

## - Windows PowerShell - Utilización de WPS - Introducción -

Algunas cuestiones prácticas de carácter general relativas a la utilización de WPS deben ser conocidas por sus usuarios.

### Historia de los comandos

La consola de WPS guarda los últimos 64 comandos tecleados en una History, la que puede ser listada con "Get-History".

Con el parámetro "-Count" puede listarse un número determinado de comandos, a saber, los últimos n comandos:

```
Get-History -count 12
```

De la misma forma puede llamarse un comando determinado a partir de la posición del mismo en esa historia

```
Invoke-History 7
```

La cantidad de comandos en dicha historia puede ser definido en la variable "\$MaximumHistoryCount".

La historia de comandos puede ser exportada en forma de archivo script o como un archivo XML.

El archivo script se usa cuando se pretende mantener la ejecución de los comandos en el orden sucesivo en que fueron tecleados.

Por su parte, el archivo XML se utiliza cuando se desee reestablecer la historia de los comandos sin ejecutarlos.

	Archivo script	Archivo XML
Exportar	Get-History -count 7   format-table commandline - HideTableHeader   Out-file "c:\miscscript.ps1"	Get-History   Export-CliXml "d:\scripts\histora.xml"
Importar (y ejecutar)	."c:\miscscript.ps1"	Import-CliXml "d:\scripts\historia.xml"   Add-History

### Informaciones de Host y sistema

El cmdlet "Get-Host" y la variable "\$Host" ofrece información sobre el entorno actual de WPS. Tanto el cmdlet como la variable brindan una instancia de la clase "System.Management.Automation.Internal.Host.Internal-Host". "InternalHost" contiene informaciones y permite, además, modificaciones a través del subobjeto "UI.RawUI", por ejemplo:

- > \$Host.Name: Nombre del Host (con esto es posible una diferenciación del entorno. Por ejemplo, PowerShell Host brinda otro valor que la consola normal de WPS)
- > \$Host.Version: versión del entorno
- > \$Host.UI.RawUI.WindowTitle = "Titulo": establece el nombre de la ventana
- > \$Host.UI.RawUI.ForegroundColor = [System.ConsoleColor]::White: establece el color del texto
- > \$Host.UI.RawUI.BackgroundColor = [System.ConsoleColor]::DarkBlue: establece el color de fondo del texto.

"Get-Culture" (o "\$Host.CurrentCulture") y "Get-UICulture" (o "\$Host.CurrentUICulture") brindan información sobre el idioma actual como instancias de la clase "System.Globalization.CultureInfo.Get-Culture", la cual se refiere a la salida de fecha, hora y mensajes. "Get-UICulture" se refiere al idioma de la interfase del usuario. Generalmente, la anotación en ambos son idéntica, pero el usuario puede hacer modificaciones.

### Borrar la visualización

"Clear-Host" (alias clear) borra el contenido de la consola de WPS, pero no elimina la historia de los comandos.

### Configuración del perfil para la consola de WPS

Al terminar la sesión, la consola "olvida" los elementos incorporados a la misma (ejemplo: Snap-Ins cargados, alias y funciones definidas, PowerShell-Provider integrados y la historia de comandos). Con la ayuda del archivo de perfil es posible devolverle a WPS los elementos "olvidados".

Los perfiles son archivos script con el nombre "Profile" y la extensión ".ps1"

Un "Profile.ps1" puede existir en dos niveles:

-> Global para todos los usuarios en el directorio de instalación de WPS (normalmente, en "c:\windows\system32\windowspowershell\v1.0\")

-> relativo a un usuario en la carpeta de sistema (en Vista: c:\users\nombre del usuario\documents\windowspowershell)

### Enumeración

Para la utilización de algunas clases .NET (como "FileSystemRights" para administrar derechos relativos al sistema de archivos), es necesario combinar diversas banderas "flags" con un "o" (or) binario.

WPS puede buscar en una cadena de caracteres, con una coma como separador, el valor correspondiente del flag en la enumeración y combinarlo con un "or" binario. En lugar de

```
$Rights= [System.Security.AccessControl.FileSystemRights]::Read'
-bor [System.Security.AccessControl.FileSystemRights]::ReadExtendedAttributes'
-bor [System.Security.AccessControl.FileSystemRights]::ReadAttributes'
-bor [System.Security.AccessControl.FileSystemRights]::ReadPermissions
```

puede escribirse

```
$Rights = [System.Security.AccessControl.FileSystemRights] "ReadData,
ReadExtendedAttributes, ReadAttributes, ReadPermissions"
```