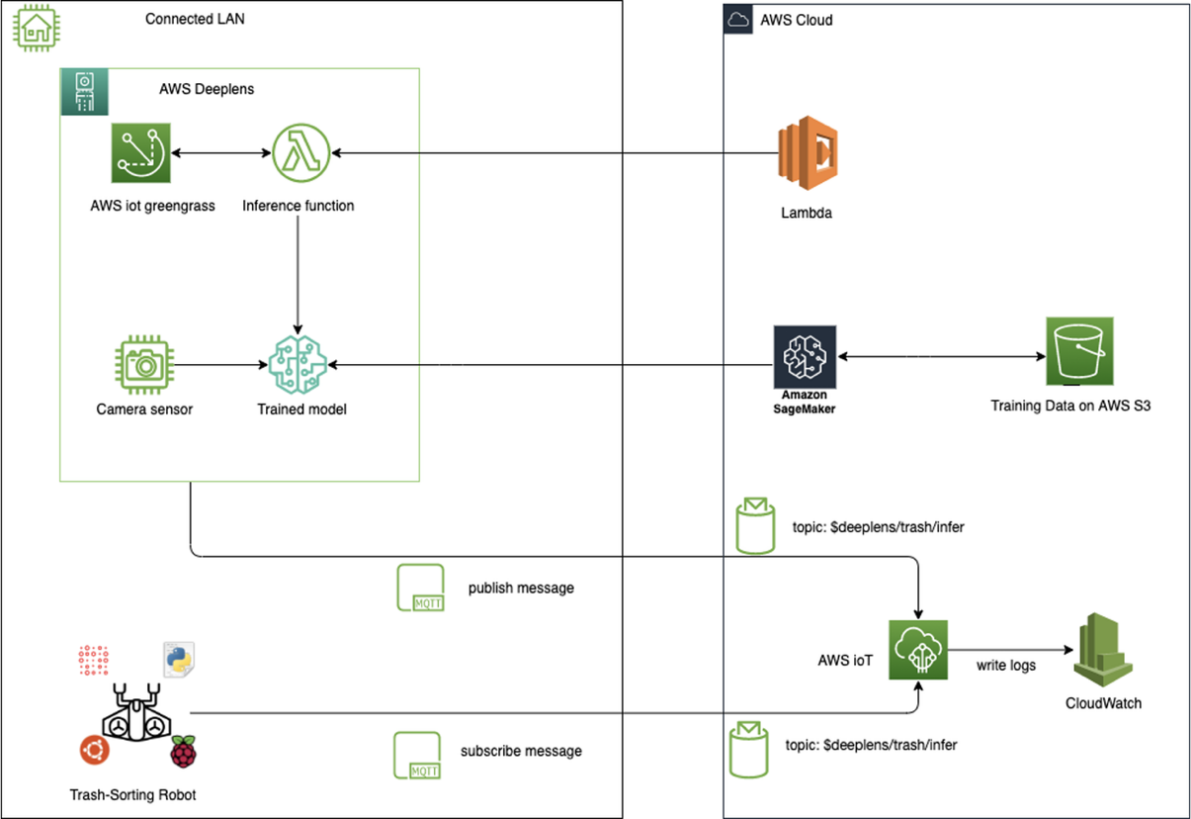
**垃圾分类机器人实验手册**

# 一 、架构图



# 二、所需材料

* 账号准备：AWS global账号；
* 硬件准备1：AWS Deeplens 摄像头；
* 硬件准备2:Hiwonder机械手臂；
* 服务配额申请：ML.p2.xlarge实例的notebook instance using, Train job using的许可分别配额>1个；
* 代码准备：拷贝github到本地：https://github.com/William2005-git/TrashSorting/blob/main/subTrash.py

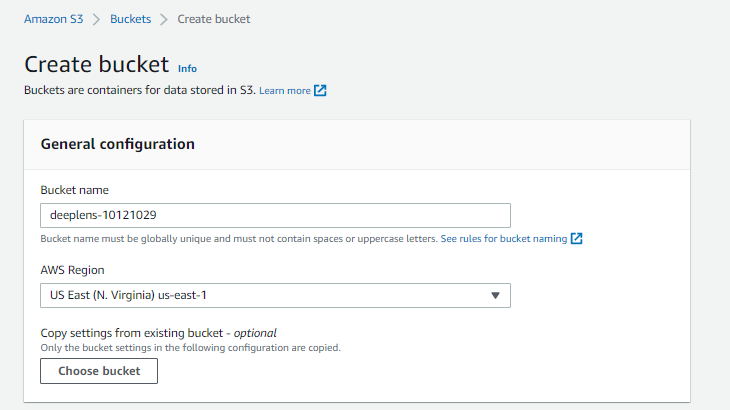
# 三、实验步骤预览

1. 使用AWS SageMaker训练模型；
2. 使用AWS Lambda编写 ML Inference function；
3. 注册摄像头到AWS账户内的Deeplens服务，并部署模型与function到摄像头边缘端；
4. 在机械手臂的树莓派内编辑垃圾分类动作；
5. 在机械手臂的树莓派内安装AWS IoT SDK，准备运行环境；
6. 将机械手臂硬件注册到AWS IoT Core服务,建立与Deeplens的通讯联系；
7. 编写垃圾智能分类python程序subTrash.py；
8. 运行和调测垃圾智能分类程序。

## 3.1模型训练与生成

### 3.1.1 训练数据于模型输出的S3桶准备

* 登陆AWS海外账号管理控制台, 选择us-east-1区域
* 选择S3服务， 创建储存桶用来存储垃圾分类图片数据和模型输出的对象桶。



* 创建步骤参考如下链接<https://docs.amazonaws.cn/AmazonS3/latest/userguide/creating-bucket.html>
* 注意：桶命名必须为***deeplens-\*\*\*\*\**** *(IAM角色权限限制，桶名区域唯一，可以加随机数字)。*

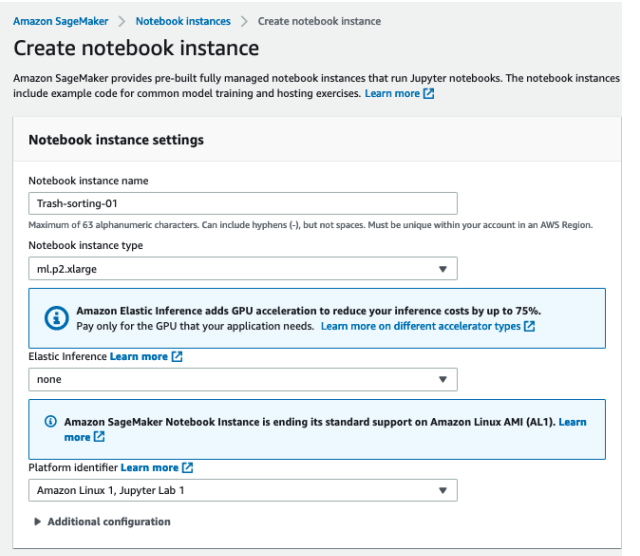
### 3.1.2 创建SageMarker的notebook实例

* 登录SageMaker服务。选择Notebook笔记本实例服务

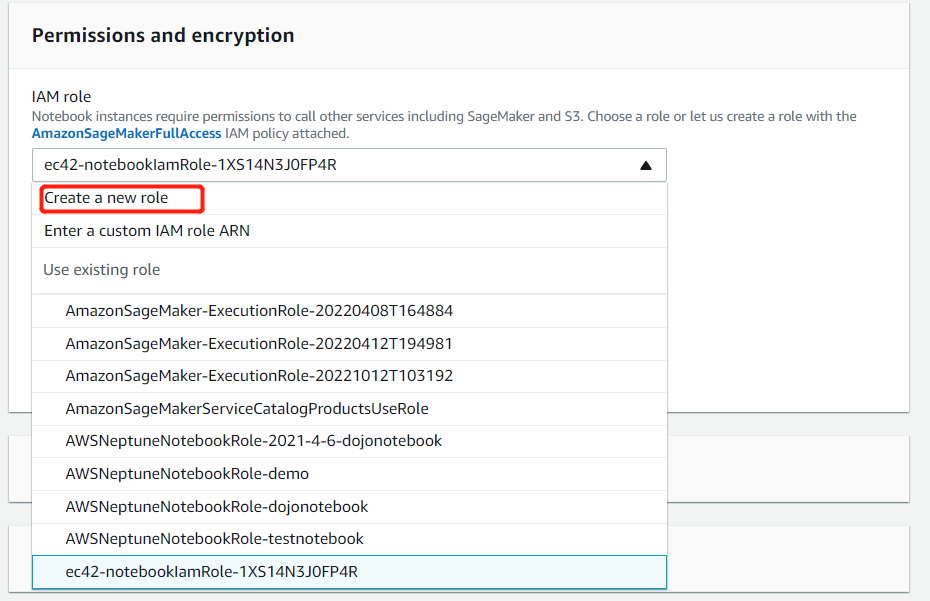


* 选择笔记本实例，创建笔记本实例, 需要使用机型 ml.p2.xlarge



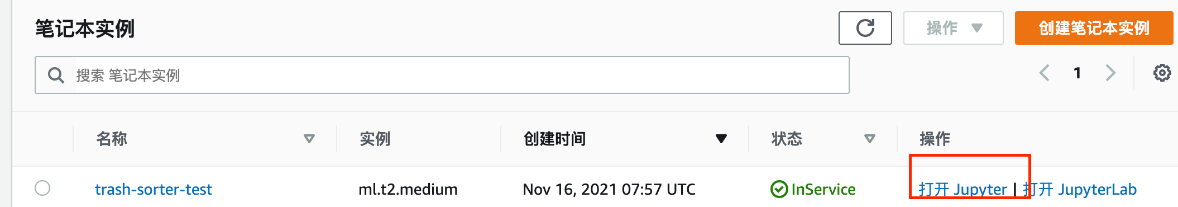


* 选择创建新的IAM角色。其他采用默认参数。等待实例创建完成。

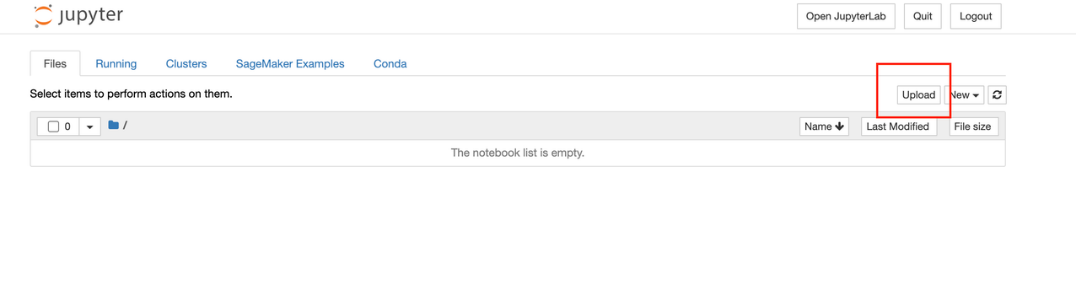


### 3.1.3 执行模型训练

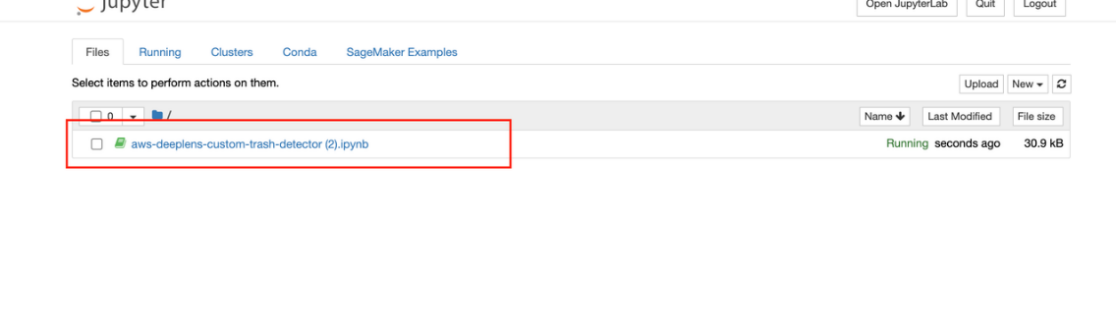
* 选择Notebook实例，选择【打开Jupter】按钮，登录训练实例。



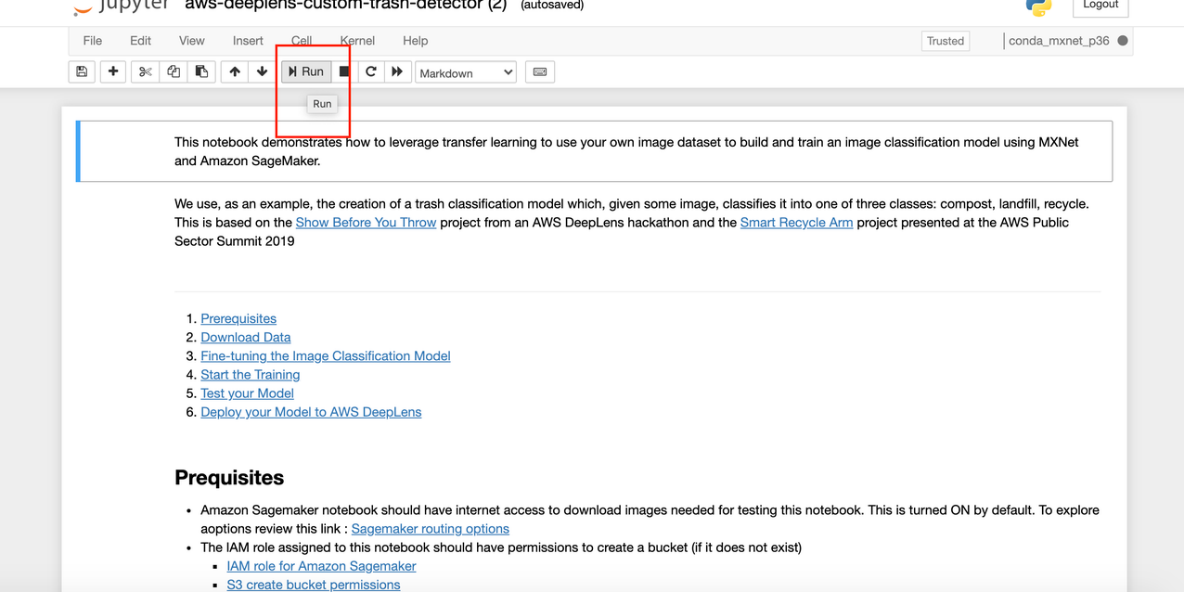
* 上传github上下载到本地上的aws-deeplens-custom-trash-detector-v2.ipynbw模型代码。



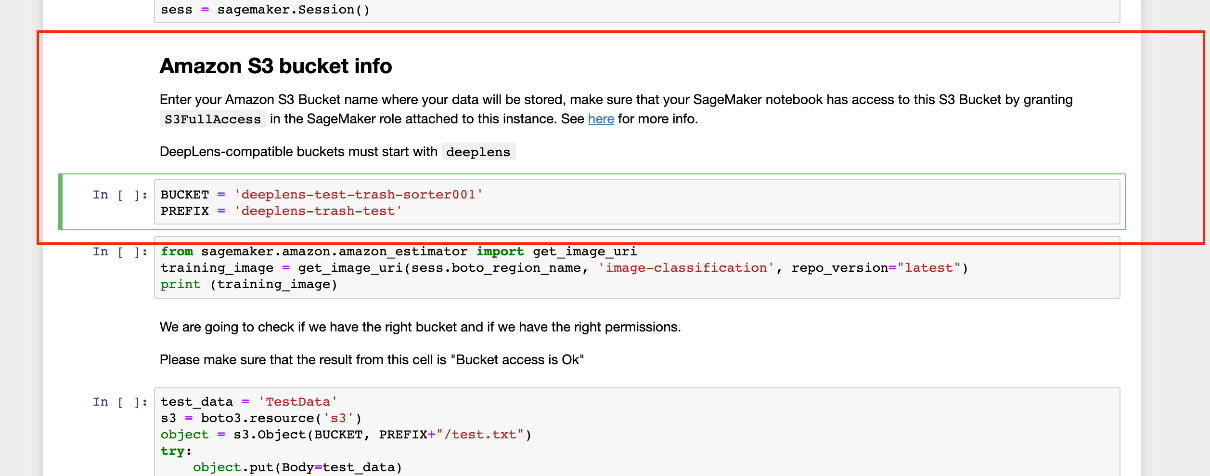
* 打开上传完成的[aws-deeplens-custom-trash-detector.ipynb](https://www.awsdeeplens.recipes/code/trash-sorter/aws-deeplens-custom-trash-detector.ipynb)。里面包含了下载训练图片数据集，整理数据，训练模型，测试模型等步骤。



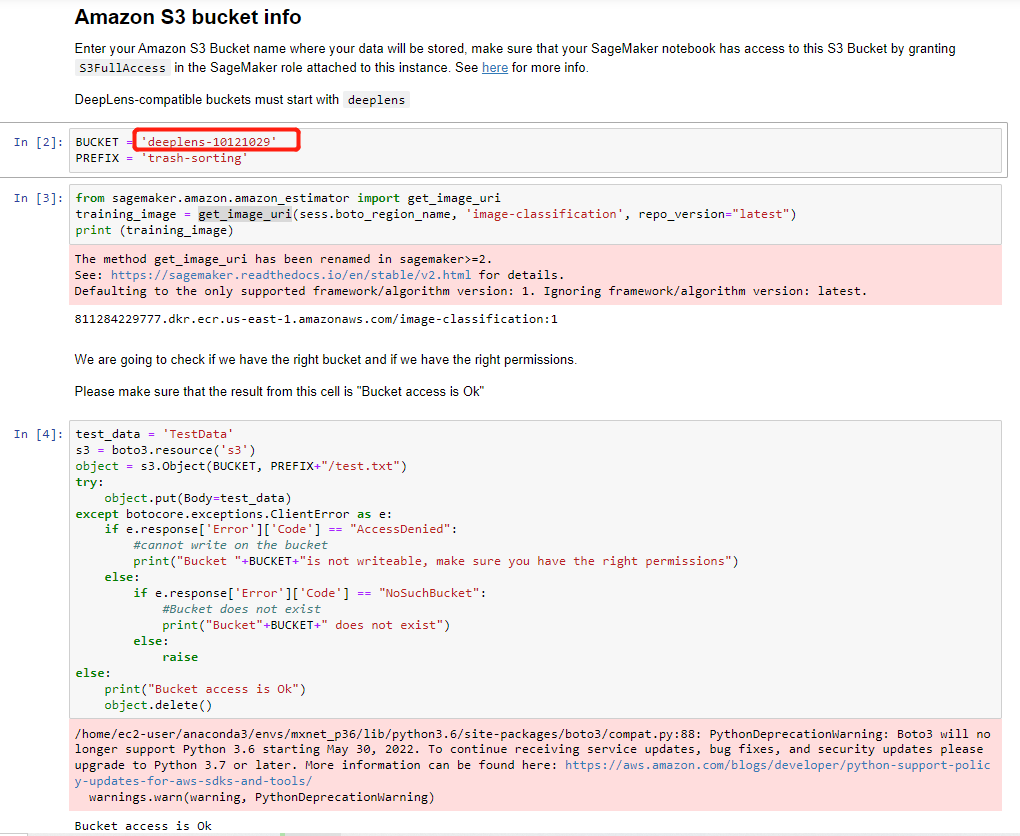
* 阅读和理解模型代码中的文字说明，按文字指示步骤执行所有cells。留意部分代码中的参数设置。



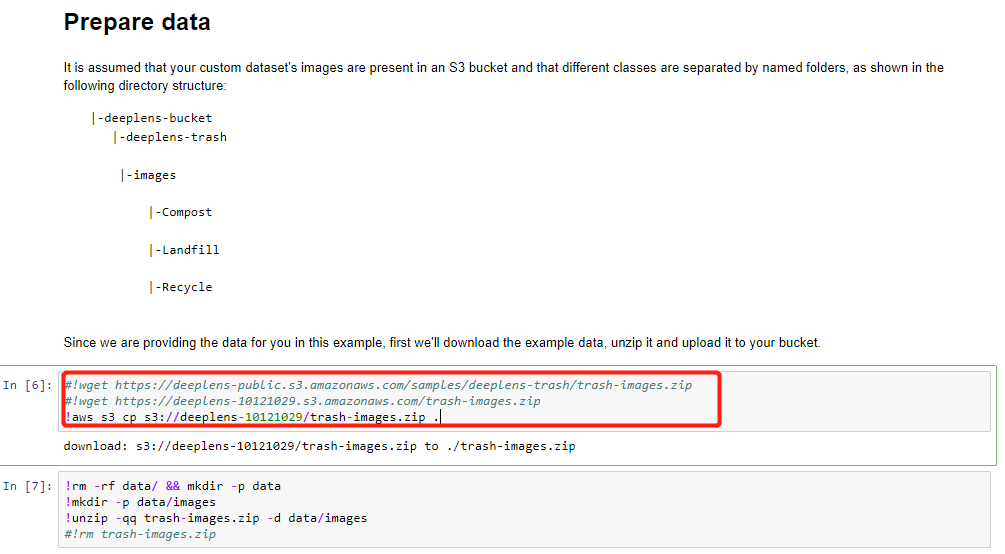
* 注意，正在执行的cell，带有In [\*]，已经执行完的cell有序号,In [1]，有些步骤执行需要时间，请耐心等待。
* S3 桶变量参数设置**，**指定为之前创建的S3桶，譬如桶名 **deeplens-*test-trash-sorter001***



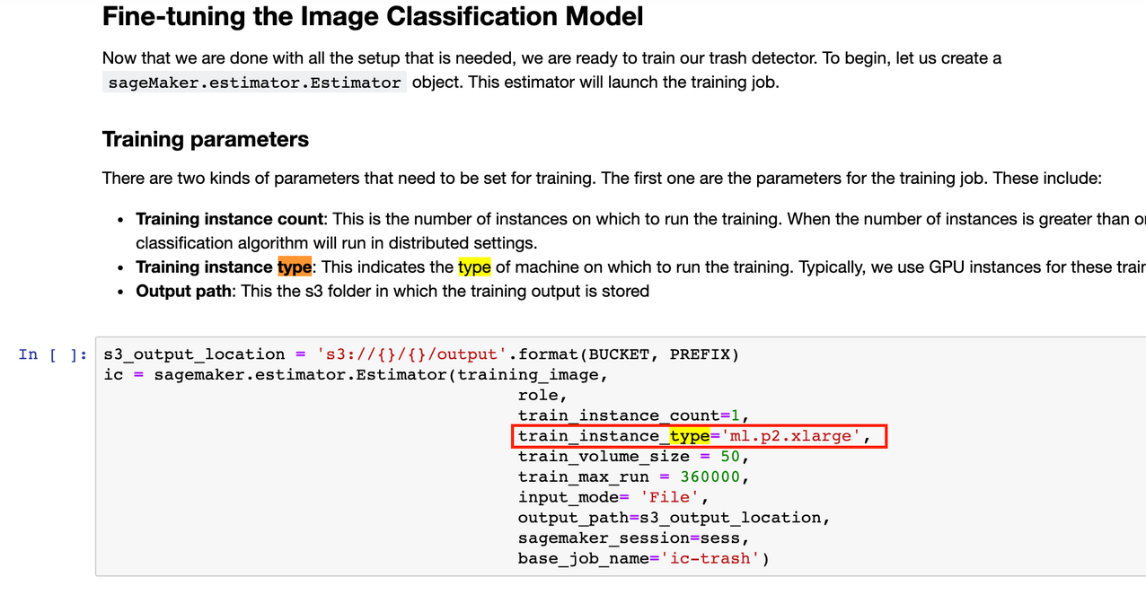
* S3 桶访问权限测试，确认sagemaker具备桶的操作权限。#使用前面创建的Bucket名称（deeplens-xxxx）。



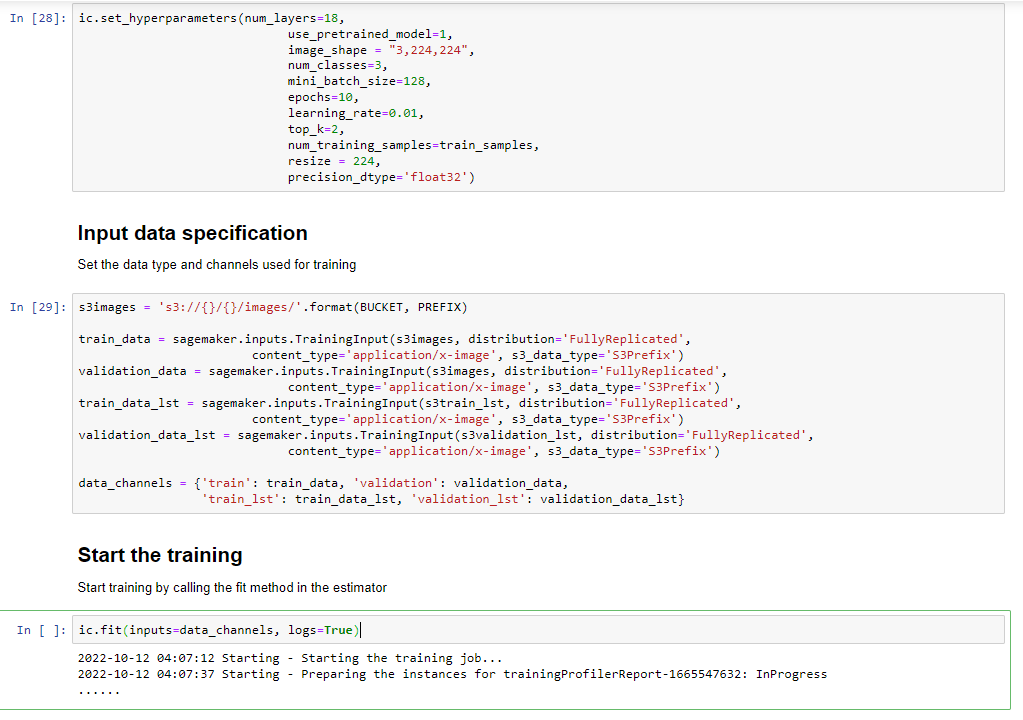
* 模型训练数据准备，将”垃圾分类样本数据.zip”下载到sagemaker 的notebook中，并完成检查和预处理后，上传到S3桶。



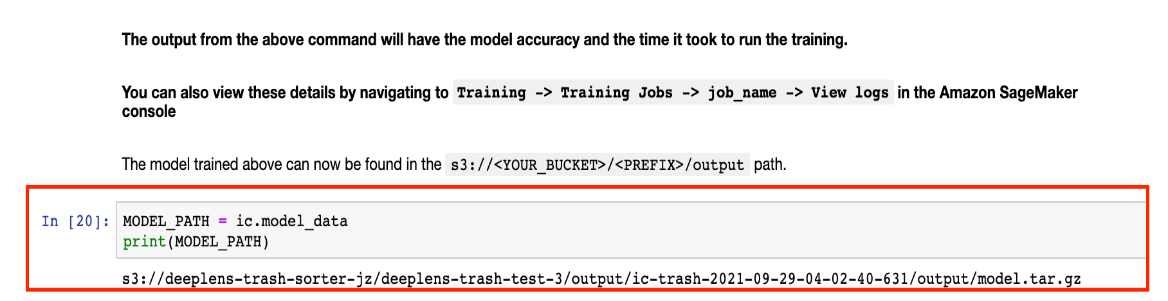
* 修改模型实例参数，输入自己所使用的实例机型。本模型代码执行，默认采用ml.p2.xlarge类型实例，需提前审批1个配额。



* 训练模型，#需要申请ml.p2.xlarge的training job using的配额申请。训练过程大概持续6～10分钟。



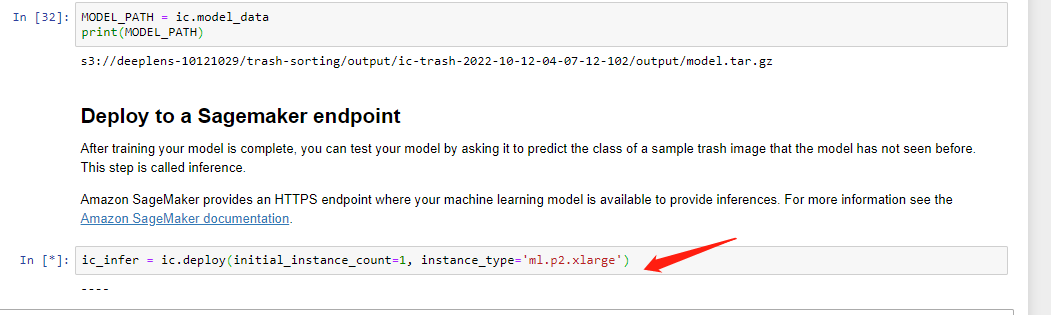
* 当你看到这个输出的时候，表示模型训练完成，存储于此路径。



* 本项目已经有训练效果更好的模型，通过S3桶的操作，使用本地文件的model.tar.gz，替换原有的文件。因为Deeplend的模型加载是通过Sage Marker的train job关联模型，不能指定模型的存储路径。

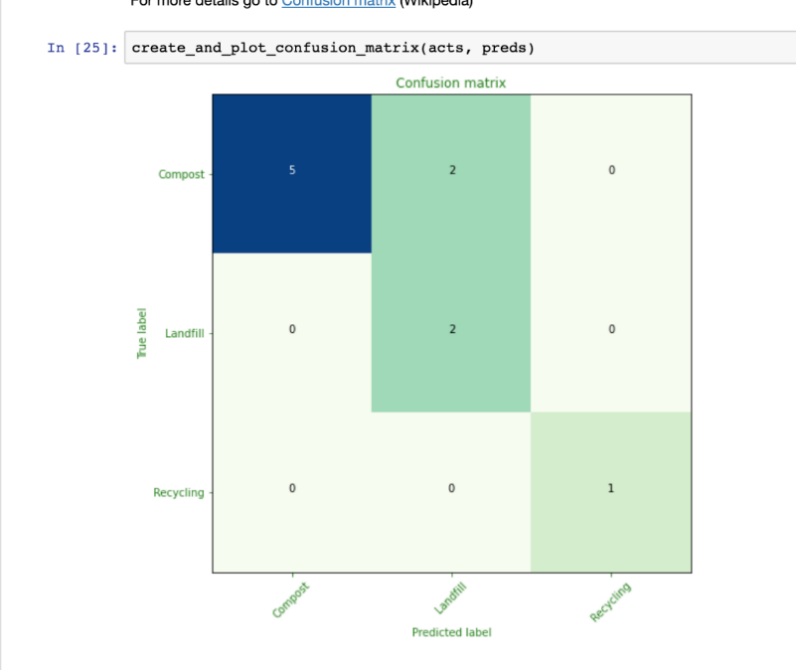
### 3.1.4 执行模型部署

* 完成模型训练和生成后，可以将模型部署起来，执行模型的校验和评估。从而检查模型的精确度。
* 部署模型，将模型部署到ml.p2.xlarge实例上，部署数量为1。生成部署模型的访问端点。



### 3.1.5 执行模型评估

* confusiton matrix衡量模型的准确性，可以通过这个测试来调整模型输入参数，优化模型。混搅矩阵模型说明链接如下：<https://en.wikipedia.org/wiki/Confusion_matrix>
* 配置模型参数，执行模型评估，评估后的混搅矩阵如下图所示。



## 3.2 Lambda函数定义

### 3.2.1用AWS Lambda编写 ML Inference function

* 下载Inference function 代码例子到本地。代码的Github库链接如下，将代码打包成Zip下载。
* [https://github.com/William2005-git/TrashSorting/raw/main/trash-sorter-lambda-a95a5a86-6dfd-4867-b7cc-027da7dda51a.zip](https://github.com/jzamazon/trash-sorter/tree/main/trash-sorter-lambda)
* 打包代码成zip，部署到lambda。执行操作参考如下链接：<https://docs.aws.amazon.com/zh_cn/lambda/latest/dg/python-package.html>



* 本步骤已经具备有ZIP包，直接在Lambda中上传部署即可。确定代码运行成功后，发布新版本。



## 3.3 Deeplens摄像头配置

### 3.3.1注册AWS deeplens摄像头

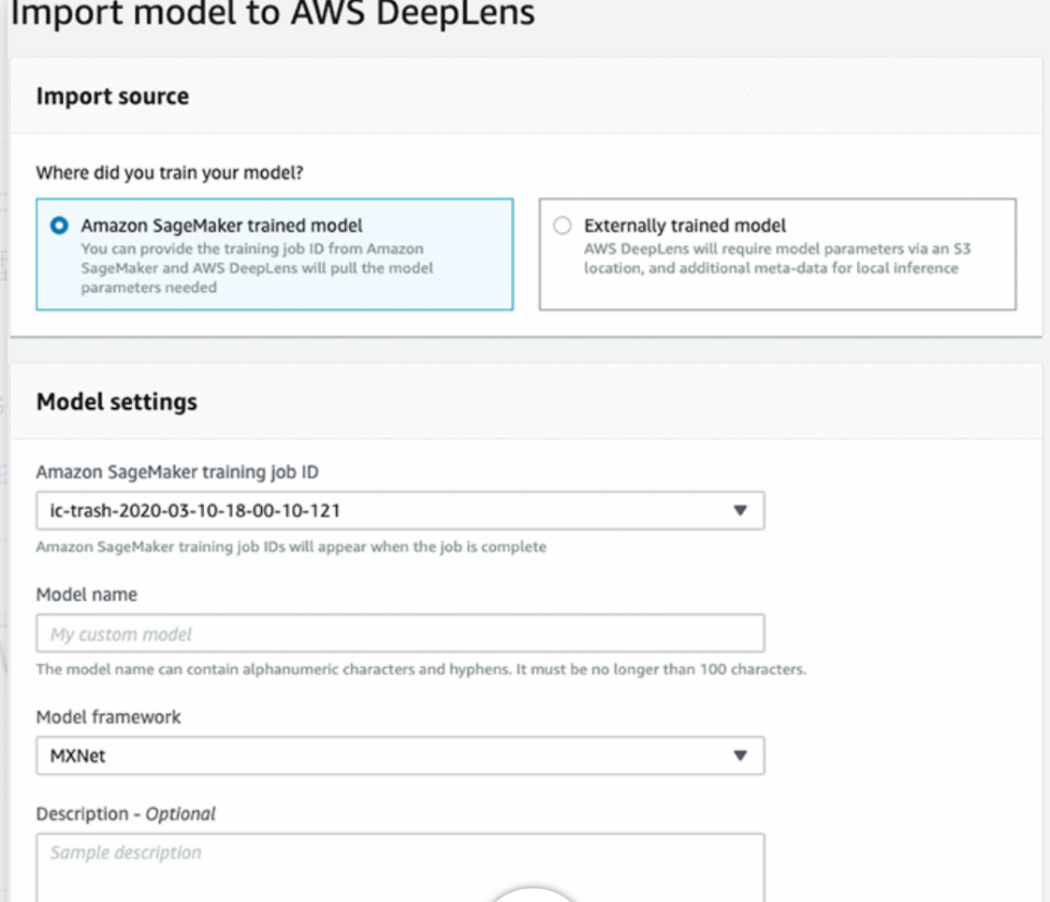
* 登录AWS Deeplens服务。注册摄像头。
* 首次注册需要输入AWS Deeplens摄像头密码，必须保存好此密码，之后无法再忘记密码或者修改。建议采用其本地Wi-Fi密码。
* V1.1 版本 <https://docs.aws.amazon.com/zh_cn/deeplens/latest/dg/deeplens-register-device-v1.1.html>
* V1.0 版本 <https://docs.aws.amazon.com/zh_cn/deeplens/latest/dg/deeplens-register-device-v1.html>
* 本项目配置设备外标识为V1.1，但测试显示只能使用V1.0版本添加方式，添加时切记采用根账号登录。AWS员工账号需要从<https://isengard.amazon.com/的admin角色登录，否则V1.0>注册提示没有权限。
* 参数定义：Device my-name:deepcom-1
* create roles，下载Deeplens控制台的证书。

### 3.3.2 deeplens摄像头本地配置

* 连接Deeplen WI-FI AMDC0C137 password: 8XF9Nw1x，本地操作电脑需断开VPN网络，如果找不到热点，需重置一个摄像头的复位键（细针按入后约10秒）。
* 切换到deeplen的本地配置WEB，设置网络，上传证书和密码。
* 本地访问地址：<http://deeplens.amazon.net>或<http://deeplens.config>。（http://192.168.0.1，软件包版本低于 1.2.4，否则为<http://10.105.168.217>）

### 3.3.3 模型加载和项目配置

* 切回互联网的网络，回到AWS DeepLen Console端。
* 将3.1 步骤里训练好的模型导入到AWS Deeplens。
* 在 [AWS DeepLens console](https://console.aws.amazon.com/deeplens), 选择Model， Import model。选择 Amazon SageMaker trained model. 选择job ID（SageMarker任务必须顺利完成），输入 model name. 选择 MXNet 作为 model framework导入模型。选择导入模型。
* 注意：如果桶名不符合权限要求的“deeplens”开头，会报错“\*\*\*文件和目录不正确。请确认模型文件路径\*\*\*\*”。



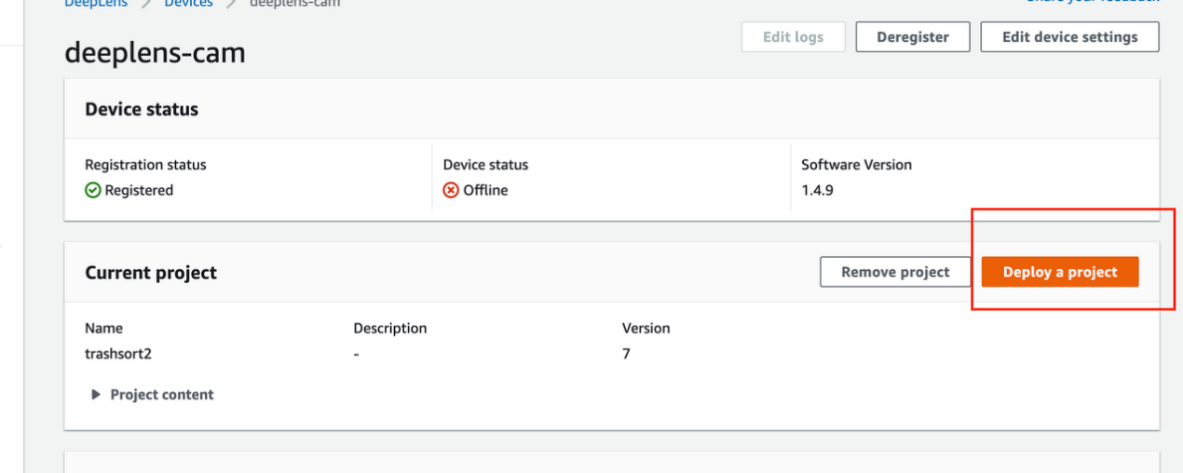
* 创建AWS Deeplens projects。操作链接参考如下：

<https://docs.aws.amazon.com/zh_cn/deeplens/latest/dg/deeplens-get-start-easy.html>

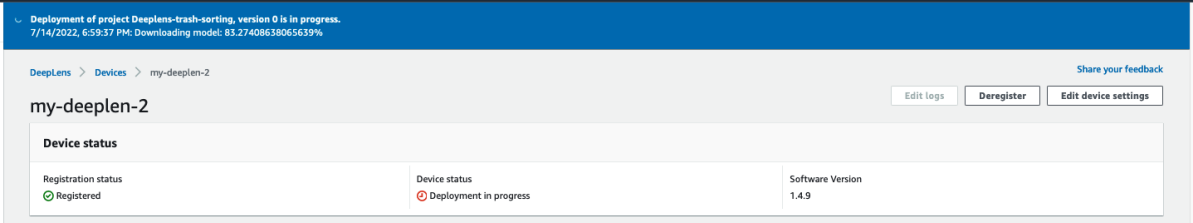
* 选择导入的模型和3.2步骤中部署发布的Lambda function。
* Lambda 选择添加已经整理好的Lambda zip文件trash-sorter-lambda-a95a5a86-6dfd-4867-b7cc-027da7dda51a.zip直接导入即可。

### 3.3.4 将项目部署到Deeplen摄像头

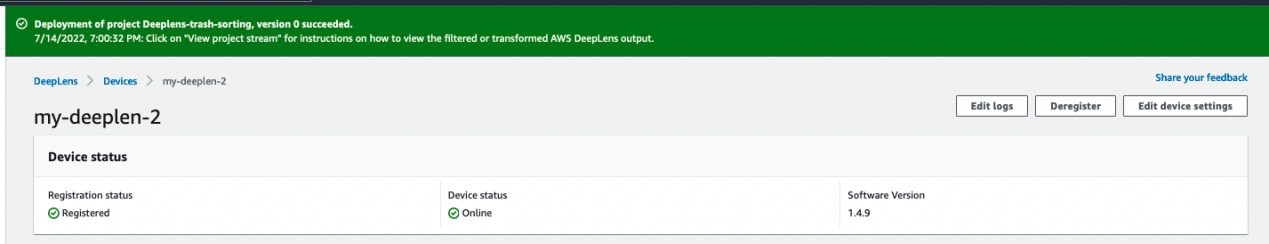
* 将创建好的AWS Deeplens projects部署到摄像头。留意部署过程中的状态变化。



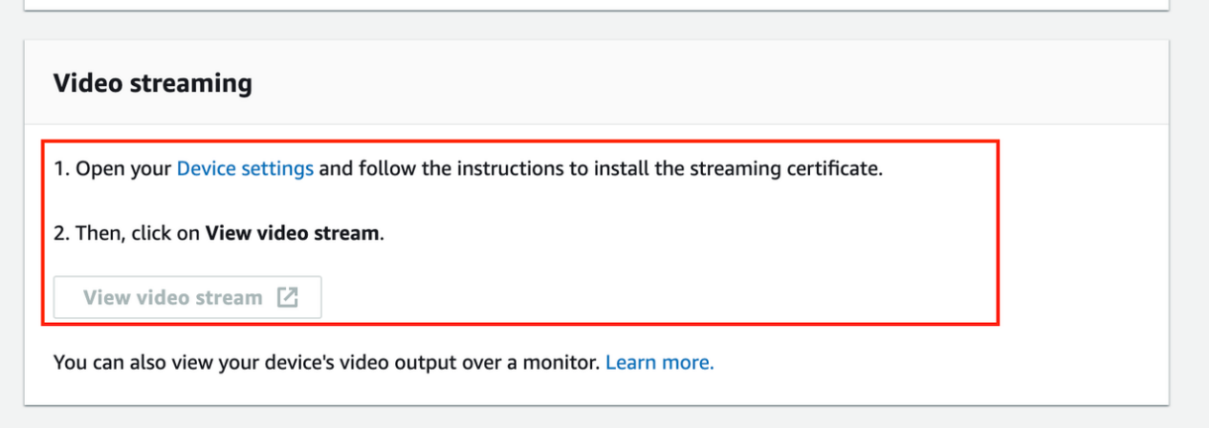
* 正在部署过程中的状态显示。



* 部署完成后，配置浏览器，安装DeepLens的流认证证书。



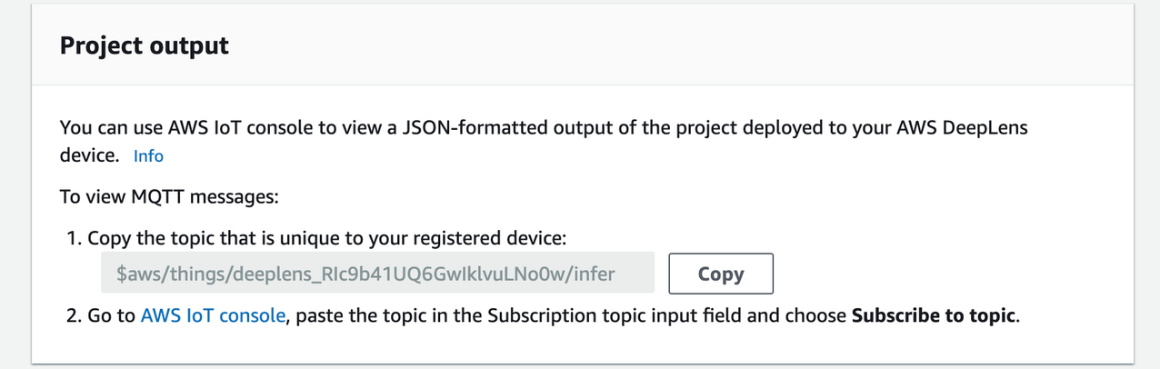
* 本地切换网络，登录Deeplens的本地地址：<http://deeplens.amazon.net>或<http://deeplens.config>。下载流认证证书。
* 本地计算机，选择浏览器类型，按要求安装流认证证书，上传证书到firefox/chrome 浏览器。对于 FireFox（Windows 和 macOS Sierra 或更高版本），请执行以下步骤：
* 在 FireFox 中选择 Preferences (首选项)。
* 选择 Privacy & Security (隐私与安全性)。
* 选择 View certificate (查看证书)。
* 选择 Your Certificates (您的证书) 选项卡。
* 选择 Import (导入)。
* For certificate password, type "DeepLens" and choose OK
* You will be able to view the output stream for registered devices via the AWS DeepLens console
* AWS控制台设备里面，点击view video stream，查看流视频。



* 或在浏览器验证模型是否在摄像头中运行
* https://10.1.0.107:4000
* <https://docs.aws.amazon.com/zh_cn/deeplens/latest/dg/deeplens-viewing-device-output-in-browser.html>。

### 3.3.5 Deeplan摄像头的MQTT消息流检查

* 在AWS IoT平台。订阅菜单，输入本项目的MQTT的Topic。可以看到MQTT消息流。



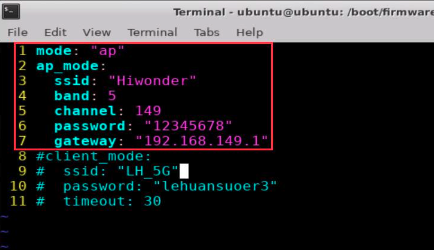
## 3.4 机械臂配置

### 3.4.1机械臂初始化

* Wi-Fi网络配置。抽出机械臂的记忆卡，通过转换头，插入其他电脑后，执行如下Wi-Fi网络配置（需要与Deeplen摄像头处于同一网络）。
* 修改：

# cd /boot/firmware/Hiwonder

# sudo vim wifi.yaml ##修改ssid和password 参数，”:wq”保持退出。



Note:本机修改，重启Wi-Fi服务

# sudo systemctl restart hw\_wifi.service

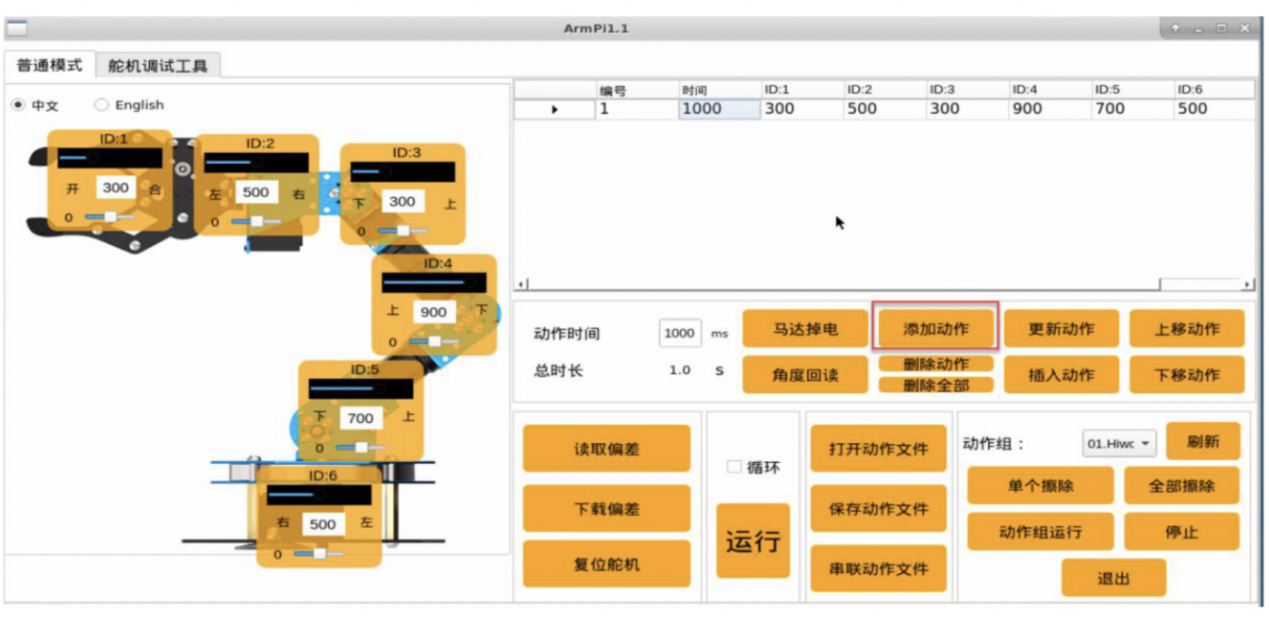
* 安装记忆卡，在断电的情况下， 金手指向上，插回记忆卡。

### 3.4.2启动机械臂

* 给机械臂接入电源，本地登陆需连接显示器、键盘和鼠标。
* 启动机械臂机，直到出现操作桌面。

### 3.4.3 配置机械臂动作

* 本地[登录机械手臂系统的桌面](https://github.com/jzamazon/trash-sorter/blob/main/robotic-arm/%E8%BF%9C%E7%A8%8B%E5%B7%A5%E5%85%B7%E5%AE%89%E8%A3%85%E4%B8%8E%E8%BF%9E%E6%8E%A5.pdf)。（系统支持远程登录，通过说明文档配置Windows登录软件）。
* 打开动作编辑软件，设置动作参数，编辑垃圾分类的拾取动作并保存。
* 如果机械手臂的钳子太小，可以更换尺寸大的钳子，或者更换钳子为垃圾箱。

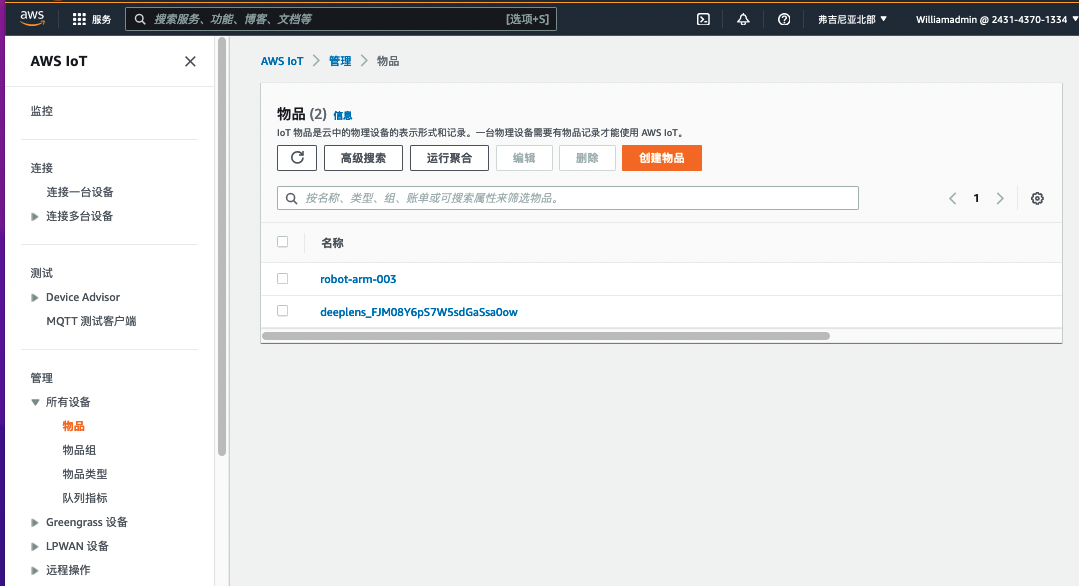


* 编辑动作，参考*[上位机链接](https://github.com/jzamazon/trash-sorter/blob/main/robotic-arm/%E4%B8%8A%E4%BD%8D%E6%9C%BA%E7%9A%84%E4%BB%8B%E7%BB%8D.pdf)。*动作编辑*[教学链接](https://github.com/jzamazon/trash-sorter/blob/main/robotic-arm/%E5%8A%A8%E4%BD%9C%E6%89%A7%E8%A1%8C%E6%95%99%E5%AD%A6.pdf)。*

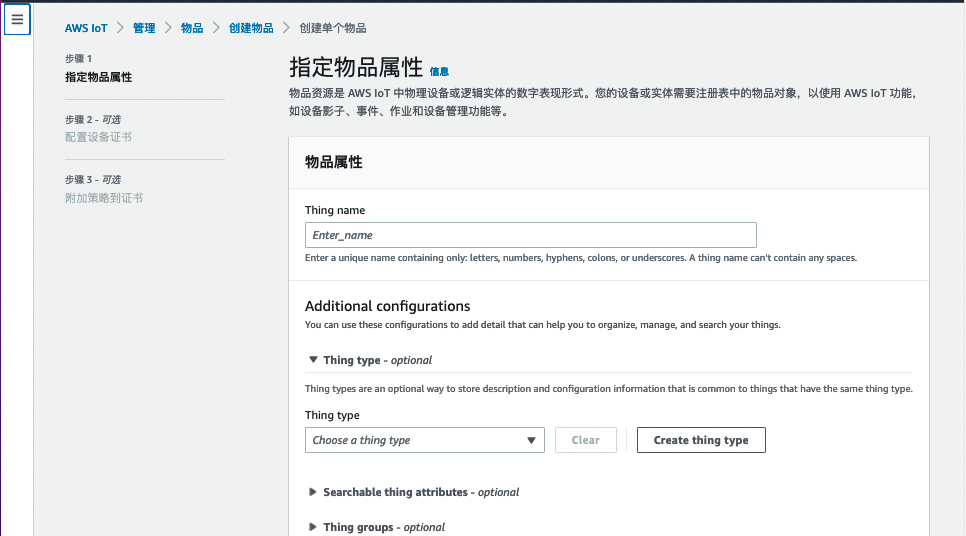
## 3.5 AWS IoT 平台配置

### 3.5.1 IoT Greengrass设备注册

* 登录AWS IoT Core管理控制台
* 左侧导航栏，选择所有设备，物品，注册物品-机械臂。



* 选择单一物品，输入备注信息，设备影子选择“No Shadow”完成新设备注册。



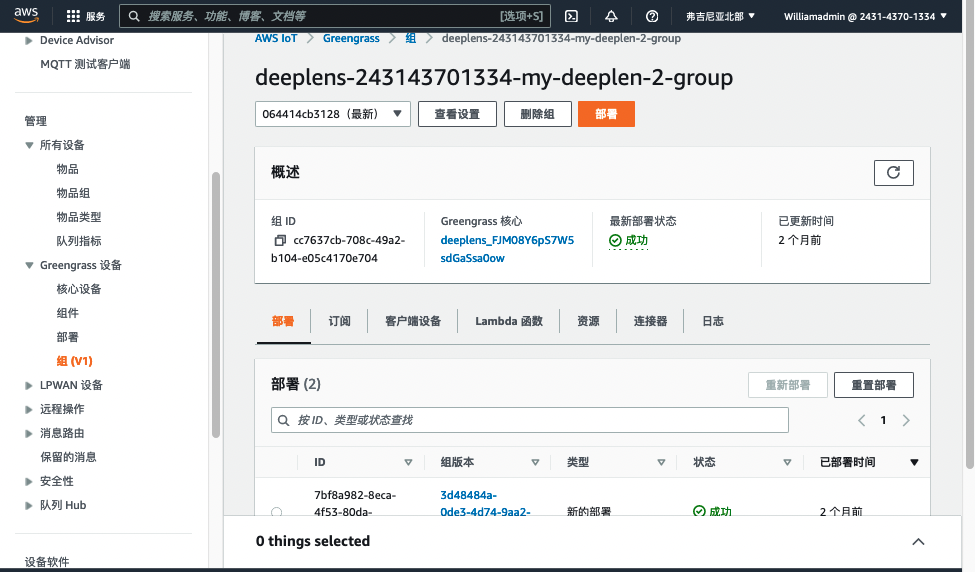
* 参考操作文档的链接如下：<https://docs.aws.amazon.com/greengrass/v1/developerguide/device-group.html>

### 3.5.2 证书生成

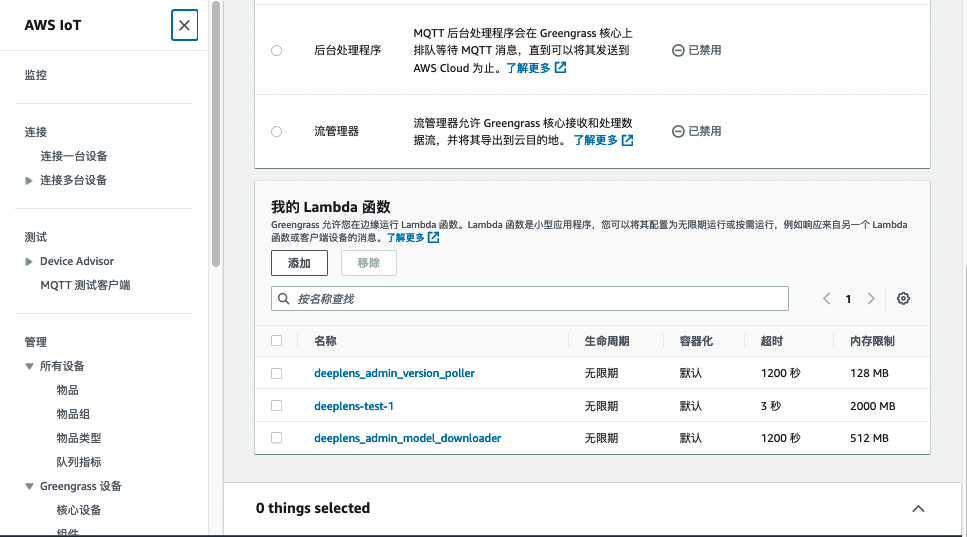
* 自动生成物品的证书，公有密钥和私有密钥。文件名称是证书的随机码，不能修改或调整。
* 保存证书与密钥到机械手臂的本地目录。

### 3.5.3 添加Lambda函数

* 添加3.2步骤中创建好的Lambda函数到AWS Greengrass组中。
* 左侧导航栏，选择【Greengrass设备】，【组（V1）】，选择系统前期生成的Deeplen设备。



* 选择【添加】Lambda函数。



## 3.6 安装AWS IoT SDK到机械手臂

### 3.6.1 安装机械臂的开发环境（可选）

* 远程登录机械手臂系统
* 安装Python3
* [安装AWS IoT Python3 SDK](https://docs.aws.amazon.com/greengrass/v1/developerguide/IoT-SDK.html) 到ArmPI\_PC\_Software路径下

### 3.6.2测试与平台通讯

* [测试是否能成功监听AWS IoT设备](https://docs.aws.amazon.com/greengrass/v1/developerguide/test-comms.html)。（需替换红色部分的变量参数）。

python basicDiscovery.py --endpoint *AWS\_IOT\_ENDPOINT* --rootCA *root-ca-cert.pem* --cert *publisher.cert.pem* --key publisher.private.key --thingName HelloWorld\_Publisher --topic *'hello/world/pubsub'* --mode publish --message 'Hello, World! Sent from HelloWorld\_Publisher'

## 3.7 编写垃圾分类代码

### 3.7.1代码执行逻辑

* 在ArmPI\_PC\_Software目录下创建subTrash.py Python脚本。脚本算法逻辑：

1.启动机械手臂，完成设备初始化检测。

2.监听DeepLen摄像头IoT endpoint，接收来自摄像头MQTT消息。

3.执行MQTT消息处理，譬如 Recycle时，触发机械手臂垃圾回收动作。

4.进入设备休眠周期，等待下一动作循环。

### 3.7.2 代码调用命令

* 测试和执行脚本样例：

python3 pubsub.py --topic topic\_1 --root-ca ~/Desktop/greengrass/root-ca-cert.pem --cert ~/Desktop/greengrass/3500fb7b82.cert.pem --key ~/Desktop/greengrass/3500fb7b82.private.key --endpoint [avr0vt4155e1d-ats.iot.us-east-1.amazonaws.com](http://avr0vt4155e1d-ats.iot.us-east-1.amazonaws.com)

### 3.7.3 源代码脚本

* 源代码链接：https://github.com/William2005-git/TrashSorting/blob/main/subTrash.py

## 3.8 实验验证

* 实验如果部署成功后。在摄像头监测到可回收垃圾后，机械手臂会运行垃圾分类动作。
* Deeplen的本地具备实时视频显示。
* IoT Core的MQTT可以检测到DeepLen发送的垃圾分类消息。