1 Animações

Animação é a exibição rápida de uma sequência de imagens para criar uma ilusão de movimento. O método mais comum de apresentar animação é como um filme ou um programa de vídeo, embora existam outros métodos. Este tipo de apresentação é realizado geralmente em uma visualização no computador, câmera fotográfica ou um projetor. Geralmente 24, 25 ou 30 quadros por segundo já é o bastante para causar o efeito de animação.

2 PImage

Uma coisa que eu, particularmente, gosto muito em processing é a possibilidade de se carregar imagens nos sketchs e trabalhar elas muitas maneiras. Uma já vimos no documento passado utilizando tint para colorir imagens.

Atributos/Campos:

- pixels[] Arranjo de pixels contendo cada um a cor correspondente na imagem.
- width
- height

Métodos/Ações:

- loadPixels() Carrega o campo pixels[]
- updatePixels() Recarrega o campo pixels[]
- resize() Redimensiona para novos width e height
- get() Le 1 pixel, ou pode também pegar um retângulo de pixels
  - img.get(x, y, w, h)

- set() Configura 1 pixel, ou uma imagem para outra
  - img.set(x, y, c)
  - img.set(x, y, img)

2.1 Exercício 1

Vamos começar entendendo um pouco de física e animação ao mesmo tempo. Quando se vai animar algo. É bom que se pense em tentar ser fiel a física espacial da coisa. Não precisa se prender ao máximo a realidade. Mas por exemplo, criar uma bolinha que bate e volta. Como podemos representar isso de uma maneira que convença?

```
//code
```

2.1.1 Exercício 2

```
PImage imagem;

void setup() {
  size(200, 200);
  a = loadImage("tarrafa.png"); // Carrega a imagem
  noLoop();
}

void draw() {
  image(imagem, 0, 0);
  image(a, 100, 0, imagem.width/2, imagem.height/2);
  //image é objeto de PImage ou seja ele tem seus atributos e métodos
  //cada objeto imagem de Pimage tem width e height próprios
  //e eles são acessáveis diretamente
}
```

2.2 Exercício 3

Getter and Setter

Método Get() PImage

```
PImage img = loadImage("tarrafa.png");
background(img);
noStroke();
color c = img.get(60, 90);
fill(c);
rect(25, 25, 50, 50);
```

```
PImage img = loadImage("tarrafa.png");
background(img);
PImage img2 = img.get(50, 0, 50, 100);
image(img2, 0, 0);
```

Método Set() PImage

```
PImage img;

void setup() {
  img = loadImage("tarrafa.png");
  color black = color(0);
  img.set(30, 20, black);
  img.set(85, 20, black);
  img.set(85, 75, black);
  img.set(30, 75, black);
}

void draw() {
  image(img, 0, 0);
}
```

2.3 Exercício 4

Save é outro método de PImage onde você pode salvar imagens em alguns formatos. Esse exercício server para vocês aplicarem o código a seguir como forma de salvar a tela inteira do sketch rodando. É bastante útil para salvar o trabalho de vocês em imagens.

- TIFF

- PNG
- JPEG

```
void keyPressed() {
  this.set(0, 0, this.get(0, 0, width, height));
  this.save("pics/frame-" + this.nf(frameCount, 4) + ".png");
}
```

2.4 Exercício 5

A animação a seguir foi retirada dos exemplos do Processing (Exemplos, Drawing, Animator).

```
int currentFrame = 0;
PImage[] frames = new PImage[24];
int lastTime = 0;

void setup()
{
  size(640, 200);
  strokeWeight(12);
  smooth();
  background(204);
  for (int i = 0; i < frames.length; i++) {
    frames[i] = get(); // Create a blank frame
  }
}

void draw()
{
  //Essa lógica do tempo a seguir é uma tecnica
  //Utilizando millis() que é um tempo que começa do 0
  //Então se verifica intervalos passados desse tempo
  //Que não para até o sketch do processing ser finalizado
  //É conhecido no Arduino como Blinking Without Delay.

  int currentTime = millis();
```

```
  if (currentTime > lastTime+30) {
    nextFrame();
    lastTime = currentTime;
  }
  if (mousePressed) {
    line(pmouseX, pmouseY, mouseX, mouseY);
  }
}

void nextFrame()
{
  //Nesse sketch esse método é um pouco mais complicado
  //Antes os frames foram iniciados em branco
  //E o sistema está esperando pelo mousepressed
  //Toda vez que o mouse for pressionado
  //Ele adiciona a imagem do frame no array de PImages
  //E fica repetindo-as até que se sobreponha com nova images
  //Ou seja se frameAtual >= quantidadeDeTelas reseta o sisteminha
  //E começa a sobrepor imagens.
  frames[currentFrame] = get(); // Get the display window
  currentFrame++; // Increment to next frame
  if (currentFrame >= frames.length) {
    currentFrame = 0;
  }
  image(frames[currentFrame], 0, 0);
}
```