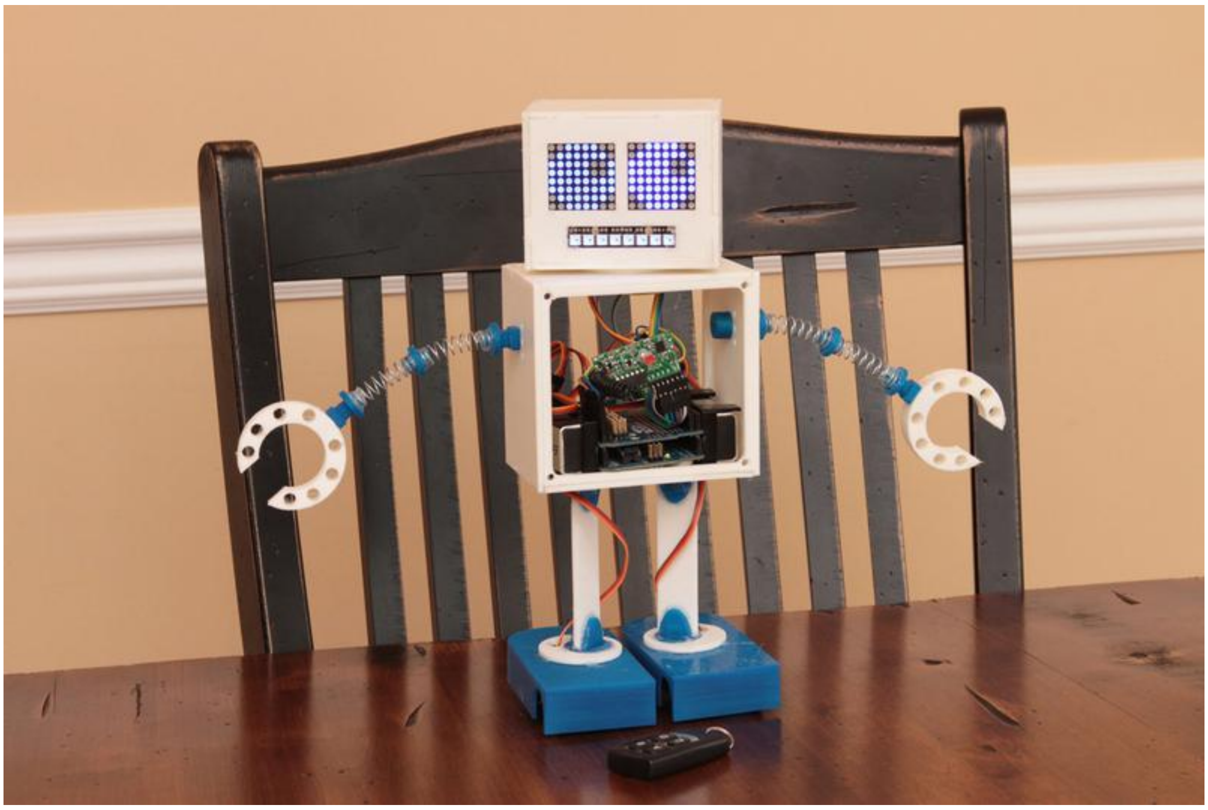
Making Adabot: Part 2

Created by Rick Winscot



https://learn.adafruit.com/making-adabot-part-2

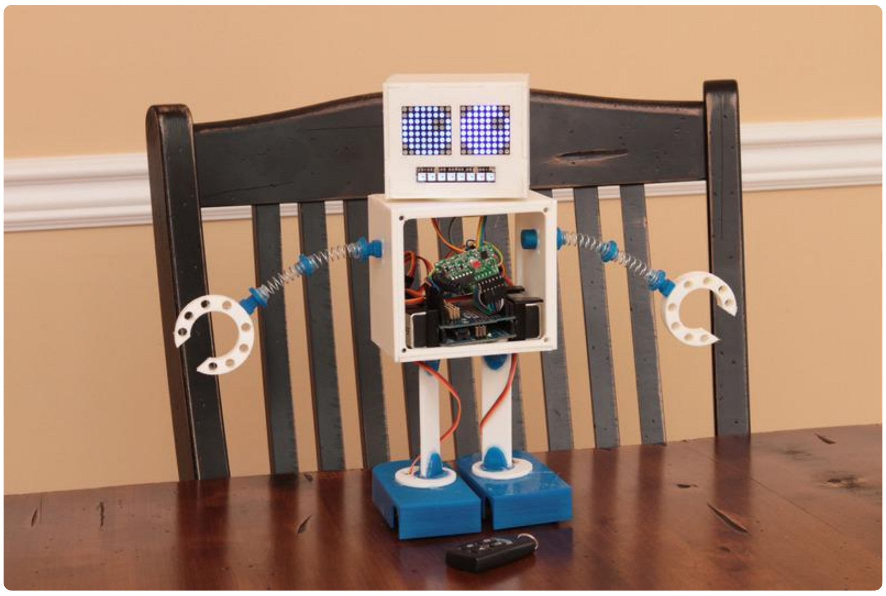
Last updated on 2022-12-01 02:04:15 PM EST

Table of Contents

Overview

Time to make your robot scoot around, blink, and smile.

We'll put the skin on in part three.



For best results, follow the AdaBot part 1 tutorial first! ()

Tools / Materials

Let's take inventory of the tools you'll need for this guide.

- hot-glue gun and glue sticks
- wire cutters
- screw driver
- soldering iron and solder

And materials...

- Arduino UNO R3
- PWM / Servo Shield
- 2x stacking shield headers
- 1x micro servo
- 2x continuous rotation servo
- #10 o-rings
- 2x LED matrix w/I2C backpack
- 1x NeoPixel Stick
- RF receiver (momentary)

- RF 4-button transmitter
- 5v 2A power supply (for testing on your desktop)
- 2X 9v batteries (when you're ready to go wireless)
- 2X 9v battery clips
- Female DC power adapter
- Hook-up wire

KLEIN TOOLS
WARNING: NOT INSULATED WILL NOT PROTECT AGAINST ELECTRICAL SHOCK

With regard to the servos, this design was crafted around the dimensions of the popular and inexpensive Hextronik HXT900.

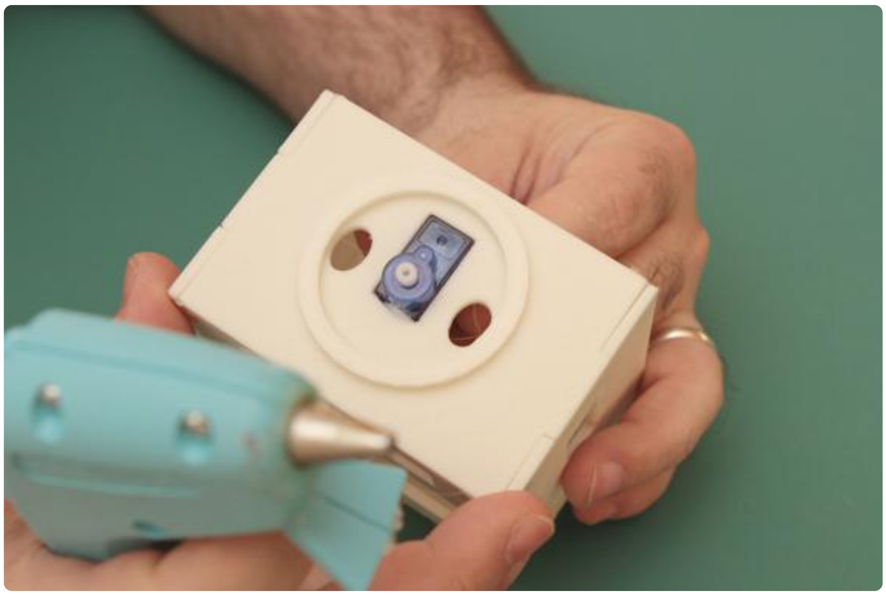
We've modified a set of these for continuous rotation using this guide...

Modifying Servos for Continuous Rotation ()

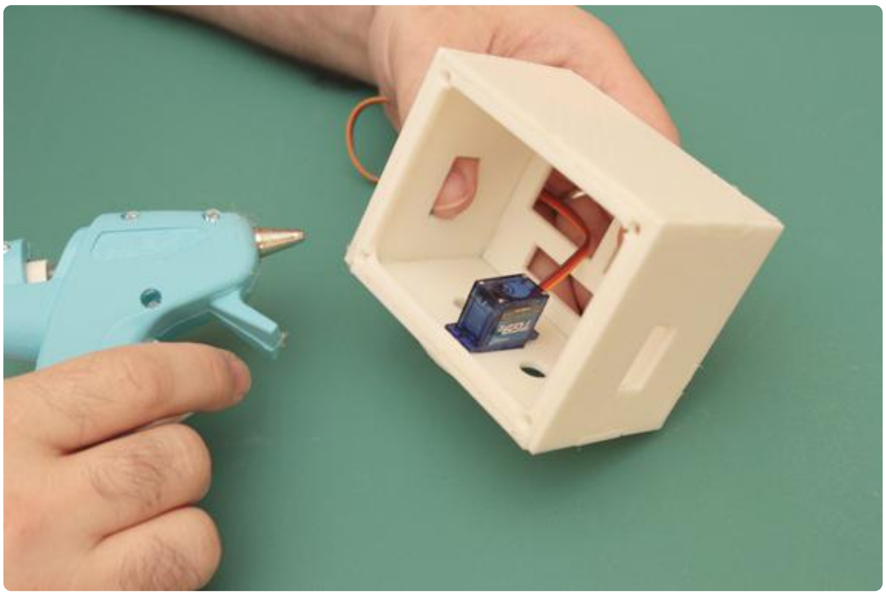
Let's put the pieces together!

Neck Servo

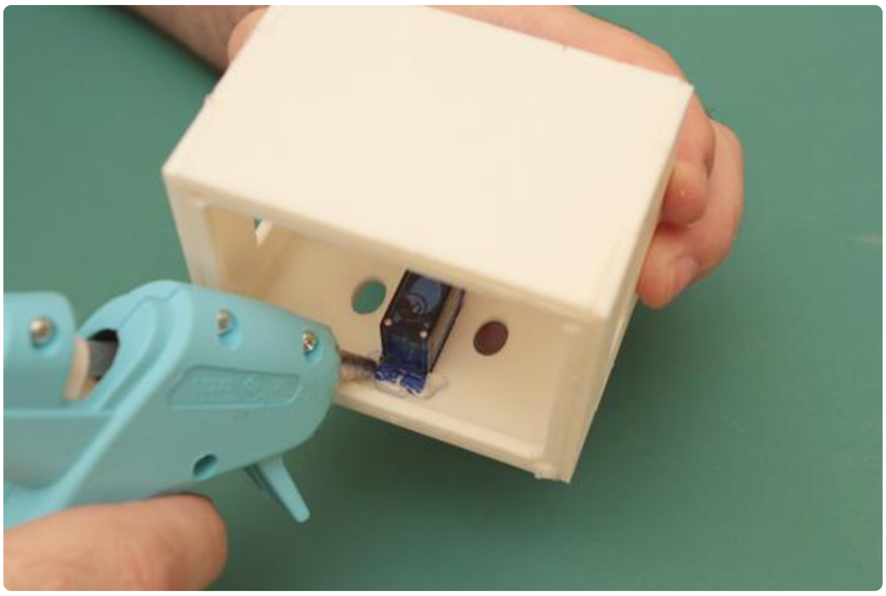
Lets start with the neck servo, a small 9g one to be exact. Put a little hot-glue on the underside of the tab and push it into place.



Orientation is important... notice how the servo aligns with the holes.



Make the servo extra secure by adding more hot-glue over the top of the tabs.

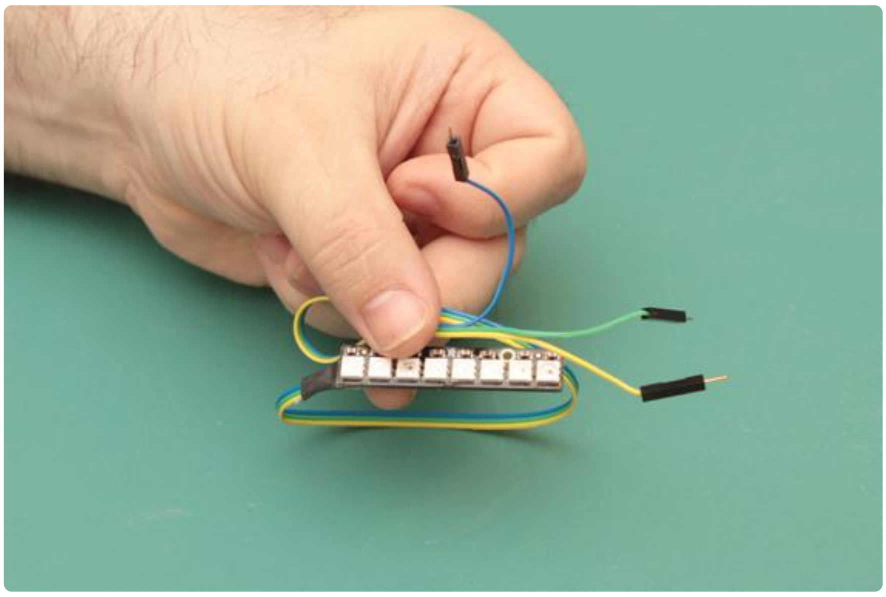


If you need a little more detail, you can download a complete photostream of each step.

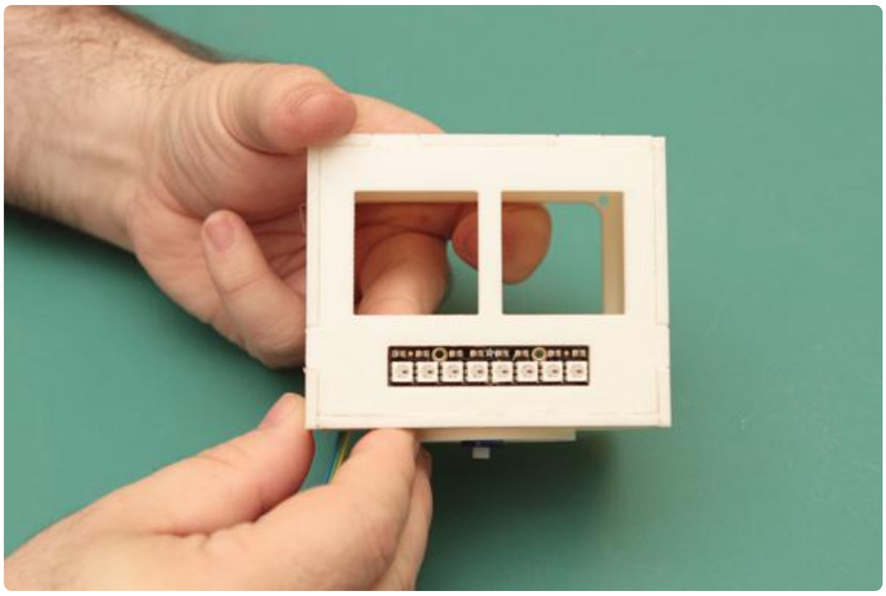
Neck Servo Photostream

Mouth

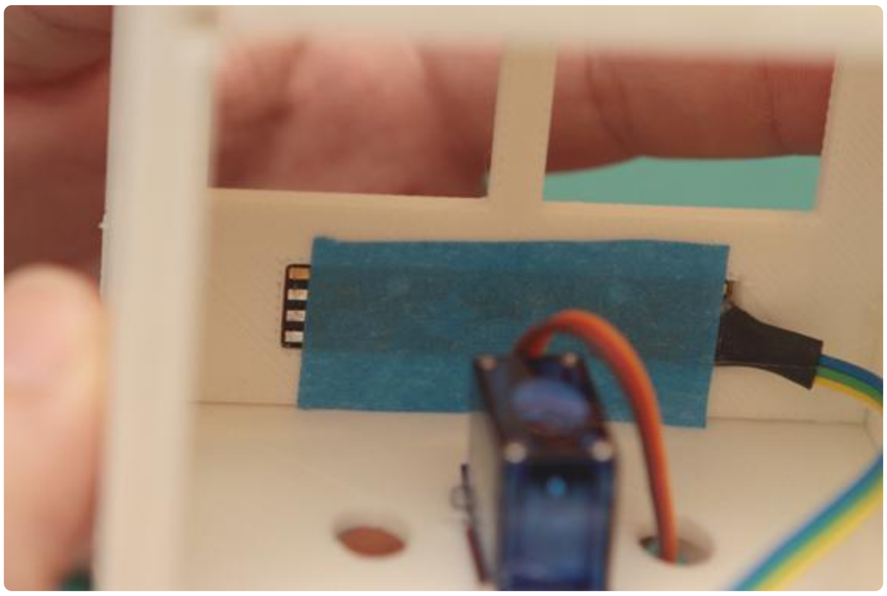
Grab a NeoPixel stick and consult the Adafruit NeoPixel Überguide () for hook-up details o' plenty.



Insert the stick into the mouth hole...



This isn't a snap-fit, so you're going to have to secure it with something.



I used a bit ot painters tape on the back and then added a bit of hot-glue on the front to hold it in place.

Connect the Neo Pixel stick as follows:

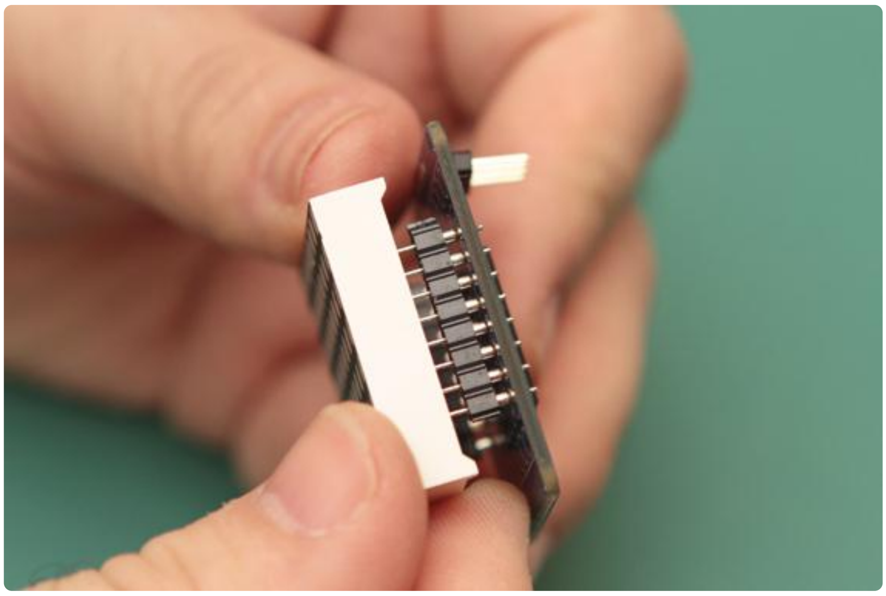
- VCC --> 3v
- GND --> GND
- SIG --> D7 (digital pin)

Mouth Photostream

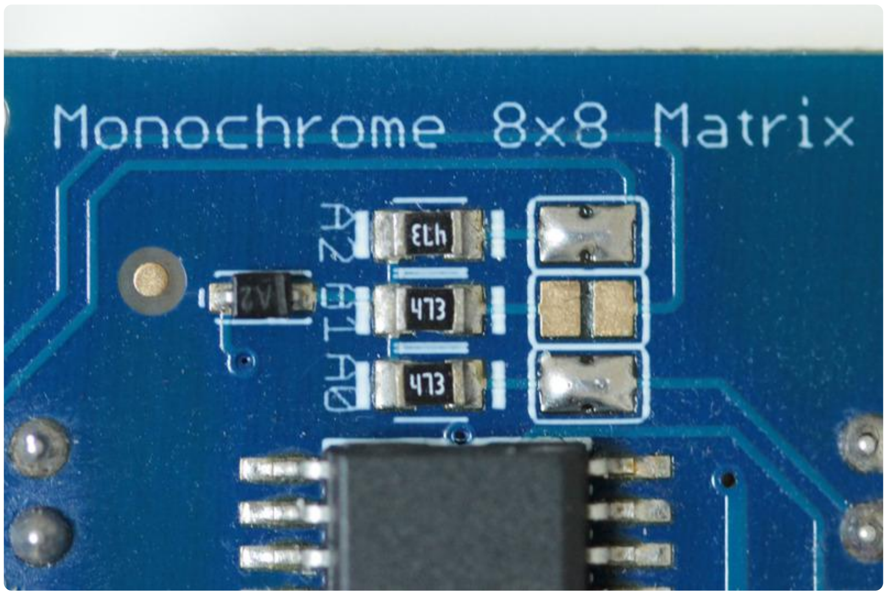
Eyes

Either solder the LED matrix into place on the backpack, or add some machine pin headers so that you can swap out the matrix as you like.

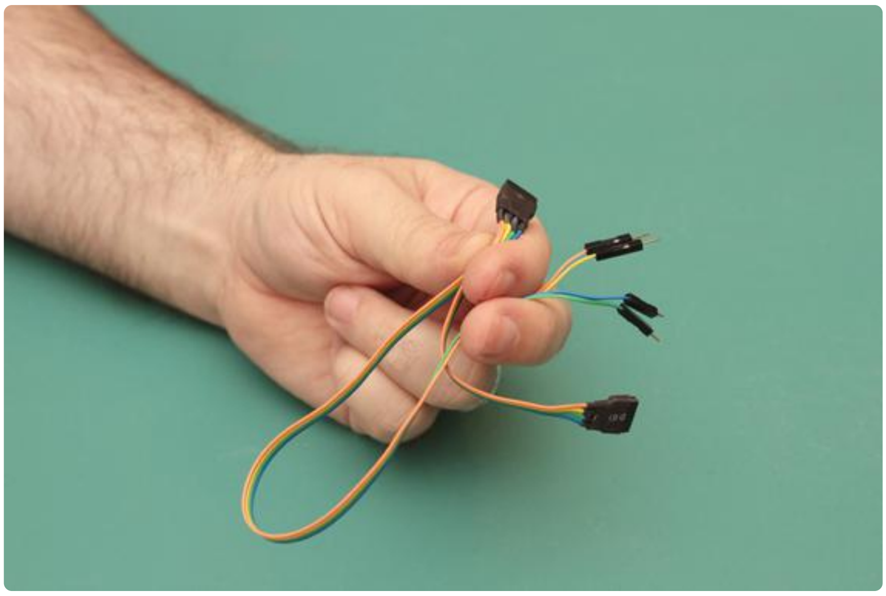
It's a good idea to pre-assemble all the electronics loosely before gluing everything into place.



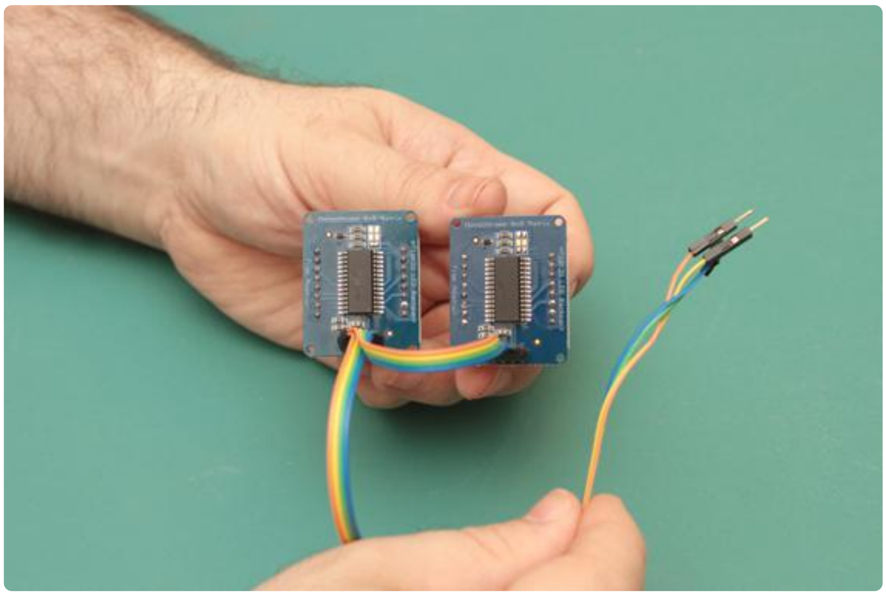
Now is a good time to adjust the address settings () on each matrix. I found the 0x73 works a treat!



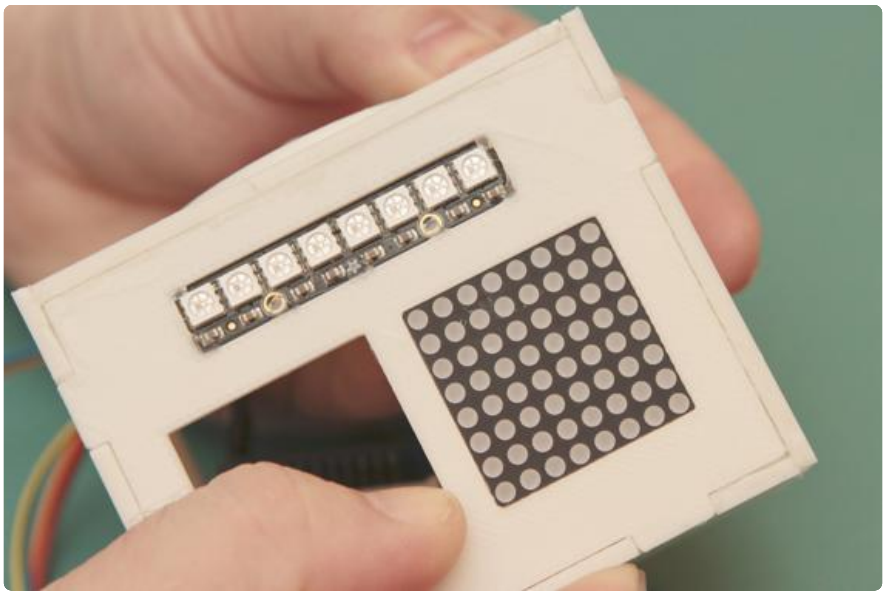
Grab your hook-up wire and create a harness that chains the eyes together and then connects to the Arduino...



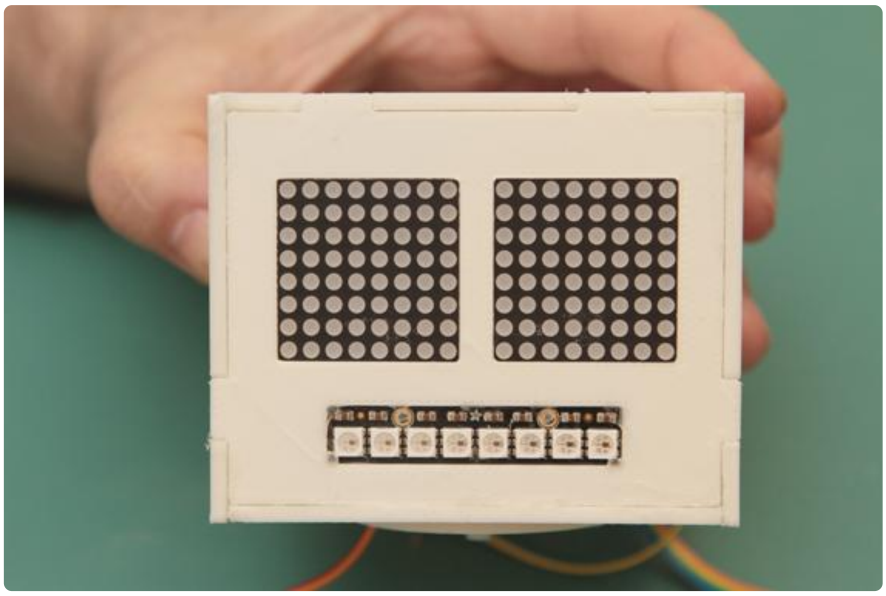
Or you can solder the wires directly to the backpack.



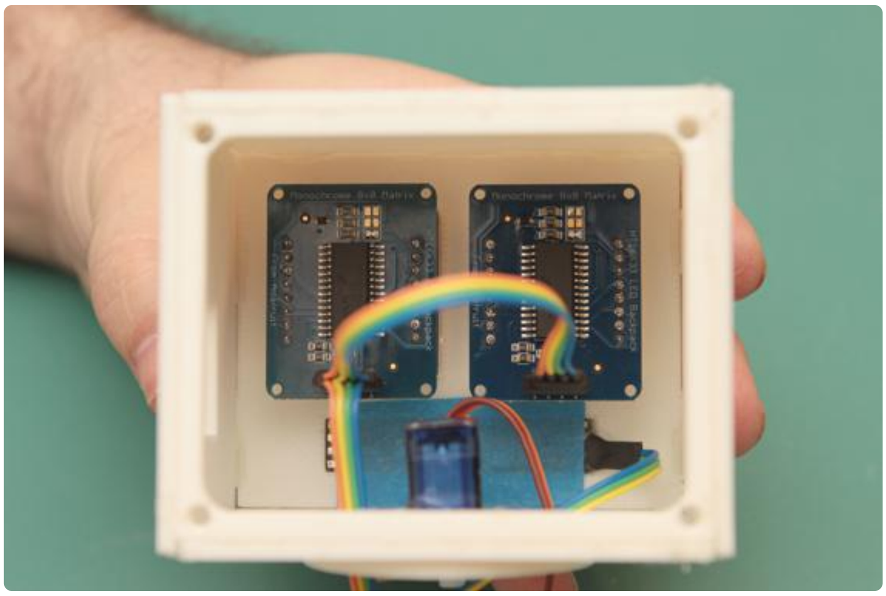
Insert each matrix into the eye hole...



And secure with just a touch of hot-glue on the corners.



Double check the wiring.



And pull the ends through the holes in the base of the head.

When you're ready, connect the pins of the eye harness into the Arduino as follows:

- VCC --> VIN
- GND --> GND
- SDA --> A4
- SCL --> A5

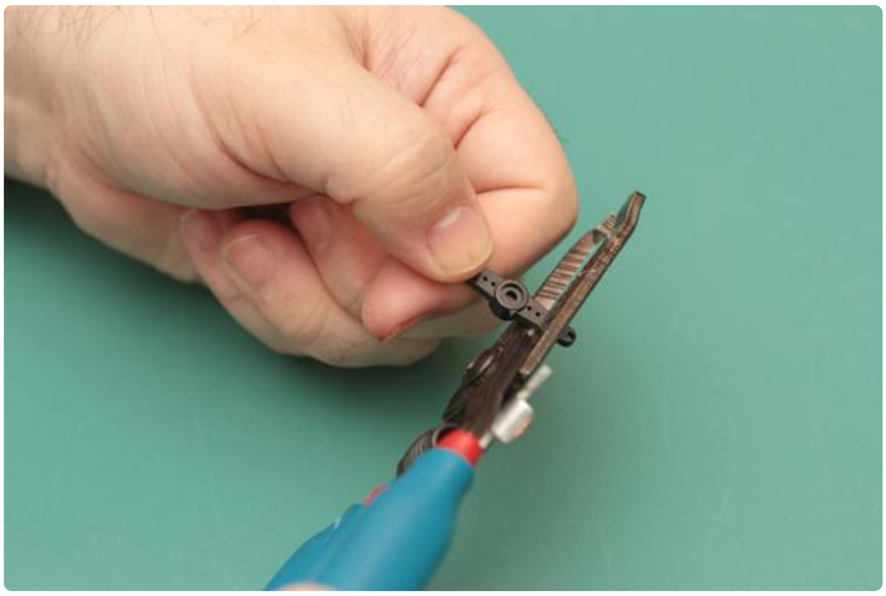
If you plug VCC into the Arduino 5v line... power to the radio will sag below it's minimum requirement. Alternatively, you could plug VCC directly into your battery source as long as it won't exceed 9 volts.

Eyes Photostream

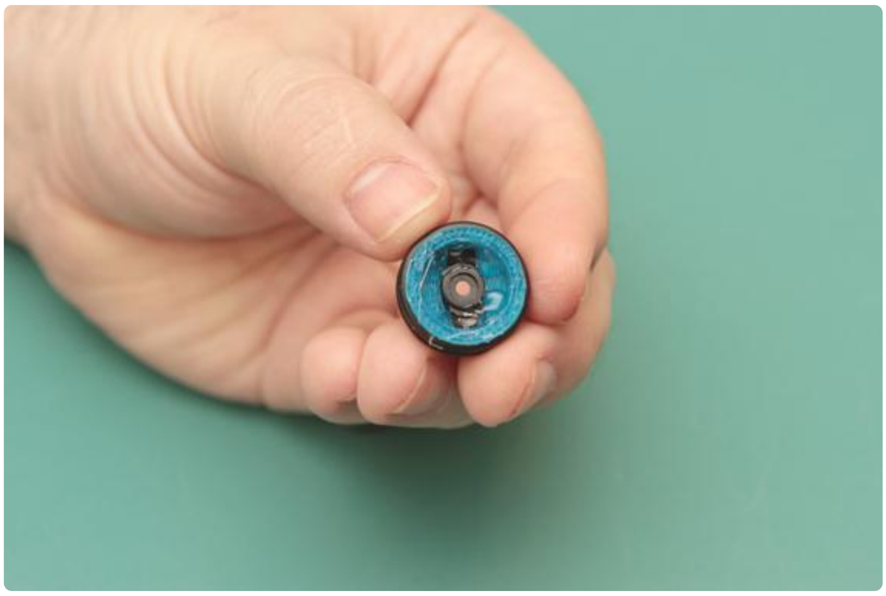
Footie Wheels

You'll need your continuous rotation servos and some more #10 o-rings (two per hub) to complete this step.

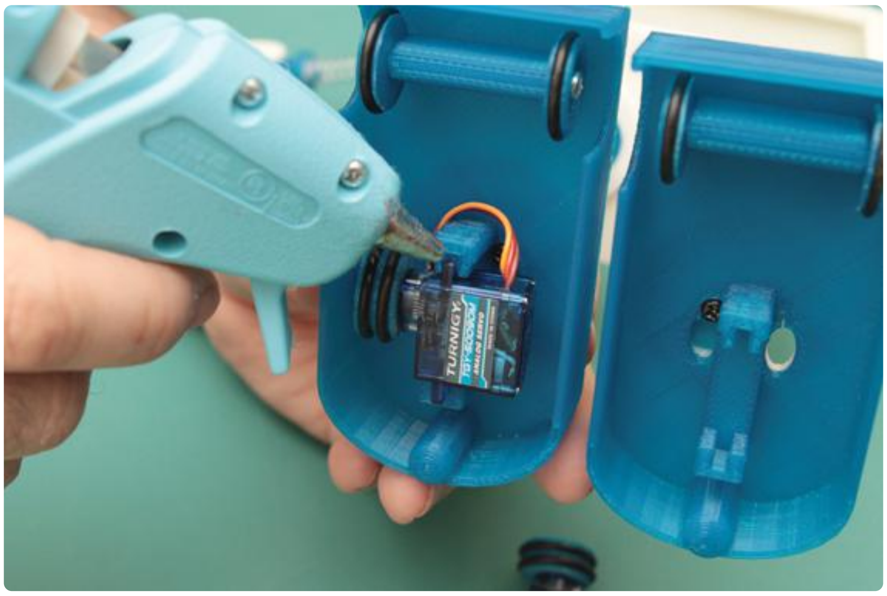
Trim the servo horn down a bit so that it will fit inside the hub.



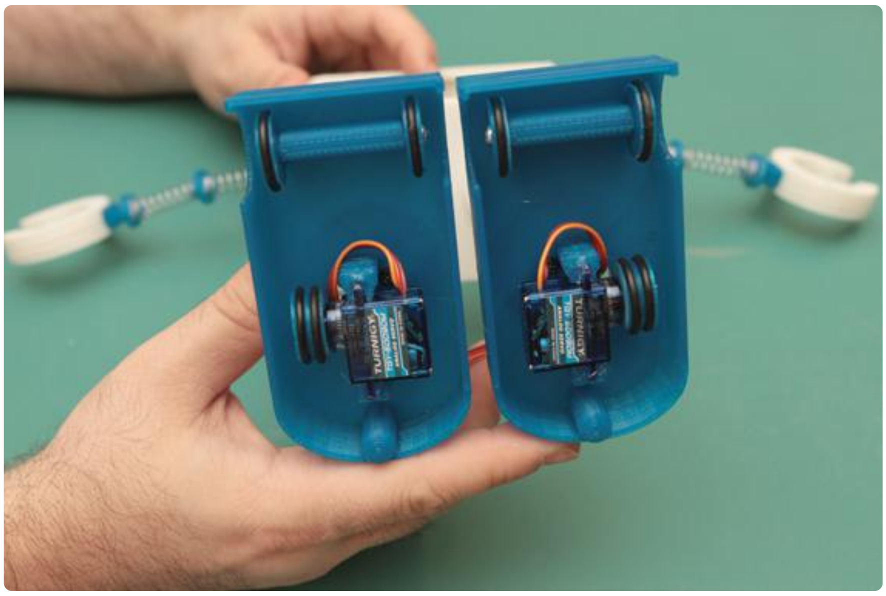
Make sure it's oriented correctly, and then hot-glue in place.



Screw the hub to the continuous rotation servo and hot-glue the servo into the footie.



Rise. Wash. Repeat for the other foot.

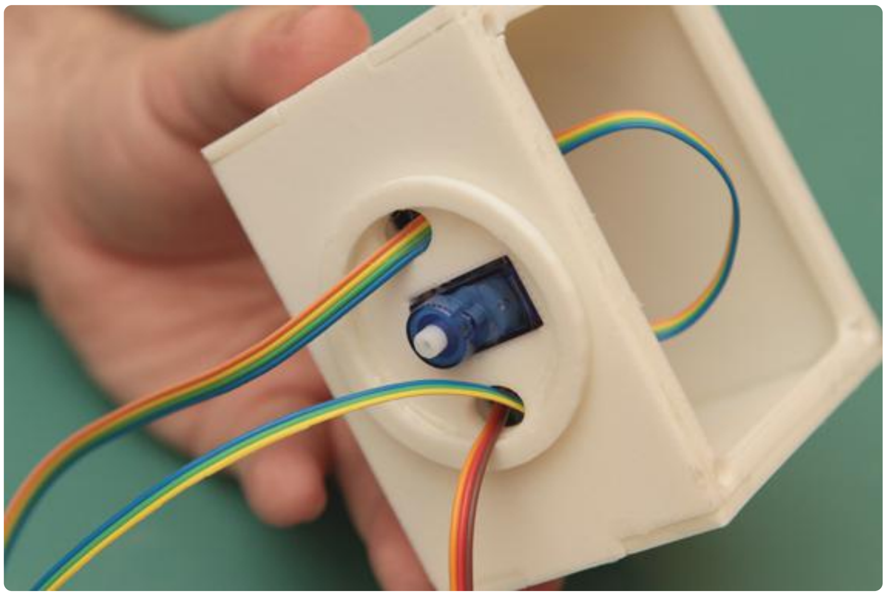


Ready to roll!

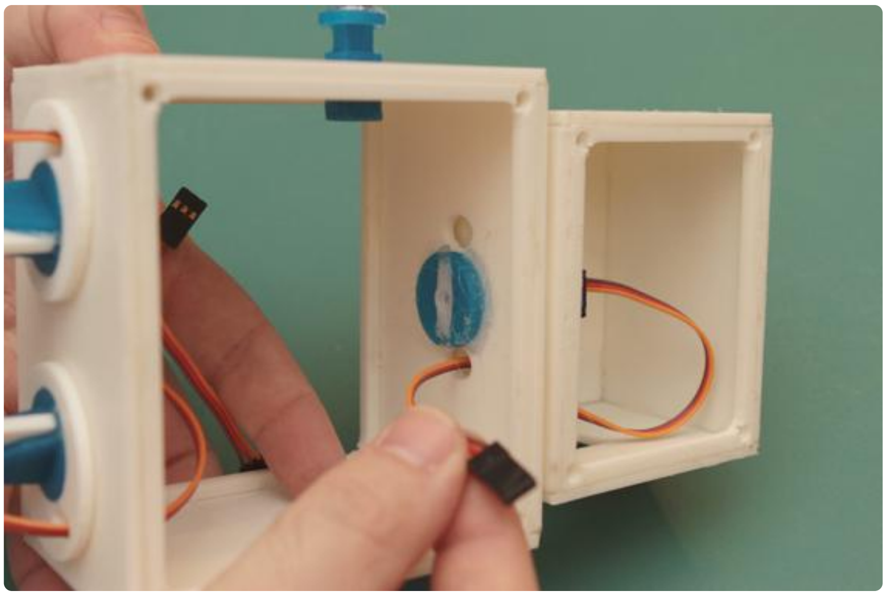
Footie Servo Photostream

Servo Controller

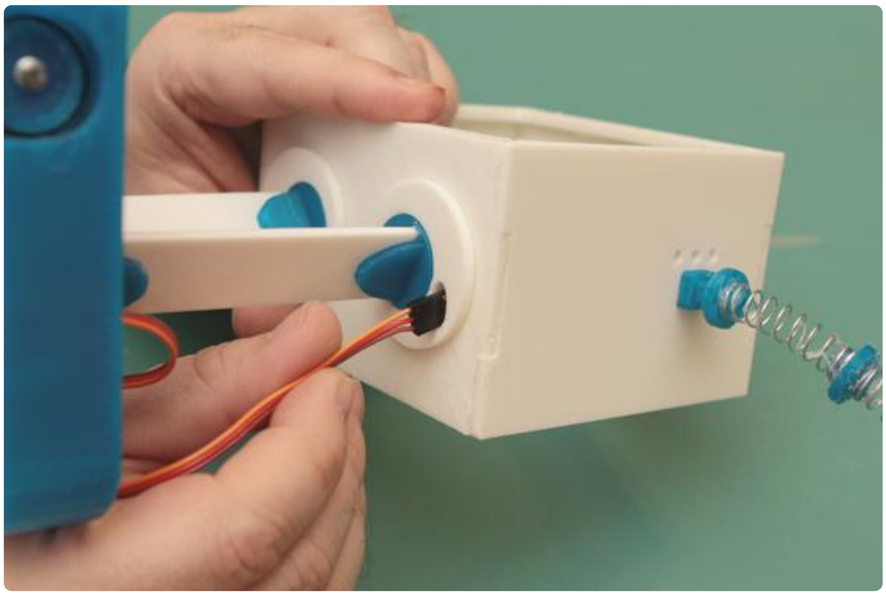
Take the wires that you passed through the base of the head...



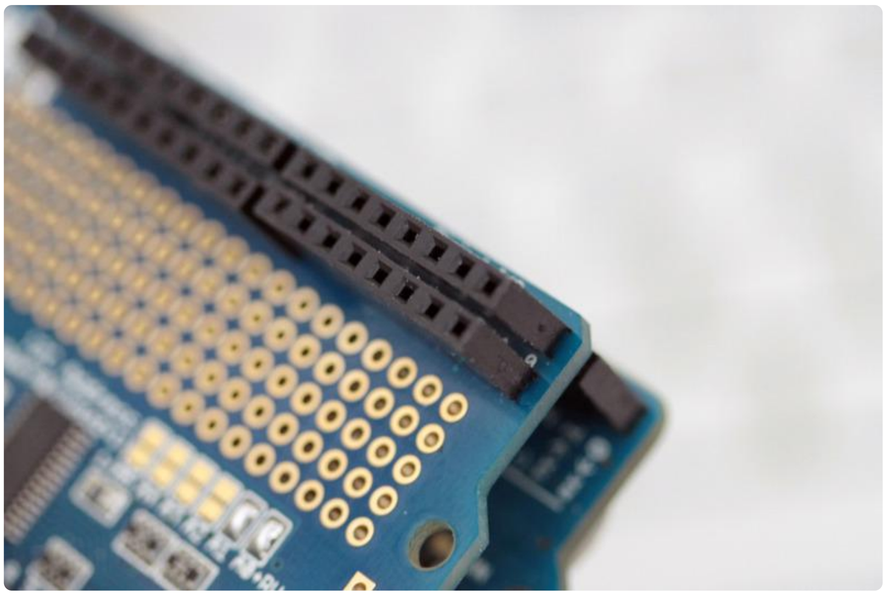
And pass them through the holes in the body. We still have to add the skin... so wait before screwing-in the servo horn.



Pass the wires from the servos in the feet through the body as well.

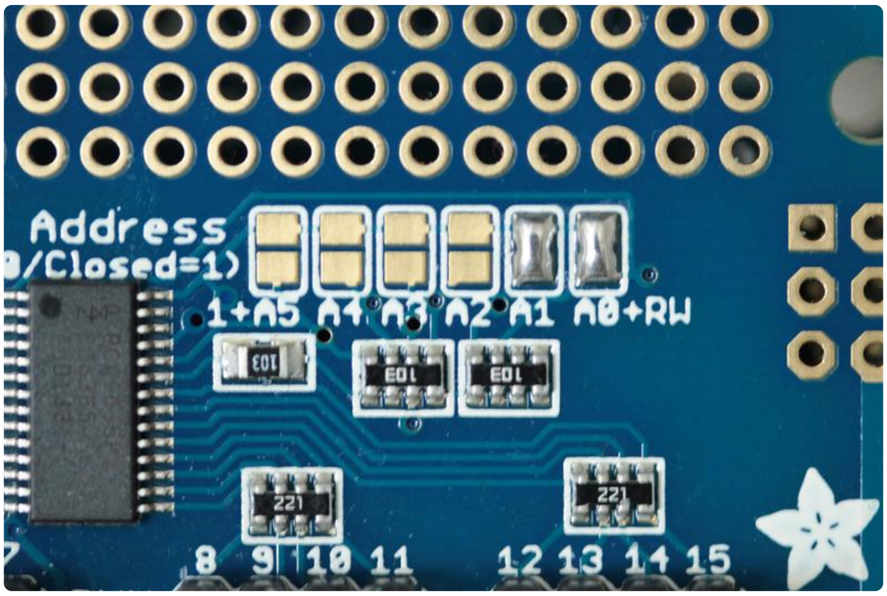


Swap out the header pins for stacking headers to give you twice as much awesome.



Adjust the address () on your Servo Shield (http://adafru.it/1411) ... remember that you will need to make adjustments in Adafruit_PWMServoDriver.h (Adafruit PWM Library).

I've used 0x43 here.



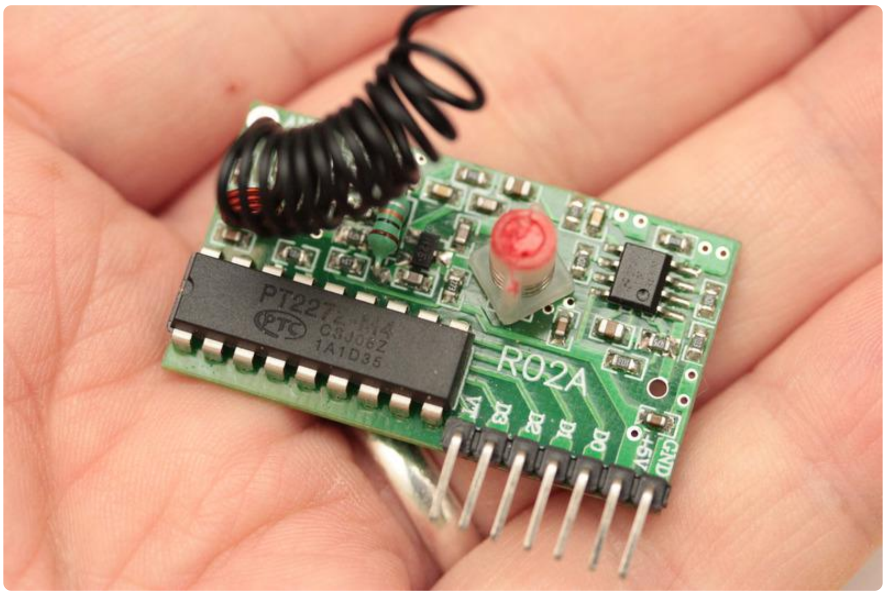
Grab your Arduino UNO (http://adafru.it/50) and plug the shield into it.

Connect your servos to the shield PWM inputs as follows:

- Left servo --> PWM 0
- Right servo-> PWM 1
- Neck servo-> PWM 2

RF Remote

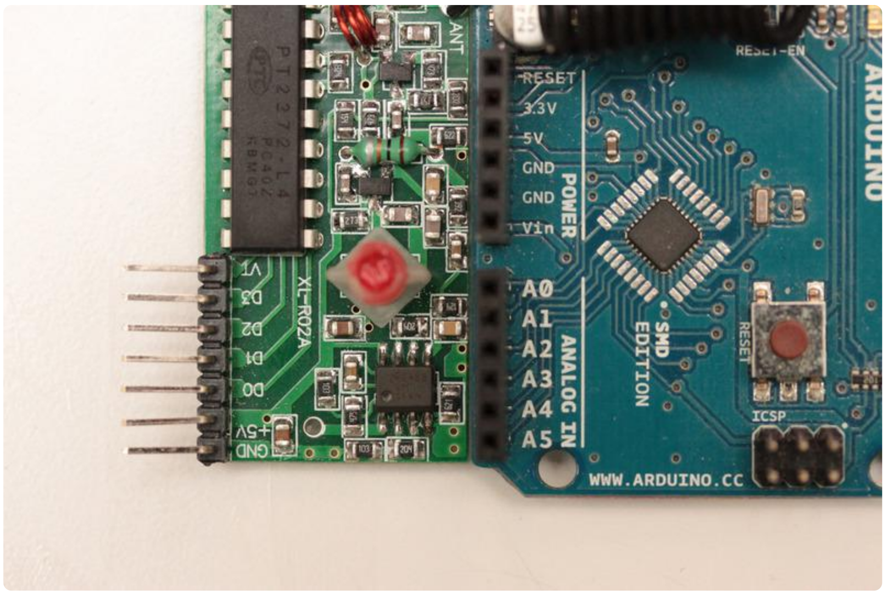
Hidden treasures abound in the Adafruit store... and this is one. A little radio module with four pins that can be controlled from a battery powered key fob.



We will use those four pins to control the direction of each servo: forward, backward, left, and right.

Connect the pins of the radio module to the Arduino as follows.

- GND -> GND
- +5V --> 5v
- D0 ----> A0
- D1 ----> A1
- D2 ----> A2
- D3 ----> A3



If your robot keeps resetting or the radio doesn't seem to be working... voltage could be sagging below 5 volts - the radio module doesn't operate reliably below that threshold.

Source Code

Here is the code that makes your robot go. You'll need to have a few libraries installed in the Arduino IDE.

- PWM Servo Driver Library ()
- LED Backpack Library ()
- GFX Library () (used in the LED Backpack Library)
- BusIO Library ()
- Neo Pixel Library ()

When you're ready to go wireless, cut and tin the leads of one of the 9V battery clips and insert the wires into the PWM shield PC terminal.

The other battery clip is for the Arduino... via the barrel connector.

```
#include &lt;Arduino.h&gt;
#include &lt;Wire.h&gt;
#include &lt;Adafruit_PWMServoDriver.h&gt;
#include &lt;Adafruit_NeoPixel.h&gt;
#include "Adafruit_LEDBackpack.h"
#include "Adafruit_GFX.h"

Adafruit_NeoPixel mouth = Adafruit_NeoPixel(8, 7, NEO_GRB + NEO_KHZ800);
Adafruit_PWMServoDriver servos = Adafruit_PWMServoDriver();
Adafruit_8x8matrix eyes = Adafruit_8x8matrix();
```

```
// Eye bitmaps are stored in program memory.
static uint8_t PROGMEM
blinkImg[][8] = {
  {
    B00111100,
    B01111110,
    B11111111,
    B11111111,
    B11111111,
    B11111111,
    B01111110,
    B00111100 }
  ,
  {
    B00000000,
    B01111110,
    B11111111,
    B11111111,
    B11111111,
    B11111111,
    B01111110,
    B00111100 }
  ,
  {
    B00000000,
    B00000000,
    B00111100,
    B11111111,
    B11111111,
    B11111111,
    B00111100,
    B00000000 }
  ,
  {
    B00000000,
    B00000000,
    B00000000,
    B00111100,
    B11111111,
    B01111110,
    B00011000,
    B00000000 }
  ,
  {
    B00000000,
    B00000000,
    B00000000,
    B00000000,
    B10000001,
    B01111110,
    B00000000,
    B00000000 }
};

// Analog threshold used to determine when FOB buttons are pressed.
const byte BUTTON_THRESHOLD = 50;

uint8_t
blinkIndex[] = { 1, 2, 3, 4, 3, 2, 1 }, // Blink bitmap sequence.
blinkCountdown = 100,  // Countdown to next blink (in frames).
gazeCountdown  =  75,  // Countdown to next eye movement.
gazeFrames     =  50;

int8_t
eyeX = 3, eyeY = 3,     // Current eye position.
newX = 3, newY = 3,     // Next eye position.
dX   = 0, dY   = 0;
```

```
unsigned long tme = 0;  // Last time we processed a mouth animation.
unsigned long slc = 50; // Milliseconds between mouth animations.
unsigned long ms = 0;   // Used to track a millis() time-slice.

int stt = 0;            // Revolving start position for the neck servo.
int pos = 0;            // Current 'moving' position of the neck servo.
int dst = 0;            // Random destination for the neck servo.
int dur = 0;            // Number of frames used by the easing function.
int fme = 0;            // Animation frame counter.

void setup() {
  // Make sure to adjust the address jumpers on the shield, and
  // in Adafruit_PWMServoDriver.h on line ~48 -- I've used 0x43.
  eyes.begin(0x73);
  // Brightness is set _very_ low to conserve power.
  eyes.setBrightness(1);

  // We're using a Neo Pixel Stick for the mouth.
  mouth.begin();
  mouth.show();

  // We have two servos in the feet and one in the neck.
  servos.begin();
  servos.setPWMFreq(60);
}

void loop() {

  // Every few milliseconds we will animate the mouth.
  ms = millis();
  if ( tme + slc &lt; ms ) {
    tme = ms;

    // 10% chance that the mouth will animate.
    if ( random( 1, 10 ) == 1 ) {
      // LEDs are set to ~1% brightness.
      animateMouth(mouth.Color(0, 0, 5), 15);
    }
    else {
      // LEDs are set to ~10% brightness.
      animateMouth(mouth.Color(0, 0, 25), 15);
    }

    // Every few milliseconds we will update the neck position.
    if ( pos != dst &amp;&amp; fme &lt; dur )
    {
      // Update position until we arrive at dst.
      pos = (int)easeInOut( fme, stt, ( dst - stt ), dur );
      fme += 1;
      servos.setPWM(2, 0, pos);
    }
    else {
      // Advance to the new destination.
      stt = pos;
      // Adjust this range if necessary.
      dst = random(150, 350);
      fme = 0;
      dur = 50;
    }
  }

  // Mirror animation of each 8x8 eye matrix.
  eyes.clear();
  eyes.drawBitmap(0, 0,
  // Blinking? Yes, look-up bitmap #. No, show bitmap 0.
  blinkImg[ (blinkCountdown &lt; sizeof(blinkIndex)) ? blinkIndex[blinkCountdown] :
0 ], 8, 8, LED_ON);
  // Decrement blink counter.  At end, set random time for next blink.
  if(--blinkCountdown == 0) blinkCountdown = random(5, 180);
  // Add a pupil (2x2 black square) atop the blinky eyeball bitmap.
  // Periodically, the pupil moves to a new position...
  if(--gazeCountdown &lt;= gazeFrames) {
    // Eyes are in motion - draw pupil at interim position.
    eyes.fillRect(
    newX - (dX * gazeCountdown / gazeFrames),
    newY - (dY * gazeCountdown / gazeFrames),
    2, 2, LED_OFF);
    if(gazeCountdown == 0) {    // Last frame?
      eyeX = newX;
      eyeY = newY; // Yes.  What's new is old, then...
      do { // Pick random positions until one is within the eye circle.
        newX = random(7);
        newY = random(7);
        dX   = newX - 3;
        dY   = newY - 3;
      }
      while((dX * dX + dY * dY) &gt;= 10);        // Thank you Pythagoras.
      dX            = newX - eyeX;             // Horizontal distance to move.
      dY            = newY - eyeY;             // Vertical distance to move.
      gazeFrames    = random(3, 15);           // Duration of eye movement.
      gazeCountdown = random(gazeFrames, 120); // Count to end of next movement.
    }
  }
  else {
    // Not in motion yet -- draw pupil at current static position.
    eyes.fillRect(eyeX, eyeY, 2, 2, LED_OFF);
  }
  eyes.writeDisplay();

  // Now, lets check to see if any of the buttons on the FOB are pressed.
  byte a = analogRead(3); // forward right
  byte b = analogRead(2); // forward left
  byte c = analogRead(1); // reverse right
  byte d = analogRead(0); // reverse left

  if ( ( b &gt; BUTTON_THRESHOLD &amp;&amp; d &gt; BUTTON_THRESHOLD ) || ( b &lt;
BUTTON_THRESHOLD &amp;&amp; d &lt; BUTTON_THRESHOLD ) ) {
    // Turn servo 0 off if FOB buttons B + D are both pressed or released.
    servos.setPWM(0, 0, 0);
  }
  else {
    if ( b &gt; BUTTON_THRESHOLD ) {
      // FOB button B is pressed... forward left.
      servos.setPWM(0, 0, 400);
    }
    if ( d &gt; BUTTON_THRESHOLD ) {
      // FOB button D is pressed... reverse left.
      servos.setPWM(0, 0, 350);
    }
  }

  if ( ( a &gt; BUTTON_THRESHOLD &amp;&amp; c &gt; BUTTON_THRESHOLD ) || ( a &lt;
BUTTON_THRESHOLD &amp;&amp; c &lt; BUTTON_THRESHOLD ) ) {
    // Turn servo 1 off if FOB buttons A + C are both pressed or released.
    servos.setPWM(1, 0, 0);
  }
  else {
    if ( a &gt; BUTTON_THRESHOLD ) {
      // FOB button A is pressed... forward left.
      servos.setPWM(1, 0, 350);
    }
    if ( c &gt; BUTTON_THRESHOLD ) {
      // FOB button C is pressed... reverse left.
      servos.setPWM(1, 0, 400);
    }
  }
}
```

```
void animateMouth(uint32_t c, uint8_t wait) {
  // Quick loop that does a symmetric animation on a single Neo Pixel Strip.
  for(uint16_t i=( mouth.numPixels() / 2 ); i&lt;mouth.numPixels(); i++) {
    // Pixels 3, 2, 1, and 0.
    mouth.setPixelColor(mouth.numPixels() - i - 1, c);
    // Pixels 4, 5, 6, and 7.
    mouth.setPixelColor(i, c);
    mouth.show();
    delay(wait);
  }
}

float easeInOut( float t, float b, float c, float d )
{
  // Function used to smooth servo movements.
  if ((t/=d/2) &lt; 1)
    return c/2*t*t*t + b;

  return c/2*((t-=2)*t*t + 2) + b;
}
```