Programação VBScript

REDES DE COMPUTADORES

Prof. Dr. João Paulo Aramuni

UNIVERSIDADE FUMEC

Trabalhando com objetos

* O VBScript permite a administradores de rede criar scripts complexos usando recursos de programação avançados, como árvores de decisão, loop, tratamento de erros e a capacidade de chamar funções e sub-rotinas. Porém, o VBScript não inclui funções intrínsecas à execução de tarefas de administração de sistemas.

Trabalhando com objetos

* O VBScript possui funções internas para a determinação da raiz quadrada de um número ou do valor ASCII de um caractere por exemplo, mas não possui funções internas para a interrupção de serviços, para a recuperação de eventos em logs de eventos ou para a execução de outras tarefas do interesse de administradores de sistemas.

Usando objetos COM

* Felizmente, há maneiras de se executar essas tarefas por meio de programação. Isso é feito basicamente através do uso de objetos de automação. Objetos de automação são um subconjunto de **COM** (modelo de objeto componente), uma forma padrão de aplicativos (arquivos .exe) ou bibliotecas de programação (arquivos .dll) apresentarem seus recursos como uma série de objetos.

* Por sua vez, os programadores (ou escritores de script) podem utilizar esses objetos -- e os recursos do aplicativo ou da biblioteca de programação -- em seus próprios projetos.

Usando objetos COM

* Por exemplo, um aplicativo de processamento de texto pode expor o verificador ortográfico como um objeto de automação, fornecendo aos escritores de script uma maneira de adicionar a verificação ortográfica a seus projetos.

* A capacidade de trabalhar com objetos de automação e utilizar as propriedades e os métodos desses objetos torna o VBScript uma ferramenta poderosa para administração de redes.

Usando objetos COM

* O VBScript sozinho não pode ler eventos em um log; no entanto, ele pode usar os recursos incluídos na WMI para recuperar esses eventos. O VBScript não possui funções intrínsecas à criação de contas de usuário no Active Directory; porém, a linguagem pode usar os recursos da ADSI para criar essas contas.

Usando objetos COM

* Na verdade, o VBScript é frequentemente chamado de "linguagem cola", pois uma de suas aplicações principais é "colar" objetos.

* Em vez de fornecer um número infinito de funções intrínsecas dedicadas à administração de sistemas, o VBScript fornece duas funções, **GetObject** e **CreateObject**, e os elementos de linguagem necessários para o uso dos métodos e das propriedades de objetos de automação.

GetObject

* Retorna um objeto de automação de um arquivo. Pode ser usado para ler, alterar ou inserir dados em arquivos que aceitem esse tipo de automação, 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])

GetObject

* **Parâmetros**

* ***arquivo:*** Arquivo de onde será retornado o objeto de automação.
* ***progID:*** String contendo o tipo do objeto a ser retornado.
* ***Prefix:*** Chama um procedimento dentro do script cujo nome é formado por *prefixo* mais o caractere "_", mais o nome de um evento gerado pelo objeto.

GetObject

* Exemplo
  * Abre um arquivo do Excel chamado exemplo.xls:
    * **Dim obj**
    * **Set obj = GetObject (“C:\exemplo.xls”)**

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.

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

CreateObject

* **Parâmetro**

* ***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.

CreateObject

* Exemplos
  * Criando um objeto do excel
    * **Dim objExcel**
    * **Set objExcel = CreateObject("Excel.Application")**
  * Criando um objeto chamado WshShell
    * **Dim WshShell**
    * **Set WshShell = Wscript.CreateObject(“Wscript.Shell”)**

Usando objetos COM

* O script mostrado a seguir ilustra a importância de objetos de automação no VBScript.

* Esse script relata a quantidade de espaço livre em disco na unidade C do computador local. E, ainda, ele faz isso usando pouco código VBScript. Em vez disso, o script:

Usando objetos COM

* 1. Usa a função VBScript GetObject para se conectar à WMI através da biblioteca de scripts WMI (um objeto de automação).

* 2. Usa o método Get, fornecido pelo objeto de automação WMI, para recuperar as propriedades da unidade C.

* 3. Usa o método Echo do Windows Script Host (WSH) para relatar a quantidade de espaço livre em disco na unidade C. Aliás, WSH é apenas outro objeto de automação.

Usando objetos COM

* Set objWmiService = **GetObject**("winmgmts:")
* Set objLogicalDisk = objWmiService.Get ("Win32_LogicalDisk.DeviceID='C:'")
* WScript.Echo objLogicalDisk.FreeSpace

* No exemplo anterior, o objetivo principal do VBScript era colar os recursos da WMI e do WSH. Isso permitiu a você recuperar o espaço livre em disco (algo que o WSH não pode fazer sozinho) e exibir o valor de volta para a tela (algo que a WMI não pode fazer sozinha).

WSH – Windows Script Host

* O Vbscript é uma linguagem de programação interpretada, ao invés de ser compilada como programas mais conhecidos de programação, como C++, Pascal e o próprio Visual Basic.

* Para que um arquivo em Vbscript seja reconhecido e corretamente executado, o Windows possui um ambiente específico para reconhecer e executar as tarefas descritas no script. Esse ambiente é chamado de WSH (Windows Script Host).

* Quando um arquivo .vbs é executado, quem faz toda a interpretação das linhas de comando é o WSH.

WSH – Windows Script Host

* O WSH possui um componente chamado de *core object model* (modelo de objeto de núcleo) que fornece ao Vbscript um acesso direto aos recursos do Windows. Alguns exemplos:
    - Desktop
    - Menu Iniciar
    - Aplicativos do Windows
    - Arquivos de sistemas do Windows
    - Windows Quick Launch Toolbar
    - Impressoras de rede
    - Drives de rede
    - Registro do Windows

WSH – Windows Script Host

* O Sistema Operacional Windows pode ser visto como uma coleção de objetos. Por exemplo, um arquivo é um objeto, assim como uma pasta, uma impressora, etc. O que o *core object model* faz é expor esses objetos em um formato que permite aos scripts reconhecê-los, acessá-los e manipulá-los.

* Cada objeto está associado a propriedades e métodos que os scripts podem usar para interagir com o objeto. Por exemplo, um arquivo é um objeto em que o nome e a sua extensão são duas de suas propriedades.

WSH – Windows Script Host

* O WSH permite aos scripts acessar esse arquivo e alterar o seu nome ou a sua extensão. Um arquivo também possui métodos associados a eles, como, por exemplo, operações de copiar, colar e mover. Usando esses métodos, os scripts podem mover um arquivo de uma pasta para outra ou de um computador para outro.

Objetos Intrínsecos

* Alguns objetos são intrínsecos. Objetos intrínsecos são os objetos criados sem nunca ter sido necessário efetuar uma chamada para GetObject ou CreateObject.

* Veja o exemplo a seguir:

    * Wscript.Echo(“Hello World”)

Objetos Intrínsecos

* Observe que nenhuma sequência de caracteres semelhante é usada para criar uma referência ao objeto Wscript do WSH no exemplo.

* Em vez disso, o método Echo é chamado sem a criação anterior de qualquer tipo de objeto WSH. Isso ocorre porque WScript é um objeto intrínseco. Não é necessário criar um objeto Wscript, pois Wscript será criado automaticamente quando você executar um script de VBScript.

WSH – Windows Script Host

* No *WSH core object model* há 14 objetos. Cada um desses objetos fornece acesso a uma categoria em particular dos recursos do Windows.

* Vejamos algumas informações sobre cada um deles:

Objeto WScript

* É o objeto raiz do WSH. Fornece acesso à um número de propriedades e métodos que dizem respeito ao próprio script. Também fornece acesso ao restante dos objetos do *WSH core object model*.

* **Propriedades**: Arguments, FullName, Interactive, Name, Path, ScriptFullName, ScriptName, StdErr, StdIn, StdOut, and Version.

* **Métodos**: ConnectObject(), CreateObject(), DisconnectObject(), Echo(), GetObject(), Quit(), and Sleep().

Objeto WshArguments

* Habilita o acesso aos argumentos da linha de comando passado para o script na hora da execução.

* **Propriedades:** Count, Item, and Length, Named, and Unnamed.

* **Métodos:** Count() and ShowUsage().

Objeto WshNamed

* Fornece acesso à um conjunto de argumentos de linhas de comandos nomeados.
* **Propriedades:** Item and Length.
* **Métodos:** Count() and Exists().

Objeto WshUnnamed

* Esse objeto fornece acesso à um conjunto não nomeado de argumentos de linhas de comando.

* **Propriedades:** Item and Length.

* **Métodos:** Count().

Objeto WshController

* Fornece a capacidade de criar um processo remoto de script.
* **Propriedades:** This object does not support any properties.
* **Métodos:** CreateScript.

Objeto WshRemote

* Permite administrar remotamente um sistema de computadores.
* **Propriedades:** Status and Error.
* **Métodos:** Execute() and Terminate().

Objeto WshRemoteError

* Fornece acesso à informações sobre erros produzidos por scripts remotos.

* **Propriedades:** Description, Line, Character, SourceText, Source, and Number.

* **Métodos:** This object does not support any methods.

Objeto WshNetwork

* Acessa um número de diferentes recursos da rede, como impressoras e drives.

* **Propriedades**: ComputerName, UserDomain, and UserName.

* **Métodos**: AddWindowsPrinterConnection(), AddPrinterConnection(), EnumNetworkDrives(), EnumPrinterConnection(), MapNetworkDrive(), RemoveNetworkDrive(), RemovePrinterConnection(), and SetDefaultPrinter().

Objeto WshShell

* Acessa o registro do Windows, eventos de log, variáveis do ambiente, atalhos e aplicações.

* **Propriedades**: CurrentDirectory, Environment, and SpecialFolders.

* **Métodos**: AppActivate(), CreateShortcut(), ExpandEnvironmentStrings(), LogEvent(), Popup(), RegDelete(), RegRead(), RegWrite(), Run(), SendKeys(), and Exec().

Objeto WshShortcut

* Fornece métodos e propriedades para criar e manipular atalhos.
* **Propriedades**: Arguments, Description, FullName, Hotkey, IconLocation, TargetPath, WindowStyle, and WorkingDirectory.
* **Método**: Save().

Objeto WshUrlShortcut

* Fornece métodos e propriedades para criar e manipular atalhos URL.

* **Propriedades:** FullName and TargetPath.

* **Método**: Save().

Objeto WshEnvironment

* Fornece acesso a variáveis de ambiente do Windows.
* **Propriedades:** Item and Length.
* **Métodos:** Remove() and Count().

Objeto WshSpecialFolders

* Fornece acesso às pastas especiais do Windows, como o do menu iniciar, favoritos e outras pastas especiais.

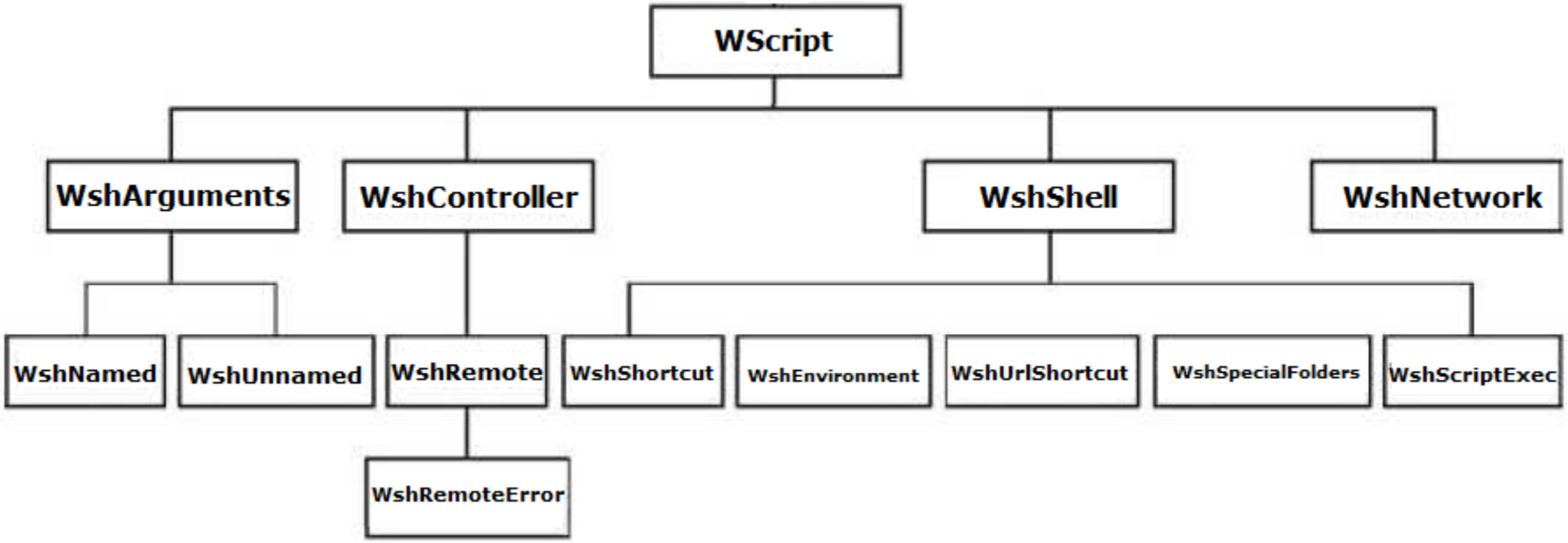
* **Propriedades:** Item.

* **Métodos:** Count().

Objeto WshScriptExec

* Fornece acesso às informações de erro de scripts que estejam usando o método Exec() do WshShell.
* **Propriedades:** Status, StdOut, StdIn, and StdErr.
* **Métodos:** Terminate().

Relação hierárquica entre os objetos



WSH – Windows Script Host

* No topo da figura está a raiz do *WSH core object model* que é o objeto Wscript. Todos os outros objetos são instanciados (fundamentados) a partir desse objeto. O objeto Wscript é automaticamente estabelecido pelo WSH de modo que ele pode ser utilizado nos scripts sem ser instanciado.

WSH – Windows Script Host

* O Wscript é um objeto referenciado como *Public*, assim como WshController, WshShell e WshNetwork. Esses três últimos devem ser instanciados utilizando o método CreateObject() do Wscript. Todos os demais objetos da figura podem apenas ser instanciados com o uso das propriedades ou métodos de Wscript, WshController, WshShell ou WshNetwork.

* A tabela a seguir apresenta o resto dos objetos da *WSH core object model,* assim como o método utilizado para instanciá-los.

WSH – Windows Script Host

Objeto	Método de instanciação
WshArguments	WScript.Arguments
WshNamed	WScript.Arguments.Named
WshUnnamed	WScript.Arguments.Unnamed
WshRemote	WshController.CreateScript()
WshRemoteError	WshRemote.Error
WshShortcut	WshShell.CreateShortcut()
WshUrlShortcut	WshShell.CreateShortcut()
WshEnvironment	WshShell.Environment
WshSpecialFolders	WshShell.SpecialFolders
WshScriptExec	WshShell.Exec()

Primeiro VBS com objetos

* Para ficar mais claro considere o seguinte exemplo:

```
'Obtém o nome do Computador:
set WshNetwork=Wscript.CreateObject("Wscript.Network")
strComputerName=WshNetwork.ComputerName
Wscript.Echo strComputerName
```

Primeiro VBS com objetos

* Outro exemplo mais elaborado:

```
Set Rede = WScript.CreateObject("WScript.Network")
PropertyInfo = "Domínio do usuário" & vbTab & "= " & Rede.UserDomain & _
vbCrLf & "Computador" & vbTab & "= " & Rede.ComputerName & _
vbCrLf & "Nome do usuário" & vbTab & "= " & Rede.UserName & vbCrLf
MsgBox(PropertyInfo)
```

Primeiro VBS com objetos

* A primeira linha do script acima é o que estamos chamando de instanciação. Assim, **Rede** é uma variável (cujo nome é definido pelo programador) que está associada a uma instância do objeto WshNetwork.

* Após o estabelecimento da instância, é possível acessar as propriedades e métodos do objeto fazendo uso da variável associada. Assim, as propriedades UserDomain, ComputerName e UserName do objeto WshNetwork podem ser acessadas.

Primeiro VBS com objetos

* Para testar, basta criar um arquivo, inserir o código descrito, salvar com a extensão **.vbs** e clicar duas vezes sobre ele.

Exercícios

* exer12_computerName.vbs
* exer13_network.vbs
* exer14_dir.vbs
* exer15_calc.vbs
* exer16_notepad.vbs
* exer17_word.vbs
* exer18_var_ambiente.vbs
* exer19_hd.vbs
* exer20_listDrivers.vbs

Desafio

* Utilizando o objeto WScript.Shell, escreva um script vb que execute o shutdown.exe da pasta system32 do Windows.

* Dica: Você precisará pesquisar sobre o método **Run** do objeto WScript.Shell.

UNIVERSIDADE FUMEC
DE MINAS GERAIS PARA O MUNDO

Desafio

* Solução:

exer21_shutdown.vbs

UNIVERSIDADE FUMEC
DE MINAS GERAIS PARA O MUNDO

Obrigado.

Contato: joaopauloaramuni@gmail.com

UNIVERSIDADE FUMEC

50 Anos