Powershell

PowerShell — это кроссплатформенное решение для автоматизации задач, которое включает оболочку командной строки, скриптовый язык и платформу управления конфигурацией. PowerShell поддерживается в Windows, Linux и macOS.

Windows PowerShell позволяет системным администраторам автоматизировать большинство рутинных задач. С ее помощью можно менять настройки, останавливать и запускать сервисы, а также производить обслуживание большинства установленных приложений.

О**сновные возможности**

Разумеется Windows PowerShell — это в первую очередь командная оболочка с языком сценариев, изначально созданная на основе платформы .NET Framework, а позднее — на .NET Core. В отличие от принимающих и возвращающих текстовые данные оболочек, Windows PowerShell работает с классами .NET, у которых есть свойства и методы. PowerShell позволяет выполнять обычные команды, а также дает доступ к объектам COM, WMI и ADSI. В ней используются различные хранилища, вроде файловой системы или реестра Windows, для доступа к которым созданы т.н. поставщики (providers). Стоит отметить возможность встраивания исполняемых компонентов PowerShell в другие приложения для реализации различных операций, в т.ч. через графический интерфейс. Верно и обратное: многие приложения для Windows предоставляют доступ к своим интерфейсам управления через PowerShell.

Windows PowerShell позволяет:

- Менять настройки операционной системы;
- Управлять службами и процессами;
- Настраивать роли и компоненты сервера;
- Устанавливать программное обеспечение;

При вводе команды в командной строке PowerShell сохраняет команду в журнале команд. Команды в журнале можно использовать в качестве записи о работе. Кроме того, вы можете отозвать и выполнить команды из журнала команд.

PowerShell имеет два разных поставщика журналов:

встроенный журнал и журнал, управляемый модулем **PSReadLine**. Журналы управляются отдельно, но обе истории доступны в сеансах, где **загружается PSReadLine** .

Использование журнала PSReadLine

Журнал PSReadLine отслеживает команды, используемые во всех сеансах PowerShell. Журнал записывается в центральный файл на узел. Этот файл журнала доступен для всех сеансов и содержит весь прошлый журнал. Журнал не удаляется при завершении сеанса. Кроме того, этот журнал не может управляться командлетами `*-History` .

Использование встроенного журнала сеансов

Встроенный журнал отслеживает только команды, используемые в текущем сеансе. Журнал недоступен для других сеансов и удаляется после окончания сеанса.

Командлеты журнала

PowerShell содержит набор командлетов, управляющих журналом команд.

Сочетания клавиш для управления журналом

В консоли PowerShell для управления журналом команд можно использовать следующие сочетания клавиш.

- UpArrow — отображает предыдущую команду.
- DownArrow — отображает следующую команду.
- F7 — отображает журнал команд.
- ESC — чтобы скрыть журнал.
- F8 — находит команду. Введите один или несколько символов, а затем нажмите . Снова нажмите для следующего экземпляра. клавишу F8
- F9 — поиск команды по идентификатору журнала. Введите идентификатор журнала и нажмите . Нажмите, чтобы найти идентификатор журнала и

нажмите клавишу F9. Нажмите клавишу F7 , чтобы найти идентификатор.

- #`<string>` вкладка поиск журнала `<string>*` и возврат последнего совпадения. Если вы несколько раз нажимаете Tab, она циклически проходит по соответствующим элементам в журнале.

Примечание

Эти ключевые привязки реализуются ведущим приложением консоли. Другие приложения, такие как Visual Studio Code или Терминал Windows, могут иметь разные привязки ключей. Привязки можно переопределить модулем PSReadLine. PSReadLine загружается автоматически при запуске сеанса PowerShell. При загрузке PSReadLine F7 и F9 не привязаны к какой-либо функции. PSReadLine не предоставляет эквивалентную функциональность. Дополнительные сведения см. **в about_PSReadLine**.

MaximumHistoryCount

Переменная `$MaximumHistoryCount` предпочтения определяет максимальное количество команд, которые PowerShell сохраняет в журнале команд. Значение по умолчанию— 4096.

Например, следующая команда снижает `$MaximumHistoryCount` до 100 команд:
$MaximumHistoryCount = 100

Чтобы применить этот параметр, перезапустите PowerShell.

Чтобы сохранить новое значение переменной для всех сеансов PowerShell, добавьте инструкцию присваивания в профиль PowerShell.

Порядок команд в журнале

Команды добавляются в журнал после завершения выполнения команды, а не после ввода команды. Если выполнение команд занимает некоторое время или если команды выполняются во вложенной строке, команды могут оказаться неупорядоченными в журнале. Команды, выполняемые во вложенной строке, выполняются только при выходе из уровня запроса.

CMD vs PowerShell

Первоначальное различие заключается в том, что PowerShell использует использование так называемых **командлетов.** Эти командлеты позволяют

пользователю выполнять ряд административных задач, таких как управление реестром и работу с инструментарием управления Windows. Командная строка не может выполнять такие задачи.

Если вы хоть немного знакомы с компьютерным программированием, вы узнаете о **переменных**. Эти переменные используются для хранения данных, которые можно использовать для выполнения различных операций. Командлеты PowerShell можно использовать для операций в другом командлете. Это позволяет объединить несколько командлетов для создания сложного, но эффективного командлета, который выполняет задачу.

Windows PowerShell поставляется с **Windows PowerShell ISE,** что делает его отличной средой сценариев, которую можно использовать для создания и управления различными сценариями PowerShell.

Командная строка Windows не может делать все эти вещи. Это устаревшая среда. Она вдохновлена MS-DOS и не имеет большого доступа к административным привилегиям, как Windows PowerShell.

Вообщем Командная строка является для обычного пользователя, задавать базовые команды, такие как **sfc /scannow** и т.п.

PowerShell уже более профессиональный. К примеру вот этой одной командой можно переустановить все UWP приложения в Windows 10

Просмотр журнала событий Windows PowerShell

Журнал событий Windows PowerShell можно просмотреть в средстве просмотра событий или с помощью командлетов `Get-EventLog` и . `Get-WmiObject` Чтобы просмотреть содержимое журнала Windows PowerShell, введите: Get-EventLog -LogName "Windows PowerShell"

Чтобы изучить события и их свойства, используйте `Sort-Object` командлет, `Group-Object` командлет и командлеты, содержащие `Format` глагол (`Format` командлеты).

Версия

Чтобы **узнать** установленную **версию PowerShell**, запустите консоль Windows **PowerShell** любым из способов и выполните следующую **команду**: $PSversionTable. В строке PSVersion вы увидите **версию PowerShell**. Также, чтобы **узнать версию PowerShell**, можно использовать дополнительные **команды**
: Get-Host|Select-Object Version $host.version.