Pętla WHILE - info

Jak działa pętla WHILE

Dlaczego blok poleceń za pętlą WHILE może sie wcale nie wykonać?

Łączenie instrukcji WHILE z wyrażeniami warunkowymi

Działanie polecenia BREAK

Typowe zastosowania instrukcji WHILE

Ogólna zasada pętli WHILE jest taka, aby wykonywać polecenia lub blok poleceń znajdujących się poniżej warunku tak długo jak długo warunek zapisany przy WHILE jest prawdziwy.

Możliwe jest, że gdy w skrypcie umieścisz pętle WHILE to nie wykona się ona ani razu, było by tak wtedy, gdy warunek za pierwszym razem jest nieprawdziwy, jest to jeden z częstszych błędów, ale czasami działanie zamierzone.

```
1   $i=0
2
3   while($i -lt 10)
4   {
5   $i++
6   Write-Host "$i"
7   }
```

W przykładzie zadeklarowaliśmy zmienną $i=0 od której wszystko w naszym skrypcie będzie zależało

Na początku przypisujemy jej wartość 0

Następnie za każdym wejściem do pętli będziemy sprawdzać czy $i jest -lt (mniejszy) niż 10

Jeśli tak to $i będzie zwiększany o 1 i następnie wyświetlany. Za pierwszym razem wyświetli się 1, za drugim 2, itd. do 10, po wyświetleniu 10 w pętli WHILE będzie sprawdzone czy $i jest mniejsze niż 10 i tym razem wartość tego testu to fałsz, więc pętla się zakończy

```
1   $i=0
2
3   while($i -lt 10)
4   {
5   $i++
6   Write-Host "$i"
7   }
```

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

Przykład bardziej złożony

Pętlę będziemy wykonywać 10 razy sprawdzając czy $i jest mniejsze od 10. Wyświetlany wewnątrz komunikat będzie zależny od tego czy wartość $i w danej chwili jest podzielna przez 2 ($i%2 -eq 0), jeśli jest równe to wyświetli się napis "$i liczba jest parzysta" w przeciwnym wypadku komunikat "$i liczba jest nieparzysta"

```
1   $i=0
2
3   while($i -lt 10)
4   {
5   $i++
6   if($i%2 -eq 0)
7       {
8           Write-Host "$i liczba jest parzysta"
9       }
10  Else
11      {
12          Write-Host "$i liczba jest nieparzysta"
13      }
14  }
```

```
1 liczba jest nieparzysta
2 liczba jest parzysta
3 liczba jest nieparzysta
4 liczba jest parzysta
5 liczba jest nieparzysta
6 liczba jest parzysta
7 liczba jest nieparzysta
8 liczba jest parzysta
9 liczba jest nieparzysta
10 liczba jest parzysta
```

Poniższy przykład prezentuje działanie polecenia Break (nie jest napisany zgodnie z najlepszymi praktykami)

Pętla ma się wykonać 100 razy sprawdza czy $i jest mniejsze niż 100

$i za każdym razem jest powiększany o jeden co sugeruje, że pętla wykona się 1000 razy

Wyświetlany wewnątrz komunikat będzie zależny od tego czy wartość $i w danej chwili jest podzielna przez 2 ($i%2 -eq 0), jeśli jest równe to wyświetli się napis "$i liczba jest parzysta" w przeciwnym wypadku komunikat "$i liczba jest nieparzysta"

Różnica jest w ostatniej części skryptu. Gdy $i jest równy 10 to wyświetli się komunikat, że „Doszliśmy do '10', więc przerywam pętlę” Przerwanie pętli będzie zrealizowane przez polecenie Break, mimo że ($i i warunek pętli wskazują na to, że pętla ma się wykonywać to instrukcja Break spowoduje natychmiastowe wyjście z pętli while bez dodatkowego sprawdzania warunku zadeklarowanego na początku.

```powershell
$i=0

while($i -lt 100)
{
$i++
if($i%2 -eq 0)
    {
        Write-Host "$i liczba jest parzysta"
    }
Else
    {
        Write-Host "$i liczba jest nieparzysta"
    }
If($i -eq 10)
    {
        Write-Host "Doszliśmy do '10', więc przerywam pętlę"
        Break
    }
}
```

```
1 liczba jest nieparzysta
2 liczba jest parzysta
3 liczba jest nieparzysta
4 liczba jest parzysta
5 liczba jest nieparzysta
6 liczba jest parzysta
7 liczba jest nieparzysta
8 liczba jest parzysta
9 liczba jest nieparzysta
10 liczba jest parzysta
Doszliśmy do '10', więc przerywam pętlę
```

While jest instrukcją przydatną, gdy chcesz wykonać pewne czynności tak długo jak długo utrzymuje się pewien stan. Można np. badać ilość dostępnej pamięci tak jak działa pewien program, można spowodować, że zatrzyma się na tak długo aż inny serwer zacznie odpowiadać na polecenie ping, można uruchamiać usługę aż jej status będzie uruchomiana lub przekroczymy dopuszczalna ilość prób uruchomienia usługi.

Przedstawiłem

Jak działa pętla WHILE

Dlaczego blok poleceń za pętlą WHILE może się wcale nie wykonać?

Łączenie instrukcji WHILE z wyrażeniami warunkowymi

Działanie polecenia BREAK

Typowe zastosowania instrukcji WHILE