

# SOUND INTERACTION WORKSHOP INTERACTION

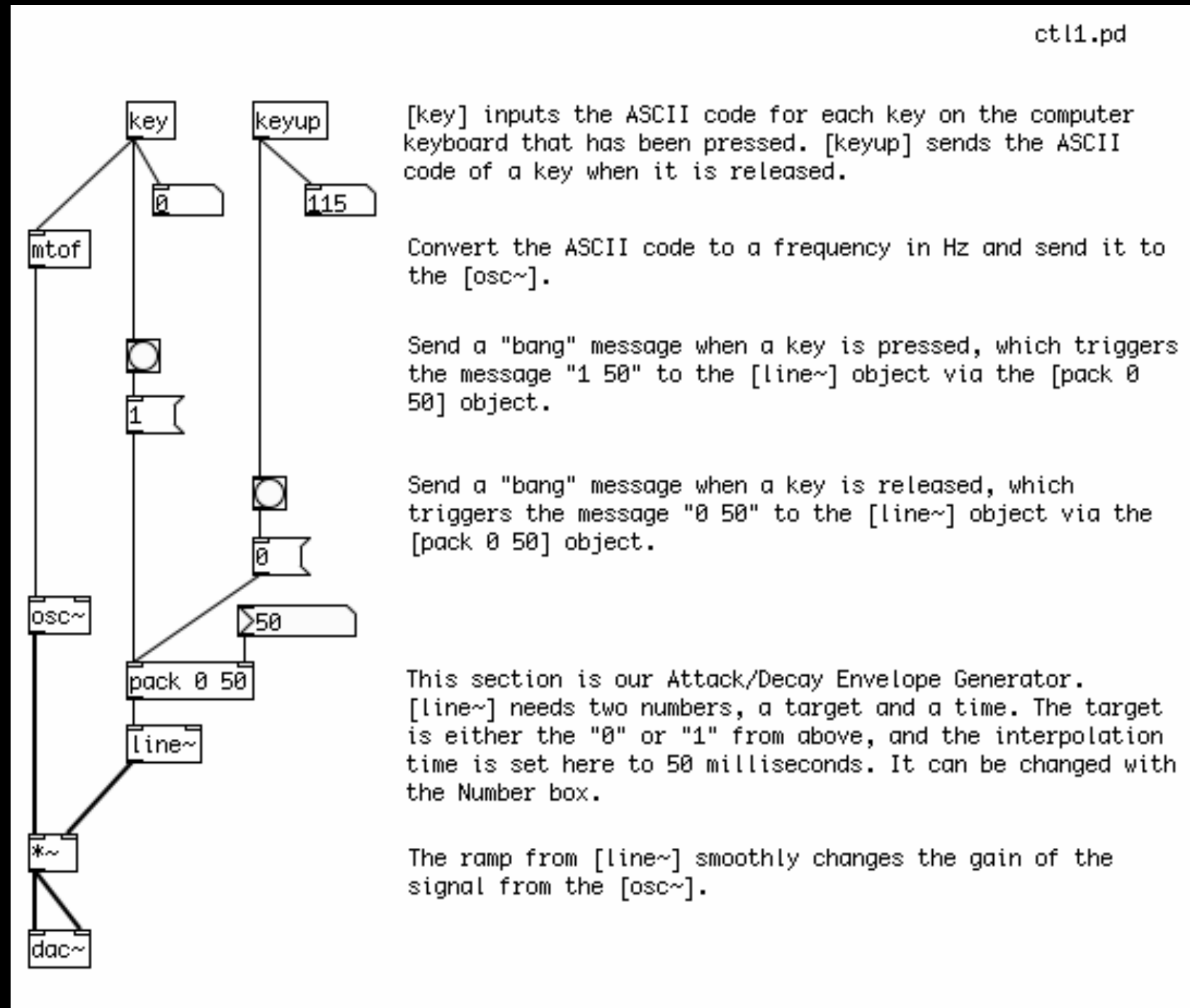
# INTRODUCTION

# Repository

[https://github.com/ImanolGo/  
SoundInteractionWorkshop](https://github.com/ImanolGo/SoundInteractionWorkshop)

Keyboard

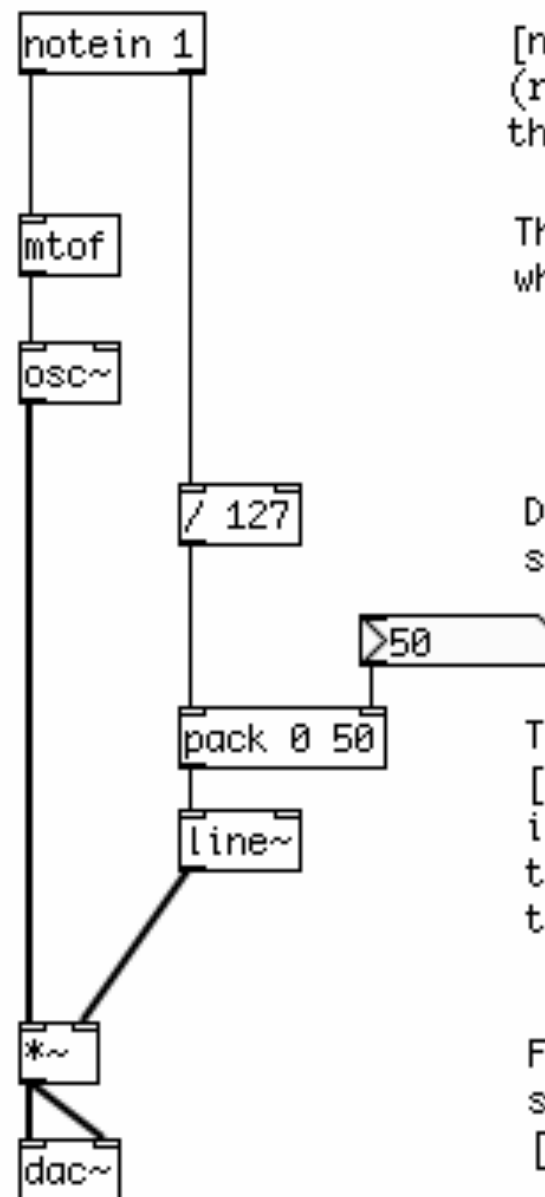
# Input from the Computer Keyboard



MIDI

# Input from a MIDI Keyboard

ctl2.pd



[notein] receives MIDI notes (left outlet) and Velocity (right outlet) from a MIDI keyboard. Use Preferences to set the proper MIDI devices to receive.

The MIDI Note output gets converted to a frequency in Hertz which controls the [osc~].

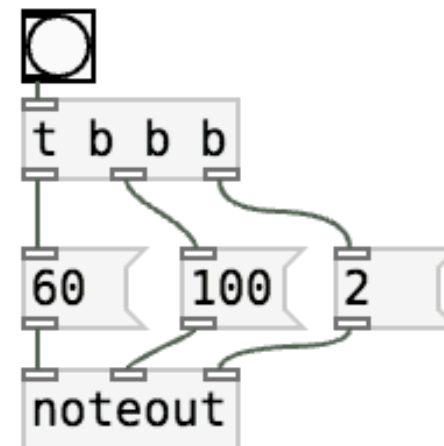
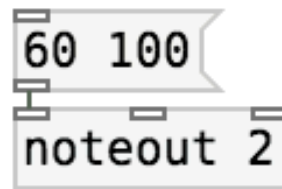
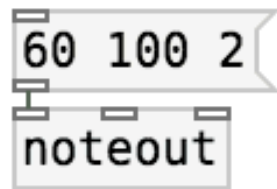
Divide the Velocity output (0-127) by 127 to get a new, scaled value between 0 and 1

This section is our Attack/Decay Envelope Generator. [line~] needs two numbers, a target and a time. The target is the scaled Velocity from above, and the interpolation time is set here to 50 milliseconds. It can be changed with the Number box.

Finally, multiply the audio from the oscillator by the scaled Velocity number, as it is interpolated by the [line~].

# [noteout]

Transmit MIDI notes

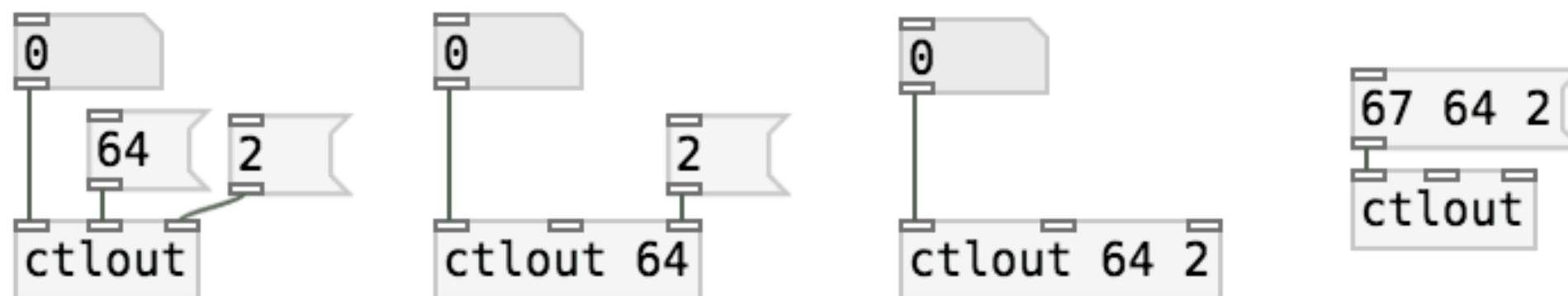


(all three above) middle-c with velocity 100 sent to channel  
2



# [ctlout]

Sends control messages to the MIDI port. See a MIDI specification chart for various controller numbers/values descriptions.



All these examples will transmit control no 64 on channel 2 (remember to click the message boxes to initialize).

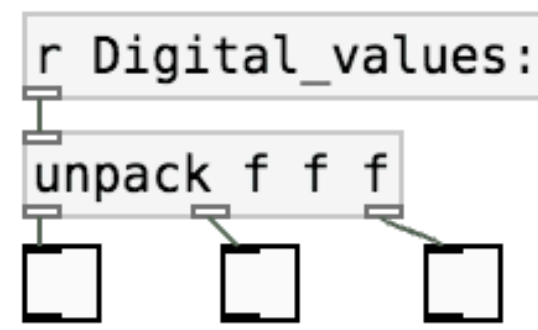
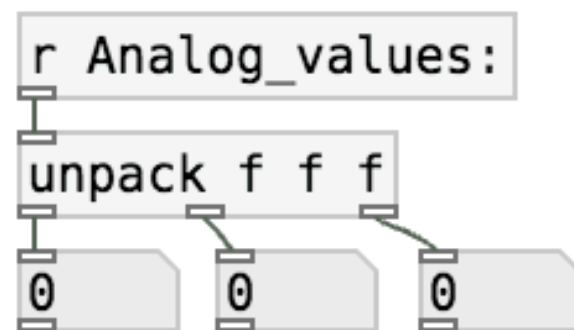
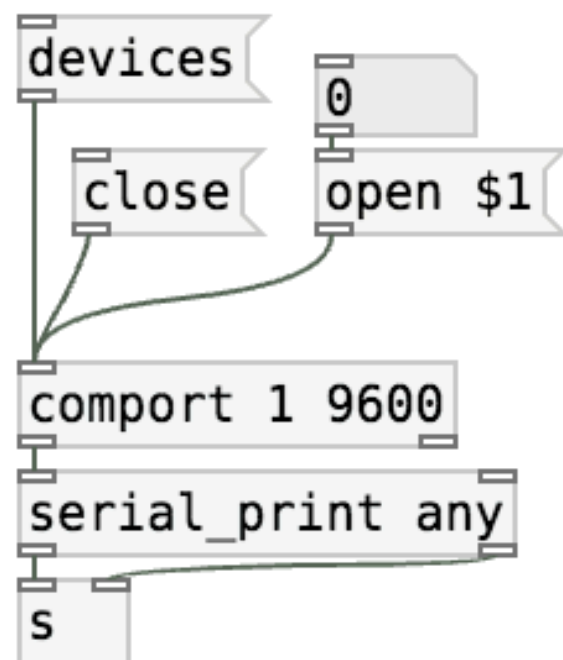
SERIAL

# [Comport]



# [serial\_print]

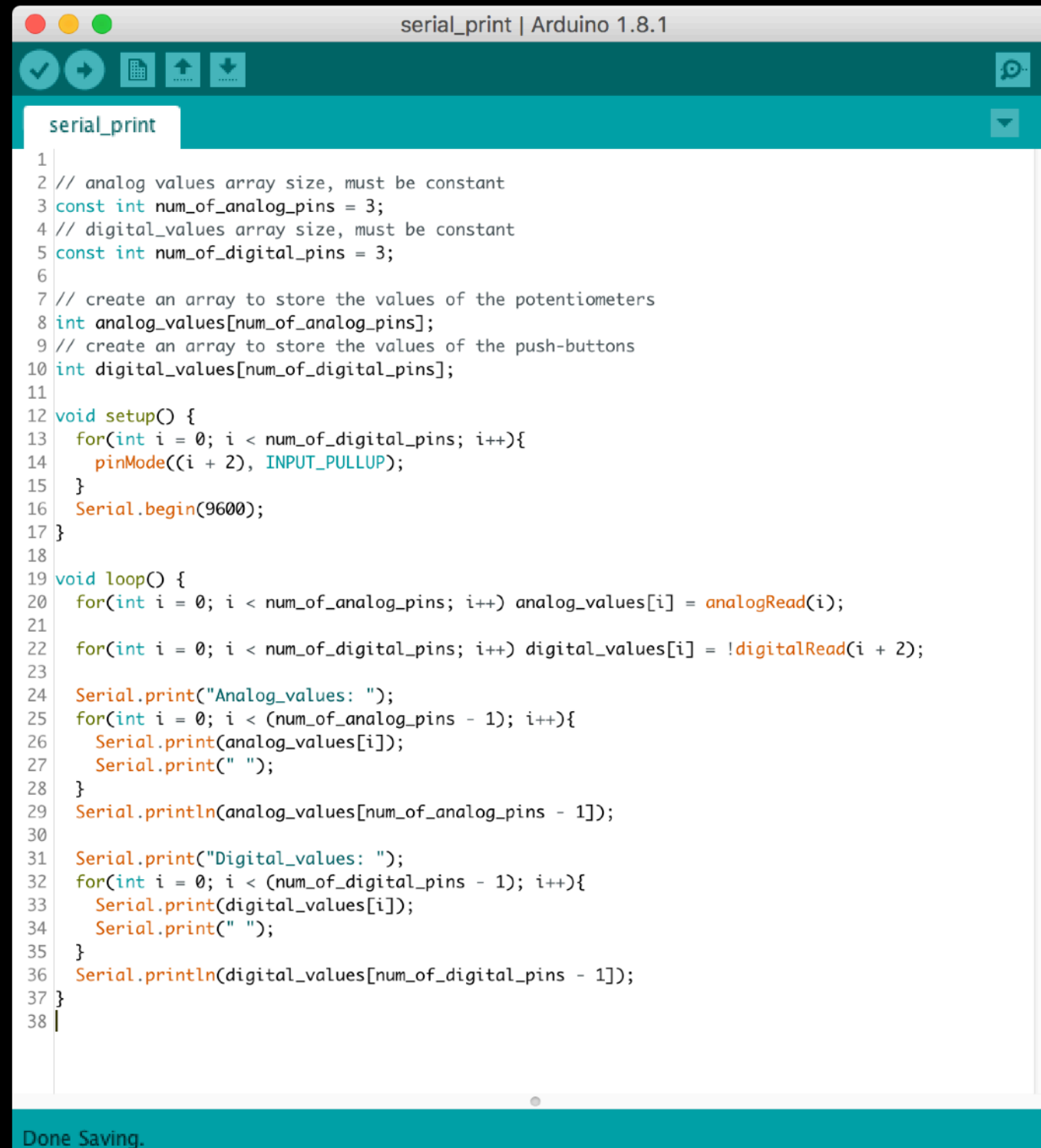
Abstraction that works with [comport] and Arduino's Serial.print() and Serial.println() functions.



# [serial\_print]

1. Always copy the serial\_print patch!
2. In Arduino use `Serial.print(value) + Serial.print(" ")`
3. A string must precede a value group
4. The last value of each group must be printed with `Serial.println()`
5. Take a look on `serial_print-help.pd` and `serial_print.ino`

# [serial\_print]



The screenshot shows the Arduino IDE interface with a file named 'serial\_print' open. The code is a C++ sketch for an Arduino Uno, using the Arduino 1.8.1 IDE. It defines two arrays: 'analog\_values' for 3 analog pins and 'digital\_values' for 3 digital pins. The 'setup' function initializes the digital pins (pins 2, 3, and 4) as inputs with pull-up resistors and starts the serial communication at 9600 baud. The 'loop' function reads the values from the analog and digital pins and prints them to the serial monitor. The analog values are printed as a single line, and the digital values are printed as a single line. The code is as follows:

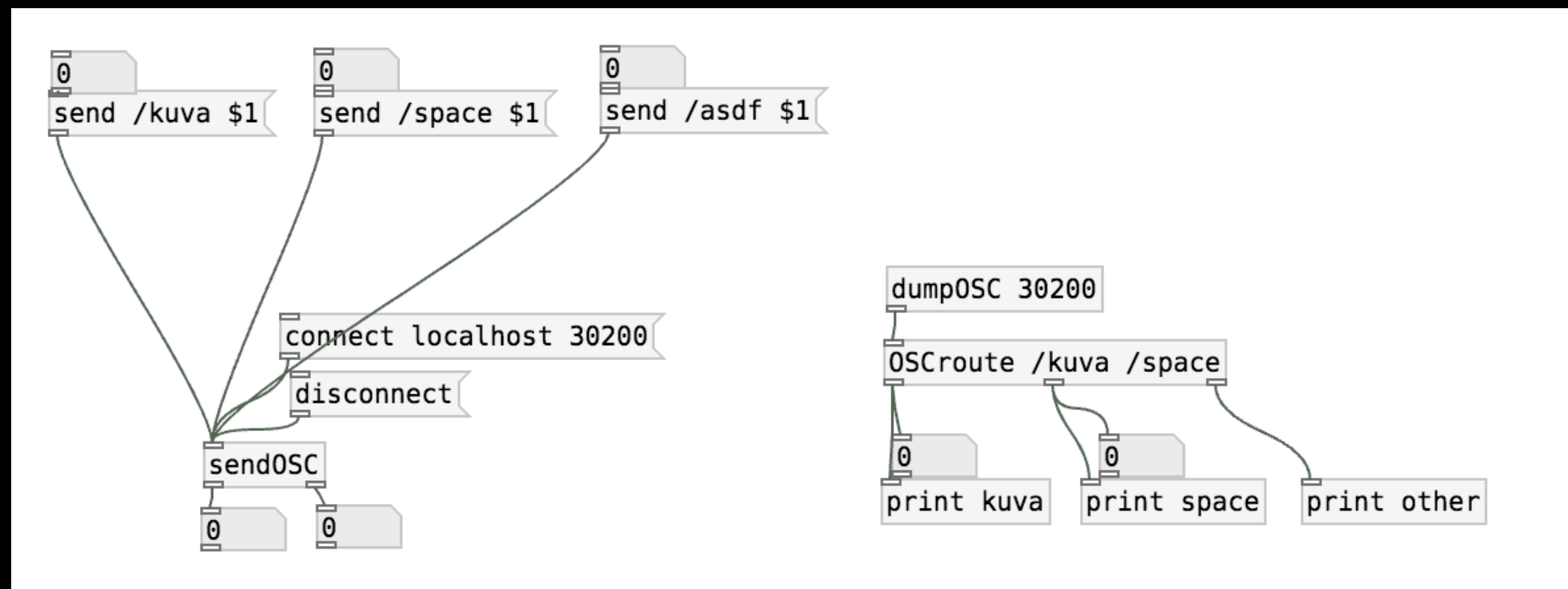
```
1
2 // analog values array size, must be constant
3 const int num_of_analog_pins = 3;
4 // digital_values array size, must be constant
5 const int num_of_digital_pins = 3;
6
7 // create an array to store the values of the potentiometers
8 int analog_values[num_of_analog_pins];
9 // create an array to store the values of the push-buttons
10 int digital_values[num_of_digital_pins];
11
12 void setup() {
13   for(int i = 0; i < num_of_digital_pins; i++){
14     pinMode((i + 2), INPUT_PULLUP);
15   }
16   Serial.begin(9600);
17 }
18
19 void loop() {
20   for(int i = 0; i < num_of_analog_pins; i++) analog_values[i] = analogRead(i);
21
22   for(int i = 0; i < num_of_digital_pins; i++) digital_values[i] = !digitalRead(i + 2);
23
24   Serial.print("Analog_values: ");
25   for(int i = 0; i < (num_of_analog_pins - 1); i++){
26     Serial.print(analog_values[i]);
27     Serial.print(" ");
28   }
29   Serial.println(analog_values[num_of_analog_pins - 1]);
30
31   Serial.print("Digital_values: ");
32   for(int i = 0; i < (num_of_digital_pins - 1); i++){
33     Serial.print(digital_values[i]);
34     Serial.print(" ");
35   }
36   Serial.println(digital_values[num_of_digital_pins - 1]);
37 }
38
```

Done Saving.

OSC

# Open Sound Control

OSC is a protocol for sharing data across networks and applications.





Now, is your turn!

# Questions?

**Imanol Gómez**

[imanolgomez.net](http://imanolgomez.net)

[yo@imanolgomez.net](mailto:yo@imanolgomez.net)