Title: Nginx can't find upstream node app when running via Docker-compose

Post Body:

I have a super simple Node app and an Nginx config which acts as a reverse proxy to the Node app. Everything works fine if I run Nginx (via homebrew) and the Node app locally. If I visit the server defined by the Nginx config at port 8080 I get the output from the node app, which is running on port 3000.

I've been trying to convert this simple setup to use Docker and have written the following Docker-compose file:

```
version: '3.0' services:   web:     build: .     ports:       - 3000:3000   nginx:     build:       context: .       dockerfil
```

On running `docker-compose up` the images are built and there are no error messages in the console. On visiting `localhost:3000` I get the response from the Node app but on visiting `localhost:8080` I get an an Nginx 502 error page and the following error in the terminal:

> connect() failed (111: Connection refused) while connecting to upstream, client: 172.18.0.1, server: localhost, request: 'GET / HTTP/1.1', upstream: 'http://127.0.0.1:3000/', host: 'localhost:8080'

My Dockerfile for the node app looks like so:

```
FROM node:carbon  WORKDIR /app  ADD . /app  RUN npm install  CMD ['node', '.']  EXPOSE 3000
```

and the Dockerfile.nginx looks like so:

```
FROM nginx COPY nginx.conf /etc/nginx/nginx.conf
```

and the nginx.conf looks like so:

```
events {   worker_connections  1024; }  http {    upstream node_app {     server 127.0.0.1:3000;   }    server_tokens off;
```

On starting Docker up I can see that Nginx is running on port 8080 (because I see the 502 Nginx page) and I can see that the node app is running (because I can visit it at localhost:3000). I can't work out why I get the 502 from nginx.

I've tried using various different things like using `links` to link the containers and `depends_on` but nothing seems to make any difference. I'm also using `docker-compose up --build` to make sure I'm not caching previous builds each time I make a change.

EDIT: Something that seems to make it work is to add a container_name property in to the docker-compose:

```
  web:     container_name: nodeapp     build:       context: .       dockerfile: Dockerfile.node     ports:       - 3000:3000
```

and then using that container name in the upstream node_app config in nginx.conf:

```
  upstream node_app {     server nodeapp:3000;   }
```

which makes no sense to me?!

Accepted Answer:

The problem is that in your Nginx configuration you are referencing the IP of the Web service as 127.0.0.1 which is the loopback address of the Host machine running the docker container. This may work depending on your setup (OS, firewall) or may not.

The correct way would be to make the `nginx` service depends on the `web` service in your docker-compose.yml file and update the Nginx config to reference the Web service by name (`web`) instead of by IP address. Here you can find more info related to docker compose depends on capability.

The updated docker-compose.yml file would be:

```
version: '3.0' services:   web:     build: .   nginx:     build:       context: .       dockerfile: Dockerfile.nginx     ports
```

Notice that I have stop exposing the port of the `web` service. May be you need to keep it to monitor the Web service but is not required for the `nginx` service.

With this update to the docker-compose.yml file the Nginx config will be as follows:

```
events {   worker_connections  1024; }  http {    upstream node_app {     server web:3000;   }    server_tokens off;    # Defi
```

Highest Rated Answer: None