Если вы видите что-то необычное, просто сообщите мне.

# Проблемы различия библиотек при компиляции

# Ошибка

Во время компиляции и не соответствия версий систем может возникнут подобная ошибка

```
/lib/x86_64-linux-gnu/libc.so.6: version `GLIBC_2.28' not found (required by appname)
```

Ошибка говорит о том, что в системе нет версии `GLIBC_2.28`

Чтобы узнать какая версия находится в системе можно воспользоваться следующей команадой

```
ls /lib/x86_64-linux-gnu/libc
libc-2.24.so            libcap.so.2.25          libcom_err.so.2.1       libcrypt.so.1
libcap-ng.so.0          libcidn-2.24.so         libcrypt-2.24.so        libc.so.6
libcap-ng.so.0.0.0      libcidn.so.1            libcryptsetup.so.4
libcap.so.2             libcom_err.so.2         libcryptsetup.so.4.7.0
```

Отсюда мы видим, что системная библиотека отличается от системы на котрой собирается приложение: 2.24 вместо 2.28

# Решение

Для того, чтобы обойти эту проблему можем воспользоваться docker.

Узнаем какая версия операционной системы у нас стоит:

```
 uname -a
Linux hostname 4.9.0-12-amd64 #1 SMP Debian 4.9.210-1+deb9u1 (2020-06-07) x86_64 GNU/Linux
```

Мы видим, что наша операционная система Debian 9.1 что сильно упрощает нам поиск нужного образа. Находим в Docker hub нужный нам образ, в нашем случае, всё очень просто, дебиан тегирует образы по разному и наш образ будет `debian:9.1`, это завсит от авторов образа. Его мы и будем использовать. Устанавливаем необходимые пакеты `wget` и `gcc` и скачиваем свеженький `go`.

1. В корне проекта над которым работаем создаем `Dockerfile`.

```
FROM debian:9.1
RUN apt update && apt install wget gcc -y
RUN wget https://go.dev/dl/go1.18.4.linux-amd64.tar.gz
RUN rm -rf /usr/local/go && tar -C /usr/local -xzf go1.18.4.linux-amd64.tar.gz
WORKDIR /code
```

2. Там же создаем `docker-compose.yml` который будем использовать для компиляции проекта:

```
version: "3"

services:
  servicename:
    build: .
    volumes:
      - ".:/code"
```

3. Билдим докер и в нем собираем наш проект:

```
docker-compose build --no-cache
docker-compose run  servicename bash -c 'export PATH=$PATH:/usr/local/go/bin && GOOS=linux
GOARCH=amd64 go build -o bin/appname .'
```

Первую команду в дальнейшем можно не запускать.

# Послесловие

Таким способом можно решать различные проблемы возникающие при сборке приложения на окружениях отличных от того окружения где будет происходить работа. Так в случае необходимости, можно изменять Dockerfile до стостояние близкого к состоянию рабочего окружения. В этом случае команда `docker-compose build --no-cache` обязательна.

---

Revision #1

Created 2022-07-23 10:31:47 UTC by gasick

Updated 2022-07-23 10:46:58 UTC by gasick