В процессе наших странствий по Go, можно увидеть переменные goroot, gopath и goproxy. В описании этих переменных можно увидеть, что поставьте свой goroot туда и туда, и неплохо было бы понимать, что это значит.

Итак, все, на самом деле, довольно просто. Во-первых, начнем с того, что goroot и gopath с go1.8 уже указывать (НРЗ 00:24) не надо. Указываете просто go в своем path. И все, у вас есть goroot. Но если вдруг надо, то goroot – это переменная, определяющая путь установки вашего языка. Если у вас на машине установлено несколько версий языка, несколько go, то вы можете использовать переменную goroot локально для того, чтобы поставить, соответственно, нужную версию к вашему Builder-у. Это, в принципе, тоже самое, что поменять версию go в вашей переменной path.

Gopath, напротив, определит, где лежат файлы вашего проекта, так называемый, Workspace. То есть, она определит, где, значит, у вас корень проекта. Как правило, это папочка src внутри goroot, но вы, естественно, можете поставить ее в своем проекте. Соответственно, бывает глобальный – для всех проектов, бывает проектный и бывает модульный.

И goproxy – самое интересное. Go базируется на системе модулей, как мы уже увидели, да. И на системе зависимостей от этих модулей. Соответственно, бывает, что у вас закрытая локальная сеть. Бывает, что у вас нет доступа в интернет. Бывает, что вы в Китае и собираете там какие-то приложения, а у них там свои правила на тему интернета. И вам нужно эти модули скачать каким-то образом и их использовать внутри вашей сети. Вот таким образом, вы можете поставить goproxy, и goproxy будет лазить в ваш как бы репозиторий и качать оттуда зависимости. Еще очень удобно, если у вас в проекте какие-то, например, статические зашиты зависимости, вы не хотите обновлять библиотеки, не хотите по каким-то причинам использовать (НРЗ 02:04), об этом чуть позже, вы можете,

соответственно, использовать goproxy. Туда сохранить, сложить все библиотеки и, соответственно, использовать это, как ваш локальный репозиторий.

Итак, давайте рассмотрим переменные goroot и gopath на примере (НРЗ 2:20). Я не буду рассматривать это в Linux-е, не буду показывать эти переменные path. Наверняка, вы уже знаете, как устанавливается переменная path. И знаете, как выглядят (НРЗ 02:32) переменные. Соответственно, что я здесь хочу показать, это то, что вот у меня установлено несколько версий. Они установлены вообще на разные диски физически. И вот одна из версий это 1.16, вторая версия 1.15, соответственно, установленная в D go. И вторая, версия 1.16 установлена в C-users и так далее.

Соответственно, я могу их менять, и это будет влиять на мой, соответственно, компилятор, на мои команды там go build, go run. Вот эти все команды, которыми мы запускаем приложение, потому что они будут использовать физически разные версии go, физически лежащие в разных местах. Соответственно, структура проекта, структура go выглядит таким образом: в папочке go есть… go bin, содержащbt в себе команду go и gofmt. Также содержатся всякие стандартные lib-ы, которые там есть, содержит Package-ы, которые вы качаете, и содержит ваш, соответственно, содержит ваш…, исходники.

Соответственно, что еще здесь можно сказать? В принципе, это все. Если у вас на машине физически разные версии go, вам нужно просто переключать их путем перестановки переменной goroot, и у вас все будет работать. Либо, вы просто ставите одну версию go и из-под нее работаете.

Соответственно, версия gopath это, в принципе, тоже самое. То есть вот у меня есть. Вернее, это не тоже самое. Это переменная, которая вам позволяет указать путь к своей рабочей папке. И в чем прикол? То есть в этой папке будет у вас 3,

соответственно, 3 подпапки. Ну здесь у меня немножко мусора, потому что у меня здесь же лежит версия go, но, в принципе, все должно быть понятно. То есть есть папочка bin – куда складываются всякие приложеньки, нужные нам, есть папочка Package – куда складываются наши скачанные зависимости для проектов и есть папочка src. То есть раньше, как было, до версии 1.8. Вот это все складывалось в папочку Package. И вы только могли пользоваться вещами оттуда, то есть, если у вас там какие-то несколько модулей было, у вас все там перезатиралось, либо раскладывалось по подпапкам, зависимости лежали там же, ну, в общем, было неудобно. Теперь с модулями стало несколько более удобно. Сейчас я вам, соответственно, покажу уже модули.