自定义一个 msg
msg 的功能性单一
自定义的消息类型
文本文件类似于 c 的机构体
每一行是一个字段　定义变量的类型和字段 就是名字

```
{} c_cpp_properties.json    {} tasks.json    demo01_pub.
src > plumbing_pub_sub > msg > Person.msg
1  string name
2  int32 age
3  float32 height
```

1、在功能包下面
创建一个 msg 的文件夹
2、建立一个自定义名字的 msg 文件
msg 文件 类似于 c 的结构体

每一行要定义 类型 + 数据名称
3、在 package.xml 文件和 cmakelist 文件中加入 msg 文件
编译的时候的依赖 package.xml 文件

```
49  <!-- Use doc_depend for packages you need only for building docume
50  <!--   <doc_depend>doxygen</doc_depend> -->
51  <buildtool_depend>catkin</buildtool_depend>
52  <build_depend>roscpp</build_depend>
53  <build_depend>rospy</build_depend>
54  <build_depend>std_msgs</build_depend>
55  <build_depend>message_generation</build_depend>
56
57  <build_export_depend>roscpp</build_export_depend>
58  <build_export_depend>rospy</build_export_depend>
59  <build_export_depend>std_msgs</build_export_depend>
60
61  <exec_depend>roscpp</exec_depend>
62  <exec_depend>rospy</exec_depend>
63  <exec_depend>std_msgs</exec_depend>
64  <exec_depend>message_runtime</exec_depend>
65
66  <!-- The export tag contains other, unspecified, tags -->
```

在 cmakelist 的文件
1、首先是 find_package 我们要依赖我们的 message_generation
编译我们的功能包需要这个 message generation 区生存 message

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

2、添加我们的 message
这个是 51 行 把我们在 msg 文件夹 下面自定义的 Person.msg 文件添加到功能包
里面自带的 Message1.msg Message2.msg 我们不需要就舍弃

```
49
50  ## Generate messages in the 'msg' folder
51  add_message_files(
52    FILES
53    Person.msg
54  )
55
56  ## Generate services in the 'srv' folder
57  # add_service_files(
58  #   FILES
```

ctrl / 这个是? 和右 shift 那个位置的
在 cmakelist 的文件中多行的注释放开
3、generate 的 msg 我们需要依赖
我们编译的复合 复杂 msg 需要依赖基础的 msg 告诉我们编译的 msg 依赖什么

```
69
70  ## Generate added messages and service
71  generate_messages(
72    DEPENDENCIES
73    std_msgs
74  )
75
76  ######################################
77  ## Declare ROS dynamic reconfigure par
```

catkin_package 是 find_package 对应的
就是我们建立这个功能包 需要找到这些基础库
自定义的功能包 必须依赖 find package 的功能包
find package 依赖 catkin package 包
find package 编译时的依赖
catkin package 编译是依赖

```
## DEPENDS: system dependencies of this project that dependent projects also nee
catkin_package(
#  INCLUDE_DIRS include
#  LIBRARIES plumbing_pub_sub
  CATKIN_DEPENDS roscpp rospy std_msgs message_runtime
#  DEPENDS system_lib
)

###########
```

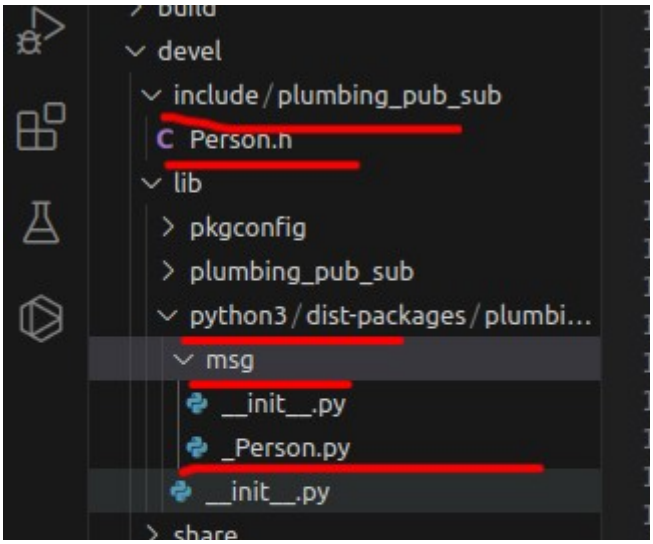
之后直接编译
中间文件在 devel 文件夹下面有个 person.h 的头文件

```
## Declare things to be passed to dependent projects
## INCLUDE_DIRS: uncomment this if your package contains header files
## LIBRARIES: libraries you create in this project that dependent projects also need
## CATKIN_DEPENDS: catkin_packages dependent projects also need
## DEPENDS: system dependencies of this project that dependent projects also need
catkin_package(
#  INCLUDE_DIRS include
```

问题 输出 调试控制台 终端 端口 注释

```
[ 72%] Built target plumbing_pub_sub_generate_messages_nodejs
[ 72%] Built target plumbing_pub_sub_generate_messages_cpp
[ 81%] Generating Python msg __init__.py for plumbing_pub_sub
[ 81%] Built target plumbing_pub_sub_generate_messages_py
[ 81%] Built target plumbing_pub_sub_generate_messages_eus
Scanning dependencies of target plumbing_pub_sub_generate_messages
[ 81%] Built target plumbing_pub_sub_generate_messages
[ 90%] Linking CXX executable /home/qinghuan/env_cv/demo04_ws/devel/lib/plumbing_pub_sub/demo01_pub
[ 90%] Built target demo01_pub
[100%] Linking CXX executable /home/qinghuan/env_cv/demo04_ws/devel/lib/plumbing_pub_sub/demo02_sub
[100%] Built target demo02_sub
* 终端将被任务重用，按任意键关闭。
```

我们直接调用 Person.h 就可以了
python 的文件在 lib 的 python3 的 msg 文件夹下面



c++使用 自定义的 msg 消息

vscode 的配置 如果不配置 可能会有异常

1、需要包含头文件

2、include 需要配置文件 配置 cpp include path 把自定义的 path 放进去与

文件夹 右键 选择集成 终端打开 之后 pwd 打印路径 之后 把这个路径添加到 include 的 path 中

注意英文逗号

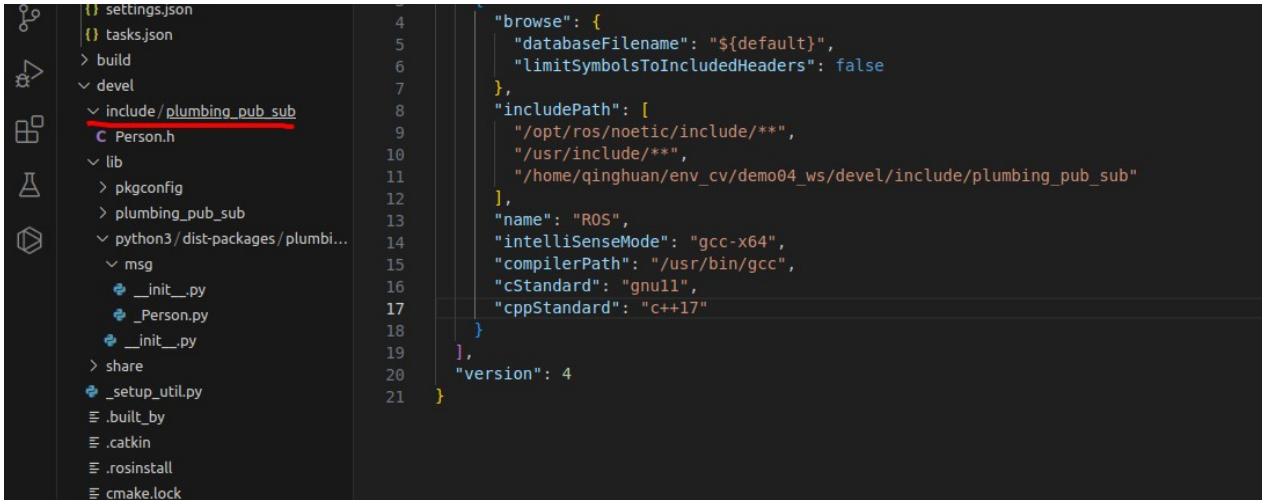
"/home/qinghuan/env_cv/demo04_ws/devel/include/plumbing_pub_sub"

这个效果是包含具体的这个文件家

"/home/qinghuan/env_cv/demo04_ws/devel/include/**"

这个样子** 是这个文件夹 include 下的所有头文件

配置这个的话 会有提示 而且不会报异常



add 添加依赖 因为 cpp 文件需要 msg 文件先编译出来文件 之后我们的 cpp 调用这个是来说明 msg 文件和 cpp 文件的依赖关系 msg 先编译出来
保证调用的依赖关系（避免 编写完 msg 和 cpp 统一编译的时候 这个时候 vscode 可能先编译 cpp 再 msg 文件 因此 这个是保证先编译了 msg 再编译 cpp

```
add_dependencies(demo03_pub ${PROJECT_NAME}_generate_messages_cpp)
```

ctrl shift +B

```
-- Using PYTHON_EXECUTABLE: /usr/bin/python3
-- Using Debian Python package layout
-- Using empy: /usr/lib/python3/dist-packages/em.py
-- Using CATKIN_ENABLE_TESTING: ON
-- Call enable_testing()
-- Using CATKIN_TEST_RESULTS_DIR: /home/qinghuan/env_cv/demo04_ws/build/test_result
-- Forcing gtest/gmock from source, though one was otherwise available.
-- Found gtest sources under '/usr/src/googletest': gtests will be built
-- Found gmock sources under '/usr/src/googletest': gmock will be built
-- Found PythonInterp: /usr/bin/python3 (found version "3.8.10")
-- Using Python nosetests: /usr/bin/nosetests3
-- catkin 0.8.10
-- BUILD_SHARED_LIBS is on
-- BUILD_SHARED_LIBS is on
-- ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
-- ~~  traversing 1 packages in topological order:
-- ~~  - plumbing_pub_sub
-- ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
-- +++ processing catkin package: 'plumbing_pub_sub'
-- ==> add_subdirectory(plumbing_pub_sub)
-- Using these message generators: gencpp;geneus;genlisp;gennodejs;genpy
-- plumbing_pub_sub: 1 messages, 0 services
-- Configuring done
-- Generating done
-- Build files have been written to: /home/qinghuan/env_cv/demo04_ws/build
####
#### Running command: "make -j12 -l12" in "/home/qinghuan/env_cv/demo04_ws/build"
####
[  0%] Built target std_msgs_generate_messages_py
[  0%] Built target std_msgs_generate_messages_nodejs
[ 30%] Built target demo01_pub
[ 30%] Built target demo02_sub
[ 30%] Built target std_msgs_generate_messages_eus
[ 30%] Built target std_msgs_generate_messages_lisp
[ 30%] Built target std_msgs_generate_messages_cpp
[ 30%] Built target _plumbing_pub_sub_generate_messages_check_deps_Person
[ 38%] Built target plumbing_pub_sub_generate_messages_cpp
[ 46%] Built target plumbing_pub_sub_generate_messages_lisp
[ 61%] Built target plumbing_pub_sub_generate_messages_py
[ 76%] Built target plumbing_pub_sub_generate_messages_eus
[ 84%] Built target plumbing_pub_sub_generate_messages_nodejs
[ 84%] Built target plumbing_pub_sub_generate_messages
[100%] Built target demo03_pub
 * 终端将被任务重用，按任意键关闭。
```

编译成功

之后注意 因为用自定义 msg 文件 确定 pub 发布的消息成功发送
用 rostopic echo newmsg
newmsg 是 自定义的话题
要进入我们自己 demo03 的空间
用 source 刷一下环境变量
source ./devel/setup.bash

```
问题    输出    调试控制台    终端    端口    注释

qinghuan@qinghuan-System-Product-Name:~/env_cv/demo04_ws$ rostopic echo newmsg
ERROR: Cannot load message class for [plumbing_pub_sub/Person]. Are your messages built
qinghuan@qinghuan-System-Product-Name:~/env_cv/demo04_ws$ cd demo0e_ws/
bash: cd: demo0e_ws/: 没有那个文件或目录
qinghuan@qinghuan-System-Product-Name:~/env_cv/demo04_ws$ source ./devel/setup.bash
qinghuan@qinghuan-System-Product-Name:~/env_cv/demo04_ws$ rostopic echo newmsg
name: "\u5F20\u4E09"
age: 129
height: 1.850000023841858
---
name: "\u5F20\u4E09"
age: 130
height: 1.850000023841858
---
name: "\u5F20\u4E09"
age: 131
height: 1.850000023841858
---
name: "\u5F20\u4E09"
age: 132
height: 1.850000023841858
---
name: "\u5F20\u4E09"
age: 133
height: 1.850000023841858
---
name: "\u5F20\u4E09"
age: 134
height: 1.850000023841858
---
name: "\u5F20\u4E09"
age: 135
height: 1.850000023841858
---
name: "\u5F20\u4E09"
age: 136
height: 1.850000023841858
---
name: "\u5F20\u4E09"
age: 137
height: 1.850000023841858
```

订阅的实现 具体看代码 这个是陪葬 cmakelist 文件
添加 3 行
add_executable
add_dependencies 添加依赖保证 msg 在 cpp 文件之前编译 ok
target_link_libraries 添加执行的的文件

```
add_executable(demo01_pub src/demo01_pub.cpp)

add_executable(demo02_sub src/demo02_sub.cpp)
add_executable(demo03_pub src/demo03_pub.cpp)
add_executable(demo04_sub src/demo04_sub.cpp)
## Rename C++ executable without prefix
## The above recommended prefix causes long target names, the following renam
## target back to the shorter version for ease of user use
## e.g. "rosrun someones_pkg node" instead of "rosrun someones_pkg someones_p
# set_target_properties(${PROJECT_NAME}_node PROPERTIES OUTPUT_NAME node PREF

## Add cmake target dependencies of the executable
## same as for the library above
add_dependencies(demo03_pub  ${PROJECT_NAME}_generate_messages_cpp)
add_dependencies(demo04_sub  ${PROJECT_NAME}_generate_messages_cpp)
## Specify libraries to link a library or executable target against
target_link_libraries(demo01_pub
  ${catkin_LIBRARIES}
)

target_link_libraries(demo02_sub
  ${catkin_LIBRARIES}
)
target_link_libraries(demo03_pub
  ${catkin_LIBRARIES}
)
target_link_libraries(demo04_sub
  ${catkin_LIBRARIES}
)
```

发布和订阅实现

订阅图片

```
问题    输出    调试控制台    终端    端口    注释

qinghuan@qinghuan-System-Product-Name:~/env_cv/demo04_ws$ source ./devel/setup.bash
qinghuan@qinghuan-System-Product-Name:~/env_cv/demo04_ws$ rosrun plumbing_pub_sub demo04_sub
[ INFO] [1709968971.100280922]: this is a subscriber
[ INFO] [1709969000.874744299]: 订阅的信息:  人的名字: 张三 , 人的年龄:  25, 人的身高: 1.85
[ INFO] [1709969000.875638072]: 订阅的信息:  人的名字: 张三 , 人的年龄:  26, 人的身高: 1.85
[ INFO] [1709969001.874848234]: 订阅的信息:  人的名字: 张三 , 人的年龄:  27, 人的身高: 1.85
[ INFO] [1709969002.874953454]: 订阅的信息:  人的名字: 张三 , 人的年龄:  28, 人的身高: 1.85
[ INFO] [1709969003.874952360]: 订阅的信息:  人的名字: 张三 , 人的年龄:  29, 人的身高: 1.85
[ INFO] [1709969004.874949282]: 订阅的信息:  人的名字: 张三 , 人的年龄:  30, 人的身高: 1.85
[ INFO] [1709969005.875135884]: 订阅的信息:  人的名字: 张三 , 人的年龄:  31, 人的身高: 1.85
[ INFO] [1709969006.874791613]: 订阅的信息:  人的名字: 张三 , 人的年龄:  32, 人的身高: 1.85
^Z
[1]+  已停止               rosrun plumbing_pub_sub demo04_sub
qinghuan@qinghuan-System-Product-Name:~/env_cv/demo04_ws$
```

发布的图片

```
问题    输出    调试控制台    终端    端口    注释

qinghuan@qinghuan-System-Product-Name:~/env_cv/demo04_ws$ source ./devel/setup.bash
qinghuan@qinghuan-System-Product-Name:~/env_cv/demo04_ws$ rosrun plumbing_pub_sub demo03_pub
^Z
[1]+  已停止               rosrun plumbing_pub_sub demo03_pub
qinghuan@qinghuan-System-Product-Name:~/env_cv/demo04_ws$
```