LAB3 DELIVERABLES
Jason Zhai
It contains everything for LAB3
The specified deliverables are highlighted with large red square

Task 1 - Setup a Workspace

1. **Action:** Create a workspace called `roscourse_ws`. This workspace should contain a subfolder named `src`
2. **Action:** Move into the `src` folder.
3. **Note:** It is recommended to have a terminal dedicated to building. This terminal should have the working directory `roscourse_ws`.

```
-----------------------------------------
ROS VERSION: ros-foxy | ROS_DOMAIN_ID: 66
-----------------------------------------
yahboom@VM:~$ cd Desktop/S25_RobotProgramming_Ardent
yahboom@VM:~/Desktop/S25_RobotProgramming_Ardent$ cd LAB3
yahboom@VM:~/Desktop/S25_RobotProgramming_Ardent/LAB3$ mkdir. roscourse_ws

Command 'mkdir.' not found, did you mean:

  command 'mkdir' from deb coreutils (8.30-3ubuntu2)

Try: sudo apt install <deb name>

yahboom@VM:~/Desktop/S25_RobotProgramming_Ardent/LAB3$ mkdir roscourse_ws
yahboom@VM:~/Desktop/S25_RobotProgramming_Ardent/LAB3$ cd roscourse_ws
yahboom@VM:~/Desktop/S25_RobotProgramming_Ardent/LAB3/roscourse_ws$ mkdir src
yahboom@VM:~/Desktop/S25_RobotProgramming_Ardent/LAB3/roscourse_ws$ cd src
```

Task 2 - Create the Webcam Package

1. **Action:** Within the `src` folder, create a package called `webcam` using `ros2 pkg create --build-type ament_python webcam`
2. **Question:** What sub-folders does this command create?

```
yahboom@VM:~/Desktop/S25_RobotProgramming_Ardent/LAB3/roscourse_ws/src$ ros2 pkg create --build-type ament_python webcam
going to create a new package
package name: webcam
destination directory: /home/yahboom/Desktop/S25_RobotProgramming_Ardent/LAB3/roscourse_ws/src
package format: 3
version: 0.0.0
description: TODO: Package description
maintainer: ['yahboom <"jzhai87@gmail.com">']
licenses: ['TODO: License declaration']
build type: ament_python
dependencies: []
creating folder ./webcam
creating ./webcam/package.xml
creating source folder
creating folder ./webcam/webcam
creating ./webcam/setup.py
creating ./webcam/setup.cfg
creating folder ./webcam/resource
creating ./webcam/resource/webcam
creating ./webcam/webcam/__init__.py
creating folder ./webcam/test
creating ./webcam/test/test_copyright.py
creating ./webcam/test/test_flake8.py
creating ./webcam/test/test_pep257.py
```

4. **Action:** Insert the template `cam_pub.py` and `cam_sub.py` files. Fill in the blanks. (These lines will have a set of hashtags above and below the line to edit.)

5. **Action:** Add the package dependencies: Open the file `package.xml` in `src/webcam`. After the lines with `<test_depend>`, add the following:

   - `<exec_depend>rclpy</exec_depend>`
   - `<exec_depend>sensor_msgs</exec_depend>`

6. **Action:** Add the entry points for the nodes: Edit the `setup.py` file to include the entry points (see Lecture 7 slide 19)

   - `'webcam_pub = webcam.cam_pub:main'`

```
yahboom@VM: ~/Desktop/S25_RobotProgramming_Ardent/...
-----------------------------------------
ROS VERSION: ros-foxy | ROS_DOMAIN_ID: 66
-----------------------------------------
yahboom@VM:~$ cd ~/Desktop/S25_RobotProgramming_Ardent/LAB3/roscourse_ws/src/web
cam/webcam
yahboom@VM:~/Desktop/S25_RobotProgramming_Ardent/LAB3/roscourse_ws/src/webcam/we
bcam$ touch cam_pub.py cam_sub.py
yahboom@VM:~/Desktop/S25_RobotProgramming_Ardent/LAB3/roscourse_ws/src/webcam/we
bcam$ nano cam_pub.py
yahboom@VM:~/Desktop/S25_RobotProgramming_Ardent/LAB3/roscourse_ws/src/webcam/we
bcam$ nano cam_sub.py
yahboom@VM:~/Desktop/S25_RobotProgramming_Ardent/LAB3/roscourse_ws/src/webcam/we
bcam$
```

yahboom@VM: ~/Desktop/S25_RobotProgramming_Ardent/...

```
  GNU nano 4.8                    cam_pub.py                    Modified
import rclpy
from rclpy.node import Node
from sensor_msgs.msg import Image
import cv2
from cv_bridge import CvBridge
import numpy as np
import sys

def imgmsg_to_cv2(img_msg):
    if img_msg.encoding != "bgr8":
        logger = rclpy.logging.get_logger("my_logger")
        logger.error("This Coral detect node has been hardcoded to the 'bgr8' e>

    dtype = np.dtype("uint8")
    dtype = dtype.newbyteorder('>' if img_msg.is_bigendian else '<')
    image_opencv = np.ndarray(shape=(img_msg.height, img_msg.width, 3), dtype=d>

    if img_msg.is_bigendian == (sys.byteorder == 'little'):
        image_opencv = image_opencv.byteswap().newbyteorder()

                                [ Mark Set ]
^G Get Help   ^O Write Out  ^W Where Is   ^K Cut Text   ^J Justify    ^C Cur Pos
^X Exit       ^R Read File  ^\ Replace    ^U Paste Text ^T To Spell   ^_ Go To Line
```

```
yahboom@VM:~/Desktop/S25_RobotProgramming_Ardent/LAB3/roscourse_ws/src/webcam/we
bcam$ nano ~/Desktop/S25_RobotProgramming_Ardent/LAB3/roscourse_ws/src/webcam/pa
ckage.xml
```

yahboom@VM: ~/Desktop/S25_RobotProgramming_Ardent/...

```
  GNU nano 4.8                    cam_sub.py
import rclpy
from rclpy.node import Node
from sensor_msgs.msg import Image
import cv2
from cv_bridge import CvBridge, CvBridgeError
import numpy as np
import sys

class WebcamSubscriber(Node):
    def __init__(self):
        super().__init__('webcam_subscriber')

        self.bridge = CvBridge()

        self.img_subscription = self.create_subscription(
            Image, 'image_raw', self.img_callback, 1
        )
        self.img_subscription  # Prevent unused variable warning

    def img_callback(self, img_msg):
                                [ Mark Set ]
^G Get Help   ^O Write Out  ^W Where Is   ^K Cut Text   ^J Justify    ^C Cur Pos
^X Exit       ^R Read File  ^\ Replace    ^U Paste Text ^T To Spell   ^_ Go To Line
```

```
yahboom@VM: ~/Desktop/S25_RobotProgramming_Ardent/...
.../Desktop/S25_RobotProgramming_Ardent/LAB3/roscourse_ws/src/webcam/package.xml
<?xml version="1.0"?>
<?xml-model href="http://download.ros.org/schema/package_format3.xsd" schematyp>
<package format="3">
  <name>webcam</name>
  <version>0.0.0</version>
  <description>TODO: Package description</description>
  <maintainer email="“jzhai87@gmail.com”">yahboom</maintainer>
  <license>TODO: License declaration</license>

  <test_depend>ament_copyright</test_depend>
  <test_depend>ament_flake8</test_depend>
  <test_depend>ament_pep257</test_depend>
  <test_depend>python3-pytest</test_depend>
  <exec_depend>rclpy</exec_depend>
  <exec_depend>sensor_msgs</exec_depend>


  <export>
    <build_type>ament_python</build_type>
  </export>
                                   [ Mark Set ]
^G Get Help   ^O Write Out ^W Where Is   ^K Cut Text   ^J Justify    ^C Cur Pos
^X Exit       ^R Read File ^\ Replace    ^U Paste Text^T To Spell   ^_ Go To Line
yahboom@VM:~/Desktop/S25_RobotProgramming_Ardent/LAB3/roscourse_ws/src/webcam/we
bcam$ nano ~/Desktop/S25_RobotProgramming_Ardent/LAB3/roscourse_ws/src/webcam/se
tup.py
```

```
yahboom@VM: ~/Desktop/S25_RobotProgramming_Ardent/...
...oom/Desktop/S25_RobotProgramming_Ardent/LAB3/roscourse_ws/src/webcam/setup.py
        ('share/ament_index/resource_index/packages',
            ['resource/' + package_name]),
        ('share/' + package_name, ['package.xml']),
    ],
    install_requires=['setuptools'],
    zip_safe=True,
    maintainer='yahboom',
    maintainer_email='“jzhai87@gmail.com”',
    description='TODO: Package description',
    license='TODO: License declaration',
    tests_require=['pytest'],
    entry_points={
    'console_scripts': [
        'webcam_pub = webcam.cam_pub:main',
        'webcam_sub = webcam.cam_sub:main',
    ],
}

)

^G Get Help   ^O Write Out ^W Where Is   ^K Cut Text   ^J Justify    ^C Cur Pos
^X Exit       ^R Read File ^\ Replace    ^U Paste Text^T To Spell   ^_ Go To Line
```

7. **Note:** We next need to build the package. It is recommended to have a dedicated build terminal. Whichever terminal is used to build, be sure you are in the root folder (`roscourse_ws`).

8. **Action:** Run `colcon build --symlink-install`

9. **Question:** What subfolders does this command create?

```
yahboom@VM:~/Desktop/S25_RobotProgramming_Ardent/LAB3/roscourse_ws/src/webcam/webcam$ cd ~/Desktop/S25_RobotProgramming_Ardent/LAB3/roscourse_ws
yahboom@VM:~/Desktop/S25_RobotProgramming_Ardent/LAB3/roscourse_ws$ colcon build --symlink-install
[1.125s] WARNING:colcon.colcon_core.package_identification:Failed to parse ROS package manifest in 'src/webcam': Error(s) in package 'src/webcam/package.xml':
Invalid email ""jzhai87@gmail.com"" for person "yahboom"
Starting >>> webcam
Finished <<< webcam [2.05s]

Summary: 1 package finished [2.91s]
yahboom@VM:~/Desktop/S25_RobotProgramming_Ardent/LAB3/roscourse_ws$
```

Test the Package

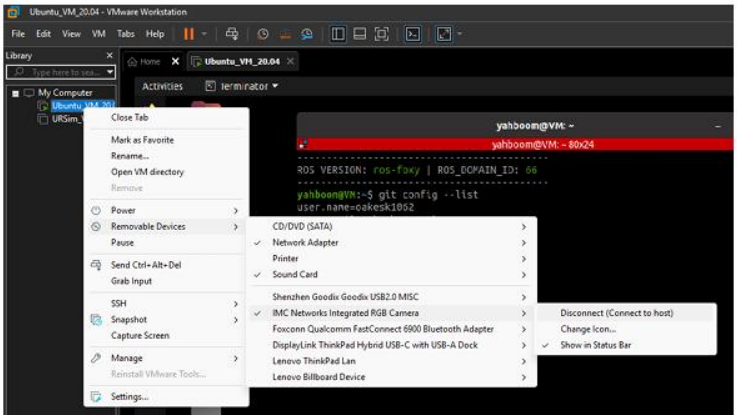
10. **Note:** Be sure to source in each terminal that you open to have access to these packages we are creating (note the ... is for you to fill in the necessary path between home and the workspace):
    `source ~/.../roscourse_ws/install/setup.bash`

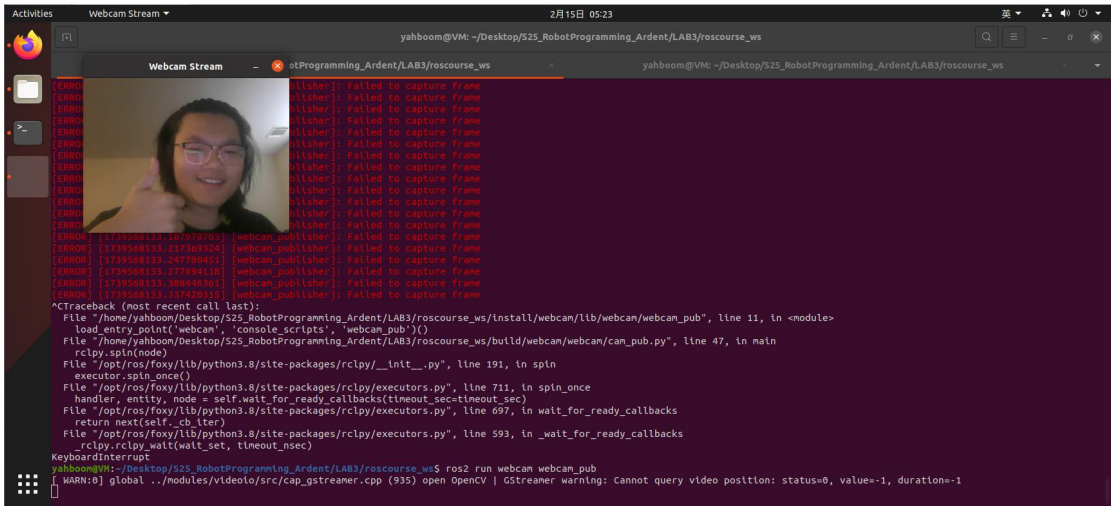
11. **Action:** Run `ros2 pkg executables webcam`

12. **Question:** What nodes are available to be run?

13. **Action:** Try running the publisher in one terminal and the subscriber in another.

14. **Note:** Webcam not accessible? Open the camera app on the host machine. A popup should appear allowing you to connect the camera to the virtual machine. If the popup does not appear, in VMware Workstation right-click on the VM name listed in the Library panel. From the menu select Removable Devices > Camera > Connect (Disconnect from Host)





Task 3 - Create the Python Turtle Interfaces

Create New Message Type

1. **Action:** Move to `~/roscourse_ws/src` and run `ros2 pkg create --build-type ament_cmake turtle_interfaces`
2. **Note:** We used the ament_cmake build type. Why? Custom interfaces (msg, srv, action files) can only be created in a CMake package. These interfaces can be used by any nodes after they have been created.
3. **Note:** We will first create some custom message types. We will store these in an msg folder.

```
yahboom@VM:~/Desktop/S25_RobotProgramming_Ardent/LAB3/roscourse_ws$ cd ~/Desktop/S25_RobotProgramming_Ardent/LAB3/roscourse_ws/src
yahboom@VM:~/Desktop/S25_RobotProgramming_Ardent/LAB3/roscourse_ws/src$ ros2 pkg create --build-type ament_cmake turtle_interfaces
going to create a new package
package name: turtle_interfaces
destination directory: /home/yahboom/Desktop/S25_RobotProgramming_Ardent/LAB3/roscourse_ws/src
package format: 3
version: 0.0.0
description: TODO: Package description
maintainer: ['yahboom <"jzhai87@gmail.com">']
licenses: ['TODO: License declaration']
build type: ament_cmake
dependencies: []
creating folder ./turtle_interfaces
creating ./turtle_interfaces/package.xml
creating source and include folder
creating folder ./turtle_interfaces/src
creating folder ./turtle_interfaces/include/turtle_interfaces
creating ./turtle_interfaces/CMakeLists.txt
yahboom@VM:~/Desktop/S25_RobotProgramming_Ardent/LAB3/roscourse_ws/src$
```

```
yahboom@VM:~/Desktop/S25_RobotProgramming_Ardent/LAB3/roscourse_ws/src$ mkdir ~/Desktop/S25_RobotProgramming_Ardent/LAB3/roscourse_ws/src/turtle_interfaces/msg
yahboom@VM:~/Desktop/S25_RobotProgramming_Ardent/LAB3/roscourse_ws/src$ nano ~/Desktop/S25_RobotProgramming_Ardent/LAB3/roscourse_ws/src/turtle_interfaces/msg/TurtleMsg.msg
yahboom@VM:~/Desktop/S25_RobotProgramming_Ardent/LAB3/roscourse_ws/src$
```

4. **Action:** Within `roscourse_ws/src/turtle_interfaces`, create a folder named `msg`
5. **Action:** Within this new folder, create the file `TurtleMsg.msg` containing the following
   - string name
   - geometry_msgs/Pose turtle_pose
   - string color
6. **Note:** Each line provides a different attribute within that message. The first part of the line gives the data type, the second the attribute name.
7. **Note:** To make this message usable, we need to add the appropriate dependencies to the `CMakeLists.txt` and `package.xml` files.

```
yahboom@VM: ~/Desktop/S25_RobotProgramming_Ardent/LAB3/roscourse_ws/src
yahboom@VM: ~/Desktop/S25_RobotProgramming_Ardent/LAB3/roscourse_ws/src    yahboom@VM: ~/Desktop/S25_RobotProgramming_Ardent/LAB3
  GNU nano 4.8                    /home/yahboom/Desktop/S25_RobotProgramming_Ardent/LAB3/roscourse_ws/src/turtle_interfaces/msg/TurtleMsg.msg
string name
geometry_msgs/Pose turtle_pose
string color
```

8. **Action:** Add the following lines to `CMakeLists.txt` BEFORE the line `ament_package()`
   - find_package(geometry_msgs REQUIRED)
   - find_package(rosidl_default_generators REQUIRED)
   - rosidl_generate_interfaces(${PROJECT_NAME}
     "msg/TurtleMsg.msg"
     DEPENDENCIES geometry_msgs)
9. **Note:** The rosidl handles the conversion of the interfaces into language specific code (e.g. C++ or Python).
10. **Note:** The first argument in the interface generator must be ${PROJECT_NAME} or there will be errors.

```
  GNU nano 4.8                    /home/yahboom/Desktop/S25_RobotProgramming_Ardent/LAB3/
endif()

if(CMAKE_COMPILER_IS_GNUCXX OR CMAKE_CXX_COMPILER_ID MATCHES "Clang")
  add_compile_options(-Wall -Wextra -Wpedantic)
endif()

# find dependencies
find_package(ament_cmake REQUIRED)
# uncomment the following section in order to fill in
# further dependencies manually.
# find_package(<dependency> REQUIRED)

if(BUILD_TESTING)
  find_package(ament_lint_auto REQUIRED)
  # the following line skips the linter which checks for copyrights
  # uncomment the line when a copyright and license is not present in all source files
  #set(ament_cmake_copyright_FOUND TRUE)
  # the following line skips cpplint (only works in a git repo)
  # uncomment the line when this package is not in a git repo
  #set(ament_cmake_cpplint_FOUND TRUE)
  ament_lint_auto_find_test_dependencies()
endif()
find_package(geometry_msgs REQUIRED)
find_package(rosidl_default_generators REQUIRED)

rosidl_generate_interfaces(${PROJECT_NAME}
  "msg/TurtleMsg.msg"
  "srv/SetPose.srv"
  "srv/SetColor.srv"
  DEPENDENCIES geometry_msgs
)

^G Get Help    ^O Write Out   ^W Where Is    ^K Cut Text    ^J Justify     ^C Cur Pos
^X Exit        ^R Read File   ^\ Replace     ^U Paste Text  ^T To Spell    ^_ Go To Line
```

11. **Action:** In the `package.xml` file, add the following:

- `<depend>geometry_msgs</depend>`
- `<buildtool_depend>rosidl_default_generators</buildtool_depend>`
- `<exec_depend>rosidl_default_runtime</exec_depend>`
- `<member_of_group>rosidl_interface_packages</member_of_group>`

12. **Action:** Create a new folder on the same level as `msg` called `srv`
13. **Note:** We will use this folder to store service interface definitions.
14. **Action:** In this `srv` folder, create a new file called `SetPose.srv` containing the following lines
    - geometry_msgs/PoseStamped turtle_pose
    - - - -
    - int8 ret
15. **Note:** Recall the '- - -' separates the 'request' part from the 'response' part of the service exchange. So the geometry message is sent by the client to the server. The server returns an integer.
16. **Action:** Create another file called `SetColor.srv` that contains
    - string color
    - - - -
    - int8 ret
17. **Question:** Based on the names, what do you anticipate services using these definitions will do?
18. **Note:** Like when we added the .msg file, we need to ensure the CMakeLists.txt and package.xml files have the required dependencies. The geometry msgs were already included as well as the rosidl when we modified the files last time. So all we need to do is add the .srv interfaces to the CMakesLists file.
19. **Action:** Within the rosidl_generate_interfaces section in `CMakeLists.txt`, add the following before DEPENDENCIES:

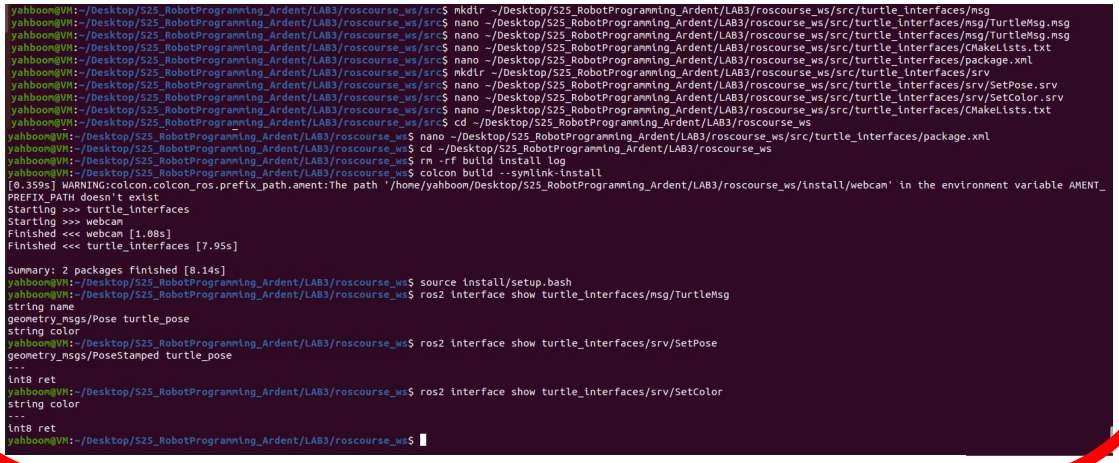
19. **Action:** Within the rosidl_generate_interfaces section in `CMakeLists.txt`, add the following before DEPENDENCIES:

    - "srv/SetPose.srv"
    - "srv/SetColor.srv"

Build the Package

20. **Action:** Within the root workspace folder (roscourse_ws), run the command `colcon build`.
21. **Note:** Be sure to source in each terminal that you open to have access to these packages we are creating: `source ~/roscourse_ws/install/setup.bash`
22. **Action:** Test your new package by using the following commands:

    - `ros2 interface show turtle_interfaces/msg/TurtleMsg`
    - `ros2 interface show turtle_interfaces/srv/SetPose`
    - `ros2 interface show turtle_interfaces/srv/SetColor`



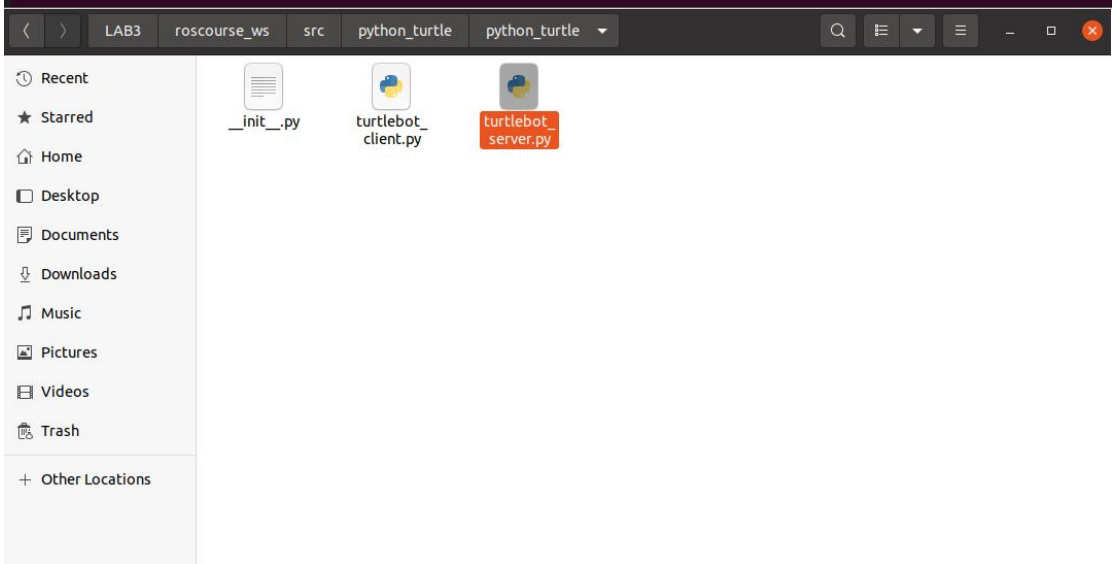
```
yahboom@VM:~/Desktop/S25_RobotProgramming_Ardent/LAB3/roscourse_ws/src$ mkdir ~/Desktop/S25_RobotProgramming_Ardent/LAB3/roscourse_ws/src/turtle_interfaces/msg
yahboom@VM:~/Desktop/S25_RobotProgramming_Ardent/LAB3/roscourse_ws/src$ nano ~/Desktop/S25_RobotProgramming_Ardent/LAB3/roscourse_ws/src/turtle_interfaces/msg/TurtleMsg.msg
yahboom@VM:~/Desktop/S25_RobotProgramming_Ardent/LAB3/roscourse_ws/src$ nano ~/Desktop/S25_RobotProgramming_Ardent/LAB3/roscourse_ws/src/turtle_interfaces/msg/TurtleMsg.msg
yahboom@VM:~/Desktop/S25_RobotProgramming_Ardent/LAB3/roscourse_ws/src$ nano ~/Desktop/S25_RobotProgramming_Ardent/LAB3/roscourse_ws/src/turtle_interfaces/CMakeLists.txt
yahboom@VM:~/Desktop/S25_RobotProgramming_Ardent/LAB3/roscourse_ws/src$ nano ~/Desktop/S25_RobotProgramming_Ardent/LAB3/roscourse_ws/src/turtle_interfaces/package.xml
yahboom@VM:~/Desktop/S25_RobotProgramming_Ardent/LAB3/roscourse_ws/src$ mkdir ~/Desktop/S25_RobotProgramming_Ardent/LAB3/roscourse_ws/src/turtle_interfaces/srv
yahboom@VM:~/Desktop/S25_RobotProgramming_Ardent/LAB3/roscourse_ws/src$ nano ~/Desktop/S25_RobotProgramming_Ardent/LAB3/roscourse_ws/src/turtle_interfaces/srv/SetPose.srv
yahboom@VM:~/Desktop/S25_RobotProgramming_Ardent/LAB3/roscourse_ws/src$ nano ~/Desktop/S25_RobotProgramming_Ardent/LAB3/roscourse_ws/src/turtle_interfaces/srv/SetColor.srv
yahboom@VM:~/Desktop/S25_RobotProgramming_Ardent/LAB3/roscourse_ws/src$ nano ~/Desktop/S25_RobotProgramming_Ardent/LAB3/roscourse_ws/src/turtle_interfaces/CMakeLists.txt
yahboom@VM:~/Desktop/S25_RobotProgramming_Ardent/LAB3/roscourse_ws/src$ cd ~/Desktop/S25_RobotProgramming_Ardent/LAB3/roscourse_ws
yahboom@VM:~/Desktop/S25_RobotProgramming_Ardent/LAB3/roscourse_ws$ nano ~/Desktop/S25_RobotProgramming_Ardent/LAB3/roscourse_ws/src/turtle_interfaces/package.xml
yahboom@VM:~/Desktop/S25_RobotProgramming_Ardent/LAB3/roscourse_ws$ cd ~/Desktop/S25_RobotProgramming_Ardent/LAB3/roscourse_ws
yahboom@VM:~/Desktop/S25_RobotProgramming_Ardent/LAB3/roscourse_ws$ rm -rf build install log
yahboom@VM:~/Desktop/S25_RobotProgramming_Ardent/LAB3/roscourse_ws$ colcon build --symlink-install
[0.359s] WARNING:colcon.colcon_ros.prefix_path.ament:The path '/home/yahboom/Desktop/S25_RobotProgramming_Ardent/LAB3/roscourse_ws/install/webcam' in the environment variable AMENT_
PREFIX_PATH doesn't exist
Starting >>> turtle_interfaces
Starting >>> webcam
Finished <<< webcam [1.08s]
Finished <<< turtle_interfaces [7.95s]

Summary: 2 packages finished [8.14s]
yahboom@VM:~/Desktop/S25_RobotProgramming_Ardent/LAB3/roscourse_ws$ source install/setup.bash
yahboom@VM:~/Desktop/S25_RobotProgramming_Ardent/LAB3/roscourse_ws$ ros2 interface show turtle_interfaces/msg/TurtleMsg
string name
geometry_msgs/Pose turtle_pose
string color
yahboom@VM:~/Desktop/S25_RobotProgramming_Ardent/LAB3/roscourse_ws$ ros2 interface show turtle_interfaces/srv/SetPose
geometry_msgs/PoseStamped turtle_pose
---
int8 ret
yahboom@VM:~/Desktop/S25_RobotProgramming_Ardent/LAB3/roscourse_ws$ ros2 interface show turtle_interfaces/srv/SetColor
string color
---
int8 ret
yahboom@VM:~/Desktop/S25_RobotProgramming_Ardent/LAB3/roscourse_ws$
```

Task 4 - Create the Python Turtle Package

1. **Action:** Create a new package called `python_turtle` with build type `ament_python`
2. **Action:** Copy the files `turtlebot_server.py` and `turtlebot_client.py` into the `roscourse_ws/src/python_turtle/python_turtle` folder.
3. **REQUIRED Question:** Provide a description of what each of these files do. You could highlight some key lines or commands, what they do, what the arguments signify, etc.

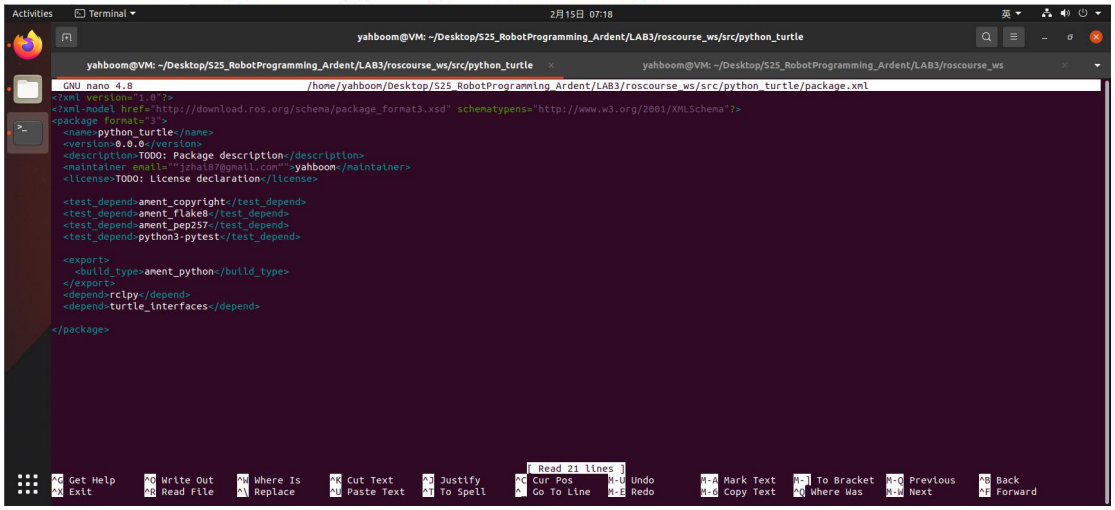
```
yahboom@VM:~/Desktop/S25_RobotProgramming_Ardent/LAB3/roscourse_ws$ cd ~/Desktop/S25_RobotProgramming_Ardent/LAB3/roscourse_ws/src
yahboom@VM:~/Desktop/S25_RobotProgramming_Ardent/LAB3/roscourse_ws/src$ ros2 pkg create --build-type ament_python python_turtle
going to create a new package
package name: python_turtle
destination directory: /home/yahboom/Desktop/S25_RobotProgramming_Ardent/LAB3/roscourse_ws/src
package format: 3
version: 0.0.0
description: TODO: Package description
maintainer: ['yahboom <"jzhai87@gmail.com">']
licenses: ['TODO: License declaration']
build type: ament_python
dependencies: []
creating folder ./python_turtle
creating ./python_turtle/package.xml
creating source folder
creating folder ./python_turtle/python_turtle
creating ./python_turtle/setup.py
creating ./python_turtle/setup.cfg
creating folder ./python_turtle/resource
creating ./python_turtle/resource/python_turtle
creating ./python_turtle/python_turtle/__init__.py
creating folder ./python_turtle/test
creating ./python_turtle/test/test_copyright.py
creating ./python_turtle/test/test_flake8.py
creating ./python_turtle/test/test_pep257.py
yahboom@VM:~/Desktop/S25_RobotProgramming_Ardent/LAB3/roscourse_ws/src$
```

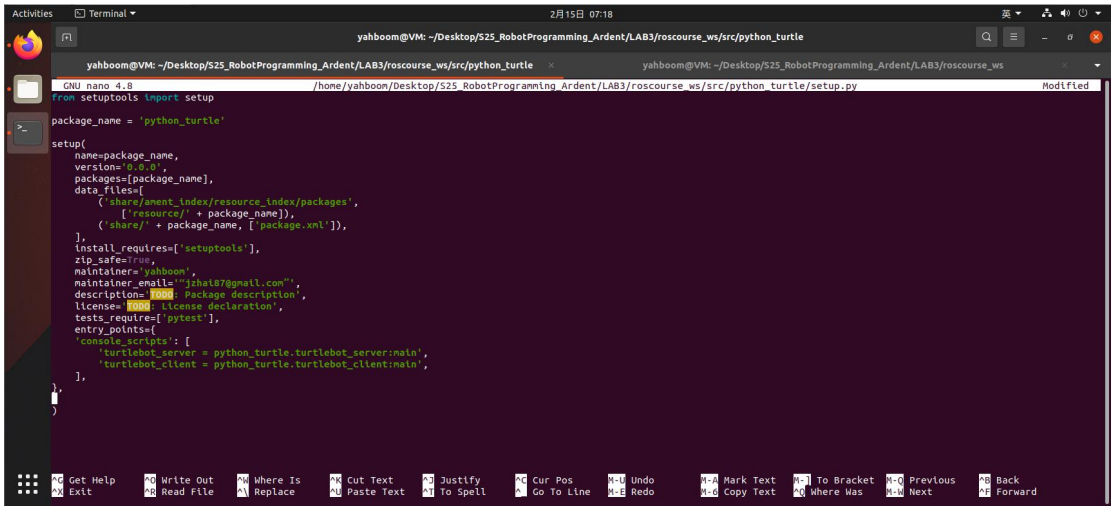


4. **Action:** In the package.xml file, add <depend> tags for rclpy and turtle_interfaces.

5. **Action:** In the setup.py file, add the following entry points (be sure to use commas between entries):

   - 'turtlebot_server = python_turtle.turtlebot_server:main'
   - 'turtlebot_client = python_turtle.turtlebot_client:main'



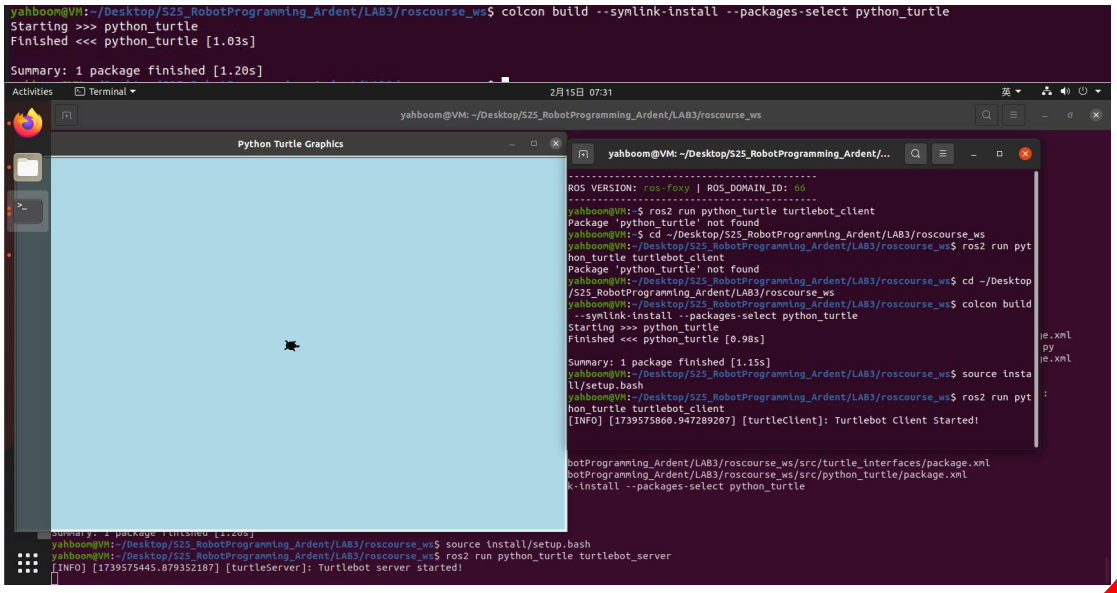


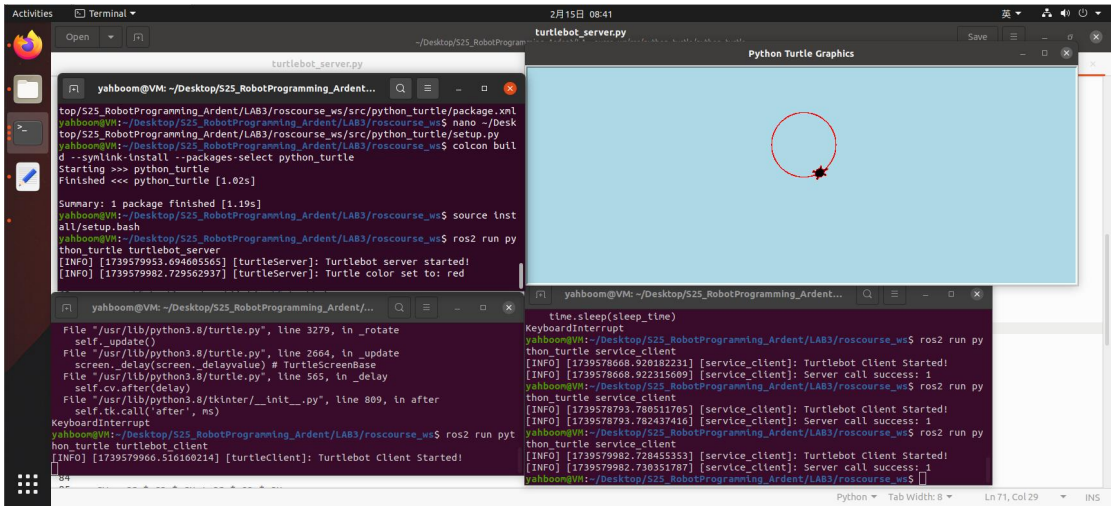
6. **Action:** Build and source as we have done previously.

   - **Note:** Want to save time? In the terminal, type: Ctrl+R. This allows you to search your command history. Start typing source and your previous input appears! Tab to put this command in the terminal. Press Ctrl+R again to cycle through all commands that meet the search criteria.

7. **Action:** Run the turtlebot server in one terminal and the client in the other.

8. **Question:** What happened?




Activities
Terminal
2月15日 08:41
Open
turtlebot_server.py
Save
Python Turtle Graphics
yahboom@VM: ~/Desktop/S25_RobotProgramming_Ardent...
top/S25_RobotProgramming_Ardent/LAB3/roscourse_ws/src/python_turtle/package.xml
yahboom@VM:~/Desktop/S25_RobotProgramming_Ardent/LAB3/roscourse_ws$ nano ~/Desk
top/S25_RobotProgramming_Ardent/LAB3/roscourse_ws/src/python_turtle/setup.py
yahboom@VM:~/Desktop/S25_RobotProgramming_Ardent/LAB3/roscourse_ws$ colcon buil
d --symlink-install --packages-select python_turtle
Starting >>> python_turtle
Finished <<< python_turtle [1.02s]
Summary: 1 package finished [1.19s]
yahboom@VM:~/Desktop/S25_RobotProgramming_Ardent/LAB3/roscourse_ws$ source inst
all/setup.bash
yahboom@VM:~/Desktop/S25_RobotProgramming_Ardent/LAB3/roscourse_ws$ ros2 run py
thon_turtle turtlebot_server
[INFO] [1739579953.694605565] [turtleServer]: Turtlebot server started!
[INFO] [1739579982.729562937] [turtleServer]: Turtle color set to: red
yahboom@VM: ~/Desktop/S25_RobotProgramming_Ardent/...
File "/usr/lib/python3.8/turtle.py", line 3279, in _rotate
self._update()
File "/usr/lib/python3.8/turtle.py", line 2664, in _update
screen._delay(screen._delayvalue) # TurtleScreenBase
File "/usr/lib/python3.8/turtle.py", line 565, in _delay
self.cv.after(delay)
File "/usr/lib/python3.8/tkinter/__init__.py", line 809, in after
self.tk.call('after', ms)
KeyboardInterrupt
yahboom@VM:~/Desktop/S25_RobotProgramming_Ardent/LAB3/roscourse_ws$ ros2 run pyt
hon_turtle turtlebot_client
[INFO] [1739579966.516160214] [turtleClient]: Turtlebot Client Started!
yahboom@VM: ~/Desktop/S25_RobotProgramming_Ardent...
time.sleep(sleep_time)
KeyboardInterrupt
yahboom@VM:~/Desktop/S25_RobotProgramming_Ardent/LAB3/roscourse_ws$ ros2 run py
thon_turtle service_client
[INFO] [1739578668.920182231] [service_client]: Turtlebot Client Started!
[INFO] [1739578668.922315609] [service_client]: Server call success: 1
yahboom@VM:~/Desktop/S25_RobotProgramming_Ardent/LAB3/roscourse_ws$ ros2 run py
thon_turtle service_client
[INFO] [1739578793.780511705] [service_client]: Turtlebot Client Started!
[INFO] [1739578793.782437416] [service_client]: Server call success: 1
yahboom@VM:~/Desktop/S25_RobotProgramming_Ardent/LAB3/roscourse_ws$ ros2 run py
thon_turtle service_client
[INFO] [1739579982.728455353] [service_client]: Turtlebot Client Started!
[INFO] [1739579982.730351787] [service_client]: Server call success: 1
yahboom@VM:~/Desktop/S25_RobotProgramming_Ardent/LAB3/roscourse_ws$
Python
Tab Width: 8
Ln 71, Col 29
INS