1. Docker 실행 명령어를 일일이 입력하기가 복잡해서
   1)Nginx 실행하기
      $ docker run –it nginx
   2)Nginx Container 실행 + Host 8080 Port 연결하기
      $ docker run -it –p 8080:80 nginx
   3)Nginx Container 실행 + Host 8080 Port 연결 + Container 종료시 자동 삭제
      $ docker run -it –p 8080:80 --rm nginx
   4)Nginx 컨테이너 실행 + Host 8080 Port 연결 + Container 종료시 자동 삭제 + Host의 Directory를 Container 안에서 링크하기
      $ vi index.html
      <h1>Hello, Docker Compose</h1>

      $ docker run -it –p 8080:80 --rm –v ${PWD}:/usr/share/nginx/html/ nginx

2. 컨테이너끼리 연결하기 편해서 --1) 후 바로 --3)실행
   1)준비 : django-sample 이미지를 빌드
      $ git clone https://github.com/raccoonyy/django-sample-for-docker-compose.git django-sample
      $ cd django-sample
      $ docker build -t django-sample .

   2)django 컨테이너 실행 + postgres 컨테이너 실행
      $ docker run --rm -d --name django -p 8000:8000 django-sample

      $ docker ps -a

      -Web Browser를 열고 http://ip:8000
         --django 잘 실행되고 있음을 확인

      $ docker run --rm -d --name postgres -e POSTGRES_DB=djangosample \
      > -e POSTGRES_USER=sampleuser \
      > -e POSTGRES_PASSWORD=samplesecret \
      > postgres

      -Web Browser를 열고 http://ip:8000
         --그냥 django만 잘 실행되고 있음.

   3)postgres 컨테이너 실행 + django 컨테이너 실행 + 서로 연결하기
      $ docker run --rm -d --name postgres -e POSTGRES_DB=djangosample \
      > -e POSTGRES_USER=sampleuser \
      > -e POSTGRES_PASSWORD=samplesecret \
      > postgres

      $ docker run -d --rm -p 8000:8000 -e DJANGO_DB_HOST=db \
      > --link postgres:db \
      > django-sample

3. 특정 컨테이너끼리만 통신할 수 있는 가상 네트워크 환경을 편리하게 관리하고 싶어서
   1)postgres 컨테이너 실행 + django1 컨테이너 연결
      $ docker run --rm -d --name postgres \
      > -e POSTGRES_DB=djangosample \
      > -e POSTGRES_USER=sampleuser \
      > -e POSTGRES_PASSWORD=samplesecret \
      > postgres

      $ docker run -d --rm --name django1 \
      > -p 8000:8000 \
      > -e DJANGO_DB_HOST=db \
      > --link postgres:db \
      > django-sample

   2)postgres 컨테이너는 호스트의 다른 컨테이너들이 모두 접근할 수 있음
      $ docker run -d --rm --name django2 \
      > -p 8001:8000 \
      > -e DJANGO_DB_HOST=db \
      > --link postgres:db \

```
   > django-sample

3)postgres 컨테이너 + django1 컨테이너만 통신할 수 있는 가상 네트워크 만들기
   -도커 네트워크 살펴보기
      $docker network ls

   -도커 네트워크 생성하기
      $ docker network create --driver bridge web-service
      $ docker network ls

   -컨테이너 실행하기
      $ docker run --rm -d --name postgres \
      > --network web-service \
      > -e POSTGRES_DB=djangosample \
      > -e POSTGRES_USER=sampleuser \
      > -e POSTGRES_PASSWORD=samplesecret \
      > postgres

      $ docker run -d --rm --name django1 \
      > --network web-service \
      > -p 8000:8000 \
      > -e DJANGO_DB_HOST=db \
      > --link postgres:db \
      > django-sample

      $ docker run -d --rm --name django2 \
      > -p 8001:8000 \
      > -e DJANGO_DB_HOST=db \
      > --link postgres:db \
      > django-sample


4. 이 모든 것을 간단한 명령어로 관리하고 싶어서
   1)실행 명령어와 종료 명령어
      $ docker network create --driver bridge web-service

      $ docker run --rm -d --name postgres \
      > --network web-service \
      > -p 5432:5432 \
      > -e POSTGRES_DB=djangosample \
      > -e POSTGRES_USER=sampleuser \
      > -e POSTGRES_PASSWORD=samplesecret \
      > postgres

      $ docker run -d --rm --name django1 \
      > --network web-service \
      > -p 8000:8000 \
      > -e DJANGO_DB_HOST=db \
      > --link postgres:db \
      > django-sample

      $ docker kill django1 postgres
      $ docker network rm web-service

   2)docker-compose.yml
      version: '3'

      volumes:
         postgres_data: {}

      services:
         db:
            image: postgres
            volumes:
              - postgres_data:/var/lib/postgres/data
            environment:
              - POSTGRES_DB=djangosample
```

```
      - POSTGRES_USER=sampleuser
      - POSTGRES_PASSWORD=samplesecret
  django:
    build:
      context: .
      dockerfile: ./compose/django/Dockerfile-dev
      volumes:
        - ./:/app/
      command: ["./manage.py", "runserver", "0:8000"]
      environment:
        - DJANGO_DB_HOST=db
      depends_on:
        - db
      restart: always
      ports:
        - 8000:8000
```

3)도커 컴포즈로 실행하고 종료하기

-모든 docker process 중지

-모든 docker images 삭제

-django-sample folder 삭제

```
$ docker-compose up -d
$ docker-compose down
```

-웹 브라우저로 확인할 것