Pre-study distributed systems

Table of Contents

[Introduction 1](#_Toc494699960)

[Docker 1](#_Toc494699961)

[Dockerfile 1](#_Toc494699962)

[Docker-compose 2](#_Toc494699963)

[TravisCI 3](#_Toc494699964)

# Introduction

A distributed system is a way of solving a task by multiple computers. To help us with this we use Docker. Docker is used to create a container of the application. A Docker container is like a package, it can run anywhere. Docker-compose allows to combine multiple containers together. The whole system can be run at once.

The whole distributed system can be tested at once. The tests will be triggered every time someone commits. We use the free service TravisCI to do these tests. When the tests pass the whole distributed system will be deployed.

# Docker

## Dockerfile

Docker is a way to package an application. The container is also packaged with an Operating System. An example of such an OS is alphine. This Linux image is only 5MB big.

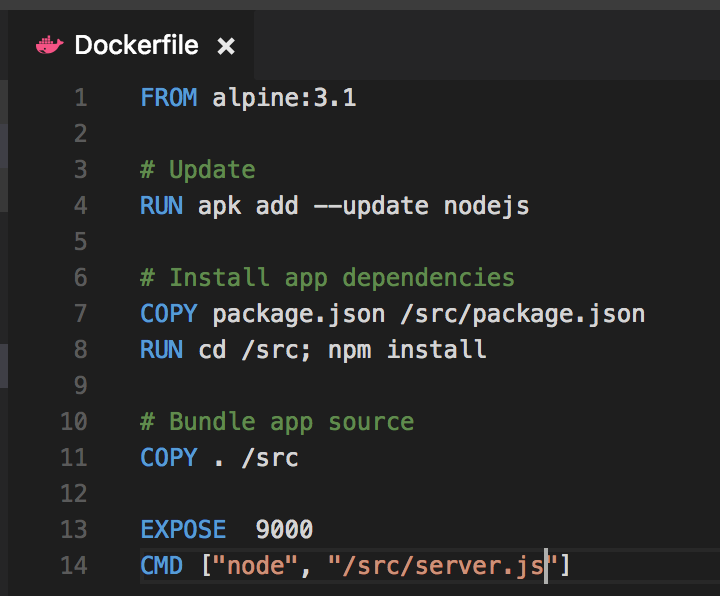


Figure 1 Example of Dockerfile

A container is configured inside the Dockerfile. Every time the container is built from scratch. Docker containers are portable and easy to share. Many containers are already publicly available. This make it run a database server.

## Docker-compose

Docker-compose is one of the ways to run containers together.

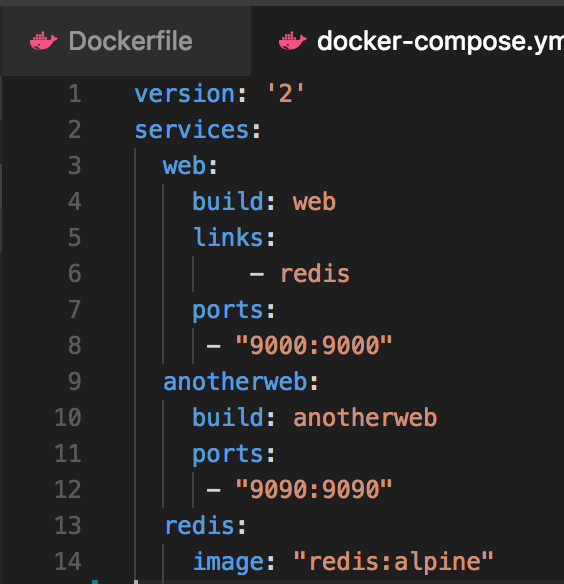


Figure 2 Example of docker-compose file

The containers are configured inside the docker-compose file. All containers can be run at once on a machine. It is also possible to publish the configuration to a cloud provider. Unfortunately, Azure have a high entry cost for using their service.

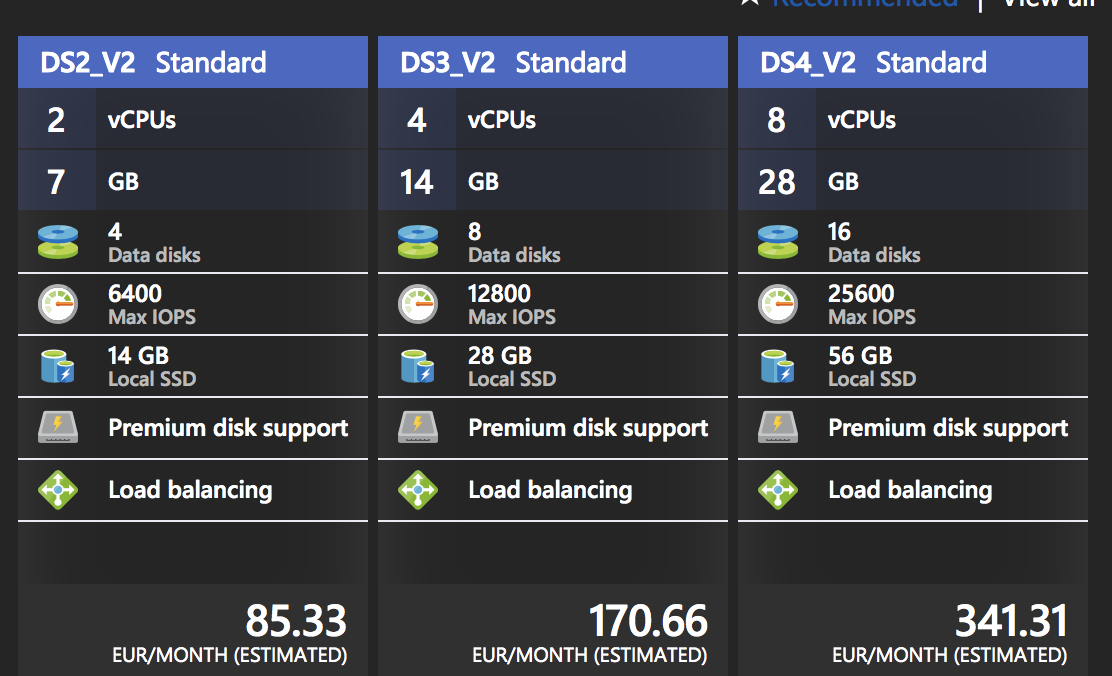


Figure 3 Azure pricing for container service

Fortunately, Docker is very portable. Without many efforts can the system be deployed to a cluster of Raspberry PI’s.

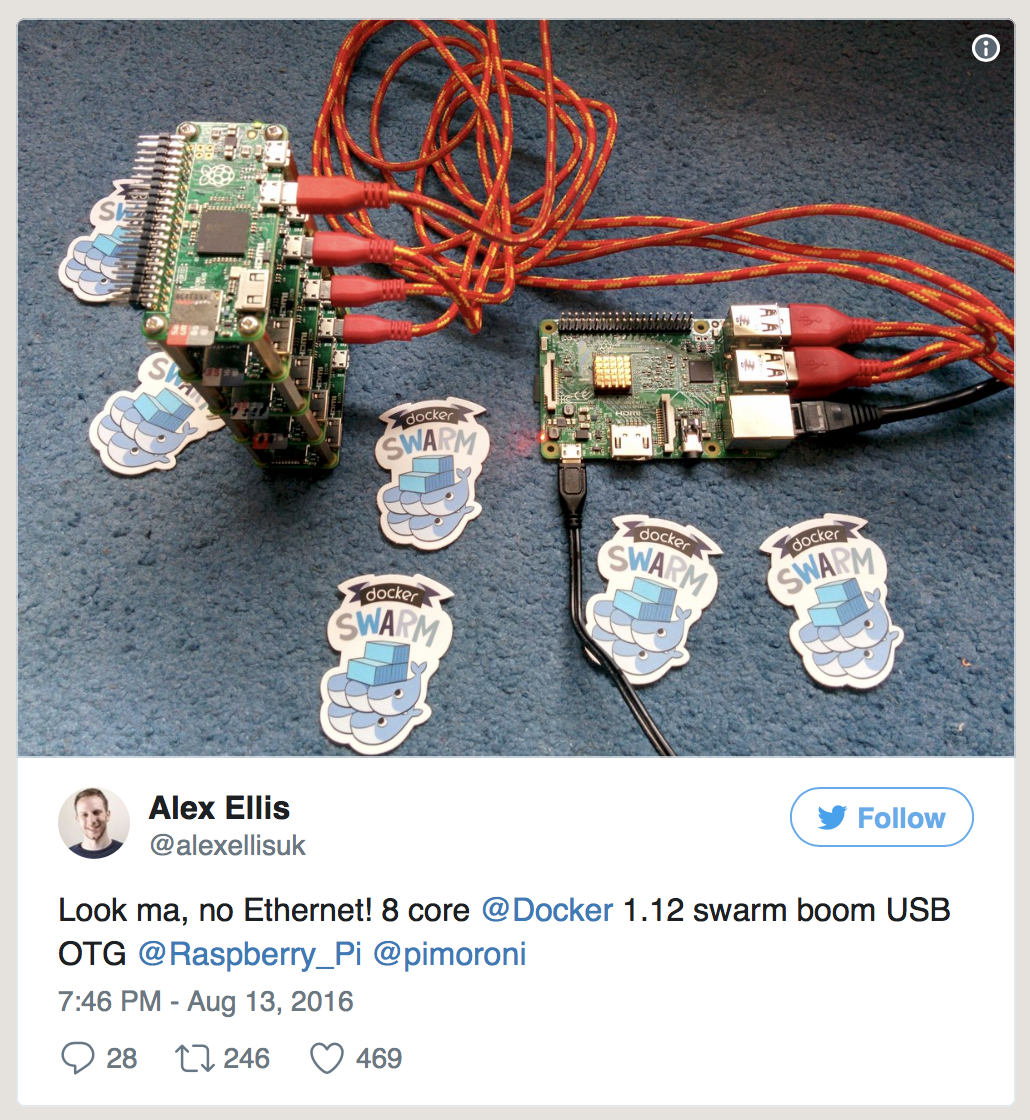


Figure 4 Raspberry PI Docker cluster

Our system is hosted in a single free VM on Azure.

## TravisCI

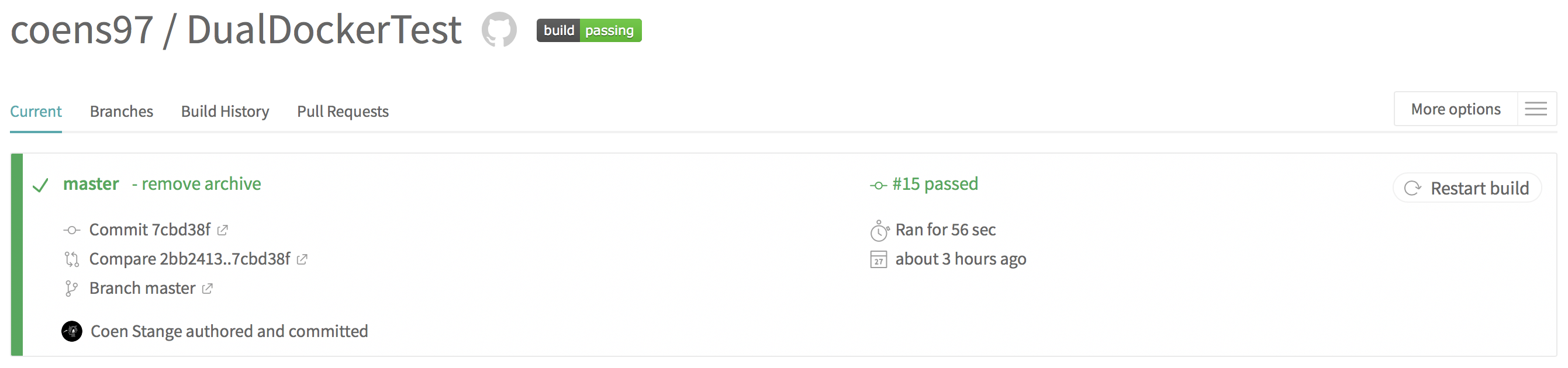


Figure 5 TravisCI test result

TravisCI is used for automatically running tests. The tests are run after GIT repository has been updated. The tests have to be configured in the Travis configuration file as shown below.

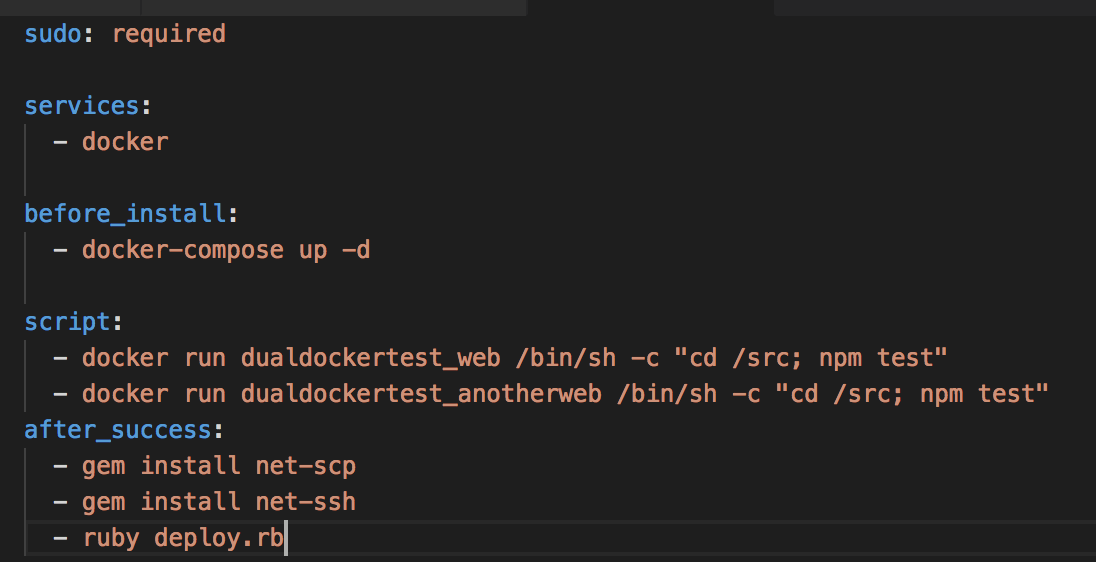


Figure 6 Travis configuration file

Using Docker makes running the tasks a bit trickier. The test run from within the Docker container.

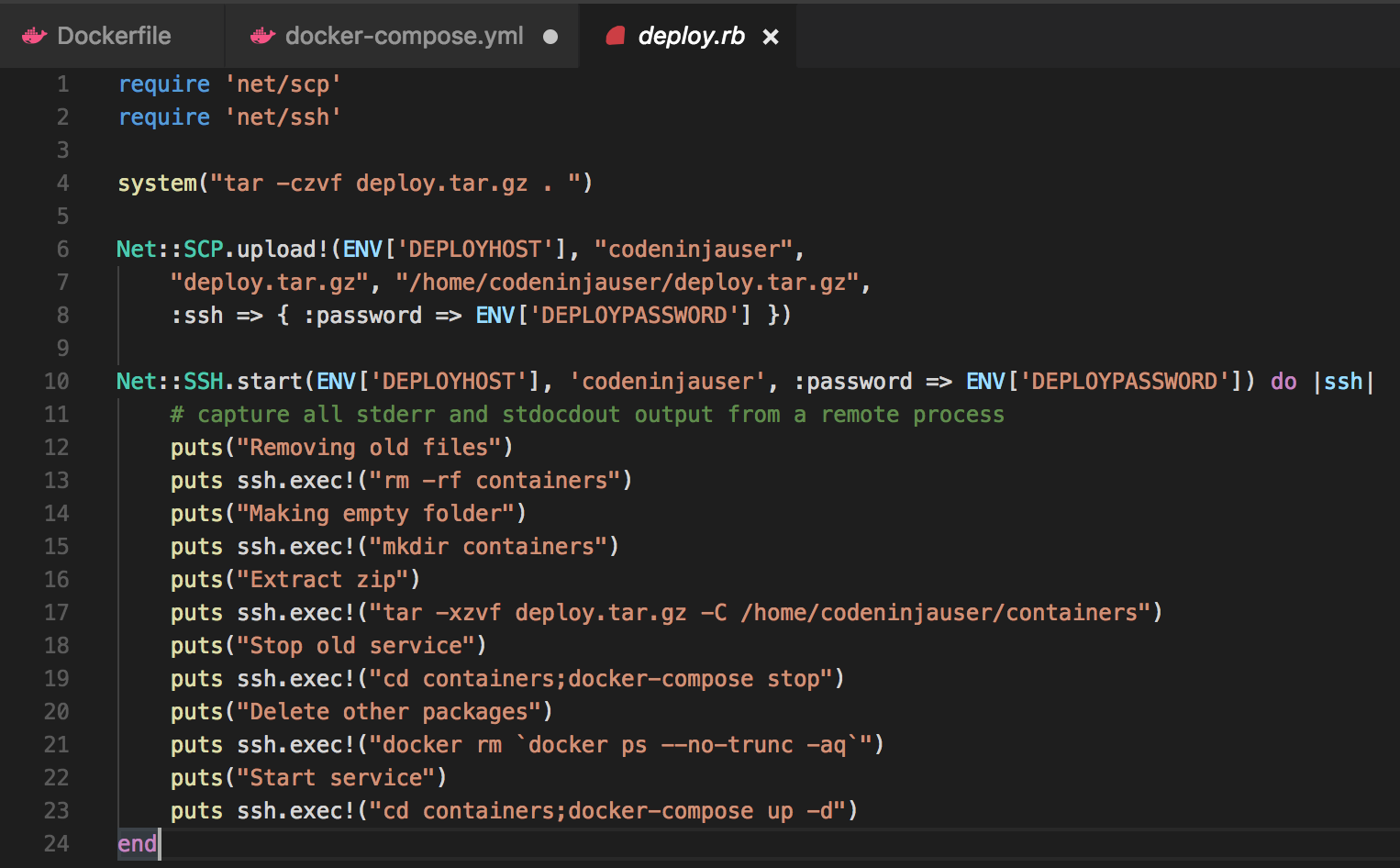


Figure 7 Deployment script

TravisCI has built-in deployment functionality. This allows when all tests pass that the new version of the system is published. The system is hosted inside a single VM on Azure. TravisCI doesn’t support this kind of configuration. Therefor we created a custom deployment script.

# Why Docker, Docker compose and TravisCI?

* Docker allows to run an application in a container, which can run on any machine
* Docker-compose allows to run the full system at once, which gives the following benefits:
  + Containers can be linked to each other on how they are dependent. E.g. all the containers which requires a database are linked. This automatically creates a private network (not open to the public).
  + The ports that are opened to the public are configurable.
  + There is a wide offer of public containers, including many popular database systems. These can without much hassle be added to the system.
  + Because the system can be run at once allows the tests to depend on other parts of the system. Hence integration tests are possible, and a mock (fake implementation) of the other API’s is not needed.
  + The whole system can be build, test and deployed at once (on TravisCI). Because of this the deployed system doesn’t have incompatible services. All the containers are tested together and deployed as a whole.
  + Docker-compose allow to deploy easily to a cluster of computers. Or to a supported cloud provider.
  + Docker-compose also allow to run the system on a single computer.
* TravisCI allows to build, test and deploy the whole distributed system at once.
  + TravisCI always build the distributed system from an empty machine. This makes sure that the system doesn’t depend on system configuration.
  + Because of TravisCI we don’t have to deploy by hand
  + TravisCI allows to deploy to different environments based on the branch. E.g. master branch run the stable environment. The develop branch runs in the development environment. Hence the customer can have a stable version which he can show to his friends.

# Nice commands to know

## Docker

Be aware these commands are only used to run a single container! Our system runs on docker-compose which depends on multiple containers.

**Test if docker is working:**

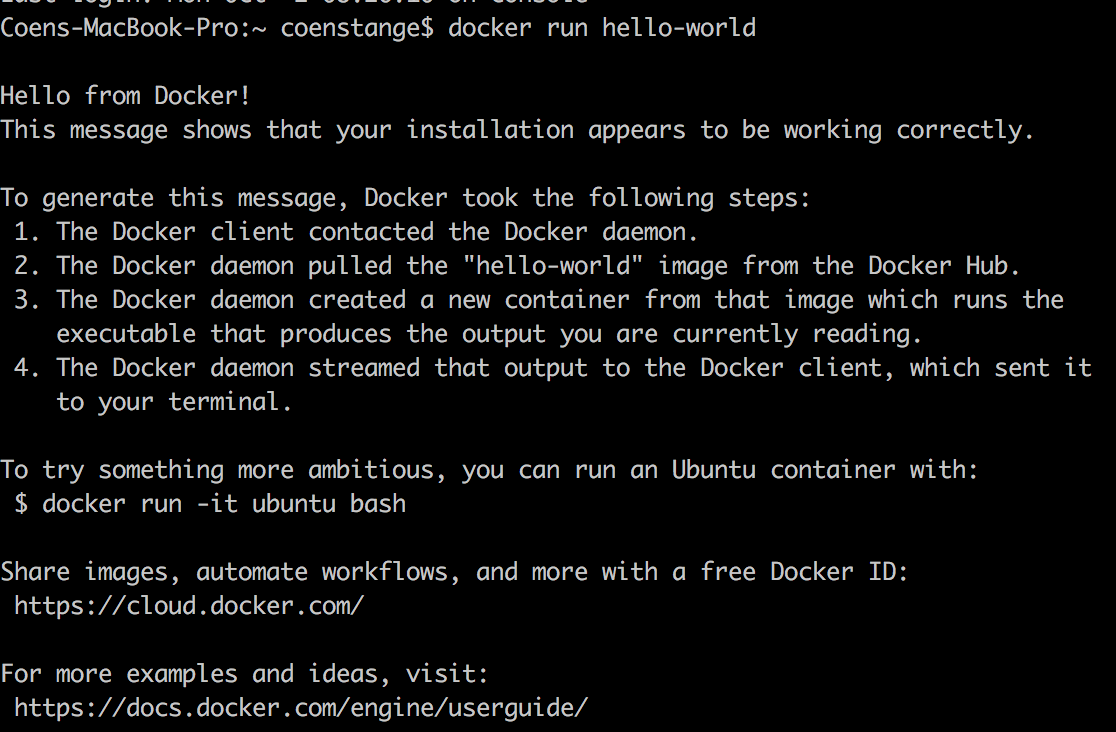


Figure docker run hello-world

docker build -t testjs .

Build a single docker container, the docker file is in the current directory. The name of the docker is testjs.

docker run -it -p 9000:9000 testjs

Run the docker image testjs. Expose the port 9000 to the public.

## Docker-compose

These commands will be used to run the distributed system.

docker-compose build

Build the distributed system, this will download all the required files. And copy the required files to the containers. Every time a file is changed this command needs to be built (unless the files are used within a volume).

docker-compose up

Run the distributed system, this command will also built the system if it is not built yet.

docker-compose up -d

Run the distributed system in the background.

docker-compose ps

Show the running processes.