# DOCKER + SHINY APP + SHINYPROXY + APACHE

## Shiny Proxy, why?

Because we need Enterprise features but stay open source (we need multiple user using David’s web page at the same time).

## Prerequisites

In order to have Shiny Proxy working properly, we need to install in our server:

### Java 8 (or higher)

java –version

### Docker

#### Installation

Regarding the installation, these tutorials can be useful:

* Docker on [Linux](https://docs.docker.com/install/)
* Docker on [Mac OS](https://docs.docker.com/docker-for-mac/install/)
* Docker on [Windows](https://docs.docker.com/docker-for-windows/install/)

#### Configuration

ShinyProxy needs to connect to the docker daemon to spin up the containers for the Shiny apps. By default, ShinyProxy will do so on port 2375 of the docker host. In order to allow for connections on port 2375, the docker startup options need to be edited.

On CentOS7:

1. Open the docker configuration file:

sudo systemctl edit docker

1. Add following lines:

[Service]

ExecStart=

ExecStart=/usr/bin/dockerd -H unix:// -D -H tcp://127.0.0.1:2375

1. Restart docker:

sudo systemctl daemon-reload

sudo systemctl restart docker

### Shiny Proxy

[Download](https://www.shinyproxy.io/downloads/) and install Shiny Proxy on CentOS7.

sudo yum install downloaded\_file\_name.rpm

Once the installation has finished, we need to locate where the shinyproxy.jar file is. This file is the one, which will allow us to run Shiny Proxy in this way:

java -jar shinyproxy-2.0.5.jar

## Deploying a Shiny App

If we want to deploy our Shiny app (David’s web page in this case), we need to build our own Docker image for the app.

### Write a Dockerfile

Docker images are built starting from a Dockerfile. The Dockerfile starts from a preexisting image and builds up the image command by command. A Docker image will typically contain:

* an R installation with
* all R packages the Shiny app depends on ('dependencies') and
* a folder which contains the files for our Shiny app.

#### Dockerfile

#Image base

FROM openanalytics/r-base

#System libraries

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

sudo \

pandoc \

pandoc-citeproc \

libcurl4-gnutls-dev \

libcairo2-dev \

libxt-dev \

libssl-dev \

libssh2-1-dev \

libssl1.0.0

RUN apt-get update -qq && apt-get -y --no-install-recommends install \

libxml2-dev \

libsqlite3-dev \

libmariadbd-dev \

libmariadb-client-lgpl-dev \

libpq-dev \

libssl-dev \

libcurl4-openssl-dev \

libssh2-1-dev \

unixodbc-dev \

&& R -e "source('https://bioconductor.org/biocLite.R')" \

&& install2.r --error \

--deps TRUE \

tidyverse \

dplyr \

devtools \

formatR \

remotes \

selectr \

caTools

RUN apt-get install dialog apt-utils -y

RUN apt-get install libssl-dev -y

RUN apt-get install libv8-3.14-dev -y

RUN apt-get install libjpeg-dev –y

#R packages

RUN R -e "install.packages(c('shiny','rmarkdown'), repos='http://cran.rstudio.com/', dependencies=T)"

RUN R -e "install.packages(c('shinyjs'), repos='http://cran.rstudio.com/', dependencies=T)"

RUN R -e "install.packages(c('openssl'), repos='http://cran.rstudio.com/', dependencies=T)"

RUN R -e "install.packages(c('httr'), repos='http://cran.rstudio.com/', dependencies=T)"

RUN R -e "install.packages(c('ggpubr'), repos='http://cran.rstudio.com/', dependencies=T)"

RUN R -e "source('https://bioconductor.org/biocLite.R'); biocLite('biomaRt')"

RUN R -e "source('https://bioconductor.org/biocLite.R'); biocLite('GenomicFeatures')"

RUN R -e "source('https://bioconductor.org/biocLite.R'); biocLite('Gviz')"

RUN R -e "install.packages(c('DT'), repos='http://cran.rstudio.com/', dependencies=T)"

RUN R -e "source('https://bioconductor.org/biocLite.R'); biocLite('regioneR')"

RUN R -e "source('https://bioconductor.org/biocLite.R'); biocLite('GenomicScores')"

#ShinyApp files

RUN mkdir /root/GenomeBrowser

COPY GenomeBrowser /root/GenomeBrowser

COPY Rprofile.site /usr/lib/R/etc/

#Port

EXPOSE 3838

#Execute

CMD ["R", "-e", "shiny::runApp('/root/GenomeBrowser')"]

#### Rprofile.site

local({

options(shiny.port = 3838, shiny.host = "0.0.0.0")

})

### Build the Docker image

Now the Dockerfile has been created, we need to build the Docker image using the following command (we must be located in the same folder where the Dockerfile is located):

sudo docker build -t name\_of\_the\_docker\_image .

### Configure ShinyProxy

The configuration of ShinyProxy happens in a single file application.yml. If we want to add an application (David’s app), it has to be specified in the **specs** block as follows:

server:

servlet:

context-path: /browser/ #will be deployed in www.xxxx.com/browser/

proxy:

hide-navbar: true

heartbeat-rate: 100000

heartbeat-timeout: 300000

container-wait-time: 300000

authentication: none

port: 2525

docker:

url: http://localhost:2375

port-range-start: 20000

port-range-max: 20099

specs:

- id: GenomeBrowser

display-name: Genome Browser

container-cmd: ["R", "-e shiny::runApp('/root/GenomeBrowser')"]

container-image: genomebrowser-shinyproxy

logging:

file:

shinyproxy.log

More details [here](https://www.shinyproxy.io/configuration/).

## Apache configuration

This Shiny App will be deployed at http://servername/browser/

<Proxy \*>

Allow from localhost

</Proxy>

RedirectMatch permanent ^/browser$ /browser/

RewriteEngine on

RewriteCond %{HTTP:Upgrade} =websocket

RewriteRule /browser/(.\*) ws://0.0.0.0:2525/browser/$1 [P,L]

RewriteCond %{HTTP:Upgrade} !=websocket

RewriteRule /browser/(.\*) http://0.0.0.0:2525/browser/$1 [P,L]

ProxyPass /browser/ http://0.0.0.0:2525/browser/ connectiontimeout=3000 timeout=3000

ProxyPassReverse /browser/ http://0.0.0.0:2525/browser/

Header edit Location ^/ /browser/

ProxyRequests Off