

## Escrevendo um nodo que publica e um que se inscreve em um tópico

---

- **Criando o nodo listener (se inscreve em um tópico):**
  - Se inscreve no tópico “chatter”
  - Mostra mensagens recebidas

## Escrevendo um nodo que publica e um que se inscreve em um tópico

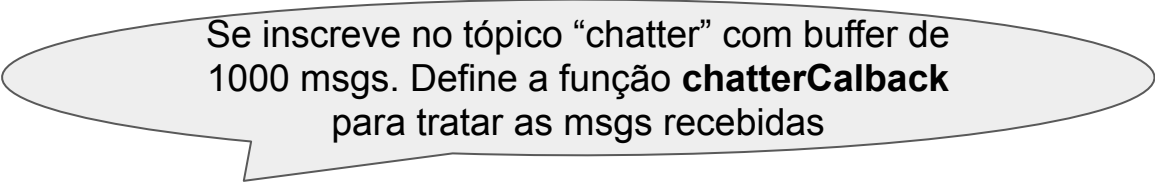
- Criando o nodo listener (se inscreve em um tópico): Arquivo: **listener.cpp**

```
#include "ros/ros.h"
#include "std_msgs/String.h"
void chatterCallback(const std_msgs::String::ConstPtr& msg)
{
    ROS_INFO("I heard: [%s]", msg->data.c_str());
}
int main(int argc, char **argv)
{
    ros::init(argc, argv, "listener");
    ros::NodeHandle n;
    ros::Subscriber sub = n.subscribe("chatter", 1000, chatterCallback);
    ros::spin();
    return 0;
}
```

## Escrevendo um nodo que publica e um que se inscreve em um tópico

- Criando o nodo listener (se inscreve em um tópico):

```
#include "ros/ros.h"
#include "std_msgs/String.h"
void chatterCallback(const std_msgs::String::ConstPtr& msg)
{
    ROS_INFO("I heard: [%s]", msg->data.c_str());
}
int main(int argc, char **argv)
{
    ros::init(argc, argv, "listener");
    ros::NodeHandle n;
    ros::Subscriber sub = n.subscribe("chatter", 1000, chatterCallback);
    ros::spin();
    return 0;
}
```



Se inscreve no tópico “chatter” com buffer de 1000 msgs. Define a função **chatterCallback** para tratar as msgs recebidas

## Escrevendo um nodo que publica e um que se inscreve em um tópico

- Criando o nodo listener (se inscreve em um tópico):

```
#include "ros/ros.h"
#include "std_msgs/String.h"
void chatterCallback(const std_msgs::String::ConstPtr& msg)
{
    ROS_INFO("I heard: [%s]", msg->data.c_str());
}
int main(int argc, char **argv)
{
    ros::init(argc, argv, "listener");
    ros::NodeHandle n;
    ros::Subscriber sub = n.subscribe("chatter", 1000, chatterCallback);
    ros::spin();
    return 0;
}
```

Processa todas as requisições internas do ROS eternamente (até ROS:ok() retorna false). Se essa função não for chamada, os callbacks não funcionam.

# Compilando os códigos

---

- Editando CMakeList.txt do pacote: (os comandos estão comentados no arquivo)
- Adiciona nodos talker e listener:
  - `add_executable(talker_node src/talker.cpp)`
  - `add_executable(listener_node src/listener.cpp)`
- Adiciona link de compilação com bibliotecas padrões para os 2 nodos:
  - `target_link_libraries(talker_node ${catkin_LIBRARIES})`
  - `target_link_libraries(listener_node ${catkin_LIBRARIES})`
- Compilando o pacote:
  - Na pasta raiz do workspace (catkin\_ws)
  - `$catkin_make`
- Executando os nodos (com roscore rodando):
  - `$roslaunch c3 talker_node`    `$roslaunch c3 listener_node`