

事前準備

SA 新田・田中

目次

- aicastセットアップ手順
- Google Drive
- AWSにログイン
- インスタンスを立てる
- ssh接続
- 環境構築
- 参考


aicastセットアップ手順

<https://actcast.io/docs/ja/ForVendor/ApplicationDevelopment/aicast/SetupActsim/>

Google Drive


[Google Drive](#)から必要なファイルをDLしておく


 devcontainer.json

 Dockerfile

 hailo_dataflow_compiler-3.23.0-py3-none-linux_x86_64.whl

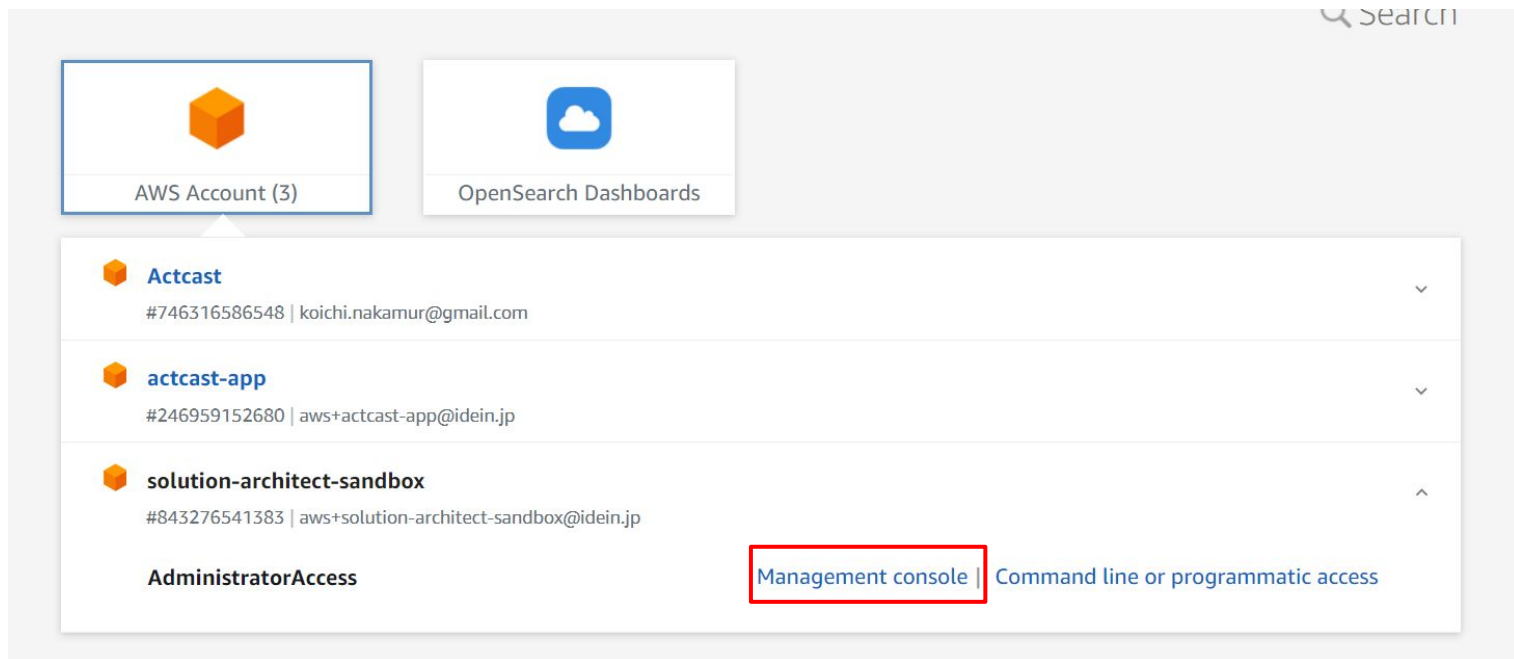
 install_docker.sh

 install_nvidia_container_toolkit.sh

 sa-aicast-handon.pem

AWSにログイン

<https://idein.awsapps.com/start#/> (solution-architect-sandbox前提)



The screenshot shows the AWS IAM console interface. At the top, there are two main sections: 'AWS Account (3)' and 'OpenSearch Dashboards'. Below these, a list of users is displayed. The user 'solution-architect-sandbox' is selected, and its details are shown in a panel below. The panel includes the user's name, ID, email, and a list of permissions. The 'Management console' link is highlighted with a red box.

Search

AWS Account (3)

OpenSearch Dashboards

Actcast
#746316586548 | koichi.nakamur@gmail.com

actcast-app
#246959152680 | aws+actcast-app@idein.jp

solution-architect-sandbox
#843276541383 | aws+solution-architect-sandbox@idein.jp

AdministratorAccess

Management console | Command line or programmatic access

インスタンスを立てる

ホーム画面でec2と検索。EC2に移動し「インスタンスを起動」をクリックする

a. 名前を入力

名前とタグ 情報

名前

alicast-handson-tanaka

さらにタグを追加

b. **ubuntu20.04**を選択

Amazon Linux macOS **Ubuntu** Windows Red Hat SUSE Linux

aws Mac Microsoft Red Hat SUS

その他のAMIを閲覧する

AWS, Marketplace, コミュニティからのAMIを含む

c. インスタンスタイプで**g4dn.xlarge**を選択

Q g4d

※g4adもあるので注意

g4dn.xlarge

ファミリー: **g4dn** 4 vCPU 16 GiB メモリ 現行世代: true

オンデマンド Linux 料金: 0.526 USD 1 時間あたり

オンデマンド SUSE 料金: 0.582 USD 1 時間あたり

オンデマンド Windows 料金: 0.71 USD 1 時間あたり

オンデマンド RHEL 料金: 0.586 USD 1 時間あたり

d. キーペアを「sa-aicast-handson.pem」を選択(個人で作ってもOK)

▼ キーペア (ログイン) 情報

キーペアを使用してインスタンスに安全に接続できます。インスタンスを起動する前に、選択したキーペアにアクセスできることを確認してください。

キーペア名 - 必須

sa-aicast-handson

新しいキーペアの作成

e. 編集を押して下記のように設定する

▼ ネットワーク設定 情報

VPC - 必須 情報

vpc-0f12b41359dc2708e (sa-aicast-handson-vpc) 10.0.0.0/16

サブネット 情報

subnet-03ba547262dc6fa75 sa-aicast-handson-1a

VPC: vpc-0f12b41359dc2708e 所有者: 843276541383 アベイラビリティーゾーン: us-east-1a 利用可能な IP アドレス: 251 CIDR: 10.0.1.0/24

パブリック IP の自動割り当て 情報

有効化

ファイアウォール (セキュリティグループ) 情報

セキュリティグループとは、インスタンスのトラフィックを制御する一連のファイアウォールルールです。特定のトラフィックがインスタンスに到達できるようにルールを追加します。

☐ セキュリティグループを作成する ☒ 既存のセキュリティグループを選択する

共通のセキュリティグループ 情報

セキュリティグループを編集

sa-aicast-handson-sg sg-0a5d2b29605737720 X

VPC: vpc-0f12b41359dc2708e

ここで追加または削除したセキュリティグループは、すべてのネットワークインターフェイスに追加、またはすべてのネットワークインターフェイスから削除されます。

▶ 高度なネットワーク設定

f. ストレージを**50GiB**にする

▼ ストレージを設定 情報

1x 30 GiB gp3 ルートボリューム (暗号化なし)

g. インスタンスを起動

キャンセル

インスタンスを起動

コマンドを確認

パブリックIPアドレスを控えておく

インスタンス (1/2) 情報

Q 属性またはタグ (case-sensitive) で インスタンス を検索



接続

インスタンスの状態 ▼

アクション ▼

インスタンスを起動 ▼

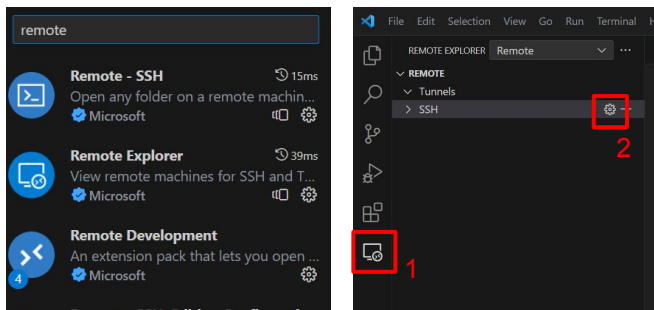
< 1 > ⚙

	Name ▼	インスタンス ID ▼	インスタンス... ▼	イ... ▼	ステータスチエ... ▼	アラームの状態 ▼	アベ... ▼	パブリック IPv4 DNS ▼	パブリック IPv4 アドレス ▼	Elastic IP ▼	IPv6 IP ▼	モニタリング ▼	セキュリティグルー... ▼	キー名 ▼
<input type="checkbox"/>	-	i-0c5c091e32fadc267	実行中	t2.micro	2/2 のチェックに:	アラーム...	us-east-1d	ec2-3-221-46-147.compute-...	3.221.46.147	3.221.46.147	-	disabled	launch-wizard-2	crest-
<input checked="" type="checkbox"/>	sa-aicast-handson2	i-0d204ce7008c9a03d	実行中	t2.micro	-	アラーム...	us-east-1a	-	3.88.218.176	-	-	disabled	sa-aicast-handson-sg	sa-aic

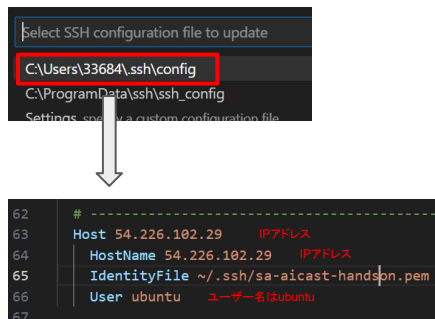
ssh接続

sshでEC2インスタンスに接続するための設定をする

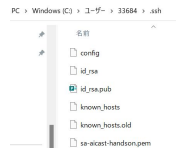
a. vscodeに下記3つの拡張機能をインストールし設定画面を開く



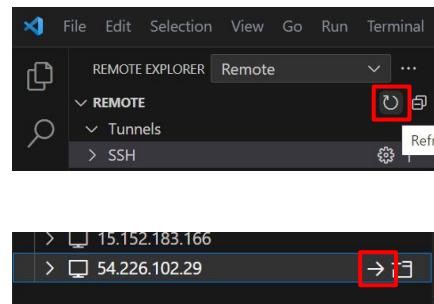
b. configを編集する(下記参照)



- Host: インスタンスのIP
- HostName: インスタンスのIP
- IdentityFile: キーペアのパス
- User: ubuntu



c. Refresh後、設定したIPアドレスに接続する



popupが表示されたら、Linuxを選択しcontinueをクリックする

※windowsの人は認証キーをUserの中の.sshに入れる必要があるかも？(私の場合、wslの.sshではパスが通らない)

環境構築(1/5)

- a. 自身のPCでターミナルを起動し下記コマンドを実行する(winscpでも可)

```
sudo scp -i sa-aicast-handson.pem hailo_dataflow_compiler-3.23.0-py3-none-linux_x86_64.whl  
ubuntu@54.234.98.10:
```

※キーパス、whlパス、IPアドレスは適宜変更してください

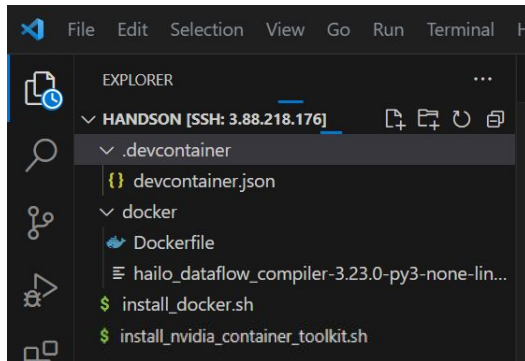
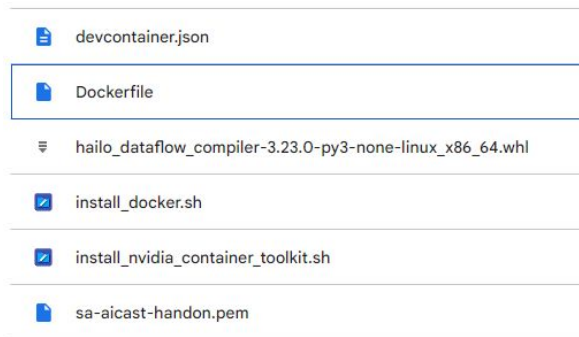
※hailo_dataflow_compiler-3.23.0-py3-none-linux_x86_64.whlはGoogle Driveに格納されています

- b. scpが終わったらsshで接続したサーバーのターミナルを開いて下記コマンドを実行する

```
mkdir handson  
mkdir handson/docker  
mkdir handson/.devcontainer  
mv hailo_dataflow_compiler-3.23.0-py3-none-linux_x86_64.whl handson/docker  
cd handson  
code .
```

環境構築(2/5)

c. [Google Drive](#)からDLしたファイルをAWSのサーバーに作った各フォルダにドラッグ & ドロップでファイルを格納



d. dockerをインストールする※確認が出たら「y」を押してエンター

```
bash install_docker
```

e. NVIDIAドライバのインストール

```
sudo apt-get install -y nvidia-driver-470
```

環境構築(3/5)

f. NVIDIA Container Toolkitをインストールする

```
bash install_nvidia_container_toolkit.sh
```

g. 再起動(ターミナルは落とさずに再接続されるまで待機)⇒ Reload Windowが表示されたらクリック

```
sudo reboot
```

h. インストールされたことを確認する

```
nvidia-smi
```

```
ubuntu@ip-10-0-1-102:~/handson$ nvidia-smi
Wed Jul 12 08:24:12 2023
```

NVIDIA-SMI 470.199.02 Driver Version: 470.199.02 CUDA Version: 11.4									
GPU	Name	Persistence-M	Bus-Id	Disp.A	Volatile Uncorr. ECC				
Fan	Temp	Perf	Par:Usage/Cap	Memory-Usage	GPU-Util	Compute M.	MIG M.		
0	Tesla K80	Off	00000000:00:1E:0	Off	0				
N/A	41C	P8	27W / 145W	4MiB / 11441MiB	0%	Default	N/A		

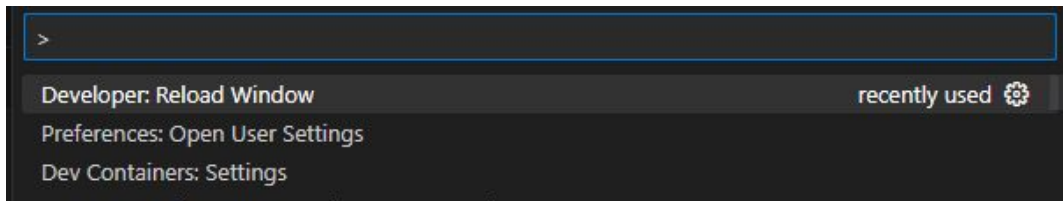
Processes:						
GPU	GI	CI	PID	Type	Process name	GPU Memory Usage
ID	ID	ID				
0	N/A	N/A	763	G	/usr/lib/xorg/Xorg	3MiB

環境構築(4/5)

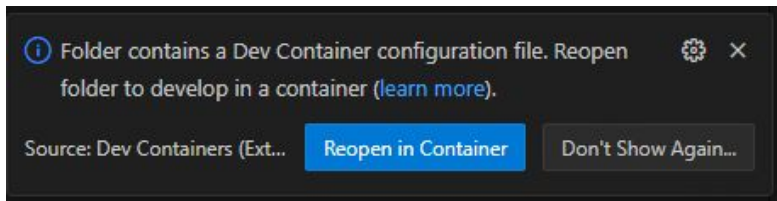
h. handsonに移動する(handsonにいればOK)

```
cd handson  
code .handson
```

i. ctrl + shift + Pを押して「Developer: Reload Window」をクリックする

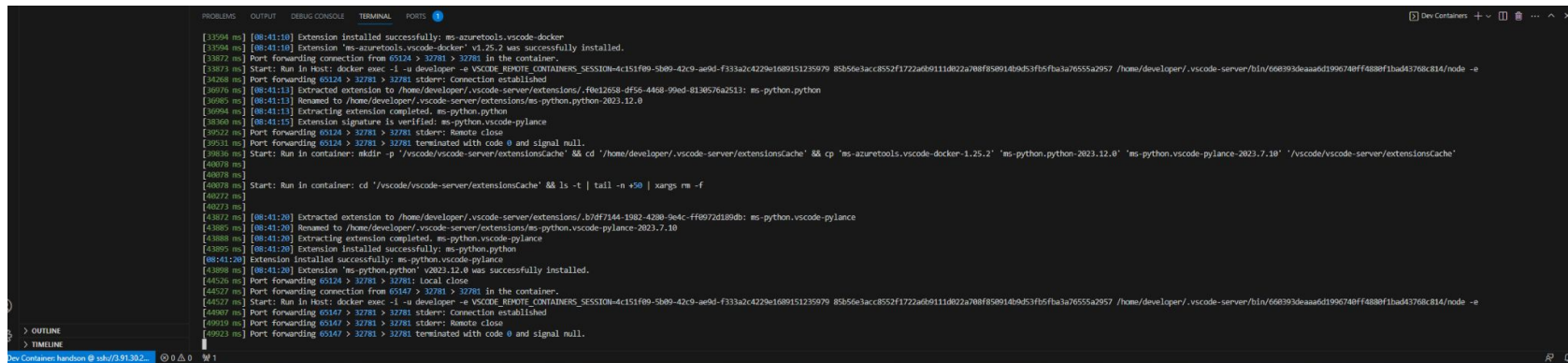


j. 再度windowが立ち上がるので、「Reopen in Container」をクリックする(Containerが作成される)



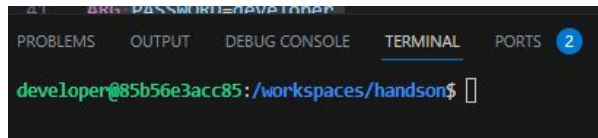
環境構築(5/5)

k. Containerでvscodeを起動することができれば OK



```
[13594 ms] [08:41:10] Extension installed successfully: ms-azuretools.vscode-docker
[13594 ms] [08:41:10] Extension 'ms-azuretools.vscode-docker' v1.25.2 was successfully installed.
[13672 ms] Port forwarding connection from 65124 > 32781 > 32781 in the container.
[13873 ms] Start: Run in Host: docker exec -i -u developer -e VSCODE_REMOTE_CONTAINERS_SESSION=4c151f09-5b09-42c9-ae9d-f333a2c4229e1689151235970 85b56e3acc8552f1722ad60111d802a708f8589140d953f56fba3a76555a2957 /home/developer/.vscode-server/bin/66b393deaaa6d1996740ff4888f1bad43768c814/node --e
[14268 ms] Port forwarding 65124 > 32781 > 32781 stable: Connection established
[14976 ms] [08:41:13] Extracted extension to /home/developer/.vscode-server/extensions/.f8e12658-df56-4408-99ed-8130576a2513: ms-python.python
[15065 ms] [08:41:13] Renamed to /home/developer/.vscode-server/extensions/ms-python.python-2023.12.0
[16094 ms] [08:41:13] Extracting extension completed, ms-python.python
[16360 ms] [08:41:15] Extension signature is verified: ms-python.python
[19022 ms] Port forwarding 65124 > 32781 > 32781 stable: remote close
[19931 ms] Port forwarding 65124 > 32781 > 32781 terminated with code 0 and signal null.
[19836 ms] Start: Run in container: mkdir -p '/vscode/vscode-server/extensionsCache' && cd '/home/developer/.vscode-server/extensionsCache' && cp 'ms-azuretools.vscode-docker-1.25.2' 'ms-python.python-2023.12.0' 'ms-python.python-2023.12.0' '/vscode/vscode-server/extensionsCache'
[40078 ms]
[40078 ms]
[40078 ms] Start: Run in container: cd '/vscode/vscode-server/extensionsCache' && ls -t | tail -n +50 | xargs rm -f
[40272 ms]
[40272 ms]
[43872 ms] [08:41:20] Extracted extension to /home/developer/.vscode-server/extensions/.b7df744-1982-4280-9e4c-ff097d189db: ms-python.python
[43885 ms] [08:41:20] Renamed to /home/developer/.vscode-server/extensions/ms-python.python-2023.7.10
[43888 ms] [08:41:20] Extracting extension completed, ms-python.python
[43895 ms] [08:41:20] Extension installed successfully: ms-python.python
[43901 ms] [08:41:20] Extension installed successfully: ms-python.python
[43998 ms] [08:41:20] Extension 'ms-python.python' v2023.12.0 was successfully installed.
[44026 ms] Port forwarding 65124 > 32781 > 32781 local close
[44527 ms] Port forwarding connection from 65147 > 32781 > 32781 in the container.
[44527 ms] Start: Run in Host: docker exec -i -u developer -e VSCODE_REMOTE_CONTAINERS_SESSION=4c151f09-5b09-42c9-ae9d-f333a2c4229e1689151235970 85b56e3acc8552f1722ad60111d802a708f8589140d953f56fba3a76555a2957 /home/developer/.vscode-server/bin/66b393deaaa6d1996740ff4888f1bad43768c814/node --e
[44589 ms] Port forwarding 65147 > 32781 > 32781 stable: Connection established
[49019 ms] Port forwarding 65147 > 32781 > 32781 stable: remote close
[49923 ms] Port forwarding 65147 > 32781 > 32781 terminated with code 0 and signal null.
```

l. ターミナルを開く



```
developer@85b56e3acc85:/workspaces/hands-on$
```

ここまでで事前準備は以上です。GPUインスタンスは高額なので、当日まではインスタンスを停止しておいてください

参考

- [推奨GPUインスタンス](#)
- [AWS GPUインスタンス コスト比較](#)
- [NVIDIA GPUスペック\(機械学習用\)](#)