Center for
Human-Computer Interaction
University of Salzburg

# PHYSICAL COMPUTING

## (Basics of electronics, Arduino, Processing)

19.5.2019

Jakub Sypniewski

jakub.sypniewski@sbg.ac.at

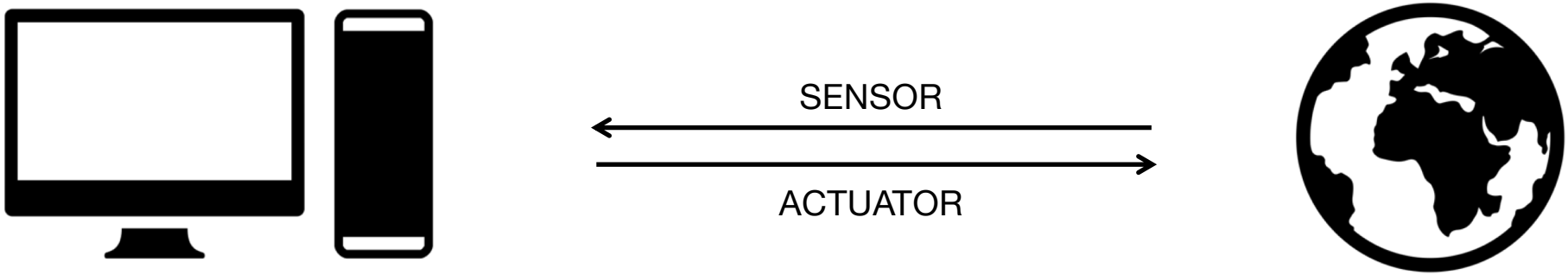
Center for
Human-Computer Interaction
University of Salzburg

# DISCLAIMER:

Not a computer scientist
or an electrical engineer

# PHYSICAL COMPUTING

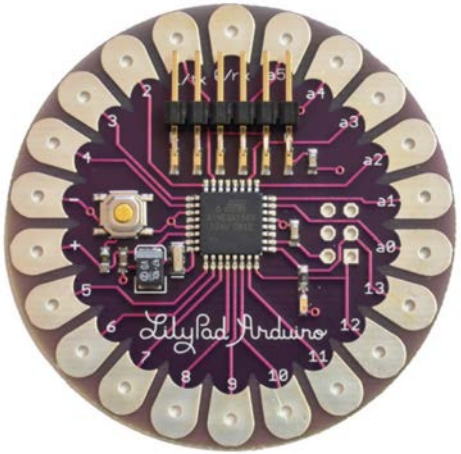
Center for
Human-Computer Interaction
University of Salzburg




AN INTERACTIVE SYSTEM

THE REAL WORLD

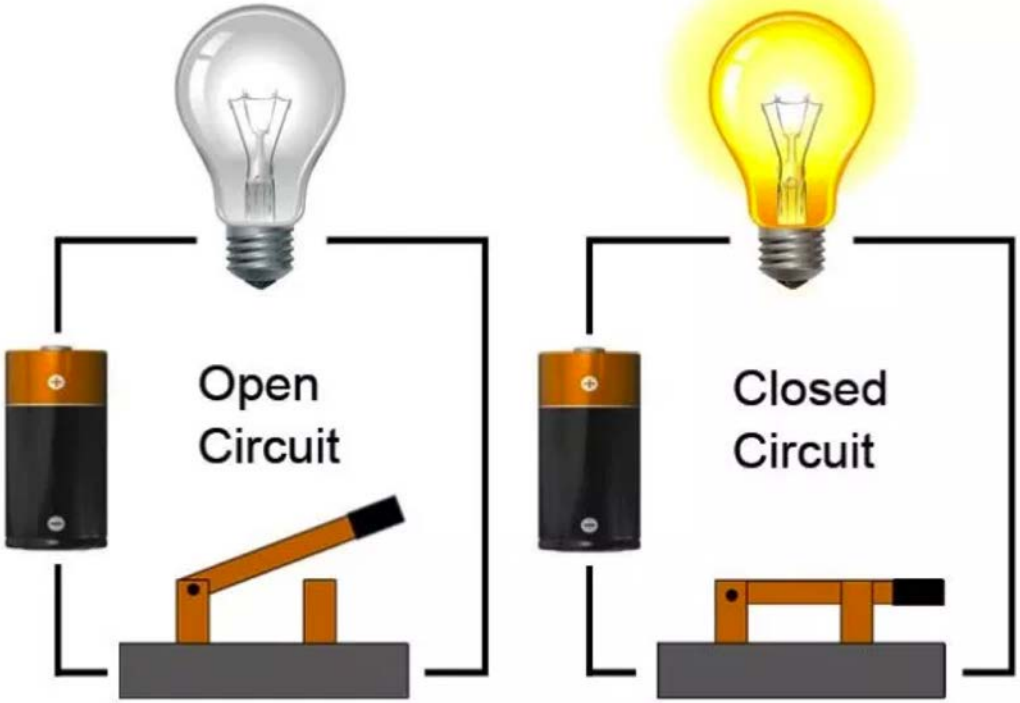
Center for
**Human-Computer** Interaction
University of Salzburg




# ELECTRICAL CIRCUIT

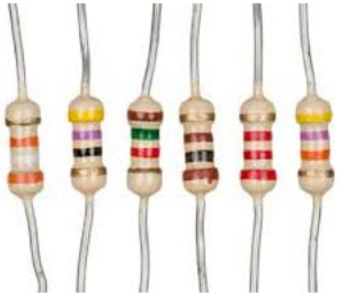
Center for
**Human-Computer** Interaction
University of Salzburg



# ELECTRONIC COMPONENTS

Center for
Human-Computer Interaction
University of Salzburg

Switch



Resistor

Variable Resistor (Potentiometer)

Light-Dependent Resistor (LDR)

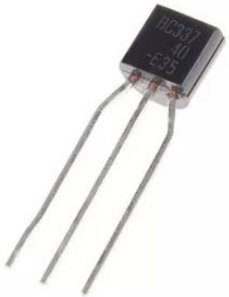
Relay

Servo Motor

Capacitor

Diode

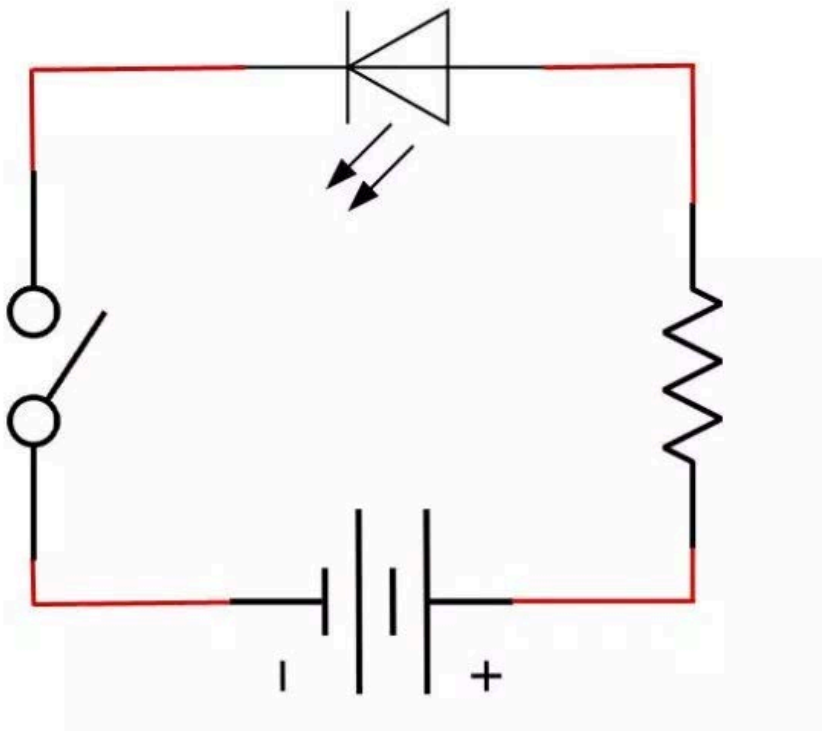
Light-Emitting Diode (LED)



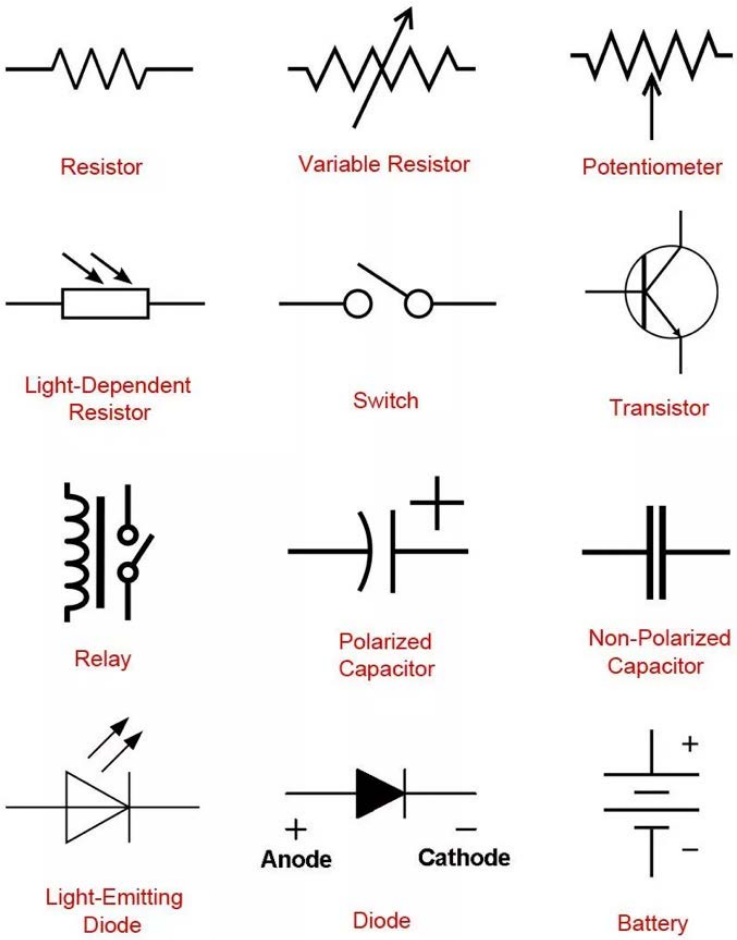
Transistor

Integrated Circuit (IC)

DC Motor




Center for
**Human-Computer** Interaction
University of Salzburg



# RESISTANCE (A.K.A. HOW NOT TO BURN THINGS)

Center for Human-Computer Interaction
University of Salzburg

- **Ohm's Law - Resistance (R) = Voltage (V) / Current (I)**
  - Resistance is measured in Ohms (Ω)
  - Voltage is measured in volts (V)
  - Current is measured in amps (A)

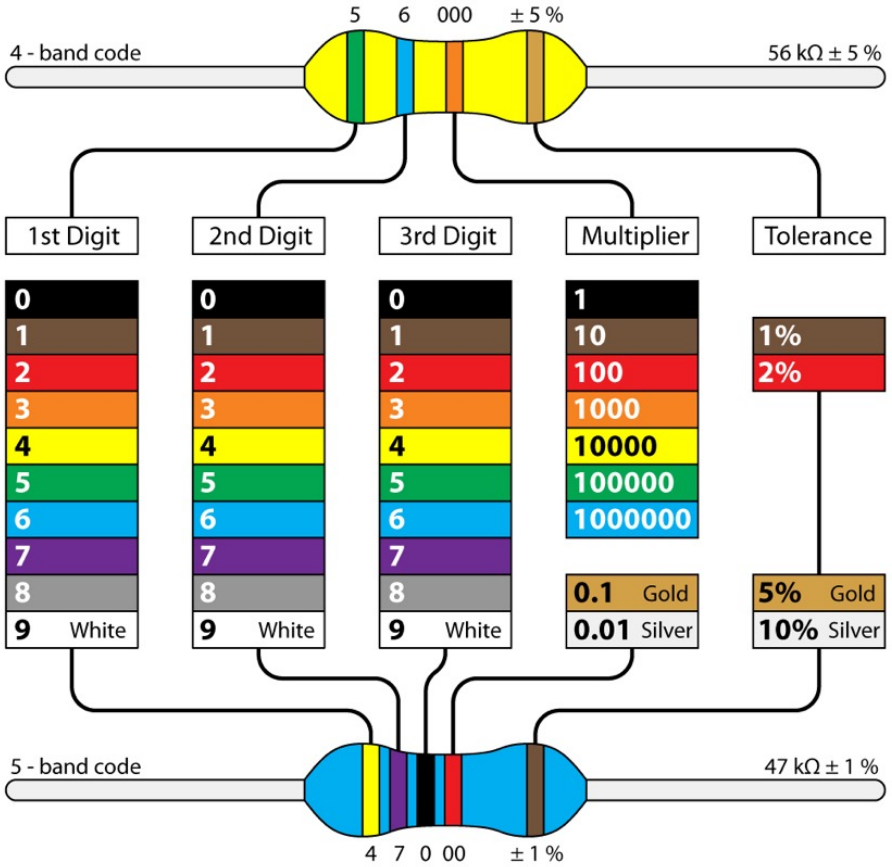
$$R = \frac{V_{Bat} - V_{LED}}{I_{LED}}$$

  - Battery has 9V
  - LED needs 2V
  - LED has current rating of .02A (20mA)

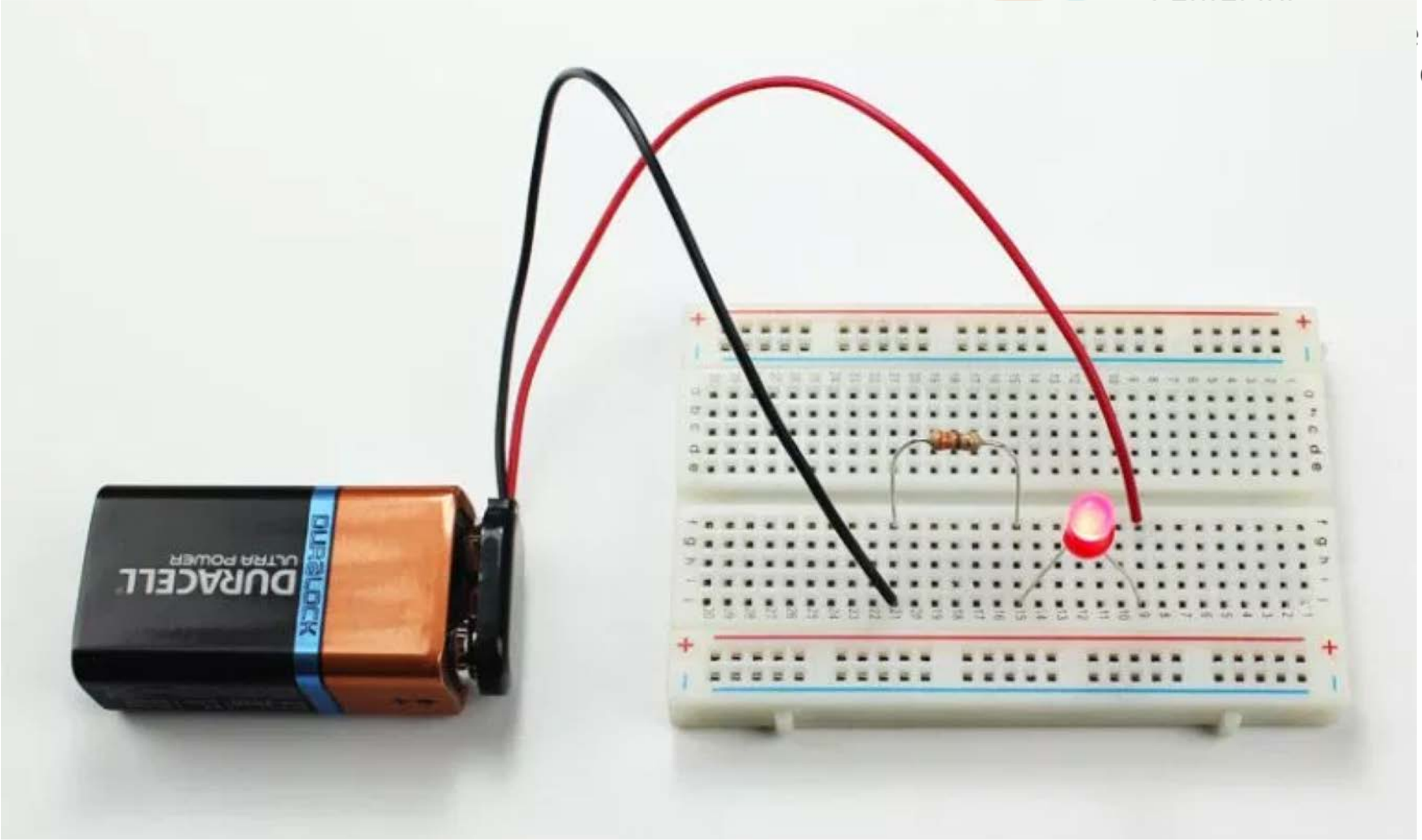
$$350 = \frac{9V - 2V}{.02A}$$

Center for
**Human-Computer** Interaction
University of Salzburg

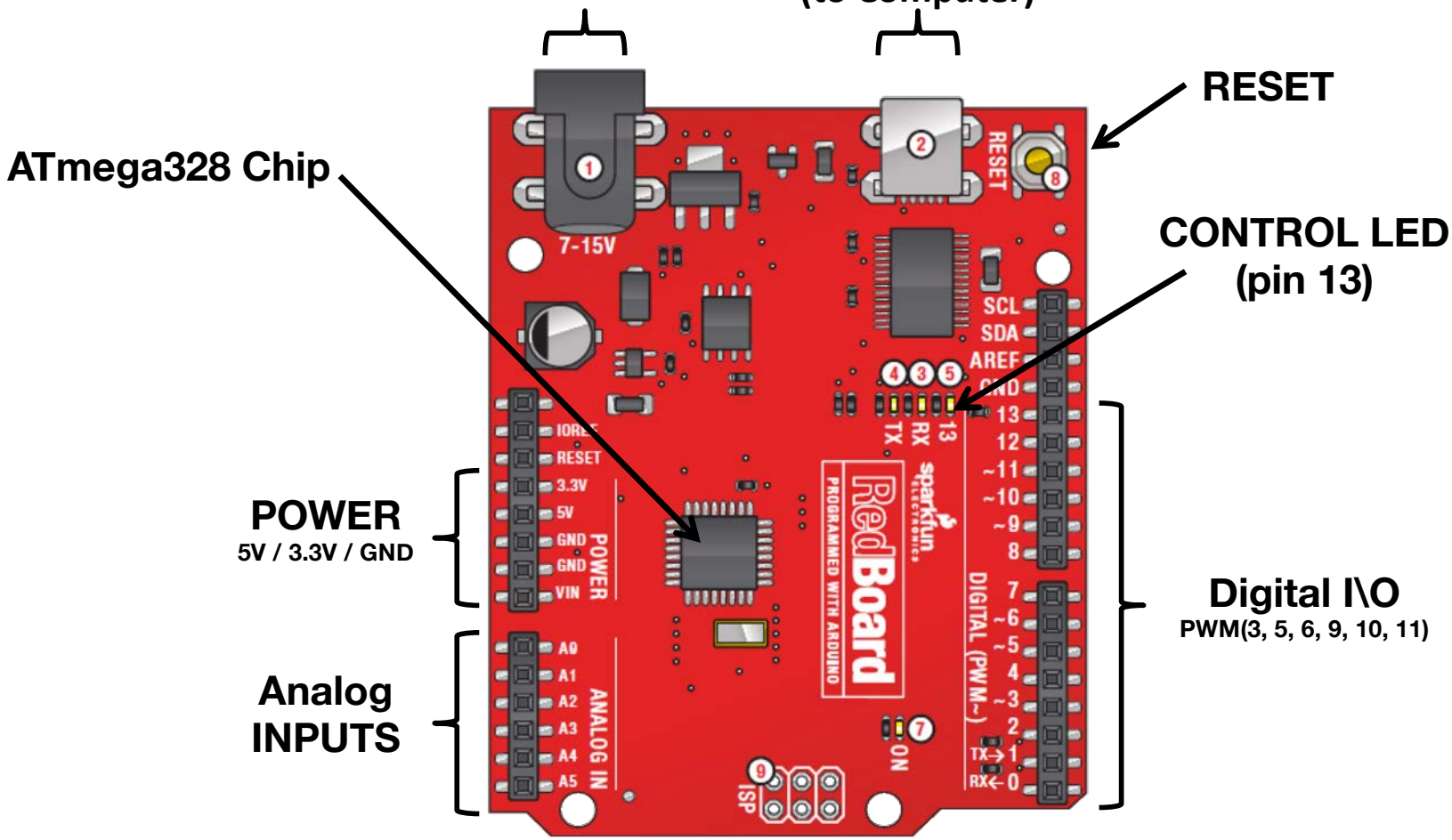
# Resistor colour code




Center for
raction
of Salzburg



Center for
Human-Computer Interaction
University of Salzburg

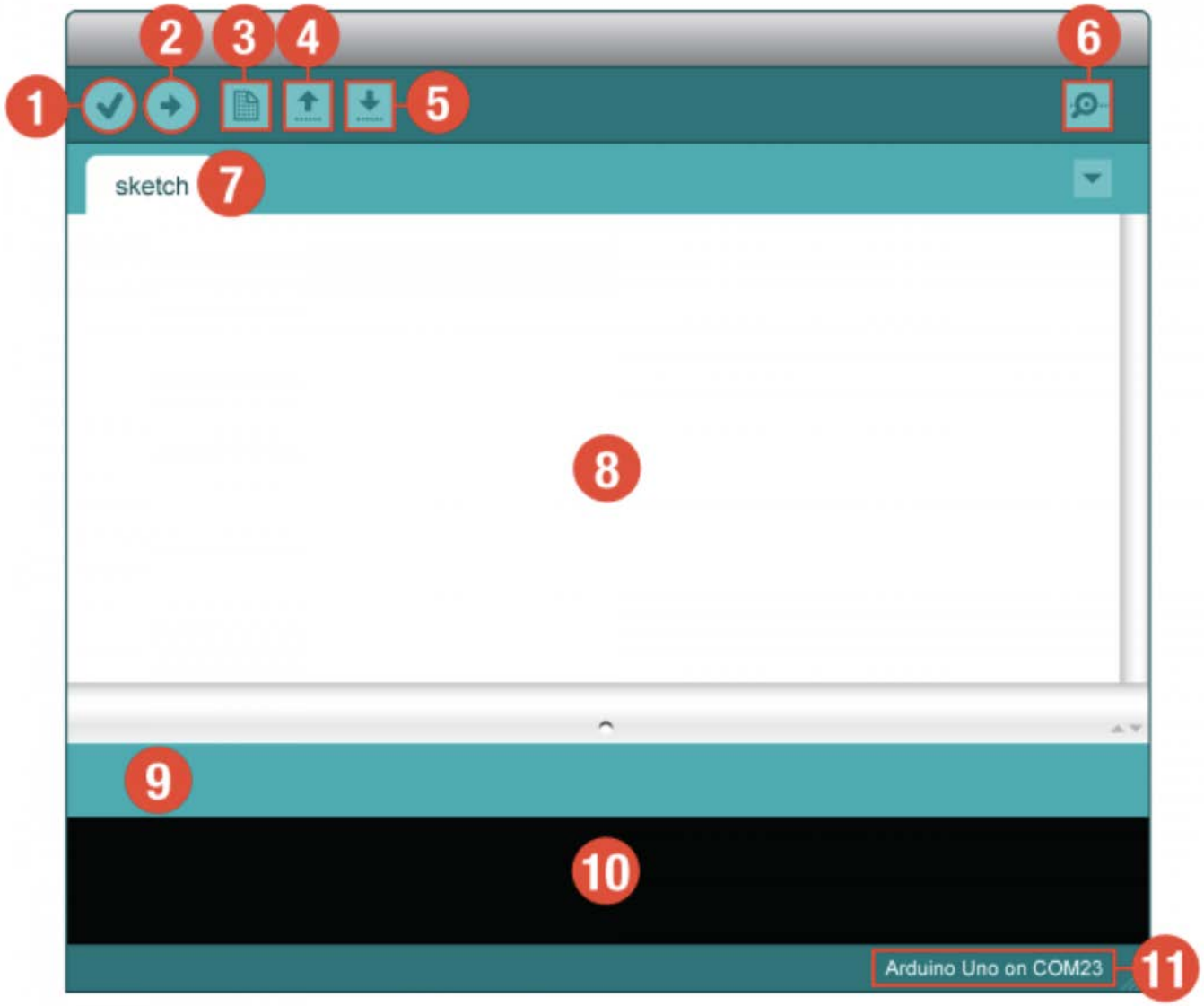


Center for
Human-Computer Interaction
Salzburg

# ARDIUNO IDE

Center for
**Human-Computer** Interaction
University of Salzburg

www.arduino.cc/en/Main/Software




Center for
**Human-Computer** Interaction
University of Salzburg

## THE ARDUINO IDE

1. Verify
2. Upload
3. New
4. Open
5. Save
6. Serial Monitor
7. Sketch Name
8. Code Area
9. Message Area
10. Text Console
11. Board and Serial Port

```
void setup() {
  // put your setup code here, to run once:

}

void loop() {
  // put your main code here, to run repeatedly:

}
```

New

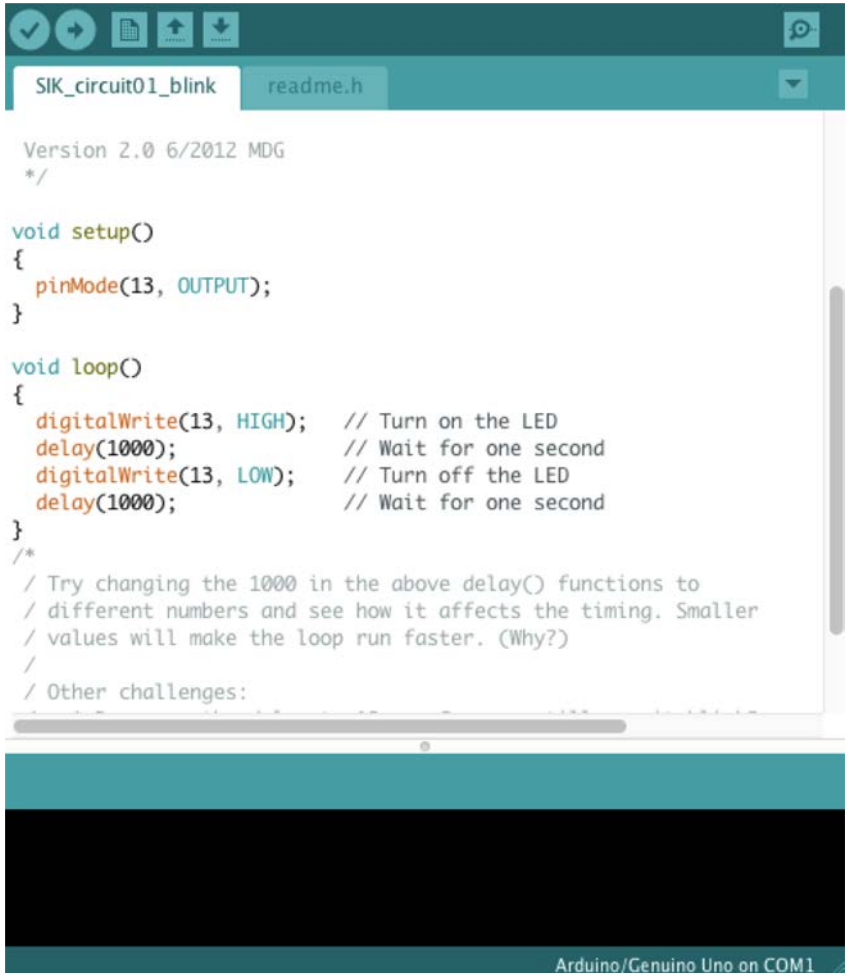
sketch_may04a §

```
int buttonPin = 3;


void setup()
{
  Serial.begin(9600);
  pinMode(buttonPin, INPUT);
}


void loop()
{
  if (digitalRead(buttonPin) == HIGH)
    Serial.write('H');
  else
    Serial.write('L');

  delay(1000);
}
```

10 Arduino/Genuino Uno on COM1

Center for
**Human-Computer** Interaction
University of Salzburg

# BLINK




Center for
**Human-Computer** Interaction
University of Salzburg

1. Plug in your RedBoard
2. Make sure you're connected to the right COM port

   *tools > port*

   and that you have the right board selected

   *tools > board > Arduino Uno*

3. Type in the code on the left. Skip things after // and between /* and */
4. Verify your code
5. If no error, hit 'upload'
6. See what happens to the control LED on your board

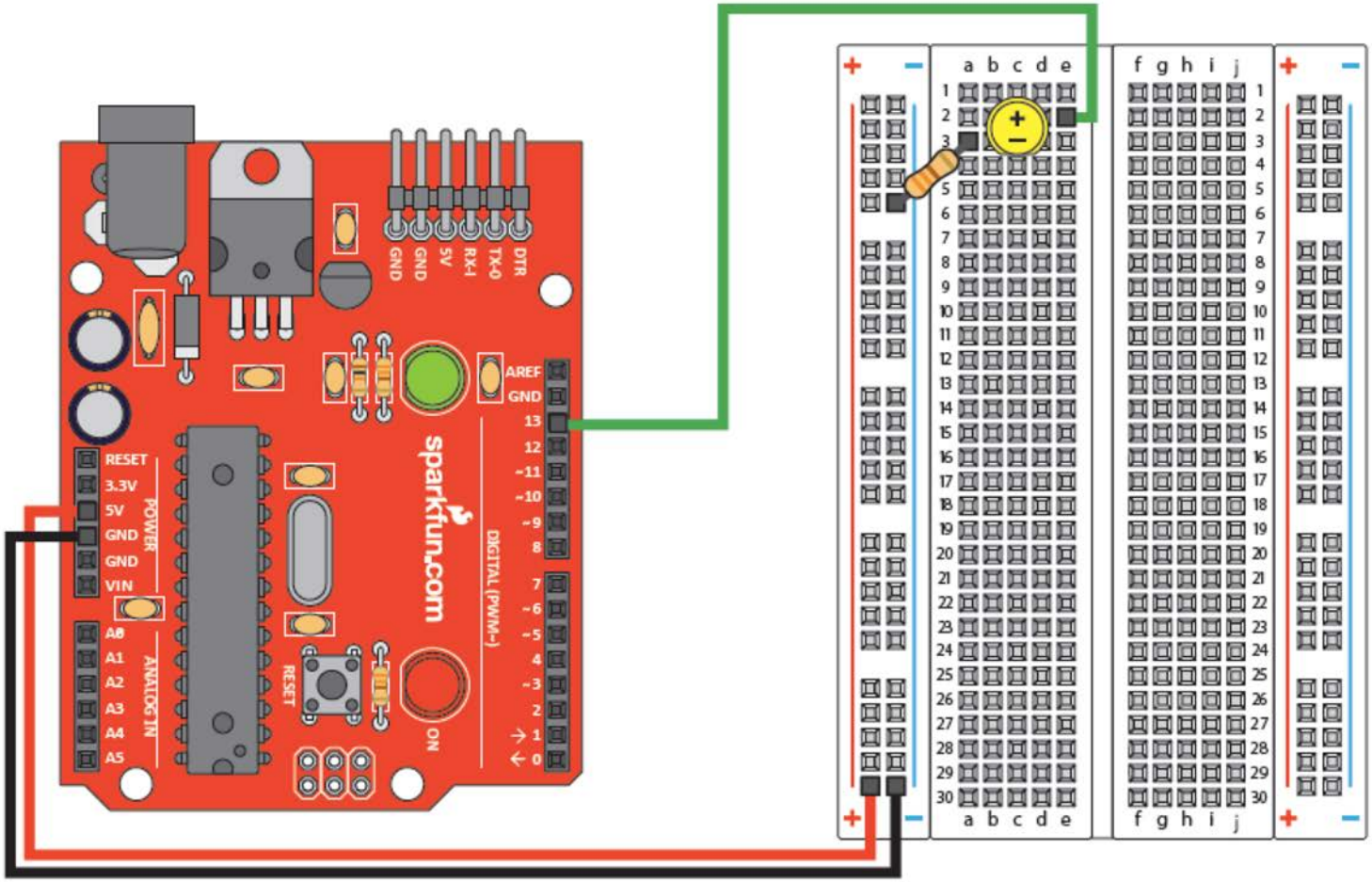
Center for
**Human-Computer** Interaction
University of Salzburg

**Needed:**

1 LED

1 330 Ω Resistor

Jumper cables



```
void setup(){
      pinMode(13, OUTPUT);
      }

void loop(){
      digitalWrite(13, HIGH);
      delay(1000);
      digitalWrite(13, LOW);
      delay(1000);
      }
```

# SIK EXAMPLES

Center for
Human-Computer Interaction
University of Salzburg

Download the zip:

https://www.sparkfun.com/sikcode (direct download)

1. Unzip >
2. Find the Arduino Folder on your PC > Go to 'Examples' >
3. Drop the unzipped folder in the Examples

(Restart Arduino)

# PHOTO RESISTOR

Center for
Human-Computer Interaction
University of Salzburg

Open:

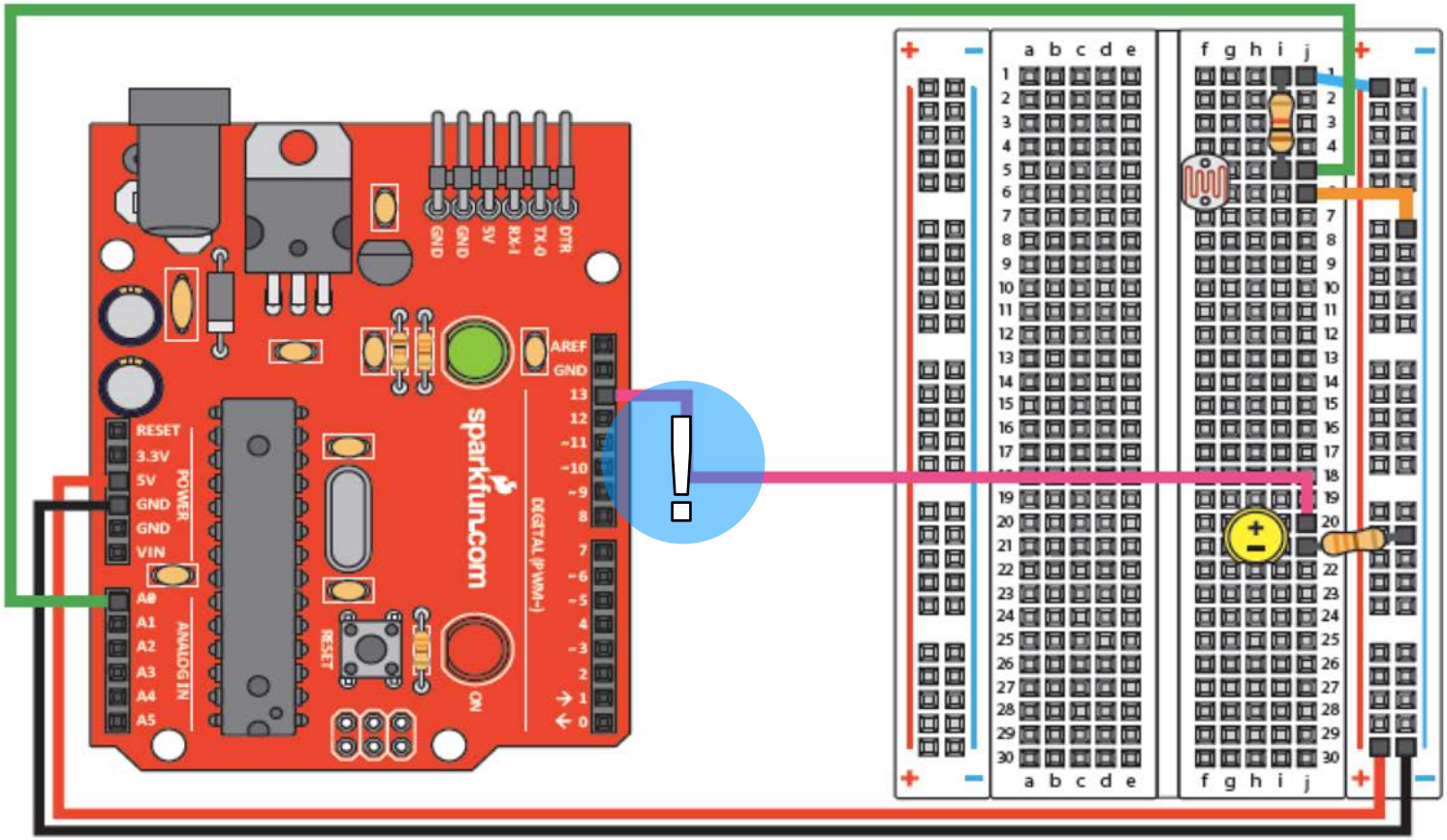
file > examples > SIK Guide > Circuit 6: Photoresistor

Center for
**Human-Computer** Interaction
University of Salzburg

**Needed:**

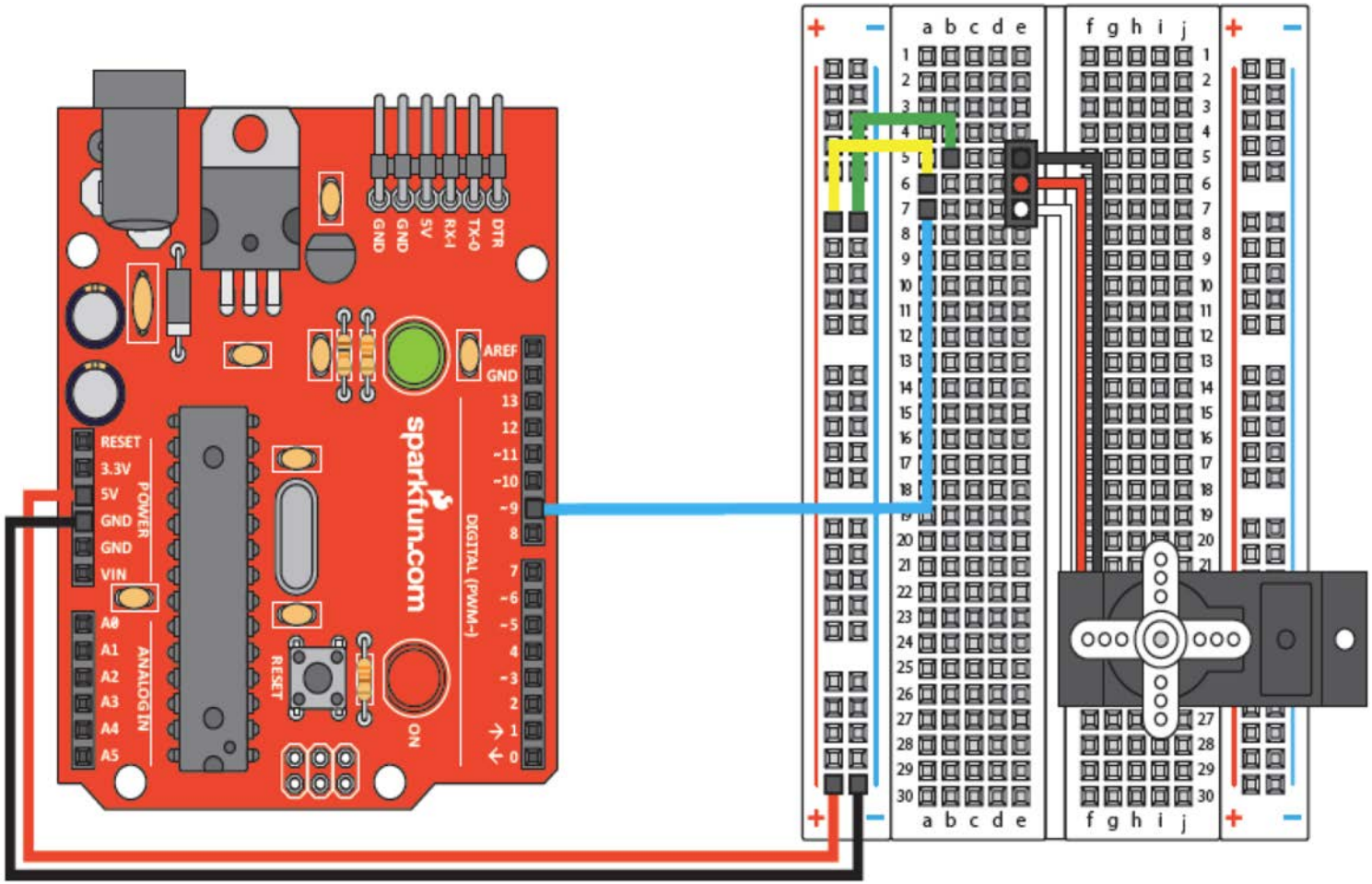
- Previous Blink setup **(to Pin9!)**
- Photo resistor
- 10k resistor
- Jumper cables

The voltage is divided over two resistors (the photo resistor and the 10k resistor)



```
void loop(){
      lightLevel = analogRead(sensorPin);
      Serial.print(lightLevel);


      calibratedlightLevel = map(lightLevel, 0, 1023, 0, 255);

      Serial.print("\t");
      Serial.print(calibratedlightLevel);
}
```

Center for
**Human-Computer** Interaction
University of Salzburg




**Needed:**

- Servo
- Jumper cables

```
#include <Servo.h>

Servo servo1;

void setup(){
     servo1.attach(9, 900, 2100);
}


…
```

```
void loop(){
    int position;

    servo1.write(90);
    delay(1000);
    servo1.write(180);
    delay(1000);
    servo1.write(0);
    delay(1000);

…
```

```
...
        for(position = 0; position < 180; position += 2)  {
            servo1.write(position);
            delay(20);
}

        for(position = 180; position >= 0; position -= 1){
            servo1.write(position);
            delay(20);
        }
}
```

Center for
Human-Computer Interaction
University of Salzburg

# PROCESSING

# WHAT IS PROCESSING?

Center for
Human-Computer Interaction
University of Salzburg

Very basic programming language with a focus on *visually oriented applications*.

Based on Java.

Very similar to the Arduino IDE, with additional libraries for Arduino/Pi integration (Serial.library / Firmata)

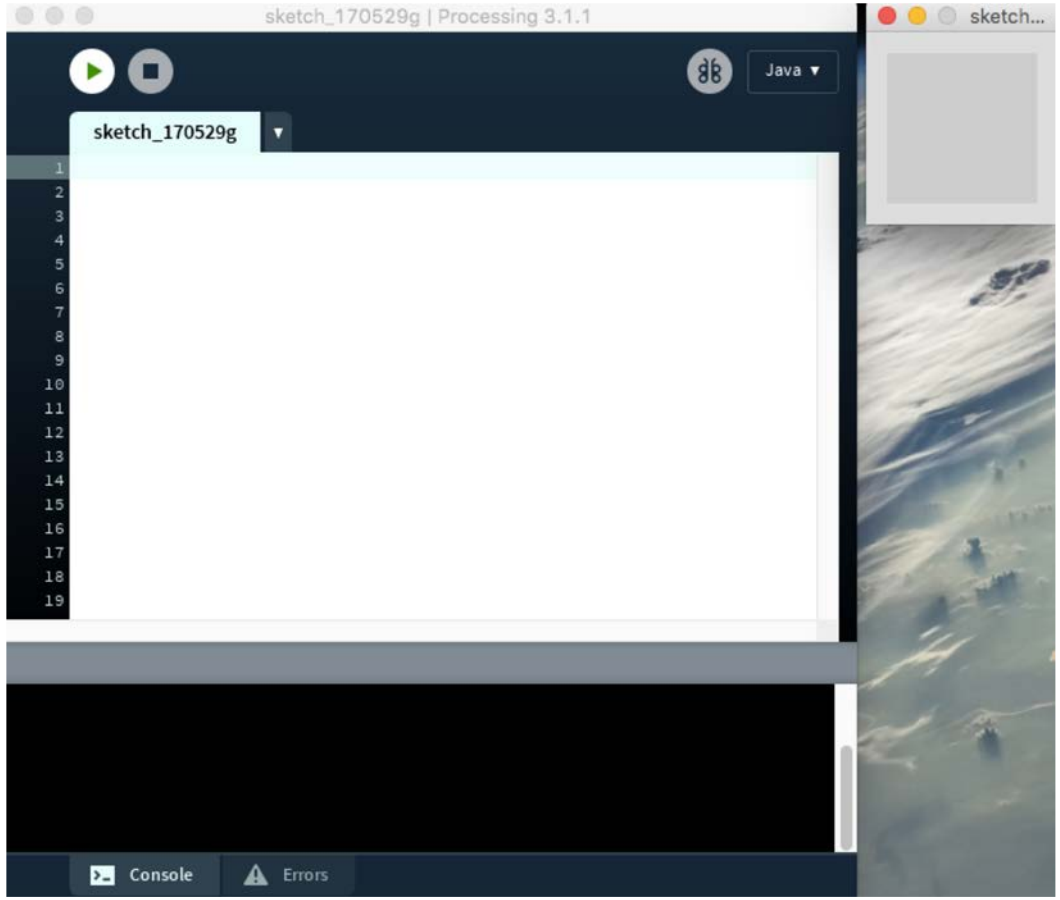
# WHY PROCESSING?

Center for
Human-Computer Interaction
University of Salzburg

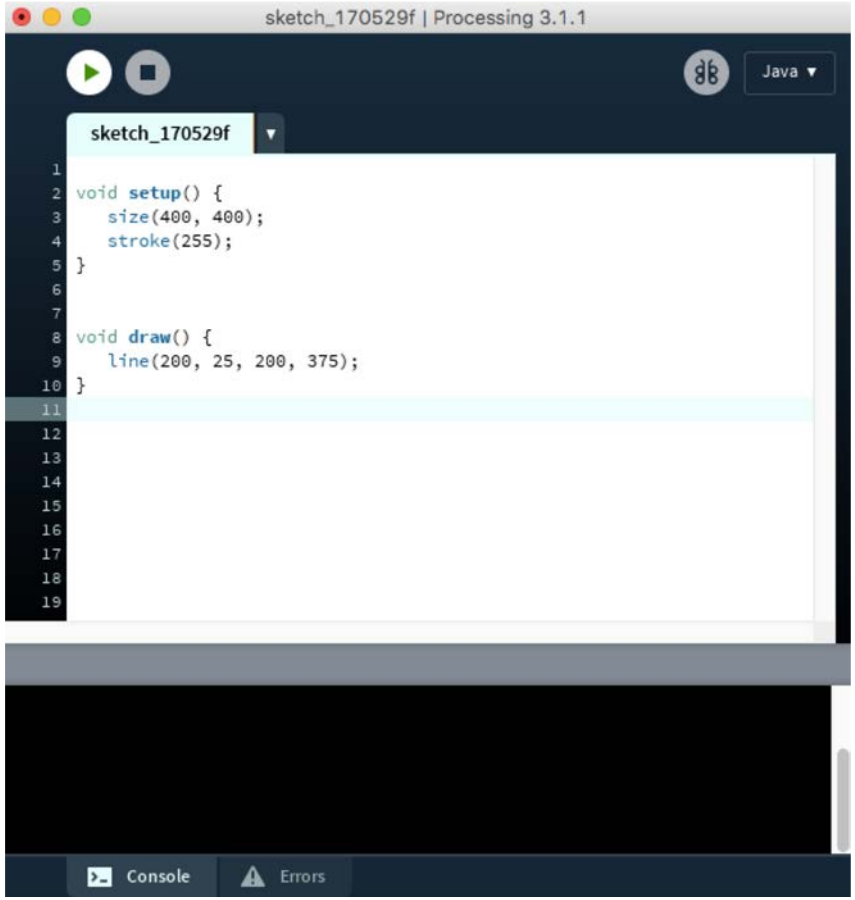
- Easy to get into
- Runs of most OS's
- Super-supportive community
- Lots of options to expand
- Able to connect via serial
- Can be exported as an executable

# LET'S SKETCH

Center for
**Human-Computer** Interaction
University of Salzburg



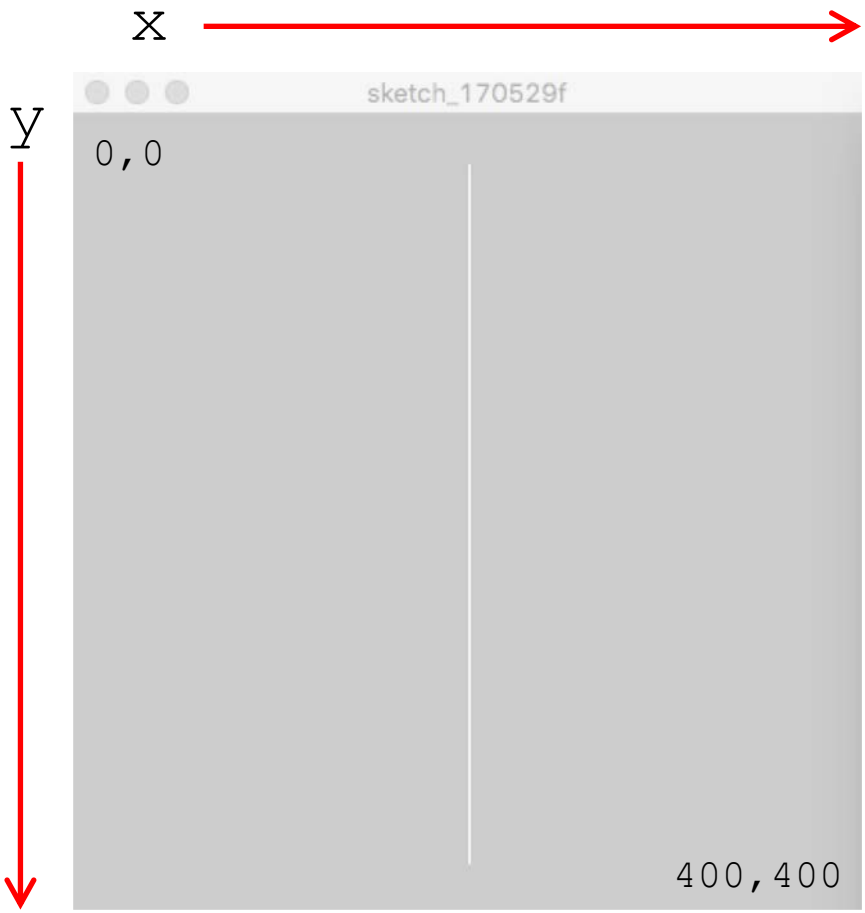
# A LINE




```
void setup() {
        size(400, 400);
        stroke(255);
}

void draw() {
        line(200, 25, 200, 375);
}
```

Center for
Human-Computer Interaction
University of Salzburg

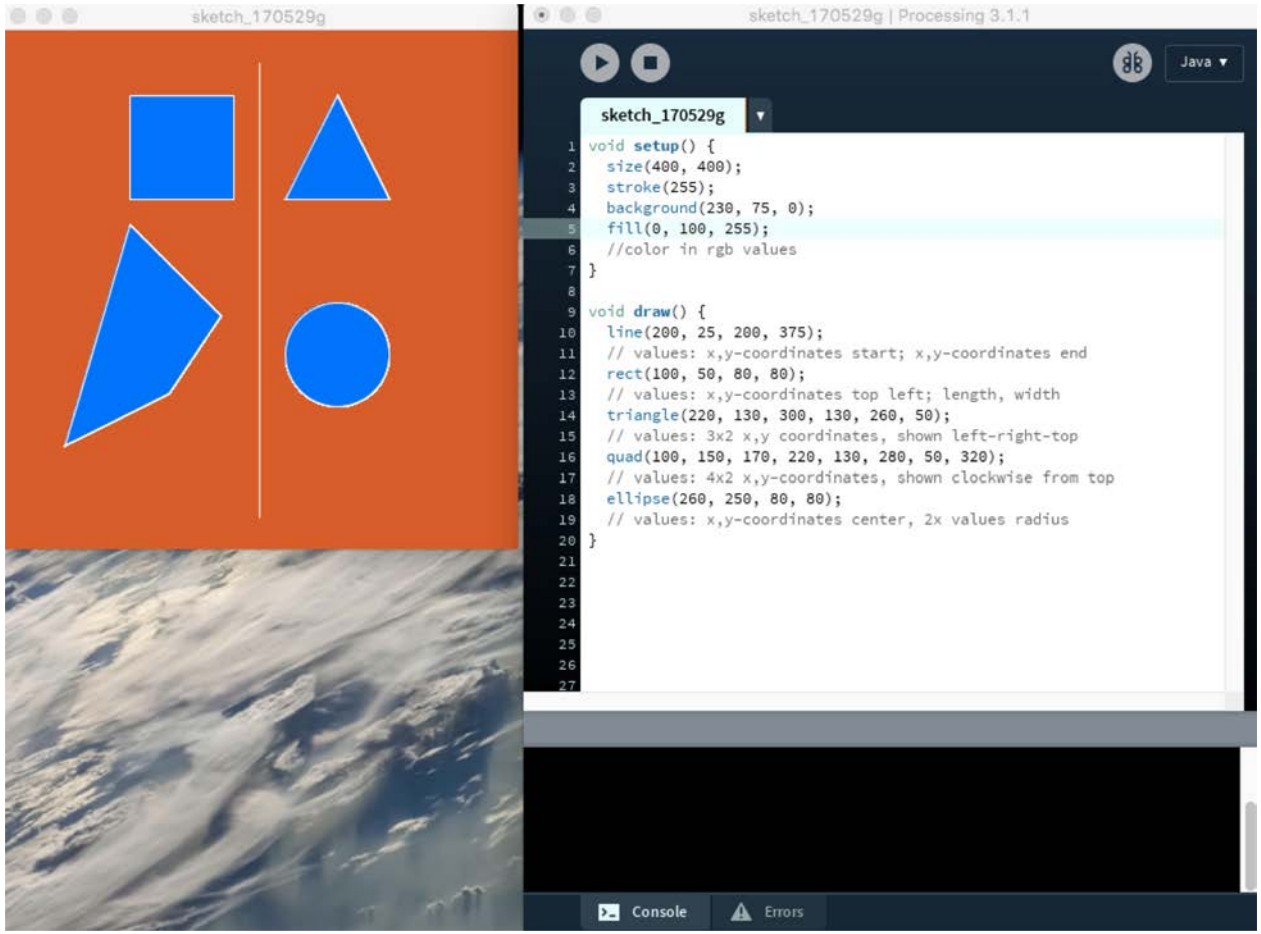
Center for
Human-Computer Interaction
University of Salzburg

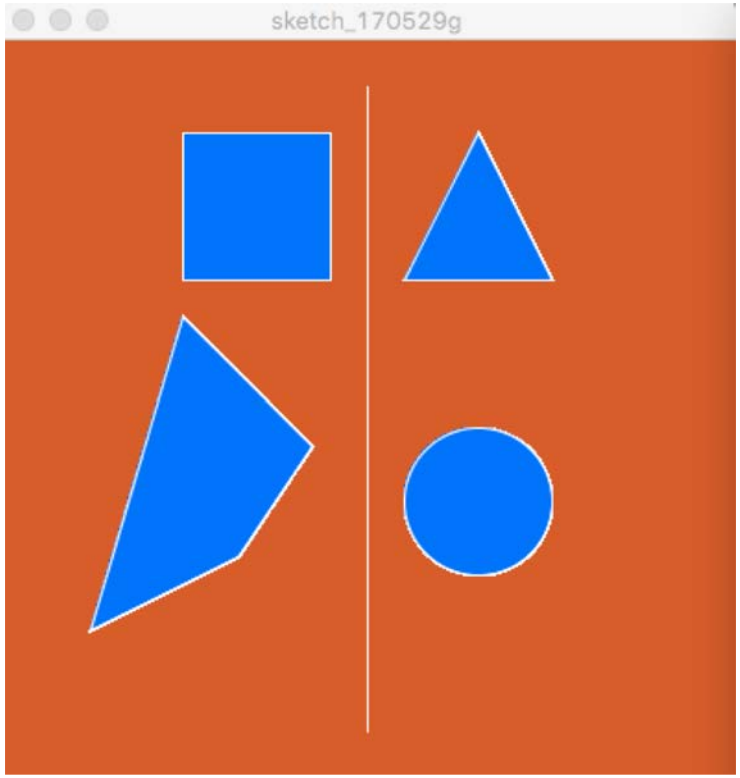



```
void setup() {
        size(400, 400);
        stroke(255);
}

void draw() {
        line(200, 25, 200, 375);
}
```

# SOME SHAPES

Center for
**Human-Computer** Interaction
University of Salzburg

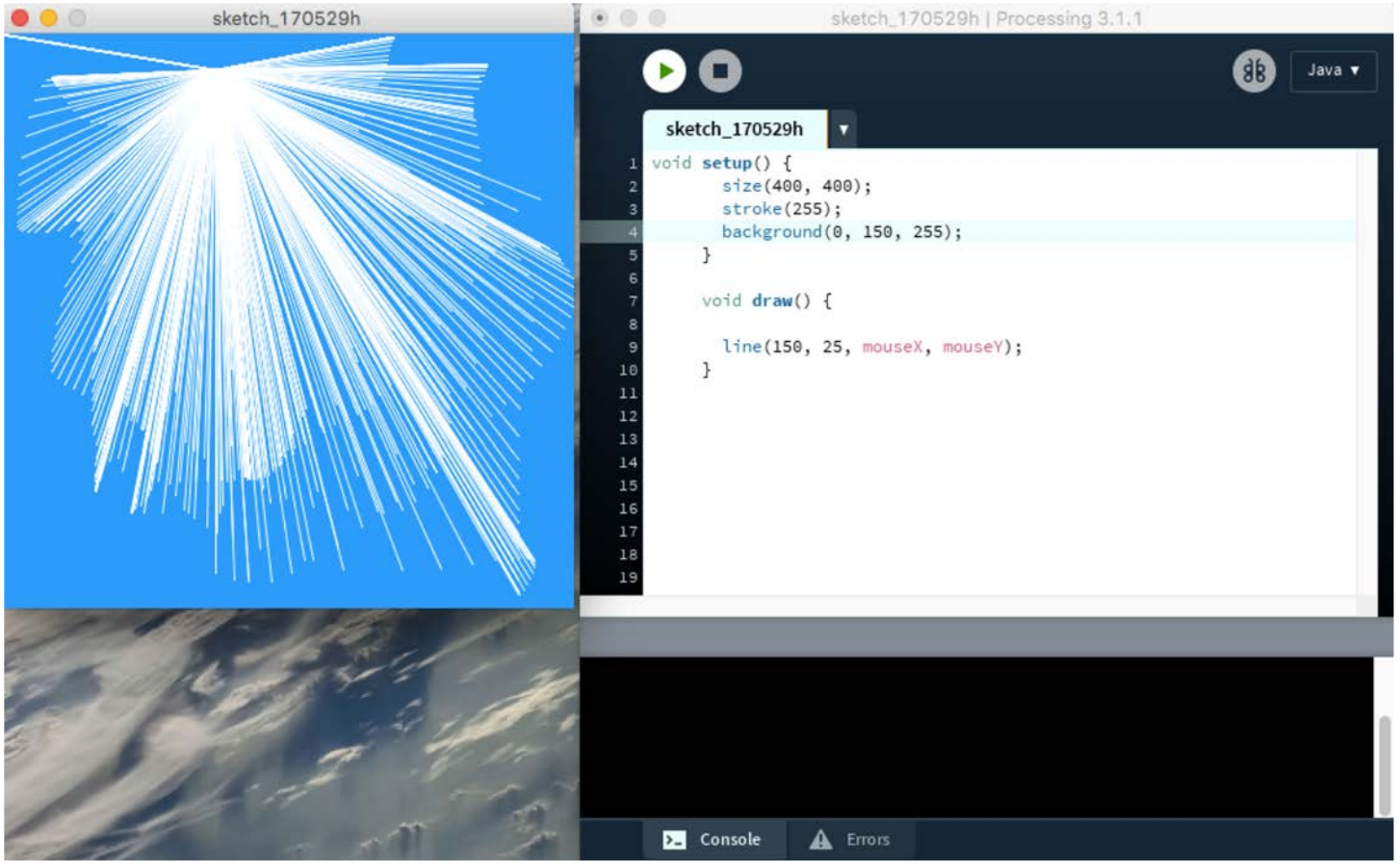


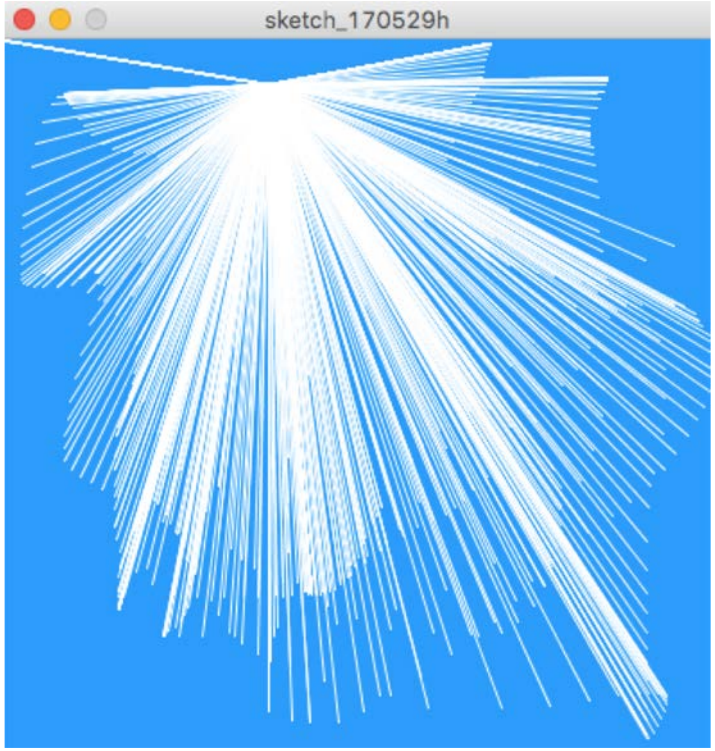



```
void setup() {
   size(400, 400);
   stroke(255);
   background(230, 75, 0);
   fill(0, 100, 255);
}

void draw() {
   line(200, 25, 200, 375);
   rect(100, 50, 80, 80);
   triangle(220, 130, 300, 130, 260, 50);
   quad(100, 150, 170, 220, 130, 280, 50, 320);
   ellipse(260, 250, 80, 80);
}
```

# MOVING THE LINE

Center for
**Human-Computer** Interaction
University of Salzburg






```
void setup() {
   size(400, 400);
   stroke(255);
   background(0, 150, 255);
}
void draw() {
      line(150, 25, mouseX, mouseY);
}
```

# ASSIGNMENT (15-20 MIN)

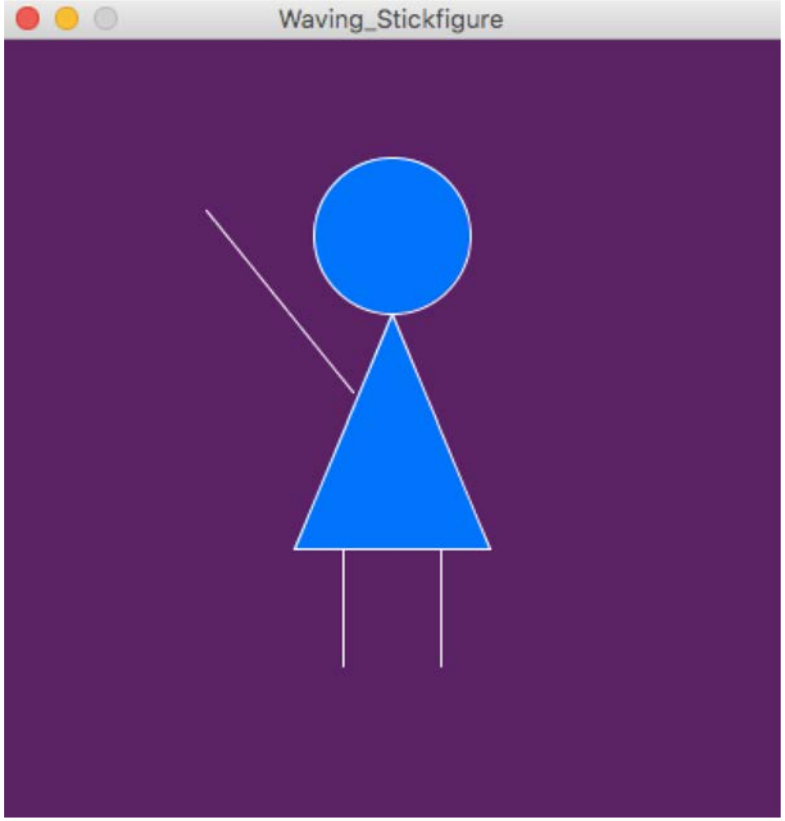
Center for
Human-Computer Interaction
University of Salzburg

Draw a stick figure that waves when you move the mouse

Bonus: make it wave automatically

# POSSIBLE SOLUTION:

Center for
Human-Computer Interaction
University of Salzburg




```
void setup() {
   size(400, 400);
   stroke(255);
   fill(0, 100, 255);
}
void draw() {
   background(100, 0, 100);
   triangle(150, 260, 250, 260, 200, 140);
   ellipse(200, 100, 80, 80);
   line(175, 260, 175, 320);
   line(225, 260, 225, 320);
   line(mouseX, mouseY, 180, 180);
}
```

# USING THE REFERENCE / COMMUNITY

Center for
Human-Computer Interaction
University of Salzburg

https://processing.org/reference/ (or *help* > *find in reference*) for the basics of Processing

https://forum.processing.org/two/ (new forum)

https://forum.processing.org/one/ (previous forum)

And most importantly: *file* > *examples* > *basics*

# ADDING AN ARDUINO

Center for
Human-Computer Interaction
University of Salzburg

If you have the board with you, open up your Arduino IDE and add the following code*:

```
void setup() {
Serial.begin(9600);
}

void loop() {
Serial.println("Hello, world!");
delay(1000);
}
```

Add your board and upload.

*courtesy of Sparkfun

Switch to Processing.

*Sketch > Import Library > Serial*

`import processing.serial.*;` should appear

Add the following code after your import statement:

```
Serial myPort;
String val;

void setup() {
   String portName = Serial.list()[0];
   //change the 0 to the COM port you're actually using
   myPort = new Serial(this, portName, 9600);
}
```

For mac:

```
import processing.serial.*;
Serial myPort;
printArray(Serial.list());
```

Add the following code as your void draw:

```
void draw() {
    if ( myPort.available() > 0) {
        val = myPort.readStringUntil('\n');
    }

    println(val);
}
```

# FROM PROCESSING TO ARDUINO

Center for
Human-Computer Interaction
University of Salzburg

Start a new Processing sketch, and **import the serial library**.
Add the following code:

```
Serial myPort;

void setup() {
   size(200,200);
   String portName = Serial.list()[0];
   //change to fit COM port
   myPort = new Serial(this, portName, 9600);
}
```

Add:

```
void draw() {
     if (mousePressed == true) {
          myPort.write('1');
     }

     else {
          myPort.write('0');
     }
}
```

Hit ‘Run’ and see what happens when you click your mouse in the grey area.

Now, open your Arduino IDE (and, if you want to, stick an LED in pin13 and GND of your board).

Declare the following global variables before your setup:

```
char val;

int ledPin = 13;
```

Add the following setup:

```
void setup() {
  pinMode(ledPin, OUTPUT);
  Serial.begin(9600);
}
```

Center for
Human-Computer Interaction
University of Salzburg

Add this loop, run the code and click your mouse:

```
void loop() {
     if (Serial.available()) {
          val = Serial.read();
     }
     if (val == '1') {
          digitalWrite(ledPin, HIGH);
     }
     else {
          digitalWrite(ledPin, LOW);
     }
     delay(10);
}
```

# FIRMATA FIRMWARE

Center for
Human-Computer Interaction
University of Salzburg

Lets you communicate directly to Arduino via the Processing IDE

Advantage: some examples are included

(*Processing* > *File* > *Examples*)

Upload *Examples* > *Firmata* > *StandardFirmata* to your board

Quit the Arduino IDE

Include the Firmata library in Processing

*Sketch* > *Import Library* > *Add Library …* > *Arduino (Firmata)*

Open & Run

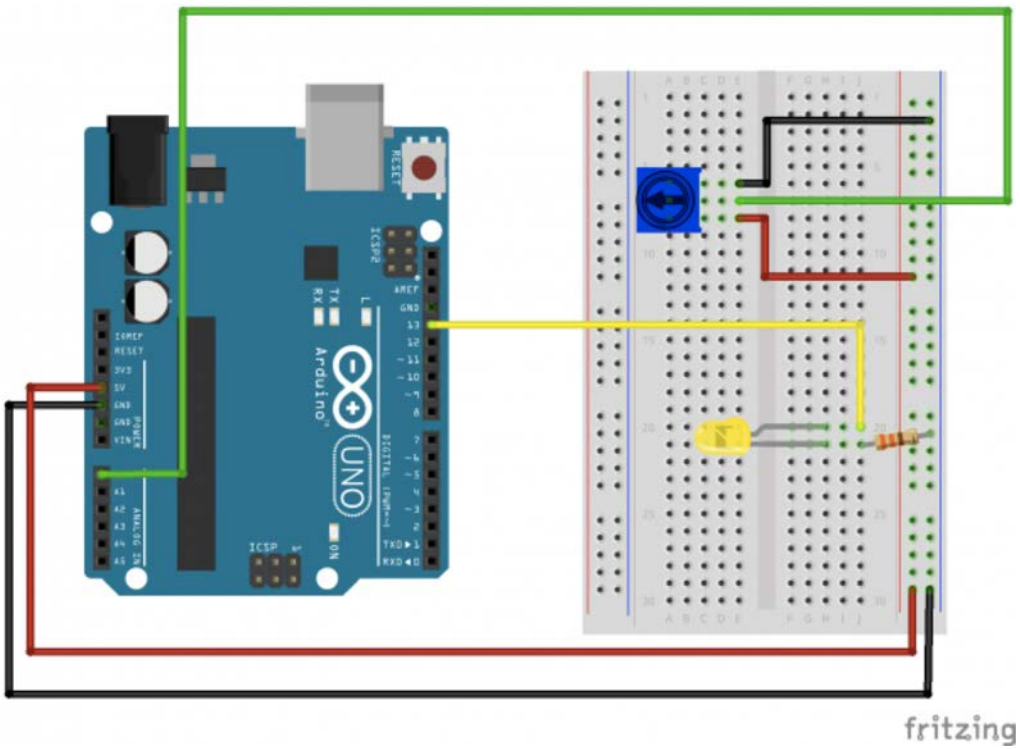
*Examples* > *Topics* > *Fractals and L-Systems* > *Tree*

No need to understand the code, but take a look at this line:

```
float a = (mouseX / (float) width) * 90f;
```

How can we change that variable to take an input from our Arduino?

Center for
Human-Computer Interaction
University of Salzburg




Build the top part of this circuit (forget about the LED, we don't need it)

Potentiometer
+ to 5V
- to GND
output pin to A0

Back to processing. Before setup, add the lines:

```
import processing.serial.*; // reference the serial library

import cc.arduino.*; // reference the arduino library

Arduino arduino; // create a variable of the Arduino data type
```

Then, in your setup, add:

```
arduino = new Arduino(this, Arduino.list()[0], 57600);
```

Take care to choose the right COM port!

Finally, replace:

```
float a = (mouseX / (float) width) * 90f;
```

with:

```
float a = (arduino.analogRead(0) / (float) width) * 90f;
```

Run the Processing sketch,
and see what happens when you turn the pot meter.

# YOUR ASSIGNMENT

Center for
Human-Computer Interaction
University of Salzburg

- Minimum – make an illustration by hand (code shapes and colours) with simple interaction (input - mouse, keyboard)
- Add complex interaction (input – camera, microphone, microcontroller, …)
- Add automation/math/gravity/physics

- For inspiration
- https://www.openprocessing.org/browse/#
- https://processing.org/examples/
- https://www.youtube.com/user/shiffman

- PS on 16th June 2020

Center for
Human-Computer Interaction
University of Salzburg

For intermediate questions:

jakub.sypniewski@sbg.ac.at