DevOps Journey : DevHackThorsDay

various authors

DEVOPS Journey : DevHackThorsDay

DevOps is a set of software development practices that combines software development (Dev) and information technology operations (Ops) to shorten the systems development life cycle while delivering features, fixes, and updates frequently in close alignment with business objectives. Different disciplines collaborate, making quality everyone’s job.

The goals of DevOps span the entire delivery pipeline including :

- improved deployment frequency
- faster time to market
- lower failure rate of new releases
- shortened lead time between fixes
- faster mean time to recovery (in the event of a new release crashing or otherwise disabling the current system)

chapter 0 - getting started from zero

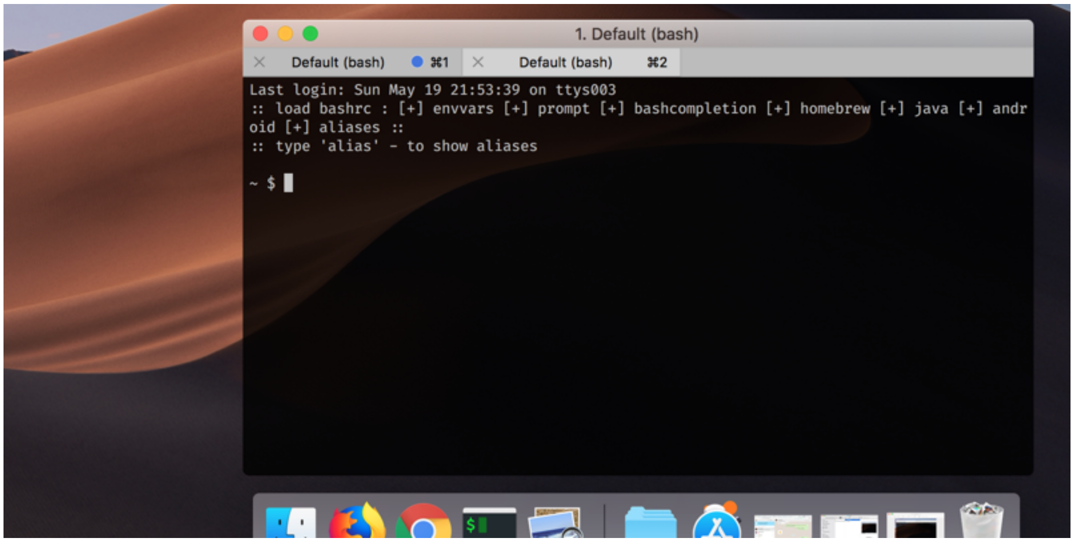
You may have come from various backgrounds, front end developer, backend developer, sys admin, freelancer design or even an accountant wanting to pick up some skills on development. This mob book will be covering topics on a wide spectrum of software development methods and technologies under the sun; so we will have you covered.

A few things to get yourself embarked on this journey.

- sign up for meetups on technology often; be it Ruby language, cloud initiative or AWS summits; sign up!
- start a github account or heroku account
- familiarise yourself with a basic coding language; ruby, python are good picks to get you started quickly
- sign up to stackoverflow where there is a large range of tech Q and As
- make your laptop your temple; clear it up and start building it as your personalised tool for development

for your day to day development, either use a linux flavor or use macosx (which is bsd a variant of unix); windows is possible with powershell or cygwin, which allows bash shell on the machine. recommend to stick to linux or unix flavoured machines; it makes more sense.

if you are on mac, use iterm2 (https://iterm2.com/), its much better than the default terminal.



```
Last login: Sun May 19 21:53:39 on ttys003
:: load bashrc : [+] envvars [+] prompt [+] bashcompletion [+] homebrew [+] java [+] andr
oid [+] aliases ::
:: type 'alias' - to show aliases

~ $
```

chapter 1 - docker

Docker is a collection of inter-operating services that employs operating-system-level virtualization to facilitate development and delivery of software inside standardised software packages called containers. The software that hosts and manages the containers is called docker engine. Its first started in 2013 and is developed by docker, inc.

The docker containers are isolated from each other and bundle their own software, libraries and configuration files; they can communicate with each other through well-defined channels. All containers are run by a single operating-system kernel and are thus more lightweight than virtual machines such as vagrant on virtualbox. Containers are created from images that specify their precise contents and are often created by combining and modifying standard images downloaded from public repositories.

priming

here are some docker commands to get you started. give them a spin in your terminal and check docker website to see more about other commands, such as volume, compose etc.

```
docker ps -a
# to show all running docker processes

docker images
# to see all local images

docker images purge
# to remove older version of images. such as called dangling images

docker build -t [name] .
# to build a container based on current dockerfile and tag it

docker run -it myruby
# to run load image (myruby) and run container application it interactively

docker run -d -p 3000:3000 --name myproc myruby
# to run image (myruby) as a daemon

docker run -d -p 8080:80 --name myproc myruby
# map from 8080 (host) to 80 (container)

docker exec -it [containerid] sh -c "echo a && echo b"
```

```
# to run commands in a running container

docker exec -it [containerid] bash
# to enter the interactive shell of the running container

docker rmi [image id]
# to remove docker images from system. example with image id 4c0d231f6a74

docker rm [container id]
# to remove running/stopped containers

docker system prune [-a]
# to remove all danging containers (which are stopped) and networks. optional -
a for all

docker container prune
# to remove all danging containers (which are stopped)

docker volume prune
# to clean up unused volumes
```

challenge one

fulfill the following requirements :

- on your local machine install docker
- create a dockerfile based off alpine linux
- install nginx as a web server in the container
- spin up and share the container on public docker hub (https://hub.docker.com/)
- have the final image under 30 mb

suggested solutions :

- JamesMannion (https://github.com/mannion007/devopsjourney/tree/master/src/assignment1)

challenge two

fulfill the following requirements :

- include php-fpm based off your public nginx container
- show basic php info
- the host port to access the running container should be on port 8081