sphero®

BOLT+™

EDUCATOR GUIDE

For information about this title or to order other books and/or electronic media, contact the publisher:
Sphero, Inc
education@sphero.com
www.sphero.com
Library of Congress Control Number: 978-1-7331447-9-7

sphero®

BOLT+™

POWER PACK

EDUCATOR GUIDE

BOLT+ POWER PACK EDUCATOR GUIDE

TABLE OF CONTENTS

Welcome **6**

BOLT+ Says Hello, Class 6

Getting Started **8**

Connecting to Your BOLT+ 8

The Block Canvas 10

BOLT+ Challenge Cards **12**

Introduction 12

BOLT+ Challenges **18**

Challenge #1: Get Rollin' 18

Challenge #2: Ready, Set, Drive 20

Challenge #3: Sound Timing 22

Challenge #4: Two Truths & a Lie 24

Challenge #5: Matrix Pictures 26

Challenge #6: Fade to Black 28

Challenge #7: A Ring of Lights 30

Challenge #8: Four Corners 32

Challenge #9: Metamorphosis 34

Challenge #10: Program a Circle 36

Challenge #11: Random Roller 38

Challenge #12: Data Ball 40
Challenge #13: Drive Countdown 42
Challenge #14: Knock-Knock Machine 44
Challenge #15: BOLT+ Feels Sick 46
Challenge #16: Light at the End of the Tunnel 48
Challenge #17: A BOLT+ Timer 50
Challenge #18: A Polygon Algorithm 52
Challenge #19: Flying BOLT+ 54
Challenge #20: Red Light Green Light 56

Next Steps **58**
Lessons & Support 58

FAQs **59**
What's in My BOLT+ Power Pack? 59
What's in BOLT+? 60
Programming Options 61
BOLT+ Care & Storage 62

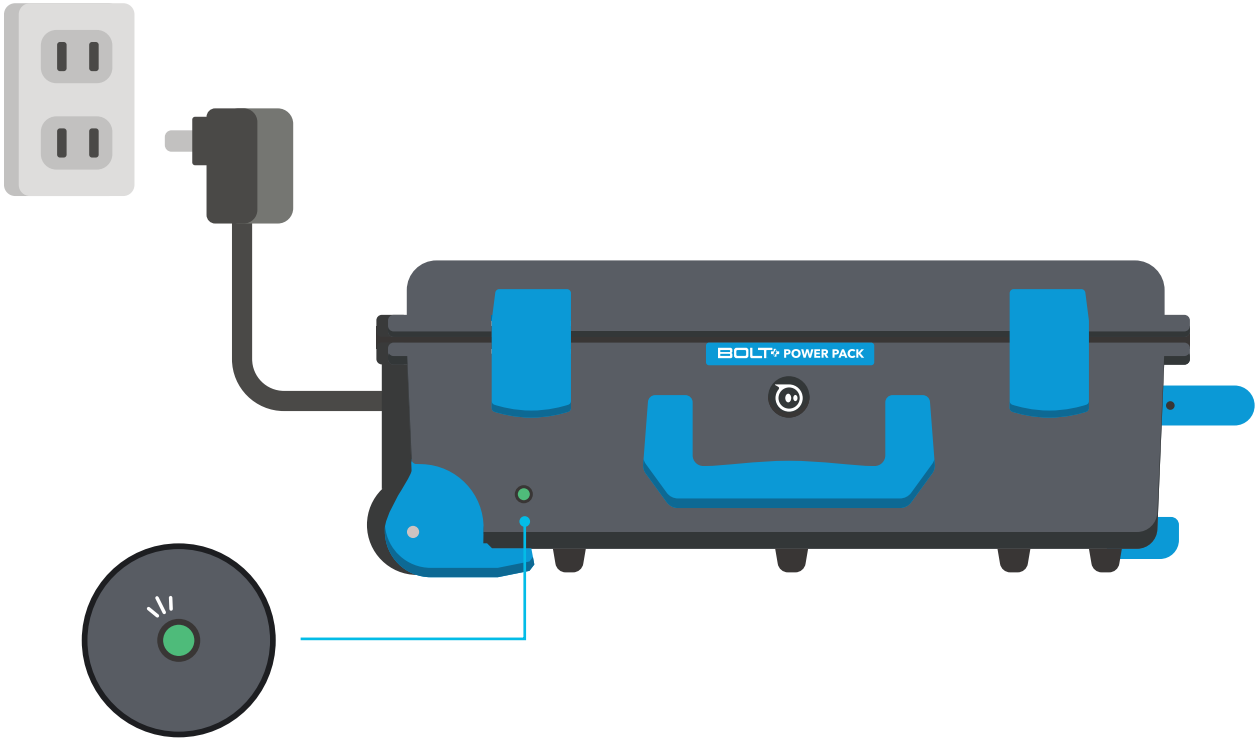
Reference **64**
Glossary 64
Block Library 66

WELCOME

BOLT+ SAYS HELLO, CLASS

You might be wondering: What are all these spheres and why are they in my classroom?

Welcome to the BOLT+ Power Pack–a precocious gaggle of BOLT+ robots ready to bring to life everything from computer science to the engineering design process.
But first, if you haven't already, **plug in** your Power Pack so your BOLT+s will be ready for the fun stuff.



Charging

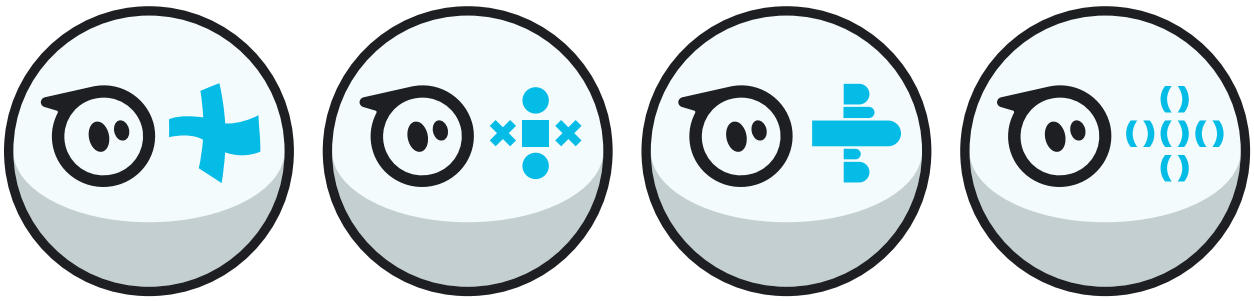
Great! As your BOLT+s prepare for fun, let's do the same. You might be curious: What, exactly, will your students be learning?

Technology may be all around us, but that doesn't make teaching students about technology any easier. STEM, STEAM, computational thinking, critical thinking, engineering design principles—many terms describe what students should learn to best prepare for the future. Keeping track of everything

can get confusing quickly. Our role at Sphero is not to prioritize what your students need to learn–that is best left up to you because you know your classroom best—our goal is to offer you many ways to engage and delight your students. Of course, they'll learn a bunch of stuff along the way, and hopefully, you will too!

Now that we've gotten that out of the way: What makes having 15 BOLT+s in your room such a delight, you ask?

BOLT+ isn't just fun, it is **incredibly flexible** for meeting learning needs. Want to integrate technology into your **math** block? Great! Want to blend **art** and technology to unleash student creativity? Fantastic. Dip your toes into the world of **block coding**? Wonderful. Flex your coding muscles and program in **JavaScript**? Also wonderful!



ART *MATH* *BLOCK CODING* *JAVASCRIPT*

This all sounds great, but you're probably wondering: How?

Well, you're in the right place. The BOLT+ Power Pack comes with **20 challenge cards** and, in this guide, you'll find **corresponding support** with background knowledge, possible student responses, and extensions. These will help launch BOLT+ in your classroom and set your students up for success with more challenging lessons in the future.

CONNECTING TO YOUR BOLT+

To use BOLT+ you first need to connect it to your **programming device** (computer, tablet, or smartphone).

Go to sphero.cc/edu-d to find the right app for your device.

Open your app and connect to a BOLT+ robot.

Take BOLT+ off its charger. Shake it to turn it on. Connect to the robot in the app.

Aim and drive your robot

Select Drive.

Then Aim your robot to tell it which way is forward.

Create a new program.

Select the new program button.

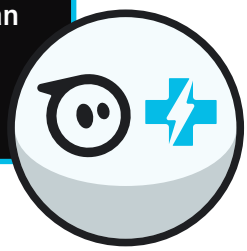
1. Select **New Program**.

2. **Name** your Program.

3. Choose a **program type**.

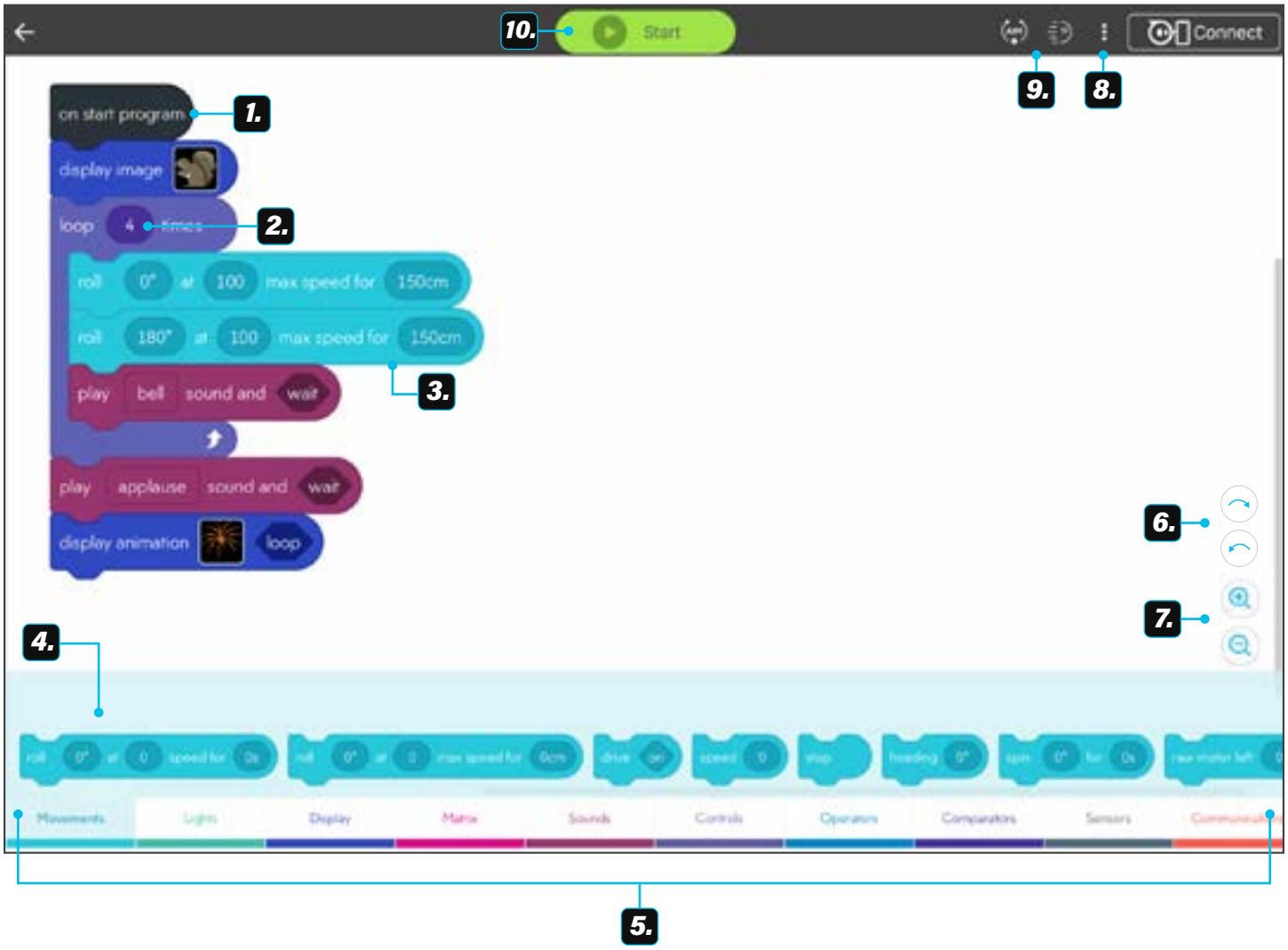
4. Choose **BOLT+**.

TIP: **Encourage students to name their programs. It can become very tricky to find your old programs without clear names, especially if you use shared devices.**



GETTING STARTED

THE BLOCK CANVAS



1. Connect blocks to **on start program** to build a program. This is the only block that will always be on the canvas.
2. Select a block **input**, or editable field, to modify what the block does. You can also drop other blocks into inputs if they are the same shape.
3. Long press or right click on a block for more options:
    - **duplicate:** Make a copy of one or more blocks.
    - **copy from canvas:** Make a copy so you can paste it in another program.
    - **delete:** Delete the block; you can also drag blocks to the bottom of the canvas to delete them.
    - **add comment:** Add a note to help others understand your code.
    - **block help:** Access information about how the block works.

4. Drag blocks from the **block library** onto the program canvas.
5. Use **block categories** to help you find the block you are looking for. The blocks within each category will be the same color.
6. **Undo** or **redo** your last action.
7. **Zoom in** or **out** to get a better look at your program.
8. Select the three dots to access the menu:
    - **Sensor Data:** Access sensor data from up to 5 previous programs.
    - **JavaScript Code:** see your block program in JavaScript.
    - **Block Canvas Help:** Open the Sphero Programming Wiki.
    - **Clean Up Blocks:** Organize the blocks on your canvas.
    - **Reset View:** Reset the canvas to original zoom settings.
9. **Aim** or **drive** your BOLT+. If you aren't connected to a BOLT+, you'll be prompted to connect.
10. Tap or click **Start** to run (execute) your program.

BOLT+ CHALLENGE CARDS

INTRODUCTION

So you've got BOLT+ rolling around, what's next? Let's explore one way to introduce block programming to your students: with our BOLT+ Challenge Cards.

Each of the 20 challenge cards included in your BOLT+ Power Pack is a short introduction to programming BOLT+ on the Block Canvas, requiring about 15 minutes. Find a digital version of the cards at: sphero.cc/bplus-cc

Each two-sided challenge card is divided into three sections:

Students rebuild a program from an image.

Students run the program and adjust the program to solve a problem.

Students explore related prompts.

Challenges **do not require** prior experience and can be completed in **any order**. However, they are divided into two different levels: **beginner** and **intermediate**. If your students are new to BOLT+, we recommend starting with beginner and moving to intermediate after your students are comfortable programming BOLT+ with movement, light, and sound blocks.

BEGINNER

Set 1

1. Get Rollin'
2. Ready, Set, Drive!
3. Sound Timing
4. Two Truths and a Lie
5. Matrix Pictures

Set 2

6. Fade to Black
7. A Ring of Lights
8. Four Corners
9. Metamorphosis
10. Program a Circle

INTERMEDIATE

Set 3

11. Random Roller
12. Data Ball
13. Drive Countdown
14. Knock-Knock Machine
15. BOLT+ Feels Sick

Set 4

16. Light at the End of the Tunnel
17. A BOLT+ Timer
18. A Polygon Algorithm
19. Flying BOLT+
20. Red Light Green Light

Not just for students, these challenge cards are a great way for you to familiarize yourself with BOLT+, the Block Canvas, and some basic programming concepts. We recommend going through them **on your own** while **thinking about where your students might have questions** and different ways you could **extend the challenges** in your classroom.

INTRODUCTION

IMPLEMENTATION RECOMMENDATIONS

Challenge cards can be used in a variety of ways in the classroom. **Consider the following implementation models:**

Whole Group

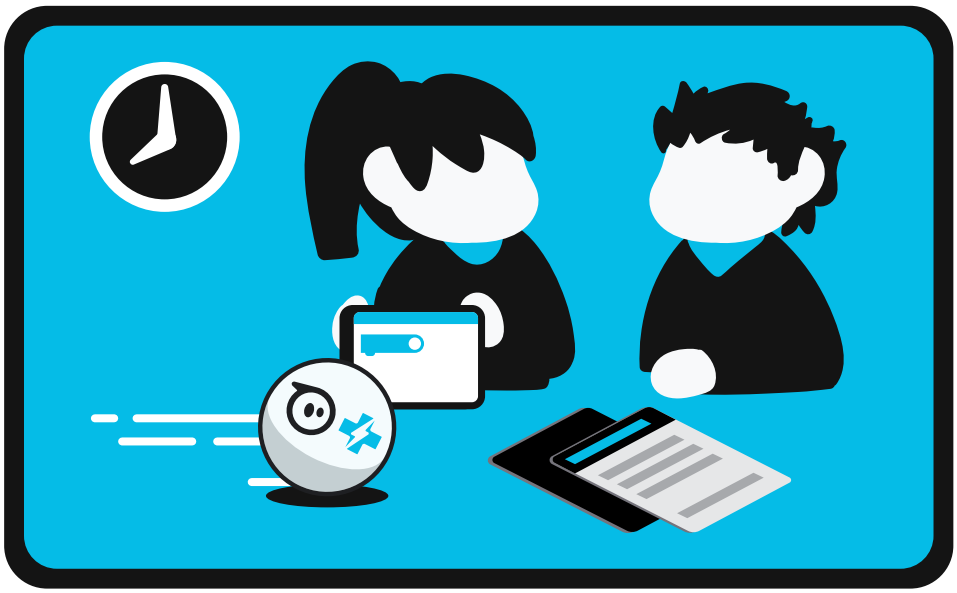
Project the challenge cards one at a time for **all students** to see. Preview the challenge together, then give students time to build the program and try the ***PROBLEM SOLVE*** and ***PLAY*** portions of the challenge. Make sure to save time at the end of each card to **review** what students learned and **share** their programs.

Small Group

Programming with BOLT+ is often more fun and productive when students work with one or more **partners**. Distribute cards to **small groups** of students and let them work through a set of cards at their own pace. If students get through all the cards before the allotted time is up, **ask students** to combine all that they've learned about block programming into one program. Bring the whole group together at the end to **share** what students learned as well as **generate ideas** for future explorations.

Stations

Set up **learning stations** with cards. Put a different card at each station and give students 15 minutes to explore each card before rotating to a new station. Since all stations are working on different cards, you can float and support as needed or remain at one station that you feel is more challenging. As with the other models, be sure to save time at the end for **reflection**.

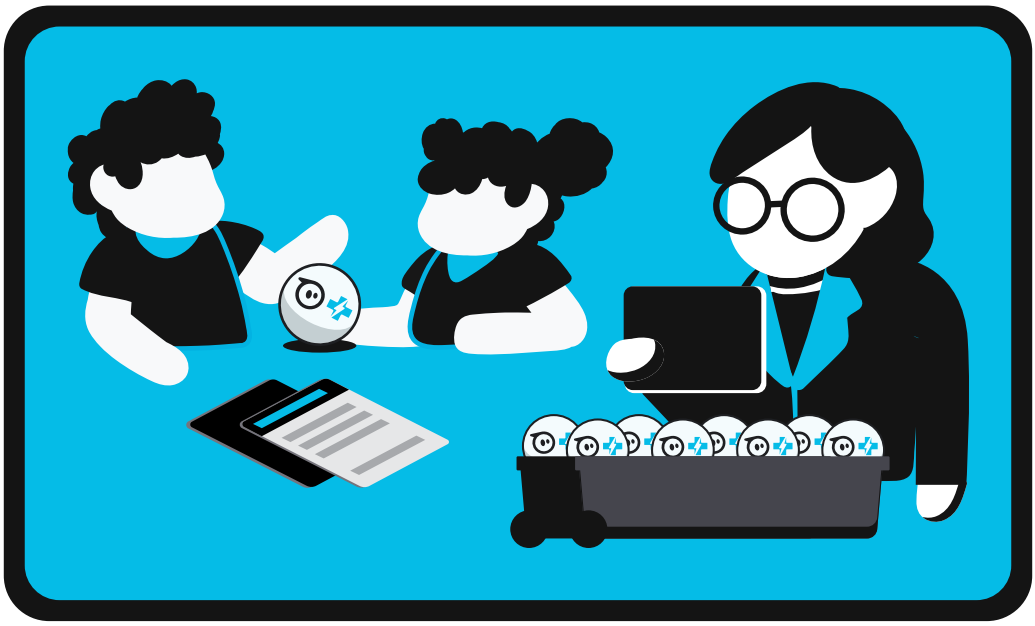


INTRODUCTION

PLANNING & FACILITATION TIPS

Before

- Take the time to complete the challenge card **on your own**. This will not only help you anticipate challenges your students may face but it's also the best way to learn how to program BOLT+!
- **Review** the information about each challenge in this guide. You'll find related **background information** on programming concepts, the Block Canvas, and BOLT+. You'll also find solutions for the ***PROBLEM SOLVE*** and ***PLAY*** sections for each challenge and ideas for giving students extra practice.
- Choose your delivery method. If possible, plan for **two students** per BOLT+ robot.
- Make sure all robots and programming devices are fully charged.
- Decide if and how you want students to record their learning.



During

- Encourage **open-ended exploration** and refrain from giving solutions.
- Celebrate effort and risk-taking. Emphasize the idea that students can make mistakes and that it's okay! The real objective here is learning to persevere in solving problems.
- If students are looking for **help**, tell them to right click or long press on a block to select ***"block help"***.
- Periodically pause the class to let students who have had breakthroughs **share** their discoveries with others.
- As students explore the ***PLAY*** section, allow them to follow their own lines of inquiry. Encouraging students to come up with their own creative questions and solutions is a great way to develop curiosity and critical thinking skills.

After

- Bring students together to **review** and **summarize** what they have learned.
- Capture questions or ideas that students want to investigate in the future.
- Use the ideas in the ***EXTRA PRACTICE*** section to plan other related lessons. Developing proficiency with a block or a programming concept requires repetition.
- If you've completed all of these challenges, you can use our **DIY BOLT+ Challenge Card template** (sphero.cc/bplus-cc-diy) to create some of your own. Alternatively, if your students are ready, they can make new challenges and put them to the test with their classmates.
- Once you finish all of these challenges, check out sphero.cc/bplus-lessons for **more lessons** to **expand** on what students have learned.

CHALLENGE #1

GET ROLLIN'

Challenge Card #1

BEGINNER

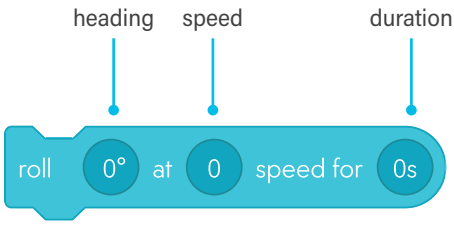
*Students explore the difference between the **roll** and **roll to distance blocks**.*

BACKGROUND

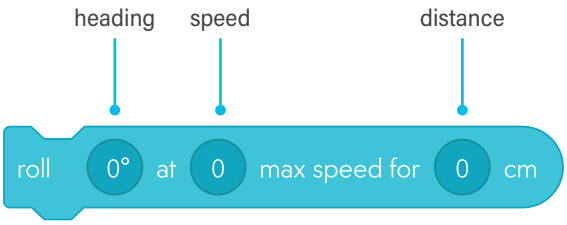
The **roll** and **roll to distance blocks** are the two main blocks students will use to program BOLT+ to move.

The **roll block** has three inputs:

- **heading:** The direction BOLT+ will roll, 0 to 359, in degrees.
- **speed**: How fast BOLT+ will roll, 0 to 255.
- **duration:** The amount of time that your BOLT+ will roll, in seconds.



roll block



roll to distance block

Each input can be changed by selecting its oval inputs. For more precise values, tap or click the numbers icon to bring up a numpad.

PROBLEM SOLVE

Students can change the inputs in both the roll and roll to distance blocks to make BOLT+ end where it started. To change the distance traveled with the roll block, they can either change the speed or duration input. To change the distance traveled with the roll to distance block, they can change the distance input.

PLAY

Set BOLT+ to max speed. How fast can it go?

BOLT+'s max speed is 255. Note that if students are using the roll to distance block, BOLT+ will accelerate up to 255, but if the distance is too short, the robot will never reach max speed.

If your students are ready for it, make a connection to computer memory. In computers, information is stored in bits and one bit represents a 0 or 1. The next unit above a bit is a byte, which stores 8 bits and can represent values from 0 to 255. Since BOLT+ robots have limited memory for storing values, it's important to be efficient, so Sphero engineers use a single byte with possible values from 0 - 255 to represent speed. The byte, 00000000, is equal to speed 0 and the byte, 11111111, is equal to speed 255.

See how BOLT+ moves on different surfaces.

BOLT+ will move differently depending on the texture and firmness of the surface. In general, you'll need to use higher speeds on softer surfaces like carpet compared to hard surfaces like a cement floor.

*Decide your favorite: the **roll block** or **roll to distance block**.*

Encourage students to share and discuss why they like one more than the other. Some students may like the roll to distance block better because BOLT+'s movement is more accurate.

EXTRA PRACTICE

- **Challenge students** to program BOLT+ to make a **shape**. You can integrate mathematical concepts by incorporating definitions of different polygons and calculating angles, perimeter, or area.
- Use maze tape and classroom objects to set up **obstacle courses**; students can then develop a program to successfully navigate BOLT+ through their course.
- **Ask students** to program BOLT+ to trace their names. If you use word sorts for spelling, you can have students create the columns and headers on the floor and program BOLT+ to roll to the correct column, say the spelling word(s) that fit that pattern, and move on to the next one.

CHALLENGE #2

READY, SET, DRIVE!

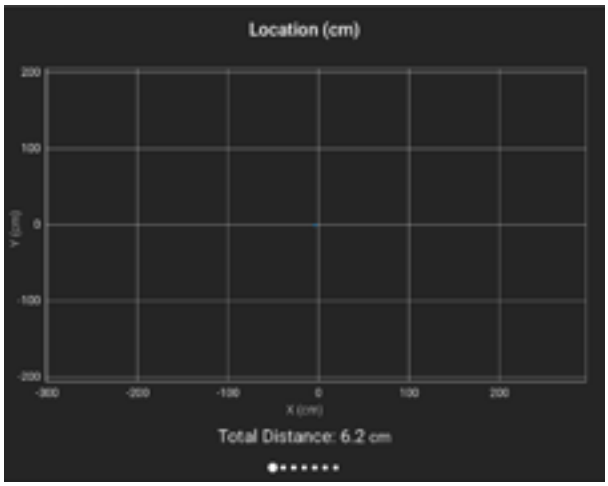
Challenge Card #2

BEGINNER

*Students learn how to turn **drive** on and off in a BOLT+ program.*

BACKGROUND

The **drive block** turns on the ability to manually drive BOLT+ using a keyboard, mouse, or touchscreen while executing the program. Use the same block to disable driver control by tapping or selecting on to toggle it to **off**.



PROBLEM SOLVE

Students have 5 seconds to drive their BOLT+ robot around their desk before the driver control is turned off. If they need more time, they can increase the number in the **delay block**.

PLAY

*Try a **roll block** when driver controls are turned on. What happens?*

Movement blocks, such as **roll** and **roll to distance blocks**, will override the **drive on block** and turn off the ability to drive BOLT while the program is running.

Add display image blocks to make your program more fun.

Students can use the **display image block** to show any image they think fits. Some students may want to show a green light when driver control is turned on and a red light when driver control is turned off.

EXTRA PRACTICE

- The ability to turn **on** and **off driver controls** takes some of the pressure off of students to program accurate movement. **Ask students** to program their robots to show text and images about something they are learning in class. Then have them drive their BOLT+ to a peer and share the information.
- **Have students** program BOLT+'s movement through an **obstacle course**. Then have them turn **on driver control** and **drive** BOLT+ back through the course!

CHALLENGE #3

SOUND TIMING

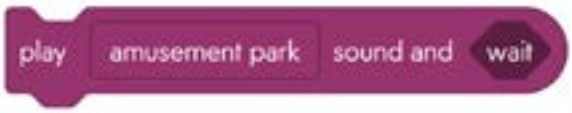
Challenge Card #3

BEGINNER

*Students learn to control how their BOLT+ program plays **sounds**.*

BACKGROUND

Programmers can control whether their blocks execute in sequence or at the same time.



When a **sound** or **speak block** is set to **wait**, students are using **synchronous programming**. Blocks will be executed in sequence and each block must finish executing before another block runs.

When you set a block to **continue**, students are using **asynchronous programming**, meaning multiple blocks can execute at the same time!

PROBLEM SOLVE

Students will notice that changing **continue** to **wait** will affect how the program executes. By setting the block to **continue** they can add things like lights and movements that occur while the sound is playing. Amusement Park is a pretty long sound and students might not want to wait that long–which makes asynchronous programming with the use of **continue** even more useful! With the block set to **wait**, they need to wait until the sound is completed before anything else can happen.

PLAY

Try other sounds besides Amusement Park.

The Sphero sound library has over 400 sounds. **Encourage students** to expand and collapse the accordion menus to find sounds they like.

Play two sounds at the same time. Can you?

A program can only play **one sound** or **one speak block** at once. If a program has two **sound blocks** and the first block is set to continue, the second sound block will be skipped. By adding a **delay block**, students can add extra time to help a second sound block execute.

EXTRA PRACTICE

- **Ask students** to make a play or skit with BOLT+. They can program BOLT+ to explain the main events in a book they just read, share what they did the previous weekend, or tell a creative story. **Encourage students** to use **movements**, **lights**, and **sounds** to make their skit come to life.

CHALLENGE #4

TWO TRUTHS & A LIE

Challenge Card #4

BEGINNER

*Students learn how to **show text** on the BOLT+ display.*

BACKGROUND

The **display text block** shows text on the BOLT+ screen. By default, the display text block shows white text on a black background, but these colors can be modified in the block.

Click or tap the input to open the **text editor**. You can choose a small or large text size. The small font size allows for 60 characters on the screen, 10 characters on six lines. The large font size allows for 35 characters, seven characters on five lines. Keep in mind that spaces count as characters.

The text editor helps ensure the text displays as you'd like it on the screen. For example, you can add text to the center of the screen by typing it on line 3.

PROBLEM SOLVE

Students type their 2 truths and 1 lie into the **display text blocks**. Remind students to keep their statements short so that they fit on the screen. Students can adjust the number in the **delay blocks** to display the text on the screen for a longer amount of time.

PLAY

Change the color of the text and the background.

Students can choose their favorite colors to **customize** their two truths and a lie program. Playing with colors will help students recognize the importance of choosing **contrasting** text and background colors for readability.

Add a block to clear the display at the end of the program.

Students will need to find the **clear display block**, also part of the basic library, to remove all text from the screen at the end of their program.

Program BOLT+ to tell the answer.

Students need to add one more **delay block** and one more **display text block** to show the solution to their puzzle. Encourage students to play around with the program and try something fancier. For example, they may want to add an image or an animation to the display before showing the answer.

EXTRA PRACTICE

- There will be times when students will want to display more text than fits on the screen at one time. **Challenge students** to break the opening sentence from *A Tale of Two Cities* into **multiple display text blocks**, "It was the best of times, it was the worst of times, it was the age of wisdom, it was the age of foolishness..."
- **Ask students** to use **speak blocks** and **display text blocks** together to narrate a skit with BOLT+.

CHALLENGE #5

MATRIX PICTURES

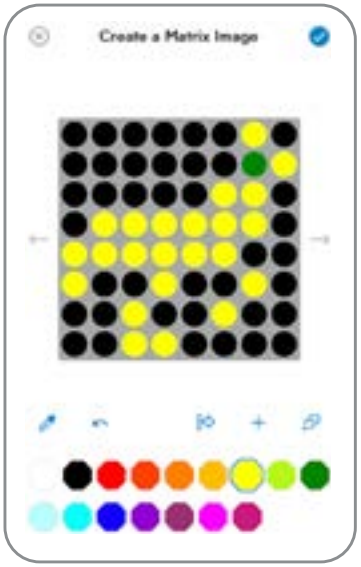
Challenge Card #5

BEGINNER

*Students program their own pictures with the **matrix animation block**.*

BACKGROUND

The BOLT+ screen has a light matrix emulator that divides the screen into an 8x8 array of circles. The **matrix animation block** allows each of these 64 circles to be programmed to show colors and create original pictures and animations. The **matrix animation editor** has a lot of features and it can take some getting used to. The best way to learn how to use it is through play and experimentation.



PROBLEM SOLVE

Students will create their own pictures. **Encourage them** to keep their pictures **simple**. Detail is difficult with only 64 circles.

If your students are ready or interested, this could be a good opportunity to talk about **pixel density**. The BOLT+ matrix screen is 8x8 and only has 64 circles, but the screen itself is 128x128 and has 16,384 tiny circles that are called pixels. The smaller the circles are, the more we can fit into the screen and the greater detail your images will have. For example, a 4k TV has 3840x2160 or 8,294,400 pixels which is why the image is so crisp.

PLAY

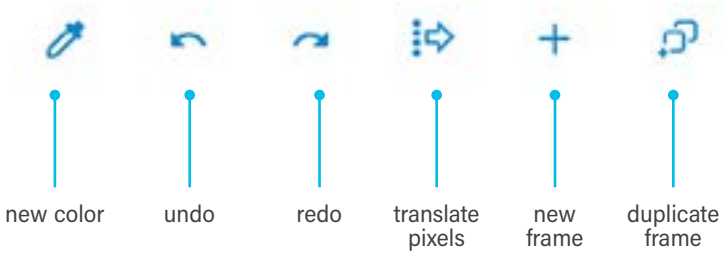
Add frames to animate your picture.

After students have created their picture, they can **add a frame** by selecting the **add frame button**. Then they can draw their second picture on the matrix. When executed, the block will now loop between the two pictures.

Students can also use the **duplicate frame button** to create a new frame with the same image. Then they can make small modifications or translate the images to make smooth animations.

*Try all the buttons in the **matrix animation editor**.*

The labeled image shows the function of each button. To quickly animate an image students have drawn, they can duplicate the frame and then translate the pixels in the new frame up, down, left, or right.



EXTRA PRACTICE

- **Ask students** to explore some of the Sphero animations at the bottom of the **matrix animation editor**. They can edit them to decompose how many frames they have and how they were created. This will help them create their own, more complex animations.

CHALLENGE #6

FADE TO BLACK

Challenge Card #6

BEGINNER

*Students learn how to **fade** the BOLT+ LEDs from one color to another.*

BACKGROUND

The **fade block** changes the color of all 6 LEDs on top of BOLT+ from one color to another over a duration, specified in seconds.

Choose colors using the **color wheel** or by **entering specific values** for red, green, and blue (RGB). You can also adjust the brightness level using the slider.

RGB values are often represented like this: R: 255, G: 255, B: 255. In this example, the maximum value (255) is selected for each color resulting in white. If you were to change each value to 0 (R:0, G:0, B:0) it would be black.

Each input can be changed by selecting its oval inputs. For more precise values, tap or click the numbers icon to bring up a numpad.

PROBLEM SOLVE

Students will modify the program to fade from one color to another.

PLAY

*Use **fade blocks** to show your three favorite colors.*

Students can use three **fade blocks** to make the LEDs on BOLT+ fade from their first to their second to their third favorite color.

Set red to 255, blue to 255, and green to 0. Which color does that make?

This combination of red and blue makes pink. Challenge students to figure out the RGB combinations for other common colors.

*Drag the **random color block** from the operators category into the color input. What happens?*

When placed in a color block, the **random color block** will choose a random color. It does this by selecting three random numbers: one for red, green, and blue.

EXTRA PRACTICE

- **Direct students' attention** to the **strobe block**. The **strobe block** flashes a color for a specified amount of time and a specified number of times. For example, the following block will flash red ten times, on for 0.5 seconds and off for 0.5 seconds, in essence making a ten-second timer.

- Play a song that your students like with a consistent rhythm. **Ask students** to create a light show to go with the song. When the programs are ready, turn off all the lights and put on a musical light performance.

CHALLENGE #7

A RING OF LIGHTS

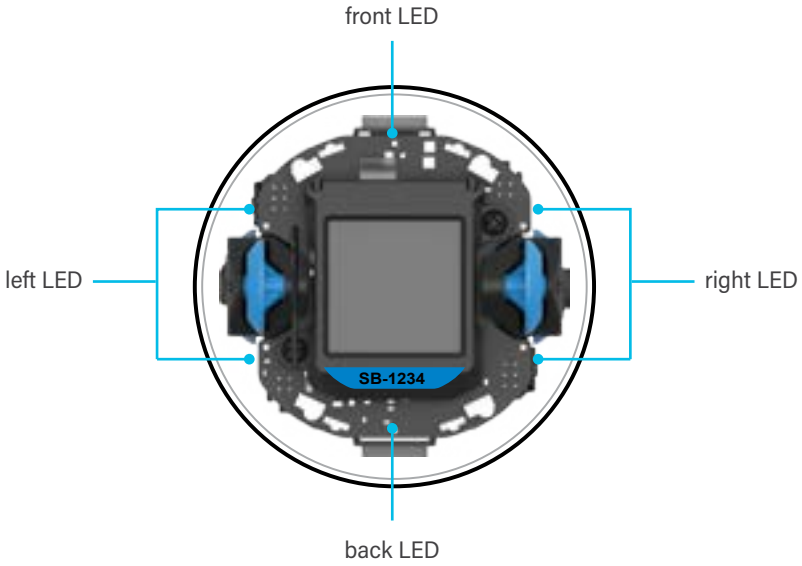
Challenge Card #7

BEGINNER

Students program BOLT+ to turn on and off LEDs to make a light show.

BACKGROUND

BOLT+ has six LEDs around its circuit board. Use the **front**, **right**, **back**, and **left LED blocks** to set these LEDs to specific colors using red, green, and blue color values. Learn more about RGB color values in the support for ***Challenge Card 6: Fade to Black***.



PROBLEM SOLVE

Students have to **add more blocks** to the program so that LEDs move around BOLT+. Here's what the completed program should look like.

PLAY

Change the color of the LEDs to your favorite color.

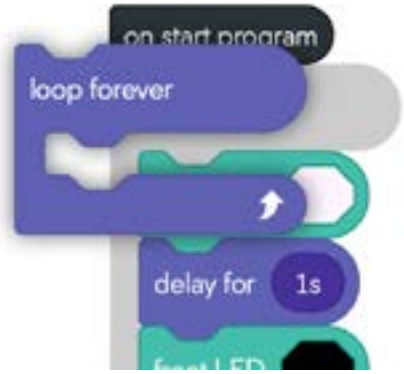
Students can either set the color with the color wheel or by inputting RGB values directly.

Change the time in the delay blocks so the lights move faster or slower.

Increasing the time in the **delay blocks** will make the lights travel around BOLT+ more slowly. Decreasing the time will make the lights turn on and off more quickly. Prompt students to find a speed that they like best.

Add a block from the controls category to make the ring of light repeat forever.

Students can try either the **loop x times** or the **loop forever block**. When they drag either of these two blocks over the first block in the program, the loop will wrap around all the blocks below.



EXTRA PRACTICE

- **Challenge students** to make other light patterns with the **front**, **right**, **back**, and **left LED blocks**. Then have them incorporate some of these patterns into a light show with other blocks in the **light category** of the block library.

CHALLENGE #8

FOUR CORNERS

Challenge Card #8

BEGINNER

Students learn how to program individual LEDs on the matrix.

BACKGROUND

The **matrix pixel block** sets a small area of the display to a specific color using the coordinate system. Coordinates start at (0, 0) in the lower left corner. Note that the coordinates run from 0 to 7 on both the X and Y axes, not 1 to 8.

To clear the pixels, use the **clear matrix block**.

PROBLEM SOLVE

Here is one possible solution. The order of the **matrix pixel blocks** can be changed.

PLAY

Create your first initial on the matrix.

Students can use the **matrix pixel block** to turn on individual pixels and make a letter.

Experiment with other blocks that make it easier to display your initial.

Students can also use other blocks to make their first initial, including the **display text** or **matrix character block**.

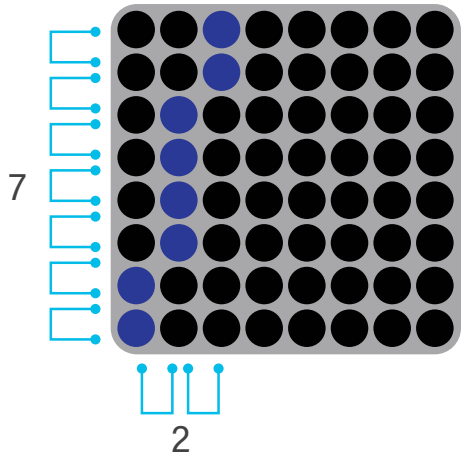
*Turn **stabilization** on. What happens?*

The BOLT+ left and right motors will be enabled and will work to keep the BOLT+ matrix facing upward. Turning stabilization off is a handy trick if students ever want to hold up their BOLT+ and show what is on the matrix to a classmate.

EXTRA PRACTICE

- After students have created their initial with the matrix pixel block, **challenge them** to make it spin 360° with a series of **delay** and matrix **rotation blocks**.
- Connect this block to learning about coordinate grid systems in math. When they are ready, use the **matrix line block** and ask them to calculate the slope (rise/run) between two points.

NOTE: **Think of that dot in the very bottom left to be the origin (0,0). From there you "hop" two places to the right and up seven places.**



$$\text{slope} = \frac{7}{2} = 3.5$$

CHALLENGE #9

METAMORPHOSIS

Challenge Card #9

BEGINNER

*Students explore the **BOLT+ image library**.*

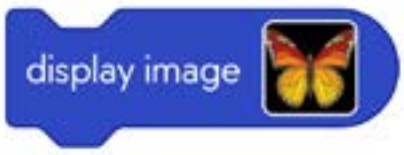
BACKGROUND

The **display image block** has hundreds of images. When selected, students will be able to search or browse the image library by category to find an image to display on the BOLT+ screen. Some images, like the ones about the life cycle of a butterfly, go together particularly well.

NOTE: The **display image block** does not include a time duration. Use a **delay block** in between **display image blocks** to see them for longer periods on the BOLT+ screen.

PROBLEM SOLVE

Students will need to add one more **display image block** and select the butterfly image to finish the life cycle.



PLAY

Show a different life cycle on the BOLT+ display.

Students can choose from a variety of other life cycles in the BOLT+ image library, including the life cycle of a frog or tomato plant.

*Try the **display animation block** in the program.*

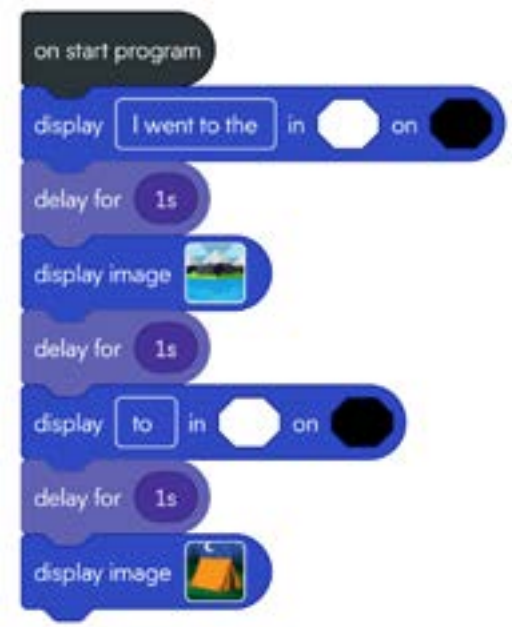
The **display animation block** offers more than 30 animations to show on the BOLT+ screen. Students can choose to show an animation once or loop the animation, similar to the controls on the **matrix animation block**.

Add a block to clear the display.

To clear the screen, students need to add the **clear display block**, the same block they would use to clear text from the screen.

EXTRA PRACTICE

- **Ask students** to browse the image library for other examples of plant or animal life cycles. Have them modify their existing program to demonstrate the new life cycle.
- **Challenge students** to tell a story with words and images. Share the following image as an example.



CHALLENGE #10

PROGRAM A CIRCLE

Challenge Card #10

BEGINNER

Students learn how to make BOLT+ roll and spin at the same time.

BACKGROUND

The **spin block** has two inputs: spin degrees and duration. BOLT+ will spin a specified number of degrees over a specified amount of time. 360° is equal to one full revolution.

The **speed block** makes BOLT+ roll forward at speed 100. It will continue rolling at this speed until it is told to stop. The **spin block** makes BOLT+ rotate 360° over a duration of 3 seconds. The combination of rolling while spinning makes BOLT+ trace a circle. After BOLT+ finishes executing the **spin block**, the **stop block** sets the speed to 0.

PROBLEM SOLVE

Students will observe that BOLT+ rolls in a circle. They can make the circle larger by either increasing the duration in the **spin block** or by increasing the speed in the **speed block**.

PLAY

Find another way to modify the program to make a larger circle.

Students should find two different ways to make BOLT+ make a larger circle:

- increase the duration in the **spin block**
- increase the speed in the **speed block**

*Remove the **stop block**. What happens?*

Without the **stop block**, BOLT+ will continue to roll in a straight line after the **spin block** finishes executing.

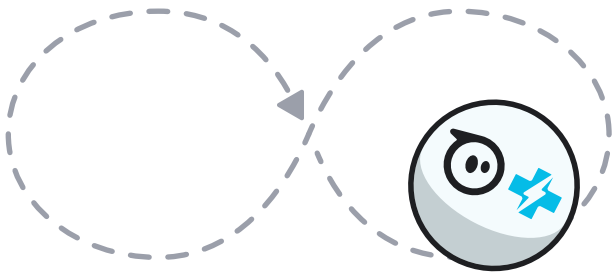
Modify the program to make BOLT+ roll in a circle in the opposite direction.

Students have two options:

- change the **speed** to a negative number (i.e. change 100 to -100)
- change the **spin degrees** to a negative number (i.e. change 360° to -360°)

EXTRA PRACTICE

- Connect this learning to **math** and give students more opportunities to program circles with accuracy and precision. Get out rulers or measuring tape and ask students to make circles with diameters of 1, 2, and 3 meters. With older students, use formulas to calculate the circumference and check the accuracy of BOLT+'s **total distance sensor**.
- **Challenge students** to figure out how to program a **figure eight path**. To do this students will need to program two circles both with the same start point: one that goes in a clockwise direction and one that goes in a counterclockwise direction. A delay in between the circles will help BOLT+ change direction smoothly.



CHALLENGE #11

RANDOM ROLLER

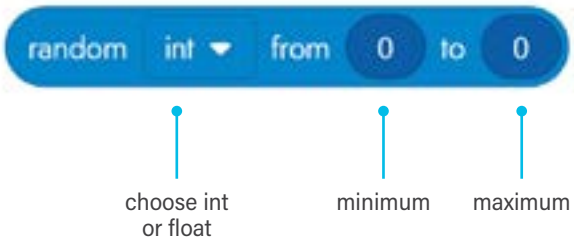
Challenge Card #11

INTERMEDIATE

Students make BOLT+ roll in random directions forever.

BACKGROUND

The **random int block** generates a random number from the range that includes the **minimum** and **maximum** number.



Int stands for **integer** or the set of positive whole numbers (1, 2, 3,...), negative whole numbers (-1, -2, -3,...), and zero (0). **Float** includes both **decimals** and **integers**.

PROBLEM SOLVE

Students place a **random int block** in the **speed** and **duration** inputs.

- The **range** for the **speed input** can be set from 0 to 255.
- The **range** in the **duration input** should be set from 0 to 5. Students should not exceed 5 seconds in the max input so that BOLT+ doesn't roll for too long in any one direction.

PLAY

*Remove the **loop forever block**. What happens?*

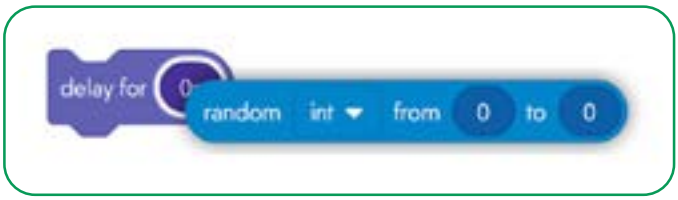
If students remove the **loop forever block**, BOLT+ will only execute one roll block and then stop.

*Change the numbers in the **random int blocks**.*

Changing the numbers increases or decreases the range. For example:

- limiting the heading from -45° to 45° will make sure BOLT+ doesn't travel backwards
- keeping the max speed at 255 may make BOLT+ roll out of control

Add **random int blocks** to other blocks.
The **random int block** will fit in any circular input.



YES

NO

EXTRA PRACTICE

- Make this challenge into a goofy **racing game**. All students place their BOLT+ robots on start with a finish approximately 10 feet away. Agree upon ranges for the minimum and maximum in the heading and/or speed input and then start the programs all at the same time. *Whose robot wins?*

CHALLENGE #12

DATA BALL

Challenge Card #12

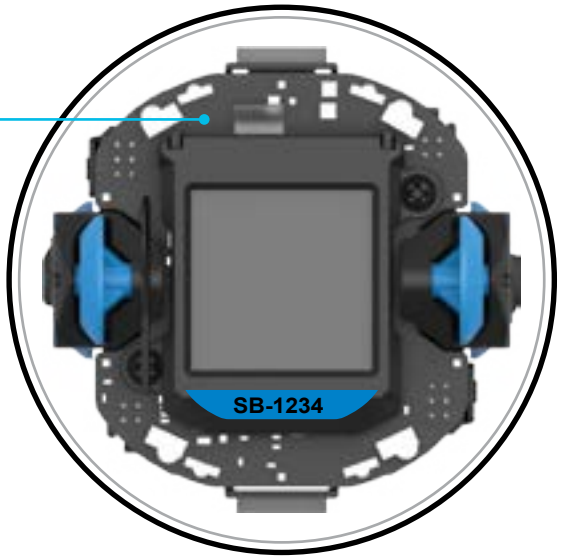
INTERMEDIATE

Students learn how to make BOLT+ show sensor values on its display.

BACKGROUND

The **inertial measurement unit (IMU)** inside of BOLT+ captures data about its velocity, acceleration, and orientation.

The **display sensor value block** will show the sensor value from the IMU on the BOLT+ screen.



PROBLEM SOLVE

Students run the program and discover that:

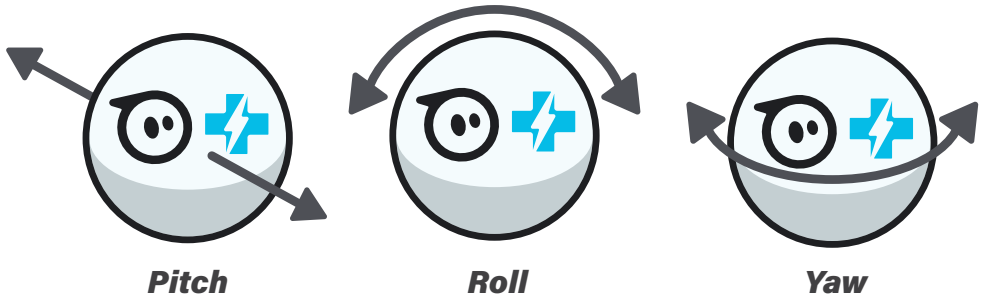
- **total accelerometer > 1g** when BOLT+ is being shaken (the IMU is being bounced back and forth against the BOLT+ shell).
- **total accelerometer < 1g** when BOLT+ is in freefall (the IMU is experiencing weightlessness).
- **total accelerometer = 1g** when BOLT+ is at rest (the IMU senses the force of gravity on Earth's surface).

PLAY

*Choose **orientation** in the **display sensor data block**. What data is BOLT+ reading now?*

Choosing orientation will report the position of BOLT+ in three different axes:

- **Pitch:** The forward or backward tilt of BOLT+ from -180° to 180°
- **Roll:** The right or left tilt of BOLT+ from -90° to 90°
- **Yaw:** The spin (twist) angle of BOLT+ from -180° to 180°



Pitch ***Roll*** ***Yaw***

Select something else from the drop-down menu. What do you notice?

Give students time to explore this block on their own. If you have time, ask students to share what they noticed.

EXTRA PRACTICE

- **Prompt students** to use the **display sensor block** to learn more about when **if then blocks** trigger other blocks to execute. In the following example, students can notice the value of the total accelerometer when the abrupt sound is played. The total accelerometer should be above 2g.

CHALLENGE #13

DRIVE COUNTDOWN

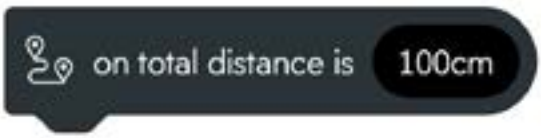
Challenge Card #13

INTERMEDIATE

Students learn how to use an event to change the behavior of BOLT+ after it has traveled a certain distance.

BACKGROUND

Event blocks cause the blocks attached to them to execute when a specified event occurs. The **on distance event** triggers after BOLT+ has traveled a specified distance, as approximated from data gathered from the BOLT+ motors. All distances are measured in centimeters.



Notice that the **on start program block** is an **event block** itself. This is the only block that is always on the canvas and cannot be deleted. BOLT+ will execute the blocks attached to the **on start block** whenever Start is selected at the top of the program.

PROBLEM SOLVE

BOLT+ will travel for around 100 cm before stopping and exiting the program. Students can increase the number in the **on distance event** to make BOLT+ travel farther. For example, if they insert 200, BOLT+ will travel around 200 cm.

PLAY

Change the speed to 50. Change the speed to 200. What's the difference?

BOLT+ should travel around the same distance whether the speed is set to 50, 100, or 200. The difference will be how quickly BOLT+ moves from point A to point B. Students should notice that faster speeds, like 200, will result in less consistency in how the program executes. Because of inertia, BOLT+ may take longer to come to a complete stop after the **on distance event** is triggered.

*Swap the **on time elapsed** event for an **on distance event**.*

The **on time elapsed event** is similar to the **on distance event** except instead of BOLT+ stopping after traveling a certain distance, BOLT+ will stop after a specified amount of time from when the program started. For another use of the **time elapsed sensor**, check out ***Challenge Card 17: A BOLT+ Timer***.

EXTRA PRACTICE

- Turn this concept into a fun little game. Replace the **speed block** connected to **on start program** with the **drive on block**. Replace the **stop block** connected to **on start program** with the **drive off block**. Set up short obstacle courses or mazes and see if students can drive BOLT+ through the course before their distance runs out!

CHALLENGE #14

KNOCK-KNOCK MACHINE

Challenge Card #14

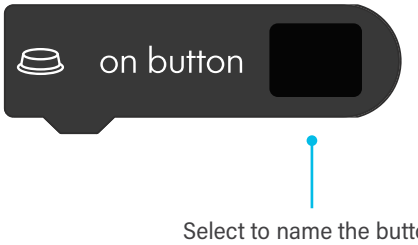
INTERMEDIATE

Students learn how to run blocks of code with the push of a button.

BACKGROUND

Many programming events are triggered by the sensors inside of BOLT+, but **on button events** are triggered by someone pressing a button in the programming app while a program is executing. You can add up to three **on button events** to any program. Click or tap the input on the button event to give it a unique and recognizable name.

After all the blocks in an event are executed, BOLT+ will continue executing the main program.



Select to name the button

PROBLEM SOLVE

When students want to start the joke, they should press button 1. Then they should press button 2 and button 3 to finish the joke. Students can change the strings in all three of the **on button events** to tell their favorite knock-knock jokes.

PLAY

*See how many **on button events** you can add to a program.*

Programmers are only allowed to add three **on button events** to a program.

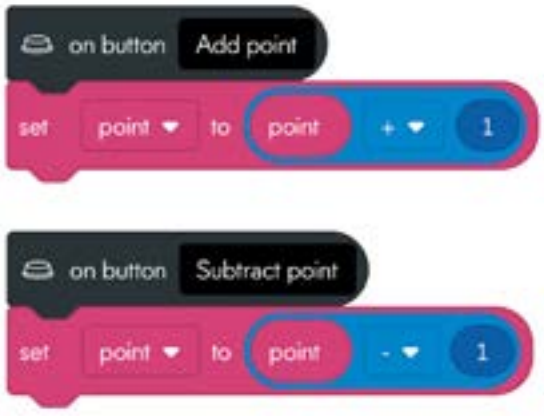
*Use an **on button event** to trigger a different BOLT+ behavior.*

The options are nearly limitless here. Students can use the **on button event** to:

- turn on drive
- make BOLT+ roll
- play a sound
- display an image or animation on the screen
- create a combination of many different behaviors

EXTRA PRACTICE

- Modify the Data Ball Challenge to use an **on button event**. **Ask students** to create a program that tells BOLT+ to display different sensor values when different buttons are pressed.
- **Show students** how to use **on button events** to add 1 to or subtract 1 from a number variable. **Discuss** how this programming concept could be applied, like to show a score that changes over time.



CHALLENGE #15

BOLT+ FEELS SICK

Challenge Card #15

INTERMEDIATE

*Students learn how to trigger blocks with an **on gyro max event**.*

BACKGROUND

Some BOLT+ programming events are detected by the sensors—like the **inertial measurement unit (IMU)**—inside of BOLT+. The IMU measures BOLT+'s acceleration and rotation.

The **on gyro max event** executes any attached blocks when BOLT+ is rotated at a rate greater than 2,000° per second (5.5 revolutions per second). In other words, this event is triggered by spinning BOLT+ around really fast. You need to spin it around faster than 5.5 revolutions per second.

PROBLEM SOLVE

When students spin BOLT+ in either direction faster than 5.5 revolutions per second, BOLT+ will play the "vomit" sound, show the dizzy robot eyes animation for 3 seconds, and then return to the regular eyes.

PLAY

*Turn **stabilization** on. What happens?*

If stabilization is turned on, it will be difficult to trigger the **on gyro max event** because the BOLT+ motors will keep the robot pointed in the direction it is aimed.

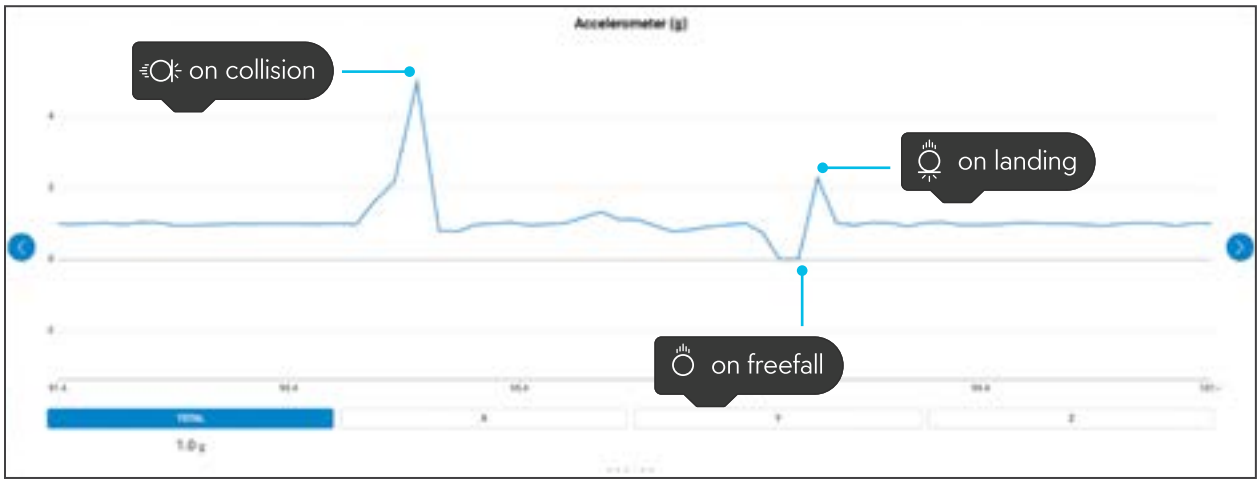
*Try other **event blocks** like **on collision**, **on freefall**, and **on landing**.*

Other events that use data from the BOLT+ IMU are fun to swap into the program, too:

- **on collision** will trigger when BOLT+ detects a collision or a jolt (total acceleration > 2.5g)
- **on freefall** will trigger when BOLT+ detects weightlessness (total acceleration = 0g)
- **on landing** will trigger when BOLT+ detects a collision after being in a state of weightlessness (total acceleration = 0g followed by total acceleration > 2.5g)

*Look at the **sensor data**. What does it tell you about when the **event blocks** are triggered?*

Sensor data is viewable in real time as a program executes on most devices. It can also be downloaded as a CSV file by selecting the three dots in the upper right corner of a program.



EXTRA PRACTICE

- Events are fun for playing sounds on command. **Challenge students** to build a simple program that plays their favorite sound and shows their favorite animation whenever BOLT+ is shaken!

CHALLENGE #16

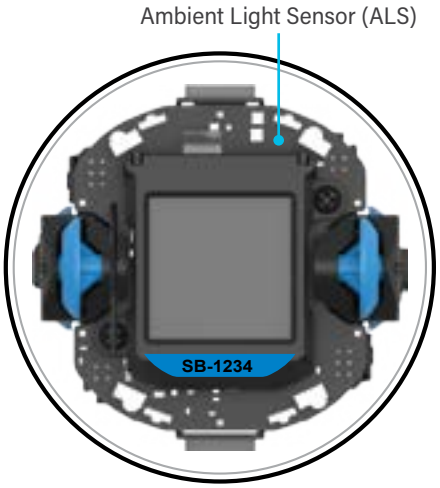
LIGHT AT THE END OF THE TUNNEL

Challenge Card #16

INTERMEDIATE

*Students program BOLT+ to read sensor values and execute different **speak blocks**.*

BACKGROUND



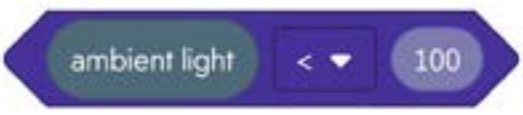
BOLT+ has a **light sensor**! Can you find it? Look closely for the text "ALS," which stands for **"Ambient Light Sensor,"** on the very front of the BOLT+ circuit board. The light sensor reports the intensity of ambient light from 0–100,000 lux, where 0 lux is full darkness and 30,000–100,000 lux is direct sunlight. The ambient light value in the average classroom is anywhere from 100 to 400 lux.

The program uses an **if then else block** to change which **speak block** BOLT+ executes in different lighting conditions. The conditions are established with a **hexagonal comparator block**.

- If ambient light is less than 300, the program says, "It's dark in here."
- If ambient light is not less than 300, the program says, "I can see!"

PROBLEM SOLVE

Students will need to lower the value in the comparator until the program says, "I can see!" in your room's light conditions. When the robot is covered the program will say, "It's dark in here." Setting the value to 100 should work for most classroom light levels.



NOTE: **If the BOLT+ LEDs are on, they will impact the light sensor, which will not go to zero.**

PLAY

Change the loop to something that won't go on for... forever.

If students remove the **loop forever block**, the program will only check the BOLT+ light sensor one time and then either say "It's dark in here" or "I can see!" Students could also replace the **loop forever block** with the **loop x times block** or even the **loop until block**.

Change what the program says in different light conditions.

Students can select the string in the **speak block** to type something else for the program to say.

*Add sounds and images to go with the **speak blocks** in this program.*

There are lots of sounds in the sound library. Two that might be relevant to this program are the "snore" and "yawn" sounds in the **Personality category**. Additionally, they can use the **display image block** to show an image of the sun when ambient light is above a certain level and an image of the moon when ambient light is below a certain level.

EXTRA PRACTICE

- **Ask students** to change **less than (<)** to **greater than (>)** in the **comparator block** so that it reads luminosity > 100. Then **challenge students** to change the blocks in the program to make BOLT+ behave the same.
- **Ask students** to program BOLT+ to act as a backpack alarm. When a backpack is opened, the BOLT+ light sensor should trigger an alarm and alert the owner.

CHALLENGE #17

A BOLT+ TIMER

Challenge Card #17

INTERMEDIATE

*Students use the **time elapsed sensor** to control when BOLT+ exits a loop.*

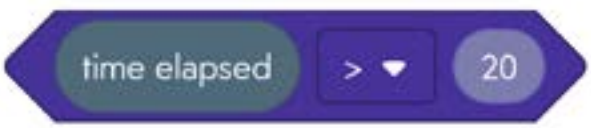
BACKGROUND

The **loop until block** repeats the block(s) inside until a specified condition in a **comparator block** is met. In this case, the condition uses the **time elapsed sensor**, which returns the number of seconds the program has been running.

- **While time elapsed is not > 10**, the program will repeat the **strobe block**.
- **After time elapsed is > 10**, the program will continue to the rest of the blocks in the program.

PROBLEM SOLVE

To modify BOLT+ into a 20-second timer, students need to change the value in the **comparator** from 10 to 20.



PLAY

Make the program show a different animation or play a different sound.

Students can change animations and sounds or even add other blocks to customize their timer.

*Experiment with the inputs on the **strobe block**.*

The strobe block has three inputs:

- color is the color of the six LEDs
- period is the time, in seconds, the LEDs stay on during a single blink
- cycles are the total number of blinks

*Invent a game you could play with the **random int block**.*

Students can create many games with this simple program. A simple game is a variation of hot potato where students pass BOLT+ around until the time is up. Then, that person has to answer a question!

EXTRA PRACTICE

- Have students recreate the functionality of the program with the **on time elapsed event**. Here is one possible solution:

- Have some fun with the **strobe block**. **Challenge students** to time the period to the rhythm of a favorite song. Then add a **random color block** from the **operators category** to the color input and put on a light show.

CHALLENGE #18

A POLYGON ALGORITHM

Challenge Card #18

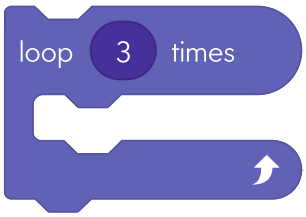
INTERMEDIATE

Students modify an algorithm to program a different polygon.

BACKGROUND

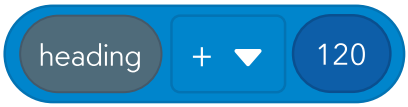
This program makes BOLT+ roll in an equilateral triangle. You can also do this with three **roll to distance blocks**, but the advantage of this program is that it is easy to modify for any regular polygon. Three blocks are key to making this algorithm work:

- The **loop x times block** repeats the blocks inside of the loop a specified number of times.



- The **heading block** is a sensor block. It returns the current heading or direction BOLT+ is facing in degrees from 0° to 359°.

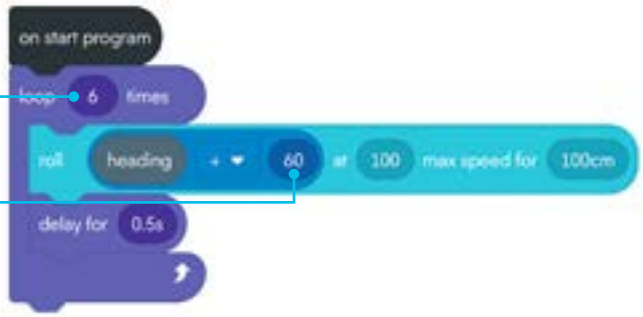
- The **math operator block** adds two numbers together, in this case, it adds the heading + 120 each time the roll to distance block is executed. You can use the drop-down menu inside of the math operator to select different math operations, like subtraction (–), multiplication (*), and division (/).



PROBLEM SOLVE

To program BOLT+ to roll in a hexagon, students need to modify two **number values**:

- change the number in the **loop x times block** from 3 to 6.
- change the number in the **math operator** from 120° to 60°.



PLAY

Modify the program to make polygons with longer sides.

To increase the length of the sides, students should increase the distance in the **roll to distance block**.

Make other polygons like squares, octagons, or decagons.

Students should be able to make any polygon they want by changing the number inside of the **loop x times block** to the number of sides in the chosen polygon and changing the number in the math operator to the value of 360 divided by the number of sides in their chosen polygon. For example, for a pentagon, students would need to change the:

- loop x times block to 5
- value in the math operator to 360 ÷ 5 = 72°

EXTRA PRACTICE

- Keep adding more and more sides and the polygon will begin to look like a circle. **Challenge students** to make a regular polygon with 36 sides to see if the sensor data resembles a circle.
- **Ask students** to modify distances in the **roll to distance blocks** to make polygons with specific perimeters.
- This algorithm makes regular polygons. **Challenge students** to change the algorithm to make a rhombus, a quadrilateral with equal opposite angles.

CHALLENGE #19

FLYING BOLT+

Challenge Card #19

INTERMEDIATE

Students figure out how to make BOLT+ play a sound like it is flying!

BACKGROUND

The **total velocity sensor** returns the speed at which BOLT+ is moving in centimeters per second.

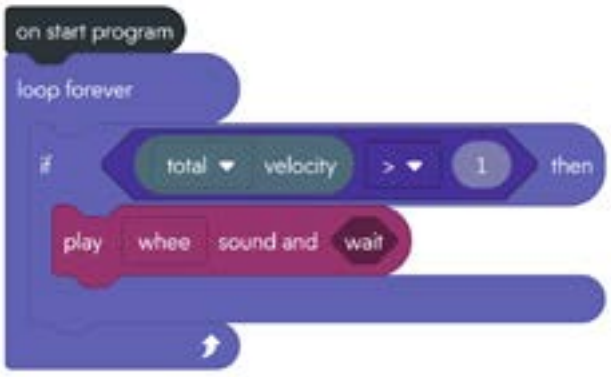
The program then uses an **if then block** to play the "whee" sound if BOLT+ is moving greater than 1 centimeter per second (not very fast!). Notice the difference between an **if then block** and an **if then else block**.

- **if then blocks** will do nothing if the condition is not met.
- **if then else blocks** will execute a second set of blocks if a condition is not met.

NOTE: **The total velocity data, as well as any BOLT+ data that involves distance, can be inaccurate due to limitations of the IMU and how BOLT+ uses its motors to estimate movement which is impacted by the type of surface BOLT+ is traveling on.**

PROBLEM SOLVE

Students need to wrap the blocks in a **loop forever block**. The starter program will only check the value of total velocity once, right when the program is started.



PLAY

*Adjust the **velocity value** to something other than 1.*

Encourage students to play around with the values in the comparator. Here are some possible adjustments:

- **If total velocity > -5**, the program will always play the "whee" sound because the total velocity of BOLT+ is always positive.
- **If total velocity > 1000**, it will be very difficult to make the program play the "whee" sound because it is difficult to move BOLT+ that quickly.

*Adjust the total velocity to the **x-axis** or **y-axis**. Can you tell the difference?*

The total velocity sensor combines the velocities along both the x- and y-axis. Note that unlike the total velocity sensor, the x-axis and y-axis velocity sensors can return negative values.

- **X-axis** will measure the BOLT+ speed right (+) or left (–).
- **Y-axis** velocity will measure the BOLT+ speed forward (+) or backward (–).

EXTRA PRACTICE

- Investigate velocity further with BOLT+ movement. **Ask students** to manually calculate BOLT+'s velocity by measuring the distance it travels over an amount of time. Then **compare** this to BOLT+'s sensor data. Do they match up?
- **Ask students** to gather data with the **display sensor value block**. What is the velocity of different BOLT+ speeds on different surfaces? Try testing with speed set to 50, 100, 150, 200, and 250. Is there a linear relationship?

CHALLENGE #20

RED LIGHT GREEN LIGHT

Challenge Card #20

INTERMEDIATE

Students explore how to adjust the number in a ***loop x times block*** *to program a game.*

BACKGROUND

All the blocks in this program have been introduced in other challenge cards. To learn more about individual blocks refer to the following challenges:

- **drive block:** Challenge 2: Ready, Set, Drive!
- **random int block:** Challenge 11: Random Roller
- **loop x times block:** Challenge 18: A Polygon Algorithm

PROBLEM SOLVE

When students change the number in the **loop x times blocks** to 10, the program will repeat the blocks inside 10 times. This means the program will enable and disable drive a total of 10 times. Depending on the random integers selected, drive could be enabled for a minimum of 20 seconds or a maximum of 40 seconds. Depending on the size of the room, this is likely enough time to drive from one side to another.

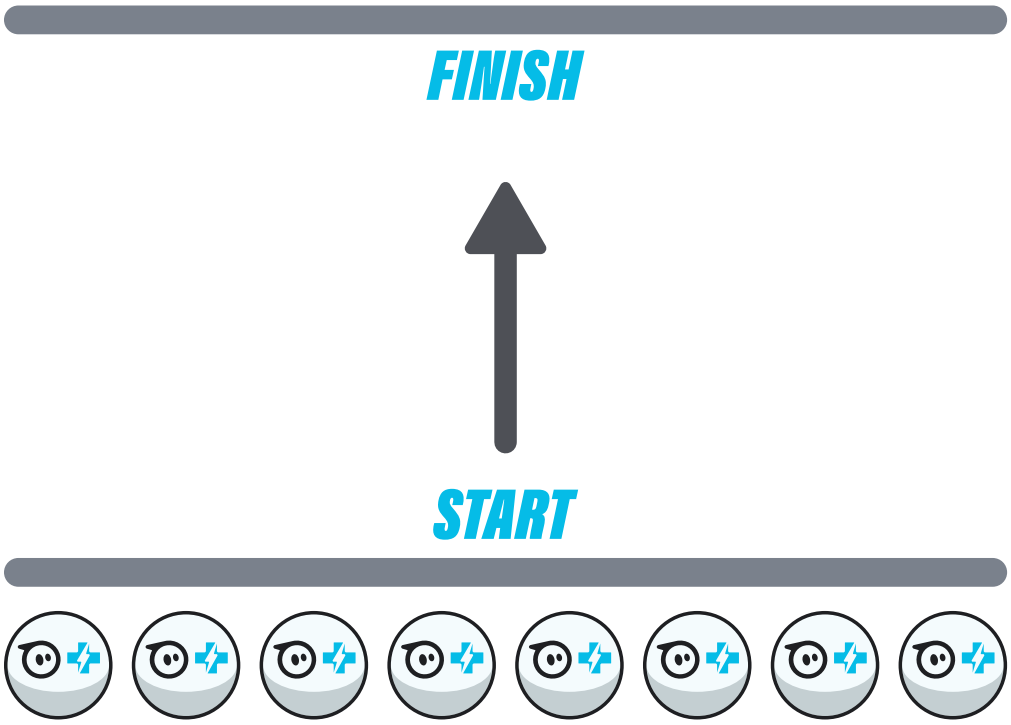
PLAY

Add images or animations to go along with the drive on and drive off blocks.

Encourage students to have fun adding more blocks to the program to make it their own. They can add images and animations to represent starting and stopping. They can also add fun **sound** and **speak blocks**.

Make this into a game you play against a partner.

There are lots of fun games that can be made from this programming pattern of randomly turning on and off **drive**. Imagine a classroom game of red light green light where all students place their BOLT+ robots on the start line and execute their program at the same time. Each person will need some skill and some luck to make it to the finish line before their peers.



EXTRA PRACTICE

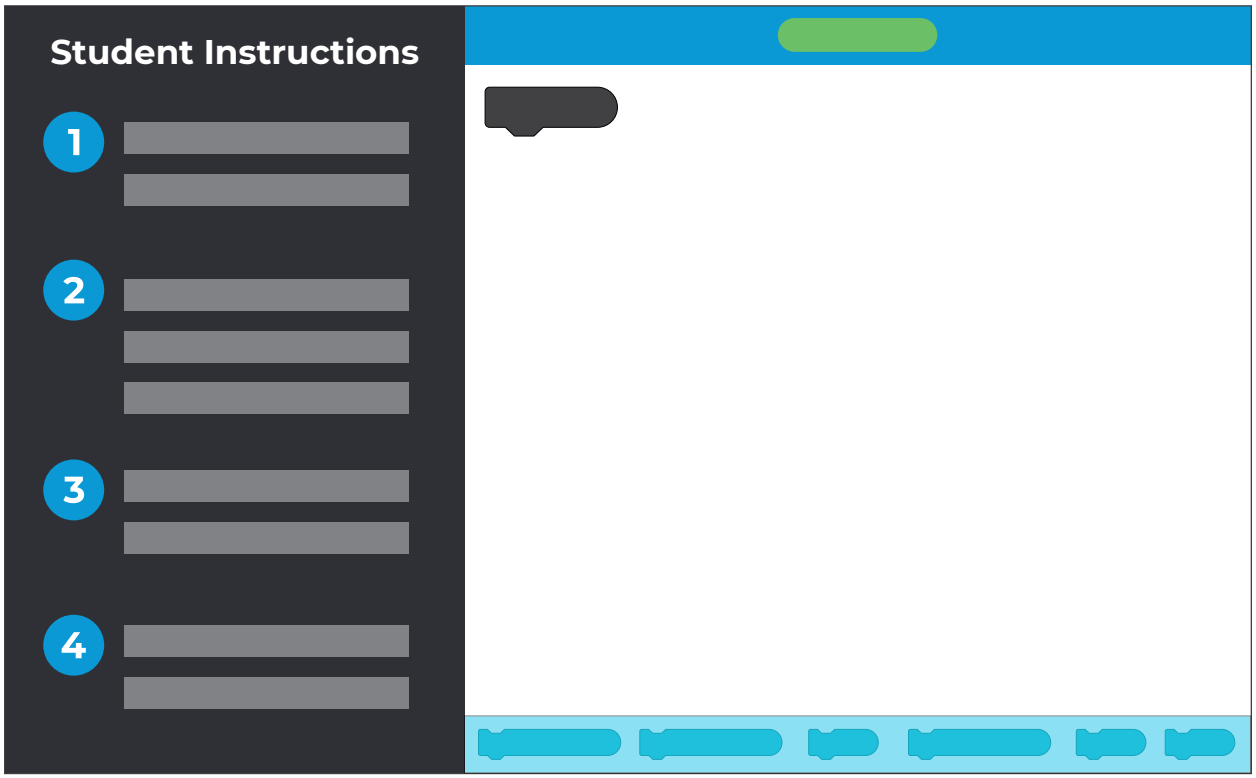
- Students have learned a lot at this point. If you think they're ready, try offering a few of the following prompts that ask students to **create their own program from scratch**:
  - Make BOLT+ drive in random directions at random speeds and make a noise and display an animation if it crashes into something.
  - Create a program that makes your programming device tell the beginning, middle, and end of a popular story using the **on button events**.
  - Design a program that allows you to drive BOLT+ when the ambient light is less than 100 but drives on its own in random directions when ambient light is greater than 100.

NEXT STEPS

LESSONS & SUPPORT

READY FOR MORE?

Finished with the challenge cards? Try some Sphero lessons with your students. A lesson combines step-by-step instructions with a program.



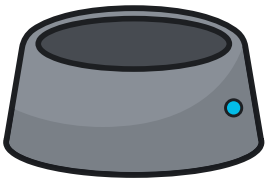
Go to sphero.cc/bplus-lessons to browse to find the right materials for your classroom!

LOOKING FOR SUPPORT?

Go to our support page: sphero.cc/bplus-support

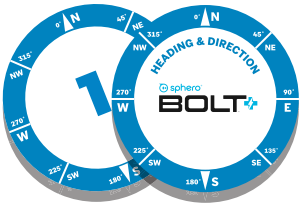
WHAT'S IN MY BOLT+ POWER PACK?

15 BOLT+ robots

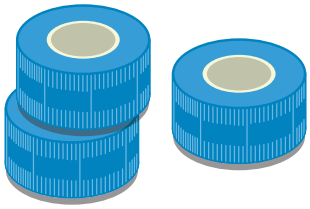


15 inductive charging bases

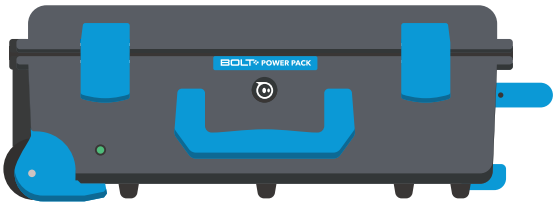
15 Turbo Covers



15 protractors and targets



5 rolls of Maze Tape



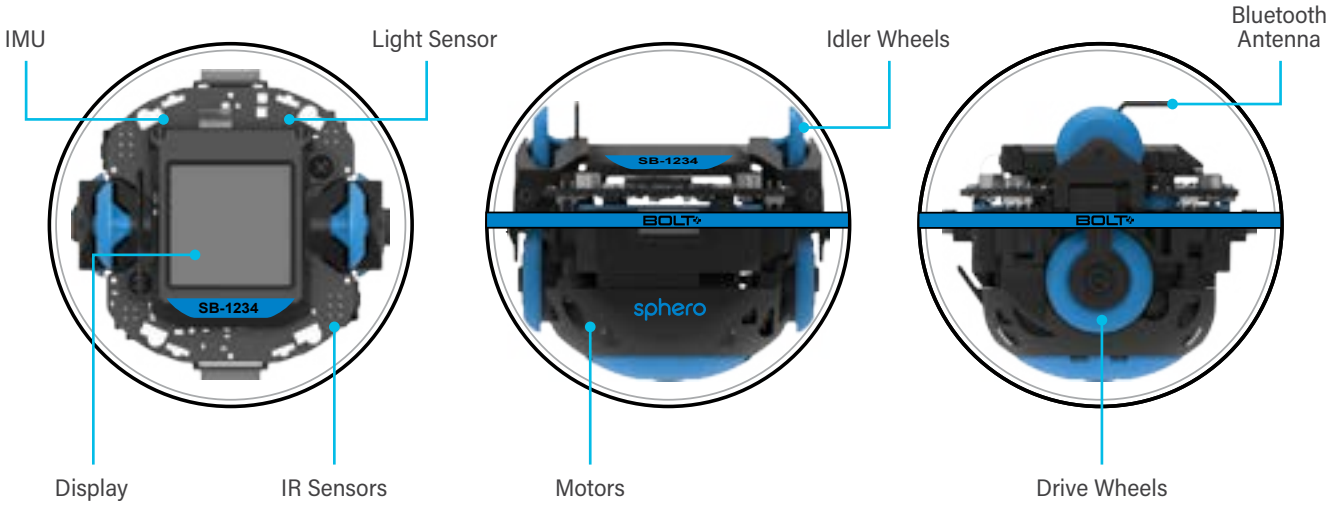
1 transportable, durable charging case

20 challenge cards

This educator guide

FAQs

WHAT'S IN BOLT+?



A lot is going on inside of BOLT+. Here are some highlights:

- **Display:** The 128x128 screen can display hundreds of images and animations as well as live sensor data. It is also able to emulate an 8x8 dot matrix allowing students to create their own images and animations.
- **LEDs:** The 6 LEDs inside of BOLT+ can be programmed individually or collectively to show any color of the rainbow using red, green, and blue color values.
- **Wheel Motors:** BOLT+ can be programmed to roll for a specified duration (in seconds) or a specified distance (in centimeters). Motor encoders are used to calculate location and distance traveled.
- **Inertial Measurement Unit (IMU):** A sensor that captures data about a robot's velocity, acceleration, and orientation. Data from the IMU can be used to detect collisions, changes in speed, and rotational movement around the pitch (x-axis), roll (y-axis), and yaw (z-axis).
- **Light Sensor:** The light sensor reads the light intensity (ambient light) in your environment from 0– 100,000 lux, where 0 lux is full darkness and 30,000–100,000 lux is direct sunlight. The ambient light in most classrooms ranges from 100–400 lux.
- **Infrared (IR) Sensors:** The IR sensors can be used to send and receive messages between BOLT+ robots.

PROGRAMMING OPTIONS

WHAT ARE MY OPTIONS FOR PROGRAMMING BOLT+?

Three Methods of Programming

The Sphero Edu app (sphero.cc/edu-d) offers three different coding canvases: **Draw**, **Block**, and **Text**.

Students can begin by drawing images and shapes on the **Draw Canvas** to see how their BOLT+ recreates them in the real world.

From there, learners can progress to the **Block Canvas**, where they can use dozens of unique coding blocks to program their BOLT+

Finally, when students are ready, they can move on to the **Text Canvas** where they can program their BOLT+ using JavaScript.

FAQs

BOLT+ CARE & STORAGE

HOW DO I TAKE CARE OF MY BOLT+ ROBOTS?

The BOLT+ hardware is encased in a durable polycarbonate shell and is shockproof from falls up to three feet (0.9 meters). While built to withstand falls, we don't recommend testing this theory from, say, the top of a tall building.

BOLT+ is completely waterproof and doesn't have charging ports or openings to worry about.

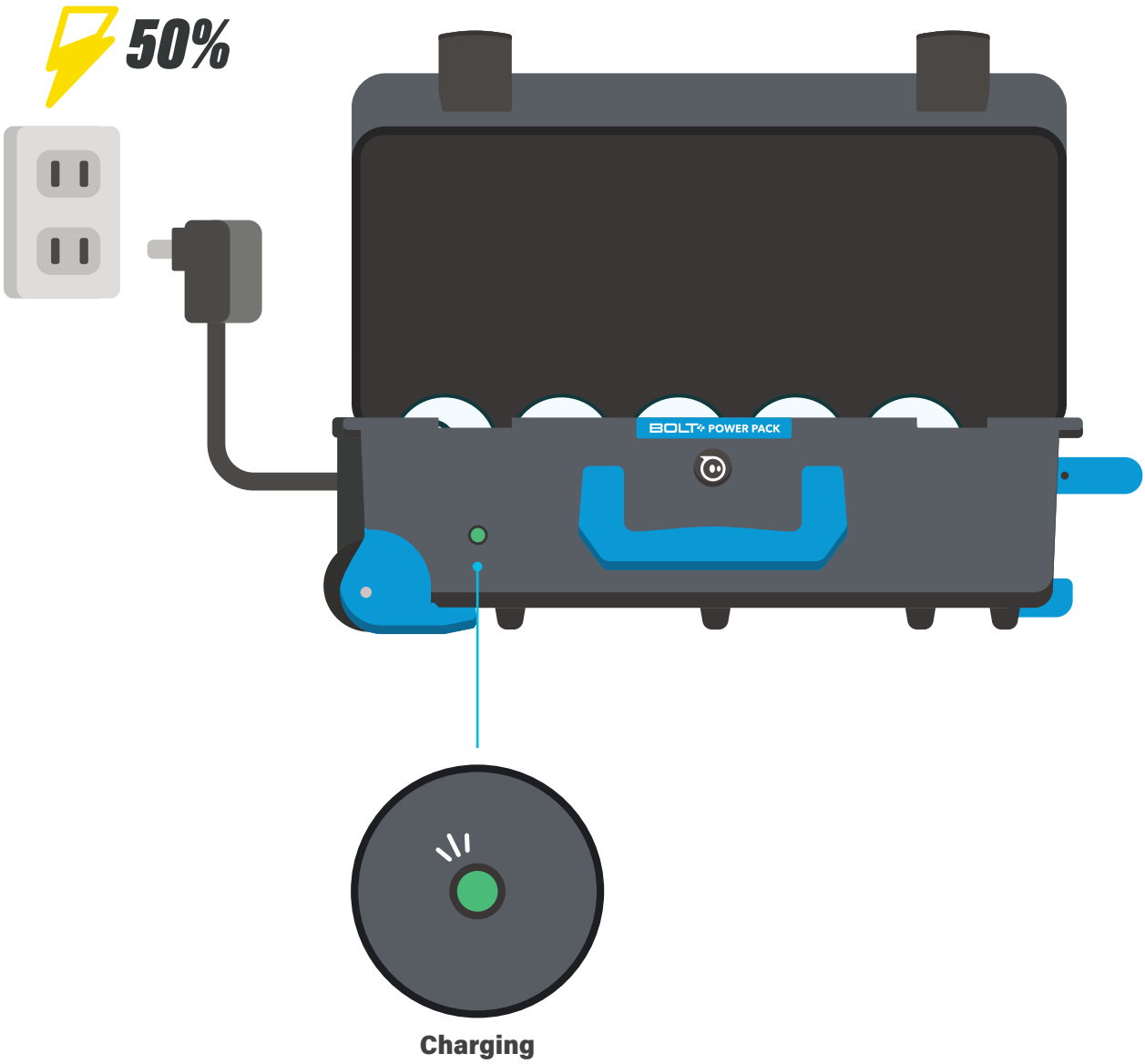
To clean and disinfect your robots:

1. Use your preferred cleaning spray, gloves, paper towels, and/or disinfecting wipes.
2. Spray and wipe away. Wipe down the outer surface of BOLT+. Just be sure not to use harsh solvents or anything abrasive or sharp to clean them.
3. Allow BOLT+ to dry completely before placing it back in the Power Pack or on a charger.

HOW DO I STORE MY POWER PACK WHEN IT'S NOT IN USE?

If you are not using the BOLT+ robots for a few weeks, like over a winter or summer holiday, it's best to return all BOLT+ robots to their cradles in the Power Pack, charge to about 50%, unplug the Power Pack, and store at room temperature.

When you are ready to use your BOLT+ robots again, simply plug in your Power Pack. After storing for several months, plug in your Power Pack, making sure charging indication LEDs are on for each BOLT+ a day ahead of time to make sure the robots are fully charged for student use. From empty, it takes about 8 hours to fully charge a BOLT+.


50%
BOLT POWER PACK
Charging

REFERENCE

GLOSSARY

INSTRUCTIONAL TIP: **Learning domain-specific vocabulary is hard! If you can integrate the terms students are learning into other areas of your classroom, that additional exposure can help with retention. Consider adding the terms to a word wall or, if you have space, creating a "computer science word wall;" if possible, incorporate them into your regular vocabulary work or writer's workshop.**

Asynchronous: Programming that executes multiple blocks or commands at the same time.

Binary: Code with a 0 or 1 value that can be used to represent numbers, letters, or symbols.

Bit: The smallest piece of information computers store; usually represented by a 0 or 1.

Block Canvas: The canvas used to program Sphero robots with blocks.

Boolean: A data type that can either be True or False.

Byte: A byte usually consists of 8 bits and can represent values from 0 to 255.

Comment: Text strings that are used to explain the functionality of code without impacting the program.

Comparators: Used to compare two different values. Ex. 1 > 0, 5 == 5, time elapsed > 10.

Controls: Structures in code that help determine the order in which something occurs. Ex. loops and if true blocks.

Debug: Locate and fix errors in a program.

Draw Canvas: The canvas used to program Sphero robots by drawing lines and shapes.

Event: Pre-written functions that trigger certain code to be executed based on Sphero robot sensors.

Execute: To run or start a program.

Float: A shortened form of floating point number, which refers to a number that usually has a decimal point. Ex. -2.5, 1.0, or 3.145

Function: A piece of code that has a name and completes a specific task; functions can be called (used) as many times as you need in a program.

Heading: Input to program the direction a BOLT+ is pointed during or before a roll, between 0° and 359°.

Inertial Measurement Unit (IMU): A sensor that captures data about a robot's velocity, acceleration, and orientation.

Input: A place to add numeric or text values into a programming block or text statement.

Integer: Positive or negative whole numbers, including 0.

JavaScript: The text programming language used in the Text Canvas; also the language behind every block in the Block Canvas.

Light Emitting Diode (LED): A small device that emits light when electricity is applied to it.

Loop: A control structure that allows you to repeat an action a specified number of times. Ex. loop forever in our Block Canvas.

Operators: Used to assign values to variables (assignment operators), perform mathematical operations (arithmetic operators), or compare values (comparison operators).

Orientation: Orientation describes the spatial positioning of BOLT+ and is measured in three axes:

- Pitch: The forward or backward tilt of BOLT+ from -180° to 180°
- Roll: The right or left tilt of BOLT+ from -90° to 90°
- Yaw: The spin (twist) angle of BOLT+ from -180° to 180°

Parameter: A variable used when creating functions that can pass information to the function when it is called (used).

Pixel: The smallest unit that emits light in a screen and combines with other pixels to collectively form images. Ex. The matrix emulator on BOLT+ imitates an 8x8 screen with 64 pixels; the screen itself is 128x128 with 16,384 pixels.

Pseudocode: Expressing what you want your program to do without writing actual code; typically used in the planning process.

Sensor: A device that detects and measures physical properties and conditions in a robot's surroundings.

Speed: Input to program how quickly (or slowly) a BOLT+ will move; values from 0 to 255.

Statement: A piece of code that completes an action. Ex. if-then statements.

String: A series of letters, numbers, or characters.

Synchronous: Programming that executes one block or command at a time.

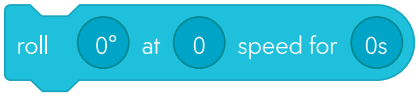
Text Canvas: The canvas used to program Sphero robots in JavaScript.

Variable: Used to store information, like a string or integer, so it can be used or changed throughout a program.

REFERENCE

BLOCK LIBRARY

Movements



Roll: Combine heading, speed, and duration to make BOLT+ roll.

Roll to Distance: Combine heading, maximum speed, and distance (cm) to make BOLT+ roll.

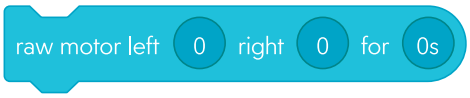
Drive: Turn drive controls on or off to control BOLT+ while a program is running.

Spin: Spin BOLT+ for a specified number of degrees over time.

Speed: Set the target speed for BOLT+ from -255 to 255. Positive numbers are forward, negative are backwards, and 0 is stationary.

Stop: Stop BOLT+.

Heading: Set the direction BOLT+ rolls. 0° is forward in the direction BOLT+ is aimed, 90° is right, 270° is left, and 180° is backwards.



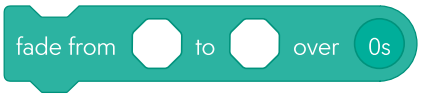
Raw Motor: Control the electrical power sent to the left and right motors independently, from -255 to 255, for a duration of seconds.

Stabilization*: Turn stabilization on or off. When off, BOLT+ will not balance itself inside the shell.

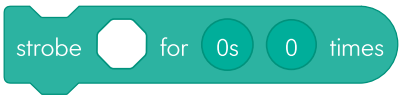
Reset Aim: Reset the aim angle to use the current front-facing direction of the robot as 0°.

Lights

Main LED: Change the color of the LEDs. Set this using the color wheel and brightness slider, or the exact RGB (red, green, blue) values, from 0 to 255.



Fade: Change the LEDs from one color to another, for a duration of seconds.



Strobe: Blink the LEDs for a period of seconds and a count of cycles. A short period will produce a fast blink while a long period will produce a slow blink.

LED Blocks: Change the color of the front, back, left, and right LEDs. Set this using the color wheel and brightness slider, or the exact RGB (red, green, blue) values, from 0 to 255.

Display

Display Animation: Display an animation on the screen.

Display Image: Display an image on the screen.

Display Rotation: Rotate the display direction for animations, images, and text.

Display Text: Display text in a specified font color, background color, and size.

Display Clear: Clear the display, including animations, images, and text.

Display Sensor Data: Choose which sensor data to display.

Matrix

Matrix Animation: Display an image or animation on the 8x8 light matrix.

Matrix Rotation: Rotate the display direction for animations, images, and text on the 8x8 light matrix.

Clear Matrix: Turn off the 8x8 light matrix.

Matrix Character: Display one character on the 8x8 light matrix in a specified color.

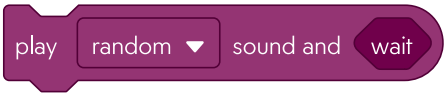
Matrix Scroll: Display a string of characters on the 8x8 light matrix.

BLOCK LIBRARY

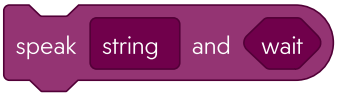
Matrix Pixel: Set a single matrix pixel X (0 to 7), Y (0 to 7), in a specific color. Coordinates start at (0, 0) in the bottom left corner.

Matrix Line: Draw a line or fill a rectangle from one X, Y coordinate pair to another.

Sounds



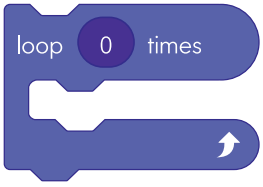
Sound: Play a specified sound from your programming device.



Speak: Make your programming device read a specified string.

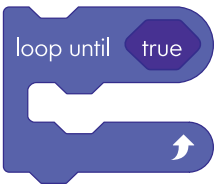
Controls

Delay: Delay execution of the next block for a duration of seconds.

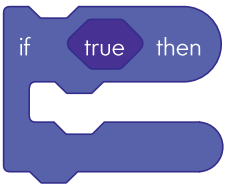


Loop: Repeat the blocks inside for a specified number of times.

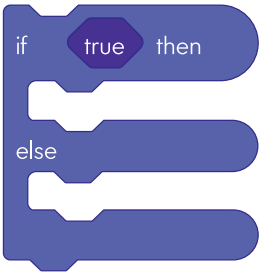
Loop Forever: Repeat the blocks inside for forever.



Loop Until: Repeat the blocks inside until a condition is met. Drag a comparator into the "true" field to create a condition.



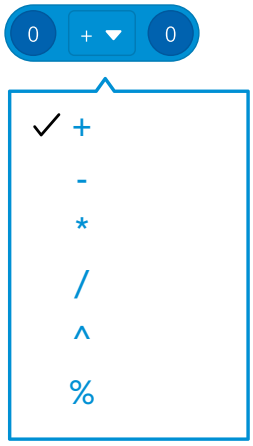
If Then: Run the "if" blocks if the condition is true.



If Then Else: Run the "if" blocks if the condition is true. If it's not true, run the "else" blocks.

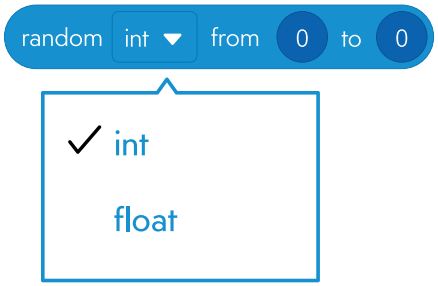
Exit Program: Stop all code and end the program.

Operators



Operator: Perform mathematical operations on two values.

Build String: Combine multiple values into a single string.



Random Int: Generate random integers.

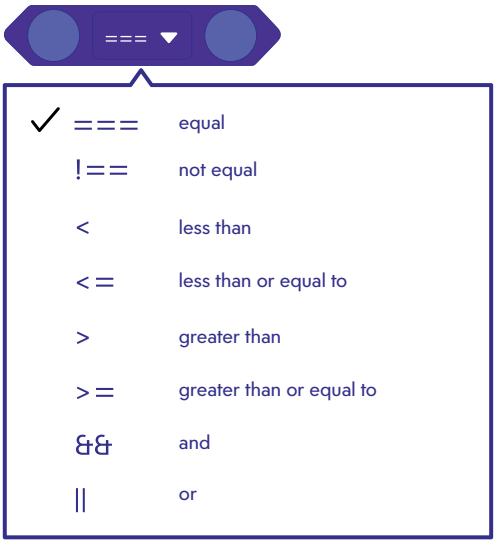
Random Float: Generate random floats (number with a decimal).

Color Channel: Find the red, green, or blue value (channel) of a specified color.

Color Mixer: Create a new color by changing the red, green, or blue values of a specified color.

Random Color: Select a random color when used with a color block.

Comparators



Comparator: Compare two values.

Sensors

Accelerometer - Total: Measure the change in force on all three axes.

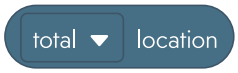
Orientation: Measure the orientation (pitch, roll, yaw) of BOLT+.

REFERENCE

BLOCK LIBRARY

Gyroscope: Measure the rate of spin in degrees per second.

Velocity: Measure the speed in centimeters per second.



Location: Measure the distance from where BOLT+ started at the beginning of the program.

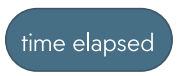
Distance: Measure total distance traveled, in centimeters.

Speed: Target speed value, from –255 to 255.

Heading: Target directional angle of BOLT+.

Main LED: Measure RGB color of the main LEDs, from 0 to 255 for each color.

Back LED - Blue Only: Measure brightness of back LED, from 0 to 255.

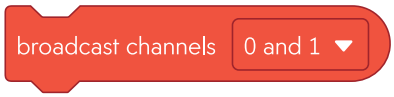


Time Elapsed: Measure how long the program has run, in seconds.

Ambient Light: Measure the light around BOLT+. 0 lux is complete darkness, 30,000–100,000 is direct sunlight.

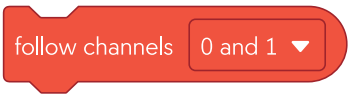
Last Message Received: Measure the light around BOLT+. 0 lux is complete darkness, 30,000–100,000 is direct sunlight.

Communications



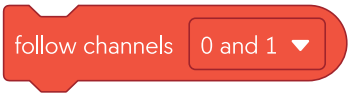
Broadcast Channels: Send out infrared messages on two specified channels.

Stop Broadcasting: Stop sending out infrared messages.

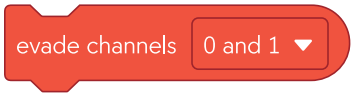


Follow: Follow another BOLT+ that is broadcasting on a specified channel.

Stop Following: Stop the following behavior.

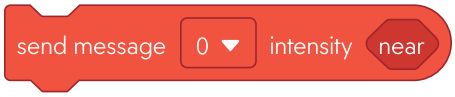


Evade: Evade another BOLT+ that is broadcasting on a specified channel.



Evade: Evade another BOLT+ that is broadcasting on a specified channel.

Stop Evading: Stop the evading behavior.



Send Message: Send a message on a specified infrared channel at a specified intensity.

Events

On Collision: Trigger the attached blocks if BOLT+ collides with an object.

On Freefall: Trigger the attached blocks if BOLT+ enters freefall (BOLT+ is falling).

On Landing: Trigger the attached blocks if BOLT+ lands after freefall.

On Gyro Max: Trigger the attached blocks if BOLT+ exceeds the maximum measurable velocity.

On Message Received: Trigger the attached blocks if an infrared message is received on a specified channel.

On Ambient Light: Trigger the attached blocks if the ambient light level is greater than or less than a specified value.

On Total Distance Is: Trigger the attached blocks if distance traveled (in cm) exceeds a specified value.

On Time Elapsed Is: Trigger the attached blocks if time elapsed (in seconds) is greater than a specified value.

On Button: Trigger the attached blocks if a software button is selected. Up to 3 buttons can be defined.

BLOCK LIBRARY

Variables

Variable: Create a variable with a specified type (string, number, Boolean, color) and value.

Functions

Function: Create a function with specified parameters and labels.

sphero®

BOLT+™

ISBN 978-1-7331447-9-7

55000>

9 781733 144797