Title: Serving Rails' precompiled assets using nginx in Docker

Post Body:

Currently I'm setting up my app using docker. I've got a minimal rails app, with 1 controller. You can get my setup by running these:

```
rails new app --database=sqlite --skip-bundle cd app rails generate controller --skip-routes Home index echo 'Rails.applicatio
```

And I have the following setup:

`Dockerfile`:

```
FROM ruby:2.3  # Install dependencies RUN apt-get update && apt-get install -y \       nodejs \       sqlite3 \       --no-ins
```

Where `VOLUME '$APP_DIR/public'` is creating a volume that's shared with the Nginx container, which has this in the `Dockerfile`:

```
FROM nginx  ADD nginx.conf /etc/nginx/nginx.conf
```

And then `docker-compose.yml`:

```
version: '2'  services:   web:     build: config/docker/web     volumes_from:       - app     links:       - app:app     ports
```

This works, but only the first time I build it. If I change any assets, and build the images again, they're not updated. Possibly because volumes are not updated on image build, I think because how Docker handles caching.

I want the assets to be updated every time I run `docker-compose built && docker-compose up`. Any idea how to accomplish this?

Accepted Answer: None

Highest Rated Answer:

Compose preserves volumes on recreate.

You have a couple options:

1. don't use volumes for the assets, instead build the assets and `ADD` or `COPY` them into the web container during build
2. `docker-compose rm app` before running `up` to remove the old container and volumes.