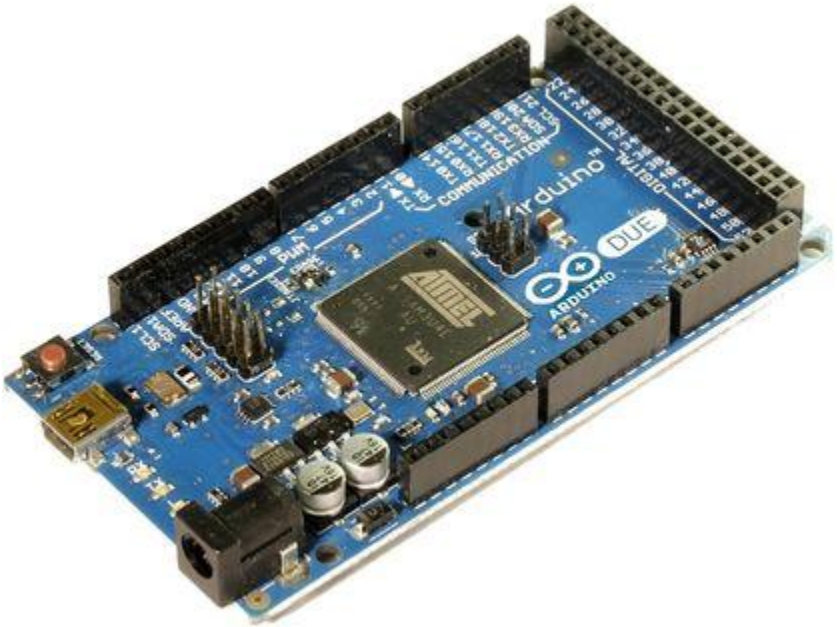
# Arduino e Processing




Professor: Paulo Marcos Trentin

# O que é Processing?

É uma linguagem criada em 2001 por Casey Reas e Benjamin Fry, na época estudantes do MIT.

Seu objetivo é server como ferramenta de desenvolvimento de projetos gráficos e interativos para artistas e designers com pouco conhecimento de pgoramação.

É fácil de aprender e usar. Com ela podemos, por exemplo, exibir graficamente no computdor o valor de um sensor anlógico conectado ao Arduino

Fonte: Experimentos com o Arduino - livro

# Baixando o Processing

Acesse:

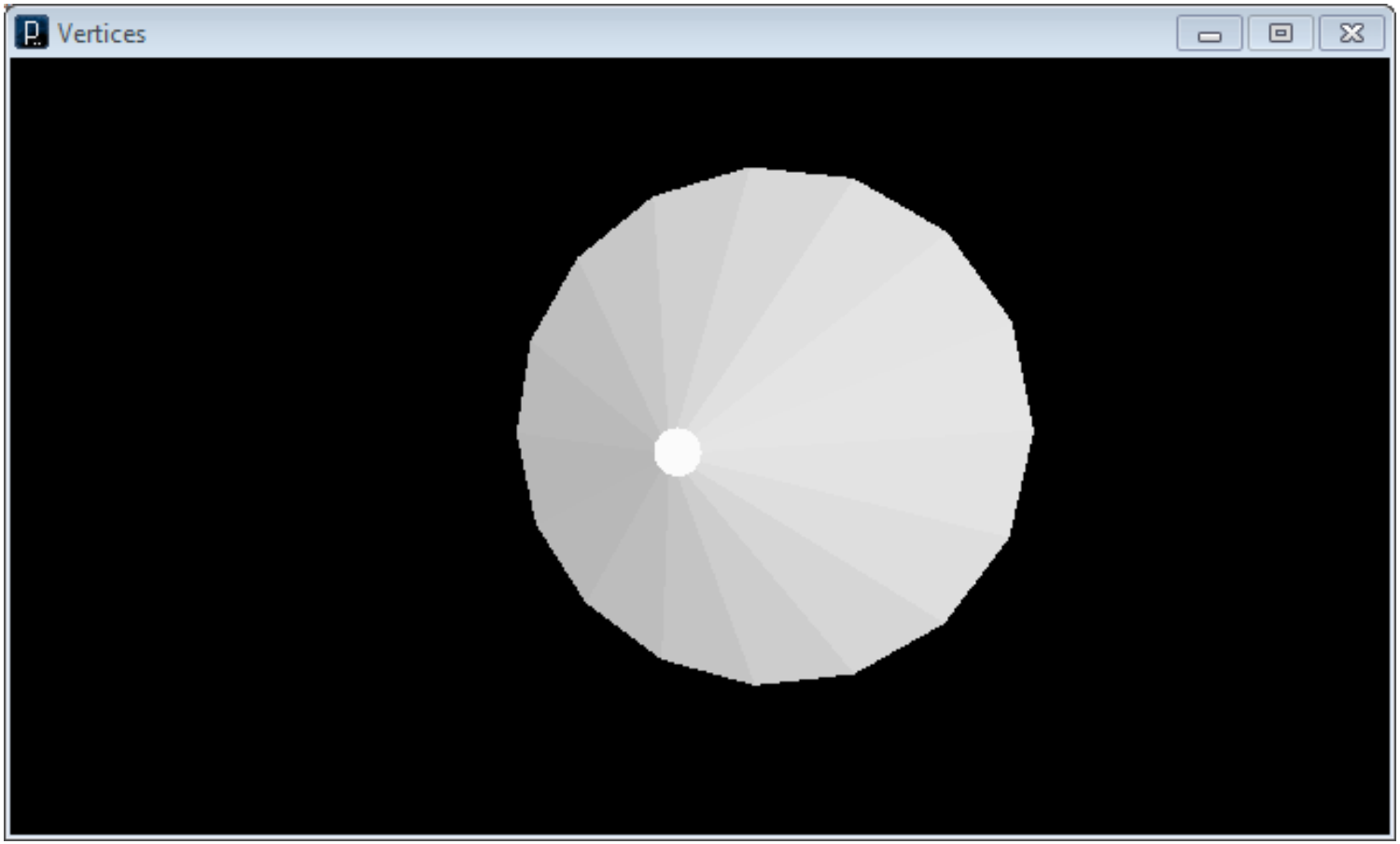
http://www.processing.org/download/

Baixe a última versão clicando no link de seu sistema operacional

# Hello world com Processing

Inicialmente abra a IDE.

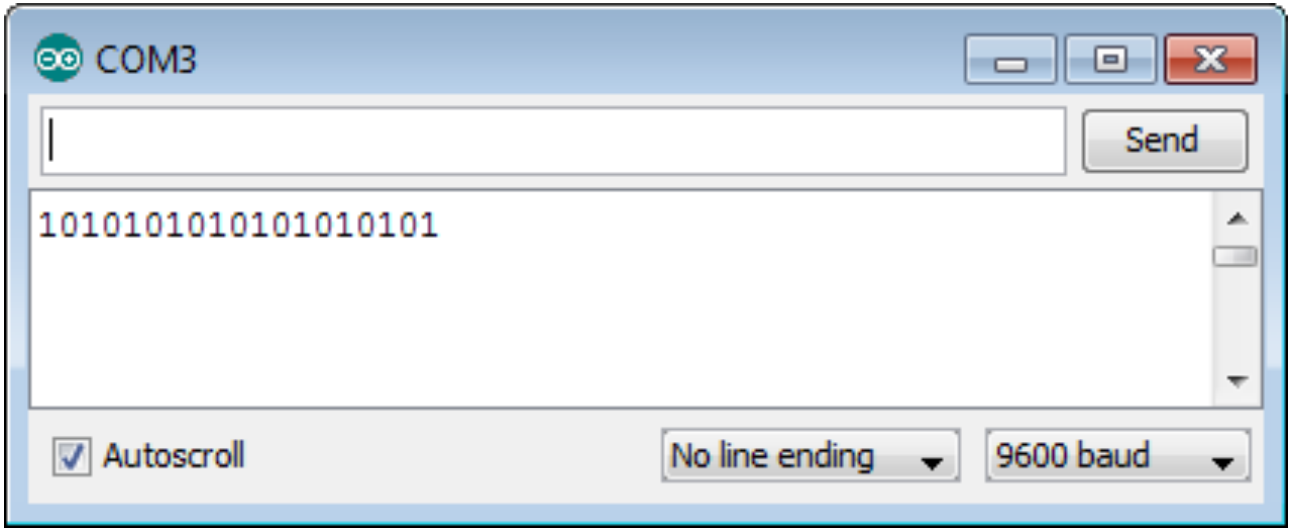
Abra o exemplo: 3D > Form > Vertices



# Unindo Arduino e Processing

O primeiro passo é fazer o Arduino enviar dados via serial.

Abra o exemplo blink e modifique-o fazendo com que envie o número 1 quando o LED ligar, e o número 0 quando o LED desligar.



# Arduino + Processing Tutorial
# código fonte do Arduino

```
int led = 13;

void setup() {
        pinMode(led, OUTPUT);
        Serial.begin(9600);
}

void loop() {
        Serial.print(1);
        digitalWrite(led, HIGH);
        delay(1000);
        Serial.print(0);
        digitalWrite(led, LOW);
        delay(1000);
}
```

# Recebendo os dados via Processing - p1

```
//  Este código fonte irá na IDE do Processing!
//  Importa bibliotecas para Serial
import processing.serial.*;
Serial port; // Inicia instância de porta serial

void setup(){
  // Define tamanho da janela
  size(200,200);
  // Inicia porta serial na COM3 a 9600 bauds
  port = new Serial(this,  "COM3", 9600);
  // Atenção: veja se seu Arduino está na COM3!
}
```

# Recebendo os dados via Processing - p2

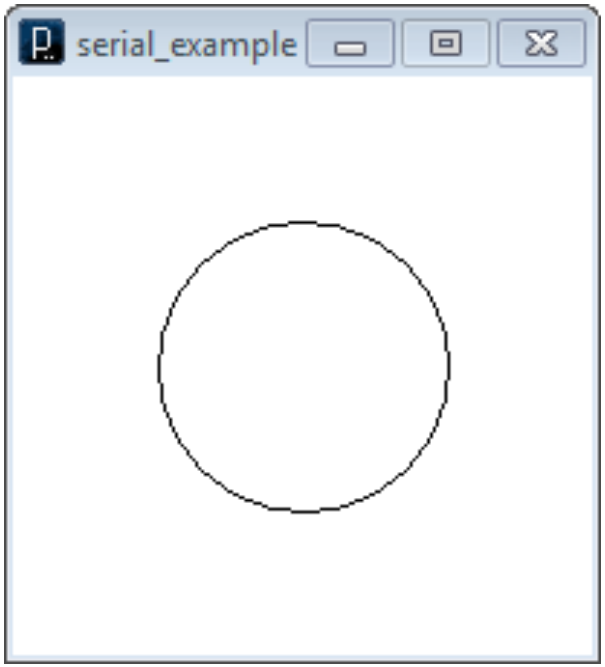
```
void draw(){
  // Enquanto receber algo pela serial
  while (port.available() > 0){
      background(255); // Define o fundo branco
      // Se recebeu 1 da Serial
      if (port.read() == '1'){
         fill(#00ff00); // Define cor da elipse (verde)
         // Cria um círculo (posX, posY, largura, altura)
         ellipse(100,100,100,100);
       }
    ...
```

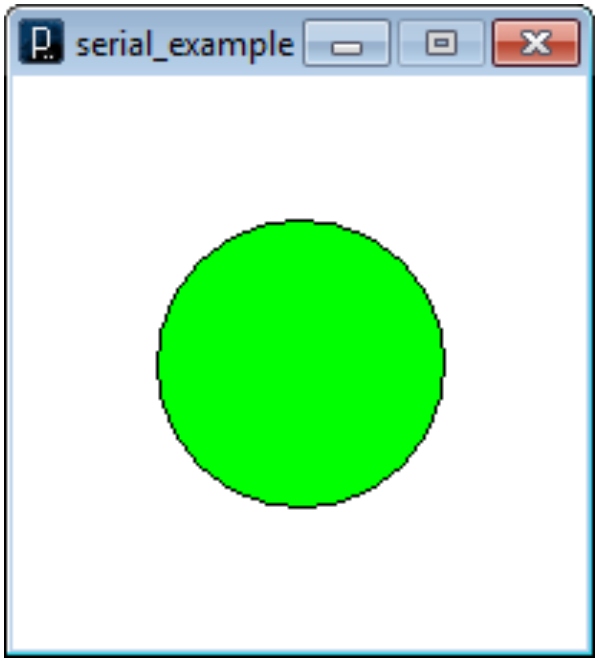
# Recebendo os dados via Processing - p3

...

```
    // Se recebeu qualquer coisa diferente de 1
    else{
        // Insere fundo branco na ellipse
        fill(255);
         // Redesenha a ellipse
        ellipse(100,100,100,100);
     }
  }
}
```

# Recebendo os dados via Processing Resultado




Repare no sincronismo entre o LED do Arduino e nosso LED virtual

# Processing e Serial

Podemos receber e enviar dados para a Serial através do Processing.

Isso significa que podemos fazer um programa de computador acionar um motor em nosso Arduino!

Mais sobre Processing e Serial:
http://processing.org/reference/libraries/serial/index.html

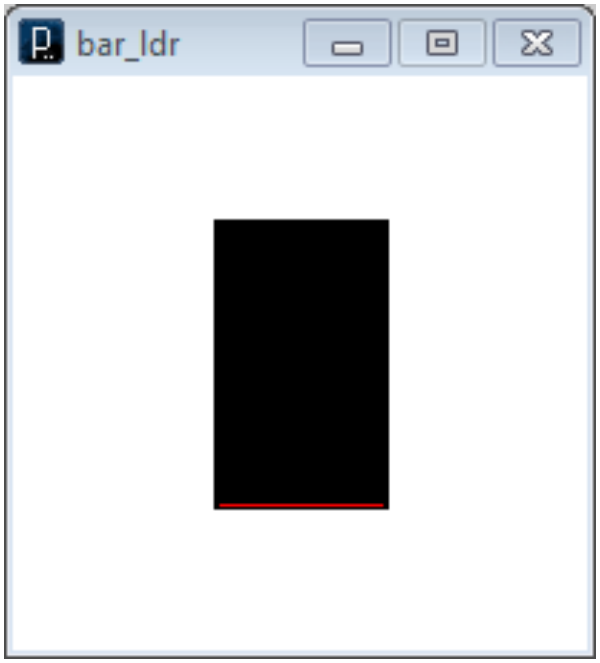
# O mesmo exemplo, porém com uma barra

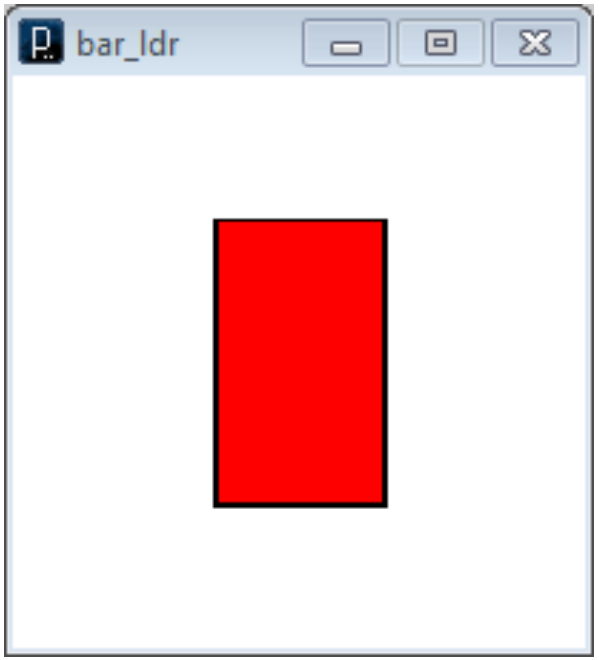
```
void draw(){
  // Enquanto receber algo pela serial
  while (port.available() > 0){
      background(255); // Define o fundo branco
      // Se recebeu 1 da Serial
      if (port.read() == '1'){
        // Preenche com preto
        fill(#000000);
        // Cria um retângulo de 60 x 100. coordenada x = 70 e y = 50
        rect(70, 50, 60, 100);
        // Preenche com vermelho
        fill(#ff0000);
        // Cria retângulo de 58 x 99 (para não cobrir o retângulo preto)
        // Coordenada x = 71 e y = 50
        rect(71, 50, 60 - 2, 99);
      } ...
```

# O mesmo exemplo, porém com uma barra

```
...
else{
        // Preenche com preto
        fill(#000000);
        // Cria um retângulo de 60 x 100. coordenada x = 70 e y = 50
        rect(70, 50, 60, 100);
        // Preenche com vermelho
        fill(#ff0000);
        // Cria um retângulo de 58 x 2. coordenada x = 71 e y = 148
        // 98 vem da diferença de 100 - 2, "jogando" o objeto 98 px
         // a mais para baixo
        rect(71, 50 + 98, 60 - 2, 2);
      }
  }
}
```

# Recebendo os dados via Processing
# Resultado com a Barra

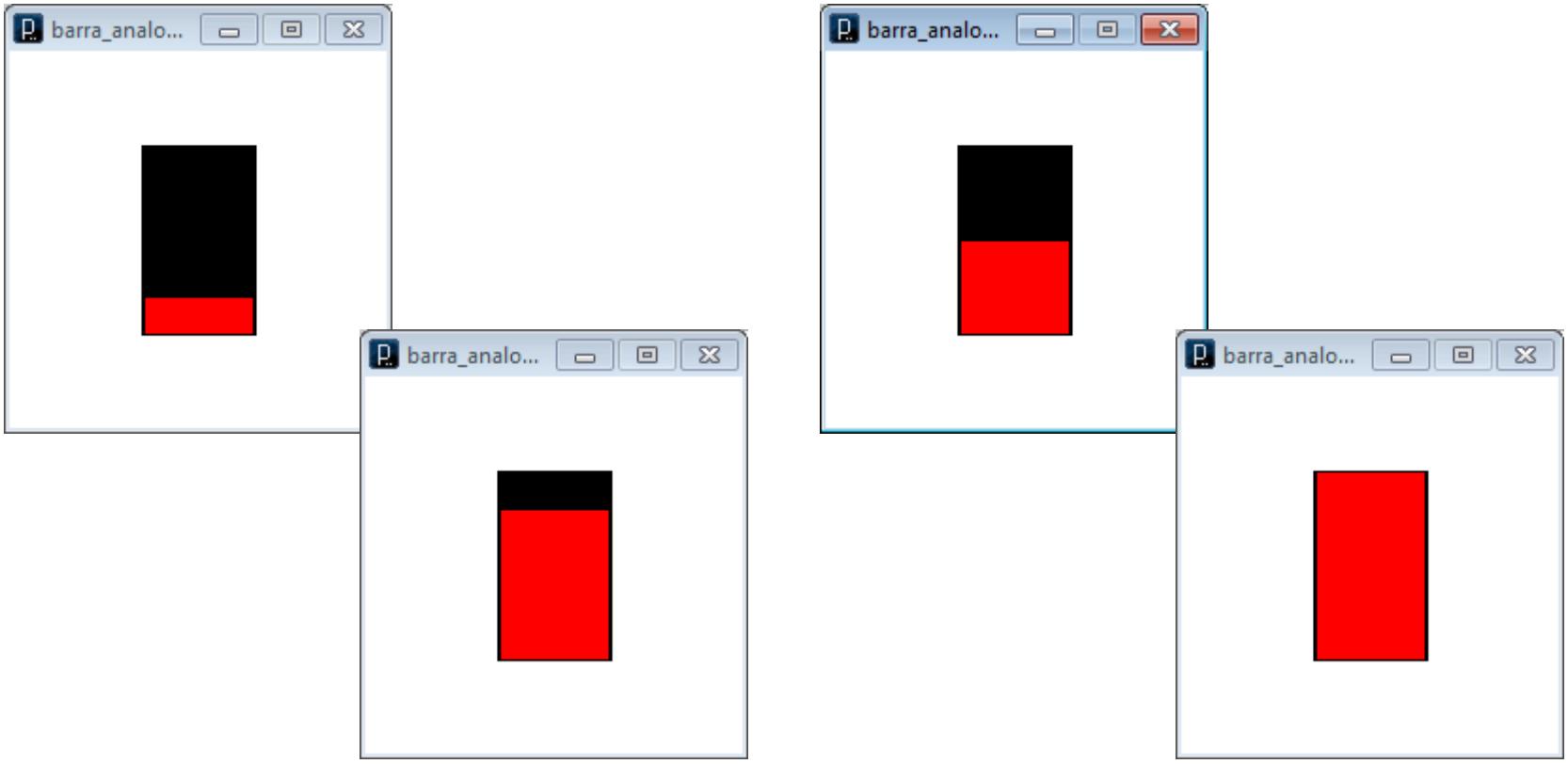



Repare que na barra a esquerda, só aparece 1px da barra vermelha. Enquanto que na direita, aparece 98px

# Barra analógica

Que tal vermos graficamente o estado de um sensor conectado ao Arduino?



# Barra analógica com Código Processing - p1

```
//  Importa bibliotecas para Serial
import processing.serial.*;
Serial port; // Inicia instância de porta serial
int valorRecebido; // Armazena o valor recebido via serial

void setup(){
  // Define tamanho da janela
  size(200,200);
  // Inicia porta serial na COM3 a 9600 bauds
  port = new Serial(this,  "COM3", 9600);
}
```

# Barra analógica com Código Processing - p2

```
void draw(){
  // Enquanto receber algo pela serial
  while (port.available() > 0){
    // Converte o char para int (-48)
    valorRecebido = port.read() - 48;
    redrawBarra();

    // Exibe no console o valor que recebeu pela serial
    print(valorRecebido);
    print("  ");
  }
}
```

# Barra analógica com Código Processing - p3

```
/**
 *  Redesenha o gráfico da barra
 */
void redrawBarra(){
        background(255); // Define o fundo branco
        // Preenche com preto
        fill(#000000);
        // Cria um retângulo de 60 x 100. coordenada x = 70 e y = 50
        rect(70, 50, 60, 100);
        // Preenche com vermelho
        fill(#ff0000);

        ...
```

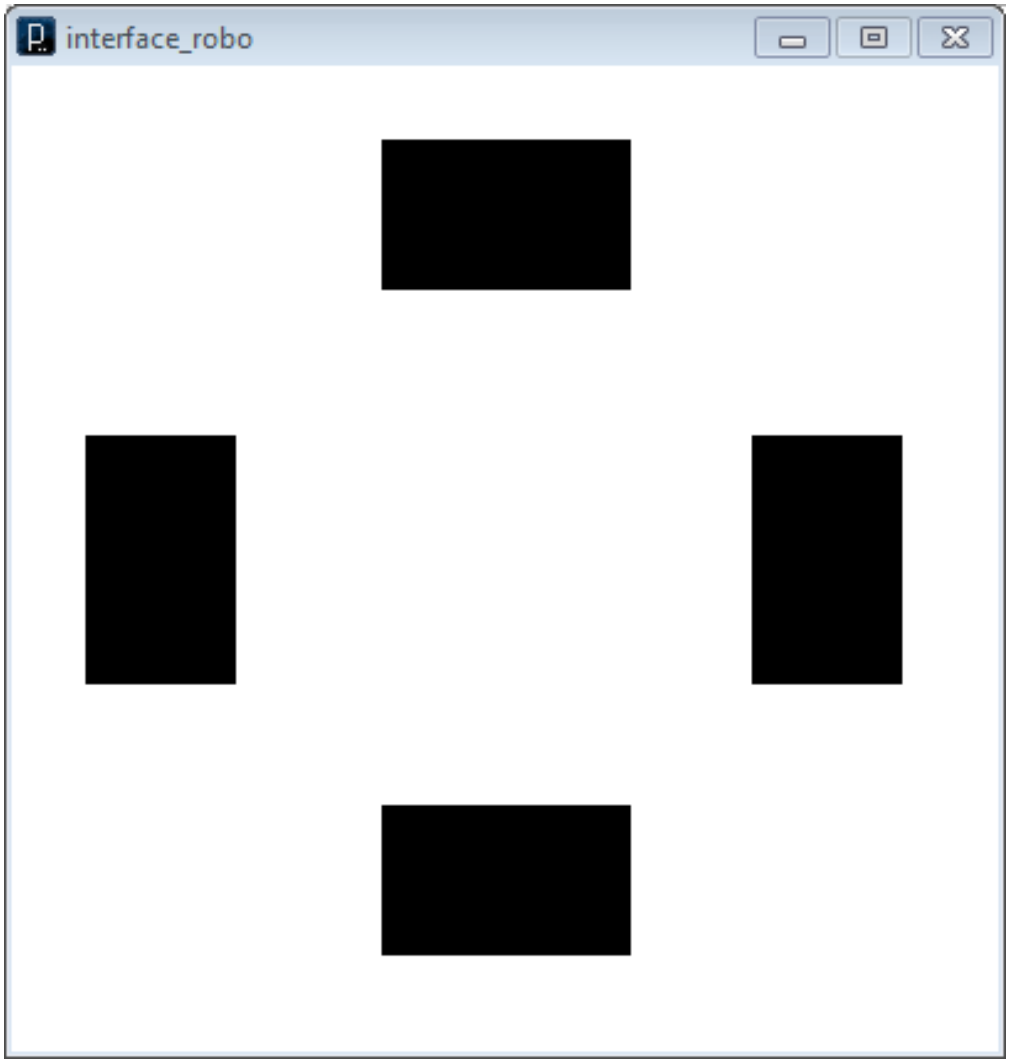
# Barra analógica com Código Processing - p4

```
    ...

    //  Recebe pela serial valores de 0 à 9, para gerar
    // um gráfico melhor, adiciona um, tendo valores de 1 à 10
    valorRecebido += 1;

    // Prepara o valor para inserir no gráfico
    int valorConvertido = valorRecebido * 10;

    // Não altera o posicionamento da altura da barra (faça testes!)
    //rect(71, 0, 60 - 2, valorConvertido);
    // Gera Altera posicionamento da altura da barra de acordo com o
  valor recebido
    rect(71, 50 + (100 - valorConvertido), 60 - 2, valorConvertido);
}
```

# Barra analógica com Código Arduino - função loop

```
/**
 *  Baseado no código do LED anterior, alteramos
 *  somente a função loop
 */
void loop() {
        // Faz a leitura analógica (conecte um potenciômetro aqui)
        int valorLido = analogRead(A0);
        // Converte valor de 0-1023 para 0-9
        int valorConvertido = map(valorLido, 0, 1023, 0, 9);
        // Envia valor para serial
        Serial.print(valorConvertido);
        delay(50);
}
```

# E se criássemos uma interface para controlar nosso robô pelo PC?



# Criando a interface de controle para o robô pelo PC - p1

```
//  Importa bibliotecas para Serial
import processing.serial.*;
Serial port; // Inicia instância de porta serial

void setup(){
        // Define tamanho da janela
        size(400,400);
        // Inicia porta serial na COM3 a 9600 bauds
        port = new Serial(this,  "COM3", 9600);
}
```

# Criando a interface de controle para o robô pelo PC - p2

```
void draw(){
        // Enquanto receber algo pela serial
        while (port.available() > 0){
          // Exibe a resposta do Arduino
          int byteRecebido = port.read();
          // Converte valor inteiro para char (typecasting)
          char byteLegivel = (char) byteRecebido;
          print(byteLegivel);
        }

        // Desenha botões na tela
        desenhaBotao();
}
```

# Criando a interface de controle para o robô pelo PC - p3

```
// Função que desenha botões na tela
void desenhaBotao(){
        background(255); // Define o fundo branco
        // Preenche com preto
        fill(#000000);
        // Cria botão para ir para frente
        rect(150, 30, 100, 60);
        // Cria botão para ir para trás
        rect(150, 300, 100, 60);
        // Cria botão para ir para esquerda
        rect(30, 150, 60, 100);
        // Cria botão para ir para direita
        rect(300, 150, 60, 100);
}
```

# Criando a interface de controle para o robô pelo PC - p4

```
// Verifica se clicou no botão de ir para frente
boolean botaoFrente(){
        if (mouseX > 150 && mouseX < 150 + 100
                && mouseY > 30 && mouseY < 30 + 60) {

                return true;
        }
        else{
                return false;
        }
}
```

# Criando a interface de controle para o robô pelo PC - p5

```
// Verifica se clicou no botão de ir para trás
boolean botaoTras(){
        if (mouseX > 150 && mouseX < 150 + 100
                && mouseY > 300 && mouseY < 300 + 60) {

                return true;
        }
        else{
                return false;
        }
}
```

# Criando a interface de controle para o robô pelo PC - p6

```
// Verifica se clicou no botão de ir para direita
boolean botaoDireita(){
        if (mouseX > 300 && mouseX < 300 + 60
                && mouseY > 150 && mouseY < 150 + 100) {

                return true;
        }
        else{
                return false;
        }
}
```

# Criando a interface de controle para o robô pelo PC - p7

```
// Verifica se clicou no botão de ir para esquerda
boolean botaoEsquerda(){
        if (mouseX > 30 && mouseX < 30 + 60
                && mouseY > 150 && mouseY < 150 + 100) {

                return true;
        }
        else{
                return false;
        }
}
```

# Criando a interface de controle para o robô pelo PC - p8

```
// Quando o mouse é clicado, o Processig chama esta função
void mousePressed() {
        println("  ");
        println("Coordenada x: " + mouseX + " e y: " + mouseY);

        if (botaoFrente()){
                println("Clicou no botao de ir para frente");
                // Envia código 1 para Arduino
                port.write(1);
                println("Enviado codigo 1");
        }

...
```

# Criando a interface de controle para o robô pelo PC - p9

```
// ... continuação da função mousePressed
        if (botaoTras()){
                println("Clicou no botao de ir para tras");
                // Envia código 2 para Arduino
                port.write(2);
                println("Enviado codigo 2");
        }

        if (botaoEsquerda()){
                println("Clicou no botao de ir para esquerda");
                println("Enviado codigo 3");
                // Envia código 3 para Arduino
                port.write(3);
        }  ...
```

# Criando a interface de controle para o robô pelo PC - p10

```
// ... continuação da função mousePressed, parte final
        if (botaoDireita()){
                println("Clicou no botao de ir para direita");
                // Envia código 4 para Arduino
                port.write(4);
                println("Enviado codigo 4");
        }
}
```

# Código fonte do Arduino para controle do robô - p1

```
void setup() {
        pinMode(13, OUTPUT);
        Serial.begin(9600);
}

/**
 *  Função loop aguarda códigos vindos via Serial
 */
void loop() {
        if (Serial.available() > 0){
                char valorLido = Serial.read();
                Serial.print("Arduino diz: ");
                 ...
```

# Código fonte do Arduino para controle do robô - p2

```
...

if (valorLido == 1){
        digitalWrite(13,HIGH);
        Serial.println("Estou indo para frente");
}

if (valorLido == 2){
        digitalWrite(13,LOW);
        Serial.println("Estou indo para tras");
}
```

# Código fonte do Arduino para controle do robô - p3

```
        if (valorLido == 3){
          Serial.println("Estou indo para esquerda");
        }

        if (valorLido == 4){
          Serial.println("Estou indo para direita");
        }
    } // Fim do primeiro if dentro do Loop


    delay(100);
} // Fim da função loop
```

# Mais estudos

Tutoriais Processing:

http://processing.org/learning

Avançando um pouco...

Biblioteca para gerar uma interface bonita visualmente:

http://www.lagers.org.uk/g4ptool/index.html