FIRST FORWARD℠
PRESENTED BY Qualcomm

PRESENTED BY Raytheon Technologies

2021-2022 *FIRST*® Tech Challenge

OnBot Java Guide

FIRST TECH CHALLENGE

FIRSTINSPIRES.ORG/ROBOTICS/FTC

Sponsor Thank You

Thank you to our generous sponsors for your continued support of the *FIRST*® Tech Challenge!

***FIRST*® TECH CHALLENGE**
SEASON PRESENTING SPONSOR

***FIRST*® TECH CHALLENGE**
PROGRAM SPONSOR

***FIRST*® TECH CHALLENGE**
KEY SPONSOR

Revision History		
Revision	**Date**	**Description**
1	08/12/2021	Initial Release

Contents

Introduction

What is FIRST® Tech Challenge?

FIRST® Tech Challenge is a student-centered program that focuses on giving students a unique and stimulating experience. Each year, teams engage in a new game where they design, build, test, and program autonomous and driver operated robots that must perform a series of tasks. To learn more about *FIRST*® Tech Challenge and other *FIRST*® Programs, visit www.firstinspires.org.

Gracious Professionalism®

FIRST® uses this term to describe our programs' intent.

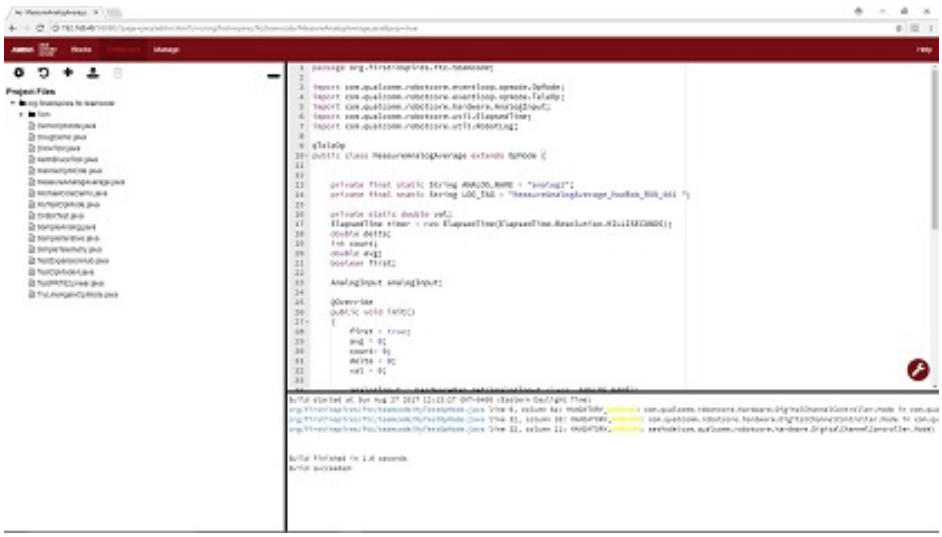
Gracious Professionalism® is a way of doing things that encourages high-quality work, emphasizes the value of others, and respects individuals and the community.

Watch Dr. Woodie Flowers explain *Gracious Professionalism* in this short video.

1. Introduction to OnBot Java Programming

This tutorial will take you step-by-step through the process of configuring, programming, and operating your Control System. This tutorial uses the OnBot Java Programming Tool to help you get started programming your robot.

The FTC OnBot Java Programming Tool is a text-based programming tool that lets programmers use a web browser to create, edit and save their Java op modes. This tool is recommended for programmers who have basic to advanced Java skills and who would like to write text-based op modes.



2. The FTC Control System

2.1. Introduction

The *FIRST* Tech Challenge seeks to inspire youth to become the next generation of STEM leaders and innovators through participation in mentor-guided robotics competition. Teams who participate in the *FIRST* Tech Challenge must build a robot that performs a variety of tasks. The tasks vary from season to season, and are based on a set of game rules that are published at the start of each season. The more tasks that a robot can complete, the more points a team will earn.

2.2. Autonomous Vs. Driver Controlled

A *FIRST* Tech Challenge match has an *autonomous* phase and a *driver-controlled* or *tele-operated* phase. In the autonomous phase of a match the robot operates without any human input or control. In the driver-controlled phase, the robot can receive input from up to two human drivers.

2.3. Point-to-Point Control System

The *FIRST* Tech Challenge uses Android devices to control its robots. During a competition, each team has two Android devices.

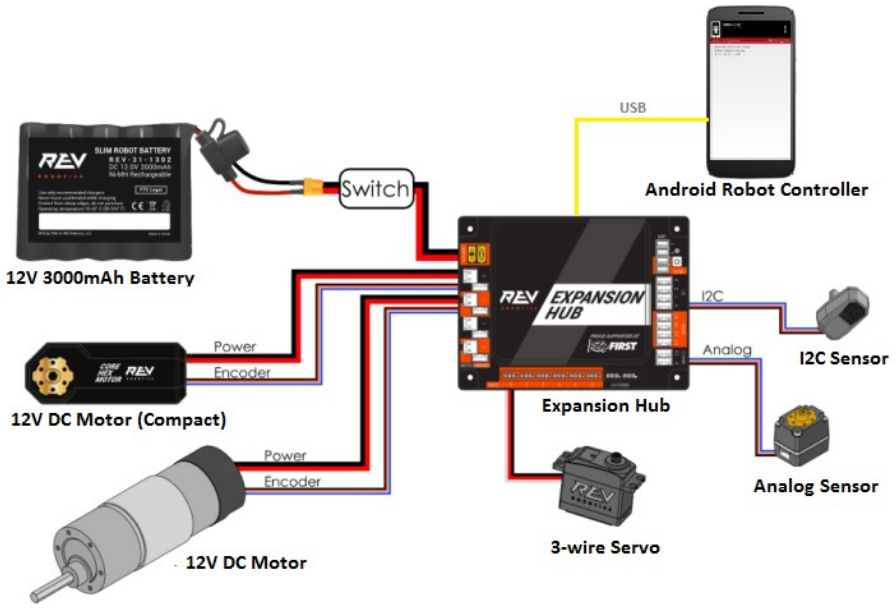
One Android device is mounted onto the robot and is called the *Robot Controller*. The Robot Controller acts as the “brains” of the robot. It does all of the thinking for the robot and tells the robot what to do. It consists of an Android device running an FTC Robot Controller app. There are two hardware options currently being used: REV Robotics Expansion Hub or the REV Robotics Control Hub.

A second Android device sits with the team drivers and has one or two gamepads connected. This second device is known as the *Driver Station*. The Driver Station is sort of like a remote control that you might use to control your television. The Driver Station allows a team to communicate remotely (using a secure, wireless connection) to the Robot Controller and to issue commands to the Robot Controller. The Driver Station consists of an Android device (smartphone or REV Driver Hub) running an FTC Driver Station app. The REV Driver Hub has the FTC Driver Station app pre-installed on the device. Note: You do not have to reinstall the app on the Hub once charged and connected to Wi-Fi.

2.4. REV Robotics Expansion Hub

The REV Robotics Expansion Hub is the electronic input/output (or “I/O”) module that lets the Robot Controller talk to the robot’s motors, servos, and sensors. The Robot Controller communicates with the Expansion Hub through a serial connection. For the situation where an Android smartphone is used as the Robot Controller, a USB cable is used to establish the serial connection. For the situation where a REV Robotics Control Hub is used, an internal serial connection exists between the built-in Android device and the Expansion Hub.

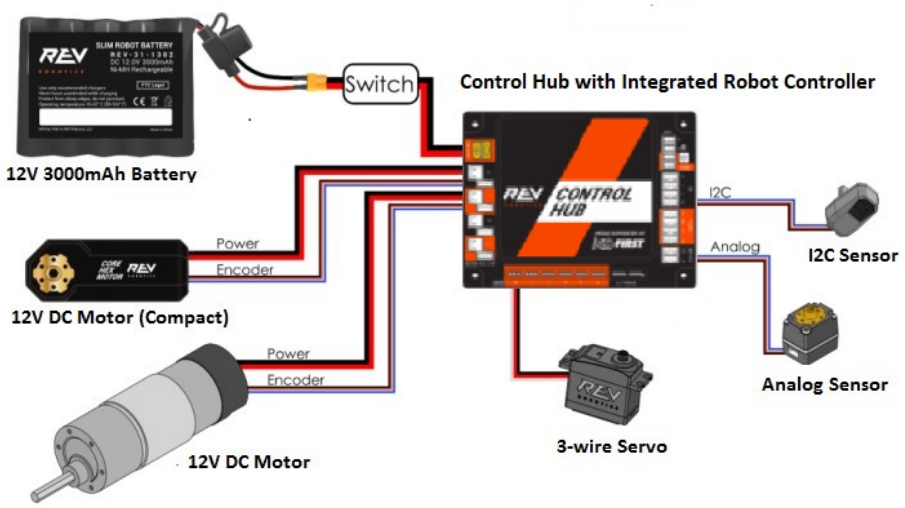
The Expansion Hub is also connected to a 12V battery which is used to power the Expansion Hub, the motors, the servos and sensors. If an Android smartphone is used as the Robot Controller, then the smartphone will have its own independent battery. If a REV Robotics Control Hub is used as the Robot Controller, then the Control Hub will use the main 12V battery to power its internal Android device.



2.5. REV Robotics Control Hub

The Control Hub has an integrated version of the Robot Controller. It combines an Android device built into the same case as a REV Robotics Expansion Hub.

The Control Hub, which has its built-in Android device connected directly to the Expansion Hub using an internal serial bus, eliminates the need for an external USB connection between the Android Robot Controller and the I/O module.



2.6. What's an Op Mode?

During a typical *FIRST* Tech Challenge match, a team's robot has to perform a variety of tasks in an effort to score points. For example, a team might want their robot to follow a white line on the competition floor and then score a game element (such as a ball) into a goal autonomously during a match. Teams write "op modes" (which stand for "operational modes") to specify the behavior for their robot.

Op modes are computer programs that are used to customize the behavior of a competition robot. The Robot Controller can *execute* a selected op mode to perform certain tasks during a match.

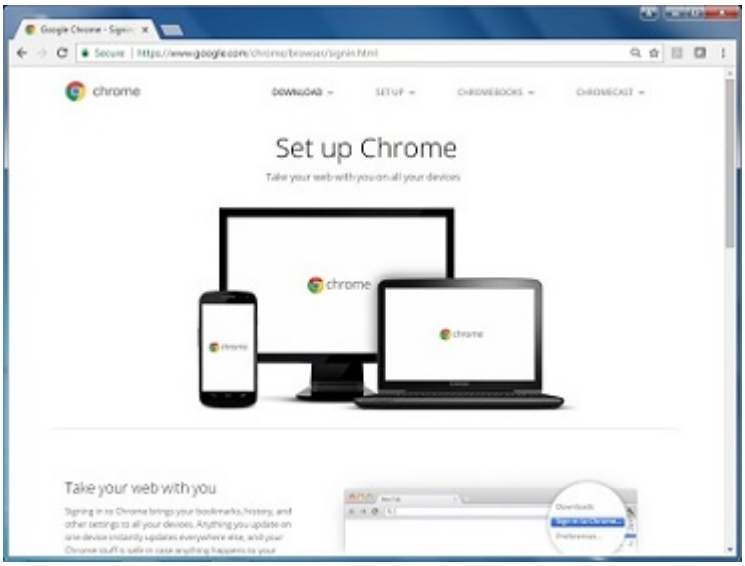
Teams who are participating in the *FIRST* Tech Challenge have a variety of programming tools that they can use to create their own op modes. Teams can use a visual ("drag and drop") programming tool called the *FTC Blocks Programming Tool* to create their op modes. Teams can also use a text-based Java tool known as the *FTC OnBot Java Programming Tool* or Google's *Android Studio* integrated development environment (also known as an "IDE") to create their op modes.

3. Required Materials

3.1. Required Materials List

This guide contains tutorials that demonstrate how to configure, program, and operate the FTC control system. In order to complete the tutorials, you will need to have the following materials available:

Required Item(s)	Image
Two (2) *FIRST*-approved* Android devices (A Robot Controller smartphone and a Driver Station smartphone or Driver Hub). Or... One (1) Control Hub and one (1) *FIRST*-approved* Android smartphone. ***For a list of *FIRST*-approved Android smartphones, refer to the current FTC Game Manual Part 1, rule <RE06>.**	 Or... 
Wireless Internet access.	

Required Item(s)	Image
Laptop with Microsoft Windows 7, 8 or 10 and Wi-Fi capability. **Note:** that your laptop should have the most current service packs and system updates from Microsoft. If you are using a different type of machine (such as a Chromebook, Android Tablet, etc.) as your programming device, the steps might differ slightly on how to access the Programming Server on the Robot Controller. Refer to your device's user documentation for details on how to connect to a Wi-Fi network.	
Javascript-enabled web browser (Google Chrome is the recommended browser).	
If you are using a smartphone as your Robot Controller, you will also need a REV Robotics Expansion Hub (REV-31-1153) to connect to the motors, servos, and sensors. Control Hub users will use the integrated Expansion Hub.	

Required Item(s)	Image
REV Robotics Switch, Cable, & Bracket (REV-31-1387).	
If you are using an approved 12V battery that has an Tamiya connector (like the Tetrix W39057 battery) you will need a >REV Robotics Tamiya to XT30 Adapter Cable (REV-31-1382). If you have a REV Robotics Slim Battery (REV-31-1302) then you will not need this adapter since the REV battery already has an XT30 connector.	
FIRST-approved* 12V Battery (such as Tetrix W39057 or REV Robotics REV-31-1302). ***For a list of *FIRST*-approved 12V batteries, refer to the current FTC Game Manual Part 1, rule <RE03>.**	 Or...

Required Item(s)	Image
FIRST-approved* 12V Battery (such as Tetrix W39057 or REV Robotics REV-31-1302). *For a list of ***FIRST***-approved 12V batteries, refer to the current FTC Game Manual Part 1, rule <RE03>.	
FIRST-approved* 12V DC Motor (such as Tetrix W39530, with power cable W41352). *For a list of ***FIRST***-approved 12V motors, refer to the current FTC Game Manual Part 1, rule <RE09>.	
REV Robotics Anderson to JST VH Cable (REV-31-1381).	

Required Item(s)	Image
180-Degree Standard Scale Servo (such as Hitec HS-485HB).	
REV Robotics Color Sensor with 4-Pin Cable (REV-31-1154).	
REV Robotics Touch Sensor with 4-Pin Cable (REV-31-1425).	
If you are using a smartphone as your Robot Controller, you will need a USB Type A male to type mini-B male cable. Control Hub users do not need this cable.	

Required Item(s)	Image
If you are using a smartphone as your Robot Controller, you will need two (2) micro USB OTG adapters. If you are using a Control Hub as your Robot Controller, you will need one (1) micro USB OTG adapter.	
Logitech F310 USB Gamepad.	

4. Using Your Android Device

Before you get started with your control system, it is helpful if you familiarize yourself with the basic operation of your Android device.

4.1. Unlocking Your Screen

When you first power on an Android phone, it usually starts off with the screen in a "locked" state. For the Motorola smartphones that are used in the *FIRST* Tech Challenge, you must touch the locked screen and then slide your finger upwards along the screen to unlock the phone. Note that different devices might require a slightly different procedure to unlock the screen.

Depending on your security settings, you might be challenged for a pass code or PIN number. Use the touch screen to enter in the pass code or PIN value and tap on the check mark to log into the device.

4.2. ***Navigating in Android***

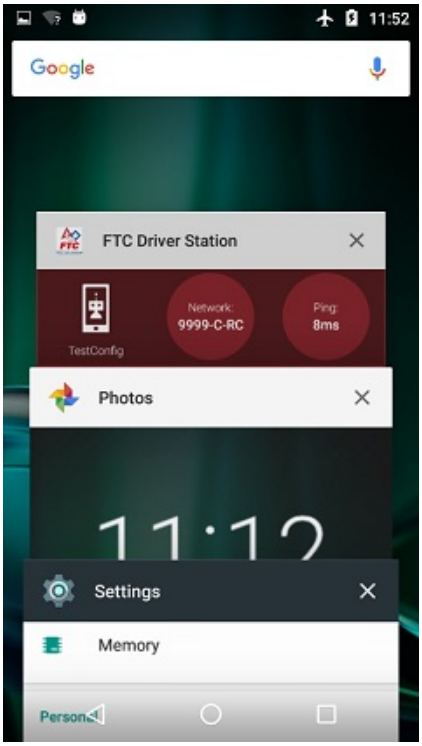
Your phone should display its home screen if you just powered it on and unlocked it. Note that the actual screens on your smartphone might differ slightly from the screens depicted in this tutorial.

At the bottom of the screen there should be some buttons that you can use to navigate the screens on your Android device.

The leftmost button (see image above) is the "Back" button. You can use this button to return to the previous screen on your Android device.

The center button is the "Home" button. Pressing this button should take you back to the home or opening screen of your Android device.

The rightmost button is the "Recent Apps" button. If you click on this button it will display the apps that were recently run and are dormant in the background. You can close a recent app by tapping the "X" button on the app's listing.



Note: Some Android smartphones have an auto-hide feature which automatically hides the bottom navigation buttons. If your smartphone has this feature, you might need to swipe up from the bottom of the screen to display the navigation buttons.

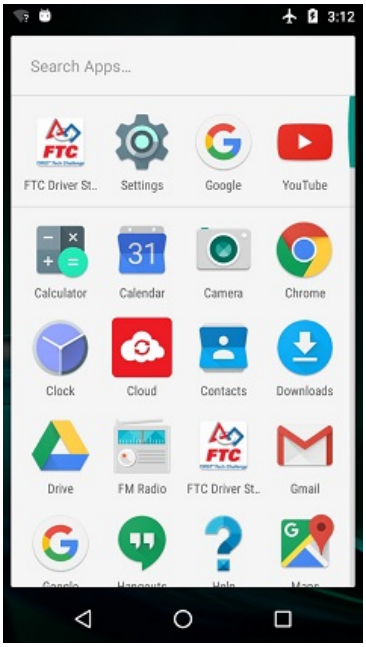
5. Displaying Available Apps on your Android Phone

5.1. Android Marshmallow Users

If you are using a device with Android Marshmallow (6.x) or earlier, you can display the available apps using the *Android App Drawer* button that is available on the home screen.

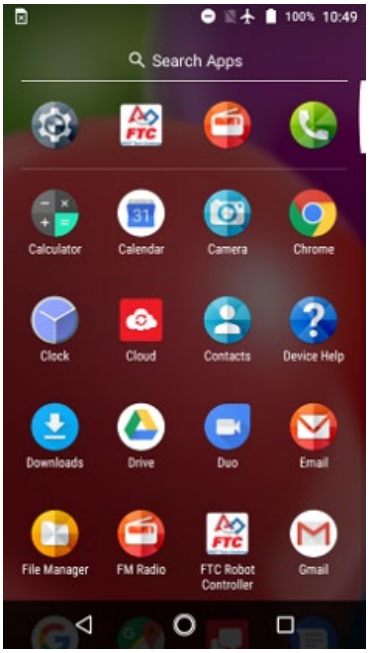
There should be another row of buttons visible above the "Back", "Home" and "Recent Apps" buttons. In the center of this row of buttons is a button that has an array of dots or squares.

Tapping on this button will launch the *Android App Drawer*. The App Drawer displays a list of all of the apps that are available on your Android device. You can scroll through the App Drawer screens to find and launch an app.



5.2. ***Android Nougat Users***

If you are using a device with Android Nougat (7.x) or newer, you can display the available apps by simply swiping upwards from the bottom of the touchscreen. Newer versions of Android no longer have the *App Drawer* feature.



6. Configuring Your Android Devices

6.1. *What Needs to Be Configured for My Control System?*

6.1.1. **Control Hub Users**

Teams who are using a Control Hub with the integrated Robot Controller will only need to configure a single Android device for use as a Driver Station. The process is as follows:

- Rename the smartphone to "<TEAM NUMBER>-DS" (where <TEAM NUMBER> is replaced by your team number).
- Install the Driver Station app onto the Driver Station phone. Note: the REV Driver Hub has pre-installed software.
- Put your phone into Airplane Mode (with the WiFi radio still on).
- Pair (i.e., wirelessly connect) the Driver Station to the Control Hub.

IMPORTANT NOTE: Eventually the Control Hub will need be renamed so that its name complies with Game Manual rule< RS01>, but for now we will use the Control Hub with its default name. You can learn how to manage a Control Hub (and modify its name, password, etc.) in this tutorial.

6.1.2. **Users with Two Android Smartphones**

Teams who have two smartphones and are not using a Control Hub will need to configure one smartphone for use as a Robot Controller and a second smartphone for use as a Driver Station. The process is as follows,

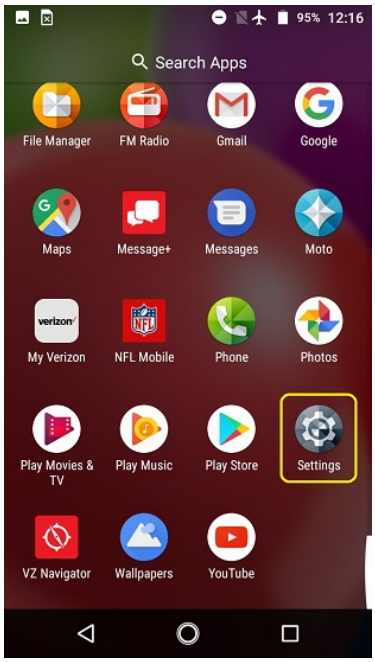
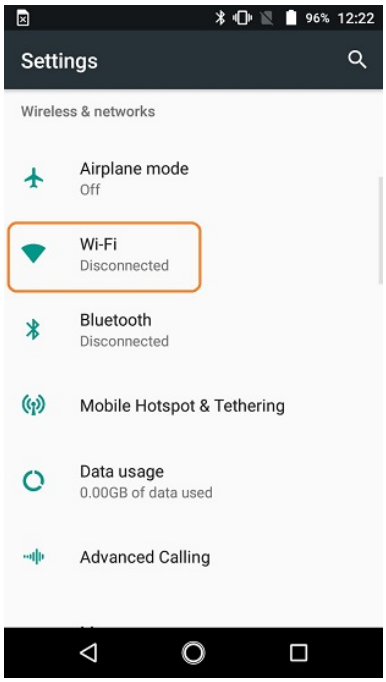
- Rename one smartphone to "<TEAM NUMBER>-RC" (replace <TEAM NUMBER> with your team number).
- Install the Robot Controller app onto the Robot Controller phone.
- Rename a second smartphone to "<TEAM NUMBER>-DS" (where <TEAM NUMBER> is replaced by your team number).
- Install the Driver Station app onto the Driver Station phone.
- Put your phones into Airplane Mode (with the WiFi radios still on).
- Pair (i.e., wirelessly connect) the Driver Station to the Robot Controller.

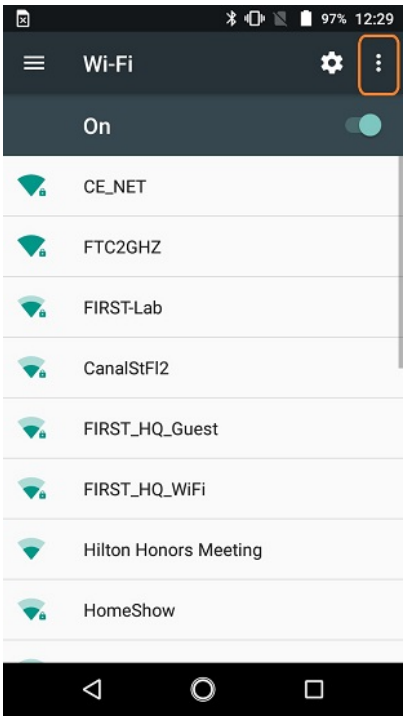
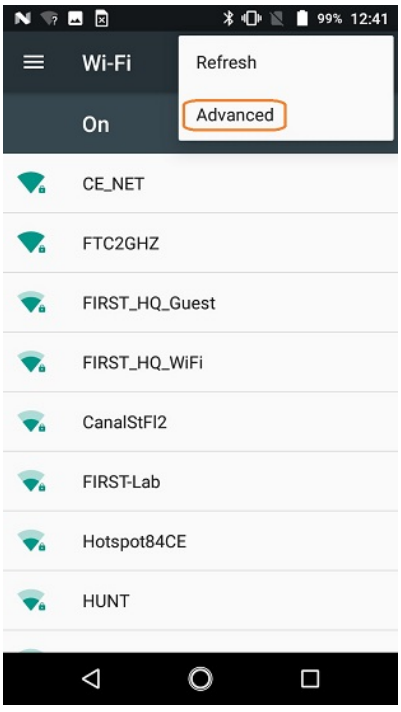
6.2. *Renaming Your Smartphones*

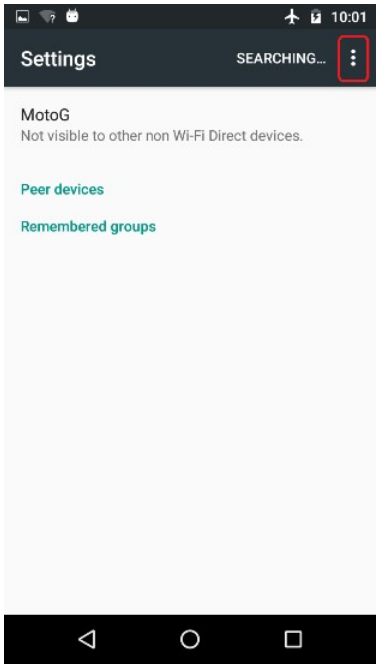
The official rules of the FIRST Tech Challenge (see <RS01>) require that you change the Wi-Fi name of your smartphones to include your team number and “-RC” if the phone is a Robot Controller or “-DS” if it is a Driver Station. A team can insert an additional dash and a letter (“A”, “B”, “C”, etc.) if the team has more than one set of Android phones.

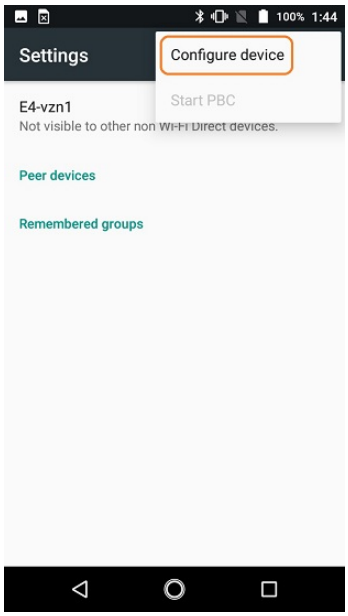
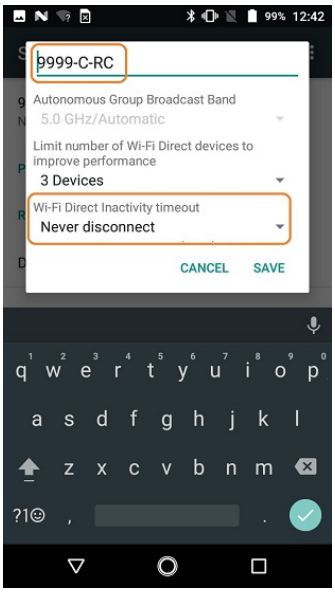
If, for example, a team has a team number of 9999 and the team has multiple sets of phones, the team might decide to name one phone “9999-C-RC” for the Robot Controller and the other phone “9999-C-DS” for the Driver Station. The “-C” indicates that these devices belong to the third set of phones for this team.

> **NOTE**: it will take an estimated 5 minutes per phone to complete this task.

Step	Image
1. Browse the list of available apps on the smartphone and locate the **Settings** icon. Click on **Settings** icon to display the Settings screen.	
2. Click on **Wi-Fi** to launch the Wi-Fi screen.	

Step	Image
3. Touch the three vertical dots to display a pop-up menu.	
4. Select **Advanced** from the pop-up menu.	

Step	Image
5. Select **Wi-Fi Direct** from the **Advanced Wi-Fi** screen.	
6. Touch the three vertical dots to display a pop-up menu.	

Step	Image
7. Select **Configure Device** from the pop-up menu.	
8. Use touch pad to enter new name of device. • If the device will be a Robot Controller, specify your team number and "-RC". • If the device will be a Driver Station, specify your team number and "-DS". You can also set the Wi-Fi Direct inactivity timeout to "Never disconnect" and then hit the **SAVE** button to save your changes. • **NOTE:** In the screenshot shown to the right, the team number is “9999”. The “-C” indicates that this is from the third pair of smartphones for this team. The “-RC” indicates that this phone will be a Robot Controller.	
9. After renaming phone, power cycle the device.	

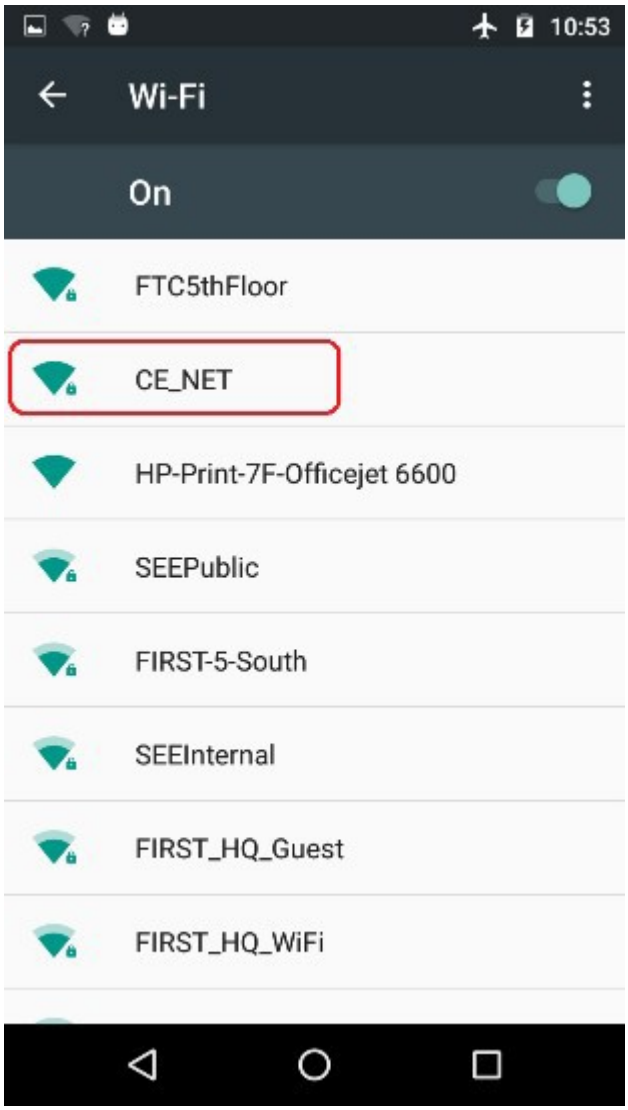
6.3 Installing the FTC Apps

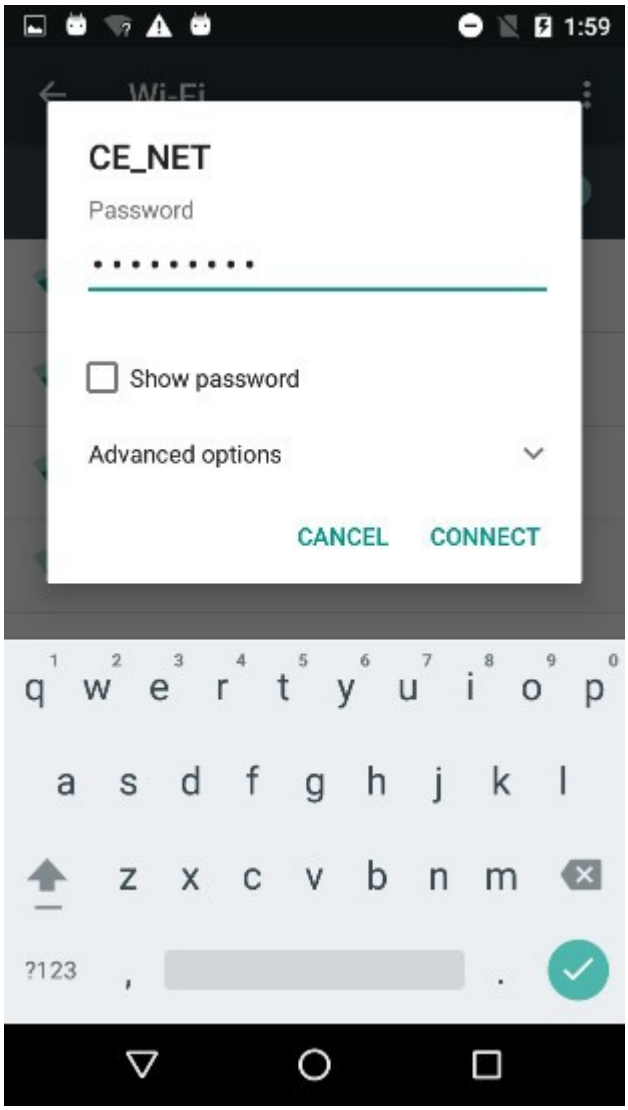
As of 2021, the FTC apps (v 6.1 and higher) are not available on Google Play. The REV Hardware Client software will allow you to download the apps to FTC devices: REV Control Hub, REV Expansion Hub, REV Driver Hub, FTC-approved android devices. Here are some of the benefits:

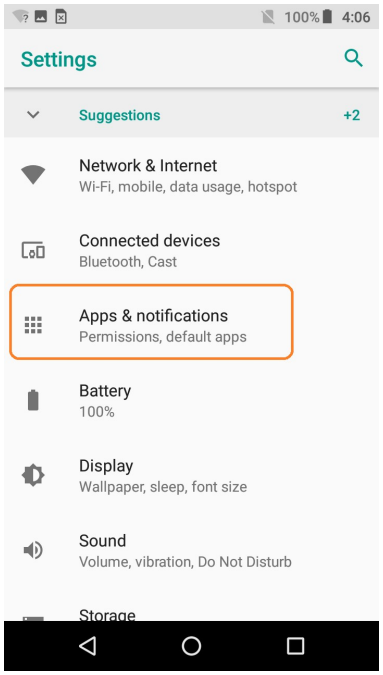
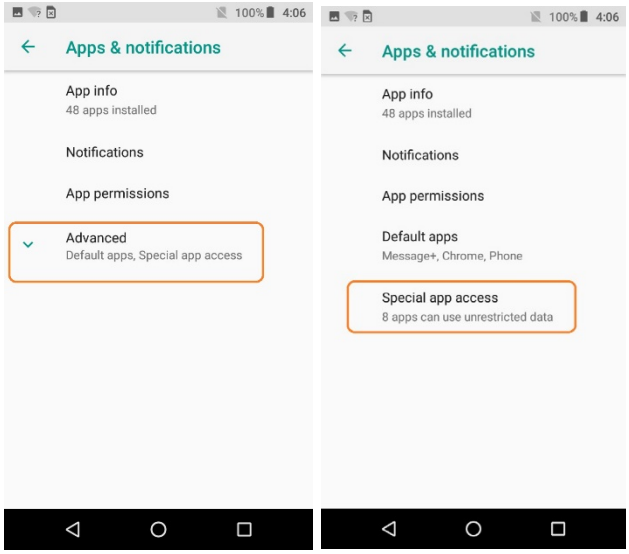
- Connect a REV Control Hub via WiFi.
- One Click update of all software on connected devices.
- Pre-download software updates without a connected device.
- Back up and restore user data from Control Hub.
- Install and switch between DS and RC applications on Android Devices.
- Access the Robot Control Console on the Control Hub.

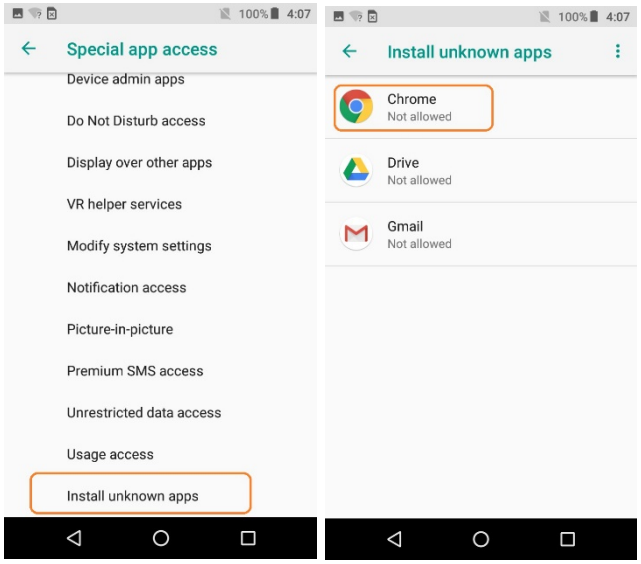
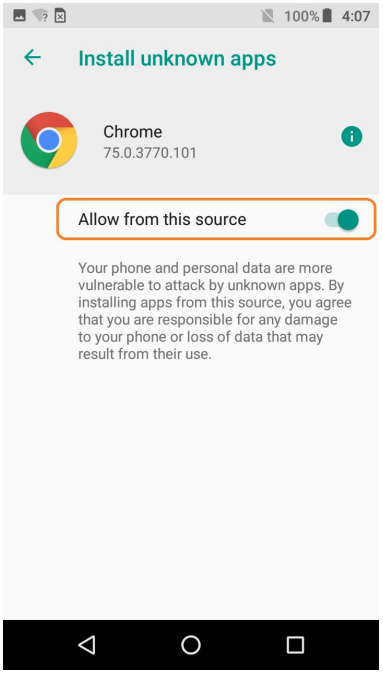
The FTC app releases are available on the FTCRobotController Github.

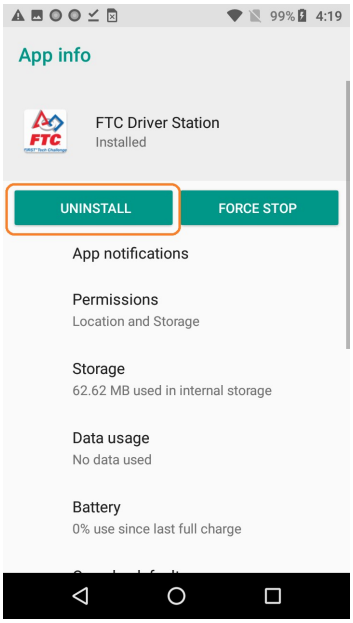
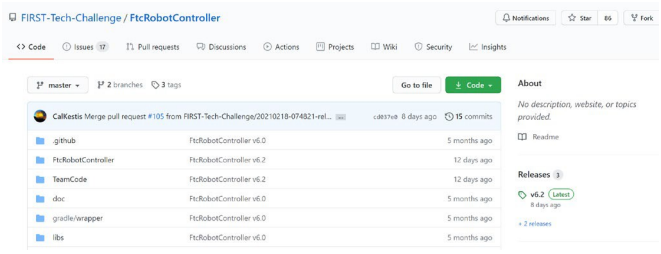
NOTE: it will take an estimated 7.5 minutes per device to complete this task.

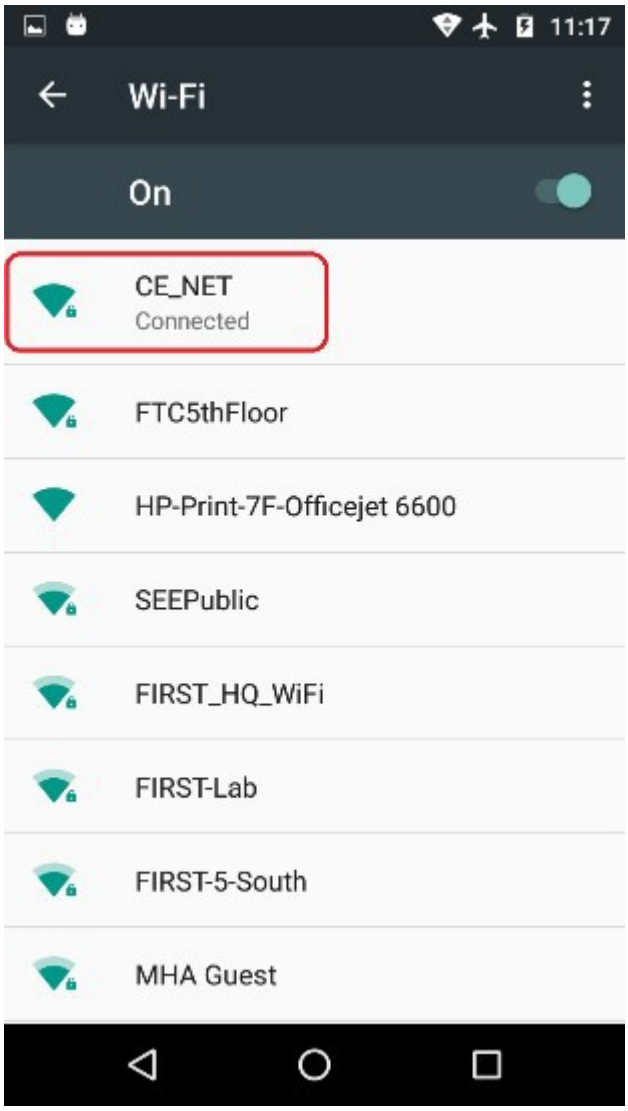
Step	Image
1. From the Android Wi-Fi screen look for the name of your wireless network (“CE_NET” in this example) and touch the wireless network name to log into the network.	

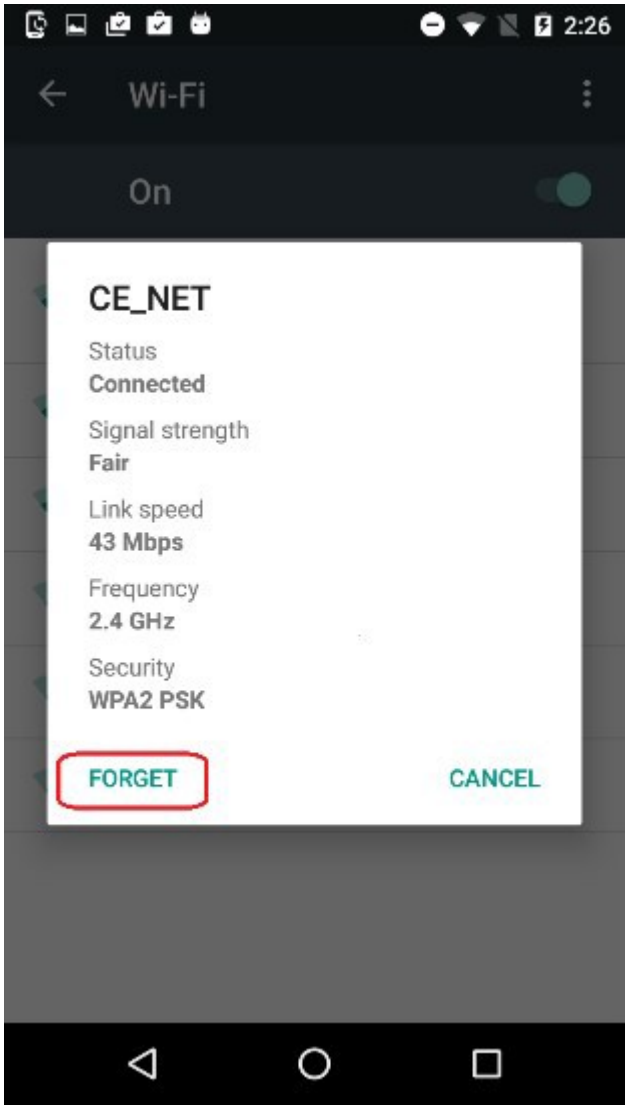
Step	Image
2. Specify the password using the touch keypad and press **CONNECT** to connect to this wireless network.	

Step	Image
3. Grant Google Chrome permission to ***install from unknown sources:*** a. Go to Settings, then Apps & notifications.	
b. Select Advanced, then Special app access.	

Step	Image
c. Install unknown apps, then select Chrome.	
d. Allow access for Google Chrome	

Step	Image
4. Uninstall the old application.	
5. Download the desired FTC app from the FtcRobotController GitHub repository	

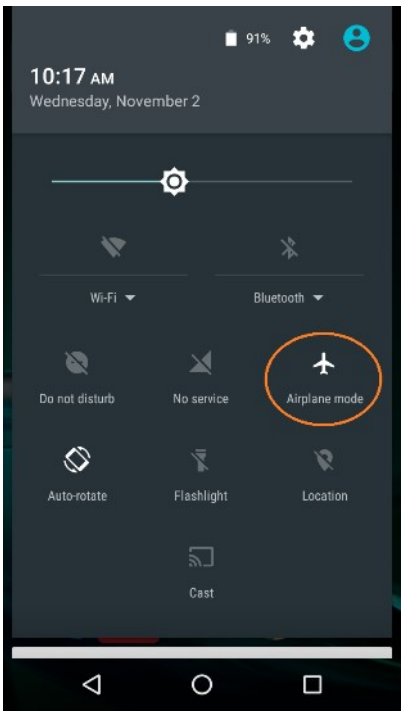
Step	Image
6. After you have successfully installed the app, you should forget the external wireless network on your phone. 7. Go to the Android Wi-Fi screen, find the name of the currently connected network, and tap on the network name to bring up a pop-up box with info about the network.	

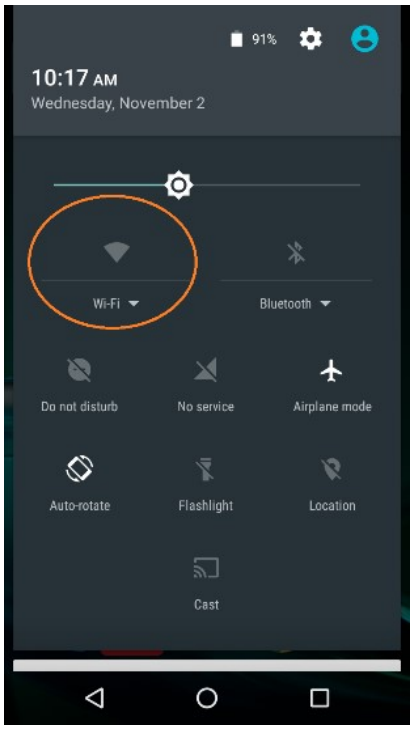
Step	Image
8. Click on the **FORGET** button to forget the wireless network.	

6.3. *Placing Phones into Airplane Mode with Wi-Fi On*

For the *FIRST* Tech Challenge competitions, it is important that you place your Robot Controller and Driver Station phones into Airplane mode but keep their Wi-Fi radios turned on. This is important because you do not want any of the cellular telephone functions to be enabled during a match. The cellular telephone functions could disrupt the function of the robot during a match.

> **NOTE:** it will take an estimated 2.5 minutes per phone to complete this task. Also note that the screens displayed on your Android devices might differ slightly from the images contained in this wiki.

Step	Image
1. On the main Android screen of each smartphone, use your finger to slide from the top of the screen down towards the bottom of the screen to display the quick configuration screen. **Note**: that for some smartphones you might have to swipe down more than once to display the quick configuration screen, particularly if there are messages or notifications displayed at the top of your screen. Look for the Airplane mode icon (which is shaped like an airplane) and if the icon is not activated, touch the icon to put the phone into airplane mode.	

Step	Image
2. Placing the phone into airplane mode will turn off the Wi-Fi radio. If the Wi-Fi icon has a diagonal line through it (see Step 1 above), then the Wi-Fi radio is disabled. You will need to touch the **Wi-Fi** icon on the quick configuration screen to turn the Wi-Fi radio back on.	

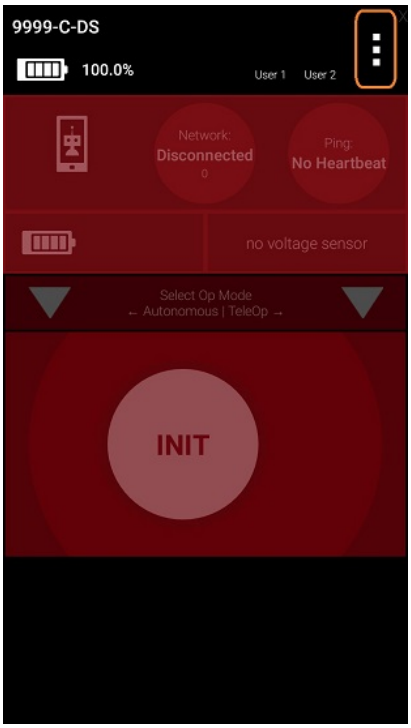
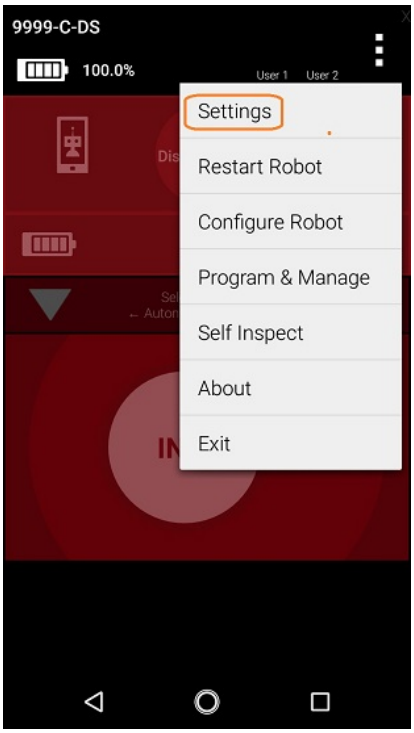
7. Pairing the Driver Station to the Robot Controller

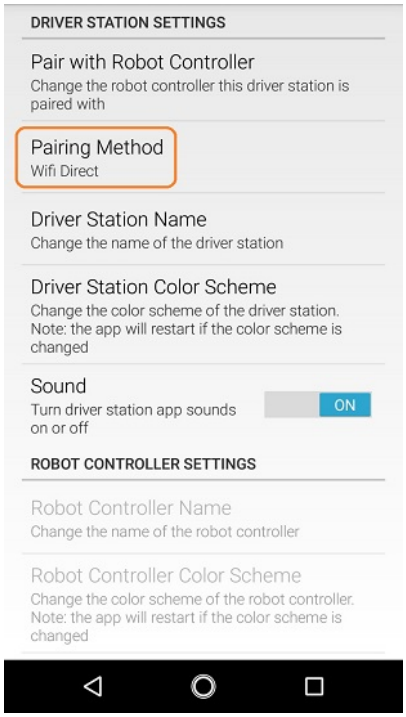
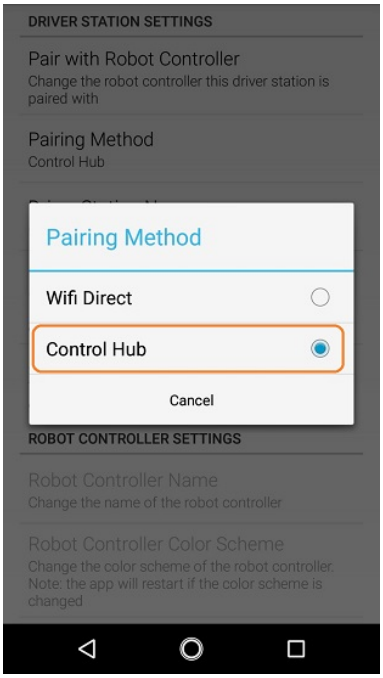
7.1. Control Hub Users

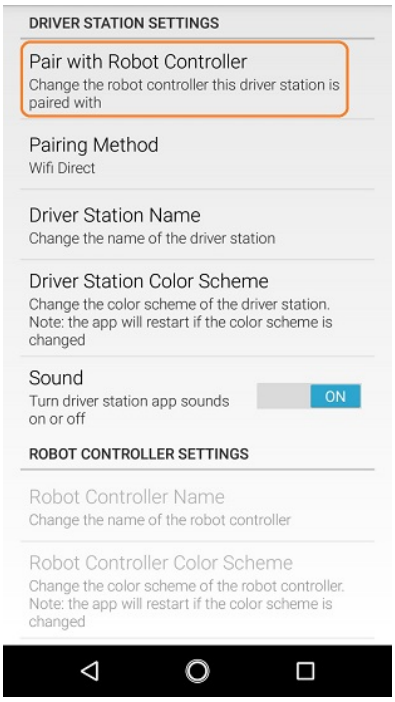
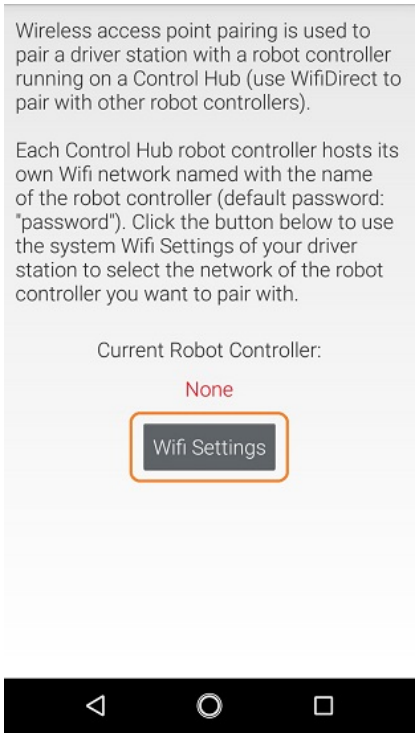
The REV Robotics Control Hub should come with the Robot Controller app pre-installed. The REV Robotics Driver Hub should also come with the Driver Station app pre-installed. If you are using a smartphone have successfully installed the FTC Driver Station on an Android phone, you will want to establish a secure wireless connection between the Control Hub and the Driver Station. This connection will allow your Driver Station phone to select op modes on your Robot Controller and send gamepad input to these programs. Likewise, it will allow your op modes running on your Robot Controller to send telemetry data to your Driver Station where it can be displayed for your drivers. The process to connect the two devices is known as “pairing.”

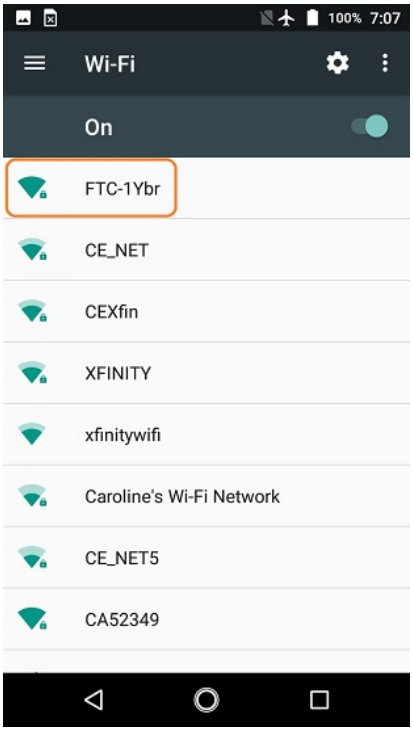
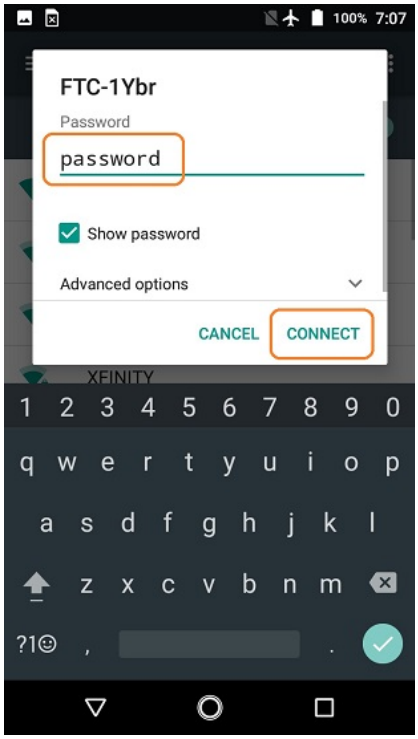
> **NOTE:** the Control Hub does not have its own internal battery. Before you can connect a Driver Station to the Control Hub, you must connect the Control Hub to a 12V battery.
>
> **Also note**: that it will take an estimated 10 minutes to complete this task.

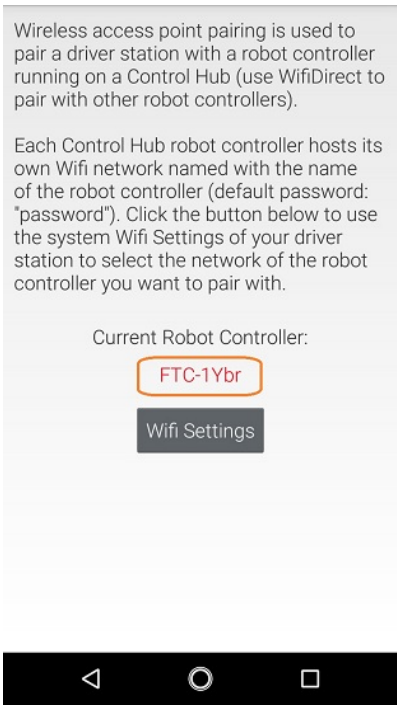
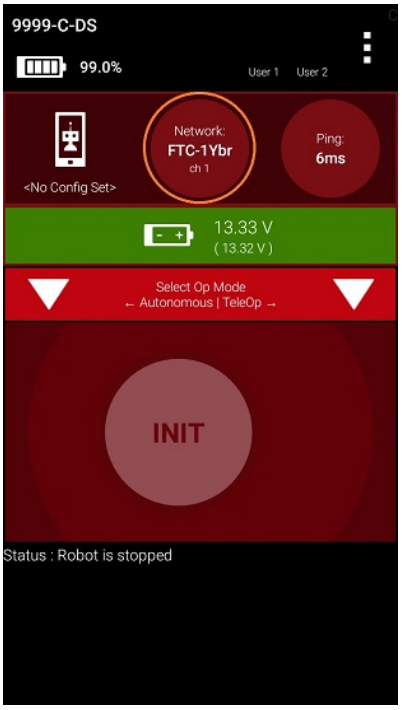
Step	Image
1. Connect an approved 12V battery to the power switch (REV-31-1387) and make sure the switch is in the off position. Connect the switch to an XT30 port on the Control Hub and turn the switch on. The LED should initially be blue on the Control Hub.	
2. It takes approximately 18 seconds for the Control Hub to power on. The Control Hub is ready to pair with the Driver Station when the LED turns green. **Note:** the light blinks blue every ~5 seconds to indicate that the Control Hub is healthy.	
Follow the instructions for Downloading the FTC Apps. This app is pre-installed on the Driver Hub and will appear after the Wi-Fi password has been updated. Note that the first time you launch the app your Android device might prompt you for permissions that the app will need to run properly. Whenever prompted, press **Allow** to grant the requested permission.	

Step	Image
4. Touch the three vertical dots on the upper right-hand corner of the main screen of the FTC Driver Station app. This will launch a pop-up menu.	
5. Select **Settings** from the pop-up menu.	

Step	Image
6. From the **Settings** screen, select **Pairing Method.**	
7. Press **Control Hub.**	

Step	Image
8. From the **Settings** screen, select **Pair with Robot Controller**.	
9. From **Pair with Robot Controller** screen, press the **Wifi Settings.**	

Step	Image
10. Select **Wi-Fi**, then select the correct network. If this is the first time you are connecting to the Control Hub, then the default network name should begin with the prefix "FTC-" ("FTC-1Ybr" in this example). The default network name should be listed on a sticker attached to the bottom side of the Control Hub.	
11. When prompted, specify the password for the Control Hub's WiFi network and press **Connect** to connect to the Hub. **Note:** that the default password for the Control Hub network is "password". Also note that when you connect to the Control Hub's WiFi network successfully, the Driver Station will not have access to the Internet.	

Step	Image
12. After you successfully connected to the Hub, use the back arrow to navigate to the previous screen. You should see the name of the WiFi network listed under "Current Robot Controller:". Use the back-arrow key to return to the Settings screen. Then press the back-arrow key one more time to return to the main Driver Station screen.	
13. Verify that the Driver Station screen has changed and that it now indicates that it is connected to the Control Hub. The name of the Control Hub's WiFi network ("FTC-1Ybr" in this example) should be displayed in the Network field on the Driver Station.	

7.2. *Users with Two Android Smartphones*

> **Important Note:** If your Driver Station was previously paired to a Control Hub, and you currently would like to connect to an Android smartphone Robot Controller, then before attempting to pair to the Robot Controller, you should forget the Wi-Fi network for the previous Control Hub (using the Android Wifi Settings screen on the Driver Station) and then power cycle the Driver Station phone. If the previous Control Hub is powered on and if you haven't forgotten this network, then the Driver Station might try and connect to the Control Hub and might be unable to connect to the Robot Controller smartphone.

Once you have successfully installed the FTC apps onto your Android phones, you will want to establish a secure wireless connection between the two devices. This connection will allow your Driver Station phone to select op modes on your Robot Controller phone and send gamepad input to these programs. Likewise, it will allow your op modes running on your Robot Controller phone to send telemetry data to your Driver Station phone where it can be displayed for your drivers. The process to connect the two phones is known as “pairing.”

> **Note**: that it will take an estimated 10 minutes to complete this task.

8. Connecting Devices to a Control or Expansion Hub

This section explains how to connect a motor, a servo, and some sensors to your REV Robotics Control Hub or REV Robotics Expansion Hub. While the Control Hub differs from the Expansion Hub because of its built in Android device, the layout of the external motor, servo, and sensor ports are identical for the Control Hub and Expansion Hub.

The images in this section use an Expansion Hub to demonstrate how to connect the devices. The process, however, is identical for a Control Hub.

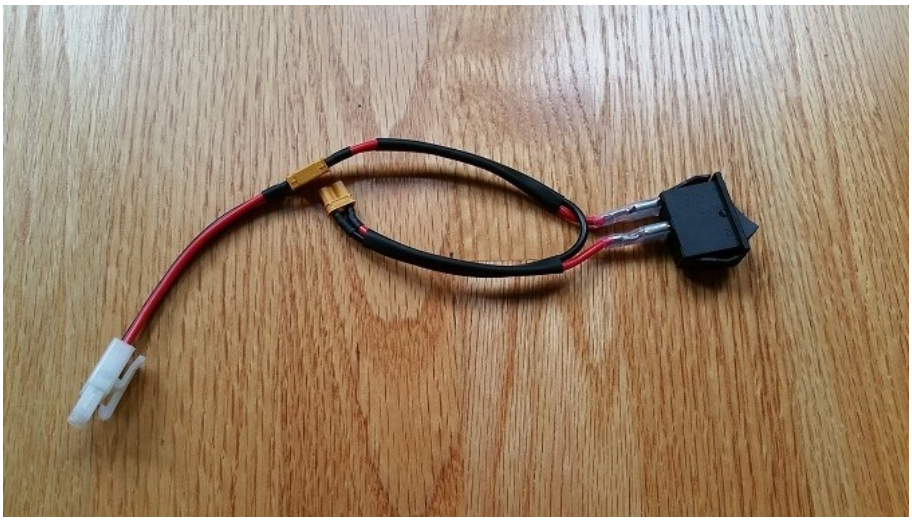
When the instructions in this section use the word "Hub", they are referring to a Control Hub or Expansion Hub.

8.1. *Connecting 12V Power to the Hub*

The Hub draws power from a 12V rechargeable battery. For safety reasons, the battery has a 20A fuse built in. A mechanical switch is used to turn on/turn off the power.

Note that it will take an estimated 5 minutes to complete this task.

1. If your 12V battery has a Tamiya style connector, connect the Tamiya to XT30 adapter cable to the matching end of the switch cable.



Important Note: Do not connect the 12V battery to the Tamiya adapter yet. We will connect the battery during a later step.

2. Connect the other end of the switch cable to a matching XT30 port on the Hub.

3. Verify that the switch is in the OFF position.

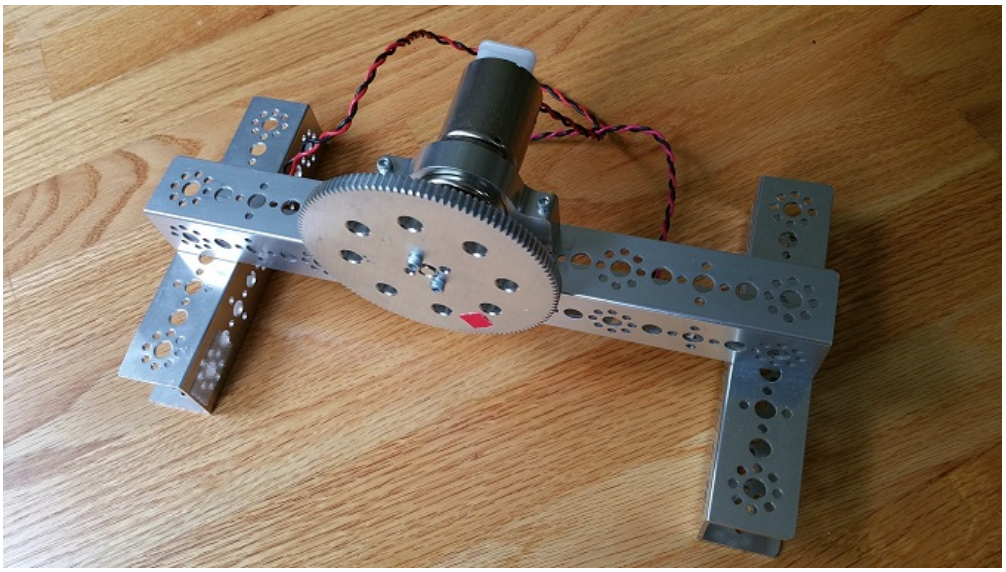
4. Connect the 12V battery to the Tamiya to XT30 cable.

5. Turn on the switch and verify that the Hub is drawing power from the battery. Note that the Hub's LED should be illuminated (notice the blue LED in upper right-hand corner of the Hub in the image below).

6. Turn off the switch and verify that the Hub is off. Note that the Hub's LED should not be illuminated.

8.2. *Connecting a Motor to the Hub*

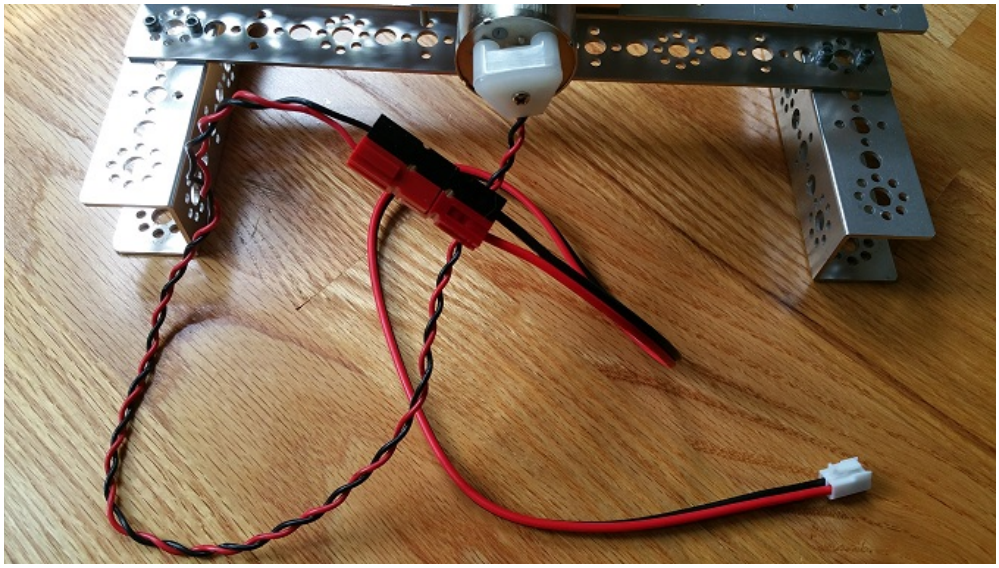
The Hub can drive up to four (4) 12V DC motors per Hub. The Hub uses a type of electrical connector known as a 2-pin JST VH connector. Many of the FIRST-approved 12V DC motors are equipped with Anderson Powerpole connectors. An adapter cable can be used to connect the Anderson Powerpole connectors to the Hub motor port (see FIRST Tech Challenge Robot Wiring Guide for more information).



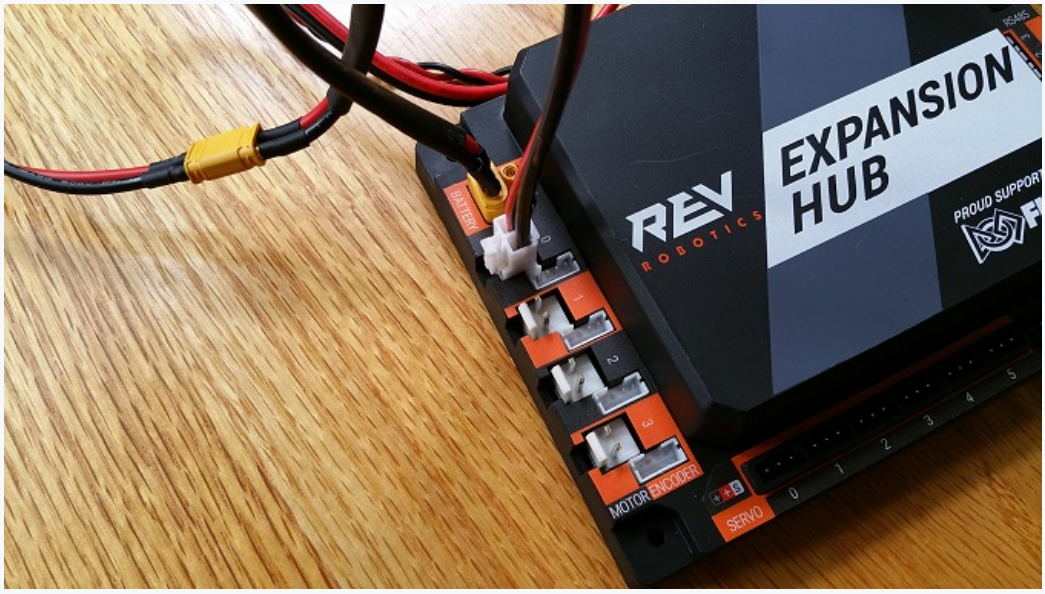
For the examples in this wiki, FIRST recommends that the user build a simple rig to secure the motor in place and prevent it from moving about during the test runs. The image above shows a Tetrix motor installed in a rig built with a Tetrix motor mount and some Tetrix C-channels. A gear was mounted on the motor shaft to make it easier for the user to see the rotation of the shaft.

Note: it will take an estimated 2.5 minutes to complete this task.

1. Connect the Anderson Powerpole end of the motor's power cable to the Powerpole end of the Anderson to JST VH adapter cable.



2. Connect the other end of the Anderson to JST VH adapter cable into the motor port labeled "0" on the Hub.

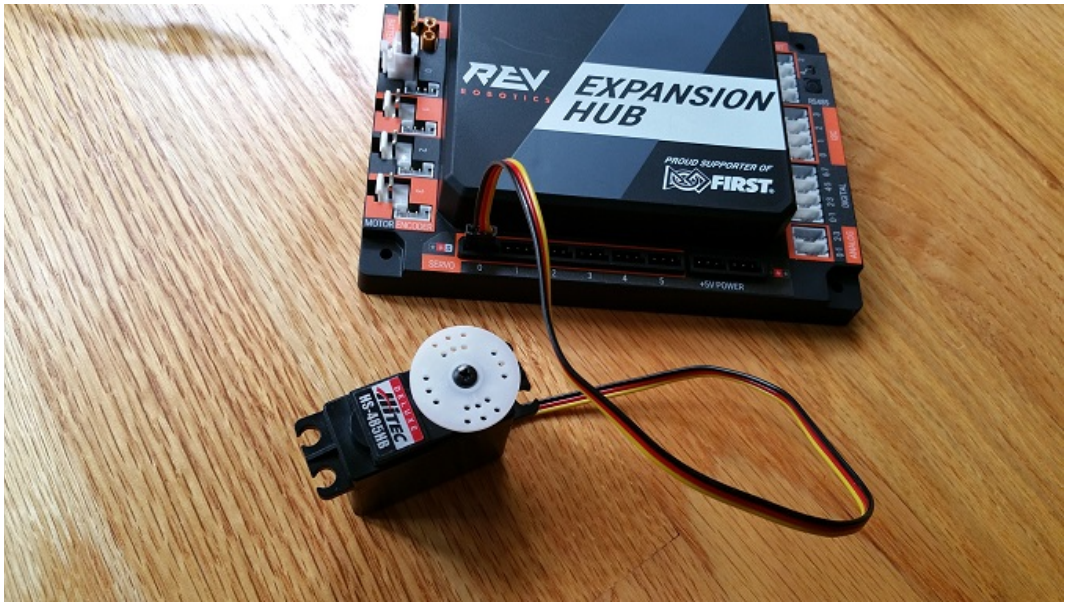


8.3. *Connecting a Servo to the Hub*

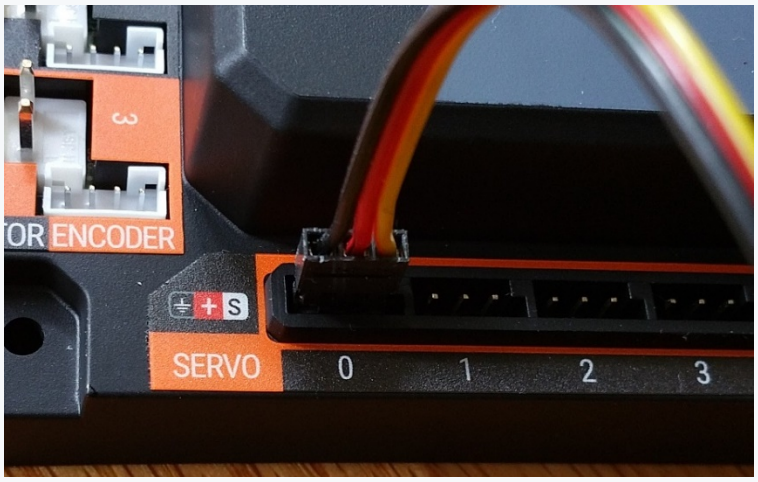
The Hub has 6 built-in servo ports. The servo ports accept the standard 3-wire header style connectors commonly found on servos. Note that ground pin is on the left side of the servo port.

Note: that it will take an estimated 2.5 minutes to complete this task.

1. Connect the servo cable to the servo port labeled “0” on the Hub. Note that the ground pin is on the left side of the servo port.



2. Verify that the black ground wire of the servo cable matches the ground pin of the servo port (which is aligned on the left side of the port).



9. Connecting a Color-Distance Sensor to the Hub

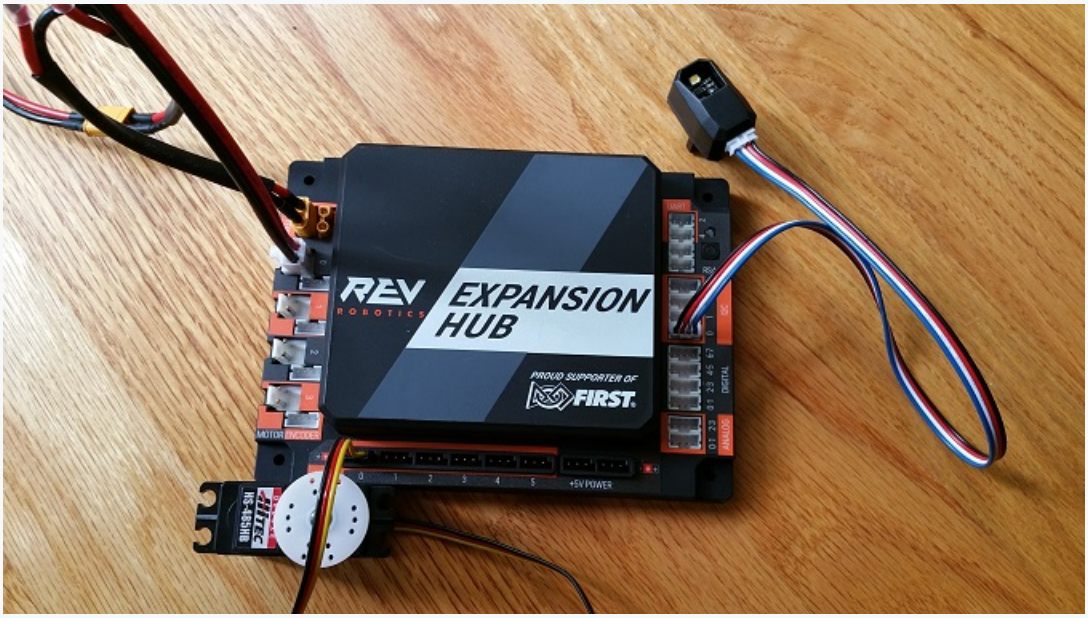
The Hub has 4 independent I2C buses. Each bus has its own port on the Hub. We will connect a REV Robotics Color-Distance sensor to the I2C bus #0 on the Hub.

Note: that it will take an estimated 2.5 minutes to complete this task.

9.1. Connecting a Color-Distance Sensor to the Hub

1. Connect the one end of the 4-pin JST PH cable to the REV Robotics Color-Distance sensor.

2. Plug the other end of the 4-pin JST PH cable to the I2C port labeled “0” on the Hub.



10. Connecting a Touch Sensor to the Hub

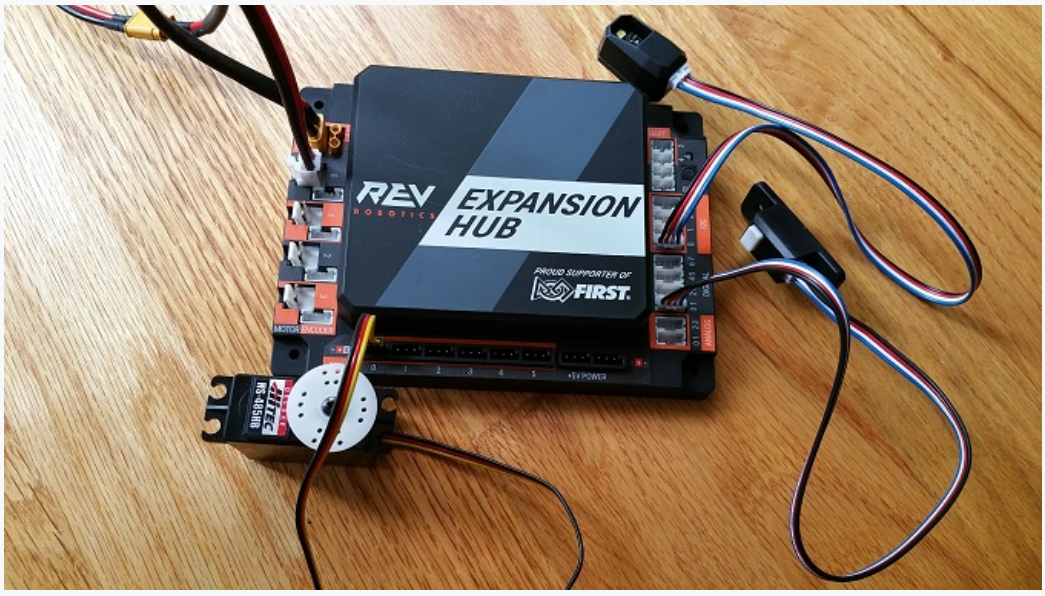
The Hub has 4 independent digital input/output (I/O) ports. Each port has two digital I/O pins for a total of 8 digital I/O pins on a Hub. You will connect a REV Robotics Touch sensor to one of the digital I/O ports.

> **Note:** that in the case of the REV Robotics Touch Sensor, the device has a connector port for a 4-pin sensor cable. However, the device only needs to connect to one of the two available digital I/O pins. For the REV Robotics Touch Sensor, the second digital I/O pin in the port is the one that gets connected when a standard REV Robotics 4-pin JST PH cable is used. For the “0-1” port, it is the pin labeled “1” that gets connected through the 4-pin cable. Similarly, for the “2-3” port, it is the pin labeled “3” that gets connected through the 4-pin cable.
>
> **Note:** that it will take an estimated 2.5 minutes to complete this task.

10.1. Connecting a Touch Sensor to the Hub

1. Connect the one end of the 4-pin JST PH cable to the REV Robotics Touch sensor.

2. Plug the other end of the 4-pin JST PH cable to digital I/O port labeled “0-1” on the Hub.



11. Configuring Your Hardware

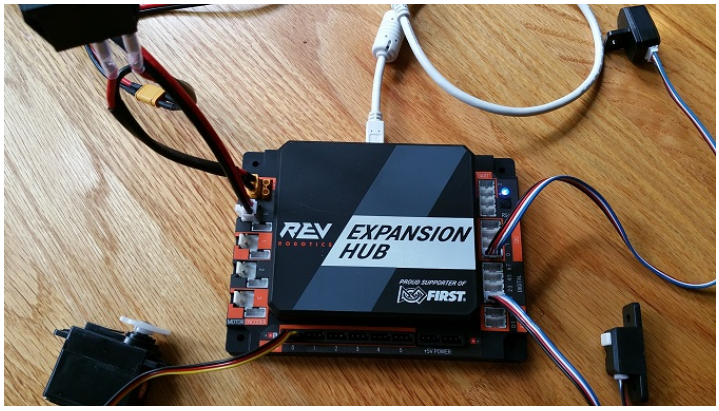
11.1. *Before You Begin...*

Before you can communicate with the motor, servo and sensors that are connected to the Control Hub or Expansion Hub, you first must create a configuration file on your Robot Controller, so that the Robot Controller will know what hardware is available on the Control Hub's or Expansion Hub's external ports.

11.2. *Connecting an Android Smartphone to an Expansion Hub*

If you are using an Android smartphone as a Robot Controller, you must physically connect the Robot Controller smartphone to the Expansion Hub using a USB cable and an On-The-Go (OTG) adapter. Also, you should verify that the Driver Station is currently paired to the Robot Controller.

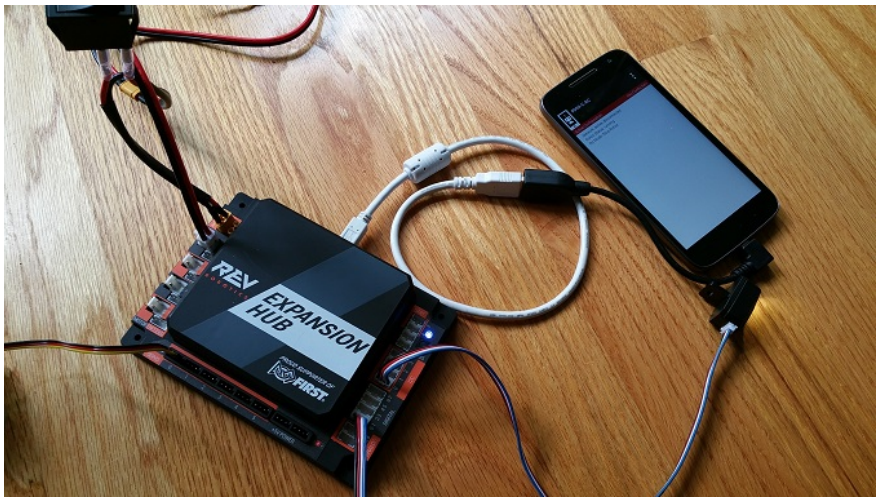
1. Power on the Expansion Hub by turning on the power switch.



2. Plug the Type B Mini end of the USB cable into the USB mini port on the Expansion Hub.

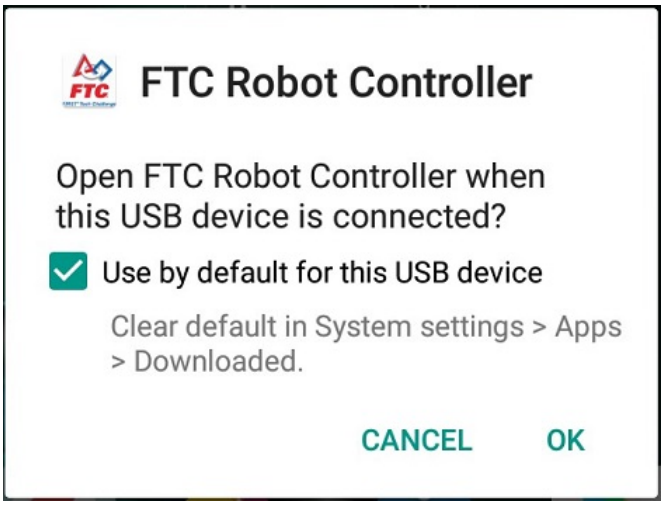
3. Plug the Type A end of the USB cable into the OTG adapter.

4. Verify that your Robot Controller smartphone is powered on and unlocked. Plug in the USB Micro OTG adapter into the OTG port of the Robot Controller phone.



Note: that when the OTG adapter is plugged into the smartphone, the phone will detect the presence of the Expansion Hub and launch the Robot Controller app.

5. The first time you connect the Robot Controller smartphone to the Expansion Hub, the Android operating system should prompt you to ask if it is OK to associate the newly detected USB device (which is the Expansion Hub) with the FTC Robot Controller app.



Important Information!
You might be prompted multiple times to associate the USB hardware with the FTC Robot Controller. Whenever you are prompted by your phone with this message, you should always select the "Use by default for this USB device" option and hit the "OK" button to associate the USB device with the FTC Robot Controller app.

If you fail to make this association, then the Robot Controller app might not reliably connect to this Expansion Hub the next time you turn your system on.

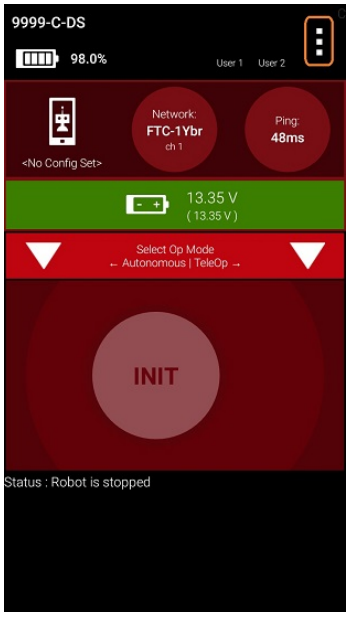
11.3. *Getting the Control Hub Ready*

If you are using a Control Hub, you do not need to make any additional connections. You simply need to make sure that the Control Hub is powered on and paired to the Driver Station.

11.4. *Creating a Configuration File Using the Driver Station*

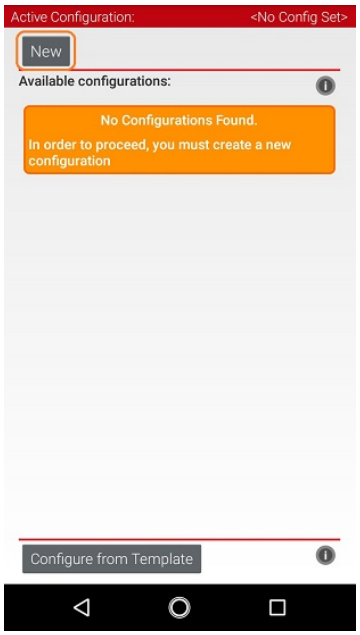
Although the configuration file needs to reside on the Robot Controller, for this tutorial we will use the Driver Station app to create the configuration file remotely. The Driver Station can be used to create a configuration file for a Control Hub or for an Android smartphone Robot Controller.

1. Touch the three vertical dots in the upper right hand corner of the Driver Station app. This will launch a pop-up menu.



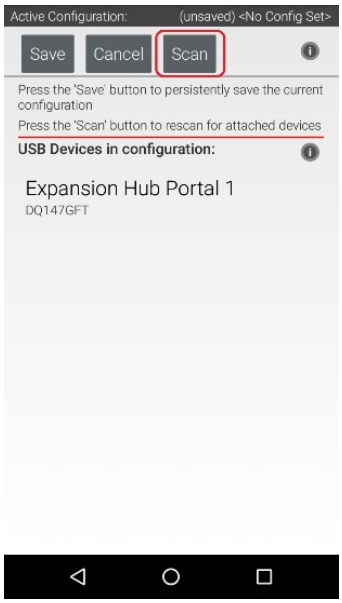
2. Select **Configure Robot** from the pop up menu to display the **Configuration** screen.

3. If your Robot Controller does not have any existing configuration files, the screen will display a message indicating that you need to create a file before proceeding.



Hit the **New** button to create a new configuration file for your Robot Controller.

4. When the new configuration screen appears, the Robot Controller app will do a scan of the serial bus to see what devices are connected to the Robot Controller.

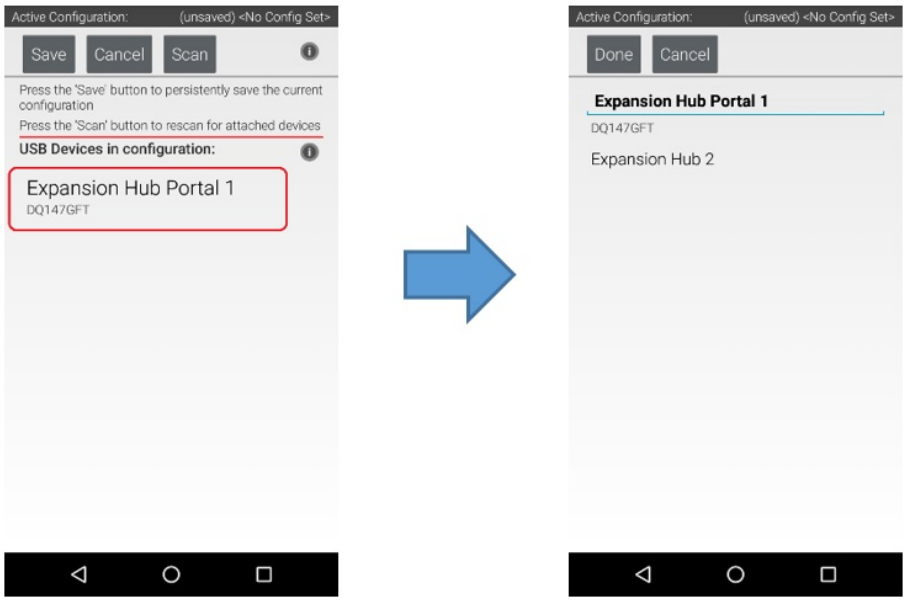


It will display the devices that it found in a list underneath the words “USB Devices in configuration.” You should see an entry that says something like “Expansion Hub Portal 1” in the list.

Your Expansion Hub is listed as a Portal because it is directly connected to the Robot Controller phone through the USB cable or in the case of the Control Hub through the internal serial bus.

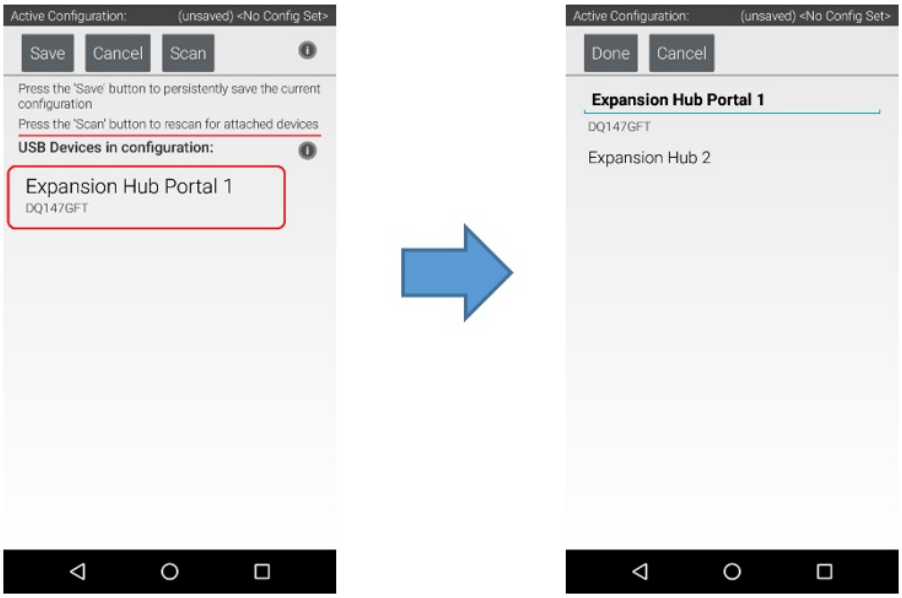
If you do not see your Expansion Hub Portal listed and you are using a smartphone as a Robot Controller, check the wired connections to make sure they are secure and then press the Scan button one or two times more to see if the smartphone detects the device on a re-scan of the USB bus.

5. Touch the Portal listing (“Expansion Hub Portal 1” in this example) to display what Expansion Hubs are connected through this Portal.



Since we only have a single Expansion Hub connected, we should only see a single Expansion Hub configured (“Expansion Hub 2” in this example).

6. Touch the Expansion Hub listing (“Expansion Hub 2” in this example) to display the Input/Output ports for that device.



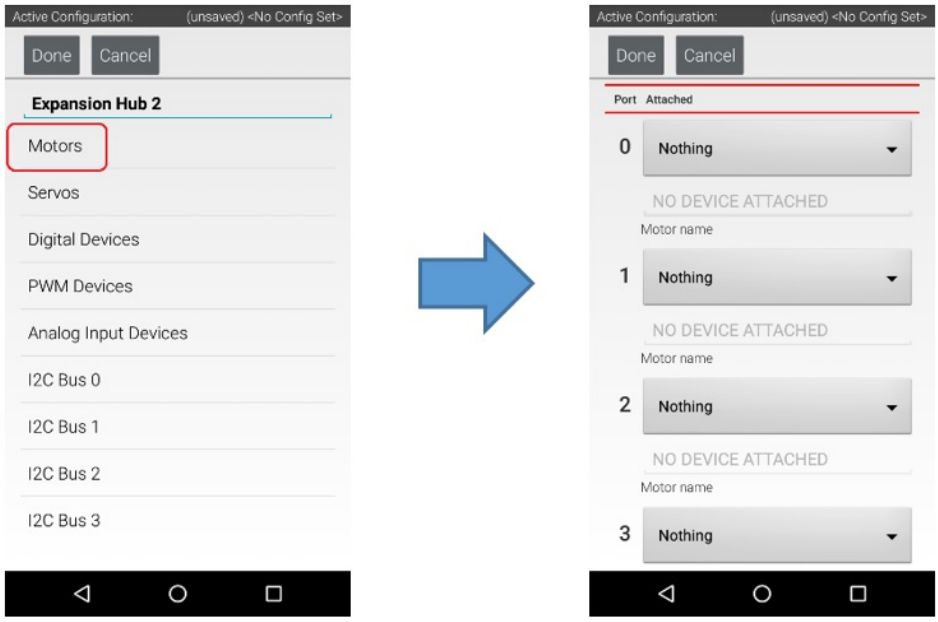
The screen should change and list all the motor, servo and sensor ports that are available on the selected Expansion Hub.

11.5. *Configuring a DC Motor*

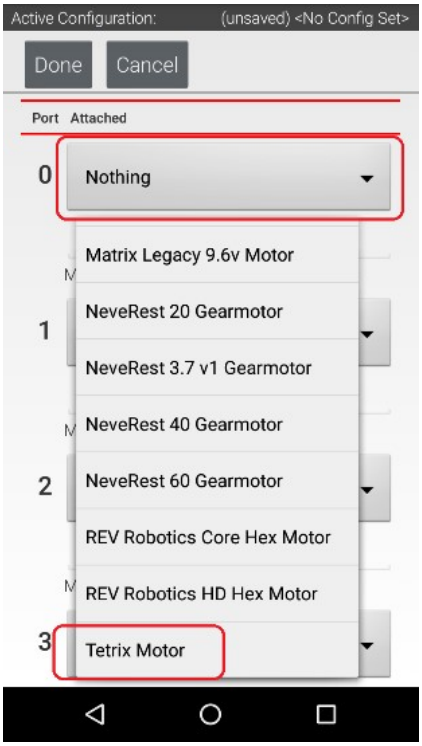
Now that you've created a file, you will need to add a DC Motor to the configuration file.

> **Important Note:** At this point, although you have created your configuration file, you have not yet saved its contents to the Robot Controller. You will save the configuration file in a later step.

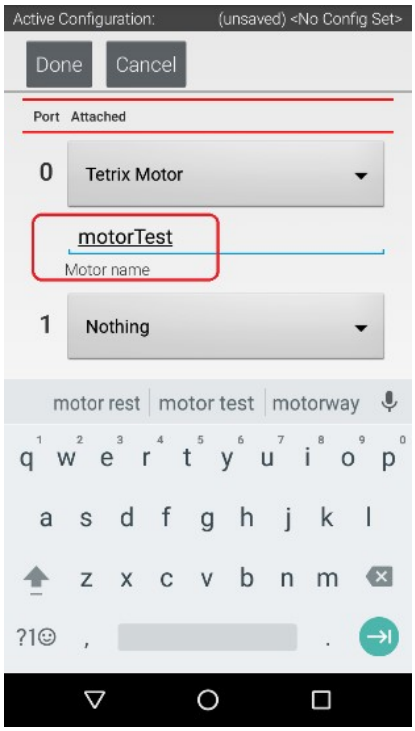
1. Touch the word **Motors** on the screen to display the Motor Configuration screen.



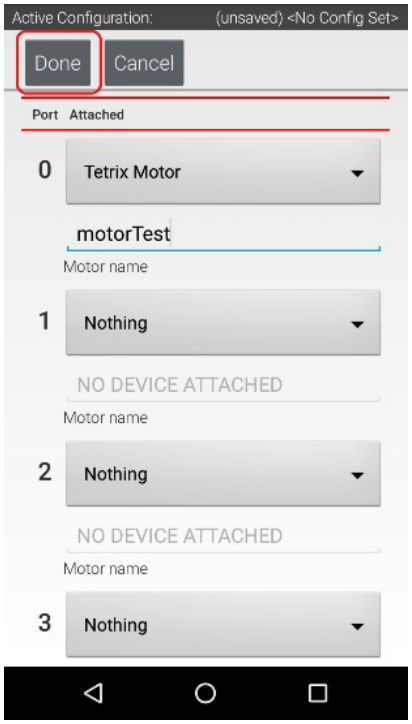
2. Since we installed our motor onto port #0 of the Expansion Hub, use the dropdown control for port 0 to select the motor type (Tetrix Motor for this example).



3. Use the touch screen keypad to specify a name for your motor (“motorTest” in this example).



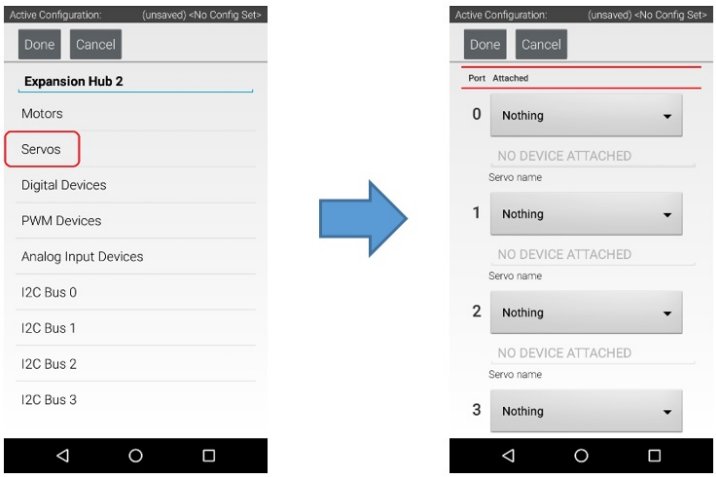
4. Press the **Done** button to complete the motor configuration. The app should return to the previous screen.



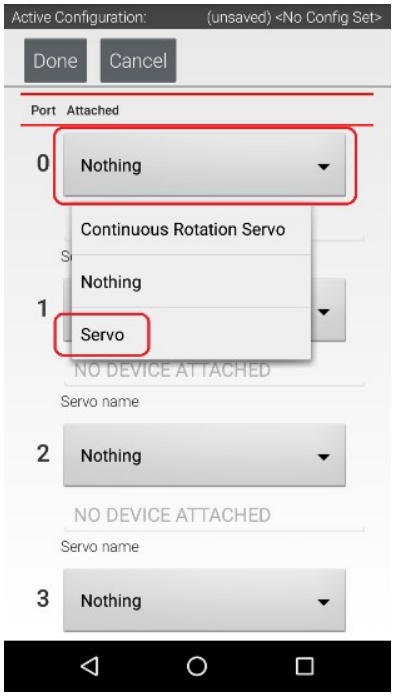
11.6. *Configuring a Servo*

You will also want to add a servo to the configuration file. In this example, you are using a standard 180-degree servo.

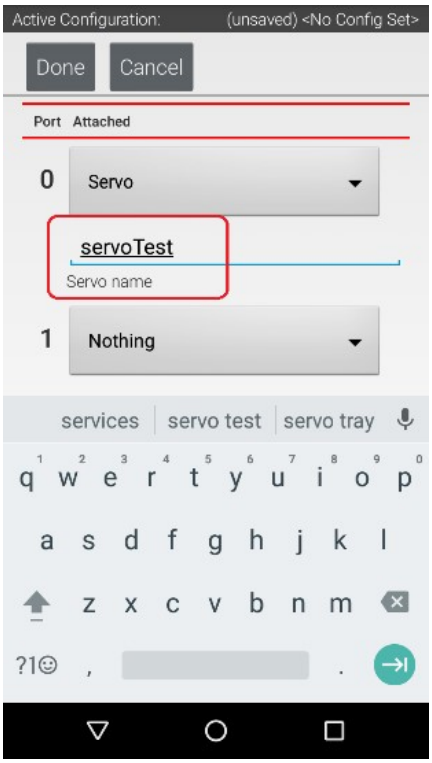
1. Touch on the word **Servos** on the screen to display the **Servo Configuration** screen.



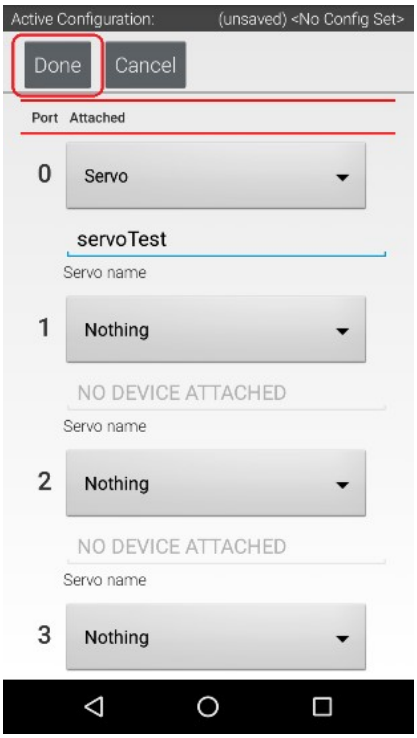
2. Use the dropdown control to select “Servo” as the servo type for port #0.



3. Use the touch pad to specify the name of the servo (“servoTest” for this example) for port #0.



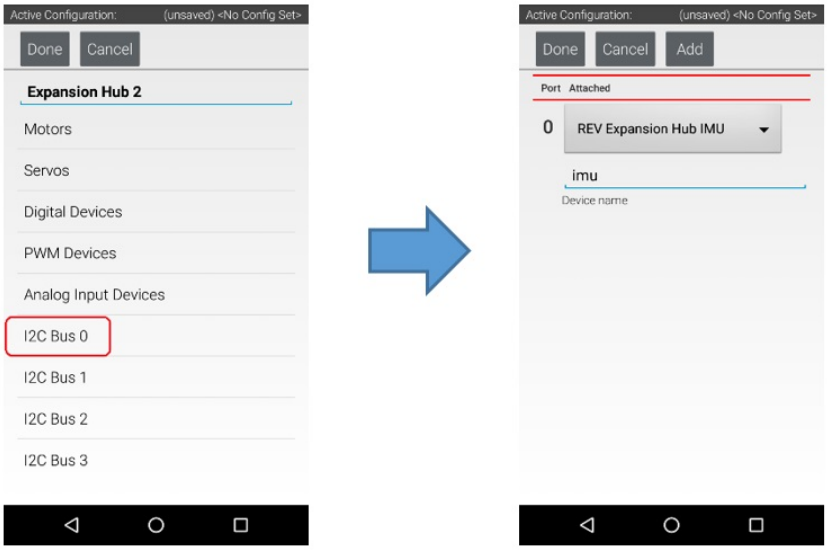
4. Press the **Done** button to complete the servo configuration. The app should return to the previous screen.



11.7. *Configuring a Color Distance Sensor*

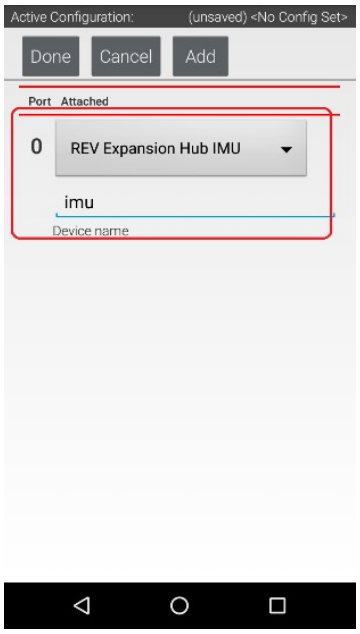
The REV Robotics Color Distance Sensor is an I2C sensor. It actually combines two sensor functions into a single device. It is a color sensor, that can determine the color of an object. It is also a distance or range sensor, that can be used to measure short range distances. Note that in this tutorial, the word "distance" is used interchangeably with the word "range".

1. Touch the words **I2C Bus 0** on the screen to launch the I2C configuration screen for this I2C bus.



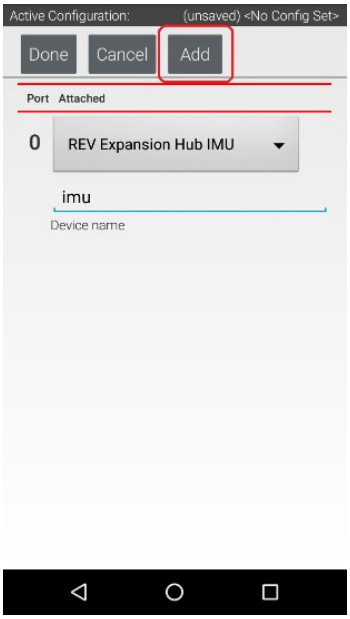
The Expansion Hub has four independent I2C buses, labeled “0” through “3”. In this example, since you connected the Color Sensor to the port labeled “0”, it resides on I2C Bus 0.

2. Look at the **I2C Bus 0** screen. There should already be a sensor configured for this bus. The Expansion Hub has its own built-in inertial measurement unit (IMU) sensor. This sensor can be used to determine the orientation of a robot, as well as measure the accelerations on a robot.

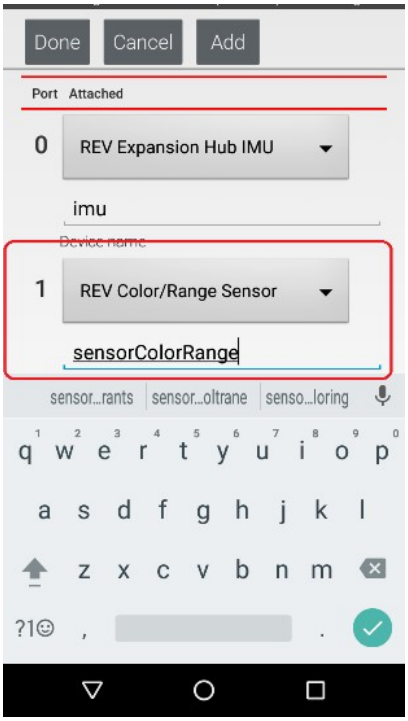


The built-in IMU is internally connected to I2C Bus 0 on each Expansion Hub. Whenever you configure an Expansion Hub using the Robot Controller, the app automatically configures the IMU for I2C Bus 0. You will need to add another I2C device for this bus to be able to configure the color sensor.

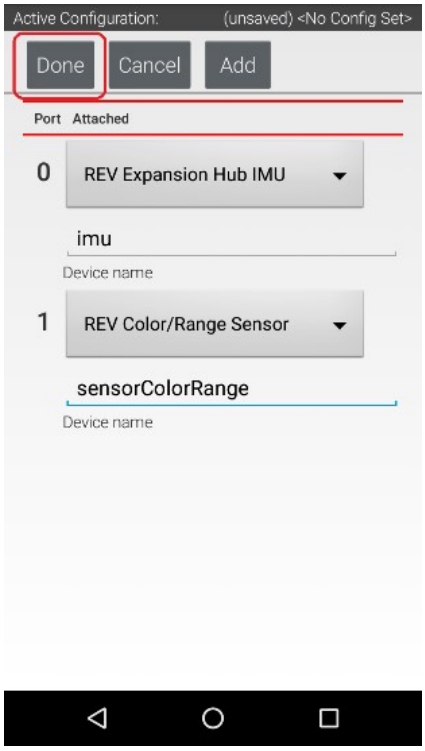
3. Press the **Add** button to add another I2C device to this bus.



4. Select “REV Color/Range Sensor” from the dropdown selector for this new device. Use the touchscreen keyboard to name this device “sensorColorRange”.



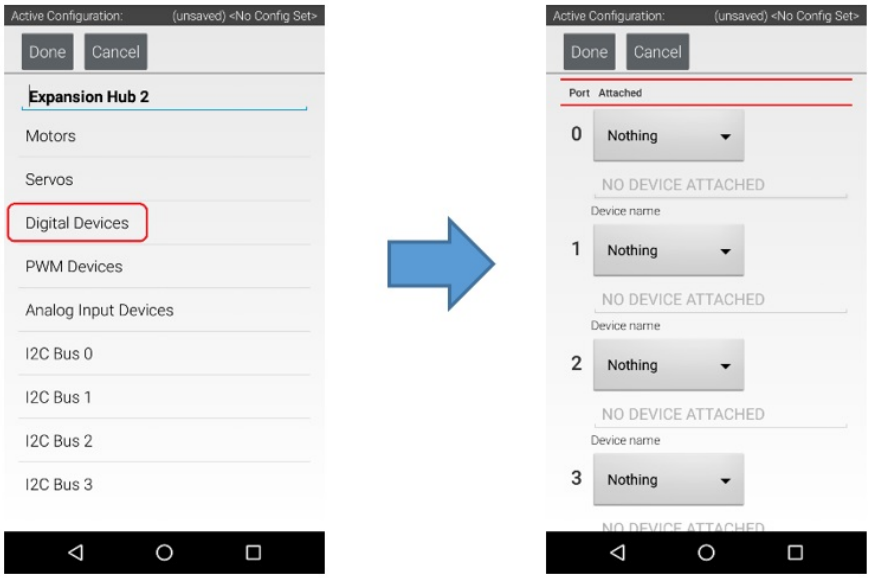
5. Press the **Done** button to complete the I2C sensor configuration. The app should return to the previous screen.



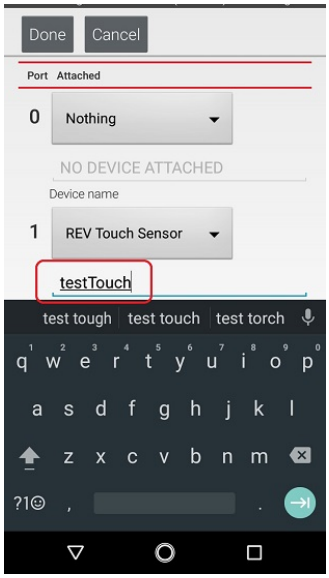
11.8. *Configuring a Digital Touch Sensor*

The REV Robotics Touch Sensor is a digital sensor. An Op Mode can query the Touch Sensor to see if its button is being pressed or not.

1. Touch the words **Digital Devices** on the screen to launch the Digital I/O configuration screen.

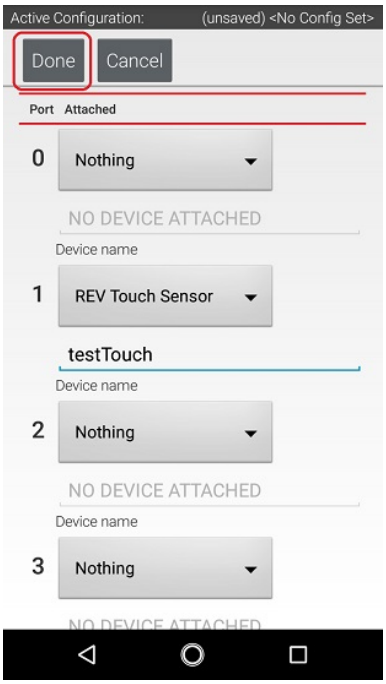


2. Use the touch screen to add a “REV Touch Sensor” for port #1 and name the device “testTouch”.



Notice that we are configuring the Touch Sensor on port #1 instead of port #0. This is because when the REV Robotics Touch Sensor is connected to a digital port using a standard 4-wire JST sensor cable, it is the second digital pin that is connected. The first pin remains disconnected.

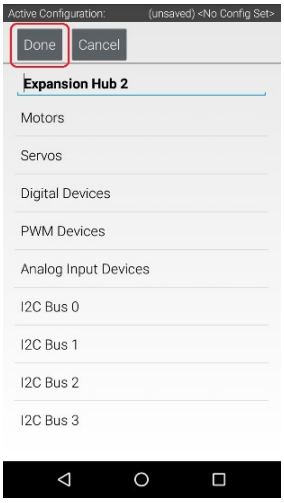
3. Press the **Done** button to return to the previous screen.



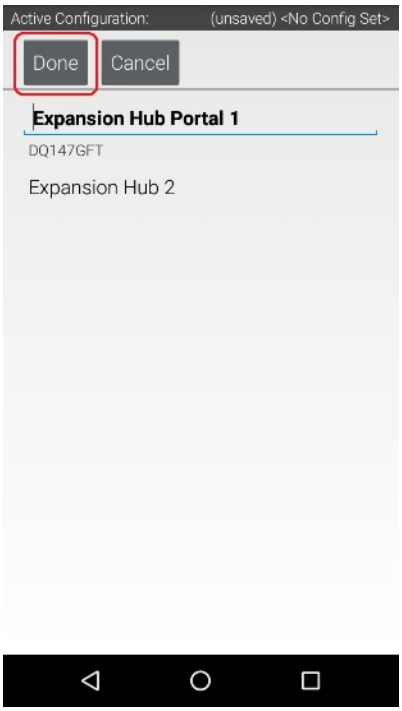
11.9. *Saving the Configuration Information*

Once you have configured your hardware, you must save the information to the configuration file. If you do not save this information, it will be lost and the robot controller will be unable to communicate with your hardware.

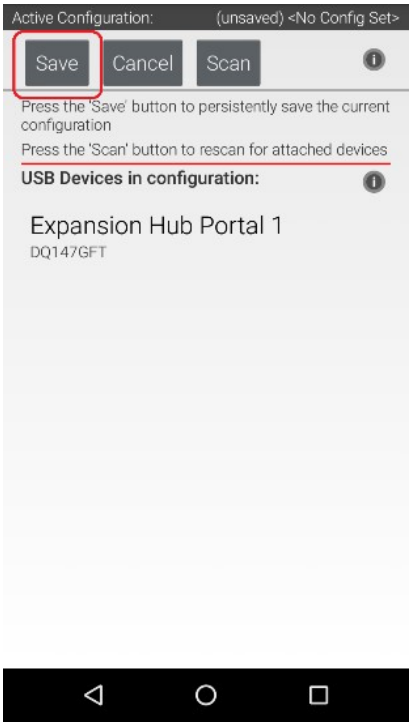
1. Press the **Done** button to go up one level in the configuration screens.



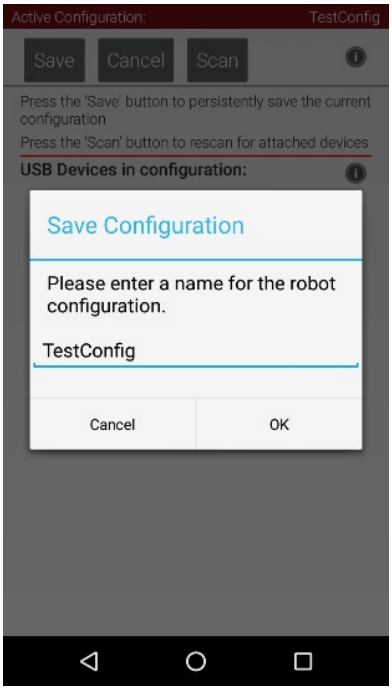
2. Press the **Done** button again to return to the highest level in the configuration screens.



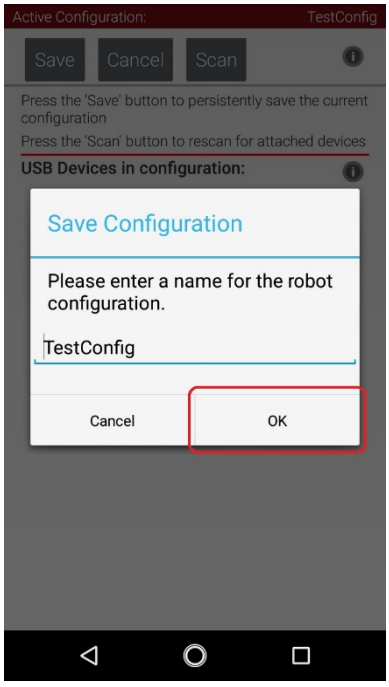
3. Press the **Save** button.



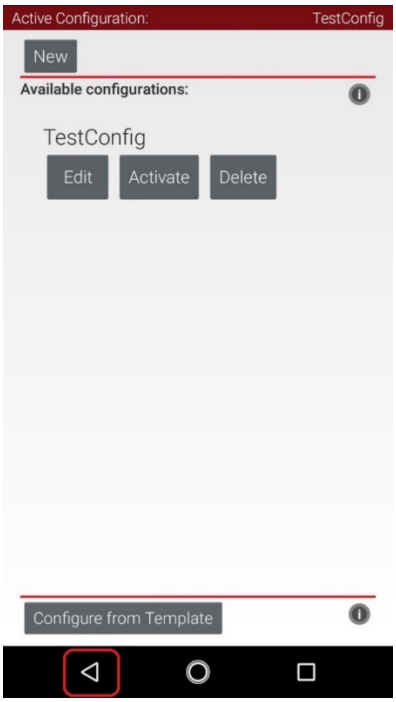
4. When prompted, specify a configuration file name using the touchscreen's keypad (use "TestConfig" for this example).



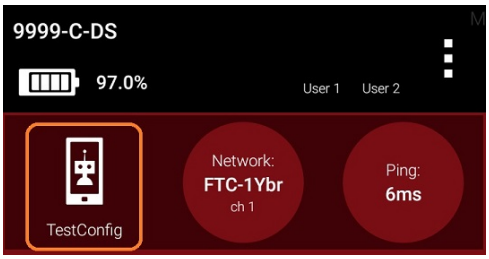
5. Press the **OK** button to save your configuration information using that file name.



6. After the configuration file has been saved, touch the Android back-arrow button to return to the main screen of the app.



7. Verify that the configuration file is the active configuration file on the main Driver Station screen.



12. Installing a Javascript Enabled Browser

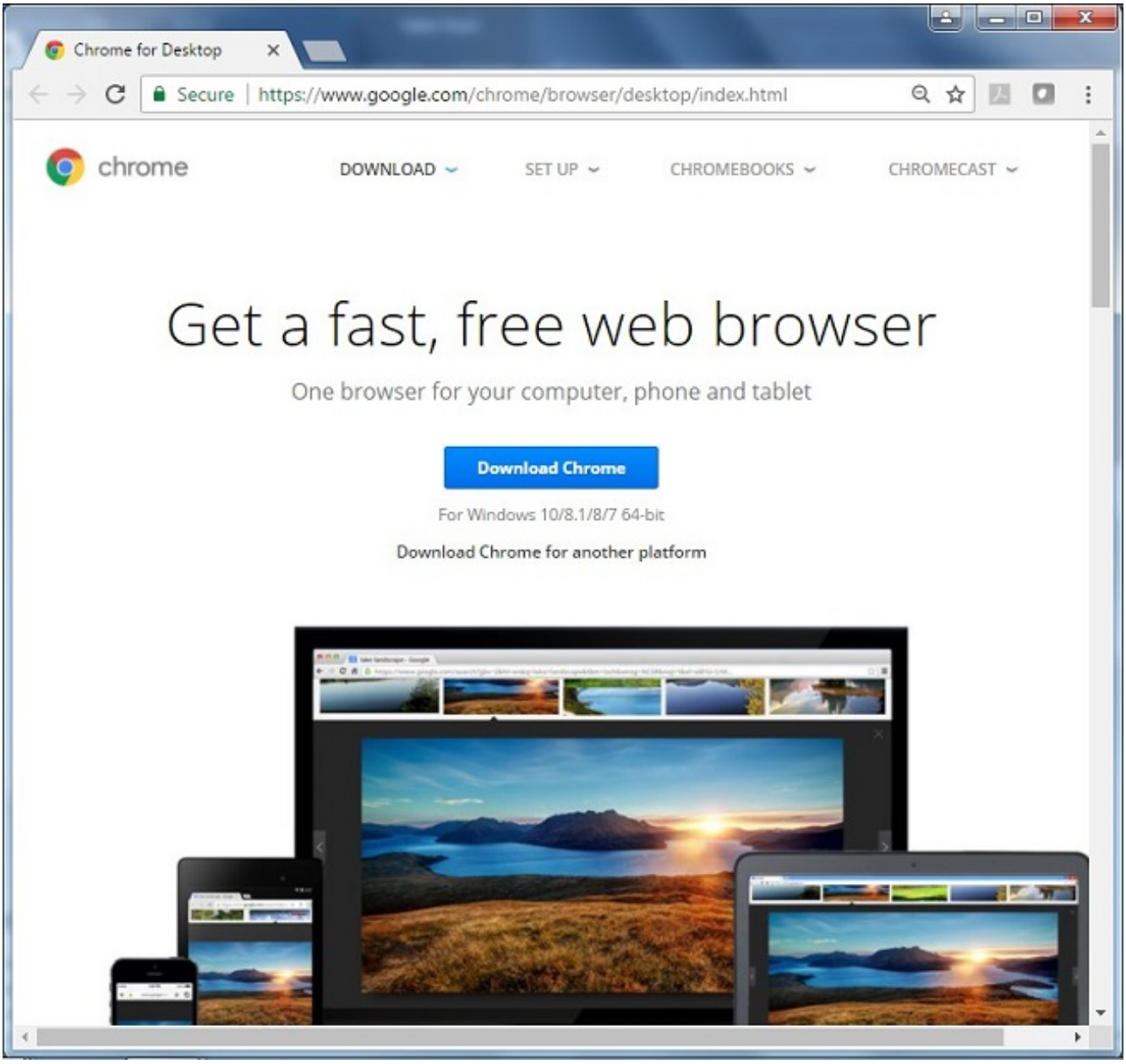
In order to be able to program your Robot Controller using the Blocks Programming Tool or the OnBot Java Programming tool, your laptop will need a Javascript-enabled browser. Both tools are Javascript applications that are served up by the Program and Manage server of the Robot Controller.

The Blocks Programming Tool and the OnBot Java Programming Tool should work with most modern web browsers. However, *FIRST* strongly recommends the use of Google Chrome with these tools. If you would like to use Google Chrome as your browser, you can download it for free from the Google Chrome website.

Note: that it will take an estimated 15 minutes (depending on the speed of your Internet connection) to download and install the Javascript-enabled browser.

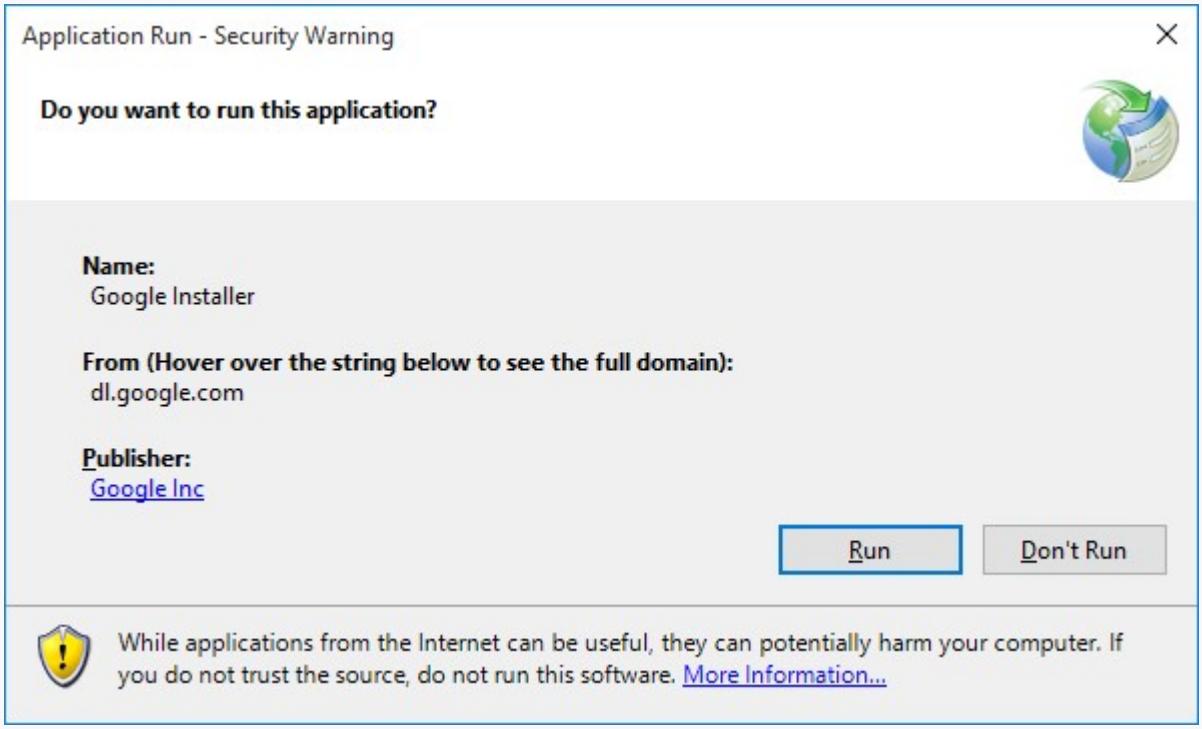
12.1. Installing a Javascript-Enabled Browser

1. Visit the Google Chrome website (using your computer's existing browser) and follow onscreen instructions to download and install Chrome.



https://www.google.com/chrome/browser/desktop/index.html

2. Note that your computer might prompt you with a security warning during the installation process. If you are prompted with this warning, click on the “Run” button to continue with the installation.



13. Connecting a Laptop to the Program & Manage Network

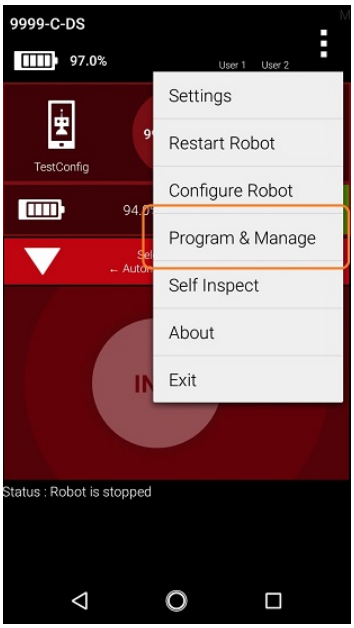
13.1. Introduction

In order to write an Op Mode, you will need to connect your programming laptop to the Program & Manage Wi-Fi network. The Program & Manage Wi-Fi network is a wireless network created by your Robot Controller. Before you begin this exercise, please make sure that your Windows laptop has the most current service pack and system update from Microsoft installed.

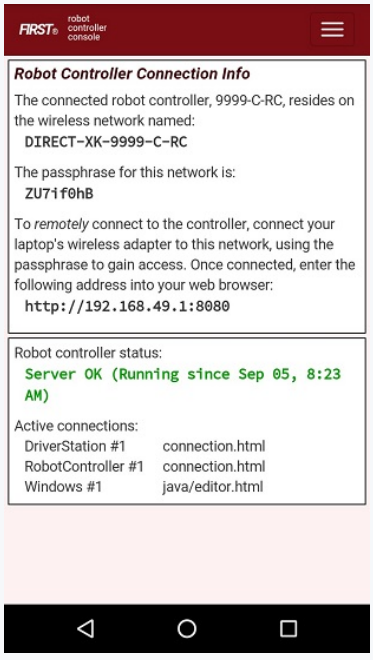
> **Note:** This example assumes the user has a Windows 10 laptop. If you are not using a Windows 10 laptop, the procedure to connect to the Programming & Manage Wi-Fi network will differ. Refer to your device's documentation for details on how to connect to a Wi-Fi network.

13.2. Connecting Your Laptop to the Program & Manage Network

1. On the Driver Station, touch the three dots in the upper right hand corner of the screen to launch the pop-up menu. Select **Program & Manage** from the pop-up menu to display the **Program & Manage** access information.

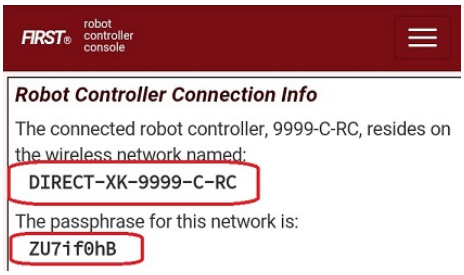


2. The Program & Manage screen displays important information that you can use to connect your laptop to the FTC Blocks or OnBot Java Programming Mode server.



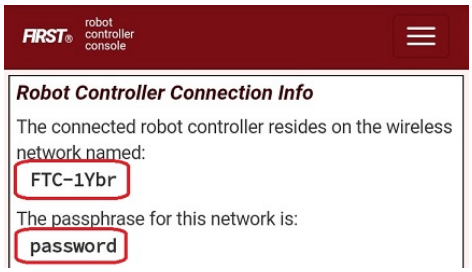
3. Verify the network name and passphrase for the Program & Manage wireless network. Towards the top of the screen, the name of the Program & Manage wireless network is displayed. If you are using an Android smartphone as your Robot Controller, then the wireless network name will begin with the phrase "DIRECT-".

In this example, the name of the Wi-Fi network is “DIRECT-XK-9999-C-RC” and the secure passphrase is “ZU7if0hB”

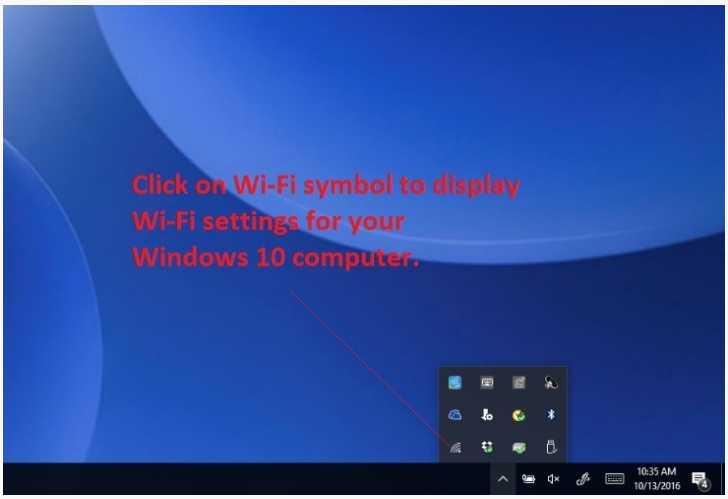


If you are using a Control Hub, then the wireless network name will be whatever you specified when you configured your Control Hub. If you haven't changed the Control Hub's name yet, then by default the wireless network's name will begin with "FTC-". If you haven't changed its password yet, then by default the wireless network's passphrase will be "password".

In the screenshot below, the Control Hub's wireless network name is "FTC-1Ybr" and the secure passphrase is "password".



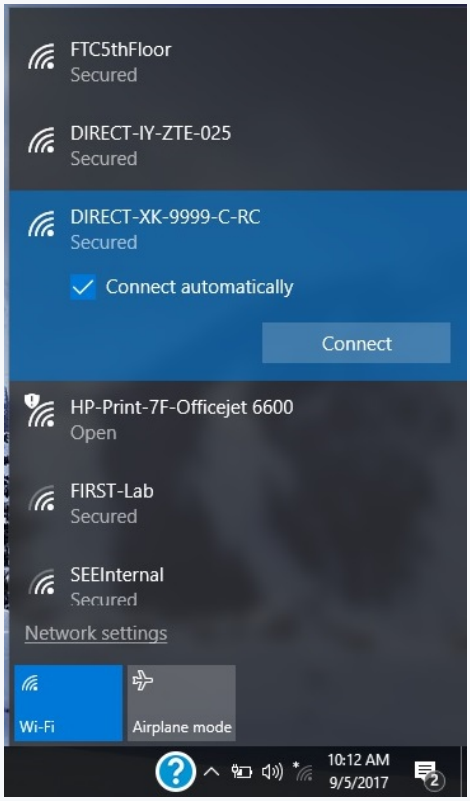
4. On your Windows 10 computer, look in the lower right hand corner of your desktop for a Wi-Fi symbol. Click on the Wi-Fi symbol to display a list of available Wi-Fi Networks in your vicinity.



5. Look for the wireless network that matches the name displayed on the Program & Manage screen.

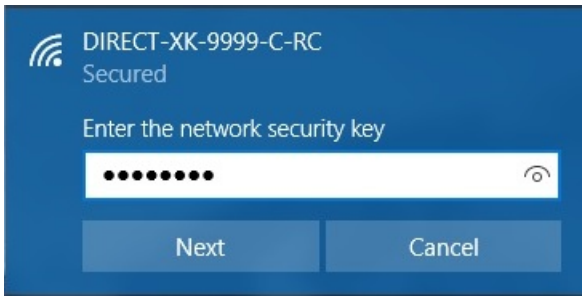
In this example, the name of the wireless network for the Android Robot Controller is “DIRECT-XK-9999-C-RC” and the network is visible in the list displayed on the Windows 10 computer.

6. Once you have found the target network in the list, click on it to select it.



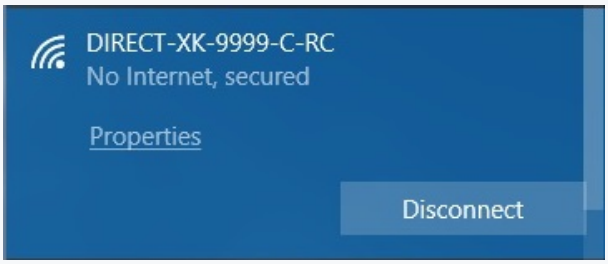
Press the Connect button to connect to the network.

7. When prompted, provide the network passphrase (in this example “ZU7if0hB”) and press “Next” to continue.



Note that the passphrase is case sensitive. Make sure that your spelling and capitalization matches the original spelling and capitalization shown on the Program & Manage screen.

8. Once you have successfully established a wireless connection between your Windows 10 laptop and your Robot Controller Android device, the status should be displayed in the wireless settings for your laptop.



If the display is not updated as shown after a few seconds, try clicking on Network Connections at the bottom of the blue box showing the Wi-Fi connections. This will bring up a Setting dialog box that includes a link to "Show available networks.", which can be used to force the list of Wi-Fi connections to be updated.

> **Note:** that when you are connected to the blocks programming mode server on your Robot Controller, your laptop *will not have access to the Internet*. It only has direct access to the Robot Controller.

13.3. *Troubleshooting Your Wireless Connection*

If you cannot see your Programming Mode wireless network in the list of available networks, or, if you are having problems connecting your laptop to the Program & Manage wireless network, make sure you answer the following questions:

1. Is the Robot Controller running and connected to the Driver Station?
2. Is your Windows laptop updated with the most current system updates and service packs? Older versions of Windows 8 and 10, for example, had issues that could prevent the laptop from displaying the Program & Manage wireless network in the list of available networks.

If you are still having issues with connecting the laptop to the Robot Controller, visit Troubleshooting section of this wiki for instructions on how to manually connect to the Program & Manage wireless network with a Windows 10 laptop.

14. Creating and Running an Op Mode (OnBot Java)

14.1. *The Java Programming Language*

This tutorial assumes that you have a sound understanding of the Java programming language. If you do not know Java, then you should consider using the FTC Blocks Programming Tool, which is a visual development tool. Information about the FTC Blocks Programming Tool can be found at the following link:

FTC Blocks Tutorial

Or, you can learn the Java programming language by completing the Oracle Java Tutorial, which is available at the following address:

https://docs.oracle.com/javase/tutorial/

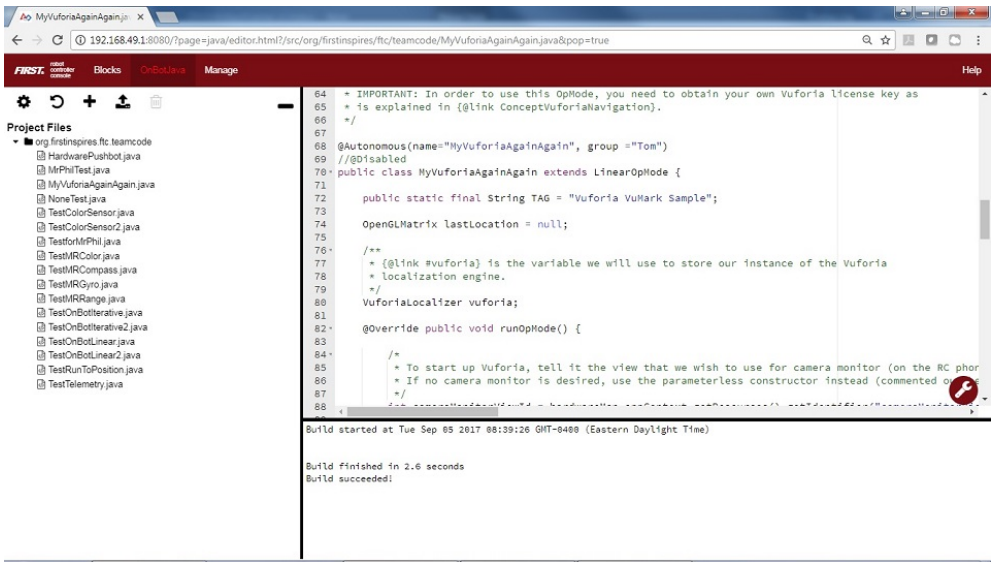
14.2. *What's an Op Mode?*

During a typical FIRST Tech Challenge match, a team's robot must perform a variety of tasks to score points. For example, a team might want their robot to follow a white line on the competition floor and then score a game element into a goal autonomously during a match. Teams write programs called "op modes" (which stands for "operational modes") to specify the behavior for their robot. These op modes run on the Robot Controller phone after being selected on the Driver Station phone.

Teams who are participating in the FIRST Tech Challenge have a variety of programming tools that they can use to create their own op modes. This document explains how to use the FTC OnBot Java Programming Tool to write an op mode for an FTC robot.

14.3. *The FTC OnBot Java Programming Tool*

The FTC OnBot Java Programming Tool is a user-friendly programming tool that is served up by the Robot Controller phone. A user can create custom op modes for their robot using this tool and then save these op modes directly onto the Robot Controller. Users write their op modes using Java. The op modes are compiled very quickly on the Robot Controller and then loaded dynamically by the Robot Controller during run time.



The examples in this document use a Windows laptop computer to connect to the Robot Controller. This Windows laptop computer has a Javascript-enabled web browser installed that is used to access the FTC OnBot Java Programming Tool.

Note: that the process used to create and edit an op mode is identical if you are using a Control Hub as your Robot Controller.

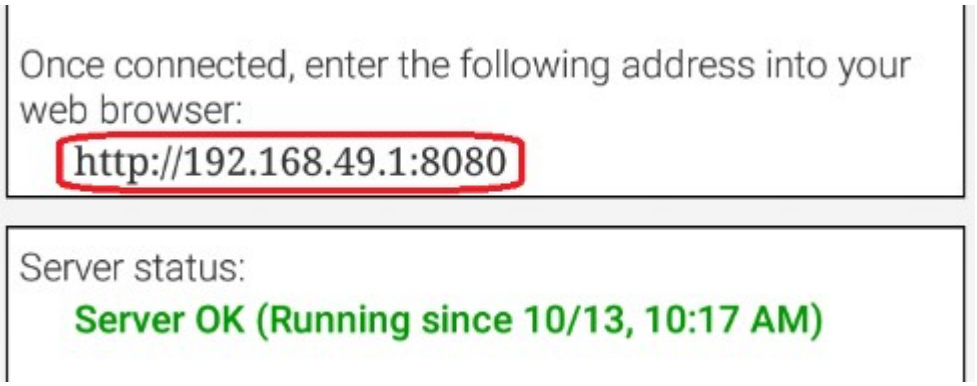
Note: that if you prefer, you can use an alternate device, such as an Apple Mac laptop, Chromebook, or an iPad instead of a Windows computer to access the OnBot Java Programming Tool. The instructions included in this document, however, assume that you are using a Windows laptop.

Note: that this section of the wiki assumes that you have already setup and configured your Android devices and robot hardware. It also assumes that you have successfully connected your laptop to the Progam & Manage server on the Robot Controller device.

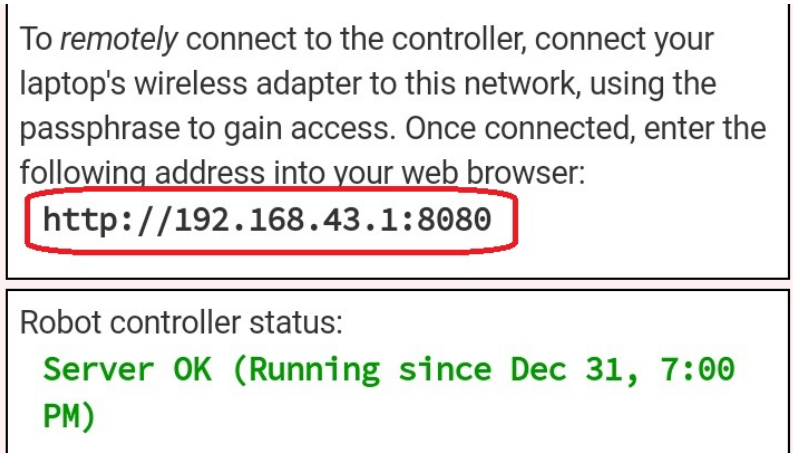
14.4. *Creating Your First Op Mode*

If you connected your laptop successfully to the Program & Manage wireless network of the Robot Controller, then you are ready to create your first op mode. In this section, you will use the OnBot Java Programming Tool to create the program logic for your first op mode.

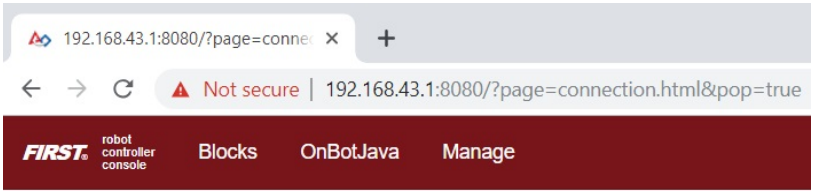
1. Launch the web browser on your laptop (FIRST recommends using Google Chrome) and find the web address that is displayed on the Program & Manage screen of the Robot Controller.
Important Note: If your Robot Controller is an Android smartphone, then the address to access the Program & Manage server is "192.168.49.1:8080".



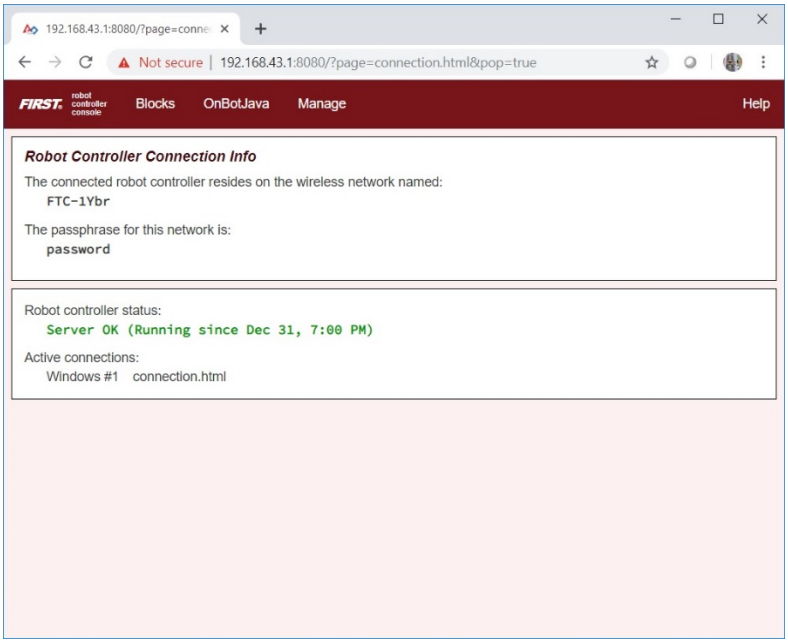
Important Note: If your Robot Controller is a Control Hub, then the address to access the Program & Manage server is "192.168.43.1:8080". Notice the difference in the third octet of the IP addresses (the Control Hub has a "43" instead of a "49").



Type this web address into the address field of your browser and press RETURN to navigate to the Program & Manage web server.

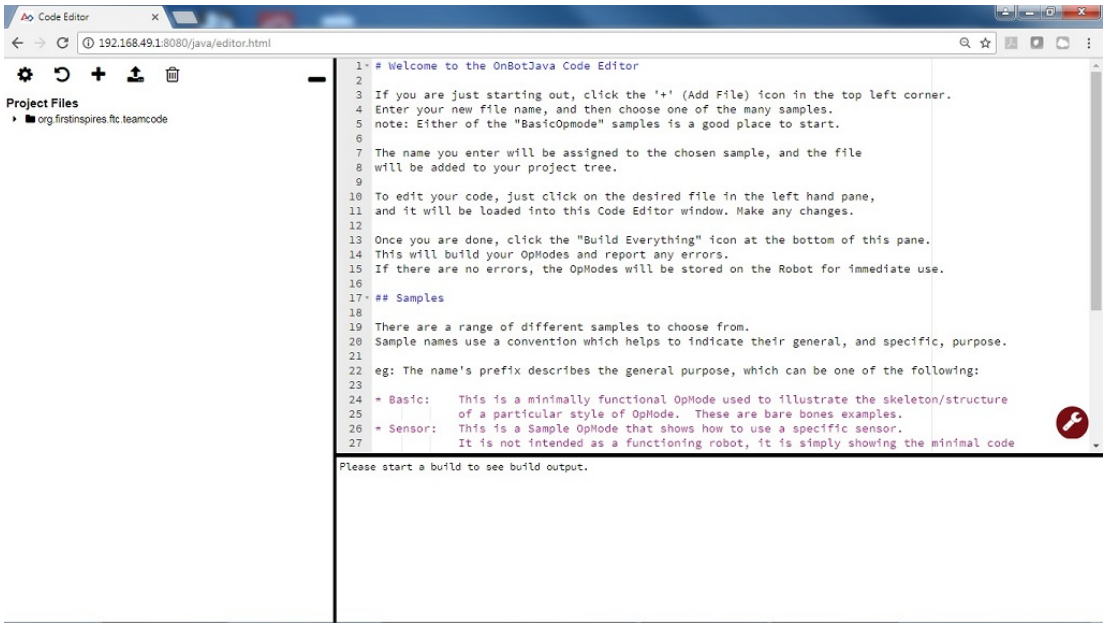


2. Verify that your web browser is connected to the programming mode server. If it is connected to the programming mode server successfully, the Robot Controller Console should be displayed.

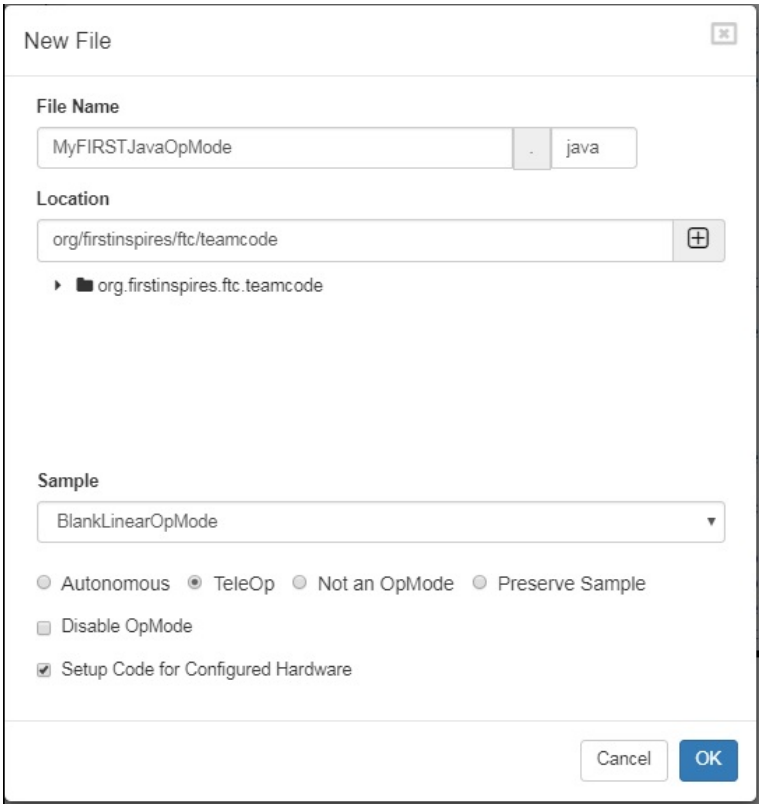


3. Click on the word “OnBotJava” towards the top of the screen. This will switch the browser to OnBot Java Programming mode.

4. Take a look at the OnBot Java user interface. On the left hand side, there is the project browser pane. In the upper right hand corner, there is the source code editing pane. In the lower right hand corner, there is the message pane.



5. In the project browser pane, press the “+” symbol to create a new file. Pushing this button will launch the New File dialog box. This dialog box has several parameters that you can configure to customize your new file.



For this example, specify “MyFIRSTJavaOpMode” as the File Name in the New File dialog box.

Using the Sample dropdown list control, select “BlankLinearOpMode” from the list of available sample op modes (see image above). By selecting “BlankLinearOpMode” the OnBot Java editor will automatically generate a basic LinearOpMode framework for you.

Check the option labeled “TeleOp” to ensure that this new file will be configured as a tele-operated (i.e., driver controlled) op mode.

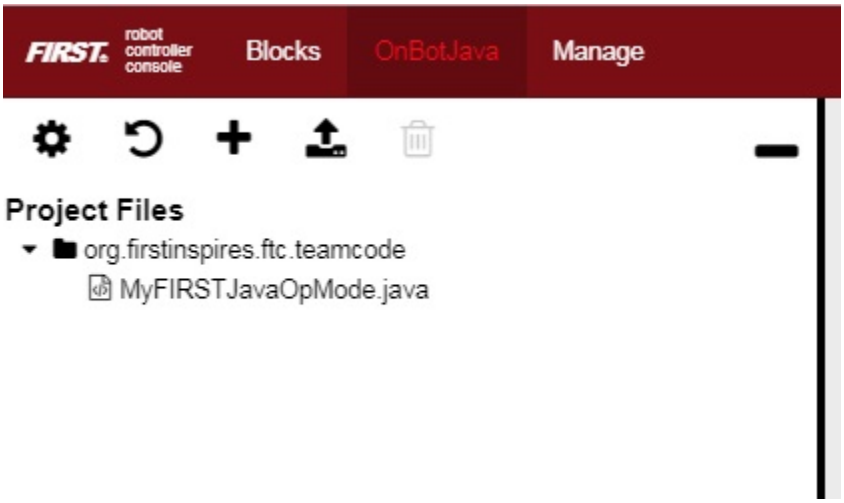
Also, make sure you check the “Setup Code for Configured Hardware” option. If this option is enabled, the OnBot Java editor will look at the hardware configuration file for your Robot Controller and automatically generate the code that you will need to access the configured devices in your op mode.

Press the “OK” button to create your new op mode.

6. You should see your newly created op mode in the editing pane of the OnBot Java user interface.

```
/*
Copyright 2017 FIRST Tech Challenge Team 9999

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and
associated documentation files (the "Software"), to deal in the Software without restriction,
including without limitation the rights to use, copy, modify, merge, publish, distribute,
sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is
furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all copies or substantial
portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT
NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND
NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM,
DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM,
OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.
*/
package org.firstinspires.ftc.teamcode;

import com.qualcomm.robotcore.eventloop.opmode.LinearOpMode;
import com.qualcomm.robotcore.hardware.Gyroscope;
import com.qualcomm.robotcore.hardware.DigitalChannel;
import com.qualcomm.robotcore.hardware.DistanceSensor;
import com.qualcomm.robotcore.hardware.Servo;
import com.qualcomm.robotcore.eventloop.opmode.TeleOp;
```

Congratulations, you created your first op mode! The op mode currently does not do much, but you will eventually modify it to make it more useful.



Note: When you create an OnBot op mode, you create a .java file that is stored on the Robot Controller. You can access your saved op modes using the project browser on the left side of the screen. You can also organize your saved op modes by right mouse clicking on the project browser to display a list of options to create, edit or delete files and folders.

Also, note that the OnBot Java editor automatically saves your op mode as you are editing it, provided that you are connected to the Program & Manage server.

14.4.1. Examining the Structure of Your Op Mode

It can be helpful to think of an op mode as a list of tasks for the Robot Controller to perform. For a linear op mode, the Robot Controller will process this list of tasks sequentially. Users can also use control loops (such as a while loop) to have the Robot Controller repeat (or iterate) certain tasks within a linear op mode.

If you think about an op mode as a list of instructions for the robot, this set of instructions that you created will be executed by the robot whenever a team member selects the op mode called “MyFIRSTJavaOpMode” from the list of available op modes for this Robot Controller.

Let’s look at the structure of your newly created op mode. Here’s a copy of the op mode text (minus some comments, the package definition, and some import package statements):

```
@TeleOp


public class MyFIRSTJavaOpMode extends LinearOpMode {

   private Gyroscope imu;

   private DcMotor motorTest;

   private DigitalChannel digitalTouch;

   private DistanceSensor sensorColorRange;
```

```
    private Servo servoTest;



    @Override
    public void runOpMode() {
        imu = hardwareMap.get(Gyroscope.class, "imu");
        motorTest = hardwareMap.get(DcMotor.class, "motorTest");
        digitalTouch = hardwareMap.get(DigitalChannel.class, "digitalTouch");
        sensorColorRange = hardwareMap.get(DistanceSensor.class, "sensorColorRange");
        servoTest = hardwareMap.get(Servo.class, "servoTest");

        telemetry.addData("Status", "Initialized");
        telemetry.update();
        // Wait for the game to start (driver presses PLAY)
        waitForStart();

        // run until the end of the match (driver presses STOP)
        while (opModeIsActive()) {
            telemetry.addData("Status", "Running");
            telemetry.update();

        }
    }
}
```

At the start of the op mode there is an annotation that occurs before the class definition. This annotation states that this is a tele-operated (i.e., driver controlled) op mode:

```
@TeleOp
```

If you wanted to change this op mode to an autonomous op mode, you would replace the “@TeleOp” with an “@Autonomous” annotation instead.

You can see from the sample code that an op mode is defined as a Java class. In this example, the op mode name is called “MyFIRSTJavaOpMode” and it inherits characteristics from the LinearOpMode class.

```
public class MyFIRSTJavaOpMode extends LinearOpMode {
```

You can also see that the OnBot Java editor created five private member variables for this op mode. These variables will hold references to the five configured devices that the OnBot Java editor detected in the configuration file of your Robot Controller.

```
    private Gyroscope imu;
    private DcMotor motorTest;
    private DigitalChannel digitalTouch;
    private DistanceSensor sensorColorRange;
    private Servo servoTest;
```

Next, there is an overridden method called runOpMode. Every op mode of type LinearOpMode must implement this method. This method gets called when a user selects and runs the op mode.

```
    @Override
    public void runOpMode() {
```

At the start of the runOpMode method, the op mode uses an object named hardwareMap to get references to the hardware devices that are listed in the Robot Controller's configuration file:

```
        imu = hardwareMap.get(Gyroscope.class, "imu");
        motorTest = hardwareMap.get(DcMotor.class, "motorTest");
        digitalTouch = hardwareMap.get(DigitalChannel.class, "digitalTouch");
        sensorColorRange = hardwareMap.get(DistanceSensor.class, "sensorColorRange");
        servoTest = hardwareMap.get(Servo.class, "servoTest");
```

The hardwareMap object is available to use in the runOpMode method. It is an object of type HardwareMap class.

> **Note:** When you attempt to retrieve a reference to a specific device in your op mode, the name that you specify as the second argument of the HardwareMap.get method must match the name used to define the device in your configuration file. For example, if you created a configuration file that had a DC motor named "motorTest", then you must use this same name (it is case sensitive) to retrieve this motor from the hardwareMap object. If the names do not match, the op mode will throw an exception indicating that it cannot find the device.

In the next few statements of the example, the op mode prompts the user to push the start button to continue. It uses another object that is available in the runOpMode method. This object is called telemetry and the op mode uses the addData method to add a message to be sent to the Driver Station. The op mode then calls the update method to send the message to the Driver Station. Then it calls the waitForStart method, to wait until the user pushes the start button on the driver station to begin the op mode run.

```
        telemetry.addData("Status", "Initialized");
        telemetry.update();
```

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

> **Note:** All linear op modes should have a waitForStart statement to ensure that the robot will not begin executing the op mode until the driver pushes the start button.

After a start command has been received, the op mode enters a while loop and keeps iterating in this loop until the op mode is no longer active (i.e., until the user pushes the stop button on the Driver Station):

```
// run until the end of the match (driver presses STOP)
while (opModeIsActive()) {
   telemetry.addData("Status", "Running");
   telemetry.update();


}
```

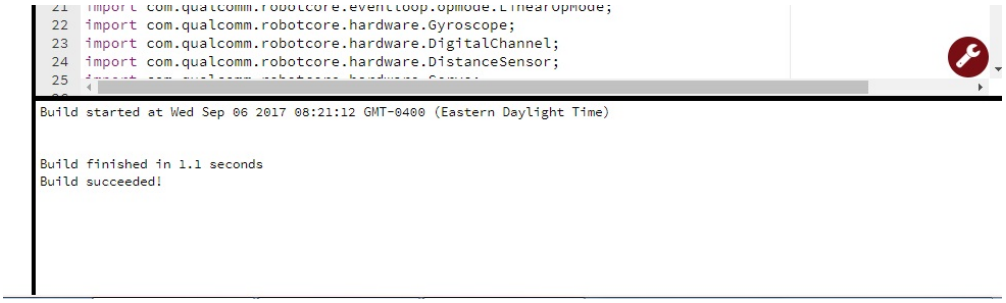
As the op mode iterates in the while loop, it will continue to send telemetry messages with the index of “Status” and the message of “Running” to be displayed on the Driver Station.

14.5. *Building Your Op Mode*

When you create or edit an op mode the OnBot Java editor will auto-save the .java file to the file system of the Robot Controller. However, before you can execute your changes on the Robot Controller, you must first build the op mode and convert it from a Java text file to a binary that can be loaded dynamically into the FTC Robot Controller app.

If you are satisfied with your op mode and are ready to build, press the Build button (which is the button with the wrench symbol, see image below) to start the build process. Note that the build process will build **all of the .java files** on your Robot Controller.

You should see messages appear in the message pane, which is located in the lower right hand side of the window. If your build was successful, you should see a “Build succeeded!” message in the message pane.



Once you have built the binary files with your updated op modes, they are ready to run on the Robot Controller. Before we run our example op mode, let's see what happens if a problem occurs during the build process.

14.6. *Troubleshooting Build Messages*

In the previous section, the build process went smoothly. Let's modify your op mode slightly to cause an error in the build process.

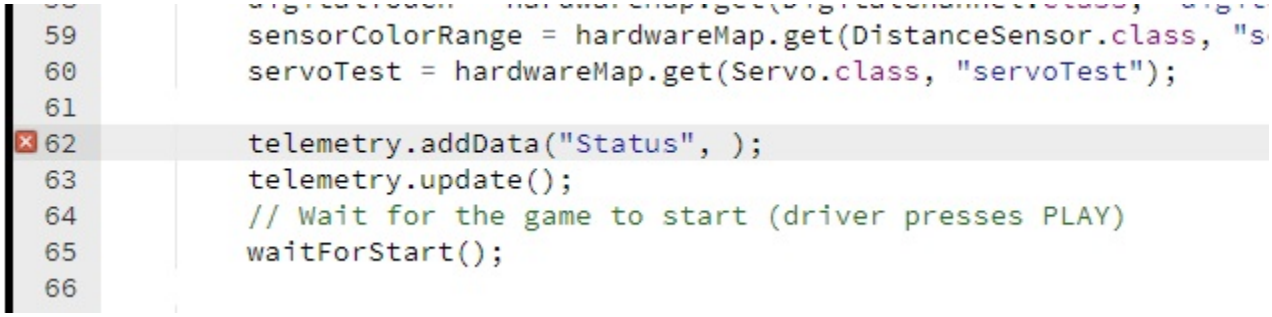
In the editing pane of the OnBot Java window, look for the line that reads "private Servo servoTest;". This should appear somewhere near the beginning of your op mode class definition. Change the word "Servo" to the word "Zervo":

```
private Zervo servoTest;
```

Also, let's modify the telemetry statement that informs the user that the op mode has been initialized, and let's remove one of the two arguments so that the statement looks like this:

```
telemetry.addData("Status",);
```

Note that when you eliminate the second argument, a little "x" should appear next to the line with the modified addData statement. This "x" indicates that there is a syntax error in the statement.



After you have modified your op mode, you can press the build button and see what error messages appear.

```
Build started at Wed Sep 06 2017 09:03:41 GMT-0400 (Eastern Daylight Time)
org/firstinspires/ftc/teamcode/MyFIRSTJavaOpMode.java line 62, column 37: ERROR: illegal start of expression


Build finished in 0.2 seconds
Build FAILED!
```

When you first attempt to build the op mode, you should get an “illegal start of expression error”. This is because the addData method is missing its second argument. The OnBot Java system also directs you to the file that has the error, and the location within the file where the error occurs.

In this example, the problem file is called “org/firstinspires/ftc/teamcode/MyFIRSTJavaOpMode.java” and the error occurs at line 62, column 37. It is important to note that the build process builds all of the .java files on the Robot Controller. If there is an error in a different file (one that you are not currently editing) you will need to look at the file name to determine which file is causing the problem.

Let’s restore this statement back to its original, correct form:

```
telemetry.addData("Status", "Initialized");
```

After you have corrected the addData statement, push the build button again to see what happens. The OnBot Java system should complain that it cannot find the symbol “Zervo” in a source file called “org/firstinspires/ftc/teamcode/MyFIRSTJavaOpMode.java” at line 51, column 13.

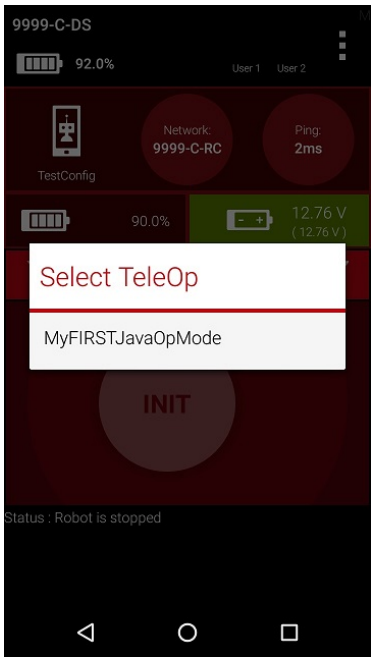
```
Build started at Wed Sep 06 2017 09:10:46 GMT-0400 (Eastern Daylight Time)
org/firstinspires/ftc/teamcode/MyFIRSTJavaOpMode.java line 51, column 13: ERROR: cannot find symbol
  symbol:   class Zervo
  location: class org.firstinspires.ftc.teamcode.MyFIRSTJavaOpMode


Build finished in 0.4 seconds
Build FAILED!
```

You should restore the statement back to its original form and then push the build button and verify that the op mode gets built properly.

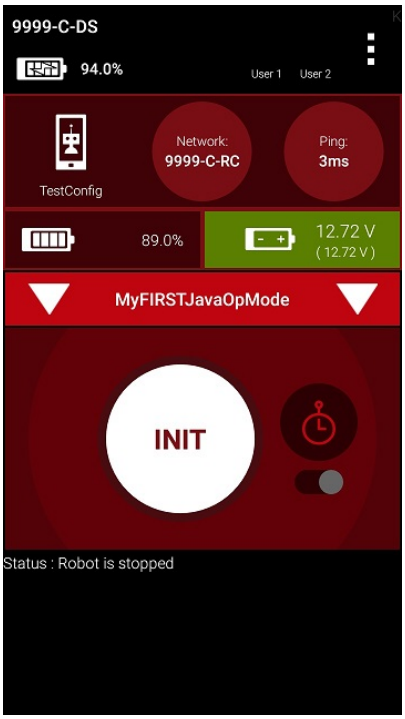
```
private Servo servoTest;
```

14.7. *Running Your Op Mode*

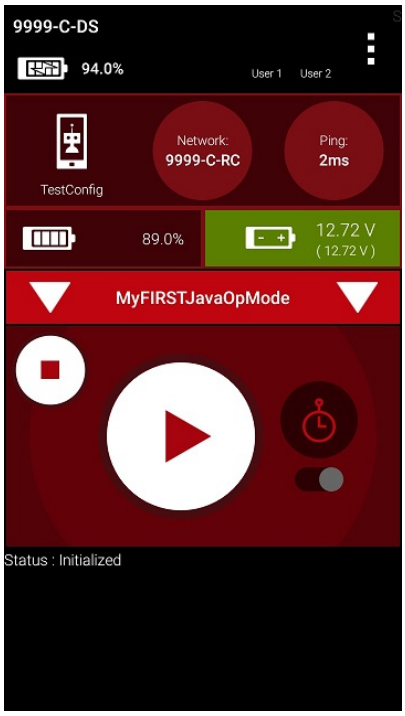
If you successfully rebuilt your op mode, you are ready to run the op mode. Verify that the Driver Station is still connected to the Robot Controller. Since you designated that your example op mode is a tele-operated op mode, it will be listed as a “TeleOp” op mode.
On the Driver Station, use the “TeleOp” dropdown list control to display the list of available op modes. Select your op mode (“MyFIRSTJavaOpMode”) from the list.



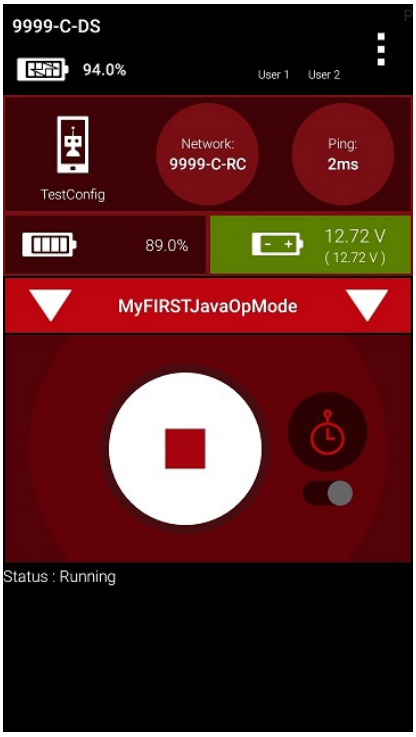
Press the INIT button to initialize the op mode.



The op mode will execute the statements in the runOpMode method up to the waitForStart statement. It will then wait until you press the start button (which is represented by the triangular shaped symbol) to continue.



Once you press the start button, the op mode will continue to iterate and send the “Status: Running” message to the Driver Station. To stop the op mode, press the square-shaped stop button.



Congratulations! You ran your first java op mode!

14.8. *Modifying Your Op Mode to Control a Motor*

Let's modify your op mode to control the DC motor that you connected and configured for your REV Expansion Hub. Modify the code for the program loop so that it looks like the following:

```
// run until the end of the match (driver presses STOP)
double tgtPower = 0;
while (opModeIsActive()) {
    tgtPower = -this.gamepad1.left_stick_y;
    motorTest.setPower(tgtPower);
    telemetry.addData("Target Power", tgtPower);
    telemetry.addData("Motor Power", motorTest.getPower());
    telemetry.addData("Status", "Running");
    telemetry.update();

}
```

If you look at the code that was added, you will see that we defined a new variable called target power before we enter the while loop.

```
double tgtPower = 0;
```

At the start of the while loop we set the variable tgtPower equal to the negative value of the gamepad1's left joystick:

```
tgtPower = -this.gamepad1.left_stick_y;
```

The object gamepad1 is available for you to access in the runOpMode method. It represents the state of gamepad #1 on your Driver Station. Note that for the F310 gamepads that are used during the competition, the Y value of a joystick ranges from -1, when a joystick is in its topmost position, to +1, when a joystick is in its bottommost position. In the example code above, you negate the left_stick_y value so that pushing the left joystick forward will result in a positive power being applied to the motor. Note that in this example, the notion of forwards and backwards for the motor is arbitrary. However, the concept of negating the joystick y value can be very useful in practice.

The next set of statements sets the power of motorTest to the value represented by the variable tgtPower. The values for target power and actual motor power are then added to the set of data that will be sent via the telemetry mechanism to the Driver Station.

```
tgtPower = -this.gamepad1.left_stick_y;

motorTest.setPower(tgtPower);

telemetry.addData("Target Power", tgtPower);

telemetry.addData("Motor Power", motorTest.getPower());
```

After you have modified your op mode to include these new statements, press the build button and verify that the op mode was built successfully.

14.9. *Running Your Op Mode with a Gamepad Connected*

Your op mode takes input from a gamepad and uses this input to control a DC motor. To run your op mode, you will need to connect a Logitech F310 gamepad to the Driver Station.
Before you connect your gamepad to the phone, verify that the switch on the bottom of the gamepad is set to the “X” position.

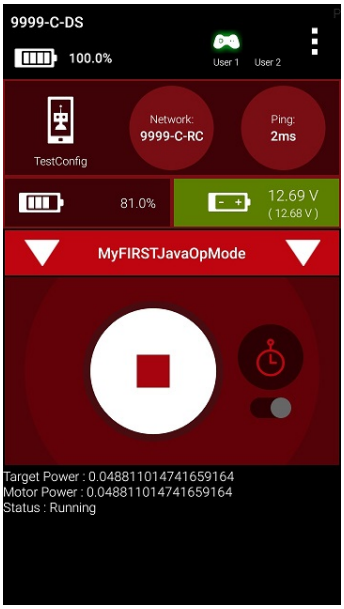
Connect the gamepad to the Driver Station using the Micro USB OTG adapter cable.

Your example op mode is looking for input from the gamepad designated as the user or driver #1. Press the Start button and the A button simultaneously on the Logictech F310 controller to designate your gamepad as user #1. Note that pushing the Start button and the B button simultaneously would designate the gamepad as user #2.

If you successfully designated the gamepad to be user #1, you should see a little gamepad icon above the text “User 1” in the upper right hand corner of the Driver Station Screen. Whenever there is activity on gamepad #1, the little icon should be highlighted in green. If the icon is missing or if it does not highlight in green when you use your gamepad, then there is a problem with the connection to the gamepad.

Select, initialize and run your “MyFIRSTJavaOpMode” op mode. It is important to note that whenever you rebuild an op mode, you must stop the current op mode run and then restart it before the changes that you just built take effect.

If you configured your gamepad properly, then the left joystick should control the motion of the motor. As you run your op mode, be careful and make sure you do not get anything caught in the turning motor. Note that the User #1 gamepad icon should highlight green each time you move the joystick. Also note that the target power and actual motor power values should be displayed in the telemetry area on the Driver Station.



15. Controlling a Servo (OnBot Java)

In this section, you will modify your op mode to control a servo motor with the buttons of the gamepad.

15.1. *What is a Servo Motor?*

A servo motor is a special type of motor. A servo motor is designed for precise motion. A typical servo motor has a limited range of motion.

In the figure below, “standard scale” 180-degree servo is shown. This type of servo is popular with hobbyists and with FIRST Tech Challenge teams. This servo motor can rotate its shaft through a range of 180 degrees. Using an electronic module known as a servo controller you can write an op mode that will move a servo motor to a specific position. Once the motor reaches this target position, it will hold the position, even if external forces are applied to the shaft of the servo.

Servo motors are useful when you want to do precise movements (for example, sweep an area with a sensor to look for a target or move the control surfaces on a remotely controlled airplane).

15.2. *Modifying Your Op Mode to Control a Servo*

Let's modify your op mode to add the logic required to control a servo motor. For this example, you will use the buttons on the Logitech F310 gamepad to control the position of the servo motor.

With a typical servo, you can specify a target position for the servo. The servo will turn its motor shaft to move to the target position, and then maintain that position, even if moderate forces are applied to try and disturb its position.

For the FIRST Tech Challenge control system, you can specify a target position that ranges from 0 to 1 for a servo. A target position of 0 corresponds to zero degrees of rotation and a target position of 1 corresponds to 180 degrees of rotation for a typical servo motor.

In this example, you will use the colored buttons on the right side of the F310 controller to control the position of the servo. Initially, the op mode will move the servo to the midway position (90 degrees of its 180-degree range). Pushing the yellow “Y” button will move the servo to the zero-degree position. Pushing the blue “X” button or the red “B” button will move the servo to the 90-degree position. Pushing the green “A” button will move the servo to the 180-degree position.

Modify your op mode to add the following code:

```
// run until the end of the match (driver presses STOP)
double tgtPower = 0;
while (opModeIsActive()) {
   tgtPower = -this.gamepad1.left_stick_y;
   motorTest.setPower(tgtPower);
   // check to see if we need to move the servo.
   if(gamepad1.y) {
      // move to 0 degrees.
      servoTest.setPosition(0);
   } else if (gamepad1.x || gamepad1.b) {
      // move to 90 degrees.
      servoTest.setPosition(0.5);
   } else if (gamepad1.a) {
      // move to 180 degrees.
      servoTest.setPosition(1);
   }
   telemetry.addData("Servo Position", servoTest.getPosition());
   telemetry.addData("Target Power", tgtPower);
   telemetry.addData("Motor Power", motorTest.getPower());
   telemetry.addData("Status", "Running");
   telemetry.update();

}
```

This added code will check to see if any of the colored buttons on the F310 gamepad are pressed. If the Y button is pressed, it will move the servo to the 0-degree position. If either the X button or B button is pressed, it

will move the servo to the 90-degree position. If the A button is pressed, it will move the servo to the 180-degree position. The op mode will also send telemetry data on the servo position to the Driver Station.

After you have modified your op mode, you can build it and then run it. Verify that gamepad #1 is still configured and then use the colored buttons to move the position of the servo.

16. Using Sensors (OnBot Java)

16.1. *Color-Distance Sensor*

A sensor is a device that lets the Robot Controller get information about its environment. In this example, you will use a REV Robotics Color-Distance sensor to display range (distance from an object) info to the driver station.

The Color-Range sensor uses reflected light to determine the distance from the sensor to the target object. It can be used to measure close distances (up 5” or more) with reasonable accuracy. Note that at the time this document was most recently edited, the REV Color-Range sensor saturates around 2” (5cm). This means that for distances less than or equal to 2”, the sensor returns a measured distance equal to 2” or so.

Modify your op mode to add a telemetry statement that will send the distance information (in centimeters) to the Driver Station.

```
telemetry.addData("Servo Position", servoTest.getPosition());
telemetry.addData("Target Power", tgtPower);
telemetry.addData("Motor Power", motorTest.getPower());
telemetry.addData("Distance (cm)", sensorColorRange.getDistance(DistanceUnit.CM));
telemetry.addData("Status", "Running");
telemetry.update();
```

After you have modified your op mode, push the build button, then run the op mode to verify that it now displays distance on your Driver Station. Note that if the distance reads “NaN” (short for “Not a Number”) it probably means that your sensor is too far from the target (zero reflection). Also note that the sensor saturates at around 5 cm.

16.2. *Touch Sensor*

The REV Robotics Touch Sensor can be connected to a digital port on the Expansion Hub. The Touch Sensor is HIGH (returns TRUE) when it is not pressed. It is pulled LOW (returns FALSE) when it is pressed.

The Expansion Hub digital ports contain two digital pins per port. When you use a 4-wire JST cable to connect a REV Robotics Touch sensor to an Expansion Hub digital port, the Touch Sensor is wired to the second of the two digital pins within the port. The first digital pin of the 4-wire cable remains disconnected.

For example, if you connect a Touch Sensor to the “0,1” digital port of the Expansion Hub, the Touch Sensor will be connected to the second pin (labeled “1”) of the port. The first pin (labeled “0”) will stay disconnected.

Modify the code in your op mode that occurs before the waitForStart command to set the digital channel for input mode.

```
// set digital channel to input mode.
digitalTouch.setMode(DigitalChannel.Mode.INPUT);

telemetry.addData("Status", "Initialized");
telemetry.update();
// Wait for the game to start (driver presses PLAY)
waitForStart();
```

Also, modify the code in your while loop to add an if-else statement that checks the state of the digital input channel. If the channel is LOW (false), the touch sensor button is pressed and being pulled LOW to ground. Otherwise, the touch sensor button is not pressed.

```
// is button pressed?
if (digitalTouch.getState() == false) {
    // button is pressed.
    telemetry.addData("Button", "PRESSED");
} else {
    // button is not pressed.
    telemetry.addData("Button", "NOT PRESSED");
}

telemetry.addData("Status", "Running");
telemetry.update();
```

Rebuild your op mode, then reinitialize and restart your op mode. The op mode should now display the state of the button (“PRESSED” or “NOT PRESSED”).

17. OnBot Java Reference Info

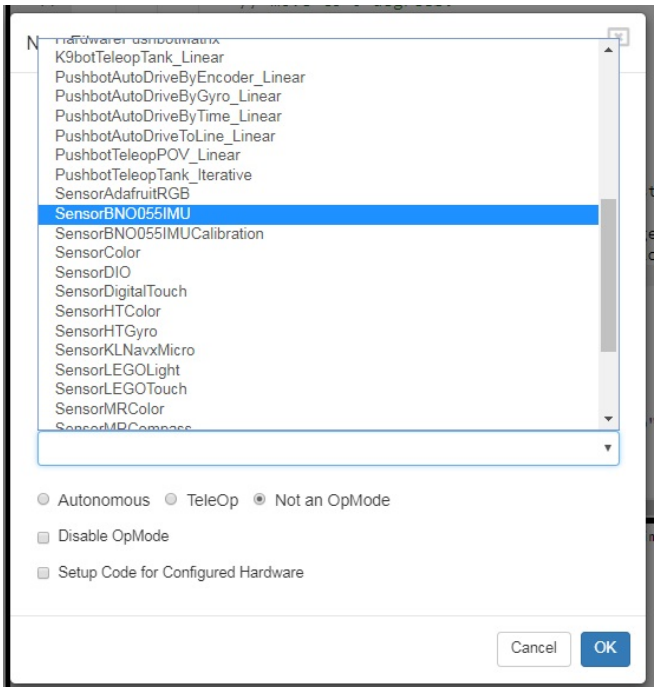
17.1. *Javadoc Reference Pages*

As you start to write more complicated op modes, you will need to use more features of the FIRST Tech Challenge software development kit (SDK). You can reference online Javadoc material that provide descriptions of the available FIRST Tech Challenge-related classes and methods, at the following web address:

http://ftctechnh.github.io/ftc_app/doc/javadoc/index.html

17.2. *Sample Op Modes*

The OnBot Java Programming Tool has several built-in example op modes that demonstrate how to do different tasks with the FIRST Tech Challenge control system. As you create a new file, you can use the Sample dropdown list control to display a list of available sample op modes or templates. The comments in these examples help explain what the program statements do.



17.3. *Technology Forum*

Registered teams can create user accounts on the FIRST Tech Challenge forum. Teams can use the forum to ask questions and receive support from the FIRST Tech Challenge community.

The technology forum can be found at the following address:

https://ftcforum.usfirst.org/forum/ftc-technology?156-FTC-Technology

17.4. *REV Robotics Expansion Hub Documentation*

REV Robotics Expansion Hub Getting Started Guide

17.5. REV Driver Hub and Control Hub Tutorial Videos

Getting Started with... - YouTube

Appendix A – Resources

Game Forum Q&A

https://ftc-qa.firstinspires.org/

Anyone may view questions and answers within the *FIRST*® Tech Challenge game Q&A forum without a password. To submit a new question, you must have a unique Q&A system user name and password for your team.

Volunteer Forum

Volunteers can request access to role specific volunteer forums by emailing FTCTrainingSupport@firstinspires.org. You will receive access to the forum thread specific to your role.

FIRST Tech Challenge Game Manuals

Part 1 and 2 - https://www.firstinspires.org/resource-library/ftc/game-and-season-info

FIRST Headquarters Pre-Event Support

Phone: 603-666-3906
Mon – Fri
8:30am – 5:00pm
Email: Firsttechchallenge@firstinspires.org

FIRST Websites

FIRST homepage – www.firstinspires.org

FIRST Tech Challenge Page – For everything *FIRST* Tech Challenge.

FIRST Tech Challenge Volunteer Resources – To access public volunteer manuals.

FIRST Tech Challenge Event Schedule – Find *FIRST* Tech Challenge events in your area.

FIRST Tech Challenge Social Media

FIRST Tech Challenge Twitter Feed - If you are on Twitter, follow the *FIRST* Tech Challenge Twitter feed for news updates.

FIRST Tech Challenge Facebook page - If you are on Facebook, follow the *FIRST* Tech Challenge page for news updates.

FIRST Tech Challenge YouTube Channel – Contains training videos, game animations, news clips, and more.

FIRST Tech Challenge Blog – Weekly articles for the *FIRST* Tech Challenge community, including outstanding volunteer recognition!

FIRST Tech Challenge Team Email Blasts – contain the most recent *FIRST* Tech Challenge news for teams.

Feedback

We strive to create support materials that are the best they can be. If you have feedback about this manual, please email firsttechchallenge@firstinspires.org. Thank you!