

コードとデザイン

東京藝術大学 芸術情報センター開設科目 金曜4-5限 第7週

2023.05.26 松浦知也 (matsura.tomoya@noc.geidai.ac.jp teach@matsuuratomo.ya.com)



本日のスケジュール

- Arduinoでライブラリを使ってみよう
- 自分だけのコンピューター用入力装置を作ってみよう

ライブラリのインストール

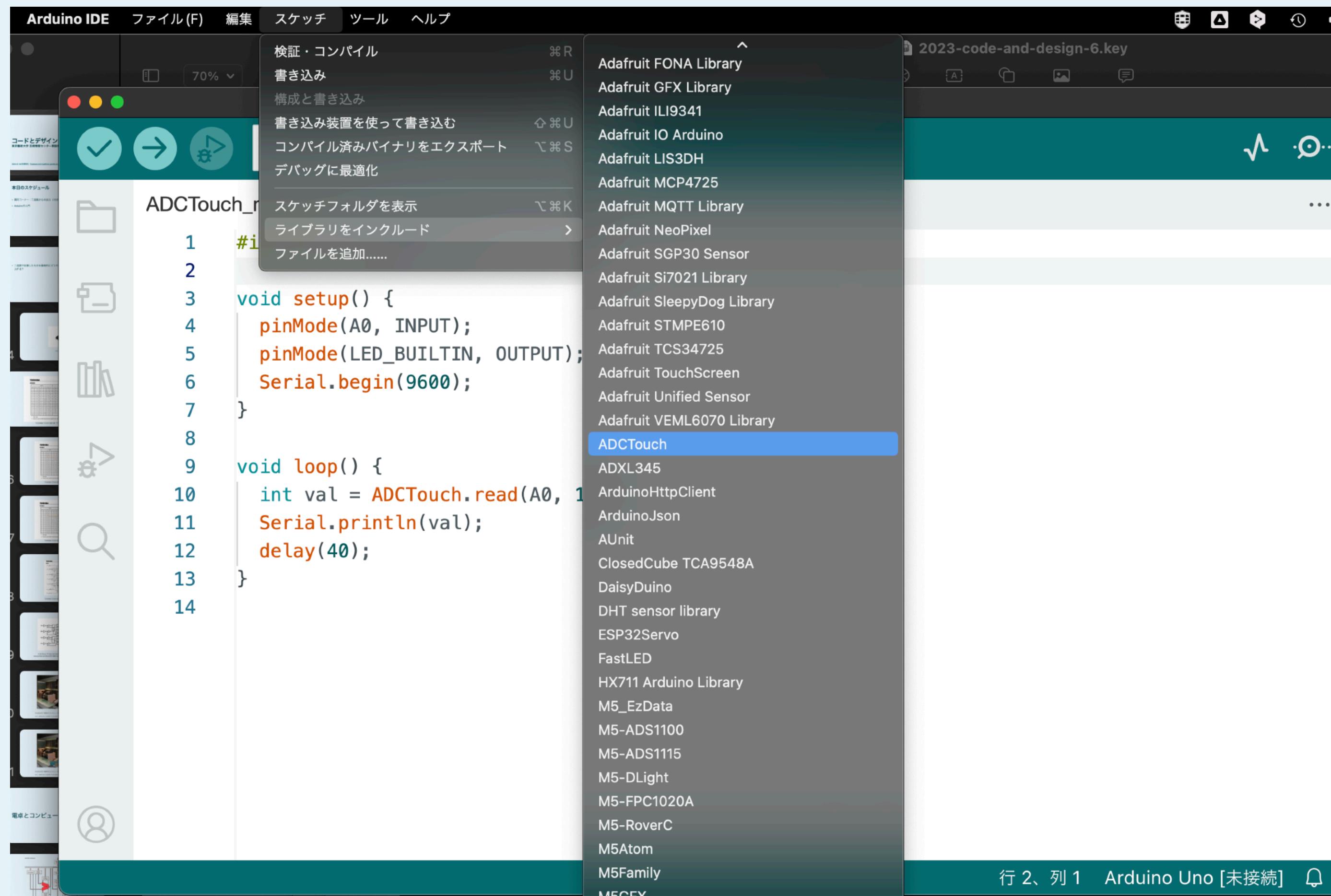
ライブラリ

- よく使う機能を再利用可能な形でまとめたもの
- Arduino IDEではライブラリマネージャーから、公式/有志の作ったライブラリをインストール可能
- インストールしたものは、`#include <LibraryName.h>`のようにソースコード先頭に記述することで使用可能になる
- 今回は例として、アナログ入力にワイヤを挿すだけでタッチセンサができる“ADCTouch”ライブラリを使ってみる

検索



*AMCの端末はシャットダウン時にインストールしたライブラリが消えます



スケッチ>ライブラリをインクルード で#includeするべきファイル名を自動で入れてくれる

```
#include <ADCTouch.h>

void setup() {
    pinMode(A0, INPUT);
    Serial.begin(9600);
}

void loop() {
    int val = ADCTouch.read(A0, 100); // (使用するピン番号, 平均を取るサンプル数)
    Serial.println(val);
    delay(40);
}
```

A0にワイヤを刺して、触ってみよう
シリアルプロッタで値の範囲を確認してみよう

スケッチのアップロード時や、
書き込み後の電源立ち上げ時（**setup**
関数が走るとき）に、
ワイヤを触ってると範囲が狂うことが
あるので注意

```
#include <ADCTouch.h>
//setup()とloop()で共通して使う値はグローバル変数(関数の外側)で定義しておく
int ref = 0;

void setup() {
    pinMode(A0, INPUT);
    pinMode(LED_BUILTIN, OUTPUT);
    Serial.begin(9600);
    //繋げた導体の状態で初期値が変わるので、それを保存しておく必要がある
    ref = ADCTouch.read(A0, 100);
}

void loop() {
    //初期値分を引き算してオフセット
    int val = ADCTouch.read(A0, 100) - ref;
    if (val > 40) {
        digitalWrite(LED_BUILTIN, HIGH);
    } else {
        digitalWrite(LED_BUILTIN, LOW);
    }

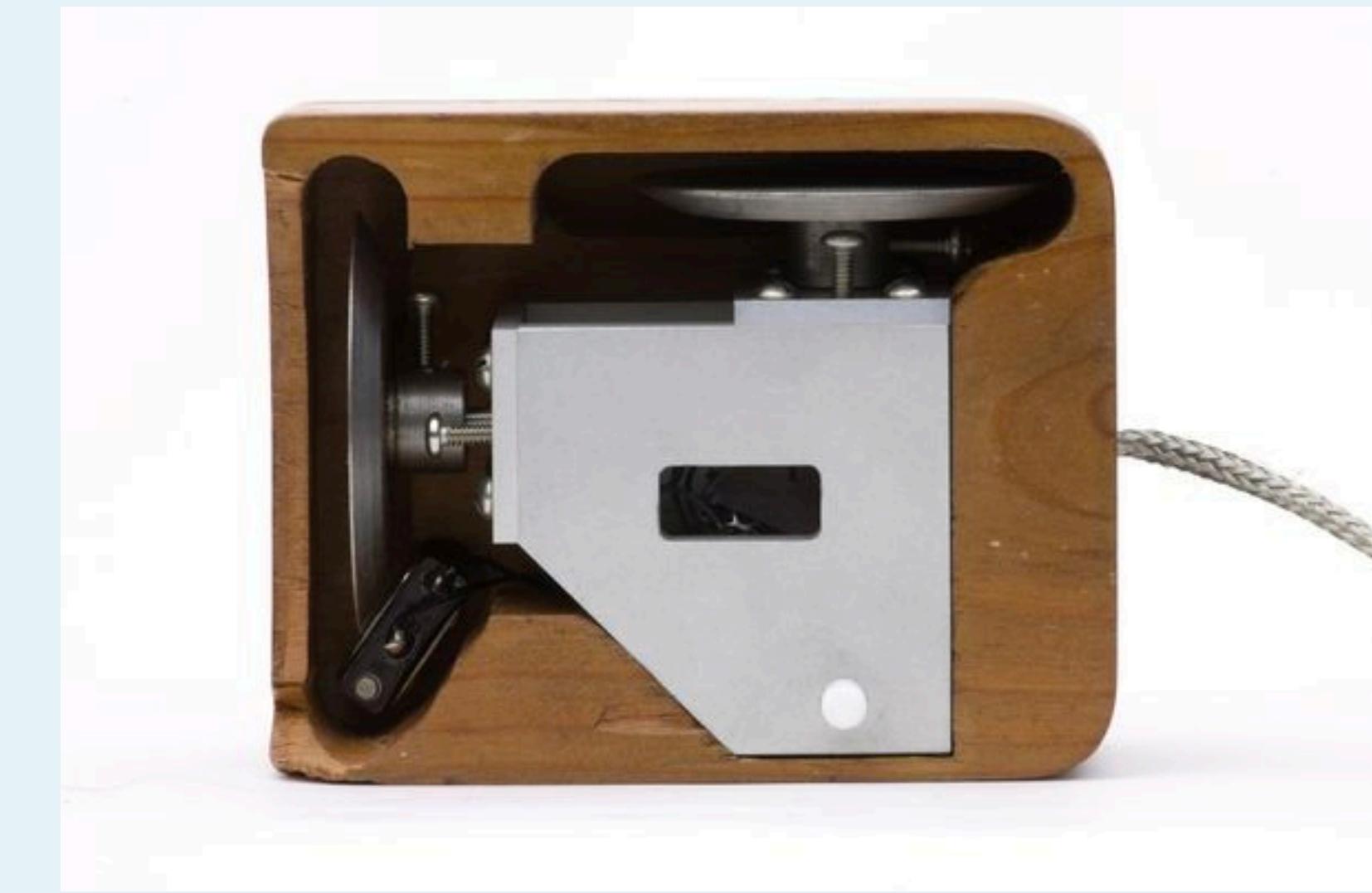
    Serial.println(val);

    delay(40);
}
```

不条理なマウスを作る



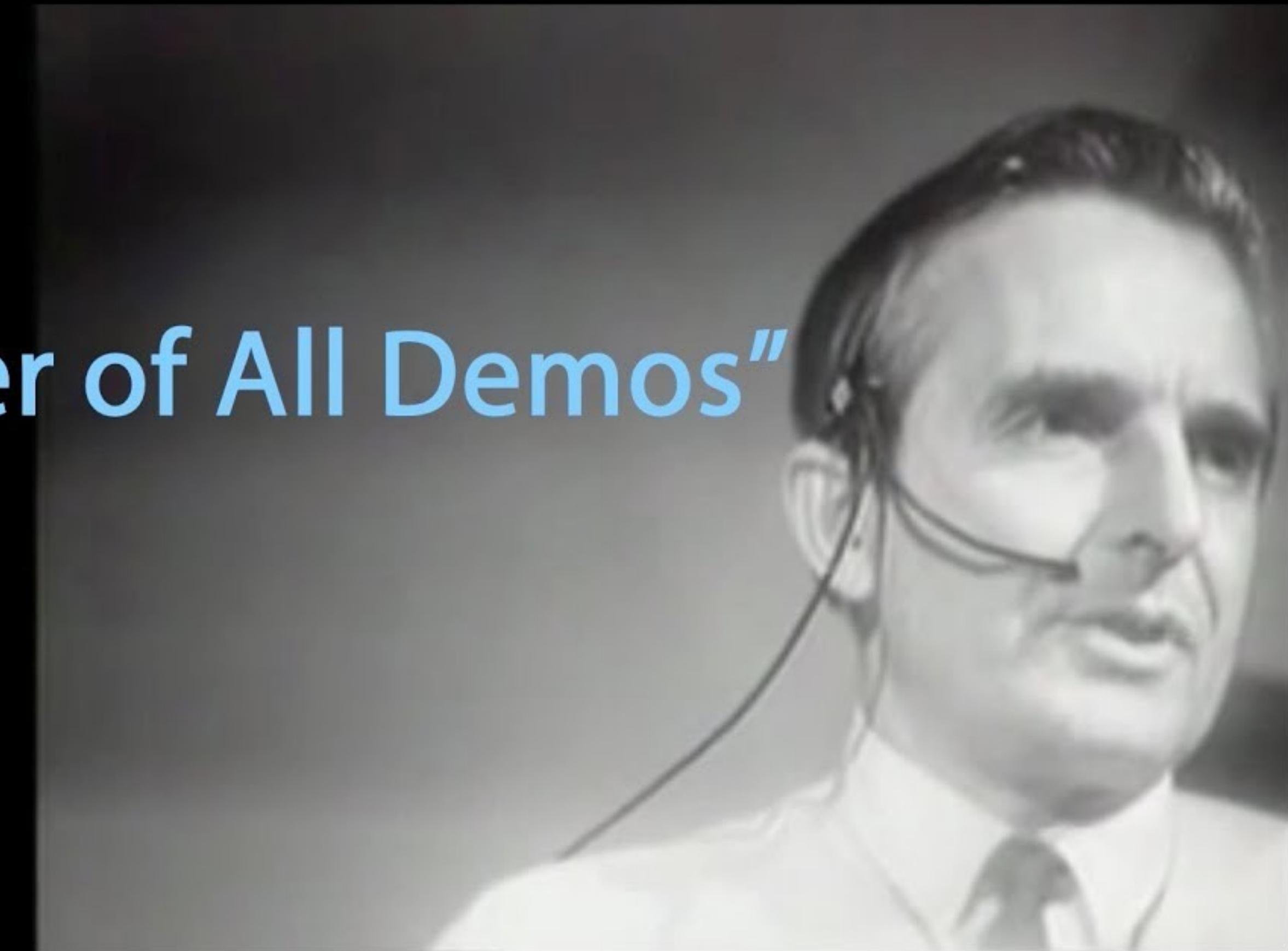
DATAR trackball(1952) - Computer History Museum, Courtesy of J. Vardalas
<https://www.computerhistory.org/revolution/input-output/14/350/1881>



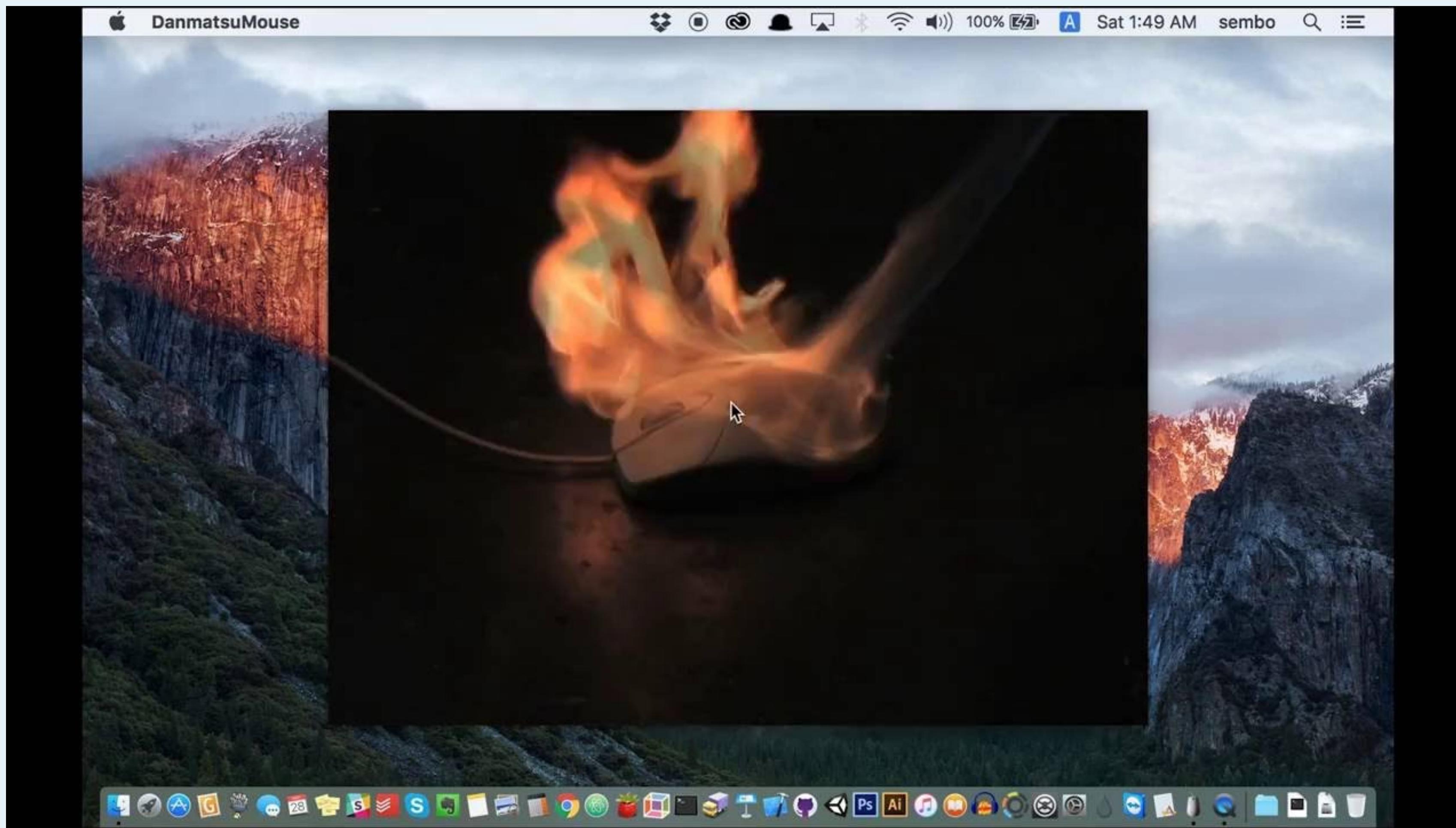
Prototype Engelbart mouse (replica), Computer History Museum, © Mark Richards

<https://www.computerhistory.org/revolution/input-output/14/350/1546?position=3>

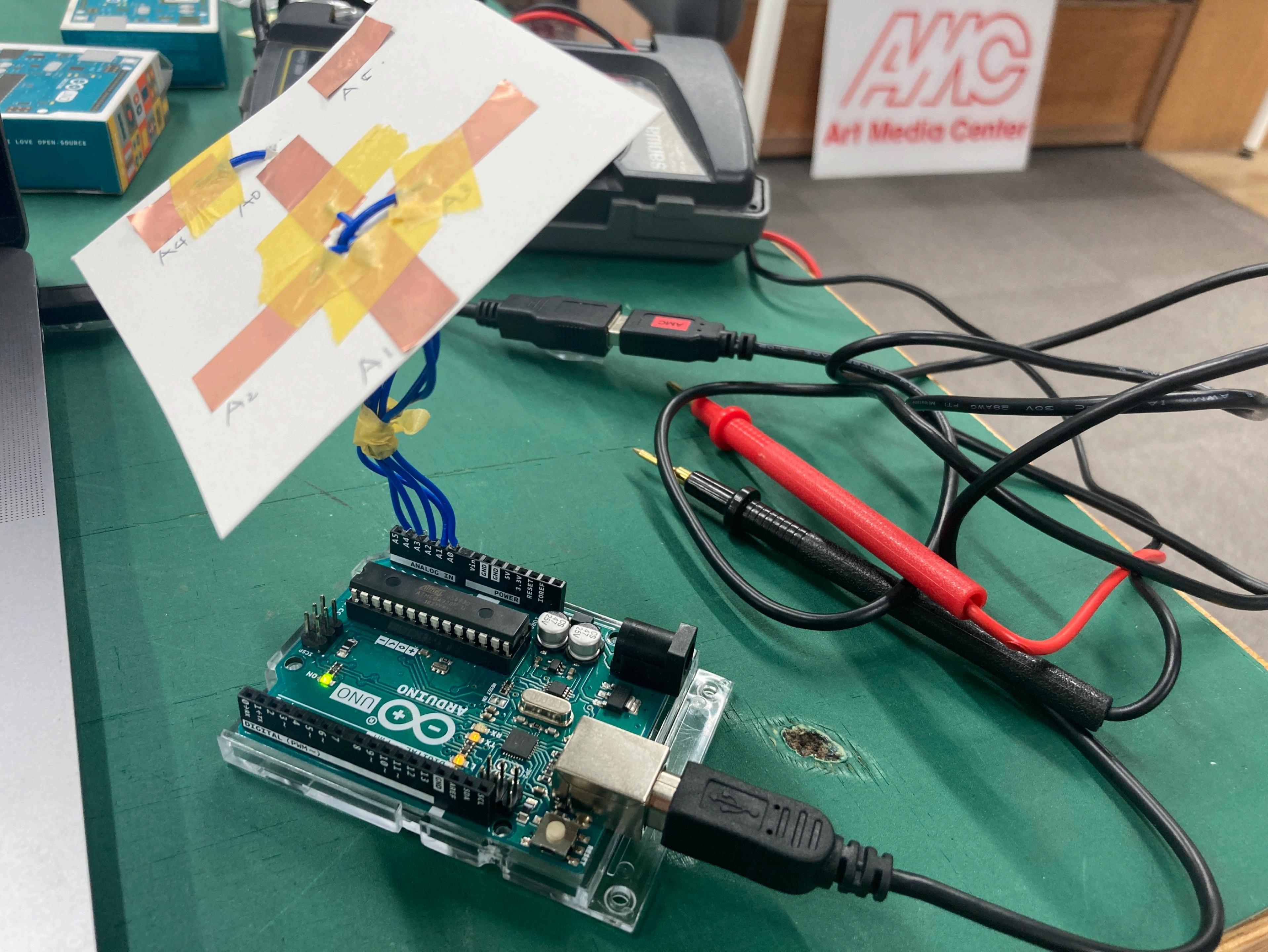
“The Mother of All Demos”

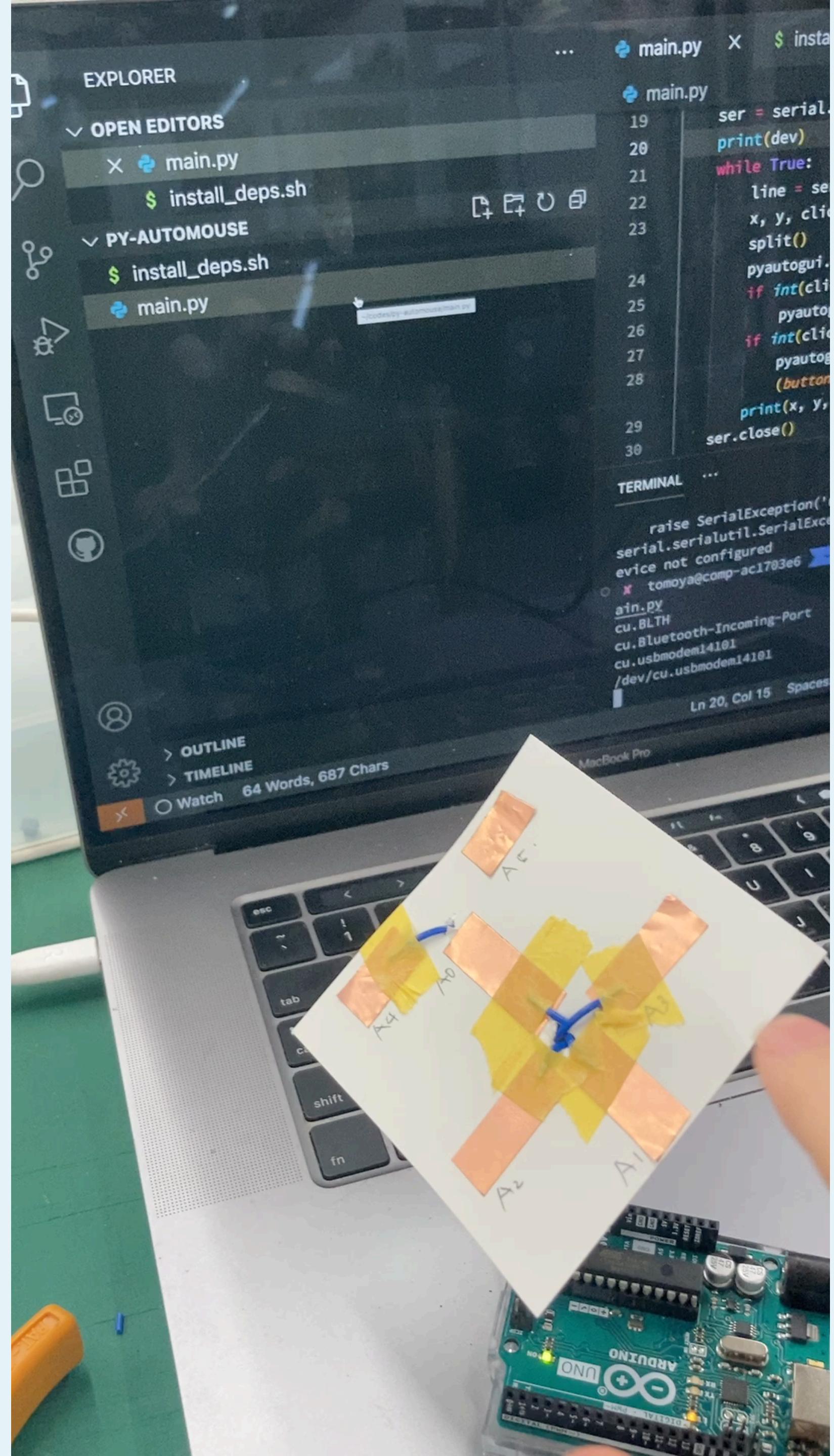


1968 “Mother of All Demos” by SRI’s Doug Engelbart and Team
<https://www.youtube.com/watch?v=B6rKUf9DWRI>



DanmatsuMouse(2007), exonemo,
<http://exonemo.com/works/danmatsumouse/?ja>





適当のごった煮

Pythonと境界標とQGISを中心にいろいろと

Arduino Nano(Uno)でジョイスティックマウスを作る

Arduino 工作 Python

スポンサードリンク

2018-11-08

Arduino Nanoを使ったジョイスティックマウスを作りました。Arduino Unoでも動作しました。



youtu.be

実体部分は写真のような感じです。

検索

最新記事

[Python Sphinxの使い方を学び始める](#)

[Python timeモジュールのまとめ](#)

[Pythonで横断検索スクリプトを作る](#)

[JR東日本 みどりの窓口がある駅を地図上で表示する](#)

[令和元年台風19号の堤防決壊箇所一覧を地図にする](#)

月別アーカイブ

▶ [2020 \(3\)](#)

▶ [2019 \(8\)](#)

▼ [2018 \(12\)](#)

[2018 / 11 \(1\)](#)

[2018 / 10 \(2\)](#)

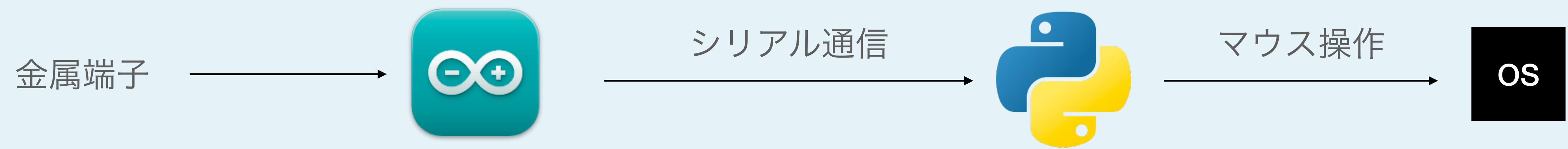
[2018 / 8 \(2\)](#)

Arduino Nano(Uno)でジョイスティックマウスを作る(2018) - 適当のごった煮

<https://www.tekito-gottani.com/entry/2018/11/08/142506>

Arduinoをマウスにする

- Pythonというプログラミング言語で、Arduinoからシリアル通信で送られた数値をもとに、マウスを強制的に動かします
- 今回はPythonのスクリプトは用意したので、中身は理解できなくてもOK（もちろん、わかる人は自分でカスタマイズしてもOK）
- Pythonではpyserialとpyautoguiというライブラリを使います
 - Arduino同様に事前にライブラリのインストールが必要です
- 今回はMacのみ対応ですが、PythonをインストールしてあればWinでも動くはず



半角スペース区切りで
x y click

と送ると、水平方向に $\pm x$ ピクセル、
垂直方向に $\pm y$ ピクセル移動
clickが1なら左クリック

Pythonスクリプトの準備

- `install_deps.command`を実行
- 自分のPCの人は、ターミナル.appを開いて、`pip3 install pyserial pyautogui`とタイプしてEnter
- Arduino IDEでシリアルプロッタやシリアルモニタを開いてないことを確認して、`move_relative.command`を実行
- ターミナルの人は`python3 move_relative.py`
- 右下にロケットのアイコンが出てればOK

逆に、このPythonアプリが立ち上がっている間は、Arduino IDEでの書き込みやシリアルモニタは使えない
ので注意

```
#!/usr/bin/python3
# -*- coding: utf-8 -*-
import serial
import serial.tools.list_ports
import pyautogui

run = False
# Arduinoをデバイス名から検出
info_list = serial.tools.list_ports.comports()
for info in info_list:
    print(info.name)
    if 'usbmodem' in info.name:
        run = True
        dev = "/dev/" + info.name

if run:
    ser = serial.Serial(dev)
    print(dev)
    while True:
        line = ser.readline()
        x, y, click = line.decode('shift-jis').split()
        pyautogui.moveRel(int(x), int(y))
        if int(click) == 1:
            pyautogui.click()
        if int(click) == 2:
            pyautogui.click(button="right")
        print(x, y, click)
    ser.close()
```

参考 : move_relative.py

Arduinoスケッチの準備

- ADCTouchで使うピンをA0~A4の5つに増やす
 - それぞれ上下左右、左クリックに機能割り当て
 - それぞれ初期化を忘れずに

```

#include <ADCTouch.h>

const int upButton = A0;
const int downButton = A1;
const int leftButton = A2;
const int rightButton = A3;
const int mouseButton = A4;
int ref_up = 0;
int ref_down = 0;
int ref_left = 0;
int ref_right = 0;
int ref_mouse = 0;

int range = 10;           // マウスカーソルの移動幅

void setup() {
    pinMode(upButton, INPUT);
    pinMode(downButton, INPUT);
    pinMode(leftButton, INPUT);
    pinMode(rightButton, INPUT);
    pinMode(mouseButton, INPUT);

    ref_up = ADCTouch.read(upButton, 100);
    ref_down = ADCTouch.read(downButton, 100);
    ref_left = ADCTouch.read(leftButton, 100);
    ref_right = ADCTouch.read(rightButton, 100);
    ref_mouse = ADCTouch.read(mouseButton, 100);

    Serial.begin(9600);
}

```

```

//マウス移動に関する処理をひとまとめに

void moveMouse(int x, int y, int mouse_state) {
    Serial.print(x);
    Serial.print(" ");
    Serial.print(y);
    Serial.print(" ");
    Serial.println(mouse_state);
}

void loop() {
    // ボタンの検出

    int v1 = ADCTouch.read(upButton, 100) - ref_up;
    int v2 = ADCTouch.read(downButton, 100) - ref_down;
    int v3 = ADCTouch.read(leftButton, 100) - ref_left;
    int v4 = ADCTouch.read(rightButton, 100) - ref_right;
    int v5 = ADCTouch.read(mouseButton, 100) - ref_mouse;
    //40以上ならクリックしたとみなし、1を、40以下なら0を
    // A ? B : C;はif (A) { return B;} else {return C;} 的なもの(三項演算子)
    int upState = v1 > 40 ? 1 : 0;
    int downState = v2 > 40 ? 1 : 0;
    int rightState = v3 > 40 ? 1 : 0;
    int leftState = v4 > 40 ? 1 : 0;
    int clickState = v5 > 40 ? 1 : 0;
    // 移動距離を算出(左右同時押しも考慮)
    int xDistance = (rightState - leftState) * range;
    int yDistance = (downState - upState) * range;

    // 移動もクリックもしていないならシリアル送信は冗長なので、しない
    // && は AND、!はNOT
    if (!(xDistance == 0 && yDistance == 0 && clickState == 0)) {
        moveMouse(xDistance, yDistance, clickState);
    }

    delay(5);
}

```

```

#include <ADCTouch.h>
定数(書き換えない変数) の定義
const int upButton = A0;
const int downButton = A1;
const int leftButton = A2;
const int rightButton = A3;
const int mouseButton = A4;
int ref_up = 0;
int ref_down = 0;
int ref_left = 0;
int ref_right = 0;
int ref_mouse = 0;

int range = 10;           // マウスカーソルの移動幅

void setup() {
    pinMode(upButton, INPUT);
    pinMode(downButton, INPUT);
    pinMode(leftButton, INPUT);
    pinMode(rightButton, INPUT);
    pinMode(mouseButton, INPUT);

    ref_up = ADCTouch.read(upButton, 100);
    ref_down = ADCTouch.read(downButton, 100);
    ref_left = ADCTouch.read(leftButton, 100);
    ref_right = ADCTouch.read(rightButton, 100);
    ref_mouse = ADCTouch.read(mouseButton, 100);

    Serial.begin(9600);
}

```

```

//マウス移動に関する処理をひとまとめに
void moveMouse(int x, int y, int mouse_state) {
    Serial.print(x);
    Serial.print(" ");
    Serial.print(y);
    Serial.print(" ");
    Serial.println(mouse_state);
}

void loop() {
    // ボタンの検出

    int v1 = ADCTouch.read(upButton, 100) - ref_up;
    int v2 = ADCTouch.read(downButton, 100) - ref_down;
    int v3 = ADCTouch.read(leftButton, 100) - ref_left;
    int v4 = ADCTouch.read(rightButton, 100) - ref_right;
    int v5 = ADCTouch.read(mouseButton, 100) - ref_mouse;
    //40以上ならクリックしたとみなし、1を、40以下なら0を
    // A ? B : C; は if (A) { return B;} else {return C;} の略記(三項演算子)
    int upState = v1 > 40 ? 1 : 0;
    int downState = v2 > 40 ? 1 : 0;
    int rightState = v3 > 40 ? 1 : 0;
    int leftState = v4 > 40 ? 1 : 0;
    int clickState = v5 > 40 ? 1 : 0;
    // 移動距離を算出(左右同時押しも考慮)
    int xDistance = (leftState - rightState) * range;
    int yDistance = (upState - downState) * range;

    // 移動もクリックもしていないならシリアル送信は冗長なので、しない
    // && は AND、!はNOT
    if (!(xDistance == 0 && yDistance == 0 && clickState == 0)) {
        moveMouse(xDistance, yDistance, clickState);
    }
    delay(5);
}

```

自分で作った関数の呼び出し

マウスを使って色々やってみる

- ネットサーフィン
- ブラウザゲーム
- Photoshopでお絵描き
-

マウスの挙動 자체をソフトウェアで変える

- 例えば、、、触ってなくてもランダムに動き続ける
-

```

#include <ADCTouch.h>

const int upButton = A0;
const int downButton = A1;
const int leftButton = A2;
const int rightButton = A3;
const int mouseButton = A4;
int ref_up = 0;
int ref_down = 0;
int ref_left = 0;
int ref_right = 0;
int ref_mouse = 0;

int range = 10;           // マウスカーソルの移動範囲

void setup() {
  randomSeed(analogRead(A5));
  pinMode(upButton, INPUT);
  pinMode(downButton, INPUT);
  pinMode(leftButton, INPUT);
  pinMode(rightButton, INPUT);
  pinMode(mouseButton, INPUT);

  ref_up = ADCTouch.read(upButton, 100);
  ref_down = ADCTouch.read(downButton, 100);
  ref_left = ADCTouch.read(leftButton, 100);
  ref_right = ADCTouch.read(rightButton, 100);
  ref_mouse = ADCTouch.read(mouseButton, 100);

  Serial.begin(9600);
}

```

```

//マウス移動に関する処理をひとまとめに

void moveMouse(int x, int y, int mouse_state) {
  Serial.print(x);
  Serial.print(" ");
  Serial.print(y);
  Serial.print(" ");
  Serial.println(mouse_state);
}

```

randomSeedに与える値が固定だと、
起動後毎回同じ疑似乱数の数列を返す

毎回異なる結果が欲しい場合は、
起動時に異なる値を返すもの...例えば
現在時刻や適当なセンサの値をSeedにする

```

int leftState = v4 > 40 ? 1 : 0;
int clickState = v5 > 40 ? 1 : 0;
// 移動距離を算出 (左右同時押しも考慮)
int xDistance = (leftState - rightState) * range + random(-10,10);
int yDistance = (upState - downState) * range + random(-10,10);

// 移動もクリックもしていないならシリアル送信は冗長なので、しない
// && は AND、!はNOT
if (!(xDistance == 0 && yDistance == 0 && clickState == 0)) {
  moveMouse(xDistance, yDistance, clickState);
}

delay(5);
}

```

```
#!/usr/bin/python3
# -*- coding: utf-8 -*-
import serial
import serial.tools.list_ports
import pyautogui

run = False
# Arduinoをデバイス名から検出
info_list = serial.tools.list_ports.comports()
for info in info_list:
    print(info.name)
    if 'usbmodem' in info.name:
        run = True
        dev = "/dev/" + info.name

if run:
    ser = serial.Serial(dev)
    print(dev)
    while True:
        line = ser.readline()
        x, y, click = line.decode('shift-jis').split()
        pyautogui.moveRel(int(x), int(y))
        if int(click) == 1:
            pyautogui.click()
        if int(click) == 2:
            pyautogui.click(button="right")
    print(x, y, click)
ser.close()
```

参考 : move_relative.py

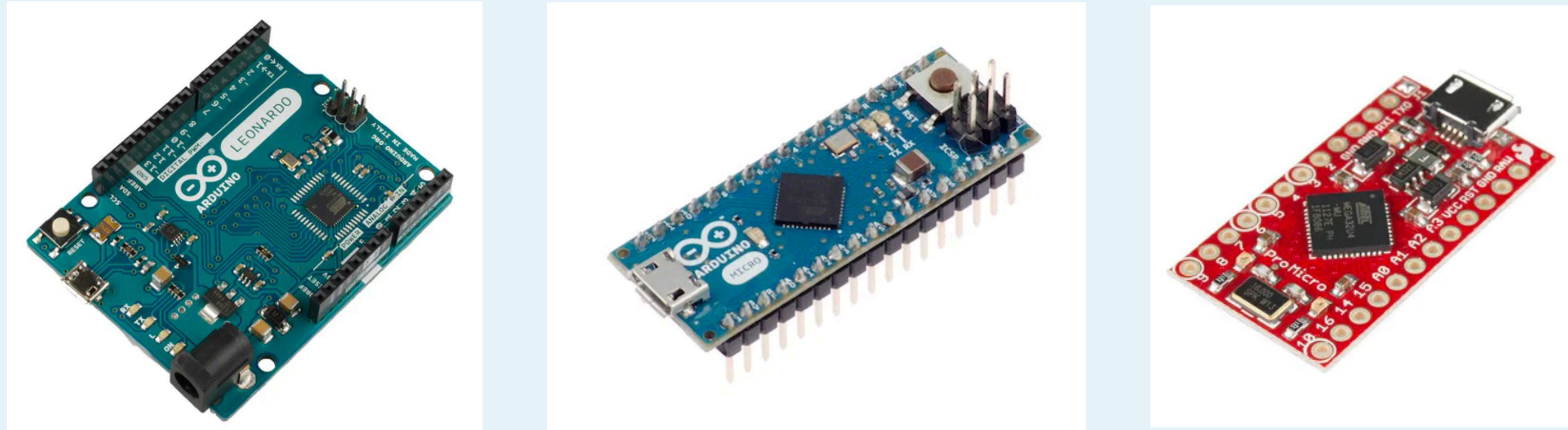
```
#!/usr/bin/python3
# -*- coding: utf-8 -*-
import serial
import serial.tools.list_ports
import pyautogui

run = False
info_list = serial.tools.list_ports.comports()

for info in info_list:
    print(info.name)
    if 'usbmodem' in info.name:
        run = True
        dev = "/dev/" + info.name
w,h = pyautogui.size();

if run:
    ser = serial.Serial(dev)
    print(dev)
    while True:
        line = ser.readline()
        x, y, click = line.decode('shift-jis').split()
        x_abs = float(x) * float(w) / 1024.0;
        y_abs = float(y) * float(h) / 1024.0;
        pyautogui.moveTo(int(x_abs), int(y_abs), duration=0.05)
        if int(click) == 1:
            pyautogui.click()
        if int(click) == 2:
            pyautogui.click(button="right")
        print(x, y, click)
    ser.close()
```

シリアルで “x y click”
x,yを0~1024で指定すると、
画面左上からの絶対位置で
カーソルを移動してくれる



Arduino Leonardo、Arduino Micro、SparkFun Pro Microなど、ATmega32u4が乗っているものは、"Mouse"ライブラリを使うことで本当にUSBマウスとして認識させられる
(実際自作キーボードはこの系列を使っていることが多い)

```

#include <Mouse.h>
#include <ADCTouch.h>

const int upButton = A0;
const int downButton = A1;
const int leftButton = A2;
const int rightButton = A3;
const int mouseButton = A4;
int ref_up = 0;
int ref_down = 0;
int ref_left = 0;
int ref_right = 0;
int ref_mouse = 0;

int range = 10;

void setup() {
    pinMode(upButton, INPUT);
    pinMode(downButton, INPUT);
    pinMode(leftButton, INPUT);
    pinMode(rightButton, INPUT);
    pinMode(mouseButton, INPUT);

    ref_up = ADCTouch.read(upButton, 100);
    ref_down = ADCTouch.read(downButton, 100);
    ref_left = ADCTouch.read(leftButton, 100);
    ref_right = ADCTouch.read(rightButton, 100);
    ref_mouse = ADCTouch.read(mouseButton, 100);

    Serial.begin(9600);
}

```

```

void moveMouse(int x, int y, int mouse_state) {
    Mouse.move(x, y, 0);
    if (mouse_state == 1) {
        if (!Mouse.isPressed(MOUSE_LEFT)) {
            Mouse.press(MOUSE_LEFT);
        }
    } else {
        if (Mouse.isPressed(MOUSE_LEFT)) {
            Mouse.release(MOUSE_LEFT);
        }
    }
}

void loop() {

    int v1 = ADCTouch.read(upButton, 100) - ref_up;
    int v2 = ADCTouch.read(downButton, 100) - ref_down;
    int v3 = ADCTouch.read(leftButton, 100) - ref_left;
    int v4 = ADCTouch.read(rightButton, 100) - ref_right;
    int v5 = ADCTouch.read(mouseButton, 100) - ref_mouse;
    int upState = v1 > 40 ? 1 : 0;
    int downState = v2 > 40 ? 1 : 0;
    int rightState = v3 > 40 ? 1 : 0;
    int leftState = v4 > 40 ? 1 : 0;
    int clickState = v5 > 40 ? 1 : 0;

    int xDistance = (leftState - rightState) * range;
    int yDistance = (upState - downState) * range;

    if (!(xDistance == 0 && yDistance == 0 && clickState == 0)) {
        moveMouse(xDistance, yDistance, clickState);
    }

    delay(5);
}

```

追加ライブラリのインポート

```
#include <Mouse.h>
#include <ADCTouch.h>

const int upButton = A0;
const int downButton = A1;
const int leftButton = A2;
const int rightButton = A3;
const int mouseButton = A4;
int ref_up = 0;
int ref_down = 0;
int ref_left = 0;
int ref_right = 0;
int ref_mouse = 0;

int range = 10;

void setup() {
    pinMode(upButton, INPUT);
    pinMode(downButton, INPUT);
    pinMode(leftButton, INPUT);
    pinMode(rightButton, INPUT);
    pinMode(mouseButton, INPUT);

    ref_up = ADCTouch.read(upButton, 100);
    ref_down = ADCTouch.read(downButton, 100);
    ref_left = ADCTouch.read(leftButton, 100);
    ref_right = ADCTouch.read(rightButton, 100);
    ref_mouse = ADCTouch.read(mouseButton, 100);

    Serial.begin(9600);
}
```

moveMouseの中身のみ書き換え

```
void moveMouse(int x, int y, int mouse_state) {
    Mouse.move(x, y, 0);
    if (mouse_state == 1) {
        if (!Mouse.isPressed(MOUSE_LEFT)) {
            Mouse.press(MOUSE_LEFT);
        }
    } else {
        if (Mouse.isPressed(MOUSE_LEFT)) {
            Mouse.release(MOUSE_LEFT);
        }
    }
}

void loop() {

    int v1 = ADCTouch.read(upButton, 100) - ref_up;
    int v2 = ADCTouch.read(downButton, 100) - ref_down;
    int v3 = ADCTouch.read(leftButton, 100) - ref_left;
    int v4 = ADCTouch.read(rightButton, 100) - ref_right;
    int v5 = ADCTouch.read(mouseButton, 100) - ref_mouse;
    int upState = v1 > 40 ? 1 : 0;
    int downState = v2 > 40 ? 1 : 0;
    int rightState = v3 > 40 ? 1 : 0;
    int leftState = v4 > 40 ? 1 : 0;
    int clickState = v5 > 40 ? 1 : 0;

    int xDistance = (leftState - rightState) * range;
    int yDistance = (upState - downState) * range;

    if (!(xDistance == 0 && yDistance == 0 && clickState == 0)) {
        moveMouse(xDistance, yDistance, clickState);
    }

    delay(5);
}
```