FIRST Tech (FTC) Robotics: New Programming Platform Workshop

FTC Team 9901

Techie Titans

Aug 14, 2016

FTC Team 9901

Techie Titans - About us

- ➔ We are an **FTC** team with 7 team members, from grade 7 to 11 – from 6 different schools
  - ◆ John Champe High school (Chantilly)
  - ◆ Thomas Jefferson High School
  - ◆ Lunsford Middle school (Chantilly)
  - ◆ Mercer Middle school (Chantilly)
  - ◆ Eagle Ridge Middle school (Ashburn)
  - ◆ Frost Middle School (Fairfax)
- ➔ We are part of **Nova-Labs Robotics**, a non-profit and well-known Maker-Space in this area

FTC Team 9901

Techie Titans - About us

Achievements last season:

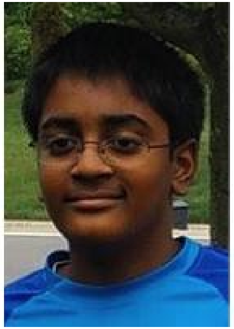
- Qualified to East Super-Regional Championship
- Qualified to VA and MD State Tournament
- Awards -
    - Inspire Award 2nd
    - Winning Alliance
    - Control Award
    - Think Award
    - Innovate Award 2nd
    - Connect Award 3rd

FTC Team 9901

About us - team members

Jasmine, 11th

Sonika, 9th


Vinay, 8th

Ridha, 8th

Faraaz, 8th

Prashanth, 7th

Arsh, 7th

?

FTC Team 9901

About us - Coaches and Mentors

Coaches

Hossain (Tauhid) Rahman

https://www.linkedin.com/in/tauhid-hossain-rahman-pmp-pmi-acp-csm/

Farzana Afrin

https://www.linkedin.com/in/farzana-afrin

Mentors

M. Ruhul Chowdhury

https://www.linkedin.com/in/ruhul-chowdhury

Jagan Manickam

https://www.linkedin.com/in/jagan-manickam

Agenda

Part 1:

- Brief overview of New platform
- Software development Environment
  - ZTE Speed Phone
  - FTC Apps (Driver Station and Robot Controller)
  - Android Studio
  - FTC SDK
  - Event driven and linear programming model
- Example Op Modes
- Build, Deployment and Drive!!
- Test and Debug
- Tele Op

Agenda

Part 2:

- Resources and helpful links
    - Starting a new team
    - How to Get Organized
- Award categories
- Engineering Notebook requirements
- Lesson learned
- Q&A

Non Goals

- App Inventor
- Compatibility/reusability with legacy HW
- Advanced Topics (If time permits)
  - Parallel threads
  - Motor Calibration, Stalling
  - Autonomous techniques (i.e. Line tracker, IR Beacon follower)

Attendee Poll

- Experience: Rookie Team? 1-2 years? More than 2 year?`
- Received the Kit? Tetrix? Matrix?
- Installed Android Studio? Built an app?
- Installed the FTC App?
- Ran a OpMode?
- Wrote and tested an OpMode?

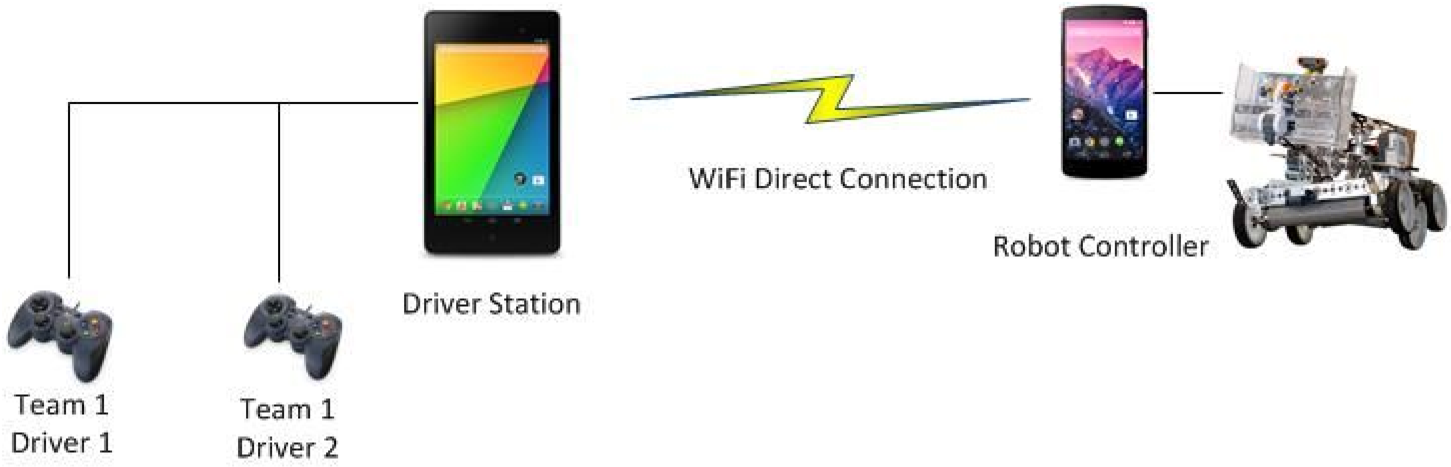
Part 1

FTC New Programming Platform

New Platform

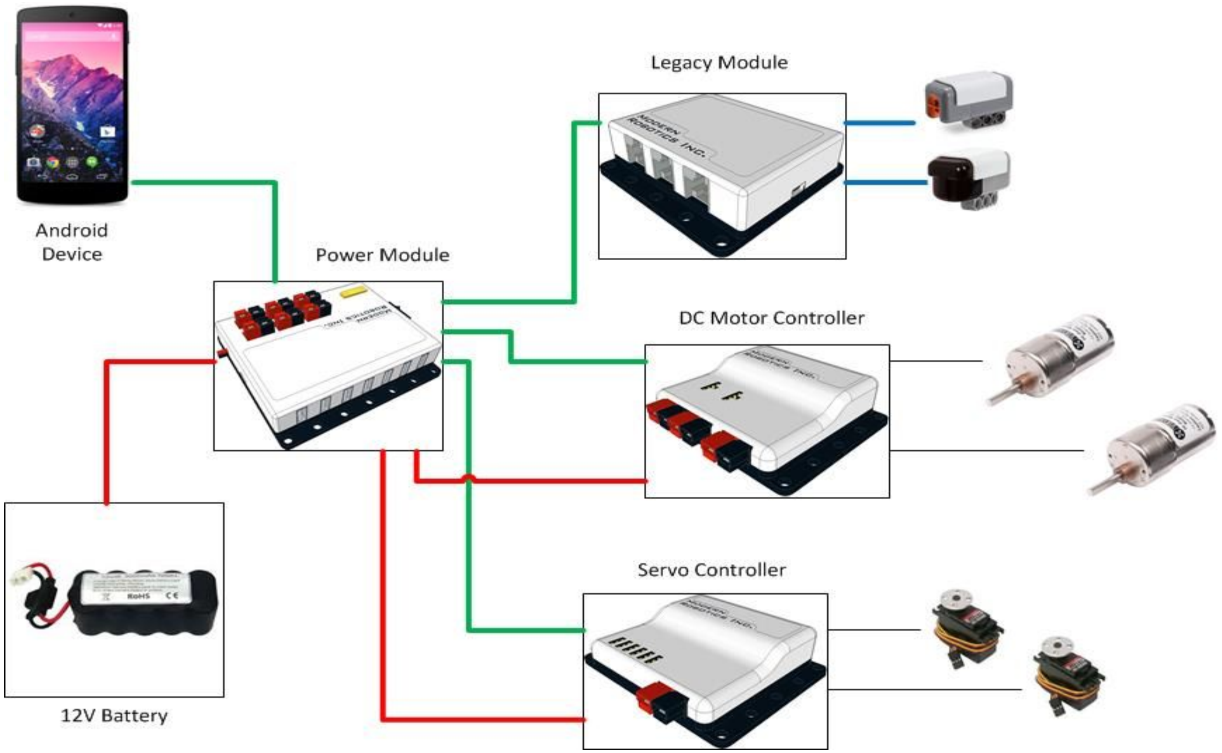
Overview

- → Based on the Android OS and Java
- → Uses two phones/tablets: Robot Controller and Driver's Station
- → Devices are connected over USB



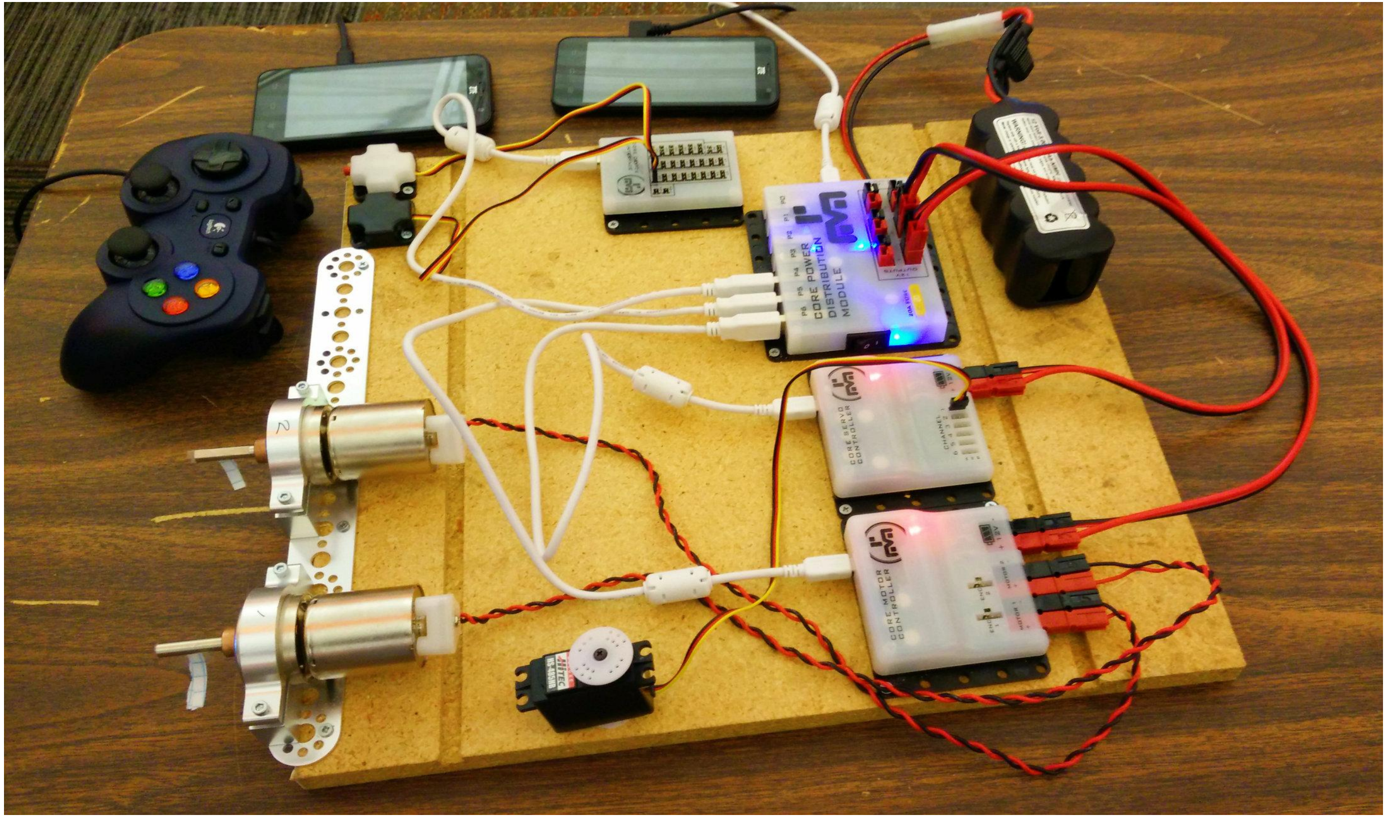
New Platform

Controllers



FTC Hardware

First learning setup -



FTC Robot

Last year's robot game

Our last season's robot run:

- https://www.youtube.com/watch?v=PBJYaj82Op0

- https://www.youtube.com/watch?v=WML5Yn8Rv80

Android Device

2015-2016 *FIRST*® Tech Challenge

Game Manual Part I

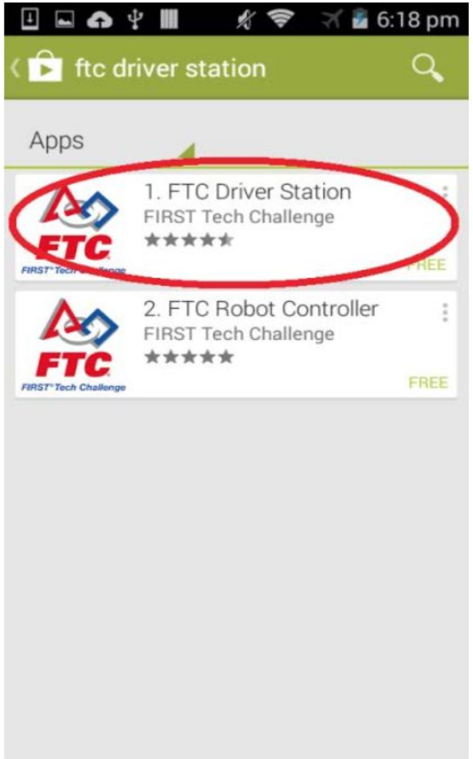
→ There are two (3) allowed Android devices that Teams will use to control their Robot:

- ZTE Speed.
- Motorola Moto G (3rd Generation)
- Google Nexus 5

FTC Apps

Overview

→ Download Driver and controller App from Google app store



6:18 pm

ftc driver station

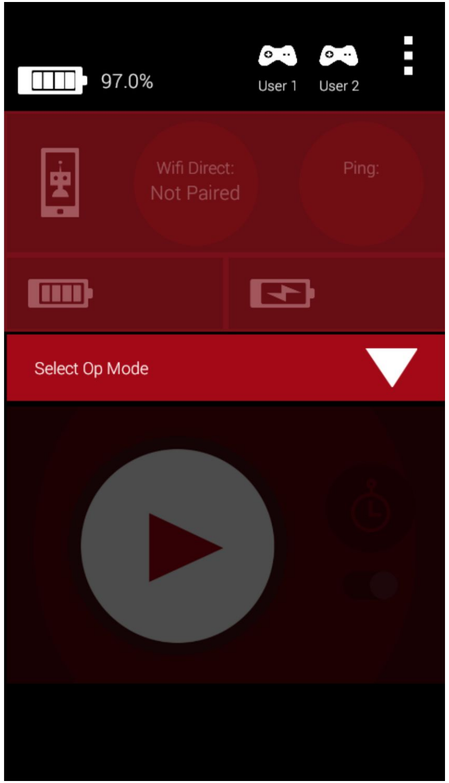
Apps

1. FTC Driver Station
FIRST Tech Challenge
★★★★★
FREE

2. FTC Robot Controller
FIRST Tech Challenge
★★★★★
FREE

FTC Apps

Driver Station



97.0%

User 1

User 2

Wifi Direct:
Not Paired

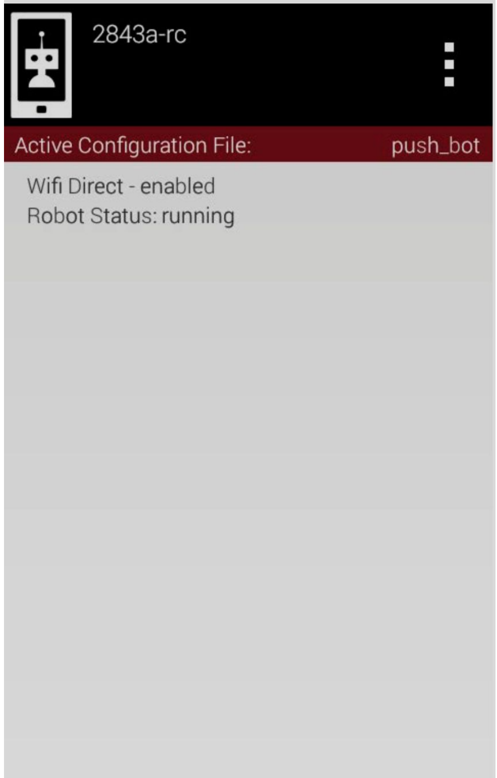
Ping:

Select Op Mode

- → Closed Source
- → No code is deployed
- → Front end app:
  - ◆ Program(OpMode) selection, Start and Stop
  - ◆ Gamepad is connected via micro USB
  - ◆ Telemetry (message from robot) is displayed
- → Setup for Wifi communication to Robot Controller

FTC Apps

Robot Controller



2843a-rc

Active Configuration File: push_bot

Wifi Direct - enabled
Robot Status: running

- Partial Open Source
- Part of Robot Assembly:
  - Connects to Power module/USB Hub
  - Integrates and executes programs (OpModes)
  - Broadcasts Telemetry messages to Driver station
- Setup and configuration of HW (i.e. Motors, sensors).

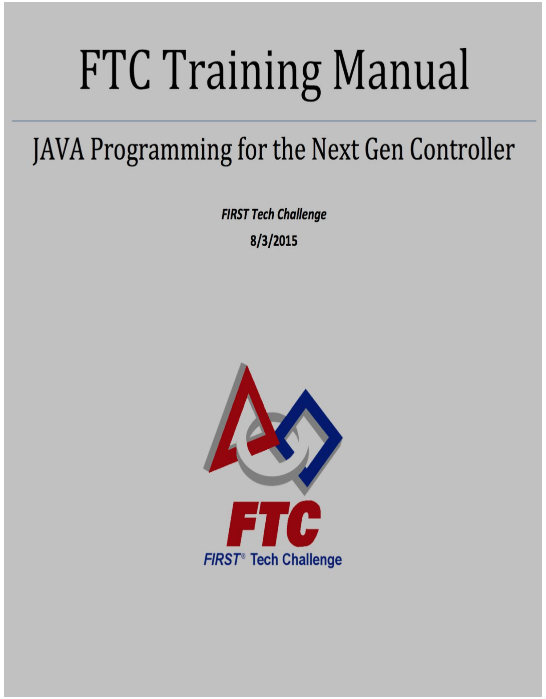
Android and Java

Our learning

- Early summer we spent some time learning Java Basics and Android Studio development environment.
- Our programming coach developed a lesson plan for the team members to follow.
- Leveraged online resources. We liked the following:
  - http://stackoverflow.com/
  - http://www.tutorialspoint.com/java/index.htm
- 2 projects we did:
  - Simple Calculator
  - Tic Tac Toe

Android and Java

Android Studio



FTC Training Manual

JAVA Programming for the Next Gen Controller

FIRST Tech Challenge

8/3/2015

FTC

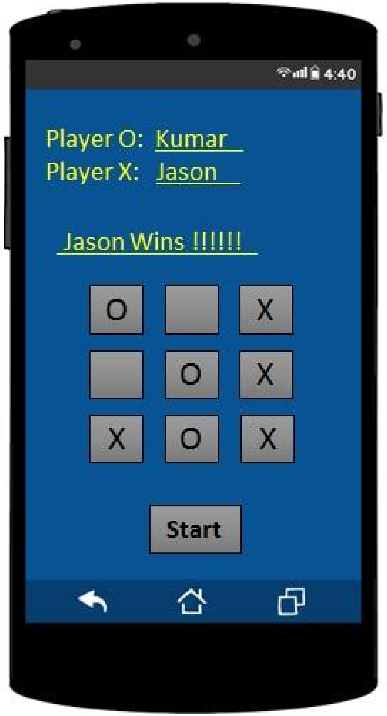
FIRST® Tech Challenge

- Team members installed Java and Android Studio using installation instructions in FTC Manual (Page 14 - 21)

- We deployed and tested Apps in:
    - Built in Emulator/Geny Motion
    - ZTE phone

Android and Java

Our learning: Tic Tac Toe



Player O: Kumar

Player X: Jason

Jason Wins !!!!!!

O		X
	O	X
X	O	X

→ 3 Team members and Coach came up with 4 different and working solutions!

→ We learned:

- Class - its structure and pieces
- Member Variables. Explore different types of Variables and where they are used.
- Methods - Its structure and pieces
- Java Data Types
- Controls (if, for, while, switch/case etc.)

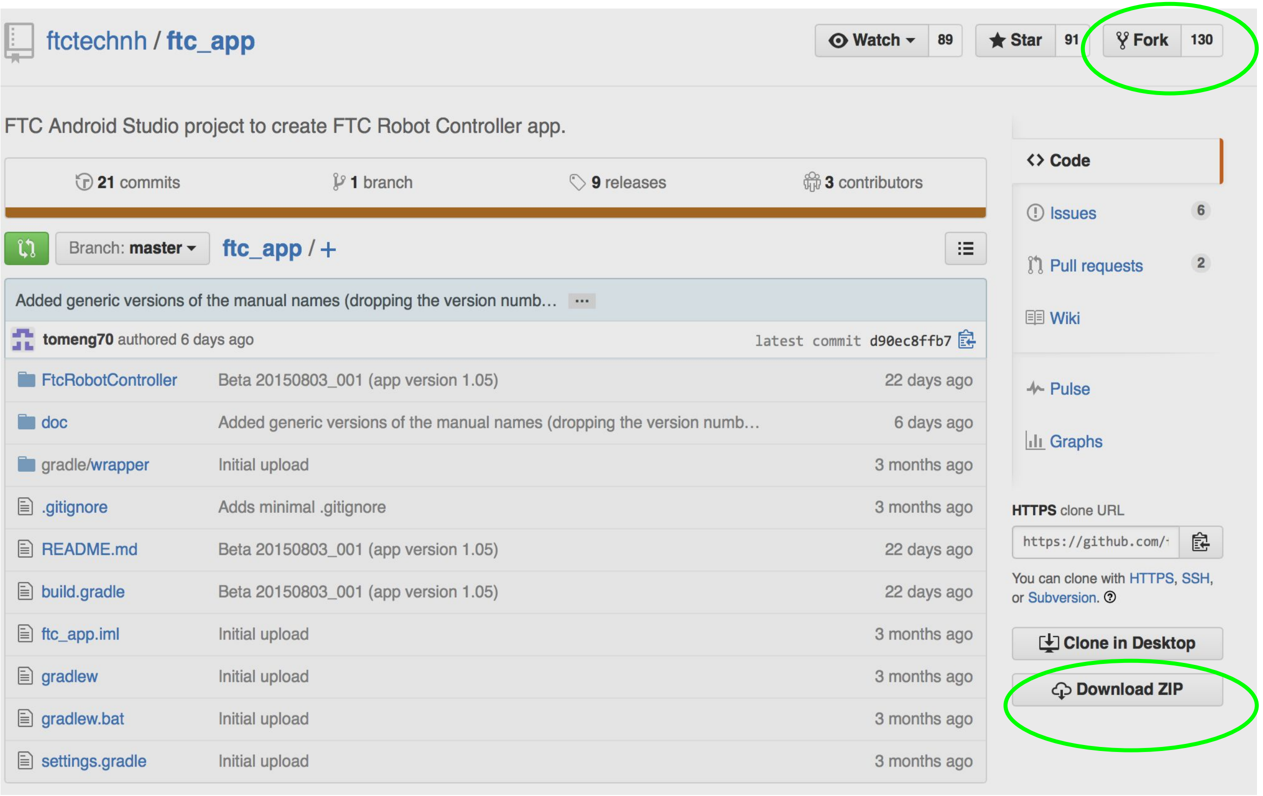
FTC Software

Basics

- ➔ Developed by QualComm for the FTC program
- ➔ Published in GitHub: https://github.com/ftctechnh/ftc_app
- ➔ Includes:
  - ◆ Robot Controller Source Code in an Android Studio Project that teams will use to create their own programs(Op Modes)
  - ◆ Sample programs (Op Modes)
  - ◆ Documentation

FTC Software

Download



FTC Android Studio project to create FTC Robot Controller app.

21 commits | 1 branch | 9 releases | 3 contributors

Added generic versions of the manual names (dropping the version numb… ···

tomeng70 authored 6 days ago — latest commit d90ec8ffb7

FtcRobotController	Beta 20150803_001 (app version 1.05)	22 days ago
doc	Added generic versions of the manual names (dropping the version numb…	6 days ago
gradle/wrapper	Initial upload	3 months ago
.gitignore	Adds minimal .gitignore	3 months ago
README.md	Beta 20150803_001 (app version 1.05)	22 days ago
build.gradle	Beta 20150803_001 (app version 1.05)	22 days ago
ftc_app.iml	Initial upload	3 months ago
gradlew	Initial upload	3 months ago
gradlew.bat	Initial upload	3 months ago
settings.gradle	Initial upload	3 months ago

HTTPS clone URL

`https://github.com/`

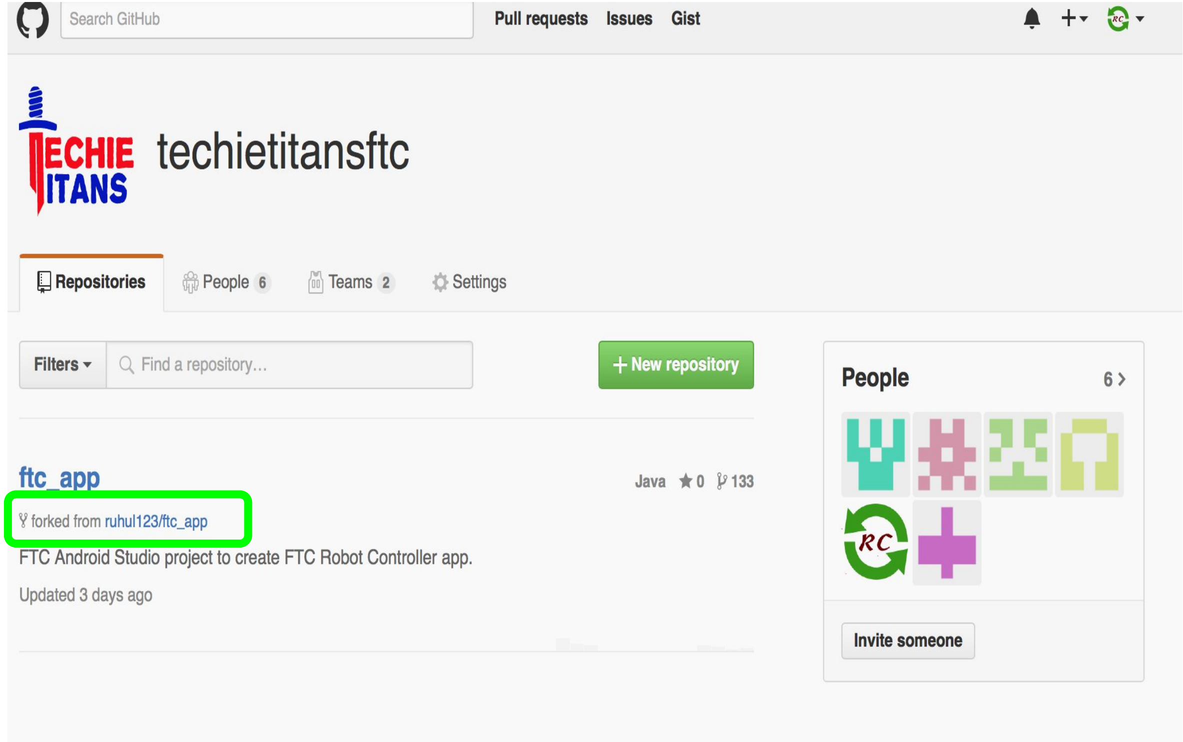
You can clone with HTTPS, SSH, or Subversion.

→ Can be downloaded as:

- .zip
- Forked (to your github) and cloned to desktop

FTC Software

GIT Management Model



techietitansftc

ftc_app

Java ★ 0 133

forked from ruhul123/ftc_app

FTC Android Studio project to create FTC Robot Controller app.

Updated 3 days ago

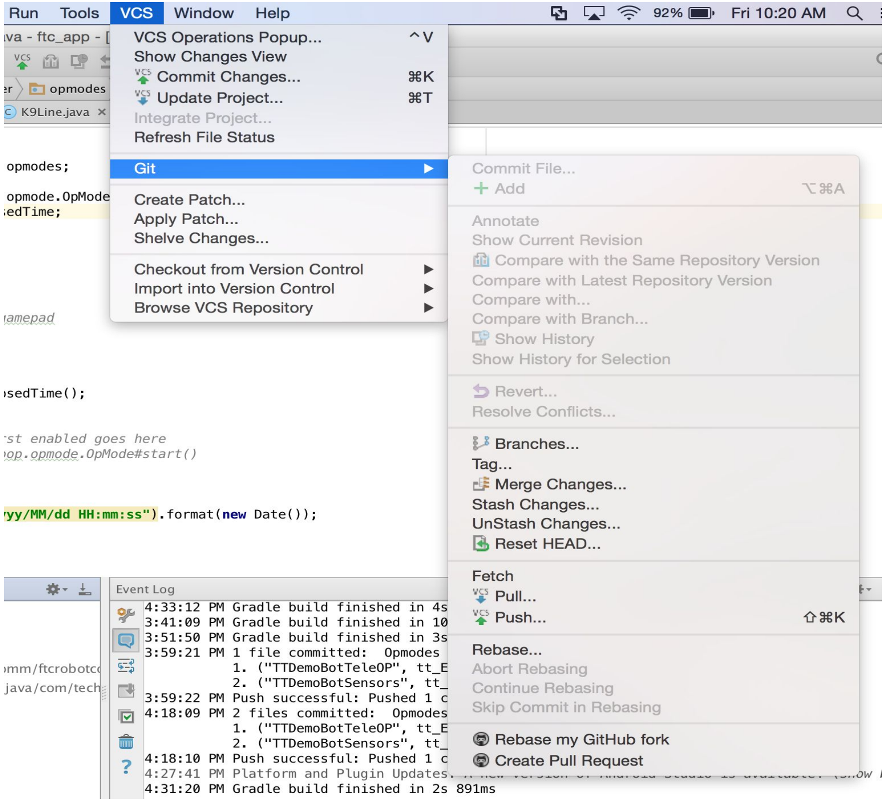
- Set a GROUP in github to manage and share code changes
- Kids picked it up quickly !!

NOVA Labs ROBOTICS

TECHIE TITANS

FTC Software

GIT Management Model



```
opmodes;

opmode.OpMode
sedTime;

amepad

sedTime();

st enabled goes here
op.opmode.OpMode#start()

yy/MM/dd HH:mm:ss").format(new Date());
```

```
4:33:12 PM Gradle build finished in 4s
3:41:09 PM Gradle build finished in 10
3:51:50 PM Gradle build finished in 3s
3:59:21 PM 1 file committed:  Opmodes
           1. ("TTDemoBotTeleOP", tt_E
           2. ("TTDemoBotSensors", tt_
3:59:22 PM Push successful: Pushed 1 c
4:18:09 PM 2 files committed:  Opmodes
           1. ("TTDemoBotTeleOP", tt_E
           2. ("TTDemoBotSensors", tt_
4:18:10 PM Push successful: Pushed 1 c
4:27:41 PM Platform and Plugin Updates:
4:31:20 PM Gradle build finished in 2s 891ms
```

- → All git(and github) features are in Android Studio
- → Google did a great job!!

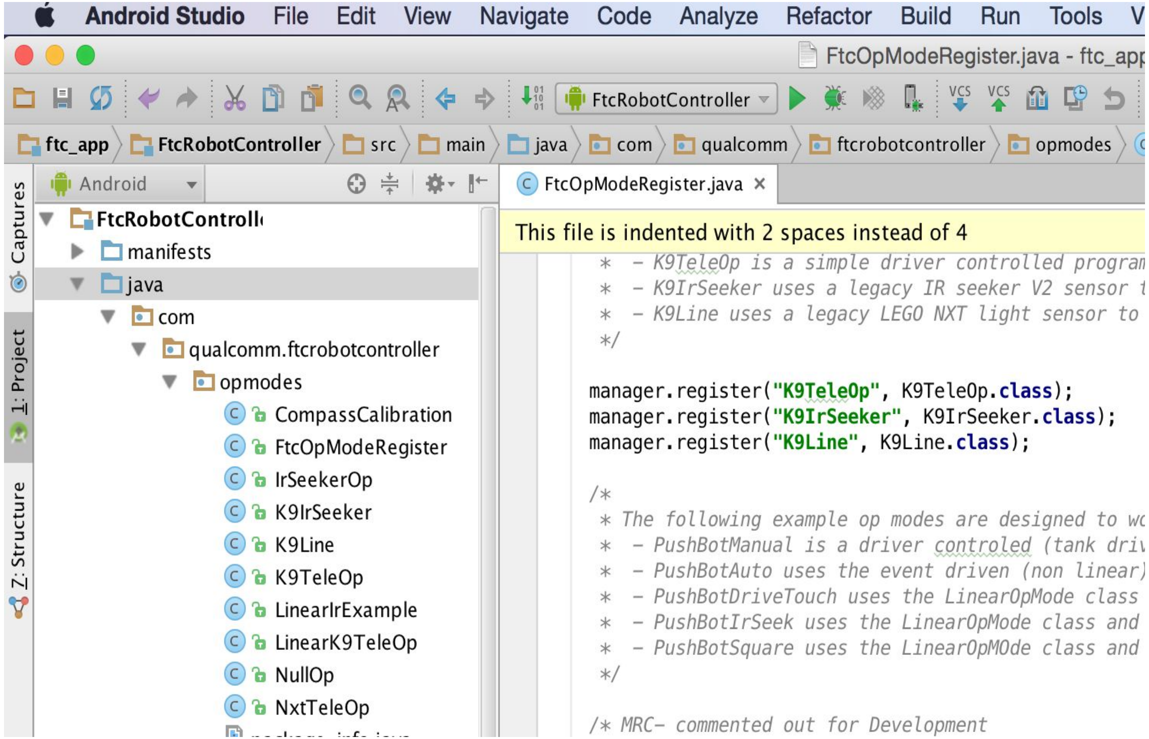
FTC Software

Op Mode (Your Custom Code)

- An Op Mode is what teams use in order to create custom behavior for their robot. It is a Java class.
- Op Modes are similar to the tele-op and autonomous programs that teams wrote for their LEGO NXT controllers
- During a match, Op Modes are executed on the robot controller, but are selected by the team from the driver station
- Two Types: Event based and Linear

FTC Software

Op Mode



This file is indented with 2 spaces instead of 4

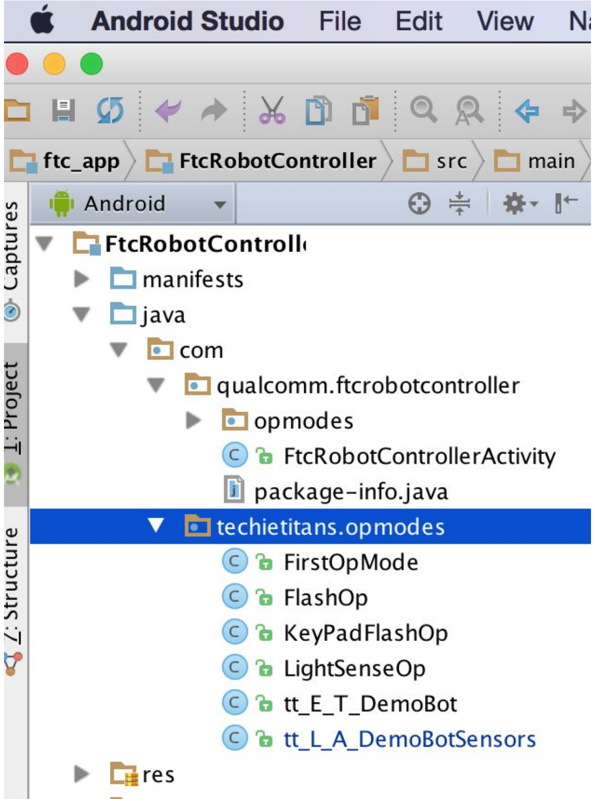
```
 *  - K9TeleOp is a simple driver controlled program
 *  - K9IrSeeker uses a legacy IR seeker V2 sensor t
 *  - K9Line uses a legacy LEGO NXT light sensor to
 */

manager.register("K9TeleOp", K9TeleOp.class);
manager.register("K9IrSeeker", K9IrSeeker.class);
manager.register("K9Line", K9Line.class);

/*
 * The following example op modes are designed to wo
 *  - PushBotManual is a driver controled (tank driv
 *  - PushBotAuto uses the event driven (non linear)
 *  - PushBotDriveTouch uses the LinearOpMode class
 *  - PushBotIrSeek uses the LinearOpMode class and
 *  - PushBotSquare uses the LinearOpMOde class and
 */

/* MRC- commented out for Development
```

➔ Your code (Op Modes) are integrated in same project with FTC code

➔ Only registered Op Modes will be available to Driver station

FTC Software

Code Separation



→ Our own namespace/package: `com.techietitans.opmodes`

→ Easier to manage future updates to FTC software

FTC Software

Event based Op Mode

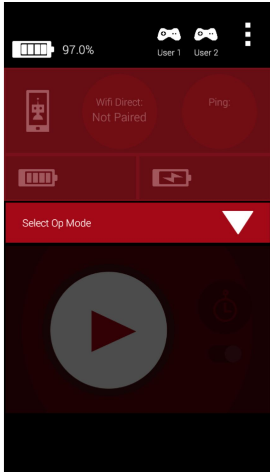
- Inherited from `OpModes` base class
- Loop() method is continuously executed until the program terminates
- HW communication (Sensor reads, Motor control) is done at the end of each loop() execution
- Useful for program with distinct states and state transition

FTC Software

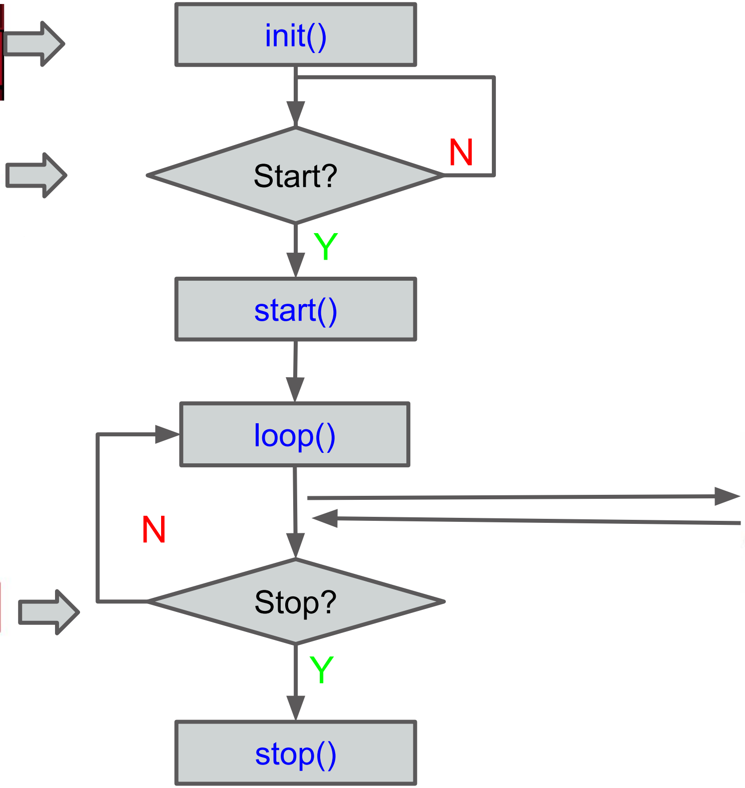
Event based Op Mode life cycle

Op Mode selected

Start Button click



Stop Button click



FTC Software

Linear Op Mode

- Commands are executed sequentially one after the other
- Similar to the model used to program a LEGO NXT with a tool like RobotC
- Inherited from `LinearOpModes` base class
- Can use blocking statements like `Thread.sleep()`
- HW communication (Sensor reads, Motor control) is done on demand, as needed
- Useful for Autonomous

FTC Software

Linear Op Mode life cycle

Op Mode selected

PushBotManual

Start Button click ▶

Stop Button click ■

```
public class tt_L_A_DemoBotSensors extends
LinearOpMode {
    @Override
    public void runOpMode()  throws
InterruptedException {
            // Wait for the start button to be pressed
            waitForStart();



            while(opModeIsActive())  {


            }
        }
}
```

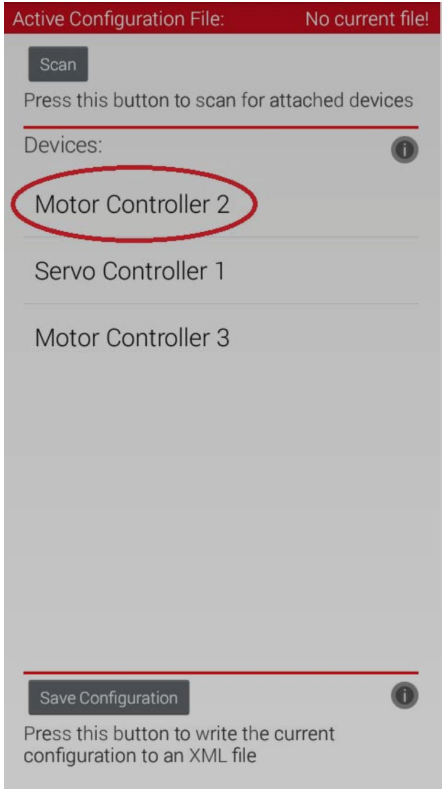
FTC Software

Op Mode

[Code DEMO - Vinay]

HW configuration

Robot Controller App



Active Configuration File: No current file!

Scan

Press this button to scan for attached devices

Devices:

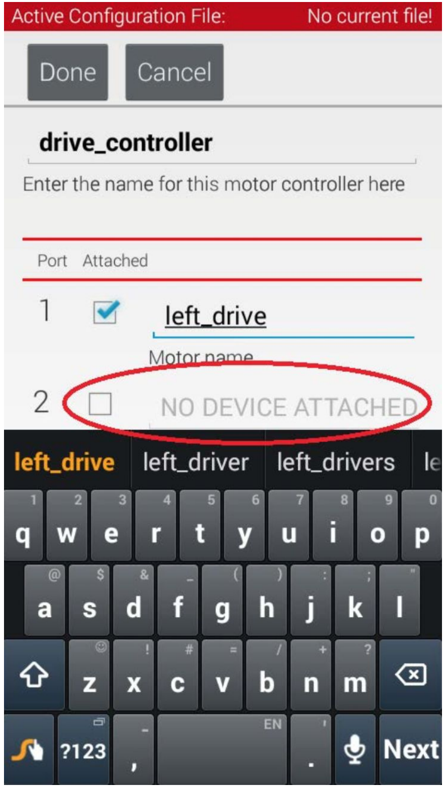
Motor Controller 2

Servo Controller 1

Motor Controller 3

Save Configuration

Press this button to write the current configuration to an XML file



Active Configuration File: No current file!

Done Cancel

drive_controller

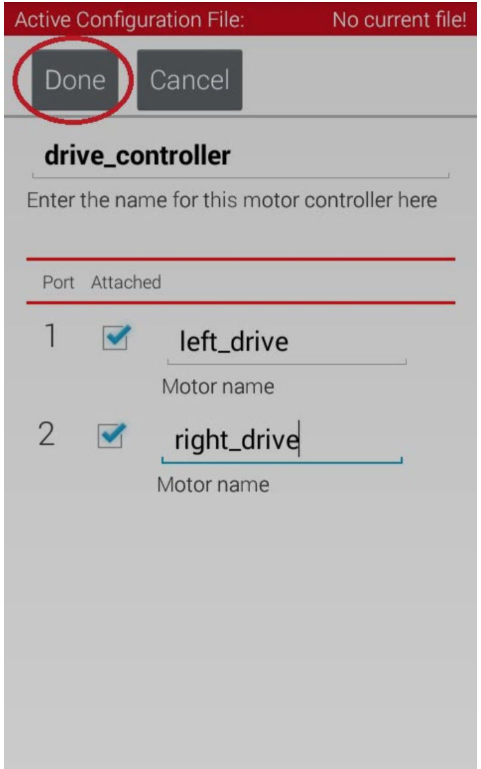
Enter the name for this motor controller here

Port	Attached	
1	☑	left_drive Motor name
2	□	NO DEVICE ATTACHED

- → **<u>Scan</u>** to auto discover connected controllers
- → Select a controller
- → Enter name for devices (motor, sensor)

HW configuration

Hardware map in the Op Mode

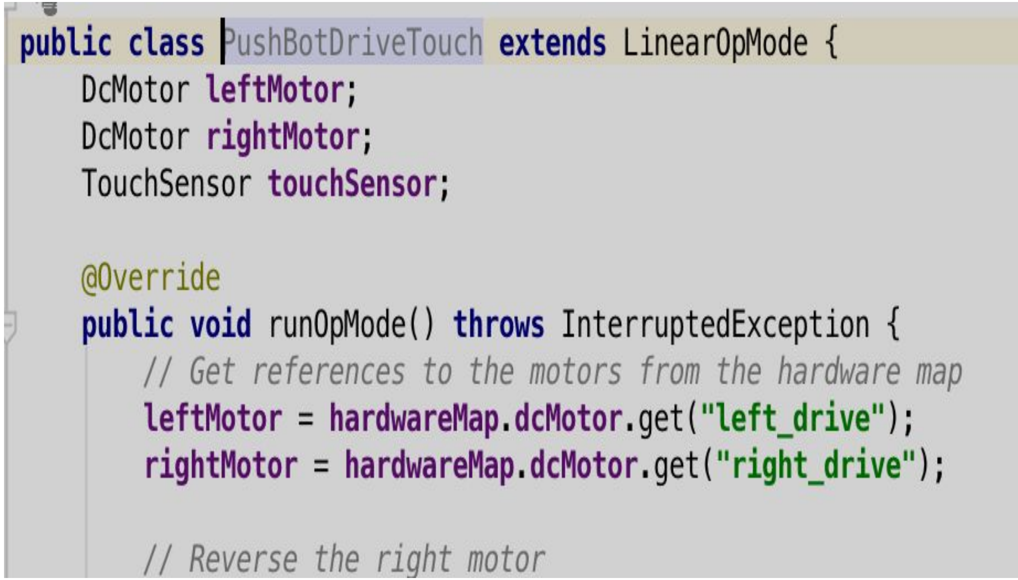


Active Configuration File: No current file!

drive_controller

Enter the name for this motor controller here

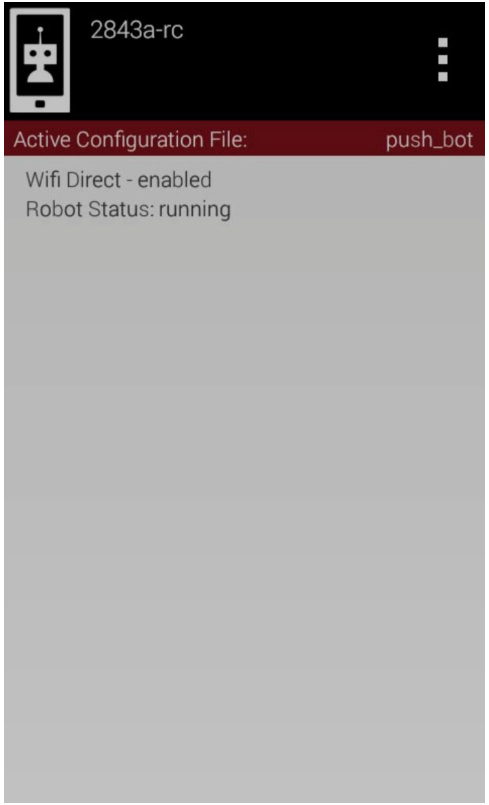
Port	Attached	
1	☑	left_drive Motor name
2	☑	right_drive Motor name



```
public class PushBotDriveTouch extends LinearOpMode {
    DcMotor leftMotor;
    DcMotor rightMotor;
    TouchSensor touchSensor;

    @Override
    public void runOpMode() throws InterruptedException {
        // Get references to the motors from the hardware map
        leftMotor = hardwareMap.dcMotor.get("left_drive");
        rightMotor = hardwareMap.dcMotor.get("right_drive");

        // Reverse the right motor
```

Execution Preparation

Robot Controller - Ready



2843a-rc

Active Configuration File: push_bot

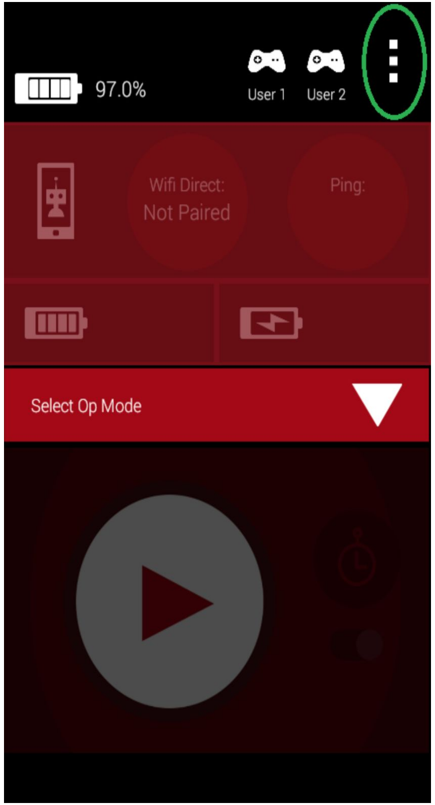
Wifi Direct - enabled

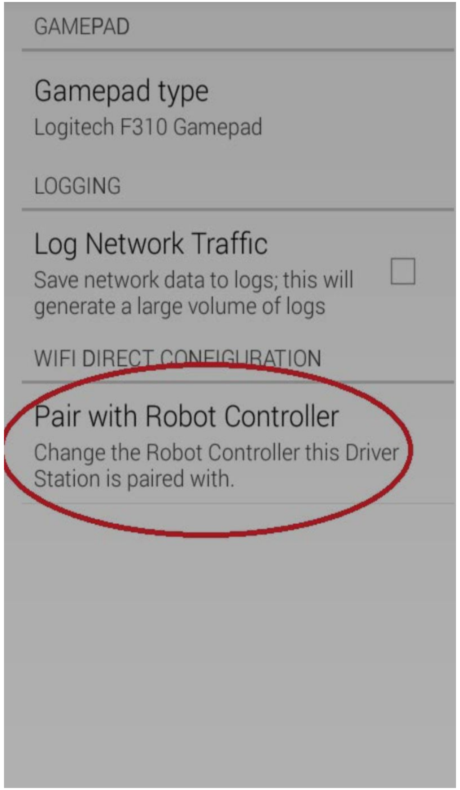
Robot Status: running

- → With our desired configuration file active - we are ready to start
- → We can have multiple configuration files, and swap between them as needed
- → An error here indicated that there is a mismatch between the file and attached HW

Execution Preparation

Driver Station - Pair Wifi





GAMEPAD

Gamepad type
Logitech F310 Gamepad

LOGGING

Log Network Traffic ☐
Save network data to logs; this will generate a large volume of logs

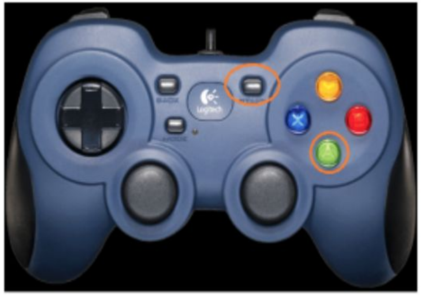
WIFI DIRECT CONFIGURATION

Pair with Robot Controller
Change the Robot Controller this Driver Station is paired with.

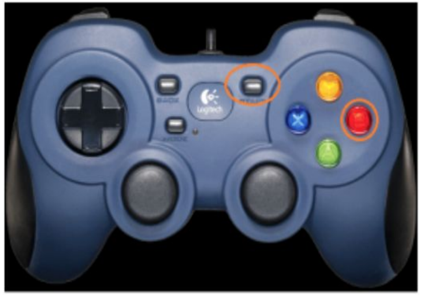
- → Initiate Pairing from FTC Driver Station app
- → Accept on Robot Controller

Execution Preparation

Driver Station - Joystick



Joystick 1



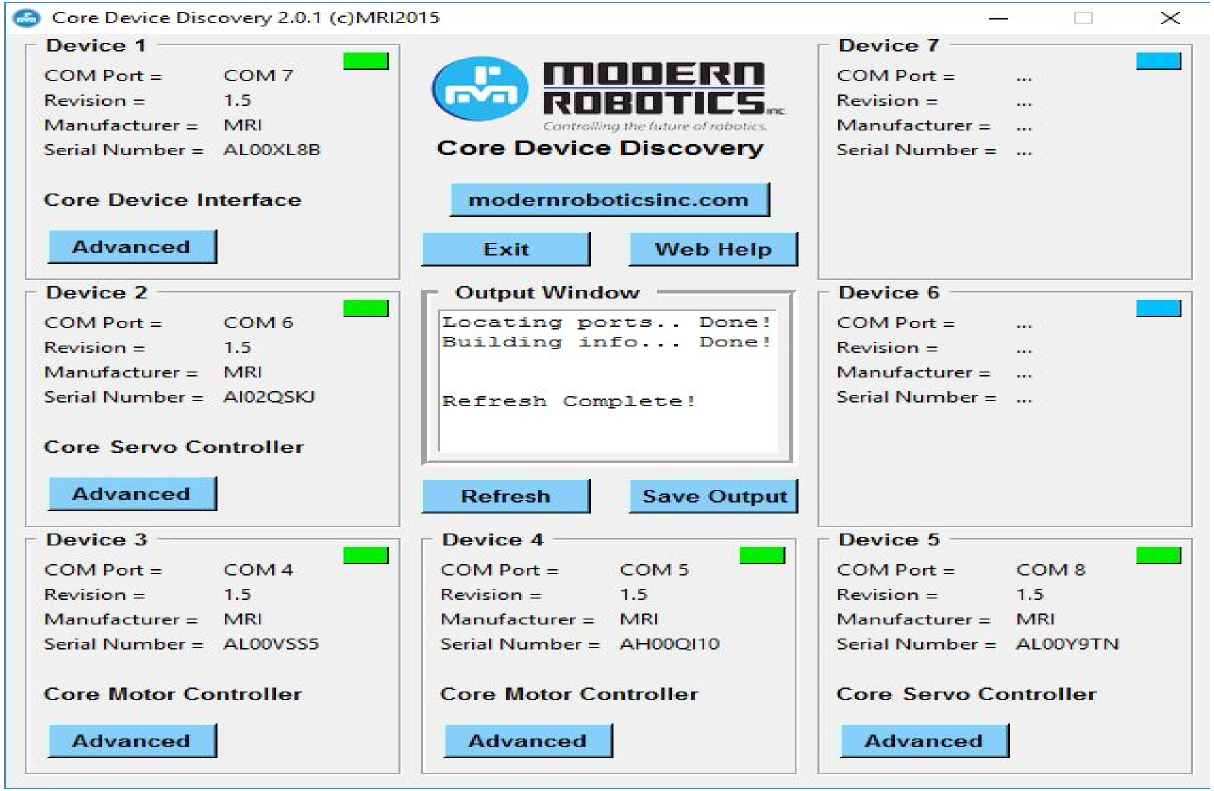
Joystick 2

- → Connect Gamepad/Joysticks to driver station ZTE phone using OTG USB Hub
- → Enable Driver 1 with START+A, Driver 2 with START+B

Test and Debug

Modern Robotics Core Device Discovery

This is the Modern Robotics Core Device Discovery. It is used to test the electronics.



Core Device Discovery 2.0.1 (c)MRI2015

Device 1 🟩
COM Port = COM 7
Revision = 1.5
Manufacturer = MRI
Serial Number = AL00XL8B
Core Device Interface

Device 2 🟩
COM Port = COM 6
Revision = 1.5
Manufacturer = MRI
Serial Number = AI02QSKJ
Core Servo Controller

Device 3 🟩
COM Port = COM 4
Revision = 1.5
Manufacturer = MRI
Serial Number = AL00VSS5
Core Motor Controller

Device 4 🟩
COM Port = COM 5
Revision = 1.5
Manufacturer = MRI
Serial Number = AH00QI10
Core Motor Controller

Device 5 🟩
COM Port = COM 8
Revision = 1.5
Manufacturer = MRI
Serial Number = AL00Y9TN
Core Servo Controller

Device 6 🟦
COM Port = ...
Revision = ...
Manufacturer = ...
Serial Number = ...

Device 7 🟦
COM Port = ...
Revision = ...
Manufacturer = ...
Serial Number = ...

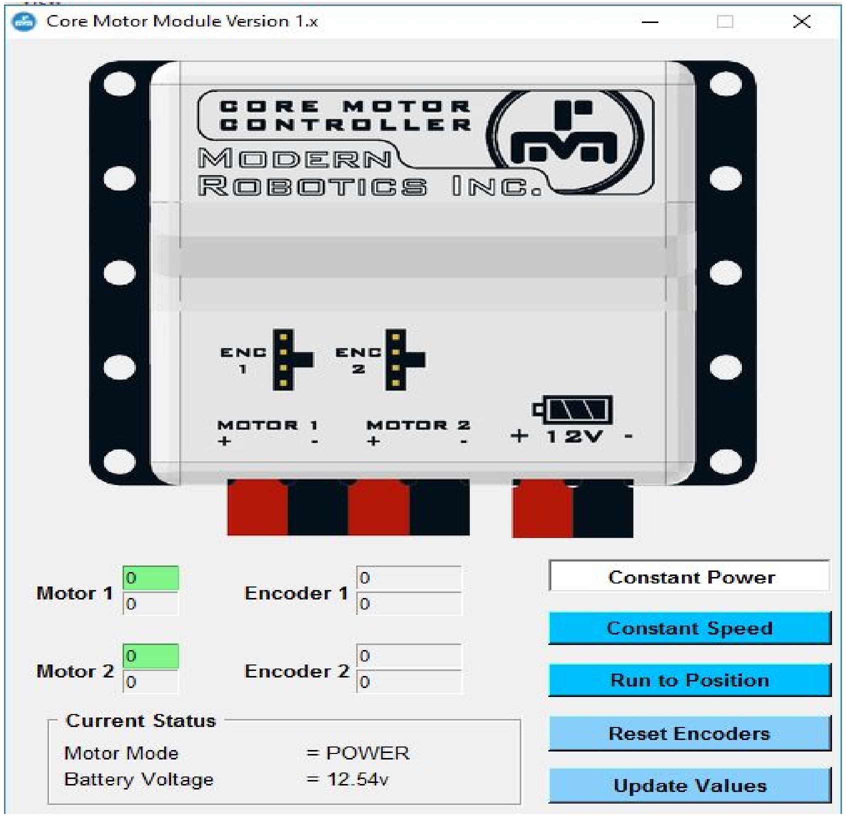
Output Window

```
Locating ports.. Done!
Building info... Done!


Refresh Complete!
```

Test and Debug

DC Motors

We have located a motor controller on the discovery tool and when you click on it this is what you will see.


Core Motor Module Version 1.x

Motor 1 0 0
Encoder 1 0 0
Motor 2 0 0
Encoder 2 0 0

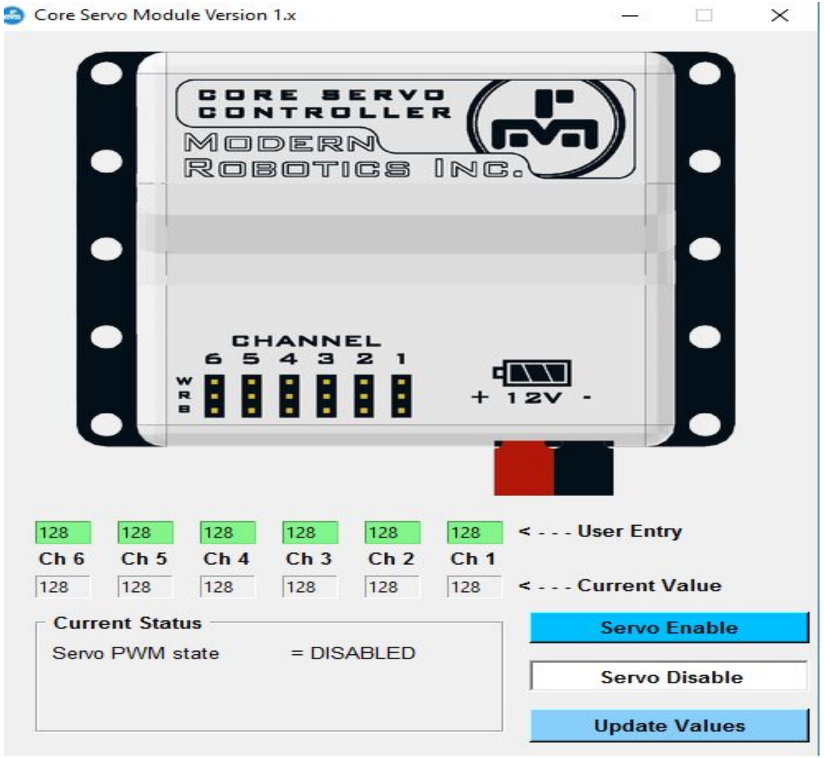
Current Status

Motor Mode = POWER
Battery Voltage = 12.54v

Test and Debug

Servo Motors

We have found the core servo controller so now we will click on it and the screen on the right will show up.



Core Servo Module Version 1.x

128	128	128	128	128	128	< - - - User Entry
Ch 6	Ch 5	Ch 4	Ch 3	Ch 2	Ch 1	
128	128	128	128	128	128	< - - - Current Value

Current Status

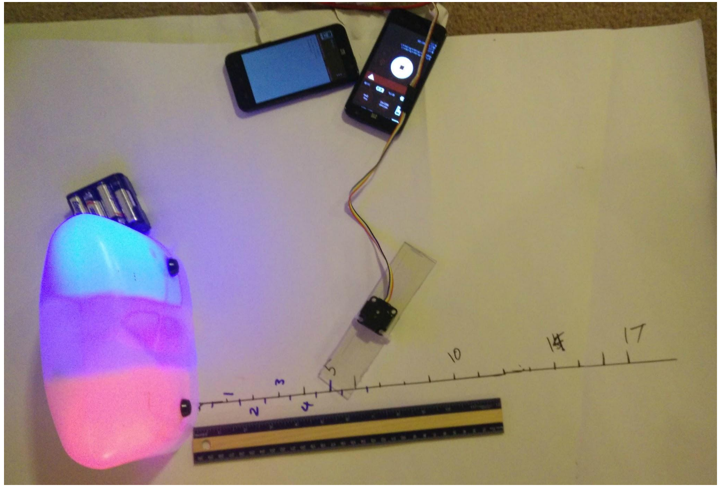
Servo PWM state = DISABLED

Test and Debug

Calibration

→ Wheel Rotation

→ Color Sensor



Debug and Test

[DEMO -- Faraaz]

Tele Op

Focus areas

- → Navigation - moving around in the field
- → Attachment Controls
- → Gamepad Assignments
- → Execution Strategy

Tele Op

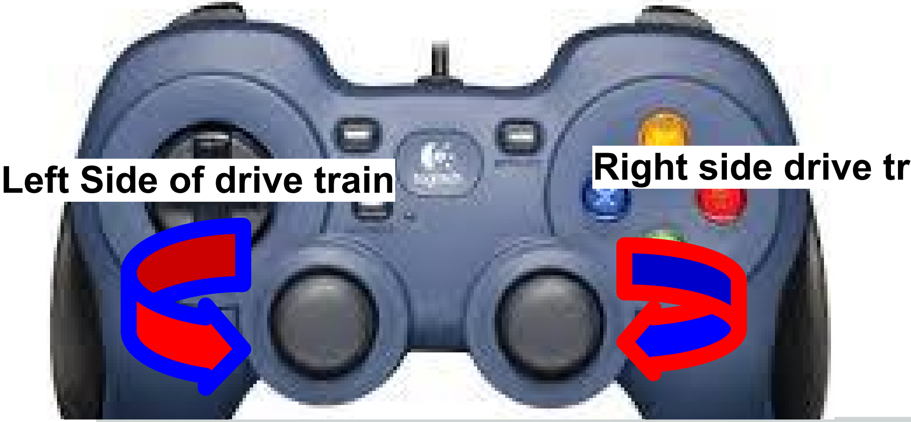
Navigation

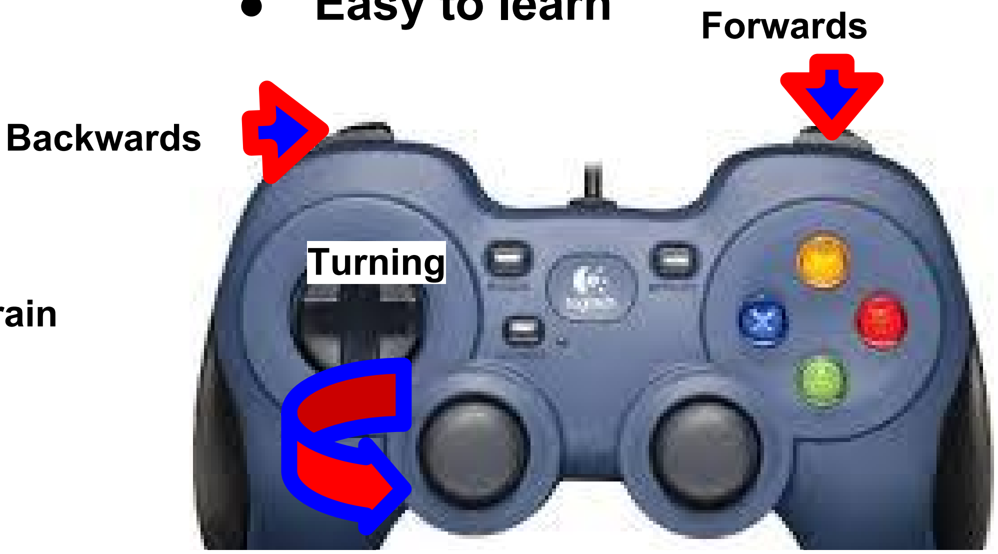
Tank Drive

- Point turning
- Swing turning
- Precise controls on each side of the drive train

Race car drive

- Less room to mess up going forwards or backwards
- Easy to learn





Tele Op

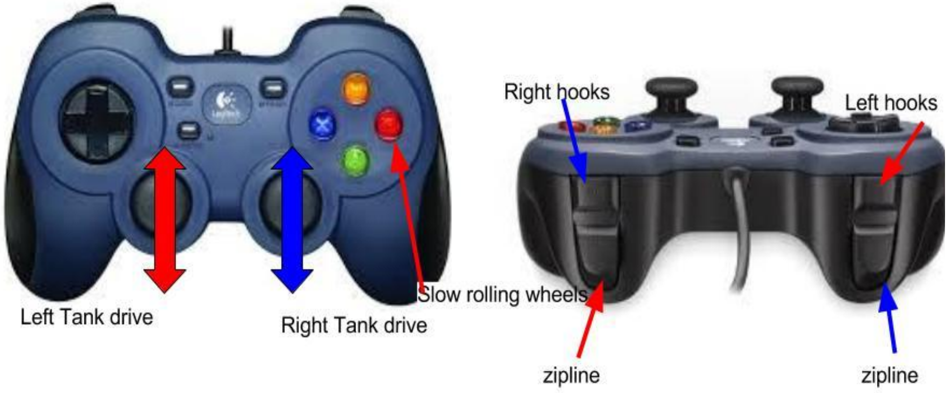
Gamepad Assignments

- We needed the controls to be convenient
- We needed the gamepads to have equal controls.
- But we didn't want synchronization to be required between drivers
- Wanted to avoid toggle controls

Tele Op

Gamepad Assignments

Gamepad1



Right hooks

Left hooks

Left Tank drive

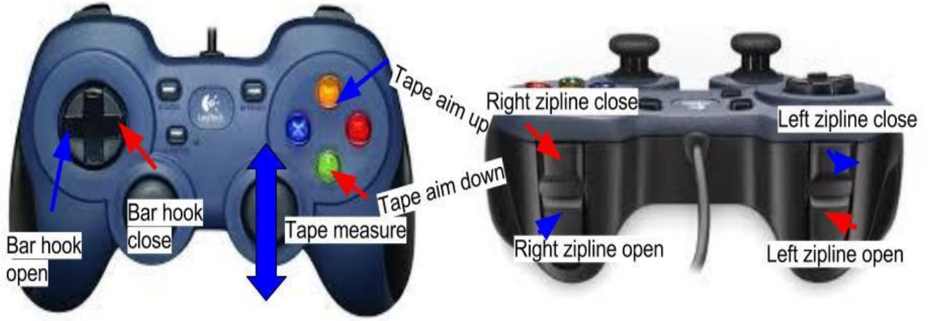
Right Tank drive

Slow rolling wheels

zipline

zipline

Gamepad2



Tape aim up

Right zipline close

Left zipline close

Tape aim down

Bar hook close

Bar hook open

Tape measure

Right zipline open

Left zipline open

Drive

[Drive DEMO]

Programming Q&A

Part 2

- Resources and helpful links
    - Starting a new team
    - How to Get Organized
- Engineering Notebook requirements
- Award Categories
- Lesson Learned

FTC Team 9901

Resources and helpful links

* **Intellitek training:** http://ftc.edu.intelitek.com/ (select course, login as guest)

- Module 1: HW and Control Systems
- Module 3: Java Programming with Android Studio

* **Game Manual**

- FTC Game manual part 1 2016 – 2017 (released already)
- FTC Game manual part 2 2016 – 2017 (Coming 9/10/2016)

* **FTC forum** (official/moderated): http://ftcforum.usfirst.org/forum.php

* Unofficial forums (unmoderated):

- https://www.chiefdelphi.com/forums/ [FIRST Tech Challenge]
- https://www.reddit.com/r/FTC/

* **Official FTC Blog:** http://firsttechchallenge.blogspot.com/

* **PushBot build guide:**

http://www.firstinspires.org/sites/default/files/uploads/resource_library/ftc/pushbot-build-guide.pdf
http://www.firstinspires.org/sites/default/files/uploads/resource_library/ftc/pushbot-build-guide-tetrix-sensors-supplement.pdf

FTC Team 9901

Resources and helpful links

* **Starting a new team**

http://www.firstinspires.org/ftc-start-a-team

- Registration
- Mentor Manual
- Team Budget
- Fundraising
- Robot Building Resource (http://www.firstinspires.org/node/5181)

FTC Team 9901

How to Get Organized

http://techietitans-ftc.com (our Website)

https://www.facebook.com/TechieTitans9901 (our Facebook page)

https://www.youtube.com/ (our YouTube channel)

https://github.com/ (code repository - revision control and code management)

https://drive.google.com/drive (document storage in Cloud, for sharing)

https://slack.com (messaging app, for team collaboration)

FTC Team 9901

Engineering Notebook Requirements

- A complete documentation of the Team’s journey throughout the season.
  - Sketches
  - Discussions
  - Design evolution
  - Obstacles
  - Team member’s thoughts throughout the journey for the entire season.
  - A new notebook should be created for each new season.
  - Through explanations and thought processes

- All Awards, other than robot game competition, are given based on Engineering Notebook content (and the team presentation/interview).

- Can be either HANDWRITTEN or ELECTRONIC.

- Detailed rules, examples, and templates for documents are in “Engineering Notebook Guidelines”

FTC Team 9901

Award Categories

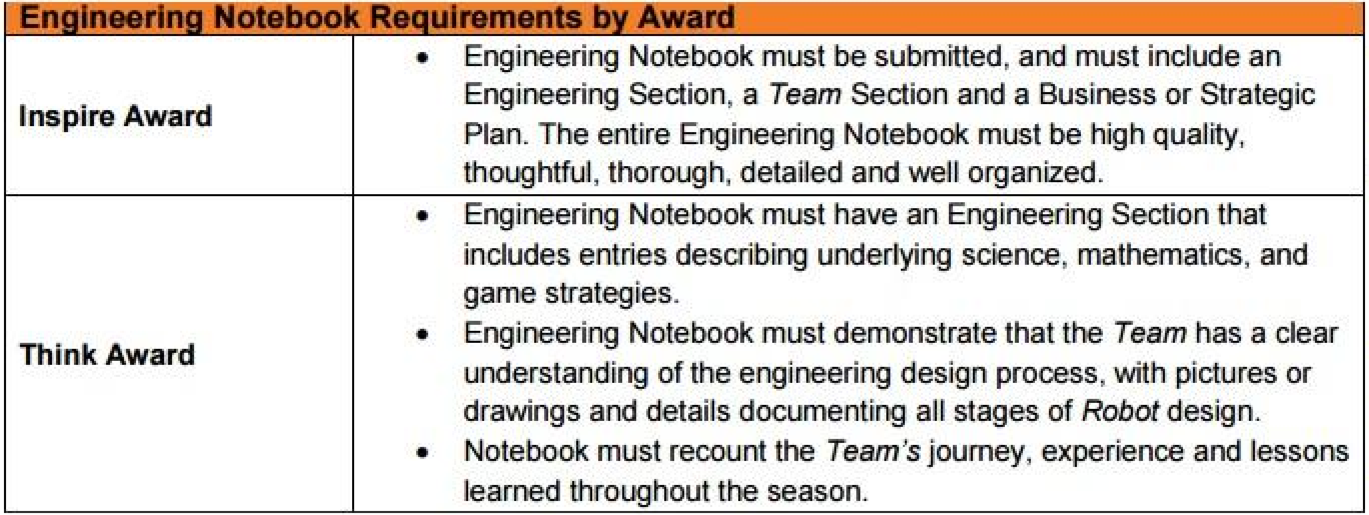
- Robot Game Award
  - Winning Alliance
  - Finalist Alliance

- Individual Awards
  - Inspire Award
  - Think Award
  - Connect Award
  - Rockwell Collins Innovate Award
  - PTC Design Award
  - Motivate Award
  - Control Award

- Advancement criteria is based on award ranking (see “Game Manual Part 1”)

FTC Team 9901

Awards and the Engineering Notebook



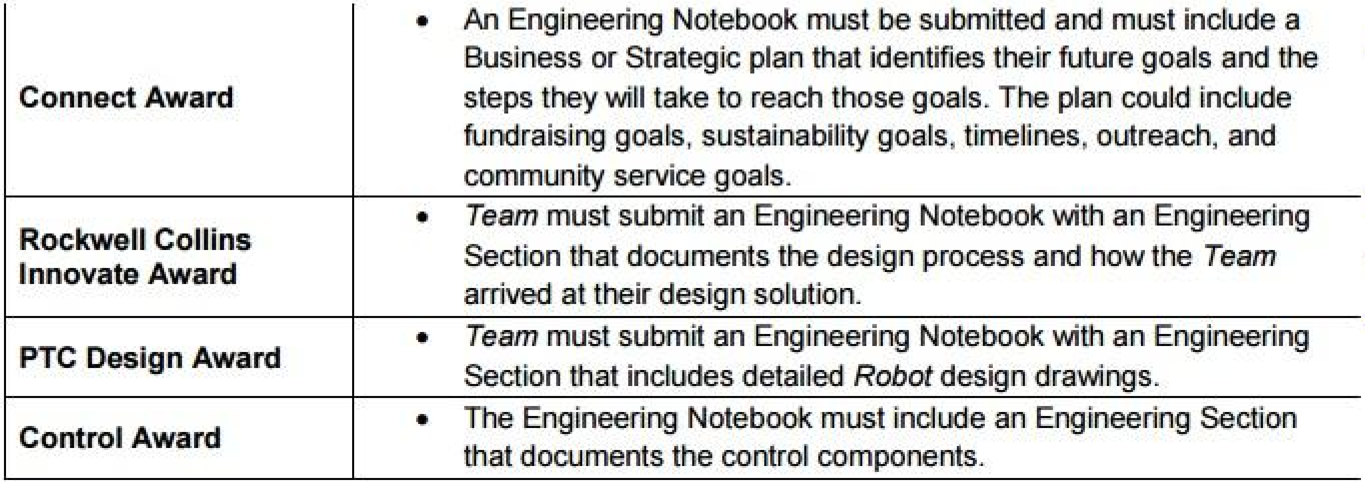
Engineering Notebook Requirements by Award	
Inspire Award	- Engineering Notebook must be submitted, and must include an Engineering Section, a *Team* Section and a Business or Strategic Plan. The entire Engineering Notebook must be high quality, thoughtful, thorough, detailed and well organized.
Think Award	- Engineering Notebook must have an Engineering Section that includes entries describing underlying science, mathematics, and game strategies. - Engineering Notebook must demonstrate that the *Team* has a clear understanding of the engineering design process, with pictures or drawings and details documenting all stages of *Robot* design. - Notebook must recount the *Team's* journey, experience and lessons learned throughout the season.

-Team Section
-Business Section
-Engineering Sect

- Eng. Section
- Design Process
- Team's Journey

FTC Team 9901

Awards and the Engineering Notebook



Connect Award	• An Engineering Notebook must be submitted and must include a Business or Strategic plan that identifies their future goals and the steps they will take to reach those goals. The plan could include fundraising goals, sustainability goals, timelines, outreach, and community service goals.
Rockwell Collins Innovate Award	• *Team* must submit an Engineering Notebook with an Engineering Section that documents the design process and how the *Team* arrived at their design solution.
PTC Design Award	• *Team* must submit an Engineering Notebook with an Engineering Section that includes detailed *Robot* design drawings.
Control Award	• The Engineering Notebook must include an Engineering Section that documents the control components.

- Business Plan

- Engineer. Sect.
- Robot drawings
- Engineering

Design

FTC Team 9901

Lesson Learned

* Understand the game requirements

* Use a test / prototype environment or board

- to work on programming in parallel

* Build prototypes with cardboard, plastic pieces (avoid tetrix parts)

* Test incrementally

* Think about maintenance of your design

- complex vs. simple (reliable, easy to maintain)
- need to sustain multiple tournaments

* Tournament dates and location (only 1 in NoVA region)

* Tournament readiness (taking everything needed)

- inventory, checklist, team roster, inspection checklist)

* Final robot for competition

- inspection checklist, team # display (only one robot)

Q & A