//

// Show the movie and sends the data to bt2.

//

//

// time delay

//

//

//

import processing.serial.\*;

import processing.video.\*;

final static int SENSITIVITY = 1;

final static int WAIT\_TIME = 10;

Serial myPort;

//Serial bt1, bt2;

String[] path;

Mov[] movArr;

int[] jumpArr;

int jumpCnt; // to jump mov 0

//int timeCnt; // to delay

//boolean isTimeChecked;

int movCnt; // steak or pizza or chicken

int[] movNumArr;

int hashNum;

int nowMov; // movie now Playing.

int prevMov;

boolean isDown;

boolean isUp;

int time1;

boolean isTimeChecked1;

int time2;

boolean isTimeChecked2;

//boolean isFirst;

char ch; // variable to receive data

class Mov

{

Movie myMovie;

String path;

boolean isShown;

boolean isJumped;

Mov()

{

this(null, "", false, false);

}

Mov(Movie m, String p, boolean isShown, boolean isJumped)

{

this.myMovie = m;

this.path = p;

this.isShown = isShown;

this.isJumped = isJumped;

}

void disableMovie(int num)

{

if (this.isShown)

{

println("Movie : " + (num + 1) + " disabled.");

this.isShown = false;

//println(this.myMovie.duration());

if (!this.isJumped)

{

if (num == 0)

{

this.myMovie.jump(0);

jumpCnt++;

if (jumpCnt >= 4)

{

jumpCnt = 0;

}

} else

{

this.myMovie.jump(0);

}

this.myMovie.stop();

this.isJumped = true;

}

}

}

void enableMovie(int num)

{

if (!this.isShown)

{

println("Movie : " + (num + 1) + " enabled.");

this.isShown = true;

this.isJumped = false;

this.myMovie.loop();

}

}

void showImage()

{

image(this.myMovie, 0, 0, displayWidth, displayHeight);

}

};

void setup()

{

size(displayWidth, displayHeight);

btInit();

movNumArrInit();

jumpArrInit();

pathInit();

dataInit();

movInit();

println("setup");

background(0);

}

void draw()

{

ch -= '0';

//println("draw");

//movCnt++;

if (ch >= SENSITIVITY)

{

isTimeChecked2 = false;

//println("ch is up above 5");

if (!isTimeChecked1)

{

isTimeChecked1 = true;

time1 = 0;

time2 = 0;

}

time1++;

println("time1 : " + time1);

if (time1 > WAIT\_TIME)

{

if (!isDown)

{

isDown = true;

isUp = false;

prevMov = nowMov;

movArr[prevMov].disableMovie(prevMov);

nowMov = movNumArr[hashRandFunc(movCnt)];

movCnt++;

if (movCnt > 20000)

{

movCnt = 0;

}

movArr[nowMov].enableMovie(nowMov);

}

}

}

else

{

isTimeChecked1 = false;

if (!isTimeChecked2)

{

isTimeChecked2 = true;

time2 = 0;

time1 = 0;

}

time2++;

//println("time2 : " + time2);

//println("ch is below 5");

if (time2 > WAIT\_TIME)

{

if (!isUp)

{

isDown = false;

isUp = true;

prevMov = nowMov;

movArr[prevMov].disableMovie(prevMov);

nowMov = 0;

movArr[nowMov].enableMovie(nowMov);

}

}

}

//println("movCnt : " + movCnt);

// println("nowMov : " + nowMov);

movArr[nowMov].showImage();

}

void movInit()

{

movArr = new Mov[4];

for (int i = 0; i < movArr.length; i++)

{

movArr[i] = new Mov(new Movie(this, path[i]),

path[i], false, false);

}

}

void dataInit()

{

//timeCnt = 0;

//isTimeChecked = false;

hashNum = 12345;

nowMov = 0;

ch = 0;

time1 = 0;

time2 = 0;

isTimeChecked1 = false;

isTimeChecked2 = true;

}

void pathInit()

{

path = new String[4];

path[0] = "C:\\Users\\user\\Desktop\\ProcessingData\\0.mov";

path[1] = "C:\\Users\\user\\Desktop\\ProcessingData\\1.mov";

path[2] = "C:\\Users\\user\\Desktop\\ProcessingData\\2.mov";

path[3] = "C:\\Users\\user\\Desktop\\ProcessingData\\3.mov";

}

void movNumArrInit()

{

movNumArr = new int[6];

movNumArr[0] = 1;

movNumArr[1] = 2;

movNumArr[2] = 3;

movNumArr[3] = 1;

movNumArr[4] = 2;

movNumArr[5] = 3;

movCnt = 0;

}

void jumpArrInit()

{

jumpArr = new int[6];

jumpArr[0] = 0;

jumpArr[1] = 20;

jumpArr[2] = 40;

jumpArr[3] = 60;

jumpArr[4] = 80;

jumpArr[5] = 100;

jumpCnt = 100;

}

void serialEvent(Serial s)

{

String inString = myPort.readStringUntil('\n');

//println(inString);

if (inString != null)

{

ch = inString.charAt(0);

}

}

void btInit()

{

myPort = new Serial(this, Serial.list()[0], 9600);

myPort.bufferUntil('\n');

}

void movieEvent(Movie m) {

m.read();

}

int hashRandFunc(int count)

{

hashNum += ((count \* 223 + 2) / (double)12 + 97);

if (hashNum > 1000000000) hashNum = 12352;

return int (hashNum % 6);

}