基于Docker的微服务自动化测试

第三天：Docker微服务测试编排篇

> 核心内容：docker-compose

内容提要：

- 微服务架构特点
- docker-compose 常用命令
- docker-compose 语法
- 如何进行容器之间的测试
- fastapi+beifan+allure的容器化测试架构实战

第一天：docker是什么，怎么安装，怎么启动

第二天：镜像怎么来，怎么封装，怎么传输

第三天：多容器直接的交互

微服务基本特点：

1. 对外暴露单一：对外只有一个HTTP接口
2. 对内服务特别多：接口、数据、redis、消息队列、日志收集

将测试代码，放入微服务架构内

两个前提：

1. 把测试代码打包为镜像 （上节课讲过了）
2. 容器编排：让容器们一起运行

一、docker-compose

是一个用于定义和运行**多容器**的docker工具

借助它，可以使用 **yaml** 配置程序和服务

使用**单个命令**，创建并启动所有的服务

1. 安装docker-compose

在ubuntun安装此工具

```
sudo apt install docker-compose
```

2. docker-compose 命令

```
$ docker-compose --help
Define and run multi-container applications with Docker.

Usage:
  docker-compose [-f <arg>...] [options] [COMMAND] [ARGS...]
  docker-compose -h|--help

Options:
  -f, --file FILE             Specify an alternate compose file
                              (default: docker-compose.yml)
  -p, --project-name NAME     Specify an alternate project name
                              (default: directory name)
  --verbose                   Show more output
  --log-level LEVEL           Set log level (DEBUG, INFO, WARNING, ERROR,
CRITICAL)
  --no-ansi                   Do not print ANSI control characters
  -v, --version               Print version and exit
  -H, --host HOST             Daemon socket to connect to

  --tls                       Use TLS; implied by --tlsverify
  --tlscacert CA_PATH         Trust certs signed only by this CA
  --tlscert CLIENT_CERT_PATH  Path to TLS certificate file
  --tlskey TLS_KEY_PATH       Path to TLS key file
  --tlsverify                 Use TLS and verify the remote
  --skip-hostname-check       Don't check the daemon's hostname against the
                              name specified in the client certificate
  --project-directory PATH    Specify an alternate working directory
                              (default: the path of the Compose file)
  --compatibility             If set, Compose will attempt to convert keys
                              in v3 files to their non-Swarm equivalent
  --env-file PATH             Specify an alternate environment file

Commands:
  build              构建
  bundle             Generate a Docker bundle from the Compose file
  config             Validate and view the Compose file
  create             Create services
  down               停止并删除
  events             Receive real time events from containers
  exec               Execute a command in a running container
  help               Get help on a command
  images             List images
  kill               Kill containers
  logs               View output from containers
```

```
  pause               Pause services
  port                Print the public port for a port binding
  ps                  List containers
  pull                Pull service images
  push                Push service images
  restart             Restart services
  rm                  Remove stopped containers
  run                 Run a one-off command
  scale               Set number of containers for a service
  start               Start services
  stop                Stop services
  top                 Display the running processes
  unpause             Unpause services
  up                  创建并启动
  version             Show the Docker-Compose version information
```

3. docker-compose.yaml

是一个yaml文件，用到了yaml 语法

docker-compose内容结构

问题来了：

> dockerfile一定以什么内容开头?
>
> from开头（指令建议使用大写）

```yaml
version: '3'  # 指定 docker-compose版本
services:
  postgres:    # 服务名称
    image: library/postgres:11.5-alpine  # 镜像
    container_name: postgres115  # 容器名称 ，不是必填
    restart: always  # 启动方式
    networks:
      - postgresql   # 指定的网络
    ports:
      - 5432:5432
    environment:       # 环境变量  ** 重点
      TZ:Asia/Shanghai
      POSTGRES_USER:postgres
      POSTGRES_PASSWORD:postgres_pass

    volumes:
      - ./data:/var/lib/postgresql/data

  pgadmin:
    image: dpage/pgadmin4
    container_name: pgadmin
    restart: unless-stopped
    ports:
      - 5431:80
    environment:   # 环境变量是 服务编排  重点
```

```
      PGADMIN_DEFAULT_EMAIL:san@q.com
      PGADMIN_DEFAULT_PASSWORD:PGadmin
    depends_on:    # 启动依赖 重点
      - postgres

    networks:
      - postgresql

volumes:
  data:

networks:
  postgresql:
    external:
      name: postgresql
```

最核心的是`services`

其实是服务之间的关系

二、编排容器

接口项目容器 + 测试框架容器 = 测试结果容器

1. 构建镜像

编排的前提是有镜像

在实践中，是由开发团队完成

提前给搭建构建了一个接口项目：`api:v2`

启动

```
docker run \
--name api_v2 \
--rm    \
-p 80:80 \
-e TEST=1 \
api:v2
```

```
docker run --name api_v2 --rm    -p 80:80 -e TEST=1 api:v2
```

问题：命令有点复杂，参数有点多，一不小心就会出错

2. docker-compose

文件化的配置

```yaml
version: '3'  # 指定 docker-compose版本
services:
  api_server:    # 服务名称
    image: api:v2  # 镜像
    environment:       # 环境变量  ** 重点
      - DTABASE_URL='mysql://api_v2:api_v2pass@192.168.56.103/api_v2'
      - TEST=1
    ports:
      - "80:80"
```

3. 构建测试镜像

有镜像，才有有容器，有容器，才能和其他微服务一同启动，完成测试!

打包的测试镜像：`beifan`

```
>beifan --help
Usage: beifan [OPTIONS] COMMAND [ARGS]...

  BeiFan : 基于 yaml + pytest 的接口自动化测试框架

  默认执行 run 命令

Options:
  --help  Show this message and exit.

Commands:
  report  调用allure，生成HTML报告
  run     启动pytest，执行测试用例
  start   根据har文件，生成yaml测试用例
```

将一个复杂强大的测试框架，打包成镜像，然后通过启动的方式变成容器

4. 执行容器间测试

启动

```
docker-compose up
```

自动停止：

当测试结束的时候，所有的服务全部关闭

```
docker-compose up --abort-on-container-exit
```

测试报告：

使用allure生成HTML测试报告

allure是一个java程序，需要java

我们花了三天的时间学习docker，

再问一次：当我们需要**某个环境，来做某事**的，该怎么办？

再开一个docker就好了

生成测试报告URL

不要掉进docker蜜罐里

技术体系掌握之后：

- 一线城市： 20k 问题不打
- 非一线城市（保守一点）：15k左右问题不大

跟着北凡老师一起学习，成为测试大佬，难道15k 甚至20k的薪资