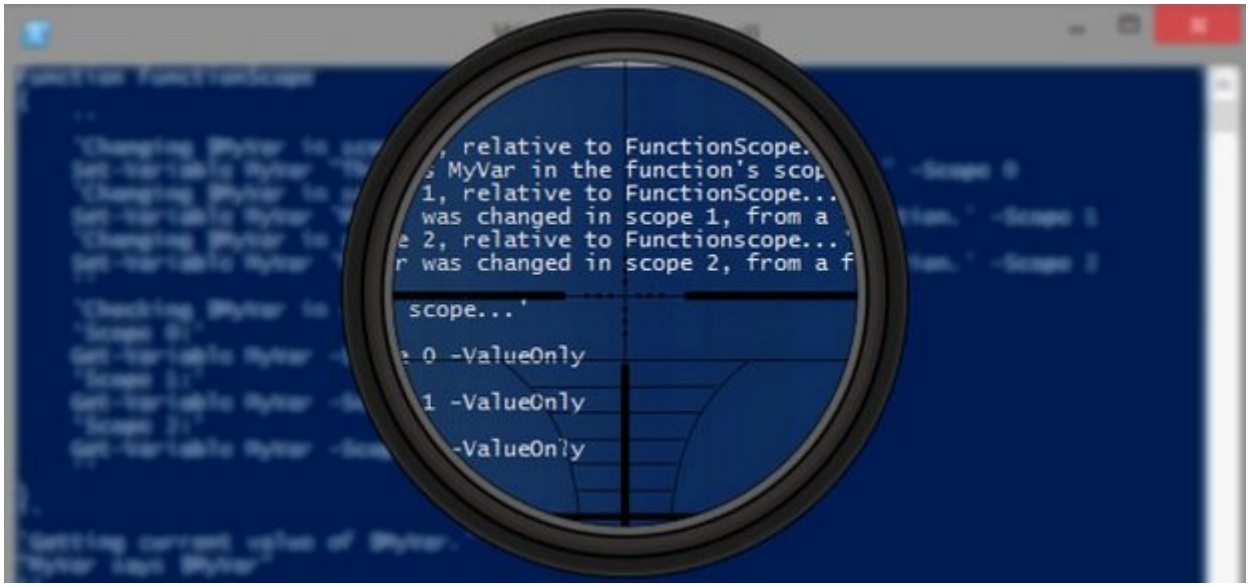
Cómo Afectan Los Alcances De Un Script La PowerShell

En las **secuencias de comandos por lotes**, los cambios en las variables de entorno tienen un impacto global en la sesión actual de forma predeterminada. Para PowerShell, es lo opuesto debido a los ámbitos que se utilizan para aislar las modificaciones de un script. Aquí, vamos a explorar cómo los **alcances de un Script afectan la PowerShell** y cómo trabajar en y alrededor de estos.



¿Qué son los alcances de un

Script?

En PowerShell, un «alcance» se refiere al **entorno actual en el que una secuencia de comandos** o shell de comandos está operando. Los alcances se utilizan para proteger ciertos objetos en el entorno de ser modificado involuntariamente por los scripts o funciones. En particular, las siguientes cosas están protegidos frente a posibles modificaciones por los comandos ejecutados desde otro ámbito, a menos que se especifique lo contrario por los parámetros en los comandos:

- Variables
- Alias
- funciones
- Unidades PowerShell (PSDrives)

Nuevos ámbitos se crean cada vez que se ejecuta un script o función, o cuando se crea una nueva sesión o **instancia de la PowerShell**. Los alcances creados mediante la ejecución de secuencias de comandos y funciones tienen una relación «padre/hijo» con el alcance de la que fueron creados. Hay unos pocos ámbitos que tienen un significado muy especial, y se puede acceder por nombre:

- El alcance global es el alcance que se crea cuando se inicia la PowerShell. Incluye las variables, alias, funciones y PSDrives que están integrados a PowerShell así como cualquier que son hechos por su perfil de PowerShell.
- El ámbito local se refiere a lo que es el alcance actual. Al iniciar PowerShell se refieren al ámbito global, dentro de un script que será el alcance de secuencias de comandos, etc.
- Ámbitos privados pueden ser definidos dentro del ámbito actual, para evitar que los comandos en otros ámbitos

puedan leer o modificar los elementos de lo que podrían tener acceso.

Los alcances también pueden hacer referencia a este número en ciertos comandos, en el ámbito actual se conoce como cero y sus antepasados son referenciados por el aumento de los números enteros. Por ejemplo, dentro de un script que se ejecuta desde el ámbito global, el alcance de secuencias de comandos sería 0 y el alcance global sería 1. Un ámbito que se anida además dentro del alcance de la secuencias de comandos, como una función, se referiría al ámbito global como 2.

¿Cómo afecta los alcances a los comandos?

Como se mencionó anteriormente, los comandos ejecutados dentro de un ámbito no afectarán a las cosas en otro ámbito a menos que se especifique lo contrario. Por ejemplo, si **$MiVar** existe con un alcance global y una secuencia de comandos se ejecuta un comando para establecer **$MiVar** a un valor diferente, la versión global de **$MiVar** permanecerá inalterada mientras que una copia de **$MiVar** se coloca en el ámbito de aplicación de secuencias de comandos con el nuevo valor. Si **$MiVar no existe**, un script lo creará en el ámbito de secuencias de comandos por defecto, no en el ámbito global. Esto es importante recordar a medida que aprende acerca de la relación real de padre / hijo entre ámbitos.

El ámbito de la relación padre/hijo en PowerShell es de un solo sentido. Los comandos se pueden ver, y opcionalmente modificar el alcance actual, su padre, y cualquier alcances superior. Sin embargo, ellos no pueden ver o modificar las cosas en aquellos alcances que heredan del alcance actual.

Existe un montón de casos en los que necesitas de un script o cambios de función los cuales se conserven después de su finalización, pero no tantos en que había necesidad de realizar cambios en los objetos antes o después de que se ejecute el ámbito de la función de secuencia de comandos.

Por supuesto, ¿cuáles son las reglas sin excepciones? Una excepción a lo anterior son los **alcances privados**. Los objetos en los ámbitos privados sólo son accesibles a los comandos que se ejecutan en el ámbito de la aplicación en la que fueron creados. Otra excepción importante es los artículos que tienen la propiedad AllScope. Éstas son variables especiales y alias para el que un cambio en cualquier ámbito afectará a todos los ámbitos. Los siguientes comandos le mostrará cuáles son las variables y los alias tienen la propiedad AllScope:

[bash]

```
Get-Variable | Where-Object {$_.Options -match 'AllScope'}
Get-Alias | Where-Object {$_.Options -match 'AllScope')
```

[/bash]

Alcances de un Script en Acción!

Para nuestro primer vistazo a los ámbitos en acción, vamos a empezar en una sesión de PowerShell donde la variable $MyVar está establecida en una cadena, 'Soy una variable global!', Desde la línea de comandos. Entonces, el siguiente script se ejecuta desde un archivo llamado Scope-Demo.ps1:

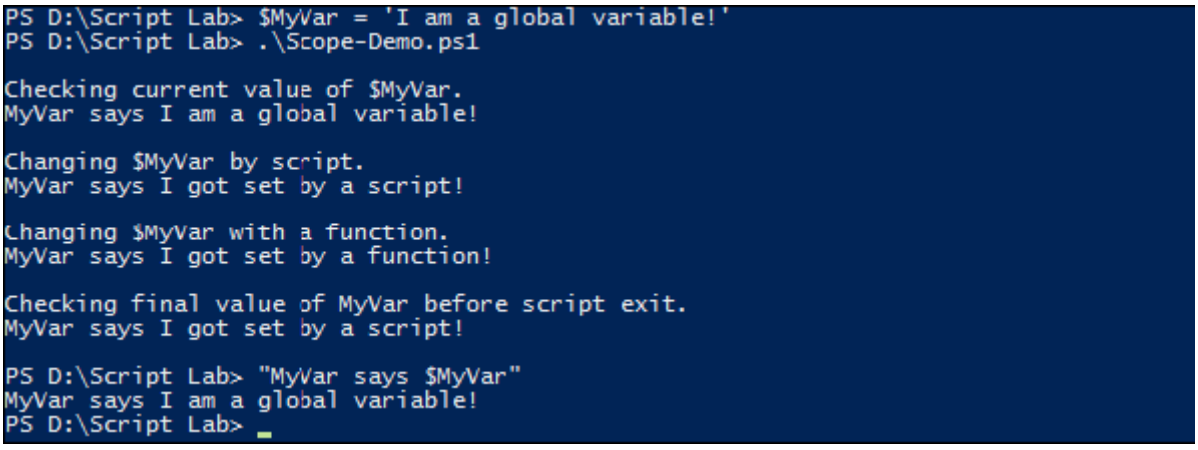
[bash]

```
Function FunctionScope
{
```

```
‘Changing $MyVar with a function.’
$MyVar = ‘I got set by a function!’
"MyVar says $MyVar"
}
»
‘Checking current value of $MyVar.’
"MyVar says $MyVar"
»
‘Changing $MyVar by script.’
$MyVar = ‘I got set by a script!’
"MyVar says $MyVar"
»
FunctionScope
»
‘Checking final value of MyVar before script exit.’
"MyVar says $MyVar"
»
```

[/bash]

A continuación podemos apreciar la salida del anterior código:



Como se puede ver la variable pareció cambiar a medida que avanzábamos a través de la escritura, ya que, hasta que se completó la **función FunctionScope**, hicimos el registro de la variable desde dentro del mismo ámbito se modificó por última vez. Después **FunctionScope** se ejecuto, sin embargo, nos mudamos de nuevo en el ámbito de script donde $MyVar fue

dejado intacto por la función. Entonces, cuando el script termino, volvimos a cabo en el ámbito global en el que no se ha modificado en absoluto.

Información adicional

Todavía hay mucho más que se puede hacer con los alcances, los cuales no se mencionan en el articulo. Los alcances afectan más que las variables, y todavía hay más por aprender acerca de los alcances privados y las variables AllScope. Para obtener información más útil, puede ejecutar el **siguiente comando desde PowerShell**:

[bash]Get-Help about_scopes[/bash]