Docker Compose

Preparar el entorno de trabajo

Instalar Docker Compose

```
sudo apt-get -y install python-pip
sudo pip install docker-compose
```

Crear un container

Creamos un directorio y dentro de él un archivo para definir nuestros containers:

```
mkdir hello-world
cd hello-world
nano docker-compose.yml
```

Ingresar el siguiente texto en el archivo:

```
my-test:
  image: hello-world
```

Guardar y cerrar (CTRL-O y luego CTRL-X)

En el mismo directorio donde está el archivo .yml ejecutar

```
docker-compose up
```

Crear un container Worpress

Creamos un directorio y dentro de él un archivo para definir nuestros containers:

```
mkdir wordpress
cd wordpress
nano docker-compose.yml
```

Ingresar el siguiente texto en el archivo:

```
wordpress:
  image: wordpress

  ports:
    - 8088:80
```

Guardar y cerrar (CTRL-O y luego CTRL-X)

En el mismo directorio donde está el archivo .yml ejecutar

```
docker-compose up -d
```

Detenemos el container con:

```
docker-compose stop
```

Ahora agregamos el container para la base de datos editando el **composer.yml** para que quede:

```
wordpress:
  image: wordpress

  ports:
    - 8088:80

  links:
    - wordpress_db:mysql

wordpress_db:
  image: mariadb

  environment:
     MYSQL_ROOT_PASSWORD: examplepass
```

Ejecutar nuevamente

```
docker-compose up -d
```

Navegar a la página http://localhost:8088

Estará la página de instalación de wordpress. Instalarlo.

Veamos los containers con:

```
docker ps
```

Podemos agregar además la herramienta de adminstración de base de datos PHPMyAdmin.

Detener los containers con:

```
docker-compose stop
```

Ahora agregamos el container de PHPMyAdmin editando el **composer.yml** para que quede:

```
wordpress:
  image: wordpress

  ports:
    - 8088:80

  links:
    - wordpress_db:mysql

wordpress_db:
  image: mariadb

  environment:
     MYSQL_ROOT_PASSWORD: examplepass

phpmyadmin:
  image: corbinu/docker-phpmyadmin

  links:
    - wordpress_db:mysql

  ports:
    - 8188:80

  environment:
    MYSQL_USERNAME: root
    MYSQL_ROOT_PASSWORD: examplepass
```

Ejecutar nuevamente

```
docker-compose up -d
```

PHPMyAdmin debería estar disponibe en http://localhost:8188

Veamos los containers con:

```
docker ps
```

Ahora mapearemos el directorio de trabajo con el directorio de archivos de wordpess

Detener los containers con:
docker-compose stop

Ahora agregamos el container de PHPMyAdmin editando el **composer.yml** para que quede:

```
wordpress:
  image: wordpress

  ports:
    - 8088:80

  links:
    - wordpress_db:mysql

  volumes:
    - ~/wordpress/wp_html:/var/www/html

wordpress_db:
  image: mariadb

  environment:
     MYSQL_ROOT_PASSWORD: examplepass

phpmyadmin:
  image: corbinu/docker-phpmyadmin

  links:
    - wordpress_db:mysql

  ports:
    - 8188:80

  environment:
    MYSQL_USERNAME: root
    MYSQL_ROOT_PASSWORD: examplepass
```

Eliminamos el container wordpress con
docker-compose rm wordpress

Y ejecutamos nuevamente
docker-compose up -d

Veamos el contenido del directorio ~/wordpress

Explicar qué hace cada una de las líneas del archivo **composer.yml** final.