Title: Best way to reduce the size of a custom Docker image

Post Body:

I've built a custom Docker image based off an official PHP FPM image `php:7.0.14-fpm-alpine`

I wanted to keep the size of the image small, so I went for the official `alpine` PHP-FPM version as it weighs only **27 MB**.

I installed only few additional packages through my `Dockerfile`, and the image grew in size to as much as **277.5 MB**. Here is my Dockerfile:

```
FROM php:7.0.14-fpm-alpine  COPY ./config/www-pool.conf /usr/local/etc/php-fpm.d/www.conf COPY ./scripts/download-composer.sh
```

277.5 MB is a ten-fold increase in size compared to the base image. Apart from Composer, all I needed were several PHP extensions:

- mongodb
- xdebug
- pdo
- soap

I'm not sure what contributed the most to increasing the size of my image so much. I suspect that it might be due to the `dev` dependencies that needed to be installed in order to successfully run `pecl` (`openssl-dev`, `libxml2-dev`), and that might have installed their own tree of dependencies.

Could you please advise on how I can reduce the size of my custom PHP-FPM image and still keep the necessary extensions?

Accepted Answer:

I built the initial part of your image two different ways to test this. A common answer to this question is that the package index takes up extra space. In the case of Alpine Linux (using APK), you can clean up the package index like this:

```
rm -rf /var/cache/apk/*
```

However, I built the first part of the image both with and without that cleanup. It made hardly any difference (0.8 MB).

```
FROM php:7.0.14-fpm-alpine  WORKDIR /root  RUN ['mkdir', '/var/log/php-fpm']  RUN apk --update add \       autoconf g++ make \
```

Whether the cleanup command is present or not, the image weighs in at 267MB.

```
REPOSITORY             TAG      IMAGE ID       CREATED          SIZE php-fpm-alpine-test2   latest   b87f5e2d629d   23 seconds
```

The space used is simply the packages you're installing.

```
Step 4 : RUN apk --update add       autoconf g++ make       openssl-dev       libxml2-dev   && rm -rf /var/cache/apk/*  ---> R
```

As you can see from the summary at the end of this installation, apk has installed 220 MiB of new content.

My best advice would be to run all of your installation and then you can try to remove some of the packages that are only needed for build, not at run-time. For instance you may not need some of the dev packages anymore, or the compiler, automake, etc.

However, you have to keep in mind that each RUN command makes a new layer. To actually save space this way, you would have to run your apk command, all of your other installs, and the post-install cleanup, in a single RUN command, to make a single layer of it. Otherwise, whether you clean up or not, the earlier layers will still have the content and will still contribute to the image size.

```
RUN apk --update add \       autoconf g++ make \       openssl-dev \       libxml2-dev \     && pecl install \         xdebug
```

Highest Rated Answer: None