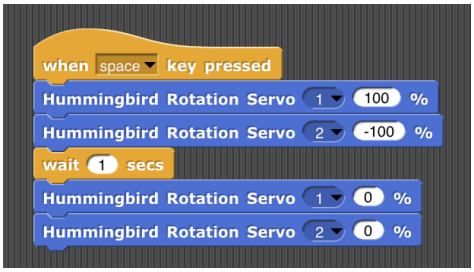
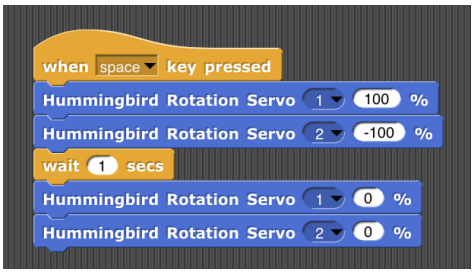
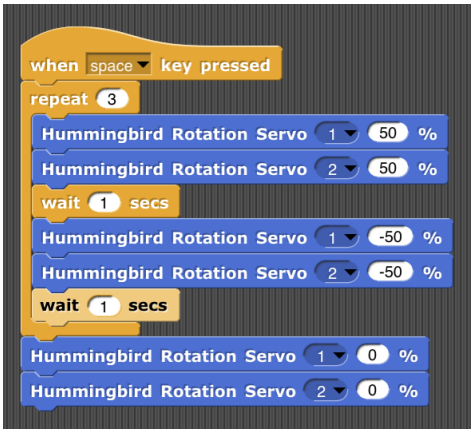
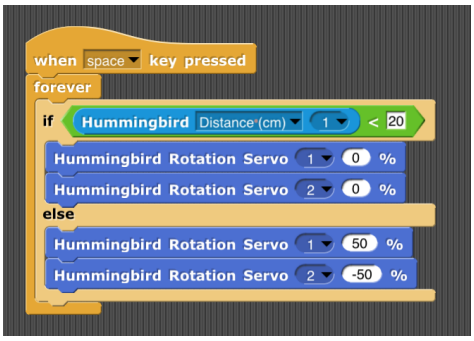
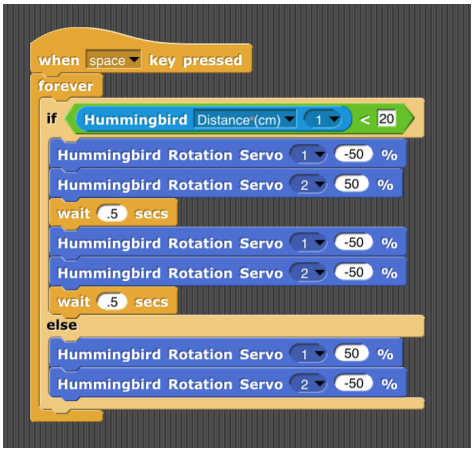
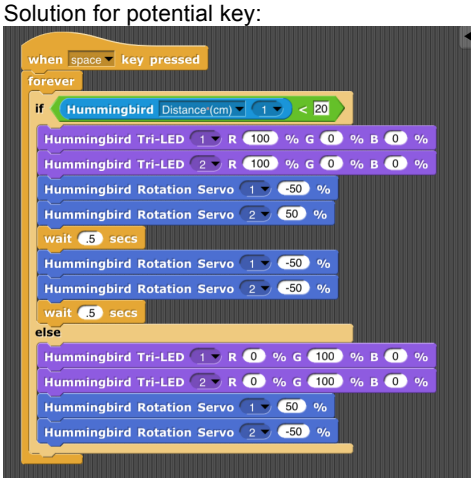
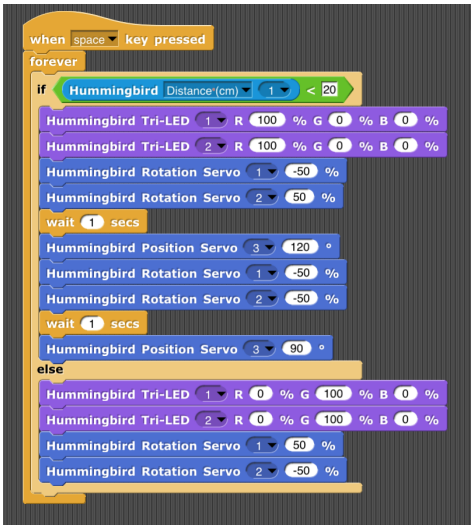
Rover Modules: Snap!
Draft – January 2019

Rover Module 1: Driving and Avoiding Obstacles

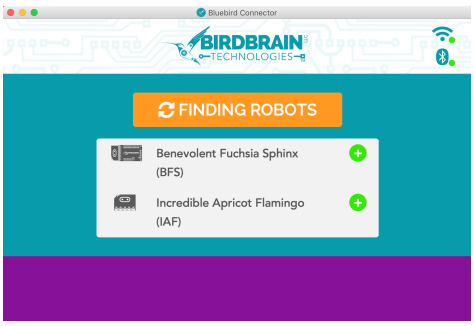
In this module, you will program a mobile robot to move and avoid obstacles. You will need a rover for this module, so go ahead and build one with this tutorial. We kept our rover pretty simple, but feel free to make yours a parade float, a bananamobile, or whatever you can dream up! Notes: The rover modules assume that you already know how the basics of using LEDs, rotation servos, and sensors with your Hummingbird Bit. The rover modules work best with bluetooth. If you are using the Hummingbird Bit with the USB cord, try the flower modules instead.	
To move the rover forward, the two rotation servos must move in opposite directions. This is because the two servos point in opposite directions on the bottom of the rover.	
Code Exploration: Experiment with different wheel speeds to see how you can move the rover in different ways. How can you move backward?	

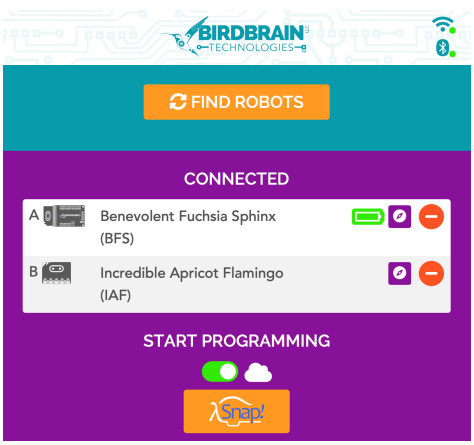
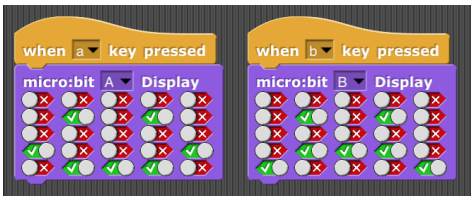
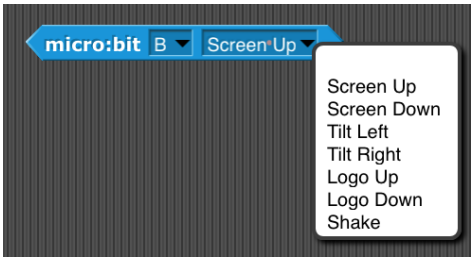
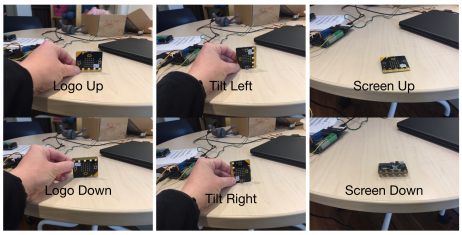
Code Exploration: When the speeds of the rotation servos are equal, the robot turns in place. How can you change how far the robot turns? Notice that this code turns the rotation servos off at the end of the program. Otherwise, they will remain on when the script ends, and the rover will keep spinning.	
Challenge: Make your rover move in a square.	
You can use the distance sensor to help your robot avoid obstacles. This program will stop the rover when it sees an obstacle less than 20 cm away. Otherwise, the rover will move forward.	
Challenge: Modify your program to make the rover back up and turn left when it sees an obstacle. Now your rover can wander around the room!	Solution: We don't want to show this, but we might want a teacher key

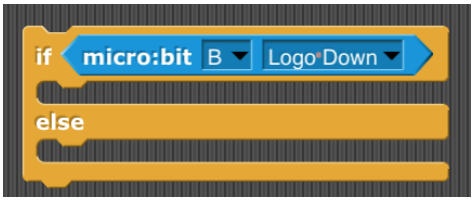
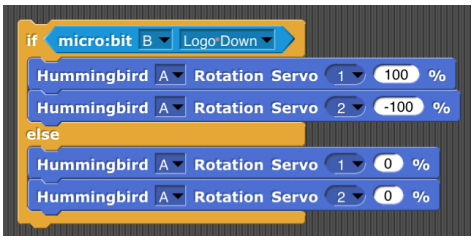
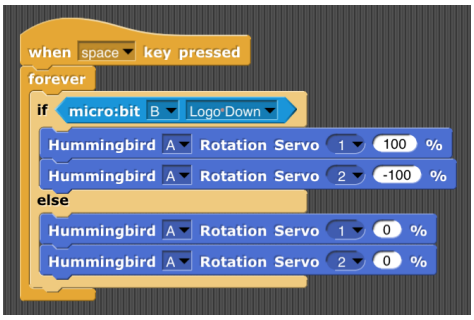
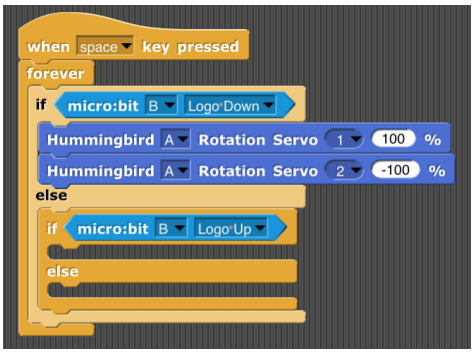
	
Challenge: Make the headlights turn red when the rover is moving away from an obstacle and green when the rover is moving forward.	Solution for potential key: 
Challenge: Make your driver steer! Modify the program to make the driver turn the steering wheel to the left as it turns left and back to the center when it is moving forward or back.	Solution for potential key or could upload solution xml file for the whole module (saved as Wheeled Rover on my account):

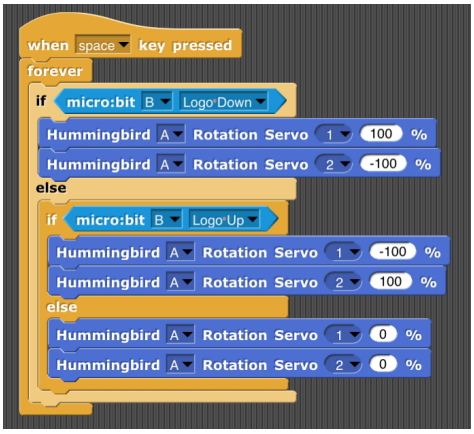
	

Rover Module 2: Remote Control with Second micro:bit

You can connect up to three Hummingbird Bits or micro:bits with the BlueBird Connector! This means that you can create robots that interact with one another. In this module you will use a second micro:bit to control the rover. **Important Note:** For this module, you will need a second micro:bit and a small battery pack to power it.	Show the rover and the second micro:bit with a battery pack.
Download this file (/downloads/installers/BitFirmware.hex) onto the second micro:bit. When you open the BlueBird Connector, it should show that both the Bit and the micro:bit are available. Each device will be flashing its initials on the screen.	

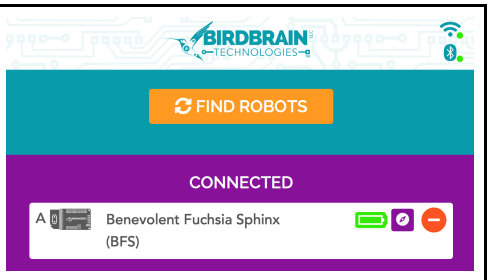
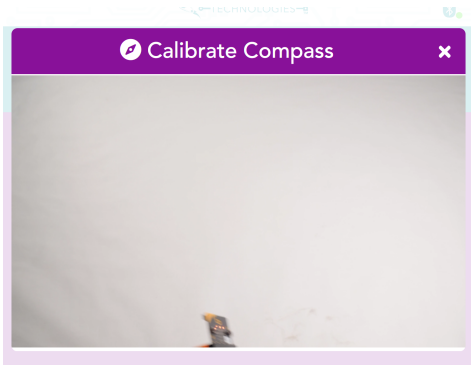
Connect the Hummingbird Bit first and the micro:bit second. In the Connected section of BlueBird Connector, notice that the Hummingbird Bit is device A and the micro:bit is device B. **This tutorial assumes that the the rover is device A and the second micro:bit is device B.**Click the Snap! button to open Snap. If you have any other Snap! windows open, close those. You need to open a new Snap! window from the BlueBird Connector in order to use multiple devices.	
Now every block for Hummingbird or micro:bit has an extra drop-down menu in which you can select **A**, **B**, or **C**. You select the letter of the device for that block. For example, the script on the left makes a smiley face on device A when you press ‘a’, and the script on on the right makes a frowny face on device B when you press ‘b.’	
You will use the orientation of device B (the micro:bit) to control the motors of device A (the Hummingbird Bit). To do this, you will use the micro:bit orientation block on the **Sensing** menu. This block is a Boolean block in which you can select an orientation. When the micro:bit is in that orientation, the block is true. Otherwise, the block is false.	
Challenge: Try placing the micro:bit in different orientations as you click on the micro:bit orientation block.	

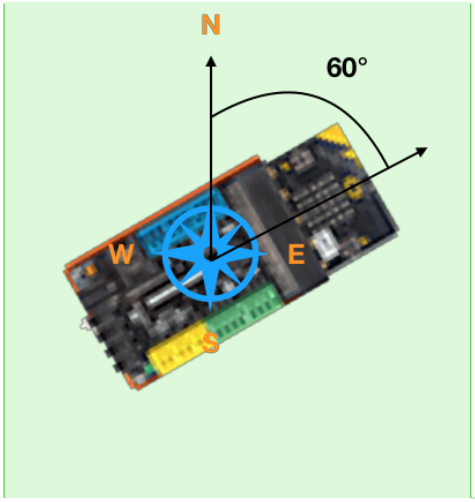
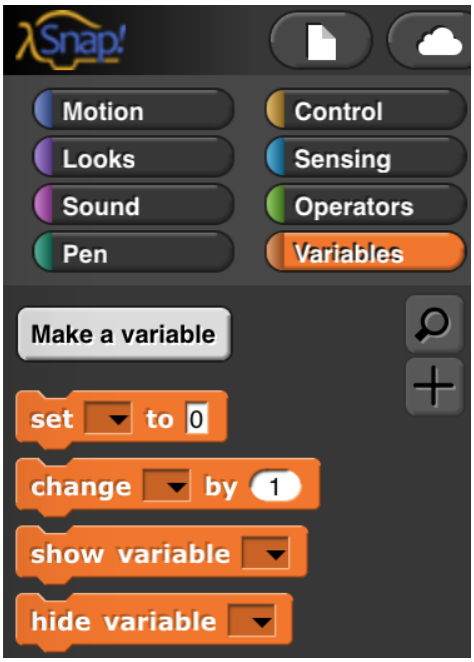
Select **Logo Down** in the micro:bit orientation block and place the block inside an **if else** block.	
When the micro:bit is in the **Logo Down** position, move the rover forward. Otherwise, stop the rover.	
Repeat this decision over and over so that every time device B is logo down, the rover moves forward!	
Now the rover can move forward or stop, but we also want the rover to move backward and turn. Drag the blocks that stop the rover out of the **if else** block. Inside the **else** section, add another **if else**; this is called nesting one **if else** inside another. The nested **if else** will check whether device B is in the **Logo Up** orientation.	

If device B is not in the **Logo Down** orientation, check the orientation again. If device B is in the **Logo Up** orientation, move the rover backward. If device B is neither **Logo Down** nor **Logo Up**, stop the rover.	
Challenge: Modify the program so that when you tilt device B to the right or left, the rover turns in that direction.	
Challenge: Make the driver turn the steering wheel to the appropriate position when moving forward, back, right, or left.	
Challenge: In addition to the micro:bit orientation block, the **Sensing** menu also contains the **micro:bit Button** block. Use this block write a script to turn the headlights on when button A is pressed is pressed on device B. Turn the lights off when button B is pressed on device B.	

Rover Module 3: Direction with the Compass

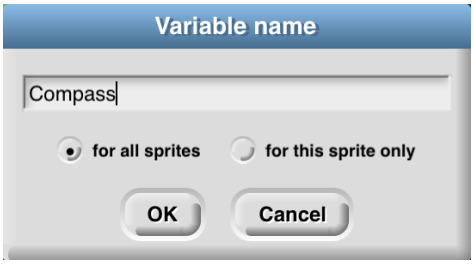
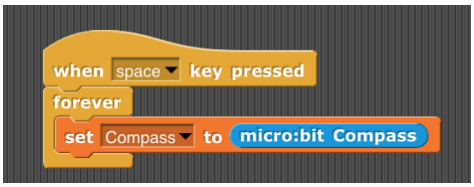
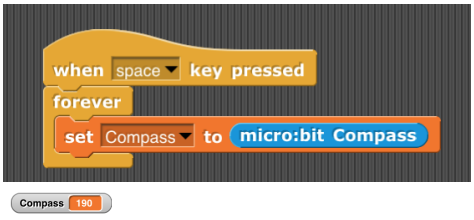
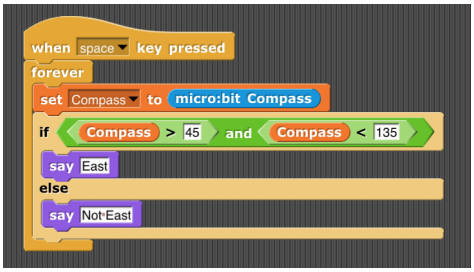
In this module, you will learn to use the compass that is on the micro:bit. You can use the compass to make the robot move in a particular direction, no matter which direction you point it in The compass tells you the direction of your micro:bit relative to magnetic north.	

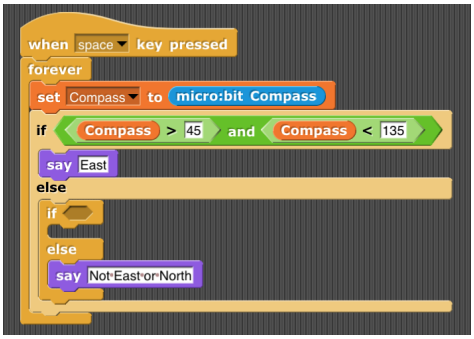
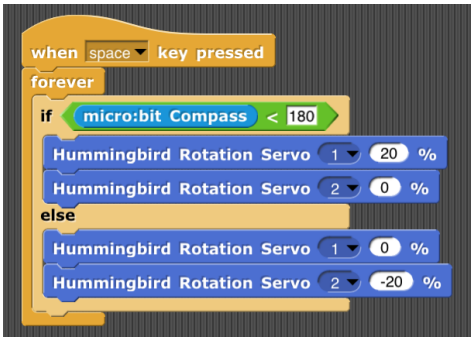
Before you use the compass, you need to calibrate it in the BlueBird Connector. To do this, connect to your Hummingbird Bit and then click on the purple compass button next to the name of your device.	
Follow along with the video to move your Bit around in different directions to calibrate it. You should see a green check when you have successfully calibrated. If you see a red X, try again.	
Once you have successfully calibrated, you can use the **micro:bit Compass** button on the **Sensing** menu to read the value of the compass. This value will be between 0° and 359°. 0° corresponds to the direction of magnetic north. The angle increases as the robot turns clockwise, so 90* is east, 180° is south, and 270° is west.	

	
To use the compass, the micro:bit must be positioned so that the LED display is parallel to the ground. Otherwise, the compass will not provide useful measurements. Position the Bit as shown in your rover.	
As you work with the compass, it is helpful to see it's value as the rover is moving. You can do this using a variable. To create a variable, click **Make a variable** on the **Variables** menu.	

Bambi Brewer 12/18/2018 7:20 PM

Comment [1]: This is not a good picture, it is just an example of what we might have here.

Give your variable a logical name, like **Compass**.	
The **Variables** menu also contains blocks that you can use to set and change the value of your variable. Use the **set to** block to continually set the value of your variable equal to the value of the **micro:bit Compass** block.	
Code Exploration: The value of your variable is shown in the Stage Area in Snap!. Watch this value change as you slowly turn the rover.	
You can use the compass to make the rover tell you which direction the rover is heading in. For example, if the value of the compass is between 45 and 135, use the **say** block on the **Looks** meu to make the sprite on the screen say “East.” You can use the **and** block on **Operators** menu to combine comparisons. The **and** block will be true only when both Boolean blocks inside it are true.	

Challenge: Add a second **if else** block to your script as shown. Modify the nested **if else** so that the sprite says “South” when the value of the compass is between 135 and 225.	
Challenge: Modify your script so that the sprite says “West” when the value of the compass is between 225 and 315. When the direction is not “East,” “South,” or “West,” the sprite should say “North.”	
Code Exploration: You can also use the compass to make the rover move in a particular direction. For example, this code makes the robot turn clockwise when the compass is less than 180° and counterclockwise otherwise. Try placing the rover in different positions before running this script. Which direction does the rover drive in?	
Challenge: Modify your program so that the robot drives south as long as there is no obstacle in the way. When the robot encounters an obstacle, it should back up and turn left.	