PowerShell

проф. д-р инж. Христо Вълчанов

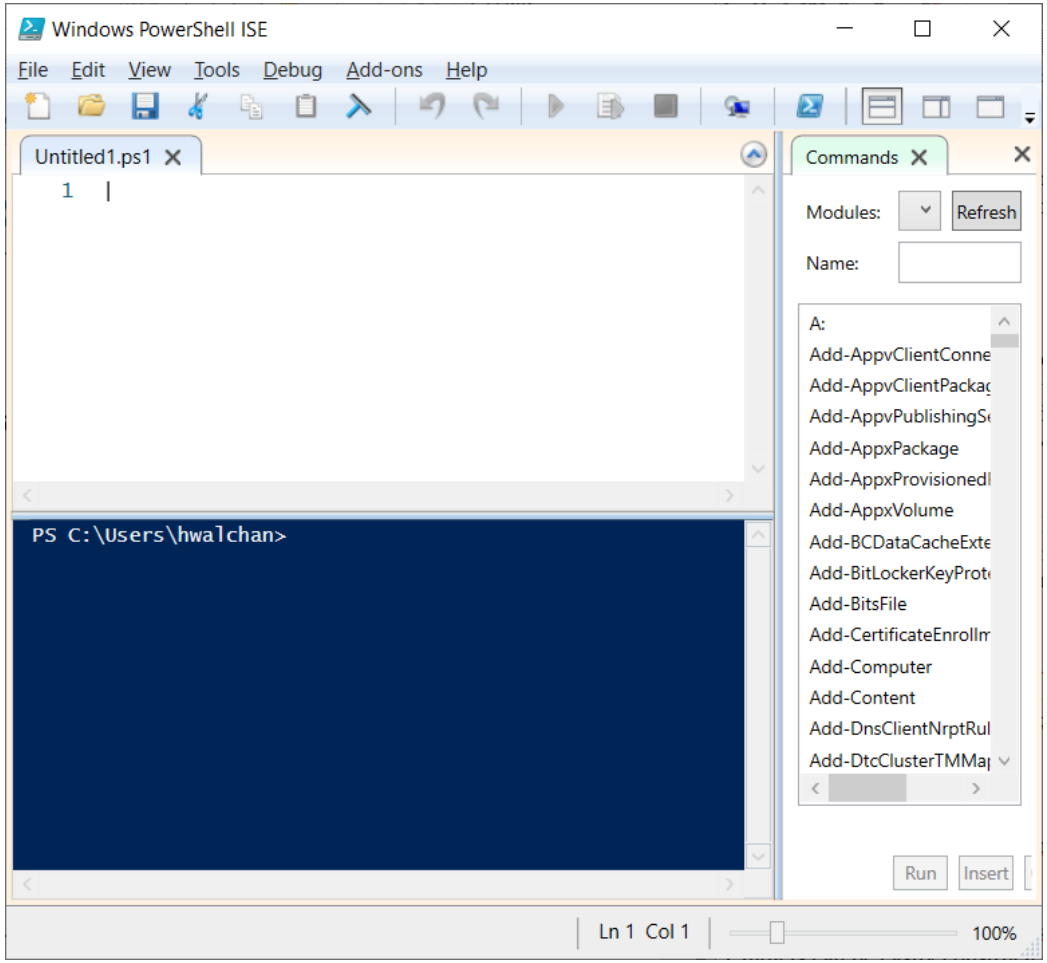
http://cs.tu-varna.bg

Windows PowerShell

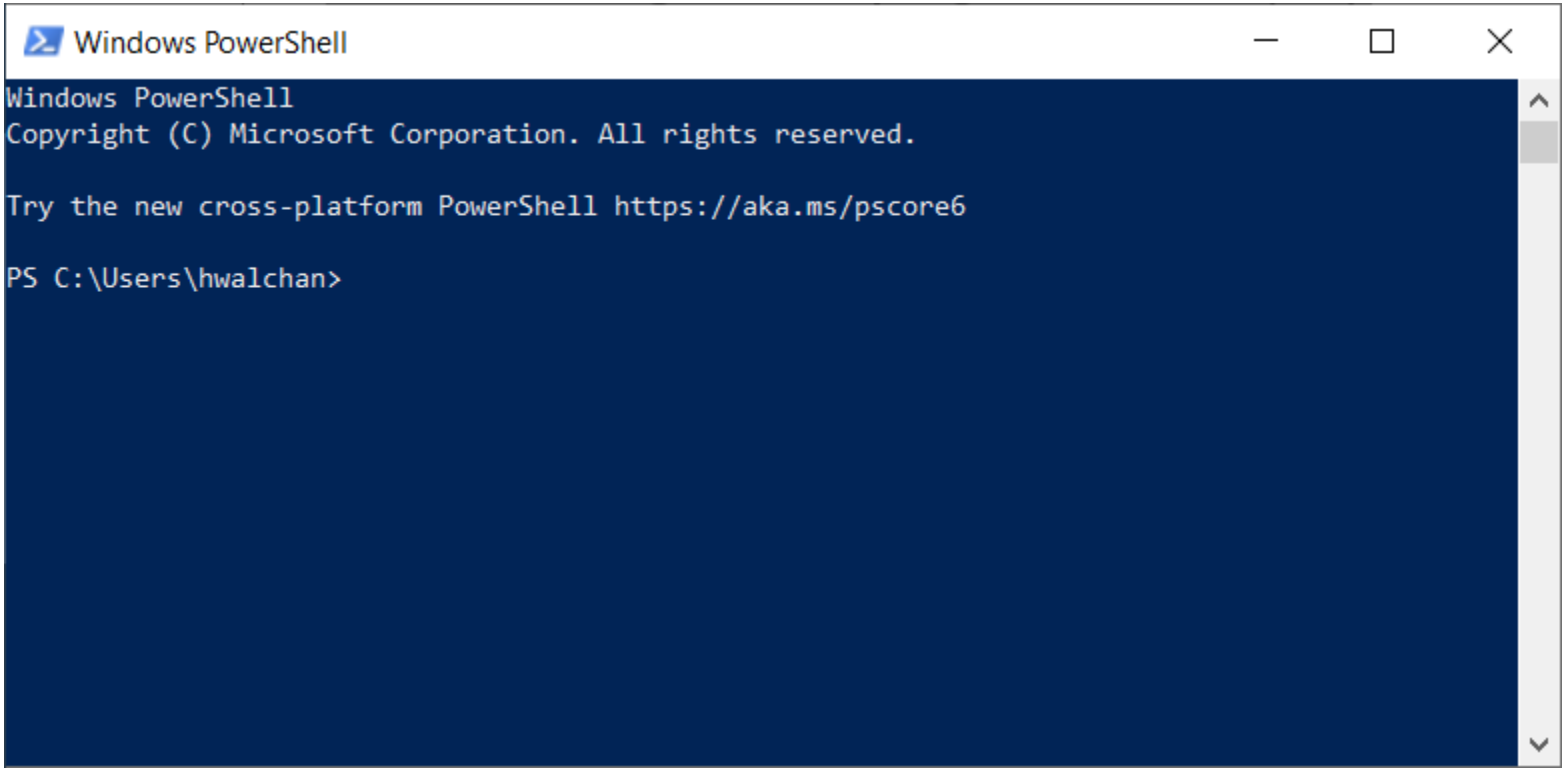
- Команден shell и скриптов език.
- Проектиран е за системно администриране.
- Изграден е на базата на .NET Framework.

Интегрирана среда

- **PowerShell Integrated Scripting Environment (ISE)**



Работа от конзола



Версия на PowerShell

$PSversionTable

Administrator: Windows PowerShell

```
PS C:\> $PSVersionTable

Name                           Value
----                           -----
PSVersion                      5.1.19041.610
PSEdition                      Desktop
PSCompatibleVersions           {1.0, 2.0, 3.0, 4.0...}
BuildVersion                   10.0.19041.610
CLRVersion                     4.0.30319.42000
WSManStackVersion              3.0
PSRemotingProtocolVersion      2.3
SerializationVersion           1.1.0.1


PS C:\>
```

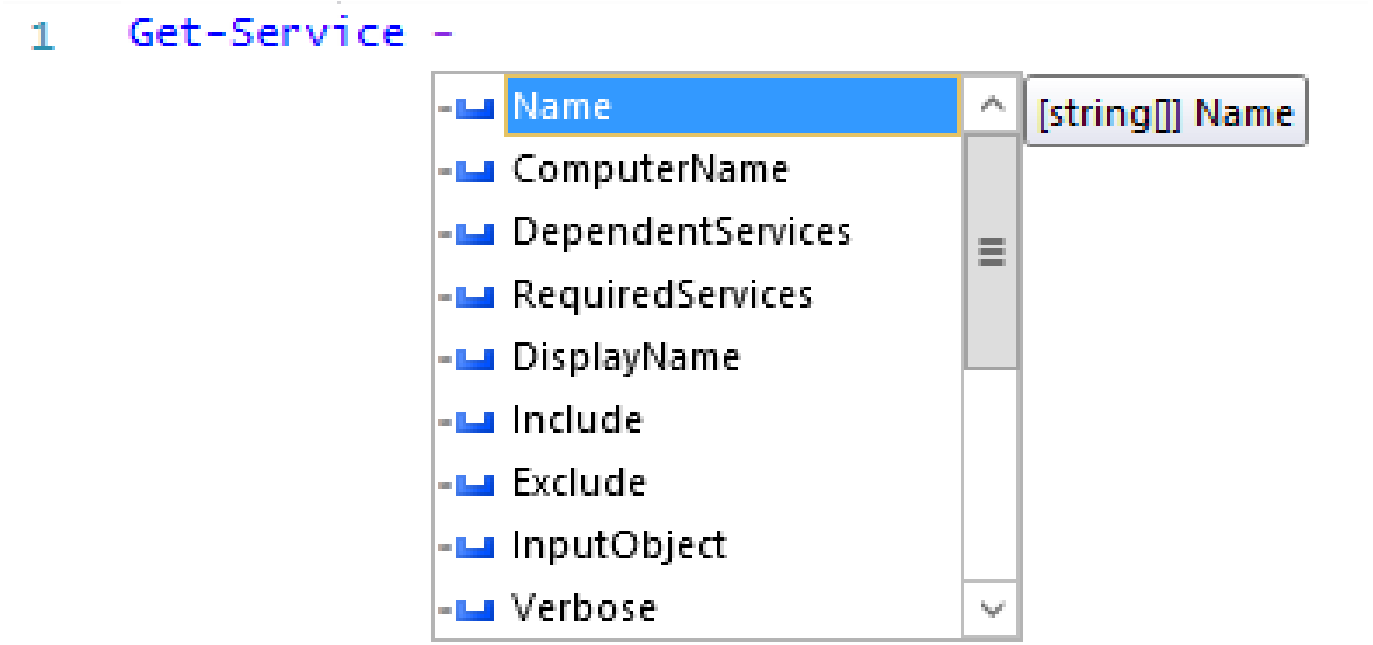
Cmdlet и Command

Командлетите (**cmdlets**) се различават от командите:

- Командлетите са класове обекти от .NET Framework, не са самостоятелно изпълними файлове.
- Могат лесно да се създадат от няколко реда код.
- Парсването, извеждането на грешки и форматирането на изхода се извършва не от командлетите, а от Windows PowerShell изпълнителната среда.
- Командлетите работят с обекти. Обекти могат да се изпратят като изход за използване с канали (pipes).
- Командлетите обработват единичен обект в даден момент.

Параметри на командлетите

Всеки командлет има множество параметри. PowerShell ISE автоматично предлага възможните параметри:



Политика на изпълнение

- По подразбиране Windows не позволява директно стартиране на скриптове (двойно кликване).
- Лошо направени скриптове или изтеглени от Интернет, които са зловредни могат да повредят цялата система.
- Необходимо е явно указване да се стартират с Windows PowerShell (*Run with PowerShell*).

Политика на изпълнение

- Извеждане на текущата политика:

```
Get-ExecutionPolicy
```

Възможни политики:

Restricted - не се разрешава изпълнение на скриптове.

AllSigned - може да се изпълняват скриптове само от доверени разработчици

RemoteSigned – могат да се стартират собствени скриптове и такива от доверени разработчици

Unrestricted - могат да се изпълняват всякакви скриптове. Не се препоръчва.

Политика на изпълнение

- Установяване на текуща политика:

```
Set-ExecutionPolicy политика
```

Най-често се използва **Remote-Signed.**

Синтаксис на командлет

`VERB-NOUN параметри`

VERB – действие (Get, Set, Add, Write, Read …)

NOUN – предмет на действието (файлове, сървъри, обекти …)

Командлетите връщат като резултат обекти, които могат да се използват в последствие.

Получаване на атрибутите и методите на обект:

```
Get-Location | Get-Member
```

Помощна информация

Get-Command

Administrator: Windows PowerShell

```
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

Try the new cross-platform PowerShell https://aka.ms/pscore6

PS C:\WINDOWS\system32> get-command

CommandType     Name                                               Version    Source
-----------     ----                                               -------    ------
Alias           Add-AppPackage                                     2.0.1.0    Appx
Alias           Add-AppPackageVolume                               2.0.1.0    Appx
Alias           Add-AppProvisionedPackage                          3.0        Dism
Alias           Add-ProvisionedAppPackage                          3.0        Dism
Alias           Add-ProvisionedAppxPackage                         3.0        Dism
Alias           Add-ProvisioningPackage                            3.0        Provisioning
Alias           Add-TrustedProvisioningCertificate                 3.0        Provisioning
Alias           Apply-WindowsUnattend                              3.0        Dism
Alias           Disable-PhysicalDiskIndication                     2.0.0.0    Storage
Alias           Disable-StorageDiagnosticLog                       2.0.0.0    Storage
```

Помощна информация

Get-Command команда

Administrator: Windows PowerShell

```
PS C:\WINDOWS\system32> get-command Enable-NetAdapter

CommandType     Name                                               Version    Source
-----------     ----                                               -------    ------
Function        Enable-NetAdapter                                  2.0.0.0    NetAdapter


PS C:\WINDOWS\system32>
```

Подробна помощна информация

Get-Help –Name команда

Administrator: Windows PowerShell

```
PS C:\WINDOWS\system32> Get-Help -name Enable-NetAdapter

NAME
    Enable-NetAdapter

SYNOPSIS
    Enables a network adapter.


SYNTAX
    Enable-NetAdapter [-Name] <String[]> [-AsJob] [-CimSession <CimSession[]>] [-IncludeHidde
    n] [-PassThru] [-ThrottleLimit <Int32>] [-Confirm] [-WhatIf] [<CommonParameters>]

    Enable-NetAdapter [-AsJob] [-CimSession <CimSession[]>] [-PassThru] [-ThrottleLimit <Int3
    2>] -InputObject <CimInstance[]> [-Confirm] [-WhatIf] [<CommonParameters>]

    Enable-NetAdapter [-AsJob] [-CimSession <CimSession[]>] [-IncludeHidden] [-PassThru] [-Th
    rottleLimit <Int32>] -InterfaceDescription <String[]> [-Confirm] [-WhatIf] [<CommonParame
    ters>]


DESCRIPTION
    The Enable-NetAdapter cmdlet enables a network adapter. A network adapter must be enabled
     to connect to a network.


RELATED LINKS
    Online Version:
    Disable-NetAdapter

REMARKS
    To see the examples, type: "get-help Enable-NetAdapter -examples".
    For more information, type: "get-help Enable-NetAdapter -detailed".
    For technical information, type: "get-help Enable-NetAdapter -full".
    For online help, type: "get-help Enable-NetAdapter -online"
```

Атрибути и методи на обекти

```
Get-Service -Name w32time
```

```
PS C:\WINDOWS\system32> C:\Users\hwalchan\Documents\Labs\Virtualization\LabsRemot

Status   Name               DisplayName
------   ----               -----------
Stopped  w32time            Windows Time

PS C:\WINDOWS\system32>
```

Атрибути (properties) – Status, Name, DisplayName

Атрибути и методи на обекти

Get-member

```
Get-Service -Name w32time | Get-member
```

```
PS C:\WINDOWS\system32> C:\Users\hwalchan\Documents\Labs\Virtualization\LabsRemot

   TypeName: System.ServiceProcess.ServiceController

Name                      MemberType     Definition
----                      ----------     ----------
Name                      AliasProperty  Name = ServiceName
RequiredServices          AliasProperty  RequiredServices = ServicesDependedOn
Disposed                  Event          System.EventHandler Disposed(System....
Close                     Method         void Close()
Continue                  Method         void Continue()
CreateObjRef              Method         System.Runtime.Remoting.ObjRef Creat...
Dispose                   Method         void Dispose(), void IDisposable.Dis...
Equals                    Method         bool Equals(System.Object obj)
ExecuteCommand            Method         void ExecuteCommand(int command)
GetHashCode               Method         int GetHashCode()
GetLifetimeService        Method         System.Object GetLifetimeService()
GetType                   Method         type GetType()
InitializeLifetimeService Method         System.Object InitializeLifetimeServ...
Pause                     Method         void Pause()
```

Атрибути и методи на обекти

Получаване на атрибутите на обект

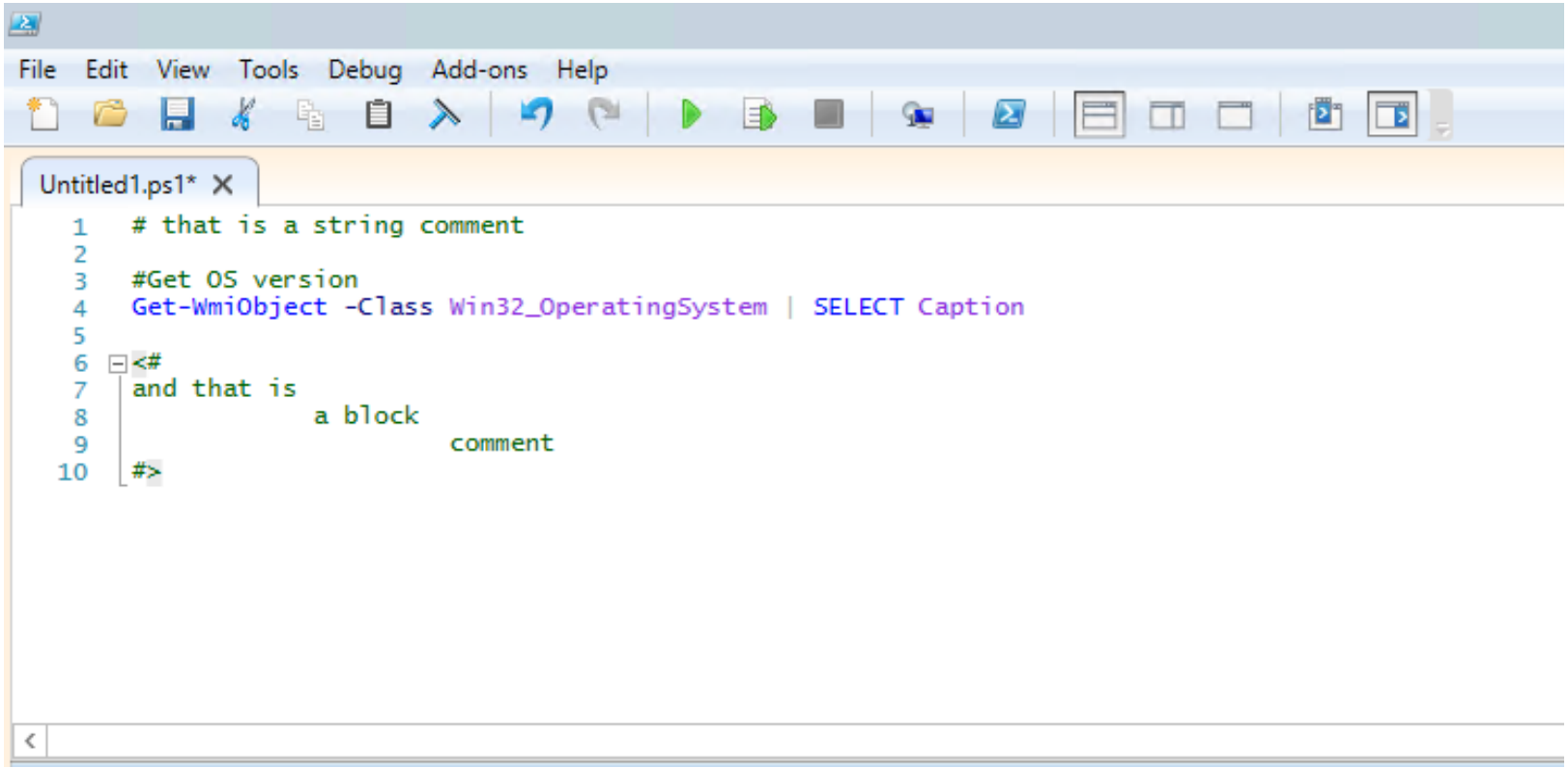
```
Get-Service -Name w32time | Select-Object -Property *
```

```
PS C:\WINDOWS\system32> C:\Users\hwalchan\Documents\Labs\Virtualization\LabsRemot


Name                  : w32time
RequiredServices      : {}
CanPauseAndContinue   : False
CanShutdown           : False
CanStop               : False
DisplayName           : Windows Time
DependentServices     : {}
MachineName           : .
ServiceName           : w32time
ServicesDependedOn    : {}
ServiceHandle         : SafeServiceHandle
Status                : Stopped
ServiceType           : Win32OwnProcess, Win32ShareProcess
StartType             : Manual
Site                  :
Container             :
```

Коментари

- Низов коментар
- Блоков коментар



Променливи

Променливите са именувани обекти.

Създаване на променлива:

```
$location = Get-Location
```

Използване на променлива:

```
$location
```

Примерен изход:

```
Path
------
D:\test
```

Аритметични оператори

$A=10

$B=20

Operator	Example
+ (Addition)	A + B will give 30
- (Subtraction)	A - B will give -10
*** (Multiplication)**	A * B will give 200
/ (Division)	B / A will give 2
% (Modulus)	B % A will give 0

Оператори за сравнение

$A=10

$B=20

Operator	Example
eq (equals)	A -eq B will give false
ne (not equals)	A -ne B will give true
gt (greater than)	B -gt A will give true
ge (greater than or equals to)	B -ge A will give true
lt (less than)	B -lt A will give false
le (less than or equals to)	B -le A will give false

Оператори за присвояване

Operator	Example
=	C = A + B will assign value of A + B into C
+=	C += A is equivalent to C = C + A
-=	C -= A is equivalent to C = C - A

Логически оператори

$A=$TRUE

$B=$FALSE

Operator	Description	Example
AND (logical and)	Ако и двата операнда са не-нулеви, резултатът е TRUE	(A -AND B) is false
OR (logical or)	Ако някои от операндите е не-нулев, резултатът е TRUE	(A -OR B) is true
NOT (logical not)	Отрицание	

Оператори за цикъл

for

```
> $array = @("item1", "item2", "item3")
> for($i = 0; $i -lt $array.length; $i++){ $array[$i] }

item1
item2
item3
```

Оператори за цикъл

forEach

```
> $array = @("item1", "item2", "item3")

> foreach ($element in $array) { $element }
item1
item2
item3

> $array | foreach { $_ }
item1
item2
item3
```

Оператори за цикъл

while

```
> $array = @("item1", "item2", "item3")
$counter = 0;

while($counter -lt $array.length){
   $array[$counter]
   $counter += 1
}

item1
item2
item3
```

Оператори за цикъл

do..while

```
> $array = @("item1", "item2", "item3")
$counter = 0

do {
   $array[$counter]
   $counter += 1
} while($counter -lt $array.length)

item1
item2
item3
```

Условни оператори

if

```
if(Boolean_expression) {
   // Executes when the Boolean expression is true
}else {
   // Executes when the Boolean expression is false
}
```

Условни оператори

вложен if

```
if(Boolean_expression 1) {
   // Executes when the Boolean expression 1 is true
   if(Boolean_expression 2) {
      // Executes when the Boolean expression 2 is true
   }
}

if(Boolean_expression 1) {
   // Executes when the Boolean expression 1 is true
   elseif(Boolean_expression 2) {
      // Executes when the Boolean expression 2 is true
   }
}
```

Условни оператори

switch

```
switch(<test-expression>) {
   < result-to-be-matched > { <action> ; break }
        // ; break - optional
   < result-to-be-matched > { <action> ; break }
        // ; break - optional
  default { <action> } // optional
}
```

Масиви

Създаване на масив

$име_масив = с-ст1, с-ст2, …. , с-стN

```
$arr = 10, 15, 20, 11, 5
```

$име_масив = @(оператор1, оператор2, ... , операторN)

```
$arr = @("item1", "item2", "item3")
```

Достъп до елемент на масив

```
$arr[3]
```

Масиви и методи

Administrator: Windows PowerShell

```
PS C:\WINDOWS\system32> $arr | Get-Member


   TypeName: System.Int32

Name        MemberType Definition
----        ---------- ----------
CompareTo   Method     int CompareTo(System.Object value), int CompareTo(int value), int I...
Equals      Method     bool Equals(System.Object obj), bool Equals(int obj), bool IEquatab...
GetHashCode Method     int GetHashCode()
GetType     Method     type GetType()
GetTypeCode Method     System.TypeCode GetTypeCode(), System.TypeCode IConvertible.GetType...
ToBoolean   Method     bool IConvertible.ToBoolean(System.IFormatProvider provider)
ToByte      Method     byte IConvertible.ToByte(System.IFormatProvider provider)
ToChar      Method     char IConvertible.ToChar(System.IFormatProvider provider)
ToDateTime  Method     datetime IConvertible.ToDateTime(System.IFormatProvider provider)
ToDecimal   Method     decimal IConvertible.ToDecimal(System.IFormatProvider provider)
ToDouble    Method     double IConvertible.ToDouble(System.IFormatProvider provider)
ToInt16     Method     int16 IConvertible.ToInt16(System.IFormatProvider provider)
ToInt32     Method     int IConvertible.ToInt32(System.IFormatProvider provider)
ToInt64     Method     long IConvertible.ToInt64(System.IFormatProvider provider)
ToSByte     Method     sbyte IConvertible.ToSByte(System.IFormatProvider provider)
ToSingle    Method     float IConvertible.ToSingle(System.IFormatProvider provider)
ToString    Method     string ToString(), string ToString(string format), string ToString(...
ToType      Method     System.Object IConvertible.ToType(type conversionType, System.IForm...
ToUInt16    Method     uint16 IConvertible.ToUInt16(System.IFormatProvider provider)
ToUInt32    Method     uint32 IConvertible.ToUInt32(System.IFormatProvider provider)
ToUInt64    Method     uint64 IConvertible.ToUInt64(System.IFormatProvider provider)


PS C:\WINDOWS\system32>
```

Пренасяне на нов ред

Ако командата трябва да се въведе на няколко реда, се използва обратен апостроф:

```
Get-Service * | Sort-Object ServiceType `
| Format-Table Name, ServiceType, Status -AutoSize
```

Въвеждане на данни в скрипт

Въвеждане на текст от клавиатурата:

```
$Answer = Read-Host "Please enter Y/N„
```

Въвеждане на текст и маскиране на информацията (излизат *):

```
$pwd_string = Read-Host "Enter a Password" `
                -AsSecureString
```

Изпълнение на PowerShell скрипт

Скриптовете се съхраняват с разширение .ps1

Стартиране от PowerShell:

```
C:\Scripts\MyScript.ps1
```

Стартиране от ISE:

с бутон „Play” или от меню “Debug”

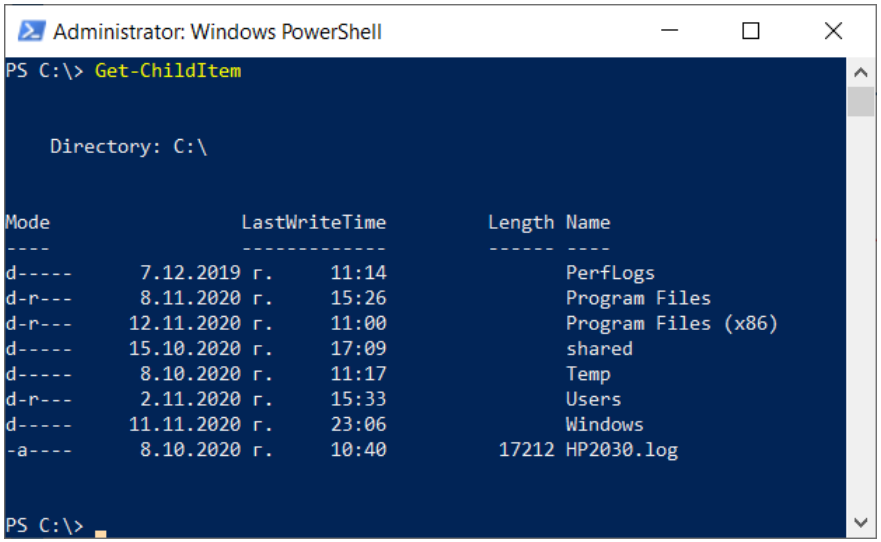
Стартиране от команден промпт на Windows:

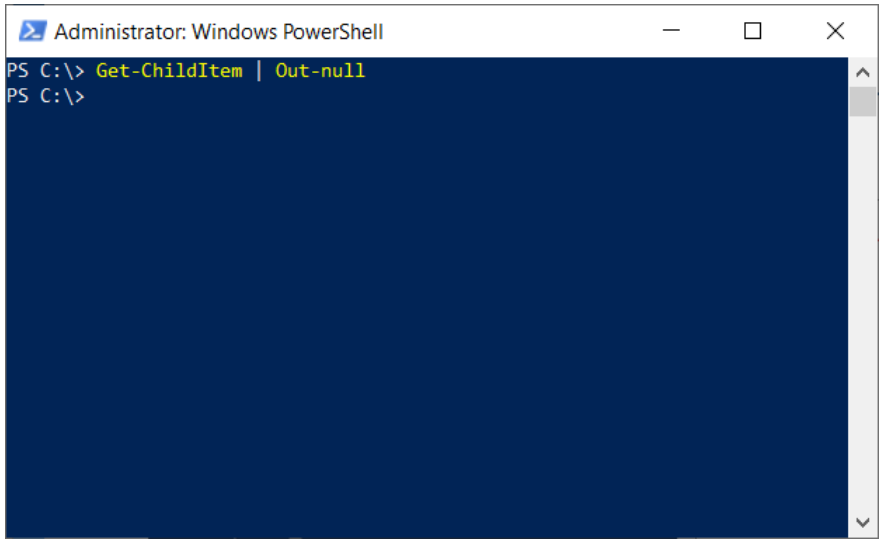
```
C:\powershell -F C:\Scripts\MyScript.ps1
```

Примери

Игнориране на изхода от команди – Out-null

```
Get-ChildItem | Out-Null
```





Примери

Използване на канали и сортиране - |

```
Get-Service | Sort-Object -property Status
```

Administrator: Windows PowerShell

```
PS C:\> Get-Service | Sort-Object -property Status

Status   Name               DisplayName
------   ----               -----------
Stopped  RasAuto            Remote Access Auto Connection Manager
Stopped  QWAVE              Quality Windows Audio Video Experience
Stopped  PushToInstall      Windows PushToInstall Service
Stopped  RemoteAccess       Routing and Remote Access
Stopped  RpcLocator         Remote Procedure Call (RPC) Locator
Stopped  RetailDemo         Retail Demo Service
Stopped  RemoteRegistry     Remote Registry
Stopped  PNRPsvc            Peer Name Resolution Protocol
Stopped  PNRPAutoReg        PNRP Machine Name Publication Service
Stopped  pla                Performance Logs & Alerts
Stopped  PolicyAgent        IPsec Policy Agent
Stopped  PrintWorkflowUs... PrintWorkflow_dae827
Stopped  PrintWorkflowUs... PrintWorkflow_d18709
Stopped  PrintNotify        Printer Extensions and Notifications
Stopped  ScDeviceEnum       Smart Card Device Enumeration Service
Stopped  SharedAccess       Internet Connection Sharing (ICS)
```

Примери

Пренасочване на изход във файл

```
"Hello, World!" | Out-File C:\ps\test.txt
```

Използване на множество канали и филтриране

```
Get-Service | WHERE {$_.status -eq "Running"} | SELECT displayname
```

$_ е специална променлива на PowerShell, указваща текущия обект (в примера, който е в канала)

Новият вариант от версия 3.0 е:

```
Get-Service | Where-Object status -eq Running |
        SELECT  displayname
```

Примери

Извеждане на информация от PowerShell скрипт – Write-Host:

```
Get-date | Write-Host
```

```
PS C:\WINDOWS\system32> C:\Users\hwalchan\Documents\Labs\Virtualization\LabsRemote\scr1.ps1
14.11.2020 г. 16:01:06

PS C:\WINDOWS\system32>
```

Примери

Създаване на текстов файл, записване в него и четене:

```
New-Item C:\temp\test.txt
Set-Content C:\temp\test.txt „This is a test.“
$file = Get-Content C:\temp\test.txt
Write-Host $file
```

```
PS C:\WINDOWS\system32> C:\Users\hwalchan\Documents\Labs\Virtualization\LabsRemot

    Directory: C:\temp

Mode                LastWriteTime         Length Name
----                -------------         ------ ----
-a----     14.11.2020 г.     16:06              0 test.txt
This is a test.


PS C:\WINDOWS\system32> |
```

Примери

Извеждане на първите 10 процеса по използване на CPU:

```
$res = Get-Process | Sort-Object CPU -descending | `
                Select-Object -first 10
$res
```

```
PS C:\WINDOWS\system32> C:\Users\hwalchan\Documents\Labs\Virtualization\LabsRemot

Handles  NPM(K)    PM(K)      WS(K)     CPU(s)     Id  SI ProcessName
-------  ------    -----      -----     ------     --  -- -----------
   1591      58   790720     758088   3 696,72  11612   4 dwm
   4814       0      236      20048   2 981,64      4   0 System
   1044      86   338156     260268     788,31   5052   0 MsMpEng
    443      14     4004      14696     774,31   3660   0 svchost
   1979      75   205440     251640     591,59   2692   4 POWERPNT
   3138     175    97584     177252     521,81   2704   4 explorer
   1366      72   190884     248196     309,75  16216   4 WINWORD
    682      25     2656       6764     302,45   8088   4 csrss
   2304     166   264332     394792     229,94   3572   4 firefox
    820      61   329312     338948     185,95   1236   4 firefox
```

Примери

Стартиране на Notepad, изчакване на 10 сек и затваряне:

```
Start-Process notepad
Start-Sleep 10
Stop-Process -processname notepad
```

Примери

Проверка за съществуване на файл/директория:

```
if (Test-Path "C:\Temp") {
    Write-Host "C:\temp exists"
} else {
    Write-Host "Not exists"
}
```

Примери

Получаване на IP адреса на активния интерфейс:

```
$res=Get-WmiObject -Class
Win32_NetworkAdapterConfiguration `
    -Filter IPEnabled=$true | SELECT IPAddress
$res
```

```
PS C:\WINDOWS\system32> C:\Users\hwalchan\Documents\Labs\Virtualization\LabsRemot

IPAddress
---------
{192.168.0.101, fe80::bda8:7226:e847:de87}


PS C:\WINDOWS\system32>
```

Въпроси?