Virginia Beach Public Library Sphero Code Club Manual

Contents

Sphero: Technical Specs

Overview

Sphero is a ball robot with some amazing capabilities. With a simple drag-and-drop interface through the Sphero edu app, you can code Sphero to move in any direction and control its speed and LED lights. The robot also has a full suite of sensors that you can utilize when coding, including a collision sensor, gyroscope, velocity sensor, accelerometer and more.

Battery Life

Sphero typically has a battery life of approximately 1 to 1.5 hours. With constant use (full motor functions), the battery life is typically 1 hour. With sporadic use, Sphero can sometimes stretch to 2 hours.

Sensors

Read/Write: These are sensors that can be recorded by the robot as well as changed by the robot in response to a condition.

- Heading
- Speed
- Color

Read Only: These are sensors that can only be recorded or read by the robot. They cannot be changed through the code in response to a condition.

- Location
- Velocity
- Orientation
- Accelerometer
- Gyroscope
- Vertical Acceleration

You can learn more about how each sensor works by double tapping its icon within the Sphero edu app.

Pairing Sphero with a Mobile Device

Pairing the Sphero with a mobile device is incredibly easy. Just make sure that the mobile device's Bluetooth is on and open the Sphero app you want to use. The Sphero SPRK+ will then automatically connect.

Goals of Programming with Sphero (and with teens)

There are 3 major goals of the Sphero Code Club. These are to demystify and spark an interest in coding and robotics, foster 21st Century skills, and strengthen the 40 Developmental Assets of Young Adults.

Demystify and Inspire

A major goal of the Code Club is to demystify and spark and interest in coding and robotics. The Sphero robot is a great low-barrier tool for introducing coding concepts. Because the coding interface uses drag & drop blocks of code, teens can gain proficiency coding the Sphero quickly, which builds confidence and lowers the intimidation factor. When it becomes accessible to them, teens have the comfort and confidence needed to become inspired by the possibilities of coding and robotics.

Foster 21st Century Skills

Another goal of the code club is to foster 21st Century skills that are necessary to thrive and be competitive in current job markets. The skills referenced here have been identified by the Institute of Museum and Library Services.

Critical Thinking & Problem Solving

Coding the Sphero robots allows teens to practice effective reasoning and systems thinking to solve challenges in new ways. By challenging them to examine the different coding blocks available and figure out how to complete a challenge or fix errors in their code, teens must critically examine the different pieces of code and how they come together to produce a desired result.

Creativity & Innovation

At the code club meetings, teens are given the freedom to be creative and are encouraged to let their curiosity lead them. Teens create their own activities and coding challenges, and therefore have the flexibility to develop new and innovative solutions. Teens will often work in pairs and small groups on different activities, giving them the chance to share their ideas and see each other's differing perspectives. They also are given the freedom to fail and see failure as a chance to learn and improve.

Communication & Collaboration

Since the code clubs are social gatherings, there are plenty of opportunities for teens to collaborate on activities. They gain experience compromising to achieve a shared goal and communicating through instructing, encouraging, convincing and questioning others.

Strengthen 40 Developmental Assets

The Sphero Code Club also strives to strengthen the building blocks of support that are needed for teens to grow into healthy and responsible adults. The club strengthens several of the 40 Developmental Assets of Young Adults, as defined by the Search Institute.

Boundaries, Expectations, & Constructive Use of Time

The club provides adult role models that the teens see on a regular basis. The staff facilitators and the teens bond over the activities during the program. Staff model positive behaviors such as showing an appreciation for learning and resilience when a piece of code fails to perform as intended. Teens also model positive peer behaviors for each other as they engage in group activities. The code club also provides a youth program to the teens that allows them to engage in creative activities.

Social Competencies & Positive Identity

The Sphero Code Club provides many opportunities for teens to improve their social competencies. During each meeting, teens plan and make decisions about what activities they wish to do. They forge new or stronger friendships with other teens and get opportunities to practice peaceful conflict resolution in a safe setting.

Finally, the goal of the club is also empower teens by giving them personal power and fostering strong self-esteem. By giving the teens control over the activities in the club and letting them choose their own learning directions, the club fosters a sense of agency, that they can control what happens to them. This is key to developing healthy self-esteem and a sense that they have a power over their own lives.

Notes on Teens

Social Engagement

Social engagement is the #1 most important thing to most teens, and for good reason. It's important to understand this when programming for this audience. Sometimes socializing will take precedence over the program itself, whether it's intended to or not!

To maintain a healthy balance between focus and social time, try to read the audience and break up activities that require heavy thinking with periods of low-key recreation. Teen programming shouldn't be about controlling teen behavior, nor should it be about letting the teens disregard the activities. It's about providing channels for socializing alongside the focus of the program.

Format: Teens as Deciders, Staff as Guides

Teens become more engaged when they have a say. Whenever possible, try to let the teens choose their own activities and approaches to solving challenges. They will be more motivated to participate and will often surprise you with their ingenuity.

That said, some teens don't work well with a completely blank slate. Some parameters may be required to help them get their footing and choose a path. For instance, in the pilot program of Sphero Code Club, we originally left choosing activities with the Sphero wide open, essentially telling the teens they could do whatever project they wanted. They struggled with that much ambiguity and needed some more boundaries to pick an idea. So we modified our approach, and instead gave them prompts like, "pick a board game and code Sphero to help you play it", or "Build an obstacle course for Sphero to navigate". With these simple prompts, the teens jumped into the challenges, often adding more parameters for themselves and further developing the challenges into more complex activities.

Code Club Program Models

This section provides a review of the recommended program models for Sphero-related programming. Sphero programs should have a duration of 1 to 2 hours. The Sphero Battery life is about 1 hour of constant usage, and this affects the duration of the program. There are three recommended approaches to offering programs with Sphero…

Weekly After-School Club (3 month duration)

These are part learning, part social groups that follow a loose structure of coding lesson, followed by group activities and social time. Attendance varies, as some teens will drop off and new teens will join the group from week to week. Modules are available and teens are encouraged to move through the first couple in order to get a basic understanding of Sphero. After that, the club can take it in their own direction, developing their own challenges and games and working on projects that interest them. Curriculum is very loose and moves at different paces, depending on the teens and the flow of the club.

6 Week Series

These are month-long or week-long series structured around a set of activities that focuses on certain learning objectives.

Single Session Activity

These are non-coding related or code-dabbling activities that can be completed in 1.5 – 2 hours. Coding is either not introduced at all or covers just the basics of controlling Sphero movement. Activities are usually game-centric and can be learned and enjoyed in a short period of time.

Why Not a Monthly Code Club?

Running a code club that meets monthly creates large gaps of time between coding sessions. In order for teens to retain and build on their knowledge, they need more frequent chances to play and experiment with coding. Basically, if they only meet monthly, there is too much time in between for them to forget what they did last session.

Weekly After-School Club

How to Start a Club

A weekly after school club should be arranged with the school contact, which is typically the School Media Specialist (SMS). Schedule an in-person meeting with the SMS to provide a description of what will happen at the club, what resources you are bringing, and what you need from the school.

When to Run the Club

Work with the SMS to pick a day when the club can consistently meet every week, such as every Monday or every Thursday. For High School, activity buses leave around 4:30 p.m., so the club will run from 2:30 – 4:30 p.m. For Middle school, activity buses leave around 5:30, so the club would run from 4:00 – 5:30 p.m. Note: The battery life of Sphero is about 1 hour of constant usage, so the Sphero robots will typically start to power down after 1 to 1.5 hours, depending on usage intensity. The school contact will help define the duration and time of year to run the program.

Location

Whenever possible, Sphero Code Clubs should meet in the school library. This provides the space needed to work comfortably with the Sphero robots.

Staffing

2 staff are needed to run the club. Ideally both staff members should commit to the duration of the club, but there needs to be at least one constant staff member who is always at club meetings. Constancy in staffing is critical for the health of the club. The best arrangement is for a school staff member and a library staff member to co-host the club sessions.

Registration & Marketing

Work with the SMS to advertise for the club at least a month prior to your first meeting. Having the school do registration for the club is a good way to get initial interest and set up reminders for the students when the club starts. It also helps you manage your capacity. It is recommended that the SMS handles registration and the initial reminder emails to the students for the first couple of meetings. Keep a sign-in sheet for each meeting, but registration can be dropped after you begin the club.

The Strategy behind the After-School Club

Sphero Code Club is built around the HAMAGO approach to learning. HAMAGO stands for “hanging out, messing around, geeking out”.

Hanging Out: Put the technology around the students and let the students hang out around the tech. They may interact with the technology or they may not right away, and that’s okay. Eventually they will approach the tech and start to play with it. In the case of Sphero, this usually means driving them around like racecars and playing with changing the colors of the LEDs.

Messing Around: The students start to experiment with the tech to see what else it can do. They will move past initial playing and start to explore the various capabilities of the tech. With Sphero, this

happens when the teens begin to code the Sphero Robots with basic commands. They will also experiment with different independent code blocks to see what the Sphero can do. They may also start to make simple programs that Sphero can execute.

Geeking Out: The students start to really dig into the tech with intent. They become more project driven and make the tech their own. In Sphero Code Club, this is when the teens start developing their own challenges to complete, work on specific projects, and start to modify existing code created by others.

Notes on Attendance, Adapting to your Audience, and Non-Coding Activities

Teens will come and go on a regular basis during the duration of the club, and you will be orienting new teens to the club on an almost weekly basis, especially in the beginning. At times it may not be possible to move the club through learning modules all together. This is fine. A good strategy is to have groups of teens at similar knowledge levels work on projects together. You may also get more teens than you have robots. Partnering teens together is a great way to support a little extra capacity without having a robot for each teen.

Group cohesion can be a challenge when you have some teens working independently or in several small groups. In order to keep through club close knit, engage the teens with challenges that don't require a high level of coding knowledge, or have room for teens to be as simple or complex as they like when coding. Also include non-coding activities with the Sphero that they can do together. Some ideas for activities are listed in Appendix 2.

How to Run the Club

The outline for Sphero Code Club is broken into 3 learning "modules". These modules are adapted in part from learning activities available through the SPRK Sphero Edu app and cover all the instruction of how to operate and code the Sphero robot. The modules can take multiple club meetings to complete. They don't need to, and in fact shouldn't, happen back-to-back. Feel free to interject different fun activities in between modules to keep the teens engaged, and be open to projects or ideas that the teens suggest. Because the Code Club often runs for several months, there is no pressure to quickly move through the modules.

6 Week Series

When to Run the Series

The series can either be done through weekly meetings any time of the year or as a week-long camp over the summer. Note: The battery life of Sphero is about 1 hour of constant usage, so the Sphero robots will typically start to power down after 1 to 1.5 hours, depending on usage intensity.

Location:

This program model is versatile and would work in a library location or as an extended outreach. A meeting room-size space is required to comfortably use the Sphero robots.

Staffing

One staff member is required. A second assisting staff member is preferable but not required, depending on staff experience and comfort level.

Registration & Marketing

Registration should be limited to the number of Sphero robots that are available. Registration should close once the series begins.

Notes on Attendance

The advantage to this program model is more control over attendance, meaning you can arrange an environment where there is a more constant group of participants. This will allow the teens to move through content together toward a specific end goal.

How to Run the Series

As opposed to an after-school club, which is more open ended, these 4 session series can target particular learning objectives and skills. The first two sessions should focus on learning to use Sphero and completing a coding module. Use one of the learning modules from Appendix 1.

The second two sessions should focus on completing a project or challenge using the coding techniques learned in the module. There are projects available within the coding modules. You can also find some ideas in Appendix 2, or let the participants create their own challenge to complete. Some flexibility may still be needed based on the participants. If you have a group that learns quickly, spend just one session on coding tutorials and the next three sessions on projects.

Stand Alone Activities

When to Run the Series

A stand alone activity, or "one-off" program, can be done at any time, even as a pop-up program. Note: The battery life of Sphero is about 1 hour of constant usage, so the Sphero robots will typically start to power down after 1 to 1.5 hours, depending on usage intensity.

Location:

Adequate space, such as a meeting room or other large open area, is required.

Staffing

One staff member is required. A second assisting staff member is beneficial but not required.

Registration & Marketing

Registration should be limited to the number of Sphero robots that are available. If the activity lends itself to participants taking turns, registration for the event may not be required. Marketing should focus more on the play aspect of the robots as opposed to coding.

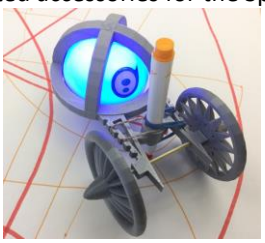
How to Run the Series

This type of activity or program should focus on exposure to the Sphero and building interest in future coding and robotics programs, not on coding itself. Referencing the *HAMAGO* strategy to learning, this type of program sits squarely in "hanging out" with Sphero robots. This program model should be between 1 to 2 hours in duration. Consider providing a fun game that participants can play using Sphero. You can find pre-built activities and teen-created challenges in Appendix 2 that work well for this program model. You can also allow the participants to dabble with the Sphero Sphero Edu app and coding under a guest profile (see Appendix 3 for details).

Technology Plan, Programmer Materials, and Participant Materials

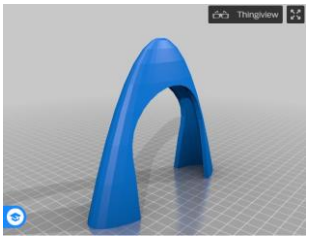
Below is an outline for the materials you need to effectively run the program. 3d printed items are beneficial but not always required.

- Sign in sheets
- Session outlines
- Pens, pencils, markers, paper, and other miscellaneous office supplies
- Several Sphero SPRKs for the instructors and students
- Several Samsung Galaxy Tab S2 tablets for the instructors and students
- Sphero Edu app on the Samsung Galaxy Tab S2 tablets
- Instructor email address for the Sphero Edu app
- 3-D printed accessories for the Sphero SPRKs:



http://www.thingiverse.com/thing:1266810

http://www.thingiverse.com/thing:1575004



- http://www.thingiverse.com/thing:1637250
- Multi-port chargers (for the Samsung Galaxy Tabs)
- Container of miscellaneous arts and crafts materials. These items will be used for competitive design. Ex: pipe cleaners, buttons, cups, straws, pieces of cardboard, popsicle sticks
- Session outlines
- Miscellaneous arts and crafts items

Appendix 1: Sphero Learning Modules

Block 1 Module

Description

Participants will learn how to program Sphero SPRK with block programming. The blocks represent Oval code that compile into a program. Students will use the Sphero Edus app to program the Sphero robot. Block 1 covers blocks of code that control actions, like movement and lighting, loop blocks, and basic operator blocks, such as addition or subtraction.

Session Goals

At the conclusion of the module, participants will have demonstrated that they've learned the programming language within the Sphero Edu app to program Sphero SPRK to perform various actions.

Session Learning Objectives

After attending this session, students will be able to:

1. Log into Student accounts within the SPRK Sphero Edu App
2. Connect a Sphero to the Galaxy Tab S2 devices
3. Navigate the interface of the SPRK Sphero Edu App

MATERIALS AND RESOURCES

Instructional Materials (Materials needed for the lesson such as specific handouts, PowerPoint files, worksheets, sample computer files, etc.):

1. Student materials:
   a. Sphero for each student
   b. Samsung Galaxy Tab S2 for each student
   c. Block 1 Student Handout (attached at the end of this module)
   d. Evaluation (for the end of the series)

Resources (Supplementary information and/or places where you found information for the lesson):

- Troubleshooting: https://sphero.freshdesk.com/support/solutions/folders/9000163900

PROGRAM OUTLINE

Session Preparation Checklist – 30 minutes

- Sphero charged and working
- Samsung Galaxy Tab S2 devices on, charged, and working. Student (restricted) profiles loaded
- Handout packets printed
- Sign-in sheet printed (if necessary)

After Session Tasks

- Secure and store all Spheros and Samsung Galaxy Tab S2 devices
- Clean up leftover handouts
- Retrieve evaluations and review (end of the series)
- Enter statistics into Program Statistics Data Form

I. Introduction

Time: 10 minutes

Content:

- Introductions
  - Instructors introduce themselves
  - Have students introduce themselves
- Housekeeping
  - Length of session
  - Encourage questions during session

II. Session Learning Objectives

By the end of this module, you will be able to:

1. Connect a Sphero to the Galaxy Tab S2 devices
2. Use the controls in the Sphero Edu app
3. Log into the Sphero Edu app on the Galaxy Tab S2 device
4. Understand the Sphero Edu interface.
5. Do Sphero Edu's introductory block 1 tutorial that's available in the Activities area of the app

Activity #1:

- Ask students if they've played with a Sphero or other robot. If they have played with a robot, what did they do with it?
- Ask students if they're familiar with programming. If they're familiar with programming, what kind of languages have they used in the past?

III. Log into the Sphero Edu app and connect Sphero

Time: 5 minutes

Content: Show students how to log into student accounts in the Sphero Edu app and pair Sphero with a tablet.

- Give each student the login information to a student account and have them log in.
- Sphero robots should pair automatically with the tablets once the SPRK Sphero Edu app is open. If this doesn't happen automatically, tap the "Connect Robot" button and select "SPRK+".
- Click on the steering wheel icon to make sure that the Sphero is being recognized by the app.

IV. Open the Sphero Edu App and Play with Sphero

Time: 5 to 30 minutes

Content: Ask students to click on the steering wheel in the top right of the screen.

Activity #1:

- Students will learn to aim and pilot Sphero and adjust Sphero's colors.
- Some groups may want to play with Sphero for a longer period of time, which is fine. If students wish to move on to coding, continue to the next activity.

V. Introduction to Coding in Sphero Edu

Students will do steps 3 and 4 of the Block 1: Intro tutorials.

Time: 30 – 45 minutes

Activity #1:

Provide the students with a brief demo of how to control the coding portion of the Sphero Edu app.

1. Have them open up a new program by tapping the "Programs" button at the bottom of the screen, then the "+" in the bottom right. Then tap "Create".
2. Demonstrate that you can tap on the tabs at the bottom of the screen to pull up different coding blocks. Scroll left & right to see all the blocks under each tab.
3. Double tapping a block tells you what it does.
4. Drag and drop code blocks on the blank canvas to add or move blocks of code.
5. Tap on the bubbles within a block to change numerical inputs. Ex: In the "Roll" block, you can change the duration, speed, and heading of the Sphero.

6. Tap and hold your finger on a block in order move it. Dragging a block up to the trash can at the top of the screen deletes the block.
7. Tapping a block on the canvas once brings up different options. You can highlight multiple blocks at once, copy and paste, or write a note to yourself about the code.
8. The Sphero Edu saves automatically.

Activity #2:

Students may follow along with the instructor and tap on the screens and buttons that are being discussed or they may move independently through the tutorials. Let them know that the video tutorials are available for review.

- Students will be given a handout that has all of the steps listed from steps 3-4 of the Blocks 1 tutorial. The instructor should use this handout to instruct.

Wrap Up

Time: 5 minutes
Content:

- Have students log out of the app and power down the tablets.
- Review what they learned today
- Ask whether anyone has questions
- Review resources for learning more – online resources, upcoming sessions, etc..
- Share additional resources for today's session and any upcoming sessions.
- Remind students to complete evaluations (end of series).
- Thanks students for coming!

Block 1 Tutorial Handout – Steps 3 & 4

This Module teaches how to use:

- Action Blocks
- Control Blocks (Delay and Loop)
- Operators (Add/Subtract/Multiply/Divide)
- Sensors (Heading- read & write)

Step 3 – Program a Square

1. Tap the **Program** button.
2. Start in the **Actions** category.
3. Drag the **Roll** block on to the canvas. Place it under the **On Start Program** block.
4. Change the properties of the **Roll** block.
    a. Duration - It's measured in seconds. **Set it to 2s.**
    b. Speed - It's measured on a scale of 0-255. **Set it to 50.**
    c. Heading - **Set it to 0 degrees.**
5. Add three more **Roll** blocks.
    a. Change the heading angle on the new blocks.
        i. Block two is **90 degrees.**
        ii. Block three is **180 degrees.**
        iii. Block four is **270 degrees**.
6. Place the Sphero on the ground. Tap the **Aim Ring** (looks like steering wheel) icon and orient the blue tail light so that it's facing you.
7. Press **Start**.
8. You will notice that the turns are not sharp. The next steps will fix this issue.
    a. Add a **Delay** after each **Roll** command.
    b. In the **Controls** category, select a **Delay** command and drop it after each **Roll** command. Set each **Delay** command to **1 second.**
9. Place the Sphero on the ground. Tap the **Aim Ring** (looks like steering wheel) icon and orient the blue tail light so that it's facing you.
10. Press **Start**.

Step 4 – Square with Loops

1. Delete all but one of the **Roll** and **Delay** block combo. Drag them into the trash that's at the top.
2. Open the **Controls** category. The **Controls** category allows you to change the flow of the program and add conditional logic to a program.
3. Drag the **Loop** block to the canvas. Drag it to the top so that it's underneath the **On Start Program** block.
4. Drag the **Roll** and **Delay** blocks inside of the **Loop** block.
5. Set the **Loop to 4**.
6. Go to the **Operator** category. You will add a **Heading**.
    a. Select **Add** and drag it into the **Heading** field of the **Roll** block.
7. Go to the **Sensors** category
    a. Select **Heading** and drag it to the left side of the **Operators** statement.
8. Tap on the **Bubble** to the right of the +sign. Set the value to **90 degrees**. It means that it will add 90 degrees to the heading angle.
9. Place the Sphero on the ground. Tap the **Aim Ring** icon and orient the blue tail light so that it's facing you.
10. Press **Start.**
11. Fix the starting angle.
    a. Add a block before the loops start that will add an angle of 270 degrees. 270+90=360 degrees. That's the same as 0 degrees and will allow the Sphero to move straight ahead.
        i. Go to the **Action** category and add a **Set Heading** block to the canvas. Drag it so that it's placed under the **On Start Program** block and before the **Loop** block.
12. Place the Sphero on the ground. Tap the **Aim Ring** icon and orient the blue tail light so that it's facing you.
13. Press **Start.**
14. Change Sphero's color while it is in motion.
    a. Go to the **Actions** category and drag a **Set Color** block into the loop. Drop it before the **Roll** command. Tap on the color block and pick a color.
    b. Go the **Actions** category and drag another color block onto the canvas. Drop it after the **Roll** command. Tap on the color block and choose an ending color.
15. Change Sphero's sound while it is in motion. *The sounds will play from the tablet, not the robot. Adjust the volume on your device.
    a. Go to the **Actions** category and drag a **Sound** block to the loop. Make sure that it's the first block in the loop. Select a sound that you'd like to play.
16. Place the Sphero on the ground. Tap the **Aim Ring** icon and orient the blue tail light so that it's facing you.
17. Press **Start.**

Block 2 Module

Description

Participants will continue to learn how to program Sphero SPRK with block programming. Block 2 covers more advanced usage of actions, loops, and operators while introducing sensors and variables. The projects in this module include creating a jack-o-lantern and a spinning top. Students will also "draw" a lollipop with light using Sphero and a long exposure camera app. Note: Working in groups of two is required for drawing the lollipop as two tablets are needed.

Session Goals

At the conclusion of the module, participants will have demonstrated that they've learned the programming language within the Sphero Edu app to program Sphero SPRK to perform various actions.

Module Learning Objectives

After attending this session, students will be able to:

1. Utilize coding to make Sphero perform more complex actions.
2. Use variables and sensors to control Sphero.

MATERIALS AND RESOURCES

Instructional Materials (Materials needed for the lesson such as specific handouts, PowerPoint files, worksheets, sample computer files, etc.):

2. Student materials:
   e. Sphero for each student
   f. Samsung Galaxy Tab S2 for each student
   g. Block 2 Student Handout (attached at the end of this module)
   h. Evaluation (for the end of the series)

Resources (Supplementary information and/or places where you found information for the lesson):

- Troubleshooting: https://sphero.freshdesk.com/support/solutions/folders/9000163900

PROGRAM OUTLINE

Session Preparation Checklist – 30 minutes

- Sphero charged and working
- Samsung Galaxy Tab S2 devices on, charged, and working. Student (restricted) profiles loaded
- Handout packets printed
- Sign-in sheet printed (if necessary)

After Session Tasks

- Secure and store all Spheros and Samsung Galaxy Tab S2 devices
- Clean up leftover handouts
- Retrieve evaluations and review (end of the series)
- Enter statistics into Program Statistics Data Form

I. Introduction

Time: 5 minutes
Content:

- Introduction
- Housekeeping
    - Review what was covered in Module 1.
    - Logging into accounts
    - Viewing Block 1 tutorial
    - Practicing with the Spheros
- Reminder– resources for further study

II. Session Learning Objectives

Time: 5 minutes
Content: By the end of this module, you will be able to:

1. Use variables to sensors to control Sphero.
2. Use a long exposure app to "draw" an image of light with Sphero.

III. Block 2 Tutorial of Coding with Sphero Edu

Time: 30 minutes to 1 hour+
Content: Students will move through the Block 2 tutorial attached at the end of this module.

Activity #1:
Students may follow along with the instructor and tap on the screens and buttons that are being discussed or they may move independently through the tutorials. Let them know that the video tutorials are available for review.

- Students will be given a handout that has all of the steps listed from the Block 2 Tutorial. The instructor should use this handout to instruct.

Wrap Up

Time: 5 minutes

Content:

- Have students log out of the app and power down the tablets.
- Review what they learned today
- Ask whether anyone has questions
- Review resources for learning more – online resources, upcoming sessions, etc..
- Share additional resources for today's session and any upcoming sessions.
- Remind students to complete evaluations (end of series).
- Thanks students for coming!

Block 2 Tutorial – Jack-O'-Lantern, Spinning Top, and Lollipop

Jack-O'-Lantern

Excited Personality

1. Drag the **Loop Forever** block to the canvas.
2. In the **Controls** category, drag a **Loop** block to nest inside the **Loop Forever** block
    a. Set the count to 1.
3. In the **Actions** category, drag the **Play Sound** block to the canvas. Nest it inside the **Loop** block
    a. Under Personality, select the Celebrate sound.
4. In the **Actions** category, drag the **Fade** block to the canvas. Nest it inside the **Loop** block, but under the **Play Sound** block.
    a. Select Yellow (first color)
    b. Select Blue (second color)
    c. Set time to 4 seconds
5. In the **Actions** category, drag the **Play Sound** block to the canvas. Nest it inside the **Loop** block, but under the **Fade** block.
    a. Under Personality, select Dreaming
6. In the **Actions** category, drag a **Fade** block to the canvas. Nest it inside the **Loop** block, but under the **Play Sound** Dreaming block.
    a. Select Green (first color)
    b. Select Purple (second color)
    c. Set time to 4 seconds

Scary Personality

7. In the **Controls** category, drag a **Loop** block below the first **Loop** block.
    a. Set the count to 1.
8. In the **Actions** category, drag a **Play Sound** block into the second **Loop** block**.**
    a. Under Mechanical, select the **Chainsaw** sound.
9. In the **Actions** category, drag a **Strobe** block to the canvas. Nest it inside the second **Loop**, but under the **Play Sound** Chainsaw block.
    a. Select Orange
    b. Set time to 0.1 seconds
    c. Set the count to 20
10. In the **Actions** category, drag the **Play Sound** block under the **Strobe** block.
    a. Under Personality, set sound to Laugh Evil
11. In the **Actions** category, drag the **Strobe** bock to the canvas. Nest it inside the second **Loop**, but under the **Play Sound** Laugh Evil block.
    a. Select Red
    b. Set time to .15 seconds
    c. Set the cycles to 15
12. Run the program. **You may borrow a cut out Pumpkin from staff.

Spinning Top

1. In the **Actions** category, drag the **Stabilization** block to the canvas.
    a. Set it to Off
2. In the **Controls** category, drag a **Loop Forever** block to the canvas. Put it under the **Stabilization** block.
3. In the **Controls** category, drag an **If Then Else** block. Nest it inside the **Loop Forever** block.
    a. In the **Comparators** category, drag a **Greater Than Or Equal** block to the canvas. Place it inside the gray True field.
    b. In the **Operators** category, drag a **Multiply** block to the canvas.
        i. Place it onto the zero to the left of the greater than sign
    c. In the **Sensors** category, drag a **Gyroscope** block to the canvas.
        i. Place it on the first zero in the operation
            1. Choose the Yaw axis
    d. Change the second zero in the operation to 100
    e. Change the zero on the right of the **Greater Than** block to 1.
4. In the **Operators** category, drag a **Set** block to the canvas. Place it in the If block. It should be placed right above the **Else** section.
    a. In the **Operators** category, drag a **Divide** block to the canvas. Place it on the numerator to the right of the Heading.
    b. In the **Sensors** category, drag a **Color** block to the canvas. Place it on top of the **Heading**.
        i. Set the color to Green
    c. In the **Sensors** category, drag a **Gyroscope** block to the canvas. Place it on top of the first zero in the division operation.
        i. Set it to Yaw
    d. Change the denominator zero to 8.6
5. In the **Operators** category, drag a **Set** block to the canvas. Place it inside the **Else** portion of the **If Then Else** block.
    a. In the **Sensors** category, drag a **Color** block to the canvas. Place it on top of the **Heading** portion of the Set block
    b. In the **Operators** category, drag a **Divide** block to the canvas. Place it on the zero in the **Set** block.
    c. In the **Operators** category, drag an **Abs** block to the canvas. Place it on top of the numerator.
    d. In the **Sensors** category, drag a **Gyroscope** block to the canvas. Place in on top of the numerator**.**
        i. Set it to Yaw
    e. Change the denominator zero to 8.6

Lollipop

1. In the **Actions** category, drag a **Set Speed** block to the canvas.
    a. Set to 60
2. In the **Controls** category, drag a **Delay** block to the canvas.
    a. Set to .25 seconds
3. In the **Controls** category, drag a **Loop** block to the canvas. Drag it underneath the other blocks.
    a. Set count to 15
4. In the **Actions** category, drag a Back **LED** block to the canvas. Nest it inside the Loop block.
    a. Set brightness to 255
5. In the **Actions** category, drag a **Set Color** block to the canvas. Nest it inside the Loop block, but underneath the **Back LED** block.
    a. Set it to Random.
6. In the **Actions** category, drag a Spin block to the canvas. Nest it in the Loop block, but underneath the Set Color block.
    a. Set it 90 degrees
    b. Leave the time at 0 seconds.
7. Go to **Variables** category
    a. Click **Add New**
        i. Name it **Time**
            1. Set it to .4
8. Drag the **Time** variable into the **Duration** field in the **Spin** block.
9. In the **Operators** category, drag a **Set** block to the canvas. Nest it inside the **Loop** block, but under the **Spin** block.
    a. Drag the **Multiply** block to the right of the equals sign.
    b. In the **Variables** category, drag the **Time** variable on top of the **Heading** field.
    c. At the **Time** variable to the left side of the multiplication operation.
    d. Change the field to the right to 1.05
10. In the **Actions** category, drag a **Set Color** block to the canvas. Nest it inside the **Loop**, but below the **Set** block.
11. In the **Controls** category, drag a **Delay** block to the canvas. Nest it inside the **Loop**, but below the **Set Color** block.
    a. Set to .05 seconds
12. In the **Actions** category, drag a **Stop** block to the canvas. Place it below the **Loop** block.
13. In the **Actions** category, drag a **BackLED** block to the canvas. Place it below the **Stop** block.
    a. Set to 0
14. In the **Actions** category, drag a **Set Color** block to the canvas. Place it below the **BackLED** block.
    a. Leave color white
15. In the **Controls** category, drag a **Delay** block to the canvas. Place it under the **Set Color** block.
    a. Set it to 1 second.
16. In the **Actions** category, drag a **Roll** block to the canvas. It should be the last block on the canvas.

a. Set delay to 1.5 seconds
b. Set speed to 45
c. Set the heading to 180 degrees

17. Open the long exposure app on the Galaxy Tab.
18. Run the app and program at the same time. You may need another person to assist you.

Block 3 Module

Description

Participants will continue to learn how to program Sphero SPRK with block programming. Block 3 focuses extensively using variables to create more advanced programs. The projects in this module include creating the game "hot potato" and making a digital "fortune teller".

Session Goals

At the conclusion of the module, participants will have demonstrated that they've learned the programming language within the Sphero Edu app to program Sphero SPRK to perform various actions.

Module Learning Objectives

After attending this session, students will be able to:

1. Utilize coding to make Sphero perform more complex actions.
2. Use variables and sensors to control Sphero.

MATERIALS AND RESOURCES

Instructional Materials (Materials needed for the lesson such as specific handouts, PowerPoint files, worksheets, sample computer files, etc.):

1. Student materials:
   i. Sphero for each student
   j. Samsung Galaxy Tab S2 for each student
   k. Block 2 Student Handout (attached at the end of this module)
   l. Evaluation (for the end of the series)

Resources (Supplementary information and/or places where you found information for the lesson):

- Troubleshooting: https://sphero.freshdesk.com/support/solutions/folders/9000163900

Block 3 Tutorial: Variables with Speeding Ticket, Hot Potato, and Fortune Teller

Speeding Ticket

1. Go to **Actions** and drag **Play Sound** block to the canvas.
    a. Under Mechanical, select Start Engine
2. In **Actions** and drag **Strobe** block to the canvas. Place it under the **Play Sound** block. Keep the color **white**.
    a. Set the period to 0.05s
    b. Set the count to 4
3. In **Actions**, drag the **Fade** block to the canvas. Place it under the **Stroke** block.
    a. Change the first color to black by dragging the brightness slider all the way to the left.
    b. Set the duration to 2s
    c. Set the second color to yellow
4. In the **Controls** category, drag the **Loop Until** block to the canvas. Place it under the **Fade** block.
    a. In the **Comparators** category, drag an **Equals Sign** and drag it onto the **True** field of the **Loop Until** block.
        i. In the **Sensors** category, drag the **Speed** sensor into the left side of the double equals sign.
        ii. Set the right side of the equals sign to 80
    b. In the **Actions** category, drag a **Set Speed** block to the canvas. Nest it inside the **Loop Until** block.
        i. Go to **Operators** and drag the **Addition** operator and drop on the **Set Speed** value
            1. Go to **Sensors** and drag a **Speed** sensor to the left side of the addition sign
            2. Make the value to the right of the addition sign 20
    c. In the **Controls** category, drag a **Delay** block to the canvas. Nest it inside the **Loop Until** block, but under the **Set Speed** block.
        i. Set it 1s
    d. In **Actions**, drag a **Play Sound** block to the canvas. Nest it inside of the **Loop Until** block, but above the **Set Speed** block.
        i. Go to the **Effects** category and select the Ding sound.
    e. In **Actions**, drag a **Strobe** block to the canvas. Nest it inside of the **Loop Until** block, but between the **Play Sound** and **Set Speed** blocks.
        i. Set the color to green.
        ii. Set the period to 0.05s
        iii. Set the count to 1
5. In **Actions**, drag a **Set Speed** block below the loop.
    a. Set it to 85
6. In **Actions**, drag a **Play Sound** block to the canvas and set it below the **Set Speed** block that was just added to the canvas.
    a. Go to Effects and choose Laser

7. In **Controls**, drag a **Loop** block to the canvas and set it below the **Play Sound** block.
    a. Set the count to 3
        i. In **Actions**, drag a 2 **Strobe** blocks to the canvas and nest it inside of the Loop block.
            1. 1st block: Make the color red, set the period to .1s, and set the count to 1
            2. 2nd block: Make the color blue, set the period to .1s, and set the count to 1
8. In **Controls**, drag a **Delay** block below the loop.
    a. Set the duration to .5s
9. In **Actions**, drag a **Set Color** block below the **Delay** block.
    a. Set it to yellow
10. In **Actions**, drag a **Play Sound** block below the **Set Color** block.
    a. In the Mechanical category, select Skid.
11. In **Actions**, drag a **Set Heading** block below the **Play Sound** block.
    a. Set it to 45 degrees.
12. In **Controls**, drag a **Delay** block below the **Set Heading** block.
    a. Set it to 1s
13. In **Actions**, drag a **Stop** block below the **Delay** block.

Hot Potato

1. In the **Controls** category, drag the **Loop Forever** block to the canvas.
2. Go to the **Variables** Category and click Add New.
    a. Name the first variable “expire”
        i. Set to 0
    b. Name another variable “toss”
        i. Set to 0
3. In the **Operators** category, drag a **Set** block inside of the **Loop Forever** block.
    a. In **Variables**, drag the **Toss** variable to the canvas and set it on top of the **Heading**. It will replace **Heading**.
4. In **Operators**, drag another **Set** block inside of the **Loop Forever** block, but below the first **Set** block.
    a. In **Variables**, drag the **Expire** variable to the canvas and set it on top of the **Heading**.
    b. In **Operators**, drag a **Random** operator to the canvas and place it to the right of the equals sign.
        i. Min Value: 2
        ii. Max Value: 10 (if playing with more than 3 players, increase the max value.)
    c. In **Controls**, drag a **Loop Until** block to the canvas and nest it inside of the **Loop Forever** block, but under the **Set** blocks.
        i. In **Comparators**, drag a **Greater Than** variable on top of **True**
            1. Go to **Variables** and drag **Toss** to the left of **Greater Than**
            2. Stay in **Variables** and drag **Expire** to the right of **Greater Than**
    d. In **Actions**, drag a **Set Color** block to the canvas and nest it inside the **Loop Until** block.
        i. Set the color to green.
    e. In **Controls**, drag an **If Then** block and nest it inside of the **Loop Until** block, but below the **Set Color** block.
        i. In **Comparators**, drag a **Greater Than** block and place it in the **True** field.
            1. In **Sensors**, drag an **Accelerometer** block into the left side of the **Greater Than** block.
                a. Choose the **Combined** axis.
            2. On the right side of the **Greater Than** block, set the value to 4.
        ii. In **Operators**, drag a **Set** block to the canvas and nest it in the **If Then** block.
            1. In **Operators**, drag an **Addition** block to the right of the equals sign.
                a. Drag the **Toss** variable block to the left of the equals sign and *also* on the left of the addition operation.
                b. Change the 0 to 1.
        iii. In **Actions**, drag a **Play Sound** block inside of the **If Then** block, but place it before the **Set** block.
            1. In Effects, choose Ding.
        iv. In **Actions**, drag a **Strobe** block inside of the **If Then** block, but place it below the **Play Sound** block.

1. Set the color to white
2. Set the period to .1s
3. Set the count to 1

5. In **Controls**, drag a **Delay** block to the canvas, but place it *after* the **Loop Until** block.
    a. Set it to 1s
6. In **Actions**, drag a **Play Sound** block to the canvas and place it after the **Delay** block
    a. In Personality, choose Sad
7. In **Controls**, drag a **Delay** block to the canvas, but place it after the **Play Sound** block.
    a. Set it to .5s
8. In **Actions**, drag **Set Color** block to the canvas and place it after the **Delay** block.
    a. Set it to red
9. In **Controls**, drag a **Loop** to the canvas and place it after the **Set Color** block.
    a. Set it to 5
        i. In **Actions**, drag 2 **Raw Motor** blocks to the canvas and nest it inside the **Loop**.
            1. 1st block: set the first and second motors to 3000 and the duration to .2s
            2. 2nd block: set the first and second motors to -3000 and the duration to .2s
10. In **Controls**, drag a **Delay** block to the canvas and set it below the previous **Loop**.
    a. Set it to 1s
11. In **Operators**, drag a Set block to the canvas and set it below the **Delay** block.
    a. In **Variables**, drag the **Toss** block to the left side of the equals sign. You're placing it on top of the Heading.
    b. Set the value to 0

Fortune Teller

1. In the **Controls** category, drag a **Loop Forever** block to the canvas. *We will nest all of our logic in this loop.
2. In **Actions**, drag a **Set Color** block to the canvas
    a. Set the color to blue
3. In **Controls**, add an **If Then** block to the canvas and place it below the **Set Color** block.
    a. In **Comparators**, drag a **Greater Than** block and place it into the **True** field.
        i. In **Sensors**, drag an **Accelerometer** sensor to the left side of the comparator.
            1. Select the Combined axis
            2. Change the 0 to 6
    b. In **Actions**, drag a **Play Sound** block to the canvas and nest it inside the **If Then** block.
        i. In Effects, choose Fairy
    c. In **Actions**, drag a **Strobe** block to the canvas and place it below the **Play Sound** block.
        i. Set the color to white
        ii. Set the period to .2s
        iii. Set the count to 12
4. In **Variables**, select **Add New**
    a. Name it “predictor”
        i. Set it to 0
5. In **Controls**, place a **Set** block on the canvas after the **Strobe** block.
    a. In **Variables**, drag the “predictor” variable to the left of the equals sign.
    b. In **Operators**, drag a **Random** block to the right of the equals sign of the **Set** block.
        i. Set the random bounds from 0 to 3
6. In **Controls**, drag an **If Then Else** block to the canvas and nest it inside of the existing **If Then** block. You’ll set it below the **Set** block.
7. Drag a second **If Then Else** block to the canvas, but place it *inside* the **Else** portion of the newly added **If Then Else** block.
8. In the first **If Then Else** block, drag a **Less Than or Equal** block into the first **True** field.
    a. Go to Variables, and drag the predictor block to the left side of the comparison
    b. Change the 0 to 1
        i. In **Actions**, drag a **Play Sound** block to the canvas, and nest it inside this block
            1. In Personality, choose Celebrate
        ii. In **Actions**, drag a **Set Color** block to the canvas, and nest it inside this block, but below the **Play Sound** block.
            1. Set color to green.
9. In **Controls**, drag a **Loop** block and place it under the **Set Color** block. Set it to 6.
        i. In Actions, place 2 **Raw Motor** blocks inside this loop.
            1. 1st block: left motor power is 2000, right motor power is 0, and the duration is .2s
            2. 2nd block: left motor power is 0, the right motor power is 2000, and the duration is .2s

10. In **Comparators**, and drag and **Less Than or Equal** block into the second **True** field.
    a. Go to Variables, and drag the Predictor variable to the left side of the comparison
    b. Change the 0 to the right of the comparison to 2.
11. In **Actions**, drag a **Play Sound** block onto the canvas and place it inside of the **If Then Else** block. You'll place it in the If portion.
    a. In **Personality**, choose **Dizzy**.
12. In **Actions**, drag a **Set Color** block onto the canvas and place it inside the **If Then Else** block, but below the **Play Sound** block.
13. In **Actions**, drag a **Spin** block onto the canvas and place it below the **Play Sound** block.
    a. Set the degrees to 1440
    b. Set the duration to 2s
14. In **Actions**, drag a **Play Sound** block to the canvas and place it in the **Else** portion of the block.
    a. In Personality, choose Sad.
15. In **Actions**, drag a **Set Color** block to the canvas and place it in the **Else** portion of the block, but under the **Play Sound** block.
    a. Set the color to red.
16. In **Controls**, add a **Loop** block and place it underneath the **Set Color** block.
    a. Set the loop to 5
17. In **Actions**, drag 2 **Spin** blocks and place them inside the previous **Loop** block.
    a. 1st block: set it to 180 degrees and .2s
    b. 2nd block: set it -180 degrees and .2s
18. In **Controls**, drag a **Delay** block and drag it inside of the original **If Then** block that we started with.
    a. Set it to 1s
19. In **Operators**, drag a **Set** block below the **Delay** block
    a. In **Variables**, drag the **Predictor** block on the left of the equals sign.
    b. Leave the 0.

Appendix 2: Stand Alone Activities & Teen-Created Challenges

Summary

Below are a list of some activities that can be done with Sphero to supplement the learning modules or as stand-alone activities. Each activity will indicate any supplies needed, a description of how to run the activity, and a coding/non-coding options for the activity. Suggested activity pairings for the learning modules are also included, though any activity can be used at any time to add variety and engage students.

Sphero robots and tablets are required for every activity, and therefore will not be mentioned under the "Supplies Needed" section.

Note: Some activities include rules and parameters, but the teens can gain valuable critical thinking and project management practice by puzzling out the activities themselves. The rules included in the activities below are what other teens have developed.

Sphero Race Track

Description

Students build a race track or obstacle course to navigate with the Sphero and compete to see who completes the track quickest or gets the closest to completing the track.

Supplies Needed

- Large bolt of cloth (building the race track on this makes it portable and easier to use over several sessions)
- Masking Tape or Painters Tape (to boundaries and barriers)
- Assorted objects to use as obstacles or ramps (optional)

Coding/Non-Coding Options

This activity can be done without coding. Have the teens pilot the Sphero directly to complete this challenge. You can also have the students work independently or in small groups to code their robot to navigate the challenge. Coding the track usually takes a decent chunk of time.

Notes:

Encourage the teens to build a wide course to make navigation a little easier on the Sphero.

Recommended Module Pairing:

Block 1. This challenge can be coded using simple action blocks and maybe some loops.

Pac Man

Description

Students create a Pac Man-style arena to recreate this classic arcade game with Sphero robots. One Sphero is yellow and is the "Pac Man", and the other Sphero robots are different-colored "ghosts".

Supplies Needed

- Large bolt of cloth (optional, only if you plan to use it for more than one session)
- Masking Tape or Painters Tape

Coding/Non-Coding Options

This is a non-coding activity, great for taking a break from coding or for pop up activity. However, the students should be empowered to incorporate coding if they want to.

Notes:

This activity typically works well with 1 Pac Man and 4 ghosts. This activity works well as a version of "tag", where the ghosts either have to tag Pac Man, or Pac Man has to tag all the ghosts. Setting the Spheros to a slow speed also makes the game more manageable for the teens and surprisingly more intense.

Recommended Module Pairing:

Anytime. This activity doesn't involve coding.

Twister

Description

Students work in small groups to code Sphero to be the "dial" for the game Twister. The challenge is to develop a way for Sphero to tell participants which limb to put on which color.

Supplies Needed

- Twister Game (2 copies allows for greater participation)
- Masking Tape or Painters Tape (to hold down the mat)

Coding/Non-Coding Options

This activity requires coding. This can be done by beginners as well as experienced coders. If the teens are stuck and looking for coding inspiration, they can try to use the fortune teller program in the Sphero Edu as a starting point.

Notes:

Teens often take different approaches to coding this challenge, providing a good compare & contrast discussion opportunity.

Recommended Module Pairing:

Block 2. This challenge can incorporate random operators and sensors, but it can also be done using simple action blocks. This is good for participants a different coding levels.

Pong

Description

Students pair up and re-enact the most classic videogame, Pong. Sphero is the ball and the students are the bouncers. Students lightly tap Sphero with their feet to send the robot back at their opponent. Points are scored when Sphero gets past a player.

Supplies Needed

- Masking tape or painters tape (optional, for boundaries)

Coding/Non-Coding Options

This is a non-coding activity. It is pre-existing code that can be downloaded through the Sphero Edu app. However, teens can be encouraged to modify the program to spice it up.

Notes:

Make sure to model lightly tapping the Sphero with your foot for the teens. Excitement can run high and sometimes Sphero make get kicked a little hard. Sphero is durable but this shouldn't be taken for granted.

Recommended Module Pairing:

Block 1. A basic understanding of the Sphero Edu app is need to pull down the code and run it, though if you walked the teens through it you could do this activity cold.

Battle Bots

Description

Teens modify Sphero chariots using materials on hand and then "battle" them. A player wins when they manage to knock the chariot off of their opponent's Sphero.

Supplies Needed

- A little cardboard
- Pipe cleaners
- Scissors
- Sphero Chariots
- Masking tape
- Pens & pencils
- K'Nex or LEGO's (optional)
- Other craft supplies that may be available.

Coding/Non-Coding Options

This is a non-coding activity, though coding could be incorporated if teens want to experiment with the collision sensors on the Sphero to control the robot's behavior. If coding is incorporated, then you will need "walls" or some barricades to create an arena.

Notes:

3d-printed objects might be fun to add to this challenge. Give ample time for teens to modify their chariots before battling.

Recommended Module Pairing:

Any. This activity doesn't require coding and can be easily introduced at any time or as a stand-alone activity. For incorporating coding into this challenge, introduce it after Block 2.

Non-Sphero centric Activities

Sometimes bringing other technology or engaging in non-Sphero related activities can break up the club sessions and provide a break for the students. Consider occasionally adding variety by bringing in other devices to showcase, such as a 3d Printer or 3d Pens, other robot kits like Cubelets, and downloading cooperative game apps such as *Space Team* on the tablets.