Custom msg,srv and action

Step 1:Create a workspace

```
mkdir -p ~/ros2_ws/src
cd ~/ros2_ws
```

Step 2: To make custom msg and srv and action we need the c++ package

```
cd ~/ros2_ws/src
ros2 pkg create custom_bringup --build-type ament_cmake
```

Step 3: Add a .msg or .srv or .action file

1. mkdir msg —> cd msg
gedit CustomMsg.msg
[Location: custom_bringup/msg/CustomMsg.msg] for custom msg

2. mkdir srv —>cd srv
gedit CustomSrv.srv
[Location: custom_bringup/srv/CustomSrv.srv] for custom srv

3. mkdir action —>cd action
gedit Customaction.action
[Location: custom_bringup/action/Customaction.action] for custom action

Step 4: Put this inside:

cd msg —>gedit CustomMsg.msg
custom_bringup/msg/CustomMsg.msg

```
int64 datai
float32 dataf
string datas
```

cd .. & cd srv
custom_bringup/srv/CustomSrv.srv

```
int64 a
int64 b
---
int64 sum
```

cd.. & cd action
custom_bringup/action/CustomAction.action

```
int32 order
---
int32[] sequence
---
int32[] partial_sequence
```

After using list command show the folders in that Package.xml

Step 5: Edit package.xml

File path:~/ros2_ws/src/custom_bringup/package.xml:

```
<?xml version="1.0"?>
<package format="3">
  <name>my_interfaces</name>
  <version>0.0.1</version>
  <description>Custom message definitions</description>
  <maintainer email="you@example.com">Your Name</maintainer>
  <license>Apache-2.0</license>

  <buildtool_depend>ament_cmake</buildtool_depend>

  <!-- For message generation -->
  <build_depend>rosidl_default_generators</build_depend>
  <exec_depend>rosidl_default_runtime</exec_depend>
  <member_of_group>rosidl_interface_packages</member_of_group>
  <!-- Runtime dependency so other packages can use the generated code -->
  <exec_depend>rosidl_default_runtime</exec_depend>

  <member_of_group>rosidl_interface_packages</member_of_group>
</package>
```

Step 6: Edit CMakeLists.txt

File path: ~/ros2_ws/src/custom_bringup/CMakeLists.txt:

```
cmake_minimum_required(VERSION 3.5)

project(my_interfaces)

find_package(ament_cmake REQUIRED)

find_package(rosidl_default_generators REQUIRED)

find_package(builtin_interfaces REQUIRED)

# Generate code for our .msg

rosidl_generate_interfaces(${PROJECT_NAME}

  "msg/CustomMsg.msg"

  "srv/CustomSrv.srv"

  "action/CustomAction.action"

  DEPENDENCIES builtin_interfaces

)

ament_export_dependencies(rosidl_default_runtime)

ament_package()
```

Step 7:Build & source

```
cd ~/ros2_ws

colcon build

source install/setup.bash
```

# Check your new message type is visible:

1. ros2 interface show custom_bringup/msg/CustomMsg   (for custom .msg)

```
int64 datai

float32 dataf

string datas
```

2. ros2 interface show custom_bringup/srv/CustomSrv

```
int64 a

int64 b

---

int64 sum
```

3. ros2 interface show custom_bringup/action/CustomAction

```
int32 order

---

int32[] sequence

—

int32[] partial_sequence
```

Use the custom message in a Python publisher

Create a Python package

cd ~/ros2_ws/src

ros2 pkg create <pkg_name> --build-type ament_python

Inside this pkg →package.xml pkg_custom/ resource/ setup.cfg setup.py test

1.Add the node

Create ~/ros2_ws/src/<pkg_name>/<pkg_name>**/cus_pub.py:**

```
import rclpy

from rclpy.node import Node

from custom_bringup.msg import CustomMsg



class CustomPublisher(Node):

   def __init__(self):

       super().__init__('custom_publisher')

       self.publisher_ = self.create_publisher(CustomMsg,
'custom_topic', 10)

       self.timer = self.create_timer(1.0, self.timer_callback)

       self.counter = 0



   def timer_callback(self):
```

```python
        msg = CustomMsg()
        msg.datai = self.counter
        msg.dataf = float(self.counter) * 1.1
        msg.datas = f"Message number {self.counter}"
        self.publisher_.publish(msg)
        self.get_logger().info(f"Publishing: {msg.datai}, {msg.dataf:.2f}, '{msg.datas}'")
        self.counter += 1


def main(args=None):
    rclpy.init(args=args)
    node = CustomPublisher()
    rclpy.spin(node)
    node.destroy_node()
    rclpy.shutdown()


if __name__ == '__main__':
    main()
```

Create ~/ros2_ws/src/<pkg_name>**/**<pkg_name>**/cus_sub.py:**

```python
import rclpy
```

```python
from rclpy.node import Node
from custom_bringup.msg import CustomMsg


class CustomSubscriber(Node):
    def __init__(self):
        super().__init__('custom_subscriber')
        self.subscription = self.create_subscription(
            CustomMsg,
            'custom_topic',
            self.listener_callback,
            10)


    def listener_callback(self, msg):
        self.get_logger().info(
            f"Received: datai={msg.datai}, dataf={msg.dataf:.2f}, datas='{msg.datas}'"
        )


def main(args=None):
    rclpy.init(args=args)
    node = CustomSubscriber()
    rclpy.spin(node)
    node.destroy_node()
    rclpy.shutdown()
```

```python
if __name__ == '__main__':
    main()
```

- Use previous code in server and client to import custom srv

  from custom_bringup.srv import CustomSrv

- Use pervious code in action server and client to import custom action

  from custom_bringup.msg import CustomAction

2. Edit ~/ros2_ws/src/<pkg_name>/setup.py:

Example:

```
entry_points={
        'console_scripts': [
            'servo_pub = servo_pub_py.servo_pub:main',
        ],
    },
)
```

3. Update package.xml

```
<depend>custom_bringup</depend>
```

4. Build & run

```
cd ~/ros2_ws
```

```
colcon build

source install/setup.bash

# Terminal 1 (publisher):

ros2 run servo_pub_py servo_pub


# Terminal 2 (subscriber):

source ~/ros2_ws/install/setup.bash

ros2 run servo_sub_cpp servo_sub
```