



Armadillo-IoT G4 発売記念！ Edge AI作品開発コンテスト meet up！

## 物体認識ハンズオン

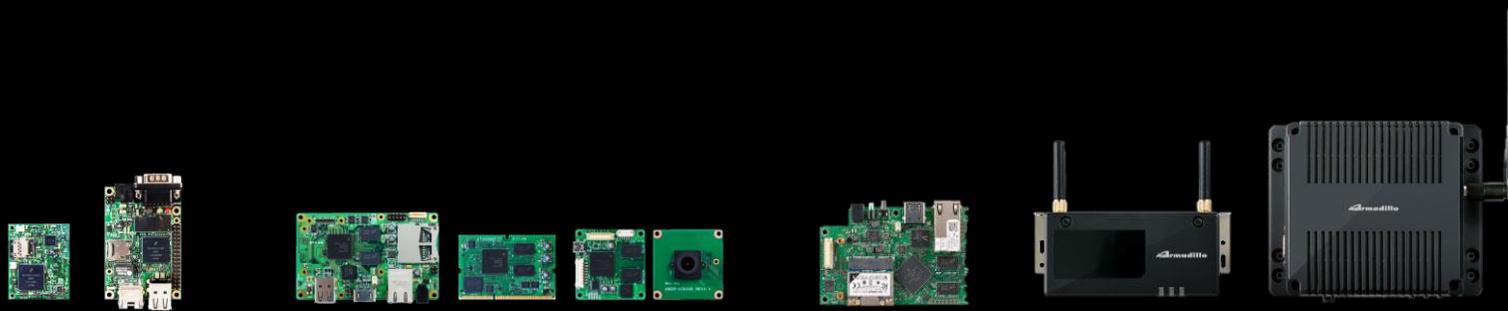
株式会社アットマークテクノ

[www.atmark-techno.com](http://www.atmark-techno.com)



# ハンズオン

---



## ハンズオン事前説明

- ・ハンズオン概要説明
- ・事前準備確認
- ・ハンズオンの進め方説明

## Armadillo-IoT ゲートウェイ G4 セットアップ

- ・ハードウェア接続、開発環境確認
- ・podman ネットワーク設定、データ保存先設定

## 物体認識

- ・コンテナ作成
- ・実行

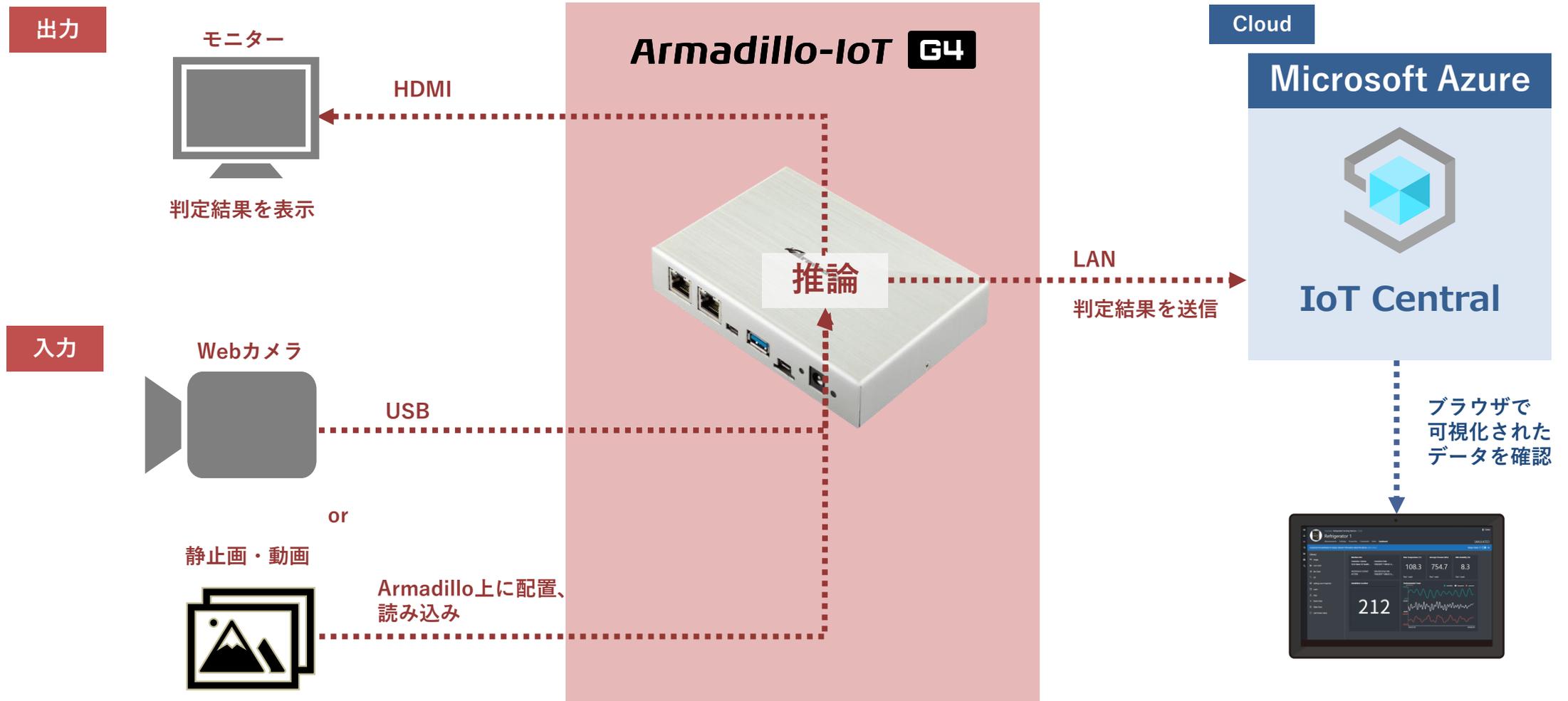
## 物体認識結果 Azure IoT Central 送信

- ・コンテナ作成
- ・実行

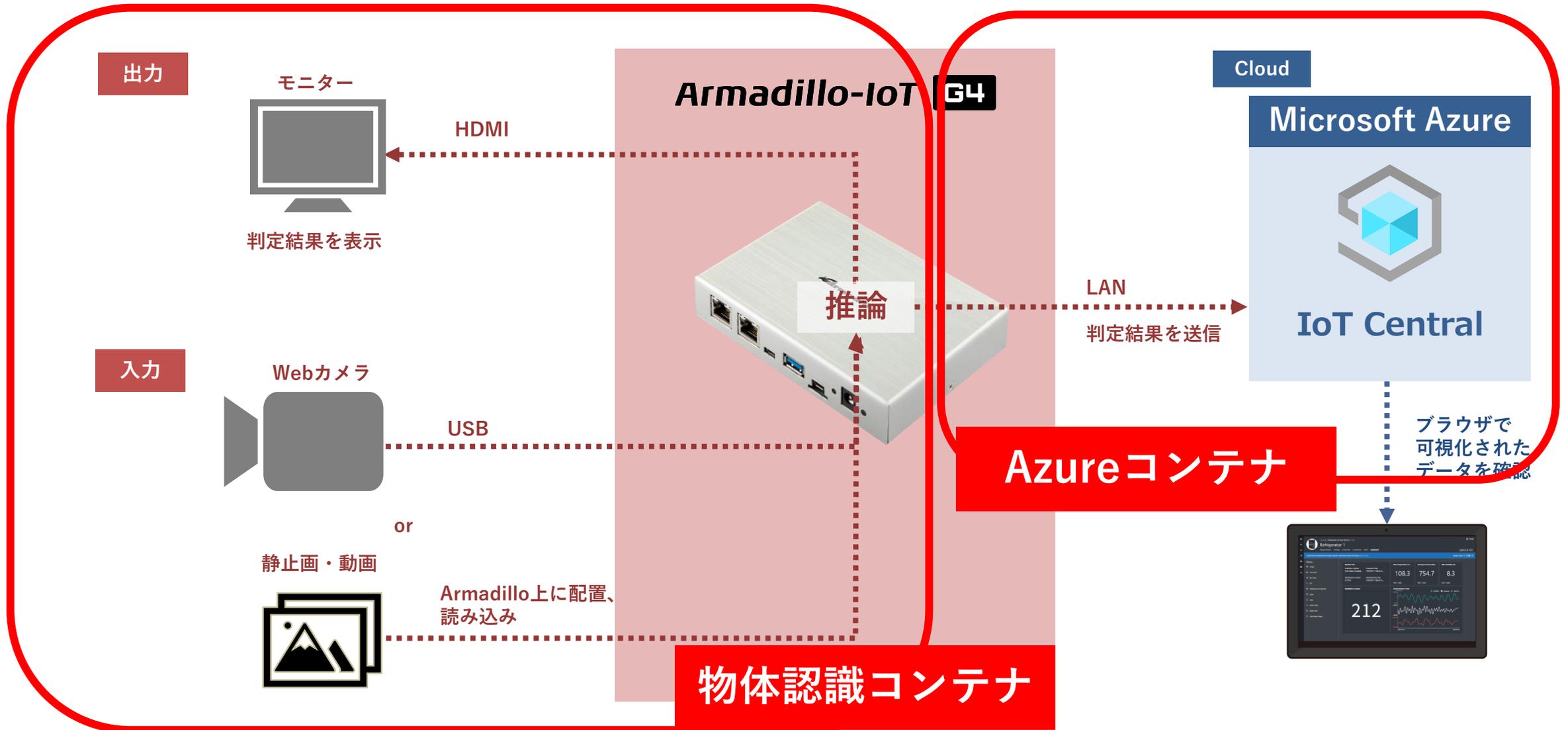
# ハンズオンの目標



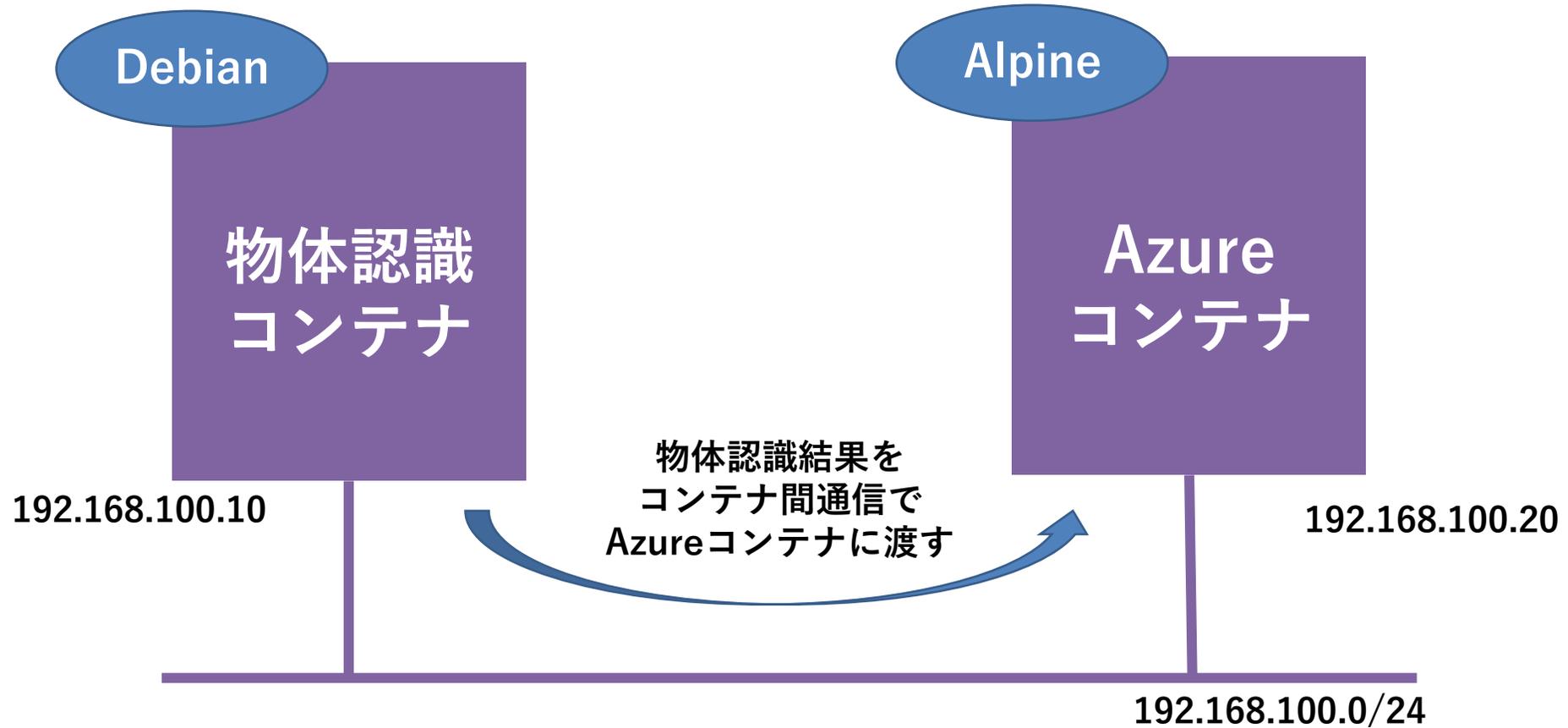
# ハンズオン・コンテンツシステム図



# コンテナの構成



# コンテナの構成



## ■ プレゼント品

すべて揃っていますか？

Armadillo-IoTゲートウェイG4  
LANモデル

ACアダプタ  
(Armadilloの箱に同梱)

USBメモリ

microUSBケーブル  
(Armadilloの箱に同梱)

HDMIケーブル

## ■必要な機材

### ◆ USB (タイプA) が接続可能なPC (Mac およびWindows11は非対象)

- **ターミナルソフト (Tera Termなど)** をあらかじめインストールしてください → ハンズオンでは例として Tera Termを使用します
- **Windows PC** の場合は、下記の手順に従って **USBコンソール用のドライバ** をインストールしてください

<https://armadillo.atmark-techno.com/howto/g4-windows-serial>

### ◆ HDMI接続可能な fullHD 以上のモニタ

### ◆ LANケーブル 1本

### ◆ UVC規格に対応した USB Webカメラ → なくてもOKですが、あった方がよりリアルタイムの物体認識を体感できます！

## ■必要な環境

- ◆DHCP接続可能な有線LAN環境(100BaseT以上推奨)
- ◆Microsoft アカウント
  - ・ Azure サブスクリプション

# 事前準備・確認

- 下記にあるファイルを全てダウンロードしてください

URLをコピーしてアドレスバーに貼り付ける場合は、念のためURLが合っているかご確認ください

- ダウンロードURL ★ 2022年1月4日以降公開します★

[https://download.atmark-techno.com/sample/algyan/220108\\_g4\\_meetup/file/](https://download.atmark-techno.com/sample/algyan/220108_g4_meetup/file/)

- 以下のファイルをUSBメモリのルートディレクトリに置いてください

## ファイル構成

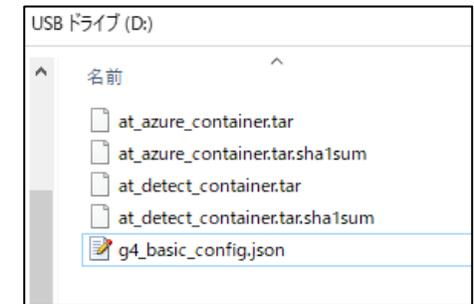
/	
at_detect_container.tar	物体認識コンテナ ビルド済イメージ
at_azure_container.tar	Azureコンテナ ビルド済イメージ
at_detect_container.tar.sha1sum	物体認識コンテナ チェックサム
at_azure_container.tar.sha1sum	Azureコンテナ チェックサム
g4_basic_config.json	IoT Central接続設定ファイル



at\_detect\_container.tar はファイルサイズが 1GBほどあります。

- 以下のファイルはPC上に置いてください

220108_commandlist.txt	実行コマンドリスト
iotc_g4_meetup_handson.json	IoT Centralデバイステンプレート



- 下記の手順に従ってArmadillo-IoT ゲートウェイ G4 の初期化を行ってください

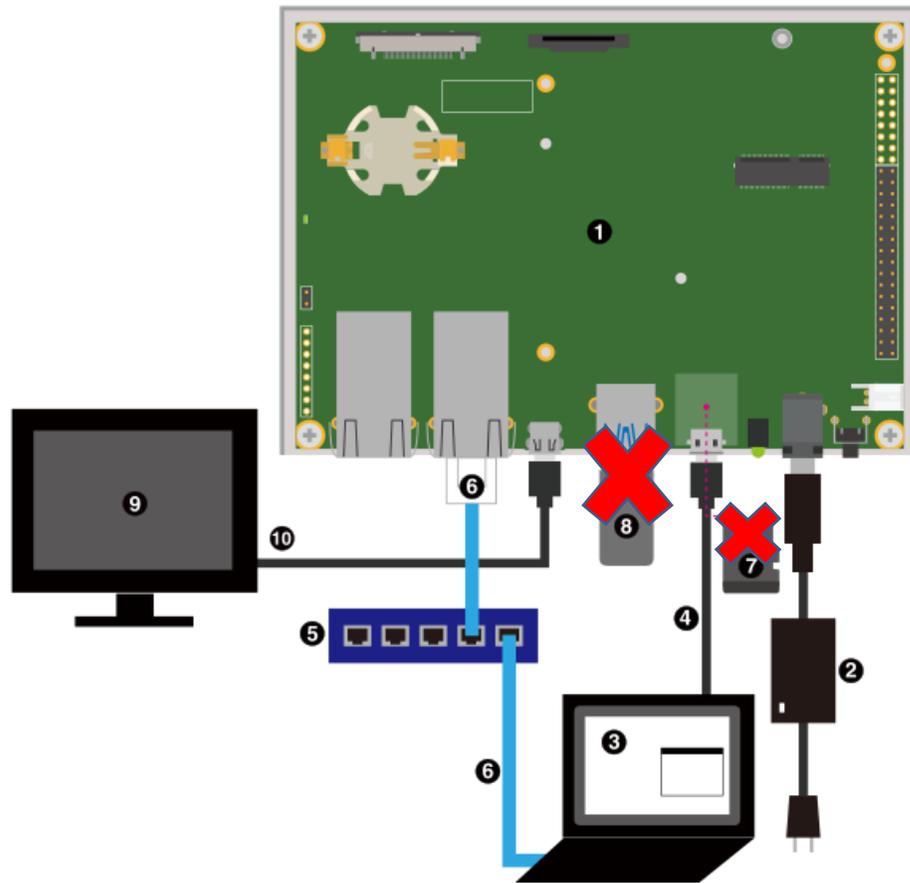


microSDカードが必要です

- Armadillo-IoT ゲートウェイ G4 製品マニュアル  
9.6.2. インストールディスクを使用した初期化

<https://armadillo.atmark-techno.com/resources/documents/armadillo-iot-g4/manuals>

## ■下記の通り周辺機器を接続



- ① Armadillo-IoT ゲートウェイ G4
- ② ACアダプタ (5V/3A)
- ③ 作業用PC
- ④ シリアル通信用USBケーブル(A-microB)
- ⑤ LAN HUB
- ⑥ Ethernetケーブル
- ⑦ ※今回のハンズオン手順では不要です
- ⑧ ※まだ何も挿さないでください
- ⑨ ディスプレイ(HDMI対応)
- ⑩ HDMIケーブル



ACアダプタはハンズオン中に指示があるまで挿さないでください。

# ハンズオンの進め方

- ハンズオン手順中、コンソールに入力するコマンドは **220108\_commandlist.txt** にまとめているので是非コピペにご活用ください。



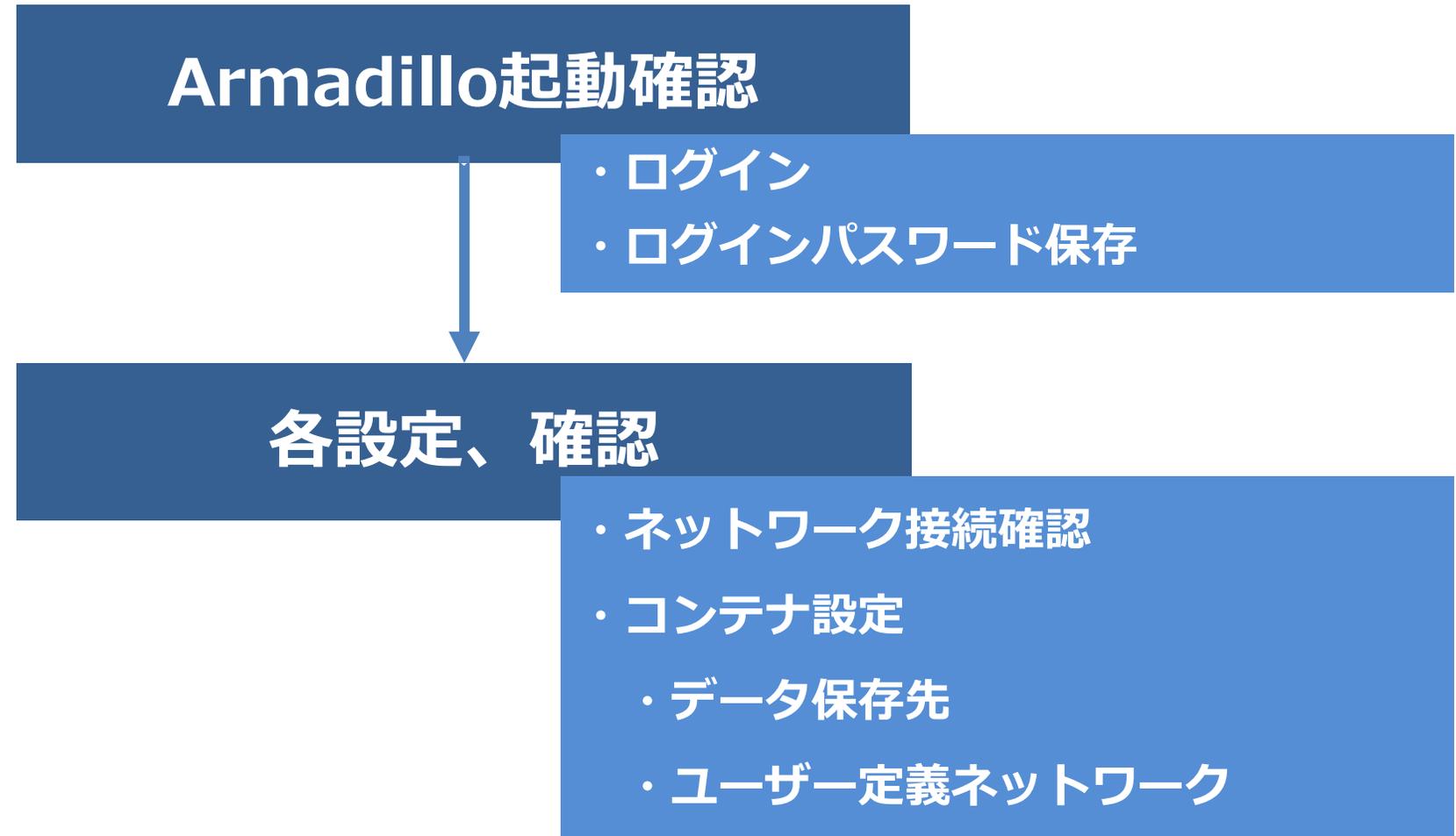
# ハンズオンの進め方

- ハンズオンでは エディタ、ブラウザ、ターミナルソフト、ハンズオン資料など複数のウィンドウを切り替えながら作業を行う事となります。  
スムーズに作業を行うためには、
  - ・モニター上にそれぞれのウィンドウを並べながら作業する
  - ・PCのサブモニターを用意する
  - ・資料を事前に印刷するなどがおすすめです。



# Armadillo-IoT ゲートウェイ G4 セットアップ

# このセクションで実施すること

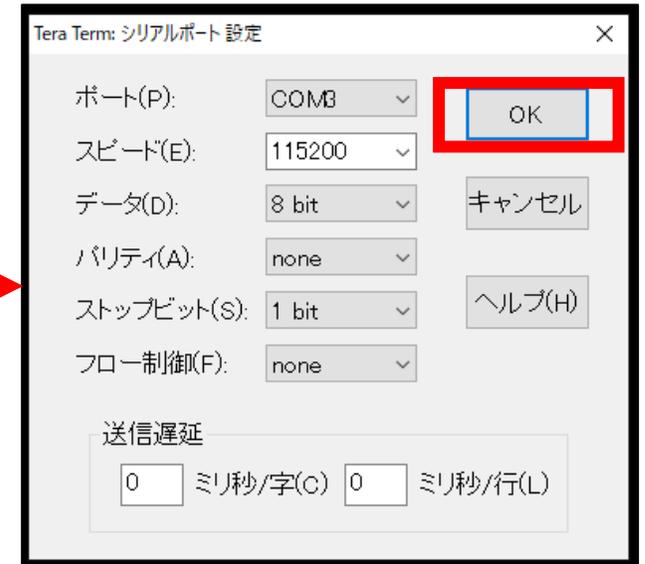
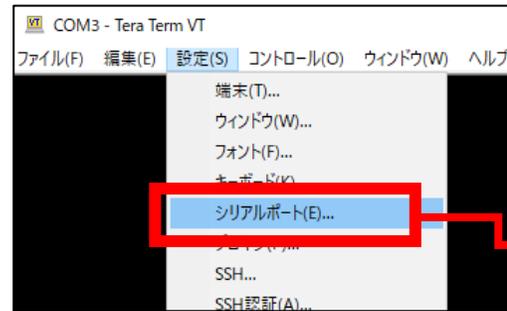


# 起動確認

## ①ターミナルソフト通信設定

項目	設定
転送レート	115,200 bps
データ長	8bit
ストップビット	1bit
パリティ	なし
フロー制御	なし

### TeraTermの場合



## ②ターミナルソフト接続

**TeraTermの場合**

新しい接続(N)...

COM5: Silicon Labs CP210x USB to UART Bridge (COM5)

OK キャンセル ヘルプ(H)

デバイス マネージャー

ポート (COM と LPT)

Silicon Labs CP210x USB to UART Bridge (COM5)

**“Silicon Labs CP210x USB to UART Bridge”  
を選択**

## ③ Armadilloに電源を投入



電源投入前、ArmadilloのUSBポートには何も挿していないことを確認してください。

```
Welcome to Alpine Linux 3.14
Kernel 5.10.35-3-at on an aarch64 (/dev/ttyMXC1)

armadillo login:
```

このログが表示されたら  
起動OK

Armadilloを安全に終了させる場合は、次のコマンドを実行し「reboot: Power down」と表示されたことを確認してから電源を切断してください。

```
armadillo:~# poweroff
:
:
[ 62.862470] reboot: Power down
```

## ■ root ログイン

初回ログインではパスワードの変更が必要です

```
armadillo login: root
```

```
You are required to change your password immediately (administrator enforced).
```

```
New password: 新しいパスワードを入力
```

```
Retype new password: 新しいパスワードを再入力
```

```
Welcome to Alpine!
```

設定するパスワードには以下を含めることができます

- ・ 大文字のアルファベット
- ・ 小文字のアルファベット
- ・ 0から9までの数字
- ・ 記号、句読点など

## ■ 以下のコマンドでパスワードを保存

```
armadillo:~# persist_file /etc/shadow
```



Armadillo Base OS ではルートファイルシステムに overlayfs を採用しているためシステムをOFFすると内容が消えてしまいます。  
persist\_file コマンドを使用すると、ファイル単位で永続化をすることができます。



overlayfs とは:

分離したReadonlyなlower層と、Writableなupper層をマージして1つのファイルシステムに見せる仕組み。Armadilloでは tmpfs(メモリ上に作成するファイルシステム)を upper層にしています。

## ■ 以下のコマンドで確認

```
armadillo:~# ip address
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN qlen 1000
(省略)
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc mq state UP qlen 1000
   link/ether 00:11:0c:00:0a:bb brd ff:ff:ff:ff:ff:ff
   inet 172.16.2.68/16 brd 172.16.255.255 scope global dynamic noprefixroute eth0
       valid_lft 69192sec preferred_lft 69192sec
   inet6 fe80::8480:cb92:98a7:ec8b/64 scope link noprefixroute
       valid_lft forever preferred_lft forever
3: eth1: <NO-CARRIER,BROADCAST,MULTICAST,UP> mtu 1500 qdisc mq state DOWN qlen 1000
   link/ether 00:11:0c:00:0a:bc brd ff:ff:ff:ff:ff:ff
```

“eth0”の方に  
IPアドレスが振られている  
ことを確認

# Podman データ保存先変更

## ■ 以下のコマンドでPodmanのデータ保存先を eMMCに変更

```
armadillo:~# podman_switch_storage --disk  
Creating configuration for persistent container storage  
Create subvolume '/mnt/containers_storage'
```



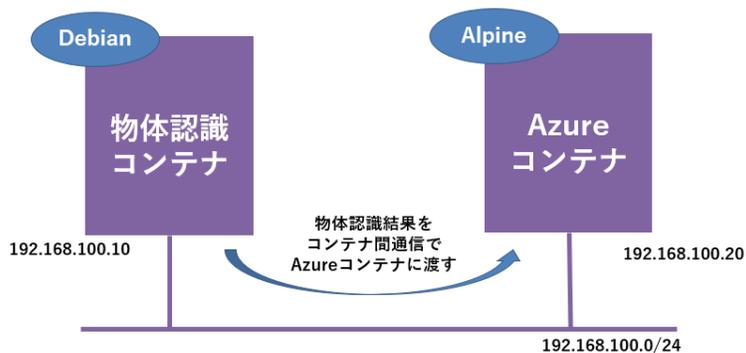
デフォルトでは Podman のデータは tmpfs(メモリ上に作成するファイルシステム)に保存されるため、電源をOFFするとデータが消えてしまいます。

そのため、本ハンズオンでは保存先を eMMC に変更していますが、実際の運用では eMMCの書き込みを最小限にする観点から、運用時は tmpfsに保存するのが適切です。

オプションに `-status` をつけると、現在の設定内容を確認することができます。

```
armadillo:~# podman_switch_storage --status  
Currently in disk mode, run with --tmpfs to switch
```

# コンテナのネットワーク設定



ご利用のネットワーク環境と  
IPアドレスが被っている場合、  
お知らせください

※こちらのコマンドで確認できます

```
armadillo:~# nmcli
```

- 各コンテナに固定のIPアドレスを持たせるため、  
下記のコマンドでユーザー定義のネットワークを作成

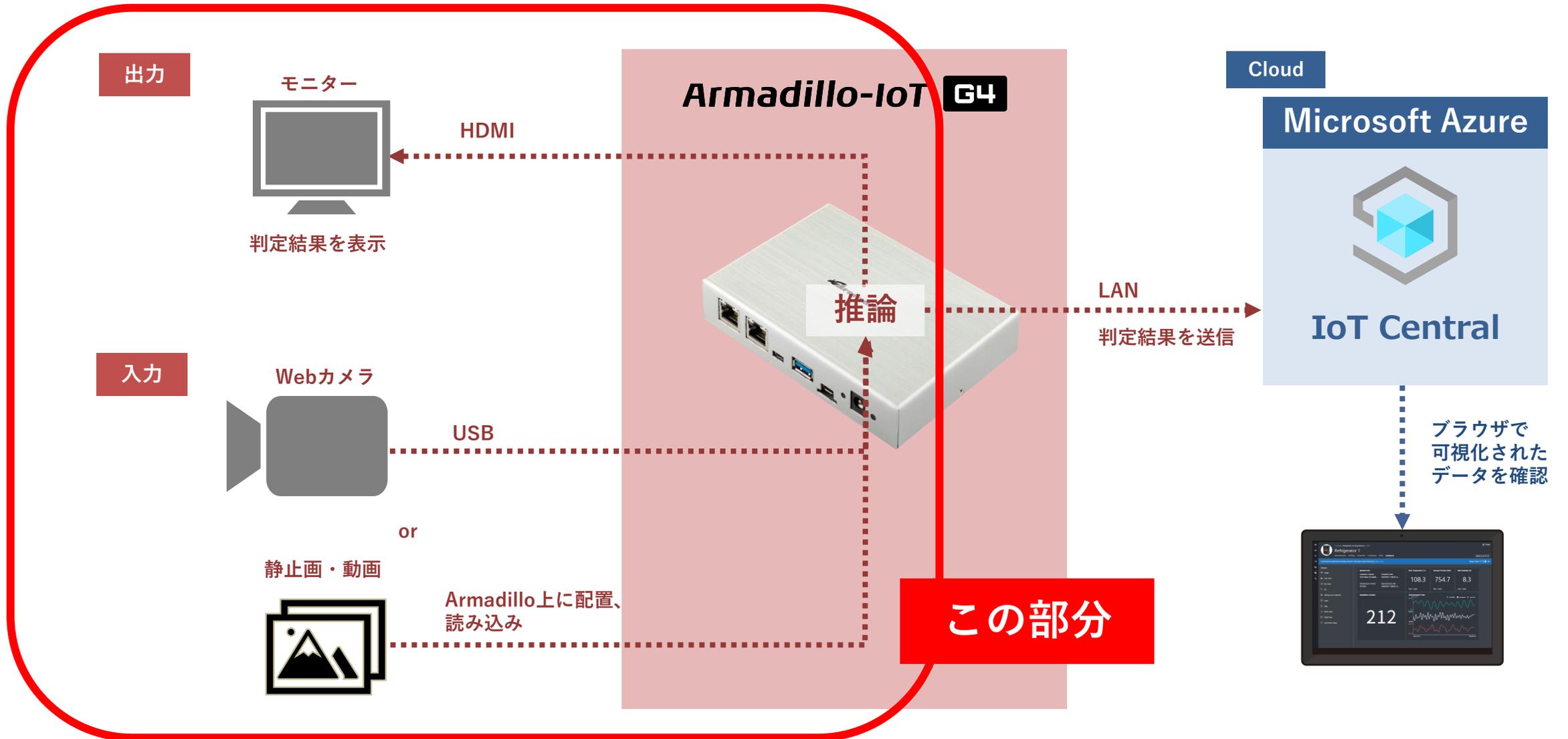
```
armadillo:~# podman network create --subnet=192.168.100.0/24 my_network
```

- ネットワーク設定ファイルが  
再起動で消えてしまわないよう永続化する

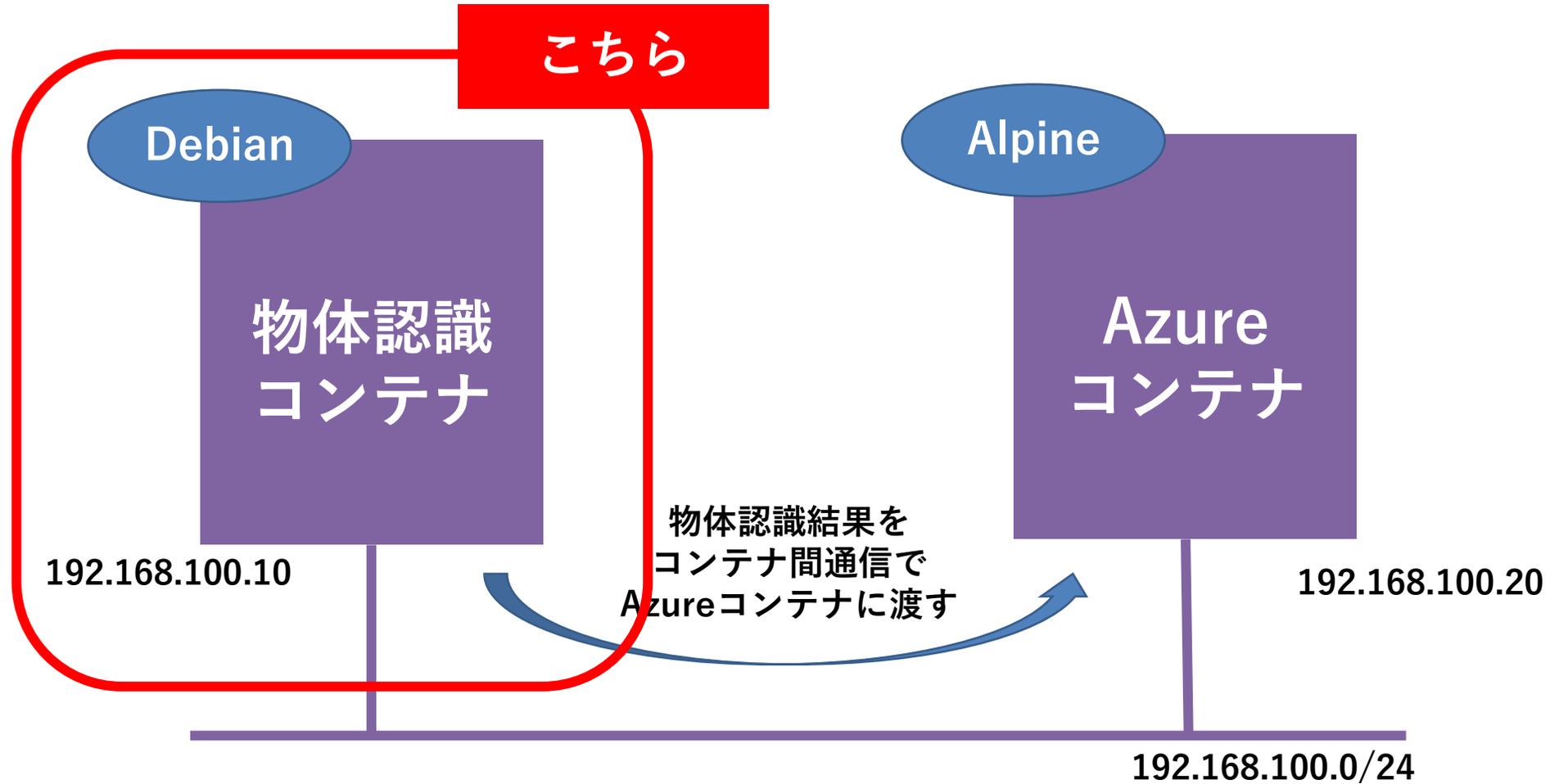
```
armadillo:~# persist_file /etc/cni/net.d/my_network.conflist
```

# 物体認識

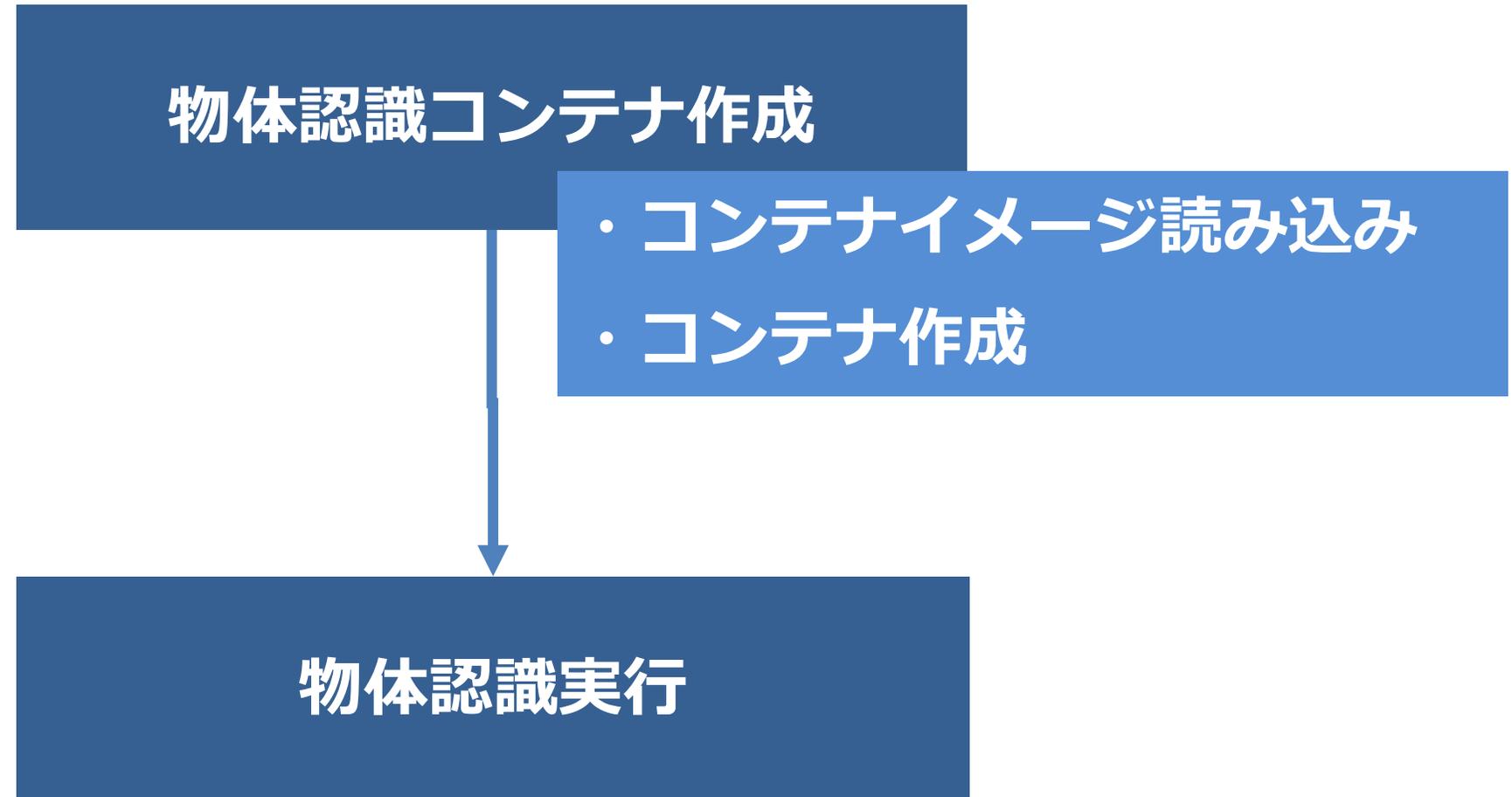
# 物体認識コンテナ



# 物体認識コンテナ



# このセクションで実施すること



# コンテナ作成(Webカメラ有無不問)

## ① USBメモリをArmadilloのUSBコネクタに挿す



USBメモリをArmadilloに挿すと  
このようなログが表示されます。

```
armadillo:~# [16577.714566] usb 2-1: new SuperSpeed Gen 1 USB device number 4 using xhci-hcd
[16577.749031] usb-storage 2-1:1.0: USB Mass Storage device detected
[16577.756937] scsi host0: usb-storage 2-1:1.0
[16578.793638] scsi 0:0:0:0: Direct-Access   BUFFALO  USB Flash Disk  1.00 PQ: 0 ANSI: 6
[16578.806287] sd 0:0:0:0: [sda] 60555264 512-byte logical blocks: (31.0 GB/28.9 GiB)
[16578.815266] sd 0:0:0:0: [sda] Write Protect is off
[16578.820606] sd 0:0:0:0: [sda] Write cache: disabled, read cache: enabled, doesn't support DPO or FUA
[16578.952278] sda: sda1
[16578.958885] sd 0:0:0:0: [sda] Attached SCSI removable disk
```

## ② USBメモリをマウントする

```
armadillo:~# mount /dev/sda1 /mnt
```

## ③ マウント先に移動

```
armadillo:~# cd /mnt
```

## ④ ビルド済みのイメージがあるか確認

```
armadillo:/mnt# ls at_detect_container.tar
at_detect_container.tar
```

# コンテナ作成 (Webカメラ有無不問)

## ⑤ チェックサムを確認

```
armadillo:/mnt# sha1sum -c at_detect_container.tar.sha1sum
at_detect_container.tar: OK
```

## ⑥ イメージを読み込む

2分弱かかります

```
armadillo:/mnt# podman load -i at_detect_container.tar
```

## ⑦ イメージが読み込めたか確認

```
armadillo:/mnt# podman images
REPOSITORY          TAG          IMAGE ID      CREATED      SIZE
localhost/at_detect_container  latest      87a2fcf04d24  13 days ago  1.2 GB
```

## ⑧ ルートに戻る

```
armadillo:/mnt# cd
```

## ⑨ USBメモリをアンマウントし、Armadilloから取り外す

```
armadillo:~# umount /mnt
```

Webカメラなしの場合 → p.37 にジャンプ

# コンテナ作成 (Webカメラあり)



## ⑩ WebカメラをUSBコネクタに挿す

## ⑪ Webカメラ認識状態を確認する

```
armadillo:~# ls -la /dev/v4l/by-id
```

```
total 0
```

```
drwxr-xr-x 2 root root 80 Jan  1 09:04 .
```

```
drwxr-xr-x 4 root root 80 Jan  1 09:04 ..
```

```
lrwxrwxrwx 1 root root 12 Jan  1 09:04 usb-046d_HD_Pro_Webcam_C920_3E1FF2AF-video-index0 -> ../../video3
```

```
lrwxrwxrwx 1 root root 12 Jan  1 09:04 usb-046d_HD_Pro_Webcam_C920_3E1FF2AF-video-index1 -> ../../video4
```

xxx-index0 のリンク先パスを確認  
※この場合は /dev/video3

# コンテナ作成 (Webカメラあり)

## ⑫ コンテナを作成する

```
armadillo:~# podman run -itd --name=detect ¥
--env=XDG_RUNTIME_DIR=/tmp ¥
--env=LD_LIBRARY_PATH=/opt/firmware/usr/lib/aarch64-linux-gnu ¥
--env=QT_QPA_PLATFORM=wayland ¥
--device=/dev/tty1 ¥
--device=/dev/input ¥
--device=/dev/dri ¥
--device=/dev/galcore ¥
--volume /run/udev:/run/udev:ro ¥
--cap-add "CAP_SYS_TTY_CONFIG" ¥
--device=/dev/video3:/dev/video3 ¥
--device=/dev/mxc_hantro ¥
--device=/dev/mxc_hantro_vc8000e ¥
--device=/dev/ion ¥
--volume=/opt/firmware:/opt/firmware:ro ¥
--network=my_network --ip 192.168.100.10 ¥
localhost/at_detect_container /bin/bash
```

・ Webカメラあり

この行を p.34 ⑪で確認したデバイスのパスに合わせて適宜修正してください。

/dev/video2 の場合 :  
--device=/dev/video2:/dev/video3 ¥

/dev/video3 の場合 :  
--device=/dev/video3:/dev/video3 ¥

# コンテナ作成(Webカメラあり)



## ⑬ コンテナが作成できたか確認する

```
armadillo:~# podman ps -a
```

CONTAINER ID	IMAGE	COMMAND	CREATED	STATUS	PORTS	NAMES
21571d0fc440	localhost/ at_detect_container:latest	/bin/bash	8 weeks ago	Up 2 hours ago		detect

p.39 にジャンプ

作成したコンテナは下記のコマンドで削除することができます。

```
armadillo:~# podman rm detect
```

# コンテナ作成 (Webカメラなし)

## ⑩ コンテナを作成する

Webカメラなし

```
armadillo:~# podman run -itd --name=detect ¥
--env=XDG_RUNTIME_DIR=/tmp ¥
--env=LD_LIBRARY_PATH=/opt/firmware/usr/lib/aarch64-linux-gnu ¥
--env=QT_QPA_PLATFORM=wayland ¥
--device=/dev/tty1 ¥
--device=/dev/input ¥
--device=/dev/dri ¥
--device=/dev/galcore ¥
--volume /run/udev:/run/udev:ro ¥
--cap-add "CAP_SYS_TTY_CONFIG" ¥
--device=/dev/mxc_hantro ¥
--device=/dev/mxc_hantro_vc8000e ¥
--device=/dev/ion ¥
--volume=/opt/firmware:/opt/firmware:ro ¥
--network=my_network --ip 192.168.100.10 ¥
localhost/at_detect_container /bin/bash
```

接  
続  
な  
し

# コンテナ作成(Webカメラなし)



## ⑪ コンテナが作成できたか確認する

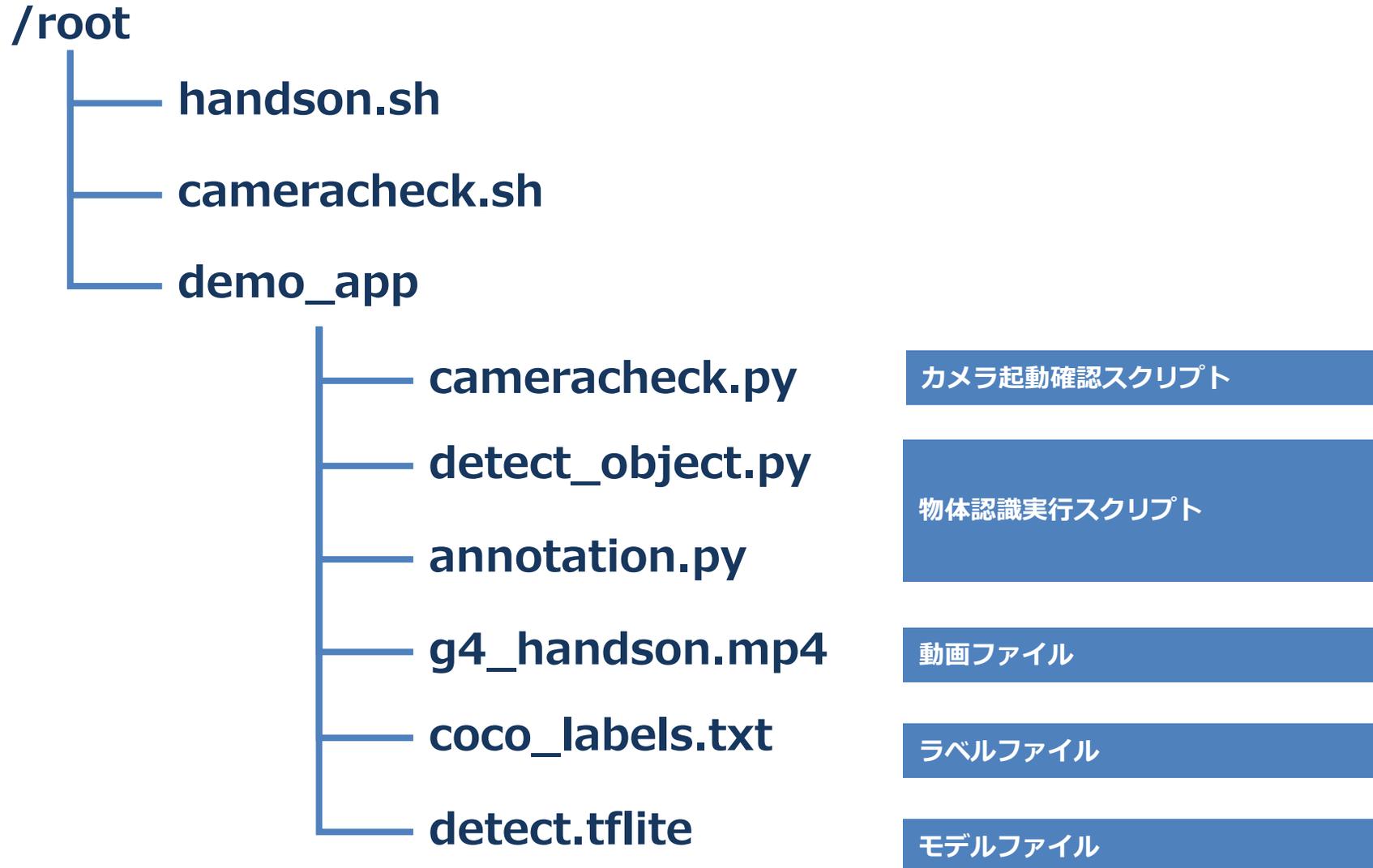
```
armadillo:~# podman ps -a
```

CONTAINER ID	IMAGE	COMMAND	CREATED	STATUS	PORTS	NAMES
21571d0fc440	localhost/ at_detect_container:latest	/bin/bash	8 weeks ago	Up 2 hours ago		detect

作成したコンテナは下記のコマンドで削除することができます。

```
armadillo:~# podman rm detect
```

# コンテナ内コード構成



# カメラ/HDMI出力動作確認

Webカメラありの場合のみ実行

## ① コンテナの中に入る

```
armadillo:~# podman exec -it detect /bin/bash
root@f6107fd74be7:/#
```

## ② スクリプト実行

```
root@f6107fd74be7:~# /root/cameracheck.sh
```

## ③ HDMIで接続されたモニタにWebカメラから取得した動画が表示されることを確認



エラーが発生する場合、今回は「Webカメラなし」パターンでの実行をお願いします。  
「Webカメラあり」のコンテナでも「カメラなし」パターンは動作するので、コンテナを作り直す必要はありません。

## ④ コンテナから出る

```
root@f6107fd74be7:/# exit
exit
armadillo:~#
```

## ■ Webカメラあり

```
armadillo:~# podman exec -d detect /root/handson.sh --camera
```

## ■ Webカメラなし(動画を使用)

```
armadillo:~# podman exec -d detect /root/handson.sh
```

実行イメージ

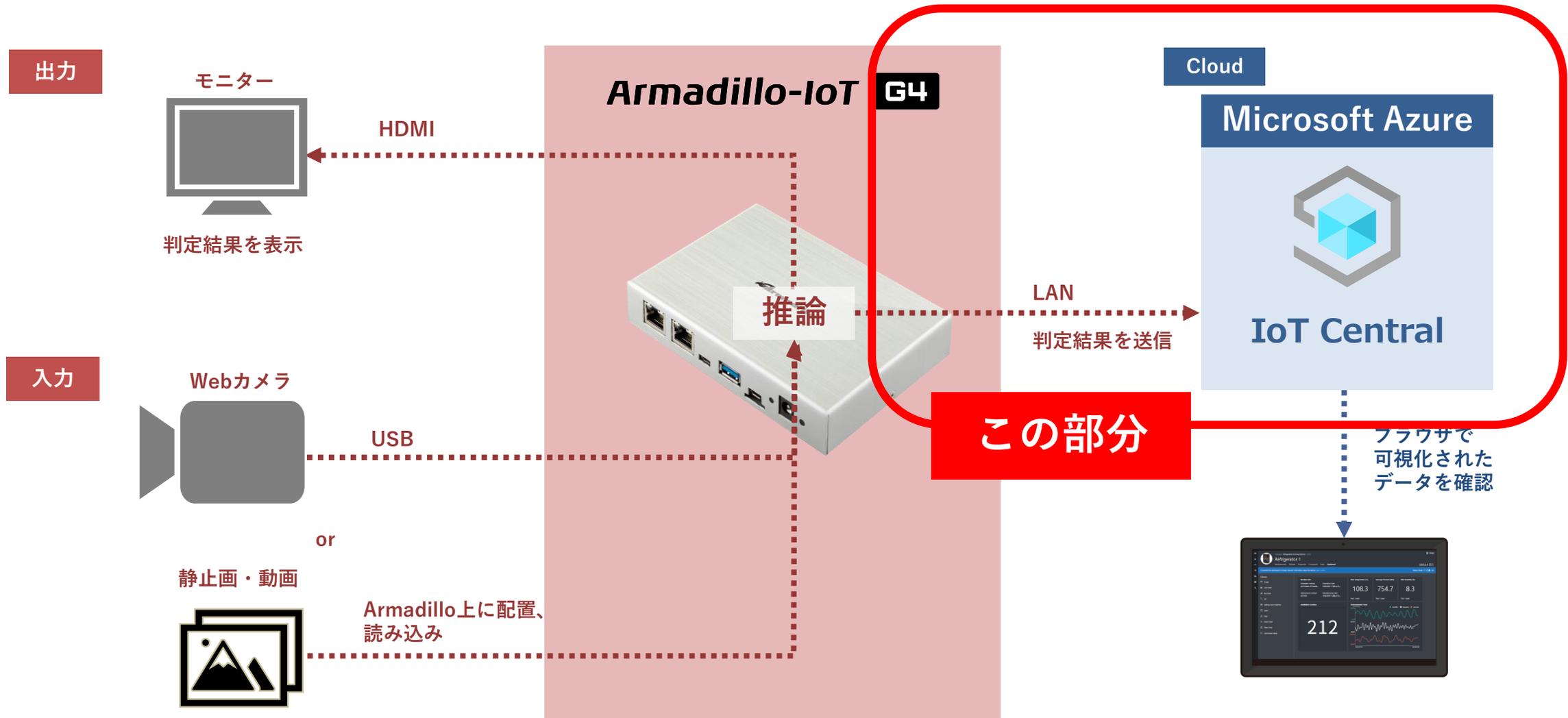


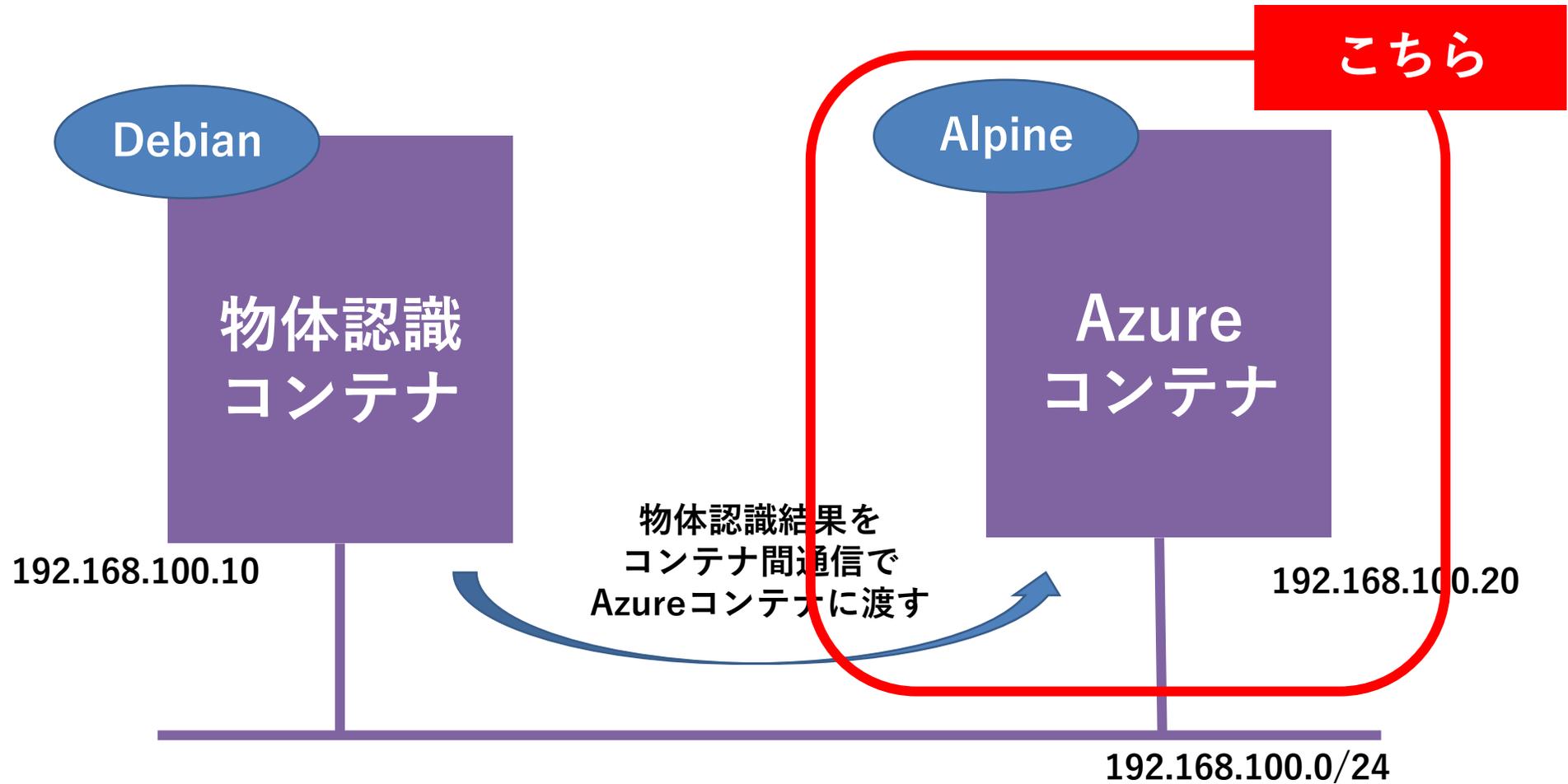
次のセクションに移るため、下記のコマンドで物体認識コンテナを再起動します。

```
armadillo:~# podman restart detect
```

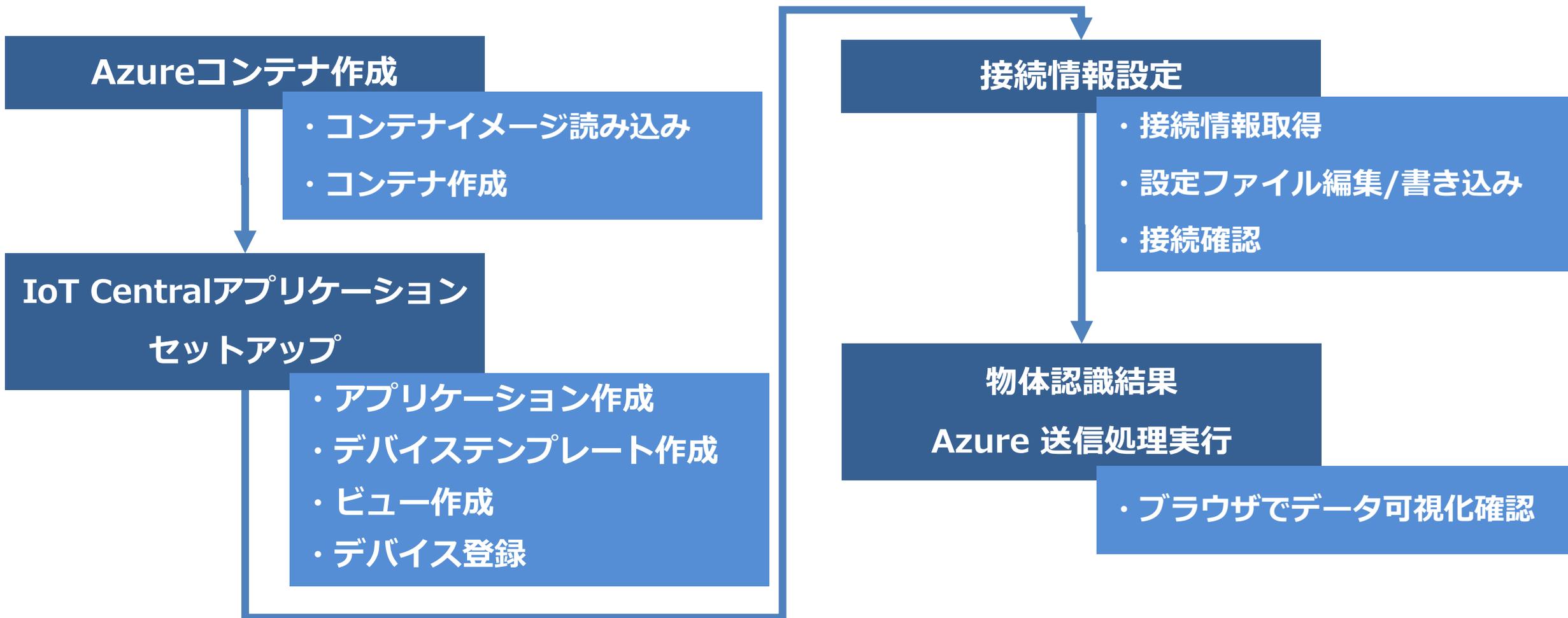
# 物体認識結果 Azure IoT Central送信

# Azureコンテナ





# このセクションで実施すること



# コンテナ作成



CON4

U  
S  
B  
メ  
モ  
リ

① ArmadilloからWebカメラを取り外し、USBメモリを挿す

② USBメモリをマウントする

```
armadillo:~# mount /dev/sda1 /mnt
```

③ マウント先に移動

```
armadillo:~# cd /mnt
```

④ ビルド済みのイメージがあるか確認

```
armadillo:/mnt# ls at_azure_container.tar  
at_azure_container.tar
```

⑤ チェックサムを確認

```
armadillo:/mnt# sha1sum -c at_azure_container.tar.sha1sum  
at_azure_container.tar: OK
```



CON4

U  
S  
B  
メ  
モ  
リ

## ⑥ イメージを読み込む

```
armadillo:/mnt# podman load -i at_azure_container.tar
```

## ⑦ イメージが読み込めたか確認

```
armadillo:/mnt# podman images
```

REPOSITORY	TAG	IMAGE ID	CREATED	SIZE
localhost/at_azure_container	latest	c6ca6cf31232	46 hours ago	91.6 MB
localhost/at_detect_container	latest	87a2fcf04d24	13 days ago	1.2 GB

## ⑧ ルートに戻る

```
armadillo:/mnt# cd
```

## ⑨ USBメモリをアンマウントし、Armadilloから取り外す

```
armadillo:~# umount /mnt
```

## ⑩ コンテナを作成する

```
armadillo:~# podman run -itd --name=azure --env=NUMBER=ALGYAN0108 ¥  
--net=my_network --ip=192.168.100.20 ¥  
localhost/at_azure_container /bin/sh
```

## ⑪ コンテナが作成できたか確認する

```
armadillo:~# podman ps -a
```

CONTAINER ID	IMAGE	COMMAND	CREATED	STATUS	PORTS	NAMES
d6c973e30dc6	localhost/at_detect_container	/bin/bash	2 days ago	Up About an hour ago		detect
6c9b7777673c	localhost/at_azure_container:latest	/bin/sh	2 days ago	Up About an hour ago		azure

# コンテナ内コード構成

## Azure-IoT-samples

### └─ Armadillo-IoT\_GW



# IoT Central アプリケーション作成

## ① 下記の URL にアクセス

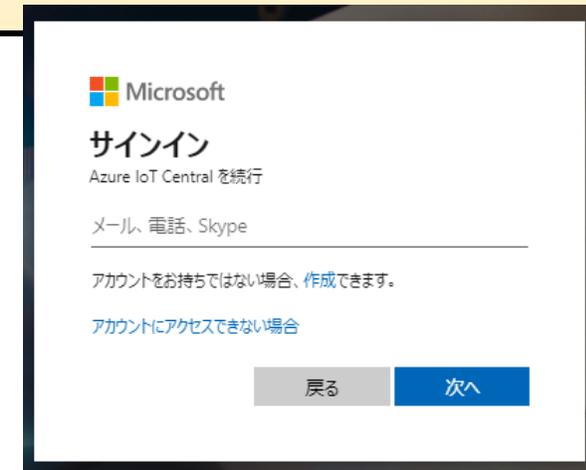
■ Azure IoT Central – IoT アプリケーションのビルド

<https://apps.azureiotcentral.com/build>

## ② カスタムアプリ「アプリの作成」を選択



次の画面でサインインを求められたら、  
お持ちの Azure アカウントでログインしてください



# IoT Central アプリケーション作成

1アカウントにつき、1つまで7日間無料のアプリケーションを作成することができます

Azure IoT Central

ビルド > 新しいアプリケーション

新しいアプリケーション **カスタム**

いくつかの簡単な質問に回答すると、アプリを起動して実行できます。

お客様のアプリについて

アプリケーション名\* ① **1.任意の名前**  
G4-ALGYAN-Meetup

URL\* ① **2.任意のURL**  
g4-algyan-meetup

アプリケーションテンプレート\* ①  
カスタム アプリケーション

料金プラン **3.料金プランを選択**

- 無料  
7日間契約なしでお試ください  
5台の無料デバイス
- Standard 0  
1日数件のメッセージを送信するデバイスの場合  
2台の無料デバイス 400個のメッセージ/mo
- Standard 1  
1時間ごとに数件のメッセージを送信するデバイスの場合  
2台の無料デバイス 5,000個のメッセージ/mo
- Standard 2 (最も人気)  
数分ごとにメッセージを送信するデバイスの場合  
2台の無料デバイス 30,000個のメッセージ/mo

●有料を選択した場合、サブスクリプション情報を入力

課金情報

ディレクトリ\* ①  
既定のディレクトリ

Azure サブスクリプション\* ① サブスクリプションをお持ちではない場合、[サブスクリプションを作成してください](#)

場所\* ① **任意の場所**  
東日本

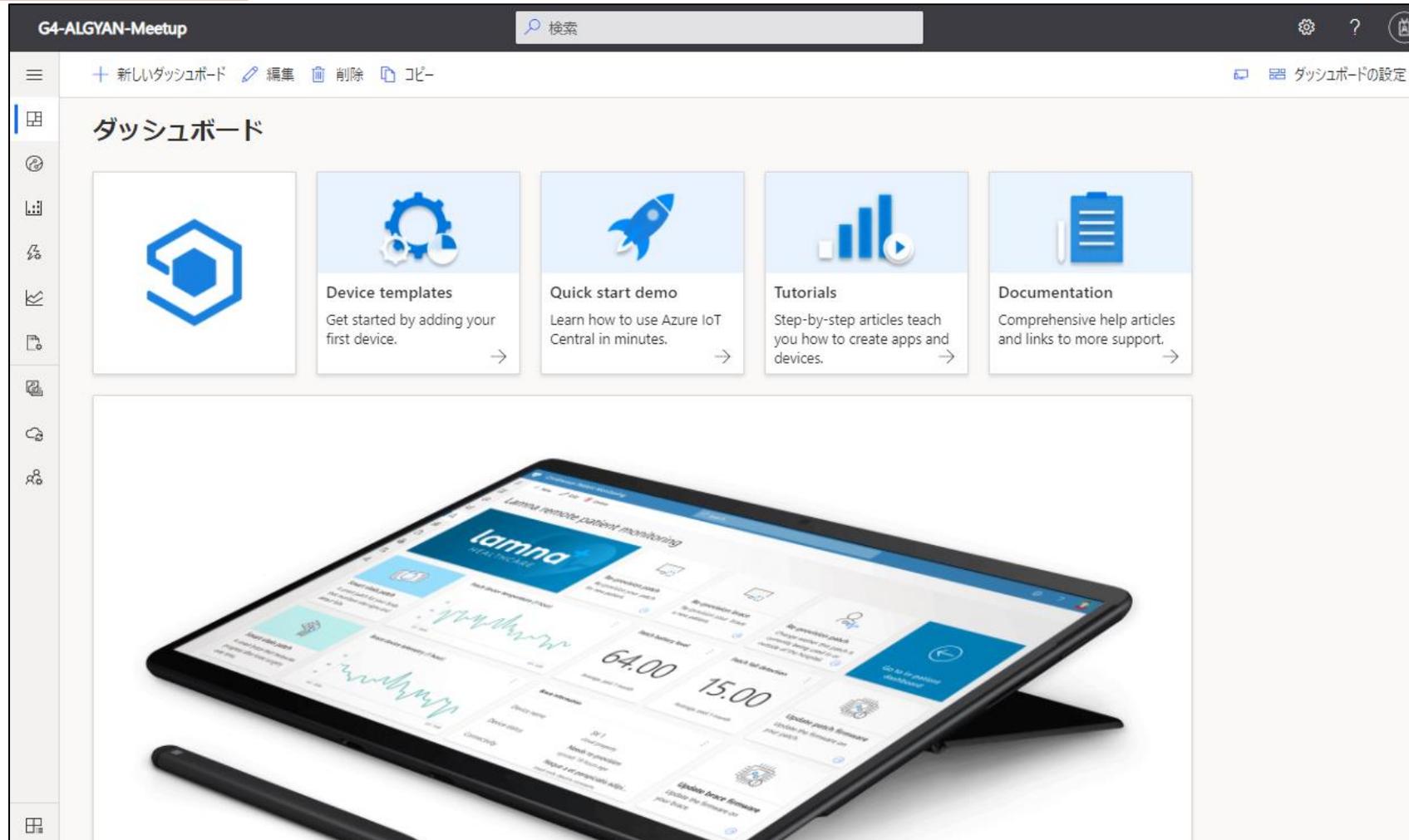
[作成] をクリックすると、[サブスクリプション契約](#)と[プライバシーに関する声明](#)に同意したものと見なされます。価格、キャンセル料、支払い、データ保有に関する契約の規定は、「Free」には適用されません。「Standard」プランには Azure サブスクリプションが必要であり、[Azure サブスクリプション](#)に適用される条件でこのサービスのライセンスが交付されることに同意いただきます。

[作成] [キャンセル]

全ての情報を入力したらクリック

# IoT Central アプリケーション作成

## アプリケーションTOP画面



# デバイステンプレート作成

デバイステンプレート： デバイスが IoT Central とどのようにやり取りするか記述されたモデル

ナビゲーションメニューを開きます

種類を選択

確認

デバイステンプレート > 新しいデバイステンプレートの作成

**種類の選択**

デバイステンプレートはブループリントのようなものです。アプリケーションに接続するデバイスの特性と動作を定義します。

カスタム デバイス テンプレートの作成

IoT デバイス  
機能モデルをインポートするか、初めから機能を作成します。

Azure IoT Edge  
Azure IoT Edge とゲートウェイのシナリオを備えたテンプレートを作成します。

おすすめのデバイステンプレート

次へ: カスタマイズ

# デバイステンプレート作成

デバイステンプレート > 新しいデバイステンプレートの作成

種類を選択  
カスタマイズ  
確認

カスタマイズ

デバイステンプレート名\*

G4Meetup

これはゲートウェイデバイスです。詳細をご確認ください。🔗

デバイステンプレート名は「G4Meetup」と入力

[前へ]をクリックします。 **次 レビュー**



デバイステンプレート > 新しいデバイステンプレートの作成

種類を選択  
カスタマイズ  
確認

確認

空のテンプレートを作成して、機能とインターフェイスを追加できるようにします。インターフェイスは、インポートすることも最初から作成することもできます。完了すると、テンプレートを公開してデバイスに接続できるようになります。

基本情報

デバイステンプレートの種類	IoT デバイス
デバイステンプレート名	G4Meetup

デバイステンプレートの種類が IoTデバイス、デバイステンプレート名が「G4Meetup」になっていることを確認してください

[前へ]をクリックします。 **作成**

# モデルの作成

バージョン テストデバイスの管理 公開 名前の変更 削除

デバイステンプレート > G4Meetup

G4Meetup

モデルの作成  
カスタムモデルを最初から作成するか、既存のモデルをインポートします。

カスタムモデル  
空のモデルで開始し、最初からデバイスを作り上げます。

モデルのインポート  
まずモデル ファイルをインポートします。

ファイル選択画面が表示されるので  
予めダウンロードした  
**iotc\_g4\_meetup\_handson.json**  
を選択



【注意】  
ダウンロードしたファイルにはjsonファイル  
が2つあります。指定するファイルを間違え  
ないように注意してください。

iotc\_g4\_meetup\_handson.json には  
デバイスが IoT Central に送信する  
テレメトリ等の定義を記述しています。

# デバイステンプレート作成

## モデルインポート後画面

バージョン テストデバイスの管理 公開 名前の変更 削除

デバイステンプレート > G4Meetup > モデル > G4Meetup

 **G4Meetup**  
アプリケーションが更新されました: Never インターフェイスが公開されました: Never

> G4Meetup **ルート** ドラフト

このデバイスモデルに固有の機能を追加します。 [詳細情報](#)

保存 + 機能の追加 IDの編集 エクスポート 削除 ... DTDLの編集

表示名	名前 *	機能の種類 * ①	セマンティックの種類 ①		
Serial Number	serialNumber	Property	なし	×	▽
Most detected object	detectObject	Telemetry	なし	×	▽
Total number of detecte...	totalNumDetectObject	Telemetry	なし	×	▽

+ 機能の追加

# デバイステンプレート作成

## 各パラメータに関して

The screenshot shows the configuration page for a device template named 'G4Meetup'. At the top, there are navigation links: バージョン, テスト デバイスの管理, 公開, 名前の変更, and 削除. Below that, the breadcrumb is 'デバイステンプレート > G4Meetup > モデル > G4Meetup'. The main header shows 'G4Meetup' with a status 'アプリケーションが更新されました: Never' and 'インターフェイスが公開されました: Never'. There are buttons for 'ルート' and 'ドラフト'. A description says 'このデバイス モデルに固有の機能を追加します。詳細情報'. Below are action buttons: 保存, 機能の追加, ID の編集, エクスポート, 削除, and DTDL の編集. A table lists the parameters:

表示名	名前 *	機能の種類 *	セマンティックの種類
Serial Number	serialNumber	Property	なし
Most detected object	detectObject	Telemetry	なし
Total number of detecte...	totalNumDetectObject	Telemetry	なし

At the bottom, there is a '+ 機能の追加' button.

デバイス固有の番号。  
今回のハンズオンではコンテナ作成時に  
指定しています。  
IoT Centralに接続した後に1回だけ送信します。

物体検出結果。Object型で定義しています。  
5秒ごとに一番多く検出した物体名・その数を  
送信します。

The screenshot shows the DTDL configuration window with the following table:

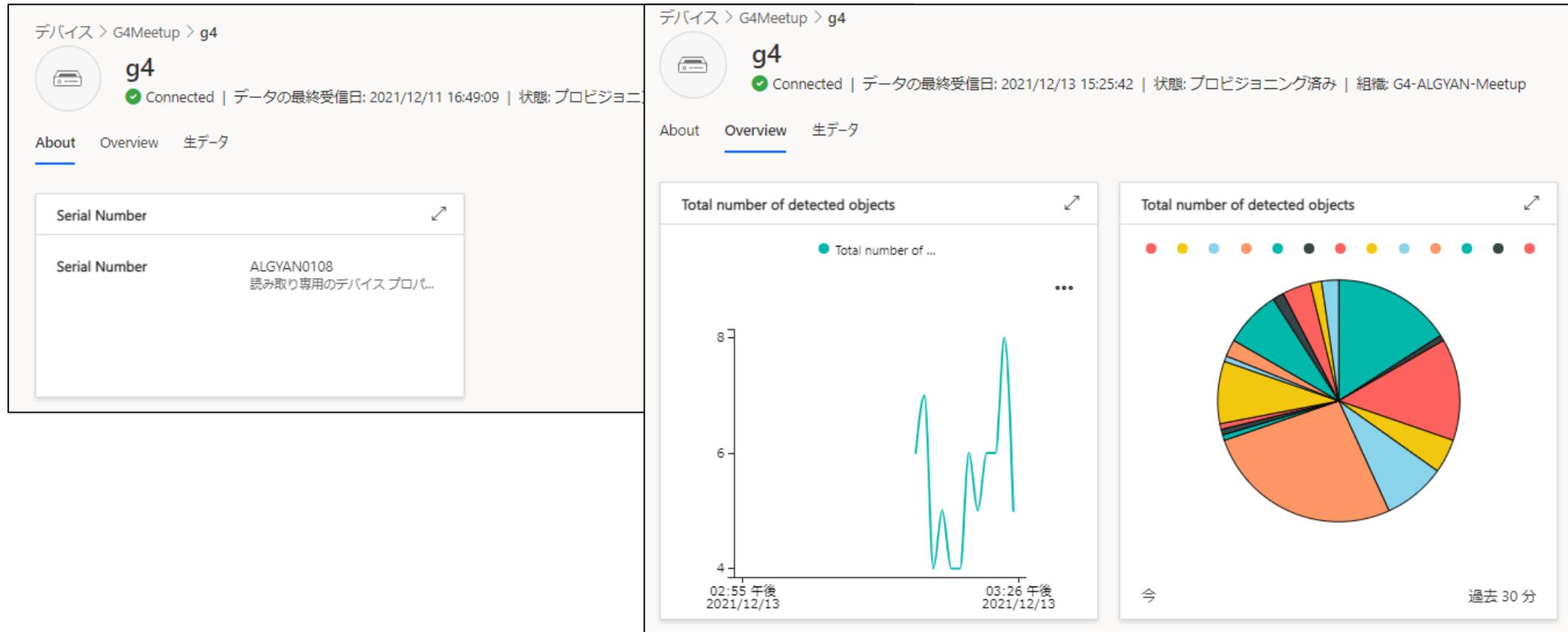
表示名	名前 *	スキーマ *
object name	objectName	String
number of detections	numObject	Integer

There is a '+ 追加' button at the bottom.

物体検出総数。5秒ごとに検出した物体の総数  
を送信します。

# ビュー作成

## ビュー完成図



# ビュー作成

## 規定のビューの作成

デバイステンプレート > G4Meetup > ビュー > 新規

G4Meetup  
アプリケーションが更新されました: Never インターフェイスが公開されました: Never

モデル

- G4Meetup
- クラウドのプロパティ
- 生データ
- カスタマイズ
- ビュー**

新しいビューを追加するための選択

**デバイスとクラウドのデータの編集**  
このビューを使用して、機能モデルとソリューション モデルで定義されているプロパティを編集および表示するためのフォームを作成します

**デバイスの視覚化**  
このビューで、数々のグラフ、ゲージ、メトリック タイルを使用した、表現力豊かな機能モデルのダッシュボードを作成します

**既定のビューの生成**  
直感的なダッシュボード操作でデバイス情報の表示をすばやく開始するために、既定のデバイス ビューを生成します

生成する適用可能なビューを選択します。

コマンド - デバイスへのディスパッチを可能にするデバイス コマンドを含むビューを提供します。  
 オン

概要 - デバイス テレメトリを含むビューを提供し、グラフとメトリックを表示します。  
 オン

詳細情報 - デバイス情報を含むビューを提供し、そのプロパティを表示します。  
 オン

**既定のダッシュボード ビューの生成**

デバイステンプレート > G4Meetup > モデル > G4Meetup

G4Meetup  
アプリケーションが更新されました: Never インターフェイスが公開されました: Never

モデル

- G4Meetup
- クラウドのプロパティ
- 生データ
- カスタマイズ
- ビュー**

Overview

About

Serial Number

Most detected obje

Total number of det

Overview, About ビューが自動的に生成されます。

# ビュー作成

## 「About」ビュー

デバイステンプレート > G4Meetup > モデル > G4Meetup

G4Meetup  
アプリケーションが更新されました: Never インターフェイスが公

モデル <

G4Meetup

クラウドのプロパティ

生データ ↗

カスタマイズ

ビュー

Overview ↗

About

G4Meetup ルー

このデバイス モデル

保存 + 機能

表示名

Serial Number

Most detected obje

Total number of det

ビューの編集 <

ビューの設定

ビュー名 \* ①

About

タイトルの追加

Serial Number

内容の確認だけでOKです。

プロパティタイルの構成 ×

タイトル ①

Serial Number ×

プロパティ

Serial Number

表示: テキスト ⚙ ×

+ 機能

クラウドプロパティ

+ 機能

タイトルの形式

テキスト サイズ

10.5 pt

小数点以下 ①

更新 キャンセル

# ビュー作成

## 「Overview」ビュー

デバイステンプレート > G4Meetup > モデル > G4Meetup

**G4Meetup**  
アプリケーションが更新されました: Never インターフェイスが公開

モデル

- G4Meetup
- クラウドのプロパティ
- 生データ
- カスタマイズ

ビュー

- Overview**
- About

戻る 保存 削除 コピー プレビュー デバイスの構成

### ビューの編集

ビューの設定

ビュー名\* ①  
Overview

### タイトルの追加

ビジュアルから始める ...

タイトルに表示するビジュアルの種類を選択し、[タイトルの追加]をクリックします(または、キャンバスにドラッグアンドドロップするだけです)。新しいタイトルの[設定]アイコンをクリックして、コンテンツまたはデバイスデータを追加します。

Total number of detected objects

### グラフの構成

タイトル\* ①  
Total number of detected objects

凡例の表示 ①  
 オン

X軸の表示 ①  
 オン

Y軸の表示 ①  
 オン

表示範囲 ①  
過去 30分 平均

間隔 ①  
1分 合計

テレメトリ

Total number of de...  
平均

更新 キャンセル

### グラフの構成

タイトル\* ①  
Total number of detected objects

凡例の表示 ①  
 オン

X軸の表示 ①  
 オン

Y軸の表示 ①  
 オン

表示範囲 ①  
過去 30分

間隔 ①  
1分

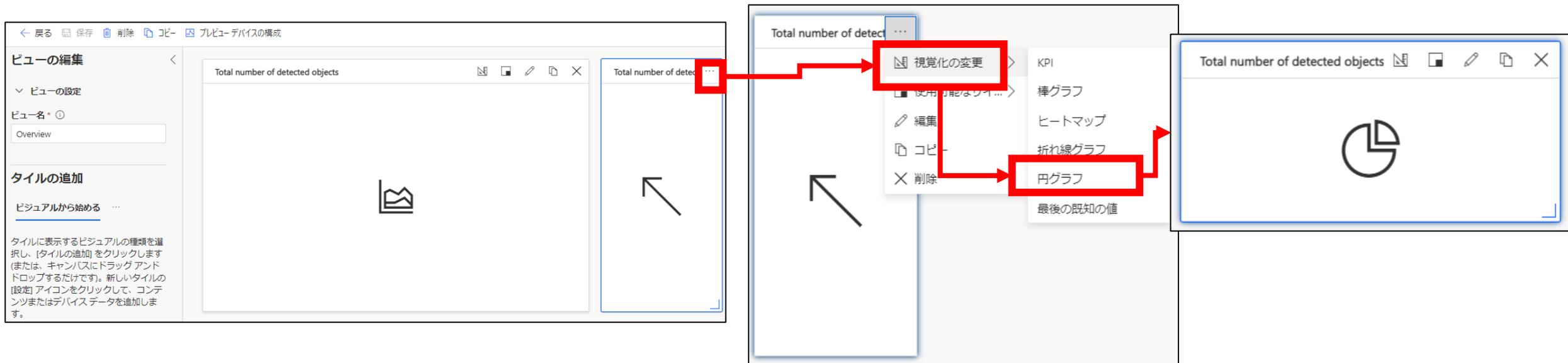
テレメトリ

Total number of de...  
最大

更新 キャンセル

# ビュー作成

## 「Overview」ビュー



# ビュー作成

## 「Overview」ビュー

「Overview」ビュー

グラフの構成

タイトル\* ①  
Ratio of detected objects

凡例の表示 ①  
 オン

X 軸の表示 ①  
 オン

Y 軸の表示 ①  
 オン

表示範囲 ①  
過去 30 分

▼ テレメトリ

Total number of de...  
平均

+ 機能

更新 キャンセル

グラフの構成

タイトル\* ①  
Ratio of detected objects

凡例の表示 ①  
 オン

X 軸の表示 ①  
 オン

Y 軸の表示 ①  
 オン

表示範囲 ①  
過去 30 分

▼ テレメトリ

+ 機能

グラフの構成

タイトル\* ①  
Ratio of detected objects

凡例の表示 ①  
 オン

X 軸の表示 ①  
 オン

Y 軸の表示 ①  
 オン

表示範囲 ①  
過去 30 分

▼ テレメトリ

機能を選択する

number of detections

object name

Total number of detected objects

グラフの構成

タイトル\* ①  
Ratio of detected objects

凡例の表示 ①  
 オン

X 軸の表示 ①  
 オン

Y 軸の表示 ①  
 オン

表示範囲 ①  
過去 30 分

▼ テレメトリ

object name  
カウント

+ 機能

更新 キャンセル

# ビュー作成

## 「Overview」ビュー

タイトル

② ①

「保存」→「戻る」の順にクリック

ビューの編集

ビューの設定

ビュー名\* ①

Overview

タイトルの追加

ビジュアルから始める ...

タイトルに表示するビジュアルの種類を選択し、[タイトルの追加]をクリックします (または、キャンバスにドラッグアンドドロップするだけです)。新しいタイトルの[設定]アイコンをクリックして、コンテンツまたはデバイスデータを追加します。

Total number of detected objects

Total number of detected objects

ウィンドウサイズ調整の要領で

各タイトルの大きさの調整をすることができます

# デバイステンプレート公開

デバイスを接続する前に、デバイステンプレートを「公開(publish)」する必要があります。

このデバイス テンプレートのアプリケーションへの発行

テンプレートのビルドが完了し、実際のデバイスやシミュレートされたデバイスを作成する準備ができたなら、デバイステンプレートを発行します。接続されているデバイスがある場合、デバイステンプレートを発行すると、それらのデバイスに最新の変更がプッシュされます。

変更が加えられて、発行される内容を次に示します。

項目	状態
デバイステンプレート	はい
インターフェイス	はい
カスタマイズ	いいえ
クラウドのプロパティ	いいえ
ビュー	はい

公開

公開完了

このデバイス テンプレートは公開されています。公開されている機能を編集すると、ダッシュボード、ジョブ、ルール、またはデータ エクスポートで破壊的変更が発生するおそれがあります。 [詳細情報](#)

デバイステンプレート > G4Meetup > モデル > G4Meetup

G4Meetup

アプリケーションが更新されました: 1分... インターフェイスが公開されました: 1分以内

モデル

- G4Meetup
- クラウドのプロパティ
- 生データ
- カスタマイズ
- ビュー
  - Overview
  - About

G4Meetup ルート 発行済み

このデバイス モデルに固有の機能を追加します。 [詳細情報](#)

保存 + 機能の追加 IDの編集 エクスポート 削除 ...

表示名	名前*	機能の種類
Serial Number	serialNumber	Property
Most detected object	detectObject	Telemetry
Total number of detected objects	totalNumDetectObject	Telemetry

# デバイス作成

接続情報を取得するため、デバイスを登録します。

The image shows a two-part interface for creating a device. On the left, a sidebar menu has 'デバイス' (Devices) selected. The main area shows a list of device groups, with 'G4Meetup' highlighted. A red box around the '+ 新規' (New) button in the top right of the main area has an arrow pointing to the right-hand form. The form is titled '新しいデバイスを作成します' (Create New Device) and contains the following fields:

- デバイス名 \* ①: g4
- デバイス ID \* ①: g4
- 組織 \* ①: G4-ALGYAN-Meetup
- デバイス テンプレート \*: G4Meetup

Below the form, there is a question: 'このデバイスをシミュレートしますか?' (Do you want to simulate this device?). The answer is 'いいえ' (No), indicated by a radio button. At the bottom right, there are two buttons: '作成' (Create) and 'キャンセル' (Cancel). A red box highlights the '作成' button.

**デバイス名、デバイスIDは「g4」と入力**

# デバイス作成

## デバイス作成後画面



デバイス

テンプレートをフィルタリングして...

すべてのデバイス

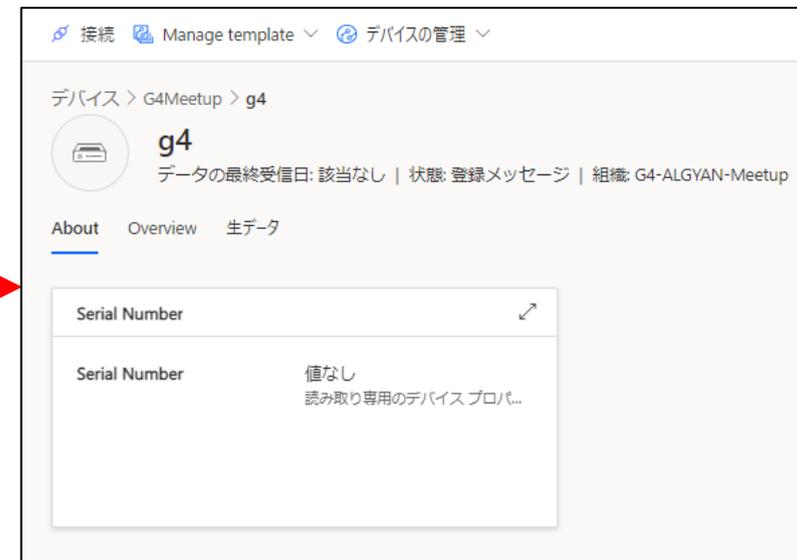
G4Meetup

+ 新規 ← インポート

G4Meetup

デバイス名	デバイスID	デバイスの状態	組織	シミュレート済み
g4	g4	登録済み	...GYAN-Meetup	いいえ

デバイス名をクリックすると、  
詳細を見ることができます。



接続 Manage template デバイスの管理

デバイス > G4Meetup > g4

g4  
データの最終受信日: 該当なし | 状態: 登録メッセージ | 組織: G4-ALGYAN-Meetup

About Overview 生データ

Serial Number

Serial Number 値なし  
読み取り専用のデバイス プロパ...

# デバイス接続情報取得



デバイス接続

ID スコープ ①  
One00 [redacted] **コピー**

デバイス ID ①  
g4 **コピー**

このデバイスの接続の種類を選択します。これは、必要に応じて後で変更できます。

認証の種類  
Shared Access Signature (SAS)

キー QR コード

Shared Access Signature (SAS) は、セキュリティ トークンとキーを使用して IoT Central に接続します。下に表示されている既定の登録グループの SAS キーを使用して、デバイスを登録してください。 [詳細情報](#)

主キー ①  
[redacted] **コピー**

セカンダリ キー ①  
[redacted] **コピー**

**コピーしたら閉じる**

**閉じる**

必要な情報は下記の3つ

- ・ IDスコープ
- ・ デバイスID ※今回は固定
- ・ 主キー

# 接続設定ファイル編集

- ① PCにUSBメモリを挿し、予めDLした g4\_basic\_config.json をエディタで開く
- ② 下記の箇所を編集、保存する

## 編集前

```
{
  "auth": {
    "IOTHUB_DEVICE_DPS_ENDPOINT": "global.azure-devices-provisioning.net",
    "IOTHUB_DEVICE_DPS_ID_SCOPE": "",
    "IOTHUB_DEVICE_DPS_DEVICE_ID": "g4",
    "IOTHUB_DEVICE_DPS_DEVICE_KEY": ""
  },
  (省略)
}
```

IDスコープ

デバイス主キー



## 編集後

```
{
  "auth": {
    "IOTHUB_DEVICE_DPS_ENDPOINT": "global.azure-devices-provisioning.net",
    "IOTHUB_DEVICE_DPS_ID_SCOPE": "0ne00xxxxxx",
    "IOTHUB_DEVICE_DPS_DEVICE_ID": "g4",
    "IOTHUB_DEVICE_DPS_DEVICE_KEY": "xxxxxx/xxxxxxxxxxxxxxxxxxxxxxxxxx"
  },
  (省略)
}
```



- ・ コピペ誤りなど、編集箇所を間違えると Armadillo が IoT Central に接続できないためご注意ください
- ・ エディタに Visual Studio Code を使用している場合は自動的にjson構文チェックが行われます

# 接続設定ファイル書き込み



① PCからUSBメモリを取り外し、ArmadilloのUSBコネクタに挿す

② USBメモリをマウントする

```
armadillo:~# mount /dev/sda1 /mnt
```

③ 前作業で編集した設定ファイルがあるか確認

```
armadillo:~# ls /mnt/g4_basic_config.json  
/mnt/g4_basic_config.json
```

④ 設定ファイルを コンテナ内にコピー

```
armadillo:~# podman cp /mnt/g4_basic_config.json azure:/root/Azure-IoT-samples/Armadillo-IoT_GW/.
```

# 接続設定ファイル書き込み

## ⑤コピーできているか確認

```
armadillo:~# podman exec azure cat /root/Azure-IoT-samples/Armadillo-IoT_GW/g4_basic_config.json
{
  "auth": {
    "IOTHUB_DEVICE_DPS_ENDPOINT": "global.azure-devices-provisioning.net",
    "IOTHUB_DEVICE_DPS_ID_SCOPE" " 0ne00xxxxxx",
    "IOTHUB_DEVICE_DPS_DEVICE_ID": "g4",
    "IOTHUB_DEVICE_DPS_DEVICE_KEY" "xxxxxx/xxxxxxxxxxxxxxxxxxxxxxxxxxxx"
  },
  (省略)
}
```

IDスコープ

デバイス主キー

## ⑥USBメモリをアンマウントし、Armadilloから取り外す

```
armadillo:~# umount /mnt
```



CON4

U  
S  
B  
メ  
モ  
リ

## ① Azureコンテナの中に入る

```
armadillo:~# podman exec -it azure /bin/sh
```

```
/ #
```

## ② スクリプト実行

```
/ # python3 /root/Azure-IoT-samples/Armadillo-IoT_GW/azure_g4_basic.py
```

デバイスが接続されると以下のログが表示されます。

```
Device was assigned
```

```
iotc-xxx-xxx-xxx-xxx-xxx.azure-devices.net
```

```
g4
```

```
Press Q to quit
```

## ③ IoT Central 上で接続を確認する



## ④ スクリプト停止

```
Device was assigned
iotc-xxx-xxx-xxx-xxx-xxx.azure-devices.net
g4
Press Q to quit
Q
Quitting...
all threads are quited, then shutdown the IoT client...
```

Q を入力してエンター

## ① Azureコンテナから出る

```
/ # exit  
armadillo:~#
```

## ② ArmadilloにWebカメラを接続する



## ③ Azure 接続スクリプトを実行する

```
armadillo:~# podman exec -d azure python3 /root/Azure-IoT-samples/Armadillo-IoT_GW/azure_g4_basic.py -d
```

## ④ 物体認識を実行する

### ■ Webカメラあり

```
armadillo:~# podman exec -d detect /root/handson.sh --camera --cloud
```

### ■ Webカメラなし(動画を使用)

```
armadillo:~# podman exec -d detect /root/handson.sh --cloud
```

# 物体認識 + Azure 実行

## IoT Central デバイス画面

およそ 1分周期で表示が更新されます

