ROS 2講習 第二回

-トピック通信のプログラム-

4S altair

参考文献:ROS2とPythonで作って学ぶAIロボット入門

環境

- python 3.10
- ThinkPad L380 ubuntu22.04.3tls
- ROS2 humble

トピック通信プログラムの作り方

パブリッシャを使ったクラスの一般的な書き方

```
class パブリッシャのクラスの名前(Node):
    def __init__(self):  # コンストラクタ
        super().__init__('ノード名')
        self.pub = self.create_publisher(メッセージ型, トピック名, 通信品質)
        # 1.パブリッシャの生成
        # メッセージ型：トピック通信に使うメッセージの型
        # トピック名：メッセージを送るトピック
        # 通信品質：通信品質に影響を及ぼすバッファの数
        self.timer = self.create_timer(タイマの間隔[s], コールバック)
        # 2.タイマーの生成
        # 一定周期でパブリッシュする必要がなければタイマ不要
        # その他の必要な処理を書く

    def timer_callback(self):  # コールバック関数
        #メッセージに値を代入するなど
        #3.パブリッシュ
        self.pub.publish(メッセージ)
        #  メッセージをパブリッシュする
        # その他必要な処理を書く
```

注意点

Pythonを使うときは型をあまり意識しないが，ROS2の通信ではメッセージ型を間違えると通信ができなくなる．

標準メッセージ型

bool,Char,Float32,Float64,Int8,Int16,Int32,Int64,String,MultiArray

通信品質のバッファのデフォルトは１０

では新しくパッケージを作りましょう

```
cd ~/hazimete/src
ros2 pkg create --build-type ament_python --node-name topic_publisher_node topic_pkg
```

topic_publisher_node.py

```
import rclpy                                # ROS2のPythonモジュール
from rclpy.node import Node                 # rclpy.nodeモジュールからNodeクラスをインポート
from std_msgs.msg import String             # std_msgs.msgモジュールからStringクラスをインポート


class hazimetePublisher(Node):  # "Happy World"とパブリッシュ並びに表示するクラス
    def __init__(self):  # コンストラクタ
        super().__init__('topic_publisher_node')
        self.pub = self.create_publisher(String, 'topic', 10)   # パブリッシャの生成
        self.timer = self.create_timer(1, self.timer_callback)  # タイマーの生成
        self.i = 10

    def timer_callback(self):  # コールバック関数
        msg = String()
        if self.i > 0:
            msg.data = f'アルタイル{self.i}'
        else:
            msg.data = f"終わり"
            self.destroy_timer(self.timer)
        self.pub.publish(msg)
        self.get_logger().info(f'パブリッシュ: {msg.data}')
        self.i -= 1
```

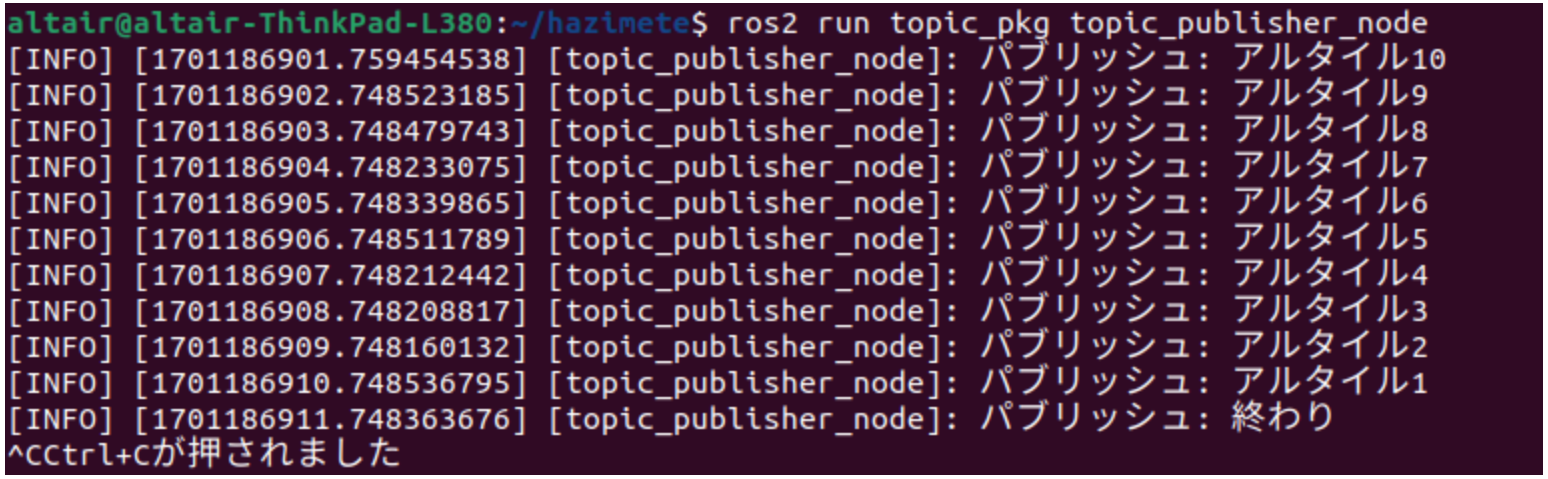
mein()関数は前とほとんど同じ

- インポート(３行目): `create_publisher()` はパブリッシャを生成．
- コンストラクト(7~11行目):文字列は `f` 文字列．self.iはカウント用

ログの重要度
DEBUG,INFO,WARN,ERROR,FATALとレベルが高くなる
標準設定の場合はINFO以上が表示される．ここではINFOを使っているのでログが端末に表示される．GUIツールrpt_consoleでも読める

実行

```
~/hazimete$ colcon build
~/hazimete$ source ~/hazimete/install/setup.bash
~/hazimete$ source /opt/ros/humble/setup.bash
~/hazimete$ ros2 run topic_pkg topic_publisher_node
```



トピック通信プログラムの作り方

サブスクライバを使ったクラスの書き方

```
class サブスクライバのクラス名(Node):
    def __init__(self):  # コンストラクタ
        super().__init__('ノード名')
        # 1.サブスクライバの生成
        self.sub = self.create_subscription(メッセージ型,
                                            トピック名, コールバック, 通信品質)
        # メッセージ型：トピック通信に使うメッセージの型
        # トピック名：メッセージを送るトピック
        # コールバック:サブスクライバが使うコールバック
        # 通信品質：通信品質に影響を及ぼすバッファの数
   # 2.コールバックの定義
   # 新しい通信が届くとrclpy.spin()がコールバック呼び出し
   # メッセージを引数として取得可能
    def callback(self, msg):  # コールバック関数
       # msgを使った処理
```

topic_subscriber_node.py

```
import rclpy                                # ROS2のPythonモジュール
from rclpy.node import Node                 # rclpy.nodeモジュールからNodeクラスをインポート
from std_msgs.msg import String             # std_msgs.msgモジュールからStringクラスをインポート


# Sring型メッセージをサブスクライブして端末に表示するだけの簡単なクラス
class hazimeteSubscriber(Node):
    def __init__(self):  # コンストラクタ
        super().__init__('topic_subscriber_node')
        # サブスクライバの生成
        self.sub = self.create_subscription(String,
                                            'topic', self.callback, 10)

    def callback(self, msg):  # コールバック関数
        self.get_logger().info(f'サブスクライブ: {msg.data}')
```

package.xmlの編集

topic_subscriber_node.pyでインポートしたモジュールを追加

```
                -
                -
                -
  <license>TODO: License declaration</license>

  <exec_depend>rclpy</exec_depend>
  <exec_depend>std_msgs</exec_depend>
  <exec_depend>geometry_msgs</exec_depend>

  <test_depend>ament_copyright</test_depend>
  <test_depend>ament_flake8</test_depend>
                -
                -
                -
```

setup.py

```
    entry_points={
        'console_scripts': [
            'topic_publisher_node = topic_pkg.topic_publisher_node:main',
            'topic_subscriber_node =topic_pkg.topic_subscriber_node:main',
        ],
```

実行は端末をに分割して行う(先にサブスクライバを起動させておこう)

```
!!
  opt = self.warn_dash_deprecation(opt, section)
/home/altair/.local/lib/python3.10/site-packages/setuptools/_distutils/
cmd.py:66: SetuptoolsDeprecationWarning: setup.py install is deprecated
.
!!

        ********************************************************************
*****************
        Please avoid running ``setup.py`` directly.
        Instead, use pypa/build, pypa/installer, pypa/build or
        other standards-based tools.

        See https://blog.ganssle.io/articles/2021/10/setup-py-deprecate
d.html for details.
        ********************************************************************
*****************

!!
  self.initialize_options()
---
Finished <<< test1 [3.23s]

Summary: 3 packages finished [4.67s]
  3 packages had stderr output: hazimetepkg test1 topic_pkg
altair@altair-ThinkPad-L380:~/hazimete$ source ~/hazimete/install/setup
.bash
altair@altair-ThinkPad-L380:~/hazimete$ ros2 run topic_pkg topic_publis
her_node
[INFO] [1701189647.779953152] [topic_publisher_node]: パブリッシュ: ア
ルタイル10
[INFO] [1701189648.768561679] [topic_publisher_node]: パブリッシュ: ア
ルタイル9
[INFO] [1701189649.768708129] [topic_publisher_node]: パブリッシュ: ア
ルタイル8
[INFO] [1701189650.768707316] [topic_publisher_node]: パブリッシュ: ア
ルタイル7
```

```
altair@altair-ThinkPad-L380:~/hazimete$ source ~/hazimete/install/setup
.bash

altair@altair-ThinkPad-L380:~/hazimete$ ros2 run topic_pkg topic_subscr
iber_node
[INFO] [1701189647.780713430] [topic_subscriber_node]: サブスクライブ:
アルタイル10
[INFO] [1701189648.768821948] [topic_subscriber_node]: サブスクライブ:
アルタイル9
[INFO] [1701189649.768993587] [topic_subscriber_node]: サブスクライブ:
アルタイル8
[INFO] [1701189650.769002184] [topic_subscriber_node]: サブスクライブ:
アルタイル7
```

サブスクライブしてパブリッシュするプログラム

```
~/hazimete/src$ ros2 pkg create --build-type ament_python --node-name pub_sub_node pub_sub
~/hazimete/src$ cd ..
~/hazimete$
```

```
import sys
import rclpy
from rclpy.executors import ExternalShutdownException
from rclpy.node import Node
from std_msgs.msg import String, Int32
class hszimetePubSub(Node):
    def __init__(self):
        super().__init__('hazimete_pub_sub')
        self.pub = self.create_publisher(String, 'hmazimete_msg', 10)
        self.sub = self.create_subscription(Int32, 'number', self.callback, 10)
        self.happy_actions = ({
            1: '他の人へ親切な行動をします．',
            2: '他の人とつながる行動をします．',
            3: '健康になるために運動をします．',
            4: 'マインドフルネスをします．',
            5: '新しいことに挑戦します．',
            6: 'ゴールを決めて，まず１歩を踏み出します．',
            7: 'レジリエンス（回復力）をつけます．',
            8: '物事の良い面を見ます．',
            9: '人は皆違っていることを受け入れます．',
            10: '皆で協力して世界を良くします．'})
    def callback(self, sub_msg):
        pub_msg = String()
        self.get_logger().info(f'サブスクライブ:{sub_msg.data}')
        pub_msg.data = self.happy_actions[sub_msg.data % 10 + 1]
        self.pub.publish(pub_msg)
        self.get_logger().info(f'パブリッシュ:{pub_msg.data}')
def main():
    rclpy.init()
    node = hszimetePubSub()
    try:
        rclpy.spin(node)
    except KeyboardInterrupt:
        pass
    except ExternalShutdownException:
        sys.exit(1)
    finally:
        rclpy.try_shutdown()
```

- インポート(５行目):Int32,Stering型をstd_msgs.msgモジュールをインポート
- hszimetePubSubクラス(８~27行目):日にちのデータをサブスクライブしてそれに一致するハッピーアクションを実行する

`ros2 topic pub`コマンドでメッセージを送り，そのノードが処理してパブリッシュしたメッセージをros2 topic echoコマンドで表示させる．端末を３分割します

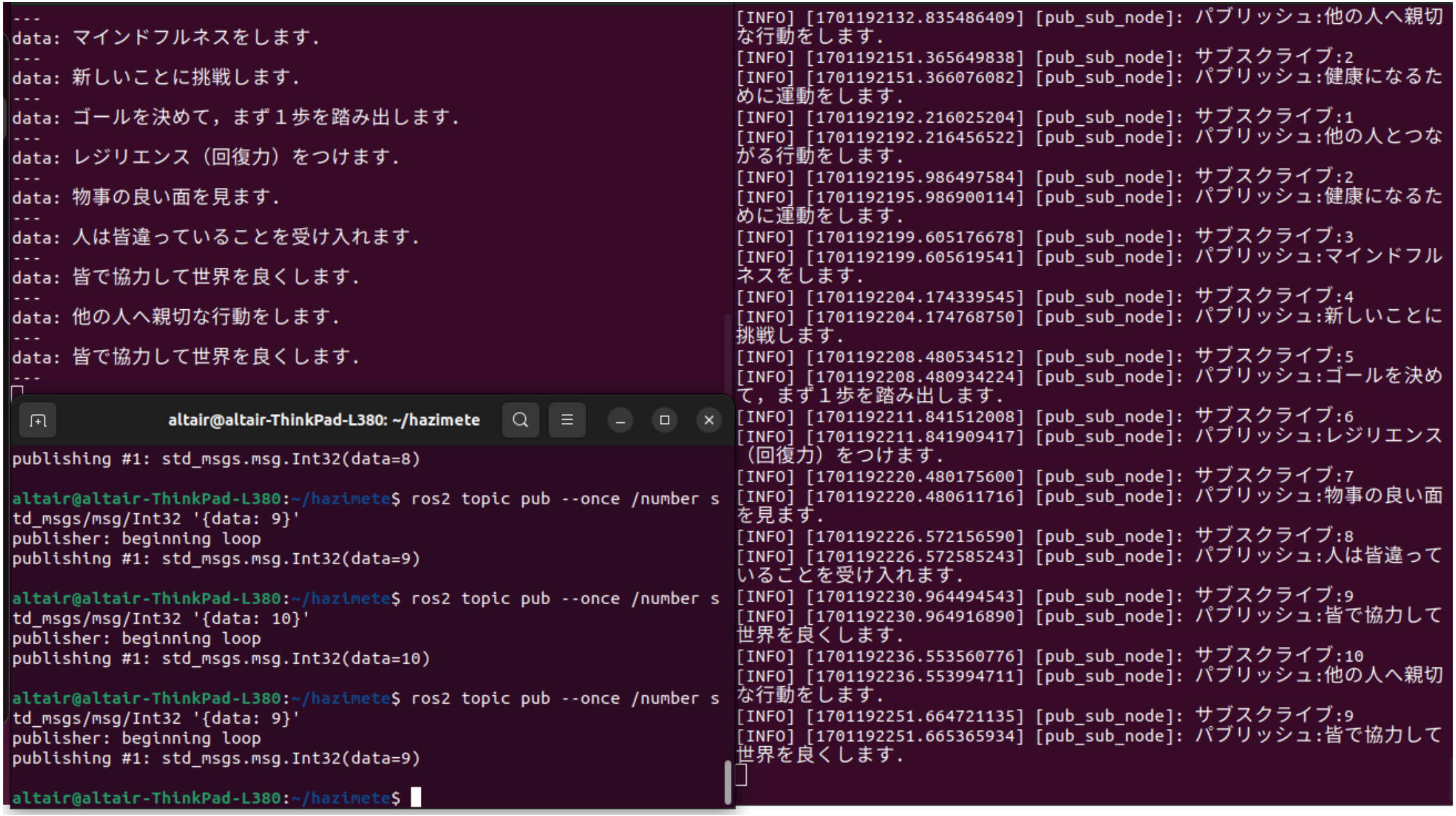
```
~/hazimete/src$ cd ..
~/hazimete$ colcon build
~/hazimete$ source ~/hazimete/install/setup.bash
```

ros2 topic listで利用可能トピックが見れる

```
ros2 topic echo /hmazimete_msg
```

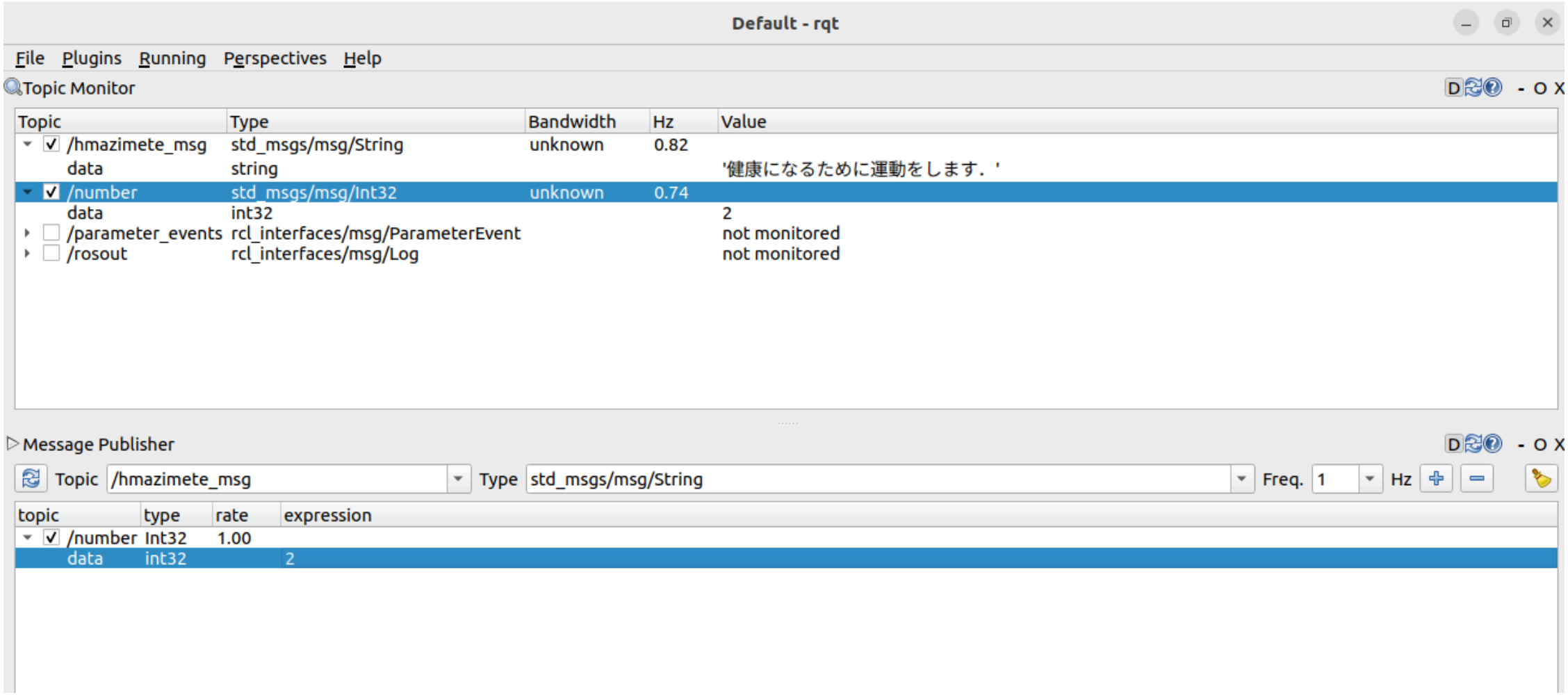
メッセージをプッシュします（--onceだから１回だけ）引数トピック名，メッセージ型，データ（辞書型）＊注：'{data: 1}'のdata:と1の間のスペース忘れずに．

```
ros2 topic pub --once /number std_msgs/msg/Int32 '{data: 1}'
```


---
data: マインドフルネスをします．
---
data: 新しいことに挑戦します．
---
data: ゴールを決めて，まず１歩を踏み出します．
---
data: レジリエンス（回復力）をつけます．
---
data: 物事の良い面を見ます．
---
data: 人は皆違っていることを受け入れます．
---
data: 皆で協力して世界を良くします．
---
data: 他の人へ親切な行動をします．
---
data: 皆で協力して世界を良くします．
---
altair@altair-ThinkPad-L380: ~/hazimete
publishing #1: std_msgs.msg.Int32(data=8)
altair@altair-ThinkPad-L380:~/hazimete$ ros2 topic pub --once /number s
td_msgs/msg/Int32 '{data: 9}'
publisher: beginning loop
publishing #1: std_msgs.msg.Int32(data=9)
altair@altair-ThinkPad-L380:~/hazimete$ ros2 topic pub --once /number s
td_msgs/msg/Int32 '{data: 10}'
publisher: beginning loop
publishing #1: std_msgs.msg.Int32(data=10)
altair@altair-ThinkPad-L380:~/hazimete$ ros2 topic pub --once /number s
td_msgs/msg/Int32 '{data: 9}'
publisher: beginning loop
publishing #1: std_msgs.msg.Int32(data=9)
altair@altair-ThinkPad-L380:~/hazimete$
[INFO] [1701192132.835486409] [pub_sub_node]: パブリッシュ:他の人へ親切な行動をします．
[INFO] [1701192151.365649838] [pub_sub_node]: サブスクライブ:2
[INFO] [1701192151.366076082] [pub_sub_node]: パブリッシュ:健康になるために運動をします．
[INFO] [1701192192.216025204] [pub_sub_node]: サブスクライブ:1
[INFO] [1701192192.216456522] [pub_sub_node]: パブリッシュ:他の人とつながる行動をします．
[INFO] [1701192195.986497584] [pub_sub_node]: サブスクライブ:2
[INFO] [1701192195.986900114] [pub_sub_node]: パブリッシュ:健康になるために運動をします．
[INFO] [1701192199.605176678] [pub_sub_node]: サブスクライブ:3
[INFO] [1701192199.605619541] [pub_sub_node]: パブリッシュ:マインドフルネスをします．
[INFO] [1701192204.174339545] [pub_sub_node]: サブスクライブ:4
[INFO] [1701192204.174768750] [pub_sub_node]: パブリッシュ:新しいことに挑戦します．
[INFO] [1701192208.480534512] [pub_sub_node]: サブスクライブ:5
[INFO] [1701192208.480934224] [pub_sub_node]: パブリッシュ:ゴールを決めて，まず１歩を踏み出します．
[INFO] [1701192211.841512008] [pub_sub_node]: サブスクライブ:6
[INFO] [1701192211.841909417] [pub_sub_node]: パブリッシュ:レジリエンス（回復力）をつけます．
[INFO] [1701192220.480175600] [pub_sub_node]: サブスクライブ:7
[INFO] [1701192220.480611716] [pub_sub_node]: パブリッシュ:物事の良い面を見ます．
[INFO] [1701192226.572156590] [pub_sub_node]: サブスクライブ:8
[INFO] [1701192226.572585243] [pub_sub_node]: パブリッシュ:人は皆違っていることを受け入れます．
[INFO] [1701192230.964494543] [pub_sub_node]: サブスクライブ:9
[INFO] [1701192230.964916890] [pub_sub_node]: パブリッシュ:皆で協力して世界を良くします．
[INFO] [1701192236.553560776] [pub_sub_node]: サブスクライブ:10
[INFO] [1701192236.553994711] [pub_sub_node]: パブリッシュ:他の人へ親切な行動をします．
[INFO] [1701192251.664721135] [pub_sub_node]: サブスクライブ:9
[INFO] [1701192251.665365934] [pub_sub_node]: パブリッシュ:皆で協力して世界を良くします．

GUIを使ってみる

```
rqt
```



次回サービス通信のプログラムの作り方