DOCKER Q & A

SHARE
(http://www.addthis.com/bookmark.php?v=250&username=khhong7)

bogotobogo.com site search:

Search

Docker Q & A

1. **Difference between Docker Image and container.**

   If a Docker application works on our local computer, it'll work anywhere that supports Docker. It greatly simplifies development process and can be a powerful tool for continuous delivery.

   To understand Docker, we need to know two key facets of how Docker works. Docker image vs container!

   Docker image is a kind of snapshot. We can think of it as a picture of a Docker virtual machine and the container is the virtual machine.

Ph.D. / Golden Gate Ave, San Francisco / Seoul National Univ / Carnegie Mellon / UC Berkeley / DevOps / Deep Learning / Visualization

Docker & K8s

Docker install on Amazon Linux AMI
(/DevOps/Docker/Docker_Install_O

Docker install on EC2 Ubuntu 14.04
(/DevOps/Docker/Docker_Install_O

Docker container vs Virtual Machine
(/DevOps/Docker/Docker_Containe

Docker install on Ubuntu 14.04
(/DevOps/Docker/Docker_Install_O

Docker Hello World Application
(/DevOps/Docker/Docker_Hello_W

Nginx image - share/copy files, Dockerfile
(/DevOps/Docker/Docker_Nginx_W

Working with Docker images : brief introduction
(/DevOps/Docker/Docker_Working

Docker image and container via docker commands (search, pull, run, ps, restart, attach, and rm)
(/DevOps/Docker/Docker_Comman

More on docker run command (docker run -it, docker run --rm, etc.)
(/DevOps/Docker/Docker_Run_Com

Docker Networks - Bridge Driver Network
(/DevOps/Docker/Docker-Bridge-Driver-Networks.php)

A container is an instance of a docker image. In other words, a running instance of an image is a container. We can see all our images with `docker images`:

```
$ docker images
REPOSITORY          TAG                 IMAGE ID            CREATED             SIZE
mysql               latest              8457e9155715        9 days ago          546MB
busybox             latest              491198851f0c        2 weeks ago         1.23MB
ubuntu              18.04               c090eaba6b94        6 weeks ago         63.3MB
```

To see our running containers with `docker ps`:

```
$ docker ps
CONTAINER ID        IMAGE               COMMAND             CREATED             STATUS
2666137d9726        busybox             "sh"                2 hours ago         Up 2 hours
55ae35026d6c        ubuntu:18.04        "/bin/bash"         6 days ago          Up 6 days
```

To see all containers including the ones stopped, we can use `docker ps -a`:

```
$ docker ps -a
CONTAINER ID        IMAGE               COMMAND                  CREATED              STATUS
5f6afe49bbd0        mysql               "docker-entrypoint.s…"   About a minute ago   Exited
2666137d9726        busybox             "sh"                     2 hours ago          Up 2 h
55ae35026d6c        ubuntu:18.04        "/bin/bash"              6 days ago           Up 6 d
```

2. **What is a Dockerfile?**

In order to build the application, we need to use a Dockerfile.

In cooking, we need a recipe to make cookies. In Docker, we need a Dockerfile to build an image.

A Dockerfile is simply a text-based script of instructions that is used to create a container image. Basically, it contains all the possible commands that a user may call on the command line to create an image.

To see how it works, we'll use https://github.com/docker/getting-started (https://github.com/docker/getting-started).

Docker Persistent Storage (/DevOps/Docker/Docker_Containe

File sharing between host and container (docker run -d -p -v) (/DevOps/Docker/Docker_File_Sha

Linking containers and volume for datastore (/DevOps/Docker/Docker_Containe

Dockerfile - Build Docker images automatically I - FROM, MAINTAINER, and build context (/DevOps/Docker/Docker_Dockerfi

Dockerfile - Build Docker images automatically II - revisiting FROM, MAINTAINER, build context, and caching (/DevOps/Docker/Docker_Dockerfi

Dockerfile - Build Docker images automatically III - RUN (/DevOps/Docker/Docker_Dockerfi

Dockerfile - Build Docker images automatically IV - CMD (/DevOps/Docker/Docker_Dockerfi

Dockerfile - Build Docker images automatically V - WORKDIR, ENV, ADD, and ENTRYPOINT (/DevOps/Docker/Docker_Dockerfi

Docker - Apache Tomcat (/DevOps/Docker/Docker_Apache_

Docker - NodeJS (/DevOps/Docker/Docker-NodeJS.php)

Docker - NodeJS with hostname (/DevOps/Docker/Docker-NodeJS-with-hostname.php)

Docker Compose - NodeJS with MongoDB (/DevOps/Docker/Docker-Compose-Node-MongoDB.php)

Docker - Prometheus and Grafana with Docker-compose (/DevOps/Docker/Docker_Prometh

Docker - StatsD/Graphite/Grafana (/DevOps/Docker/Docker_StatsD_G

Docker - Deploying a Java EE JBoss/WildFly Application on AWS Elastic Beanstalk Using Docker Containers

```
$ git clone https://github.com/docker/getting-started/tree/master/app

$ tree ../getting-started -L 2
../getting-started
├── Jenkinsfile
├── LICENSE
├── README.md
├── app
│   ├── package.json
│   ├── spec
│   ├── src
│   └── yarn.lock
├── build.sh
├── docker-compose.yml
├── docs
│   ├── css
│   ├── fonts
│   ├── images
│   ├── index.md
│   └── tutorial
├── mkdocs.yml
├── requirements.txt
└── yarn.lock
```

1. Create a file named **Dockerfile** in the same folder as the file package.json with the following contents:

   ```
   FROM node:12-alpine
   WORKDIR /app
   COPY . .
   RUN yarn install --production
   CMD ["node", "src/index.js"]
   ```

2. Go to the app directory with the **Dockerfile** and build the container image using the `docker build` command:

   ```
   ~/getting-started/app $ ls
   Dockerfile      package.json    spec            src             yarn.lock

   ~/getting-started/app $ docker build -t getting-started .

   $ docker images
   REPOSITORY          TAG                 IMAGE ID            CREATED             SIZE
   getting-started     latest              295a1b181e50        22 minutes ago      179MB
   ```

   Here, we instructed the builder that we wanted to start from the **node:12-alpine** image. The **WORKDIR** command is used to define the working directory of a Docker container at any given time. Any RUN, CMD, ADD, COPY, or ENTRYPOINT command will be executed in the specified working directory.

   After the image was downloaded, we copied in our application and used `yarn` to install our application's dependencies. The `CMD` specifies the default command to run when starting a container from this image.

   The **-t** flag tags our image to give it a human-readable name. Since we named the image getting-started, we can refer to that image when we run a container.

   The **.** at the end of the `docker build` command tells that Docker should look for the Dockerfile in the current directory.

(/DevOps/Docker/Docker_Containe

Docker : NodeJS with GCP Kubernetes Engine (/DevOps/Docker/Docker-NodeJS-GCP-Kubernetes-Engine.php)

Docker : Jenkins Multibranch Pipeline with Jenkinsfile and Github (/DevOps/Docker/Docker-Jenkins-Multibranch-Pipeline-with-Jenkinsfile-and-Github.php)

Docker : Jenkins Master and Slave (/DevOps/Docker/Docker-Jenkins-Master-Slave-Agent-ssh.php)

Docker - ELK : ElasticSearch, Logstash, and Kibana (/DevOps/Docker/Docker_ELK_Elas

Docker - ELK 7.6 : Elasticsearch on Centos 7 (/DevOps/Docker/Docker_ELK_7_6_
Docker - ELK 7.6 : Filebeat on Centos 7 (/DevOps/Docker/Docker_ELK_7_6_

Docker - ELK 7.6 : Logstash on Centos 7 (/DevOps/Docker/Docker_ELK_7_6_

Docker - ELK 7.6 : Kibana on Centos 7 Part 1 (/DevOps/Docker/Docker_ELK_7_6_

Docker - ELK 7.6 : Kibana on Centos 7 Part 2 (/DevOps/Docker/Docker_ELK_7_6_

Docker - ELK 7.6 : Elastic Stack with Docker Compose (/DevOps/Docker/Docker_ELK_7_6_

Docker - Deploy Elastic Cloud on Kubernetes (ECK) via Elasticsearch operator on minikube (/DevOps/Docker/Docker_Kuberne

Docker - Deploy Elastic Stack via Helm on minikube (/DevOps/Docker/Docker_Kuberne

Docker Compose - A gentle introduction with WordPress (/DevOps/Docker/Docker-Compose.php)

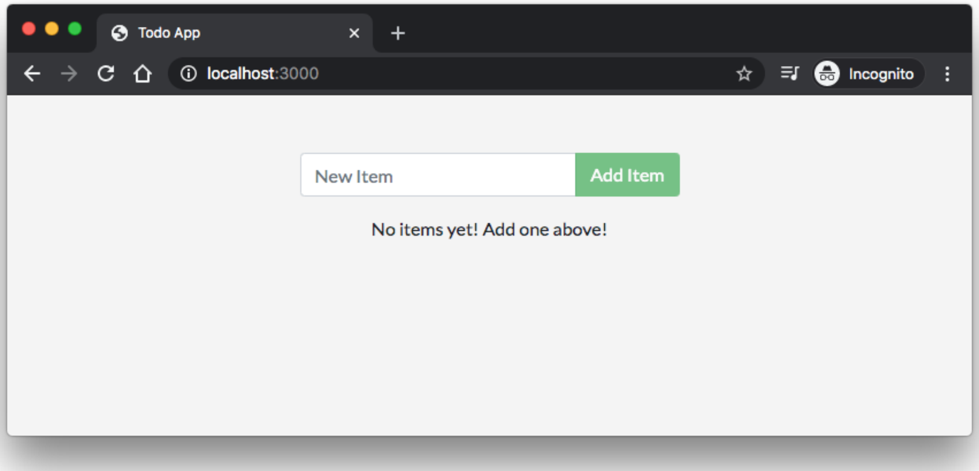
Docker Compose - MySQL (/DevOps/Docker/Docker-

Now that we have an image, it's time to run the application using `docker run` command:

```
$ docker run -d -p 3000:3000 getting-started
57198610f11c45a9eb4b7fd64fd93eabc2e0cbae17f980a8492618cb9822ba3f
```

Note that we're running the new container in "detached" mode (in the background) and creating a mapping between the host's port 3000 to the container's port 3000. Without the port mapping, we wouldn't be able to access the application.

3. Open a web browser to http://localhost:3000. We should see our app:



No items yet! Add one above!

4. One more thing about the `CMD` instruction: the difference between the `CMD` and `ENTRYPOINT` with related to the supplied to the docker run command. While the CMD will be completely over-written by the supplied command (or args), for the `ENTRYPOINT`, the supplied command will be appended to it (Dockerfile - Build Docker images automatically V - WORKDIR, ENV, ADD, and ENTRYPOINT (/DevOps/Docker/Docker_Dockerfile_to_build_images_automatically_5_WORKDIR_ENV_ADD_ENTRYPOINT.php)).

The **Dockfile** can be re-written as the following:

```
FROM node:12-alpine
WORKDIR /app
COPY . .
RUN yarn install --production
ENTRYPOINT ["node"]
CMD ["src/index.js"]
```

With the new **Dockerfile**, we can pass an arg to our `docker run` command, for example:

```
$ docker run -d -p 3000:3000 getting-started src/index.js
c85168de055f81f4390693c8029699266e69d304cdaf10e3c1c5a60d37040739
```

Here, though we used the same arg, as in `CMD ["src/index.js"]`, we could overwrite the arg provided by the `CMD` instruction.

Compose-MySQL.php)

MEAN Stack app on Docker containers : micro services (/MEAN-Stack/MEAN-Stack-NodeJS-Angular-Docker.php)

Docker Compose - Hashicorp's Vault and Consul Part A (install vault, unsealing, static secrets, and policies) (/DevOps/Docker/Docker-Vault-Consul.php)

Docker Compose - Hashicorp's Vault and Consul Part B (EaaS, dynamic secrets, leases, and revocation) (/DevOps/Docker/Docker-Vault-Consul-B.php)

Docker Compose - Hashicorp's Vault and Consul Part C (Consul) (/DevOps/Docker/Docker-Vault-Consul-C.php)

Docker Compose with two containers - Flask REST API service container and an Apache server container (/DevOps/Docker/Docker-Compose-FlaskREST-Service-Container-and-Apache-Container.php)

Docker compose : Nginx reverse proxy with multiple containers (/DevOps/Docker/Docker-Compose-Nginx-Reverse-Proxy-Multiple-Containers.php)

Docker compose : Nginx reverse proxy with multiple containers (/DevOps/Docker/Docker-Compose-Nginx-Reverse-Proxy-Multiple-Containers.php)

Docker & Kubernetes : Envoy - Getting started (/DevOps/Docker/Docker-Envoy-Getting-Started.php)

Docker & Kubernetes : Envoy - Front Proxy (/DevOps/Docker/Docker-Envoy-Front-Proxy.php)

Docker & Kubernetes : Ambassador - Envoy API Gateway on Kubernetes (/DevOps/Docker/Docker-Envoy-Ambassador-API-Gateway-for-Kubernetes.php)

For more about Dockerfile instructions, checkout the following list:

1. Dockerfile - Build Docker images automatically I - FROM, MAINTAINER, and build context (/DevOps/Docker/Docker_Dockerfile_to_build_images_automatically.php)
2. Dockerfile - Build Docker images automatically II - revisiting FROM, MAINTAINER, build context, and caching (/DevOps/Docker/Docker_Dockerfile_to_build_images_automatically_2.php)
3. Dockerfile - Build Docker images automatically III - RUN (/DevOps/Docker/Docker_Dockerfile_to_build_images_automatically_3.php)
4. Dockerfile - Build Docker images automatically IV - CMD (/DevOps/Docker/Docker_Dockerfile_to_build_images_automatically_4_CMD.php)
5. Dockerfile - Build Docker images automatically V - WORKDIR, ENV, ADD, and ENTRYPOINT (/DevOps/Docker/Docker_Dockerfile_to_build_images_automatically_5_WORKDIR_ENV_ADD_ENTRYPOINT.php)

3. **Preferred way of removing containers?**

Let's stop the "busybox" container. First we need to do `docker stop`:

```
$ docker stop 2666137d9726
2666137d9726
```

and then followed by a `docker rm`:

```
$ docker rm 2666137d9726
2666137d9726
```

4. **Difference between Docker pause and stop.**

```
$ docker run -it -d --name=busybox1 busybox /bin/sh
007d5db147718b9fe6e2d2d4054fc6e1683836a74c4df11d08458e7d2a7e7018

$ docker run -it -d --name=busybox2 busybox /bin/sh
d00834397e505f86979ede9c9bd97bc2d4945d3f0c8f8486721a04165661ec68

$ docker ps
CONTAINER ID        IMAGE               COMMAND             CREATED             STATUS
d00834397e50        busybox             "/bin/sh"           4 seconds ago       Up 4 seconds
007d5db14771        busybox             "/bin/sh"           13 seconds ago      Up 13 second
```

Docker Packer (/DevOps/Docker/Docker-Packer.php)

Docker Cheat Sheet (/DevOps/Docker/Docker-Cheat-Sheet.php)

Docker Q & A (/DevOps/Docker/Docker_Q_and_A

Kubernetes Q & A - Part I (/DevOps/Docker/Docker_Kuberne

Kubernetes Q & A - Part II (/DevOps/Docker/Docker_Kuberne

Docker - Run a React app in a docker (/DevOps/Docker/Docker-React-App.php)

Docker - Run a React app in a docker II (snapshot app with nginx) (/DevOps/Docker/Docker-React-App-2-SnapShot.php)

Docker - NodeJS and MySQL app with React in a docker (/DevOps/Docker/Docker-React-Node-MySQL-App.php)

Docker - Step by Step NodeJS and MySQL app with React - I (/DevOps/Docker/Step-by-Step-React-Node-MySQL-App.php)

Installing LAMP via puppet on Docker (/DevOps/Docker/Installing-LAMP-with-puppet-on-Docker.php)

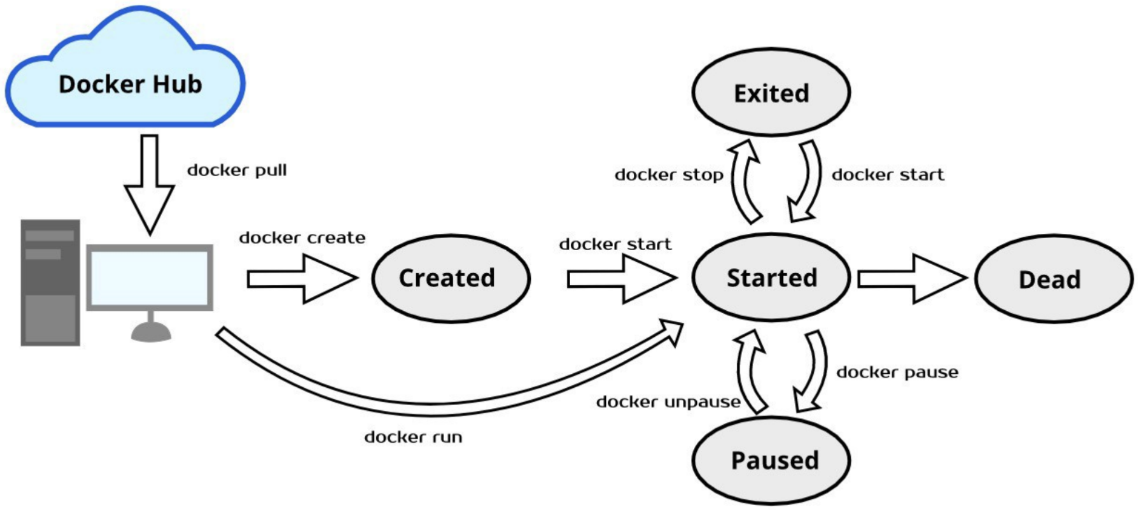
Docker install via Puppet (/DevOps/Docker/Docker_puppet.p

Nginx Docker install via Ansible (/DevOps/Ansible/Ansible-Deploy-Nginx-to-Docker.php)

Apache Hadoop CDH 5.8 Install with QuickStarts Docker (/Hadoop/BigData_hadoop_CDH5.8

Docker - Deploying Flask app to ECS (/DevOps/Docker/Docker-Flask-ALB-ECS.php)

Docker Compose - Deploying WordPress to AWS (/DevOps/Docker/Docker-Compose-WordPress-AWS.php)



Picture credit: Get Started with Docker Lifecycle (https://medium.com/future-vision/docker-lifecycle-tutorial-and-quickstart-guide-c5fd5b987e0d)

We can stop the "busybox1" and pause the "busybox2":

```
$ docker stop busybox1
busybox1

$ docker pause busybox2
busybox2

$ docker ps
CONTAINER ID        IMAGE               COMMAND             CREATED             STATUS
21f6d14521b8        busybox             "/bin/sh"           22 minutes ago      Up 45 second
```

The `docker pause` command suspends (via SIGSTOP signal) all processes in the specified containers.

The `docker stop` command. The main process inside the container will receive SIGTERM, and after a grace period, SIGKILL.

We cannot remove a paused container.

```
$ docker rm busybox2
Error response from daemon: You cannot remove a paused container 21f6d14521b8c16455166536000
Unpause and then stop the container before attempting removal or force remove
```

SIGSTOP is the pause signal that cannot be caught or ignored. The shell uses pausing (and its counterpart, resuming via SIGCONT) to implement job control.

```
$ docker restart busybox2
busybox2

$ docker ps
CONTAINER ID        IMAGE               COMMAND             CREATED             STATUS
21f6d14521b8        busybox             "/bin/sh"           About an hour ago   Up 5 seconds
55ae35026d6c        ubuntu:18.04        "/bin/bash"         6 days ago          Up 6 days
```

Docker - WordPress Deploy to ECS with Docker-Compose (ECS-CLI EC2 type) (/DevOps/Docker/Docker-ECS-CLI-Docker-Compose-Wordpress-EC2-Type.php)

Docker - AWS ECS service discovery with Flask and Redis (/DevOps/Docker/Docker-ALB-ECS-Fargate.php)

Docker - ECS Fargate (/DevOps/Docker/Docker-ECS-Service-Dicsovery-Redis-Flask.php)

Docker & Kubernetes 1 : minikube (/DevOps/Docker/Docker_Kuberne

Docker & Kubernetes 2 : minikube Django with Postgres - persistent volume (/DevOps/Docker/Docker_Kuberne

Docker & Kubernetes 3 : minikube Django with Redis and Celery (/DevOps/Docker/Docker_Kuberne

Docker & Kubernetes 4 : Django with RDS via AWS Kops (/DevOps/Docker/Docker_Kuberne

Docker & Kubernetes : Kops on AWS (/DevOps/DevOps-Kubernetes-II-kops-on-AWS.php)

Docker & Kubernetes : Ingress controller on AWS with Kops (/DevOps/Docker/Docker-Kubernetes-kops-on-AWS-Ingress.php)

Docker & Kubernetes : HashiCorp's Vault and Consul on minikube (/DevOps/Docker/Docker_Kuberne

Docker & Kubernetes : HashiCorp's Vault and Consul - Auto-unseal using Transit Secrets Engine (/DevOps/Docker/Docker_Kuberne

Docker & Kubernetes : Persistent Volumes & Persistent Volumes Claims - hostPath and annotations (/DevOps/Docker/Docker_Kuberne

Docker & Kubernetes : Persistent

5. **How can we create a Docker container in the Stopped state?**

```
$ docker create --name MyContainer ubuntu
692b7ce25a7743907be877e7a758fd6a16b390a09275d73ade607dd25c4b0ee9
```

We can see that it has created a new container. But we won't see MyContainer because though it was created, it was never started.

```
$ docker ps -a
CONTAINER ID        IMAGE               COMMAND                   CREATED             STATUS
692b7ce25a77        ubuntu              "/bin/bash"               2 minutes ago       Created
```

We can start this container with `docker start` command:

```
$ docker start MyContainer
MyContainer
```

6. `docker stats`:
The docker stats command returns a live data stream for running containers.

```
$ docker ps
CONTAINER ID        IMAGE               COMMAND               CREATED             STATUS
c85168de055f        getting-started     "node src/index.js"   2 hours ago         Up 2 hours
55ae35026d6c        ubuntu:18.04        "/bin/bash"           8 days ago          Up 8 days

$ docker stats
CONTAINER ID        NAME                  CPU %               MEM USAGE / LIMIT     MEM %
c85168de055f        modest_easley         0.00%               16.38MiB / 2.434GiB   0.66%
55ae35026d6c        laughing_mcclintock   0.00%               1.125MiB / 2.434GiB   0.05%


$ docker stats c85168de055f
CONTAINER ID        NAME                CPU %               MEM USAGE / LIMIT     MEM %
c85168de055f        modest_easley       0.00%               16.38MiB / 2.434GiB   0.66%
```

7. `docker system prune`:
It's a command used to remove all stopped containers, unused networks, build caches, and dangling images. The `prune` is one of the most useful commands in Docker:

Volumes - Dynamic volume provisioning (/DevOps/Docker/Docker_Kuberne

Docker & Kubernetes : DaemonSet (/DevOps/Docker/Docker_Kuberne

Docker & Kubernetes : Secrets (/DevOps/Docker/Docker_Kuberne

Docker & Kubernetes : kubectl command (/DevOps/Docker/Docker_Kuberne

Docker & Kubernetes : Assign a Kubernetes Pod to a particular node in a Kubernetes cluster (/DevOps/Docker/Docker_Kuberne

Docker & Kubernetes : Configure a Pod to Use a ConfigMap (/DevOps/Docker/Docker_Kuberne

AWS : EKS (Elastic Container Service for Kubernetes) (/DevOps/AWS/aws-EKS-Elastic-Container-Service-Kubernetes.php)

Docker & Kubernetes : Run a React app in a minikube (/DevOps/Docker/Docker-Kubernetes-React-App.php)

Docker & Kubernetes : Minikube install on AWS EC2 (/DevOps/Docker/Docker-Kubernetes-Minikube-install-on-AWS-EC2.php)

Docker & Kubernetes : Cassandra with a StatefulSet (/DevOps/Docker/Docker_Kuberne

Docker & Kubernetes : Terraform and AWS EKS (/DevOps/Docker/Docker_Kuberne

Docker & Kubernetes : Pods and Service definitions (/DevOps/Docker/Docker_Kuberne

Docker & Kubernetes : Headless service and discovering pods (/DevOps/Docker/Docker_Kuberne

Docker & Kubernetes : Service IP and the Service Type (/DevOps/Docker/Docker_Kuberne

Docker & Kubernetes : Kubernetes DNS with Pods and

```
$ docker system prune
WARNING! This will remove:
  - all stopped containers
  - all networks not used by at least one container
  - all dangling images
  - all dangling build cache

Are you sure you want to continue? [y/N] y
Deleted Containers:
1470e09b748c5a89a41a415d4bdfbeec61e4091d38eaceef94c96ac9edb90469
692b7ce25a7743907be877e7a758fd6a16b390a09275d73ade607dd25c4b0ee9
...

Deleted Images:
deleted: sha256:295a1b181e50c37a3a9595bd498b5c980a9f90823473abdb8704ce3308628eef
deleted: sha256:77234e845dbed0075aecab391e14bb9f1a34ec7bd34b88284d2f74b31b9837b0
...

Total reclaimed space: 89.85MB
```

8. **docker-compose**:

   Most of the time, we will most likely want to bring up all of the services listed in our **docker-compose.yml** and have the containers run their default command, so we would want to use `docker-compose up`.

   The `docker-compose run` command will spin up a new container for us to use while the `docker-compose exec` command will allow us to use a container that is already running.

9. **Multistage Image Builds**:

   While using containers to build applications can be useful, it is important to distinguish between the **build image** and the **runtime image**.

   The build image contains all the tooling and libraries that are necessary to compile the application, while the runtime image contains the application to be deployed. A Java application has a build image that contains the JDK, Gradle/Maven, and compilation and testing tooling. Then our runtime image can contain only the Java runtime and our application.

   Compiling code as part of the image build is the most common ways of accidentally building large images.

   To resolve this issue, Docker introduced **multistage builds**. Rather than producing a single image, with the multistage builds, a Docker file can actually produce multiple images where each image is considered a stage. Artifacts can be copied from preceding stages to the current stage.

   The following two Dockerfiles demonstrate the multistage image builds from Use multi-stage builds (https://docs.docker.com/develop/develop-images/multistage-build/):

   Dockerfile with number:

Services
(/DevOps/Docker/Docker_Kuberne

Docker & Kubernetes - Scaling and Updating application
(/DevOps/Docker/Docker_Kuberne

Docker & Kubernetes : Horizontal pod autoscaler on minikubes
(/DevOps/Docker/Docker-Kubernetes-Horizontal-Pod-Autoscaler.php)

Docker & Kubernetes : NodePort vs LoadBalancer vs Ingress
(/DevOps/Docker/Docker_Kuberne

Docker & Kubernetes : Load Testing with Locust on GCP Kubernetes
(/DevOps/Docker/Docker-Load-Testing-with-Locust-on-GCP-Kubernetes.php)

Docker & Kubernetes : From a monolithic app to micro services on GCP Kubernetes
(/DevOps/Docker/Docker-from-Monolithic-to-Micro-services-GCP-Kubernetes.php)

Docker & Kubernetes : Rolling updates
(/DevOps/Docker/Docker-Kubernetes-Rolling-Updates.php)

Docker & Kubernetes : Deployments to GKE (Rolling update, Canary and Blue-green deployments)
(/DevOps/Docker/Docker-Rolling-Update-Canary-Blue-Green-Deployments-to-GKE-Kubernetes.php)

Docker & Kubernetes : Slack Chat Bot with NodeJS on GCP Kubernetes
(/DevOps/Docker/Docker-Slack-NodeJS-ChatBot-GCP-Kubernetes.php)

Docker & Kubernetes : Continuous Delivery with Jenkins Multibranch Pipeline for Dev, Canary, and Production Environments on GCP Kubernetes
(/DevOps/Docker/Docker-Continuous-Delivery-with-Jenkins-Multibranch-Pipeline-for-Dev-Canary-Production-

```
FROM golang:1.7.3
WORKDIR /go/src/github.com/alexellis/href-counter/
RUN go get -d -v golang.org/x/net/html
COPY app.go .
RUN CGO_ENABLED=0 GOOS=linux go build -a -installsuffix cgo -o app .

FROM alpine:latest
RUN apk --no-cache add ca-certificates
WORKDIR /root/
COPY --from=0 /go/src/github.com/alexellis/href-counter/app .
CMD ["./app"]
```

The second FROM instruction starts a new build stage with the alpine:latest image as its base. The `COPY --from=0` line copies just the built artifact from the previous stage into this new stage. The Go SDK and any intermediate artifacts are left behind, and not saved in the final image.

Dockerfile with name:

```
FROM golang:1.7.3 AS builder
WORKDIR /go/src/github.com/alexellis/href-counter/
RUN go get -d -v golang.org/x/net/html
COPY app.go    .
RUN CGO_ENABLED=0 GOOS=linux go build -a -installsuffix cgo -o app .

FROM alpine:latest
RUN apk --no-cache add ca-certificates
WORKDIR /root/
COPY --from=builder /go/src/github.com/alexellis/href-counter/app .
CMD ["./app"]
```

By default, the stages are not named, and we refer to them by their integer number, starting with 0 for the first `FROM` instruction as shown in the first Dockerfile. However, we can name our stages, by adding an `AS <NAME>` to the `FROM` instruction. The 2nd example improves the 1st one by naming the stages and using the name in the `COPY` instruction.

Here is another sample that builds a "Go" application and runs the app.

hello.go:

Environments-GCP-Kubernetes-Engine-Namespace.php)

Docker & Kubernetes - MongoDB with StatefulSets on GCP Kubernetes Engine (/DevOps/Docker/Docker_MongoD

Docker & Kubernetes : Nginx Ingress Controller on minikube (/DevOps/Docker/Docker_Kuberne

Docker & Kubernetes : Nginx Ingress Controller for Dashboard service on Minikube (/DevOps/Docker/Docker_Kuberne

Docker & Kubernetes : Nginx Ingress Controller on GCP Kubernetes (/DevOps/Docker/Docker_Kuberne

Docker & Kubernetes : Kubernetes Ingress with AWS ALB Ingress Controller in EKS (/DevOps/Docker/Docker-Kubernetes-ALB-Ingress-Controller-with-EKS.php)

Docker & Kubernetes : MongoDB / MongoExpress on Minikube (/DevOps/Docker/Docker_Kuberne

Docker & Kubernetes : Setting up a private cluster on GCP Kubernetes (/DevOps/Docker/Docker-setting-up-private-cluster-on-GCP-Kubernetes.php)

Docker & Kubernetes : Kubernetes Namespaces (default, kube-public, kube-system) and switching namespaces (kubens) (/DevOps/Docker/Docker-Kubernetes-Namespaces.php)

Docker & Kubernetes : StatefulSets on minikube (/DevOps/Docker/Docker_Kuberne

Docker & Kubernetes : StatefulSets on minikube (/DevOps/Docker/Docker_Kuberne

Docker & Kubernetes : RBAC (/DevOps/Docker/Docker-Kubernetes-RBAC.php)

Docker & Kubernetes Service Account, RBAC, and IAM (/DevOps/Docker/Docker-

```
package main

import (
        "fmt"
        "log"
        "net/http"
)

//Hello Server responds to requests with the given URL path.
func HelloServer(w http.ResponseWriter, r *http.Request) {
        fmt.Fprintf(w, "Hello, you requested: %s", r.URL.Path)
        log.Printf("Received request for path: %s", r.URL.Path)
}
func main() {
        var addr string = ":8181"
        handler := http.HandlerFunc(HelloServer)
        if err := http.ListenAndServe(addr, handler); err != nil {
                log.Fatalf("Could not listen on port %s %v", addr, err)
        }
}
```

Dockerfile that builds the app then copy the binary into a container:

```
FROM golang:1-alpine as build
WORKDIR /app
COPY hello.go /app
RUN go build hello.go

FROM alpine:latest
WORKDIR /app
COPY --from=build /app /app
EXPOSE 8180
ENTRYPOINT ["./hello"]
```

Now, we have the following files in our working directory:

```
$ ls
Dockerfile      hello.go
```

We are now ready to build the image from the Dockerfile:

Kubernetes-Service-Account.php)

Docker & Kubernetes - Kubernetes Service Account, RBAC, IAM with EKS ALB, Part 1 (/DevOps/Docker/Docker-Kubernetes-ALB-on-EKS-1.php)

Docker & Kubernetes : Helm Chart (/DevOps/Docker/Docker_Helm_Ch

Docker & Kubernetes : My first Helm deploy (/DevOps/Docker/Docker_Kuberne

Docker & Kubernetes : Readiness and Liveness Probes (/DevOps/Docker/Docker-Kubernetes-Readiness-Liveness-Probes.php)

Docker & Kubernetes : Helm chart repository with Github pages (/DevOps/Docker/Docker_Helm_Ch

Docker & Kubernetes : Deploying WordPress and MariaDB with Ingress to Minikube using Helm Chart (/DevOps/Docker/Docker_Helm_Ch

Docker & Kubernetes : Deploying WordPress and MariaDB to AWS using Helm 2 Chart (/DevOps/Docker/Docker_Helm_Ch

Docker & Kubernetes : Deploying WordPress and MariaDB to AWS using Helm 3 Chart (/DevOps/Docker/Docker_Helm3_C

Docker & Kubernetes : Helm Chart for Node/Express and MySQL with Ingress (/DevOps/Docker/Docker_Helm_Ch

Docker & Kubernetes : Docker_Helm_Chart_Node_Expess_ (/DevOps/Docker/Docker_Helm_Ch

Docker & Kubernetes: Deploy Prometheus and Grafana using Helm and Prometheus Operator - Monitoring Kubernetes node resources out of the box (/DevOps/Docker/Docker_Kuberne

Docker & Kubernetes : Istio (service mesh) sidecar proxy on GCP Kubernetes (/DevOps/Docker/Docker_Kuberne

```
$ docker build -t hello-go .
Sending build context to Docker daemon  3.072kB
Step 1/9 : FROM golang:1-alpine as build
 ---> 14ee78639386
Step 2/9 : WORKDIR /app
 ---> Using cache
 ---> 295af3c2ffa0
Step 3/9 : COPY hello.go /app
 ---> Using cache
 ---> debfbdb3c01d
Step 4/9 : RUN go build hello.go
 ---> Using cache
 ---> e29f2ba09000
Step 5/9 : FROM alpine:latest
 ---> 49f356fa4513
Step 6/9 : WORKDIR /app
 ---> Running in 034e0cf20138
Removing intermediate container 034e0cf20138
 ---> 03048fdf20d8
Step 7/9 : COPY --from=build /app /app
 ---> f496e0e4bbcb
Step 8/9 : EXPOSE 8180
 ---> Running in 7f2b631302e1
Removing intermediate container 7f2b631302e1
 ---> c1545d176662
Step 9/9 : ENTRYPOINT ["./hello"]
 ---> Running in dfeb253925e1
Removing intermediate container dfeb253925e1
 ---> 10456b843247
Successfully built 10456b843247
Successfully tagged hello-go:latest

$ docker images
REPOSITORY                    TAG                 IMAGE ID            CREATED             S
hello-go                      latest              10456b843247        About a minute ago  1
```

To run the container and expose the internal port 8181 to our host port 8182:

```
$ docker run -d --name hello-go-container --rm -p 8182:8181 hello-go
907ad5290aa7a2db70496082906d25a291a5077933d343f6a8a7d50efa0760c7

$ docker ps
CONTAINER ID        IMAGE       COMMAND    CREATED             STATUS              PORTS
907ad5290aa7        hello-go    "./hello"  16 seconds ago      Up 15 seconds       8180/tcp,
```

Test the application:

```
$ curl localhost:8182
Hello, you requested: /
```

It appears to be working fine!

One more step. Let's go into the container and check the **/app** folder:

Docker & Kubernetes : Istio on EKS (/DevOps/Docker/Docker-Kubernetes-EKS-with-ISTIO.php)

Docker & Kubernetes : Deploying .NET Core app to Kubernetes Engine and configuring its traffic managed by Istio (Part I) (/DevOps/Docker/Docker_Kuberne

Docker & Kubernetes : Deploying .NET Core app to Kubernetes Engine and configuring its traffic managed by Istio (Part II - Prometheus, Grafana, pin a service, split traffic, and inject faults) (/DevOps/Docker/Docker_Kuberne

Docker & Kubernetes : Helm Package Manager with MySQL on GCP Kubernetes Engine (/DevOps/Docker/Docker_Helm_Pa

Docker & Kubernetes : Deploying Memcached on Kubernetes Engine (/DevOps/Docker/Docker_Helm_Pa

Docker & Kubernetes : EKS Control Plane (API server) Metrics with Prometheus (/DevOps/Docker/Docker-Kubernetes-EKS-Control-Plane-API-Server-Metrics-with-Prometheus.php)

Docker & Kubernetes : Spinnaker on EKS with Halyard (/DevOps/Docker/Docker_Kuberne

Docker & Kubernetes : Continuous Delivery Pipelines with Spinnaker and Kubernetes Engine (/DevOps/Docker/Docker_Kuberne

Docker & Kubernetes: Multi-node Local Kubernetes cluster - Kubeadm-dind(docker-in-docker) (/DevOps/Docker/Docker-Kubernetes-Multi-Node-Local-Clusters-dind.php)

Docker & Kubernetes: Multi-node Local Kubernetes cluster - Kubeadm-kind(k8s-in-docker) (/DevOps/Docker/Docker-Kubernetes-Multi-Node-Local-Clusters-kind.php)

Docker & Kubernetes :

```
$ docker exec -it hello-go-container /bin/sh
/app # ls -la
total 6044
drwxr-xr-x    1 root     root          4096 Apr  6 23:47 .
drwxr-xr-x    1 root     root          4096 Apr  6 23:53 ..
-rwxr-xr-x    1 root     root       6176661 Apr  6 21:23 hello
-rw-r--r--    1 root     root           493 Apr  6 20:22 hello.go
```

As we can see the binary (**hello**) has been copied successfully from **build** stage to the last stage.

10. **Sometimes when we run a docker container it exits immediately. Why?**

The short answer is that the container exits because it has no process to run.

In this section, we'll also learn the difference between `CMD` and `ENTRYPOINT`

When we run an Ubuntu image, it exits immediately as we can see below:

```
$ docker run ubuntu:18.04
Unable to find image 'ubuntu:18.04' locally
18.04: Pulling from library/ubuntu
6cf436f81810: Pull complete
987088a85b96: Pull complete
b4624b3efe06: Pull complete
d42beb8ded59: Pull complete
Digest: sha256:7a47ccc3bbe8a451b500d2b53104868b46d60ee8f5b35a24b41a86077c650210
Status: Downloaded newer image for ubuntu:18.04

$ docker ps -a
CONTAINER ID IMAGE        COMMAND     CREATED        STATUS                         PORTS
9ba1aa158caf ubuntu:18.04 "/bin/bash" 11 seconds ago Exited (0) 10 seconds ago
```

Why is that? Why it exited?

Unlike VMs which are meant to host OS, containers are meant to run a task or a process such as a web server/application or a db. So, once a task is complete, a container exits. A container lives as long as a process within it is running. If an application in a container crashes, container exits.

So, who defines which process should be running inside a container?

Let's look into the following Dockerfile for nginx (), specially the **CMD[]** instruction:

nodeSelector, nodeAffinity, taints/tolerations, pod affinity and anti-affinity - Assigning Pods to Nodes (/DevOps/Docker/Docker_Kuberne

Docker & Kubernetes : Jenkins-X on EKS (/DevOps/Docker/Docker_Kuberne X-EKS.php)

Docker & Kubernetes : ArgoCD App of Apps with Heml on Kubernetes (/DevOps/Docker/Docker_Kuberne

Docker & Kubernetes : ArgoCD on Kubernetes cluster (/DevOps/Docker/Docker_Kuberne

Sponsor Open Source development activities and free contents for everyone.

Donate with PayPal

Thank you.

- K Hong (http://bogotobogo.com/about_us.php)

```
#
# Nginx Dockerfile
#
# https://github.com/dockerfile/nginx
#

# Pull base image.
FROM dockerfile/ubuntu

# Install Nginx.
RUN \
  add-apt-repository -y ppa:nginx/stable && \
  apt-get update && \
  apt-get install -y nginx && \
  rm -rf /var/lib/apt/lists/* && \
  echo "\ndaemon off;" >> /etc/nginx/nginx.conf && \
  chown -R www-data:www-data /var/lib/nginx

# Define mountable directories.
VOLUME ["/etc/nginx/sites-enabled", "/etc/nginx/certs", "/etc/nginx/conf.d", "/var/log/nginx

# Define working directory.
WORKDIR /etc/nginx

# Define default command.
CMD ["nginx"]

# Expose ports.
EXPOSE 80
EXPOSE 443
```

Yes, the **CMD[]** tells the Docker which program should be run when the container starts. In our case, it is the "nginx" command.

For **mysql** Dockerfile it is **mysqld** command:

```
COPY docker-entrypoint.sh /entrypoint.sh
COPY healthcheck.sh /healthcheck.sh
ENTRYPOINT ["/entrypoint.sh"]
HEALTHCHECK CMD /healthcheck.sh
EXPOSE 3306 33060
CMD ["mysqld"]
```

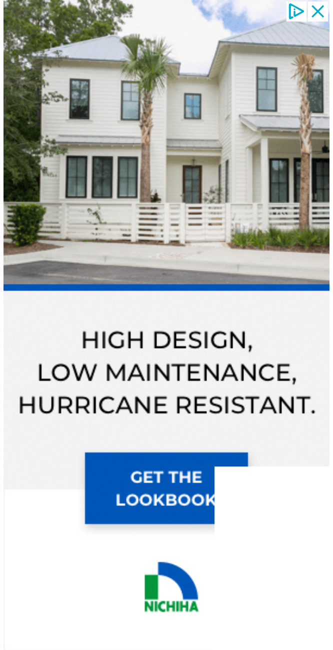
How about our Ubuntu image Dockerfile?

```
...
CMD ["/bin/bash"]
```

It uses **bash** for its default command.

Unlike the web server or a db, the **bash** is not a process, it's just a shell listening and waiting for an input. If it does not get any from a terminal, it exits.

Earlier, when we run a container from the Ubuntu image, it launches a "bash" program but the Docker, by default, not attaching any terminal to a container when it runs. So, the container could not find a terminal, and just exited.



Ansible 2.0

What is Ansible? (/DevOps/Ansible/Ansible_What_is

Quick Preview - Setting up web servers with Nginx, configure environments, and deploy an App (/DevOps/Ansible/Ansible_SettingU

SSH connection & running commands (/DevOps/Ansible/Ansible-SSH-Connection-Setup-Run-Command.php)

Ansible: Playbook for Tomcat 9 on Ubuntu 18.04 systemd with AWS (/DevOps/Ansible/Ansible-Tomcat9-Ubuntu18-Playbook.php)

Modules (/DevOps/Ansible/Ansible-Modules.php)

Playbooks (/DevOps/Ansible/Ansible-Playbooks.php)

We can make container alive for a while by overwriting the CMD ["/bin/bash"], for example, sleep 30s when we run docker:

```
$ docker run ubuntu:18.04 sleep 30s

$ docker ps
CONTAINER ID  IMAGE          COMMAND       CREATED         STATUS         PORTS  NAMES
55ab52fa884d  ubuntu:18.04   "sleep 30s"   7 seconds ago   Up 6 seconds          relaxed_eule
```

But how we can make the container always run the `sleep` command when it starts? Note that we added it to the `docker run` command.

One way to avoid adding the "sleep 30s" after the command is to use the `CMD` instruction in our Dockerfile:

```
FROM ubuntu:18.04
CMD sleep 30
```

Or we can use array:

```
FROM ubuntu:18.04
CMD ["sleep", "30"]
```

Note that we should NOT use the following because the command and args should be separated:

```
CMD ["sleep 30"] X
```

Now we can build our image with a name of "ubuntu-sleep":

```
$ docker build -t ubuntu-sleep .
Sending build context to Docker daemon  2.048kB
Step 1/2 : FROM ubuntu:18.04
 ---> 47b19964fb50
Step 2/2 : CMD ["sleep", "30"]
 ---> Running in c84ecc7a5b3d
Removing intermediate container c84ecc7a5b3d
 ---> 3f21ee94c150
Successfully built 3f21ee94c150
Successfully tagged ubuntu-sleep:latest
```

Then, run a container from the newly created image:

```
$ docker run ubuntu-sleep
```

The container always sleeps 30s after it started!

But we have a problem. What if we want to change the sleep time?

Currently, it's been hard-coded.

Handlers (/DevOps/Ansible/Ansible-Handlers.php)

Roles (/DevOps/Ansible/Ansible-Roles.php)

Playbook for LAMP HAProxy (/DevOps/Ansible/Ansible-Playbook-Lamp-HAProxy.php)

Installing Nginx on a Docker container (/DevOps/Ansible/Ansible-Deploy-Nginx-to-Docker.php)

AWS : Creating an ec2 instance & adding keys to authorized_keys (/DevOps/Ansible/Ansible-aws-creating-ec2-instance.php)

AWS : Auto Scaling via AMI (/DevOps/Ansible/Ansible-aws-AutoScaling.php)

AWS : creating an ELB & registers an EC2 instance from the ELB (/DevOps/Ansible/Ansible-aws-creating-elb-and-register-ec2-instance.php)

Deploying Wordpress micro-services with Docker containers on Vagrant box via Ansible (/DevOps/Ansible/Docker-WordPress-Microservices-with-Nginx-reverse-proxy-Varnish-Mysql-Deployed-via-Ansible.php)

Setting up Apache web server (/DevOps/Ansible/Ansible_SettingU

Deploying a Go app to Minikube (/DevOps/Ansible/Ansible-Deploying-a-Go-App-to-Minikube.php)

Ansible with Terraform (/DevOps/Ansible/Ansible-Terraform-null_resource-local-exec-remote-exec-triggers.php)

Terraform

Introduction to Terraform with AWS elb & nginx

Of course, we can overwrite the command like this:

```
$ docker run ubuntu-sleep sleep 5
```

However, because the image name itself is already indicating it would sleep, we need to find a way of just feeding the seconds as an argument not with the `sleep` command, and the image automatically invoke the "sleep" command needing only the parameter. Something like this:

```
$ docker run ubuntu-sleep 5
```

That's why we need the `ENTRYPOINT` instruction.

It simply specifies a program to run when a container starts.

So, our Dockerfile should be changed from:

```
FROM ubuntu:18.04
CMD ["sleep", "30"]
```

to:

```
FROM ubuntu:18.04
ENTRYPOINT ["sleep"]
```

Build a new image and run the container:

```
$ docker build -t ubuntu-sleep .
Sending build context to Docker daemon  2.048kB
Step 1/2 : FROM ubuntu:18.04
 ---> 47b19964fb50
Step 2/2 : ENTRYPOINT ["sleep"]
 ---> Running in e5e6e83e9e01
Removing intermediate container e5e6e83e9e01
 ---> affbc2e6ed86
Successfully built affbc2e6ed86
Successfully tagged ubuntu-sleep:latest

$ docker run ubuntu-sleep 5
```

Note the difference between the `CMD` and `ENTRYPOINT` with related to the supplied to the `docker run` command. While the `CMD` will be completely over-written by the supplied command (or args), for the `ENTRYPOINT`, the supplied command will be appended to it.

Another problem in our Dockerfile: let's see:

```
$ docker run ubuntu-sleep
sleep: missing operand
Try 'sleep --help' for more information.
```

(/DevOps/Terraform/Terraform-Introduction-AWS-elb-nginx.php)

Terraform Tutorial - terraform format(tf) and interpolation(variables) (/DevOps/Terraform/Terraform-terraform-format-tf-and-interpolation-variables.php)

Terraform Tutorial - user_data (/DevOps/Terraform/Terraform-terraform-userdata.php)

Terraform Tutorial - variables (/DevOps/Terraform/Terraform-parameters-variables.php)

Terraform 12 Tutorial - Loops with count, for_each, and for (/DevOps/Terraform/Terraform-Introduction-AWS-loops.php)

Terraform Tutorial - creating multiple instances (count, list type and element() function) (/DevOps/Terraform/Terraform-creating-multiple-instances-count-list-type.php)

Terraform Tutorial - State (terraform.tfstate) & terraform import (/DevOps/Terraform/Terraform-state-tfstate-import.php)

Terraform Tutorial - Output variables (/DevOps/Terraform/Terraform-output-variables.php)

Terraform Tutorial - Destroy (/DevOps/Terraform/Terraform-destroy.php)

Terraform Tutorial - Modules (/DevOps/Terraform/Terraform-modules.php)

Terraform Tutorial - Creating AWS S3 bucket / SQS queue resources and notifying bucket event to queue (/DevOps/Terraform/Terraform-Introduction-AWS-S3-SQS.php)

Terraform Tutorial - AWS ASG and Modules (/DevOps/Terraform/Terraform-Introduction-AWS-ASG-Modules.php)

Terraform Tutorial - VPC,

In the command above, we did not supply an arg for the `sleep` command, and got an error when the container started.

We need a default value for the command so that container runs event though an arg is missing.

Here is where the `CMD` comes into play: the `CMD` instruction will be appended to the `ENTRYPOINT` instruction.

Here is our new Dockerfile:

```
FROM ubuntu:18.04
ENTRYPOINT ["sleep"]
CMD ["5"]
```

Build the image and run a container from the image, and we should not get any error when we do not specify sleep time:

```
$ docker build -t ubuntu-sleep .
$ docker run ubuntu-sleep
```

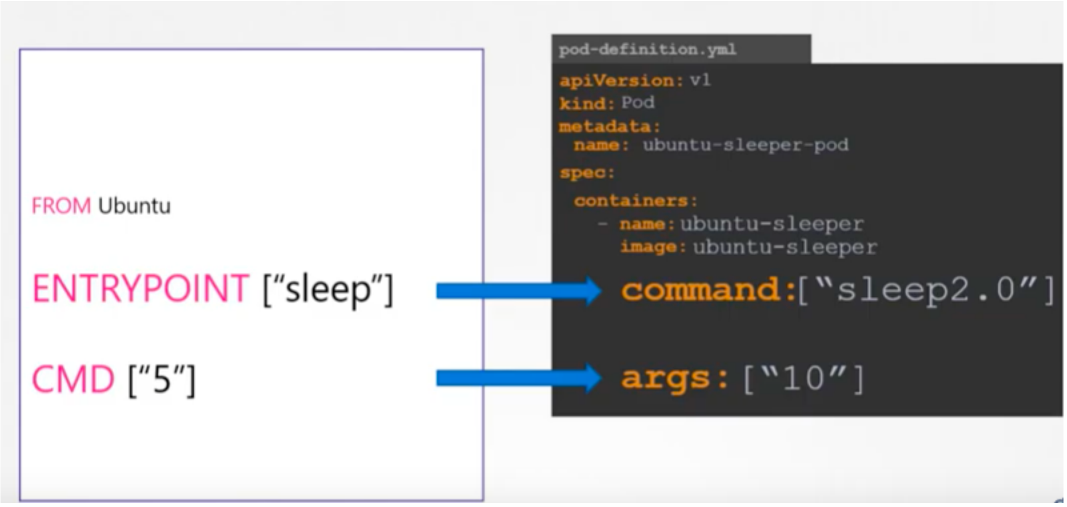
If we add a parameter to the command, it will overwrites the default value specified in `CMD`.

One more thing regarding the `ENTRYPOINT`. What if we want to override the command specified in the `ENTRYPOINT`?

In that case, we can give a new command in `docker run` command, for example:

```
$ docker run --entrypoint new-sleep-command ubuntu-sleep 60
```

Let's go further and look into how the `ENTRYPOINT` and `CMD` in Dockerfile are translated in a Pod definition yaml file:



```
FROM Ubuntu

ENTRYPOINT ["sleep"]

CMD ["5"]
```

```
pod-definition.yml
apiVersion: v1
kind: Pod
metadata:
  name: ubuntu-sleeper-pod
spec:
  containers:
    - name: ubuntu-sleeper
      image: ubuntu-sleeper
      command:["sleep2.0"]

      args: ["10"]
```

Picture source Docker for Beginners - Commands vs Entrypoint - Kubernetes (https://www.youtube.com/watch?v=OYbEWUbmk90&index=7&list=PL2We04F3Y_43dAehLMT5GxJhtk3mJtkl5)

Subnets, RouteTable, ELB, Security Group, and Apache server I (/DevOps/Terraform/Terraform-VPC-Subnet-ELB-RouteTable-SecurityGroup-Apache-Server-1.php)

Terraform Tutorial - VPC, Subnets, RouteTable, ELB, Security Group, and Apache server II (/DevOps/Terraform/Terraform-VPC-Subnet-ELB-RouteTable-SecurityGroup-Apache-Server-2.php)

Terraform Tutorial - Docker nginx container with ALB and dynamic autoscaling (/DevOps/Terraform/Terraform-docker-nginx-alb-dynamic-autoscaling.php)

Terraform Tutorial - AWS ECS using Fargate : Part I (/DevOps/Terraform/Terraform-ECS-1.php)

Hashicorp Vault (/DevOps/Terraform/Hashicorp-Vault.php)

HashiCorp Vault Agent (/DevOps/Terraform/Hashicorp-Vault-agent.php)

HashiCorp Vault and Consul on AWS with Terraform (/DevOps/Terraform/Hashicorp-Vault-and-Consul-on-AWS-with-Terraform.php)

Ansible with Terraform (/DevOps/Ansible/Ansible-Terraform-null_resource-local-exec-remote-exec-triggers.php)

AWS IAM user, group, role, and policies - part 1 (/DevOps/Terraform/Terraform_IA

AWS IAM user, group, role, and policies - part 2 (/DevOps/Terraform/Terraform_IA

Delegate Access Across AWS Accounts Using IAM Roles (/DevOps/Terraform/Terraform_AV

AWS KMS (/DevOps/Terraform/Terraform-AWS-KMS.php)

As we can see the parameters in `ENTRYPOINT` and `CMD` can be overwritten with the ones provided via "command" are "args" in "spec.containers" of the yaml.

11. **How we can run a container with "root" privilege?**

```
$ docker exec -u 0 -it my-container sh
```

12. **How to push an image to AWS ECR?**

Docker login first assuming the AWS credentials are in place either from ENV or from ~/.aws/credentials:

```
$ aws --region us-west-2 ecr get-login-password \
    | docker login \
        --password-stdin \
        --username AWS \
        437028470429.dkr.ecr.us-west-2.amazonaws.com

Login Succeeded
```

Then, push it to ECR:

```
$ docker push 437028470429.dkr.ecr.us-west-2.amazonaws.com/test-khong:0.2.1
The push refers to repository [437028470429.dkr.ecr.us-west-2.amazonaws.com/test-khong]
...
0.2.1: digest: sha256:ca2fc17f610...2ceaddf5e0cbda5e74 size: 4484
```

Terraform import (/DevOps/Terraform/Terraform_Im

Terraform commands cheat sheet (/DevOps/Terraform/Terraform_co

Terraform Cloud (/DevOps/Terraform/Terraform-Cloud.php)

Terraform 14 (/DevOps/Terraform/Terraform14.

Creating Private TLS Certs (/DevOps/Terraform/Terraform-private-tls-certs.php)

DevOps

Phases of Continuous Integration (/DevOps/Continuous_Integration_

Software development methodology (/DesignPatterns/software_develop

Introduction to DevOps (/DevOps/DevOps_Jenkins_Chef_Pu

Samples of Continuous Integration (CI) / Continuous Delivery (CD) - Use cases (/DevOps/DevOps_CI_CD_Pipeline_

Artifact repository and repository management (/DevOps/DevOps_Artifacts_Artifac

Linux - General, shell programming, processes & signals ... (/Linux/linux_tips1.php)

RabbitMQ... (/python/RabbitMQ_Celery/python

MariaDB (/DevOps/DevOps_MariaDB.php)

New Relic APM with NodeJS : simple agent setup on AWS instance (/DevOps/DevOps_NewRelic-APM-Application-Performance-Management-setup.php)

Docker & K8s

1. Docker install on Amazon Linux AMI (/DevOps/Docker/Docker_Install_On_Amazon_Linux_AMI.php)
2. Docker install on EC2 Ubuntu 14.04 (/DevOps/Docker/Docker_Install_On_EC2_Ubuntu.php)
3. Docker container vs Virtual Machine (/DevOps/Docker/Docker_Container_vs_Virtual_Machine.php)
4. Docker install on Ubuntu 14.04 (/DevOps/Docker/Docker_Install_On_Ubuntu_14.php)
5. Docker Hello World Application (/DevOps/Docker/Docker_Hello_World_Application.php)
6. Nginx image - share/copy files, Dockerfile (/DevOps/Docker/Docker_Nginx_WebServer.php)
7. Working with Docker images : brief introduction (/DevOps/Docker/Docker_Working_with_images.php)
8. Docker image and container via docker commands (search, pull, run, ps, restart, attach, and rm) (/DevOps/Docker/Docker_Commands_for_Images_Container.php)
9. More on docker run command (docker run -it, docker run --rm, etc.) (/DevOps/Docker/Docker_Run_Command.php)
10. Docker Networks - Bridge Driver Network (/DevOps/Docker/Docker-Bridge-Driver-Networks.php)
11. Docker Persistent Storage (/DevOps/Docker/Docker_Container_Persistent_Storage_Data_Share.php)
12. File sharing between host and container (docker run -d -p -v) (/DevOps/Docker/Docker_File_Share_between_Host_and_Container.php)
13. Linking containers and volume for datastore (/DevOps/Docker/Docker_Container_Linking_Connect_with_linking_system_Communication_across_links_Environment_variables.php)
14. Dockerfile - Build Docker images automatically I - FROM, MAINTAINER, and build context (/DevOps/Docker/Docker_Dockerfile_to_build_images_automatically.php)
15. Dockerfile - Build Docker images automatically II - revisiting FROM, MAINTAINER, build context, and caching (/DevOps/Docker/Docker_Dockerfile_to_build_images_automatically_2.php)
16. Dockerfile - Build Docker images automatically III - RUN (/DevOps/Docker/Docker_Dockerfile_to_build_images_automatically_3.php)
17. Dockerfile - Build Docker images automatically IV - CMD (/DevOps/Docker/Docker_Dockerfile_to_build_images_automatically_4_CMD.php)
18. Dockerfile - Build Docker images automatically V - WORKDIR, ENV, ADD, and ENTRYPOINT (/DevOps/Docker/Docker_Dockerfile_to_build_images_automatically_5_WORKDIR_ENV_ADD_ENTRYPOINT.php)
19. Docker - Apache Tomcat (/DevOps/Docker/Docker_Apache_Tomcat.php)
20. Docker - NodeJS (/DevOps/Docker/Docker-NodeJS.php)
21. Docker - NodeJS with hostname (/DevOps/Docker/Docker-NodeJS-with-hostname.php)
22. Docker Compose - NodeJS with MongoDB (/DevOps/Docker/Docker-Compose-Node-MongoDB.php)
23. Docker - Prometheus and Grafana with Docker-compose (/DevOps/Docker/Docker_Prometheus_Grafana.php)
24. Docker - StatsD/Graphite/Grafana (/DevOps/Docker/Docker_StatsD_Graphite_Grafana.php)
25. Docker - Deploying a Java EE JBoss/WildFly Application on AWS Elastic Beanstalk Using Docker Containers (/DevOps/Docker/Docker_Container_Deploy_via_AWS_Beanstalk_J2EE_JBoss_WildFly_app.php)

Nagios on CentOS 7 with Nagios Remote Plugin Executor (NRPE) (/DevOps/DevOps_CentOS_Nagios Remote-Plugin-Executor-NRPE.php)

Nagios - The industry standard in IT infrastructure monitoring on Ubuntu (/DevOps/DevOps_Nagios_Infrastr Remote-Plugin-Executor-NRPE.php)

Zabbix 3 install on Ubuntu 14.04 & adding hosts / items / graphs (/DevOps/DevOps-Zabbix3-Server-and-Agent-Install-Ubuntu14-Adding-Hosts-Items-Graphs.php)

Datadog - Monitoring with PagerDuty/HipChat and APM (/DevOps/DevOps-Monitoring-with-Datadog-PagerDuty-HipChat.php)

Install and Configure Mesos Cluster (/DevOps/DevOps_Mesos_Install.p

Cassandra on a Single-Node Cluster (/DevOps/DevOps-Cassandra-On-A-Single-Node-Cluster.php)

Container Orchestration : Docker Swarm vs Kubernetes vs Apache Mesos (/DevOps/DevOps-Docker-Swarm-vs-Kubernetes-vs-Apache-Mesos.php)

OpenStack install on Ubuntu 16.04 server - DevStack (/DevOps/OpenStack-Install-On-Ubuntu-16-Server.php)

AWS EC2 Container Service (ECS) & EC2 Container Registry (ECR) | Docker Registry (/DevOps/DevOps-ECS-ECR.php)

CI/CD with CircleCI - Heroku deploy (/DevOps/DevOps-CircleCI-Heroku-Deploy.php)

Introduction to Terraform with AWS elb & nginx (/DevOps/DevOps-Terraform.php)

Docker & Kubernetes (/DevOps/Docker/Docker_Kuberne

26. Docker : NodeJS with GCP Kubernetes Engine (/DevOps/Docker/Docker-NodeJS-GCP-Kubernetes-Engine.php)
27. Docker : Jenkins Multibranch Pipeline with Jenkinsfile and Github (/DevOps/Docker/Docker-Jenkins-Multibranch-Pipeline-with-Jenkinsfile-and-Github.php)
28. Docker : Jenkins Master and Slave (/DevOps/Docker/Docker-Jenkins-Master-Slave-Agent-ssh.php)
29. Docker - ELK : ElasticSearch, Logstash, and Kibana (/DevOps/Docker/Docker_ELK_ElasticSearch_Logstash_Kibana.php)
30. Docker - ELK 7.6 : Elasticsearch on Centos 7 (/DevOps/Docker/Docker_ELK_7_6_Elasticsearch.php)
31. Docker - ELK 7.6 : Filebeat on Centos 7 (/DevOps/Docker/Docker_ELK_7_6_Filebeat.php)
32. Docker - ELK 7.6 : Logstash on Centos 7 (/DevOps/Docker/Docker_ELK_7_6_Logstash.php)
33. Docker - ELK 7.6 : Kibana on Centos 7 (/DevOps/Docker/Docker_ELK_7_6_Kibana.php)
34. Docker - ELK 7.6 : Elastic Stack with Docker Compose (/DevOps/Docker/Docker_ELK_7_6_Elastic_Stack_Docker_Compose.php)
35. Docker - Deploy Elastic Cloud on Kubernetes (ECK) via Elasticsearch operator on minikube (/DevOps/Docker/Docker_Kubernetes_Elastic_Cloud_on_Kubernetes_ECK_minikube.php)
36. Docker - Deploy Elastic Stack via Helm on minikube (/DevOps/Docker/Docker_Kubernetes_ElasticSearch_with_Helm_minikube.php)
37. Docker Compose - A gentle introduction with WordPress (/DevOps/Docker/Docker-Compose.php)
38. Docker Compose - MySQL (/DevOps/Docker/Docker-Compose-MySQL.php)
39. MEAN Stack app on Docker containers : micro services (/MEAN-Stack/MEAN-Stack-NodeJS-Angular-Docker.php)
40. MEAN Stack app on Docker containers : micro services via docker-compose (/MEAN-Stack/MEAN-Stack-NodeJS-Angular-Docker-Compose.php)
41. Docker Compose - Hashicorp's Vault and Consul Part A (install vault, unsealing, static secrets, and policies) (/DevOps/Docker/Docker-Vault-Consul.php)
42. Docker Compose - Hashicorp's Vault and Consul Part B (EaaS, dynamic secrets, leases, and revocation) (/DevOps/Docker/Docker-Vault-Consul-B.php)
43. Docker Compose - Hashicorp's Vault and Consul Part C (Consul) (/DevOps/Docker/Docker-Vault-Consul-C.php)
44. Docker Compose with two containers - Flask REST API service container and an Apache server container (/DevOps/Docker/Docker-Compose-FlaskREST-Service-Container-and-Apache-Container.php)
45. Docker compose : Nginx reverse proxy with multiple containers (/DevOps/Docker/Docker-Compose-Nginx-Reverse-Proxy-Multiple-Containers.php)
46. Docker & Kubernetes : Envoy - Getting started (/DevOps/Docker/Docker-Envoy-Getting-Started.php)
47. Docker & Kubernetes : Envoy - Front Proxy (/DevOps/Docker/Docker-Envoy-Front-Proxy.php)
48. Docker & Kubernetes : Ambassador - Envoy API Gateway on Kubernetes (/DevOps/Docker/Docker-Envoy-Ambassador-API-Gateway-for-Kubernetes.php)
49. Docker Packer (/DevOps/Docker/Docker-Packer.php)
50. Docker Cheat Sheet (/DevOps/Docker/Docker-Cheat-Sheet.php)
51. Docker Q & A #1 (/DevOps/Docker/Docker_Q_and_A.php)
52. Kubernetes Q & A - Part I (/DevOps/Docker/Docker_Kubernetes_Q_A_1.php)
53. Kubernetes Q & A - Part II (/DevOps/Docker/Docker_Kubernetes_Q_A_2.php)
54. Docker - Run a React app in a docker (/DevOps/Docker/Docker-React-App.php)
55. Docker - Run a React app in a docker II (snapshot app with nginx) (/DevOps/Docker/Docker-React-App-2-SnapShot.php)
56. Docker - NodeJS and MySQL app with React in a docker (/DevOps/Docker/Docker-React-Node-MySQL-App.php)
57. Docker - Step by Step NodeJS and MySQL app with React - I (/DevOps/Docker/Step-by-Step-React-Node-MySQL-App.php)
58. Installing LAMP via puppet on Docker (/DevOps/Docker/Installing-LAMP-with-puppet-on-Docker.php)
59. Docker install via Puppet (/DevOps/Docker/Docker_puppet.php)
60. Nginx Docker install via Ansible (/DevOps/Ansible/Ansible-Deploy-Nginx-to-Docker.php)
61. Apache Hadoop CDH 5.8 Install with QuickStarts Docker (/Hadoop/BigData_hadoop_CDH5.8_QuickStarts_Docker_Install.php)
62. Docker - Deploying Flask app to ECS (/DevOps/Docker/Docker-Flask-ALB-ECS.php)

Kubernetes I - Running Kubernetes Locally via Minikube (/DevOps/DevOps-Kubernetes-1-Running-Kubernetes-Locally-via-Minikube.php)

Kubernetes II - kops on AWS (/DevOps/DevOps-Kubernetes-II-kops-on-AWS.php)

Kubernetes III - kubeadm on AWS (/DevOps/DevOps-Kubernetes-III-Kubernetes-on-Linux-with-kubeadm.php)

AWS : EKS (Elastic Container Service for Kubernetes) (/DevOps/AWS/aws-EKS-Elastic-Container-Service-Kubernetes.php)

DEVOPS / SYS ADMIN Q & A

(1A) - Linux Commands (/DevOps/DevOps-Sys-Admin-Interview-Questions-Commands.php)

(1B) - Linux Commands (/DevOps/DevOps-Sys-Admin-Interview-Questions-Commands-2.php)

(2) - Networks (/DevOps/DevOps-Sys-Admin-Interview-Questions-Networks.php)

(2B) - Networks (/DevOps/DevOps-Sys-Admin-Interview-Questions-Networks-2.php)

(3) - Linux Systems (/DevOps/DevOps-Sys-Admin-Interview-Questions-Systems.php)

(4) - Scripting (Ruby/Shell) (/DevOps/DevOps-Sys-Admin-Interview-Questions-Scripting.php)

(5) - Configuration Management (/DevOps/DevOps-Sys-Admin-Interview-Questions-Configurations.php)

63. Docker Compose - Deploying WordPress to AWS (/DevOps/Docker/Docker-Compose-WordPress-AWS.php)
64. Docker - WordPress Deploy to ECS with Docker-Compose (ECS-CLI EC2 type) (/DevOps/Docker/Docker-ECS-CLI-Docker-Compose-Wordpress-EC2-Type.php)
65. Docker - WordPress Deploy to ECS with Docker-Compose (ECS-CLI Fargate type) (/DevOps/Docker/Docker-ECS-CLI-Docker-Compose-Wordpress-Fargate-Type.php)
66. Docker - ECS Fargate (/DevOps/Docker/Docker-ALB-ECS-Fargate.php)
67. Docker - AWS ECS service discovery with Flask and Redis (/DevOps/Docker/Docker-ALB-ECS-Fargate.php)
68. Docker & Kubernetes : minikube (/DevOps/Docker/Docker_Kubernetes_Minikube.php)
69. Docker & Kubernetes 2 : minikube Django with Postgres - persistent volume (/DevOps/Docker/Docker_Kubernetes_Minikube_2_Persistent_Volume_Django_with_Postgres.php)
70. Docker & Kubernetes 3 : minikube Django with Redis and Celery (/DevOps/Docker/Docker_Kubernetes_Minikube_3_Django_with_Redis_Celery.php)
71. Docker & Kubernetes 4 : Django with RDS via AWS Kops (/DevOps/Docker/Docker_Kubernetes_AWS_4_Django_with_RDS_Kops.php)
72. Docker & Kubernetes : Kops on AWS (/DevOps/DevOps-Kubernetes-II-kops-on-AWS.php)
73. Docker & Kubernetes : Ingress controller on AWS with Kops (/DevOps/Docker/Docker-Kubernetes-kops-on-AWS-Ingress.php)
74. Docker & Kubernetes : HashiCorp's Vault and Consul on minikube (/DevOps/Docker/Docker_Kubernetes_Vault_Consul_minikube.php)
75. Docker & Kubernetes : HashiCorp's Vault and Consul - Auto-unseal using Transit Secrets Engine (/DevOps/Docker/Docker_Kubernetes_Vault_Consul_minikube_Auto_Unseal_Vault_Transit.php)
76. Docker & Kubernetes : Persistent Volumes & Persistent Volumes Claims - hostPath and annotations (/DevOps/Docker/Docker_Kubernetes_PersistentVolumes_PersistentVolumeClaims.php)
77. Docker & Kubernetes : Persistent Volumes - Dynamic volume provisioning (/DevOps/Docker/Docker_Kubernetes_Persistent_Volumes_Dynamic_Volume_Provisioning.php)
78. Docker & Kubernetes : DaemonSet (/DevOps/Docker/Docker_Kubernetes_DaemonSet.php)
79. Docker & Kubernetes : Secrets (/DevOps/Docker/Docker_Kubernetes_Secrets.php)
80. Docker & Kubernetes : kubectl command (/DevOps/Docker/Docker_Kubernetes_kubectl_command.php)
81. Docker & Kubernetes : Assign a Kubernetes Pod to a particular node in a Kubernetes cluster (/DevOps/Docker/Docker_Kubernetes_assign_a_Pod_to_a_particular_node_in_a_Kubernetes_cluster.php)
82. Docker & Kubernetes : Configure a Pod to Use a ConfigMap (/DevOps/Docker/Docker_Kubernetes_Configure_a_Pod_to_Use_a_ConfigMap.php)
83. AWS : EKS (Elastic Container Service for Kubernetes) (/DevOps/AWS/aws-EKS-Elastic-Container-Service-Kubernetes.php)
84. Docker & Kubernetes : Run a React app in a minikube (/DevOps/Docker/Docker-Kubernetes-React-App.php)
85. Docker & Kubernetes : Minikube install on AWS EC2 (/DevOps/Docker/Docker-Kubernetes-Minikube-install-on-AWS-EC2.php)
86. Docker & Kubernetes : Cassandra with a StatefulSet (/DevOps/Docker/Docker_Kubernetes_StatefulSet_for_Cassandra.php)
87. Docker & Kubernetes : Terraform and AWS EKS (/DevOps/Docker/Docker_Kubernetes_Terraform_EKS.php)
88. Docker & Kubernetes : Pods and Service definitions (/DevOps/Docker/Docker_Kubernetes_Pods_Services_Yaml.php)
89. Docker & Kubernetes : Service IP and the Service Type (/DevOps/Docker/Docker_Kubernetes_Service_IP_and_Service_Type.php)
90. Docker & Kubernetes : Kubernetes DNS with Pods and Services (/DevOps/Docker/Docker_Kubernetes_DNS_with_Pods_Services.php)
91. Docker & Kubernetes : Headless service and discovering pods (/DevOps/Docker/Docker_Kubernetes_Headless_Service.php)
92. Docker & Kubernetes : Scaling and Updating application (/DevOps/Docker/Docker_Kubernetes_Scaling_and_Updating_Applications.php)

(6) - AWS VPC setup (public/private subnets with NAT) (/DevOps/DevOps-Sys-Admin-Interview-Questions-AWS-VPC-Setup.php)

(6B) - AWS VPC Peering (/DevOps/DevOps-Sys-Admin-Interview-Questions-AWS-VPC-Peering.php)

(7) - Web server (/DevOps/DevOps-Sys-Admin-Interview-Questions-Web-HTTP.php)

(8) - Database (/DevOps/DevOps-Sys-Admin-Interview-Questions-Database.php)

(9) - Linux System / Application Monitoring, Performance Tuning, Profiling Methods & Tools (/DevOps/DevOps-Sys-Admin-Interview-Questions-Linux-Monitoring-System-Application-Performance-Tuning-Tools.php)

(10) - Trouble Shooting: Load, Throughput, Response time and Leaks (/DevOps/DevOps-Sys-Admin-Interview-Questions-Trouble-Shooting-Slow-Application-Performance-BottleNecks-Leaks.php)

(11) - SSH key pairs, SSL Certificate, and SSL Handshake (/DevOps/DevOps-Sys-Admin-Interview-Questions-SSH-Connection-SSL-Certificates.php)

(12) - Why is the database slow? (/DevOps/DevOps-Sys-Admin-Interview-Questions-Why-is-database-slow.php)

(13) - Is my web site down? (/DevOps/DevOps-Sys-Admin-Interview-Questions-Is-Website-down.php)

(14) - Is my server down? (/DevOps/DevOps-Sys-Admin-Interview-Questions-Is-Server-down.php)

(15) - Why is the server sluggish? (/DevOps/DevOps-Sys-Admin-Interview-Questions-Why-is-theServer-slow.php)

93. Docker & Kubernetes : Horizontal pod autoscaler on minikubes (/DevOps/Docker/Docker-Kubernetes-Horizontal-Pod-Autoscaler.php)
94. Docker & Kubernetes : From a monolithic app to micro services on GCP Kubernetes (/DevOps/Docker/Docker-from-Monolithic-to-Micro-services-GCP-Kubernetes.php)
95. Docker & Kubernetes : Rolling updates (/DevOps/Docker/Docker-Kubernetes-Rolling-Updates.php)
96. Docker & Kubernetes : Deployments to GKE (Rolling update, Canary and Blue-green deployments) (/DevOps/Docker/Docker-Rolling-Update-Canary-Blue-Green-Deployments-to-GKE-Kubernetes.php)
97. Docker & Kubernetes : Slack Chat Bot with NodeJS on GCP Kubernetes (/DevOps/Docker/Docker-Slack-NodeJS-ChatBot-GCP-Kubernetes.php)
98. Docker & Kubernetes : Continuous Delivery with Jenkins Multibranch Pipeline for Dev, Canary, and Production Environments on GCP Kubernetes (/DevOps/Docker/Docker-Continuous-Delivery-with-Jenkins-Multibranch-Pipeline-for-Dev-Canary-Production-Environments-GCP-Kubernetes-Engine-Namespace.php)
99. Docker & Kubernetes : NodePort vs LoadBalancer vs Ingress (/DevOps/Docker/Docker_Kubernetes_NodePort_vs_LoadBalancer_vs_Ingress.php)
100. Docker & Kubernetes : MongoDB / MongoExpress on Minikube (/DevOps/Docker/Docker_Kubernetes_MongoDB_MongoExpress.php)
101. Docker & Kubernetes : Load Testing with Locust on GCP Kubernetes (/DevOps/Docker/Docker-Load-Testing-with-Locust-on-GCP-Kubernetes.php)
102. Docker & Kubernetes : MongoDB with StatefulSets on GCP Kubernetes Engine (/DevOps/Docker/Docker_MongoDB_with_StatefulSets_on_GCP_Kubernetes.php)
103. Docker & Kubernetes : Nginx Ingress Controller on Minikube (/DevOps/Docker/Docker_Kubernetes_Nginx_Ingress_Controller.php)
104. Docker & Kubernetes : Nginx Ingress Controller for Dashboard service on Minikube (/DevOps/Docker/Docker_Kubernetes_Nginx_Ingress_Controller_for_Dashboard_on_Minikube.php)
105. Docker & Kubernetes : Nginx Ingress Controller on GCP Kubernetes (/DevOps/Docker/Docker_Kubernetes_Nginx_Ingress_Controller_GCP_Kubernetes.php)
106. Docker & Kubernetes : Kubernetes Ingress with AWS ALB Ingress Controller in EKS (/DevOps/Docker/Docker-Kubernetes-ALB-Ingress-Controller-with-EKS.php)
107. Docker & Kubernetes : Setting up a private cluster on GCP Kubernetes (/DevOps/Docker/Docker-setting-up-private-cluster-on-GCP-Kubernetes.php)
108. Docker & Kubernetes : Kubernetes Namespaces (default, kube-public, kube-system) and switching namespaces (kubens) (/DevOps/Docker/Docker-Kubernetes-Namespaces.php)
109. Docker & Kubernetes : StatefulSets on minikube (/DevOps/Docker/Docker_Kubernetes_StatefulSet.php)
110. Docker & Kubernetes : RBAC (/DevOps/Docker/Docker-Kubernetes-RBAC.php)
111. Docker & Kubernetes Service Account, RBAC, and IAM (/DevOps/Docker/Docker-Kubernetes-Service-Account.php)
112. Docker & Kubernetes - Kubernetes Service Account, RBAC, IAM with EKS ALB, Part 1 (/DevOps/Docker/Docker-Kubernetes-ALB-on-EKS-1.php)
113. Docker & Kubernetes : Helm Chart (/DevOps/Docker/Docker_Helm_Chart.php)
114. Docker & Kubernetes : My first Helm deploy (/DevOps/Docker/Docker_Kubernetes_Helm_myFirst_Deploy.php)
115. Docker & Kubernetes : Readiness and Liveness Probes (/DevOps/Docker/Docker-Kubernetes-Readiness-Liveness-Probes.php)
116. Docker & Kubernetes : Helm chart repository with Github pages (/DevOps/Docker/Docker_Helm_Chart_Repository_with_Github_Pages.php)
117. Docker & Kubernetes : Deploying WordPress and MariaDB with Ingress to Minikube using Helm Chart (/DevOps/Docker/Docker_Helm_Chart_WordPress_MariaDB_Minikube_with_Ingress.php)
118. Docker & Kubernetes : Deploying WordPress and MariaDB to AWS using Helm 2 Chart (/DevOps/Docker/Docker_Helm_Chart_WordPress_MariaDB_AWS_with_KOPS.php)
119. Docker & Kubernetes : Deploying WordPress and MariaDB to AWS using Helm 3 Chart (/DevOps/Docker/Docker_Helm3_Chart_WordPress_MariaDB_AWS_with_KOPS.php)
120. Docker & Kubernetes : Helm Chart for Node/Express and MySQL with Ingress (/DevOps/Docker/Docker_Helm_Chart_Node_Expess_MySQL_Ingress.php)

(16A) - Serving multiple domains using Virtual Hosts - Apache (/DevOps/DevOps-Sys-Admin-Interview-Questions-Serving-Multiple-Domains-Using-Virtual-Hosts-Apache.php)

(16B) - Serving multiple domains using server block - Nginx (/DevOps/DevOps-Sys-Admin-Interview-Questions-Serving-Multiple-Domains-Using-Virtual-Hosts-Nginx.php)

(16C) - Reverse proxy servers and load balancers - Nginx (/DevOps/DevOps-Sys-Admin-Interview-Questions-Reverse-proxy-servers-and-load-balancing-Nginx.php)

(17) - Linux startup process (/DevOps/DevOps-Sys-Admin-Interview-Questions-Linux-Boot-Startup-Process.php)

(18) - phpMyAdmin with Nginx virtual host as a subdomain (/DevOps/DevOps_phpMyAdmin_N

(19) - How to SSH login without password? (/DevOps/DevOps-Sys-Admin-Interview-Questions-SSH-login-without-password.php)

(20) - Log Rotation (/DevOps/DevOps-Sys-Admin-Interview-Questions-Log-Rotation.php)

(21) - Monitoring Metrics (/DevOps/DevOps-Sys-Admin-Interview-Questions-Monitoring-Metrics.php)

(22) - lsof (/DevOps/DevOps-Sys-Admin-Interview-Questions-lsof.php)

(23) - Wireshark introduction (/DevOps/DevOps-WireShark-Tutorial-Introduction.php)

(24) - User account management (/DevOps/DevOps-Sys-Admin-Interview-Questions-Linux-User-Account-Management.php)

(25) - Domain Name System (DNS) (/DevOps/DevOps-Sys-Admin-Interview-Questions-DNS.php)

121. Docker & Kubernetes : Deploy Prometheus and Grafana using Helm and Prometheus Operator - Monitoring Kubernetes node resources out of the box (/DevOps/Docker/Docker_Kubernetes_Prometheus_Deploy_using_Helm_and_Prometheus_Operator.php)
122. Docker & Kubernetes : Istio (service mesh) sidecar proxy on GCP Kubernetes (/DevOps/Docker/Docker_Kubernetes_Istio_Sidecar_Proxy_on_GCP_Kubernetes_Engine.php)
123. Docker & Kubernetes : Istio on EKS (/DevOps/Docker/Docker-Kubernetes-EKS-with-ISTIO.php)
124. Docker & Kubernetes : Deploying .NET Core app to Kubernetes Engine and configuring its traffic managed by Istio (Part I) (/DevOps/Docker/Docker_Kubernetes_Deploying_ASP_Net_Core_App_with_ISTIO_Configured_on_Kubernetes_Engine.php)
125. Docker & Kubernetes : Deploying .NET Core app to Kubernetes Engine and configuring its traffic managed by Istio (Part II - Prometheus, Grafana, pin a service, split traffic, and inject faults) (/DevOps/Docker/Docker_Kubernetes_Deploying_ASP_Net_Core_App_with_ISTIO_Configured_on_Kubernetes_Engine_Part_2_More_On_ISTIO.p
126. Docker & Kubernetes : Helm Package Manager with MySQL on GCP Kubernetes Engine (/DevOps/Docker/Docker_Helm_Package_Manager_MySQL_GCP_Kubernetes.php)
127. Docker & Kubernetes : Deploying Memcached on Kubernetes Engine (/DevOps/Docker/Docker_Helm_Package_Manager_MySQL_GCP_Kubernetes.php)
128. Docker & Kubernetes : EKS Control Plane (API server) Metrics with Prometheus (/DevOps/Docker/Docker-Kubernetes-EKS-Control-Plane-API-Server-Metrics-with-Prometheus.php)
129. Docker & Kubernetes : Spinnaker on EKS with Halyard (/DevOps/Docker/Docker_Kubernetes_EKS_Spinnaker.php)
130. Docker & Kubernetes : Continuous Delivery Pipelines with Spinnaker and Kubernetes Engine (/DevOps/Docker/Docker_Kubernetes_Continuous_Delivery_Pipelines_with_Spinnaker_and_Kubernetes_Engine.php)
131. Docker & Kubernetes : Multi-node Local Kubernetes cluster : Kubeadm-dind (docker-in-docker) (/DevOps/Docker/Docker-Kubernetes-Multi-Node-Local-Clusters-dind.php)
132. Docker & Kubernetes : Multi-node Local Kubernetes cluster : Kubeadm-kind (k8s-in-docker) (/DevOps/Docker/Docker-Kubernetes-Multi-Node-Local-Clusters-kind.php)
133. Docker & Kubernetes : nodeSelector, nodeAffinity, taints/tolerations, pod affinity and anti-affinity - Assigning Pods to Nodes (/DevOps/Docker/Docker_Kubernetes_nodeSelector_nodeAffinity_taints_tolerations_podAffinity_antiAffinity.php)
134. Docker & Kubernetes : Jenkins-X on EKS (/DevOps/Docker/Docker_Kubernetes_Jenkins-X-EKS.php)
135. Docker & Kubernetes : ArgoCD App of Apps with Heml on Kubernetes (/DevOps/Docker/Docker_Kubernetes_ArgoCD_with_Helm_on_Kubernetes_App_of_Apps.php)
136. Docker & Kubernetes : ArgoCD on Kubernetes cluster (/DevOps/Docker/Docker_Kubernetes_ArgoCD_on_Kubernetes_cluster.php)

(26) - NGINX SSL/TLS, Caching, and Session (/DevOps/DevOps-Sys-Admin-Interview-Questions-NGINX-SSL-TLS-Caching-Session.php)

(27) - Troubleshooting 5xx server errors (/DevOps/DevOps-Sys-Admin-Interview-Questions-5xx-http-server-errors.php)

(28) - Linux Systemd: journalctl (/DevOps/DevOps-Sys-Admin-Interview-Questions-Linux-Systemd-Journalctl.php)

(29) - Linux Systemd: FirewallD (/DevOps/DevOps-Sys-Admin-Interview-Questions-Linux-Systemd-Firewalld.php)

(30) - Linux: SELinux (/DevOps/DevOps-Sys-Admin-Interview-Questions-Linux-SELinux.php)

(31) - Linux: Samba (/DevOps/DevOps-Sys-Admin-Interview-Questions-Linux-Samba.php)

(0) - Linux Sys Admin's Day to Day tasks (/DevOps/DevOps-Sys-Admin-Interview-Questions-Day-To-Day-Tasks.php)

Jenkins

Install (/DevOps/Jenkins/Jenkins_Install.p

Configuration - Manage Jenkins - security setup (/DevOps/Jenkins/Jenkins_Configu

Adding job and build (/DevOps/Jenkins/Jenkins_Adding_

Scheduling jobs (/DevOps/Jenkins/Jenkins_Schedul

Managing_plugins (/DevOps/Jenkins/Jenkins_Manage_

Git/GitHub plugins, SSH keys

configuration, and Fork/Clone
(/DevOps/Jenkins/Jenkins_Git_Githu

JDK & Maven setup
(/DevOps/Jenkins/Jenkins_Maven_S

Build configuration for GitHub Java application with Maven
(/DevOps/Jenkins/Jenkins_GitHub_

Build Action for GitHub Java application with Maven - Console Output, Updating Maven
(/DevOps/Jenkins/Jenkins_GitHub_

Commit to changes to GitHub & new test results - Build Failure
(/DevOps/Jenkins/Jenkins_GitHub_

Commit to changes to GitHub & new test results - Successful Build
(/DevOps/Jenkins/Jenkins_GitHub_

Adding code coverage and metrics
(/DevOps/Jenkins/Jenkins_Adding_

Jenkins on EC2 - creating an EC2 account, ssh to EC2, and install Apache server
(/DevOps/Jenkins/Jenkins_on_EC2_

Jenkins on EC2 - setting up Jenkins account, plugins, and Configure System (JAVA_HOME, MAVEN_HOME, notification email)
(/DevOps/Jenkins/Jenkins_on_EC2_

Jenkins on EC2 - Creating a Maven project
(/DevOps/Jenkins/Jenkins_on_EC2_

Jenkins on EC2 - Configuring GitHub Hook and Notification service to Jenkins server for any changes to the repository
(/DevOps/Jenkins/Jenkins_on_EC2_

Jenkins on EC2 - Line Coverage with JaCoCo plugin
(/DevOps/Jenkins/Jenkins_on_EC2_

Setting up Master and Slave nodes
(/DevOps/Jenkins/Jenkins_on_EC2_

Jenkins Build Pipeline & Dependency Graph Plugins
(/DevOps/Jenkins/Jenkins_Build_Pi

Jenkins Build Flow Plugin

(/DevOps/Jenkins/Jenkins_Build_Flo

Pipeline Jenkinsfile with Classic / Blue Ocean (/DevOps/Jenkins/Jenkins_Pipeline_

Jenkins Setting up Slave nodes on AWS (/DevOps/Jenkins/Jenkins_Slave_No

Jenkins Q & A (/DevOps/Jenkins/Jenkins_Q_and_A

Puppet

Puppet with Amazon AWS I - Puppet accounts (/DevOps/Puppet/puppet_amazon

Puppet with Amazon AWS II (ssh & puppetmaster/puppet install) (/DevOps/Puppet/puppet_amazon

Puppet with Amazon AWS III - Puppet running Hello World (/DevOps/Puppet/puppet_amazon

Puppet Code Basics - Terminology (/DevOps/Puppet/puppet_basics_c

Puppet with Amazon AWS on CentOS 7 (I) - Master setup on EC2 (/DevOps/Puppet/puppet_amazon

Puppet with Amazon AWS on CentOS 7 (II) - Configuring a Puppet Master Server with Passenger and Apache (/DevOps/Puppet/puppet_amazon

Puppet master /agent ubuntu 14.04 install on EC2 nodes (/DevOps/Puppet/puppet_install_o

Puppet master post install tasks - master's names and certificates setup, (/DevOps/Puppet/puppet_master_

Puppet agent post install tasks - configure agent, hostnames, and sign request (/DevOps/Puppet/puppet_agent_p

EC2 Puppet master/agent basic tasks - main manifest with a file resource/module and immediate execution on an agent node (/DevOps/Puppet/puppet_basic_ta

Setting up puppet master and agent with simple scripts on EC2 / remote install from desktop (/DevOps/Puppet/puppet_setting_

EC2 Puppet - Install lamp with a manifest ('puppet apply') (/DevOps/Puppet/puppet_amazon

EC2 Puppet - Install lamp with a module (/DevOps/Puppet/puppet_amazon

Puppet variable scope (/DevOps/Puppet/puppet_variable

Puppet packages, services, and files (/DevOps/Puppet/puppet_package

Puppet packages, services, and files II with nginx (/DevOps/Puppet/puppet_package
Puppet templates (/DevOps/Puppet/puppet_templat

Puppet creating and managing user accounts with SSH access (/DevOps/Puppet/puppet_creating

Puppet Locking user accounts & deploying sudoers file (/DevOps/Puppet/puppet_locking_

Puppet exec resource (/DevOps/Puppet/puppet_exec_re

Puppet classes and modules (/DevOps/Puppet/puppet_classes_

Puppet Forge modules (/DevOps/Puppet/Puppet_Forge_M

Puppet Express (/DevOps/Puppet/puppet_express

Puppet Express 2 (/DevOps/Puppet/puppet_express

Puppet 4 : Changes (/DevOps/Puppet/puppet4_change

Puppet --configprint (/DevOps/Puppet/puppet_configpr

Puppet with Docker (/DevOps/Docker/Docker_puppet.

Puppet 6.0.2 install on Ubuntu 18.04 (/DevOps/Puppet/Puppet6-Install-on-Ubuntu18.0.4.php)

Chef

What is Chef? (/DevOps/Chef/Chef_What_is_Chef

Chef install on Ubuntu 14.04 - Local Workstation via omnibus installer (/DevOps/Chef/Install_Chef_on_Ub

Setting up Hosted Chef server (/DevOps/Chef/Chef_Setting_up_H

VirtualBox via Vagrant with Chef client provision (/DevOps/Chef/Chef_Virtual_Machi

Creating and using cookbooks on a VirtualBox node (/DevOps/Chef/Chef_Creating_and

Chef server install on Ubuntu 14.04 (/DevOps/Chef/Chef_Server_install

Chef workstation setup on EC2 Ubuntu 14.04 (/DevOps/Chef/Chef_Setting_Up_W

Chef Client Node - Knife Bootstrapping a node on EC2 ubuntu 14.04 (/DevOps/Chef/Chef_Client_Node_

Elasticsearch search engine, Logstash, and Kibana

Elasticsearch, search engine (/Hadoop/ELK/ELK_Elastic_Search_

Logstash with Elasticsearch
(/Hadoop/ELK/ELK_ElasticSearch_L

Logstash, Elasticsearch, and Kibana 4
(/Hadoop/ELK/ELK_ElasticSearch_L

Elasticsearch with Redis broker and Logstash Shipper and Indexer
(/Hadoop/ELK/ELK_Logstash_Shipp

Samples of ELK architecture
(/Hadoop/ELK/ELK_Architecture_Sa

Elasticsearch indexing performance
(/Hadoop/ELK/ELK_Elastic_Search_

Vagrant

VirtualBox & Vagrant install on Ubuntu 14.04
(/DevOps/Vagrant/Vagrant_Virtual

Creating a VirtualBox using Vagrant
(/DevOps/Vagrant/Creating_Virtual

Provisioning
(/DevOps/Vagrant/Vagrant_Provisi

Networking - Port Forwarding
(/DevOps/Vagrant/Vagrant_Networ

Vagrant Share
(/DevOps/Vagrant/Vagrant_Share.

Vagrant Rebuild & Teardown
(/DevOps/Vagrant/Vagrant_Rebuild

Vagrant & Ansible
(/DevOps/Vagrant/Vagrant_Ansible

Big Data & Hadoop Tutorials

Hadoop 2.6 - Installing on

Ubuntu 14.04 (Single-Node Cluster) (/Hadoop/BigData_hadoop_Install_

Hadoop 2.6.5 - Installing on Ubuntu 16.04 (Single-Node Cluster) (/Hadoop/BigData_hadoop_Install_

Hadoop - Running MapReduce Job (/Hadoop/BigData_hadoop_Runnin

Hadoop - Ecosystem (/Hadoop/BigData_hadoop_Ecosys

CDH5.3 Install on four EC2 instances (1 Name node and 3 Datanodes) using Cloudera Manager 5 (/Hadoop/BigData_hadoop_CDH5_

CDH5 APIs (/Hadoop/BigData_hadoop_CDH5_

QuickStart VMs for CDH 5.3 (/Hadoop/BigData_hadoop_QuickS VMs for CDH 5.3.x.php)

QuickStart VMs for CDH 5.3 II - Testing with wordcount (/Hadoop/BigData_hadoop_QuickS VMs for CDH 5.3.x_II.php)

QuickStart VMs for CDH 5.3 II - Hive DB query (/Hadoop/BigData_hadoop_QuickS

Scheduled start and stop CDH services (/Hadoop/BigData_hadoop_CDH5_

CDH 5.8 Install with QuickStarts Docker (/Hadoop/BigData_hadoop_CDH5.8

Zookeeper & Kafka Install (/Hadoop/BigData_hadoop_Zookee

Zookeeper & Kafka - single node single broker (/Hadoop/BigData_hadoop_Zookee

Zookeeper & Kafka - Single node and multiple brokers (/Hadoop/BigData_hadoop_Zookee

OLTP vs OLAP (/Hadoop/BigData_hadoop_OLTP_v

Apache Hadoop Tutorial I with CDH - Overview (/Hadoop/BigData_hadoop_Cloude

Apache Hadoop Tutorial II with CDH - MapReduce Word Count (/Hadoop/BigData_hadoop_Tutoria

Apache Hadoop Tutorial III with CDH - MapReduce Word Count 2 (/Hadoop/BigData_hadoop_Tutoria

Apache Hadoop (CDH 5) Hive Introduction (/Hadoop/BigData_hadoop_CDH5_

CDH5 - Hive Upgrade to 1.3 to from 1.2 (/Hadoop/BigData_hadoop_CDH5_

Apache Hive 2.1.0 install on Ubuntu 16.04 (/Hadoop/BigData_hadoop_Hive_In

Apache HBase in Pseudo-Distributed mode (/Hadoop/BigData_hadoop_HBase_

Creating HBase table with HBase shell and HUE (/Hadoop/BigData_hadoop_HBase_

Apache Hadoop : Hue 3.11 install on Ubuntu 16.04 (/Hadoop/BigData_hadoop_Hue_In

Creating HBase table with Java API (/Hadoop/BigData_hadoop_HBase_

HBase - Map, Persistent, Sparse, Sorted, Distributed and Multidimensional (/Hadoop/BigData_hadoop_HBase_

Flume with CDH5: a single-node Flume deployment (telnet example) (/Hadoop/BigData_hadoop_CDH5_

Apache Hadoop (CDH 5) Flume with VirtualBox : syslog example via NettyAvroRpcClient (/Hadoop/BigData_hadoop_CDH5_

List of Apache Hadoop hdfs commands (/Hadoop/BigData_Hadoop_fs_com

Apache Hadoop : Creating Wordcount Java Project with Eclipse Part 1 (/Hadoop/BigData_hadoop_Creatin

Apache Hadoop : Creating Wordcount Java Project with

Eclipse Part 2
(/Hadoop/BigData_hadoop_Creatin

Apache Hadoop : Creating Card Java Project with Eclipse using Cloudera VM UnoExample for CDH5 - local run
(/Hadoop/BigData_hadoop_Creatin

Apache Hadoop : Creating Wordcount Maven Project with Eclipse
(/Hadoop/BigData_hadoop_Creatin

Wordcount MapReduce with Oozie workflow with Hue browser - CDH 5.3 Hadoop cluster using VirtualBox and QuickStart VM
(/Hadoop/BigData_hadoop_QuickS

Spark 1.2 using VirtualBox and QuickStart VM - wordcount
(/Hadoop/BigData_hadoop_Apache

Spark Programming Model : Resilient Distributed Dataset (RDD) with CDH
(/Hadoop/BigData_hadoop_Apache

Apache Spark 2.0.2 with PySpark (Spark Python API) Shell
(/Hadoop/BigData_hadoop_Apache

Apache Spark 2.0.2 tutorial with PySpark : RDD
(/Hadoop/BigData_hadoop_Apache

Apache Spark 2.0.0 tutorial with PySpark : Analyzing Neuroimaging Data with Thunder
(/Hadoop/BigData_hadoop_Apache

Apache Spark Streaming with Kafka and Cassandra
(/Hadoop/BigData_hadoop_Apache

Apache Spark 1.2 with PySpark (Spark Python API) Wordcount using CDH5
(/Hadoop/BigData_hadoop_Apache

Apache Spark 1.2 Streaming
(/Hadoop/BigData_hadoop_Apache

Apache Drill with ZooKeeper install on Ubuntu 16.04 - Embedded & Distributed
(/Drill/Drill_Tutorial_Install_on_ubu

Apache Drill - Query File System, JSON, and Parquet

(/Drill/Drill_Tutorial_Query_File_Sys

Apache Drill - HBase query (/Drill/Drill_Tutorial_Query_HBase.

Apache Drill - Hive query (/Drill/Drill_Tutorial_Query_Hive.ph

Apache Drill - MongoDB query (/Drill/Drill_Tutorial_Query_Mongo

Redis In-Memory Database

Redis vs Memcached (/DevOps/Redis/Redis_vs_Memcac

Redis 3.0.1 Install (/DevOps/Redis/Redis_Install.php)

Setting up multiple server instances on a Linux host (/DevOps/Redis/Redis_Setting_up_

Redis with Python (/python/python_redis_with_pytho

ELK : Elasticsearch with Redis broker and Logstash Shipper and Indexer (/Hadoop/ELK/ELK_Logstash_Shipp

GCP (Google Cloud Platform)

GCP: Creating an Instance (/DevOps/GCP/gcp_Creating_an_In

GCP: gcloud compute command-line tool (/DevOps/GCP/gcp_gcloud_compu

GCP: Deploying Containers (/DevOps/GCP/gcp_Deploying_Con

GCP: Kubernetes Quickstart

(/DevOps/GCP/gcp_Kubernetes_Qu

GCP: Deploying a containerized web application via Kubernetes (/DevOps/GCP/gcp_Deploying_Con

GCP: Django Deploy via Kubernetes I (local) (/DevOps/GCP/gcp_Django_Deploy

GCP: Django Deploy via Kubernetes II (GKE) (/DevOps/GCP/gcp_Django_Deploy

AWS (Amazon Web Services)

AWS : EKS (Elastic Container Service for Kubernetes) (/DevOps/AWS/aws-EKS-Elastic-Container-Service-Kubernetes.php)

AWS : Creating a snapshot (cloning an image) (/DevOps/AWS/aws_snapshot_ami

AWS : Attaching Amazon EBS volume to an instance (/DevOps/AWS/aws_attaching_Ama

AWS : Adding swap space to an attached volume via mkswap and swapon (/DevOps/AWS/aws_adding_swap_

AWS : Creating an EC2 instance and attaching Amazon EBS volume to the instance using Python boto module with User data (/DevOps/AWS/aws_creating_an_in

AWS : Creating an instance to a new region by copying an AMI (/DevOps/AWS/Launching-Instance-to-a-New-Region-from-an-AMI.php)

AWS : S3 (Simple Storage Service) 1 (/DevOps/AWS/aws_S3_Simple_Sto

AWS : S3 (Simple Storage Service) 2 - Creating and Deleting a

Bucket (/DevOps/AWS/aws_S3_Simple_Sto

AWS : S3 (Simple Storage Service) 3 - Bucket Versioning (/DevOps/AWS/aws_S3_Simple_Sto

AWS : S3 (Simple Storage Service) 4 - Uploading a large file (/DevOps/AWS/aws_S3_uploading_

AWS : S3 (Simple Storage Service) 5 - Uploading folders/files recursively (/DevOps/AWS/aws-uploading-recursive-folder-file.php)

AWS : S3 (Simple Storage Service) 6 - Bucket Policy for File/Folder View/Download (/DevOps/AWS/aws-S3-bucket-policy-view-download.php)

AWS : S3 (Simple Storage Service) 7 - How to Copy or Move Objects from one region to another (/DevOps/AWS/aws-s3-Configure-Cross-Region-Replication-Source-and-Destination-Buckets-Owned-by-the-Same-AWS-Account-How-to-Copy-or-Move-Objects-from-one-Region-to-another.php)

AWS : S3 (Simple Storage Service) 8 - Archiving S3 Data to Glacier (/DevOps/AWS/aws-S3-Simple-Storage-Service-Archiving-Amazon-S3-Data-to-Amazon-Glacier.php)

AWS : Creating a CloudFront distribution with an Amazon S3 origin (/DevOps/AWS/aws_Creating_Clou

AWS : Creating VPC with CloudFormation (/DevOps/AWS/aws-creating-VPC-with-CloudFormation.php)

WAF (Web Application Firewall) with preconfigured CloudFormation template and Web ACL for CloudFront distribution (/DevOps/AWS/aws-WAF-Web-Application-Firewall.php)

AWS : CloudWatch & Logs with Lambda Function / S3 (/DevOps/AWS/aws-CloudWatch-logs-Lambda-S3.php)

AWS : Lambda Serverless Computing with EC2, CloudWatch Alarm, SNS (/DevOps/AWS/aws-Lambda-Serverless-S3-CloudWatch-Alarm-SNS.php)

AWS : Lambda and SNS - cross account (/DevOps/AWS/aws-Lambda-SNS.php)

AWS : CLI (Command Line Interface) (/DevOps/AWS/aws-CLI-Command-Line-Interface.php)

AWS : CLI (ECS with ALB & autoscaling) (/DevOps/AWS/aws-Amazon-ECS-ALB-Autoscaling-CLI.php)

AWS : ECS with cloudformation and json task definition (/DevOps/AWS/aws-ECS-with-CloudFormation-and-json-task-definition.php)

AWS : AWS Application Load Balancer (ALB) and ECS with Flask app (/DevOps/AWS/aws-ELB-ALB-Application-Load-Balancer-ECS.php)

AWS : Load Balancing with HAProxy (High Availability Proxy) (/DevOps/AWS/aws-Load-Balancing-with-HAProxy-High-Availability-Proxy.php)

AWS : VirtualBox on EC2 (/DevOps/AWS/aws_VirtualBox_On

AWS : NTP setup on EC2 (/DevOps/AWS/aws_NTP.php)

AWS: jq with AWS (/DevOps/AWS/aws-jq.php)

AWS : AWS & OpenSSL : Creating / Installing a Server SSL Certificate (/DevOps/AWS/aws-HTTPS-OpenSSL-Certificate.php)

AWS : OpenVPN Access Server 2 Install (/DevOps/AWS/aws-OpenVPN-Access-Server-Install.php)

AWS : VPC (Virtual Private Cloud) 1 - netmask, subnets, default gateway, and CIDR (/DevOps/AWS/aws-VPC-Virtual-

Private-Cloud-1-netmast-subnet-default-gateway-CIDR.php)

AWS : VPC (Virtual Private Cloud) 2 - VPC Wizard (/DevOps/AWS/aws-VPC-Virtual-Private-Cloud-2-VPC-Wizard.php)

AWS : VPC (Virtual Private Cloud) 3 - VPC Wizard with NAT (/DevOps/AWS/aws-VPC-Virtual-Private-Cloud-3-VPC-Wizard-with-NAT.php)

AWS : DevOps / Sys Admin Q & A (VI) - AWS VPC setup (public/private subnets with NAT) (/DevOps/DevOps-Sys-Admin-Interview-Questions-AWS-VPC-Setup.php)

AWS : OpenVPN Protocols : PPTP, L2TP/IPsec, and OpenVPN (/DevOps/AWS/aws-VPN-Protocols-OpenVPN-IPsec-L2TP-PPTP.php)

AWS : Autoscaling group (ASG) (/DevOps/AWS/aws-Autoscaling-Group-ASG.php)

AWS : Setting up Autoscaling Alarms and Notifications via CLI and Cloudformation (/DevOps/AWS/aws-AutoScaling-LaunchConfiguration-Notification-LoadBalancer-CloudWatch-Alert-High-Low-CPU-utilization-CLI-Cloudformation.php)

AWS : Adding a SSH User Account on Linux Instance (/DevOps/AWS/aws-adding-a-ssh-user-account-on-linux-instance.php)

AWS : Windows Servers - Remote Desktop Connections using RDP (/DevOps/AWS/aws_Windows_Serv

AWS : Scheduled stopping and starting an instance - python & cron (/DevOps/AWS/aws_stopping_start

AWS : Detecting stopped instance and sending an alert email using Mandrill smtp (/DevOps/AWS/aws_detecting_stop

AWS : Elastic Beanstalk with NodeJS (/DevOps/AWS/aws-

Elastic-Beanstalk-with-NodeJS.php)

AWS : Elastic Beanstalk Inplace/Rolling Blue/Green Deploy (/DevOps/AWS/aws-Beanstalk-InPlaceRolling-BlueGreen-Deploy.php)

AWS : Identity and Access Management (IAM) Roles for Amazon EC2 (/DevOps/AWS/aws-IAM-Roles.php)

AWS : Identity and Access Management (IAM) Policies, sts AssumeRole, and delegate access across AWS accounts (/DevOps/AWS/aws-IAM-Policies.php)

AWS : Identity and Access Management (IAM) sts assume role via aws cli2 (/DevOps/AWS/aws-sts-assume-roles-cli.php)

AWS : Creating IAM Roles and associating them with EC2 Instances in CloudFormation (/DevOps/AWS/aws-creating-IAM-Roles-and-associating-them-with-EC2-Instances-in-CloudFormation.php)

AWS Identity and Access Management (IAM) Roles, SSO(Single Sign On), SAML(Security Assertion Markup Language), IdP(identity provider), STS(Security Token Service), and ADFS(Active Directory Federation Services) (/DevOps/AWS/aws-IAM-Roles-SSO-Single-Sign-On-SAML-Security-Assertion-Markup-Language-ADFS-Active-Directory-Federation-Services.php)

AWS : Amazon Route 53 (/DevOps/AWS/aws-Route53-DNS.php)

AWS : Amazon Route 53 - DNS (Domain Name Server) setup (/DevOps/AWS/aws-Route53-DNS-Domain-Name-Server-Setup.php)

AWS : Amazon Route 53 - subdomain setup and virtual host on Nginx (/DevOps/AWS/aws-Route53-

DNS-Subdomain.php)

AWS Amazon Route 53 : Private Hosted Zone (/DevOps/AWS/aws-Route53-DNS-Private-Hosted-Zone.php)

AWS : SNS (Simple Notification Service) example with ELB and CloudWatch (/DevOps/AWS/aws-Amazon-SNS-Simple-Notification-Service-example-with-ELB.php)

AWS : Lambda with AWS CloudTrail (/DevOps/AWS/aws-Lambda-with-AWS-CloudTrail.php)

AWS : SQS (Simple Queue Service) with NodeJS and AWS SDK (/DevOps/AWS/aws-Amazon-SQS-Simple-Queue-Service-with-NodeJS-AWS-SDK.php)

AWS : Redshift data warehouse (/DevOps/AWS/aws_Redshift_data_

AWS : CloudFormation - templates, change sets, and CLI (/DevOps/AWS/aws-CloudFormation-Templates.php)

AWS : CloudFormation Bootstrap UserData/Metadata (/DevOps/AWS/aws-CloudFormation-Bootstrap-UserData.php)

AWS : CloudFormation - Creating an ASG with rolling update (/DevOps/AWS/aws-CloudFormation-Autoscaling-Group-ASG-Application-Load-Balancer-ALB-with-Update-Policy-Rolling-Updates.php)

AWS : Cloudformation Cross-stack reference (/DevOps/AWS/aws-Cloudformation-CrossStck-Reference.php)

AWS : OpsWorks (/DevOps/AWS/aws-OpsWorks.php)

AWS : Network Load Balancer (NLB) with Autoscaling group (ASG) (/DevOps/AWS/aws-Autoscaling-Group-Network-Load-Balancer.php)

AWS CodeDeploy : Deploy an Application from GitHub (/DevOps/AWS/aws-CodeDeploy-Deploy-an-Application-from-GitHub.php)

AWS EC2 Container Service (ECS) (/DevOps/AWS/aws-Amazon-EC2-Container-Service-ECS.php)

AWS EC2 Container Service (ECS) II (/DevOps/AWS/aws-Amazon-EC2-Container-Service-ECS-2.php)

AWS Hello World Lambda Function (/DevOps/AWS/aws-Hello-World-Lambda-Function.php)

AWS Lambda Function Q & A (/DevOps/AWS/aws-Lambda-Function-Q-and-A.php)

AWS Node.js Lambda Function & API Gateway (/DevOps/AWS/aws-Lambda-Nodejs-API-Gateway.php)

AWS API Gateway endpoint invoking Lambda function (/DevOps/AWS/aws-API-Gateway.php)

AWS API Gateway invoking Lambda function with Terraform (/DevOps/AWS/aws-API-Gateway-Lambda-Terraform.php)

AWS API Gateway invoking Lambda function with Terraform - Lambda Container (/DevOps/AWS/aws-API-Gateway-Lambda-Terraform-with-ECR-Container.php)

Amazon Kinesis Streams (/DevOps/AWS/aws-Amazon-Kinesis-Streams.php)

Kinesis Data Firehose with Lambda and ElasticSearch (/DevOps/AWS/aws-Kinesis-Data-Firehose-TransformData-with-Lambda-send-Data-to-ElasticSearch.php)

Amazon DynamoDB (/DevOps/AWS/aws-Amazon-DynamoDB.php)

Amazon DynamoDB with Lambda and CloudWatch

(/DevOps/AWS/aws-Amazon-DynamoDB-Lambda-CloudWatch.php)

Loading DynamoDB stream to AWS Elasticsearch service with Lambda (/DevOps/AWS/aws-Amazon-Loading-DynamoDB-Stream-to-ElasticSearch-with-Lambda.php)

Amazon ML (Machine Learning) (/DevOps/AWS/aws-ML-Machine-Learning.php)

Simple Systems Manager (SSM) (/DevOps/AWS/aws-SSM.php)

AWS : RDS Connecting to a DB Instance Running the SQL Server Database Engine (/DevOps/AWS/aws_Connecting_to

AWS : RDS Importing and Exporting SQL Server Data (/DevOps/AWS/aws_RDS_Importin

AWS : RDS PostgreSQL & pgAdmin III (/DevOps/AWS/aws_RDS_PostgreS

AWS : RDS PostgreSQL 2 - Creating/Deleting a Table (/DevOps/AWS/aws_RDS_PostgreS

AWS : MySQL Replication : Master-slave (/DevOps/AWS/aws-MySQL-Replication-Master-Slave.php)

AWS : MySQL backup & restore (/DevOps/AWS/aws-MySQL-Backup-mysqldump-Restore.php)

AWS RDS : Cross-Region Read Replicas for MySQL and Snapshots for PostgreSQL (/DevOps/AWS/aws-RDS-Cross-Region-Read-Replicas-for-MySQL-Snapshot-for-PostgreSQLs.php)

AWS : Restoring Postgres on EC2 instance from S3 backup (/DevOps/AWS/aws_S3_EC2_PostG

AWS : Q & A (/DevOps/AWS/aws-Q-A.php)

AWS : Security (/DevOps/AWS/aws-Security.php)

AWS : Scaling-Up
(/DevOps/AWS/aws-Scaling-Up.php)

AWS : Networking
(/DevOps/AWS/aws-Networking.php)

Powershell 4 Tutorial

Powersehll : Introduction
(http://www.bogotobogo.com/Pow

Powersehll : Help System
(http://www.bogotobogo.com/Pow

Powersehll : Running commands
(http://www.bogotobogo.com/Pow

Powersehll : Providers
(http://www.bogotobogo.com/Pow

Powersehll : Pipeline
(http://www.bogotobogo.com/Pow

Powersehll : Objects
(http://www.bogotobogo.com/Pow

Powershell : Remote Control
(http://www.bogotobogo.com/Pow

Windows Management Instrumentation (WMI)
(http://www.bogotobogo.com/Pow

How to Enable Multiple RDP Sessions in Windows 2012 Server
(http://www.bogotobogo.com/Pow

How to install and configure FTP server on IIS 8 in Windows 2012 Server
(http://www.bogotobogo.com/Pow

How to Run Exe as a Service on Windows 2012 Server
(http://www.bogotobogo.com/Pow

SQL Inner, Left, Right, and Outer Joins
(http://www.bogotobogo.com/Pow

Git/GitHub Tutorial

(/cplusplus/Git/GitHub_Account_SS

Uploading to GitHub
(/cplusplus/Git/GitHub_Uploading.

GUI
(/cplusplus/Git/GitHub_GUI.php)

Branching & Merging
(/cplusplus/Git/Git_Branching_Mer

Merging conflicts
(/cplusplus/Git/Git_Branching_Mer

GIT on Ubuntu and OS X - Focused on Branching
(/cplusplus/Git/Git_Ubuntu.php)

Setting up a remote repository / pushing local project and cloning the remote repo
(/cplusplus/Git/Git_Setting_Up_Rer

Fork vs Clone, Origin vs Upstream
(/cplusplus/Git/GitHub_Fork_Clone

Git/GitHub Terminologies
(/cplusplus/Git/Git_Terminologies.

Git/GitHub via SourceTree I : Commit & Push
(/cplusplus/Git/Git_GitHub_Source

Git/GitHub via SourceTree II : Branching & Merging
(/cplusplus/Git/Git_GitHub_Source

Git/GitHub via SourceTree III : Git Work Flow
(/cplusplus/Git/Git_GitHub_Source

Git/GitHub via SourceTree IV : Git Reset
(/cplusplus/Git/Git_GitHub_Source

Git wiki - quick command reference
(/cplusplus/Git/Git_GitHub_quick_c

Subversion

Subversion Install On Ubuntu 14.04
(/cplusplus/Subversion/Subversion

Subversion creating and

accessing I
(/cplusplus/Subversion/Subversion

Subversion creating and
accessing II
(/cplusplus/Subversion/Subversion

CONTACT

BogoToBogo
contactus@bogotobogo.com (mailto:contactus@bogotobogo.com)

FOLLOW BOGOTOBOGO

(https://www.facebook.com/KHongSanFrancisco) (https://twitter.com/KHongTwit)

ABOUT US (/ABOUT_US.PHP)

contactus@bogotobogo.com (mailto:contactus@bogotobogo.com)

Golden Gate Ave, San Francisco, CA 94115

Golden Gate Ave, San Francisco, CA 94115