Arduino Accessibility 'Around Me' Buttons & Switch Control Connected to Your Phone

by jazzang

Our Goal

Blind and low-vision travelers use apps such as Microsoft Soundscape and Blindsquare to build a richer awareness of their surroundings, thus becoming more confident and empowered to get around. Unlike step-by-step navigation apps such as Google Maps, Soundscape uses 3D audio cues to enrich ambient awareness and provide a new way to relate to the environment. These apps make it easy to control their features by re-purposing the regular media controls (play/pause, rewind and forward) to give important information on-demand while commuting. Though these can be invoked using media controls on headphones, it may be not as hassle free to stop using one's cane and tap these controls while on the move.

With this, we were inspired to create an 'Around Me' sets of buttons that enable blind and low-vision travelers to conveniently access the functions of these app and their phone with the click of these buttons. These buttons could also potentially be clamped to the cane.

This project is jointly co-created by Jazz Ang, Ana Liu & Venkatesh Potluri.

Overview of Accessibility Buttons

In these set of instructables, you would be creating the following six buttons:

Play/Pause (of media player and videos)

When Microsoft Soundscape is on the background, this button will start (or pause) the synthesised binaural audio, such as the heartbeating audio cues from Soundscape, and audio notifications of where the user is. The play/pause works by activating with the media player running on the background.

Mute Volume

This button mute (and unmute) the volume.

Next

When Soundscape is on the background, it describes points of interest in front of you, for example when walking down the street. On music/video player, the button plays the next song or video.

Previous

Likewise, this button plays past places of interest on Soundscape and plays the previous song or video.

Home

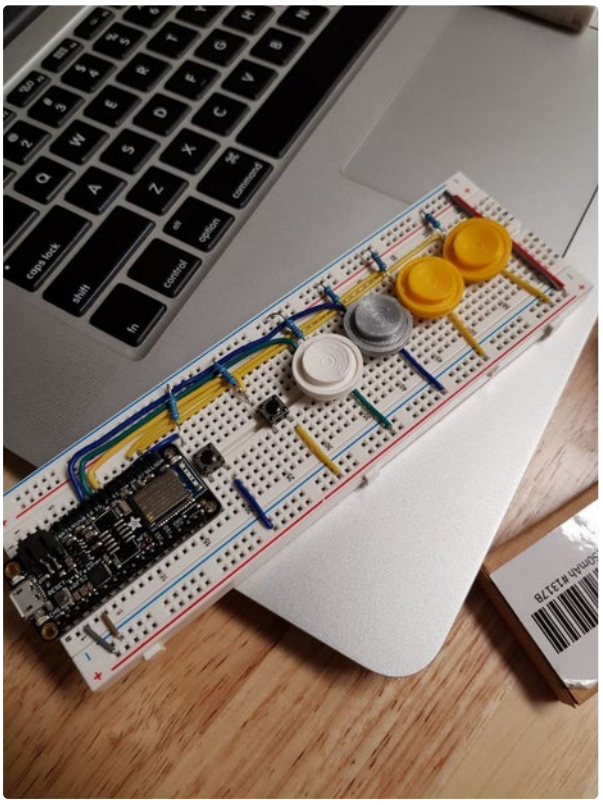
This button acts as the home button on your phone, it directs users back to the main home screen.

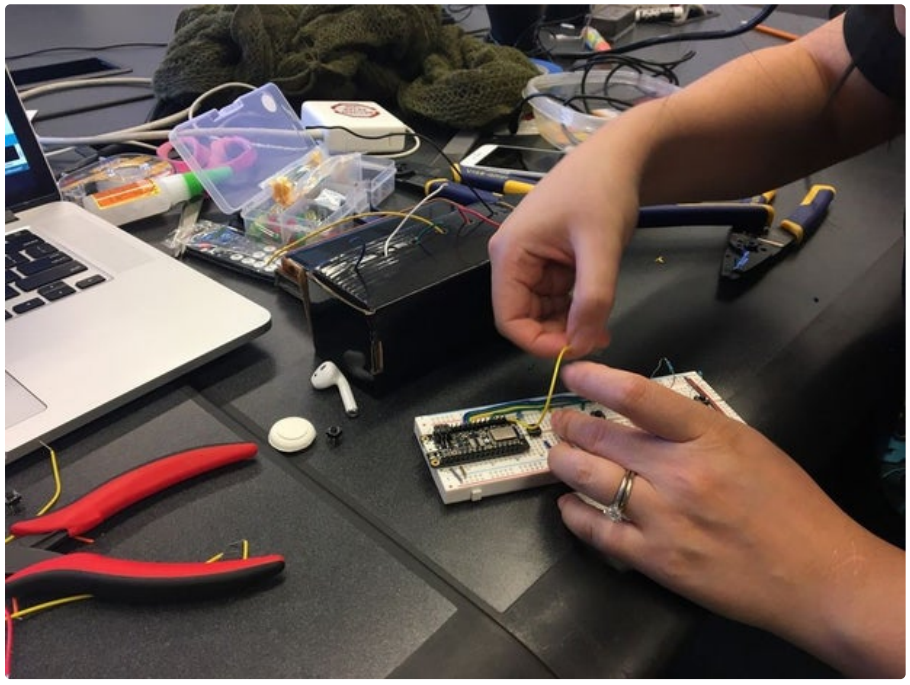
Switch Control

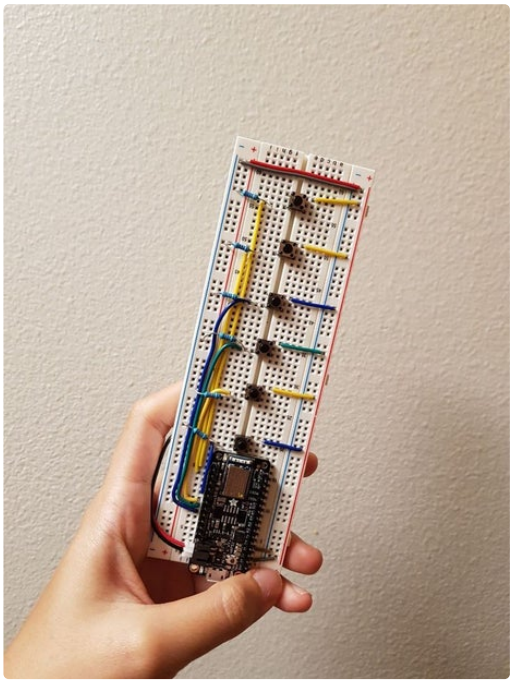
This button will be connected to the iOS switch control options that could be configured to serve as the button to select on the switch control mode. This will be set up later.

Supplies:

Adafruit Feather NRF52 Bluefruit Adafruit / Amazon
Full-sized breadboard Adafruit / Amazon
Wires
6 x Tactile button switches Adafruit
10K ohm resistors Adafruit

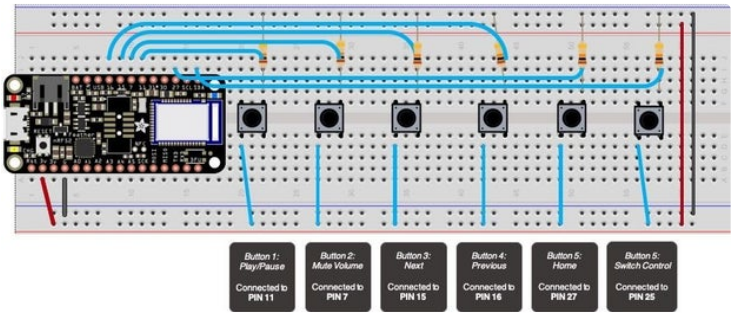






Step 1: Circuit Wiring

The circuit is simple to set up with basically 6 tactile switches connected to each individual pins. In this setup, we are using the Adafruit Feather nRF52 Bluefruit LE and the six tactile switches is connected to Pins 11, 7, 15, 16, 27 and 25.



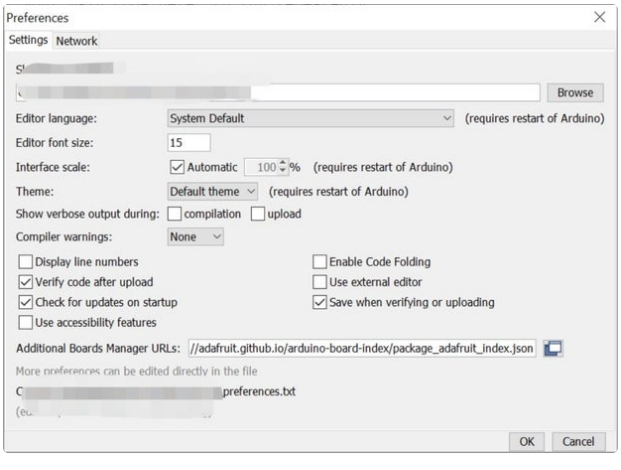
Step 2: Install Board and library

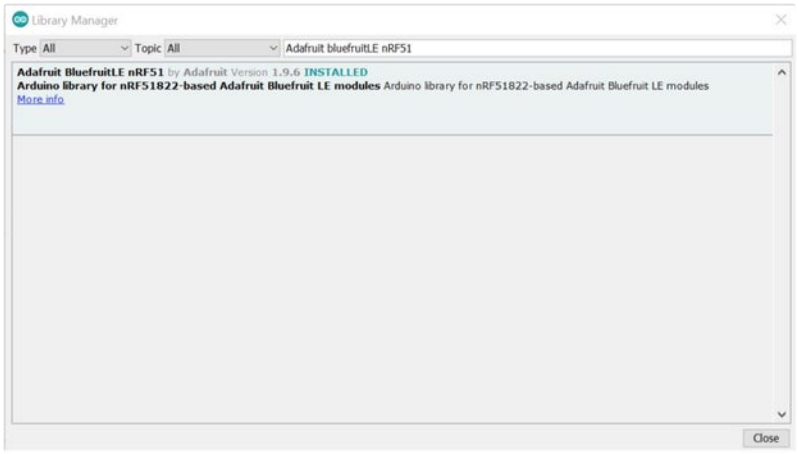
Open the Arduino Software (IDE) and Preferences. Then type “https://adafruit.github.io/arduino-board-index/package_adafruit_index.json” in the Additional Boards Manger URL, and click “Okay”.
Open Tools -> Board… -> Board Manager.
Type Adafruit nRF52 in the search box. Click on Adafruit nRF52 and click “Install”.
Open Tools -> Manage Libriaries…
Type Adafruit BluefruitLE nRF51 in the search box. Click on Adafruit BluefruitLE nRF51 and click “Install”.
Create hid_keyboard file
Open File -> Examples -> Adafruit Bluefruit nRF52 Libraries -> Peripheral -> hid_keyboard
Here are the codes:

```
#include <bluefruit.h>

#define CONTROL_PLAY_PAUSE      0x00CD
#define CONTROL_SCAN_NEXT       0x00B5
#define CONTROL_SCAN_PREVIOUS   0x00B6
#define CONTROL_MUTE            0x00E2
#define AC_FORWARD              0x0225
#define AC_BACK                 0x0224
#define CONSUMER_BROWSER_HOME   0x0223

BLEDis bledis;
BLEHidAdafruit blehid;

bool hasKeyPressed = false;
//connect pins in the board
int playpauseButtonPin = 11;
int muteButtonPin = 7;
int nextButtonPin = 15;
int backButtonPin = 16;
int homeButtonPin = 27;
int switchControlButtonPin = 25;

void setup()
{
  pinMode(playpauseButtonPin, INPUT);
  pinMode(muteButtonPin, INPUT);
  pinMode(nextButtonPin, INPUT);
  pinMode(backButtonPin, INPUT);
  pinMode(homeButtonPin, INPUT);
  pinMode(switchControlButtonPin, INPUT);

  Serial.begin(115200);
  while ( !Serial ) delay(10);   // for nrf52840 with native usb

  Bluefruit.begin();
  Bluefruit.setTxPower(4);    // Check bluefruit.h for supported values
  Bluefruit.setName("TESTArroundMeBluefruit52");

  // Configure and Start Device Information Service
  bledis.setManufacturer("Adafruit Industries");
  bledis.setModel("Bluefruit Feather 52");
  bledis.begin();

  /* Start BLE HID
   * Note: Apple requires BLE device must have min connection interval >= 20m
   * ( The smaller the connection interval the faster we could send data).
   * However for HID and MIDI device, Apple could accept min connection interval
   * up to 11.25 ms. Therefore BLEHidAdafruit::begin() will try to set the min and max
   * connection interval to 11.25  ms and 15 ms respectively for best performance.
   */
  blehid.begin();

  /* Set connection interval (min, max) to your perferred value.
   * Note: It is already set by BLEHidAdafruit::begin() to 11.25ms - 15ms
   * min = 9*1.25=11.25 ms, max = 12*1.25= 15 ms
   */
  /* Bluefruit.Periph.setConnInterval(9, 12); */
```

```
  // Set up and start advertising
  startAdv();
}

void startAdv(void)
{
  // Advertising packet
  Bluefruit.Advertising.addFlags(BLE_GAP_ADV_FLAGS_LE_ONLY_GENERAL_DISC_MODE);
  Bluefruit.Advertising.addTxPower();
  Bluefruit.Advertising.addAppearance(BLE_APPEARANCE_HID_KEYBOARD);

  // Include BLE HID service
  Bluefruit.Advertising.addService(blehid);

  // There is enough room for the dev name in the advertising packet
  Bluefruit.Advertising.addName();

  /* Start Advertising
   * - Enable auto advertising if disconnected
   * - Interval:  fast mode = 20 ms, slow mode = 152.5 ms
   * - Timeout for fast mode is 30 seconds
   * - Start(timeout) with timeout = 0 will advertise forever (until connected)
   *
   * For recommended advertising interval
   * https://developer.apple.com/library/content/qa/qa1931/_index.html
   */
  Bluefruit.Advertising.restartOnDisconnect(true);
  Bluefruit.Advertising.setInterval(32, 244);    // in unit of 0.625 ms
  Bluefruit.Advertising.setFastTimeout(30);      // number of seconds in fast mode
  Bluefruit.Advertising.start(0);                // 0 = Don't stop advertising after n seconds
}

//using this function to control phone
void sendCommand(uint16_t command) {
  // Make sure we are connected and bonded/paired
  for (uint16_t conn_hdl=0; conn_hdl < BLE_MAX_CONNECTION; conn_hdl++)
  {
    BLEConnection* connection = Bluefruit.Connection(conn_hdl);

    if ( connection && connection->connected() && connection->paired() )
    {
      // Turn on red LED when we start sending data
      digitalWrite(LED_RED, 1);
      Serial.println("Sending command...");

      // Send key press
      blehid.consumerKeyPress(conn_hdl, command);

      // Delay a bit between reports
      delay(10);

      // Send key release
      blehid.consumerKeyRelease(conn_hdl);

      // Turn off the red LED
      digitalWrite(LED_RED, 0);
      Serial.println("Command sent!");
    }
  }

  delay(250);
}

//using this function to configure with switch control
void sendSwitchControl(String command) {
  for (int i = 0; i < command.length(); i++) {
    blehid.keyPress(command.charAt(i));
    Serial.write(command.charAt(i));
    delay(5);
  }
  delay(250); // avoid sending the same command again and again
}

void loop()
{
```

```
  //button to play or pause players
  if (digitalRead(playpauseButtonPin) == HIGH) {
    sendCommand(CONTROL_PLAY_PAUSE);
  }

  //button to mute the phone
  if (digitalRead(muteButtonPin) == HIGH) {
    sendCommand(CONTROL_MUTE);
  }

  //button to play next media
  if (digitalRead(nextButtonPin) == HIGH) {
    sendCommand(CONTROL_SCAN_NEXT);
    sendCommand(AC_FORWARD);
  }

  //button to play last media
  if (digitalRead(backButtonPin) == HIGH) {
    sendCommand(CONTROL_SCAN_PREVIOUS);
    sendCommand(AC_BACK);
  }

//button to go back home
  if (digitalRead(homeButtonPin) == HIGH) {
    sendCommand(CONSUMER_BROWSER_HOME);
  }

//switch control 'select'
  if (digitalRead(switchControlButtonPin) == HIGH) {
    sendSwitchControl("SELECT");
    if (pressed){
      blehid.keyRelease(switchControlButtonPin);
      pressed=false;
      }
  }

  delay(5);
}
```

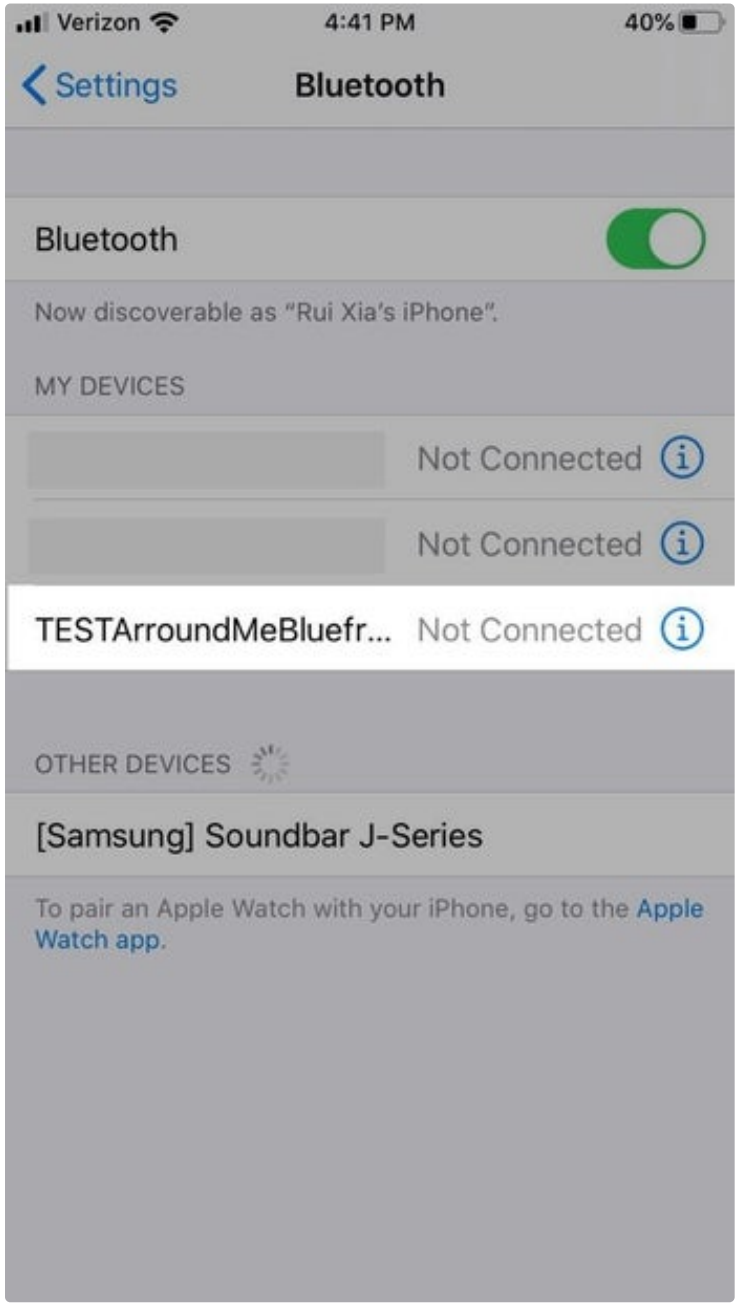




Step 3: Connect to Bluetooth

You would have to pair your Arduino board to your iPhone via bluetooth. To do this, follow these simple steps:

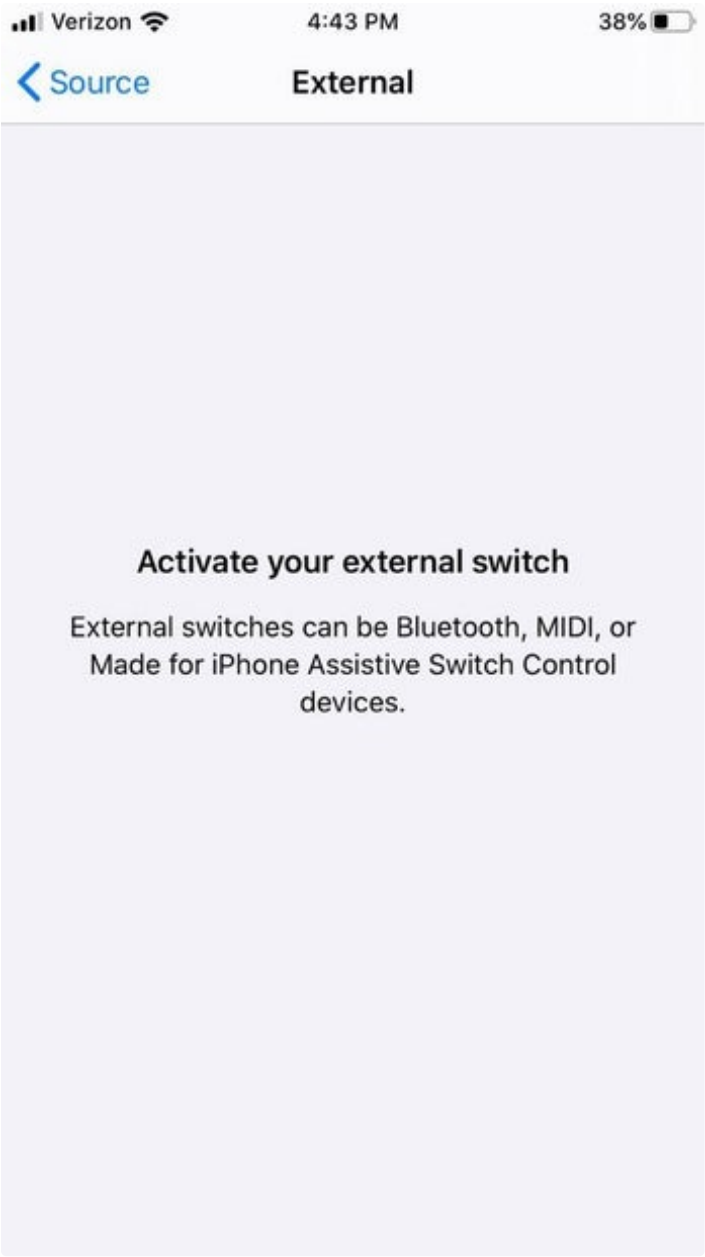
Upload your codes into your board after connecting your board through your micro-usb cable
Open up the Bluetooth options in Settings on your iPhone
You should see the name of your board in one of the bluetooth devices available. In this case, our board is called 'TESTAroundMeButton'
Pair both devices together and make sure it is connected



Step 4: Set Up Switch Control on IOS Accessibility Options

By now, you would have paired your Arduino board to your phone's bluetooth. You would need to link the switch control button (Button 6) to your iPhone's accessibility options by following these steps:

Go to Settings
Select Accessibility > Switch Control > Switches > Add New Switch... > External (and you will see the screen shown above)
Press on the button for switch control (button 6) and your iPhone would ask to name the switch. You can name it 'Select'
Next, select 'Select Item' so that the button will select chosen item when switch control is enabled on your phone
Once you have done this, your button is set up to select item when switch control is activated (Enable Switch Control on the Switch Control options)



Step 5: 3D Print Buttons (Optional)

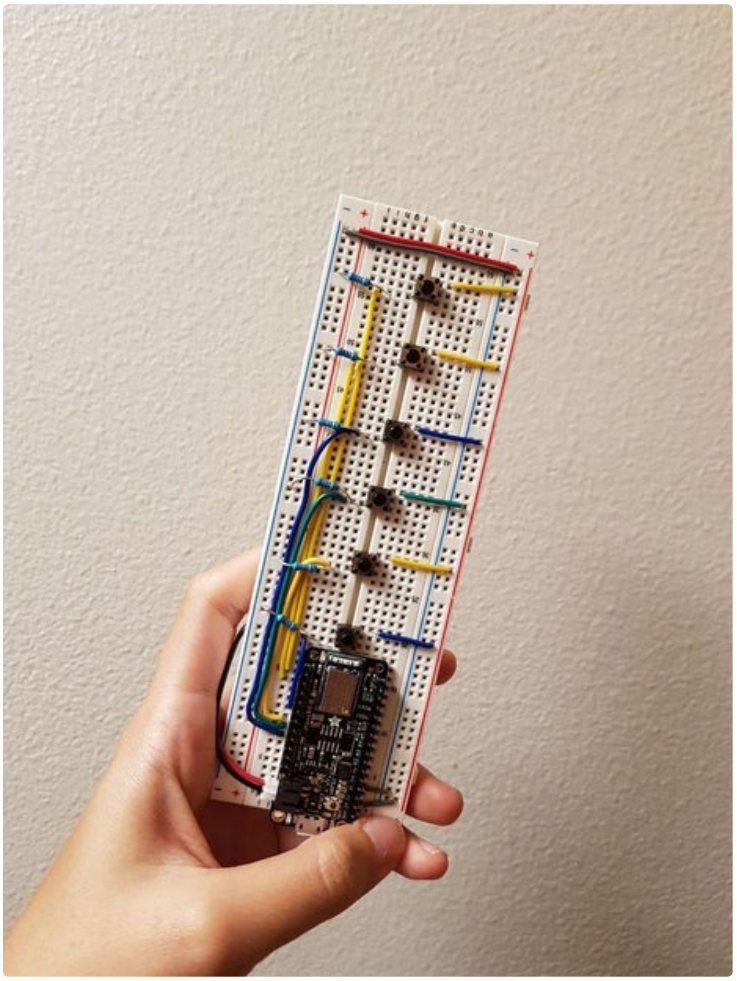
Download the files from https://www.thingiverse.com/thing:2911224
Convert files into gcode format and print from a 3D printer
Insert the tactile switches onto the printed button cases. You might need to attach it using double-sided tape, or rubbers with double-sided tape on their sides.

Step 6: Test & Done!

Upload the codes into the Arduino and start testing your buttons. To start the Microsoft Soundscape, download the app from the app store. Open Microsoft Soundscape and keep it running on the background while you test the buttons.

https://www.youtube.com/watch?v=YEX6B0baSmM&feature=youtu.be

https://youtu.be/5aFlNRv771k



Nice job putting this all together :)