

BeatListener

```
class BeatListener implements
AudioListener
{
private BeatDetect beat;

private AudioPlayer source;

BeatListener(BeatDetect beat, AudioPlayer source)

{
this.source = source;

this.source.addListener(this);
this.beat = beat;

}

void samples(float[] samps)
{
beat.detect(source.mix);
}

void samples(float[] sampsL, float[] sampsR)
{
beat.detect(source.mix);
}

}
```

BeatWrite

```
*/

import processing.serial.*;

import ddf.minim.*;
import ddf.minim.analysis.*
;
import cc.arduino.*;

Minim minim;
AudioPlayer song;

BeatDetect beat;

BeatListener bl;

Arduino arduino;

int ledPin = 12;
```

```

// LED connected to digital pin 12
int ledPin2 = 8;
// LED connected to digital pin 1
int ledPin3 = 2;
// LED connected to digital pin 0

float kickSize, snareSize, hatSize;

void setup()
{
  size(512, 200, P3D);

  minim = new Minim(this);

  arduino = new Arduino(this, Arduino.list()[1], 57600);

  song = minim.loadFile("freebird.mp3", 2048);

  song.play();
  // a beat detection object that is FREQ_ENERGY mode that
  // expects buffers the length of song's buffer size
  // and samples captured at songs's sample rate

  beat = new BeatDetect(song.bufferSize(), song.sampleRate());

  // set the sensitivity to 300 milliseconds
  // After a beat has been detected, the algorithm will wait for 300 milliseconds
  // before allowing another beat to be reported. You can use this to dampen the
  // algorithm if it is giving too many false-positives. The default value is 10,
  // which is essentially no damping. If you try to set the sensitivity to a negative value,
  // an error will be reported and it will be set to 10 instead.

  beat.setSensitivity(100);

  kickSize = snareSize = hatSize = 16;
  // make a new beat listener, so that we won't miss any buffers for the analysis

  bl = new BeatListener(beat, song);

  textFont(createFont("Helvetica", 16));

  textAlign(CENTER);

  arduino.pinMode(ledPin, Arduino.OUTPUT);

  arduino.pinMode(ledPin2, Arduino.OUTPUT);

  arduino.pinMode(ledPin3, Arduino.OUTPUT);

}

```

```

void draw() {
  background(0);
  fill(255);

  if(beat.isKick()) {
    arduino.digitalWrite(ledPin, Arduino.HIGH); // set the LED on

    kickSize = 32;
  }

  if(beat.isSnare()) {
    arduino.digitalWrite(ledPin2, Arduino.HIGH); // set the LED on

    snareSize = 32;
  }
  if(beat.isHat()) {
    arduino.digitalWrite(ledPin3, Arduino.HIGH); // set the LED on

    hatSize = 32;
  }
  arduino.digitalWrite(ledPin, Arduino.LOW); // set the LED off

  arduino.digitalWrite(ledPin2, Arduino.LOW); // set the LED off

  arduino.digitalWrite(ledPin3, Arduino.LOW); // set the LED off

  textSize(kickSize);
  text("KICK", width/4, height/2);

  textSize(snareSize);
  text("SNARE", width/2, height/2);

  textSize(hatSize);
  text("HAT", 3*width/4, height/2);

  kickSize = constrain(kickSize * 0.95, 16, 32);

  snareSize = constrain(snareSize * 0.95, 16, 32);

  hatSize = constrain(hatSize * 0.95, 16, 32);
}

void stop() {
  // always close Minim audio classes when you are finished with them

  song.close();
  // always stop Minim before exiting
  minim.stop();
  // this closes the sketch

  super.stop();
}

```