Deploying Your First Docker Container

~Neha Singh

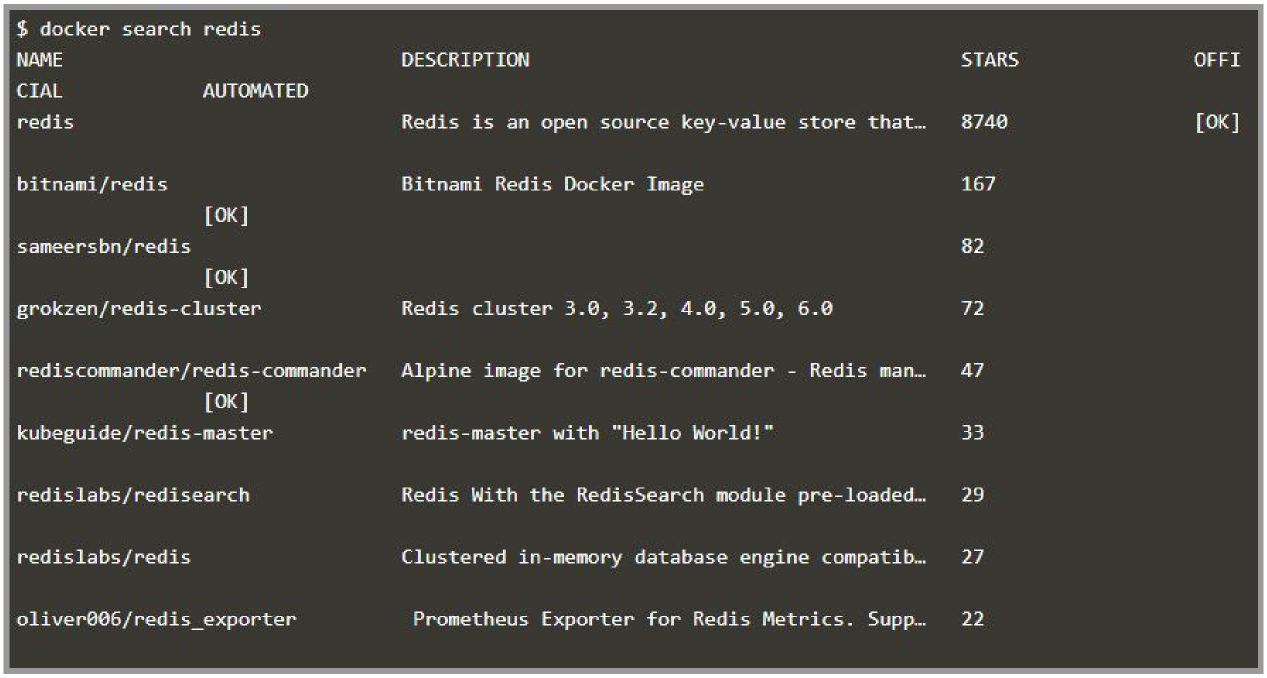
What Is Docker?

Docker describes themselves as "an open platform for developers and sysadmins to build, ship, and run distributed applications".

Docker allows you to run containers. A container is a sandboxed process running an application and its dependencies on the host operating system. The application inside the container considers itself to be the only process running on the machine while the machine can run multiple containers independently.

Step 1 - Running A Container

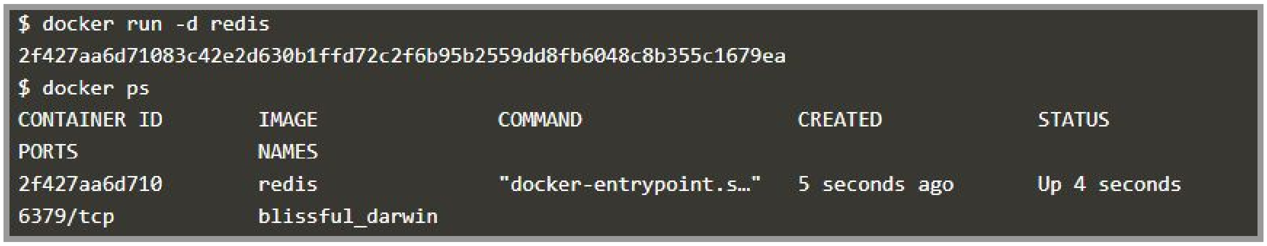
existing images at registry.hub.docker.com/ or by using the command `docker search <name>`. For example, to find an image for *Redis*, you would use `docker search redis`.



```
$ docker search redis
NAME                             DESCRIPTION                                      STARS               OFFI
CIAL            AUTOMATED
redis                            Redis is an open source key-value store that…  8740                [OK]

bitnami/redis                    Bitnami Redis Docker Image                       167
                [OK]
sameersbn/redis                                                                   82
                [OK]
grokzen/redis-cluster            Redis cluster 3.0, 3.2, 4.0, 5.0, 6.0            72

rediscommander/redis-commander   Alpine image for redis-commander - Redis man…  47
                [OK]
kubeguide/redis-master           redis-master with "Hello World!"                 33

redislabs/redisearch             Redis With the RedisSearch module pre-loaded…   29

redislabs/redis                  Clustered in-memory database engine compatib…    27

oliver006/redis_exporter          Prometheus Exporter for Redis Metrics. Supp…   22
```

Docker will run a command in the foreground. To run in the background, the option *-d* needs to be specified.

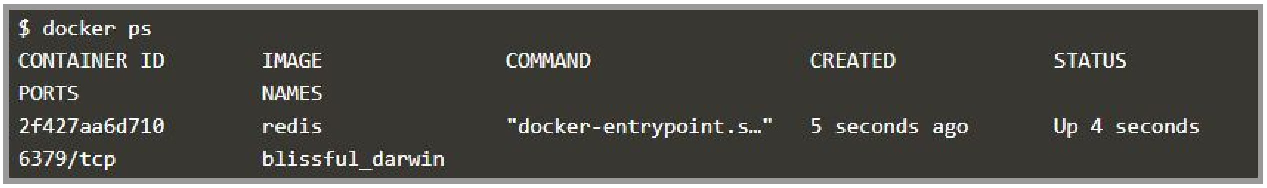
`docker run -d redis`



```
$ docker run -d redis
2f427aa6d71083c42e2d630b1ffd72c2f6b95b2559dd8fb6048c8b355c1679ea
$ docker ps
CONTAINER ID        IMAGE               COMMAND                  CREATED             STATUS
PORTS               NAMES
2f427aa6d710        redis               "docker-entrypoint.s…"   5 seconds ago       Up 4 seconds
6379/tcp            blissful_darwin
```

Step 2 - Finding Running Containers

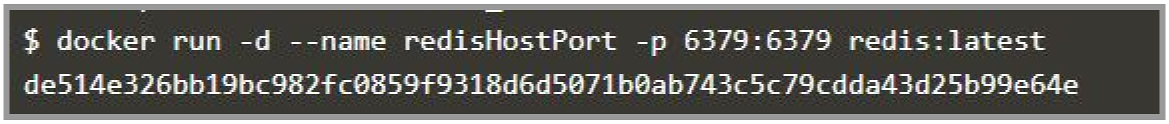
The launched container is running in the background, the `docker ps` command lists all running containers, the image used to start the container and uptime.



```
$ docker ps
CONTAINER ID        IMAGE               COMMAND                  CREATED             STATUS
PORTS               NAMES
2f427aa6d710        redis               "docker-entrypoint.s…"   5 seconds ago       Up 4 seconds
6379/tcp            blissful_darwin
```

Step 3 - Accessing Redis

it's useful to define a name when starting the container, running *Redis* in the background, with a name of *redisHostPort* on port *6379* is using the following command `docker run -d --name redisHostPort -p 6379:6379 redis:latest`

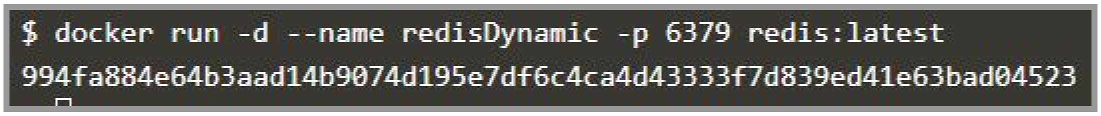


```
$ docker run -d --name redisHostPort -p 6379:6379 redis:latest
de514e326bb19bc982fc0859f9318d6d5071b0ab743c5c79cdda43d25b99e64e
```

Step 4 - Accessing Redis

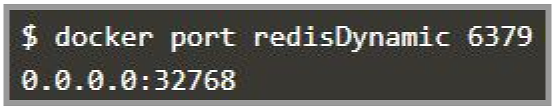
Run processes on a fixed port are that you can only run one instance. To run multiple *Redis* instances and configure the application depending on which port Redis is running on.

```
docker run -d --name redisDynamic -p 6379 redis:latest
```



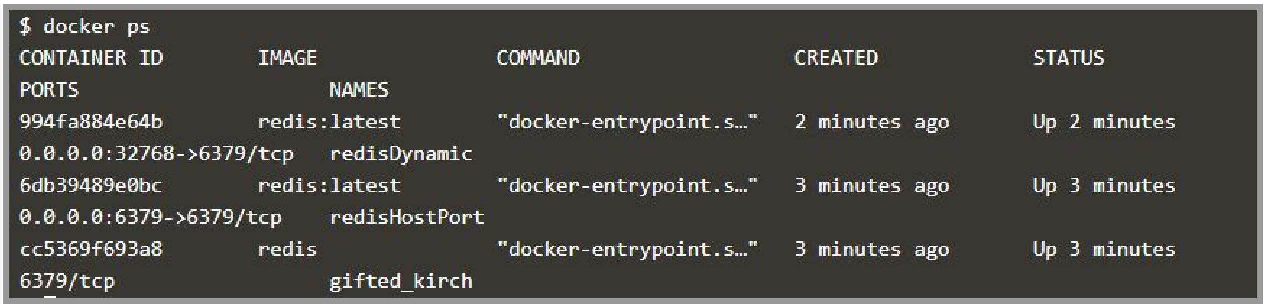
```
$ docker run -d --name redisDynamic -p 6379 redis:latest
994fa884e64b3aad14b9074d195e7df6c4ca4d43333f7d839ed41e63bad04523
```

To know which port has been assigned, this can be discovered via `docker port redisDynamic 6379`



```
$ docker port redisDynamic 6379
0.0.0.0:32768
```

To find the listing the containers displays the port mapping information, `docker ps`



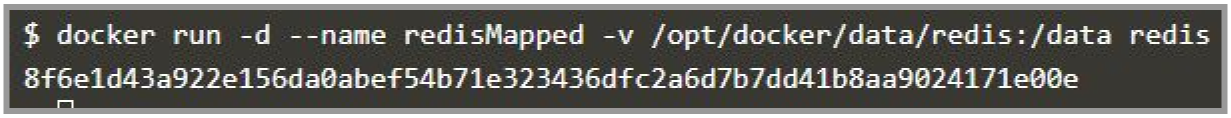
```
$ docker ps
CONTAINER ID        IMAGE               COMMAND                  CREATED             STATUS
PORTS                     NAMES
994fa884e64b        redis:latest        "docker-entrypoint.s…"   2 minutes ago       Up 2 minutes
0.0.0.0:32768->6379/tcp   redisDynamic
6db39489e0bc        redis:latest        "docker-entrypoint.s…"   3 minutes ago       Up 3 minutes
0.0.0.0:6379->6379/tcp    redisHostPort
cc5369f693a8        redis               "docker-entrypoint.s…"   3 minutes ago       Up 3 minutes
6379/tcp                  gifted_kirch
```

Step 5 - Persisting Data

To investigate that the official Redis image stores logs and data into a /data directory.

Any data which needs to be saved on the Docker Host, and not inside containers, should be stored in *opt/docker/data/redis*.

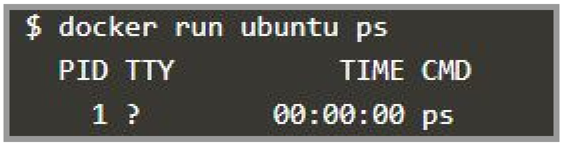
The complete command to solve the task is `docker run -d --name redisMapped -v /opt/docker/data/redis:/data redis`



```
$ docker run -d --name redisMapped -v /opt/docker/data/redis:/data redis
8f6e1d43a922e156da0abef54b71e323436dfc2a6d7b7dd41b8aa9024171e00e
```

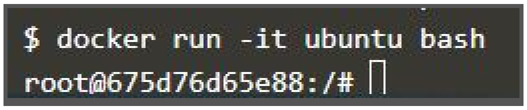
Step 6 - Running A Container In The Foreground

The command `docker run ubuntu ps` launches an Ubuntu container and executes the command *ps* to view all the processes running in a container.



```
$ docker run ubuntu ps
  PID TTY          TIME CMD
    1 ?        00:00:00 ps
```

Using `docker run -it ubuntu bash` allows getting access to a bash shell inside of a container.



```
$ docker run -it ubuntu bash
root@675d76d65e88:/# 
```