FTC Control Systems + Java 101

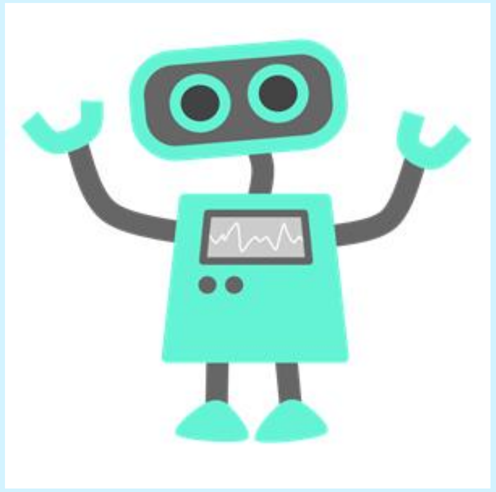
8949 The Gifted Gears

Outline

- Control Systems
    - REV
    - Modern Robotics
    - Phones and Phone Setup
- Programming Options and Setup
- Navigating Android Studios
- Java!
    - OpModes
    - Basics
    - Actuators
    - Teleop

What Am I Programming in FTC?

- Autonomous
  - Like FLL, robot must complete tasks on its own
    - Robot must make decisions on its own



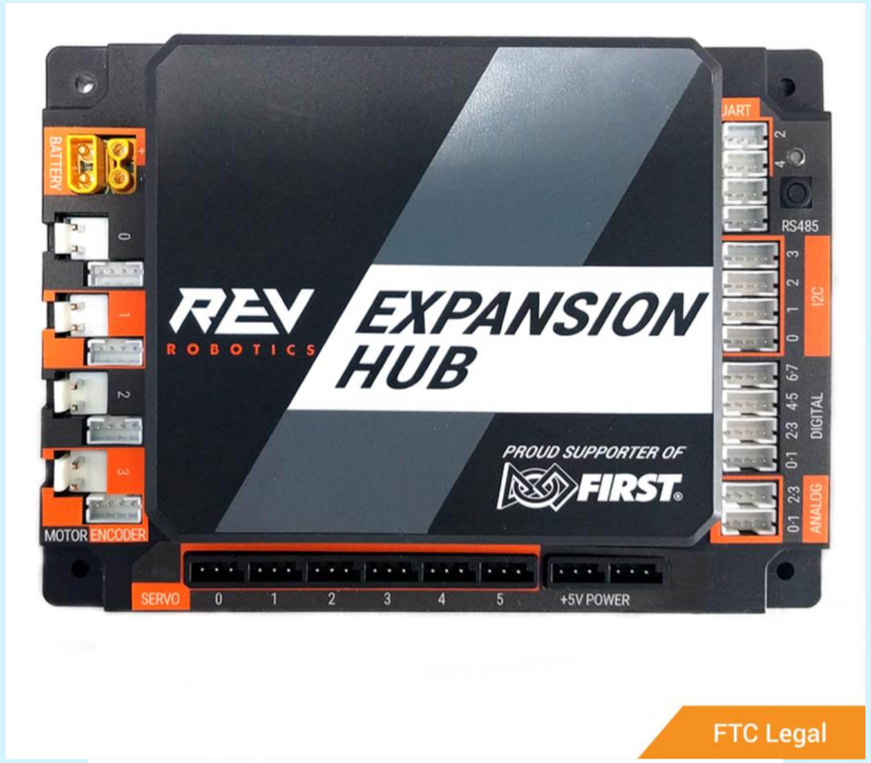
- Teleop (Teleoperations)
  - Gamepad Controls
    - How do you want to control your robot with the gamepad?

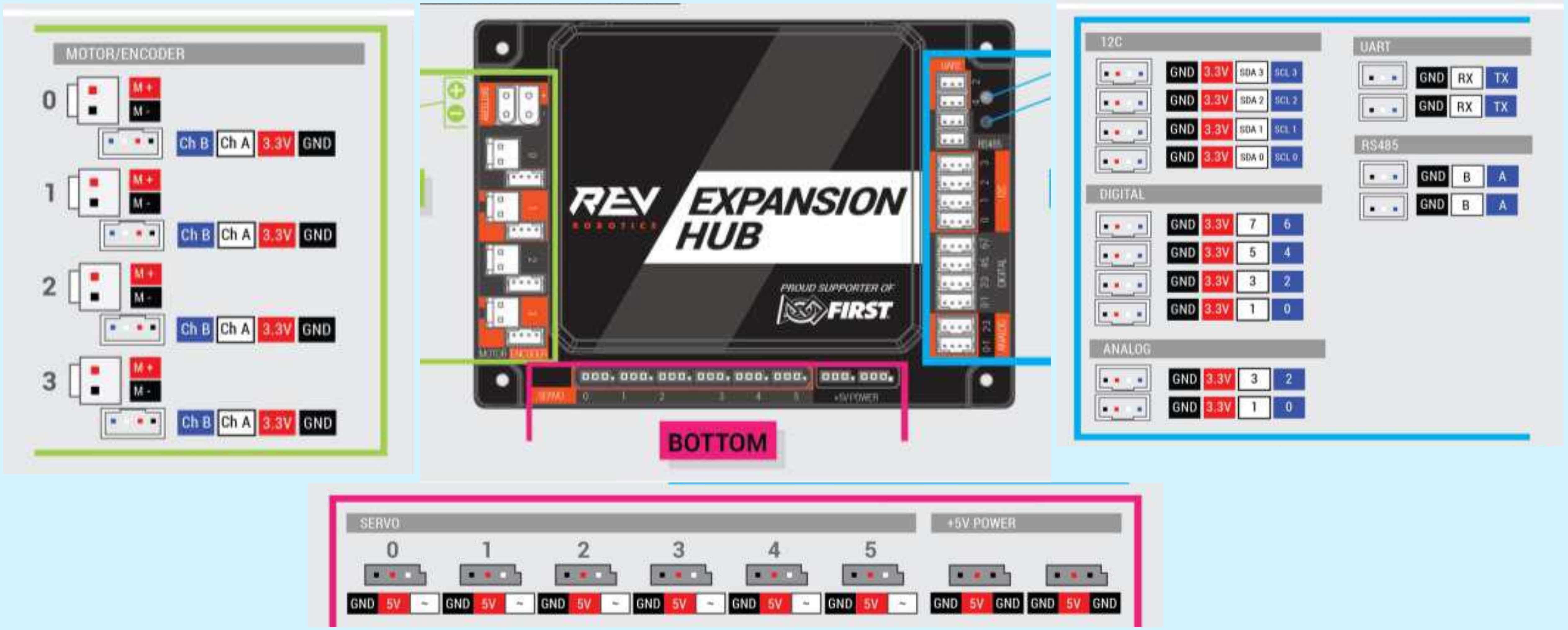
Example Match

FTC Control System

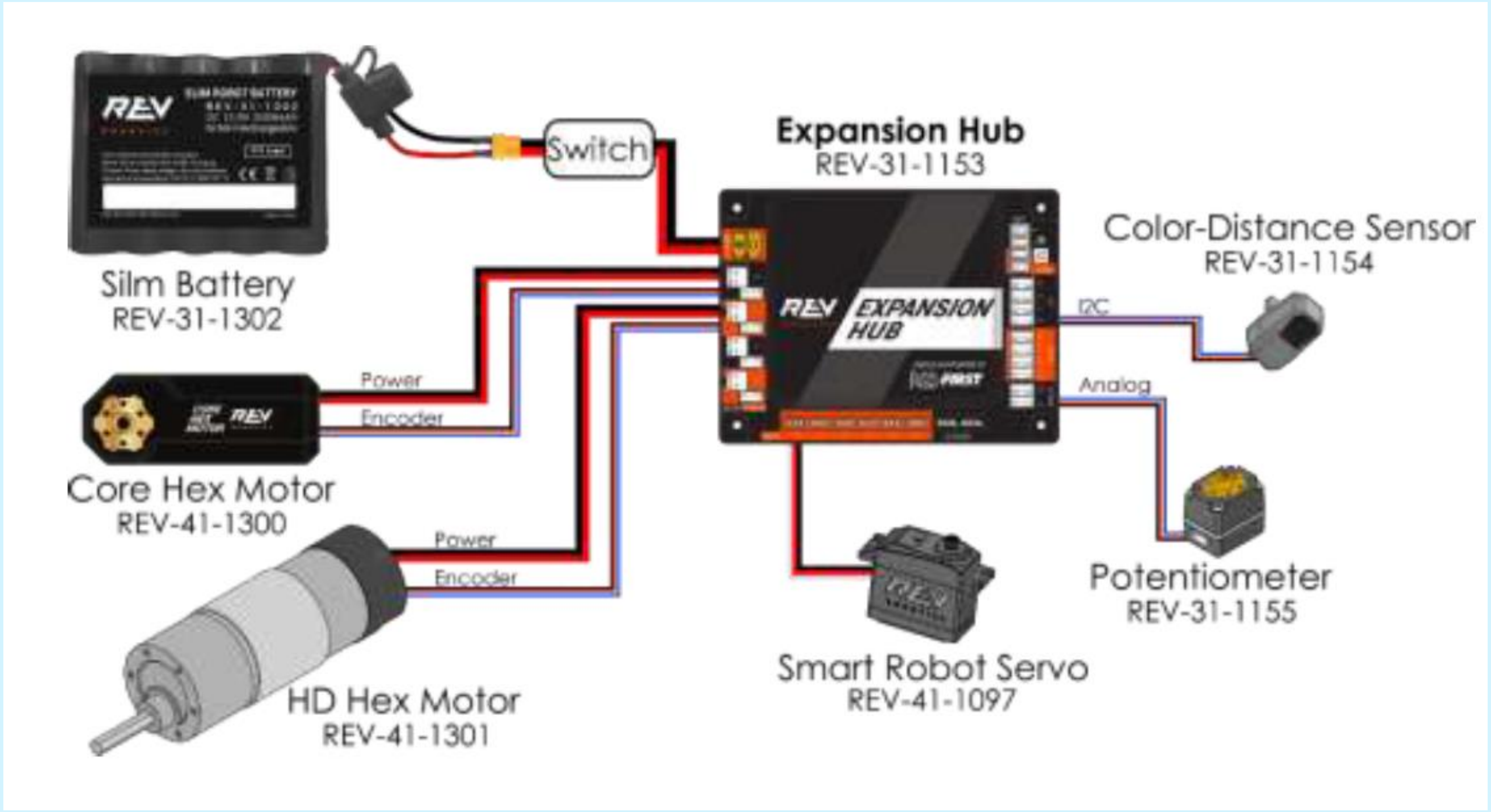
REV Robotics

- Expansion hub = hardware controller that can communicate with computers, Android tablets/phones
  - This is where you plug all your hardware in!
  - All ports (motor, sensor, servo) consolidated onto one hub
  - Up to 2 hubs per robot
- Servo Power Module
  - Specifically for servos
  - Up to 1 module



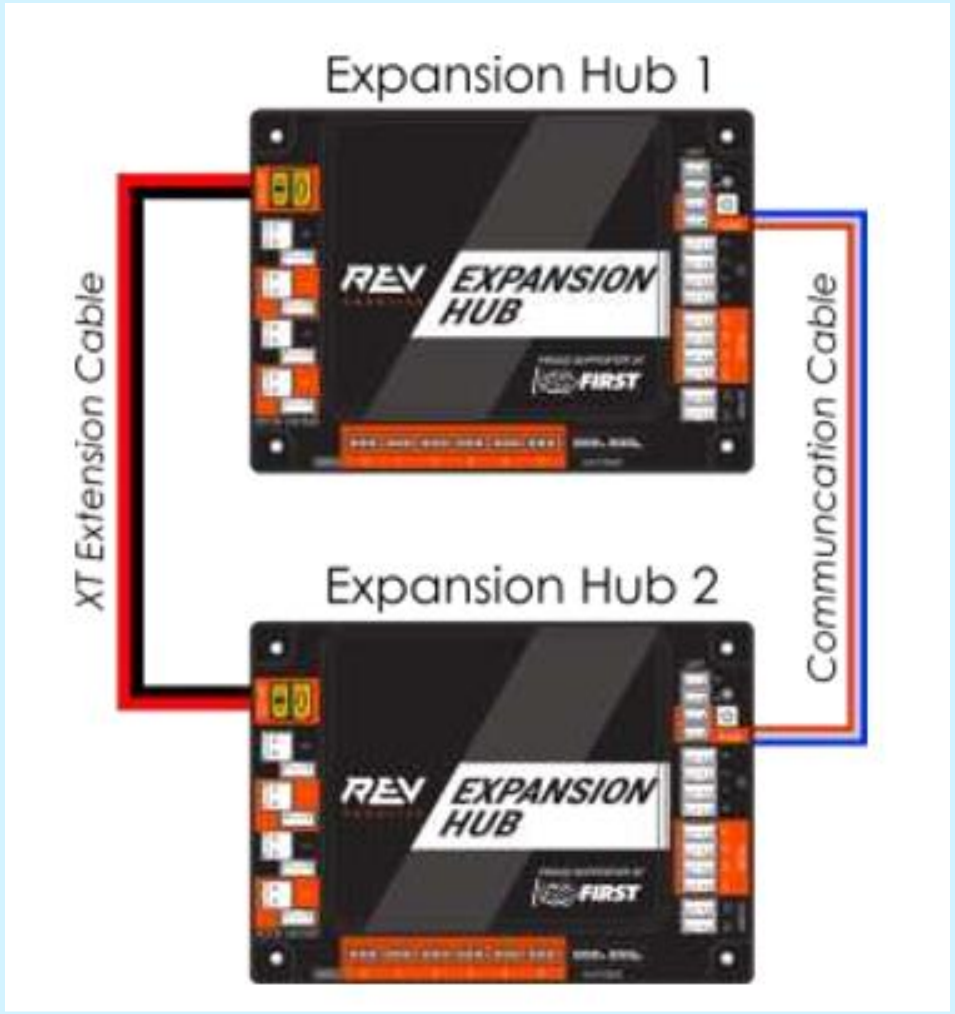


⚙ 8949 The Gifted Gears ⚙


Switch
Expansion Hub
REV-31-1153
Silm Battery
REV-31-1302
Color-Distance Sensor
REV-31-1154
I2C
Analog
Power
Encoder
Core Hex Motor
REV-41-1300
Power
Encoder
HD Hex Motor
REV-41-1301
Smart Robot Servo
REV-41-1097
Potentiometer
REV-31-1155

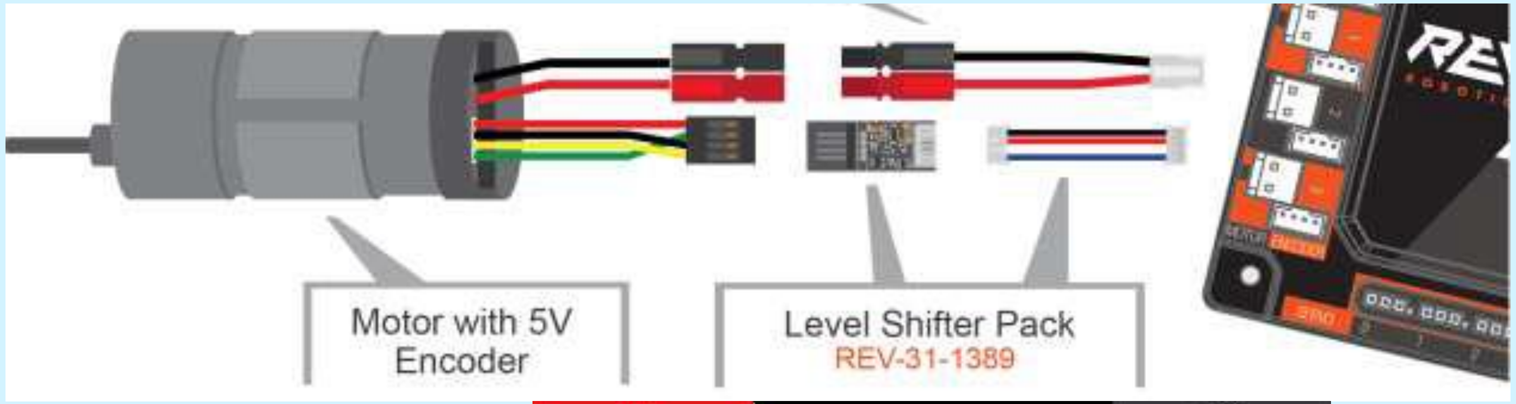
Using Multiple REV Hubs

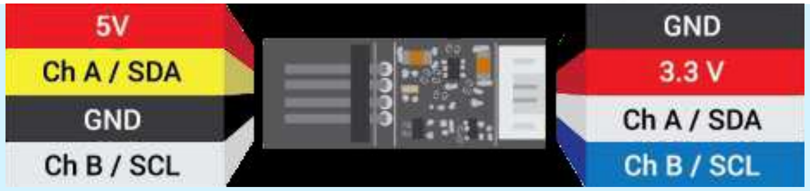
- Daisy Chain
  - If using multiple REV hubs, must be connected with XTE cable and Communication cable RS485
- Change wifi addresses of one of the hubs
  - Advanced Settings in the app
  - Connect to computer and change through REV software



Level Shifter Required!

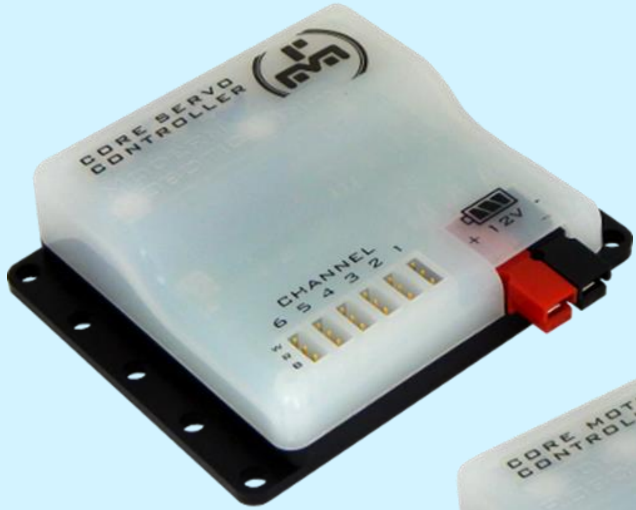
- REV Hub is incompatible with some motor encoders and sensors, need a level shifter to connect the two
  - REV hub is a 3.3V device, many encoders/sensors are 5V
  - For complete list of sensors/encoders that require level shifters, look it up on the REV website

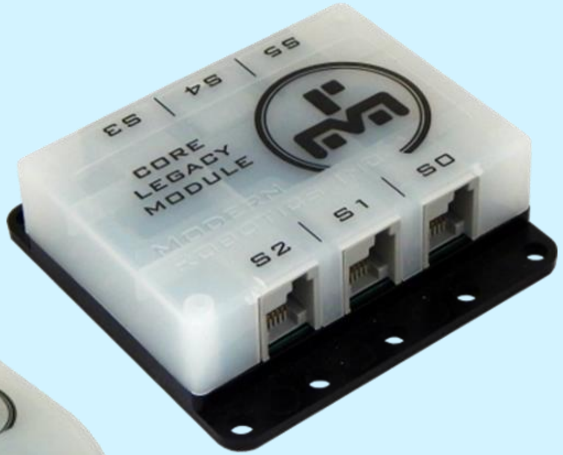


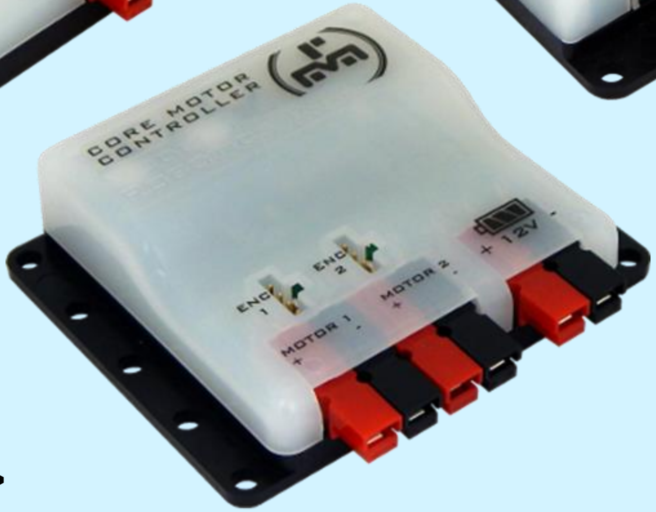


Modern Robotics

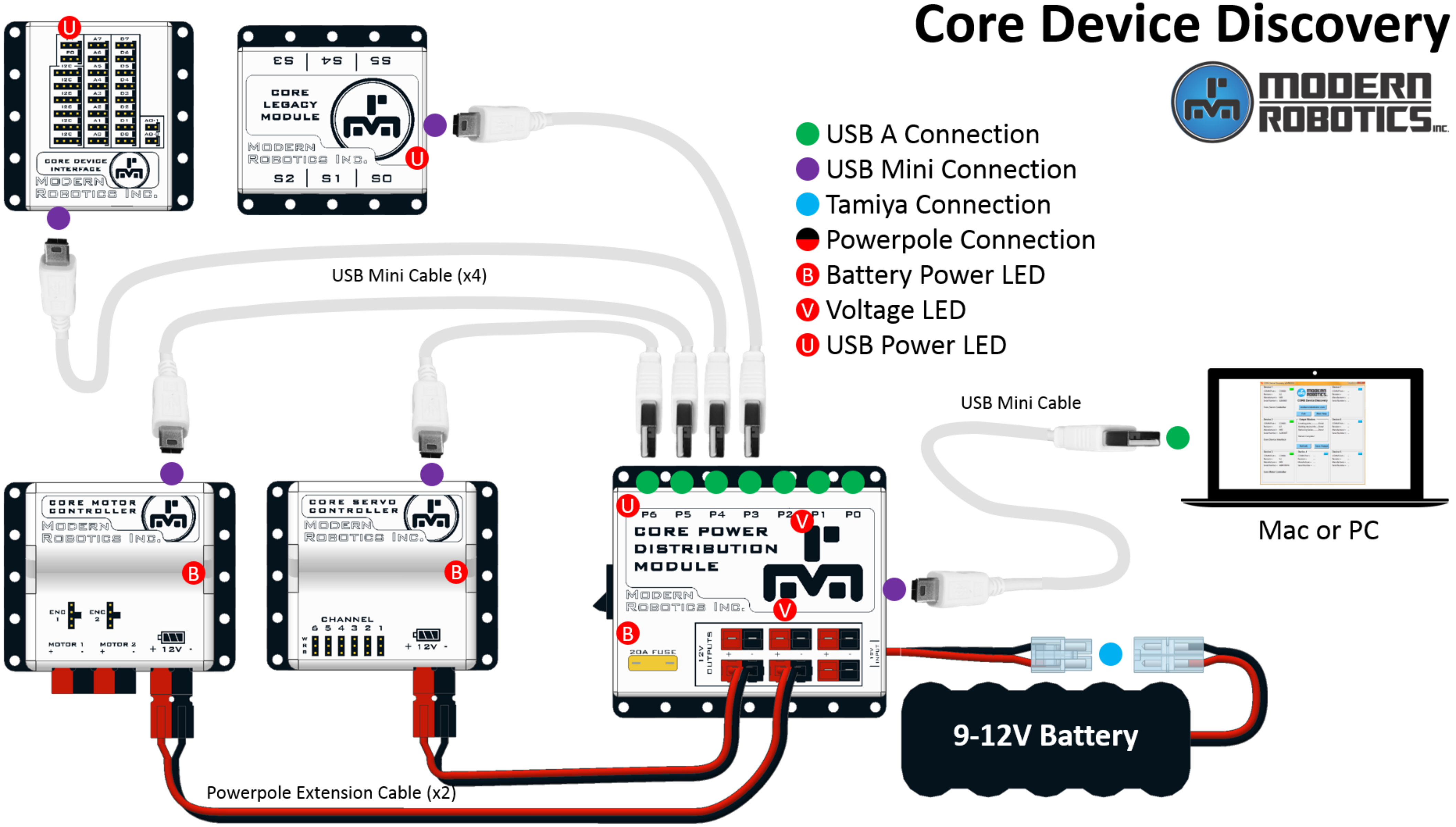
- This is where you plug all your hardware in!
- Separate modules with different ports
  - Core power distribution
  - Motor
  - Servo
  - Legacy/Core Device Interface
- Up to 4 motor controllers, 2 servo controllers, 1 legacy/core device interface, 1 core power distribution







Core Device Discovery



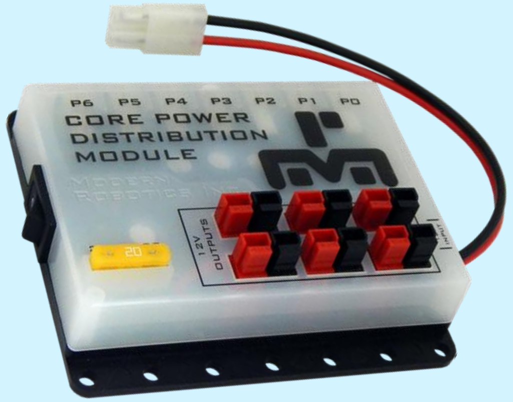
- 🟢 USB A Connection
- 🟣 USB Mini Connection
- 🔵 Tamiya Connection
- ● Powerpole Connection
- B Battery Power LED
- V Voltage LED
- U USB Power LED

8949 The Gifted Gears

Comparison: REV and Modern Robotics

- REV
  - All ports (motor, sensor, servo) consolidated onto one hub
  - Up to 2 hubs and 1 servo
  - Built in IMU sensor
  - Level shifter required for encoders and some sensors

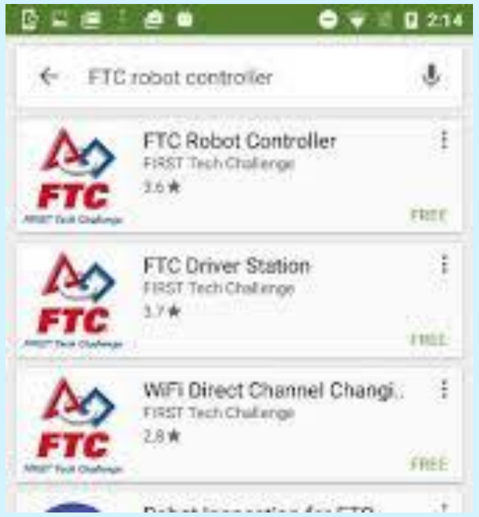
- Modern Robotics
  - Separate modules (power distribution, motor, servo, sensor)
  - Up to 4 motor, 2 servo, 1 sensor, 1 core power distribution
  - External IMU/gyro needed
  - No level shifter required

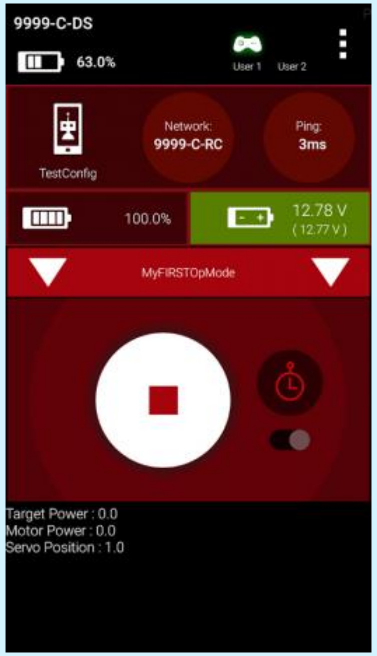


Phones

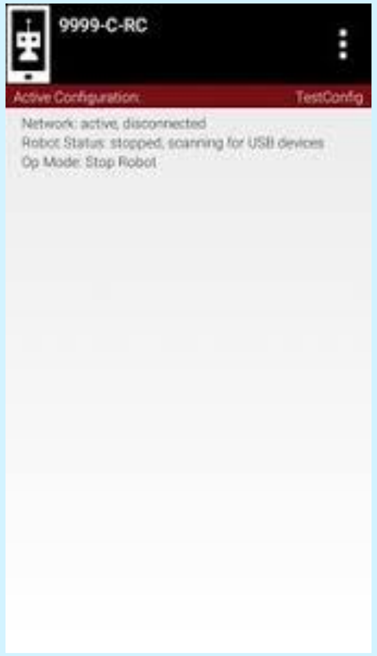
Overview

- Connects user input, hardware, and software
- 2 Android phones needed
  - Download respective apps (FTC Robot Controller, FTC Driver Station) from the Play Store
- Robot Controller
  - Connects to robot
  - Download program here
- Driver Station
  - Connects to gamepad(s)
  - Select, start, and stop programs
  - Monitor battery levels



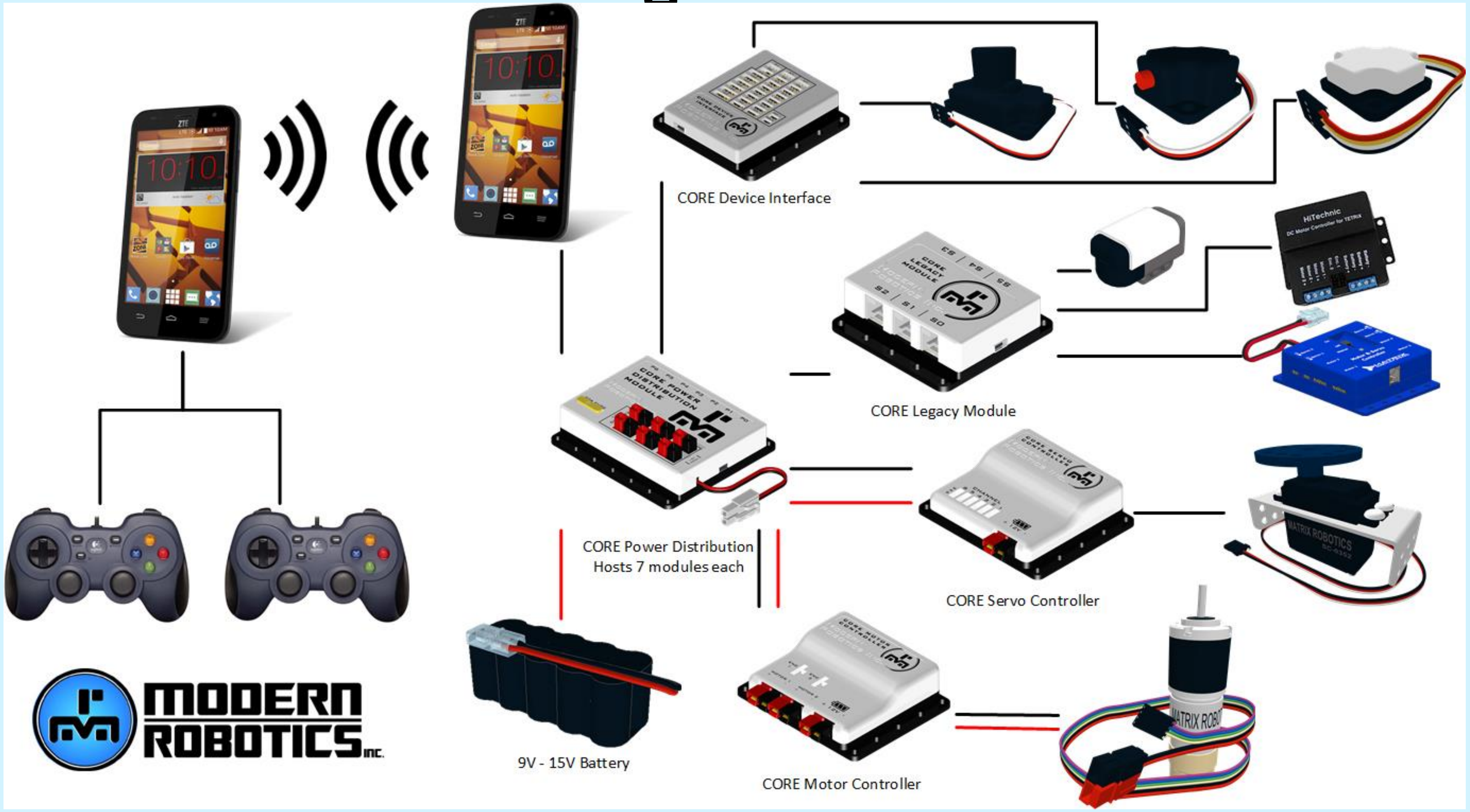


Target Power : 0.0
Motor Power : 0.0
Servo Position : 1.0



Network: active, disconnected
Robot Status: stopped, scanning for USB devices
Op Mode: Stop Robot

How It All Works Together



8949 The Gifted Gears

Legal Phones

- ZTE Speed
- Motorola Moto G 2nd Generation
- Motorola Moto G 3rd Generation
- Motorola Moto G4 Play
- Motorola Moto G5
- Motorola Moto E4
- Google Nexus 5
- Samsung Galaxy S5

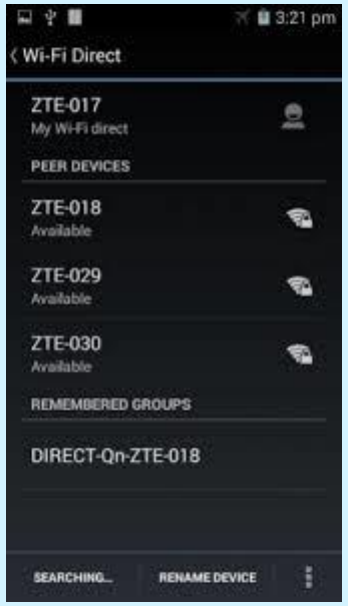
Phone Setup: Rename Phones

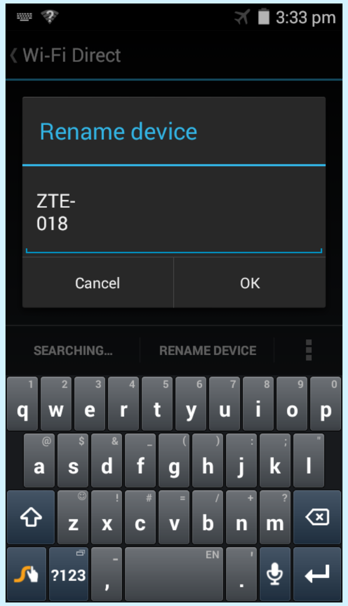
- MUST name with team number and –RC or –DS (will not pass inspection otherwise!)
  - 12345-RC, 12345-DS
- Spare Android devices should be named with team number–A–RC/DS
  - 12345-A-RC, 12345-A-DS

- Settings > Wifi > Wifi Direct > Rename Device

OR

- Go to app > 3 dots at the top right corner > Settings > RC/DS Name > Enter new name



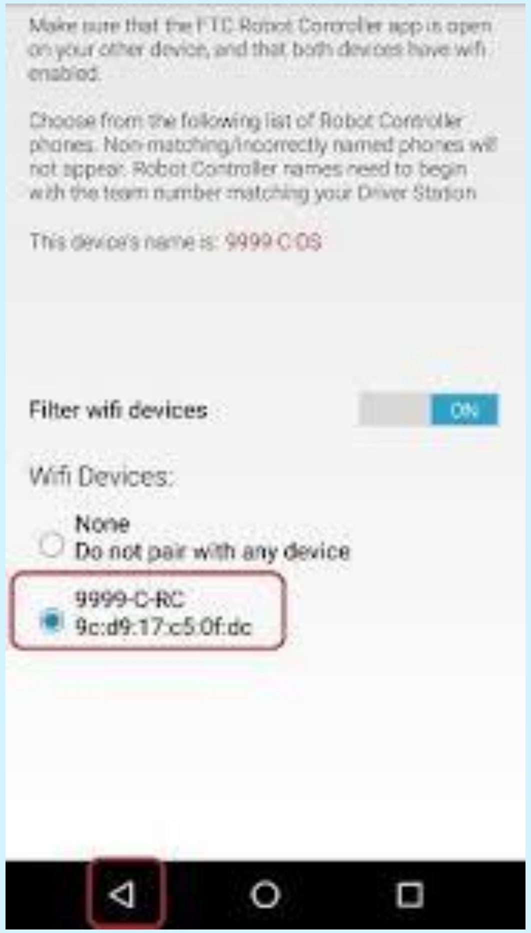


Phone Setup: Connect RC + DS

- Settings > Wifi > Wifi Direct > Select name of the RC/DS you're pairing with > Accept invitation on the other phone

OR

- Go to app > 3 dots at the top right corner > Settings > Pair with RC/DS > Select name of the RC/DS you're pairing with > Accept invitation on the other phone



Make sure that the FTC Robot Controller app is open on your other device, and that both devices have wifi enabled.

Choose from the following list of Robot Controller phones. Non-matching/incorrectly named phones will not appear. Robot Controller names need to begin with the team number matching your Driver Station

This device's name is: 9999-C-DS

Filter wifi devices ON

Wifi Devices:

○ None
Do not pair with any device

● 9999-C-RC
9c:d9:17:c5:0f:dc

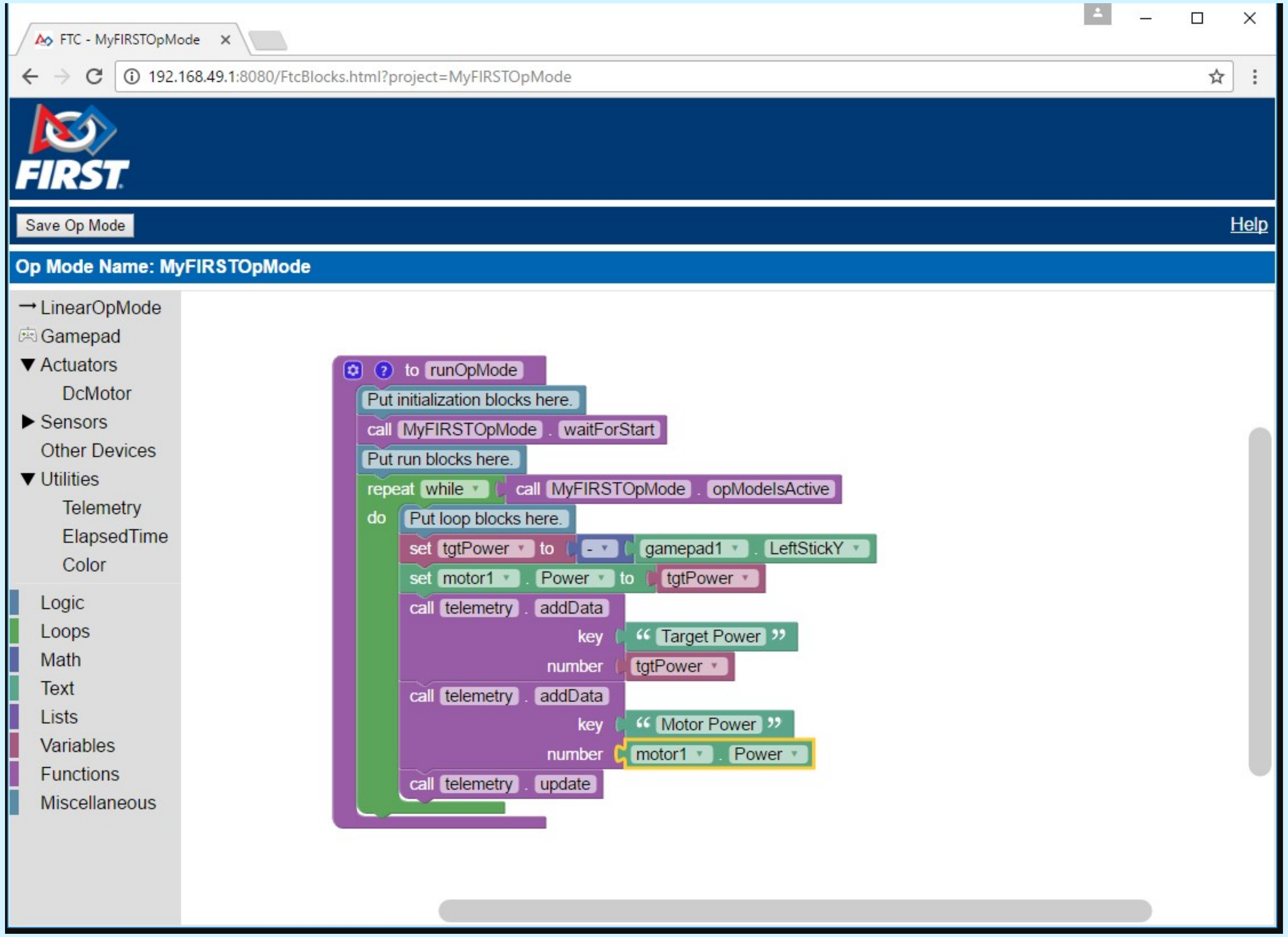
FTC Programming Setup

Available Programming Platforms

- Blocks
- OnBot Java
- Java

Blocks

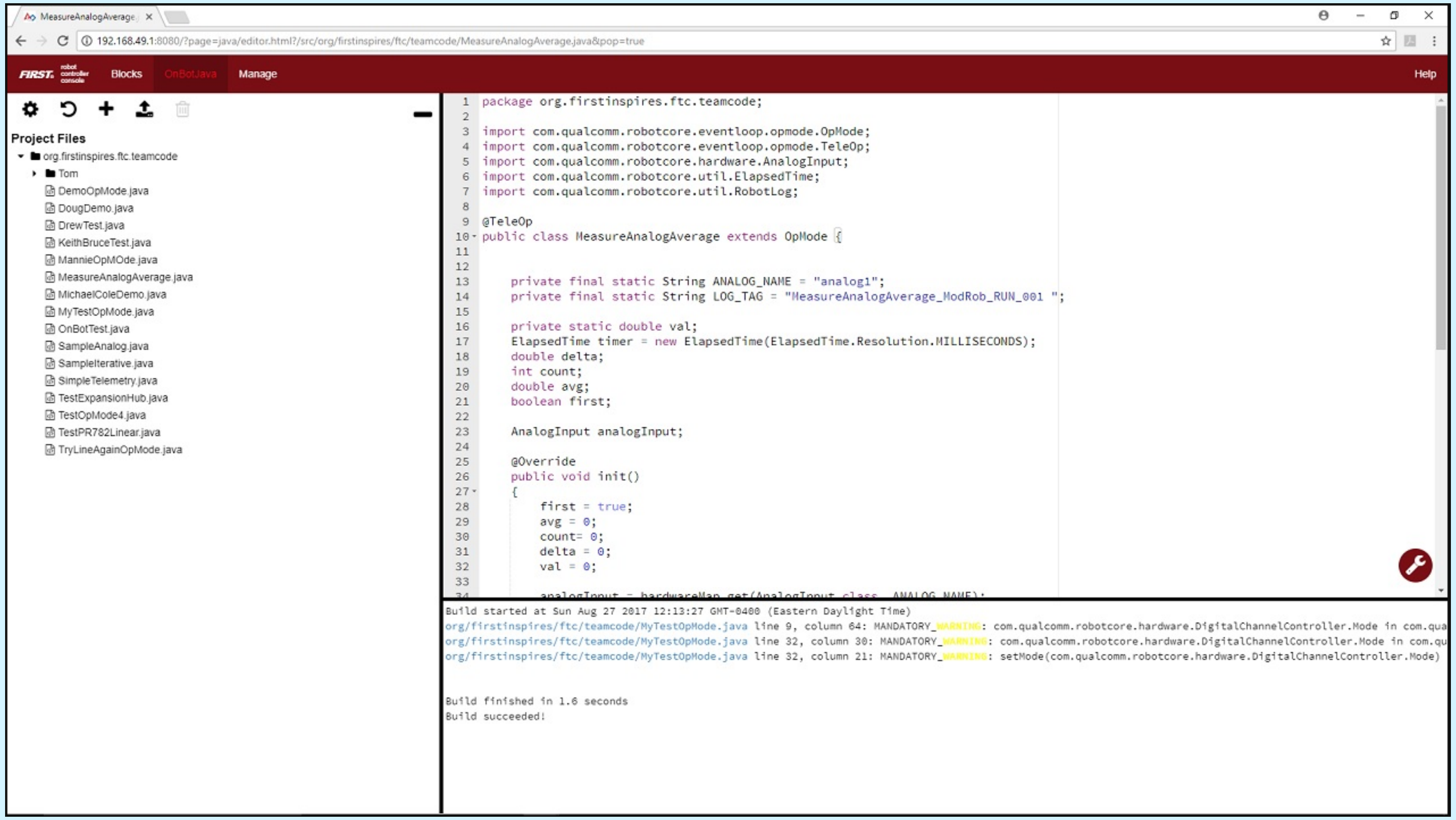
- Web-based hosted by the RC
  - RC acts as server that you connect to and program on
- Visual, drag and drop
- No setup necessary



```
to runOpMode
  Put initialization blocks here.
  call MyFIRSTOpMode . waitForStart
  Put run blocks here.
  repeat while  call MyFIRSTOpMode . opModeIsActive
  do  Put loop blocks here.
      set tgtPower to  -  gamepad1 . LeftStickY
      set motor1 . Power to  tgtPower
      call telemetry . addData
                       key  " Target Power "
                    number  tgtPower
      call telemetry . addData
                       key  " Motor Power "
                    number  motor1 . Power
      call telemetry . update
```

OnBot Java

- Object-oriented text based language
- Hosted on RC
- No setup necessary


MeasureAnalogAverage.j
192.168.49.1:8080/?page=java/editor.html?/src/org/firstinspires/ftc/teamcode/MeasureAnalogAverage.java&pop=true
FIRST robot controller console
Blocks
OnBotJava
Manage
Help

Project Files

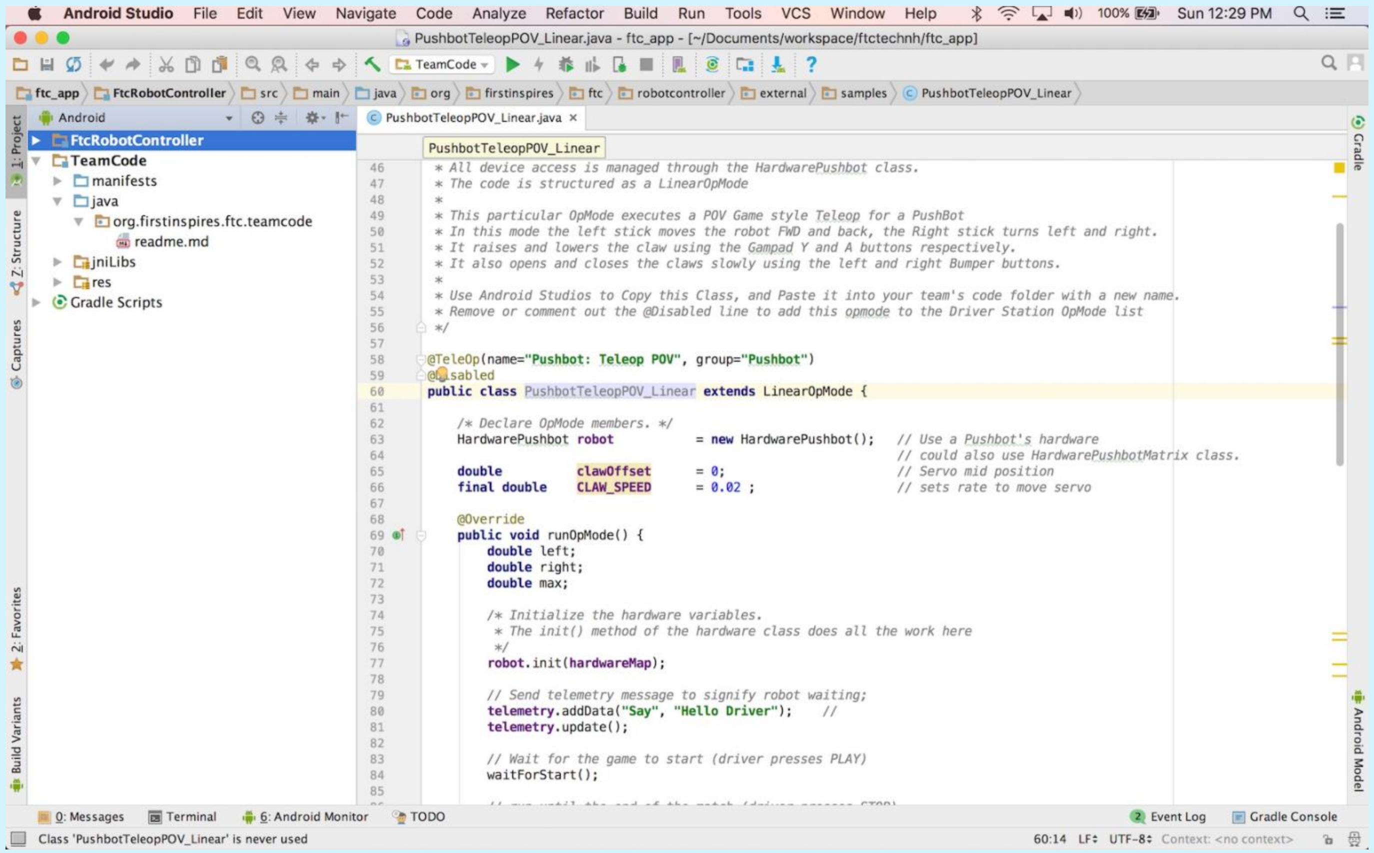
- org.firstinspires.ftc.teamcode
  - Tom
  - DemoOpMode.java
  - DougDemo.java
  - DrewTest.java
  - KeithBruceTest.java
  - MannieOpMOde.java
  - MeasureAnalogAverage.java
  - MichaelColeDemo.java
  - MyTestOpMode.java
  - OnBotTest.java
  - SampleAnalog.java
  - SampleIterative.java
  - SimpleTelemetry.java
  - TestExpansionHub.java
  - TestOpMode4.java
  - TestPR782Linear.java
  - TryLineAgainOpMode.java

```java
package org.firstinspires.ftc.teamcode;

import com.qualcomm.robotcore.eventloop.opmode.OpMode;
import com.qualcomm.robotcore.eventloop.opmode.TeleOp;
import com.qualcomm.robotcore.hardware.AnalogInput;
import com.qualcomm.robotcore.util.ElapsedTime;
import com.qualcomm.robotcore.util.RobotLog;

@TeleOp
public class MeasureAnalogAverage extends OpMode {


    private final static String ANALOG_NAME = "analog1";
    private final static String LOG_TAG = "MeasureAnalogAverage_ModRob_RUN_001 ";

    private static double val;
    ElapsedTime timer = new ElapsedTime(ElapsedTime.Resolution.MILLISECONDS);
    double delta;
    int count;
    double avg;
    boolean first;

    AnalogInput analogInput;

    @Override
    public void init()
    {
        first = true;
        avg = 0;
        count= 0;
        delta = 0;
        val = 0;

        analogInput = hardwareMap.get(AnalogInput.class, ANALOG_NAME);
```

```
Build started at Sun Aug 27 2017 12:13:27 GMT-0400 (Eastern Daylight Time)
org/firstinspires/ftc/teamcode/MyTestOpMode.java line 9, column 64: MANDATORY_WARNING: com.qualcomm.robotcore.hardware.DigitalChannelController.Mode in com.qua
org/firstinspires/ftc/teamcode/MyTestOpMode.java line 32, column 30: MANDATORY_WARNING: com.qualcomm.robotcore.hardware.DigitalChannelController.Mode in com.qu
org/firstinspires/ftc/teamcode/MyTestOpMode.java line 32, column 21: MANDATORY_WARNING: setMode(com.qualcomm.robotcore.hardware.DigitalChannelController.Mode)


Build finished in 1.6 seconds
Build succeeded!
```

Java

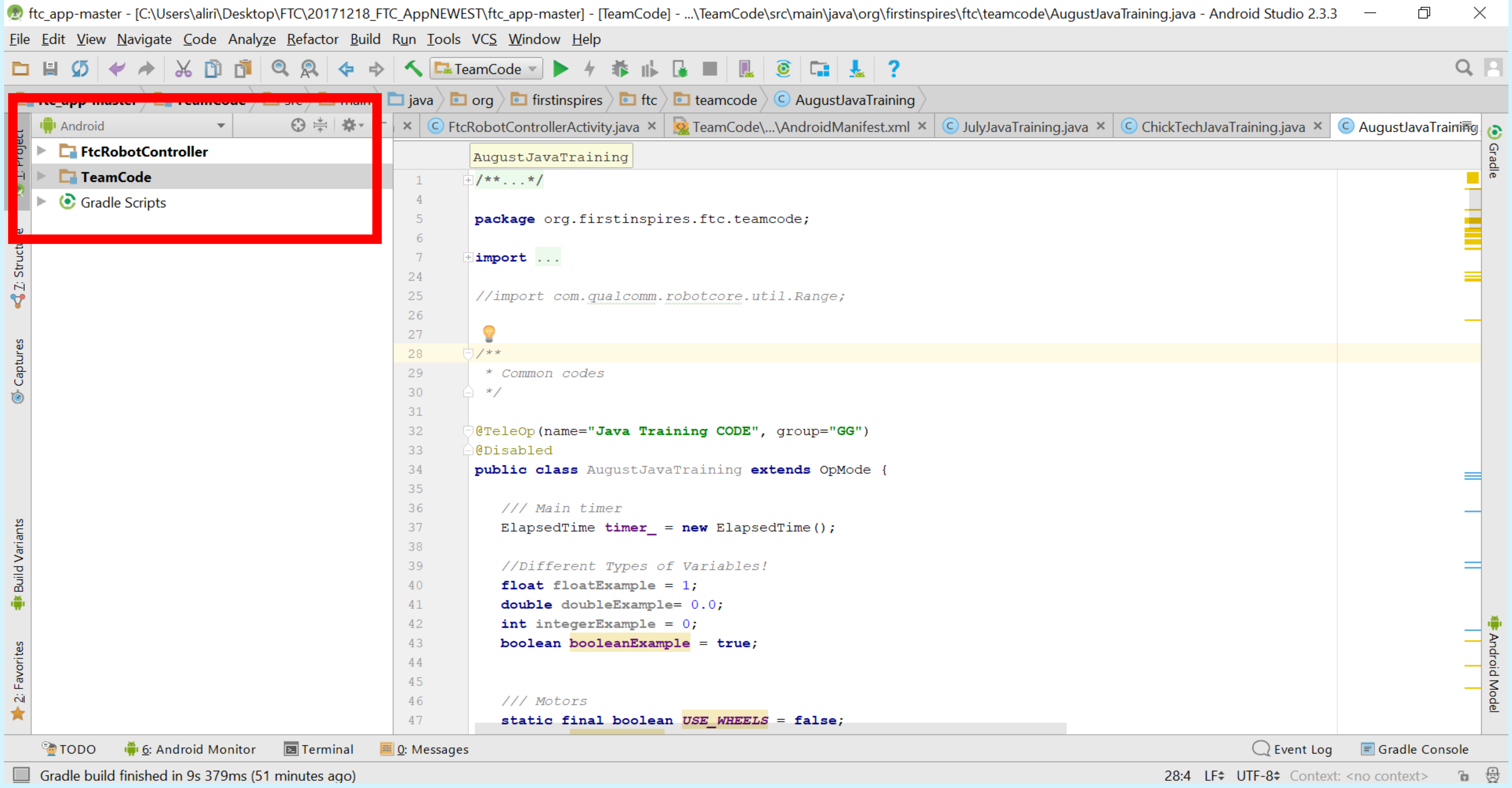
- Object-oriented text based language
- Setup required
  - Download your IDE (integrated development environment) Android Studios
  - Download FTC SDK (software development kit) from GitHub



```java
 * All device access is managed through the HardwarePushbot class.
 * The code is structured as a LinearOpMode
 *
 * This particular OpMode executes a POV Game style Teleop for a PushBot
 * In this mode the left stick moves the robot FWD and back, the Right stick turns left and right.
 * It raises and lowers the claw using the Gampad Y and A buttons respectively.
 * It also opens and closes the claws slowly using the left and right Bumper buttons.
 *
 * Use Android Studios to Copy this Class, and Paste it into your team's code folder with a new name.
 * Remove or comment out the @Disabled line to add this opmode to the Driver Station OpMode list
 */

@TeleOp(name="Pushbot: Teleop POV", group="Pushbot")
@Disabled
public class PushbotTeleopPOV_Linear extends LinearOpMode {

    /* Declare OpMode members. */
    HardwarePushbot robot           = new HardwarePushbot();   // Use a Pushbot's hardware
                                                               // could also use HardwarePushbotMatrix class.
    double          clawOffset      = 0;                       // Servo mid position
    final double    CLAW_SPEED      = 0.02 ;                   // sets rate to move servo

    @Override
    public void runOpMode() {
        double left;
        double right;
        double max;

        /* Initialize the hardware variables.
         * The init() method of the hardware class does all the work here
         */
        robot.init(hardwareMap);

        // Send telemetry message to signify robot waiting;
        telemetry.addData("Say", "Hello Driver");    //
        telemetry.update();

        // Wait for the game to start (driver presses PLAY)
        waitForStart();
```

Class 'PushbotTeleopPOV_Linear' is never used

Navigating Android Studios

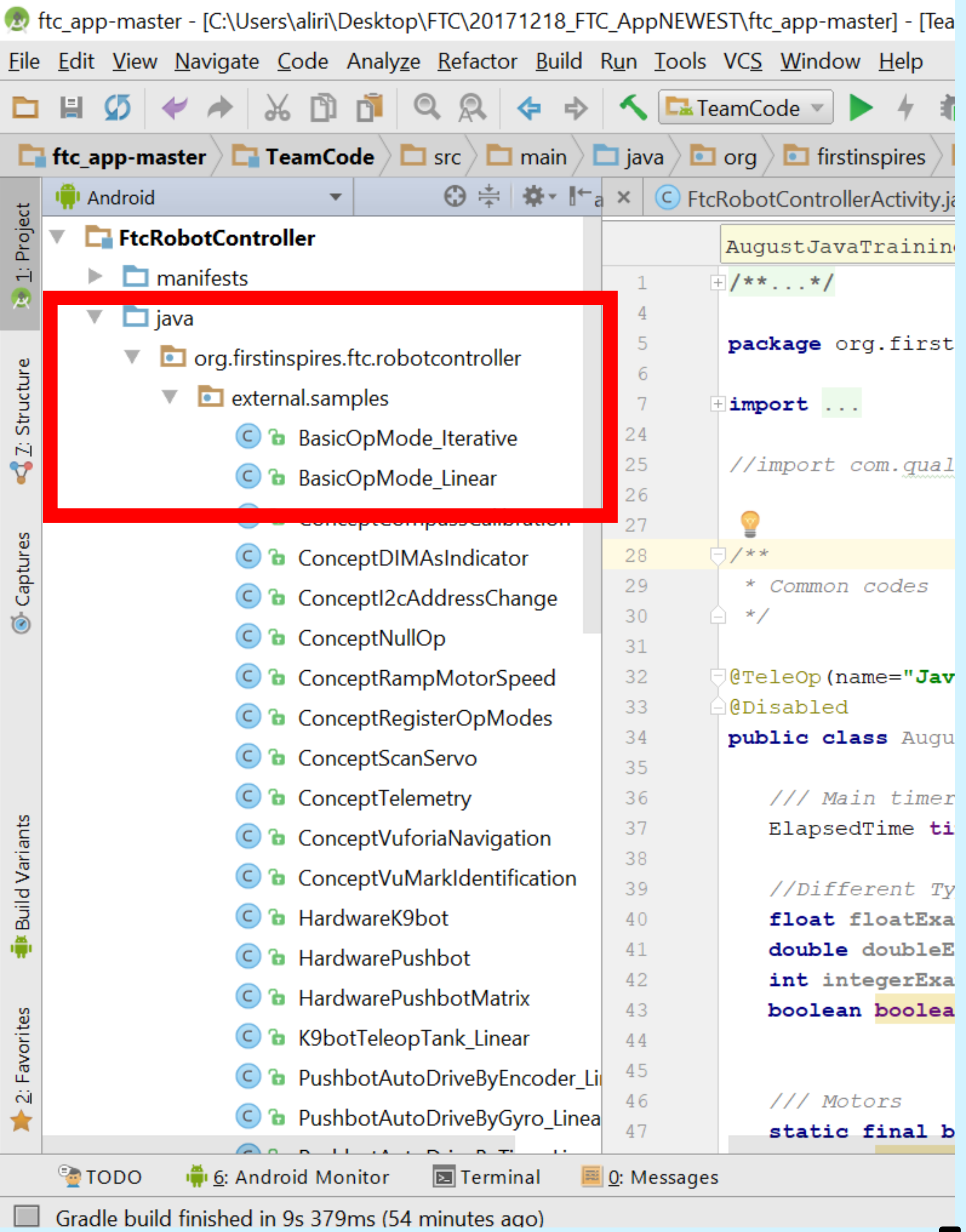


ftc_app-master - [C:\Users\aliri\Desktop\FTC\20171218_FTC_AppNEWEST\ftc_app-master] - [TeamCode] - ...\TeamCode\src\main\java\org\firstinspires\ftc\teamcode\AugustJavaTraining.java - Android Studio 2.3.3

```java
/**...*/

package org.firstinspires.ftc.teamcode;

import ...

//import com.qualcomm.robotcore.util.Range;


/**
 * Common codes
 */

@TeleOp(name="Java Training CODE", group="GG")
@Disabled
public class AugustJavaTraining extends OpMode {

    /// Main timer
    ElapsedTime timer_ = new ElapsedTime();

    //Different Types of Variables!
    float floatExample = 1;
    double doubleExample= 0.0;
    int integerExample = 0;
    boolean booleanExample = true;


    /// Motors
    static final boolean USE_WHEELS = false;
```

Gradle build finished in 9s 379ms (51 minutes ago)

28:4 LF UTF-8 Context: <no context>


ftc_app-master - [C:\Users\aliri\Desktop\FTC\20171218_FTC_AppNEWEST\ftc_app-master] - [Tea

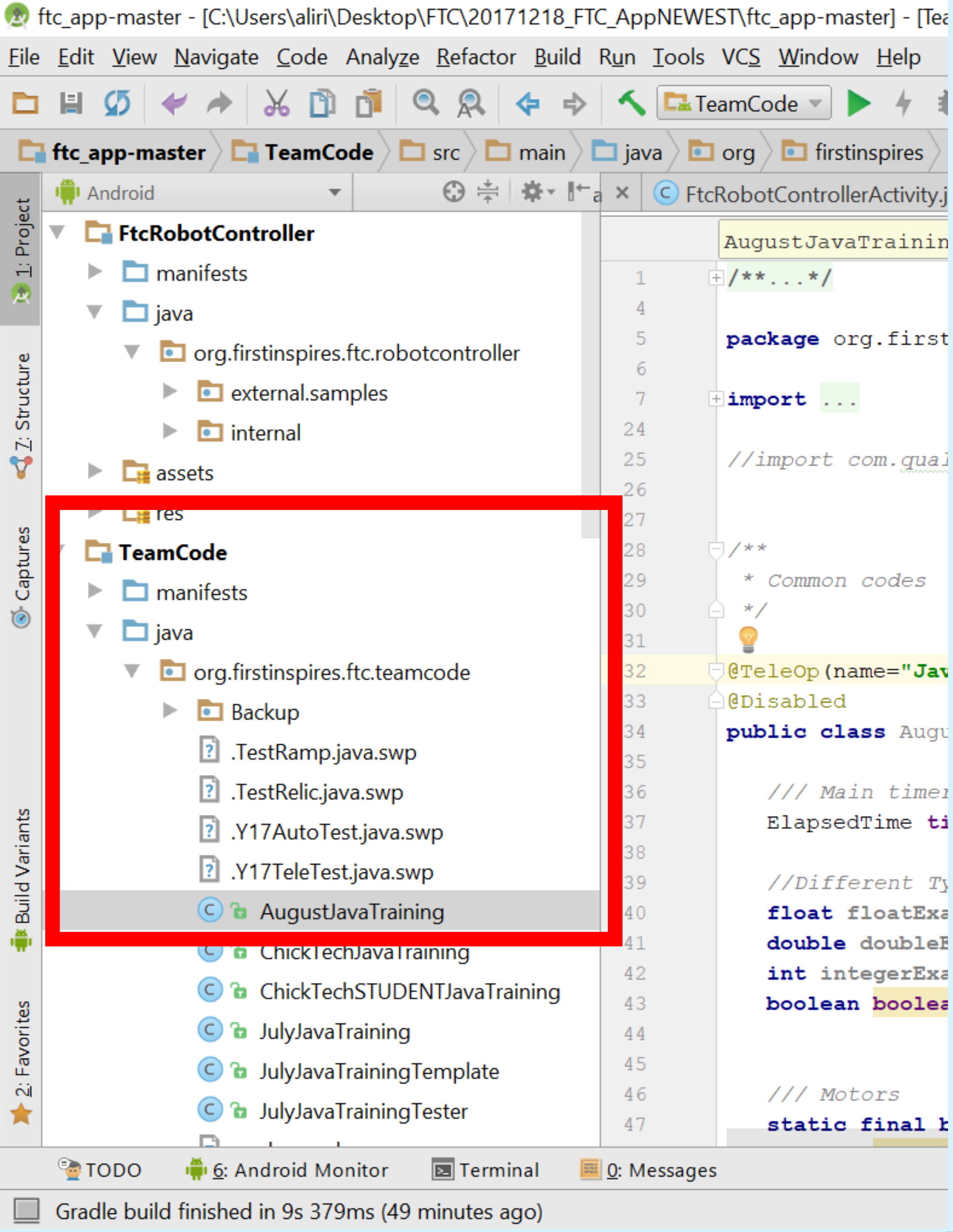
File Edit View Navigate Code Analyze Refactor Build Run Tools VCS Window Help
TeamCode
ftc_app-master > TeamCode > src > main > java > org > firstinspires
Android
FtcRobotControllerActivity.ja
1: Project
7: Structure
Captures
Build Variants
2: Favorites
TODO
6: Android Monitor
Terminal
0: Messages

```
FtcRobotController
  manifests
  java
    org.firstinspires.ftc.robotcontroller
      external.samples
        BasicOpMode_Iterative
        BasicOpMode_Linear
        ConceptCompassCalibration
        ConceptDIMAsIndicator
        ConceptI2cAddressChange
        ConceptNullOp
        ConceptRampMotorSpeed
        ConceptRegisterOpModes
        ConceptScanServo
        ConceptTelemetry
        ConceptVuforiaNavigation
        ConceptVuMarkIdentification
        HardwareK9bot
        HardwarePushbot
        HardwarePushbotMatrix
        K9botTeleopTank_Linear
        PushbotAutoDriveByEncoder_Li
        PushbotAutoDriveByGyro_Linea
```

```
AugustJavaTrainin
/**...*/

package org.first

import ...

//import com.qual

/**
 * Common codes
 */

@TeleOp(name="Jav
@Disabled
public class Augu

    /// Main timer
    ElapsedTime ti

    //Different Ty
    float floatExa
    double doubleE
    int integerExa
    boolean boolea


    /// Motors
    static final b
```

Gradle build finished in 9s 379ms (54 minutes ago)

- FtcRobotController > java
  - Tons of sample code!

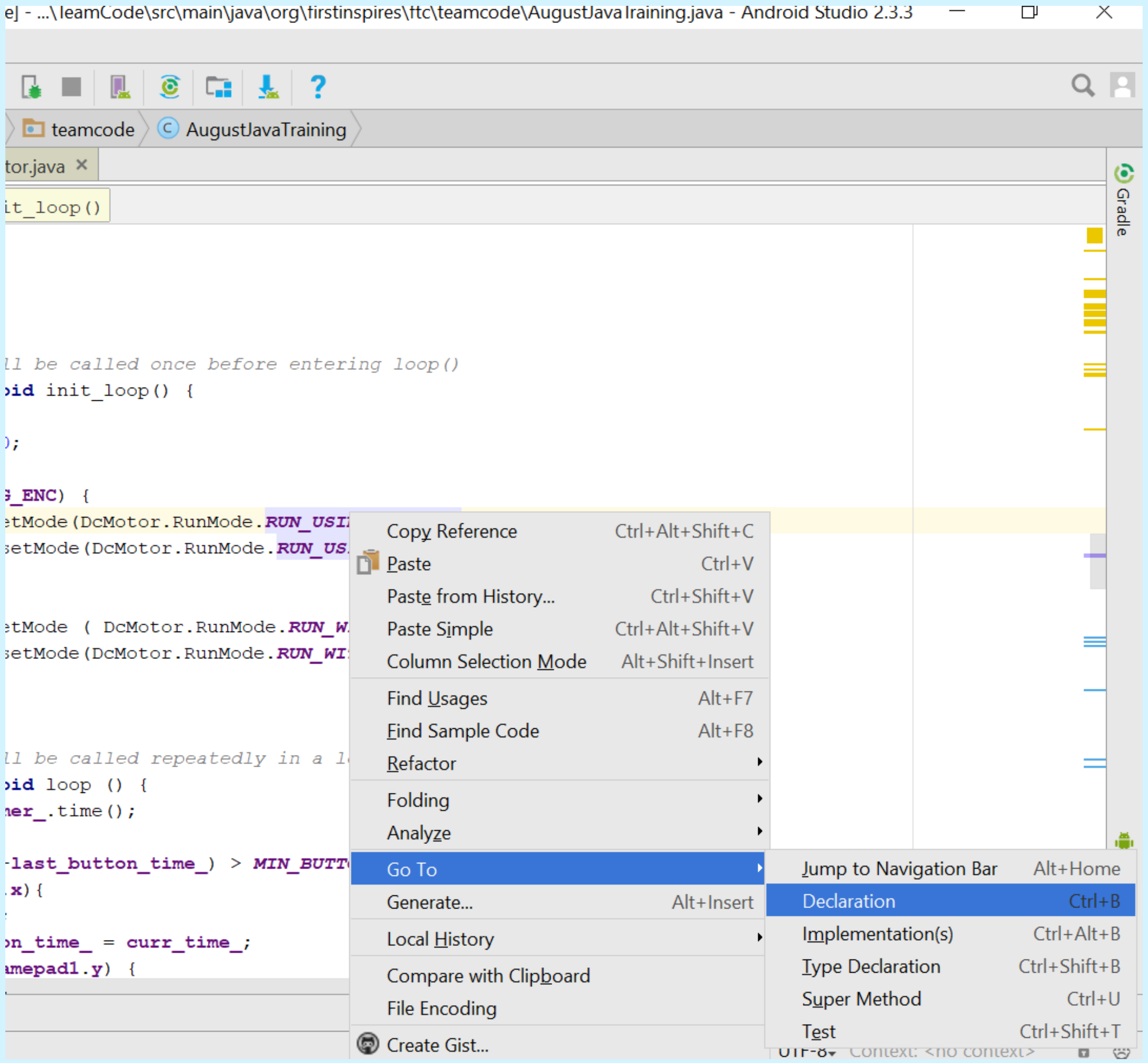

ftc_app-master - [C:\Users\aliri\Desktop\FTC\20171218_FTC_AppNEWEST\ftc_app-master] - [Tea
File Edit View Navigate Code Analyze Refactor Build Run Tools VCS Window Help
TeamCode
ftc_app-master > TeamCode > src > main > java > org > firstinspires
1: Project
7: Structure
Captures
Build Variants
2: Favorites
Android
FtcRobotControllerActivity.j
AugustJavaTrainin

```
FtcRobotController
    manifests
    java
        org.firstinspires.ftc.robotcontroller
            external.samples
            internal
    assets
    res
TeamCode
    manifests
    java
        org.firstinspires.ftc.teamcode
            Backup
            .TestRamp.java.swp
            .TestRelic.java.swp
            .Y17AutoTest.java.swp
            .Y17TeleTest.java.swp
            AugustJavaTraining
            ChickTechJavaTraining
            ChickTechSTUDENTJavaTraining
            JulyJavaTraining
            JulyJavaTrainingTemplate
            JulyJavaTrainingTester
```

```
/**...*/

package org.first

import ...

//import com.qual


/**
 * Common codes
 */

@TeleOp(name="Jav
@Disabled
public class Augu

    /// Main timer
    ElapsedTime ti

    //Different Ty
    float floatExa
    double doubleE
    int integerExa
    boolean boolea


    /// Motors
    static final b
```

Gradle build finished in 9s 379ms (49 minutes ago)

- TeamCode > java
    - Your code goes here!


e] - ...\TeamCode\src\main\java\org\firstinspires\ftc\teamcode\AugustJavaTraining.java - Android Studio 2.3.3

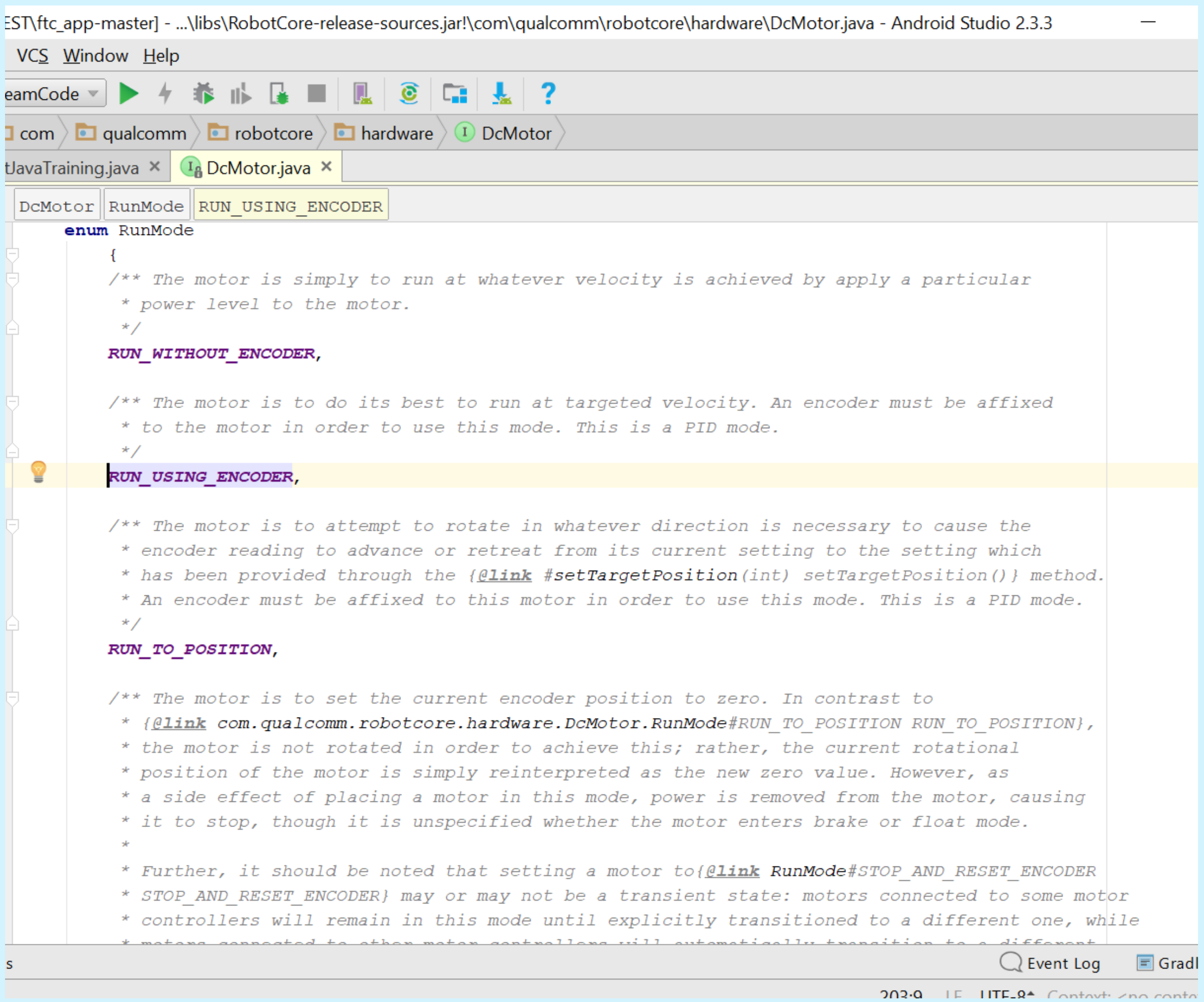
teamcode 〉 AugustJavaTraining

tor.java ×

it_loop()

```
ll be called once before entering loop()
oid init_loop() {

);

3_ENC) {
etMode(DcMotor.RunMode.RUN_USI
setMode(DcMotor.RunMode.RUN_US

etMode ( DcMotor.RunMode.RUN_W
setMode(DcMotor.RunMode.RUN_WI

ll be called repeatedly in a l
oid loop () {
ner_.time();

-last_button_time_) > MIN_BUTT
.x){

on_time_ = curr_time_;
amepad1.y) {
```

Copy Reference Ctrl+Alt+Shift+C
Paste Ctrl+V
Paste from History... Ctrl+Shift+V
Paste Simple Ctrl+Alt+Shift+V
Column Selection Mode Alt+Shift+Insert
Find Usages Alt+F7
Find Sample Code Alt+F8
Refactor
Folding
Analyze
Go To
Generate... Alt+Insert
Local History
Compare with Clipboard
File Encoding
Create Gist...
Jump to Navigation Bar Alt+Home
Declaration Ctrl+B
Implementation(s) Ctrl+Alt+B
Type Declaration Ctrl+Shift+B
Super Method Ctrl+U
Test Ctrl+Shift+T
Gradle
UTF-8 Context: <no context>


EST\ftc_app-master] - ...\libs\RobotCore-release-sources.jar!\com\qualcomm\robotcore\hardware\DcMotor.java - Android Studio 2.3.3

VCS Window Help

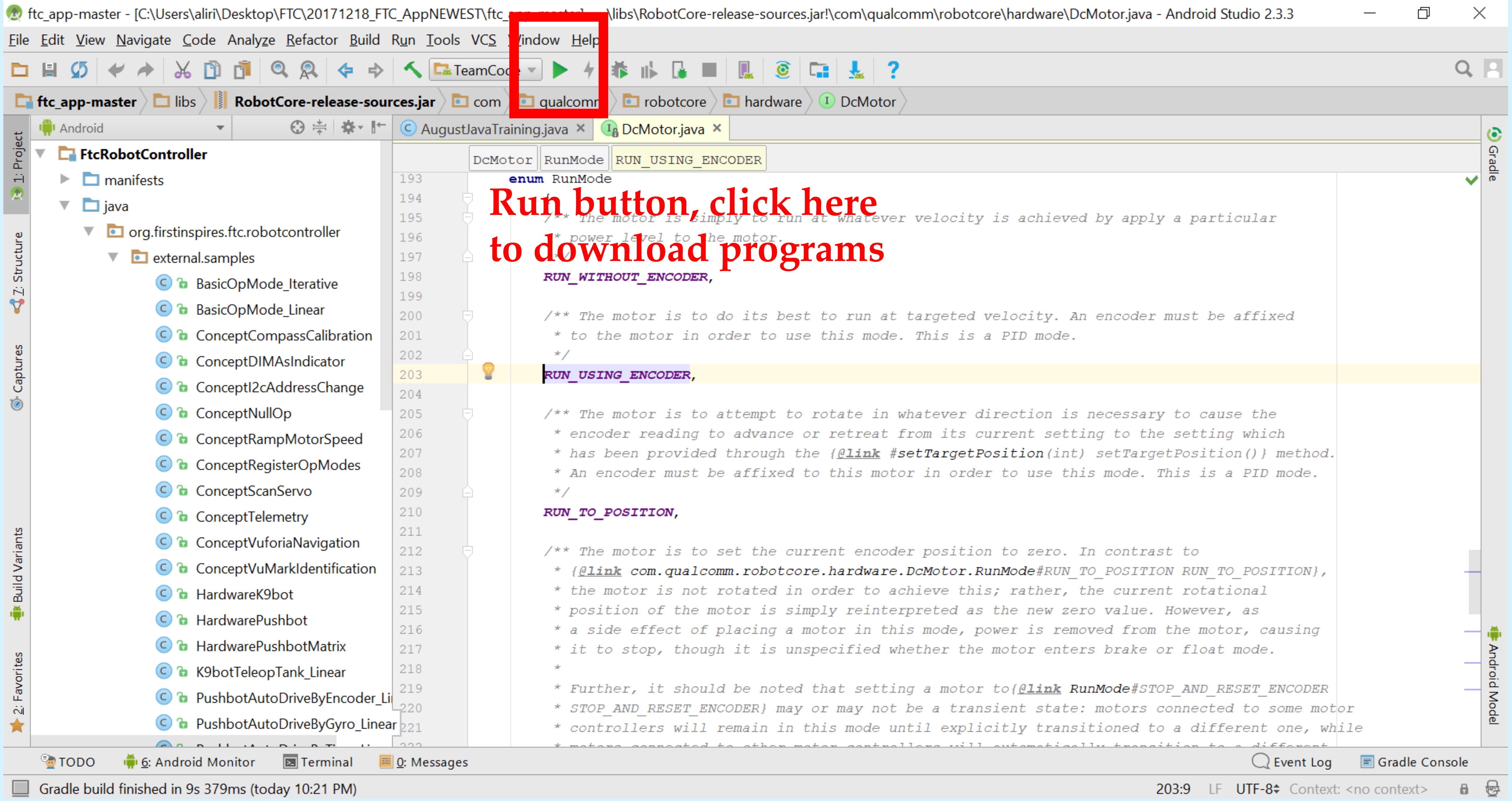
eamCode

com 〉 qualcomm 〉 robotcore 〉 hardware 〉 DcMotor

tJavaTraining.java × DcMotor.java ×

DcMotor 〉 RunMode 〉 RUN_USING_ENCODER

```
enum RunMode
    {
    /** The motor is simply to run at whatever velocity is achieved by apply a particular
     * power level to the motor.
     */
    RUN_WITHOUT_ENCODER,

    /** The motor is to do its best to run at targeted velocity. An encoder must be affixed
     * to the motor in order to use this mode. This is a PID mode.
     */
    RUN_USING_ENCODER,

    /** The motor is to attempt to rotate in whatever direction is necessary to cause the
     * encoder reading to advance or retreat from its current setting to the setting which
     * has been provided through the {@link #setTargetPosition(int) setTargetPosition()} method.
     * An encoder must be affixed to this motor in order to use this mode. This is a PID mode.
     */
    RUN_TO_POSITION,

    /** The motor is to set the current encoder position to zero. In contrast to
     * {@link com.qualcomm.robotcore.hardware.DcMotor.RunMode#RUN_TO_POSITION RUN_TO_POSITION},
     * the motor is not rotated in order to achieve this; rather, the current rotational
     * position of the motor is simply reinterpreted as the new zero value. However, as
     * a side effect of placing a motor in this mode, power is removed from the motor, causing
     * it to stop, though it is unspecified whether the motor enters brake or float mode.
     *
     * Further, it should be noted that setting a motor to{@link RunMode#STOP_AND_RESET_ENCODER
     * STOP_AND_RESET_ENCODER} may or may not be a transient state: motors connected to some motor
     * controllers will remain in this mode until explicitly transitioned to a different one, while
```


ftc_app-master - [C:\Users\aliri\Desktop\FTC\20171218_FTC_AppNEWEST\ftc_app-master] - ...\libs\RobotCore-release-sources.jar!\com\qualcomm\robotcore\hardware\DcMotor.java - Android Studio 2.3.3
File Edit View Navigate Code Analyze Refactor Build Run Tools VCS Window Help
TeamCode
ftc_app-master libs RobotCore-release-sources.jar com qualcomm robotcore hardware DcMotor
Android
AugustJavaTraining.java
DcMotor.java
DcMotor RunMode RUN_USING_ENCODER
FtcRobotController
manifests
java
org.firstinspires.ftc.robotcontroller
external.samples
BasicOpMode_Iterative
BasicOpMode_Linear
ConceptCompassCalibration
ConceptDIMAsIndicator
ConceptI2cAddressChange
ConceptNullOp
ConceptRampMotorSpeed
ConceptRegisterOpModes
ConceptScanServo
ConceptTelemetry
ConceptVuforiaNavigation
ConceptVuMarkIdentification
HardwareK9bot
HardwarePushbot
HardwarePushbotMatrix
K9botTeleopTank_Linear
PushbotAutoDriveByEncoder_Li
PushbotAutoDriveByGyro_Linear
1: Project
7: Structure
Captures
Build Variants
2: Favorites
Gradle
Android Model
TODO
6: Android Monitor
Terminal
0: Messages
Event Log
Gradle Console
203:9 LF UTF-8 Context: <no context>

Run button, click here to download programs

```
    enum RunMode
        {
        /** The motor is simply to run at whatever velocity is achieved by apply a particular
         * power level to the motor.
         */
        RUN_WITHOUT_ENCODER,

        /** The motor is to do its best to run at targeted velocity. An encoder must be affixed
         * to the motor in order to use this mode. This is a PID mode.
         */
        RUN_USING_ENCODER,

        /** The motor is to attempt to rotate in whatever direction is necessary to cause the
         * encoder reading to advance or retreat from its current setting to the setting which
         * has been provided through the {@link #setTargetPosition(int) setTargetPosition()} method.
         * An encoder must be affixed to this motor in order to use this mode. This is a PID mode.
         */
        RUN_TO_POSITION,

        /** The motor is to set the current encoder position to zero. In contrast to
         * {@link com.qualcomm.robotcore.hardware.DcMotor.RunMode#RUN_TO_POSITION RUN_TO_POSITION},
         * the motor is not rotated in order to achieve this; rather, the current rotational
         * position of the motor is simply reinterpreted as the new zero value. However, as
         * a side effect of placing a motor in this mode, power is removed from the motor, causing
         * it to stop, though it is unspecified whether the motor enters brake or float mode.
         *
         * Further, it should be noted that setting a motor to{@link RunMode#STOP_AND_RESET_ENCODER
         * STOP_AND_RESET_ENCODER} may or may not be a transient state: motors connected to some motor
         * controllers will remain in this mode until explicitly transitioned to a different one, while
```

Gradle build finished in 9s 379ms (today 10:21 PM)

OpModes

OpMode Run Structure

- Every program you create is an "OpMode"
  - Base class for user defined operation modes
  - Template for all your programs
  - Prewritten functions

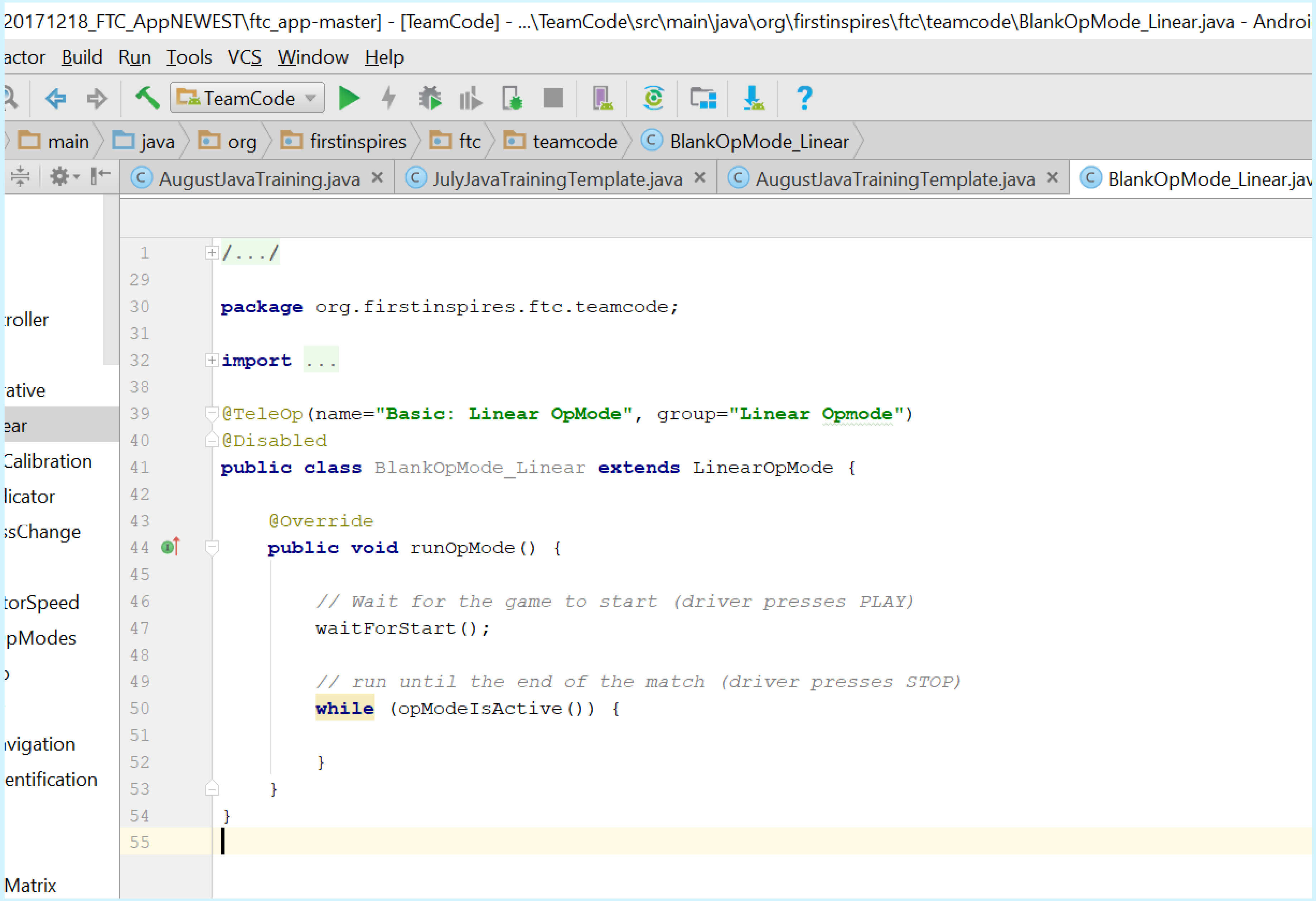
- 2 different OpModes: LinearOpMode, OpMode
  - Differ in run cycle (how the program is run) and some of the tools you can use

LinearOpMode v. OpMode

- Code falls within 1 main function (runOpMode)
  - Once init is pressed, your code runs once from start to finish

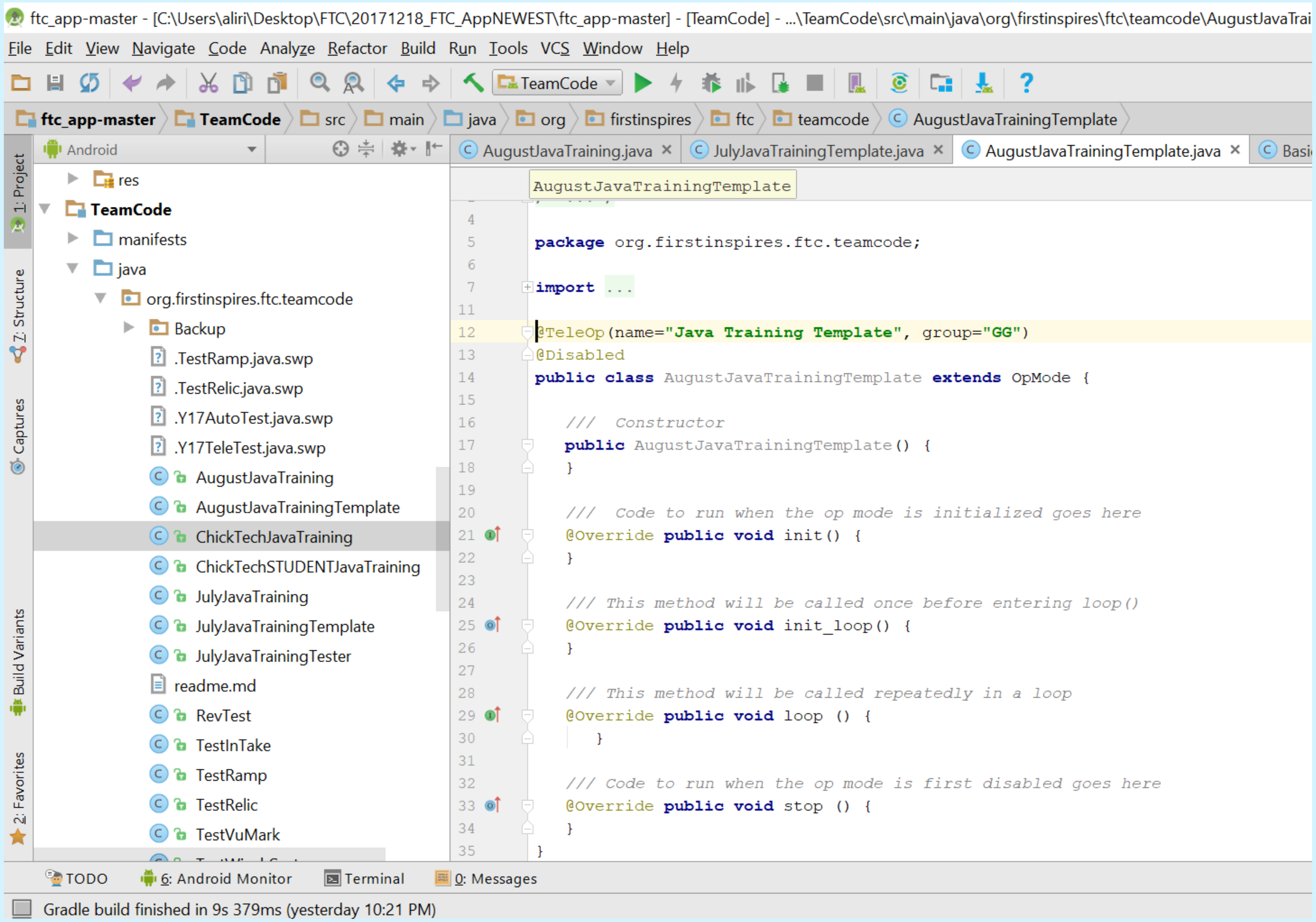
- 5 main functions (init, init loop, start, loop, stop)
- Code in each function runs at different times
  - Once init is pressed, only code from init and init loop runs
  - Once play is pressed, code in start and loop runs
  - Code in stop function runs when stop is pressed

LinearOpMode


20171218_FTC_AppNEWEST\ftc_app-master] - [TeamCode] - ...\TeamCode\src\main\java\org\firstinspires\ftc\teamcode\BlankOpMode_Linear.java - Androi
actor Build Run Tools VCS Window Help
TeamCode
main java org firstinspires ftc teamcode BlankOpMode_Linear
AugustJavaTraining.java
JulyJavaTrainingTemplate.java
AugustJavaTrainingTemplate.java
BlankOpMode_Linear.jav
roller
ative
ear
Calibration
licator
sChange
torSpeed
pModes
vigation
entification
Matrix

```java
/.../

package org.firstinspires.ftc.teamcode;

import ...

@TeleOp(name="Basic: Linear OpMode", group="Linear Opmode")
@Disabled
public class BlankOpMode_Linear extends LinearOpMode {

    @Override
    public void runOpMode() {

        // Wait for the game to start (driver presses PLAY)
        waitForStart();

        // run until the end of the match (driver presses STOP)
        while (opModeIsActive()) {

        }
    }
}
```

OpMode

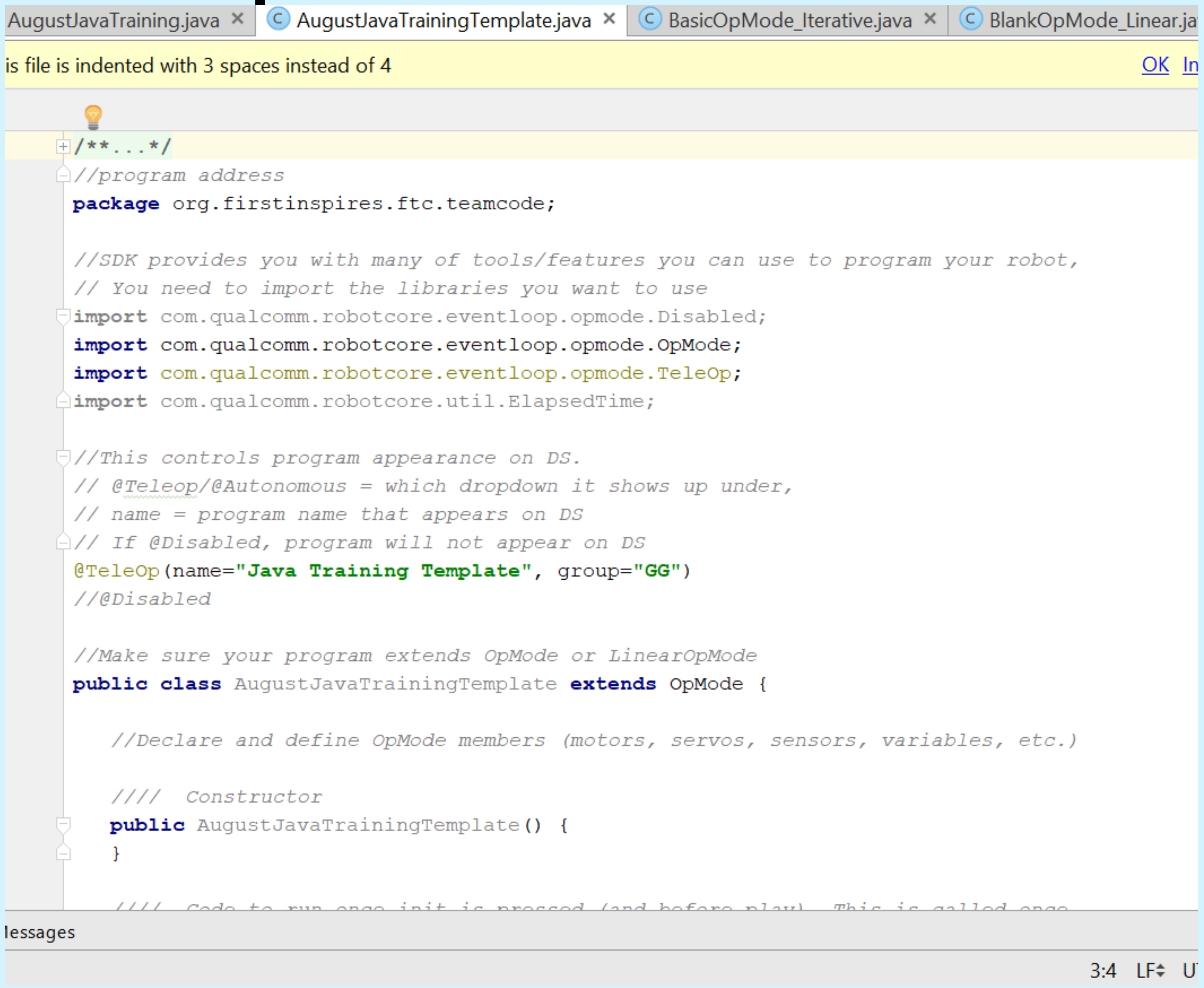


ftc_app-master - [C:\Users\aliri\Desktop\FTC\20171218_FTC_AppNEWEST\ftc_app-master] - [TeamCode] - ...\TeamCode\src\main\java\org\firstinspires\ftc\teamcode\AugustJavaTrai

```java
package org.firstinspires.ftc.teamcode;

import ...

@TeleOp(name="Java Training Template", group="GG")
@Disabled
public class AugustJavaTrainingTemplate extends OpMode {

    ///  Constructor
    public AugustJavaTrainingTemplate() {
    }

    ///  Code to run when the op mode is initialized goes here
    @Override public void init() {
    }

    /// This method will be called once before entering loop()
    @Override public void init_loop() {
    }

    /// This method will be called repeatedly in a loop
    @Override public void loop () {
        }

    /// Code to run when the op mode is first disabled goes here
    @Override public void stop () {
    }
}
```

Gradle build finished in 9s 379ms (yesterday 10:21 PM)

OpMode Overview/Structure



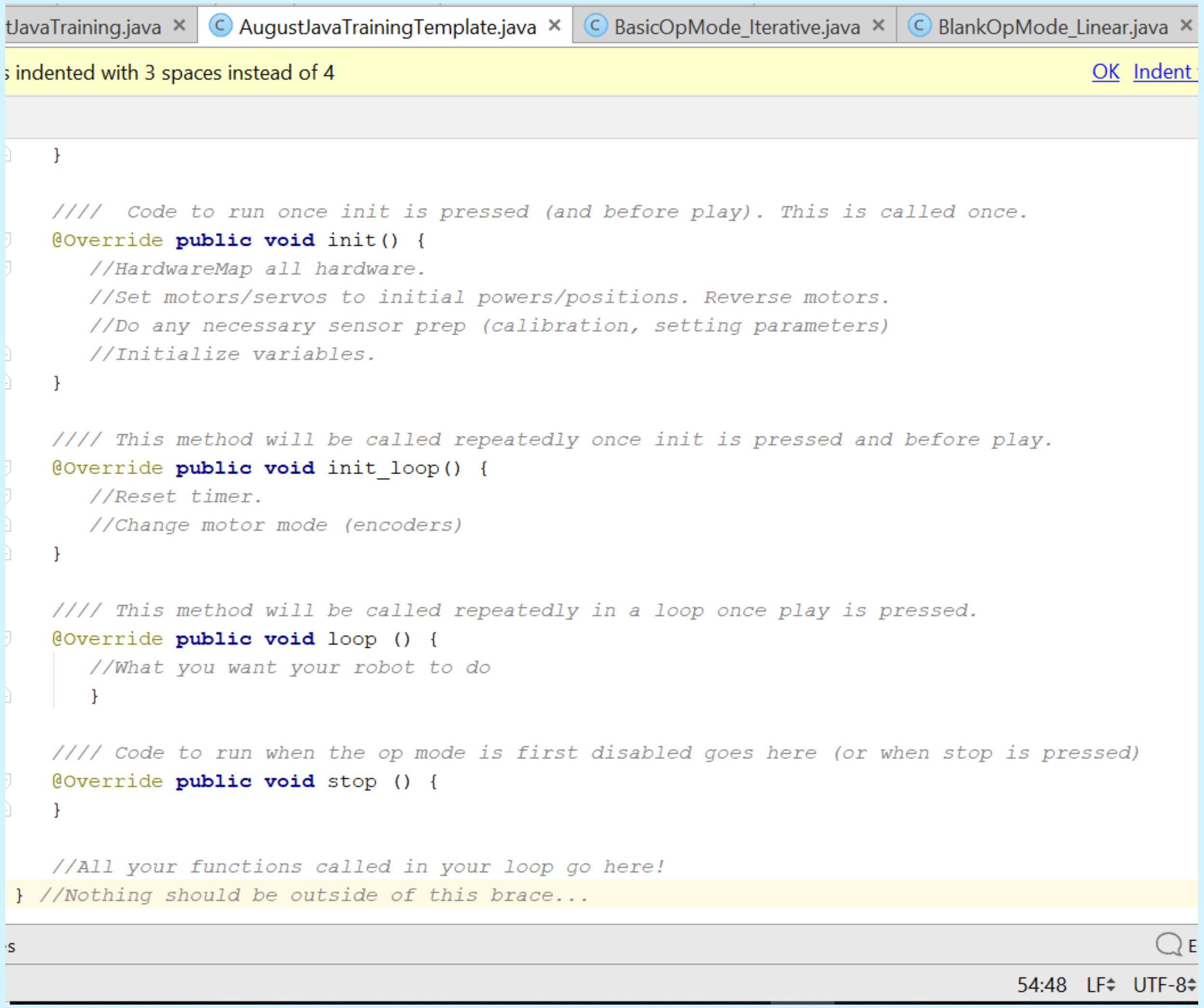
AugustJavaTraining.java × | AugustJavaTrainingTemplate.java × | BasicOpMode_Iterative.java × | BlankOpMode_Linear.ja

is file is indented with 3 spaces instead of 4 OK In

```java
/**...*/
//program address
package org.firstinspires.ftc.teamcode;

//SDK provides you with many of tools/features you can use to program your robot,
// You need to import the libraries you want to use
import com.qualcomm.robotcore.eventloop.opmode.Disabled;
import com.qualcomm.robotcore.eventloop.opmode.OpMode;
import com.qualcomm.robotcore.eventloop.opmode.TeleOp;
import com.qualcomm.robotcore.util.ElapsedTime;

//This controls program appearance on DS.
// @Teleop/@Autonomous = which dropdown it shows up under,
// name = program name that appears on DS
// If @Disabled, program will not appear on DS
@TeleOp(name="Java Training Template", group="GG")
//@Disabled

//Make sure your program extends OpMode or LinearOpMode
public class AugustJavaTrainingTemplate extends OpMode {

   //Declare and define OpMode members (motors, servos, sensors, variables, etc.)

   ////  Constructor
   public AugustJavaTrainingTemplate() {
   }
```

lessages

3:4 LF÷ U



JavaTraining.java × | AugustJavaTrainingTemplate.java × | BasicOpMode_Iterative.java × | BlankOpMode_Linear.java ×

; indented with 3 spaces instead of 4 OK Indent

```java
   }

   ////  Code to run once init is pressed (and before play). This is called once.
   @Override public void init() {
      //HardwareMap all hardware.
      //Set motors/servos to initial powers/positions. Reverse motors.
      //Do any necessary sensor prep (calibration, setting parameters)
      //Initialize variables.
   }

   //// This method will be called repeatedly once init is pressed and before play.
   @Override public void init_loop() {
      //Reset timer.
      //Change motor mode (encoders)
   }

   //// This method will be called repeatedly in a loop once play is pressed.
   @Override public void loop () {
      //What you want your robot to do
      }

   //// Code to run when the op mode is first disabled goes here (or when stop is pressed)
   @Override public void stop () {
   }

   //All your functions called in your loop go here!
} //Nothing should be outside of this brace...
```

54:48 LF÷ UTF-8÷

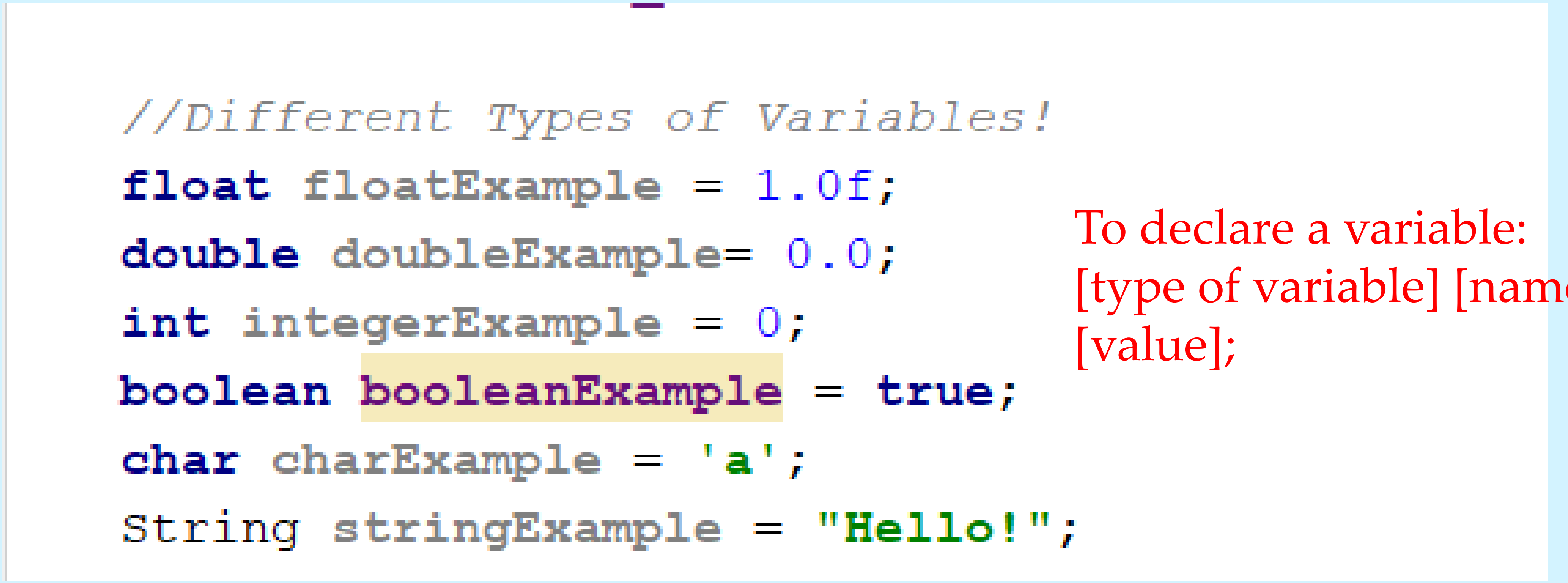
8949 The Gifted Gears

Java Basics

Syntax, General Guidelines

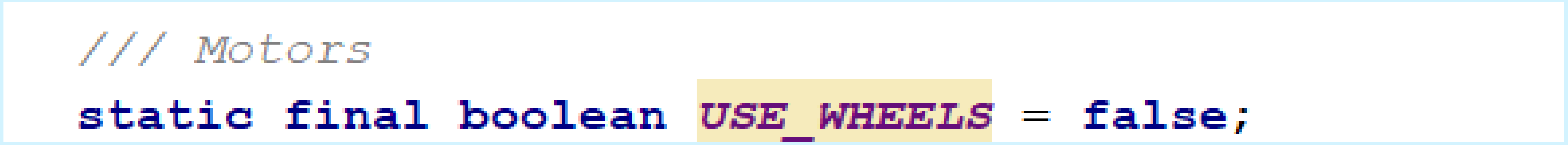
- Semicolons ;
  - End of every argument
- Squiggly Brackets {}
  - Surround body of a class/function
  - Keep track of your brackets!
- Parentheses ()
  - Conditions, function parameters
- Ways of commenting
  - // asldfkj
  - /* asdlkfj */

Variables



```
//Different Types of Variables!
float floatExample = 1.0f;
double doubleExample= 0.0;
int integerExample = 0;
boolean booleanExample = true;
char charExample = 'a';
String stringExample = "Hello!";
```

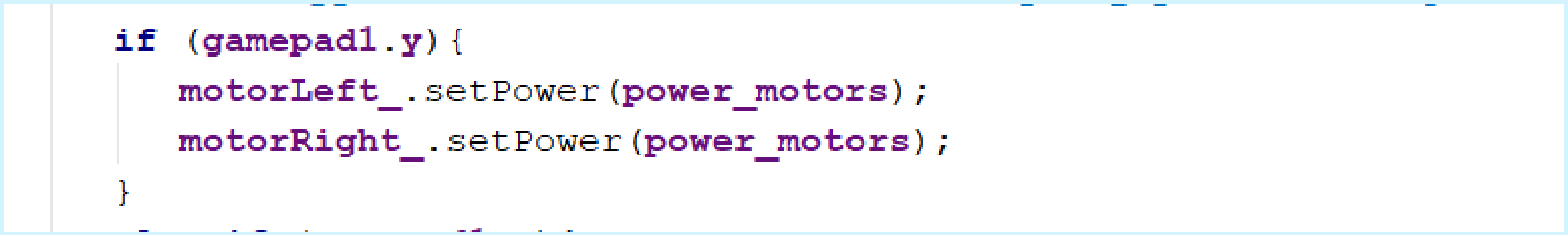
To declare a variable:
[type of variable] [name of variable] = [value];



```
/// Motors
static final boolean USE_WHEELS = false;
```

"static final" = constant, cannot be updated later in the code

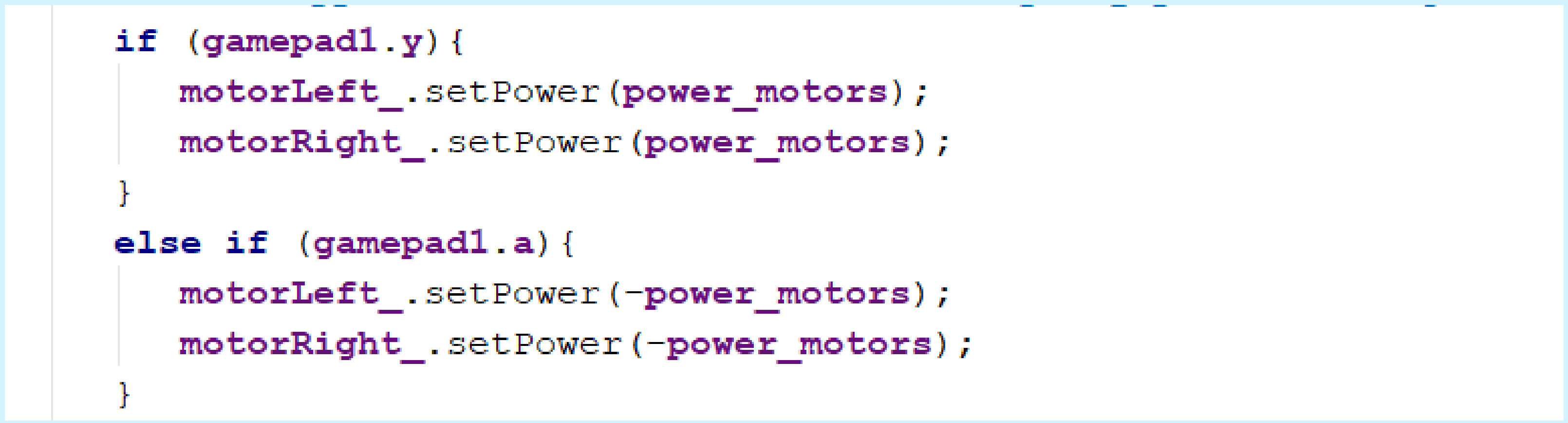
Control Flow Statements: If



```
if (gamepad1.y){
    motorLeft_.setPower(power_motors);
    motorRight_.setPower(power_motors);
}
```

```
if ([insert condition]) {

[execute this code if condition is true,
if false, skip over this if statement and
execute the code after it]

}
```

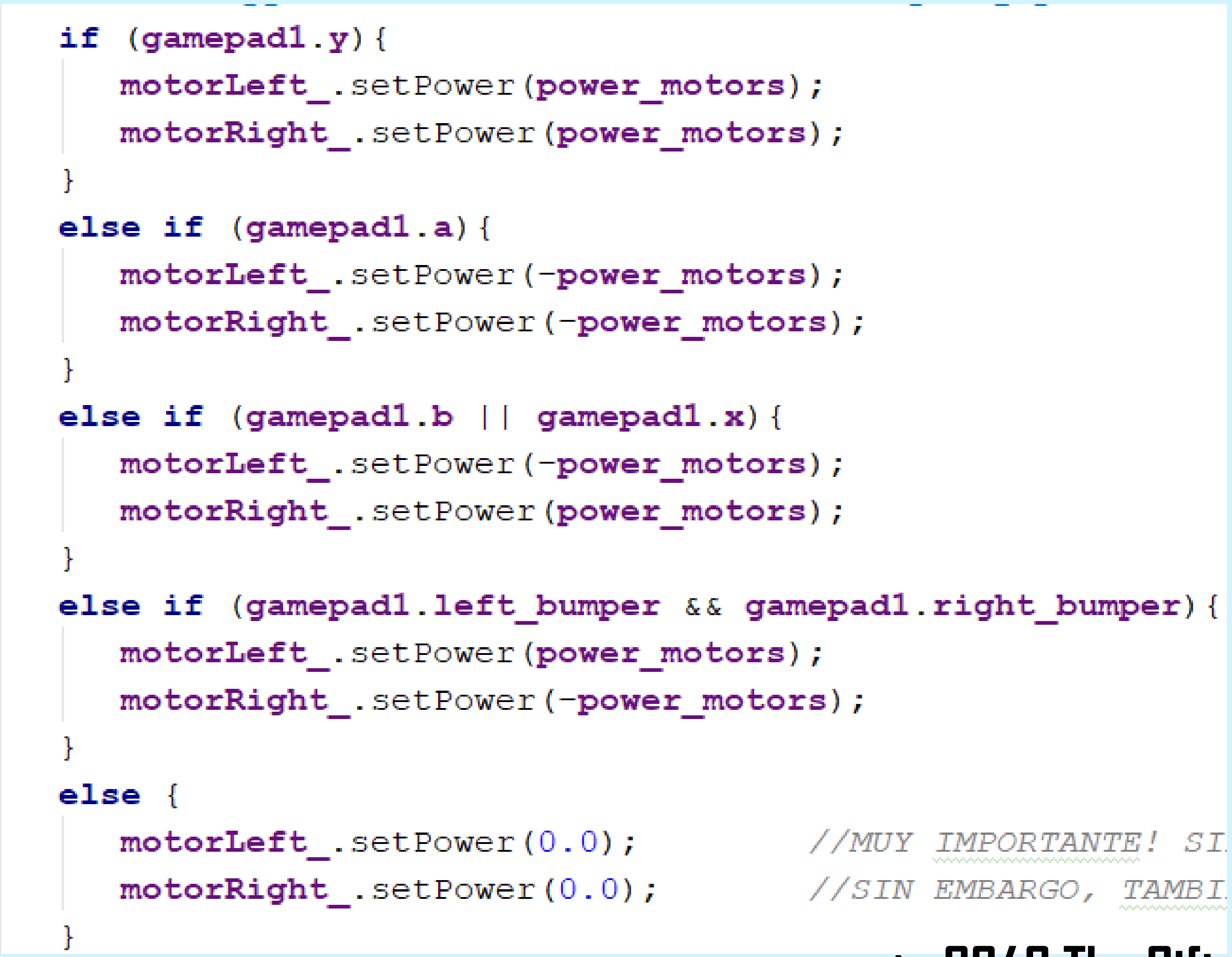
Control Flow Statements: If, Else If



```
if (gamepad1.y){
    motorLeft_.setPower(power_motors);
    motorRight_.setPower(power_motors);
}
else if (gamepad1.a){
    motorLeft_.setPower(-power_motors);
    motorRight_.setPower(-power_motors);
}
```

```
if ([insert condition]) {
        [do this]
}
else if ([insert different condition]) {
        [if 1st condition is false but 2nd
condition is true, do this]
}
```

Control Flow Statements: If, Else If, Else

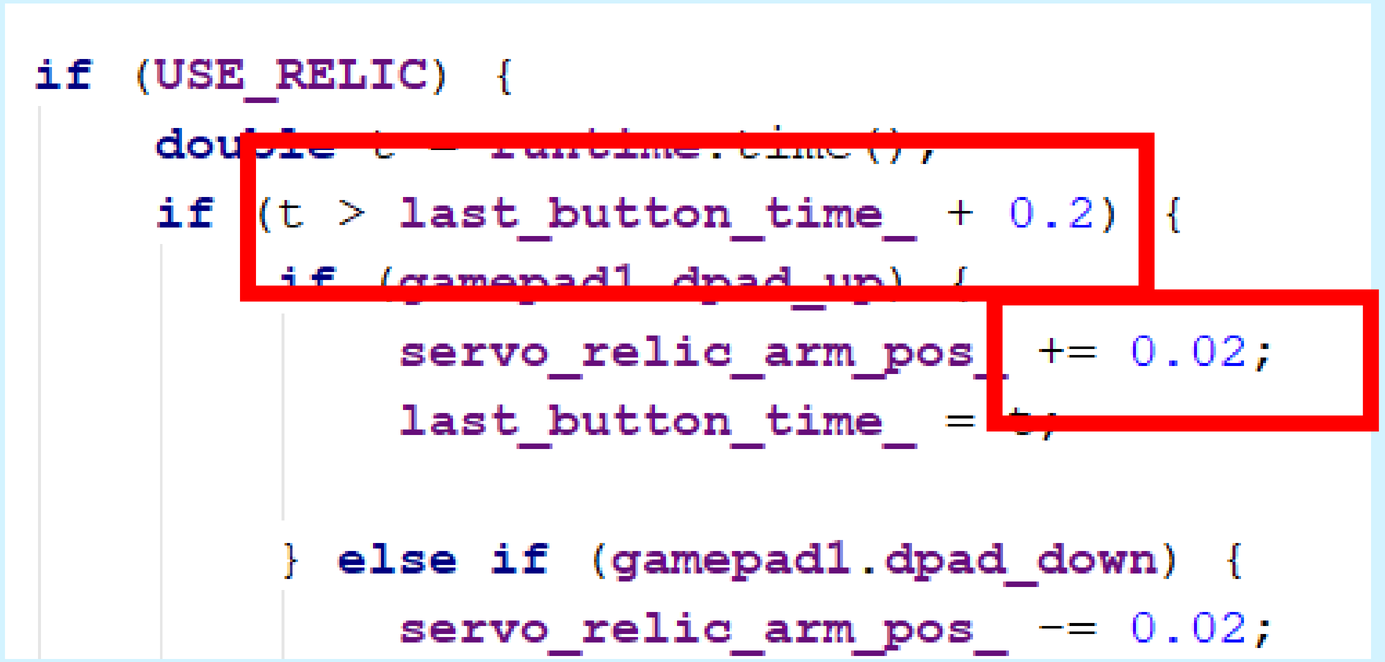


```
if (gamepad1.y){
    motorLeft_.setPower(power_motors);
    motorRight_.setPower(power_motors);
}
else if (gamepad1.a){
    motorLeft_.setPower(-power_motors);
    motorRight_.setPower(-power_motors);
}
else if (gamepad1.b || gamepad1.x){
    motorLeft_.setPower(-power_motors);
    motorRight_.setPower(power_motors);
}
else if (gamepad1.left_bumper && gamepad1.right_bumper){
    motorLeft_.setPower(power_motors);
    motorRight_.setPower(-power_motors);
}
else {
    motorLeft_.setPower(0.0);        //MUY IMPORTANTE! SI
    motorRight_.setPower(0.0);       //SIN EMBARGO, TAMBI
}
```

```
if ([insert condition]) {
        [do this]
}
else if ([insert different condition]) {
        [if 1st condition is false but 2nd condition is true, do this]
}
else {
        [none of the conditions above are true, do this – no condition]
}
```

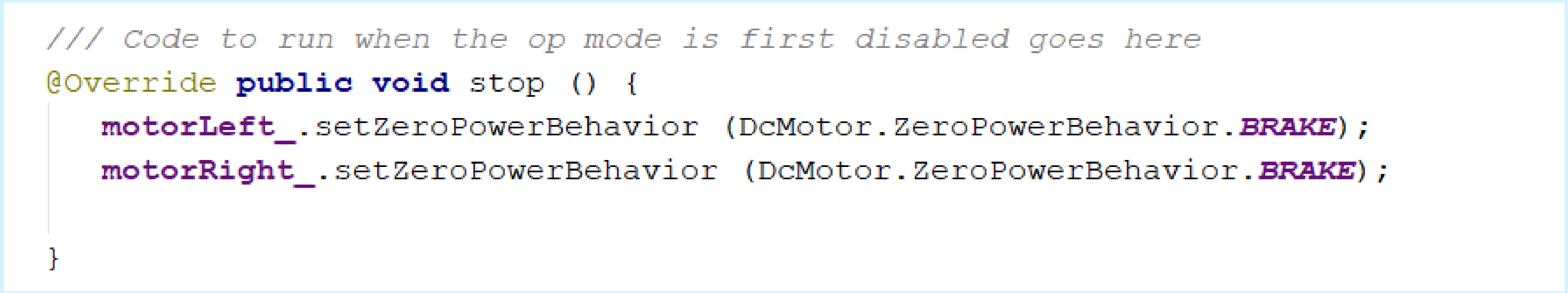
Math Functions + Conditionals

- Math Operations
  - + - * / %
  - Shorthand: += -= *= /=
  - x += 1 equivalent to x = x + 1
  - When changing variable value/setting it equal to something, updated value is on LEFT
- Conditionals
  - < <= >= >
  - && || == !=



```
if (USE_RELIC) {
    double t = runtime.time();
    if (t > last_button_time_ + 0.2) {
        if (gamepad1.dpad_up) {
            servo_relic_arm_pos_ += 0.02;
            last_button_time_ = t;

        } else if (gamepad1.dpad_down) {
            servo_relic_arm_pos_ -= 0.02;
```

Functions



```
/// Code to run when the op mode is first disabled goes here
@Override public void stop () {
   motorLeft_.setZeroPowerBehavior (DcMotor.ZeroPowerBehavior.BRAKE);
   motorRight_.setZeroPowerBehavior (DcMotor.ZeroPowerBehavior.BRAKE);

}
```

To declare a function:
[date type of return value] [name of function] ([insert function parameters, if any]) {

        [what your function does]

return [value you’re returning];
}

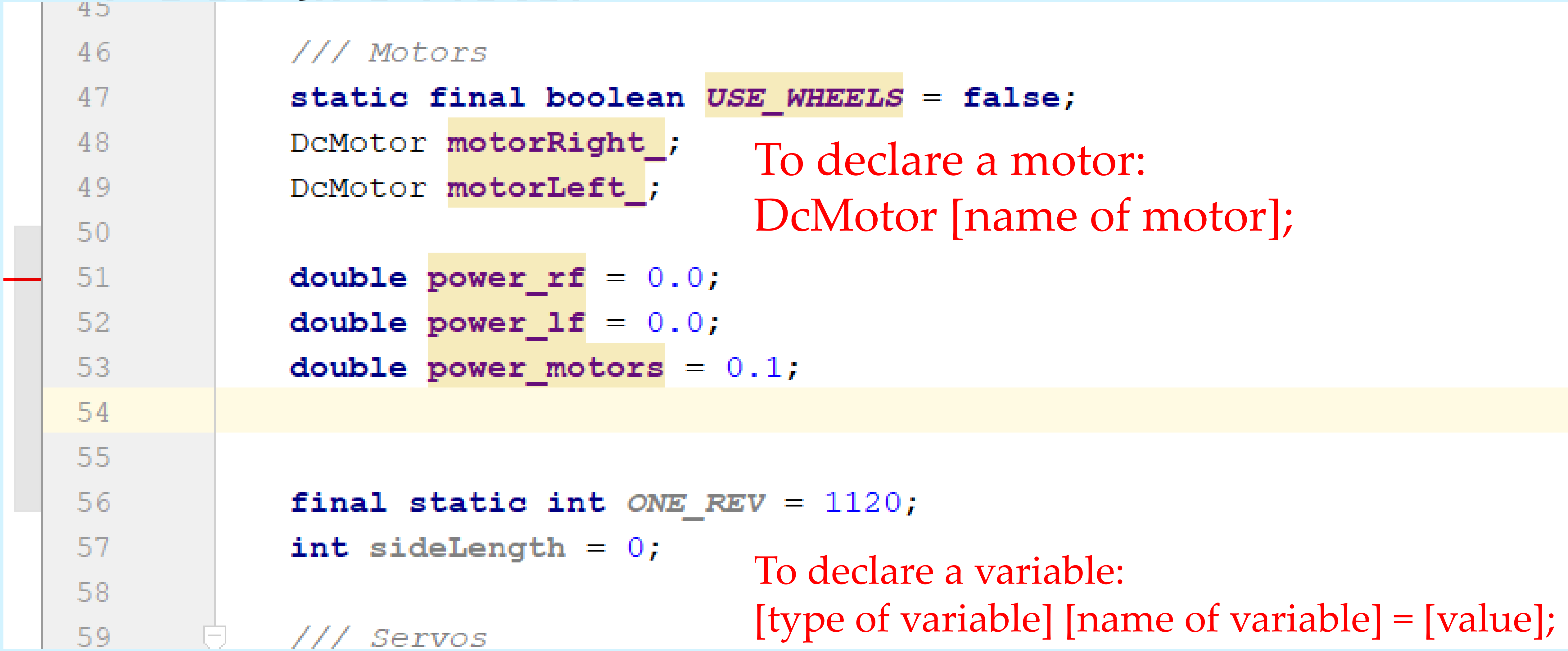
*** if there’s no return value, put void in front of function name

Motors

Overview

1. Pre-Init
   1. Declare motor and any variables associated with it
2. Init
   1. hardwareMap
   2. Reverse
3. Play
   1. setPower
   2. Range = -1 to 1

1. Declare Motor



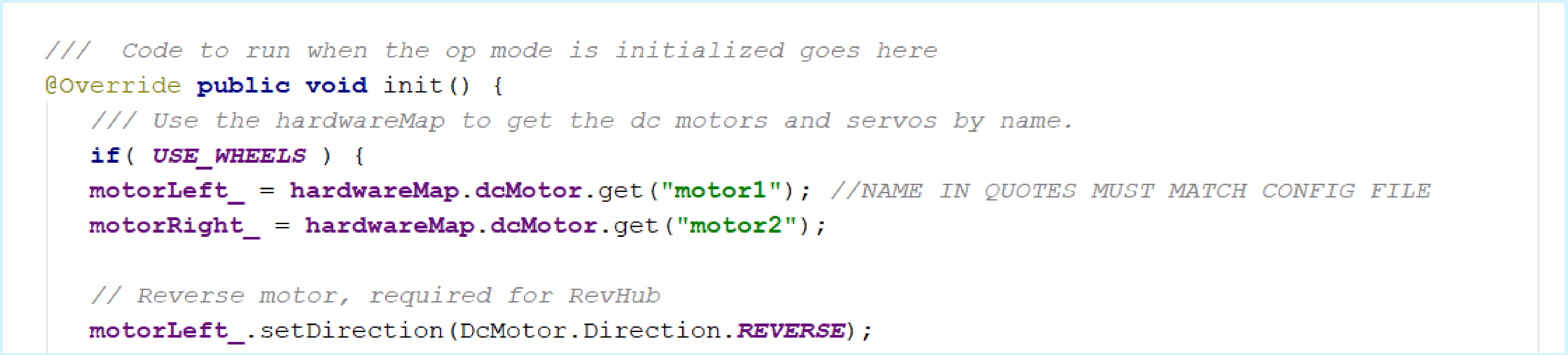
```
/// Motors
static final boolean USE_WHEELS = false;
DcMotor motorRight_;
DcMotor motorLeft_;

double power_rf = 0.0;
double power_lf = 0.0;
double power_motors = 0.1;


final static int ONE_REV = 1120;
int sideLength = 0;

/// Servos
```

To declare a motor:
DcMotor [name of motor];

To declare a variable:
[type of variable] [name of variable] = [value];

8949 The Gifted Gears

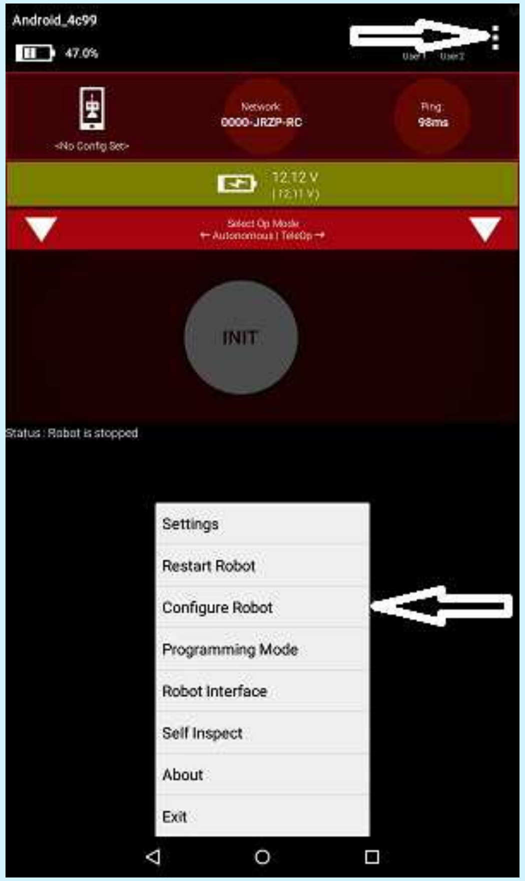
2. Initialize Motor

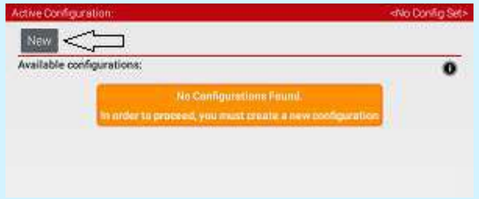


```
///  Code to run when the op mode is initialized goes here
@Override public void init() {
   /// Use the hardwareMap to get the dc motors and servos by name.
   if( USE_WHEELS ) {
   motorLeft_ = hardwareMap.dcMotor.get("motor1"); //NAME IN QUOTES MUST MATCH CONFIG FILE
   motorRight_ = hardwareMap.dcMotor.get("motor2");

   // Reverse motor, required for RevHub
   motorLeft_.setDirection(DcMotor.Direction.REVERSE);
```

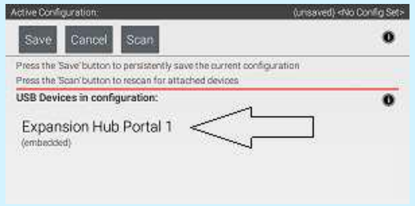
hardwareMap connects the object in the program to the physical object
Name in " " MUST match what's entered in the configuration file, else robot will tell there's an error

Configuration

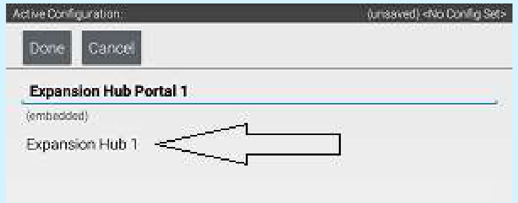


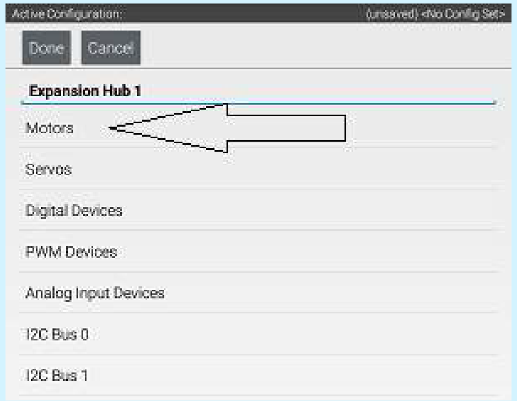


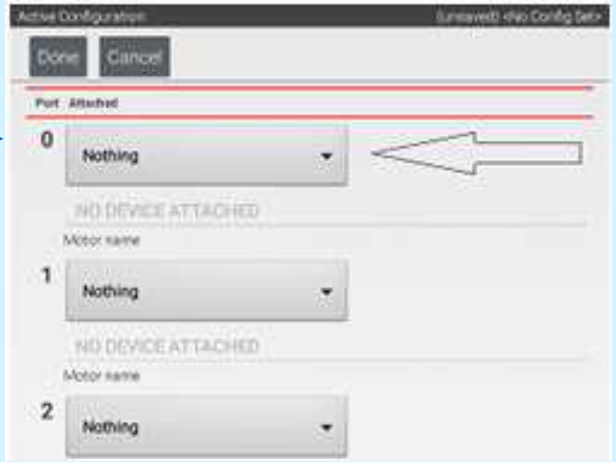
No Configurations Found.
In order to proceed, you must create a new configuration



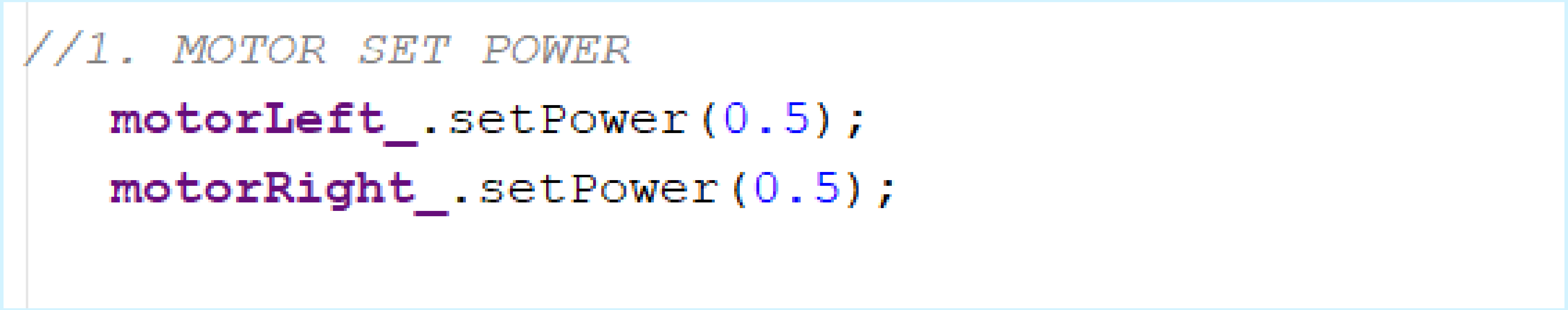
Press the 'Save' button to persistently save the current configuration
Press the 'Scan' button to rescan for attached devices







3. setPower



```
//1. MOTOR SET POWER
    motorLeft_.setPower(0.5);
    motorRight_.setPower(0.5);
```

[name of motor].setPower(double value);
Motor powers range from -1 to 1

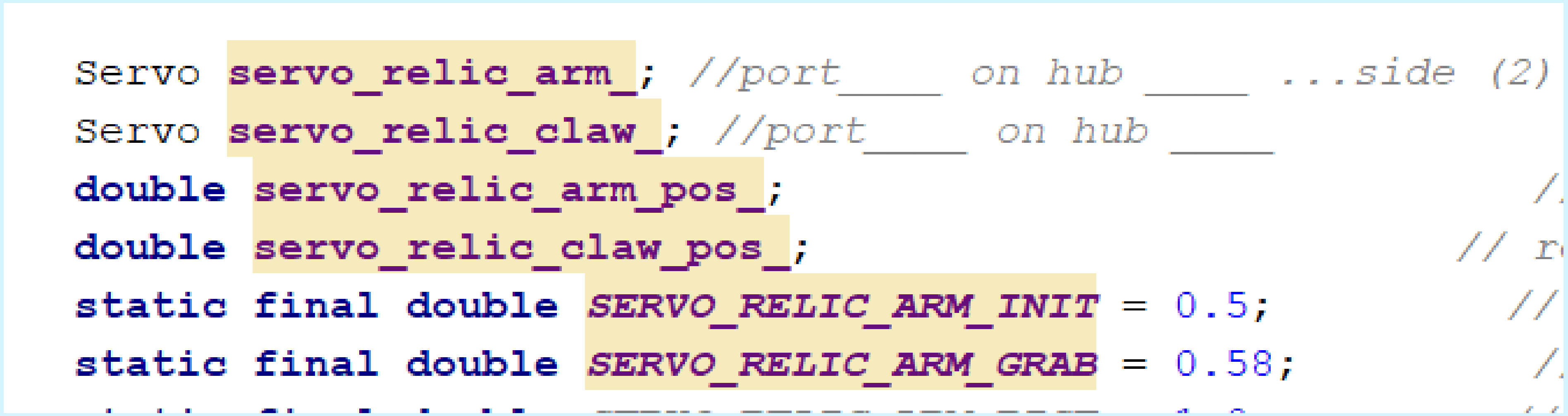
Servos

180 and Continuous Rotation (CR)

Overview

1. Pre-Init
    1. Declare servo and any variables associated with it
2. Init
    1. hardwareMap
    2. Set initial position
3. Play
    1. setPosition
    2. Range = 0 to 1 on both 180 and CR
        1. CR: 0 = full power CW, 0.5 = stopped, 1.0 = full power CCW

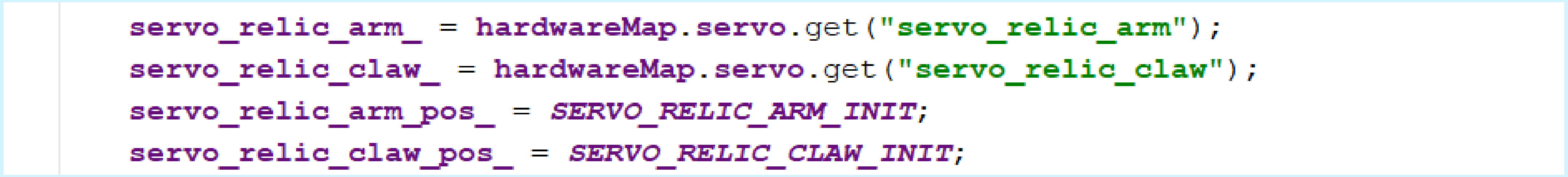
1. Declare Servo



```
Servo servo_relic_arm_; //port____ on hub ____ ...side (2)
Servo servo_relic_claw_; //port____ on hub ____
double servo_relic_arm_pos_;                                /
double servo_relic_claw_pos_;                            // r
static final double SERVO_RELIC_ARM_INIT = 0.5;           //
static final double SERVO_RELIC_ARM_GRAB = 0.58;           /
```

To declare a servo:
Servo [name of servo];

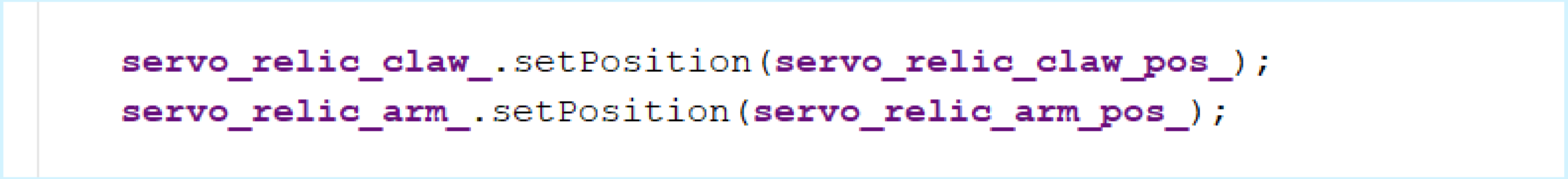
2. Initialize Servo



```
servo_relic_arm_ = hardwareMap.servo.get("servo_relic_arm");
servo_relic_claw_ = hardwareMap.servo.get("servo_relic_claw");
servo_relic_arm_pos_ = SERVO_RELIC_ARM_INIT;
servo_relic_claw_pos_ = SERVO_RELIC_CLAW_INIT;
```

hardwareMap connects the object in the program to the physical object
Name in " " MUST match what's entered in the configuration file
Generally good practice to set your servo to an initial position; if you don't set, it will go to 0 position by default

3. setPosition



```
servo_relic_claw_.setPosition(servo_relic_claw_pos_);
servo_relic_arm_.setPosition(servo_relic_arm_pos_);
```

[name of servo].setPosition(double value);
Position ranges from 0 to 1
IF it's a continuous rotation servo:
0 = rotating CW at full speed, 0.5 = stopped, and 1 = rotating CCW at full speed

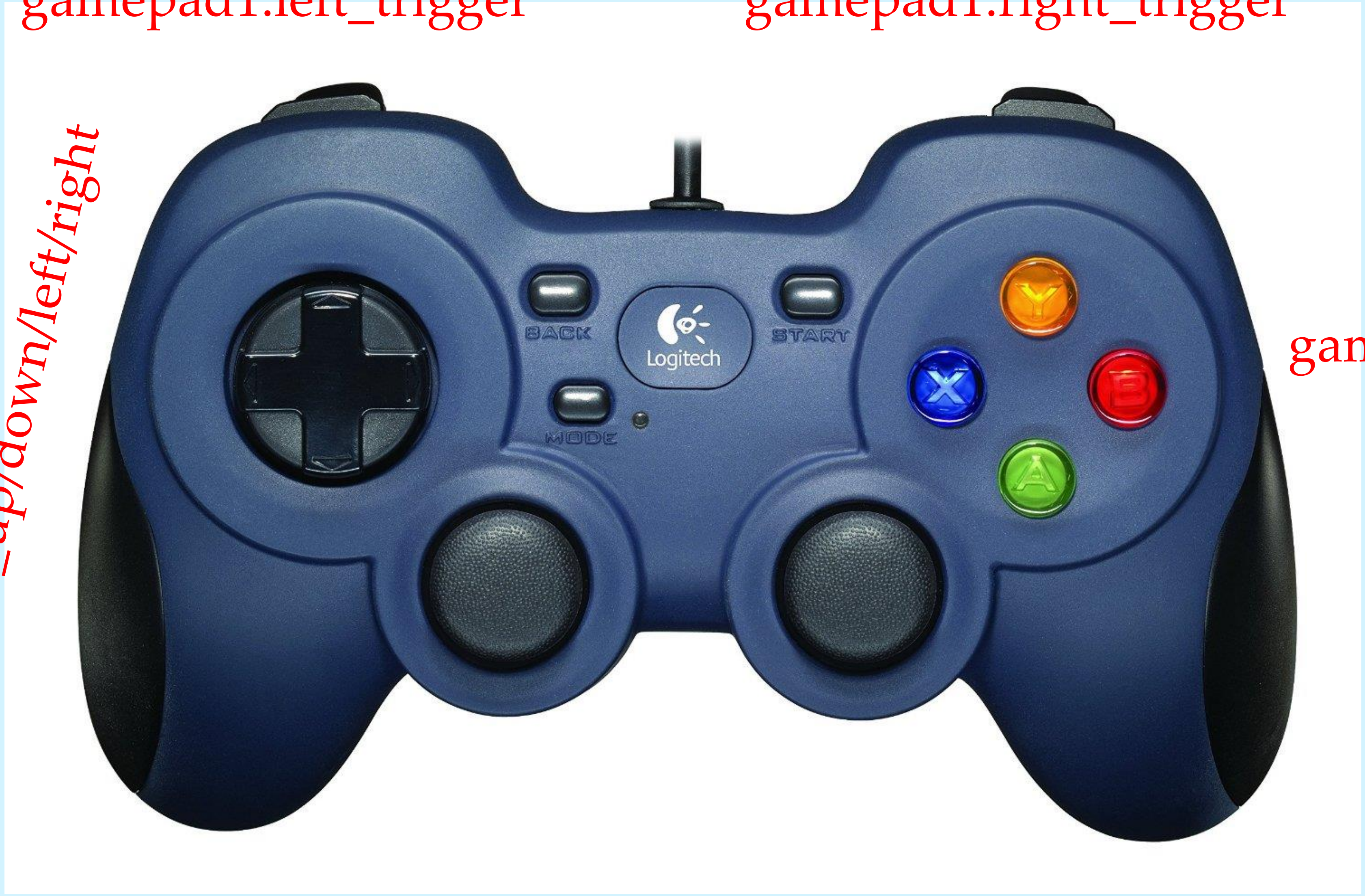
Application: Teleop

gamepad1.left_bumper
gamepad1.left_trigger

gamepad1.right_bumper
gamepad1.right_trigger

***
Buttons = Booleans
Joysticks and Triggers = Returns number value

gamepad1.dpad_up/down/left/right



gamepad1.a/b/x/y

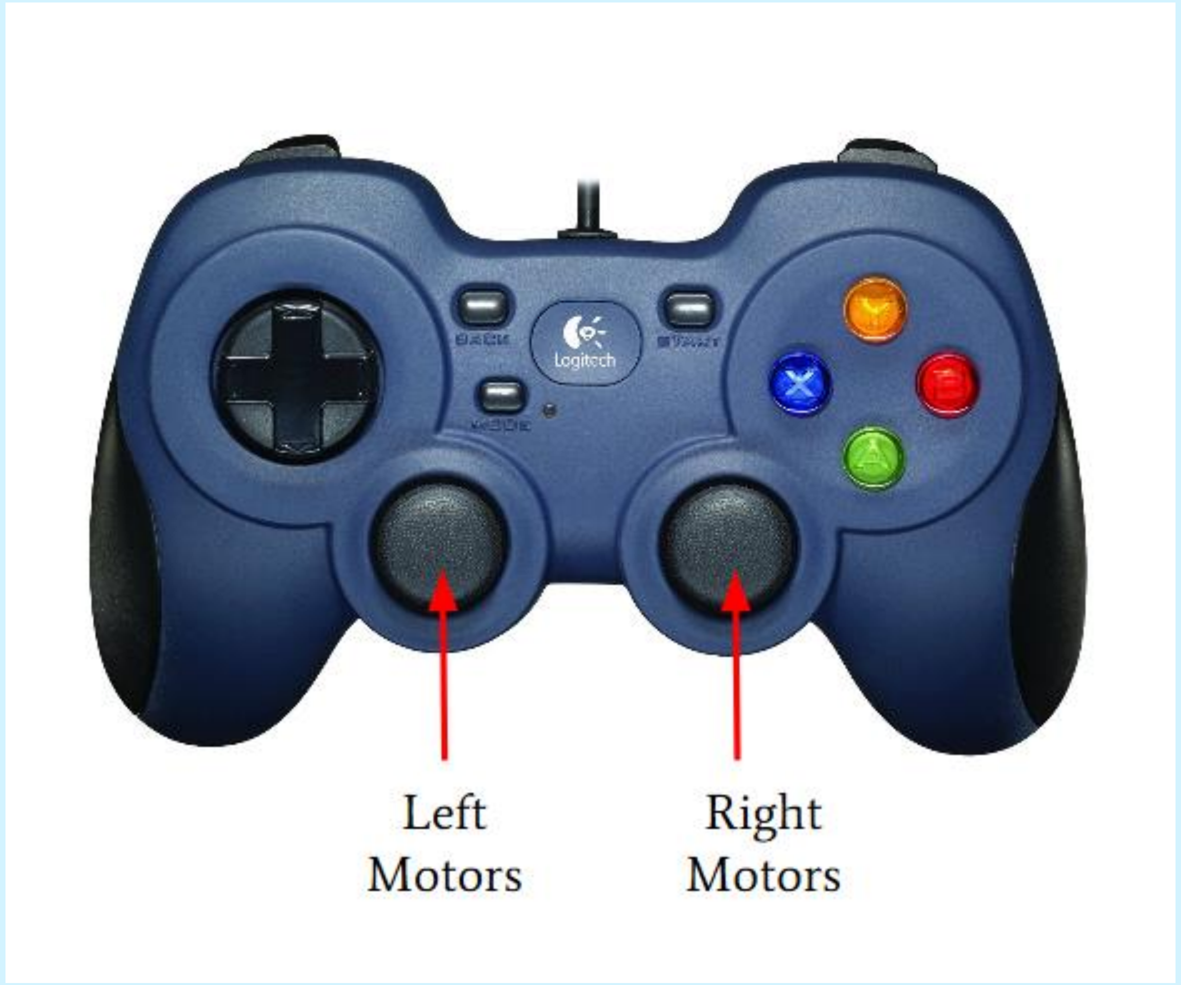
gamepad1.left_stick_x/y

gamepad1.right_stick_x/y

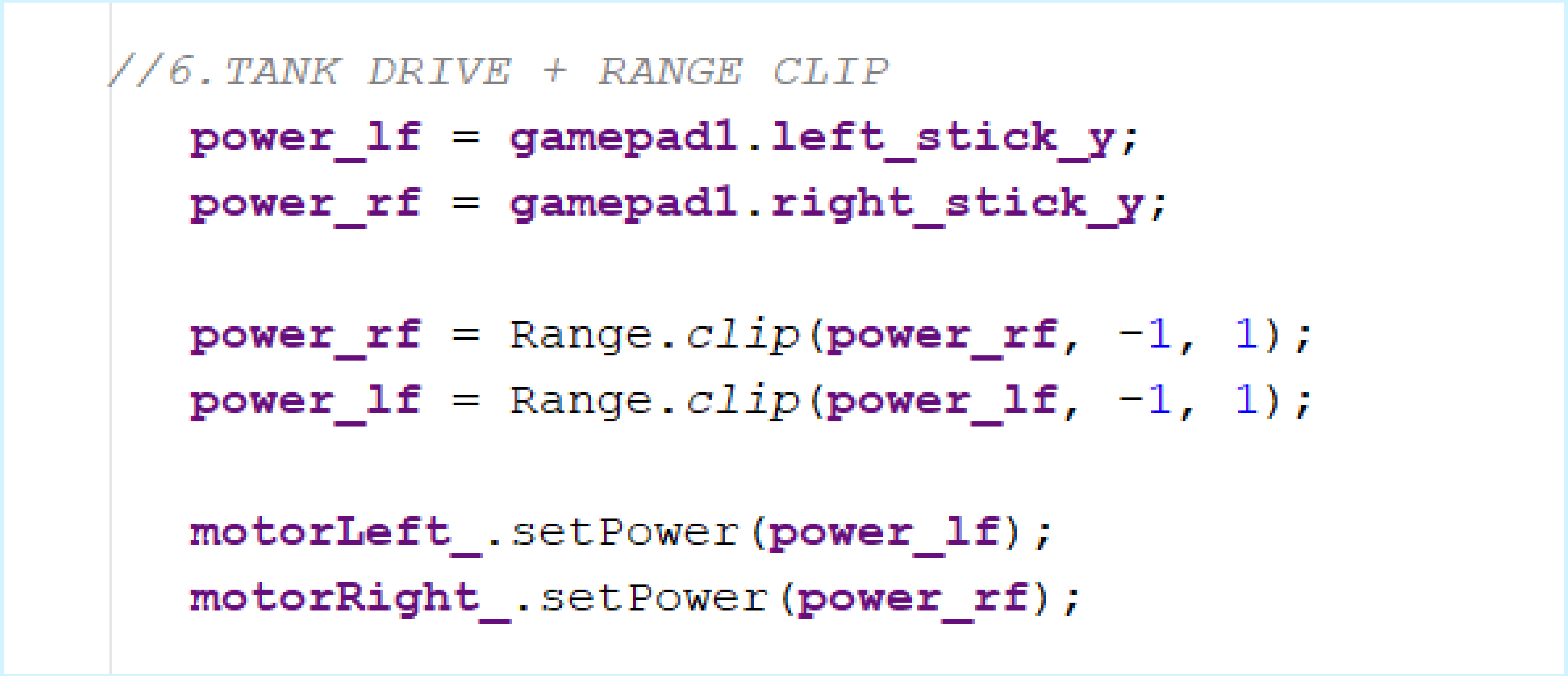
Exercises

- Make it so that robot moves
  - Forward when a is pressed
  - Backwards when left bumper and right bumper are pressed
  - Rotates left when b or y is pressed
  - Rotates right when left_trigger > 0.5

Tank Drive

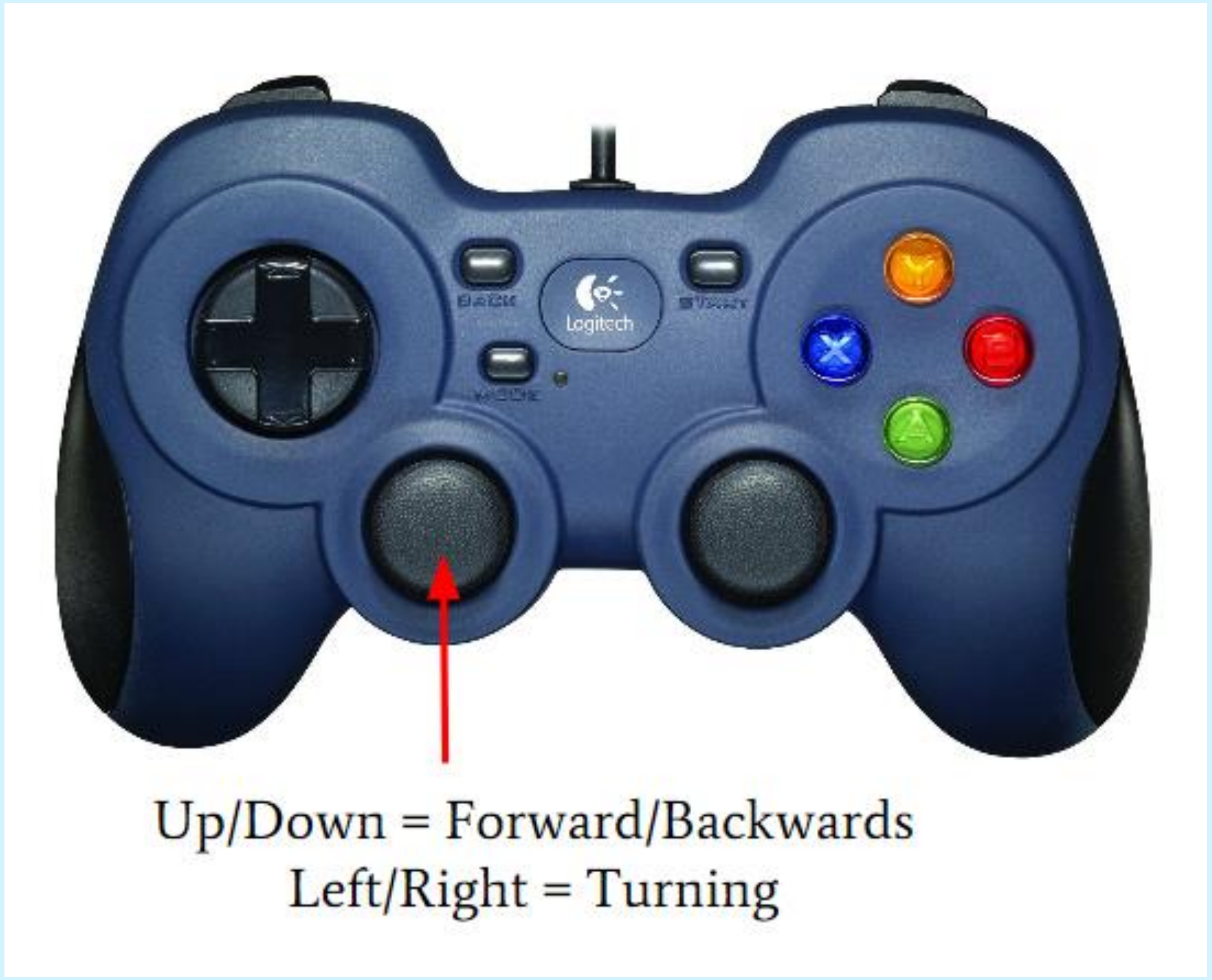


Tank Drive Solution



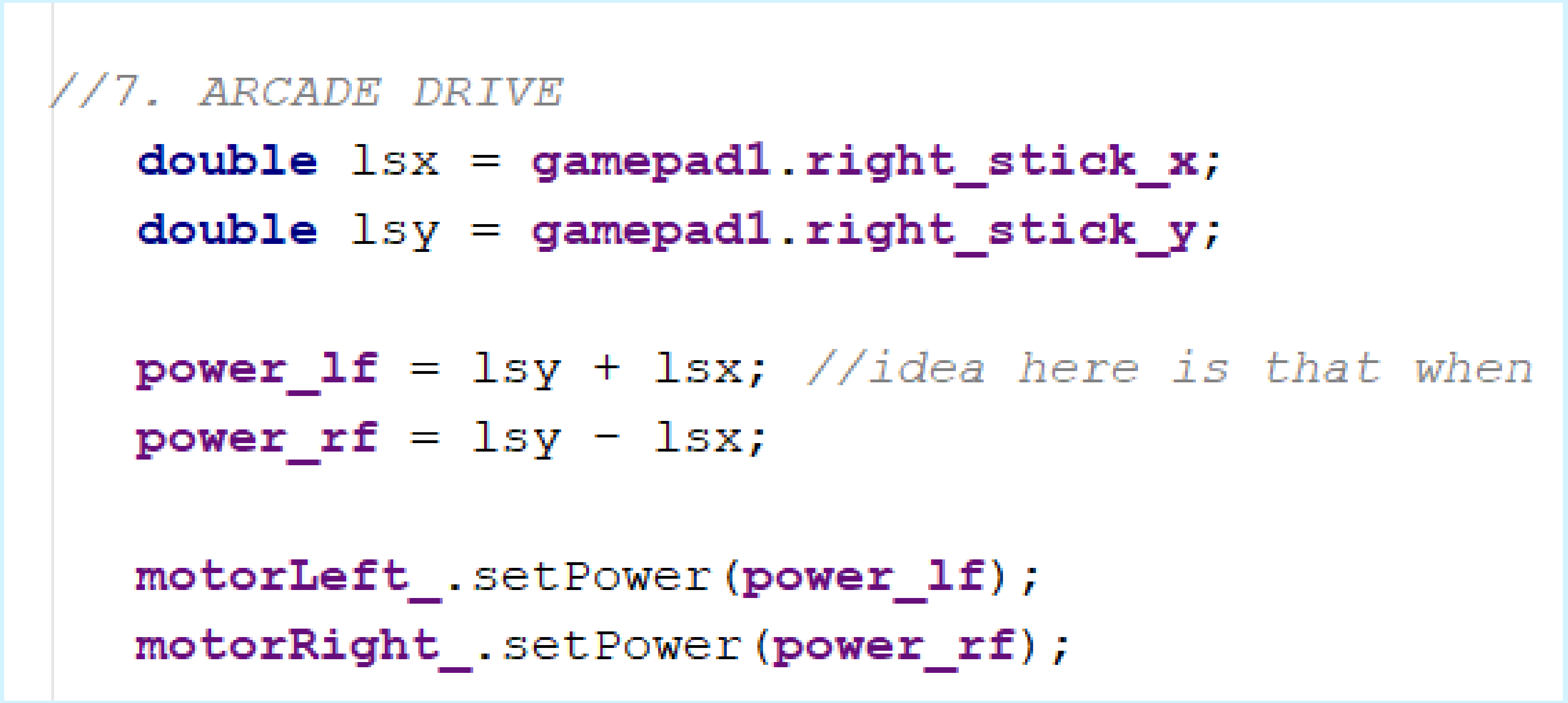
```
//6.TANK DRIVE + RANGE CLIP
   power_lf = gamepad1.left_stick_y;
   power_rf = gamepad1.right_stick_y;

   power_rf = Range.clip(power_rf, -1, 1);
   power_lf = Range.clip(power_lf, -1, 1);

   motorLeft_.setPower(power_lf);
   motorRight_.setPower(power_rf);
```

Arcade Drive



Up/Down = Forward/Backwards
Left/Right = Turning

Arcade Drive Solution

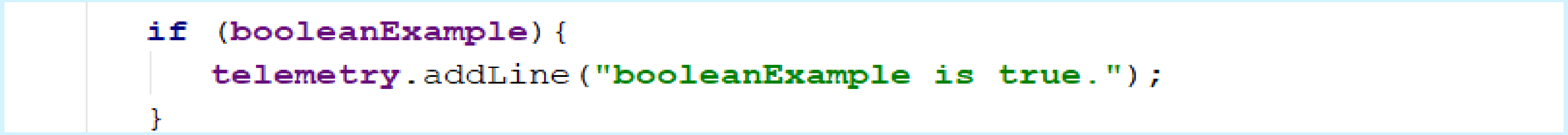


```
//7. ARCADE DRIVE
    double lsx = gamepad1.right_stick_x;
    double lsy = gamepad1.right_stick_y;

    power_lf = lsy + lsx; //idea here is that when
    power_rf = lsy - lsx;

    motorLeft_.setPower(power_lf);
    motorRight_.setPower(power_rf);
```

//idea here is that when turning (ie joystick is full R, lsx = 1 and lsy = 0. that means one motor will be 1, the other -1. opposite signs! exactly what we need!

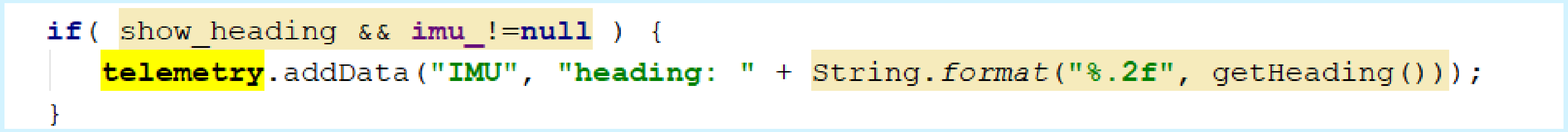
Telemetry

- Display messages on DS
- Couple different ways
  - If string/text: telemetry.addLine(“[text you want displayed]”);



```
if (booleanExample){
    telemetry.addLine("booleanExample is true.");
}
```

  - If you want data: telemetry.addData(“[string]”, String.format([number value]));
  - Strings separated by comma



```
if( show_heading && imu_!=null ) {
    telemetry.addData("IMU", "heading: " + String.format("%.2f", getHeading()));
}
```

Review!

- Kahoot:

FTC Resources

- Game Manual 1
- Youtube
- FIRST and ORTOP Sites
- Other teams!

General Programming Resources

- Scratch
  - https://scratch.mit.edu/
- Code.org
  - Tutorials: https://code.org/learn
  - Make Your own phone apps at: https://code.org/educate/applab
- App Labs:
  - Scratch but with tutorials https://code.org/educate/applab
- Made with Code
  - https://www.madewithcode.com/