

Clean up Exited Docker Containers With Docker PS, AWK, and Grep

As we start to mess around more with Docker, one of the things that I have to do regularly is to purge out the containers that are exited and unused. I will put the disclaimer here that this is a way to remove ALL exited containers. In other words, any container that is not running in detached mode will be destroyed. This is very handy for development systems where you test out lots of containers, and we will learn one of the most useful Docker commands out there - docker ps!

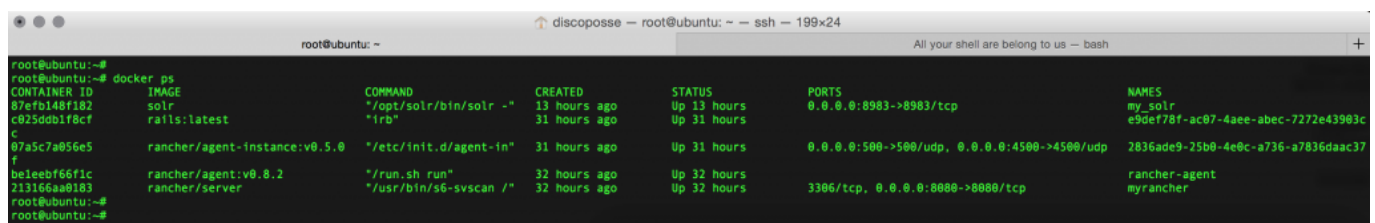
Docker PS and Pattern Matching the Awk-ward Way

I won't dive into the goodness of awk, but I do recommend that you get familiar with it as it is a powerful command line tool to do pattern matching and parsing of data. First, we will take a look at how to view containers that are running.

This is done with the docker ps command, which simply lists containers, and you can pass some parameters to it. docker ps is one of the most useful commands you will use in docker.

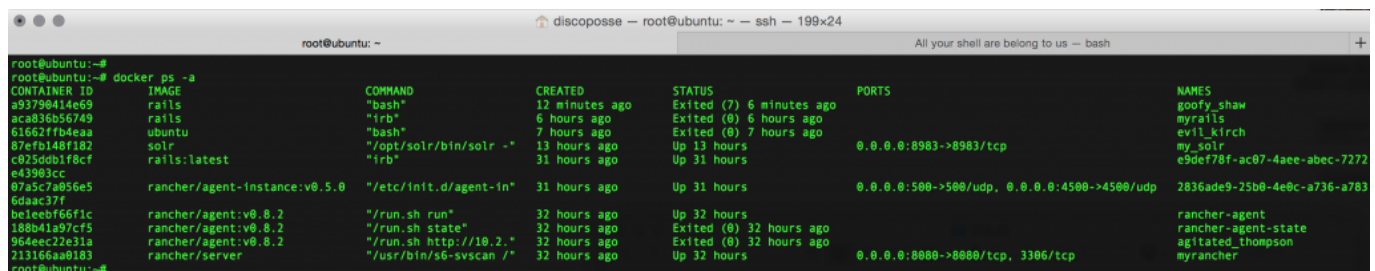
Running the Docker PS Command

By default, docker ps shows you the active containers:



```
root@ubuntu:~# docker ps
CONTAINER ID   IMAGE          COMMAND                  CREATED        STATUS        PORTS                               NAMES
87feb148f182   solr           "/opt/solr/bin/solr -"  13 hours ago  Up 13 hours  0.0.0.0:8983->8983/tcp              my_solr
e925ddb1f8cf   rails:latest   "irb"                   31 hours ago  Up 31 hours  3306/tcp, 0.0.0.0:8080->8080/tcp    myrancher
07a5c7a056e5   rancher/agent-instance:v0.5.0 "/etc/init.d/agent-in"  31 hours ago  Up 31 hours  0.0.0.0:500->500/udp, 0.0.0.0:4500->4500/udp  2836ade9-25b0-4e8c-a736-a7836daac37f
be1eebf66f1c   rancher/agent:v0.8.2 "/run.sh run"          32 hours ago  Up 32 hours  3306/tcp, 0.0.0.0:8080->8080/tcp    rancher-agent
213166aa8183   rancher/server "/usr/bin/s6-svscan /"  32 hours ago  Up 32 hours  3306/tcp, 0.0.0.0:8080->8080/tcp    myrancher
```

Since we know we have been mucking around with other containers that are not running, we need to run docker ps -a to view all containers:



```
root@ubuntu:~# docker ps -a
CONTAINER ID   IMAGE          COMMAND                  CREATED        STATUS        PORTS                               NAMES
a93798414e69   rails         "bash"                 12 minutes ago  Exited (7) 6 minutes ago
aca836b56749   rails         "irb"                 6 hours ago    Exited (0) 6 hours ago
616627fb4eaa   ubuntu       "bash"                 7 hours ago    Exited (0) 7 hours ago
87feb148f182   solr           "/opt/solr/bin/solr -"  13 hours ago  Up 13 hours  0.0.0.0:8983->8983/tcp              my_solr
e925ddb1f8cf   rails:latest   "irb"                 31 hours ago  Up 31 hours  3306/tcp, 0.0.0.0:8080->8080/tcp    myrancher
e43983cc       rancher/agent-instance:v0.5.0 "/etc/init.d/agent-in"  31 hours ago  Up 31 hours  0.0.0.0:500->500/udp, 0.0.0.0:4500->4500/udp  2836ade9-25b0-4e8c-a736-a7836daac37f
6daac37f       rancher/agent:v0.8.2 "/run.sh run"          32 hours ago  Up 32 hours  3306/tcp, 0.0.0.0:8080->8080/tcp    rancher-agent
188b41a9cf5    rancher/agent:v0.8.2 "/run.sh state"       32 hours ago  Exited (0) 32 hours ago
964eccc22c31a  rancher/agent:v0.8.2 "/run.sh http://10.2." 32 hours ago  Exited (0) 32 hours ago
213166aa8183   rancher/server "/usr/bin/s6-svscan /"  32 hours ago  Up 32 hours  3306/tcp, 0.0.0.0:8080->8080/tcp, 3306/tcp    agitated_thompson
myrancher
```

Adding Options to the Docker PS Command

We can also add a filter option to the command by using docker ps -a -f STATUS=exited to give us the list of containers which have been exited:

```
root@ubuntu: ~
root@ubuntu:~# docker ps -a -f STATUS=exited
CONTAINER ID    IMAGE    COMMAND    CREATED    STATUS    PORTS    NAMES
a93790414e69    rails    "bash"     14 minutes ago    Exited (7) 8 minutes ago
aca836b56749    rails    "irb"      6 hours ago    Exited (0) 6 hours ago
61662ffb4eaa    ubuntu  "bash"     7 hours ago    Exited (0) 7 hours ago
188b41a97cf5    rancher/agent:v0.8.2    "/run.sh state"    32 hours ago    Exited (0) 32 hours ago
964eec22e31a    rancher/agent:v0.8.2    "/run.sh http://10.2."    32 hours ago    Exited (0) 32 hours ago
root@ubuntu:~#
```

Combining awk and docker ps

Let's refine the results using awk to pull only what we need to pass to our `docker rm` command to remove the containers. In this case, we want the container ID which is the first column value. We will use the `docker ps -a -f STATUS=exited | awk '{print $1}'` command to do that:

```
root@ubuntu: ~
root@ubuntu:~# docker ps -a -f STATUS=exited | awk '{print $1}'
CONTAINER
a93790414e69
aca836b56749
61662ffb4eaa
188b41a97cf5
964eec22e31a
root@ubuntu:~#
```

You can use a lower case for the filter parameter, but for some reason I like to use it the way it displays in the table. So, now that we have a command that can give us a list of the container IDs that we want to remove, there is only one step. Let's remove all of these using the `docker rm $(docker ps -a -f STATUS=exited | awk '{print $1}')` command.

This command does a docker remove for all values pulled from the `docker ps` command:

```
root@ubuntu: ~
root@ubuntu:~# docker rm $(docker ps -a -f STATUS=exited | awk '{print $1}')
Error response from daemon: no such id: CONTAINER
a93790414e69
aca836b56749
61662ffb4eaa
188b41a97cf5
964eec22e31a
Error: failed to remove containers: [CONTAINER]
root@ubuntu:~#
```

You can see that there will be an error thrown because we pulled the entire first column including the header which said "CONTAINER", so the `docker rm` command tripped on this one.

We can confirm the results by running our `docker ps -a` and you will see that now we only have our running containers in the list:

```
root@ubuntu: ~
root@ubuntu:~#
root@ubuntu:~# docker ps -a -f STATUS=exited
CONTAINER ID    IMAGE    COMMAND    CREATED    STATUS    PORTS    NAMES
root@ubuntu:~#
```

You could also do further limiting by adding a grep command. Perhaps we only want to remove exited containers from the Ubuntu image. That can be done using `docker rm $(docker ps -a -f STATUS=exited | grep -i ubuntu | awk '{print $1}')` as an example:

```
root@ubuntu:~#
root@ubuntu:~# docker ps -a -f STATUS=exited
CONTAINER ID    IMAGE    COMMAND    CREATED    STATUS    PORTS    NAMES
f497355915b3    rails    "bash"    15 seconds ago    Exited (0) 12 seconds ago    tender_hamilton
69ef453a7858    ubuntu  "bash"    25 seconds ago    Exited (0) 23 seconds ago    ecstatic_hamilton
bff673b75fd2    ubuntu  "bash"    29 seconds ago    Exited (0) 28 seconds ago    sick_mclean
root@ubuntu:~#
root@ubuntu:~# docker rm $(docker ps -a -f STATUS=exited | grep -i ubuntu | awk '{print $1}')
69ef453a7858
bff673b75fd2
root@ubuntu:~# docker ps -a -f STATUS=exited
CONTAINER ID    IMAGE    COMMAND    CREATED    STATUS    PORTS    NAMES
f497355915b3    rails    "bash"    43 seconds ago    Exited (0) 40 seconds ago    tender_hamilton
root@ubuntu:~#
```

Between awk and grep you can do some pretty powerful stuff with a little practice. Hopefully this is a helpful tip for you.

More About docker ps

`docker ps` is one of the most useful and commonly used commands in Docker, since it lists containers. Remember, if you want to list ALL containers, including running and stopped, use the `docker ps -a` command.

Be sure to check out the Docker Documentation for the [official and complete](#) `docker ps` reference.