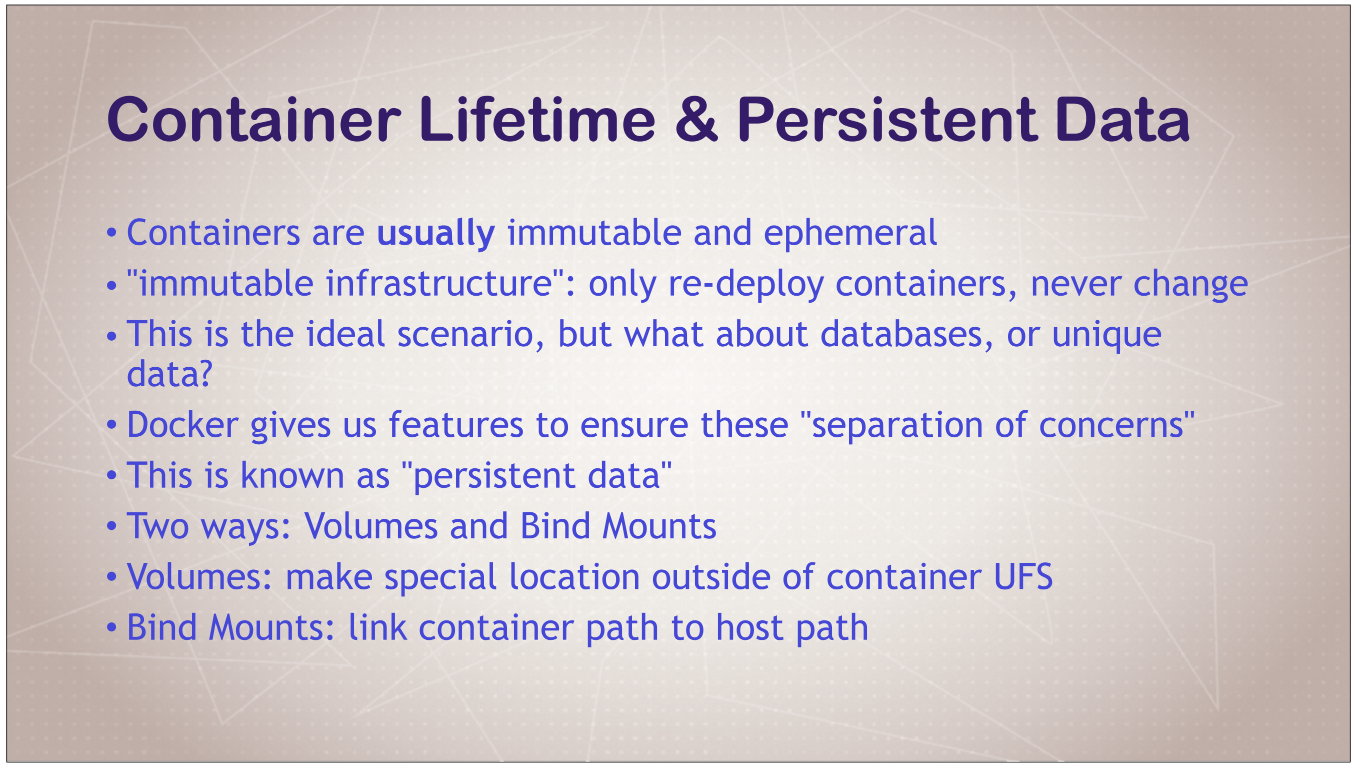
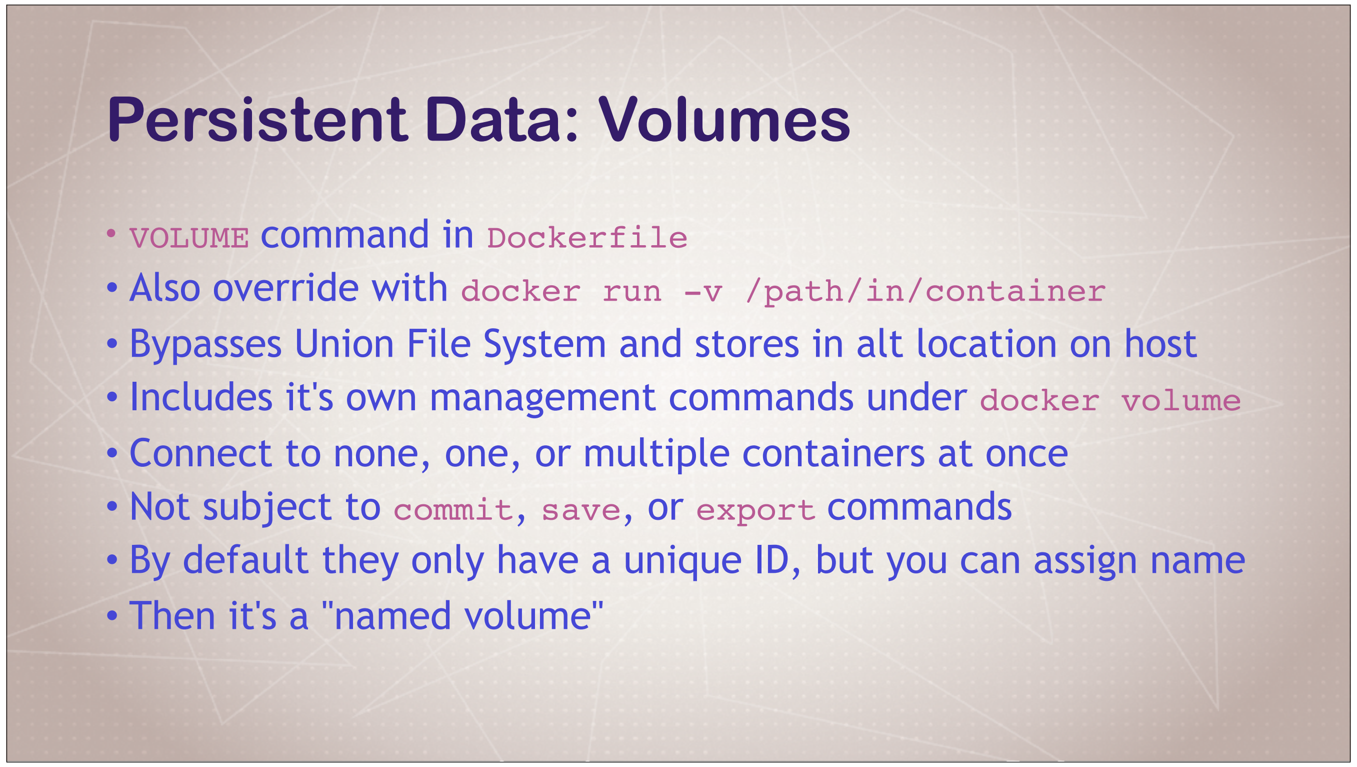
Section Overview

- Defining the problem of persistent data
- Key concepts with containers: immutable, ephemeral
- Learning and using Data Volumes
- Learning and using Bind Mounts
- Assignments



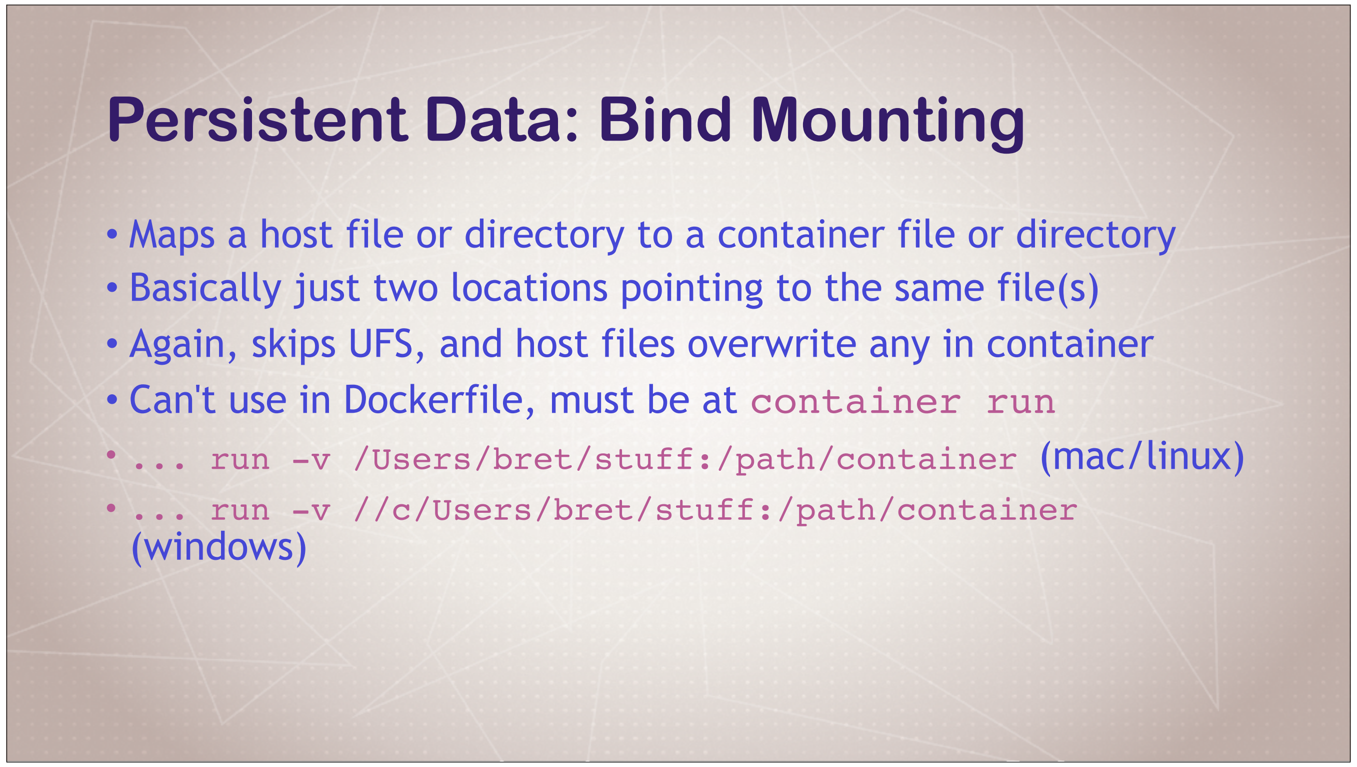
Container Lifetime & Persistent Data

- Containers are **usually** immutable and ephemeral
- "immutable infrastructure": only re-deploy containers, never change
- This is the ideal scenario, but what about databases, or unique data?
- Docker gives us features to ensure these "separation of concerns"
- This is known as "persistent data"
- Two ways: Volumes and Bind Mounts
- Volumes: make special location outside of container UFS
- Bind Mounts: link container path to host path



Persistent Data: Volumes

- `VOLUME` command in `Dockerfile`
- Also override with `docker run -v /path/in/container`
- Bypasses Union File System and stores in alt location on host
- Includes it's own management commands under `docker volume`
- Connect to none, one, or multiple containers at once
- Not subject to `commit`, `save`, or `export` commands
- By default they only have a unique ID, but you can assign name
- Then it's a "named volume"



Persistent Data: Bind Mounting

- Maps a host file or directory to a container file or directory
- Basically just two locations pointing to the same file(s)
- Again, skips UFS, and host files overwrite any in container
- Can't use in Dockerfile, must be at `container run`
- `... run -v /Users/bret/stuff:/path/container` (mac/linux)
- `... run -v //c/Users/bret/stuff:/path/container` (windows)

Assignment: Named Volumes

- Database upgrade with containers
- Create a `postgres` container with named volume psql-data using version `9.6.1`
- Use Docker Hub to learn `VOLUME` path and versions needed to run it
- Check logs, stop container
- Create a new `postgres` container with same named volume using `9.6.2`
- Check logs to validate
- (this only works with patch versions, most SQL DB's require manual commands to upgrade DB's to major/minor versions, i.e. it's a DB limitation not a container one)

Assignment: Bind Mounts

- Use a Jekyll "Static Site Generator" to start a local web server
- Don't have to be web developer: this is example of bridging the gap between local file access and apps running in containers
- source code is in the course repo under `bindmount-sample-1`
- We edit files with editor on our host using native tools
- Container detects changes with host files and updates web server
- start container with
  - `docker run -p 80:4000 -v $(pwd):/site bretfisher/jekyll-serve`
- Refresh our browser to see changes
- Change the file in `_posts\` and refresh browser to see changes