

Automated Continious Integration with Travis

John Ghatas

Contents

1	Project definition	3
2	Running the application	3
3	Exporting to docker	4
4	Configuring TravisCI	5

1 Project definition

Goal This project was pulled from a Lynda tutorial teaching the basics of Docker, the Docker container created runs an express backend using MongoDB as the database server.

2 Running the application

The first order of business was to ensure that the application was able to run locally before deploying it in a Docker container. To do this we ran the following commands in the main directory:

```
$ npm install  
$ npm start
```

To start the backend locally **MongoDB** is required to have a local install. An example of the backen is shown in **Figure 1**

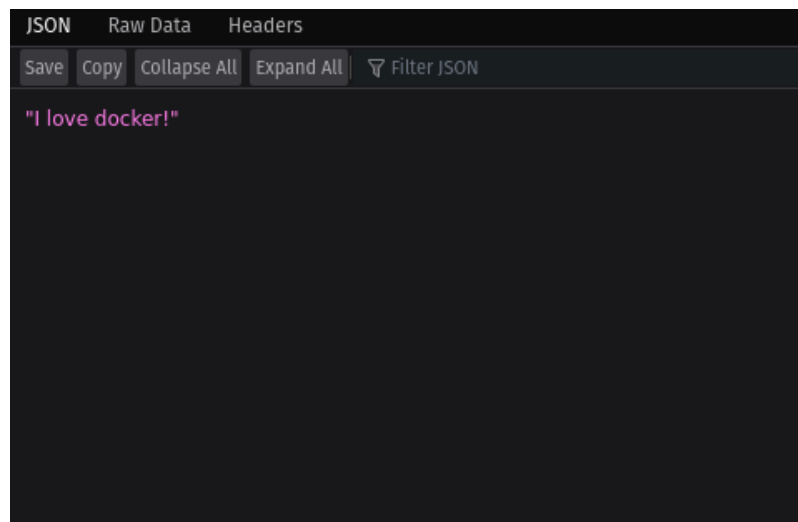


Fig. 1: The root route of the backend running

3 Exporting to docker

The next step was to export the image to docker locally, to ensure the setup was running before deploying the automation of the Docker image to the TravisCI servers. To build a Docker image, I had to create a Dockerfile specifying the steps that needed to be taken to deploy the Express backend in a Docker container. The Dockerfile is shown in **Figure 2**.

```
1 FROM node
2 RUN mkdir /usr/src/app
3 WORKDIR /usr/src/app
4 ENV PATH /usr/src/app/node_modules/.bin:$PATH
5 COPY package.json /usr/src/app
6
7 RUN npm install
8
9 COPY . /usr/src/app
10
11 EXPOSE 4000
12 CMD [ "npm", "start" ]
13
```

Fig. 2: The dockerfile containing the instructions for building the container

To prevent any unnecessary files being transferred to the Docker container, a **.dockerignore** file is defined. We ignored the Node Modules folder and the debug log from the Node Package Manager, thus reducing the total size of the container. Now we could build the image with following command (assuming you are in the root folder of the repository):

```
$ docker build . -t [domain]/[name]
$ docker run -p [local port]:[docker port]
-d [domain]/[name]
```

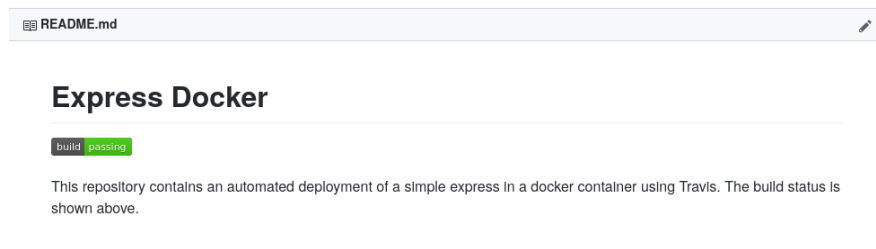
4 Configuring TravisCI

Now that I ensured the backend runs locally, we were ready to automate the building of the docker image on Travis. To configure Travis, a `.travis.yml` file was created to specify the instructions for the CI tool. In **Figure 3** the configuration is shown of the travis file.

```
1  sudo: required
2  services:
3    - docker
4
5  script:
6    - docker build -t johnggh/node .
7    - docker images johnggh/node
8
```

Fig. 3: The configuration file of for the CI tool

The configuration file will be picked by the plugin TravisCI, that I installed on the repository automating the build. After the build is done the status will be updated in the README of the repository as can be seen in **Figure 4**.



The screenshot shows a GitHub repository page for a file named `README.md`. The title of the repository is "Express Docker". Below the title, there is a green badge that says "build passing". The main text of the README states: "This repository contains an automated deployment of a simple express in a docker container using Travis. The build status is shown above."

Fig. 4: The build status of the build of the Docker container