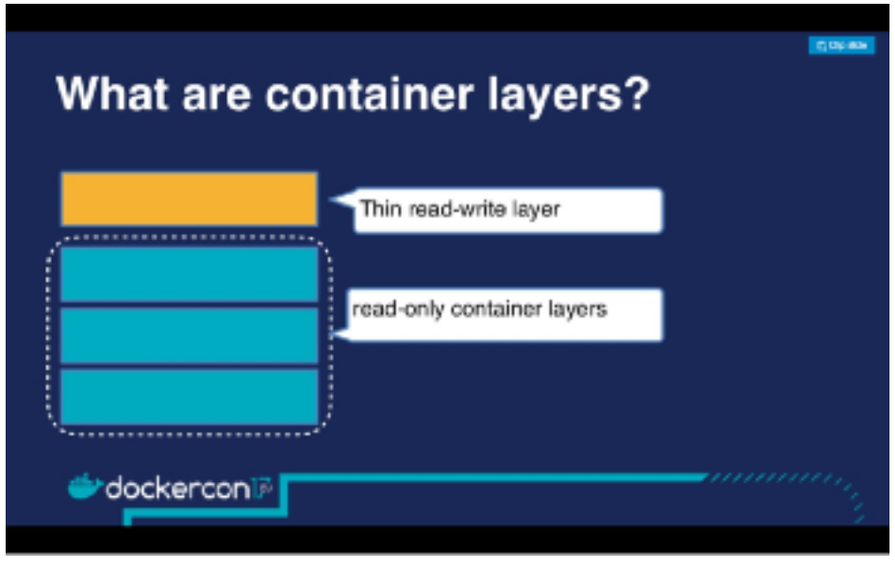
docker image management

What is it?

- a snapshot of a file system
- layered, so generally maintains a list of FS changes kind of like Git - at the expense of space
- only top level is R/W
- images can be tagged and stored locally in the image index, and remotely in a registry
- provides root access by default



When managed individually, images are generally created by this process:

- “**docker pull <user>/<image>:<tag>**” - starting with a OS base image (eg. library/busybox:latest)
- “**docker run -d**” or* “**docker create**” to apply modifications
- “**docker diff**” to inspect changes
- “**docker commit**” to save the changes to a new image name
- “**docker push**” to push the changes to a repo

* the difference between run (-d is detached) and create is that “create” never starts the container

manual image modification

```
            start a registry ->  docker run -d -p 5000:5000 registry:2
  pull latest from docker hub ->  docker pull busybox:latest
  make a new tag for our repo ->  docker tag busybox:latest localhost:5000/busybox
push the copied image to repo ->  docker push localhost:5000/busybox
    remove the busy box image ->  docker rmi busybox:latest
   test pulling the new image ->  docker pull localhost:5000/busybox:latest
    run our image with a name ->  docker run --name mylinux -it localhost:5000/busybox:latest
                  modify it ->  touch helloworld.txt
```

```
   list of running containers ->  2> docker ps
    differences to base image ->  2> docker diff mylinux
commit changes to a new image ->  2> docker commit mylinux localhost:5000/mylinux
        push them to the repo ->  2> docker push localhost:5000/mylinux
     remove the old container ->  2> docker rm -f mylinux
         remove the new image ->  2> docker rmi localhost:5000/busybox:latest
   test pulling the new image ->  2> docker pull localhost:5000/mylinux:latest
    run our image with a name ->  2> docker run --name mylinux2 -it localhost:5000/mylinux
check helloworld.txt is there ->  2> ls
```