Subscriber创建流程

1. 设置环境和编码

```
#!/usr/bin/env python
#coding:utf-8
```

2. 创建节点

```
rospy.init_node(nodeName)
```

3. 创建订阅者

```
rospy.Subscriber(topicName, String, topicCallback)
rospy.spin()
```

4. 实现订阅回调

```
def topicCallback(msg):
    print msg
```

完整示例代码

```
#!/usr/bin/env python
# coding:utf-8

import rospy
from std_msgs.msg import String


def topicCallback(msg):
    print msg


if __name__ == '__main__':
    nodeName = "pysubscriber"
    topicName = "pytopic"

    # 初始化节点
    rospy.init_node(nodeName)

    # 创建订阅者
    rospy.Subscriber(topicName, String, topicCallback)

    # 阻塞线程
    rospy.spin()
```

调试订阅者

调试Subscriber主要是查看是否能收到数据，也就是提供一个发布的调试工具。ROS提供了命令行工具和图形化工具进行调试。

1. 通过自己编写的publisher进行调试

```
rosrun demo_topic pypublisher.py
```

2. 通过rostopic工具进行调试

查询主题所需要的数据类型

```
rostopic type pytopic
```

模拟发布数据

```
rostopic pub pytopic std_msgs/String hello -r 10
```

:::tip

`rostopic pub` 是模拟发布数据的命令

`cpptopic` 是将数据发送到那个主题，根据自己实际调试的主题来写。

`std_msgs/String` 是这个主题所需要的数据类型，我们是通过 `rostopic type cpptopic` 进行查询出来的。

`hello` 是发送的数据，根据自己的调试需求来写。

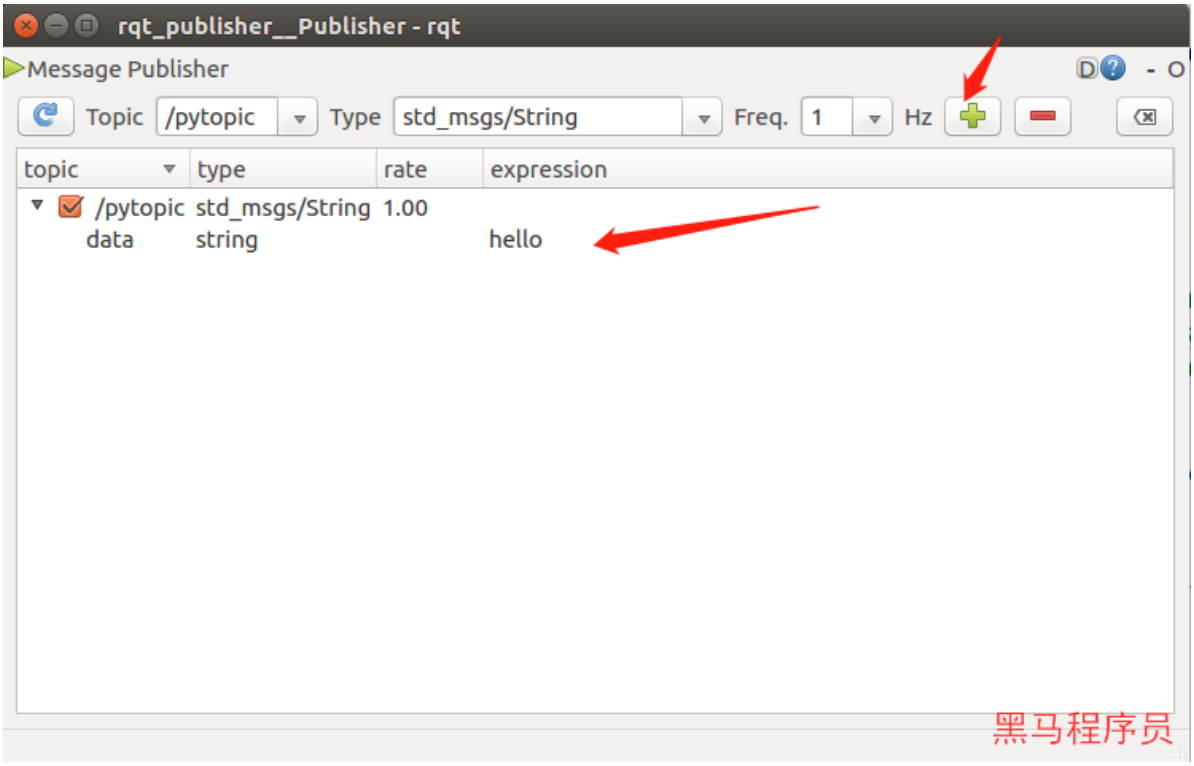
`-r` 指的是发送频率

:::

3. 通过rqt_publisher工具进行调试

通过命令启动rqt_publisher工具

```
rosrun rqt_publisher rqt_publisher
```


rqt_publisher__Publisher - rqt
Message Publisher
Topic
/pytopic
Type
std_msgs/String
Freq.
1
Hz
topic
type
rate
expression
/pytopic
std_msgs/String
1.00
data
string
hello
黑马程序员