Especifica uma parte do corpo de uma função, junto com as

palavras-chave Begin, Process e End. Os parâmetros dinâmicos

são adicionados no tempo de execução.

Sintaxe:

function <nome> {

dynamicparam {<lista de instruções>}

begin {<lista de instruções>}

process {<lista de instruções>}

end {<lista de instruções>}

}

Else

----

Usado com a palavra-chave If para especificar a lista de

instrução padrão.

Sintaxe:

if (<condição>) {<lista de instruções>}

else {<lista de instruções>}

Elseif

------

Usado com as palavras-chave If e Else para especificar outros

condicionais. A palavra-chave Else é opcional.

Sintaxe:

if (<condição>) {<lista de instruções>}

elseif (<condição>) {<lista de instruções>}

else {<lista de instruções>}

End

---

Especifique uma parte do corpo de uma função, junto com as

palavras-chave Dynamicparam, Begin e End. A lista de instruções

End é executada uma vez depois que todos os objetos forem

recebidos do pipeline.

Sintaxe:

function <nome> {

dynamicparam {<lista de instruções>}

begin {<lista de instruções>}

process {<lista de instruções>}

end {<lista de instruções>}

}

Exit

----

Faz com que o Windows PowerShell saia de um script ou uma

instância do Windows PowerShell.

Sintaxe:

exit

Filter

------

Especifica uma função na qual a lista de instruções é executada

uma vez para cada objeto de entrada. Ele tem o mesmo efeito que

uma função que contém somente um bloco Process.

Sintaxe:

filter <nome> {<lista de instruções>}

Finally

-------

Define uma lista de instruções que é executada depois que as

instruções são associadas a Try e Catch. Uma lista de

instruções Finally é executada mesmo se você pressionar CTRL+C

para sair de um script ou se você usar a palavra-chave Exit no

script.

Sintaxe:

try {<lista de instruções>}

catch [<error type] {<lista de instruções>}

finally {<lista de instruções>}

For

---

Define um loop utilizando uma condição.

Sintaxe:

for (<inicializar>; <condição>; <iterar>) {<lista de

instruções>}

Foreach

-------

Define um loop utilizando cada membro de uma coleção.

Sintaxe:

foreach (<item> in <coleção>){<lista de instruções>}

From

-----

Reservado para uso futuro.

Function

--------

Cria uma lista de instruções nomeada de código reutilizável.

Você pode nomear o escopo ao qual uma função pertence. Você

também pode especificar um ou mais parâmetros nomeados utilizando a

palavra-chave Param. Com a lista de instruções da função, você pode

incluir as listas de instrução Dynamicparam, Begin, Process e End.

Sintaxe:

function [<scope:>]<name> {

param ([type]<$pname1> [, [type]<$pname2>])

dynamicparam {<statement list>}

begin {<statement list>}

process {<statement list>}

end {<statement list>}

}

Você também tem a opção de definir um ou mais parâmetros fora

da lista de instruções depois do nome da função.

Sintaxe:

function [<scope:>]<name> [([type]<$pname1>, [[type]<$pname2>])] {

dynamicparam {<statement list>}

begin {<statement list>}

process {<statement list>}

end {<statement list>}

}

If

--

Define uma condicional.

Sintaxe:

if (<condição>) {<lista de instruções>}

In

--

Usado em uma instrução Foreach para criar um loop que usa cada

membro de uma coleção.

Sintaxe:

foreach (<item> in <coleção>){<lista de instruções>}

Param

-----

Define os parâmetros em uma função.

Sintaxe:

function [<scope:>]<name> {

param ([type]<$pname1>[, [[type]<$pname2>]])

<statement list>

}

Process

-------

Especifique uma parte do corpo de uma função, junto com as

palavras-chave Dynamicparam, Begin e End. Quando uma lista de

instruções Process recebe uma entrada do pipeline, ela é

executada uma vez para cada elemento do pipeline. Se o pipeline não

fornecer nenhum objeto, a lista de instruções Process não será

executada. Se o comando for o primeiro comando no pipeline, a lista

de instruções Process será executada uma vez.

Sintaxe:

function <nome> {

dynamicparam {<lista de instruções>}

begin {<lista de instruções>}

process {<lista de instruções>}

end {<lista de instruções>}

}

Return

------

Faz com que o Windows PowerShell saia do escopo atual, como um

script ou uma função e grava a expressão opcional na saída.

Sintaxe:

return [<expressão>]

Switch

------

Especifica uma variedade de ações a ser executada em itens a

partir do pipeline ou de um arquivo. Você pode usar um dos

seguintes modelos de sintaxe.

Sintaxe 1:

switch [-regex|-wildcard|-exact][-casesensitive] ( pipeline )

{

<cadeia de caracteres>|<número>|<variável>|{<expressão> }

{<lista de instruções>}

<cadeia de caracteres>|<número>|<variável>|{<expressão> }

{<lista de instruções>}

...

default {<lista de instruções>}

}

Sintaxe 2:

switch [-regex|-wildcard|-exact][-casesensitive] -file filename

{

<cadeia de caracteres>|<número>|<variável>|{<expressão> } {<lista de instruções>}

<cadeia de caracteres>|<número>|<variável>|{<expressão> } {<lista de instruções>}

...

default {<lista de instruções>}

}

Throw

-----

Lança um objeto como um erro.

Sintaxe:

throw [<objeto>]

Trap

----

Define uma lista de instruções para ser executada se um erro for

encontrado. Um tipo de erro requer colchetes. O segundo par de

colchetes indica que o tipo de erro é opcional.

Sintaxe:

trap [[<tipo de erro>]] {<lista de instruções>}

Try

---

Define uma lista de instruções a serem verificadas quanto a

erros enquanto as instruções são executadas. Se ocorrer um

erro, o Windows PowerShell continua em execução em uma instrução

Catch ou Finally. Um tipo de erro requer colchetes. O segundo par

de colchetes indica que o tipo de erro é opcional.

Sintaxe:

try {<lista de instruções>}

catch [[<tipo de erro>]] {<lista de instruções>}

finally {<lista de instruções>}

Until

-----

Usado em uma instrução Do como uma construção de loop em que a

lista de instruções é executada pelo menos uma vez.

Sintaxe:

do {<lista de instruções>} until (<condição>)

While

-----

Usado em uma instrução Do como uma construção de loop em que a

lista de instruções é executada pelo menos uma vez.

Sintaxe:

do {<lista de instruções>} while (<condição>)

CONSULTE TAMBÉM

about\_escape\_characters

about\_special\_characters

about\_wildcards

C:\Users\PRTS\Documents\GitHub [master +4 ~0 -0 !]>