ORC-PG-ROS2-WEEK7

ROS 기본 프로그래밍

기본 이름 규칙

1. Snake_case : 파일 이름 및 변수명, 함수명
2. CamelCased : 타입 및 클래스
3. ALL_CAPITALS : 상수

:: ROS 인터페이스 파일은 /msg 및 /srv 또는 /action 폴더에 위치시키며 인터페이스 파일명은 CamelCased 규칙을 따른다.

C++ Style

1. **기본 규칙 :** C++14 Standard5
2. **라인 길이 :** 최대 100문자
3. **이름 규칙(Naming)**

■ CamelCased, snake_case, ALL_CAPITALS만을 사용한다.
• CamelCased： 타입, 클래스, 구조체. 열거형
• snake.case： 파일, 패키지, 인터페이스, 네임스페이스, 변수, 함수, 메소드
• ALL_CAPITALS： 상수, 매크로
■ 소스 파일은 cpp 확장자를 사용한다.
■ 헤더 파일은 hpp 확장자를 사용한다.
■ 전역변수(Global variable)를 반드시 사용해야 할 경우 접두어(g_)를 붙인다.
■ 클래스 멤버 변수(Class member variable} 는 마지막에 밑줄(_)을 붙인다.

4. **공백 문자 대 탭(Spaces vs. Tabs)**

■ 기본 들여쓰기(Indent)는 공백 문자(Space) 2개를 사용한다(탭(Tab} 문자 사용 금지).
■ Class의 접근 지정자(public, protected, private)는 들여쓰기를 하지 않는다.

5. **괄호(Brace)**

■ if, else, do, while, for 구문에 괄호를 사용

Python Style

1. **기본 규칙 :** 파이썬 3(파이썬 3.5 이상)을 사용
2. **라인 길이 :** 최대 100문자
3. **이름 규칙(Naming)**

■ CamelCased, snake_case, ALL_CAPITALS만을 사용한다.
- CamelCased： 타입, 클래스
- snake_case： 파일, 패키지, 인터페이스, 모듈, 변수, 함수, 메소드
- ALL_CAPITALS： 상수

4. **공백 문자 대 탭(Spaces vs. Tabs)**

■ 기본 들여쓰기(Indent)는 공백 문자(Space) 4개를 사용한다(탭(Tab) 문자 사용 금지)

5. **괄호(Brace) :**

괄호는 계산식 및 배열 인덱스로 사용하며, 자료형에 따라 적절한 괄호(대괄호 [ ],중괄호 {}, 소괄호 ())를 사용

6. **주석**

■ 문서 주석은 """을 사용하며 Docstring Conventions을 기술한 PEP 257 을 준수한다.
■ 구현 주석은 #을 사용

7. **린터(Linters)**

■ 파이썬 코드 스타일의 자동 오류 검출을 위하여 ament_flake8을 사용

8. **기타**

■ 모든 문자는 큰 따옴표(", Double quotes)가 아닌 작은 따옴표(’, Single quote)를 사용하여 표현

ROS 프로그래밍 기초(파이썬)

패키지 생성

```
$ ros2 pkg create [패키지이름] --build-type [빌드 타입]
--dependencies [의존하는패키지1] [의존하는패키지 n]
```

패키지 설정 파일(package.xml)

패키지 설정 파일(package.xml)은 사용할 RCL(ROS 2 client libraries)에 따라 달라지는데 C++ 는 build_=type을 ament_cmake로 설정하고, 파이썬은 ament_python으로 설정

파이썬 패키지 설정 파일(setup.py)

`주의사항`

패키지 설정 파일(setup.py)에서 주의할 점은 entry_points 옵션의 console_scripts 키를 사용한 실행 파일 설정

ex)

helloworld_publisher와 helloworld_subscriber 콘솔 스크립트는 각각
my_first_ros_rclpy_pkg.helloworld_publisher 모듈과
my_first_ros_rclpy_pkg.helloworld_subcriber 모듈의 main 함수를 호출.

해당 설정을 통해 ros2 run 또는 ros2 launch 명령어로 해당 스크립트를 실행시킬 수 있음

~~파이썬 패키지 환경설정 파일(setup.cfg) pass~~

~~퍼블리셔 노드 작성 pass~~

~~서브스크라이버 노드 작성 pass~~

빌드

ROS2의 특정 패키지 또는 전체 패키지 빌드 시에는 colcon 빌드 툴 사용

특정 패키지 선택 빌드 `—-packages-select`

심볼릭 파일 형태 저장 `—-symlink-install 옵션`

ROS 2 TIPS

ROS 2 프로그래밍에 필요한 설정

설정 스크립트(setup script)

새로운 패키지를 빌드했다면 항상 설정 스크립트를 터미털 창에 실행

`$ source ~/robot_ws/install/local_setup.bash`

Setup.bash vs local_setup.bash

overlay 개발환경은 설치된 ROS 패키지들에 의존하기에 underlay 개발환경에 종속적. 그렇기 때문에 설정 스크립트(Setup script)라고 하는 setup.bash의 호출 순서 및 사용 방법이 조금씩 다름

setup.bash 및 local_setup.bash 사용 방법

local_setup.bash

스크립트가 위치해 있는 접두사(prefix) 경로의 모든 패키지에 대한 환경 설정. 상위 작업 공간에 대한 설정 포함x

setup.bash

현재 작업 공간이 빌드될 때 환경에 제공된 다른 모든 작업 공간에 대한 local_setup.bash 스크립트를 포함. underlay 개발환경의 설정 정보까지 포함

colcon_cd

colcon_cd 명령을 사용하면 셸의 현재 작업 디렉터리를 패키지 디렉터리로 빠르게 변경 가능.

`$ colcon_cd 패키지이름`

사용하기 위한 설정

```
source /usr/share/colcon_cd/function/colcon_cd.sh
export _colcon_cd_root=~/robot_ws
```

ROS_DOMAIN_ID 환경변수

DDS Global Space를 손쉽게 변경하는 방법

동일 네트워크를 다른 사람들과 함께 사용하고 있다면 서로 다른 ROS_DOMAIN_ID를 각각 지정하여 사용

```
$ export ROS_DOMAIN_ID=11
$ ros2 run demo_nodes_cpp talker
[INFO]: Publishing: 'Hello World: 1'
[INFO]: Publishing: 'Hello World: 2'
[INFO]: Publishing: 'Hello World: 3'
(생략)
```

```
$ export ROS_DOMAIN_ID=12
$ ros2 run demo_nodes_cpp listener
(서로 다른 도메인을 사용하고 있기에 아무런 반응이 없을 것이다.)
```

```
$ export ROS_DOMAIN_ID=11
$ ros2 run demo_nodes_cpp listener
```

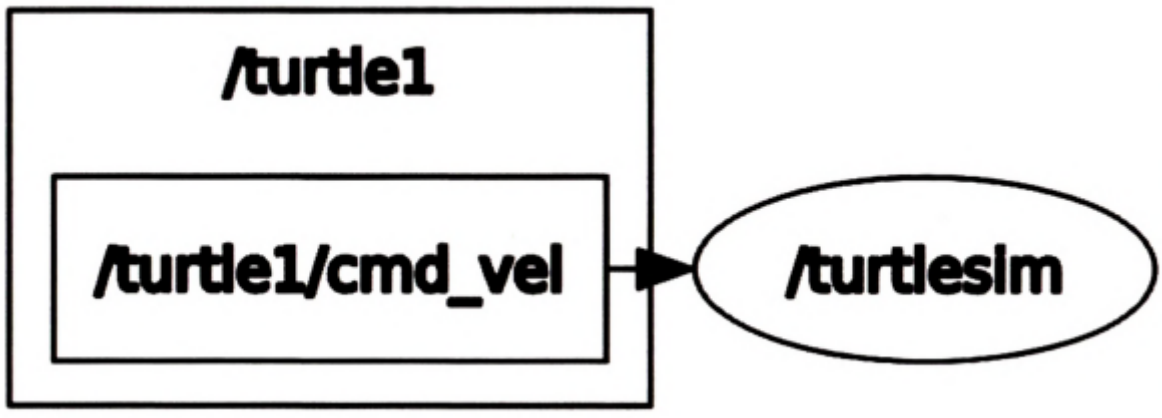
```
[INFO]: I heard: [Hello World: 13]
[INFO]: I heard: [Hello World: 14]
[INFO]: I heard: [Hello World: 15]
(생략)
```

ROS 2 Namespace

설정된 고유 이름에 Namespace를 붙여 주면 독립적 네트워크 그룹화 가능

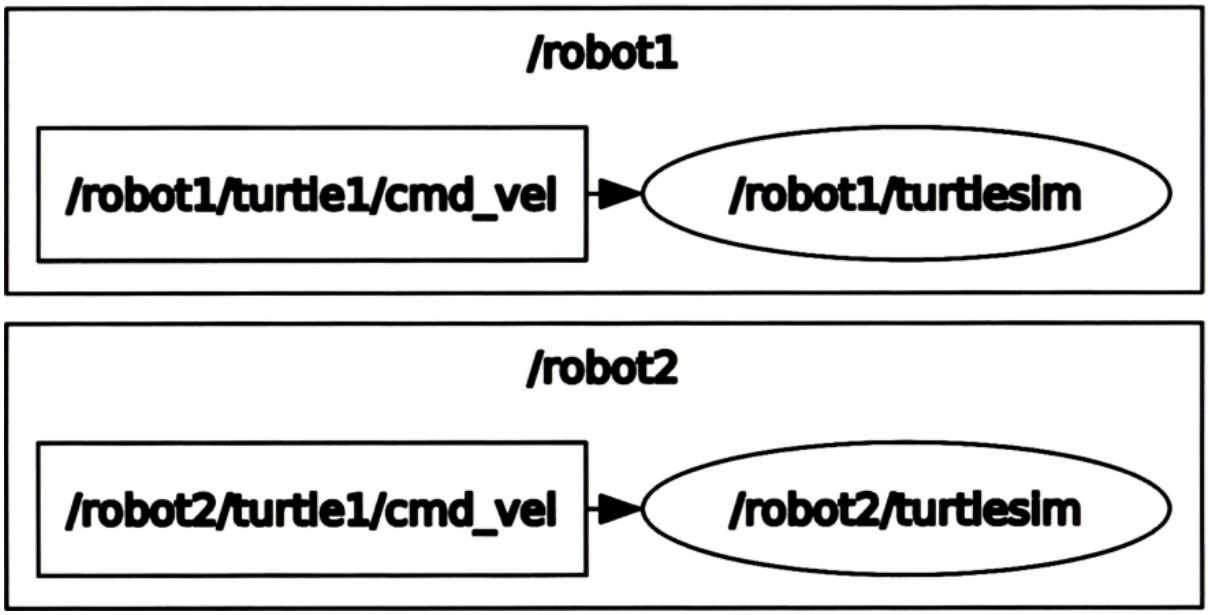
`$ ros2 run turtlesim turtlesim_node`

`$ rqt_graph`



`$ ros2 run turtlesim turtlesim_node --ros-args -r __ns:=/robot1`

`$ ros2 run turtlesim turtlesim_node --ros-args -r __ns:=/robot2`



colcon과 vcstool 명령어 자동 완성 기능 추가

자동 완성 기능을 위해 파일 소싱 필요

```
source /usr/share/colcon_argcomplete/hook/colcon-argcomplete.bash
source /usr/share/vcstool-completion/vcs.bash
```