Модуль 03 - Robot Operating System

Написание Action и Service

Резюме: ROS предлагает дополнительные способы взаимодействия между нодами и, в данном модуле, вы их изучите - действия (Action) и сервисы (Service).

Глава I

Преамбула

Эффективные методы коммуникации важны как для отдельных программ роботов, так и для межроботного или межкомпьютерного взаимодействия в рамках одного робота. Использование ROS позволяет упростить создание "мозгов" роботов с распределенными вычислениями, обработкой датчиков и сетевыми возможностями. Готовые и сложные коммуникационные механизмы, предоставляемые ROS, позволяют быстро разрабатывать роботов.

Глава II

Инструкции

• Используйте эту страницу как единственное описание задач. Не слушайте никаких слухов и домыслов о том, как приготовить свой программное решение.
• Здесь и далее мы используем ROS2 Humble и C++/Python.
• Обратите внимание на тип ваших файлов и каталогов.
• Для оценки ваше решение должно находиться в вашем репозитории GitHub.
• Вы не должны оставлять никаких дополнительных файлов в своем каталоге, кроме тех, которые явно указаны в теме. Рекомендуется изменить ваш .gitignore, чтобы избежать конфликтных случаев.
• Когда вам нужно получить точный результат в вашей программе, запрещается отображать предварительно рассчитанный результат вместо правильного выполнения упражнения.
• Есть вопрос? Спросите у соседа справа. В противном случае спросите вашего соседа слева.
• Ваш справочник: коллеги/интернет/google.
• Вы можете задавать вопросы в telegram.
• Внимательно прочитайте примеры. Они вполне могут прояснить детали, которые явно не упомянуты в теме.

Глава III

Упражнение 01

Упражнение 01: Написание простого сервиса и клиента
Каталог для хранения вашего решения: ex01/
Файлы которые должны находиться в каталоге: Все необходимые файлы пакета
Разрешенные функции: Название пакета “service_full_name”, узлы должны называться и запускаться по именам “service_name” и “client_name” через ros2 run. Имя сервиса должно быть SummFullName.
Комментарии:

В этом упражнении вам нужно будет дописать шаблон, созданный в предыдущем модуле.

Прочтите и следуйте этим руководствам шаг за шагом:
https://docs.ros.org/en/humble/Tutorials/Beginner-Client-Libraries/Writing-A-Simple-Cpp-Service-And-Client.html
https://docs.ros.org/en/humble/Tutorials/Beginner-Client-Libraries/Writing-A-Simple-Py-Service-And-Client.html

Напишите узел который будет состоять из двух программ. “service_name” будет через сервис получать три строки с фамилией, именем и отчеством, складывать их в одну строку и отправлять результат обратно клиенту. “client_name” будет получать фамилию, имя, отчество через аргументы и склеивать их в одну строку через сервис. Имя сервиса должно быть SummFullName.

В папку ex01 скопируйте все необходимые файлы для компиляции и запуска вашего пакета.

Упражнение 02

Упражнение 02: Написание простого сервера действий и клиента действий
Каталог для хранения вашего решения: ex02/
Файлы которые должны находиться в каталоге: Все необходимые файлы пакета
Разрешенные функции: Название пакета “action_turtle_commands”, узлы должны называться и запускаться по именам “action_turtle_server” и “action_turtle_client” через ros2 run. Имя action должно быть MessageTurtleCommands. Симулятор TurtleSim. Можно только посылать команды черепахе в топик cmd_vel и подписываться на топик pose.
Комментарии: Телепорт использовать нельзя!

Литература для понимания принципа Actions:
https://docs.ros.org/en/humble/Concepts/Basic/About-Actions.html
https://docs.ros.org/en/humble/Tutorials/Beginner-CLI-Tools/Understanding-ROS2-Actions/Understanding-ROS2-Actions.html
Руководства для непосредственного выполнения задания:
https://docs.ros.org/en/humble/Tutorials/Intermediate/Creating-an-Action.html
https://docs.ros.org/en/humble/Tutorials/Intermediate/Writing-an-Action-Server-Client/Cpp.html
https://docs.ros.org/en/humble/Tutorials/Intermediate/Writing-an-Action-Server-Client/Py.html

Напишите пакет который будет состоять из двух программ. Узел “action_server” будет через Action получать команду на движение и управлять черепахой через топик cmd_vel, возвращая результат только после того как черепаха выполнила действие. Набор действий которые Action должен выполнять: проехать вперед на заданное расстояние указанное в метрах или повернуть на заданный угол указанный в градусах. Разработать узел “action_client” который должен послать последовательность команд Action, которые примет “action_server”, так чтобы черепаха проехала вперед 2 метра, повернула направо на 90 градусов и проехала еще один метр. В процессе выполнения Action должен возвращать текущее пройденное расстояние (одометрия) с момента начала выполнения команды.

Структура сообщений Action:
MessageTurtleCommands.action

```
# goal definition
string command      # “forward”,”turn_left”, “turn_right”
int32 s             # расстояние в метрах которое должна проехать черепаха
int32 angle         # угол в градусах на который должна повернуть черепаха
---
# result definition
bool result         # true - черепаха выполнила команду, false - черепаха не смогла выполнить команду, например потому что выполнение команды прервали
---
#feedback
int32 odom          # пройденное черепахой расстояние в метрах с момента начала выполнения команды
```

Упражнение 03

Упражнение 03: Запись и воспроизведение данных
Каталог для хранения вашего решения: ex03/
Файлы, которые должны находиться в каталоге: turtle_cmd_vel.mcap, pose_speed_x1.yaml, pose_speed_x2.yaml
Разрешенные функции:
Комментарии:

В этом упражнении вам нужно будет изучить "Запись и воспроизведение данных".

Прочтите и следуйте этому руководству:
https://docs.ros.org/en/humble/Tutorials/Beginner-CLI-Tools/Recording-And-Playing-Back-Data/Recording-And-Playing-Back-Data.html

Запустите turtlesim и запишите в ros2 bag файл команды скорости из топика /cmd_vel управляя черепахой через teleop_key. Черепаха должна начать двигаться из центра по периметру на расстоянии от границ симулятора и примерно вернуться в тоже место. Записан должен быть только один топик /turtle1/cmd_vel. Bag файл должен называться turtle_cmd_vel.mcap.

Проиграйте ros2 bag файл turtle_cmd_vel.mcap и сохраните координаты движения черепахи из топика /turtle1/pose в файл pose_speed_x1.yaml.

Проиграйте ros bag файл turtle_cmd_vel.mcap с удвоенной скоростью воспроизведения и сохраните координаты движения черепахи из топика /turtle1/pose в файл pose_speed_x2.yaml.

Воспроизведенная траектория не обязана точно повторять записанную.

Затем сохраните файлы turtle_cmd_vel.mcap, pose_speed_x1.yaml, pose_speed_x2.yaml в папку ex03.

Упражнение 04

Упражнение 04: Начало работы с ros2 doctor
Каталог для хранения вашего решения: ex04/
Файлы должны находиться в каталоге: doctor.txt
Разрешенные функции:
Комментарии:

В этом упражнении вам необходимо изучить "Использование ros2doctor для выявления проблем".

Прочитайте и следуйте этому руководству:
https://docs.ros.org/en/humble/Tutorials/Beginner-Client-Libraries/Getting-Started-With-Ros2doctor.html

Проверьте состояние вашей ROS системы и сохраните его в файл doctor.txt (полный отчет).
Если есть проблемы, то устраните их.

Упражнение 05

Упражнение 05: Навигация по вики ROS
Каталог для хранения вашего решения: ex05/
Файлы в каталоге: Все необходимые файлы пакета
Разрешенные функции: топики turtleX/cmd_vel и turtleX/pose.
Запрещенные функции: turtleX/teleport_absolute, turtleX/teleport_relative. Нельзя использовать управление черепахой с клавиатуры.
Комментарии:

Просмотрите предыдущие туториалы по TurtleSim с помощью поиска по документации: https://docs.ros.org/en/humble/search.html?q=turtlesim&check_keywords=yes&area=default

После чего напишите ноду, которая будет перемещать черепаху в заданные координаты. Координаты цели должны задаваться через параметры в командной строке к вашему cpp/py исполняемому файлу в виде x, y, theta. Черепаха должна перемещаться вашей программой автономно только с использованием отправки скорости в топик cmd_vel, команды turtleX/teleport_absolute и turtleX/teleport_relative использовать нельзя.

Ваш пакет и cpp/py файл должны называться move_to_goal. Он должен запускаться через ros2 run с тремя параметрами обозначающими координаты цели к которой черепаха должна проехать.

В папку ex05 скопируйте все необходимые файлы для компиляции и запуска вашего пакета.