasta***)**

f.**Move** *destino*

Parâmetro	Descrição
destino	Nova localização da pasta.

CreateTextFile

O método *CreateTextFile()* é semelhante ao método de mesmo nome do objeto *FileSytemObject*.

Set *fso* **= CreateObject("Scripting.FileSystemObject")**

Set *f* **=** *fso*.**CreateFolder(***strNovaPasta***)**

ou

Set *f* **=** *fso*.**GetFolder(***strPasta***)**

Set *ArquivoDeTexto* **=** *f*.**CreateTextFile(***strArquivoTexto* *[,fSubstituir] [,unicode]***)**

Parâmetro	Descrição
strArquivoTexto	Nome do arquivo de texto.
fSubstituir	Se *True*, substituirá um arquivo homônimo pelo arquivo que estiver sendo criado. Se *False*, o arquivo original será mantido.
unicode	Se *True*, um arquivo Unicode será criado. Se *False* ou omitido, será criado um arquivo ASCII.

Objeto TextStream

O objeto *TextStream* é instanciado quando da utilização de métodos dos objetos *FileSystemObject*, *File* ou *Folder* que criam ou abrem arquivos de texto. Na descrição da sintaxe e nos exemplos dos métodos e propriedades do objeto *TextStream*, utilizarei o método *OpenTextFile()* do objeto *FileSystemObject*.

Propriedades	
AtEndOfLine	Booleano. Determina se o final de uma linha foi atingido.
AtEndOfStream	Booleano. Determina se o final de um arquivo de texto foi atingido.
Column	Ponteiro para a coluna atual em um arquivo de texto.
Line	Ponteiro para a linha atual em um arquivo de texto.
Métodos	
Close	Fecha um arquivo de texto.
Read	Lê um número específico de caracteres em um arquivo de texto.
ReadAll	Lê todo o arquivo de texto.
ReadLine	Lê uma linha de um arquivo de texto.
Skip	Ignora um determinado número de caracteres quando lendo um arquivo de texto.
SkipLine	Ignora toda uma linha quando lendo um arquivo de texto.
Write	Escreve uma string num arquivo de texto.
WriteLine	Escreve uma string e uma quebra de linha.
WriteBlankLines	Escreve linhas em branco num arquivo de texto.

Propriedades do objeto TextStream

AtEndOfLine

Retornará *True* se o final de uma linha num arquivo de texto que esteja sendo lido for atingido.

Set *fso* **= CreateObject("Scripting.FileSystemObject")**

Const *ForReading* **= 1**, *ForWriting* **= 2**, *ForAppending* **= 8**

Set *f = fso.***OpenTextFile(***strArquivo [,intModoDeAbertura] [,fCriarNovo] [,intFormato]***)**

fRetorno **=** *f.***AtEndOfLine**

AtEndOfStream

Retornará *True* se o final de um arquivo de texto que esteja sendo lido for atingido.

Set *fso* **= CreateObject("Scripting.FileSystemObject")**

Const *ForReading* **= 1**, *ForWriting* **= 2**, *ForAppending* **= 8**

Set *f = fso.***OpenTextFile(***strArquivo [,intModoDeAbertura] [,fCriarNovo] [,intFormato]***)**

fRetorno **=** *f.***AtEndOfStream**

Column

Retorna um ponteiro para a coluna atual em um arquivo de texto que esteja sendo lido.

Set *fso* **= CreateObject("Scripting.FileSystemObject")**

Const *ForReading* **= 1**, *ForWriting* **= 2**, *ForAppending* **= 8**

Set *f = fso.***OpenTextFile(***strArquivo [,intModoDeAbertura] [,fCriarNovo] [,intFormato]***)**

intColuna **= f.Column**

Line

Retorna um ponteiro para a linha atual em um arquivo de texto.

Set *fso* **= CreateObject("Scripting.FileSystemObject")**

Const *ForReading* **= 1**, *ForWriting* **= 2**, *ForAppending* **= 8**

Set *f = fso.***OpenTextFile(***strArquivo [,intModoDeAbertura] [,fCriarNovo] [,intFormato]***)**

intLinha **= f.Line**

Métodos do objeto TextStream

Close

Fecha um arquivo de texto.

Set *fso* **= CreateObject("Scripting.FileSystemObject")**

Const *ForReading* **= 1**, *ForWriting* **= 2**, *ForAppending* **= 8**

Set *f = fso.***OpenTextFile(***strArquivo [,intModoDeAbertura] [,fCriarNovo] [,intFormato]***)**

*f.***Close**

Read

Lê um número específico de caracteres em um arquivo de texto, retornando a string correspondente.

Set *fso* **= CreateObject("Scripting.FileSystemObject")**

Const *ForReading* **= 1**, *ForWriting* **= 2**, *ForAppending* **= 8**

Set *f = fso.***OpenTextFile(***strArquivo,* **ForReading** *[,fCriarNovo] [,intFormato]***)**

strRetorno **= f.Read(***intNúmeroDeCaracteres***)**

Parâmetro	Descrição
intNúmeroDeCaracteres	Número de caracteres a serem lidos do arquivo, a partir da posição atual.

Exemplo

```
Rem Lê dez caracteres do arquivo "c:\x.txt":
Set fso = CreateObject("Scripting.FileSystemObject")
Const ForReading = 1, ForWriting = 2, ForAppending = 8
Set f = fso.OpenTextFile("c:\x.txt", ForReading)
strRetorno = f.Read(10)
f.Close
Wscript.Echo strRetorno
```

ReadAll

Lê todo o arquivo de texto, retornando a string correspondente.

Set *fso* **= CreateObject("Scripting.FileSystemObject")**

Const *ForReading* **= 1,** *ForWriting* **= 2,** *ForAppending* **= 8**

Set *f = fso.***OpenTextFile(***strArquivo,* **ForReading** *[,fCriarNovo] [,intFormato]***)**

strRetorno **=** *f.***ReadAll**

Exemplo

```
Rem Lê todo o arquivo "c:\x.txt":
Set fso = CreateObject("Scripting.FileSystemObject")
Const ForReading = 1, ForWriting = 2, ForAppending = 8
Set f = fso.OpenTextFile("c:\x.txt", ForReading)
strRetorno = f.ReadAll
f.Close
Wscript.Echo strRetorno
```

ReadLine

Lê uma linha num arquivo de texto, a partir da linha atual, retornando a string correspondente e movendo o ponteiro para a próxima linha.

Set *fso* **= CreateObject("Scripting.FileSystemObject")**

Const *ForReading* **= 1,** *ForWriting* **= 2,** *ForAppending* **= 8**

Set *f = fso.***OpenTextFile(***strArquivo,* **ForReading** *[,fCriarNovo] [,intFormato]***)**

strRetorno **=** *f.***ReadLine**

Exemplo

```
Rem Lê as quatro primeiras linhas do arquivo "c:\x.txt":
Dim i
i = 1
Set fso = CreateObject("Scripting.FileSystemObject")
Const ForReading = 1, ForWriting = 2, ForAppending = 8
Set f = fso.OpenTextFile("c:\x.txt", ForReading)
Do While f.AtEndOfStream <> True And i <= 4
    strRetorno =  strRetorno + f.ReadLine
    i = i+1
Loop
f.close
Wscript.Echo strRetorno
```

Skip

Ignora um determinado número de caracteres quando lendo um arquivo de texto.

Set *fso* **= CreateObject("Scripting.FileSystemObject")**

Const *ForReading* **= 1,** *ForWriting* **= 2,** *ForAppending* **= 8**

Set *f = fso.***OpenTextFile(***strArquivo [,intModoDeAbertura] [,fCriarNovo] [,intFormato]***)**

*f.***Skip(***intNúmeroDeCaracteres***)**

Parâmetro	Descrição
intNúmeroDeCaracteres	Número de caracteres a serem ignorados, a partir da posição atual.

Exemplo

```
Rem Lê cinco caracteres do arquivo "c:\x.txt" a partir do quarto caractere:
Set fso = CreateObject("Scripting.FileSystemObject")
Const ForReading = 1, ForWriting = 2, ForAppending = 8
Set f = fso.OpenTextFile("c:\x.txt", ForReading)
f.Skip(3)
strRetorno = f.Read(5)
f.Close
Wscript.Echo strRetorno
```

SkipLine

Ignora toda uma linha quando lendo um arquivo de texto.

Set *fso* **= CreateObject("Scripting.FileSystemObject")**

Const *ForReading* **=** *1*, *ForWriting* **=** *2*, *ForAppending* **=** *8*

Set *f = fso.***OpenTextFile(***strArquivo [,intModoDeAbertura] [,fCriarNovo] [,intFormato]***)**

*f.***SkipLine**

Exemplo

```
Rem Lê as três primeiras linhas do arquivo "c:\x.txt", a partir da segunda:
Dim i
i = 1
Set fso = CreateObject("Scripting.FileSystemObject")
Const ForReading = 1, ForWriting = 2, ForAppending = 8
Set f = fso.OpenTextFile("c:\x.txt", ForReading)
f.SkipLine
Do While f.AtEndOfStream <> True And i <= 3
    strRetorno =  strRetorno + f.ReadLine
    i = i+1
Loop
f.close
Wscript.Echo strRetorno
```

Write

Escreve uma string num arquivo de texto.

Set *fso* **= CreateObject("Scripting.FileSystemObject")**

Const *ForReading* **=** *1*, *ForWriting* **=** *2*, *ForAppending* **=** *8*

Set *f = fso.***OpenTextFile(***strArquivo,*
ForWriting|ForAppending *[,fCriarNovo] [,intFormato]***)**

*f.***Write** *string*

Exemplo

```
Rem Escreve "Olá mundo!" no arquivo "c:\x.txt":
Set fso = CreateObject("Scripting.FileSystemObject")
Const ForReading = 1, ForWriting = 2, ForAppending = 8
Set f = fso.OpenTextFile("c:\x.txt", ForWriting)
f.Write "Olá mundo!"
f.Close
```

WriteLine

Escreve uma string e uma quebra de linha num arquivo de texto.

Set *fso* **= CreateObject("Scripting.FileSystemObject")**

Const *ForReading* **= 1,** *ForWriting* **= 2,** *ForAppending* **= 8**

Set *f = fso.***OpenTextFile(***strArquivo,*
ForWriting|ForAppending *[,fCriarNovo]*
*[,intFormato]***)**

*f.***WriteLine** *string*

Parâmetro	Descrição
string	String a ser escrita no arquivo de texto.

Exemplo

```
Rem Escreve "Olá mundo!" no arquivo "c:\x.txt":
Set fso = CreateObject("Scripting.FileSystemObject")
Const ForReading = 1, ForWriting = 2, ForAppending = 8
Set f = fso.OpenTextFile("c:\x.txt", ForWriting)
f.WriteLine "Olá mundo!"
f.Close
```

WriteBlankLines

Escreve linhas em branco num arquivo de texto.

Set *fso* **= CreateObject("Scripting.FileSystemObject")**

Const *ForReading* **= 1,** *ForWriting* **= 2,** *ForAppending* **= 8**

Set *f = fso.***OpenTextFile(***strArquivo,*
ForWriting|ForAppending *[,fCriarNovo]*
*[,intFormato]***)**

*f.***WriteBlankLines** *intNúmeroDeLinhas*

Parâmetro	Descrição
intNúmeroDeLinhas	Número de linhas em branco a serem escritas no arquivo de texto.

Exemplo

```
Rem Escreve três linhas em branco no final do arquivo "c:\x.txt":
Set fso = CreateObject("Scripting.FileSystemObject")
Const ForReading = 1, ForWriting = 2, ForAppending = 8
Set f = fso.OpenTextFile("c:\x.txt", ForAppending)
f.WriteBlankLines 3
f.Close
```

Recursos Avançados

Teoricamente, não há limites para aquilo que os scripts do WSH podem fazer dentro do ambiente Windows de 32 bits. Quaisquer recursos do sistema operacional, inclusive aqueles relacionados ao hardware, podem ser manipulados por um script do WSH, desde que você saiba como acessá-los.

Nesta seção, farei uma breve explanação sobre as classes WIN32 e sobre a ADSI.

Classes WIN32

As classes WIN32 expandem enormemente as funcionalidade de scripts do WSH e de aplicações que necessitam de acesso a recursos avançados do sistema operacional. A maneira mais simples de instanciar um objeto a partir de uma classe WIN32 é:

Set *objWIN32* **=**
GetObject("WinMgmts::Win32_*Classe[='ItemRef']*"**)**

Exemplo

```
Rem Instancia um objeto da classe Win32_Service fazendo referência ao serviço
NetLogon:
dim obj
Set obj = GetObject("WinMgmts::Win32_Service='NetLogon'")
Select Case obj.State
    Case "Stopped"
                obj.StartService
                msgbox "O serviço NetLogon foi ativado!"
     Case "Running"
                msgbox "O serviço NetLogon já está ativo!"
     Case Else  msgbox "O status atual do serviço é " & obj.State
End Select
```

As classes WIN32 dividem-se em quatro categorias, que por sua vez se compõem-se de dezenas de classes com centenas de propriedades e métodos a elas associados.

Categorias	Descrição
Computer System Hardware	Classes que representam objetos relacionados ao hardware.
Operating System	Classes que representam objetos relacionados ao sistema operacional.
Installed Applications	Classes que representam objetos relacionados ao software instalado.
WMI Service Management	Classes utilizadas no gerenciamento do WMI (Windows Management Instrumentation).

Informações completas sobre as classes WIN32, incluindo uma lista das classes disponíveis com uma descrição detalhada de suas propriedades e métodos, bem como sobre a tecnologia WMI, podem ser obtidas no site http://msdn.microsoft.com/library/toc.asp?PP=/library/toc/psdk/psdk6.xml&tocPath=psdk6.

ADSI

Com o lançamento do Windows 2000, a Microsoft apresentou ao mundo sua nova implementação de serviços de diretório: o Active Directory. Serviços de diretório são serviços que o sistema operacional de rede oferece para permitir a criação, exclusão e alteração organizada e centralizada de objetos da rede, tais como usuários, grupos, computadores, etc. O Active Directory é baseado nas especificações da versão 3.0 do protocolo LDAP (Lightweight Directory Access Protocol). Os desenvolvedores de scripts poderão manipular os objetos armazenados no diretório através das Active Directory Service Interfaces (ADSI).

Embora a ADSI tenha sido desenvolvida tendo o Active Directory em mente (o que é compreensível), também trabalha com outros "provedores", a saber:

- Windows NT SAM,
- NDS e
- Netware 3.x.

Exemplo

```
Rem Retornando os usuários contidos no grupo global Domain Admins de um
domíno do Windows NT:
Dim Grupo,  Membro
Set Grupo = GetObject("WinNT://Dominio/Domain Admins")
For Each Membro in Grupo.Members
   Wscript.echo Membro.Name
Next

Rem Criando um usuário chamado "joão" em um diretório do Active Directory e
definindo a data de expiração da conta para o dia 1º de
Rem janeiro de 2001:
Dim ou, usuario
Set ou = GetObject("LDAP://OU=X,DC=Microsoft,DC=COM")
Set usuario = ou.Create("user", "cn=Joao")
usuario.Put "samAccountName", "joao"
usuario.AccountExpirationDate = "01/01/2001"
usuario.SetInfo
```

No site http://www.microsoft.com/adsi você poderá obter maiores informações sobre a ADSI, bem como baixar a versão do produto para o Windows NT. Poderá obter também a documentação da ADSI em formato HTMLHelp.

Scripts de Logon

A utilização mais evidente e comum para scripts do WSH, sem sombra de dúvidas, é em scripts de logon. Scripts de logon são arquivos executáveis (.exe, .com, .bat, etc.) que são executados assim que um usuário é validado por um domínio do Windows NT ou do Windows 2000. Normalmente, os scripts de logon são armazenados no compartilhamento NETLOGON de um controlador de domínio.

Existem, no entanto, alguns problemas relacionados à utilização do WSH para essa finalidade. Em primeiro lugar, apenas as versões de 32 bits do Windows podem executar scripts do WSH. Isso exclui clientes de rede utilizando o MS-DOS ou o Windows 3.11. Em segundo, o Windows NT e o Windows 95 não vêm com o WSH instalado por padrão. Por fim, só o Windows 2000 reconhece os arquivos do WSH (.vbs, .js, .wsf, etc.) como arquivos executáveis.

Assim, a menos que você esteja numa rede onde todos os servidores e clientes utilizem o Windows 2000, dificilmente conseguirá implantar scripts de logon integralmente desenvolvidos no WSH. Algumas saídas que poderá adotar são as seguintes:

- Se sua rede tiver clientes MS-DOS e Windows 3.11, crie um arquivo .bat que execute um script de logon em separado para esses ambientes e o script WSH para os demais.
- Caso tenha máquinas utilizando o Windows 95 ou o Windows NT, verifique, utilizando o código apropriado dentro de um script .bat, se os arquivos Wscript.exe e Cscript.exe existem. Caso não existam, utilize o mesmo script .bat para iniciar o programa de instalação do WSH, convenientemente localizado num compartilhamento da rede.
- Uma vez que somente os sistemas operacionais da família Windows 2000 vêem os arquivos do WSH como autênticos executáveis, para os outros ambientes (exceçto o MS-DOS e o Windows 3.11, é claro) você deverá usar algo como "cscript *caminhodoscript*" quando estiver configurando scripts de logon para uma conta de usuário num controlador de domínio Windows NT ou Windows 2000.

Windows Script Components

Os componentes de script do Windows Script Components (WSC) são a maneira mais fácil de criar componentes COM para serem utilizados por páginas do Active Server Pages ou por scripts do WSH.

Um arquivo do WSC nada mais é que um arquivo texto com extensão .wsc que utiliza a XML juntamente com uma linguagem de script compatível com a tecnologia ActiveX (como o VBScript e o JScript) para criar um componente de script reutilizável. Para desenvolver um componente de script, basta um editor de texto simples como o Notepad. No entanto, no site http://msdn.microsoft.com/scripting é possível obter gratuitamente um *wizard* que auxilia no desenvolvimento de componentes de script do WSC.

Para ser utilizado, um componente de script WSC necessita:

- do arquivo scrobj.dll (run-time do WSC) instalado;
- de quaisquer arquivos (executáveis, bibliotecas, etc.) requeridos por objetos utilizados no desenvolvimento do componente; e
- que o arquivo .wsc seja registrado (para registrá-lo, clique com o botão direito do mouse sobre o arquivo .wsc e escolha, no menu suspenso, a opção **Register**).

Para instanciar um objeto a partir de um componente de script, utilize a seguinte sintaxe:

Set *obj* **= CreateObject(***progID***)**

O *progID* é definido pela atribuição de uma string à variável *progID* entre as tags <registation></registration>.

Abaixo, o esqueleto típico de um arquivo .wsc onde um componente de script está sendo difinido:

```
<?XML version="1.0"?>
<component>
<registration>
  description=DescriçãoDoComponente
  progid=progID
  version=VersãoDoComponente
  classid=clsid
</registration>
<public>
  <property name=NomeDeUmaPropriedade/>
  <method name=NomeDeUmMétodo/>
</public>
<script language="VBScript">
  <![CDATA[
  Function|Sub NomeDeUmMétodo([Parâmetro1][,Parâmetro2]...[,ParâmetroN])
    Código do método NomeDeUmMétodo
  End Function|End Sub
  ]]>
</script>
</component>
```

Informações mais completas sobre o WSC podem ser obtidas no site http://msdn.microsoft.com/scripting. Desse site também poderá ser baixada a documentação completa do WSC, em um arquivo no formato HTMLHelp.

Criptografando Scripts do WSH

Você sabe o que é isso aí embaixo?

```
#@~^DwEAAA==]□:~A/1D+7+,0DêkPsk
    tCd,+:,8DmxmK~UW,0bxCV~[KPCD$;k7W~E1)-a D6OJ=@#@&?nDP6/
    W,',/
    D□IO+}4%n1Y'rjmMkwDrUocsbVn?zdD+hr(%+1YEb@#@&ZGUkY~sK."+C9k
    oP{P8SPwW.□MkOr oP{~ BPsK.)wa+
    NrxL~{PO@#@&j+DPW~{POdGcr2+
    P□6Owk^+'rm=w6cYaYr~~oKDba2+ Nk
    Lb@#@&0c□.kOn~VCx0Jk+d~2@#@&W ;VG/
    □@#@&eFUAAA==^#~@
```

Trata-se do exemplo de utilização do método *WriteBlankLines* do objeto *TextStream* criptografado. No site http://msdn.microsoft.com/scripting você poderá obter, gratuitamente, o ScriptEncoder.

O ScriptEncoder é uma ferramenta simples de linha de comando que você pode utilizar para criptografar scripts do WSH e arquivos ASP e HTML. A sintaxe para utilizar o ScriptEncoder é a seguinte:

SCRENC [**/s**] [**/f**] [**/xl**] [**/l** *strLinguagem*] [**/e** *strExtensão*] *strArquivoOriginal strNovoArquivo*

Opções	Descrição
/s	Esta chave determina que o ScriptEncoder não deve fornecer nenhuma saída na tela.
/f	Esta chave determina que *strArquivoOriginal* deve ser sobrescrito por *strNovoArquivo*.
/xl	Esta chave especifica que a diretiva *@language* do ASP não está no topo do arquivo.
/l *strLinguagem*	Determina qual a linguagem de script padrão. Não precisa ser utilizada com arquivos do WSH, pois a extensão do arquivo (.vbs ou .js) já determina por si só qual é a linguagem de script padrão.
/e *strExtensão*	Associa a extensão do arquivo *strArquivoOriginal* com um tipo definido de arquivo.
strArquivoOriginal	Arquivo a ser criptografado.
strNovoArquivo	Arquivo critpografado resultante. Salve seus arquivos .vbs com a extensão .vbe e os arquivos .js com a extensão .jse, para que o WSH saiba que são arquivos criptogrados.

VBScript

Introdução

O Visual Basic Scripting Edition (VBScript) é uma linguagem de programação em grande parte compatível com o Visual Basic que foi desenvolvida pela Microsoft visando sua utilização, inicialmente, na Internet. Não se trata de uma linguagem compilada, ou seja, não é possível gerar executáveis a partir do VBScript. Com o VBScript você pode:

- Desenvolver páginas do Active Server Pages;
- Embutir rotinas de programação em páginas HTML; e
- Criar scripts do WSH (que é o que nos interessa).

Conforme já foi citado anteriormente, os scripts do WSH escritos em VBScript devem ser salvos com a extensão de arquivo .vbs. Isso instrui o WSH a chamar o mecanismo de scripting do VBScript para executar o script. De forma análoga, quem tem mais intimidade com o JScript deve salvar os scripts com a extensão de arquivo .js.

Não pretendo que este guia seja uma fonte de referência completa sobre o VBScript, até porque essa não é sua finalidade. Portanto, abordarei apenas o essencial para que o leitor seja capaz de implementar scripts do WSH realmente funcionais fazendo uso do VBScript como linguagem de programação.

Instruções condicionais

Como a quase totalidade das linguagens de programação, o VBScript faz uso intensivo de instruções condicionais. Instruções condicionais são aquelas em que um determinado fragmento de código só será executado caso uma expressão seja avaliada como sendo verdadeira.

Para o leitor que já tiver alguma familiaridade com o Visual Basic, as instruções condicionais do VBScript são implementadas da mesma forma.

If

A instrução **If** na sua forma mais simples tem a seguinte sintaxe:

If *expressão* **then** *declaração*

Quando mais de uma declaração for executada caso a expressão seja avaliada como verdadeira, usa-se:

If *expressão* **then**
    Declaração1
    Declaração2
    ...
    DeclaraçãoN
End If

Instruções condicionais **If** podem ser aninhadas umas dentro das outras em vários níveis. Quando aninhar instruções **If**, cuidado com as instruções de fechamento **End If**, que são uma causa comum de erros de tempo de execução.

Else

A instrução **Else** permite que uma ou mais declarações sejam executadas caso a expressão avaliada por **If** seja falsa.

If *expressão* **then**
    Declaração1
    Declaração2
    ...
    DeclaraçãoN
Else
    DeclaraçãoElse
End If

Uma variação da construção condicional **If... Then... Else** é **If... Then... ElseIf... Then... Else**. Não é muito usual nem elegante a sua utilização.

If *expressão1* **then**
    Declaração1
    Declaração2
    ...
    DeclaraçãoN
ElseIf *expressão2* **then**
    DeclaraçãoElseIf
Else
    DeclaraçãoElse
End If

Podem ser usados tantos **ElseIf** quantos o programador julgar necessários.

Exemplos

Um exemplo de utilização da construção condicional If... Then... Else:

```
Dim X, Y
X = InputBox("Entre um número:")
If X<10 Then
        Y = "O número que você forneceu é menor que 10!"
Else
        Y = "O número que você forneceu é maior ou igual a 10!"
End If
Wscript.Echo Y
```

Um exemplo de utilização de Ifs aninhados:

```
Dim X, Y
X = InputBox("Entre um número:")
If X Mod 2=0 then
        If X<10 then
          Y="O número que você forneceu é par e menor que 10!"
        Else
          Y="O número que você forneceu é par e maior ou igual a 10!"
        End If
Else
        If X<10 then
          Y="O número que você forneceu é ímpar e menor que 10!"
        Else
           Y="O número que você forneceu é ímpar e maior que 10!"
        End If
End If
Wscript.Echo Y
```

Select Case

Uma alternativa ao uso do **ElseIf**, que é bem mais legível e elegante, é a instrução **Select Case**.

A sintaxe para a instrução **Select Case** é a seguinte:

Select Case *variável*
    Case *Valor1*
        Declaração 1
    Case *Valor2*
        Declaração2
    Case *Valor3*
        Declaração3
    Case *ValorN*
        DeclaraçãoN
    Case Else
        DeclaraçãoElse
End Select

Exemplo

Um exemplo de utilização da instrução Select Case:

```
Dim strEstado, strMsg
strEstado=InputBox("Entre o nome do Estado em que você nasceu:")
Select Case strEstado
        Case "Minas Gerais"
                strMsg="Mineiros são sempre bem-vindos!"
        Case "São Paulo"
                strMsg="Paulistas são sempre bem-vindos!"
        Case "Rio de Janeiro"
                strMsg="Cariocas são sempre bem-vindos!"
    Case "Bahia"
                strMsg="Baianos são sempre bem-vindos!"
        Case "Rio Grande do Sul"
                strMsg="Gaúchos são sempre bem-vindos!"
        Case Else
                strMsg="Não importa de onde você venha, será sempre
     bem-vindo!"
End Select
Wscript.Echo strMsg
```

Loops

Outro tipo de instrução bastante popular em praticamente todas as linguagens de programação (e o VBScript não foge à regra) são os Loops. As instruções de Loop repetem a execução de uma ou mais declarações um determinado número de vezes.

For

A instrução **For** é a instrução de loop mais usada.

For *Variável=ValorInicial* **To** *ValorFinal* **[Step** *step***]**

Declarações

Next

A instrução opcional **Step** determina de quanto a variável será incrementada cada vez que o laço for executado.

Exemplos

Contando de 1 até 10 usando um laço For:

```
Dim X, Y(9), strMsg
For X=1 To 10
        Y(X-1)=X
        strMsg = strMsg & chr(10) & Y(X-1)
Next
Wscript.Echo strMsg
```

Contando os números ímpares de 1 até 25:

```
Dim X, Y(25), strMsg
For X=1 To 25 Step 2
        Y(X-1)=X
        strMsg = strMsg & chr(10) & Y(X-1)
Next
Wscript.Echo strMsg
```

While... Wend

A construção **While... Wend** é utilizada quando não se tem certeza do número de vezes que o loop será executado.

While *Expressão*

Declarações

Wend

Como a expressão é avaliada antes que o loop seja executado pela primeira vez, caso ela seja falsa, as declarações no corpo do loop **While... Wend** podem nunca ser executadas.

Do... Loop

A construção **Do... Loop** pode ser implementada de várias formas, como demonstrado a seguir:

Do While *Expressão*

Declarações

Loop

Esta primeira forma será executada enquanto a expressão avaliada for verdadeira.

Do

Declarações

Loop While *Expressão*

Esta segunda forma é em quase tudo semelhante à anterior, só que executará pelo menos uma vez, mesmo que a expressão avaliada seja falsa logo na primeira iteração do laço.

Do Until *Expressão*

Declarações

Loop

Funciona de maneira análoga à construção **Do While... Loop**, só que as declarações só serão executadas enquanto a expressão for falsa.

Do

Declarações

Loop Until *Expressão*

É semelhante à construção **Do... Loop While**, só que as declarações, após serem executadas pelo menos uma vez, só continuarão a ser executadas caso a expressão avaliada seja falsa.

Exit

A instrução **Exit**, através das suas variações (**Exit Do**, **Exit For**), força a saída prematura de um loop.

Exemplos

Um exemplo de Do While... Loop com Exit Do:

```
Dim X, strMsg
Do While X < 10
        X = X+1
        if X=7 then
                Exit Do
        Else
                strMsg = strMsg &  chr(10) & "Olá, mundo!"
        End If
Loop
Wscript.Echo strMsg
```

For Each

Um tipo de instrução de loop especial do VBScript é a **For Each**. Permite enumerar todos os elementos de uma coleção.

For Each *Elemento* **In** *Coleção*

Declarações

Next

Dentro do WSH, como já tivemos a oportunidade de ver, existem muitas coleções e o **For Each** pode ser bastante útil.

Sub-Rotinas e Funções

Assim como no seu irmão mais velho, o Visual Basic, a principal tarefa de programação no âmbito do VBScript é o desenvolvimento de sub-rotinas e funções.

Sub-Rotinas

Sub-rotinas são blocos de código contidos entre as instruções **Sub... End Sub**. Podem ou não receber parâmetros e não retornam valores. A sintaxe de uma sub-rotina é:

Sub *NomeDaSubRotina([Parâmetro1], [Parâmetro2],... [ParâmetroN])*

Declarações

End Sub

Uma nota importante é que os procedimentos **Sub** devem ser chamados dentro do código VBScript sem os parênteses.

Exemplo

```
Sub Mensagem(strMensagem)
        Wscript.Echo strMensagem
End Sub
Dim strMsg
strMsg=InputBox("Entre uma mensagem:")
Mensagem strMsg.
```

Funções

Funções são blocos de código contidos dentro das instruções **Function... End Function**. Como as sub-rotinas, as funções podem ou não ter parâmetros. A principal diferença entre elas é que as funções retornam valores e devem ser chamadas no código VBScript com os parênteses. A sintaxe para definição de uma função é:

Function *NomeDaFunção([Parâmetro1], [Parâmetro2],... [ParâmetroN])*

Declarações

End Function

Exemplo

```
Function Mensagem()
        Mensagem=InputBox("Entre uma mensagem:")
End Function
Wscript.Echo Mensagem()
```

Operadores

Uma das principais funcionalidades de uma linguagem de programação é a disponibilização de operadores, visto que são eles que permitem a construção de expressões.

Os operadores fornecidos pelo VBScript são:

Operadores	Significado
+	Adição de duas ou mais expressões numéricas ou concatenação de strings.
-	Subtração de uma expressão numérica de outra.
*	Multiplicação de duas ou mais expressões numéricas.
/	Divisão de uma expressão numérica pela outra.
^	Exponenciação.
=	Igualdade.
\	Divisão de inteiros sem parte fracionária.
Mod	Retorna o resto da divisão entre dois números inteiros.
And	Retornará verdadeiro se ambos os valores utilizados forem verdadeiros.
Or	Retornará verdadeiro se pelo menos um dos valores utilizados for verdadeiro.
Xor	Retornará verdadeiro se apenas um dos valores utilizados for verdadeiro.
Imp	Implicação.
Eqv	Equivalência.
<>	Diferente de.
>	Maior que.
<	Menor que.
>=	Maior ou igual a.
<=	Menor ou igual a.
&	Concatenação de strings.

Tipos de Dados

O VBScript, ao contrário do Visual Basic, não permite a definição explícita do tipo de dado que uma variável poderá conter quando da sua declaração, ou seja, todas as variáveis são declaradas obrigatoriamente como do tipo Variant. Entretanto, o VBScript suporta grande variedade de subtipos de dados e qual subtipo de dado será atribuído a uma variável é algo que será definido no contexto em que essa variável estiver sendo utilizada ou pelo uso de funções de conversão, tais como CInt(), CStr()ou CDate(), por exemplo.

Subtipo	Prefixo Usual	Descrição
Boolean	f, b ou boo	Valor lógico, que pode ser True ou False.
Byte		Inteiro sem sinal entre 0 e 255.
Integer	int	Inteiro com sinal entre –32.768 e 32.767.
Long	lng	Inteiro com sinal entre –2.147.483.648 e 2.147.483.647.
Single		Número fracionário de ponto flutuante entre –3,402823E38 e –1,401298E-45 para valores negativos e entre 1,401298E-45 e 3,402823E38 para valores positivos.
Double	dbl	Número fracionário de ponto flutuante entre –4,94065645841247E-324 e –1,79769313486232E308 para valores negativos e entre 1,79769313486232E308 e 4,94065645841247E-324 para valores positivos.
Currency	cur	Valores monetários.
Date	d	Valores de datas.
Object	obj	Objeto.
String	str	String (texto) de até aproximadamente dois bilhões de caracteres.

Outras instruções do VBScript

Existem ainda algumas instruções que não mencionei e que são muito importantes:

Instruções	Significado
Dim	Declara uma variável. Utilize-a sempre, para não se perder em seu próprio código.
Option Explicit	Faz com que todas as variáveis sejam declaradas.
Erase	Apaga o conteúdo de uma matriz.
Preserve	Copia o conteúdo de uma matriz dinâmica para outra matriz dinâmica que tenha sido redimensionada.
Redim	Redimensiona uma matriz dinâmica.
Set	Atribui um objeto a uma variável.
On Error	Usada em conjunto com Resume Next para o tratamento de erros.
Call	Chama um procedimento.
Rem	Insere uma linha de comentário.

Funções, constantes e objetos do VBScript

O VBScript fornece inúmeras funções e constantes e alguns objetos, tantos que seria necessário um outro guia exclusivamente para enumerá-los e descrevê-los. Abaixo, os mais utilizados:

Funções	
Asc	Retorna o código ASCII de um caractere. Sintaxe: Asc(string).
CBool	Converte seu parâmetro para o subtipo Boolean. Sintaxe: CBool(expressão).
CByte	Converte seu parâmetro para o subtipo Byte. Sintaxe: CByte(expressão).
CDate	Converte seu parâmetro para o subtipo Date. Sintaxe: CDatel(expressão).
CInt	Converte seu parâmetro para o subtipo Integer. Sintaxe: CInt(expressão).
CStr	Converte seu parâmetro para o subtipo String. Sintaxe: CStr(expressão).
Chr	Retorna o caractere correspondente a um determinado código ASCII. Sintaxe: Chr(número).
Date	Retorna a data atual do sistema. Sintaxe: Date().
InputBox	Exibe uma caixa de diálogo solicitando ao usuário a entrada de dados. Sintaxe: InputBox(Mensagem [,título][,valor][,x][,y]).
Left	Retorna os caracteres *x* mais à esquerda. Sintaxe: Left(string, x).
Len	Retorna o tamanho de uma string. Sintaxe: Len(string).
MsgBox	Apresenta uma caixa de diálogo com informações. Sintaxe: MsgBox(Mensagem[,definição][,título]).
Now	Retorna a data e a hora atuais do sistema. Sintaxe: Now().
Right	Retorna os caracteres *x* mais à direita. Sintaxe: Right(string, x).
Time	Retorna a hora atual do sistema. Sintaxe: Time().
Trim	Elimina todos os espaços em branco antes e depois da string. Sintaxe: Trim(string).

Constantes/Literais	
Empty	Declara um valor para uma variável que ainda não foi inicializada.
Nothing	Elimina a atribuição de um objeto a uma variável.
Null	Usada na atribuição de valores nulos a variáveis.
True/False	Usadas em expressões booleanas.

Objetos	
Err	Objeto utilizado no tratamento de erros.
RegExp	Objeto que permite executar uma expressão regular.

Objeto Err

O objeto **Err** contém informações sobre erros de tempo de execução (run-time). Aceita os métodos **Raise** e **Clear** para criar e desativar erros run-time. Não é necessário criar uma instância para objeto **Err**.

No exemplo abaixo é verificado o valor da propriedade **Number** e, se ela contiver um valor diferente de zero, exibirá os detalhes numa caixa de mensagem.

```
Dim intNumeroDoErro, strDescDoErro
On Error Resume Next
Err.Raise 6
intNumeroDoErro = Err.number
strDescDoErro = Err.description
If intNumeroDoErro <> 0 Then
Wscript.Echo "Ocorreu um erro! Erro número " & intNumeroDoErro & " do tipo '" 
    & strDescDoErro & "'."
End If
```

Métodos do objeto Err

Clear

Define todas as propriedades de um objeto **Err** para os valores padrão, por exemplo, definindo a propriedade **Number** como 0 (zero).

Err.**Clear**

Raise

Permite simular um erro de run-time.

Err.**Raise** (*número* [,*fonte*] [,*descrição*] [,*arqhelp*][,*contexto*])

Argumento	Descrição
número	Natureza do erro.
fonte	Objeto ou aplicação que originalmente gerou o erro.
descrição	Descrição do erro.
arqhelp	Localização do arquivo Help no qual a ajuda para esse erro pode ser encontrada.
contexto	Identifica o tópico dentro do arquivo Help que fornece ajuda para o erro.

Propriedades do objeto Err

Description

Define ou retorna uma string que descreve o erro.

Err.**Description** [= *string*]

Number

Define ou retorna um valor numérico que identifica o erro.

Err.**Number** [= *número*]

Source

Retorna ou define o nome do objeto ou da aplicação que originalmente gerou o erro.

Err.**Source** [=*nomeaplic*]

Argumento	Descrição
nomeaplic	Nome da aplicação.

HelpContext

Define ou retorna o número de contexto para um tópico no arquivo Help.

Err.**HelpContext** [=*contextoID*]

Argumento	Descrição
contextoID	Identificador de um tópico de ajuda no arquivo Help.

HelpFile

Define ou retorna o path para um arquivo Help, que é acionado quando o usuário clica no botão Help ou pressiona a tecla F1 na caixa de diálogo de uma mensagem de erro.

Err.**HelpFile** [=*strCaminho*]

Argumento	Descrição
strCaminho	Localização do arquivo Help.

Objeto RegExp

O objeto **RegExp** é usado para criar e executar uma expressão regular, a qual permite pesquisar uma seqüência de caracteres em uma string.

Um objeto **Match** é criado cada vez que o objeto **RegExp** encontra uma correspondência (match). Zero ou mais correspondências (matches) poderão ocorrer e o objeto **RegExp** retornará uma coleção **Matches** de objetos **Match**.

Exemplo

```
strAlvo = "O rato roeu a roupa do rei de Roma"
Set objRegExp = New RegExp
objRegExp.Pattern = "r"
objRegExp.IgnoreCase = False
objRegExp.Global = True
Set colMatches = objRegExp.Execute(strAlvo)
For Each Match In colMatches
    Wscript.Echo "Correspondência encontrada em " & Match.FirstIndex & ". "
    Wscript.Echo "Valor encontrado é '" & Match.Value & "." & chr(10)
Next
```

Métodos do objeto RegExp

Execute

Executa a busca de uma expressão regular em uma string, retornando uma coleção **Matches** contendo um objeto **Match** para cada correspondência (match) encontrada. A propriedade **Pattern** deve ter sido especificada previamente com a expressão regular. Sintaxe:

Set *objRegExp* **= New RegExp**

objRegExp.**Execute(***string***)**

Replace

Executa a busca de uma expressão regular em uma string original (*str1*), substituindo cada correspondência (match) encontrada por uma outro string (*str2*). Se nenhuma correspondência for encontrada, retornará a string original.

Set *objRegExp* **= New RegExp**

objRegExp.**Replace (***str1, str2***)**

Test

Executa a busca de uma expressão regular em uma string e retorna um valor Booleano True ou False, indicando se a expressão regular foi encontrada.

Set *objRegExp* **= New RegExp**

objRegExp.**Test(***string***)**

Propriedades do objeto RegExp

Global

Retorna ou define um valor boleano, que indica se devem ser procuradas todas as ocorrências (True) da expressão regular ou somente a primeira ocorrência (False) em uma string.

Set *objRegExp* **= New RegExp**

objRegExp.**Global = True | False**

IgnoreCase

Define ou retorna um valor boleano que determina se a expressão regular distingue (False) ou não (True) minúsculas de maiúsculas.

Set *objRegExp* **= New RegExp**

objRegExp.**IgnoreCase = True | False**

Pattern

Define ou retorna uma string contendo a expressão regular. Deve ser especificada antes do objeto **RegExp** ser utilizado. Mais informações podem ser obtidas em http://msdn.microsoft.com/workshop/languages/clinic/scripting051099.asp ou em http://www.oreilly/catalog/regexp/.

Set *objRegExp* **= New RegExp**

objRegExp.**Pattern** [=*expr_regular*]

Coleção Matches

É uma coleção criada pelo método **Execute** do objeto **RegExp**, contendo objetos **Match** que são criados para cada correspondência (match) encontrada da expressão regular em uma string.

Objeto Match

O objeto **Match** é criado toda vez que o método **Execute** do objeto **RegExp** encontra uma correspondência (match) da expressão regular em uma string. Armazena em suas propriedades os detalhes de cada correspondência (match) encontrada na pesquisa.

Propriedades do objeto Match

FirstIndex

Retorna a posição (a partir de 0) dentro da string onde ocorreu a correspondência (match) com a expressão regular.

objMatch.**FirstIndex**

Length

Retorna o tamanho do texto encontrado na pesquisa da expressão regular.

objMatch.**Length**

Value

Retorna o valor ou o texto encontrado na pesquisa da expressão regular.

objMatch.**Value**

Símbolos usados em Expressões

Símbolo	Função
^	Somente localiza (match) no início de uma string. Por exemplo: "^P" localiza o primeiro "P" em "Pedro pegou o pato".
$	Somente localiza (match) no final de uma string. Por exemplo: a expressão "o$" localiza o último "o" em "Pedro pegou o pato".
\b	Localiza qualquer final de palavra. Por exemplo: "mente\b" localiza em "possivelmente amanhã".
\B	Localiza qualquer coisa que não seja final de palavra.
literal	Localiza o literal especificado.
\n	Localiza o caractere "new line"
\f	Localiza o caractere "form feed"
\r	Localiza o caractere "carriage return"
\t	Localiza o caractere "tab horizontal"
\v	Localiza o caractere "tab vertical"
\?	Localiza o caractere "?"
*	Localiza o caractere "*"
\+	Localiza o caractere "+"
\.	Localiza o caractere "."
|	Localiza o caractere "|"
\{	Localiza o caractere "{"
\}	Localiza o caractere "}"
\\	Localiza o caractere "\"
\[	Localiza o caractere "["
\]	Localiza o caractere "]"
\(	Localiza o caractere "("
\)	Localiza o caractere ")"
\xxx	Localiza o caractere ASCII expresso pelo número octal xxx. Por exemplo: "\50" localiza "(" ou chr (40).
\xdd	Localiza o caractere ASCII expresso pelo número hexa dd. Por exemplo: "\x28" localiza "(" ou chr (40).
\uxxxx	Localiza o caractere ASCII expresso pelo código Unicode xxxx. Por exemplo: "\u00A3" localiza "£".
[a-z]	Localiza qualquer caractere no intervalo especificado. Por exemplo: a expressão "[a-c]" localiza "a", "b"ou "c".
[xyz]	Localiza qualquer caractere especificado em [ ]. Por exemplo: "col[iu]na" localiza "colina" e "coluna".
[^xyz]	Localiza qualquer caractere que não esteja em [ ]. Por exemplo: "dr[^o]ga" localiza qualquer combinação, exceto "droga".
.	Localiza qualquer caractere exceto \n. Por exemplo: a expressão "a.b" localiza "aab" e "a3b", mas não "ab".
\w	Localiza letras e dígitos. Equivale a [a-zA-Z_0-9].
\W	Localiza qualquer caractere que não seja letra ou dígito. Equivale a [^a-zA-Z_0-9].

\d	Localiza qualquer dígito. Equivale a [0-9].
\D	Localiza qualquer não-dígito. Equivale a [^0-9].
\s	Localiza qualquer caractere espaço. Equivale a [\t\r\n\v\f].
\S	Localiza qualquer caractere diferente de espaço. Equivale a [^\t\r\n\v\f].
{x}	Localiza exatamente x ocorrências de uma expressão regular. Por exemplo: "a{3}" equivale a "aaa".
(x,}	Localiza x ou mais ocorrências de uma expressão regular. Por exemplo: "\s{2,}" localiza pelo menos 2 caracteres espaço.
{x,y}	Localiza entre x e y ocorrências de uma expressão regular. Por exemplo: "\d{2,3}" localiza pelo menos 2, mas não mais que 3 dígitos.
?	Localiza zero ou uma ocorrência do caractere precedente. Equivale a {0,1}. Por exemplo: a expressão "casas?" localiza ambos "casa" e "casas".
*	Localiza zero ou mais ocorrências do caractere precedente. Equivale a {0,}. Por exemplo: a expressão "fo*" localiza ambos "f" e "foo".
+	Localiza uma ou mais ocorrências do caractere precedente. Equivale a {1,}. Por exemplo: a expressão "fo+" localiza "foo" mas não "f".
()	Agrupa uma cláusula para criar uma nova cláusula. Pode ser aninhada. Por exemplo: "(ab)?(c)" localiza "abc" or "c".
|	Localiza qualquer uma das cláusulas individuais. Por exemplo: a expressão "a|b" localiza "a" ou "b:".

Informações Adicionais

Sobre o Autor

Roberto Gomes de Aguiar Veiga é natural de Belo Horizonte, Minas Gerais, cidade onde reside. Trabalha como Técnico de Informática no Serviço Federal de Processamento de Dados (Serpro), empresa pública da área de informática vinculada ao Ministério da Fazenda. Está lotado atualmente no Centro de Especialização BackOffice da Superintendência de Atendimento a Clientes do Serpro.

Dentre suas qualificações, as mais importantes são a de Microsoft Certified Systems Engineer (MCSE) e Microsoft Certified Professional + Internet (MCP + Internet).

O autor pode ser contatado através do e-mail raveiga@yahoo.com.

Sobre Este Guia

Este guia busca ser uma obra de referência sucinta sobre um assunto sobre o qual não há títulos disponíveis nas livrarias. É incrível como o mercado editorial brasileiro, na área de informática, restringe-se a alguns poucos assuntos. Eu conheço pelo menos uma centena de livros sobre Windows e Visual Basic e dúzias de livros sobre Linux e Delphi, mas não conheço nenhum outro, em língua portuguesa, sobre o WSH que não este modesto guia.

Quando decidiu adquirir este guia, provavelmente o que o motivou foi o fato de que, assim como eu, você também acredita no grande potencial do WSH dentro das redes empresariais baseadas na plataforma Windows. Acredito que o WSH vá se tornar tão importante para o Windows quanto o ASP é importante para o Internet Information Server, por isso me dediquei tanto ao seu aprendizado. Acho que o leitor também fará o mesmo e que este guia lhe será bastante útil!

Palavra Final do Autor

Para os profissionais que decidiram dedicar suas vidas à informática, o que vou dizer agora pode até parecer lugar comum: manter-se atualizado em novas tecnologias é uma questão de continuar ou não valioso para o mercado de trabalho – e de continuar empregado também! Portanto, espero que o leitor, como profissional de informática que é, não veja este guia como algo definitivo sobre WSH. Consulte principalmente os sites que indiquei ao longo deste guia em busca de novidades. Eu lhes asseguro, elas existirão sempre.

Críticas e sugestões serão bem-vindas (todo mundo sempre diz isso!) e podem ser enviadas ao e-mail informado abaixo.

Por fim, espero sinceramente que este guia auxilie você, meu caro leitor, nas suas tarefas cotidianas. Acredite em mim, foi isso que me motivou a escrevê-lo.

Um grande abraço,

O Autor.

Notação utilizada neste Guia

Notação	Significado
opção1 | *opção2* ...	Barras verticais separam itens alternativos em uma lista.
[ ]	Identifica valores opcionais que podem ou não ser fornecidos pelo usuário. Digite somente a informação dentro dos parênteses; não digite os parênteses.
itálico	Identifica um valor que deve ser fornecido pelo usuário.
negrito	Identifica palavras reservadas.

Comentários e Sugestões

Comentários e sugestões sobre este guia serão bastante apreciados. Podem ser enviados para o e-mail:

leitor@novateceditora.com.br

Conheça o site da Novatec Editora em:

www.novateceditora.com.br

Símbolos

A

B

C

D

E

F

G

H

I

K