

## Llançament amb docker run

Crearem una xarxa de docker amb el nom *runJoaquim* executarem la comanda ‘*docker network create runJoaquim*’

```
xescrnhuelc@JoaquimPPS:~/Examen_PPS$ docker network create runJoaquim
547392da7761e11afee858d11b46c8a3f4b04856fe86a04734c0ca7a058a5594
xescrnhuelc@JoaquimPPS:~/Examen_PPS$ docker network ls

```

NETWORK ID	NAME	DRIVER	SCOPE
2d371a4e0ebe	bridge	bridge	local
58b6f2a141cd	host	host	local
7f820f716b84	none	null	local
547392da7761	runJoaquim	bridge	local

```
xescrnhuelc@JoaquimPPS:~/Examen_PPS$
```

Posarem en segon pla del contenidor amb l’opció ‘-d’ el posarem el nom del contenidor «aplicacionjava», posarem el contenidor a la xarxa *runJoaquim* amb la opció ‘--network runJoaquim’, amb ‘--name=aplicacionjava’ i amb ‘-v ./sample.war:/usr/local/tomcat/webapps/sample.war’ per passar l’arxiu sample.war al directori ‘/usr/local/tomcat/webapps’ del contenidor com a volumen i al final posarem el nom de la imatge que gastara el qual será tomcat en la versio 9.0 (tomcat:9.0). Amb les opcions posades al terminal posarem al pricipi de la comanda ‘docker run’ i finalment executarem tota la comanda

```
xescrnhuelc@JoaquimPPS:~/Examen_PPS$ docker run -d --name=aplicacionjava --network runJoaquim -v ./sample.war:/usr/local/tomcat/webapps/sample.war tomcat:9.0
```

Una vegada llaçada ja el contenidor podem veure que efectivament el arxiu que voliem passar al contenidor esta on voliem deixar-ho i que funciona correctament

```
xescrnhuelc@JoaquimPPS:~/Examen_PPS$ docker run -d --name=aplicacionjava --network runJoaquim -v ./sample.war:/usr/local/tomcat/webapps/sample.war tomcat:9.0
af5cc9e7f0dbefa0d798a134686dec1cb1483ae991e55bc0957d2aefebdc0895
xescrnhuelc@JoaquimPPS:~/Examen_PPS$ docker exec aplicacionjava ls -l /usr/local/tomcat/webapps
total 12
drwxr-x--- 5 root root 4096 Feb  3 16:08 sample
-rwxrwx--- 1 1000 1000 4606 Feb  3 15:12 sample.war
xescrnhuelc@JoaquimPPS:~/Examen_PPS$ docker container ls

```

CONTAINER ID	IMAGE	COMMAND	CREATED	STATUS	PORTS	NAMES
af5cc9e7f0db	tomcat:9.0	"catalina.sh run"	About a minute ago	Up About a minute	8080/tcp	aplicacionjava

```
xescrnhuelc@JoaquimPPS:~/Examen_PPS$
```

Una vegada llançada el contenidor del tomcat crearem ara crearem el contenidor del proxy invers amb nginx

Per fer-ho executarem un altre ‘docker run’ llançant un altre contenidor en la mateixa xarxa, estaguent en segon pla i el posarem de nom ‘*proxyJoaquim*’, farém que es redirigisca el port 80 del contenidor al 80 del amfitrió amb ‘-p 80:80’ passarem com a volúm el fitxer default.conf cap al directori ‘/etc/nginx/conf.d’ del contenidor amb ‘-v ./default.conf:/etc/nginx/conf.d/default.conf’.

```
JoaquimPPS:~/Examen_PPS$ docker run -d --name=proxyJoaquim --network runJoaquim -v ./default.conf:/etc/nginx/conf.d/default.conf -p 80:80 nginx
```

Una vegada executada aquesta comanda comprovarem que es ha llançat correctament.

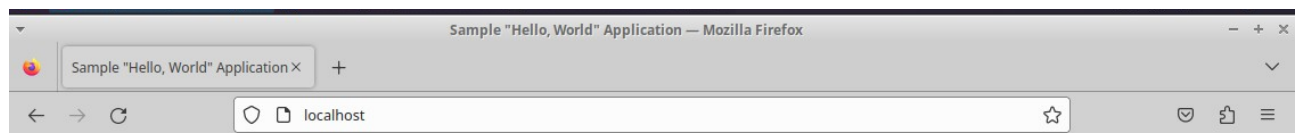
```
xescrnhuelc@JoaquimPPS:~/Examen_PPS$ docker run -d --name=proxyJoaquim --network runJoaquim -v ./default.conf:/etc/nginx/conf.d/default.conf -p 80:80 nginx
7353e3f235d0171121f57fa224c4a9e278f4ec687b9029dbc22d5dc9bddd14aa
xescrnhuelc@JoaquimPPS:~/Examen_PPS$ docker exec proxyJoaquim cat /etc/nginx/conf.d/default.conf
server {
    listen      80;
    listen     [::]:80;
    server_name localhost;

    location / {
        root    /usr/share/nginx/html;
        proxy_pass http://aplicacionjava:8080/sample/;
    }

    error_page 500 502 503 504 /50x.html;
    location = /50x.html {
        root    /usr/share/nginx/html;
    }
}

xescrnhuelc@JoaquimPPS:~/Examen_PPS$ docker container ls
CONTAINER ID   IMAGE     COMMAND                  CREATED        STATUS        PORTS                               NAMES
7353e3f235d0   nginx    "/docker-entrypoint..." 35 seconds ago Up 35 seconds 0.0.0.0:80->80/tcp, :::80->80/tcp   proxyJoaquim
af5cc9e7f0db   tomcat:9.0 "catalina.sh run"        6 minutes ago Up 6 minutes   8080/tcp                               aplicacionjava
xescrnhuelc@JoaquimPPS:~/Examen_PPS$
```

Comprobat que es ha llançat correctament farém la prova de foc posant al navegador 'localhost' i podrem amb aquesta captura veure que els dos contenidors junt amb les seues configuracions funcionen correctament



## Sample "Hello, World" Application

This is the home page for a sample application used to illustrate the source directory organization of a web application utilizing the principles outlined in the Application Developer's Guide.

To prove that they work, you can execute either of the following links:

- To a [JSP page](#).
- To a [servlet](#).

## Creació dels Dockerfile

Per fer possible la creació de contenidors amb Docker, crearem el arxiu Docker amb aquestes noves imatges els quals serán els següents:

Imatge contenidor 'aplicaciojava':

```
xescrhucl@JoaquimPPS:~/Examen_PPS$ cat Dockerfile
FROM tomcat:9.0
EXPOSE 8080
COPY ./sample.war /usr/local/tomcat/webapps/sample.war
xescrhucl@JoaquimPPS:~/Examen_PPS$
```

Amb el següent contingut al Docker file farem que la nova imatge es nombre 'tomfile' en la version 1 amb la comanda '*docker build . -t tomfile:1*'

```
xescrhucl@JoaquimPPS:~/Examen_PPS$ docker build . -t tomfile:1
[+] Building 0.1s (7/7) FINISHED
=> [internal] load build definition from Dockerfile                                0.0s
=> => transferring dockerfile: 120B                                              0.0s
=> [internal] load .dockerignore                                                 0.0s
=> => transferring context: 2B                                                  0.0s
=> [internal] load metadata for docker.io/library/tomcat:9.0                    0.0s
=> [internal] load build context                                                0.0s
=> => transferring context: 32B                                                  0.0s
=> [1/2] FROM docker.io/library/tomcat:9.0                                     0.0s
=> CACHED [2/2] COPY ./sample.war /usr/local/tomcat/webapps/sample.war         0.0s
=> => exporting to image                                                         0.0s
=> => exporting layers                                                           0.0s
=> => writing image sha256:8fadcaf9c99c3045808f5e77f4279cbb38308f6258477b81565409f7c9a2fe64 0.0s
=> => naming to docker.io/library/tomfile:1                                    0.0s
xescrhucl@JoaquimPPS:~/Examen_PPS$ docker run -d --name=aplicacionjava --network runJoaquim tomfile:1
730403c8f14043357fe382f0a60f6e8b03b716f6e41634a6788bc717ddc39e2b
xescrhucl@JoaquimPPS:~/Examen_PPS$ docker container ls
CONTAINER ID   IMAGE     COMMAND                  CREATED        STATUS        PORTS          NAMES
730403c8f140   tomfile:1 "catalina.sh run"       24 seconds ago Up 23 seconds 8080/tcp       aplicacionjava
xescrhucl@JoaquimPPS:~/Examen_PPS$ docker exec aplicacionjava ls -l /usr/local/tomcat/webapps/
total 12
drwxr-x--- 5 root root 4096 Feb 3 17:03 sample
-rwxrwx--- 1 root root 4606 Feb 3 15:12 sample.war
xescrhucl@JoaquimPPS:~/Examen_PPS$
```

Com podem veure una vegada creada la nova imatge simplement llançarem la comanda '*docker run -d --name=aplicacionjava --network runJoaquim tomfile:1*' fent que funcione i complisca les condicions del quan estabem fent els 'docker run'

Podem veure que fent les mateixes instruccions hem fet en el apartat del 'docker run' quan vam llançar-ho tindrem el mateix resultat que hem tractat anteriorment

Imatge contenidor 'proxyJoaquim' (Hi ha que modificar el Dockerfile per sigui de la següent forma):

```
xescruihuelc@JoaquimPPS:~/Examen_PPS$ cat Dockerfile
FROM nginx
EXPOSE 80
COPY ./default.conf /etc/nginx/conf.d/default.conf
xescruihuelc@JoaquimPPS:~/Examen_PPS$
```

Amb el següent contingut al Docker file farem que la nova imatge es nombre 'tomfile' en la version 1 amb la comanda 'docker build . -t redirjoaquim:2'

```
xescruihuelc@JoaquimPPS:~/Examen_PPS$ docker build . -t redirjoaquim:2
[+] Building 0.1s (7/7) FINISHED
=> [internal] load build definition from Dockerfile
=> => transferring dockerfile: 109B
=> [internal] load .dockerignore
=> => transferring context: 2B
=> [internal] load metadata for docker.io/library/nginx:latest
=> [internal] load build context
=> => transferring context: 34B
=> [1/2] FROM docker.io/library/nginx
=> CACHED [2/2] COPY ./default.conf /etc/nginx/conf.d/default.conf
=> exporting to image
=> exporting layers
=> writing image sha256:fac9aaf2869df768bfac49b3ea9962cb3f63a7926747432e3bb2746e1d8981c4
=> naming to docker.io/library/redirjoaquim:2
xescruihuelc@JoaquimPPS:~/Examen_PPS$ docker image ls
REPOSITORY          TAG             IMAGE ID        CREATED         SIZE
redirjoaquim        2               fac9aaf2869d   About a minute ago   142MB
tomfile             1               8fadcaf9c99c   20 minutes ago     476MB
tomcat              9.0            b07e16b11088   2 days ago         476MB
nginx               latest          a99a39d070bf   3 weeks ago        142MB
```

Com podem veure una vegada creada la nova imatge simplement llançarem la comanda 'docker run -d --name=filejoaquim --network runJoaquim -p 80:80 redirjoaquim:2' fent que funcione i complisca les condicions del quan estabem fent els 'docker run'

Podem veure que fent les mateixes instruccions hem fet en el apartat del 'docker run' quan vam llançar-ho tindrem el mateix resultat que hem tractat anteriorment

```
xescruihuelc@JoaquimPPS:~/Examen_PPS$ docker run -d --name=filejoaquim --network runJoaquim -p 80:80 redirjoaquim:2
6970bd97e7cfac3227db2db1ca9984862bade21f3f68868a9d92a0be6843f5f5
xescruihuelc@JoaquimPPS:~/Examen_PPS$ docker container ls
CONTAINER ID   IMAGE          COMMAND                  CREATED        STATUS        PORTS                               NAMES
6970bd97e7cf   redirjoaquim:2 "/docker-entrypoint..." 9 seconds ago   Up 8 seconds   0.0.0.0:80->80/tcp, :::80->80/tcp   filejoaquim
730403c8f140   tomfile:1     "catalina.sh run"        13 minutes ago Up 13 minutes   8080/tcp                             aplicacionjava
xescruihuelc@JoaquimPPS:~/Examen_PPS$ docker exec filejoaquim cat /etc/nginx/conf.d/default.conf
server {
    listen      80;
    listen     [::]:80;
    server_name localhost;

    location / {
        root    /usr/share/nginx/html;
        proxy_pass http://aplicacionjava:8080/sample/;
    }

    error_page 500 502 503 504 /50x.html;
    location = /50x.html {
        root    /usr/share/nginx/html;
    }
}
```

Una vegada llançades els contenidors comprovarem que l'aplicació funciona anant al navegador i posant a la barra de navegació 'localhost' i com podem veure funciona correctament



# Llançament amb docker-compose

Creant un docker-compose amb 'touch docker-compose.yaml' i modificant-ho pe a que tinga el següent contingut:

```
xescrnhuelc@JoaquimPPS:~/Examen_PPS$ cat docker-compose.yaml
#Fet per Joaquim Escrinhuela Cholvi
version: '3.9'

services:
  tomcat:
    image: tomcat:9.0
    container_name: aplicacionjava
    restart: always
    volumes:
      - /home/xescrnhuelc/Examen_PPS/sample.war:/usr/local/tomcat/webapps/sample.war
    networks:
      - runJoaquim

  proxy:
    image: nginx
    container_name: composejoaquim
    restart: always
    ports:
      - "80:80"
    depends_on:
      - tomcat
    volumes:
      - /home/xescrnhuelc/Examen_PPS/default.conf:/etc/nginx/conf.d/default.conf
    networks:
      - runJoaquim

networks:
  runJoaquim:
    name: runJoaquim
    driver: bridge
xescrnhuelc@JoaquimPPS:~/Examen_PPS$
```

Llançem el *docker-compose* amb la comanda 'docker-compose up -d' i després amb un 'docker container ls' veurem com es han creat els contenidors correctament amb les regles establides

```
xescrnhuelc@JoaquimPPS:~/Examen_PPS$ docker-compose up -d
Creating network "runJoaquim" with driver "bridge"
Creating aplicacionjava ... done
Creating composejoaquim ... done
xescrnhuelc@JoaquimPPS:~/Examen_PPS$ docker container ls
CONTAINER ID   IMAGE     COMMAND                  CREATED        STATUS        PORTS                               NAMES
f2003fd1489d   nginx     "/docker-entrypoint..." 5 seconds ago  Up 4 seconds  0.0.0.0:80->80/tcp, :::80->80/tcp  composejoaquim
23f6bce98c5    tomcat:9.0 "catalina.sh run"        5 seconds ago  Up 5 seconds  8080/tcp                           aplicacionjava
xescrnhuelc@JoaquimPPS:~/Examen_PPS$ docker exec composejoaquim cat /etc/nginx/conf.d/default.conf
server {
    listen      80;
    listen     [::]:80;
    server_name localhost;

    location / {
        root    /usr/share/nginx/html;
        proxy_pass http://aplicacionjava:8080/sample/;
    }

    error_page 500 502 503 504 /50x.html;
    location = /50x.html {
        root    /usr/share/nginx/html;
    }
}
xescrnhuelc@JoaquimPPS:~/Examen_PPS$ docker exec aplicacionjava ls -l /usr/local/tomcat/webapps/
total 12
drwxr-x-- 5 root root 4096 Feb  3 17:59 sample
-rwxrwx-- 1 1000 1000 4606 Feb  3 15:12 sample.war
xescrnhuelc@JoaquimPPS:~/Examen_PPS$
```


Si fem la prova al navegador veurem que funciona correctament

```
xescrnhuelc@JoaquimPPS:~/Examen_PPS$ docker-compose up -d
Creating network "runJoaquim" with driver "bridge"
Creating aplicacionjava ... done
Creating composejoaquim ... done
xescrnhuelc@JoaquimPPS:~/Examen_PPS$ docker container ls
CONTAINER ID   IMAGE     COMMAND                  CREATED        STATUS        PORTS                               NAMES
f2003fd1489d   nginx     "/docker-entrypoint..." 5 seconds ago  Up 4 seconds  0.0.0.0:80->80/tcp, :::80->80/tcp  composejoaquim
23f6bce98c5    tomcat:9.0 "catalina.sh run"        5 seconds ago  Up 5 seconds  8080/tcp                           aplicacionjava
xescrnhuelc@JoaquimPPS:~/Examen_PPS$
```

Sample "Hello, World" Application — Mozilla Firefox

Sample "Hello, World" Application x

localhost



## Sample "Hello, World" Application

This is the home page for a sample application used to illustrate the source directory organization of a web application utilizing the principles outlined in the Application Developer's Guide.

To prove that they work, you can execute either of the following links:

- To a [JSP page](#).
- To a [servlet](#).