APRENDA ARDUINO E ELETRÔNICA

ATHOS ELECTRONICS .info

Sobre o Livro e o Autor

Meu nome é Athos, e sou criador do site Athos Electronics!

Nesse E-book, pretendo reunir vários conceitos práticos sobre Eletrônica Básica e Arduino.

O foco deste E-book é a prática mesmo, então não entrarei em detalhes nos conceitos teóricos da eletrônica. Focaremos no aprendizado prático.

Nos próximos dias, também vou enviar direto no seu e-mail alguns conteúdos sobre a área, espero que goste!

Estamos abertos sempre a críticas construtivas, comentários e sugestões, então caso queira, vou deixar nosso e-mail e site para contato:

athos@athoselectronics.info

http://www.athoselectronics.info

Boa leitura e ótimos estudos!

Eletrônica Básica

A eletrônica é a ciência que estuda a forma de controlar a energia elétrica em circuitos.

Para entender o básico da eletrônica, vamos começar com três princípios: Tensão, Corrente e Resistencia.

Tensão:

É a força que faz os elétrons se moverem. Também é conhecida como Potencial Elétrico.

A tensão é medida em Volts, e por isso os leigos gostam de chamar a tensão de “voltagem”.

Um exemplo de tensão no nosso dia-a-dia é quando falamos que uma tomada é 110 ou 220, quer dizer que ela tem 110 volts ou 220 volts.

Corrente:

É o fluxo de elétrons em um condutor, quando temos uma certa tensão ou diferença de potencial.

A corrente é medida em amperes, e por isso seu termo leigo é a “amperagem”.

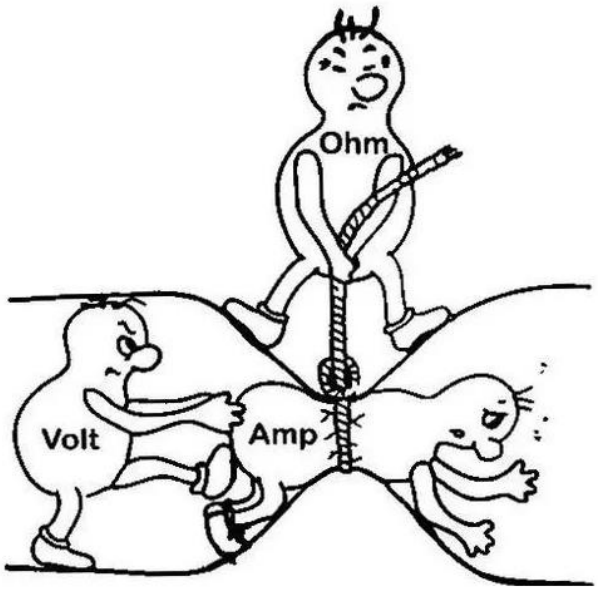
Resistencia:

É a oposição a passagem de corrente em um circuito.

Quanto maior a resistência, menos corrente, e quanto menor a resistência, mais corrente.

A resistência é medida em Ohms.

Para ilustrar esses três conceitos, encontrei na internet uma imagem bem cômica:



Lei de Ohm

Tensão, Corrente e Resistencia serão conceitos relacionados pela Lei de Ohm, que afirma:

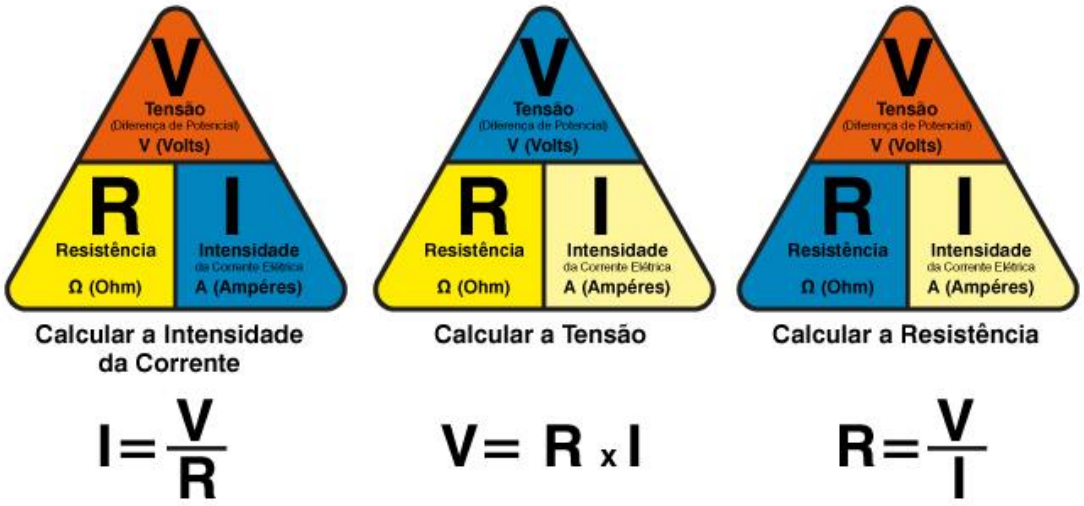
"Para um condutor mantido à temperatura constante, a razão entre a tensão entre dois pontos e a corrente elétrica é constante. Essa constante é denominada de resistência elétrica."

E disso, temos a fórmula:

$$V = R.I$$

Onde:

- V = Tensão (Volts)
- R = Resistencia (Ohms)
- I = Corrente (Amperes)



Essa lei será muito usada quando você precisar calcular um resistor.

Vamos supor que você tenha um LED que suporta no máximo 20 miliamperes, que vai ser alimentado por uma fonte de 9 volts. Você vai precisar limitar a corrente com um resistor, certo? E como saber qual o valor da resistência adequada? Lei de Ohm.

Basta colocar os valores na fórmula, ou seja, 9=0,02R, que vai resultar em 450 ohms! Esse é o resistor que você deverá usar em seu circuito.

Potência Elétrica

Essa é a Potência dissipada por um determinado circuito, medida em Watts.

O valor de um Watt equivale a multiplicação da tensão pela corrente no circuito, ou seja:

$$P = V.\mathrm{I}$$

Onde:

- P = Potência Elétrica (Watts)
- V = Tensão (Volts)
- I = Corrente (Amperes)

No nosso dia-a-dia, quando vamos comprar um eletrodoméstico vem a informação de quantos Watts ele consome, assim como a conta de luz também é cobrada pelos Watts.

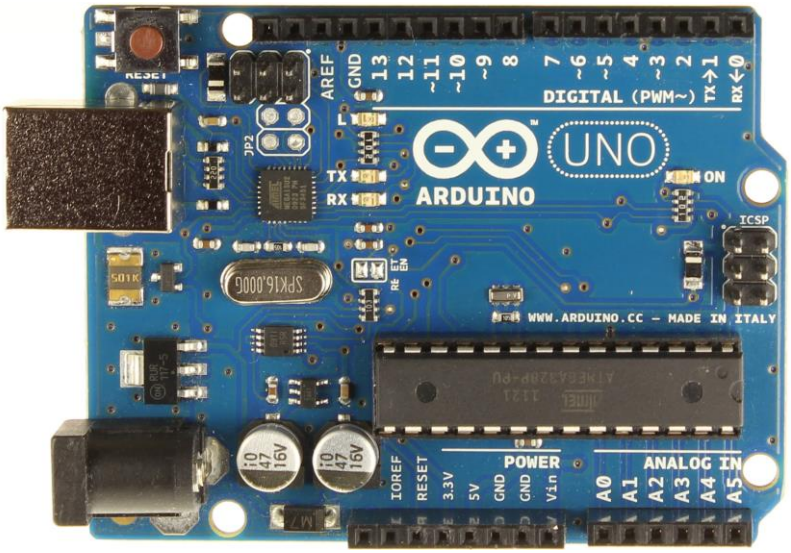
Componentes Eletrônicos

Os componentes Eletrônicos são as partes que constituem os circuitos, que vão servir para manipular a energia elétrica. Alguns exemplos de componentes são: Resistores, Capacitores, Indutores, Transistores, Diodos, Relés, entre outros.

Para esse livro não ficar muito extenso e cansativo de ler, não vou entrar em detalhes dos componentes eletrônicos, mas, vou deixar o endereço de uma categoria do meu site, onde você vai encontrar artigos sobre cada componente eletrônico.

http://athoselectronics.info/category/componentes-eletronicos/

Introdução ao Arduino



O que é o Arduino?

O Arduino faz parte do conceito de hardware e software livre e está aberto para uso e contribuição de todos.

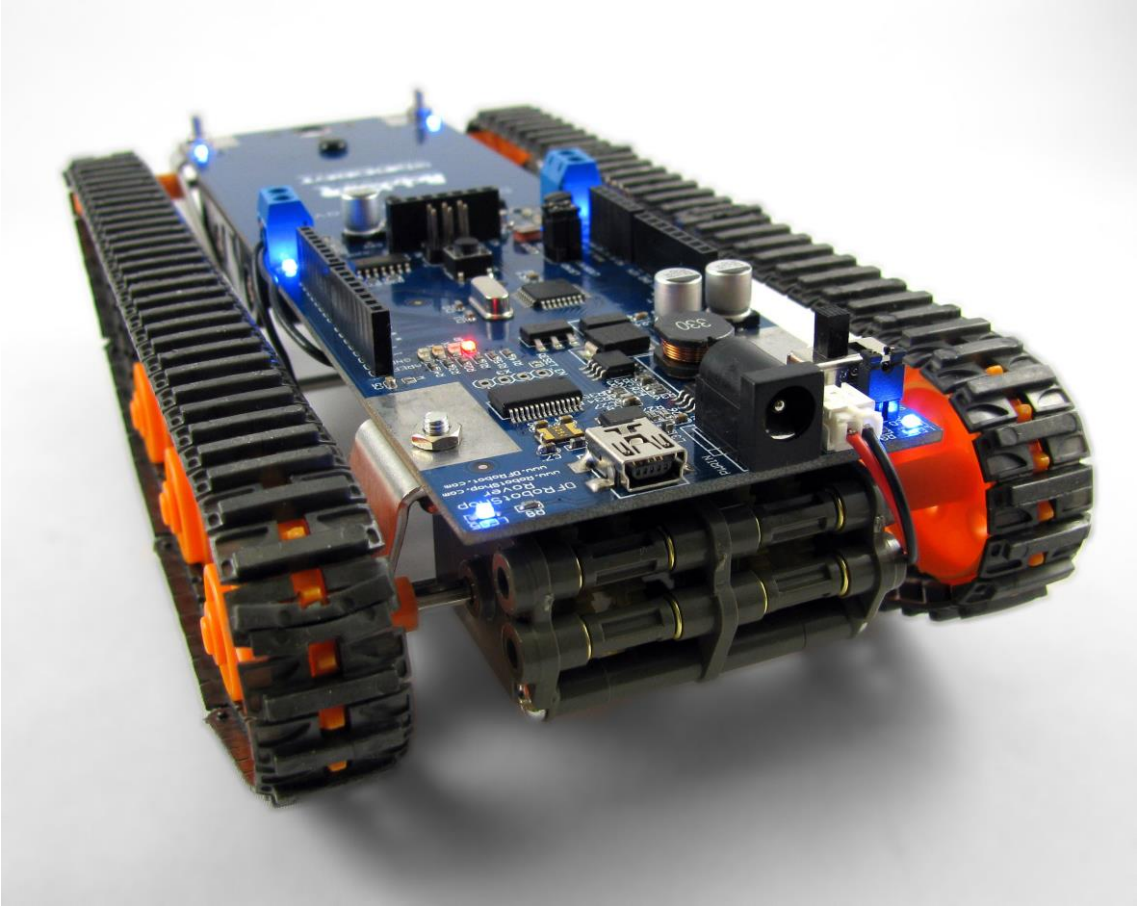
O conceito do Arduino surgiu na Itália, em 2005, com o objetivo de criar um dispositivo que fosse utilizado em projetos/protótipos construídos de uma forma menos dispendiosa do que outros sistemas disponíveis no mercado, era muito caro trabalhar com micro controladores na época.

Ele pode ser usado para desenvolver diversos projetos, podendo ser ligado ao computador ou outros circuitos.

O Arduino foi projetado com a finalidade de ser extremamente simples e intuitivo de se trabalhar, além de ser multiplataforma, podendo ser configurado em ambientes Linux, Mac OS e Windows.

Um grande diferencial deste dispositivo é ser mantido por uma comunidade que trabalha na filosofia do open-source, desenvolvendo e divulgando gratuitamente seus projetos.

Ele é baseado em uma placa micro controlada, com acessos de Entrada e Saída, sobre a qual foram desenvolvidas bibliotecas com funções que simplificam a sua programação, por meio de uma sintaxe parecida com a das linguagens C e C++.



O Arduino utiliza o micro controlador Atmega, que é um computador em um chip, que contém um microprocessador, memória e periféricos de entrada/saída.

Um microcontrolador pode ser embarcado no interior de algum outro dispositivo, que, neste caso, é o Arduino, para que possa controlar suas funções ou ações. Em resumo, o Arduino é um kit de desenvolvimento, que pode ser visto como uma unidade de processamento capaz de mensurar variáveis do ambiente externo, transformadas em um sinal elétrico correspondente, através de sensores ligados aos seus terminais de entrada.

Com a informação, ele pode processá-la computacionalmente.

Por fim, ele pode ainda atuar no controle ou no acionamento de algum outro elemento eletroeletrônico conectado ao terminal de saída.

Programação do Arduino

Agora iremos ver um pouco sobre como são estruturados os programas para o Arduino, conhecendo a linguagem usada como referência e suas principais funções.

No site Athos Electronics você também encontra um artigo completo sobre a programação, no link abaixo:

http://athoselectronics.info/arduino-programacao/

Também veremos as funcionalidades extras que o uso de bibliotecas nos proporciona.

Linguagem de Programação

Os programas para o Arduino são implementados tendo como base a linguagem C++.

Ele preserva a sintaxe clássica na declaração de variáveis, nos operadores, nos ponteiros, nos vetores, nas estruturas e em muitas outras características da linguagem.

Com isso, temos as referências da linguagem. Elas podem ser divididas em três partes principais: As estruturas, os valores (variáveis e constantes) e as funções.

As estruturas de referências:

- Estruturas de controle (if, else, break, ...)
- Sintaxe básica (#define, #include, ; , ...)
- Operadores aritméticos e de comparação (+, -, =, ==, !=, ...)
- Operadores booleanos (&&, ||, !)
- Acesso a ponteiros (*, &)
- Operadores compostos (++, –, +=, ...)
- Operadores de bits (|, ˆ , , ...)

Os valores de referências:

- Tipos de dados (byte, array, int , char , ...)
- Conversões (char(), byte(), int(), ...)
- Utilitários (sizeof(), diz o tamanho da variável em bytes)

O software que vem no Arduino já tem várias funções e constantes para facilitar a programação, como:

- setup()
- loop()
- Constantes (HIGH | LOW , INPUT | OUTPUT , ...)
- Bibliotecas (Serial, Servo, Tone, etc.)

Funções

As funções são usadas para o desenvolvimento de um projeto usando o Arduino, principalmente para os iniciantes no assunto.

Essas funções já implementadas e disponíveis em bibliotecas direcionam e exemplificam as funcionalidades básicas do Arduino.

Temos como funções básicas e de referências as seguintes funções:

- Digital I/O

  pinMode()
  digitalWrite()
  digitalRead()

- Analógico I/O

  analogReference()
  analogRead()
  analogWrite() - PWM

- Avançado I/O

  tone()
  noTone()
  shiftOut()
  pulseIn()

- Tempo

  millis()
  micros()
  delay()
  delayMicroseconds()

- Matemática

  min()
  max()
  abs()
  constrain()
  map()
  pow()
  sqrt()

- Trigonométrica

  sin()
  cos()
  tan()

- Números aleatórios

  randomSeed()
  random()

- bits e Bytes

  lowByte()
  highByte()
  bitRead()
  bitWrite()
  bitSet()
  bitClear()
  bit()

- Interrupções externas

  attachInterrupt()
  detachInterrupt()
  Interrupções
  interrupts()
  noInterrupts()

- Comunicação Serial

  Serial.read()
  SerialEvent() 3.3

Bibliotecas

Usar bibliotecas nos proporciona um horizonte de programação mais amplo e diverso se compararmos com a utilização apenas de estruturas, valores e funções.

Isso é perceptível quando analisamos os assuntos que são abordados por cada biblioteca em específico.

Lembrando que, para usar uma biblioteca, ela já deve estar instalada e disponível na sua máquina.

Instalando a IDE do Arduino

A IDE - Integrated Development Environment ou Ambiente de Desenvolvimento Integrado é onde você vai programar o Arduino, a partir de um computador.

Para fazer a instalação, basta acessar a página de Download do Arduino:

https://www.arduino.cc/en/Main/Software

E selecionar a sua plataforma.

O Download vai começar em seguida, e então é só seguir com a instalação, sem complicações.

Quando você abrir a IDE, vai se deparar com a seguinte tela:

Clique no botão Tools ou Ferramentas > Board > e selecione sua placa.

Pronto! Sua IDE está instalada e configurada!

Projetos com Arduino

Agora vamos ver alguns projetos simples, para dar os primeiros passos com Arduino e ainda aprender eletrônica.

Acionando LED interno

Vamos começar com o exemplo *Blink,* que já vem no aplicativo.

Para encontrar o exemplo clique em File → Examples → Digital → Blink.

O programa tem como objetivo acender e apagar o LED de um em um segundo. Para fazer esse projeto não é necessário de nenhum outro componente além do Arduino.

Primeiramente, vamos criar uma variável chamada ledPin que armazenará o número da porta onde o LED está conectado:

```
int ledPin = 13;
```

Assim quando nos referirmos a variável *ledPin* estaremos nos referindo a porta 13.

O próximo passo é classificar o *ledPin* como uma Saída, isto é feito da seguinte maneira:

```
void setup() {
pinMode(ledPin, OUTPUT);
}
```

A função *pinMode()* tem como primeiro parâmetro o pino e como segundo parâmetro se ele é pino de entrada ou saída.

O programa vai rodar em um loop, pois não há ocorrências ou interferências que mudem o estado. Dentro do loop terá uma função que fará o LED ficar aceso por 1 segundo e depois ficar apagado por mais um segundo, e assim seguir o loop.

O Código vai ficar assim:

```
void loop() {
digitalWrite(ledPin, HIGH);
delay(1000);
digitalWrite(ledPin, LOW);
delay(1000); }
```

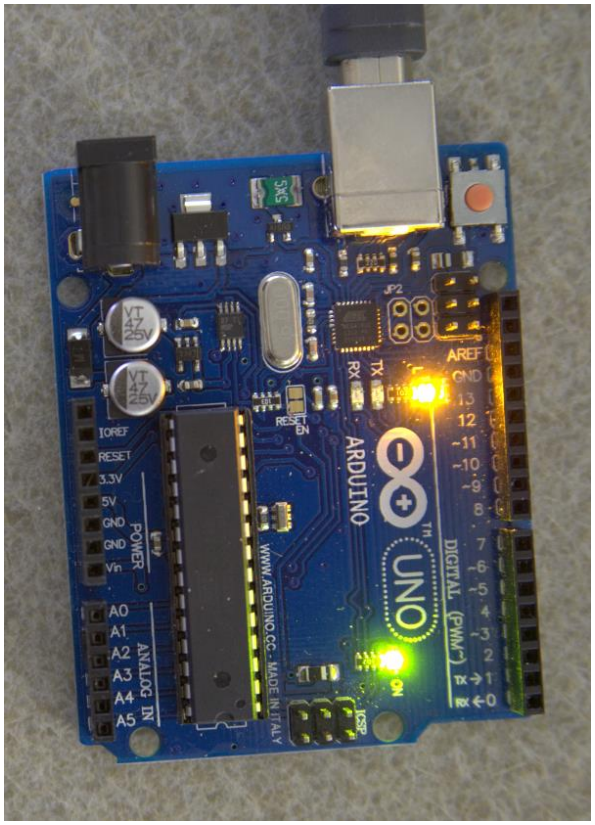
A função *digitalWrite()* escreve uma informação digital, ou seja, 0 (LOW) ou 1 (HIGH).

Seu primeiro parâmetro é o pino de saída, que no caso é o *ledPin*. O segundo parâmetro é o estado, que no caso é a saída, HIGH ou LOW.

Quando uma porta digital está em LOW, ela fica com 0V, já quando está em HIGH, fica em 5 V.

A função *delay()* é um atraso que se dá para a continuação da leitura do programa, logo como foi escrito que o *ledPin* estará aceso, só após um segundo, ou como está escrito 1000 milissegundos, irá ler a linha seguinte que escreve que a saída do *ledPin* é baixa, e o mesmo ocorre mais uma vez.

Faça o upload e você vai ver o LED do Arduino piscar.

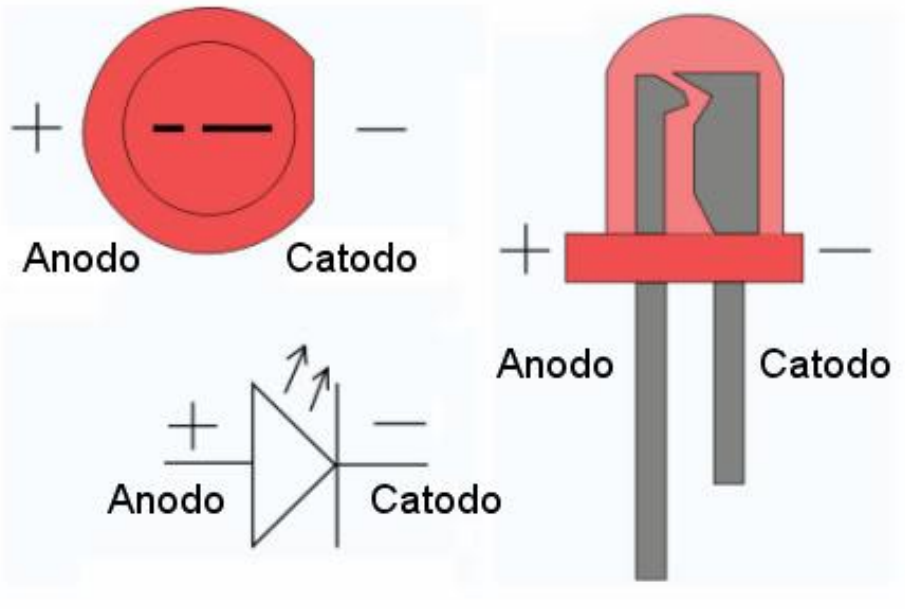


Ligando LED Externo

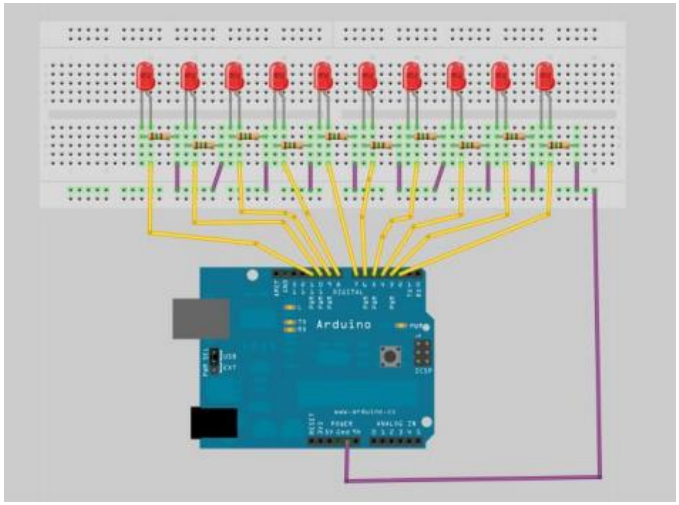
Agora, exploraremos melhor as saídas digitais. Nesse projeto vamos usar 10 LEDs.

Os LEDs vão acender em sequência e ficarão acesos por 1 segundo, até que o último apagará e todos os outros em sequência vão apagar.

Antes de seguirmos, veja na imagem abaixo como identificar qual é o positivo e o negativo dos LEDs:



Para fazer o projeto ligue o positivo dos LEDs nos pinos digitais de 2 a 11 e a outra ponta na protoboard, no polo negativo dos LEDs ligue resistores de 150 Ω, em série, e a outra ponta de todos os resistores no pino terra do Arduino, GND na placa. Veja como deve ficar:



As primeiras linhas da programação vão ser para declarar as variáveis, vamos precisar de uma variável constante e inteira de valor 10, em nosso caso chamamos de *LEDCount.* Outra variável que vamos usar é um vetor com 10 posições, enumerados de 2 a 11, que são os números dos pinos de saída digital que serão utilizados que também possuem valores inteiros, chamamos o vetor de *LEDPin*.

Veja a declaração que vamos fazer:

```
const int LEDCount = 10;
int LEDPins[] = { 2, 3, 4, 5, 6, 7,8,9,10,11 };
```

Agora com essas variáveis declaradas vamos definir o vetor *LEDPins* como pinos de saída, para isso utilizaremos um loop, dessa forma:

```
void setup() {
        for (int thisLED = 0; thisLED < LEDCount; thisLED++) {
                pinMode(LEDPins[thisLED], OUTPUT); }
}
```

Na primeira posição da estrutura *for* declaramos uma variável que começa em 0. Na segunda posição colocamos a condição para o *for*, e na terceira a cada vez que é verificada a condição da estrutura *for*, com execução da primeira, é acrescido 1 ao valor de *thisLED*, que é a variável que utilizaremos para chamar as determinadas posições do vetor *LEDPins*.

A função *pinMode(),* como vista no exemplo anterior, está declarando que o vetor *LEDPins* é um vetor de saída de dados.

Agora será iniciado um loop, que fará com que o programa sempre rode, dentro dele terá um for que acenderá todos os LEDs sequencialmente, com um intervalo de 1 segundo (1000 ms) entre cada LED. O programa vai ficar assim:

```
void loop() {
for (int thisLED = 0; thisLED < LEDCount; thisLED++) {
        digitalWrite(LEDPins[thisLED], HIGH);
                delay(1000); }
                        delay(1000);
```

Perceba que o for é da mesma maneira que o último, pois a ideia é sempre se referir às posições do vetor, a diferença aqui é que para cada posição estamos acendendo um LED, usando a função *digitalWrite().*

A função *delay()* foi utilizada para que seja mais fácil de visualizar que cada LED acende de cada vez, e que após todos os LEDs acenderem, todos ficam acesos por mais um segundo. Agora será feito o mesmo, mas para apagar os LEDs, veja o código:

```
for (int thisLED = 9; thisLED >= 0; thisLED--) {
      digitalWrite(LEDPins[thisLED], LOW);
            delay(1000); }
                  delay(1000); }
```

A variável *thisLED*, utilizada apenas no for, começa com valor 9 e é decrescida de 1 até que chegue em 0, logo os LEDs vão apagar do último a acender para o primeiro, e permanecerá apagado por 1 segundo.

E assim terminamos nosso segundo projeto.

Conclusão

Este e-book vai ir crescendo com o tempo, vou acrescentar cada vez mais projetos e conceitos de Eletrônica e Arduino.

Para conhecer mais projetos de Arduino, eletrônica e Elétrica, acesse o site Athos Electronics

http://athoselectronics.info

Conheça também nossos cursos de Eletrônica!