쿠버네티스를 활용한 멀티클라우드 도입과 운영전략
- AWS, Azure, GCP 비교와 실습

2020년 3월
아인인텔리전스
권재원, Ph.D.

Lab: Introduction to Docker

- Using GCP CloudShell

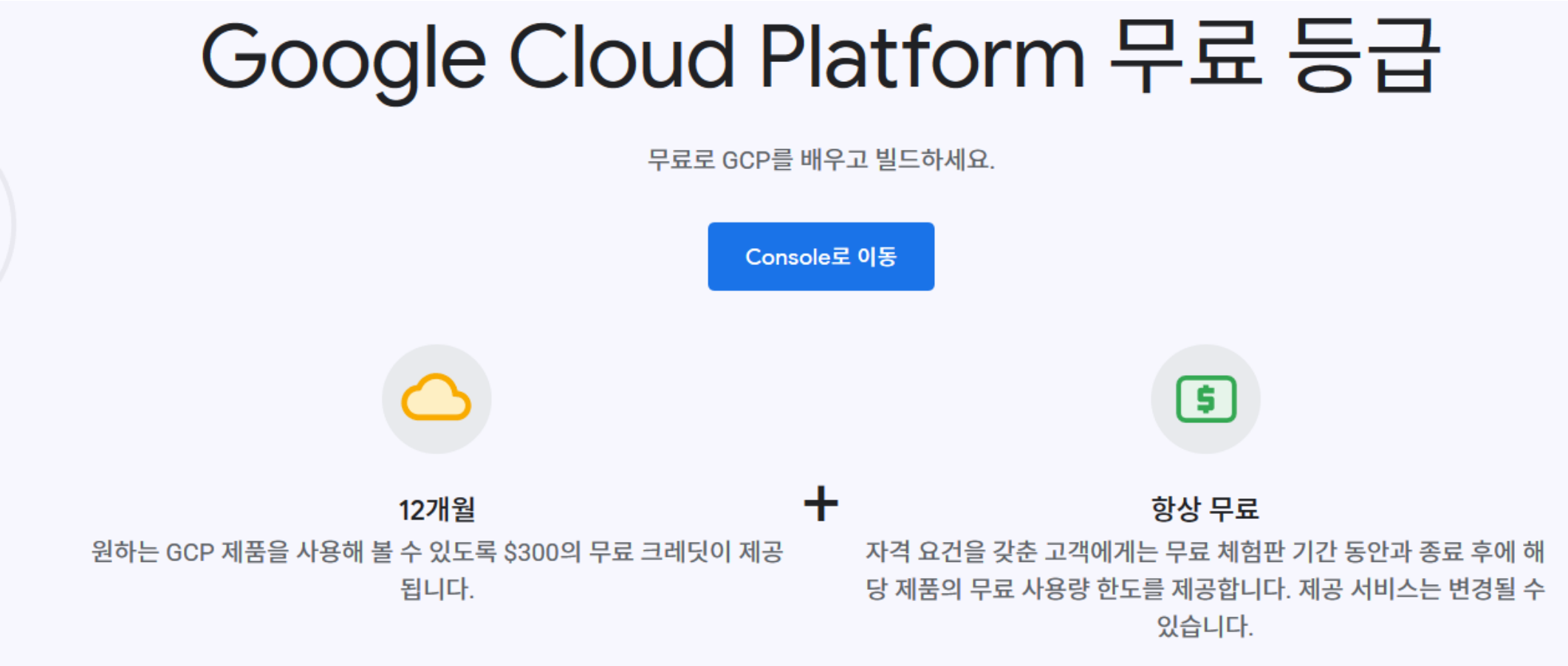
실습 : 구글클라우드 시작하기 (1)

- **Gmail account를 이용하여 계정 만들기**

https://cloud.google.com/start/?hl=ko

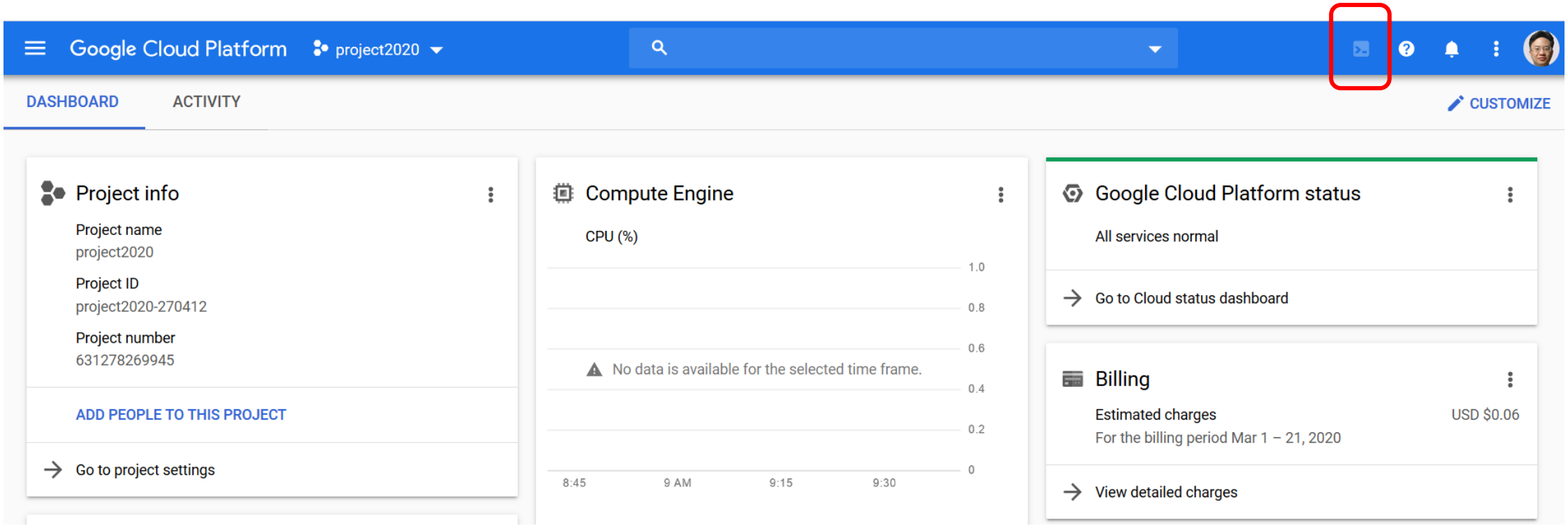
실습 : 구글클라우드 시작하기 (2)

- **GCP Free-tier**



https://cloud.google.com/free/?hl=ko

실습 : CloudShell 시작하기



Docker Lab

- Using GCP CloudShell
- Build / Run / Debug / Publish

QwikLabs - https://google.qwiklabs.com/catalog_lab/944

Docker : Hello-World

실습 : Introduction to Docker

- **Setup and Requirements (GCP CloudShell)**

```
$ gcloud auth list
$ gcloud config set project {project_name}
$ gcloud config list project
```

** Google Cloud gcloud Overview - https://cloud.google.com/sdk/gcloud*

- **Hello World**

```
$ docker run hello-world
$ docker images
$ docker run hello-world
$ docker ps
$ docker ps -a
```

CONTAINER ID	IMAGE	COMMAND	...	NAMES
6027ecba1c39	hello-world	"/hello"	...	elated_knuth
358d709b8341	hello-world	"/hello"	...	epic_lewin

** The Image ID is in SHA256 hash format*

Docker : Build

실습 : Introduction to Docker

- **Build**
  - mkdir dockertest && cd dockertest
  - nano Dockerfile

```
FROM node:6
WORKDIR /app
ADD . /app
EXPOSE 80
CMD ["node", "app.js"]
```

Dockerfile command references - https://docs.docker.com/engine/reference/builder/#known-issues-run

실습 : Introduction to Docker

- nano app.js

```
const http = require('http');
const hostname = '0.0.0.0';
const port = 80;
const server = http.createServer((req, res) => {
  res.statusCode = 200;
   res.setHeader('Content-Type', 'text/plain');
    res.end('Hello World\n');
});

server.listen(port, hostname, () => {
  console.log('Server running at http://%s:%s/', hostname, port);
});

process.on('SIGINT', function() {
  console.log('Caught interrupt signal and will exit');
  process.exit();
});
```

실습 : Introduction to Docker

○ Docker build

docker build -t node-app:0.1 .

```
Sending build context to Docker daemon 3.072 kB
Step 1 : FROM node:6
6: Pulling from library/node
...
...
...
Step 5 : CMD node app.js
 ---> Running in b677acd1edd9
 ---> f166cd2a9f10
Removing intermediate container b677acd1edd9
Successfully built f166cd2a9f10
```

- The -t is to name and tag an image with the name:tag syntax
- The name of the image is node-app and the tag is 0.1

실습 : Introduction to Docker

○ Docker images

docker images

```
REPOSITORY     TAG      IMAGE ID       CREATED            SIZE
node-app       0.1      f166cd2a9f10   25 seconds ago     656.2 MB
node           6        5a767079e3df   15 hours ago       656.2 MB
hello-world    latest   1815c82652c0   6 days ago         1.84 kB
```

- Notice node is the base image and node-app is the image you built.
- You can't remove node without removing node-app first.
- The size of the image is relatively small compared to VMs.
- Other versions of the node image such as node:slim and node:alpine can give you even smaller images for easier portability.

Docker official repository - https://hub.docker.com/_/node

Docker : Run

실습 : Introduction to Docker

- Docker run

```
docker run -p 4000:80 --name my-app (-d) node-app:0.1
```

```
Server running at http://0.0.0.0:80/
```

- The --name flag allows you to name the container if you like
- The -p instructs Docker to map the host's port 4000 to the container's port 80
- You can reach the server at http://localhost:4000.
- Without port mapping, you would not be able to reach the container at localhost.
- -d flag for running in the background

- Curl
  - 다른 터미널을 + 버튼으로 열어서,

```
curl http://localhost:4000
```

```
Hello World
```

실습 : Introduction to Docker

- Docker stop & docker rm

```
docker stop my-app && docker rm my-app
```

- Docker ps

```
docker run -p 4000:80 --name my-app -d node-app:0.1
docker ps
```

```
CONTAINER ID   IMAGE          COMMAND        CREATED          ...  NAMES
xxxxxxxxxxxx   node-app:0.1   "node app.js"  16 seconds ago   ...  my-app
```

- Docker log

```
docker logs [container_id]
```

```
Server running at http://0.0.0.0:80/
```

실습 : Introduction to Docker

○ nano app.js (기존 파일 수정)

```
const http = require('http');
const hostname = '0.0.0.0';
const port = 80;
const server = http.createServer((req, res) => {
  res.statusCode = 200;
   res.setHeader('Content-Type', 'text/plain');
    res.end('Welcome to Cloud\n');
});

server.listen(port, hostname, () => {
  console.log('Server running at http://%s:%s/', hostname, port);
});

process.on('SIGINT', function() {
  console.log('Caught interrupt signal and will exit');
  process.exit();
});
```

실습 : Introduction to Docker

- Docker build

docker build -t node-app:0.2 .

```
Step 1/5 : FROM node:6
 ---> 67ed1f028e71
Step 2/5 : WORKDIR /app
 ---> Using cache
 ---> a39c2d73c807
Step 3/5 : ADD . /app
 ---> a7087887091f
Removing intermediate container 99bc0526ebb0
Step 4/5 : EXPOSE 80
 ---> Running in 7882a1e84596
 ---> 80f5220880d9
Removing intermediate container 7882a1e84596
Step 5/5 : CMD node app.js
 ---> Running in f2646b475210
 ---> 5c3edbac6421
Removing intermediate container f2646b475210
Successfully built 5c3edbac6421
Successfully tagged node-app:0.2
```

실습 : Introduction to Docker

- Docker stop & docker rm

```
docker run -p 8080:80 --name my-app-2 -d node-app:0.2
docker ps
```

```
CONTAINER ID     IMAGE            COMMAND            CREATED
xxxxxxxxxxxx     node-app:0.2     "node app.js"      53 seconds ago
...
xxxxxxxxxxxx     node-app:0.1     "node app.js"      About an hour ago
...
```

```
curl http://localhost:8000
```

```
Welcome to Cloud
```

```
curl http://localhost:4000
```

```
Hello World
```

Docker : Debug

실습 : Introduction to Docker

- Docker stop & docker rm

docker logs -f [container_id]

```
Server running at http://0.0.0.0:80/
```

- You can look at the logs of a container using docker logs [container_id].
- If you want to follow the log's output as the container is running, use the -f option.

- Interactive Bash session inside the running container

- The -it flags let you interact with a container by allocating a pseudo-tty and keeping stdin open.
- Notice bash ran in the WORKDIR directory (/app) specified in the Dockerfile.

docker exec -it [container_id] bash

```
root@xxxxxxxxxxxx:/app#
```

ls

```
Dockerfile  app.js
root@xxxxxxxxxxxx:/app#
```

exit

실습 : Introduction to Docker

○ Docker stop & docker rm

docker inspect [container_id]

```
[
    {
        "Id": "xxxxxxxxxxxx....",
        "Created": "2017-08-07T22:57:49.261726726Z",
        "Path": "node",
        "Args": [
            "app.js"
        ],
...
```

- Use --format to inspect specific fields from the returned JSON

docker inspect --format='{{range .NetworkSettings.Networks}}{{.IPAddress}}{{end}}' [container_id]

```
192.168.9.3
```

Docker inspect - https://docs.docker.com/engine/reference/commandline/inspect/#examples
Docker exec - https://docs.docker.com/engine/reference/commandline/exec/

Docker Publish

실습 : Introduction to Docker

○ Push your image to the Google Container Registry (gcr)

- You need to tag the images with a registry name - [hostname]/[project-id]/[image]:[tag]
  - [hostname]= gcr.io
  - [project-id]= your project's ID
  - [image]= your image name
  - [tag]= any string tag of your choice. If unspecified, it defaults to "latest"
- Tag node-app:0.2. Replace [project-id] with your configuration.

```
gcloud config list project
docker tag node-app:0.2 gcr.io/[project-id]/node-app:0.2
docker images
```

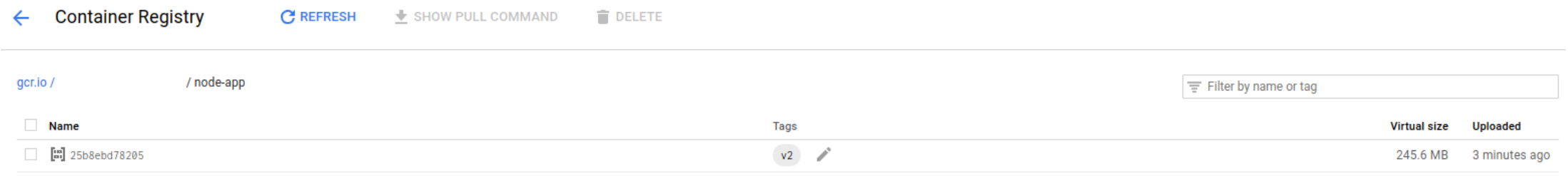
```
REPOSITORY                    TAG       IMAGE ID       CREATED
node-app                      0.2       76b3beef845e   22 hours
ago
gcr.io/[project-id]/node-app  0.2       76b3beef845e   22 hours
ago
node-app                      0.1       f166cd2a9f10   26 hours
ago
node                          6         5a767079e3df   7 days ago
hello-world                   latest    1815c82652c0   7 weeks ago
```

실습 : Introduction to Docker

docker push gcr.io/[project-id]/node-app:0.2

```
The push refers to a repository [gcr.io/[project-id]/node-app]
057029400a4a: Pushed
342f14cb7e2b: Pushed
903087566d45: Pushed
99dac0782a63: Pushed
e6695624484e: Pushed
da59b99bbd3b: Pushed
5616a6292c16: Pushed
f3ed6cb59ab0: Pushed
654f45ecb7e3: Pushed
2c40c66f7667: Pushed
0.2: digest:
sha256:25b8ebd7820515609517ec38dbca9086e1abef3750c0d2aff7f341407c743c46
size: 2419
```

- You can navigate via the console to Tools > Container Registry
- http://gcr.io/[project-id]/node-app



실습 : Introduction to Docker

- Stop and remove all containers

```
docker stop $(docker ps -q)
docker rm $(docker ps -aq)
```

- You have to remove the child images (of node:6) before you remove the node image.

```
docker rmi node-app:0.2 gcr.io/[project-id]/node-app node-app:0.1
docker rmi node:6
docker rmi $(docker images -aq) # remove remaining images
docker images
```

```
REPOSITORY          TAG                 IMAGE ID            CREATED
SIZE
```

- Docker log

```
docker pull gcr.io/[project-id]/node-app:0.2
docker run -p 4000:80 -d gcr.io/[project-id]/node-app:0.2
curl http://localhost:4000
```

```
Welcome to Cloud
```