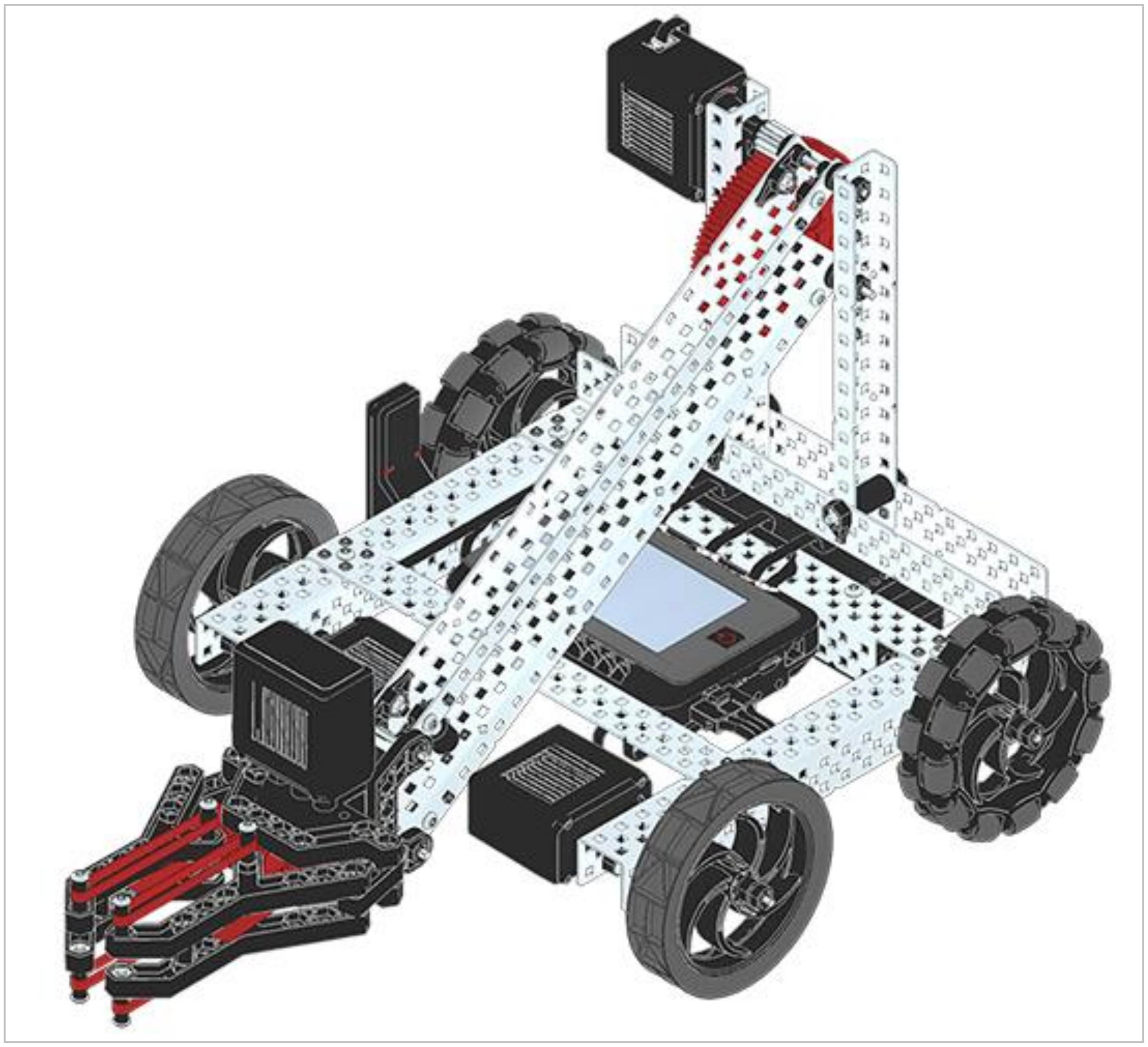
Speedy Delivery

Learn how to pick up objects and move them around with the VEX V5 Clawbot!

SPARK
SEEK

Discover new hands-on builds and programming opportunities to further your understanding of a subject matter.

The Completed Look of the Build



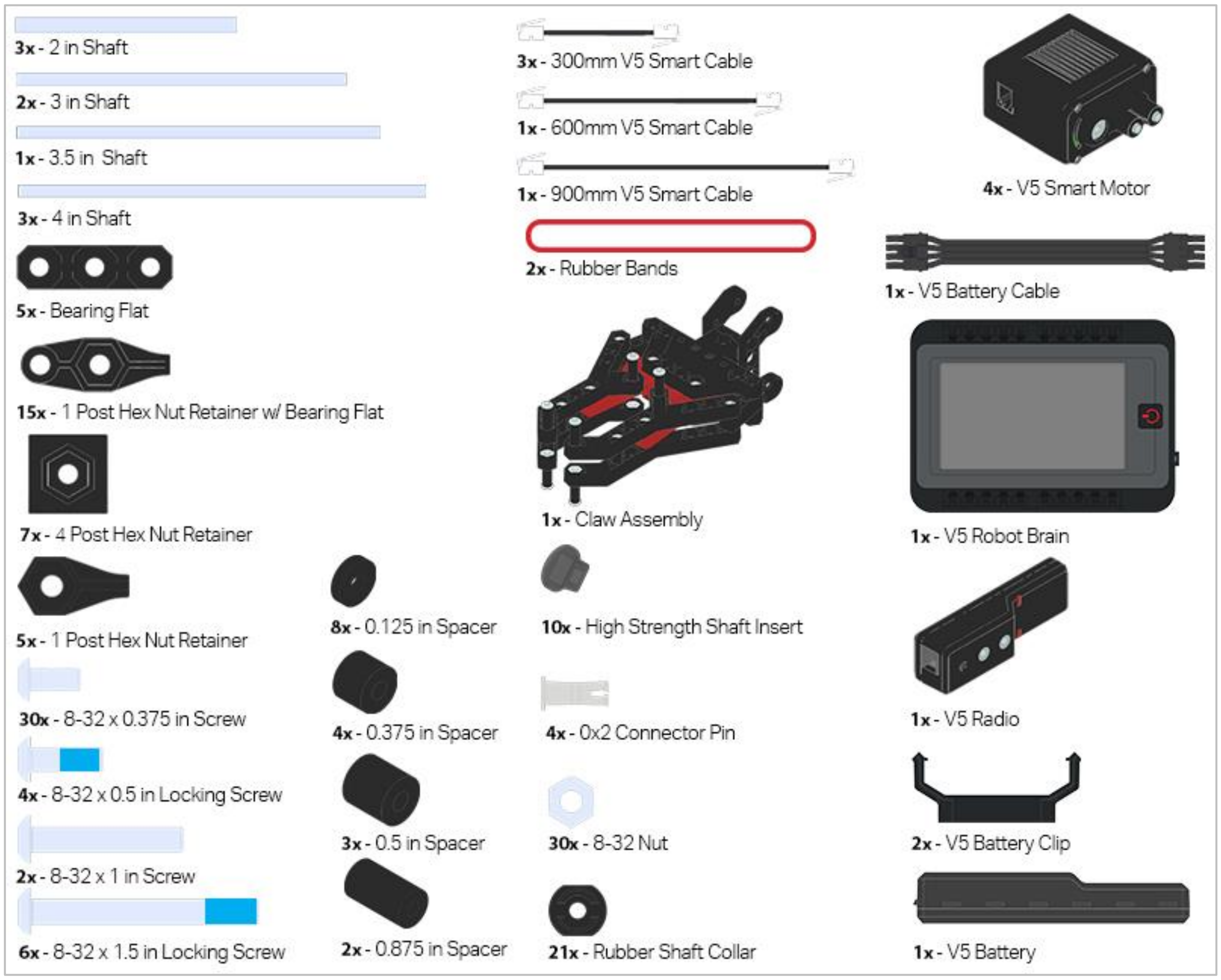
Completed VEX V5 Clawbot

The VEX V5 Clawbot is an extension of the VEX V5 Speedbot that can be programmed to move around and interact with objects.

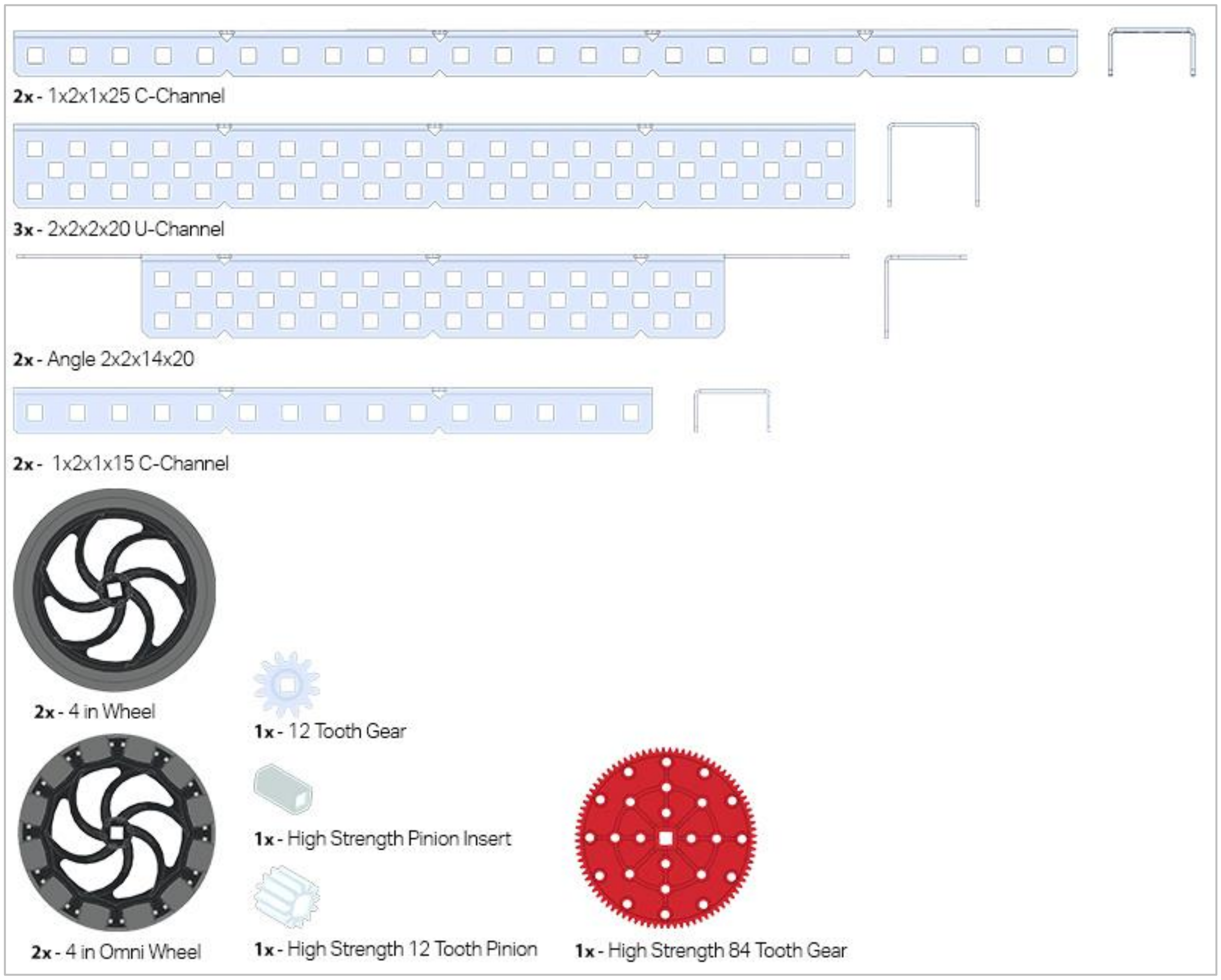
Parts Needed: Part 1

Can be built with:

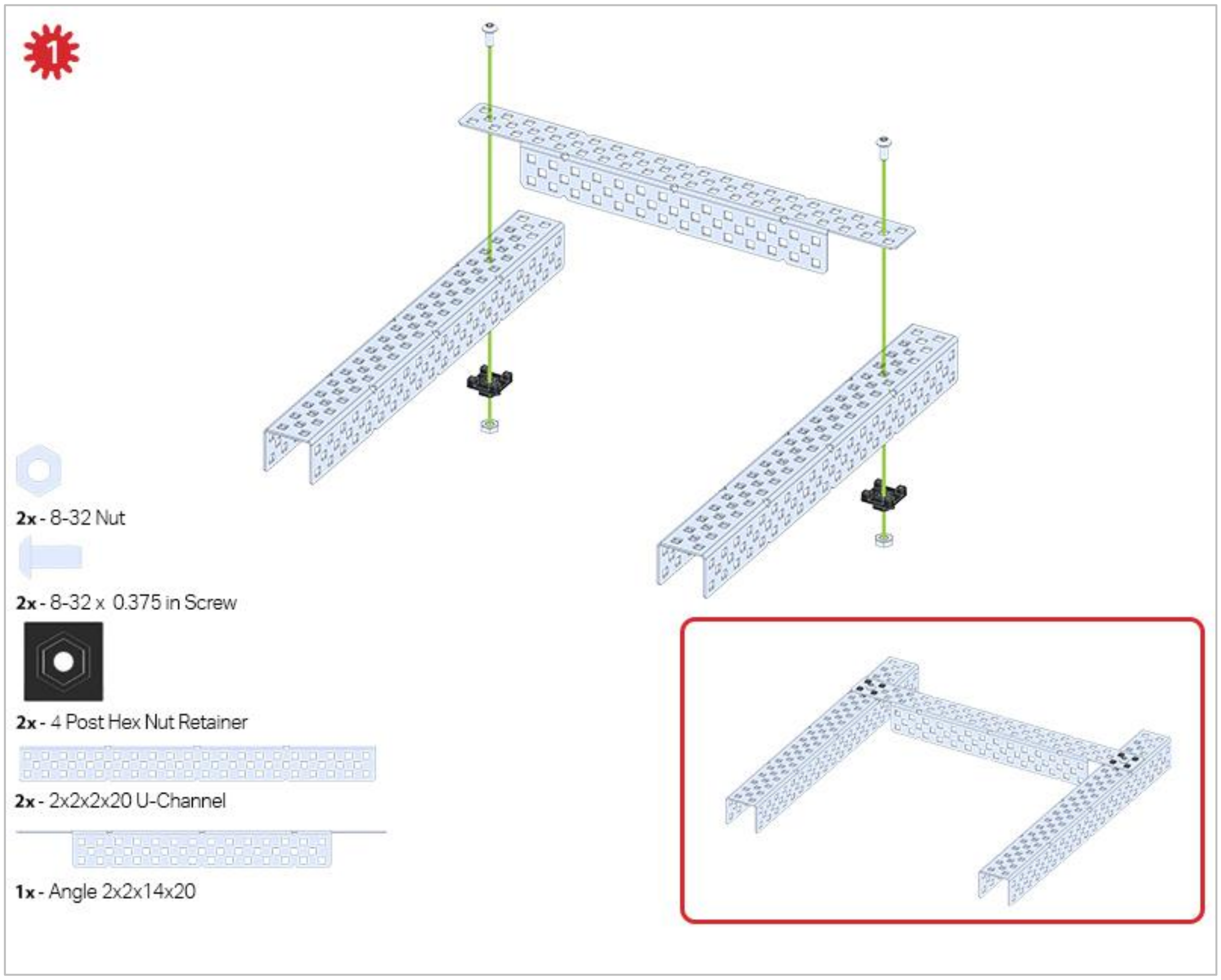
- VEX V5 Classroom Starter Kit



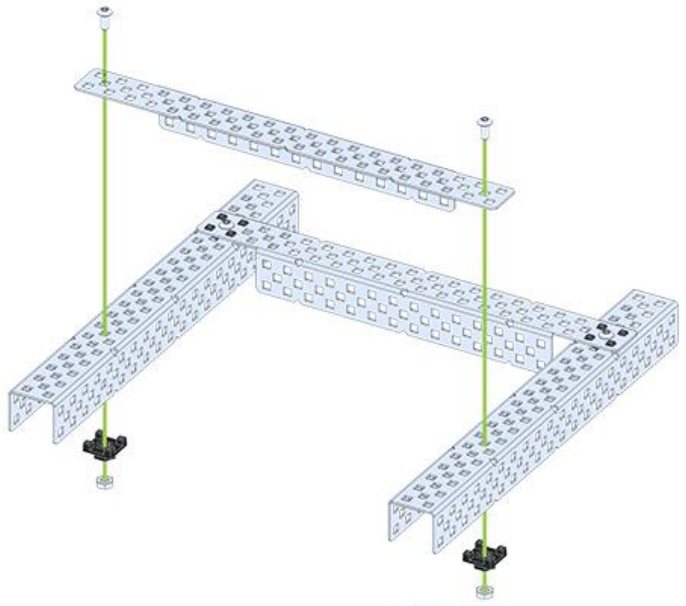
Parts Needed: Part 2



Build Instructions



2

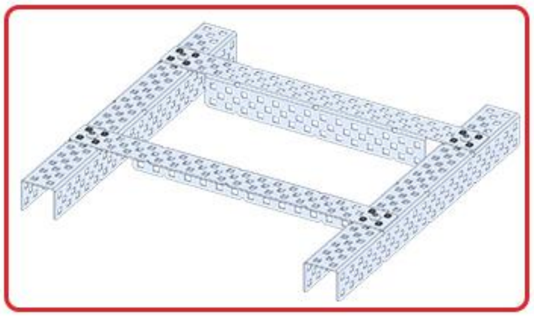


2x - 8-32 Nut

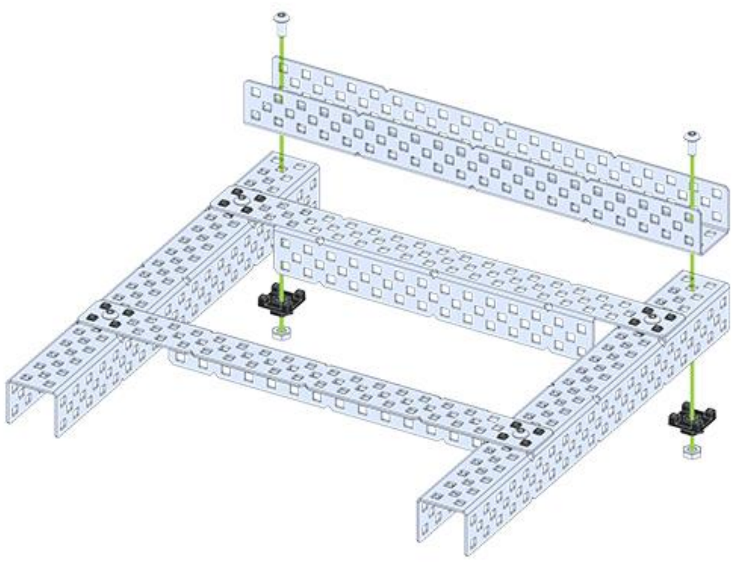
2x - 8-32 x 0.375 in Screw

2x - 4 Post Hex Nut Retainer

1x - Angle 2x2x14x20



3

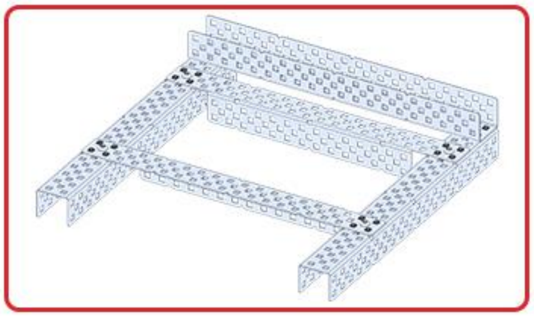


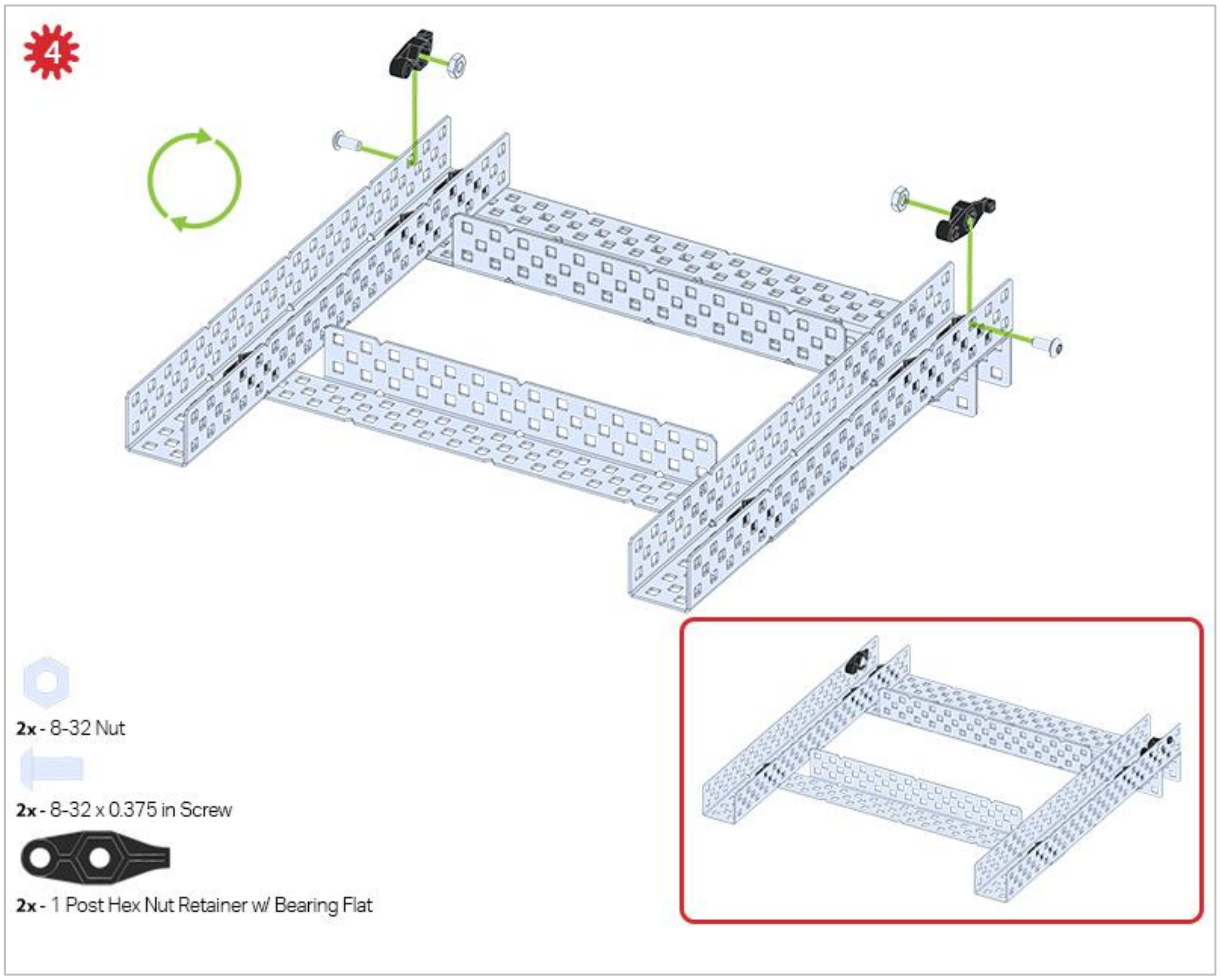
2x - 8-32 Nut

2x - 8-32 x 0.375 in Screw

2x - 4 Post Hex Nut Retainer

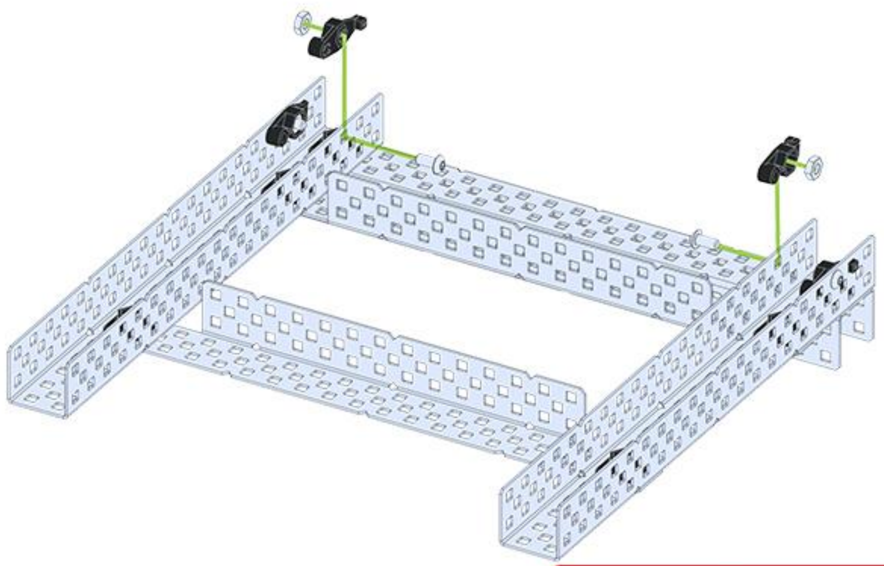
1x - 2x2x2x20 U-Channel




4
2x - 8-32 Nut
2x - 8-32 x 0.375 in Screw
2x - 1 Post Hex Nut Retainer w/ Bearing Flat

The green icon indicates that the build needs to be flipped over (upside down).

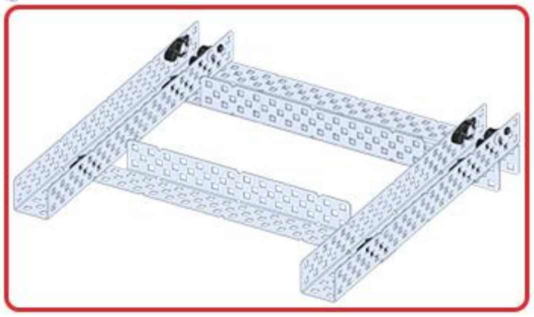
5

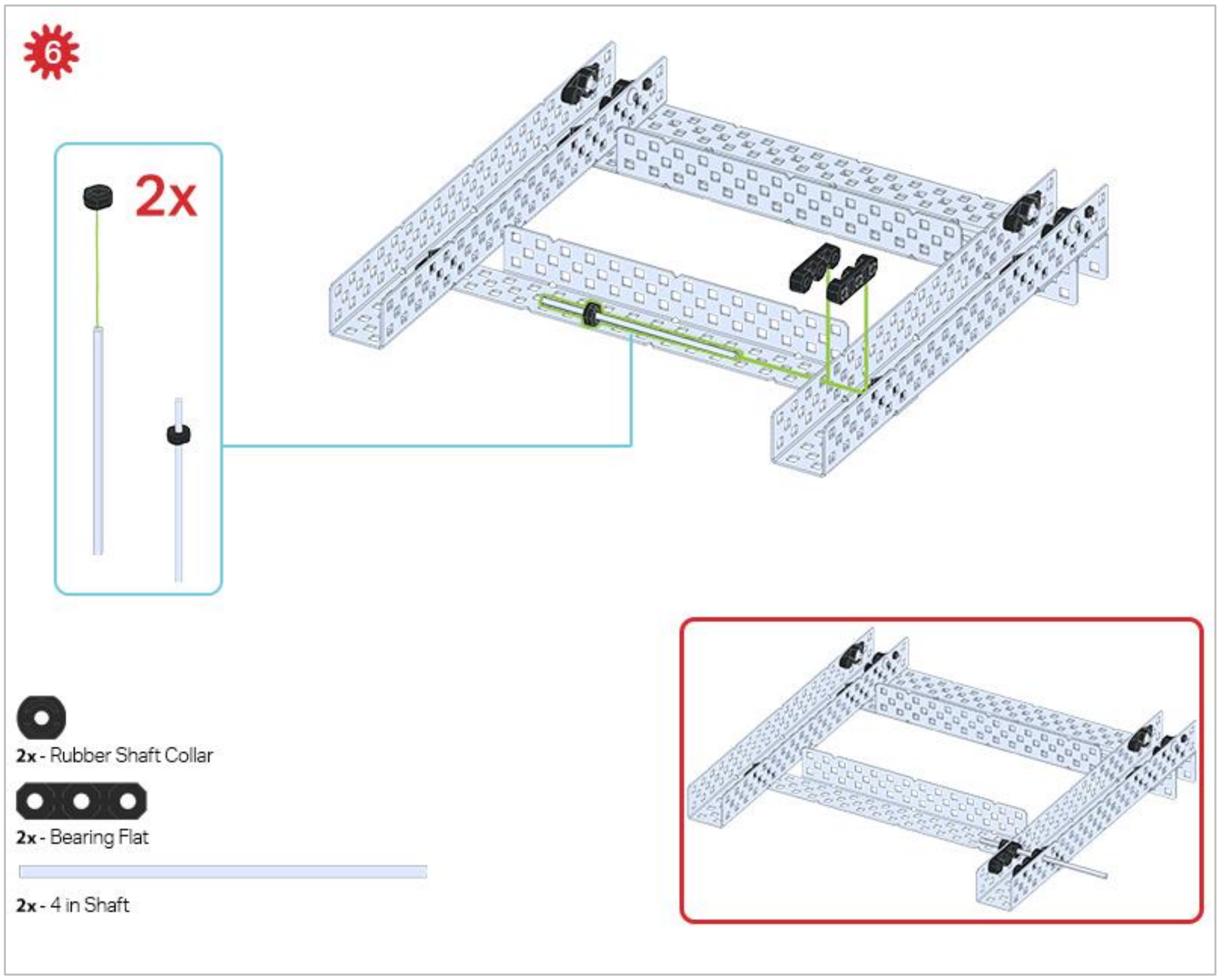


2x - 8-32 Nut

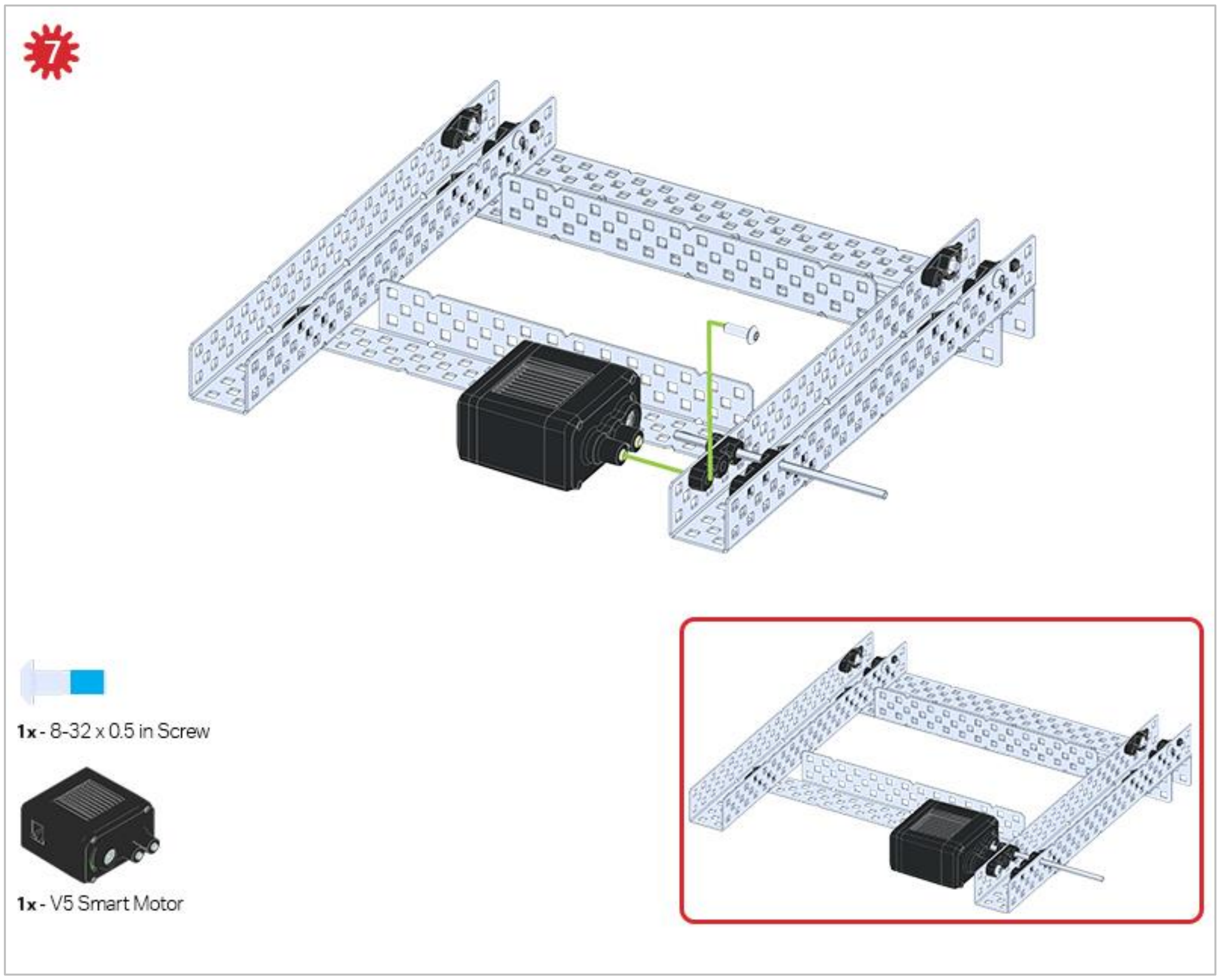
2x - 8-32 x 0.375 in Screw

2x - 1 Post Hex Nut Retainer w/ Bearing Flat



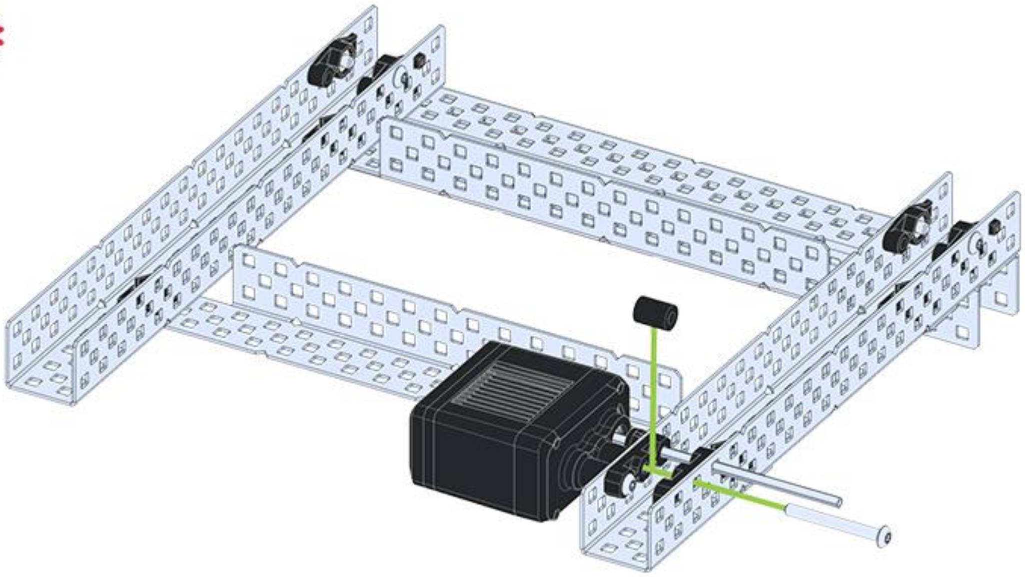


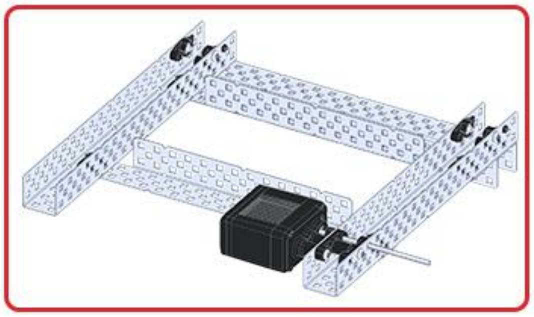
Only one of the two sub-assemblies made in this step is used right now. The other will be used later in step 9.



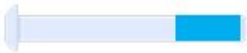
Make sure your Smart Motors are oriented in the correct direction (screw holes facing the outside of the build and the shaft hole towards the inside).

8

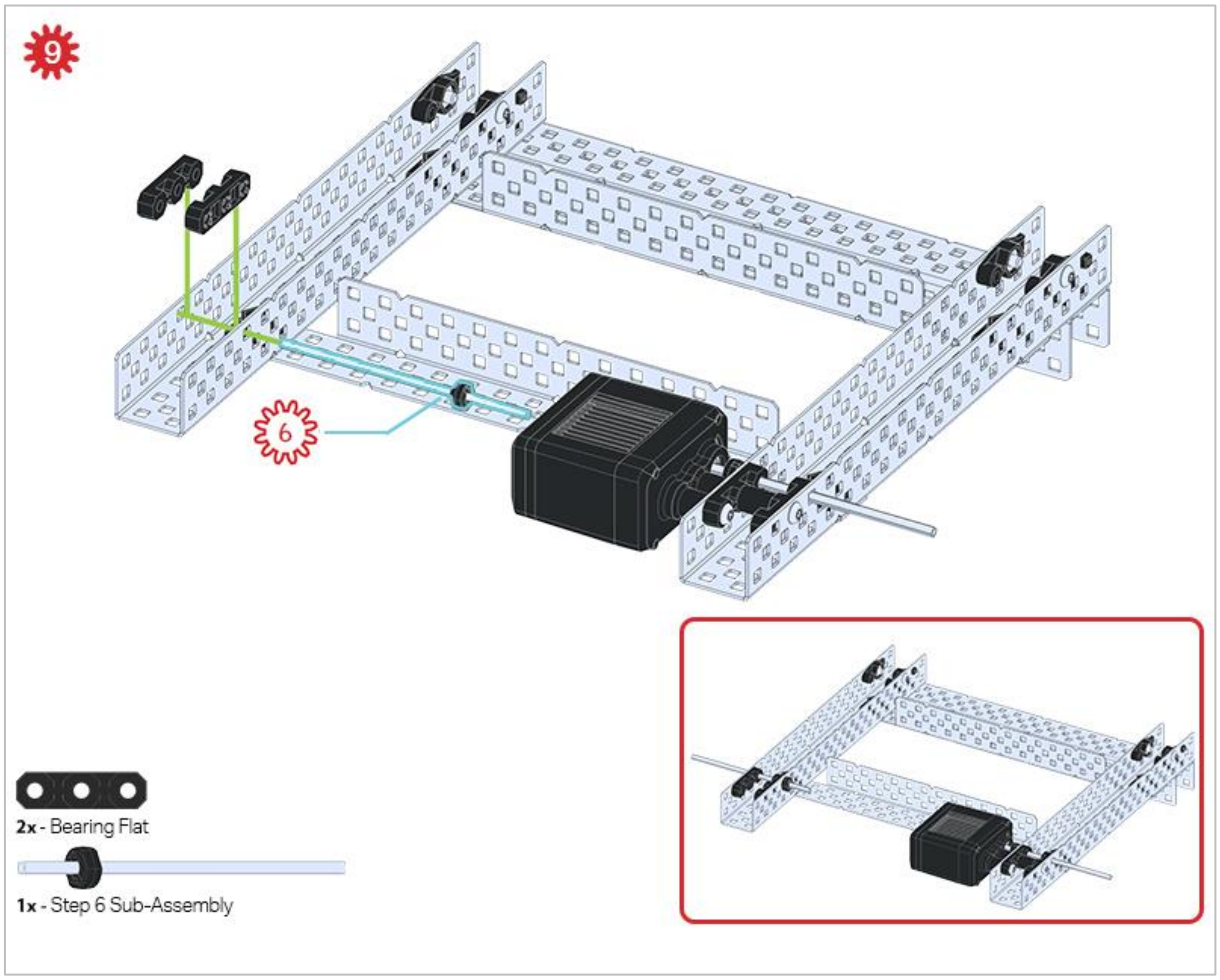


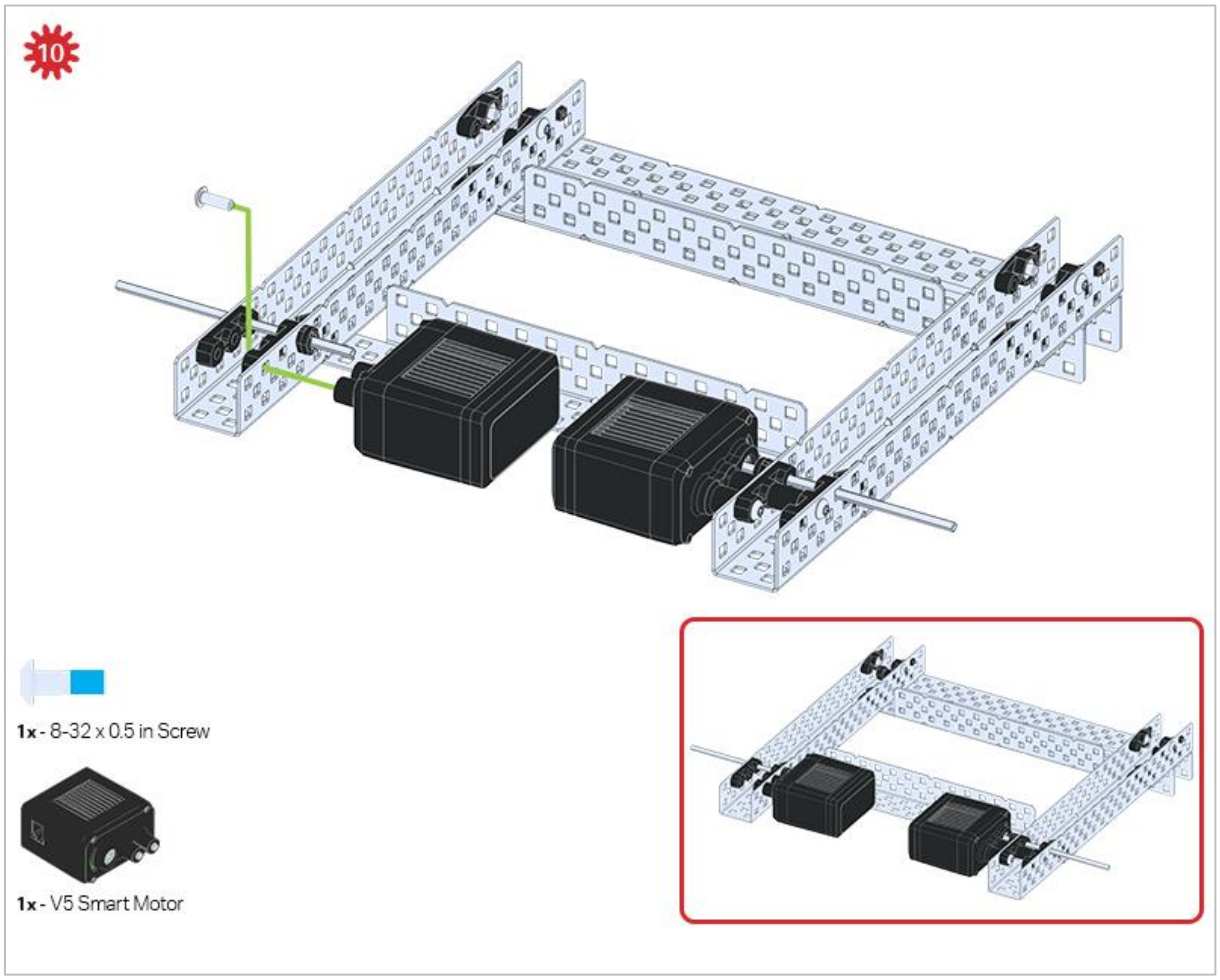


1x - 0.5 in Spacer



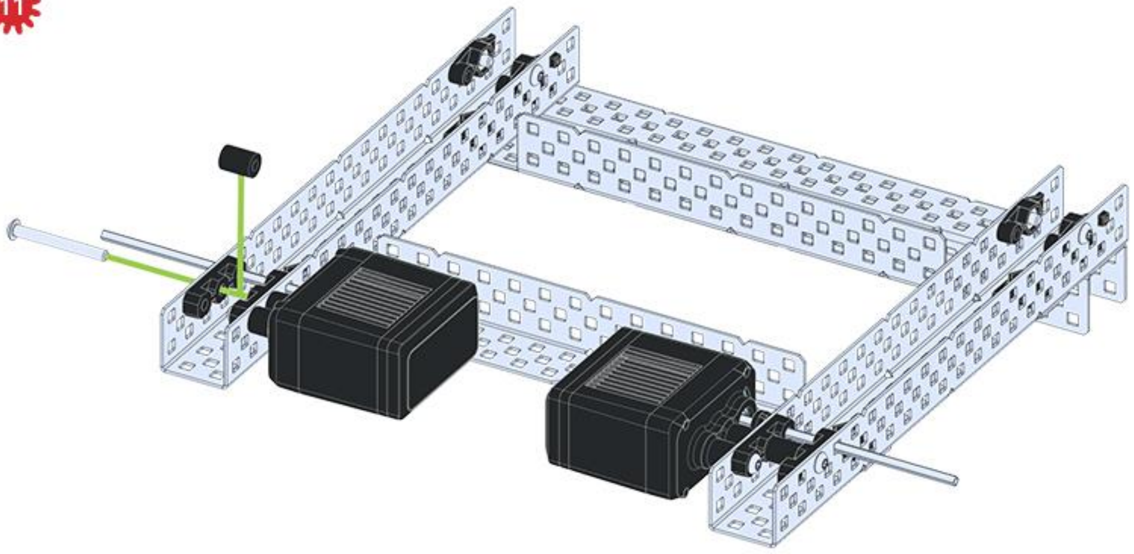
1x - 8-32 x 1.5 in Screw


9
6
2x - Bearing Flat
1x - Step 6 Sub-Assembly



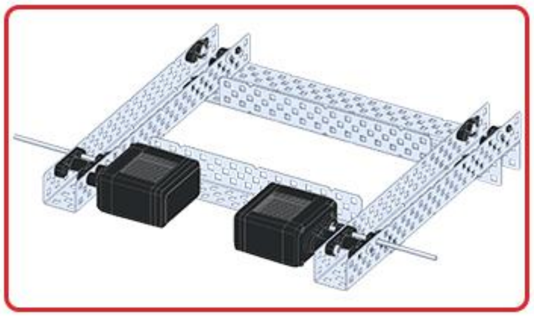
Make sure your Smart Motors are oriented in the correct direction (screw holes facing the outside of the build and the shaft hole towards the inside).

11

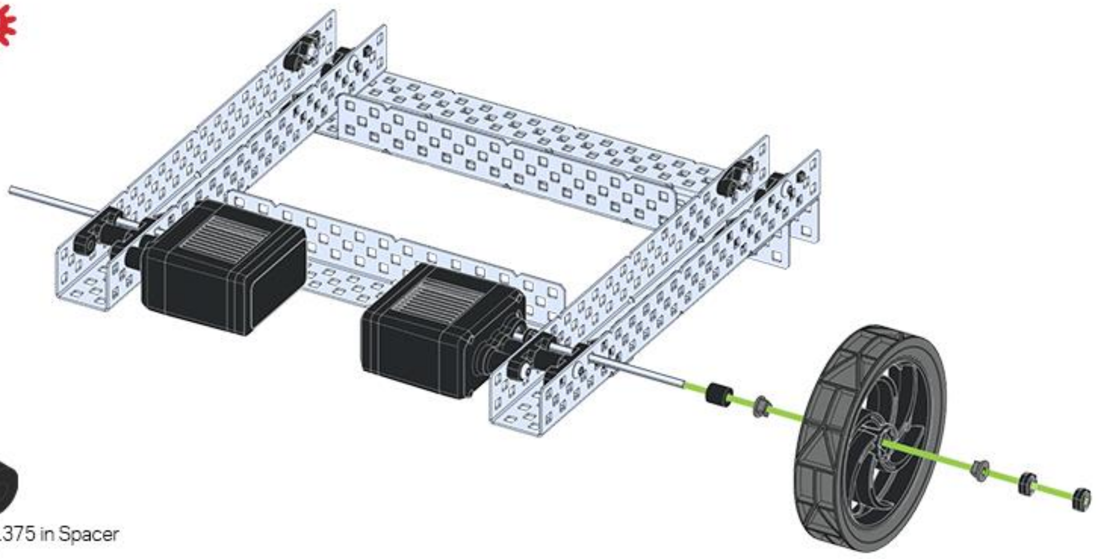


1x - 0.5 in Spacer

1x - 8-32 x 1.5 in Screw



12

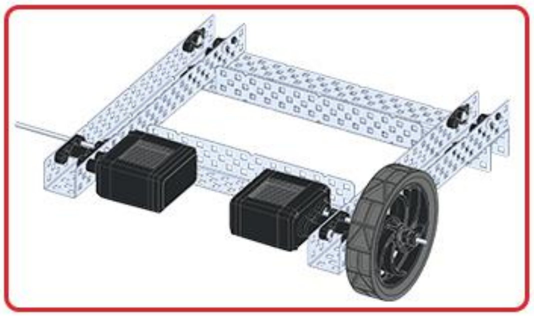


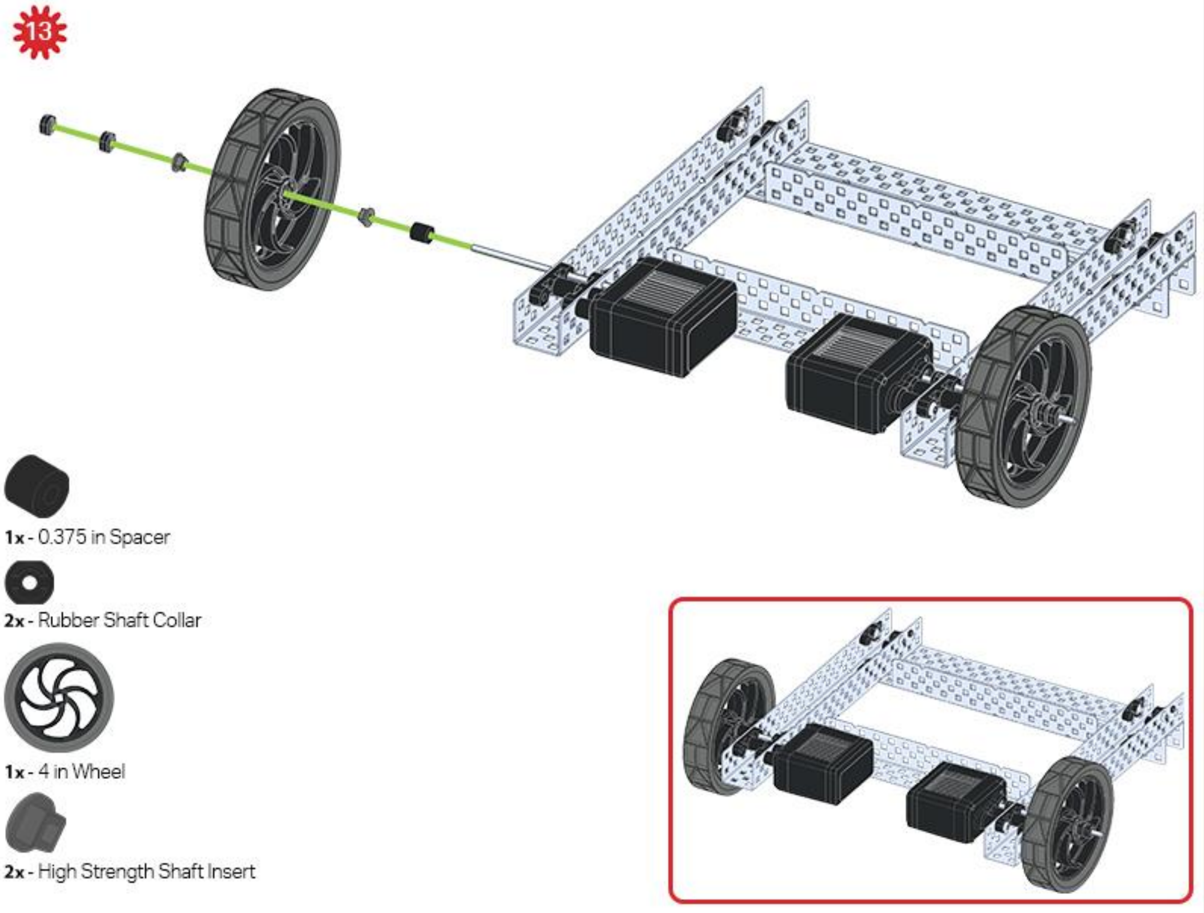
1x - 0.375 in Spacer

2x - Rubber Shaft Collar

1x - 4 in Wheel

2x - High Strength Shaft Insert



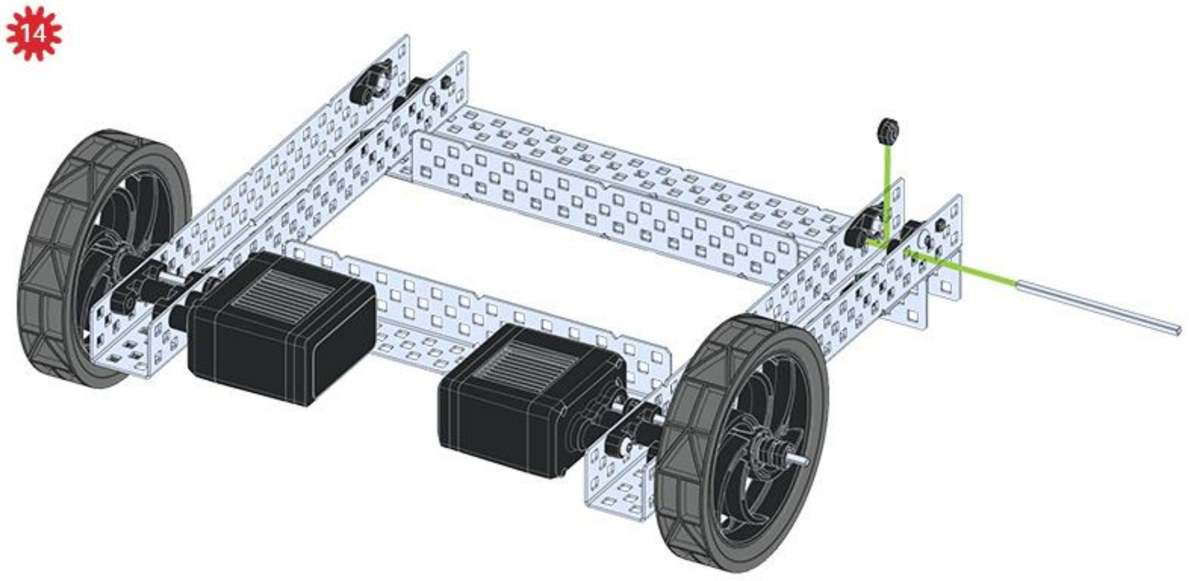


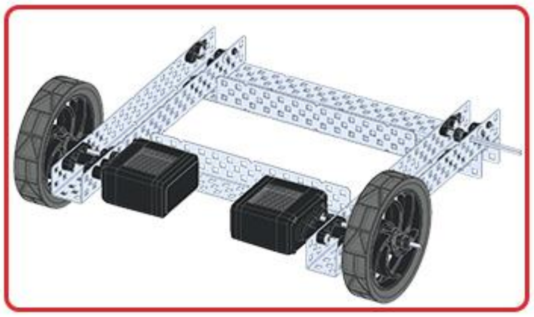
1x - 0.375 in Spacer

2x - Rubber Shaft Collar

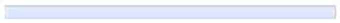
1x - 4 in Wheel

2x - High Strength Shaft Insert

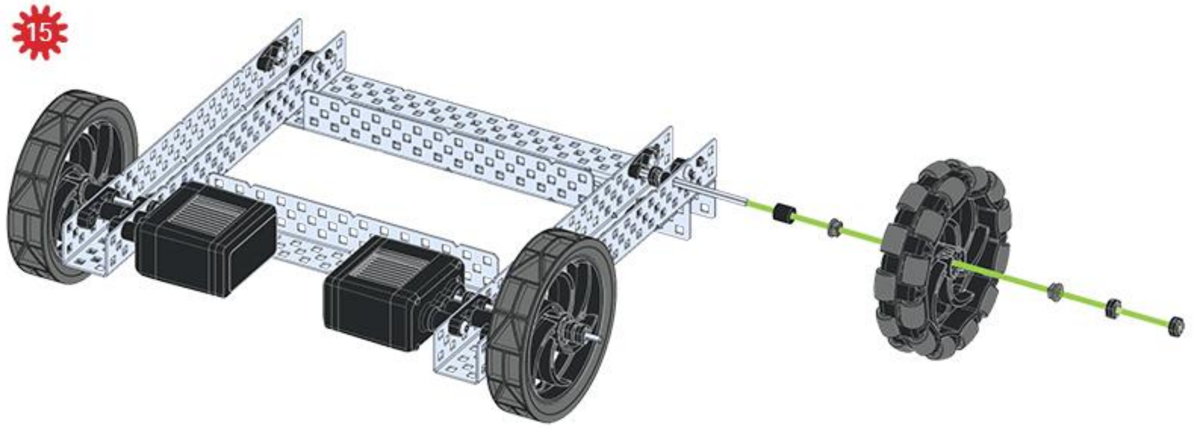




1x - Rubber Shaft Collar


1x - 3 in Shaft

15

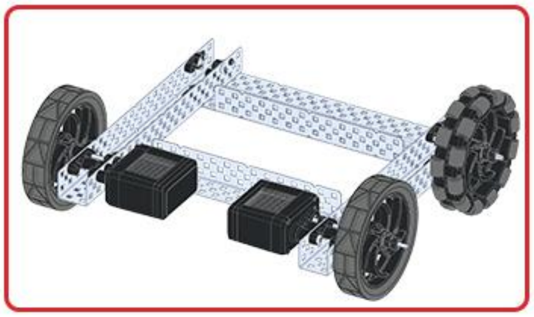


1x - 0.375 in Spacer

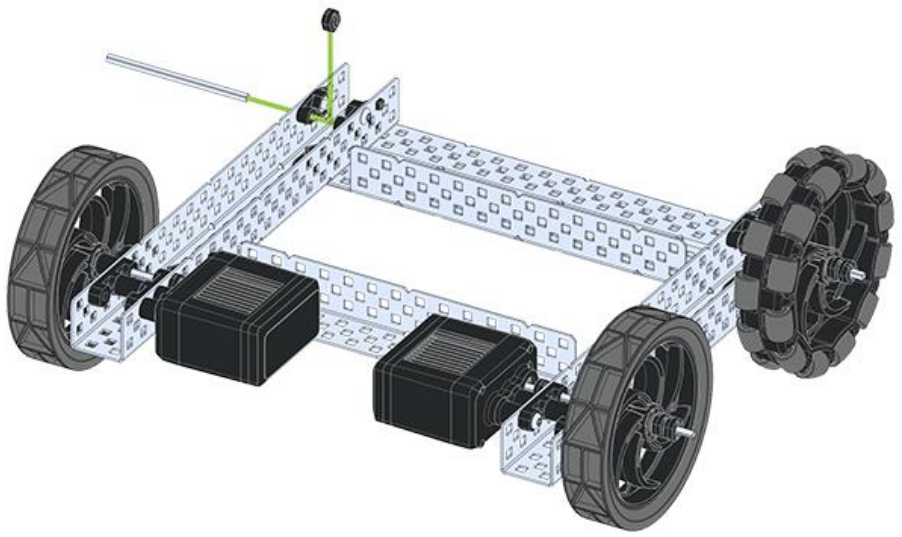
2x - Rubber Shaft Collar

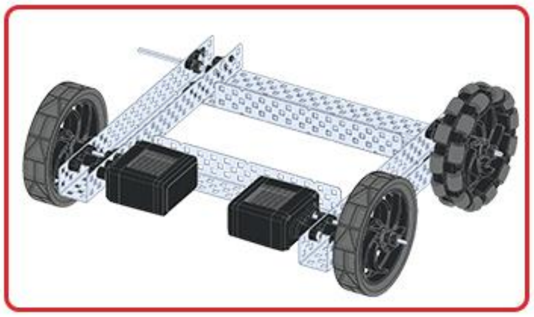
1x - 4 in Omni Wheel

2x - High Strength Shaft Insert



16

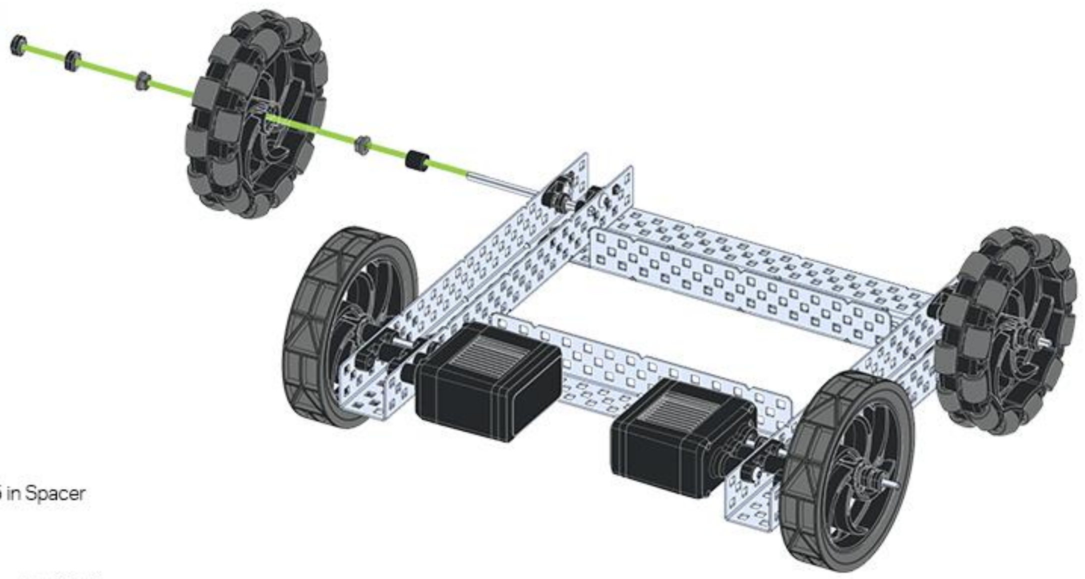




1x - Rubber Shaft Collar

1x - 3 in Shaft

17

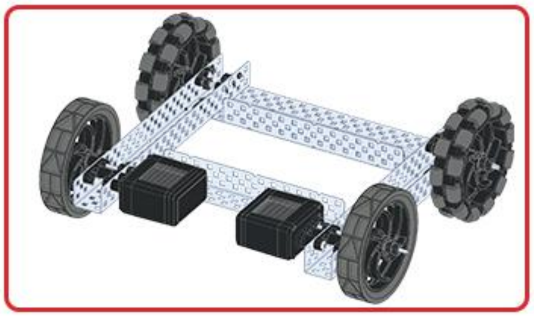


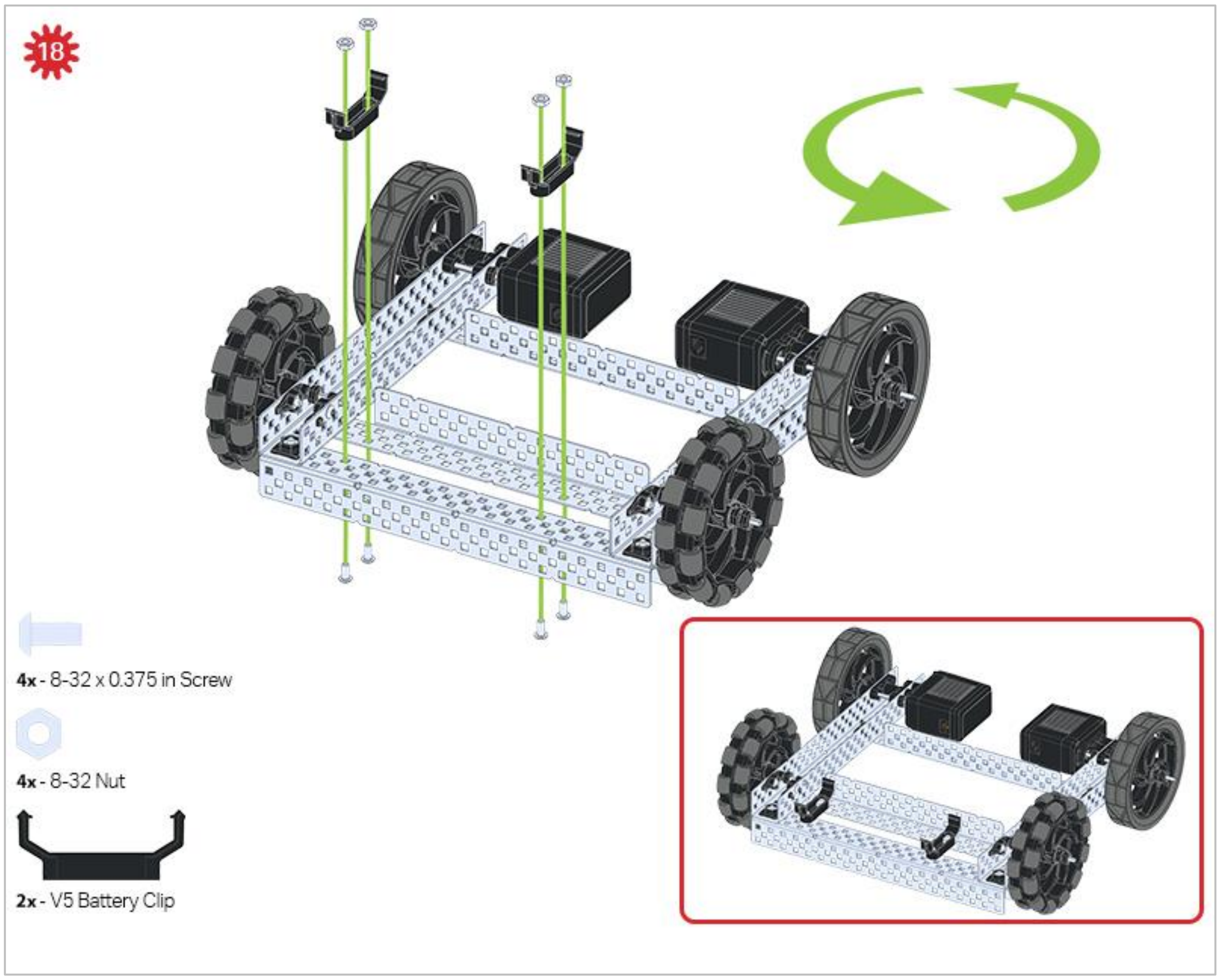
1x - 0.375 in Spacer

2x - Rubber Shaft Collar

1x - 4 in Omni Wheel

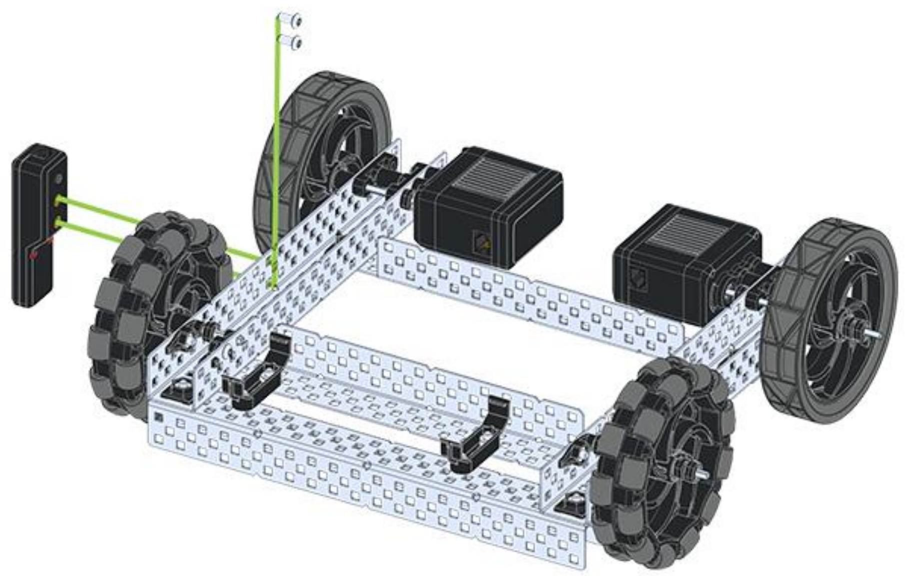
2x - High Strength Shaft Insert





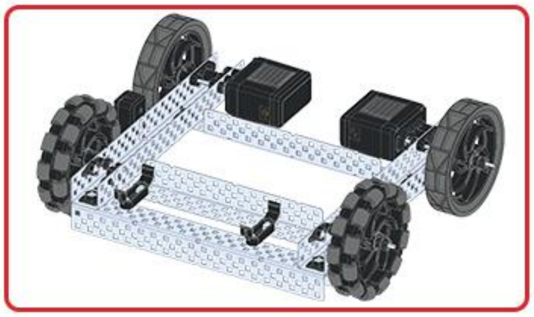
The green icon indicates that the build needs to be rotated (180 degrees).

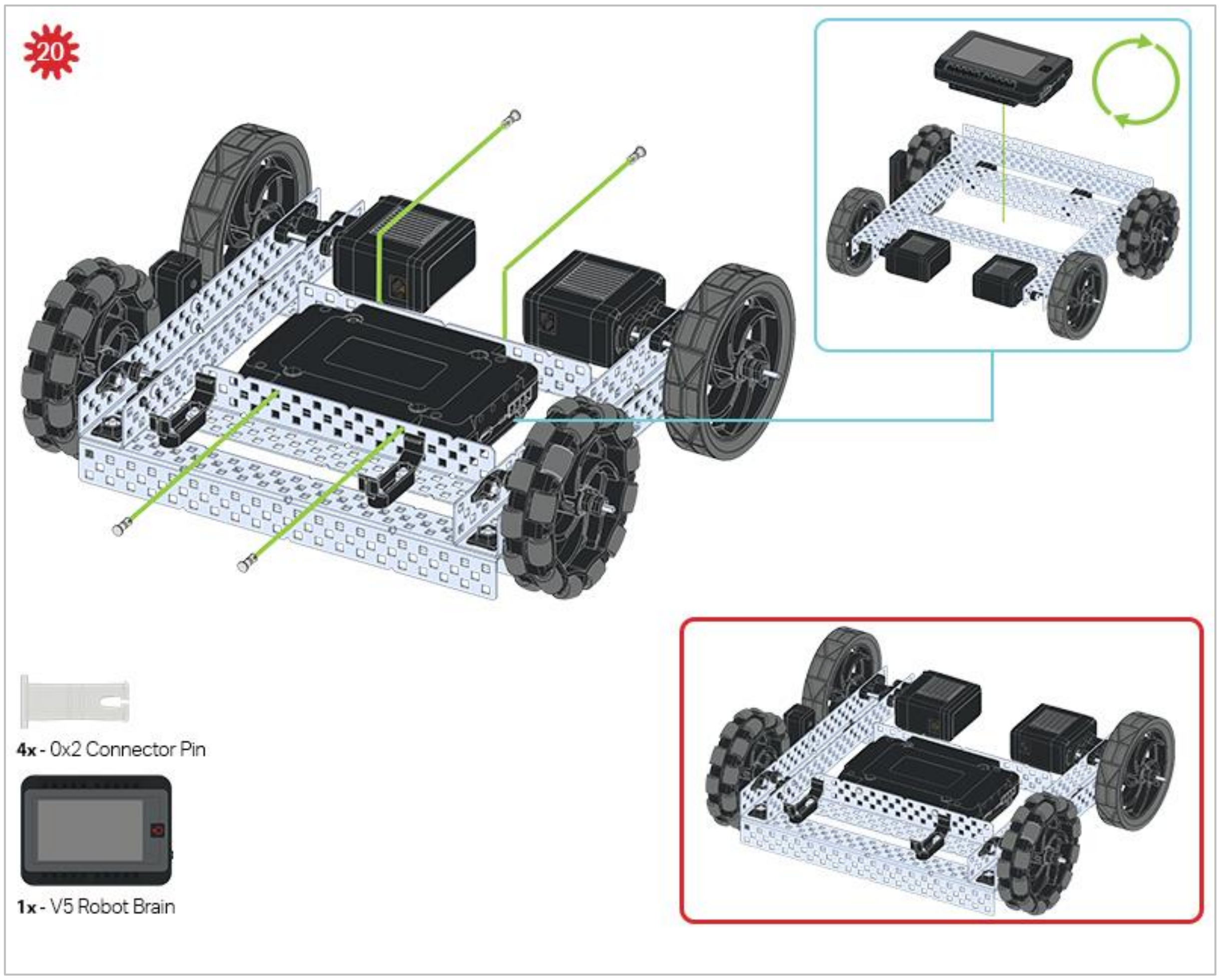
19



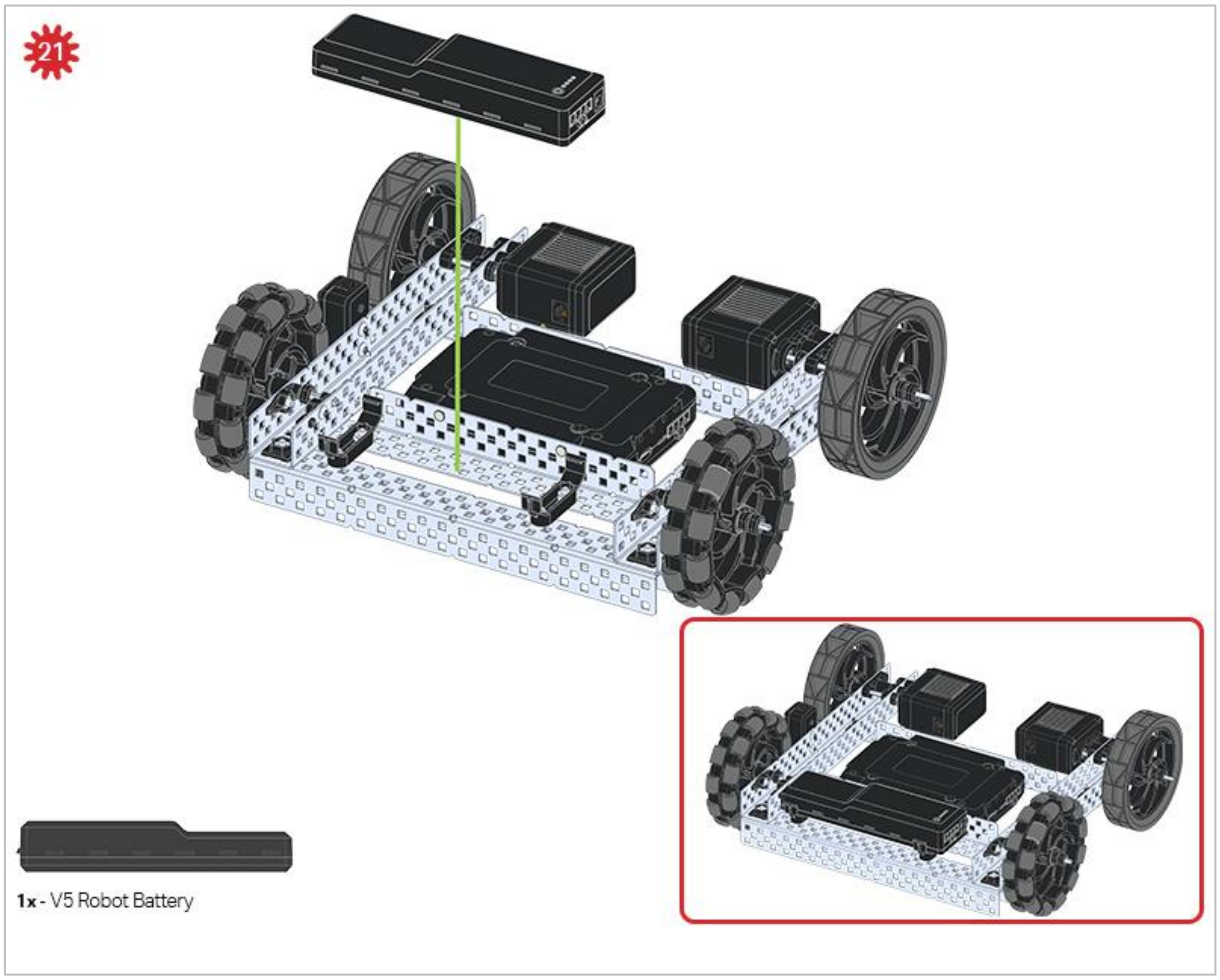
1x - V5 Radio

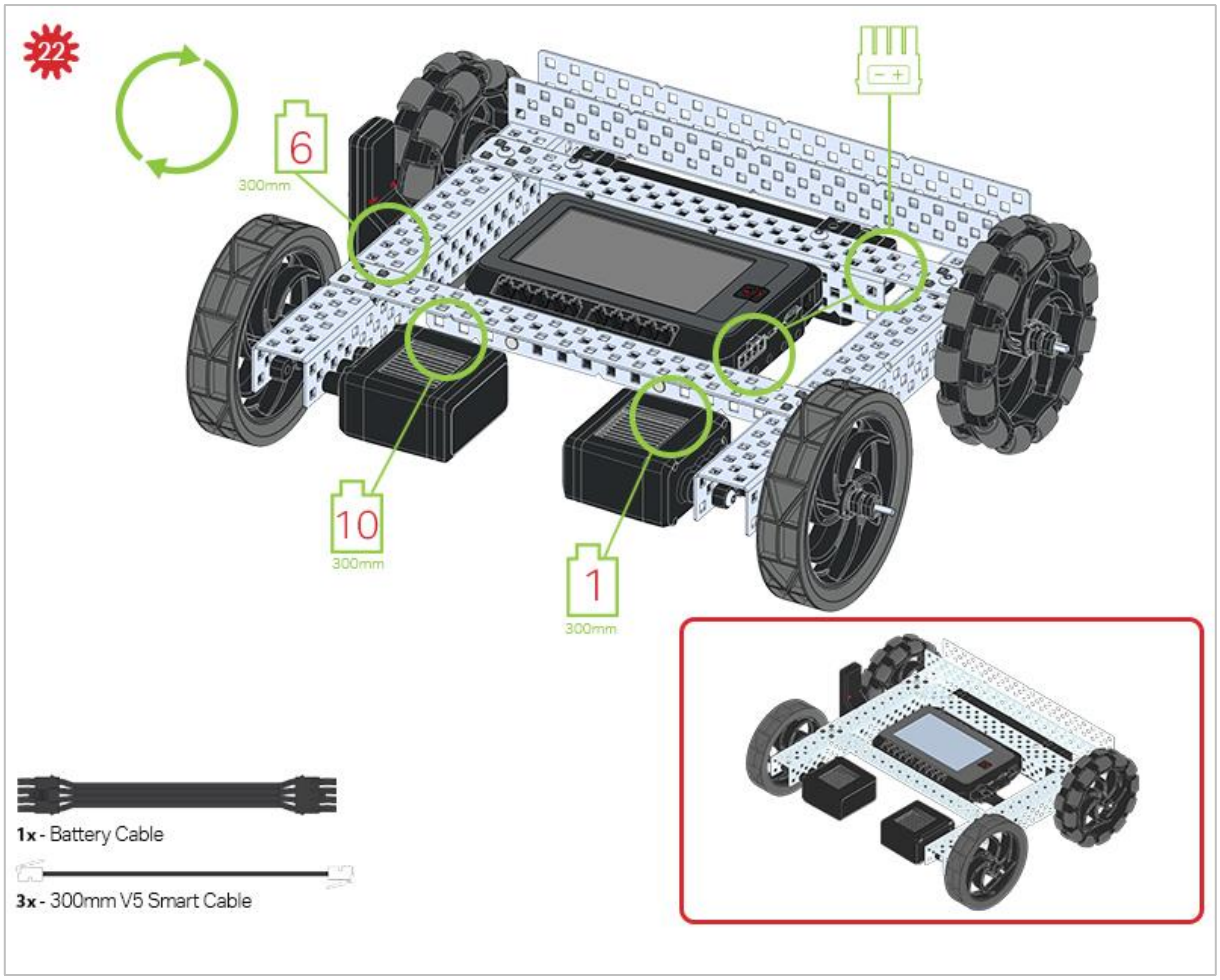
2x - 8-32 x 0.375 in Screw





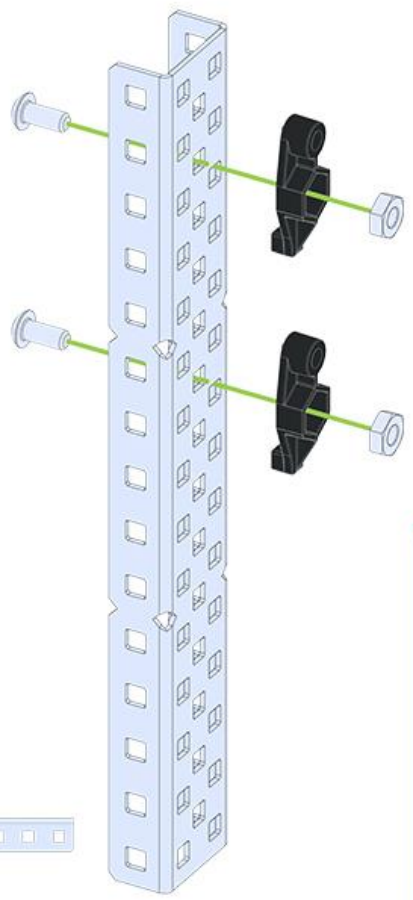
The blue call out shows what the orientation of the Robot Brain should be if the build were flipped right side up. Make sure the 3 wire ports on the Robot Brain are facing the V5 Radio!


21
1x - V5 Robot Battery



The green call outs indicate which port on the Robot Brain to plug each device into using their respective cable.

23

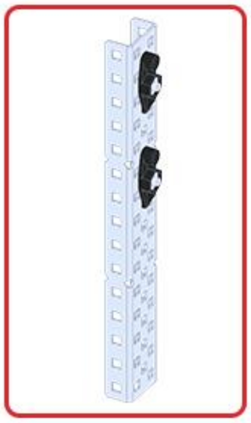


2x - 8-32 x 0.375 in Screw

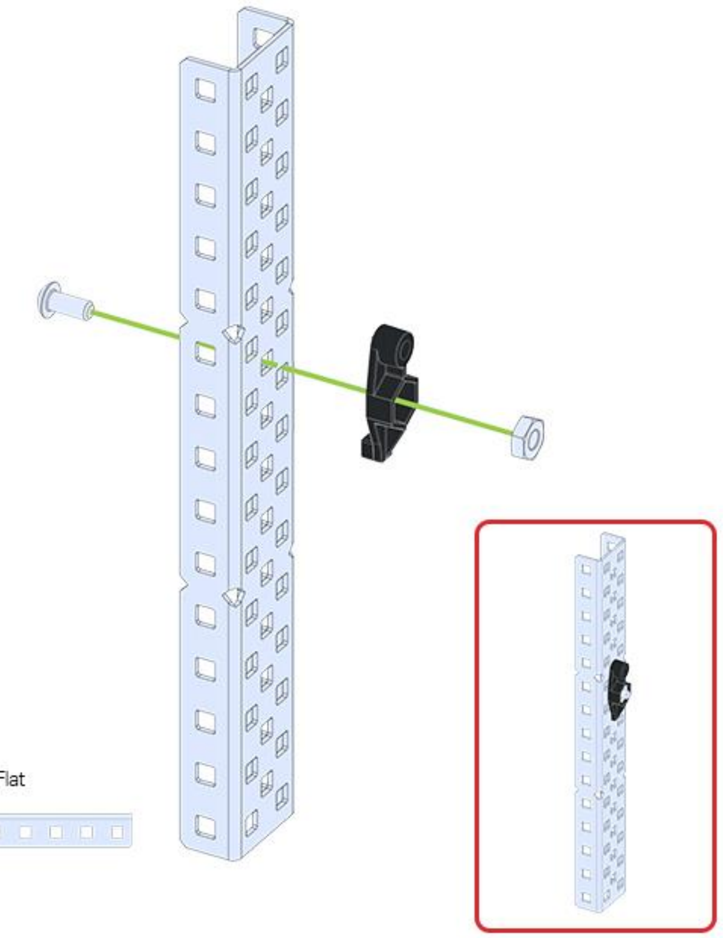
2x - 8-32 Nut

2x - 1 Post Hex Nut Retainer w/ Bearing Flat

1x - 1x2x1x15 C-Channel



24



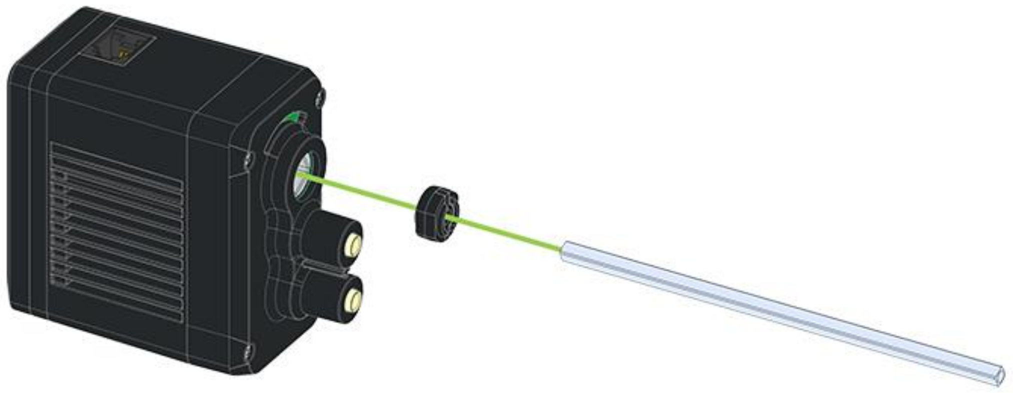
1x - 8-32 x 0.375 in Screw

1x - 8-32 Nut

1x - 1 Post Hex Nut Retainer w/ Bearing Flat

1x - 1x2x1x15 C-Channel

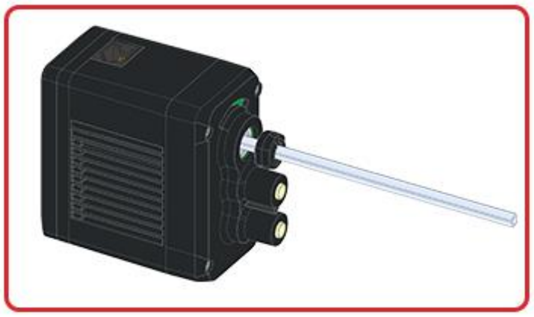
25

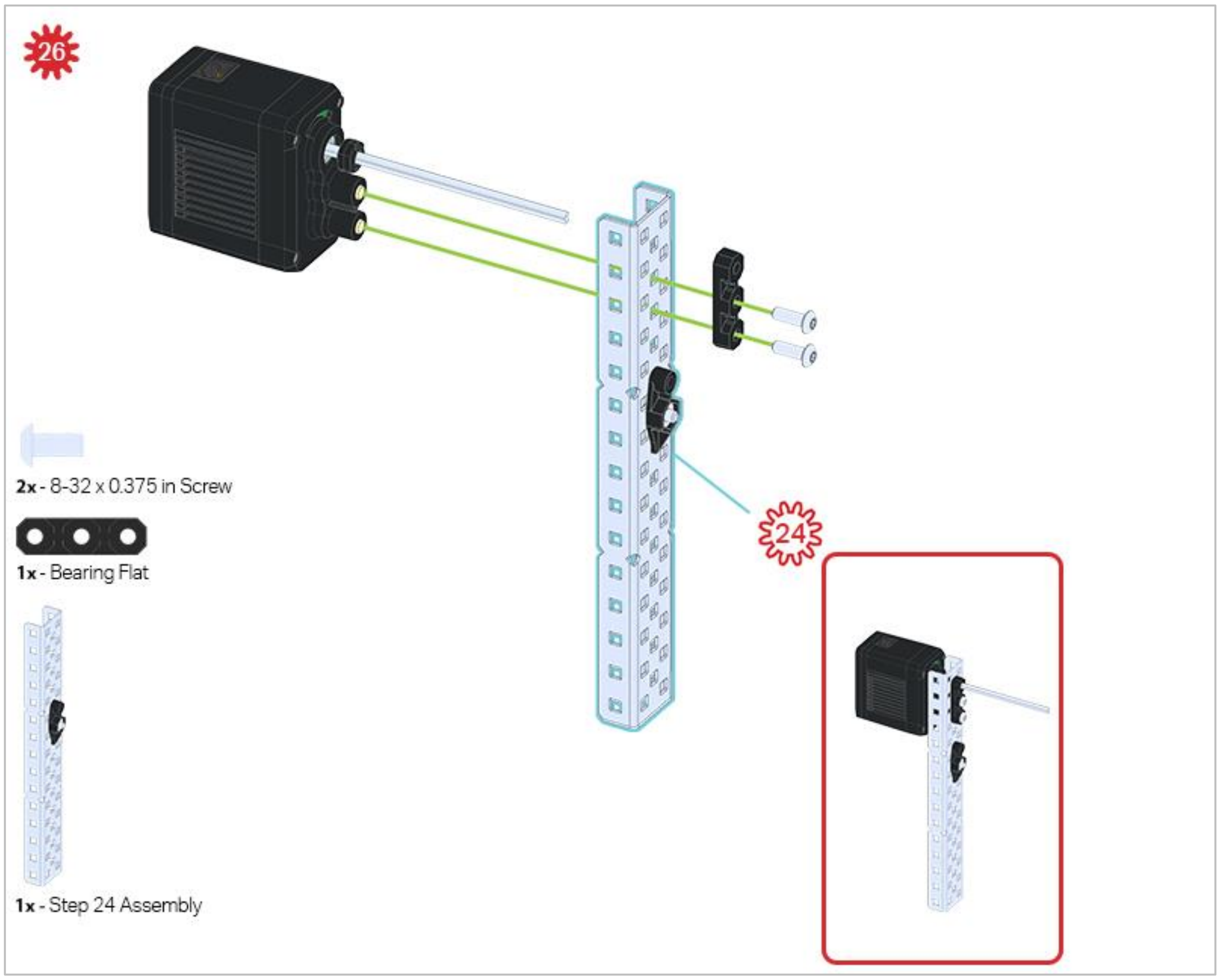


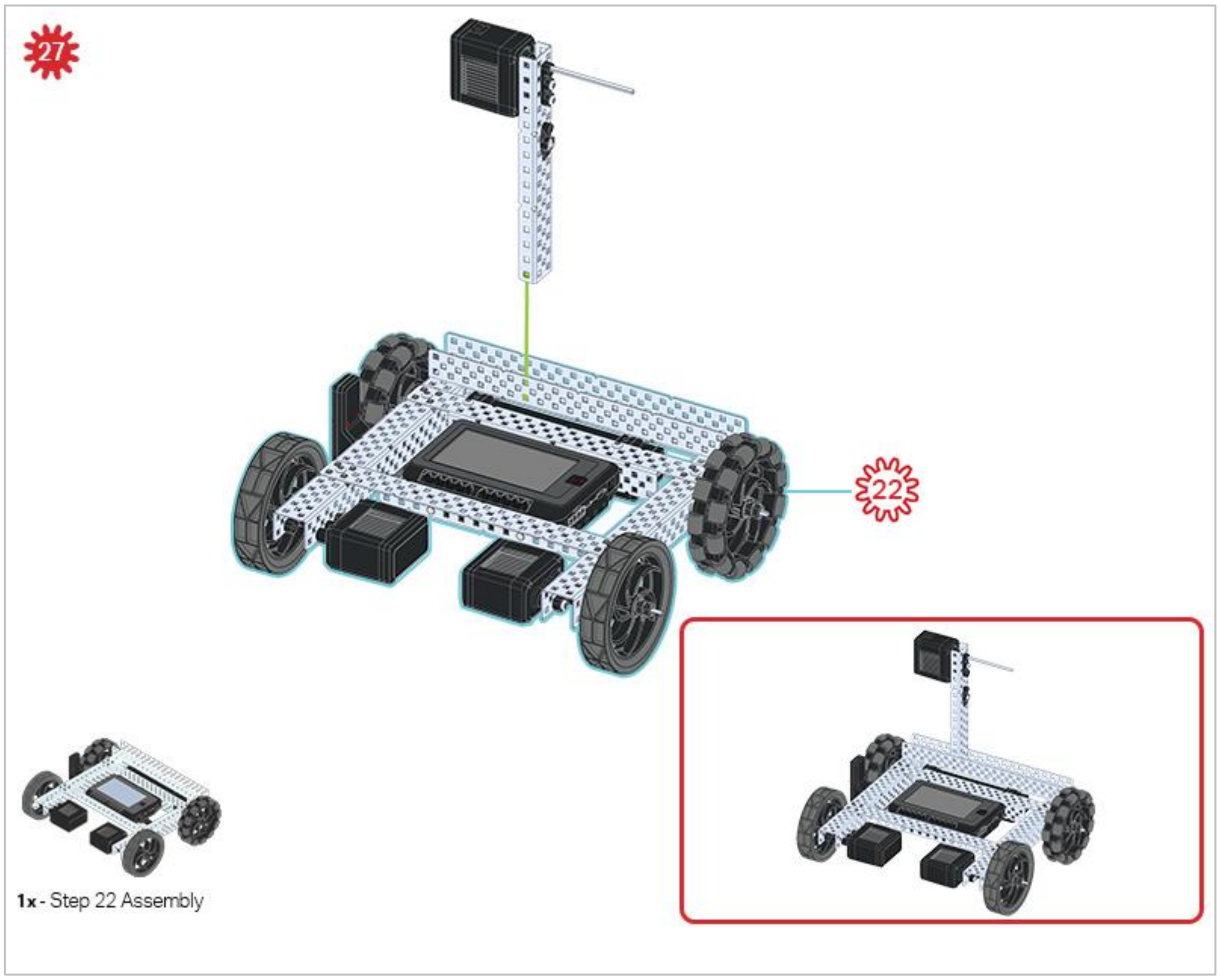
1x - Rubber Shaft Collar

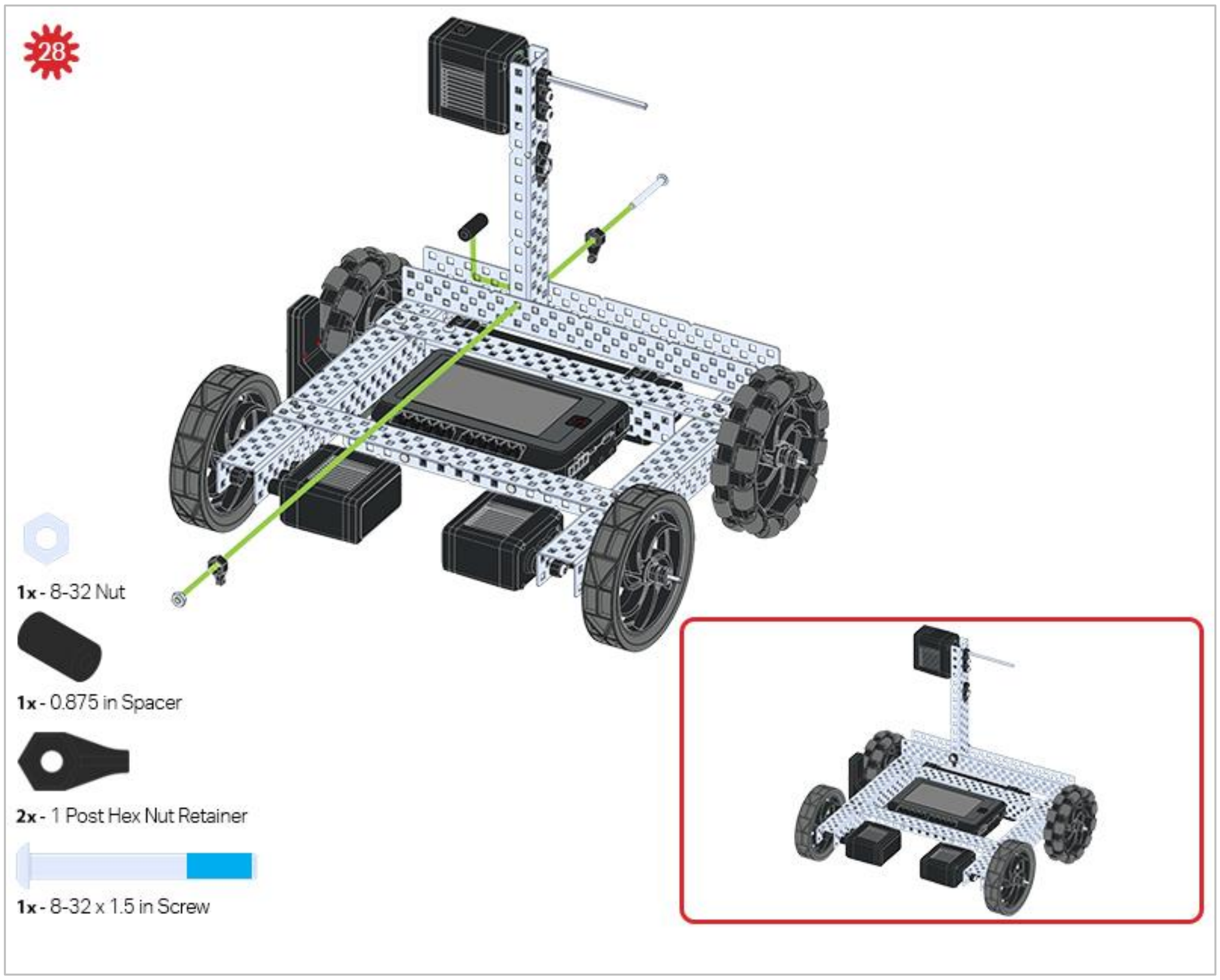
1x - V5 Smart Motor

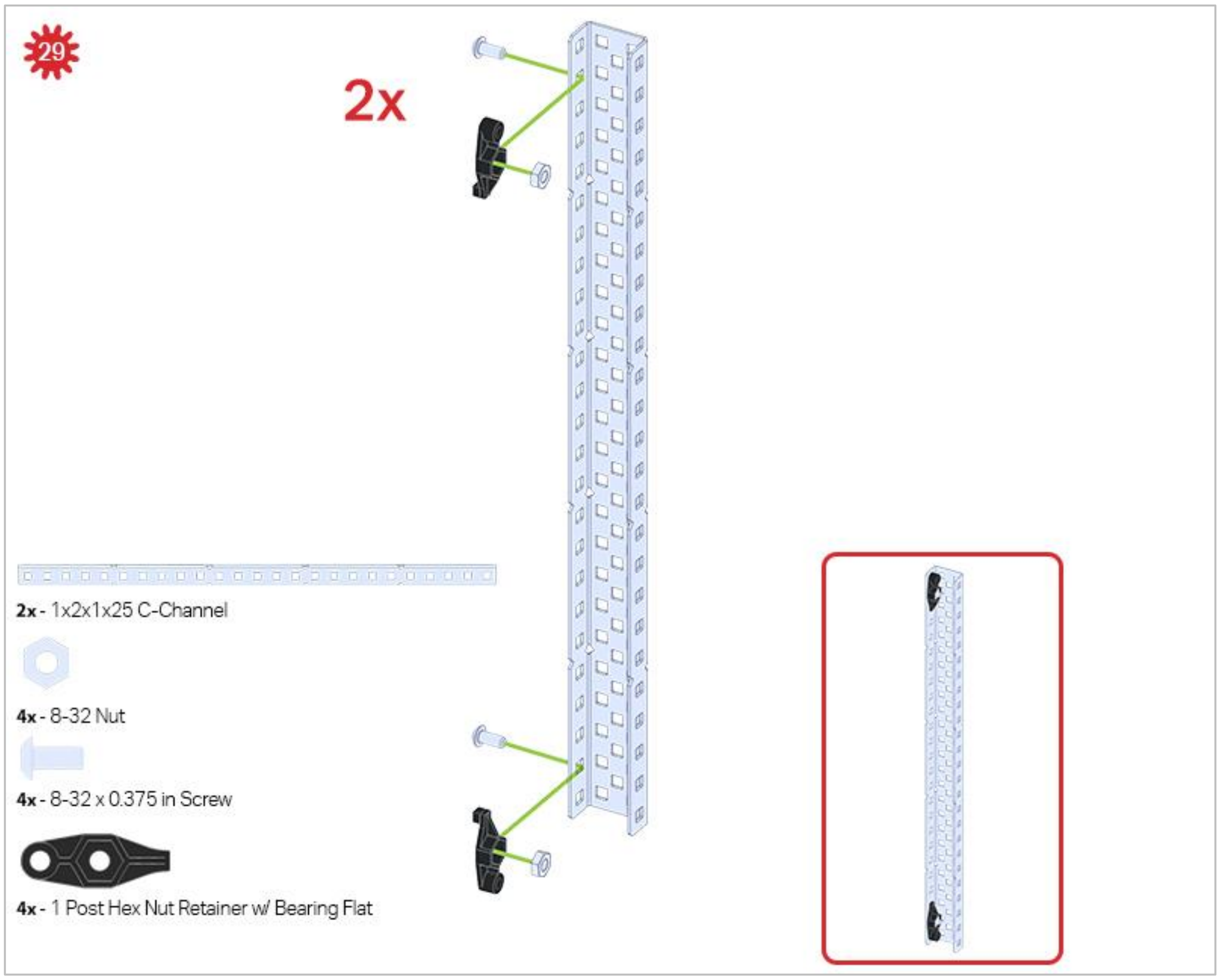
1x - 4 in Shaft



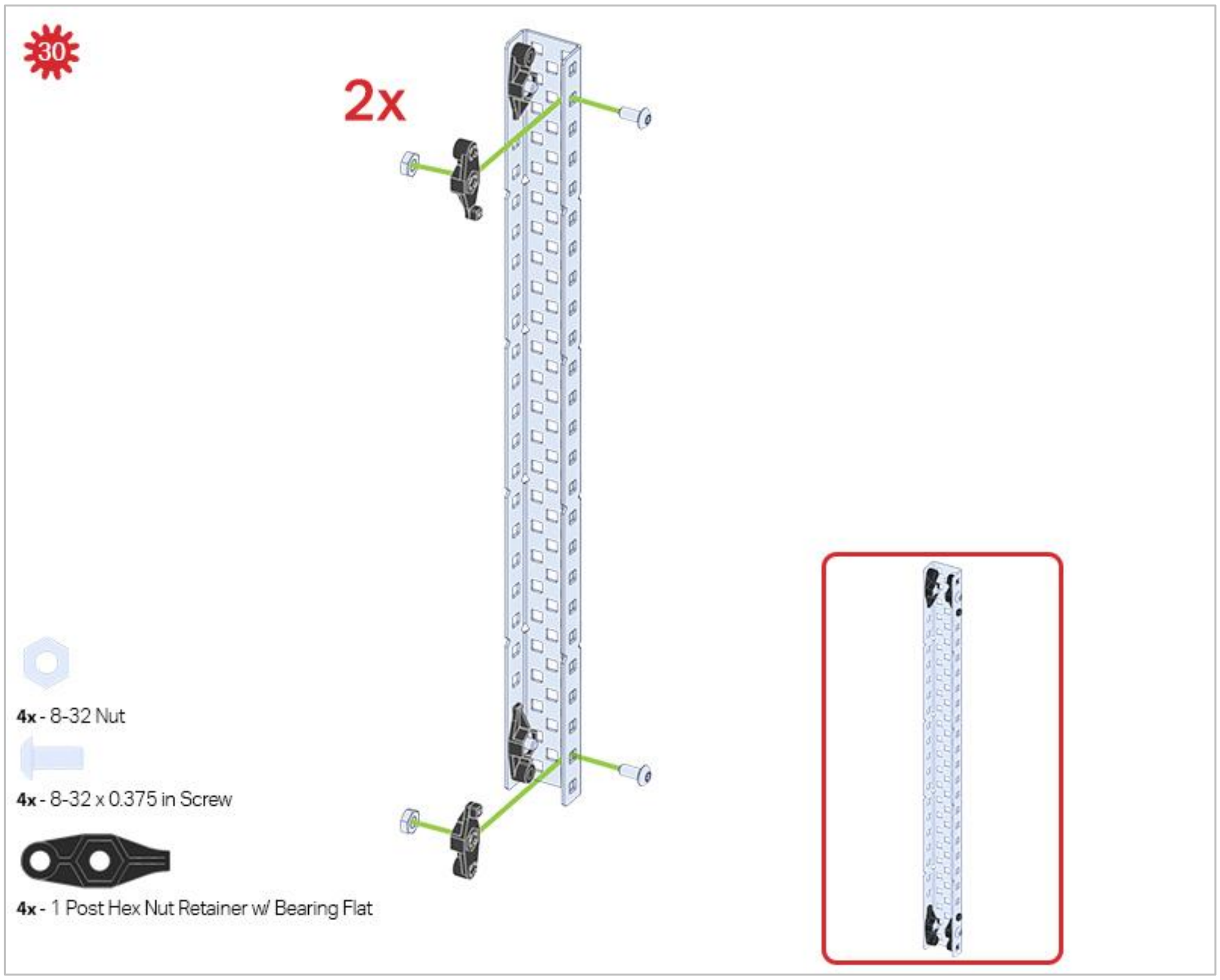

26
2x - 8-32 x 0.375 in Screw
1x - Bearing Flat
1x - Step 24 Assembly
24


27
22
1x - Step 22 Assembly

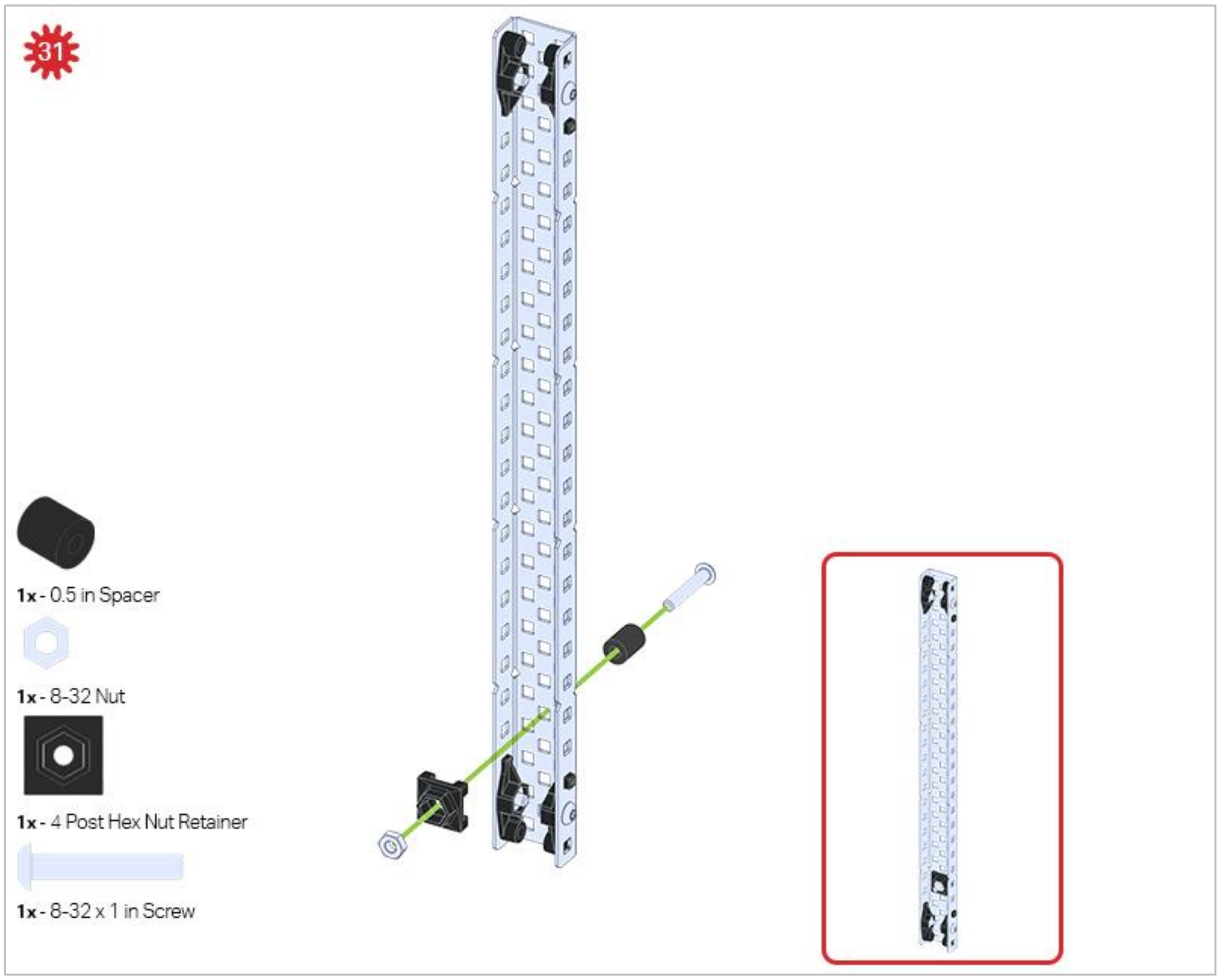

28
1x - 8-32 Nut
1x - 0.875 in Spacer
2x - 1 Post Hex Nut Retainer
1x - 8-32 x 1.5 in Screw


29
2x
2x - 1x2x1x25 C-Channel
4x - 8-32 Nut
4x - 8-32 x 0.375 in Screw
4x - 1 Post Hex Nut Retainer w/ Bearing Flat

Be sure to make two assemblies in this step!

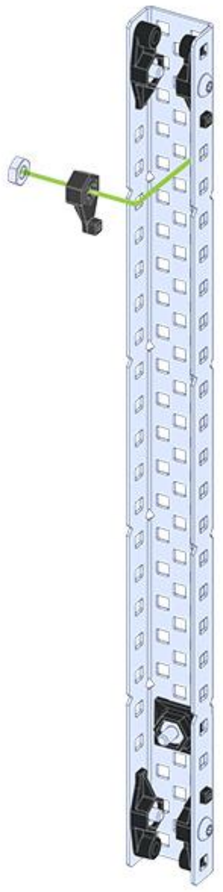

30
2x
4x - 8-32 Nut
4x - 8-32 x 0.375 in Screw
4x - 1 Post Hex Nut Retainer w/ Bearing Flat

This step adds onto the two assemblies started in Step 29.



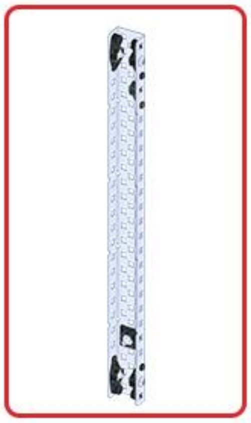
Make sure to add this to only one of the two sub-assemblies you just made.

32

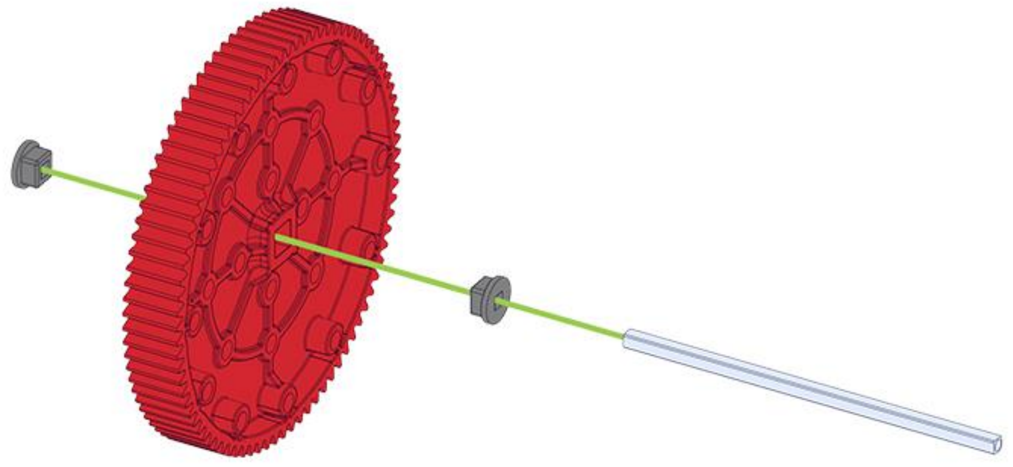


1x - 8-32 Nut

1x - 1 Post Hex Nut Retainer



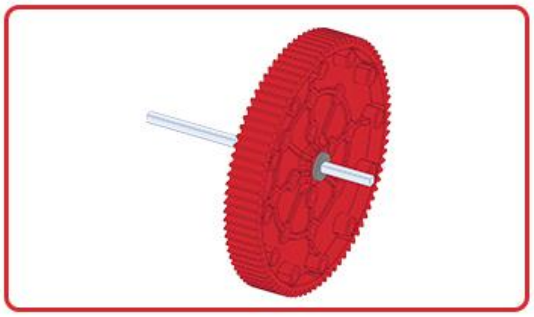
33

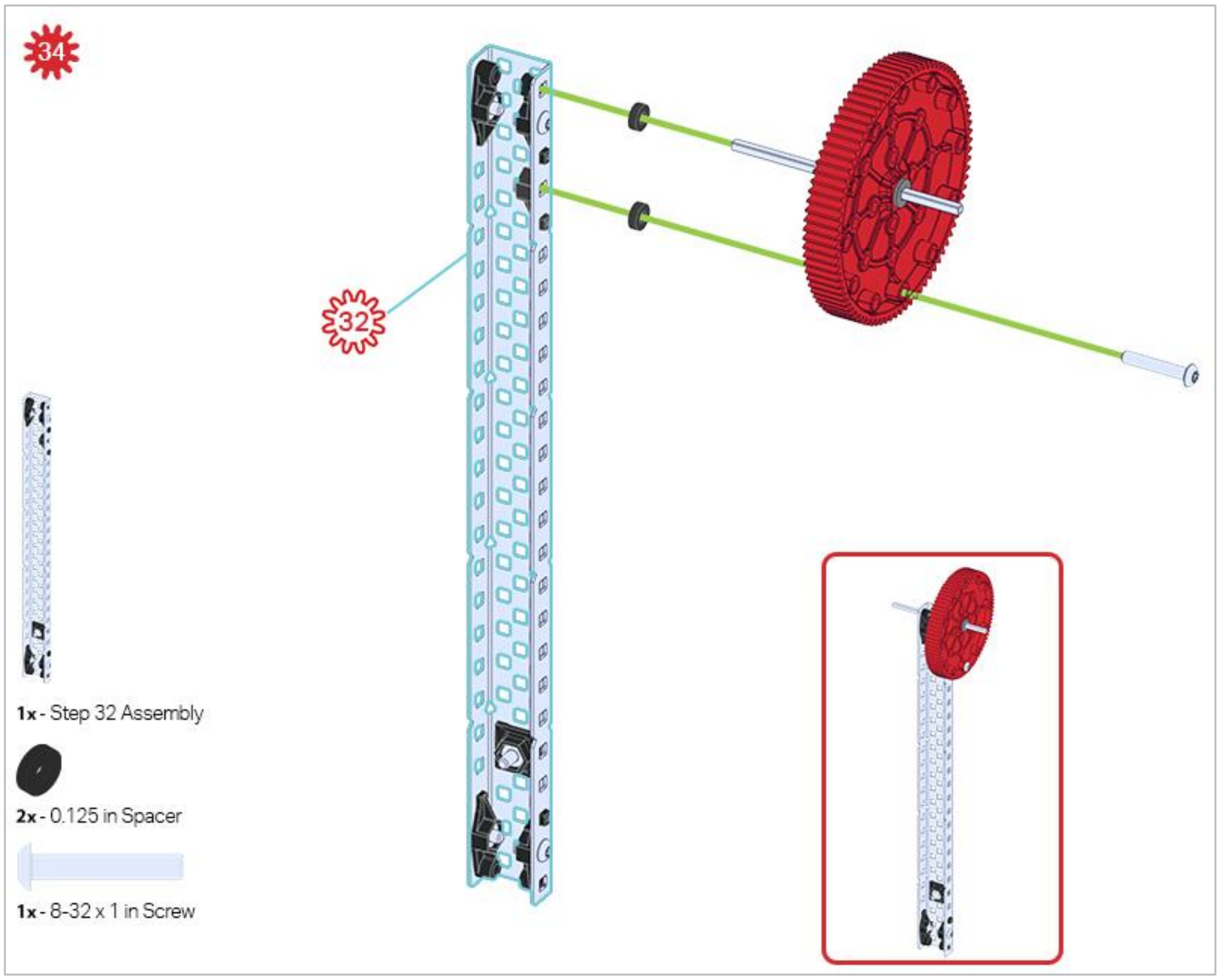


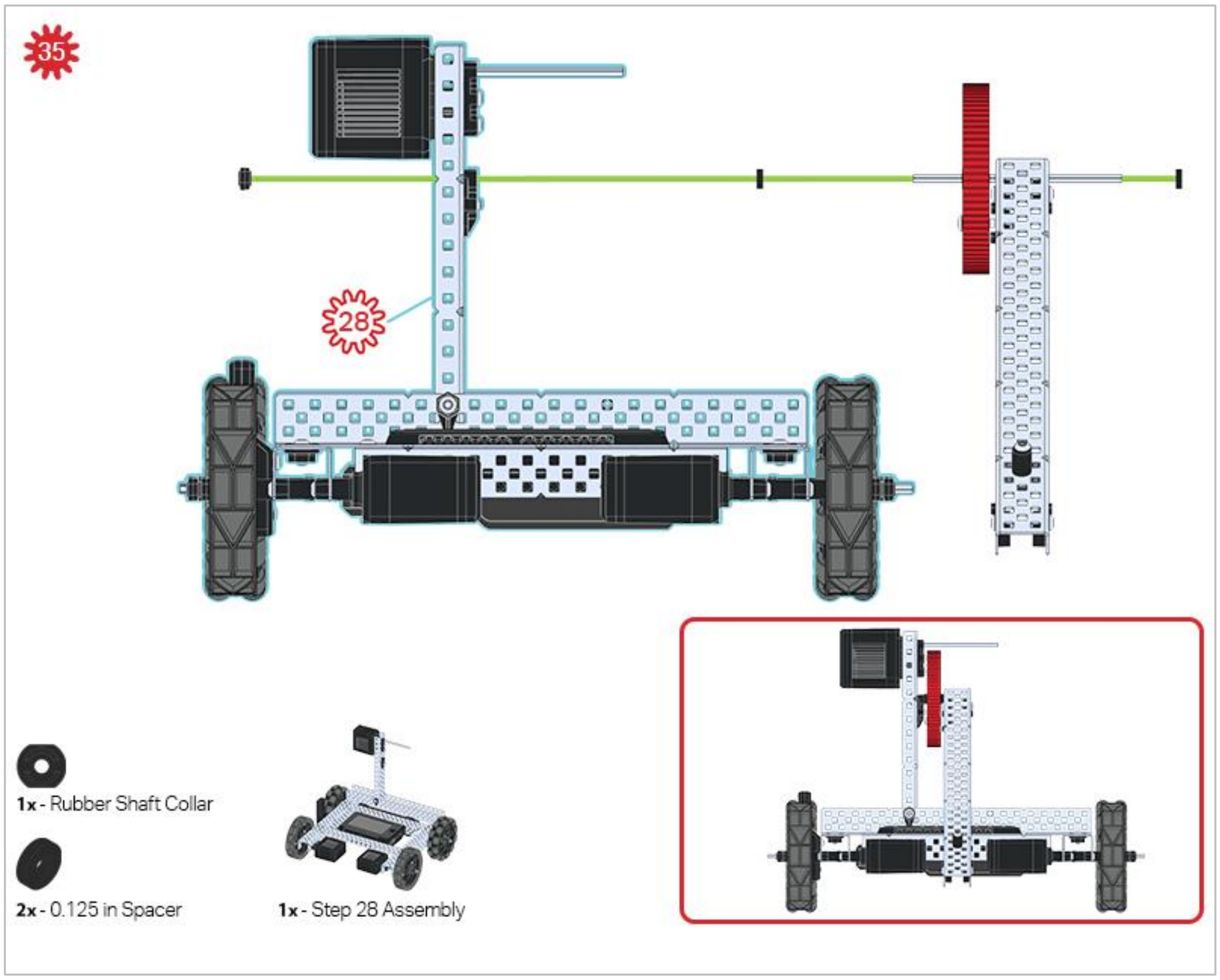
1x - High Strength 84 Tooth Gear

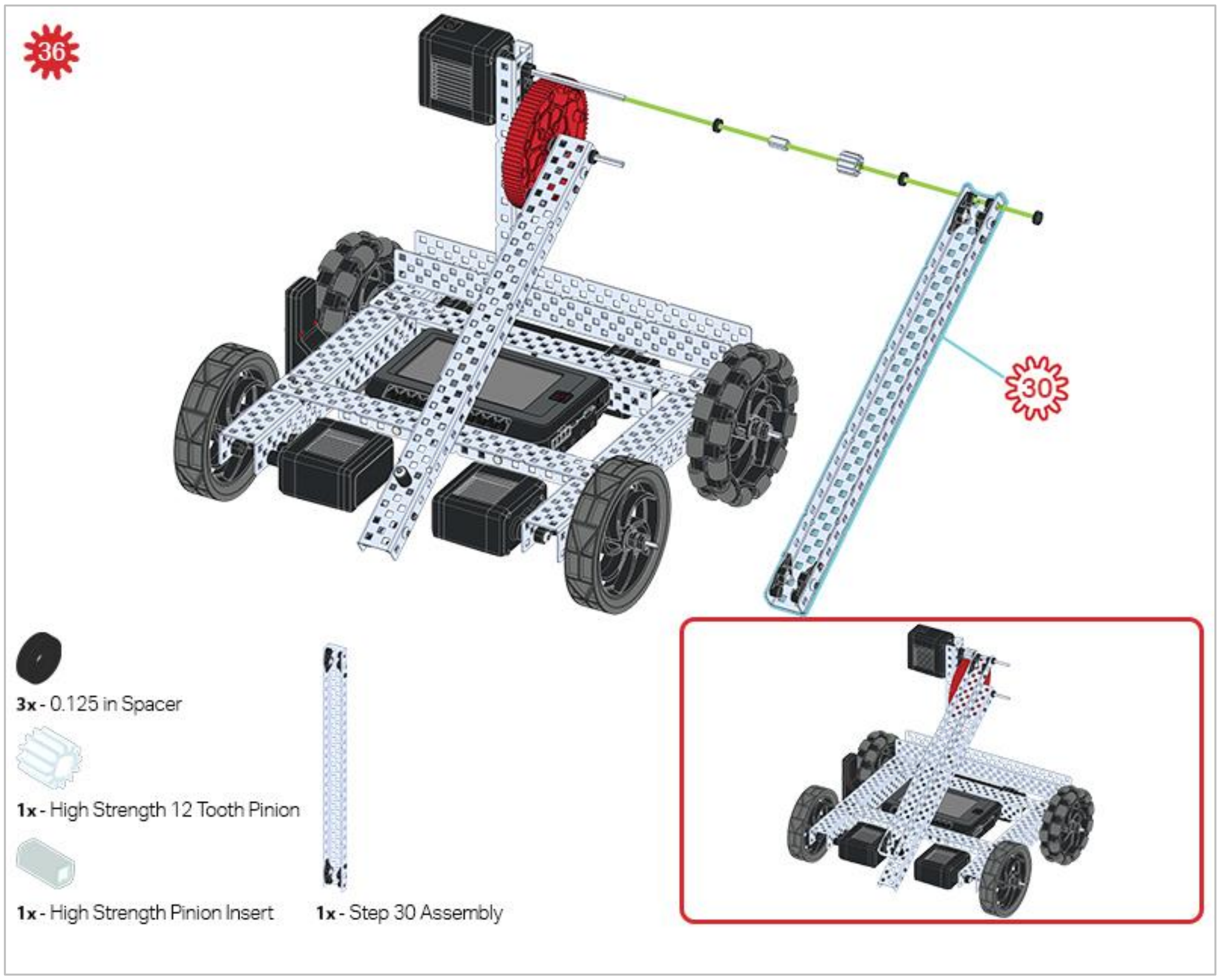
2x - High Strength Shaft Insert

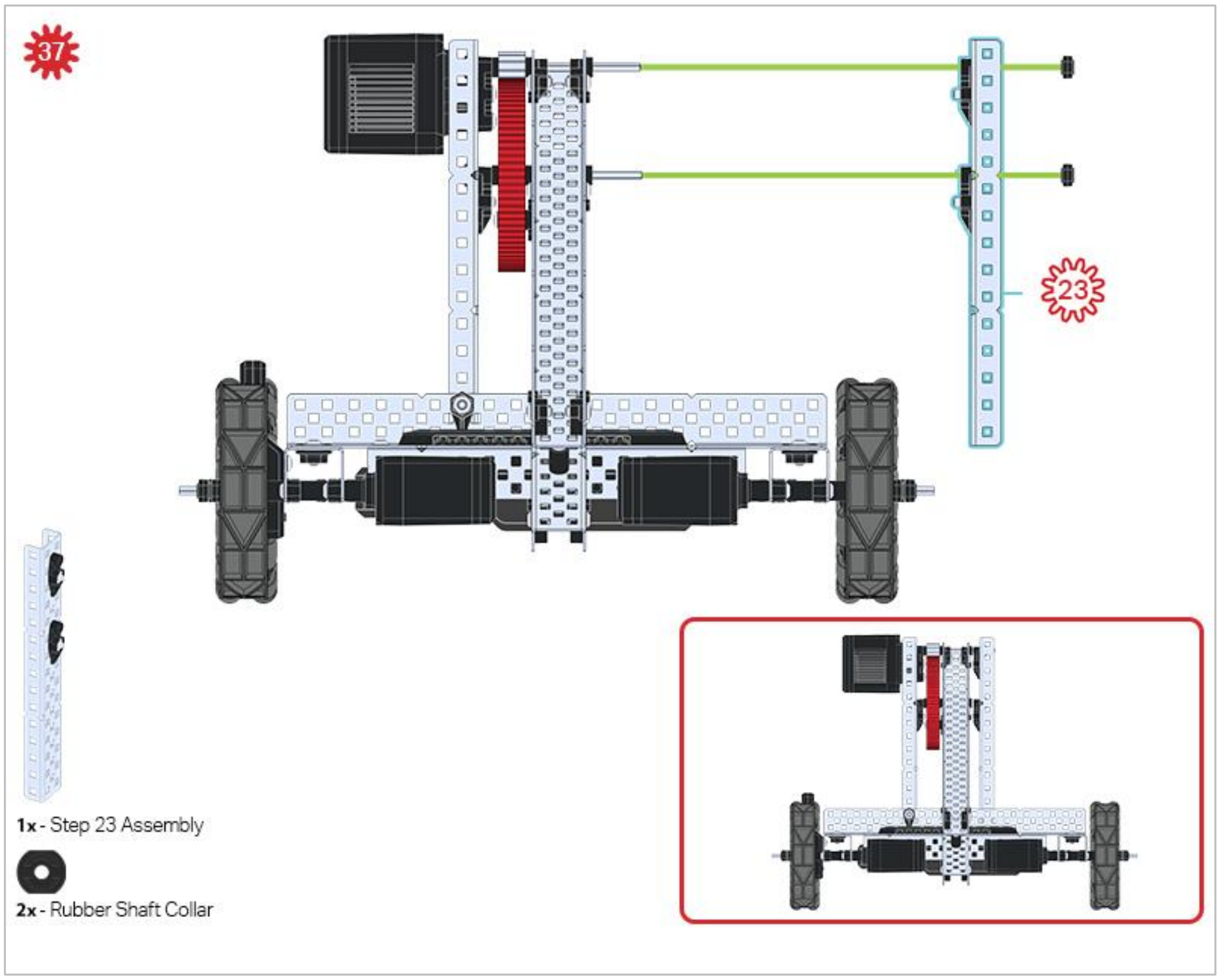
1x - 3.5 in Shaft

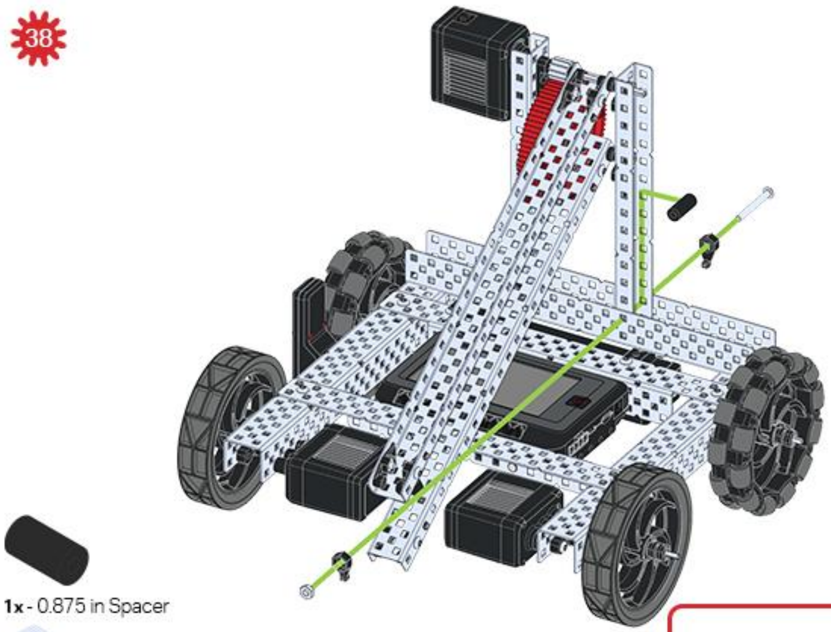



34
32
1x - Step 32 Assembly
2x - 0.125 in Spacer
1x - 8-32 x 1 in Screw


35
28
1x - Rubber Shaft Collar
2x - 0.125 in Spacer
1x - Step 28 Assembly


36
30
3x - 0.125 in Spacer
1x - High Strength 12 Tooth Pinion
1x - High Strength Pinion Insert
1x - Step 30 Assembly


37
23
1x - Step 23 Assembly
2x - Rubber Shaft Collar

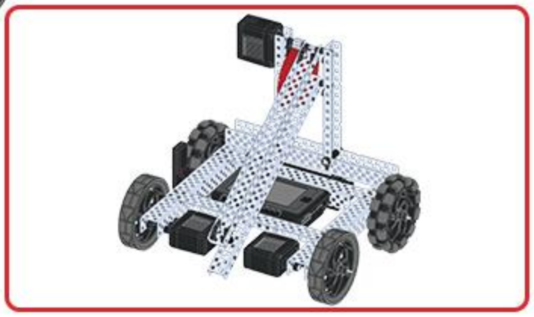


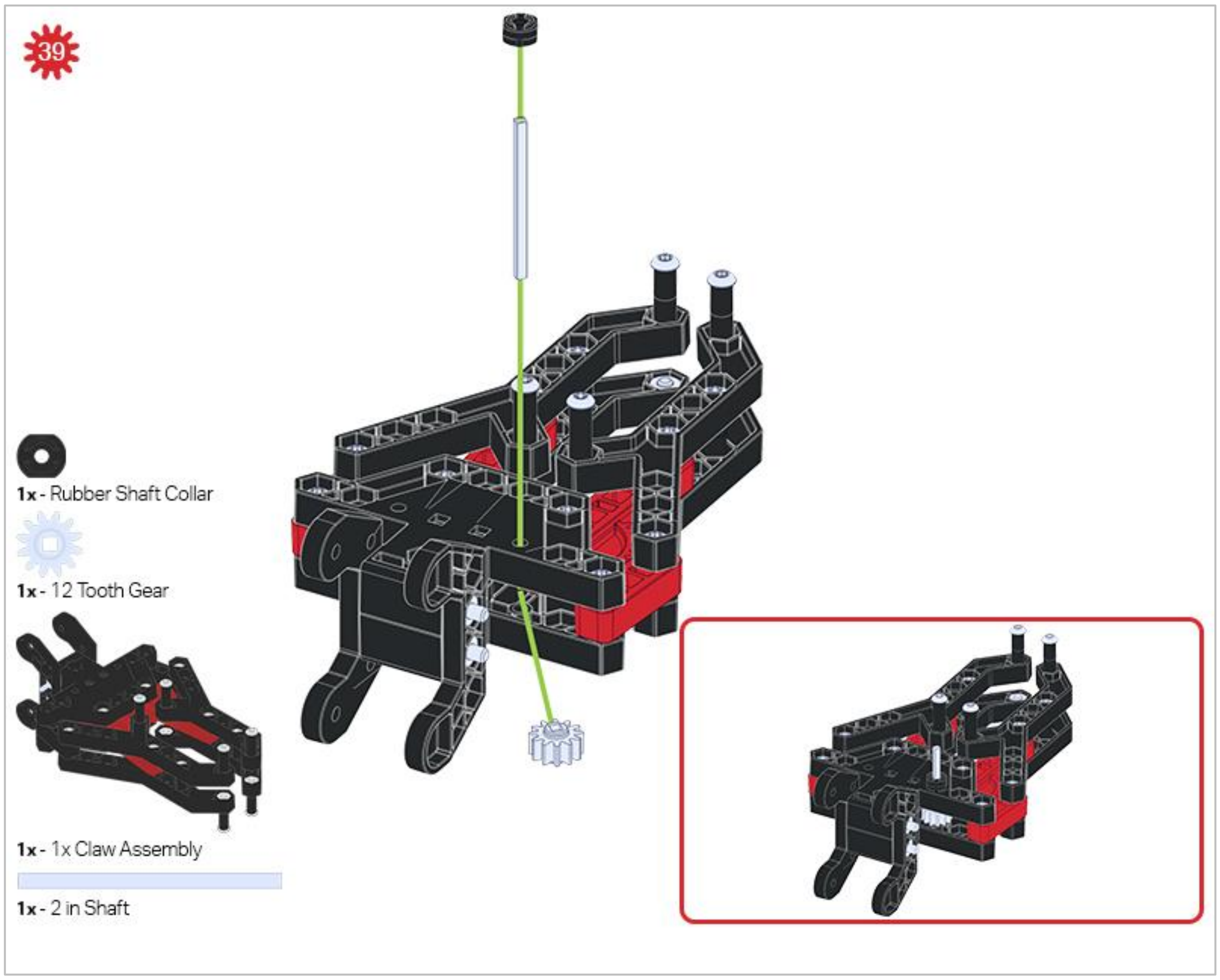
1x - 0.875 in Spacer

1x - 8-32 Nut

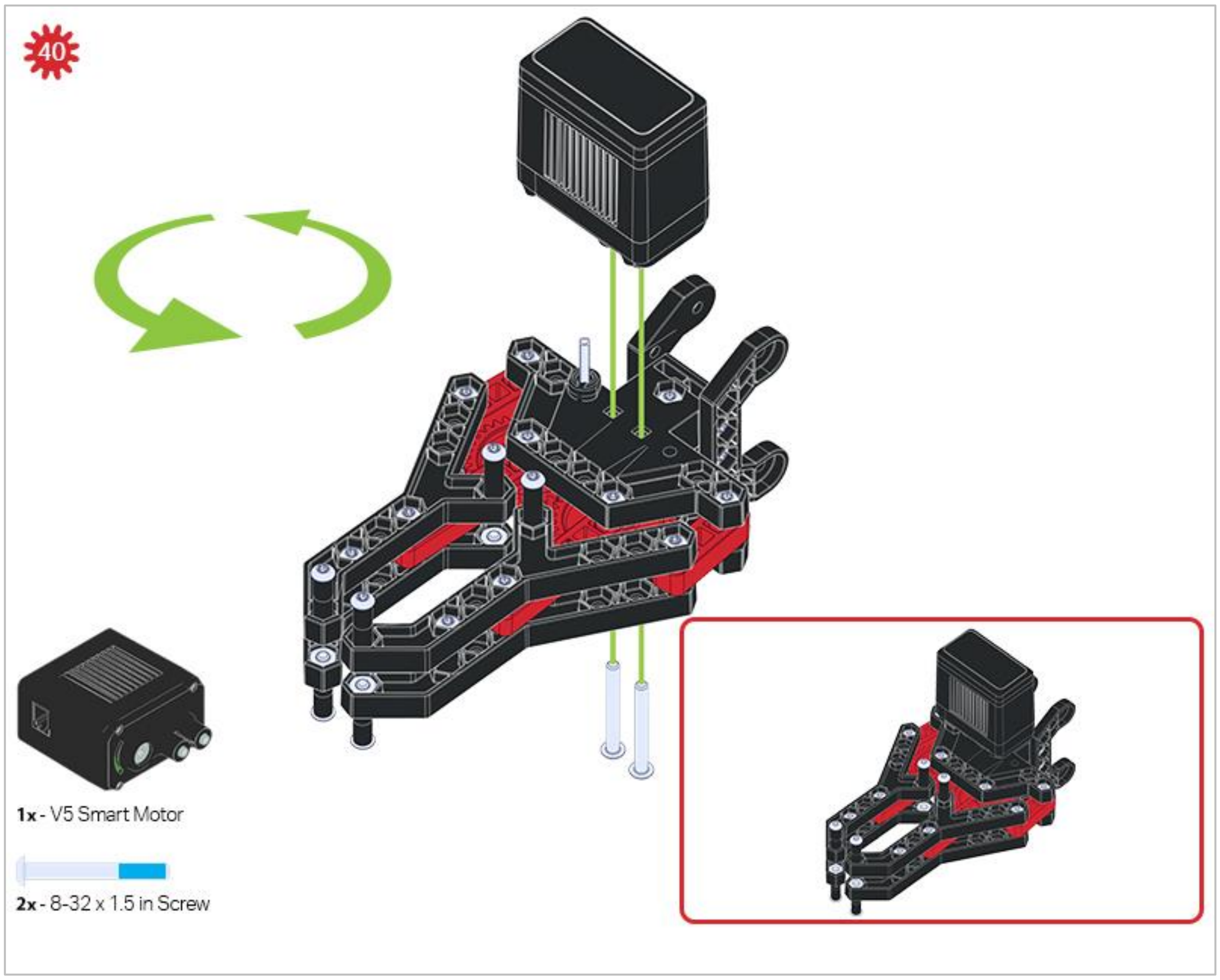
2x - 1 Post Hex Nut Retainer

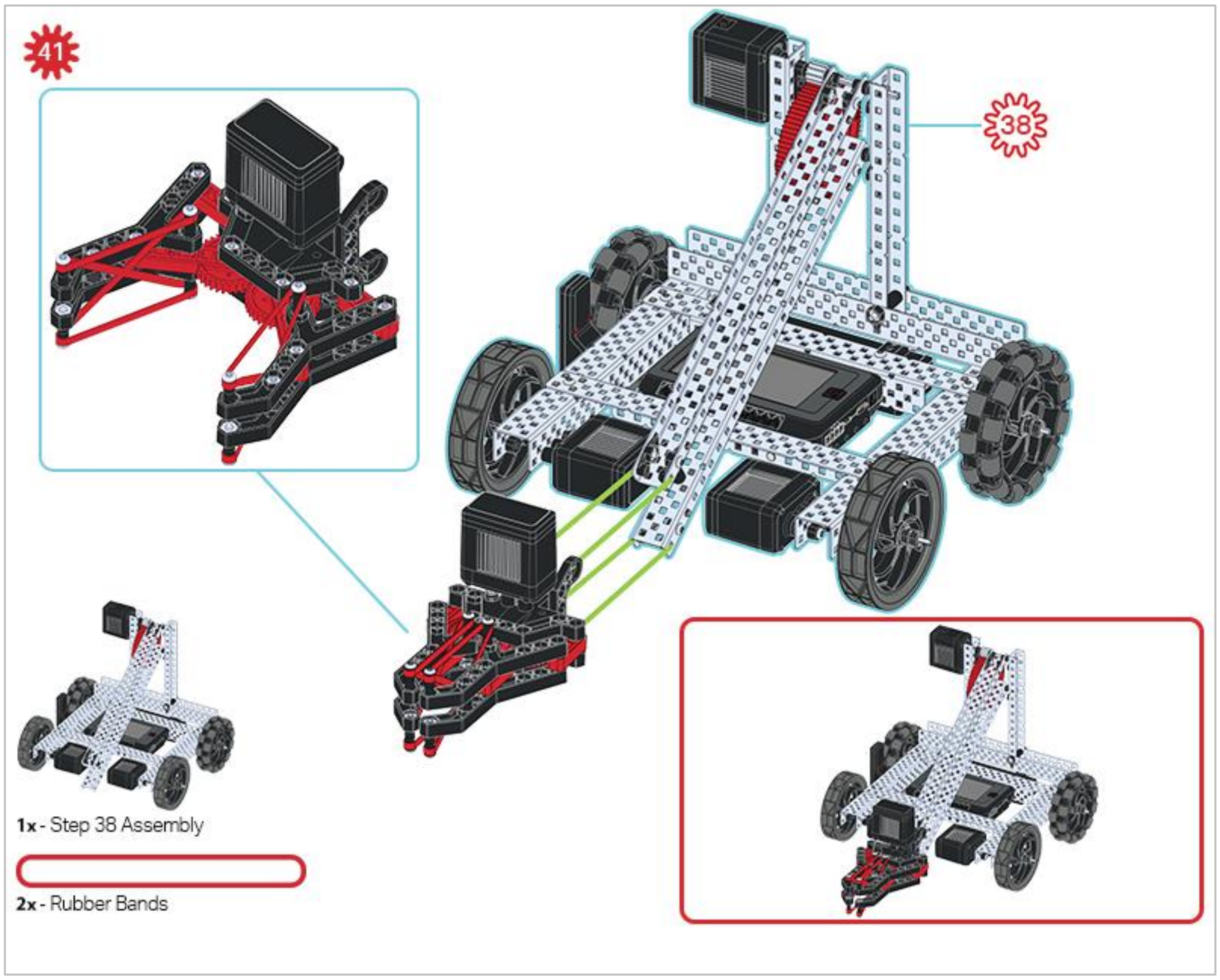
1x - 8-32 x 1.5 in Screw





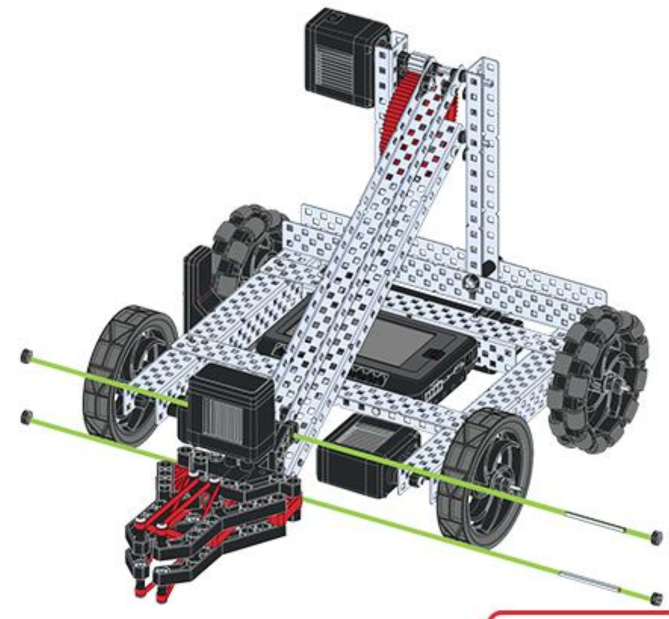
Make sure the 12- tooth gear is installed on the right side of the claw.


40
1x - V5 Smart Motor
2x - 8-32 x 1.5 in Screw



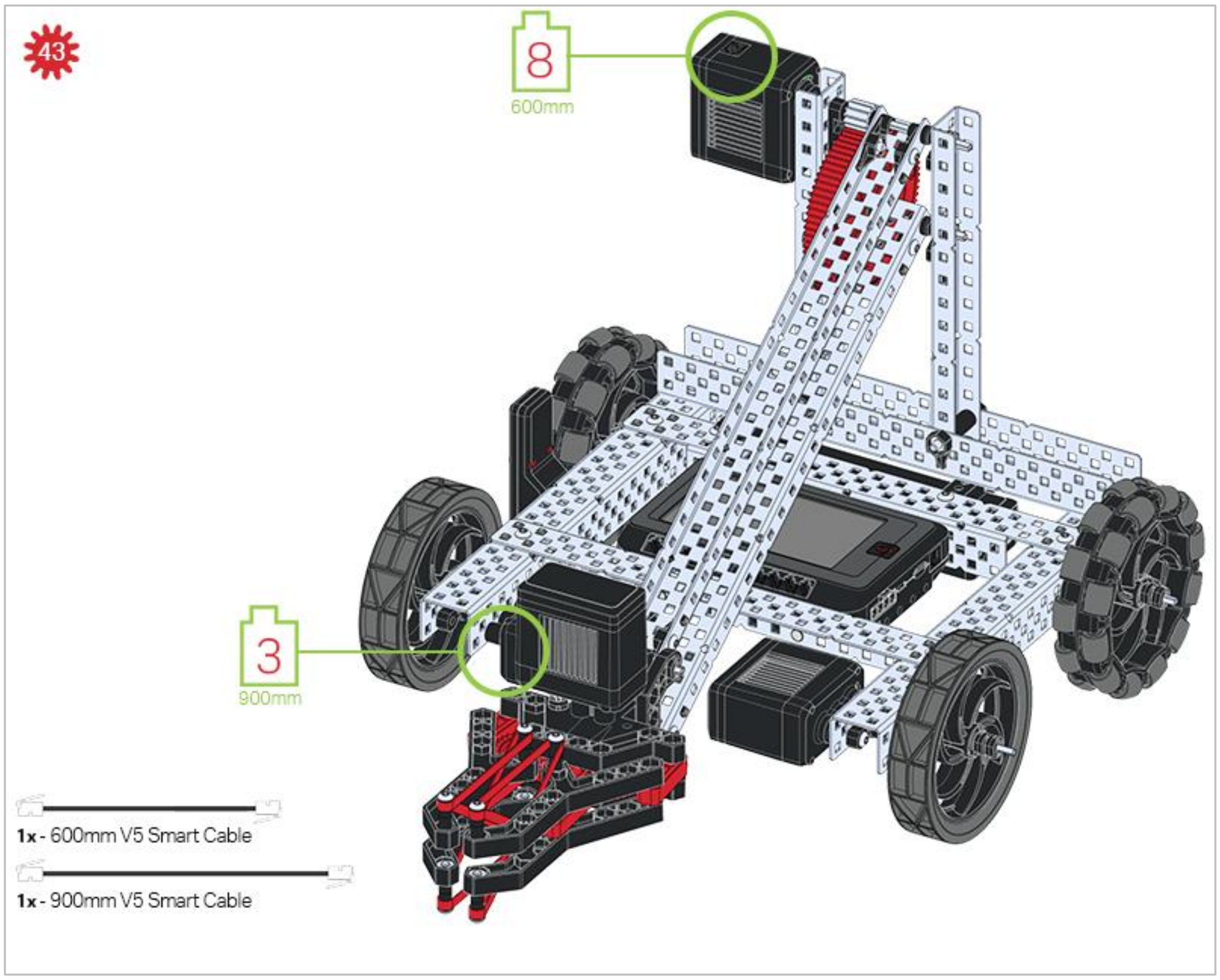
Make sure that the port on the Smart Motor is facing the right side of the robot when the claw is installed (the same side as the V5 Radio).

42



4x - Rubber Shaft Collar

2x - 2 in Shaft



Build Instruction Tips

Check the Appendix for information on how to use the new Hex Nut Retainers.

Exploration

Now that you've finished the build, test what it does. Play with your build and then answer these questions in your engineering notebook.

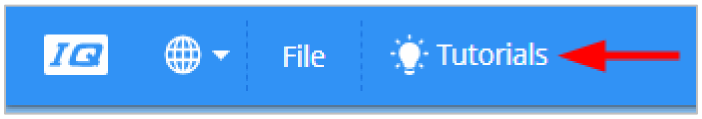
- How do you think the Clawbot could be used as a tool for measurement?
- If you didn't have a ruler to measure, which VEX piece would you choose to act as a measuring stick?
- While the Clawbot is off, lift its arm gently until it is lifted as high as it can go. Stop when you feel resistance. Measure how high the claw is off of the surface the Clawbot is on. Measure in millimeters. Why do you think it's a bad idea to move the robot at high speeds with its arm fully lifted?

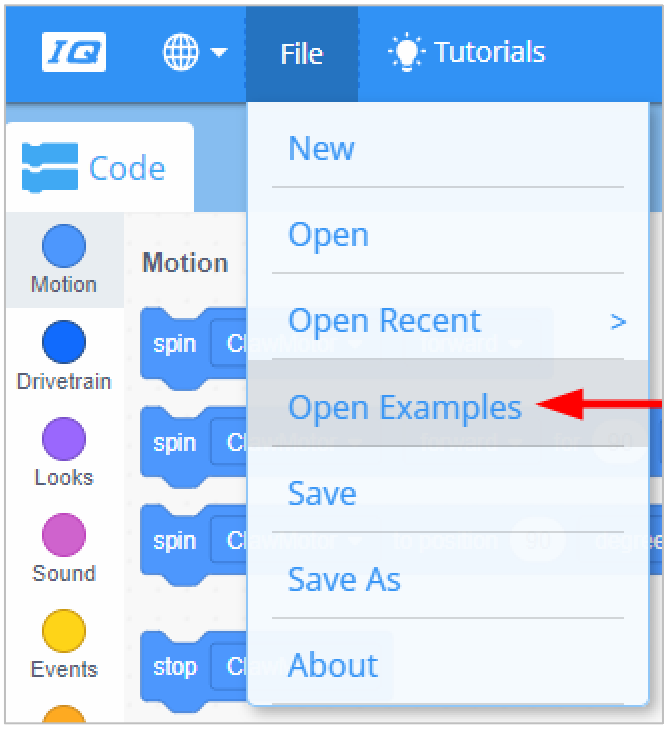
What You'll Need to Know - VEXcode V5 Blocks

In order to successfully complete this STEM Lab, you'll need to know how to program the robot to move forward, in reverse, left and right before getting started. You can use the links to other STEM Labs below, or the Tutorials, or the Example Projects in VEXcode V5 Blocks to learn how to drive and turn the robot before proceeding.

Basic Movements

- Programming Drive Forward and Reverse - VEXcode V5 Blocks
- Programming Turning Right and Left - VEXcode V5 Blocks





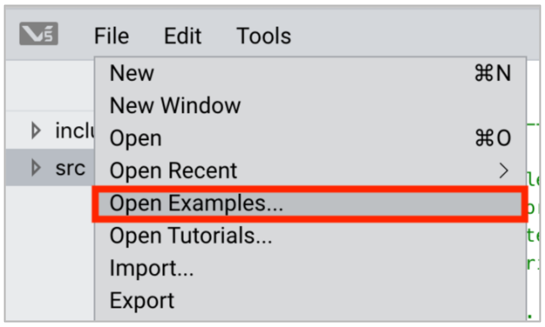
If you have not programmed your robot to drive or turn before, make sure to complete each with your robot before moving on!

What You'll Need to Know - VEXcode V5 Text

In order to successfully complete this STEM Lab, you'll need to know how to program the robot to move forward, in reverse, left and right before getting started. You can use the links to other STEM Labs below or the Example Projects in VEXcode V5 Text to learn how to drive and turn the robot before proceeding.

Basic Movements

- Programming Drive Forward and Reverse - VEXcode V5 Text
- Programming Turning Right and Left - VEXcode V5 Text

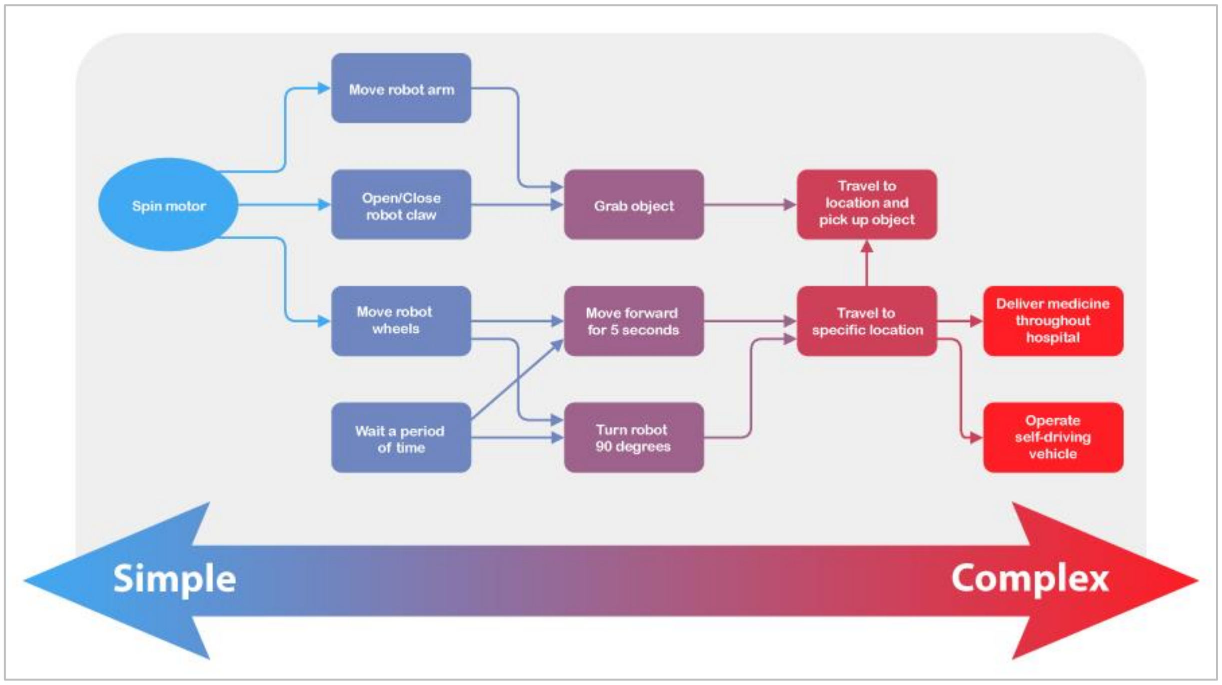


If you have not programmed your robot to drive or turn before, make sure to complete each with your robot before moving on!

SPARK
PLAY

Test your build, observe how it functions,
and fuel your logic and reasoning skills
through imaginative, creative play.

Behavior-Based Programming


Move robot arm
Spin motor
Open/Close robot claw
Grab object
Travel to location and pick up object
Move robot wheels
Move forward for 5 seconds
Travel to specific location
Deliver medicine throughout hospital
Wait a period of time
Turn robot 90 degrees
Operate self-driving vehicle
Simple
Complex

Complex behaviors rely on using multiple simple behaviors.

Programming Complexity

Robots can be designed to perform a wide range of tasks. Some of these tasks are very simple, like opening an automatic door. Others can be far more complex, like an autonomous car navigating an urban environment. No matter how complex the task is, it can be broken down into simpler tasks. These tasks are known as behaviors and are the building blocks of robotics programming.

A behavior is a way that a robot acts, and can range in complexity depending on how the robot is built or programmed. A simple mobile robot like the VEX V5 Speedbot only has two motors, while the Clawbot has four motors, including two additional motors for the Arm and Claw. Behaviors for both robots will involve turning those motors to accomplish set goals. With more design and programming, you can start from this simple behavior and do more complex behaviors.

Below is a list of robot behaviors increasing from simple to complex for both the Speedbot and the Clawbot. In parentheses, you can see the simpler behaviors that compose each one.

- Rotate a motor assigned to a certain port

- Drive forward (rotate both the left and right motors using the Drivetrain)
- Travel 5 meters (drive forward, then stop)
- Grab a distant object (travel 2 meters, rotate the claw motor to grab it)
- Retrieve an object and put it on a high shelf (grab a distant object, turn around, travel 2 meters, use the arm and claw motors to raise and release the object)

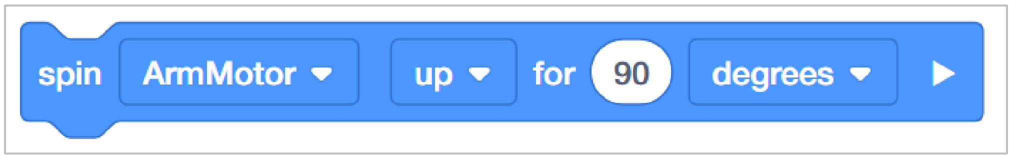
You can see how you can deconstruct any of the more complex behaviors into simpler behaviors. These become the building blocks of any complex task.

Programming the Robot Arm - VEXcode V5 Blocks

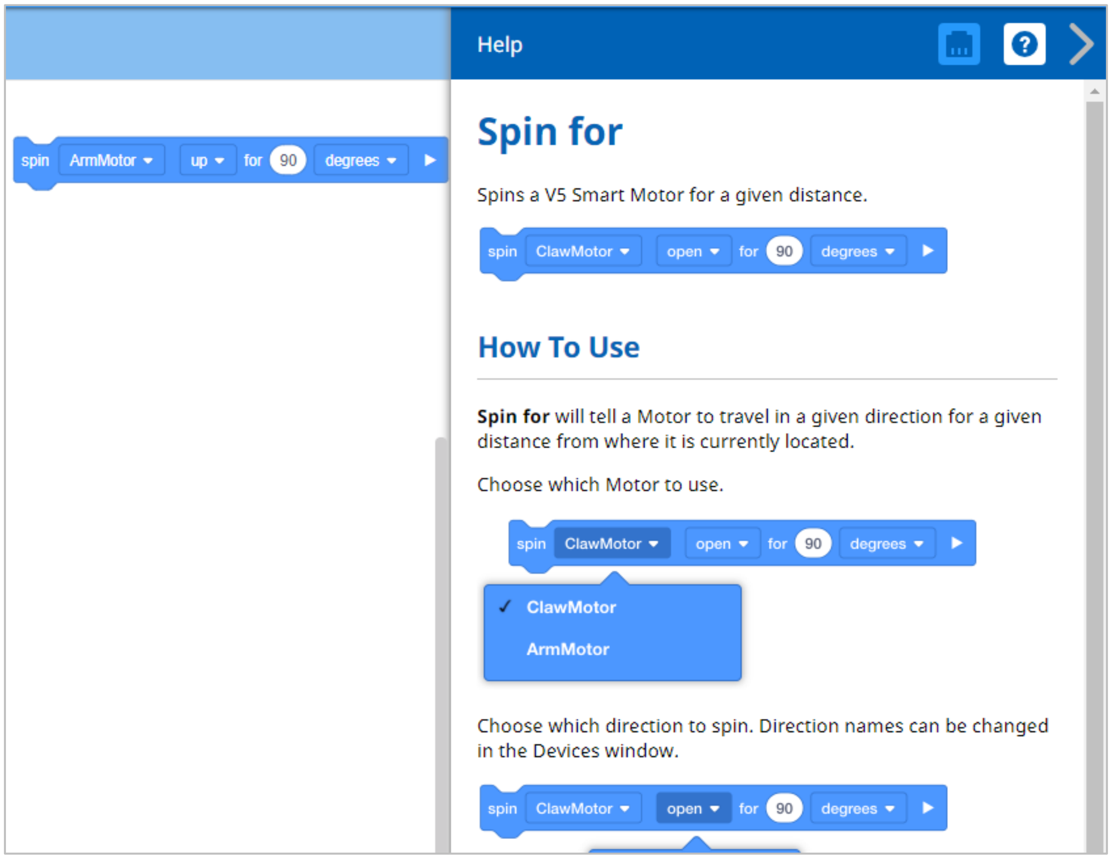
The V5 Clawbot is ready to reach!

This exploration will give you the tools to be able to start creating some cool projects that use the V5 Clawbot's arm.

- VEXcode V5 Blocks that will be used in this exploration include:



- To find out more information about the block, open the Help and then select the *spin for* block.



Help

Spin for

Spins a V5 Smart Motor for a given distance.

How To Use

Spin for will tell a Motor to travel in a given direction for a given distance from where it is currently located.

Choose which Motor to use.

Choose which direction to spin. Direction names can be changed in the Devices window.

spin ArmMotor up for 90 degrees
spin ClawMotor open for 90 degrees
ClawMotor
ArmMotor

- Make sure you have the hardware required, your engineering notebook, and VEXcode V5 Blocks downloaded and ready.

Hardware/Software Required:

Amount	Hardware/Software
1	VEX V5 Classroom Starter Kit (with up-to-date firmware)
1	VEXcode V5 Blocks (latest version, Windows, macOS, Chromebook)
1	Engineering Notebook
1	Clawbot (Drivetrain 2-motor, No Gyro) example project

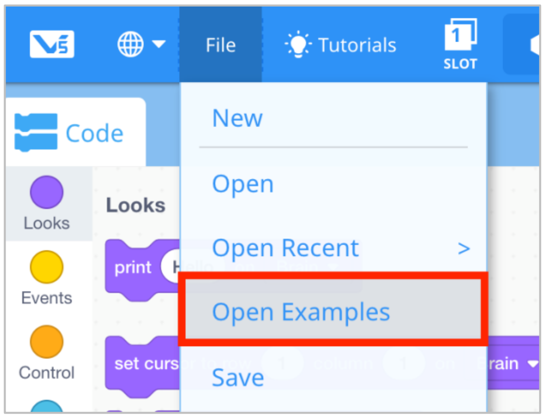
1. Preparing for the Exploration

Before you begin the activity, do you have each of these items ready? Check each of the following:

- Are the motors plugged into the correct ports?
- Are the smart cables fully inserted into all of the motors?
- Is the Brain turned on?
- Is the battery charged?

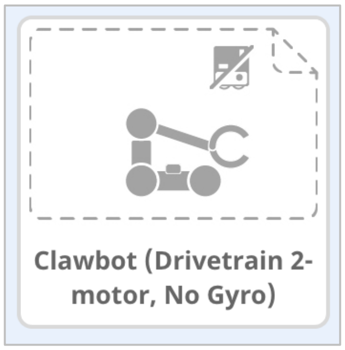
2. Start a New Project

Before you begin your project, select the correct template project. The Clawbot (Drivetrain 2-motor, No Gyro) template example project contains the Clawbot's motors configuration. If the template is not used, your robot will not run the project correctly.

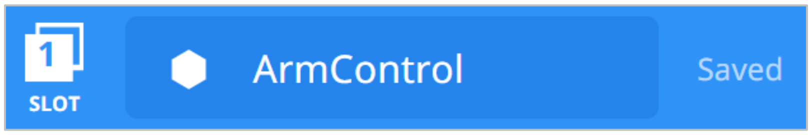


Complete the following steps:

- Open the File menu.
- Select **Open Examples**.

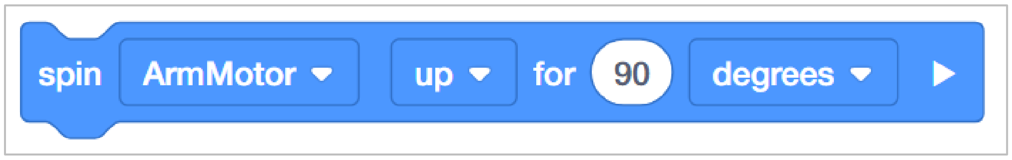


- Select and open the Clawbot (Drivetrain 2-motor, No Gyro) template example project.
- Since we will be programming to control the arm, rename your project *ArmControl*.
- **Save** your project.
- Check to make sure the project name ArmControl is now in the window in the center of the toolbar.

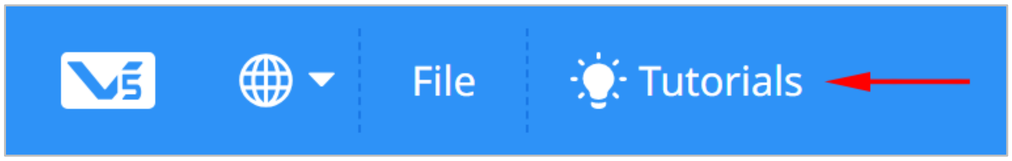


3. Move the Arm Up

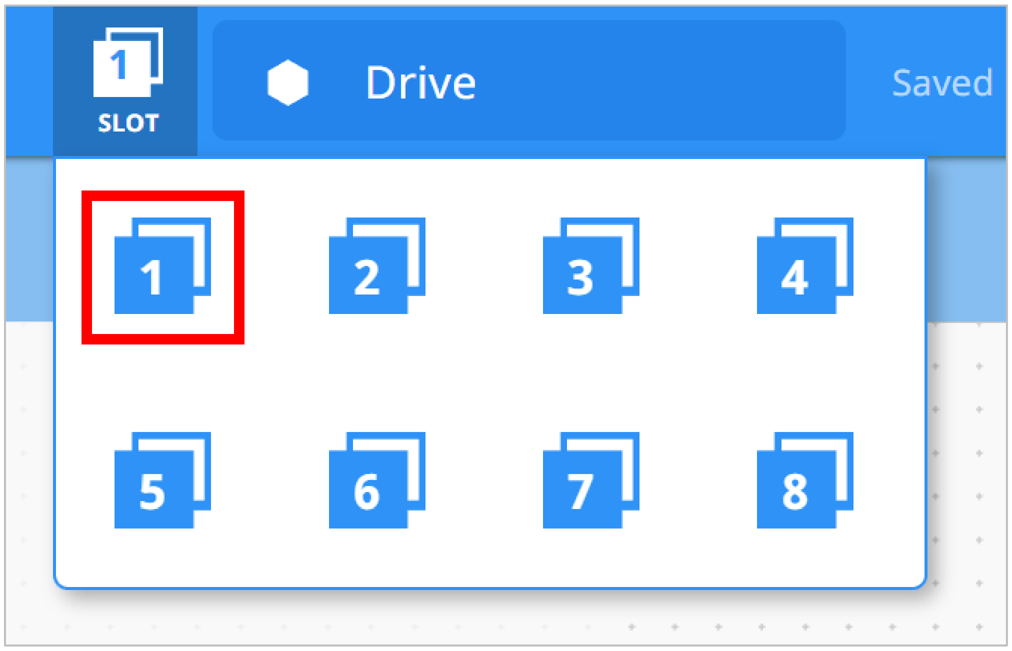
We are now going to begin by programming the arm to raise!



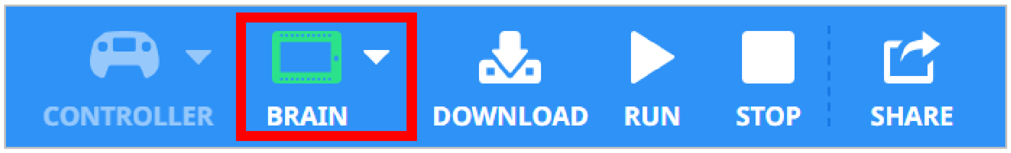
- Add the *spin for* block below the *when started* block in the programming area.



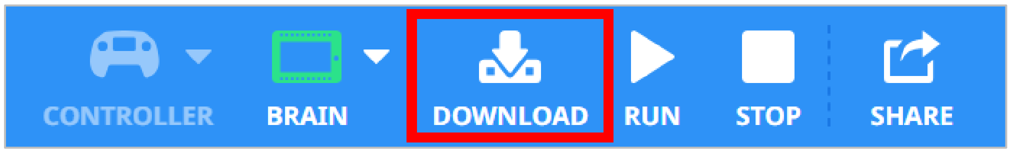
- Watch the Moving the Arm Tutorial in VEXcode V5 Blocks if you would like a demonstration.



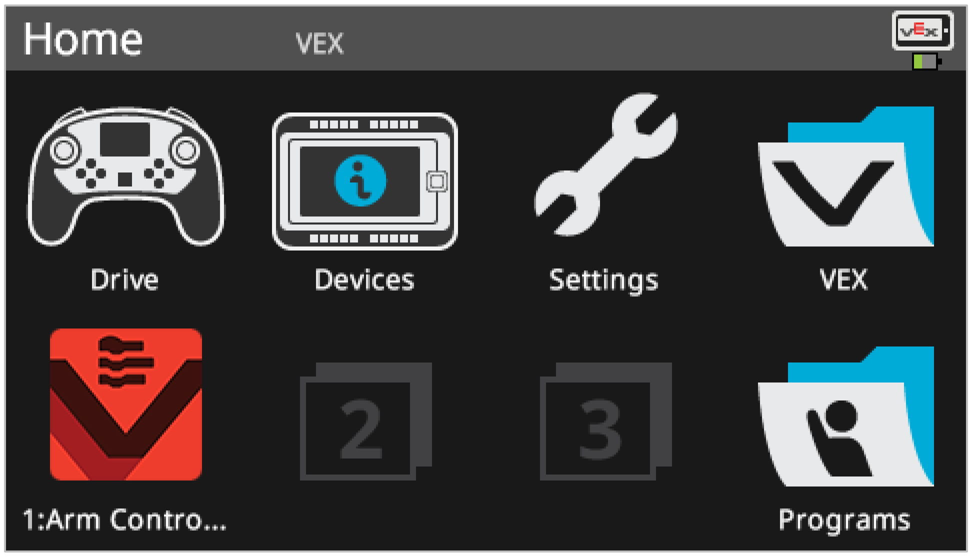
- Click on the Slot icon. You can download your project to one of the available slots in the Robot Brain. Click on Slot 1.



- Connect the robot to your computer or tablet. The Brain icon in the toolbar turns green after a successful connection has been made.



- Click the Download button on the toolbar to download the Drive project to the Robot Brain.



- Check that the ArmControl project has downloaded to the brain in the slot you chose.
- Run the project on the Clawbot by making sure the project is selected and then press the Run button. Congratulations on creating your first Claw Arm project!

4. Try This: Program the Arm Down

Now that you have programmed the arm to move up, you will now program the arm to lower or move down.

The V5 Clawbot with its arm down

- Add a *wait* block and a second *spin for* block to your ArmControl project so that the arm raises to 90 degrees, waits 3 seconds, and then returns back down.
  Hint: You will need to change the direction within the *spin for* block. For more information about the *wait* block, see VEXcode V5 Blocks' Help feature.
- Test that your revised project moves the arm up for 90 degrees, waits three seconds, and then moves the arm down for 90 degrees by downloading and running your ArmControl project.
- Notice that the arm holds its position while the *wait* block is running. The motor draws power from the battery to hold the arm up against the force of gravity. That's because the default setting for having the motor stop is the *hold* setting. There are two other settings for stopping - *brake* and *coast*. You will learn about those in another lab.

5. Complete the Flight Traffic Controller Challenge

A V5 Clawbot raising and lowering its arm

In the Flight Traffic Controller Challenge, the Clawbot must move its arm up and down each for 90 degrees, wait 3 seconds, move the arm up and down twice each for 45 degrees twice, wait 5 seconds, and then move the arm up and down three times for 90 degrees.

Here is a list of the Clawbot's behaviors:

- Move arm up and then down for 90 degrees.
- Wait 3 seconds.
- Move arm up and then down for 45 degrees.

- Move arm up and then down for 45 degrees.
- Wait 5 seconds.
- Move arm up and then down for 90 degrees.
- Move arm up and then down for 90 degrees.
- Move arm up and then down for 90 degrees.

Range of Motion

A fully open V5 Claw

Range of Motion

Robots are often comprised of several mechanical subsystems, such as arms, claws, and drivetrains. These mechanical subsystems create different types of movement, which allow the robot to complete different tasks. Each subsystem has its own range of motion, which is the term used to describe how far it can rotate or slide before hitting some sort of limit.

Subsystems such as the drivetrain usually have a full range of motion, since the motors, gears, and wheels can freely spin continuously without hitting any limit. This is important, since the robot may need to traverse substantial distances in order to complete a task.

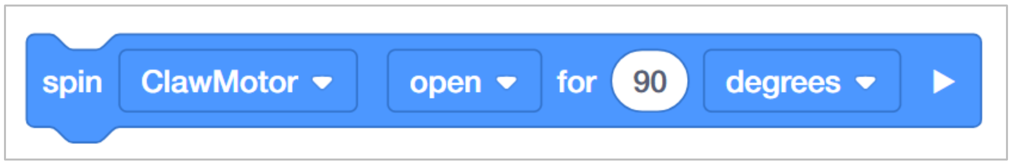
Subsystems such as claws or arms usually have a limited range of motion, which prevent them from spinning continuously. Claws can only open or close so much before reaching a mechanical limit. Likewise, the range of motion of an arm is often limited by the ground or the body of the robot itself. When working with subsystems with a limited range of motion, it is very important to stay within that range, regardless of whether you are remote controlling the robot or programming it to move autonomously. Continuing to provide power to the motors once a subsystem has reached a limit will cause unnecessary stress on the motor and any connected components.

Programming the Claw - VEXcode V5 Blocks

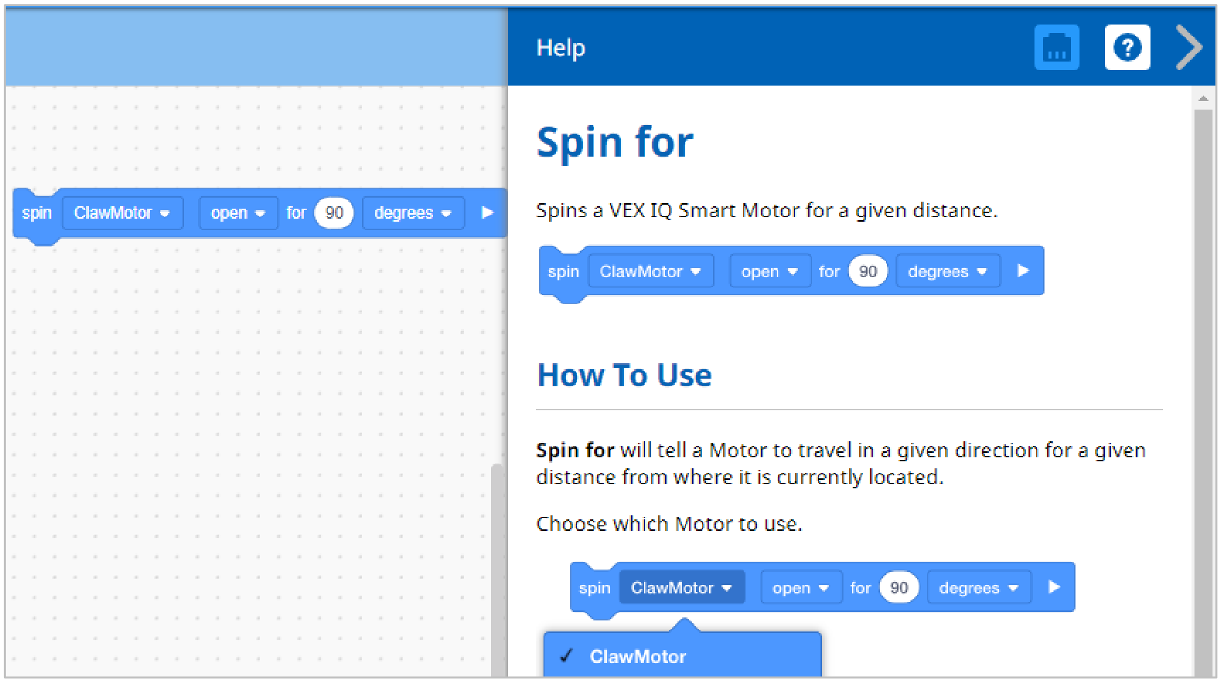
The V5 Clawbot is ready to grasp!

This exploration will let you start creating some cool projects that use the V5 Clawbot's claw to grasp objects.

- VEXcode V5 Blocks that will be used in this exploration include:



- To find out more information about the block, open the Help and then select the *spin for* block.



Spin for

Spins a VEX IQ Smart Motor for a given distance.

How To Use

Spin for will tell a Motor to travel in a given direction for a given distance from where it is currently located.

Choose which Motor to use.

✓ ClawMotor

- Make sure you have the hardware required, your engineering notebook, and VEXcode V5 Blocks downloaded and ready.

Hardware/Software Required:

Amount	**Hardware/Software**
1	VEX V5 Classroom Starter Kit (with up-to-date firmware)

Amount	Hardware/Software
1	VEXcode V5 Blocks (latest version, Windows, macOS,Chromebook)
1	Engineering Notebook
1	Clawbot (Drivetrain 2-motor, No Gyro) template example project
1	Aluminum can

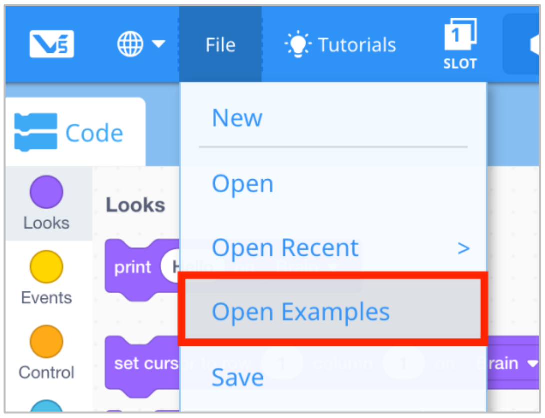
1. Preparing for the Exploration

Before you begin the activity, do you have each of these items ready? Check each of the following:

- Are the motors plugged into the correct ports?
- Are the smart cables fully inserted into all of the motors?
- Is the Brain turned on?
- Is the battery charged?

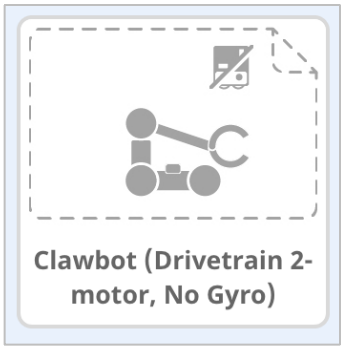
2. Start a New Project

Before you begin your project, select the correct template project. The Clawbot (Drivetrain 2-motor, No Gyro) template example project contains the Clawbot's motor configuration. If the template is not used, your robot will not run the project correctly.

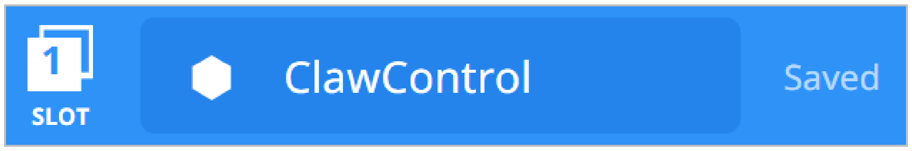


Complete the following steps:

- Open the File menu.
- Select **Open Examples**.

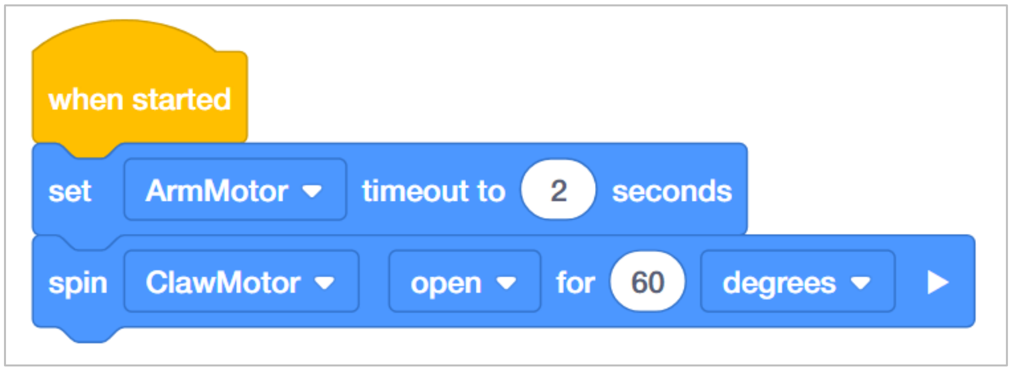


- Select and open the Clawbot (Drivetrain 2-motor, No Gyro) template example project.
- Since we will be programming to control the claw, rename your project *ClawControl*.
- **Save** your project.
- Check to make sure the project name ClawControl is now in the window in the center of the toolbar.



3. Program the V5 Claw to Open

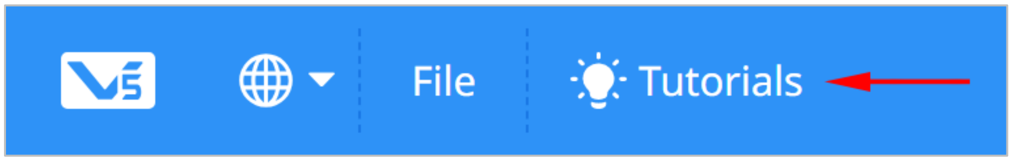
We are now going to begin by programming the claw to open!



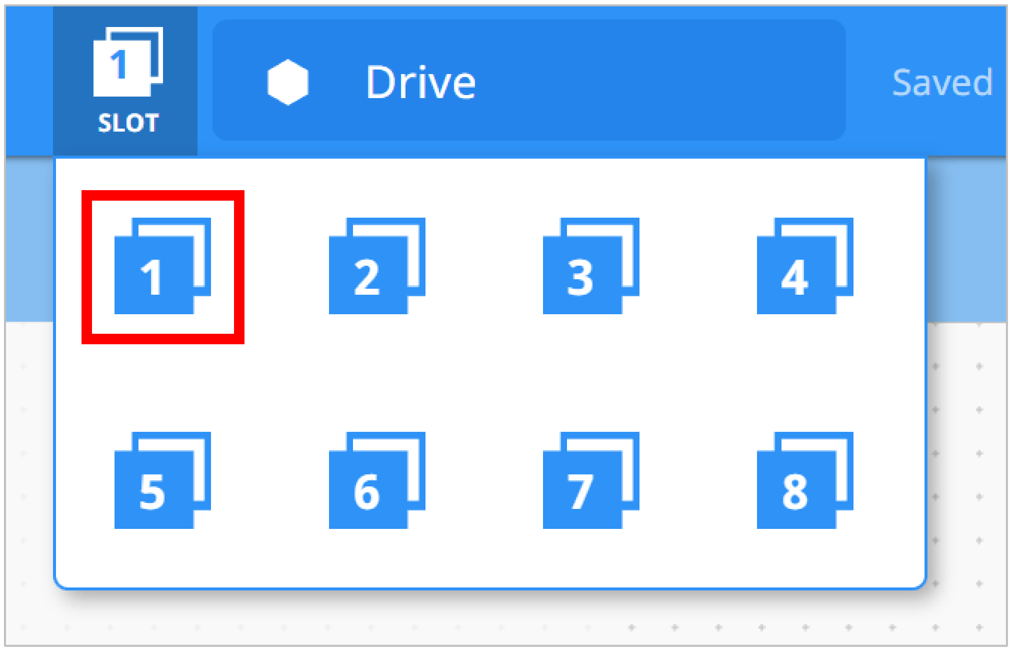
```
when started
set ArmMotor ▾ timeout to 2 seconds
spin ClawMotor ▾ open ▾ for 60 degrees ▾ ▶
```

- Add the *set motor timeout* and *spin for* blocks to the when started block and set their parameters as shown above.
- Notice that the *set motor timeout* block appears first in the stack. It sets how long the Claw Motor can run and stops it after that time.

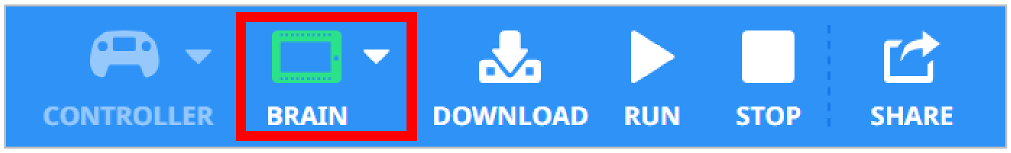
- The *set motor timeout* block in this project is set to 2 seconds. So even if the motor does not move a full 60 degrees, the project stops the Claw Motor after 2 seconds have passed.



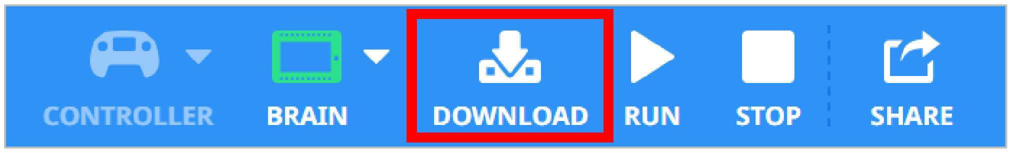
- Watch the Opening the Claw Tutorial in VEXcode V5 Blocks if you would like a demonstration.



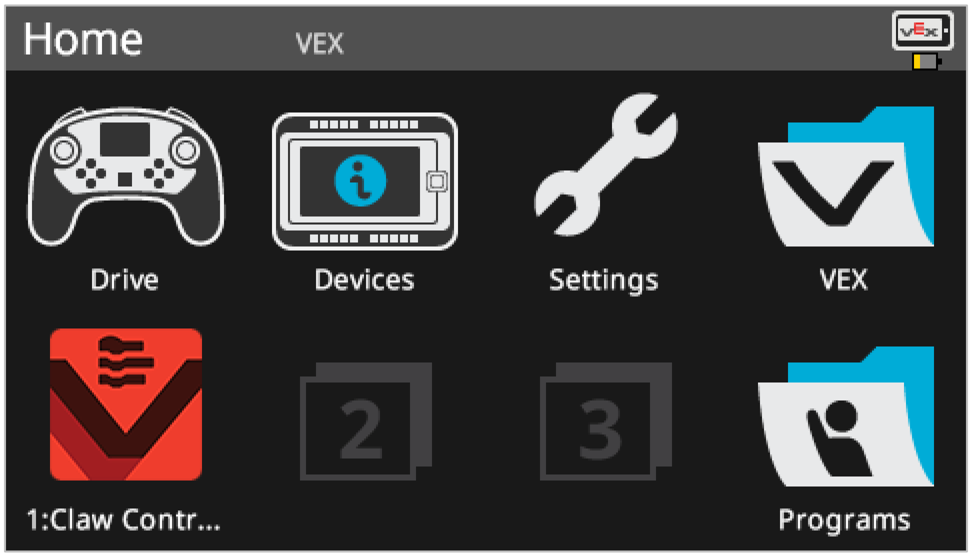
- Click on the Slot icon. You can download your project to one of the available slots in the Robot Brain. Click on Slot 1.



- Connect the robot to your computer or tablet. The Brain icon in the toolbar turns green after a successful connection has been made.



- Click the Download button on the toolbar to download the Drive project to the Robot Brain.

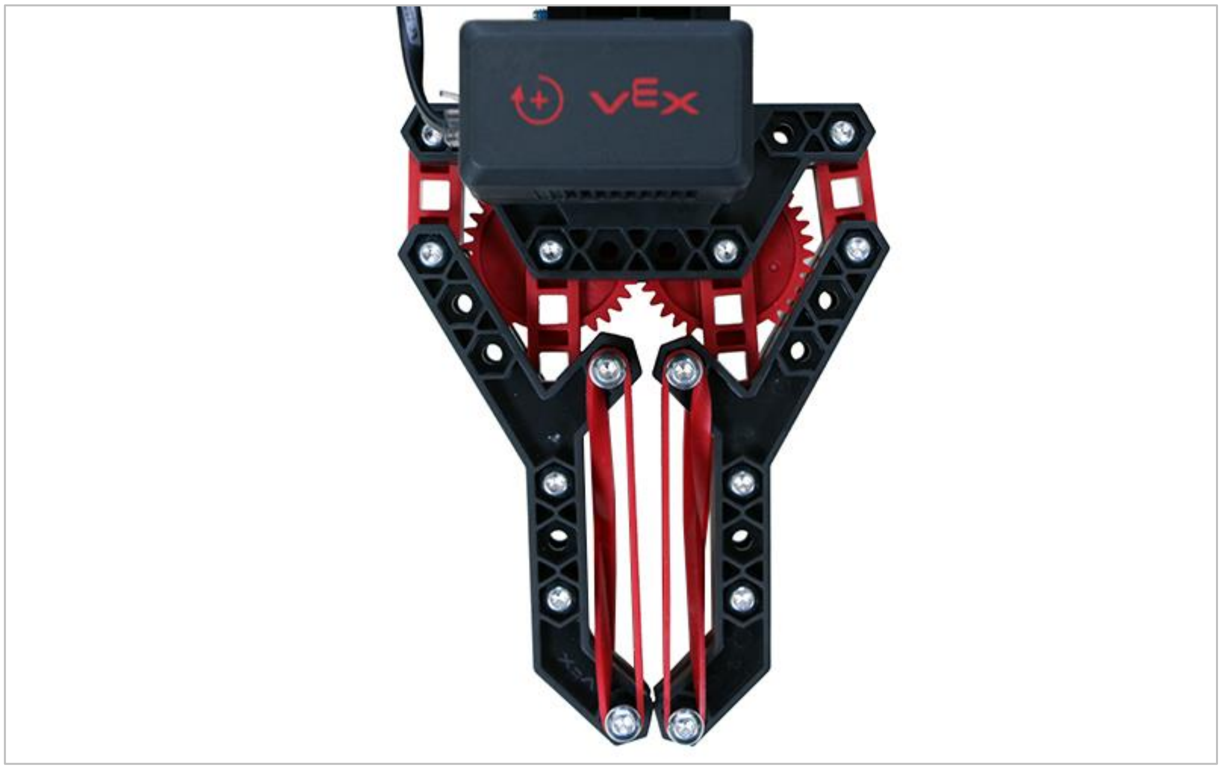


- Check that the ClawControl project has downloaded to the brain in the slot you chose.

- Run the project on the Clawbot by making sure the project is selected and then press the Run button. Congratulations on creating your first project for moving the Claw!

4. Try This: Close the V5 Claw

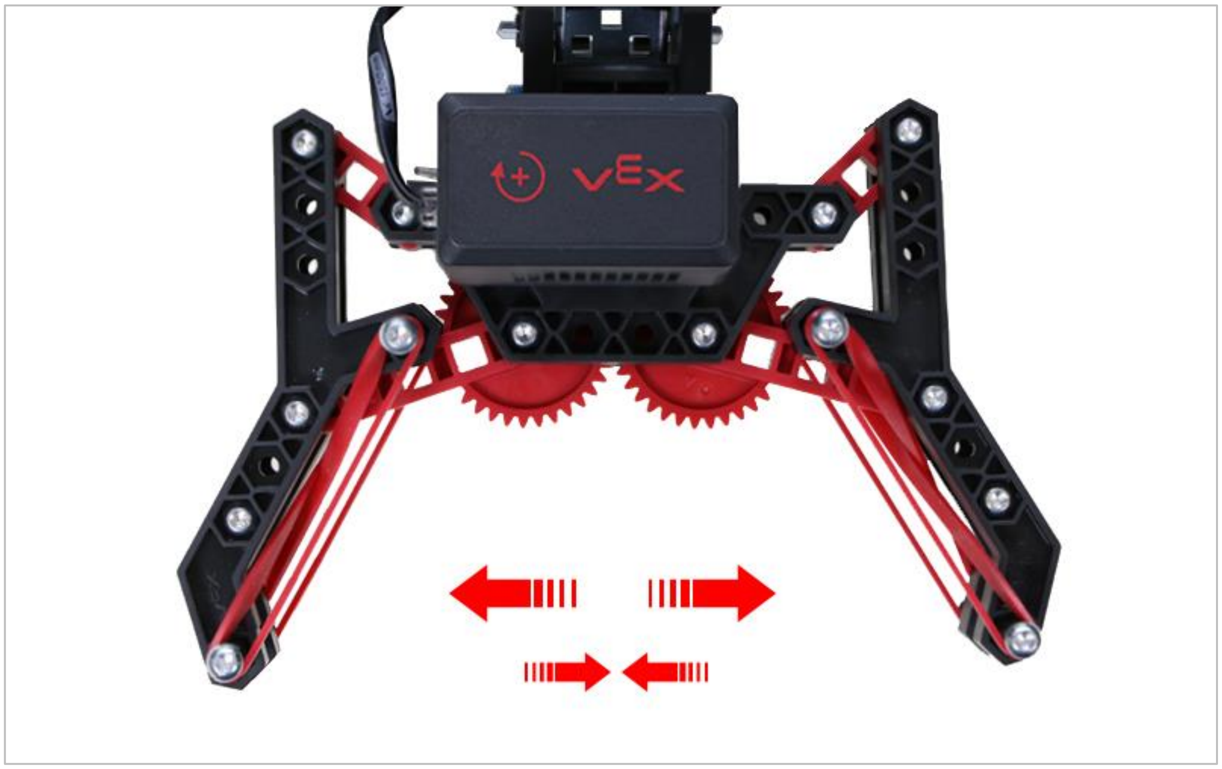
Now that you have programmed the claw to open, you will now program it to close.


V5 Claw closed

Now that you can open the claw, you will want to close it as well.

- Return to your ClawControl project and add a *spin for* block to have the Claw Motor spin closed for 30 degrees. The claw should close half of the way because it spun open for 60 degrees.
- Test that your revised project has the claw open for 60 degrees and then close for 30 degrees by downloading and running your ClawControl project.

5. Try This: Sequencing Multiple Movements



V5 Claw fully open

Not everything that you pick up with the claw will be the same size. Try opening the claw to different positions along the range of motion.

- Spin the motor in order to:
  - Open for 70 degrees
  - Close for 20 degrees
  - Open for 10 degrees
  - Close for 30 degrees
  - Close for 25 degrees
- Remember to use the *set motor timeout* block to set a timeout of 2 seconds.

- If the Claw Motor starts at 0 degrees, how many degrees is the Claw Motor open at the end of the project?

6. Completing the Lock Tight Challenge

Claw holding an aluminum can

The Lock Tight Challenge

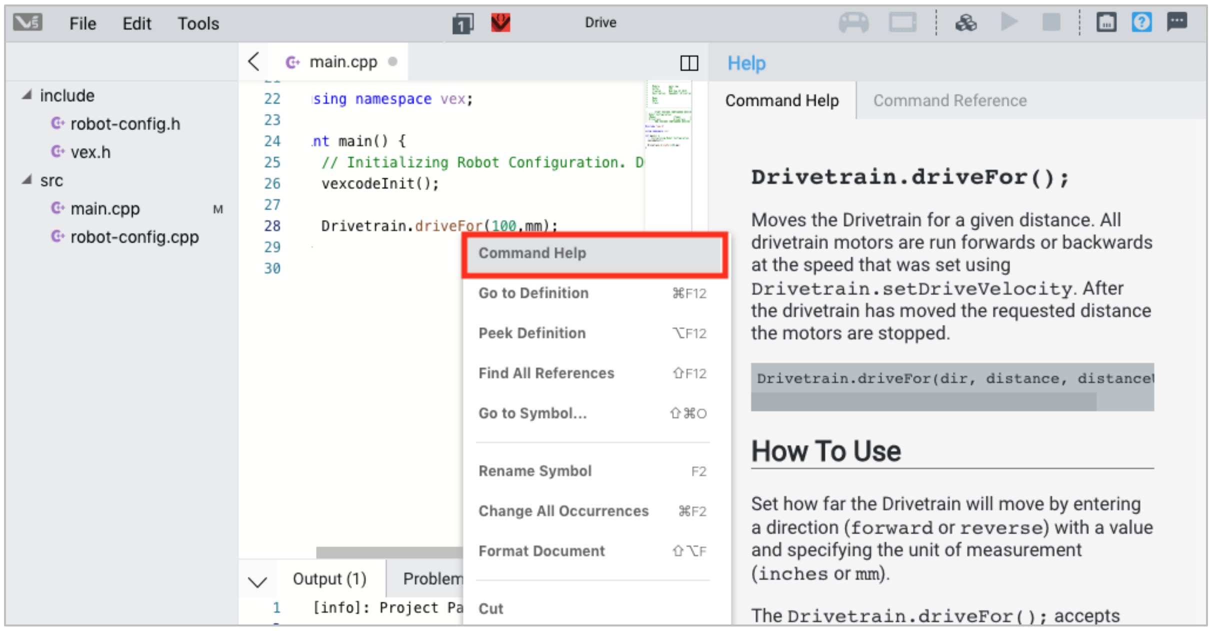
- Program the Clawbot to securely close the claw on an empty 12-ounce aluminum can without crushing the sides.
- Have the Clawbot hold onto the can while it raises and lowers its arm for 45 degrees.
- The Clawbot should then release the can and back away from it.
- Start the challenge with an opened claw and an empty can within it.

Programming the Robot Arm - VEXcode V5 Text

The V5 Clawbot is ready to reach!

This exploration will give you the tools to be able to start creating some cool projects that use the V5 Clawbot's arm.

- VEXcode V5 Text instructions that will be used in this exploration:
  - ArmMotor.spinFor(90,degrees);
  - ArmMotor.setPosition(0, degrees);
  - wait(2, seconds);
- To access additional information, right click on a command name in your workspace to see help for that command.



```
sing namespace vex;

nt main() {
  // Initializing Robot Configuration. D
  vexcodeInit();

  Drivetrain.driveFor(100,mm);
```

```
[info]: Project Pa
```

Drivetrain.driveFor();

Moves the Drivetrain for a given distance. All drivetrain motors are run forwards or backwards at the speed that was set using Drivetrain.setDriveVelocity. After the drivetrain has moved the requested distance the motors are stopped.

```
Drivetrain.driveFor(dir, distance, distance
```

How To Use

Set how far the Drivetrain will move by entering a direction (forward or reverse) with a value and specifying the unit of measurement (inches or mm).

The Drivetrain.driveFor(); accepts

- Make sure you have the hardware required, your engineering notebook, and VEXcode V5 Text downloaded and ready.

Hardware/Software Required:

Amount	Hardware/Software
1	VEX V5 Classroom Starter Kit (with up-to-date firmware)
1	VEXcode V5 Text (latest version, Windows or macOS,)

Amount	Hardware/Software
1	Engineering Notebook
1	Clawbot Template (Drivetrain 2-motor, No Gyro) example project

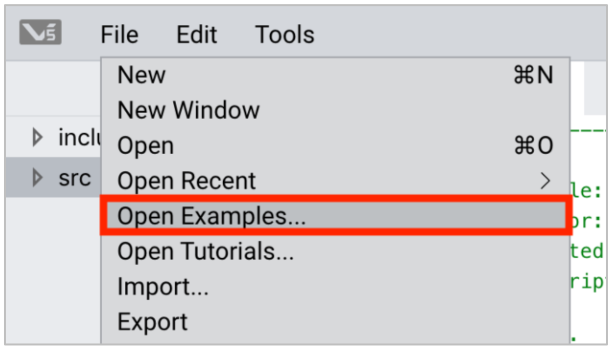
1. Preparing for the Exploration

Before you begin the activity, do you have each of these items ready? Check each of the following:

- Are the motors plugged into the correct ports?
- Are the smart cables fully inserted into all of the motors?
- Is the Brain turned on?
- Is the battery charged?

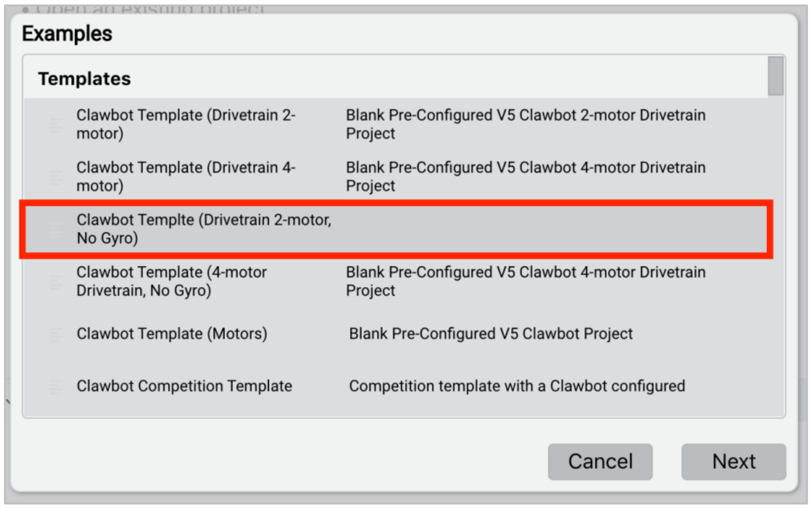
2. Start a New Project

Before you begin your project, select the correct template project. The Clawbot Template (Drivetrain 2-motor, No Gyro) example project contains the Clawbot's motors configuration. If the template is not used, your robot will not run the project correctly.



Complete the following steps:

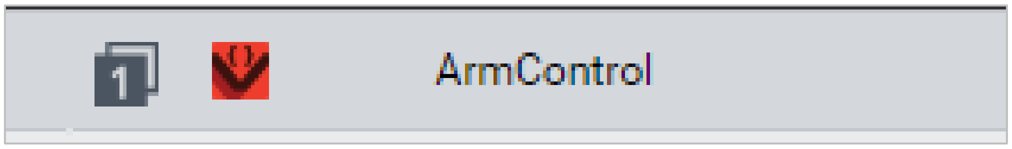
- Open the File menu.
- Select **Open Examples**.


Examples

Templates

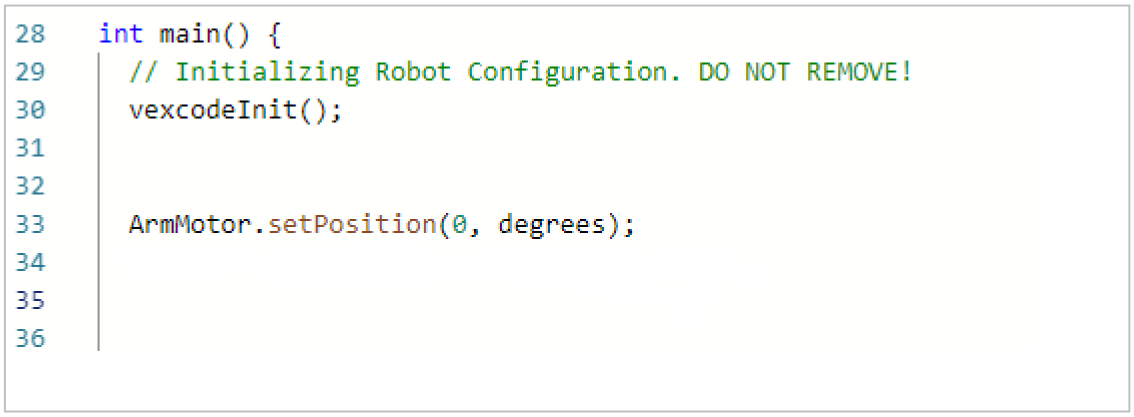
Clawbot Template (Drivetrain 2-motor)	Blank Pre-Configured V5 Clawbot 2-motor Drivetrain Project
Clawbot Template (Drivetrain 4-motor)	Blank Pre-Configured V5 Clawbot 4-motor Drivetrain Project
Clawbot Templte (Drivetrain 2-motor, No Gyro)	
Clawbot Template (4-motor Drivetrain, No Gyro)	Blank Pre-Configured V5 Clawbot 4-motor Drivetrain Project
Clawbot Template (Motors)	Blank Pre-Configured V5 Clawbot Project
Clawbot Competition Template	Competition template with a Clawbot configured

- Select and open the Clawbot Template (Drivetrain 2-motor, No Gyro) example project.
- Since we will be programming to control the arm, rename your project *ArmControl*.
- **Save** your project.
- Check to make sure the project name ArmControl is now in the window in the center of the toolbar.



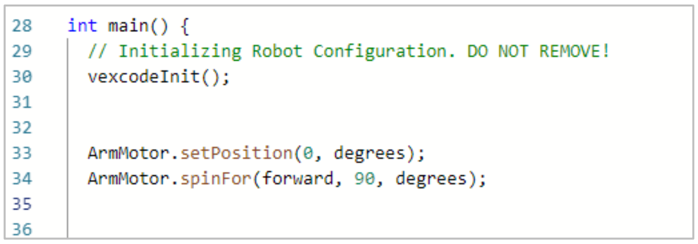
3. Move the Arm Up

We are now going to begin by programming the arm to raise!



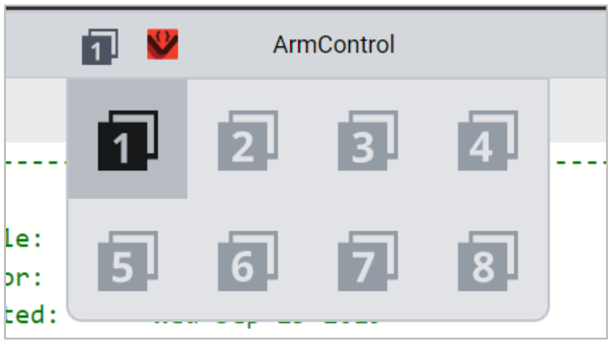
```
int main() {
  // Initializing Robot Configuration. DO NOT REMOVE!
  vexcodeInit();


  ArmMotor.setPosition(0, degrees);



```

- Write the ArmMotor.setPosition(); instruction as shown above in the programming area to set the starting position for the arm.

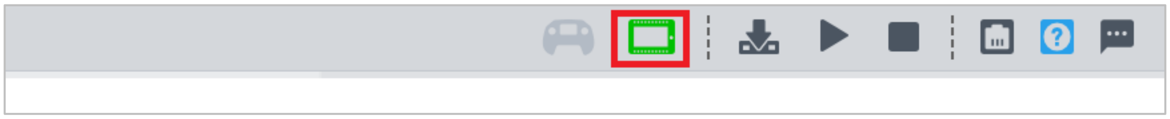


```
int main() {
  // Initializing Robot Configuration. DO NOT REMOVE!
  vexcodeInit();


  ArmMotor.setPosition(0, degrees);
  ArmMotor.spinFor(forward, 90, degrees);

```

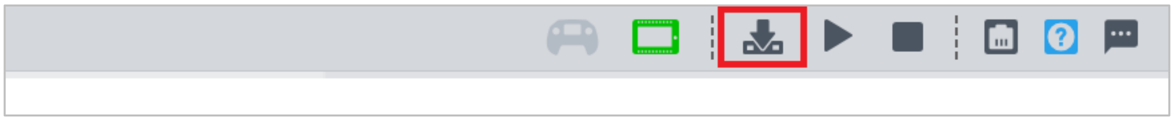
- Add the ArmMotor.spinFor(); instruction as shown above to move the arm up.



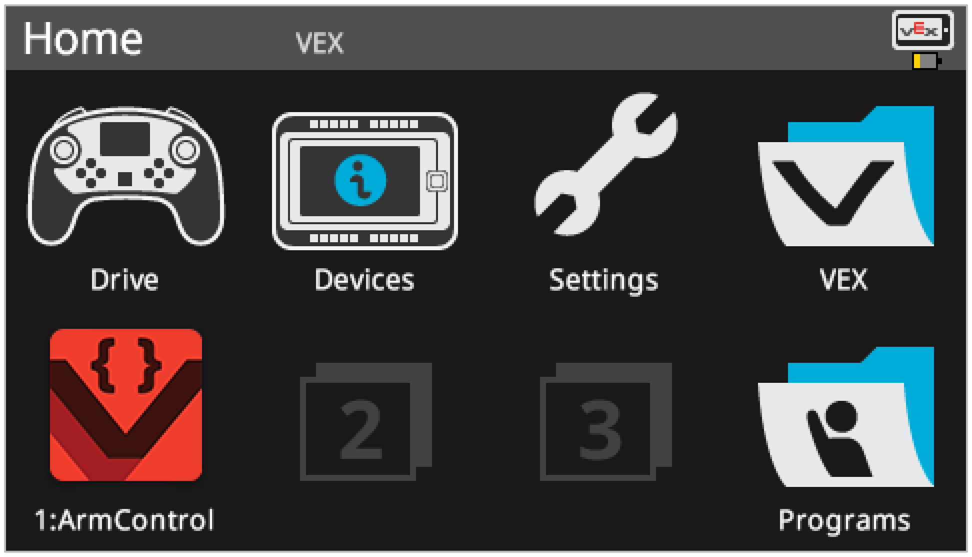
- Click on the Slot icon. You can download your project to one of the available slots in the Robot Brain. Click on Slot 1.



- Connect the robot to your computer or tablet. The Brain icon in the toolbar turns green after a successful connection has been made.



- Click the Download button on the toolbar to download the Drive project to the Robot Brain.



- Check that the ArmControl project has downloaded to the brain in the slot you chose.

- Run the project on the Clawbot by making sure the project is selected and then press the Run button. Congratulations on creating your first Claw Arm project!

4. Try This: Program the Arm Down

Now that you have programmed the arm to move up, you will now program the arm to lower or move down.

The V5 Clawbot with its arm down

- Add a second ArmMotor.spinFor(); instruction to your ArmControl project so that the arm raises to 90 degrees, waits 2 seconds, and then returns back down.
    - **Hint:** You will need to change the direction within the instruction.
- Notice that the arm holds its position while the *wait* command is running. The motor draws power from the battery to hold the arm up against the force of gravity. That's because the default setting for having the motor stop is the *hold* setting. There are two other settings for stopping - *brake* and *coast*. You will learn about those in another lab.

5. Complete the Flight Traffic Controller Challenge

A V5 Clawbot raising and lowering its arm

In the Flight Traffic Controller Challenge, the Clawbot must move its arm up and down once for 90 degrees, wait 3 seconds, move the arm up and down two times for 45 degrees, wait 5 seconds, and then move the arm up and down three times for 90 degrees.

Here is a list of the Clawbot's behaviors:

- Move arm up and then down for 90 degrees.
- Wait 3 seconds.
- Move arm up and then down for 45 degrees.

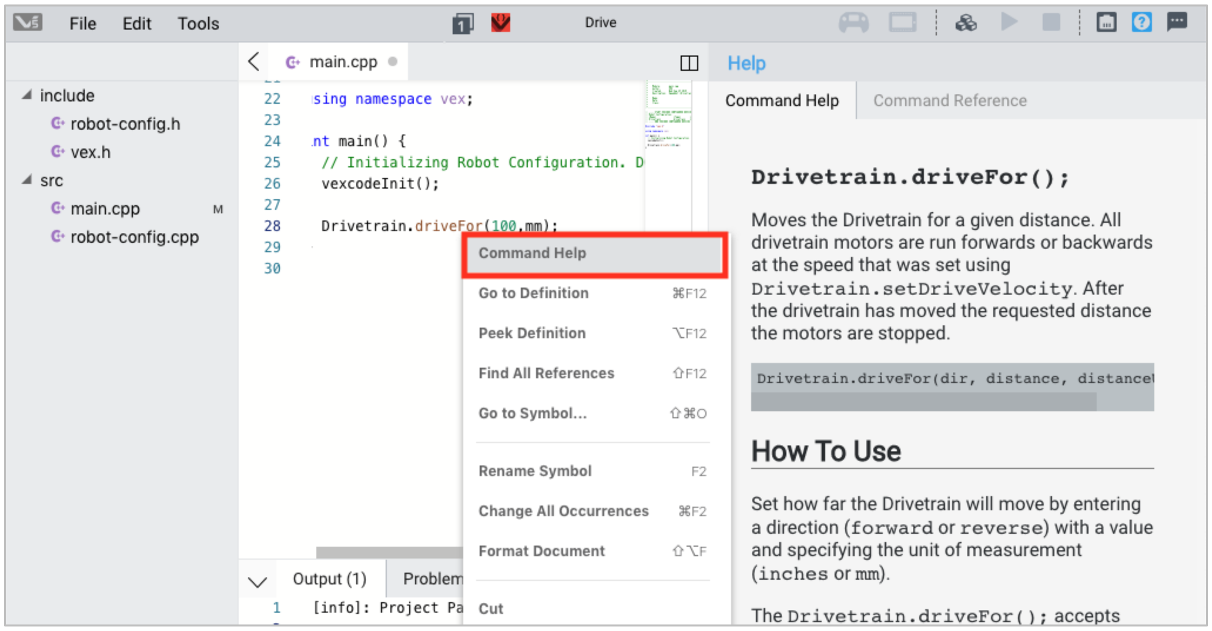
- Move arm up and then down for 45 degrees.
- Wait 5 seconds.
- Move arm up and then down for 90 degrees.
- Move arm up and then down for 90 degrees.
- Move arm up and then down for 90 degrees.

Programming the Claw - VEXcode V5 Text

The V5 Clawbot is ready to grasp!

This exploration will let you start creating some cool projects that use the V5 Clawbot's claw to grasp objects.

- VEXcode V5 Text Instructions that will be used in this exploration include:
  - ClawMotor.setPosition(0, degrees);
  - ClawMotor.spinFor(90, degrees);
  - ClawMotor.setTimeout (2, seconds);
- To access additional information, right click on a command name in your workspace to see help for that command.



```
sing namespace vex;

nt main() {
  // Initializing Robot Configuration. D
  vexcodeInit();

  Drivetrain.driveFor(100,mm);
```

```
[info]: Project Pa
```

Drivetrain.driveFor();

Moves the Drivetrain for a given distance. All drivetrain motors are run forwards or backwards at the speed that was set using `Drivetrain.setDriveVelocity`. After the drivetrain has moved the requested distance the motors are stopped.

```
Drivetrain.driveFor(dir, distance, distance
```

How To Use

Set how far the Drivetrain will move by entering a direction (`forward` or `reverse`) with a value and specifying the unit of measurement (`inches` or `mm`).

The `Drivetrain.driveFor();` accepts

- Make sure you have the hardware required, your engineering notebook, and VEXcode V5 Text downloaded and ready.

Hardware/Software Required:

Amount	Hardware/Software
1	VEX V5 Classroom Starter Kit (with up-to-date firmware)
1	VEXcode V5 Text (latest version, Windows or macOS)

Amount	Hardware/Software
1	Engineering Notebook
1	Clawbot Template (Drivetrain 2-motor, No Gyro) example project
1	Aluminum can

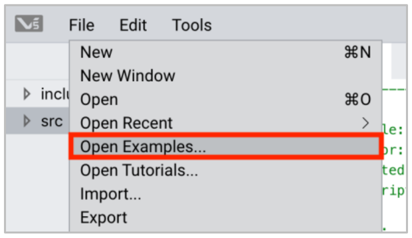
1. Preparing for the Exploration

Before you begin the activity, do you have each of these items ready? Check each of the following:

- Are the motors plugged into the correct ports?
- Are the smart cables fully inserted into all of the motors?
- Is the Brain turned on?
- Is the battery charged?

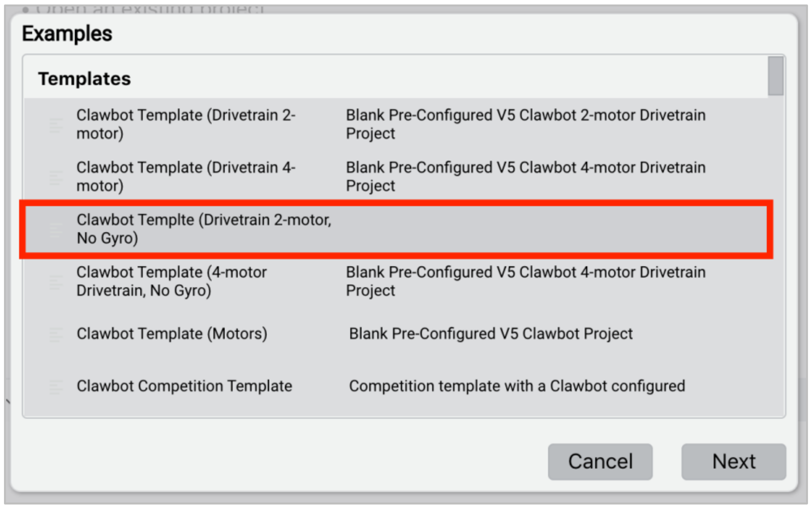
2. Start a New Project

Before you begin your project, select the correct template project. The Clawbot Template (Drivetrain 2-motor, No Gyro) example project contains the Clawbot's motor configuration. If the template is not used, your robot will not run the project correctly.



Complete the following steps:

- Open the File menu.
- Select **Open Examples**.

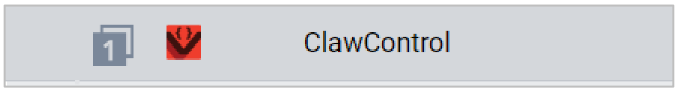


Examples

Templates

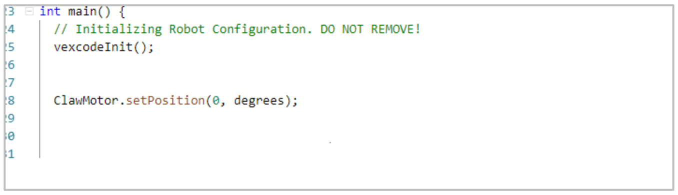
Clawbot Template (Drivetrain 2-motor)	Blank Pre-Configured V5 Clawbot 2-motor Drivetrain Project
Clawbot Template (Drivetrain 4-motor)	Blank Pre-Configured V5 Clawbot 4-motor Drivetrain Project
Clawbot Templte (Drivetrain 2-motor, No Gyro)	
Clawbot Template (4-motor Drivetrain, No Gyro)	Blank Pre-Configured V5 Clawbot 4-motor Drivetrain Project
Clawbot Template (Motors)	Blank Pre-Configured V5 Clawbot Project
Clawbot Competition Template	Competition template with a Clawbot configured

- Select and open the Clawbot Template (Drivetrain 2-motor, No Gyro) example project.
- Since we will be programming to control the claw, rename your project *ClawControl*.
- **Save** your project.
- Check to make sure the project name ClawControl is now in the window in the center of the toolbar.



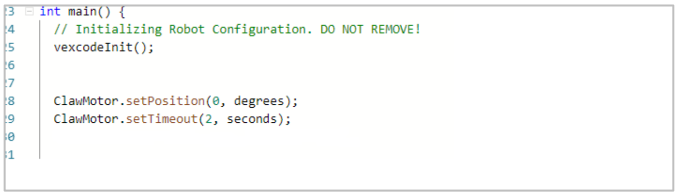
3. Program the V5 Claw to Open

We are now going to begin by programming the claw to open!



```
int main() {
  // Initializing Robot Configuration. DO NOT REMOVE!
  vexcodeInit();


  ClawMotor.setPosition(0, degrees);


```

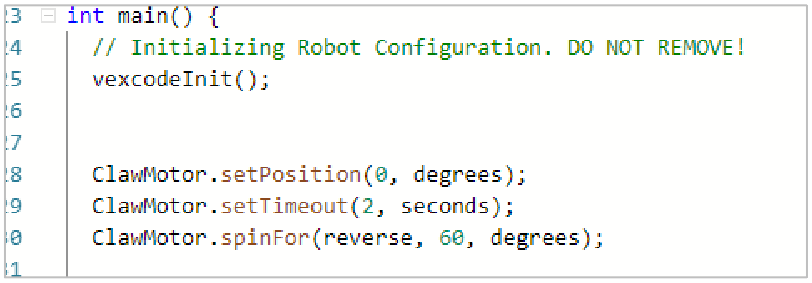
- Write the **ClawMotor.setPosition();** instruction as shown above in the programming area to set the starting position for the claw.



```
int main() {
  // Initializing Robot Configuration. DO NOT REMOVE!
  vexcodeInit();


  ClawMotor.setPosition(0, degrees);
  ClawMotor.setTimeout(2, seconds);

```

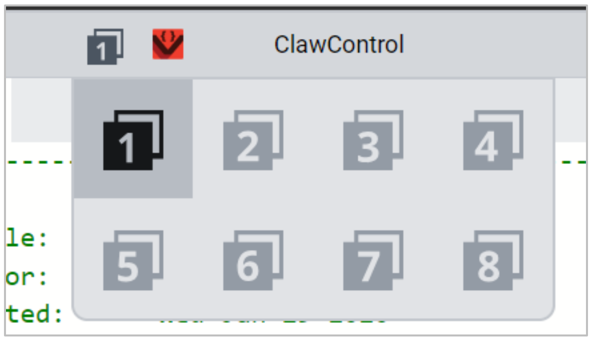
- Add the **ClawMotor.setTimeout();** instruction as shown above.

- Notice that this instruction appears before the **ClawMotor.spinFor();** instruction. Its purpose is to set how long the Claw motor can run and stops it after that time. So even if the motor doesn't move a full 60 degrees, the project stops the Claw motor after two seconds have passed.

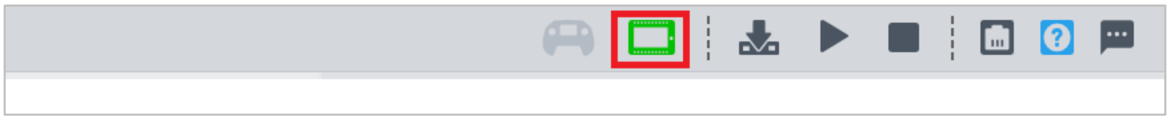


```
int main() {
  // Initializing Robot Configuration. DO NOT REMOVE!
  vexcodeInit();


  ClawMotor.setPosition(0, degrees);
  ClawMotor.setTimeout(2, seconds);
  ClawMotor.spinFor(reverse, 60, degrees);
```

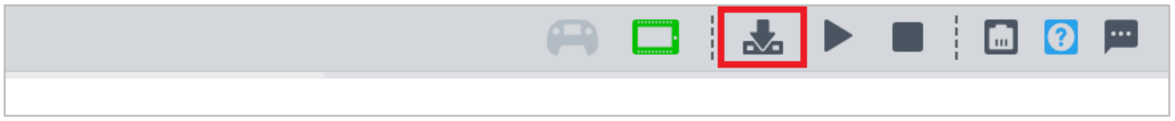
- Write the **ClawMotor.spinFor();** instruction as shown above to open the claw 60 degrees.



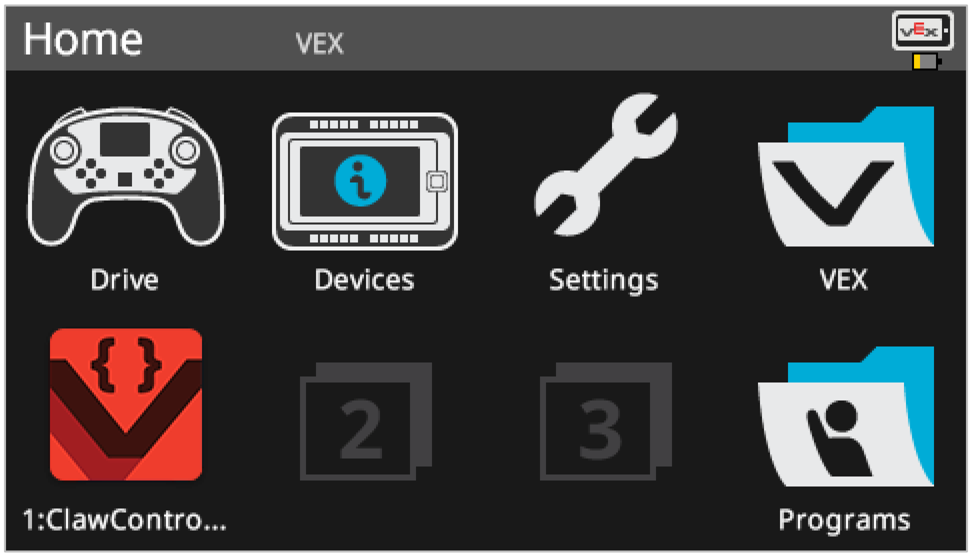
- Select the slot that the project will be downloaded to on the V5 Robot Brain by clicking on the boxed 1 icon to open up all available project slots and select the desired slot.



- Connect the robot to your computer or tablet. The Brain icon in the toolbar turns green after a successful connection has been made.



- Click on the Download icon next to the Device Info icon to download the project to the V5 Robot Brain.

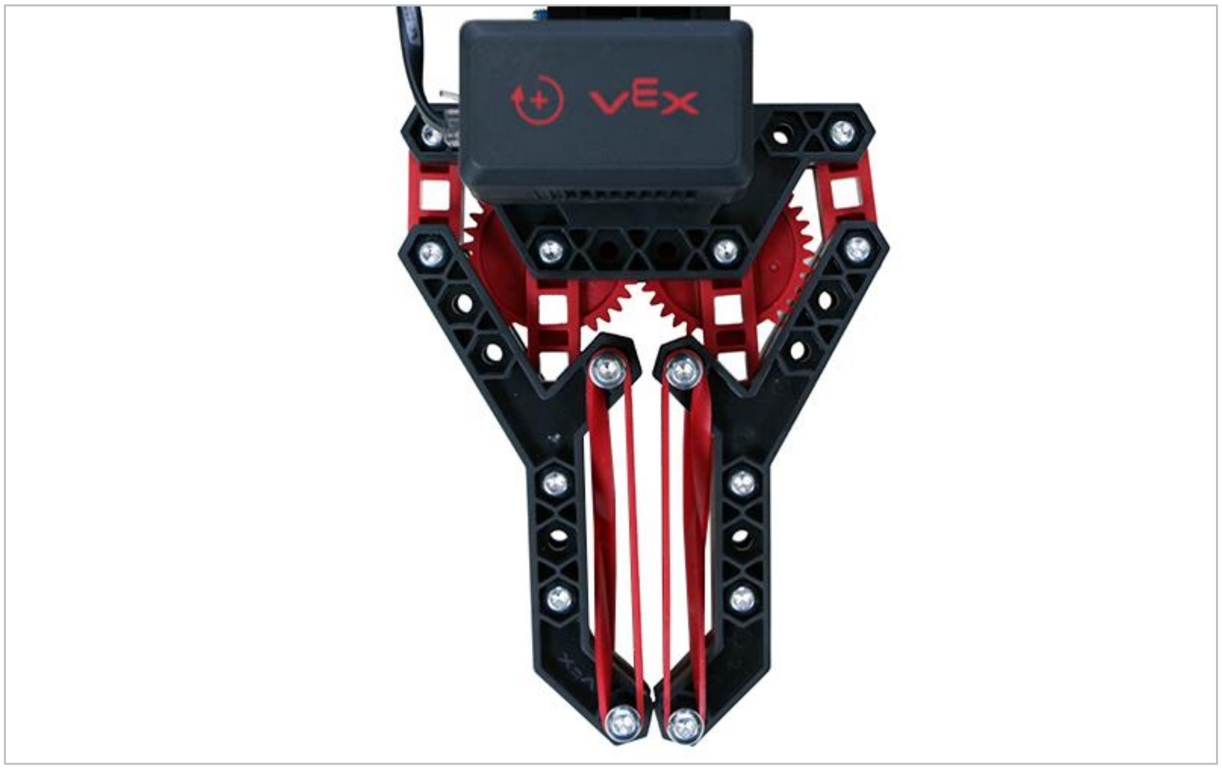


- Check that the ClawControl project has downloaded to the brain in the slot you chose.

- Run the project on the Clawbot by making sure the project is selected and then press the Run button. Congratulations on creating your first project for moving the Claw!

4. Try This: Close the V5 Claw

Now that you have programmed the claw to open, you will now program it to close.

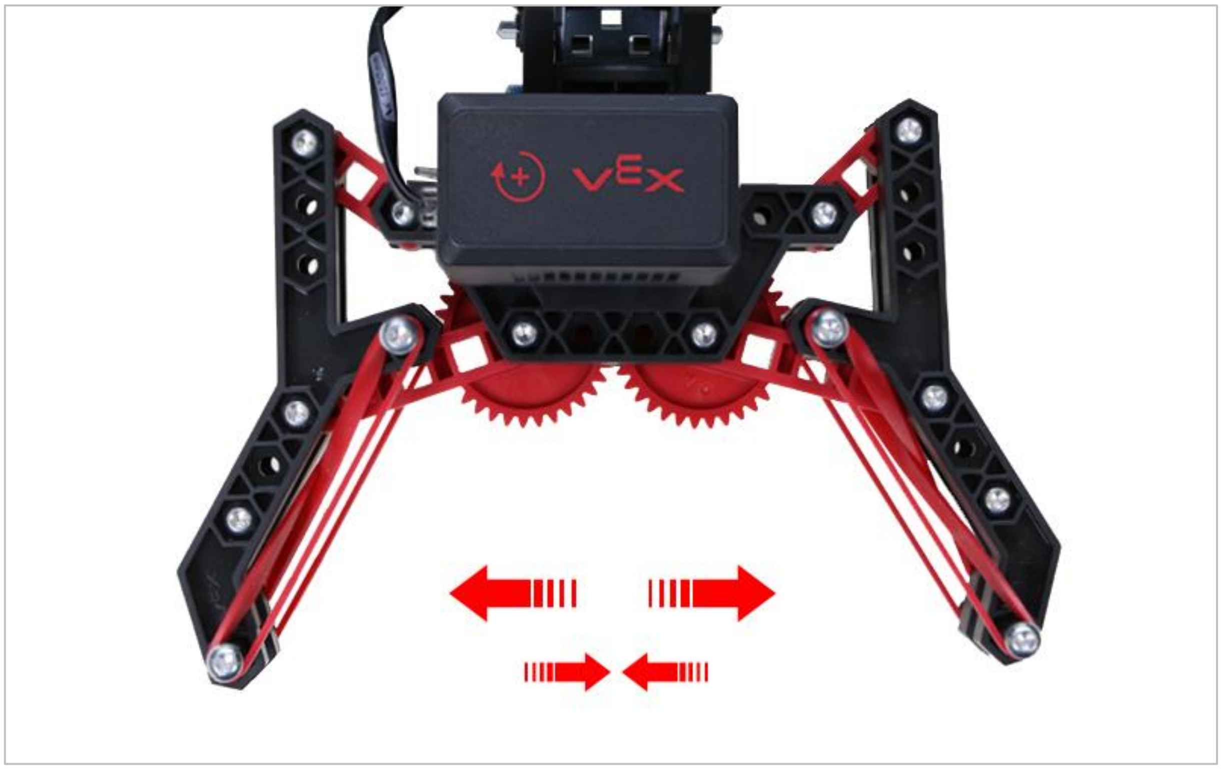


V5 Claw closed

Now that you can open the claw, you will want to close it as well.

- Return to your ClawControl project and add an additional ***ClawMotor.spinFor();*** *ins*truction to have the Claw Motor spin closed for 30 degrees. The claw should close half of the way because it originally spun open for 60 degrees.
- Test that your revised project has the claw open for 60 degrees and then close for 30 degrees by downloading and running your ClawControl project.

5. Try This: Sequencing Multiple Movements



V5 Claw fully open

Not everything that you pick up with the claw will be the same size. Try opening the claw to different positions along the range of motion.

- Spin the motor in order to:
  - Open for 70 degrees
  - Close for 20 degrees
  - Open for 10 degrees
  - Close for 30 degrees
  - Close for 25 degrees
- If the Claw Motor starts at 0 degrees, how many degrees is the Claw Motor open at the end of the project?

- Begin with the Claw Motor closed.

6. Completing the Lock Tight Challenge

Claw holding an aluminum can

The Lock Tight Challenge

- Program the Clawbot to securely close the claw on an empty 12-ounce aluminum can without crushing the sides.
- Have the Clawbot hold onto the can while it raises and lowers its arm for 45 degrees.
- The Clawbot should then release the can and back away from it.
- Start the challenge with an opened claw and an empty can within it.

SPARK
APPLY

Become a 21st century problem solver
by applying the core skills and concepts
you learned to other problems.

Warehouse Robots

Robots working in a warehouse

Meeting Consumer Needs

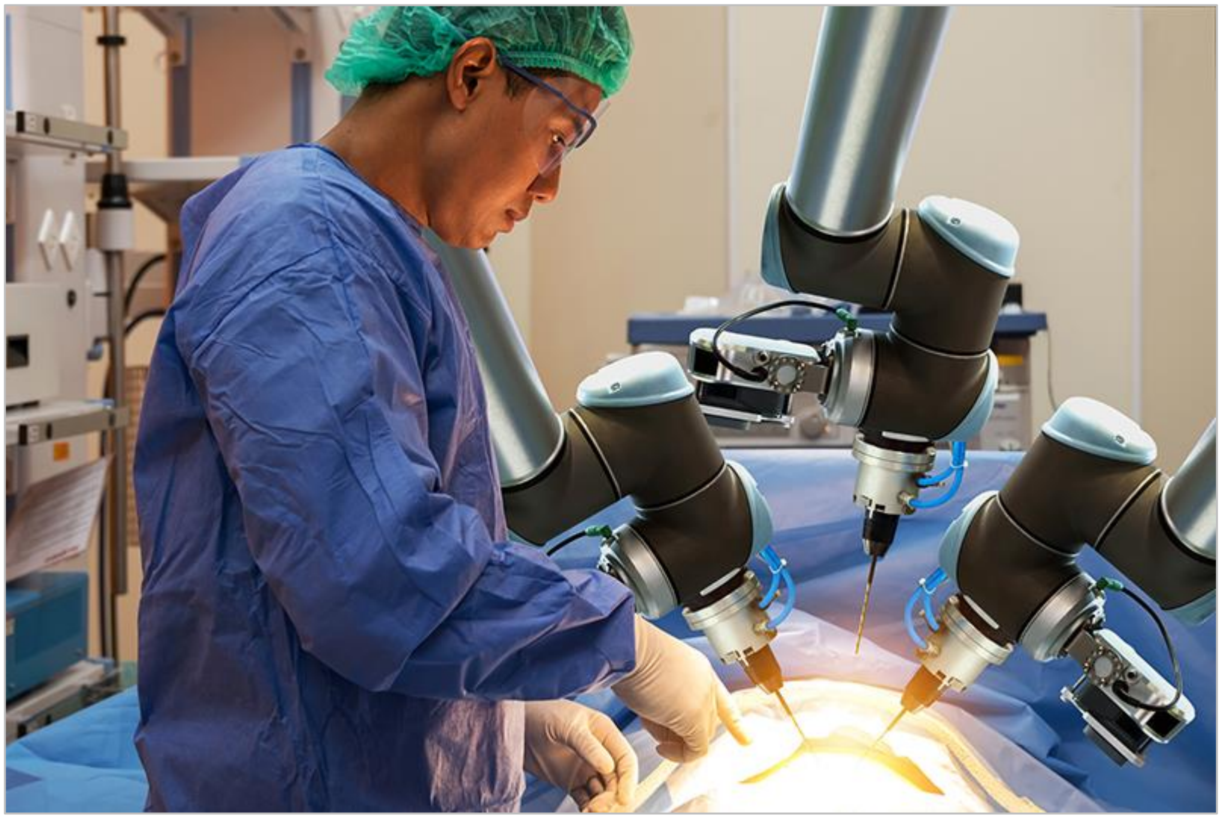
As more and more consumers shop online with guarantees of quick delivery, robots are being used to help fulfill the demand. The quicker customers' requests are fulfilled after a purchase, the happier they are and the more profit the company makes. This makes using robots to assist with orders at the warehouse a great benefit. Some companies use robots to bring the shelf stacks to human workers to select the correct product, while others use robots to travel to identified areas to grab the needed items.
Robot developers are continuing to improve the process. One company is developing robotic arms that are capable of handling fragile objects without having to give the robot detailed information on the object's size or shape. Another company is exploring "swarm robotics," where several robots work as a team by communicating together to complete delivery tasks.

Some of the benefits of using warehouse robots instead of humans are:

- Better accuracy in selecting the correct items
- More efficient (speed)
- Reduction of utility costs like air conditioning

- Less workplace theft
- Reduction of labor cost (fewer workers needed)

Robotic Precision



Robotic Surgery

Robots Come to the Rescue

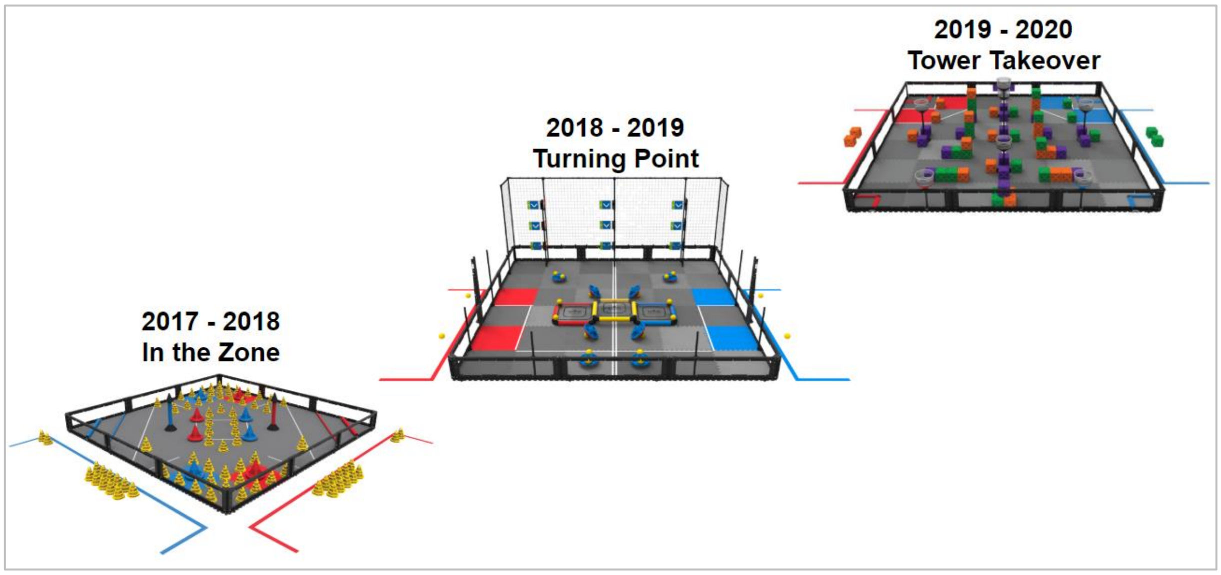
The first industrial robot was designed by George Devol in 1954. This robot was capable of carrying materials between points about twelve feet apart. A lot has changed since then. Our society keeps improving the designs of robotics to meet our ever-changing needs. One way developers are changing robotics is by making them more precise and accurate in their movements. These robots can be used in many places including warehouses, military zones, and hospitals.

As robots become more flexible and agile, they are able to handle more complex tasks in a warehouse environment. Instead of just moving packages from place to place, robots are able to sift through an assortment of packages; selecting and moving those items to designated areas without damaging them.

Robots are also being used in military zones to keep soldiers safe by detecting and clearing areas that troops will be entering. Robots are also used for tasks such as bomb disposal, keeping soldiers at a safe distance while active threats are diffused.

Humans also benefit from robotic precision in the operating room. Robotic surgery is minimally invasive when surgeons use robots to assist them in surgery. The arms of the robot are very agile and precise, allowing surgeons to operate in tight spaces in the body, without making large cuts. This lowers the risk of infections and speeds up recovery time.

Planning for Different Objects and Goals



Past VEX Robotics Competition game elements

VEX Game Elements

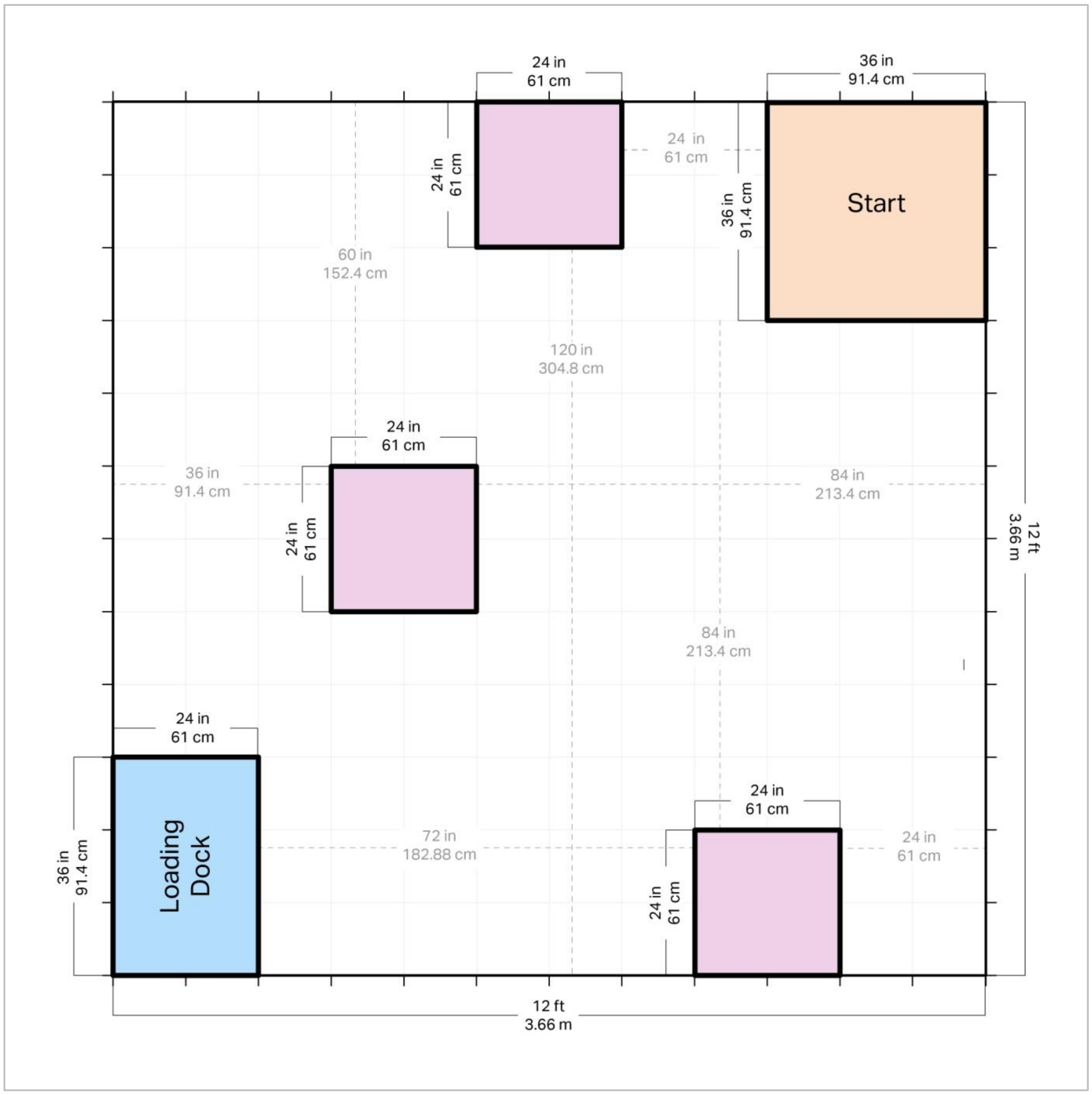
One of the more challenging aspects of the VEX Robotics Competition game is that a new game design is introduced for each competition season. This allows students to use their previous game experience to build upon as they tackle the new objects and goals of the game while giving both experienced and new teams an equal starting place. Click here to review this year's game for the VEX Robotics Competition.

Each year students will come across new game elements that their teams will have to maneuver and manipulate by moving, tossing, or flipping. These materials come in various shapes and sizes. That is why it is very important that teams design the most appropriate manipulator for their robots in order to succeed in the current game. The VEX Robotics Knowledge Base has an article about How to Decide on a Manipulator. Click on the title of the article to review the different types of manipulators and how to decide on the best one for your robot.

SPARK RETHINK

Is there a more efficient way to come to the same conclusion? Take what you've learned and try to improve it.

Prepare for the Package Dash Challenge



Package Dash Preparation Layout

Prepare for the Package Dash Challenge

In this challenge, you will program your robot to pick up packages and bring them to the Loading Dock as fast as possible! To successfully complete this challenge, you need to create a project that drives the robot to specific places (the pink squares) in the warehouse, picks up packages (aluminum cans) from the shelves (stacks of textbooks), and drops them onto the Loading Dock. The Loading Dock is a taped off area on the floor.

Ask your teacher if your team should set up the Package Dash Challenge.
When the Challenge field is ready, you should measure all of the driving distances and heights of the packages (cans) so that you can precisely plan and program.

To complete the challenge you will need:

- 12 x 12 ft. or 3.66 x 3.66 m open area
- Optional: VRC Field Perimeter and Tile kits.
- Roll of tape
- 9+ textbooks
    - Each stack of three books should be between 7 and 11 in. or 200 and 300 mm high.
- 3 aluminum cans
- A ruler or meter stick to measure distances
- Stopwatch

Design, Develop, and Iterate on your Project - VEXcode V5 Blocks

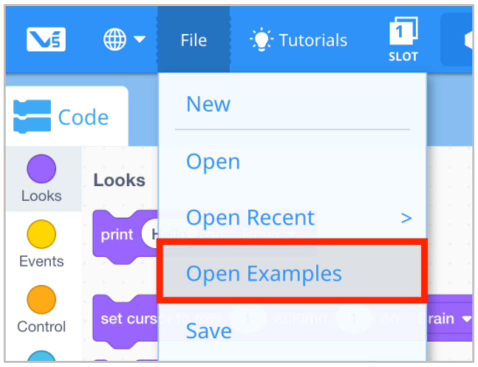
Answer the following questions in your engineering notebook as you design your project.

- What do you want the project to have the robot do? Explain with details.
- What steps will you follow to test the project? Explain with details.
- How can your robot be programmed to complete the task more efficiently? Explain how.

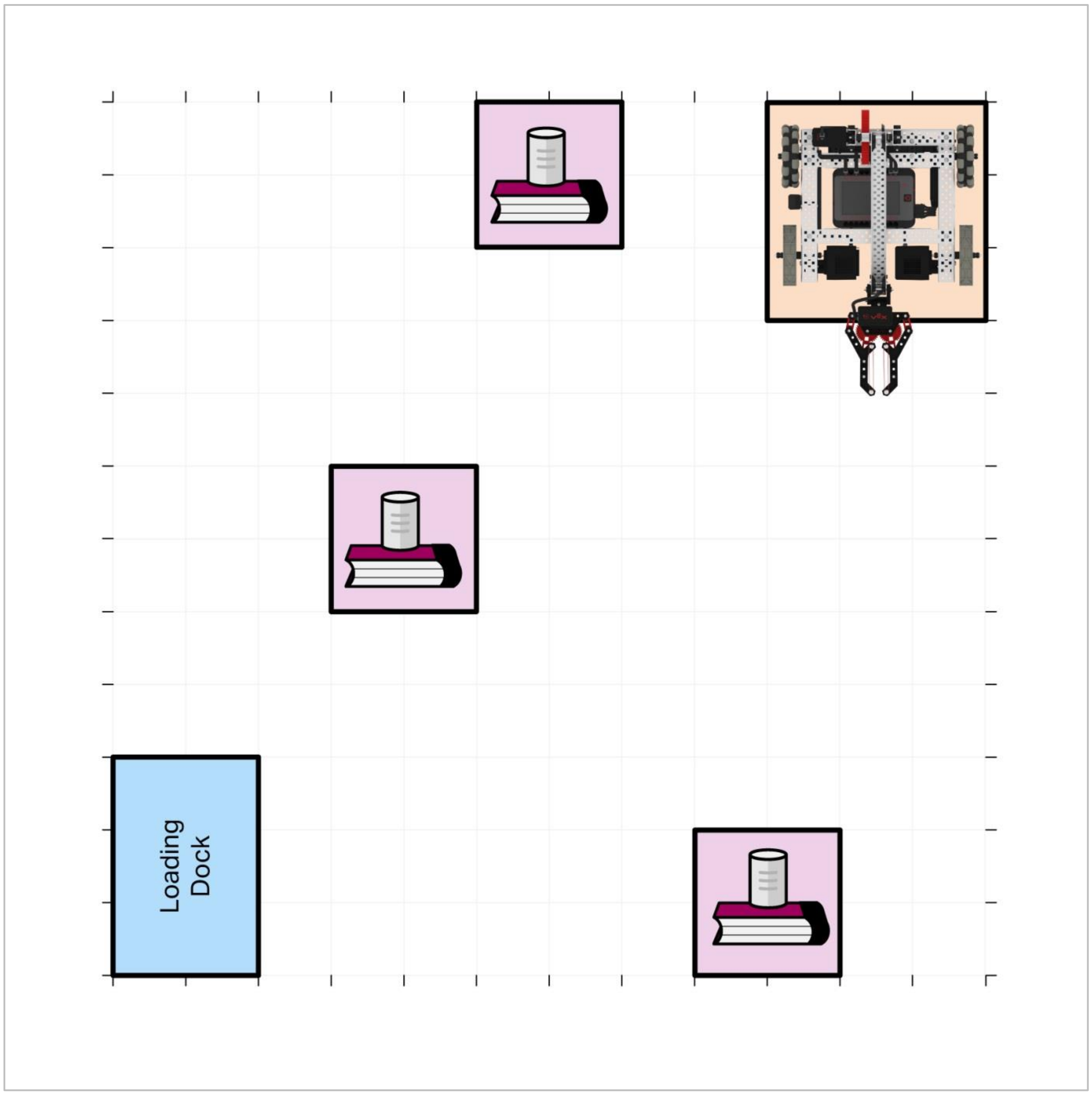
Follow the steps below as you create your project:

- Plan out the path you want to program your robot to take using drawings and pseudocode.
- Use the pseudocode you created to develop your project using blocks.
- Test your project often and iterate on it using what you learned from your testing.

If you're having trouble getting started, review the Example Projects within VEXcode V5 Blocks:



Package Dash Challenge - VEXcode V5 Blocks



The Package Dash Challenge (not to scale)

Package Dash Challenge

In this challenge, you will program your robot to pick up packages and bring them to a loading dock as fast as possible!

Challenge rules:

- The robot must begin the challenge in the Start Zone.
- The packages (aluminum cans) can only come in contact with the books, the Clawbot's claw, and the Loading Dock.
    - If a package is dropped on the warehouse ground, you must reset the field and start over again.
- The time for each run starts as soon as the robot moves.
- The time stops as soon as the last package is dropped in the Loading Dock.
- When resetting the field, everything should be returned to the exact location as it started.
- Have fun!

Design, Develop, and Iterate on your Project - VEXcode V5 Text

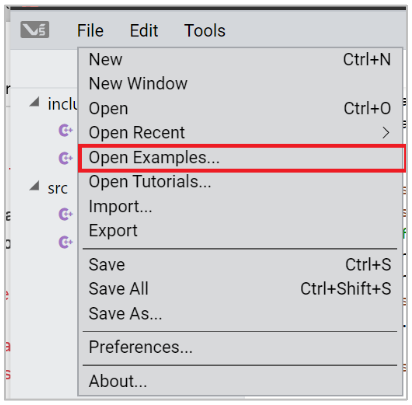
Answer the following questions in your engineering notebook as you design your project.

- What do you want the project to have the robot do? Explain with details.
- What steps will you follow to test the project? Explain with details.
- How can your robot be programmed to complete the task more efficiently? Explain how.

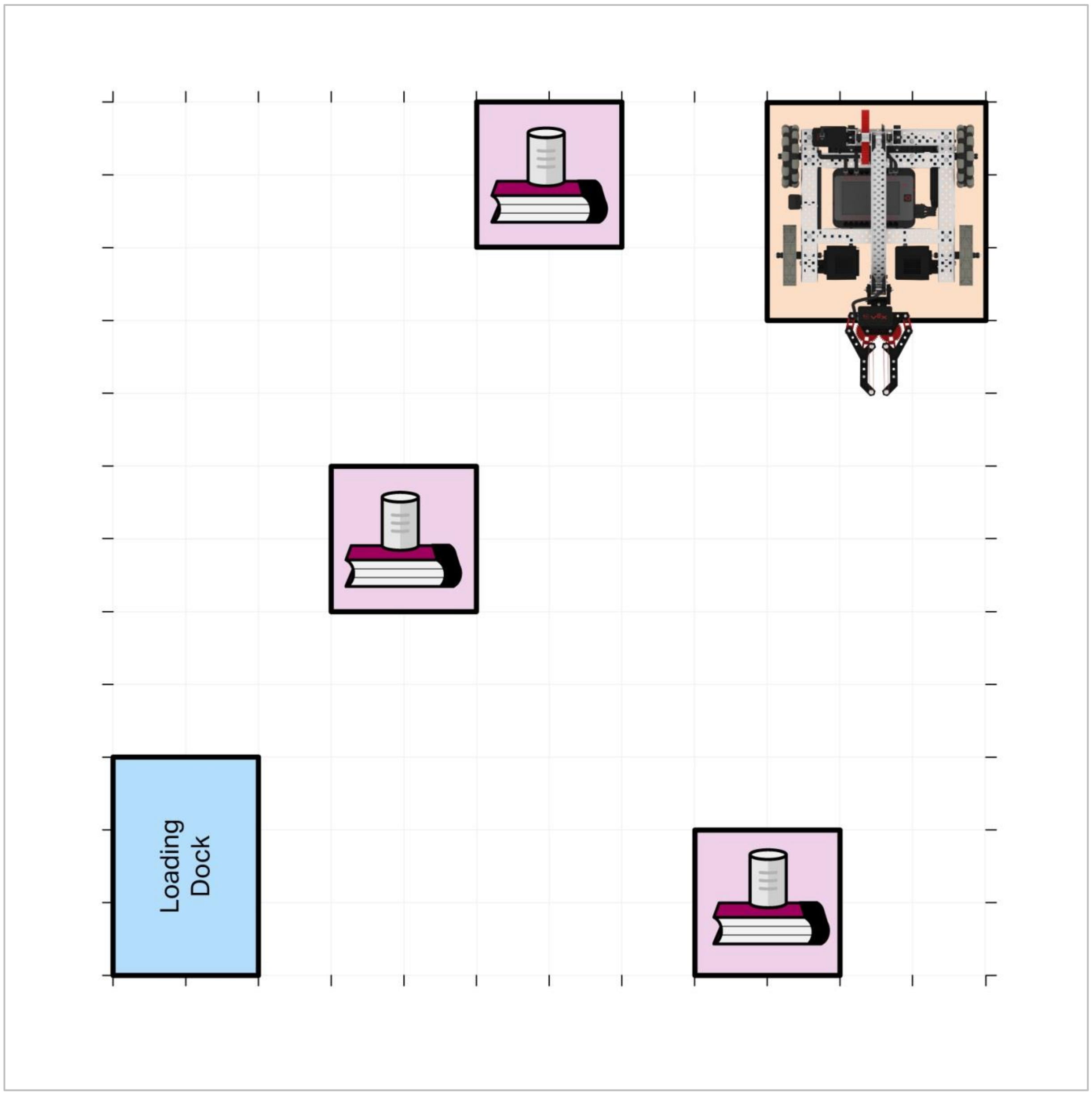
Follow the steps below as you create your project:

- Plan out the path you want to program your robot to take using drawings and pseudocode.
- Use the pseudocode you created to develop your project using text.
- Test your project often and iterate on it using what you learned from your testing.

If you're having trouble getting started, review the Example Projects within VEXcode V5 Text:



Package Dash Challenge - VEXcode V5 Text



The Package Dash Challenge (not to scale)

Package Dash Challenge

In this challenge, you will program your robot to pick up packages and bring them to a loading dock as fast as possible!

Challenge rules:

- The robot must begin the challenge in the Start Zone.
- The packages (aluminum cans) can only come in contact with the books, the Clawbot's claw, and the Loading Dock.
    - If a package is dropped on the warehouse ground, you must reset the field and start over again.
- The time for each run starts as soon as the robot moves.
- The time stops as soon as the last package is dropped in the Loading Dock.
- When resetting the field, everything should be returned to the exact location as it started.
- Have fun!

SPARK
KNOW

Understand the core concepts and how to apply them to different situations. This review process will fuel motivation to learn.

Review - VEXcode V5 Blocks

1. **Engineers frequently design specialized robot claws called _________ to enable robots to interact with a wide variety of objects that may be unsafe or unsuitable for a person.**
    - ○ end deflectors
    - ○ end effectors
    - ○ encoders
    - ○ end reflectors

2. **When the motor is set to *hold*, the motor will draw power to maintain or 'hold' its position _______.**
    - ○ for 5 minutes
    - ○ for 1 rotation
    - ○ while the next two commands are executed
    - ○ for the remainder of the project or until it is programmed to move again

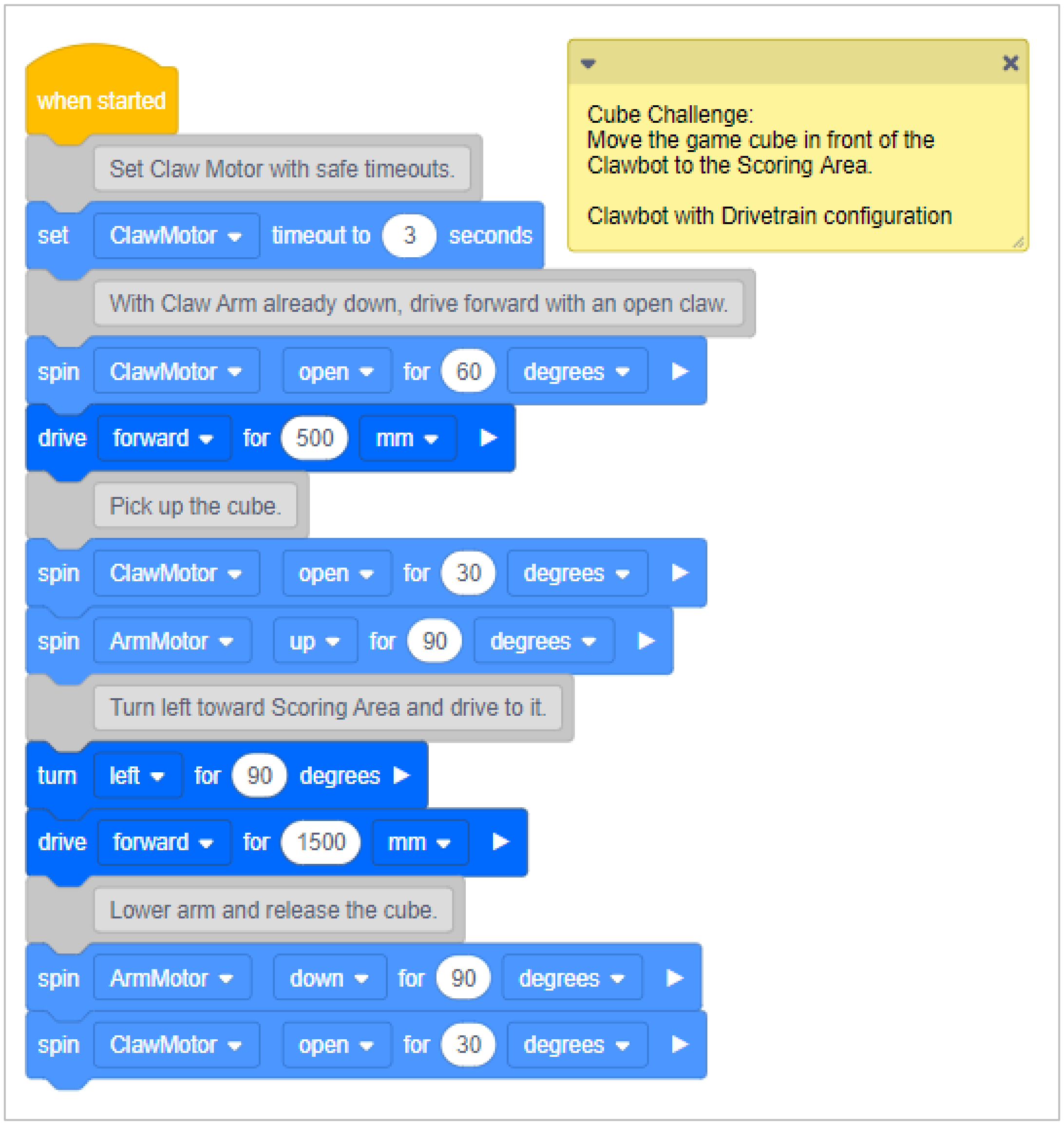
3. **A benefit of using warehouse robots instead of humans is:**
    - ○ Better accuracy in selecting the correct items.
    - ○ More efficient (speed).
    - ○ Both answers are correct.
    - ○ Neither answer is correct.

4. **True or False: The claw has a limited range of motion, so setting a timeout for its motor is good practice.**
    - ○ True
    - ○ False

5. **Which of these lines of pseudocode for my Clawbot project will make programming the easiest for me?**

- Find and pick up a cube.
- Find the cube and bring it to the scoring area.
- Drive forward, lower the arm, and open the claw to ready the robot for picking up the cube.
- Drive across the field to a cube and use the claw to pick it up and bring it to the scoring area.

6. **There is one error in this project. What is it?**



```
when started
Set Claw Motor with safe timeouts.
set ClawMotor timeout to 3 seconds
With Claw Arm already down, drive forward with an open claw.
spin ClawMotor open for 60 degrees ▶
drive forward for 500 mm ▶
Pick up the cube.
spin ClawMotor open for 30 degrees ▶
spin ArmMotor up for 90 degrees ▶
Turn left toward Scoring Area and drive to it.
turn left for 90 degrees ▶
drive forward for 1500 mm ▶
Lower arm and release the cube.
spin ArmMotor down for 90 degrees ▶
spin ClawMotor open for 30 degrees ▶
```

Cube Challenge:
Move the game cube in front of the Clawbot to the Scoring Area.

Clawbot with Drivetrain configuration

- The arm is already down when the projects starts but the project moves it down further. That risks damage to the Arm Motor or the Clawbot.
- The Claw Motor opens the claw twice, once before driving to the cube and again after reaching the cube. But, it doesn't close around the cube.
- The left turn toward the Scoring Area was programmed as a right turn.

- The release of the cube should have been programmed to open the claw first and then move the arm down.

Review - VEXcode V5 Text

7. **Engineers frequently design specialized robot claws called _________ to enable robots to interact with a wide variety of objects that may be unsafe or unsuitable for a person.**
    - o end deflectors
    - o end effectors
    - o encoders
    - o end reflectors

8. **When the motor is set to *hold*, the motor will draw power to maintain or 'hold' its position _______.**
    - o for 5 minutes
    - o for 1 rotation
    - o while the next two commands are executed
    - o for the remainder of the project or until it is programmed to move again

9. **A benefit of using warehouse robots instead of humans is:**
    - o Better accuracy in selecting the correct items.
    - o More efficient (speed).
    - o Both answers are correct.
    - o Neither answer is correct.

10. **True or False: The claw has a limited range of motion, so setting a timeout for its motor is good practice.**
    - o True
    - o False

11. **Which of these lines of pseudocode for my Clawbot project will make programming the easiest for me?**

- Find and pick up a cube.
- Find the cube and bring it to the scoring area.
- Drive forward, lower the arm, and open the claw to ready the robot for picking up the cube.
- Drive across the field to a cube and use the claw to pick it up and bring it to the scoring area.

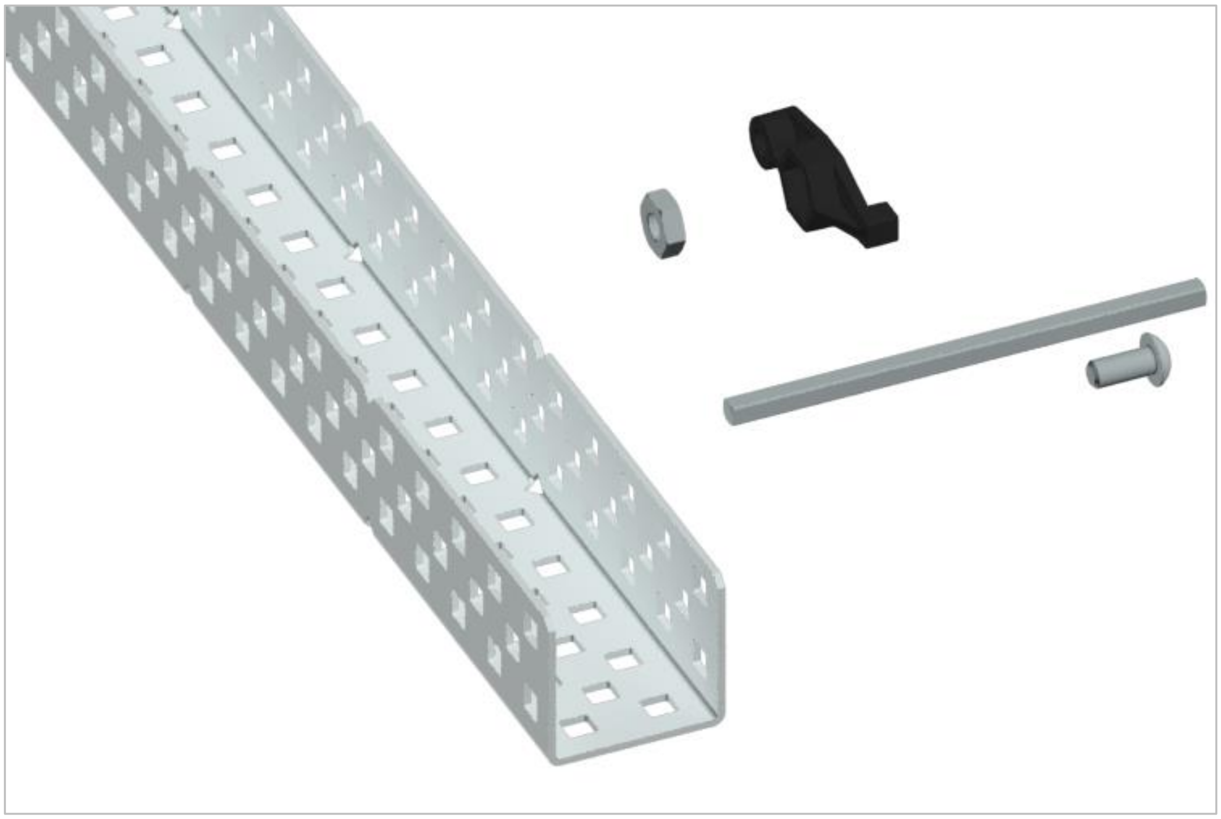
12. The two values in the ArmMotor.spinFor(); instruction are:

- Degrees and Turns
- Turns and Inches
- Degrees and Inches
- None of the answers

APPENDIX

Additional information, resources, and materials.

Using the 1 Post Hex Nut Retainer w/ Bearing Flat



1 Post Hex Nut Retainer w/ Bearing Flat

Using the 1 Post Hex Nut Retainer w/ Bearing Flat

The 1 Post Hex Nut Retainer w/ Bearing Flat allows shafts to spin smoothly through holes in structural components. When mounted, it provides two points of contact on structural components for stability. One end of the retainer contains a post sized to securely fit in the square hole of a structural component. The center hole of the retainer is sized and slotted to securely fit a hex nut, allowing a 8-32 screw to easily be tightened without the need for a wrench or pliers. The hole on the end of the Retainer is intended for shafts or screws to pass through.

To make use of the retainer:

- Align it on a VEX structural component such that the end hole is in the desired location, and the center and end sections are also backed by the structural component.

- Insert the square post extruding from the retainer into the structural component to help keep it in place.
- Insert a hex nut into the center section of the retainer so that it is flush with the rest of the component.
- Align any additional structural components to the back of the main structural component, if applicable.
- Use an 8-32 screw of appropriate length to secure the structural component(s) to the retainer through the center hole and hex nut.

Using the 4 Post Hex Nut Retainer

4 Post Hex Nut Retainer

Using the 4 Post Hex Nut Retainer

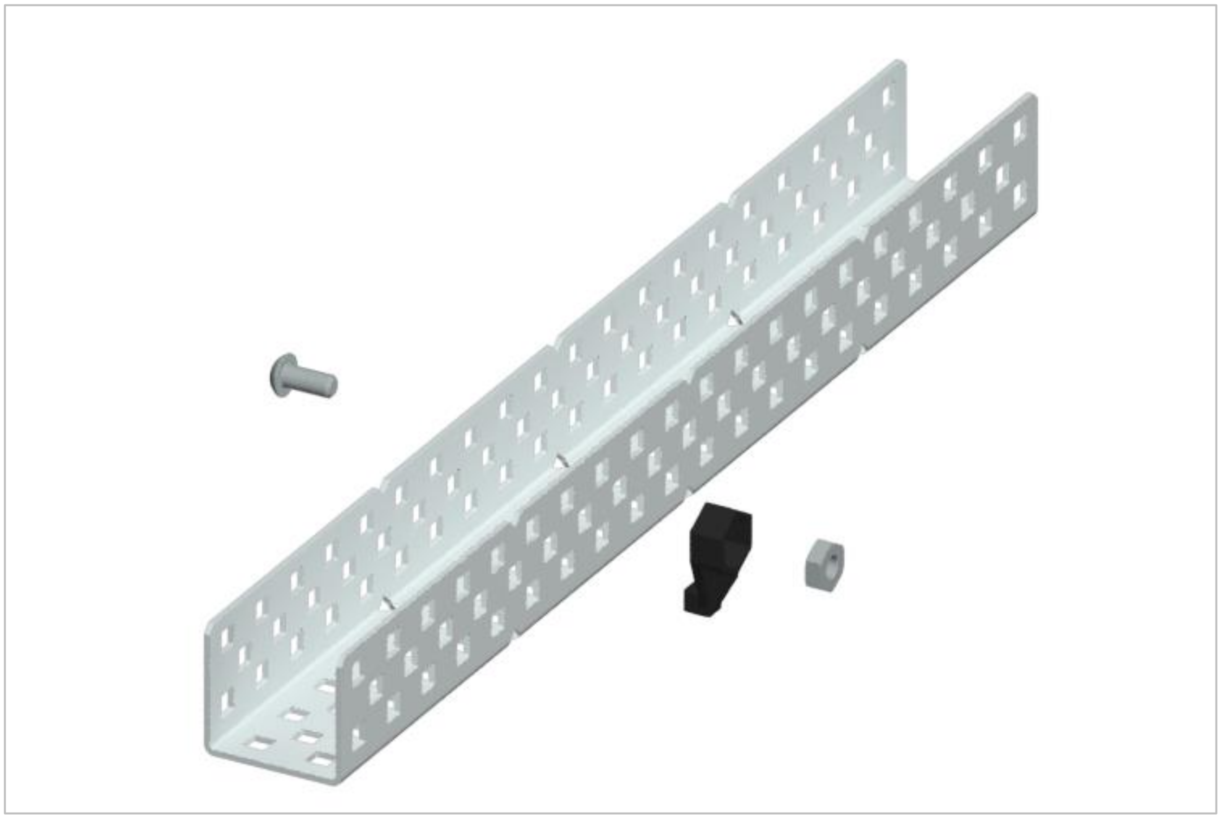
The 4 Post Hex Nut Retainer provides five points of contact for creating a strong connection between two structural components using one screw and nut. Each corner of the retainer contains a post sized to securely fit in a square hole within a structural component. The center of the retainer is sized and slotted to securely fit a hex nut, allowing a 8-32 screw to easily be tightened without the need for a wrench or pliers.

To make use of the retainer:

- Align it on a VEX structural component such that the center hole is in the desired location, and each corner is also backed by the structural component.
- Insert the square posts extruding from the retainer into the structural component to help keep it in place.
- Insert a hex nut into the center section of the retainer so that it is flush with the rest of the component.

- Align any additional structural components to the back of the main structural component, if applicable.
- Use an 8-32 screw of appropriate length to secure the structural component(s) to the retainer through the center hole and hex nut.

Using the 1 Post Hex Nut Retainer


1 Post Hex Nut Retainer

Using the 1 Post Hex Nut Retainer

The 1 Post Hex Nut Retainer provides two points of contact for connecting a structural component to another piece using one screw and nut. One end of the retainer contains a post sized to securely fit in the square hole of a structural component. The other end of the retainer is sized and slotted to securely fit a hex nut, allowing a 8-32 screw to easily be tightened without the need for a wrench or pliers.

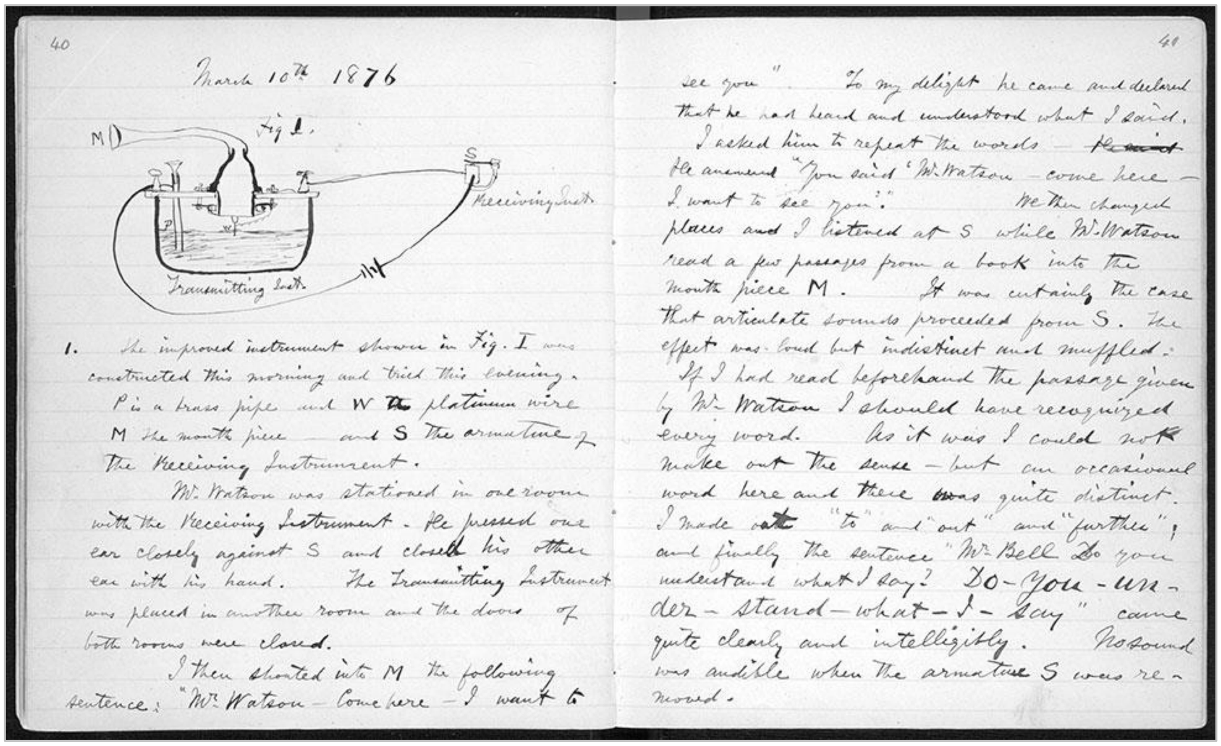
To make use of the retainer:

- Align it on a VEX structural component such that both ends are backed by the structural component and positioned to secure the second piece.
- Insert the square post extruding from the retainer into the structural component to help keep it in place.
- If the retainer is being used to secure two structural components, insert a hex nut into the other end of the retainer so that it is flush with the rest of the component. If used to secure

a different type of component, such as a standoff, it may be appropriate to insert the screw through this side.

- Align any additional components to the back of the main structural component, if applicable.
- If the retainer is being used to connect two structural components, use an 8-32 screw of appropriate length to secure the structural components through the hole and hex nut. If used to connect a different type of component, such as a standoff, secure it directly or with a hex nut.

Engineering Notebooks


M
Fig 1.
S
Receiving Inst.
P
W
Transmitting Inst.

40

March 10th 1876

1. The improved instrument shown in Fig. I was constructed this morning and tried this evening.
P is a brass pipe and W ~~the~~ the platinum wire M the mouth piece and S the armature of The Receiving Instrument.

Mr. Watson was stationed in one room with the Receiving Instrument. He pressed one ear closely against S and closed his other ear with his hand. The Transmitting Instrument was placed in another room and the doors of both rooms were closed.

I then shouted into M the following sentence: "Mr. Watson — Come here — I want to see you". To my delight he came and declared that he had heard and understood what I said.

41

I asked him to repeat the words — ~~He said~~ He answered "You said 'Mr. Watson — come here — I want to see you.'" We then changed places and I listened at S while Mr. Watson read a few passages from a book into the mouth piece M. It was certainly the case that articulate sounds proceeded from S. The effect was loud but indistinct and muffled.

If I had read beforehand the passage given by Mr. Watson I should have recognized every word. As it was I could not make out the sense — but an occasional word here and there ~~was~~ was quite distinct. I made ~~out~~ "to" and "out" and "further"; and finally the sentence "Mr. Bell Do you understand what I say? Do — you — un-der — stand — what — I — say" came quite clearly and intelligibly. No sound was audible when the armature S was removed.

Alexander Graham Bell's notebook entry from a successful experiment with his first telephone

An Engineering Notebook Documents your Work

Not only do you use an engineering notebook to organize and document your work, it is also a place to reflect on activities and projects. When working in a team, each team member will maintain their own journal to help with collaboration.

Your engineering notebook should have the following:

- An entry for each day or session that you worked on the solution
- Entries that are chronological, with each entry dated
- Clear, neat, and concise writing and organization
- Labels so that a reader understands all of your notes and how they fit into your iterative design process

An entry might include:

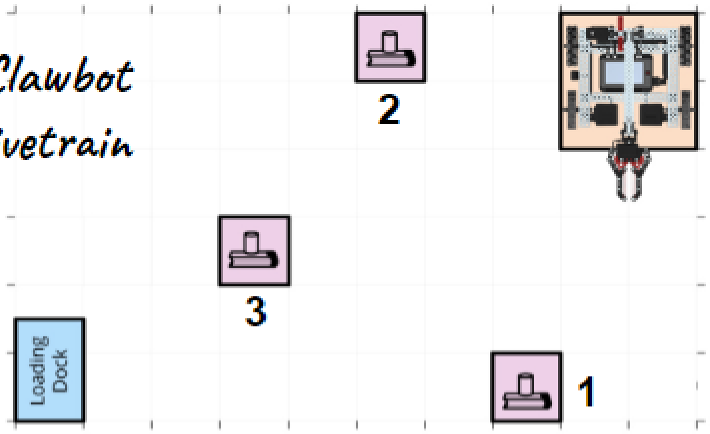
- Brainstorming ideas
- Sketches or pictures of prototypes

- Pseudocode and flowcharts for planning
- Any worked calculations or algorithms used
- Answers to guiding questions
- Notes about observations and/or conducted tests
- Notes about and reflections on your different iterations

Some of the Informational Popups within this STEM Lab

Example Pseudocode for the Package Dash Challenge

Config = Clawbot with a drivetrain



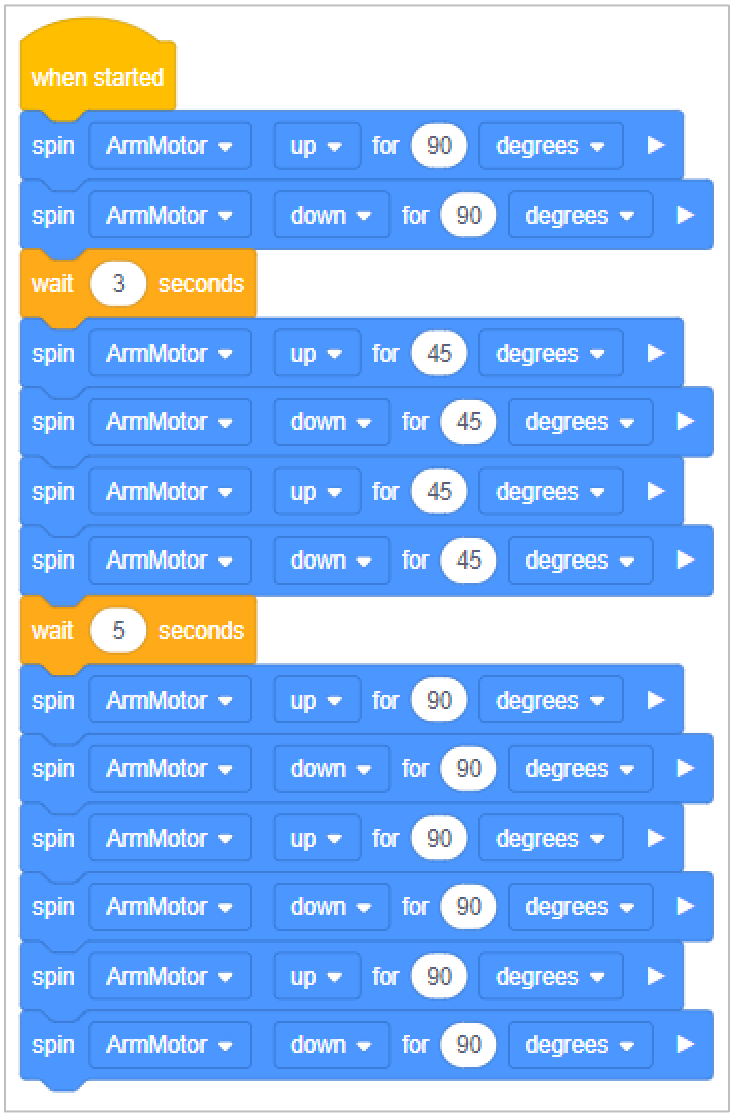
```
Safely run motor + pick up Package 1
Drive to dock + drop off Package 1
Pick up Package 2
Drive to dock + drop off Package 2
Pick up Package 3
Drive to doc + drop off Package 3
```

Robot Behaviors - VEXcode V5 Blocks

Robot Behaviors - VEXcode V5 Text

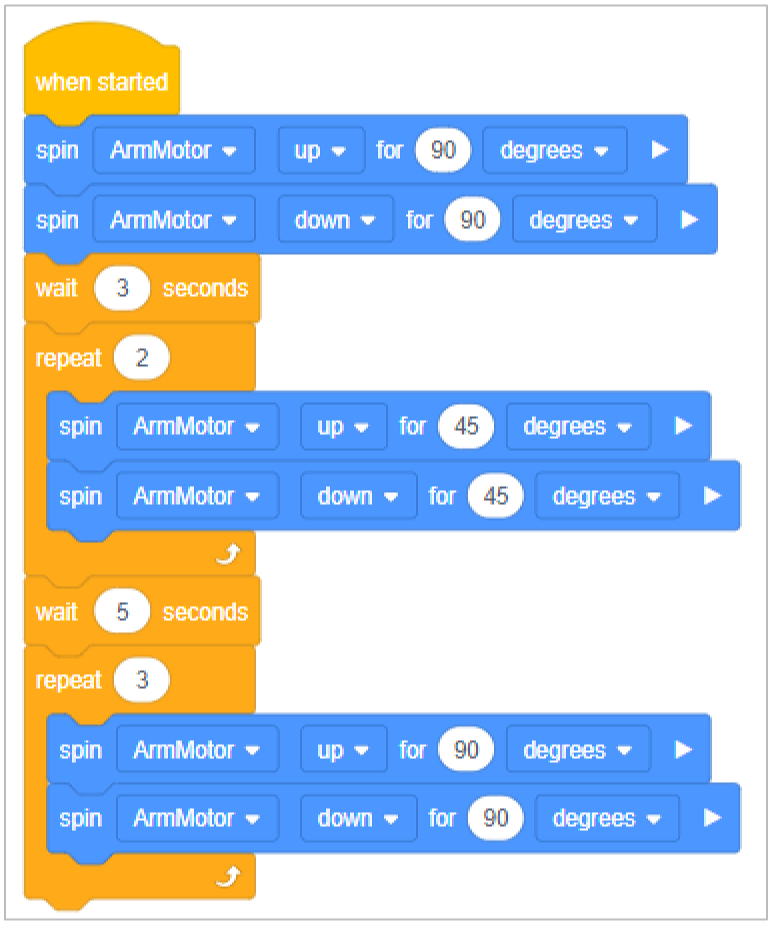
Flight Traffic Controller Challenge Solution - VEXcode V5 Blocks

Because loops have not yet been introduced, the following is a perfectly acceptable solution:



```
when started
spin ArmMotor up for 90 degrees ▶
spin ArmMotor down for 90 degrees ▶
wait 3 seconds
spin ArmMotor up for 45 degrees ▶
spin ArmMotor down for 45 degrees ▶
spin ArmMotor up for 45 degrees ▶
spin ArmMotor down for 45 degrees ▶
wait 5 seconds
spin ArmMotor up for 90 degrees ▶
spin ArmMotor down for 90 degrees ▶
spin ArmMotor up for 90 degrees ▶
spin ArmMotor down for 90 degrees ▶
spin ArmMotor up for 90 degrees ▶
spin ArmMotor down for 90 degrees ▶
```

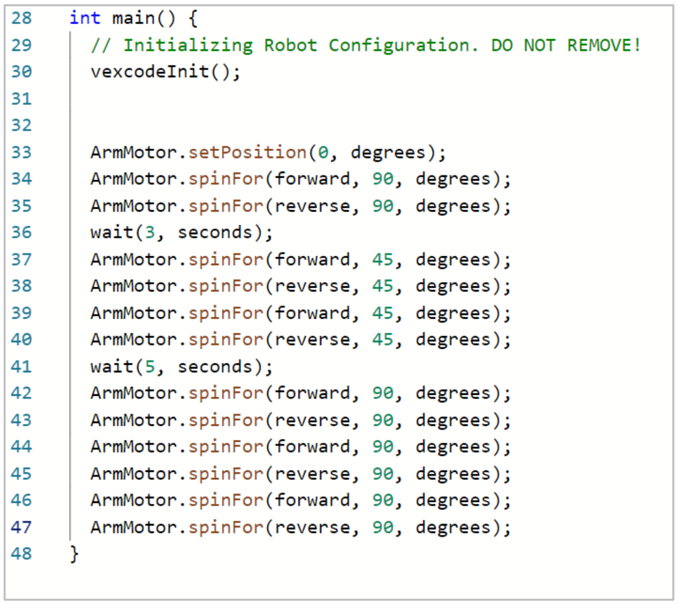
More advanced students might use loops to simplify the solution.



```
when started
spin ArmMotor up for 90 degrees
spin ArmMotor down for 90 degrees
wait 3 seconds
repeat 2
    spin ArmMotor up for 45 degrees
    spin ArmMotor down for 45 degrees
wait 5 seconds
repeat 3
    spin ArmMotor up for 90 degrees
    spin ArmMotor down for 90 degrees
```

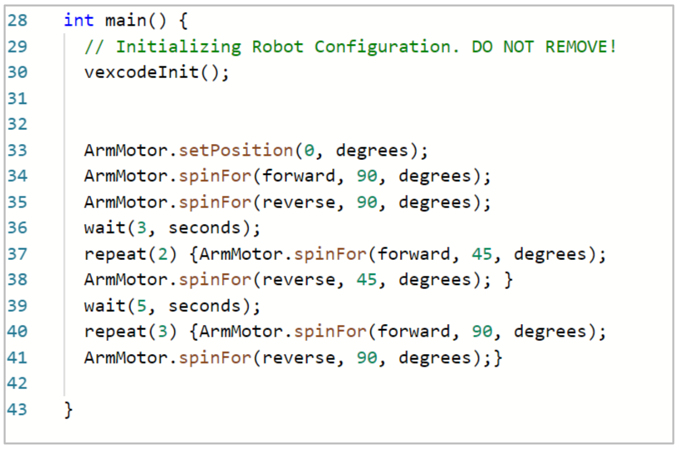
Flight Traffic Controller Challenge Solution - VEXcode V5 Text

Because loops have not yet been introduced, the following is a perfectly acceptable solution:



```
int main() {
  // Initializing Robot Configuration. DO NOT REMOVE!
  vexcodeInit();


  ArmMotor.setPosition(0, degrees);
  ArmMotor.spinFor(forward, 90, degrees);
  ArmMotor.spinFor(reverse, 90, degrees);
  wait(3, seconds);
  ArmMotor.spinFor(forward, 45, degrees);
  ArmMotor.spinFor(reverse, 45, degrees);
  ArmMotor.spinFor(forward, 45, degrees);
  ArmMotor.spinFor(reverse, 45, degrees);
  wait(5, seconds);
  ArmMotor.spinFor(forward, 90, degrees);
  ArmMotor.spinFor(reverse, 90, degrees);
  ArmMotor.spinFor(forward, 90, degrees);
  ArmMotor.spinFor(reverse, 90, degrees);
  ArmMotor.spinFor(forward, 90, degrees);
  ArmMotor.spinFor(reverse, 90, degrees);
}
```

More advanced students might use loops to simplify the solution.



```
int main() {
  // Initializing Robot Configuration. DO NOT REMOVE!
  vexcodeInit();


  ArmMotor.setPosition(0, degrees);
  ArmMotor.spinFor(forward, 90, degrees);
  ArmMotor.spinFor(reverse, 90, degrees);
  wait(3, seconds);
  repeat(2) {ArmMotor.spinFor(forward, 45, degrees);
  ArmMotor.spinFor(reverse, 45, degrees); }
  wait(5, seconds);
  repeat(3) {ArmMotor.spinFor(forward, 90, degrees);
  ArmMotor.spinFor(reverse, 90, degrees);}

}
```