unipampa

Universidade Federal do Pampa

Iniciação à Plataforma Arduino: Teoria e Prática

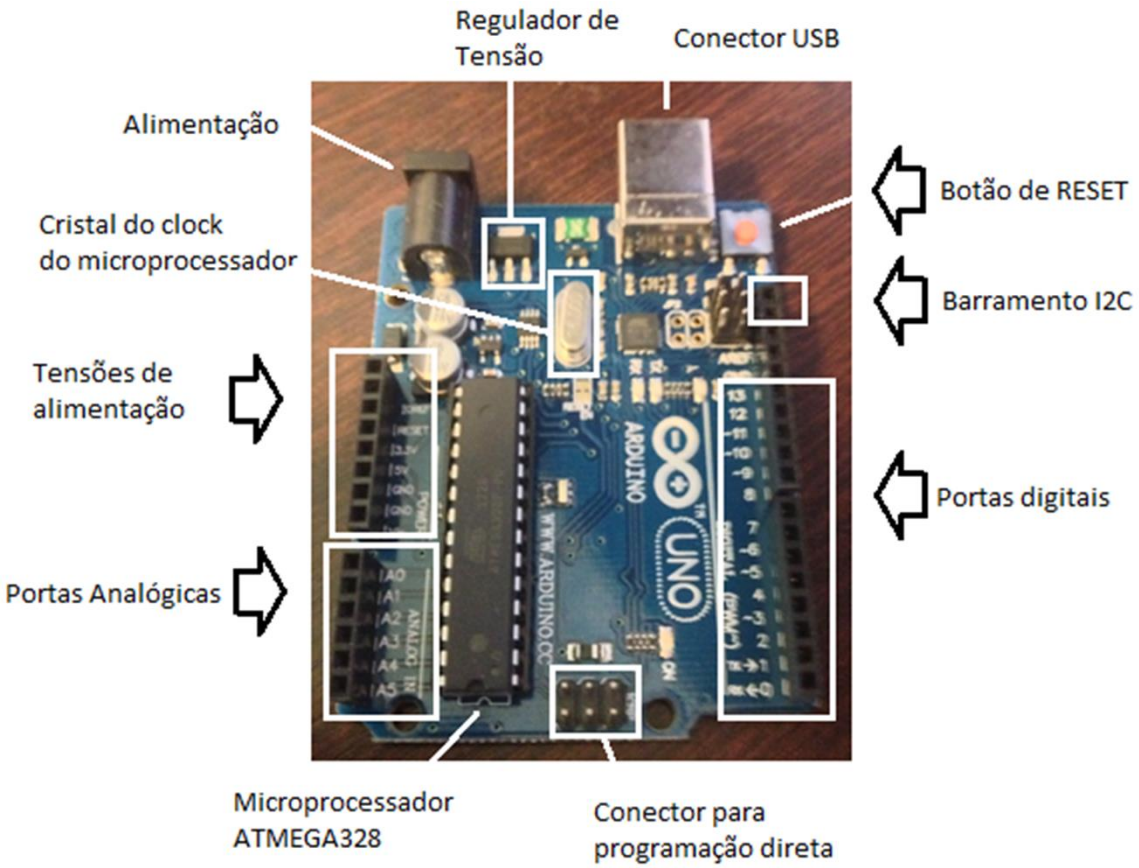
www.arduino.cc/

Januário Ribeiro
Pedro Dorneles

Junho de 2018

ARDUINO

Componentes



Fonte: http://www.arduinodolito.com.br/

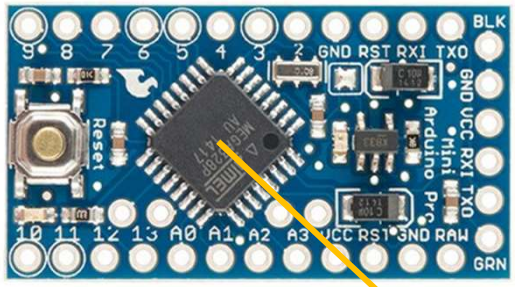
Características Básicas

Arduino Uno

Microcontrolador	ATmega328 / ATmega168
Pinos de entrada analógica	6
Pinos de I/O digitais	14 (dos quais 6 podem ser saídas PWM)
Tensão operacional	$5\ V$
Tensão de alimentação (recomendada)	$7 - 12\ V$
Tensão de alimentação (limites)	$6 - 20\ V$
Corrente contínua por pino de I/O	$40\ mA$
Corrente contínua para o pino 3.3 V	$50\ mA$

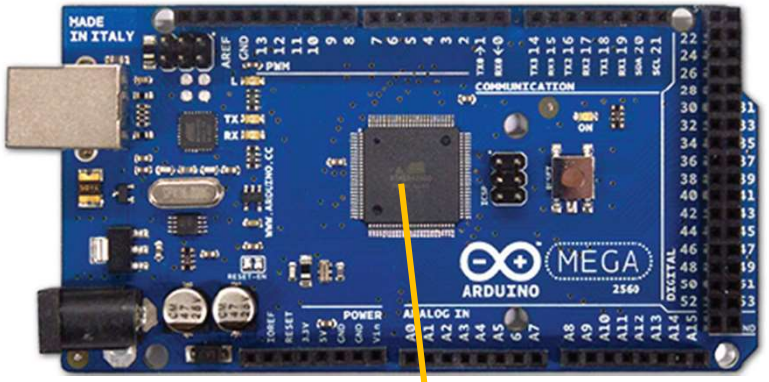
Outros Modelos

Arduino Promini



Microcontroller	ATmega328
Operating Voltage	3.3V or 5V (depending on model)
Input Voltage	3.35 -12 V (3.3V model) or 5 - 12 V (5V model)
Digital I/O Pins	14 (of which 6 provide PWM output)
Analog Input Pins	8
DC Current per I/O Pin	40 mA
Flash Memory	32 kB (of which 0.5 kB used by bootloader)
SRAM	2 kB
EEPROM	1 kB
Clock Speed	8 MHz (3.3V model) or 16 MHz (5V model)

Arduino Mega



Microcontroller	ATmega2560
Operating Voltage	5V
Input Voltage (recommended)	7-12V
Input Voltage (limit)	6-20V
Digital I/O Pins	54 (of which 15 provide PWM output)
Analog Input Pins	16
DC Current per I/O Pin	20 mA
DC Current for 3.3V Pin	50 mA
Flash Memory	256 KB of which 8 KB used by bootloader
SRAM	8 KB
EEPROM	4 KB
Clock Speed	16 MHz
Length	101.52 mm
Width	53.3 mm
Weight	37 g

Pinos Digitais

Função Entrada	Função Saída
I – INPUT	O– OUTPUT
pinoMode (led, INPUT)	pinoMode (led, OUTPUT)
0 – 1,0 V – Baixo (LOW)	0,0 V – Baixo (LOW)
3,0 – 5,0 V – Alto (HIGH)	5,0 V – Alto (HIGH)

Pinos Analógicos (Somente Entrada)

Função Entrada I – INPUT

pinoMode (led, INPUT)

Para realização de medidas um conversor analógico digital A/D gera uma representação digital (valores discretos) de uma grandeza analógica (valores contínuos)

Tensões são convertidas em uma série de números binários (sinais digitais)

O conversor A/D do Arduino:

- É de 10 bits
- Recebe sinal de entrada analógica de tensão variável de 0,0 V a 5,0 V
- Pode assumir os valores binários de 0 (0000000000) a 1023 (1111111111) – 2^{10} = 1024 combinações
- É capaz de capturar 1024 níveis discretos de um determinado sinal
- É sensível a tensões de aproximadamente 5,0 mV (5,0 V/1023 = 4,89 mV) para tensão de referência igual a 5,0 V.
- É sensível a tensões de aproximadamente 1,1 mV (1.1 V/1023) para tensão de referência igual a 1,1 V.

Representação Decimal/Binária

0	0	0	0	0	1	0	1	0	1
2^9	2^8	2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
512	256	128	64	32	16	8	4	2	1
					16		4		1

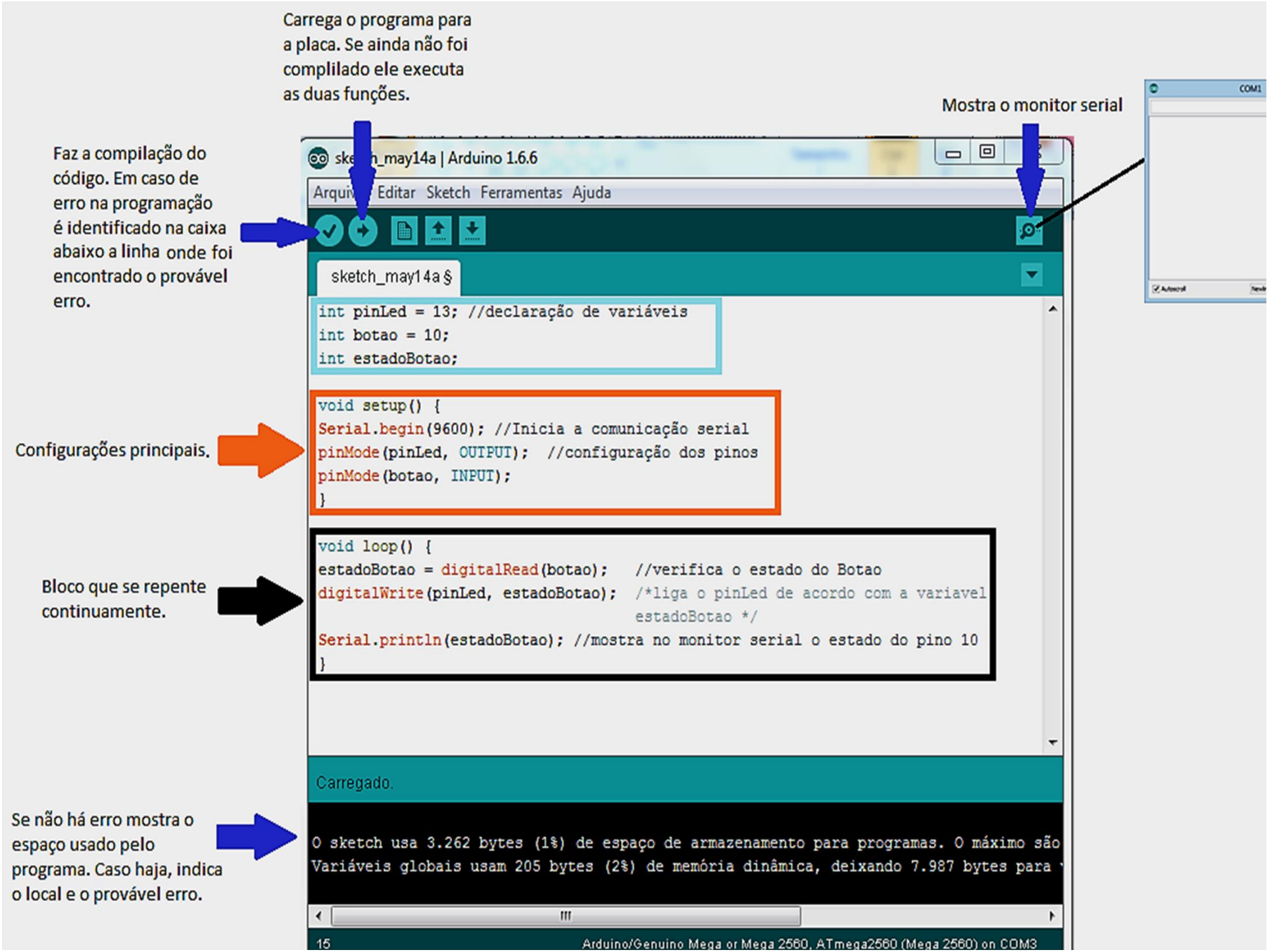
$$1023 = 2^9 + 2^8 + 2^7 + 2^6 + 2^5 + 2^4 + 2^3 + 2^2 + 2^1 + 2^0$$

21 – Representação decimal da sequência binária 0000010101.

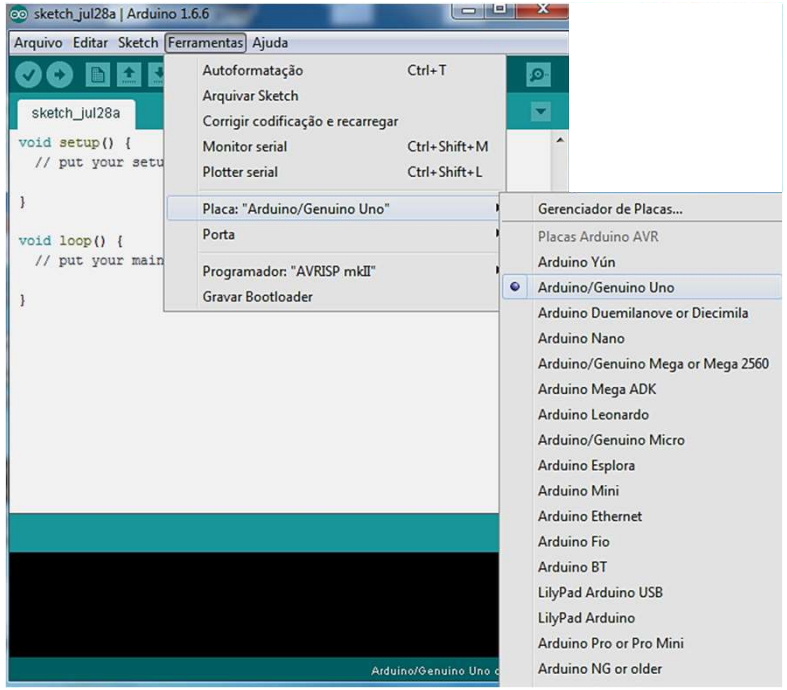
Diagrama de Blocos

LDR

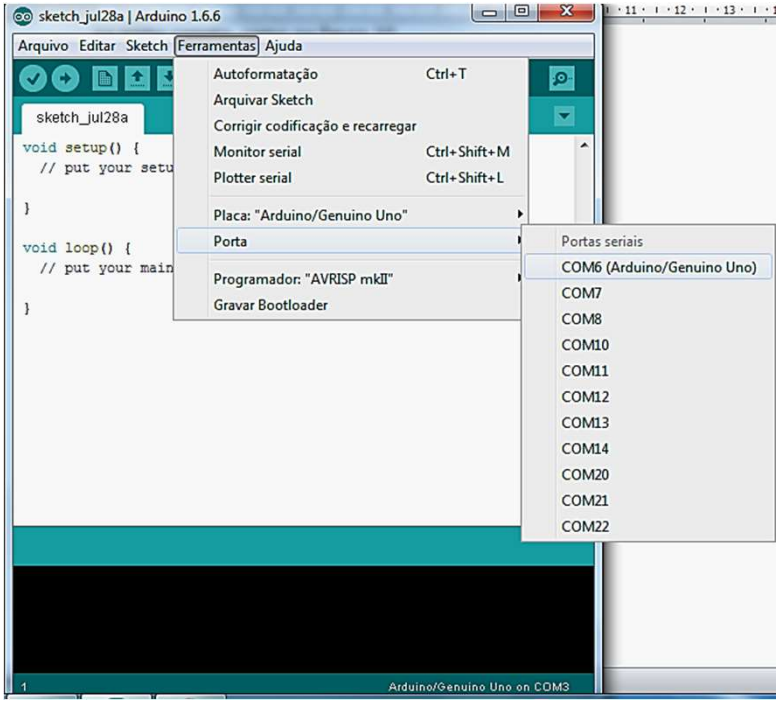
```
void loop() {
ValorLido = analogRead(LDR);
Serial.println(ValorLido);
if (ValorLido < 50)
{
digitalWrite(Led, HIGH);
}
else{
```


Carrega o programa para a placa. Se ainda não foi complilado ele executa as duas funções.
Mostra o monitor serial
Faz a compilação do código. Em caso de erro na programação é identificado na caixa abaixo a linha onde foi encontrado o provável erro.
sketch_may14a | Arduino 1.6.6
Arquivo Editar Sketch Ferramentas Ajuda
sketch_may14a §
int pinLed = 13; //declaração de variáveis
int botao = 10;
int estadoBotao;
Configurações principais.
void setup() {
Serial.begin(9600); //Inicia a comunicação serial
pinMode(pinLed, OUTPUT); //configuração dos pinos
pinMode(botao, INPUT);
}
Bloco que se repente continuamente.
void loop() {
estadoBotao = digitalRead(botao); //verifica o estado do Botao
digitalWrite(pinLed, estadoBotao); /*liga o pinLed de acordo com a variavel estadoBotao */
Serial.println(estadoBotao); //mostra no monitor serial o estado do pino 10
}
Carregado.
Se não há erro mostra o espaço usado pelo programa. Caso haja, indica o local e o provável erro.
O sketch usa 3.262 bytes (1%) de espaço de armazenamento para programas. O máximo são
Variáveis globais usam 205 bytes (2%) de memória dinâmica, deixando 7.987 bytes para
15
Arduino/Genuino Mega or Mega 2560, ATmega2560 (Mega 2560) on COM3
COM1

Configurando o Arduino

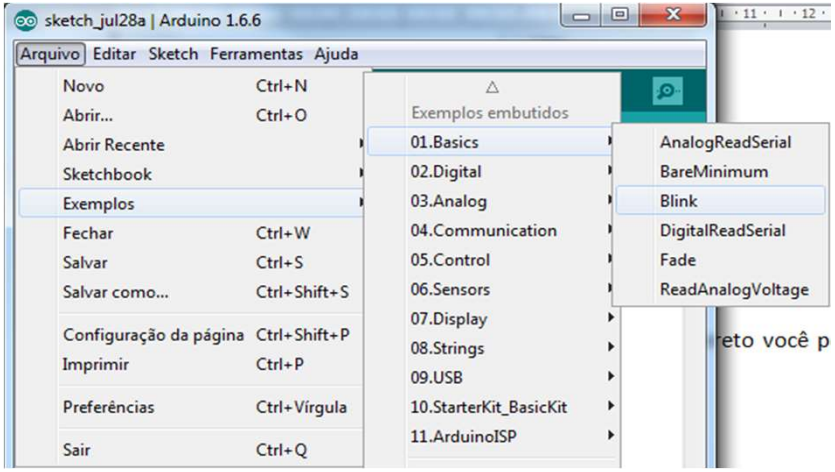


1



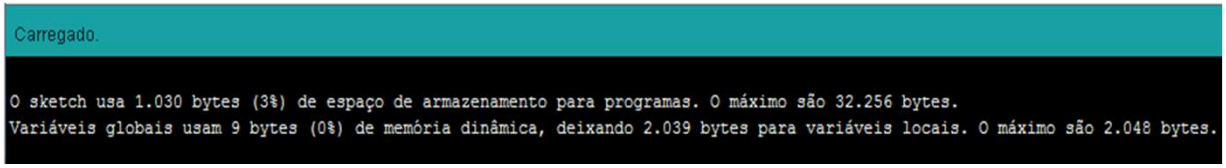
2

Configurando o Arduino



3

4



5

Programando o Arduino

Tipos de Dados

Os tipos de dados são referentes à variável que está sendo utilizada.

int (para números inteiros de dois bytes, abrange entre -32.768 e 32.768).

unsigned int (números inteiro sem sinal, abrange de 0 a 65535).

long (números inteiros sinalizados com 32 bits, abrange de -2.147.483.648 a 2.147.483.647).

unsigned long (números inteiros sem sinal com 32 bits, abrage 0 a 4.294.967.295).

float (para valores com ponto flutuante, utilizam 4 bytes ou 32 bits, abrange 3,4 E-38 a 3,4E+38).

Funções

Entradas e saídas digitais

pinMode(pino, modo): utilizado para dizer que o pino esta como INPUT(entrada) ou OUTPUT (saída).

Exemplo: pinMode(13, OUTPUT) - o pino 13 esta como saída (alto ou baixo)

Programando o Arduino

digitalWrite(pino, valor): escreve em um pino especificado o valor HIGH (alto) ou LOW (baixo). Esse valor é uma tensão de 5V (HIGH) ou 0V (LOW).

Exemplo: digitalWrite(13, HIGH) - o pino 13 tem uma tensão de 5V.

digitalRead(pino): Lê o valor do pino digital e diz seu estado HIGH ou LOW.

Exemplo: valor = digitalRead(7) - a variável “valor” agora vale o estado do pino 7

Entradas e saídas analógicas

analogRead(pino): Lê o valor do pino analógico. O valor lido é um número inteiro de 0 a 1023. Essa função mede o valor da tensão no pino, mas não mostra diretamente o valor em *volts* e sim um número inteiro de 0 a 1023.

Deve-se calcular o valor da tensão através de uma conta simples.

$$V_{pin} = \frac{\mathbf{5}\ \boldsymbol{x}\ \boldsymbol{valor\ lido}}{\mathbf{1023}}$$

Estruturas de Controle e Comparadores

if, if … else

Uma estrutura muito utilizada quando programamos o Arduino é o **if** (se). Ela é usada junto com operadores de comparação. Verifica se a sentença é verdadeira ou falsa. Se for verdadeira executa o comando que estão entre as chaves, se for falsa os ignora. Se utilizarmos junto com o **else** (senão), no caso da sentença for falsa são executado os comandos entre as chaves do else.

Operadores de comparação

igual a	==
diferente de	!=
menor que	<
maior que	>
menor e igual a	<=
maior e igual a	>=

Exemplo:

```
ValorLido = analogRead(VR);

if (ValorLido < 500)
{
digitalWrite(Led, HIGH);
}
else{
digitalWrite(Led, LOW);
}
```

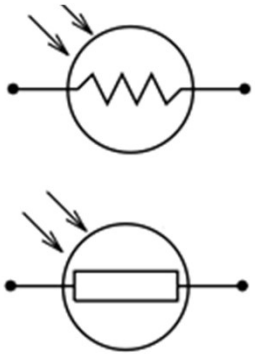
while(condição)

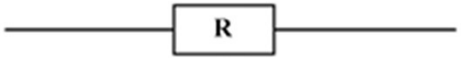
Faz um bloco de comandos que esta dentro das chaves do **while** ser executado continuamente até que a condição dentro dos parênteses não seja mais verdadeira.

Exemplo:

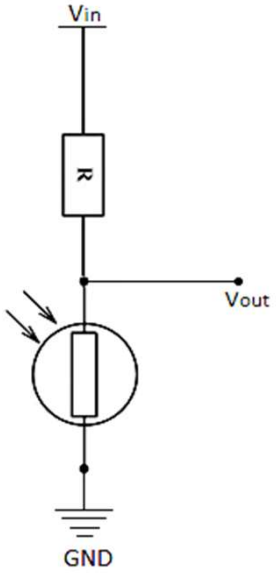
```
b = 10;
a = 0;
while (a<b) {
a = a+1;
}
```

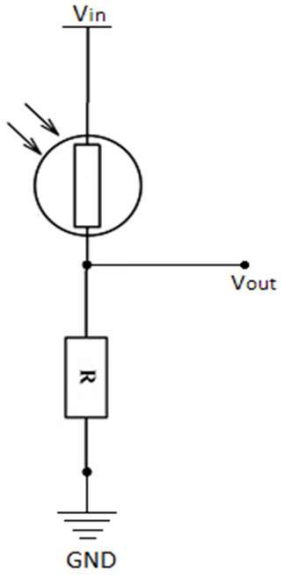
LDR e Resistor





Divisor de Tensão





$$V_{out} = \frac{R_2}{(R_1 + R_2)} V_{in}$$

Código LDR

```
int ValorLido = 0;
float VLDR;

void setup() {
Serial.begin (9600);
pinMode(13, OUTPUT);
pinMode(12, OUTPUT);
}
void loop() {
ValorLido = analogRead(0);
VLDR=5.00*ValorLido/1023;
Serial.print(VLDR);
Serial.println("     ");

if (VLDR > 3.00)
{
digitalWrite(13, HIGH);
digitalWrite(12, LOW);

}
else{
digitalWrite(13, LOW);
digitalWrite(12, HIGH);
}
delay(1000);
}
```

Referências

ARDUINO. Disponível em: http://www.arduino.cc/. Acesso em 10 de outubro de 2017.

CAVALCANTE, M. A., TAVOLARO, C. R. C & ELIO MOLISANI, E. Física com Arduino para iniciantes. Revista Brasileira de Ensino de Física, v. 33, n. 4. 2010.

ROCHA, F. S. & GUADAGNINI, P. H. Projeto de um sensor de pressão manométrica para ensino de física em tempo real. Trabalho submetido para publicação na Revista Brasileira de Ensino de Física.

WRASSE, A., SANTOS, R., TONEL, A. P., KAKUNO, E. M. & DORNELES, P. Carrinho automatizado como recurso facilitador na construção e interpretação de gráficos da cinemática. In: XX SIMPÓSIO NACIONAL DE ENSINO DE FÍSICA – SNEF 2013 – São Paulo, SP.