Getting started with C++ in the Introduction To Self Driving Cars Nanodegree with a Mac and XCode

Important note: I do not work at or for Udacity, I am a student in a ND myself and this guide shall just help other students getting an easier start. Typos and errors in this guide are for free, let me know if you found one. :)
EDIT: A friend of mine pointed out that some Americans need to be reminded to not drink boiling coffee and dry cats in the microwave and might call Saul Goodman afterwards, so here it comes ;-):

www.logiball.de

@Alyxion
Michael Ikemann @ LinkedIn

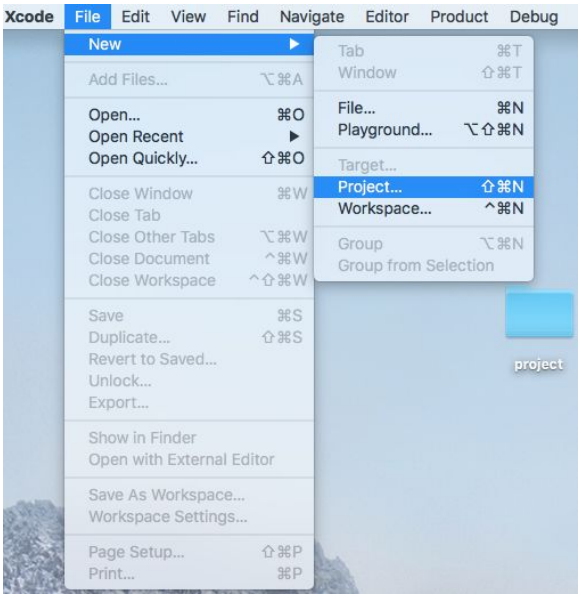
Click here for the Windows guide

Installation

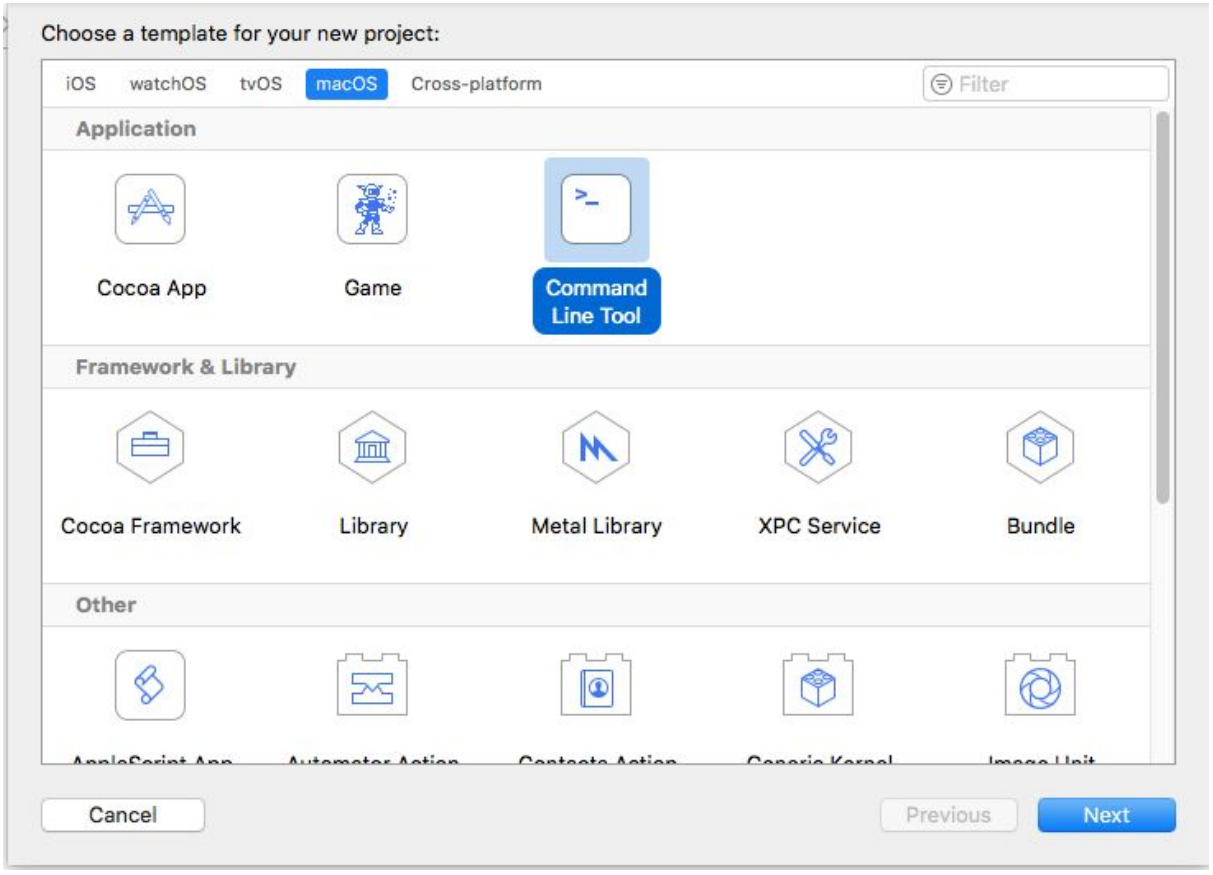
1. Download and unzip the project.zip and if not done so yet install the newest version of XCode from the Mac App Store.
After the installation you should find a symbol with a hammer in the app bar looking like the one down below in the middle. Open it now.

Project setup

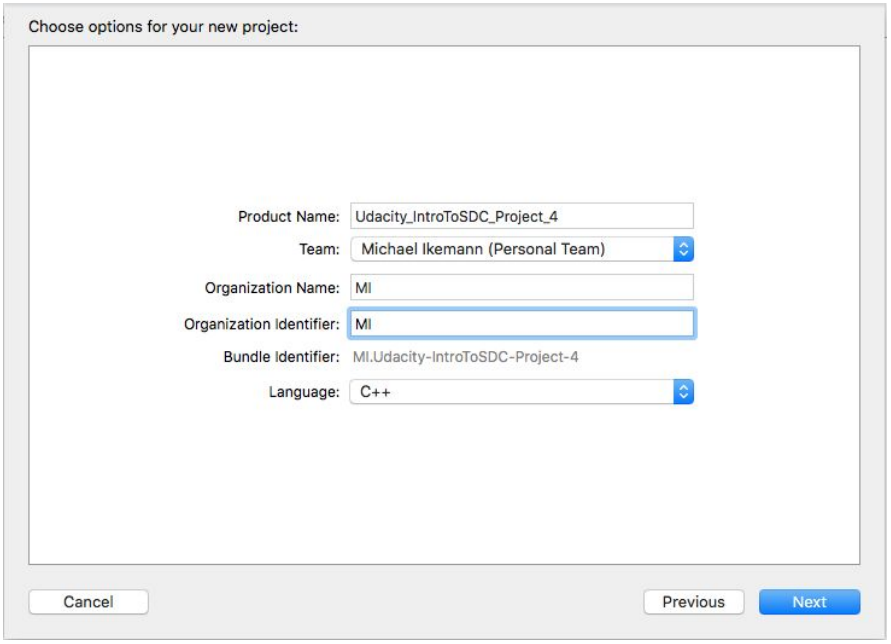
2. Click on New → Project to create a new project



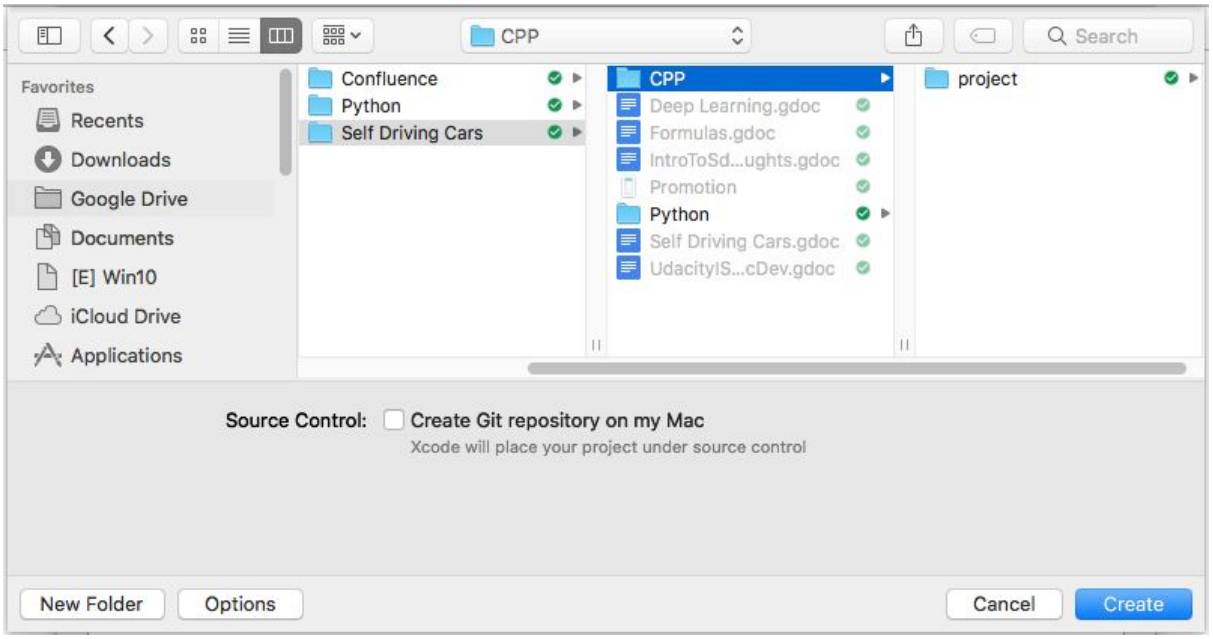
3. Select macOS in the top tab bar if and the "Command Line Tool" in the Application category like shown here.



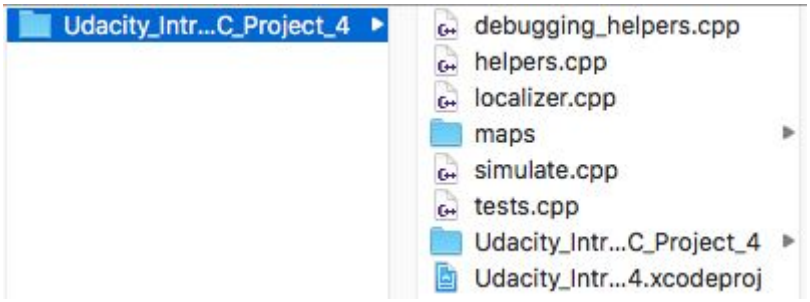
4. Choose a project name (this will be the later folder name) and enter the other credentials. If you do not have a developer account you can simply select "None" for team.



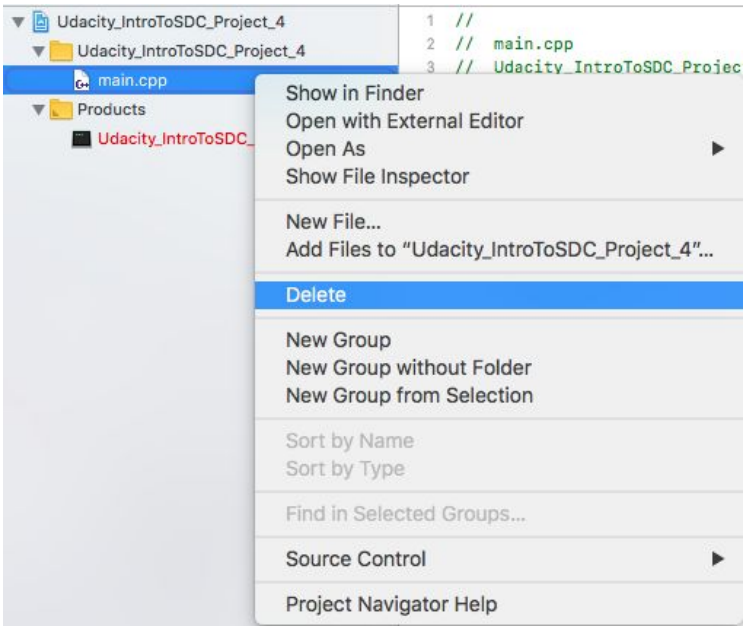
5. Select the folder where the new project folder shall be created. I placed mine next to the downloaded and extracted project folder.


CPP
Search
Favorites
Recents
Downloads
Google Drive
Documents
[E] Win10
iCloud Drive
Applications
Confluence
Python
Self Driving Cars
CPP
Deep Learning.gdoc
Formulas.gdoc
IntroToSd...ughts.gdoc
Promotion
Python
Self Driving Cars.gdoc
UdacityIS...cDev.gdoc
project
Source Control:
Create Git repository on my Mac
Xcode will place your project under source control
New Folder
Options
Cancel
Create

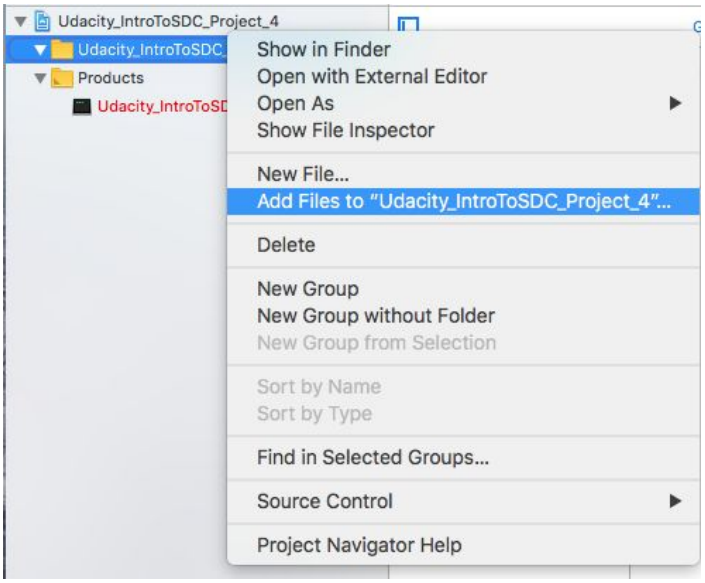
6. Move all files from the project folder into your new XCode project folder. This is not required, but this way all files are in a single folder afterwards. Should look like below when done.

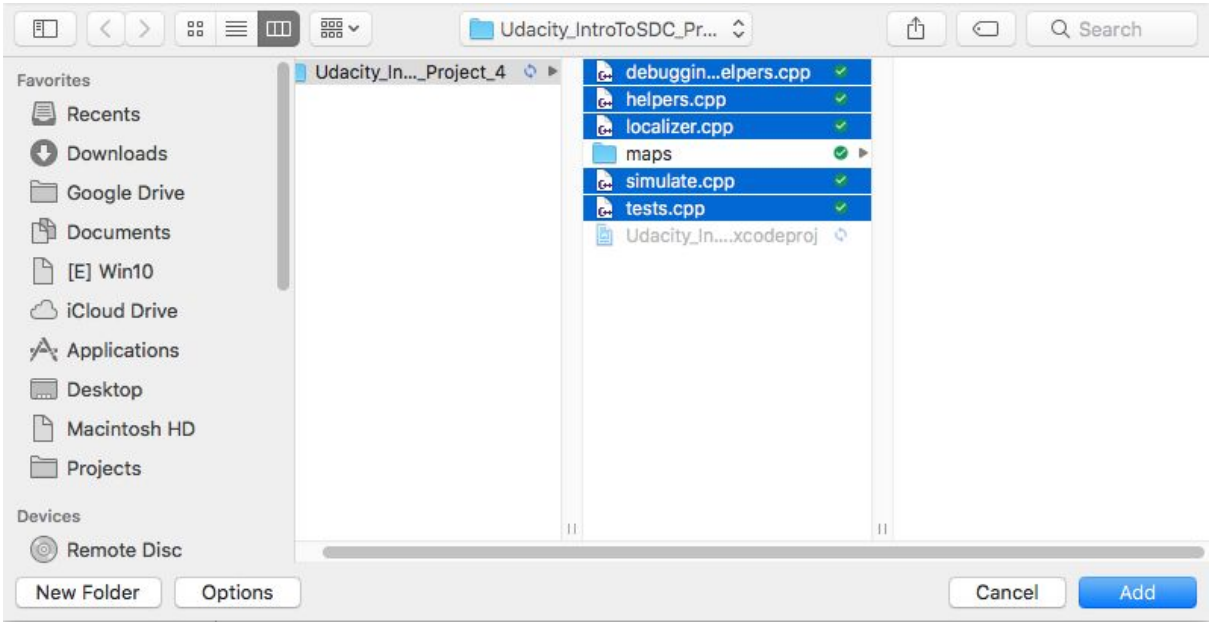


7. Remove the dummy main.cpp from the project. You can delete it.

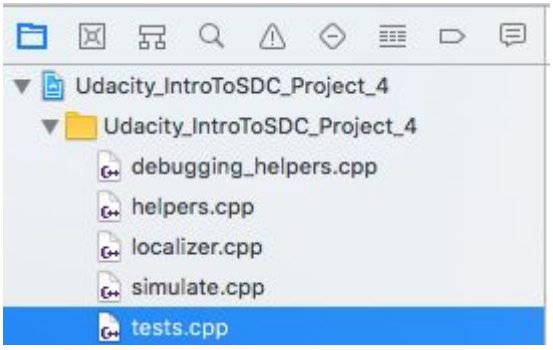


8. Add all project files ending with .cpp to the project.





9. You should now be able to see all files on the left side like here:

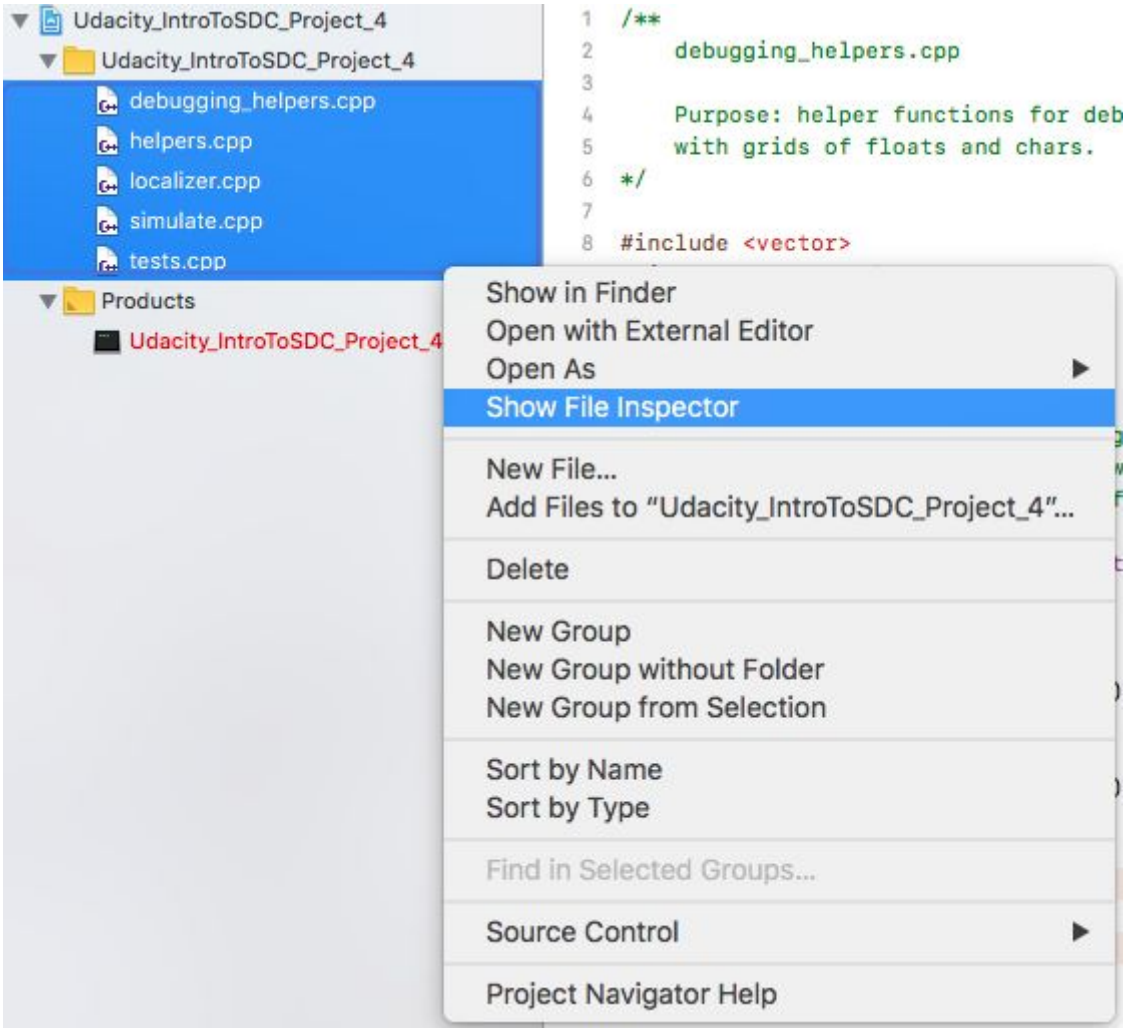


Running the app

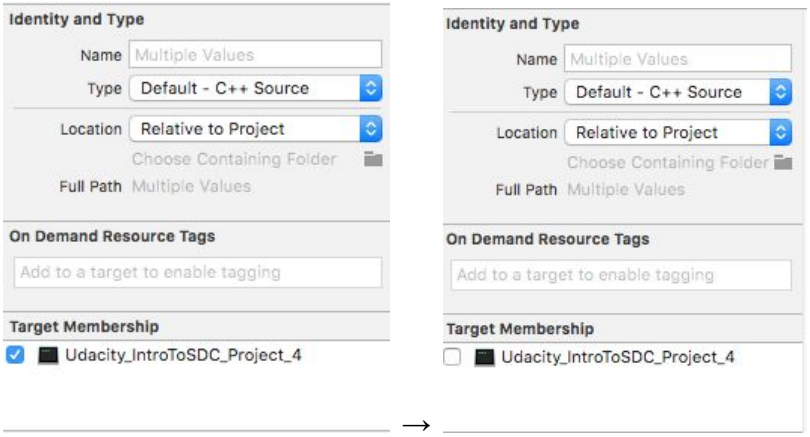
10. You can now click on Product → Run.

It won't work though, because the tests.cpp includes the other cpp files which leads to a lot of linker errors first of all.

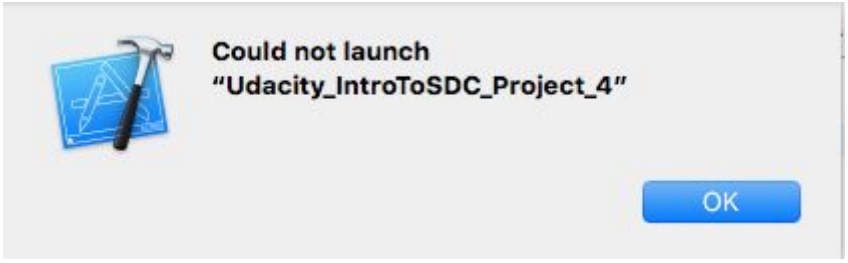
11. If you want to test in general if you theoretically would be able to run an app or to control which files shall be compiled and which not, you can do this by selecting the cpp files on the left → Right click → File inspector



12. Now you can select for which targets (OS X, iOS etc.) the currently selected files shall be taken into account/compiled. In this case our only target is an OS X console application. If we remove the check mark, the files will now be ignored. Note that for the project the check mark needs to be removed for all files except "tests.cpp".



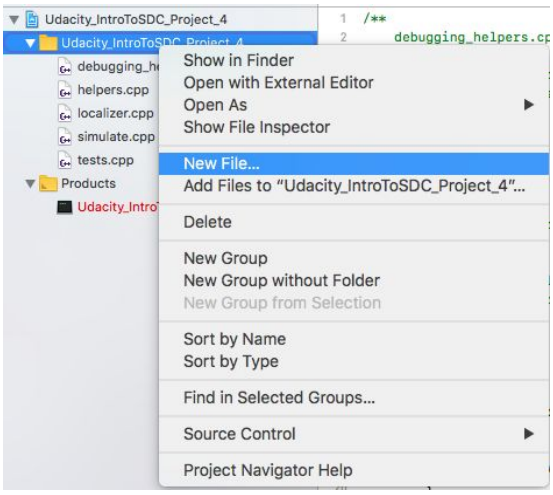
13. Great, we are now able to compile, but….



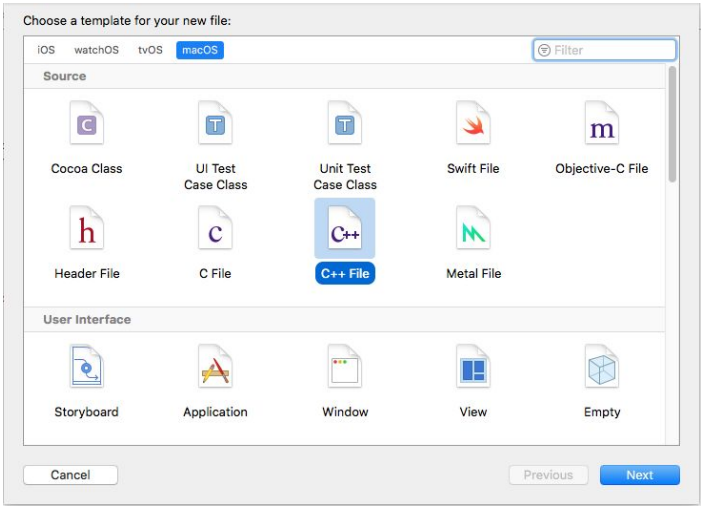
… there is nothing to run anymore of course as well.

Adding own files

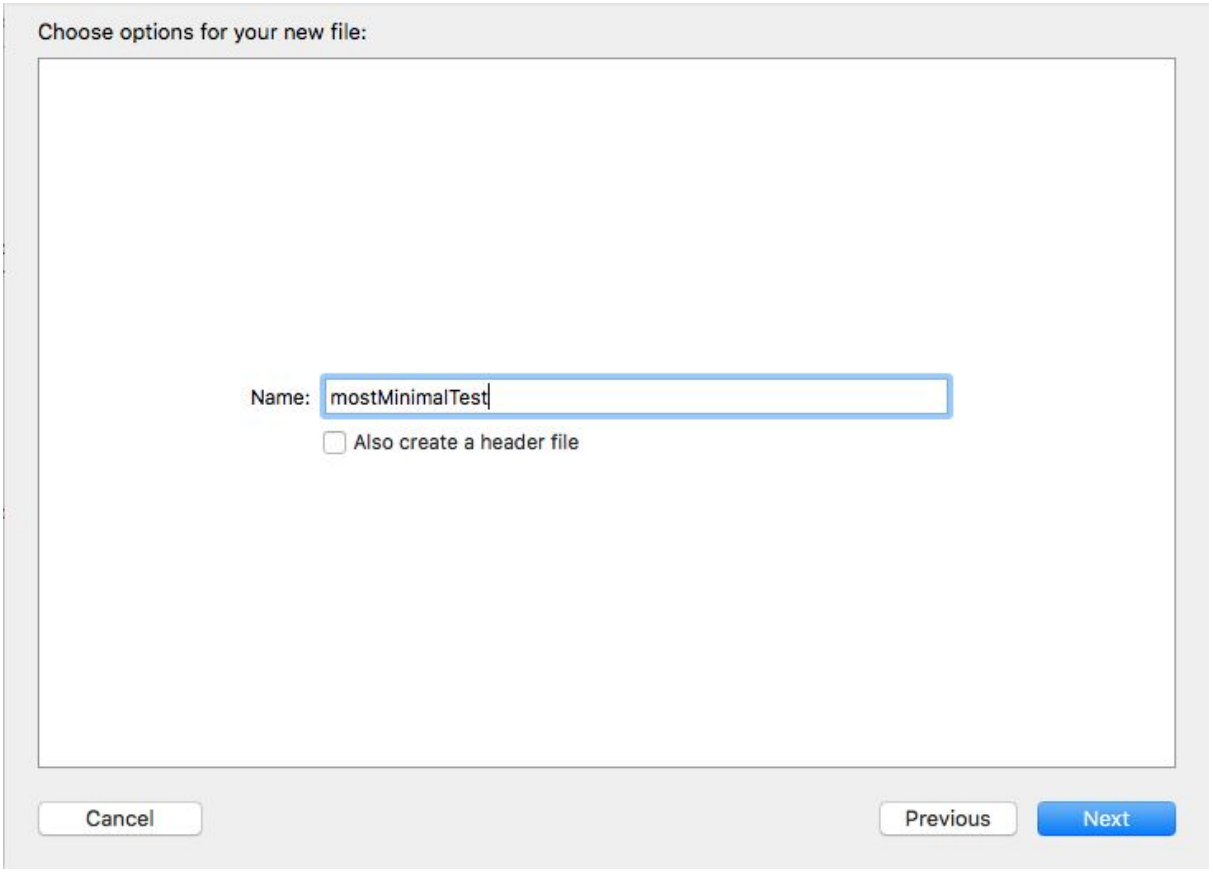
14. Right click on the yellow main folder → New File



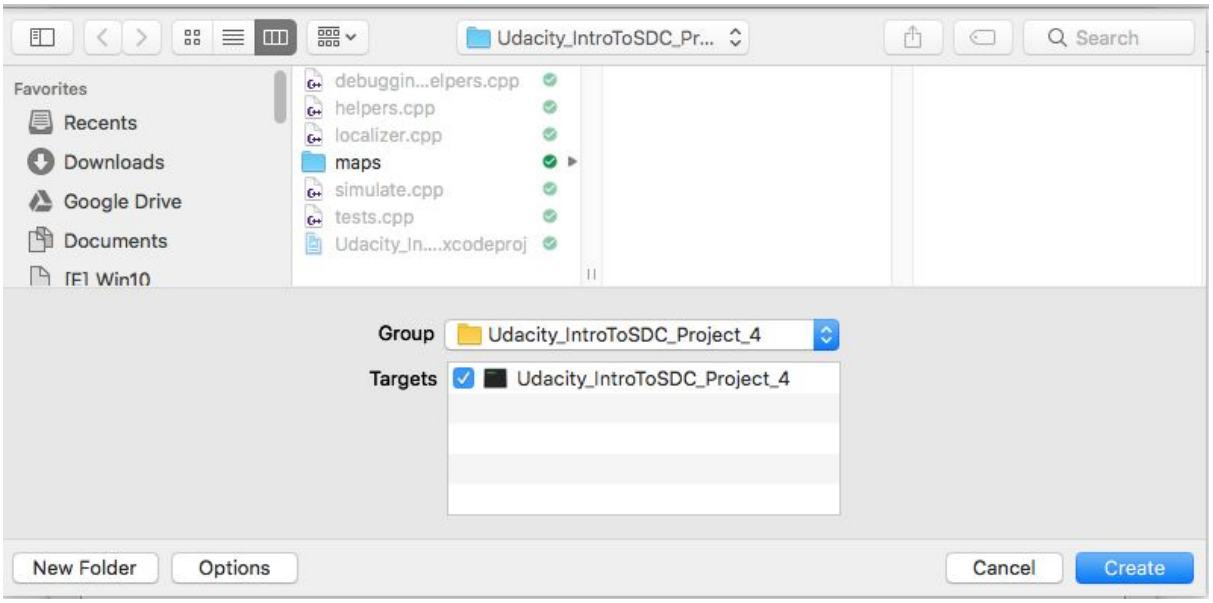
15. Select C++ File



15. Choose what ever name you like, unselect "Also create a header file"



16. Take care our Target is selected

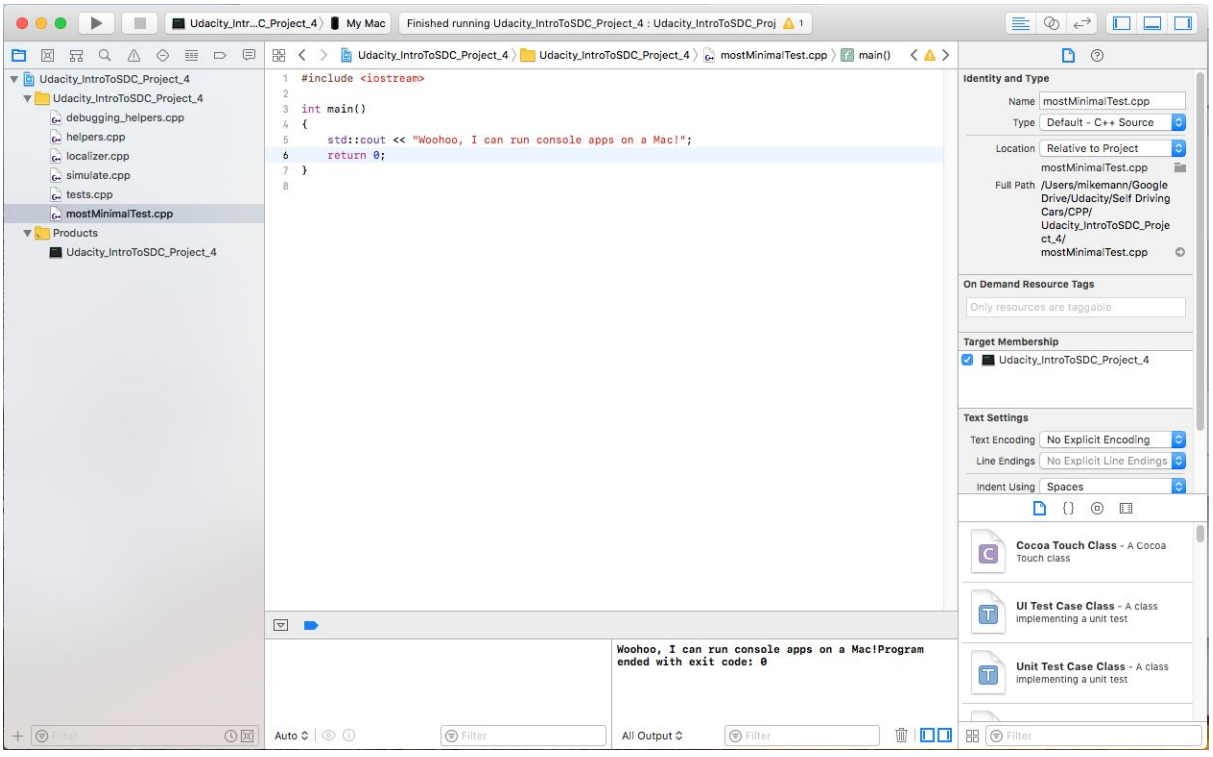


17. Remove the old code, enter some code from the previous lessons or something like this:

```
#include <iostream>

int main()
{
    std::cout << "Woohoo, I can run console apps on a Mac!";
    return 0;
}
```

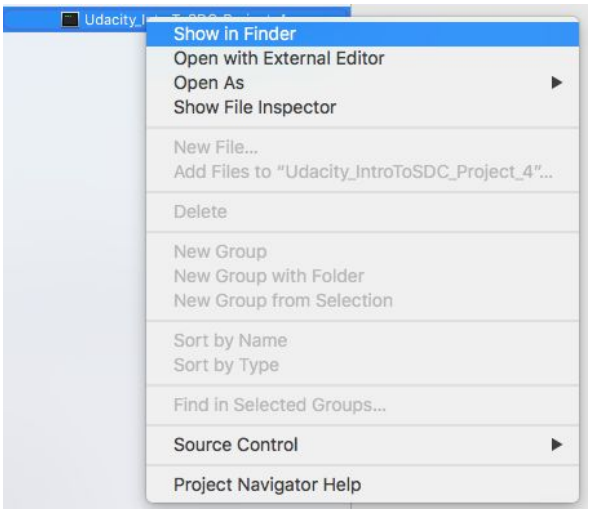
18. Now hit Product → Run again and the result should look like this, see console at the bottom center.



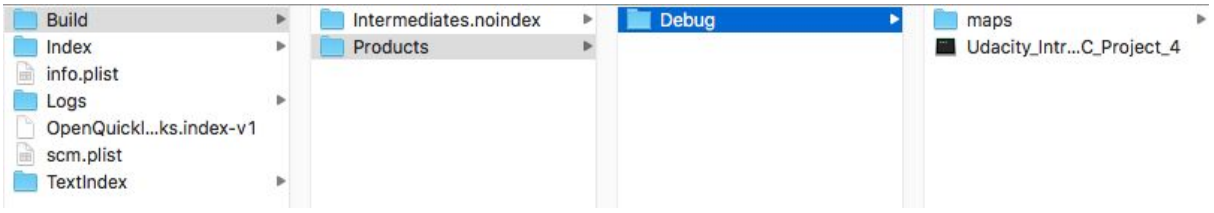
19. If everything works fine for your most minimal test you can now disable this one and reactivate the file "tests.cpp" again (remember, select files on the left → File Inspector → Target Membership on the right). Do <u>not</u> reactive the other files, as "tests.cpp" includes them only this one is required.

Making the map data files available for the app

20. When you have compiled your project successfully for the first time, your product on the left side should now show a black label. Right click it and then click on "Show in Finder".



21. Open another Finder window or tab and copy the folder "maps" from your project folder into the application folder so the test app will be able to find the map data.



22. Finalize all functions with todo (Command-Shift-F and search for "TODO"), and you're done.

```
! - normalize function worked correctly!

! - blur function worked correctly!
! - initialize_beliefs function worked correctly!

! - move function worked correctly with zero blurring

! - sense function worked correctly

Program ended with exit code: 0
```

23. So… like Sebastian would likely say in a video now :)... you are already halfway done on the way to a Self Driving Car…. the other 2.9 million lines of code are just detail ;-).

Happy coding!

Code signing trouble

If you should encounter code signing errors because you accidentally chose a team, you can disable code signing by selecting the project on the left, going to Build Settings and toggling the Identity to Don't Code Sign as it's not required for debugging purposes.