1. Основные этапы установки ОС Windows 7.
2. Установка ОС Windows 7 с сервера WDS.
3. Средства развертывания ОС Windows 7. Пакет Windows AIK.
4. Средства развертывания ОС Windows 7. Пакет MDT, его отличие от Windows AIK.
5. Совместимость приложений. Пакеты, производящие проверку на совм.
6. Обеспечение совместимости приложений.
7. Перенос пользовательских данных и настроек
8. Лицензирование и активация.
9. Языковые пакеты в Windows 7.
10. Восстановление системы.
11. Windows AIK. Создание и редактирование файла ответов.
12. Windows AIK. Разделы файла ответов.
13. Среда предустановки Windows PE.
14. Подготовка образа к копированию.
15. Создание образа системы.
16. Служба развертывания Windows.
17. Развертывание с помощью MDT.
18. Инструменты автоматизации Microsoft Windows.
19. Командная строка Windows. Структура ,перенаправление команд.
20. Командная строка Windows. Команды и символы условной обработки. Переменные среды.
21. Командная строка Windows. Команды для работы с файловой системой (Cd, Chkdsk, Copy, Del, Dir, Format, Mkdir).
22. Командная строка Windows. Команды для работы с файловой системой (Move, Rename, Replace, Tree, Type, Xcopy).
23. Командная строка Windows. Команды для работы с ОС (At, Date, Mmc, Msiexec, Msinfo32, path).
24. Командная строка Windows. Команды для работы с ОС (Pause, Reg, Rem, Systeminfo, Tskill, Ver).
25. Командная строка Windows. Команды (Netstat, Ipconfig, Route).
26. Командная строка Windows. Команды для работы с протоколом TCP/IP (Ping, Ftp, Hostname, Ipconfig, Tracert).
27. Командная строка Windows. Команды для работы с сетевыми службами (Net config, Net pause, Net statistics, Net stop).
28. Командная строка Windows. Команды для раб. с пак ф-ми (echo, For, if).
29. Ком. строка Windows. Ком-ы для раб. с пак. ф-ми (Goto, Rem, Shift, Set).
30. Microsoft JScript. Операторы выбора, циклы, функции.
31. Microsoft JScript. Объекты (Array, Date, Math, String, Enumerator).
32. Сервер сцен.WSH. Объект WScript (Application, Arguments, StdOut).
33. Сервер сценариев WSH. Объект WScript (CreateObject, , ConnectObject, DisconnectObject).
34. Сервер сценариев WSH. Объект WScript (GetObject, Echo, Quit, Sleep).
35. Сервер сценариев WSH. Объект WshShell (Environment, SpecialFolders, AppActivate, Exec).
36. Сервер сценариев WSH. Объект WshShell (Popup, RegRead, RegWrite, RegDelete, Run).
37. Сервер сценариев WSH. Объект WshShell (SendKeys, SpecialFolders).
38. Сервер сценариев WSH. Объект WshNetwork (ComputerName, EnumNetworkDrives, SetDefaultPrinter, ).
39. Сервер сценариев WSH. Объект FileSystemObject(Drives, CreateFolder, MoveFolder, FolderExists).
40. Сервер сценариев WSH. Объект FileSystemObject(DriveExists, GetDriveName, CopyFile, FileExists
41. Сервер сценариев WSH. Объект FileSystemObject(GetSpecialFolder, GetTempName, CreateTextFile, OpenTextFile
42. Сервер сценариев WSH. Объект Drive (AvailableSpace, DriveLetter, DriveType, FileSystem).
43. Сервер сценариев WSH. Объект Drive (FreeSpace, IsReady, TotalSize, VolumeName).
44. Сервер сценариев WSH. Объект Folder (Attributes, DateCreated, IsRootFolder, Path, Folders).
45. Сервер сценариев WSH. Объект File (Attributes, DateCreated, ParentFolder, Size, Files).
46. Сервер сценариев WSH. Объект TextStream (AtEndOfLine, AtEndOfStream, Line, Close).
47. Особенности Windows PowerShell
48. Типы команд Windows PowerShell.
49. Формат командлета Windows PowerShell. «Глагольные» части ком-ов.
50. Windows PowerShell. Параметры командлетов.
51. Windows PowerShell. Обработка ошибок. Справочная система. Конвейер.
52. Windows PowerShell. Форматирование выводимой инф.
53. Windows PowerShell. Перенаправление выводимой инф.
54. Windows PowerShell. Типы переменных. Командлеты для работы с пер.
55. Windows PowerShell. Основные типы данных. Области видимости пер.
56. Windows PowerShell. Преобразование типов данных.
57. Windows PowerShell. Операторы сравнения.
58. Windows PowerShell. Команды для работы со строками.
59. Windows PowerShell. Массивы.
60. Windows PowerShell. Комментарии. Условные конструкции (if, Switch).
61. Windows PowerShell. Организация циклов.

**1.Основные этапы установки ОС Windows 7**

**1. Подготовка жесткого диска.**

Если на жестком диске имеется необходимая информация, то обязательно **выполните резервное копирование** перед установкой операционной системы.

**2. Загрузка с установочного диска**

Чтобы установить операционную систему, необходим установочный дистрибутив. Для загрузки с установочного диска необходимо при включении компьютера вызвать загрузочное меню, выбрать соотв. уст-во для загрузки. Также можно зайти в BIOS Setup и в разделе Boot выставить первым в очереди загрузки DVD-привод или USB Flash-диск.

**3. Определение параметров установки.**

В начале процесса установки операционной системы может потребоваться указание базовых параметров установки.

**4. Следование инструкциям мастера установки.**

При запросе указания раздела жесткого диска для установки ОС, возможны три варианта:

* **На жестком диске нет разделов**: с помощью мастера установки создайте два раздела (один для ОС размером 30-40 Гигабайт и программ, второй раздел для файлов пользователя используя всё оставшееся пространство на жестком диске).
* **На жестком диске один раздел**: желательно удалить раздел (все данные при этом будут утеряны) и создать два раздела с помощью мастера установки
* **На жестком диске два или более разделов**: оптимальный вариант, изменять разделы нет необходимости, если первый раздел имеет размер более 30 Гб.

**5. Установка драйверов.**

**Примечание:** Если диска с драйверами нет, то их необходимо загрузить заранее, до начала процесса установки ОС.

**6. Активация ОС . 7. Настройка ОС.**

Настройка ОС включает: конфигурирование параметров доступа к сети и

интернету, задание прав учетных записей пользователей, настройку

параметров обновления ОС, конфигурирование интерфейса ОС и т.д.

1. **Установка ОС Windows 7 с сервера WDS.**

**WDS (Windows Deployment Services**) - серверная технология Microsoft для установки по сети операционных систем семейства Windows.

**Достоинства:**

* централизованное хранение образов и управление образами
* возможность одновременной установки большого количества компьютеров
* возможность выбора сервера с дистрибутивом
* возможность автоматической установки с ближайшего доступного сервера
* возможность загрузить по сети (PXE boot)

**PXE (Preboot Execution Enviroment)** – среда загрузки компьютеров с сетевой карты без использования съемных устройств.

**Автоматический захват и применение образа**

WDS связывается с конечным компьютером по PXE для загрузки Windows PE, которая выполняет установку и настройку будущей клиентской системы. WDS включает в себя хранилище для образов операционных систем, загружаемых по PXE, а также хранилище для образов ОС, которые будут в конечном итоге установлены на удаленный компьютер.

В случае если доступна мульти загрузка, WDS предоставляет конечному пользователю boot –меню для выбора образа для загрузки.

1. **Средства развертывания ОС Windows 7. Пакет WinAIK.**

**Состав пакета Windows Automated Installation Kit:**

* ImageX – используется для создания, изменения, применения и записи образов Windows
* Система обслуживания образов (DISM) – используется для добавления в образ обновлений, драйверов, языковых пакетов, установки необходимых компонентов.
* Диспетчер установки (SIM) – используется для создания файлов ответов, открытия браузеров.
* Среда предустановки (Windows PE) – минимальная среда ОС загружаемая перед началом процесса установки.
* Средство миграции пользовательской среды (USMT) – предназначено для переноса пользовательских данных.

1. **Средства развертывания ОС Windows 7. Пакет MDT, его отличие от Windows AIK.**

**Microsoft Deployment Toolkit (MDT)** предоставляет унифицированную коллекцию средств, процессов и инструкций для автоматизации развертывания настольных компьютеров и серверов. MDT обеспечивает улучшенную безопасность и непрерывное управление конфигурациями.

**Возможности:**

* Установка приложений по окончанию установки ОС
* Устранение проблем совместимости
* Перемещение профилей пользователя
* Упаковка и установка приложений
* Создание и управление общими дистрибутивными ресурсами
* Создание и обслуживание образов

Поддержка PowerShell позволяет автоматизировать многие задачи при помощи скриптов.

MDT обеспечивает единый, простой способ использования инструментов Windows AIK. Большинство инструментов запускаются только из командной строки, и у каждого — уникальный синтаксис, что увеличивает время их освоения и затрудняет использование. MDT упрощает применение инструментов Windows AIK благодаря удобным мастерам, задающим простые вопросы. Внутри механизма MDT учитываются все особенности синтаксиса каждого инструмента.

1. **Совместимость приложений. Пакеты, производящие проверку на совместимость.**

**Аппаратная совместимость -** рассматривается соответствие физической составляющей комп-ов требованиям, необходимым для корректной работы ОС. Чтобы обеспечить аппаратную совместимость, каждый тип прогр-го обеспечения разрабатывается для конкретной аппаратной платформы.

**Программная совм. -** совместимость приложений и драйверов устр-в.

**MAP( Microsoft Assessment and Planning Toolkit)** – утилита, которая собирает данные о ресурсах виртуальной или физической машины и проводит анализ.

**МАР содержит:**

* Windows 7 Readiness Assessment - определяет совместимость программ и аппаратного обеспечения для ОС.
* Windows Server 2008 Readiness - определяет совместимость прогр. и аппаратного обеспечения для поддержки Windows Server 2008
* Virtual Machine Discovery - составляет отчет о компонентах, запущенных в среде виртуализации.

**Достоинства MAP:**

* Различные способы обнаружения комп. в сети
* Хранение данных в SQL

**Недостаток MAP:** может быть получено недостаточно полная информация о программной совместимости.

**Application Compatibility Toolkit (ACT**) –набор, который помогает определить, совместимы ли приложения, устройства и компьютеры с версиями операционной системы Windows. Инструменты в составе ACT помогают проанализировать проблемы совместимости и смягчить их последствия до развертывания версии Windows.

* Application Compatibility Toolkit (ACT) - позволяет собирать хранить и анализировать информацию, поступающую от клиентов
* Internet Explorer Compatibility Test Tool (IECTT) - выполняет диагностирование веб-приложений, показывая вопросы в режиме реального времени и, может загрузки результаты в базу данных ACT.
* Standard User Analyzer

**Оценки совместимости:**

- Сборщик информации об инвентаризации (Inventory Collection)

- Оценка совместимости с контролем уч. записей

- Оценка совместимости обновлений

- Оценка совместимости с ОС Windows

**6. Обеспечение совместимости приложений**

**Оценки совместимости:**

- Сборщик информации об инвентаризации (Inventory Collection)

- Оценка совместимости с контролем уч. записей (Account Control Compatibility Evaluation)

- Оценка совместимости обновлений (Update Compatibility Evaluation)

- Оценка совместимости с ОС Windows (Windows Compatibility Evaluator)

**Исправление ошибок совместимости в ACT:**

1. Определить наиболее значимые приложения

2. Определить проблемы совместимости

3. Решить проблемы совм. при помощи прил., входящих в состав ACT

4. Развернуть прил. и фиксы и протестир. их на клиентских машинах.

**Запуск несовместимых приложений:**

 - Изменение кода приложения таким образом, чтобы оно соответствовало требования по корректной работе в ОС.

- Использование режима совместимости (Compatibility Mode)

- Запуск приложения в виртуальной среде

Возможно создание **инкапсулированного приложения**. Должен быть включен режим совместимости. Ярлык программы копируется в C:\Documents and Settings\All Users\Start Menu\Programs. После этого ярлык появляется в стартовом меню основной ОС.

**Microsoft Enterprise Desktop Virtualization (MED-V**) – корпоративное средство для доставки образов Virtual PC компьютерам с центрального хранилища.

**Windows XP Mode** – виртуальная машина с установленной XP.

**Microsoft Application Virtualization (App-V**) - корпоративное средство, которое используется для централизации управления всем жизненным циклом приложения.

1. **Перенос пользовательских данных и настроек**

Перенос пользовательских данных с помощью **WET (Windows Easy Transfer)**.

**Способы переноса данных:**

* **кабель** - Easy Transfer Cable.Один конец кабеля подключается к компьютеру-источнику, второй — к компьютеру-приёмнику. Оба компьютера должны быть включены во время переноса данных, а также на них должно быть запущено Средство переноса данных Windows.
* **сетевое подключение.** Чтобы выполнить перенос данных по сети, необходимы два компьютера, на которых запущено Средство переноса данных Windows и которые подключены к одной сети. При переносе данных через сеть на компьютере-источнике задается пароль, который затем нужно ввести на компьютере-приёмнике.
* **съемное устройство**

**Переносимые данные**: файлы и папки, учетные записи, сообщения, эл. почта, контакты, мои документы, настройки подключения к сети Интернет, настройки программ, настройки Windows, хранящиеся в реестре

**USMT(User State Migration Tools)** – набор утилит, предназначенных для переноса пользовательских данных.

Пакет может использоваться для параллельных миграций (при замене оборудования) и поэтапных (при обновлении ПО).

**USMT вкл. следующие компоненты:**

* ScanState.exe для резервного копирования. Собирает файлы и настройки с компьютера-источника.
* LoadState.exe для восстановления (применения польз-их данных)
* XML – файлы для определения файлов для резервного копирования и восстановления:

MigApp.xml, MigDocs.xml, MigUser.xml

* USMLUtils.exe - для удаления хранилища миграции.
1. **Лицензирование и активация**

**Служба управления ключами KMS** – активация системы внутри своей корпоративной сети.

**Ключ многократной активации MAK** – единовременная активация системы с использованием размещенных служб активации Microsoft (Для комп, не имеющих выход в Интернет)

Для KMS активации необходимо не менее 5 серверов и не менее 25 компьютеров.

**KMS- активация:**

 с помощью службы управления ключами активацию можно выполнить в локальной сети, что исключает необходимость подключения отдельных компьютеров к серверам Microsoft для активации продуктов, т.к. это не всегда правильно с точки зрения безопасности.

**MAK-активация:**

 ключ MAK используется для однократной активации с помощью служб активации, размещенных на веб-сайтах Microsoft.

1. **Языковые пакеты в Windows 7.**

**Языковой пакет** – пакет, представляющий пользователю локализованный язык интерфейса, установленный поверх двоичных файлов ОС.

**Варианты многоязыкового развертывания:**

* развертывание образа, содержащего несколько языков, но в качестве языка по умолчанию активен только один.
* развёртывание многоязычного образа, позволяющего пользователю переключаться между различными языками.

**Полный языковой пакет** содержит 100% ресурсов, необходимых для локализации пользовательского интерфейса, требует лицензии.

**Пакет интерфейса пользователя (LIP)** –частичный язык, не требует лицензии.

Язык по умолчанию удалить нельзя.

Образ может содержать любое нужное количество языковых пакетов.

1. **Восстановление системы**

**Среда восстановления Windows (Windows RE)** представляет собой расширяемую платформу восстановления, основанную на среде предварительной установки Windows (Windows PE). Если при загрузке компьютера происходит сбой, Windows автоматически переключается в эту среду, а средство восстановления при загрузке в Windows RE автоматизирует процедуру диагностики и восстановления незагружаемой копии Windows. Кроме того, среда Windows RE является исходной точной для запуска различных средств ручного восстановления системы.

**Преимущества:**

* добавление дополнительных средств восстановления в WinRE
* возможность использования образа в качестве платформы для восстановления
* поддержка драйверов накопителей и языковых пакетов посредством инструментов WinPE

**Ручная диагностика и восстановление:**

* Startup Repair – восстановление запуска в автоматическом режиме
* System Restore – восстановление системы позволяет вернуться к точке , созданной до возникновения проблемы
* System Image Recovery – мастер, который проведет пользователя через шаги восстановления системы образа, заранее созданного средствами архивации Windows.
* Windows Memory Diagnostic - диагностика памяти Windows проверяет оперативную память на ошибки.
* Command Prompt – командная строка
1. **Windows AIK. Создание и редактирование файла ответов**

**Windows AIK** представляет собой набор инструментальных средств, разработанных для поддержки конфигурации и развертывания операционных систем Windows.

**Файл ответов.** Отображает создаваемый файл ответов. В него можно добавлять компоненты и пакеты.

**Создание файла ответов:**

1) Открыть WSIM

2) Откроем наш образ install.wim в WSIM.
    Меню Файл -> Пункт Выбрать образ Windows -> Выбираем образ (редакцию и разрядность) из появившегося списка.

3) Создадим файл ответов.
    Меню Файл -> Пункт Новый файл ответов

 В панели Файл ответов появится новый файл ответов, содержащий в себе элементы, представляющие различные стадии (проходы) настройки Windows во время установки

**Разделы файла ответов:**

* components - содержит все параметры компонентов, применяемые в ходе установки Windows.
* packages - используется для распространения обновлений ПО.

**Редактирование файла ответов**

Отредактировать файл ответов можно с помощью инструмента Windows SIM. Группа по созданию образов компьютерных систем может редактировать файл ответов, привязанный к каждой сборке, добавленной к дистрибутивному общему ресурсу. С помощью консоли Deployment Workbench файл ответов сборки загружается в инструмент Windows SIM.

1. **Windows AIK. Разделы файла ответов.**

Windows AIK представляет собой набор инструментальных средств, разработанных для поддержки конфигурации и развертывания операционных систем Windows.

В файле ответов содержится **два раздела параметров:**

### Компоненты

Раздел компонентов файла ответов содержит все параметры компонентов, применяемые в ходе установки Windows. Компоненты организованы в соответствии с различными этапами настройки: **windowsPE**, **offlineServicing**, **generalize**, **specialize**,  **auditSystem**, **auditUser** и **oobeSystem**. Каждый этап настройки представляет собой отдельную фазу работы программы установки Windows. Параметры можно применить во время одного или нескольких этапов.

### Пакеты

Корпорация Майкрософт использует пакеты для распространения обновлений программного обеспечения, пакетов обновления и языковых пакетов. Пакеты могут также содержать отдельные компоненты Windows. Можно настроить пакеты и изменять параметры компонентов, входящих в состав пакета.

Компоненты Windows могут быть включены или отключены. Если компонент включен, все ресурсы, исполняемые файлы и параметры этого компонента доступны для пользователей системы. Если компонент отключен, то ресурсы пакета недоступны, но не удаляются из системы.

Некоторые компоненты Windows требуют установки других компонентов, прежде чем вы сможете включить установленную версию Windows.

Пакеты в файле ответов применяются к образу Windows на этапе настройки **offlineServicing**. Чтобы добавить пакеты в автономный образ Windows, вы можете использовать систему обслуживания образов развертывания и управления ими (DISM).

1. **Среда предустановки Windows PE.**

**Windows PE(Windows Preinstallation Environment)** — это предустановочная среда Windows, представляет собой сокращенную версию Windows и применяется, главным образом, для развертывания ОС на клиентских компьютерах.

**Возможности Windows PE:**

* встроенная поддержка файловой системы NTFS 5.x, включая создание динамических томов и управления ими
* встроенная поддержка сетей TCP/IP и совместного доступа к файлам
* встроенная поддержка 32-разрядных драйверов устр-в для Windows
* встроенная поддержка подмножества программ интерфейса API
* поддержка Windows Management Instrumentation
* возможность запуска с различных типов носителей
* включены все драйверы Hyper-V за исключ. драйверов монитора
* поддержка запоминающих устройств, мыши и сетевых адаптеров

**Ограничения:**

* ограниченный набор интерфейсов
* перезагрузка после 72 часов непрерывного использования
* не может использоваться в качестве файлового сервера или сервера терминалов
* поддержка только протокола TCP/IP и NetBIOS
* измерения в реестре не сохраняются
* не поддерживается оболочка Microsoft .Net

**Этапы загрузки Windows PE:**

* выполняется загрузка с использованием загрузочного сектора носителя
* управление передается программе Bootmgr которая извлекает основные сведения загрузки из данных конфигурации загрузки
* Winload.exe загружает необходимый аппаратно-зависимый уровень, куст системного реестра и необх. драйверы
* Ntoskrnl.exe завершает установку среды
* модуль SMSS загружает оставшуюся часть реестра, настраивает среду для запуска подсистемы Win32 и ее процессов
* Winlogon.exe выполняет установку основываясь на значении реестра
* Winpeshl.exe
* Wpeinit.exe загружает окно командной строки
1. **Подготовка образа к копированию**

**Sysprep** — утилита системной подготовки [Microsoft Windows](https://ru.wikipedia.org/wiki/Microsoft_Windows) к развертыванию.

Sysprep(%WINDIR%\system32\sysprep).

**Преимущества:**

* Удаление из Windows системных данных;
* Настройка Windows для загрузки в режиме аудита. Режим аудита позволяет устанавливать приложения сторонних лиц и драйверы устройств, а также проверять работоспособность компьютера.
* Настройка загрузки экрана приветствия Windows.
* Сброс активации Windows ( не более 3-ех раз)

**Удаление системных данных:**

* установить Windows на компьютер-образец
* установить необходимые драйверы и приложения
* выполнить sysprep/oobe/generalize

/generalize удаляет данные, относящиеся к операционной системе: журналы событий, SID, и т.д.

/oobe предписывает запускать экран приветствия при следующем запуске компьютера

**Режим аудита:**

* 1. нажать во время работы экрана приветствия СTRL+SHIFT+F3

При загрузке Windows существует два возможных режима запуска компьютера.

* **Экран приветствия Windows**. Экран приветствия Windows, также называемый запуском при первом включении компьютера, является первым средством, с которым сталкиваются пользователи, предлагающим им настроить свою установку
* **Режим аудита**.
	1. в файл ответов добавить компонент “Microsoft-Windows-Deployment” в этапе настройки oobeSystem. В параметре “Reseal\Mode” указать “Audit”. После установки выполнить перезагрузку в режиме аудита.
	2. выполнить команду sysprep/audit в окне командной строки
1. **Создание образа системы**

**ImageX** – средство командной строки, которое позволяет записывать, изменять, и применять образы дисков для быстрого развертывания. входит в состав Windows AIK.

**Возможности ImageX:**

* **создание образа**

Сценарий:

* 1. загрузка на технический компьютер Windows PE
	2. запись образа с помощью ImageX
	3. предоставление общего доступа к образу по сети
	4. применение образа к компьютеру
* **изменение файла образа**

 Сценарий:

1. настройка существующего образа, включая обновление папок и ф-ов;
2. добавление, удаление, редактирование и копирование файлов образа с помощью драйвера фильтра файловой системы WIM и средства управления файлами.
3. **Служба развертывания Windows**

**Запуск программы установки Windows из сетевой папки:**

1. загрузить среду Windows PE со съемного загрузочного устройства;
2. проверить подключение к сети;
3. подключится к общей сетевой папке;
4. запустить программу установки Windows

Для автоматизации данного процесса файл ответов должен находится в папке Sourse, а не на одном уровне с файлом Setup.exe.

Применение образа из сетевой папки

**Создание образа:**

* установить ОС на технический компьютер
* установить необходимые приложения, настроить ОС
* запустить утилиту sysprep для подготовки комп. к развертыванию
* создать загрузочный съемный носитель с образом Windows PE и утилитой ImageX
* загрузить систему с загрузочного носителя
* получить образ ОС с технического компьютера
* загрузить образ на сервер

**Применение образа:**

* загрузить компьютер с загрузочным диском
* проверить подключение к сети;
* разбить диск с помощью утилиты Diskpart
* применить образ к компьютеру с помощью утилиты ImageX
* при необходимости сделать загрузочный раздел диска

**Windows Deployment Services –** установщик образов на пустые системы по сети.

**Компоненты сервера:**

* хранилища образов
* PXE (Pre-Boot Execution Enviroment) службы
* TFTP службы
* сетевой уровень

**Компоненты клиента:**

* включают клиента WDS, которрый отображает меню выбора образа установки Windows. Необходимый образ записывается в хранилище образов серверов. Windows Setup устанавливает загруженный образ.

**Компоненты управления:**

* консоль Windows Deployment Services ММС
* инструмент командной строки
1. **Развертывание с помощью MDT.**

**Информационный центр** – предоставляет доступ к руководству Getting Started и прочей документации MDT.

**Общий ресурс развертывания –** подпапки приложения, ОС, драйверы и тд

**Файлы конфигурации:**

* customSettings.ini - глабольный файл конфигурации, используется для управления сценариями, которые автоматизируют процесс установки.
* bootstap.ini - используется в сценариях развертывания новых компонентов и позволяет целевым ком. подключаться к точке установки, чтобы начать процесс установки.

**Microsoft Deployment Toolkit (MDT)**

Шаг 1. Настройте разрешения Active Directory

Шаг 2. Настройте общую папку развертывания MDT

Шаг 3. Добавление пользовательского образа

Шаг 4. Добавление приложения

Шаг 5. Подготовьте хранилище драйверов

Шаг 6. Создайте последовательность задач развертывания

Шаг 7. Настройте общую папку развертывания MDT для рабочей среды

Шаг 8. Развертывание образа Windows

1. **Инструменты автоматизации Microsoft Windows**

**Требования к инструменту автоматизации:**

* возможность использования в разных версиях ОС без установки какого-либо дополнительного программного обеспечения;
* интеграция с командной строкой;
* согласованный и непротиворечивый синтаксис команд и утилит;
* наличие подробной встроенной справки
* возможность выполнения сценариев

**Инструменты:**

* **командная строка** - часть ОС обеспечивающая базовые возможности управления компьютеров посредством ввода команд через интерфейс командной строки или последовательное исполнение пакетных командных файлов.
* **Windows Script Host** - компонент Microsoft Windows, предназначенный для запуска сценариев на скриптовых языках JScript и VBScript, а также и на других дополнительно устанавливаемых языках.
* **Microsoft PowerShell** – расширяемое средство автоматизации, состоящее из оболочки с интерфейсом командной строки и сопутствующего языка сценариев.
1. **Командная строка Windows. Структура команд. Перенаправление команд**

**Командный интерпретатор или оболочка командной строки** - часть ОС обеспечивающая базовые возможности управления компьютеров посредством ввода команд через интерфейс командной строки или последовательное исполнение пакетных командных файлов.

**Структура команд:**

* внутренние, распознаваемые и выполняемые непосредственно самим командным интерпретатором;
* внешние, требуют загрузки специальных программ

**Формат команд:**

> **команда [параметры и ключи]**

**Правила написания команд:**

* параметры отделяются от команд пробелом;
* для вкл./выкл. опции перед параметром используется знак + или –
* параметр не должен начинаться со специальных символов
* параметр : дополнительный параметр

Большинство команд имеют дополнительную справку (/?)

**Возможно:**

* выводить сообщения команд не на экран, а в файл или на принтер;
* читать входные данные не с клавиатуры, а из заранее подготовленного файла;
* передавать сообщение, выводимое одной программой , в качестве входных данных для другой команды;

**Дескрипторы:**

* **STDIN (0) –** ввод с клавиатуры;
* **STDOUT (1) –** вывод в окно командной строки;
* **STDERR (2) –** ошибка вывода в окно командной строки;
* **UNDEFINDER (3-9) –** прикладная программа.

Для перенаправления команд используется след. запись:

 **&номер\_дескриптора**

номер\_дескриптора\_1>&номер\_дескриптора\_2 – считывает входные данные дескриптора 2 как входные данные дескриптора 1.

номер\_дескриптора\_1<&номер\_дескриптора\_2 – наоборот

**Команды перенаправления:**

* команда >(<)имя файла
* команда >>имя файла (добавление)
1. **Командная строка Windows. Команды и символы условной обработки. Переменные среды.**

**Командный интерпретатор или оболочка командной строки** - часть ОС обеспечивающая базовые возможности управления компьютеров посредством ввода команд через интерфейс командной строки или последовательное исполнение пакетных командных файлов.

**Команды и символы условной обработки**

|  |  |  |
| --- | --- | --- |
| Символ | Синтаксис | Определение |
| & | ком1&ком2 | разделение нескольких команд |
| && | ком1&&ком2 | ком2 выполняется только, если ком1 была выполнена успешно |
| **| |** | ком1 **| |** ком2 | ком2 выполняется только, если ком1 была выполнена с ошибкой |
| ( ) | (ком1&ком2) | группирование или вложение команд |

**Переменные среды** используются для хранения текстовых строк пользователя и информации о настройках операционных систем.

|  |  |  |
| --- | --- | --- |
| Переменные | Тип | Описание |
| ALLUSERSPRO-FILE | лок. | Возвращает путь к «All Users». |
| APPDATA | сист | Возвращает путь к Application Data |
| HOMEPATH | сист | Возвращает полный путь к основному каталогу пользователя.  |
| HOMESHARE | сист | Возвращает сетевой путь к [общему](https://ru.wikipedia.org/wiki/%D0%9E%D0%B1%D1%89%D0%B8%D0%B9_%D1%80%D0%B5%D1%81%D1%83%D1%80%D1%81) основному каталогу пользователя. |
| OS | сист | Возвращает имя ОС |
| PATH | сист | Указывает путь поиска исполняемых файлов. |
| SYSTEMDRIVE | сист | Диск, на котором расположен корневой каталог Windows. |
| SYSTEMROOT | сист | Путь к корневому каталогу Windows. |
| TEMP и TMP | сист и лок | Возвращает путь временным папкам |

**21. Командная строка Windows. Команды для работы с файловой системой (Cd, Chkdsk, Copy, Del, Dir, Format, Mkdir).**

**Командная строка –** часть ОС, обеспеч. базовые возможности управления комп. посредством ввода команд через интерфейс ком.строки или последов. исполнения пакетных командных файлов.

**Chdir (Cd)** – ввод имени текущего каталога или переход в др папку.

Cd\- переход в корневой каталог

Cd/de: - переход на диск e:

Cd.. – переход в родит.каталог

c: - переход на диск c

**Chkdsk –** выводит на экран отчет о состоянии диска, сост список ошибок и исправления их.

**Copy –** копирование одного или нескольких файлов из одного расположения в другие. Для объединения нескольких файлов в один можно перечислить их через знак «+» или подставочные знаки (\* или ?). Команда не копирует файлы нулевой длины.

Copy /ve:\a.txtd: - копирует файл a.txt с диска e: на диск d:

Copy /va.txt + b.txtc.txt – копир. 2-ух файлов

**Del (erase) –** удаление заданных файлов.

**Dir –** вывод списка файлов и подкаталогов каталога.

dire:\n \*.\*/os – выводит файлы имена, которые начинаются с символа «n», отсортир. по размеру файла, таблицу.

**Format –** форматирование диска. ТипФС: FAT, FAT32 или NTFS.

**Mkdir (md) –** создание папки или подпапки.

**22. Командная строка Windows. Команды для работы с файловой системой (Move, Rename, Replace, Tree, Type, Xcopy).**

**Командная строка –** часть ОС, обеспеч. базовые возможности управления комп. посредством ввода команд через интерфейс ком.строки или последов. исполнения пакетных командных файлов.

**Move –** перемещение одного/ нескольких ф-ов из одного кат. в другой.

**Rename (ren) –** изменение имени ф-ла или набора файлов.

**Replace –** заменяет файлы в одном каталоге файлами с теми же именами из др каталога. Команда так же может быть использована для добавления только тех ф-ов, кот.еще нет в каталоге.

**Tree –** представляет графически дерево каталогов заданного пути или диска.

**Type –** вывод содержимого текстового ф-ла на экран. Применяется для просмотра текстового ф-ла без его изменения.

**Xcopy –** копирует файлы и каталоги, включая подкаталоги.

**23. Командная строка Windows. Команды для работы с ОС (At, Date, Mmc, Msiexec, Msinfo32, path).**

**Командная строка –** часть ОС, обеспеч. базовые возможности управления комп. посредством ввода команд через интерфейс ком.строки или последов. исполнения пакетных командных файлов.

**At –** запуск программ и команд по расписанию можно использовать только при запущенной службе расписаний.

**Date –** отображениетекущей системы даты с возможностью введения новой.

**Mmc –** открытие окна ММС.

**Msiexec –** устанавливает Windows, позволяют из командной строки установить и настроить продукт.

**Msinfo32 –** отображение подробных сведений об оборудовании, сист. комп. и среде ПО.

**Рath–**устанавливает путь поиска в переменной среды, кот.явл. набором каталогов

**24. Командная строка Windows. Команды для работы с ОС (Pause, Reg, Rem, Systeminfo, Tskill, Ver).**

**Командная строка –** часть ОС, обеспеч. базовые возможности управления комп. посредством ввода команд через интерфейс ком.строки или последов. исполнения пакетных командных файлов.

**Pause –** приостановка выполнения пакетной программы с выводом сообщ. об ожидании нажатия польз.любой клавиши.

**Reg-** набор команд, кот добавляют, измен и отображают на экране информацию разделов реестра и зн записей реестра.

**Rem–**добавляет комментарии в пакетные файлы или файлы надстроки.

**Systeminfo–**выводит на экран подробные сведения о конфигурации компьютера и ОС.

**Tskill–**завершает процесс.

**Ver–**отображение номера версии Windows.

**25. Командная строка Windows. Команды для работы в сети (Netstat, Ipconfig, Route).**

**Командная строка –** часть ОС, обеспеч. базовые возможности управления комп. посредством ввода команд через интерфейс ком.строки или последов. исполнения пакетных командных файлов.

**Netstat –** отображение активных подключений ТСР портов, таблиц материализованных IP и тд.

**Ipconfig –** отображение всех текущих параметров сети TCP/IP и обновления параметров DHCP и DNS.

**Route –** выводит на экран и изменяет записи в локальной **таблице** IP- маршрутизации.

**26. Командная строка Windows. Команды для работы с протоколом TCP/IP (Ping, Ftp, Hostname, Ipconfig, Tracert).**

**Командная строка –** часть ОС, обеспеч. базовые возможности управления комп. посредством ввода команд через интерфейс ком.строки или последов. исполнения пакетных командных файлов.

**Ping –** проверка соединения на уровне протокола **IP,** с др компьютером поддерживает **TCP/IP.**

**Ftp –** позволяет передавать файлы комп и на комп с пом программ, поддерживая протокол **Ftp.**

**Hostname –** отображение имени узла, входящего в состав полного имени компьютера.

**Ipconfig –** отображают всех текущих параметров сети **TCP/IP** и обновление параметров DHCP и DNS.

**Tracert–** определяет путь до точки назначения с помощью посылки в точку назначения эхо-сообщения проток ICMP с постоянувелич срока жизни.

**27. Командная строка Windows. Команды для работы с сетевыми службами (Netconfig, Netpause, Netstatistics, Netstop).**

**Командная строка –** часть ОС, обеспеч. базовые возможности управления комп. посредством ввода команд через интерфейс ком.строки или последов. исполнения пакетных командных файлов.

**Netconfig –** вывод сведений о запущнастраев службах, а также просмотра и изменения параметров службы

**Netpause –** приостановка системных служб в windows.

**Netstatistics –** вывод журнала статистики.

**Netstop –** остановка службы.

**28. Командная строка Windows. Команды для работы с пакетными файлами (echo, For, if).**

**Командная строка –** часть ОС, обеспеч. базовые возможности управления комп. посредством ввода команд через интерфейс ком.строки или последов. исполнения пакетных командных файлов.

**Echo [{on|off}] [сообщение] –** выводит сообщение на экран текст.

**{on|off} –** вкл или отключение режима отображения на экране инф о работе команды.

**[сообщение] –** текст, выводимый на экран если требуется отключить отображение команд, то необходимо перед командой поставить знак @.

@ echooff – отключение вывода команд.

**For -** Команда **FOR** используется для выполнения, заданной в качестве параметра, команды для каждого элемента из набора. В качестве элементов могут использоваться файлы, каталоги, наборы строк.
Формат командной строки:
**FOR %переменная IN (набор) DO команда [параметры]**
Параметры:
**%переменная** - Однобуквенный подставляемый параметр.
**(набор)** - Определяет набор, состоящий из одного или нескольких элементов, обрабатываемых командой.
**команда** - Команда, которую следует выполнить для каждого элемента набора.
**параметры**- Параметры для команды, выполняемой по отношению к элементам набора.

**If** обработкаусловий в пакетных файлах

 **Not**-команда выполн в случае, если условие имеет значение false.

 **Условие**-ком. , кот. Должна выполн в случае, если условие имеет значение true.

 **Команда**-ком., кот должна быть отображена в случае выполнения условия.

 **Выражение**-ком и все ее параметры для обработки в ком строке при выполн оператора else.

**29. Командная строка Windows. Команды для работы с пакетными файлами (Goto, Rem, Shift, Set).**

**Командная строка –** часть ОС, обеспеч. базовые возможности управления комп. посредством ввода команд через интерфейс ком.строки или последов. исполнения пакетных командных файлов.

**Goto метка.**Переход к строке, об.меткой. Метка должна находиться в отдельной строке и начинаться с двоеточия. Выполнение файла продолж. со след поле метки, строки. Команда goto:eof позволяет передать управление в конец файла.

**Rem[ текст ] –** комментарий.

**Shift–** изменяет положение пактных параметров в пак. ф-ах. Изменяет значения заменяемых параметров %0-%9 путем копирования каждого параметра в предыдущем. Используется при написании пакетных файлов, выполняющих одну и ту же операцию над любым числом параметра.

**Set[ ключ ] [переменная = [значение параметра]]** создание, изменение, отображение и удаление переменных среды.

**Ключи:**

**/A переменная = [арифметическое выражение]** указывает на то, что строка справа от знака = является числом или арифметич выражения. При использовании строк, содерж строковое представление числа, они преобразуются в число. Если переменная не определена, то она получает 0.

**30. MicrosoftJScript. Операторы выбора, циклы, функции.**

**Jscript –** реализация стандарта языка JavaScriptвыполняется **Microsoft.**

**Jscript –** интерпретируемый объектно-ориентированный язык.

В**Jscript**для **WSH (WindowsScriptHost)** отсутствуюттэгиHTML, т.к. он в основном используется для автоматизации администрирования систем **MicrosoftWindows.**

**Особенности языка:1)**Прописные и строчные буквы различны.**2)**Комментарии // или /\* \*/. **3)**Конец строки заканчивается «;». **4)**Блоки помещаются в {}. **5)**Операция присваивания «=», операция сравнения «==». **6)**Разрешено не объявлять переменные. В выражениях могут использоваться только в объявлениях переменных. При объявлении желательное использование конструкции var перед именем переменной var обязательно только приобъявлении переменной внутри функции. **7)**Преобразование типов выполняется автоматически, при необходимости. Например при сложении строки и числп, число будет преобразовываться в строку.

**Типы данных:** 1. Переменные не имеют строгого типа. 2. Как правило тип переменной соответствует данным которые помещены в переменную. 3. При объявлении возможна инициализация.

for( переменная in{ объект| массив} )

{ операторы }

**Прототип пользования функции:**

functionимя\_ф-ии (параметры)

{ операторы

return [ возвращает результат] }

Встроенные ф-ии служат для обработки выражений и специальных символов, а так же преобразования числовых и строковых переменных.

Объекты являются совокупностями методов и свойств. Обработка объектов и массивов идентична. Можно обратиться к любой части объекта (его свойствам и методам) либо по имени, либо по индексу (нумерация начинается с нуля).

**31. Microsoft JScript. Объекты (Array, Date, Math, String, Enumerator).**

**Jscript –** реализация стандарта языка JavaScript выполняется **Microsoft.**

**Jscript –** интерпретируемый объектно-ориентированный язык.

В **Jscript**для **WSH (WindowsScriptHost)** отсутствуют тэги HTML, т.к. он в основном используется для автоматизации администрирования систем **MicrosoftWindows.**

**Array.** Создание и работа с массивами.

имя\_массива=newArray([size]);

имя\_массива=newArray([эл0, эл1, эл2…элN]);

**Date–**работа с датой и временем.

**Math–**работа с математическими функциями.

**String–**работа со строками, форматирование строк, поиск подстрок в строках.

имя=newString ([«Строка»])

**Enumerator–**работа с элементами коллекции.

имя=newEnumerator ([коллекция])

К этим коллекциям нельзя обратиться напрямую (по индексу), можно только перемещать указатель на первый или последний элемент коллекции.

Методы**Enumerator:**

* atEnd() возвращает . если элемент находиться в конце.
* item() возвращает текущий элемент коллекции.
* moveFirst() переход к 1-ому элементу коллекции
* moveNext() перемещает текущий элемент к следующему элементу коллекции.

**32. Серверсценариев WSH. ОбъектWScript (Application, Arguments, StdOut).**

Главный объект WSH (WindowsScriptHost) служит для создания других объектов или связи с ними содерж инф о сервере сценариев, а также позвол выводить информацию в стандартный выходной поток и читать данные из стандвх потока, предоставл методы для работы внутри сценария с объектами автоматизации.

Экземпляр объекта**WScript**создается сервером сценариев автоматически (для использования других объектов нужно использовать метод CreateObject или св-во другого объекта).

**Application–**представляет интерфейс ID для объекта **WScript.**

Name= WScript. Application

**Arguments–**содержит указатель на коллекцию, содержащую параметры командной строки для исполнения сценария.

**StdOut–**позволяет запущенный сценарий записывать инф в стандартный выходной поток.

Доступ к станд входным и выходным потокам имеется только в консольном режиме.

**33. Серверсценариев WSH. ОбъектWScript (CreateObject, , ConnectObject, DisconnectObject).**

Главный объект WSH (WindowsScriptHost) служит для создания других объектов или связи с ними содерж инф о сервере сценариев, а также позвол выводить информацию в стандартный выходной поток и читать данные из стандвх потока, предоставл методы для работы внутри сценария с объектами автоматизации.

Экземпляр объекта**WScript**создается сервером сценариев автоматически (для использования других объектов нужно использовать метод CreateObject или св-во другого объекта).

**CreateObject(PrID [,Pref]) –** создает объект, необязателный параметр Pref позволяет при возникновении опред. события вызывать ф-цию, имя которой состоит из заданного префикса Pref и имени события.

**ConnectObject(Obj, Pref)**Устанавливает соединение с объектом Obj, что позволяет обрабатывать его события функциями, имена которых совпадают с Pref**.**

**DisconnectObject(Obj)**Описывает объект**.**

**34. Сервер сценариев WSH. ОбъектWScript (GetObject, Echo, Quit, Sleep).**

Главный объект WSH (WindowsScriptHost) служит для создания других объектов или связи с ними содерж инф о сервере сценариев, а также позвол выводить информацию в стандартный выходной поток и читать данные из стандвх потока, предоставл методы для работы внутри сценария с объектами автоматизации.

Экземпляр объекта**WScript**создается сервером сценариев автоматически (для использования других объектов нужно использовать метод CreateObject или св-во другого объекта).

**GetObject(fname [,PrID][Pref])**Активизирует объект автоматизации, определенный заданным файлом(fname) или парам. PrID**.**

**Echo**Вывод текстовой информации на экран**.**

**Quit([err])**Прерывает выполнение сценария с заданным Err кодом выхода**.**

**Sleep(t)**Приостанавливает выполнение сценария на t миллисекунд**.**

**35. Серверсценариев WSH. ОбъектWshShell (Environment, SpecialFolders, AppActivate, Exec).**

Главный объект WSH (WindowsScriptHost) служит для создания других объектов или связи с ними содерж инф о сервере сценариев, а также позвол выводить информацию в стандартный выходной поток и читать данные из стандвх потока, предоставл методы для работы внутри сценария с объектами автоматизации.

**WshShell –** осуществляет запуск новых процессов, позволяет создавать ярлыки, работать с системным реестром, получать доступ к переменным среды и специальным папкам Windows

**Environment(tp)–** представляет доступ к переменным среды.

По умолчанию предоставляется доступ к переменным типа «System» (переменные среды ОС). Возможно использование типов «User» (переменные среды пользователя), «volatile» (временные переменные) и «Process»

**SpecialFolders(str)** – возвращает строку, содержащую путь к специальной папке Windows. Пример специальных папок: desktop– рабочий стол; fonts – каталог со шрифтами; mydocuments– мои документы;startup – автозагрузка и т.п.

**AppActivate(title)**– активизирует заданное параметром title (именем или идентификатором) окно приложения (приложение должно быть запущено)

**Exec(имя\_процесса)** – запускает указанный дочерний процесс

**36. СерверсценариевWSH. ОбъектWshShell (Popup, RegRead, RegWrite, RegDelete, Run).**

Главный объект WSH (WindowsScriptHost) служит для создания других объектов или связи с ними содерж инф о сервере сценариев, а также позвол выводить информацию в стандартный выходной поток и читать данные из стандвх потока, предоставл методы для работы внутри сценария с объектами автоматизации.

**WshShell –** осуществляет запуск новых процессов, позволяет создавать ярлыки, работать с системным реестром, получать доступ к переменным среды и специальным папкам Windows

**Popup(str[,sec][,tit][,tp])** – выводит на экран информацию: окно с сообщением str

**RegRead(имя)** – возвращает значение с параметра реестра или значение по умолчанию для раздела реестра

**RegWrite(имя, значение [тип])** – записывает в реестр значение, заданное. Параметра или значения по умолчанию

**RegDelete (имя)** – удаляет из системного реестра указанные параметры или раздел

**Run (команда, [стиль окна][,метка])** – запуск внешних приложений

**37. Серверсценариев WSH. ОбъектWshShell (SendKeys, SpecialFolders).**

Главный объект WSH (WindowsScriptHost) служит для создания других объектов или связи с ними содерж инф о сервере сценариев, а также позвол выводить информацию в стандартный выходной поток и читать данные из стандвх потока, предоставл методы для работы внутри сценария с объектами автоматизации.

**WshShell –** осуществляет запуск новых процессов, позволяет создавать ярлыки, работать с системным реестром, получать доступ к переменным среды и специальным папкам Windows

**SendKey-**Иммитация нажатия клавиши. Данные будут обрабатываться тем приложением, кот. явл. активным.Символы: +,^,%,~,[ ] заключ в {}. Для задания неотображаемых символов используются коды: {BACKSPASE} <F1> {F1}; {ENTER} <=>; {BREAK} <F16> {F16}; {CAPSLOOK}; {DEL} <> <=> {RIGHT}; {END}; {ESC}; {NOME}; {INSERT}. Для задания комбинации с <SHIFT>, <Ctrl>,<alt> перед соотв. Клавишей необх. Поставить один/неск кодов: SHIFT +, Ctrl ^,alt %.

**SpecialFolders(str)** – возвращает строку, содержащую путь к специальной папке Windows. Пример специальных папок: desktop– рабочий стол; fonts – каталог со шрифтами; mydocuments– мои документы;startup – автозагрузка и т.п.

**38. Серверсценариев WSH. ОбъектWshNetwork (ComputerName, EnumNetworkDrives, SetDefaultPrinter).**

Главный объект WSH (WindowsScriptHost) служит для создания других объектов или связи с ними содерж инф о сервере сценариев, а также позвол выводить информацию в стандартный выходной поток и читать данные из стандвх потока, предоставл методы для работы внутри сценария с объектами автоматизации.

**WshNetwork –** предназначен для работы с ресурсами локальной сети.

**ComputerName –** Содержит имя комп-а, на котором запущен сценарий.

**EnumNetworkDrives** – возвращает коллекцию в которой хранятся буква и сетевые пути ко всем подключенным сетевым дискам.

**SetDefaultPrinter –** делает заданный сетевой принтер принтером по умолч.

**39. Серверсценариев WSH. ОбъектFileSystemObject(Drives, CreateFolder, MoveFolder, FolderExists).**

Главный объект WSH (WindowsScriptHost) служит для создания других объектов или связи с ними содерж инф о сервере сценариев, а также позвол выводить информацию в стандартный выходной поток и читать данные из стандвх потока, предоставл методы для работы внутри сценария с объектами автоматизации.

**FileSystemObject**– используется для создания, копирования, перемещения или удаления файлов или каталогов. Является основным объектом, обеспечивающим доступ к файловой системе компа.

**Drives**– хранит коллекцию содержащую информацию о всех доступных дисках

**CreateFolder(имя\_каталога)** – создаёт новый каталог

**MoveFolder(имя\_каталога, dest)** – перемещает один или несколько каталогов в каталог dest

**FolderExists(имя\_каталога)** – возвращает true, если катало существует и false в противном случае

**40. Серверсценариев WSH. ОбъектFileSystemObject(DriveExists, GetDriveName, CopyFile, FileExists).**

Главный объект WSH (WindowsScriptHost) служит для создания других объектов или связи с ними содерж инф о сервере сценариев, а также позвол выводить информацию в стандартный выходной поток и читать данные из стандвх потока, предоставл методы для работы внутри сценария с объектами автоматизации.

**FileSystemObject**– используется для создания, копирования, перемещения или удаления файлов или каталогов. Является основным объектом, обеспечивающим доступ к файловой системе компа.

**DriveExists(**устройство**) -** возвращ true, если устройство существует и false в обратном случае.

**GetDriveName(**путь**) -** выделяет строку содерж имя диска из заданного пути, в случае ошибки будет возвращена пустая строка.

**CopyFile(**имя\_файла dest[,ovr]**) -** копирует один или несколько файлов в каталог dest. При указании имен ф-лов можно использ подстановочные знаки. Парам ovr показ будет ли произв замена(знач true).

**FileExists(**имя\_файла**) -** воозвр true, если указ файл существует.

**41. Сервер сценариев WSH. Объект FileSystemObject (GetSpecialFolder, GetTempName, CreateTextFile, OpenTextFile)**

 Метод **GetSpecialFolder(n)** – возвращает объект Folder для некоторых специальных папок Windows, заданных чсиловым параметром n.

Метод **GetTempName() –** возвращает (но не создаёт) случайным образом сгенерированное имя файла или каталога, которое может быть использовано для операций требующих наличия временного файла или папки.

Метод **CreateTextFile(имя\_файла[/err[,unicode]])** – создаёт новый текстовый файл и возвращает указывающий на этот файл объект TextStream. Если err = true, то происходит перезапись одноименных файлов. Если Unicode = true, то файл создаётся в формате Unicode, иначе в режиме ASCII.

Метод **OpenTextFile[,mod[,create[,form]]] –** открывает заданный текстовый файл и возвращает объект TextStream для работы с этим файлом.

**42. Сервер сценариев WSH. Объект Drive (AvailableSpace, DriveLetter, DriveType, FileSystem)**

Объект **Drive** позволяет получить доступ к свойствам заданного локального или сетевого диска.

Метод **AvailableSpace -** cодержит количество доступного для пользователя места (в байтах) на диске.

Метод **DriveLetter -** содержит букву, ассоциированную с локальным устройством или сетевым ресурсом. Это свойство доступно только для чтения

Метод **DriveType -** Содержит числовое значение, определяющее тип устройства:

0 — неизвестное устройство;

1 — устройство со сменным носителем;

2 — жесткий диск;

3 — сетевой диск;

4 --- CD-ROM;(CD-R и CD-RW не различаются)

5 — RAM-диск

Метод **FileSystem –** содержит тип файловой системы, использующейся на диске (FAT, NTFS или CDFS).

**43. Сервер сценариев WSH. Объект Drive (FreeSpace, IsReady, TotalSize, VolumeName)**

Метод **FreeSpace** - содержит количество свободного места (в байтах) на локальном диске или сетевом ресурсе. Доступно только для чтения.

Метод **IsReady -** содержит true, если устройство готово, и false — в противном случае. Для устройств со сменными носителями и приво­дов CD-ROM, IsReady возвращает true только в том случае, когда в дисковод вставлен соответствующий носитель и устройство готово предоставить доступ к этому носителю.

Метод **TotalSize -** содержит общий объем в байтах локального диска или сетевого ресурса.
 Метод **VolumeName -** содержит метку тома для диска. Доступно для чтения и записи.

**44. Сервер сценариев WSH. Объект Folder (Attributes, DateCreated, IsRootFolder, Path, Folders)**

Метод **Attributes -** позволяет просмотреть или установить атрибуты каталога.

Виды атрибутов:

* Normal – обычный, без установленных атрибутов (0)
* Readonly – только чтение (1)
* Hidden – скрытый (2)
* System – системный (4)
* Directory – папка (16)
* Archive – архивный (32)
* Alias – ярлык (1024)
* Compressed – сжатый (2048)

 Метод **DateCreated -** содержит дату и время создания каталога. Доступно только для чтения.

 Метод **IsRootFolder -** возвращает True, если каталог является корневым, и False в противном случае.

 Метод **Path -** содержит путь к каталогу.

Метод **Folders -** коллекция. Содержит список всех папок, находящихся в конкретной папке.

**45. Сервер сценариев WSH. Объект File (Attributes, DateCreated, ParentFolder, Size, Files)**

Метод **Attributes -** позволяет просмотреть или установить атрибуты файла.

 Виды атрибутов:

* Normal – обычный, без установленных атрибутов (0)
* Readonly – только чтение (1)
* Hidden – скрытый (2)
* System – системный (4)
* Directory – папка (16)
* Archive – архивный (32)
* Alias – ярлык (1024)
* Compressed – сжатый (2048)

 Метод **DateCreated -** содержит дату и время создания файла. Доступно только для чтения.

 Метод **ParentFolder -** содержит объект File для родительского каталога.

Метод **Size –** возвращает размер файла в байтах.

Метод **Files -** коллекция. Содержит список всех файлов, находящихся в данной папке.

**46. Сервер сценариев WSH. Объект TextStream (AtEndOfLine, AtEndOfStream, Line, Close)**

 **TextStream** обеспечивает последовательный доступ к текстовому файлу. Методы этого объекта позволяют читать информацию из файла и записывать её в него.

Метод **AtEndOfLine –** содержит true, если указатель достиг конца строки в файле и false в противном случае.

 Метод **AtEndOfStream -** содержит true, если указатель достиг конца файла и false в противном случае.

 Метод **Line –** содержит номер текущей строки в текстовом файле.

 Метод **Close –** закрывает открытый файл.

**47. Особенности Windows PowerShell**

При разработке PowerShell были решены **след.задачи**:

1. Обеспечение прямого доступа из командной строки и объектам COM,WME и .NET
2. Организация работы с произвольными источниками данных в командной строке по принципу файловой системы.
3. Разработка интуитивно понятной структуры встроенных команд, основанный на их функциональных назначениях.
4. Обеспечение возможности расширения встроенных набора команд.
5. Организация поддержки знакомых команд из других оболочек.
6. Разработка полноценной встроенной справочной системы для внутренних команд.
7. Реализация автоматического завершения при вводе с клавиатуры имен команд.

Единицей обработки и передачи информации в PowerShell является **объект**,а не строка текста.

**48. Типы команд Windows PowerShell**

1) Командлета

2) Функции

3) Исполняемые формулы

4) Сценарии

Командлет – аналог внутренних команд традиционных оболочек и представляет собой класс .NET, порожденный от базового класса Cmdlet

Использование единого базового класса Cmdlet позволяет исполнять совместимый синтаксис всех командлетов.

Функция – блок кода на языке PS, имеющий заголовок.

<function> <param1> <param2>

Сценарий – блок кода на языке PS, хранящийся во внешнем файле с расширением ps1.

Исполняемые формулы выполняются обычным образом ОС.

**49.** **Формат командлета Windows PowerShell. «Глагольные» части командлетов.**

Имена командлета нечувствительны к регистру символов и соответствуют шаблону «глагол – существительное», где глагол задает действие, а существительное определяет объект, над которым это действие будет совершено.

«Глагольные» части командлета:

1. Add – создает
2. Clear – удаляет
3. ConvertTo – преобразовывает элемент к заданному формату.
4. Disable/Enable – откл./включает ф-ции
5. New – создает экземпляр элемента, такого как переменная или событие.
6. Remote – удаляет экземпляр элемента
7. Set – изменяет заданный параметр объекта
8. Start/Stop – запускает/останавливает
9. Test – проверяет состояние или значения
10. Write – записывает

**50. Windows PowerShell. Параметры командлетов.**

Перед параметрами командлетов всегда стоит дефис.

Имена параметров можно не указывать, если они передаются в заданном порядке. Все командлеты поддерживают общий набор параметров.

**Параметры:**

**-Confirm** приостанавливает использование до получения подтверждения от пользователя.

 Remove-item E:\\*.js –confirm

**-Debug** представляет отладочную информацию об операции на уровне программирования. Параметр работает только с командлетами, которые генерируют сообщения отладки.

**-ErrorAction** определяет реакцию команды на возникновение ошибки.

Допустимые значения:

1. SilentlyContinue (продолжить исполнение, игнорирую ошибку)
2. Continue (вывести сообщение об ошибке и продолжить; задано по умолч.)
3. Inquire (вывести сообщение об ошибке и продолжить после получения подтверждения пользователя)
4. Stop (вывести сообщение об ошибке и прервать исполнение)

**-ErrorVariable** задает имя переменной для хранения сведений об ошибках.

**-OutBuffer** задает буфер для выходных данных командлета.

**-OutVariable** задает имя переменной для хранения выходных объектов команды. Для добавления новых данных перед именем переменной указывают знак «плюс» (+).

**-Verbose** предоставляет подробные сведения о действиях выполняемой команды.

**-WarningAction** определяет реакцию команды на предупреждение.

Допустимые значения:

1. SilentlyContinue
2. Continue
3. Inquire
4. Stop

-**WarningVariable** задает имя переменной для хранения предупреждений.

**-WhatIf** показывает пользователю, каков будет результат запуска командлета с заданным параметром.

**51. Windows PowerShell. Обработка ошибок. Справочная система. Конвейеризация команд.**

**Обработка ошибок**

При работе с командлетом возникает 2 стандартных типа **ошибок**:

> возникает ошибка и сценарий прекращается.

>возник. ошибка, выводится сообщение и продолж. выполен-е сценария.

**Справочная система**

Стандартный справочный документ содержит полный синтаксис командлета.

**Справочн. информацию** можно получить след образом:

> параметр **-?**

> командлет **Get-Help** с параметрами Detailed, Full, Examples.

**Конвейеризация команд**

**Конвейер** – последовательность команд, разделенных между собой знаком |

(вертикальная черта). Команда в конвейере получает объект от предыдущей команды, выполняет операции над ним и передает след. команде, т.к. используется ссылка на объект, а не текст, => на экране могут не отображаться некоторые слова и методы.

 **Get-Member** – список св-в и методов, указанного класса объекта.

Например: Get-ChildItem|Get-Member

Если необходимо знать, напр., только св-ва:

Get-ChildItem|GetMember- MemberTypePropererty.

Допускается фильтрация объектов в конвейере, удаление из конвейера объектов, не удовл. некот. условию.

 **Пример**. Вывести список запрещенных в системе процессов, упорядоченный по уменьшению затраченного процессорного времени. Get-Process|Sort-Objectcpu-Descending.

 **Пример**. Удалить зависшие процессы.

Get-Process|Where-Object{-not$\_.Responding}||Stop-Process.

**$\_** - обращение к каждому переданному в конвейере объекту.

Условие проверки задается в виде блока сценария в {}.

Результатом должно быть true/false.

**52. Windows PowerShell. Форматирование выводимой информации.**

**Format-Table** – форматирует вывод команды в виде таблицы, столбцы которой содержат св-ва объекта. Поддерживается возможность группировки выводимых данных.

**Format-List** – вывод формат. как список св-в , в котором каждое св-во отображается на первой строке. Поддерживается возможность группировки выводимых данных.

**Format-Custom** – для формат. Вывода используется пользовательское представление (view).

**Format-Wide** – форм. Объекты в виде широкой таблицы, в кот. отображ. только одно св-во каждого объекта.

**Пример**. Вывести процессы, начинающиеся на букву “p”.

Get-Process p\*

Get-Process p\*|Format-Table-ProperertyCPU,Name выводит столбцы с именем с загрузкой процессора и именем процесса.

Пример. Вывести все имеющиеся у объектов св-ва можно с помощью параметра \*, например:

Get-Process p\*|Format-Table\*

Get-Process p\*|Format-List

**53. Windows PowerShell. Перенаправление выводимой информации.**

**Командлеты для управления выводом данных** нач. со слова **out**, их список можно увидеть след образом: Get-Command out-\*|Format-Table Name.

По умолчанию выводимая информация передается командлету **Out-Default**, который, в свою очередь, посылает всю работу по выводу строк на экран командлету **Out-Host**.

**Out-File** – направляет вывод в файл, расположенный по заданному пути. Если выходной файл существует, можно перезаписать его (указав параметр – Force) или дописать к нему новые данные (с помощью параметра Append).

**Пример**. Get-Process|out-File e:\process.txt-append.

**Out-GridView** – направляет вывод в интерактивную таблицу, кот. откры-вается в отдельном окне и поддерживает сортировку, копирование и фильтрацию результатов, а так же объединение.

**Out-Host** – направляет вывод в командную строку. Параметр – Paging включает постраничный вывод.

**Out-Null** – направляет вывод на null-порт, удаляет его без отображения.

**Out-Printer** – направляет вывод на принтер по умолчанию, либо на принтер, заданный параметром -Name.

**Out-String** – преобразует все объекты в выводе в единую строку и направляет результат на консоль.

Параметр **–Stream** заставляет направлять строки для каждого из объектов по отдельности.

Параметр **–Width** задает длину строки в знаках (значение по умолчанию – 80 знаков), строки, превышающие заданный предел, усекаются.

**Команда>[путь]файл** – направление вывода в заданный файл. Если файл не существует, то он создается, существующий файл может быть перезаписан или дописан.

**Команда>>[путь]файл** – дописывает вывод к заданному файлу, если файл не существует, то он создается.

**Команда\_err>&1** – направляет сообщение об ошибках туда же, куда и стандартный вывод.

**54. Windows PowerShell. Типы переменных. Командлеты для работы с переменными.**

Переменная в PS начинается со знака $ и может содержать буквы, цифры и символ подчеркивания.

**Типы переменных:**

**Пользовательские** – создаются и поддерживаются пользователем. Определены только для текущего сеанса. Для использования в др. сеансах переменные необходимо добавить в профиль.

**Автоматические** – создаются и поддерживаются средой PS и предназначены для хранения состояния PS(напр, &? Содержит состояние выполнения последней операции (1 или 0). Дополнительные сведения можно получать командой Get-Help about\_Automatic\_Variables

**Привелигерованные** – создаются оболочкой Windows PS и заполняются настройками пользователя, заданными по умолчанию. Дополнительные сведения можно получить командлетом Get-Help about\_Preference-Variables.

**Окружения** – среды хранят сведения о среде ОС. Дополнительные сведения можно получить: Get-Help about\_Environment\_Variables.

**Создание новой переменной: $<имя\_переменной>=<значение>**

**Вывод значения переменной: $<имя\_переменной>**

В переменной можно хранить любые типы объектов, включая целые числа, строки, массивы и хэш-таблицы, объекты, представл. процессы, службы, журналы событий и компьютеры.

**Командлеты для работы с переменными**

Get-Variable – выводит имена и значения всех или только заданных переменных, установленных в текущем сеансе.

Nev-Variable – объявляет новую переменную.

Set-Variable – объявляет новую или изменяет объявление существующей переменной.

Remove-Variable – удаляет переменную и ее значение.

Clear-Variable – сбрасывает значение переменной в NULL.

Если имя переменной содержит специальные символы, то оно помещается в фигурные скобки {}.

Пример ${d/q^&}=333

Для задания переменной указанного типа, перед ее именем указывают имя типа.

Допускается использование любых типов данных Microsoft.NetFramework.

**55. Windows PowerShell. Основные типы данных. Области видимости переменных.**

Основные типы данных: **String, char, bool, int, long, decimal, double, single, DateTime, array, hashtable**.

Значение переменной типа **bool**:

$null -> false

Ссылка на объект -> true

Число, отличное от нуля -> true

Ноль -> false

Непустая строка -> true

Пустая строка - > false

Непустой массив - > true

Пустой массив -> false

Если значение массива присвоить переменную типа bool она примет значение true или false.

**Переменные могут быть:**

-Глобальные/локальные

-Локальными для сценария (script)

-Закрытыми (private)

 Область видимости определена для переменных, псевдонимов, функций и дисков. Все переменные, псевдонимы и функции объявляются в сценарии, существуют только в области этого сценария. Так же и для функций. Элемент, определенный в некоторой области действия, доступен в ней и в дочерней областях действия (кроме private).

При запуске сенса его область действия становится глобальной. Новая область, созданная при запуске сценария, становится дочерней.

**Области видимости** переменной задаются **явно** и **неявно**.

Для явного задания к имени переменной добавляется префикс **Global, Local, Script или Private.**

Например: $Global:aaa=55.

Глобальные области действия разных экземпляров PS не перекрываются.

**56. Windows PowerShell.** **Преобразование типов данных.**

Узнать тип данных можно, используя **имя\_переменной.GetType().**

Если тип данных для переменной определен, то при выполнении операции присваиваемые данные будут приводиться к указанному типу.

При несоответствии типов происходит ошибка.

Для явного преобразования типов используются методы класса [connect]

ToBoolean, ToChar, ToDateTime…

**Пример**: $c=$t.ToString()

**57. Windows PowerShell.** **Операторы сравнения.**

-**eq** – равно -**ne** – не равно

**-lt** – меньше **-Ll** – меньше или равно

**-gt** – больше **-ge** – больше или равно

**-match** – использование регулярных выражений для поиска соответствующему образцу.

**-nomatch** – использование регулярных выражений для несоответствия образцу.

-**like** – сравнение на совпадение с учетом подстановочного знака в тексте

-**notlike** – сравнение на несовпадение с учетом подстановочного знака в тексте.

-**contains** – содержит

**-notcontains** – не содержит

Допустимо использование **логических операторов:**

**and** – и **or** – или **not** – не **!** – не

При использовании операторов сравнения PS не учитывает реестр символов при сравнении.

Для сравнения с учетом регистра перед операцией добавляют букву **с**

**“A”-eq=>true “A”-ceq“a”=>false**

При работе с массивами и наборами операторы сравнения возвращают все элементы массива, соответствующие заданному выражению.

Операторы **–contains** и **–notcontains** используют для поиска в строках, массивах и наборах значений.

Операторы –**like, -notlike** используются для поиска по шаблону.

**Знаки, используемые в регулярных выражениях:**

**“подстрока”** – соответствует подстроке в любой позиции заданной строки

* - соответствует любому символу

**[список]** – соответствует любому символу из списка

**[диапазон**] – соответствует любому значению из диапазона

**[^]** – соответствует любому символу, кроме заданного

**^** - соответствует начальной части строки

**$** - соответствует окончанию строки

**\***- соответствует любой подстроке

**58. Windows PowerShell. Команды для работы со строками.**

В PS начало и конец литеральных строковых выражений помечают одинарными (‘ ‘) или двойными (“ ”) кавычками.

**Length** – возвращает длину строки в символах.

**Contains(str2)** – возвращает true, если строка str2 входит в заданную строку.

**EndsWith(str2)** – возвращает true, если строка оканчивается строкой str2.

**Insert(n,str)** – вставляет строку str в строку $s после n-го символа.

**Replace()** – заменяет все вхождения заданной подстроки другой заданной подстрокой.