

IoT Platform: Dockerization

- Date: 02-06-2024
- Dockerization for [MERN Application](#)

1. Materials

- Inspired from:
 1. [\(373\) Learn Docker - DevOps with Node.js & Express - YouTube](#) (Recommend)
 2. [Deploying a MERN Application \(with Docker, Atlas, and Digital Ocean!\)](#) (youtube.com)

2. First Demo: EggJS with Docker

- NodeJS framework: [egg - Born to build better enterprise frameworks and apps - Egg \(eggjs.org\)](#)
- Presquisite: Node.js Runtime: 8.x or newer: [Node.js — Download Node.js® \(nodejs.org\)](#)
- VS Code Plugin: [Docker - Visual Studio Marketplace](#)

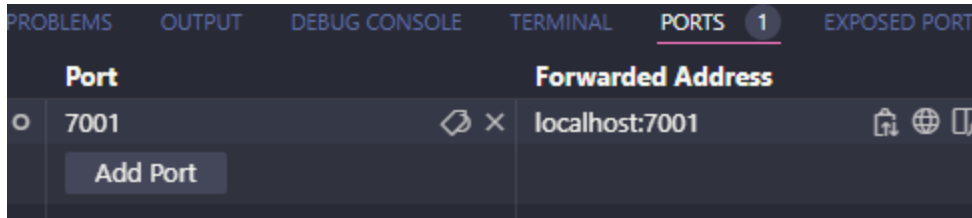
2.1. Create EggJS Project

- Follow: [Quick Start - Egg \(eggjs.org\)](#)
- Install initializer (only do it once): `qtiot@qtiot:~/egg-example$ npm i -g egg-init`
- `qtiot@qtiot:~/egg-example$ npm init egg --type=simple`
 - Choose the type: “simple - Simple egg app boilerplate”
- `qtiot@qtiot:~/egg-example$ npm i`
- `qtiot@qtiot:~/egg-example$ npm run dev`

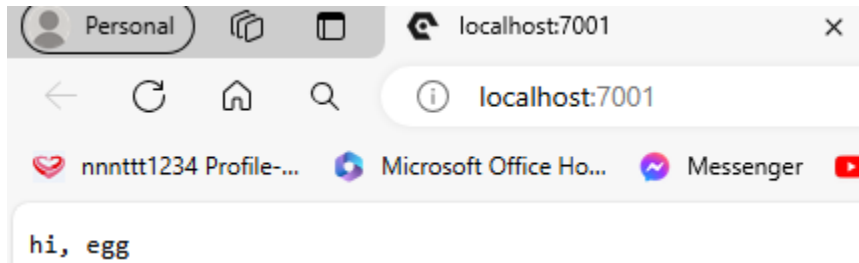
```
qtiot@qtiot:~/egg-example$ npm run dev
> example@1.0.0 dev
> egg-bin dev

[egg-ts-helper] create typings/app/controller/index.d.ts (1ms)
[egg-ts-helper] create typings/config/index.d.ts (10ms)
[egg-ts-helper] create typings/config/plugin.d.ts (0ms)
[egg-ts-helper] create typings/app/index.d.ts (0ms)
2024-06-02 08:26:31,073 INFO 3031 [master] node version v20.14.0
2024-06-02 08:26:31,074 INFO 3031 [master] egg version 3.23.0
2024-06-02 08:26:31,422 INFO 3031 [master] agent_worker#1:3050 started (347ms)
2024-06-02 08:26:31,779 INFO 3031 [master] egg started on http://127.0.0.1:7001 (705ms)
```

-
- The EggJS has run in the port 7001 of the host machine



- Should forward the port 7001 from the Virtual machine to show the result.



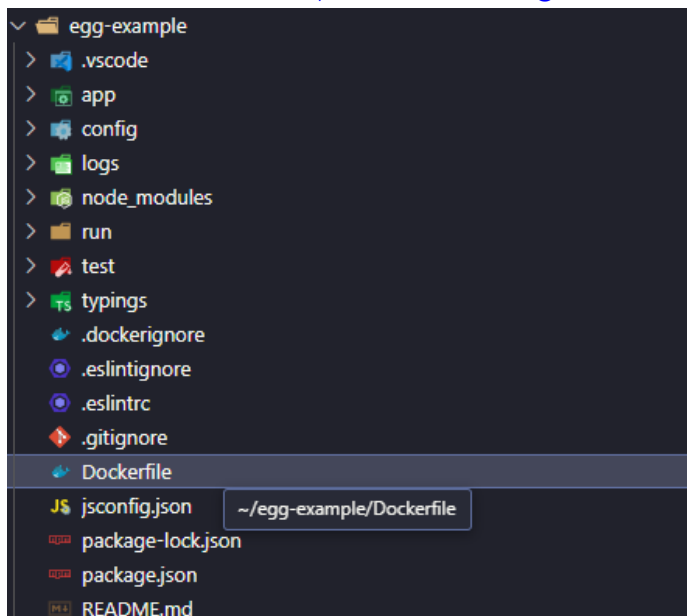
- Result when open the port 7001 of the localhost.

2.2. Dockerize the EggJS app.

- Using [Dockerization](#) technique.

2.3. Dockerization Step 1: Create Dockerfile

- [What Is a Dockerfile? | Cloudbees Blog](#)



- Create the “Dockerfile” in the project root directory.

```
egg-example > Dockerfile > ...
1  FROM node:20.14.0-slim
2
3  WORKDIR /app/egg-example
4
5  COPY . .
6
7  RUN npm i
8
9  EXPOSE 7001
10
11 CMD ["npm", "run", "dev"]
```

- This file simply illustrates how to build a [Docker image](#).
- Line 1: using the base image containing NodeJS: “[node:20.14.0-slim](#)”
- Line 3: Choose the directory “/app/egg-example” in the docker image. This command acts as “cd “/app/egg-example” in a host machine.
- Line 5: copy all file from the project root directory (the first dot) to the working directory “/app/egg-example” of the image (the second dot).
- Line 7: install node packages.
- Line 9: Open port 7001 of “**a container**”.
- Line 11: run the EggJS “npm run dev” in “**a container**”.

2.4. Build a Dockerfile Image

```
egg-example > .dockerignore
1 logs/
2 npm-debug.log
3 yarn-error.log
4 node_modules/
5 *-lock.json
6 *-lock.yaml
7 yarn.lock
8 coverage/
9 .idea/
10 run/
11 .DS_Store
12 *.sw*
13 *.un~
14 typings/
15 .nyc_output/
```

- - First, create a file “.dockerignore” to list ignore files/folders to prevent them from being copiable from the command “COPY . .” in the Dockerfile.
 - List the “node_modules” folder to prevent it from being copied to the container.
- qtiot@qtiot:~/egg-example\$ **docker build -t egg-example .**
 - [docker build | Docker Docs](#)
 - [How to Build a Docker Image from Dockerfile | Cherry Servers](#)

```
qtiot@qtiot:~/egg-example$ docker build -t egg-example .
[+] Building 2.9s (9/9) FINISHED
=> [internal] load build definition from Dockerfile
=> => transferring dockerfile: 146B
=> [internal] load metadata for docker.io/library/node:20.14.0-slim
=> [internal] load .dockerignore
=> => transferring context: 190B
=> [1/4] FROM docker.io/library/node:20.14.0-slim@sha256:a16301294ba66d2ad22d3beded4a52720f96ab208c1db0973c034d0127a4ccb0
=> [internal] load build context
=> => transferring context: 723B
=> CACHED [2/4] WORKDIR /app/egg-example
=> CACHED [3/4] COPY . .
=> CACHED [4/4] RUN npm i
=> exporting to image
=> => exporting layers
=> => writing image sha256:3ee84302a0b98f7532a92fe4d68990da77f38e2d345f5ac860cf959dd2157eb8
=> => naming to docker.io/library/egg-example
```

- - Build process.

```

qtiot@qtiot:~/egg-example$ docker image ls
REPOSITORY    TAG       IMAGE ID       CREATED        SIZE
egg-example    latest    3ee84302a0b9   9 seconds ago  706MB

```

- After building the image, use the command “docker image ls” to show the built image.
- qtiot@qtiot:~\$ **docker run --name egg-example -p 7001:7001 -d egg-example**
 - Run a docker container with detached mode: [Run your Go image as a container | Docker Docs](#)

```

qtiot@qtiot:~/egg-example$ docker logs -f egg-example

> example@1.0.0 dev
> egg-bin dev

[egg-ts-helper] create typings/app/controller/index.d.ts (3ms)
[egg-ts-helper] create typings/config/index.d.ts (10ms)
[egg-ts-helper] create typings/config/plugin.d.ts (0ms)
[egg-ts-helper] create typings/app/index.d.ts (0ms)
2024-06-02 12:45:04,356 INFO 45 [master] node version v20.14.0
2024-06-02 12:45:04,356 INFO 45 [master] egg version 3.23.0
2024-06-02 12:45:04,710 INFO 45 [master] agent_worker#1:52 started (352ms)
2024-06-02 12:45:05,083 INFO 45 [master] egg started on http://127.0.0.1:7001 (727ms)

```

- View the docker container’s logs through the command:
“qtiot@qtiot:~/egg-example\$ **docker logs -f egg-example**”.

The screenshot shows the Docker Desktop interface. On the left, the 'CONTAINERS' panel shows the 'egg-example' container, which is up and running for 30 minutes. Below it, the 'Files' panel shows the contents of the container, including 'app', 'config', 'logs', 'node_modules', 'run', 'test', 'typings', and various configuration files like '.dockerignore', '.eslintrc', '.gitignore', 'Dockerfile', 'jsconfig.json', and 'package-lock.json'. On the right, the 'Dockerfile' is displayed with the following content:

```

1 FROM node:20.14.0-slim
2
3 WORKDIR /app/egg-example
4
5 COPY . .
6
7 RUN npm i
8
9 EXPOSE 7001
10
11 CMD ["npm", "run", "dev"]

```

- Through the Docker plugin, we can see what sources in “/app/egg-example” of the executed container.

```
qtiot@qtiot:~/egg-example$ docker ps
CONTAINER ID   IMAGE      COMMAND                  CREATED        STATUS        PORTS                               NAMES
861c2c1e6a34   egg-example "docker-entrypoint.s..." 2 minutes ago  Up 2 minutes  0.0.0.0:7001->7001/tcp, :::7001->7001/tcp  egg-example
```

- Show information of executed containers through the command
“qtiot@qtiot:~/egg-example\$ **docker ps**”.
- GitHub source for this example: [DESLab-Resources/egg-example \(github.com\)](https://github.com/DESLab-Resources/egg-example)