

DOCKER_102-JAR2DOCKER

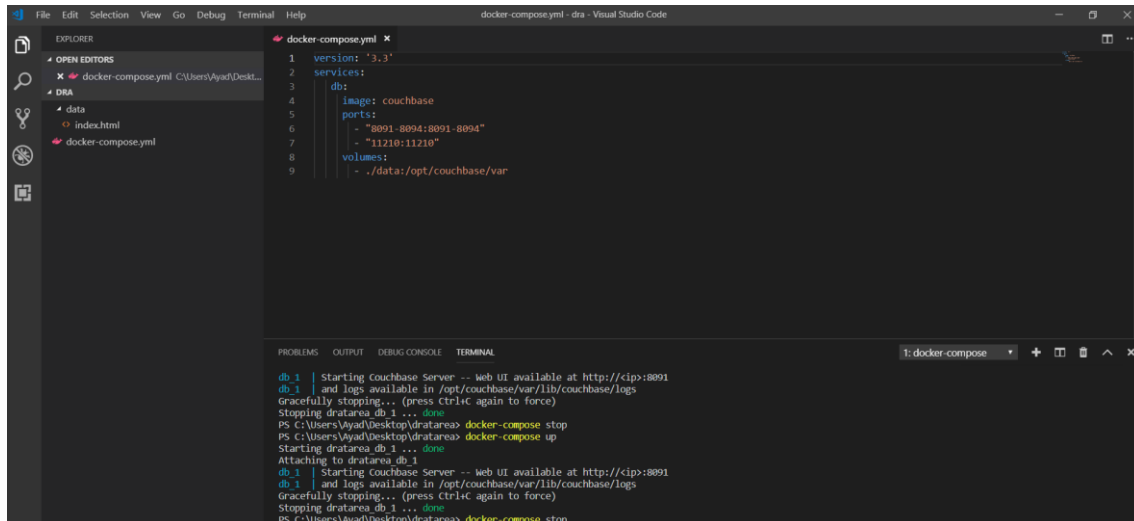
Contenido

Dockerhub Couchbase to compose	2
Comprobamos el funcionamiento usando eclipse	3
Desplegar un .jar usando Dockerfile	4

Dockerhub Couchbase to compose

1. En el siguiente ejemplo vamos a ver como pasar desde una imagen alojada en Docker Hub a un archivo docker-compose.
2. En el ejemplo usaremos la imagen de Couchbase, una base de datos documental que acepta el lenguaje de consultas SQL.

En la siguiente captura tenemos el docker-compose.yml con los datos analizados. En la terminal podemos apreciar el docker funcionando correctamente.

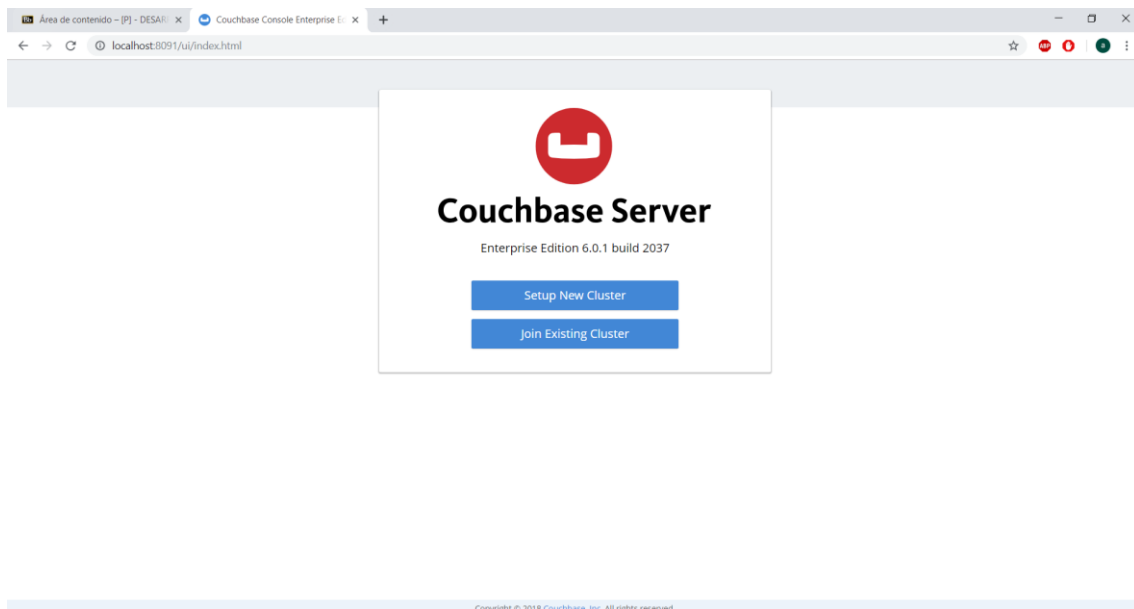


The screenshot shows a Visual Studio Code editor with a file named `docker-compose.yml` open. The file content is as follows:

```
1 version: '3.3'
2 services:
3   db:
4     image: couchbase
5     ports:
6       - "8091-8094:8091-8094"
7       - "11210:11210"
8     volumes:
9       - ./data:/opt/couchbase/var
```

The terminal at the bottom shows the execution of `docker-compose up` and `docker-compose stop` commands. The output indicates that the Couchbase Server is starting successfully, with the web UI available at `http://<ip>:8091`. The logs are available in `/opt/couchbase/var/lib/couchbase/logs`. The terminal also shows the graceful stopping of the service.

Funciona correctamente en el puerto 8091.



Hello, World!

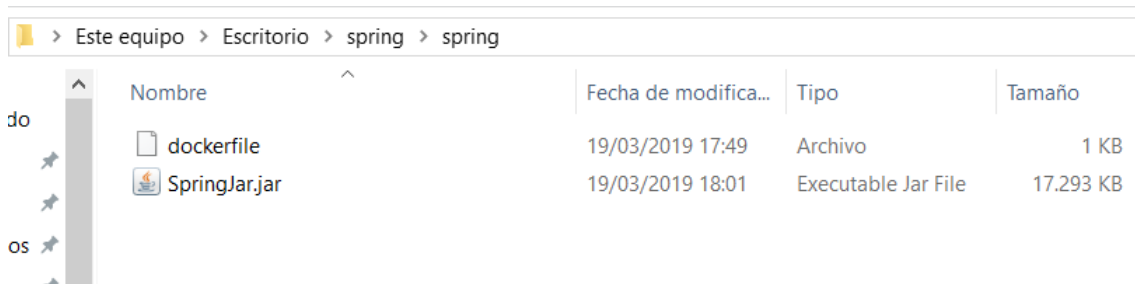
Desplegar un .jar usando Dockerfile

En el siguiente ejemplo vamos a partir de una aplicación de Spring compilada en un jar podemos generar un Dockerfile y Compose que se encargue de desplegarlo.

Exportamos nuestro proyecto a un .jar en eclipse.

```
[INFO] Scanning for projects...
[INFO]
[INFO] -----
[INFO] Building helloworld 0.1.0
[INFO] -----
[INFO]
[INFO] --- maven-resources-plugin:3.0.2:resources (default-resources) @ helloworld-serving-web-content ---
[INFO] Using 'UTF-8' encoding to copy filtered resources.
[INFO] Copying 1 resource
[INFO] Copying 2 resources
[INFO]
[INFO] --- maven-compiler-plugin:3.7.0:compile (default-compile) @ helloworld-serving-web-content ---
[INFO] Nothing to compile - all classes are up to date
[INFO]
[INFO] --- maven-resources-plugin:3.0.2:testResources (default-testResources) @ helloworld-serving-web-content ---
[INFO] Using 'UTF-8' encoding to copy filtered resources.
[INFO] skip non existing resourceDirectory C:\Users\Ayad\jar2compose\src\test\resources
[INFO]
[INFO] --- maven-compiler-plugin:3.7.0:testCompile (default-testCompile) @ helloworld-serving-web-content ---
[INFO] Nothing to compile - all classes are up to date
[INFO]
[INFO] --- maven-surefire-plugin:2.21.0:test (default-test) @ helloworld-serving-web-content ---
[INFO]
[INFO] -----
[INFO] T E S T S
[INFO] -----
[INFO] Running hello.ApplicationTest
19:45:19.668 [main] DEBUG org.springframework.test.context.junit4.SpringJUnit4ClassRunner - SpringJUnit4ClassRunner constructor called
19:45:19.676 [main] DEBUG org.springframework.test.context.BootstrapUtils - Instantiating CacheAwareContextLoaderDelegate from class [o
19:45:19.684 [main] DEBUG org.springframework.test.context.BootstrapUtils - Instantiating BootstrapContext using constructor [public or
19:45:19.705 [main] DEBUG org.springframework.test.context.BootstrapUtils - Instantiating TestContextBootstrapper for test class [hello
19:45:19.727 [main] INFO org.springframework.boot.test.autoconfigure.web.servlet.WebMvcTestContextBootstrapper - Neither @ContextConfig
19:45:19.731 [main] DEBUG org.springframework.test.context.support.AbstractContextLoader - Did not detect default resource location for
19:45:19.731 [main] DEBUG org.springframework.test.context.support.AbstractContextLoader - Did not detect default resource location for
```

Añadimos nuestro .jar a la carpeta spring con el Dockerfile.



The screenshot shows a Windows File Explorer window with the address bar displaying the path: > Este equipo > Escritorio > spring > spring. The main area shows a list of files and folders. On the left, there is a sidebar with 'do' and 'os' folders. The main list has columns for 'Nombre', 'Fecha de modifica...', 'Tipo', and 'Tamaño'. There are two files listed: 'dockerfile' and 'SpringJar.jar'.

Nombre	Fecha de modifica...	Tipo	Tamaño
dockerfile	19/03/2019 17:49	Archivo	1 KB
SpringJar.jar	19/03/2019 18:01	Executable Jar File	17.293 KB

Creamos nuestro Dockerfile.

```
1 FROM openjdk:8-jdk-alpine
2 RUN mkdir -p /usr/app/
3 WORKDIR /usr/app
4 COPY SpringJar.jar /usr/app
5 EXPOSE 8091
6 CMD ["java", "-jar", "SpringJar.jar"]
```

Creamos nuestro docker-compose.yml.

```
1  version: '3.3'
2  |
3  services:
4  |   spring:
5  |       build: spring
6  |       ports:
7  |       - "8091:8091"
```

Primero debemos hacer un docker-compose build y después un docker-compose up.

```
PS C:\Users\Ayad\Desktop>spring> docker-compose build
Building spring
Step 1/6 : FROM openjdk:8-jdk-alpine
--> e0ea51023687
Step 2/6 : RUN mkdir -p /usr/app/
--> Using cache
--> 9aa8027f783f
Step 3/6 : WORKDIR /usr/app
--> Using cache
--> a016930c2678
Step 4/6 : COPY SpringJar.jar /usr/app
--> 7cccf78584dd
Step 5/6 : EXPOSE 8091
--> Running in cda9e9fc8e9c
Removing intermediate container cda9e9fc8e9c
--> ffb9413a6b86
Step 6/6 : CMD ["java", "-jar", "SpringJar.jar"]
--> Running in 3f18b64473a6
Removing intermediate container 3f18b64473a6
--> a63330948048

Successfully built a63330948048
Successfully tagged spring:spring:latest
PS C:\Users\Ayad\Desktop>spring> docker-compose up
Recreating spring_spring_1 ... done
Attaching to spring_spring_1
spring_1
spring_1
spring_1
spring_1
spring_1
spring_1
spring_1
spring_1
spring_1
spring_1
:: Spring Boot ::      (v2.0.5.RELEASE)

2019-03-19 17:25:07.308 INFO 1 --- [main] hello.Application           : Starting Application v0.1.0 on 573060a6153c with P
```

Podemos comprobar que funciona correctamente en el puerto 8091.

← → ↺ ⓘ localhost:8091

Get your greeting [here](#)

← → ↻ ⓘ localhost:8091/greeting

Hello, World!