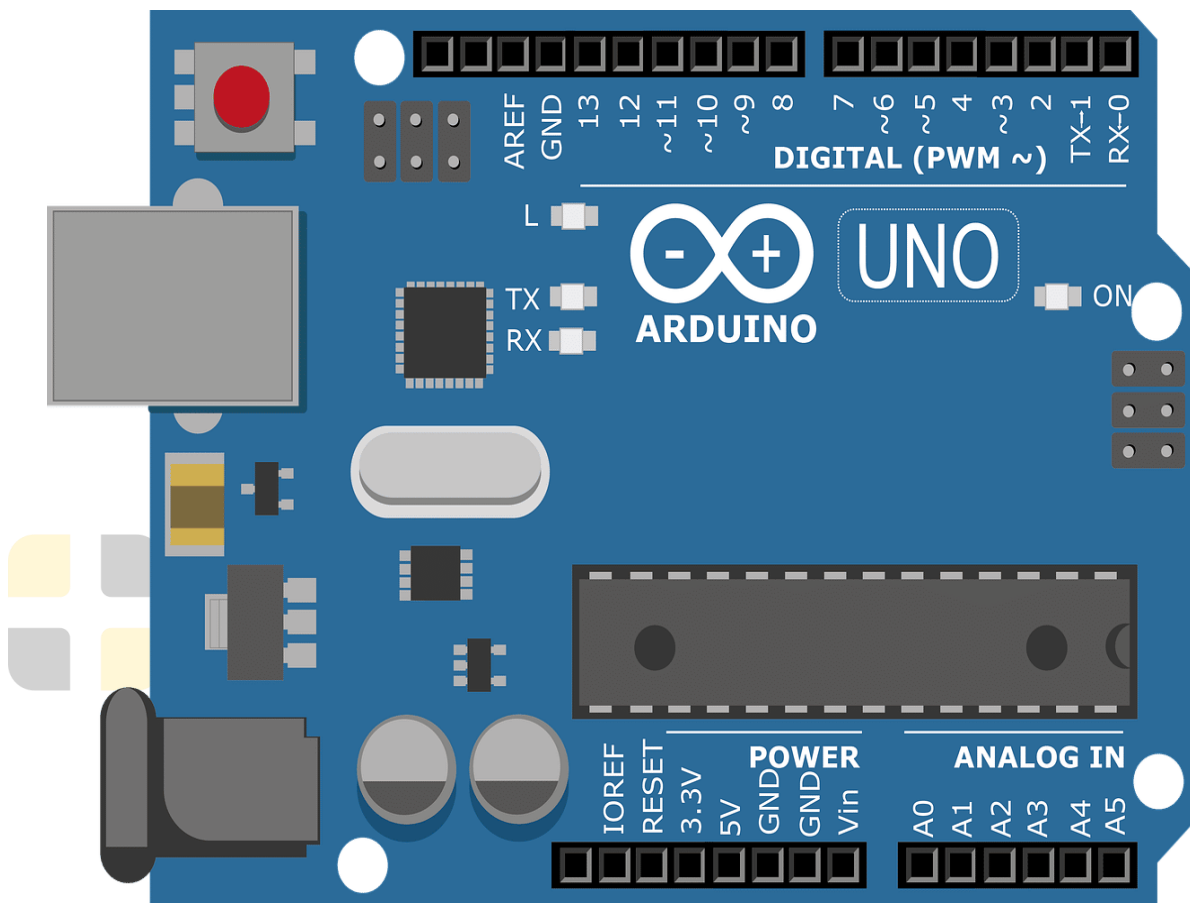


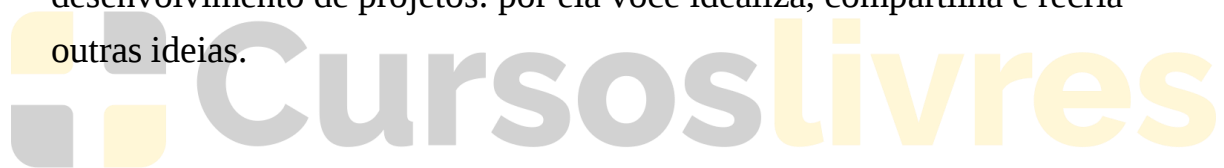
Arduino



Arduino é uma **plataforma de prototipagem eletrônica** muito versátil e amplamente utilizada por estudantes, hobbistas e profissionais das mais diversas áreas. O objetivo principal do Arduino é tornar o acesso à prototipagem eletrônica mais fácil, mais barata e flexível. As versões mais simples da palca utilizam um microcontrolador da família Atmel AVR e uma linguagem de programação baseada em C/C++. Com ele é possível criar projetos variados em eletrônica, desde os mais simples até aplicações intermediárias como **Internet das Coisas (IoT)**, **Robôs**, **Sistemas de Automação** Residencial ou Industrial, **Alarmes** e outros.

As funcionalidades do Arduino também podem ser facilmente ampliadas, ou seja, você não precisa trocar a placa principal caso queira expandir os recursos do seu projeto. Basta acrescentar **sensores, módulos e shields** para incorporar novas funções. Além disso, depois de programado, o Arduino pode ser utilizado sem a necessidade de um computador, já que o programa instalado na placa permanece em loop, repetindo sem parar, sendo necessário apenas uma fonte de alimentação para que a placa funcione.

O Arduino foi desenvolvido com base no conceito **open-source**, em tradução literal “código aberto”, que significa que o projeto da placa e o firmware podem ser utilizados livremente por outros desenvolvedores e fabricantes. Essa forma de inserção na eletrônica e programação inovou o movimento maker, também conhecido por sua característica “faça você mesmo”. A tecnologia e os softwares livres têm promovido uma quarta revolução industrial, que reflete na comunidade maker e no modelo de criação e desenvolvimento de projetos: por ela você idealiza, compartilha e recria outras ideias.



Como e quando surgiu?

Antes de chegarmos no Arduino, temos que abordar a criação da linguagem que inspirou essa plataforma: o Wiring. Desenvolvido por Hernando Barragán, essa linguagem foi formulada em sua tese de mestrado no Instituto Design de Interação de Ivrea (IDII). O objetivo da tese era tornar mais fácil para artistas e designers trabalharem com eletrônica, abstraindo os detalhes frequentemente complicados dessa área para que eles pudessem se concentrar em seus próprios objetivos.

Estes foram os principais elementos resultantes do Wiring:

- Ambiente de desenvolvimento integrado simples (IDE), baseado no IDE Processing.org em execução no Microsoft Windows, Mac OS X e Linux para criar programas de software ou “esboços”, com um editor simples;
- “Linguagem” simples ou “estrutura” de programação para microcontroladores;
- Integração completa do conjunto de ferramentas (transparente para o usuário);
- Bootloader para upload fácil de programas;
- Monitor serial para inspecionar e enviar dados de/para o microcontrolador;
- Software open source;
- Projetos de hardware de código aberto baseados em um microcontrolador Atmel;
- Referência online abrangente para os comandos e bibliotecas, exemplos, tutoriais, fórum e uma vitrine de projetos feitos usando o Wiring.

Quando a IDII fabricou o primeiro conjunto de placas de Wiring, o custo era provavelmente em torno de US\$ 50,00 cada. Essa queda foi bastante considerável em relação ao preço das placas que estavam disponíveis, mas ainda era um custo significativo para a maioria das pessoas.

Em 2005, Massimo Banzi, junto com David Mellis (um aluno do IDII na época) e David Cuartielles, adicionaram um suporte para o microcontrolador ATmega8 mais barato à Wiring. Em seguida, eles bifurcaram (ou copiaram) o código-fonte do Wiring e começaram a executá-lo como um projeto separado, chamado Arduino.

A primeira placa foi composta por um microcontrolador Atmel e programada via Ambiente de Desenvolvimento Integrado (IDE), com linguagem baseada em C/C++. Já os circuitos de entrada e saída, poderiam ser conectados no computador por cabo USB.

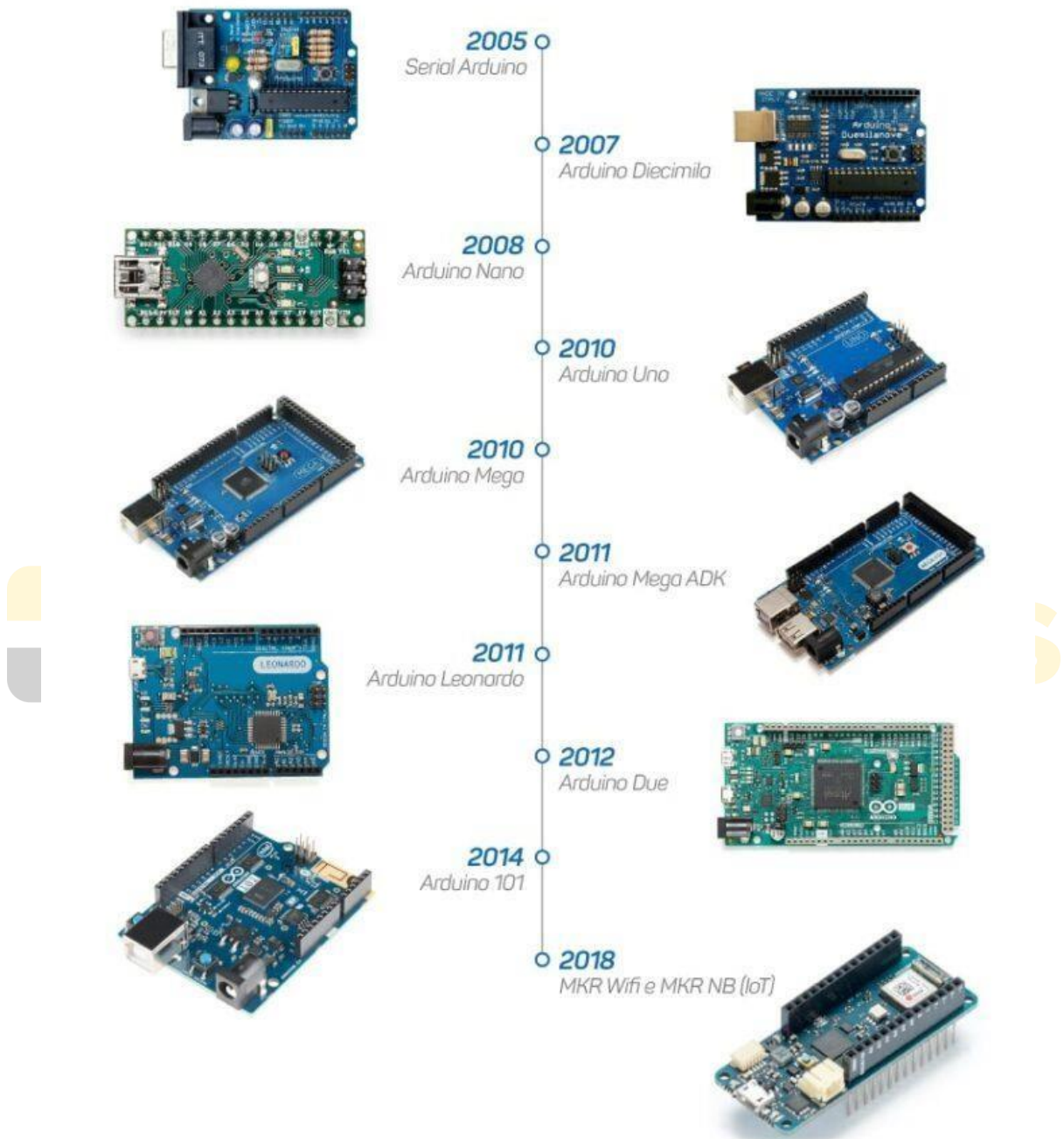


Serial Arduino – a primeira placa

Na estruturação do Arduino, o conceito de hardware livre (*open hardware*) foi adotado, e isso significa que qualquer pessoa pode montar, modificar, melhorar e personalizar, partindo do mesmo projeto básico de hardware. A única restrição é quanto a replicação da marca em novos projetos, o que caracterizaria plágio.

O primeiro Arduino criado foi o Serial Arduino, lançado em 2005, demarcando um novo momento no universo maker. Seguido da **Diecimila** (2007), **Nano** (2008), **Uno** e **Mega** (2010), **Mega ADK** e **Leonardo** (2011), **Due** (2012), **101** (2014) e **MKR Wifi e NB IoT** (2018). Na imagem abaixo, podemos conferir a evolução das placas Arduino.

LINHA DO TEMPO ARDUINO



Curiosidade: A palavra Arduino possui origem germânica Harduwin ou Hardwin, nativa do latim Teutoni, que significa “firmeza e resistência”. Outra curiosidade é que a placa não possuía um nome, até que os criadores frequentaram um pub chamado Bar di Re Arduino, em homenagem ao rei

d'Ivrea, que governou o país entre 1002 a 1014, marcado na história por seus projetos inovadores.

Tipos de placa Arduino

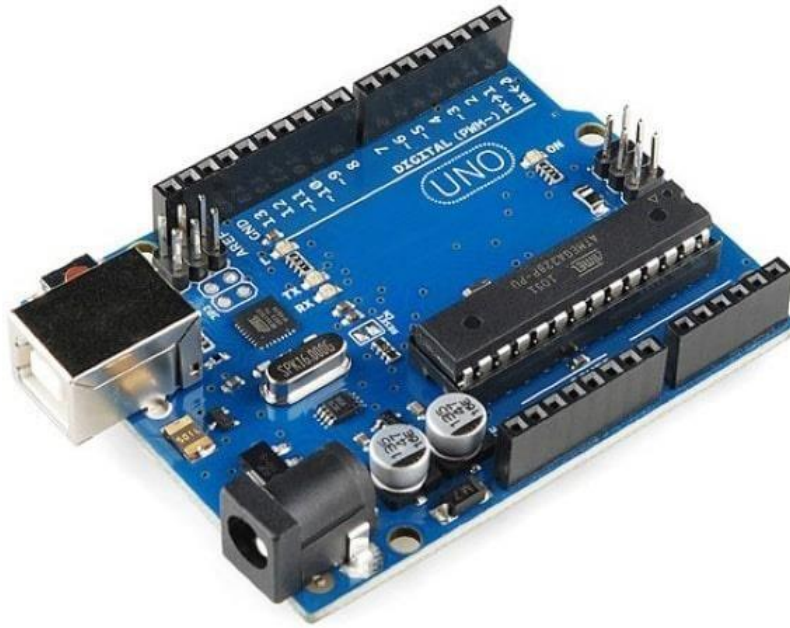
O processo de desenvolvimento das tecnologias do Arduino, desde sua primeira versão, produziu um conjunto de placas que podem ser classificadas em função dos recursos e poder de processamento ou controle:

Placas Básicas

São placas mais simples e com menos recursos, ideais para os **makers iniciantes** ou utilização em **projetos de baixa complexidade**.

Arduino Uno

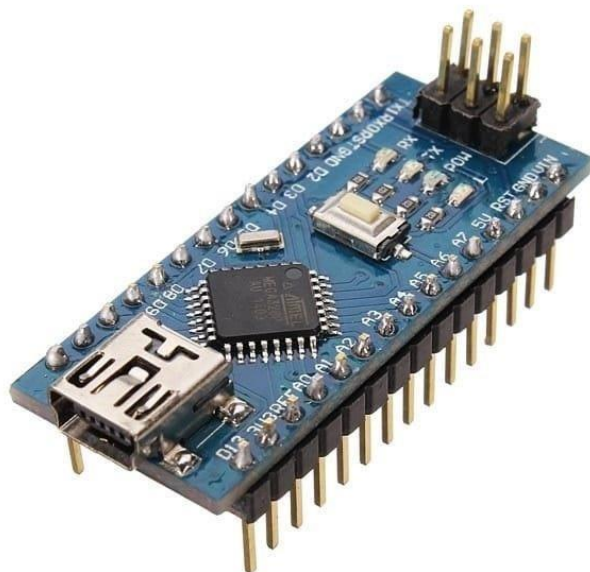
É a placa mais recomendada para quem está começando na plataforma. Ela possui **excelente custo-benefício**, quantidade de porta (entrada/saída) suficiente para a criação de protótipos com vários sensores e módulos conectados. O microcontrolador da placa **Uno** é o ATMega328P, com clock de 16MHz, 14 pinos de I/O, sendo 6 analógicos e 6 com função PWM (*Pulse Width Modulation*). A placa Uno tem 32KB de memória flash, onde são armazenados os programas. A conexão com o computador usa um cabo USB A/B, o mesmo utilizado em impressoras USB, podendo ser alimentado com uma fonte externa chaveada de 7 a 12 VDC.



Placa Arduino Uno

Arduino Nano

A placa **Nano** trata-se da **versão reduzida da Uno**, indicada para projetos compactos, como robóticos e estações meteorológicas. Diferente da placa anterior, a Nano possui uma porta mini USB para conexão, mas sem acesso para fonte externa. No entanto, também possui controlador ATmega328, 16MHz, 22 portas de entrada e saída, sendo 8 analógicas e 6 com função PWM. A quantidade de memória flash também é a mesma da Uno, 32KB.



Placa Arduino Nano

Arduino Leonardo

Primeira placa da linha Arduino que possui funções USB incorporadas ao microcontrolador ATmega32u4. Com isso, além de eliminar a necessidade de outros circuitos para comunicação com o computador, permite sua detecção pelo sistema como um dispositivo USB comum, da mesma forma que acontece com um mouse ou teclado, por exemplo. A memória flash da **placa Leonardo** possui 32KB, com 20 pinos I/O, sendo 7 com função PWM e 12 analógicas.



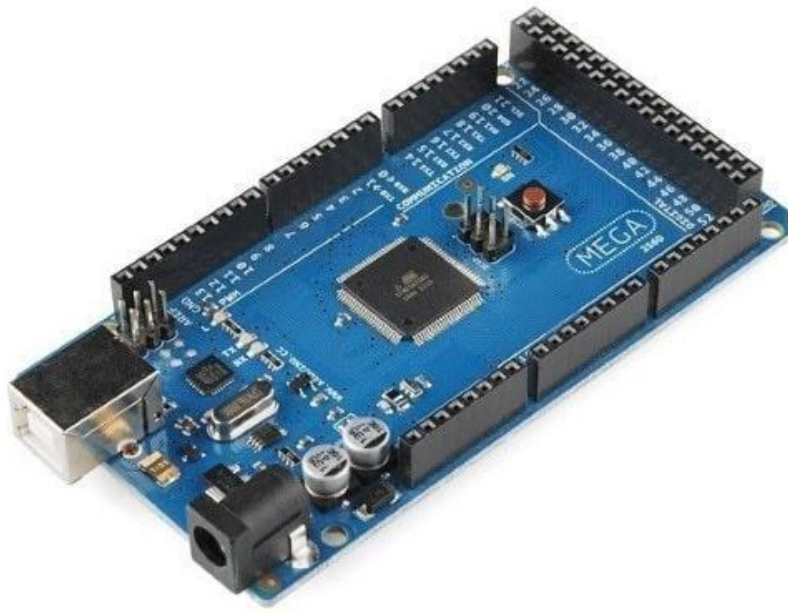
Placa Arduino Leonardo

Placas com Recursos Avançados

As placas mais avançadas se destacam pela **rapidez e quantidade de recursos**, dependendo do modelo.

Arduino Mega 2560

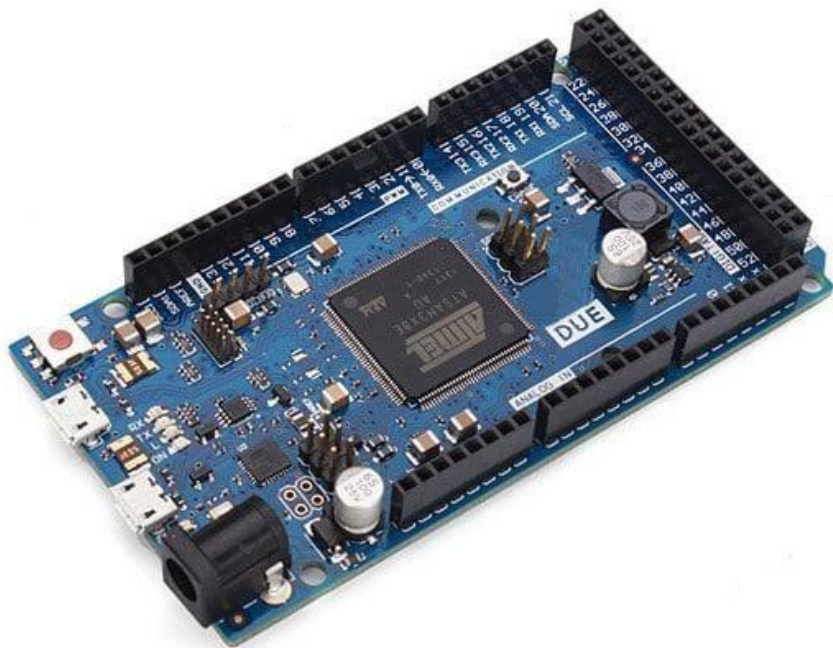
Pelos aspectos semelhantes, acabou sendo conhecido como “**irmã maior**” da **Arduino Uno**, porém com uma quantidade maior de portas, sendo 54 digitais, dessas 15 podem ser PWM, e 16 analógicas. Apesar de possuir um processador com o mesmo clock da versão mais simples, 16MHz, em conjunto com a memória flash de 256KB entrega um maior processamento, sendo ideal para programas mais pesados. Uma das grandes vantagens do Arduino Mega é utilizar a mesma disposição dos pinos da Uno, ou seja, os shields projetados para o Arduino Uno vão funcionar com no Mega.



Placa Arduino Mega2560

Arduino Due

Primeira placa da linha a ser equipada com um microcontrolador ARM de 32 bits, com Atmel SAM3X8E ARM Cortex-M3 rodando em 84MHz. Em termos de portas, ela não é muito diferente da Mega, sendo 54 pinos de I/O, dos quais 12 podem ser usados como saídas PWM, e 12 portas analógicas. Ela também possui mais memória flash, 512KB. Além disso, a conexão da **placa Due** com o computador também é feita por um cabo micro USB.



Cursoslivres

Placa Arduino Due

Arduino Uno WiFi

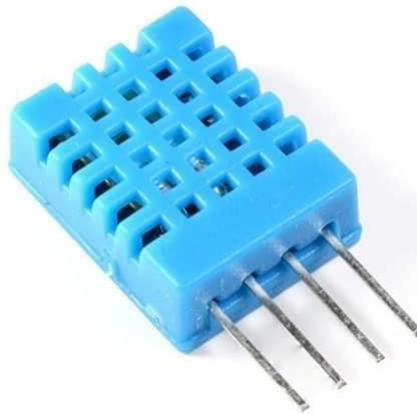
A novidade desta versão é a conectividade wireless. Diferentemente dos outros modelos, os circuitos de conexão wifi já estão embutidos na placa, reduzindo consideravelmente o caminho para a criação de projetos IoT. O **Arduino Uno WiFi** usa um processador ATmega4809, também compatível com os programas escritos para o ATmega328P. Nesta edição são 14 pinos de I/O, 5 deles com função PWM e 6 portas analógicas. A memória flash também é um pouco maior, 48KB.



Placa Arduino Uno Wifi

Sensores, módulos e motores

Para expandir as aplicações das placas foram desenvolvidos diversos componentes, sendo que a maioria deles funciona com qualquer tipo de placa da linha, requerendo, às vezes, pequenas adaptações. O **módulo sensor ultrassônico HC-SR04** é um item popular, básico e essencial em todo projeto. Esse tipo de sensor pode ser usado para os “olhos” de um robô, por exemplo, detectando obstáculos, para mudarmos a direção das rodas ou medindo uma distância, como mostramos no post [Como conectar o Sensor Ultrassônico HC-SR04 ao Arduino.](#)



Módulo sensor ultrassônico HC-SR04.

Outro módulo indispensável na bancada de makers é o **sensor de temperatura e umidade DHT11**. Muito fácil de usar, o DHT permite que você monte, por exemplo, uma estação meteorológica, armazenando e enviando informações do clima para um smartphone, ou monitorando ambientes e gerando alarmes quando a temperatura aumenta ou declina.



Sensor de temperatura e umidade DHT11

Já o **motor de passo** é um item que une versatilidade e precisão para seu projeto, por possibilitar um controle total sobre a movimentação do eixo. Em robôs, por exemplo, pode ser usado para movimentar as rodas para uma posição exata.



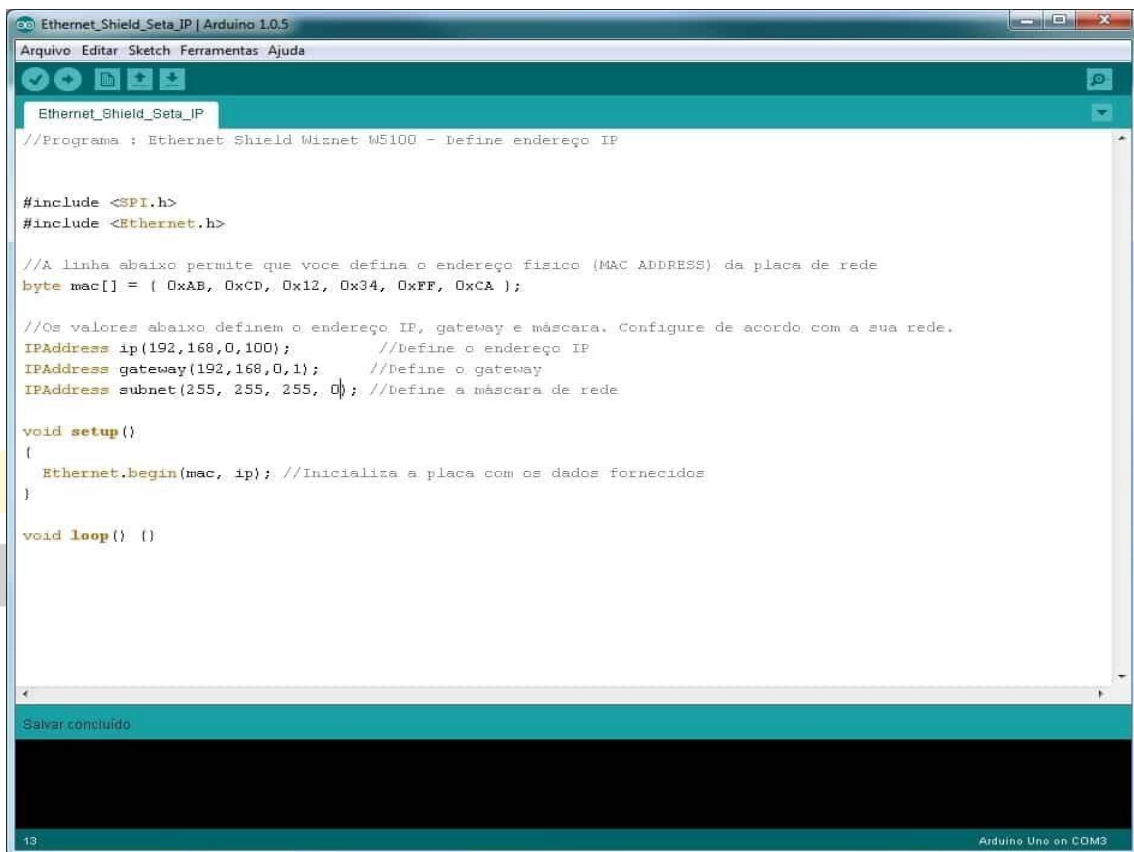
Motor de passo com shield Arduino.

Qual a linguagem de programação usada em Arduino?

O Arduino utiliza uma linguagem de programação própria que foi desenvolvida baseada na linguagem Wiring, um framework de programação open source para microcontroladores que teve origem nas linguagens C/C++. O Arduino também utiliza uma IDE (Integrated development environment) para escrever, compilar e transmitir o código para as placas Arduino. No site oficial do Arduino você pode fazer o download da IDE gratuitamente.

Programando o Arduino

Escrever um programa em Arduino é muito simples. Tudo o que você precisa é conectá-lo ao computador por meio de um cabo USB e utilizar um ambiente de programação chamado IDE, onde você digita, faz os testes, transfere para o dispositivo e a placa já começa operar.



```
//Programa : Ethernet Shield Wiznet W5100 - Define endereço IP

#include <SPI.h>
#include <Ethernet.h>

//A linha abaixo permite que voce defina o endereço físico (MAC ADDRESS) da placa de rede
byte mac[] = { 0xAB, 0xCD, 0x12, 0x34, 0xFF, 0xCA };

//Os valores abaixo definem o endereço IP, gateway e máscara. Configure de acordo com a sua rede.
IPAddress ip(192,168,0,100);           //Define o endereço IP
IPAddress gateway(192,168,0,1);        //Define o gateway
IPAddress subnet(255, 255, 255, 0);    //Define a máscara de rede

void setup()
{
  Ethernet.begin(mac, ip); //Inicializa a placa com os dados fornecidos
}

void loop() {}
```

IDE do Arduino – Tela inicial do sistema

Uma vez feito o programa, basta transferi-lo para o Arduino e o mesmo começa a funcionar.

Você não precisa ter um conhecimento avançado em linguagem C para programar o Arduino. Os primeiros passos podem considerar a estrutura básica, composta por dois blocos:

setup() – É nessa parte do programa que você configura as opções iniciais do seu programa: os valores iniciais de uma variável, se uma porta será utilizada como entrada ou saída, mensagens para o usuário, etc.

loop() – Essa parte do programa repete uma estrutura de comandos de forma contínua ou até que algum comando de “parar” seja enviado ao Arduino.

Vamos ver exatamente como isso funciona, levando em consideração o programa abaixo, que acende e apaga o led embutido na placa em intervalos de 1 segundo:

```
1  //Programa : Pisca Led Arduino
2  //Autor : FILIPEFLOP
3
4  void setup()
5  {
6    //Define a porta do led como saída
7    pinMode(13, OUTPUT);
8
9  void loop()
10 {
11   //Acende o led
12   digitalWrite(13, HIGH);
13
14   //Aguarda o intervalo especificado
15   delay(1000);
16
17   //Apaga o led
18   digitalWrite(13, LOW);
19
20   //Aguarda o intervalo especificado
21   delay(1000);
```

21 }
22
23

A primeira coisa que fazemos no início do programa é colocar uma pequena observação sobre o nome do programa, sua função e quem o criou:

```
1  | //Programa : Pisca Led Arduino  
2  | //Autor : FILIPEFLOP
```

Comece uma linha com barras duplas (//) e tudo o que vier depois dessa linha será tratado como um comentário. Uma das boas práticas de programação é documentar o seu código por meio das linhas de comentário. Com elas, você pode inserir observações sobre como determinada parte do programa funciona ou o que significa aquela variável **AbsXPT** que você criou. Isso será útil não só para você, se precisar alterar o código depois de algum tempo, como também para outras pessoas que utilizarão o seu programa.

Após os comentários, vem a estrutura do **SETUP**. É nela que definimos que o pino 13 da placa será utilizado como saída.

```
4  | void setup()  
5  | {  
6  |   //Define a porta do led como saída  
7  |   pinMode(13, OUTPUT);  
8  | }
```

Por último, temos o **LOOP**, que contém as instruções para acender e apagar o led, e também o intervalo entre essas ações:

```
10 void loop()
11 {
12     //Acende o led
13     digitalWrite(13, HIGH);
14
15     //Aguarda o intervalo especificado
16     delay(1000);
17
18     //Apaga o led
19     digitalWrite(13, LOW);
20
21     //Aguarda o intervalo especificado
22     delay(1000);
23 }
```

A linha do código contendo **digitalWrite(13, HIGH)** coloca a porta 13 em nível alto (**HIGH**, ou 1), acendendo o led embutido na placa. O comando **delay(1000)**, especifica o intervalo, em milisegundos, no qual o programa fica parado antes de avançar para a próxima linha.

O comando **digitalWrite(13, LOW)**, apaga o led, colocando a porta em nível baixo (**LOW**, ou 0), e depois ocorre uma nova parada no programa, e o processo é então reiniciado.

Porque utilizar o Arduino?

As principais razões para se utilizar a plataforma Arduino em seus projetos são as seguintes:

- Baixo custo de prototipagem;
- Softwares de simulação gratuitos disponíveis;
- Fácil de programar;
- Grande número de tutoriais, artigos e projetos prontos na internet;
- Extensa comunidade de desenvolvedores e *hobbyistas*;
- Não requer experiência ou grandes conhecimentos prévios de eletrônica/programação (no entanto, é recomendável saber os conceitos básicos pelo menos).

O Arduino não é a única plataforma de prototipagem eletrônica do mercado. Existem outros projetos e kits de desenvolvimento, os mais comuns são o Raspberry Pi e o BeagleBone.

Cada qual utiliza um microcontrolador diferente e possui projeto de hardware com características próprias. Os preços também variam bastante e algumas outras plataformas não são tão populares.

A escolha de qual kit de prototipagem utilizar depende das demandas e necessidades que o seu projeto impõe.

Usos e Aplicações

O número de aplicações possíveis com o Arduino é imensurável, porém separamos aqui algumas das principais funções que essa plataforma pode nos oferecer. Confira a seguir.

Eletrônica

Uma das principais utilidades do Arduino é seu uso em projetos de eletrônica. Sua portabilidade e seu custo fazem com que sua escolha para esses tipos de projetos seja a mais viável.

Selecionamos esse post para exemplificar na prática como um projeto de eletrônica funciona com Arduino. Aqui, foi montado um Gamepad apenas utilizando uma plataforma Arduino juntamente com alguns componentes eletrônicos. Super prático, simples de montar e barato. Confira abaixo!

[Gamepad de PC: Arduino Pro Micro](#)

Robótica

Outra grande utilização do Arduino é em projetos de robótica. Essa área está em ascensão constante em nossa sociedade, e a inserção do Arduino nesses projetos também.

Abaixo, exemplificamos esse tipo de aplicação com nosso post sobre Robô seguidor de linha. Ideal para ser utilizado em locomoções de peças e ferramentas em empresas, sua fácil implementação e custo baixo tornam o projeto uma ótima opção para essa área. Confira!

Robô Seguidor de Linha – Tutorial Completo



Uma das aplicações mais úteis do Arduino em nossas vidas é na automação residencial. A possibilidade de facilitar a execução de tarefas e de otimizar o funcionamento de aparelhos eletrônicos dentro de sua própria casa torna essa aplicabilidade extremamente desejável por todos.

Um bom exemplo seria o projeto evidenciado abaixo. A combinação da tecnologia Alexa juntamente com o Arduino torna sua experiência com componentes eletrônicos dentro de sua residência muito mais proveitosa. Confira já!

Automação Residencial com Alexa (Amazon) e NodeMCU

Segurança

Uma das áreas mais importantes para o desenvolvimento da tecnologia é a segurança. A necessidade da minimização de erros e da confiabilidade de garantia nesse ramo torna projetos eletrônicos uma boa pedida para essa área. Alguns exemplos de desenvolvimentos tecnológicos nesse âmbito são a criação de câmeras inteligentes e de sistemas de segurança otimizados. A plataforma Arduino pode ser aplicada de diversas formas em projetos desse tipo.

Escolhemos o seguinte post para elucidar na prática como essas aplicações podem ser implementadas. Nele, é desenvolvido uma porta automática com biometria, na qual, com a simples elaboração de um projeto em Arduino, é possível garantir a segurança de locais desejáveis. Confira abaixo!

[Porta automática com biometria.](#)

Jogos

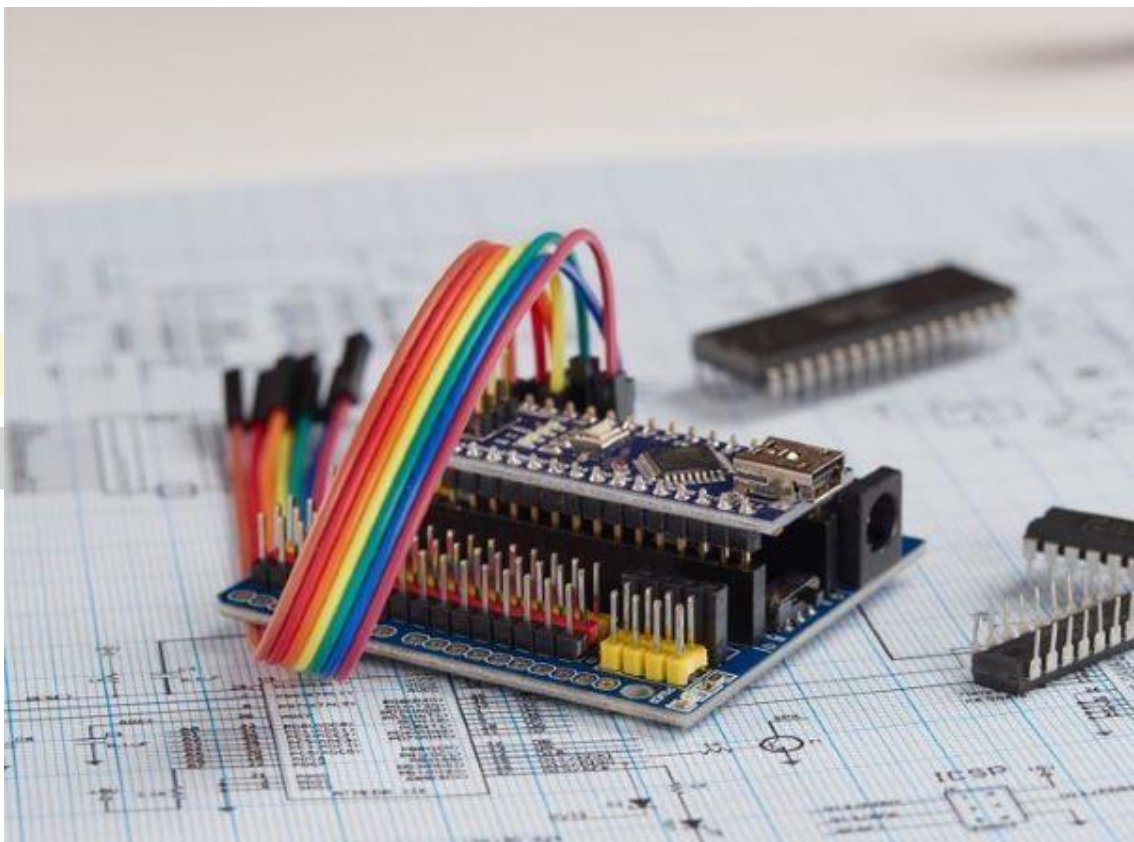
Uma das aplicações mais divertidas do Arduino é na criação de jogos. As possibilidades são numerosas, e a diversão está tanto no desenvolvimento e montagem do projeto quanto no jogo em si.

Um bom jogo desenvolvido em Arduino é o exemplificado abaixo. O objetivo consiste em passar por todo o caminho do fio de cobre/alumínio sem encostar nele. Você pode jogar sozinho e desafiar suas coordenações motoras ou com

algum amigo para saber quem é o melhor. Acesse nosso post abaixo e confira como confeccionar esse projeto!

Não encoste no fio! – Jogo no Arduino

Benefícios do Arduino na criação de projetos: por que utilizar Arduino?



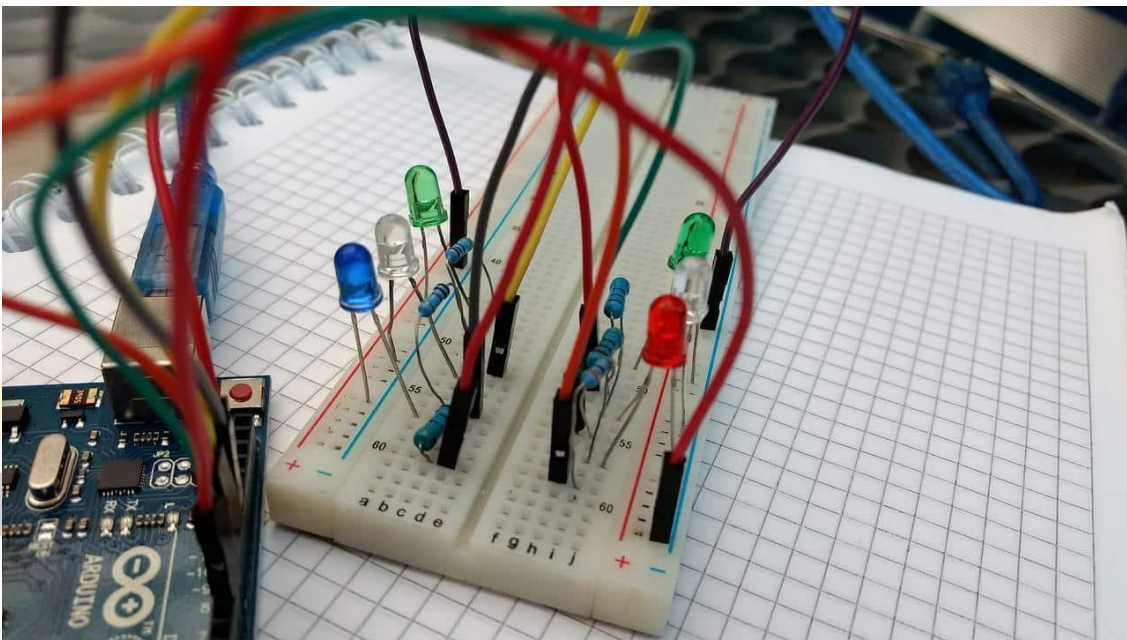
Até aqui, você já sabe o que é Arduino e quais os tipos existentes no mercado. No entanto, já parou para pensar sobre os benefícios dessa placa de código aberto?

Entre os principais benefícios de utilizar o Arduino para a elaboração de projetos, estão:

- prototipagem de baixo custo;

- diversos tutoriais de projetos na internet;
- linguagem simples de programação;
- alto número de possibilidades de execução.

Características da plataforma



Um dos principais motivos que fazem muitos quererem aderir ao Arduino é o seu baixo custo para prototipagem. Além disso, há muitos softwares para simulação distribuídos de forma gratuita.

A facilidade de programar essa plataforma é mais uma vantagem, sendo que também existe uma comunidade de desenvolvedores que é muito grande.

Mas apesar do uso do Arduino não exigir experiência em **eletrônica**, faz-se necessário que o indivíduo tenha um conhecimento mínimo sobre.

A placa do Arduino pode ser alimentada por um adaptador AC/DC, como também se pode utilizar a entrada USB para isso. E há nessas placas **entradas analógicas** que servem para a leitura dos sensores externos.

Mas nessas placas há ainda as **saídas digitais** e PWM, que são usadas com o objetivo de fazer o controle de atuadores, de motores, além de serem usadas para fazer o acionamento de drivers referentes a cargas que vem de fora.

