Doosan Robot

M0609 | M0617 | M1013 | M1509

ROS Programming Manual

1. Doosan Robotics ROS in AWS 3

2. Setup development environment 4

3. Running the application on simulator 8

4. Create robot application... 19

5. Create fleet 23

6. Register a robot to AWS RoboMaker 24

7. Setup the ROS Master PC 28

8. Build & Bundle robot app.. 30

9. Deploy to a robot 31

10. References 34

1. Doosan Robotics ROS in AWS

- This document describes how to use the Doosan robotics ROS package with AWS RoboMaker.

- This document only covers the basics of robot simulation, robot app creation, and deployment of Doosan Robotics ROS packages in an AWS RoboMaker environment. For usage or applications outside this topic, please refer to the AWS Robomaker manual or detailed documentation.

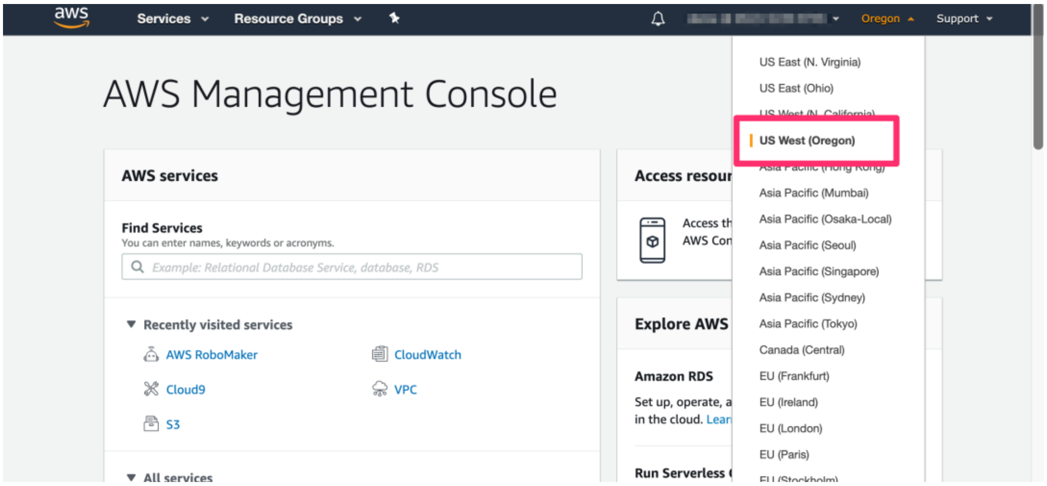
  https://aws.amazon.com/robomaker

- Try the Doosan Robotics ROS package in your local environment, not in the AWS cloud environment, so you can understand what you see more quickly and clearly.

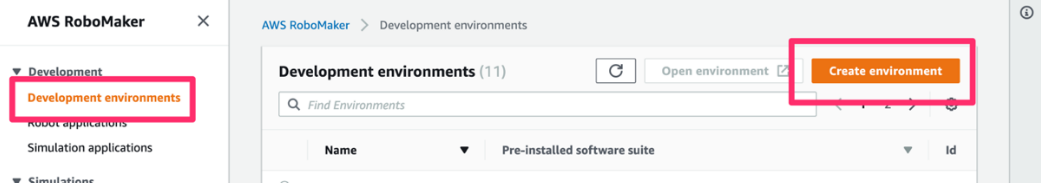
  https://github.com/doosan-robotics/doosan-robot

2. Setup development environment

- **[2.1]** Open AWS Management Console (**https://console.aws.amazon.com**). For region, please choose **US West Oregon** this time.)

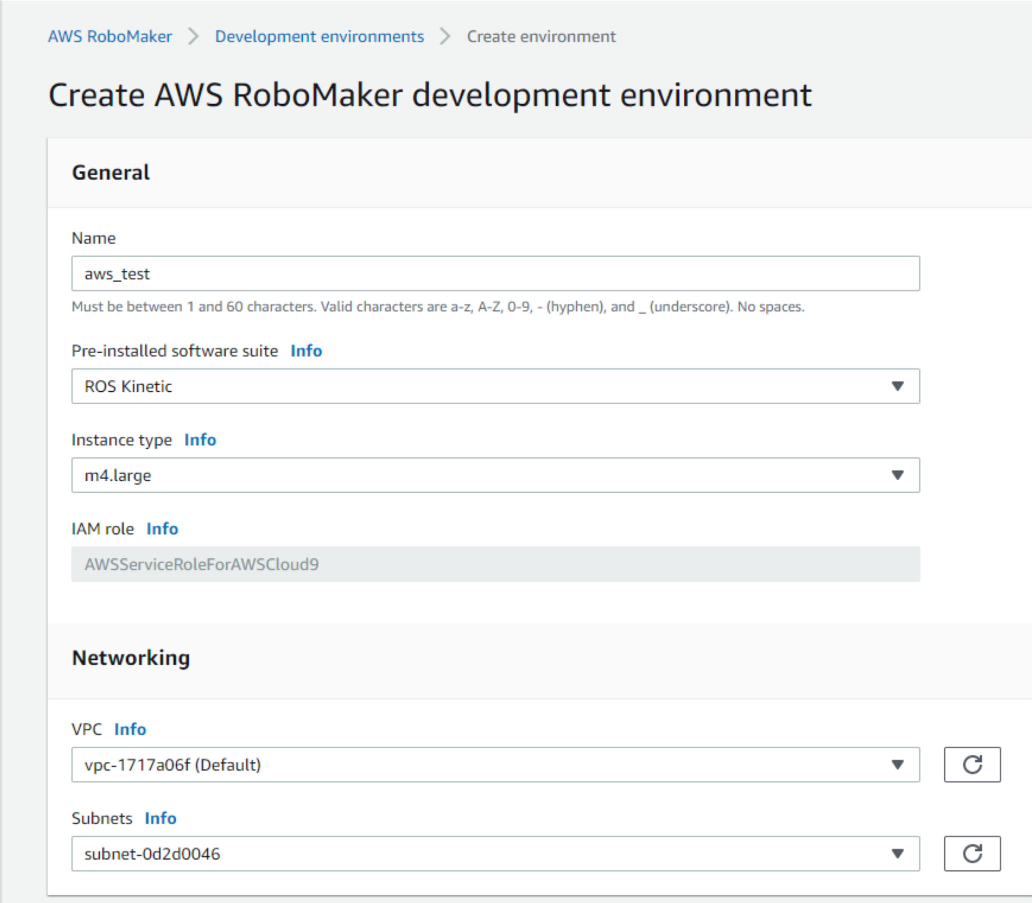


- **[2.2]** Select AWS RoboMaker from services and from left side navigation pain, select ***Development ->Development environments***. Click **[Create environment]** button on top right of the "Development environment" page.



- **[2.3]** Set "Name" of your development environment. Choose **ROS Kinetic** as the Pre-installed software suite. Choose **m4.large** for Instance type. For Networking settings, choose default VPC from the list and select one subnet from the list shown after selecting VPC. Press **[Create]** button on bottom right. Development environment now start coming up. (It takes around 2~3 minutes.)

  - If the environment creation fails, clear the creation environment and select another subnet to proceed.


AWS RoboMaker > Development environments > Create environment

Create AWS RoboMaker development environment

General

Name: aws_test

Must be between 1 and 60 characters. Valid characters are a-z, A-Z, 0-9, - (hyphen), and _ (underscore). No spaces.

Pre-installed software suite: ROS Kinetic

Instance type: m4.large

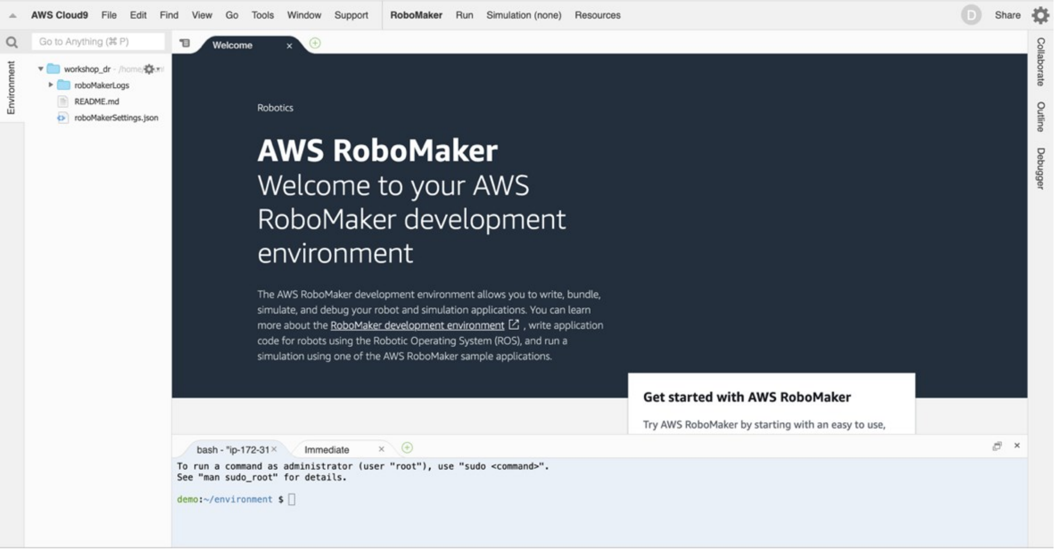
IAM role: AWSServiceRoleForAWSCloud9

Networking

VPC: vpc-1717a06f (Default)

Subnets: subnet-0d2d0046

- **[2.4]** RoboMaker development environment which is based on Cloud9(AWS's Cloud IDE service) will now be launched.



Robotics

AWS RoboMaker
Welcome to your AWS RoboMaker development environment

The AWS RoboMaker development environment allows you to write, bundle, simulate, and debug your robot and simulation applications. You can learn more about the RoboMaker development environment, write application code for robots using the Robotic Operating System (ROS), and run a simulation using one of the AWS RoboMaker sample applications.

Get started with AWS RoboMaker

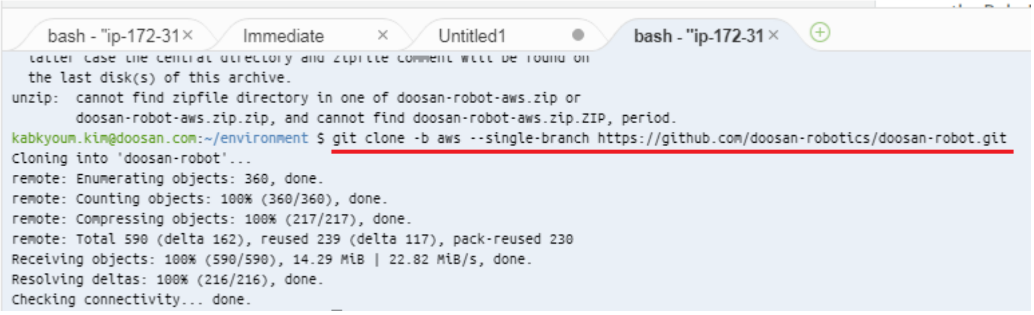
Try AWS RoboMaker by starting with an easy to use,

```
To run a command as administrator (user "root"), use "sudo <command>".
See "man sudo_root" for details.

demo:~/environment $
```

- **[2.6]** Enter the following command in the terminal window to upload the Doosan ROS package source to the created cloud environment.

```
git clone -b aws --single-branch https://github.com/doosan-robotics/doosan-robot.git
```



```
  latter case the central directory and zipfile comment will be found on
  the last disk(s) of this archive.
unzip:  cannot find zipfile directory in one of doosan-robot-aws.zip or
        doosan-robot-aws.zip.zip, and cannot find doosan-robot-aws.zip.ZIP, period.
kabkyoum.kim@doosan.com:~/environment $ git clone -b aws --single-branch https://github.com/doosan-robotics/doosan-robot.git
Cloning into 'doosan-robot'...
remote: Enumerating objects: 360, done.
remote: Counting objects: 100% (360/360), done.
remote: Compressing objects: 100% (217/217), done.
remote: Total 590 (delta 162), reused 239 (delta 117), pack-reused 230
Receiving objects: 100% (590/590), 14.29 MiB | 22.82 MiB/s, done.
Resolving deltas: 100% (216/216), done.
Checking connectivity... done.
```

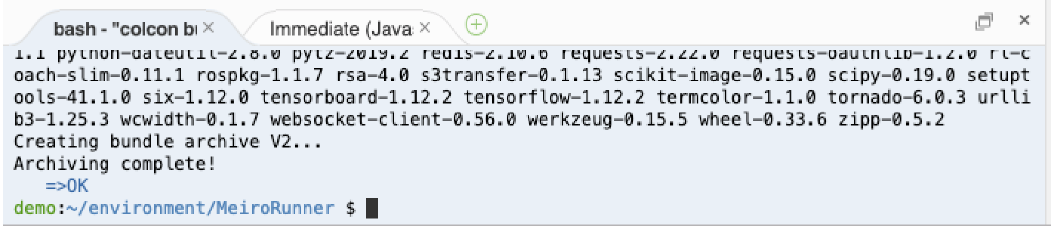
- **[2.7]** Enter following commands to run the setup script

```
cd doosan-robot

chmod +x setup.sh

./setup.sh
```

- **[2.8]** Setup script will be started. It will take ~30 min to complete. When completed, terminal will show something like this:



```
1.1 python-dateutil-2.8.0 pytz-2019.2 redis-2.10.6 requests-2.22.0 requests-oauthlib-1.2.0 rl-c
oach-slim-0.11.1 rospkg-1.1.7 rsa-4.0 s3transfer-0.1.13 scikit-image-0.15.0 scipy-0.19.0 setupt
ools-41.1.0 six-1.12.0 tensorboard-1.12.2 tensorflow-1.12.2 termcolor-1.1.0 tornado-6.0.3 urlli
b3-1.25.3 wcwidth-0.1.7 websocket-client-0.56.0 werkzeug-0.15.5 wheel-0.33.6 zipp-0.5.2
Creating bundle archive V2...
Archiving complete!
   =>OK
demo:~/environment/MeiroRunner $ 
```

The script you just executed does following.
- Update the development environment.
- Prepare the resources.
  . S3 bucket to store the bundle and the output of simulation jobs.
   (S3 bucket name : robomaker-ws-us-west-2-[AWS Account Number]-[YYMMDD] –[HHMMSS]
   e.g.: robomaker-ws-us-west-2-123456789012-190808-135139)
  . IAM role for running simulation.

. IAM role for deploy the code to robots.

. Name of robot and simulation application.

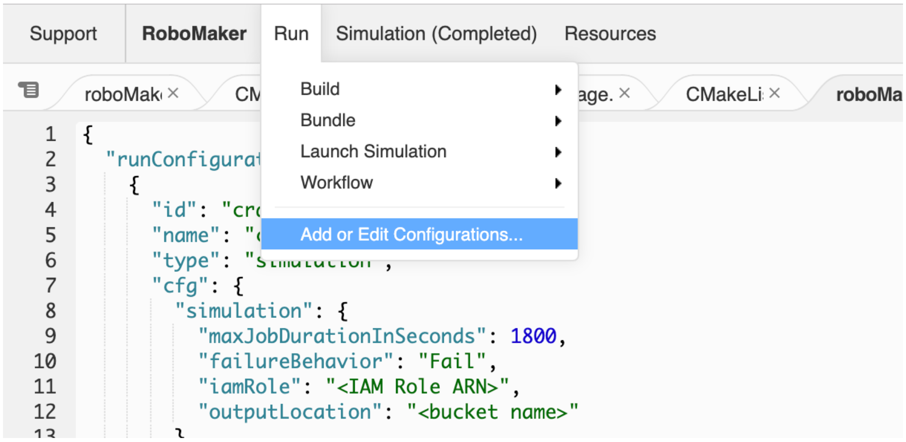
. Update project setting file roboMakerSettings.json to match with the resources prepared by this setup script.

. Execute initial build and bundle source code.

3. Running the application on simulator

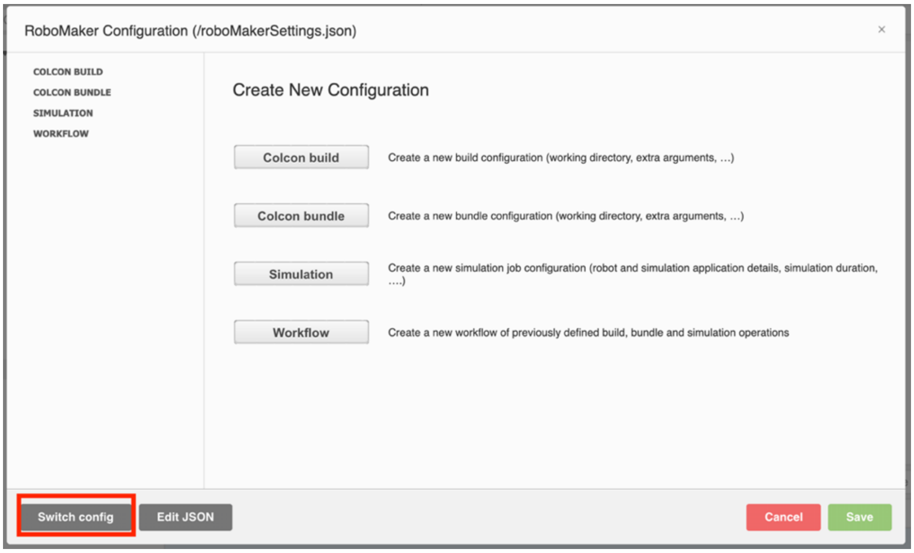
- **[3.1]** Load the development environment configuration file to the development environment.

  - From menu, choose ***Run -> Add or Edit Configurations...***



```
{
  "runConfigurat
    {
      "id": "cr
      "name": "
      "type": "simulation",
      "cfg": {
        "simulation": {
          "maxJobDurationInSeconds": 1800,
          "failureBehavior": "Fail",
          "iamRole": "<IAM Role ARN>",
          "outputLocation": "<bucket name>"
        }
```

- **"RoboMaker Configuration"** window will be opened. Click **[Switch Config]** on bottom left



Create New Configuration

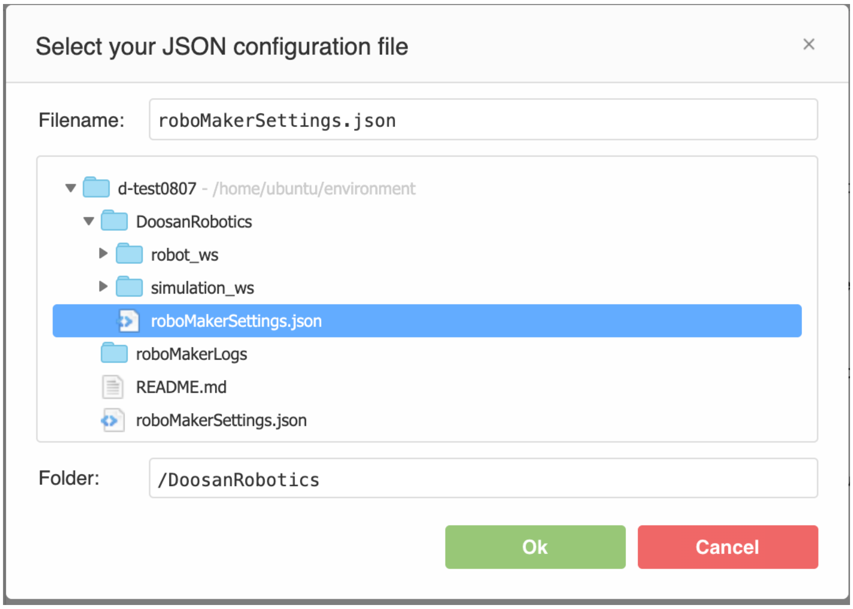
Colcon build — Create a new build configuration (working directory, extra arguments, ...)

Colcon bundle — Create a new bundle configuration (working directory, extra arguments, ...)

Simulation — Create a new simulation job configuration (robot and simulation application details, simulation duration, ...)

Workflow — Create a new workflow of previously defined build, bundle and simulation operations

- **Open the doosan-robot directory** and select the **roboMakerSettings.json** file we just edited. Click **[OK]** button, then click [Save] button on bottom right of **"RoboMaker Configuration"** window.



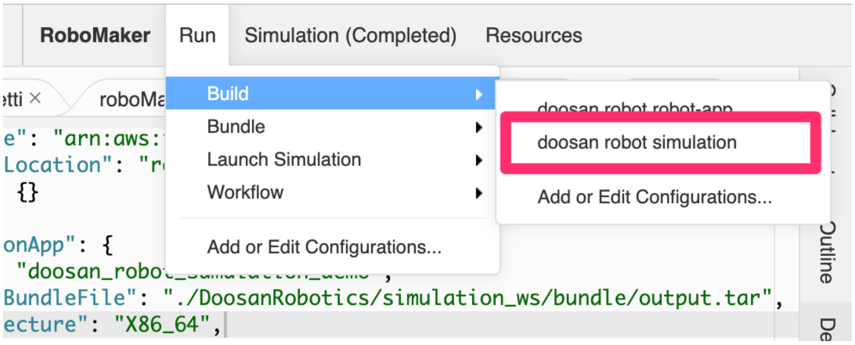
Select your JSON configuration file ×

Filename: roboMakerSettings.json

- d-test0807 - /home/ubuntu/environment
  - DoosanRobotics
    - robot_ws
    - simulation_ws
    - roboMakerSettings.json
  - roboMakerLogs
  - README.md
  - roboMakerSettings.json

Folder: /DoosanRobotics

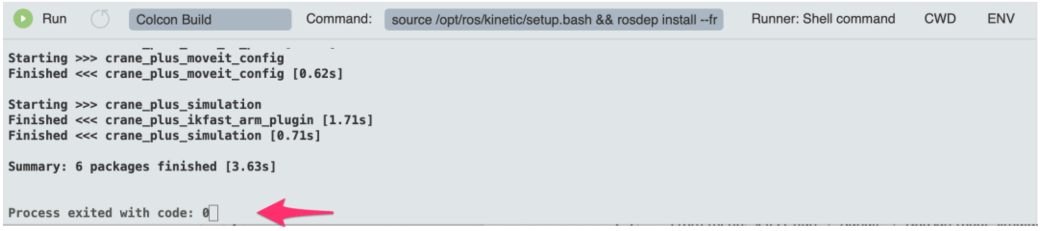
- **[3.2]** You will now have menu items for this workshop project. From menu, select ***Run -> Build -> doosan robot simulation***. This will build your simulation application.



```
e": "arn:aws:
Location": "r
 {}

onApp": {
 "doosan_robo
BundleFile": "./DoosanRobotics/simulation_ws/bundle/output.tar",
ecture": "X86_64",
```

Running for building Doosan robot simulation results are as below;



Command: source /opt/ros/kinetic/setup.bash && rosdep install --fr

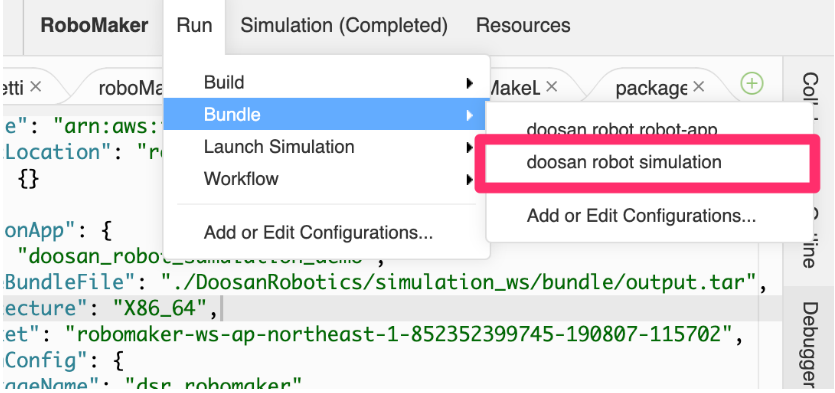
```
Starting >>> crane_plus_moveit_config
Finished <<< crane_plus_moveit_config [0.62s]

Starting >>> crane_plus_simulation
Finished <<< crane_plus_ikfast_arm_plugin [1.71s]
Finished <<< crane_plus_simulation [0.71s]

Summary: 6 packages finished [3.63s]

Process exited with code: 0
```

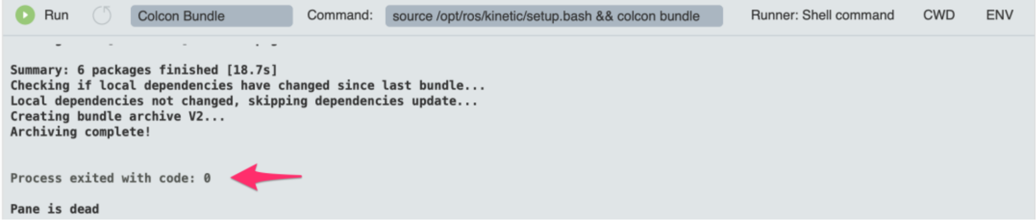
- **[3.3]** From menu, select ***Run -> Bundle -> Doosan robot simulation***. This will create **bundle** for the application. **Bundle** is the process to archive a ROS application. RoboMaker launches applications in simulation environment / robot hardware using file crated by **bundle**. Bundled file is downloading to the environment and extracted there for running it on the simulation environment / robot hardware.



```
e": "arn:aws:
Location": "r
 {}

onApp": {
 "doosan_robo
BundleFile": "./DoosanRobotics/simulation_ws/bundle/output.tar",
ecture": "X86_64",
et": "robomaker-ws-ap-northeast-1-852352399745-190807-115702",
Config": {
ageName": "dsr_robomaker"
```

Running for bundling Doosan robot simulation results are as below;



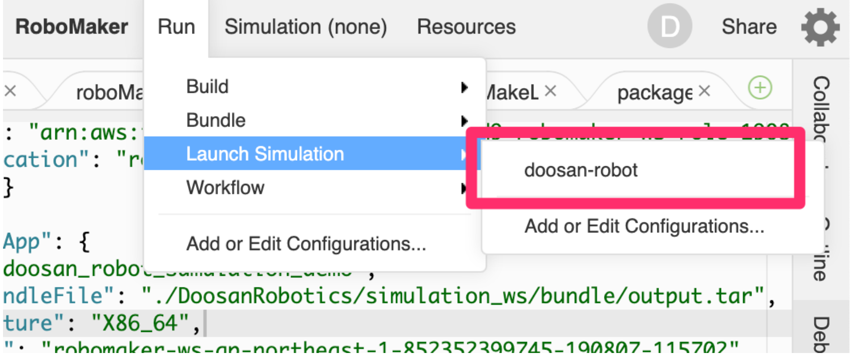
Command: source /opt/ros/kinetic/setup.bash && colcon bundle

```
Summary: 6 packages finished [18.7s]
Checking if local dependencies have changed since last bundle...
Local dependencies not changed, skipping dependencies update...
Creating bundle archive V2...
Archiving complete!

Process exited with code: 0

Pane is dead
```

- **[3.4]** From menu, select ***Run -> Launch Simulation -> doosan-robot***. This will launch the simulation application.

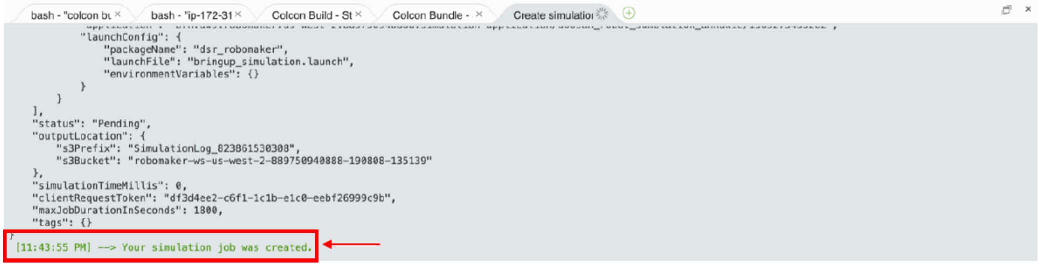


```
: "arn:aws:
cation": "r
}

App": {
doosan_robo
ndleFile": "./DoosanRobotics/simulation_ws/bundle/output.tar",
ture": "X86_64",
": "robomaker-ws-ap-northeast-1-852352399745-190807-115702"
```

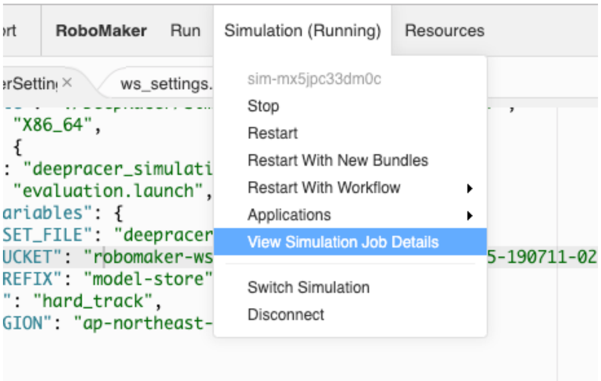
Behind the scene, this operation will copy bundled file to **S3**(cloud file storage) and launch a simulation job. Simulation job then load the file from the S3, extract it and launch the application.

Launching Doosan-robot simulation job results are as below;



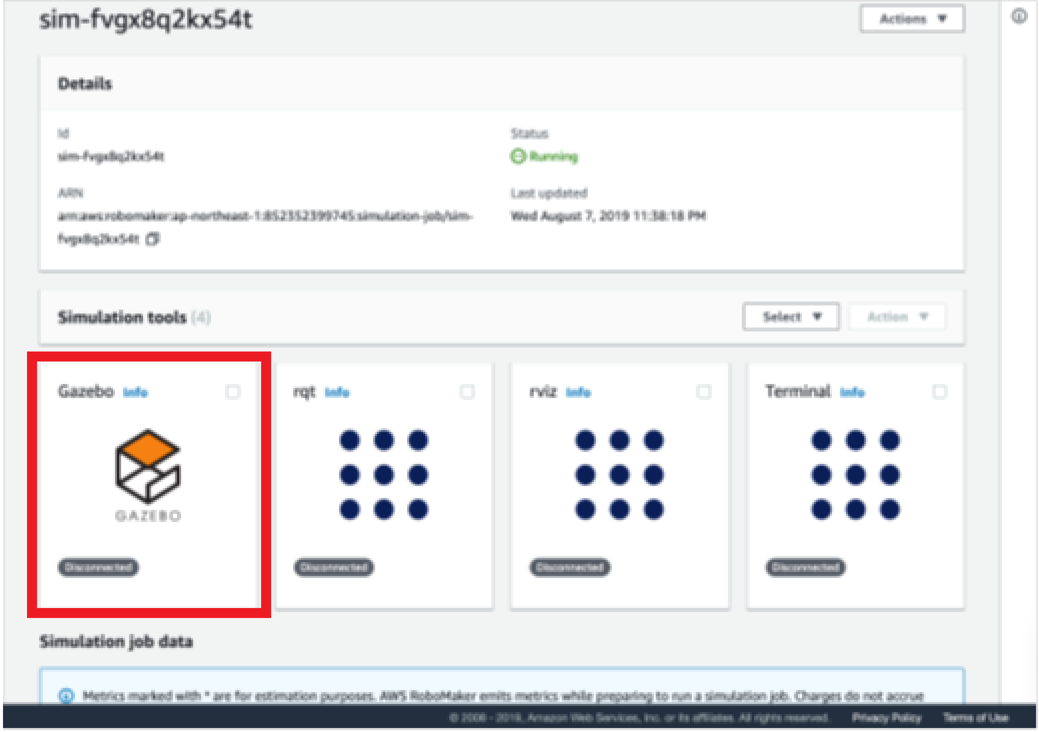
```
            "launchConfig": {
                "packageName": "dsr_robomaker",
                "launchFile": "bringup_simulation.launch",
                "environmentVariables": {}
            }
        }
    ],
    "status": "Pending",
    "outputLocation": {
        "s3Prefix": "SimulationLog_823861530308",
        "s3Bucket": "robomaker-ws-us-west-2-889750940888-190808-135139"
    },
    "simulationTimeMillis": 0,
    "clientRequestToken": "df3d4ee2-c6f1-1c1b-e1c0-eebf26999c9b",
    "maxJobDurationInSeconds": 1800,
    "tags": {}
}
[11:43:55 PM] --> Your simulation job was created.
```

- **[3.5]** Simulation job is now bringing up. From menu, select ***Simulation -> View Simulation Job Details***. This will open simulation job detail page.



```
"X86_64",
{
: "deepracer_simulati
"evaluation.launch",
ariables": {
SET_FILE": "deepracer
UCKET": "robomaker-ws                5-190711-02
REFIX": "model-store"
": "hard_track",
GION": "ap-northeast-
```

- **[3.6]** After waiting for about 3 minutes, the **status** will change from **'preparing'** to **'Running'** and the status will be as below.



sim-fvgx8q2kx54t

Details

Id
sim-fvgx8q2kx54t

Status
Running

ARN
arn:aws:robomaker:ap-northeast-1:852352399745:simulation-job/sim-fvgx8q2kx54t

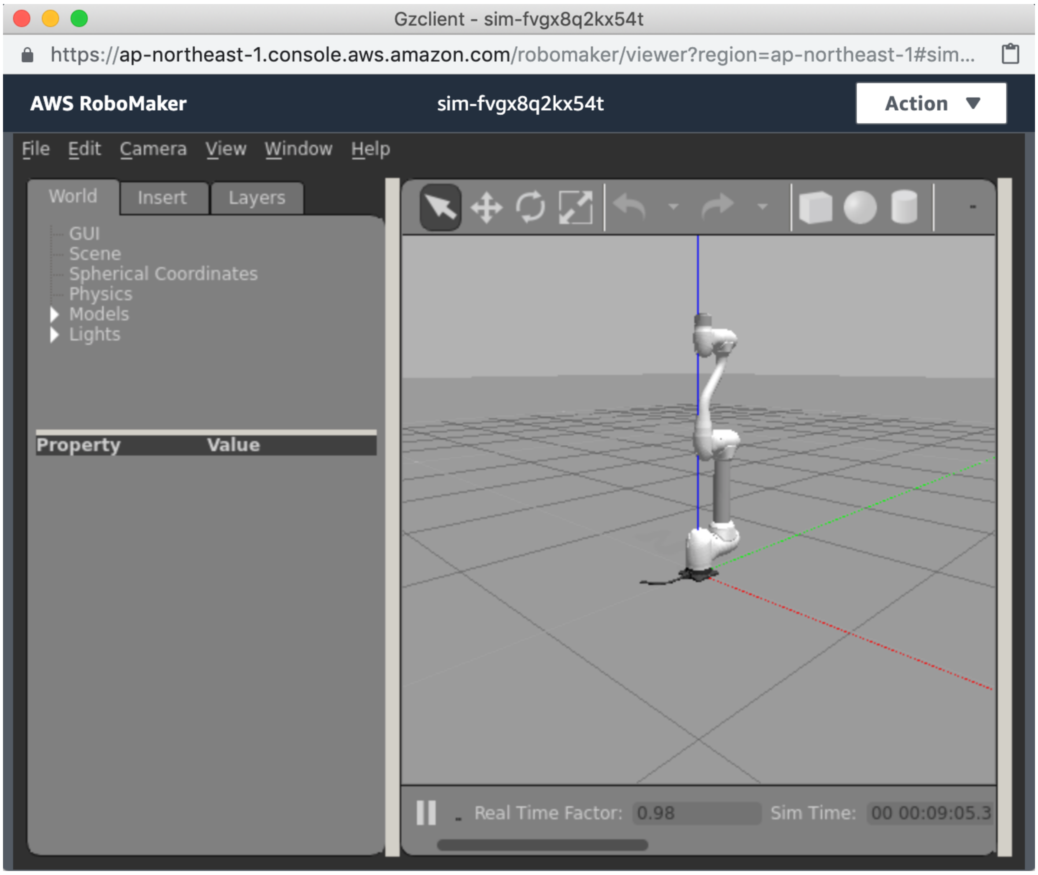
Last updated
Wed August 7, 2019 11:38:18 PM

Simulation tools (4)

Simulation job data

Metrics marked with * are for estimation purposes. AWS RoboMaker emits metrics while preparing to run a simulation job. Charges do not accrue

- **Gazebo** is a simulator simulating ROS applications. By clicking **Gazebo** icon, you can interact with the window of Gazebo simulation.



- Model of the robot is loaded and simulation is launched.

Note:

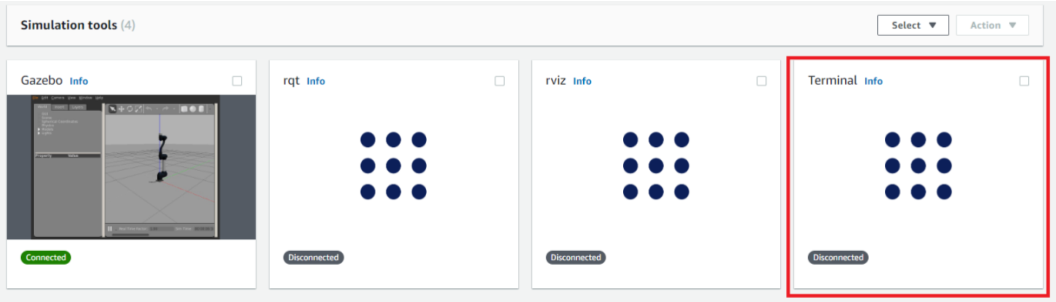
simulation_ws/dsr_launcher /launch/**aws_bringup_simulation.launch**.

This file is use to launch simulation application. It's specified in **roboMakerSettings.json** file as follow.

```
…
"simulationApp": {
  "name": "doosan_robot_sumulation_annakie",
  "sourceBundleFile": "./doosan-robot/simulation_ws/bundle/output.tar",
  "architecture": "X86_64",
  "s3Bucket": "robomaker-ws-us-west-2-000000000000-000000-000000",
  "launchConfig": {
  "packageName": "dsr_launcher",
  "launchFile": "aws_bringup_simulation.launch",
  "environmentVariables": {}
  },
  "simulationSoftwareSuite": {
  "name": "Gazebo",
  "version": "7"
  },
  "renderingEngine": {
  "name": "OGRE",
  "version": "1.x"
  },
  "robotSoftwareSuite": {
  "name": "ROS",
  "version": "Kinetic"
  }
}
…
```

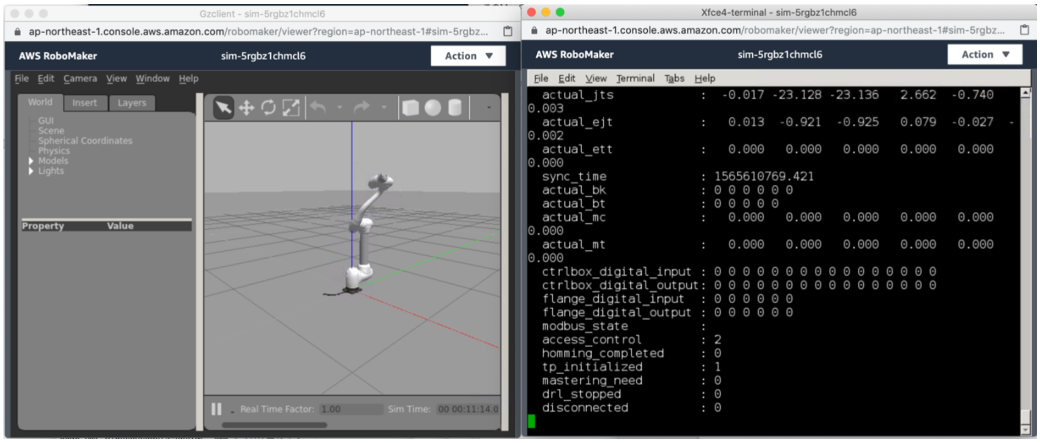
- It’s ready, you can try ROS application controls doosan robots.

[3.7] Running the example programs Click the terminal icon to launch a terminal window.



- type following command.

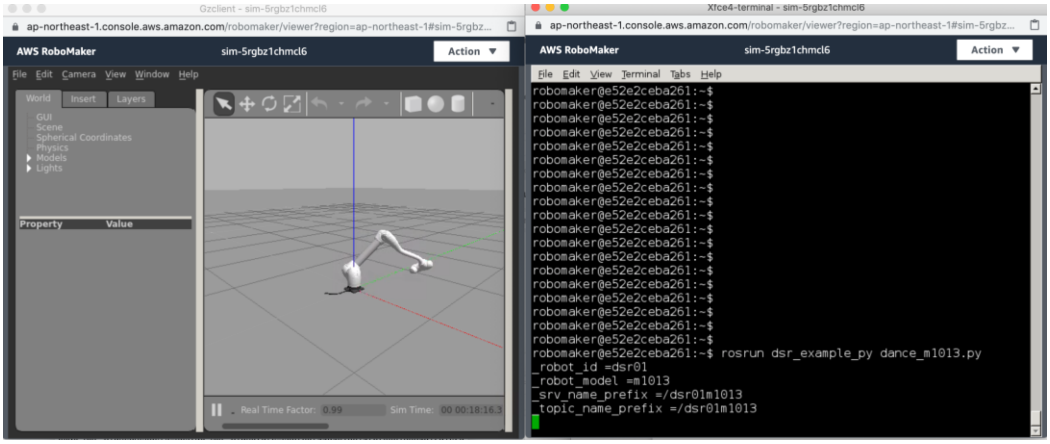
```
rosrun dsr_example_py single_robot_simple.py
```


Gzclient - sim-5rgbz1chmcl6
ap-northeast-1.console.aws.amazon.com/robomaker/viewer?region=ap-northeast-1#sim-5rgbz...
AWS RoboMaker
sim-5rgbz1chmcl6
Action ▼
File Edit Camera View Window Help
World Insert Layers
GUI
Scene
Spherical Coordinates
Physics
Models
Lights
Property Value
Real Time Factor: 1.00
Sim Time: 00 00:11:14.6
Xfce4-terminal - sim-5rgbz1chmcl6
ap-northeast-1.console.aws.amazon.com/robomaker/viewer?region=ap-northeast-1#sim-5rgbz...
AWS RoboMaker
sim-5rgbz1chmcl6
Action ▼
File Edit View Terminal Tabs Help

```
  actual_jts            :  -0.017 -23.128 -23.136   2.662  -0.740
0.003
  actual_ejt            :   0.013  -0.921  -0.925   0.079  -0.027  -
0.002
  actual_ett            :   0.000   0.000   0.000   0.000   0.000
0.000
  sync_time             : 1565610769.421
  actual_bk             : 0 0 0 0 0 0
  actual_bt             : 0 0 0 0 0
  actual_mc             :   0.000   0.000   0.000   0.000   0.000
0.000
  actual_mt             :   0.000   0.000   0.000   0.000   0.000
0.000
  ctrlbox_digital_input : 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
  ctrlbox_digital_output: 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
  flange_digital_input  : 0 0 0 0 0 0
  flange_digital_output : 0 0 0 0 0 0
  modbus_state          :
  access_control        : 2
  homming_completed     : 0
  tp_initialized        : 1
  mastering_need        : 0
  drl_stopped           : 0
  disconnected          : 0
```

- In the terminal window, press Ctl + c to stop single_robot_simle.py and enter the following command:

```
rosrun dsr_example_py dance_m1013.py
```


Gzclient - sim-5rgbz1chmcl6
ap-northeast-1.console.aws.amazon.com/robomaker/viewer?region=ap-northeast-1#sim-5rgbz...
AWS RoboMaker
sim-5rgbz1chmcl6
Action
File Edit Camera View Window Help
World Insert Layers
GUI
Scene
Spherical Coordinates
Physics
Models
Lights
Property Value
Real Time Factor: 0.99
Sim Time: 00 00:18:16.3
Xfce4-terminal - sim-5rgbz1chmcl6
ap-northeast-1.console.aws.amazon.com/robomaker/viewer?region=ap-northeast-1#sim-5rgbz...
AWS RoboMaker
sim-5rgbz1chmcl6
Action
File Edit View Terminal Tabs Help

```
robomaker@e52e2ceba261:~$
robomaker@e52e2ceba261:~$
robomaker@e52e2ceba261:~$
robomaker@e52e2ceba261:~$
robomaker@e52e2ceba261:~$
robomaker@e52e2ceba261:~$
robomaker@e52e2ceba261:~$
robomaker@e52e2ceba261:~$
robomaker@e52e2ceba261:~$
robomaker@e52e2ceba261:~$
robomaker@e52e2ceba261:~$
robomaker@e52e2ceba261:~$
robomaker@e52e2ceba261:~$
robomaker@e52e2ceba261:~$
robomaker@e52e2ceba261:~$
robomaker@e52e2ceba261:~$
robomaker@e52e2ceba261:~$
robomaker@e52e2ceba261:~$
robomaker@e52e2ceba261:~$
robomaker@e52e2ceba261:~$ rosrun dsr_example_py dance_m1013.py
_robot_id =dsr01
_robot_model =m1013
_srv_name_prefix =/dsr01m1013
_topic_name_prefix =/dsr01m1013
```

- You can launch multiple robots by changing simulation configuration.

Change **aws_bringup_simulation.launch** as follow,

(simulation_ws/src/dsr_launcher/launch/aws_bringup_simulation.launch)

```
<?xml version="1.0"?>
<launch>
  <!-- node name="drcf" pkg="common" type="run_drcf.sh" output="screen" 
required="true"/-->
  </include>
  <!-- Start Gazebo with an empty world. -->
  <!--include file="$(find dsr_launcher)/launch/single_robot_gazebo.launch"-->
  <include file="$(find dsr_launcher)/launch/multi_robot_gazebo.launch">
    <arg name="mode"     value="virtual"/>
    <arg name="model"    value="m1013"/>
    <arg name="color"    value="white"/>
    <arg name="gripper" value="none"/>
    <arg name="mobile"   value="none"/>
  </include>
</launch>
```

- do build and bundle simulation again.

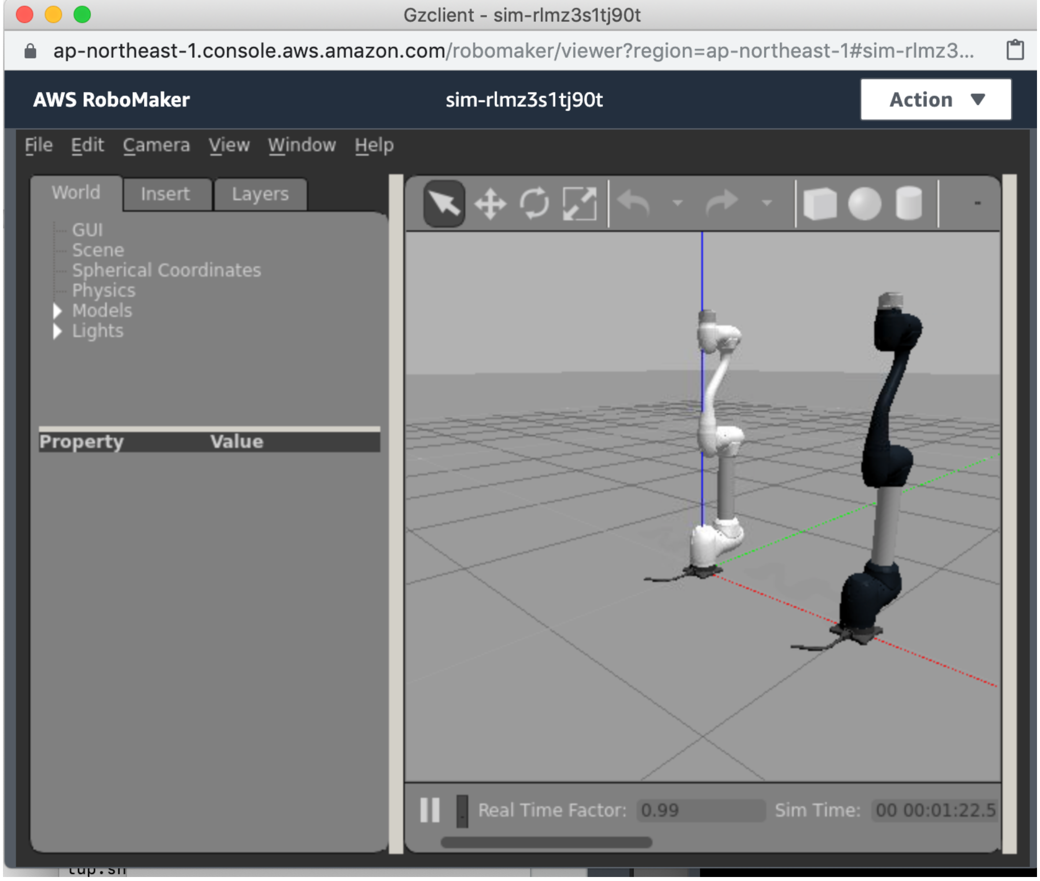
  Run -> Build -> doosan robot simulation.

  Run -> Bundle -> doosan robot simulation.

- By launching simulation, you will now see two robots in Gazebo.

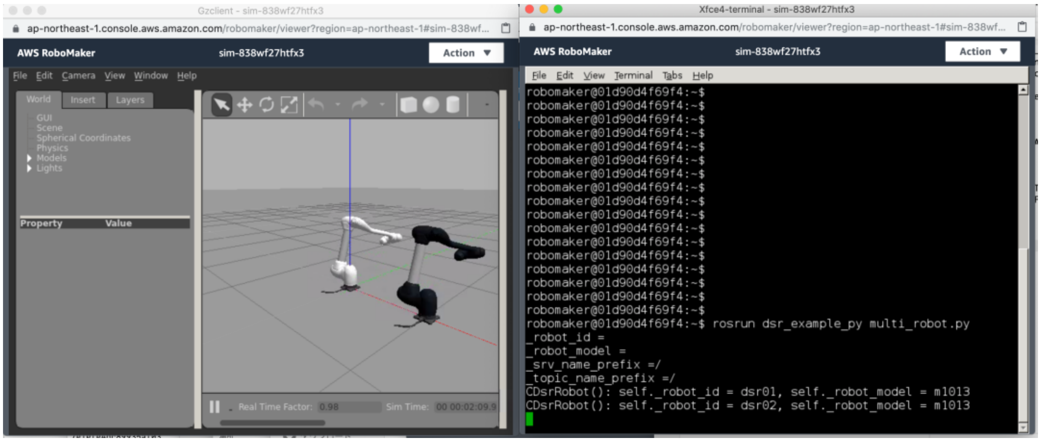
  Run -> Launch Simulation-> doosan-robot.

  Simulation -> View Simulation Job Details.



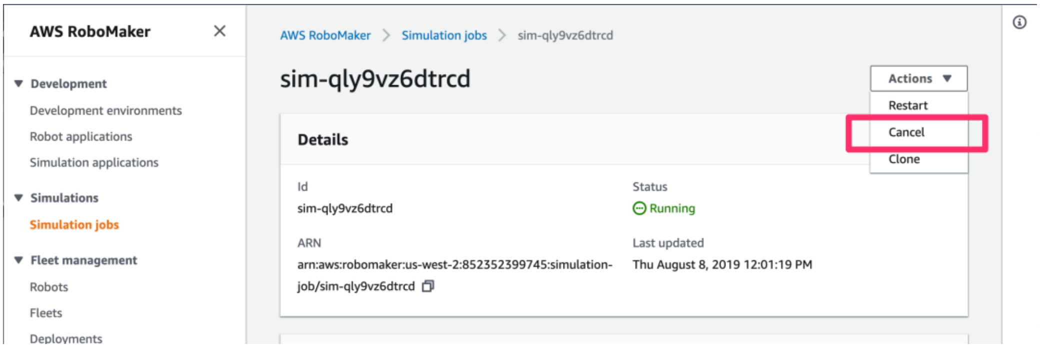
- Click Terminal icon to launch terminal window and type following command.

```
rosrun dsr_example_py multi_robot.py
```


Gzclient - sim-838wf27htfx3
ap-northeast-1.console.aws.amazon.com/robomaker/viewer?region=ap-northeast-1#sim-838wf...
AWS RoboMaker
sim-838wf27htfx3
Action ▼
File Edit Camera View Window Help
World Insert Layers
GUI
Scene
Spherical Coordinates
Physics
Models
Lights
Property Value
Real Time Factor: 0.98
Sim Time: 00 00:02:09.9
Xfce4-terminal - sim-838wf27htfx3
ap-northeast-1.console.aws.amazon.com/robomaker/viewer?region=ap-northeast-1#sim-838wf...
AWS RoboMaker
sim-838wf27htfx3
Action ▼
File Edit View Terminal Tabs Help

```
robomaker@01d90d4f69f4:~$
robomaker@01d90d4f69f4:~$
robomaker@01d90d4f69f4:~$
robomaker@01d90d4f69f4:~$
robomaker@01d90d4f69f4:~$
robomaker@01d90d4f69f4:~$
robomaker@01d90d4f69f4:~$
robomaker@01d90d4f69f4:~$
robomaker@01d90d4f69f4:~$
robomaker@01d90d4f69f4:~$
robomaker@01d90d4f69f4:~$
robomaker@01d90d4f69f4:~$
robomaker@01d90d4f69f4:~$
robomaker@01d90d4f69f4:~$
robomaker@01d90d4f69f4:~$
robomaker@01d90d4f69f4:~$
robomaker@01d90d4f69f4:~$
robomaker@01d90d4f69f4:~$ rosrun dsr_example_py multi_robot.py
_robot_id =
_robot_model =
_srv_name_prefix =/
_topic_name_prefix =/
CDsrRobot(): self._robot_id = dsr01, self._robot_model = m1013
CDsrRobot(): self._robot_id = dsr02, self._robot_model = m1013
```

- **[3.8]** To finish the simulation, go back to the simulation job detail page and press **[Actions] -> [Cancel]**



sim-qly9vz6dtrcd

Details

Id
sim-qly9vz6dtrcd

Status
Running

ARN
arn:aws:robomaker:us-west-2:852352399745:simulation-job/sim-qly9vz6dtrcd

Last updated
Thu August 8, 2019 12:01:19 PM

4. Create robot application

AWS RoboMaker maintains ROS applications as two types. **“Robot Application”** and **“Simulation Application”**. “Robot Application” is an application which can launch both simulation and real robot. On the other hand, ROS application maintained as **“Simulation Application”** can only be launched on simulation environment.

	In Simulation Job	Deploy to robot	Main purpose
Simulation Application	Mandatory	N/A	- Launch simulation environment - Launch simulation robot
Robot Application	Optional (Simulation can be launched without robot application)	Yes	- Control robot - ROS node which interact with HW

We’ve been only worked with **Simulation application** so far. Let’s look at **Robot application**. Reorganized the Doosan ROS package source to make it more suitable for AWS RoboMaker integration. While this restructuring, it’s already constructed by with Robot application and Simulation application.

Following is the reorganized default file structure:

```
doosan-robot
+ robot_ws --- Robot application
  + src -- source directory for robot app. ROS packages related to the application stays
+ simulation_ws --- Robot application
  + src -- source directory for simulation app. ROS packages related to the application stays
roboMakerSettings.json --- setting file for this project.
```

Doosan ROS packages require minor modifications to be used in AWS RoboMaker. This modified part is annotated **"for aws robomaker"**, You can find the modified part by searching for **"for aws robomaker"**. To understand why this change is necessary, the following link may be helpful.

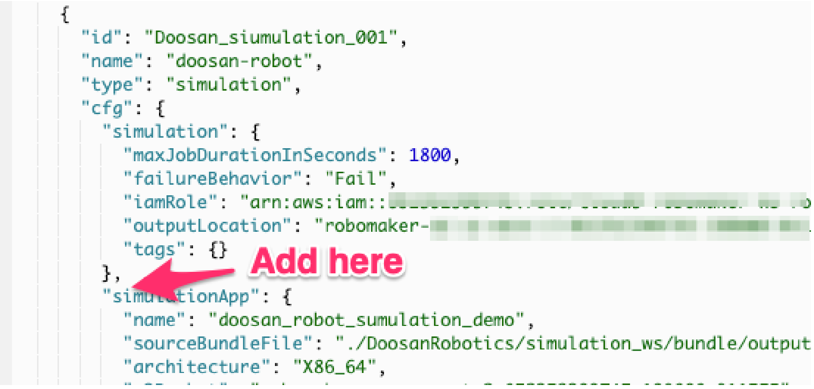
https://aws.amazon.com/jp/blogs/opensource/building-bundling-ros-app-aws-robomaker/)

Package	SIMULATION	ROBOT	Note
common	✔	✔	
doosan_robot	✔	✔	
dsr_control	✔	✔	
dsr_description	✔	✔	
dsr_example	✔	✔	
dsr_gazebo	✔		
dsr_launcher	✔	✔	
dsr_msgs	✔	✔	
moveit_config_m0609	✔	✔	
moveit_config_m0617	✔	✔	
moveit_config_m1013	✔	✔V	
moveit_config_m1509	✔	✔	

Simulation job has already configured but it's not included in the Robot Application, let's include it in Robot Application now.

- **[4.1]** Open **roboMakerSettings.json** file. Edit the file to include robot application to a simulation job.

  - To do so, add item below between “simulation” and “simulationApp”



```
{
  "id": "Doosan_siumulation_001",
  "name": "doosan-robot",
  "type": "simulation",
  "cfg": {
    "simulation": {
      "maxJobDurationInSeconds": 1800,
      "failureBehavior": "Fail",
      "iamRole": "arn:aws:iam::[redacted]
      "outputLocation": "robomaker-[redacted]
      "tags": {}
    },
    "simulationApp": {
      "name": "doosan_robot_sumulation_demo",
      "sourceBundleFile": "./DoosanRobotics/simulation_ws/bundle/output
      "architecture": "X86_64",
```

Add here

  - Following is the item to add.

```
"robotApp": {
   "name": "<robot app name>",
   "s3Bucket": "<bucket name>",
   "sourceBundleFile": "./doosan-robot/robot_ws/bundle/output.tar",
   "architecture": "X86_64",
   "robotSoftwareSuite": {
      "version": "Kinetic",
      "name": "ROS"
   },
   "launchConfig": {
      "packageName": "dsr_launcher",
      "launchFile": "aws_bringup_robot.launch"
   }
},
```

After adding the items, please modify following part:

-For **<robot app name>**, replace it to **robot_app_name** in **doosan-robot/ws_settings.yaml**

-For **<bucket name>**, replace it to **bucket_name** in **doosan-robot/ws_settings.yaml**

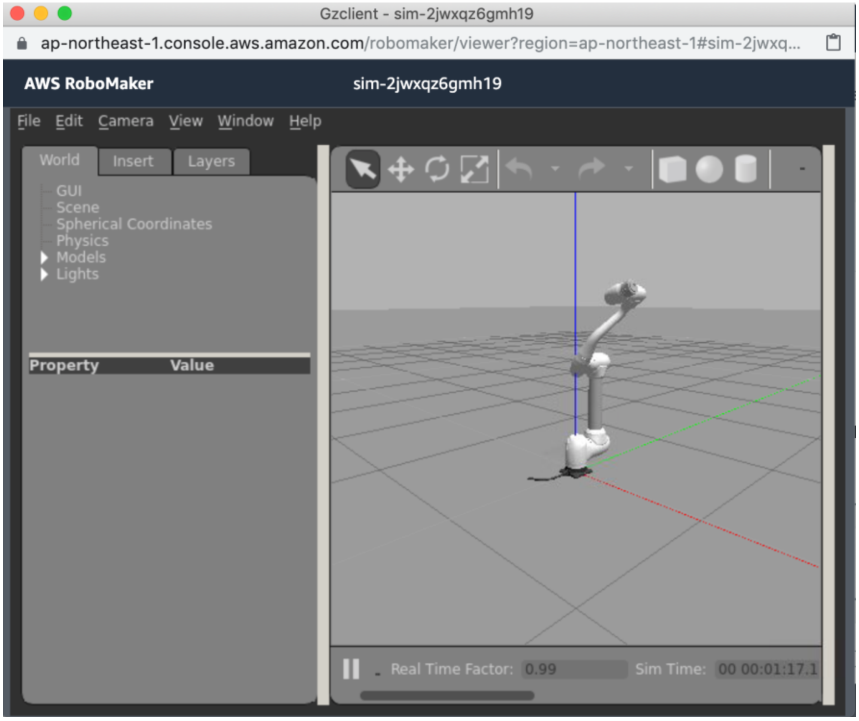
-Press **[Ctrl] + s** to save changes.

(Note: ws_settings.yaml file records the resources created by setup.sh setup script)

- **[4.2]** From menu select ***Run -> Build -> doosan robot robot-app*** to build robot application and ***Run -> Bundle -> doosan robot robot-app*** to bundle the robot application.

- **[4.3]** Launch simulation job again by selecting ***Run -> Launch Simulation -> doosan-robot*** from the menu.

- **[4.4]** This time, both robot application and simulation application are created and executed on simulator. Robot application launches **single_robot_simple.py** (through **aws_bringup_robot.launch** in **dsr_launcher** package), so it when simulation launched, the robot starts to move.


Gzclient - sim-2jwxqz6gmh19
ap-northeast-1.console.aws.amazon.com/robomaker/viewer?region=ap-northeast-1#sim-2jwxq...
AWS RoboMaker
sim-2jwxqz6gmh19
File Edit Camera View Window Help
World Insert Layers
GUI
Scene
Spherical Coordinates
Physics
Models
Lights
Property Value

Real Time Factor: 0.99 Sim Time: 00 00:01:17.1

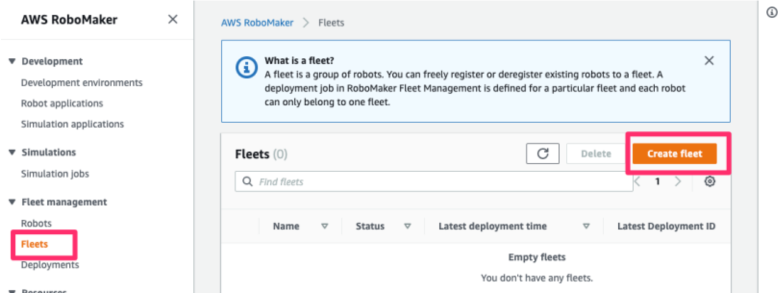
5. Create fleet

Fleet Management is the feature to manage robots. By using fleet management, you can install robot applications to robots remotely.

To understand how to use fleet management, we first create **fleet**, then register a **robot** and add the robot to the **fleet**. A **fleet** is kind of a group to maintain the robot application. You deploy robot application to a **fleet** and all robots belong to the **fleet** download automatically the application and install in to itself.

- **[5.1]** Open AWS RoboMaker (**https://console.aws.amazon.com/robomaker**)

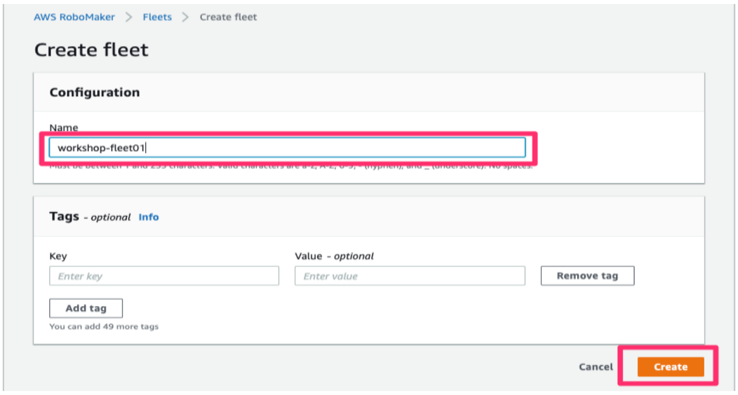
- **[5.2]** From navigation pain on left, select Fleets in Fleet Management. Fleets list will be displayed. Select **[Create fleet]** button on top right.



What is a fleet?
A fleet is a group of robots. You can freely register or deregister existing robots to a fleet. A deployment job in RoboMaker Fleet Management is defined for a particular fleet and each robot can only belong to one fleet.

Empty fleets
You don't have any fleets.

- **[5.3]** Create fleet window will be displayed. Put fleet name in the name field and click [Create] button.



Name
workshop-fleet01

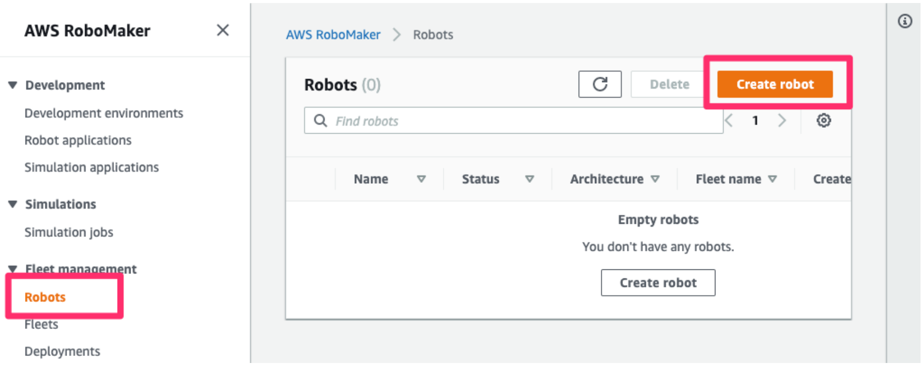
Must be between 1 and 255 characters. Valid characters are a-z, A-Z, 0-9, - (hyphen), and _ (underscore). No spaces.

You can add 49 more tags

6. Register a robot to AWS RoboMaker

Next, let's register a robot to fleet management. AWS RoboMaker then be able to have control to the robot.

- **[6.1]** From navigation pain of left, select **Robots** in **Fleet management**. Robot list will be displayed. Select **[Create robot]** button on top right.



Empty robots

You don't have any robots.

- **[6.2]** Create robot window will be displayed. Set input fields accordingly.

  - **Name**: name of the robot. Some arbitrary name you easy to recognize which one is which.

  - **Architecture**: CPU architecture which the robot application will run. Choose X86_64 this time.

  - **AWS Greengrass group**: Choose **[Create new]**

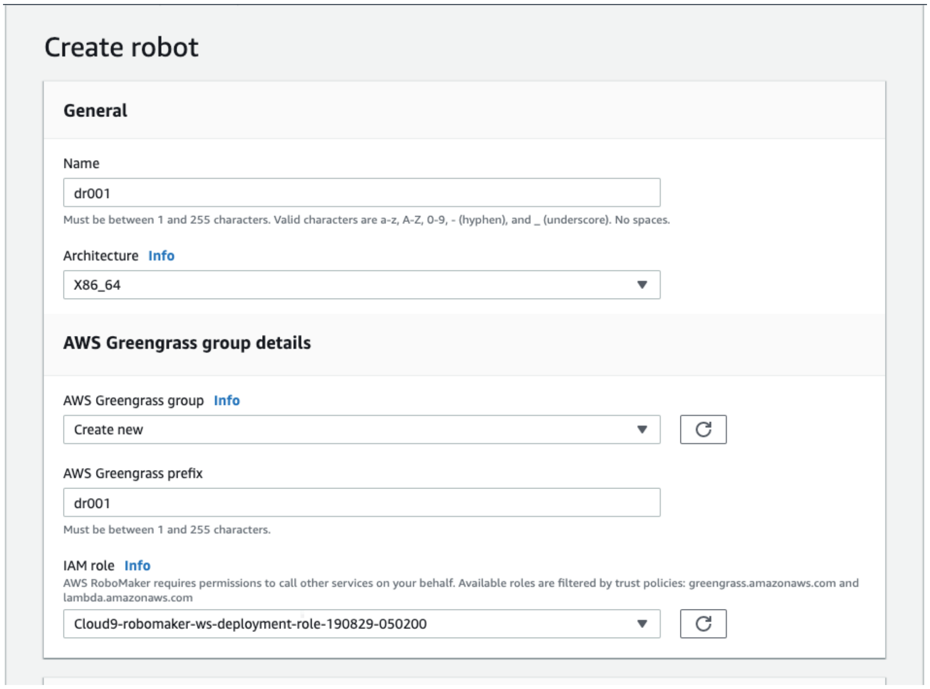
  - **AWS Greengrass prefix**: Name of the robot will automatically be applied. Just accept it this time.

  - **IAM Role**: Find value of **iam_role_for_deployment in doosan-robot/ws_settings.yaml file**.

  The setting script created a role for you.

  (This IAM role is used to define the access permission of AWS resources from the robot. If you want to create a role manually, please refer:

  https://docs.aws.amazon.com/robomaker/latest/dg/create-robot.html#create-robot-role)



Create robot

General

Name

dr001

Must be between 1 and 255 characters. Valid characters are a-z, A-Z, 0-9, - (hyphen), and _ (underscore). No spaces.

Architecture Info

X86_64

AWS Greengrass group details

AWS Greengrass group Info

Create new

AWS Greengrass prefix

dr001

Must be between 1 and 255 characters.

IAM role Info

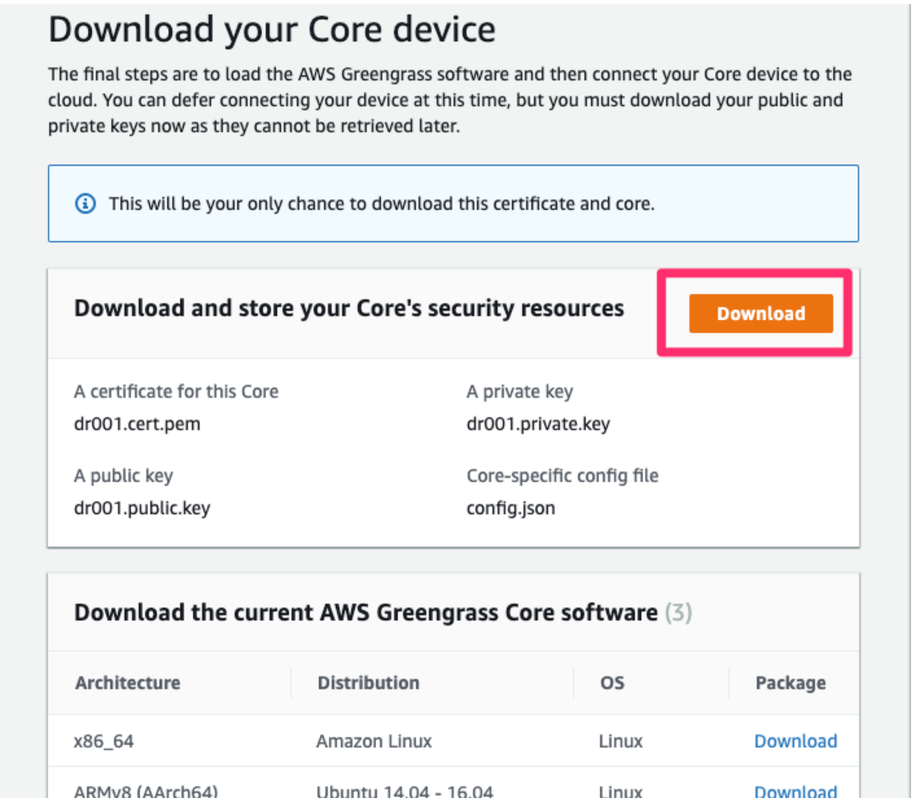
AWS RoboMaker requires permissions to call other services on your behalf. Available roles are filtered by trust policies: greengrass.amazonaws.com and lambda.amazonaws.com

Cloud9-robomaker-ws-deployment-role-190829-050200

- **[6.3]** Click **[Create]** button on bottom right, **"Download your Core device"** window will be displayed.

- **[6.4]** From **"Download your Core device"**, you download files need to be placed to the ROS Master PC to control the robot. First, click **[Download]** button next to **Download and store your Core's security resources**. This download security key files and the settings. Note that you only can download the private key from here and you won't be able to come back to this page afterword. So, please be sure that you download the file right and store it in safe location. This saved file will be used in **Chapter 7. Setup the ROS Master PC.**

- **[6.5]** From **Download the current AWS Greengrass Core software**, you can download AWS IoT Greengrass software. You have to install it into your ROS Master PC to control the robot. The instruction how to setup the **Greengrass** into device is described **chapter 7. Setup the ROS Master PC**.



Download your Core device

The final steps are to load the AWS Greengrass software and then connect your Core device to the cloud. You can defer connecting your device at this time, but you must download your public and private keys now as they cannot be retrieved later.

This will be your only chance to download this certificate and core.

Download and store your Core's security resources

A certificate for this Core
dr001.cert.pem

A private key
dr001.private.key

A public key
dr001.public.key

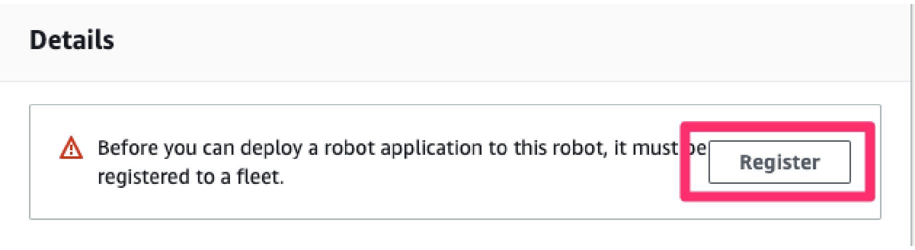
Core-specific config file
config.json

Download the current AWS Greengrass Core software (3)

Architecture	Distribution	OS	Package
x86_64	Amazon Linux	Linux	Download
ARMv8 (AArch64)	Ubuntu 14.04 - 16.04	Linux	Download

- Press **[View robot]** button on bottom right.

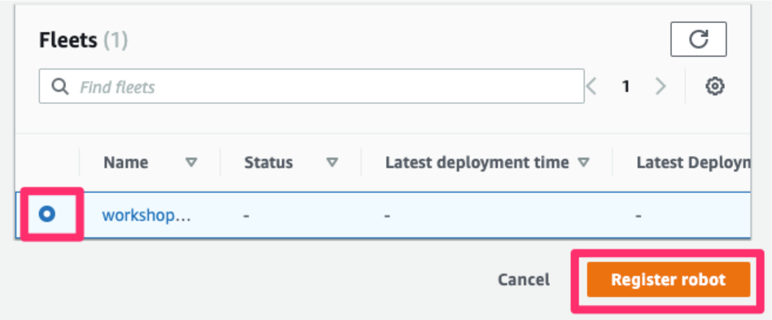
- **"Details"** page will be displayed. You will find **[Register]** button on top left. This assist you to register the robot to a fleet. Let's click the button and register the robot to the **fleet** we just created.



Details

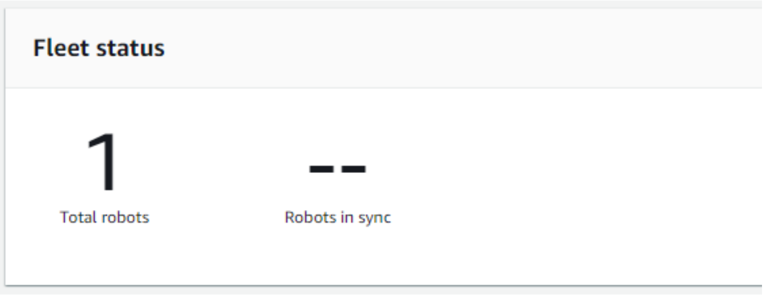
Before you can deploy a robot application to this robot, it must be registered to a fleet.

- **"Select fleet to register robot to"** window will be displayed. Select the radio button next to the fleet you just created and click **[Register robot]** button.



	Name	Status	Latest deployment time	Latest Deployn
●	workshop...	-	-	-

- The robot is now belonging to the fleet you just created. Click the link under the Fleet name of Details page, it will navigate you to the fleet page. Fleet page now show number of total robots as 1.



Fleet status

1
Total robots

--
Robots in sync

Now, setup for AWS RoboMaker side for deployment is ready. Let's then setup the robot (DRCF server)

7. Setup the ROS Master PC

Let's set up a ROS Master PC to control the robot so we can connect to AWS RoboMaker. After completing these steps, your ROS Master PC connects to AWS RoboMaker. Greengrass must be installed on the ROS Master PC. Please refer to the link below for details.

https://docs.aws.amazon.com/greengrass/latest/developerguide/setup-filter.other.html

The following should be done on the ROS Master PC.

- **[7.1]** Setup to install AWS IoT Greengrass

  - Execute following to create user named ggc_user and group named ggc_group

```
sudo adduser --system ggc_user
sudo addgroup --system ggc_group
```

  - Execute Following Check if the Linux PC meets the software requirement for installing AWS IoT Greengrass.

```
cd /
mkdir greengrass-dependency-checker-GGCv1.9.x
cd greengrass-dependency-checker-GGCv1.9.x
wget https://github.com/aws-samples/aws-greengrass-samples/raw/master/greengrass-dependency-checker-GGCv1.9.x.zip
unzip greengrass-dependency-checker-GGCv1.9.x.zip
cd greengrass-dependency-checker-GGCv1.9.x
sudo ./check_ggc_dependencies | more
```

  - If you see any error, install the required software to correct the error.

  Install the software required before installing Greengrass.

  - As of Ubuntu 16.04 you will need python upgrade, nodejs, and java8 installation.

  After installing the program, be sure to symbolically link the executables to the names that Greengrass requires.

- **[7.2]** Download greengrass-linux-x86-64-1.9.2.tar.gz

  https://d1onfpft10uf5o.cloudfront.net/greengrass-core/downloads/1.9.2/greengrass-linux-x86-64-1.9.2.tar.gz

  - and extract the file to root

```
sudo tar -xzvf greengrass-linux-x86-64-1.9.2.tar.gz -C /
```

- **[7.3]** Execute following to place root CA.

```
cd /greengrass/certs/

sudo wget -O root.ca.pem
ttps://www.amazontrust.com/repository/AmazonRootCA1.pem
```

- **[7.4]** Copy the security resource file (**<robot name>-setup.zip**) downloaded from **Chapter 6. Register a robot to AWS RoboMaker** to your ROS master PC.

- **[7.5]** On the ROS Master PC, execute following command to extract the file. You may be asked if you should overwrite existing files with new one. Answer "A" to overwrite all.

```
sudo unzip <robotname>-setup.zip -d /greengrass
```

  (Please replace <robotname>-setup.zip to the actual file name you've copied.)

- **[7.6]** Execute following command on the robot (ROS Master PC). When succeeded, message "Greengrass successfully started" will be shown.

```
sudo /greengrass/ggc/core/greengrassd start
```

Now robot is ready. Let's deploy.

8. Build & Bundle robot app

If you don't make any change to the source code after section 4 completed, you don't need to do this section. If you make any change to source code and want to deploy it before testing it on simulation, you need step here.

- **[8.1]** Do build and bundle.

  - From development environment menu, select ***Run -> Build -> doosan robot robot-app*** and ***Run -> Bundle -> doosan robot robot-app***. This will compile and bundle the robot application.

- **[8.2]** Upload latest bundle to S3.

  - This operation can be done automatically or manually. Recommend how to do it automatically.

  - [How to do it automatically]

  Run the simulation ***Run-> Launch Simulation-> doosan-robot*** and the bundle will be uploaded to S3 automatically.

  - [How to do it manually]

  In a terminal in the AWS RoboMaker development environment, run the following command:

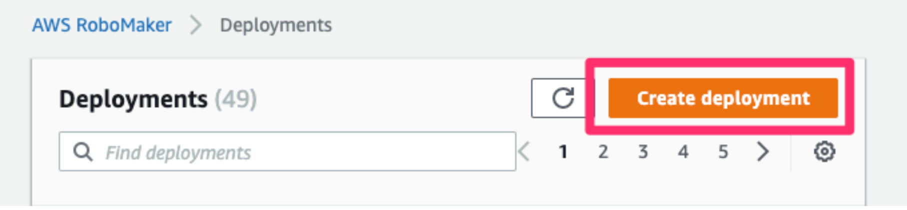
```
bucket=<buckt_name>

aws s3 cp ~/environment/doosan-robot/robot_ws/bundle/output.tar
s3://${bucket}/robot_ws/bundle/output.tar
```

  Please replace ***<bucket_name>*** to the bucket_name in doosan-robot/ws_settings.yaml

9. Deploy to a robot

- **[9.1]** Open AWS RoboMaker from AWS Management Console. From navigation pain on left, select **Deployments** in **Fleet management**. Click [Create deployment] button on top right.



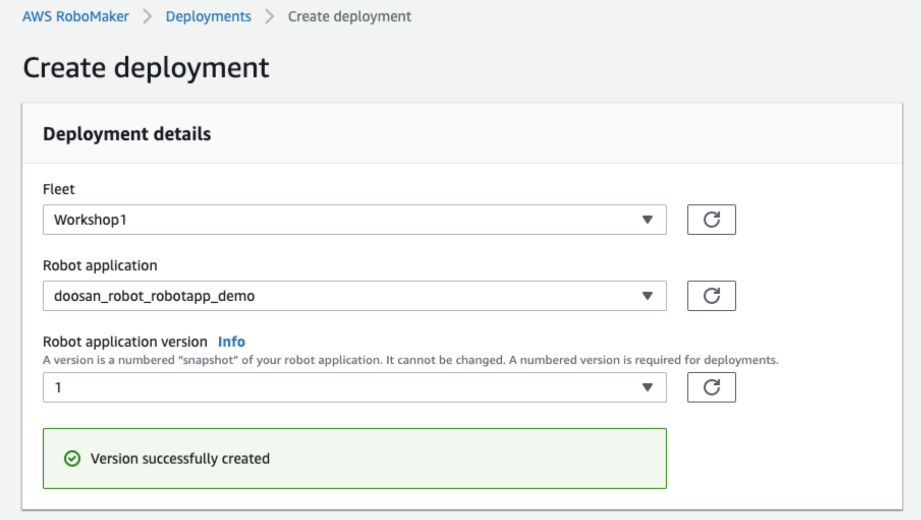
- **[9.2]** Create deployment window will be displayed.

  - **Fleet:** Choose the fleet you just created.

  - **Robot application:** Choose the application you created. The name is auto generated and registered when you launched a simulation.

  You can find the robot application name from the value of **robot_app_name** in **doosan-robot/ws_settings.yaml file.**

  - **Robot application version:** Choose Create new to create a new version.



Create deployment

Deployment details

Fleet: Workshop1

Robot application: doosan_robot_robotapp_demo

Robot application version Info
A version is a numbered "snapshot" of your robot application. It cannot be changed. A numbered version is required for deployments.
1

Version successfully created

- **[9.3]** In Deployment launch configuration, set as follow:

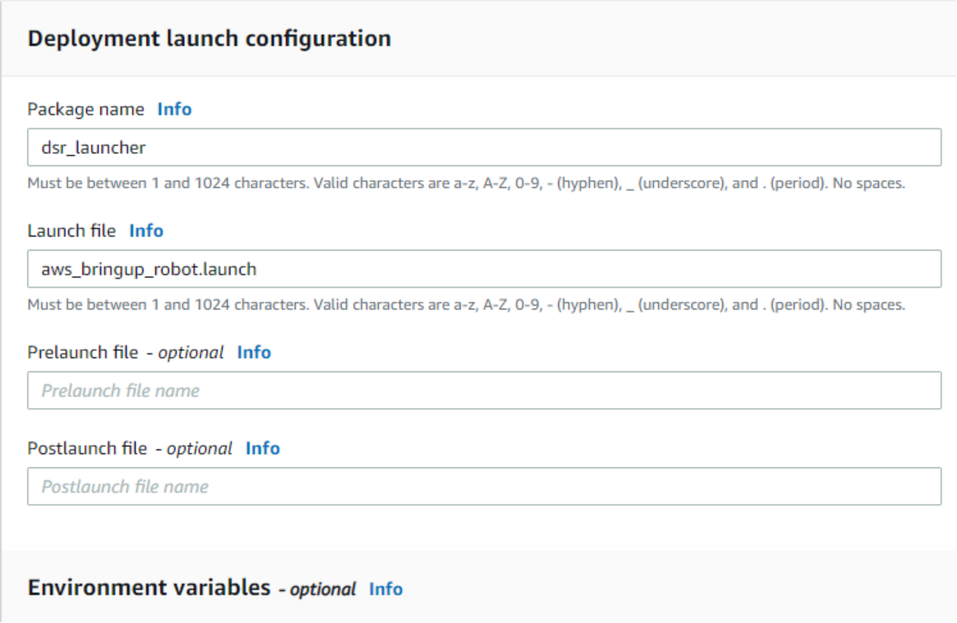
  - Package name: dsr_launcher

  - **Launch file:** aws_bringup_robot.launch

  - **Prelaunch file:** optional. By setting here, you can run a script file before ROS application is coming up.

  - **Postlaunch file:** optional. By setting here, you can run a script just after ROS application is coming up.

  - **Environment variables:** Optional. You can define environment variables here. The environment variables are given to the launched ROS application.



Deployment launch configuration

Package name Info

dsr_launcher

Must be between 1 and 1024 characters. Valid characters are a-z, A-Z, 0-9, - (hyphen), _ (underscore), and . (period). No spaces.

Launch file Info

aws_bringup_robot.launch

Must be between 1 and 1024 characters. Valid characters are a-z, A-Z, 0-9, - (hyphen), _ (underscore), and . (period). No spaces.

Prelaunch file - *optional* Info

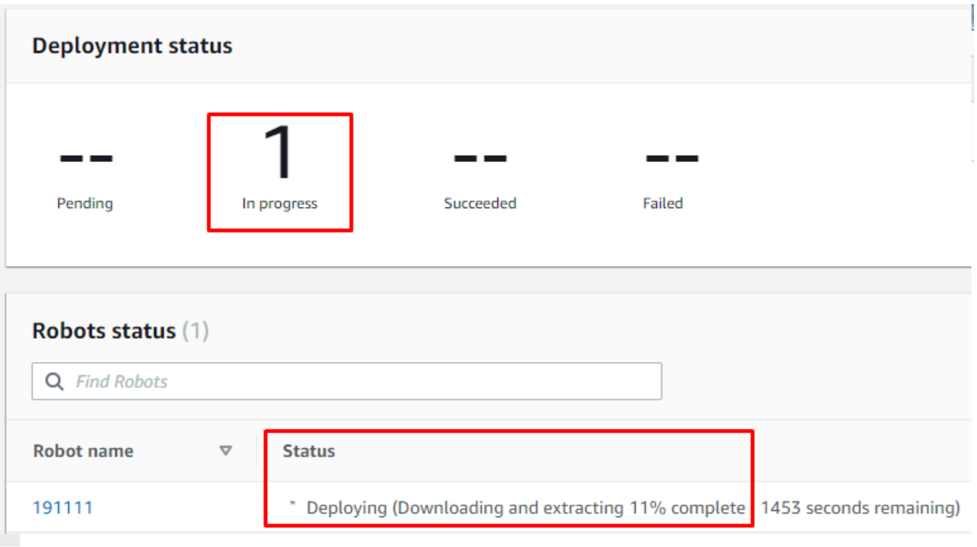
Prelaunch file name

Postlaunch file - *optional* Info

Postlaunch file name

Environment variables - *optional* Info

- **[9.4]** Click **[Create]** button on bottom right. Deployment is now starting.



Deployment status

--	1	--	--
Pending	In progress	Succeeded	Failed

Robots status (1)

Find Robots

Robot name	Status
191111	Deploying (Downloading and extracting 11% complete 1453 seconds remaining)

- When everything goes right, the count of Succeeded will be 1 and the robot will start moving.

- **Note:**

  - First deployment would take time. It downloads all bundle from the beginning. If you made small change and deploy the change, the deployment time will be shorten. This is because the fleet management would only download the changed part.

  - If you want to deploy again with the same settings, you can choose **[Action] -> [Close]** from existing deploy, you will then not need to setup the deployment configuration part once again.

10. References

- **Links:**

  - **Working with Robot Applications**

  https://docs.aws.amazon.com/robomaker/latest/dg/managing-robot-applications.html

  - Working with Simulation Applications

  https://docs.aws.amazon.com/robomaker/latest/dg/managing-simulation-applications.html

  - Fleet management

  https://docs.aws.amazon.com/robomaker/latest/dg/fleets.html

Doosan Robotics

www.doosanrobotics.com