Getting Started

Getting Started

Hello and welcome to FRC C++. Do not panic you'll be fine. The first thing you need to do is get FRC VS code installed. If you already have VS code you can link the install to that version of VS Code. If you are on Windows you will have to install Visual Studio with C++ packages and link FRC to that. When you start a project you will need to install all the libraries from wpilib. The link: https://github.com/wpilibsuite/allwpilib.git will take you to a github where you can download everything you need. You will need these folders to be in your include path for Gradle to work properly. If you've never programmed in C++ before don't worry. Most of the programming is restricted to the WPI API anyways. That being said you should still learn the basics online. There are several websites that will teach you the basics for free. I'd suggest Codecademy or Sololearn.

Note: at the time of writing you need to manually compile an old ssl version on Linux if that's what you are using. Follow this guide to do that: here. To my knowledge WPILib should install on windows and MacOS fine. When following this guide modify the commands to open ssl 1.1.1.

Project Overview

When you open VSCode you'll have to make a project in WPI's tool-chain. There should be a red W

symbol that looks like this:

You will click it and type create a new project into the search bar. Make sure you know what project template your team is using. Your Project file system will contain several folders containing some of the dependencies. What we really care about is the 'src' folder which will contain a 'main' and a 'test' sub-directory. When merging to GitHub you will have to isolate the src folder. Just copy it into your GitHub project folder and open GitHub desktop.

GitHub

GitHub is an online server hosting service which hosts git servers for free. An account is free, simple to setup, and required to commit to the code. It allows us to store all of our code online and as we change it, it will make separate 'forks'(individual versions of the code) which isolate conflicting code. When we merge the code onto the main 'branch'(type of code AKA main, or test) we take each fork and compare it against the others to see what needs to be removed and what can stay in the code. This way we can all work on separate parts of the code asynchronously. You can download the code without an account but to upload you will need to make an account, download GitHub desktop(If you can't do this ask a software lead or a mentor), and clone the repository your team is using. When you clone a repository you are basically downloading all the files in the same file structure (similar to tar.gz). You can copy it into the project folder and replace your src folder. When you make changes to the code you will copy your src unto your GitHub directory from earlier and open GitHub desktop which will notice the changes and ask you to push your change. It is important to create a pull request as if you don't your code will only be added to your fork and not to the team's fork.

Coding

When you feel ready to look at code you have two options. If the team already has a programming lead ask where to see the code. If you are the programming lead then I suggest starting a project and making a github. There should be git merges done more often rather than less. It is a good idea to comment most segments of code with what they are accomplishing. By default we have no access to the Main() function. Rather the Main() function is in the background of the template which contains functions called internally. Most phases of the robotics match have their own init() and periodic() functions. The function of these should be fairly self explanatory. We set up the robot in the init() functions and implement decision making in the periodic() functions.

Best practices

1. LIMIT GLOBAL VARIABLE USE! When working with multiple people it is crucial to limit the amount of global variables you create as we don't want to have to worry about conflicting variable names.
2. Don't use the 'auto' variable allocation, if you don't know what type of data you need then you don't understand what you are trying to program. Ask for algorithmic help and someone will be happy to provide!
3. COMMENT YOUR CODE!!! There are multiple people who are going to be looking at your code and comments explain what you are doing.