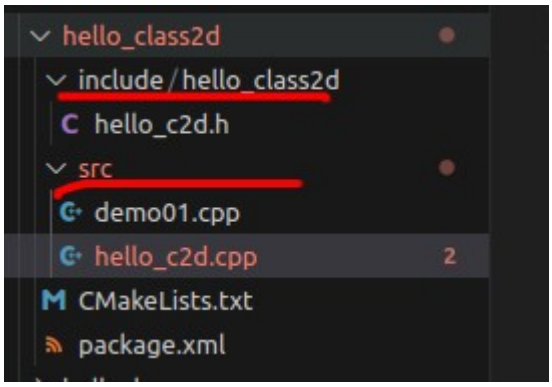
分文件 执行 class 文件 的记录

1、创建一个功能包 这个功能包的名字是 hello_class2d



2、在功能包的 src 文件夹下的 include/功能包下创建一个头文件
include/hello_class2d/hello_c2d.h
这个文件的具体如下图所展示

3、在 src 文件下 创建一个 cpp 文件 hello_class2d.cpp

1、引用的方法：功能包/hello_c2d.h 头文件的名字

2、在 hello_cpp namespace 中使用 因此也要在这里用

```cpp
#include"ros/ros.h"
#include"hello_class2d/hello_c2d.h"
//头文件是功能包 名字加上头文件将 之后放在include/功能包下面 因此我们也要这样引用
namespace hello_cpp{


void myhello::run()
{
    ROS_INFO("run a divie class cpp");
}

}
```

4、创建主要运行文件

demo01.cpp 文件

1、引用头文件的方法 hello_class2d/hello_c2d.h

2、使用 class 的方法 namecpass::class::成元函数 创建 使用大括号小括号等方法实现参数的传入

src > hello_class2d > src > demo01.cpp > main(int, char * [])

```cpp
#include"ros/ros.h"
#include"hello_class2d/hello_c2d.h"

int main(int argc, char *argv[])

{

    ros::init(argc,argv,"class_hello_desperate");
    hello_cpp::myhello obj;
    obj.run();
    return 0;
}
```

5、编辑 cmakelist 文件

1、把分文件实现的 class 的头文件放到里面

```
## Specify additional locations of header files
## Your package locations should be listed before other locations
include_directories(
include
  ${catkin_INCLUDE_DIRS}
)
```

2、建立一个连接 add_library(hello_c2d
头文件
cpp 文件)

add_library 建立一个连接 如果我们让 hello_c2d 就算与这个 class 的头文件和 实现文件关联 的方法

将 hello_c2d 与 头文件和 cpp 文件 关联

第一个是头文件 我们的头文件放在 功能包/src/include/功能包名/ .h 文件
我们主要从 include 开始写就 ok 了

第二个是 我们的 cpp 文件
src/src/hello_c2d.cpp
我们在功能包的 src 文件下 的 src 文件下实现 class 类的成员函数的 cpp 文件

```
123  ## Declare a C++ library 声明c++库 把头文件 和 源文件放进去  ${PROJECT_NAME}==hello_class2d
124  add_library(hello_c2d
125    include/${PROJECT_NAME}/hello_c2d.h
126    src/hello_c2d.cpp
127  )
128
```

3、建立 add execute 和 target link librariese

注意引用和使用的方法

src > hello_class2d > src > demo01.cpp > main(int, char * [])

```
1   #include"ros/ros.h"
2   #include"hello_class2d/hello_c2d.h"
3
4   int main(int argc, char *argv[])
5
6   {
7
8       ros::init(argc,argv,"class_hello_desperate");
9       hello_cpp::myhello obj;
10      obj.run();
11      return 0;
12  }
```

我们通过 demo01 调用 demo01.cpp 文件 在这个文件我们使用 class 的实例化的 对象

rosrun hello_class2d demo01 这样运行的时候 是可以调用我们函数内部的 class 文件

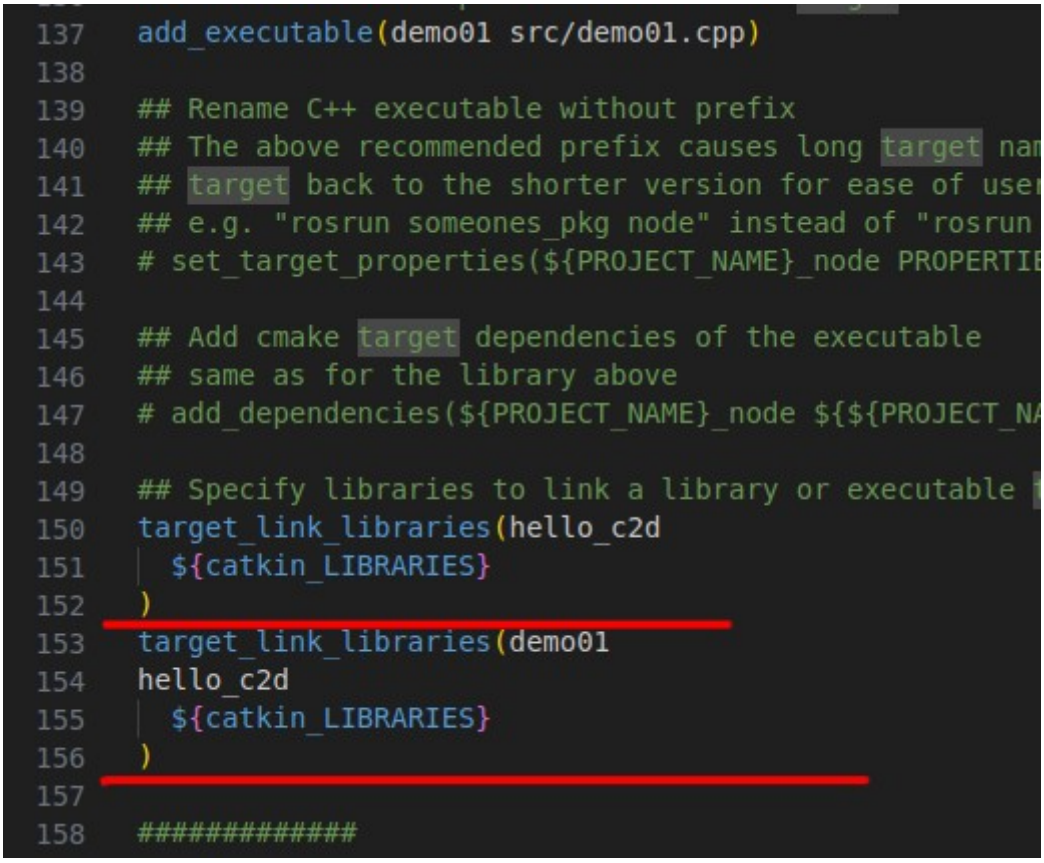
首先 我们需要看以下以前的写法
我们是在 add_execute 定义的名字　实际执行的 cpp 文件
targtet_link_libraries 定义的名字 和编译空间关联上

```
36  add_executable(demo01_param_set src/demo01_param_set.cpp)
37  add_executable(demo02_param_get src/demo02_param_get.cpp)
38  add_executable(demo03_param_del src/demo03_param_del.cpp)
39  ## Rename C++ executable without prefix
40  ## The above recommended prefix causes long target names, the following r
41  ## target back to the shorter version for ease of user use
42  ## e.g. "rosrun someones_pkg node" instead of "rosrun someones_pkg someon
43  # set_target_properties(${PROJECT_NAME}_node PROPERTIES OUTPUT_NAME node
44
45  ## Add cmake target dependencies of the executable
46  ## same as for the library above
47  # add_dependencies(${PROJECT_NAME}_node ${${PROJECT_NAME}_EXPORTED_TARGET
48
49  ## Specify libraries to link a library or executable target against
50  target_link_libraries(demo01_param_set
51    ${catkin_LIBRARIES}
52  )
53  target_link_libraries(demo02_param_get
54    ${catkin_LIBRARIES}
55  )
56  target_link_libraries(demo03_param_del
57    ${catkin_LIBRARIES}
58  )
59  #############
```

现在我们的实现方法
份文件实现的 class 文件需要与 编译空间关联
图的 150-152 行
我们在之前通过 hello_c2d 这个名字 与 class 的头文件和实现文件关联了

```
123  ## Declare a C++ library 声明c++库 把头文件 和 源文件放进去  ${PROJECT_NAME}==hello_class2d
124  add_library(hello_c2d
125    include/${PROJECT_NAME}/hello_c2d.h
126    src/hello_c2d.cpp
127  )
128
```

我们是通过 demo01 调用 主文件
因此 dmeo01 的这个名字需要和 class 定义的名字和编译空间关联



```
137  add_executable(demo01 src/demo01.cpp)
138
139  ## Rename C++ executable without prefix
140  ## The above recommended prefix causes long target nam
141  ## target back to the shorter version for ease of user
142  ## e.g. "rosrun someones_pkg node" instead of "rosrun
143  # set_target_properties(${PROJECT_NAME}_node PROPERTIE
144
145  ## Add cmake target dependencies of the executable
146  ## same as for the library above
147  # add_dependencies(${PROJECT_NAME}_node ${${PROJECT_NA
148
149  ## Specify libraries to link a library or executable
150  target_link_libraries(hello_c2d
151    ${catkin_LIBRARIES}
152  )
153  target_link_libraries(demo01
154  hello_c2d
155    ${catkin_LIBRARIES}
156  )
157
158  #############
```

在这里总结就算 通过 hello_c2d 将 分文件实现 class 的头文件和 cpp 文件与当前编译文件关联上

demo01 是 add_execute 中 main 程序的 别名

main 程序是调用了 class 文件 因此 main 程序要与 class 关联
main 函数要与编译空间关联

class 的头文件 和实现的 cpp 文件是通过 hello_c2d