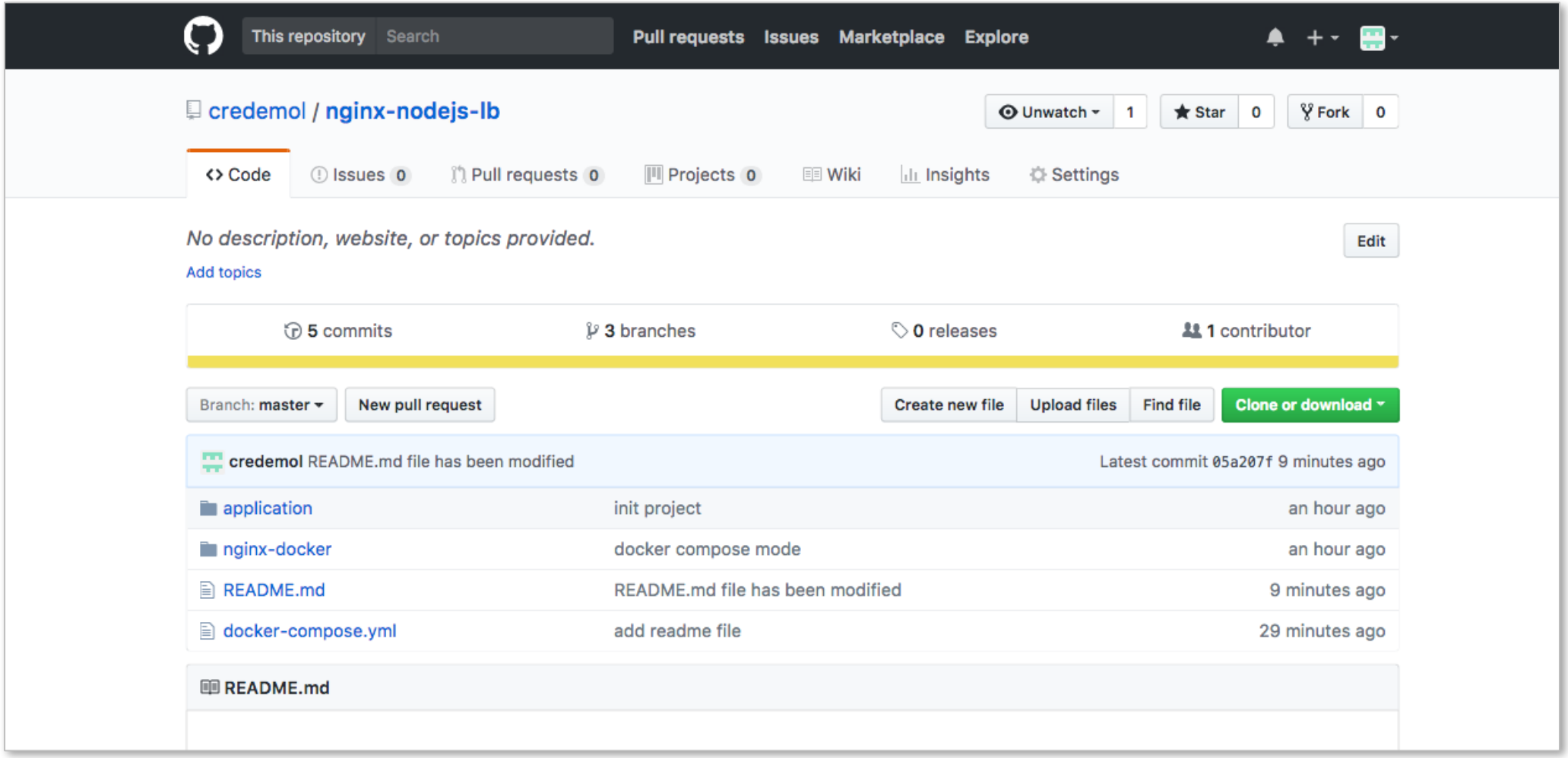
Load Balancing with Nginx and Docker

Younggyu kim (younggyu.kim@oracle.com, credemol@gmail.com)

Cloud Platform, Oracle Korea

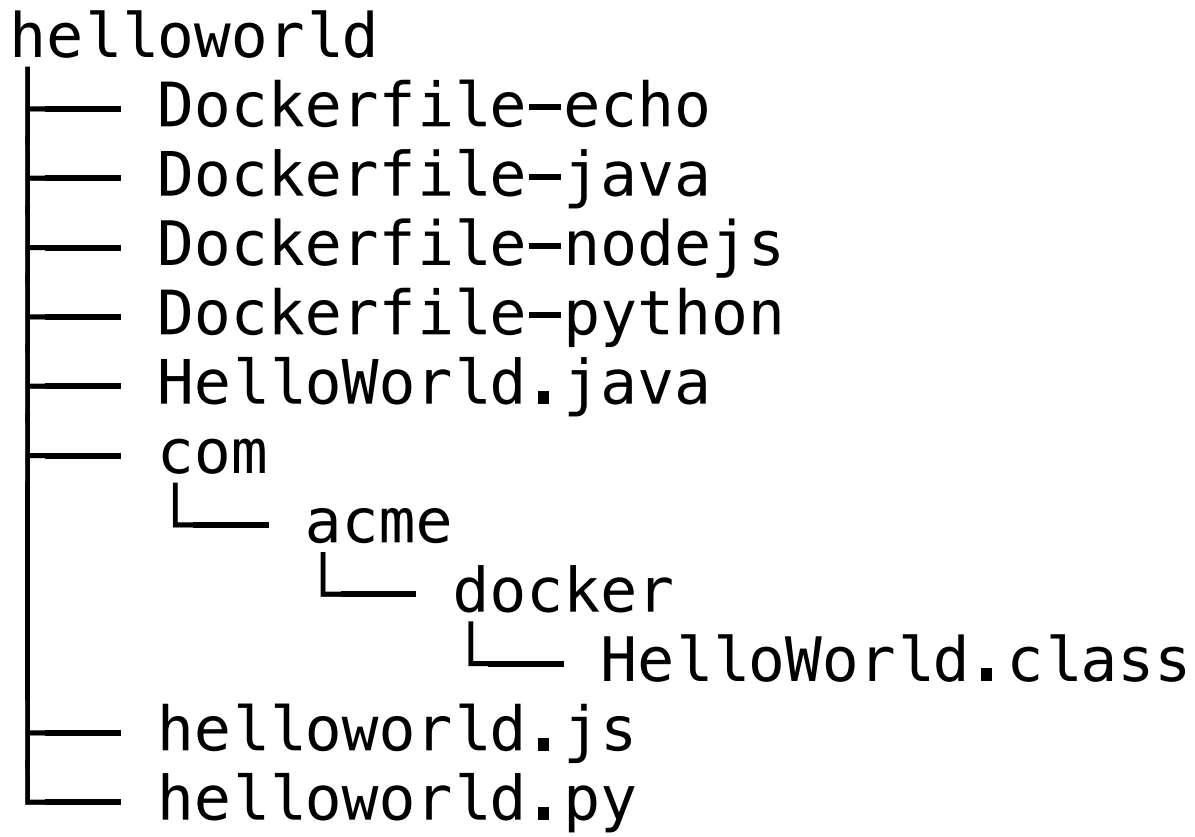
https://github.com/credemol/nginx-nodejs-lb



HelloWorld

A Docker Tutorial for Beginners

helloworld file structure



Create directories

```
$ mkdir helloworld
$ cd helloworld
```

Helloworld- with echo command

```
$ vi Dockerfile-echo
```

```
FROM ubuntu

CMD echo "[ECHO] Hello World! It is " $(date)
```

```
$ docker build -t helloworld-echo -f Dockerfile-echo .
$ docker image ls

$ docker run -it helloworld-echo                    (← Interactive mode)

$ docker run -d --name=echo1 helloworld-echo        (← Background mode)
$ docker container logs echo1
```

Helloworld- with Java

```
$ vi HelloWorld.java
```

```
package com.acme.docker;

public class HelloWorld {
    public static void main(String[] args) {
        System.out.println("[JAVA] Hello World. It is " + new java.util.Date());
    }
}
```

```
$ javac -d . HelloWorld.java
$ java -cp . com.acme.docker.HelloWorld
```

Helloworld- with Java

```
$ vi Dockerfile-java
```

```
FROM openjdk:8-jdk-alpine
RUN mkdir -p /usr/src/app
COPY ./com /usr/src/app/com
CMD java -cp /usr/src/app com.acme.docker.HelloWorld
```

```
$ docker build -t helloworld-java -f Dockerfile-java .
$ docker image ls

$ docker run -it helloworld-java                    (← Interactive mode)

$ docker run -d --name=java1 helloworld-java        (← Background mode)
$ docker container logs java1
```

Helloworld- with Java(compile java on the image)

```
$ vi Dockerfile-java2
```

```
FROM openjdk:8-jdk-alpine
RUN mkdir -p /usr/src/app
COPY HelloWorld.java /usr/src/app
RUN javac -d /usr/src/app /usr/src/app/HelloWorld.java
CMD java -cp /usr/src/app com.acme.docker.HelloWorld
```

```
$ docker build -t helloworld-java2 -f Dockerfile-java2 .
$ docker image ls

$ docker run -it helloworld-java2                    (← Interactive mode)

$ docker run -d --name=java2 helloworld-java2        (← Background mode)
$ docker container logs java2
```

Helloworld- with Node.JS

```
$ vi helloworld.js
```

```
var today = new Date();
console.log(`[Node.JS] Hello World. It is ${today}`);
```

```
$ node helloworld
```

Helloworld- with Node.JS

```
$ vi Dockerfile-nodejs
```

```
FROM node
RUN mkdir -p /usr/src/app
COPY helloworld.js /usr/src/app
CMD node /usr/src/app/helloworld
```

```
$ docker build -t helloworld-nodejs -f Dockerfile-nodejs .
$ docker image ls

$ docker run -it helloworld-nodejs                    (← Interactive mode)

$ docker run -d --name=nodejs1 helloworld-nodejs        (← Background mode)
$ docker container logs nodejs1
```

Helloworld- with Python

```
$ vi helloworld.py
```

```
from datetime import datetime
now = datetime.now()

print("[Python] Hello World! It is", now)
```

```
$ python3 helloworld.py
```

Helloworld- with Python

```
$ vi Dockerfile-python
```

```
FROM python:3.6.3-alpine3.6
RUN mkdir -p /usr/src/app
COPY helloworld.py /usr/src/app
CMD python /usr/src/app/helloworld.py
```

```
$ docker build -t helloworld-python -f Dockerfile-python .
$ docker image ls

$ docker run -it helloworld-python                    (← Interactive mode)

$ docker run -d --name=python1 helloworld-python        (← Background mode)
$ docker container logs python1
```

Load Balancing with Nginx and Docker

A Docker Tutorial for Beginners

File Tree Structure (nginx-nodejs-lb: working dir)

```
nginx-nodejs-lb/
├── application
│   ├── Dockerfile
│   └── index.js
├── docker-compose.yml
└── nginx-docker
    ├── Dockerfile
    ├── nginx-each-container.conf
    └── nginx.conf
```

Create directories

```
$ mkdir nginx-nodejs-lb
$ cd nginx-nodejs-lb

$ mkdir application
$ mkdir nginx-docker
```

application/index.js

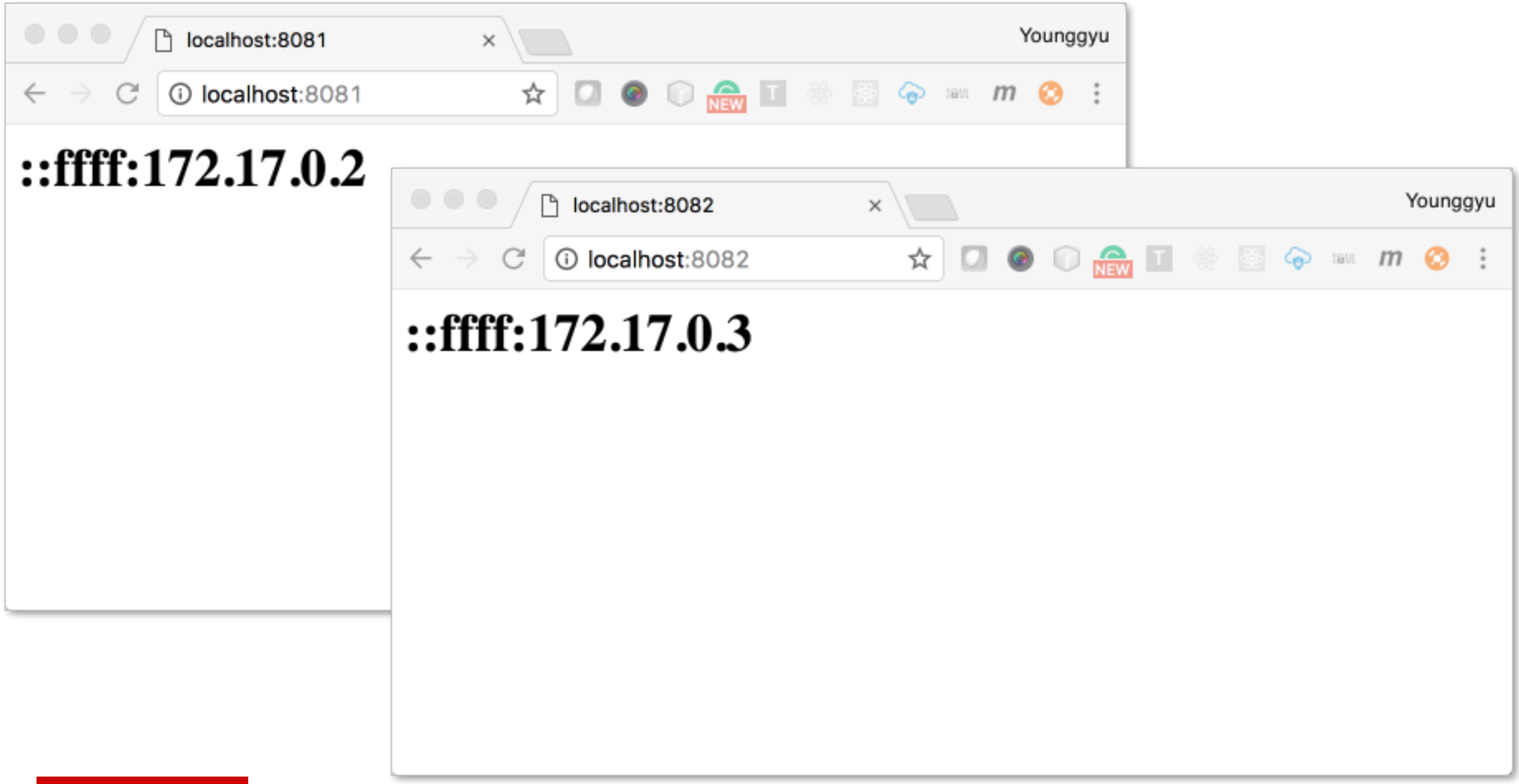
```
var http = require('http');
var fs = require('fs');

http.createServer(function (req, res) {
  res.writeHead(200, {'Content-Type': 'text/html'});
  res.end(`<h1>${req.connection.localAddress}</h1>`);
}).listen(8080);
```

application/Dockerfile

```
FROM node
RUN mkdir -p /usr/src/app
COPY index.js /usr/src/app
EXPOSE 8080
CMD [ "node", "/usr/src/app/index" ]
```

application: docker build

```
$ docker build -t load-balanced-app .

$ docker image ls

$ docker run --name=app_1 -p 8081:8080 -d load-balanced-app
$ docker run --name=app_2 -p 8082:8080 -d load-balanced-app
```



nginx-docker/nginx.conf

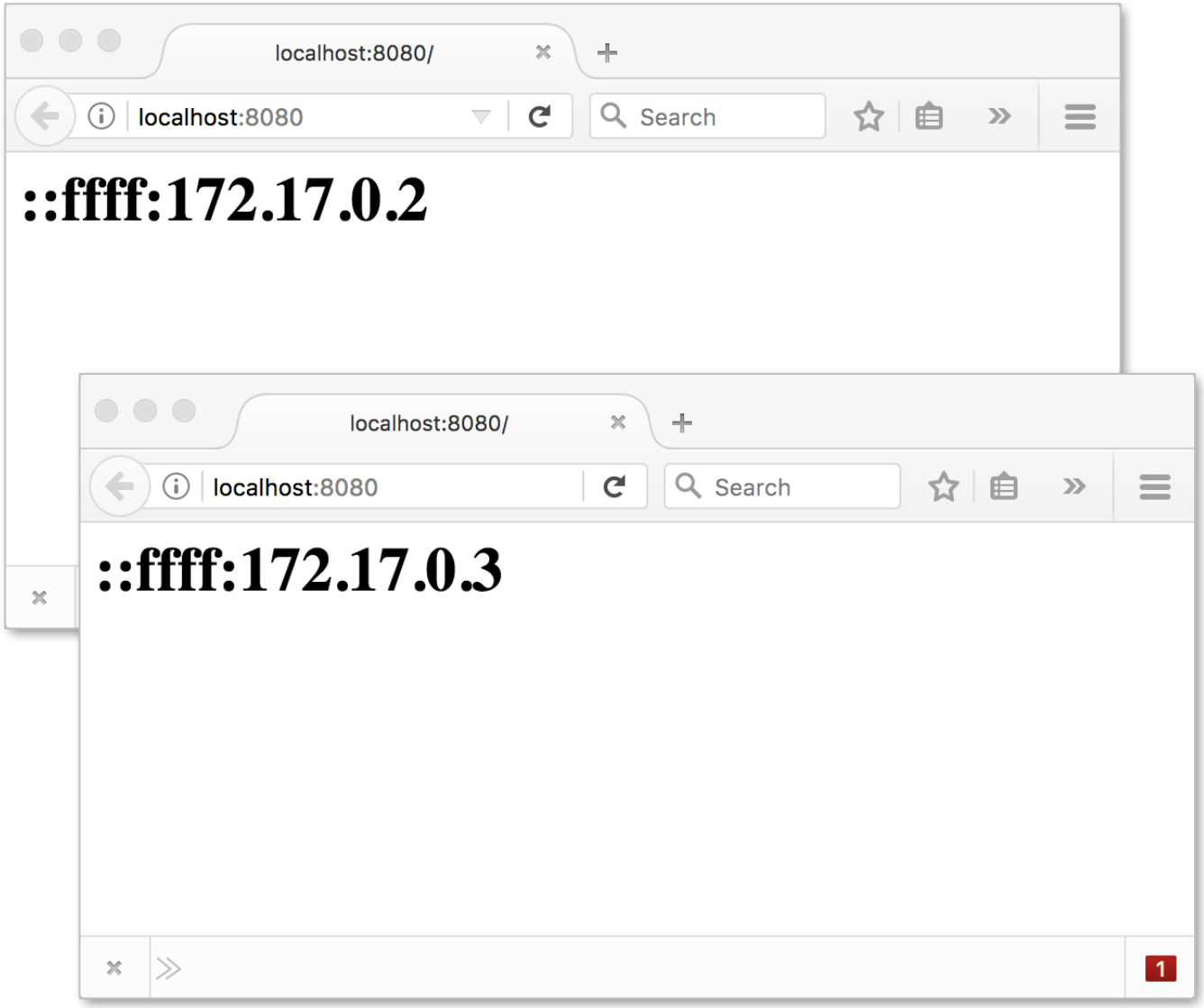
```
upstream my-app {
    server 172.17.0.1:8081 weight=1;
    server 172.17.0.1:8082 weight=1;
}

server {
    location / {
        proxy_pass http://my-app;
    }
}
```

nginx-docker/Dockerfile

```
FROM nginx
RUN rm /etc/nginx/conf.d/default.conf
COPY nginx.conf /etc/nginx/conf.d/default.conf
```

nginx: docker build

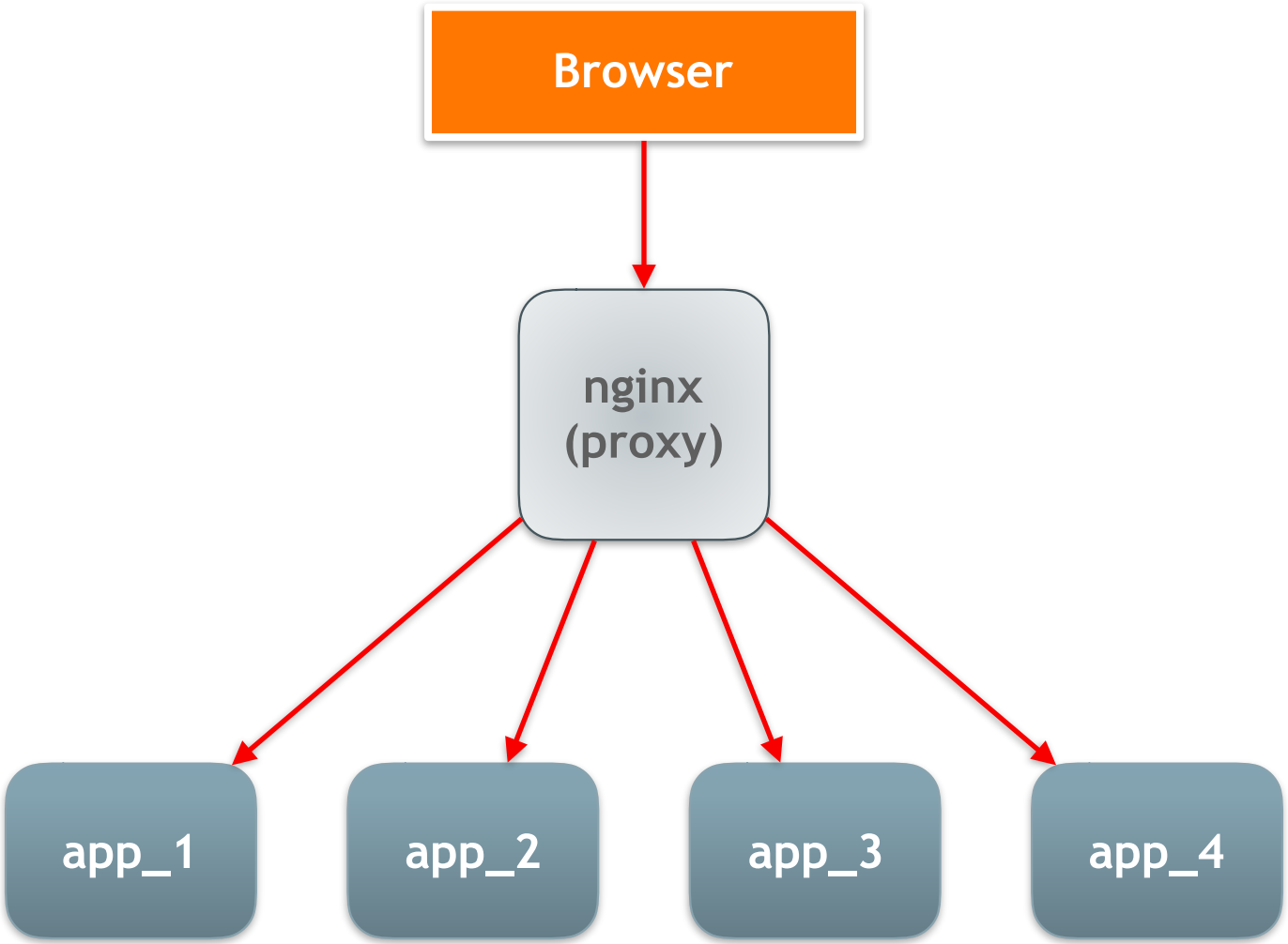
```
$ docker build -t load-balance-nginx .

$ docker image ls

$ docker run --name=lb-nginx -p 8080:80 -d load-balance-nginx
```



※ Docker Toolbox User

1. Open Docker Quickstart Terminal
2. run docker-machine ls
3. change localhost into ip address allocated to the docker machine

Docker-Compose



clean up: stop containers, rm containers, rm images

```
$ docker container stop $(docker container ls -q)
$ docker container ls
$ docker container ls -a

$ docker container rm $(docker container ls -qa)
$ docker container ls -a

$ docker image rm $(docker image ls -q)
$ docker image ls
```

nginx-docker/nginx.conf

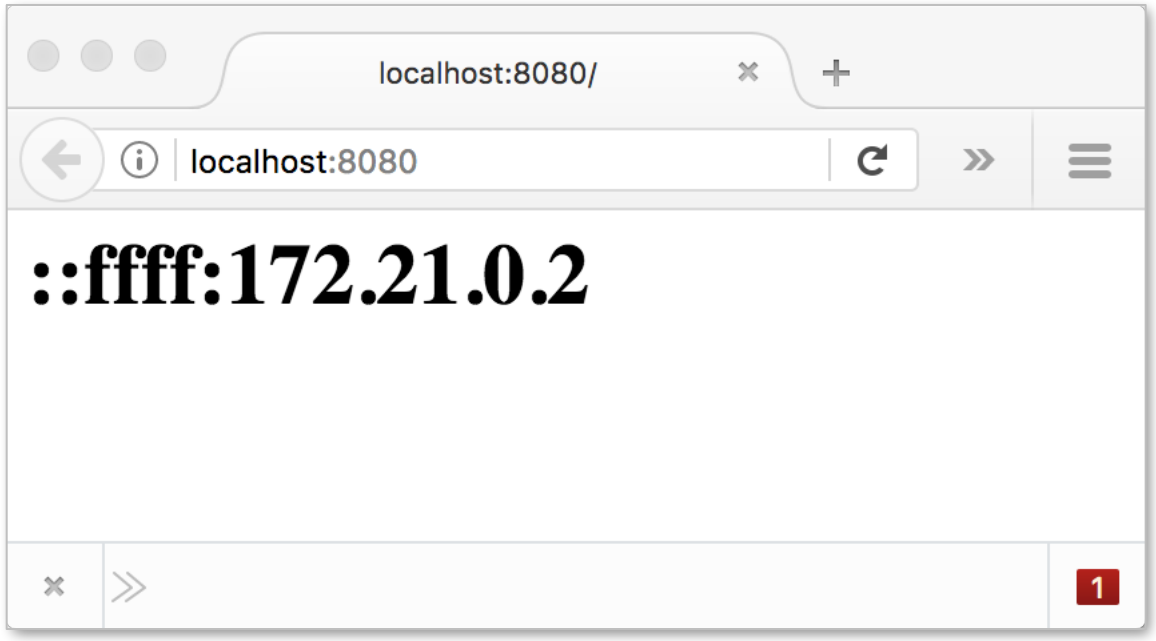
```
upstream my-app {
  server nginxnodejslb_app_1:8080 weight=1;
  server nginxnodejslb_app_2:8080 weight=1;
}

server {
  location / {
    proxy_pass http://my-app;
  }
}
```

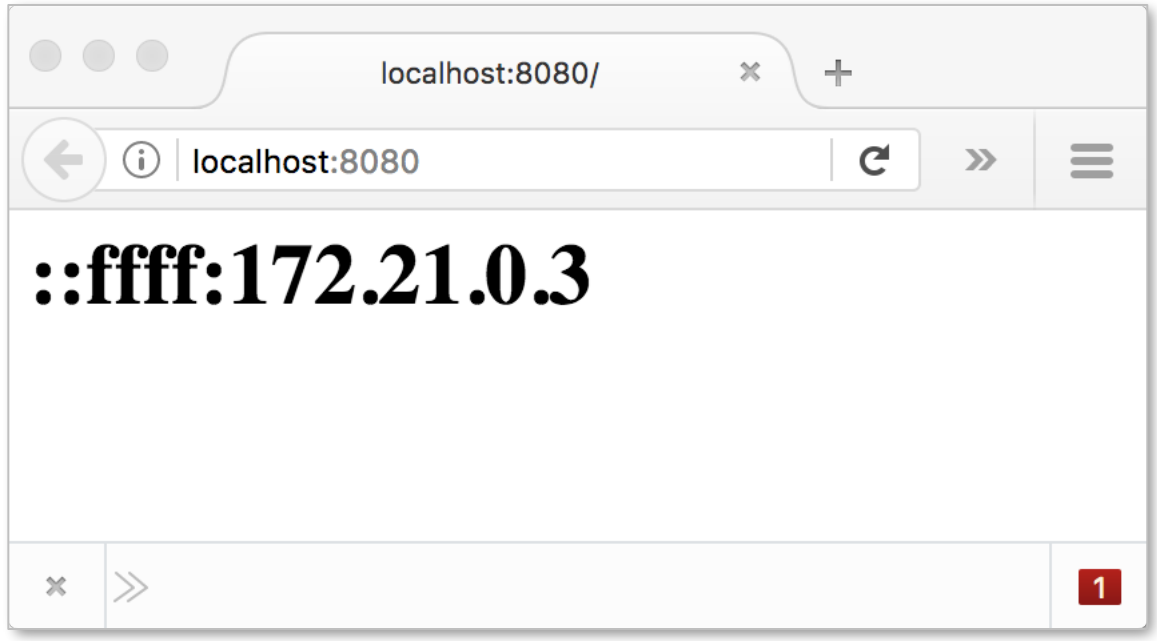
docker-compose.yml

```
version: '2'

services:
  app:
    build:
      context: ./application
      dockerfile: Dockerfile
    expose:
      - “8080"

  proxy:
    build:
      context: ./nginx-docker
      dockerfile: Dockerfile
    ports:
      - "8080:80"
    links:
      - app
```

docker compose build and up

```
$ docker-compose build

$ docker image ls

$ docker-compose up --scale app=2 -d

$ docker container ls
```





docker compose down

```
$ docker-compose down

$ docker container ls -a

$ docker image ls
```

Resources

https://auth0.com/blog/load-balancing-nodejs-applications-with-nginx-and-docker/

https://www.sep.com/sep-blog/2017/02/28/load-balancing-with-nginx-and-docker/


ORACLE®