OTUS

Онлайн образование

otus.ru

Включи запись!

Меня хорошо видно?

Меня хорошо слышно?

Ставим “+”, если все хорошо
“-”, если есть проблемы

Преподаватель

Назаров Денис

Тимлид группы инфраструктурных сервисов в компании Ситимобил.
Вообще то я преподаватель, а тимлид в такси – это для души.

Более 12 лет в IT, половину из них с высоконагруженными проектами. В течении всей карьеры администрирую Linux на больших и маленьких инсталляциях.

В работе активно использую автоматизации на Ansible и Python.

Правила вебинара

Активно
участвуем

Off-topic обсуждаем
в Slack

Задаем вопрос
в чат или голосом

Вопросы вижу в чате,
могу ответить не сразу

Условные обозначения

Индивидуально

Время, необходимое
на активность

Пишем в чат

Говорим голосом

Документ

Ответьте себе или
задайте вопрос

OTUS

Grep sed, awk и другие

Маршрут вебинара

- Цели вебинара
- Утилиты для работы с текстом
- Pipe: одна труба чтоб править всеми
- Grep: фильтр по тексту
- Регулярные выражения
- Поточный текстовый редактор awk
- Поточный текстовый редактор sed
- find + xargs: найти и исполнить
- Итоги вебинара

Цели вебинара

После занятия вы сможете

1.	Научиться работать с утилитами-фильтрами (текста)
2.	Читать и составлять регулярные выражения
3.	Использовать потоковые редакторы
4.	Добавлять новые функции в систему

Смысл

Зачем вам это уметь

Эффективно обрабатывать текстовую информацию. А текст это:

- исходные тексты программ, включая скрипты на Shell
- конфигурационные файлы
- лог файлы
- основной формат ввода/вывода данных для программ и утилит

This is the Unix philosophy:
Write programs that do one thing and do it well.
Write programs to work together.
Write programs to handle text streams because that is a universal interface.

Утилиты для работы с текстом

- cat
- tac
- head
- tail
- sort
- uniq
- wc
- grep
- rev
- paste
- cut
- tr

Утилиты для работы с текстом

- **cat tac** вывести файл целиком
- **head tail** вывести начало и конец файла
- **sort uniq** сортировка и удаление повторов
- **wc** счётчик строк, слов и байт в тексте
- **grep** поиск по образцу
- **rev** перевернуть строку
- **paste** объединить файлы построчно
- **cut** вырезать данные из текста
- **tr** замена или удаление символов

Pipe: одна труба чтоб править всеми

Linux pipe

Pipe (англ.) — труба.

Используется для связи нескольких команд путем перенаправления вывода одной команды (stdout) на вход (stdin) последующей.

```
cat /dev/urandom | grep 'Вечность'
```

Свойства pipe

1. Команды выполняются асинхронно и параллельно в отдельных процессах.
2. Ввод/вывод буферизируется. Буфер 64 Кб. При переполнении запись блокируется.
3. Все операции выполняются в памяти, без нагрузки на диск.

Никак, по крайне мере в ядрах вплоть до версии 5.19

Grep — фильтр по тексту

Название представляет собой акроним английской фразы «*search **g**lobally for lines matching the **r**egular **e**xpression, and **p**rint them*» — «искать везде строки, соответствующие регулярному выражению, и выводить их».

Команда grep ищет заданную строку текста там, где вы скажете: в файле или стандартном потоке.

Есть ещё **e**grep, **f**grep, **p**grep, **z**grep. Раньше это были разные грепы, теперь все фичи* доступны в обычном grep.

*Фича — жаргонизм *от английского feature - особенность, необычное свойство*

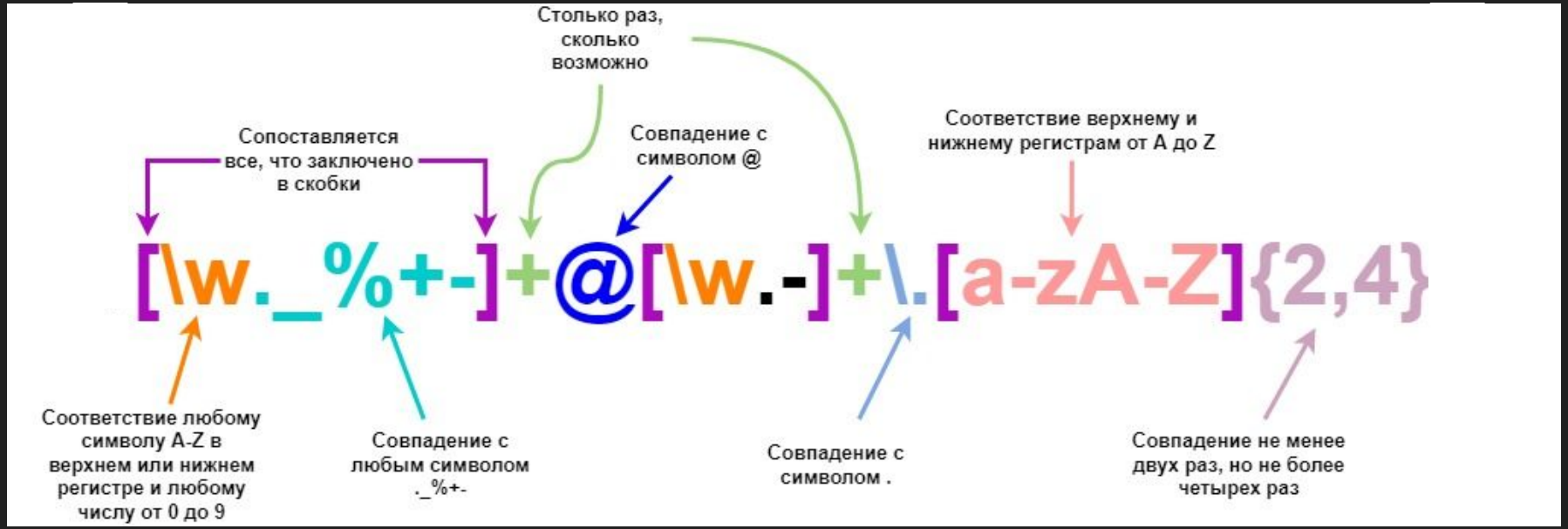
Стандартные потоки Linux: STDIN, STDOUT, STDERR.
Ввод, вывод, ошибки.

Grep: шпаргалка

Настоящая шпаргалка это команда **man grep**. Здесь частоупотребимые опции.

```
# Поиск строк, в которых встречается слово otus
grep 'otus' file
# Поиск строк, в которых встречается слово otus без учёта регистра
grep -i 'otus' file
# Подсчёт количества строк, в которых встречается ключевое слово otus
grep -c 'otus' file
# Поиск номера строки, в которой находится ключевое слово otus
grep -n 'otus' file
# Поиск строк, в которых строка otus отсутствует
grep -v 'otus' file
# Найти строки со словом otus, и вывести ещё 2 после каждой найденной строки A – after
grep -A2 'otus' file
# Найти строки со словом otus, и вывести ещё 2 перед каждой найденной строкой B – before
grep -B2 'otus' file
# Найти строки со словом otus, и вывести 2 строки до и 2 после найденных строк C – context]
grep -C2 'otus' file
# рекурсивный обход всех файлов в каталоге для поиска строк, в которых встречается слово
otus
grep -r 'otus'
```

Удивительный мир регулярных выражений



Регулярное выражение для поиска e-mail адреса

Шпаргалка по регуляркам

[0-9] — Любая цифра
[a-z][A-Z][a-zA-Z] — 3 примера сразу: строчные буквы, прописные буквы, и те и другие буквы
[^0-9][^a-zA-Z]— 2 примера: всё кроме цифр, всё кроме букв (“крышечка” – отрицание)

{2-5} — Модификатор “число повторений”
[0-9]{2} — Число из двух цифр
[а-яА-Я]{2,5} — Слово не короче двух и не длиннее пяти букв

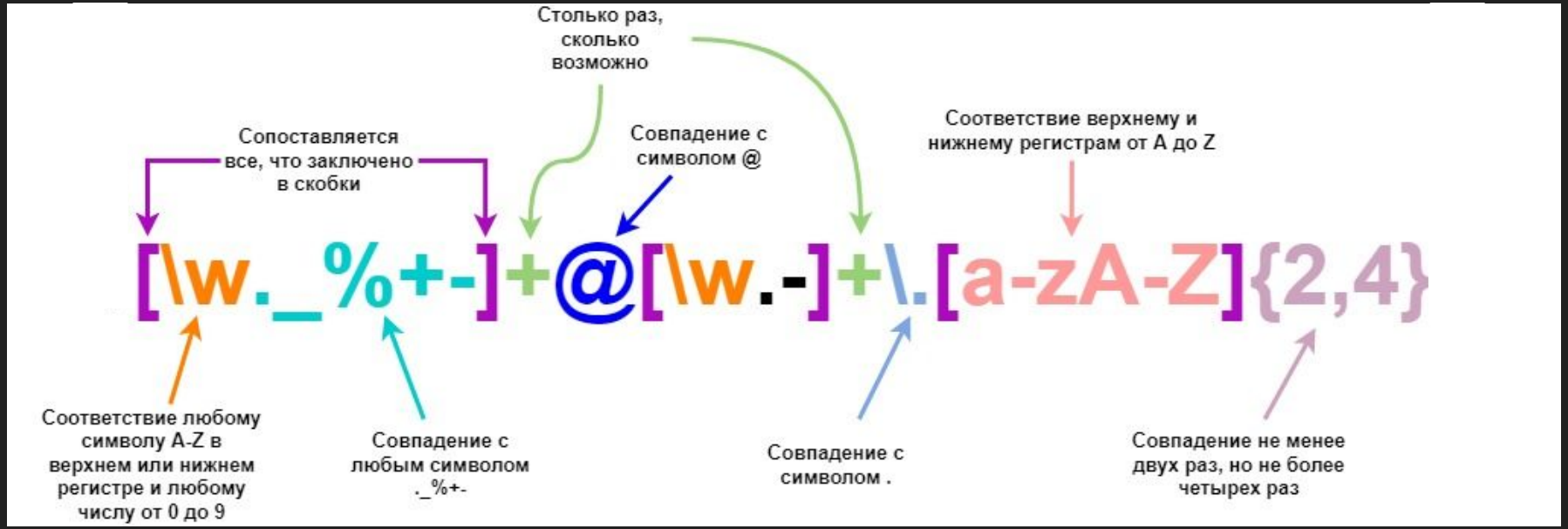
^ — Начало строки
$ — Конец строки
^Otus — Строка начинается со слова Otus
Otus$ — Строка заканчивается словом Otus
^Otus$ — Строка состоит из слова Otus и больше в ней ничего нет

. — заменяет любой символ (любой **один** символ)
* — модификатор “ноль или больше”
+ — модификатор “один или больше”

O.us — Подходит: Otus, Opus, Oous. Не подходит: Ous, Ottus
O.*us — Подходит: Otus, Opus, Oous, Ottus, Ous. Не подходит: OttUS Otsu
O.+us — Подходит: Otus, Opus, Oous, Ottus. Не подходит: Ous Otsu

[0-9]* — Ноль или больше цифр
[0-9]+[a-zA-Z]{2} — Как минимум 1 цифра (или больше одной) и ровно 2 буквы
(42|31) — Вертикальная черта | означает “или”. В данном примере ищем или 42 или 31

Удивительный мир регулярных выражений



Регулярное выражение для поиска e-mail адреса

Ещё более удивительный мир

https://habr.com/ru/post/114156/

```
/(?:(?:\s*[+>~,]\s*|\s+)|[^:+>~,\s\\[\]]+(?:\\.[^
:+>~,\s\\[\]]*)*)|\[(?:[^\\[\]]*(?:\\.[^\\[\]]*)*
|[^=]+=~?\s*(?:"[^\\"]*(?:\\.[^"\\]*)*"|'[^\\']*(
?:\\.[^'\\]*)*'))\]|:[^\\:([]+(?:\\.[^\\:([]*)*(?
:\((?:[^\\()]*(?:\\.[^\\()]*)*|"[^\\"]*(?:\\.[^"\
\]*)*"|'[^\\']*(?:\\.[^'\\]*)*')\))?/g
```

Регулярное выражение для … да какая разница…

OTUS

Перерыв?

Включи запись!

Маршрут вебинара

- Цели вебинара
- Утилиты для работы с текстом
- Pipe: одна труба чтоб править всеми
- Grep: фильтр по тексту
- Регулярные выражения
- Поточный текстовый редактор awk
- Поточный текстовый редактор sed
- find + xargs: найти и исполнить
- Итоги вебинара

AWK. Поточный текстовый редактор

AWK — си-подобный сценарный язык построчного разбора и обработки входного потока (например, текстового файла) по заданным шаблонам (регулярным выражениям). Может использоваться в сценариях командной строки.

Название AWK складывается из первых букв фамилий разработчиков языка — Ахо, Уайнбергера (англ. *Peter J. Weinberger*) и Кернигана. Первая версия была написана в 1977 году в AT&T Bell Laboratories.

Шпаргалка по AWK

`awk '{print $1}'`	– напечатать первый блок
`awk '{print $1 $2}'`	– напечатать первый и второй блоки
`awk '{print $2,$NF}'`	– напечатать первый и последний блоки
`awk 'FS=":" {print $2,$NF}'`	– аналог если разделитель двоеточие
`awk 'FNR==2 {print $0}'`	– напечатать вторую строку
`awk '{print $0}' ORS=' '`	– напечатать файл в одну строку

Переменные:

FS – Разделитель полей (по умолчанию пробел)
NF – Количество строк
FNR – Количество строк в файле
ORS – Выходной разделитель

SED. Поточный текстовый редактор

Sed (от англ. ***S****tream* ***ED****itor*) — потоковый текстовый редактор (а также язык программирования), применяющий различные предопределённые текстовые преобразования к последовательному потоку текстовых данных.

Шпаргалка по sed

Команда d – удалить

sed -e '10 d' – удалить десятую строку

sed -e '2,4 d' – удалить 2,3,4 строки

sed -e '/pts/ d' – удалить строки в которых есть **pts**

sed -e '/^$/d;/^\s*#/d' – удалить пустые и закомментированные строки

Команда s – заменить

sed -e "s/USER/user/g – заменить USER на user

sed -e "1,3 s/USER/user/ – заменить USER на user только в строках 1,2,3

sed -e "3,$ s/USER/user/ – заменить начиная с третьей строки и до конца

Изменение файла

sed -i.bak '<command>' file.txt – выполнить команду <command> и заменить содержимое файла file.txt

sed -i.bak '<command>' file.txt – заменить содержимое файла и сохранить бекап в file.txt.bak

Команда find

Поиск файлов

Онлайн сервис по построению команды с аргументами:

- https://blog.marcinchwedczuk.pl/assets/apps/findform/

Искать по запросу:

linux find command builder

Шпаргалка по команде find

```
find . -type f                          – найти все файлы
find . -type d                          – найти все директории
find . -maxdepth 2 -mindepth 1 – ограничить глубину поиска
find . -type f -mtime +10               – найти файлы с mtime старше 10 дней
find . -type f -mtime +10 -mtime -20 – найти файлы с mtime в диапазоне от 10 до 20 дней
find . -type f -delete                  – удалить найденное
find . -type f -exec echo "founded {}" \;       – передать найденное внешней команде
find ./* -type f | xargs -0 -i mv {} {}.rename  – обработать найденное через xargs
```

Что показалось самым полезным?

Заполните, пожалуйста, опрос о занятии по ссылке в чате