Los jobs en segundo plano: Start-Job, Receive-Job, Remove-Job

Aunque sólo está disponible con PowerShell v2, ya es posible ejecutar comandos y scripts en segundo plano (o de forma asíncrona) sin interacción con la consola. Esta característica es particularmente interesante cuando se tiene un script bastante largo a ejecutar, ya que la ejecución en segundo plano devuelve el mando inmediatamente a la consola sin bloquearla. Las tareas realizadas en segundo plano son lo que se denomina «jobs» en PowerShell.

Para conocer el conjunto de comandos relacionados con la utilización de los jobs, escriba el comando siguiente:

```
PS > Get-Command *Job* -Type cmdlet


CommandType Name        Definition
----------- ----        ----------
Cmdlet      Get-Job     Get-Job [[-Id] <Int32[]>] [-Verbose] [-Debug] [-E...
Cmdlet      Receive-Job Receive-Job [-Job] <Job[]> [[-Location] <String[]...
Cmdlet      Remove-Job  Remove-Job [-Id] <Int32[]> [-Force] [-Verbose] [-...
Cmdlet      Start-Job   Start-Job [-ScriptBlock] <ScriptBlock> [[-Initial...
Cmdlet      Stop-Job    Stop-Job [-Id] <Int32[]> [-PassThru] [-Verbose] [...
Cmdlet      Wait-Job    Wait-Job [-Id] <Int32[]> [-Any] [-Timeout <Int32>...
```

Commandlet	Descripción
Get-Job	Comando que permite listar todas las tareas que se ejecutan en segundo plano.
Receive-Job	Comando que permite obtener el o los resultados de las tareas que han sido ejecutadas en segundo plano.
Remove-Job	Comando que permite suprimir las tareas que se ejecutan en segundo plano.
Start-Job	Comando que permite iniciar una tarea en segundo plano.
Wait-Job	Comando que permite esperar a que una o varias tareas finalicen para devolver el control.

Cuando un Job concluye su ejecución, no devuelve nada a la consola donde se ha iniciado, pero en su lugar el resultado de la ejecución se guarda en un objeto de tipo «job». Bastará entonces con manipular el objeto para recuperar el contenido; contenido que puede ser sólo parcial si el job no ha terminado completamente su ejecución.

```
PS > Start-Job -scriptblock {get-service}

Id   Name   State      HasMoreData     Location            Command
--   ----   -----      -----------     --------            -------
1    Job1   Running    True            localhost           get-service
```

Como puede observar, la utilización del comando Start-Job en su versión básica es relativamente sencilla. Bastará con añadirle el argumento scriptblock e insertar un bloque de ejecución. Después de introducir el comando, recuperamos el acceso a la consola sin esperar el final de la ejecución del mismo. Vemos que el estado del comando está actualmente en ejecución (State: running). Observamos también que los jobs están identificados no sólo por un número de identificación (ID) sino también por un nombre.

Ejemplo:

Tomemos como ejemplo uno pequeño bucle que varía de 1 a 10 donde mostraremos buenos días 1, buenos días 2, ..., buenos días 10. Vamos a ejecutar este bucle en segundo plano con el comando siguiente:

```
PS > Start-Job -Name Job_Tabla -ScriptBlock {1..10 | Foreach { Write-Host
"buenos días $_" }}

Id   Name        State      HasMoreData    Location           Command
--   ----        -----      -----------    --------           -------
5    Job_Tabla   Running    True           localhost          1..10 |
forea...
```

Ahora podemos observar gracias al commandlet **`Get-Job`** la lista de jobs en segundo plano en ejecución o finalizados:

```
PS > Get-Job

Id    Name          State      HasMoreData    Location          Command
--    ----          -----      -----------    --------          -------
1     Job1          Completed  True           localhost         get-service
5     Job_Tabla     Completed  True           localhost         1..10 |
forea...
```

Vemos que el estado ha cambiado y que nuestro job ha terminado (*Completed*). Para obtener el resultado de éste o más bien la visualización del resultado podemos utilizar el comando: **`Receive-Job`**

```
PS > Get-Job -Id 1 | Receive-Job

Status   Name               DisplayName
------   ----               -----------
Stopped  AeLookupSvc        Experiencia de aplicación...
Stopped  ALG                Servicio de la pasarela d...
Stopped  AppIDSvc           Identidad de la aplicación...
...
```

O también, filtrando por el nombre, en la tarea Job_Tabla por ejemplo:

```
PS > Get-Job -Name Job_Tabla | Receive-Job

buenos días 1
buenos días 2
buenos días 3
buenos días 4
buenos días 5
buenos días 6
buenos días 7
buenos días 8
buenos días 9
buenos días 10
```

Ahora, con el fin de liberar memoria debemos suprimir el job con el comando **`Delete-PsJob`**. Si no lo hacemos, el job permanece en el caché de la consola; no obstante será de todos modos destruido al cierre de la consola.

```
PS > Remove-Job -Id 5
```

O también, de otra forma:

```
PS > Remove-Job -Name Job_Tabla
```

Por último, si desea borrar todos los jobs:

```
PS > Remove-Job *
```

También es posible ejecutar comandos o scripts PowerShell en máquinas remotas. No es cuestión de abrir una consola interactiva a distancia; pero si únicamente de poder ejecutar comandos o scripts a distancia de manera no interactiva. Este punto se abordará en el capítulo Ejecución remota.