Консольные приложения — используют для взаимодействия с пользователем консоль.

Консоль – интерфейс, который использует стандартный ввод и вывод данных.(это текстовый интерфейс, в котором команды отдаются путем ввода текстовых строк с клавиатуры.)

*Интерпретатор командной строки*, или shell (shell -- оболочка) -- эта та программа, которая принимает команды от пользователя, исполняет их, и выводит результат.

Кроме того, shell - это специализированный язык программирования, в котором есть переменные, конструкции while, if, for и т.д., функции и много чего еще.

В основном функционал интерпретатора заключается в предоставлении пользователю возможности запускать другие приложения.

**CMD –** командный интерпретатор для windows (является отдельной программой).

**Bash –** командный интерпретатор unix.

*Поток — это байтовая последовательность, передаваемая в про­цессе ввода-вывода.* (Название произошло от того,что информация вводится и выводится в виде потока байтов – символ за символом.)

*Стандартные потоки — механизм, позволяющий осуществлять взаимодействие с пользователем как чтение и запись в файл.*

*Стандартные потоки назначаются и открываются системой автоматически.*

stdin (0)— поток стандартного ввода (обычно связан с клавиатурой);

stdout (1) — поток стандартного вывода (обычно связан с дисплеем);

stderr (2) — вывод сообщений об ошибках (связан с диспле­ем).

- Сложны для пользователя (отсутствует интерфейс)

+ Возможность вывод одного приложения перенаправить в ввод другого приложения

+ Нересурсоёмкие

+ Автоматизация рутинных действий

Конвеер – инструмент операционной системы, позволяющий перенаправлять поток вывода одного консольного приложения в ввод другого (или в файл).

цепочка консольных приложений, так называемые скрипты, где перенаправлен вывод из одного приложения в ввод другого.

Результат выполнения можно выводить не только на экран.

Для перенаправления вывода одной программы на ввод другой используется знак “|”.

Самый простой пример:

**dir | sort**

Операционная система позволяет:

* Выводить сообщения программ не на экран (стандартный выходной поток), а в файл или на принтер (перенаправление вывода);
* читать входные данные не с клавиатуры (стандартный входной поток), а из заранее подготовленного файла (перенаправление ввода);
* передавать сообщения, выводимые одной программой, в качестве входных данных для другой программы (конвейеризация или композиция команд).

Знаки:

* **команда > имя\_файла** перенаправить вывод команды в текстовый файл. Если файл существовал, то он перезаписывается. Иначе создается новый файл.
* **Команда >> имя\_файла** дописывать в конец файла, а не перезаписывать.
* **Команда < имя\_файла** прочитать входные данные для команды из файла.
* Для группировки команд и повышения приоритета используются скобки **(команда)**
* Так же можно использовать фигурные скобки **{команда}.** Выражение в круглых скобках выполняется в отдельном потоке, а в фигурных в том же.
* **2>** номер дескриптора который нужно перенаправить (по умолчанию 1).
* **&>** перенаправить оба потока (вывода и ошибок)

Существуют также стандартные устройства:

* **CON –** консоль
* **NUL –** специальное нулевое устройство (чтобы вывод некуда не записывался)
* **PRN -** принтер

Скрипты:

Консольных программ масса, и они используются часто для того, чтобы упростить рутинные действия пользователя. Дело в том, что с ними можно обращаться точно так же, как и с обычными командами. А те, в свою очередь, можно записать в специальный текстовый файл с расширением BAT или CMD (такой файл называется командным), и их можно потом выполнить все сразу как обычную программу - достаточно запустить командный файл.

Для получения справочной информации используется команда **help**. На экран выведется список основных команд.

Для того, чтобы получить справочную информацию об определенной команде можно использовать **command /?**.

В консольные приложения можно передавать параметры, которые делятся на 2 вида:

* Опции. Необязательные параметры. Могут указываться в любом порядке. Обозначаются либо одним тире **-o** короткий вариант, либо двумя **- -option** полный вариант.
* Аргументы. Параметры, которые необходимо указать, для работы программы. Их порядок строго определен.
* Агрументы в угловых скобках **<>** являются позиционными, их нужно обязательно указывать в установленном порядке.

my\_program <host> <port>

* Элементы в квадратных скобках являются необязательными (опциональными). Их можно либо использовать, либо нет. Могут быть как аргументами, так и опциями.
* Иногда обязательные элементы заключают в круглые скобки. Например, чтобы сгруппировать взаимоисключающие элементы:

my\_program (--either-this <and-that> | <or-this>)

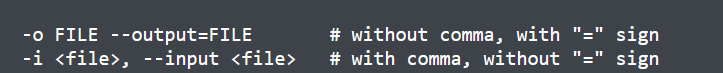
Или обозначить, что если есть один элемент, то должен быть и другой.

my\_program [(<one-argument> <another-argument>)]

* Вертикальная черта (или), указывает на взаимоисключающие элементы.

my\_program go (--up | --down | --left | --right)

* Троеточие … указывает на то, что может быть несколько аргументов с таким именем
* Чтобы указать, что опция имеет аргумент, поместите слово, описывающее этот аргумент, после пробела (или знака равенства "=")

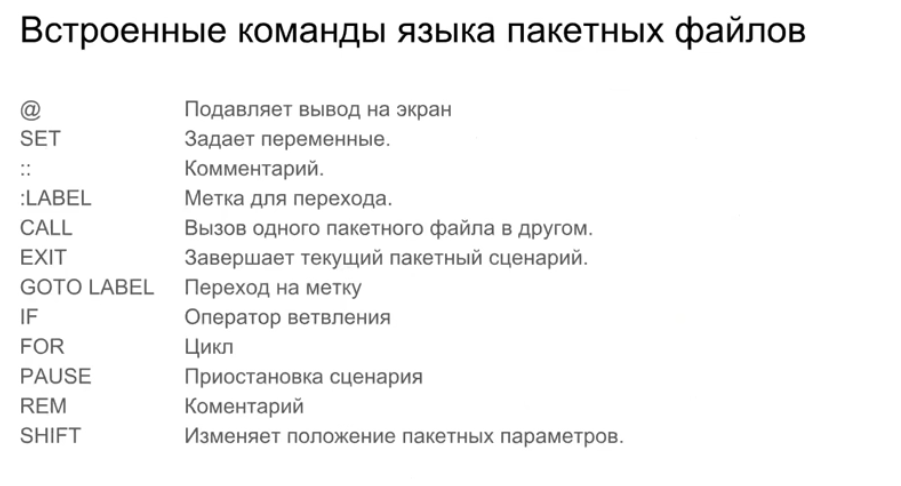


**Разница между консольной командой и приложением** состоит в том, что команды вшиты в консоль, а приложения хранятся отдельным исполняемым файлом. При вызове чего-то в cmd, оно сначала ищется в консольных командах, затем в текущей папке, затем в PATH (%Windir%/System32).

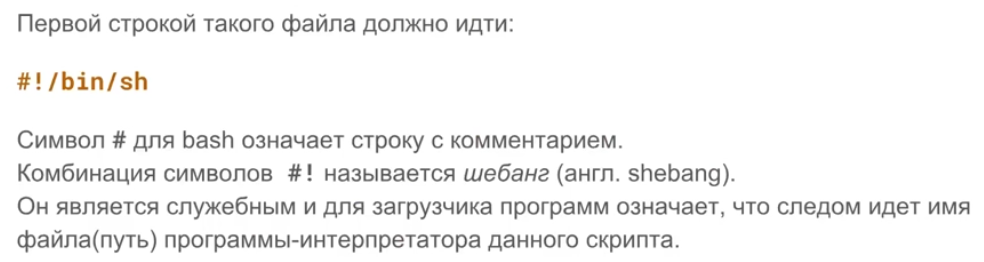
**Встроенные:**

* **MD –** создать каталог.
* **RD –** удалить каталог.
* **CD –** смена текущего каталога.
* **DIR –** просмотр содержимого каталога.
* **MOVE –** перемещение\переименование каталога.

**Пакетные файлы –** текстовый файл, имеющий расширение bat или cmd в windows (В linux можно запускать файлы с любым расширением, но обычно им дают расширение sh), и содержащий последовательность команд, выполняемых интерпретатором.



В Linux



**POSIX –** Portable Operating System Interface – набор стандартов, описывающих интерфейс взаимодействия с операционной системой. Создан для обеспечения совместимости различных систем (В основном unix) и переносимости прикладных программ на уровне исходного кода.

**Переменные окружения –** текстовые переменные операционной системы, хранящие в себе какие-то данные. Данные хранятся в виде имя=значение.