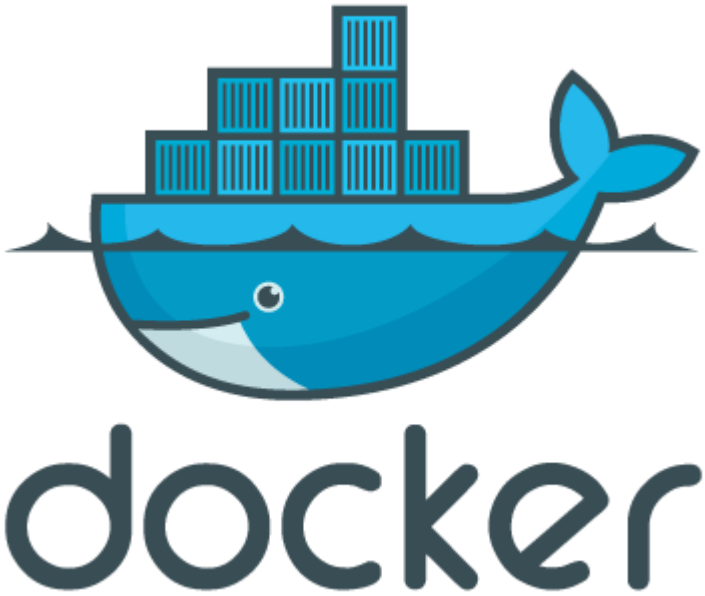

Docker

Table des matières

Qu'est-ce que Docker ?

[WORK IN PROGRESS]

Commandes

[WORK IN PROGRESS]

Créer un container ReactJs

Admettons que nous ayons une application qui se décompose ainsi

```
app
|-- node_modules/
|-- src/
|-- public/
|-- package.json
|-- package-lock.json
|-- yarn.lock
|-- README.md
```

Créer un Dockerfile

```
# app/Dockerfile

# pull official base image
FROM node:16.13.1-alpine

# set working directory
WORKDIR /app

# add `/app/node_modules/.bin` to $PATH
ENV PATH /app/node_modules/.bin:$PATH

# install app dependencies
COPY package.json ./
COPY package-lock.json ./
RUN npm install

# add app
COPY . ./

# start app
CMD ["npm", "start"]
```

Créer un .dockerignore

```
# app/.dockerignore

node_modules
build
.dockerignore
Dockerfile
```

Créer une image Docker

```
docker build -t reactjs_app:dev .
```

Créer et lancer un container de l'image créée

```
docker run \
-it \
--rm \
-v ${PWD} \
-v /app/node_modules \
-p 3001:3000 \
-e CHOKIDAR_USEPOLLING=true \
reactjs_app:dev
```

Explications

- [ ] `docker run` créer et lance une nouvel instance d'un container à partir de l'image `reactjs_app` qu'on vient de créer
- [ ] `-it` lance le container en "interactive mode". le mode intéractif permet d'éviter que le container quitte après avoir démarré à cause de `react-scripts`
- [ ] `--rm` suppprime le container et le volume après avoir quitté le container
- [ ] `-v ${PWD}` monte le code dans le container
- [ ] `-v /app/node_modules` afin d'utiliser la version du container de "node_modules"
- [ ] `-p 3001:3000` permet de faire pointer le port 3000 sur le port 3001 de l'hôte
- [ ] `-e CHOKIDAR_USEPOLLING=true` permet la mise en place du méchanisme de "polling" via `chokidar` qui permet le "hot-reloading"

Accéder à l'application sur le localhost

`http://localhost:3001/`

Créer un docker-compose

```yaml
# app/docker-compose.yaml

# version of docker-composer
version: '3.8'

services:
  app:
    container_name: reactjs_app
    build:
      context: .
      dockerfile: Dockerfile
    volumes:
      - '.:/app'
      - '/app/node_modules'
    ports:
      - 3001:3000
    environment:
      - CHOKIDAR_USEPOLLING=true
```

Créer l'image et lancer le container

```
docker-compose up -d --build
```

Arrêter l'exécution de l'application

```
docker-compose stop
```