Getting Started Guide for GreengrassV2 on DHCON DRC02

Table of Contents

1 Document Information

1.1 Revision History (Version, Date, Description of change)

Revision	Date	Description
0.1	*May 6, 2022*	*Initial draft*

2 Overview

2.1 About AWS IoT GreengrassV2

The AWS IoT Greengrass client software, also called AWS IoT Greengrass Core software, runs on Windows and Linux-based distributions, such as Ubuntu or Raspberry Pi OS, for devices with ARM or x86 architectures. With AWS IoT Greengrass, you can program devices to act locally on the data they generate, run predictions based on machine learning models, and filter and aggregate device data. AWS IoT Greengrass enables local execution of AWS Lambda functions, Docker containers, native OS processes, or custom runtimes of your choice.

To learn more about AWS IoT GreengrassV2, see how it works and what's new.

It is important to note that this example distribution as well as this document are focused on getting a base platform ready for development. Additional considerations must be given to security both on the platform as well as cloud configuration to advance beyond the development stages.

The system image as well as the Yocto project contain the required dependencies for the IDT Suite to run the required tests as well as the optional docker tests, but do not contain the dependencies for the ML tests.

2.2 About DRC02 IoT Gateway

The DHCON products are optimized for smart home and building as well as Industry 4.0 and IoT applications.

The DRC02 platform powered by DHCOM STM32MP1 is a universally usable computer for control and connectivity tasks and is well suited for use as a gateway in a Greengrass implementation.

The DHCOM STM32MP1 is suitable for many applications and stands out from the crowd with its diverse analogue and digital possibilities. Thus, the STM32MP1 microprocessors enable powerful IoT and/or HMI applications from sensor and actuator to the cloud with only one chip.

Applications

- Industrial Automation
- Machine controls and operator panels (HMI)
- Home & Building
- Medical Technology

More information concerning the DHCON DRC02 platforms can be found here and additional information on the DHCOM STM32MP1 module can be found here.

3 Hardware Description

3.1 User Provided items

In addition to the DHCON DRC02 platform, it is expected that the end user will supply

- 24v power source
- Serial cable (USB to USART)
- Network cable
- Micro SDCard (16Gb is recommended)
- Router with internet connectivity

4 Set up your Development Environment

This section describes Greengrass bring upon the DRC02 platform. This process can be broken down into two phases.

- Downloading the generated system image or building the target image
- Setting up the HW platform

4.1 Downloading the Generated System Image

The filesystem image generated by the archived build system can be found here (Link location TBD).

We recommend the filesystem image be downloaded and written to an SDCard to avoid setting up a full development system and rebuilding the image.

Once downloaded, the image can be written to an SD card from the host Linux system.

Enter the command below, please take care to ensure the <SDCARD_DEVICE> is replaced with the proper host system identifier.

Writing to the incorrect device can overwrite your Linux install.

```
unxz -c core-image-minimal-dh-stm32mp1-dhcom-drc02.wic.xz | sudo dd
of=/dev/<SDCARD_DEVICE> bs=1M iflag=fullblock oflag=direct conv=fsync status=progress
```

4.2 Building the Target Image

This section is for developers who want to rebuild the full system image, a project is available to build the target image.

We recommend a Linux development environment be used to regenerate the GreenGrass supporting file system image. The archived project is a yocto build that generates the full root file system image. This project is also used to generate a cross platform application development environment and for kernel extensions.

There are additional packages required for the OpenEmbedded / Yocto build process, details can be found here. Please refer to section 3.2 of the PC prerequisites page for a list of the packages to install on the host development machine.

- From the host development machine ensure the repo utility is available
  ```
  curl http://commondatastorage.googleapis.com/git-repo-downloads/repo > ~/bin/repo
  chmod a+x ~/bin/repo
  ```

- Create a working directory
  ```
  mkdir working
  cd working
  ```

- Clone the project to the working directory
  ```
  Git clone git://projectpath.git        -- project location and link TBD
  ```

- Change to the project directory and execute the script provided
  ```
  cd drc02-gg/projects/drc02-greengrass
  . drc02-greengrass.sh
  ```

- From the projects/drc02-greengrass/working/drc02-gg directory the build can be started
  ```
  bitbake core-image-minimal
  ```

- The output of the build will be the full filesystem image. Refer to section 4.1 for instructions on how to write that image to an SDCard

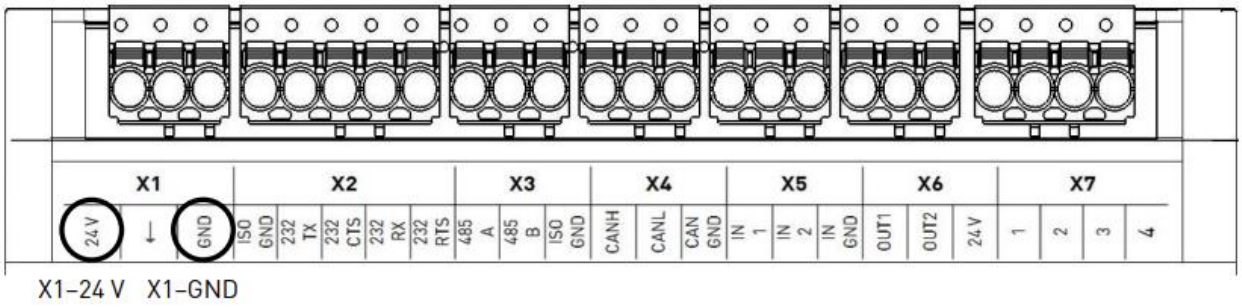
  ```
  tmp/deploy/images/dh-stm32mp1-dhcom-drc02
  ```

4.3 Setting up the HW platform

The connections needed for initial bring up of the DRC02 platform are Power, Serial and Network.

Power and Serial terminal connections are exposed on the DHCON DRC02 platform headers. 24v and GND are required from the power supply on connector X1, while TX, RX and GND on connector X2 are required for a serial interface.

Pin assignment top of the device (X1 – X7)



The host PC should be configured for 115200, 8N1 for serial port communication.

5 Initial Boot of the DRC02 Platform

5.1 Modify default boot parameters

The DHCON DRC02 platform ships with a default Linux filesystem in the on board eMMC device. The u-Boot bootloader is loaded in SPI nor memory. By default, the bootloader configuration will mount the eMMC file system. This behavior can be modified by setting the environment variables used by the bootloader.

Once the power and serial connections are established, start the preferred serial terminal software (115200, 8N1) on the host machine and apply power to the DH Platform.

Halt the bootloader by hitting any key after the message below appears.

```
U-Boot SPL 2021.01 (Oct 04 2021 - 12:16:13 +0000)
Model: DH Electronics STM32MP15xx DHCOM DRC02
Code:  SoM:rev=1,ddr3=3 Board:rev=0
RAM: DDR3L 32bits 2x4Gb 533MHz
Trying to boot from SPI


U-Boot 2021.01 (Oct 04 2021 - 12:16:13 +0000)

CPU: STM32MP157CAA Rev.Z
Model: DH Electronics STM32MP15xx DHCOM DRC02
Board: stm32mp1 in basic mode (dh,stm32mp15xx-dhcom-drc02)
DRAM:  1 GiB
Clocks:
- MPU : 650 MHz
- MCU : 208.878 MHz
- AXI : 266.500 MHz
- PER : 24 MHz
- DDR : 533 MHz
MMC:   STM32 SD/MMC: 2, STM32 SD/MMC: 0, STM32 SD/MMC: 1
```

```
Loading Environment from SPIFlash... SF: Detected w25q16cl with page size 256
Bytes, erase size 4 KiB, total 2 MiB
OK
In:    serial
Out:   serial
Err:   serial
Net:   eth0: ethernet@5800a000, eth1: ks8851mll@64000000
Hit any key to stop autoboot:  0
STM32MP>
```

Issue the following commands

```
env set boot_targets mmc0 mmc1 mmc2 usb pxe
env save
```

The board will now default to mounting the internal SDCard for the root file system.

5.2 Proper SD Card Slot

It is required to use the SDCard interface located internally on the vertically mounted SOM as shown in the upper right corner of the image below. Insert the SDCard containing the file system supporting Greengrass. In order to do this, the outer plastic shell needs to be removed.

Power cycling on the platform will result in a login prompt to the system.

The default login for the build is “root” with no password.

6 Setup your AWS account and Permissions

Refer to the instructions at Set up your AWS Account. Follow the steps outlined in the following sections to create your user account:

- Sign up for an AWS account
- Create a user and grant permissions.

Once you have followed the above steps and created the user "Administrator" and the group "Administrators", create a second user as described below.

1. Sign in to the IAM console
2. In the navigation pane, choose **Users** and then choose **Add users.**
3. For **User name**, enter **drc02_gg_user**.
4. Select the check box next to **Access Key – Programmatic access**.
5. Choose **Next: Permissions**.
6. Under **Set permissions**, choose **Add user to group**.
7. From the list of groups, select the "Administrators" group.
8. Choose **Next: Tags**.
9. Choose **Next: Review** to see the list of group memberships to be added to the new user. When you are ready to proceed, choose **Create user**.

Select "download .csv" to save the Key information for this user. This will be required in the next steps.

7 Run Greengrass on the DRC02 Platform

7.1 Boot Platform with the Greengrass Enabled File System

With the SD card in the proper slot:

- Ensure Serial terminal is available
- Power on the platform
- Default login is root with no password

Verify the version of the AWS IoT Greengrass Core software as follows:

```
java -jar /greengrass/v2/alts/init/distro/lib/Greengrass.jar --version
```

AWS Greengrass v2.4.0 should be displayed

If you do not see the /greengrass directory in the filesystem, check that you have followed section 5.1 and configured the bootloader to use mmc0 as the target filesystem.

7.2 Provide your credentials

Run the following commands to provide the credentials to the AWS IoT Greengrass Core.
The key values are found in the .CSV file that was downloaded while creating the user in step 6.

```
export AWS_ACCESS_KEY_ID=<the access key id for your account>
export AWS_SECRET_ACCESS_KEY=<the secret access key for your account>
```

7.3 Run the installer

Run the installer as shown below. Modify the values as per your region and thing name.

Use the **--provision true** option to have the installer set up the "thing" and required policies. If you prefer to configure Greengrass manually, see the online guide.

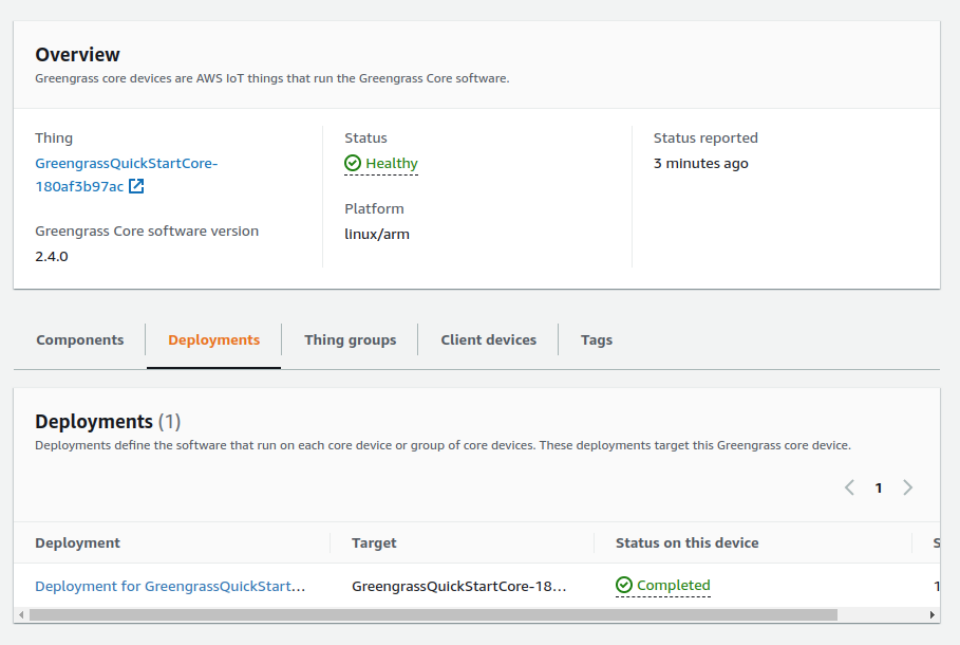
```
sudo -E java -Droot="/greengrass/v2" -Dlog.store=FILE \
  -jar /greengrass/v2/alts/init/distro/lib/Greengrass.jar \
  --aws-region us-east-1 \
  --thing-name drc02_001 \
  --thing-policy-name GreengrassV2IoTThingPolicy \
  --tes-role-name GreengrassV2TokenExchangeRole \
  --tes-role-alias-name GreengrassCoreTokenExchangeRoleAlias \
  --component-default-user ggc_user:ggc_group \
  --provision true \
  --setup-system-service true --deploy-dev-tools true
```

You will see the following output on the device console:

Successfully configured Nucleus with provisioned resource details!
Configured Nucleus to deploy aws.greengrass.Cli component
Successfully set up Nucleus as a system service

Note: The local development tools (specified by the --deploy-dev-tools option) deployment takes some time.

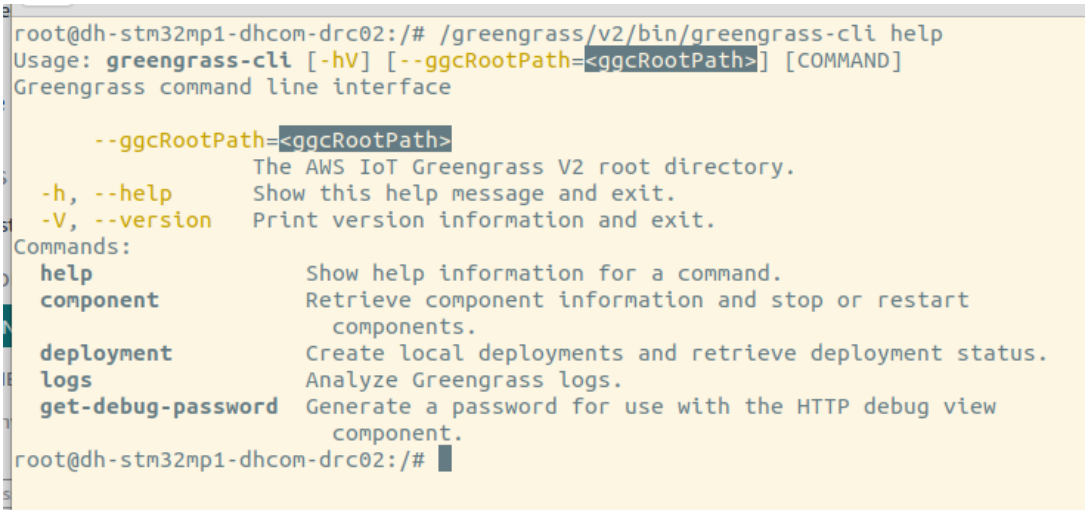
The status of the development tools can be checked at the AWS dashboard
AWS IoT -> Greengrass -> CoreDevices -> DeviceName



When the status is Completed, run the following command to verify that the Greengrass CLI is installed and runs.

```
/greengrass/v2/bin/greengrass-cli help
```

You should see output from the greengrass-cli tool that was installed during the dev-tool deployment as shown below.



There are two common problems encountered at this step.

- Insufficient permissions on the IAM->User account being used
- Incorrect export of the AWS Keys provided in the running shell

To debug, it is possible to attach the IAMFullAccess, AWSIoTFullAccess, AWSGreengrassFullAccess policies to the AWS User account used for provisioning.

8 Running Hello World Example

To confirm connectivity and functionality a Hello World example can be deployed to the platform. The steps below are taken from the AWS documentation, with a focus on Steps 5 – *Create your component in the AWS IoT Greengrass* service and 6 - *Deploy your component.*

8.1 Create your component in AWS IoT Greengrass (console)

1) Use an S3 bucket in your AWS account to host AWS IoT Greengrass component artifacts. When you deploy the component to a core device, the device downloads the component's artifacts from the bucket.
    a) In the Amazon S3 console, under Buckets, choose Create bucket.
    b) For Bucket name, enter a unique bucket name.
    c) For AWS region, select the AWS Region that you use for this tutorial.
    d) Choose Create bucket.
    e) Under Buckets, choose the bucket that you created, upload the hello_world.py script to the artifacts/com.example.HelloWorld/1.0.0 folder in the bucket.

i) The hello_world.py script can be found in section 12.1 of this document

f) Copy the S3 URI of the hello_world.py object in the S3 bucket. This URI should look like the following example. Replace DOC-EXAMPLE-BUCKET with the name of the S3 bucket. You will need this S3 bucket name in the next steps.

```
s3://DOC-EXAMPLE-
BUCKET/artifacts/com.example.HelloWorld/1.0.0/hello_world.py
```

2) Allow the core device to access component artifacts in the S3 bucket.
Each core device has a core device IAM role that allows it to interact with AWS IoT and send logs to the AWS Cloud. This device role doesn't allow access to S3 buckets by default, so you must create and attach a policy that allows the core device to retrieve component artifacts from the S3 bucket.
If your device's role already allows access to the S3 bucket, you can skip this step. Otherwise, create an IAM policy that allows access and attach it to the role, as follows:
   a) In the IAM console navigation menu, choose Policies, and then choose Create policy.
   b) On the JSON tab, replace the placeholder content with the following policy. See section 12.2 for the JSON script. It is necessary to replace DOCEXAMPLE-BUCKET with the name of the S3 bucket that contains component artifacts for the core device to download.
   c) Choose Next: Tags, and then choose Next: Review.
   d) For Name, enter MyGreengrassV2ComponentArtifactPolicy.
   e) Choose Create policy.
   f) In the IAM console navigation menu, choose Role, and then choose the name of the role for the core device. You specified this role name when you installed the AWS IoT Greengrass Core software.
   g) Under Permissions, choose Add permissions, then choose Attach policies.
   h) On the Add permissions page, select the check box next to the MyGreengrassV2ComponentArtifactPolicy policy that you created, and then choose Attach policies.
3) Use the component recipe to create a component in the AWS IoT Greengrass console.
   a) In the AWS IoT Greengrass console navigation menu, choose Components, and then choose Create component.
   b) Under Component information, choose Enter recipe as JSON. The placeholder recipe should look like the example in section 12.3 Hello World Component Recipe (JSON)
   c) Replace the placeholder URI in each Artifacts section with S3 URI of your hello_world.py object.
   d) Choose Create component.
   e) On the com.example.HelloWorld component page, verify that the Status of the component is Deployable.

At this point the S3 bucket has been created to store the Hello World artifact, the hello-world.py file has been uploaded to the S3 bucket, access has been given to the S3 bucket, and a component recipe has been created to allow for deploying the component.

8.2 Deploy to Target Device (from console)

The steps below are taken from the AWS Getting Started Tutorial, Step 6 – Deploy your component, with a focus on deploying the Hello World component from the Console interface to the target platform.

1) In the AWS IoT Greengrass console navigation menu, choose Components.
2) On the Components page, on the My components tab, choose com.example.HelloWorld.
3) On the com.example.HelloWorld page, choose Deploy.
4) From Add to deployment, choose Create new deployment, then choose Next.
5) On the Specify target page, do the following:
    a) In the Name box, enter Deployment for MyGreengrassCore.
    b) For Deployment target, choose Core device, and the name of the AWS IoT thing for your core device. The default value in this tutorial is MyGreengrassCore.
    c) Choose Next.
6) On the Select components page, under My components, verify that the com.example.HelloWorld component is selected, and choose Next.
7) On the Configure components page, choose com.example.HelloWorld, and do the following:
    a) Choose Configure component.
    b) Under Configuration update, in Configuration to merge, enter the following configuration.

```
{
     "Message": "universe"
}
```

    This configuration update sets the Hello World Message parameter to universe for the device in this deployment.
    c) Choose Confirm.
    d) Choose Next.
8) On the Configure advanced settings page, keep the default configuration settings, and choose Next.
9) On the Review page, choose Deploy.
10) Verify that the deployment completes successfully. The deployment can take several minutes to complete. Check the Hello World log to verify the change. Run the following command on your Greengrass core device.

On the target platform

```
sudo tail -f /greengrass/v2/logs/com.example.HelloWorld.log
```

```
root@dh-stm32mp1-dhcom-drc02:~# tail -f /greengrass/v2/logs/
aws.greengrass.Nucleus.log  com.example.HelloWorld.log  greengrass.log             main.log
root@dh-stm32mp1-dhcom-drc02:~# tail -f /greengrass/v2/logs/com.example.HelloWorld.log
2022-05-15T16:13:41.985Z [INFO] (pool-2-thread-22) com.example.HelloWorld: shell-runner-start.
{scriptName=services.com.example.HelloWorld.lifecycle.Run, serviceName=com.example.HelloWorld,
currentState=STARTING, command=["python3 -u /greengrass/v2/packages/artifacts/com.example.Hello
World/1.0.0/hell..."]}
2022-05-15T16:13:42.493Z [INFO] (Copier) com.example.HelloWorld: stdout. Hello, Universe! Greet
ings from your first cloud deployed Greengrass component.. {scriptName=services.com.example.Hel
loWorld.lifecycle.Run, serviceName=com.example.HelloWorld, currentState=RUNNING}
2022-05-15T16:13:42.529Z [INFO] (Copier) com.example.HelloWorld: Run script exited. {exitCode=0
, serviceName=com.example.HelloWorld, currentState=RUNNING}
^C
root@dh-stm32mp1-dhcom-drc02:~#
```

9 Additional Platform Configuration

9.1 Wireless Configuration

An additional Yocto layer is provided to allow for ease of configuring a wireless network. This can be found in the /layers/meta-amefae-connectivity/ directory.

The file listed below can be modified with the SSID and password required for the wireless network connection.

```
layers/meta-amefae-connectivity/recipes-connectivity/wpa-supplicant/wpa_supplicant-
wlan0.conf
```

This file can also be modified directly on the file system in the /etc/wpa_supplicant/ directory.

9.2 Bluetooth Tools

The BlueZ and HCI tools are installed as part of the Yocto build. By default, the Bluetooth functionality is disabled.
To enable Bluetooth functionality

- echo "options rsi_sdio dev_oper_mode=13" > /etc/modprobe.d/rsi_sdio.conf
- Reboot the platform

Once rebooted, the hciconfig command will show the available device.

```
root@dh-stm32mp1-dhcom-drc02:~# hciconfig
hci0:   Type: Primary  Bus: SDIO
        BD Address: 80:C9:55:C2:53:1F  ACL MTU: 1021:3  SCO MTU: 64:3
        DOWN
        RX bytes:651 acl:0 sco:0 events:39 errors:0
        TX bytes:0 acl:0 sco:0 commands:39 errors:0

root@dh-stm32mp1-dhcom-drc02:~#
```

This device can be brought up by the command

- `hciconfig hci0 up`

Then bluetoothctl can be used to power on the device and control scanning and pairing.

9.3 Additional Configuration for AWS IDT suite

To run the IDT Suite against the platform there are additional steps to take.

- add ggc_user to sudoers file
  echo "ggc_user ALL=(ALL) NOPASSWD: ALL" >> /etc/sudoers

- mkdir /home/ggc_user

- chown ggc_user:ggc_group /home/ggc_user

- change password for ggc_user
  passwd ggc_user

- Ensure the shell listed for ggc_user in /etc/passwd is correct
  ggc_user 996:996::/home/ggc_user:/bin/sh

- from HOST PC
  ssh-copy-id ggc_user@Platform_IP_Address

10 Debugging

The serial connection detailed above will provide terminal access to the platform, the default user login is root, with no password set.

An ssh server is also started by default on the platform, this is also configured to allow root login and can be accessed by the command “ssh root@platform_IP_address”.

If difficulties arise in booting to the generated file system it is possible to boot to the file system image contained in eMMC and then mount the SD Card in order to further investigate any issues caused by the generated file system.

11 Hello World Scripts and Files

The scripts and files in this section are from the AWS getting started tutorial

11.1 Hello World Python Script

```
import sys

message = "Hello, %s!" % sys.argv[1]
message += " Greetings from your first cloud deployed Greengrass component."

# Print the message to stdout, which Greengrass saves in a log file.
print(message)
```

11.2 JSON for IAM policy to allow S3 Bucket access

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
          "Effect": "Allow",
           "Action": [
               "s3:GetObject"
            ],
            "Resource": "arn:aws:s3:::DOC-EXAMPLE-BUCKET/*"
      }
    ]
}
```

11.3 Hello world Component Recipe (JSON)

```
 {
 "RecipeFormatVersion": "2020-01-25",
 "ComponentName": "com.example.HelloWorld",
 "ComponentVersion": "1.0.0",
 "ComponentDescription": "My first AWS IoT Greengrass component.",
 "ComponentPublisher": "Amazon",
 "ComponentConfiguration": {
  "DefaultConfiguration": {
   "Message": "world"
  }
 },
 "Manifests": [
  {
   "Platform": {
    "os": "linux"
   },
   "Lifecycle": {
    "Run": "python3 -u {artifacts:path}/hello_world.py \"{configuration:/Message}\""
   },
   "Artifacts": [
    {
```

```
        "URI": "s3://DOC-EXAMPLE-
BUCKET/artifacts/com.example.HelloWorld/1.0.0/hello_world.py"
      }
    ]
  },
  {
    "Platform": {
      "os": "windows"
    },
    "Lifecycle": {
      "Run": "py -3 -u {artifacts:path}/hello_world.py \"{configuration:/Message}\""
    },
    "Artifacts": [
      {
        "URI": "s3://DOC-EXAMPLE-
BUCKET/artifacts/com.example.HelloWorld/1.0.0/hello_world.py"
      }
    ]
  }
 ]
}
```