

Google Cloud Platform 사용법

AlphaZero를 학습시키는 데는 고성능의 컴퓨터가 필요합니다. 따라서 이번 OSS 겨울 캠프에서는 Google Cloud Platform(이하 GCP)를 사용하도록 하겠습니다

1. GCP 시작하기

우선 GCP 홈페이지(<https://cloud.google.com/>)에 접속합니다. 구글 로그인을 하고 **무료로 시작하**기를 누르면 다음과 같은 화면이 나옵니다.

Google Cloud Platform 무료로 사용해 보기

1/2단계

국가

대한민국

계속

개인정보처리방침 | FAQ

모든 Cloud Platform 제품에 액세스

Firebase, Google Maps API 등을 포함해 앱, 웹사이트, 서비스를 구축하고 실행하는 데 필요한 모든 기능을 이용할 수 있습니다.

\$300의 무료 크레딧

가입하여 Google Cloud Platform에서 12개월간 사용할 수 있는 \$300 크레딧을 받아 보세요.

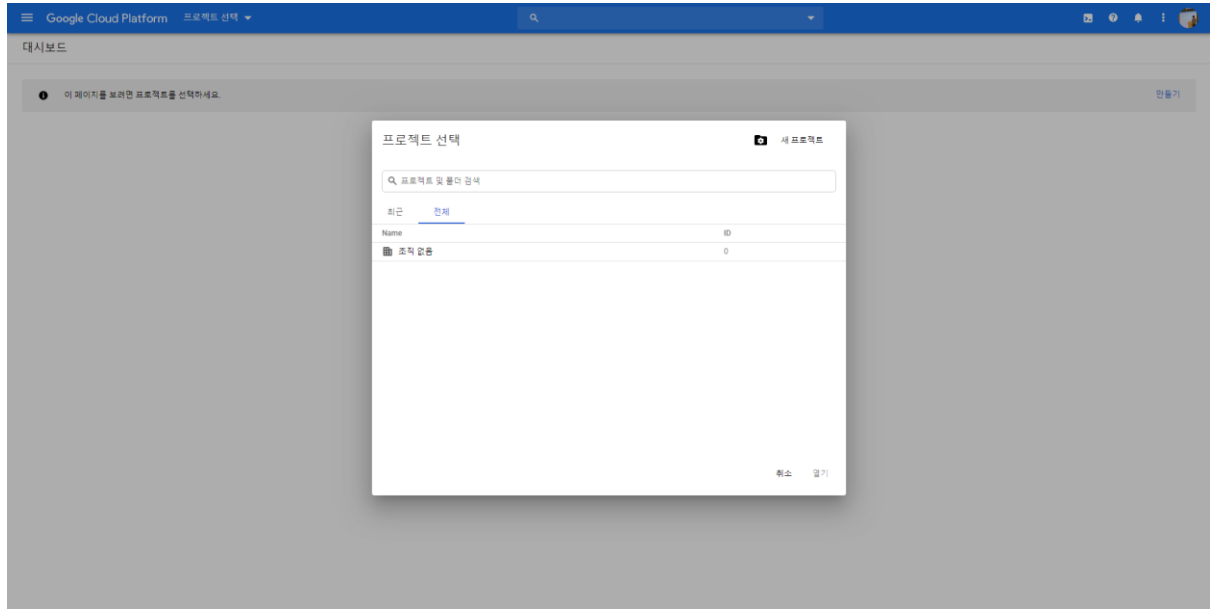
무료 체험판 종료 후 자동 청구되지 않음

신용카드를 요청하는 이유는 자동 가입을 방지하기 위해서입니다. 유료 계정으로 직접 업그레이드하지 않는 한 요금이 청구되지 않습니다.

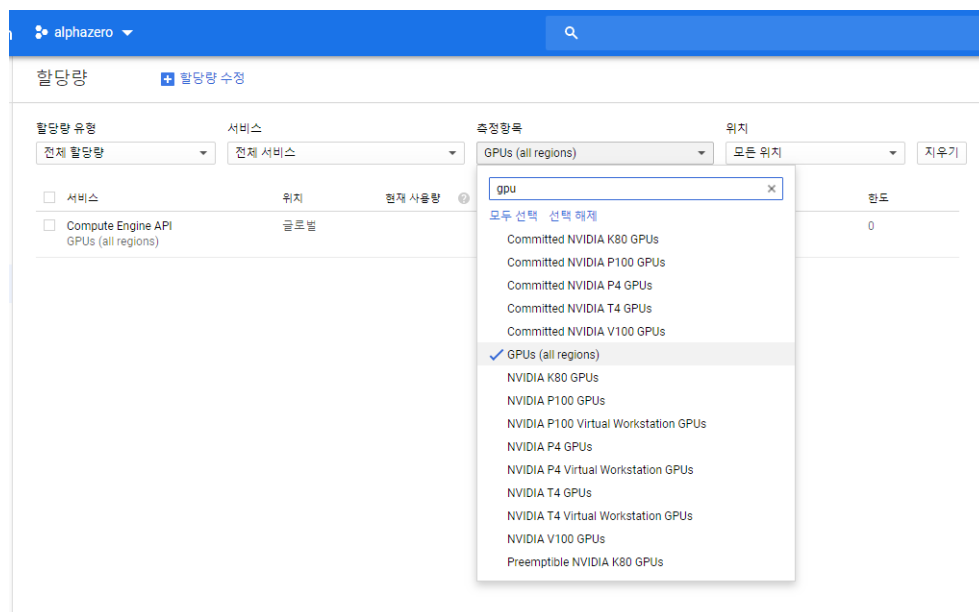
이후 구글이 안내하는 대로 과정을 진행하면 됩니다.

2. 프로젝트 생성

처음 나오는 화면에서 좌상단의 **프로젝트 선택**을 누르고 **새 프로젝트**를 눌러 프로젝트를 만들어 줍니다.



그 이후 좌상단의 ≡ 버튼을 눌러 **IAM 및 관리자의 할당량**에 들어갑니다. 이후 측정항목에서 모두 선택 취소 후 **GPUs (all regions)**를 검색해 다음과 같이 선택합니다.



이후 다음과 같이 선택한 뒤 **할당량 수정**을 누르고 1로 할당량을 수정한 뒤 **요청제출**을 누릅니다.

할당량 수정

할당량 유형: 전체 할당량 | 서비스: 전체 서비스 | 측정 항목: GPUs (all regions) | 위치: 모든 위치 | 지우기

서비스	위치	현재 사용량	7일 최고 사용량	한도
Compute Engine API GPUs (all regions)	글로벌		-	0

할당량: GPUs (all regions)

새 할당량 한도
새 할당량 한도를 입력하세요. 승인을 위해 해당 요청이 서비스 제공업체에 전송됩니다.

1

요청 설명
필수

I need gpu for Deep Reinforcement Learning research.

완료 취소

요청 제출 뒤로

할당량 수정의 승인까지는 약간의 시간이 소요됩니다.

3. VM 인스턴스 만들기

승인이 완료됐다면 이제 VM 인스턴스를 만들 차례입니다. 이번엔 **Compute Engine**의 **VM 인스턴스** 메뉴를 선택합니다.

만들기 버튼을 누른 뒤, 다음과 같이 설정합니다.

이름 [?]
이름은 영구적입니다.

리전 [?]
리전은 영구적입니다.

영역 [?]
영역은 영구적입니다.

머신 구성 [?]

머신 계열
☒ 일반 용도 ☐ 메모리 최적화
일반적인 작업 부하에 적합한 머신 유형이며 가격 및 유연성을 위해 최적화되었습니다.

시리즈

Intel Skylake CPU 플랫폼 또는 이전 버전의 플랫폼에서 제공

머신 유형

코어
 vCPU 1 - 96

메모리
 GB 11 - 78

☐ 메모리 확장 [?]

CPU 플랫폼 [?]
CPU 플랫폼 구성은 영구적입니다.


GPU 유형 GPU 수 ✕
☐ 가상 워크스테이션 사용 설정(NVIDIA GRID)

디스플레이 기기
스크린 캡처 및 녹화 도구를 사용하려면 디스플레이 기기를 사용 설정하세요.
☐ 디스플레이 기기 사용 설정

[CPU 플랫폼 및 GPU](#)

컨테이너 [?]
☐ 이 VM 인스턴스에 컨테이너 이미지를 배포합니다. [자세히 알아보기](#)

부팅 디스크 [?]

 새로운 50GB 표준 영구 디스크 이미지
Deep Learning Image: PyTorch 1.3.0 an...

ID 및 API 액세스 [?]

서비스 계정 [?]

액세스 범위 [?]
☒ 기본 액세스 허용
☐ 모든 Cloud API에 대한 전체 액세스 허용
☐ 각 API에 액세스 설정

방화벽 [?]
태그 및 방화벽 규칙을 추가하여 인터넷에서 특정 네트워크 트래픽을 허용합니다.
☐ HTTP 트래픽 허용
☐ HTTPS 트래픽 허용

[관리, 보안, 디스크, 네트워킹, 단독 임대](#)


다음 옵션이 맞춤설정되었습니다.

호스트 유지관리 시

이 인스턴스의 요금이 청구됩니다. [Compute Engine 가격 책정](#)

동등한 REST 또는 명령줄

설정이 완료됐다면 **만들기** 버튼을 누릅니다.

<input type="checkbox"/> 이름 ^	영역	권장사항	다음에서 사용	내부 IP	외부 IP	연결
<input type="checkbox"/> <input checked="" type="checkbox"/> instance-1	us-central1-a			 (nic0)		SSH ▾ ⋮

이후의 작업은 **SSH** 버튼을 눌러 진행합니다.

4. AlphaZero 개발 환경 구축

ssh를 처음 실행시킨다면 다음의 메시지가 뜹니다.

```
Would you like to install the Nvidia driver? [y/n]
```

y를 입력하여 Nvidia 드라이버를 설치합니다.

다음의 명령어를 이용해 tensorboard를 설치합니다.

```
python3 -m pip install tensorboard==1.13.1 tensorboardX==1.4
```

이제 모든 준비가 끝났습니다! 고생하셨습니다. 아래 부록은 알아 두면 좋을 리눅스 명령어를 적어두었습니다.

Appendix A. 유용한 리눅스 명령어 구문 모음

- screen
 - ssh 접속이 끊어져도 학습이 멈추지 않도록 할 때 사용할 수 있음.
 - screen -S <이름>
<이름>으로 된 가상의 스크린을 만들어줌.
 - screen -r <이름>
이미 만들어진 <이름>의 스크린에 다시 접속함.
 - screen -list
만들어진 스크린들의 목록을 표시함.
 - Ctrl+a d
스크린에 접속 해제함. (이때, 스크린은 사라지지 않고 계속 동작함)
 - Ctrl+d
스크린을 종료함. (해당 스크린에서 작업하던 것은 모두 날아감)
- cat <파일 이름>
작은 파일을 볼 때 유용함.
- less <파일 이름>
큰 파일을 볼 때 유용함. 상하 방향키로 이동 가능. (자매품 j, k)

- `tail -f <파일 이름>`
파일의 마지막 10줄을 실시간으로 계속 보내준다.

- `ps -ef | grep <프로세스 이름>`
현재 실행중인 프로세스 이름을 가져올 수 있다.

```
jun@raspberrypi:~$ ps -ef | grep python3
root      1206      1  0   2019 ?        00:00:02 /usr/bin/python3 /usr/bin/networkd-dispatcher -
-run-startup-triggers
root      1242      1  0   2019 ?        00:00:00 /usr/bin/python3 /usr/share/unattended-upgrades
/unattended-upgrade-shutdown --wait-for-signal
jun       18152 18145  0  11:28 pts/1    00:00:00 python3
jun       18156 18118  0  11:28 pts/0    00:00:00 grep --color=auto python3
```

- `kill <pid>`
해당 프로세스를 강제로 종료시킨다. (pid는 위 그림에서 사용자 이름 뒤에 오는 숫자를 의미함. 예컨대 python3의 pid는 18152)
- `top` (예쁜 버전: `htop`)
Windows의 작업관리자와 유사한 역할을 한다.

[top 실행 화면]

```
top - 11:32:11 up 19 days, 9:01, 1 user, load average: 0.00, 0.00, 0.00
Tasks: 122 total, 1 running, 121 sleeping, 0 stopped, 0 zombie
%Cpu(s):  0.1 us,  0.1 sy,  0.0 ni, 99.8 id,  0.0 wa,  0.0 hi,  0.0 si,  0.0 st
MiB Mem :  3791.4 total,  2247.8 free,  215.9 used,  1327.8 buff/cache
MiB Swap:   0.0 total,   0.0 free,   0.0 used.  3516.9 avail Mem

  PID USER      PR  NI  VIRT  RES  SHR S %CPU  %MEM    TIME+  COMMAND
18164 jun       20   0  10644  3204  2764 R   0.3   0.1   0:00.05 top
   1 root       20   0 166912 10092  6892 S   0.0   0.3   0:37.82 systemd
   2 root       20   0      0      0      0 S   0.0   0.0   0:01.73 kthreadd
   3 root       0 -20      0      0      0 I   0.0   0.0   0:00.00 rcu_gp
   4 root       0 -20      0      0      0 I   0.0   0.0   0:00.00 rcu_par_gp
   8 root       0 -20      0      0      0 I   0.0   0.0   0:00.00 mm_percpu_wq
   9 root       20   0      0      0      0 S   0.0   0.0   0:04.88 ksoftirqd/0
  10 root       20   0      0      0      0 I   0.0   0.0   0:48.49 rcu_sched
  11 root       rt    0      0      0      0 S   0.0   0.0   0:09.17 migration/0
  12 root      -51   0      0      0      0 S   0.0   0.0   0:00.00 idle_inject/0
```

[htop 실행 화면]

```
 1  [                               0.0%]  Tasks: 29, 32 thr; 1 running
 2  [                               0.0%]  Load average: 0.00 0.00 0.00
 3  [                               0.0%]  Uptime: 19 days, 09:02:35
 4  [|||||                          1.1%]
Mem[|||||                          219M/3.70G]
Swp[                                0K/0K]

  PID USER      PRI  NI  VIRT  RES  SHR S %CPU  %MEM    TIME+  Command
18167 jun       20   0  7148  3000  2384 R   1.1   0.1   0:00.18 htop
   1 root       20   0 163M 10092  6892 S   0.0   0.3   0:37.82 /sbin/init fixrtc
  726 root       19  -1 66924 19336 18096 S   0.0   0.5   0:10.29 /lib/systemd/systemd-journald
  737 root       20   0 19956  5256  3388 S   0.0   0.1   0:06.24 /lib/systemd/systemd-udev
 1096 systemd-n  20   0 25480  6320  5424 S   0.0   0.2   0:09.92 /lib/systemd/systemd-networkd
F1 Help F2 Setup F3 Search F4 Filter F5 Tree F6 SortBy F7 Nice F8 Nice + F9 Kill F10 Quit
```

- `diff <파일 이름1> <파일 이름2>`
두 파일의 차이점을 알려준다.

```
jun@raspberrypi:~$ diff a b
2c2
< abcd
---
> qwer
jun@raspberrypi:~$

jun@raspberrypi:~$ cat a
1234
abcd
jun@raspberrypi:~$ cat b
1234
qwer
jun@raspberrypi:~$
```

[0] 0:bash* "raspberrypi" 11:39 08-Jan-20

- mv <원본 파일> <옮길 위치>
파일을 이동합니다. (파일 이름 바꾸기 할 때도 사용됨)
- cp <원본 파일> <위치>
파일을 복사합니다.
- rm <파일 이름>
파일을 삭제합니다.
- rm -rf <폴더 이름>
폴더를 삭제합니다.
- cd <경로>
해당 경로로 이동합니다.
- ls
현재 위치에 있는 파일 목록을 보여줍니다.
- ls <경로>
경로에 있는 파일 목록을 보여줍니다.
- pwd
현재 작업 경로를 보여줍니다.
- grep -r "내용" *
현재 폴더의 파일 중 해당 내용이 있는 파일을 모두 찾습니다. (하위 폴더도 포함)