# A qui va dirigit

Aquest how-to va dirigit a tots aquells desenvolupadors/arquitectes que vulguin desplegar una aplicació Canigó 3.1 utilitzant docker.

# Versió de Canigó

Els passos descrits en aquest document apliquen a la versió 3.1.x del Framework Canigó.

# Introducció

En aquest HowTo s’explica com dockeritzar una aplicació Canigó 3.1 REST. Per a fer-ho desplegarem l’aplicació demo que genera el plugin de Canigó en un tomcat amb el contingut estàtic servit per un apache, i una base de dades MySQL.

# Separar Codi Estàtic i Dinàmic

Primer de tot configurem l’aplicació per a que generi separat el codi estàtic i dinàmic.

A **java/resources/assemble/dynamic.xml** indiquem que el format de sortida sigui un directori

<assembly>

<id>dynamic</id>

<formats>

<format>dir</format>

</formats>

<includeBaseDirectory>false</includeBaseDirectory>

<fileSets>

<fileSet>

<directory>target/howtoJuliol</directory>

<outputDirectory>/</outputDirectory>

<includes>

<include>WEB-INF/\*\*/\*.\*</include>

<include>\*\*/\*.jsf</include>

<include>\*\*/\*.xhtml</include>

<include>\*\*/\*.cfg</include>

</includes>

<excludes>

<exclude>\*\*/\*.html</exclude>

<exclude>\*\*/\*.htm</exclude>

<exclude>css/\*\*/\*</exclude>

<exclude>images/\*\*/\*</exclude>

<exclude>scripts/\*\*/\*</exclude>

<exclude>config/\*\*/\*</exclude>

<exclude>js/\*\*/\*</exclude>

</excludes>

</fileSet>

</fileSets>

</assembly>

I el mateix format de sortida per a la part estàtic a **java/resources/assemble/static.xml**:

*<assembly>*

*<id>static</id>*

*<formats>*

*<format>dir</format>*

*</formats>*

*<includeBaseDirectory>false</includeBaseDirectory>*

*<fileSets>*

*<fileSet>*

*<directory>target/howtoJuliol</directory>*

*<outputDirectory>/</outputDirectory>*

*<includes>*

*<include>\*\*/\*.html</include>*

*<include>\*\*/\*.htm</include>*

*<include>css/\*\*/\*</include>*

*<include>js/\*\*/\*</include>*

*<include>images/\*\*/\*</include>*

*<include>scripts/\*\*/\*</include>*

*</includes>*

*</fileSet>*

*</fileSets>*

*</assembly>*

D’aquesta manera quan compilem l’aplicació es generaran dos directories: target/howtoJuliol-dynamic i target/howtoJuliol-static que s’utilitzaran al docker-compose per a pasar-li la part estàtica al Apache i la part dinàmica al Tomcat.

# Configuració Base de Dades a l’aplicació

Al dockeritzar una aplicació i la seva Base de Dades utilitzarem un link per a que l’aplicació tingui visibilitat sobre la Base de Dades, al fitxer **jdbc.properties** al paràmetre **jdbc.url** hem d’indicar el nom del link en comptes de la IP

*\*.jdbc.url=jdbc:mysql://mybbdd:3306/test*

En aquest howto utilitzem “*mybbdd*” com el nom del link (més endavant ho veurem al docker-compose)

# Configuració Imatge Apache

docker/httpd/Dockerfile

*FROM gencatcloud/httpd:2.2*

*COPY httpd.conf /usr/local/apache2/conf/*

*CMD ["/entrypoint.sh"]*

Utilitzem la imatge gencatcloud/httpd:2.2 com a imatge base del nostre Apache i sobreescrivim el fitxer httpd.conf per a afegir els proxyPass:

*ProxyPass /howtoJuliol/api http://howto:8080/howtoJuliol/api*

*ProxyPassReverse /howtoJuliol/api http://howto:8080/howtoJuliol/api*

De la mateixa manera que hem utilitzat el link “*mybbdd*” per a bbdd, en aquesta ocasió utilitzem el link “howto” per a relacionar el contenidor Apache amb el contenidor de l’aplicació

# Configuració Imatge MySQL

docker/bbdd/Dockerfile

*FROM gencatcloud/mysql:5.7*

*COPY script.sql /docker-entrypoint-initdb.d/*

*CMD ["/entrypoint.sh"]*

Utilitzem la imatge gencatcloud/mysql:5.7 com a imatge base del nostre MySQL i li passem el fitxer script.sql amb els scripts inicials de base de dades. (Els fitxers copiats a /docker-entrypoint-initdb.d només s’executen la primera vegada que s’arrenca el contenidor MySQL)

*CREATE TABLE `equipaments` (*

*`id` bigint(20) NOT NULL,*

*`nom` varchar(200) NOT NULL,*

*`municipi` varchar(200),*

*PRIMARY KEY (`id`)*

*) ENGINE=InnoDB DEFAULT CHARSET=utf8;*

*insert into equipaments (id, nom, municipi) values (1, 'estació autobusos', 'Ripoll');*

*insert into equipaments (id, nom, municipi) values (3, 'Llar d''infants Món Petit', 'l''Escala');*

*insert into equipaments (id, nom, municipi) values (4, 'Jutjat de Pau', 'Sant Esteve de Palautordera');*

*insert into equipaments (id, nom, municipi) values (2, 'centre obert Alba', 'Cambrils');*

*insert into equipaments (id, nom, municipi) values (5, 'LLI privada Quitxalla', 'Mollet del Vallès');*

*insert into equipaments (id, nom, municipi) values (6, 'PISTA POLIESPORTIVA MUNICIPAL', 'Aguilar de Segarra');*

*insert into equipaments (id, nom, municipi) values (7, 'La Salle Manlleu', 'Manlleu');*

*insert into equipaments (id, nom, municipi) values (8, 'SERVEI COMARCAL DE JOVENTUT DE LA TERRA ALTA', 'Gandesa');*

*insert into equipaments (id, nom, municipi) values (9, 'Residència d''Avis de Capellades Fundació Consorts Guasch', 'Capellades');*

*insert into equipaments (id, nom, municipi) values (10, 'PISTA DE BASQUET AL CARRER (PL DR. BONET)', 'Vilafranca del Penedès');*

*insert into equipaments (id, nom, municipi) values (11, 'Punt TIC - Biblioteca Municipal de Capellades', 'Capellades');*

*insert into equipaments (id, nom, municipi) values (12, 'Parc de Recerca Biomèdica de Barcelona', 'Barcelona');*

*insert into equipaments (id, nom, municipi) values (13, 'Instal·lació Esportiva CEIP BLANCA DE VILALLONGA-ZER ALT SEGRIÀ', 'Portella la');*

*insert into equipaments (id, nom, municipi) values (14, 'INDOORKARTING', 'INDOORKARTING');*

*insert into equipaments (id, nom, municipi) values (15, 'Biblioteca de l''Hospital de Tortosa Verge de La Cinta', 'Tortosa');*

De la mateixa manera que hem utilitzat el link “mysql” per a bbdd, en aquesta ocasió utilitzem el link “howto” per a relacionar el contenidor Apache amb el contenidor de l’aplicació

# Configuració Imatge Tomcat

Per a poder utilitzar una base de dades MySQL hem d’afegir la llibreria al tomcat, en aquest howto també hem configurat el fitxer setenv.sh per a habilitar el mode debug

docker/app/Dockerfile

*FROM gencatcloud/tomcat:7*

*COPY setenv.sh /opt/tomcat/bin/*

*COPY mysql-connector-java-5.1.8-bin.jar /opt/tomcat/lib/*

*CMD ["/entrypoint.sh"]*

El contingut del fitxer setenv.sh

*CATALINA\_OPTS="-Xdebug -Xrunjdwp:transport=dt\_socket,address=8000,server=y,suspend=n"*

# Docker-Compose

Una vegada ja tenim preparats els 3 contenidors (base dades, apache i tomcat) utilitzem un fitxer docker-compose per a relacionar-los, afegir-hi volums i posar-los en marxa

docker/docker-compose.yml

*httpd:*

*build: ./httpd/*

*links:*

*- howto:howto*

*ports:*

*- 80:80*

*volumes:*

*- /home/canigo/howtoJuliol-static:/usr/local/apache2/htdocs/howtoJuliol*

*db:*

*build: ./bbdd/*

*ports:*

*- 3306:3306*

*environment:*

*MYSQL\_USER : user*

*MYSQL\_PASSWORD : password*

*MYSQL\_ROOT\_PASSWORD: root*

*MYSQL\_DATABASE: test*

*volumes:*

*- /home/canigo/mysql-datadir:/var/lib/mysql*

*howto:*

*build: ./app/*

*links:*

*- db:mybbdd*

*volumes:*

*- /home/canigo/howtoJuliol-dynamic:/opt/tomcat/webapps/howtoJuliol*

*ports:*

*- 8080:8080*

*- 8000:8000*

En aquest fitxer tenim els 3 serveis diferenciats:

**httpd**, és el contenidor Apache. Aquí es configura el link cap al contenidor tomcat i s’exposa el port 80. A /home/canigo/howtoJuliol-dynamic s’ha deixat el directori que genera maven al compilar per a la part estàtica.

**db**, és el contenidor de la base de dades. S’exposa el port 3306 i s’indica mitjançant variables d’entorn que al crear-se el contenidor generi un usuari (user) i una base dades (test). Utilitzem un volum també en aquest contenidor per a mantenir les dades de base de dades després d’aturar el contenidor.

**howto,** és el contenidor tomcat. Aquí es configura el link cap a la base de dades, exposem els ports 8080 i 8000 i utilitzem també un volum per a passar la part dinàmica al tomcat.

Per a posar en marxa els contenidors, des de la carpeta docker (on es troba el fitxer docker-compose.yml i les carpetes httpd, bbd i app:

**docker-compose –f docker-compose.yml up –d**