ROS2 Actions Cheat Sheet

ROS2 Robotics Developer Course - Using ROS2 In Python

Create Action File

Example: action/ActionName.action

```
geometry_msgs/Point my_goal
---
float32 my_result
---
float32 my_feedback
```

Modify package.xml

Include IDL And Action Msgs Dependencies

```
<build_depend>rosidl_default_generators</build_depend>
<exec_depend>rosidl_default_runtime</exec_depend>
<member_of_group>rosidl_interface_packages</member_of_group>

<depend>action_msgs</depend>
```

Modify CMakeLists.txt

Include IDL Generation, Any Message Dependencies

```
find_package(rosidl_default_generators REQUIRED)
find_package(geometry_msgs REQUIRED)
```

Include Custom Action Files, Dependencies

```
rosidl_generate_interfaces(${PROJECT_NAME}
  "action/Navigate2D.action"
  DEPENDENCIES geometry_msgs
)
```

.py Create Python Scripts

Import Action Functionality, Your Custom Action

```
import rclpy
from rclpy.node import Node
from rclpy.action import ActionServer, ActionClient
from my_pkg_name.action import ActionName
```

Create Action Server Node

```
class NavigateActionServer(Node):
    def __init__(self):
        super().__init__("my_action_server_node")
        self._action_server = ActionServer(self, ActionName, "action_topic",
                                           self.action_callback)

    def action_callback(self, goal_handle):
        goal_point = goal_handle.request.my_goal
        feedback_msg = ActionName.Feedback()

        while waiting_for_my_goal_to_be_met():
            feedback_msg.my_feedback = do_something()
            goal_handle.publish_feedback(feedback_msg)
            rclpy.spin_once(self, timeout_sec=1)

        goal_handle.succeed()
        result = ActionName.Result()
        result.my_result = do_something_2()

        return result
```

Create Action Client Node

```python
class NavigateActionClient(Node):
    def __init__(self):
        super().__init__("my_action_client_node")
        self._action_client = ActionClient(self, ActionName, "action_topic")

    def send_goal(self, user_point_input):
        goal_msg = ActionName.Goal()
        goal_msg.my_goal = user_point_input

        self._action_client.wait_for_server()

        self._send_goal_future = self._action_client.send_goal_async(goal_msg,
                                 feedback_callback=self.feedback_callback)

        self._send_goal_future.add_done_callback(self.goal_response_callback)


    def feedback_callback(self, feedback_msg):
        do_something(feedback_msg.feedback.my_feedback)

    def goal_response_callback(self, future):
        goal_handle = future.result()

        if goal_handle.accepted == False:
            print("Goal Rejected")
            return None

        self._get_result_future = goal_handle.get_result_async()
        self._get_result_future.add_done_callback(self.get_result_callback)


    def get_result_callback(self, future):
        result = future.result().result
        do_something_2(result.my_result)
```