Guia/Recapitulação da 1ª aula

Objetivo: Usar o Arduino para ler sensores de ambiente e controlar objetos eletrônicos.

Precisamos sempre lembrar que o conceito fundamental do Arduino está em ler as entradas elétricas (input) e com essas informações, controlar saídas de energia (output).

E para fazer esse trabalho usamos códigos de programação de computadores. E para isso, precisamos fazer o download, e instalação do programa do Arduino, no site www.arduino.cc

Iniciamos o programa, conectamos a placa do Arduino no computador, pela porta USB. Lembre-se de verificar se o programa do Arduino está reconhecendo a placa que está conectada. Abra o menu das placas em **Ferramentas >> Porta**. Em Windows os nomes das porta USB são COM1, COM2, COM3 e assim por diante. Geralmente o programa reconhece a porta certa, mas se surgir algum problema, que mencione que a placa não foi encontrada, ou que diga que a porta não está acessível, alterar a opção de porta USB pode resolver.

Ao iniciar um novo projeto, você verá que um código inicial é dado:

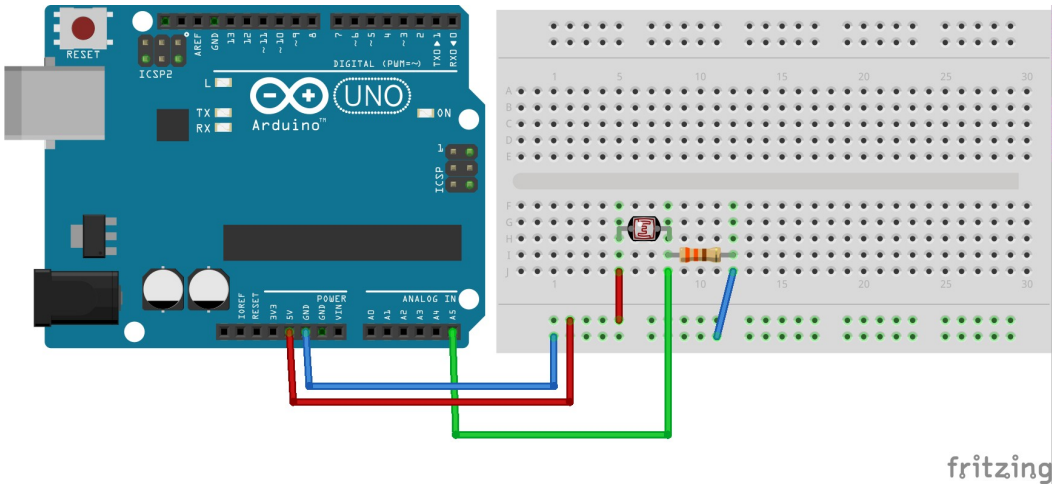
```
void setup() {
  // put your setup code here, to run once:

}

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

}
```

Relembrando:
Programação é um conjunto de instruções que damos ao computador. E quando um computador inicia um software, geralmente, precisamos iniciar configurações iniciais, e então inciar o programa de fato.

Esse modelo que código dado pelo programa do Arduino, já oferece os dois blocos principais de comandos. O setup, e o loop.
No setup, colocamos os códigos para configurar o programa. Esses códigos são lidos apenas uma vez pelo computador.
No loop, colocamos as instruções que queremos executar constantemente. Lembre-se do conceito fundamental do Arduino: ler entradas elétricas, para controlar saídas. E geralmente, criamos uma rotina de leituras, condições e saídas de energia, que são executadas o tempo todo, uma vez depois da outra, enquanto o Arduino tiver energia para rodar.

Nosso primeiro circuito
Objetivo: Ler a quantidade de luz ambiente, e controlar o liga/desliga de um led.
Para isso conectamos o circuito mostrado na sequência, e iniciamos o código.

Materiais:
- placa Arduino
- protoboard (placa de protótipos)
- LDR – leitor de luminosidade
- jumpers (os cabinhos com pinos nas pontas)
- resistor de 330 ohms



SEMPRE OBSERVE COM CUIDADO A LIGAÇÃO ELÉTRICA! Ligações erradas podem provocar curtos-circuitos e queimar os equipamentos, inclusive a placa do Arduino, e seu dinheiro vai pro buraco. Dica fundamental, nunca se esqueça de que a energia tem a carga positiva e negativa, e trocá-las pode ser muito ruim.

No código fizemos:

```
int portaLDR = A5; //Porta analógica utilizada pelo LDR

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

void loop()  {
  int valor = analogRead(portaLDR);  //Lê o valor fornecido pelo LDR

  Serial.println(valor);
}
```

Esse código super simples apenas faz o trabalho de ler a entrada da célula foto sensível e enviar para o computador, via porta Serial.
Após digitar o código, com o Arduino conectado, **precisamos clicar no botão de compilação/verificação do código**, para ver se não há erros, na digitação. Se tudo der certo, apenas uma mensagem em texto de cor branca aparecerá no rodapé do programa do Arduino. Em caso de erros, mensagens na cor laranja, amarela, ou vermelha aparecerão, e você terá de solucionar.

Com o código certo, **clicando no botão de carregar**, o software transfere o programa para o

Arduino, e inicia a execução.

Para visualizar os dados que a célula foto sensível está coletando, precisamos abrir o Monitor Serial. **Vá em Ferramentas >> Monitor Serial.**

Passo dois: controlar o LED
Depois incrementamos outros comandos para ligar, ou desligar o led da placa, dependendo da leitura do sensor que já montamos.
O sensor foto sensível, durante a nossa aula estava lendo valores entre 8 e 15. E jogando o flash do celular sobre ele, os valores chegavam até 100. Observar esses valores dos sensores é muito importante para pensar em como tomaremos as decisões, baseados nesses valores.
Então decidimos fazer que, se o valor lido pelo sensor ultrapassasse 50, então desligamos o led. Ou seja, se estiver claro, desliga a luz, no escuro, liga a luz.

Então o código:

```
void setup()
{
  Serial.begin(9600);
  pinMode(13, OUTPUT);
}

void loop()
{
  int valor = analogRead(A5);

  if( valor > 50 ){
    digitalWrite(13, LOW);
    }
  else{
    digitalWrite(13, HIGH);
    }

  Serial.println(estado);
}
```

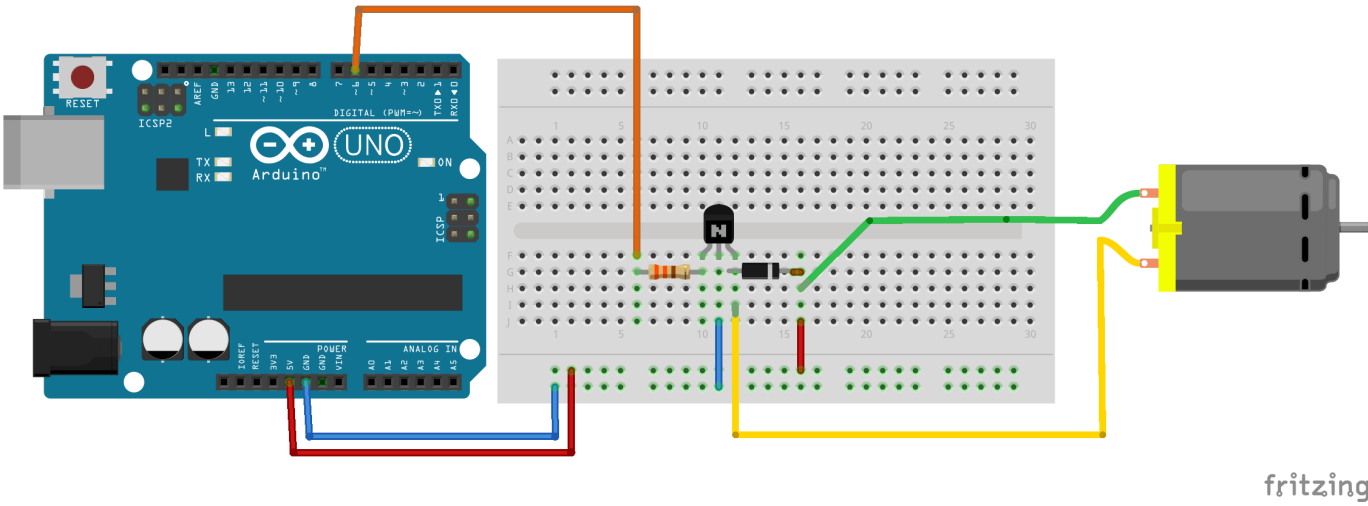
Ou seja, ao capturar o valor da leitura do sensor, agora criamos uma lógica onde, de acordo com a leitura do sensor, ativamos uma reação de outro equipamento.

O comandos chaves para o Arduino são justamente os grifados em laranja. São esses comandos que usamos para manipular aquele conceito fundamental da placa, ler entradas (`analogRead`) de uma das portas analógicas, e ligar, ou desligar saídas de energia (`digitalWrite`) das portas digitais.

Guia/Recapitulação da 2ª aula

Na segunda aula focamos em um circuito que será usado para ligar e desligar uma válvula selenoide. Para o exercício nós usamos um motor DC, que possui a mesma estrutura de circuito, e no código pudemos explorar o recurso de PWM (Pulso com modulação) do Arduino.

O circuito que precisamos montar é o que segue:



Basicamente temos um transistor que será na nossa chave liga/desliga, como centro do circuito. Usamos um transistor NPN(duas “pernas” negativas e uma positiva), e isso é bem importante, porque com um transistor PNP, a configuração seria diferente. O transistor permitirá que a energia circule apenas quando ele receber um pulso de energia.
Nesse nosso caso, a base do transistor é a primeira perna (esquerda para direita) que está conectada ao pino 6 do Arduino. A perna central do transistor conecta-se ao terra (ground - GND) do Arduino. A terceira perna possui uma conexão tripla, com o diodo e com o negativo do motor. A “saída” do diodo conecta-se à ligação de 5V que saí do Arduino e alimenta o motor.

No código temos:

```
int motorPin = 6;
int velocidade = 100;

void setup() {
     pinMode(motorPin, OUTPUT);
     Serial.begin(9600);
}

void loop() {
     //digitalWrite(motorPin, HIGH);
     analogWrite(motorPin, velocidade);

     delay(1000);

     //controle de potencia
     velocidade = velocidade+10;
     if( velocidade >= 240) velocidade = 30;
}
```

Nesse código, temos a escolha do pino onde o motor está ligado, e posteriormente sua configuração com saída de energia.

Na função loop, temos duas linhas de código com dois comandos, digitalWrite, e analogWrite. Como vimos na primeira aula o comando digitalWrite, é dado dizendo qual pino será alterado, e qual o estado da ligação HIGH, ou LOW, ou seja, ligado ou desligado.
O comando analogWrite, funciona com a tecnologia PWM, ele liga de desliga o pino, em uma frequência, que simula uma variação de potencia no equipamento. No nosso exercício, quando maior a potencia, mais rápido gira o motos. Isso só funciona nos pinos digitais do Arduino que tem o símbolo do til (~) antes no número. Veja na placa.

Após o chamado de uma dessas funções, o código pede uma pausa de 1 segundo para o Arduino: delay(1000)
Então temos um controle muito simples da velocidade do motor. A cada segundo ele aumenta a velocidade em 10, e se a velocidade ultrapassar o valor de 240, ele retorna ao 30. Com isso teremos um resultado onde o motor irá granar velocidade paulatinamente até um certo ponto, e então irá parar, ou reduzir muito sua velocidade.

Para o uso final, com a válvula selenoide, usaremos a função digitalWrite.

E assim, terminamos o nosso segundo encontro.