Circuit Playground Examples

Girls Who Code

BlackRock July 20th, 2017

On the following pages you will find a series of examples and exercises for use during the Girls Who Code sessions at BlackRock. These examples are meant for use with the Circuit Playground which is a type of Arduino.

Before you begin...

1. Make sure your Arduino environment has the **board** set to “Adafruit Circuit Playground.” You can set it under the menu: Tools->Board-> Adafruit Circuit Playground.
2. Make sure your Arduino environment has the **port** set. You can set it under the menu: Tools→Port. The exact port name will look something like “Circuit Playground.” It depends on what kind of computer you have and which USB port you plugged it into.

General Instructions

1. Open up each Arduino program (for example b\_led\_2) and upload it to the Circuit Playground.
2. Look at both the code and the behavior of the program.
3. Are you able to understand why the program behaves as it does? Ask one of the volunteers for assistance if anything is unclear.
4. Save the program under a new name (for example b\_led\_2\_test1) and then try out the suggested exercises.
5. Do the same for all of the exercises.
6. You are welcome to try out variations on all of the programs.

**We encourage experimentation because it will help you understand why things work as they do!**

1. skeleton\_only
   1. This program contains the smallest amount of stuff required by a Circuit Playground program. Things to notice include the following:
      1. Comments start with “//” Comments are ignored by the computer. They are for you, the programmer.
      2. You must include “#include <Adafruit\_CircuitPlayground.h>” near the top of your program.
      3. There are two required functions:
         1. setup()
         2. loop()
2. a\_led\_1
   1. This program shows how to light a single Red/Green/Blue LED. These are usually referred to as RGB LED's. Adafruit refers to them as “NeoPixels.” Try the following:
      1. Light different LED's
      2. Light more than one LED
      3. Try different color combinations. For example, how do you make Yellow?
3. b\_led\_2
   1. This program shows the use of the delay() function. The delay() function pauses your program. Try the following:
      1. Vary the delays
      2. What happens if the delays are very small?
4. c\_variable\_delays
   1. This program shows how to assign to variables. Variables are named “thingies” where values, like integers, can be stored. Try the following:
      1. What happens if you change the value of the delay\_time variable?
      2. Can you change the name of the delay\_time variable?
      3. Create a second variable. You can use any name you'd like. The purpose is to set up 2 different delays based on two different variables.
      4. What happens if the delays are very small?
      5. What happens if you move the variable assignments to the end of the setup() function?
5. d\_loop\_with\_blink
   1. This program shows how to use the loop() function. The loop() function repeatedly runs all of the code within its body. Try the following:
      1. What do you have to do to change how fast the LED blinks?
      2. Can you blink more than one LED at a time?
      3. Can you make the “on” and “off” times different?
      4. Replace the fixed value delays with delays based on variables. (Look at c\_variable\_delays for a reminder of how to use variables.)
6. e\_loop\_with\_button
   1. This program shows how your program can behave DIFFERENTLY as it runs by making use of the if statement to evaluate conditions. This program also demonstrates how to determine if a button is being pressed on the CircuitPlayground. Try the following:
      1. Light an LED only when the button is NOT pressed.
      2. Light one LED when the button is pressed; light a different LED when the button is NOT pressed.
      3. Can you do more than one thing when the button is pressed?
7. f\_led\_chaser
   1. This program demonstrates a technique for variable assignment and how it can be used to count up to a fixed value. The demo uses the variable to select which LED to light. Try the following: (For many of these experiments you can look at e\_loop\_with\_button for guidance on how to determine if a button is being pushed and how to make use of the if statement):
      1. Make the LED light in the opposite direction
      2. Make the LED blink at a different speed when a button (either left or right) is pressed. Hint, use a button to set the value of a variable which is used by delay().
      3. Only light the LED when a button is pressed.
      4. Change the color of the LED when a button is pressed
      5. Have another LED also circle around so it appears one LED is following another
         1. Light one LED when the left button is pushed; light the other LED when the right button is pushed
8. g\_programmer\_makes\_loop
   1. This program is based on f\_led\_chaser. It demonstrates a technique for creating your own loops. Look closely at the “while” statement in the setup() function. Try the following:
      1. Make the “startup” LED blink 7 times before the loop() function starts.
      2. Using the technique in the setup() function, add code to the loop() function so one LED blinks 5 times EVERY time the code in the loop() function runs. KEEP THE EXISTING LED code in loop() as it is! The idea is one of the LED's blinks once and a different LED blinks 5 times.
9. h\_serial\_output
   1. This program demonstrates serial output which is useful for understanding what your program is doing “under the covers.” The key new statements are:
      1. Serial.begin() used once in the setup() function
      2. Serial.print() used to print messages and the values of variables
      3. Serial.println() is like Serial.print() but also makes sure the next printed messages occur on a new line
   2. Try the following:
      1. Change the message the program prints.
      2. Go back to some of your old programs and add serial output to them. For example, print the led number in the f\_led\_chaser program.
      3. In e\_loop\_with\_button, modify the program to print a message every time the button is pressed. Did it work the way you expected?!
10. i\_light\_sensor\_demo
    1. This program does not introduce any new programming techniques like the use of if or while. Instead it demonstrates a capability of the CircuitPlayground – the light sensor. It does make use of A LOT of what you have already learned, including delays, variable declarations and assignments and serial output using the Serial Monitor.
    2. Try the following:
       1. Light an LED but ONLY when you hold your hand over the board.
       2. Change the brightness of the LED (or light more than one LED) as it gets “darker” in the room.
11. j\_sound\_sensor\_demo
    1. This program does not introduce any new programming techniques like if or while. Instead it demonstrates a capability of the CircuitPlayground – the sound sensor.
    2. Try the following:
       1. Look at the serial output from this program. What the heck is going on? Change the program as necessary. It might help if you change the delay between readings.
12. k\_temp\_sensor\_demo
    1. This program demonstrates a capability of the CircuitPlayground – the temperature sensor.
    2. Try the following:
       1. Light an LED when the temperature rises because you put your hand on the sensor. Don't worry there's NO danger.
       2. Print a serial message saying “HOT!” when you put your hand on the sensor.
13. l\_speaker\_demo
    1. This program demonstrates a capability of the CircuitPlayground - using the speaker to play a tone.
    2. Try the following:
       1. Make the program play a different tone when the other button is pushed.
       2. Make the program play the a tone half a second AFTER the button is pushed.
       3. Make the program play a different tone when BOTH buttons are pushed. - This might be hard!
14. m\_acceleration\_sensor\_demo
    1. This program demonstrates a capability of the CircuitPlayground - using the “accelerometer” to measure the PULL on the board. Using the serial monitor as you did in some of the earlier programs, you can see how the sensor “feels the pull” on the board.
    2. Try the following:
       1. Make the program play a tone when the CircuitPlayground board is upside down!
       2. Make the program light different LED's depending on how the board is held.
15. n\_cap\_sensor\_demo
    1. This program demonstrates a capability of the CircuitPlayground - using the “cap sensor.” The cap sensor can be thought of as a TOUCH sensor. Using the serial monitor as you did in some of the earlier programs, you can see how the sensor “knows” when you are touching the board.
    2. Try the following:
       1. Make the program play a tone when copper ring number 6 is touched.
       2. Connect an alligator cable to copper ring number 6. Make your program light an LED when you touch the OTHER end of the alligator cable.

Sample Projects

1. Make a musical instrument. Use one or more of the sensors, for example the light sensor to play different tones.
2. Create some kind of clock. How can you use what is on the Circuit Playground to indicate the time? Be creative!
3. Create a combination lock where someone has to interact with the board and when the “combination” is entered, the board plays a tone.
4. Make an “orientation” sensor so when you hold the board up, you light the LED at the bottom of the board.
5. Two person project – can blinking LED's on one Circuit Playground cause some kind of behavior on another Circuit Playground?