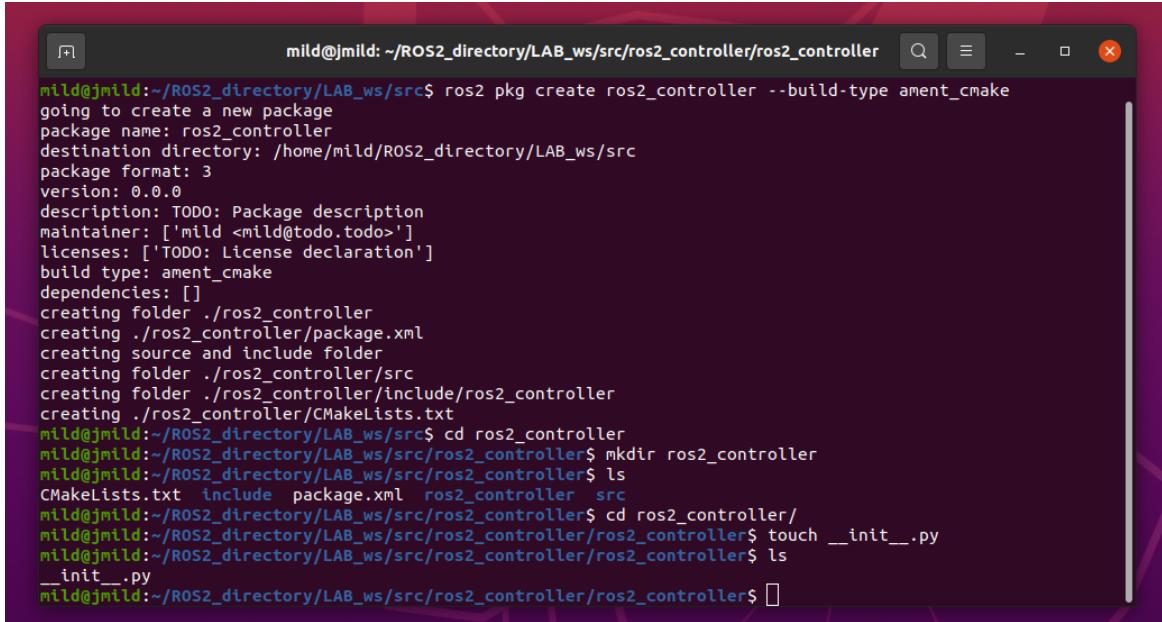


Lab3 Report_08

Create package: ros2_controller

Create folder: ros2_controller

Create file: __init__.py (in folder: ros2_controller)



```
mild@mild:~/ROS2_directory/LAB_ws/src$ ros2 pkg create ros2_controller --build-type ament_cmake
going to create a new package
package name: ros2_controller
destination directory: /home/mild/ROS2_directory/LAB_ws/src
package format: 3
version: 0.0.0
description: TODO: Package description
maintainer: ['mild <mild@todo.todo>']
licenses: ['TODO: License declaration']
build type: ament_cmake
dependencies: []
creating folder ./ros2_controller
creating ./ros2_controller/package.xml
creating source and include folder
creating folder ./ros2_controller/src
creating folder ./ros2_controller/include/ros2_controller
creating ./ros2_controller/CMakeLists.txt
mild@mild:~/ROS2_directory/LAB_ws/src$ cd ros2_controller
mild@mild:~/ROS2_directory/LAB_ws/src/ros2_controller$ mkdir ros2_controller
mild@mild:~/ROS2_directory/LAB_ws/src/ros2_controller$ ls
CMakeLists.txt include package.xml ros2_controller src
mild@mild:~/ROS2_directory/LAB_ws/src/ros2_controller$ cd ros2_controller/
mild@mild:~/ROS2_directory/LAB_ws/src/ros2_controller/ros2_controller$ touch __init__.py
mild@mild:~/ROS2_directory/LAB_ws/src/ros2_controller/ros2_controller$ ls
__init__.py
mild@mild:~/ROS2_directory/LAB_ws/src/ros2_controller/ros2_controller$ 
```

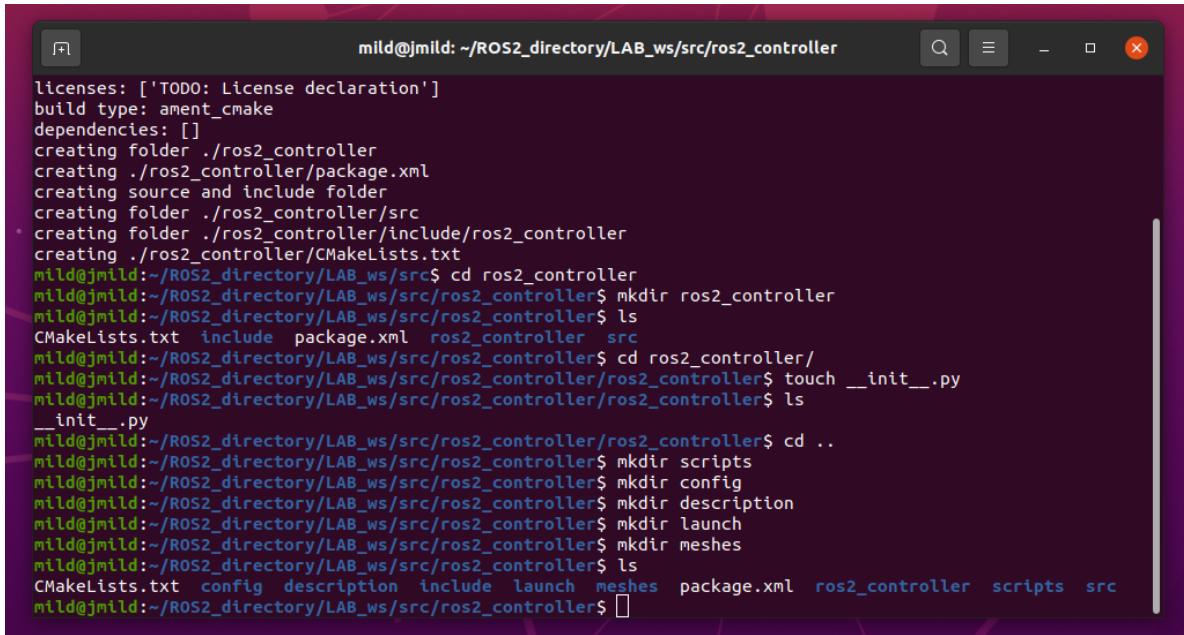
Create folder: scripts

Create folder: config

Create folder: description

Create folder: launch

Create folder: meshes

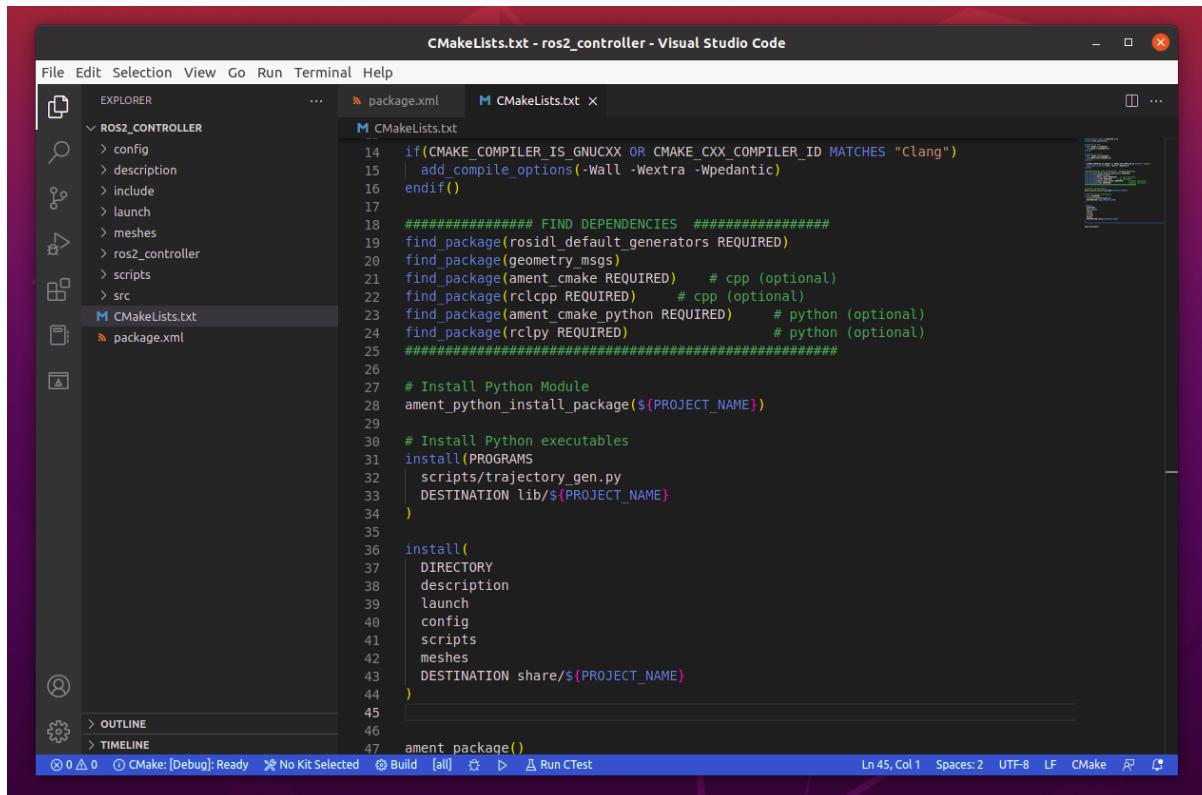


```
licenses: ['TODO: License declaration']
build type: ament_cmake
dependencies: []
creating folder ./ros2_controller
creating ./ros2_controller/package.xml
creating source and include folder
creating folder ./ros2_controller/src
creating folder ./ros2_controller/include/ros2_controller
creating ./ros2_controller/CMakeLists.txt
mild@mild:~/ROS2_directory/LAB_ws/src$ cd ros2_controller
mild@mild:~/ROS2_directory/LAB_ws/src/ros2_controller$ mkdir ros2_controller
mild@mild:~/ROS2_directory/LAB_ws/src/ros2_controller$ ls
CMakeLists.txt include package.xml ros2_controller src
mild@mild:~/ROS2_directory/LAB_ws/src/ros2_controller$ cd ros2_controller/
mild@mild:~/ROS2_directory/LAB_ws/src/ros2_controller/ros2_controller$ touch __init__.py
mild@mild:~/ROS2_directory/LAB_ws/src/ros2_controller/ros2_controller$ ls
__init__.py
mild@mild:~/ROS2_directory/LAB_ws/src/ros2_controller/ros2_controller$ cd ..
mild@mild:~/ROS2_directory/LAB_ws/src/ros2_controller$ mkdir scripts
mild@mild:~/ROS2_directory/LAB_ws/src/ros2_controller$ mkdir config
mild@mild:~/ROS2_directory/LAB_ws/src/ros2_controller$ mkdir description
mild@mild:~/ROS2_directory/LAB_ws/src/ros2_controller$ mkdir launch
mild@mild:~/ROS2_directory/LAB_ws/src/ros2_controller$ mkdir meshes
mild@mild:~/ROS2_directory/LAB_ws/src/ros2_controller$ ls
CMakeLists.txt config description include launch meshes package.xml ros2_controller scripts src
mild@mild:~/ROS2_directory/LAB_ws/src/ros2_controller$ 
```

Create file: trajectory_gen.py (in folder: scripts) and adds the execute permission (chmod +x)

```
mild@mild: ~/ROS2_directory/LAB_ws/src/ros2_controller/scripts
creating folder ./ros2_controller
creating ./ros2_controller/package.xml
creating source and include folder
creating folder ./ros2_controller/src
creating folder ./ros2_controller/include/ros2_controller
creating ./ros2_controller/CMakeLists.txt
mild@mild:~/ROS2_directory/LAB_ws/src$ cd ros2_controller
mild@mild:~/ROS2_directory/LAB_ws/src/ros2_controller$ mkdir ros2_controller
mild@mild:~/ROS2_directory/LAB_ws/src/ros2_controller$ ls
CMakeLists.txt  include  package.xml  ros2_controller  src
mild@mild:~/ROS2_directory/LAB_ws/src/ros2_controller$ cd ros2_controller/
mild@mild:~/ROS2_directory/LAB_ws/src/ros2_controller/ros2_controller$ touch __init__.py
mild@mild:~/ROS2_directory/LAB_ws/src/ros2_controller/ros2_controller$ ls
__init__.py
mild@mild:~/ROS2_directory/LAB_ws/src/ros2_controller/ros2_controller$ cd ..
mild@mild:~/ROS2_directory/LAB_ws/src/ros2_controller$ mkdir scripts
mild@mild:~/ROS2_directory/LAB_ws/src/ros2_controller$ mkdir config
mild@mild:~/ROS2_directory/LAB_ws/src/ros2_controller$ mkdir description
mild@mild:~/ROS2_directory/LAB_ws/src/ros2_controller$ mkdir launch
mild@mild:~/ROS2_directory/LAB_ws/src/ros2_controller$ mkdir meshes
mild@mild:~/ROS2_directory/LAB_ws/src/ros2_controller$ ls
CMakeLists.txt  config  description  include  launch  meshes  package.xml  ros2_controller  scripts  src
mild@mild:~/ROS2_directory/LAB_ws/src/ros2_controller$ cd scripts/
mild@mild:~/ROS2_directory/LAB_ws/src/ros2_controller/scripts$ touch trajectory_gen.py
mild@mild:~/ROS2_directory/LAB_ws/src/ros2_controller/scripts$ chmod +x trajectory_gen.py
mild@mild:~/ROS2_directory/LAB_ws/src/ros2_controller/scripts$ 
```

Configure CMakeLists.txt



```
CMakeLists.txt - ros2_controller - Visual Studio Code
File Edit Selection View Go Run Terminal Help
EXPLORER          ...  package.xml  CMakeLists.txt ...
M CMakeLists.txt
M CMakeLists.txt
14 if(CMAKE_COMPILER_IS_GNUCXX OR CMAKE_CXX_COMPILER_ID MATCHES "Clang")
15 | add_compile_options(-Wall -Wextra -Wpedantic)
16 endif()
17 #####
18 ##### FIND DEPENDENCIES #####
19 find_package(rosidl_default_generators REQUIRED)
20 find_package(geometry_msgs)
21 find_package(ament_cmake REQUIRED)      # cpp (optional)
22 find_package(rclcpp REQUIRED)          # cpp (optional)
23 find_package(ament_cmake_python REQUIRED)    # python (optional)
24 find_package(rclpy REQUIRED)           # python (optional)
25 #####
26 # Install Python Module
27 ament_python_install_package(${PROJECT_NAME})
28
29 # Install Python executables
30 install(PROGRAMS
31   scripts/trajectory_gen.py
32   DESTINATION lib/${PROJECT_NAME}
33 )
34
35 install(
36   DIRECTORY
37   description
38   launch
39   config
40   scripts
41   meshes
42   DESTINATION share/${PROJECT_NAME}
43
44 )
45
46 ament_package()
```

Configure package.xml

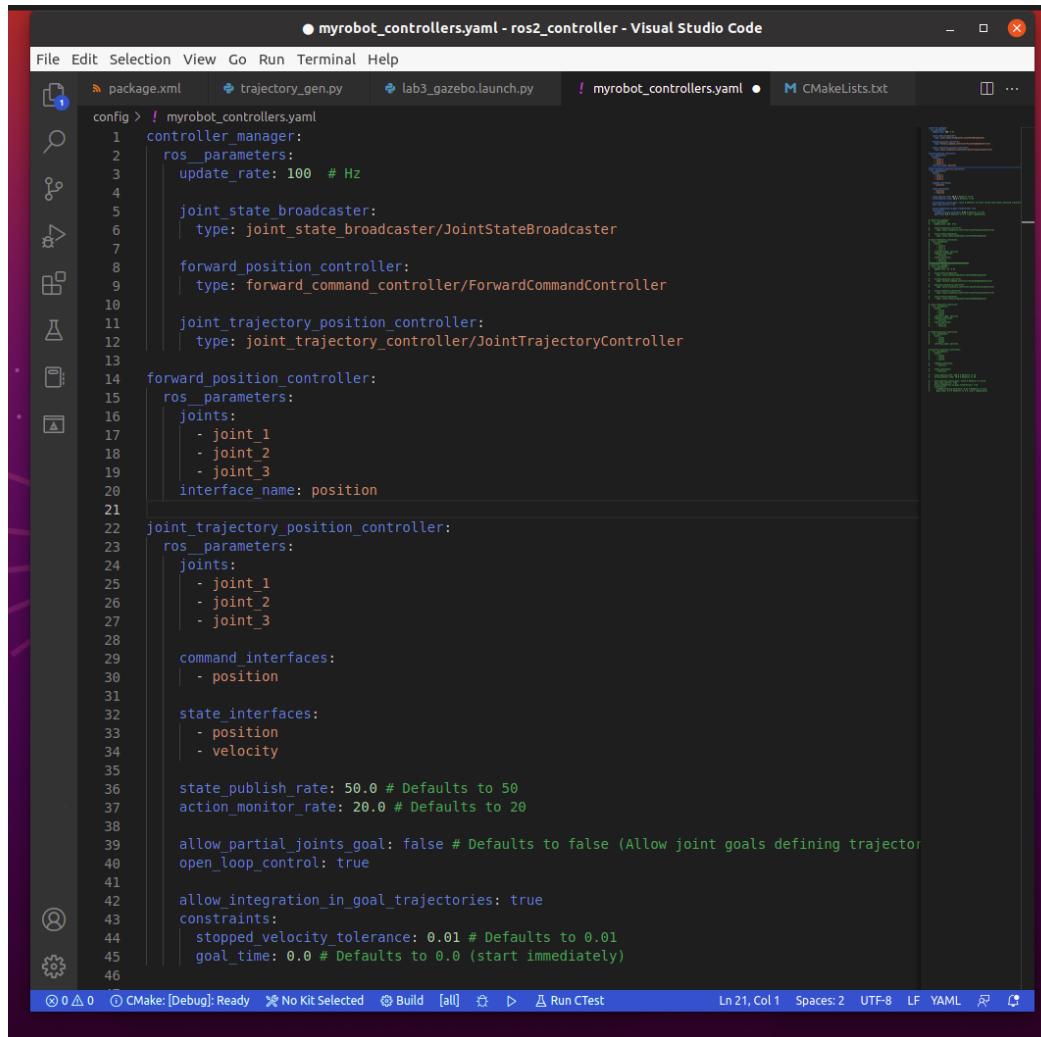
```
mild@jmild: ~/ROS2_directory/LAB_ws$ colcon build --packages-select ros2_controller
Starting >> ros2_controller
Finished <<< ros2_controller [0.27s]

Summary: 1 package finished [0.56s]
mild@jmild: ~/ROS2_directory/LAB_ws$
```

```
<?xml version="1.0"?>
<?xml-model href="http://download.ros.org/schema/package_format3.xsd" schematypens="http://www.w3.org/2001/XMLSchema#"/>
<package format="3">
  <name>ros2_controller</name>
  <version>0.0.0</version>
  <description>TODO: Package description</description>
  <maintainer email="mild@todo.todo">mild</maintainer>
  <license>TODO: License declaration</license>
  <buildtool_depend name="ament_cmake"/>
  <buildtool_depend name="ament_cmake_python"/>
  <buildtool_depend name="rosidl_default_generators"/>
  <depend name="rclcpp"/>
  <depend name="rclpy"/>
  <test_depend name="ament_lint_auto"/>
  <test_depend name="ament_lint_common"/>
  <export>
    <build_type>ament_cmake</build_type>
  </export>
</package>
```

colcon build package: ros2_controller

Writing ROS Joint Trajectory Controller YAML File for Robotic Arm to Control its Joints.



The screenshot shows a Visual Studio Code window with a dark theme. The title bar reads "● myrobot_controllers.yaml - ros2_controller - Visual Studio Code". The editor pane displays a YAML configuration file for a ROS joint trajectory controller. The file defines a controller manager with various controllers and parameters. Key sections include:

- controller_manager**:
 - ros_parameters**:
 - update_rate**: 100 # Hz
 - joint_state_broadcaster**:
 - type**: joint_state_broadcaster/JointStateBroadcaster
 - forward_position_controller**:
 - type**: forward_command_controller/ForwardCommandController
 - joint_trajectory_position_controller**:
 - type**: joint_trajectory_controller/JointTrajectoryController
- forward_position_controller**:
 - ros_parameters**:
 - joints**:
 - joint_1
 - joint_2
 - joint_3
 - interface_name**: position
- joint_trajectory_position_controller**:
 - ros_parameters**:
 - joints**:
 - joint_1
 - joint_2
 - joint_3
 - command_interfaces**:
 - position
 - state_interfaces**:
 - position
 - velocity
 - state_publish_rate**: 50.0 # Defaults to 50
 - action_monitor_rate**: 20.0 # Defaults to 20
 - allow_partial_joints_goal**: false # Defaults to false (Allow joint goals defining trajectory)
 - open_loop_control**: true
 - allow_integration_in_goal_trajectories**: true
 - constraints**:
 - stopped_velocity_tolerance**: 0.01 # Defaults to 0.01
 - goal_time**: 0.0 # Defaults to 0.0 (start immediately)

Connect IMU with Library XICRO

Git: <https://github.com/imchin/Xicro>

-Create Workspace for Install Xicro Package

1. Create Workspace: xicro_ws
2. Make Directory: mkdir Xicro
3. Clone Git: sudo apt-get install git and git clone <https://github.com/imchin/Xicro>

```
bash: /home/onnalin/IMU_tools/install/setup.bash: No such file or directory
$ sudo apt-get install git
[sudo] password for onnalin:
Reading package lists... Done
Building dependency tree
Reading state information... Done
git is already the newest version (1:2.25.1-1ubuntu3.6).
The following packages were automatically installed and are no longer required:
  chromium-codecs-ffmpeg-extra gstreamer1.0-vaapi libfwupdplugin1
  libgstreamer-plugins-bad1.0-0 libvba-wayland2 libxml2
Use 'sudo apt autoremove' to remove them.
0 upgraded, 0 newly installed, 0 to remove and 12 not upgraded.
$ git clone https://github.com/imchin/Xicro
Cloning into 'Xicro'...
remote: Enumerating objects: 300, done.
remote: Counting objects: 100% (300/300), done.
remote: Compressing objects: 100% (181/181), done.
remote: Total 300 (delta 156), reused 189 (delta 77), pack-reused 0
Receiving objects: 100% (300/300), 102.74 KiB | 251.00 KiB/s, done.
Resolving deltas: 100% (156/156), done.
$
```

4. colcon build your workspace at: cd ~/.xicro_ws/src

-Setup parameter in yaml

First step before Setup

pip3 uninstall serial
pip3 install pyserial
pip3 install numpy

1. Go to config: cd ~/.xicro_ws/src/Xicro/xicro_pkg/config
2. Open code .yaml: code setup_xicro.yaml

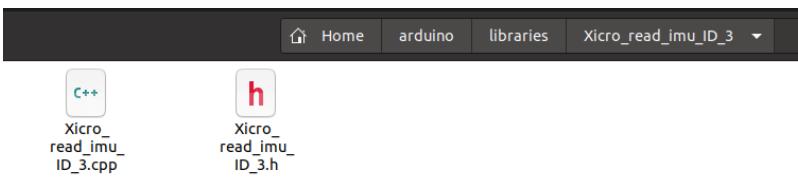
```
! setup_xicro.yaml M ●
home > onnalin > xicro_ws > src > Xicro > xicro_pkg > config > ! setup_xicro.yaml
1 Idmcu: 3
2 Namespace: "read_imu"
3 Port: "/dev/ttyACM0"
4 generate_library_Path: "arduino/libraries"
5 Baudrate: 115200
6 Setup_Publisher: [      [1,"Imu_arduino","sensor_msgs/Imu.msg"]      ]
7 Setup_Subscriber: [      ]
8 Setup_Srv_client: [      ]
9 |
```

Before Create xicro library ,Check your arduino IDE (This lab we use arduino Version 18)

- Create xicro library

1. colcon build your workspace
2. Generate library: ros2 run xicro_pkg generate_library.py arduino

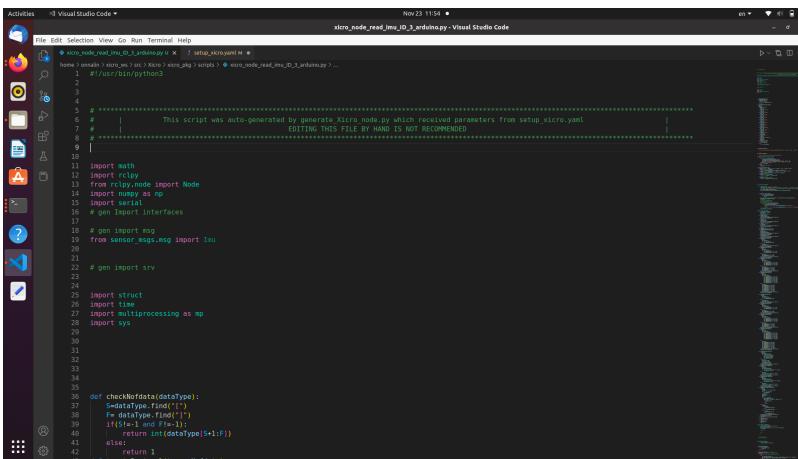
Check file generate library



-Generate Node of Xicro

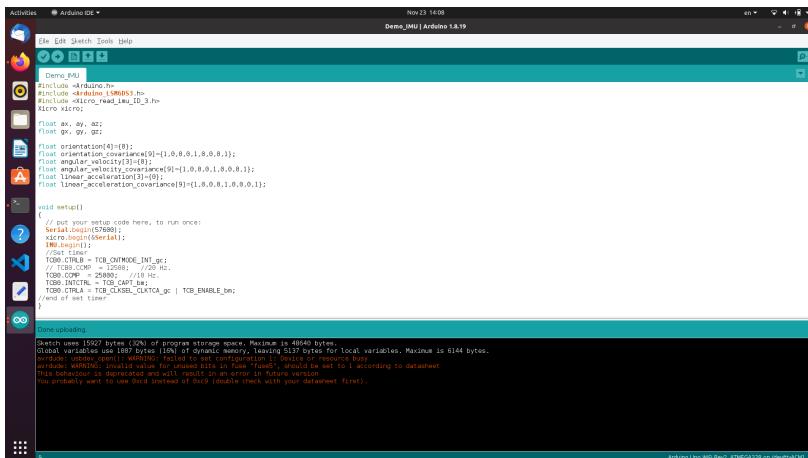
1. run on your workspace: `ros2 run xicro_pkg generate_xicro_node_py arduino`

```
monalilagonnalla-ASUS-TUF-Gaming-A15-FA506IU-FA506IU:~/xicro_ws/src/doc$ ros2 run xicro_pkg generate_xicro_node.py arduino
Get Idmcu Done.
Get Setup_Subscriber Done.
Get Namespace Done.
Get Setup_Subscriber Done.
Get Setup_Publisher Done.
Get Setup_Srv_client Done.
Generate Import Interface Done.
Get Topic_Done Done.
done load YAML pub.
Generate variable from msg Done.
Get Baudrate Done.
Topic >>> Imu_arduino Use : 217 bytes
Max Frequency On Topic Imu_arduino is : 53.08755760368663 Hz.
*****Calculate Only 1 Topic per Second*****
All topic average is : 53.08755760368663 Hz.
Get Setup_Srv_client Done.
Get Idmcu Done.
done load YAML srv-client.
generate Node Done.
generate Callback Done.
Get Baudrate Done.
-----generate xicro_node.py Done-----
Get Namespace Done.
Get Idmcu Done.
-----generate Entry_Point Done-----
```



-Connect Xicro with Arduino

1. Upload this Code On Board

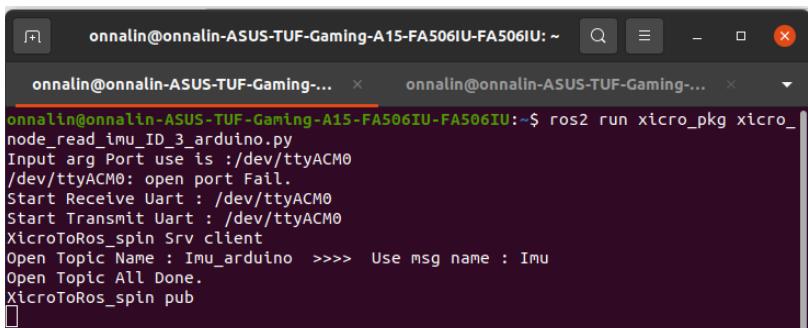


```
Demo.ino
#include <Arduino.h>
#include <Arduino_LSM6DS1.h>
#include <xicro_node_read_imu_ID_3.h>
Xicro xicro;
float ax, ay, az;
float gx, gy, gz;
float orientation[3];
float orientation_covariance[9] = {1,0,0,0,1,0,0,0,1};
float angular_velocity[3] = {0,0,0};
float angular_velocity_covariance[9] = {1,0,0,0,1,0,0,0,1};
float linear_acceleration[3] = {0,0,0};
float linear_acceleration_covariance[9] = {1,0,0,0,1,0,0,0,1};

void setup()
{
    // put your setup code here, to run once:
    Serial.begin(9600);
    xicro.beginSerial();
    IMU.begin();
    IMU.setI2CAddress(0x6B);
    IMU.setClock(25000); // 10 Hz
    IMU.setControl(CAPTURE);
    IMU.setControl(TOD_CTRLA | TOD_CXLSEL_CLATCA_0); // TOD_ENABLE_BE;
    IMU.setControl(0);
}

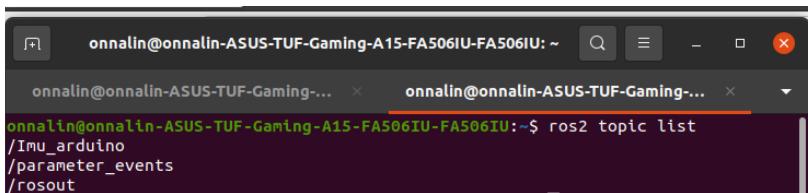
void loop()
{
    // put your main code here, to run repeatedly:
    xicro.read();
}
```

2. run command: ros2 run xicro_pkg xicro_node_read_imu_ID_3_arduino.py



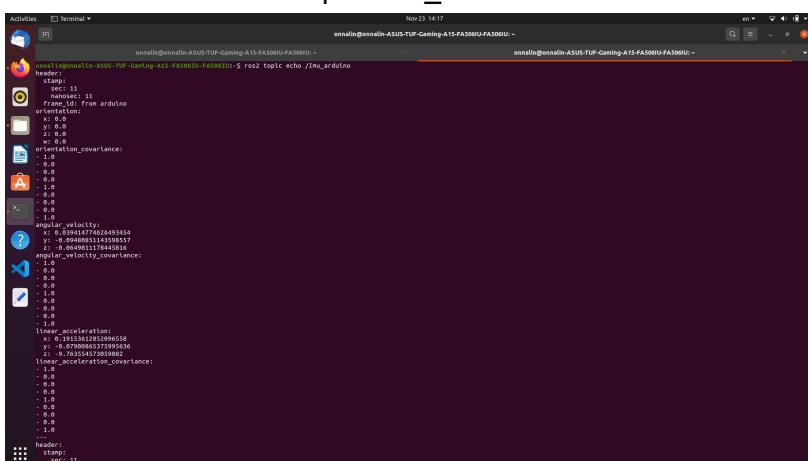
```
onnanlin@onnanlin-ASUS-TUF-Gaming-A15-FA506IU-FA506IU:~$ ros2 run xicro_pkg xicro_node_read_imu_ID_3_arduino.py
Input arg Port use is :/dev/ttyACM0
/dev/ttyACM0: open port Fail.
Start Receive Uart : /dev/ttyACM0
Start Transmit Uart : /dev/ttyACM0
XicroToRos_spin Srv client
Open Topic Name : Imu_arduino >>> Use msg name : Imu
Open Topic All Done.
XicroToRos_spin pub
```

3. Check Topic: ros2 topic list



```
onnanlin@onnanlin-ASUS-TUF-Gaming-A15-FA506IU-FA506IU:~$ ros2 topic list
/Imu_arduino
/parameter_events
/rosout
```

4. Echo data on topic as your interest: ros2 topic echo /topic_name This lab interest about topic /Imu_arduino

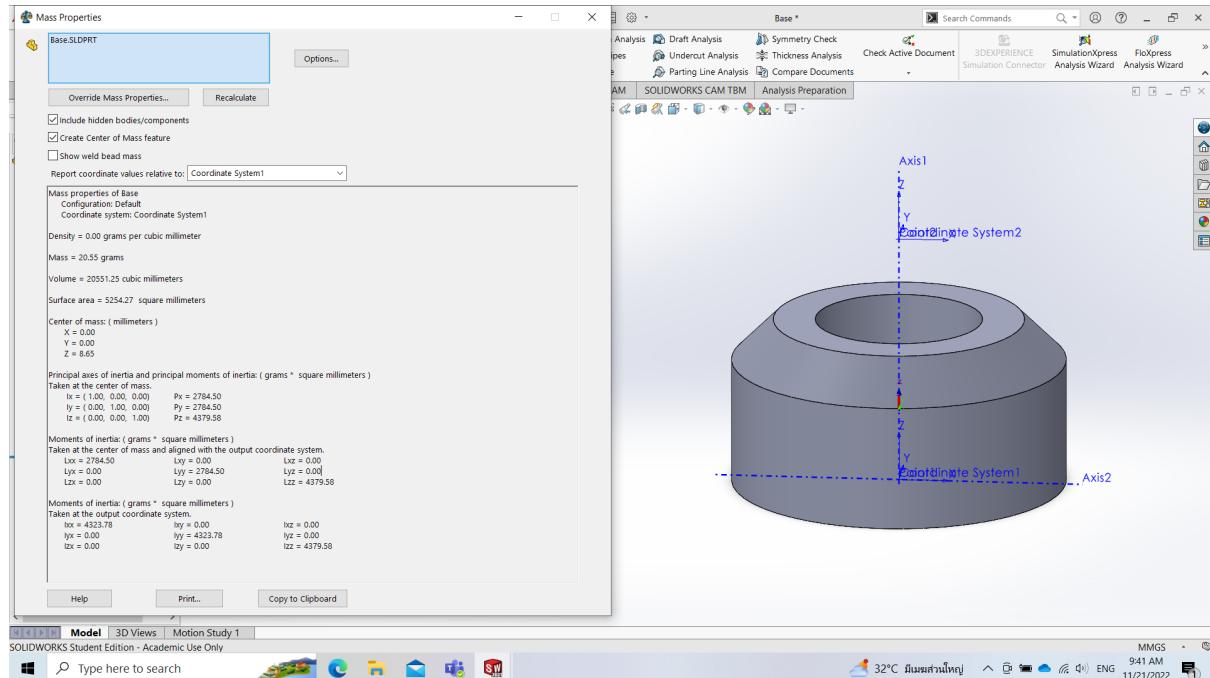


```
onnanlin@onnanlin-ASUS-TUF-Gaming-A15-FA506IU-FA506IU:~$ ros2 topic echo /Imu_arduino
header:
  seq: 13
  stamp:
    nanosec: 11
    sec: 1638447740
    usec: 2000000
  frame_id: 
  reference_id: 
  reference_frame: 
  orientation:
    x: 0.0
    y: 0.0
    z: 0.0
    w: 1.0
  orientation_covariance:
    0.0
    0.0
    0.0
    0.0
    0.0
    0.0
    0.0
    0.0
    0.0
  angular_velocity:
    x: 0.03914774026493454
    y: -0.00010000000000000002
    z: -0.064981178445816
  angular_velocity_covariance:
    1.0
    0.0
    0.0
    0.0
    1.0
    0.0
    0.0
    0.0
    1.0
  linear_acceleration:
    x: 0.151535853996558
    y: -0.0790086537399536
    z: 0.00010000000000000002
  linear_acceleration_covariance:
    1.0
    0.0
    0.0
    0.0
    1.0
    0.0
    0.0
    0.0
    1.0
  header:
    seq: 14
    stamp:
      nanosec: 11
      sec: 1638447740
      usec: 2000000
```

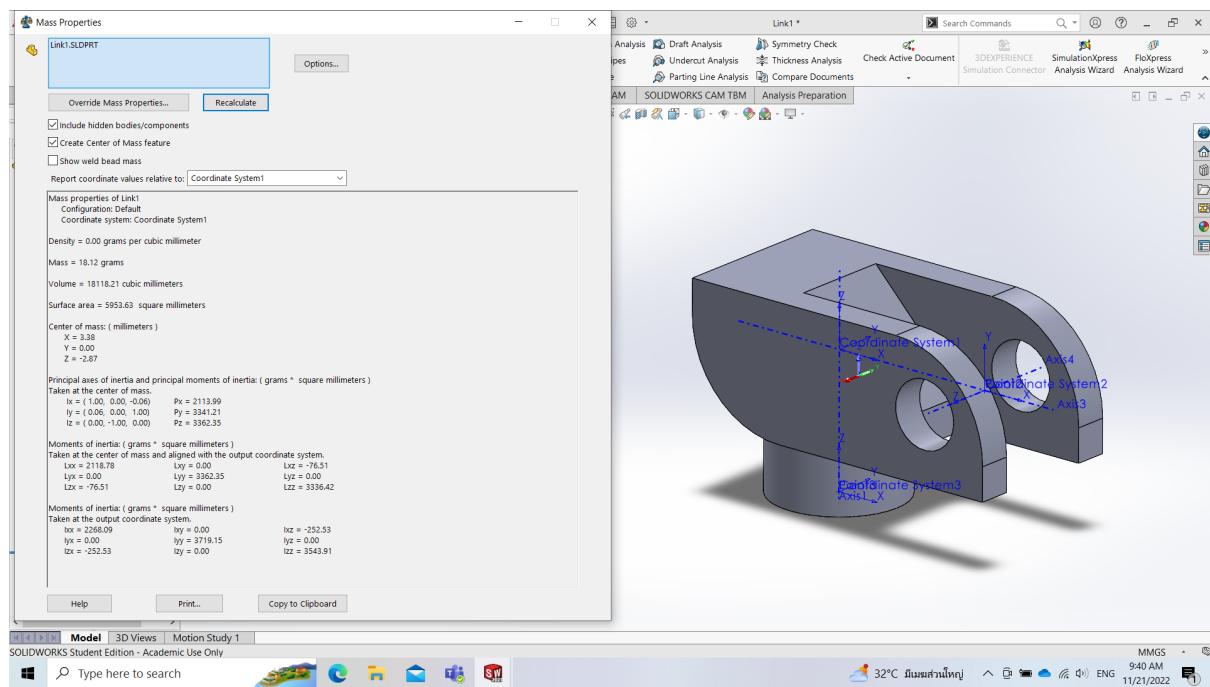
Now we can use topic /Imu_arduino for subscribe or something if you need.

Xacro file with Gazebo

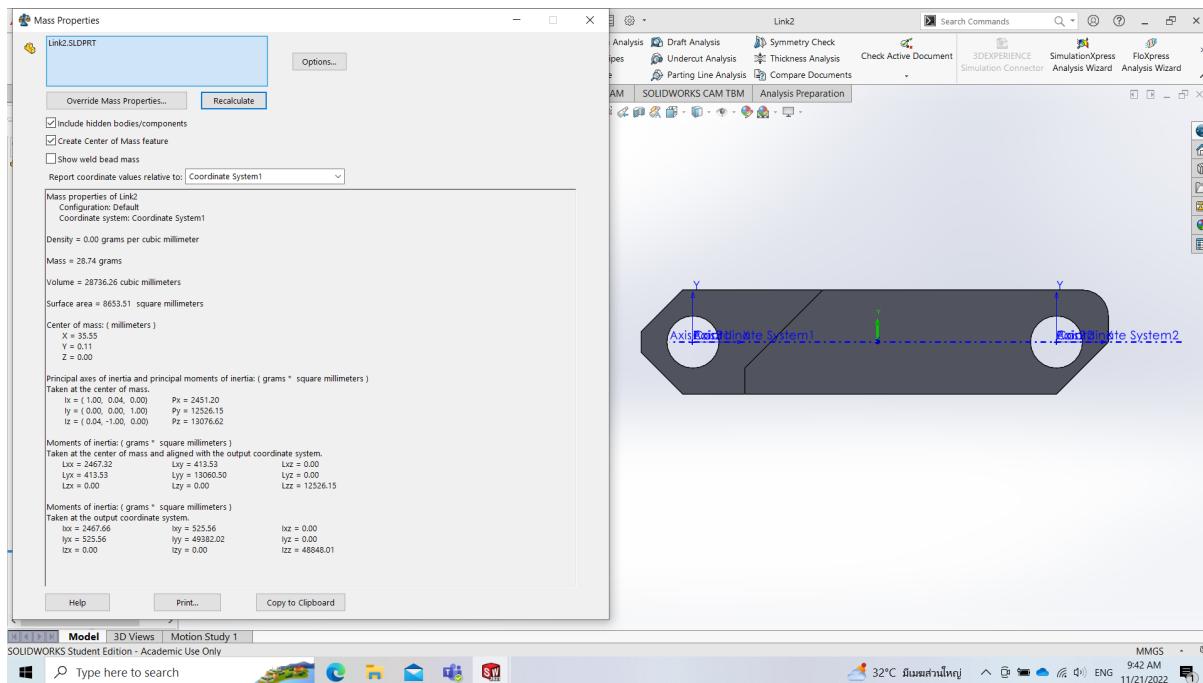
1. Design and Import Mass Properties from Solidworks link_0



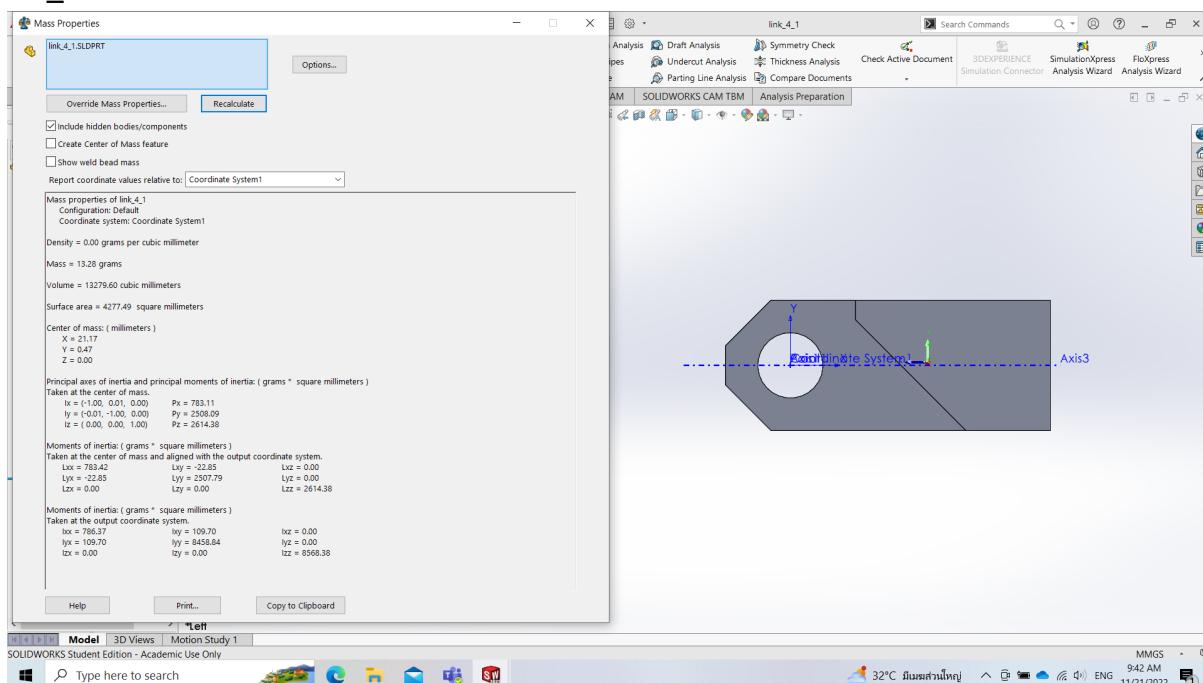
link_1



link_2



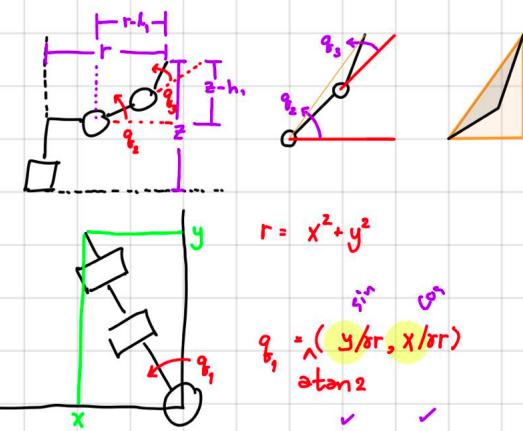
link_3



-Calculate Inverse Kinematics

Algorithm. X.

-Inverse kinematics.



$$\begin{cases} x = r - l_1 \\ y = z - h_1 \end{cases}$$

$$q_2 = \text{atan}_2(y, x) - \text{atan}_2(l_3 s_3, l_2 + l_3 s_3)$$

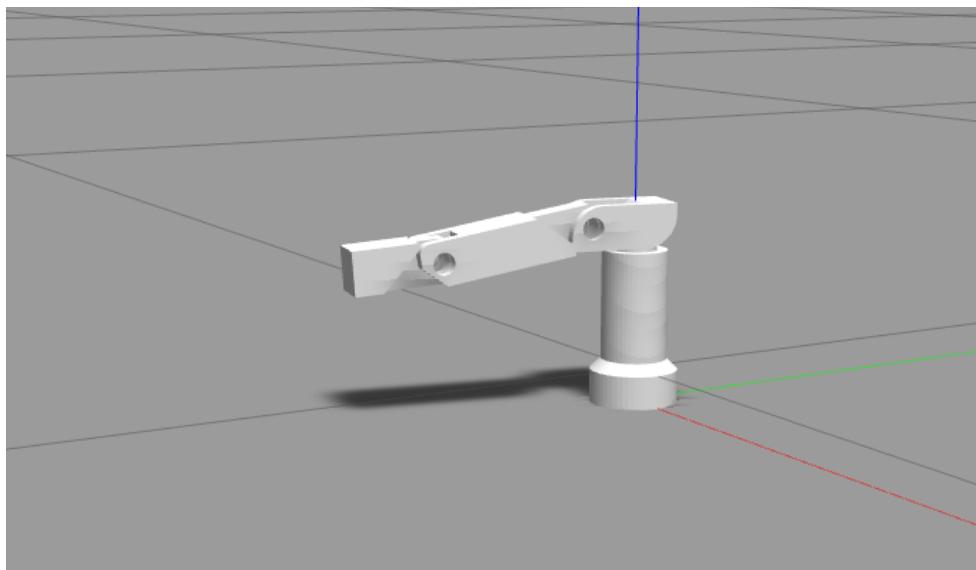
$$c_3 = \frac{x^2 + y^2 - l_2^2 - l_3^2}{2l_2 l_3}$$

$$s_3 = \sqrt{1 - c_3^2}$$

$$q_3 = \text{atan}_2(s_3, c_3)$$

in $q_1, x_2, y_2 \Rightarrow x_3, y_3$

ภาพการทดลอง



CODE: urdf.xacro

นำค่า จาก CAD มาใส่ใน urdf.xacro

```
<?xml version="1.0"?>
<robot xmlns:xacro="http://www.ros.org/wiki/xacro" name="robot">

    <!-- This is an example of a URDF. -->
    <!-- As we move through the file, new things to note will be pointed out. -->
    <!-- It's not meant an example of GOOD design, but an example of some of the
various features of URDF/xacro. -->

    <!-- This will include all the contents of example_include.xacro first. Go check
it out! -->
    <xacro:include filename="example_include.xacro" />

    <!-- This first link called "world" is empty -->
    <link name="world"></link>

    <!-- A simple fixed joint from our empty world link, to our base. -->
    <!-- The base origin is offset from the world origin. -->
    <joint name="base_joint" type="fixed">
        <origin xyz="0 0 0" rpy="0 0 0"/>
        <parent link="world"/>
        <child link="link_0"/>
    </joint>

    <!-- base_link is a large rectangular plate. Some things to note: -->
    <!-- - We set the visual origin Z to half the box height, so that the link origin sits
at the bottom of the box -->
    <!-- - We set the collision to be identical to the visual -->
    <!-- - We specified the colour manually (but still need to enter a name) -->
    <!-- - We specified all the inertial parameters manually -->
    <link name="link_0">
        <visual>
            <origin xyz="0 0 0.0" rpy="0 0 0"/>
            <geometry>
                <mesh filename="file:///$(find ros2_controller)/meshes/link_0.STL"/>
            </geometry>
            <material name="green">
                <color rgba="0.5 0.1 0.5 1"/>
            </material>
        </visual>
        <collision>
            <origin xyz="0 0 0.0" rpy="0 0 0"/>
            <contact_coefficients mu="0.5" kp="1000.0" kd="1.0"/>
            <geometry>
                <mesh filename="file:///$(find ros2_controller)/meshes/link_0.STL"/>
            </geometry>
        </collision>
    </link>
```

```

</collision>
<inertial>
<origin xyz="0 0 0.00865" rpy="0 0 0"/>
<mass value="0.02055" />
<inertia ixx="21.735" ixy="0.0" ixz="0.0" iyy="21.735" iyz="0.0" izz="7.570" />
</inertial>
</link>

<joint name="joint_0" type="fixed">
<parent link="link_0"/>
<child link="link_1"/>
<origin rpy="0 0 0.0" xyz="0 0 0.08"/>
<axis xyz="0 0 1"/>
<limit lower="-3.14" upper="3.14" effort="1000.0" velocity="1000.0"/>
</joint>

<link name="link_1">
<visual>
<origin xyz="0 0 0.0" rpy="0 0 0"/>
<geometry>
    <mesh filename="file://$(find ros2_controller)/meshes/link_1.STL"/>
</geometry>
<material name="green">
    <color rgba="0.2 1 0.2 1"/>
</material>
</visual>
<collision>
<origin xyz="0 0 0.0" rpy="0 0 0.0"/>
<contact_coefficients mu="0.2" kp="1000.0" kd="1.0"/>
<geometry>
    <mesh filename="file://$(find ros2_controller)/meshes/link_1.STL"/>
</geometry>
</collision>
<inertial>
<origin xyz="0.00338 0 -0.00287" rpy="0 0 0"/>
<mass value="0.01812" />
<inertia ixx="2.118" ixy="0.0" ixz="-0.07651" iyy="3.362" iyz="0.0" izz="3.336"
/>
</inertial>
</link>

<joint name="joint_1" type="revolute">
<parent link="link_0"/>
<child link="link_1"/>
<origin rpy="0 0 0.0" xyz="0 0 0.08"/>
<axis xyz="0 0 1"/>
<limit lower="-3.14" upper="3.14" effort="1500.0" velocity="1000.0"/>
<dynamics damping="0" friction="0"/>

```

```

</joint>

<link name="link_2">
<visual>
<origin xyz="0 0 0.0" rpy="0 0 0"/>
<geometry>
    <mesh filename="file:///$(find ros2_controller)/meshes/link_2.STL"/>
</geometry>
<material name="green">
    <color rgba="0.2 1 0.2 1"/>
</material>
</visual>
<collision>
<contact_coefficients mu="0" kp="1000.0" kd="1.0"/>
<origin xyz="0 0 0.0" rpy="0 0 0"/>
<geometry>
</geometry>
</collision>
<inertial>
<origin xyz="0.03555 0.00011 0.0" rpy="0 0 0.0"/>
<mass value="0.02874" />
<inertia ixx="2.467" ixy="0.413" ixz="0.0" iyy="13.060" iyz="0.0" izz="12.526" />
</inertial>
</link>

<joint name="joint_2" type="revolute">
<parent link="link_1"/>
<child link="link_2"/>
<origin rpy="1.57 0 0" xyz="0.025 0 0"/>
<axis xyz="0 0 1"/>
<limit lower="-3.14" upper="3.14" effort="1000.0" velocity="1000.0"/>
<dynamics damping="0" friction="0"/>
</joint>

<link name="link_3">
<visual>
<origin xyz="0 0 0" rpy="0 0 0"/>
<geometry>
    <mesh filename="file:///$(find ros2_controller)/meshes/link_3.STL"/>
</geometry>
<material name="green">
    <color rgba="0.2 1 0.2 1"/>
</material>
</visual>
<collision>
<contact_coefficients mu="0" kp="1000.0" kd="1.0"/>
<origin xyz="0 0 0.0" rpy="0 0 0"/>
<geometry>

```

```
</geometry>
</collision>
<inertial>
<origin xyz="0.02117 0 0.00047" rpy="0 0 0"/>
<mass value="0.01328" />
<inertia ixx="0.783" ixy="-0.0228" ixz="0.0" iyy="2.507" iyz="0.0" izz="2.614" />
</inertial>
</link>

<joint name="joint_3" type="revolute">
<parent link="link_2"/>
<child link="link_3"/>
<origin rpy="0 0.0 1.57" xyz="0.07 0 0"/>
<axis xyz="0 0 -1"/>
<limit lower="-3.14" upper="3.14" effort="1000.0" velocity="1000.0"/>
<dynamics damping="10" friction="5"/>
</joint>

<joint name="joint_eff" type="fixed">
<parent link="link_3"/>
<child link="end_effector"/>
<origin rpy="-1.57 0 1.57" xyz="0.4 0 0"/>
<axis xyz="0 0 1"/>
</joint>

<link name="end_effector">
</link>

<xacro:include filename="example_gazebo.xacro" />

</robot>
```