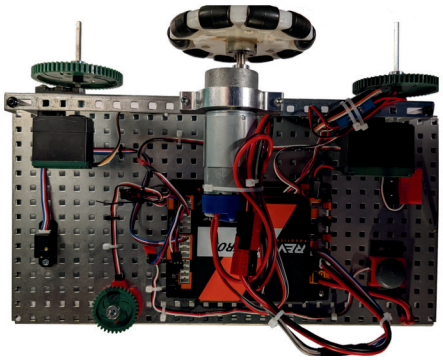
# CONTROL SYSTEMS

## ASSIGNMENT #1B


An introduction to the Rev Control Hub, the Driver Station App, and the testBoard

# REQUIRED ELEMENTS



TESTBOARD


ANDROID PHONE

BATTERY

COMPUTER

# ANDROID STUDIO

The code that is running on the Control Hub was written in a program called Android Studio, using the Java programming language. In this assignment, you will be making small modifications to this code and beginning to write your own programs to run on the testBoard.

# ASSIGNMENT #1B

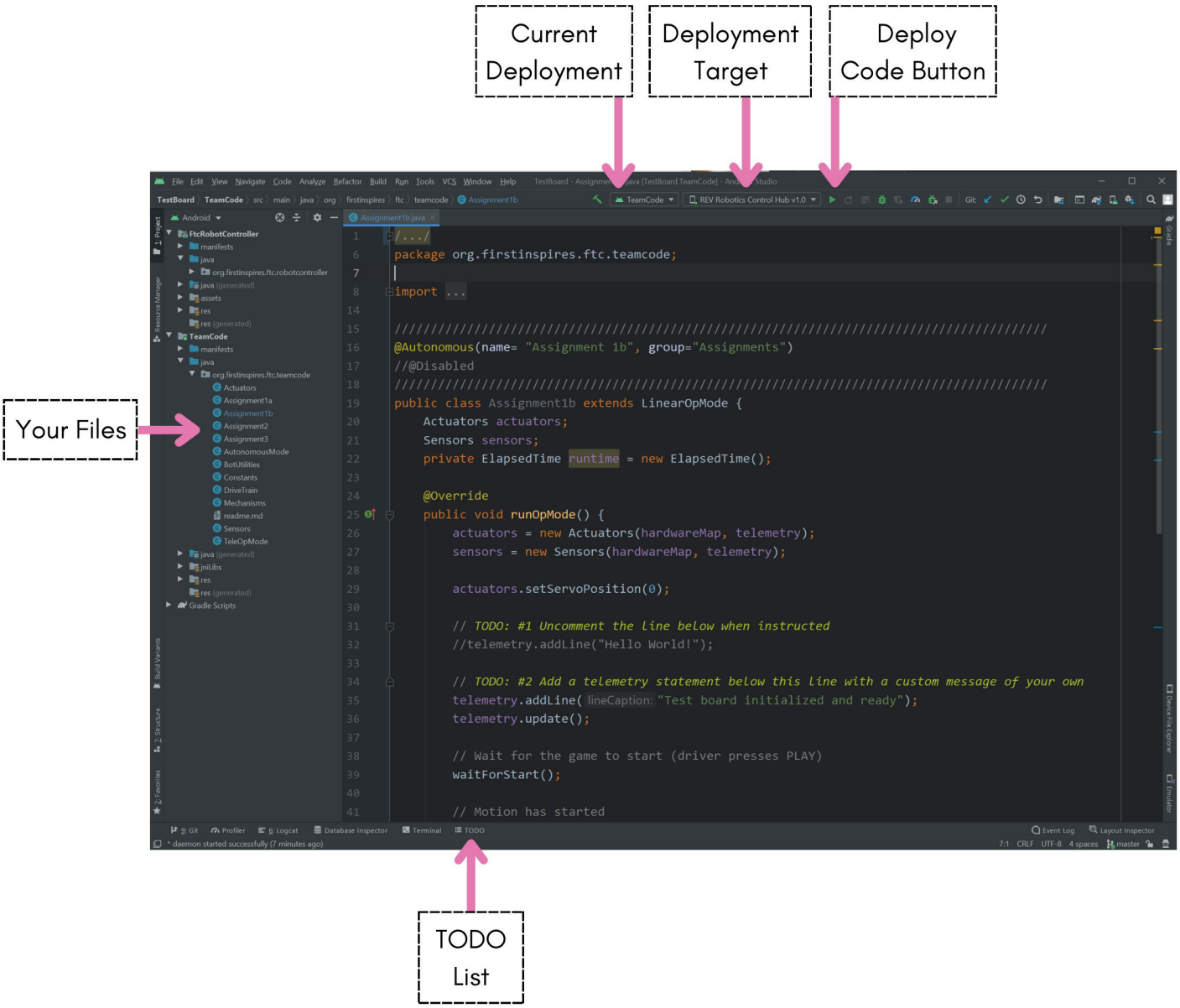
- The objective of this assignment is to get an introduction into how written code can cause changes in the output of a program.

1. Start by selecting the Assignment 1B program on the DriverStation app.
2. Turn off the timer.
3. Press the 'INIT' button
    a. Did anything happen?
4. Press the Play button.
    a. What is happening?
5. Stop the program.
6. With your hand, turn the gear connected to the servo to some other position than the one it is currently at.
7. Press the 'INIT' button
    a. Did anything happen?

Let's investigate the code behind this program.

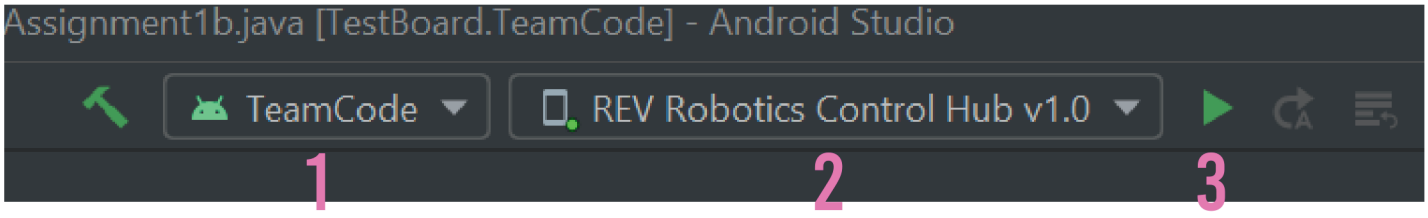
# THE INTERFACE

Android Studio is an Integrated Development Environment (IDE) that we use for writing code that can be deployed to a Control Hub. The IDE has many useful features. The screen below shows what you should see on your computers when you are first starting out.



# THE INTERFACE

In order to deploy your code, we need to verify some of the settings in your interface. At the top of the window, you should see something that looks very similar to what is shown below.




## 1. Current Deployment

This dropdown represents what specific package of code will be sent to the Control Hub. For these assignments, this should always read 'TeamCode'

## 2. Deployment Target

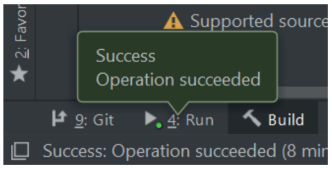
This dropdown represents where this specific package of code will be sent. For these assignments, this should always read 'REV Robotics Control Hub'.

- Occasionally, typically after disconnecting/reconnecting with the computer, this setting will be lost. Hovering over the dropdown while the device is physically connected to the computer will get it to repopulate the field.

## 3. Run Code Button

This is the button you will utilize to send your code to the Control Hub.

# THE PROCESS




The first time you deploy your code after turning on the Control Hub, the process may take upwards of two minutes. All subsequent deployments should only take a handful of seconds. However, this does not mean your program is ready to run yet! The program must then be installed and launched on the Control Hub. You will know when this is completed by looking for the success message (see above), the light on the Control Hub returns to solid green, and the phone display has reset back to normal.

# // TODO

Throughout all code files in these assignments, you will find many "to-do" instructions written into the code. A TODO is a special type of comment in the file.

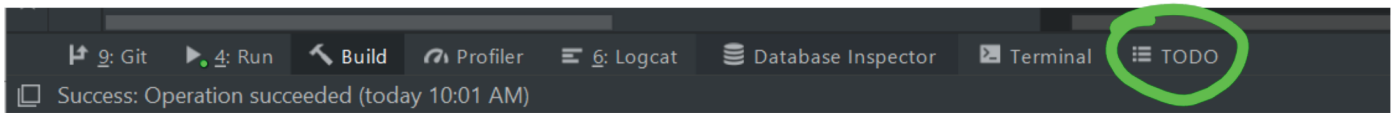
# // COMMENTS

Comments in a program are remarks, instructions, or information placed there for the user. The computer ignores all comments when running a program.

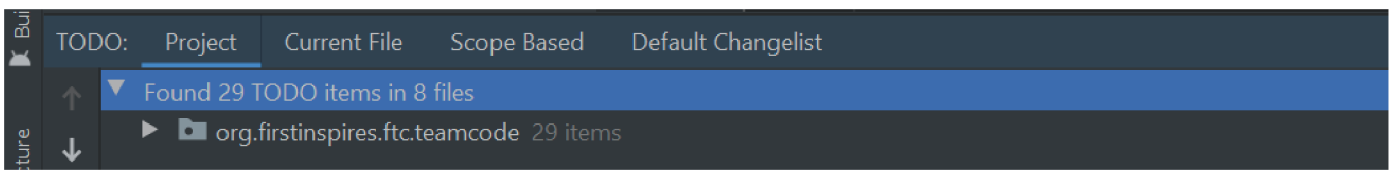
- Anything can be a comment, such as:
  - Explanations to the purpose of a section of code
  - Reminders for future work
  - Sections of code you intentionally don't want to run
- Types of comments:
  - // Starts a single-line comment
    - This comment ends automatically on the next line
  - /* Starts a multi-line comment
    - This comment will not end until the end character (*/) is written

// TODO is a special type of single line comment. It is recognized by the IDE as something you want to do.

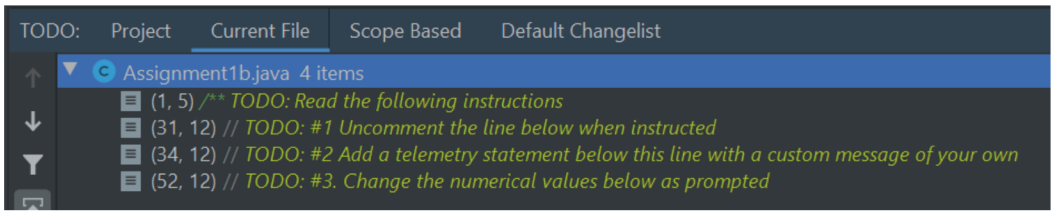
For these assignments, TODOs are used to help you identify the sections of code you need to be working on. A list of all TODOs can be found on the bottom of your window. The exact location may vary depending on the size of your screen.




Clicking on this will open all TODOs in your active project (all files in this unit).




Clicking 'Current File' will show you the TODOs for the file currently active in Android Studio. From here, you can click on individual TODOs to be taking directly to them in code.



# WHAT TODO

Before making changes to your program, run the program again and take note of what does (and does not) appear in the telemetry window on the phone.

Open the file 'Assignment1b' in Android Studio.

## TODO 1. Basic Telemetry

The first change you are going to make to your program is to **uncomment** a line of code to add a telemetry message to the program.

1. Find TODO #1
2. Uncomment the line below the TODO statement
    a. Do this by deleting the **//** in front of `telemetry.addLine("Hello World!");`
3. Deploy your code
    a. After verifying that the interface looks as described on the 'The Interface' page, press the green play button.
    b. Wait for the hub to be ready (watch the phone)
    c. Run the program again

## Q: What changed?

## TODO 2. Basic Telemetry

The second change you are going to make to your program is to **write** a line of code to add a telemetry message to the program.

1. Find TODO #2
2. Create a new line following the TODO statement. You should not delete anything.
    a. Model your code after the other code already present. The syntax of your code must be exact in order for the program to compile and run. However, the message you write can be unique.
3. Deploy your code
    a. Wait for the hub to be ready (watch the phone)
    b. Run the program again

## Q: What changed?

# WHAT TODO

For TODO #3, you will be making several changes to your code. Whenever you are prompted to modify code, assume you should redeploy and test after making the changes.

## TODO 3. Modifying Outputs

All items on the test board are controlled using code. For this section, you will be modifying and adding code to control the actuators on the testBoard.

1. Find TODO #3
2. Find the two lines of code that set power to the continuous rotation servo, the standard servo, and the testBoard motor
3. Change the numerical values in the code to another number
    a. The values you choose must be between **-1** and **1** (inclusive)
    b. Try the following types of numbers
        i. A negative number
        ii. A positive number
        iii. Zero
    c. You will need to redeploy and test each time you change a number

# QUESTIONS

1. What was the difference between using a positive or negative number when controlling the motor and/or continuous rotation servo?
2. What effect does typing in 0 as the input value have?
3. What is the difference in output between typing in 0.2 and 0.9?
4. What is the difference in function between a continuous rotation servo and the standard servo?