Lesson 4. Multi-container application

Overview of Docker Compose

Docker Compose is a tool for defining and running multi-container Docker applications.

Features:

- Multiple isolated environments on a single host
- Preserve volume data when containers are created
- Only recreate containers that have changed
- Variables and moving a composition between environments

Use cases:

- Development environments
- Automated testing environments
- Single host deployments

> Read more on https://docs.docker.com/compose/overview/.

”Hits” application

Project structure from homework-3

```
.
├── Dockerfile
├── Dockerfile.at
├── README.txt
├── hits
│   └── app.py
├── requirements.txt
├── test-requirements.txt
└── tests
    ├── conftest.py
    └── test_app.py
```

Run “Hits”

```
# build
docker build -t hits .
# start
docker network create --driver=bridge hits
docker run -d --name redis --net=hits redis:alpine
docker run -d -p 5000:5000 --name hits --net=hits hits
# use
curl localhost:5000
# clean
docker stop redis hits
docker rm redis hits
docker network rm hits
```

Docker Compose

docker-compose.yml file

Sample file

```
# docker-compose.yml
# docker-compose.yaml

version: '3.7' # docker engine 18.06.0+
```

Major versions:

- `1` - https://docs.docker.com/compose/compose-file/compose-file-v1/
- `2` - https://docs.docker.com/compose/compose-file/compose-file-v2/
- `3` - https://docs.docker.com/compose/compose-file/

`docker-compose` CLI command

```
docker-compose --help
docker-compose up --help
docker-compose config --help
```

“Hits” in Docker Compose

Basic services

Task: run "Hits" application in the compose

`docker-compose.yml` file

```
version: '3.7'
services:
  hits:
    build: .
    image: hits:lesson-4
    ports:
     - 5000:5000
    depends_on:
      - redis
  redis:
    image: redis:alpine
```

Usage:

```
# build and run
docker-compose up --build -d
# demonstrate
docker-compose ps
docker network ls
# work
curl localhost:5000
docker-compose logs
# destroy
docker-compose down
```

> Read more on https://docs.docker.com/compose/compose-file/#service-configuration-reference

Services & Volumes

Task: save counter from Redis

`docker-compose.yml` file

```
version: '3.7'
services:
  hits:
    build: .
    image: hits:lesson-4
    ports:
     - 5000:5000
    depends_on:
      - redis
  redis:
    image: redis:alpine
    volumes:
     - redis-data:/data
volumes:
  redis-data:
    driver: local
```

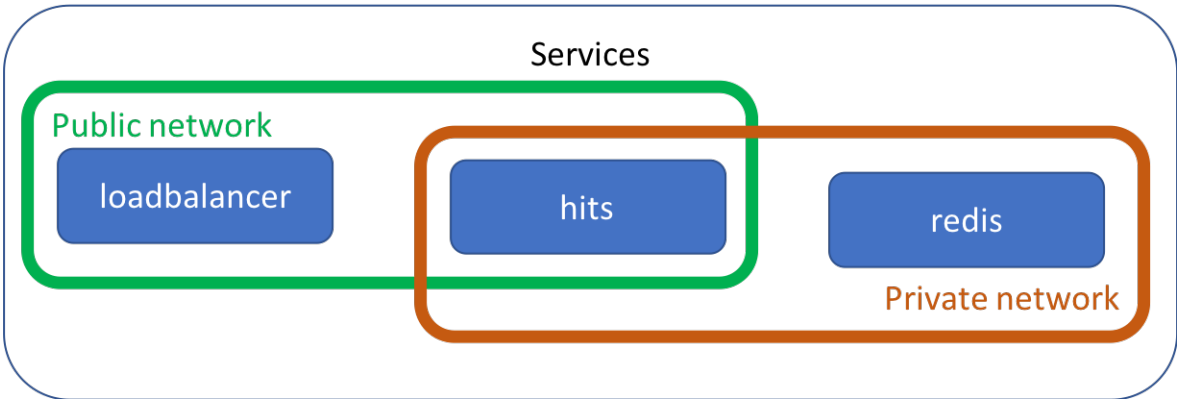
Usage:

```
# run
docker-compose up -d
docker volume ls | grep redis
# do some work
curl localhost:5000
# re-run
docker-compose down
docker-compose up -d
# check counter
curl localhost:5000
```

> Read more on https://docs.docker.com/compose/compose-file/#volume-configuration-reference

Services & Networks

Task: protect Redis from external access



`docker-compose.yml` file

```
version: '3.7'
services:
  hits:
    build: .
    image: hits:lesson-4
    depends_on:
      - redis
    networks:
      - public
      - secret
  redis:
    image: redis:alpine
    volumes:
     - redis-data:/data
    networks:
     - secret
  loadbalancer:
    image: dockercloud/haproxy:latest
    links:
      - hits
    volumes:
      - /var/run/docker.sock:/var/run/docker.sock
    ports:
      - 8080:80
    networks:
      - public
volumes:
  redis-data:
    driver: local
networks:
  secret:
    driver: bridge
  public:
    driver: bridge
```

Usage:

```
docker-compose -p net up -d
docker network ls
docker network inspect net_secret
docker network inspect net_public
curl localhost:8080
docker-compose -p net down
```

> Read more on https://docs.docker.com/compose/compose-file/#network-configuration-reference and https://docs.docker.com/compose/networking/

Scaling

```
docker-compose up -d --scale hits=10
docker-compose ps
```

> If your container can be scaled, test it with 3 instances.

Docker Container Orchestration

Services:

- Docker Swarm
- Kubernetes
- DC/OS
- Google Container Engine
- The Amazon EC2 Container Service (ECS)
- Azure Container Service (ACS)

Please check out a Swarm example if you are interested in.

Homeworks

Please send the results of homeworks as an email.
Please use the following template:

- *Subject:* `[Docker] Homework 4`
- *To:* `trainer's email`
- *Body:* `your homework as a plain text - NO ATTACHMENTS!!!`

Homework 4.1 (mandatory)

There is a template of a `docker-compose.yml`:

```
version: '3.7'
services:
  hits:
    build: <relevant path>
    maybe custom Dockerfile name?
    image: <your tag>
    depends_on:
      - redis
    networks:
      - public
      - secret
  redis:
    image: redis:alpine
    volumes:
     - redis-data:/data
    networks:
     - secret
  loadbalancer:
    image: dockercloud/haproxy:latest
    links:
      - hits
    volumes:
      - /var/run/docker.sock:/var/run/docker.sock
    ports:
      - 8080:80
    networks:
      - public
volumes:
  redis-data:
    driver: local
networks:
  secret:
    driver: bridge
  public:
    driver: bridge
```

Add this template to your directory with homework-3. Update `hits` service whit relevant build information. Please use `hits:h4` suffix for image name. Please push the image with `docker-compose push`.

Then, you need to configure `docker-compose.yml` in a way, when the log file is shared across all 'hits' instances. To test it, scale `hits` service at least up to 3 instances, `curl localhost:8080` a couple of times, call `curl localhost:8080/logs`. You should see messages for all your received ids.

Please send `docker-compose.yml` for review.

Homework 4.2 (optional)

Using the output of homework 4.1, integrate automated tests as a separate service. Also, please push the image with tests (and use `hits:h4` suffix for image name). The tests have to call `loadbalancer` service instead of `hits`.

> Please read https://docs.docker.com/compose/startup-order/ before working on this task.

Please send `docker-compose.yml` for review.