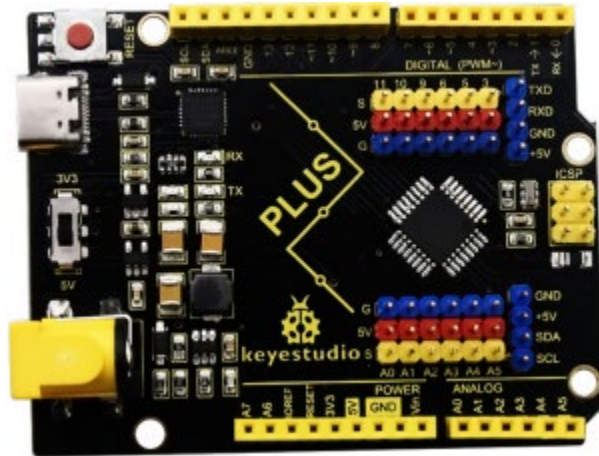


Educational home automation for Arduino

Placa de controlo Keystudio PLUS

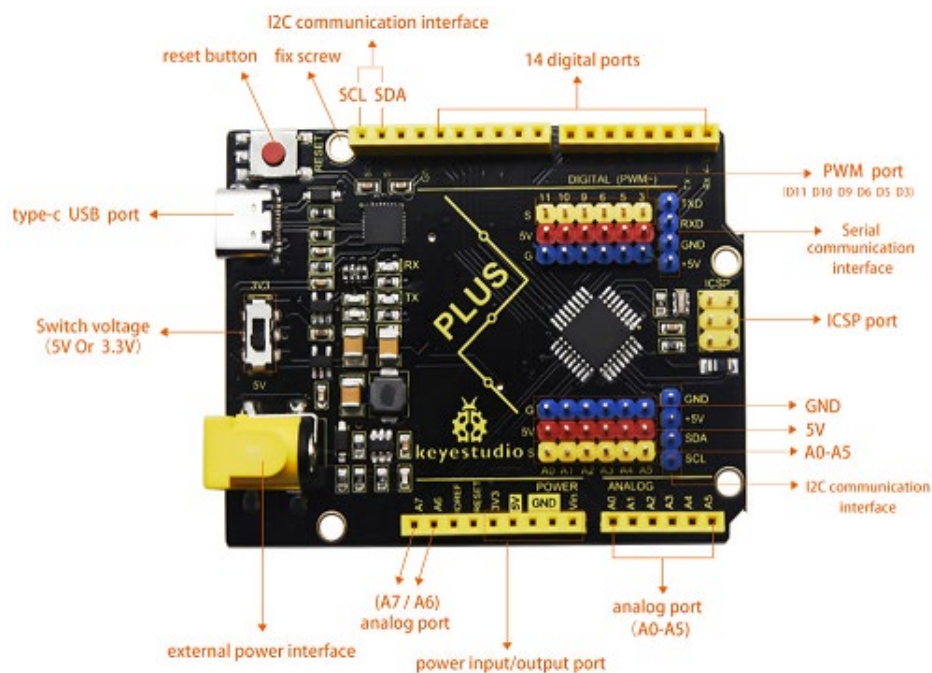
Descrição



A placa de controlo Keystudio PLUS é totalmente compatível com o ambiente de desenvolvimento Arduino IDE. Contém todas as funções da placa Arduino UNO.

Para além disso, foram introduzidos alguns melhoramentos que reforçam a sua função. É a melhor escolha para aprender a construir circuitos e a escrever código.

Vamos lá começar!

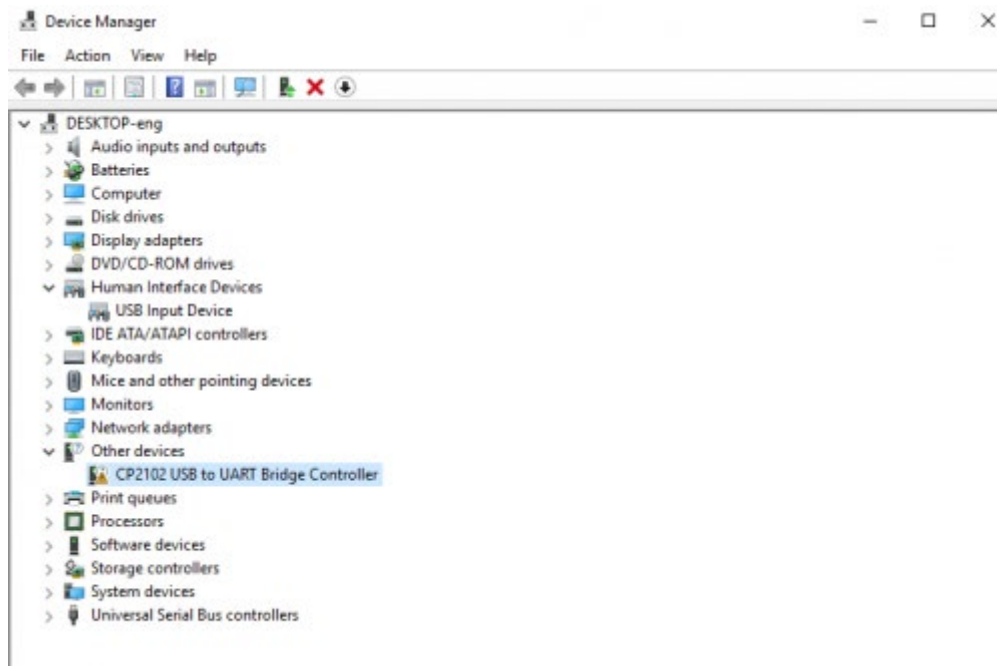


- Interface de comunicação em série: D0 é RX, D1 é TX
- Interface PWM (modulação de largura de pulso): D3 D5 D6 D9 D10 D11
- Interface de interrupção externa: D2 (interrupção 0) e D3 (interrupção 1)
- Interface de comunicação SPI: D10 é SS, D11 é MOSI, D12 é MISO, D13 é SCK
- Porta de comunicação IIC: A4 é SDA, A5 é SCL

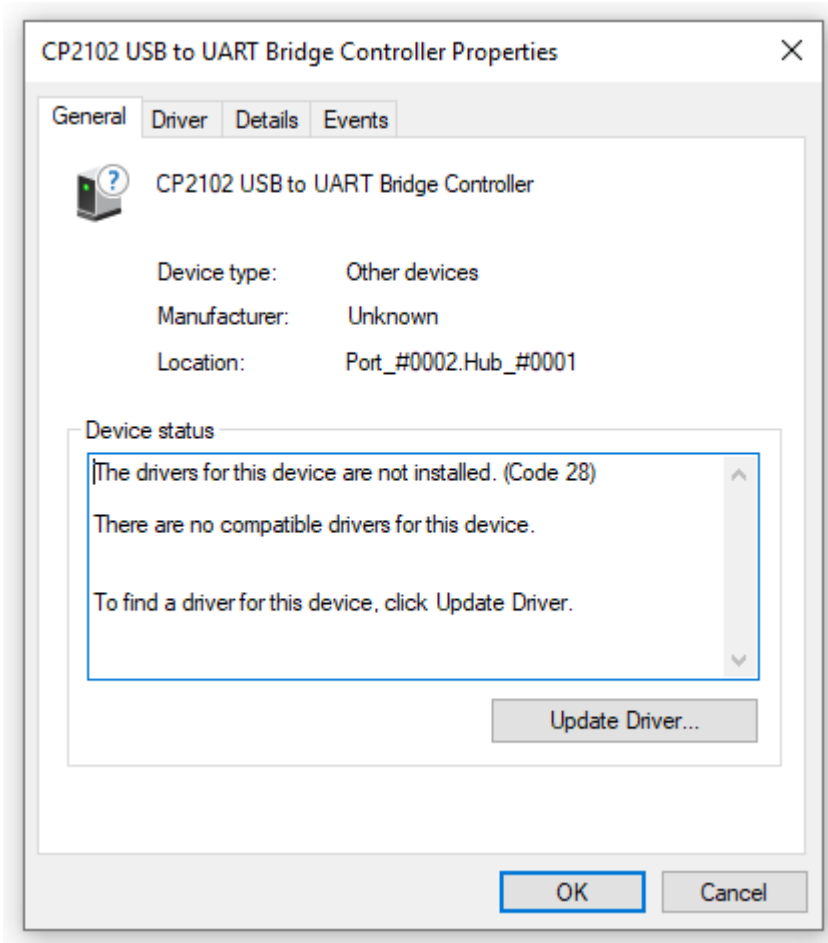
Installing driver

Vamos instalar o driver da placa de controlo keystudio PLUS. O chip USB-TTL da placa PLUS adopta o chip de série CP2102. O programa de driver deste chip está incluído na versão Arduino 1.8 e superior, o que é conveniente. Ligar a porta USB da placa, o computador pode reconhecer o hardware e instalar automaticamente o driver do CP2102.

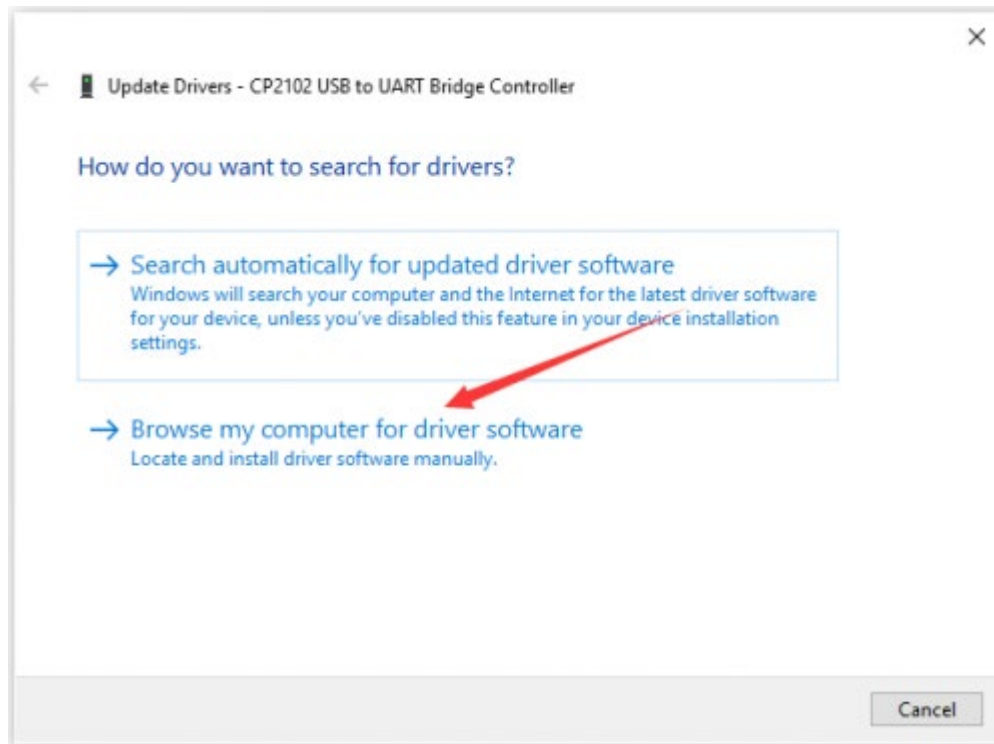
Se a instalação não for bem sucedida, ou se pretender instalar manualmente, abra o gestor de dispositivos do computador. Clique com o botão direito do rato em Computador----- Propriedades----- Gestor de dispositivos.




Existe um ponto de exclamação amarelo na página, o que implica que a instalação do controlador do CP2102 não foi bem sucedida. Em seguida, fazemos duplo clique no hardware e actualizamos o controlador.



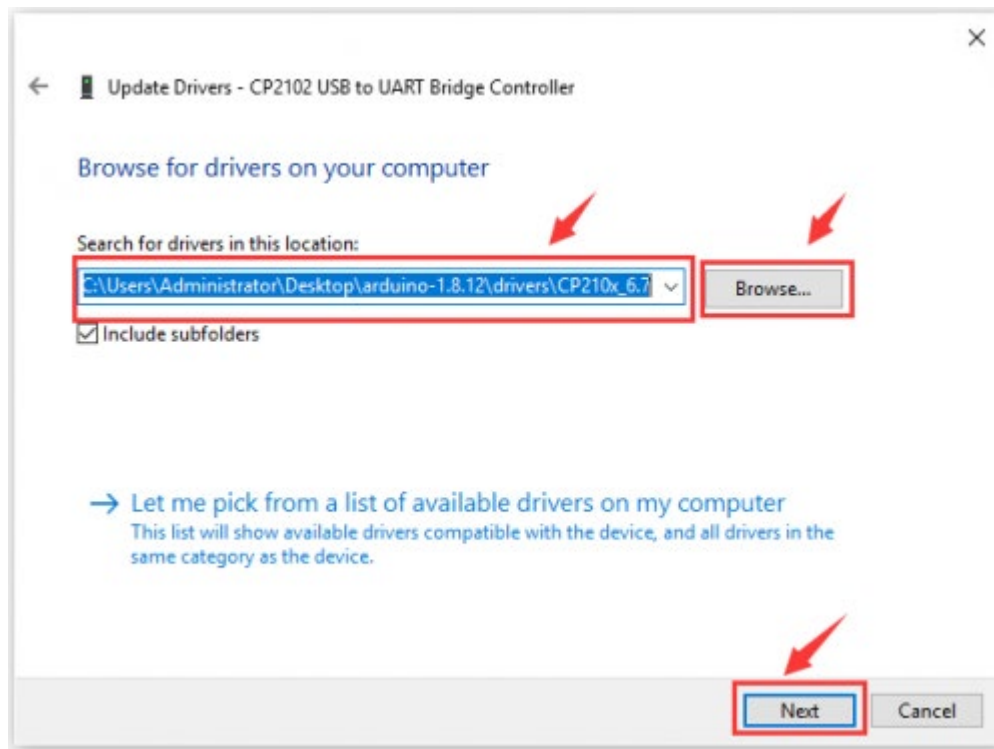
Clique em "OK" para aceder à página seguinte, clique em "procurar no meu computador software de controladores atualizado", descubra o software ARDUINO instalado ou transferido. Como mostrado abaixo:



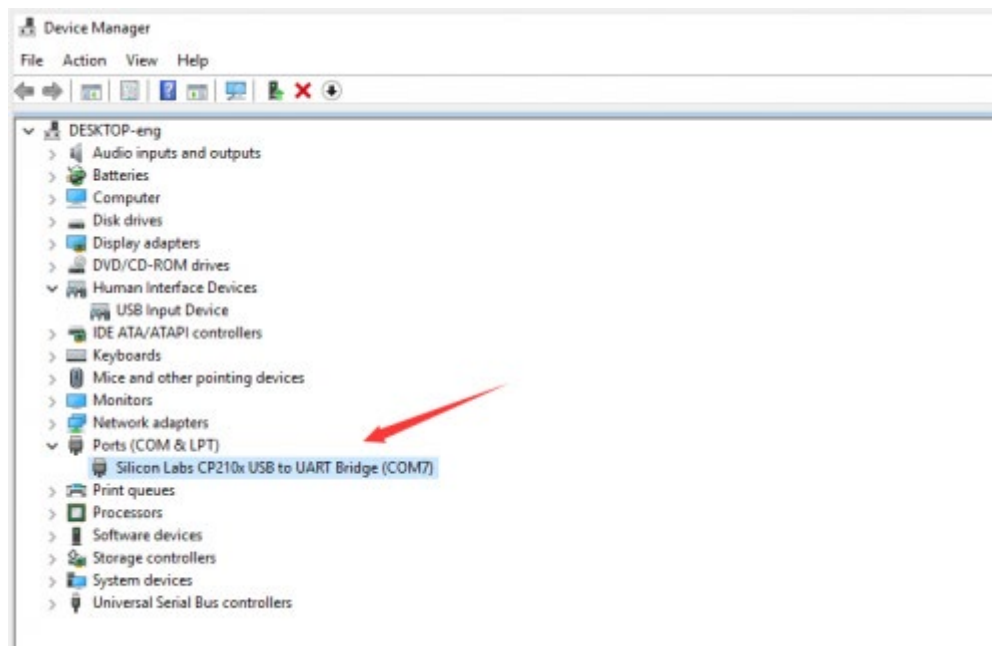
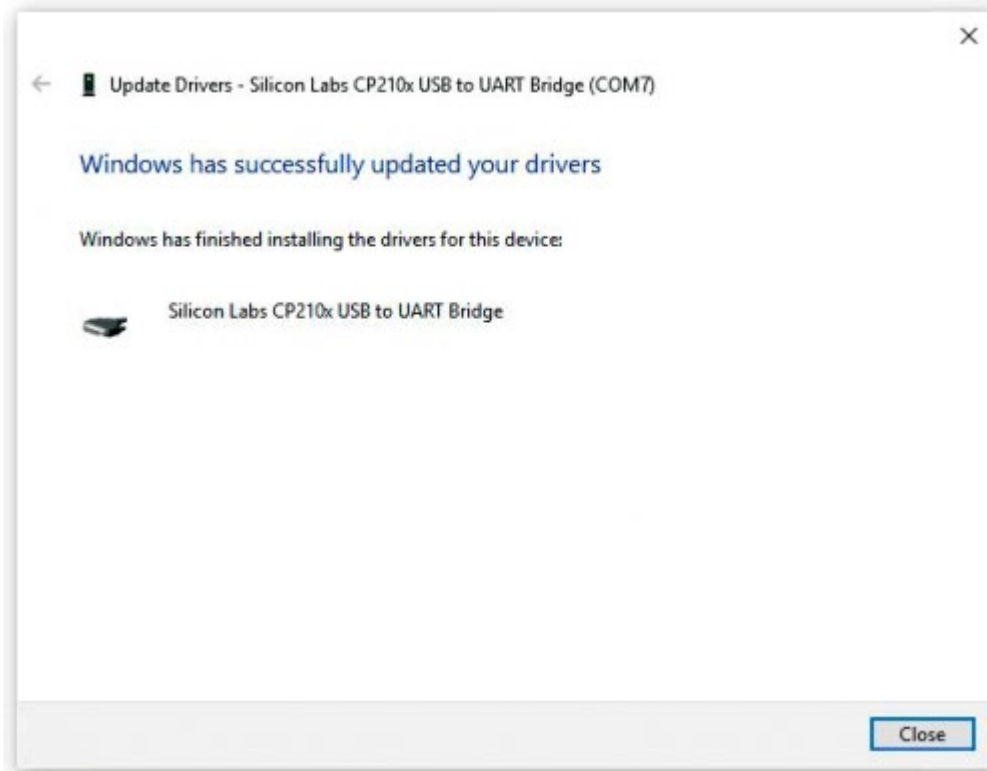
Existe uma pasta DRIVERS no pacote de software Arduino instalado

( arduino-1.8.12) , abra a pasta do driver e poderá ver o driver dos chips da série CP210X.

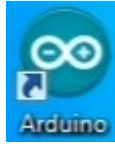
Clique em "Browse" (Procurar), depois descubra a pasta do controlador, ou introduza "driver" para procurar na caixa retangular, depois clique em "next" (seguinte), o controlador será instalado com êxito. (Eu coloco a pasta do software Arduino no ambiente de trabalho, podem seguir o meu exemplo)



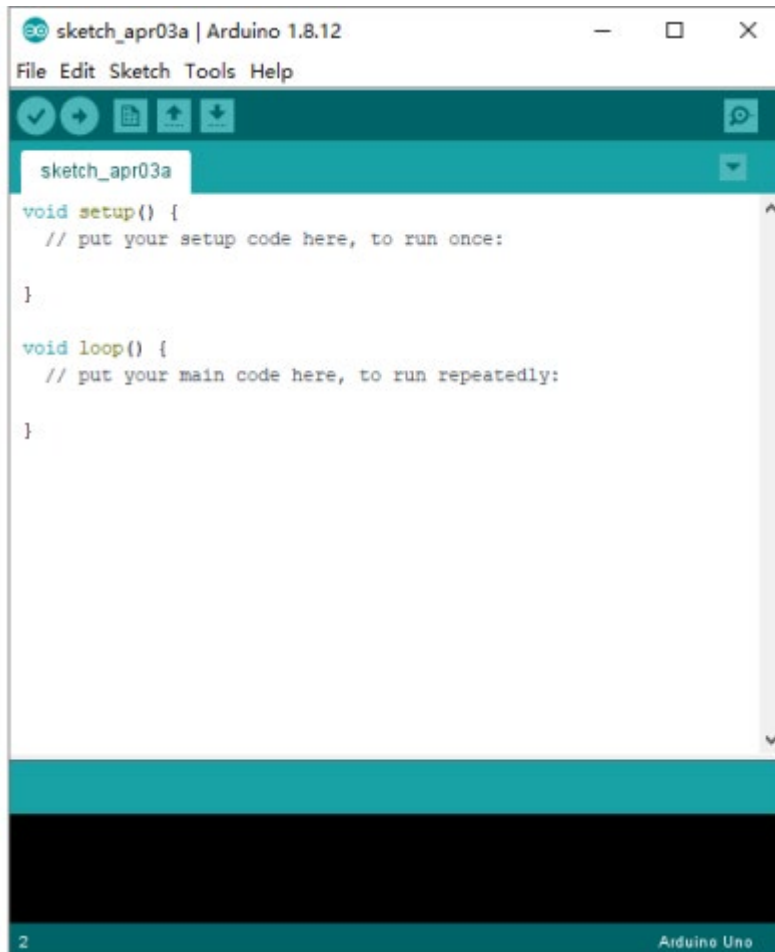
Abrir o gestor de dispositivos, veremos que o ponto de exclamação amarelo desaparece. O controlador do CP2102 foi instalado com êxito.



Configuração do IDE Arduino

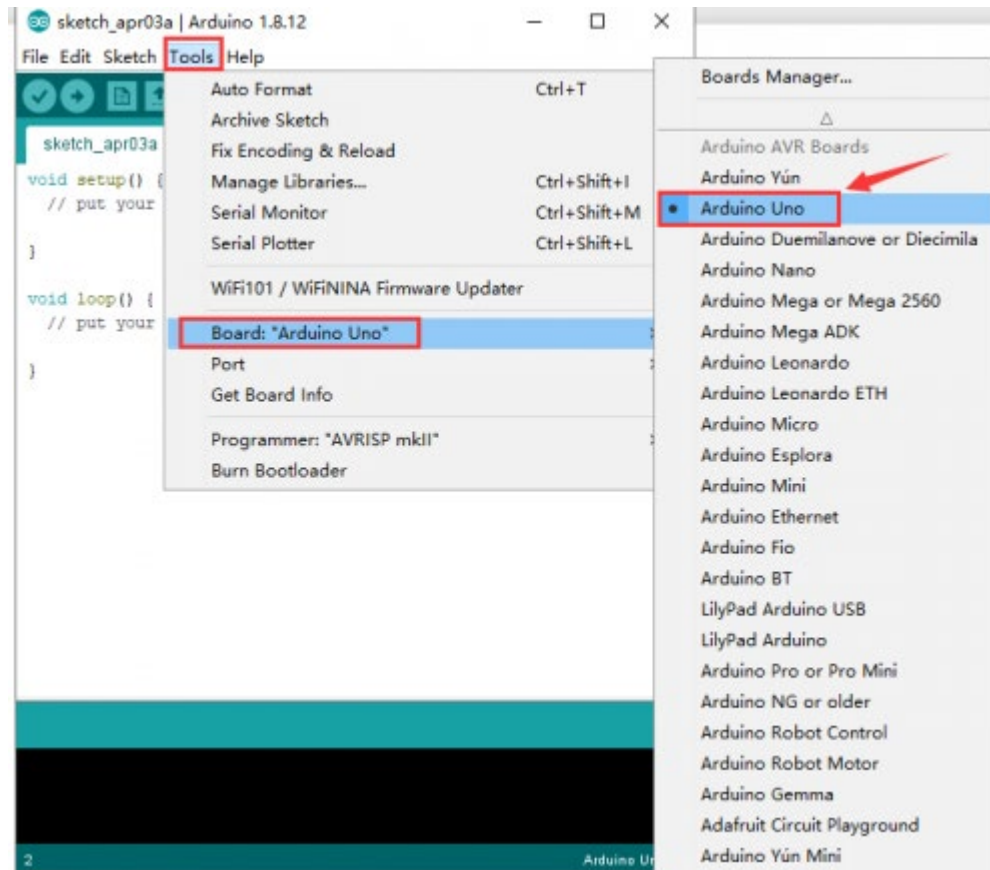


Clicar ícone, abrir o Arduino IDE.

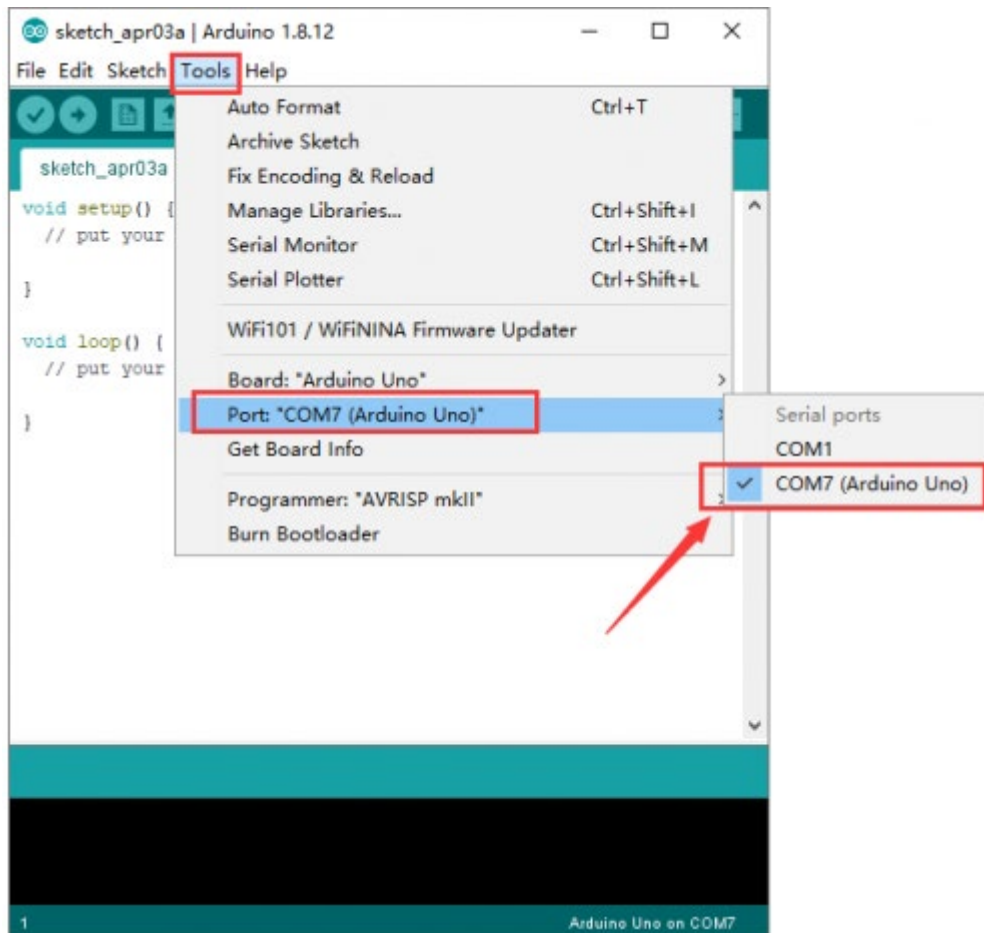


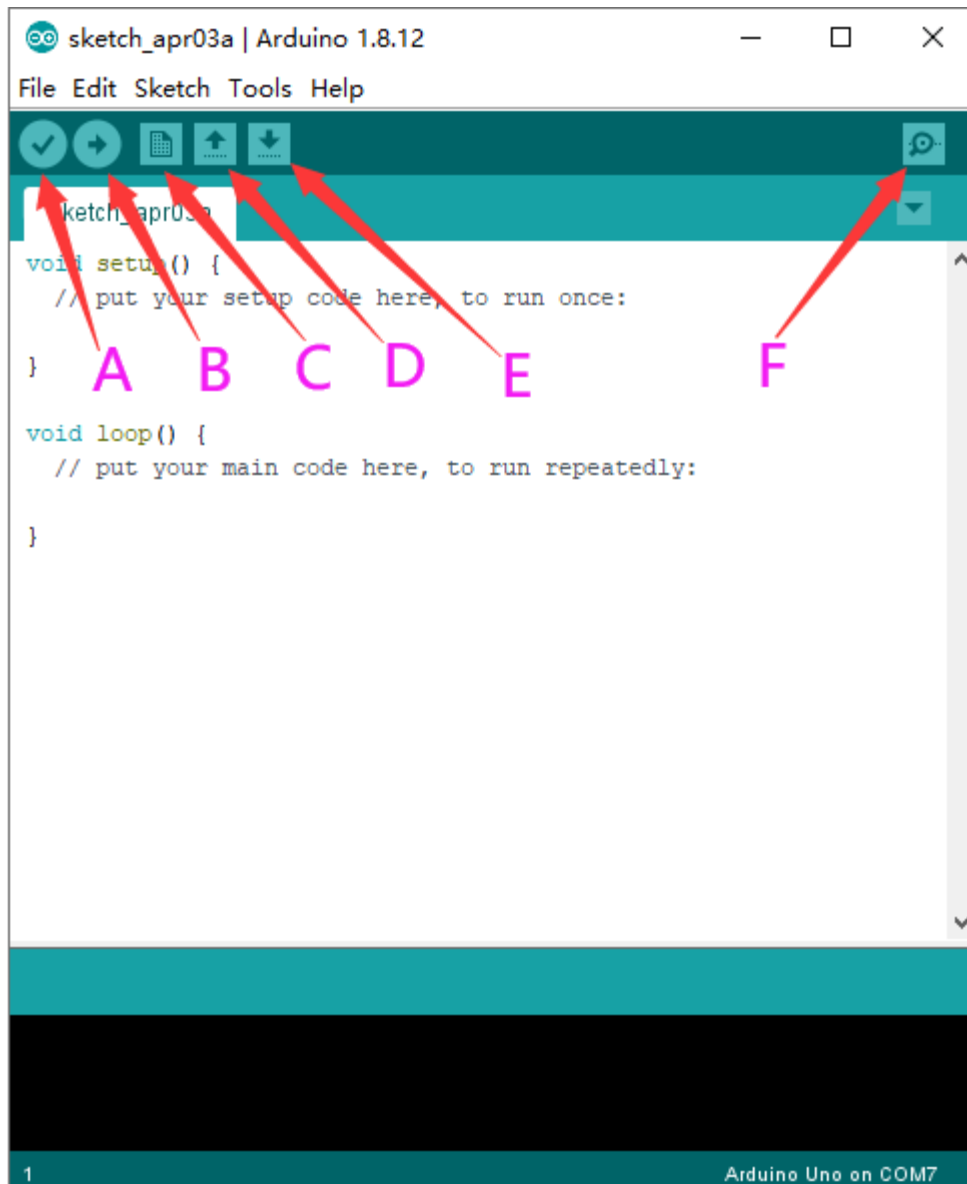
Para evitar erros ao carregar o programa para a placa, é necessário seleccionar a placa Arduino correcta que corresponda à placa ligada ao computador. Em seguida, volte ao software Arduino e clique em Tools→Board

(Ferramentas→Placa) e seleccione a placa. (como mostrado abaixo)



Em seguida, seleccione a porta COM correcta (pode ver a porta COM correspondente depois de o controlador ter sido instalado com êxito)





Antes de carregar o programa para a placa, vamos demonstrar a função de cada símbolo na barra de ferramentas do IDE Arduino.

A- Utilizado para verificar se existe algum erro de compilação ou não.

B- Utilizado para carregar o sketch para a sua placa Arduino.

C- Utilizado para criar uma janela de atalho de um novo sketch.

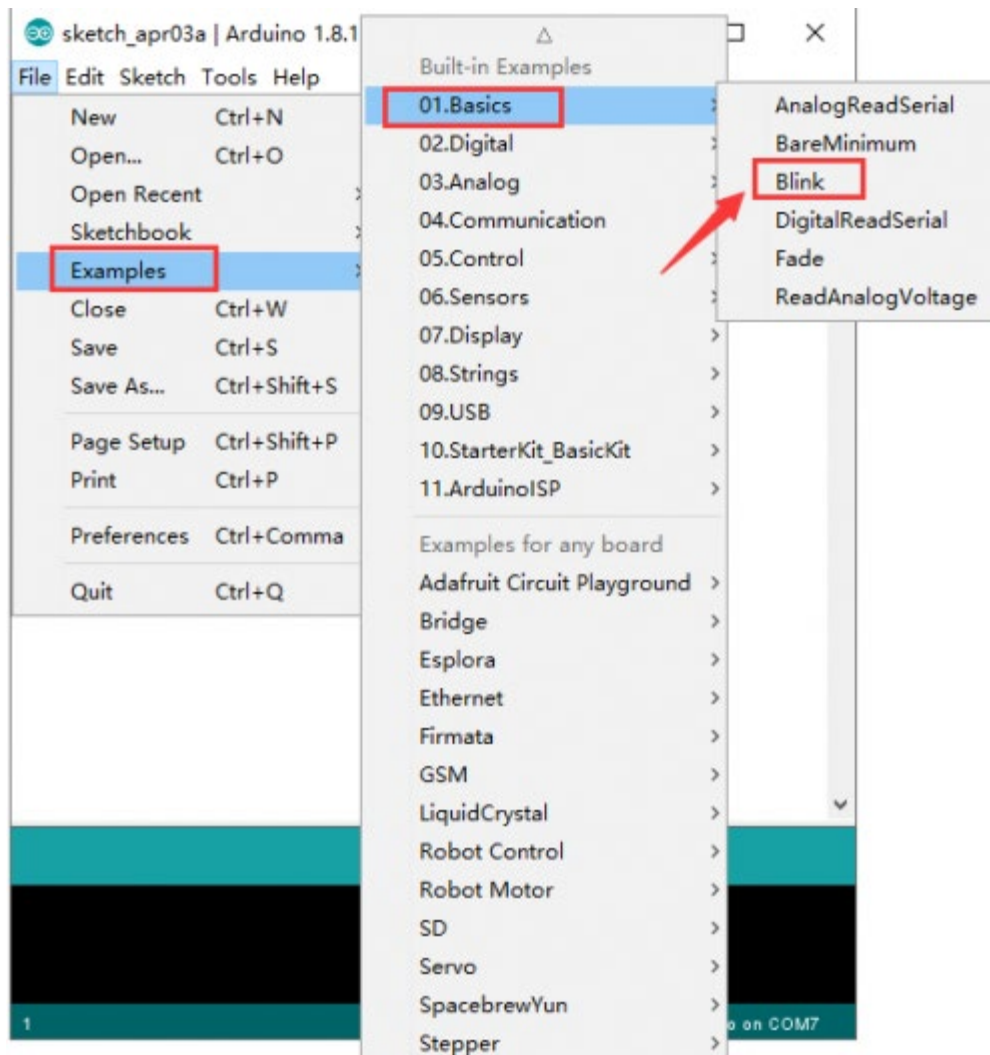
D- Utiliza-se para abrir diretamente um sketch de exemplo.

E- Utilizado para guardar o sketch.

F- Utilizado para enviar os dados de série recebidos da placa para o monitor de série.

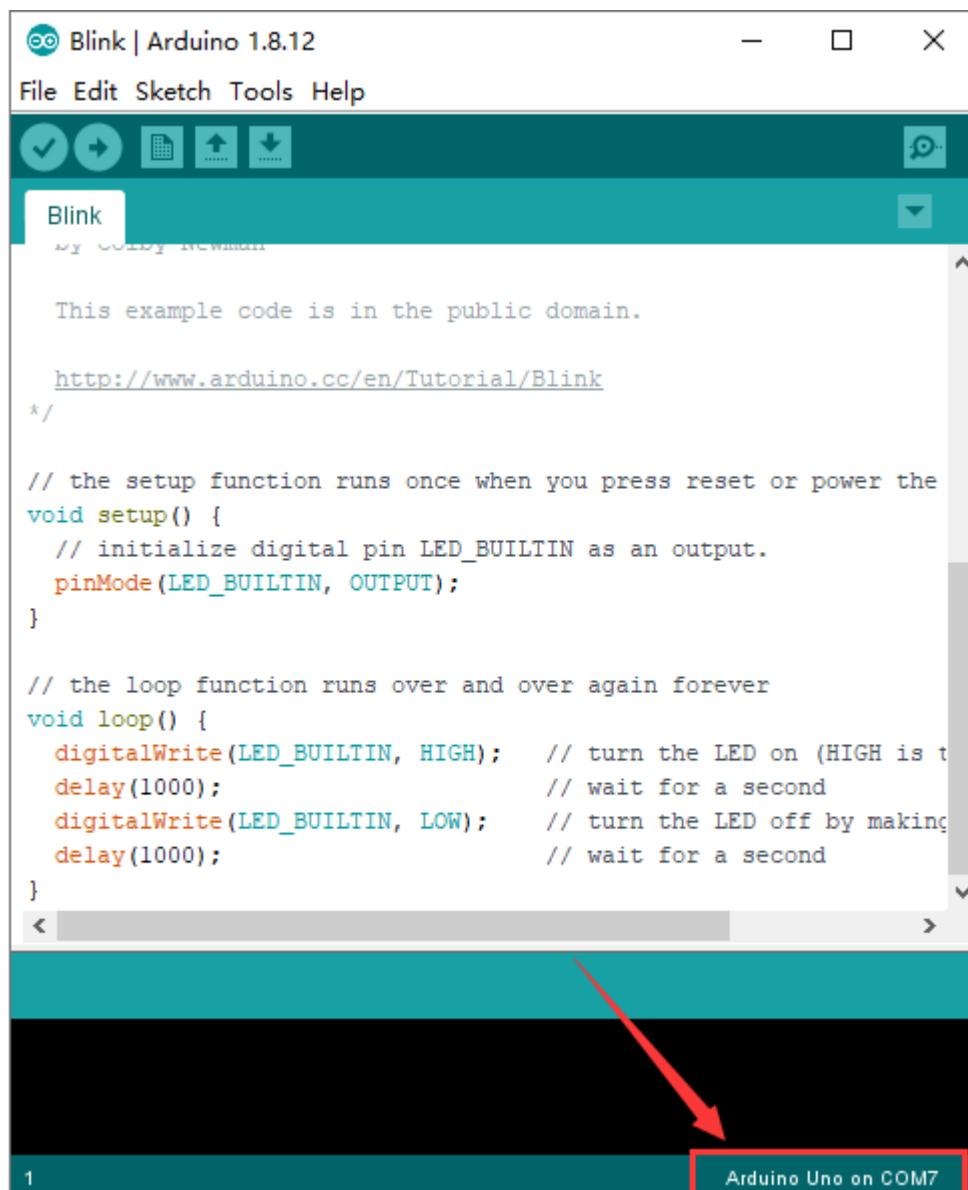
Iniciar o seu primeiro programa

Abrir o ficheiro para seleccionar Exemplo, escolher BLINK a partir de BASIC, como se mostra abaixo:






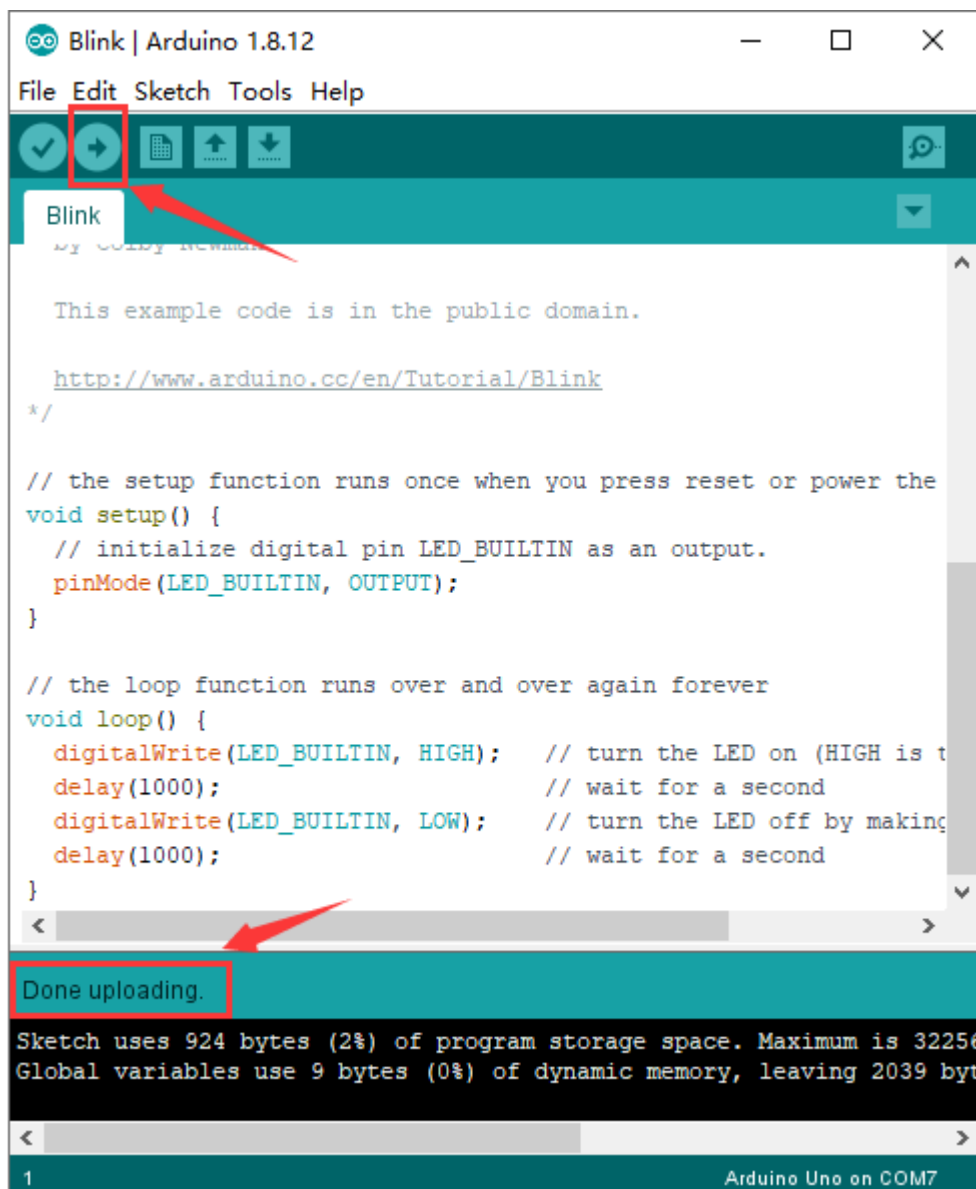
Definir a placa e a porta COM, a placa correspondente e a porta COM são apresentadas no canto inferior direito do IDE.



Clicar em  para iniciar a compilação do programa, verificar os erros.



Clicar em  para carregar o programa, carregar com êxito.



Carrega o programa com sucesso, o LED de bordo acende-se durante 1s, apaga-se durante 1s. Parabéns, terminou o primeiro programa.

Como adicionar uma biblioteca?

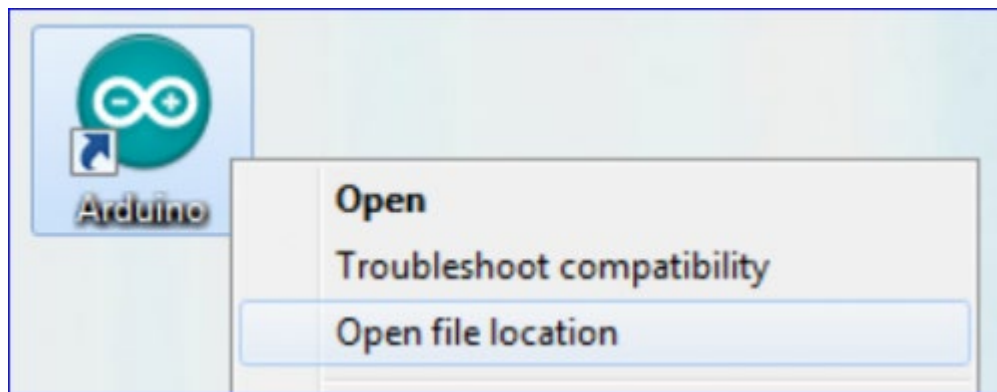
O que são bibliotecas? As bibliotecas são uma coleção de código que facilita a ligação a um sensor, ecrã, módulo, etc. Por exemplo, a biblioteca LiquidCrystal integrada ajuda a falar com ecrãs LCD. Existem centenas de bibliotecas adicionais

disponíveis na Internet para descarregar. As bibliotecas incorporadas e algumas destas bibliotecas adicionais estão listadas na referência.

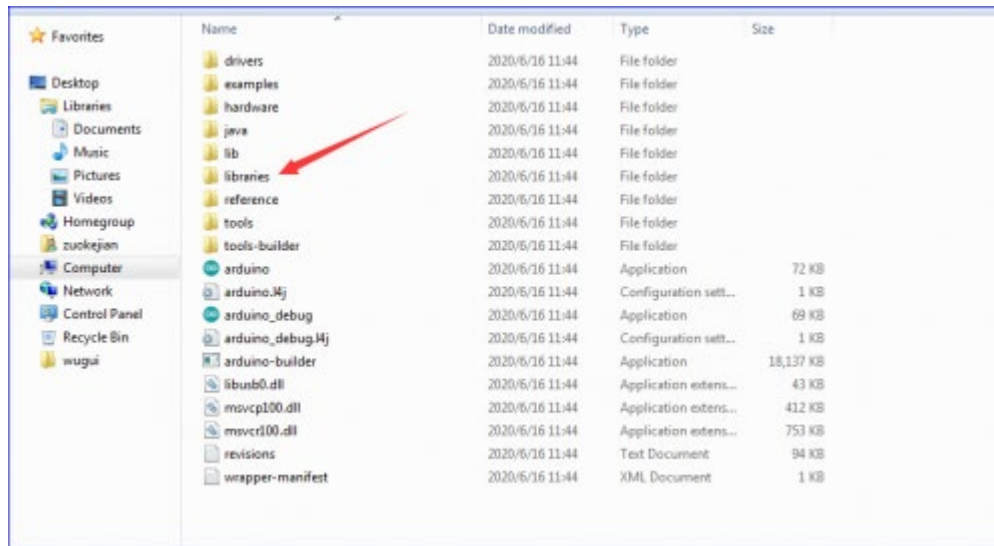
Aqui vamos apresentar a forma mais simples de adicionar bibliotecas.

Passo 1: Depois de baixar bem o Arduino IDE, você pode clicar com o botão direito do mouse no ícone do Arduino IDE.

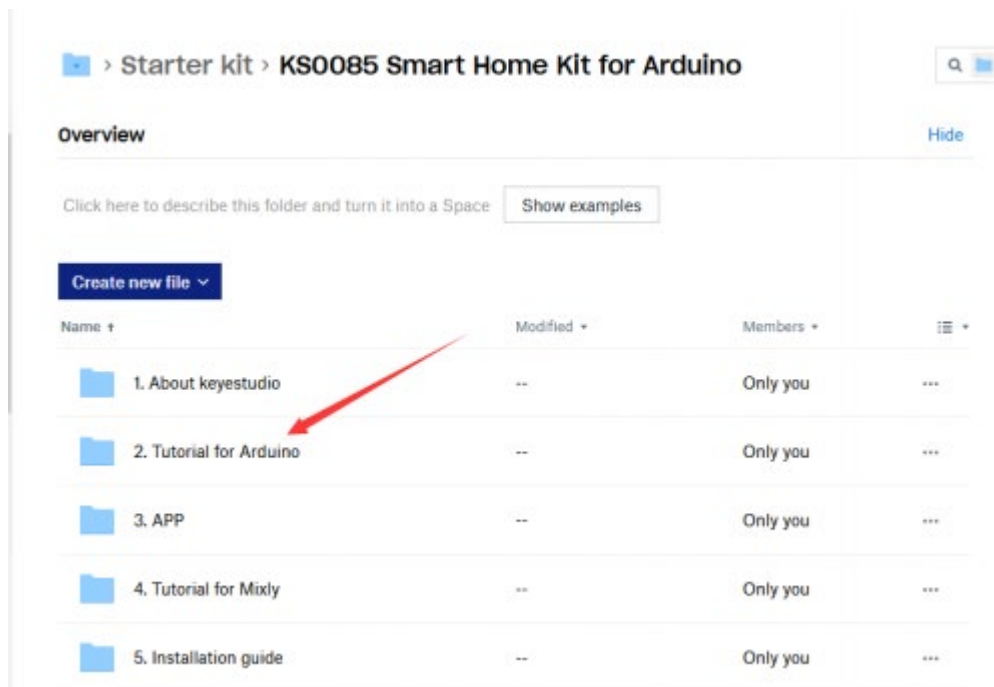
Encontre a opção "Abrir local do arquivo" mostrada abaixo:



Passo 2: Introduza-o para descobrir a pasta libraries, que é o ficheiro de biblioteca do Arduino.



Passo 3: Em seguida, para descobrir as "bibliotecas" da casa inteligente (visto no link: <https://fs.keyestudio.com/KS0085>), como indicado abaixo:



> 2. Tutorial for Arduino



Overview

Hide

Click here to describe this folder and turn it into a Space

Show examples

Create new file

Name	Modified	Members	
1. Arduino Software	--	Only you	...
2. Getting Started With Arduino	--	Only you	...
3. Tutorial	--	Only you	...
4. Arduino code	--	Only you	...
5. Arduino library files	--	Only you	...

> 2. Tutorial for Arduino > 5. Arduino library files

Overview

Click here to describe this folder and turn it into a Space

Show examples

Create new file

<input checked="" type="checkbox"/>	Name	Modified	Members
<input checked="" type="checkbox"/>	LiquidCrystal_I2C	--	Only <input type="button" value="Share"/>
<input checked="" type="checkbox"/>	Servo	--	Only <input type="button" value="Share"/>



Só precisa de replicar e colar na pasta de bibliotecas do Arduino IDE. The library of home smart is successfully installed, as shown below:

Adafuit_Circuit_Playground	2020/2/13 10:32	文件夹
Bridge	2020/2/13 10:32	文件夹
Esplora	2020/2/13 10:32	文件夹
Ethernet	2020/2/13 10:32	文件夹
Firmata	2020/2/13 10:32	文件夹
GSM	2020/2/13 10:32	文件夹
IRremote	2020/8/18 14:15	文件夹
Keyboard	2020/2/13 10:32	文件夹
LiquidCrystal	2020/2/13 10:32	文件夹
LiquidCrystal_I2C	2020/8/26 9:38	文件夹
Mouse	2020/2/13 10:32	文件夹
Robot_Control	2020/2/13 10:32	文件夹
Robot_Motor	2020/2/13 10:32	文件夹
RobotIRremote	2020/2/13 10:32	文件夹
SD	2020/2/13 10:32	文件夹
Servo	2020/2/13 10:32	文件夹
SpacebrewYun	2020/2/13 10:32	文件夹
SR04	2020/8/17 15:51	文件夹
Stepper	2020/2/13 10:32	文件夹
Temboo	2020/2/13 10:32	文件夹
TFT	2020/2/13 10:32	文件夹
WiFi	2020/2/13 10:32	文件夹
.keep	2020/2/13 10:32	KEEP 文件

Projectos



Muito bem, vamos diretamente aos nossos projectos. Neste kit, estão incluídos 14 sensores e módulos. Vamos começar com o sensor simples para o fazer conhecer profundamente a casa inteligente. No entanto, se for um entusiasta com

conhecimentos de Arduino. Pode saltar estes passos, montar o kit de casa inteligente diretamente (há um vídeo de montagem na pasta)

Nota: Neste curso, a interface de cada sensor/módulo assinalada com (G,-, GND) indica o pólo negativo, G está ligado a G ou - ou GND da blindagem do sensor ou da placa de controlo; "V" implica o pólo positivo que está ligado a V ou VCC ou 5V.

Projeto 1: LED Pisca

Descrição:








Na lição anterior, instalámos o controlador da placa de desenvolvimento keystudio V4.0 e começámos com projectos simples. Nesta lição, vamos executar o projeto "Arduion blinks LED", que é a prática básica para quem está a começar. Fornecemos um código de teste para controlar o LED e realizar o efeito de piscar. No código, é possível definir uma cena intermitente distinta, alterando o tempo de ativação e desativação da iluminação. Ligar o GND e o VCC, o LED acende-se quando o sinal S é de nível alto, pelo contrário, o LED apaga-se quando o sinal S é de nível baixo.

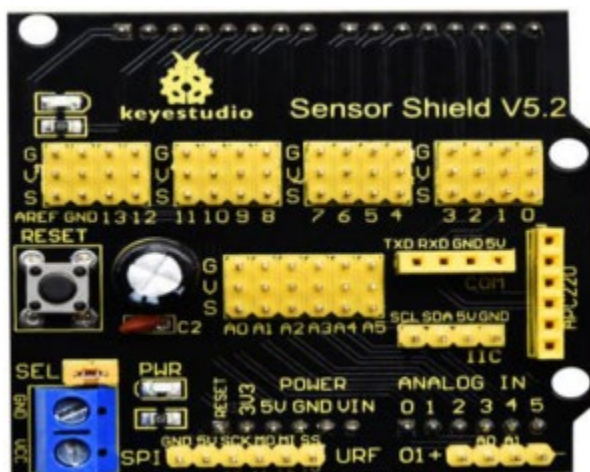
Specifications:

- Interface de controlo: porta digital
- Tensão de funcionamento: DC 3.3-5V
- Passo do pino: 2.54mm
- Cor do ecrã LED: branco
- Tamanho: 30 * 20mm
- Peso: 3g

Equipment:

PLUS control board*1	Sensor shield*1	White LED module *1	USB cable*1	3pin F-F Dupont line*1
				

Sensor shield



Normalmente, combinamos a placa de controlo Arduino com outros sensores, módulos e sensores múltiplos, o que é difícil de ligar. Por outro lado, este escudo

de sensor cobre este problema, basta empilhar a placa de controlo keyestudio

PLUS quando a utilizar.

Esta blindagem pode ser inserida diretamente em sensores 3PIN, e também permite a utilização de portas de comunicação comuns, tais como comunicação em série, comunicação IIC e comunicação SPI. Para além disso, o shield inclui um botão de reset e 2 luzes de sinalização.

Pinos Descrição

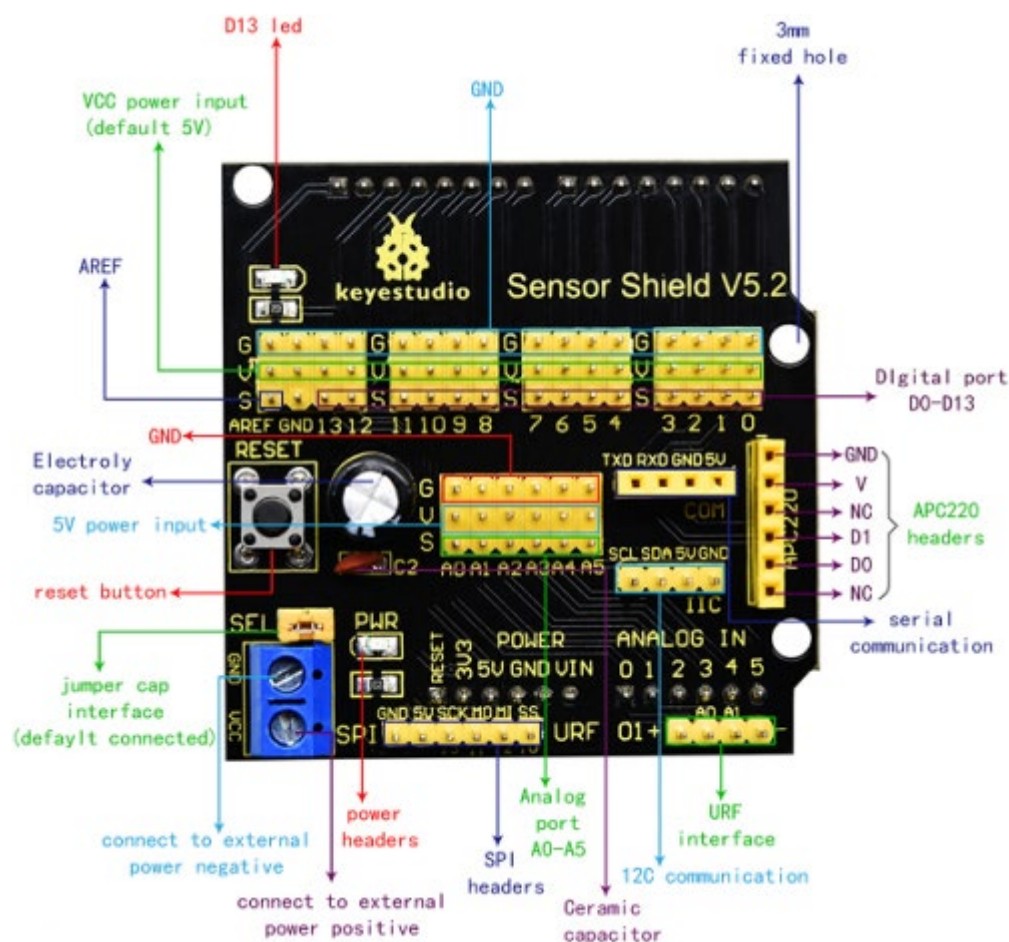
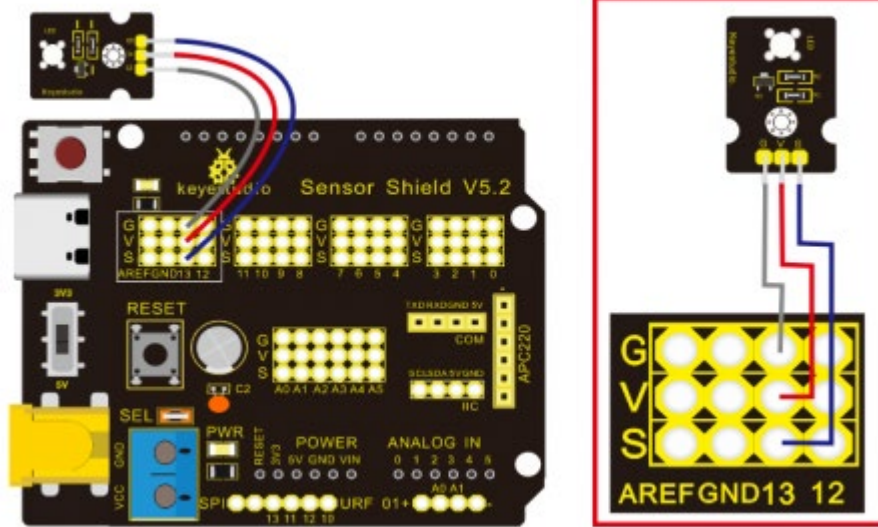


Diagrama de ligação:

Ao lado do fio, ligar o módulo LED ao D13 da proteção.



Nota: G, V e S do módulo de LED branco estão ligados a G, V e 13 da placa de expansão.

Código de teste:

```
/*
```

```
Keystudio smart home Kit for Arduino
```

```
Project 1
```

```
Blink
```

```
http://www.keystudio.com
```

```
*/
```

```
void setup() {
```

```
    // initialize digital pin 13 as an output.
```

```
pinMode(13, OUTPUT);

}

// the loop function runs over and over again forever

void loop() {

    digitalWrite(13, HIGH); // turn the LED on (HIGH is the voltage level)

    delay(1000);           // wait for a second

    digitalWrite(13, LOW); // turn the LED off by making the voltage LOW

    delay(1000);           // wait for a second

}//
```

Resultado do teste: :

Carregar o código de teste com êxito, o LED branco começa a piscar, acende-se durante 1000 ms e apaga-se durante 1000 ms, alternadamente.

Explicação do código:

O código parece longo e confuso, mas a maior parte é comentada. A gramática do Arduino é baseada em C.

Os comentários têm geralmente duas formas de expressão:

`/**/` : suitable for long paragraph comments

`//` : suitable for mono line comments

So the code contains the many vital information, such as the author, the issued agreement, etc.

Most people omit comments, starter should develop a good habit of looking through code. Firstly, check comments. They contain the provided information and do help you understand Código de teste quickly. Secondly, form the habit of writing comments

```
// the setup function runs once when you press reset or power the board

void setup() {

    // initialize digital pin 13 as an output.

    pinMode(13, OUTPUT);

}
```

According to comments, we will find that author define the D13 pin mode as digital output in setup() function. Setup() is the basic function of Arduino. It will execute once in the running of program, usually as definition pin, define and ensure the variables.

```
// the loop function runs over and over again forever

void loop() {

    digitalWrite(13, HIGH); // turn the LED on (HIGH is the voltage level)
```

```
delay(1000);           // wait for a second

digitalWrite(13, LOW); // turn the LED off by making the voltage LOW

delay(1000);           // wait for a second

}
```

Loop() is the necessary function of Arduino, it can run and loop all the time after

“setup()” executes once

In the loop()function, author uses

```
digitalWrite(13, HIGH); // turn the LED on (HIGH is the voltage level)
```

digitalWrite(): set the output voltage of pin to high or low level. We make D13

output high level, then the LED lights on.

```
delay(1000); // wait for a second
```

Delay function is used for delaying time, 1000ms is 1s, unit is ms

```
digitalWrite(13, LOW); // turn the LED off by making the voltage LOW
```

Similarly, we make D13 output low level, LED will turn off.

```
delay(1000); // wait for a second
```

Delay for 1s, light on LED--keep on 1s--light off LED--stay on 1s, iterate the

process. LED flashes with 1-second interval. What if you want to make LED flash

rapidly? You only need to modify the value of delay block. Reducing the delay

value implies that the time you wait is shorter, that is, flashing rapidly. Conversely, you could make LED flash slowly.

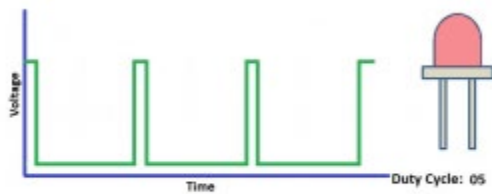


Projeto 2: Respirar a luz



Breathing light

Descrição



Na lição anterior, controlámos o LED para ligar e desligar e fazê-lo piscar. Neste projeto, vamos controlar o brilho do LED através de PWM para simular o efeito de respiração. Da mesma forma, pode alterar o comprimento do passo e o tempo de atraso no código, de modo a demonstrar diferentes efeitos de respiração.

O PWM é um meio de controlar a saída analógica através de meios digitais. O controlo digital é utilizado para gerar ondas quadradas com diferentes ciclos de funcionamento (um sinal que alterna constantemente entre níveis altos e baixos) para controlar a saída analógica. Em geral, a tensão de entrada da porta é de 0V e 5V. E se for necessário 3V? Ou se for necessário alternar entre 1V, 3V e 3,5V? Não podemos mudar a resistência constantemente. Para esta situação, precisamos de controlar por PWM.

Para a saída de tensão da porta digital do Arduino, existem apenas LOW e HIGH, que correspondem à saída de tensão de 0V e 5V. Pode-se definir LOW como 0 e HIGH como 1, e deixar o Arduino emitir quinhentos sinais 0 ou 1 em 1 segundo. Se a saída de quinhentos sinais for 1, isso é 5V; se todos forem 0, isso é 0V. Se a saída for 01010101010101 desta forma, então a porta de saída é de 2,5 V, o que é

como ver um filme. Os filmes que vemos não são completamente contínuos. Na verdade, ele produz 25 imagens por segundo. Neste caso, o ser humano não consegue distinguir, nem o PWM. Se quisermos uma tensão diferente, temos de controlar o rácio de 0 e 1. Quanto mais sinais 0,1 forem emitidos por unidade de tempo, maior será a precisão do controlo.

Equipamento:

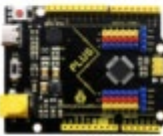




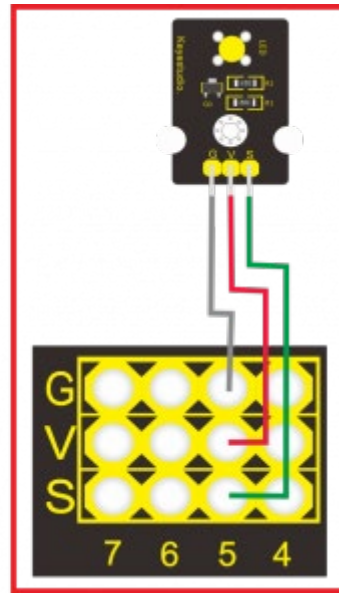
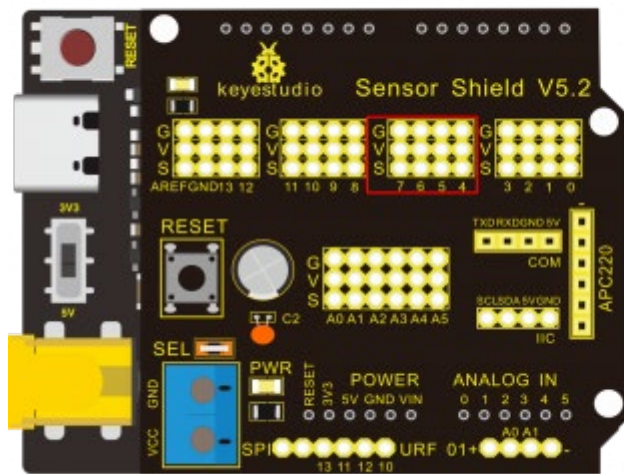
PLUS control board*1	Sensor shield*1	Yellow LED module*1	USB cable*1	3pin F-F Dupont line*1
				

Diagrama de ligação:



Nota: na blindagem do sensor, os pinos G, V e S do módulo de LED amarelo estão ligados a G, V e 5.

Código de teste:

```
/*
```

```
Keystudio smart home Kit for Arduino
```

```
Project 2
```

```
PWM
```

```
http://www.keystudio.com
```

```
*/
```

```
int ledPin = 5; // Define the LED pin at D5
```

```
void setup () {
```

```
    pinMode (ledPin, OUTPUT); // initialize ledpin as an output.
```



```
}  
  
void loop () {  
  
for (int value = 0; value<255; value = value + 1) {  
  
    analogWrite (ledPin, value); // LED lights gradually light up  
  
    delay (5); // delay 5MS  
  
}  
  
for (int value = 255; value>0; value = value-1) {  
  
    analogWrite (ledPin, value); // LED gradually goes out  
  
    delay (5); // delay 5MS  
  
}  
  
}  
  
//
```

Resultado do teste:

Carregar o código de teste com êxito, o LED torna-se gradualmente mais brilhante e depois mais escuro, como a respiração humana, em vez de se acender e apagar imediatamente



Análise de Código

Quando precisamos de repetir algumas instruções, temos de utilizar a instrução "for". O formato da instrução "for" é o seguinte

```
    ①          ② condition is true  ④  
for (cycle initialization; cycle condition; cycle adjustment statement) {  
    ③ loop body statement; ←  
}
```

"for" cyclic sequence:

Round 1: 1 → 2 → 3 → 4

Round 2: 2 → 3 → 4

... Until number 2 is not established, "for" loop is over,

After knowing this order, go back to code:

```
for (int value = 0; value < 255; value=value+1){  
    ...  
}  
  
for (int value = 255; value >0; value=value-1){  
    ...  
}
```

As duas declarações "for" fazem aumentar o valor de 0 para 255, depois reduzem de 255 para 0 e depois aumentam para 255,...infinite loop

Existe uma nova função na instrução "for" ----- analogWrite()

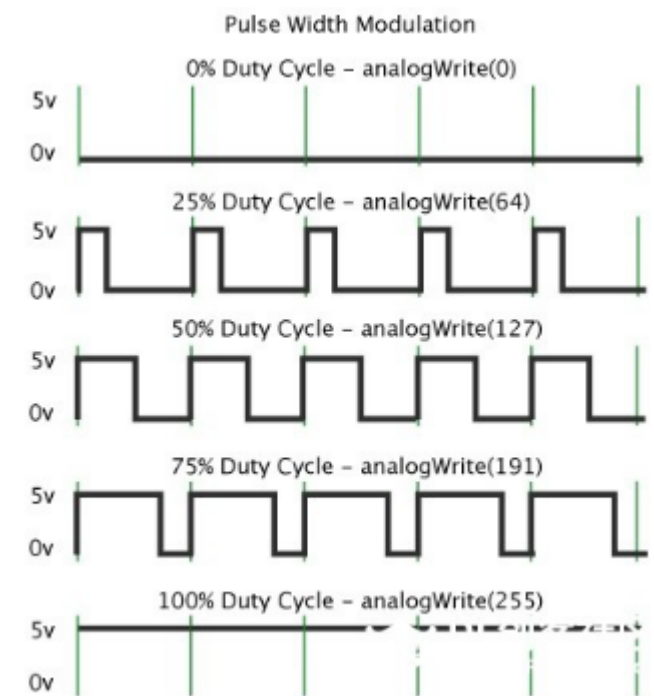
Sabemos que a porta digital só tem dois estados 0 e 1. Então, como enviar um valor analógico para um valor digital? Aqui, precisamos desta função, observa a placa Arduino e encontrarás 6 pinos com "~". Eles são diferentes dos outros pinos e podem enviar sinais PWM.

O formato da função é o seguinte:

analogWrite(pin,value) analogWrite() é usado para escrever um valor analógico de 0~255 para a porta PWM, por isso o valor está no intervalo de 0~255, atenção que só escreves os pinos digitais com função PWM, como o pino 3, 5, 6, 9, 10, 11.

O PWM é uma tecnologia que permite obter quantidades analógicas através de um método digital. O controlo digital forma uma onda quadrada, e o sinal de onda quadrada tem apenas dois estados de comutação (ou seja, níveis altos ou baixos dos nossos pinos digitais). Ao controlar a relação entre a duração de ligado e desligado, é possível simular uma tensão que varia de 0 a 5V. O tempo decorrido (academicamente referido como nível alto) é chamado de largura de pulso, por isso PWM também é chamado de modulação de largura de pulso.

Através das cinco ondas quadradas seguintes, vamos saber mais sobre a PWM.



Na figura acima, a linha verde representa um período e o valor de `analogWrite()` corresponde a uma percentagem que também é designada por ciclo de funcionamento. O ciclo de trabalho implica que a duração do nível alto é dividida pela duração do nível baixo num ciclo. De cima para baixo, o ciclo de trabalho da

primeira onda quadrada é 0% e o seu valor correspondente é 0. O brilho do LED é o mais baixo, ou seja, desliga-se. Quanto mais tempo durar o nível alto, mais brilhante será o LED. Portanto, o último ciclo de trabalho é 100%, o que corresponde a 255, o LED é mais brilhante. 25% significa mais escuro. O PWM é utilizado principalmente para ajustar a luminosidade do LED ou a velocidade de rotação do motor.

Projeto 3: Buzzer passivo

Descrição



Há muitos trabalhos interactivos realizados com o Arduino. O mais comum é o ecrã de som e luz. Usamos sempre LED para fazer experiências. Para esta lição, projectamos um circuito para emitir som. Os componentes sonoros universais são a campainha e as buzinas. A campainha é mais fácil de utilizar. E a campainha inclui a campainha ativa e a campainha passiva. Nesta experiência, adoptamos a campainha passiva. Ao utilizar a campainha passiva, podemos controlar diferentes sons introduzindo ondas quadradas com frequências diferentes. Durante a experiência, controlamos o código para fazer soar a campainha, começamos com

o som "tick, tick", depois fazemos com que a campainha passiva emita "do re mi fa so la si do" e toca músicas específicas.

Equipamento






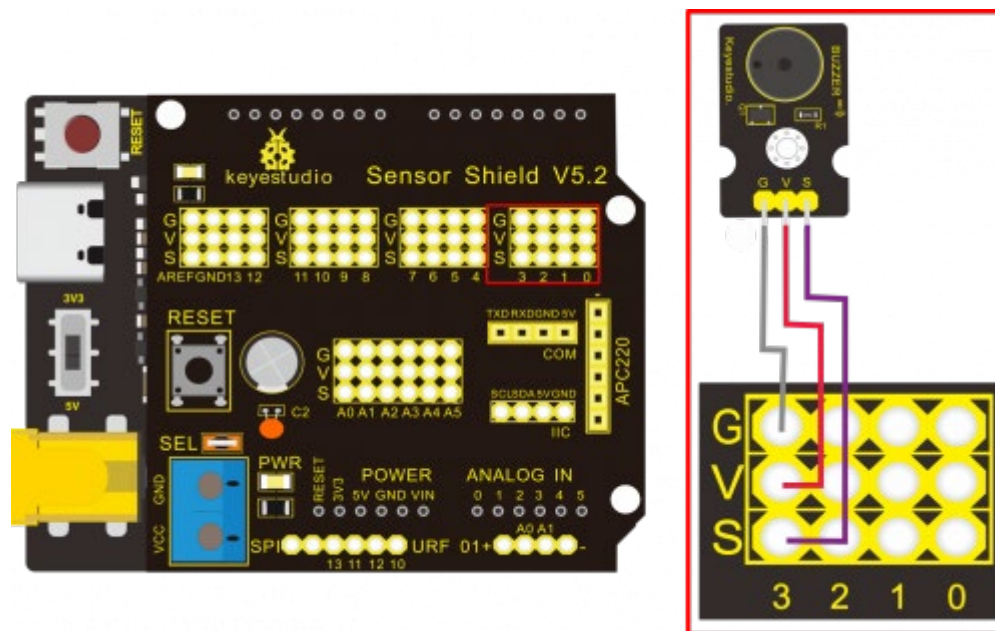
PLUS control board*1	Sensor shield*1	Passive buzzer*1	USB cable*1	3pin F-F Dupont line*1
				

Diagrama de ligação:



Nota: As portas G, V e S do módulo da campainha passiva são ligadas separadamente a G, V e 3 na blindagem, ligar.

Código de teste:

```
/*  
  
Keystudio smart home Kit for Arduino  
  
Project 3.1  
  
Buzzer  
  
http://www.keystudio.com  
  
*/  
  
int tonepin = 3; // Set the Pin of the buzzer to the digital D3  
  
void setup ()  
{  
  
    pinMode (tonepin, OUTPUT); // Set the digital IO pin mode to output  
  
}  
  
void loop ()  
{  
  
    unsigned char i, j;  
  
    while (1)  
    {  
  
        for (i = 0; i <80; i ++ ) // output a frequency sound  
        {  
  
            digitalWrite (tonepin, HIGH); // Sound  
  
            delay (1); // Delay 1ms  
  
            digitalWrite (tonepin, LOW); // No sound  
  
        }  
  
    }  
  
}
```

```
    delay (1); // Delay 1ms

}

for (i = 0; i <100; i ++) // output sound of another frequency

{

    digitalWrite (tonepin, HIGH); // Sound

    delay (2); // delay 2ms

    digitalWrite (tonepin, LOW); // No sound

    delay (2); // delay 2ms

}}

//
```

Resultado do teste:

No código acima, 80 e 100 decidem a frequência na instrução "for". O atraso controla a duração, tal como a batida na música.

Tocaremos uma música fabulosa se controlarmos bem a frequência e as batidas, por isso vamos descobrir a frequência dos tons. Como mostrado abaixo:

Bass:

Tone Note	1#	2#	3#	4#	5#	6#	7#
A	221	248	278	294	330	371	416
B	248	278	294	330	371	416	467
C	131	147	165	175	196	221	248
D	147	165	175	196	221	248	278
E	165	175	196	221	248	278	312
F	175	196	221	234	262	294	330
G	196	221	234	262	294	330	371

Alto:

Tone Note	1	2	3	4	5	6	7
A	441	495	556	589	661	742	833
B	495	556	624	661	742	833	935
C	262	294	330	350	393	441	495
D	294	330	350	393	441	495	556
E	330	350	393	441	495	556	624
F	350	393	441	495	556	624	661
G	393	441	495	556	624	661	742

Treble:

Tone Note	1#	2#	3#	4#	5#	6#	7#
A	882	990	1112	1178	1322	1484	1665
B	990	1112	1178	1322	1484	1665	1869
C	525	589	661	700	786	882	990
D	589	661	700	786	882	990	1112
E	661	700	786	882	990	1112	1248
F	700	786	882	935	1049	1178	1322
G	786	882	990	1049	1178	1322	1484

Depois de conhecer a frequência do tom, é necessário controlar o tempo de execução da nota. A música será produzida quando cada nota tocar num determinado período de tempo. O ritmo das notas divide-se em uma batida, meia batida, 1/4 de batida, 1/8 de batida. Estipulamos que o tempo de uma nota é 1, meia batida é 0,5, 1/4 de batida é 0,25, 1/8 de batida é 0,125....., portanto, a música é tocada. Vamos pegar no exemplo do "Ode to joy"

Ode to joy

Beethoven

1=D $\frac{4}{4}$

3 3 4 5 | 5 4 3 2 | 1 1 2 3 | 3 . 2 2 - |
Joy-ful, joy-ful, we a-dore thee, God of glo-ry, lord of love.

3 3 4 5 | 5 4 3 2 | 1 1 2 3 | 2 . 1 1 - |
Heart un-flod like flowers be-fore thee, O-pening to the sun a-bove.

2 2 3 1 | 2 3 4 3 1 | 2 3 4 3 2 | 1 2 5 3 |
Melt the clouds of sin and sad-ness. rieve the dark of doubts a-way. Giv-

3 3 4 5 | 5 4 3 4 2 | 1 1 2 3 | 2 . 1 1 - :|
-ver of im - mor-tal glad-ness. full us with the light of day.

Pela notação, a música tem um compasso 4/4.

Há notas especiais que precisamos de explicar:

1. a nota normal, como a primeira nota 3, corresponde a 350(frequência), ocupa 1 compasso
2. a nota com sublinhado significa 0,5 tempo
- 3 - A nota com ponto (3.) significa que é adicionado 0,5 tempo, ou seja 1+0,5 tempo
4. a nota com "-" representa que se acrescenta 1 tempo, ou seja 1+1 tempo.

5. as duas notas sucessivas com arco implicam legato, pode modificar ligeiramente a frequência da nota por detrás do legato (precisa de o depurar você mesmo), tal como reduzir ou aumentar alguns valores, o som será mais suave.

```
/*
```

```
Keystudio smart home Kit for Arduino
```

```
Project 3.2
```

```
Buzzer music
```

```
http://www.keystudio.com
```

```
*/
```

```
#define NTD0 -1
```

```
#define NTD1 294
```

```
#define NTD2 330
```

```
#define NTD3 350
```

```
#define NTD4 393
```

```
#define NTD5 441
```

```
#define NTD6 495
```

```
#define NTD7 556
```

```
#define NTDL1 147
```

```
#define NTDL2 165
```

```
#define NTDL3 175
```

```
#define NTDL4 196
```

```
#define NTDL5 221

#define NTDL6 248

#define NTDL7 278

#define NTDH1 589

#define NTDH2 661

#define NTDH3 700

#define NTDH4 786

#define NTDH5 882

#define NTDH6 990

#define NTDH7 112

// List all D-tuned frequencies

#define WHOLE 1

#define HALF 0.5

#define QUARTER 0.25

#define EIGHTH 0.25

#define SIXTEENTH 0.625

// List all beats

int tune [] = // List each frequency according to the notation

{

    NTD3, NTD3, NTD4, NTD5,

    NTD5, NTD4, NTD3, NTD2,
```

```
NTD1, NTD1, NTD2, NTD3,  
NTD3, NTD2, NTD2,  
NTD3, NTD3, NTD4, NTD5,  
NTD5, NTD4, NTD3, NTD2,  
NTD1, NTD1, NTD2, NTD3,  
NTD2, NTD1, NTD1,  
NTD2, NTD2, NTD3, NTD1,  
NTD2, NTD3, NTD4, NTD3, NTD1,  
NTD2, NTD3, NTD4, NTD3, NTD2,  
NTD1, NTD2, NTDL5, NTD0,  
NTD3, NTD3, NTD4, NTD5,  
NTD5, NTD4, NTD3, NTD4, NTD2,  
NTD1, NTD1, NTD2, NTD3,  
NTD2, NTD1, NTD1
```

```
};
```

```
float durt [] = // List the beats according to the notation
```

```
{
```

```
1,1,1,1,
```

```
1,1,1,1,
```

```
1,1,1,1,
```

```
1 + 0.5,0.5,1 + 1,
```

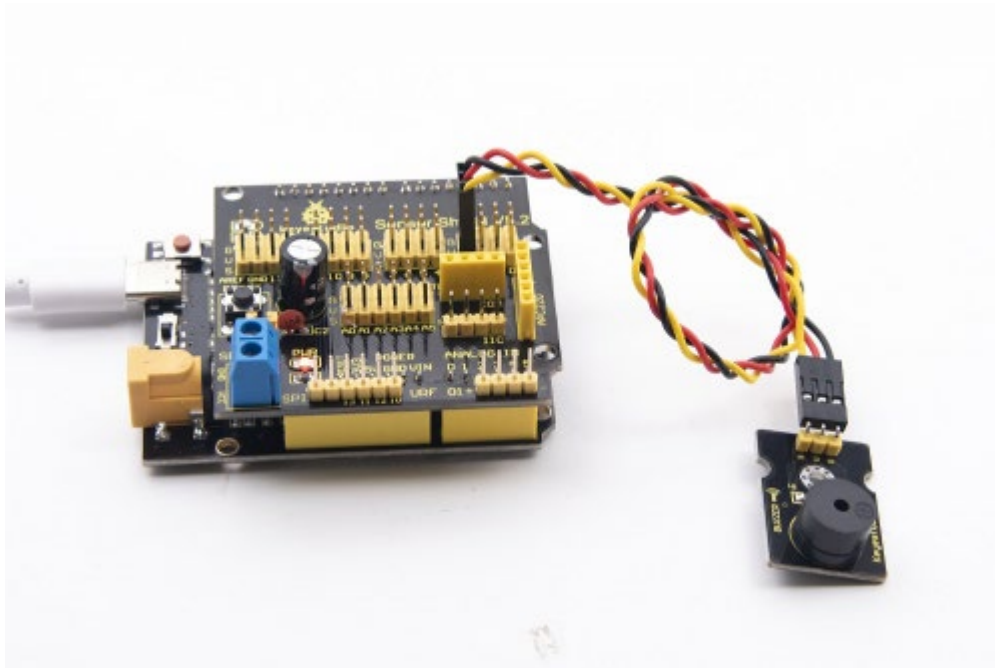
```
1,1,1,1,
```

```
1,1,1,1,  
1,1,1,1,  
1 + 0.5,0.5,1 + 1,  
1,1,1,1,  
1,0.5,0.5,1,1,  
1,0.5,0.5,1,1,  
1,1,1,1,  
1,1,1,1,  
1,1,1,0.5,0.5,  
1,1,1,1,  
1 + 0.5,0.5,1 + 1,  
};  
  
int length;  
  
int tonepin = 3; // Use interface 3  
  
void setup ()  
{  
    pinMode (tonepin, OUTPUT);  
  
    length = sizeof (tune) / sizeof (tune [0]); // Calculate length  
}  
  
void loop ()  
{  
  
    for (int x = 0; x <length; x ++)
```



```
{  
  tone (tonepin, tune [x]);  
  
  delay (350* durt [x]); // This is used to adjust the delay according to the beat,  
350 can be adjusted by yourself.  
  
  noTone (tonepin);  
}  
  
delay (2000); // delay 2S  
  
}  
  
//
```

Carregar o código de teste na placa de desenvolvimento, está a ouvir "Ode to joy" ?



Projeto 4: Controlar o LED através de um módulo de botões

Descrição:

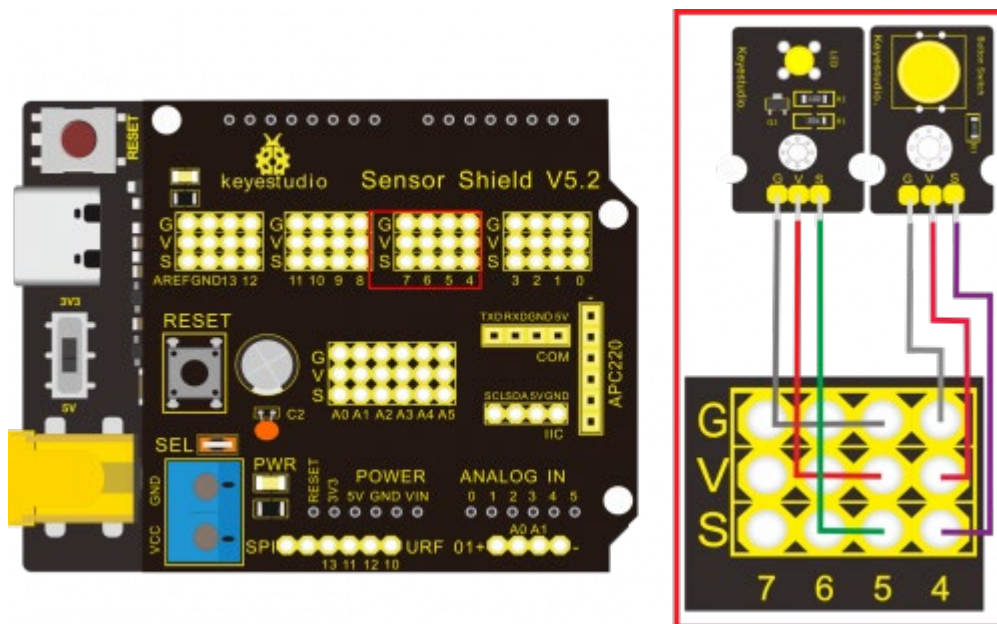


Neste projeto, vamos controlar o LED para acender e apagar através do módulo de botões. Quando o botão é premido, a extremidade do sinal emite um nível baixo (0); quando é libertado, a extremidade do sinal do sensor mantém o nível alto (1).

Equipamento

PLUS control board*1	Sensor shield*1	Yellow LED module*1	Button sensor*1	USB cable*1	3pinF-F Dupont line*2

Diagrama de ligação:



Nota: Os pinos G, V e S do módulo do sensor de botões estão ligados separadamente a G, V e 4 na proteção, e os pinos G, V e S do módulo do LED amarelo estão ligados a G, V e 5 na proteção.

Código de teste:

Em seguida, para conceber o programa, criámos um botão para ligar o LED. Em comparação com as experiências anteriores, adicionamos uma declaração de julgamento condicional. Utilizamos a instrução `if`. As frases escritas do Arduino baseiam-se na linguagem C, por isso, a declaração de julgamento condicional de C é adequada para o Arduino, como `while`, `switch`, etc.

Para esta lição, utilizamos a instrução "if" simples como exemplo para demonstrar: Se o botão for premido, o digital 4 está em nível baixo, então fazemos com que o digital 5 saia em nível alto, e o LED acende-se; inversamente, se o botão for

libertado, o digital 4 está em nível alto, fazemos com que o digital 5 saia em nível baixo, e o LED apaga-se.

Como para sua referência:

```
/*
```

```
Keystudio smart home Kit for Arduino
```

```
Project 4
```

```
Button
```

```
http://www.keyestudio.com
```

```
*/
```

```
int ledpin = 5; // Define the led light in D5
```

```
int inpin = 4; // Define the button in D4
```

```
int val; // Define variable val
```

```
void setup ()
```

```
{
```

```
pinMode (ledpin, OUTPUT); // The LED light interface is defined as output
```

```
pinMode (inpin, INPUT); // Define the button interface as input
```

```
}
```

```
void loop ()
```

```
{
```

```
val = digitalRead (inpin); // Read the digital 4 level value and assign it to val
```

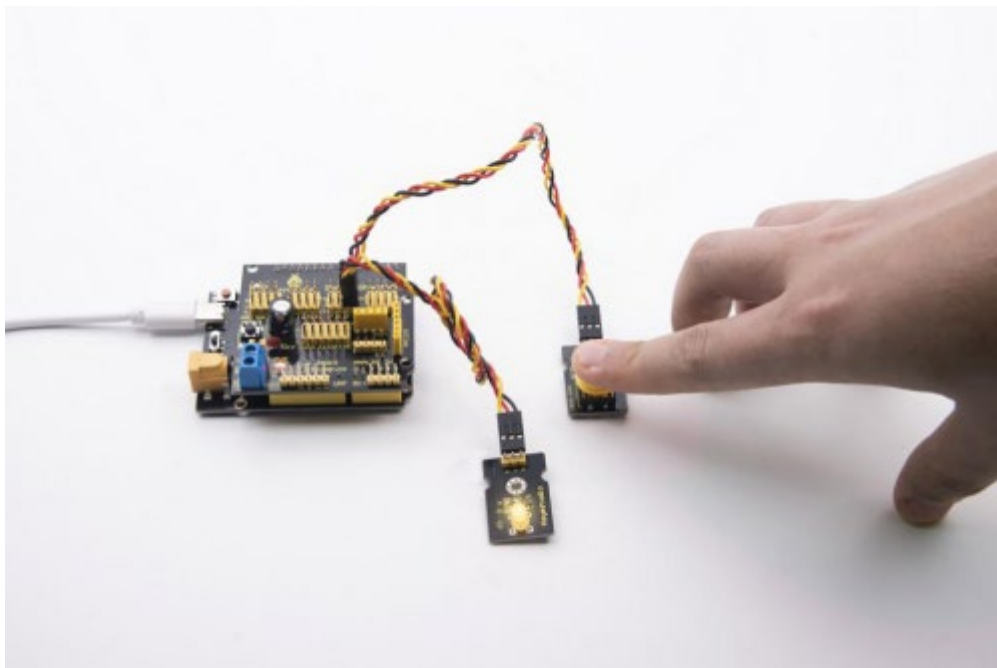
```
if (val == LOW) // Whether the key is pressed, the light will be on when pressed
```

```
{digitalWrite (ledpin, HIGH);}
```

```
else  
  
{digitalWrite (ledpin, LOW);} }  
  
//
```

Resultado do teste:

Esta experiência é bastante simples e é amplamente aplicada a vários circuitos e aparelhos eléctricos. Na nossa vida, podemos encontrar este princípio em qualquer aparelho, como a luz de fundo que se acende ao premir um botão, que é o aparelho típico.



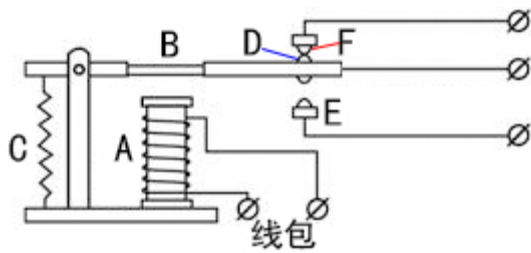
Projeto 5: Módulo de relé de 1 canal

Descrição



Este módulo é um módulo dedicado ao Arduino e é compatível com a placa de expansão de sensores do Arduino. Tem um sistema de controlo (também designado por circuito de entrada) e um sistema controlado (também designado por circuito de saída). Normalmente utilizado em circuitos de controlo automático, o módulo de relé é um "interrupor automático" que controla uma corrente maior e uma tensão menor com uma corrente menor e uma tensão menor.

Por conseguinte, desempenha o papel de ajuste automático, proteção de segurança e circuito de conversão no circuito. Permite ao Arduino acionar cargas inferiores a 3A, tais como fitas de luz LED, motores DC, bombas de água em miniatura, interface de ligação de válvulas solenóides. Os principais componentes internos do módulo de relé são o eletroímã A, a armadura B, a mola C, o contacto móvel D, o contacto estático (contacto normalmente aberto) E e o contacto estático (contacto normalmente fechado) F, (como mostra a figura).



Enquanto uma certa tensão for aplicada a ambas as extremidades da bobina, uma certa corrente fluirá através da bobina para gerar efeitos electromagnéticos, e a armadura atrairá o núcleo de ferro contra a força de tração da mola de retorno sob a ação da atração da força electromagnética, levando assim o contacto móvel e o contacto estático (contacto normalmente aberto) a atraírem-se. Quando a bobina é desligada, a sucção electromagnética também desaparece e a armadura regressa à posição original sob a força de reação da mola, libertando o contacto móvel e o contacto estático original (contacto normalmente fechado). Este processo puxa e liberta, atingindo assim o objetivo de ligar e desligar o circuito. Os contactos "normalmente abertos e fechados" do relé podem ser distinguidos da seguinte forma: os contactos estáticos no estado desligado quando a bobina do relé está desligada são chamados "contactos normalmente abertos"; os contactos estáticos no estado ligado são chamados "contactos normalmente fechados". O módulo é fornecido com 2 orifícios de posicionamento para fixar o módulo a outro equipamento.

Especificações:

- Tensão de funcionamento: 5V (DC)

- Interface: Interface G, V, S
- Sinal de entrada: sinal digital (nível alto 1, nível baixo 0)
- Contactos: contactos estáticos (contactos normalmente abertos, contactos normalmente fechados) e contactos móveis
- Corrente nominal: 10A (NA) 5A (NF)
- Tensão máxima de comutação: 150 V (AC) 24 V (DC)
- Corrente de choque elétrico: inferior a 3A
- Peso: 15g
- Tempo de ação do contacto: 10ms

Equipamento:

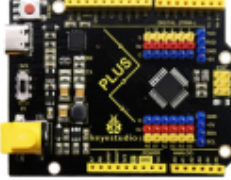



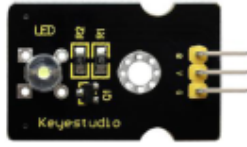



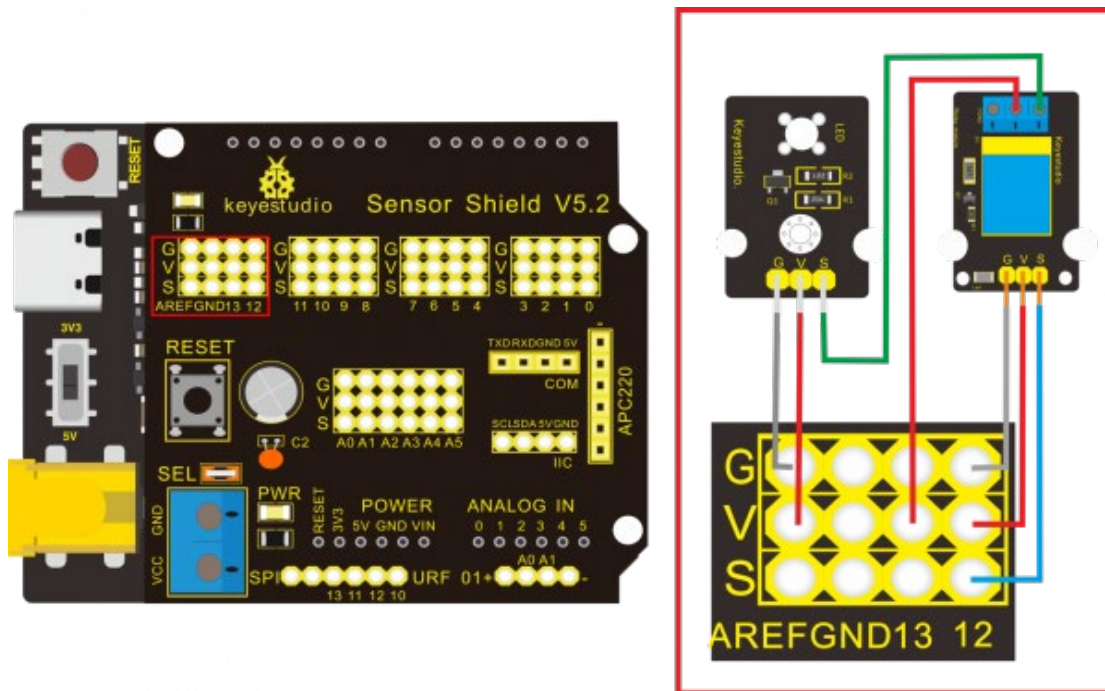
PLUS control board*1	Sensor shield*1	USB cable*1
		
Relay module*1	White LED*1	3pin F-F Dupont Line*1
		
Female to Female Dupont Lines*2	Male to Female Dupont Line*2	
		

Diagrama de ligação:



Nota: Na blindagem, os pinos G, V e S do módulo de relé de 1 canal estão ligados a G, V e 12, respetivamente. O NO está ligado a V; os pinos G, V e S do LED branco estão ligados respetivamente a G, V e ao contacto estático do NO no módulo de relé.

Código de teste:

/*

Keystudio smart home Kit for Arduino

Project 5

Relay

<http://www.keystudio.com>

*/

```
int Relay = 12; // Define the relay pin at D12

void setup ()

{

pinMode (13, OUTPUT); // Set Pin13 as output

digitalWrite (13, HIGH); // Set Pin13 High

pinMode (Relay, OUTPUT); // Set Pin12 as output

}

void loop ()

{

digitalWrite (Relay, HIGH); // Turn off relay

delay (2000);

digitalWrite (Relay, LOW); // Turn on relay

delay (2000);

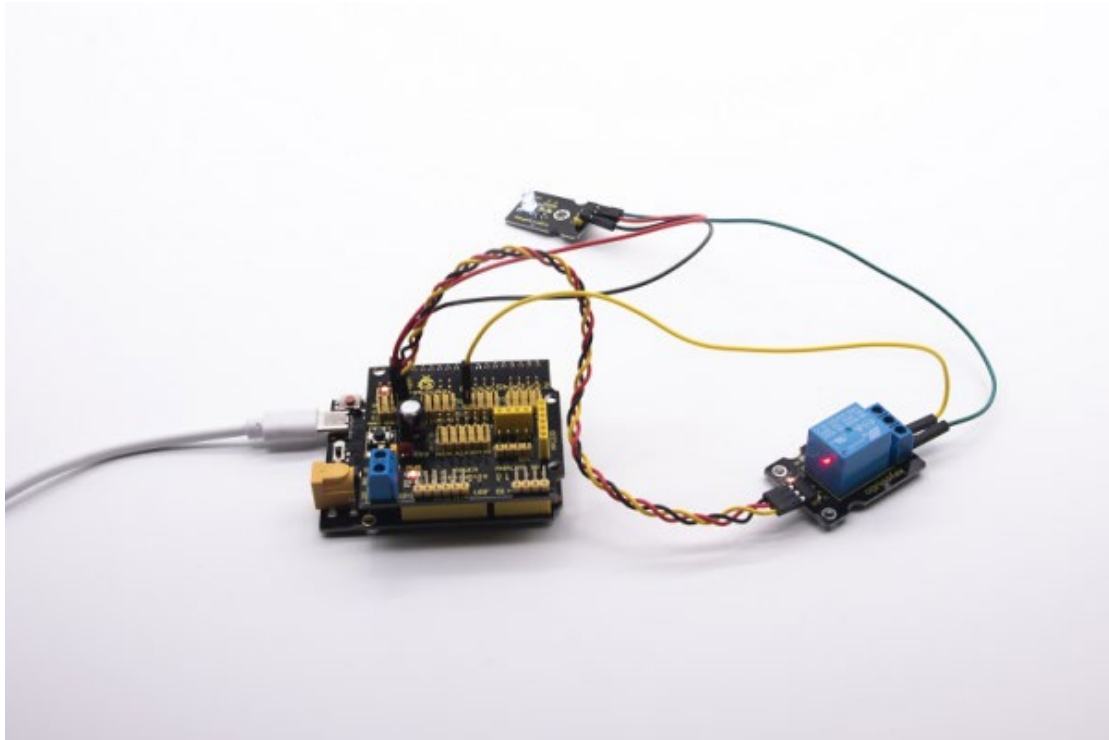
}

//
```

Resultado do teste:

Ligar o cabo, ligar a corrente e carregar o código de teste. O relé é ligado ("NO" está ligado, NC está desligado) durante 0,5s, depois desligado durante 0,5s (NC está ligado, NO está desligado), e alternadamente. Quando o relé está ligado, o

LED branco acende-se e, inversamente, o LED branco apaga-se.



Projeto 6: Sensor de fotocélula

Descrição



O sensor de fotocélula (fotoresistor) é uma resistência produzida pelo efeito fotoelétrico de um semicondutor. É muito sensível à luz ambiente, pelo que o seu valor de resistência varia com diferentes intensidades de luz. Utilizamos as suas características para conceber um circuito e criar um módulo sensor de fotoresistência. A extremidade do sinal do módulo é ligada à porta analógica do

microcontrolador. Quando a intensidade da luz aumenta, a resistência diminui e a tensão da porta analógica aumenta, ou seja, o valor analógico do microcontrolador também aumenta. Caso contrário, quando a intensidade da luz diminui, a resistência aumenta e a tensão da porta analógica diminui. Ou seja, o valor analógico do microcontrolador torna-se mais pequeno. Por conseguinte, podemos utilizar o módulo sensor de fotorresistência para ler o valor analógico correspondente e detetar a intensidade da luz no ambiente. É normalmente aplicado à medição, controlo e conversão da luz, bem como ao circuito de controlo da luz.

Especificações:

- Tensão de funcionamento: 3,3V-5V (DC)
- Interface: Interface 3PIN
- Sinal de saída: sinal analógico
- Peso: 2.3g

Equipamento:

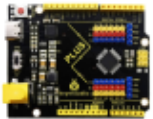


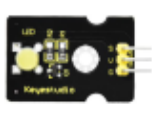


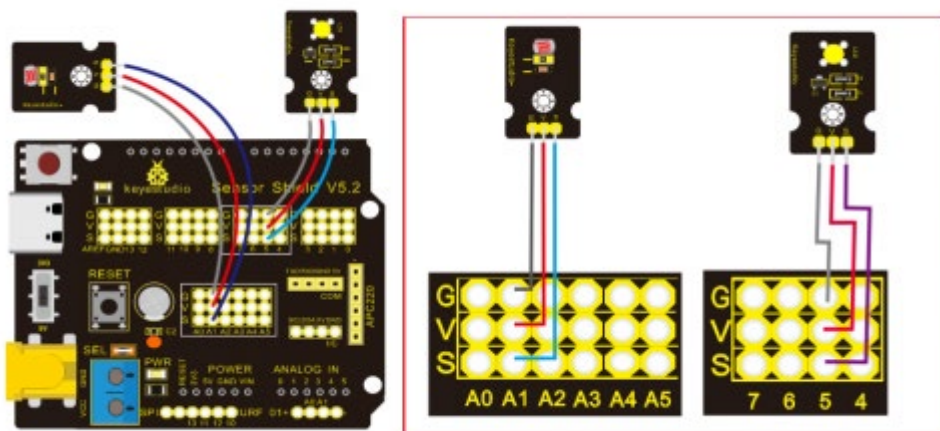
PLUS control board*1	Sensor shield*1	Photocell sensor*1	Yellow LED module*1	USB cable*1	3pin F-F Dupont line*2
					

Diagrama de ligação:



Nota: Na placa de expansão, os pinos G, V e S do módulo de sensor de fotocélula estão ligados a G, V e A1; os pinos G, V e S do módulo de LED amarelo estão ligados a G, V e 5 separadamente.

Código de teste:

/*

Keyestudio smart home Kit for Arduino

Project 6

photocell

<http://www.keyestudio.com>

*/

```
int LED = 5; // Set LED pin at D5
```

```
int val = 0; // Read the voltage value of the photodiode
```

```
void setup () {  
  
    pinMode (LED, OUTPUT); // LED is output  
  
    Serial.begin (9600); // The serial port baud rate is set to 9600  
  
}  
  
void loop () {  
  
    val = analogRead (A1); // Read the voltage value of A1 Pin  
  
    Serial.println (val); // Serial port to view the change of voltage value  
  
    if (val <900)  
  
        {  
  
        // Less than 900, LED light is off  
  
        digitalWrite (LED, LOW);  
  
        }  
  
    else  
  
        {  
  
        // Otherwise, the LED lights up  
  
        digitalWrite (LED, HIGH);  
  
        }  
  
    delay (10); // Delay 10ms  
  
}  
  
//
```

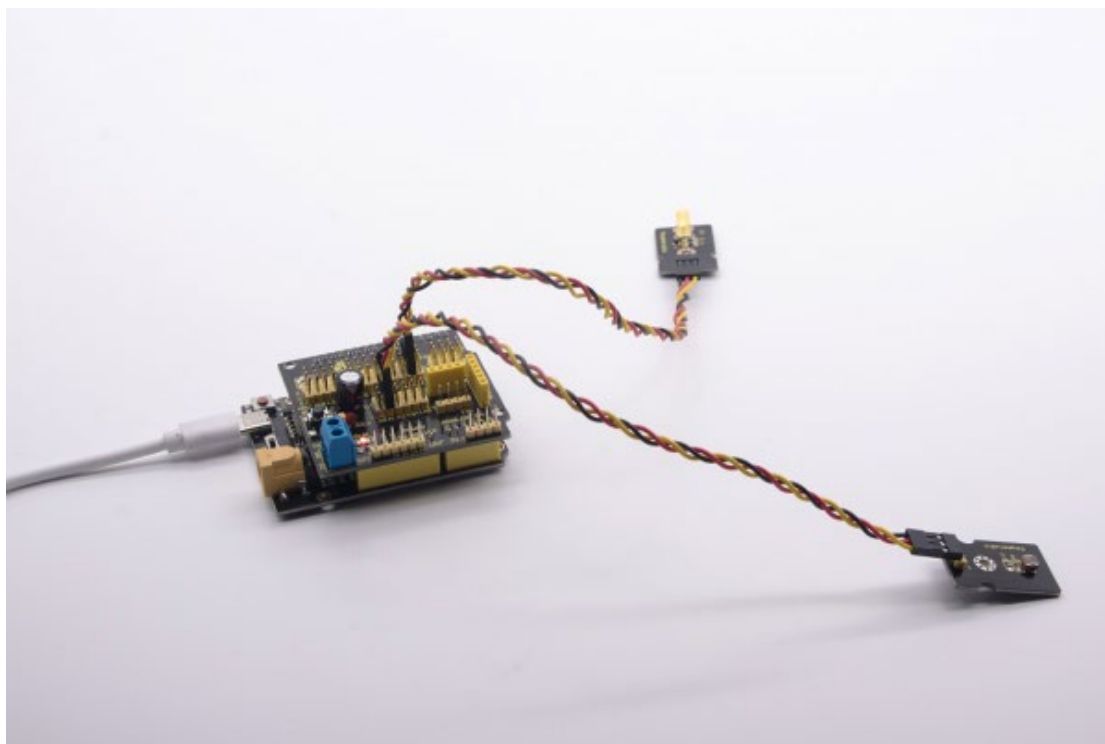
Resultado do teste:

O LED acende-se depois de carregar o código de teste, aponta-se para o sensor da fotocélula com a lanterna (ou o flash do telemóvel), verifica-se que o LED se apaga automaticamente. No entanto, se retirar a lanterna, o LED acende-se de novo.

Review

Para esta cadeia de código, é simplesmente. Lemos o valor através da porta analógica, tenha em atenção que a quantidade analógica não necessita de modo de entrada e saída.

O valor analógico diminui gradualmente quando há luz, o valor é de até 1000, este valor pode ser escolhido de acordo com o brilho que você precisa. Método de seleção: colocar todo o dispositivo no ambiente onde o LED está desligado, abrir o monitor de série para verificar o valor apresentado, substituir 1000 por este valor. O valor de leitura do monitor de série é uma boa forma de modular o código



Projeto 7: Ajustar o ângulo do servo motor

Descrição

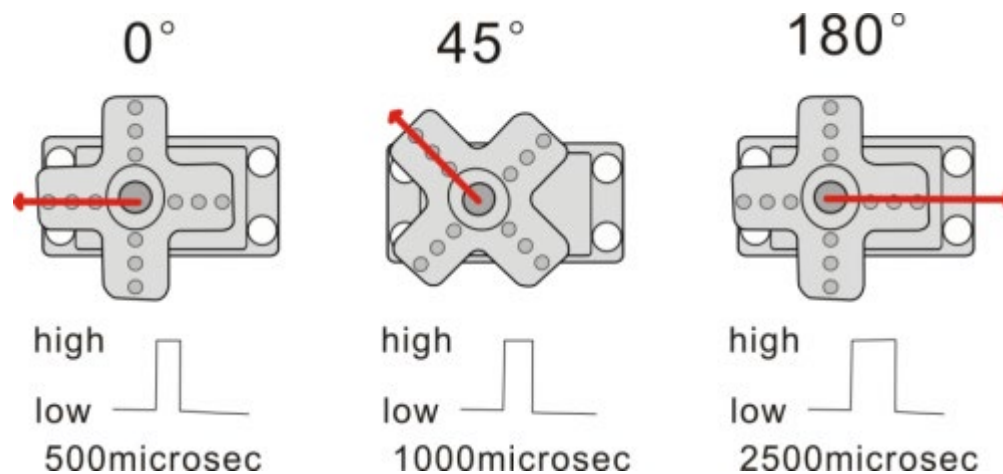


Quando fazemos este kit, é frequente controlarmos as portas e as janelas com servos. Neste curso, vamos apresentar o seu princípio e como utilizar os servomotores. O servo motor é um atuador rotativo de controlo de posição. É constituído principalmente por caixa, placa de circuito, motor sem núcleo, engrenagem e sensor de posição. O seu princípio de funcionamento é que o servo recebe o sinal enviado pelo MCU ou pelo recetor e produz um sinal de referência com um período de 20ms e uma largura de 1,5ms, comparando depois a tensão de polarização CC adquirida com a tensão do potenciómetro e produzindo uma diferença de tensão.

O servo motor tem muitas especificações. Mas todos eles têm três fios de ligação, distinguidos pelas cores castanha, vermelha e laranja (marcas diferentes podem ter cores diferentes). O castanho é para o GND, o vermelho para o positivo de potência, o laranja para a linha de sinal.

O ângulo de rotação do servomotor é controlado através da regulação do ciclo de trabalho do sinal PWM (Pulse-Width Modulation). O ciclo padrão do sinal PWM é de 20ms (50Hz). Teoricamente, a largura é distribuída entre 1ms-2ms, mas, de facto, é entre 0,5ms-2,5ms. A largura corresponde ao ângulo de rotação de 0° a 180°. Mas note-se que para motores de marcas diferentes, o mesmo sinal pode

ter um ângulo de rotação diferente.



Há duas formas de controlar um servomotor com o Arduino. Uma é utilizar uma porta de sensor digital comum do Arduino para produzir uma onda quadrada com diferentes ciclos de funcionamento para simular o sinal PWM e utilizar esse sinal para controlar o posicionamento do motor. Outra forma é utilizar diretamente a função Servo do Arduino para controlar o motor. Desta forma, o programa será mais fácil, mas só pode controlar o motor de dois contactos porque, para a função servo, só podem ser utilizados os pinos digitais 9 e 10. A capacidade de acionamento do Arduino é limitada. Por isso, se precisar de controlar mais do que um motor, precisará de alimentação externa.

Equipamento para experiências:

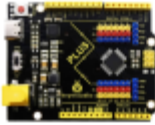



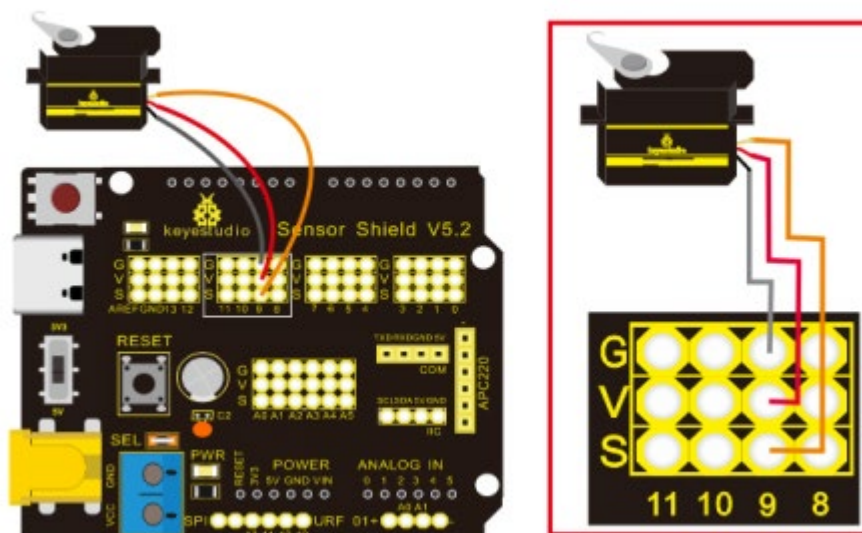
PLUS control board*1	Sensor shield*1	Servo*1	USB cable*1
			

Diagrama de ligação:



Nota: O servo está ligado a G (GND), V (VCC), 9. O fio castanho do servo está ligado a Gnd (G), o fio vermelho está ligado a 5v (V) e o fio laranja está ligado ao pino digital 9.

Código de teste:

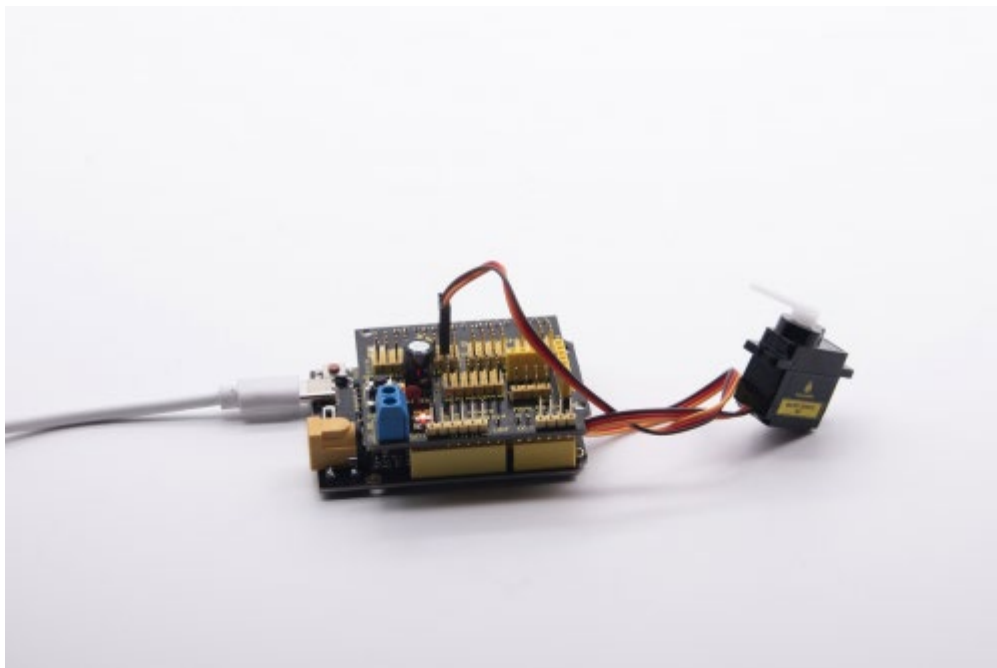
```
/*  
  
Keystudio smart home Kit for Arduino  
  
Project 7  
  
Sevro  
  
http://www.keystudio.com  
  
*/  
  
#include <Servo.h> // Servo function library  
  
Servo myservo;  

```

```
}  
  
for(pos = 180; pos>=1; pos-=1) // Angle from 180 to 0 degrees  
  
{  
  
myservo.write (pos); // The angle of the servo is pos  
  
delay (15); // Delay 15ms  
  
}  
  
}  
  
//
```

Resultado do teste:

Carregue o código, ligue-o de acordo com o diagrama de ligação e ligue-o. O servo roda de 0° a 180° e depois de 180°~0°



Projeto 8: Módulo de ventilador

Descrição



O módulo de ventilador L9110 adopta o chip de controlo do motor L9110, que pode controlar o sentido de rotação e a velocidade do motor. Além disso, este módulo é eficiente e tem uma ventoinha de alta qualidade, que pode apagar a chama a uma distância de 20 cm. Do mesmo modo, é também uma parte importante do robô de incêndio

Especificações:

- Diâmetro do ventilador: 75mm
- Tensão de funcionamento: 5V

Equipamento:






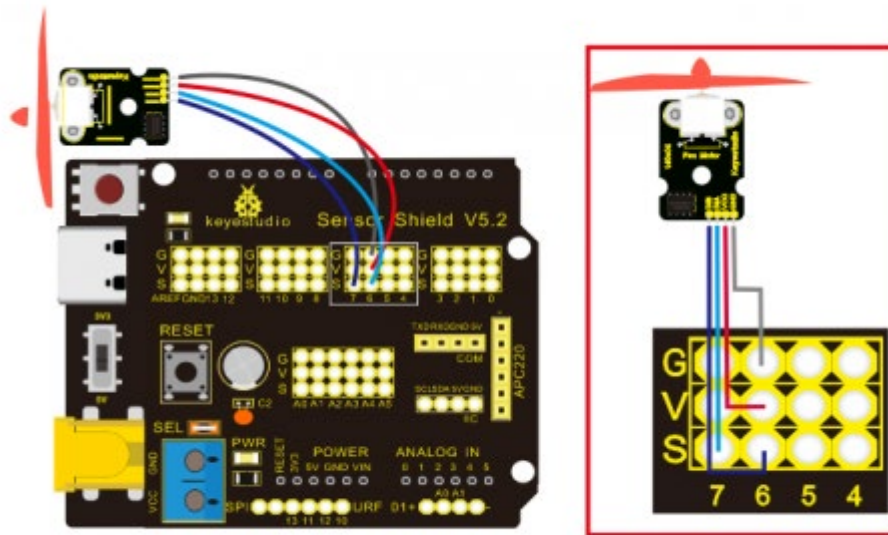
PLUS control board*1	Sensor shield*1	Fan module*1	USB cable*1	Female to Female Dupont lines*4
				

Diagrama de ligação:



Nota: Na blindagem, os pinos GND, VCC, INA e INB do módulo do ventilador estão ligados respetivamente a G, V, 7, 6.

Código de teste:

/*

Keyestudio smart home Kit for Arduino

Project 8

Fan

<http://www.keyestudio.com>


```

*/

void setup () {

    pinMode (7, OUTPUT); //define D7 pin as output

    pinMode (6, OUTPUT); //define D6 pin as output

}

void loop () {

    digitalWrite (7, LOW);

    digitalWrite (6, HIGH); // Reverse rotation of the motor

    delay (3000); // delay 3S

    digitalWrite (7, LOW);

    digitalWrite (6, LOW); // The motor stops rotating

    delay (1000); //delay 1S

    digitalWrite (7, HIGH);

    digitalWrite (6, LOW); // The motor rotates in the forward direction

    delay (3000); // delay 3S

}

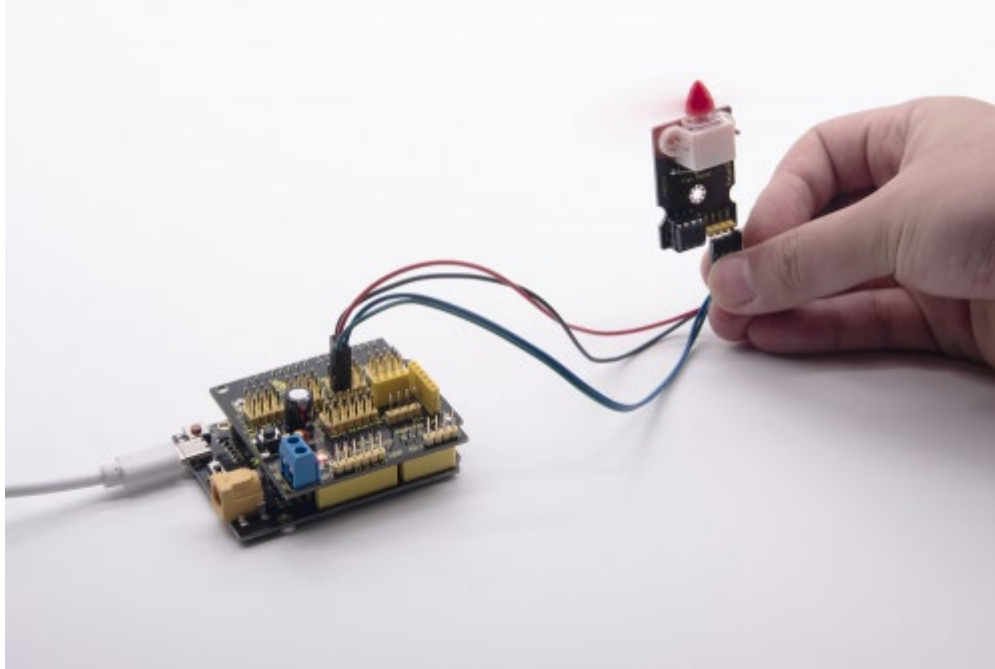
//

```

Resultado do teste: :

Carregar o código de teste, ligar os fios de acordo com o diagrama de ligação, o interruptor DIP é colocado no lado direito e ligar. A ventoinha roda no sentido

contrário ao dos ponteiros do relógio durante 3000ms, pára durante 1000ms e depois roda no sentido dos ponteiros do relógio durante 3000ms.



Projeto 9: Sensor de vapor

Descrição



Este é um sensor de vapor comumente utilizado. O seu princípio é detetar a quantidade de água através de linhas paralelas impressas na placa de circuitos. Quanto maior for a quantidade de água, mais fios serão ligados. À medida que a área de contacto condutor aumenta, a tensão de saída aumenta gradualmente.

Também pode detetar vapor de água no ar. O sensor de vapor pode ser utilizado como detetor de água da chuva e interruptor de nível. Quando a humidade na superfície do sensor aumenta, a tensão de saída aumenta.

O sensor é compatível com várias placas de controlo de microcontroladores, como os microcontroladores da série Arduino. Ao utilizá-lo, fornecemos o guia para operar o sensor de vapor e a placa de controlo Arduino. Ligar a extremidade do sinal do sensor à porta analógica do microcontrolador, detetar a alteração do valor analógico e apresentar o valor analógico correspondente no monitor de série.

Nota: a parte de ligação não é à prova de água, por isso não a mergulhe na água.

Especificações:

- Tensão de funcionamento: DC 3.3-5V
- Corrente de trabalho: <20mA
- Faixa de temperatura de operação: -10 °C ~ + 70 °C;
- Sinal de controle: saída de sinal analógico
- Interface: interface de pinos de 3 pinos de 2,54 mm
- Tamanho: 35 * 20 * 8mm
- Peso: 2.2g
- S: saída de sinal
- V (+): Fonte de alimentação (VCC)

- G (-): Terra (GND)

Equipamento:






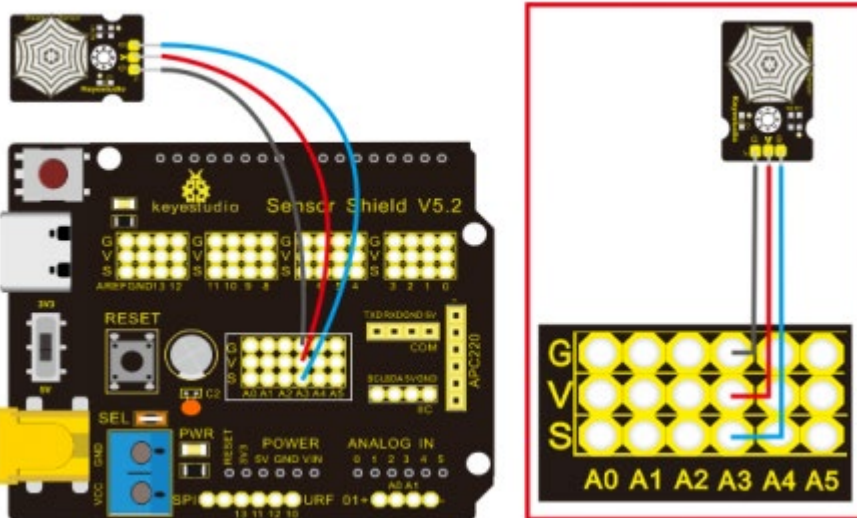
PLUS control board*1	Sensor shield*1	Steam sensor*1	USB cable*1	3pinF-F Dupont line*1
				

Diagrama de ligação:



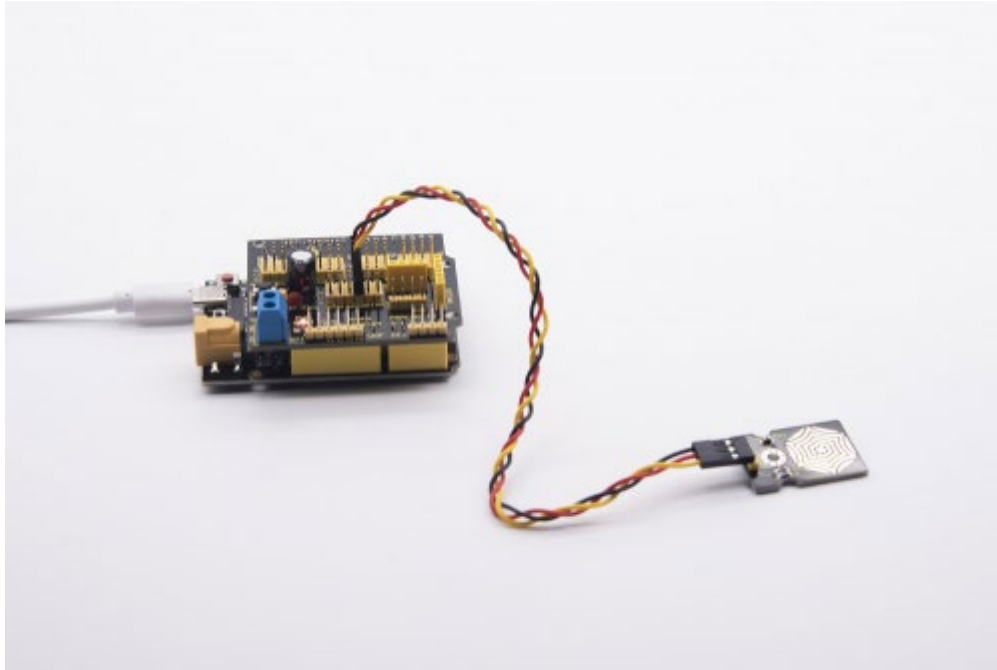
Nota: Na blindagem do sensor, os pinos G, V e S do sensor de vapor estão ligados a G, V e A3; os pinos G, V e S do sensor de fotocélula estão ligados a G, V e A1; os pinos G, V e S do LED amarelo estão ligados a G, V e 5; a linha castanha está ligada a G, o fio vermelho a V, o fio laranja a 9.

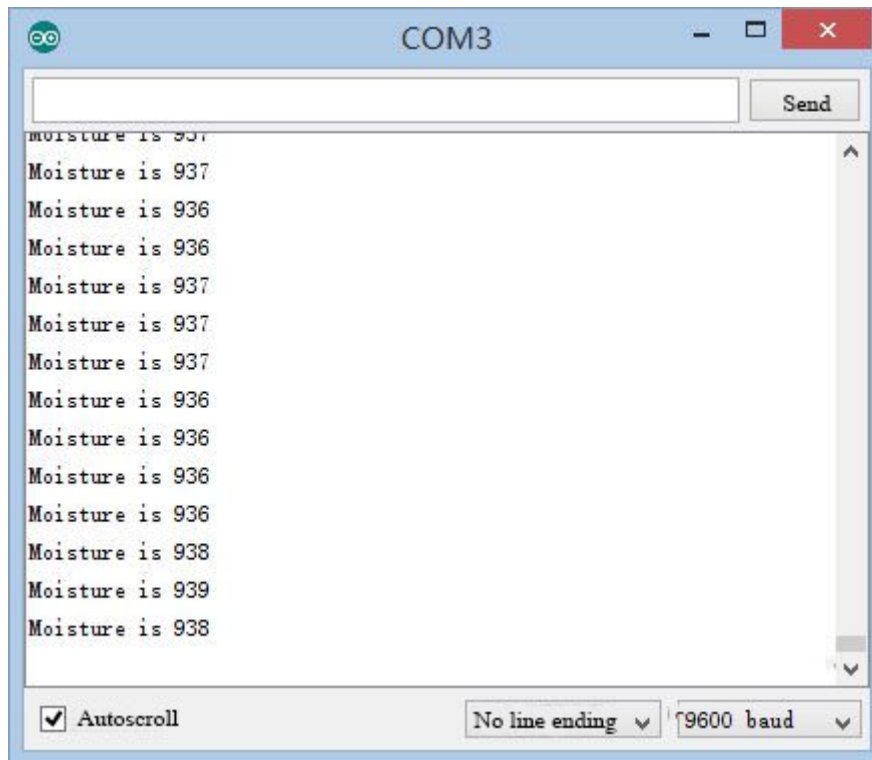
Código de teste:

```
/*  
  
Keyestudio smart home Kit for Arduino  
  
Project 9  
  
Steam  
  
http://www.keyestudio.com  
  
*/  
  
void setup()  
  
{  
  
Serial.begin(9600); //open serial port, and set baud rate at 9600bps  
  
}  
  
void loop()  
  
{  
  
int val;  
  
val=analogRead(3); //plug vapor sensor into analog port 3  
  
Serial.print("Moisture is ");  
  
Serial.println(val,DEC); //read analog value through serial port printed  
  
delay(100); //delay 100ms  
  
}
```

Resultado do teste: :

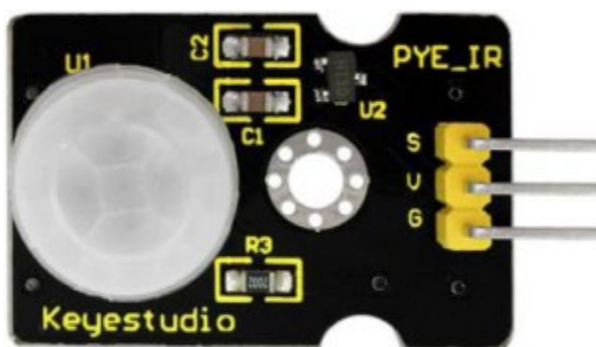
Ao detetar diferentes graus de humidade, o sensor obterá o feedback de diferentes valores de corrente. Apresentado na imagem seguinte. Quando o sensor detecta o vapor de água fervida, o valor da humidade é apresentado no monitor de série do software Arduino.





Projeto 10: Sensor de movimento PIR

Descrição



O sensor de movimento infravermelho piroelétrico pode detetar sinais infravermelhos de uma pessoa ou animal em movimento e emitir sinais de comutação. Pode ser aplicado numa variedade de ocasiões para detetar o

movimento do corpo humano. Os sensores de infravermelhos piroelétricos convencionais são muito maiores, com circuitos complexos e menor fiabilidade. Agora lançamos este novo sensor de movimento infravermelho piroelétrico, especialmente concebido para Arduino. Este sensor integra um sensor de infravermelhos piroelétrico digital integrado e os pinos de ligação. Apresenta maior fiabilidade, menor consumo de energia e um circuito periférico mais simples.

Especificações:

- Tensão de entrada: DC 3.3V ~ 18V
- Corrente de trabalho: 15uA
- Temperatura de funcionamento: -20 ~ 85 graus Celsius
- Tensão de saída: alta 3 V, baixa 0 V
- Tempo de atraso de saída (nível alto): cerca de 2,3 a 3 segundos
- Ângulo de deteção: cerca de 100 °
- Distância de deteção: 3-4 metros
- LED indicador de saída (luz de nível elevado)
- Corrente limite do pino: 100mA

Special note:

- - 1. A distância máxima é de 3-4 metros durante o teste.

- - 2. Durante o teste, abra primeiro a lente branca e poderá ver a parte de detecção retangular. Quando a linha longa da parte de detecção retangular está paralela ao chão, a distância é a melhor.
- - 3. Durante o teste, o sensor tem de ser coberto com a lente branca, caso contrário, a distância será afetada.
- - 4. A distância é melhor em 25 °C, e a distância de detecção é encurtada quando excede 30 °C.
- - 5. Done ligar e fazer o upload do código, você precisa esperar 5-10 segundos, em seguida, começar a testar, caso contrário, não é sensível.

Equipamento:

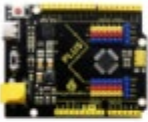







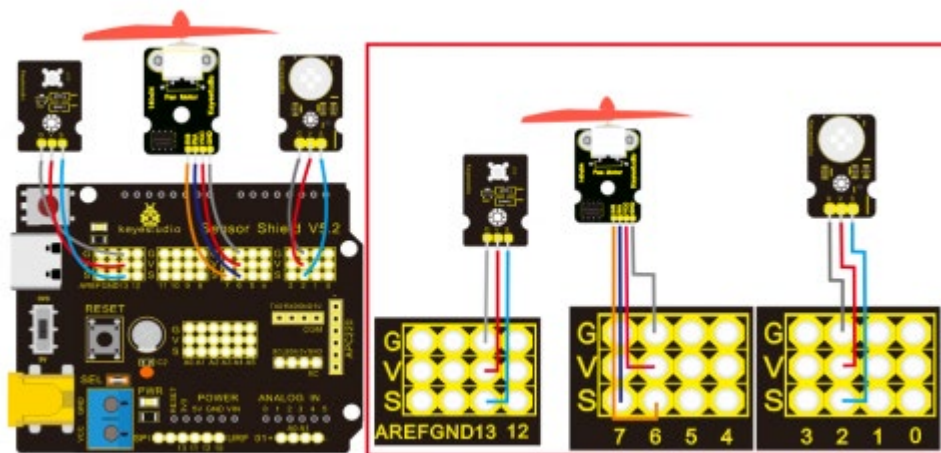
PLUS control board*1	Sensor shield*1	PIR Motion Sensor*1	Female to Female Dupont lines*4
			
Fan module*1	White LED module*1	USB cable*1	3pinF-F Dupont line*2
			

Diagrama de ligação:



Nota: Na blindagem, os pinos G, V e S do sensor de movimento PIR estão ligados a G, V e 2; os pinos GND, VCC, INA e INB do módulo do ventilador estão ligados separadamente a G,V,7,6. Os pinos G, V e S do módulo LED estão ligados a G, V e 13.

Código de teste:

```
/*  
Keystudio smart home Kit for Arduino  
Project 10  
PIR  
http://www.keystudio.com  
*/
```

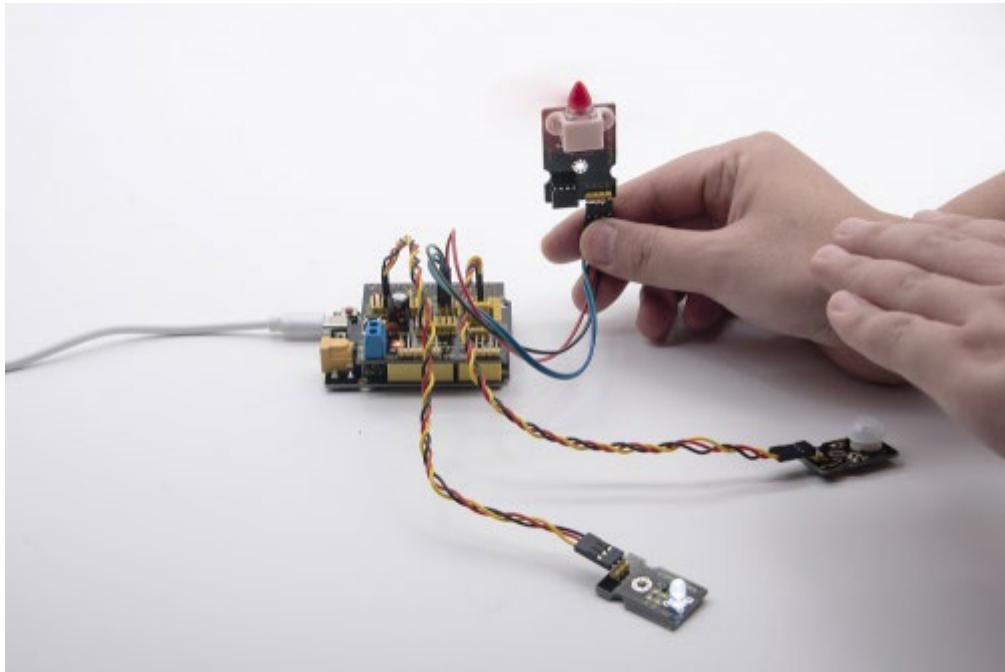
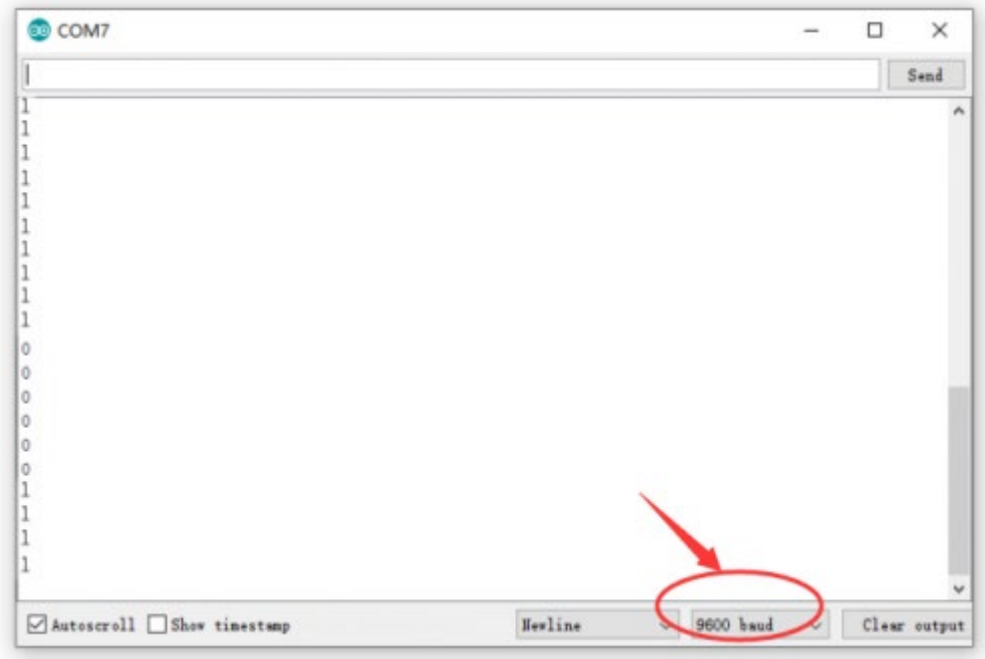
```
void setup () {  
  
    Serial.begin (9600); // open serial port, and set baud rate at 9600bps  
  
    pinMode (2, INPUT); // Define PIR as input in D2  
  
    Serial.begin (9600);  
  
    pinMode (13, OUTPUT); // Define LED as output in D13  
  
    pinMode (7, OUTPUT); // Define D7 as output  
  
    pinMode (6, OUTPUT); // Define D6 as output  
  
}
```

```
void loop () {  
  
    Serial.println (digitalRead (2));  
  
    delay (500); // Delay 500ms  
  
    if (digitalRead (2) == 1) // If someone is detected walking  
    {  
  
        digitalWrite (13, HIGH); // LED light is on  
  
        digitalWrite (7, HIGH);  
  
        analogWrite (6,150); // Fan rotates  
  
    } else // If no person is detected walking  
    {  
  
        digitalWrite (13, LOW); // LED light is not on  
  
        digitalWrite (7, LOW);  
  
    }  
  
}
```

```
    analogWrite (6,0); // The fan does not rotate  
  
  }  
  
  //
```

Resultado do teste: :

Carregar o Código de teste, abrir o monitor de série e definir a taxa de transmissão para 9600. Se o sensor de movimento PIR detetar a presença de pessoas, o monitor de série apresenta "1", o indicador D13 e o LED branco acendem ao mesmo tempo, a ventoinha roda. Se não estiver nenhuma pessoa por perto, o monitor de série apresenta "0", o indicador D13 e o LED branco estão desligados. A ventoinha pára de rodar.



Projeto 11: Analógico (MQ-2) Sensor

Descrição



Este sensor de gás é utilizado para alarmes de fuga de gás doméstico, alarmes de gás combustível industrial e instrumentos portáteis de deteção de gás. É adequado para a deteção de gás liquefeito, benzeno, alceno, álcool, hidrogénio, etc., e amplamente utilizado em vários sistemas de alarme de incêndio. O sensor de fumo MQ-2 pode ser precisamente um detetor multigás e tem as vantagens de alta sensibilidade, resposta rápida, boa estabilidade, longa vida útil e circuito de acionamento simples. Pode detetar a concentração de gás inflamável e fumo na gama de 300~10000ppm. Entretanto, tem alta sensibilidade ao gás natural, gás de petróleo liquefeito e outros fumos, especialmente ao fumo de alcanos. Tem de ser aquecido durante um período de tempo antes de utilizar o sensor de fumo, caso contrário a resistência e a tensão de saída não são exactas. No entanto, a tensão de aquecimento não deve ser demasiado elevada, caso contrário, fará com que a minha linha de sinal interna rebente.

Pertence ao material sensível ao gás semiconductor de dióxido de estanho e pertence ao semiconductor de tipo N de iões de superfície. A uma determinada

temperatura, o dióxido de estanho adsorve o oxigénio no ar e forma uma adsorção de iões negativos de oxigénio, reduzindo a densidade de electrões no semiconductor, aumentando assim o seu valor de resistência. Quando em contacto com gás inflamável no ar e smog, se a barreira de potencial no limite do grão for ajustada pelo smog, isso fará com que a condutividade da superfície se altere. Deste modo, é possível obter informações sobre a presença de fumo ou de gás inflamável. Quanto maior for a concentração de fumo ou gás inflamável no ar, maior será a condutividade, e quanto menor for a resistência de saída, maior será o sinal analógico de saída. O sensor é fornecido com um orifício de posicionamento, o que é conveniente para fixar o sensor a outros dispositivos. Além disso, a sensibilidade pode ser ajustada rodando o potenciómetro.

Especificações:

- Tensão de funcionamento: 3,3-5V (DC)
- Interface: 4 pinos (VCC, GND, D0, A0)
- Sinal de saída: sinal digital e sinal analógico
- Peso: 7,5g

Equipamento:

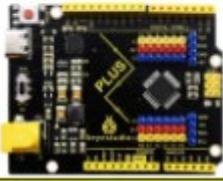






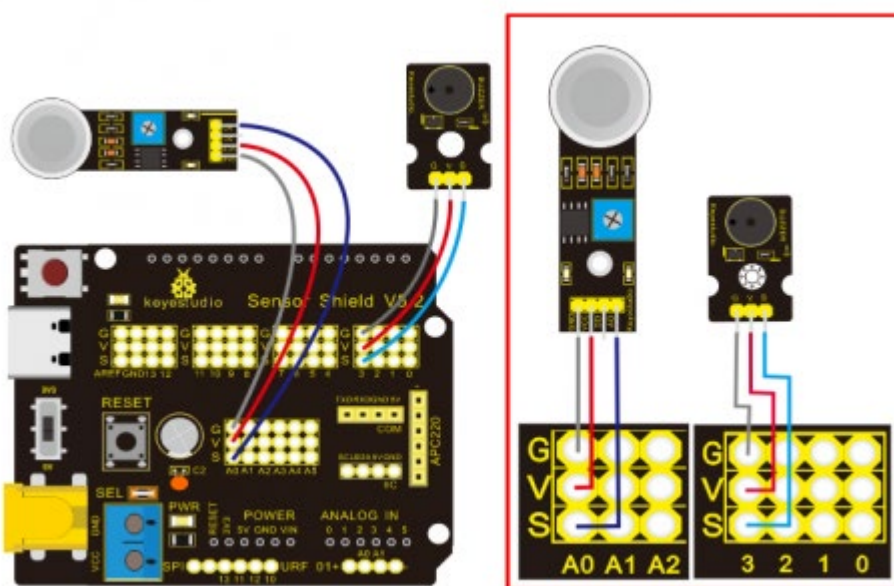
PLUS control board*1	Sensor shield*1	MQ-2 gas sensor*1	3pinF-F Dupont line*1
			
Passive buzzer*1	USB cable*1	F-F Dupont line*3	
			

Diagrama de ligação:



Nota: Na blindagem, os pinos GND, VCC, D0 e A0 do sensor de gás estão ligados aos pinos GND, VCC, D0 e A0. Os pinos GND, VCC, INA e INB do módulo do

ventilador estão ligados a G,V, 7 e 6. Os pinos G,V e S da campainha passiva estão ligados a G,V e 3; os pinos G, V e S do LED amarelo estão ligados a G,V e 5.

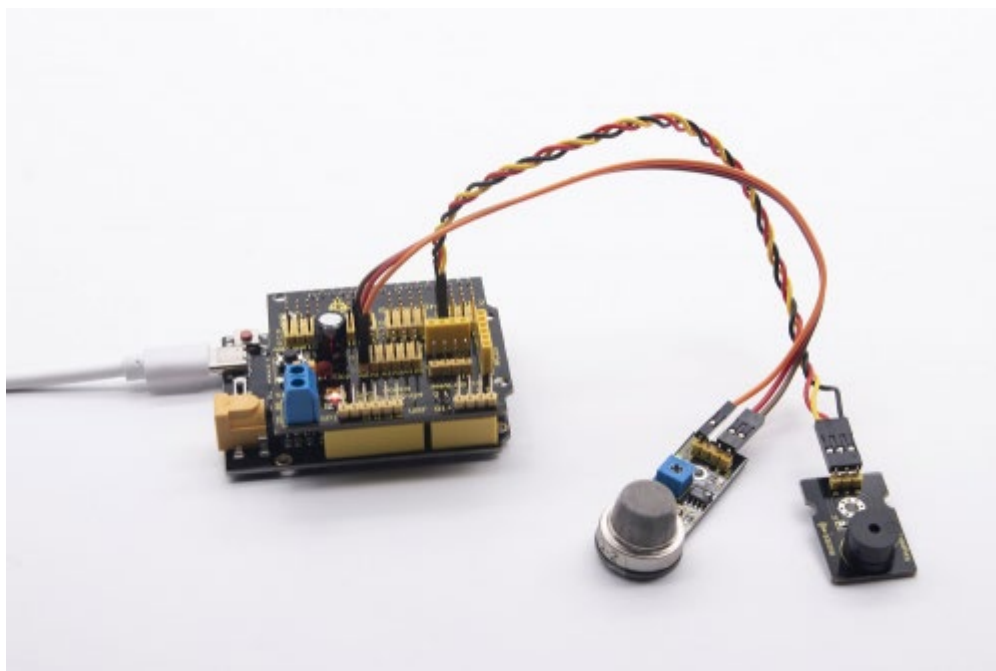
Código de teste:

```
/*  
Keyestudio smart home Kit for Arduino  
Project 11  
Gas  
http://www.keyestudio.com  
*/  
int MQ2 = A0; // Define MQ2 gas sensor pin at A0  
int val = 0; // declare variable  
int buzzer = 3; // Define the buzzer pin at D3  
void setup ()  
{  
pinMode (MQ2, INPUT); // MQ2 gas sensor as input  
Serial.begin (9600); // Set the serial port baud rate to 9600  
pinMode (buzzer, OUTPUT); // Set the digital IO pin mode for output  
}  
void loop ()
```

```
{  
  
val = analogRead (MQ2); // Read the voltage value of A0 port and assign it to val  
  
Serial.println (val); // Serial port sends val value  
  
if (val > 450)  
{  
  
tone (buzzer, 589);  
  
delay(300);  
  
}  
  
else  
  
{  
  
noTone (buzzer);  
  
}  
  
}  
  
//
```

Resultado do teste: :

Carregar o código de teste, ligar os fios de acordo com o esquema de ligação e ligar a alimentação. Quando o sensor de gás detecta o gás inflamável, o sinal sonoro passivo soa, a ventoinha roda e o LED amarelo acende-se; quando não há gás inflamável, o sinal sonoro passivo não soa, a ventoinha não roda e o LED amarelo apaga-se.



Projeto 12: Ecrã LCD 1602

Descrição



Com o módulo de comunicação I2C, este é um módulo de visualização que pode mostrar 2 linhas com 16 caracteres por linha. Apresenta um fundo azul e uma palavra branca e liga-se à interface I2C do MCU, o que permite poupar muitos

recursos do MCU. Na parte de trás do ecrã LCD, existe um potenciómetro azul para ajustar a luz de fundo. O endereço de comunicação predefinido é 0x27. O LCD 1602 original pode arrancar e funcionar com 7 portas IO, mas o nosso é construído com o interface Arduino IIC/I2C, poupando 5 portas IO. Em alternativa, o módulo vem com 4 orifícios de posicionamento com um diâmetro de 3 mm, o que é conveniente para fixar noutros dispositivos.

Repare que, quando o ecrã fica mais claro ou mais escuro, os caracteres tornam-se mais ou menos visíveis.

Especificações:

- Endereço I2C: 0x27
- Luz de fundo (azul, branca)
- Tensão de alimentação: 5V
- Contraste ajustável
- GND: Um pino que se liga à terra
- VCC: Um pino que se liga a uma fonte de alimentação de +5V
- SDA: Um pino que se conecta à porta analógica A4 para comunicação IIC
- SCL: Um pino que se conecta à porta analógica A5 para comunicação IIC

Equipamento:






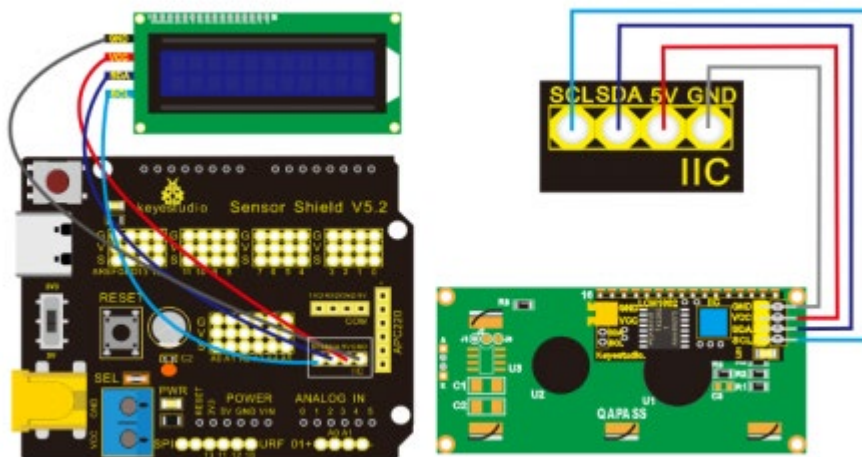
PLUS control board*1	Sensor shield*1	1602 LCD Display*1	USB cable*1	4pinF-F Dupont line*1
				

Diagrama de ligação:



Nota: existem pinos GND, VCC, SDA e SCL no módulo 1602LCD. O GND está ligado ao GND (da comunicação IIC), o VCC está ligado a 5V (+), SDA a SDA, SCL a SCL.

Código de teste:

/*

Keystudio smart home Kit for Arduino

Project 12

1602 LCD

<http://www.keystudio.com>

```
*/
```

```
#include <Wire.h>
```

```
#include <LiquidCrystal_I2C.h>
```

```
LiquidCrystal_I2C lcd (0x27,16,2); // set the LCD address to 0x27 for a16 chars and
```

```
2 line display
```

```
void setup ()
```

```
{
```

```
  lcd.init (); // initialize the lcd
```

```
  lcd.init (); // Print a message to the LCD.
```

```
  lcd.backlight ();
```

```
  lcd.setCursor (3,0);
```

```
  lcd.print ("Hello, world!"); // LED print hello, world!
```

```
  lcd.setCursor (2,1);
```

```
  lcd.print ("keystudio!"); // LED print keystudio!
```

```
}
```

```
void loop ()
```

```
{
```

```
}
```

```
//
```

Resultado do teste: :

Após a ligação e o carregamento do código de amostra, a primeira linha no LCD imprime "Hello, world!", a segunda linha imprime "keyestudio!", com um potenciômetro para ajustar a luz de fundo do LCD.



Nota: Ligar de acordo com o diagrama de ligação, carregar o código e, depois de ligar, quando o visor não mostrar caracteres, pode ajustar o potenciômetro atrás do 1602LCD e a luz de fundo para fazer com que o 1602LCD mostre a cadeia de caracteres correspondente.



Projeto 13: Sensor de humidade do solo

Descrição



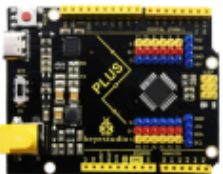






Este é um sensor simples de humidade do solo que visa detetar a humidade do solo. Se o solo estiver com falta de água, o valor analógico emitido pelo sensor irá diminuir; caso contrário, irá aumentar. Se utilizar este sensor para criar um dispositivo de rega automático, pode detetar se a sua botânica está com sede para evitar que murche quando sair. A utilização do sensor com o controlador Arduino torna a sua planta mais confortável e o seu jardim mais inteligente. O

módulo de sensor de humidade do solo não é tão complicado como possa pensar, e se precisar de detetar o solo no seu projeto, será a sua melhor escolha. O sensor é configurado com duas sondas inseridas no solo e, em seguida, com a corrente a atravessar o solo, o sensor obtém o valor da resistência através da leitura das alterações de corrente entre as duas sondas e converte esse valor de resistência em teor de humidade. Quanto maior for a humidade (menor resistência), maior é a condutividade do solo. Insira-o no solo e, em seguida, utilize o conversor AD para o ler. Com a ajuda deste sensor, a planta pode lembrar-se de si: Preciso de água.

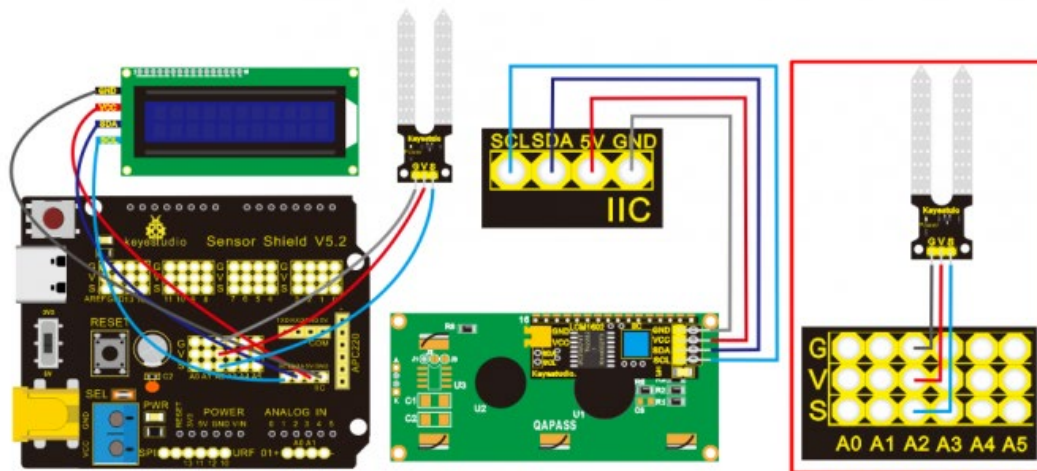
Especificação

- Tensão de alimentação: 3,3V ou 5V
- Corrente de trabalho: $\leq 20\text{mA}$
- Tensão de saída: 0-2,3V (quando o sensor está totalmente imerso em água, a tensão será de 2,3V) quanto maior a humidade, maior a tensão de saída
- Tipo de sensor: Saída analógica
- Definição da interface: S- sinal, G- GND, V - VCC
- Embalagem: Saco de selagem eletrostática
- Tamanho: 63 * 20 * 8mm
- Peso: 2,5g

Equipamento

PLUS control board*1	Sensor shield*1	Soil humidity sensor*1	1602 LCD display*1
			
USB cable*1	4pinF-F Dupont line*1	3pinF-F Dupont line*1	
			

Esquema de ligação



Nota: Na blindagem, os pinos G, V e S do sensor de humidade do solo estão ligados a G, V e A2; o GND do 1602LCD está ligado ao GND da comunicação ICC, o VCC está ligado a 5V (+) , SDA a SDA, SCL a SCL.

Código de teste:

```
/*  
  
Keystudio smart home Kit for Arduino  
  
Project 13  
  
Soil Humidity  
  
http://www.keyestudio.com  
  
*/  
  
#include <Wire.h>  
  
#include <LiquidCrystal_I2C.h>  
  
volatile int value;  
  
LiquidCrystal_I2C mylcd (0x27,16,2); // set the LCD address to 0x27 for a16 chars  
and 2 line display  
  
void setup () {  
  
    Serial.begin (9600); // Set the serial port baud rate to 9600  
  
    value = 0;  
  
    mylcd.init ();  
  
    mylcd.backlight (); // Light up the backlight  
  
    mylcd.clear (); // Clear the screen  
  
    Serial.begin (9600); // Set the serial port baud rate to 9600  
  
    pinMode (A2, INPUT); // Soil sensor is at A2, the mode is input  
  
}  
  
void loop () {  
  
    Serial.print ("Soil moisture value:"); // Print the value of soil moisture
```

```
Serial.print ("");

Serial.println (value);

delay (500); // Delay 0.5S

value = analogRead (A2); // Read the value of the soil sensor

if (value <300) // If the value is less than 300
{
    mylcd.clear (); // clear screen

    mylcd.setCursor (0, 0);

    mylcd.print ("value:"); //

    mylcd.setCursor (6, 0);

    mylcd.print (value);

    mylcd.setCursor (0, 1);

    mylcd.print ("dry soil"); // LCD screen print dry soil

    delay (300); // Delay 0.3S

}

else if ((value>=300) && (value <= 700)) // If the value is greater than 300 and
less than 700
{
    mylcd.clear (); //clear screen

    mylcd.setCursor (0, 0);

    mylcd.print ("value:");

    mylcd.setCursor (6, 0);
```

```
mylcd.print (value);

mylcd.setCursor (0, 1);

mylcd.print ("humid soil"); // LCD screen printing humid soil

delay (300); // Delay 0.3S

} else if (value > 700) // If the value is greater than 700
{
    mylcd.clear (); // clear screen

    mylcd.setCursor (0, 0);

    mylcd.print ("value:");

    mylcd.setCursor (6, 0);

    mylcd.print (value);

    mylcd.setCursor (0, 1);

    mylcd.print ("in water"); // LCD screen printing in water

    delay (300); // Delay 0.3S

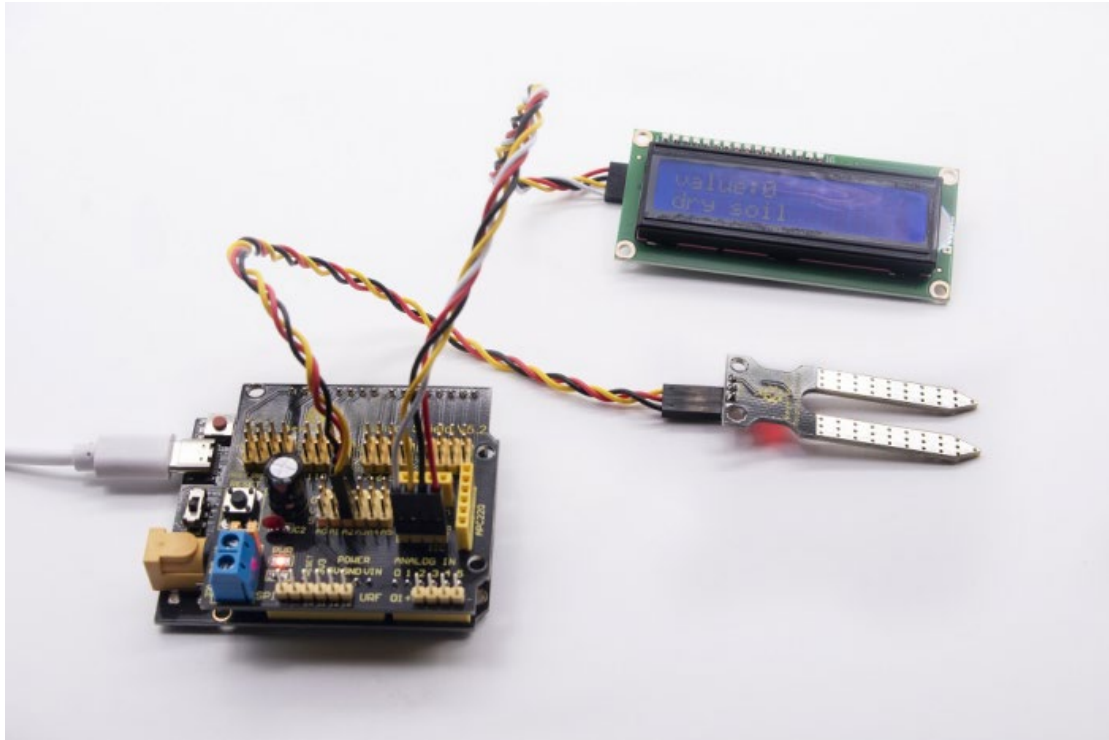
}}

//
```

Resultado do teste: :

Ligar de acordo com o diagrama de cablagem, gravar o programa e ligar. Abrir o monitor de série e inserir o sensor de humidade do solo no solo. Quanto maior for a humidade, maior será o número, na gama de 0-1023. O sensor de solo é

inserido no solo e na água com humidade diferente, e o 1602LCD apresenta o valor correspondente.



Projeto 14: Teste Bluetooth

No século XX, a tecnologia mudou a nossa vida. As pessoas podem trabalhar em casa com dispositivos sem fios, como o rato, os auscultadores, a impressora e o altifalante, o que melhora muito o nosso nível de vida. O Bluetooth pode facilitar o trabalho em casa, bem como o entretenimento. Os utilizadores podem controlar sem fios o ficheiro áudio do PC ou do Apple iPod num raio de 30 polegadas. A tecnologia Bluetooth também pode ser utilizada em adaptadores, permitindo que

as pessoas partilhem a sua vida quotidiana com amigos da Internet e das redes sociais.

Controlo remoto Bluetooth



A tecnologia Bluetooth é uma tecnologia padrão sem fios que permite a troca de dados a curta distância entre dispositivos fixos, dispositivos móveis e redes de área pessoal de edifícios (utilizando ondas de rádio UHF na banda ISM de 2,4 a 2,485 GHz). Este kit está equipado com o módulo Bluetooth HM-10, que é uma máquina mestre-escravo. Quando usado como anfitrião, pode enviar comandos para o escravo ativamente; quando usado como escravo, só pode receber comandos do anfitrião. O módulo Bluetooth HM-10 suporta o protocolo Bluetooth 4.0, que não só é compatível com o telemóvel Android, mas também com o sistema iOS. Na experiência, utilizamos por defeito o módulo Bluetooth HM-10 como Slave e o telemóvel como Host. Instalamos a aplicação Bluetooth no telemóvel, ligando o módulo Bluetooth; finalmente, usamos a aplicação Bluetooth para controlar as partes do kit de casa inteligente. Também fornecemos 2 tipos de APP para telemóvel, para o sistema Android e iOS.

Parâmetros do módulo Bluetooth HM-10:

Pins	Description
BRK	<p>As input pin, short press control, or input single pulse of 100ms low level to achieve the following functions:</p> <ol style="list-style-type: none">1. When module is in sleep state: Module is activated to normal state, if open AT+NOTI, serial port will send OK+WAKE.2. When in connected state: Module will actively request to disconnect <p>When in standby mode: Module will be in initial state</p>
RXD	Serial data inputs
TXD	Serial data outputs
GND	ground lead
VCC	Positive pole of power, input 5V
STATE	<p>As output pin, show the working state of module</p> <p>Flash slowly in standby state——repeat 500ms pulse;</p> <p>Always light in connected state——high level</p> <p>You could set to no flashing in standby state, always light in connected state</p>

- Protocolo Bluetooth: Especificação Bluetooth V4.0 BLE

- Sem limite de bytes na porta de série Transceção
- Em ambiente aberto, comunicação de ultra-distância de 100m com o iphone4s
- Protocolo USB: USB V2.0
- Frequência de funcionamento: banda ISM de 2,4 GHz
- Método de modulação: GFSK (chaveamento de mudança de frequência gaussiana)
- Potência de transmissão: -23dbm, -6dbm, 0dbm, 6dbm, pode ser modificada pelo comando AT.
- Sensibilidade: $\leq -84\text{dBm}$ a 0,1% BER
- Taxa de transmissão: Assíncrona: 6K bytes; Síncrona: 6k Bytes
- Elemento de segurança: Autenticação e encriptação
- Serviço de suporte: UUID central e periférico FFE0, FFE1
- Consumo de energia: Modo de espera automático, corrente de espera 400uA~800uA, 8,5mA durante a transmissão.
- Fonte de alimentação: 5V DC
- Temperatura de funcionamento: -5 a +65 graus centígrados

Utilizar a APP Bluetooth

Descrição:

Na lição anterior, apresentámos o princípio básico dos parâmetros do módulo Bluetooth HM-10. Neste projeto, vamos mostrar-lhe como utilizar o módulo HM-10 Bluetooth. Para controlar eficazmente este kit através do módulo Bluetooth HM-10, concebemos especialmente uma APP, como se mostra abaixo.



Existem doze botões de controlo e quatro cursores na aplicação. Quando ligamos o módulo Bluetooth HM-10 e a aplicação, basta premir o botão de controlo da aplicação e o Bluetooth do telemóvel envia um carácter de controlo. O módulo Bluetooth receberá um carácter de controlo correspondente. Ao programar, definimos a função correspondente de cada sensor ou módulo de acordo com o carácter de controlo da tecla correspondente. De seguida, vamos testar 12 botões na aplicação.


APP para telemóvel Android:

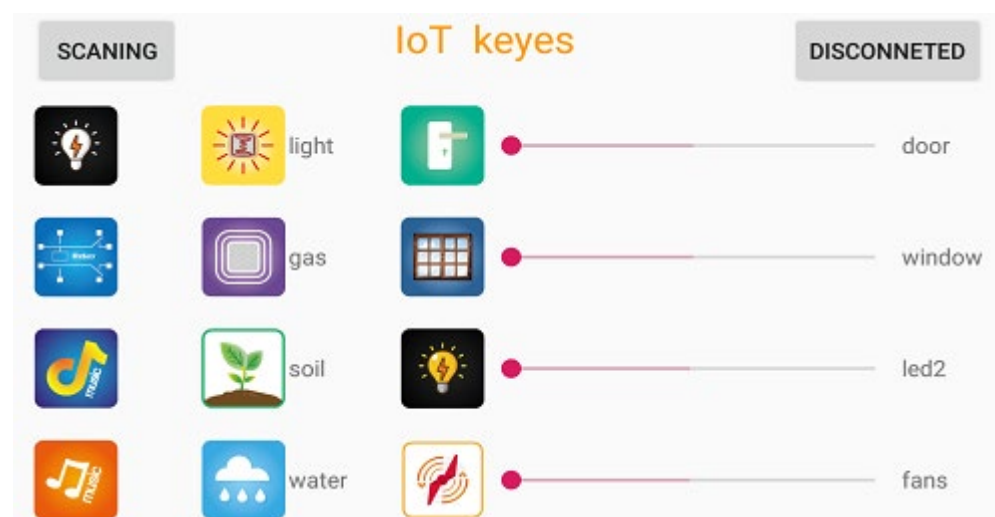
Nota: Permitir que a APP acesse a "localização" nas definições do seu telemóvel quando ligar ao módulo Bluetooth, caso contrário, o Bluetooth pode não ser ligado.

Entre no google play, pesquise "keyes IoT", se você não pode pesquisá-lo na loja de aplicativos, faça o download do aplicativo no seguinte link:

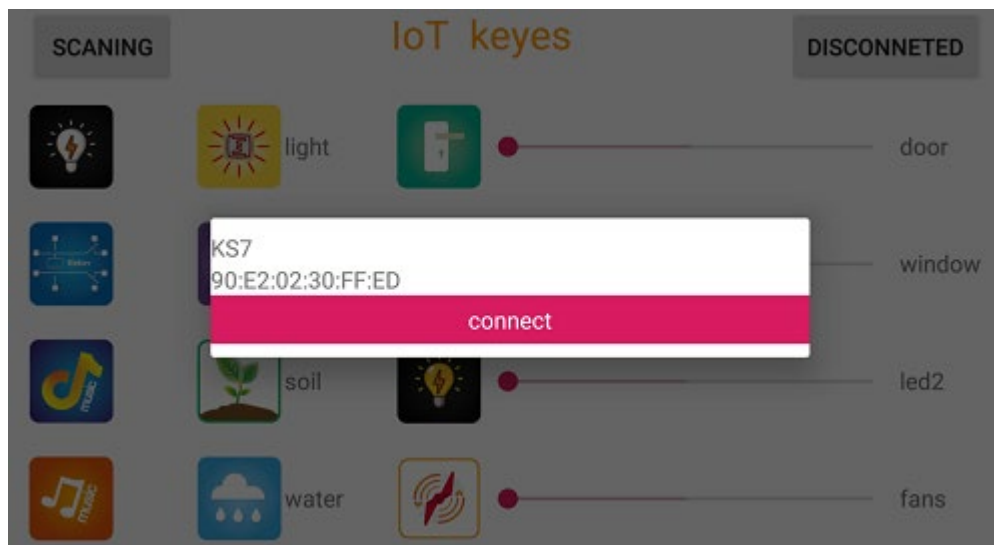
https://play.google.com/store/apps/details?id=com.keyestudio.iot_keys



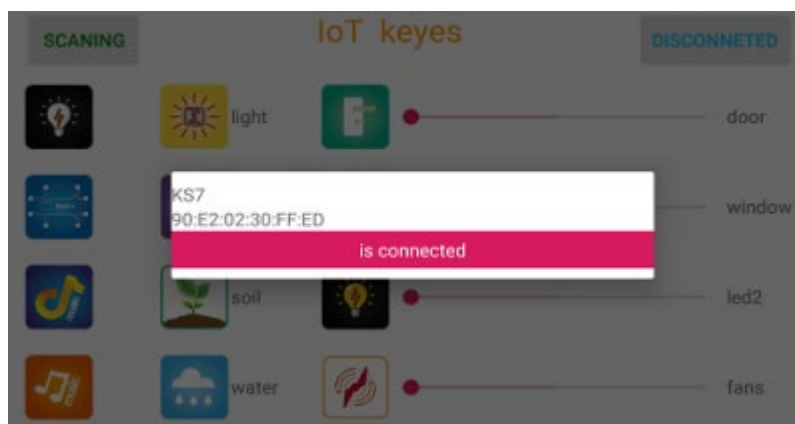
Depois de instalar e abrir a aplicação , a interface é apresentada da seguinte forma:



Carregar o código e ligar, o LED do módulo Bluetooth fica intermitente. Iniciar o Bluetooth e abrir a aplicação para clicar em "CONNECT" (ligar) para ligar.



Clique em "Ligar" e o Bluetooth é ligado com êxito. Como mostrado abaixo, o LED do módulo Bluetooth está normalmente ligado.

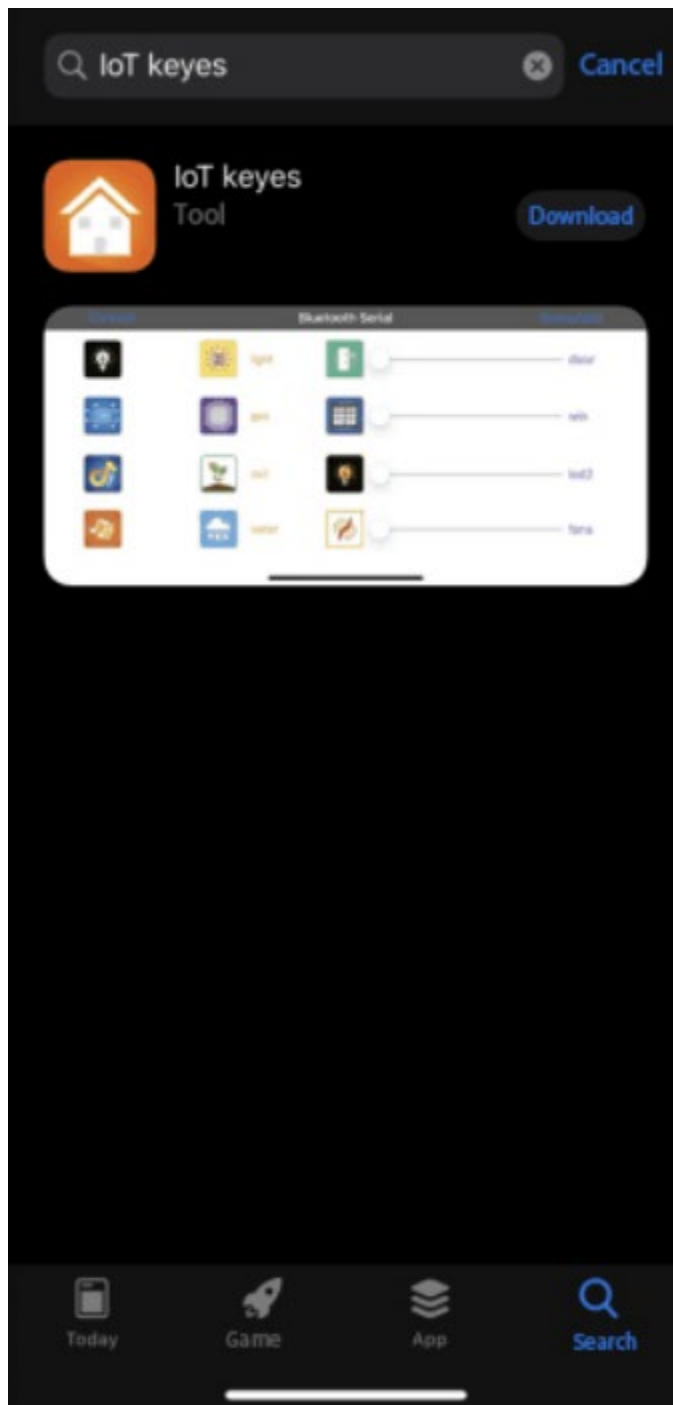


For IOS system:




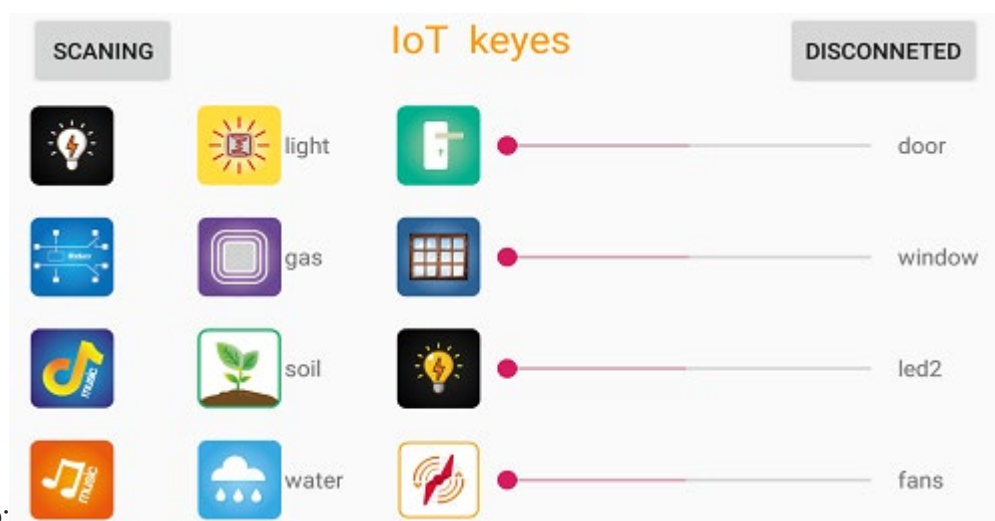
(1) Abrir a App Store

(2) Pesquise "IoT keys" na APP Store, depois clique em "downlaod".





(3) Depois de instalar com sucesso e abrir , a interface é mostrada



abaixo:

(4) Carregue o Código de teste com êxito, insira o módulo Bluetooth e ligue-o. O

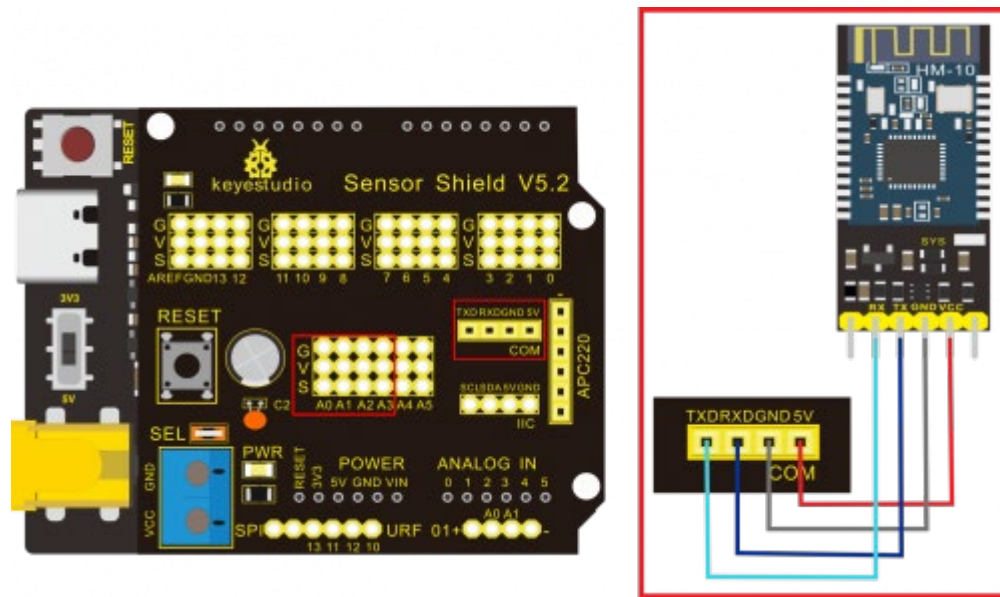
LED do módulo Bluetooth está a piscar. Iniciar o Bluetooth no telemóvel e, em seguida, clicar em "ligar" à esquerda para procurar o Bluetooth e emparelhar.

Após o emparelhamento bem sucedido, o LED do módulo Bluetooth acende-se.

Nota: Retire o módulo Bluetooth quando estiver a carregar o Código de teste.

Caso contrário, o programa não será carregado. Ligar o Bluetooth e o módulo Bluetooth para emparelhar depois de carregar o Código de teste.

4. diagrama de ligação



Nota: Na placa de expansão do sensor, os pinos RXD, TXD, GND e VCC do módulo Bluetooth são ligados respetivamente a TXD, RXD, GND e 5V, e os pinos STATE e BRK do módulo Bluetooth não precisam de ser ligados. Ligar a fonte de alimentação.

Código de teste:

/*

Keystudio smart home Kit for Arduino

Project 14

Bluetooth

<http://www.keyestudio.com>

*/

```
char val;

void setup()

{

Serial.begin(9600);// Set the serial port baud rate to 9600

}

void loop()

{

while (Serial.available(>0)

{

val=Serial.read();// Read the value sent by Bluetooth

Serial.print(val);// The serial port prints the read value

}









}

//
```




Função-chave na aplicação:

No.	Button	Control Character	Function	No	Button	Control Character	Function
1	SCANNING	Pair and connect to HM-10 Bluetooth module		2	DISCONNECT	Disconnect Bluetooth	
3		Click to send "a", click again to send "b"	Click to turn on white LED, click again to turn off LED	4		Click to send "c", click again to send "d"	Click to turn on relay module; click again to turn off relay module
5		Hold and press to send "e" release to send "g"	Click to play music	6		Hold and press to send "f" release to send "g"	Click to play music (alternative song)
7		Click to send "h", click again to send "s"	Click to turn on photocell sensor, light shows the data; click again to turn off photocell sensor	8		Click to send "i" click again to send "s"	Click to turn on gas sensor, gas displays the detected data; click again to turn off gas sensor
9		Click to send "j" click again to send "s"	Click to turn on soil humidity sensor, soil shows data, click again to turn off soil humidity sensor	10		Click to send "k" click again to send "s"	Click to turn on steam sensor, water displays the detected data; click again to turn off steam sensor

11		Click to send "l" ; click again to send "m"	Click to open the door; click again to close the door	12		Drag slider to send "t 50 #", "t" represents initial character; 50 is the angle of servo 1 ; '#'implies termination character	Slider controls the angle of servo 1 to rule the door, door displays the angle value of servo 1
13		Click to send "n"; click again to send "o"	Click to open the window; click again to close the window	14		Drag slider to send "u 34 #", "u" represents initial character; 34 is the angle of servo 2; '#' stands for termination character	Slider controls the angle of servo 2 to rule the window, win shows the angle value of servo 2
15		Click to send "p"; click again to send "q"	Click to turn on LED; click again to turn off LED	16		Drag slider to send "v 100 #", "v" represents initial character; 100 is the PWM value of led2; '#' stands for termination character	Slider controls LED brightness, led2 displays brightness value
17		Click to send "r"; click again to send "s"	Click to turn on fan; click again to turn off fan	18		Drag slider to send "w 153 #", "w" represents initial character; 153 is the PWM value of fan ; '#'stands for termination character	Slider controls rotation speed, fans indicates the rotation speed value

Guia montado

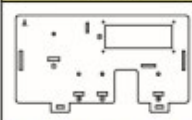










Verificar em primeiro lugar o quadro A~I e as peças

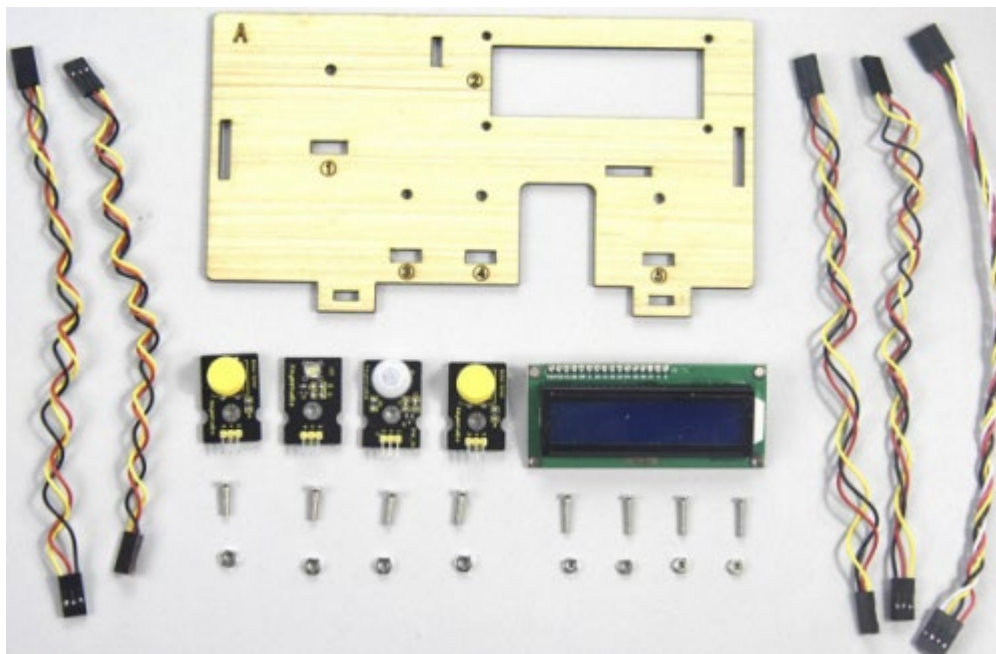


Passo 1: Instalar os sensores da placa A

Prepare a placa A * 1, parafuso redondo M3 * 10MM * 4, porca niquelada M3 * 4, parafuso redondo M2.5 * 10MM * 4, sensor de botão * 2, LED branco * 1, sensor

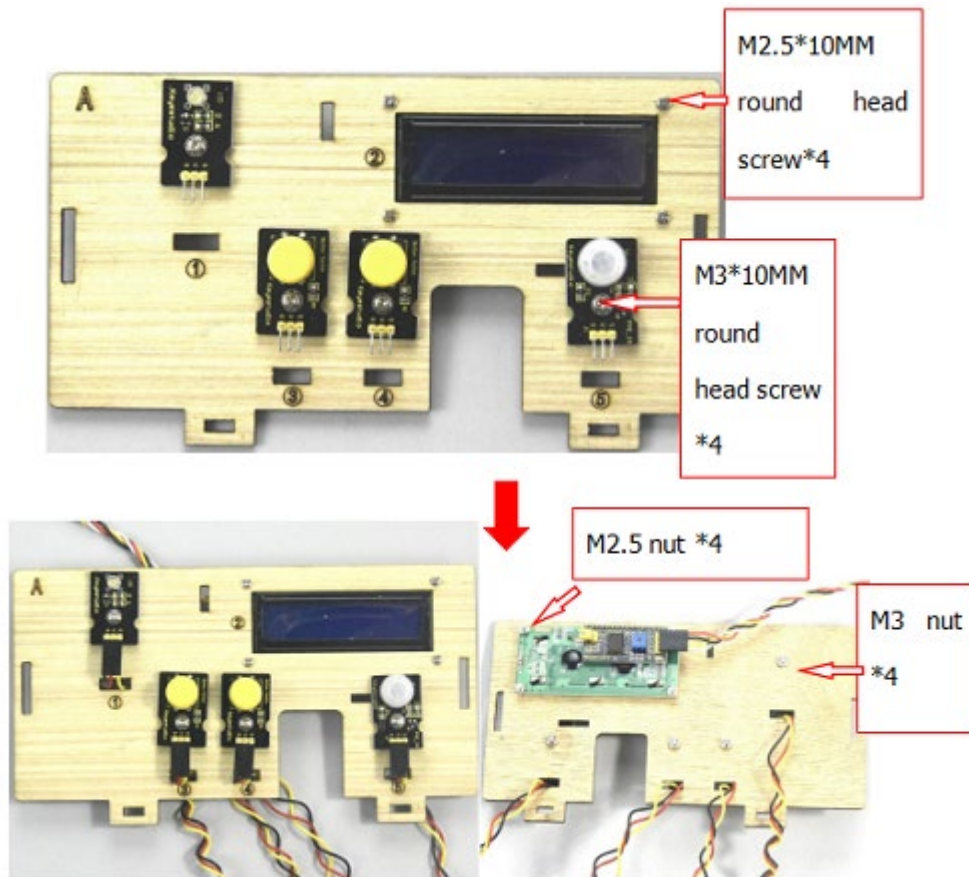
de movimento PIR * 1, display LCD1602 * 1, linha dupont F-F de 4 pinos * 1, linha dupont F-F de 3 pinos * 4

A Board*1	Button module*2	White LED*1	PIR motion sensor*1	LCD1602 Display*1	4pin F-F Dupont line*1
					
M2.5 Nickel plated nut*4	M3 Nickel plated nut*4	M2.5*10MM Round head screw*4	M3*10MM Round head screw*4	3pin F-F Dupont line*4	
					








- a.Fixar o LED branco, os 2 sensores de botão e o sensor de movimento PIR na área correspondente da placa A com 4 parafusos de cabeça redonda M3*10MM e 4 porcas M3.
- b.Em seguida, instale o ecrã LCD1602 na placa A com 4 parafusos de cabeça redonda M2.5*10MM e 4 porcas M2.5.

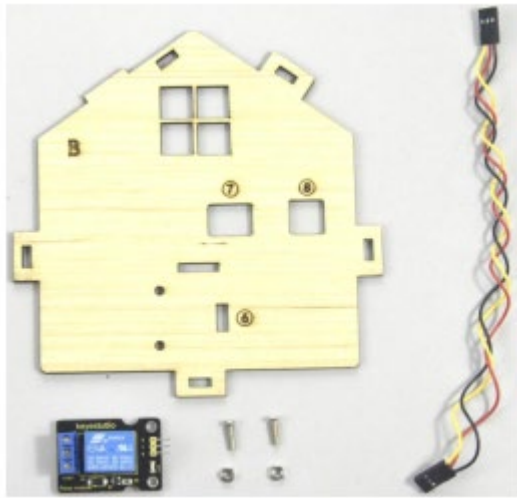
- c. Ligue-os com linhas dupont de 3 e 4 pinos.



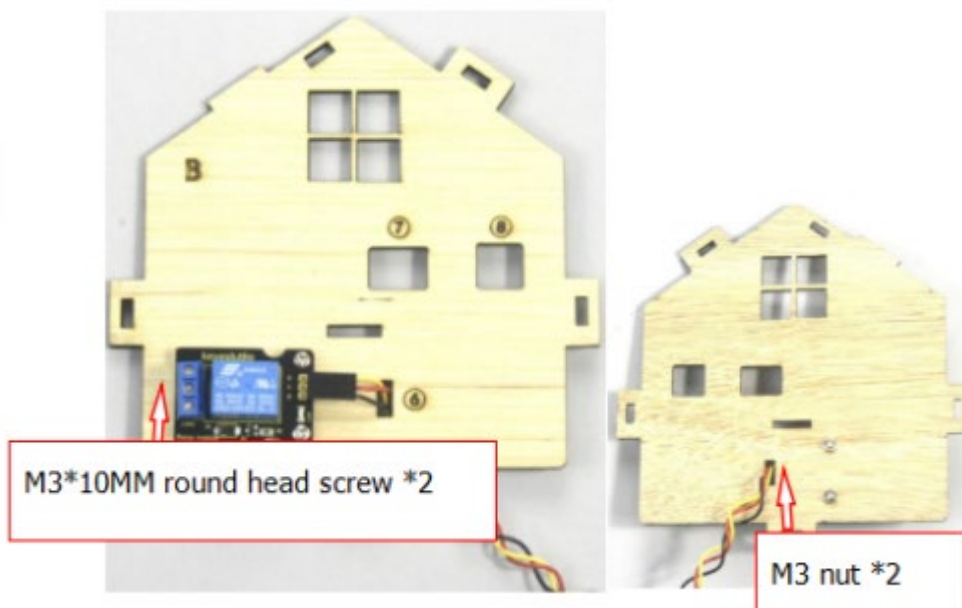
Passo 2: Instalar o sensor da placa B

Prepare uma placa B, uma linha F-F Dupont de 3 pinos, 2 parafusos de cabeça redonda M3 * 10MM, 2 porcas M3 niqueladas e um módulo de relé

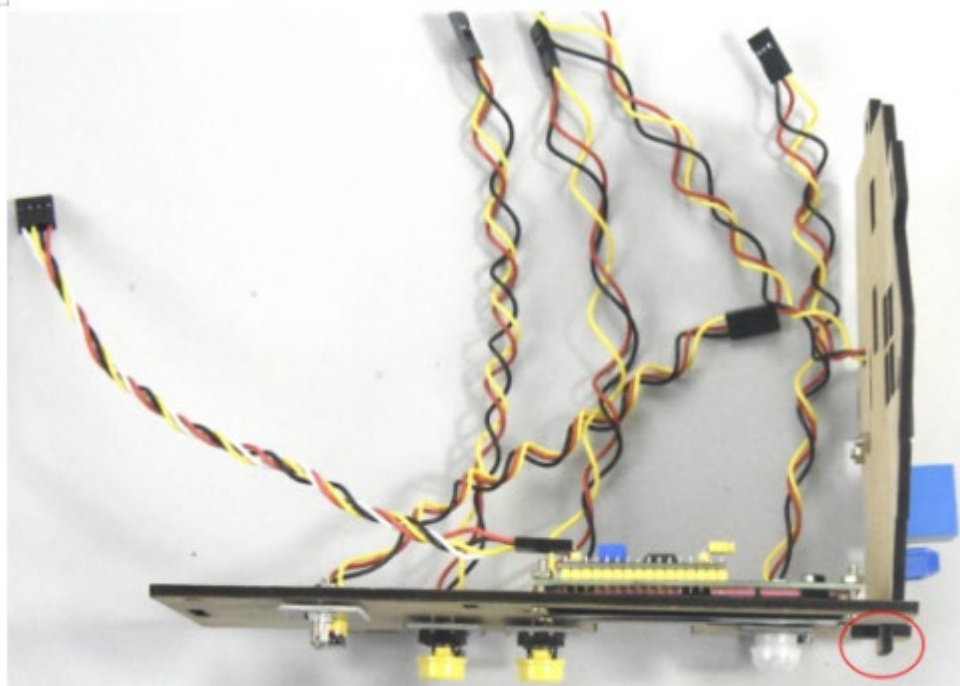
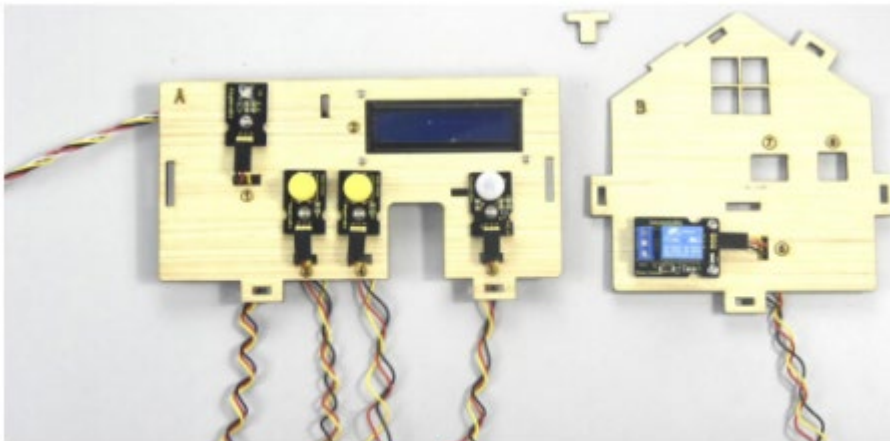
B Board*1	Relay module*1	M3 Nickel plated nut*2	M3*10MM Round head screw*2	3pin F-F Dupont line*1
				

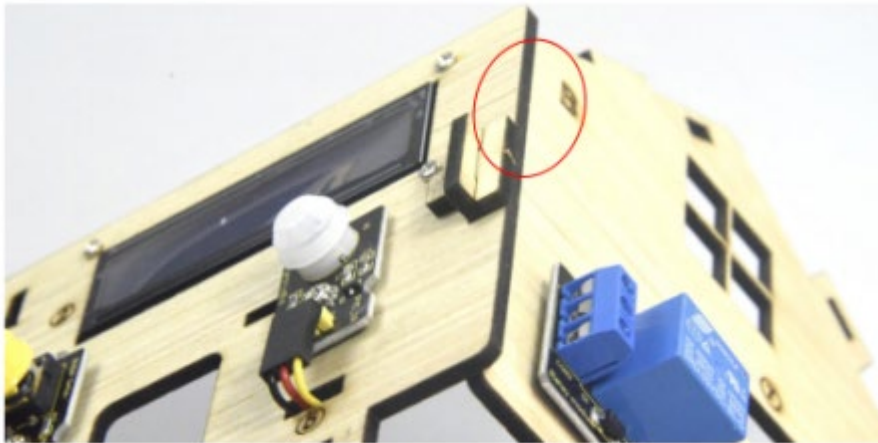


Montar o módulo de relé na placa B com 2 parafusos M3*10MM e 2 porcas M3, ligá-los entre si com uma linha dupont de 3 pinos




Etapa 3: Fixar a placa A e a placa B com um parafuso "T".

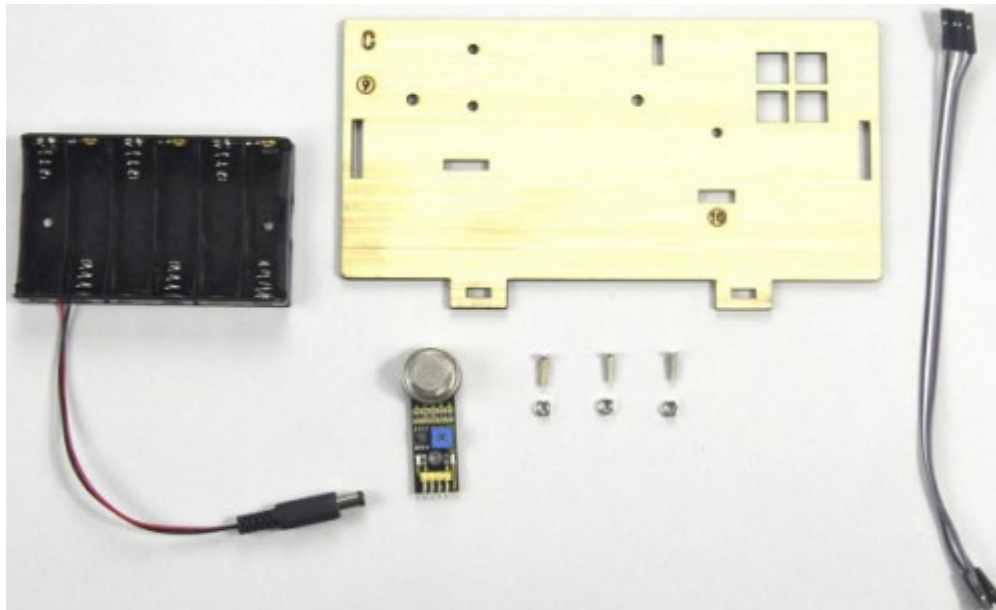




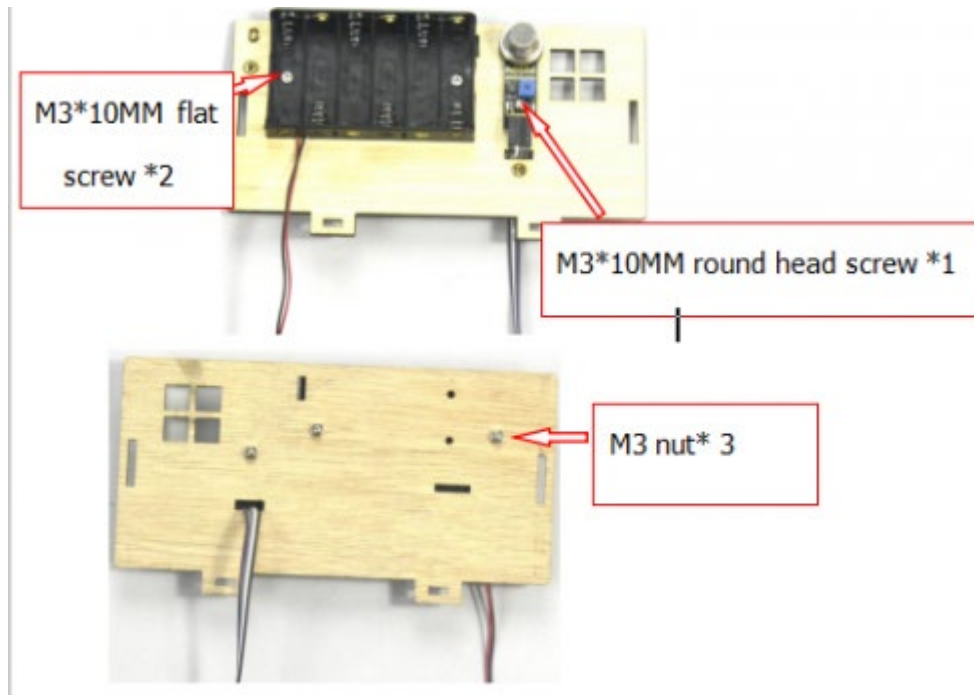
Passo 4: Montar os sensores e o suporte da bateria da placa C

Prepare uma placa C, sensor de gás MQ-2, suporte de bateria, 2 parafusos de cabeça chata M3 * 10MM, um parafuso de cabeça redonda M3 * 10MM, 3 porcas niqueladas M3 e 4 linhas dupont F-F.

C Board*1	MQ-2 Gas sensor*1	Battery holder*1	M3*10MM Flat head screw*2	M3*10MM Round head screw*1	M3 Nickel plated nut*3	F-F Dupont line*4
						



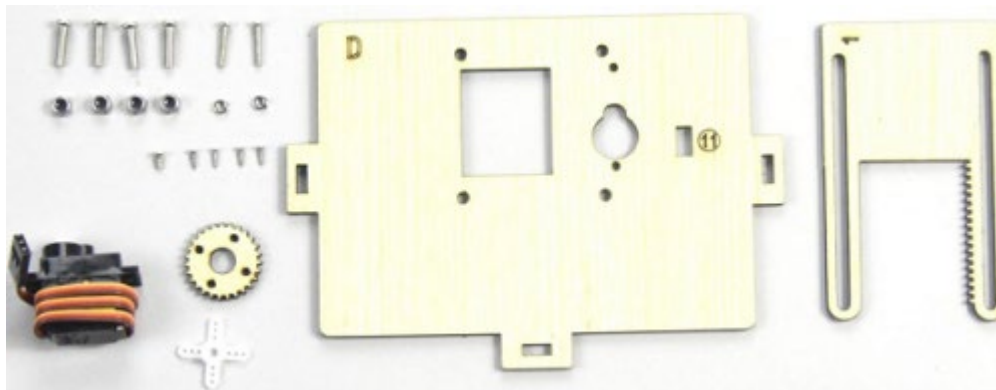
- A. Fixar o suporte da bateria na placa C com 2 parafusos de cabeça chata M3*10MM e 2 porcas M2
- B. Em seguida, instalar o sensor de gás MQ-2 na área correspondente da placa C com um parafuso de cabeça redonda M3*10MM e uma porca M3.
- C. Ligue-os com 4 linhas dupont fêmea a fêmea



Etapa 5: Instalar os sensores e as peças da placa D

Prepare um servo, 4pcs M1.2 * 5 parafusos auto-roscantes, a montagem em cruz branca (incluído no servo) , a M2 * 5 parafuso de cabeça redonda (incluído em servo) , 2pcs M2 * 12MM parafusos de cabeça redonda, 2pcs M2 porcas níqueladas, 4pcs M3 * 12MM parafusos de cabeça redonda, 4pcs M3 porcas de travamento automático inoxidável, a placa D, a engrenagem, uma placa1.

D Board*1	Board 1*1	Gear*1	Servo motor*1	White cross mount*1	M2*5 Round head screw*1
M2 Nickel plated nut*2	M3 Stainless self-locking nut*4	M3*12MM Round head screw*4	M2*12MM Round head screw*2	M1.2*5 Self-tapping screw*4	

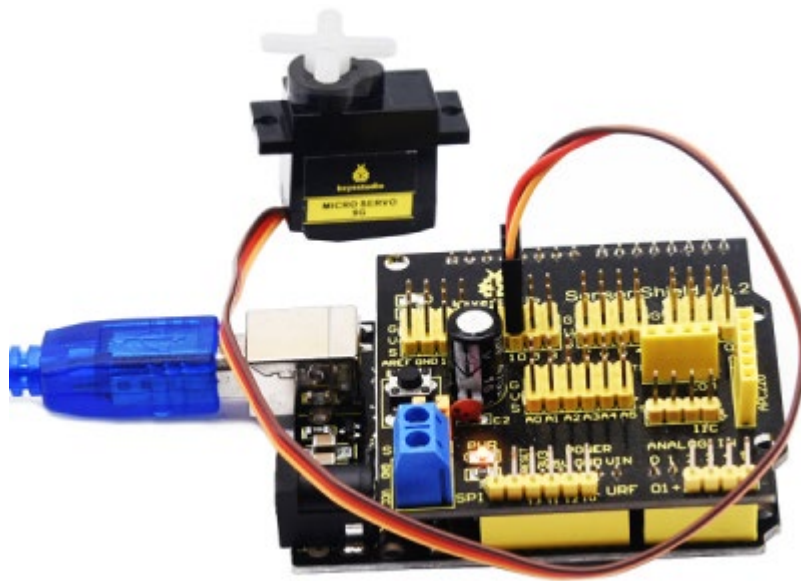


Rodar o servo a 90° antes da instalação, ligar o servo à placa de controlo

keystudio PLUS; carregar o Código de teste na placa de controlo e fazer o servo

rodar a 90°

Servo Motor	
Brown wire	GND
Red wire	5V
Orange wire	S (10)



Código de teste:

```
#include <Servo.h>

Servo servo_10;

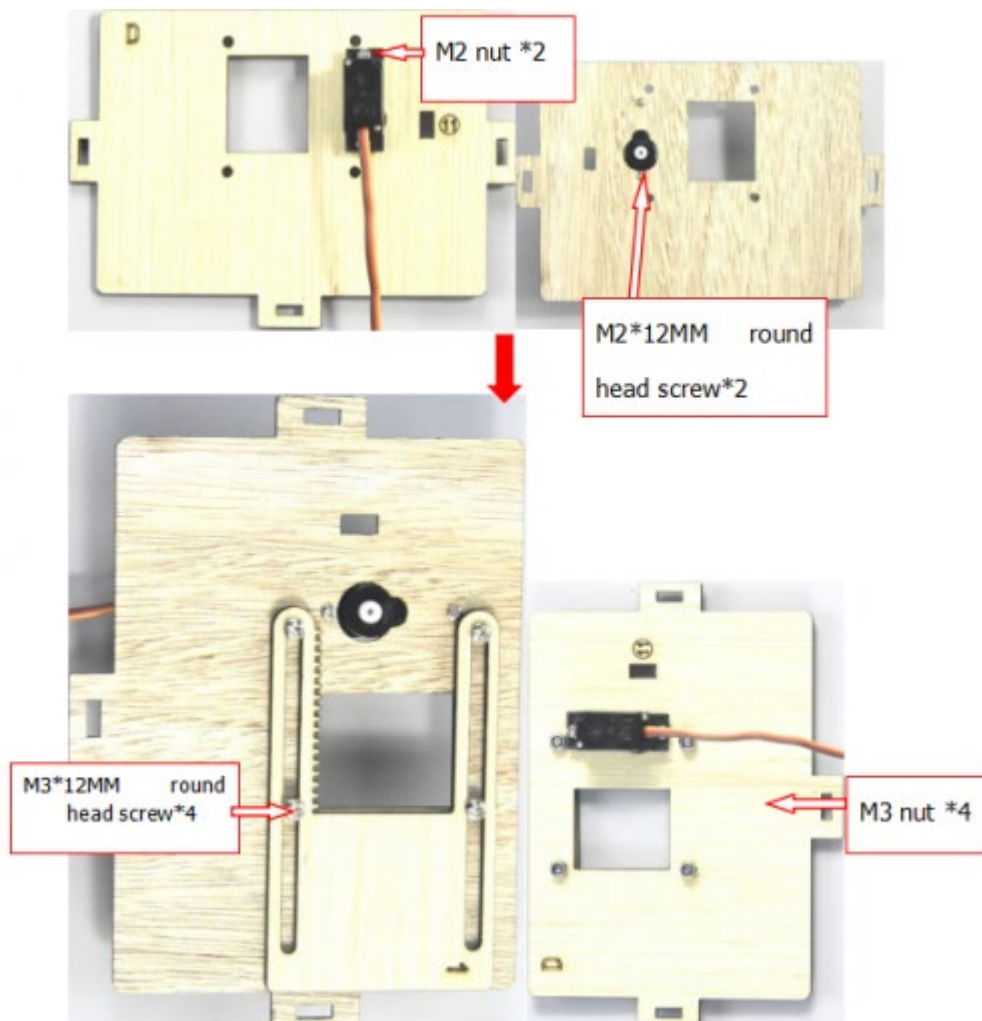
void setup(){

  servo_10.attach(10);
```

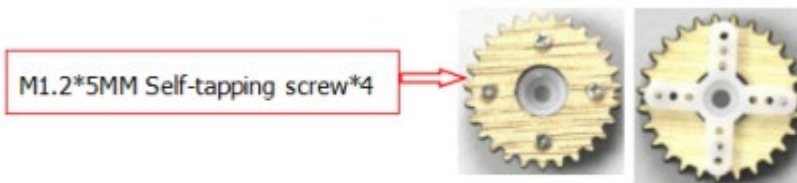
```
}  
  
void loop(){  
  
  servo_10.write(90);  
  
  delay(500);}
```

Carregar o Código de teste com êxito, o servo roda para 90°

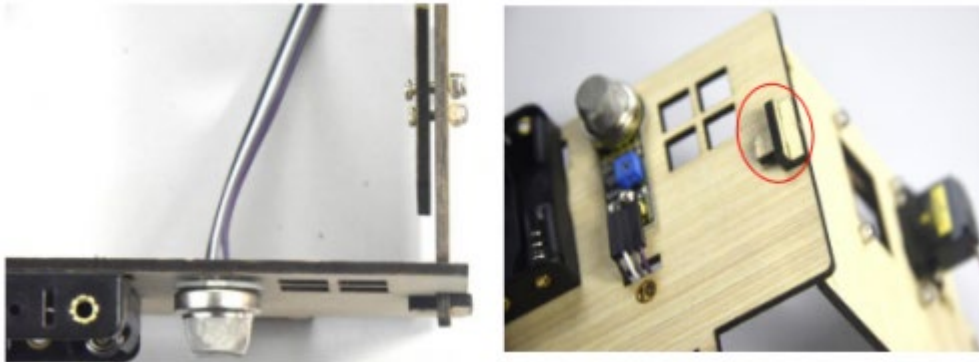
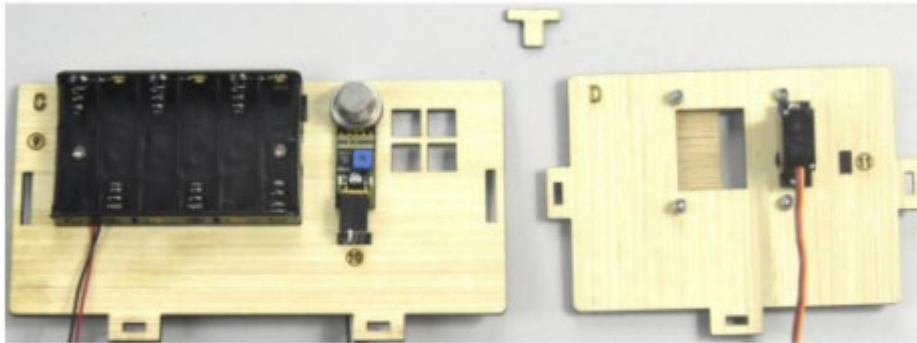
- A. Fixar o servo na área correspondente da placa D com 2 parafusos de cabeça redonda M2*12MM e 2 porcas M2.
- B. Em seguida, instalar a placa quadrada 1 na placa D com 4 parafusos de cabeça redonda M3*12MM e 4 porcas autoblocantes M3.



Fixar a montagem em cruz branca na engrenagem com 4 parafusos auto-roscentes M1.2*5MM e montar a engrenagem no servo motor com 1 parafuso de cabeça redonda M2*5MM.



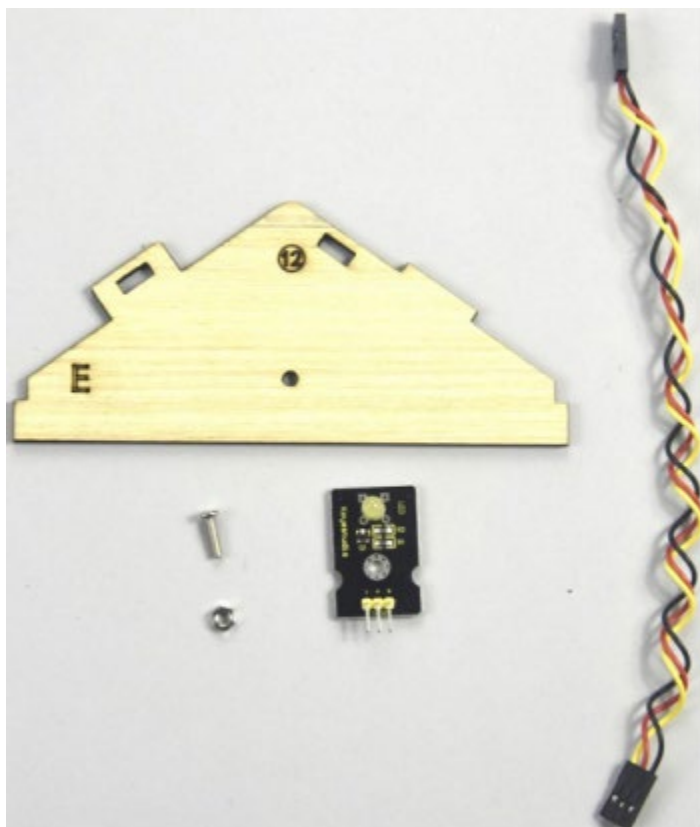
Etapa 6: Montar a placa C com a placa D através de um parafuso de tipo "T".



Etapa 7: instalar o sensor da placa E

Preparar um módulo LED amarelo, uma placa E, um parafuso de cabeça redonda M3*10MM, uma porca níquelada M3 e uma linha Dupont F-F de 3 pinos

E Board*1	Yellow LED*1	M3 Nickel plated nut*1	M3*10MM Round head screw*1	3pin F-F Dupont line*1

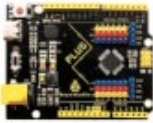

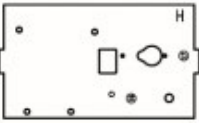
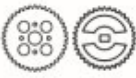
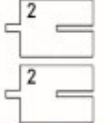













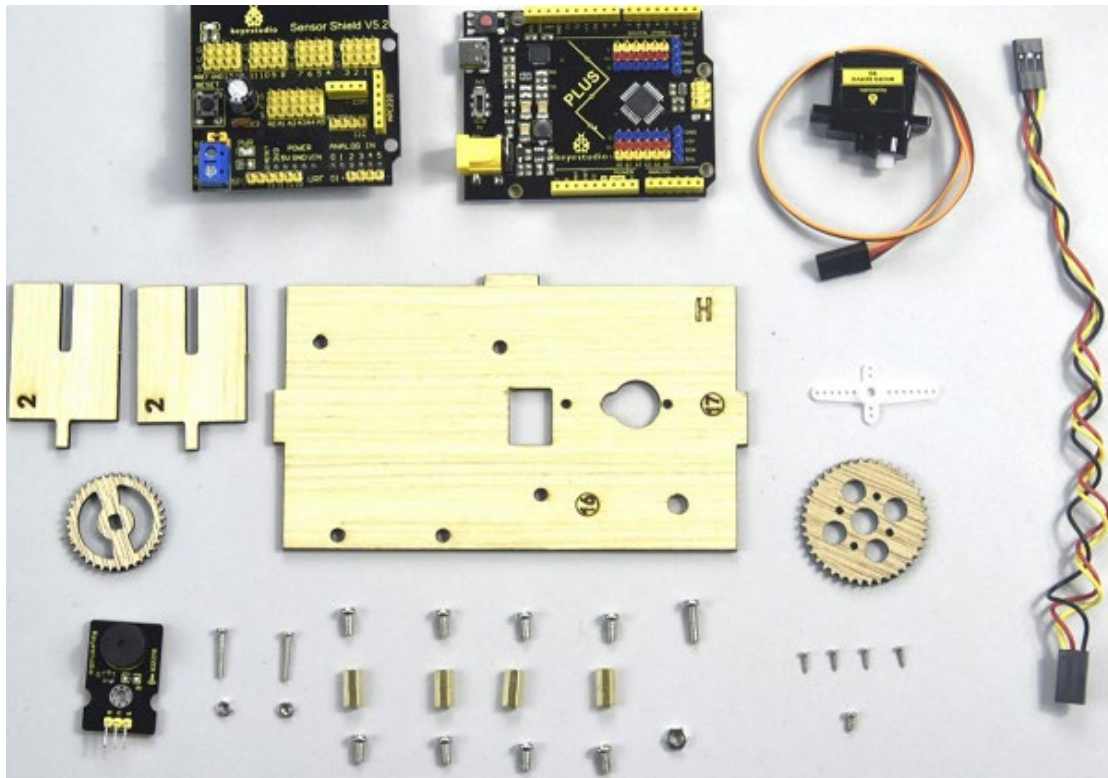
Monte o LED amarelo na área correspondente da placa E com 1 parafuso de cabeça redonda M3 * 10MM e 1 porca níquelada M3, em seguida, conecte com uma linha dupont de 3 pinos.



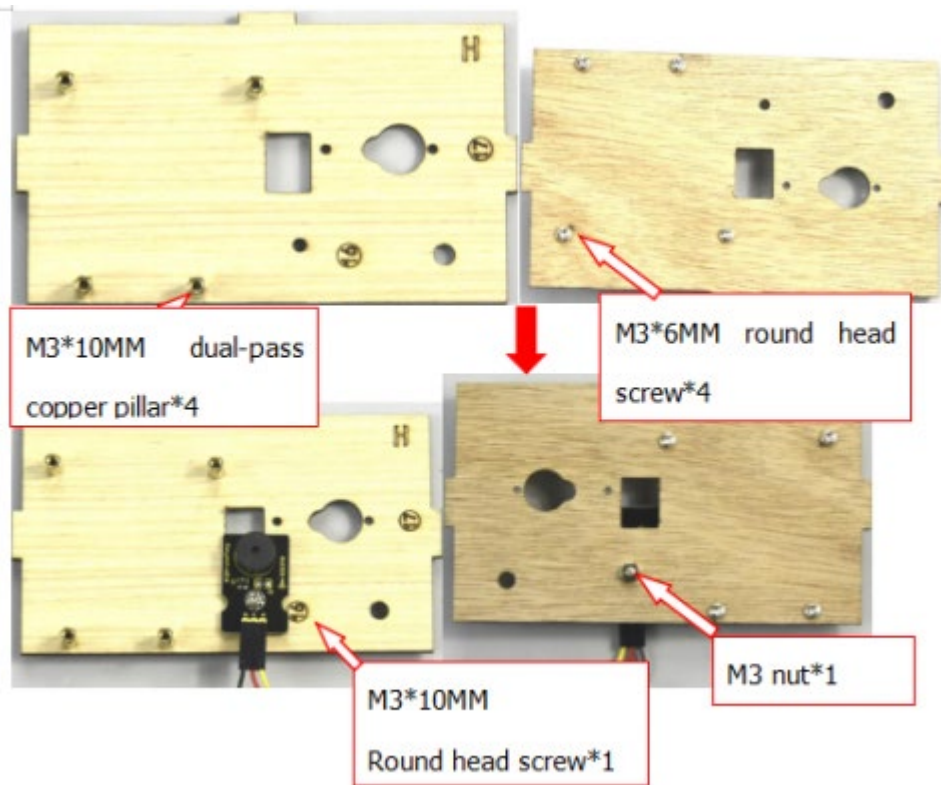
Etapa 8: Instalar a placa de controlo, os sensores e as peças da placa H

Preparar um servo, uma campainha passiva, 4pcs M1. 2 * 5 parafusos auto-roscentes, uma montagem em cruz branca (incluída no servo), um parafuso M2 * 5 (incluído no servo), 2pcs M2 * 12MM parafusos de cabeça redonda, 2pcs M2 porcas níqueladas, um parafuso redondo M3 * 10MM, um M3 porca níquelada, 8pcs M3 * 6MM parafusos de cabeça redonda, 4pcs M3 * 10MM pilares de cobre de passagem dupla, uma placa de controle Keyestudio PLUS, uma blindagem de sensor, uma linha Dupont 3pinF-F, uma placa E, 2 engrenagens e placa 2pcs 2.

Keyestudio PLUS control board 	Sensor shield 	H Board*1 	Gear*2 	Board 2*2 	M3*6MM Round head screw*8 
Servo motor*1 	Passive buzzer*1 	M2 Nickel plated nut*2 	M3 Nickel plated nut*1 	M3*10MM Round head screw*1 	M2*12MM Round head screw*2 
M1.2*5 Self-tapping screw*4 	White cross mount*1 	M2*5 screw*1 	M3*10MM Dual-pass pillar*4 	3pin F-F Dupont line*1 	

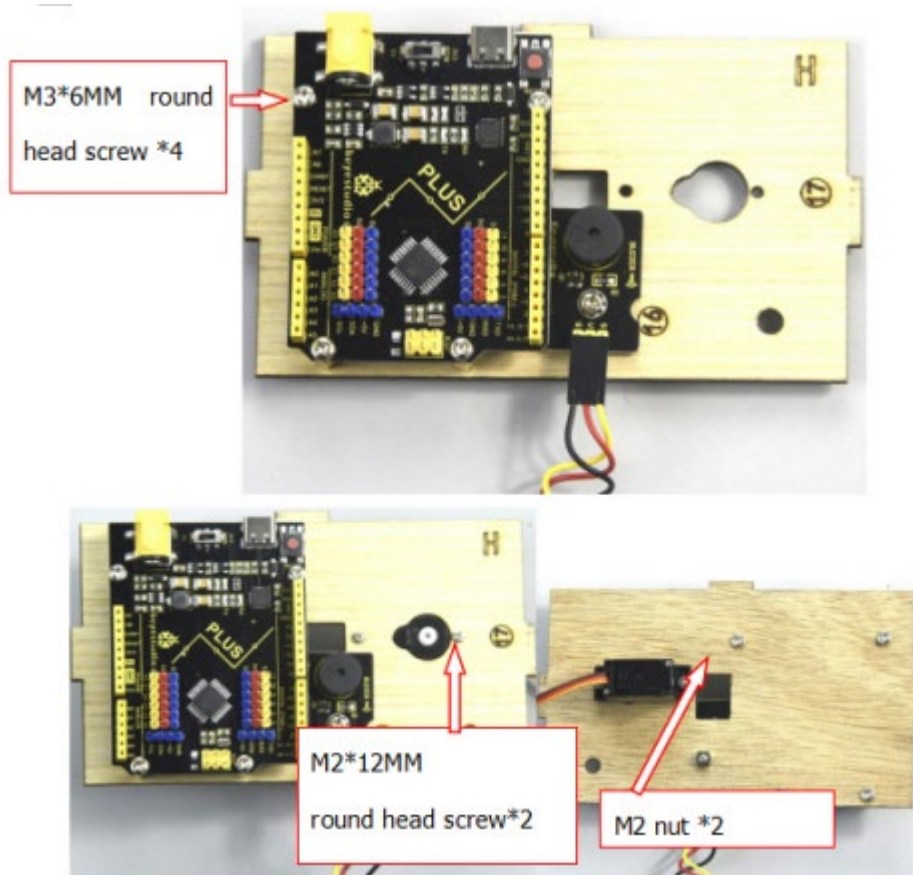


- A. Montar 4 pilares de cobre de dupla passagem na placa H com 4 parafusos M3*6MM
- B. Em seguida, fixar a campainha passiva na placa H com 1 parafuso de cabeça redonda M3*10MM e 1 porca MS.
- C. Ligá-los com um fio dupont de 3 pinos fêmea a fêmea

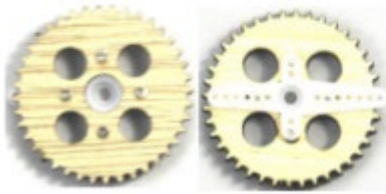


Rodar o servo para 90° antes da instalação, o método é o mesmo que o passo 6.

Fixar os pilares de cobre 4pcs M3*10MM na placa de controlo keystudio PLUS com 4 parafusos de cabeça redonda M3*6MM, depois fixar o servo na área correspondente da placa H com 2 parafusos de cabeça redonda M2*12MM e 2 porcas M2.

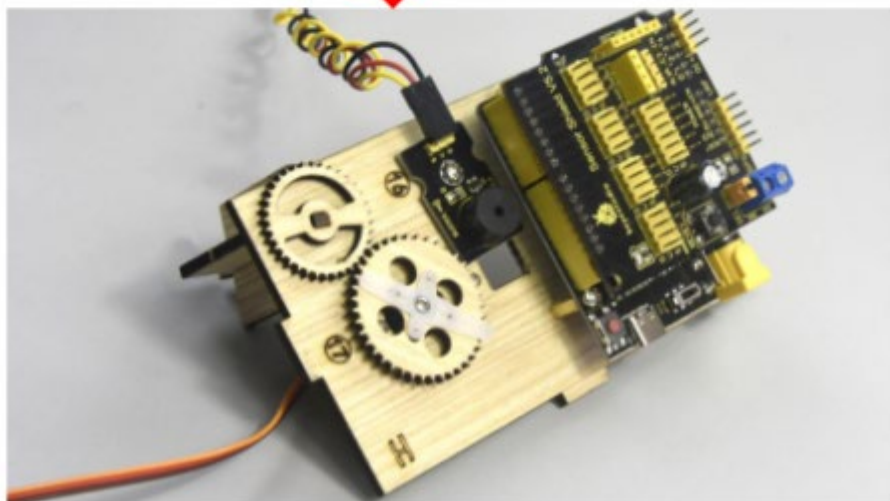
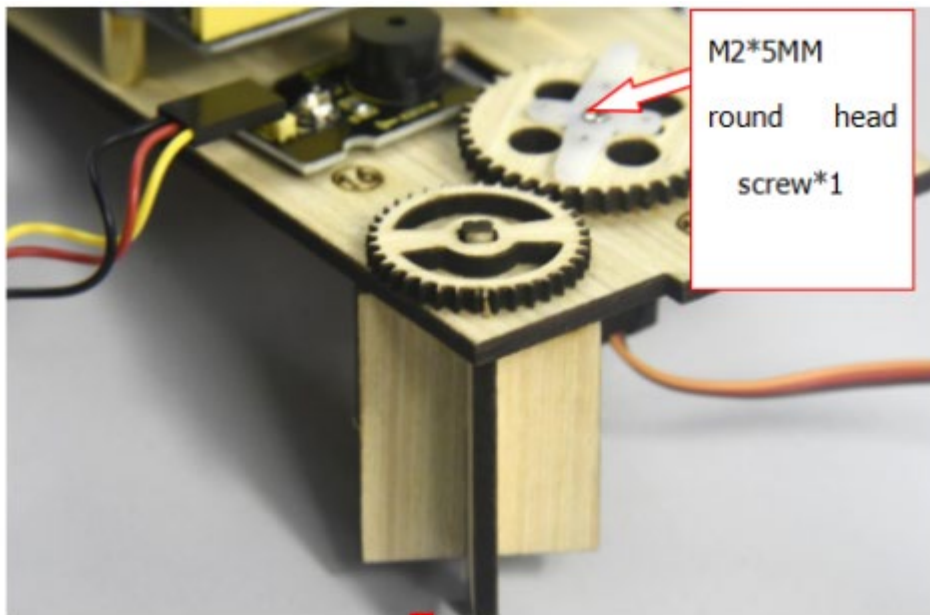


Montar a placa 2 de 2 peças em conjunto e, em seguida, fixar a montagem em cruz branca na engrenagem com 4 parafusos auto-roscentes M1.2*5



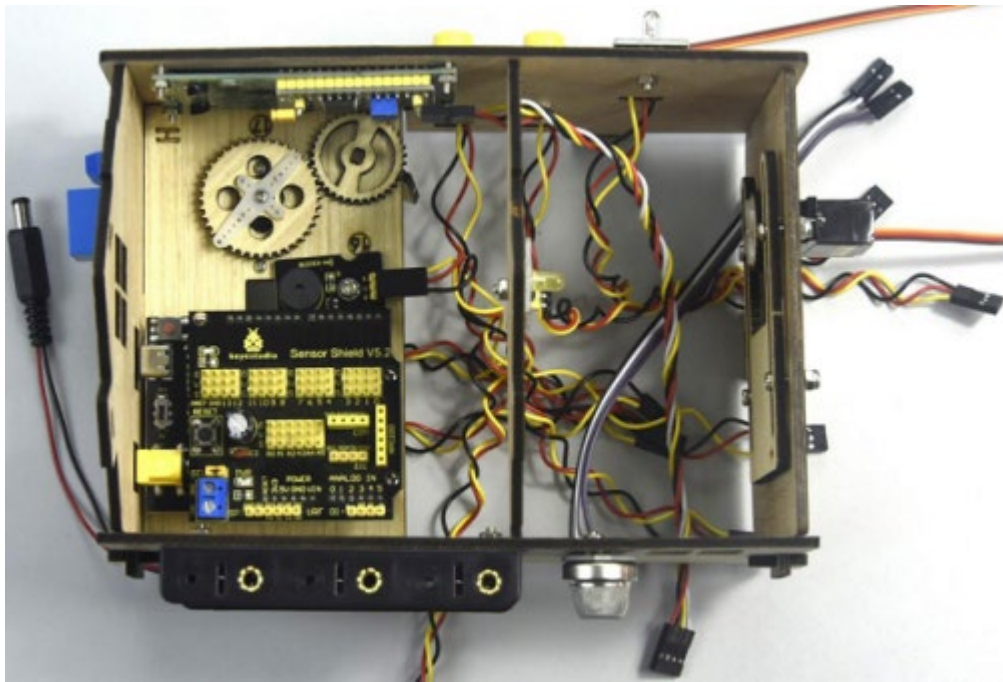
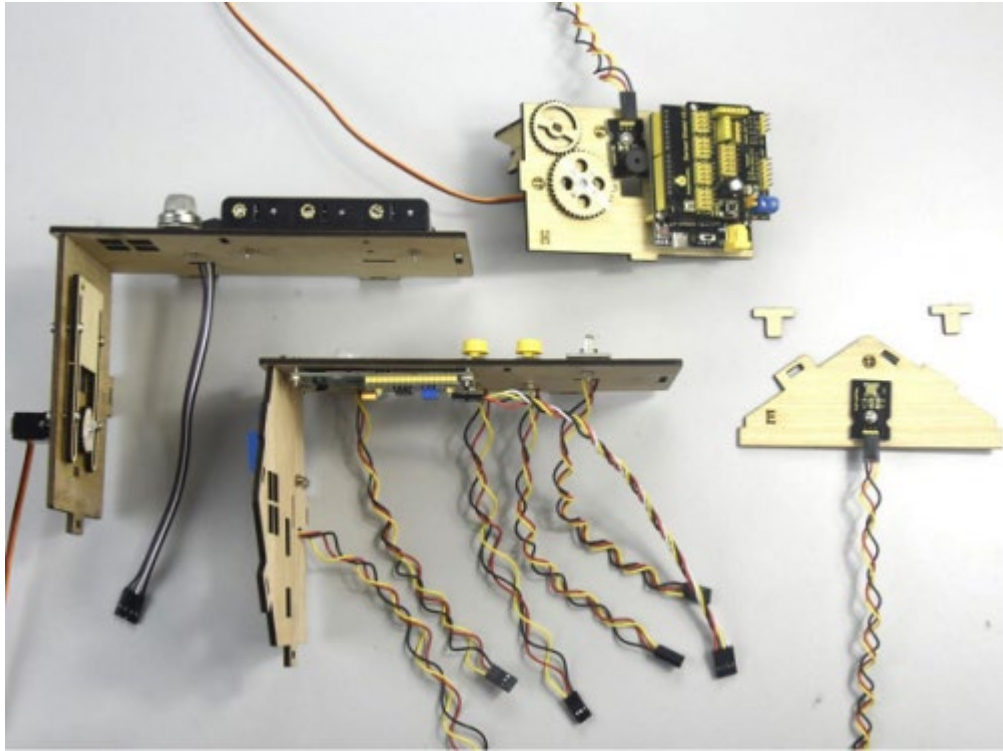
Fixe a engrenagem com montagem em cruz branca no servo preto com 1 parafuso M2*5MM (incluído no servo), depois instale a combinação da placa 2 de 2pcs e outro servo na área correspondente da placa H, finalmente empilhe a

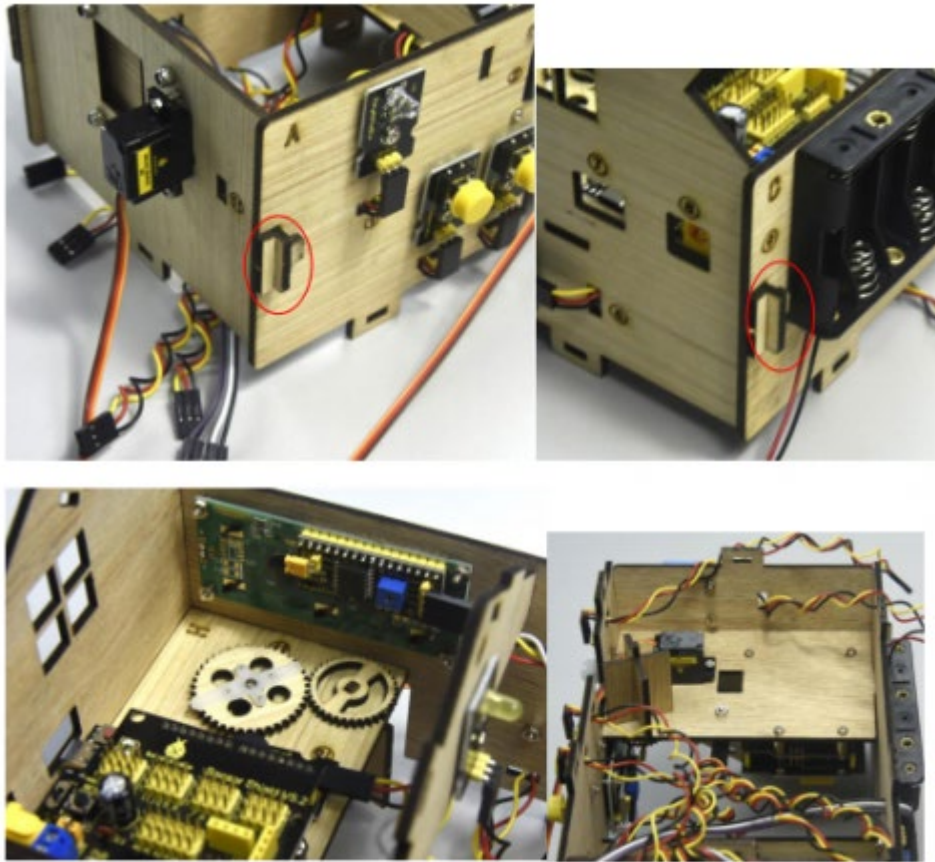
proteção do sensor no keystudio PLUS placa de controlo.



Etapa 9: Montar as placas A, B, C, D, E e H e, em seguida, fixá-las com 2 parafusos tipo "T".

(Nota: a interface de alimentação da placa de controlo PLUS está alinhada com o orifício ⑧ na placa B, e a interface do cabo USB está alinhada com o orifício ⑦ na placa B)

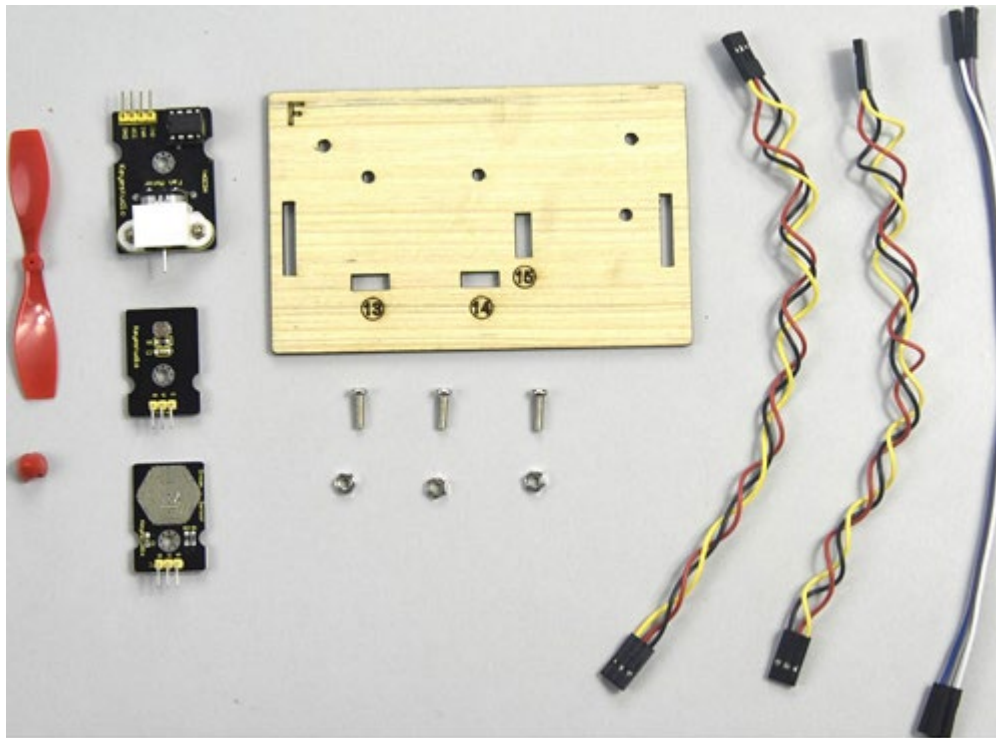




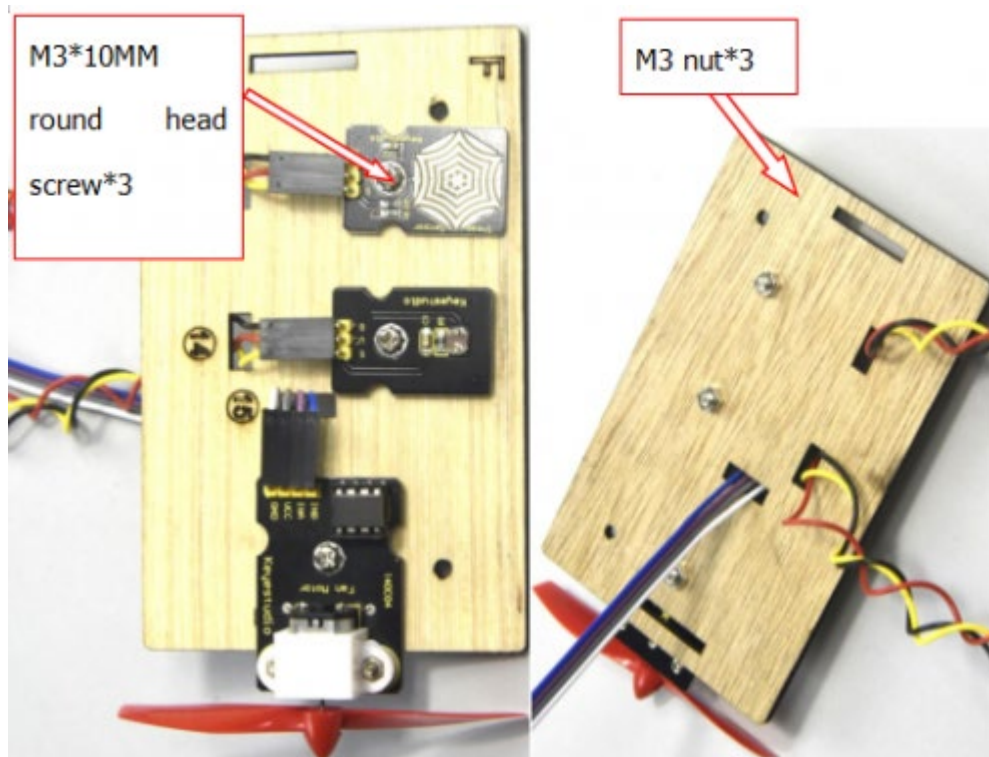
Passo 10: Instalar o sensor da placa F

Preparar um sensor de vapor, um sensor de fotocélula, um módulo de ventilador (com ventilador), uma placa F, 2pcs 3pin F-F Dupont line, 4pcs F-F dupont lines, 3pcs M3*10MM parafusos de cabeça redonda e 3pcs M3 porcas níqueladas.

F board*1	Steam sensor*1	Photocell sensor*1	Fan module*1	M3*10MM Round head screw*3	M3 Nickel plated nut*3	F-F Dupont line*4	3pin F-F Dupont line*2



Fixar separadamente o sensor de vapor, o sensor de fotocélula e o módulo do ventilador na placa F com 3 parafusos de cabeça redonda M3*10MM e 3 porcas M3 e, em seguida, ligá-los com linhas dupont de 3 e 4 pinos.



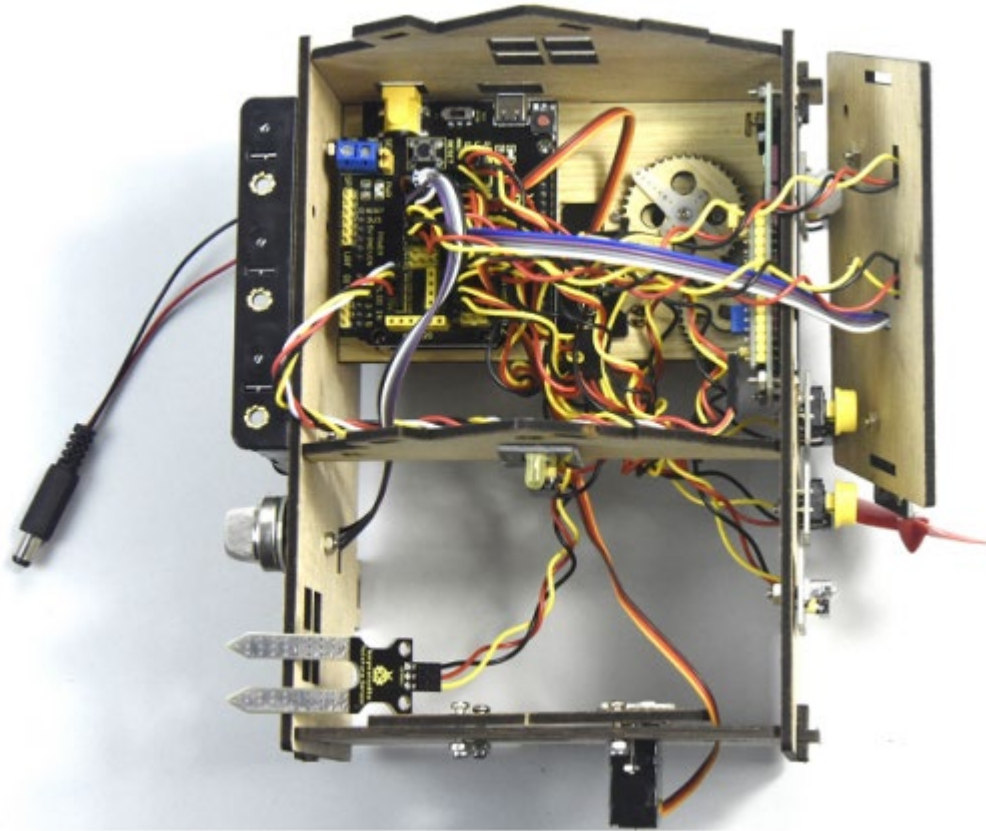
Etapa 11: Ligar o sensor/módulo

Ligar uma extremidade da linha dupont de 3 pinos ao pino do sensor de humidade do solo e, em seguida, ligar todos os sensores à blindagem do sensor.

(fixar 2 servos e fazer passar o fio dupont pelos orifícios da placa)

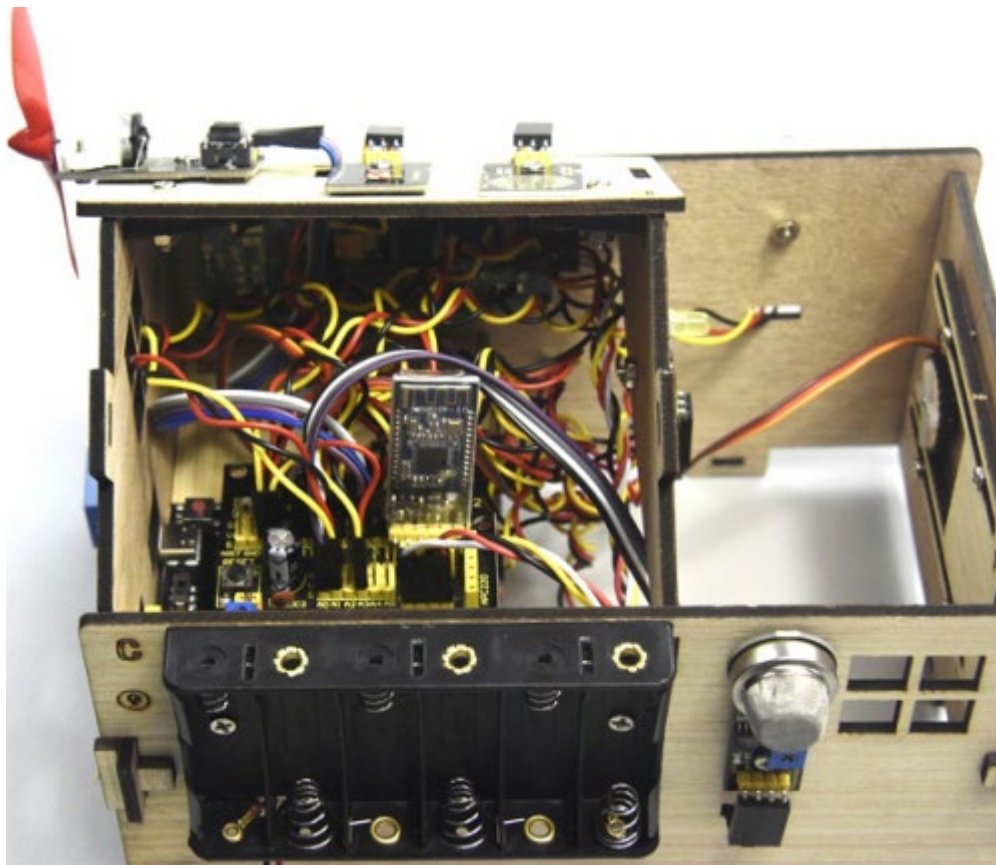


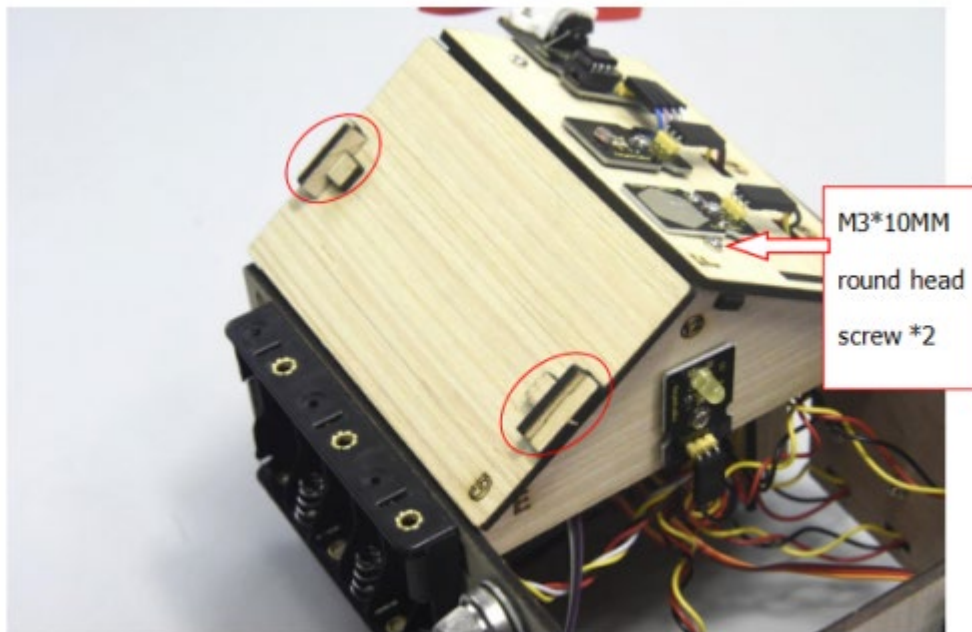
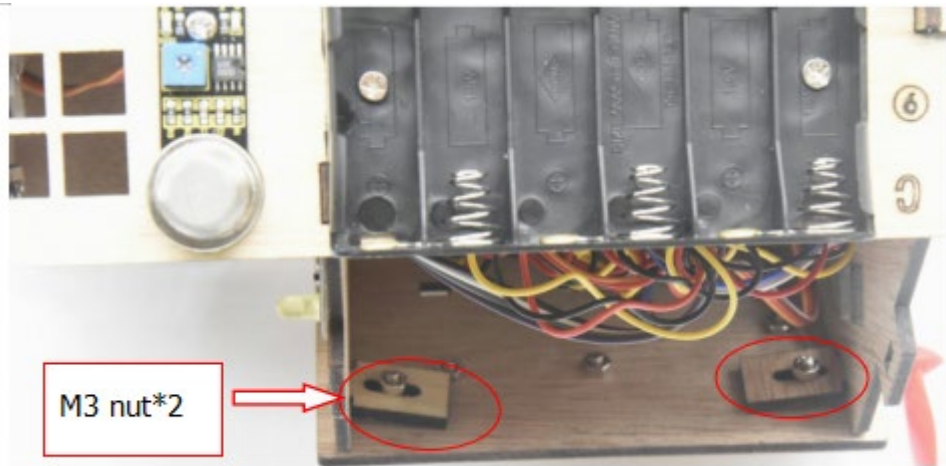
Name	The corresponding interfaces of sensors and sensor shield		The corresponding installed area on the board
PIR Motion Sensor	G/V/S	G/V/2	⑤
Passive buzzer	G/V/S	G/V/3	⑬
Button module 1	G/V/S	G/V/4	③
Yellow LED	G/V/S	G/V/5	⑫
Fan module	GND/VCC/INA/INB	G/V/7/6	⑤
Button module 2	G/V/S	G/V/8	④
Servo 1 controlling the door	Brown/Red/Orange wire	G/V/9	⑰
Servo 2 controlling the windows	Brown/Red/Orange wire	G/V/10	①
MQ-2 Gas Sensor	GND/VCC/D0/A0	G/V/11/A0	⑩
Relay Module	G/V/S	G/V/12	⑥
White LED	G/V/S	G/V/13	①
LCD1602 Display	GND/VCC/SDA/SCL	GND/5V/SDA/SCL	②
Photocell Sensor	G/V/S	G/V/A1	⑭
Soil humidity sensor	G/V/S	G/V/A2	
Steam sensor	G/V/S	G/V/A3	⑮



Inserir o módulo Bluetooth na blindagem do sensor, depois fixar a placa F com 2 parafusos de cabeça redonda M3*10MM, 2 porcas M3 e 2 peças com orifícios no meio, montar bem a placa G com 2 parafusos tipo "T".

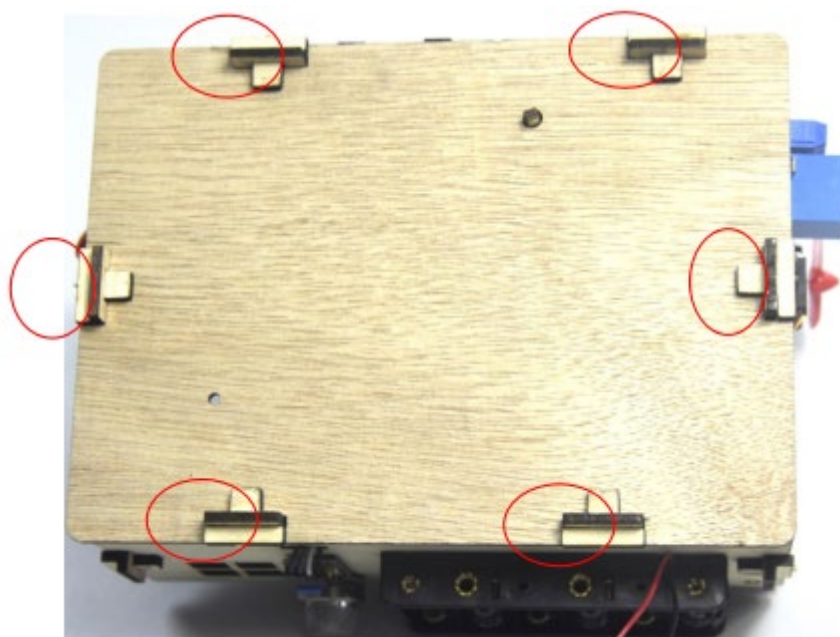
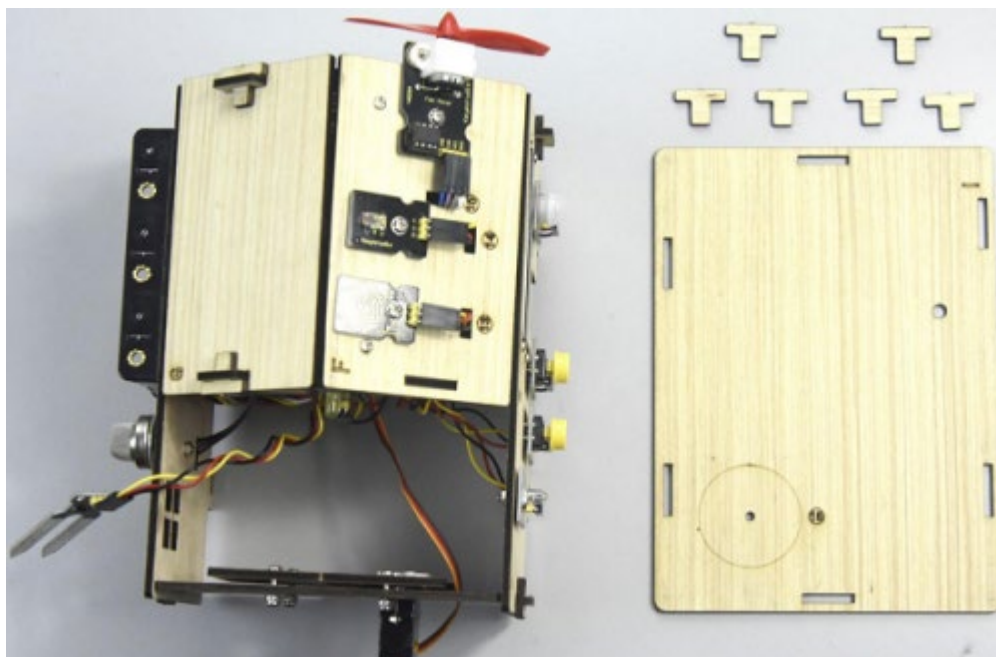
Bluetooth Module	Sensor Expansion Board
VCC	5V
GND	GND
TXD	RXD
RXD	TXD

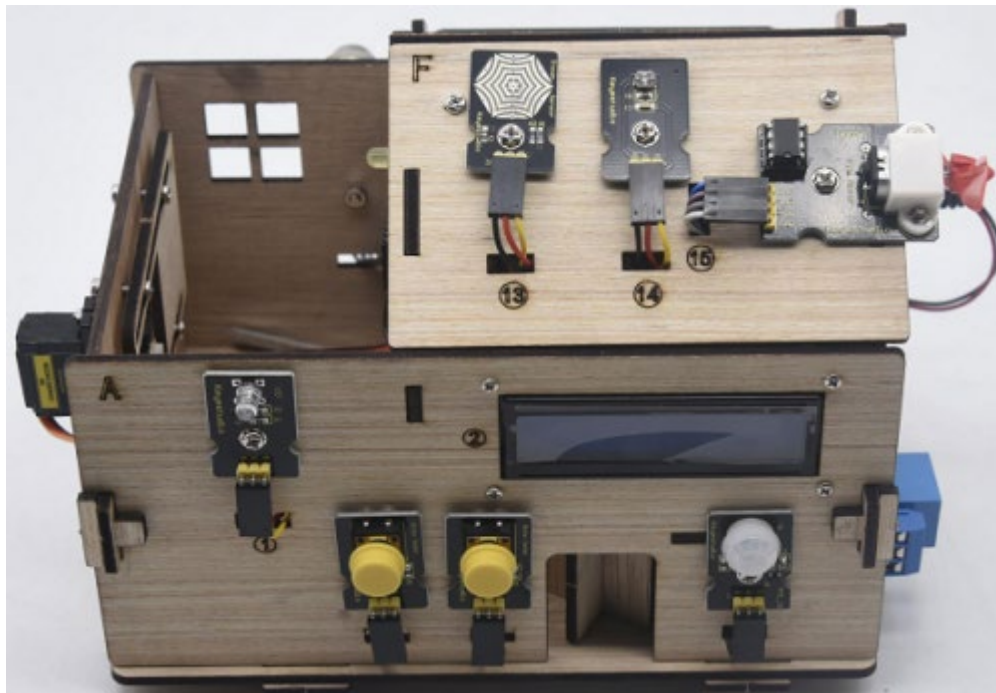




Passo 12: Montar o kit

Fixar o quadro I com 6 parafusos "T"





Projeto 15: Kit multiusos para casa inteligente

Descrição

Nos projectos anteriores, apresentámos a forma de utilizar sensores, módulos e o módulo Bluetooth HM-10. Nesta lição, vamos executar todas as funções e obter o efeito que se segue:

- -Sensor de fotocélula, sensor de movimento PIR e LED. Quando, à noite, alguém passa, o LED acende-se; se não estiver ninguém por

perto, o LED apaga-se.



- A placa contém um ecrã 1602LCD, 2 botões e 1 servo. Prima o botão 1 para introduzir a palavra-passe (pode definir a palavra-passe no Código de teste), o 1602LCD apresentará "*" e, em seguida, prima o botão 2 para "garantir". Se a palavra-passe estiver correcta, o 1602LCD mostrará "open" (aberto) e a porta será aberta. No entanto, se a palavra-passe estiver incorrecta, aparece a mensagem "erro", após 2s, "erro" transforma-se em "novamente", pode introduzir a palavra-passe de novo.

A porta será fechada quando o sensor de movimento PIR não detetar pessoas ao redor. Além do mais, pressione e segure o botão2, a campainha soará, o LCD exhibe "wait". Se a senha estiver correta, o servo girará para 180 °, caso contrário, o servo não girará)

Nota: A palavra-passe correcta é ". - - - . - .", o que significa premir brevemente o botão1, premir longamente o botão1, premir longamente o botão1, premir brevemente o botão1, premir longamente o botão1, premir brevemente o botão1.

" - " meios **long press button1**, " . " meios **short press button1**

- Insira a humidade do solo no vaso de plantas, quando o solo estiver demasiado seco, a campainha irá alarmar e receberá a notificação na



aplicação.

- Quando o sensor de gás detecta o gás com uma concentração elevada,



a campainha emite um som de alarme "tick,tick".

- - Quando o sensor de vapor detecta chuva, o servo 2 é ativado e a janela fecha-se automaticamente, caso contrário, a janela abre-

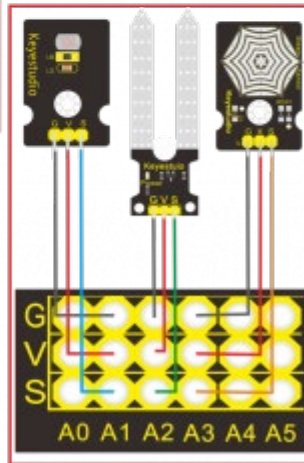
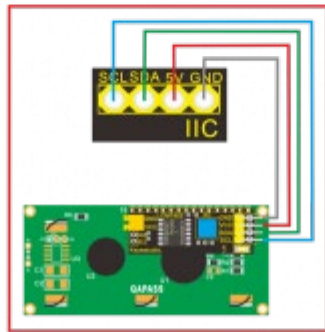
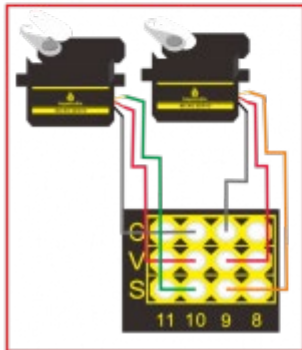
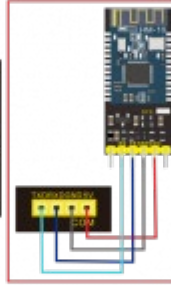
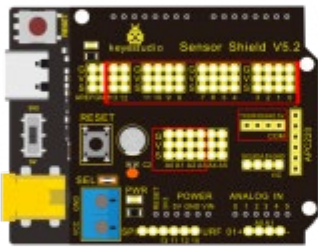
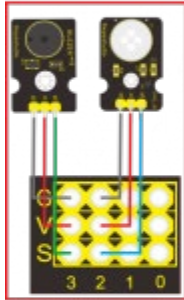
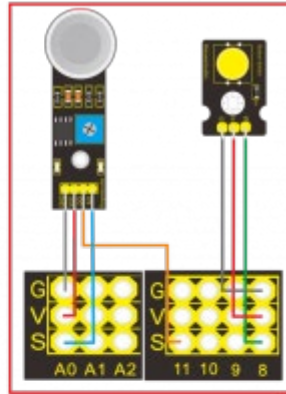
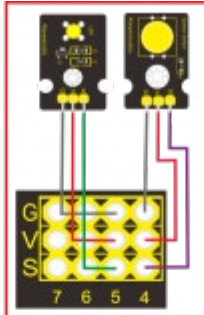
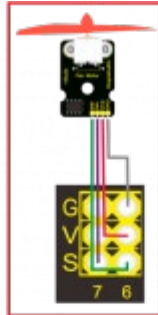
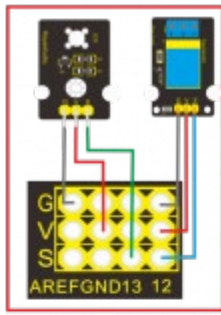


se.

Equipamento:

Keyestudio PLUS control board	Sensor shield	Fan module*1	Servo motor*2	LCD1602 display*1	Button sensor*2	White LED*1
Relay module*1	Passive buzzer*1	PIR motion sensor*1	Steam sensor*1	photocell sensor*1	Bluetooth module*1	Yellow LED*1
Soil humidity sensor*1	MQ-2 Gas sensor*1	4pin F-F Dupont line*1	F-F Dupont lines	USB cable*1	3pin F-F Dupont line*10	

Diagrama de ligação:



Name	The corresponding interfaces of sensors and sensor shield		The corresponding installed area on the board
PIR Motion Sensor	G/V/S	G/V/2	⑤
Passive buzzer	G/V/S	G/V/3	⑬
Button module 1	G/V/S	G/V/4	③
Yellow LED	G/V/S	G/V/5	⑫
Fan module	GND/VCC/INA/INB	G/V/7/6	⑤
Button module 2	G/V/S	G/V/8	④
Servo 1 controlling the door	Brown/Red/Orange wire	G/V/9	⑰
Servo 2 controlling the windows	Brown/Red/Orange wire	G/V/10	①
MQ-2 Gas Sensor	GND/VCC/D0/A0	G/V/11/A0	⑩
Relay Module	G/V/S	G/V/12	⑥
White LED	G/V/S	G/V/13	①
LCD1602 Display	GND/VCC/SDA/SCL	GND/5V/SDA/SCL	②
Photocell Sensor	G/V/S	G/V/A1	⑭
Soil humidity sensor	G/V/S	G/V/A2	
Steam sensor	G/V/S	G/V/A3	⑮

Código de teste:

```
//call the relevant library file

#include <Servo.h>

#include <Wire.h>

#include <LiquidCrystal_I2C.h>

//Set the communication address of I2C to 0x27, display 16 characters every line,
two lines in total

LiquidCrystal_I2C mylcd(0x27, 16, 2);
```

```
//set ports of two servos to digital 9 and 10

Servo servo_10;

Servo servo_9;

volatile int btn1_num;//set variable btn1_num

volatile int btn2_num;//set variable btn2_num

volatile int button1;//set variable button1

volatile int button2;//set variable button2

String fans_char;//string type variable fans_char

volatile int fans_val;//set variable fans_char

volatile int flag;//set variable flag

volatile int flag2;//set variable flag2

volatile int flag3;//set variable flag3

volatile int gas;//set variable gas

volatile int infrar;//set variable infrar

String led2;//string type variable led2

volatile int light;//set variable light

String pass;//string type variable pass

String passwd;//string type variable passwd

String servo1;//string type variable servo1

volatile int servo1_angle;//set variable light
```

```
String servo2;//string type variable servo2

volatile int servo2_angle;//set variable servo2_angle

volatile int soil;//set variable soil

volatile int val;//set variable val

volatile int value_led2;//set variable value_led2

volatile int water;//set variable water

int length;

int tonepin = 3; //set the signal end of passive buzzer to digital 3

//define name of every sound frequency

#define D0 -1

#define D1 262

#define D2 293

#define D3 329

#define D4 349

#define D5 392

#define D6 440

#define D7 494

#define M1 523

#define M2 586

#define M3 658
```

```
#define M4 697

#define M5 783

#define M6 879

#define M7 987

#define H1 1045

#define H2 1171

#define H3 1316

#define H4 1393

#define H5 1563

#define H6 1755

#define H7 1971

#define WHOLE 1

#define HALF 0.5

#define QUARTER 0.25

#define EIGHTH 0.25

#define SIXTEENTH 0.625

//set sound play frequency

int tune[] =

{

    M3, M3, M4, M5,
```

```
M5, M4, M3, M2,  
  
M1, M1, M2, M3,  
  
M3, M2, M2,  
  
M3, M3, M4, M5,  
  
M5, M4, M3, M2,  
  
M1, M1, M2, M3,  
  
M2, M1, M1,  
  
M2, M2, M3, M1,  
  
M2, M3, M4, M3, M1,  
  
M2, M3, M4, M3, M2,  
  
M1, M2, D5, D0,  
  
M3, M3, M4, M5,  
  
M5, M4, M3, M4, M2,  
  
M1, M1, M2, M3,  
  
M2, M1, M1  
  
};
```

```
//set music beat
```

```
float durt[] =
```

```
{
```

```
1, 1, 1, 1,
```

```
1, 1, 1, 1,
```



```
1, 1, 1, 1,  
1 + 0.5, 0.5, 1 + 1,  
1, 1, 1, 1,  
1, 1, 1, 1,  
1, 1, 1, 1,  
1 + 0.5, 0.5, 1 + 1,  
1, 1, 1, 1,  
1, 0.5, 0.5, 1, 1,  
1, 0.5, 0.5, 1, 1,  
1, 1, 1, 1,  
1, 1, 1, 1,  
1, 1, 1, 0.5, 0.5,  
1, 1, 1, 1,  
1 + 0.5, 0.5, 1 + 1,  
};
```

```
void setup() {  
    Serial.begin(9600); //set baud rate to 9600  
  
    mylcd.init();  
  
    mylcd.backlight(); //initialize LCD
```

```
//LCD shows "passcord:" at first row and column

mylcd.setCursor(1 - 1, 1 - 1);

mylcd.print("passcord:");

servo_9.attach(9);//make servo connect to digital 9

servo_10.attach(10);//make servo connect to digital 10

servo_9.write(0);//set servo connected digital 9 to 0°

servo_10.write(0);//set servo connected digital 10 to 0°

delay(300);

pinMode(7, OUTPUT);//set digital 7 to output

pinMode(6, OUTPUT);//set digital 6 to output

digitalWrite(7, HIGH); //set digital 7 to high level

digitalWrite(6, HIGH); //set digital 6 to high level

pinMode(4, INPUT);//set digital 4 to input

pinMode(8, INPUT);//set digital 8 to input

pinMode(2, INPUT);//set digital 2 to input

pinMode(3, OUTPUT);//set digital 3 to output

pinMode(A0, INPUT);//set A0 to input

pinMode(A1, INPUT);//set A1 to input

pinMode(13, OUTPUT);//set digital 13 to input
```

```
pinMode(A3, INPUT); //set A3 to input

pinMode(A2, INPUT); //set A2 to input

pinMode(12, OUTPUT); //set digital 12 to output

pinMode(5, OUTPUT); //set digital 5 to output

pinMode(3, OUTPUT); //set digital 3 to output

length = sizeof(tune) / sizeof(tune[0]); //set the value of length
}

void loop() {

  auto_sensor();

  if (Serial.available() > 0) //serial reads the characters

  {

    val = Serial.read(); //set val to character read by serial

    Serial.println(val); //output val character in new lines

    pwm_control();

  }

  switch (val) {

    case 'a': //if val is character 'a', program will circulate

      digitalWrite(13, HIGH); //set digital 13 to high level, LED lights up

      break; //exit loop

    case 'b': //if val is character 'b', program will circulate
```

```
digitalWrite(13, LOW); //Set digital 13 to low level, LED is off

break;//exit loop

case 'c'://if val is character 'c', program will circulate

    digitalWrite(12, HIGH); //set digital 12 to high level, NO of relay is
connected to COM

    break;//exit loop

case 'd'://if val is character 'd', program will circulate

    digitalWrite(12, LOW); //set digital 12 to low level, NO of relay is
disconnected to COM

    break;//exit loop

case 'e'://if val is character 'e', program will circulate

    music1();//play birthday song

    break;//exit loop

case 'f'://if val is character 'f', program will circulate

    music2();//play ode to joy song

    break;//exit loop

case 'g'://if val is character 'g', program will circulate

    noTone(3);//set digital 3 to stop playing music

    break;//exit loop

case 'h'://if val is character 'h', program will circulate

    Serial.println(light);//output the value of variable light in new lines
```

```
    delay(100);

    break;//exit loop

case 'i'://if val is character 'i', program will circulate

    Serial.println(gas);//output the value of variable gas in new lines

    delay(100);

    break;//exit loop

case 'j'://if val is character 'j', program will circulate

    Serial.println(soil);//output the value of variable soil in new lines

    delay(100);

    break;//exit loop

case 'k'://if val is character 'k', program will circulate

    Serial.println(water);//output the value of variable water in new lines

    delay(100);

    break;//exit loop

case 'l'://if val is character 'l', program will circulate

    servo_9.write(180);//set servo connected to digital 9 to 180°

    delay(500);

    break;//exit loop

case 'm'://if val is character 'm', program will circulate

    servo_9.write(0);//set servo connected to digital 9 to 0°

    delay(500);

    break;//exit loop
```

```
case 'n'://if val is character 'n', program will circulate

    servo_10.write(180);//set servo connected to digital 10 to 180°

    delay(500);

    break;//exit loop

case 'o'://if val is character 'o', program will circulate

    servo_10.write(0);//set servo connected to digital 10 to 0°

    delay(500);

    break;//exit loop

case 'p'://if val is character 'p', program will circulate

    digitalWrite(5, HIGH); //set digital 5 to high level, LED is on

    break;//exit loop

case 'q'://if val is character 'q', program will circulate

    digitalWrite(5, LOW); // set digital 5 to low level, LED is off

    break;//exit loop

case 'r'://if val is character 'r', program will circulate

    digitalWrite(7, LOW);

    digitalWrite(6, HIGH); //fan rotates anticlockwise at the fastest speed

    break;//exit loop

case 's'://if val is character 's', program will circulate

    digitalWrite(7, LOW);

    digitalWrite(6, LOW); //fan stops rotating

    break;//exit loop
```

```
    }  
}  
  
////////////////////////////////////set birthday song////////////////////////////////////  
void birthday()  
{  
    tone(3, 294); //digital 3 outputs 294HZ sound  
  
    delay(250); //delay in 250ms  
  
    tone(3, 440);  
  
    delay(250);  
  
    tone(3, 392);  
  
    delay(250);  
  
    tone(3, 532);  
  
    delay(250);  
  
    tone(3, 494);  
  
    delay(500);  
  
    tone(3, 392);  
  
    delay(250);  
  
    tone(3, 440);  
  
    delay(250);  
  
    tone(3, 392);  
  
    delay(250);
```

```
tone(3, 587);
```

```
delay(250);
```

```
tone(3, 532);
```

```
delay(500);
```

```
tone(3, 392);
```

```
delay(250);
```

```
tone(3, 784);
```

```
delay(250);
```

```
tone(3, 659);
```

```
delay(250);
```

```
tone(3, 532);
```

```
delay(250);
```

```
tone(3, 494);
```

```
delay(250);
```

```
tone(3, 440);
```

```
delay(250);
```

```
tone(3, 698);
```

```
delay(375);
```

```
tone(3, 659);
```

```
delay(250);
```

```
tone(3, 532);
```

```
delay(250);
```



```
tone(3, 587);

delay(250);

tone(3, 532);

delay(500);

}

//detect gas

void auto_sensor() {

    gas = analogRead(A0);//assign the analog value of A0 to gas

    if (gas > 700) {

//if variable gas>700

        flag = 1;//set variable flag to 1

        while (flag == 1)

            //if flag is 1, program will circulate

            {

                Serial.println("danger");//output "danger" in new lines

                tone(3, 440);

                delay(125);

                delay(100);

                noTone(3);
```

```
delay(100);

tone(3, 440);

delay(125);

delay(100);

noTone(3);

delay(300);

gas = analogRead(A0); //gas analog the value of A0 to gas

if (gas < 100) //if variable gas is less than 100

{

    flag = 0; //set variable flag to 0

    break; //exit loop exist to loop

}

}

} else

    //otherwise

{

    noTone(3); // digital 3 stops playing music

}

light = analogRead(A1); //Assign the analog value of A1 to light

if (light < 300) //if variable light is less than 300

{
```

```
infrar = digitalRead(2);//assign the value of digital 2 to infrar

Serial.println(infrar);//output the value of variable infrar in new lines

if (infrar == 1)

    // if variable infra is 1

    {

        digitalWrite(13, HIGH); //set digital 13 to high level, LED is on

    } else//Otherwise

    {

        digitalWrite(13, LOW); //set digital 13 to low level, LED is off

    }

}

water = analogRead(A3);//assign the analog value of A3 to variable water

if (water > 800)

    // if variable water is larger than 800

    {

        flag2 = 1;//if variable flag 2 to 1

        while (flag2 == 1)

            // if flag2 is 1, program will circulate

            {

                Serial.println("rain");//output "rain" in new lines

                servo_10.write(180);// set the servo connected to digital 10 to 180°
```

```
delay(300); //delay in 300ms

delay(100);

water = analogRead(A3); //assign the analog value of A3 to variable water

if (water < 30) // if variable water is less than 30
{
    flag2 = 0; // set flag2 to 0

    break; //exit loop
}

}

} else //Otherwise
{
    if (val != 'u' && val != 'n')
        //if val is not equivalent 'u' either 'n'
        {
            servo_10.write(0); //set servo connected to digital 10 to 0°

            delay(10);

        }

}

}

soil = analogRead(A2); //assign the analog value of A2 to variable soil
```

```
if (soil > 50)

    // if variable soil is greater than 50

{

    flag3 = 1;//set flag3 to 1

    while (flag3 == 1)

        //If set flag3 to 1, program will circulate

        {

            Serial.println("hydropenia ");//output "hydropenia " in new lines

            tone(3, 440);

            delay(125);

            delay(100);

            noTone(3);

            delay(100);

            tone(3, 440);

            delay(125);

            delay(100);

            noTone(3);//digital 3 stops playing sound

            delay(300);

            soil = analogRead(A2);//Assign the analog value of A2 to variable soil

            if (soil < 10)//If variable soil<10

            {

                flag3 = 0;//set flag3 to 0
```

```
        break;//exit loop
    }
}

} else//Otherwise
{
    noTone(3);//set digital 3 to stop playing music
}

door();//run subroutine
}

void door() {
    button1 = digitalRead(4);// assign the value of digital 4 to button1
    button2 = digitalRead(8);//assign the value of digital 8 to button2

    if (button1 == 0)//if variablebutton1 is 0
    {
        delay(10);//delay in 10ms

        while (button1 == 0) //if variablebutton1 is 0, program will circulate
        {
            button1 = digitalRead(4);// assign the value of digital 4 to button1
```

```

    btn1_num = btn1_num + 1;//variable btn1_num plus 1

    delay(100);// delay in 100ms

}

}

if (btn1_num >= 1 && btn1_num < 5) //1≤if variablebtn1_num<5
{

    Serial.print(".");

    Serial.print("");

    passwd = String(passwd) + String(".");//set passwd
pass = String(pass) + String(".");//set pass

    //LCD shows pass at the first row and column

    mylcd.setCursor(1 - 1, 2 - 1);

    mylcd.print(pass);

}

if (btn1_num >= 5)

    //if variablebtn1_num ≥5

{

    Serial.print("-");

    passwd = String(passwd) + String("-");//Set passwd

    pass = String(pass) + String("-");//set pass

    //LCD shows pass at the first row and column

```

```
mylcd.setCursor(1 - 1, 2 - 1);

mylcd.print(pass);

}

if (button2 == 0) //if variablebutton2 is 0

{

    delay(10);

    if (button2 == 0)//if variablebutton2 is 0

    {

        if (passwd == ".-.-.")//if passwd is ".-.-."

        {

            mylcd.clear();//clear LCD screen

            //LCD shows "open!" at first character on second row

            mylcd.setCursor(1 - 1, 2 - 1);

            mylcd.print("open!");

            servo_9.write(100);//set servo connected to digital 9 to 100°

            delay(300);

            delay(5000);

            passwd = "";

            pass = "";

            mylcd.clear();//clear LCD screen

            //LCD shows "password:" at first character on first row
```



```
    mylcd.setCursor(1 - 1, 1 - 1);

    mylcd.print("password:");

} else //Otherwise

{

    mylcd.clear();//clear LCD screen

    //LCD shows "error!"at first character on first row

    mylcd.setCursor(1 - 1, 1 - 1);

    mylcd.print("error!");

    passwd = "";

    pass = "";

    delay(2000);

    //LCD shows "again" at first character on first row

    mylcd.setCursor(1 - 1, 1 - 1);

    mylcd.print("again");

}

}

}

infrar = digitalRead(2);//assign the value of digital 2 to infrar

if (infrar == 0 && (val != 'l' && val != 't'))

    //if variable infrar is 0 and val is not 'l' either 't'

{
```

```
servo_9.write(0);//set servo connected to digital 9 to 0°

delay(50);

}

if (button2 == 0)//if variablebutton2 is 0

{

    delay(10);

    while (button2 == 0) //if variablebutton2 is 0, program will circulate

    {

        button2 = digitalRead(8);//assign the value of digital 8 to button2

        btn2_num = btn2_num + 1;//variable btn2_num plus 1

        delay(100);

        if (btn2_num >= 15)//if variablebtn2_num ≥15

        {

            tone(3, 532);

            delay(125);

            mylcd.clear();//clear LCD screen

            //LCD shows "password:" at the first character on first row

            mylcd.setCursor(1 - 1, 1 - 1);

            mylcd.print("password:");

            //LCD shows "wait" at the first character on first row

            mylcd.setCursor(1 - 1, 1 - 1);

            mylcd.print("wait");
```

```
    } else//Otherwise

    {

        noTone(3);//digital 3 stops playing music

    }

}

}

btn1_num = 0;//set btn1_num to 0

btn2_num = 0;//set btn2_num to 0

}

// Birthday song

void music1() {

    birthday();

}

//Ode to joy

void music2() {

    Ode_to_Joy();

}

void Ode_to_Joy()//play Ode to joy song

{

    for (int x = 0; x < length; x++)
```

```

{
    tone(tonepin, tune[x]);

    delay(300 * durt[x]);

}

}

//PWM control

void pwm_control() {

    switch (val)

    {

        case 't'://if val is 't', program will circulate

            servo1 = Serial.readStringUntil('#');

            servo1_angle = String(servo1).toInt();

            servo_9.write(servo1_angle);//set the angle of servo connected to digital 9

to servo1_angle

            delay(300);

            break;//exit loop

        case 'u'://if val is 'u', program will circulate

            servo2 = Serial.readStringUntil('#');

            servo2_angle = String(servo2).toInt();

            servo_10.write(servo2_angle);//set the angle of servo connected to digital 10

to servo2_angle

```

```
    delay(300);

    break;//exit loop

case 'v'://if val is 'v', program will circulate

    led2 = Serial.readStringUntil('#');

    value_led2 = String(led2).toInt();

    analogWrite(5, value_led2); //PWM value of digital 5 is value_led2

    break;//exit loop

case 'w'://if val is 'w', program will circulate

    fans_char = Serial.readStringUntil('#');

    fans_val = String(fans_char).toInt();

    digitalWrite(7, LOW);

    analogWrite(6, fans_val); //set PWM value of digital 6 to fans_val, the larger

the value, the faster the fan

    break;//exit loop

}

}
```

Carregue o código e veja o resultado!

Nota: Retire o módulo Bluetooth quando estiver a carregar o Código de teste. Caso contrário, o programa não será carregado. Ligue o Bluetooth e o módulo Bluetooth para emparelhar depois de carregar o Código de teste.

CE Declaration of Conformity

Company: Shenzhen Scope Corporation, Ltd.

Address: 12/13 Floors, C2 Building, I Park, No. 1001, College Road, Nanshan, Shenzhen, Guangdong, China

Product Name: Folding Soft Box Light Kit

Product Model: FX60

Directives and Standard applicable:

LVD, 2014/35/EU

EN60598-1 :2015+A1 :2018

EN60598-2-9:1989+A1:1994

RoHS, 2011/65/EU

IEC 62321-4:2013+A1:2017

IEC 62321-5:2013

IEC 62321-6:2015

IEC 62321-7-1:2015

IEC 62321-7-2:2017

IEC 62321-8:2017

Signature: _____



Date: _____

2023.8.11

Internal Reference: BKC21102WC