FTC Training Manual

Using App Inventor with the Android Controller

Thomas Eng

10/3/2016

This document contains training material to teach students and mentors how to program their FTC robots using a tool known as the App Inventor.

Contents

1 Introduction

The *FIRST* Tech Challenge (FTC) will be adopting a new controller for its robot competitions. This new platform uses Android devices that are powered by QualComm Snapdragon processors. This document will help you learn how to program the new controller using a very powerful, yet easy-to-use tool called the *App Inventor*.

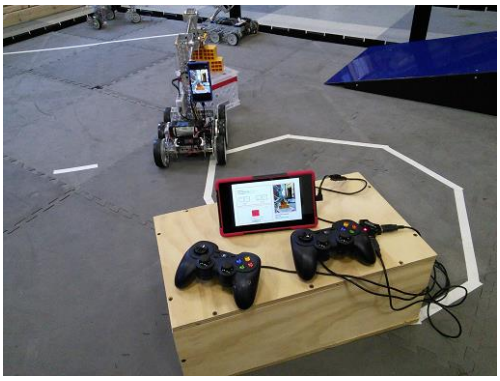


Figure 1 - FTC's new Android-based platform.[1]

2 Overview of the Next Gen Platform

The new FTC controller system has two main components, the *Driver Station* and the *Robot Controller*. The Driver Station is the part of the system that is used by a human driver to select special programs or *op modes* (short for "operational modes") that will run on the Robot Controller. The Robot Controller is the device that is mounted on the robot. The Robot Controller acts as the "brains" of the robot. It runs the op modes that determine how the robot will behave during an FTC competition.

Figure 2 - The new FTC platform consists of a Driver Station and a Robot Controller.

[1] Note that the user interfaces that are pictured on the Android devices are simulated.

If you would like to customize your robot's behavior, then you will need to learn how to write your own op modes. The App Inventor is a visual design tool that can help you quickly create a custom app for your Robot Controller.

3 What is the App Inventor?

The App Inventor is a design tool that makes it easy for you to create your own Android apps using a very user-friendly drag-and-drop, visual interface. In order to customize the behavior for your robot, you will need to use the App Inventor to create a Robot Controller app with one or more op modes that you can use to control your robot.

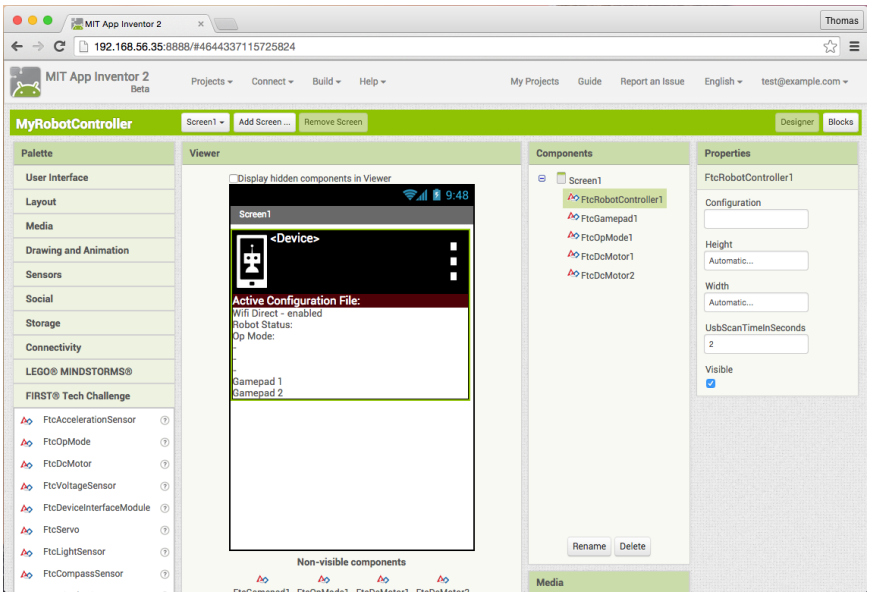


Figure 3 - The App Inventor is a browser-based application that makes it easy to create custom Android apps.

The App Inventor is a browser-based application. This means that you run the application using a web browser such as Google Chrome, Apple Safari, or Mozilla Firefox.[2] The App Inventor is normally available as a web site run by MIT. This means that you typically need to be connected to the Internet in order to use App Inventor.

However, for the *FIRST* Tech Challenge (FTC) competitions teams might not always have access to the Internet at competition venues. Instead of using the Cloud-based version of App Inventor, there is a local version of the App Inventor that you can install onto your personal computer (Windows, Mac or Linux) that will allow you to use the App Inventor, even if your computer is not connected to the Internet. The instructions in this manual assume you are working with the local installation of the App Inventor.

[2] Note that we do not recommend the use of Microsoft's Internet Explorer browser with the App Inventor. Windows users should consider installing Google Chrome which is available for free from Google.

3.1 Important Note Regarding this Training Guide

This training guide assumes that you already have the App Inventor installed locally onto your personal computer and that you are able to build and install apps onto your Android device. For detailed instructions on how to install and use the App Inventor locally, please refer to the following FTC Training Manuals,

- *FTC Training Manual: Running App Inventor Locally for Windows PCs*
- *FTC Training Manual: Running App Inventor Locally for Macs*

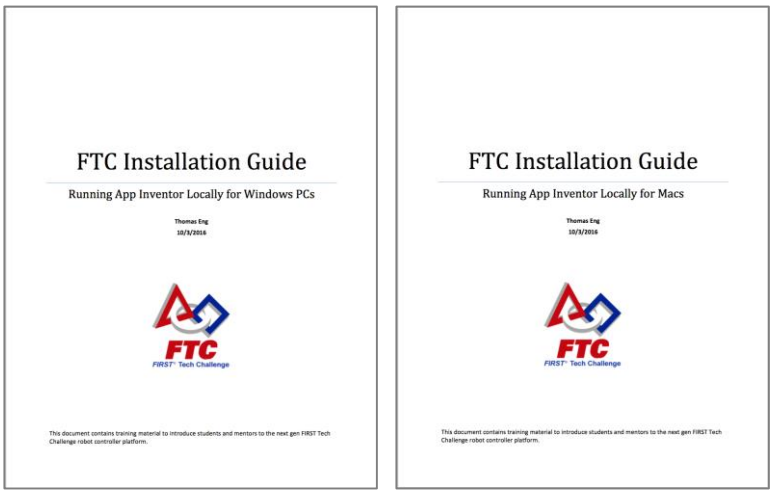


Figure 4 - Refer to either of these manuals to install App Inventor locally onto your computer.

FIRST has created an FTC App Inventor page where you can download these and other manuals. Use the following address to access the FTC App Inventor download page:

https://frc-events.firstinspires.org/ftcimages/2016

4 The FTC Driver Station

Each FTC team will need at least two Android devices to control their robot (see Figure 2). One of the Android devices will be mounted on the robot and be called the *Robot Controller*. The second Android device will be placed next to the team drivers and will be connected to one or two gamepads. This second Android device is referred to as the *Driver Station*.

You are going to use the App Inventor to create the app that will run on the Robot Controller. You will need to download the *FTC Driver Station* app from the Google Play store and install it on the Driver Station device.

4.1 Installing the FTC Driver Station App

The FTC Driver Station app is located on Google Play. To install the app, you first need to connect the ZTE phone to an available wireless network that has access to the Internet. Launch the **Settings** activity on your phone, and select the **Wi-Fi** item to display the Wi-Fi activity.

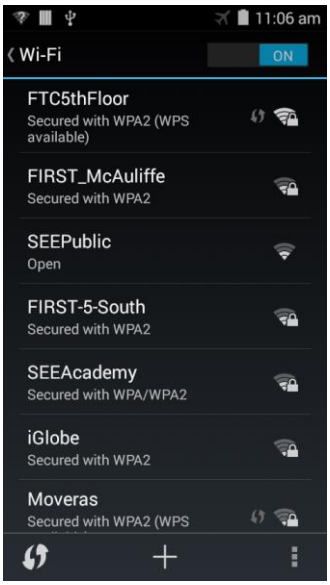


Figure 5 - Select your wireless network from the Wi-Fi activity.

Select your desired wireless network from the Wi-Fi activity and provide the password information required to access the wireless network.

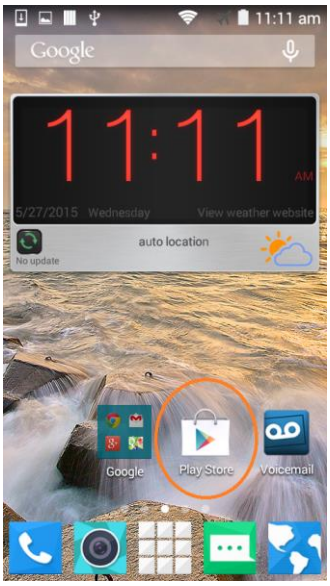


Figure 6 - Launch the Google Play Store app.

Once you have connected successfully to your wireless network, launch the Google **Play Store** app from your phone. The Play Store app might prompt you to either login to an existing Google account or create a new one. Follow the onscreen instructions to either create a new (free) account or login to your existing account.

Once you have successfully logged in to Google Play, click on the search icon (a little magnifying glass) and search for the phrase “FTC Driver Station”

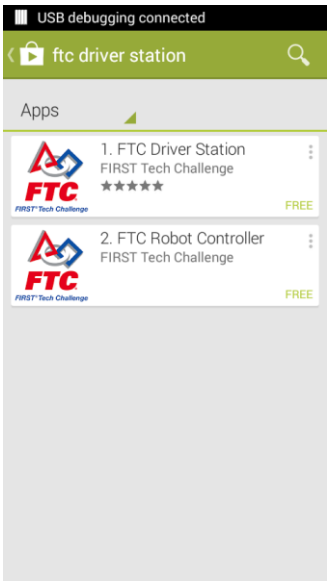


Figure 7 - Search for the phrase "FTC Driver Station"

Once you have found the FTC Driver Station app, click on it and follow the onscreen instructions to install. Note that the application might prompt you to enter a credit card number or some other method of payment. The app is free and no payment method is required. You should be able to hit the "Skip" button to skip the process of providing a method of payment.

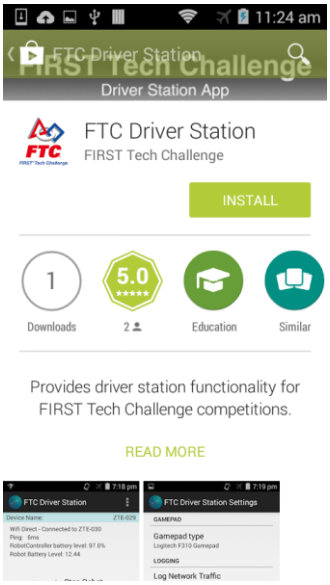


Figure 8 - Follow the on-screen instructions to install the app.

If you were able to install the app successfully, then there should be an FTC Driver station icon on your Android device.

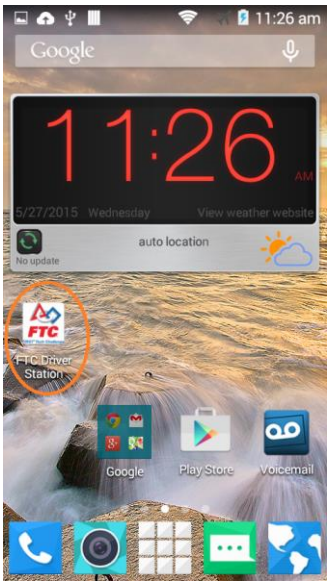


Figure 9 - After installing the app, you should see an icon on your Android's screen.

IMPORTANT NOTE: After you have successfully installed the app, go to your Wi-Fi settings menu and "forget" the wireless network that you used to connect to Google Play. In general, you do not want to be connected to any wireless networks with the exception of your FTC Robot Controller device.

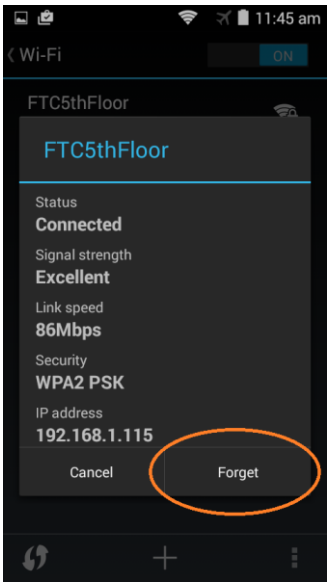


Figure 10 - Don't forget to forget the wireless network after the installation is complete!

IMPORTANT NOTE: You do not want to install the FTC Driver Station on the same Android device as the FTC Robot Controller app. These two apps should be installed on separate devices. One of the apps (the Robot Controller) will configure the device to operate as a WiFi Direct Group Owner. It is important that the two apps are installed and run on separate devices.

4.2 Launching the FTC Driver Station

To launch the FTC Driver Station app, simply touch its icon on the screen (see Figure 9). The FTC Driver Station app will launch into its main screen (see Figure 11). Towards the top of the screen, you can see some information about the status of the wireless connection. Right now in Figure 11 the driver station is not connected to a robot. If the driver station were connected to a robot, you would see ping statistics (i.e., it would show the time it takes for a message to get sent to the robot and get acknowledged by the robot) as well as other status and wireless connection info.

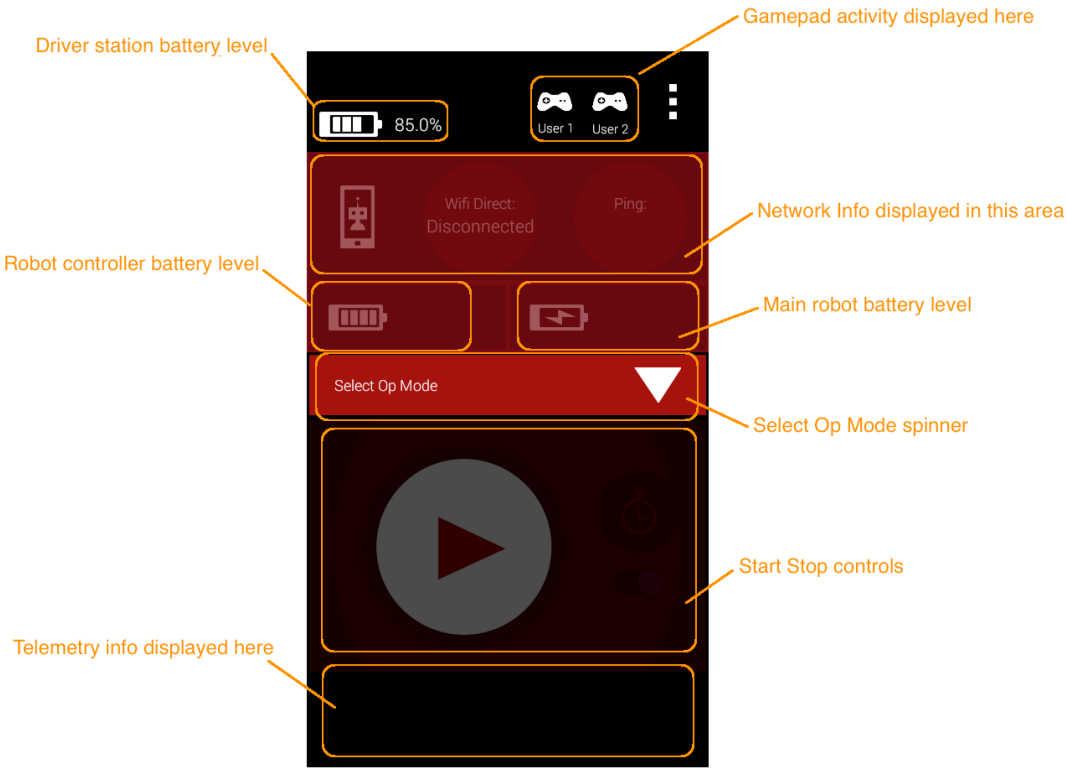


Figure 11 - Main screen for FTC Driver Station.

Towards the middle of the screen, there is a "spinner" that you can use to select the *operational mode* or *op mode* of the robot. Op modes are pre-programmed robot behaviors (autonomous and driver controlled) that you can launch from the driver station. Beneath the Select Op Mode spinner, there are start/stop controls that let you start and stop the selected op mode. In Figure 11 the start/stop controls are disabled out because the driver station has not yet connected to a robot.

Below the Op Mode controls is an area on the screen where telemetry data is displayed. The new platform has the ability to send information from the robot to the driver station. This information (such as servo position, motor power, sensor data, etc.) can be displayed on the driver station.

Towards the upper right top of the screen, there is an area that is used to display information about the gamepads that are connected to the Android device. The gamepad icons will be highlighted in green whenever the app detects gamepad activity for a user.

4.2.1 Driver Station Menu

Towards the upper right corner of the main screen, there is a set of three vertical dots. Touch these dots to display a pop-up menu of FTC Driver Station selections.

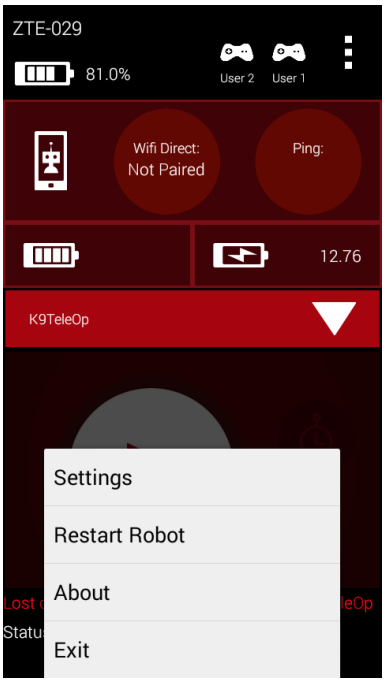


Figure 12 - Click on the dots in the upper right hand corner to display menu options.

4.2.1.1 Settings Screen

If you select the **Settings** the FTC Driver Station Settings screen will appear. This is the screen that you can use to modify the settings for your device.

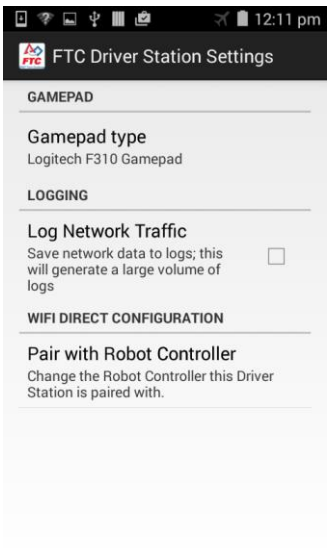


Figure 13 - The Settings screen.

Through the FTC Driver Station Settings menu, you can select which type of gamepad is connected to the device. Note that we recommend the use of the Logitech F310 gamepad controller with the device set in Xbox controller emulation mode (i.e., with the switch on the bottom set to "X").

Before you use your gamepad, make sure that the Logitech F310 is in Xbox emulation mode. This means that the little switch on the bottom of the controller should be set to the "X" position.

Figure 14 - Make sure the switch is in the "X" position.

The Settings menu also lets you select the option to "Log Network Traffic". This option can be useful if you are debugging your wireless connection (it logs information about packets sent to and from the robot controller), but this option can use up all of the storage on your Android device very quickly. We do not recommend that you use this option and you should never have this feature enabled during a match.

The Settings menu also has an option to pair the driver station to the robot controller. You need to connect or pair your driver station to a specific robot controller. We will review this pairing process in greater detail later on in this document.

4.2.1.2 Restart Robot

The FTC Driver Station menu has an option called **Restart Robot.** This option will do a remote soft restart of the robot controller. Sometime the robot controller might enter a stop mode. In the event that this happens, the driver can issue a remote restart command to the robot controller with this option. The Driver Station needs to be connected to a Robot Controller in order to restart the robot.

4.2.1.3 About Menu

Selecting the **About** menu option will bring up a screen which displays info about the app version and the build version of your FTC Driver Station app.

4.2.2 Using the Gamepads

As it was mentioned in a previous section, if you are using the Logitech F310 gamepad with your driver station, then make sure the switch on the bottom of the pad is set to the "X" position. The driver station app is designed to work with up to two gamepads.

When you connect your gamepads to the Android device (through a USB hub), you need to tell the FTC Driver Station app which gamepad will be used to represent driver #1 (User 1) and which gamepad will be used to represent driver #2 (User 2).

You can select which driver a gamepad will represent by pushing the **Start** key on the gamepad while simultaneously pressing the **A** button if you want to be driver #1, or the **B button** if you want to be driver #2.

Figure 15 - Press Start + A to be driver 1 or Start + B to be driver 2.

When you first connect your gamepads to the Android device, you must push a button combination to designate which driver your gamepad will represent.

Once you have designate which driver your gamepad will represent the gamepad status information will appear below displayed on the gamepad icons in the upper right hand side of the screen.

For example, if the gamepad that represents driver #1 is active, then the User 1 gamepad icon (in upper right hand corner of main driver station screen) will be highlighted green whenever the app detects user activity on User 1's gamepad.

If the gamepad that represents driver #2 is active, then the User 2 gamepad icon will be highlighted green whenever the app detects activity on User 2's gamepad.

Remember, before you can use a gamepad, you need to tell the Driver Station which driver or user that gamepad represents.

4.2.3 Op Mode Controls

The center of the FTC Driver Station main screen contains controls that can be used to select an operation mode or op mode of a robot. Op modes are preprogrammed robot behaviors that you can select and execute from the driver station. The **Select Op Mode** spinner will display a list of available op

modes that can run on your robot. The **Start** button is used to start the selected op mode. When an op mode is running, the Start button will toggle to a **Stop** button. The **Stop** button stops the current op mode.

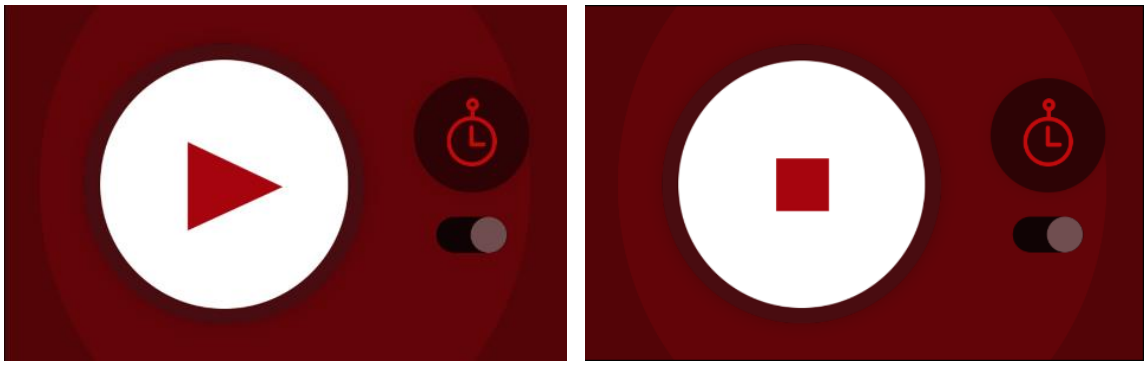


Figure 16 - The Start button toggles to a Stop button when an Op Mode is running.

The buttons will not be enabled until the Driver Station is connected wirelessly to a robot controller. We will revisit the Op Mode controls in greater detail at a later section of this manual.

5 Building a Simple Robot Controller

5.1 Create a New Project

Let's build a simple robot controller app that has only one, very simple Op Mode. Make sure that your App Inventor Local appliance is running in VirtualBox and that the status of the App Inventor and Build servers are both "OK".[3]

Launch your web browser[4] and type in the following address,

http://192.168.56.35:8888

This is the address and port number of the App Inventor Local virtual server that should be running on your machine.

[3] Refer to the *Running App Inventor Locally* manual (Windows or Mac) for detailed instructions on how to start the App Inventor Local appliance.

[4] Windows users should not use the Internet Explorer browser to access the App Inventor. Instead, they should use Google Chrome, which is available for free from Google.

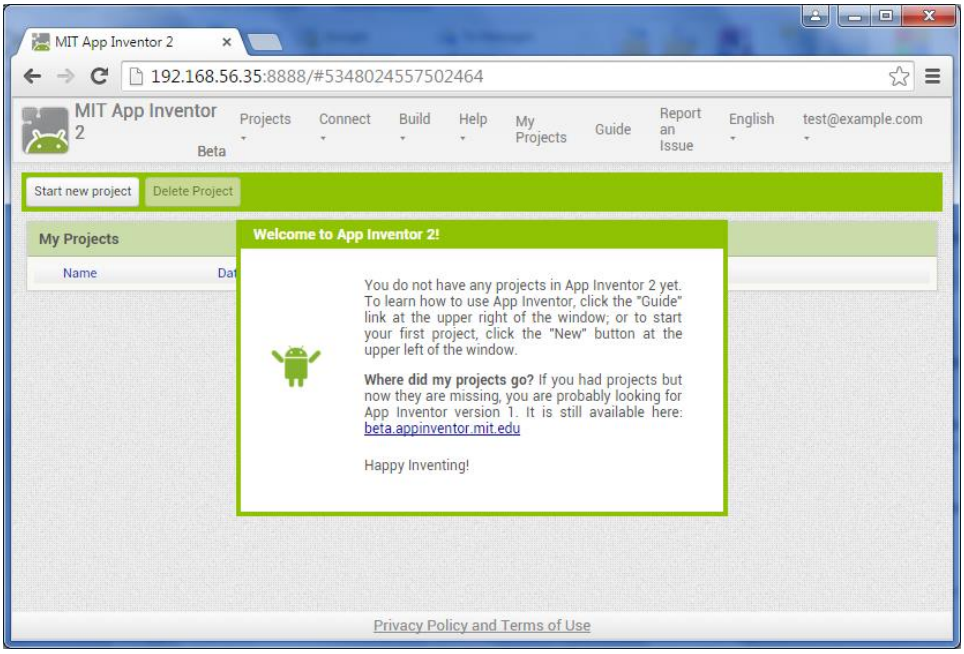


Figure 17 - Launch your browser and enter in the address http://192.168.56.35:8888

Near the top of the App Inventor screen click on the **Start new project** button to start a new App Inventor project.

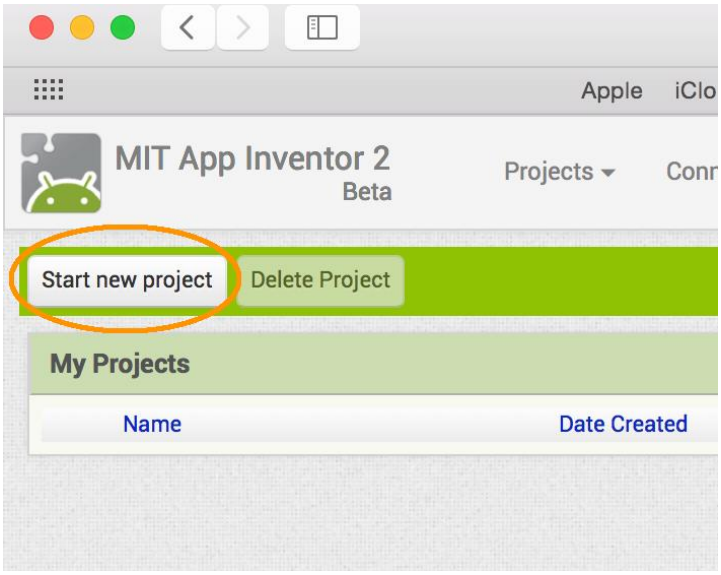


Figure 18 - Click on the Start new project button to create a new project.

Specify a name for your project and hit **OK** to create the new project.

Figure 19 - Specify a name for your new project.

Let's change the title of app's main screen. By default its title is "Screen1" but let's change it to something more descriptive. Verify that the Screen1 component is selected in the **Components** pane and in the **Properties** pane change the property Title from "Screen1" to "FTC Robot Controller".

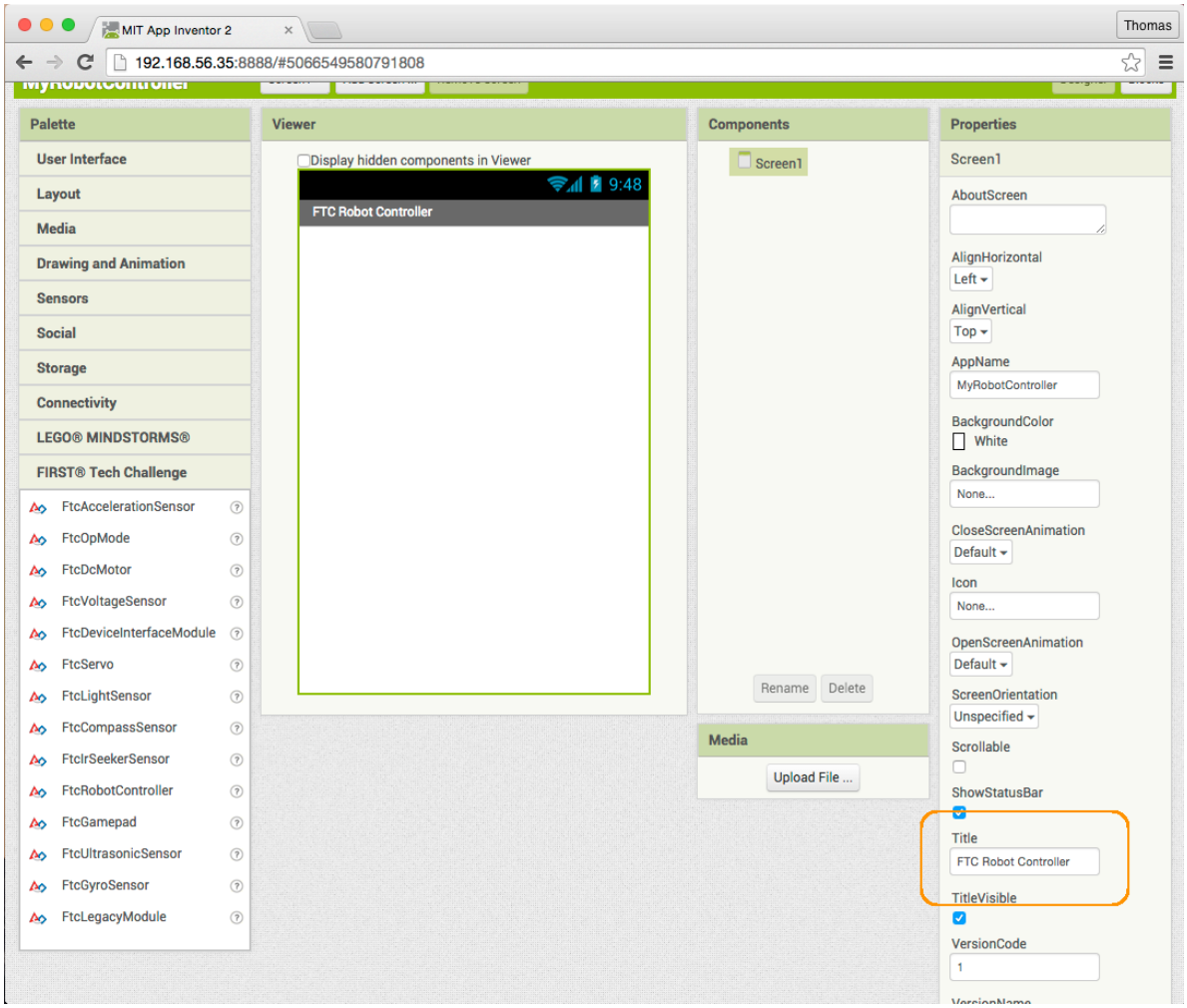


Figure 20 - Change Screen1's title to "FTC Robot Controller"

5.2 FTC Component Palette

If you look at your App Inventor work area, on the upper left portion of the screen you should see a special palette of *FIRST* Tech Challenge related design components. If it's not already expanded, click on the words **"FIRST Tech Challenge"** to expand the palette and see the FTC-related design components.

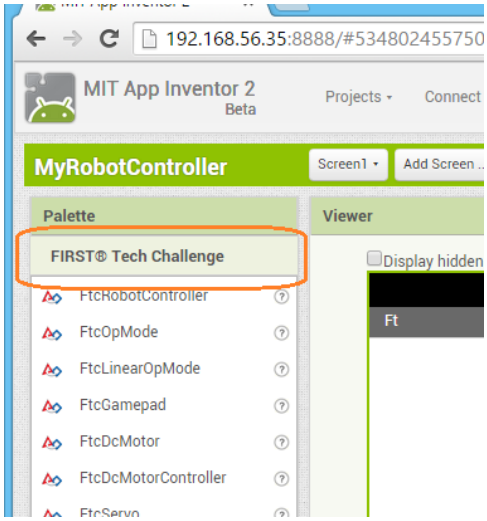


Figure 21 - Click on the words "FIRST Tech Challenge" to expand the palette and see the FTC-related components.

The *FIRST* Tech Challenge palette contains several design components that you can use to build your own Robot Controller app and to create one or more custom op modes.

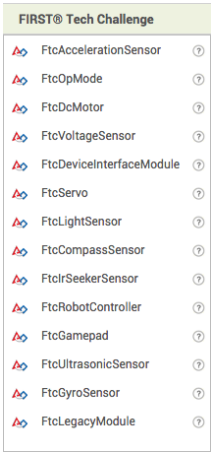


Figure 22 - The *FIRST* Tech Challenge palette contains the components you will use to make a robot controller app.

5.3 FtcRobotController Design Component

For our simple robot controller app, we are going to start by adding an **FtcRobotController** design component from the palette to the empty screen in the Viewer pane. The FtcRobotController design component is a visible design component (i.e., it will not be hidden from the user during run time). The FtcRobotController component provides your app with the same framework that is available with the

Google Play version of the FtcRobotController. The FtcRobotController will establish communication with the Driver Station and it will also load the robot configuration information from a configuration file.

Drag an FtcRobotController component from the *FIRST* Tech Challenge palette and place the component onto Screen1 in the Viewer pane.

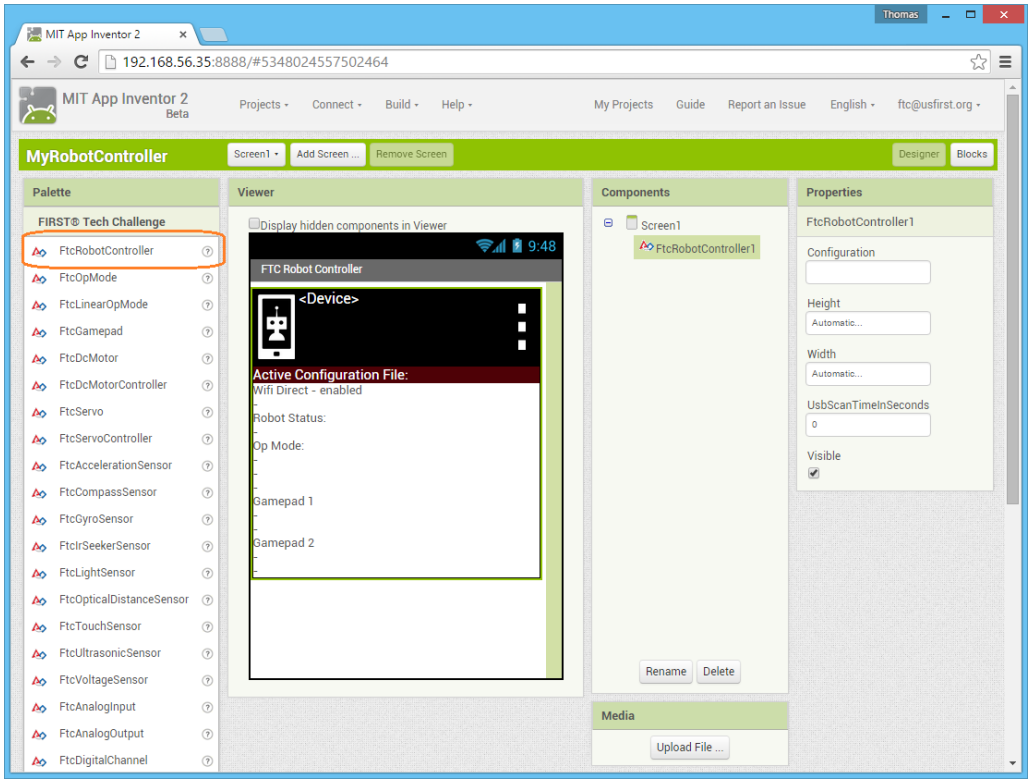


Figure 23 - Drag the FtcRobotController component from the palette and place it on the screen in the Viewer pane.

5.3.1 Specifying the Name of the Configuration File

Now you need to specify the name that you would like to use for the configuration file of your robot. Select the FtcRobotController1 component from the **Components** pane of your browser's window. In the **Properties** pane of your browser, locate the Configuration property and type in the name that you would like to use for your robot's configuration file.

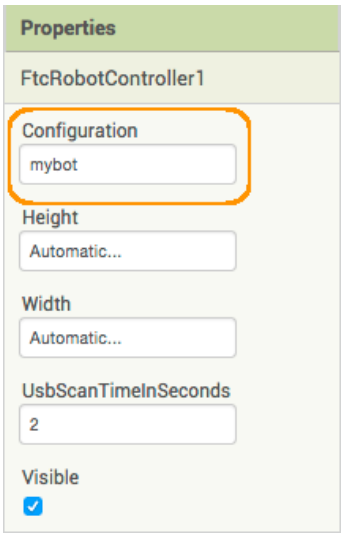


Figure 24 - Specify the name of the configuration file for your robot.

For this example let's use "mybot" as the configuration file name. This means that when you run your Robot Controller app, the FtcRobotController1 component is going to look for a configuration file on your Android device with the name of "mybot". Note that the file name is case sensitive so the name that you specify in the Configuration property will need to match your configuration filename *exactly*.

5.4 Creating an Op Mode

5.4.1 Adding the Op Mode Component

Now that we have our FtcRobotController component in place, let's add an op mode to our app. From the *FIRST* Tech Challenge palette, drag an FtcOpMode component and place it onto your app's main screen in the **Viewer** pane.

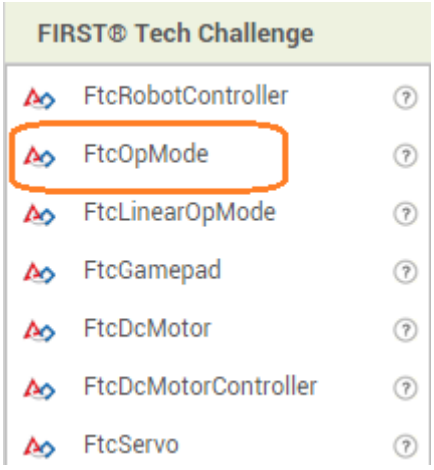


Figure 25 - Drag an FtcOpMode component onto the app's screen.

Note that the FtcOpMode is a hidden component so by default it will not be visible on the actual app screen. It should be listed as a non-visible component towards the bottom of the **Viewer** pane.

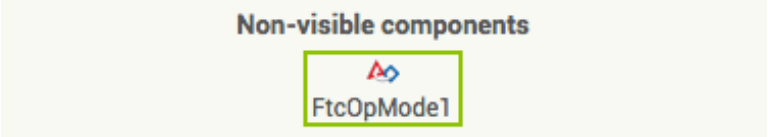


Figure 26 - The FtcOpMode component is a hidden component.

Let's rename the op mode component that you just added. In the **Components** pane, select the component with the name "FtcOpMode1" and click on the **Rename** button. Specify a new name of "opElapsed" in the Rename Component window and hit **OK** to rename the component.

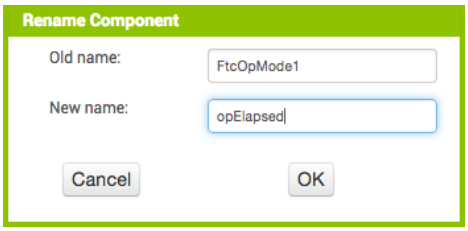


Figure 27 - Specify a new name and hit OK.

Now let's specify the name of the op mode (which will appear in the list of available op modes on the Driver Station). In the **Properties** pane for the opElapsed component, change the OpModeName property from "Unnamed Op Mode" to "Elapsed Op Mode".

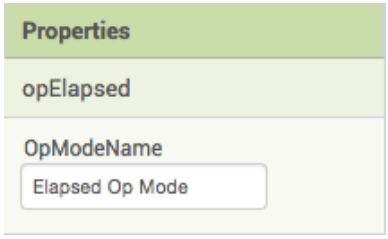


Figure 28 - Change OpModeName to "Elapsed Op Mode".

5.4.2 Switching to Blocks Mode

We are now ready to program our app and op mode. In the App Inventor window, click on the **Blocks** button (near the upper right hand corner of the window) to switch to the block programming mode.

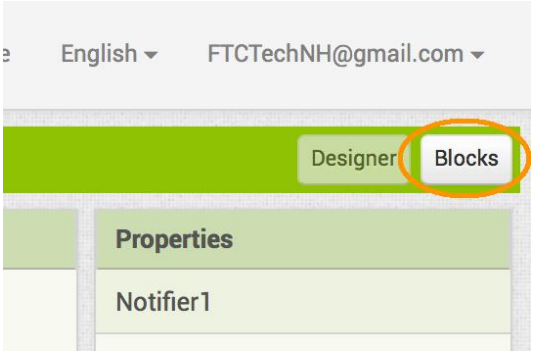


Figure 29 - Click on Blocks button to switch to block programming mode.

After you have switched to block programming mode, look in the **Blocks** pane (on the left hand side of the window) and click the **opElapsed** component from the list of available design components. When you click on this component, a set of programming blocks should appear in the **Viewer** pane of your App Inventor window (see Figure 30).

Figure 30 - Clicking on opElapsed reveals a set of programming blocks that are associated with this component.

5.4.3 A Closer Look at the Op Mode Programming Blocks

Let's take a closer look at the op mode programming blocks. The first few programming blocks (the gold-colored ones that begin with the word "when") are *event handlers*. You can use these blocks to specify what the app should do when one of these events are triggered:

- Init – This event gets triggered when the op mode is first initialized. Currently, this is when the driver pushes the Start button on the Driver Station. Eventually, however, this will occur when the driver pushes the Init button (before he/she presses the Start button) on the Driver Station.
- Start – This event gets triggered when the op mode is first executed (i.e., when the driver pushes the Start button on the Driver Station). You can use this event to do tasks that only need to be performed at the start of the op mode.
- Loop – This event gets triggered regularly (approximately every 30 milliseconds). This is where most of your programming logic will be placed.

- Stop – This event gets triggered when the op mode is stopped (i.e., when the driver pushes the Stop button on the Driver Station). You can use this event to clean up after your op mode has completed its run.

The purple programming block represents a method (also known as a procedure) that you can call for the op mode object:

- GetRuntime – This is a "getter" method that returns the current runtime (in seconds) that this op mode has been running.

The final block in Figure 30 represents the opElapsed op mode object. This block can be used for more advanced App Inventor programs. We will not need this programming block for our example op mode.

Let's look at the programming blocks for the FtcRobotController1 object. Click on the FtcRobotController1 object in the **Blocks** pane and then scroll down till you see the last two purple methods for that object.

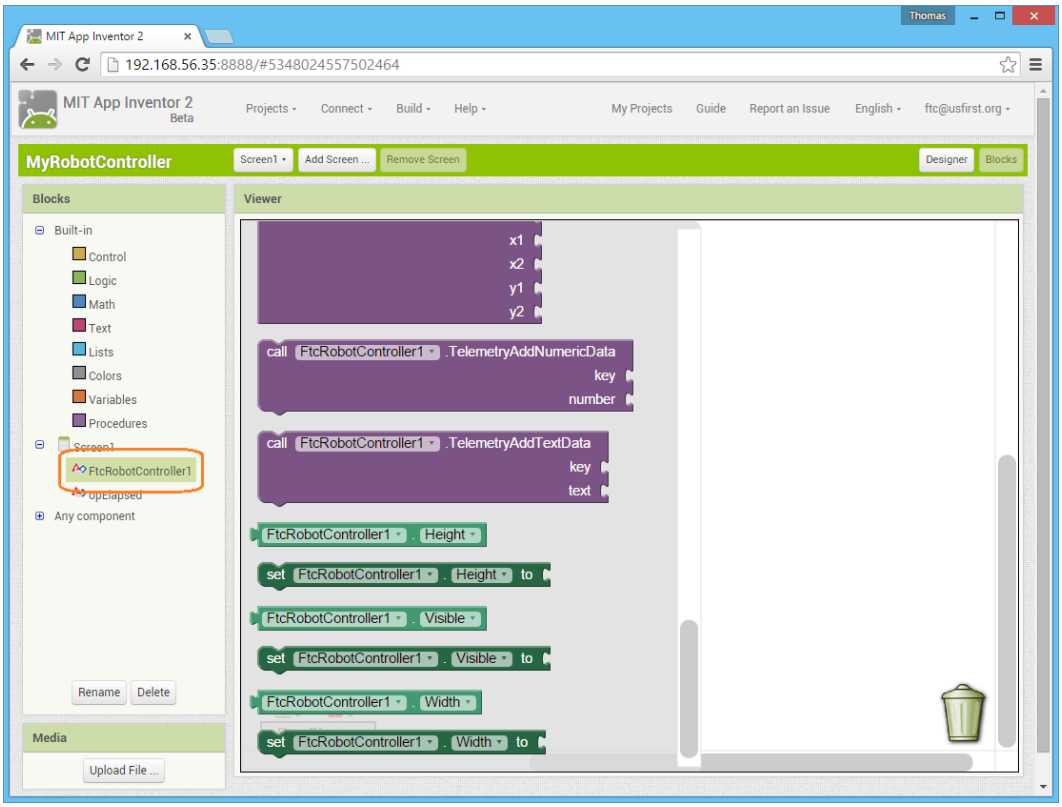


Figure 31 - Click on FtcRobotController1 and scroll down towards the last two (purple) methods for that object.

The purple blocks are methods that are associated with the FtcRobotController1 object:

- TelemetryAddNumericData – This function lets you send numeric data from the Robot Controller to the Driver Station. You need to specify a key (that will be used as a reference for the data) and the numeric value that you want to send to the driver station.

- TelemetryAddTextData – This function lets you send a text string from the Robot Controller to the Driver Station. You need to specify a key (that will be used as a reference for the data) and the text string that you want to send to the driver station.

5.4.4 Creating the Op Mode Logic

Before we start writing the program for our op mode, we should first define what we want the op mode to do. For this example, let's create an op mode that will send back (using the telemetry function) the current run time whenever the Loop event gets triggered.

Select the Loop event handler and place the block onto the **Viewer** pane of your browser window.

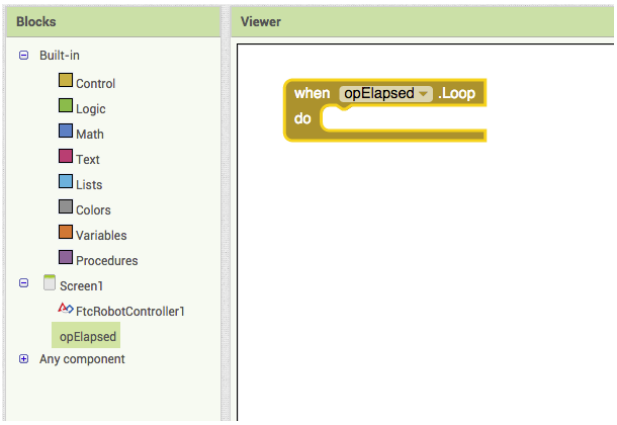


Figure 32 - Drag and drop the Loop event handler block onto the Viewer pane.

Click on the FtcRobotController1 component in the **Blocks** pane to display the programming blocks again. Select the **call FtcRobotController1.TelemetryAddNumericData** block and place it inside the Loop event handler block. Note that the interior shape of the event handler should match the exterior shape of the of the procedure block.

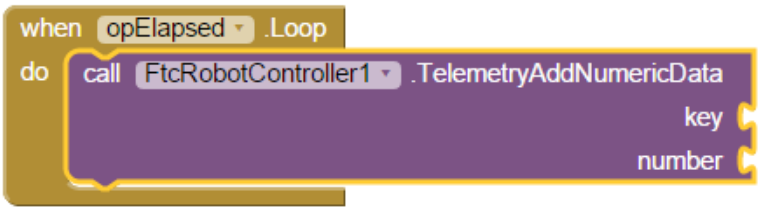


Figure 33 - Drag the TelemetryAddNumericData block onto the Loop event handler block.

Now we have to specify a key for the value – this key will be used as a reference for this piece of data. Click on the Text category within the **Blocks** pane.

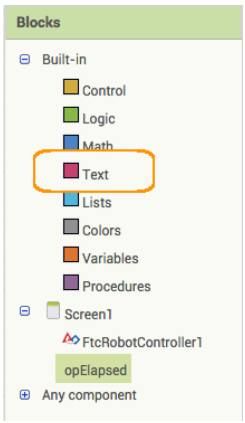


Figure 34 - Click on the Text category in the Blocks pane.

Select the text string programming block and drag it to the **key** notch on the **TelemetryAddNumericaData** block.

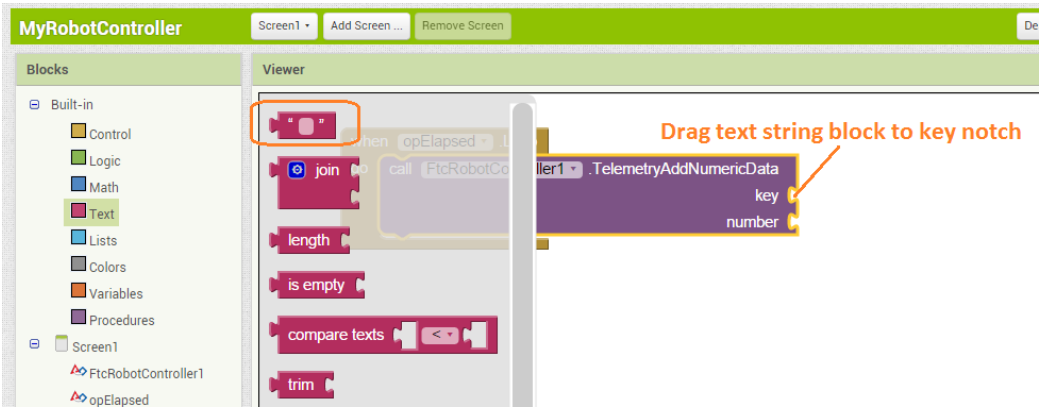


Figure 35 - Drag the text string block and place it on the key notch of the TelemetryAddNumericData block.

Click on the text block and specify a name of “rtime” for the key.

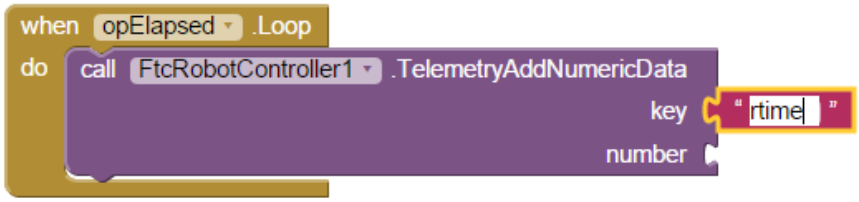


Figure 36 – Edit the text string block and specify “rtime” as the name for the key.

The last thing we need to do is get the current run time so we can send it back to the driver station using the telemetry function. Click on the opElapsed component in the **Blocks** pane of the App Inventor window and select the **call opElapsed.GetRuntime** block. Place the block in the **number** notch of the telemetry programming block.

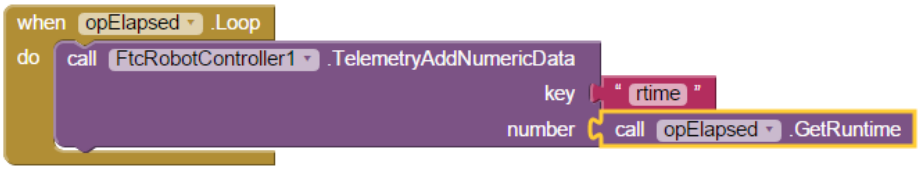


Figure 37 - Drag and drop the GetRuntime block to the number notch on the telemetry block.

That's it! Your program is complete. Let's review the logic. Each time the Loop event for opElapsed is triggered, the app will use the TelemetryAddNumericData procedure to send the current run time (with a key named "rtime") to the Driver Station. The Driver Station will receive this telemetry data and display it on the screen so the driver can see the current run time.

5.4.5 Important Note Regarding the Robot Controller App

When you install your Robot Controller app that you created with the App Inventor, it is *very* important that this Robot Controller app be the only Robot Controller app on your Android device. In general, you do ***not*** want to have more than one Robot Controller app per robot Android device. If you create a Robot Controller app with the App Inventor, it should be the only Robot Controller app on your smartphone. It should also be the default app for any of the USB hardware modules that are used on your robot (such as the Core Legacy Module, the Core Motor Controller, the Core Servo Controller and the Core Device Interface Module). If you have any other Robot Controller apps on your Android device, you should uninstall them and leave only one Robot Controller app on the device.

To uninstall an app, launch the **Settings** app of your Android Device. Click on the **Apps** option in the **Settings** activity.

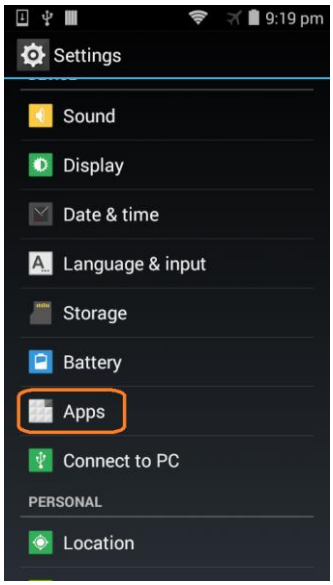


Figure 38 - Select Apps from the Settings activity.

From the **Apps** activity, click on the app that you would like to uninstall,

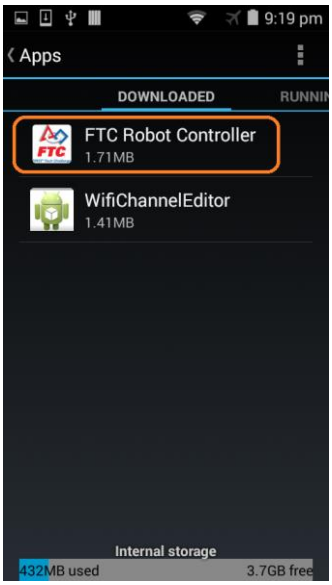


Figure 39 - Click on the app that you would like to uninstall.

Click on the **Uninstall** button to uninstall the app.

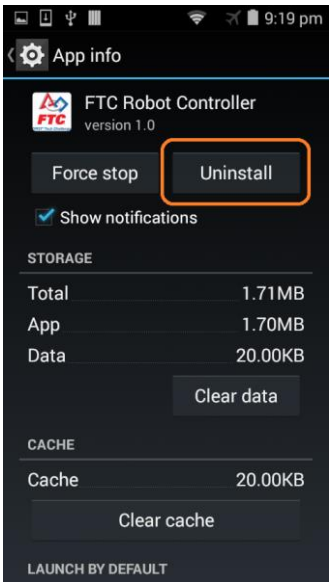


Figure 40 - Click on Uninstall to remove the app.

5.4.6 Building and Installing Your App

Your app is ready to be built. For detailed instructions on how to build and install your app (using a USB cable) please consult the *Running App Inventor Locally* training manuals. **Select Build->App (save .apk to my computer)** menu option to build the app inventor and to download the .apk file from the virtual server. Once the .apk file has been built and downloaded, use a USB cable to transfer the file to your Android device. Install the app onto your Android device using the File Manager app on the Android device.

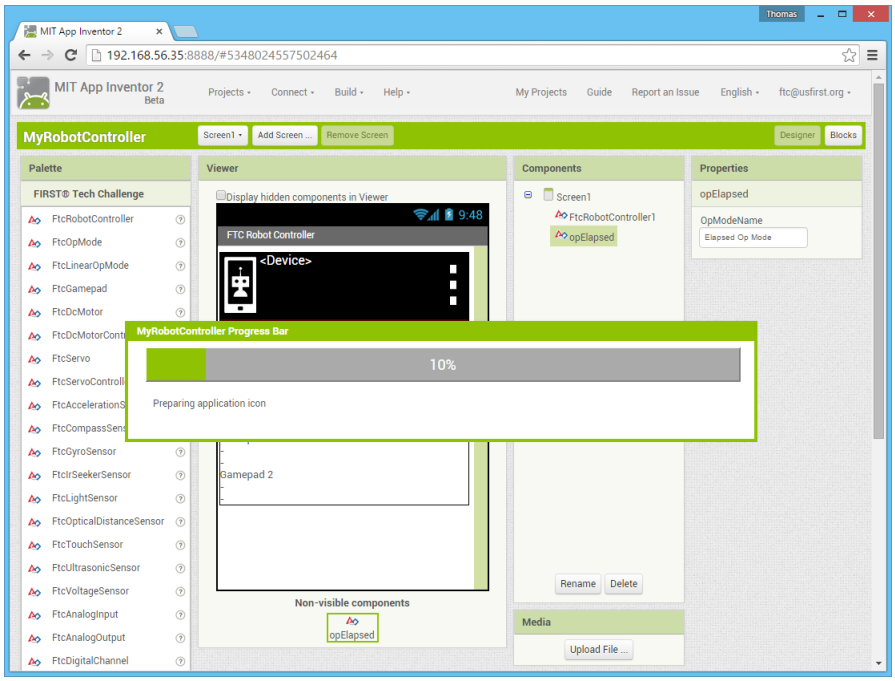


Figure 41 - Build the .apk file then transfer the file to your Android device using a USB cable.

Once you have successfully transferred the app to the Android device and installed it onto the device, there should be a new app icon called **MyRobotController** listed under the Apps screen of your device. Click on the icon to launch the app.

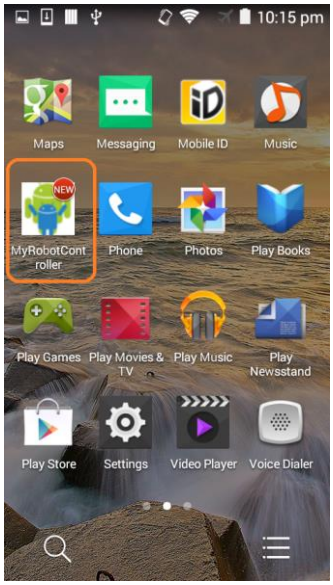


Figure 42 - Click on MyRobotController icon to launch the app.

Congratulations! You've created your first Robot Controller app. When you first launch the app, you will see a notification indicating that the Robot Controller app could not open the robot configuration file called "mybot.xml". You will need to create this configuration file for your Robot Controller.

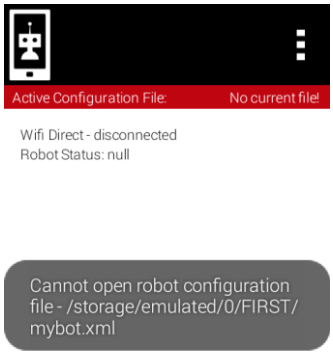


Figure 43 -

5.5 Creating a Simple Configuration File

The purpose of the configuration file is to tell the Robot Controller app what hardware devices are connected to the Robot Controller through the device's USB port. For this example, we are going to create a very simple configuration file. Our configuration will be empty (i.e., it won't have any hardware devices configured). Don't worry, we will eventually learn how to create a proper configuration file (with real hardware devices included in the file). However, for this very simple example, we will create an "empty" configuration file. This will allow us to run our op mode without any robot hardware connected. All we will need is our two Android phones (one with the FTC Driver Station app, the other with our newly created Robot Controller app).

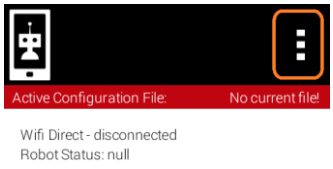
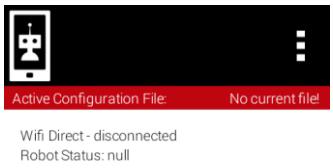


Figure 44 - Touch the three dots in the upper right hand corner to launch pop up menu.

To create a new configuration file, touch the three dots in the upper right hand corner of the main screen on your Robot Controller app. This will launch a pop up menu.



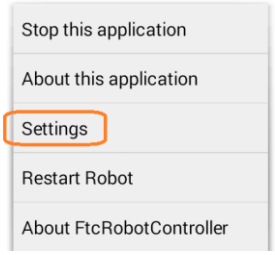


Figure 45 - Select Settings to launch the Settings menu.

Click on the **Settings** item to launch the Settings menu.

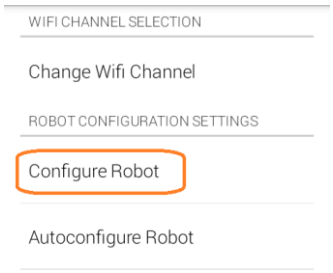


Figure 46 - From the Settings menu, click on Configure Robot.

Click on the Configure Robot item to launch the Configure Robot screen.

Figure 47 - Click on the New button to create a new configuration file.

Click on the **New** button to create a new configuration file.

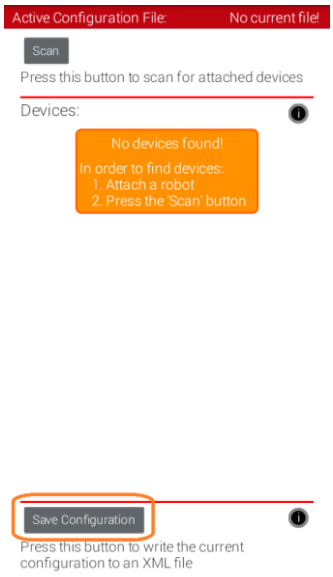


Figure 48 - Since we are creating an empty file, click on Save Configuration to save the empty file.

Since for this example we are creating an empty configuration file, simply click on the **Save Configuration** button to save the empty configuration file.

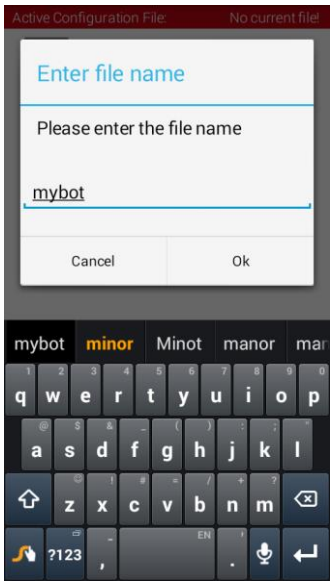


Figure 49 - When prompted enter "mybot" as the file name.

When prompted, enter “mybot” as the file name. Note that the name specified for the configuration file ***must*** match the name specified in the Configuration property for the FtcRobotController1 component that we used to build our app (see Figure 24). Click on the **OK** button to save the file.

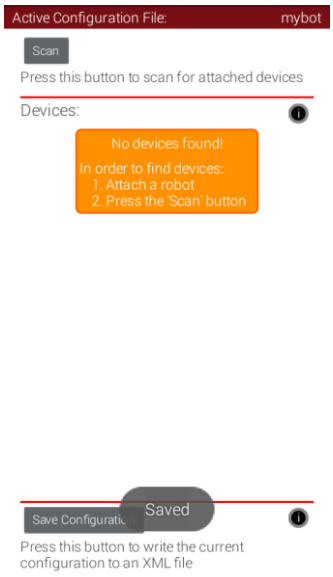


Figure 50 - You should see a "Saved" notification appear.

You should see a “Saved” notification appear if the file was successfully saved. Use your Android device’s back arrow button to return to the previous screen. Keep pressing the back arrow button until you are back at the main Robot Controller screen. You should see in the upper right hand corner that the configuration file “mybot” is now the active file.

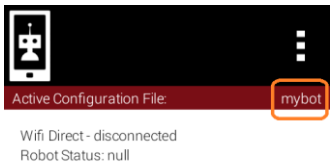


Figure 51 - The file "mybot" should now be the active configuration file.

5.6 Pairing the Driver Station to the Robot Controller

Now that we have a configuration file (even if it is an empty one) we are ready to pair the Driver Station to the Robot Controller. Before you can control your robot from your driver station, you must first "pair" the driver station to the robot controller.

The new control system uses a technology known as WiFi Direct to establish a unique and persistent connection between the driver station and the robot controller. For this system, the robot controller acts as the WiFi Direct *group owner*. Once a driver station has been paired to a robot controller, then the driver station will always look for that robot, until it is paired with a different robot controller.

In order to pair your two devices, you need to make sure that both Android devices are powered on and the FTC Robot Controller app and the FTC Driver Station app are running on their respective machines. Your Driver Station should display its WiFi direct status. In the screen shots below the driver station and the robot controller are not yet paired together.

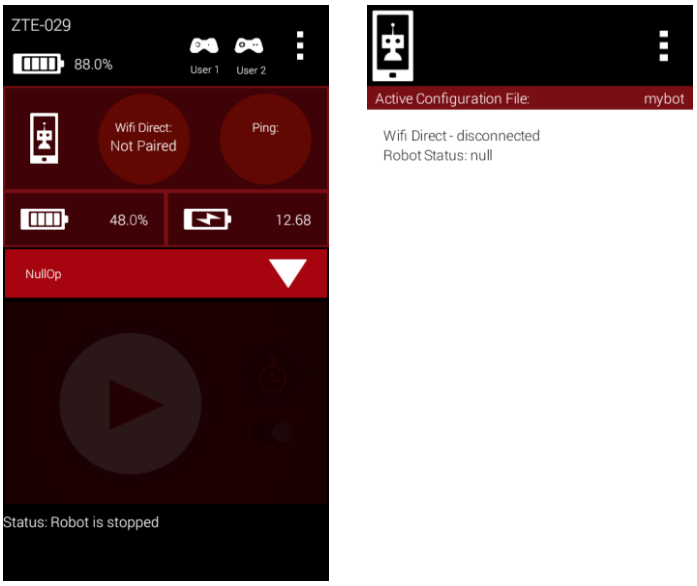


Figure 52 - The driver station and robot controller have not yet been paired together.

The pairing process is initiated through the driver station app. To start the pairing process touch the three vertical dots on the upper right hand corner of the FTC Driver Station app and then select "Settings" from the pop up menu.

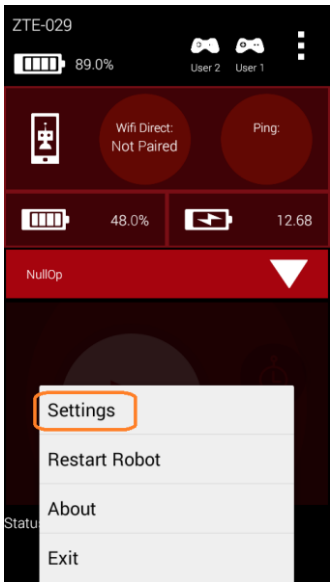


Figure 53 - Touch the three dots and select "Settings" from the pop-up menu.

Once you have launched the FTC Driver Station Settings menu, press the "Pair with Robot Controller" option to start the pairing process.

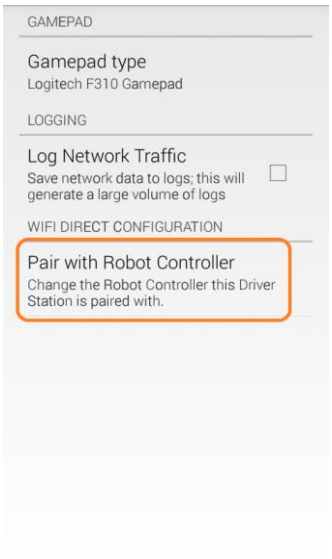


Figure 54 - Touch the "Pair with Robot Controller" option to start the pairing process.

From the driver station's WiFi Direct pairing activity, look for your robot controller in the list of available devices. In the example below, the robot controller is named "ZTE-030". Select the device and then use the Android back arrow to make a connection request from your driver station to the robot controller.

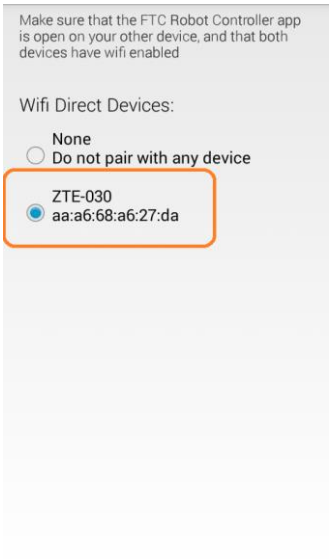


Figure 55 - Look for your robot controller in the list of available devices.

If this is the first time you are pairing to the robot controller, the robot controller device might display a prompt asking you if it is OK to allow an Android device to establish a WiFi Direct connection. In the example below, the driver station ("ZTE-029") has requested a connection with the robot controller ("ZTE-030"). Click on "Accept" to accept the connection request.

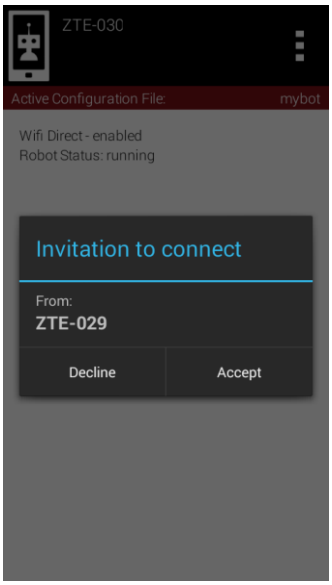


Figure 56 - Click Accept to accept the WiFi Direct connection request.

Once you have successfully paired your device to the robot controller, you should see a message on the driver station main screen indicating that the driver station is now connected to a robot controller. You should also start seeing Ping times being displayed in the upper right hand section of the Driver Station user interface.

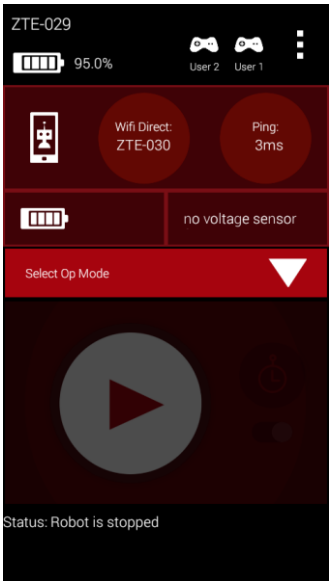


Figure 57 – This driver station is now connected to a robot controller named "ZTE-030".

Once your driver station is paired with its robot controller you are ready to begin executing op modes!

5.7 Selecting and Running Op Modes

Operational Modes or *Op Modes* are program modules that you can execute to have your robot perform specific behaviors. If your robot has been successfully paired to the driver station, then the Select Op Mode spinner on the FTC Driver Station app should now be enabled.

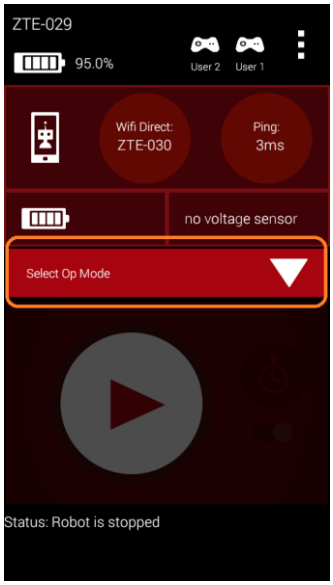


Figure 58 – Select Op Mode spinner is now enabled. Robot is stopped.

Note that if the Driver Station app is connected successfully to the Robot Controller app, then some ping statistics should be visible at the top right hand corner of the Driver Station screen. In the example of Figure 58 the ping statistics reads "3 ms". This means it took 3 milliseconds for the driver station to send a packet to the robot and for the robot to acknowledge the packet and send its acknowledgement back to the driver station.

The ping round trip time is a rough measure of network quality. As the round trip time increases, the network performance usually degrades. In our testing, an ideal round trip time is anything on the order of 10 milliseconds or lower. Sometimes you might see the ping time period increase if there is a surge in network traffic or interference. If the period starts to increase to the order of or greater than 100 milliseconds, then there might be excessive network traffic or noise on the wireless channel that your system is using.

Touch the **Select Op Mode** spinner to display a list of the available Op Modes for this robot. You can select an Op Mode from the list and the FTC Driver Station should indicate that the Op Mode has been queued for execution. For our example, there should only be two op modes available, **Stop Robot** and **Elapsed Op Mode** (which is the op mode we created in section 5.4.1).

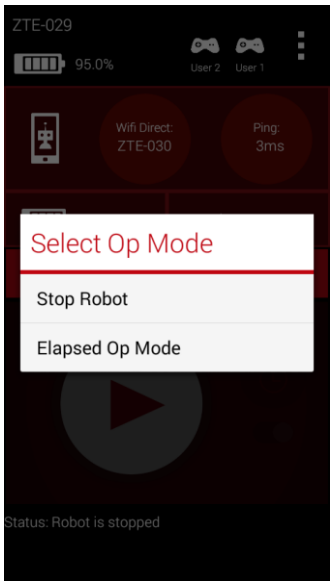


Figure 59 - Select Elapsed Op Mode from the list.

Select the **Elapsed Op Mode** from the list. Once you've made your selection, the selected op mode will be displayed on the main window of the Driver Station. Also, the **Start** button will become enabled.

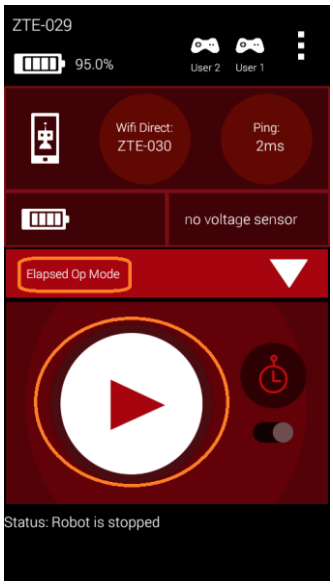


Figure 60 - Once an Op Mode has been selected, is listed in the Select Op Mode spinner and the Start button is enabled.

Press the **Start** button to begin running the op mode.

Figure 61 - Press the Start button to start the op mode.

When you start running your op mode, near the bottom of the screen you should see the telemetry information (current run time) displayed.

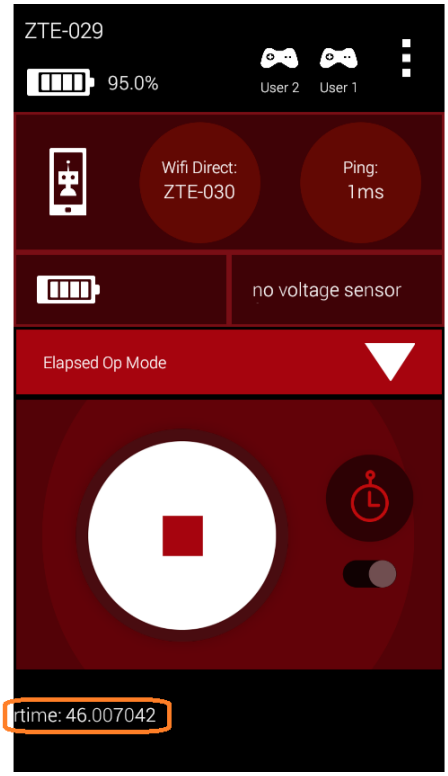


Figure 62 - The telemetry data (with key "rtime") is displayed towards the bottom of the screen

If you look closely at this telemetry information, you see that the key name “rtime” is displayed next to the telemetry data. In our example, we are sending the current time from the Robot Controller to be displayed on the Driver Station.

To stop the op mode, simply press the **Stop** button.

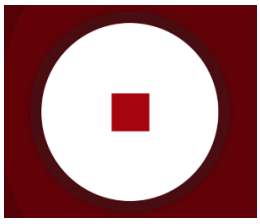


Figure 63 - Press the Stop button to stop the op mode.

Congratulations! You have just successfully run the op mode that you created with the App Inventor!

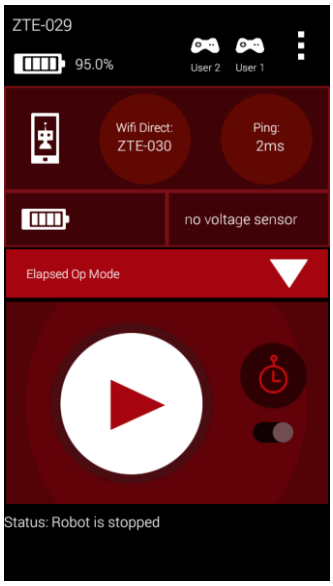


Figure 64 - Congratulations! You just ran the op mode that you had created with App Inventor!

6 Using Gamepad to Control a Motor

6.1 Hardware Setup

Let us now add an op mode to our Robot Controller app that will allow us to use a gamepad to control a DC motor. For this example, we will use a simple test setup. On the Driver Station side, we will have a single Logitech F310 gamepad connected to the Driver Station Android phone (using a Micro USB OTG adapter cable). Don't forget to make sure that the Logitech F310 gamepad has the little switch on the bottom set to the "X" position (see Figure 14).

Figure 65 - The Driver Station device should be connected to a single F310 gamepad using a micro USB OTG adapter.

In the previous example, we did not have any hardware devices connected to our Robot Controller. However, for this example, we will have a single Core DC Motor Controller connected to the phone through a Power Module.

Figure 66 – Use a 12V battery and the Power Module to connect the Motor Controller to the Android device.

In Figure 66 a 12V battery and a Power Module are used to connect an Android phone to a Motor Controller. A piece of red tape is placed on the shaft of the DC motor to make it easier to see if the shaft is moving. Turn the Power Module on *before* you connect it to the phone.

6.2 Adding the Design Components

Now that we have our hardware setup, let's use App Inventor to add another op mode to our controller app.

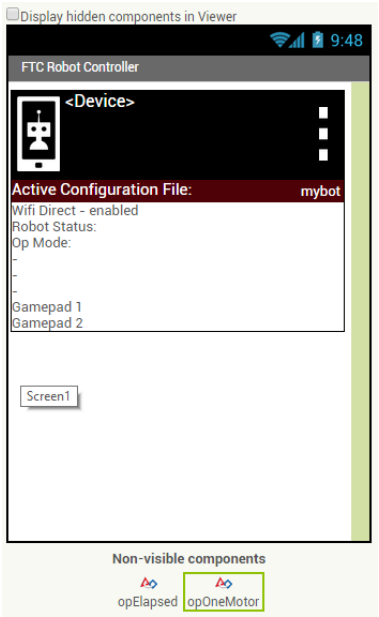


Figure 67 - Drag an FtcOpMode component onto your app screen and rename it to "opOneMotor".

Make sure you are in the **Designer** mode of the App Inventor and drag another FtcOpMode design component onto the main app screen in the **Viewer** pane. Rename the component to "opOneMotor" and in the **Properties** pane change the OpModeName property to "Drive One Motor".

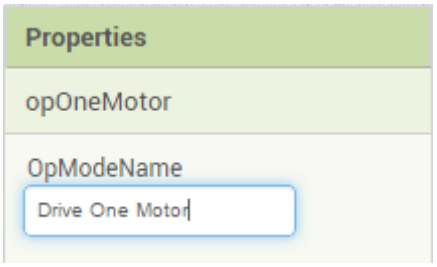


Figure 68 - Change the OpModeName property to "Drive One Motor".

Now we need to drag an FtcDcMotor design component onto the app screen. The FtcDcMotor is a hidden component that represents a DC motor on your robot.

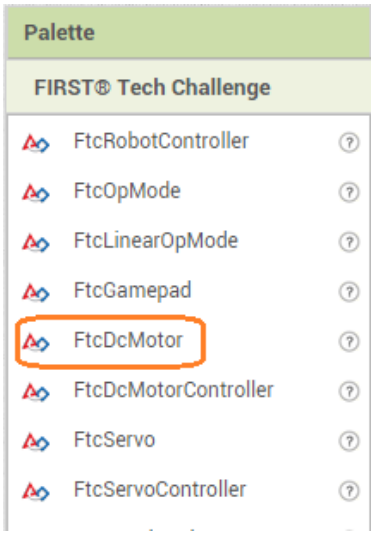


Figure 69 - Drag an FtcDcMotor component onto your app's main screen.

Rename the component to “dcMotor1”.

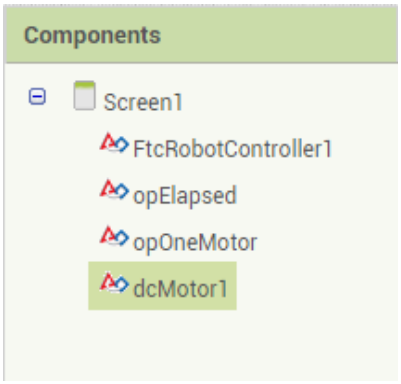


Figure 70 - Rename the component to “dcMotor1”.

In the **Properties** pane, for the dcMotor1 object, there is a property called **DeviceName**. The name that you specify in this property should be *the same* as the name that you use for this motor in the

configuration file that you will create for this robot. If the names do not match, your app will not be able to find this device during run time.

Change the **DeviceName** to “motor_1” in the **Properties** pane.

Figure 71 - Change DeviceName to "motor_1".

Now that we have a motor component in our app, we need to add a gamepad component. The gamepad component will allow us to receive user input from the driver station. Drag an FtcGamepad component onto the screen in the **Viewer** pane.

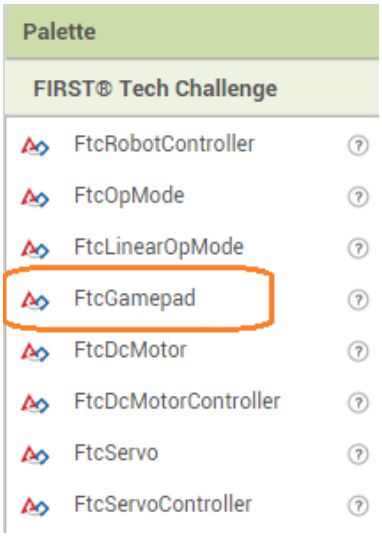


Figure 72 – Drag an FtcGamepad onto your app.

Rename the component to “gamepad1” and make sure that the **GamepadNumber** property is set to 1 (indicating that this represents driver #1).

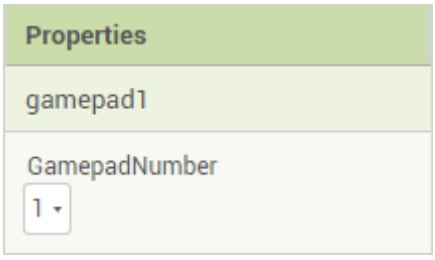


Figure 73 - Verify that the GamepadNumber is set to 1.

6.3 Adding the Op Mode Logic

Press the **Blocks** button (near the upper right hand corner of browser screen) to enter the block programming mode. Start by dragging a loop event handler for the opOneMotor op mode onto the **Viewer** window.

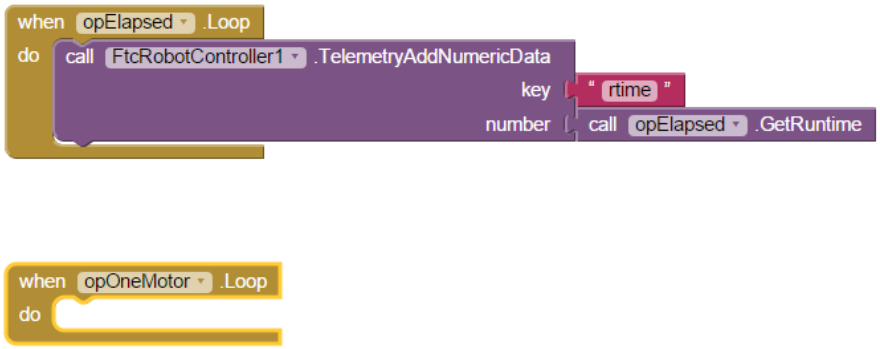


Figure 74 - Drag a Loop event handler onto the Viewer pane.

Let's define a global variable called "tgtPower" that we will use to store the value of the target power for our DC motor. A global variable is a variable that can be "seen" by all of the objects in our program. Click on the **Variables** category in the **Blocks** pane of the browser window.

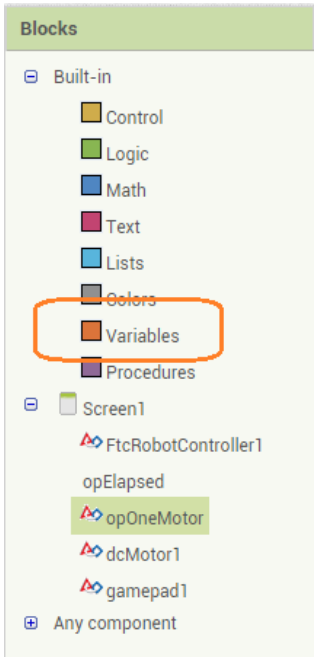


Figure 75 - Click on the Variables category of the Blocks pane.

Drag an "initialize global" block to the **Viewer** pane and drop it near the **opOneMotor.Loop** event handler block. We will use this block to create a global variable called "tgtPower".

Figure 76 - Drag an "initalize global" block to the Viewer pane.

Place the block near the whenOpOndeMotor.Loop block and change the name in the initialize global variable block to “tgtPower”.

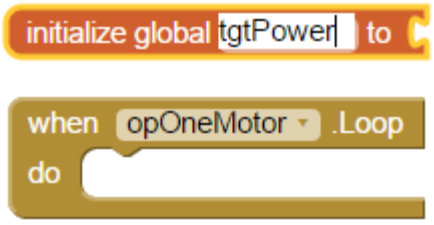


Figure 77 - Change the name of the global variable to "tgtPower".

From the **Math** category in the **Blocks** pane, drag a **0** Math block onto the initialize global block.

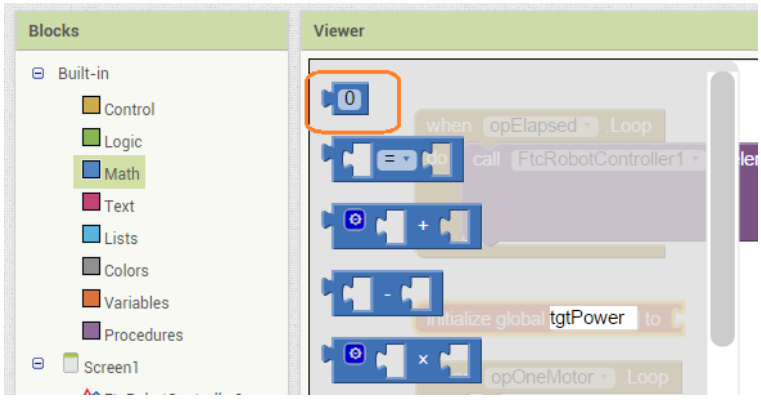


Figure 78 - Drag a "0" block (which represents a numeric value) onto the Viewer pane.

Initialize the global variable called tgtPower to a value of zero.

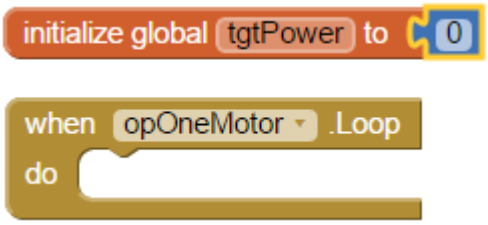


Figure 79 – Initialize the global variable to a value of zero.

Now we want to use the global variable called tgtPower. From the Variables category in the **Blocks** pane, drag a **set-to** block onto the **when opOneMotor.Loop** block.

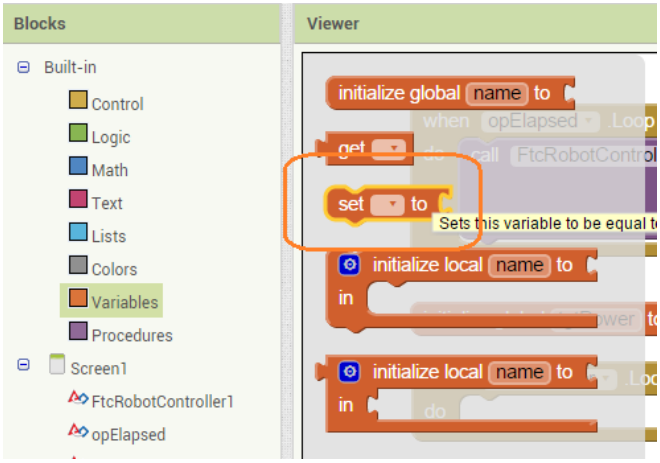


Figure 80 - Drag a set-to block onto the Viewer pane.

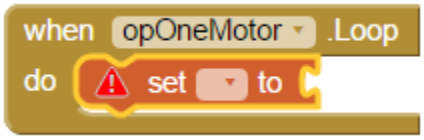


Figure 81 - Place the set-to block inside the Loop block.

From the **Math** category in the **Blocks** pane, drag a **neg** Math block onto the set-to block.

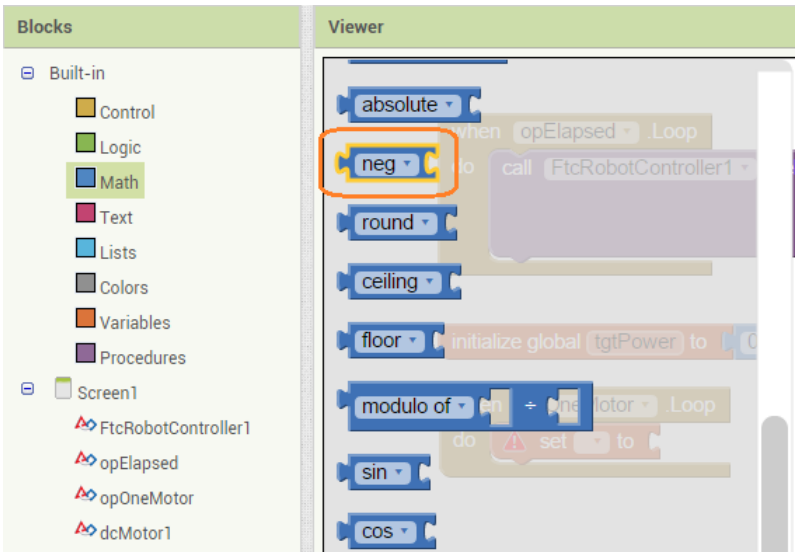


Figure 82 - Drag a "neg" block (which represents a minus sign) onto the Viewer pane.

Place the "neg" programming block in the "to" notch of the set-to block.

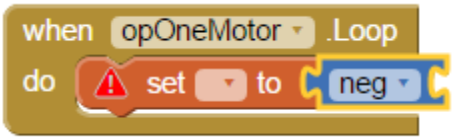


Figure 83 – Place the negative program block onto the "to" notch on the set-to block.

Click on the drop down selector of the set-to block and choose the global variable called tgtPower.

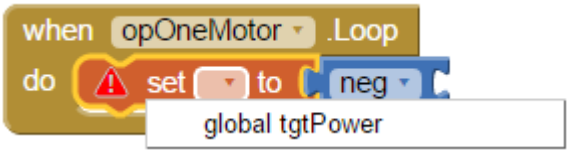


Figure 84 – Click on the drop down selector and select the global variable called tgtPower.

Click on the gamepad1 object in the **Blocks** pane of the browser and drag a gamepad1.LeftStickY programming block and place it in the notch of the **neg** programming block.

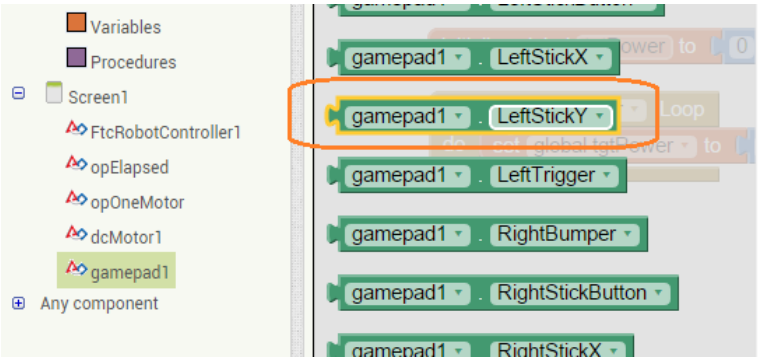


Figure 85 - Drag a gamepad1.LeftStickY block and place it onto the neg block in the Viewer.

Place the **gamepad1.LeftStickY** block into the notch of the **neg** programming block. This group of blocks will initialize a local variable called "tgtPower" equal to the negative value of Y position (vertical axis) of gamepad1's left joystick.

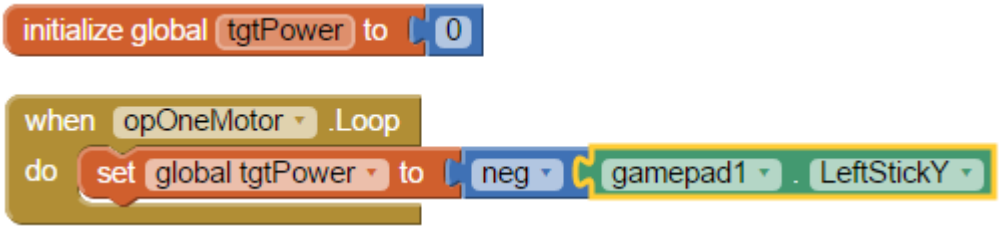


Figure 86 – This group of blocks initializes"tgtPower" to be equal to the negative value of the Y axis of the left stick.

We use the negative of the Y axis because a value of -1 for the Y axis of the gamepad represents the top most position and a value of +1 represents the bottom most position. A value of 0 represents the neutral position (when the joystick is in the middle). We want the opposite value for our motor power. When the joystick is pushed all of the way to the top (forward) we want the motor power to be the max value. When the joystick is pulled all of the way to the bottom, we want the motor power to be the minimum value.

We want to set the power level for our DC motor. Click on the **dcMotor1** object and select a **set dcMotor1.Power to** programming block.

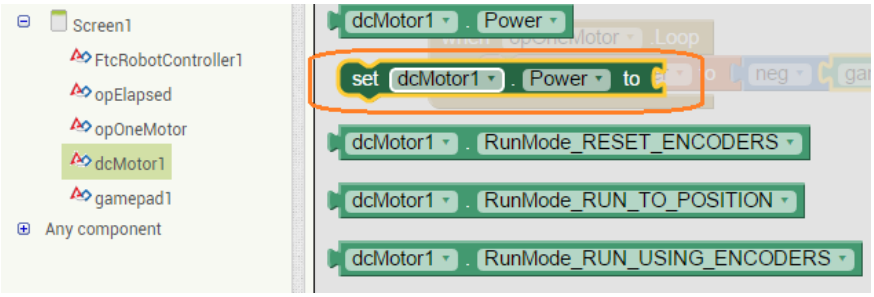


Figure 87 - Select a set dcMotor1.Power to programming block.

Drag the block below the **set global tgtPower** block.

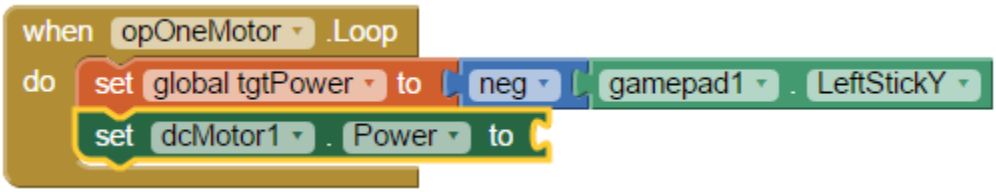


Figure 88 - Drag the set dcMotor1.Power to block to below the set global tgtPower block.

Select a **get** programming block from the **Variables** category and place it into the notch of the **set dcMotor1.Power to** programming block.

Figure 89 - Select a get programming block from the Variables category.

Click on the drop down selection window of the **get** programming block and select **global tgtPower**. This programming block will now return the value of the global variable called ”tgtPower”.

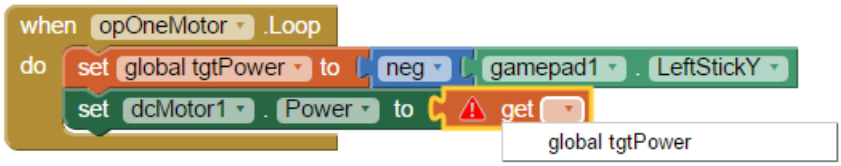


Figure 90 – Select global tgtPower for the get programming block.

This group of programming blocks is going to set the power of dcMotor1 to the value of the global variable called “tgtPower”.

Figure 91 - This group of blocks sets the power of dcMotor1 to the value of the variable with the name" tgtPower".

Let's add some programming blocks so that we can send the tgtPower value back to the driver station using the telemetry function. Click on the FtcRobotController1 component in the **Blocks** menu and drag a **call FtcRobotController1.TelemetryAddNumericData** programming block to the position under the **set dcMotor1.Power to** programming block.

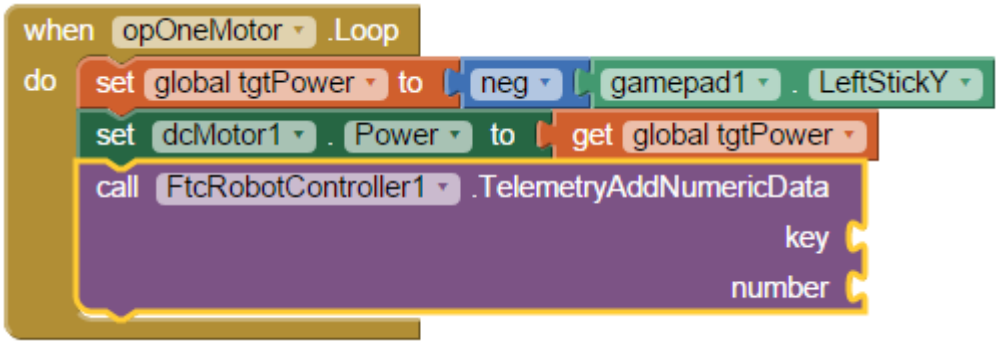


Figure 92 – Drag a call FtcRobotController1.TelemetryAddNumericData block to the space below the set dcMotor1.Power block.

Set the **key** value of the programming block to "power" and drag a **get** programming block for the **number** value. For the **get** programming block, select "global tgtPower" as the variable.

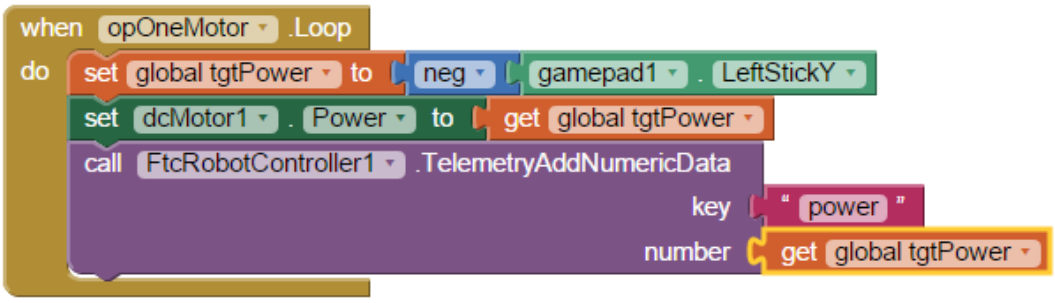


Figure 93 - Set the key to "power" and the number to the value of the global variable "tgtPower".

6.4 Configuring Your Robot Controller with the New Hardware Info

Build your app (select **Build->App (save .apk to my computer)**) and use a USB cable to copy it to your Android device.

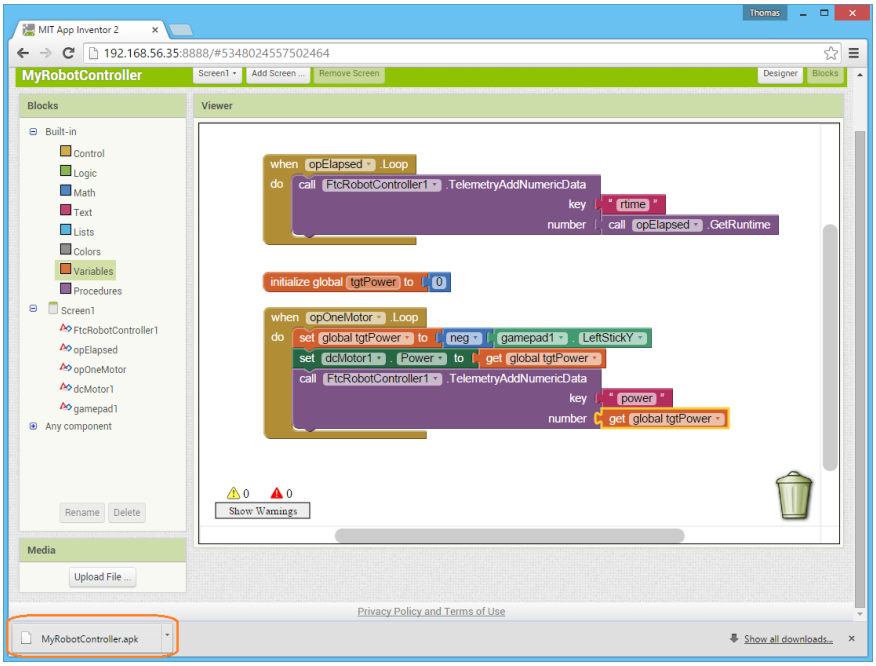


Figure 94 - Build the app and download it from the virtual server to your PC.

Before you install your Robot Controller app onto your Android device, you should first uninstall any previous copy of the app from your Android device.

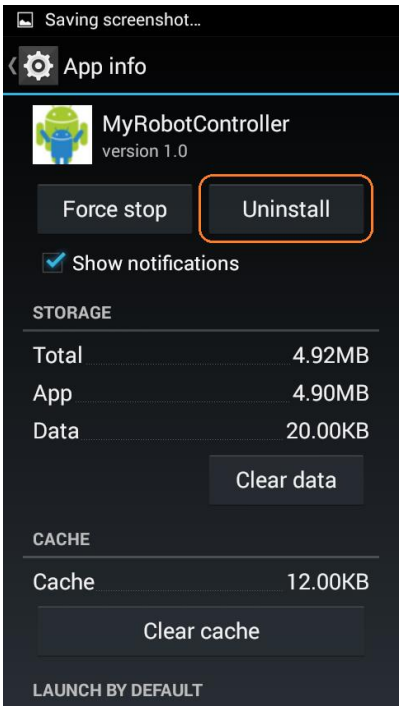


Figure 95 - Uninstall the previous copy of MyRobotController before you attempt to install the new copy.

Once you have installed the app, we will need to update the configuration file to include the DC motor controller. Make sure that your Power Module and DC Motor Controller are turned on *before* you connect your Android device to the DC Motor Controller.

Figure 96 - Make sure the DC motor controller has 12V power. Connect Android device to the motor controller.

When you connect the motor controller to your Android device, your Android device should ask you if it is OK for the MyRobotController app to access this USB device (the DC motor controller). You should check the checkbox ("Use by default for this USB device") so that the MyRobotController app will be the default app whenever this type of USB device is detected. Hit **OK** to allow the MyRobotController app to access the DC Motor Controller.

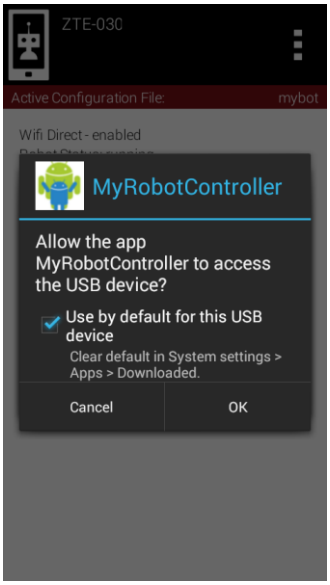


Figure 97 - Check the "Use by default for this USB device" and hit OK to grant access to the DC Motor Controller.

Your Android device might ask you if it's OK to open the MyRobotController app whenever the DC motor controller device is detected on the USB port. Click on the checkbox ("Use by default for this USB

device") and hit OK to allow the app to be launched whenever the DC motor is detected by your Android device.

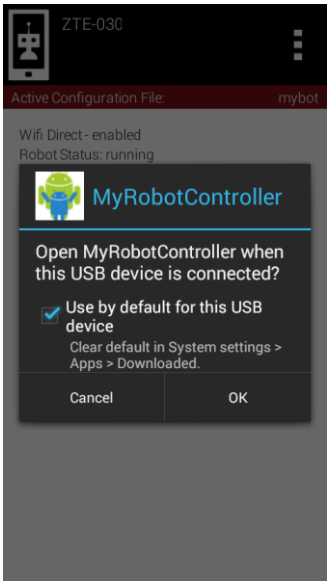


Figure 98 - Click on the check box and hit OK.

The main Robot Controller screen should appear. Touch the three dots in the upper right hand corner of the screen to display the **Settings** screen. Select the **Configure Robot** item to launch the configuration screen.

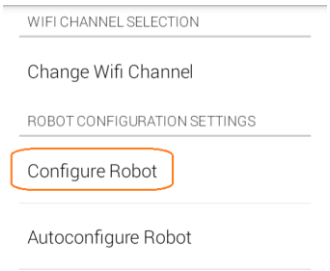


Figure 99 - Click on Configure Robot from the Settings screen.

In the configuration screen, you should see a listing for a configuration file that is called "mybot". Up till now, this configuration file should be empty (i.e., no hardware devices defined within the file). Click on the **Edit** button to edit the "mybot" file.

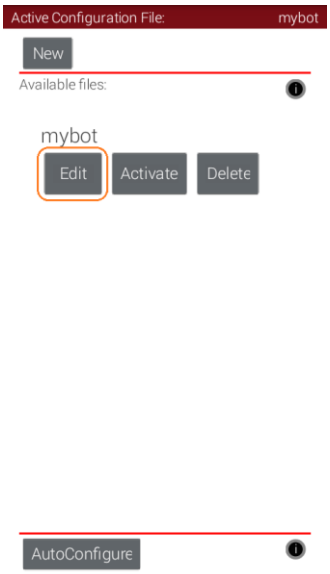


Figure 100 - Click on Edit to edit the "mybot" configuration file.

The screen for the “mybot” configuration file should not have any devices listed. Click on the **Scan** button to make the app scan for available hardware modules on the USB bus.

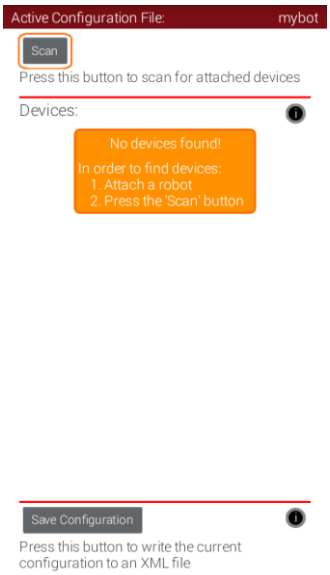


Figure 101 - Press the Scan button to scan for hardware devices.

When you press the **Scan** button, the app should detect the DC motor controller and list it as “Motor Controller 1” on the screen.

Figure 102 - The app should detect the motor controller and list it on the screen.

The app was able to determine that there is a DC motor controller connected to its USB port. However, you still need to configure that motor controller and tell your app what devices are connected to the motor controller. It cannot determine this information on its own. You will need to edit the "Motor Controller 1" device and tell the app that you have a DC motor plugged into channel 1 of the motor controller. Click on the words **"Motor Controller 1"** to edit this device. A screen should appear that allows you to edit the configuration for "Motor Controller 1". Check the "Attached" checkbox near port #1 and click on the **Motor name** field to specify the name of the motor.

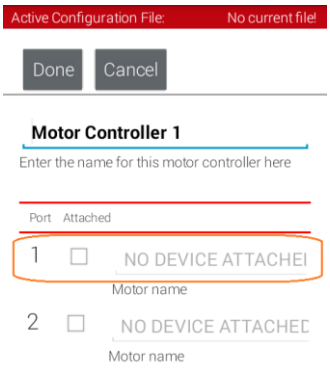


Figure 103 - You need to tell the app that there is a motor attached to Port #1.

The name that you specify for this motor should be the *same* name that you specified in the **Properties** pane for the dcMotor1 object (see Figure 71 on page 41). If the name doesn't match, then your app will

not be able to find the device during run time. Type in the name “motor_1” for the motor attached to port #1 of your motor controller then hit the **Done** button.

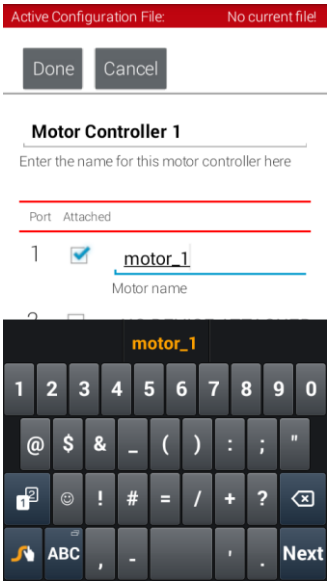


Figure 104 - Make sure the Attached box is checked and specify the name "motor_1" in the Motor name field.

After you have finished editing the “Motor Controller 1” definition, click on the **Save Configuration** button to save this updated configuration file.

Figure 105 - Click on Save Configuration to save the updated configuration file.

When prompted for a file name, use the name “mybot” to match the name that you specified in your App Inventor design (see Figure 24 on page 18). If your filename does not match the one that you specified in your design, then your app will not be able to find the configuration file during runtime.

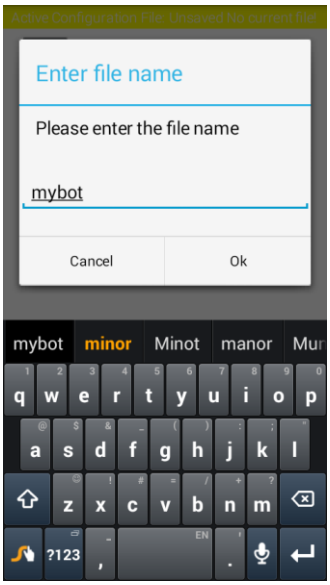


Figure 106 - Specify a file name of "mybot" and hit Ok to save.

Once your configuration file has been saved, you should use the back arrow to return to the main screen of the Robot Controller app. When you return to this screen, the app should scan its USB port to see if it can find the DC motor controller that you configured in the "mybot" configuration file. If it finds the device successfully, it should list its status as "running".

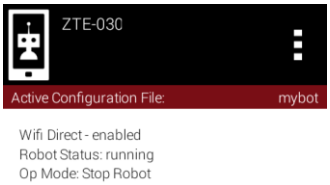


Figure 107 - Your Robot Status should be listed as "running".

6.5 Running the "Drive One Motor" Op Mode

Now that you have successfully installed the new version of your app onto your Robot Controller Android device, you need to switch to your Driver Station to select and launch the op mode.

From the driver station, verify that your gamepad has been designated User #1 by pressing the **Start** and **A** buttons at the same time on the gamepad (see section 4.2.2 on page 12).

Figure 108 - Designate your gamepad as User #1 by pushing the Start and A buttons simultaneously.

Click on the **Select Op Mode** spinner to display a list of available op modes.

Figure 109 - Click on the Select Op Mode spinner to display a list of available op modes.

Select **Drive One Motor** from the list of available op modes.

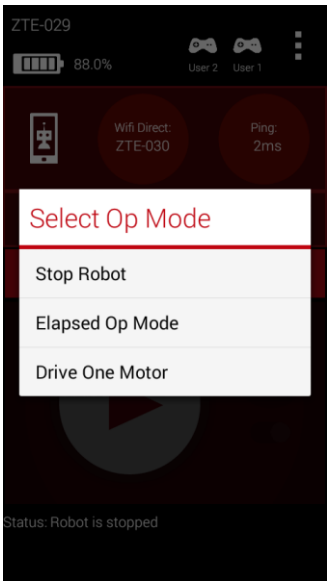


Figure 110 - Select the Drive One Motor op mode from the list of available op modes.

Press the Start button to run the op mode.

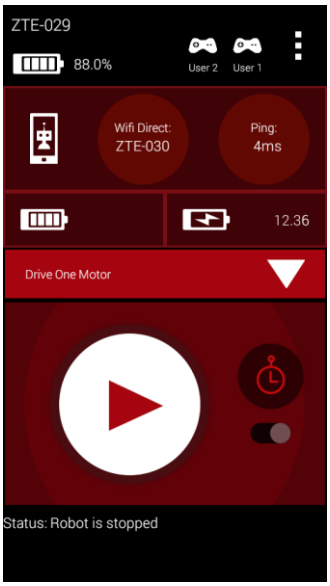


Figure 111 - Press the Start button to run the op mode.

As the op mode is running, you should be able to use the left joystick on the gamepad to control the motor. If you push the joystick forward, the motor will spin one way. If you pull the stick backwards, the motor will spin the opposite way. The target motor power should be displayed as a telemetry item in the lower portion of the driver station screen.

Figure 112 – The target power should be displayed as a telemetry item in the lower portion of the screen.

Congratulations! You have written a simple driver-controlled app using the App Inventor!

7 Using a Gamepad to Control a Servo

This section provides an example on how to control a servo using a gamepad. Unlike the previous section, this will not follow the process in a step-by-step manner. Instead this section will just provide an overview on how to modify your existing app to control a servo using two buttons on the gamepad.

7.1 Hardware Setup

For this example we are adding a servo controller to our test setup. Since we now have more than one module, we need to use a Power Module to provide 12V power to the DC Motor Controller and to the Servo Controller. We also need to use the Power Module to provide USB connectivity from the Android device to the Motor and Servo Controllers.

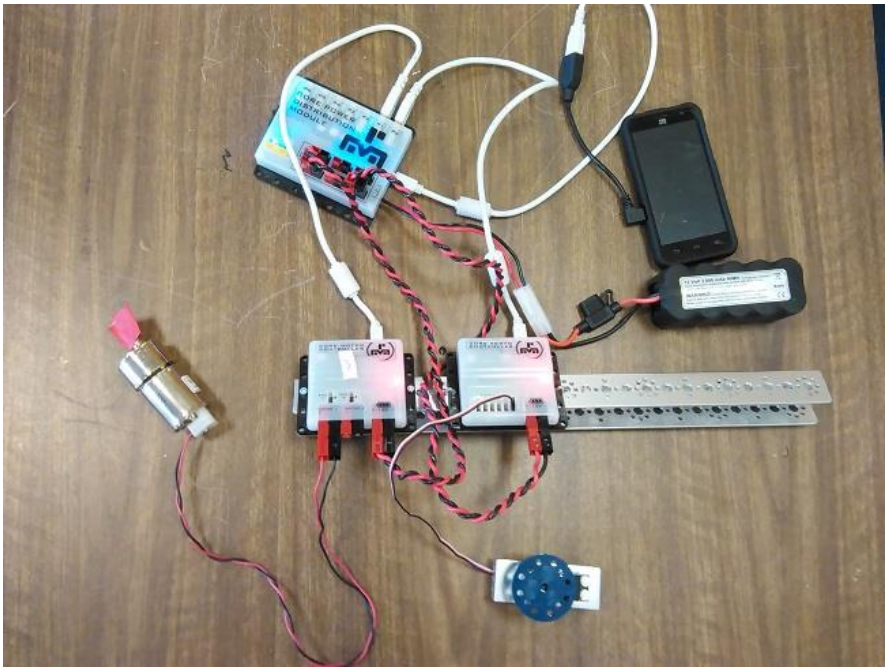


Figure 113 - Add a Servo Controller, a servo, and a Power Module (plus cables) to the setup.

7.2 Modifying the Visual Design

For this example, add the following components to your app:

- FtcOpMode – Change its name to **opSingleServo** and the OpModeName property to “Run Single Servo”.
- FtcServo – Change its name to **servo1** and the DeviceName property to “servo_1”.

Remember that the DeviceName for the servo will have to match the name used in the configuration file.

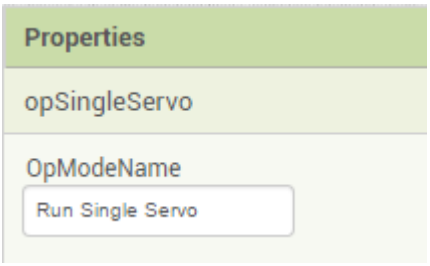


Figure 114 - Rename the FtcOpMode component to "opSingleServo" and change its OpModeName to "Run Single Servo".

Figure 115 - Rename the FtcServo component to "servo1" and change its DeviceName to "servo_1".

7.3 Adding the Op Mode Logic

Switch to Blocks mode and create the following logic with the programming blocks.

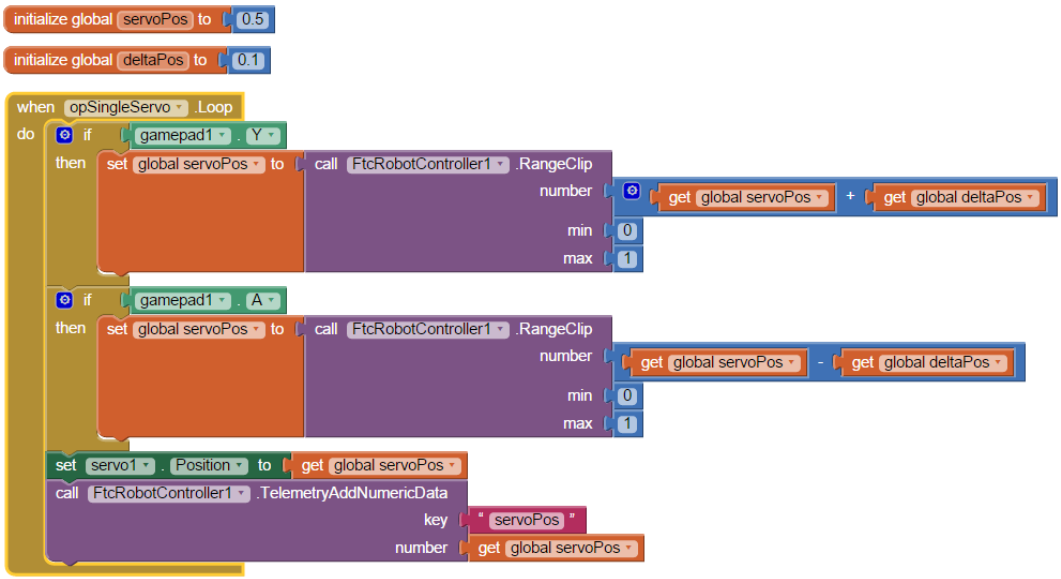


Figure 116 - Add the following logic to your app.

The programming blocks shown in Figure 116 will check the status of the **Y** and **A** buttons on gamepad #1 of the driver station. If the **Y** button is depressed, then a global variable called "servoPos" will be incremented. If the **A** button is depressed, then the global variable "servoPos" will be decremented. The range of "servoPos" is limited to a minimum value of 0 and a maximum value of 1. The servo position is set to the value the global variable "servoPos". Also the "servoPos" value is sent back to the driver station using the telemetry function.

7.4 Modifying the Configuration File

Currently, with the FTC Robot Controller module, if you want to add an additional piece of hardware to the configuration file, you will have to redo the entire configuration file. In this example, since we added a servo controller to the USB bus, you will need to reconfigure the "mybot" configuration file.

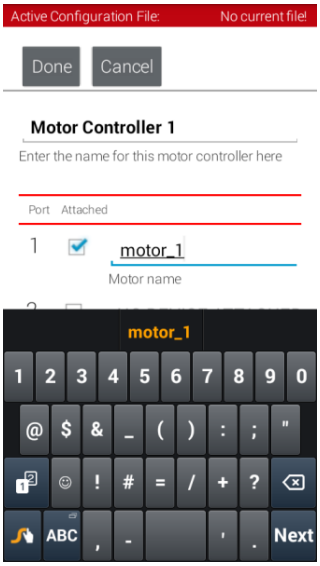


Figure 117 - You'll have to reconfigure the motor controller again.

You'll have to reconfigure the motor controller again. You should reconfigure the motor controller so that port #1 is named "motor_1". Also, for this example op mode, you'll need to configure the servo controller.

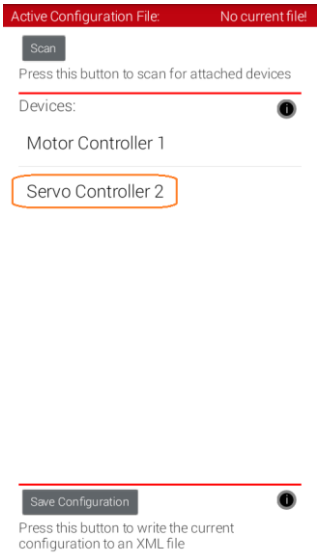


Figure 118 - Click on the Servo Controller item to configure it.

Modify the Servo Controller configuration to add a servo called "servo_1" to port #1 and click **Done** to close the configuration screen.

Figure 119 - Check the Attached box for port #1 and name the servo "servo_1".

Save the configuration file as “mybot”.

7.5 Running the “Run Single Servo” Op Mode

From the driver station, select the “Run Single Servo” op mode on the Driver Station and push the Start button to run op mode.

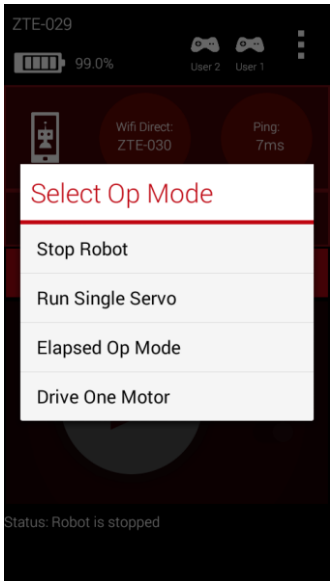


Figure 120 - Select the "Run Single Servo" op mode and push the Start button to run the op mode.

When you start the op mode, if you push the **Y** or **A** buttons on gamepad #1, the servo will rotate one direction or the other. The servo position will be displayed as a telemetry item on the lower portion of the Driver Station screen.

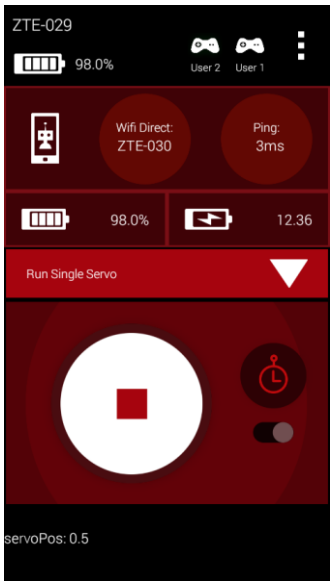


Figure 121 - The Y and A buttons control the servo motion, the servo position is displayed near the bottom of the screen.

8 Using the Core Device Interface Module

Let's now use the Core Device Interface Module to interact with some sensors. For this example, let's create a brand new project file and create a new Robot Controller app.

8.1 Important Note Regarding Multiple Robot Controller Apps

We are going to create a new project and build a new Robot Controller app that is different from the project file that was used for sections 5, 6 and 7 of this document. We will call our new app "CDITestController". It will be a Robot Controller app like the previous "MyRobotController" app that you built. However, when we want to test our new app, it is important that we first remove the old "MyRobotController" app from our phone before we install the new one.

In general, you only want to have at most *one* Robot Controller app per Android device. If you have more than one Robot Controller app per Android device, then some problems can occur when you connect your Android device to your USB modules (such as the DC motor controller and the servo controller). If a USB module is associated with one Robot Controller app, and a different type of USB module is associated with another Robot Controller app, then you might have problems when you try to execute your op modes. We recommend that you only have at most one robot controller app per Android device.

8.2 Hardware Setup

For this example, we will have a Core Device Interface Module connected to our phone (through a Power Module that is powered by a 12V battery). To start we will have a single Touch Server connected to digital port #7 (there are eight digital ports numbered 0 through 7).

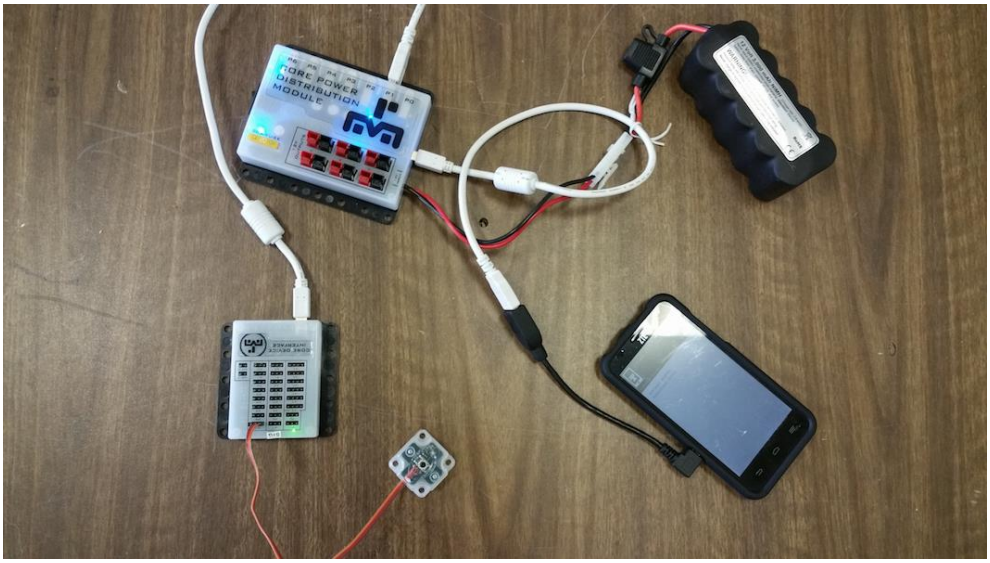


Figure 122 - Connect the CDI to the phone through the Power Module. Plug Touch Sensor into port D7.

Important note: when connecting the touch sensor to the digital I/O port, the ground wire (which is brown) should be on the right hand side of the CDI module when looking at the module with the words right-side up (see figure below).

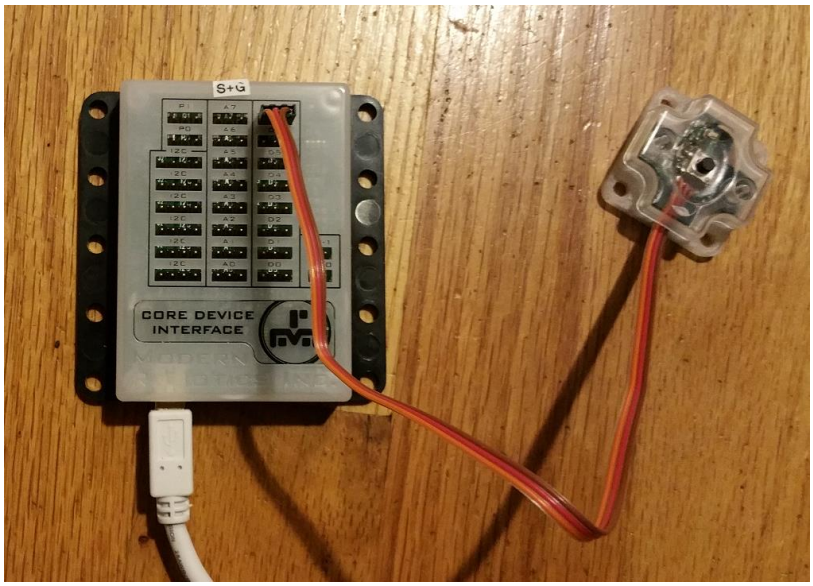


Figure 123 - The ground cable (brown in this example) should be plugged into the right side of digital port #7.

8.3 Create a New Project

From the **Projects** menu near the top of the screen, select the **Start new project** menu item.

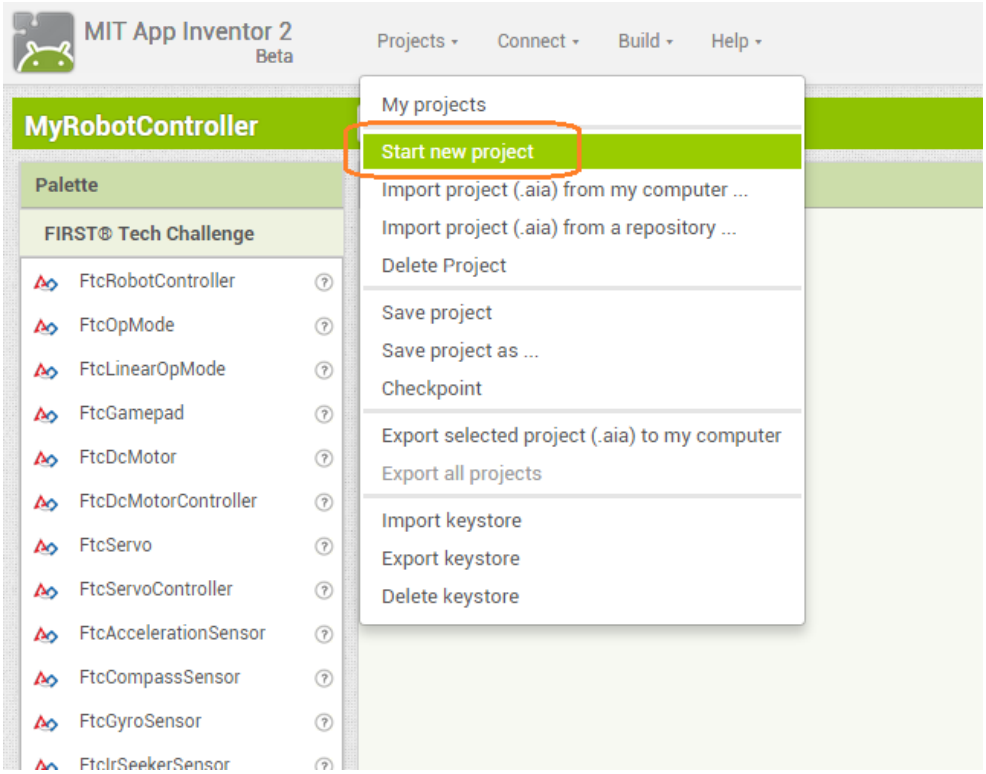


Figure 124 - Select Start new project from the Projects menu.

Name the new project “CDITestController”.

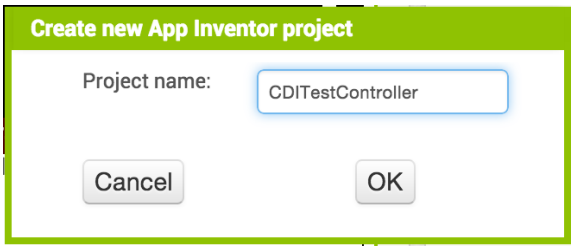


Figure 125 - Name your project "CDITestController" and hit OK.

Drag an **FtcRobotController** design component onto your app and specify the **Configuration** property as “cdi_config”. This is the name that we will have to use when we create our configuration file for our test setup.

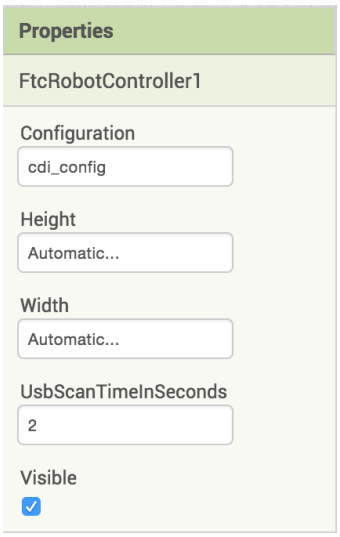


Figure 126 - Drag an FtcRobotController component onto your app and specify "cdi_config" in the Configuration property.

Next, drag an **FtcOpMode** component onto your app and rename it “opTouch” and set the **OpModeName** property to “Touch Test”.

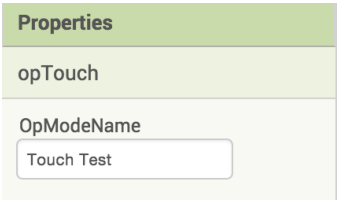


Figure 127 - Drag an FtcOpMode component onto your app and specify "Touch Test" as its OpModeName.

Drag an FtcTouchSensor block onto the app, and rename it to sensorTouch. Specify its device name as “touch_1”. This is the name that you will need to use for this device in your configuration file. If the names do not match, then your app will not be able to find and access this device at runtime.

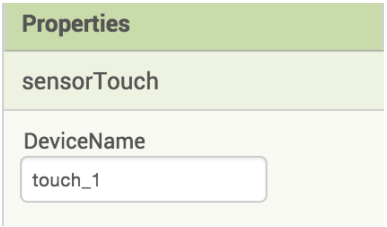


Figure 128 - Drag a touch sensor component and specify its DeviceName as "touch_1" in the Properties pane.

Finally, drag an FtcDeviceInterfaceModule component onto your app. Rename the component to “devinterface” and set its DeviceName to “cdi”. This is the name (“cdi”) that you need to specify for this module in the configuration file. If the names do not match, then the app will not be able to find the module at runtime.

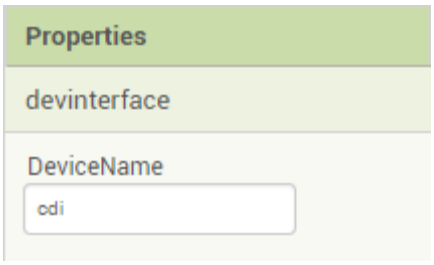


Figure 129 - Change the DeviceName to "cdi".

When you are done, your app should have an FtcRobotController block, an FtcOpMode block an FtcTouchSensor block, and an FtcDeviceInterfaceModule in its Design mode **Viewer** pane.

Figure 130 - You should have FtcRobotController, FtcOpMode, FtcTouchSensor, FtcDeviceInterfaceModule blocks.

8.4 Adding the Op Mode Logic

Switch to the Blocks programming mode and use the FTC programming blocks to create the following Blockly program.

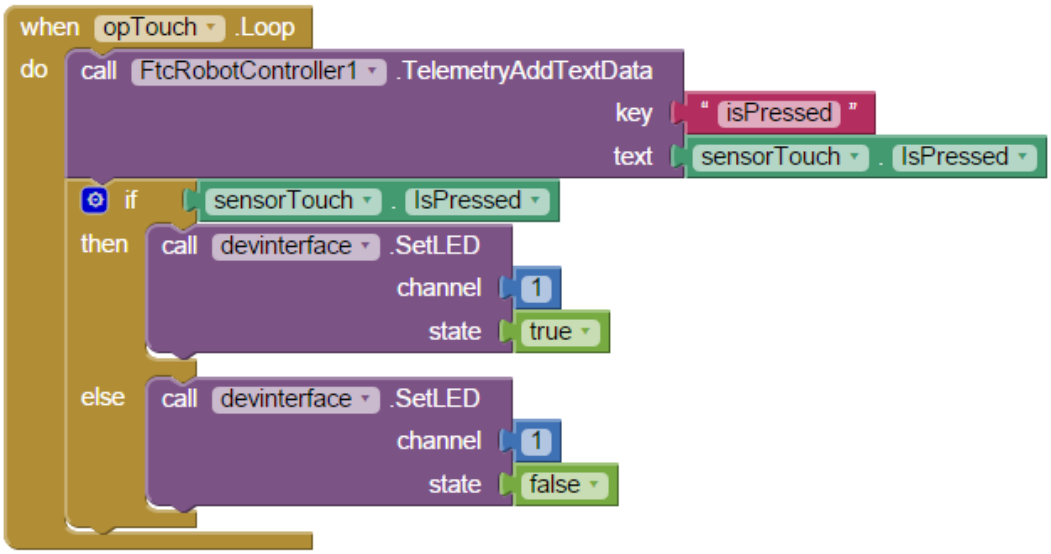


Figure 131 - Drag the programming blocks onto the Viewer pane to create this Blockly program.

Note that the Blockly program in Figure 131 uses an **if then else** programming block. If you look under the **Control** category of programming blocks you will only see an **if then** programming block.

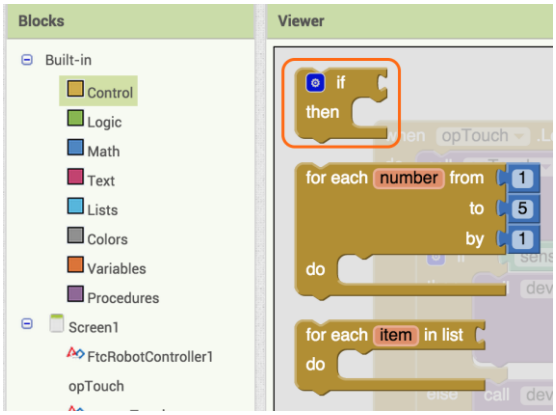


Figure 132 - There is an if then programming block under the Control category of Built-in blocks.

You can convert an **if then** programming block into an **if then else** block by placing an **if then** block on the **Viewer** pane, and then pressing the little blue symbol in the upper left hand side of the programming block. This should bring up a pop-up menu.

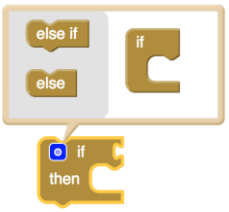


Figure 133 - Press the blue symbol in the upper left corner to bring up a pop-up menu.

In the pop-up menu, drag the **else** block and connect it to the **if** block to convert the block from an **if then** block to an **if then else** block.

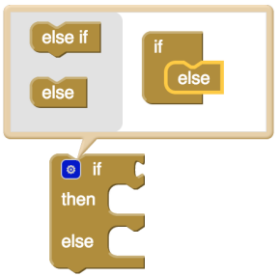


Figure 134 - In pop-up menu, drag the else-block to if-block to make an if-then-else-block.

Here is a description of the logic for the Blockly program shown in Figure 131. For each loop() event, the app will send as a telemetry item to the Driver Station the status of the digital button. The key name for this telemetry item is "IsPressed". After the telemetry data is sent, the app will check the status of the button. If it's pressed, it will set LED #1 (which should be the built-in red LED on the Core Device Interface module) to a true state (i.e., on). Otherwise, the app will set LED #1 to a false state (off).

8.5 Installing and Configuring the App

After you have completed generating the program logic for the app, select the **Build->App (save .apk to my computer)** menu item to build the .apk file and download it to your computer from the virtual server. Before you transfer this file to your Android device and install it, you should first uninstall any previous Robot Controller type of apps from the device.

Remember, we only recommend that you have a single Robot Controller type of app on your Android device. If you have more than one, it might cause problems when your apps try to detect USB modules on the USB connection/bus.

To uninstall the app, go to the **Settings** menu on your Android device. In the **Settings** menu, click on the **Apps** item to launch the Apps Settings activity.

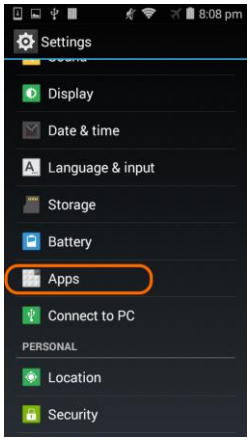


Figure 135 - Click on the Apps icon to launch the Apps activity.

Click on your older Robot Controller app to display information about this app.

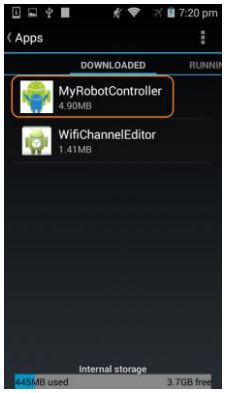


Figure 136 - Click on your old Robot Controller app.

Click on the **Uninstall** button to uninstall your old app.

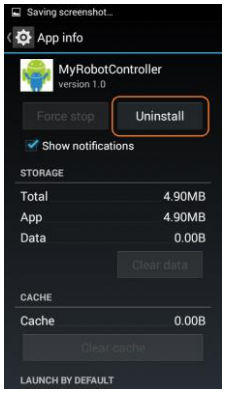


Figure 137 - Click on Uninstall to uninstall the app.

Click on **OK** to remove the app.

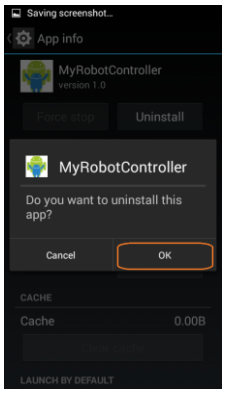


Figure 138 - Click OK to remove the app.

The app should no longer be listed in the **Downloaded** tab of the **Apps** activity.

Figure 139 - The uninstalled app should no longer be listed in the Downloaded tab.

After you uninstall any previous Robot Controller apps, transfer the .apk file to your Android device and install it. Make sure that your Power Module is turned on before you connect your Android device (if it isn't already connected).

In the CDITestController app go to the **Settings** menu and select **Configure Robot** to launch the Configuration screen. Touch the **New** button to create a new configuration file. Hit the **Scan** button to scan for any USB devices. The app should detect your CDI module and list it as "Core Interface Module 1". Touch on this listing to configure the module.

Figure 140 - After you Scan for devices you should see Core Interface Module 1 listed.

Change the module name from “Core Interface Module 1” to “cdi” so it will match the DeviceName defined in the app (see Figure 129).

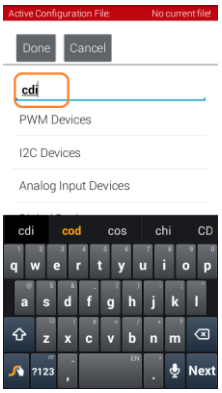


Figure 141 - Rename the device to "cdi".

After you change the module’s name, click on the **Digital Devices** item to configure the digital I/O ports on the Device Interface module.

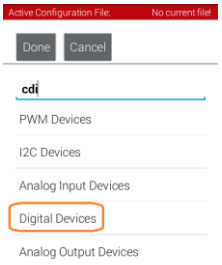


Figure 142 - Click on the Digital Devices item to configure the digital ports.

The digital touch sensor should be connected to digital port #7. Touch the drop down selector for port #7 and select **TOUCH_SENSOR** for this port.

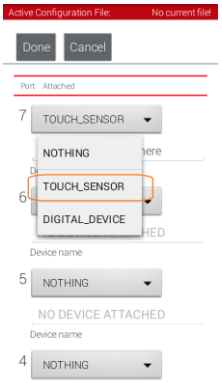


Figure 143 - Select TOUCH_SENSOR for port #7.

Name the touch sensor “touch_1” to match the DeviceName that you defined in your app (see Figure 128). Click **Done** when you are finished.

Figure 144 - Rename the touch sensor "touch_1". Click Done when you are finished.

Press the **Save Configuration** button to save this configuration information to a file.

Figure 145 - Press Save Configuration.

Name your file “cdi_config” and press **OK** to save the configuration. Remember the file name has to match the Configuration property you defined in the app (see Figure 126).

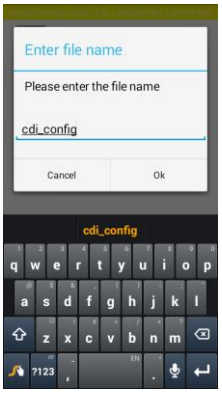


Figure 146 - Save the file as "cdi_config".

After you have saved the configuration file, use the back arrow to return to the main screen. Note that after you saved your configuration file, you might have to manually restart your robot and force the app to check for the configured hardware module.

Figure 147- After you saved the configuration file, you might need to manually restart the robot.

To restart the robot, launch the **Settings** menu, then select **Restart Robot**.

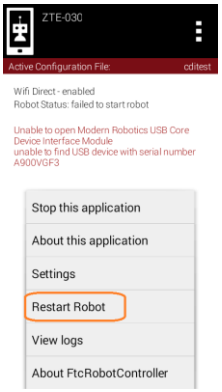


Figure 148 - Click on Restart Robot from the Settings menu to restart the robot.

The app will scan for the configured hardware module upon restart.

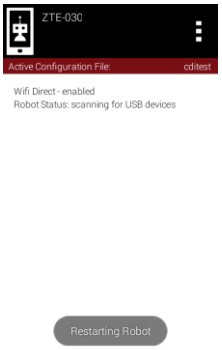


Figure 149 - The app will scan for the configured USB module upon restart.

The app should have a status of “running” upon a successful restart.

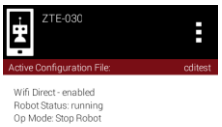


Figure 150 - The device should have a status of "running" after a successful restart.

8.6 Running the Op Mode

Use the Driver Station to select and run the Touch Test op mode.

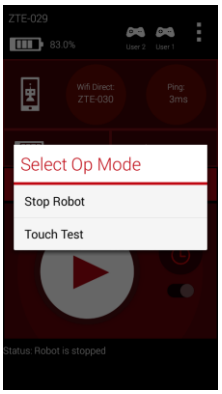


Figure 151 - Use the Driver Station to select and run the Touch Test op mode.

The Driver Station should display the state of the button at the bottom of the screen (where the telemetry info is displayed). The value should be “false” if the button is not pressed and “true” if the button is pressed.

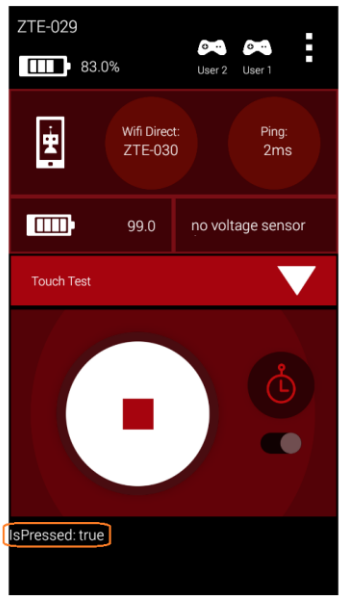


Figure 152 - The button is "true" if pressed.

Also, on the Device Interface module, the red LED should illuminate whenever the button is pressed while the op mode is running.

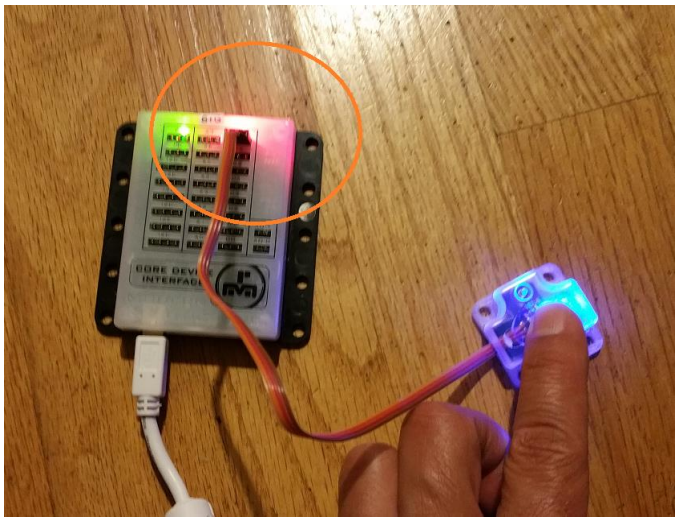


Figure 153 - The red LED should illuminate when the button is pressed while the op mode is running.

9 Optical Distance Sensor Example

The modern robotics optical distance sensor is an analog sensor that can also be used as a reflected light sensor.

Figure 154 - Modern Robotics Optical Distance Sensor

The optical distance sensor can be plugged into an analog port on the CDI.

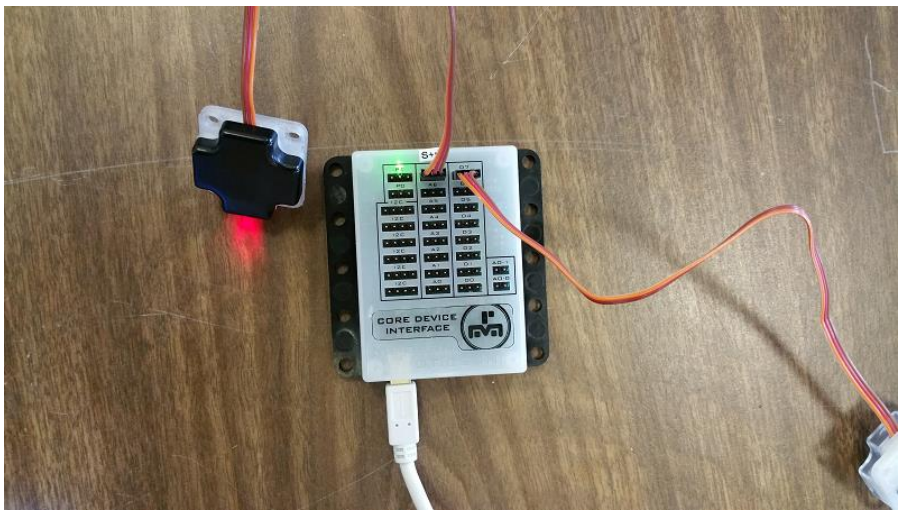


Figure 155 - You can connect the Optical Distance sensor to an analog input port on the CDI module.

You can modify your CDITestController app to include a new op mode to display data from the analog sensor.

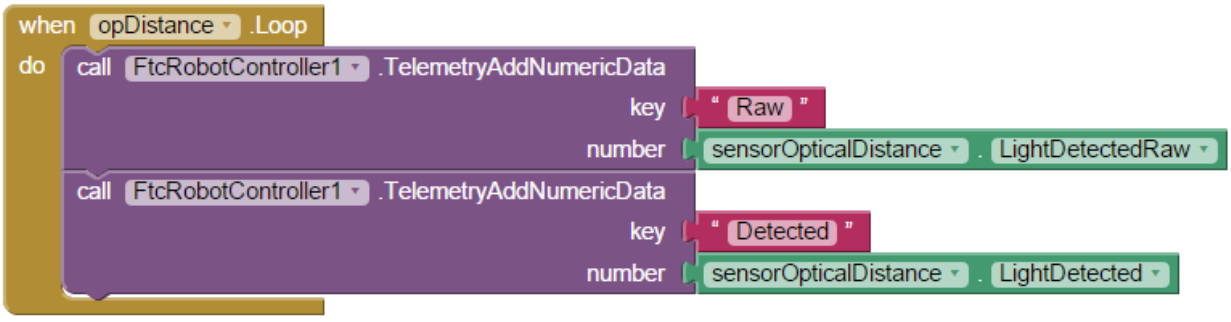


Figure 156 - You can use this group of blocks to display the sensor output on the Driver Station.

For the Raw value, the number should be higher when more reflected light is detected, and lower when less light is detected.

10 IR Seeker V3 Example

The Modern Robotics IR Seeker V3 sensor can be used to detect IR signals (AC and raw DC).

Figure 157 - Modern Robotics IR Seeker v3 sensor.

The IR Seeker v3 sensor can be plugged into one of the I²C ports on the device.

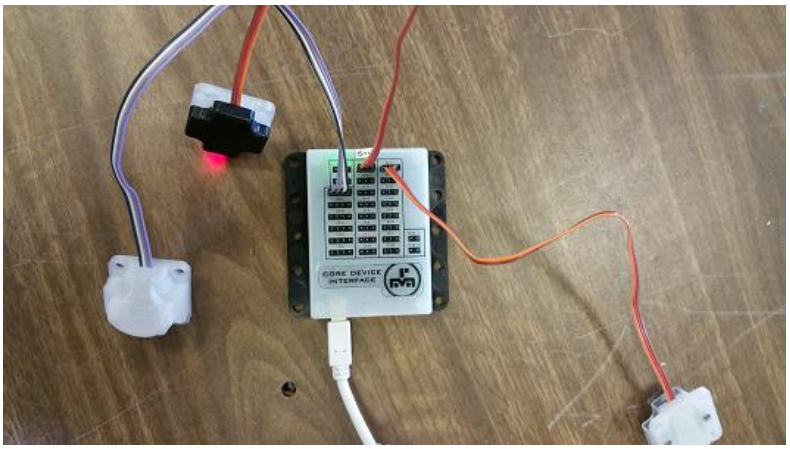


Figure 158 - The IR Seeker v3 sensor (shown in white in this figure) can be connected to an I²C port.

The following programming blocks can be used to send IR seeker data back to the Driver Station.

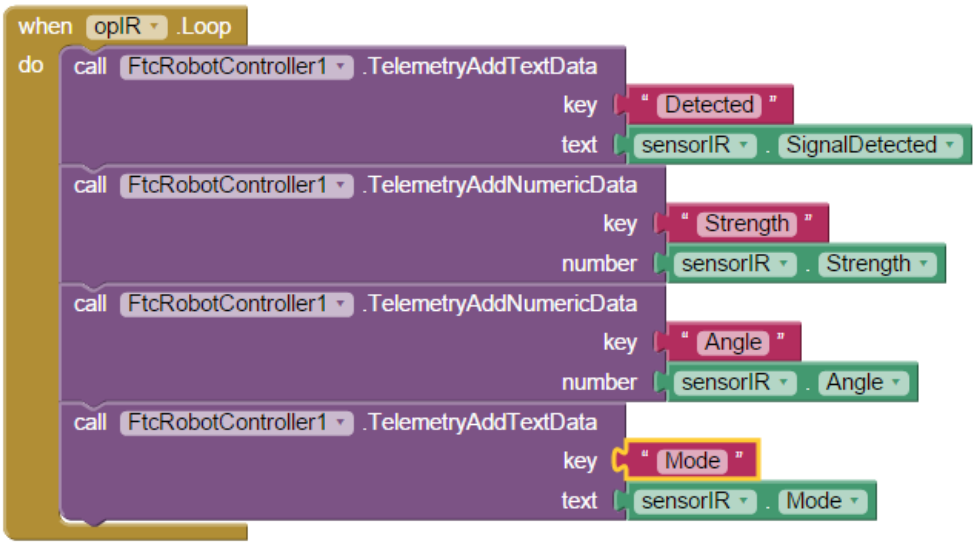
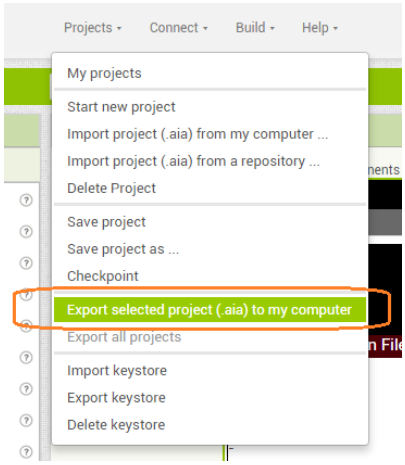


Figure 159 - These programming blocks can be used to display IR Seeker info on the Driver Station.

11 Useful Tips

11.1 Importing and Exporting App Inventor Projects

The App Inventor lets you export your App Inventor project into a file that is downloaded onto your computer. In your App Inventor browser screen, select **Projects->Export selected project (.aia) to my computer**. This will export your file to a .aia file and save it to your local computer.



Once the .aia file has been created and downloaded, you can share this file with other app inventor users. They can copy the file to their machine and import it to their local App Inventor virtual servers.

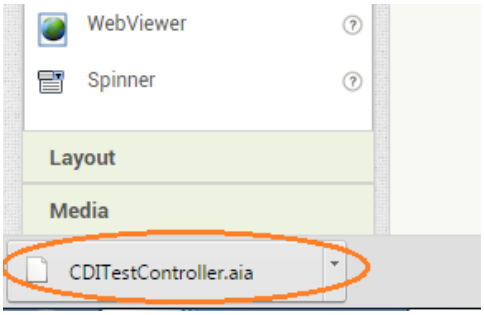


Figure 160 - The .aia file can be downloaded to your local computer.

To import a .aia file, simply choose **Projects->Import project (.aia)** from my computer… and then browse to find the file and import it.

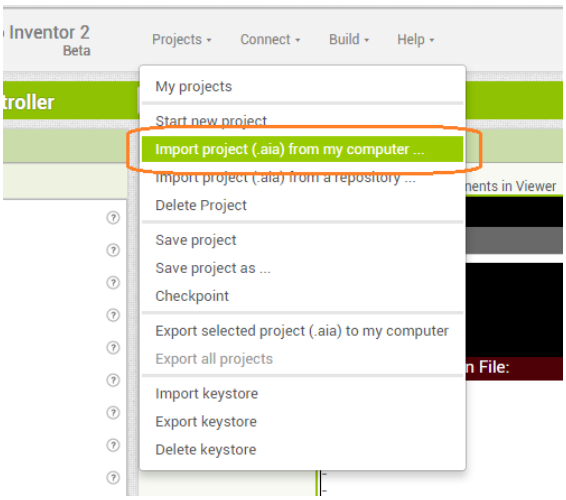


Figure 161 - Select Import project (.aia) from my computer...

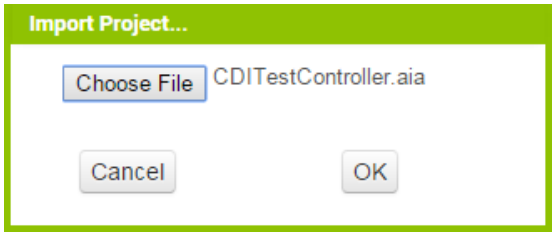


Figure 162 - Click on Choose file to browse for and select the .aia file. Hit OK to import the file.

Once you have successfully imported the file, the App Inventor screen should open up the project file in Designer mode.

11.2 Make Backup Copies of Your Project Files!!!

The ability to export an App Inventor project to an .aia file is convenient if you would like to share a project with a friend. However, this export feature is also a useful way to make backup copies of your work.

As you are working on an App Inventor project, you should consider periodically exporting your project to an .aia file to save a "snapshot" of the image in case you need a backup copy of the project later on. For example, after working on a project for a while, you can select **Projects->Export selected project (.aia) to my computer** from the menu to save a backup copy of your project.

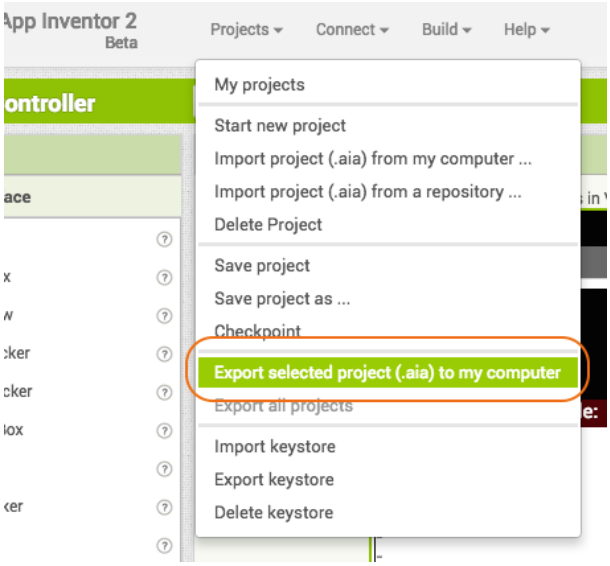


Figure 163 - Select Projects->Export selected project (.aia) to my computer.

The App Inventor will export the current project to an .aia file and it will download the file to your computer (typically to the "Downloads" folder if you are using Google Chrome as your browser).

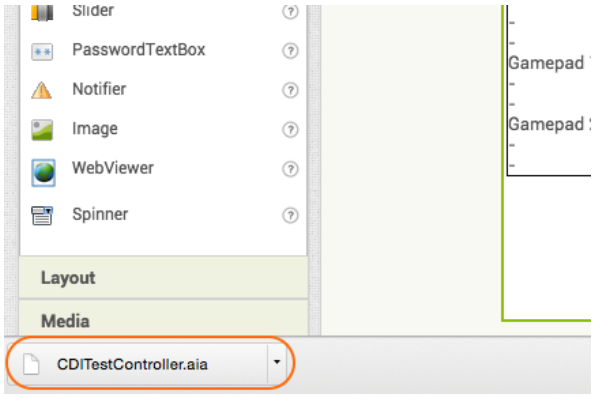


Figure 164 - App Inventor will export the project to a .aia and Download it to your computer.

You can copy or move the downloaded file ("CDITestController.aia" in Figure 164 above) to a folder for safekeeping. You can also rename the file and add a version number (such as renaming the file from Figure 164 to "CDITestController_v001.aia") so that you maintain different versions or different "snapshots" of your project. If you ever need to revert to an earlier version of the project, you can import the appropriate .aia file.

11.3 Troubleshooting Your App

If you encounter problems running your op modes with your App Inventor-generated app, here are some important troubleshooting tips to remember:

1. Make sure you only have a single Robot Controller type of app on your Android device.
2. Make sure that your Robot Controller app is the default app for all of the USB hardware modules on your robot and that it has permission to access the USB hardware.
3. Make sure the name of your configuration file matches exactly (Java is case sensitive) the name that you specified for the FtcRobotController component of your app. If the names do not match, the app will not be able to find the configuration file.
4. For each device (such as a DC motor, a servo, or a sensor) the DeviceName property must exactly match (Java is case sensitive) the name specified for the device in the configuration file.
5. Currently, when you want to add or remove a module from your robot configuration, you will have to redo the entire configuration file for the robot. For example, if you want to add a second motor controller to the configuration file, you will have to reconfigure the first motor controller and any other existing module, in addition to configuring the new motor controller.
6. Sometimes after you configure a Core Device Interface Module using the Configuration screen of your app, the app will not be able to find the CDI module after you have saved the configuration information to a file. If this happens try a Restart Robot to see if the app can find the device. If the app still cannot find the device, try disconnecting the USB cable for a few seconds, then reconnecting the cable and doing another Restart Robot to find the CDI module.
7. Don't forget, the project files on your virtual App Inventor server are associated with the e-mail address that you use to sign in to the App Inventor server. If you login to the server and do not see your project files, check to make sure that you used the proper e-mail address to identify yourself to the server.

11.4 Getting Help and Finding Additional Information

If you have questions about using the App Inventor locally on your computer to write apps for FTC, visit the App Inventor sub-forum of the FIRST Tech Challenge Technology forum:

http://ftcforum.usfirst.org/forumdisplay.php?160-MIT-App-Inventor

Also, there is a series of YouTube tutorial videos that demonstrate how to use the MIT App Inventor. If you visit YouTube and search for the official *FIRST* Tech Challenge videos, you will see several App Inventor tutorials available for viewing.

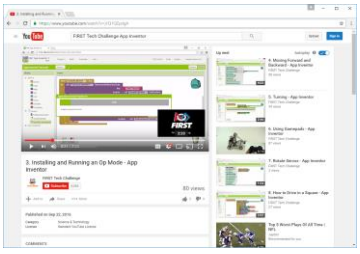


Figure 165 - The *FIRST* Tech Challenge YouTube account has several App Inventor tutorials available.

Appendix A Preparing Your ZTE Phone for Competition

In the US, the recommended Android device for the *FIRST* Tech Challenge (FTC) competition is the ZTE Speed. Before you use your ZTE Speed phone, there are steps you should take to make sure it's ready for competition.

Removing the SIM card from the ZTE Speed

The ZTE Speed is the recommended Android device for the FTC competition. Before you use your phone, you should physically remove the SIM card, and then place the phone into Airplane Mode (with WiFi still enabled) in order to make sure that the Speed phone does not try to connect to the Boost Mobile network whenever it is turned on.

Figure 166 - ZTE Speed

The first step in this process is to make sure that your phone is powered off. You will need to remove the plastic back cover of the phone. With the screen facing you, if you look in the lower right hand corner of the device, you will see a gap that you can use to pry the plastic rear cover off of the phone. Pry the back cover off of the phone and then place the phone screen down onto the table.

Along the right hand side of the phone, you should see the SIM card. Push in on the SIM card – this will eject the card partially from its slot. Remove the card and store it in a safe place. Replace the back cover and then power the phone on.

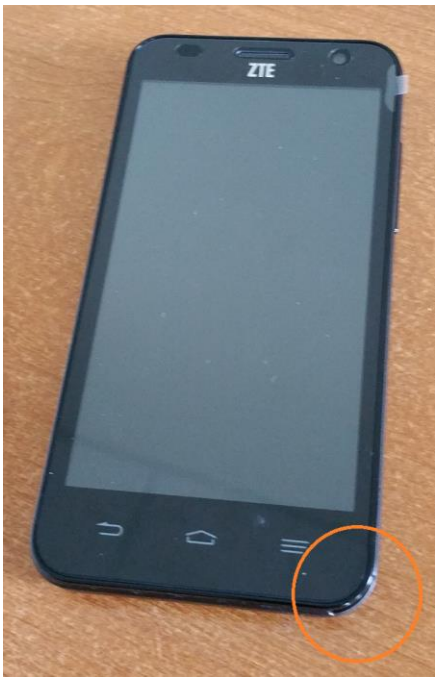


Figure 167 - There is a gap in lower right hand corner that you can use to pry the back cover off the phone.

Figure 168 - Pry the back cover off of the phone.

Figure 169 - The SIM card is installed in the right hand side.

Figure 170 - Push in on the SIM card to eject it from its slot.

Figure 171 - Remove the card from its slot and replace the back cover.

After you have removed the SIM card from the ZTE Speed phone, power the device on and step through the opening screens (see images below):

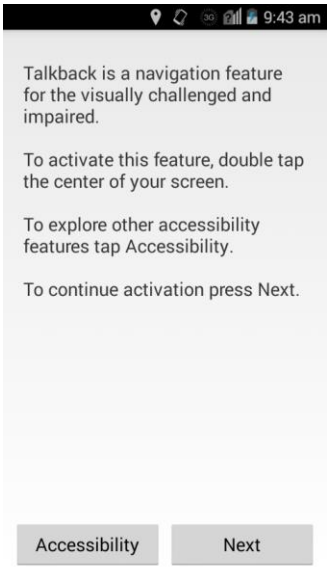


Figure 172 - The phone will ask you if you would like to set up Talkback. Hit Next to skip this step.

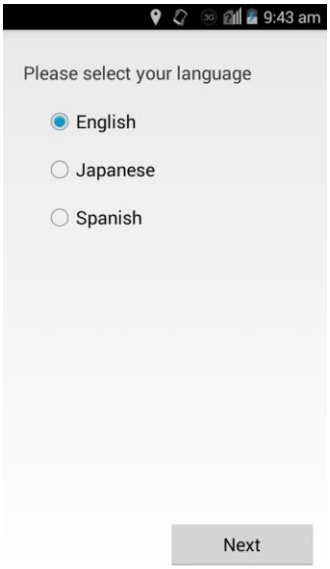


Figure 173 - Select your language and hit Next to continue.

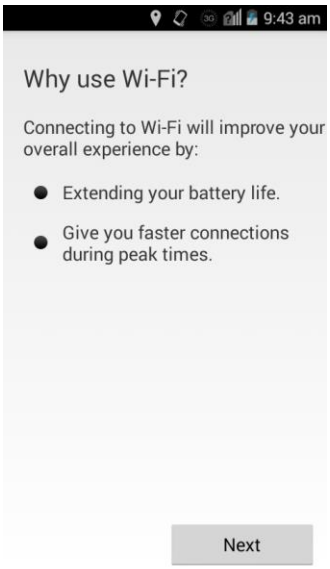


Figure 174 - The phone will prompt you to connect to a WiFi Network. Hit Next to acontinue.

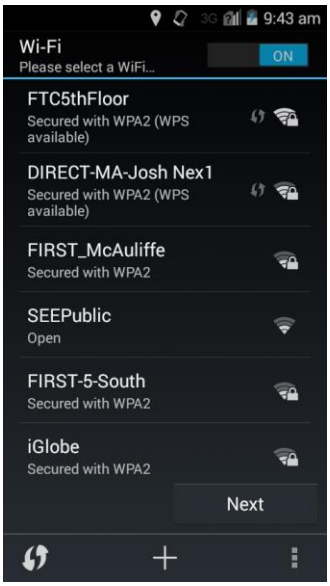


Figure 175 – You can connect to a WiFi network at this point or skip this step (hit Next).

Figure 176 - You might see this message indicating that the SIM card was not detected. Hit OK to continue.

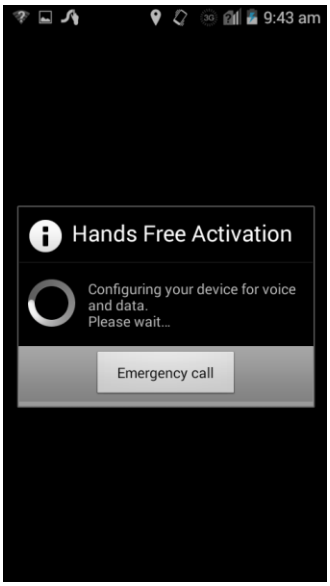


Figure 177 - The phone will display a Hands Free Activation screen.

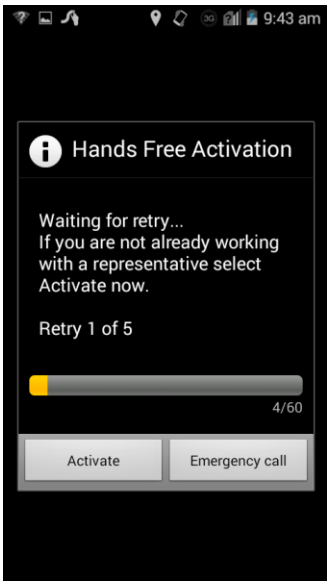


Figure 178 - The phone will try to activate itself. Hit the "Activate" button to continue.

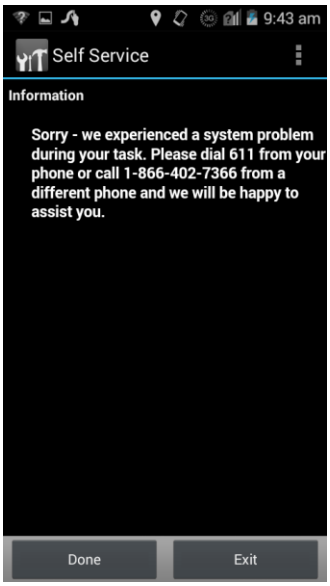


Figure 179 – The activation should fail. Hit “Done” to complete the process.

Figure 180 - You should now see your home screen. Press the Apps symbol to display the available apps on your phone.

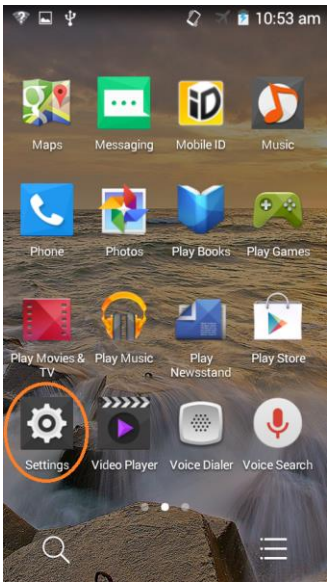


Figure 181 - Find the Settings app and press it to launch the Settings activity.

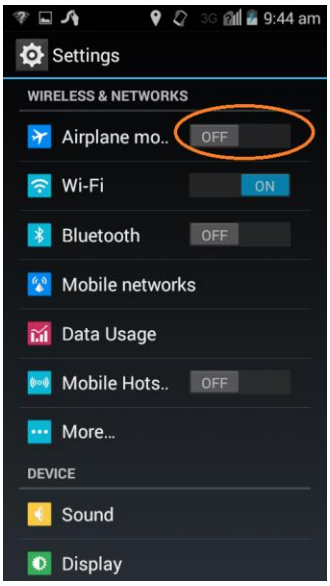


Figure 182 - Turn on Airplane Mode so the device won't try to connect to mobile network.

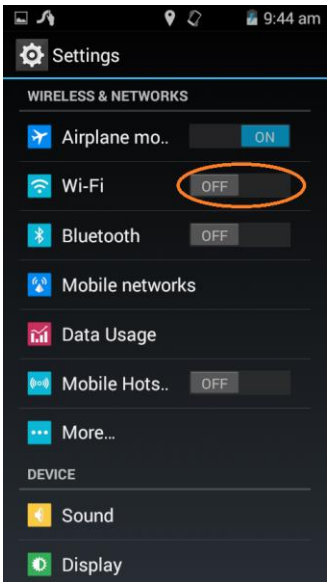


Figure 183 - Turn WiFi back on once the phone is in Airplane Mode.

Once you have removed the SIM card, switched the phone to Airplane Mode and turned WiFi back on, the phone is now ready for use.

Renaming the ZTE Speed

Before you compete, you also should rename your ZTE Speed devices (both the Robot Controller and Driver Station devices). Please consult the FTC Game Manual Part 1 (rule <RS02>) on the naming

convention that should be applied to your phones. In order to change the name on your device, launch the Android **Settings** menu again and click on the **Wi-Fi** item to launch the Wi-Fi Settings screen.

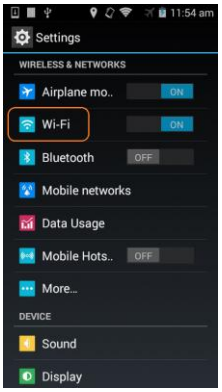


Figure 184 - Click on Wi-Fi to launch Wi-Fi Settings.

Click on the three dots in the lower right hand corner of the WiFi Settings window to display a pop-up menu.

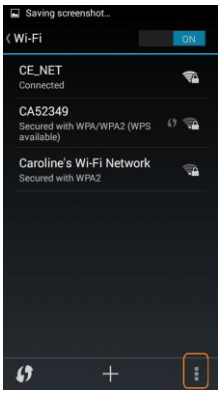


Figure 185 - Click on three dots to bring up pop-up menu.

Select the **Wi-Fi Direct** item from the pop-up menu.

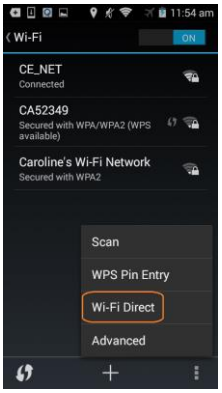


Figure 186 - Select Wi-Fi Direct.

Click on the **RENAME DEVICE** button in the Wi-Fi Direct settings screen.

Figure 187 - Click on the RENAME DEVICE button.

Specify the new name and click on **OK** to save the name.

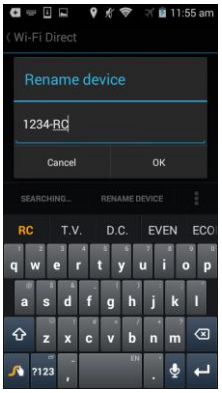


Figure 188 - Specify the new name and click on OK to save the name.

After you have changed your device's name, you might want to power cycle your Android device. Also, you will most likely need to re-pair the Driver Station device to the Robot Controller device after a name change.