

The logo consists of a red square with the text 'NEWPROLAB' in white, stacked vertically. The 'O' in 'PRO' is replaced by a circle with a diagonal slash. The logo is surrounded by several horizontal bars in green, blue, and orange.

NEW
PRO
LAB

ОСНОВЫ LINUX

Николай Марков
@enchantner

NEWPROLAB.COM

А ЧТО ТАКОЕ UNIX/*nix?

- UNIX - система, написанная в 70-х годах в Bell Labs
- Являлась собственностью компании AT&T
- Поддерживалось несколько уровней вложенных каталогов
- Первая реализация представления устройств как char/block device'ов
- Уже содержала концепцию конвейеров ("|")
- Для переноса на другую платформу надо просто переписать драйвера
- BSD - Berkeley Software Distribution - был создан в Беркли на основе системы UNIX

<http://bit.ly/2mHxuT3>

КАК НАСЧЕТ GNU?

- Это не UNIX
- Основатель проекта - Ричард Столлман
- "Давайте все перепишем и сделаем бесплатным"
- free software -> open source software
- Лицензия GPL
- Основная идея - гораздо более быстрое развитие

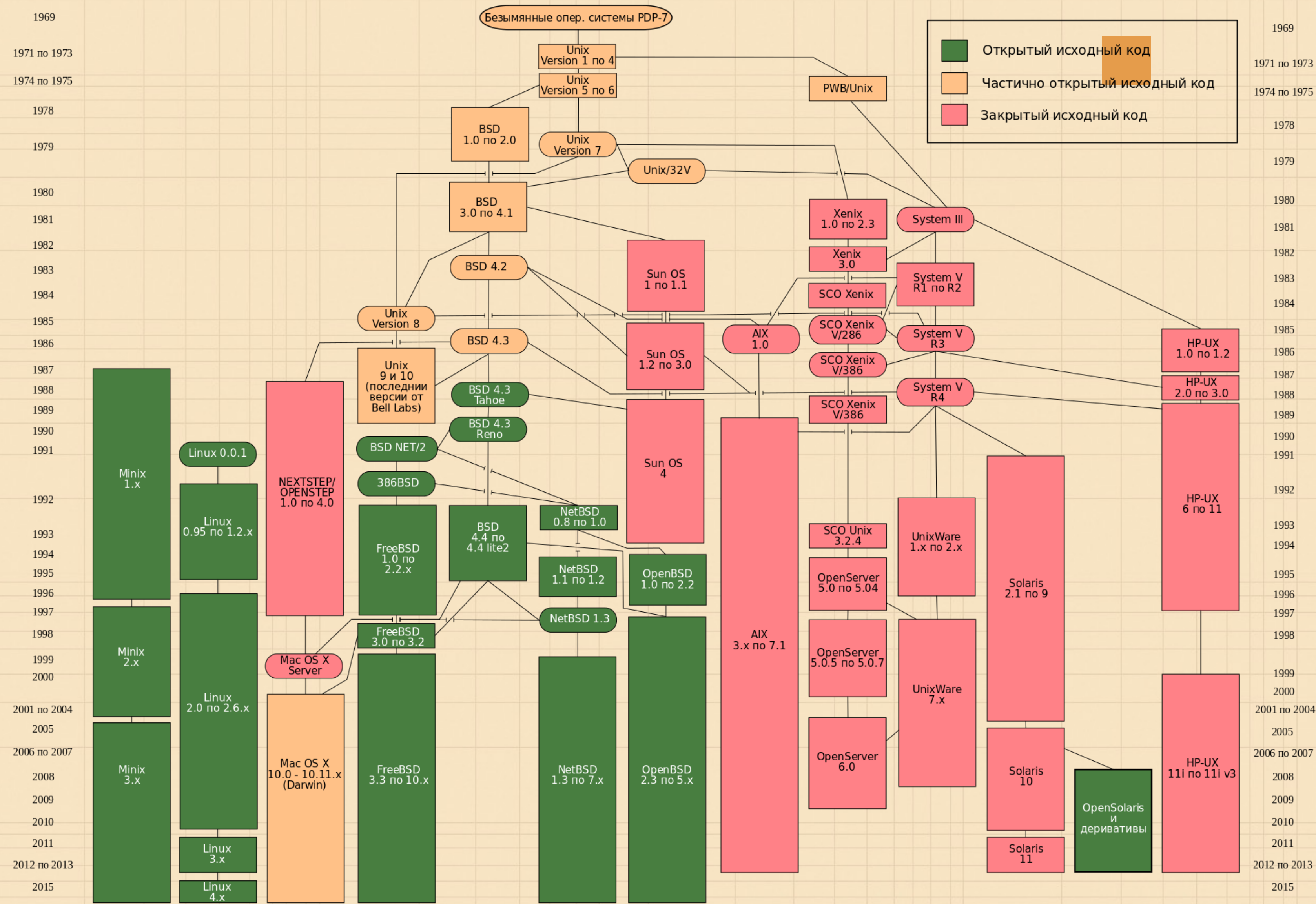


<https://www.gnu.org/>

КАК ЗАРАБАТЫВАЮТ

- Подпишись на поддержку
- Позолоти ручку (сделай пожертвование)
- Заплати фрилансеру за патч
- Купи футболку
- Если ты вендор - плати роялти

ТЫСЯЧИ ИХ

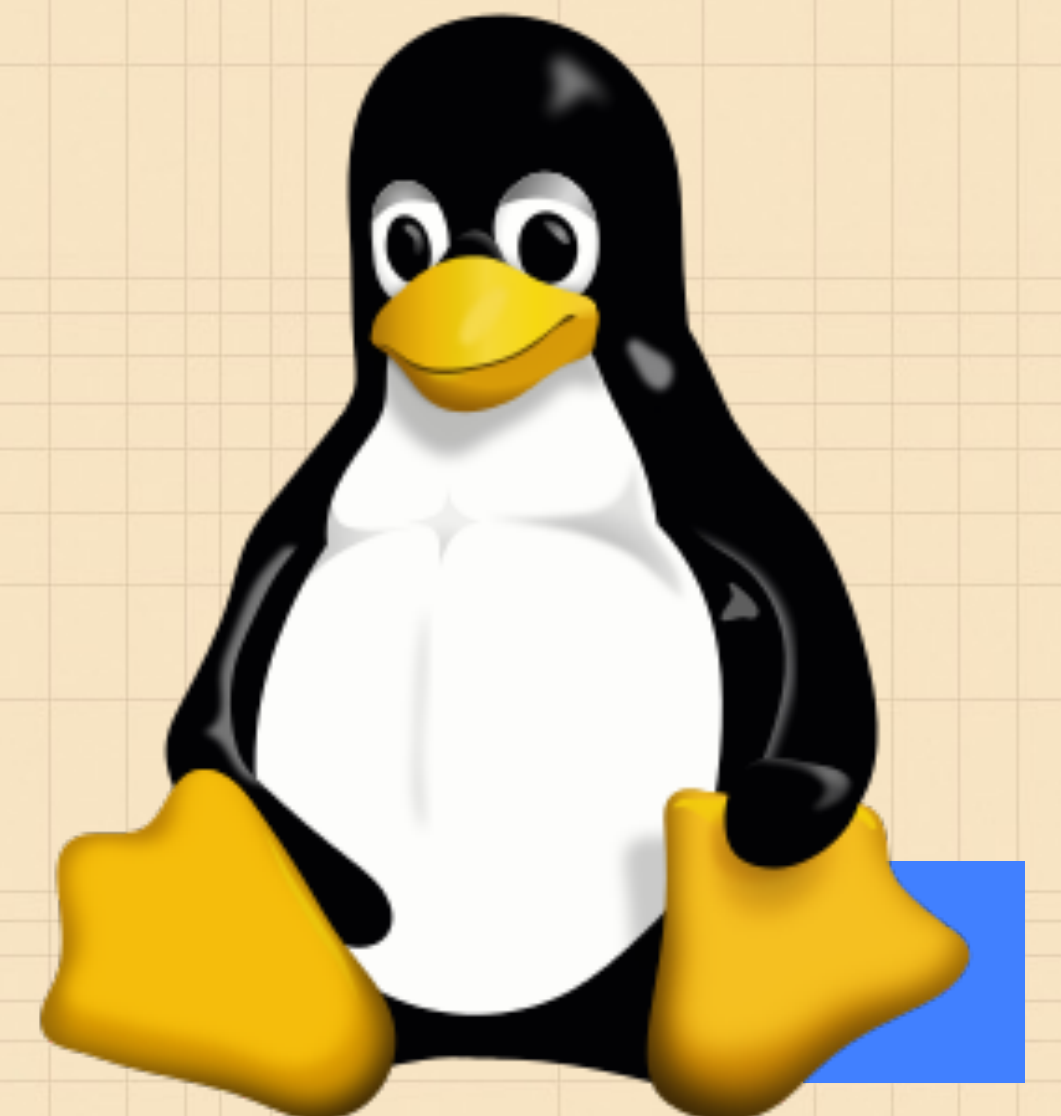


ЧУТЬ БОЛЬШЕ ПРО ЛИЦЕНЗИИ

- LGPL - разрешает распространение, запрещает модификацию
- BSD - разрешает распространять код, если лицензия BSD хотя бы упомянута, для рекламы нужно письменное согласие всех соавторов кода
- MIT - очень похожа на BSD, но не включает ограничений на рекламу
- Mozilla - редко встречается, довольно сложные правила касательно патентов и патчей
- WTFPL - делай, что хочешь

И ГДЕ ТУТ ВАШ ЛИНУКС?

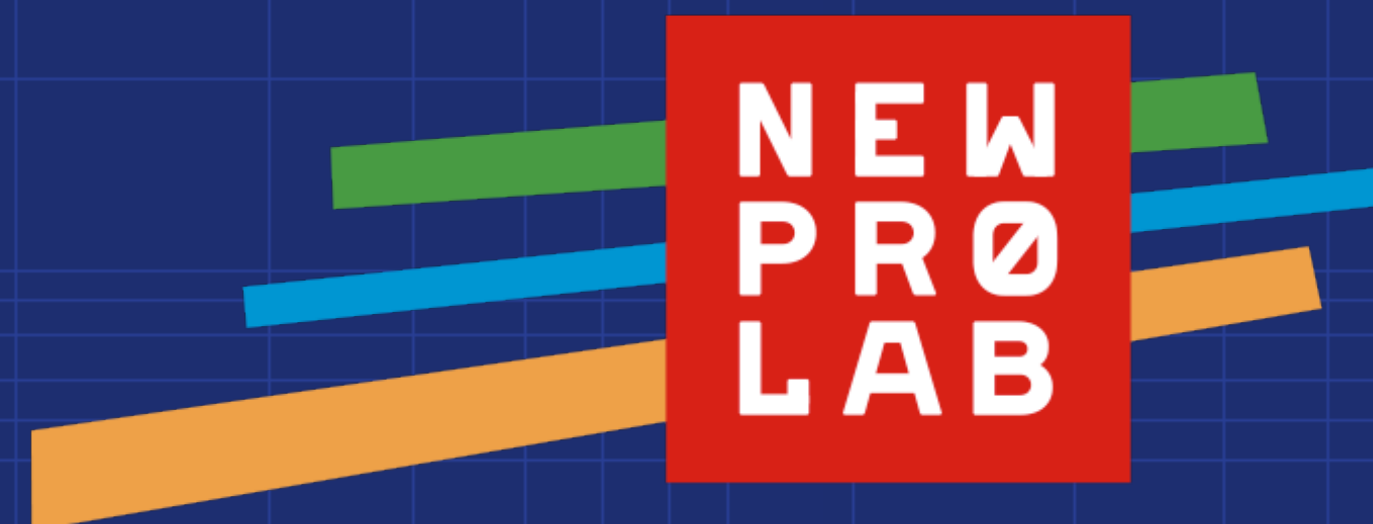
- Появился в 1991 году, как клон Minix - еще одного клона UNIX
- Распространялся под лицензией GPL и отлично сочетался с набором программ GNU
- Старался следовать разным стандартам - например, POSIX
- Холивар Таненбаума - Торвальдса



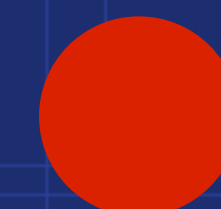
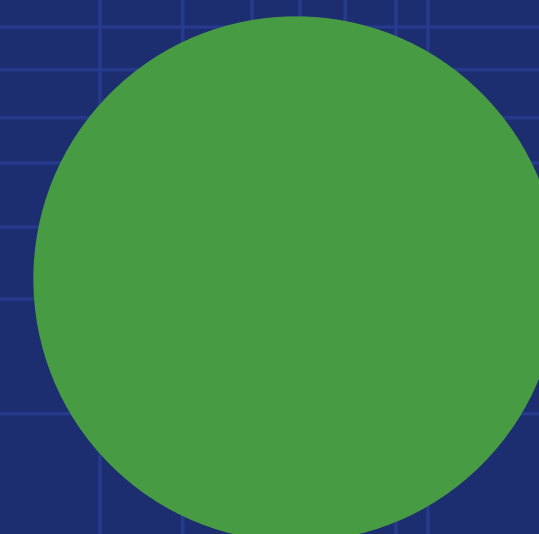
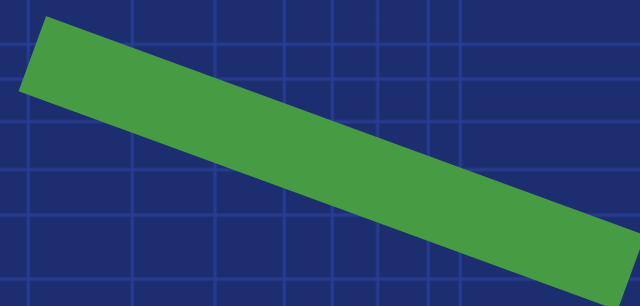
<http://bit.ly/IKpp7ER>

ЧТО ЭТО ВООБЩЕ ТАКОЕ

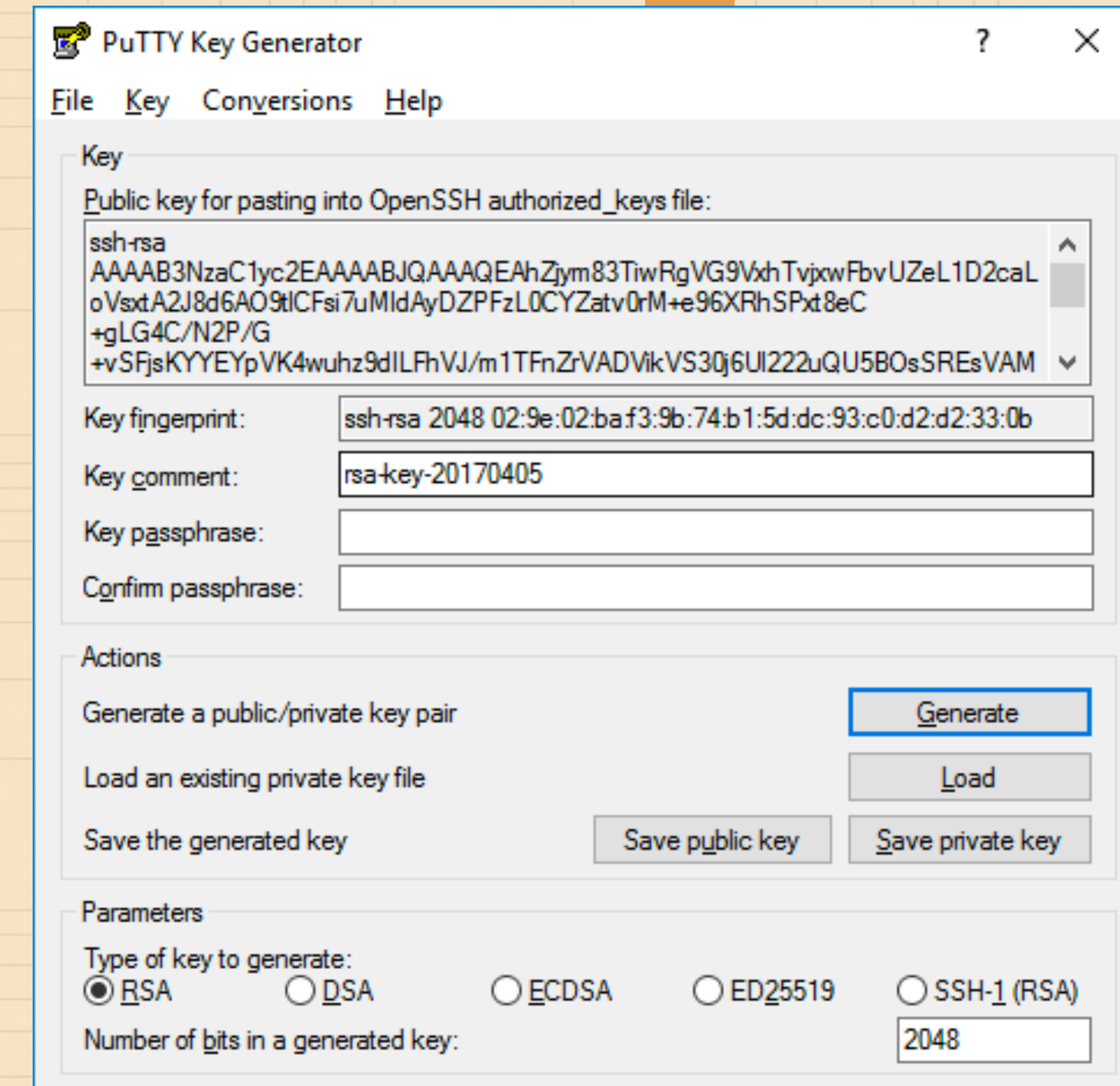
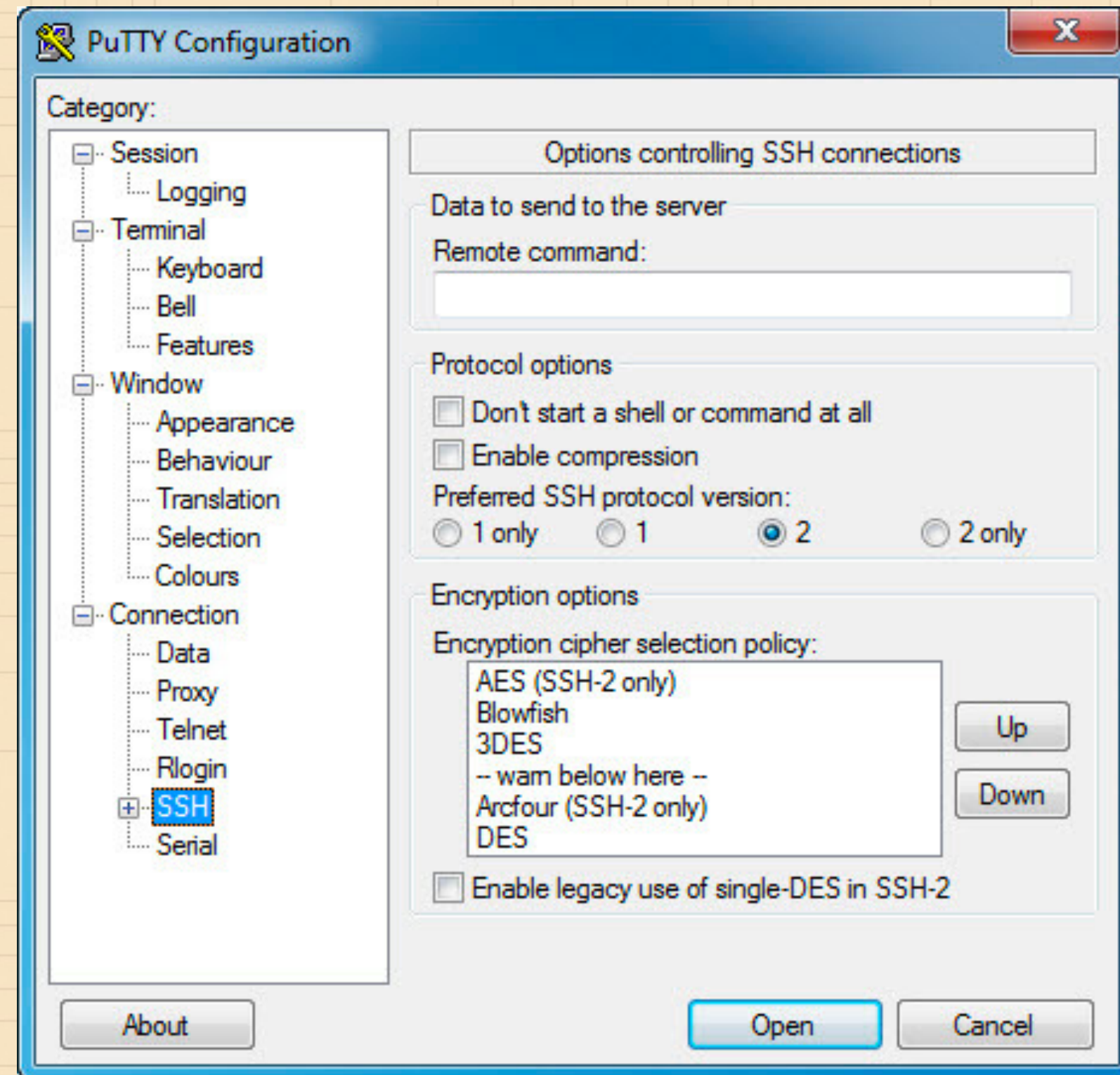
- Не система, а ядро
- Написано на чистом C - переносимость
- Встречается гораздо чаще, чем вы думаете
- Поставляется в виде различных дистрибутивов (Ubuntu, Fedora, Arch)
- Уже несколько лет пригоден для настольного компьютера
- Хорошая поддержка оборудования
- Безопасность на высоком уровне
- Поддерживает большое количество файловых систем



Подключаемся к машине



ПОЛЬЗУЕМСЯ SSH



<http://meinit.nl/using-your-openssh-private-key-in-putty>

Translation -> UTF-8

<https://www.putty.org/>

ПОЛЬЗУЕМСЯ SSH

подключаемся к серверу

~\$ *ssh username@server*

подключаемся к серверу с ключом

~\$ *ssh -i /path/to/key.pem username@server*

копируем публичный ключ на сервер, чтобы

подключаться без пароля

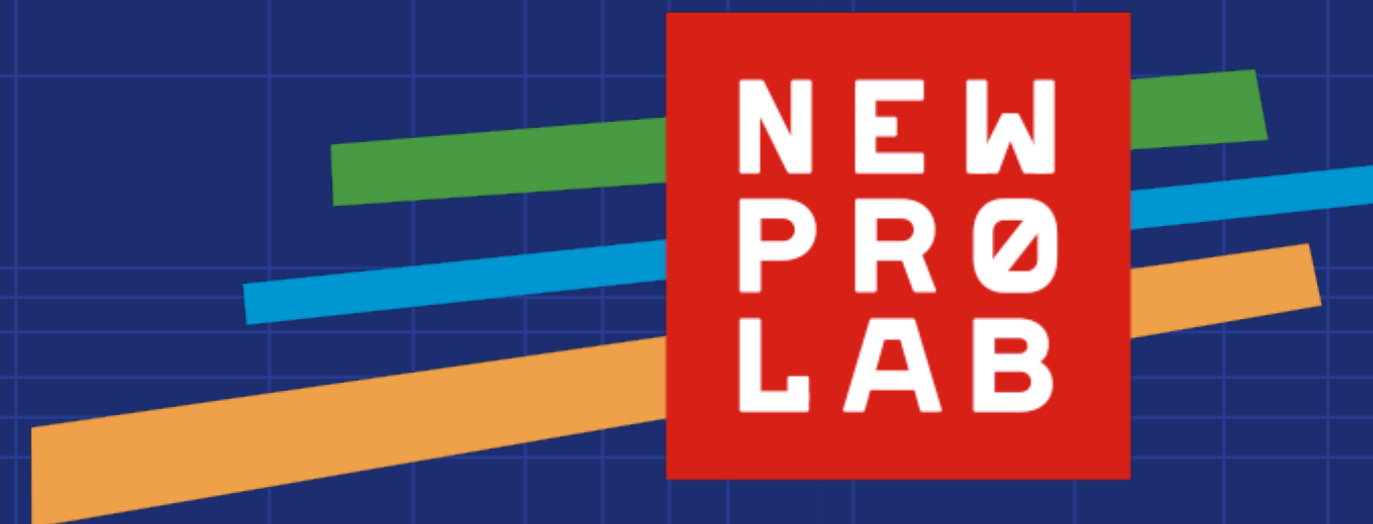
~\$ *ssh-copy-id username@server*

копируем файл на сервер в домашний каталог

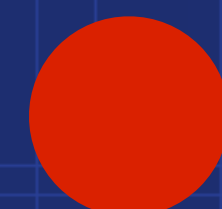
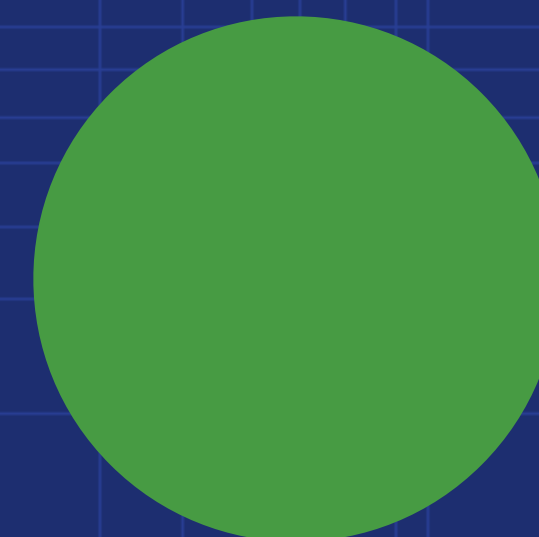
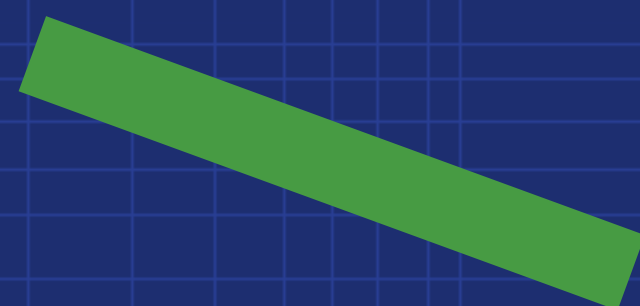
~\$ *scp ./local_file.txt username@server:/home/user*

копируем каталог рекурсивно с сервера в текущий


~\$ *scp -r username@server:/home/user/remote-dir .*



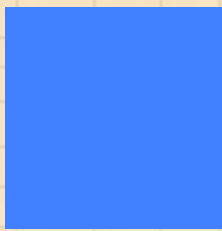



Работаем с файловой системой



ПОЧЕМУ КОМАНДНАЯ СТРОКА?

- Она есть на любой *nix/BSD системе 
- Для винды есть Cygwin + Win10 активно учат bash
- Любые операции элементарно автоматизируются написанием скриптов
- Большинство кода УЖЕ написано за нас
- Огромное количество программ имеют CLI либо версию, работающую в окне терминала (rtorrent, midnight commander, мессенджеры, почтовые клиенты, архиваторы, браузеры)

САМЫЕ ЧАСТО ИСПОЛЬЗУЕМЫЕ КОМАНДЫ



nano man scp wget/curl
cat wc -l tar -xvf
git echo sort chmod +x cd
history htop ps aux ssh
mkdir ls -la less pwd
vim grep find
head/tail which

ВЫВОД СПРАВКИ

~\$ man

~\$ man ls

- проматывать стрелками и PgUp/PgDown,
- выход кнопкой "q", перейти в конец - "G"
- поиск по '/', 'n'/'N' для перехода по результатам

~\$ which python # какая команда запустится?

<http://explainshell.com/>

ОСМАТРИВАЕМСЯ

- ~\$ *pwd* # где мы?
- ~\$ *ls* # смотрим, что у нас вообще есть в текущем каталоге
- ~\$ *ls -la* # смотрим подробности о том, что у нас есть + скрытое
- ~\$ *ls -lah* # размеры файлов в “человекочитаемом” виде
- ~\$ *ls -la /path/to/dir* # смотрим содержимое конкретного каталога
- ~\$ *cd /foo/bar* # перейти в другой каталог (по полному пути)
- ~\$ *cd ./bar/zomg* # перейти в другой каталог (по относительному пути)

. - текущий каталог



* - шаблон, он же wildcard

.. - каталог уровнем выше

~ - домашний каталог пользователя

Если в пути есть пробелы - возьмите его в кавычки

ИСТОРИЯ КОМАНД



Стрелки вверх-вниз работают!

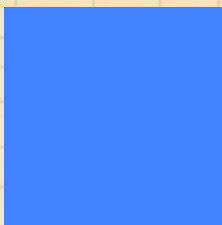

~\$ *history* # вывести историю команд

~\$ *!1337* # еще раз вызвать команду из истории под номером 1337

~\$ *!!* # еще раз запустить предыдущую команду

Быстрый поиск по истории:

Ctrl+R и начинаем набирать. Можно еще раз нажать Ctrl+R и листать дальше в прошлое. Для отмены - нажать Ctrl+C.



ДАВАЙТЕ СОЗДАДИМ ЧТО-НИБУДЬ, НАКОНЕЦ!

~\$ *echo "Hello, World\!"* # выводим строку, спецсимволы экранируем

~\$ *echo -e "1\t2\n3\t4\n"* # выводит строку со спецсимволами
'>' перенаправляет вывод команды в файл,
перезаписывая его с нуля (!)

~\$ *echo "here is my cool string" > file.txt*

'>>' дописывают вывод команды в конец файла

~\$ *echo "another cool string" >> file.txt*

~\$ *mkdir my_dir* # создать пустой каталог (-p для создания пути)

~\$ *cat file.txt* # выводим содержимое файла

~\$ *less file.txt* # читаем файл постранично в просмотрщике
(аналогично man)

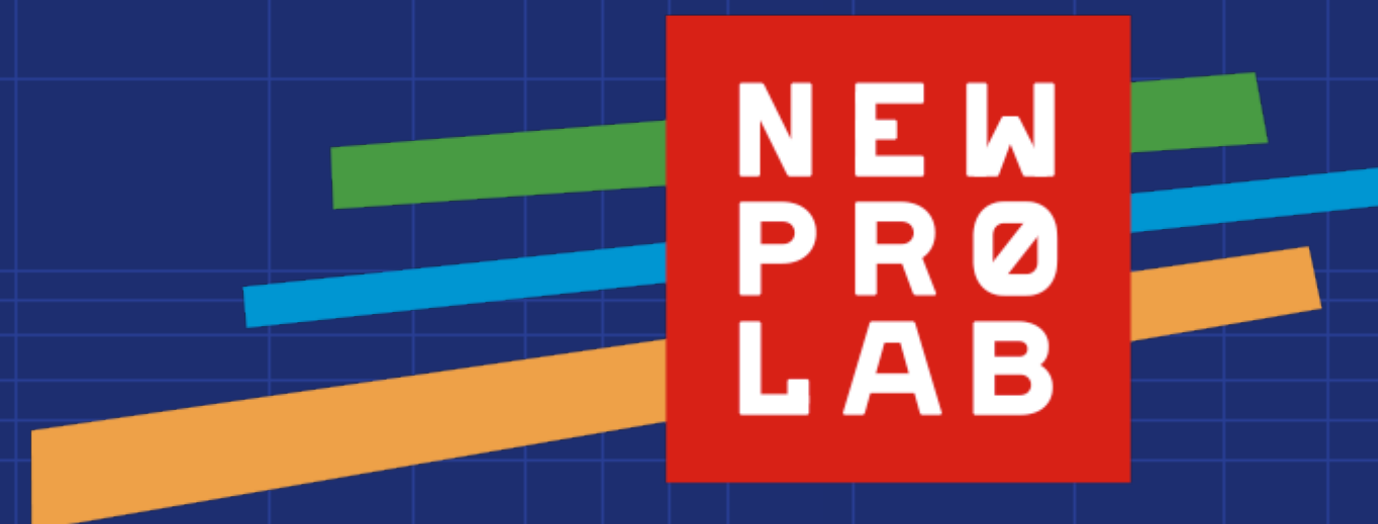
КОПИРУЕМ/ПЕРЕМЕСТИМ

скопировать (-r для каталогов)
~\$ *cp /path/to/old/file /path/to/new/file*
переместить (старый удалится)
~\$ *mv /path/to/old/file /path/to/new/file*

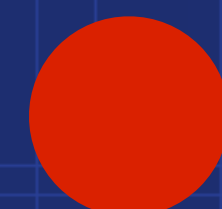
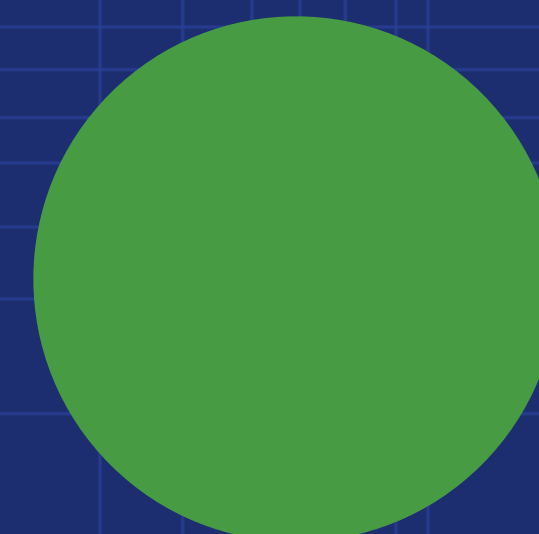
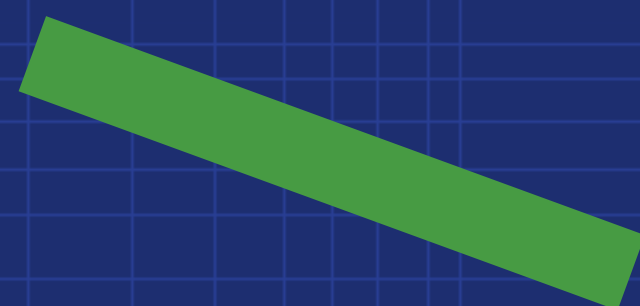
А ТЕПЕРЬ УДАЛИМ, А ТО ЗАЧЕМ ОНО НАДО

удалить файл
~\$ *rm /path/to/file.txt*
удалить каталог рекурсивно
~\$ *rm -r /path/to/dir*
удалить каталог рекурсивно и без лишних вопросов
~\$ *rm -rf /path/to/dir*



размер файла
~\$ *du -h file.txt*



Скачиваем и загружаем



ЧИТАЕМ ДАННЫЕ ИЗ WEB API



скачать и вывести на stdout, не сохраняя

~\$ *curl -s <https://www.gutenberg.org/files/76/76-0.txt>*

скачать и сохранить в текущей папке

~\$ *wget <https://www.gutenberg.org/files/76/76-0.txt>*

разные типы запросов (может понадобиться при работе с REST)


~\$ *curl [-XGET|-XPOST|-XPUT|-XHEAD]*




Старый датасет: <http://bit.ly/1RErAg6>



GITHUB

 This repository Search Pull requests Issues Marketplace Explore

 newprolab / content_bigdata8 Private

Unwatch 14 Star 4 Fork 3

<> Code

Issues 0

Pull requests 0

Projects 0

Wiki

Insights

<http://newprolab.com>

bigdata

data-science

8 commits

1 branch

0 releases

2 contributors

Branch: master


New pull request

Create new file

Upload files

Find file

Clone or download

 datamove fixed repo url in README Latest commit 8b59d6d 9 hours ago

labs/lab01	lab1 fix 7->8	15 hours ago
materials	first day 3	21 hours ago
README.md	fixed repo url in README	9 hours ago

README.md

20 марта 2018

Привет и добро пожаловать в учебное пространство! По мере прохождения курса, в этом репозитории будут появляться все необходимые материалы:

https://github.com/newprolab/content_bigdata8

СИСТЕМА КОНТРОЛЯ ВЕРСИЙ

скачать к себе репозиторий

~\$ *git clone* https://github.com/newprolab/content_bigdata6.git

~\$ *git pull* # обновить локальную копию

~\$ *git diff* # посмотреть текущие изменения до фиксации

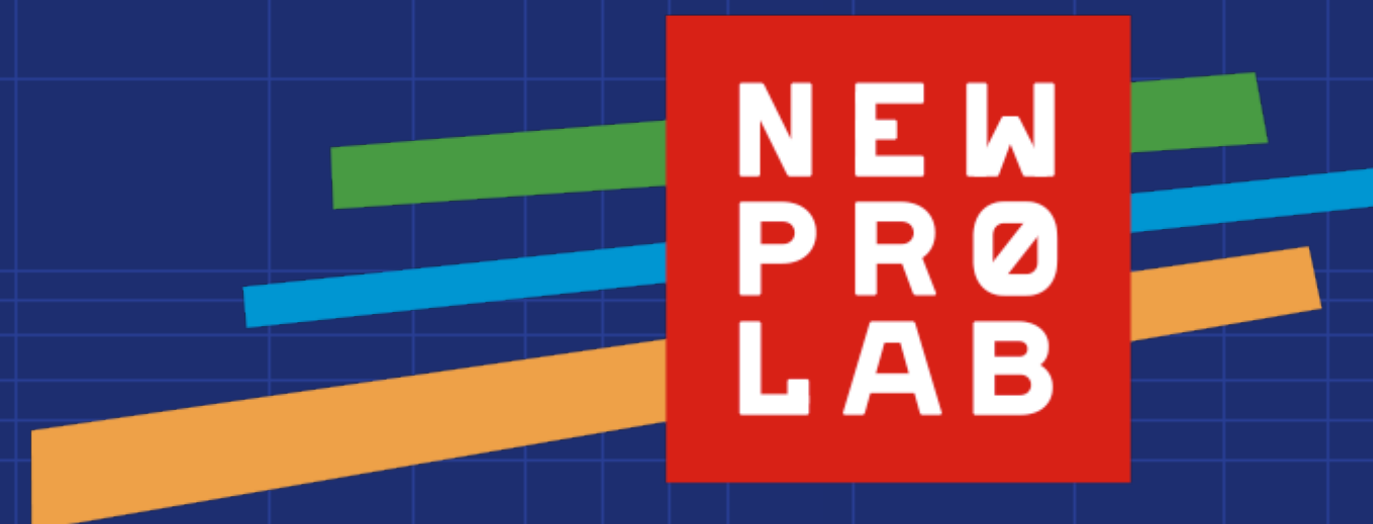
~\$ *git log* # посмотреть историю зафиксированных изменений

~\$ *git add file.txt* # добавить файл в контроль версий

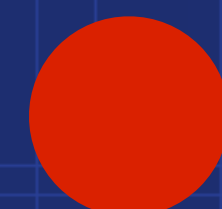
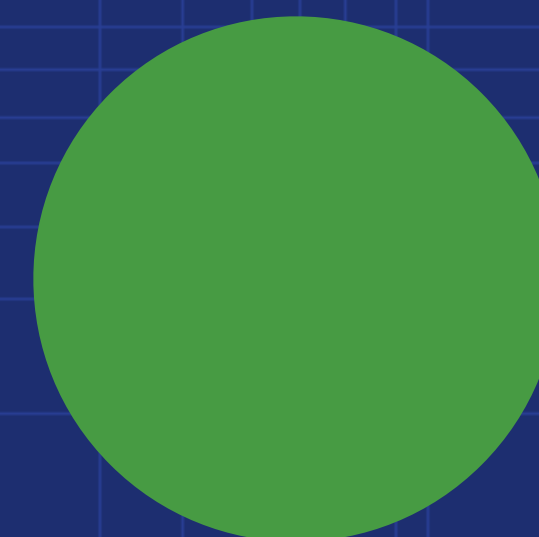
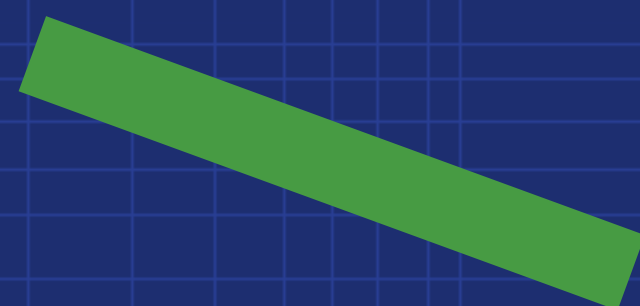
~\$ *git commit -am "some message"* # зафиксировать изменения

~\$ *git push* # залить изменения в репозиторий

.gitignore - файлы, которые не надо никуда заливать



Отлично, работаем дальше!



РЕДАКТИРУЕМ ФАЙЛ

~\$ *nano file.txt* # открываем файл в упрощенном редакторе nano

Редактируем стрелками и клавиатурой
Ctrl+X - выход, Ctrl+O - сохранить

~\$ *vim file.txt* # открываем файл в усложненном редакторе vim

Два основных режима “i” - редактирование, Esc - выход обратно в обычный

:w - сохранить, :q - выйти, :q! - выйти без вопросов, / - поиск
ZZ - сохранить и выйти, ZQ - выйти, не сохраняя

РАБОТА С АРХИВАМИ



- ~\$ *zcat file.tar.gz* # то же, что *cat*, только для архивов
- ~\$ *gunzip -c file.tar.gz* # то же самое, но более универсально
- ~\$ *tar -xvf file.tar.gz* # распаковать архив
- ~\$ *tar -zcvf archive.tar.gz dir-name* # засовываем каталог в архив

Существуют также *unzip* и *unrar*

РАБОТАЕМ С СОДЕРЖИМЫМ ФАЙЛОВ

~\$ *head -n 10 file.txt* # вывести первые 10 строчек **из** файла
~\$ *tail -n 5 file.txt* # вывести последние 5 строчек из файла
~\$ *tail -n+100 file.txt* # вывести все строчки с сотой до конца
~\$ *tail -F /var/log/myapp.log* # “следовать” за новыми строками
~\$ *cat file.txt | head -n 3* # файл подается на стандартный ввод

~\$ *cat file.txt | wc -l* # считаем число строк в файле
сортируем строчки на входе и выводим количества уникальных
~\$ *cat file.txt | sort | uniq -c*

~\$ *cat file.txt | grep Vasya* # выводим только строки, соотв. шаблону
~\$ *cat file.txt | grep -v Vasya* # НЕ соотв. шаблону

ИЩЕМ

ищем рекурсивно все питоновские файлы в текущем каталоге

~\$ *find . -name "*.py"*

ищем нерекурсивно только каталоги в корне файловой системы

~\$ *find / -maxdepth 1 -type d*

ищем все файлы с байткодом Python и тут же удаляем (осторожно!)

~\$ *find ~/Projects/myproject -type f -name "*.pyc" -delete*

Команда поиска по индексу (есть не везде!)

~\$ *locate something* # ищем по строке в названии

~\$ *updatedb* # обновляем индекс

РЕДАКТОРЫ SED/AWK

~\$ *cat file.txt | sed 's/First/Second/'* # потоковая замена
~\$ *cat file.txt | sed 's/First/Second/g'* # жадная потоковая замена
~\$ *cat file.txt | sed 's:/one/path:/another/path:g'* # другой разделитель
~\$ *sed -r/-E ...* # расширенные регулярные выражения

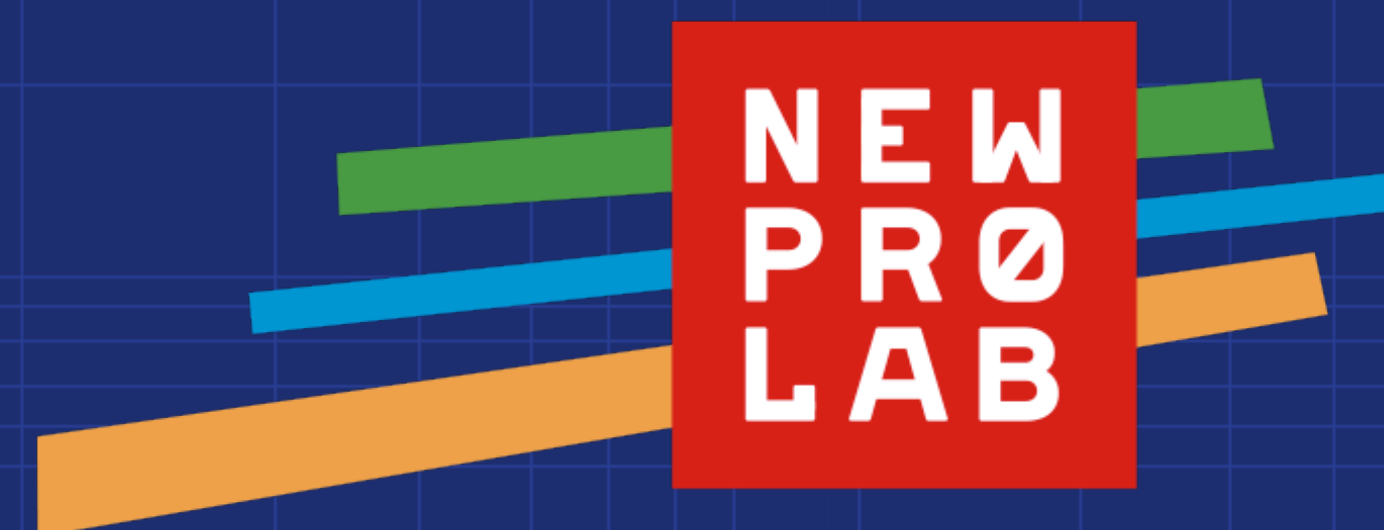
~\$ *cat file.txt | awk -F':' '{ print \$2 }'* # вывод поля по номеру
~\$ *cat file.txt | awk '{ print \$2 "," \$1 }'* # join двух полей запятой
~\$ *cat file.txt | awk '{ print \$1,\$2 }' OFS='\t'* # меняем разделитель
~\$ *cat file.txt | awk '{ x+=\$2 } END { print x }'* # суммируем поле 2
~\$ *cat file.txt | awk -v home=\$HOME/project '{ print home "/data.txt" }'*

<http://www.grymoire.com/Unix/Sed.html>

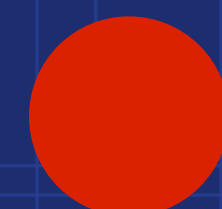
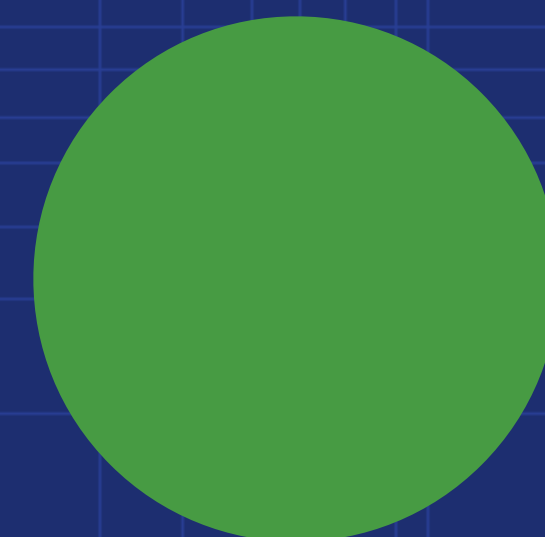
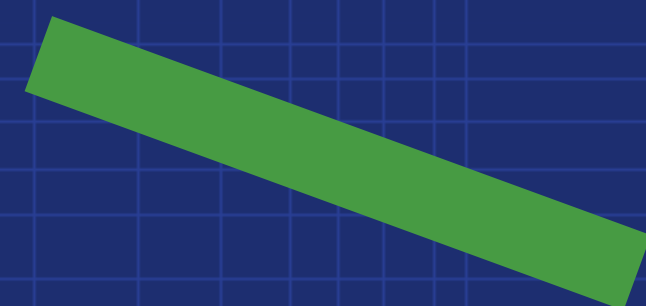
<http://www.grymoire.com/Unix/Awk.html>

WORDCOUNT В КОНСОЛИ

```
~$ cat 76-0.txt | tr '[:upper:]' '[:lower:]' | grep -oE '\w+' | sort | uniq -c |  
sort -nr -k1,1 | head -n 50 | awk '{ print $2 "\t" $1 }' | gnuplot -p -e "set  
term png; set xtic rotate; plot '-' using (column(0)):2:xtic(1) smooth  
freq with boxes" > test.png
```

Процессы и права




ПРАВА ДОСТУПА



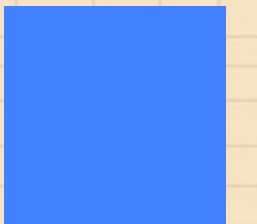
4 - чтение (r)
2 - запись (w)
1 - выполнение (x)

повысить привилегии
~\$ *sudo command*

rwxr-xr-x

меняем права 
~\$ *chmod 700 ./script.py*
~\$ *chmod +x ./script.py*
меняем рекурсивно владельца
~\$ *chown -R user ./dir*
смена группы - chgrp

- первые три - права владельца
- вторые три - права группы
- последние три - права всех остальных



ПРОЦЕССЫ В СИСТЕМЕ

~\$ *ps aux* # посмотреть список всех запущенных процессов

~\$ *htop* # посмотреть то же самое красиво и интерактивно

~\$ *sleep 10 &* # запустить процесс в фоне (прервется при разрыве SSH!)

Ctrl+C - прервать блокирующий запущенный процесс

Ctrl+Z - попытаться отправить запущенный процесс в фон

~\$ *jobs* # список фоновых процессов

~\$ *fg %1* # сделать активным фоновый процесс под номером 1

КАК УБИТЬ ПРОЦЕСС?

- ~\$ *kill 12345* # убиваем по идентификатору
- ~\$ *killall procname* # убиваем все экземпляры с таким именем
- ~\$ *kill -9 12345* # убиваем безоговорочно

КАК ПОДСТАВИТЬ ВЫВОД ОДНОГО АРГУМЕНТОМ ДРУГОМУ?

- ~\$ *find . -name "*.tar.gz" | xargs gunzip -c*
- ~\$ *cat file.csv | xargs -I'{}' -d '\n' -n1 -P8 curl {} > results.txt*

КАК ЗАПУСТИТЬ ПРОЦЕСС И УЙТИ?

~\$ *tmux* # запустить сессию терминального мультиплексора

Ctrl+B+D - отсоединиться от текущей сессии, оставив процесс работать

Ctrl+D - убить текущую сессию и выйти из нее

~\$ *tmux list-sessions* # а можно всех посмотреть?

~\$ *tmux attach* # присоединиться к существующей сессии

Крутые пацаны на своих серверах используют unit-файлы systemd