The notes below document the order in which I introduce material, going from a single 9V battery powering an LED to operating LEDs or a motor based on input to a Circuit Playground I/O pad.

Under the "Programs" folder, there are .js and .uf2 versions of each program. You can drag and drop the .uf2 files into the workspace at makecode.adafruit.com to open them. You can also copy/paste the .js (JavaScript) text into the workspace if you select "JavaScript" at the top of the workspace.

Under the "Schematics" folder are images illustrating how I wire up each circuit. The fzz folder contains the files generated by Fritzing, the software that I used to create the circuit diagrams.

Lesson 1: How does electricity flow?

* Use 9V battery to directly power an LED (Schematic 01)
  + note that the LED only works one way - electricity cannot flow through a diode in both directions.
* Draw the circuit, explaining how the electricity flows from one side of the battery to the other
* Introduce the breadboard and rewire the LED circuit to use the breadboard, still using the 9V battery as the power source (Schematic 02)
* Draw the button with its four pins on the whiteboard, explaining how the button bridges the gap between the two pairs of pins when depressed.
* Add the button to the circuit, so that it interrupts the flow from the battery to the LED unless it’s pressed (Schematic 03)

Lesson 2: Introduce the Circuit Playground

* Replace the battery from previous lesson with the Circuit Playground, using a 3.3V pad and GND in place of the battery’s terminals. Use this to show that I/O pins work just like the battery, and explain that the difference is that *some* of the pads can be programmed to change whether they are charged. (Schematic 04)
* (Program 01. Blink) Create a simple program that turns the LED on and off every half second. Add blinking for an onboard LED, as well. This is a useful thing to do – you can put LED or speaker output in a program to test which path the software is going down, since it is not dependent on wiring. You can remove the button for this program, so that it doesn't have to be held down to see the blinking. And you can remove the resistor, because the programmable pads don't put out enough voltage to blow up the LED. (Schematic 05)
* (Program 02. ButtonBlink) Update Blink so that it updates based on a button press from the Circuit Playground. This program can be updated to trigger based on the switch, shaking, light/dark, the microphone, etc.

Lesson 3: Introduce sensors

* (Program 03. MotorOscillate) Wire up a continuous servo and cause it to spin back and forth. (Schematic 06)
* (Program 04. MotorButtons) update the previous program to use the onboard buttons instead, or any other sensor input. An example with noise is (05. MotorClap) but the update from the previous program is very intuitive with MakeCode. (Schematic 06 - unchanged from Program 03)
* (Program 06. MotorButtons-Offboard) Introduce the ability to handle incoming signals, for example by reading the state of an offboard button that is wired to a pad.
* (Program 07. ProximityDetector) Introduce an offboard sensor with a program that lights up the onboard LEDs to show how close an object is to a proximity detector. (Schematic 08)