September 11, 2014 #docker | #x11 | #firefox | #netbeans | #java

Running GUI apps with Docker

I’ve been doing all of my real (paid) work on VMs / containers for a while now but when it comes to writing Java code for some projects for university I still need to move away from using vim and install some full blown IDE in order to be productive. This has been bothering me for quite some time but this week I was finally able put the pieces together to run NetBeans in a Docker container so that I can avoid installing a lot of Java stuff on my machine that I don’t use on a daily basis.

There are a few different options to run GUI applications inside a Docker container like using SSH with X11 forwarding, or VNC but the simplest one that I figured out was to share my X11 socket with the container and use it directly.

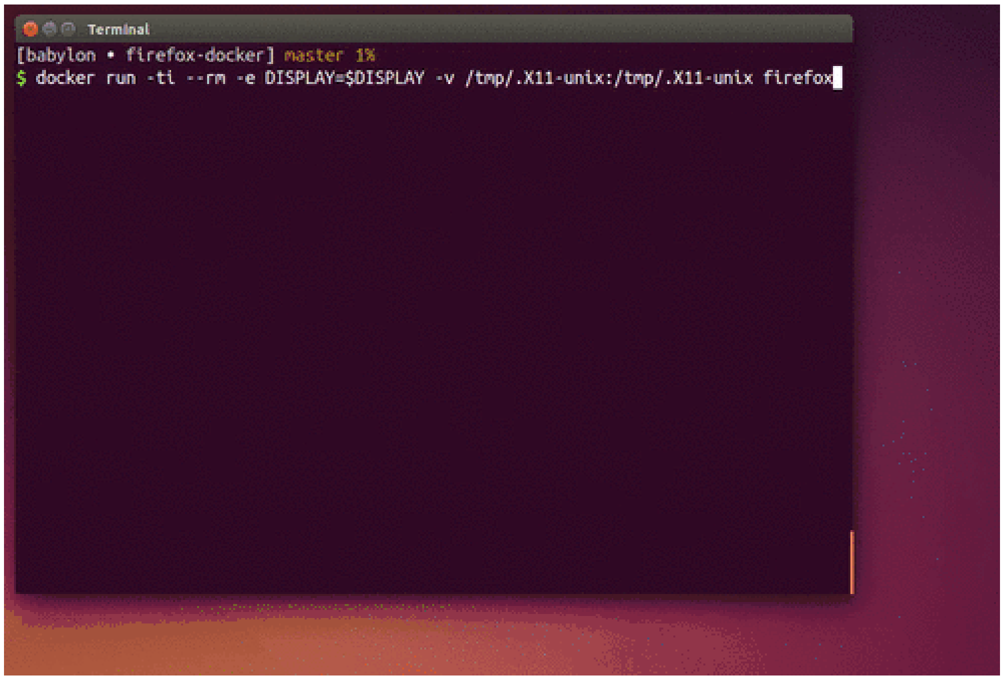
The idea is pretty simple and you can easily it give a try by running a Firefox container using the following `Dockerfile` as a starting point:

```
FROM ubuntu:14.04

RUN apt-get update && apt-get install -y firefox

# Replace 1000 with your user / group id
RUN export uid=1000 gid=1000 && \
    mkdir -p /home/developer && \
    echo "developer:x:${uid}:${gid}:Developer,,,:/home/developer:/bin/bash" >> /etc/passwd && \
    echo "developer:x:${uid}:" >> /etc/group && \
    echo "developer ALL=(ALL) NOPASSWD: ALL" > /etc/sudoers.d/developer && \
    chmod 0440 /etc/sudoers.d/developer && \
    chown ${uid}:${gid} -R /home/developer

USER developer
ENV HOME /home/developer
CMD /usr/bin/firefox
```

`docker build -t firefox .` it and run the container with:

```
docker run -ti --rm \
       -e DISPLAY=$DISPLAY \
       -v /tmp/.X11-unix:/tmp/.X11-unix \
       firefox
```

If all goes well you should see Firefox running from within a Docker container.



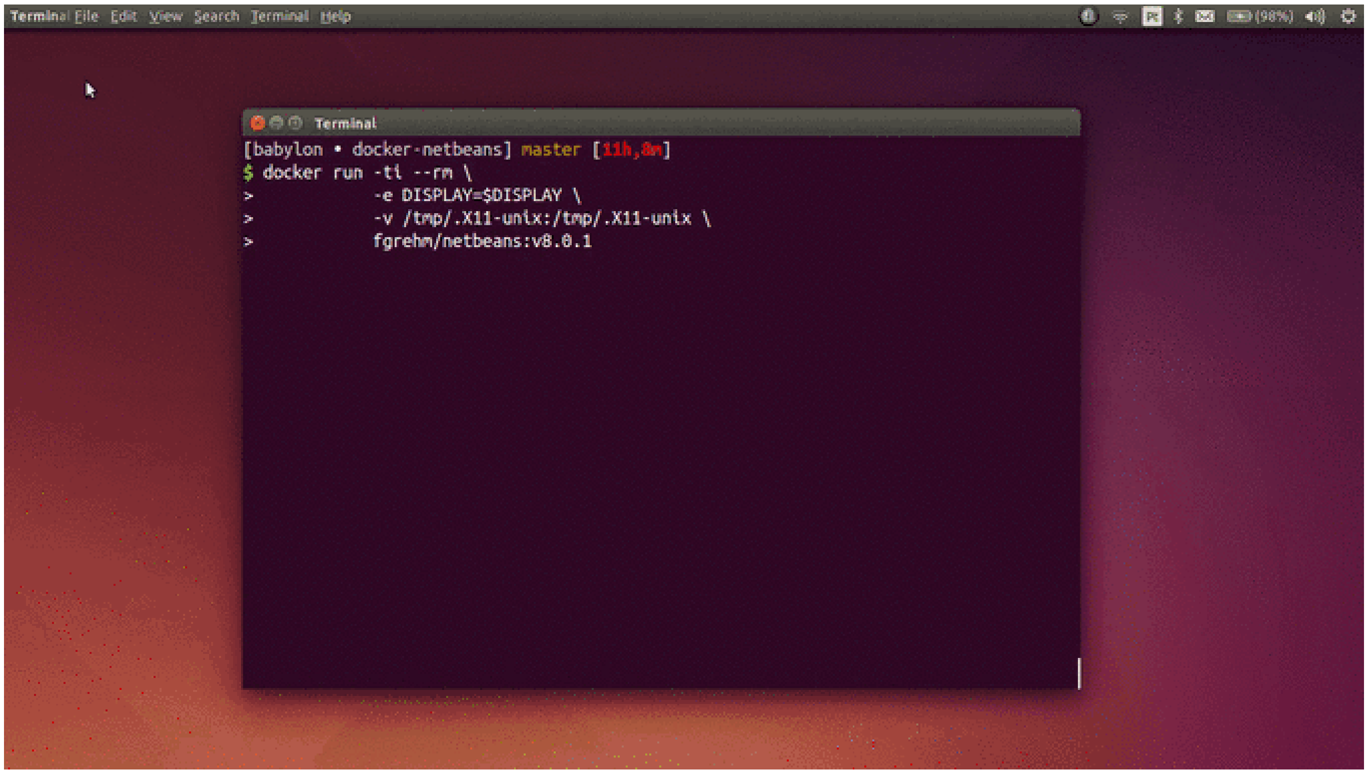
```
[babylon • firefox-docker] master 1%
$ docker run -ti --rm -e DISPLAY=$DISPLAY -v /tmp/.X11-unix:/tmp/.X11-unix firefox
```

Getting a NetBeans container up and running

Preparing a NetBeans base image was not that straightforward since we need to install some additional dependencies (namely the `libxext-dev`, `libxrender-dev` and `libxtst-dev` packages) in order to get it to connect to the X11 socket properly. I also had trouble using OpenJDK and had to switch to Oracle's Java for it to work.

After lots of trial and error, I was finally able to make it work and the result is a base image available at the Docker Hub with sources on GitHub.

Here's a quick demo of it in action:



```
[babylon • docker-netbeans] master [11h,8m]
$ docker run -ti --rm \
>           -e DISPLAY=$DISPLAY \
>           -v /tmp/.X11-unix:/tmp/.X11-unix \
>           fgrehm/netbeans:v8.0.1
```

Future work

Over the next few months I'll be working on a Play! app and will hopefully write a blog post on the workflow I used. Stay tunned for more :)

PS: This approach of sharing the X11 socket also be applied to vagrant-lxc containers and I'll document that on the project's Wiki when I have a chance.

104 Comments **Fábio Rehm blog** 1 Login

Recommend 27 Share Sort by Best

Join the discussion…

LOG IN WITH

OR SIGN UP WITH DISQUS ?

Name

Sameer Naik • 3 years ago

Based on your work I put together a docker image that packs google-chrome and the tor-browser http://git.io/-k_k1A

It also features audio redirection via pulseaudio socket connection and wrapper scripts to deal with cruft of launching the image.

23 • Reply • Share ›

Gui Ambros → Sameer Naik • a year ago

Love this. Thanks both Fabio and Sameer for your work.

2 • Reply • Share ›

fgrehm Mod → Sameer Naik • 3 years ago

That's awesome! Thanks for sharing :)

• Reply • Share ›

airtonix → Sameer Naik • 3 years ago

pulseaudio rocks!

• Reply • Share ›

easeway • 3 years ago

It doesn't work with latest docker running on ubuntu 14.04. There's a few things missing: --net=host and .Xauthority file.

For "--net=host", by default, docker creates a separate network namespace inside the container which can't access host network stack including unix sockets. Using "--net=host" will not contain network stack, the program inside the container shares the same network stack as the host.

".Xauthority" file is required if your current X11 session requires a valid user. Otherwise X11 client fails to connect to X server.

Here's my command line: