Linux: об операционной системе и файлах

В прошлом модуле вы установили операционную систему, графический интерфейс и своё первое приложение — Telegram.

Теперь у вас есть окружение, которое поможет вам многому научиться. **Окружение** — миниатюрный сервер.

Сервер, как и ваш компьютер, оперирует ресурсами:
- Сеть (Network).
- Процессор (CPU).
- Оперативная память (Random Access Memory).
- Жёсткий диск (Hard Drive, HDD).

Процессор выполняет код программ. Программы — это набор инструкций, поэтому основная работа процессора — эти инструкции очень быстро выполнять. Процессор постоянно работает с оперативной памятью: берёт оттуда данные, сохраняет там какие-то результаты своей работы. Работа процессора с оперативной памятью чуть медленнее, чем работа внутри процессора. Но гораздо быстрее, чем работа с жёстким диском.

Самый простой пример данных, которые хранятся в **оперативной памяти,** — это так называемый буфер обмена. Когда вы копируете текст или вырезаете его, чтобы вставить в другое место, он хранится в оперативной памяти. Когда вы работаете с текстовым редактором, пока вы не сохранили документ, он в основном хранится в оперативной памяти. Содержимое оперативной памяти хранится, только пока компьютер работает, и пропадает, если вы его перезагружаете или выключаете.

На **диск** сохраняются данные, доступ к которым может быть более медленным: изображения, документы и прочее.

Сеть позволяет соединять ваш компьютер с другими.

Задачи, которые выполняет диск и сеть, то есть обмен между вашим компьютером и внешним миром (а также человеком и другими компьютерами), называются термином «ввод/вывод» (Input/Output или I/O).
Клавиатура, тачпад или мышь, микрофон — примеры устройств ввода.
Монитор, аудиоустройства, принтер — примеры устройств вывода.

Задачи в компьютере выполняет процессор. Поэтому отправка электронного письма или сообщения в Telegram, сохранение файлов на диск — это тоже примеры процессов ввода/вывода.

Теперь посмотрим, как операционная система взаимодействует с этими ресурсами.

Операционная система состоит из нескольких частей:

- **Ядро.** Одна из основных его функций — обеспечивать работу с ресурсами.
- **Библиотеки.** Чаще всего программисты не пишут всю программу с нуля — они используют библиотеки. Код, который выполняет заданную функцию, можно переиспользовать из открытых источников — это и называется библиотеками. Например, мы можем воспользоваться уже реализованными функциями для шифрования или для работы с сетью. Аналогию с известной нам библиотекой, хранилищем книг, можно провести следующим образом: мы можем воспользоваться результатом чьего-то труда — например, найти математический метод для решения задачи или использовать уже доказанную теорему.
- **Пользовательское окружение.** Это пространство, где запускаются приложения. В нём работают различные программы, например Telegram.
- **Драйверы.** Это программы, необходимые для того, чтобы мы могли подключать к компьютеру различные устройства — флешки, внешние диски, клавиатуру и так далее — и компьютер знал, что это и как с этим работать.

Понимание, как операционная система работает с этими ресурсами, как мы можем оценить, насколько они заняты (утилизированы), помогает нам делать так, чтобы наши программы работали. И продолжали работать, когда к нам приходит большая нагрузка.

Представьте, что вы сделали мобильное приложение, которое что-то продаёт. За работой такого приложения обычно стоит сервер, вернее, много разных серверов. У вас появился первый клиент, второй клиент. Приложение стало популярным в пределах какого-то небольшого города. Потом случилось что-то очень хорошее, и ваше приложение стало неожиданно популярным в другом городе, в ещё одном городе и, наконец, по всей России. Продолжать обрабатывать возросшую нагрузку на ваши серверы и обслуживать всех ваших пользователей — это и есть ваша задача.

Умение читать то, что говорят различные диагностические утилиты, позволит решать самые разные проблемы и чинить самые разные сбои — мы займёмся этим позже. Сейчас продолжим собирать нашу картину мира.

Итак, вы разобрались с операционной системой как с концепцией. Есть два важнейших понятия в операционной системе Unix — файл и процесс.

Все данные, которыми мы оперируем, — это файлы. Программа сама по себе — это **файл** (помните, вы скачивали файл .exe для установки VirtualBox?), программы читают информацию из файлов и пишут информацию в файлы.

Вы будете сталкиваться с ними постоянно и всё больше узнавать о том, почему это так удобно. А вот любая работающая программа (и те ресурсы, которые предоставила ей операционная система), которую вы запускаете, — это **процесс** (или зачастую несколько процессов).

Файлы собраны в папки. Папки ещё называют каталогами или директориями.

Для чего нужно понимание работы с файлами и иерархии?

Работа с файлами

Откроем консоль.
Чтобы понять, куда вы попали, наберём:

```
pwd
```

Эта команда показывает вам путь до папки, где вы находитесь, — текущей директории. Сейчас вы в так называемой домашней директории. Если вы работаете с macOS, вы её, скорее всего, никогда не покидали (внутри неё находятся папки Downloads, Documents и другие), приблизительный аналог в Windows — «Мои документы».

Чтобы переходить в другие папки, мы используем команду **cd** (change directory). Наберём:

```
cd /
```

Директории и файлы в Linux (как и в macOS, и в Windows) организованы иерархически. Это удобно и эффективно (например, для поиска). Такая иерархия напоминает перевёрнутое дерево. Символом «слеш» (/) обозначается «корневая» директория, самый верх нашей иерархии. От неё отходят ветви, на которых расположены другие директории — как листья.

Теперь наберём ls и посмотрим, какие здесь есть файлы и директории.

```
ls
```

Вернёмся в свою домашнюю директорию. Она находится в каталоге **/home** и называется так же, как и ваш пользователь.

```
cd /home/victoria
```

Если мы введём ls в ней, увидим, что в ней ничего нет.
Но если введём:

```
ls -la
```

увидим некоторые файлы, название которых начинается с точки.
В таких файлах хранятся пользовательские настройки для различных программ. И иногда — различные данные для приложений, например для браузера и Telegram.
Обратите внимание, что **ls** показывает нам ещё две строки — с точкой и двумя точками. Точка — это обозначение текущей директории. Например, можно набрать:

```
ls .
```

и увидеть содержимое директории, где мы находимся. Две точки — обозначение директории выше по иерархии. Например, находясь в своей домашней директории, мы можем ввести:

```
cd ..
```

Затем:

```
ls .
```

и увидеть домашние директории всех пользователей компьютера.

Для **создания файла и директории** используются разные команды: **touch и mkdir.**

Файл либо директория создаются в той же директории, в которой вы находитесь, либо можно задать путь, по которому вы создаёте файл или директорию.

```
touch myfirstfile
mkdir myfirstdir
ls .
```

Помимо названия, не очень понятно, что из этого файл, а что директория.

```
ls -la
```

```
touch myseconddir/mysecondfile
```

Получили ошибку!

```
mkdir myseconddir
touch myseconddir/mysecondfile
```

Теперь директория существует, можно создать файл.