Aplicações Práticas

Projeto 1: Controle de Velocidade de um Motor DC com Potenciômetro

Projeto 1: Controle de Velocidade de um Motor DC com Potenciômetro

Objetivo: Controlar a velocidade de um motor DC usando PWM e um potenciômetro.

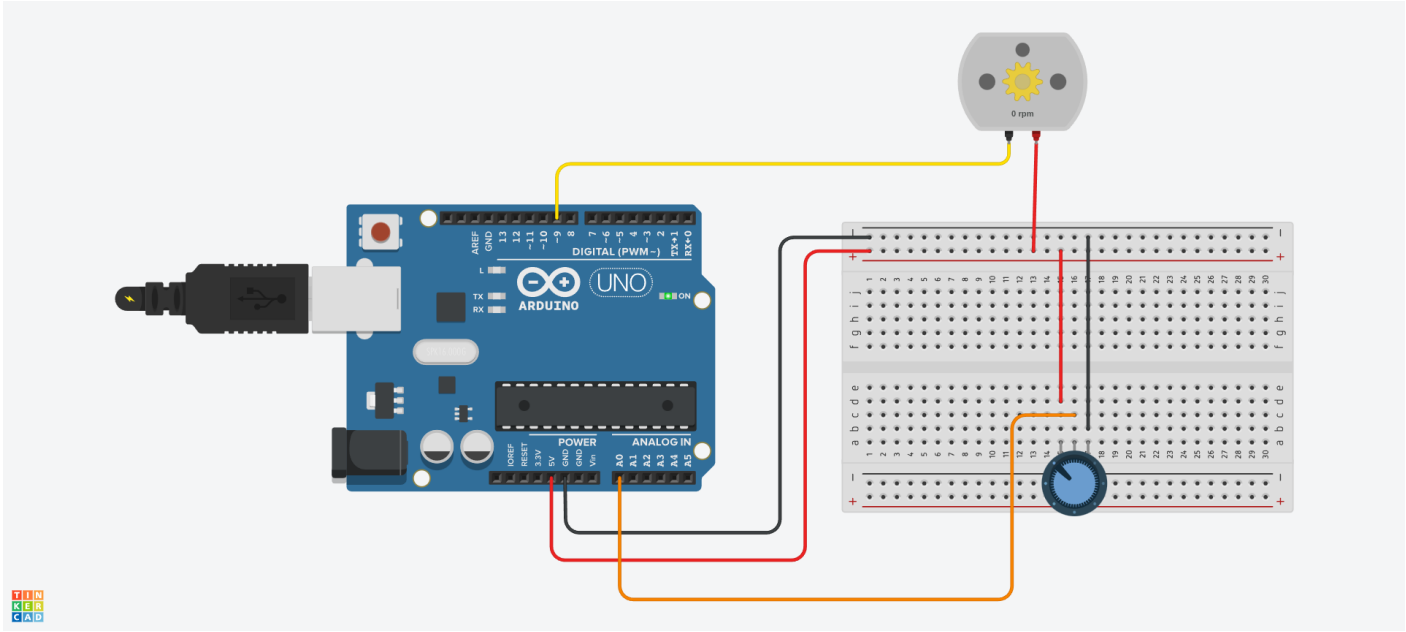
Componentes:

- Motor DC conectado ao pino PWM (com driver de motor ou transistor para proteção).
- Potenciômetro conectado ao pino A0.

Código:

```
int motorPin = 9;
int potPin = A0;
int speed = 0;

void setup() {
  pinMode(motorPin, OUTPUT);
}

void loop() {
  speed = analogRead(potPin);  // Lê o valor do potenciômetro
  speed = map(speed, 0, 1023, 0, 255);  // Converte o valor para 0-255
  analogWrite(motorPin, speed);  // Ajusta a velocidade do motor
}
```

Com esse projeto, ao girar o potenciômetro, o motor aumentará ou diminuirá a sua velocidade.


0 rpm
DIGITAL (PWM~)
ARDUINO
UNO
POWER
ANALOG IN

Projeto 2: CONTROLANDO SERVO MOTOR COM POTENCIÔMETRO

Projeto 2: CONTROLANDO SERVO MOTOR COM POTENCIÔMETRO

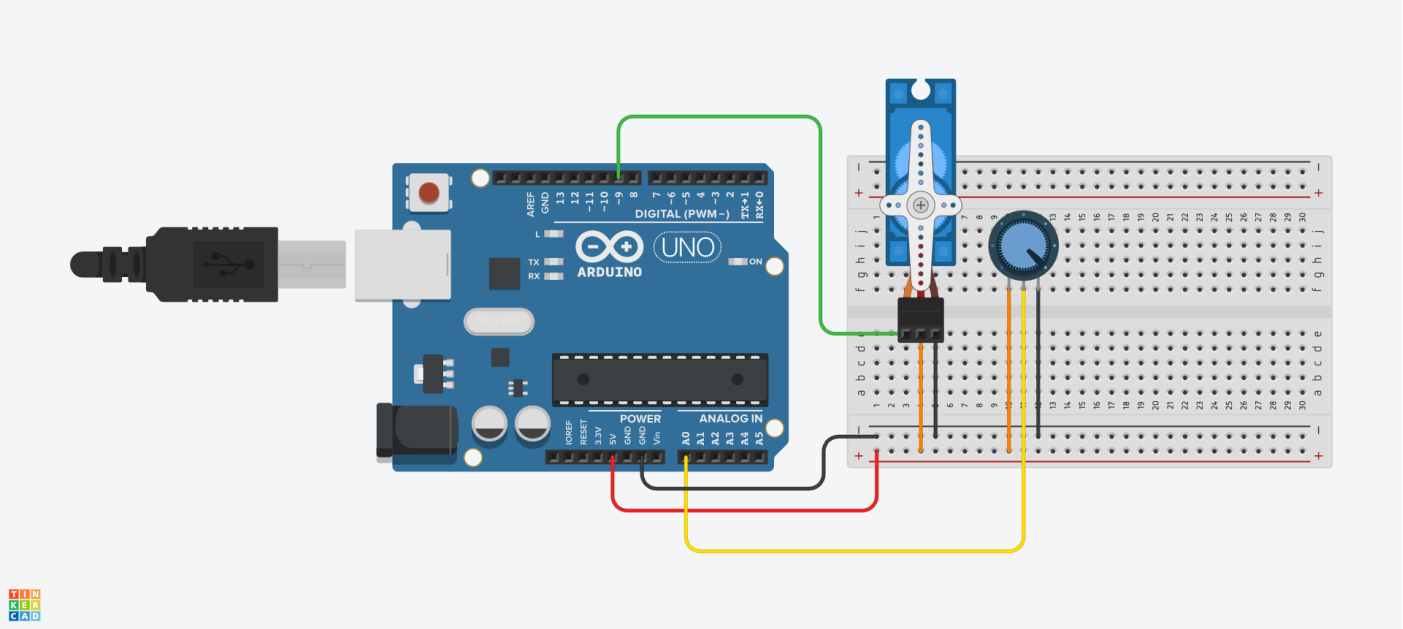
DESCRIÇÃO DO PROJETO: O projeto permitirá girar o servo motor em toda sua extensão permitida de meia volta, o soquete de trÊs pinos possui o GND tensão e sinal de pulso. A precisão do giro é verificada por meio do decodificador do servo motor, a qual efetua cálculos precisos.

Componentes:

- Servo Motor conectado ao pino 9 (ou outro pino PWM).
- Potenciômetro conectado ao pino A0.

```
#include <Servo.h>
Servo servo1; //cria um novo objeto servo
int potenciometroPino0 = 0; //conecta potenciometro na porta analogica 0
int valorDoPotenciometro = 0; //valor lido no pino 0

void setup (){

  servo1.attach(9); //conecta o objeto servo1 ao pino 9

}

void loop(){

  valorDoPotenciometro = analogRead(potenciometroPino0); // lê um valor analógico do potenciometro de  0 a
255
  valorDoPotenciometro = map(valorDoPotenciometro, 0, 1023, 0, 179);//mapeia o valor lido entre 0 a 1023 para
```

```
uma valor ente 0 e 180
  servo1.write(valorDoPotenciometro); // envia sinal para o servo posicionar
  delay(15); //aguarda movimento do servo


}
```



Nova PáginaProjeto Avançado: Dimmer de LED com Controle de Brilho Suave

Projeto Avançado: Dimmer de LED com Controle de Brilho Suave

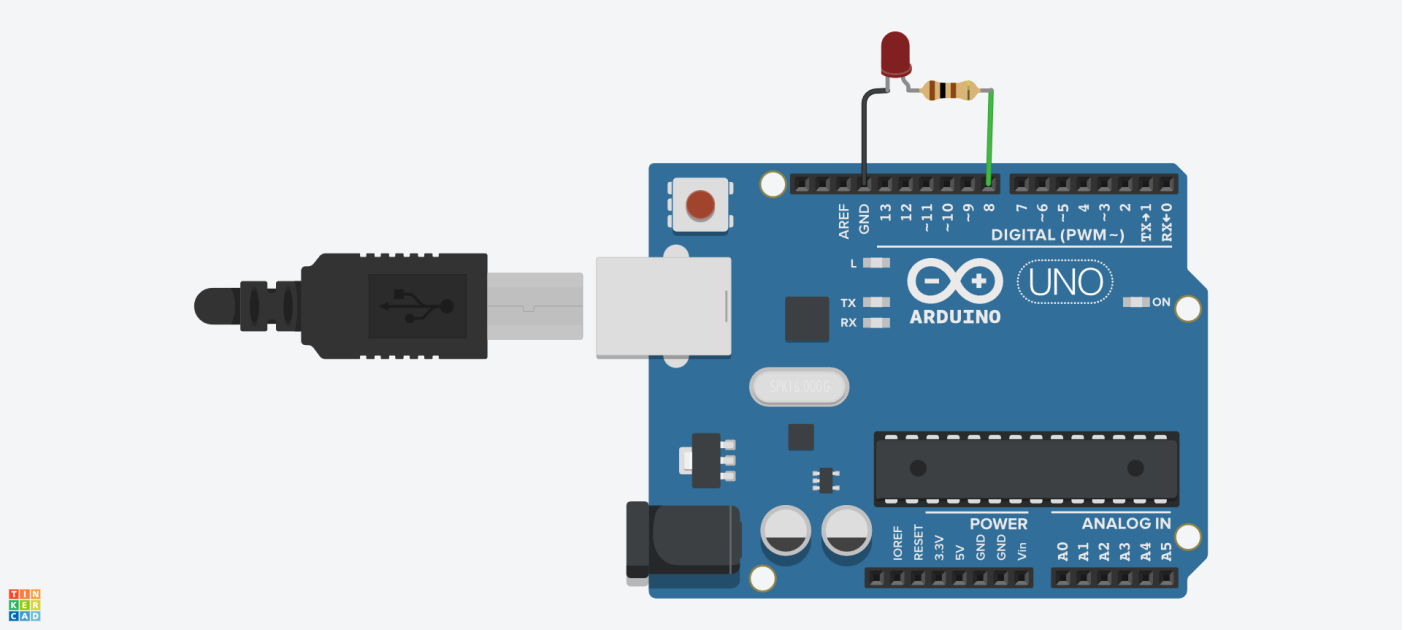
Objetivo: Criar um efeito de fade suave em um LED, onde o brilho aumenta e diminui continuamente.

Código:

```
int ledPin = 9;
int brightness = 0;
int fadeAmount = 5;

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

void loop() {
  analogWrite(ledPin, brightness);  // Ajusta o brilho do LED
  brightness = brightness + fadeAmount;  // Altera o brilho

  // Inverte a direção do fade quando atingir os extremos
  if (brightness <= 0 || brightness >= 255) {
    fadeAmount = -fadeAmount;
  }

  delay(30);  // Atraso para criar o efeito de fade suave
```

```
}
```

Nesse projeto, o brilho do LED aumenta e diminui suavemente, criando um efeito de "respiração" no LED.



Desafios

Desafios

Agora que você aprendeu a trabalhar com entradas e saídas analógicas, aqui estão alguns desafios para praticar:

1. **Desafio 1**: Crie um dimmer de LED controlado por um sensor de luz (LDR). O brilho do LED deve aumentar conforme a luminosidade diminui e vice-versa.
2. **Desafio 2**: Use um sensor de temperatura LM35 para controlar a velocidade de um motor. O motor deve girar mais rápido conforme a temperatura aumenta.

Conclusão

Conclusão

Neste capítulo, você aprendeu a trabalhar com sinais analógicos no Arduino, tanto para **ler entradas analógicas** de sensores quanto para **gerar saídas analógicas simuladas** com PWM. Você também aprendeu a usar a função `analogRead()` para ler valores de sensores e `analogWrite()` para controlar dispositivos com precisão, como motores e LEDs. Essas técnicas são fundamentais para o controle mais fino de projetos de eletrônica interativa, permitindo trabalhar com variações contínuas, em vez de apenas liga/desliga.

No próximo capítulo, exploraremos o controle de **motores e servomotores**, ampliando as possibilidades de automação e robótica em seus projetos Arduino.