多段式DockerFile构建Http容器

原始需求

- 构建本地镜像
- 编写 Dockerfile 将练习 2.2 编写的 httpserver 容器化
- 将镜像推送至 docker 官方镜像仓库
- 通过 docker 命令本地启动 httpserver
- 通过 nsenter 进入容器查看 IP 配置

作业需编写并提交 Dockerfile 及源代码。

HttpServer代码

main.go的代码与之前一致【使用81端口】:

```go
package main

import (
	"fmt"
	"os"
	"strconv"

	//"github.com/thinkeridea/go-extend/exnet"
	"io"
	"log"
	"net/http"
	"strings"
)

/*
编写一个 HTTP 服务器，大家视个人不同情况决定完成到哪个环节，但尽量把 1 都做完：

	1，接收客户端 request，并将 request 中带的 header 写入 response header
	2，读取当前系统的环境变量中的 VERSION 配置，并写入 response header
	3，Server 端记录访问日志包括客户端 IP，HTTP 返回码，输出到 server 端的标准输出
	4，当访问 localhost/healthz 时，应返回 200
*/

// Main方法入口
func main() {
	println("环境正常")

	// 功能1
	http.HandleFunc("/requestAndResponse", requestAndResponse)

	// 功能2
	http.HandleFunc("/getVersion", getVersion)

	// 功能3
```

```
    http.HandleFunc("/ipAndStatus", ipAndStatus) //注册接口句柄

    // 功能4
    http.HandleFunc("/healthz", healthz) //注册接口句柄

    err := http.ListenAndServe(":81", nil) //监听空句柄，80端口被占用，使用81端口
    if nil != err {
        log.Fatal(err) //显示错误日志
    }
}

// 功能1，接收请求及响应
func requestAndResponse(response http.ResponseWriter, request *http.Request) {
    println("调用requestAndResponse接口")
    headers := request.Header //header是Map类型的数据
    println("传入的hander：")
    for header := range headers { //value是[]string
        //println("header的key：" + header)
        values := headers[header]
        for index, _ := range values {
            values[index] = strings.TrimSpace(values[index])
            //println("index=" + strconv.Itoa(index))
            //println("header的value：" + values[index])

        }
        //valueString := strings.Join(values, "")
        //println("header的value：" + valueString)
        println(header + "=" + strings.Join(values, ","))        //打印request的
header的k=v
        response.Header().Set(header, strings.Join(values, ",")) // 遍历写入
response的Header
        //println()

    }
    fmt.Fprintln(response, "Header全部数据:", headers)
    io.WriteString(response, "succeed")

}

// 功能2，获取环境变量的version
func getVersion(response http.ResponseWriter, request *http.Request) {
    println("调用getVersion接口")
    envStr := os.Getenv("VERSION")
    //envStr := os.Getenv("HADOOP_HOME")
    //println("系统环境变量：" + envStr) //可以看到 C:\soft\hadoop-3.3.1   Win10需要
重启电脑才能生效

    response.Header().Set("VERSION", envStr)
    io.WriteString(response, "succeed")

}

// 功能3，输出IP与返回码
func ipAndStatus(response http.ResponseWriter, request *http.Request) {
    println("调用ipAndStatus接口")

    form := request.RemoteAddr
    println("Client->ip:port=" + form) //虚拟机是桥接模式。使用postman返回的全部是127.0.0.1  用手机打开网站192.168.1.139:81/ipAndStatus可以看到新IP
    ipStr := strings.Split(form, ":")
    println("Client->ip=" + ipStr[0]) //打印ip

    // 获取http响应码
    //response.WriteHeader(301) //手动设置响应码，默认200

    //response.WriteHeader(http.StatusOK)//由于默认是调用这个，∴返回码都是这个200【server.go有源码】
    println("Client->response code=" + strconv.Itoa(http.StatusOK))

    //println("response code->: " + code)

    io.WriteString(response, "succeed")
}

// 功能4，连通性测试接口
func healthz(response http.ResponseWriter, request *http.Request) {
    println("调用healthz接口")
    response.WriteHeader(200) //设置返回码200
    //response.WriteHeader(http.StatusOK)//默认会调用这个方法，默认就是200【server.go有源码】
    io.WriteString(response, "succeed")
}
```

编写DockerFile

多阶段构建【2个from】：

```
FROM golang:1.17 AS builder

ENV GO111MODULE=off \
    CGO_ENABLED=0 \
    GOOS=linux \
    GOARCH=amd64

WORKDIR /build
COPY . .
RUN go build -o httpserver .

FROM scratch
COPY --from=builder /build/httpserver /
EXPOSE 81
ENTRYPOINT ["/httpserver"]
```

本地启动

查看之前的状态：

```
zhiyong@zhiyong-vm-dev:~/dockerstudy/httpserver2$ pwd
/home/zhiyong/dockerstudy/httpserver2
zhiyong@zhiyong-vm-dev:~/dockerstudy/httpserver2$ ll
总用量 16
drwxrwxr-x 2 zhiyong zhiyong 4096  5月  8 23:18 ./
drwxrwxr-x 4 zhiyong zhiyong 4096  5月  8 23:14 ../
-rw-rw-r-- 1 zhiyong zhiyong  247  5月  8 23:15 Dockerfile
-rw-rw-r-- 1 zhiyong zhiyong 3559  5月  8 23:18 main.go
zhiyong@zhiyong-vm-dev:~/dockerstudy/httpserver2$ sudo docker images
REPOSITORY                             TAG       IMAGE ID       CREATED
SIZE
gohttpserver                           v1        ffdef51c9a42   3 days ago
83.9MB
zhiyongdocker/dockerstudy/httpserver   v1        ffdef51c9a42   3 days ago
83.9MB
zhiyongdocker/dockerstudy              v1        ffdef51c9a42   3 days ago
83.9MB
ubuntuwithvm                           latest    428e030662bc   3 days ago
170MB
ubuntu                                 latest    d2e4e1f51132   8 days ago
77.8MB
```

启动:

```
zhiyong@zhiyong-vm-dev:~/dockerstudy/httpserver2$ sudo docker build -t
dockerfilehttpserver:v2 -f Dockerfile .
Sending build context to Docker daemon  6.144kB
Step 1/9 : FROM golang:1.17 AS builder
 ---> b6bd03a3a78e
Step 2/9 : ENV GO111MODULE=off  CGO_ENABLED=0   GOOS=linux      GOARCH=amd64
 ---> Running in be9cbc34d989
Removing intermediate container be9cbc34d989
 ---> ea1e675df98d
Step 3/9 : WORKDIR /build
 ---> Running in f797c4e46357
Removing intermediate container f797c4e46357
 ---> 742b54aa983d
Step 4/9 : COPY . .
 ---> 65e6c234fdae
Step 5/9 : RUN go build -o httpserver .
 ---> Running in 5d4d0ddc1216
Removing intermediate container 5d4d0ddc1216
 ---> 0206c712d151
Step 6/9 : FROM scratch
 --->
Step 7/9 : COPY --from=builder /build/httpserver /
 ---> 8a14b05b18f3
Step 8/9 : EXPOSE 81
 ---> Running in 84fecd9218f4
Removing intermediate container 84fecd9218f4
 ---> 60e0825f8f29
Step 9/9 : ENTRYPOINT ["/httpserver"]
 ---> Running in 3d3e26e28290
Removing intermediate container 3d3e26e28290
 ---> 16e4ba2b8b99
Successfully built 16e4ba2b8b99
Successfully tagged dockerfilehttpserver:v2
zhiyong@zhiyong-vm-dev:~/dockerstudy/httpserver2$ sudo docker images
REPOSITORY                            TAG       IMAGE ID       CREATED
SIZE
dockerfilehttpserver                  v2        16e4ba2b8b99   6 seconds ago
6.08MB
<none>                                <none>    0206c712d151   7 seconds ago
962MB
gohttpserver                          v1        ffdef51c9a42   3 days ago
 83.9MB
zhiyongdocker/dockerstudy/httpserver  v1        ffdef51c9a42   3 days ago
 83.9MB
zhiyongdocker/dockerstudy             v1        ffdef51c9a42   3 days ago
 83.9MB
ubuntuwithvm                          latest    428e030662bc   4 days ago
170MB
ubuntu                                latest    d2e4e1f51132   8 days ago
 77.8MB
golang                                1.17      b6bd03a3a78e   2 weeks ago
941MB



zhiyong@zhiyong-vm-dev:~/dockerstudy/httpserver2$ sudo docker images
REPOSITORY                            TAG       IMAGE ID       CREATED
SIZE
dockerfilehttpserver                  v2        16e4ba2b8b99   6 seconds ago
6.08MB
<none>                                <none>    0206c712d151   7 seconds ago
962MB
gohttpserver                          v1        ffdef51c9a42   3 days ago
 83.9MB
zhiyongdocker/dockerstudy/httpserver  v1        ffdef51c9a42   3 days ago
 83.9MB
zhiyongdocker/dockerstudy             v1        ffdef51c9a42   3 days ago
 83.9MB
ubuntuwithvm                          latest    428e030662bc   4 days ago
170MB
ubuntu                                latest    d2e4e1f51132   8 days ago
 77.8MB
golang                                1.17      b6bd03a3a78e   2 weeks ago
941MB
zhiyong@zhiyong-vm-dev:~/dockerstudy/httpserver2$ sudo docker run -p 83:81
dockerfilehttpserver:v2
环境正常
```

可以看到容器启动，httpserver的服务也启动。

推送镜像

```
root@zhiyong-vm-dev:/home/zhiyong# sudo docker images
REPOSITORY                            TAG       IMAGE ID       CREATED
SIZE
```

```
dockerfilehttpserver                    v2        16e4ba2b8b99   16 minutes ago
6.08MB
<none>                                  <none>    0206c712d151   16 minutes ago
962MB
gohttpserver                            v1        ffdef51c9a42   3 days ago
83.9MB
zhiyongdocker/dockerstudy/httpserver    v1        ffdef51c9a42   3 days ago
83.9MB
zhiyongdocker/dockerstudy               v1        ffdef51c9a42   3 days ago
83.9MB
ubuntuwithvm                            latest    428e030662bc   4 days ago
170MB
ubuntu                                  latest    d2e4e1f51132   8 days ago
77.8MB
golang                                  1.17      b6bd03a3a78e   2 weeks ago
941MB
root@zhiyong-vm-dev:/home/zhiyong# sudo docker tag dockerfilehttpserver:v2
zhiyongdocker/dockerfilehttpserver:v1
root@zhiyong-vm-dev:/home/zhiyong# sudo docker images
REPOSITORY                              TAG       IMAGE ID       CREATED
SIZE
dockerfilehttpserver                    v2        16e4ba2b8b99   18 minutes ago
6.08MB
zhiyongdocker/dockerfilehttpserver      v1        16e4ba2b8b99   18 minutes ago
6.08MB
<none>                                  <none>    0206c712d151   18 minutes ago
962MB
gohttpserver                            v1        ffdef51c9a42   3 days ago
83.9MB
zhiyongdocker/dockerstudy/httpserver    v1        ffdef51c9a42   3 days ago
83.9MB
zhiyongdocker/dockerstudy               v1        ffdef51c9a42   3 days ago
83.9MB
ubuntuwithvm                            latest    428e030662bc   4 days ago
170MB
ubuntu                                  latest    d2e4e1f51132   8 days ago
77.8MB
golang                                  1.17      b6bd03a3a78e   2 weeks ago
941MB
root@zhiyong-vm-dev:/home/zhiyong# sudo docker push
zhiyongdocker/dockerfilehttpserver:v1
The push refers to repository [docker.io/zhiyongdocker/dockerfilehttpserver]
72f082039919: Pushed
v1: digest:
sha256:fce495f9c682b4bbb47ad35130044d228193cbba70b43829c8dc1a848b45f16a size:
528
```

看到成功推送到DockerHub。

查看网络IP

```
zhiyong@zhiyong-vm-dev:~$ sudo docker ps -a
CONTAINER ID   IMAGE                          COMMAND                   CREATED
      STATUS                       PORTS                                   NAMES
```

```
0241ac11b13a   dockerfilehttpserver:v2        "/httpserver"           3 minutes
ago   Up 3 minutes              0.0.0.0:83->81/tcp, :::83->81/tcp
quizzical_hermann
8377df17b1e1   zhiyongdocker/dockerstudy:v1   "/dockerstudy/httpse…"   3 days
ago      Exited (2) 9 minutes ago
zen_ritchie
8e97bbad0d3b   zhiyongdocker/dockerstudy:v1   "/dockerstudy/httpse…"   3 days
ago      Exited (2) 3 days ago
blissful_black
93d3eac39a12   gohttpserver:v1                "/dockerstudy/httpse…"   3 days
ago      Exited (2) 3 days ago
awesome_johnson
c515a662d46c   ubuntu                         "/bin/bash"             4 days
ago      Exited (0) 4 days ago
goofy_heisenberg
zhiyong@zhiyong-vm-dev:~$ sudo docker container top 0241ac11b13a
UID                 PID                 PPID                C
STIME               TTY                 TIME                CMD
root                23686               23665               0
23:35               ?                   00:00:00            /httpserver
zhiyong@zhiyong-vm-dev:~$ sudo nsenter -n -t23686
root@zhiyong-vm-dev:/home/zhiyong# ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group
default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
       valid_lft forever preferred_lft forever
30: eth0@if31: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc noqueue state UP
group default
    link/ether 02:42:ac:11:00:02 brd ff:ff:ff:ff:ff:ff link-netnsid 0
    inet 172.17.0.2/16 brd 172.17.255.255 scope global eth0
       valid_lft forever preferred_lft forever
```