Docker

Como instalar:
Descomplicando Docker | #1: Instalando Docker no Linux

Vídeo aulas:
Curso de Docker para iniciantes - aprenda Docker em 1 hora
Introdução ao Docker para iniciantes | Docker Tutorial #docker

Comandos para criar uma imagem docker e utilizá-la num container:

No terminal
primeiramente criar uma pasta:
`mkdir docker`
isso é muito importante pelo fato de ser a pasta raiz da imagem, e nela vai ser inserido muitos elementos para a criação e uso dela

depois entrar no diretório desta pasta:
`cd docker`

criar e entrar no arquivo de texto:
`nano Dockerfile`

Exemplo do que deve ter num arquivo Dockerfile, nesse exemplo já tem a configuração do nginx incluída, ou seja, não é necessário criar um arquivo nginx.conf:

```
FROM debian:11

# Instalação do Debian e atualizações
RUN apt-get update
RUN apt-get upgrade -y

# Instalação de facilitadores
RUN apt-get -y install locate mlocate wget apt-utils curl apt-transport-https lsb-release \
    ca-certificates software-properties-common zip unzip vim rpl apt-utils

# Correção do 'add-apt-repository command not found'
RUN apt-get install software-properties-common

# Instalação do PHP-FPM
```

```
RUN apt-get update && apt-get install -y \
    php7.4-fpm

RUN apt-get install -y php7.4-mysqlnd

#PHP Install PDO MySQL
RUN apt-get -y install php7.4-pdo php7.4-pdo-mysql php7.4-mysql

#Driver sqlite
RUN apt-get -y install php7.4-sqlite3

# Instalação do PHPUnit
RUN wget -O /usr/local/bin/phpunit-9.phar https://phar.phpunit.de/phpunit-9.0.phar; \
    chmod +x /usr/local/bin/phpunit-9.phar; \
    ln -s /usr/local/bin/phpunit-9.phar /usr/local/bin/phpunit

## Configuração personalizada do PHP para Adianti
# Set PHP custom settings
RUN echo "\n# Custom settings"                                  >> /etc/php/7.4/fpm/php.ini \
    && echo "memory_limit = 256M"                               >> /etc/php/7.4/fpm/php.ini \
    && echo "max_execution_time = 120"                          >> /etc/php/7.4/fpm/php.ini
\
    && echo "file_uploads = On"                                 >> /etc/php/7.4/fpm/php.ini \
    && echo "post_max_size = 100M"                              >> /etc/php/7.4/fpm/php.ini \
    && echo "upload_max_filesize = 100M"                        >>
/etc/php/7.4/fpm/php.ini \
    && echo "session.gc_maxlifetime = 14000"                    >>
/etc/php/7.4/fpm/php.ini \
    && echo "display_errors = On"                               >> /etc/php/7.4/fpm/php.ini \
    && echo "error_reporting = E_ALL & ~E_DEPRECATED & ~E_STRICT" >>
/etc/php/7.4/fpm/php.ini

# Set PHP security settings
RUN echo "\n# Security settings"                  >> /etc/php/7.4/fpm/php.ini \
    && echo "session.name = CUSTOMSESSID"         >> /etc/php/7.4/fpm/php.ini \
    && echo "session.use_only_cookies = 1"        >> /etc/php/7.4/fpm/php.ini \
    && echo "session.cookie_httponly = true"      >> /etc/php/7.4/fpm/php.ini \
    && echo "session.use_trans_sid = 0"           >> /etc/php/7.4/fpm/php.ini \
    && echo "session.entropy_file = /dev/urandom" >> /etc/php/7.4/fpm/php.ini \
    && echo "session.entropy_length = 32"         >> /etc/php/7.4/fpm/php.ini

## Instalação de pré-requisitos para o Drive SQL Server
RUN apt-get -y install php7.4-dev php7.4-xml php7.4-intl unixodbc-dev
```

```
# Definição da variável de ambiente ACCEPT_EULA
ENV ACCEPT_EULA=Y

# Configuração da chave GPG e lista de fontes para o repositório Microsoft SQL Server
RUN curl -s https://packages.microsoft.com/keys/microsoft.asc | apt-key add - \
    && curl -s https://packages.microsoft.com/config/debian/9/prod.list > /etc/apt/sources.list.d/mssql-release.list

RUN apt-get update

# Instalação de pacotes adicionais
RUN apt-get install -y --no-install-recommends \
    locales \
    apt-transport-https \
    && echo "en_US.UTF-8 UTF-8" > /etc/locale.gen \
    && locale-gen

## Instalação do Drive 5.9.0 para SQL Server
RUN pecl install sqlsrv-5.9.0
RUN pecl install pdo_sqlsrv-5.9.0

# Configuração para PHP CLI
RUN echo extension=pdo_sqlsrv.so >> `php --ini | grep "Scan for additional .ini files" | sed -e "s|.*:\s*||"`/30-pdo_sqlsrv.ini
RUN echo extension=sqlsrv.so >> `php --ini | grep "Scan for additional .ini files" | sed -e "s|.*:\s*||"`/20-sqlsrv.ini

# Configuração para PHP WEB
RUN echo "extension=pdo_sqlsrv.so" >> /etc/php/7.4/fpm/conf.d/30-pdo_sqlsrv.ini
RUN echo "extension=sqlsrv.so" >> /etc/php/7.4/fpm/conf.d/20-sqlsrv.ini

# Instalação do Nginx
RUN apt-get update && apt-get install -y \
    nginx

# Configuração do nginx.conf
RUN echo "user www-data;" > /etc/nginx/nginx.conf \
    && echo "worker_processes 1;" >> /etc/nginx/nginx.conf \
    && echo "" >> /etc/nginx/nginx.conf \
    && echo "events {" >> /etc/nginx/nginx.conf \
    && echo "    worker_connections 1024;" >> /etc/nginx/nginx.conf \
    && echo "}" >> /etc/nginx/nginx.conf \
    && echo "" >> /etc/nginx/nginx.conf \
```

```
    && echo "http {" >> /etc/nginx/nginx.conf \
    && echo "    include /etc/nginx/mime.types;" >> /etc/nginx/nginx.conf \
    && echo "    default_type application/octet-stream;" >> /etc/nginx/nginx.conf \
    && echo "" >> /etc/nginx/nginx.conf \
    && echo "    sendfile on;" >> /etc/nginx/nginx.conf \
    && echo "    keepalive_timeout 65;" >> /etc/nginx/nginx.conf \
    && echo "" >> /etc/nginx/nginx.conf \
    && echo "    server {" >> /etc/nginx/nginx.conf \
    && echo "        listen 880 default_server;" >> /etc/nginx/nginx.conf \
    && echo "        listen [::]:880 default_server;" >> /etc/nginx/nginx.conf \
    && echo "" >> /etc/nginx/nginx.conf \
    && echo "        root /var/www/html/template;" >> /etc/nginx/nginx.conf \
    && echo "" >> /etc/nginx/nginx.conf \
    && echo "        index index.php index.html index.htm index.nginx-debian.html;" >>
/etc/nginx/nginx.conf \
    && echo "" >> /etc/nginx/nginx.conf \
    && echo "        server_name _;" >> /etc/nginx/nginx.conf \
    && echo "" >> /etc/nginx/nginx.conf \
    && echo "        location / {" >> /etc/nginx/nginx.conf \
    && echo "            try_files \$uri \$uri/ =404;" >> /etc/nginx/nginx.conf \
    && echo "" >> /etc/nginx/nginx.conf \
    && echo "            rewrite ^/auth$
/rest.php?class=ApplicationAuthenticationRestService&method=getToken last;" >>
/etc/nginx/nginx.conf \
    && echo "            rewrite ^/api/([\w]+)/([\w]+)$
/rest.php?class=$1RestService&method=handle&id=$2&$args last;" >>
/etc/nginx/nginx.conf \
    && echo "            rewrite ^/api/([\w]+)/([\w]+)/([\w]+)$
/rest.php?class=$1RestService&method=$3&id=$2&$args last;" >>
/etc/nginx/nginx.conf \
    && echo "            rewrite ^/api/([\w]+)$
/rest.php?class=$1RestService&method=handle&$args last;" >>
/etc/nginx/nginx.conf \
    && echo "        }" >> /etc/nginx/nginx.conf \
    && echo "" >> /etc/nginx/nginx.conf \
    && echo "        location ~ \.php$ {" >> /etc/nginx/nginx.conf \
    && echo "            include snippets/fastcgi-php.conf;" >> /etc/nginx/nginx.conf \
    && echo "            fastcgi_pass unix:/var/run/php/php7.4-fpm.sock;" >>
/etc/nginx/nginx.conf \
    && echo "        }" >> /etc/nginx/nginx.conf \
    && echo "    }" >> /etc/nginx/nginx.conf \
    && echo "}" >> /etc/nginx/nginx.conf

# Copia o arquivo ou pasta para o diretório do container para rodar no navegador
```

```
COPY info.php /var/www/html/
COPY template/ /var/www/html/template/
COPY teste.php /var/www/html

# Exporta a porta 880 para o Nginx
EXPOSE 880

# Inicialização dos serviços
CMD service php7.4-fpm start && nginx -g "daemon off;"
```

Esse info.php, teste.php e o template serve de exemplo, pode ser outro arquivo ou pasta, porém ele precisa estar no mesmo diretório do Dockerfile, ou seja, na pasta raiz do projeto.

estando no diretório onde se encontra o Dockerfile, usar esse comando para criar a imagem:
`docker build -t balarotte/new_php .`
esse balarotte/new_php será o nome dado a imagem

caso seja alterado algo do arquivo de texto Dockerfile é necessário usar o comando novamente para atualizar a imagem:
`docker build -t balarotte/new_php .`

para ver se a imagem foi criada:
`docker images`

para inicializar o container a partir da imagem:

`docker run -d --network=host -p 880:880 balarotte/new_php`

nesse comando tem a especificação das portas por conta do nginx, e essa questão do network=host é para o container ter acesso ao banco de dados da máquina local

para ver se o container existe:
`docker ps -a`

para ver se o container foi inicializado:
`docker ps`

para ver as logs do container:
`docker logs 637d5c0aace8`
esses números representa o id do container

caso precise dar permissão para algum arquivo ou diretório de dentro do docker:
`docker exec -it eeebbe2f3aa5 chmod 777 -R /var/www/html/teste.php`
nessa parte onde tem vários números é o id do container

para reiniciar o servidor nginx e php do container:

`docker exec -it eeebbe2f3aa5 service php7.4-fpm restart`
`docker exec -it eeebbe2f3aa5 nginx -s restart`

porém reiniciando o servidor nginx ele vai finalizar o container, para reiniciar o container é necessário fazer o comando:

`docker start eeebbe2f3aa5`
`docker start id_do_container`

para acessar o terminal em si do sistema operacional que foi instalado no container:
`docker exec -it 2be0c92e0df3 /bin/bash`

ver os logs do nginx após entrar no terminal do sistema operacional do container:

`cd /var/log/nginx`
`tail -f error.log`

para reiniciar o servidor nginx e o php no próprio terminal do sistema operacional do container:

`service php7.4-fpm restart`
`service nginx restart`

para sair do terminal do sistema operacional do container:
`exit`

Uma dica: instalar a extensão `docker` no visual studio code, pois com ela é possível dar start, excluir e ver os diretórios do container, o que ajuda muito

Exemplo de Docker com Adianti:

https://github.com/bjverde/formDocker/tree/master/adianti_debian11_php8.1

Caso utilize o servidor apache ao invés do nginx deverá fazer isso:

ir no diretório do container `etc/apache2/ports.conf` e mudar a porta para a que foi especificada no container

Comando para restartar o apache:
dentro do terminal do sistema operacional do container:

```
service apache2 restart
```