Design Programming

Spring 2018 - 6

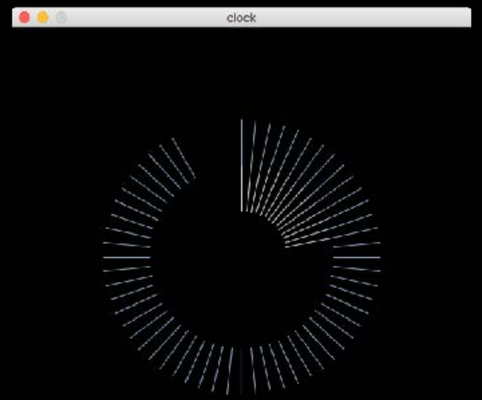
2018.5.23 Keio University, SFC

時間を扱う

関数名	関数の意味
`second();`	現在時刻の秒
`minute();`	現在時刻の分
`hour();`	現在時刻の時間
`millis();`	プログラム起動時からのミリ秒
`frameCount();`	プログラム起動時からのフレーム数

時計をつくる

```
int w, h;
int d = 50,
    dS = d + d,
    dM = dS + d;

void setup() {
    size(500, 500);
    w = width;
    h = height;
}

void draw() {
    background(0);
    for (int i = 0; i < 60; i++) {
        float s = sin(radians(6 * i));
        float c = cos(radians(6 * i));
        if (i <= second()) {
            stroke(246, 239, 247);
            line(d * s + w / 2, - d * c + h / 2, dS * s + w / 2, - dS * c + h / 2);
        }
        if (i <= minute()) {
            stroke(166, 188, 219);
            line(dS * s + w / 2, - dS * c + h / 2, dM * s + w / 2, - dM * c + h / 2);
        }
    }
}
```



06/clock/clock.pde

演習1

分と秒が一致した瞬間から１秒間赤→緑→青に

１フレームずつ画面の色を変化させなさい

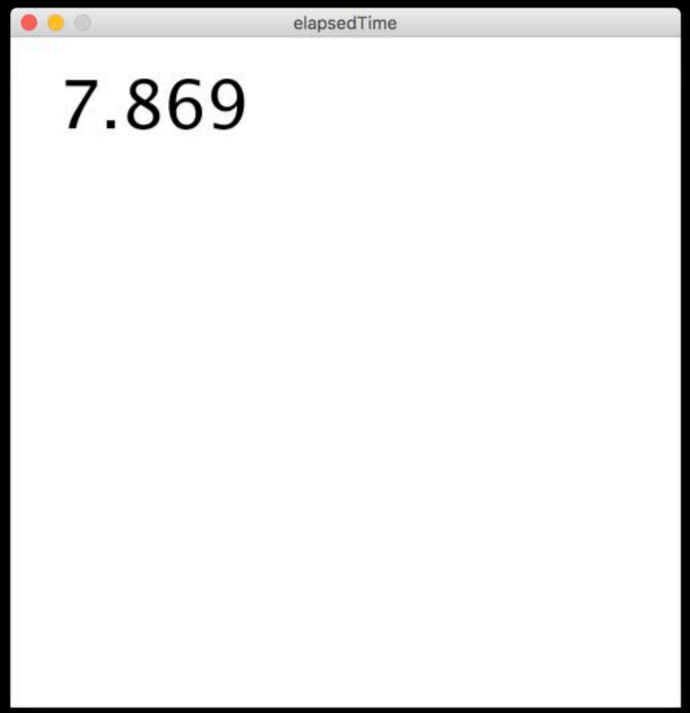
経過時間を扱う

```
float c;

void setup() {
    size(500, 500);
    ellipseMode(LEFT);
}

void draw() {
    c = (float)frameCount;
    background(255);
    fill(0);
    for (int i = 0; i < width / 10; i++) {
        float h = noise((i + c) / 100);
        fill(255 * h, 255 * h, 204 * h);
        ellipse(i * 10, height * h, 10, 10);
    }
}
```

06/frameCount/frameCount.pde

経過時間を扱う

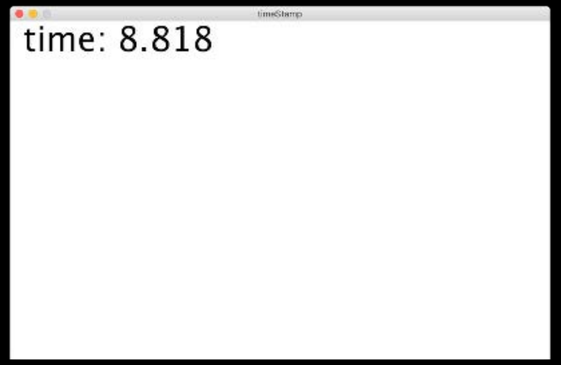
```
float elapsedTime;

void setup() {
    size(500, 500);
    textSize(50);
    textAlign(LEFT, TOP);
}

void draw() {
    elapsedTime = float(millis()) / 1000;
    background(255);
    fill(0);
    text(elapsedTime, 20, 20);
}
```



06/elapsedTime/elapsedTime.pde

時間をリセットする

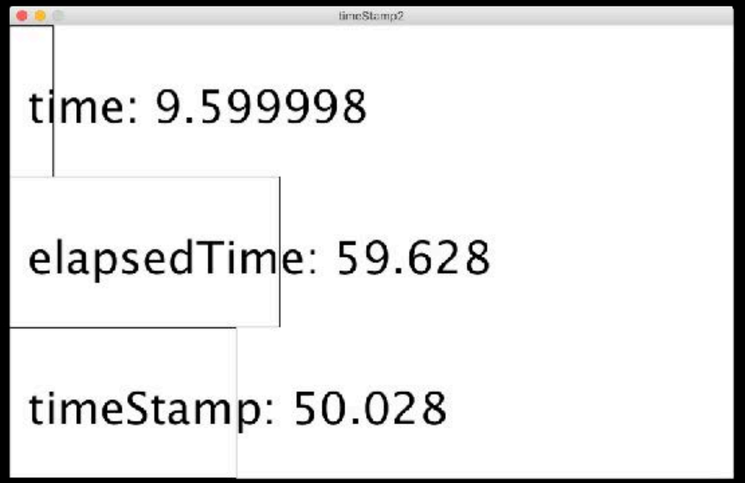
```
float elapsedTime = 0, time = 0, timeStamp = 0;

void setup() {
    size(800, 500);
    textSize(50);
    textAlign(LEFT, CENTER);
}

void draw() {
    elapsedTime = float(millis()) / 1000;
    time = elapsedTime - timeStamp;
    background(255);
    fill(0);
    text("time: " + time, 20, 20);
    if (time >= 10.0) {
        timeStamp = elapsedTime;
    }
}
```



06/timeStamp/timeStamp.pde

時間をリセットする

```
float elapsedTime = 0, time = 0, timeStamp = 0;

void setup() {
    size(800, 500);
    textSize(50);
    textAlign(LEFT, CENTER);
}

void draw() {
    elapsedTime = float(millis()) / 1000;
    time = elapsedTime - timeStamp;
    background(255);
    fill(0);
    text("time: " + time, 20, height / 6);
    text("elapsedTime: " + elapsedTime, 20, height / 2);
    text("timeStamp: " + timeStamp, 20, 5 * height / 6);
    if (time >= 10.0) {
        timeStamp = elapsedTime;
    }
    noFill();
    rect(0, 0, time * 5, height / 3);
    rect(0, height / 3, elapsedTime * 5, height / 3);
    rect(0, height * 2 / 3, timeStamp * 5, height / 3);
}
```



06/timeStamp2/timeStamp2.pde

「タイムプレッシャー」

imaginative inc.

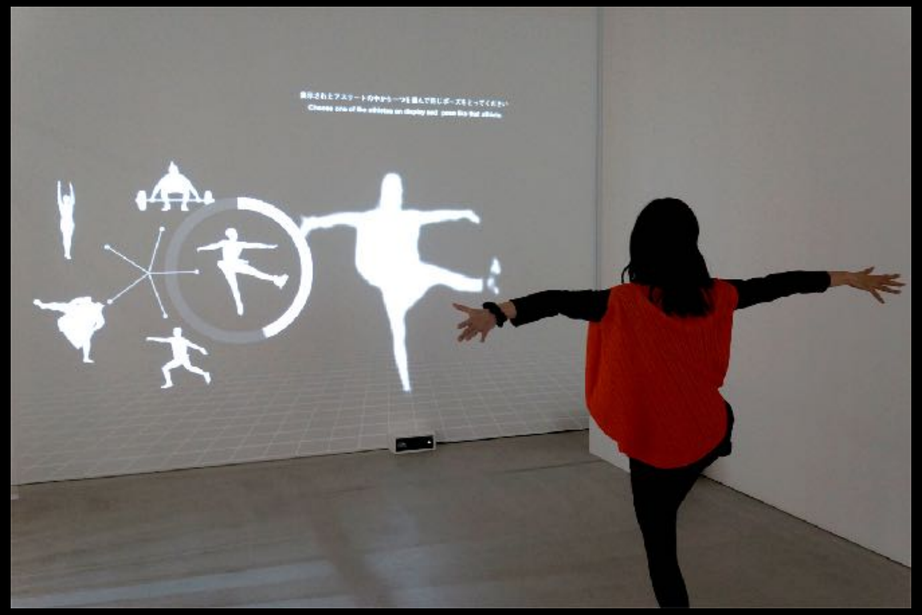
21_21_DESIGN_SIGHTで行われたアスリート展のimaginative inc.の作品．なにげなく体験した１度目のスコアを，２度目は越えられるかというプレッシャー耐性が可視化される．

https://www.japandesign.ne.jp/report/athlete-2121/

（JDNウェブサイト）より.

アスリートの体型特性

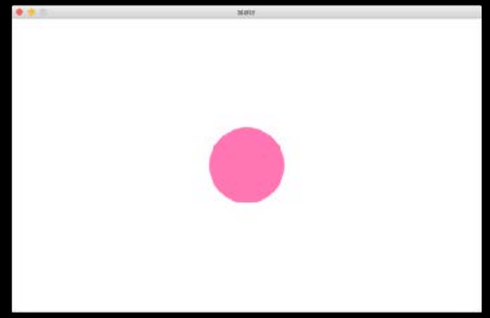
LENS（岡田憲一＋冷水久仁江）



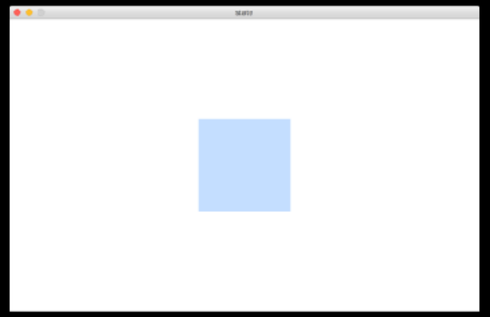
21_21_DESIGN_SIGHTで行われたアスリート展のLENSの作品.
アスリートのとる体型，体勢と体験者のポーズの相違を観ることができる.

https://www.japandesign.ne.jp/report/athlete-2121/
（JDNウェブサイト）より.

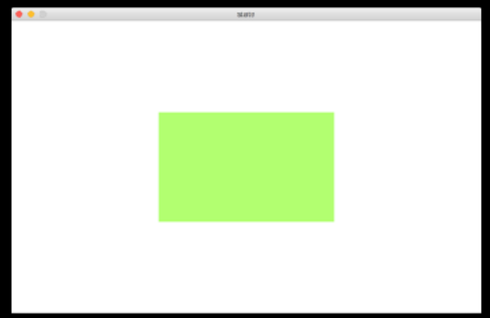
一周する



3秒たったら
時間リセット



2秒たったら
時間リセット

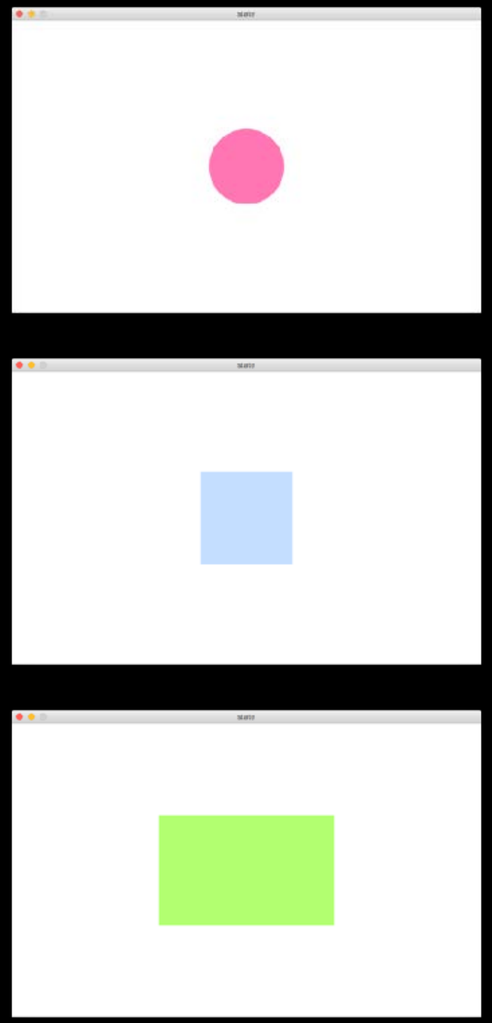


3秒たったら
時間リセット

06/state/state.pde

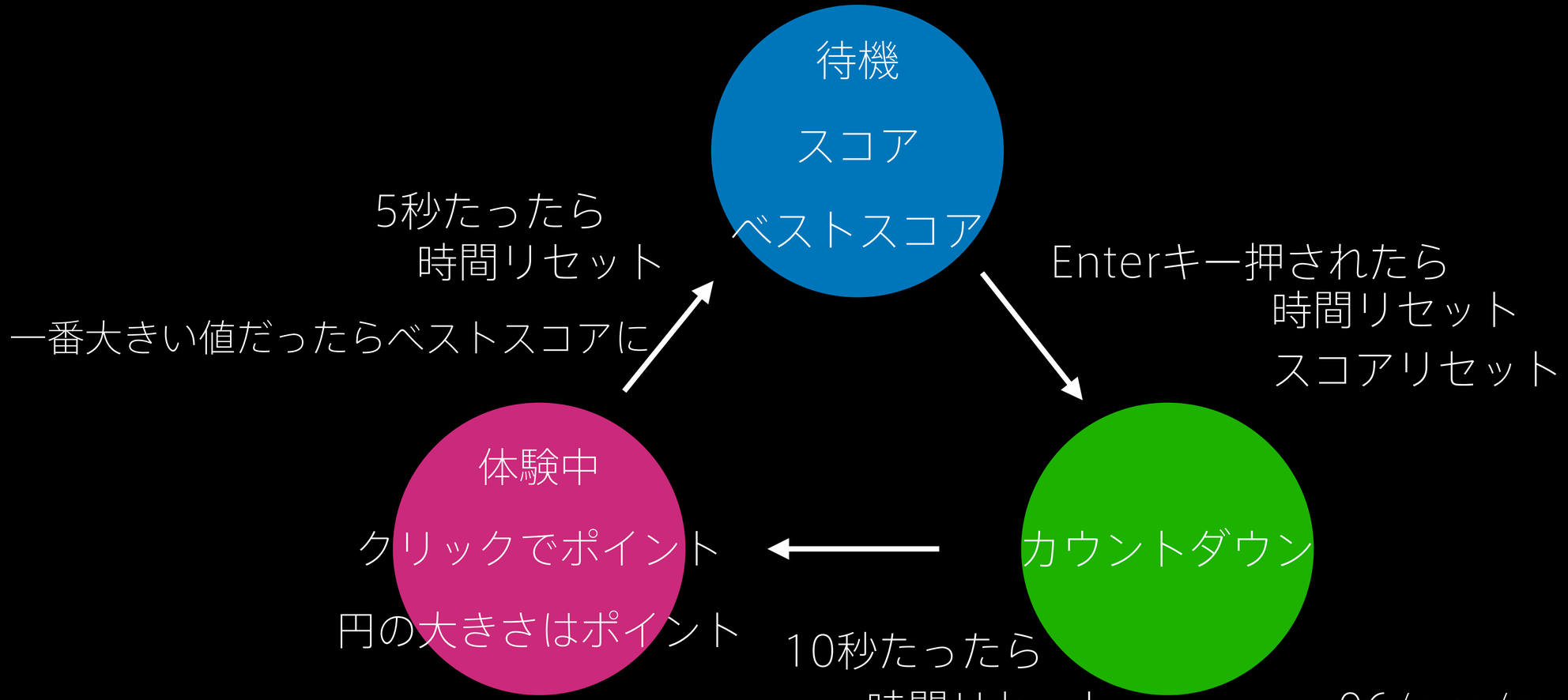
一周する

```
float elapsedTime = 0, time = 0, timeStamp = 0;
int state = 0;

void setup() {
    size(800, 500);
    noStroke();
    rectMode(CENTER);
}

void draw() {
    elapsedTime = float(millis()) / 1000;
    time = elapsedTime - timeStamp;
    background(255);
    switch (state) {
    case 0:
        fill(255, 120, 180, (3 - time) * 255);
        ellipse(width / 2, height / 2, 100 * time, 100 * time);
        if (time >= 3.0) {
            state = 1;
            timeStamp = elapsedTime;
        }
        break;
    case 1:
        fill( 120, 180, 255, (2 - time) * 255);
        rect(width / 2, height / 2, 100 * time, 100 * time);
        if (time >= 2.0) {
            state = 2;
            timeStamp = elapsedTime;
        }
        break;
    case 2:
        fill(180, 255, 120, (3 - time) * 255);
        rect(width / 2, height / 2, time * width / 3, time * height / 3);
        if (time >= 3.0) {
            state = 0;
            timeStamp = elapsedTime;
        }
        break;
    }
}
```



06/state/state.pde

体験型アプリをつくる



06/exp/exp.pde

課題

- 今日学習した「体験型アプリをつくる」を使用して何度も繰り返し体験できるスケッチを１つアップロードしなさい
  アップロード先は「05_time」とする.
  ※今日学習していない分野でもすでに自分が知っている技術は使用可能とする.

  〆切：5月29日24時まで

次回

- さまざまな運動