***Лабораторна робота №11***

***«Форматування результату виведення команд у Windows PowerShell. Збереження даних у файл»***

Розглянемо, яким чином система формує рядки тексту, які виводяться на екран в результаті виконання тієї чи іншої команди (нагадаємо, що командлети **PowerShell** повертають .NET-об'єкти, які, як правило, не знають, яким чином відображати себе на екрані).

У **PowerShell** є база даних (набір XML-файлів), що містить модулі форматування за замовчуванням для різних типів .NET-об'єктів. Ці модулі визначають, які властивості об'єкта відображаються при виведенні і в якому форматі: списку або таблиці. Коли об'єкт досягає кінця конвеєра, **PowerShell** визначає його тип і шукає його в списку об'єктів, для яких визначено правило форматування. Якщо даний тип в списку виявлений, то до об'єкта застосовується відповідний модуль форматування; якщо немає, то **PowerShell** просто відображає властивості цього .NET-об'єкта.

Також в **PowerShell** можна явно задавати правила форматування даних, що виводяться командлетами, і подібно командному інтерпретатору **Cmd.exe** перенаправляти ці дані в файл, на принтер або в порожній пристрій.

***Форматування інформації, що виводиться***

У традиційних оболонках команди і утиліти самі форматують виведені дані. Деякі команди (наприклад, ***dir*** в інтерпретаторі **Cmd.exe**) дозволяють налаштовувати формат виведення за допомогою спеціальних параметрів.

В оболонці **PowerShell** висновок форматують тільки чотири спеціальні командлети ***Format*** (табл. 1). Це спрощує вивчення, так як не потрібно запам'ятовувати засоби і параметри форматування для інших команд (інші командлети виведення не форматують).

Табл. 1. Командлети **PowerShell** для форматування виведення

|  |  |
| --- | --- |
| **Командлет** | **Опис** |
| ***Format-Table*** | Форматує виведення команди у вигляді таблиці, стовпці якої містять властивості об'єкта (також можуть бути додані обчислювані стовпці). Підтримується можливість угруповання даних, що виводяться |
| ***Format-List*** | Виведення форматується як список властивостей, в якому кожна властивість відображається на новому рядку. Підтримується можливість угруповання даних, що виводяться |
| ***Format-Custom*** | Для форматування виведення використовується представлення (view) |
| ***Format-Wide*** | Форматує об'єкти у вигляді широкої таблиці, в якій відображається тільки одна властивість кожного об'єкта |

Як уже зазначалося, якщо жоден з командлетів ***Format*** явно не вказаний, то використовується модуль форматування за замовчуванням, який визначається за типом даних, що відображаються. Наприклад, при виконанні командлету ***Get-Service*** дані за замовчуванням виводяться як таблиця з трьома стовпцями (***Status***, ***Name*** і ***DisplayName***):

***PS C:\> Get-Service***

***Status Name DisplayName***

***------ ---- -----------***

***Stopped Alerter Оповещатель***

***Running ALG Служба шлюза уровня приложения***

***Stopped AppMgmt Управление приложениями***

***Stopped aspnet\_state ASP.NET State Service***

***Running Ati HotKey Poller Ati HotKey Poller***

***Running AudioSrv Windows Audio***

***Running BITS Фоновая интеллектуальная служба пер...***

***Running Browser Обозреватель компьютеров***

***Stopped cisvc Служба индексирования***

***Stopped ClipSrv Сервер папки обмена***

***Stopped clr\_optimizatio... .NET Runtime Optimization Service v...***

***Stopped COMSysApp Системное приложение COM+***

***Running CryptSvc Службы криптографии***

***Running DcomLaunch Запуск серверных процессов DCOM***

***Running Dhcp DHCP-клиент***

...

Для зміни формату виведених даних потрібно направити їх по конвеєру відповідному командлету ***Format***. Наприклад, наступна команда виведе список служб за допомогою командлета ***Format-List***:

***PS C:\> Get-Service | Format-List***

***Name : Alerter***

***DisplayName : Оповещатель***

***Status : Stopped***

***DependentServices : {}***

***ServicesDependedOn : {LanmanWorkstation}***

***CanPauseAndContinue : False***

***CanShutdown : False***

***CanStop : False***

***ServiceType : Win32ShareProcess***

***Name : ALG***

***DisplayName : Служба шлюза уровня приложения***

***Status : Running***

***DependentServices : {}***

***ServicesDependedOn : {}***

***CanPauseAndContinue : False***

***CanShutdown : False***

***CanStop : True***

***ServiceType : Win32OwnProcess***

***…***

При використанні формату списку виводиться більше відомостей про кожну службу, ніж в форматі таблиці (замість трьох стовпців даних про кожну службі в форматі списку виводяться дев'ять рядків даних). Однак це зовсім не означає, що командлет ***Format-List*** витягує додаткові відомості про служби. Ці дані містяться в об'єктах, які повернуться командлетом ***Get-Service***, проте командлет ***Format-Table***, який буде використовуватися за замовчуванням, відкидає їх, тому що не може вивести на екран більше трьох стовпців.

При форматуванні виведення за допомогою командлетів ***Format-List*** і ***Format-Table*** можна вказувати імен властивості об'єкта, які повинні бути відображено (нагадаємо, що переглянути список властивостей, наявних у об'єкта, дозволяє розглянутий раніше командлет ***Get-Member***). Наприклад:

***PS C: \> Get-Service | Format-List Name, Status, CanStop***

***Name: Alerter***

***Status: Stopped***

***CanStop: False***

***Name: ALG***

***Status: Running***

***CanStop: True***

***Name: AppMgmt***

***Status: Stopped***

***CanStop: False***

***. . .***

Вивести всі наявні у об'єктів властивості можна за допомогою параметра ***\****, наприклад:

***PS C:\> Get-Service | Format-table \****

***Перенаправлення виведеної інформації***

В оболонці **PowerShell** є кілька командлетів, за допомогою яких можна управляти виведенням даних. Ці командлети починаються зі слова ***Out***, їх список можна отримати за допомогою командлета:

***PS C: \> Get-Command out-\* | Format-Table Name***

***Name***

***----***

***Out-Default***

***Out-File***

***Out-Host***

***Out-Null***

***Out-Printer***

***Out-String***

За замовчуванням виводиться інформація передається командлету ***Out-Default***, який, в свою чергу, делегує всю роботу з виведення рядків на екран командлету ***Out-Host***. Для розуміння даного механізму потрібно враховувати, що архітектура **PowerShell** має на увазі відмінність між власне ядром оболонки (інтерпретатором команд) і головним додатком (**host**), який використовує це ядро. В принципі, в якості головного може виступати будь-який додаток, в якому реалізований ряд спеціальних інтерфейсів, що дозволяють коректно інтерпретувати отримувану від **PowerShell** інформацію. У нашому випадку головним додатком є ​​консольне вікно, в якому ми працюємо з оболонкою, і командлет **Out-Host** передає інформацію, що виводиться в це консольне вікно.

Параметр **Paging** командлета ***Out-Host***, подібно команді ***more*** інтерпретатора **Cmd.exe**, дозволяє організувати посторінкове виведення інформації, наприклад:

***Get-Help Get-Process | Out-Host –Paging***

***Збереження даних у файл***

Командлет ***Out-File*** дозволяє направити виведені дані замість вікна консолі в текстовий файл. Аналогічне завдання вирішує оператор перенаправлення (***>***), проте командлет ***Out-File*** має кілька додаткових параметрів, за допомогою яких можна більш гнучко управляти виведенням: задавати тип кодування файлу (параметр ***-Encoding***), задавати довжину виведених рядків в знаках (параметр ***-Width***), вибирати режим додавання до файлу (параметр ***-Append***). Додати дані у кінець файлу можливо і традиційним способом з використанням оператору перенаправлення (***>>***).

Наприклад, наступна команда направить інформацію про виконуються на комп'ютері процесах в файл ***C:\Process.txt***, причому даний файл буде записаний в форматі ASCII:

***Get-Process | Out-File -FilePath C:\Process.txt -Encoding ASCII***

***Придушення виведення***

Командлет ***Out-Null*** служить для поглинання будь-яких своїх вхідних даних. Це може стати в нагоді для придушення виведення на екран непотрібних відомостей, отриманих в якості побічного ефекту виконання будь-якої команди. Наприклад, при створенні каталогу командою ***mkdir*** на екран виводиться його вміст:

***PS C:\> mkdir spo***

***Каталог: Microsoft.PowerShell.Core\FileSystem::C:\***

***Mode LastWriteTime Length Name***

***---- ------------- ------ ----***

***d---- 03.01.2009 1:01 spo***

Якщо ця інформація не потрібна, то результат виконання команди ***mkdir*** необхідно передати по конвеєру командлету ***Out-Null***:

***mkdir spo | Out-Null***

***Завдання***

1. Вивести інформацію про процеси, що мають Id, який більше за 7000, у табличному вигляді, залишивши при цьому тільки Id процесу, його ім’я та кількість процесорного часу, що зайняв процес. Результат вивести у відсортованому вигляді за спаданням.
2. Вивести інформацію про файли, що знаходяться у кореневому каталозі Вашого мережевого диску у вигляді списку, залишивши при цьому наступні властивості: ім’я, розмір, дата останньої зміни. Результат вивести у відсортованому вигляді за розширенням файлу.
3. Вивести інформацію про підкаталоги, що знаходяться у кореневому каталозі Вашого мережевого диску у вигляді списку, залишивши при цьому наступні властивості: ім’я, дата створення, дата та час останнього звертання до каталогу. Результат вивести у відсортованому вигляді за датою створення.
4. Змінити завдання 3 наступним чином: залишити тільки підкаталог, який був створений раніше за всі інші, при цьому вивести інформацію про нього у найбільш детальному вигляді (за допомогою командлету ***Format-Custom***). Що було отримано у результати? Чи є зручним таке виведення?
5. Змінити завдання 4 так, щоб інформація, що виводиться командлетом Format-Custom виводилась з можливістю її посторінкового перегляду.
6. Змінити завдання 4 так, щоб виводилась детальна інформація тільки на першому рівні вкладеності об’єктів (за допомогою параметра ***-Depth*** зі значенням ***1***).
7. Вивести інформацію про зупинені служби у системі у файли ***stopped1.txt*** та ***stopped2.txt*** двома різними способами.
8. Додати до існуючих файлів ***stopped1.txt*** та ***stopped2.txt*** двома різними способами інформацію про працюючі служби у системі.
9. Створити у кореневому каталозі Вашого мережевого диску новий файл file1.txt, не виводячі результат.
10. Видалити файл file1.txt, не виводя

***Контрольні питання***

1. Які командлети призначені для форматування виведеної інформації? У чому полягає їх різниця один від одного?
2. Які командлети призначені для перенаправлення інформації? У які пристрої можна перенапрявляти інформацію? Наведіть приклади.
3. Куди перенапрявляє інформацію командлет ***Out-Host***?
4. Який командлет дозволяє записати виведену інформацію у файл?
5. Для чого призначений командлет ***Out-Null***?