Internacionalización de los scripts

En la versión 1 de PowerShell, la edición de scripts destinados a un público integrado por personas de diferentes nacionalidades no es algo evidente, el editor se debe traducir manualmente todo el contenido textual de sus scripts. En PowerShell v2, aparece lo que llamamos «la internacionalización de script». El principio es permitir la escritura de scripts en diferentes idiomas sin modificar el código contenido en el script. Por ejemplo, si crea un script y éste debe ser ejecutado por varias personas, y cada una de ellas utiliza una versión de PowerShell diferente en términos de idioma (variable **`$PSUICulture`** diferente). El hecho de utilizar la internacionalización, permitirá a los usuarios obtener toda la parte textual del script en su idioma, y sin ninguna manipulación por su parte. Consideremos ahora cómo funciona.

La primera precaución que debemos tomar es, evidentemente, la de separar los datos (data) del código contenido en el script. Para hacer esto, utilizaremos una novedad de PowerShell que es la sección «Data». En el interior de esta sección es donde utilizaremos un nuevo commandlet de PowerShell v2 llamado **`ConvertFrom-StringData`**, que va a crear una tabla de hash de las cadenas de caracteres.

Ejemplo:

```
$TextoScript = Data {
          ConvertFrom-StringData @'
          Mensaje_1 = Buenos días
          Mensaje_2 = Entre un valor
          Mensaje_3 = Entre otro valor
          Mensaje_4 = El resultado de la suma de estos dos valores es:
'@
}
```

Observe que utilizamos aquí un **`Here-String`**. Un Here-String (véase el capítulo Descubra PowerShell) empieza con un separador @' y termina por '@ (el último separador debe ir obligatoriamente precedido de un retorno de carro). Todos los caracteres entre los delimitadores @' y '@ son considerados como texto puro.

Para permitir la traducción de estos «data», es necesario que una traducción en diferentes idiomas debe se guarde en ficheros .psd1 y bajo una estructura de directorios especial.

Los archivos .psd1 deben guardarse en un subdirectorio con formato <idioma>-<País >, como indica la variable **`$PSUICulture`**. De este modo, si el script principal llamado MiScript.ps1 se encuentra en el directorio C:\Temp, los archivos MiScript.psd1 se encontrarán bajo la estructura siguiente:

```
C:\Temp\MiScript.ps1         # Script Principal
C:\Temp\en-US\MiScript.psd1  # Archivo de datos traducido al inglés
C:\Temp\fr-FR\MiScript.psd1  # Archivo de datos traducido al francés
...
```

Ejemplo del contenido de los archivos .psd1:

Archivo C:\Temp\en-US\MiScript.psd1.

```
ConvertFrom-StringData @'
           Message_1 = Hello
           Message_2 = Enter a value
           Message_3 = Enter a second value
           Message_4 = The result of the addition of these two values is:
'@
```

Archivo C:\Temp\fr-FR\MiScript.psd1

```
ConvertFrom-StringData @’
           Message_1 = Bonjour
           Message_2 = Entrez une valeur
           Message_3 = Entrez une autre valeur
           Message_4 = Le résultat de l’addition de ces deux valeurs est :
’@
```

Por último, la última etapa, es decir, permitir la importación de cadenas de caracteres en el idioma de la interfaz de usuario vía el commandlet **`Import-LocalizedData`**. La utilización de este commandlet importa el archivo .psd1 correspondiente al idioma utilizado, y hace transparentes las acciones de traducción de los elementos textuales del script. El script completo pasa a ser el siguiente:

```
$TextoScript = Data {
  #Culture es-ES
  ConvertFrom-StringData @’
     Mensaje_1 = Buenos días
     Mensaje_2 = Entre un valor
     Mensaje_3 = Entre otro valor
     Mensaje_4 = El resultado de la suma de estos dos valores es:
’@
}

Import-LocalizedData TextoScript

# Inicio del script

Write-host $TextoScript.Mensaje_1
Write-host $TextoScript.Mensaje_2
[int]$var1 = Read-host
Write-host $TextoScript.Mensaje_3
[int]$var2 = Read-host
$resultado = $var1 + $var2
Write-host "$($TextoScript.Mensaje_4) $resultado"

# Fin del script
```

Al ejecutar el script anterior sobre Windows versión US, obtenemos el resultado siguiente:

```
PS > C:\temp\MiScript.ps1
Hello
Enter a value
5
Enter a second value
6
The result of the addition of these two values is: 11
```