1 Results

2 To do‘s

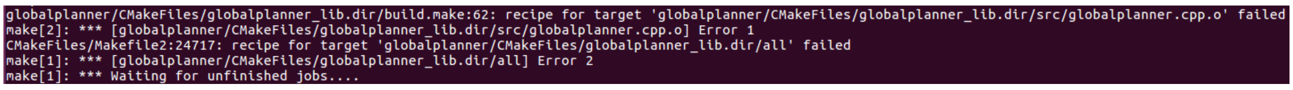
- Set up an IDE to run ROS nodes **Now the Qtcreator is set and variables can be displayed**

3 Achieved

4 Understanding the Global planner as plugin in ROS

Encountering the following issues, in following the ROS tutorial:

- **globalplanner.h file** not found - the issue must be found under some CMake stuff, see Section 5.1 about the CmakeLists.txt *find_package* **found solution by specifying the full path to the globalplanner.h file**



```
globalplanner/CMakeFiles/globalplanner_lib.dir/build.make:62: recipe for target 'globalplanner/CMakeFiles/globalplanner_lib.dir/src/globalplanner.cpp.o' failed
make[2]: *** [globalplanner/CMakeFiles/globalplanner_lib.dir/src/globalplanner.cpp.o] Error 1
CMakeFiles/Makefile2:24717: recipe for target 'globalplanner/CMakeFiles/globalplanner_lib.dir/all' failed
make[1]: *** [globalplanner/CMakeFiles/globalplanner_lib.dir/all] Error 2
make[1]: *** Waiting for unfinished jobs....
```

Figure 1: Cmake error

5 Notes on CMakeLists.txt, package.xml & CMake

5.1 CMakeLists.txt

Main important components of CMakeLists.txt files are

- 3. Find other CMake/Catkin packages needed for build (find_package())
- 7. Specify package build info export (catkin_package())
- 8. Libraries/Executables to build (add_library()/add_executable()/target_link_libraries())

Explanation

- 3. The main reason is that we need to specify which other CMake packages that need to be found to build our project using the CMake. What does the find_package do is to set CMake environmental variables to give informations about the package. The environment variables describe where the packages exported **header files** are, where **source files** are, what **libraries** the package depends on, and the paths of those libraries
- 7. This is required to specify catkin-specific information to the build system which in turn is used to generate pkg-config and CMake files. It consists of different specific components:
  - INCLUDE_DIRS - The exported included paths of the package. In the package itself this is a folder, where **headers** are placed.
  - LIBRARIES The exported libraries from the package/project
  - CATKIN_DEPENDS Other packages dependencies
- 8. Specifies the libraries of the current package to be built. Libraries specific builts are under the **devel/lib** folder

5.2 Package.xml

This file is also related to 5.1 since includes dependencies on other packages. Please in the official documentation refer to Format 3 Legacy. Section of interest is the **Build, Run and Test Dependencies**.

Dependencies are mainly of two types:

- 1. *build_depend* specify which packages are needed to build this package. This is the case when any file from these packages is required at build time. This can be including headers from these packages at compilation time, linking against libraries from these packages or requiring any other resource at build time (especially when these packages are find_package()-ed in CMake)
- 2. *run_depend* specify which packages are needed to run code in this package, or build libraries against this package. This is the case when you depend on shared libraries or transitively include their headers in public headers in this package (especially when these packages are declared as (CATKIN_)DEPENDS in catkin_package() in CMake.

VERY USEFUL LINK: https://answers.ros.org/question/217475/cmakeliststxt-vs-packagexml/ contains an explanation of CMake and package.xml in a nutshell.