Docker

המטרה - להעביר קוד של תוכנה ממקום למקום, ושהיא תעבוד out of the box.

בלי שנצטרך לקנפג או לעבוד קשה על כל מיני התקנות וכיו"ב.

באמצעות דוקר, אנו יכולים לארוז תוכנה שפיתחנו בתוך image. את ה-image הזה ניתן להריץ בתוך container על כל מחשב בעולם שמותקן עליו דוקר.

לפיכך, מרגע שאנחנו משתמשים בדוקר, כל מה שאנחנו צריכים בשביל לפבלש תוכנה (להעלות אותה לפרודקשן) זה מחשב שמותקן עליו דוקר. כדי שהוא יוכל לקבל את התוכנה שלנו ולהריץ אותה.

איך זה עובד?



בבסיס הדוקר עומדים כאמור image-ים שונים. Image הוא אוסף של שכבות בסדר מסויים. כל שכבה היא למעשה תוצאה של הרצת פקודה על המחשב.

למשל, בציור למעלה, השכבה של npm היא תוצאה של התקנת npm על מחשב שיש בו OS+Shell כמתואר למעלה.

אז למעשה image-ים הם אוסף של שכבות בסדר מסויים.

כדי להקל עלינו, לא צריך לייצר image חדש לגמרי מאפס, אלא ניתן לקחת image קיים, ולהמשיך ממנו.

**יצירת Image**

כדי ליצור image אנו נכתוב את תהליך יצירת השכבות בקובץ שנקרא Dockerfile.

כדי לבנות את האימג' מתוך ה-Dockerfile נריץ:

docker build --tag sample-rest-api .

Docker build –tag {name your image here} .

המשמעות של ה"." היא בלינוקסית: כאן. כלומר אנו מבקשים לבנות אימג' בשם הזה והזה לפי ה-Dockerfile שנמצא "כאן". כלומר בתיקייה הנוכחית.

**יצירת Container**



בדיקת אילו קונטיינרים נמצאים כרגע על המכונה שלי:

Docker ps -a

יראה לי את כולם, גם אלו שרצים וגם כבויים

Docker ps

יראה לי רק רצים

Docker images

מראה images שנמצאים על המחשב שלי

כדי ליצור קונטיינר נריץ:

Docker create –name [name the container here] imageName

כדי להתחיל קונטיינר קיים הפקודה היא:

Docker strat [id or name of container]

כדי לעצור קונטיינר רץ:

Docker stop [id or name of container]

כדי למחוק קונטיינר:

Docker rm [id or name of container]

כדי למחוק image:

Docker rmi [idf or name of image]

כדי ליצור קונטיינר מאימג' מסויים וגם למפות פורט:

Docker create –name [container name] -p 8090:8080 [image name]

סדר המיפוי:

Host port : container port (8090:8080)

**Volumes**

באופן כללי ווליום הוא שם כללי לדיסק, כלומר מקום שאפשר לשמור בו נתונים היכולים להישמר לנצח.

הדרך לעבוד עם ווליום בדוקר היא באמצעות -v

ניתן למפות ווליום מקומי לווליום בתוך הקונטיינר על ידי ציון מפורש של שמות התיקיות בהוסט ובקונטיינר. לדוגמא:

docker create --name fs-example-container -p 8085:8080 -v C:\local-volume:/db fs-example

עדיף עם זאת, להמנע מכך, ולהשתמש בווליומים "וירטואלים" המנוהלים על ידי תוכנת דוקר ומאפשרים גישה למידע הנוצר בעת פעילות הקונטיינר, מבלי לציין מפורשות באיזו תיקיה בהוסט יישמרו הנתונים:

docker create --name fs-example-container -p 8085:8080 -v local-volume:/db fs-example

**Image naming convention**

לא נקרא לאימג'ים שלנו בשמות פשוטים כי לא ניתן לקיים מעקב גרסאות כך, וגם לא נרצה למחוק כל פעם אימג/ לפני שנבנה אותו מחדש. אפשר פשוט לקרוא להם בפורמט הבא:

**owner/name:version**

למשל:

shaharsol/fs-example:1.0

לדחיפת אימג' לדוקרהאב נשתמש בפקודה:

Docker push username/imagename:version

E.g.

Docker push shaharsol/sample-rest-api:1.1

**Docker run.**

פקודת קיצור דרך שעושה 3 דברים:

1. מחפשת את האימג' אותו אנו רוצים להריץ על המחשב המקומי, ואם לא קיימת אז מורידה מדוקרהאב
2. מריצה create
3. מריצה start

חשוב לזכור את ה-d כדי שירוץ ב-detached מהטרמינל, קרי שלא יתפוס את הטרמינל וישחרר לנו אותו אחרי שהוא מוריד את האימג', יוצר את הקונטיינר ומריץ אותו.

פקודת docker run לדוגמא:

docker run --name sample-rest-api-container -p 8090:8080 -d shaharsol/sample-rest-api:1.1