RESUMO DE NOÇÕES DE ROBÓTICA

SURGIMENTO DA PALAVRA ROBÔ

A palavra "robô" foi criada pelo escritor tcheco Karel Čapek em 1920 para seu livro de ficção científica "Rossum's Universal Robots". A palavra "robô" vem do termo tcheco "robota", que significa trabalho forçado ou escravidão. O livro de Čapek apresentava seres artificiais que eram usados como trabalhadores, mas que eventualmente se rebelavam contra seus mestres humanos. A partir daí, o termo "robô" passou a ser usado para se referir a qualquer dispositivo mecânico ou eletrônico programável que possa executar tarefas automaticamente ou ser controlado remotamente. Hoje em dia, a palavra "robô" é amplamente usada para descrever uma ampla variedade de dispositivos automatizados, desde robôs industriais até robôs domésticos e de entretenimento.

PRIMEIRO ROBÔ

O primeiro robô da história foi criado pelo inventor e engenheiro norte-americano George Devol em 1954. Ele criou um dispositivo mecânico programável chamado "Unimate", que foi o primeiro robô industrial a ser usado em uma linha de montagem para realizar tarefas repetitivas. O Unimate foi inicialmente projetado para manipular objetos quentes e pesados em uma fábrica de fundição de metais, mas logo se mostrou útil em outras indústrias, como a de automóveis. O sucesso do Unimate abriu caminho para a criação de uma ampla variedade de robôs industriais e ajudou a estabelecer a robótica como uma área importante da tecnologia moderna.

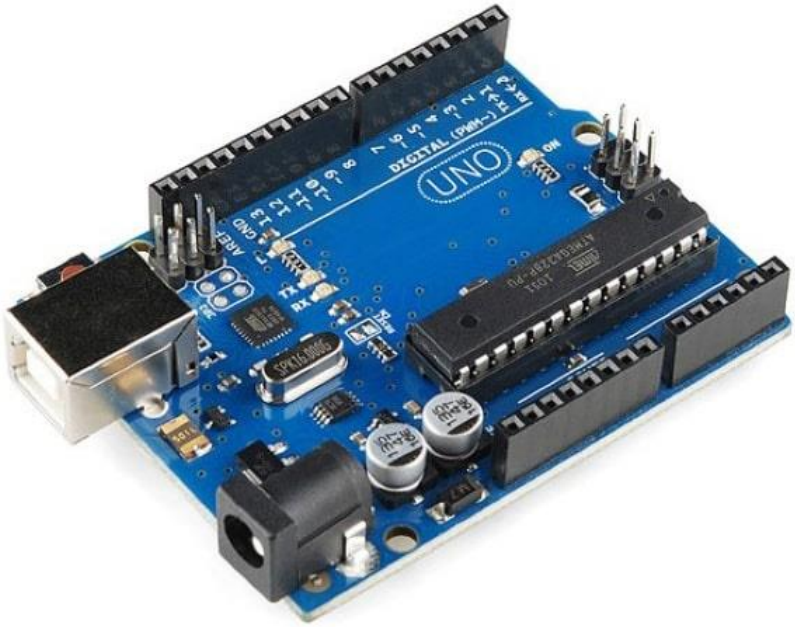
ARDUINO

Arduino é uma plataforma eletrônica de código aberto, projetada para ser fácil de usar e programável. Ela é composta por placas de circuito que contêm um microcontrolador, que pode ser programado para controlar dispositivos eletrônicos, como motores, sensores e luzes. O ambiente de programação Arduino é baseado em uma linguagem de programação simplificada e em um software que permite escrever e carregar códigos para a placa. O objetivo principal do Arduino é tornar a eletrônica acessível a um público mais amplo, permitindo que as pessoas criem projetos interativos e experimentem com tecnologia.

Arduino Uno

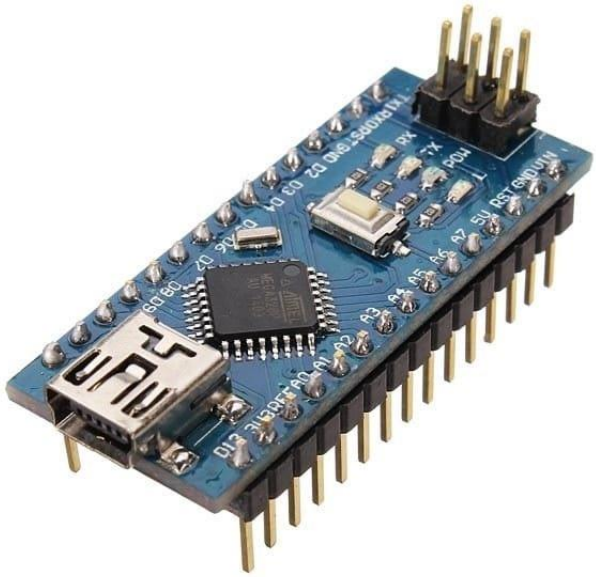
É a placa mais recomendada para quem está começando na plataforma. Ela possui **excelente custo-benefício**, quantidade de porta (entrada/saída) suficiente para a criação de protótipos com vários sensores e módulos conectados. O microcontrolador da placa **Uno** é o ATMega328P, com clock de 16MHz, 14 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.



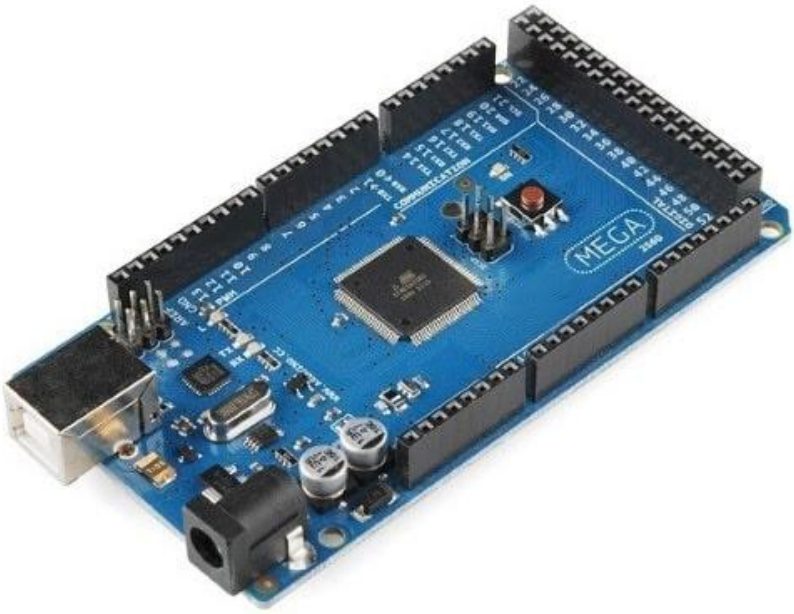
Arduino Nano

A placa **Nano** trata-se da **versão reduzida da Uno**, indicada para projetos compactos, como robóticos e estações meteorológicas.



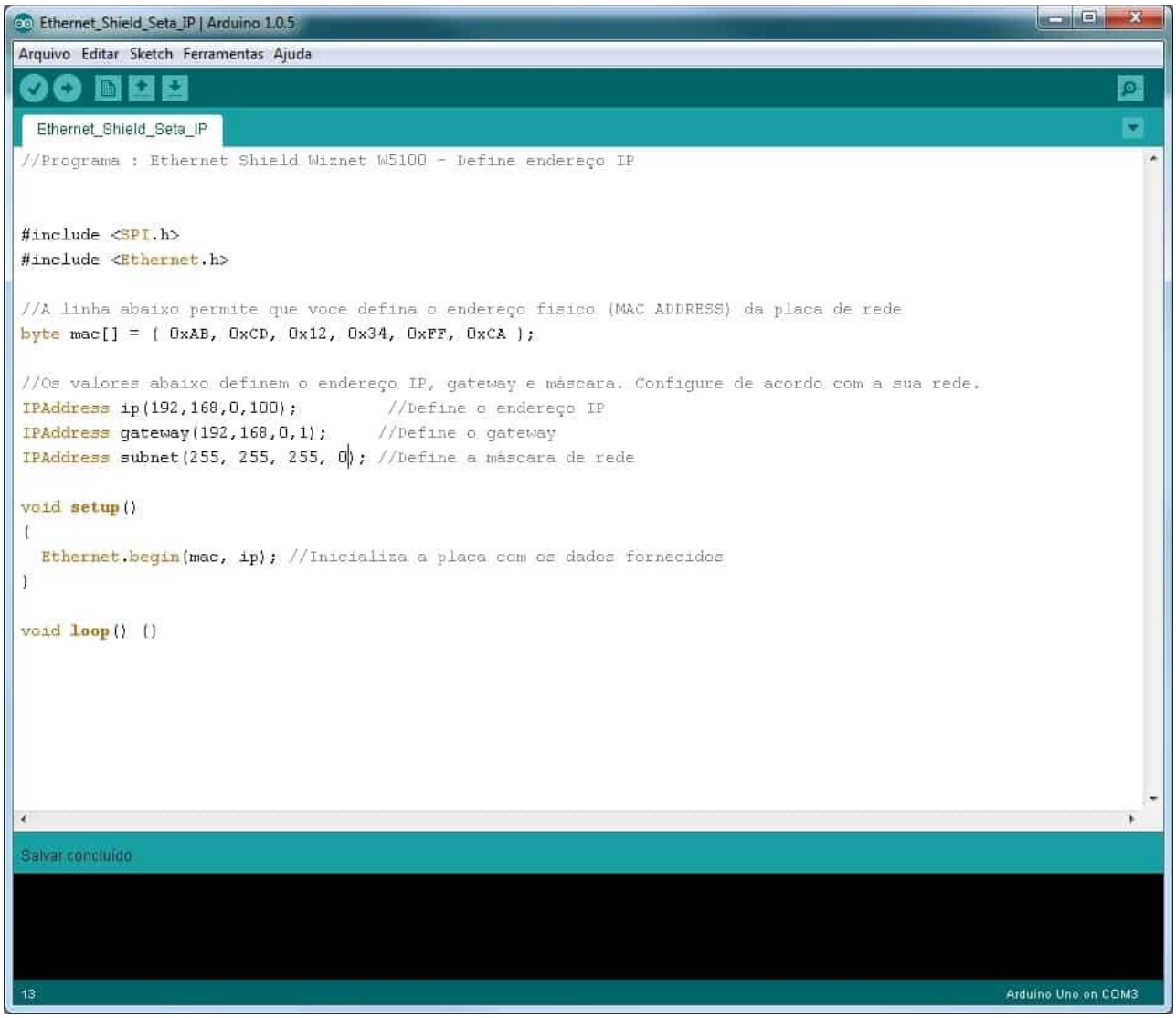
RESUMO DE NOÇÕES DE ROBÓTICA

Arduino Mega 2560



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.



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 : MakerHero
3
4     void setup()
5     {
6       //Define a porta do led como saida
7       pinMode(13, OUTPUT);
8     }
9
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 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 saida
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:

As funções setup() e loop() são duas das funções mais importantes em um programa do Arduino.

RESUMO DE NOÇÕES DE ROBÓTICA

A função setup() é executada uma vez quando a placa é ligada ou resetada. É usada para inicializar as variáveis, configurar os pinos de entrada/saída, definir a velocidade de comunicação serial, entre outras configurações necessárias para o correto funcionamento do programa. A função setup() não retorna nenhum valor e não possui parâmetros.

Exemplo:

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

Neste exemplo, a função setup() define o pino 13 (LED_BUILTIN) como saída e inicia a comunicação serial com uma velocidade de 9600 bits por segundo.

A função loop() é executada continuamente após a execução da função setup(). É usada para implementar o código principal do programa, que será executado repetidamente enquanto a placa estiver ligada. O código contido na função loop() deve ser projetado para ser executado várias vezes por segundo e responder rapidamente a entradas ou eventos. A função loop() não retorna nenhum valor e não possui parâmetros.

Exemplo:

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

Neste exemplo, a função loop() faz o LED_BUILTIN piscar alternando entre ligado e desligado com um atraso de 1 segundo entre cada mudança.

Em resumo, a função setup() é usada para configurar a placa Arduino e a função loop() é usada para implementar a lógica principal do programa que será executado continuamente enquanto a placa estiver ligada.

```
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.

THINKERCAD

O Thinkercad é uma plataforma online gratuita que permite projetar e simular circuitos eletrônicos, incluindo aqueles que usam a placa Arduino. É uma ferramenta útil para estudantes, entusiastas de eletrônica e engenheiros que desejam projetar e testar circuitos antes de implementá-los fisicamente.

O Thinkercad oferece uma interface gráfica intuitiva que permite adicionar componentes eletrônicos ao seu projeto, como resistores, capacitores, LEDs e, claro, a placa Arduino. Você pode escolher entre uma ampla variedade de componentes para criar circuitos personalizados, que podem ser simulados em tempo real para testar o funcionamento.

Uma das grandes vantagens do Thinkercad é a sua integração com a placa Arduino, o que permite criar e testar programas para o microcontrolador diretamente na plataforma. O Thinkercad oferece uma biblioteca completa de código para a placa Arduino, incluindo exemplos de programas para diferentes componentes eletrônicos. Isso torna mais fácil para os usuários iniciantes em programação aprenderem a escrever código para a placa Arduino.

Além disso, o Thinkercad oferece recursos avançados, como simulação de falhas em componentes eletrônicos e a capacidade de adicionar sensores e atuadores para criar projetos mais complexos. É uma ferramenta completa que oferece tudo o que você precisa para projetar e testar circuitos eletrônicos com a placa Arduino.

LINGUAGEM C

Existem três tipos básicos de variáveis em C:

1. Variáveis inteiras (int): armazenam valores inteiros, como 1, 2, -3, 100, etc. Exemplo:
   ```
   int idade = 30;
   ```

2. Variáveis de ponto flutuante (float): armazenam valores decimais, como 3.14, 1.5, -0.5, etc. Exemplo:
   ```
   float preco = 2.99;
   ```

3. Variáveis de caracteres (char): armazenam caracteres individuais, como 'a', 'b', 'C', '*', etc. Exemplo:

```
char letra = 'A';
```

Em C, uma string é uma sequência de caracteres terminada pelo caractere nulo ('\0'). Ela é representada por um array de caracteres, onde cada elemento do array contém um caractere da string. O tamanho da string é determinado pelo número de caracteres que ela contém, mais o caractere nulo no final.

A seguir, temos um exemplo de uma string em C:

```
char nome[20] = "João da Silva";
```

Neste exemplo, "João da Silva" é a string e "nome" é um array de caracteres com tamanho 20. Como a string tem apenas 13 caracteres, os outros elementos do array serão preenchidos com o caractere nulo ('\0').

Vale ressaltar que, em C, as strings são arrays de caracteres e, portanto, podem ser manipuladas diretamente pelo programador. No entanto, é importante ter cuidado ao manipular strings em C, já que operações mal-feitas podem resultar em comportamentos inesperados ou vulnerabilidades de segurança.

FUNÇÕES ÚTEIS PARA PROGRAMAS ARDUÍNO

Existem várias funções em C que podem ser usadas no Arduino para realizar diversas tarefas, desde a leitura de entradas e saídas digitais e analógicas até a comunicação com outros dispositivos externos. Algumas das principais funções em C no Arduino incluem:

1. pinMode(pin, mode): Define o modo de operação do pino especificado como entrada ou saída.
2. digitalWrite(pin, value): Escreve um valor lógico (HIGH ou LOW) no pino especificado.
3. digitalRead(pin): Lê o valor lógico (HIGH ou LOW) do pino especificado.
4. analogRead(pin): Lê o valor analógico (de 0 a 1023) do pino especificado.
5. analogWrite(pin, value): Escreve um valor analógico (de 0 a 255) no pino especificado.
6. delay(milliseconds): Pausa a execução do programa por um determinado período de tempo especificado em milissegundos.
7. Serial.begin(baudrate): Inicializa a comunicação serial com a taxa de transmissão especificada.
8. Serial.println(value): Envia uma mensagem para a porta serial.