nimble

Docker multi-stage builds

Rossukhon

Growth Session #25 - XXV - June 20-21 2019

Goals

- Check Docker **Multi-stage builds** feature
- Apply to The 1 project

Multi-staged builds

- Introduced in Docker 17.05 (May 2017)
- Ability to define the stages inside the dockerfile
- For optimizing Dockerfile while keeping it easy to read and maintain
- One of the best practices for writing dockerfile

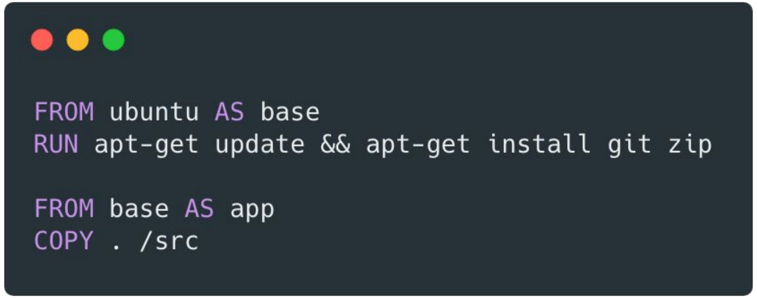
Define the stage

FROM <image>

With name

FROM <image> **AS** <name>

- Define the start of the new stage
- The image can be the remote image, local image or even the previous step name



```
FROM ubuntu AS base
RUN apt-get update && apt-get install git zip

FROM base AS app
COPY . /src
```

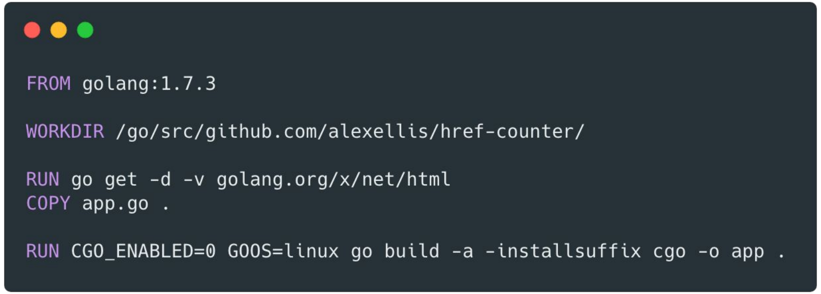
Copy artifacts

```
COPY --from=<image/stage> <source> <destination>
```

- Copy the artifacts from:
  - Another local/remote image
  - Another stage
- Useful for refactoring using the `Builder pattern`

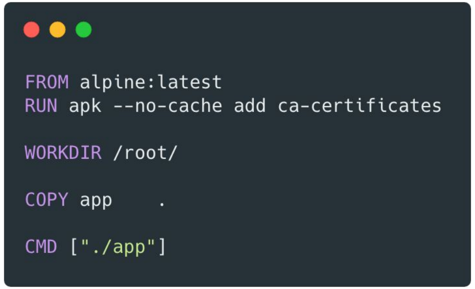
Builder pattern

Dockerfile.build



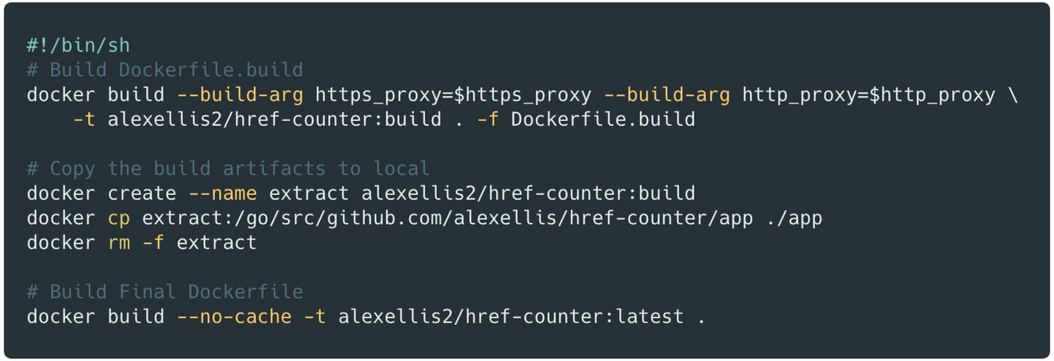
```
FROM golang:1.7.3

WORKDIR /go/src/github.com/alexellis/href-counter/

RUN go get -d -v golang.org/x/net/html
COPY app.go .

RUN CGO_ENABLED=0 GOOS=linux go build -a -installsuffix cgo -o app .
```

Dockerfile



```
FROM alpine:latest
RUN apk --no-cache add ca-certificates

WORKDIR /root/

COPY app    .

CMD ["./app"]
```

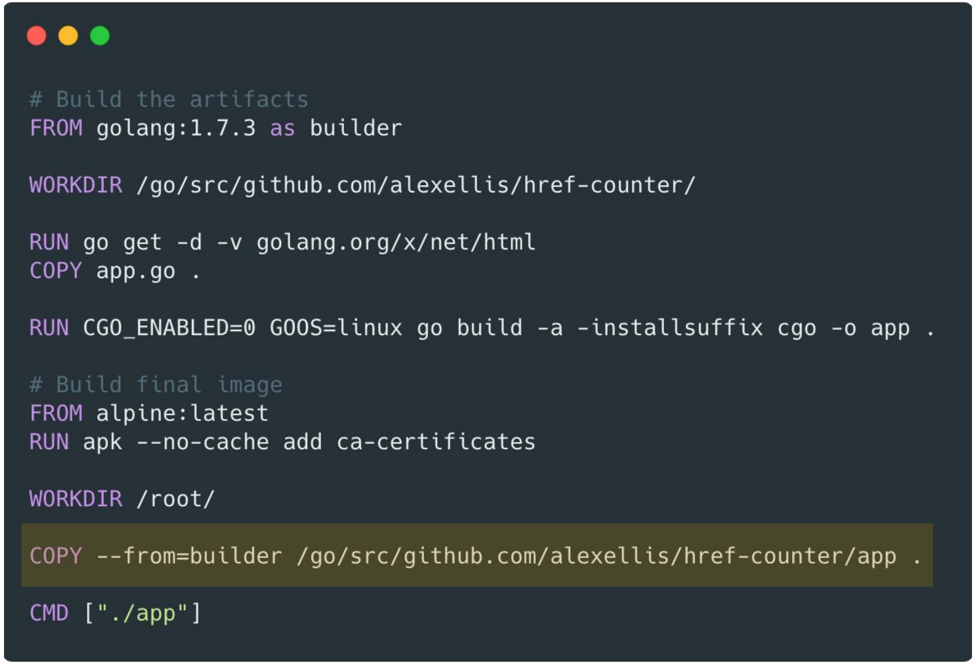
build.sh



```
#!/bin/sh
# Build Dockerfile.build
docker build --build-arg https_proxy=$https_proxy --build-arg http_proxy=$http_proxy \
    -t alexellis2/href-counter:build . -f Dockerfile.build

# Copy the build artifacts to local
docker create --name extract alexellis2/href-counter:build
docker cp extract:/go/src/github.com/alexellis/href-counter/app ./app
docker rm -f extract

# Build Final Dockerfile
docker build --no-cache -t alexellis2/href-counter:latest .
```

https://blog.alexellis.io/mutli-stage-docker-builds

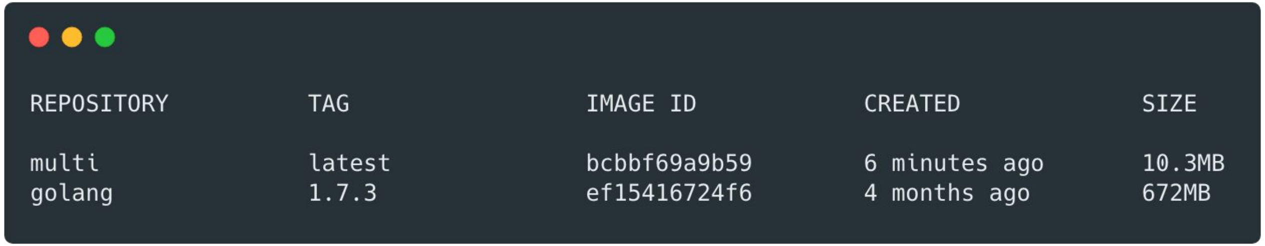
From builder pattern to multi-stage builds



```dockerfile
# Build the artifacts
FROM golang:1.7.3 as builder

WORKDIR /go/src/github.com/alexellis/href-counter/

RUN go get -d -v golang.org/x/net/html
COPY app.go .

RUN CGO_ENABLED=0 GOOS=linux go build -a -installsuffix cgo -o app .

# Build final image
FROM alpine:latest
RUN apk --no-cache add ca-certificates

WORKDIR /root/

COPY --from=builder /go/src/github.com/alexellis/href-counter/app .

CMD ["./app"]
```

https://blog.alexellis.io/mutli-stage-docker-builds

From builder pattern to multi-stage builds

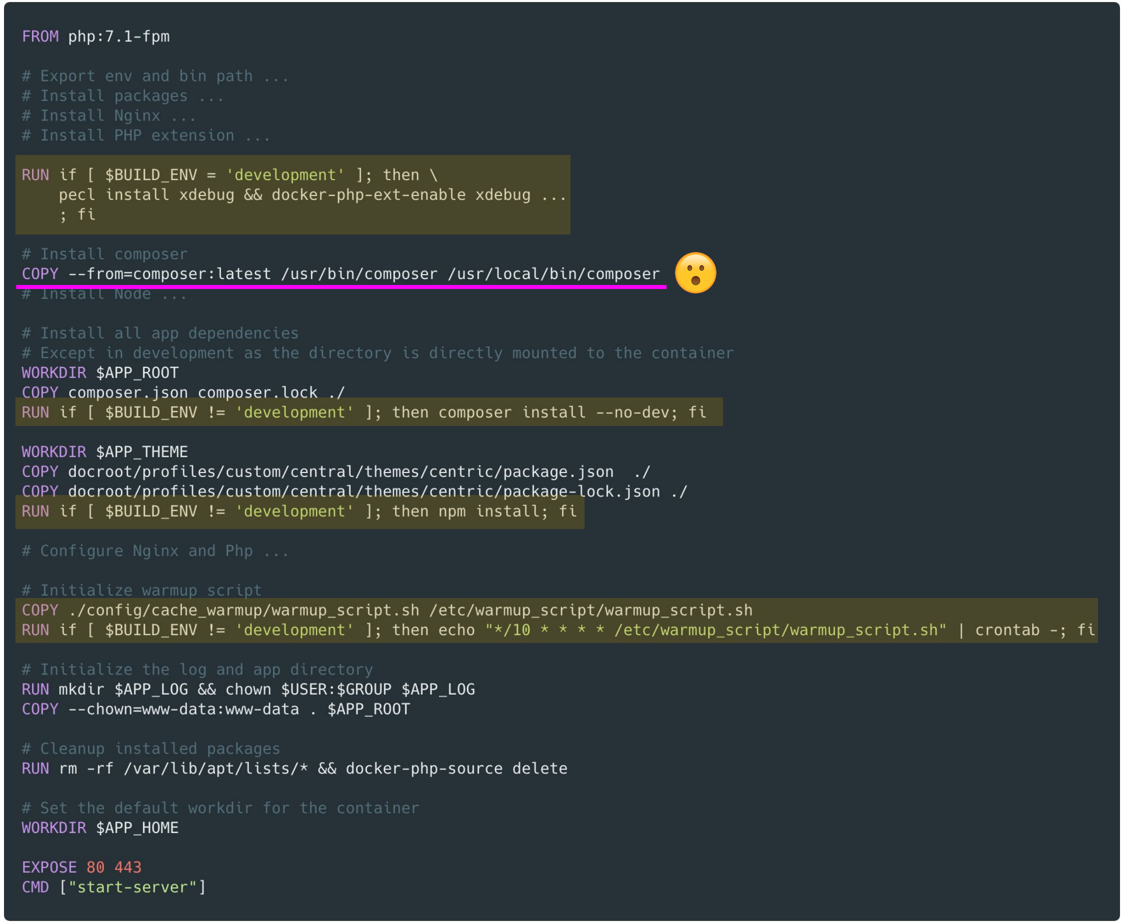


```
REPOSITORY          TAG                 IMAGE ID            CREATED             SIZE

multi               latest              bcbbf69a9b59        6 minutes ago       10.3MB
golang              1.7.3               ef15416724f6        4 months ago        672MB
```

Slimmed down version of image for production

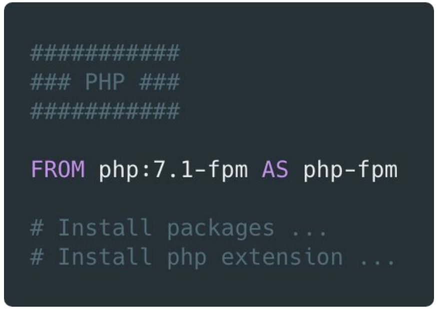
https://blog.alexellis.io/mutli-stage-docker-builds

Environment-specific setup

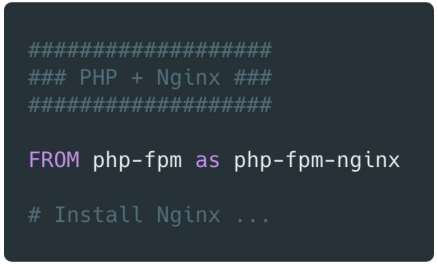


```
FROM php:7.1-fpm

# Export env and bin path ...
# Install packages ...
# Install Nginx ...
# Install PHP extension ...

RUN if [ $BUILD_ENV = 'development' ]; then \
    pecl install xdebug && docker-php-ext-enable xdebug ...
    ; fi

# Install composer
COPY --from=composer:latest /usr/bin/composer /usr/local/bin/composer
# Install Node ...

# Install all app dependencies
# Except in development as the directory is directly mounted to the container
WORKDIR $APP_ROOT
COPY composer.json composer.lock ./
RUN if [ $BUILD_ENV != 'development' ]; then composer install --no-dev; fi

WORKDIR $APP_THEME
COPY docroot/profiles/custom/central/themes/centric/package.json  ./
COPY docroot/profiles/custom/central/themes/centric/package-lock.json ./
RUN if [ $BUILD_ENV != 'development' ]; then npm install; fi

# Configure Nginx and Php ...

# Initialize warmup script
COPY ./config/cache_warmup/warmup_script.sh /etc/warmup_script/warmup_script.sh
RUN if [ $BUILD_ENV != 'development' ]; then echo "*/10 * * * * /etc/warmup_script/warmup_script.sh" | crontab -; fi

# Initialize the log and app directory
RUN mkdir $APP_LOG && chown $USER:$GROUP $APP_LOG
COPY --chown=www-data:www-data . $APP_ROOT

# Cleanup installed packages
RUN rm -rf /var/lib/apt/lists/* && docker-php-source delete

# Set the default workdir for the container
WORKDIR $APP_HOME

EXPOSE 80 443
CMD ["start-server"]
```

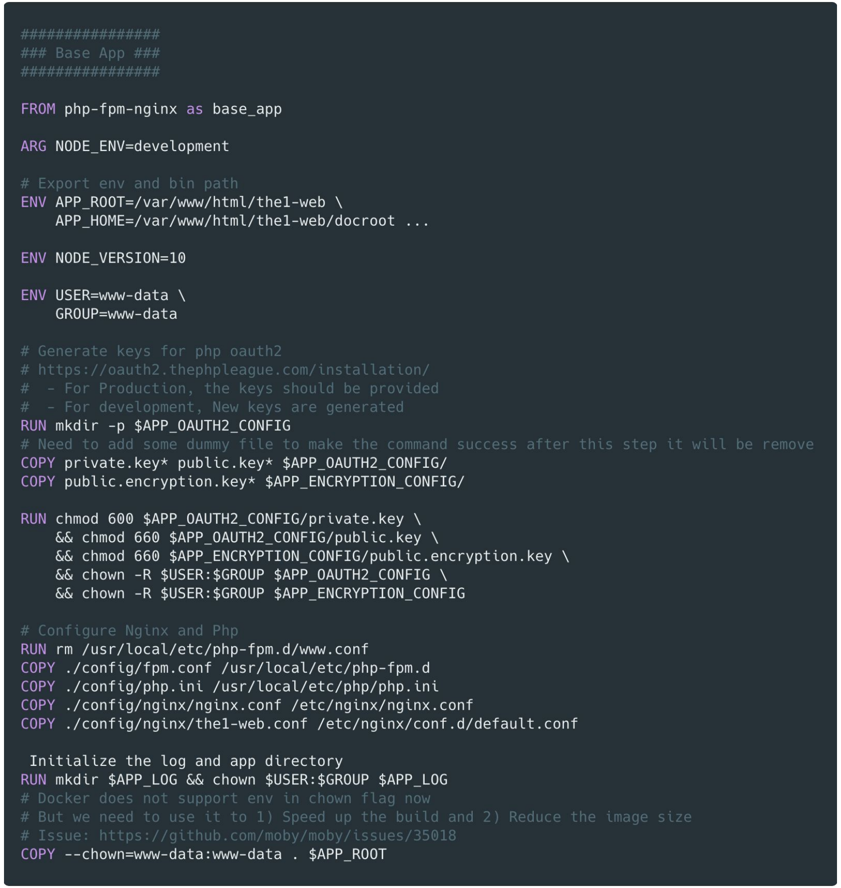
Refactor to multi-stage builds



```
###########
### PHP ###
###########

FROM php:7.1-fpm AS php-fpm

# Install packages ...
# Install php extension ...
```

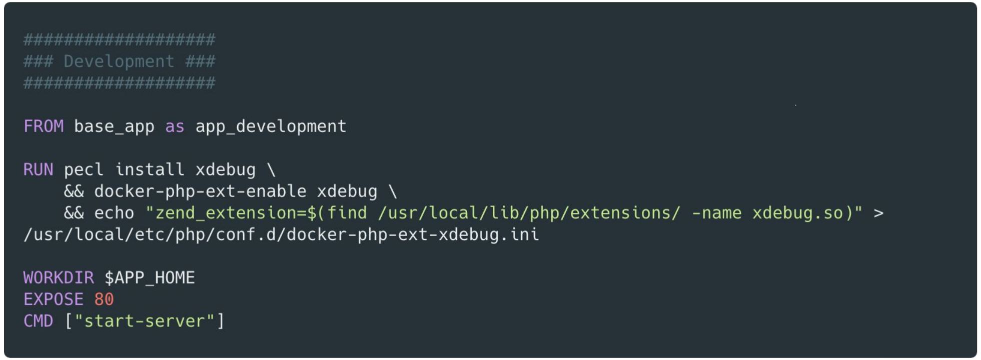


```
###################
### PHP + Nginx ###
###################

FROM php-fpm as php-fpm-nginx

# Install Nginx ...
```

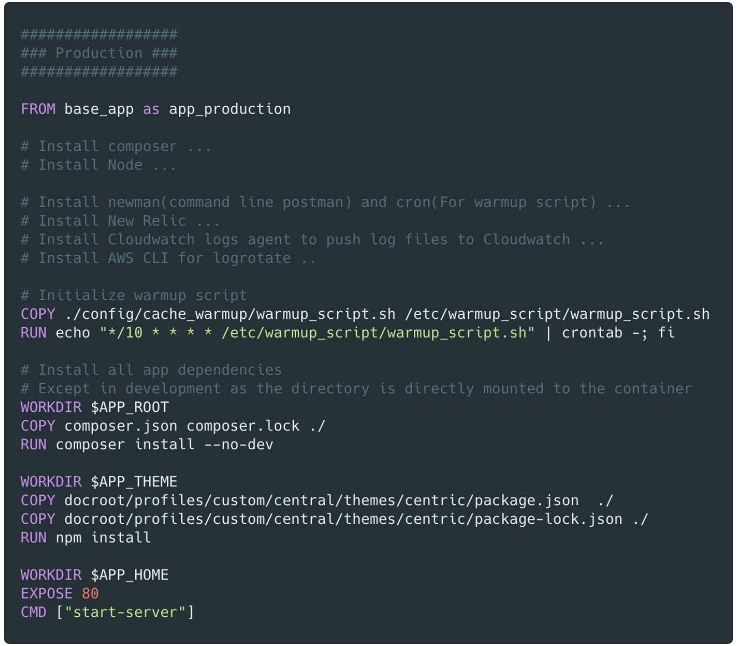


```
################
### Base App ###
################

FROM php-fpm-nginx as base_app

ARG NODE_ENV=development

# Export env and bin path
ENV APP_ROOT=/var/www/html/the1-web \
    APP_HOME=/var/www/html/the1-web/docroot ...

ENV NODE_VERSION=10

ENV USER=www-data \
    GROUP=www-data

# Generate keys for php oauth2
# https://oauth2.thephpleague.com/installation/
#  - For Production, the keys should be provided
#  - For development, New keys are generated
RUN mkdir -p $APP_OAUTH2_CONFIG
# Need to add some dummy file to make the command success after this step it will be remove
COPY private.key* public.key* $APP_OAUTH2_CONFIG/
COPY public.encryption.key* $APP_ENCRYPTION_CONFIG/

RUN chmod 600 $APP_OAUTH2_CONFIG/private.key \
    && chmod 660 $APP_OAUTH2_CONFIG/public.key \
    && chmod 660 $APP_ENCRYPTION_CONFIG/public.encryption.key \
    && chown -R $USER:$GROUP $APP_OAUTH2_CONFIG \
    && chown -R $USER:$GROUP $APP_ENCRYPTION_CONFIG

# Configure Nginx and Php
RUN rm /usr/local/etc/php-fpm.d/www.conf
COPY ./config/fpm.conf /usr/local/etc/php-fpm.d
COPY ./config/php.ini /usr/local/etc/php/php.ini
COPY ./config/nginx/nginx.conf /etc/nginx/nginx.conf
COPY ./config/nginx/the1-web.conf /etc/nginx/conf.d/default.conf

 Initialize the log and app directory
RUN mkdir $APP_LOG && chown $USER:$GROUP $APP_LOG
# Docker does not support env in chown flag now
# But we need to use it to 1) Speed up the build and 2) Reduce the image size
# Issue: https://github.com/moby/moby/issues/35018
COPY --chown=www-data:www-data . $APP_ROOT
```

Refactor to multi-stage builds

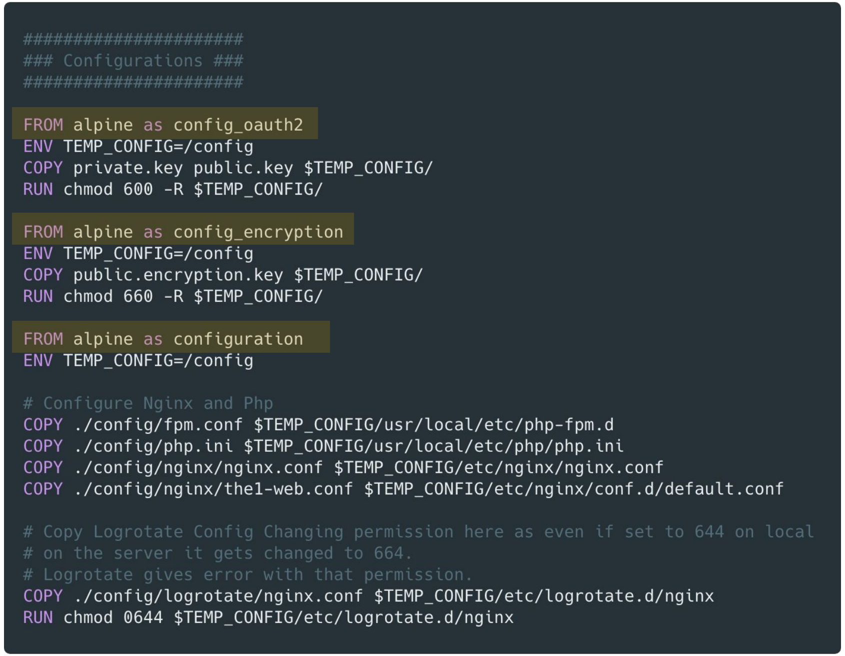


```
###################
### Development ###
###################

FROM base_app as app_development

RUN pecl install xdebug \
    && docker-php-ext-enable xdebug \
    && echo "zend_extension=$(find /usr/local/lib/php/extensions/ -name xdebug.so)" >
/usr/local/etc/php/conf.d/docker-php-ext-xdebug.ini

WORKDIR $APP_HOME
EXPOSE 80
CMD ["start-server"]
```

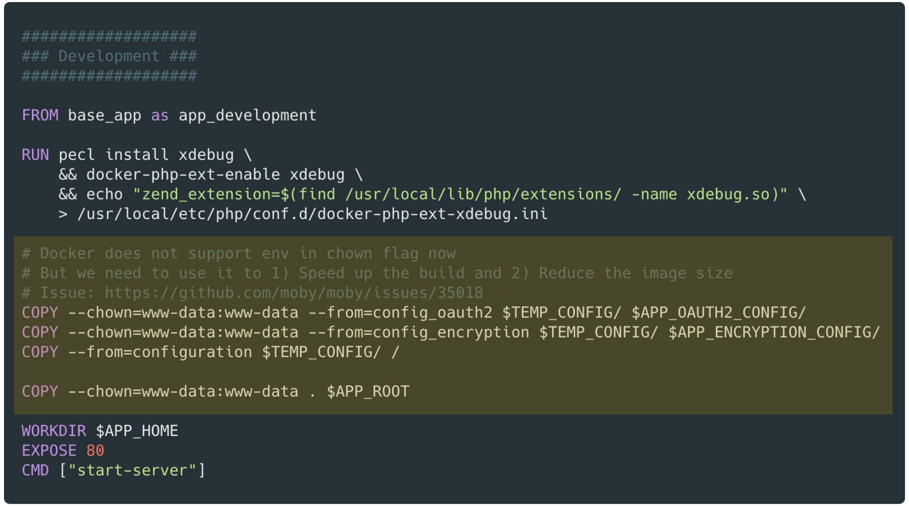


```
##################
### Production ###
##################

FROM base_app as app_production

# Install composer ...
# Install Node ...

# Install newman(command line postman) and cron(For warmup script) ...
# Install New Relic ...
# Install Cloudwatch logs agent to push log files to Cloudwatch ...
# Install AWS CLI for logrotate ..

# Initialize warmup script
COPY ./config/cache_warmup/warmup_script.sh /etc/warmup_script/warmup_script.sh
RUN echo "*/10 * * * * /etc/warmup_script/warmup_script.sh" | crontab -; fi

# Install all app dependencies
# Except in development as the directory is directly mounted to the container
WORKDIR $APP_ROOT
COPY composer.json composer.lock ./
RUN composer install --no-dev

WORKDIR $APP_THEME
COPY docroot/profiles/custom/central/themes/centric/package.json  ./
COPY docroot/profiles/custom/central/themes/centric/package-lock.json ./
RUN npm install

WORKDIR $APP_HOME
EXPOSE 80
CMD ["start-server"]
```

Refactor to multi-stage builds



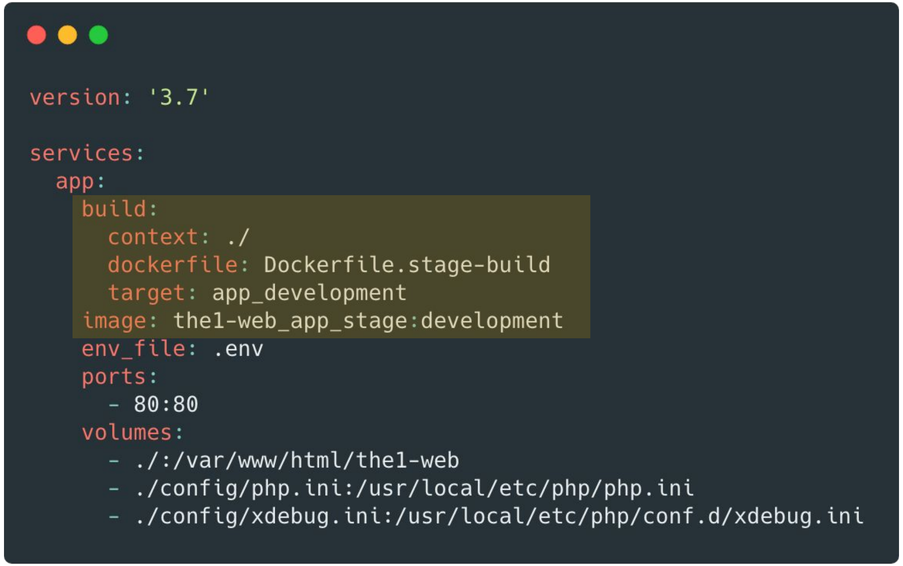
```
######################
### Configurations ###
######################

FROM alpine as config_oauth2
ENV TEMP_CONFIG=/config
COPY private.key public.key $TEMP_CONFIG/
RUN chmod 600 -R $TEMP_CONFIG/

FROM alpine as config_encryption
ENV TEMP_CONFIG=/config
COPY public.encryption.key $TEMP_CONFIG/
RUN chmod 660 -R $TEMP_CONFIG/

FROM alpine as configuration
ENV TEMP_CONFIG=/config

# Configure Nginx and Php
COPY ./config/fpm.conf $TEMP_CONFIG/usr/local/etc/php-fpm.d
COPY ./config/php.ini $TEMP_CONFIG/usr/local/etc/php/php.ini
COPY ./config/nginx/nginx.conf $TEMP_CONFIG/etc/nginx/nginx.conf
COPY ./config/nginx/the1-web.conf $TEMP_CONFIG/etc/nginx/conf.d/default.conf

# Copy Logrotate Config Changing permission here as even if set to 644 on local
# on the server it gets changed to 664.
# Logrotate gives error with that permission.
COPY ./config/logrotate/nginx.conf $TEMP_CONFIG/etc/logrotate.d/nginx
RUN chmod 0644 $TEMP_CONFIG/etc/logrotate.d/nginx
```



```
###################
### Development ###
###################

FROM base_app as app_development

RUN pecl install xdebug \
    && docker-php-ext-enable xdebug \
    && echo "zend_extension=$(find /usr/local/lib/php/extensions/ -name xdebug.so)" \
    > /usr/local/etc/php/conf.d/docker-php-ext-xdebug.ini

# Docker does not support env in chown flag now
# But we need to use it to 1) Speed up the build and 2) Reduce the image size
# Issue: https://github.com/moby/moby/issues/35018
COPY --chown=www-data:www-data --from=config_oauth2 $TEMP_CONFIG/ $APP_OAUTH2_CONFIG/
COPY --chown=www-data:www-data --from=config_encryption $TEMP_CONFIG/ $APP_ENCRYPTION_CONFIG/
COPY --from=configuration $TEMP_CONFIG/ /

COPY --chown=www-data:www-data . $APP_ROOT

WORKDIR $APP_HOME
EXPOSE 80
CMD ["start-server"]
```

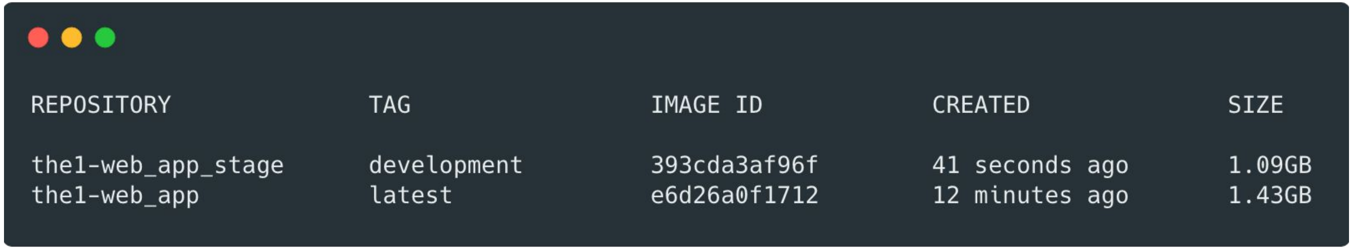
Build at specific stage

docker build --target <stage> -t <image name> .



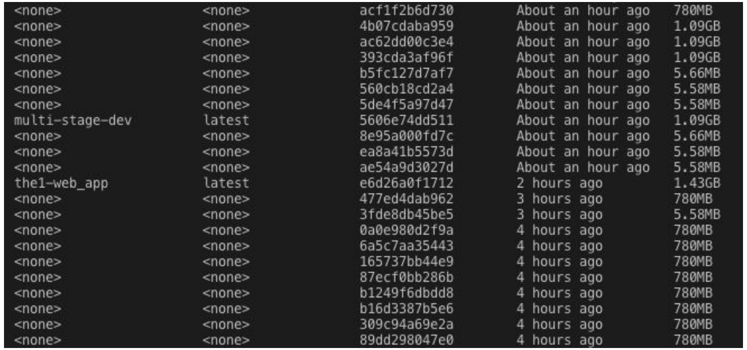
```
version: '3.7'

services:
  app:
    build:
      context: ./
      dockerfile: Dockerfile.stage-build
      target: app_development
    image: the1-web_app_stage:development
    env_file: .env
    ports:
      - 80:80
    volumes:
      - ./:/var/www/html/the1-web
      - ./config/php.ini:/usr/local/etc/php/php.ini
      - ./config/xdebug.ini:/usr/local/etc/php/conf.d/xdebug.ini
```

Environment-specific stage builds



```
REPOSITORY           TAG           IMAGE ID        CREATED            SIZE

the1-web_app_stage   development   393cda3af96f    41 seconds ago     1.09GB
the1-web_app         latest        e6d26a0f1712    12 minutes ago     1.43GB
```

Reduce a bit the size



```
<none>              <none>      acf1f2b6d730    About an hour ago   780MB
<none>              <none>      4b07cdaba959    About an hour ago   1.09GB
<none>              <none>      ac62dd00c3e4    About an hour ago   1.09GB
<none>              <none>      393cda3af96f    About an hour ago   1.09GB
<none>              <none>      b5fc127d7af7    About an hour ago   5.66MB
<none>              <none>      560cb18cd2a4    About an hour ago   5.58MB
<none>              <none>      5de4f5a97d47    About an hour ago   5.58MB
multi-stage-dev     latest      5606e74dd511    About an hour ago   1.09GB
<none>              <none>      8e95a000fd7c    About an hour ago   5.66MB
<none>              <none>      ea8a41b5573d    About an hour ago   5.58MB
<none>              <none>      ae54a9d3027d    About an hour ago   5.58MB
the1-web_app        latest      e6d26a0f1712    2 hours ago         1.43GB
<none>              <none>      477ed4dab962    3 hours ago         780MB
<none>              <none>      3fde8db45be5    3 hours ago         5.58MB
<none>              <none>      0a0e980d2f9a    4 hours ago         780MB
<none>              <none>      6a5c7aa35443    4 hours ago         780MB
<none>              <none>      165737bb44e9    4 hours ago         780MB
<none>              <none>      87ecf0bb286b    4 hours ago         780MB
<none>              <none>      b1249f6dbdd8    4 hours ago         780MB
<none>              <none>      b16d3387b5e6    4 hours ago         780MB
<none>              <none>      309c94a69e2a    4 hours ago         780MB
<none>              <none>      89dd298047e0    4 hours ago         780MB
```

Thanks!

Contact Nimble

nimblehq.co
hello@nimblehq.co

Bangkok

399 Interchange 21 Sukhumvit Road, Unit
#2402-03, Klong Toei, Wattana, Bangkok
10110, Thailand

Singapore

28C Stanley St, Singapore 068737

Hong Kong

20th Floor, Central Tower
28 Queen's Road, Central, Hong Kong

nimble