Всем привет. И добро пожаловать в блок «Работаем с файлами в GO», где мы рассмотрим работу с файлами в GO, в контексте различных операционных систем. Блок будет небольшой, но достаточно важный и интенсивный. Погнали.

Так, о чем же мы услышим сегодня, о чем поговорим? Мы поговорим, о том, как устроены файловые системы Windows и Linux, пройдемся по общим концепциям и положениям. И посмотрим на практике, как работать с файлами в Golang. Вернее, как читать из файла, как записать файл, как создать файл. В принципе это, ну удалить файл, это, в принципе, те самые операции, которые мы выполняем в языках программирования с файлами.

На всякий случай напомню, как устроены файловые системы. Базовой единицей хранения информации в файловой системе, как ни удивительно, является файл. Это некая, ограниченная на диске, область данных, в которой записано что-то. Что-то, что на нее записано, определяется форматом файла, так называемым. И, соответственно, в обычном расширении файла, после точки, указывается его формат. То есть указывается пособие системе о том, как распознавать эти файлы, как их читать. Ну и соответственно, там существуют ещё всякие служебные вещи для понимания другими приложениями, как читаются эти файлы. Файлы лежат в папках, папки лежат в папках и так далее. И файлы, и папки имеют разные права доступа и разных владельцев, почти во всех операционных системах.

И последняя теоретическая часть, непосредственно, перед практикой, это различия в их файловых системах Windows и Linux. В Windows, соответственно, диски у вас разделены на один или несколько логических дисков, или объединены в один или несколько логических дисков. Они все начинаются с буквы, и по факту файлы лежат уже в них.

В Linux-е, система монтируется, так называемый – корень. Определённый диск, который вы указываете при, значит, установки системы, и отдельные дополнительные диски монтируются, как отдельные устройства. И вы можете какие-то пути на них, как бы перемонтировать, то есть они будут перенесены физически на другие диски.

И диски в Linux-е – это тоже файлы, как таковые устройства, к ним можно обратиться. Соответственно, папки в Linux-е и файлы имеют уровни доступа для владельцев группы и других пользователей, соответственно, вы можете поставить отдельное разрешение владельцу, отдельное разрешение группе читать, писать, изменять, запускать. И отдельно, следственно, всем остальным пользователям.

В Windows-е всё намного сложнее. В Windows есть списки пользователей. У каждого пользователя можно гибко настраивать конкретные права доступа к файлу папки, всё это контролируется системой, которая называется ACL.

ACL - Access Control (НРЗ 02:51). И отдельные как бы записи в этой системе, называются ACE - Access Control (НРЗ 02:56), которые представляют собой какие-то права доступа конкретного пользователя.

Это выходит за рамки данной лекции. Забегая вперед, скажу, что именно поэтому Golang, в Golange система управления правами пользователей не идеально подходит для Windows.

Соответственно, последнее, что нам нужно помнить, это путь к папке или файлу Windows выглядит следующим образом: сначала буква диска двоеточие обратные слэши, папки и, соответственно, файлы. Путь в Linux-е выглядит: прямой слэш, соответственно, корень, папка и ещё папка, файл и его расширение.

Соответственно, в принципе, на этом всё. Давайте уже посмотрим, как же работать с файлами, папками в GO.

В рамках сегодняшней лекции мы будем работать с Package OS, опять, с его частью, которая работает с файлами.

Соответственно, что она делает? Работает с файловой системой, умеет создавать файлы, папки, в них писать, из них читать. Соответственно, она Linux native, она заточена под Linux, в частности, в разделе прав доступа. С Windows она работает с костылями. То есть записать и считать папку файла Windows вы можете. А вот поддержать ACL и прочие ACE Windows без отдельных библиотек или без, соответственно, работы уже самой системы, вы не можете. То есть это не вполне универсальная система, но к нашему счастью, не так часто, не настолько часто это нужно.

Давайте уже перейдём к практическим примерам. И посмотрим, как это делается.