ROS 기초

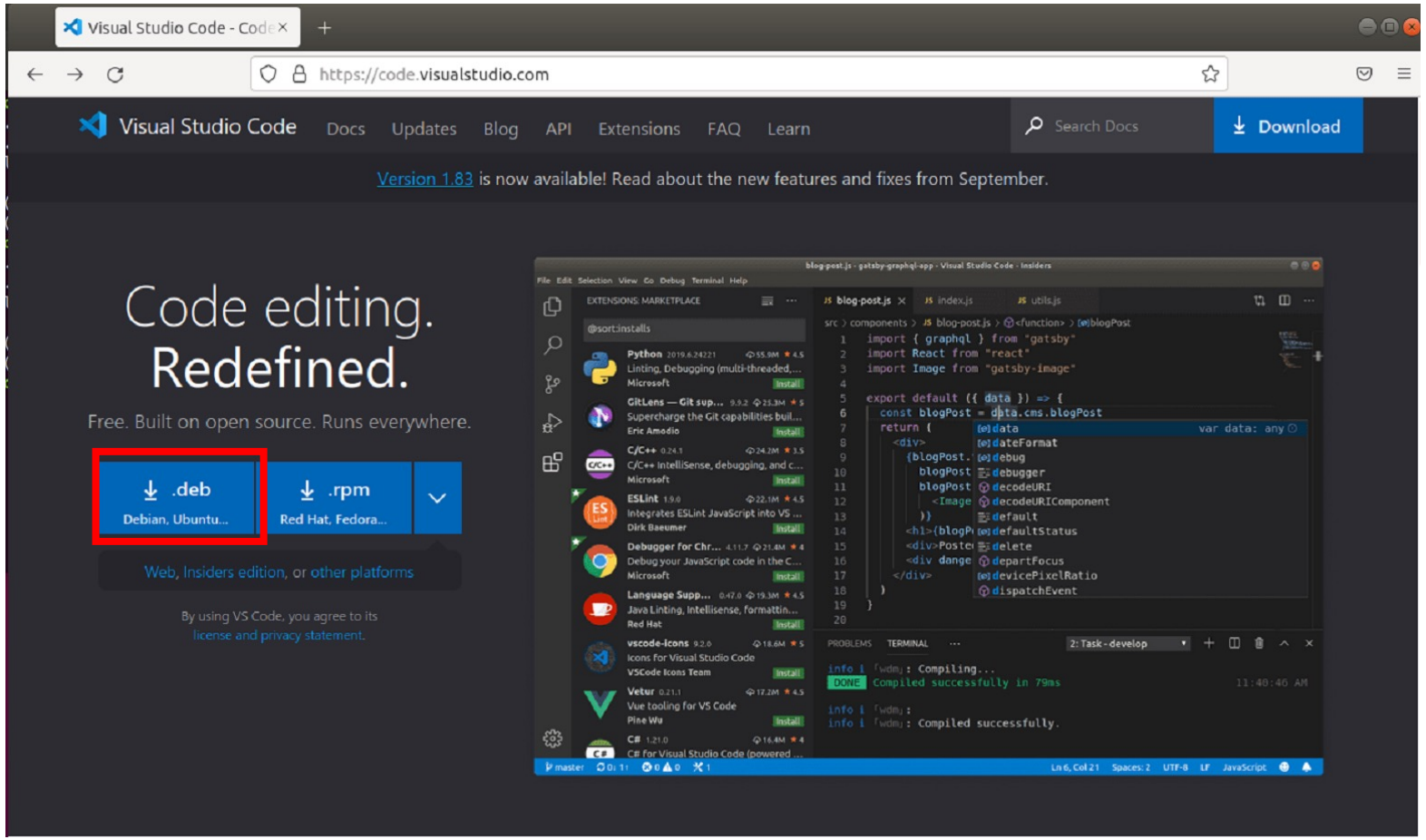
ROS
Robot Operating System

Infinyx
THINK LIFE SYNC AI

01. ROS 프로그램 개발환경 구축

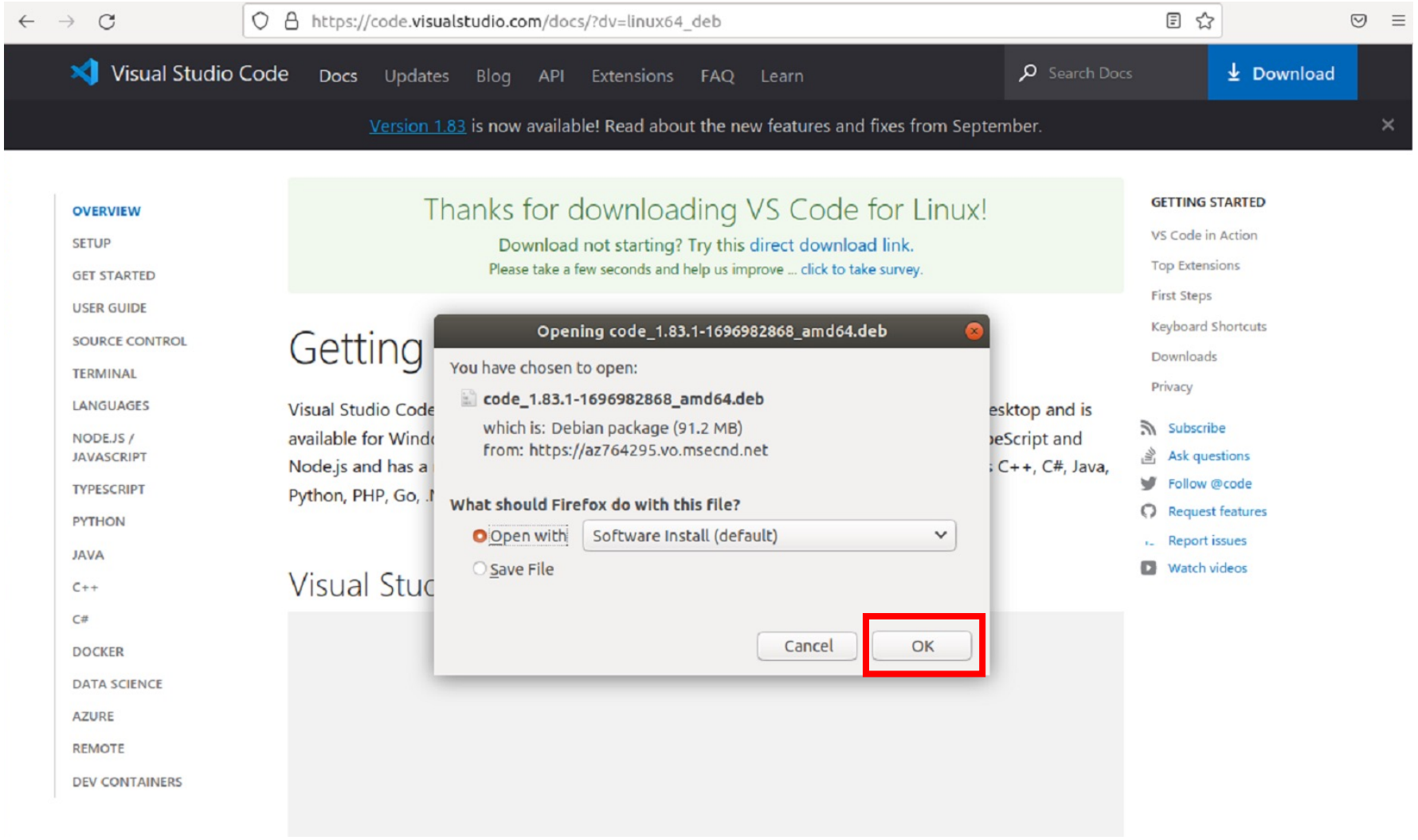
VSCODE 다운로드

- 다운로드 : https://code.visualstudio.com/



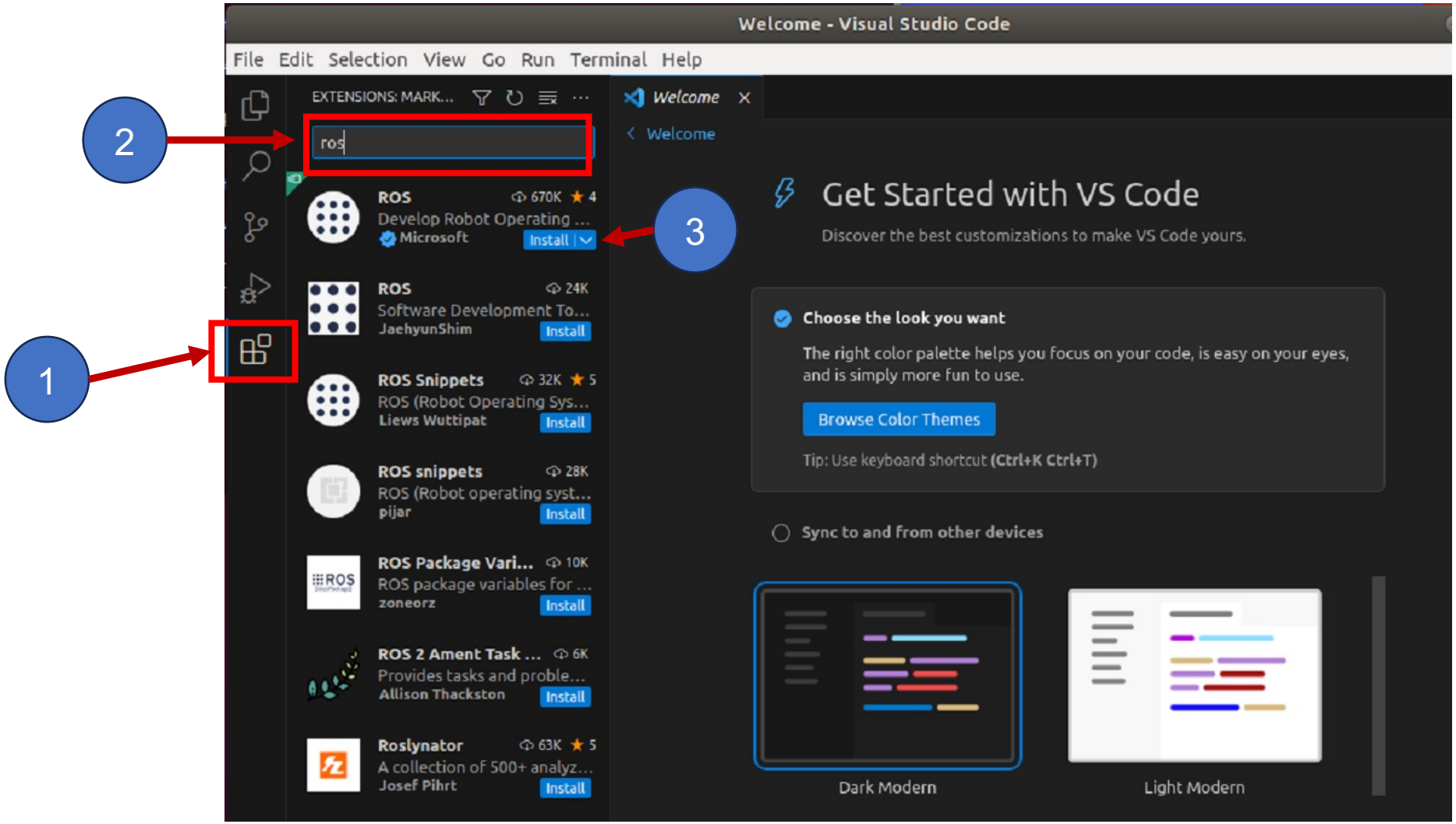
01. ROS 프로그램 개발환경 구축

VSCODE 설치



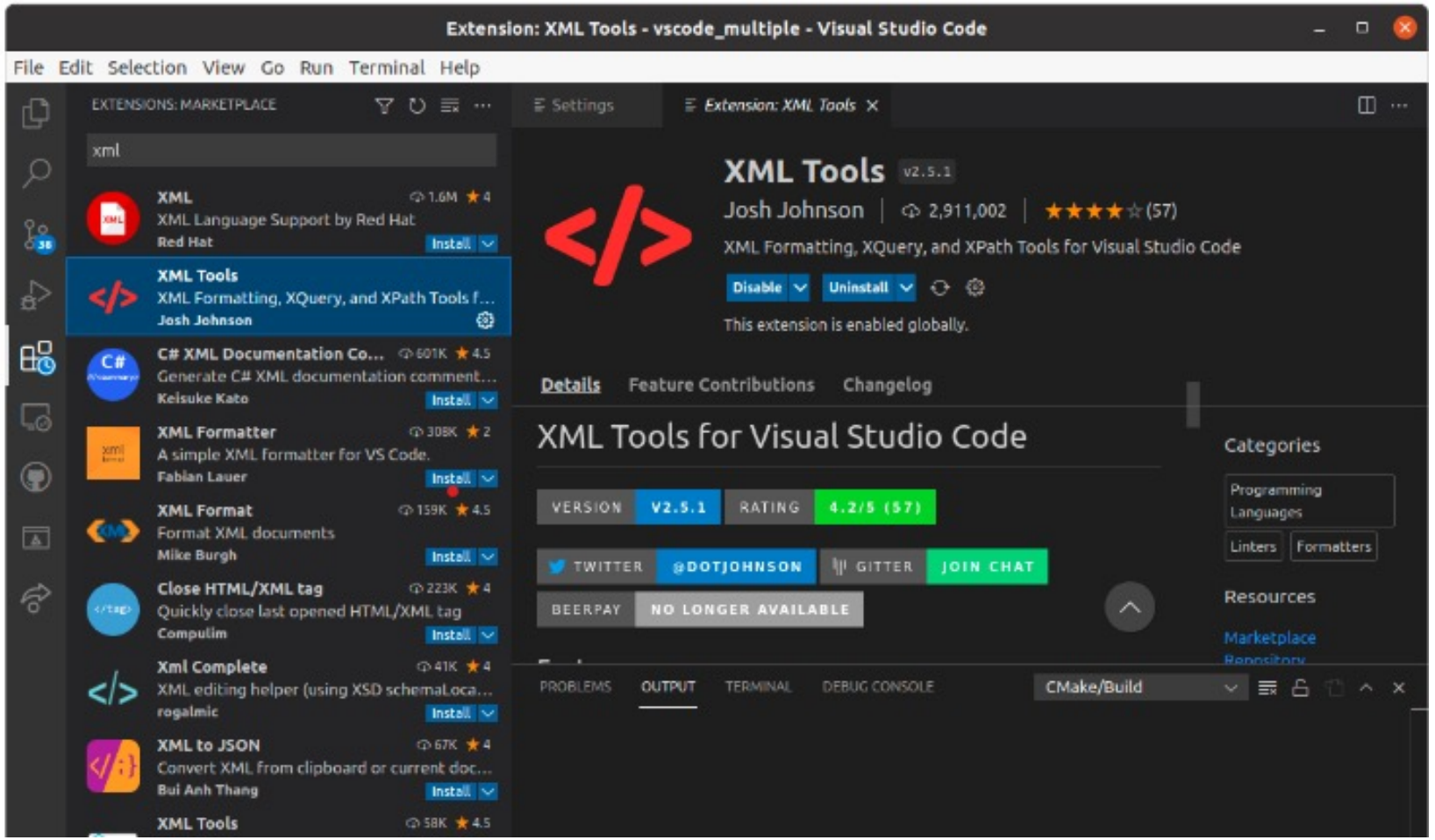
01. ROS 프로그램 개발환경 구축

VSCODE > ROS Extension 설치



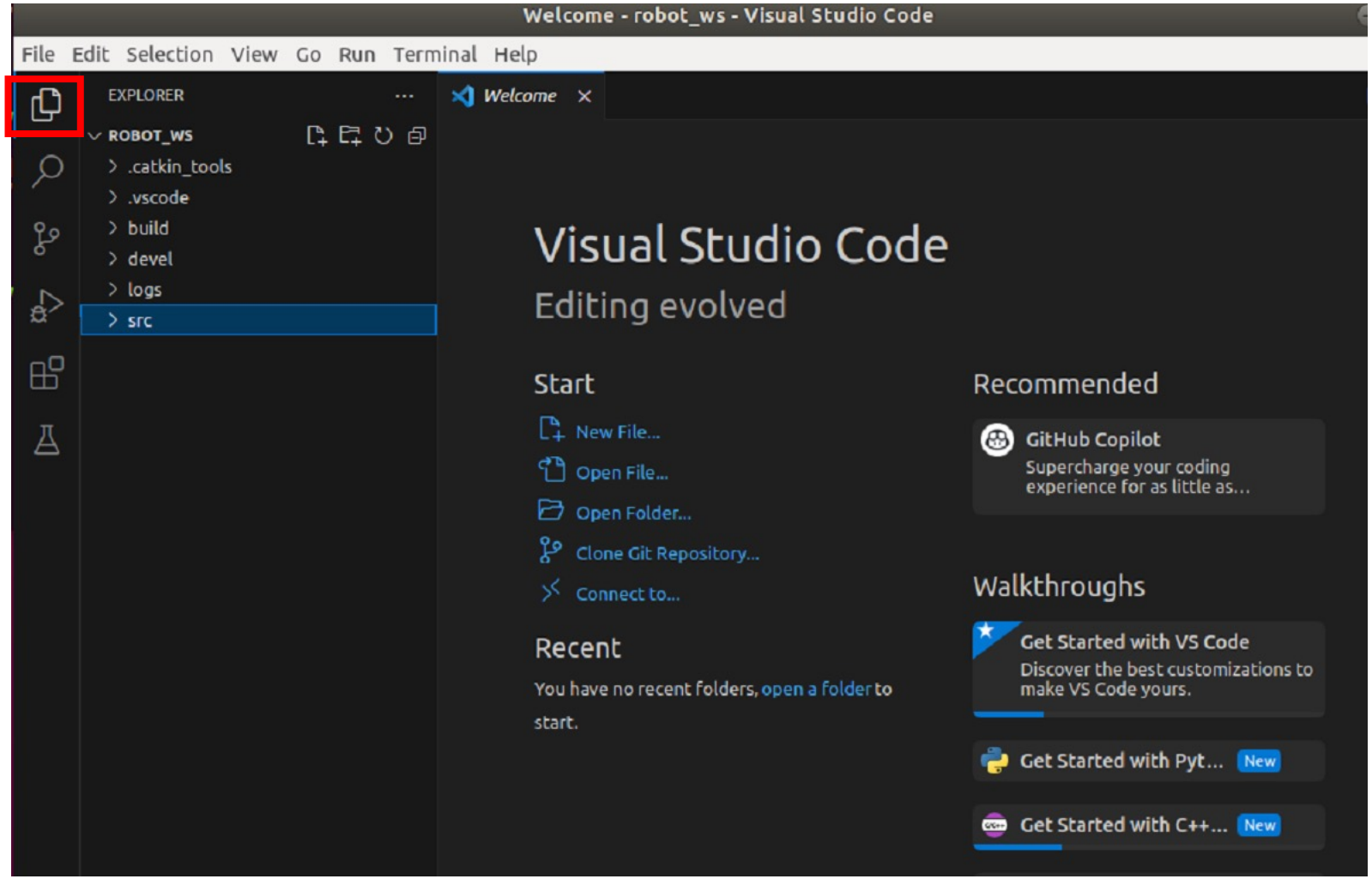
01. ROS 프로그램 개발환경 구축

VSCODE > ROS 추가 Extension 설치



01. ROS 프로그램 개발환경 구축

VSCODE > Open robot_ws 폴더



02. ROS 프로그래밍

패키지 생성

작업 폴더로 이동

```
$ cd ~/robot_ws/src
```

패키지 생성 : catkin_create_pkg [패키지 명] [의존하는 패키지1] [의존하는 패키지2]

```
$ catkin_create_pkg my_first_ros_pkg roscpp std_msgs
```

C++ 기반 패키지

```
$ catkin_create_pkg my_second_ros_pkg rospy std_msgs
```

Python 기반 패키지

```
$ catkin_create_pkg my_third_ros_pkg roscpp rospy std_msgs
```

C++, Python 복합패키지

02. ROS 프로그래밍

패키지 설정 파일 (package.xml) 수정

<name> 패키지 이름
<version> 패키지 버전
<decription> 패키지 설명
<maintainer> 패키지 관리자
<licence> 라이선스 규정
<url> 패키지 관련 URL 주소
<author> 작성자

<buildtool_depend> catkin 사용
<build_depend> build를 위한 의존성 패키지들
<exec_depend> 실행을 위한 의존성 패키지들

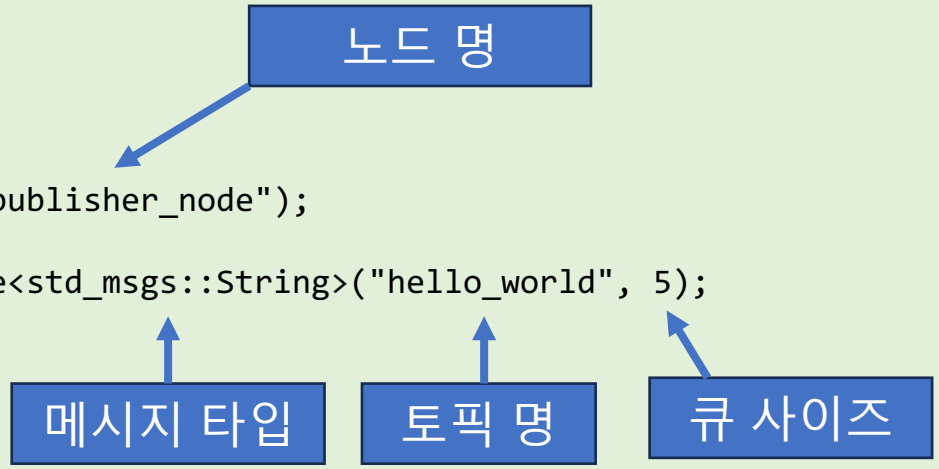
02. ROS 프로그래밍

소스 코드 작성 @ my_first_ros_pkg > src > hello_world_publisher_node.cpp

```
#include <ros/ros.h>
#include <std_msgs/String.h>
#include <sstream>

int main(int argc, char **argv)
{
    ros::init(argc, argv, "hello_world_publisher_node");
    ros::NodeHandle hNode;
    ros::Publisher pub = hNode.advertise<std_msgs::String>("hello_world", 5);

    ros::Rate loop_rate(10);
    int count = 0;
    while (ros::ok())
    {
        std_msgs::String msg;
        std::stringstream ss;
        ss << "hello world : " << count;
        msg.data = ss.str();
        ROS_INFO("%s", msg.data.c_str());
        pub.publish(msg);
        ros::spinOnce();
        loop_rate.sleep();
        count++;
    }
    return 0;
}
```



02. ROS 프로그래밍

빌드 설정 파일 (CMakeLists.txt) 수정 @ my_first_ros_pkg

```
…

add_executable(hello_world_publisher_node src/hello_world_publisher_node.cpp)
target_link_libraries(hello_world_publisher_node ${catkin_LIBRARIES})


…
```

02. ROS 프로그래밍

패키지 빌드

1. ROS workspace로 이동

```
$ cd ~/robot_ws
```

2. 빌드 : catkin_make

```
$ catkin_make
```

3. ROS 패키지 셋업

```
$ source ~/robot_ws/devel/setup.bash
```

02. ROS 프로그래밍

패키지 노드 실행

새로운 터미널(2) 에서 > Master 노드 실행

```
$ roscore
```

빌드한 터미널(1)에서 > hello_world_publisher_node 실행

```
$ rosrun my_first_ros_pkg hello_world_publisher_node
```

02. ROS 프로그래밍

패키지 노드 실행

새로운 터미널(3) 에서 > ROS 토픽 리스트 조회

```
$ rostopic list
```

새로운 터미널(3) 에서 > 토픽 출력

```
$ rostopic echo /hello_world
```

새로운 터미널(3) 에서 > 토픽 송출 빈도(Hz) 조회

```
$ rostopic hz /hello_world
```

05. ROS GUI Tools

패키지 셋업 .bashrc 에 등록

새로운 터미널 마다 수행되어야 my_first_ros_pkg 의 노드 실행이 가능함

```
$ source ~/robot_ws/devel/setup.bash
```

새로운 터미널 마다 my_first_ros_pkg 의 노드 실행이 가능하도록 .bashrc 에 추가함

```
$ echo source ~/robot_ws/devel/setup.bash >> ~/.bashrc
```

02. ROS 프로그래밍

소스 코드 작성 @ my_first_ros_pkg > src > hello_world_subscriber_node.cpp

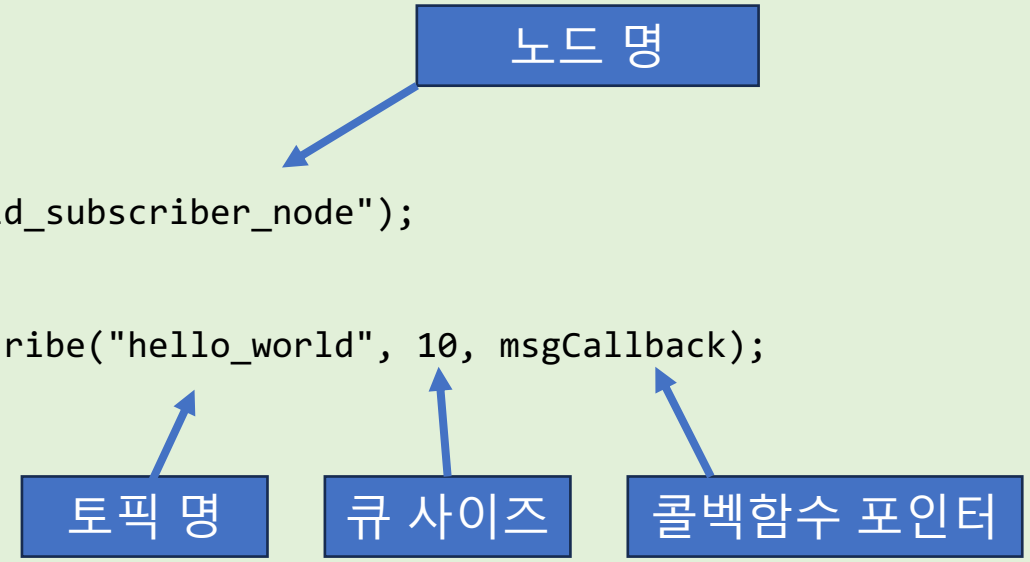
```
#include <ros/ros.h>
#include <std_msgs/String.h>

void msgCallback(const std_msgs::String::ConstPtr& msg)
{
    ROS_INFO("receive msg: %s", msg->data.c_str());
}


int main(int argc, char **argv)
{
    ros::init(argc, argv, "hello_world_subscriber_node");
    ros::NodeHandle hNode;

    ros::Subscriber sub = hNode.subscribe("hello_world", 10, msgCallback);

    ros::spin();
    return 0;
}
```



02. ROS 프로그래밍

빌드 설정 파일 (CMakeLists.txt) 수정 @ my_first_ros_pkg

```
…

add_executable(hello_world_publisher_node src/hello_world_publisher_node.cpp)
target_link_libraries(hello_world_publisher_node ${catkin_LIBRARIES})

add_executable(hello_world_subscriber_node src/hello_world_subscriber_node.cpp)
target_link_libraries(hello_world_subscriber_node  ${catkin_LIBRARIES})

…
```

02. ROS 프로그래밍

패키지 빌드

1. ROS workspace로 이동

```
$ cd ~/robot_ws
```

2. 빌드 : catkin_make

```
$ catkin_make
```

3. ROS 패키지 셋업

```
$ source ~/robot_ws/devel/setup.bash
```

02. ROS 프로그래밍

패키지 노드 실행

터미널1 에서 > Master 노드 실행

```
$ roscore
```

터미널 2에서 > hello_world_publisher_node 실행

```
$ rosrun my_first_ros_pkg hello_world_publisher_node
```

터미널 3에서 > hello_world_subscriber_node 실행

```
$ rosrun my_first_ros_pkg hello_world_scriber_node
```

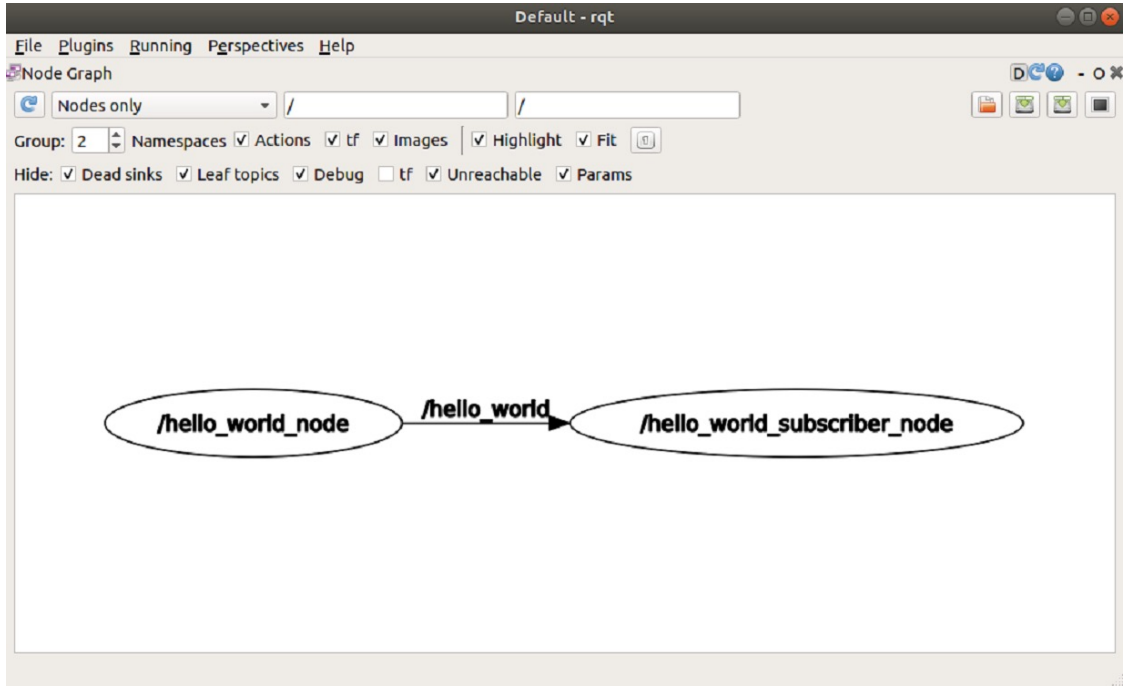
02. ROS 프로그래밍

rqt

터미널4에서 > rqt 실행

```
$ rqt
```

rqt > Plugins > Introspection > Node Graph



02. ROS 프로그래밍

소스 코드 작성 @ my_first_ros_pkg > msg > msgMyNumber.msg

새로운 메시지 타입 정의

```
int32 a
int32 b
```

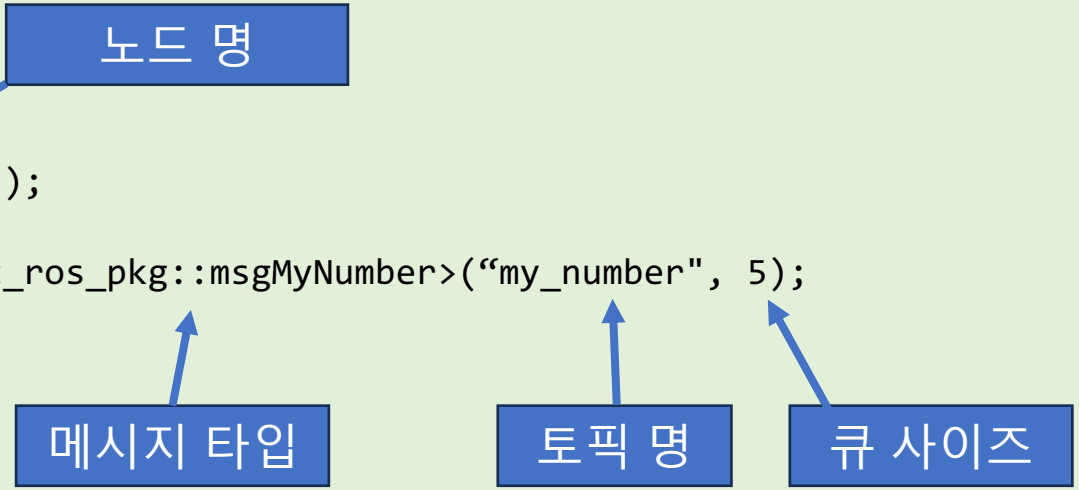
02. ROS 프로그래밍

소스 코드 작성 @ my_first_ros_pkg > src > number_publisher_node.cpp

```
#include <ros/ros.h>
#include “my_first_ros_pkg/msgMyNumber.h

int main(int argc, char **argv)
{
    ros::init(argc, argv, “number_publisher_node");
    ros::NodeHandle hNode;
    ros::Publisher pub = hNode.advertise<my_first_ros_pkg::msgMyNumber>(“my_number", 5);

    ros::Rate loop_rate(1);
    int count = 0;
    while (ros::ok())
    {
        my_first_ros_pkg::msgMyNumber msg;
        msg.data.a = 1;
        msg.data.b = count;
        ROS_INFO("%d, %d", msg.data.a, msg.dat.b);
        pub.publish(msg);
        ros::spinOnce();
        loop_rate.sleep();
        count++;
    }
    return 0;
}
```



02. ROS 프로그래밍

패키지 설정 파일 (package.xml) 수정

```
<buildtool_depend>catkin</buildtool_depend>
<build_depend>roscpp</build_depend>
<build_depend>std_msgs</build_depend>
<build_depend>message_generation</build_depend>          추가

<build_export_depend>roscpp</build_export_depend>
<build_export_depend>std_msgs</build_export_depend>
<exec_depend>roscpp</exec_depend>
<exec_depend>std_msgs</exec_depend>
<exec_depend>message_runtime</exec_depend>               추가
```

02. ROS 프로그래밍

빌드 설정 파일 (CMakeLists.txt) 수정 @ my_first_ros_pkg

```
find_package(catkin REQUIRED COMPONENTS
  roscpp
  std_msgs
  message_generation    추가
)

add_message_files(
  FILES
  msgMyNumber.msg
)

generate_messages(
   DEPENDENCIES
   std_msgs
)

catkin_package(
  #INCLUDE_DIRS include
  LIBRARIES my_first_ros_pkg
  CATKIN_DEPENDS roscpp std_msgs
  DEPENDS system_lib
)

add_executable(number_publisher_node src/number_publisher_node.cpp)
target_link_libraries(number_publisher_node  ${catkin_LIBRARIES})
```

02. ROS 프로그래밍

패키지 빌드

1. ROS workspace로 이동

```
$ cd ~/robot_ws
```

2. 빌드 : catkin_make

```
$ catkin_make
```

3. ROS 패키지 셋업

```
$ source ~/robot_ws/devel/setup.bash
```

02. ROS 프로그래밍

패키지 노드 실행

터미널 1에서 > Master 노드 실행

```
$ roscore
```

터미널 2에서 > number_publisher_node 실행

```
$ rosrun my_first_ros_pkg number_publisher_node
```

02. ROS 프로그래밍

소스 코드 작성 @ my_first_ros_pkg > src > number_subscriber_node.cpp

```
#include <ros/ros.h>
#include "my_first_ros_pkg/msgMyNumber.h"

void msgCallback(const my_first_ros_pkg::msgMyNumber::ConstPtr& msg)
{
    ROS_INFO("receive msg: %d, %d", msg->a, msg->b);
}


int main(int argc, char **argv)
{
    ros::init(argc, argv, "number_subscriber_node");
    ros::NodeHandle hNode;
    ros::Subscriber sub = hNode.subscribe("my_number", 10, msgCallback);

    ros::spin();
    return 0;
}
```

02. ROS 프로그래밍

빌드 설정 파일 (CMakeLists.txt) 수정 @ my_first_ros_pkg

```
…

add_executable(number_subscriber_node src/hello_world_subscriber_node.cpp)
target_link_libraries(number_subscriber_node  ${catkin_LIBRARIES})

…
```

02. ROS 프로그래밍

패키지 노드 실행

터미널 1에서 > Master 노드 실행

```
$ roscore
```

터미널 2에서 > number_publisher_node 실행

```
$ rosrun my_first_ros_pkg number_publisher_node
```

터미널 3에서 > number_subscriber_node 실행

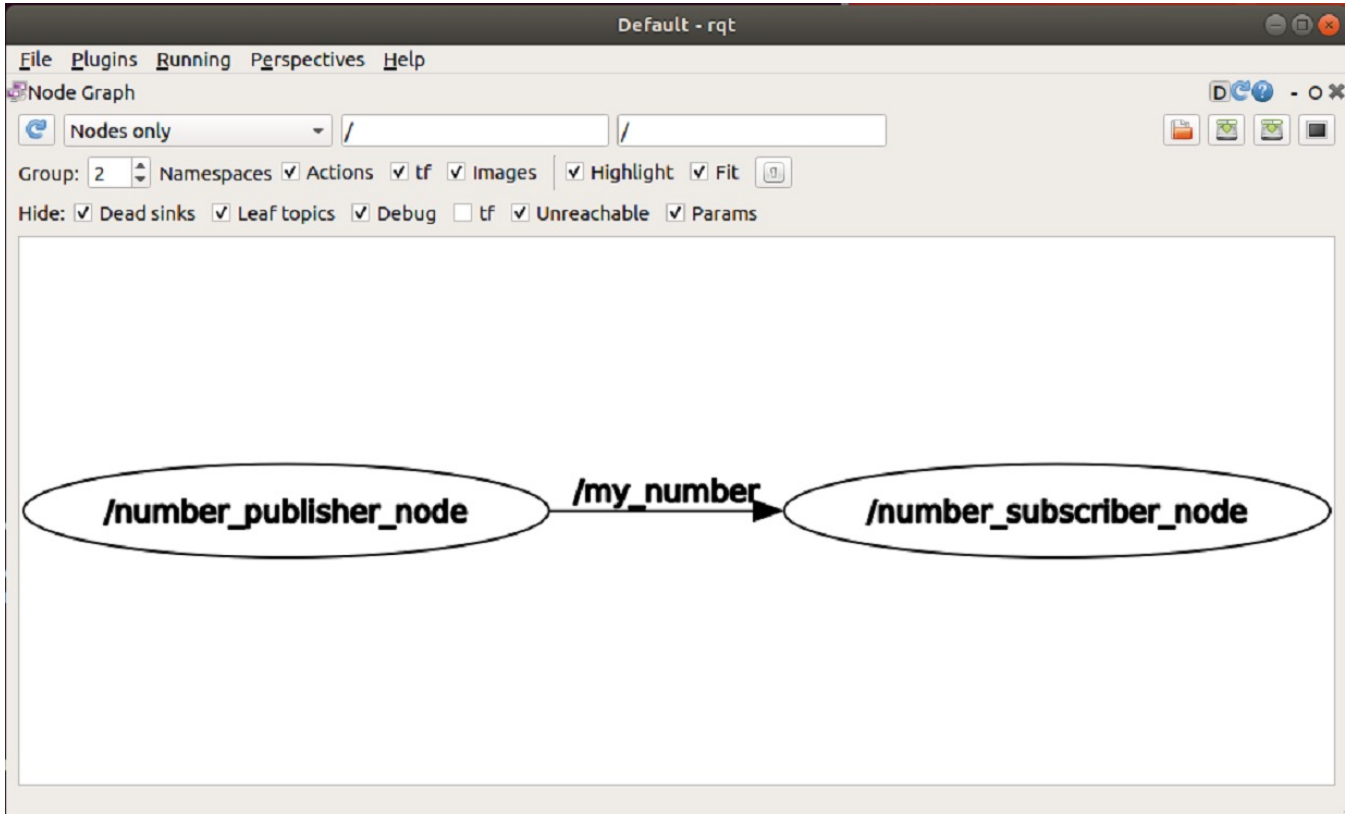
```
$ rosrun my_first_ros_pkg number_subscriber_node
```

02. ROS 프로그래밍

rqt

터미널4에서 > rqt 실행

```
$ rqt
```



02. ROS 프로그래밍

소스 코드 작성 @ my_second_ros_pkg > scripts > talker.py

```
#!/usr/bin/env python

import rospy
from std_msgs.msg import String

def talker():
    rospy.init_node('talker', anonymous=True)
    pub = rospy.Publisher('say', String, queue_size=10)

    rate = rospy.Rate(10) # 10hz
    while not rospy.is_shutdown():
        hello_str = "hello world %s" % rospy.get_time()
        rospy.loginfo(hello_str)
        pub.publish(hello_str)
        rate.sleep()

if __name__ == '__main__':
    try:
        talker()
    except rospy.ROSInterruptException:
        pass
```

02. ROS 프로그래밍

소스 코드 작성 @ my_second_ros_pkg > scripts > listener.py

```
#!/usr/bin/env python

import rospy
from std_msgs.msg import String

def callback(msg):
    rospy.loginfo('I heard %s', msg.data)

def listener():
    rospy.init_node('listener', anonymous=True)
    rospy.Subscriber('say', String, callback)
    rospy.spin()

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

02. ROS 프로그래밍

파이썬 코드 권한 수정 @ my_second_ros_pkg / scripts

Scripts 폴더로 이동

```
$ cd ~/robot_ws/src/my_second_ros_pkg/scripts
```

권한 수정

```
$ chmod +x talker.py listener.py
```

02. ROS 프로그래밍

패키지 노드 실행

터미널 1에서 > Master 노드 실행

```
$ roscore
```

터미널 2에서 > talker 노드 실행

```
$ rosrun my_second_ros_pkg talker.py
```

터미널 3에서 > listener 노드 실행

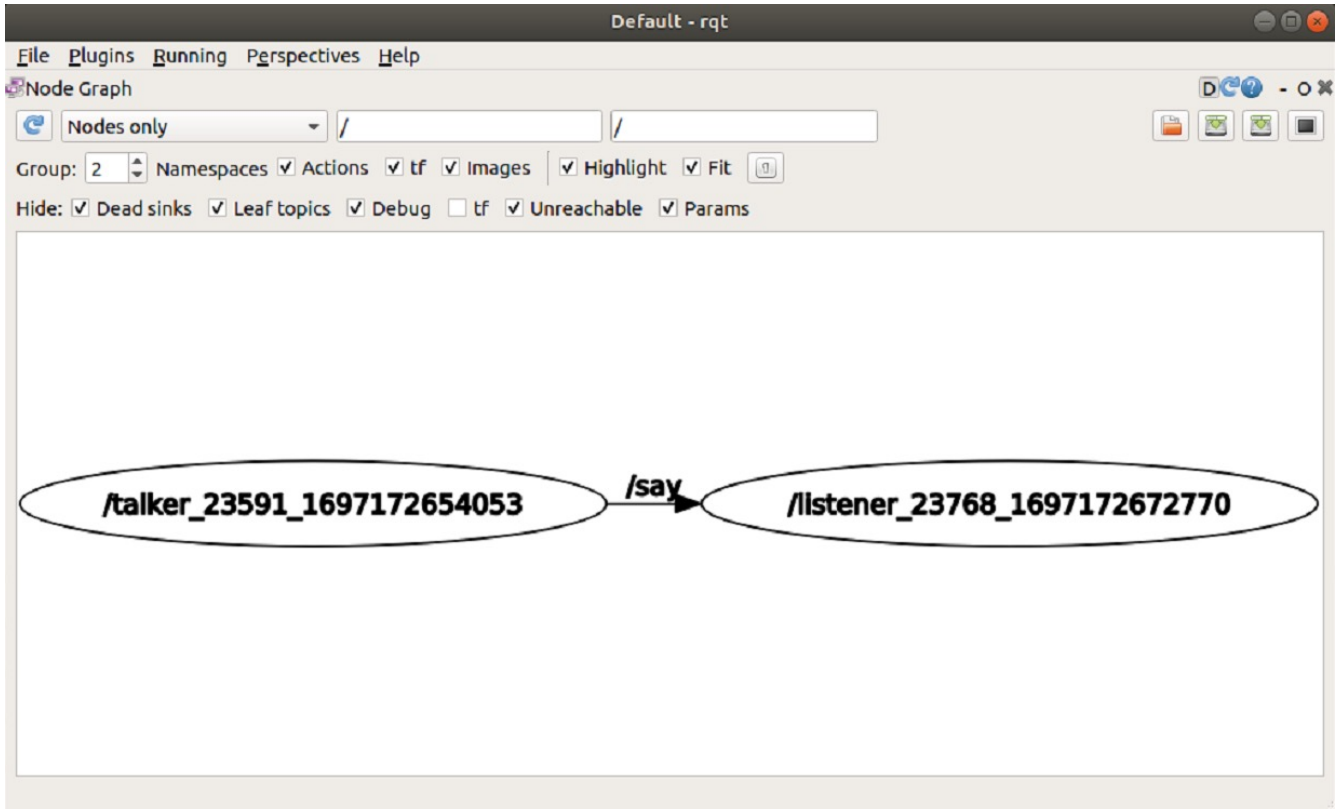
```
$ rosrun my_second_ros_pkg listener.py
```

02. ROS 프로그래밍

rqt

터미널4에서 > rqt 실행

```
$ rqt
```



02. ROS 프로그래밍

ROS에서 Python3 사용

파이썬3 ROS 패키지 추가 설치

```
$ sudo apt install python3-pip python3-all-dev python3-rospkg
$ sudo apt install ros-melodic-desktop-full --fix-missing
```

python 2 사용시 파이썬 파일 헤더에 삽입

```
#!/usr/bin/env python
```

python 3 사용시 파이썬 파일 헤더에 삽입

```
#!/usr/bin/env python3
```

02. ROS 프로그래밍

소스 코드 작성 @ my_second_ros_pkg > scripts > number_publisher.py

```
#!/usr/bin/env python3
import rospy
from my_first_ros_pkg.msg import msgMyNumber

def number_publisher():
    rospy.init_node('number_publisher', anonymous=True)
    pub = rospy.Publisher('my_number', msgMyNumber, queue_size=10)
    rate = rospy.Rate(10) # 10hz
    count = 0
    while not rospy.is_shutdown():
        msg_number = msgMyNumber()
        msg_number.a = 2
        msg_number.b = count
        count += 1
        print_str = "msg : a=%d, b=%d" % (msg_number.a, msg_number.b)
        rospy.loginfo(print_str)
        pub.publish(msg_number)
        rate.sleep()

if __name__ == '__main__':
    try:
        number_publisher()
    except rospy.ROSInterruptException:
        pass
```

02. ROS 프로그래밍

소스 코드 작성 @ my_second_ros_pkg > scripts > number_subscriber.py

```
#!/usr/bin/env python3

import rospy
from my_first_ros_pkg.msg import msgMyNumber

def callback(msg):
    rospy.loginfo('I heard a=%d, b=%d', msg.a, msg.b)

def number_subscriber():
    rospy.init_node('number_subscriber', anonymous=True)
    rospy.Subscriber('my_number', msgMyNumber, callback)
    rospy.spin()

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

02. ROS 프로그래밍

파이썬 코드 권한 수정 @ my_second_ros_pkg / scripts

Scripts 폴더로 이동

```
$ cd ~/robot_ws/src/my_second_ros_pkg/scripts
```

권한 수정

```
$ chmod +x number_publisher.py number_subscriber.py
```

02. ROS 프로그래밍

패키지 노드 실행

터미널 1에서 > Master 노드 실행

```
$ roscore
```

터미널 2에서 > talker 노드 실행

```
$ rosrun my_second_ros_pkg number_publisher.py
```

터미널 3에서 > listener 노드 실행

```
$ rosrun my_second_ros_pkg number_subscriber.py
```

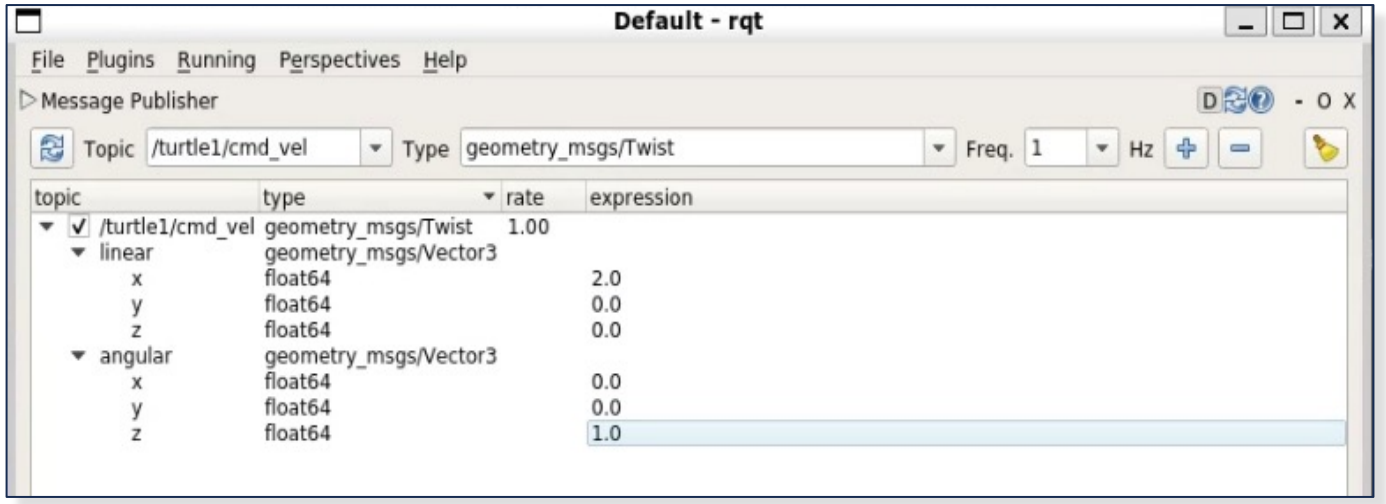
03. Quiz

1. turtlesim Turtle을 Python node로 움직여 보자

turtle_move.py. : Python으로 "/turtle1/cmd_vel" 토픽 메시지 발행(Publisher) node 를 작성해 보자!

Hint Python 에서 Twist 메시지 라이브러리 로드

```
from geometry_msgs.msg import Twist
```



03. Quiz

1. turtlesim Turtle을 Python node로 움직여 보자 > 동작 확인

```
$ cd ~/robot_ws/src/my_second_ros_pkg/scripts
```

권한 수정

```
$ chmod +x turtle_move.py
```

터미널 1에서 > Master 노드 실행

```
$ roscore
```

터미널 2에서 > turtlesim_node 실행

```
$ rosrun turtlesim turtlesim_node
```

터미널 3에서 > turtle_move.py 에 대한 turtle_move 노드 실행

```
$ rosrun my_second_ros_pkg turtle_move.py
```

03. Quiz

2. turtlesim Turtle을 Python node에서 키보드로 움직여 보자

turtlesim 패키지의 turtle_teleop_key 와 같은 python publisher node 를 작성해 보자!

Python3 키보드 입력 라이브러리 pynput 설치

```
$ pip3 install pynput
```

```
import time
from pynput import keyboard

def on_release(key):
    print('release:', key)

def on_press(key):
    print('press:', key)


keyboard_listener = keyboard.Listener(on_press=on_press, on_release=on_release)
keyboard_listener.start()

while True:
    time.sleep(1)
```

04. 웹캠 (Webcam) RGB 이미지 토픽

OpenCV Python 라이브러리 설치

Python3 OpenCV 라이브러리 설치

```
$ pip3 install opencv-python
```

에러 발생시 scikit-build 설치 후 OpenCV 라이브러리 설치

```
$ pip3 install --upgrade pip setuptools wheel
```

```
$ pip3 install scikit-build
```

04. 웹캠 (Webcam) RGB 이미지 토픽

OpenCV Python 웹캠 테스트 코드 작성 @ ~/robot_ws/my_second_ros_pkg/scripts/webcam.py

```
import cv2

cap = cv2.VideoCapture(0)

# cap.set(cv2.CAP_PROP_FRAME_WIDTH, 1280)
# cap.set(cv2.CAP_PROP_FRAME_HEIGHT, 720)

width = cap.get(cv2.CAP_PROP_FRAME_WIDTH)
height = cap.get(cv2.CAP_PROP_FRAME_HEIGHT)

print("Video Size :", width, height)


while cv2.waitKey(33) < 0:
    ret, frame = cap.read()
    if ret:
        cv2.imshow("VideoFrame", frame)

cap.release()
cv2.destroyAllWindows()
```

04. 웹캠 (Webcam) RGB 이미지 토픽

OpenCV Python 웹캠 토픽 발행 코드 작성 @ ~/robot_ws/my_second_ros_pkg/scripts/webcam_publisher.py

```
#!/usr/bin/env python3

import rospy
from sensor_msgs.msg import CompressedImage
from cv_bridge import CvBridge
import cv2

def webcam_publisher():
    rospy.init_node('webcam_publisher', anonymous=True)
    pub = rospy.Publisher('/webcam/color_image', CompressedImage, queue_size=10)
    cap = cv2.VideoCapture(0)

    bridge = CvBridge()
    rate = rospy.Rate(10)

    while not rospy.is_shutdown():
        ret, frame = cap.read()
        if ret:
            pub.publish(bridge.cv2_to_compressed_imgmsg(frame))
        rate.sleep()

if __name__ == '__main__':
    try:
        webcam_publisher()
    except rospy.ROSInterruptException:
        pass
```

04. 웹캠 (Webcam) RGB 이미지 토픽

OpenCV Python 웹캠 토픽 구독 코드 작성 @ ~/robot_ws/my_second_ros_pkg/scripts/webcam_subscriber.py

```
#!/usr/bin/env python3

import rospy
from sensor_msgs.msg import CompressedImage
from cv_bridge import CvBridge, CvBridgeError
import cv2

bridge = CvBridge()

def show_image(img):
    cv2.imshow("Webcam Image", img)
    cv2.waitKey(3)

def image_callback(img_msg):
    try:
        cv_image = bridge.compressed_imgmsg_to_cv2(img_msg)
        show_image(cv_image)
    except CvBridgeError as e:
        print(e)

rospy.init_node('webcam_subscriber', anonymous=True)
sub = rospy.Subscriber('/webcam/color_image', CompressedImage, image_callback)
cv2.namedWindow("Webcam Image", 1)

while not rospy.is_shutdown():
    rospy.spin())
```


Infinyx
THINK LIFE SYNC AI