Getting started with Sphero Bolt

What is Sphero Bolt?

Sphero BOLT is a coding robot that lets kids learn coding through hands-on play and STEAM activities, but it can also be used at the university level. You can find out more about Sphero Bolt and its creators' mission on the official website. This robot comes with an app called Sphero EDU that allows us to program it with ease. If you want to program in Javascript your major source of documentation is here. **We recommend you first follow this guide to acquaint yourself with the robot before delving deeper into the official documentation.**

What comes with the robot?

If you choose to use Sphero Bolt, you will be given access to a box containing:

- 1 Sphero Bolt
- 1 Bluetooth USB Adapter
- 1 charging cable
- A small booklet

All equipment underwent thorough testing and verification prior to its availability to students. Kindly exercise caution and ensure the return of each item enclosed in the box in the same condition as it was borrowed.

What can Sphero Bolt do?

The Sphero Bolt has several features and functionalities designed to make it an engaging and educational robot. Here are some of its main functionalities and sensors:

1. **Programmable LED Matrix:** The Sphero Bolt has a built-in 8x8 programmable LED matrix that allows users to create and display various patterns, characters, and animations on the surface of the robot. This feature is particularly useful for teaching programming concepts and making the robot visually appealing.
2. **Sensors:**
    - **Gyroscope:** The gyroscope sensor detects the orientation and tilt of the robot, enabling it to respond to changes in movement.
    - **Accelerometer:** The accelerometer measures the acceleration of the robot, providing information about its speed and changes in motion.
    - **Motor Encoders:** These sensors help track the rotation and movement of the robot's motors, allowing for precise control and feedback.

3. **Bluetooth Connectivity:** The Sphero Bolt can connect to devices, such as smartphones or tablets, via Bluetooth. This allows users to control the robot, program it, and receive data from its sensors using compatible apps.
4. **Programming Capabilities:** Sphero provides a visual programming interface that allows users to program the Bolt using a block-based coding language. This makes it suitable for educational purposes, enabling students to learn programming and robotics concepts in a fun and interactive way.
5. **Durable Design:** The Sphero Bolt is designed to be durable and capable of rolling on various surfaces. Its rugged design makes it suitable for use in educational settings and ensures that it can withstand typical wear and tear.
6. **Sounds and Speech:** There are instructions to produce speech and sounds. However, the robot does not produce any sound, instead the speech or sounds will be produced by the device you are using to program and stream the code to Sphero Bolt.

How can I program Sphero Bolt?

At the time of writing this guide, Sphero Bolt could be programmed in the following languages:

- "Draw" languages (draw on the Sphero EDU app to program the robot - not tested by the professors)
- "Block" languages (draw on the Sphero EDU app to program the robot - not tested)
- **Javascript (on the Sphero EDU App - Recommended)**

There are "Getting Started guides" and more advanced code for javascript available here. The Javascript documentation on Sphero EDU is quite comprehensive and complete, **we recommend you read the whole official Sphero EDU Javascript Wiki**. However, you can start by completing this guide and then dive deeper into the official documentation.

Getting started: Turn on and run your first program in Javascript

Installing Sphero EDU App:

Programming the robot is possible through JavaScript using the Sphero EDU app. Keep in mind that the Sphero Bolt's limited memory prevents it from running code directly. Rather than sending code to Sphero's disk for execution, we send commands to the robot. The app already contains all the necessary commands,

variables, and constants, and it connects directly to Sphero to send these commands on our behalf. For optimal results, we suggest using the app.

1. Download the app from here.
    a. The app is available for **iOS**, **macOS**, **Android** and **Chrome OS**, **Fire OS**, and **Windows**.
    b. Note: We recommend you create an account on the Sphero EDU app. This will allow you to see your javascript scripts across multiple devices (meaning, you can program on a computer and then run the code using an smartphone for example)

Create your first javascript project:

1. Open the Sphero EDU App and log in.
2. On the menu on the left, click on “My Programs” under “Programs”.
3. Click “Create Program”
4. Choose “Text” for the Program Type and choose Sphero BOLT as the compatible robot, then click create
5. A white canvas will open where you can write your program. Copy and paste the code from here.
6. Now click “Start” and pair the robot.

Pairing the robot:

1. To pair the robot to the device you’re programming on, you need to have bluetooth on the device (i.e. if you are programming on a laptop the laptop needs to have bluetooth). A USB bluetooth adapter is provided in case the device your user does not have bluetooth.
2. After turning the bluetooth on, check the list of available Sphero Bolts and select your robot.
3. Your robot and device should be paired by this point.
4. Click “Start” again on the Sphero EDU App and the code will be streamed over to Sphero Bolt via bluetooth.

Let’s go over what the code does:

Here’s a print of the code:

```
// State Machine States ------------------------------------------------------------------
const STATES = {GREETING:0, MOVING_ALONG:1, COLLIDED_ANGRY:3, END:4};
var currentState = STATES.GREETING; // represents the current state the robot is on

//
// Setting the onCollision event that is triggered if the robot collides with something
async function onCollision() {
    currentState = STATES.COLLIDED_ANGRY;

}

registerEvent(EventType.onCollision, onCollision);


// Main Loop ------------------------------------------------------------------------------
async function startProgram() {

    for(var i; i < 60; i++) {
        switch(currentState) {

        case STATES.GREETING:
            // Greet users
            await speak("hello I'm Sphero BOLT and I am a very good candidate for yout project.");

            // Show off spin ability
            await speak("i can do a lot of things, like spinning");
            await spin(360, 1);

            // Show off sound playing abilities
            await speak("or making cute sounds");
            await Sound.Animal.play(true);

            // Show off led capabilities
            await speak("And I have leds that you can change too");
            await setMainLed(getRandomColor());
            await fade({ r: 0, g: 255, b: 0 }, { r: 0, g: 0, b: 255 }, 3.0);

            // Show off moving, change state
            await speak("But my best feature is that I can move freely, let me move start moving for you");
            currentState = STATES.MOVING_ALONG;

        break;


        case STATES.MOVING_ALONG:
            // Set LED to white
            setMainLed({ r: 255, g: 255, b: 255 });

            // Move for a while
            setSpeed(100);

        break;


        case STATES.COLLIDED_ANGRY:
            stopRoll();
            setMainLed({ r: 255, g: 0, b: 0 });
            await speak("Collision", false);
            setHeading((getHeading() + Math.floor(Math.random() * 180)));
            await delay(0.5);
            currentState = STATES.MOVING_ALONG;
        break;

```

This code is meant to show you the essential features of the Sphero Bolt and guide you in using design patterns like State Machines to organize your code. It's crucial to keep your code clean and organized, especially since you can only program in a single file using the Sphero EDU app. Comments, delimiters, and good design patterns will be very helpful for your project.

We used a state machine in our code, which is explained in this video using Unity and C#. The design pattern is explained in a simple way, and the code is not much different.

Our state machine is quite simple. We have four states: GREETING, MOVING_ALONG, COLLIDED_ANGRY, and END (defined in line 15). We also defined a variable, currentState (line 16), to keep track of the robot's current state. The goal is to have the following behavior:

- The robot starts by greeting and introducing itself (currentState set to GREETING).
- After the greeting, the robot moves along (MOVING_ALONG state).
- If a collision occurs while moving, the robot changes to the COLLIDED_ANGRY state.
- The program runs for a minute, and then the robot stops following commands (END state).

Collisions can be detected, and we created a function called onCollision() (line 20) to handle what the robot should do in case of a collision. We tell the robot to call this function if a collision occurs by registering this event on line 25, specifying that if a collision happens, the onCollision function should be called.

The main loop of our program starts from line 31 to line 86. In the GREETING state, we use the speak function to activate text-to-speech on your programming device (e.g., PC or smartphone). We also produce random animal sounds and change the robot's LEDs. These functions are documented on the Sphero EDU JavaScript wiki.

In the MOVING_ALONG state, we change the LED panel, set the robot's speed to 100 for forward movement, and handle collisions by changing the LEDs to red and adjusting the robot's direction using the setHeading function.

These are the main components of our script.

Further documentation and guides:

The Javascript documentation on Sphero EDU is quite comprehensive and complete, **we recommend you read the whole official Sphero EDU Javascript Wiki**.

FAQs:

The robot does not run my programs:

It may be with low battery, it only lasts for two hours on normal usage. Please recharge it and try again.