Come creare con Docker un ambiente di lavoro Android / Cordova e fare deploy su device

> *Come App Developer, mi sono spesso trovato alle prese con gli update di Cordova, Android SDK, e compagnia bella. Ho deciso di affidarmi a Docker per non avere più problemi.*

Da quando ho scoperto Docker, mi sento un bambino alle prese con un giocattolo tutto nuovo.

Lo userei per qualsiasi cosa, vorrei un container anche per fare il caffè!

Scherzi a parte, i vantaggi di Docker sono assoluti, e se ancora non avete iniziato a usarlo, beh dovreste, assolutamente.

La comodità di portarsi dietro dei Dockerfiles, anzichè passare giornate a installare pacchetti, configurazioni, litigare con versioni e rischiare di avere problemi di compatibilità tra progetti, per non parlare di poter poi mettere il tutto su un qualsiasi server, e avere la stessa identica configurazione, è assolutamente senza pari.

Oggi, dopo l'ennesimo aggiornamento degli Android SDK, e relativi malditesta per far funzionare di nuovo tutti gli ambienti, mi sono messo in testa di eliminare dalla mia macchina tutte le configurazioni Android necessarie per lavorare con Cordova (e di conseguenza con Ionic, Meteor) e React-Native.

Per cui, spazio ai container, montati con Android SDK, e Cordova o React-Native, a seconda della necessità.

In realtà, per fare quanto stiamo per vedere, sarà necessario appoggiarsi a VirtualBox, e creare una VM su cui fare girare i container.

Il perchè è presto spiegato, e ci ho messo diverso tempo per capirlo (e devo ringraziare un paio di persone che mi hanno aiutato a capire il funzionamento).

La necessità di chi, come me, sviluppa App, è di fare deploy e debug direttamente su device. Per farlo, ovviamente, i device devono essere collegati via USB alla macchina su cui si lavora.

Purtroppo per noi, nel caso di Docker non è possibile rendere disponibili le porte USB direttamente ai container.

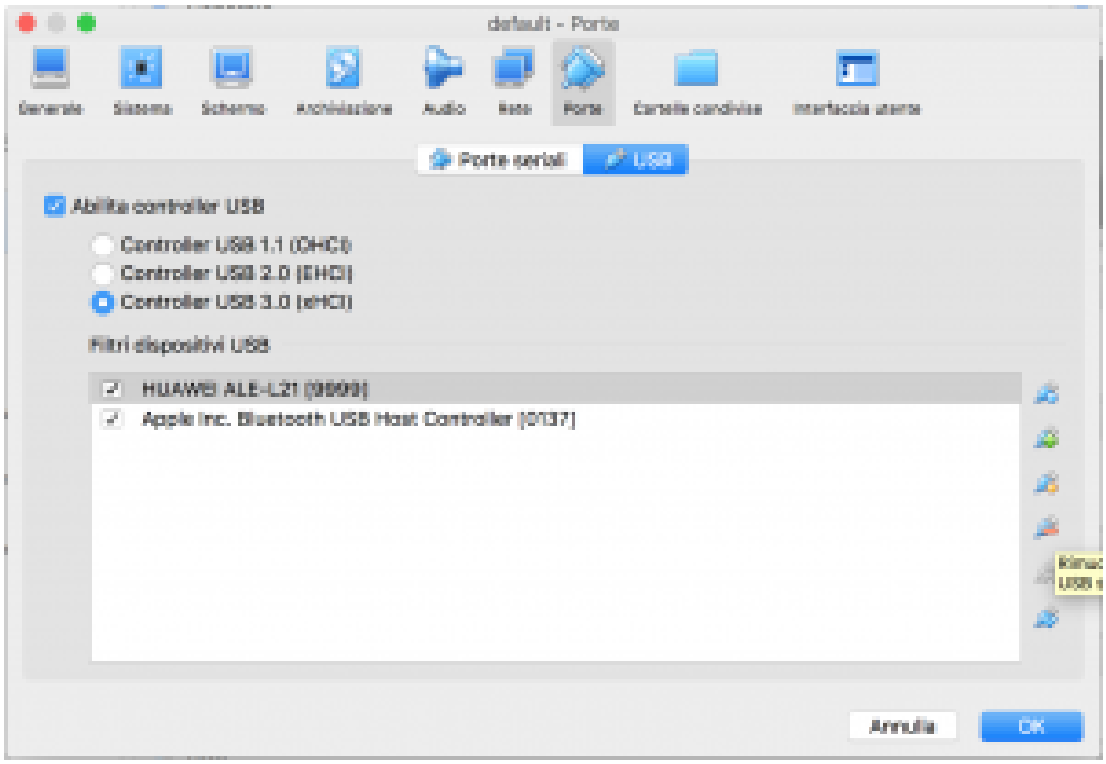
VirtualBox, però, consente di mappare le porte USB della propria macchina e di renderle disponibili alle VM installate. Ecco quindi che in nostro soccorso corre docker-toolbox, e in particolare docker-machine, ovvero una Virtual Machine su cui installare i nostri container.

Il tutto si configura con i passi che vi sto per elencare.

Procedimento

- Prima di tutto, verificate di disporre di tutti i seguenti tool:
- Installate *VirtualBox;*
- Installate *docker;*
- Installate *docker-toolbox*, che installerà una VM, basata su VirtualBox, raggiungibile da terminale tramite i comandi ***docker-machine;***

Una volta installata la VM, aprite VirtualBox, andate nelle impostazioni della suddetta VM (che dovrebbe chiamarsi "*default*", solitamente), Porte -> USB, ed abilitate il controller.



Dopodichè, da terminale, accedete alla vostra macchina virtuale, digitando:

docker-machine ssh

A questo punto siete dentro alla VM, e avete la possibilità di installare e far girare i vostri container.

Per lo sviluppo Android, sono partito da uno dei tanti disponibili, come ad esempio https://hub.docker.com/r/snowdream/docker-android , con cui ho poi compilato i miei docker-compose (a seconda che mi serva Cordova, o React-Native o altre librerie).

Installato il container, dovete fare in modo che le porte USB siano condivise tra la VM e il container su cui gira l'SDK Android.

Per farlo, digitate il seguente comando:

docker run -t -i – privileged -v /dev/bus/usb:/dev/bus/usb snowdream/android:latest

Ovviamente, sostituite *"snowdream/android:latest"* con il nome

dell’immagine che avete installato.

Fatto questo, dovreste essere dentro al container, e connettendo il vostro device via USB, dovreste vederlo digitando:

adb devices

Se questo avviene, siete a cavallo.

Da questo momento in poi, qualsiasi istruzione del tipo:

cordova run android – device

verrà eseguita come se foste in locale, e quindi facendo il *deploy* dell’app sul vostro dispositivo.

> *Evviva Docker, lunga vita a Docker!*

https://italiancoders.it/creare-docker-un-ambiente-lavoro-android-cordova-deploy-device/