

몽고DB 이미지를 풀해서 API 컨테이너가 접근할 수 있는 데이터베이스 컨테이너를 생성했다. 도커 컨테이너는 별도의 네트워크 설정을 사용하며 기본 설정에서는 localhost 주소로 연결하는 것을 허용하지 않는다.

```
youngestdev@Abduls-MacBook-Air:~  
+ - docker pull mongo  
Using default tag: latest  
latest: Pulling from library/mongo  
d4ba87bb7858: Pull complete  
cd0512bd0ae7: Pull complete  
51571d358e42: Pull complete  
aa226bdd9830: Pull complete  
8c0e3e44d3ac: Pull complete  
e106719865e3: Pull complete  
600ad43e7428: Pull complete  
0e1222e945b3: Pull complete  
8eb8d81c70ea: Pull complete  
Digest: sha256:50d8918de7b076feceb9ba1ee264afd5f67fb4baaff07949f3b9de92cdca79c2  
Status: Downloaded newer image for mongo:latest  
docker.io/library/mongo:latest  
+ -  
+ - docker pull mongo  
Using default tag: latest  
latest: Pulling from library/mongo  
Digest: sha256:50d8918de7b076feceb9ba1ee264afd5f67fb4baaff07949f3b9de92cdca79c2  
Status: Image is up to date for mongo:latest  
docker.io/library/mongo:latest  
+ -
```

그림 9-2 몽고DB 이미지 풀하기



docker pull

docker pull 명령은 레지스트리에서 이미지를 다운로드한다. 별도로 지정하지 않으면 도커 허브 레지스트리에서 이미지를 다운로드한다.

로컬에 애플리케이션 배포

이번에는 애플리케이션 배포를 담당하는 파일을 만들어보자. 여기서 사용할 도커 구성 파일(docker-compose.yml)은 API 서비스와 몽고DB 데이터베이스 서비스로 구성된다. 먼저 루트 디렉터리에 다음과 같이 구성 파일을 만든다.