Set-Variable

Set the value of a variable.

```
Syntax
      Set-Variable [-Name] String[] [-value] Object]
           [-Include string[]] [-Exclude string[]] [-option option]
              [-scope string] [-force] [-passThru] [-whatIf]
                 [-Description string] [-Visibility {Public | Private} ]
                    [-confirm] [CommonParameters]
Key
   -Name String
       The name of the variable(s), may be piped.

   -Value Object
       The value to assign to the variable.

   -Include string
       Items upon which Set-variable will act, excluding all others.

   -Exclude string
       Items upon which Set-variable will not act, include all others.

   -Option option
       The valid options are:
         None    : Set no options. (default)
         ReadOnly: The alias cannot be changed unless you use -Force.
         Constant: The alias cannot be changed, even by using -Force.
         Private : The alias is available only within the scope specified b
                   It is invisible in all other scopes.
         AllScope: The variable is copied to any new scopes that are create

   -Scope string
       The scope in which this alias is valid.
       Valid values are "Global", "Local", "Private" or "Script", or a numb
       relative to the current scope ( 0 through the number of scopes, wher
       0 is the current scope and 1 is its parent). "Local" is the default.
       For more, type "get-help about_scope".

   -force
       create a variable with the same name as an existing read-only variab
       or change the value of a read-only variable.

       By default, a variable may be overwritten, unless it has an option v
       of "ReadOnly" or "Constant". For more information, see the -Option p

   -passThru
       Pass the object created by this cmdlet through the pipeline.

   -Visibility {Public | Private}
       Whether the variable is visible outside of the session in which it w
       This parameter is designed for  use in scripts and commands that
       will be delivered to other users.

       When a variable is private, it does not appear in lists of variables
       those returned by Get-Variable, or in displays of the Variable: driv
       Commands to read or change the value of a private variable return an

       However, the user can run commands that use a private variable if th
```

```
          commands were written in the session in which the variable was defin

      -WhatIf
          Describe what would happen if you executed the command without actua
          executing the command.

      -Confirm
          Prompt for confirmation before executing the command.

      -Description string
          The description of the variable.
```

Standard Aliases for Set-Variable: `set,` **`sv`**

Examples

Set the value of the "SS64" variable to "My Sample text ":

```
PS C:\> set-variable -name SS64 -value "My Sample text"
```

Create a global, read-only variable that contains all processes on the system:

```
PS C:\> set-variable -name myprocs -value (Get-Process) -option constant
-scope global
```

In the example above, the value is enclosed in parentheses this executes the command: (Get-Process) before storing the result in the variable, rather than just storing the text "Get-Process".

“O, swear not by the moon, the fickle moon, the inconstant moon, that monthly changes in her circle orb, Lest that thy love prove likewise variable” ~ Shakespeare

Related PowerShell Cmdlets:

Clear-Variable - Remove the value from a variable.
Get-Variable - Get a PowerShell variable.
New-Variable - Create a new variable.
Remove-Variable - Remove a variable and its value.
Environment Variables
Equivalent bash command: env - Display, set, or remove environment variables.