



# Dockerizing a Node.js Web App

Version v1.5 (Latest)

[Examples \(examples/nodejs\\_web\\_app\)](#) [Articles \(/articles/basics/\)](/articles/basics/) [Reference \(/reference/commandline/cli/\)](/reference/commandline/cli/)[Contributor Guide \(/contributor-guide/\)](/contributor-guide/) [Project Who Wants For \(/project/who-wants-for/\)](/project/who-wants-for/) [root access \(/installation/binaries/#giving-non-root-access\)](/installation/binaries/#giving-non-root-access)

The goal of this example is to show you how you can build your own Docker images from a parent image using a `Dockerfile`. We will do that by making a simple Node.js hello world web application running on CentOS. You can get the full source code at <https://github.com/enokd/docker-node-hello/> (<https://github.com/enokd/docker-node-hello/>).

## Create Node.js app

First, create a directory `src` where all the files would live. Then create a `package.json` file that describes your app and its dependencies:

```
{
  "name": "docker-centos-hello",
  "private": true,
  "version": "0.0.1",
  "description": "Node.js Hello world app on CentOS using docker",
  "author": "Daniel Gasienica <daniel@gasienica.ch>",
  "dependencies": {
    "express": "3.2.4"
  }
}
```

Then, create an `index.js` file that defines a web app using the Express.js (<http://expressjs.com/>) framework:

```
var express = require('express');

// Constants
var PORT = 8080;

// App
var app = express();
app.get('/', function (req, res) {
  res.send('Hello world\n');
});

app.listen(PORT);
console.log('Running on http://localhost:' + PORT);
```

In the next steps, we'll look at how you can run this app inside a CentOS container using Docker. First, you'll need to build a Docker image of your app.

## Creating a Dockerfile

Create an empty file called `Dockerfile`:

```
touch Dockerfile
```

Open the `Dockerfile` in your favorite text editor

Define the parent image you want to use to build your own image on top of. Here, we'll use CentOS ([https://registry.hub.docker.com/\\_/centos/](https://registry.hub.docker.com/_/centos/)) (tag: `centos6`) available on the Docker Hub (<https://hub.docker.com/>):

```
FROM centos:centos6
```

Since we're building a Node.js app, you'll have to install Node.js as well as npm on your CentOS image. Node.js is required to run your app and npm to install your app's dependencies defined in `package.json`. To install the right package for CentOS, we'll use the instructions from the Node.js wiki (<https://github.com/joyent/node/wiki/Installing-Node.js-via-package-manager#rhelcentoscientific-linux-6>):

```
# Enable EPEL for Node.js
RUN rpm -Uvh http://download.fedoraproject.org/pub/epel/6/i386/epel-release-6-8.noarch.rpm
# Install Node.js and npm
RUN yum install -y npm
```

To bundle your app's source code inside the Docker image, use the `COPY` instruction:

```
# Bundle app source
COPY . /src
```

Install your app dependencies using the `npm` binary:

```
# Install app dependencies
RUN cd /src; npm install
```

Your app binds to port `8080` so you'll use the `EXPOSE` instruction to have it mapped by the `docker` daemon:

```
EXPOSE 8080
```

Last but not least, define the command to run your app using `CMD` which defines your runtime, i.e. `node`, and the path to our app, i.e. `src/index.js` (see the step where we added the source to the container):

```
CMD ["node", "/src/index.js"]
```

Your `Dockerfile` should now look like this:

```
FROM centos:centos6

# Enable EPEL for Node.js
RUN rpm -Uvh http://download.fedoraproject.org/pub/epel/6/i386/epel-release-6-8.noarch.rpm
# Install Node.js and npm
RUN yum install -y npm

# Bundle app source
COPY . /src
# Install app dependencies
RUN cd /src; npm install

EXPOSE 8080
CMD ["node", "/src/index.js"]
```

## Building your image

Go to the directory that has your `Dockerfile` and run the following command to build a Docker image. The `-t` flag lets you tag your image so it's easier to find later using the `docker images` command:

```
$ sudo docker build -t <your username>/centos-node-hello .
```

Your image will now be listed by Docker:

```
$ sudo docker images

# Example
REPOSITORY          TAG         ID              CREATED
centos              centos6    539c0211cd76    8 weeks ago
<your username>/centos-node-hello  latest     d64d3505b0d2    2 hours ago
```

## Run the image

Running your image with `-d` runs the container in detached mode, leaving the container running in the background. The `-p` flag redirects a public port to a private port in the container. Run the image you previously built:

```
$ sudo docker run -p 49160:8080 -d <your username>/centos-node-hello
```

Print the output of your app:

```
# Get container ID
$ sudo docker ps

# Print app output
$ sudo docker logs <container id>

# Example
Running on http://localhost:8080
```

## Test

To test your app, get the port of your app that Docker mapped:

```
$ sudo docker ps

# Example
ID          IMAGE          COMMAND          ...  PORTS
ecce33b30ebf  <your username>/centos-node-hello:latest  node /src/index.js  49160->8080
```

In the example above, Docker mapped the 8080 port of the container to 49160.

Now you can call your app using curl (install if needed via: sudo apt-get install curl):

```
$ curl -i localhost:49160

HTTP/1.1 200 OK
X-Powered-By: Express
Content-Type: text/html; charset=utf-8
Content-Length: 12
Date: Sun, 02 Jun 2013 03:53:22 GMT
Connection: keep-alive

Hello world
```

If you use Boot2docker on OS X, the port is actually mapped to the Docker host VM, and you should use the following command:

```
$ curl $(boot2docker ip):49160
```

We hope this tutorial helped you get up and running with Node.js and CentOS on Docker. You can get the full source code at <https://github.com/enokd/docker-node-hello/> (<https://github.com/enokd/docker-node-hello/>).

Community

Events  
(<https://www.docker.com/community/events/>)

Friends' Posts  
(<http://posts.docker.com/>)

Meetups  
(<https://www.docker.com/community/meetups/>)

Governance  
(<https://www.docker.com/community/governance/>)

Forums  
(<http://forums.docker.com/>)

IRC  
(<http://botbot.me/freenode/docker/>)

GitHub

Enterprise

Support  
(<https://www.docker.com/enterprise/support/>)

Education  
(<https://www.docker.com/enterprise/education/>)

Services  
(<https://www.docker.com/enterprise/services/>)

Partner Solutions  
(<https://www.docker.com/enterprise/partners/>)

Find a Partner  
(<https://www.docker.com/enterprise/partners/find/>)

Partner Program  
(<https://www.docker.com/enterprise/partners/program/>)

Learn More  
(<https://www.docker.com/enterprise/partners/learn-more/>)

Resources

Documentation  
(<https://docs.docker.com/>)

Help  
(<https://www.docker.com/help/>)

Use Cases  
(<https://www.docker.com/use-cases/>)

Online Tutorial  
(<https://www.docker.com/tutorial/>)

How To Buy  
(<https://www.docker.com/how-to-buy/>)

Status  
(<https://status.docker.com/>)

Security

Company

About Us  
(<https://www.docker.com/company/about-us/>)

Team  
(<https://www.docker.com/company/team/>)

News  
(<https://www.docker.com/company/news/>)

Press  
(<https://www.docker.com/company/press/>)

Careers  
(<https://www.docker.com/company/careers/>)

Contact  
(<https://www.docker.com/company/contact/>)

Connect

Subscribe to our newsletter  

Enter your email →

Blog  
(<http://blog.docker.com/>)

Slideshare  
(<http://www.slideshare.net/Docker>)

Twitter  
(<http://twitter.com/docker>)

LinkedIn  
(<http://www.linkedin.com/company/docker>)

GitHub  
(<https://github.com/docker>)

Google+  
(<https://plus.google.com/u/0/+docker>)

Facebook  
(<https://www.facebook.com/docker>)

Reddit  
(<http://www.reddit.com/r/docker>)

Angellist  
(<http://www.youangellist.com/docker>)

(<https://github.com/docker/docker>) (<https://www.docker.com/patterns/learn-docker>) (<https://www.docker.com/resources/security/>)

inc-1)

Stackoverflow

([http://stackoverflow.com/search?](http://stackoverflow.com/search?q=docker)

[q=docker](http://stackoverflow.com/search?q=docker))

Swag

(<http://www.cafepress.com/docker>)