Работа с консолью

LoftSchool
от мыслителя к создателю

Содержание

1

Терминал

Командная строка, консоль, терминал — в чём разница? В первую очередь давайте разбираться с терминами.

Командная строка (консоль) — это программная оболочка, позволяющая в текстовом виде давать компьютеру различные команды, обеспечивающая прямую связь между пользователем и операционной системой. Команды состоят из букв, цифр, символов, набираются построчно, выполняются после нажатия клавиши **Enter**. Командная строка встроена в ядро системы и будет доступна, даже если графический интерфейс не запустится.

Терминал - графическая программа, эмулирующая консоль, оболочка для командной строки, позволяющая выполнять команды, не выходя из графического режима. Терминал по сравнению с консолью имеет дополнительный функционал (различные настройки, вкладки, возможность запускать много окон, управление мышью в некоторых программах, контекстное меню, главное меню, полоса прокрутки).

Шелл (shell) - часть ОС, часто называемая интерпретатором. Шелл – это специальный интерфейс, созданный для обеспечения взаимодействия между пользователем и ядром, т. е. это программное окружение, обеспечивающее необходимые условия для запуска приложений. Можно выделить два типа шеллов: **графический** (например, Windows Explorer, Finder в Mac OS X) и **текстовый** (например, **bash, sh, tsh, csh, zsh**).

Эти три понятия часто используются, заменяя друг друга. Разница между ними сегодня становится все более и более размытой.

bash - это один из самых известных шеллов. Эту программу часто называют «Bourne again shell» («возрождённый» шелл) в честь Steven Bourn, который разработал классический **shell**. Одна из главных особенностей шеллов: все команды, как правило, являются небольшими программами, которые легко найти на своем диске. **Bash** – это тоже программа, расположенная

в каталоге **/bin/bash**. Особенно популярна в среде Linux, где она часто используется в качестве предустановленной командной оболочки. А также по умолчанию установлена в Mac OS X.

Зачем вообще нужно использовать командную строку?

Когда-то привычного всем графического интерфейса в операционных системах не было и всё делалось как раз таки в командной строке.

Дело в том, что некоторые вещи можно быстрее выполнить именно в командной строке, а некоторые настройки в принципе отсутствуют в графическом интерфейсе пользователя. Так же следует иметь ввиду, что до сих пор существуют утилиты, не имеющие графического интерфейса, а иногда он оказывается недоступен, например, из-за сбоя. Здесь и используется командная строка.

Работа с командной строкой — не такая страшная задача, как вы могли бы подумать. Чтобы использовать командную строку, не требуется специальных знаний, это такая же программа, как и все остальные. Только вот создана для того, чтобы выполнять текстовые команды, поэтому отложите свою мышку в сторону и подвиньте поближе клавиатуру.

Многие программы и утилиты, с которыми мы будем работать (например, Gulp), имеют графические оболочки (англ. graphical user interface, GUI). Однако, надо понимать, что GUI работает также через консоль (это скрыто за красивой картинкой). GUI всегда посредник между консолью и вашей программой или утилитой, что подразумевает наличие багов и отсутствие всех возможностей, которые нам предоставляет консоль.

Умение работать в консоли обязательно в командной работе над крупными проектами!

2

Работа в терминале. Mac OS

Как Mac OS, так и Linux относятся к семейству UNIX-подобных ОС, поэтому информация в этой главе будет справедлива и для Linux-систем.

Отличной альтернативой встроенному терминалу является iTerm2 - симпатичная и более продвинутая оболочка для вашего терминала.

Чтобы запустить терминал, нажмите **Ctrl +** пробел и начните вводить «Терминал».

Командная строка начинается с названия компьютера (в примере это Macintosh), затем следует название текущего каталога — по умолчанию открывается домашний каталог пользователя, который в UNIX-системах обозначается знаком **~** (тильда).

В любой момент времени работы в терминале вы находитесь в некотором каталоге. При запуске терминала текущей директорией является домашний каталог пользователя.

Текущий каталог — это то, что написано между символами **:** и **$**.

Далее следует имя пользователя, а за ним знак **$**, который называется «приглашением» — приглашением интерпретатору вводить команды.

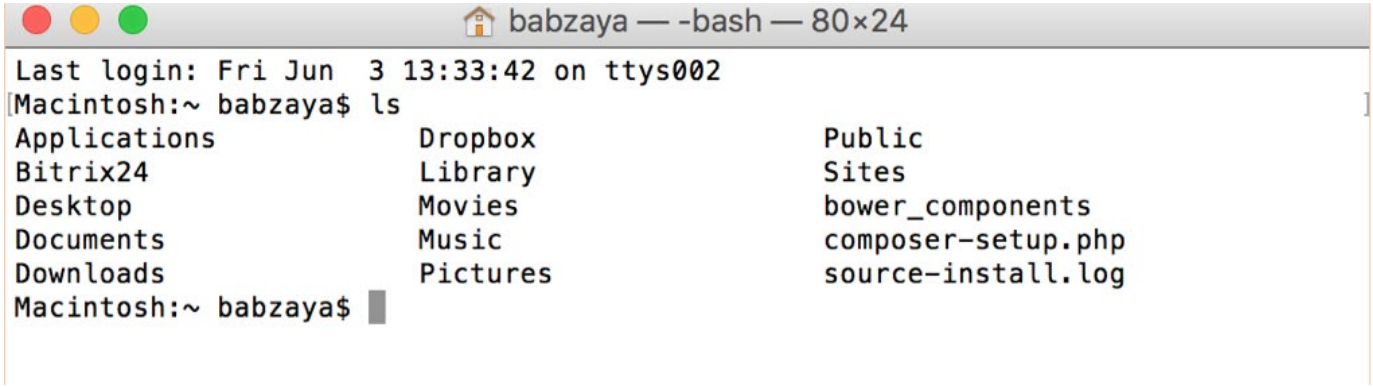
Вид командной строки и приглашения можно настраивать.

Именно после знака **$** и вводятся все команды интерпретатору.

В этом месте находится курсор — мигающий прямоугольник (кстати, его вид тоже можно настраивать).

Основные команды:

Для просмотра каталогов и файлов текущей директории используйте команду **ls** (LiSt – список).



Важно знать, что команда ls показывает не все файлы, например, скрытые файлы она не покажет.

Для того, чтобы вывести весь список файлов (в том числе скрытых) команду **ls** необходимо вводить с ключом **–a**

Вот так:

```
ls –a
```

Ключи (аргументы) можно комбинировать, например, если мы введём **ls –la**, отобразится содержимое каталога вместе со скрытыми файлами (аргумент **a**), а также полный вывод информации (аргумент **l**)

Команда **pwd** покажет абсолютный путь до текущей директории.

cd — изменяет текущую рабочую директорию. Например, если вы находитесь в домашнем каталоге пользователя и введете **cd Downloads**, то перейдете в папку **Downloads**, а если вы хотите, наоборот, выйти из каталога на уровень выше, то для этого можно воспользоваться командой

```
cd ../
```

Список терминальных команд Terminal mac OS X от A до Z

Важно: Есть небольшая особенность в работе с терминалом в UNIX-подобных системах. Если вам нужно изменить что-то в настройках системы, или установить глобальные пакеты (об этом поговорим позже), то перед такими командами необходимо вводить специальную команду **sudo**.

Она позволяет запускать последующие команды от имени администратора.

Запуская команды от имени обычного пользователя, вы получите ошибку. Увидев её, не пугайтесь, просто допишите **sudo** перед вводом команды.

Не все команды требуют прав администратора, а только те, которые могут нанести вред системе, либо изменить системные настройки, каталоги и т.д.

Это пригодится нам для установки глобальных npm-утилит, а также в некоторых ситуациях — для смены прав каталога, например:

```
sudo npm install -g gulp
```

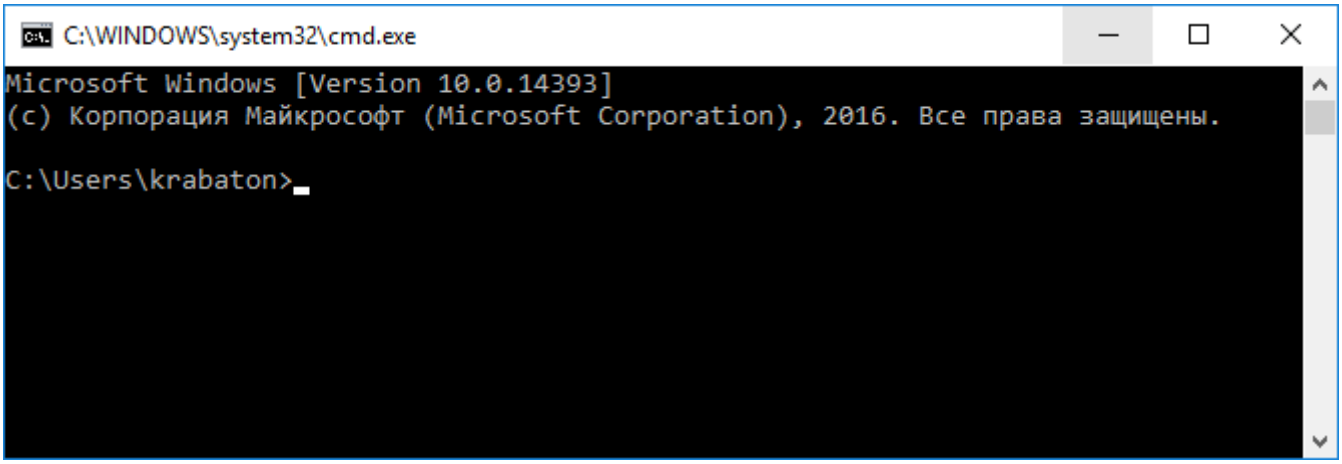
3

Командная строка и эмуляторы консоли в Windows

Программа cmd.exe позволяет получить доступ к командной строке Microsoft Windows. Это интерпретатор команд, который загружает приложения и направляет поток данных между приложениями, для перевода введённой команды в понятный системе вид.

Перейдём в меню Пуск и запустим его :

- **Пуск** → Все программы → Служебные-Windows → Командная строка.
- Или используем сочетание клавиш **Win + R** — **cmd.exe**



Мы видим имя пользователя и мигающее нижнее подчеркивание, показывающее, где будет выводиться набранная вами команда. Как видите, по умолчанию фон командной строки черный, а текст - белый. Но внешний вид командной строки можно настроить под себя. Для этого кликаем правой кнопкой мыши по адресу cmd вверху, и выбираем в меню «**Свойства**».

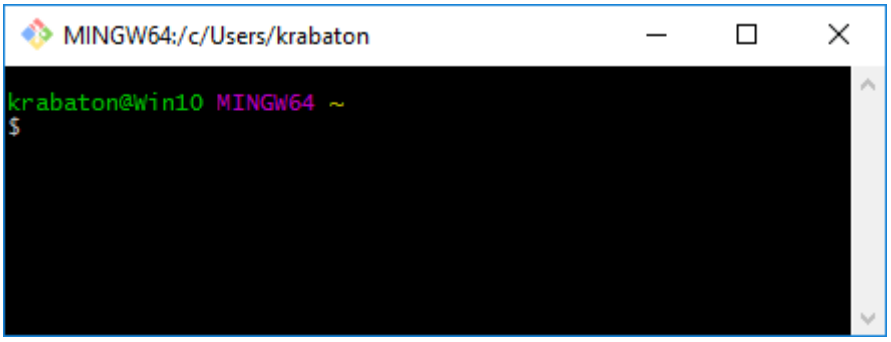
В **cmd.exe** Windows команды частично совпадают с командами в unix-системах.

Cmd команды в полном объеме

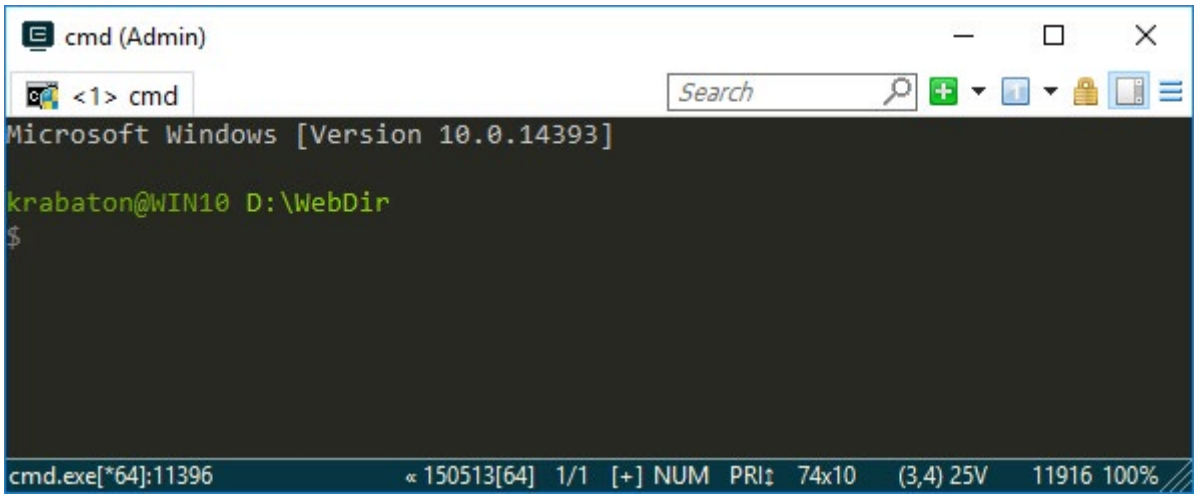
Как и в случае со стандартным терминалом в Mac OS X, мы рекомендуем вам поставить более продвинутую командную оболочку, например, **Git bash** (в котором будут работать все UNIX-команды) или **Conemu**.

Git bash - консольная утилита, которая дополнительно устанавливается непосредственно при установке git. Она представляет собой bash-эмуляцию, что позволяет работать со всеми UNIX- командами.

git bash автоматически установится в систему вместе с Git, на рабочем столе появится ярлык, теперь вы можете запускать его и пользоваться всеми UNIX-командами.



ConEmu — ещё одна оболочка командной строки, но, в отличие от **bash,** здесь есть возможность использовать различные оболочки и утилиты в разных вкладках, например, **cmd, powershell, dn, putty, notepad**.



Основные возможности **Conemu**:

В табах могут отображаться не только заголовки консолей, но и дополнительная информация вроде активного процесса, прогресса архивации, **chkdsk, powershell**, копирования в Far Manager. Например,

не нужно переключаться в таб, чтобы узнать, закончилась ли компиляция проекта, запущенная в этом табе. В статусной строке можно настроить список отображаемых «колонок», вроде координат видимой области и курсора, PID активного процесса в консоли, статусов CAPS/NUM/SCRL, коэффициента прозрачности и др. Многие колонки кликабельны, например, можно щелкнуть по «колонке» с прозрачностью для быстрого ее изменения.

Интерфейс самого терминала содержит всего два дополнительных графических элемента — табы и статусная строка. Но и их можно отключить, если вы предпочитаете «чистую» консоль.

Пользователь может настроить любое количество предопределённых задач (**Task**) для быстрого запуска приложений в ConEmu или из списка переходов (**jump list**) панели задач. Задача может запускать один или несколько процессов или шеллов (**powershell, SDK, компиляция проектов** и т.д.) Можно даже запускать простые **GUI** приложения вроде **PuTTY, TaskManager, GVim**.

Несколько предопределённых палитр (например, **Solarized, PowerShell, xterm**, и др.), возможность настройки своих цветов в консоли, поддержка управляющих кодов ANSI X3.64, 24-битный цвет при работе в Far Manager.

Умеет добавлять себя (и выбранные команды-шеллы) в контекстное меню Windows Explorer. Умеет перехватывать создание стандартного терминала Windows.

Более подробно о работе в программе рассказывает следующая статья.

Документация

Видеоурок от нашей школы.

4

Советы при работе в терминале

1. Используйте автодополнение ввода. Можно набрать только первые буквы команды и нажать клавишу **Tab** — и недостающие буквы команды будут автоматически добавлены. Если же существует несколько команд, начинающихся с тех же символов, которые вы ввели, то двойное нажатие Tab выведет все эти команды в качестве подсказки.

2. Также используйте автодополнение для имён и путей к файлам и директориям. Работает аналогично автодополнению команд.

3. Если в командной строке нажать клавишу вверх ↑, то будет выведена последняя введённая вами команда. Нажимая дальше клавишу вверх ↑, вы будете перемещаться по истории выполненных ранее команд. Полная история хранится в файле **~/.bash_history**.

4. Если ввести два восклицательных знака **!!** и нажать ввод, то вы выполните последнюю введённую команду. Также есть шорткат (**shortcut** - сочетание клавиш) и для использования аргумента от предыдущей команды, для этого надо ввести **имя_команды !$** и нажать **ввод** — вместо **!$** будет подставлен аргумент от предыдущей команды.

5. Если вы что-то напутали при вводе команд, то попробуйте нажать **Ctrl+C**, это сочетание прекращает выполнение текущей команды и закрывает её.

6. Можно прочитать руководство к любой команде и узнать, что она делает, какие у неё есть параметры и аргументы. Для этого надо набрать **man имя_команды**.

7. Для очистки окна терминала можно воспользоваться клавишами **Cmd + K (Ctrl + K)**

Обязательно просмотрите курс по Основам терминала

5

Сетевой протокол прикладного уровня SSH

SSH — это набор программ, которые позволяют регистрироваться на компьютере по сети, удалённо выполнять на нём команды, а также копировать и перемещать файлы между компьютерами. **SSH** организует защищённое безопасное соединение поверх небезопасных каналов связи.

SSH предоставляет замену традиционным r-командам удалённого доступа с тем отличием, что они обладают повышенной безопасностью. Они выполняются поверх защищенных зашифрованных соединений, которые не позволяют прослушивать или подменять трафик. Кроме того, **SSH** может обеспечивать безопасное соединение для передачи любого другого трафика: например, почтовых сообщений или файлов.

Безопасность протокола достигается использованием нескольких решений, которые сводят к минимуму риск использования соединения:

- Шифрование соединения, которое может выполняться одним из методов, выбранных в процессе переговоров. Шифрованное соединение не позволяет просто перехватить и использовать трафик. Выбор алгоритма шифрования делает систему более гибкой, позволяя не использовать алгоритмы, в которых обнаружены слабые места или которые не может поддерживать одна из сторон.

- Аутентификация сервера выполняется при любом соединении. Это не позволяет выполнить подмену сервера или подмену трафика.

- Аутентификация клиента может выполняться одним из нескольких доступных способов. Это, с одной стороны, может повысить надежность аутентификации, с другой — делает систему более гибкой и упрощает ее использование.

- Проверка целостности пакетов позволяет отследить любые незаконные изменения в трафике соединения. При обнаружении таких изменений соединение немедленно разрывается.

- Временные параметры аутентификации не позволяют воспользоваться данными соединения в том случае, если спустя некоторое время после перехвата оно все-таки было расшифровано. Устаревание обычно происходит через час.

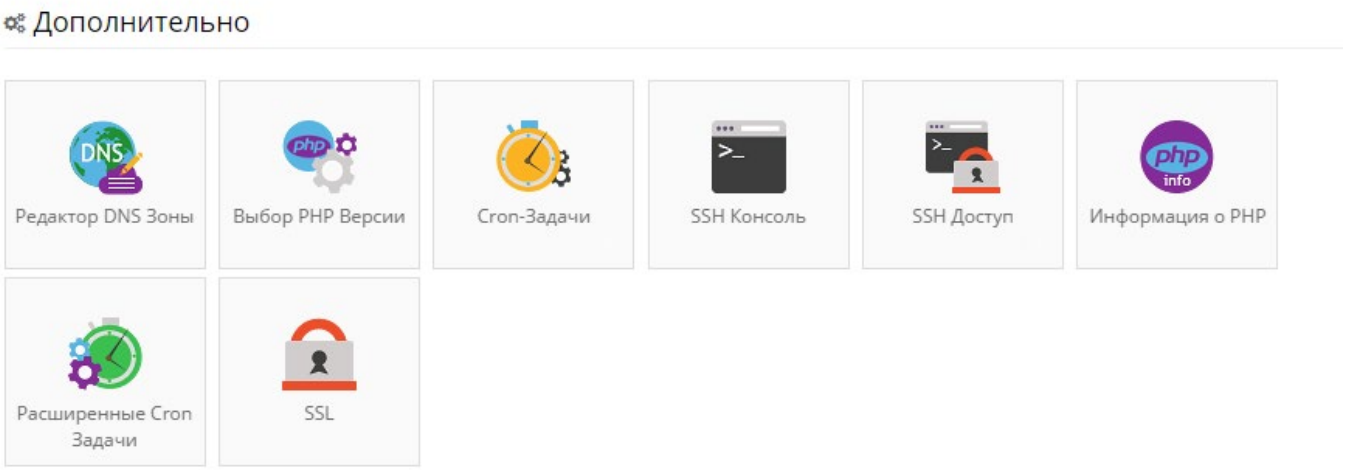
6

Как подключиться к хостингу по SSH

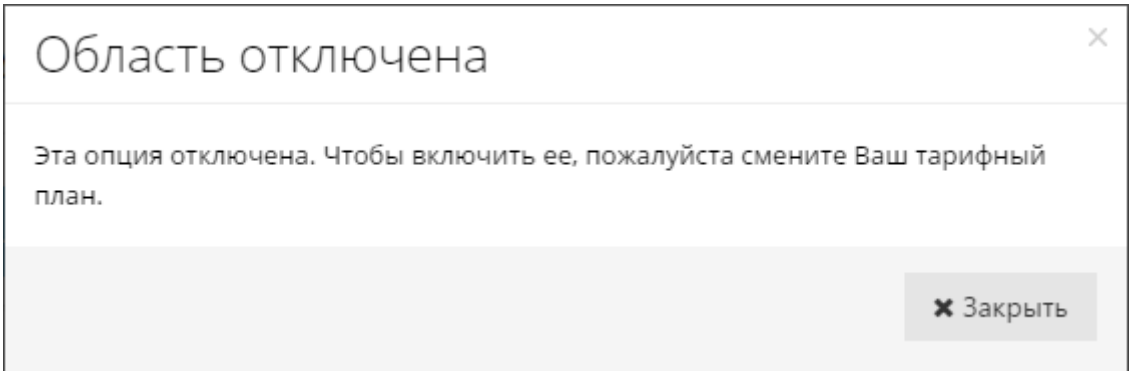
Для операционной системы Windows. ***PuTTy*** - это популярный SSH- и Telnet-клиент (***Telnet*** - тот же SSH, только без шифрованной передачи данных (пакетов)), т. е. программа для безопасного подключения к удалённому компьютеру (или к серверу) и выполнения на нем различных команд. **PuTTY** ведет логи, позволяет настраивать шрифты, цвета и разрешение консоли, допускает сохранение в своей памяти ключей авторизации, поддерживает работу через прокси-сервер. При этом утилита является бесплатной в распространении.

Для того, чтобы начать работу с **PuTTy**, скачайте её с официального сайта

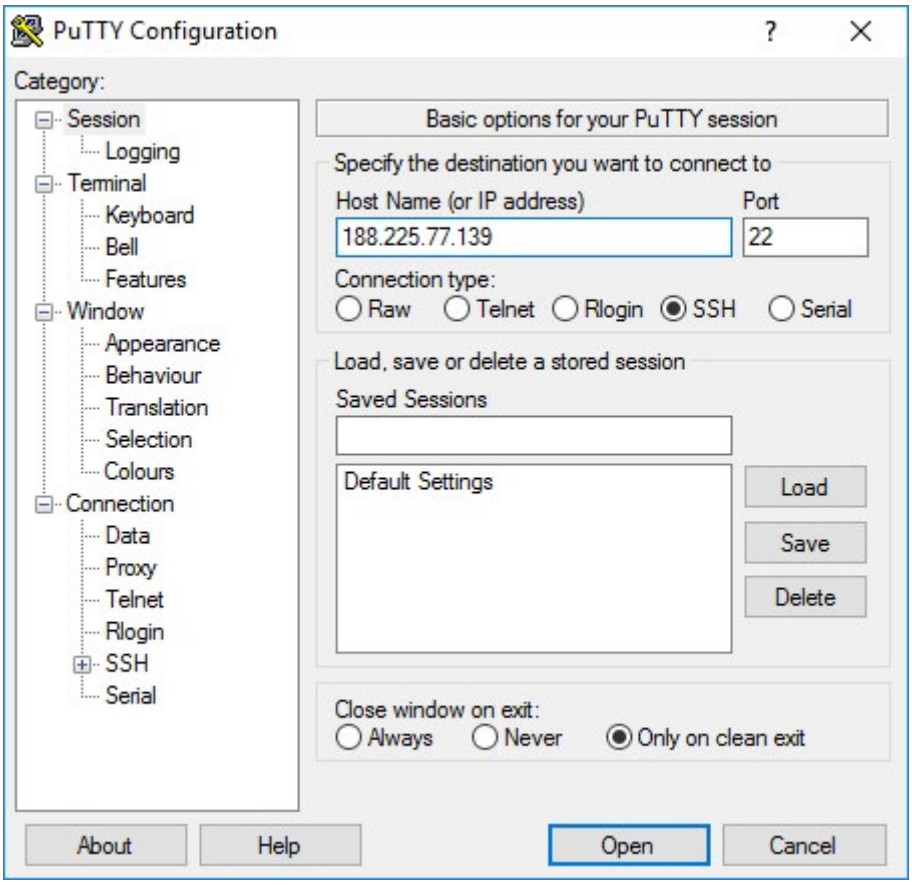
Перед тем, как начать работу с вашим аккаунтом по SSH, Вам необходимо включить SSH у себя на хостинге в Панели управления, в соответствующем разделе, всё зависит от вашего хостингера. Также там можно узнать имя сервера для подключения.



Возможно, ваш тариф не поддерживает SSH и вам нужен будет улучшенный тариф:



Для начала работы запустите файл **`putty.exe`**. Перед Вами появится окно, представленное на рисунке ниже.



В поле **Host Name (or IP address)** вводите имя сервера или его **ip**, которые Вы узнали в разделе по SSH. Порт оставляйте по умолчанию **22**. В поле **Saved Sessions** введите любое имя сессии (коннекта), например **`my_session`**, и нажмите **Save**. После этого нажмите **Open** и Вы увидите черное окно. В поле **login as** введите имя Вашего пользователя, нажмите **Enter**. После чего появится надпись **Password**. Вводите Ваш пароль для доступа по SSH. Не пугайтесь — во время ввода пароля на экране ничего не отображается (ни звёздочек, ни чего-либо подобного). После того, как Вы закончили вводить пароль, нажмите **Enter**.

Если логин и пароль введены верно, то произойдёт подключение к серверу и Вы попадете в командную оболочку Linux.

Также заметим, что сочетание **Ctrl+V** и **Ctrl+C** в **PuTTy** не работают. В буфер обмена копируётся всё, что выделено с помощью мыши, а вставка осуществляется либо правой кнопкой мыши, либо сочетанием клавиш **SHIFT+INSERT**.

Пользователи операционных систем Mac OS X или Linux могут использовать стандартное приложение terminal для подключения к виртуальному серверу по SSH-протоколу. Для подключения к Вашему виртуальному серверу используйте следующую команду (измените **188.127.236.62** на **IP** адрес вашего виртуального сервера):

```
ssh root@188.127.236.62
```

Как видите, это всё делается прямо с терминала самой операционной системы.

7

Полезные команды терминала

Рассказать о всех командах Unix будет сложно, поэтому напишем лишь несколько полезных команд:

man [имя команды]	выдаст подробную информацию по команде, например: **man mv** Для выхода из **man**, т. е. из руководства по команде, нажмите **q** (Quit - Выход). **[имя команды] --help** — также позволит посмотреть описание команды.
ls	вывести список файлов
ls -la	покажет все файлы (включая скрытые), размер файлов, владельца и группу владельца, права на них, дату последнего изменения
ls -lha	то же, что предыдущая команда, только размер файлов будет показан в удобном виде
ls -lha | less	позволит просматривать файлы постранично (если их много)
cd [имя директории]	переход в выбранную директорию
cd ../	переход на директорию выше
cd ~	переход в корневую директорию
mv	переименовать и/или переместить

mv index.html old	перемещение файла в папку
mv index.html old/new_name.txt	перемещение файла в папку с переименованием файла
mv order.txt orderNew.txt	переименовать файл
rm	удалить
rm ghost.png	удалить файл
rm –rf old	удалить папку и всё из неё
cp	копировать
cp index.html catalog.html	копирование файла **index.html** в тот же каталог с переименованием в **catalog.html**
cp index.html old/	копирование файла **index.html** в папку **old/** (всё произойдет в текущей папке)
cp temp/ temp2/ –r	копирование каталога
>	очистка файла. Например, можно применить к файлам логов (**> access.log, > error.log, > combined.log**);
chmod	установка прав на файл или директорию
cat	объединяет файл или несколько файлов, либо ввод со стандартного устройства ввода и выводит результат на стандартное устройство вывода
cat [имя файла]	выведет на экран содержимое файла

cat [имя файла] | grep [искомая строка]	выведет на экран строки файла, включающие искомую строку
mkdir [имя директории]	создание директории (папки)
mkdir project	создать папку с именем «project»
mkdir project project/css project/js	создать несколько папок
mkdir –p project/{css,js}	то же, что выше
touch index.html	создать файл
touch index.html css/ style.css js/script.js	создать файлы (папки **css/ и js/** уже должны существовать)
find . –iname '*ind*'	найти в текущей папке (и подпапках) все файлы, имена которых содержат **ind** и показать списком