Skrypty, funkcje, moduły – info

Zobaczysz, jak zbudować parametryzowany skrypt

Jakie znaczenie mają dyrektyw stosowane dla funkcji i parametrów

Jak zbudować własny help

Jak stworzyć funkcję

Jak załadować funkcję znajdującą się w pliku do pamięci

Jak stworzyć moduł

Działanie write-verbose i write-debug podczas debuggowania skryptów

Mamy do zrealizowania zadanie wysyłania wiadomości tekstowych email z poziomu PowerShell

Posłużymy się obiektem Net.Mail.SMTPClient

Zaczynamy od zadeklarowania zmiennych

```
1  $EmailFrom = "magazynpamieci@gmail.com"
2  $EmailTo = "kontakt@promail.com"
3  $Subject = "Test emila z powershella"
4  $Body = "Przepraszam to tylko test"
5  $SMTPServer = "smtp.gmail.com"
6  $SMTPClient = New-Object Net.Mail.SMTPClient($$SMTPServer, 587)
7  $SMTPClient.EnableSsl = $true
8  $SMTPClient.Credentials = New-Object System.Net.NetworkCredential("magazynpamieci", "Dobreh@sl0");
9  $SMTPClient.Send($EmailFrom, $EmailTo, $Subject, $Body)
10
```

Niezbyt dobra praktyka umieszczanie hasła w skrypcie.

W ostatniej linii wysyłamy email korzystając z przygotowanych wcześniej zmiennych $EmailFrom, $EmailTo, $Subject, $Body.

Załóżmy, że wysyłać będziemy dość często, email ma być wysyłany zawsze z tego samego konta a zamieniać będziemy tylko do kogo wysyłamy, jaki jest temat i treść wiadomości.

Z widocznych powyżej instrukcji można przygotować funkcje a później wywoływać funkcje przekazując do niej trzy parametry do, temat i treść

```
1  [CmdletBinding()]
2  Param(
3      [Parameter(Mandatory=$True)]
4      [string] $EmailTo,
5      [string] $Subject="Departament IT informacja",
6      [Parameter(Mandatory=$True)]
7      [string] $Body
8      )
9
10 $EmailFrom = "magazynpamieci@gmail.com"
11
12 $SMTPServer = "smtp.gmail.com"
13 $SMTPClient = New-Object Net.Mail.SMTPClient($SMTPServer, 587)
14 $SMTPClient.EnableSsl = $true
15 $SMTPClient.Credentials = New-Object System.Net.NetworkCredential("magazynpamieci", "Dobreh@sl0");
16 $SMTPClient.Send($EmailFrom, $EmailTo, $Subject, $Body)
```

Nie jest to jeszcze definicja funkcji a całkiem dobrze przygotowany skrypt

W nagłówku pojawia się [CmdletBilding()] – powershel będzie stosował do skryptu mechanizmy zabezpieczające między innymi przed powstawaniem banalnych błędów nie będzie można przekazywać do skryptu parametru który nie był by przez ten skrypt przyjmowany co chroni przed literówkami powstałymi po stronie użytkownika.

Następnie mamy instrukcje Param dzięki niej PowerShell wie jakich parametrów oczekuje nasz skrypt. W bloku Param można definiować dodatkowe właściwości tych parametrów np.

[Parameter(Mandatory =$True)] – oznacza parametr konieczny do uruchomienia skryptu

$EmailTo, - jest obowiązkowy,

$Subject – jest nieobowiązkowy ma wartość domyślną "Departament IT informacja" jeśli użytkownik nie poda tego parametru to parametr przyjmie wartość domyślną,

[Parameter(Mandatory=$True)]

$Body – oznacza, że nie pozwolimy na wysyłanie pustych emaili, dla body nie zdefiniowałem wartości domyślnej.

Reszta skryptu wygląda tak samo jak poprzedni z tym, że usunąłem deklaracje zmiennych, które są w parametrach skryptu.

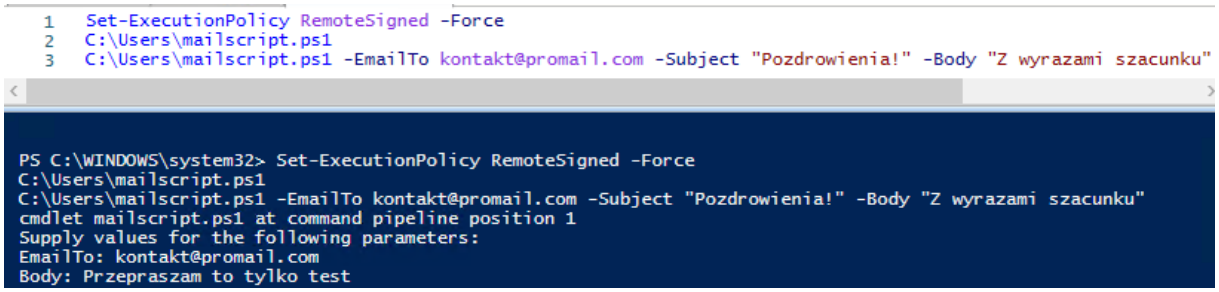
Przygotowany skrypt należy zapisać na dysku.

Jak korzystać z powyższego skryptu

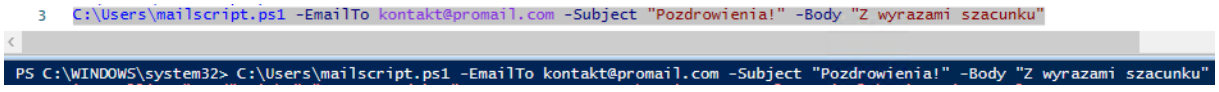
Ponieważ skrypt nie jest podpisany to

Set-ExecutionPolicy RemoteSigned -Force

C:\Users\mailscript.ps1



C:\Users\mailscript.ps1 -EmailTo kontakt@promail.com -Subject "Pozdrowienia!" -Body "Z wyrazami szacunku"



Dobrym zwyczajem jest pisanie tak skryptów, aby nawet użytkownik nie będący autorem mógł zdebagować skrypt i znaleźć linie, która w określonej sytuacji generuje błąd

W celu realizacji powyższego można się posłużyć Write-Verbose który powoduje wyświetlenie komunikatu na ekranie, ale nie zawsze, aby skrypt został wyświetlony skrypt musi być uruchomiony z przełącznikiem -Verbose

```
11
12    $SMTPServer = "smtp.gmail.com"
13  ● Write-Verbose "Tworzenie klienta SMTP ..."
14    $SMTPClient = New-Object Net.Mail.SMTPClient($
15    $SMTPClient.EnableSsl = $true
16    $SMTPClient.Credentials = New-Object System.Ne
17  ● Write-Verbose "Wysłanie emaila ..."
18    $SMTPClient.Send($EmailFrom, $EmailTo, $Subjec
19  ● Write-Verbose "Email wysłany"
```

Dodanie -Verbose powoduje, że ekran został zasypany dodatkowymi informacjami diagnostycznymi, gdyby doszło do jakiegoś błędu to komunikaty wskaż w której linii kodu dochodzi do błędu.

```
4   C:\Users\mailscript.ps1 -EmailTo kontakt@promail.com -Subject "Pozdrowienia!" -Body "Z wyrazami szacunku" -Verbose
```

```
    + FullyQualifiedErrorId : SmtpException


PS C:\WINDOWS\system32> C:\Users\mailscript.ps1 -EmailTo kontakt@promail.com -Subject "Pozdrowienia!" -Body "Z wyrazami szacunku" -
VERBOSE: Tworzenie klienta SMTP ...
VERBOSE: Wysłanie emaila ...
Exception calling "Send" with "4" argument(s): "Serwer SMTP wymaga bezpiecznego połączenia lub nie uwierzytelniono klienta. Odpo
wiedź serwera: 5.7.0 Authentication Required. Learn more at"
At C:\Users\mailscript.ps1:18 char:1
+ $SMTPClient.Send($EmailFrom, $EmailTo, $Subject, $Body)
+ ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
    + CategoryInfo          : NotSpecified: (:) [], MethodInvocationException
    + FullyQualifiedErrorId : SmtpException

VERBOSE: Email wysłany
```

Jak to się stało, że do skryptu można przesłać parametr -Verbose którego w skrypcie nie deklarowaliśmy to zasługa [CmdletBinding()] która jest na początku.

Dobry skrypt powinien być dobrze udokumentowany, to nie tylko komentarze również dobry help funkcji i skryptów. Komentarz musi być umieszczony przed nazwą funkcji bez dodatkowej pustej linii, ma specyficzną składnie.

```
1  <#
2  .SYNOPSIS
3  Send email.
4  .DESCRIPTION
5  Wysyła emaila na podany w EmailTo adres.
6  .PARAMETER EmailTo
7  Wprowadź emaila do wysłania
8  .PARAMETER Subject
9  Wprowadź tytuł
10 .PARAMETER Body
11 Wprowadź wiadomość
12 .EXAMPLE
13 .\mailscript.ps1 -EmailTo kontakt@promail.com -Subject "Pozdrowienia!" -Body "Z wyrazami szacunku" -Verbose
14 #>
15   [CmdletBinding()]
16 Param(
17     [Parameter(Mandatory=$True)]
18     [string] $EmailTo,
19     [string] $Subject="Departament IT informacja",
20     [Parameter(Mandatory=$True)]
21     [string] $Body
22     )
23
```

Blok .SYNOPSIS zawiera ogólny opis tego co robi skrypt

.DESCRIPTION opisz dokładnie co chcesz aby użytkownik dowiedział się o twoim skrypcie czy funkcji

W sekcji .PARAMETER możesz opisać poszczególne parametry, po słowie PARAMETER wpisujesz nawę parametru a pod opis tego parametru, zrób tak dla każdego parametru.

Jeśli chcesz umieść sekcję .EXAMPLE w której możesz przykład lub przykłady uruchamiania twojego skryptu lub funkcji.

Bardzo często użytkownicy korzystają z sekcję .EXAMPLE dlatego warto ją budować.

Aby zobaczyć jak korzystać z opisu wpisz

Fragment SYNTAX został wygenerowany automatycznie.

Pozostałe bloki zostały pobrane z danych wprowadzonych.

Help C:\Users\mailscript.ps1

```
5   Help C:\Users\mailscript.ps1
```

```
PS C:\WINDOWS\system32> Help C:\Users\mailscript.ps1

NAME
    C:\Users\mailscript.ps1

SYNOPSIS
    Send email.


SYNTAX
    C:\Users\mailscript.ps1 [-EmailTo] <String> [[-Subject] <String>] [-Body] <String> [<CommonParameters>]


DESCRIPTION
    Wysyła emaila na podany w EmailTo adres.


RELATED LINKS

REMARKS
    To see the examples, type: "get-help C:\Users\mailscript.ps1 -examples".
    For more information, type: "get-help C:\Users\mailscript.ps1 -detailed".
    For technical information, type: "get-help C:\Users\mailscript.ps1 -full".
```

Help C:\Users\mailscript.ps1 -Detailed

```
5   Help C:\Users\mailscript.ps1 -detailed
```

```
PS C:\WINDOWS\system32> Help C:\Users\mailscript.ps1 -detailed

NAME
    C:\Users\mailscript.ps1

SYNOPSIS
    Send email.


SYNTAX
    C:\Users\mailscript.ps1 [-EmailTo] <String> [[-Subject] <String>] [-Body] <String> [<CommonParameters>]


DESCRIPTION
    Wysyła emaila na podany w EmailTo adres.


PARAMETERS
    -EmailTo <String>
        Wprowadź emaila do wysłania

    -Subject <String>
        Wprowadź tytuł

    -Body <String>
        Wprowadź wiadomość

    <CommonParameters>
        This cmdlet supports the common parameters: Verbose, Debug,
        ErrorAction, ErrorVariable, WarningAction, WarningVariable,
        OutBuffer, PipelineVariable, and OutVariable. For more information, see
        about_CommonParameters (https:/go.microsoft.com/fwlink/?LinkID=113216).

    -------------------------- EXAMPLE 1 --------------------------

    PS C:\>.\mailscript.ps1 -EmailTo kontakt@promail.com -Subject "Pozdrowienia!" -Body "Z wyrazami szacunku" -Verbose
```

Dodatkowo została wyświetlona sekcja

PARAMETERS

EXAMPLE

Jak z skryptu zrobić funkcje

Należy dodać

Function Send-EmailFromGmail { jak poniżej

```
#>
Function Send-EmailFromGmail {
[CmdletBinding()]
Param(
```

…

I zamknąć nawias klamrowy

} jak poniżej

```
Write-Verbose "Email wysłany"
}
```

Aby skorzystać z funkcji należy wczytać skrypt do pamięci

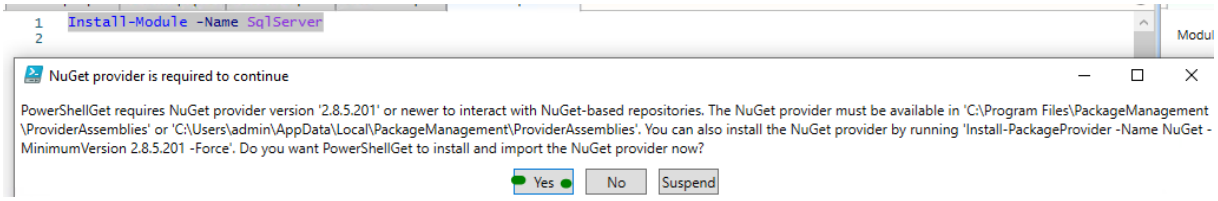
```
. C:\Users\mailscript.ps1
```

W jednym pliku skryptu można zdefiniować więcej funkcji.

Moduły pozwalają powiększyć ilość funkcji, moduły mogą być tworzone przez nas, wtedy będą zawierały funkcje automatyzujące nasze środowisko.

Moduły mogą być również dostarczane wraz z instalowanym na komputerze oprogramowaniem np.:

Install-Module -Name SqlServer



Następnie wybierz All

Pokaż zainstalowane moduły i funkcje

```
Install-Module -Name SqlServer
Get-Module *Sql* -ListAvailable
Import-Module SqlServer
dir "C:\Program Files\WindowsPowerShell\Modules\SqlServer\21.1.18256"
```

```
# Define function
function SQLSERVER: { Set-Location SQLSERVER: }

# Define aliases for backward compatibility
Set-Alias -Name Encode-SqlName -Value ConvertTo-EncodedSqlName
Set-Alias -Name Decode-SqlName -Value ConvertFrom-EncodedSqlName

# Set default values for variables used by the provider
Set-Variable -scope Global -name SqlServerMaximumChildItems -Value 0
Set-Variable -scope Global -name SqlServerConnectionTimeout -Value 30
Set-Variable -scope Global -name SqlServerIncludeSystemObjects -Value $false
Set-Variable -scope Global -name SqlServerMaximumTabCompletion -Value 1000


# SIG # Begin signature block
```

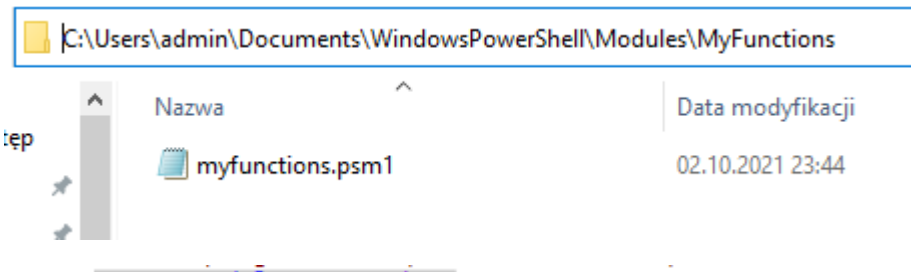
```
-a----         13.07.2021     18:36          16854 SqlServer.psm1
-a----         13.07.2021     18:36          13615 SqlServerPostScript.ps1
```

Jeżeli chcesz utworzyć swój własny moduł to należy go umieścić w odpowiednim katalogu.

Można sprawdzić listę przewidzianych do tego miejsc odwołując się do zmiennej środowiskowej

```
Wiersz polecenia
Microsoft Windows [Version 10.0.19042.1237]
(c) Microsoft Corporation. Wszelkie prawa zastrzeżone.

C:\Users\admin>echo %PSModulePath%
C:\Program Files\WindowsPowerShell\Modules;C:\WINDOWS\system32\WindowsPowerShell\v1.0\Modules
```

Nie zaleca się umieszczania swoich modułów w tych folderach, tam umieszczamy moduły SO a nie użytkowników. Umieszczamy je w




```
Import-Module MyFunctions
```

```
PS C:\WINDOWS\system32> Import-Module MyFunctions
```

Sprawdzamy wysyłając emaila

Zauważ, że parametry nazwa funkcji help są od razu dostępne.

```
Send-EmailFromGmail -EmailTo ktos@zsl.gda.pl -Body "to działa"
```

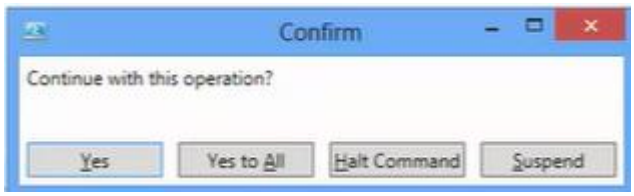
```
PS C:\WINDOWS\system32> Import-Module MyFunctions

PS C:\WINDOWS\system32> Send-EmailFromGmail -EmailTo ktos@zsl.gda.pl -Body "to działa"
```

Największym problemem w korzystaniu z modułów jest debagowanie znajdujących się w nim funkcji

Kiedy chcesz zatrzymać skrypt zaraz po wysłaniu emaila posłuż się poleceniem Write-Debug spowoduje to przerwanie wykonywania funkcji w tym miejscu, po przerwaniu będzie można sprawdzić stan funkcji w danej chwili, wartość zmiennych a nawet skorygować lub przerwać wykonanie funkcji.

```
$SMTPClient.Send($EmailFrom, $EmailTo, $Subject, $Body)
Write-Debug -Message "tera debagujesz po wysłaniu emaila "
Write-Verbose "Email wysłany"
}
```

```
6   Send-EmailFromGmail -EmailTo ktos@zsl.gda.pl -Body "to działa" -Verbose -Debug
```



Widać zmienne itd.

Aby wyjść z trybu wpisz exit

Poza tymi mamy graficzny debager środowiska ISE tylko w trybie graficznym

Jeżeli skrypt działa w trybie znakowym do debagowania stosu tryb znakowy

Zobaczyłeś' jak zbudować parametryzowany skrypt

Jakie znaczenie mają dyrektywy stosowane dla funkcji i parametrów

Jak zbudować własny help

Jak stworzyć funkcję

Jak załadować funkcję znajdującą się w pliku do pamięci

Jak stworzyć moduł

Działanie write-verbose i write-debug podczas debugguwania skryptów