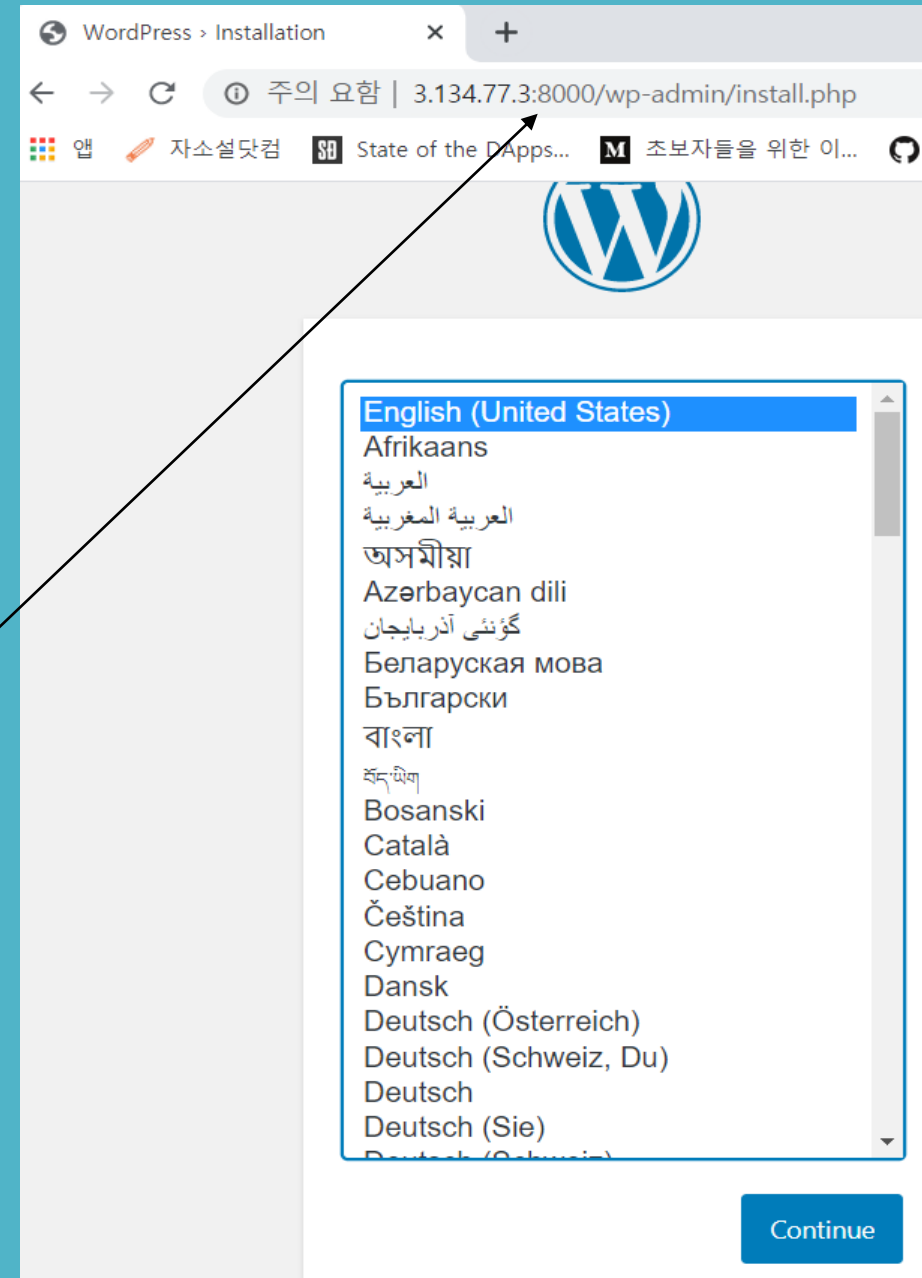


Docker compose를 이용하여 Container간 JSON data 주고받기 실습

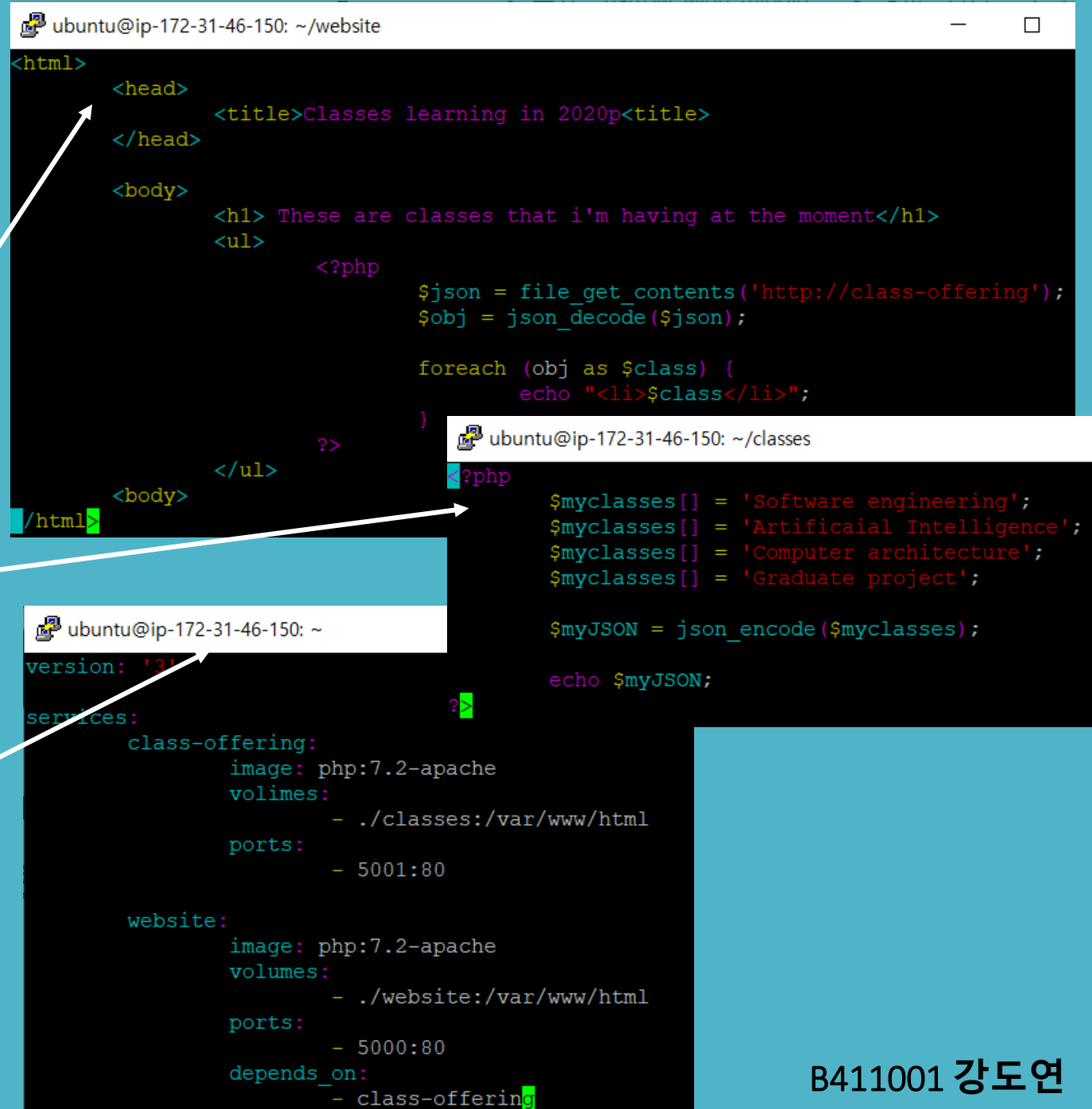
우선 전시간에 이어서,
docker-compose를 이용하여,
port 8000번으로 wordpress를 띄
워보았다.



본격적인 실습을 위해 우선,

각각 독립적인 classes, website director에 index.php 파일을 생성하고, 제일 상단 directory(home)에 실행을 위한 docker-compose.yml 파일을 생성하였다. 그 후, YAML 파일을 실행시켜보았다.

```
*** System restart required ***
Last login: Fri Apr 24 05:48:43 2020 from 222.120.195.64
ubuntu@ip-172-31-46-150:~$ sudo docker rm -f $(sudo docker ps -a -q)
42f085775c35
83860c3352e6
ubuntu@ip-172-31-46-150:~$ mkdir classes
ubuntu@ip-172-31-46-150:~$ cd classes
ubuntu@ip-172-31-46-150:~/classes$ vim index.php
ubuntu@ip-172-31-46-150:~/classes$ cd ..
ubuntu@ip-172-31-46-150:~$ mkdir website
ubuntu@ip-172-31-46-150:~$ cd website
ubuntu@ip-172-31-46-150:~/website$ vim index.php
ubuntu@ip-172-31-46-150:~/website$ vim index.php
ubuntu@ip-172-31-46-150:~/website$ cd ..
ubuntu@ip-172-31-46-150:~$ cd classes
ubuntu@ip-172-31-46-150:~/classes$ ls
index.php
ubuntu@ip-172-31-46-150:~/classes$ vim index.php
ubuntu@ip-172-31-46-150:~/classes$ cd ..
ubuntu@ip-172-31-46-150:~$ vim docker-compose.yml
ubuntu@ip-172-31-46-150:~$ sudo docker-compose up -d
ERROR: The Compose file './docker-compose.yml' is invalid because:
Unsupported config option for services.class-offering: 'volimes'
ubuntu@ip-172-31-46-150:~$ vim docker-compose.yml
ubuntu@ip-172-31-46-150:~$ sudo docker-compose up -d
```



```
ubuntu@ip-172-31-46-150: ~/website
<html>
  <head>
    <title>Classes learning in 2020p</title>
  </head>
  <body>
    <h1> These are classes that i'm having at the moment</h1>
    <ul>
      <?php
        $json = file_get_contents('http://class-offering');
        $obj = json_decode($json);

        foreach ($obj as $class) {
          echo "<li>$class</li>";
        }
      <?>
    </ul>
  </body>
</html>
```

```
ubuntu@ip-172-31-46-150: ~/classes
<?php
  $myclasses[] = 'Software engineering';
  $myclasses[] = 'Artificaiial Intelligence';
  $myclasses[] = 'Computer architecture';
  $myclasses[] = 'Graduate project';

  $myJSON = json_encode($myclasses);

  echo $myJSON;
```

```
ubuntu@ip-172-31-46-150: ~
version: '3'

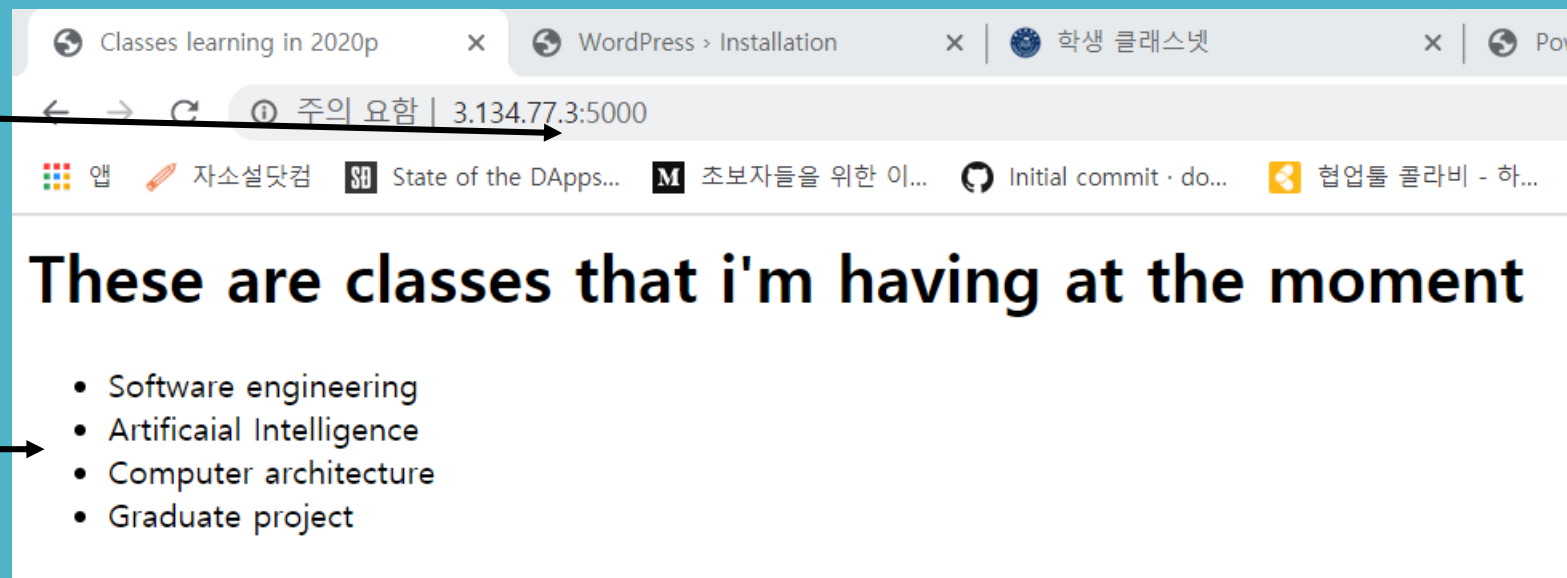
services:
  class-offering:
    image: php:7.2-apache
    volumes:
      - ./classes:/var/www/html
    ports:
      - 5001:80

  website:
    image: php:7.2-apache
    volumes:
      - ./website:/var/www/html
    ports:
      - 5000:80
    depends_on:
      - class-offering
```

Docker-compose 내용에 따라서 실행이 완료되었고, 그 결과, 독립적인 apache가 2개 작동하기 시작했다.

```
Creating ubuntu_class-offering_1 ... done
Creating ubuntu_website_1 ... done
ubuntu@ip-172-31-46-150:~$ sudo docker ps
CONTAINER ID   IMAGE          COMMAND                  CREATED        STATUS        PORTS
4de417482d2    php:7.2-apache "docker-php-entrypoi..." 2 minutes ago   Up           0.0.0.0:5000->80/tcp
895eaba7b415    php:7.2-apache "docker-php-entrypoi..." 3 minutes ago   Up           0.0.0.0:5001->80/tcp
ubuntu@ip-172-31-46-150:~$ cd website
ubuntu@ip-172-31-46-150:~/website$
```

그 결과 5000번으로 port 설정을 하였기 때문에 이에 들어가보면, 옆의 화면이 나오며, 밑의 bullet point 들은, 5001번으로 설정한 apache container에서 가져온 값이다.



Docker swarm 실습

Docker swarm init 으로 172.31.46.150 machine이 manage다 라고 initialize 하였다.

```
ubuntu@ip-172-31-46-150:~$ mkdir swarm
ubuntu@ip-172-31-46-150:~$ cd swarm
ubuntu@ip-172-31-46-150:~/swarm$ sudo docker swarm init --advertise-addr=172.31.46.150
Swarm initialized: current node (1l4hygbc4h3bgqk2e3j6h4fuw) is now a manager.

To add a worker to this swarm, run the following command:

    docker swarm join --token SWMTKN-1-4iligmadm4gyrdokfvabf13ysblz5gz7a22pr2mellgy8hndz0-c7uj6yus6zkwycyk9rv4axypz 172.31.46.150:2377

To add a manager to this swarm, run 'docker swarm join-token manager' and follow the instructions.
```

그 후, 172.31.22.113에 위에서 swarm 에 worker로서 join을 시켰다.

```
ubuntu@ip-172-31-22-113:~$ sudo docker swarm join --token SWMTKN-1-4iligmadm4gyrdokfvabf13ysblz5gz7a22pr2mellgy8hndz0-c7uj6yus6zkwycyk9rv4axypz 172.31.46.150:2377
This node joined a swarm as a worker.
ubuntu@ip-172-31-22-113:~$
```

그 후 initializing이 되었으니, swarm service를 실행시켰다. 이름은 Newweb 그리고 port forwarding은 5000:80으로 하였다. 생성은 wordpress 5개로 하였다.

```
ubuntu@ip-172-31-46-150:~/swarm$ sudo docker service create --name Newweb --replicas 5 -p 5000:80 wordpress
sfcwclcqdl09ug494bfbdpauk
overall progress: 5 out of 5 tasks
1/5: running
2/5: running
3/5: running
4/5: running
5/5: running
verify: Service converged
ubuntu@ip-172-31-46-150:~/swarm$
```

오른쪽 그림에서 같이, docker service ls command를 이용하면 상세내용을 볼 수 있으며, replicas가 5/5로 되어있다.

```
ubuntu@ip-172-31-46-150:~/swarm$ sudo docker service ls
ID                NAME                MODE                REPLICAS            IMAGE
kiy6q53wxy42     Newweb              replicated          5/5                 wordpress:latest
*:5000->80/tcp
ubuntu@ip-172-31-46-150:~/swarm$ ^C
ubuntu@ip-172-31-46-150:~/swarm$ sudo docker ps
CONTAINER ID        IMAGE                COMMAND              CREATED             STATUS              PORTS              NAMES
78292c6284be       wordpress:latest     "docker-entrypoint.s..." About a minute ago   Up About a minute   80/tcp             Newweb.1.2tr7oxagtgvwdqaljqzi32bh
30ba5e967fe9       wordpress:latest     "docker-entrypoint.s..." About a minute ago   Up About a minute   80/tcp             Newweb.3.qdpikz79hxafd2djtnjccaa4x
```

그리하여,

manager machine에서 보면, 2개의 container가 작동중이고,

Worker machine에서 보면, 3개의 container가 작동중이다.

```
ubuntu@ip-172-31-22-113:~/swarm$ sudo docker ps
CONTAINER ID        IMAGE                COMMAND              CREATED             STATUS              PORTS              NAMES
1c5a614ede8d       wordpress:latest     "docker-entrypoint.s..." 4 minutes ago       Up 4 minutes       80/tcp             Newweb.2.q6d2c3z7s4meci560k09bshjt
f0445949d493       wordpress:latest     "docker-entrypoint.s..." 4 minutes ago       Up 4 minutes       80/tcp             Newweb.4.97doh9w7wlt5qmm7vbaifwfgo
5899090e7695       wordpress:latest     "docker-entrypoint.s..." 4 minutes ago       Up 4 minutes       80/tcp             Newweb.5.ivpnsrgkyw45c5z1fus3vsjlm
```

그리하여 총 생성된 5개가 다 작동중임을 알 수 있다.

이번에는 replication 개수를 docker service scale command로 7개로 늘려보았다.

그리고, Manager machine에서의 container들을 다 없애 보았다.

처음 docker ps command를 보면, container가 모두 삭제되었지만, 그 후, 시간이 조금 지나면, 다시 원래의 개수(7)를 맞추기 위해서 다시 실행시킴을 볼 수 있다.

이때, 우리가 만든 replicas는 7개이므로, manager machine에서의 container 수 3개와 worker machine에서의 container 수 4개를 보아, 총 7개가 작동하고 있음을 알 수 있다.

```
ubuntu@ip-172-31-46-150:~/swarm$ sudo docker service scale Newweb=7
Newweb scaled to 7
overall progress: 7 out of 7 tasks
1/7: running
2/7: running
3/7: running
4/7: running
5/7: running
6/7: running
7/7: running
```

```
verify: Service converged
ubuntu@ip-172-31-46-150:~/swarm$ sudo docker rm -f $(sudo docker ps -a -q)
```

```
107e1843adfb
3c96cd0e81f1
78292c6284be
30ba5e967fe9
ubuntu@ip-172-31-46-150:~/swarm$ sudo docker ps
CONTAINER ID        IMAGE               COMMAND             CREATED             STATUS
PORTS              NAMES
ubuntu@ip-172-31-46-150:~/swarm$ sudo docker ps
CONTAINER ID        IMAGE               COMMAND             CREATED             STATUS
PORTS              NAMES
ubuntu@ip-172-31-46-150:~/swarm$ sudo docker ps
CONTAINER ID        IMAGE               COMMAND             CREATED             STATUS
PORTS              NAMES
08508361cc2a       wordpress:latest    "docker-entrypoint.s..." About a minute ago   Up
  40 seconds       80/tcp              Newweb.1.s7itqr7uzwohrz4f8s3cq9c4a
64c83e4a1e55       wordpress:latest    "docker-entrypoint.s..." About a minute ago   Up
  42 seconds       80/tcp              Newweb.3.uj6kkibxl8zzgytpaa0lqtrvo
55f18a89a113       wordpress:latest    "docker-entrypoint.s..." About a minute ago   Up
  44 seconds       80/tcp              Newweb.6.ut25j0ti6fcogwue5jswyg3jx
ubuntu@ip-172-31-46-150:~/swarm$
```

```
ubuntu@ip-172-31-22-113:~/swarm$ sudo docker ps
CONTAINER ID        IMAGE               COMMAND             CREATED
STATUS            PORTS              NAMES
877dcfb14f4a       wordpress:latest    "docker-entrypoint.s..." 30 seconds ago
Up 24 seconds     80/tcp              Newweb.7.o5wq5pdnlptj741c6oll5kmw6
1c5a614ede8d       wordpress:latest    "docker-entrypoint.s..." 7 minutes ago
Up 7 minutes      80/tcp              Newweb.2.q6d2c3z7s4meci560k09bshjt
f8445949d493       wordpress:latest    "docker-entrypoint.s..." 7 minutes ago
Up 7 minutes      80/tcp              Newweb.4.97doh9w7wlt5qmm7vbaifwfgo
5899090e7695       wordpress:latest    "docker-entrypoint.s..." 7 minutes ago
Up 7 minutes      80/tcp              Newweb.5.ivpnsrgkyw45c5z1fus3vsjlm
```

추가로, 이번 실습을 하기위해,
docker service ls,
docker service ps,
docker service leave,
docker service rm <service ID> 등을 사용해 보았다.