

Inkstrument: Interactive Musical Instrument with Conductive Ink

Jockey Club Project IDEA Inclusive Digital and Experimental Art

賽馬會科藝共融計劃

Workshop Overview

1 7 Comprehensive Lessons

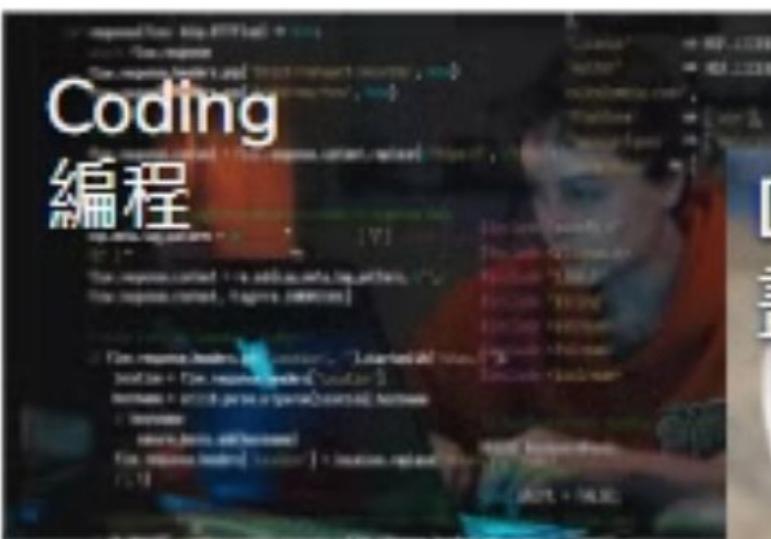
Each lesson runs 1.5-2 hours, covering conductive ink basics through to building a custom instrument.

2 Target Students

Designed for secondary school students, with no prior electronics experience required.

3 Key Learning Outcomes

Conductive ink applications, sound design, Arduino programming, and complete a musical instrument that responds to touch.



Coding
編程

Drawing
畫畫



Playfulness
有趣、玩味性

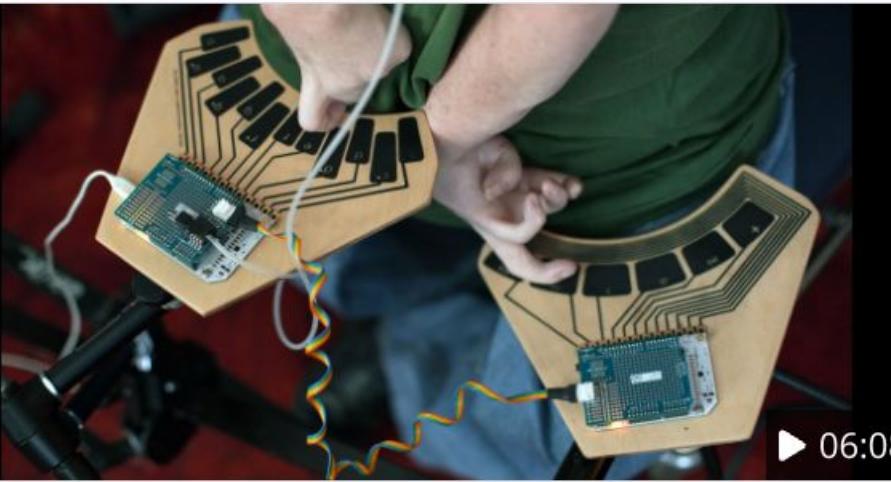
Abstract
抽象

Be Patient
耐性

Observation
觀察力

Curiosity
好奇

Frustration
沮喪



YouTube



Touch Chord - A Touch Sensitive Breath Controlled Instrument

Read the full story: <http://bit.ly/1Gg7gQ7> Designer Musician Vahagn Matossian from Human Instruments teams up with Bare Conductive to develop a new musical...

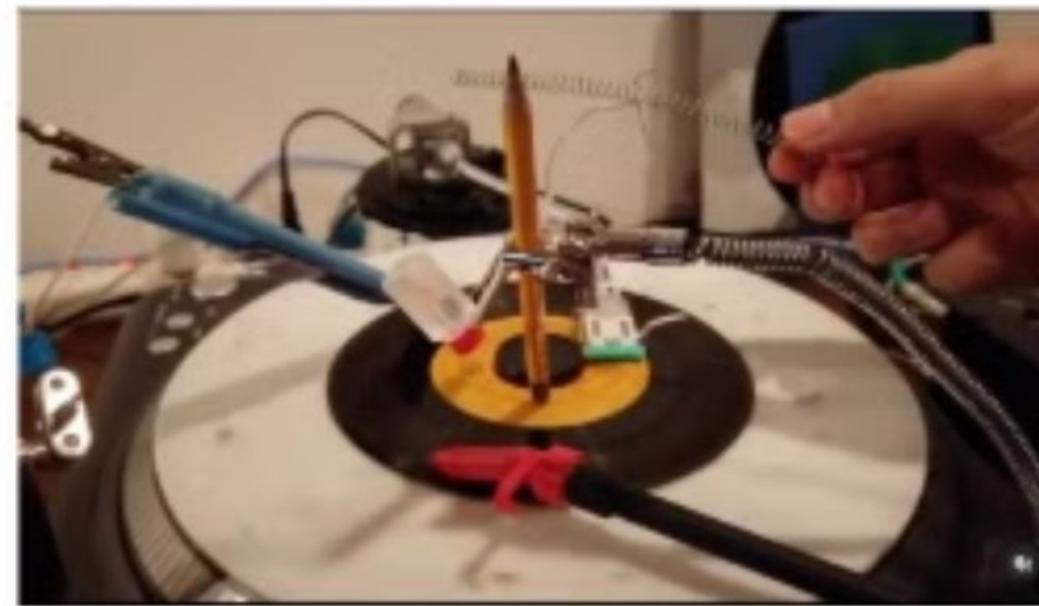


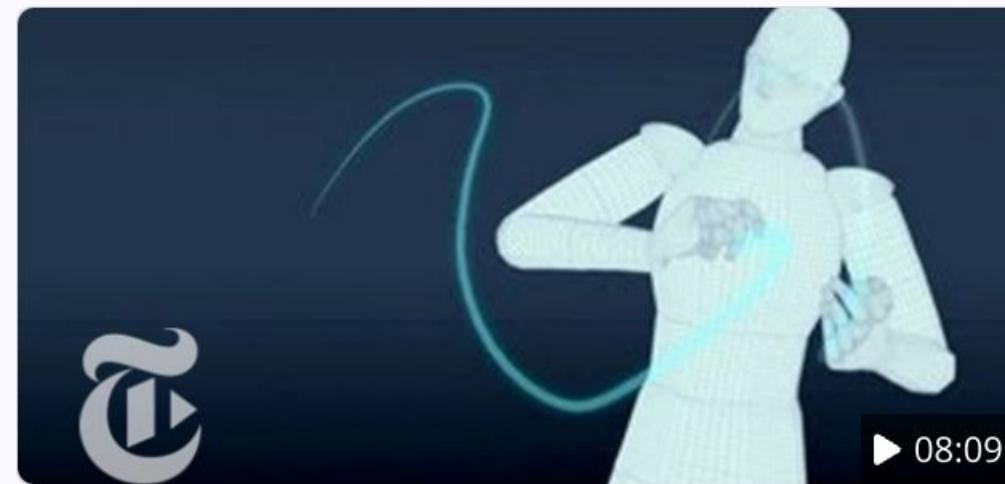
Bare Conductive



Making a Bionic Hand Touchscreen Friendly

Stephen Lowry shows us how he used Electric Paint to make his bebionic3 bionic hand touch screen friendly, allowing him to use his smart phone and other devices.



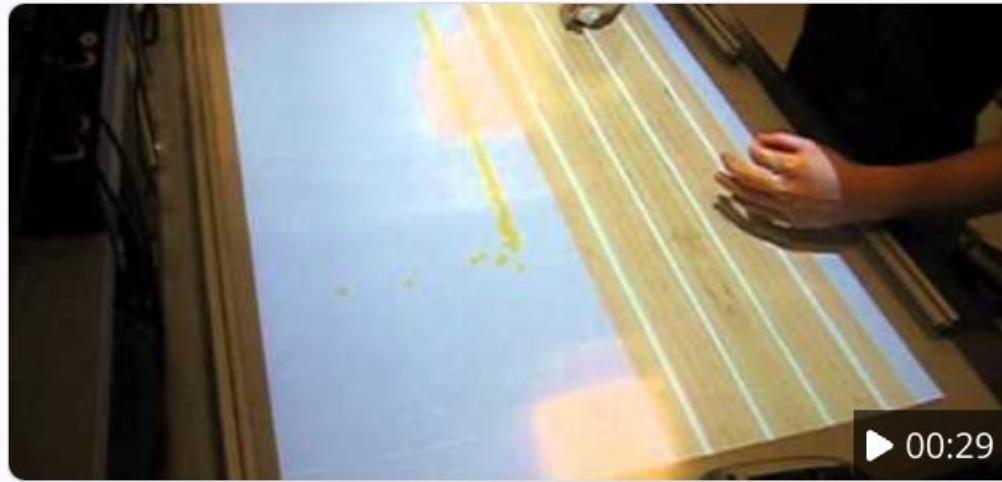


YouTube



Demystifying Conducting: The Connection Between Gesture and Musi...

Alan Gilbert, music director of the New York Philharmonic, demonstrates and discusses the role of a conductor. Subscribe to the Times Video newsletter for free...



YouTube



Gesture Based Music

In this demonstration, a ceiling mounted projector displays an image downward onto a partially retro-reflective screen material. This enables a camera, also...



YouTube



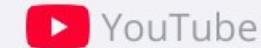
Live-Coding - programming masterly music | Juan Romero & Patrick ...

Benoît and the Mandelbrots see the laptop as their main instrument; they are mainly dedicated to live coding, the process of writing software in real time. They...

```
3 # konbanwa! Hello from Leicester
4 # hi from sheffield ! konbanwa
5
6 Scale.default="zhi"
7
8 Clock.bpm :=
9
10 z3 >> donk(linvar([0,3,-2]) + (0,7), dur=PDur(5,16), sus=linvar([-3,3],8), oct=var([5,{6,7}], [28,4]))
11 y1 >> play("/*]v v [ h s", rate=linvar([0.8,1],8), amp=.8, echo=PStep([5,4],0.75), r
12 y1 >> play("/*]v v [ h s", rate=linvar([0.8,1],8), amp=.8, echo=PStep([5,4],0.75), r
13 d1 >> play("x,x,x,x< s>yy dommune >", crush=0, psifft=2,pitch).sometimes("rate=1
14 d1 >> play("x,x,x,x< s>yy lucy >", crush=0, psifft=2,pitch).sometimes("rate=1
15 d2 >> play("w", dur=PDur(5,8,[0,2]), rate=[2,2,4,6], formant=2, crush=PRand([0,8]), pan=0
16 d2 >> play("w", dur=PDur(5,8,[0,2]), rate=[2,2,4,6], formant=2, crush=PRand([0,8]), pan=
lucy=list(P[2:10]), dur=2)
17
18 # ok we have 3 mins :) wanna slow it right down and go weird?
19 y4 >> play("-{ss}]{-----]-{ss}][ssssssssssssssssssssssssssssss]", ps
ten", 16)
20
21 #yyyyyyyyyyyyyyyy
22
23 loz : z3 >> donk(linvar([0,3,-2]) + (0,7), dur=PDur(5,16), sus=linvar([-3,3],8), oct=var([5
,{6,7}], [28,4]))
24 lucy : Clock.bpm = 45
25 yyan : d2 >> play("w", dur=PDur(5,8,[0,2]), rate=[2,2,4,6], formant=2, crush=PRand([0,8]),
pan=[-2,1], shape=PRand([0,1,0,2]).sometimes("stutter", 8, rate=list(P[2:10]), dur=2)
26 lucy : # ok we have 3 mins :) wanna slow it right down and go weird?
27 .... : y4 >> play("-{ss}]{-----]-{ss}]", psifft=[1,2,3,4,5,6,7], rate=va
r([1,-1]).every(4, "stutter", 16)
28 loz : y1 >> play("/*]v v [ h s", rate=linvar([0.8,1],8), amp=.8, echo=PStep([5,4],0.75),
room=0.1, drive=.1) # wahhhhhhhhhh =0
```



► 02:37:01

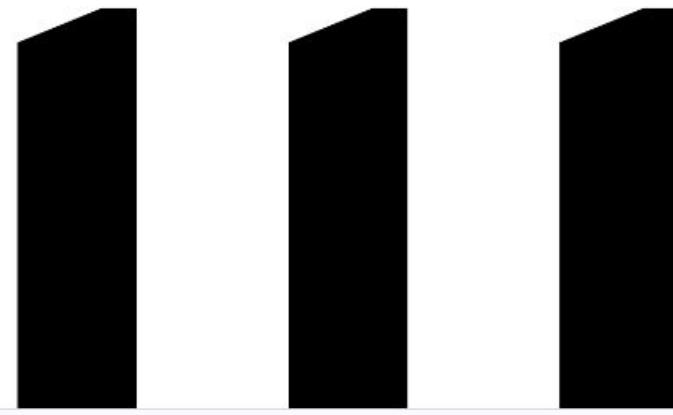


DOMMUNE Tokyo - live coding performances - algorave tokyo x yorks...

DOMMUNE - Tokyo x Yorkshire exchange transmission

<http://www.dommune.com/reserve/2018/1114/> With thanks to Great Britain...





iii instrumentinventors.org

Homepage - instrumentinventors.org

iii is an artist run, community platform supporting new interdisciplinary practices linking performance, technology and the human senses



Demo: Adafruit Circuit Playground with Accelerometer

In this demonstration, we'll explore how physical movement translates into sound using the Adafruit Circuit Playground's built-in accelerometer.

What is an Accelerometer?

A sensor that detects changes in velocity and orientation along different axes (X, Y, Z), allowing our instrument to respond to tilting, shaking, and rotation.

Movement-to-Sound Conversion

As we tilt, shake, or rotate the Adafruit board, the accelerometer data directly modifies sound frequency (Hz), creating higher pitches with faster movements and lower pitches with slower movements.

Demo 1 (Processing)

Sound Generation

An oscillator with ADSR envelope (Attack, Decay, Sustain, Release) creates sounds that dynamically respond to movement, demonstrating how electronic instruments produce and shape tones.

Motion Tracking

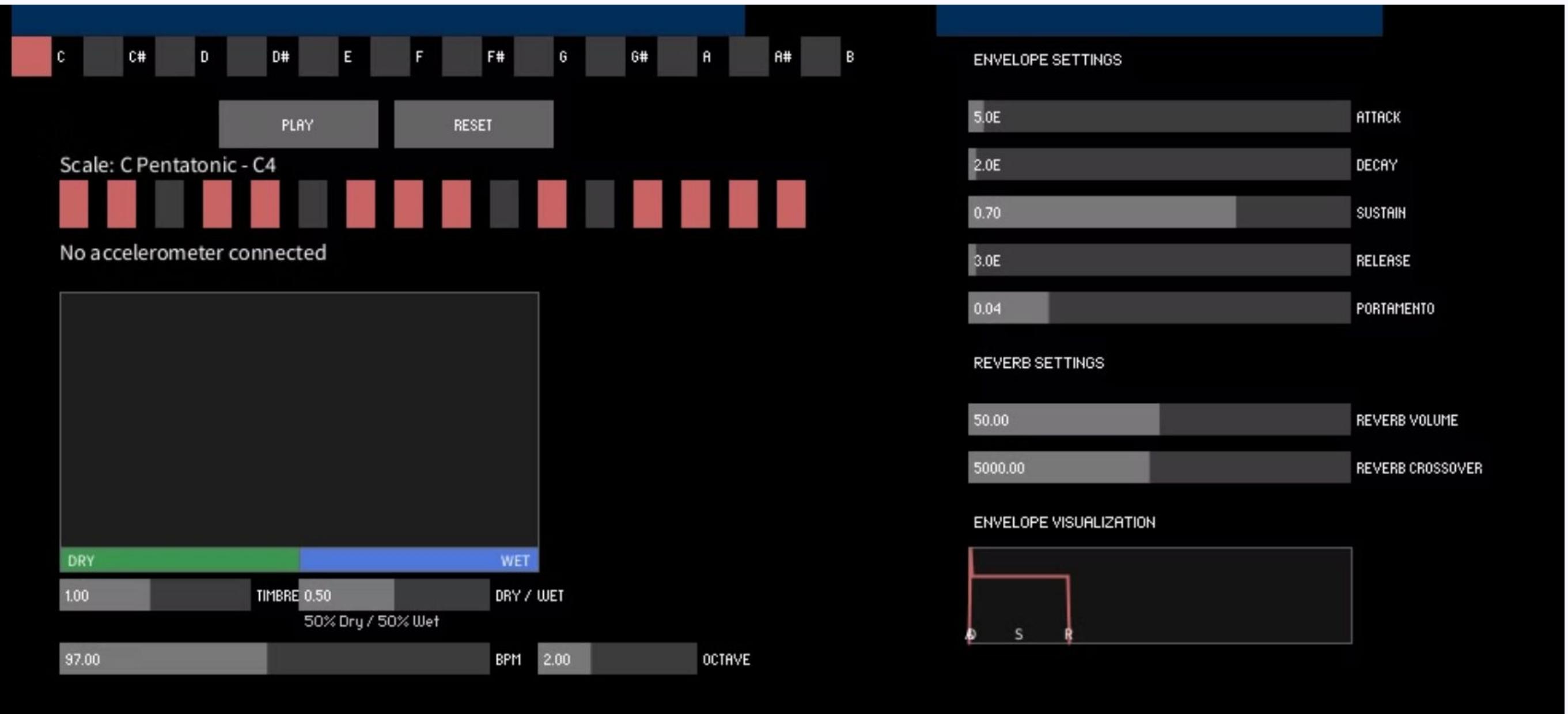
An accelerometer captures physical movements along X, Y, and Z axes.

Visual Feedback

Using Processing IDE to create real-time visualizations of the accelerometer data, showing the connection between movement and sound generation.

Audio Processing

Pure Data (PD) handles the audio signal chain, demonstrating how open-source tools can create sophisticated sound processing systems.

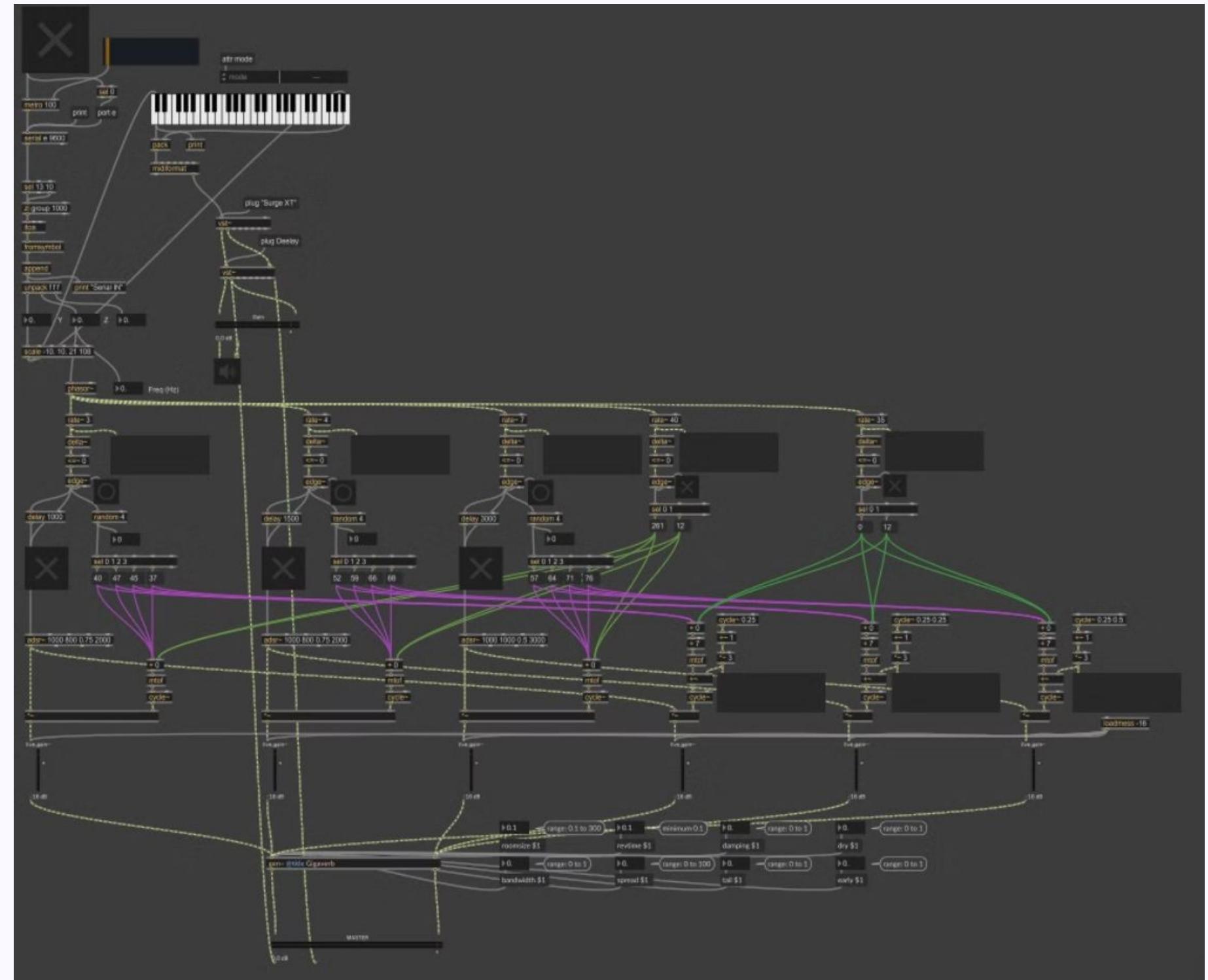


Demo 2 (Max/MSP)

In this demonstration, we'll explore how Max/MSP can transform physical movement into musical parameters.

We're using the powerful free VST synthesizer Surge XT as our sound source, paired with the Deelay effect plugin for time-based audio manipulation. The X-axis movement data (ranging from -10 to 10) is mapped to MIDI notes using the kslider object, which are then sent directly to the Surge XT synthesizer.

Simultaneously, Y-axis movement data feeds into a phasor~ object, creating cyclical patterns that control our note generator. This creates a dynamic relationship between physical gestures and musical output, similar to what we'll achieve with our conductive ink instruments.



Text vs. Node-Based Creative Coding in the AI Era

As we will develop our conductive ink instruments, we'll use text-based coding (like Arduino) to translate physical inputs into sound.

Text-Based Coding

- Offers precise control through written commands
- Essential for understanding core programming concepts
- Skills remain relevant even as AI tools like GitHub Copilot assist with code generation

Node-Based Programming

- Visual approach that connects functional blocks with "wires"
- Intuitive for sound design and signal processing
- Encourages experimentation and rapid prototyping

Understanding both paradigms gives you versatility in expressing your creative ideas and prepares you to work with both traditional coding and emerging AI-augmented tools.



Lesson 1: Introduction to Conductive Ink

What is it?

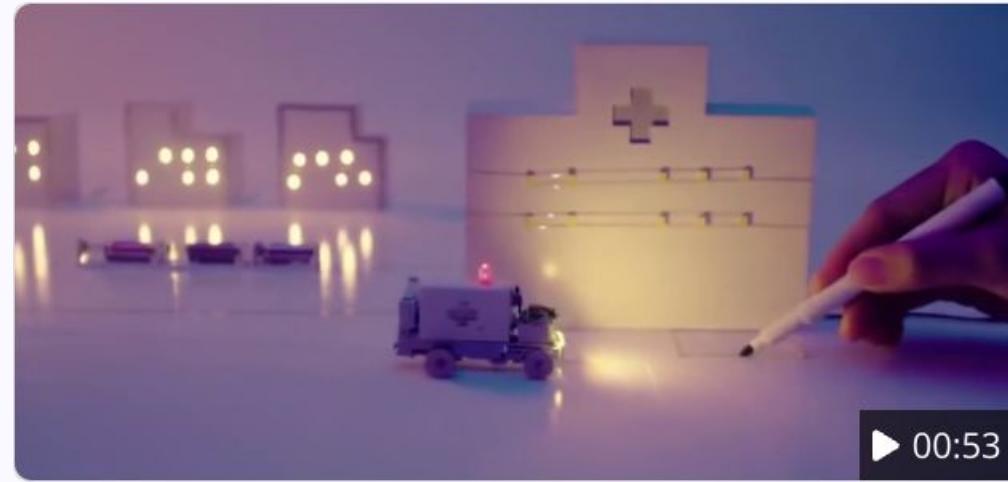
Conductive ink is a special ink that conducts electricity.

How is it used?

It can be used in electronics and art projects.

Lesson Overview

We'll explore its properties and applications.



YouTube



Pen that draws electricity - The AgIC Circuit Marker pen uses conduct...

The AgIC Circuit Marker pen uses conductive ink that allows working electrical circuits to be drawn on pieces of paper. Available on Amazon for only \$15!...

▶ 00:53

Understanding Electrical Conductivity

Conductivity refers to a material's ability to allow the flow of electrical current. Just like water flows through pipes, electricity flows through conductive materials.

Electronic Circuit

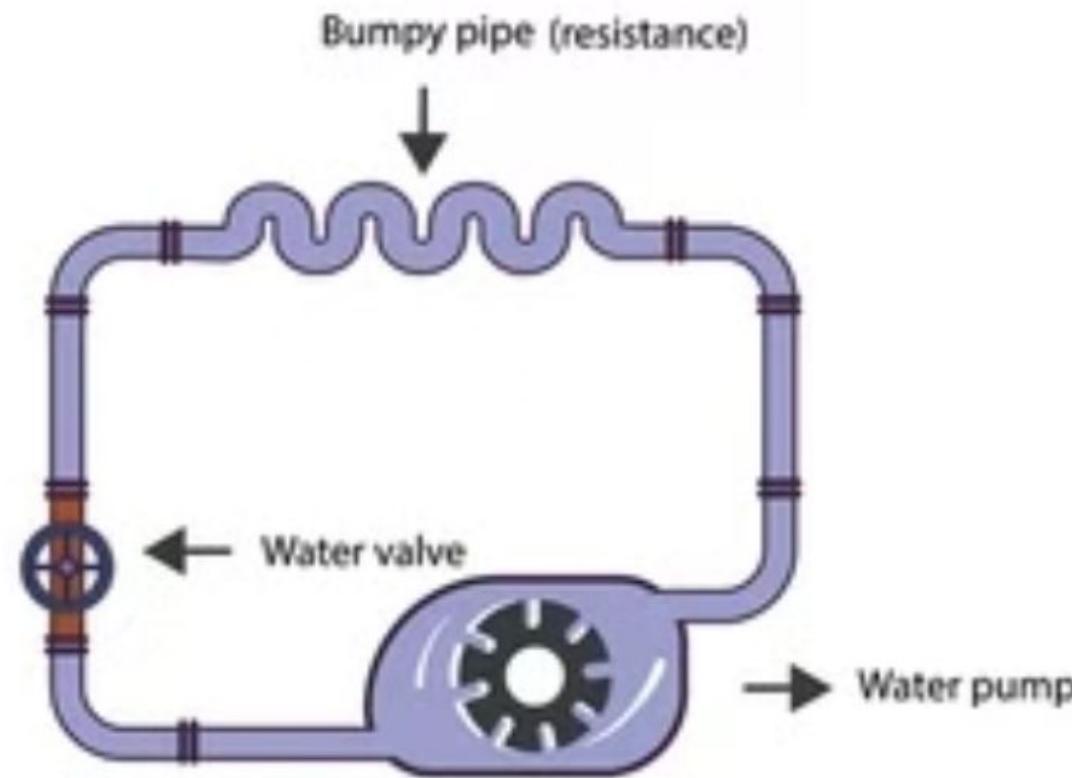
- Electricity flows through conductive wires
- Electrons move from negative to positive
- Resistance restricts electrical flow
- Voltage acts as electrical pressure

Water Pipe System

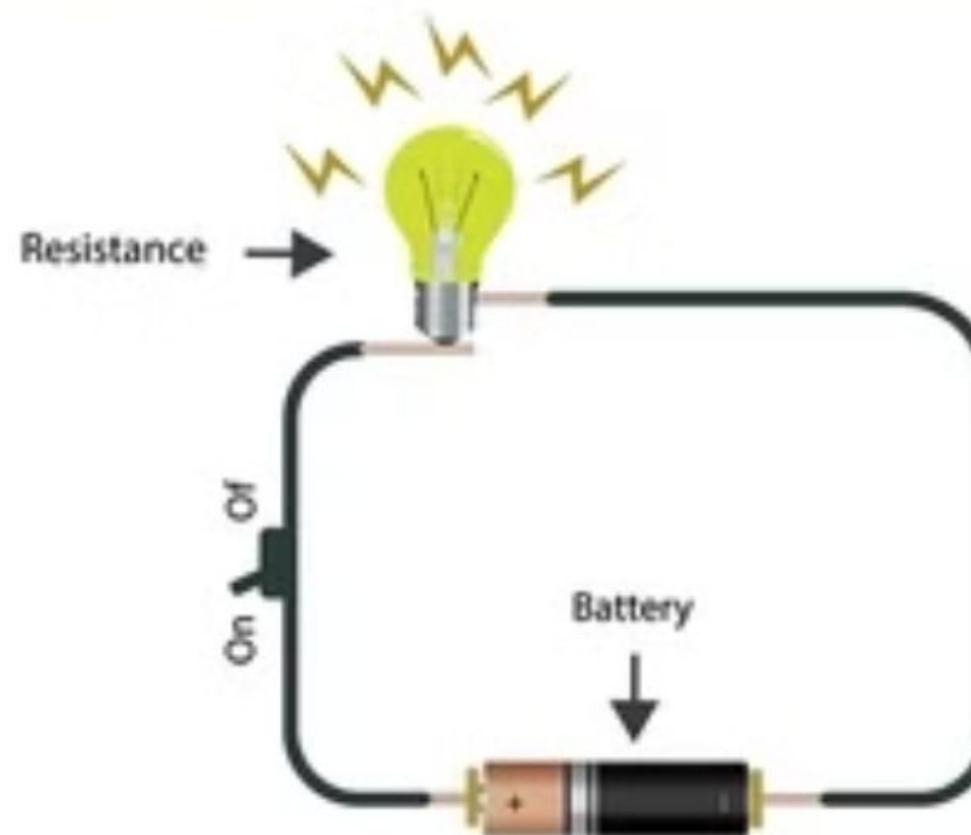
- Water flows through pipes
- Water molecules move from high to low pressure
- Narrow pipes restrict water flow
- Water pressure pushes water forward

When we draw with conductive ink, we're essentially creating flexible "pipes" for electricity to flow through. This allows us to design creative circuits without traditional rigid wiring.

SIMPLE WATER SYSTEM



SIMPLE ELECTRIC CIRCUIT





YouTube



The pen that draws electricity

The AgIC Circuit Marker pen uses conductive ink that allows working electrical circuits to be drawn on pieces of paper. Japanese venture AgIC developed. We test...

▶ 03:19



YouTube

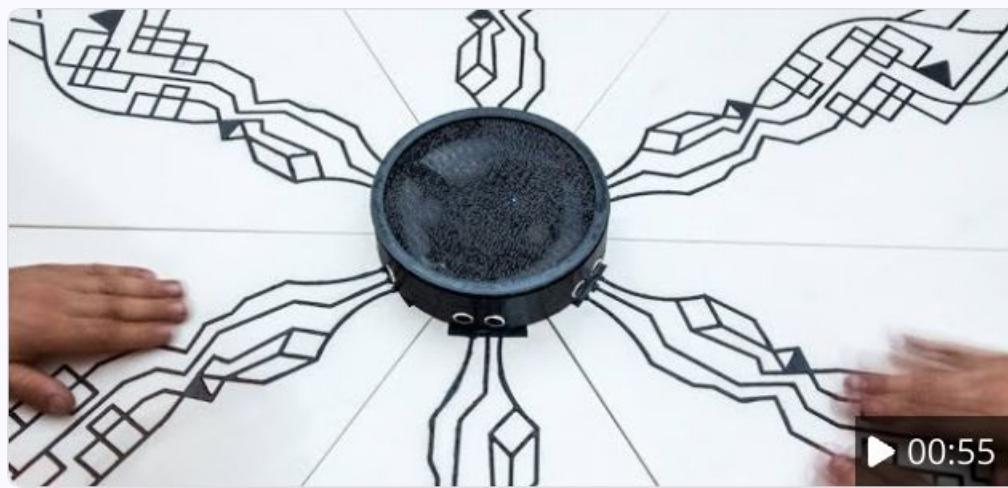


Electric Paint Lamp Kit

Find out more about the Electric Paint Lamp Kit here:

<https://www.bareconductive.com/collections/electric-paint-lamp-kit> Transform any...

▶ 01:1



YouTube



Build Your Own Interactive Sound Table with Electric Pai...

Find out more about Electric Paint here:

<https://www.bareconductive.com/collections/electric-paint> TACTO is an interactive...



YouTube



Interactive media wall (touch sensor, conductive ink, projection map...)

[PLAYDODO] Music Playing Wall : Projection Mapping, Conductive Ink, Interacitve Art
Music Playing Wall, a fun Interactive touch wall painted with instruments recognize...

Understanding Resistors in Electronic Circuits

Resistors are fundamental electronic components that limit or restrict the flow of electrical current in a circuit. They provide precise amounts of resistance measured in ohms (Ω).

Resistors in Electronic Circuits

Resistors work by converting electrical energy into heat, thereby reducing current flow.

- Low resistance (e.g., 10Ω) allows more current to flow
- High resistance (e.g., $10,000\Omega$) significantly restricts current
- Resistors protect sensitive components from excessive current
- They're used to divide voltage and create specific circuit behaviors

Just as we use different pipe sizes to control water flow in specific parts of a plumbing system, we use different resistor values to control electrical current in different parts of a circuit.

Water Pipe System Analogy

The water pipe system provides an excellent way to understand how resistors function:

- Wide pipes (low resistance) allow water to flow easily
- Narrow pipes (high resistance) restrict water flow
- Water pressure (voltage) pushes against the restriction
- Flow rate (current) decreases as pipe narrowness increases

RESISTORS EXPLAINED



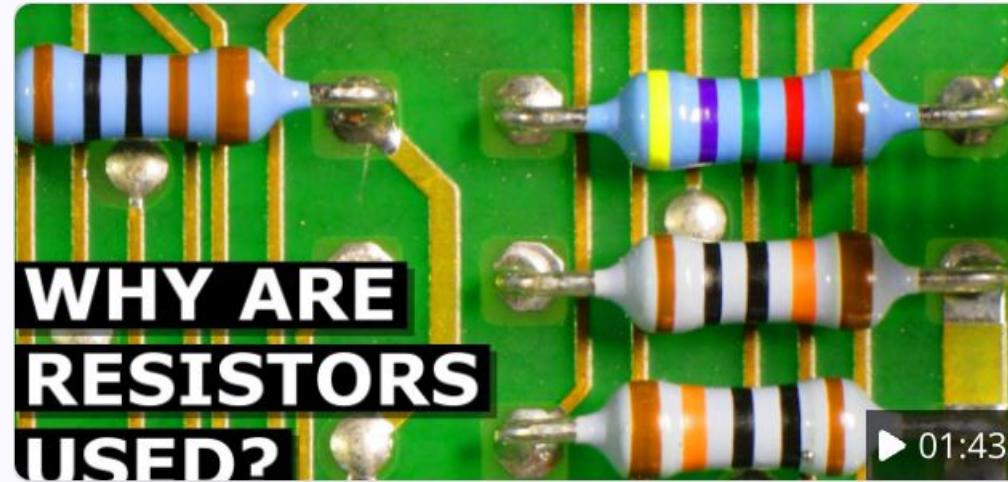
YouTube

How Resistor Work - Unravel the Mysteries of How Resistors Work!

In this video, we're going to learn about how resistors work! We'll explore the different types of resistors, how resistors work in circuits, and how to calculate...

▶ 28:23





YouTube



Why Resistors Are Used In circuits

In this video, we're going to answer the question: Why are resistors used? 👍 👍 👍

FREE design software ➡ <https://www.altium.com/asp/the-engineering-mindset/> ...

Why Resistors Are Critical for Our Inkstrument

For our conductive ink musical instrument project, resistors play several essential roles:



Component Protection

Resistors protect our Arduino microcontroller and other sensitive electronic components from receiving excessive current that could damage them.



Sound Modulation

By creating voltage dividers with different resistor values, we can generate various tones and control pitch in our Inkstrument, allowing for more expressive musical capabilities.



Touch Sensitivity

When used with conductive ink, resistors help calibrate the sensitivity of the capacitive sensors, ensuring reliable detection of finger touches and gestures on our instrument's surface.

Material list

The following components are required to build our interactive musical instrument:

Core Electronics

- Arduino Uno R3 microcontroller board
- TouchKeyUSB capacitive touch sensor module
- VS1053 module with SD card
- Small speaker with audio jack output
- USB cable

Conductive Materials

- Conductive Ink
- Copper tape
- Assorted resistors ($10k\Omega$, $1k\Omega$, and 220Ω)
- Clip wires

Lesson 2: Exploring Electronic Instruments

Theremins

Exploring theremins and gesture-based control.

Sound

Understanding electronic sound generation.



YouTube



01 Music Basics - What is Music?

Music theory from the very beginning. If you want to know how to write music, want help revising for exams at school, or just interested in how music works, then this...

Tonality

The overall sound of the music as pleasant or unpleasant

Timbre

The unique sound quality of an instrument or sound

Texture

The layers of sound, how sparse or dense the music is

Rhythm

How long or short a sound is

Dynamics

How loud or soft the music is

Form

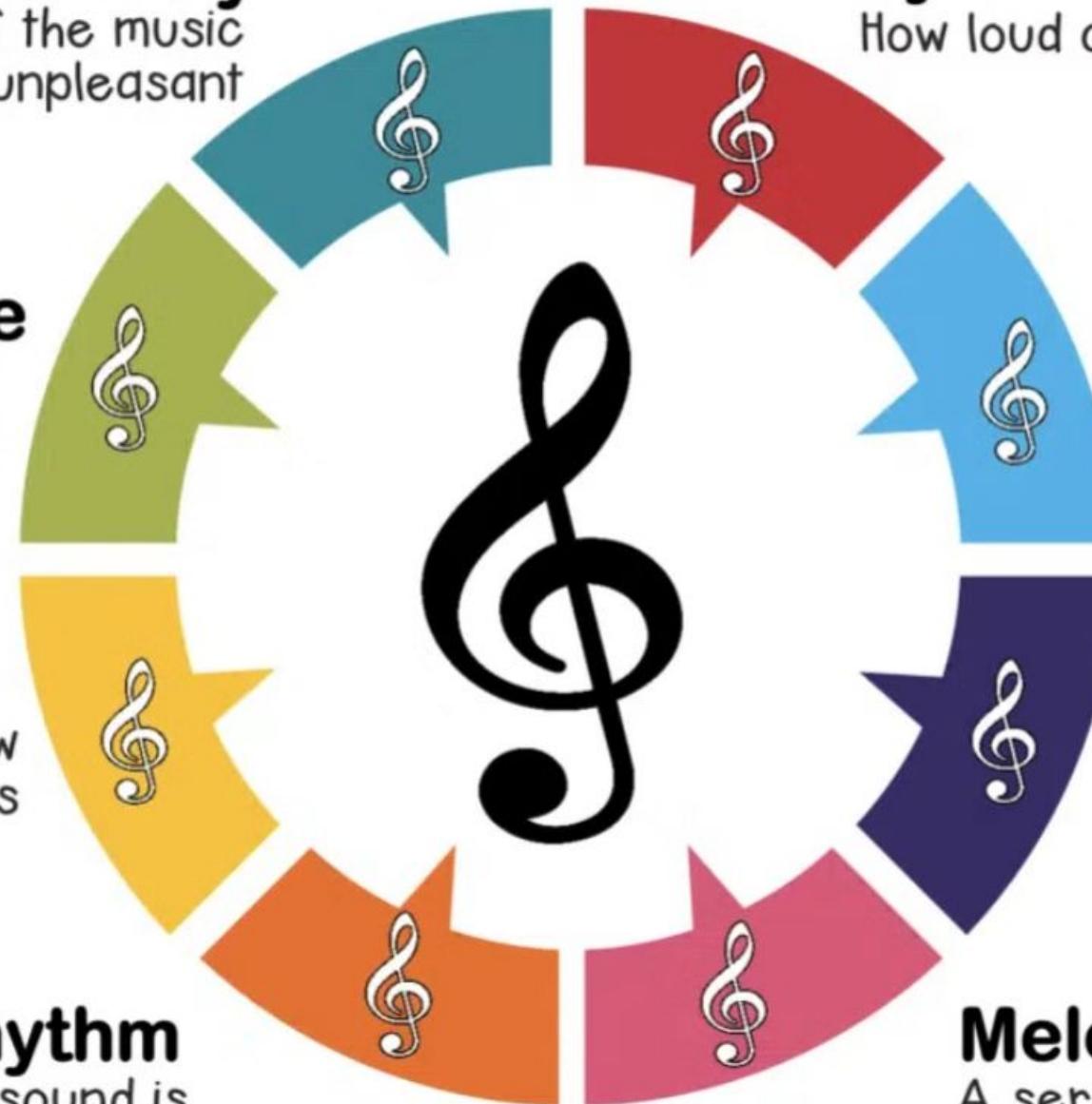
The order and arrangement of the parts of the music

Harmony

The instruments that support the melody with chords

Melody

A series of pitches that makes a tune



Core Elements of Music for Electronic Instruments



Rhythm

The pattern of time in music - controls when notes are played and for how long. In electronic instruments, rhythm can be precisely controlled through programming or touch patterns on conductive surfaces.



Melody

The sequence of notes that form the main recognizable theme. Our Inkstrument will allow different conductive ink patterns to trigger melodic sequences.



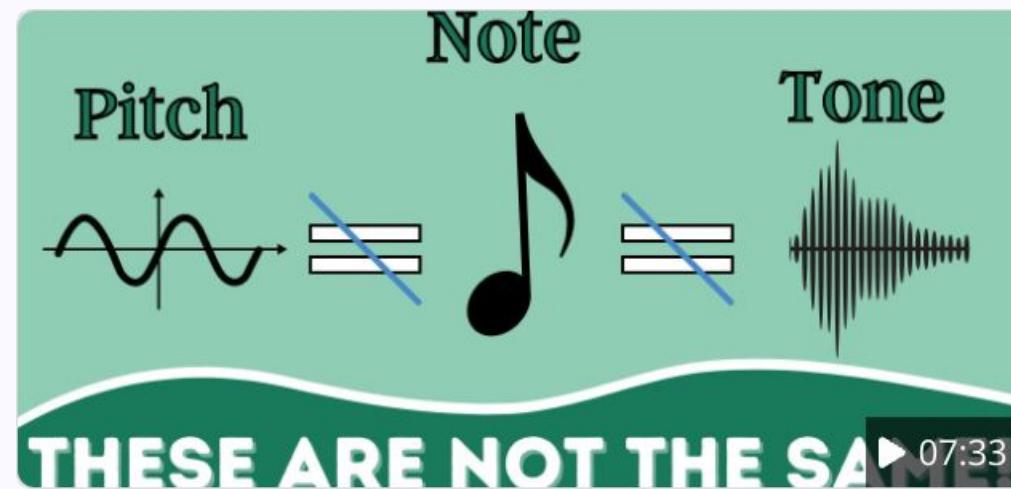
Tone/Timbre

The quality and character of sound that distinguishes different instruments. With our Arduino and VS1053 setup, we can manipulate electronic signals to produce various timbres from a single interface.



Dynamics

The variation in loudness or softness of notes.



YouTube

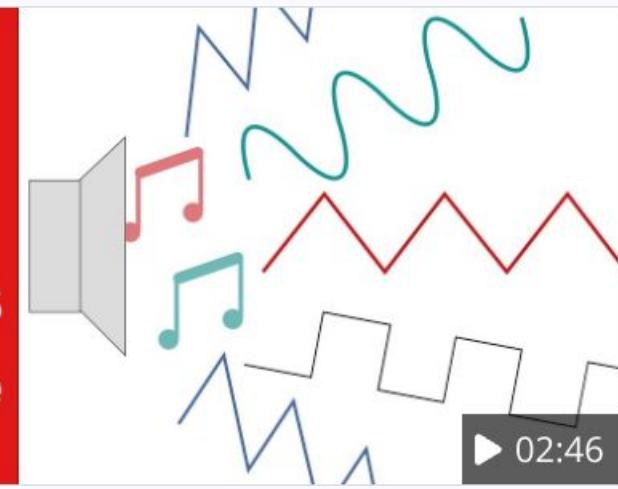
The Difference Between Pitch, Note, and Tone

If you'd like to support my channel, consider joining on Patreon:

<https://shorturl.at/Tw3Mu> This is episode 1 of the music fundamentals series!...



What do Electronic Waveforms Sound Like

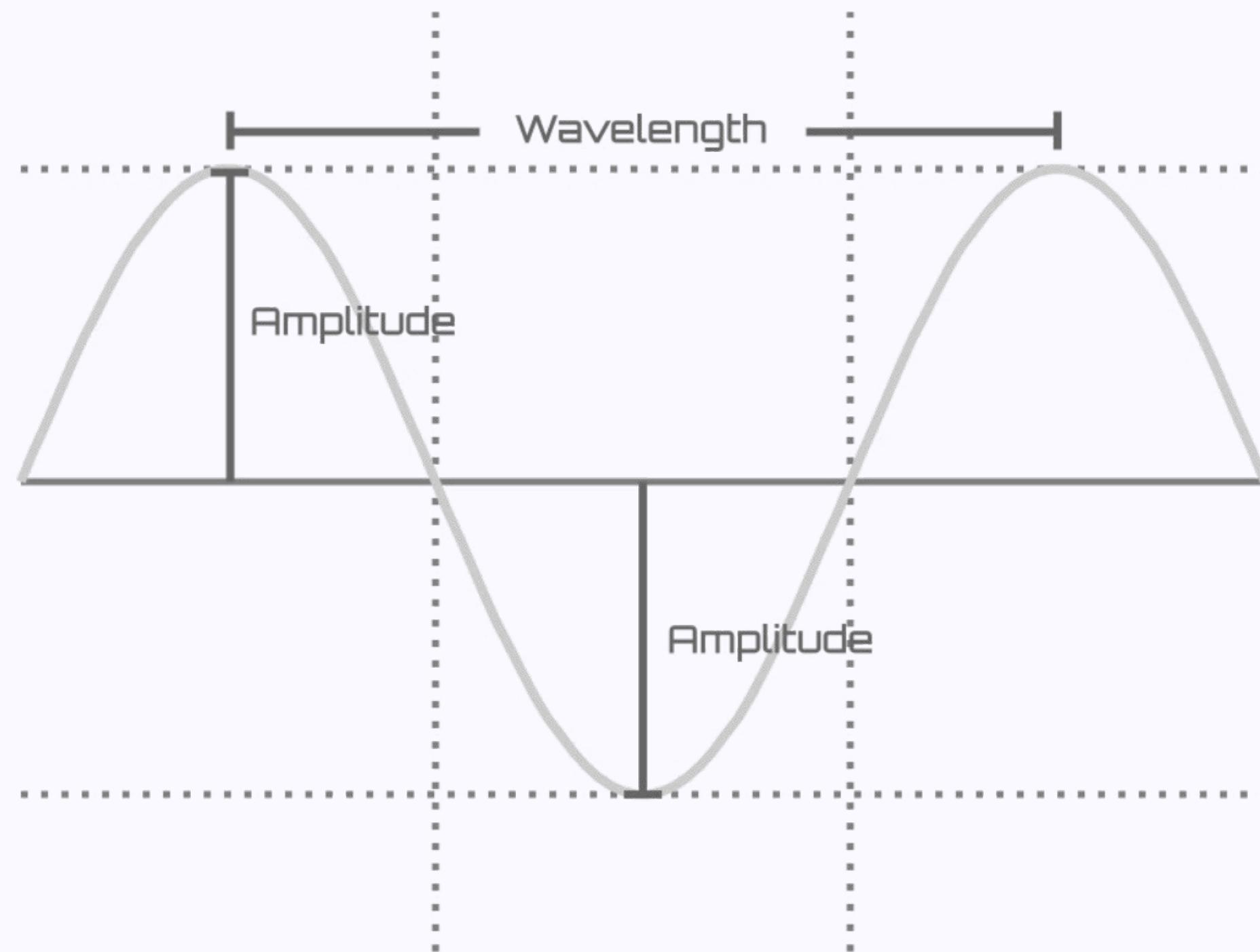


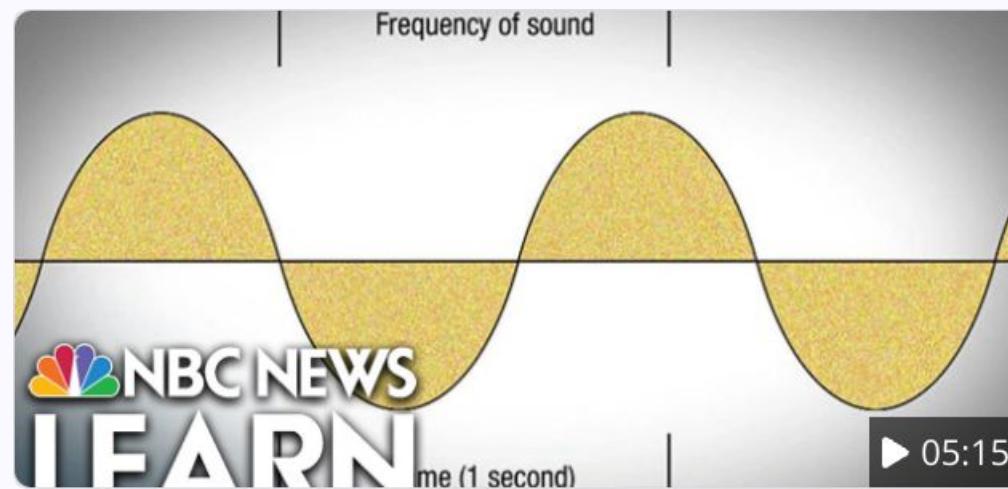
YouTube

What do Electronic Waveforms Sound Like

This video gives you the key information about common waveforms found in electronics and lets you hear what they actually sound like. The waveforms include...





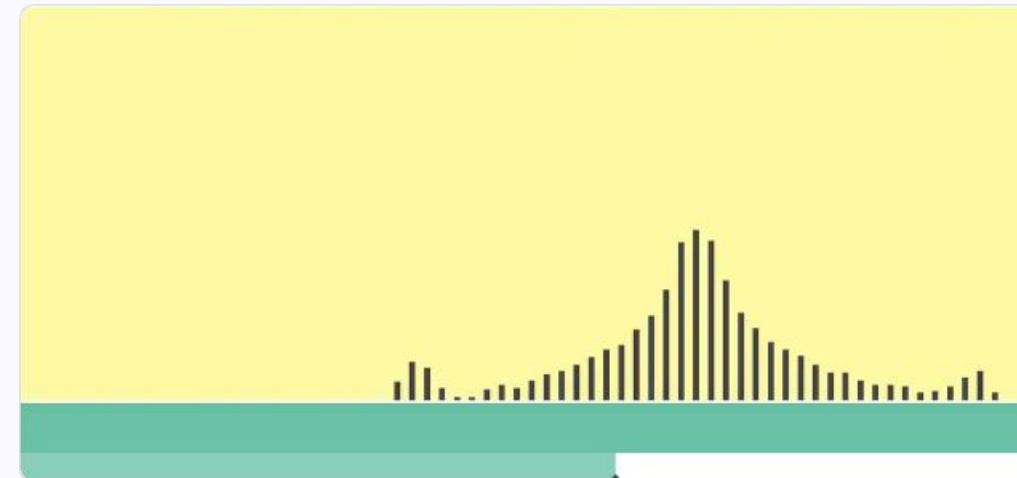


YouTube



Sound Waves

Sound, Sound Waves, Waves, Vibration, Matter, Vibrate, Vibration, Guitar, Singing, Song, Vocal Chord, String, Frequency, Pitch, Energy, Volume, Amplitude



 davay42



Theremin | Web Synths Collection by Playtronica & Chromatone

Experience the magic of the theremin with this interactive web-app by Femur Design. Harness the power of motion to create ethereal music without physical...

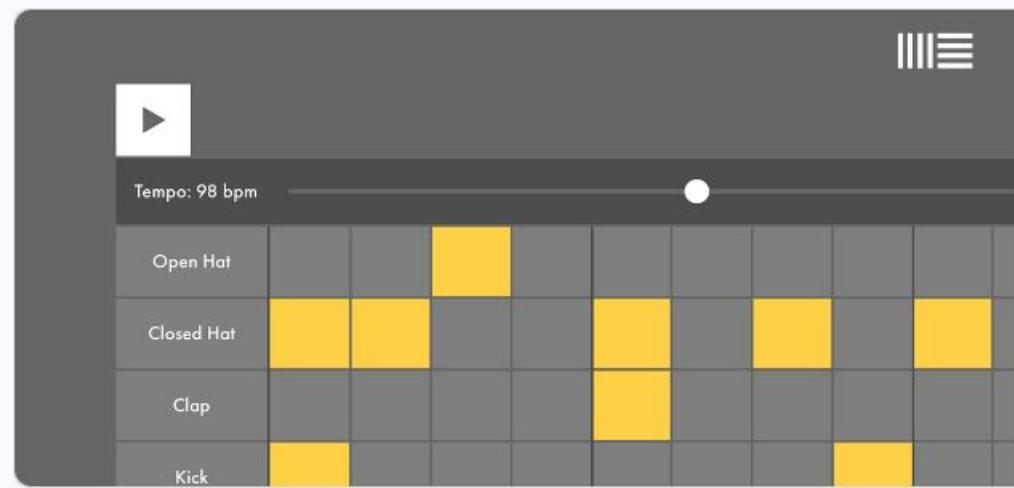


 RolandGlobal



Roland50.studio

Emulate the sound of Roland's most famous and influential musical instruments from Yuri Suzuki and Roland.



learningmusic.ableton.com



Get started | Learning Music

Explore the fundamentals of music via Ableton's interactive website. Experiment with beats, melody, harmony, basslines, and song structure in your web browser.



YouTube



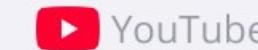
Sounds of a Glass Armonica

Composer William Zeitler plays a glass armonica, invented by Benjamin Franklin in 1761. For more on this fascinating story read the full article here on the Toronto...

D | Y
ИНСТРУМЕНТ
ДЛЯ AMBIENT И DRONE



▶ 17:0



YouTube

Я сделал Ambient Box на основе EBow | I made an Ambient Box bas...

Взял за основу принцип работы EBow и создал Ambient/Drone Box I took the principle of EBow and created an Ambient/Drone Box based on it



KONTINUUMLAB
WORKSHOPS
Cardboard MIDI
Recorder



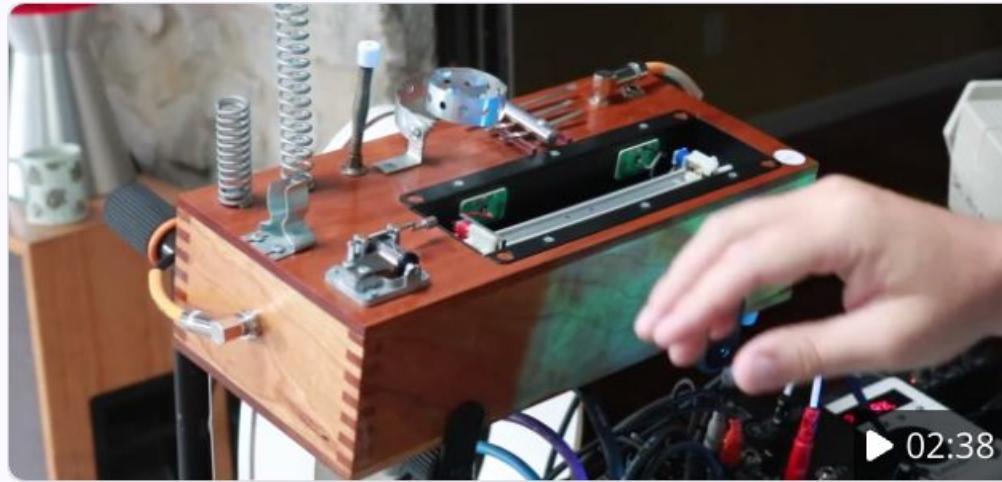
YouTube



KontinuumLAB WORKSHOPS: Recorder MIDI Controller.

In this video tutorial I show you how to make a recorder type MIDI controller from corrugated cardboard and other cheap materials and components. This is one of t...

▶ 15:5



YouTube



Noise Box 5 - experimental music instrument

Wooden noise box instrument (#5 of my collection). Perfect for film scoring, experimental music, or just improvising with friends. Reverb tank has an output, a...

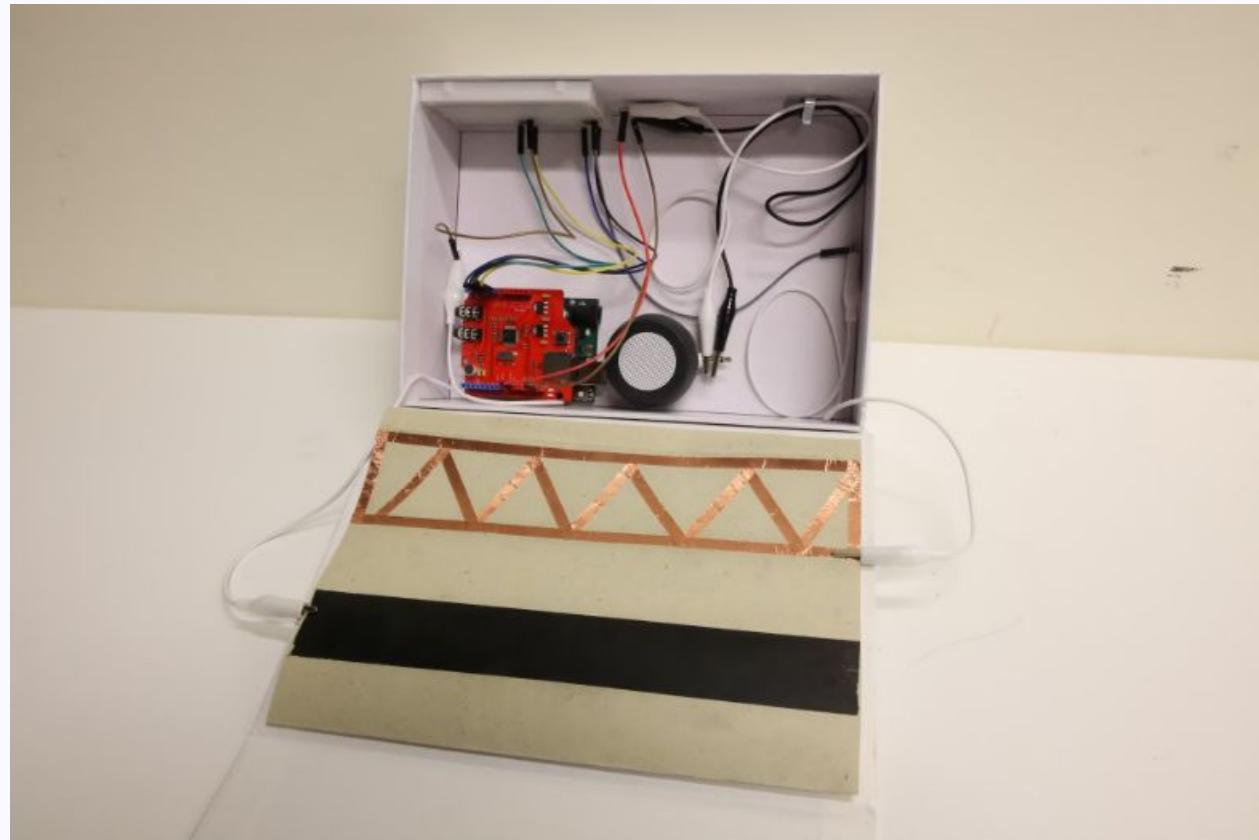
▶ 02:38



YouTube

10 Years of the Korg Volca Keys

April 2023 marks ten years since Korg's Volca Keys debuted. Chris Cline revists the popular synth to go over the features and sounds that have made it such a popula...



Lesson 3: Conductive Ink as a Sensor

Sensing

Using conductive materials for touch sensitivity.

Introduction to Arduino

Arduino is an open-source electronics platform based on easy-to-use hardware and software. It is designed to make the process of creating interactive projects more accessible to artists, designers, hobbyists, and anyone interested in creating interactive objects or environments.

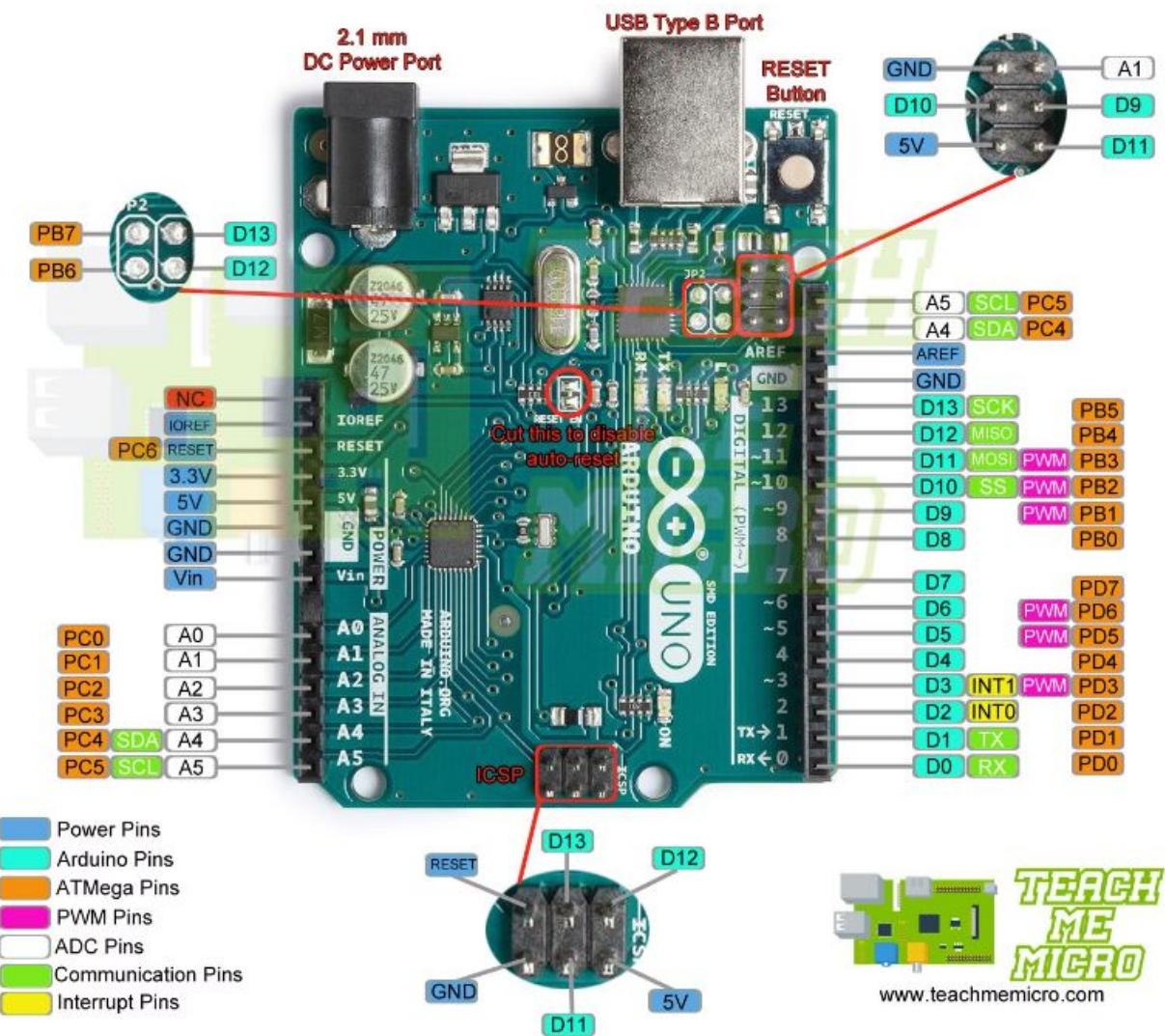


Arduino Uno R3

The Arduino Uno R3 is our microcontroller of choice for the inkstrument project because of its reliability and extensive community support. This board features an ATmega328P microprocessor with:

- 14 digital input/output pins (6 can be used as PWM outputs)
- 6 analog inputs for reading our conductive ink sensors
- USB connection for programming and serial communication

ARDUINO UNO R3 SMD PINOUT



2. USB PORT

3. USB TO SERIAL CHIP

4. DIGITAL PINS

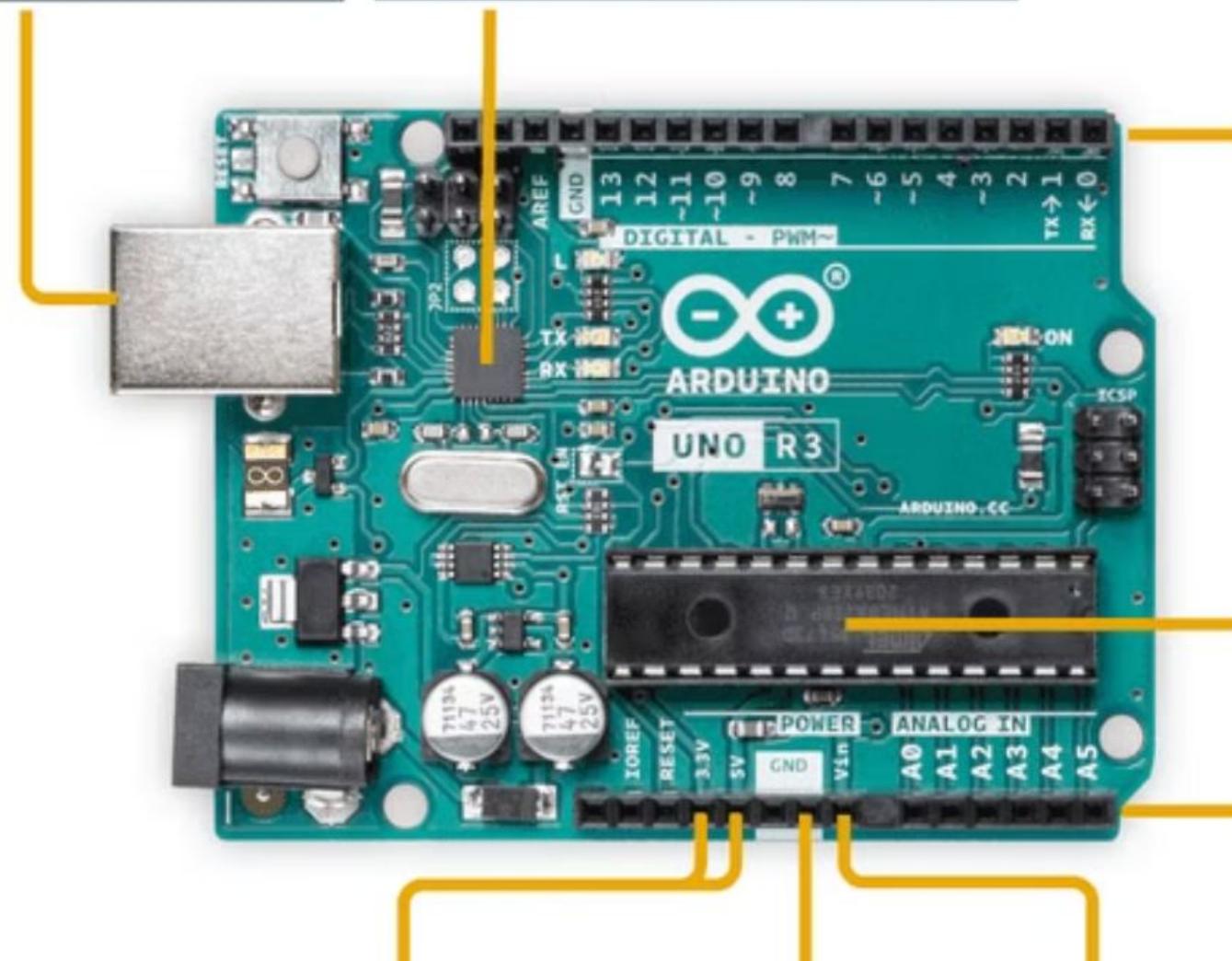
1. MICROCONTROLLER

5. ANALOG PINS

6. 5V/3.3V

7. GND

8. VIN



Serial Communication: Connecting Arduino with Creative Software

Arduino can seamlessly exchange data with creative coding platforms through serial communication protocols (RS-232, SPI, USB, CAN, I₂C, and Modbus). This connectivity enables our conductive ink sensors to control sound, visuals, and interactive experiences in software below:

Processing

Create visual responses to touch gestures by sending sensor values from Arduino to Processing sketches using the Serial library.



 Processing

Welcome to Processing!

Processing is a flexible software sketchbook and a language for learning how to code. Since 2001,...

TouchDesigner

Build interactive installations by feeding Arduino sensor data through COM ports into TouchDesigner's real-time node-based environment.



 Derivative

Derivative...

Derivative is a software company that offers TouchDesigner, a visual development platform.

Max/MSP

Generate complex sounds and musical compositions by routing conductive ink sensor readings from Arduino into Max/MSP.

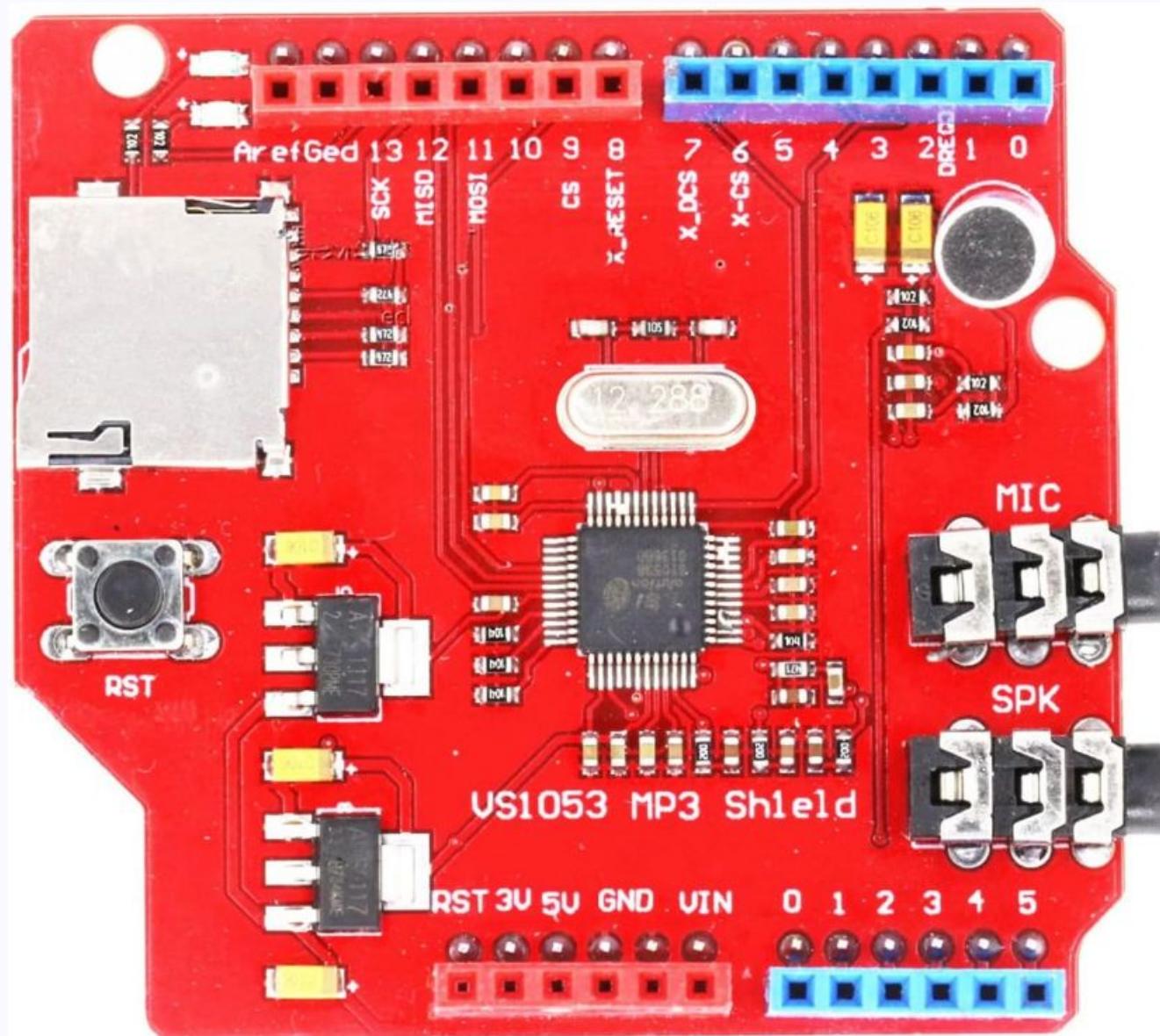


 cycling74

Cycling '74

Tools for sound, graphics, and interactivity

VS1053 MP3 Shield



The VS1053 audio codec breakout board

The VS1053 is a versatile audio codec that's perfect for creating interactive music with conductive ink instruments.

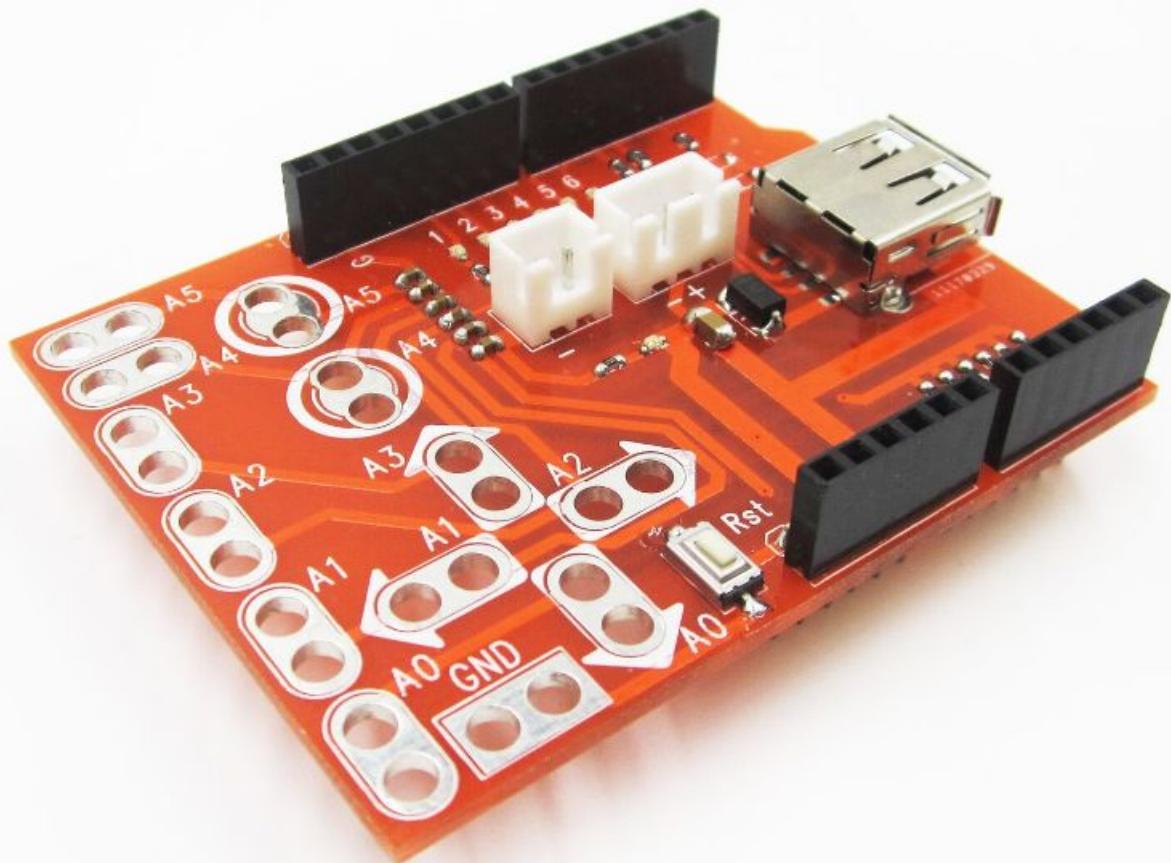
This shield enables Arduino to play MP3, WAV, MIDI, and OGG files, making it ideal for responsive sound installations.

the VS1053 translates capacitive touch inputs into dynamic sound output, creating an intuitive musical interface.

With a microSD card slot (up to 32GB), it can store and trigger complex soundscapes based on different touch gestures.

The 3.5mm audio jack and optional speaker terminals deliver high-quality stereo sound with minimal additional components.

Touch key USB



The Touch Key USB shield is an Arduino-compatible expansion board that transforms conductive surfaces into touch-sensitive inputs via capacitive sensing. This shield connects directly to our Arduino Uno R3, providing up to 16 touch-sensitive channels without requiring physical buttons. It's ideal for our conductive ink projects as it can detect touch through paper, plastic, or fabric coated with conductive ink, allowing us to create custom interactive interfaces that trigger sounds or control parameters in our electronic instruments.



Introduction to

The theremins

History

1

Invented in 1920 by Russian physicist Léon Theremin, it was one of the earliest electronic instruments.

2

Principle

The theremin is controlled by hand gestures without physical contact, using antenna that sense the position of the player's hands.

3

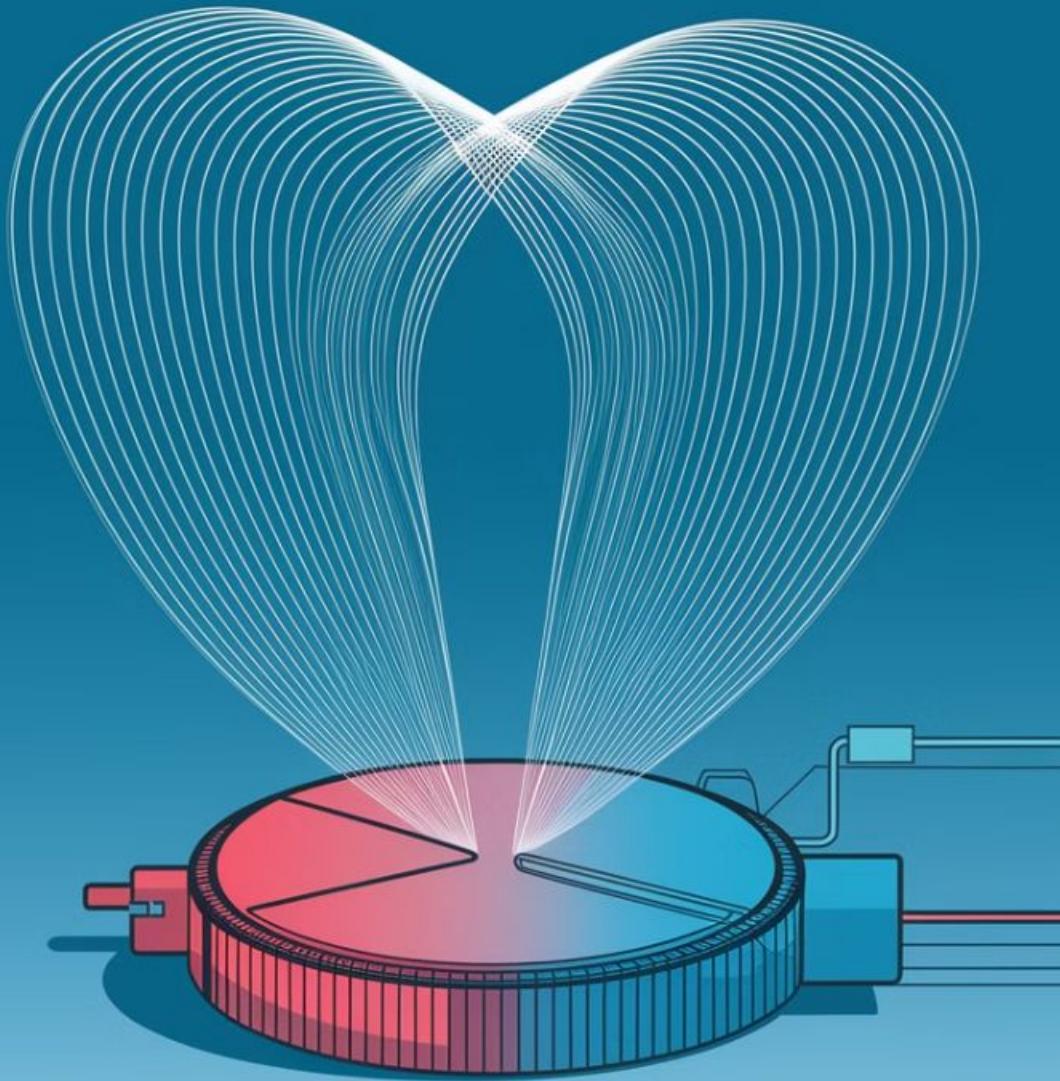
Demonstration

Let's listen to the unique, ethereal sound of a theremin and see how hand movements affect the pitch and volume.



THEREMIN

a short
introduction



Capacitive Sensing Basics

1

How it Works

Detects changes in electrical capacitance.

2

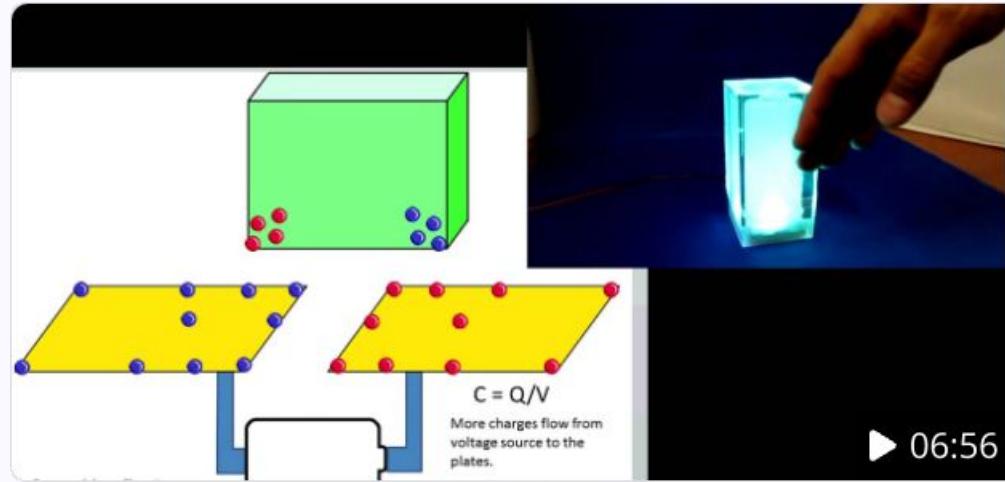
Applications

Touchscreens, proximity sensors, and more.

3

Sensitivity

Affected by sensor size and design.



YouTube

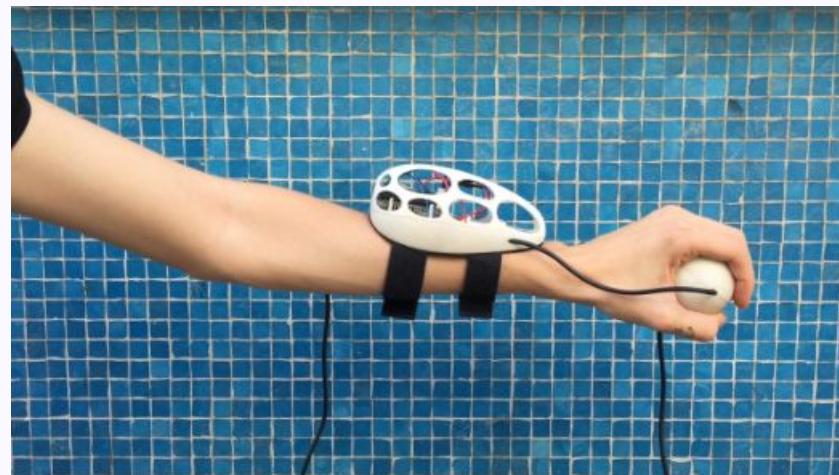


Capacitive sensor, Theory, application and design

This video explains the physics behind the surface capacitive sensors, with numerical results supporting the theory. A video for the projected capacitive senso...

Brainstorm





Lesson 4: Introduction to Arduino

What is it?

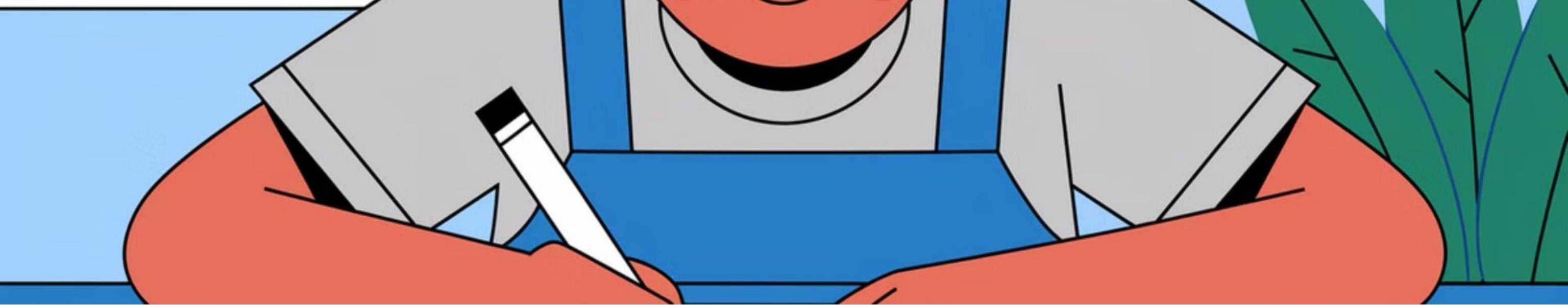
An open-source electronics development platform.

Why use it?

Easy to program and interface with sensors.

Lesson Overview

Setting up the Arduino environment.



Hands-on: Creating Simple Sensors



Drawing

Create patterns with conductive ink.



Testing

Measure touch sensitivity with a multimeter.



Connecting

Integrate with electronic components.

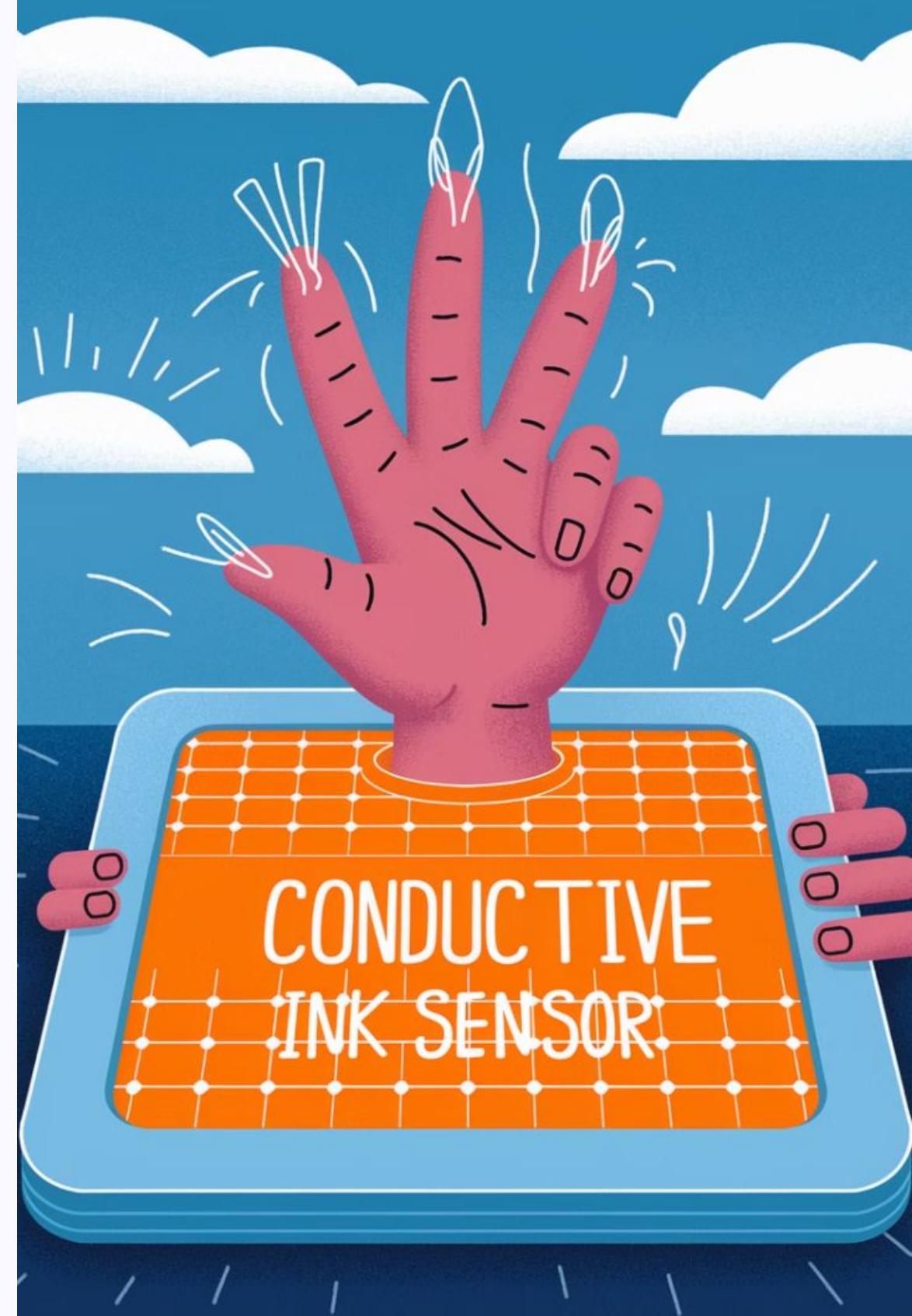
Exploring Gesture Recognition

- 1 Taps
- 2 Slides
- 3 Proximity

Single touch events for simple actions.

Continuous motion for parameter control.

Detecting hand distance for effects.





Concept Development

1

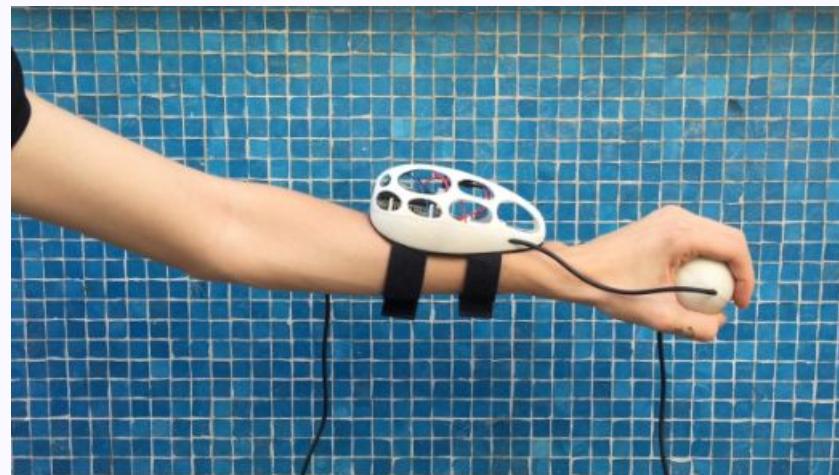
Brainstorming

Generate as many ideas as possible.

2

Sketching

Visualize your ideas on paper.





YouTube

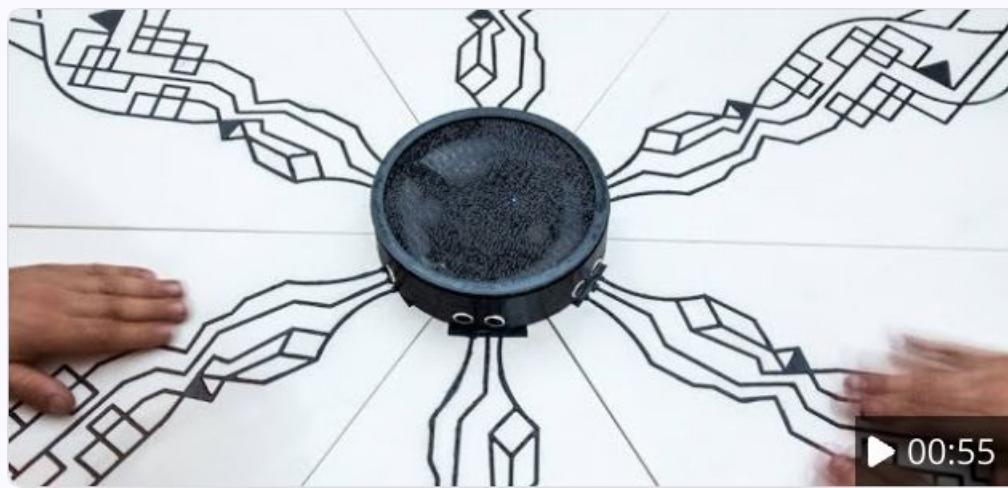


Electric Paint Lamp Kit

Find out more about the Electric Paint Lamp Kit here:

<https://www.bareconductive.com/collections/electric-paint-lamp-kit> Transform any...

▶ 01:1



YouTube



Build Your Own Interactive Sound Table with Electric Pai...

Find out more about Electric Paint here:

<https://www.bareconductive.com/collections/electric-paint> TACTO is an interactive...

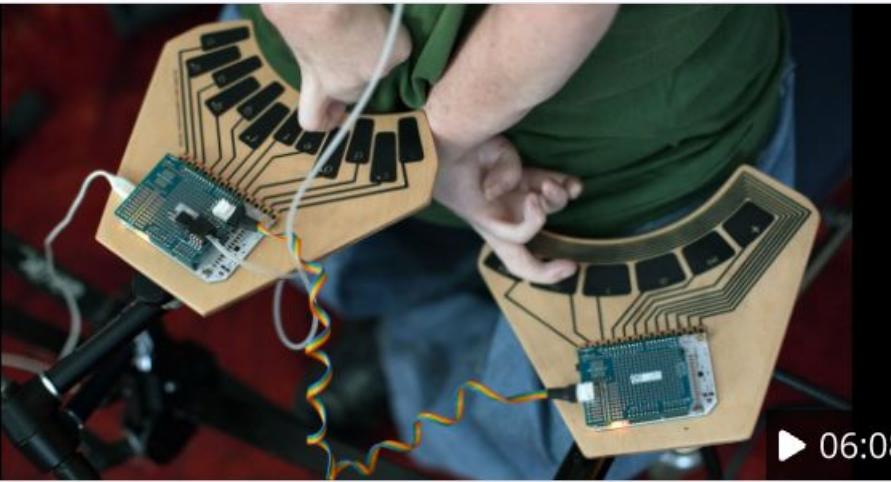


YouTube



Interactive media wall (touch sensor, conductive ink, projection map...)

[PLAYDODO] Music Playing Wall : Projection Mapping, Conductive Ink, Interacitve Art
Music Playing Wall, a fun Interactive touch wall painted with instruments recognize...



YouTube



Touch Chord - A Touch Sensitive Breath Controlled Instrument

Read the full story: <http://bit.ly/1Gg7gQ7> Designer Musician Vahagn Matossian from Human Instruments teams up with Bare Conductive to develop a new musical...

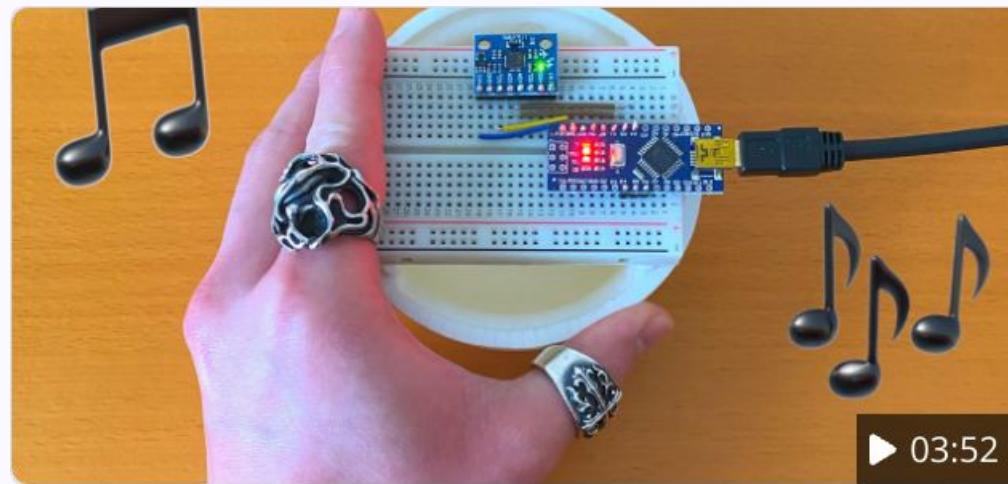


YouTube



I Built a MIDI THEREMIN! Theremidi - A DIY Arduino MIDI Controller

→ Download the files: <https://go.musicnerd.com/files-download> → Making Music with Arduino: <https://go.musicnerd.com/nerd-musician-pro/> → Curso Fazendo...



YouTube

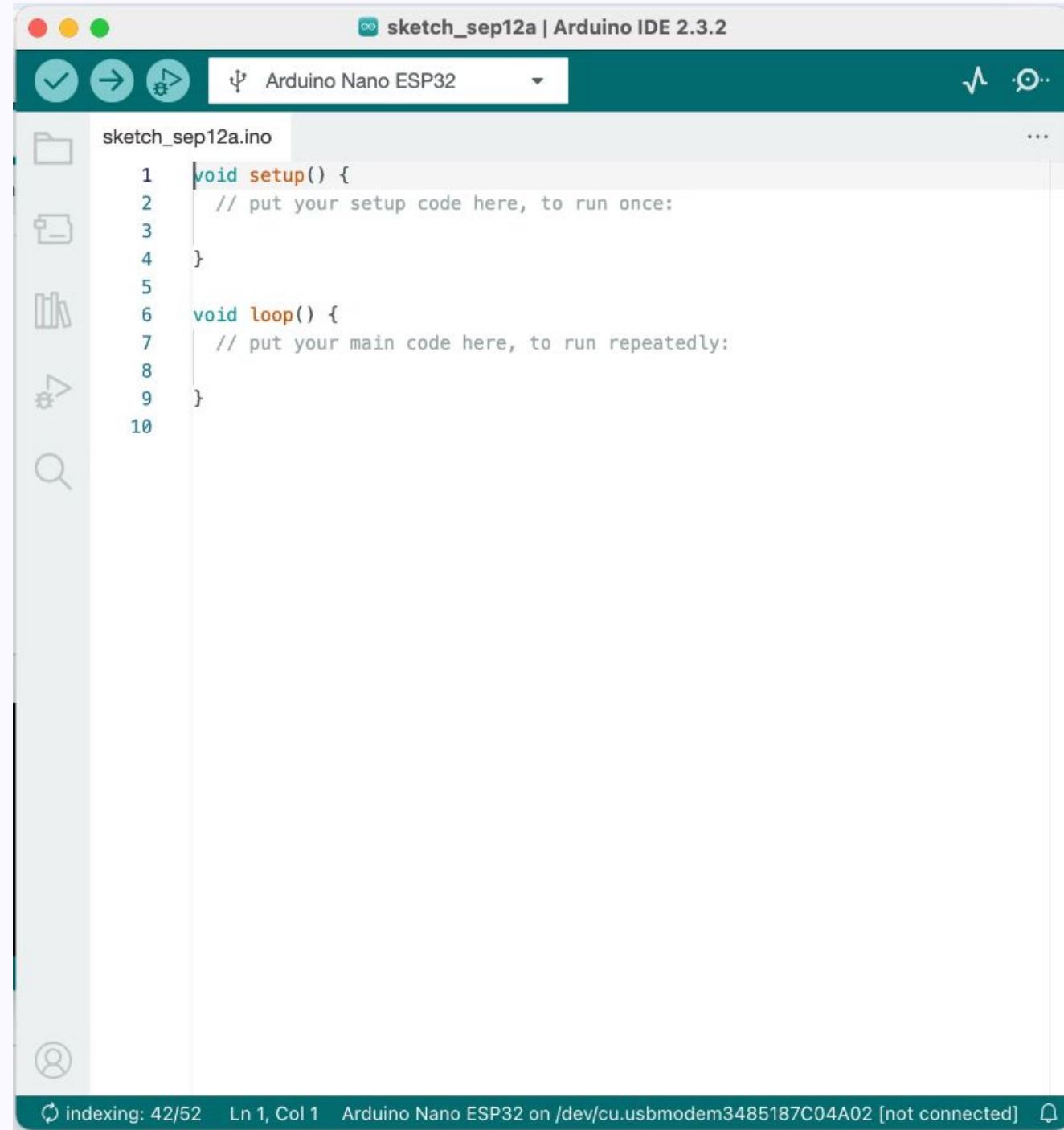


Accelerometer Music Controller (Arduino Tutorial)

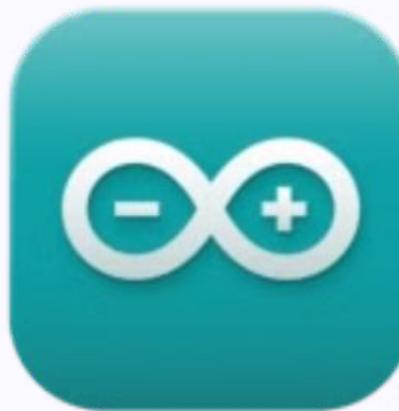
Let's use accelerometer to control sounds in Pure Data or Max/MSP!! --- To Buy -
Arduino Nano <https://www.amazon.com/ELEGOO-Arduino-ATmega328P-Without-Header/dp/B00VQHJLWU>

▶ 03:52

Lesson 5: Arduino Libraries for Sound



The screenshot shows the Arduino IDE interface. The title bar reads "sketch_sep12a | Arduino IDE 2.3.2". The central area displays the code for "sketch_sep12a.ino". The code contains two functions: "void setup()" and "void loop()". The "setup()" function has a comment: "// put your setup code here, to run once:". The "loop()" function also has a comment: "// put your main code here, to run repeatedly:". The code editor has syntax highlighting for keywords like "void", "setup", and "loop". The bottom status bar indicates "indexing: 42/52 Ln 1, Col 1" and "Arduino Nano ESP32 on /dev/cu.usbmodem3485187C04A02 [not connected]".



Arduino IDE (Integrated Development Environment)

Software application that provides everything needed to write, compile, and upload code to Arduino boards.

- Features code editor with syntax highlighting
- Includes compiler that converts your code into machine language
- Provides serial monitor for debugging
- Manages libraries for expanded functionality

Why Are Creative Coding Skills Still Important in the AI Era?



Understanding Fundamentals

AI can generate code, but understanding core principles of circuits, sensors, and programming enables you to customize instruments beyond templates and troubleshoot unexpected errors in your conductive ink projects.



Creative Problem-Solving

When building novel musical instruments like the Inkstrument, you'll encounter unique challenges that aren't in existing codebases or AI training data, requiring original solutions and adaptations.



Physical Computing Integration

Merging code with hardware like Arduino requires hands-on knowledge of how digital signals interact with analog materials like conductive ink, something AI assistance cannot fully replace.



Artistic Expression

Creative coding allows for unique artistic decisions in how gestures translate to sound, giving your instrument a signature character that generic AI-generated code cannot capture.

Balancing Fundamentals with Augmented Intelligence Tools for Creative

Troubleshooting Capability

Understanding circuit basics allows students to diagnose issues in their Inkstrument projects that AI might not recognize, especially when working with novel conductive materials.

Customization Beyond Templates

Knowledge of Arduino programming principles enables students to modify AI-generated code specifically for unique gesture-based sound control.

Material-Digital Integration

Grasping how conductive ink interacts with sensors allows students to create distinctive instruments while using AI to accelerate repetitive coding tasks.

Basic Arduino Programming

1

Structure

Sketches consist of setup() and loop().

2

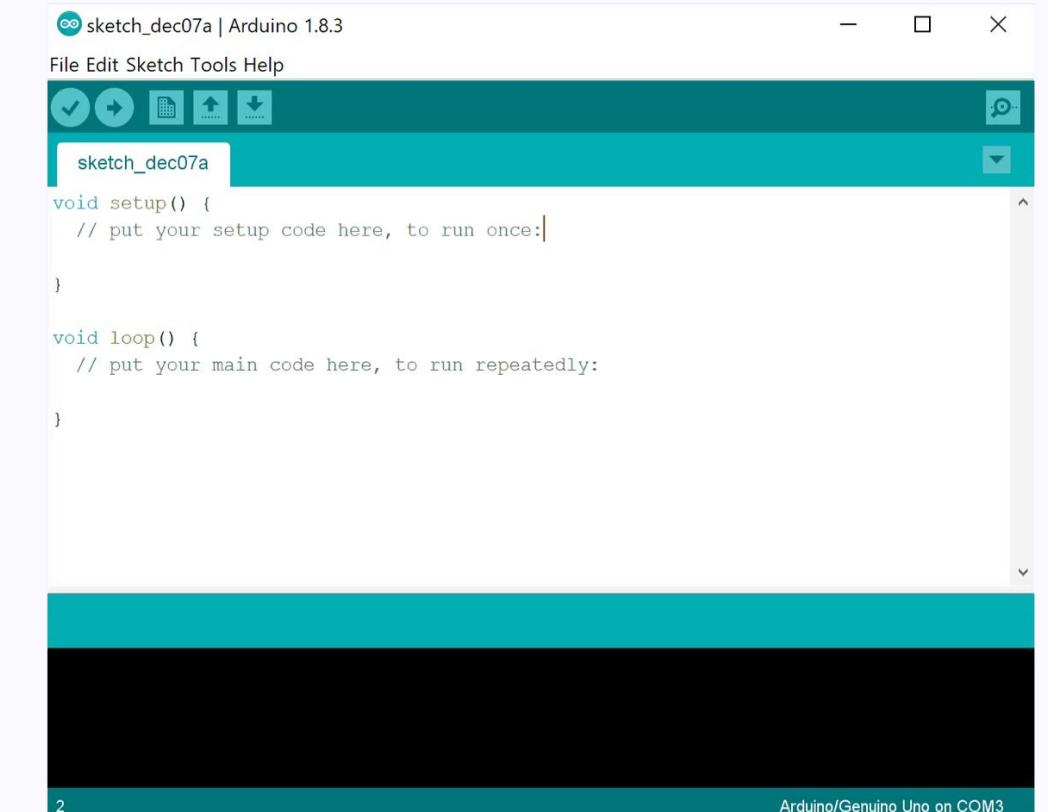
Setup

Initializes variables and pins.

3

Loop

Repeatedly executes the main code.



The screenshot shows the Arduino IDE interface with a sketch titled "sketch_dec07a". The code is as follows:

```
sketch_dec07a | Arduino 1.8.3
File Edit Sketch Tools Help
sketch_dec07a
void setup() {
  // put your setup code here, to run once:
}

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

The status bar at the bottom right indicates "Arduino/Genuino Uno on COM3".

setup() and loop()

setup()

Runs once when your Arduino powers on or resets.

- Initialize pin modes with pinMode(pin, INPUT/OUTPUT)
- Start serial communication with Serial.begin(9600)
- Set initial values for variables

```
void setup() {  
    pinMode(13, OUTPUT); // LED pin  
    Serial.begin(9600); // Start serial monitor  
    digitalWrite(13, LOW); // Initial state  
}
```

loop()

Runs continuously after setup() completes.

- Contains your main program logic
- Reads sensors and controls outputs
- Executes repeatedly until power off

```
void loop() {  
    digitalWrite(13, HIGH); // Turn LED on  
    delay(1000); // Wait 1 second  
    digitalWrite(13, LOW); // Turn LED off  
    delay(1000); // Wait 1 second  
}
```

Hands-on: Blinking LED



Connect LED

Wire an LED to an Arduino pin.



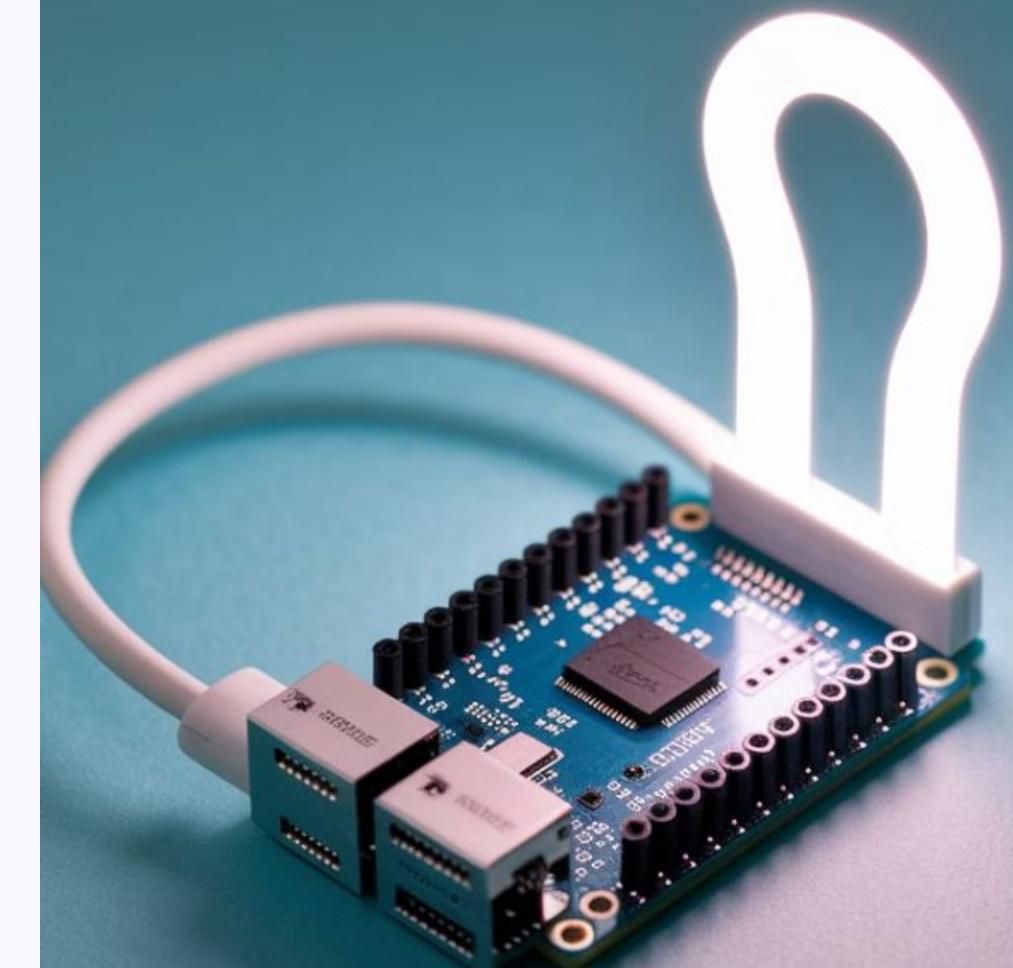
Write Code

Create a sketch to blink the LED.



Upload

Send the code to the Arduino board.



Blink

```
void setup() {  
    // initialize digital pin LED_BUILTIN as an output.  
    pinMode(LED_BUILTIN, OUTPUT);  
  
}  
  
// the loop function runs over and over again forever  
void loop() {  
    digitalWrite(LED_BUILTIN, HIGH);    // turn the LED on (HIGH is the voltage level)  
    delay(1000);                      // wait for a second  
    digitalWrite(LED_BUILTIN, LOW);     // turn the LED off by making the voltage LOW  
    delay(1000);                      // wait for a second  
}
```

What is Serial Monitor in Arduino?

A debugging tool that allows communication between Arduino and computer.

Purpose

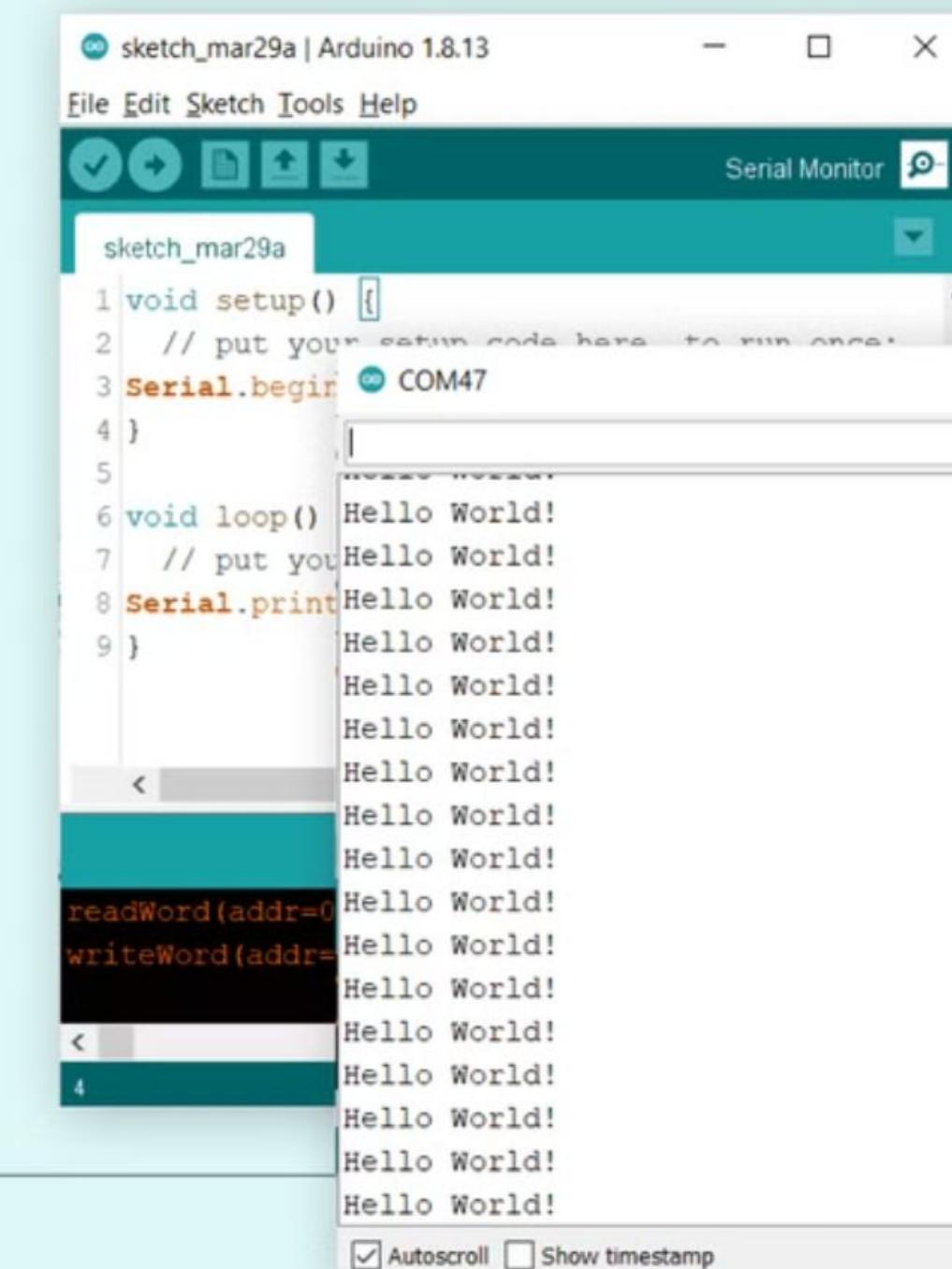
Displays data sent from Arduino to your computer, essential for troubleshooting and monitoring sensor values.

Usage

Accessed from Arduino IDE by clicking the magnifying glass icon or using Ctrl+Shift+M.

Configuration

Must match baud rate in code (e.g., `Serial.begin(9600)`) to properly display data.



The screenshot shows the Arduino IDE interface with the Serial Monitor window open. The title bar reads "sketch_mar29a | Arduino 1.8.13". The menu bar includes File, Edit, Sketch, Tools, and Help. The toolbar has icons for upload, refresh, and other functions. The Serial Monitor tab is selected, showing the port "COM47". The code in the editor is:

```
1 void setup() {
2 // put your setup code here - to run once:
3 Serial.begin(9600)
4 }
5
6 void loop() {
7 // put your main code here, to run repeatedly:
8 Serial.print("Hello World!")
9 }
```

The serial monitor window displays the output: "Hello World!" repeated multiple times. At the bottom, there are checkboxes for "Autoscroll" and "Show timestamp".



Introduction to Arduino Libraries

1

What Are They?

Collections of pre-written code.

2

Why Use Them?

Simplify complex tasks and save time.

3

Finding Them

Available online and through the Arduino IDE.

What are Programming Library?



Reusable Code Packages

Collections of pre-written code that perform specific functions, saving developers development time.



Arduino Library Structure

Arduino libraries typically consist of C++ header (.h) and implementation (.cpp) files, organized to provide easy-to-use functions.

Arduino Libraries we use



Tone Library

Generates square wave tones on any Arduino pin, essential for creating musical notes in our Inkstrument project. Provides functions like tone() and noTone() for controlling pitch output.



Capacitive Sensor Library

Enables Arduino to detect touch through conductive materials. This library is critical for transforming our conductive ink drawings into interactive touch sensors.



SFEMP3Shield Library

Interfaces with the VS1053 MP3 decoder chip to play audio files stored on an SD card. Allows our Inkstrument to produce high-quality sounds beyond simple tones.



SdFat Library

Provides efficient and robust access to SD cards in FAT16 or FAT32 format. Essential for storing and retrieving sound samples and configuration files for our instrument.

Arduino Programming Fundamentals

Building our Inkstrument requires understanding key programming concepts in Arduino:



Variables & Data Types

Declaring variables with appropriate types (int, float, boolean) to store sensor readings from our conductive ink interfaces



Global Variables

Using global variables accessible across the entire program to manage state for our touch sensors and sound playback features



Library Imports

Including external .h library files for specialized functions like the Capacitive Sensor and SFEMP3Shield libraries we downloaded for touch detection and audio playback

touch_to_play_mp3

```
// Include necessary libraries for SD card, utility functions, and MP3 player shield.  
#include <SdFat.h>  
#include <SdFatUtil.h>  
#include <SFE_MP3HAT_Shield.h>  
  
// Create instances of the SdFat (SD card) and SFEMP3Shield (MP3 player) classes.  
SdFat sd;  
SFEMP3Shield MP3player;  
  
// Variables to store analog readings from pins A0 to A5.  
int InData0 = 0; // Analog reading from pin A0  
int InData1 = 0; // Analog reading from pin A1  
int InData2 = 0; // Analog reading from pin A2  
int InData3 = 0; // Analog reading from pin A3  
int InData4 = 0; // Analog reading from pin A4  
int InData5 = 0; // Analog reading from pin A5  
int InData0 = 0;  
  
// Sensitivity threshold for detecting touch inputs.  
int TouchSensitivity = 280;
```

.h C++ library in Arduino

Header (.h) files in Arduino are C++ libraries that contain pre-written code, function definitions, and variable declarations that extend Arduino's functionality.

For our Inkstrument project, we're using two critical libraries:

The **CapacitiveSensor library** (.h) that enables our Arduino to detect touch through conductive ink patterns

The **SFEMP3Shield library** (.h) that provides functions for playing audio files from the VS1053 MP3 decoder

touch_to_play_mp3

```
void setup() {  
    // Initialize serial communication at 115200 baud rate for debugging purposes.  
    Serial.begin(115200);  
  
    // Initialize the SD card using pin 9 as the chip select pin (CS).  
    // If the SD card fails to initialize, halt the program and display an error.  
    if (!sd.begin(9, SPI_HALF_SPEED)) sd.initErrorHalt();  
  
    // Change directory to the root folder ("/") on the SD card.  
    // If the directory change fails, halt the program and display an error.  
    if (!sd.chdir("/")) sd.errorHalt("sd.chdir(\"/\")");  
  
    // Initialize the MP3 player and set the volume level to maximum (10, 10).  
    MP3player.begin();  
    MP3player.setVolume(10, 10);  
  
    // Configure pins A0 to A5 as input pins to read analog values for touch sensing.  
    for (int i = A0; i <= A5; i++) {  
        pinMode(i, INPUT);  
    }  
  
    // Disable Timer0 overflow interrupt to avoid conflicts with other operations.  
    TIMSK0 &= !(1 << TOIE0);  
}
```

Serial.begin 115200 vs 9600 ?

Serial communication in Arduino requires setting a baud rate - the speed at which data is transmitted between the Arduino and your computer.

115200 baud

- Faster data transmission (12x faster than 9600)
- Better for high-volume data like sensor readings
- Ideal for our Inkstrument's real-time responsiveness
- What we use in our touch_to_play_mp3 code

9600 baud

- Arduino's traditional default rate
- More stable for basic projects
- Sufficient for simple debugging
- Slower, but compatible with older hardware

MP3player.begin() & MP3player.setVolume()

MP3player.begin()

Initializes the VS1053 MP3 decoder shield to prepare it for audio playback. This function:

- Sets up communication between Arduino and the VS1053 chip
- Configures internal settings for optimal audio performance
- Must be called before any other MP3player functions

MP3player.setVolume(10, 10)

Adjusts the output volume level of both left and right audio channels:

- Parameters represent left and right channel volumes

touch_to_play_mp3

```
void loop() {  
    // Read analog values from pins A0 to A5 and invert them (higher values mean touch detected).  
    InData0 = 1024 - analogRead(A0); // Invert A0 reading  
    InData1 = 1024 - analogRead(A1); // Invert A1 reading  
    InData2 = 1024 - analogRead(A2); // Invert A2 reading  
    InData3 = 1024 - analogRead(A3); // Invert A3 reading  
    InData4 = 1024 - analogRead(A4); // Invert A4 reading  
    InData5 = 1024 - analogRead(A5); // Invert A5 reading  
  
    ...  
}
```

What is analogRead()?

The `analogRead()` function is an Arduino command that reads the value from a specified analog pin, converting voltage (0-5V) into integer values (0-1023).

Reading sensor inputs

In our Inkstrument project, we use `analogRead()` to measure the conductive ink sensor values from pins Ao-A5.

Value interpretation

Higher values (closer to 1023) indicate greater conductivity or touch intensity, which we invert ($1024 - \text{analogRead}()$) to make touch detection more intuitive.

Touch sensitivity threshold

We compare these analog readings against our `TouchSensitivity` value (280) to determine when a conductive ink pad is being touched.

This function is essential for converting the analog electrical signals from our conductive ink sensors into digital values our Arduino can use to trigger sounds.

Why 1024 - analogRead()?

In our Inkstrument project, we use the formula **1024 - analogRead()** to invert the sensor readings from our conductive ink touch pads. This inversion serves three critical purposes:

Intuitive Touch Response

Without inversion, a stronger touch would produce a lower value (closer to 0) because our conductive ink creates a path to ground when touched. By inverting with `1024 - analogRead()`, higher values (closer to 1024) now represent stronger touches.

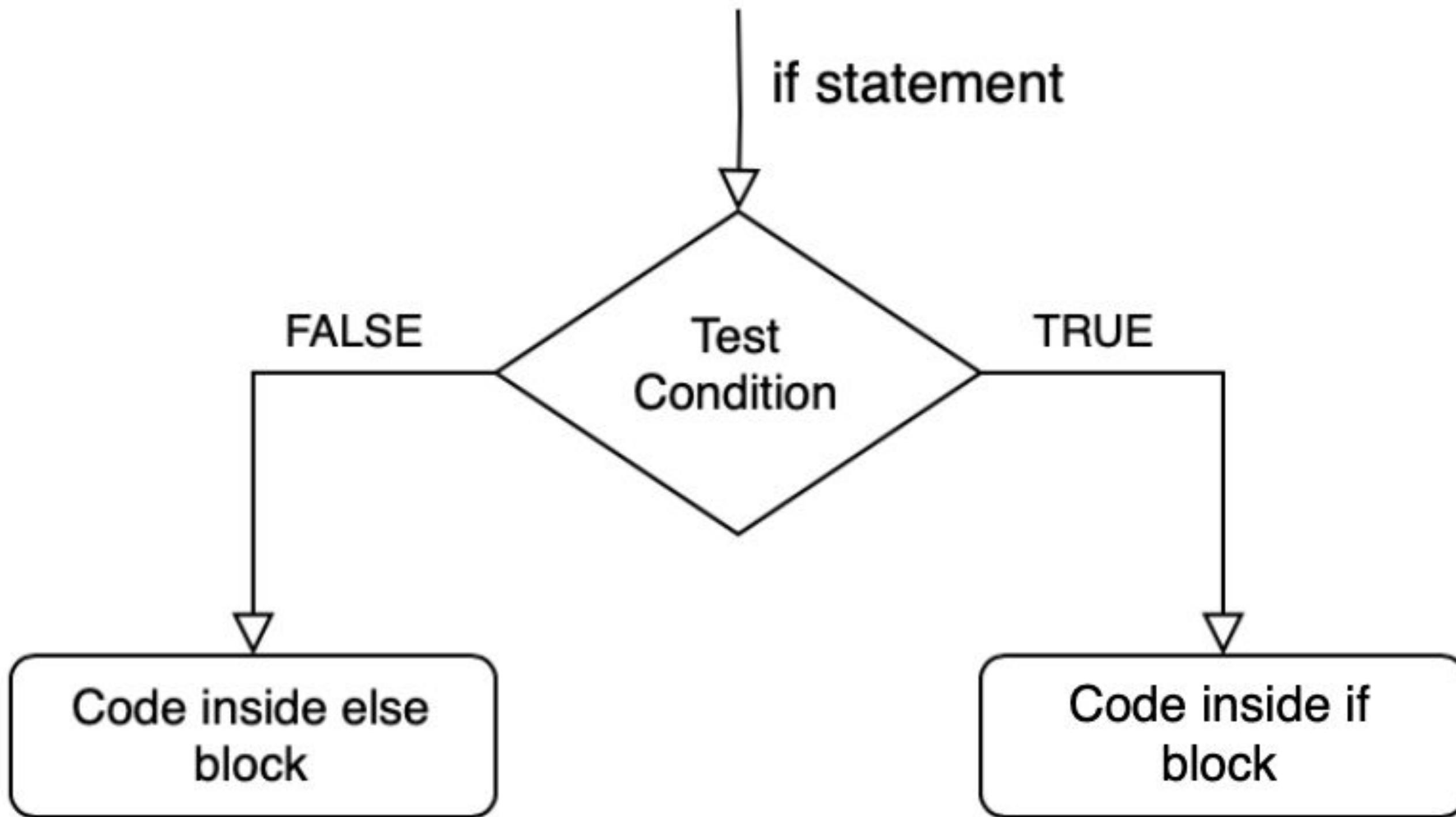
touch_to_play_mp3

```
void loop() {
    ...
    // Check which pin has been touched based on the sensitivity threshold.
    // Play the corresponding MP3 track if a touch is detected.
    if (InData0 >= TouchSensitivity) {
        MP3player.playTrack(0); // Play track 0 if A0 is touched
    } else if (InData1 >= TouchSensitivity) {
        MP3player.playTrack(1); // Play track 1 if A1 is touched
    } else if (InData2 >= TouchSensitivity) {
        MP3player.playTrack(2); // Play track 2 if A2 is touched
    } else if (InData3 >= TouchSensitivity) {
        MP3player.playTrack(3); // Play track 3 if A3 is touched
    } else if (InData4 >= TouchSensitivity) {
        MP3player.playTrack(4); // Play track 4 if A4 is touched
    } else if (InData5 >= TouchSensitivity) {
        MP3player.playTrack(5); // Play track 5 if A5 is touched
    } else {
        MP3player.stopTrack(); // Stop playback if no touch is detected
    }
    ...
}
```

Conditionals if(){else{}}

Conditional statements are crucial for our Inkstrument's interactivity. They allow the Arduino to make decisions based on sensor input values.

```
if (InData0 >= TouchSensitivity) {  
    MP3player.playTrack(0); // Play first sound if touched  
} else if (InData1 >= TouchSensitivity) {  
    MP3player.playTrack(1); // Play second sound if touched  
} else {  
    MP3player.stopTrack(); // Otherwise stop all sounds  
}
```



touch_to_play_mp3

```
void loop() {  
    ...  
    ...  
    // Print the current state of the MP3 player and analog readings to the serial monitor.  
    Serial.print(MP3player.isPlaying()); // Print whether the MP3 player is currently playing  
    Serial.print(" ");  
    Serial.print(InData0); // Print inverted value of A0  
    Serial.print(" ");  
    Serial.print(InData1); // Print inverted value of A1  
    Serial.print(" ");  
    Serial.print(InData2); // Print inverted value of A2  
    Serial.print(" ");  
    Serial.print(InData3); // Print inverted value of A3  
    Serial.print(" ");  
    Serial.print(InData4); // Print inverted value of A4  
    Serial.print(" ");  
    Serial.println(InData5); // Print inverted value of A5 followed by a newline  
  
    // Add a delay to prevent rapid polling of the touch pins.  
    delay(100);  
}
```

Serial.print()

Serial.print() is essential for debugging our Inkstrument. It sends data from Arduino to your computer, allowing you to observe sensor readings in real-time.

```
// Output touch sensor values to Serial Monitor  
Serial.print(InData0); // First ink sensor reading  
Serial.print(" "); // Add space between values  
Serial.print(InData1); // Second ink sensor reading  
Serial.println(InData2); // Add newline after last value
```

When designing your Inkstrument, use Serial.print() to:

- Calibrate touch sensitivity thresholds
- Verify that your conductive ink paths are functioning
- Debug issues when sounds aren't triggering as expected

Open the Serial Monitor (Tools → Serial Monitor) with baud rate set to 9600 to view the output.

Lesson 6 & 7: Building and Programming

Concept

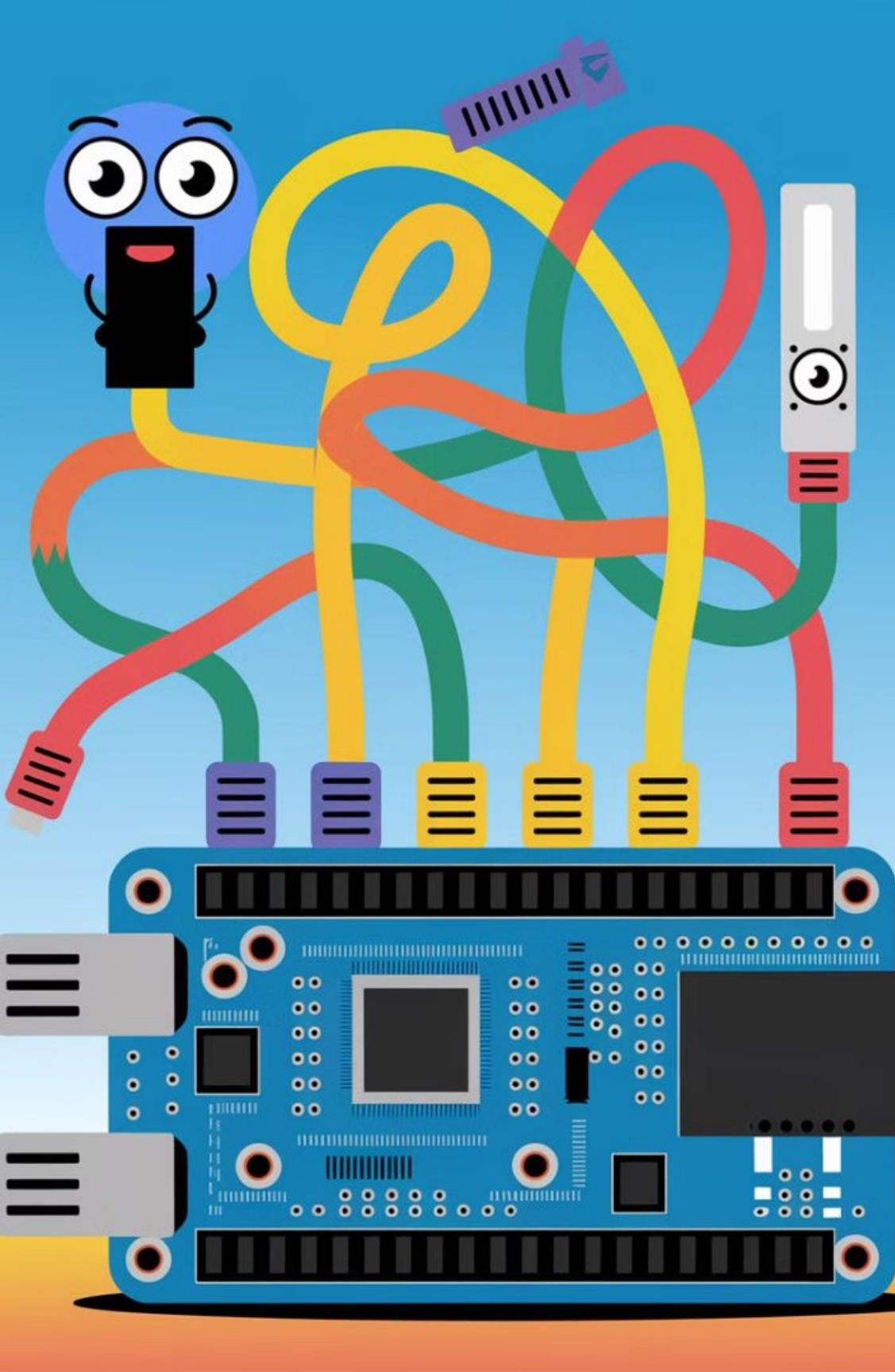
Brainstorming interactive music ideas.

Materials

Selecting the right conductive ink or objects.

Prototyping

Creating a paper prototype for testing.



Wiring the Arduino

- 1
- 2
- 3

Connections

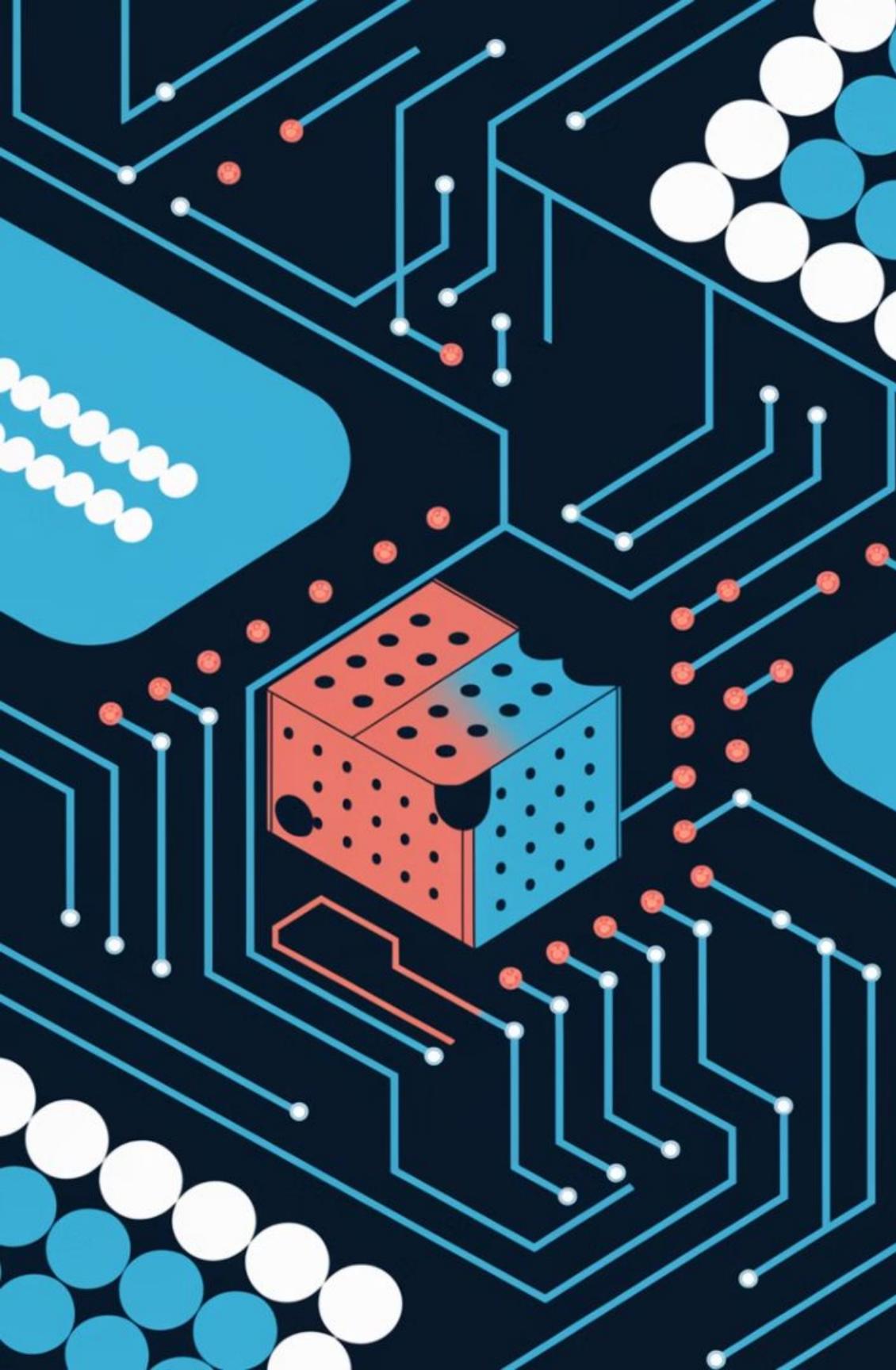
Connecting sensors to Arduino pins correctly.

Power

Ensuring stable power supply for the instrument.

Testing

Verifying all connections before programming.



Circuit Design



Layout

Plan the conductive ink patterns carefully.

Integration

Connect to Arduino pins for sensing.

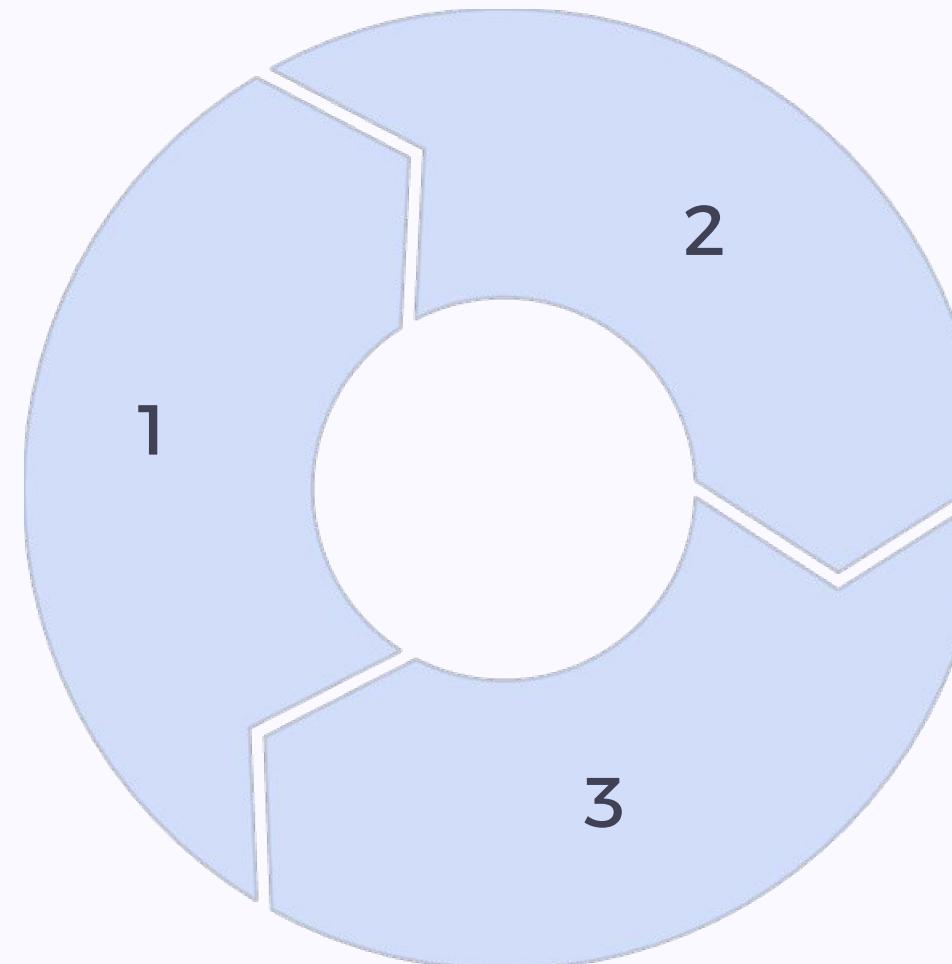
Testing

Ensure all connections are working properly.

Sound Design

Parameters

Choose synth parameters to control.



Mapping

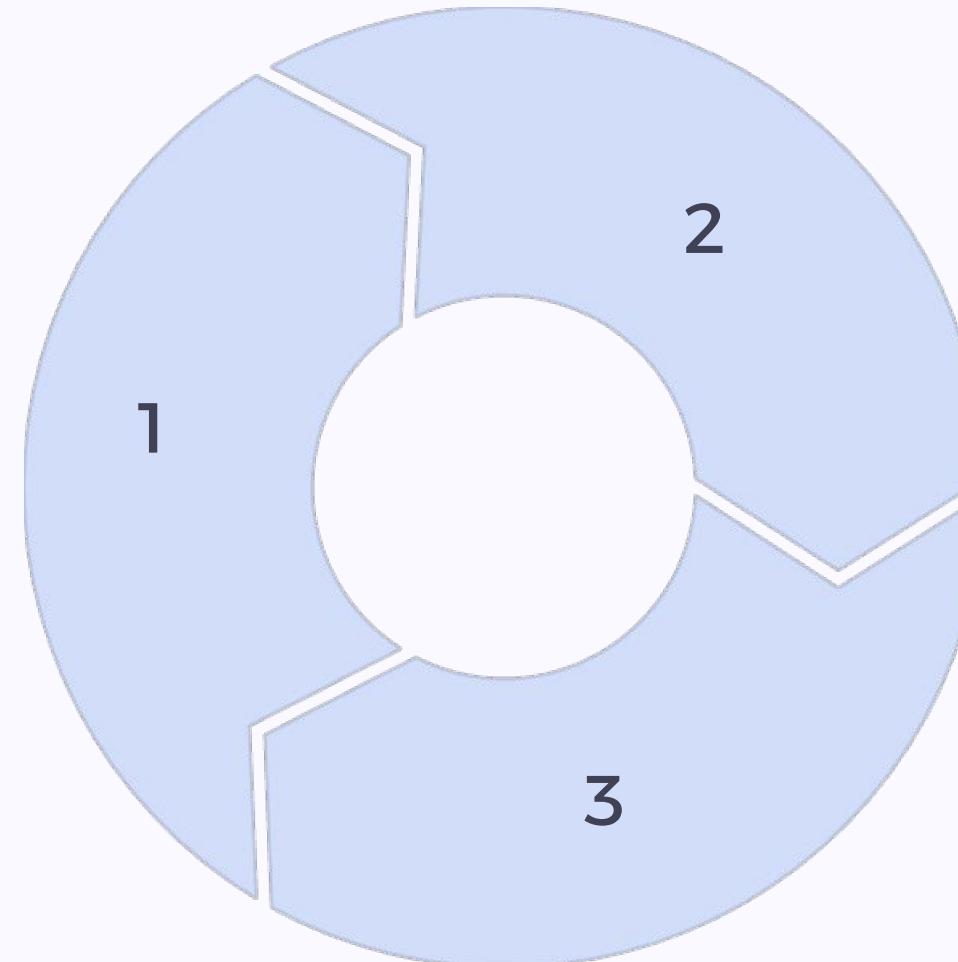
Link gestures to sound variations.

Experimentation

Fine-tune the sound for desired effects.

Programming

Main Sketch
Writing the Arduino code to control
the instrument.



- Recognition**
identify Typical problems with code
and solutions.
- Optimization**
Tuning the code for responsiveness.



Testing and Debugging

Systematic

Test each function thoroughly.

Common

Identify typical issues and their solutions.

Refinement

Fine-tune for optimal performance.

Project Showcase and Reflection



Thank You and Q&A