Title: Docker Compose: Which syntax produces a bind mount, which produces a volume

Post Body:

In the Docker Compose documentation, here, you have the following example related to the `volumes` section of `docker-compose.yml` files:

```
volumes:   # (1) Just specify a path and let the Engine create a volume   - /var/lib/mysql    # (2) Specify an absolute path m
```

Which syntaxes produce a *bind mount* and which produce a *docker volume*? At some place of the documentation, the two concepts are strictly differentiated but at this place they are mixed together... so it is not clear to me.

Accepted Answer:

Whenever you see "volume" in the comment, that will create a **volume**: so (1) and (5).

If there is not a volume in the comment, this is about a bind mount.

The documentation regarding volumes in docker-compose is here:

> Mount host paths or named volumes, specified as sub-options to a service.
>
> You can mount a host path as part of a definition for a single service, and there is no need to define it in the top level volumes key.
>
> But, if you want to reuse a volume across multiple services, then define a named volume in the top-level volumes key.
>
> The top-level volumes key defines a named volume and references it from each service’s volumes list. This replaces volumes_from in earlier versions of the Compose file format. See Use volumes and Volume Plugins for general information on volumes.

Highest Rated Answer:

Those are two completely different concepts. A volume means that given directory will be persisted between container runs. Imagine MySQL database. You don’t want to lose your data. On the other hand there’s a bind mount where you attach your local directory to the directory in the container. If the container writes something there it will appear in your file system and vice versa (synchronization).

As a side note a volume is nothing more than a symlink to the directory on your machine :) (to a `/var/lib/docker/volumes/...` directory by default)