C/C++ and Java Installation For 2020 FRC Teams

Mike Anderson
(robot_maker12@verizon.net)

Herndon High School
FRC Team #116

What We’ll Talk About

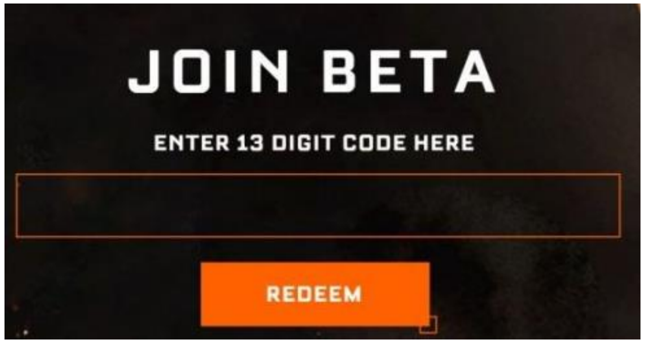
- Goals
- The development environment
- Talking to the RoboRIO
- Making it move
- Resources
- Summary

Goals

- The goal of this presentation is to help you understand how to prepare your development environment for use with C/C++ and Java
- We clearly can’t explain all of the aspects because we have limited time
  - But, you should leave here with a better understanding of the process
- We will be talking about the set up rather than the languages themselves
  - The WPILib is equivalent between the environments

Warning: Beta Code…

- What you will see is the 2020 Beta software that we've been working with over the past couple of months



- Some things are likely to change, but it's pretty feature complete at this point
- There were quite a bit of head scratching while we were working with getting things running
  - There have been some significant changes in the RoboRIO FPGA code concerning CAN bus and that impacts all CAN-centric operations

Why C/C++?

- C/C++ is a standard in embedded systems programming for over 30 years
  - It's still the most predominant language in embedded Linux, the IoT and the real-time operating system (RTOS) world
    - This gives your team valuable real-world experience
- It's compiled to native machine code
  - No virtual machine interpreters
    - No pausing due to garbage collection
  - It's fast
- It's the native language of the RoboRIO's Linux-based operating system
  - The environment is written in C and Assembler
  - You get easy, direct access to the underlying OS
- C++ is object oriented
  - Full support from WPILib

Why Not C/C++?

- C/C++ is compiled
  - This adds complexity to the build
- C/C++ is textual
  - There are no cutesy GUIs with lots of obscure symbols and squiggly lines ☺
- There is no VM to catch your mistakes
  - The syntax is similar to Java
    - Java was derived from C++
    - Java VM is written in C/C++
- C/C++ has pointers
  - Objects can be referenced in many different ways
  - This concept can be troublesome for some developers
  - Java calls pointers “references”

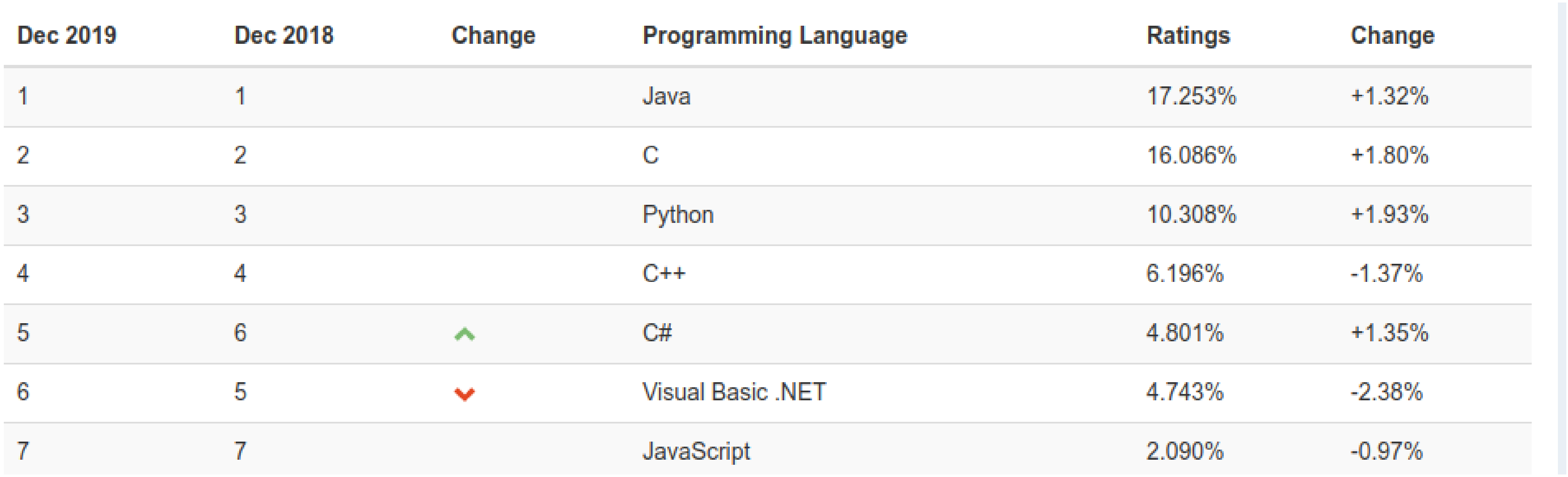
Why Java?

- Java has wide support in the industry
  - Object-oriented approach with lots of reference material
- Java is the language used on the AP exams
  - Used in many computer science classes
- Java is a byte-code interpreted language
  - The use of the Virtual Machine (VM) allows for many dynamic language features
- The VM will help catch some common memory mistakes
- The version of Java used on the RoboRIO is the OpenJDK V11.0.4
- WPILib is actually written in Java and then translated to C++

Why Not Java?

- Java is interpreted
  - Performance is lower than C/C++
- Java is also textual like C++
  - But, Java can be written using either imperative or declarative programming styles
- The version of Java on the RoboRIO is not optimized for use in control systems
  - The version is actually targeted at business applications
- Garbage collection cycle will cause the robot to hesitate during the mark-and-sweep cycle
  - Given the length of our matches, this should not be a problem

Top 7 Languages – Dec 2019

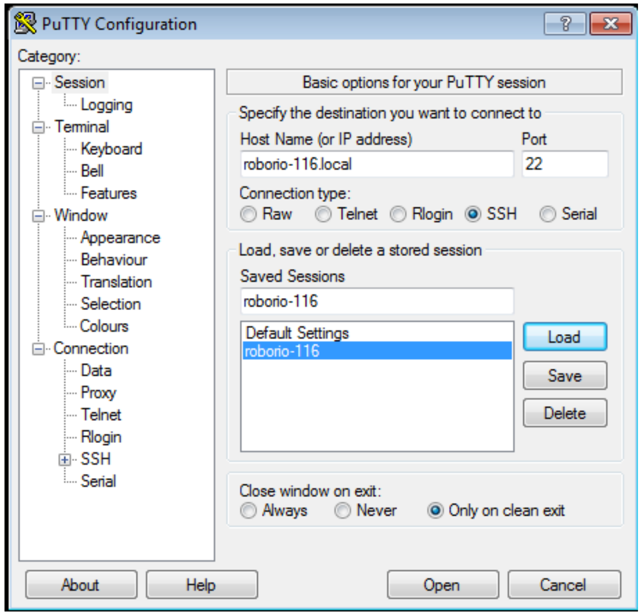


Dec 2019	Dec 2018	Change	Programming Language	Ratings	Change
1	1		Java	17.253%	+1.32%
2	2		C	16.086%	+1.80%
3	3		Python	10.308%	+1.93%
4	4		C++	6.196%	-1.37%
5	6	▲	C#	4.801%	+1.35%
6	5	▼	Visual Basic .NET	4.743%	-2.38%
7	7		JavaScript	2.090%	-0.97%

- LabVIEW was #42 on this list
  - This represents a 7 place drop from 2019

Some Useful Info…

- The RoboRIO runs Linux
  - SSH server is available
    - Use Putty on Windows to get to SSH shell
  - File transfers from IDE use SCP
- Addressing is via mDNS
  - roborio-<team #>-FRC.local
- The Web server on the RoboRIO is being redesigned at this time so we don't quite know what it will look like yet
- Do not delete "admin" account or change its password
  - All program transfers require it


PuTTY Configuration

Basic options for your PuTTY session

Specify the destination you want to connect to

Host Name (or IP address): roborio-116.local

Port: 22

Connection type: Raw Telnet Rlogin [✓ SSH] Serial

Load, save or delete a stored session

Saved Sessions: roborio-116

Default Settings
roborio-116

Close window on exit: Always Never [✓ Only on clean exit]

The Development Environment

- The FIRST-supported development platform for C/C++ and Java is Microsoft Visual Studio Code tool
  - Available for Windows, MacOS and Linux
  - The compiler is the open-source GCC 7.3 compiler
    - Supports C++11 extensions
- The C compiler is actually a cross-compiler
  - We are building on an x86 for an ARM-based system
    - Again, this is a standard approach for commercial, embedded development
- For Java, the build system will run the Java source code through the OpenJDK to produce Java bytecode

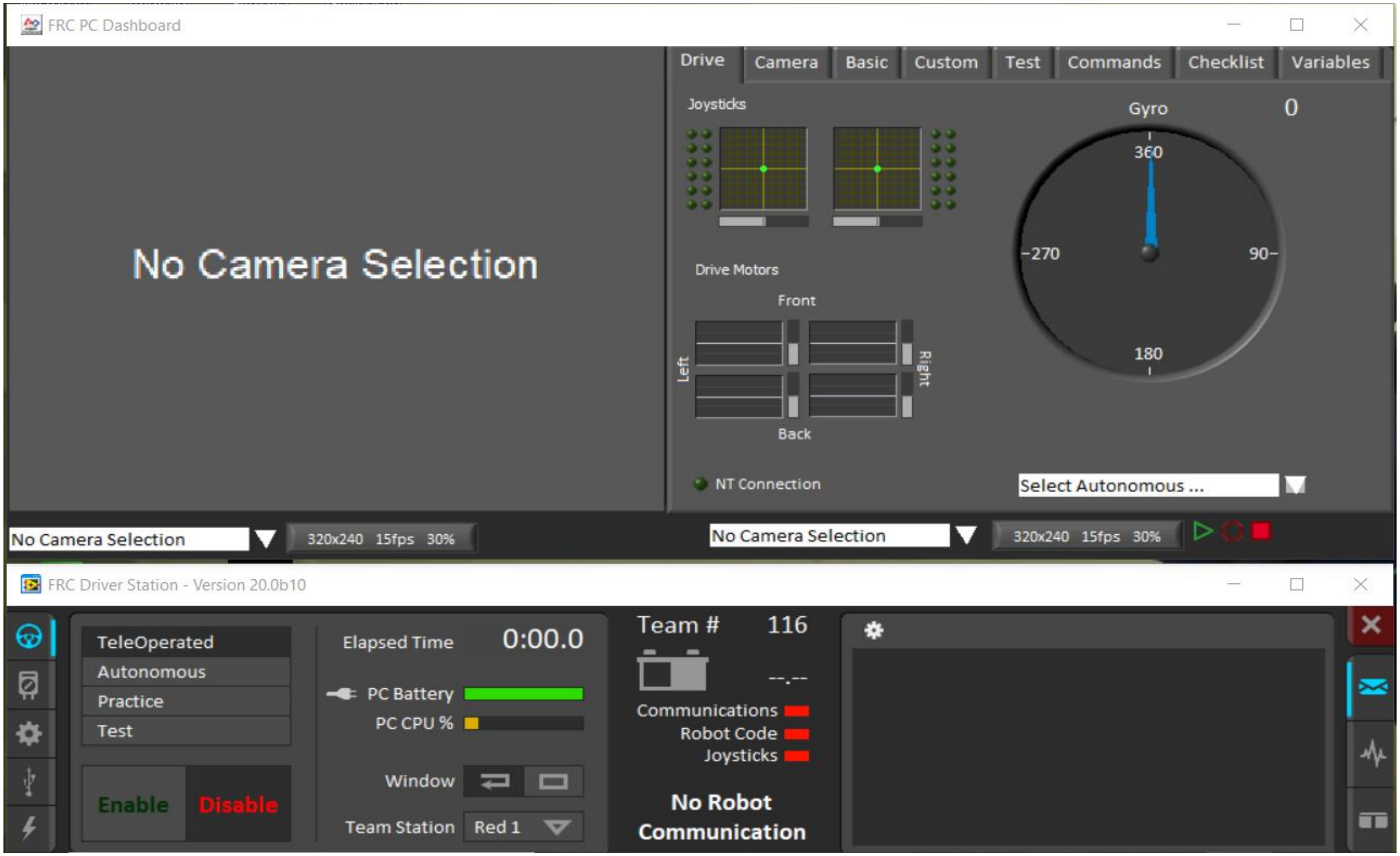
Development Environment #2

- The installation tool will install the OpenJDK
  - And, install VSCode if you select that option
  - It will install both C/C++ and Java by default
- The build environment is the GradleRIO plug-in from Github
  - https://github.com/wpilibsuite/GradleRIO
  - Uses Gradle V6
- The WPILib VSCode plug-in will have all of the tools needed to build and deploy code to the robot

Install National Instruments Update

- It's probably best if you uninstall previous versions
  - It will take at least 10-20 minutes to install
    - Longer if you need to uninstall the previous version
- This will also install the FRC Driver Station application
  - This will also install the RoboRIO imaging tool and the latest image release
    - They are still having problems with the firmware update, but the image update works fine
      - We assume they'll get this working soon
- The system will need to reboot after installation

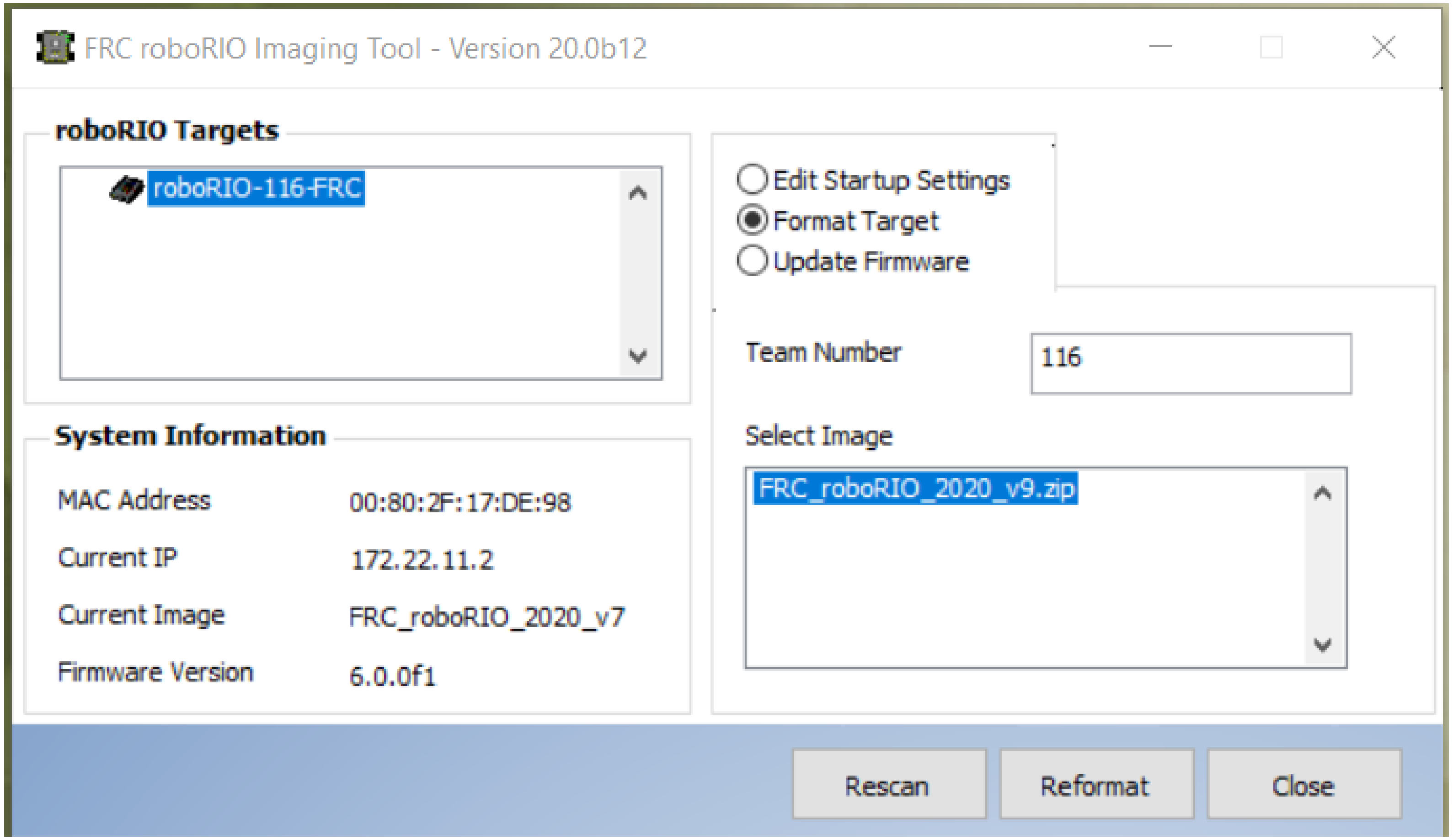
2020 Driver Station



Getting Your RoboRIO Ready

- Before you can start development, you’ll need to make sure that your RoboRIO has the proper operating system image on it
  - This is accomplished using the RoboRIO imaging tool or it can be done through LabVIEW
- Java runtime engine will be installed when you deploy your first Java program to the RoboRIO

Update the RoboRIO



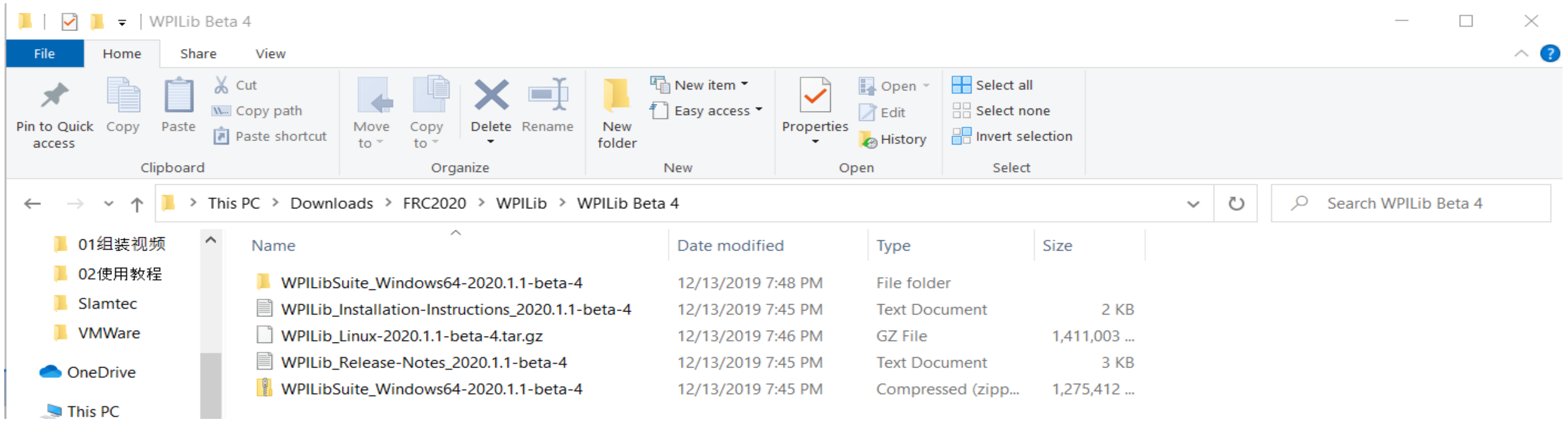
○ Edit Startup Settings
● Format Target
○ Update Firmware

Team Number	116
Select Image	FRC_roboRIO_2020_v9.zip

MAC Address	00:80:2F:17:DE:98
Current IP	172.22.11.2
Current Image	FRC_roboRIO_2020_v7
Firmware Version	6.0.0f1

Launch the WPILib/tools Install

- Unlike last year, the WPILib tools are extracted from a separate archive
  - ~ 2.6 GBs for the zipped download
- We'll look at the Windows installation, but there are install steps for both MacOS and Linux as well

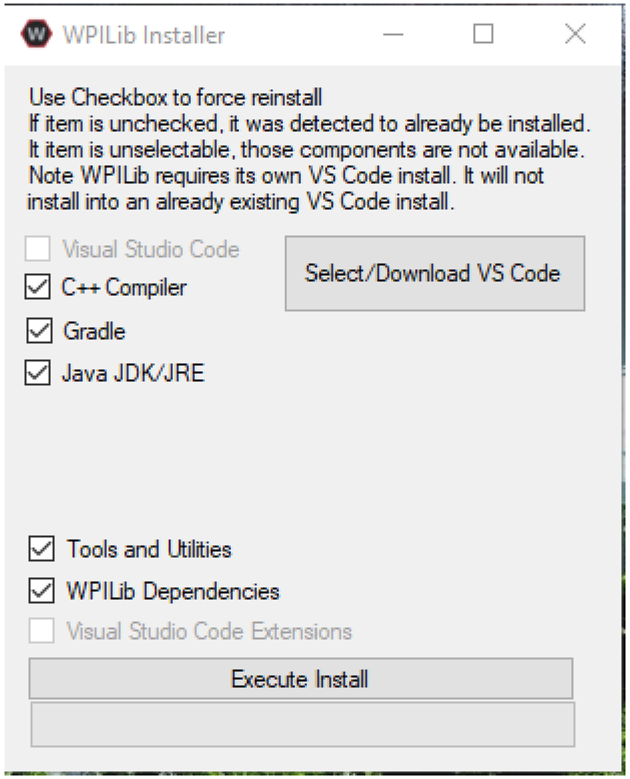


Name	Date modified	Type	Size
WPILibSuite_Windows64-2020.1.1-beta-4	12/13/2019 7:48 PM	File folder	
WPILib_Installation-Instructions_2020.1.1-beta-4	12/13/2019 7:45 PM	Text Document	2 KB
WPILib_Linux-2020.1.1-beta-4.tar.gz	12/13/2019 7:46 PM	GZ File	1,411,003 ...
WPILib_Release-Notes_2020.1.1-beta-4	12/13/2019 7:45 PM	Text Document	3 KB
WPILibSuite_Windows64-2020.1.1-beta-4	12/13/2019 7:45 PM	Compressed (zipp...	1,275,412 ...

Installation of Visual Studio Code

- In theory, you should be able to use an existing VSCode installation
  - That didn’t work too well in the Beta, so we opted to allow the installation tool to install VSCode for us
- The installation will take about 10 minutes
  - There are still some manual settings that you’ll need to do with search paths for the JDK and the JAVA_HOME environment variable
    - Requires that you run a script to update these things
  - Presumably, these things will be taken care of by kickoff

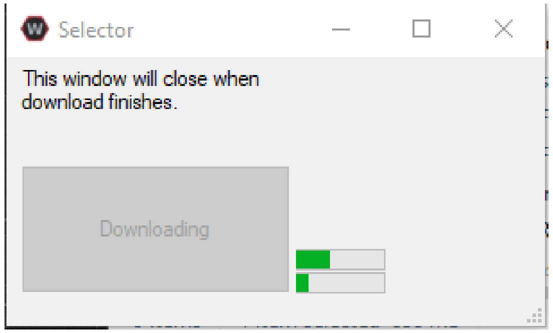
Installing WPILib/VSCode



WPILib Installer

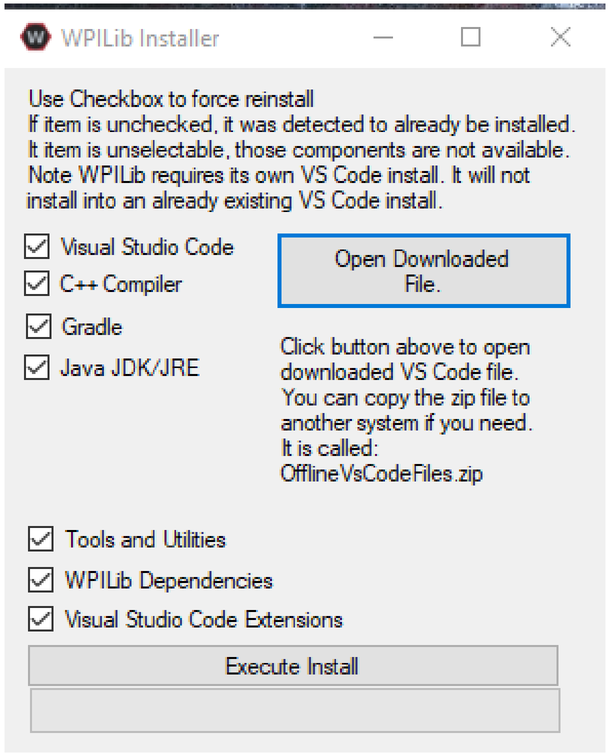
Use Checkbox to force reinstall
If item is unchecked, it was detected to already be installed.
It item is unselectable, those components are not available.
Note WPILib requires its own VS Code install. It will not install into an already existing VS Code install.

- [ ] Visual Studio Code
- [✓] C++ Compiler
- [✓] Gradle
- [✓] Java JDK/JRE
- [✓] Tools and Utilities
- [✓] WPILib Dependencies
- [ ] Visual Studio Code Extensions



Selector

This window will close when download finishes.



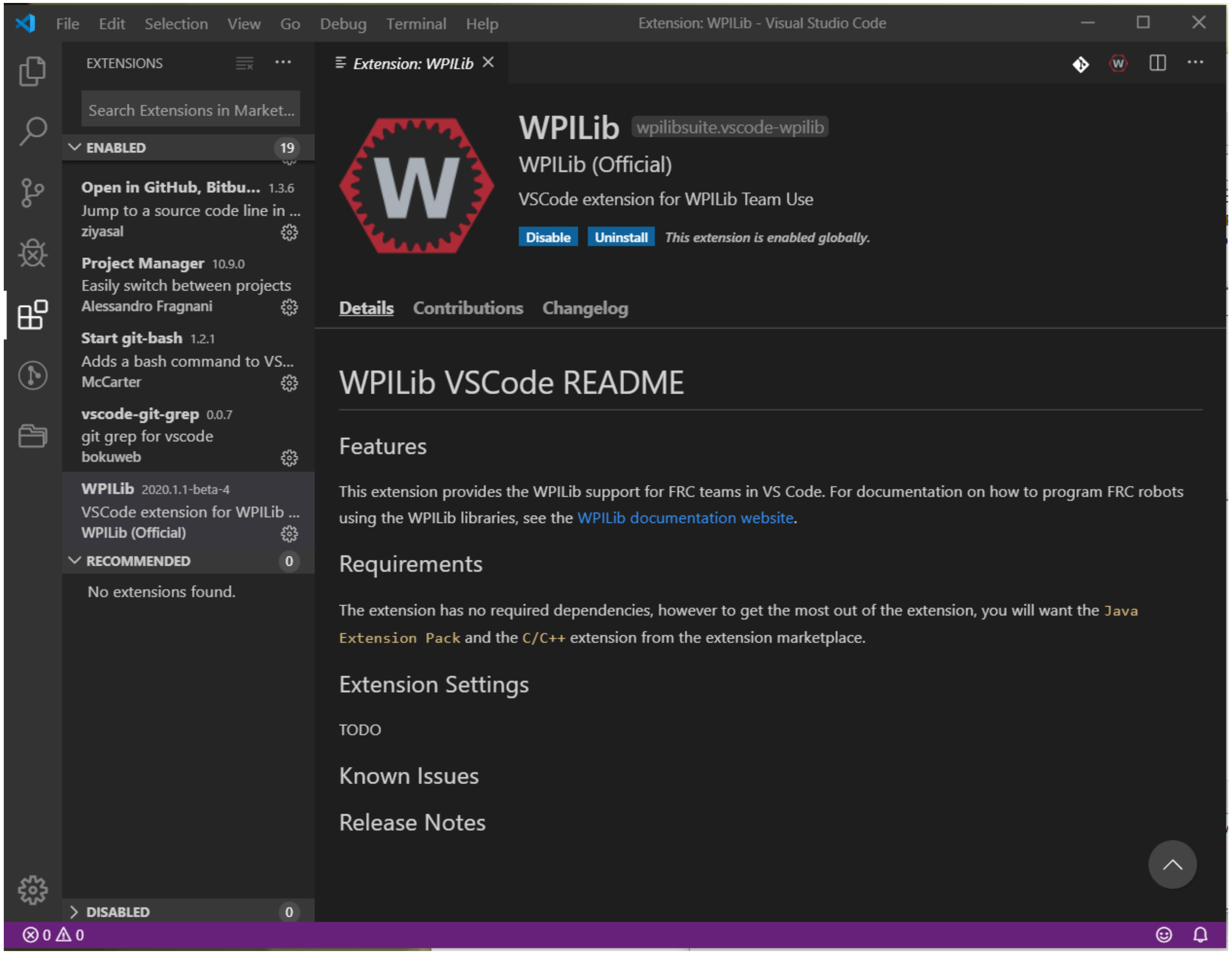
WPILib Installer

Use Checkbox to force reinstall
If item is unchecked, it was detected to already be installed.
It item is unselectable, those components are not available.
Note WPILib requires its own VS Code install. It will not install into an already existing VS Code install.

- [✓] Visual Studio Code
- [✓] C++ Compiler
- [✓] Gradle
- [✓] Java JDK/JRE
- [✓] Tools and Utilities
- [✓] WPILib Dependencies
- [✓] Visual Studio Code Extensions

Click button above to open downloaded VS Code file.
You can copy the zip file to another system if you need.
It is called:
OfflineVsCodeFiles.zip

The VSCode with WPILib Extension


File Edit Selection View Go Debug Terminal Help
Extension: WPILib - Visual Studio Code
EXTENSIONS
Search Extensions in Market...
ENABLED 19
Extension: WPILib
Disable
Uninstall
Details Contributions Changelog
RECOMMENDED 0
DISABLED 0
⊗ 0 ⚠ 0

Open in GitHub, Bitbu... 1.3.6
Jump to a source code line in ...
ziyasal

Project Manager 10.9.0
Easily switch between projects
Alessandro Fragnani

Start git-bash 1.2.1
Adds a bash command to VS...
McCarter

vscode-git-grep 0.0.7
git grep for vscode
bokuweb

WPILib 2020.1.1-beta-4
VSCode extension for WPILib ...
WPILib (Official)

No extensions found.

WPILib wpilibsuite.vscode-wpilib
WPILib (Official)
VSCode extension for WPILib Team Use
This extension is enabled globally.

WPILib VSCode README

Features

This extension provides the WPILib support for FRC teams in VS Code. For documentation on how to program FRC robots using the WPILib libraries, see the WPILib documentation website.

Requirements

The extension has no required dependencies, however to get the most out of the extension, you will want the `Java Extension Pack` and the `C/C++` extension from the extension marketplace.

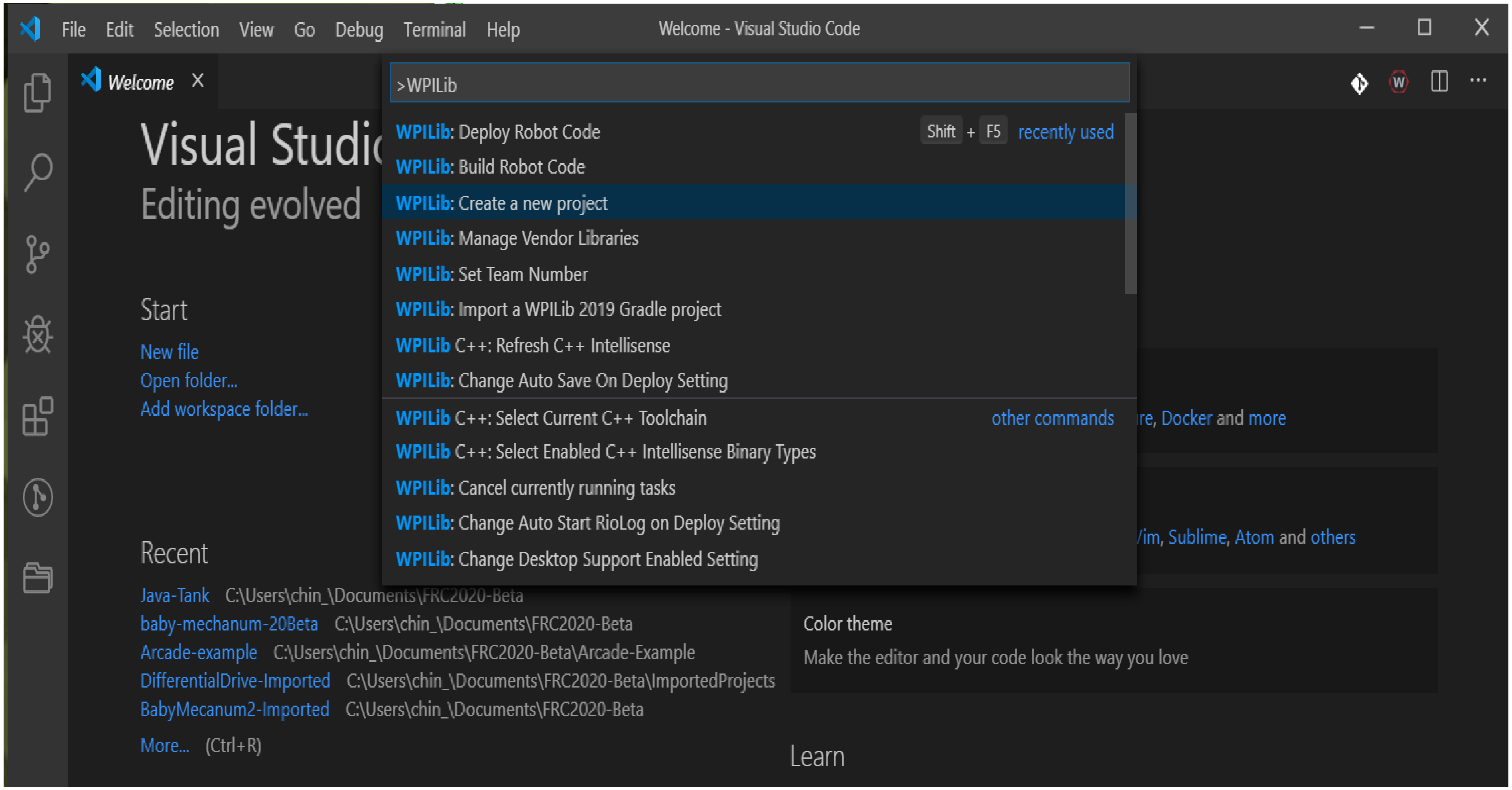
Extension Settings

TODO

Known Issues

Release Notes

Creating a Project #1



File Edit Selection View Go Debug Terminal Help
Welcome - Visual Studio Code
Welcome

```
>WPILib
WPILib: Deploy Robot Code                     Shift + F5  recently used
WPILib: Build Robot Code
WPILib: Create a new project
WPILib: Manage Vendor Libraries
WPILib: Set Team Number
WPILib: Import a WPILib 2019 Gradle project
WPILib C++: Refresh C++ Intellisense
WPILib: Change Auto Save On Deploy Setting
WPILib C++: Select Current C++ Toolchain      other commands
WPILib C++: Select Enabled C++ Intellisense Binary Types
WPILib: Cancel currently running tasks
WPILib: Change Auto Start RioLog on Deploy Setting
WPILib: Change Desktop Support Enabled Setting
```

Visual Studio
Editing evolved

Start
New file
Open folder...
Add workspace folder...

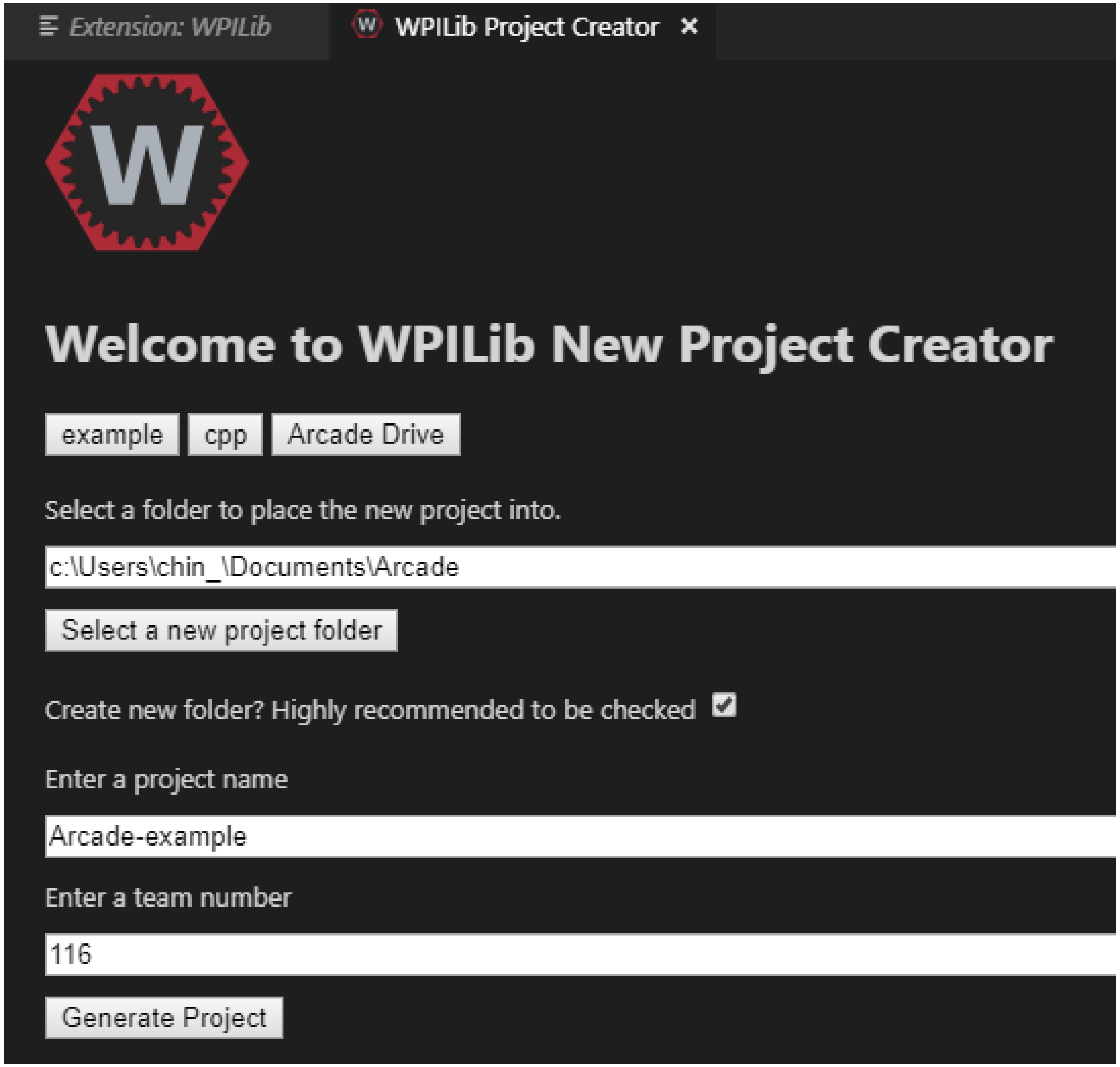
Recent
Java-Tank C:\Users\chin_\Documents\FRC2020-Beta
baby-mechanum-20Beta C:\Users\chin_\Documents\FRC2020-Beta
Arcade-example C:\Users\chin_\Documents\FRC2020-Beta\Arcade-Example
DifferentialDrive-Imported C:\Users\chin_\Documents\FRC2020-Beta\ImportedProjects
BabyMecanum2-Imported C:\Users\chin_\Documents\FRC2020-Beta
More... (Ctrl+R)

re, Docker and more
/im, Sublime, Atom and others

Color theme
Make the editor and your code look the way you love

Learn

Creating a Project #2



Extension: WPILib

WPILib Project Creator

Welcome to WPILib New Project Creator

example cpp Arcade Drive

Select a folder to place the new project into.

c:\Users\chin_\Documents\Arcade

Select a new project folder

Create new folder? Highly recommended to be checked ✓

Enter a project name

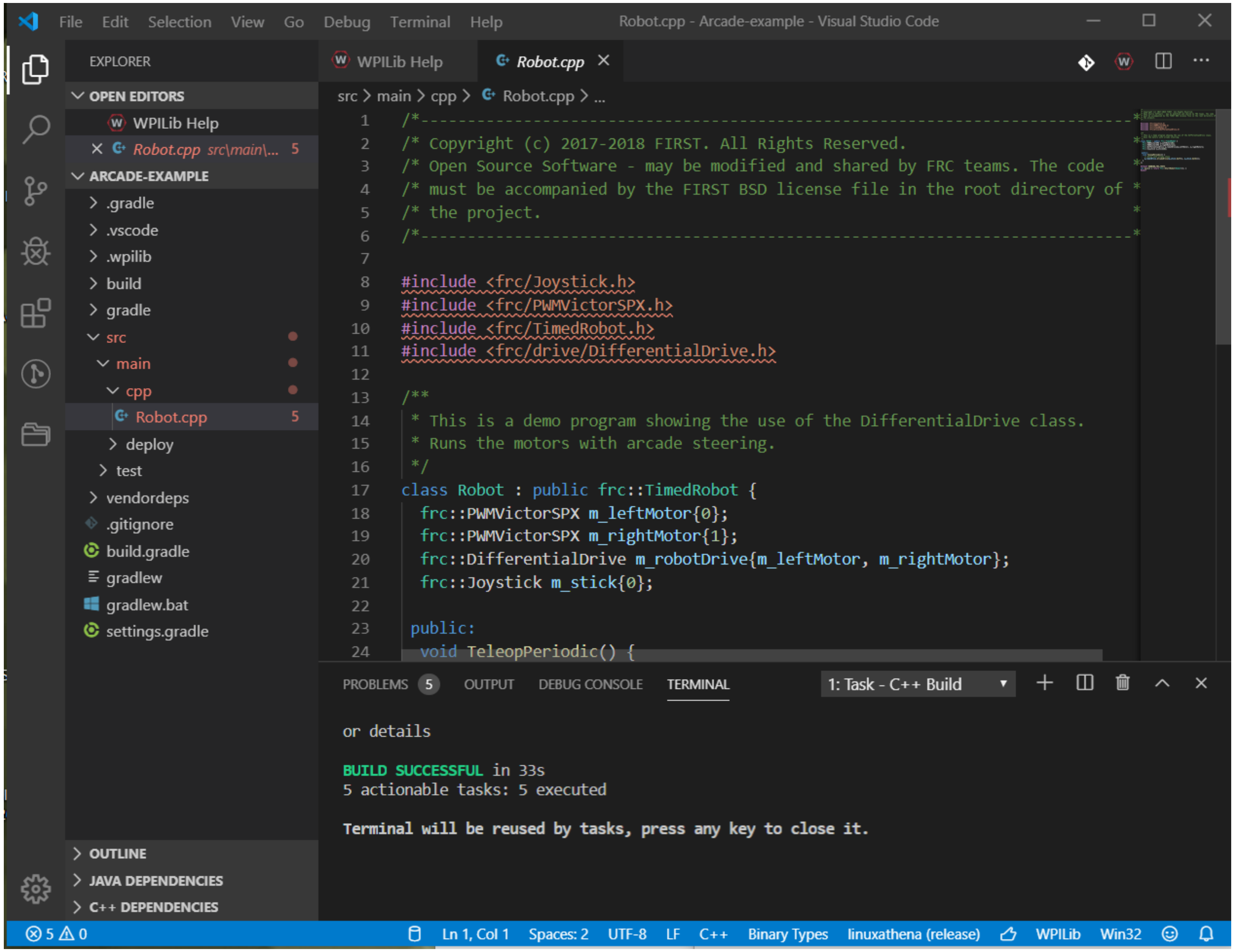
Arcade-example

Enter a team number

116

Generate Project

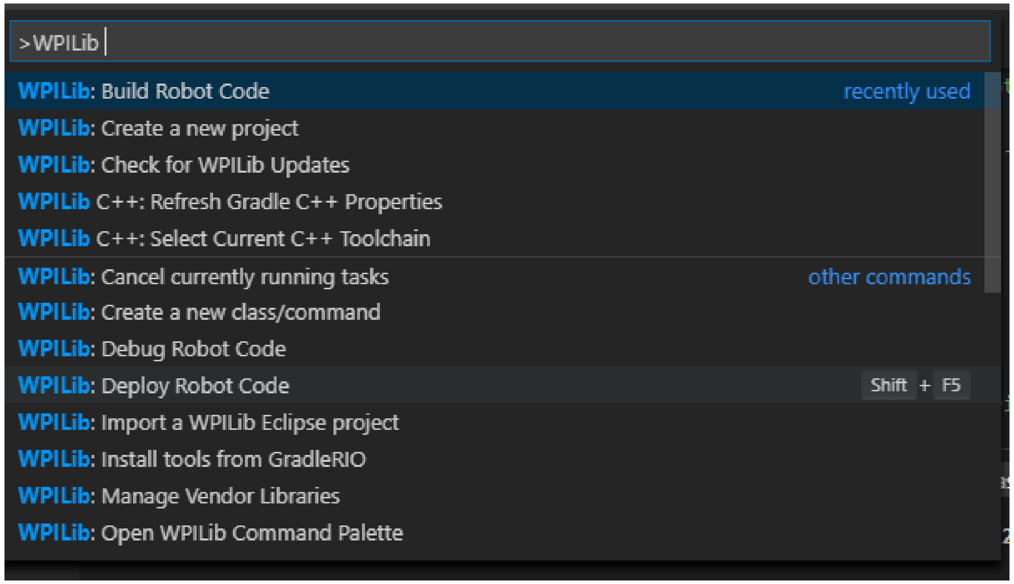
Create a Project #3



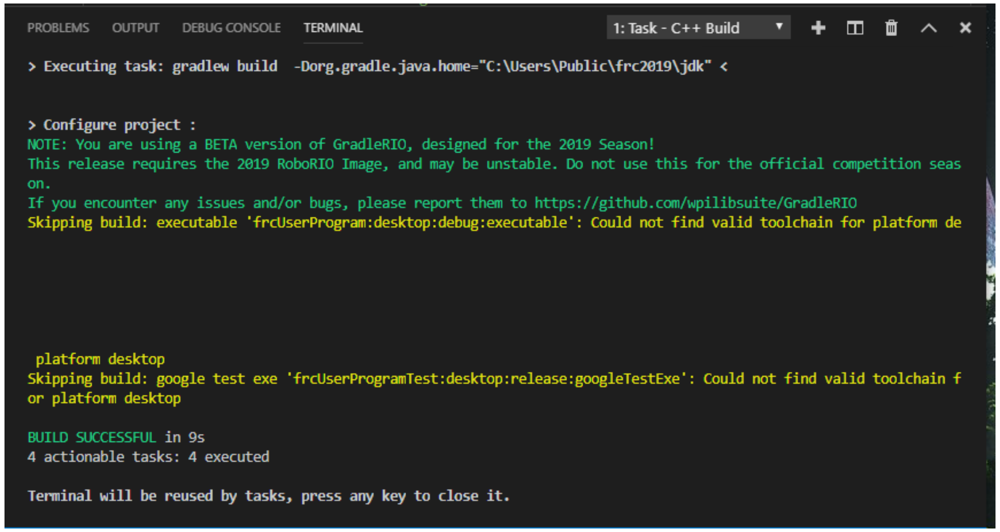
```cpp
/*----------------------------------------------------------------------------*/
/* Copyright (c) 2017-2018 FIRST. All Rights Reserved.                        */
/* Open Source Software - may be modified and shared by FRC teams. The code   */
/* must be accompanied by the FIRST BSD license file in the root directory of */
/* the project.                                                               */
/*----------------------------------------------------------------------------*/

#include <frc/Joystick.h>
#include <frc/PWMVictorSPX.h>
#include <frc/TimedRobot.h>
#include <frc/drive/DifferentialDrive.h>

/**
 * This is a demo program showing the use of the DifferentialDrive class.
 * Runs the motors with arcade steering.
 */
class Robot : public frc::TimedRobot {
  frc::PWMVictorSPX m_leftMotor{0};
  frc::PWMVictorSPX m_rightMotor{1};
  frc::DifferentialDrive m_robotDrive{m_leftMotor, m_rightMotor};
  frc::Joystick m_stick{0};

 public:
  void TeleopPeriodic() {
```

```
or details

BUILD SUCCESSFUL in 33s
5 actionable tasks: 5 executed

Terminal will be reused by tasks, press any key to close it.
```

Build and Deploy



```
>WPILib
WPILib: Build Robot Code                                   recently used
WPILib: Create a new project
WPILib: Check for WPILib Updates
WPILib C++: Refresh Gradle C++ Properties
WPILib C++: Select Current C++ Toolchain
WPILib: Cancel currently running tasks                     other commands
WPILib: Create a new class/command
WPILib: Debug Robot Code
WPILib: Deploy Robot Code                                  Shift + F5
WPILib: Import a WPILib Eclipse project
WPILib: Install tools from GradleRIO
WPILib: Manage Vendor Libraries
WPILib: Open WPILib Command Palette
```

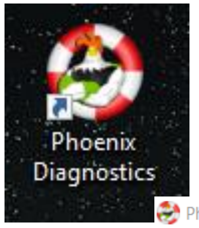


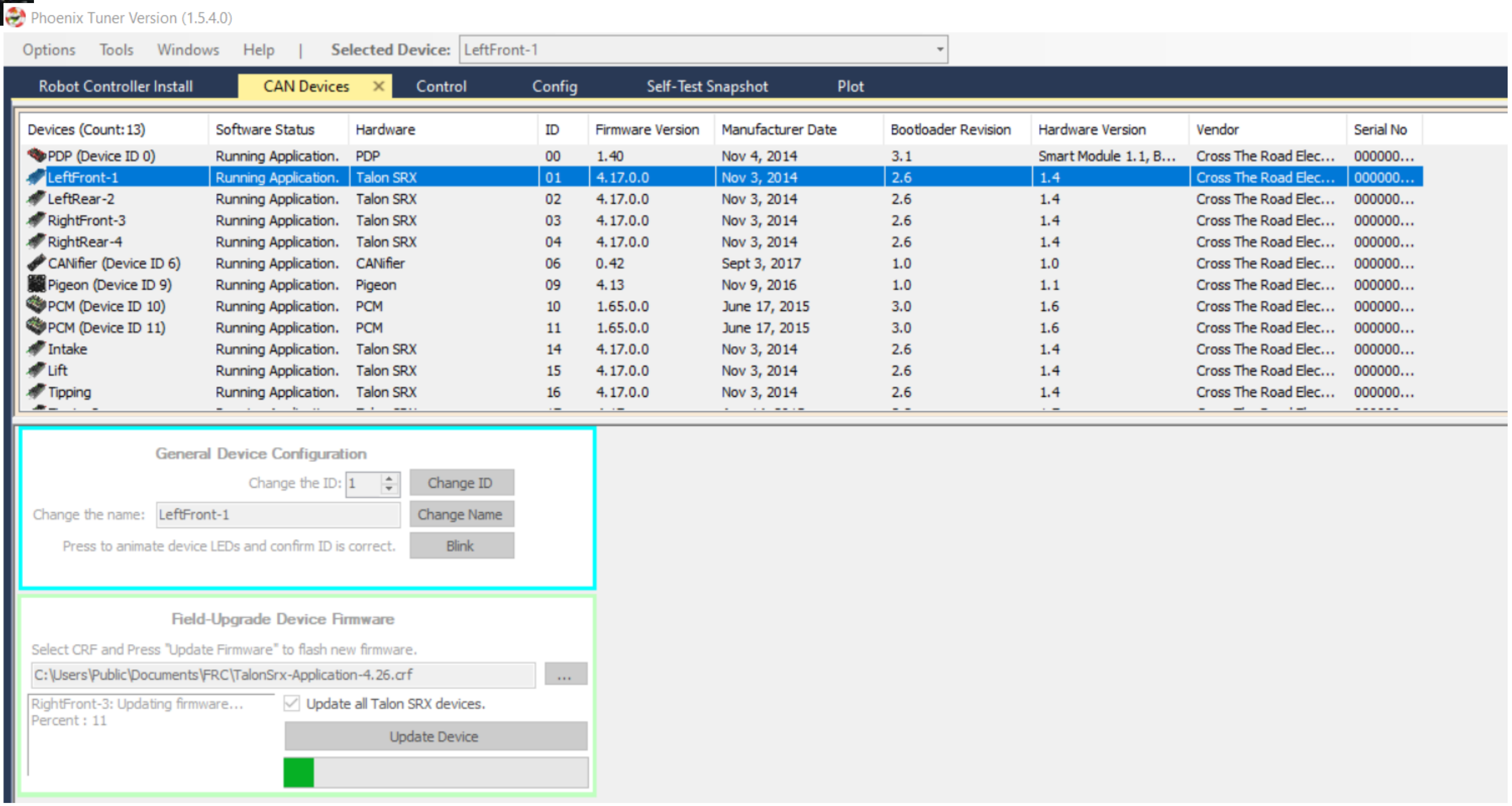
```
> Executing task: gradlew build  -Dorg.gradle.java.home="C:\Users\Public\frc2019\jdk" <


> Configure project :
NOTE: You are using a BETA version of GradleRIO, designed for the 2019 Season!
This release requires the 2019 RoboRIO Image, and may be unstable. Do not use this for the official competition seas
on.
If you encounter any issues and/or bugs, please report them to https://github.com/wpilibsuite/GradleRIO
Skipping build: executable 'frcUserProgram:desktop:debug:executable': Could not find valid toolchain for platform de




 platform desktop
Skipping build: google test exe 'frcUserProgramTest:desktop:release:googleTestExe': Could not find valid toolchain f
or platform desktop

BUILD SUCCESSFUL in 9s
4 actionable tasks: 4 executed

Terminal will be reused by tasks, press any key to close it.
```

Install the Third-Party Libraries

- The CTRE, REV and Kauaii Labs libraries are unbundled from the WPILib development environment
  - You will need to install these libraries separately into the VSCode workspace
- CAN bus is a feature now of several FRC-legal motor controllers
- For CTRE/VexPro motor controllers, you will need to install the CTRE Phoenix framework onto your platform
  - The Phoenix Diagnostics application will enable you to update your CAN firmware for the PDP, PCM, Talon SRX and Victor SPX devices
- You'll need to add the libraries and header files to the search path of your project using the VSCode external library mechanism

Configure CAN Bus (CTRE)





Selected Device: LeftFront-1

Devices (Count: 13)	Software Status	Hardware	ID	Firmware Version	Manufacturer Date	Bootloader Revision	Hardware Version	Vendor	Serial No
PDP (Device ID 0)	Running Application.	PDP	00	1.40	Nov 4, 2014	3.1	Smart Module 1.1, B...	Cross The Road Elec...	000000...
LeftFront-1	Running Application.	Talon SRX	01	4.17.0.0	Nov 3, 2014	2.6	1.4	Cross The Road Elec...	000000...
LeftRear-2	Running Application.	Talon SRX	02	4.17.0.0	Nov 3, 2014	2.6	1.4	Cross The Road Elec...	000000...
RightFront-3	Running Application.	Talon SRX	03	4.17.0.0	Nov 3, 2014	2.6	1.4	Cross The Road Elec...	000000...
RightRear-4	Running Application.	Talon SRX	04	4.17.0.0	Nov 3, 2014	2.6	1.4	Cross The Road Elec...	000000...
CANifier (Device ID 6)	Running Application.	CANifier	06	0.42	Sept 3, 2017	1.0	1.0	Cross The Road Elec...	000000...
Pigeon (Device ID 9)	Running Application.	Pigeon	09	4.13	Nov 9, 2016	1.0	1.1	Cross The Road Elec...	000000...
PCM (Device ID 10)	Running Application.	PCM	10	1.65.0.0	June 17, 2015	3.0	1.6	Cross The Road Elec...	000000...
PCM (Device ID 11)	Running Application.	PCM	11	1.65.0.0	June 17, 2015	3.0	1.6	Cross The Road Elec...	000000...
Intake	Running Application.	Talon SRX	14	4.17.0.0	Nov 3, 2014	2.6	1.4	Cross The Road Elec...	000000...
Lift	Running Application.	Talon SRX	15	4.17.0.0	Nov 3, 2014	2.6	1.4	Cross The Road Elec...	000000...
Tipping	Running Application.	Talon SRX	16	4.17.0.0	Nov 3, 2014	2.6	1.4	Cross The Road Elec...	000000...

General Device Configuration

Change the ID: 1 [Change ID]

Change the name: LeftFront-1 [Change Name]

Press to animate device LEDs and confirm ID is correct. [Blink]

Field-Upgrade Device Firmware

Select CRF and Press "Update Firmware" to flash new firmware.

C:\Users\Public\Documents\FRC\TalonSrx-Application-4.26.crf [...]

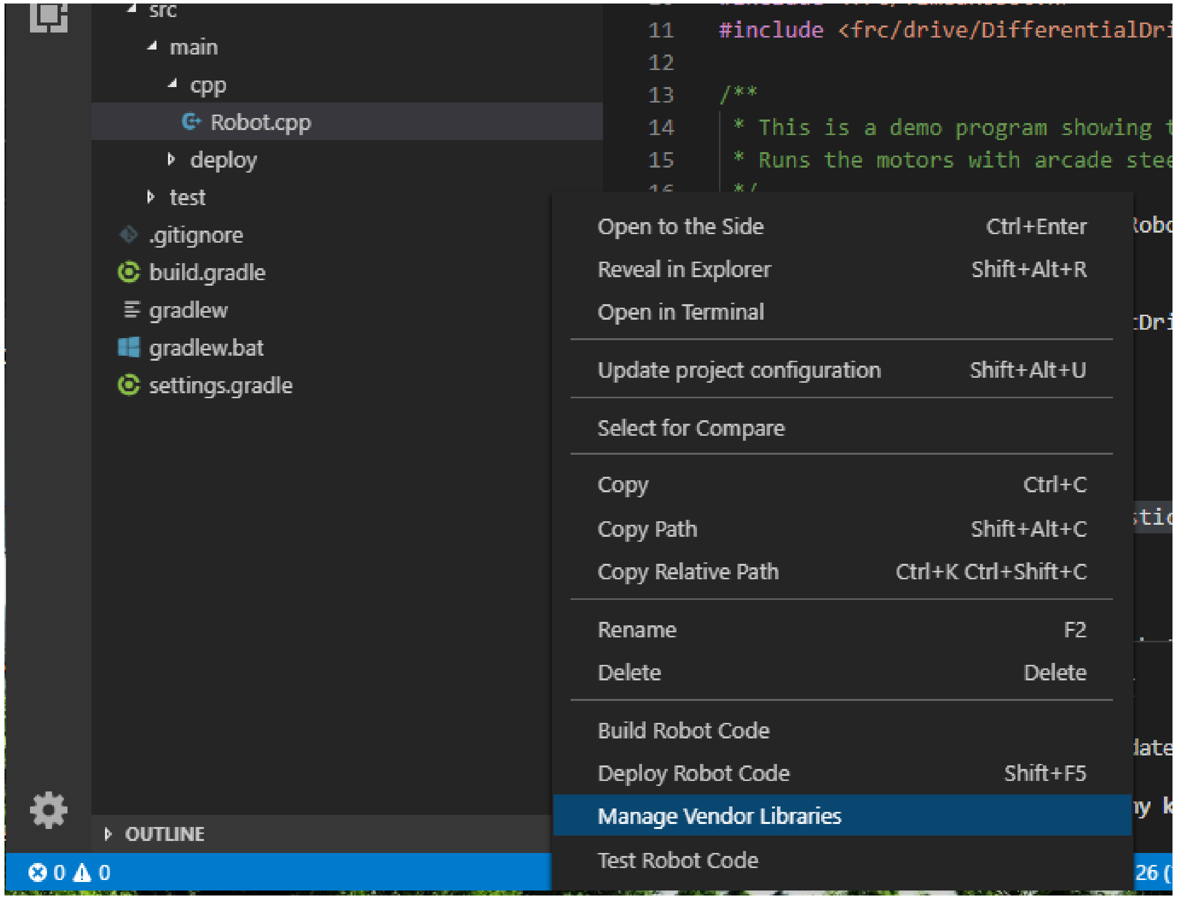
RightFront-3: Updating firmware...
Percent : 11

[✓] Update all Talon SRX devices.

[Update Device]

Install 3rd-Party Library into Your Project

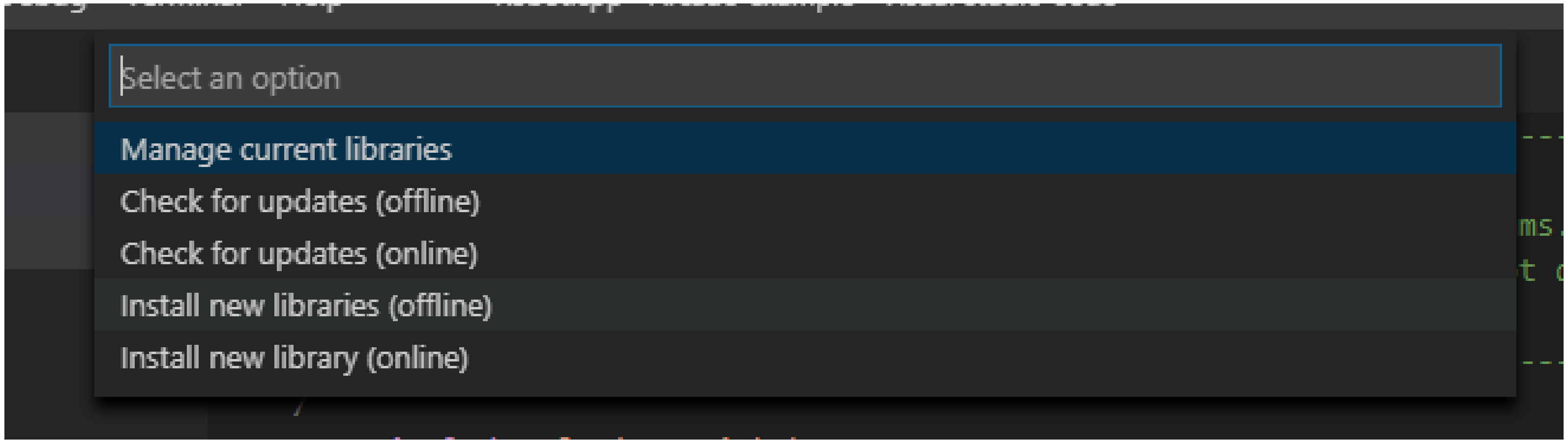
- Before you can use the 3rd-party libraries, you'll need to import them into your project



```
#include <frc/drive/DifferentialDri

/**
 * This is a demo program showing t
 * Runs the motors with arcade stee
```

3rd-Party #2

- Select the “Install new libraries (offline)” and then select the library you want to install



Select an option

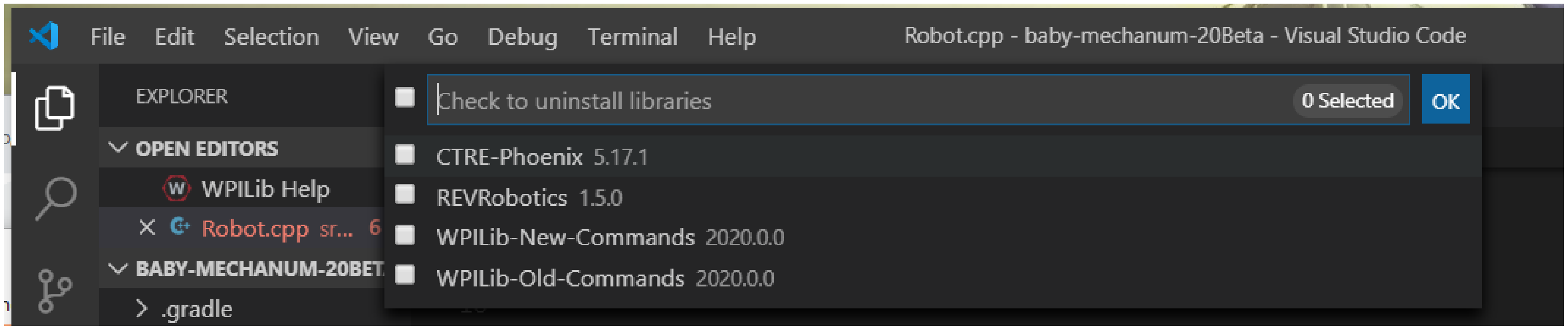
Manage current libraries

Check for updates (offline)

Check for updates (online)

Install new libraries (offline)

Install new library (online)



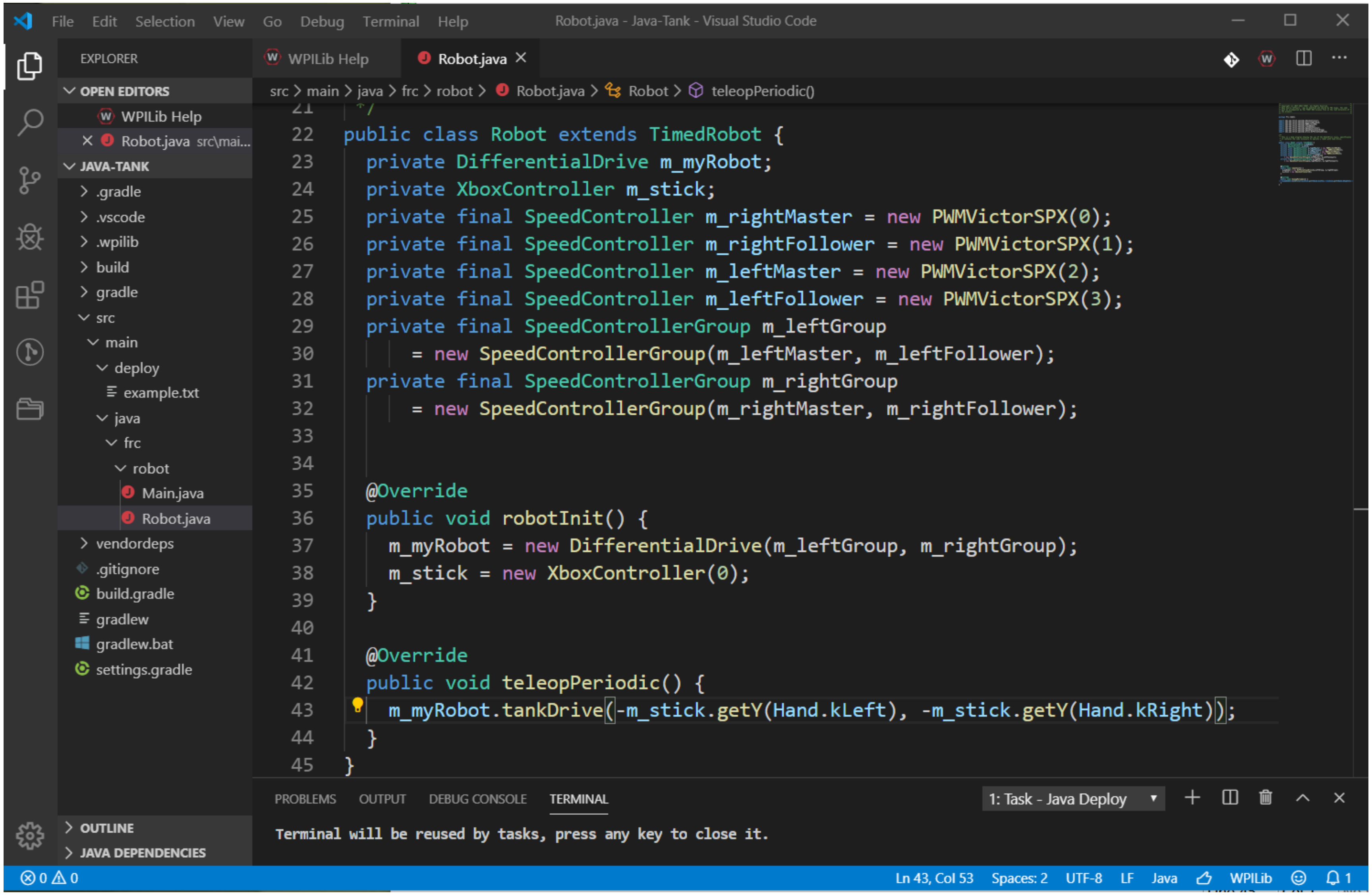
Check to uninstall libraries

- CTRE-Phoenix 5.17.1
- REVRobotics 1.5.0
- WPILib-New-Commands 2020.0.0
- WPILib-Old-Commands 2020.0.0

3rd-Party #3

- Once the library is installed in your project, you can start using the features it provides
- You'll need to make sure you've got the header files or imports listed
  - Or, the build will fail miserably
- Once built, you can deploy the 3rd-party goodness to the robot

Example Java Robot Program



```java
public class Robot extends TimedRobot {
  private DifferentialDrive m_myRobot;
  private XboxController m_stick;
  private final SpeedController m_rightMaster = new PWMVictorSPX(0);
  private final SpeedController m_rightFollower = new PWMVictorSPX(1);
  private final SpeedController m_leftMaster = new PWMVictorSPX(2);
  private final SpeedController m_leftFollower = new PWMVictorSPX(3);
  private final SpeedControllerGroup m_leftGroup
      = new SpeedControllerGroup(m_leftMaster, m_leftFollower);
  private final SpeedControllerGroup m_rightGroup
      = new SpeedControllerGroup(m_rightMaster, m_rightFollower);


  @Override
  public void robotInit() {
    m_myRobot = new DifferentialDrive(m_leftGroup, m_rightGroup);
    m_stick = new XboxController(0);
  }

  @Override
  public void teleopPeriodic() {
    m_myRobot.tankDrive(-m_stick.getY(Hand.kLeft), -m_stick.getY(Hand.kRight));
  }
}
```

```
Terminal will be reused by tasks, press any key to close it.
```

Resources

- Chief Delphi
  - http://www.chiefdelphi.com
- FIRST forums
  - http://forums.usfirst.org
- NI Community Forums
  - http://ni.com/FIRST
- WPI / *FIRST* NSF Community site (ThinkTank)
- These sites are monitored by members of:
  - WPI
  - NI
  - *FIRST*
- All source code available for team<->team assistance
- Phone support through NI
  - 866-511-6285 (1PM-7PM CST, M-F) ?

Summary

- C/C++ can be very challenging to new developers
  - C/C++ is similar enough to Java that Java developers can adapt to it quickly
    - However, pointers will require some explaining
  - Performance and fine-grain control are the biggest advantages to using C/C++
- Java has a lot of support within the FIRST community and many school systems
  - Being on the AP CS exam encourages schools to teach it
  - Java is also used in the new FTC development environment
    - Although the Java VM is slightly different for Android
- WPILib class libraries have equivalent capability between C++ and Java versions
- Java and C++ are syntactically very similar
  - You could start with one and then switch without too much trouble