**PowerShell**

Windows PowerShell – это язык сценариев и командная оболочка Windows, которые разработаны для администрирования и конфигурирования операционных систем Windows.

PowerShell разработан на основе среды CRL и платформы .NET Framework и в отличие от командной строки, которая принимает и возвращает текст, Windows PowerShell работает с объектами. У каждого объекта в PowerShell есть свойства и методы, которые можно использовать для управления этими объектами.

В Windows PowerShell Microsoft разработала концепцию командлетов (Командлет (cmdlet) – это команда Windows PowerShell, с помощью которой можно осуществлять взаимодействие с объектами операционной системы с целью их управления.), которая представляет собой систему именования команд «Глагол-Существительное». Данная система позволяет системным администраторам быстрей освоить и упростить работу с PowerShell.

***С помощью Windows PowerShell можно:***

* Получать доступ к файловой системе;
* Управлять реестром;
* Управлять службами;
* Управлять процессами;
* Настраивать операционную систему;
* Устанавливать программное обеспечение;
* Устанавливать роли и компоненты сервера;
* Осуществлять администрирование и конфигурирование ролей и компонентов сервера;
* Писать и использовать сценарии для автоматизации управления и администрирования;
* Выполнять другие задачи системных администраторов.

Для упрощения поиска нужной команды в PowerShell есть специальный командлет Get-Command, с помощью которого можно осуществлять поиск, как по глаголу, так и по существительному. Все команды в Windows PowerShell сгруппированы в модули (например, Hyper-V, NetTCPIP), что также упрощает поиск нужной команды.

После того как нужная команда найдена, можно посмотреть инструкцию по работе с этой командой, т.е. справку, для этих целей есть специальный командлет Get-Help, например следующая команда покажет справку по командлету Get-Command:



Кроме командлетов на получение данных (Get), существуют и такие типы командлетов как:

* Add – добавление данных;
* Clear – очистить;
* Enable – включить;
* Disable – выключить;
* New – создать;
* Remove – удалить;
* Set – задать;
* Start — запустить;
* Stop – остановить;
* Export – экспортировать;
* Import – импортировать;
* И еще много других.

Сценарии, функции и модули в Windows PowerShell

Если Вы написали некий алгоритм, скорей всего Вы захотите сохранить его, для того чтобы в дальнейшем им пользоваться. В Windows PowerShell для этих целей существует механизм использования сценариев.

Суть данного механизма следующая, Вы пишите нужный Вам набор команд и сохраняете его в качестве сценария — это простой текстовый файл с расширением .PS1.

Затем, для того чтобы выполнить этот сценарий, Вы просто запускаете его в оболочке PowerShell. При этом Вам необходимо либо указать полный путь к файлу сценария, либо перейти в каталог со сценарием и вызвать его по имени.

Важно!

По умолчанию выполнение сценариев в Windows запрещено! Для того чтобы посмотреть политику выполнения сценариев выполните командлет Get-ExecutionPolicy. В результате он вернет действующую политику, например:

* Restricted – блокируется выполнение любых сценариев (значение по умолчанию);
* AllSigned – разрешено выполнение сценариев, которые имеют цифровую подпись;
* RemoteSigned – разрешено выполнение локальных сценариев, все скачанные сценарии должны иметь цифровую подпись;
* Unrestricted — разрешено выполнение любых сценариев (не рекомендуется, так как небезопасно!).

Для разрешения выполнения сценариев необходимо использовать командлет Set-ExecutionPolicy с одним из вышеперечисленных параметров.

Например, для разрешения выполнения локальных сценариев выполним следующую команду, и согласимся с внесением изменений, нажав Y.



В сценарии можно передавать параметры, делать их обязательными или задавать значение по умолчанию.

В Windows PowerShell предусмотрен механизм создания собственных функций, которые также как и встроенные командлеты можно будет использовать в оболочке PowerShell.

Для этого необходимо указать ключевое слово Function и затем в фигурных скобках {} написать алгоритм работы этой функции, т.е. набор команд (например, какая-нибудь часто используемая процедура: создать пользователя с определенными правами, очистить определенные каталоги и так далее). Потом необходимо сохранить все это в сценарий, но только уже с расширением .psm1, так как этот файл будет являться уже модулем.

Это еще не все, этот файл необходимо поместить в каталог, в котором PowerShell ищет модули. Таких каталогов несколько (специальный каталог в профиле пользователя, каталог, где установлен PowerShell), их можно посмотреть в переменных окружения PowerShell. Для этого выполните следующую команду

После того как Вы поместите написанный Вами модуль в специальный каталог, Вы сможете обращаться к своим функциям как к обычным командам PowerShell.

**Полезные командлеты Windows PowerShell**

Get-Help – показывает справку по командлету, функции и общую справку по Windows PowerShell. Справка бывает нескольких типов: краткая, детальная, полная и вывод только примеров;

Update-Help — загружает и устанавливает новые файлы справки, т.е. обновление справки;

Get-Command – командлет поиска нужной команды, можно искать как по глаголу, так и по существительному, также возможно использование маски, если Вы не знаете точное наименование глагола или существительного;

Get-Alias – показывает псевдонимы, все или конкретной команды;

Get-PSDrive – показывает подключенные диски;

Get-Member – выводит свойства и методы, которые есть у объекта;

Get-WindowsFeature – выводит сведения о доступных ролях и компонентах сервера;

Install-WindowsFeature (эквивалентен Add-WindowsFeature) — устанавливает роли или компоненты на указанный сервер;

Uninstall-WindowsFeature (эквивалентен Remove-WindowsFeature) – удаляет роли или компонента сервера;

Get-History — возвращает список команд, введенных в ходе текущей сессии.

**Работа с компьютером**

Windows PowerShell позволяет выполнять административные задачи для операционной системы и компьютера в целом, например, перезапустить операционную систему или переименовать компьютер.

Restart-Computer – командлет перезапускает операционную систему (перезагружает компьютер);

Stop-Computer – выключает компьютер;

Rename-Computer – переименовывает компьютер;

Checkpoint-Computer — создает точку восстановления системы на локальном компьютере;

Restore-Computer — запускает восстановление системы на локальном компьютере;

Disable-ComputerRestore — отключает функцию восстановления системы на указанном диске файловой системы;

Enable-ComputerRestore — включает функцию восстановления системы на указанном диске файловой системы;

Remove-Computer — удаляет локальный компьютер из домена;

Get-EventLog – выводит информацию о событиях в журнале событий, или список журналов событий на локальном или удаленном компьютере;

Clear-EventLog — удаляет записи из указанных журналов событий.

**Работа со службами и процессами**

PowerShell, конечно же, умеет управлять службами и процессами в Windows, для этого существуют такие командлеты как:

Get-Process – выводит информацию о запущенных процессах на компьютере;

Start-Process – запускает один или несколько процессов на компьютере;

Stop-Process — останавливает один или несколько запущенных процессов;

Get-Service – выводит информацию о службах;

Restart-Service – перезапускает службу;

Start-Service – запускает службу;

Stop-Service — останавливает службу;

Suspend-Service – приостанавливает работу службы;

Set-Service – с помощью данного командлета можно изменить свойства службы, например, описание, отображаемое имя и режим запуска. Также его можно использовать для запуска, остановки или приостановки службы.

**Получение помощи**

Одна из трудностей использования PowerShell заключается в том, что нужно знать, какие именно команды и параметры следует использовать. Изучение эффективного применения встроенных в PowerShell команд помощи является обязательным условием для использования различных команд. Во встроенной системе помощи (Help) можно найти все необходимые инструкции вместе с примерами применения. Для получения помощи по интересующему вас модулю используйте команду:

PS SQLSERVER:sql> Get-Help SQLServer

Команду Get-Help можно использовать как для изучения самих команд, так и для знакомства с примерами их применения:

PS SQLSERVER:sql> Get-Help Invoke-Sqlcmd –Examples