

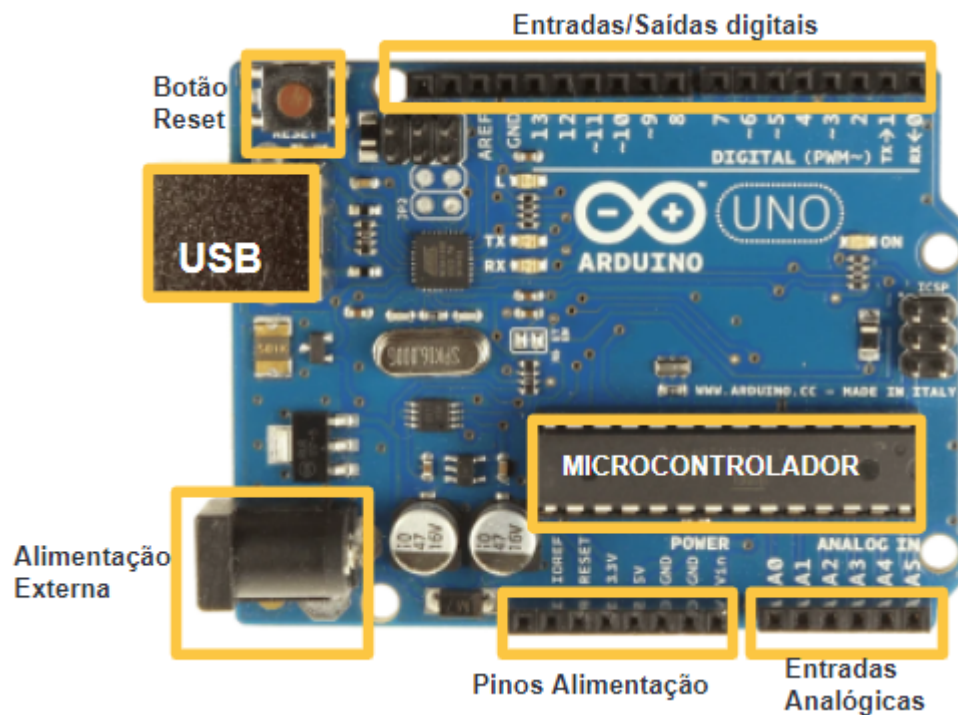
# **MINICURSO DE ARDUINO - PET TI**

## **MATERIAL DE APOIO**

## Arduino

Placa microcontroladora para desenvolvimento de projetos eletrônicos.

### Conhecendo a placa



**Botão Reset:** Reinicia a placa.

**Entrada USB:** Responsável pela alimentação da placa e comunicação com o computador.

**Alimentação Externa:** A alimentação externa pode ser uma fonte ou bateria, operando entre 7V e 12V.

**Microcontrolador:** Cérebro do arduino, o microcontrolador consiste em um único circuito integrado que reúne um núcleo de processador, memórias voláteis e não voláteis e diversos periféricos de entrada e de saída de dados. Ou seja, ele nada mais é do que um computador muito pequeno capaz de realizar determinadas tarefas de maneira eficaz e sob um tamanho altamente compacto.

**Entradas e Saídas Digitais:** O arduino UNO possui 14 pinos digitais que podem ser usados como entrada ou saída. Esses pinos trabalham com dois valores : HIGH(nível lógico alto(5V) - ligado) e LOW(nível lógico baixo(0V) - desligado).

**Entradas Analógicas:** O arduino possui 6 pinos analógicos (A0, A1, A2, A3, A4, A5). O microcontrolador do Arduino trabalha com valores digitais, assim é necessário que os valores analógicos sejam convertidos em digital. O conversor possui 10 bits, assim os pinos podem assumir valores de 0 a 1023.

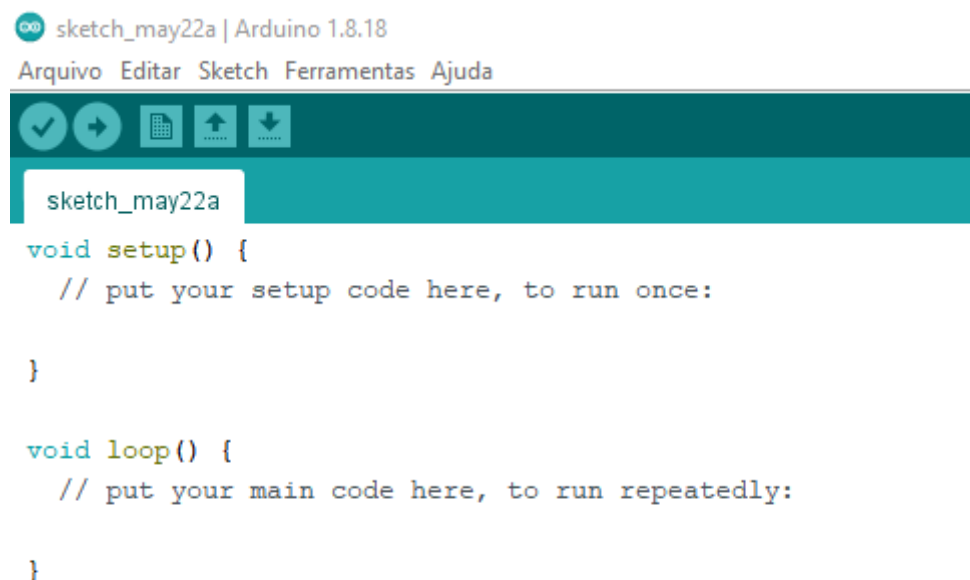
**Pinos de Alimentação:** Usados para alimentação de circuitos e componentes externos.

3,3V: Fornece tensão de 3,3V.

5V: Fornece tensão de 5V.

GND: terra (tensão 0V).

## Programando no Arduino



```
sketch_may22a | Arduino 1.8.18
Arquivo Editar Sketch Ferramentas Ajuda

✓ → 📄 ⬆ ⬇

sketch_may22a

void setup() {
  // put your setup code here, to run once:

}

void loop() {
  // put your main code here, to run repeatedly:

}
```

## Estrutura do código:

É uma boa prática, antes de partirmos para as funções void setup e void loop, definirmos os pinos que iremos utilizar, através do define:

Exemplo:

```
#define PinLed 9
```

No exemplo acima, definimos o pino 9 com o nome PinLed. Ao longo da construção do código, sempre que formos nos referir ao pino 9, usaremos PinLed.

**Função setup():** Função de inicialização. Aqui inicializamos as variáveis e o modo dos pinos.

Na inicialização do pino, temos que configurar se ele será INPUT(entrada) ou OUTPUT(saída).

Sintaxe para configuração do pino:

**PinMode(pino,modo);**

\*Substitua o pino pelo nome da variável que você definiu para ele.

**Função loop():** Nesta função controlamos a placa, especificando qual deverá ser o comportamento do circuito.

Exemplo:

```
#define ledPin 9
```

```
void setup() {
```

```
    pinMode(ledPin, OUTPUT); //Define ledPin (pino 9) como saída
```

```
}
```

```
void loop() {
```

```
    digitalWrite(ledPin, HIGH); //Coloca ledPin em nível alto (5V)
```

```
}
```

## Manipulando entradas e Saídas Digitais

### DigitalRead()

Usamos para ler o valor de um pino digital. Ela verifica se o valor do pino está em LOW(0) ou HIGH(1).

Exemplo:

```
void setup(){  
  pinMode(button, OUTPUT); // configuramos o pino como entrada  
}  
  
void loop(){  
  digitalWrite(led); // leitura do pino para ver se ele está em nível lógico baixo ou alto  
}
```

### digitalWrite()

Usamos para escrever um valor em um pino digital, ou seja, com ela habilitamos o valor de saída do pino para LOW(desligado) ou HIGH(ligado).

Na prática, usamos essa função para ligar ou desligar componentes externos definidos como saída, como o Led.

Exemplo:

```
void setup() {  
  pinMode(ledPin, OUTPUT); // Define ledPin (pino 9) como saída  
}  
  
void loop() {  
  digitalWrite(ledPin, HIGH); // Coloca ledPin em nível alto (5V)
```

```
}
```

## Manipulando entradas e Saídas Analógicas

### AnalogRead()

Usamos para ler o valor de um pino analógico definido como entrada. Os valores de uma entrada analógica variam de 0 a 1023.

Usamos para ler valores de componentes que trabalham com sinais analógicos, como o potenciômetro e o Led LDR.

Exemplo de uso do analogRead. Ligação do led através de um led LDR(resistor que varia de valor conforme a incidência de luz).

```
/*
```

```
* Prática 01: Ligar um led com LDR
```

```
*/
```

```
#define portaLed 10 //Porta a ser utilizada para ligar o led
```

```
#define portaLDR A5 //Porta analógica utilizada pelo LDR
```

```
void setup() {
```

```
  pinMode(portaLed, OUTPUT); //Define a porta do Led como saída
```

```
  pinMode(portaLDR, INPUT); //Define a porta do LDR com entrada
```

```
}
```

```
void loop() {
```

```
  int estado = analogRead(portaLDR); //Variável estado recebe o valor fornecido pelo LDR
```

```
  // Caso o valor lido na porta analógica seja maior do que
```

```
  // 800, acende o LED
```

```
  // Ajuste o valor abaixo de acordo com o seu circuito
```

```
  if (estado > 800)
```

```
  {
```

```
    digitalWrite(portaLed, HIGH);
```

```
  }
```

```
  else //Caso contrário, apaga o led
```

```
  {
```

```
    digitalWrite(portaLed, LOW);
```

```
}
```

```
}
```

## **AnalogWrite()**

Usamos para escrever um valor(de 0 a 1023) em um pino analógico definido como saída.Podemos usá-la por exemplo para controlar o brilho de um led.

## **Comunicação com a placa**

Para a comunicação entre a placa e o computador usamos a comunicação serial.

Primeiramente precisamos definir a taxa de transferência dos dados. Tal taxa se refere a quantidade de bits que é transmitida por segundo.

Para isso, usamos a função `serial.begin()` dentro da função `setup()`, já que queremos inicializar a comunicação e vimos anteriormente que a função `setup` é responsável pela inicialização.

```
setup(){  
  
    Serial.begin(9600);  
  
}
```

## **Por padrão utilize sempre a taxa 9600.**

Após a inicialização podemos usar o monitor serial da IDE para, por exemplo, imprimir alguma mensagem ou os dados que estão sendo recebidos pela placa.

Para isso usamos a função `serial.print()` ou `serial.println()`, que acrescenta uma quebra de linha ao final de cada uma .

## **Exemplo:**

```
void setup(){  
  
    Serial.begin(9600);  
  
}
```

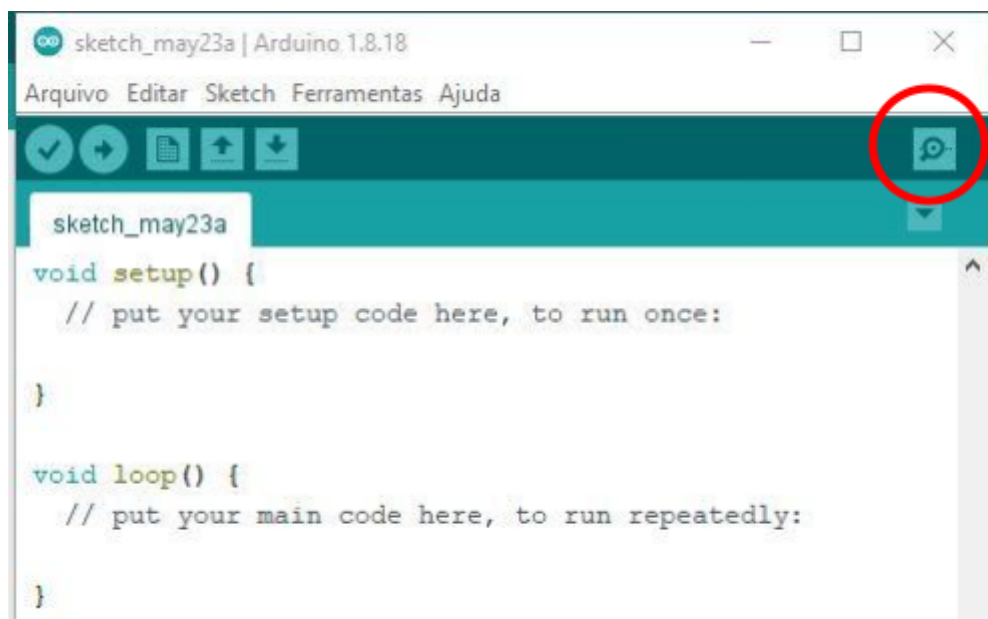
```
void loop(){  
  
    print("Minicurso de Arduino");  
  
}
```

Segue o código da prática do semáforo, do nosso último encontro, onde fizemos uso da serial para ver o valor do potenciômetro.

```
#define potenciometro A0  
#define red 12  
#define yellow 11  
#define green 10  
  
int valorPotenciometro = 0; //variavel criada para receber o valor do  
potenciometro.  
  
void setup() {  
    Serial.begin(9600);  
    pinMode(potenciometro, INPUT);  
    pinMode(red, OUTPUT);  
    pinMode(yellow, OUTPUT);  
    pinMode(green, OUTPUT);  
}  
  
void loop() {  
    valorPotenciometro = analogRead(potenciometro);  
    Serial.println(valorPotenciometro); //Imprimi o valor no monitor serial  
  
    if(valorPotenciometro < 285){  
        digitalWrite(green, HIGH);  
        digitalWrite(yellow, LOW);  
        digitalWrite(red, LOW);  
    }  
  
    else if((valorPotenciometro >= 285) && (valorPotenciometro < 600)){  
        digitalWrite(green, LOW);  
        digitalWrite(yellow, HIGH);  
        digitalWrite(red, LOW);  
    }  
    else{
```



```
digitalWrite(green, LOW);  
digitalWrite(yellow, LOW);  
digitalWrite(red, HIGH);  
}  
  
delay(20);  
  
}
```



O monitor serial é aberto clicando na lupa localizada no canto direito da tela, como apresentado na imagem acima. Lembrando que ele só é apresentado quando você estiver com o arduino conectado ao computador.

O arduino possui inúmeras funções. Neste material apresentamos as mais simples e que são sempre utilizadas. Abaixo, segue o link da documentação para você explorar e conhecer um pouco mais do que o arduino oferece e as várias possibilidades de projeto que se é possível criar utilizando o arduino.

[Site Oficial do Arduino](https://www.arduino.cc/en/Main)

[Conhecendo um pouco mais das funções do Arduino](https://www.arduino.cc/en/Reference)

