如果经常使用 docker，你会发现 docker 占用的资源膨胀很快，其中最明显也最容易被察觉的应该是对磁盘空间的占用。本文将介绍如何快速的清理 docker 占用的系统资源，具体点说就是删除那些无用的 镜像、容器、网络和数据卷。

**查看 docker 占用的资源**

在进行资源清理之前我们有必要搞清楚 docker 都占用了哪些系统的资源。这需要综合使用不同的命令来完成。

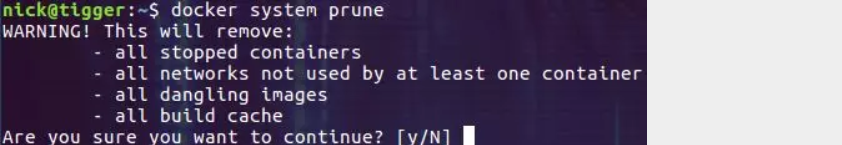
docker container ls：默认只列出正在运行的容器，-a 选项会列出包括停止的所有容器。  
docker image ls：列出镜像信息，-a 选项会列出 intermediate 镜像(就是其它镜像依赖的层)。  
docker volume ls：列出数据卷。  
docker network ls：列出 network。  
docker info：显示系统级别的信息，比如容器和镜像的数量等。

通过这些命令查看 docker 使用的资源情况后，相信你已经决定要清理 docker 占用的一些资源了！让我们先从那些未被使用的资源开始。

**只删除那些未被使用的资源**

Docker 提供了方便的 docker system prune 命令来删除那些已停止的容器、dangling 镜像、未被容器引用的 network 和构建过程中的 cache：

$ docker system prune

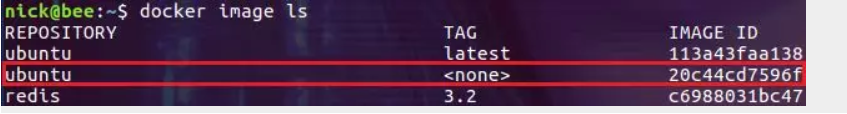


安全起见，这个命令默认不会删除那些未被任何容器引用的数据卷，如果需要同时删除这些数据卷，你需要显式的指定 --volumns 参数。比如你可能想要执行下面的命令：

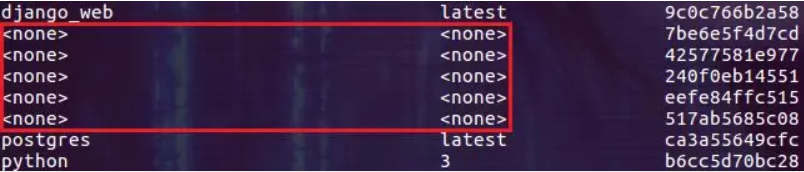
$ docker system prune --all --force --volumns

这次不仅会删除数据卷，而且连确认的过程都没有了！注意，使用 --all 参数后会删除所有未被引用的镜像而不仅仅是 dangling 镜像。

这里有必要解释一下何为 dangling images，其实可以简单的理解为未被任何镜像引用的镜像。比如在你重新构建了镜像后，那些之前构建的且不再被引用的镜像层就变成了 dangling images：



在本地的镜像更新之后，就会出现类似图中红框内的 <none> 镜像。这表示旧的镜像已经不再被引用了，此时它们就变成了 dangling images。如果使用 -a 参数，你还会发现另外一种类型的 <none> 镜像，它们的 repository 和 tag 列都表现为 <none>：



这些镜像被称为 intermediate 镜像(就是其它镜像依赖的层)。

我们还可在不同在子命令下执行 prune，这样删除的就是某类资源：  
docker container prune # 删除所有退出状态的容器  
docker volume prune # 删除未被使用的数据卷  
docker image prune # 删除 dangling 或所有未被使用的镜像

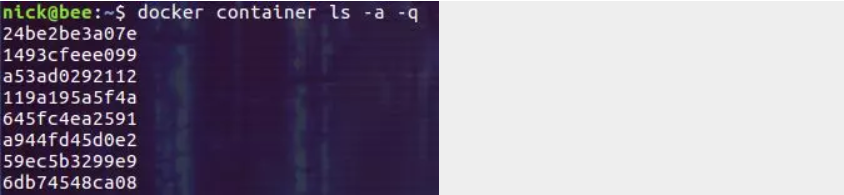
# 让 docker 回到安装时的状态

这里的 "安装时的状态" 指资源占用情况而不是 docker 的相关配置。这也是一种比较常见的用例，比如笔者就需要在一个干净的 docker 环境中自动化的还原出某天的一个生产环境(使用生产环境的备份数据)用于 bug 调查。让我们一起来看看都需要做些什么？

回想我们前面介绍的 docker system prune --all --force --volumns 命令，如果在执行这个命令前系统中所有的容器都已停止，那么这个命令就会移除所有的资源！好，现在让我们想办法停掉系统中的所有容器。

docker container stop 命令可以停止一个或多个容器，我们只需要把系统中所有在运行的容器罗列出来就可以了。由于 docker 并不介意我们再次停止一个已经停止了的容器，干脆简单粗暴点，直接列出所有的容器(包括已经停止的)！

$ docker container ls -a -q



-a 显示所有的容器，-q 只显示数字形式的容器 ID。  
然后把这里命令执行的结果作为 docker container stop 命令的参数：

$ docker container stop $(docker container ls -a -q)

完整的恢复 docker 环境的命令如下：

$ docker container stop $(docker container ls -a -q) && docker system prune --all --force --volumns

和前面的 prune 命令类似，也可以完全删除某一类资源：

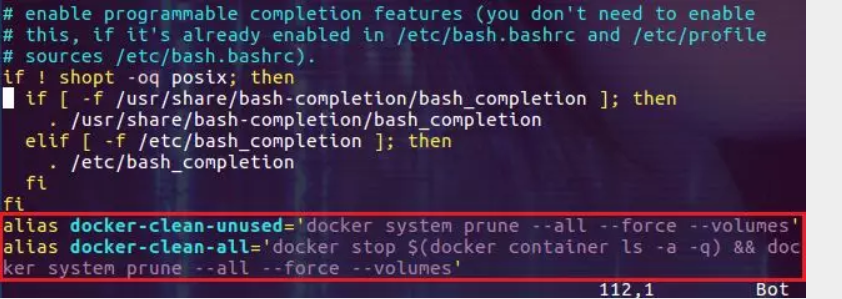
删除容器：docker container rm $(docker container ls -a -q)  
删除镜像：docker image rm $(docker image ls -a -q)  
删除数据卷：docker volume rm $(docker volume ls -q)  
删除 network：docker network rm $(docker network ls -q)

# 创建 shell 别名

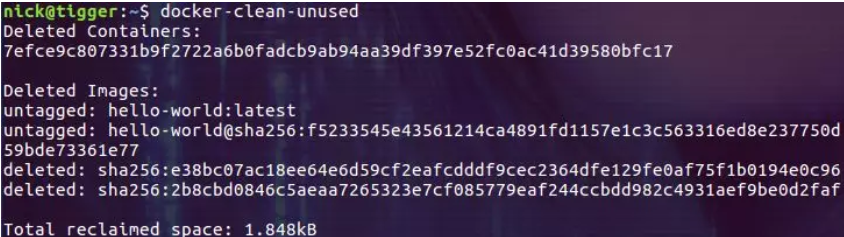
上面的命令可以完成任务但是却很繁琐，我们可以通过 shell 的别名功能来简化这些命令的执行。

alias docker-clean-unused='docker system prune --all --force --volumes'alias docker-clean-all='docker stop $(docker container ls -a -q) && docker system prune --all --force --volumes'

把上面的命令写入到用户的 ~/.bashrc 文件中就可以了！



执行一次清理任务：



# 总结

经常清理系统资源不仅能够让系统运行的更流畅，也利于我们把精力集中在相关的重点资源上面。所以建议大家能够使用相关的资源清理命令，让 docker 保持清爽和高效。

参考：  
Clean out your Docker images, containers and volumes with single commands

https://hackernoon.com/clean-out-your-docker-images-containers-and-volumes-with-single-commands-b8e38253c271

原文：

https://www.cnblogs.com/sparkdev/p/9177283.html