

Prática Arduino
Roteiro de Aula Prática – Medindo Temperatura e Humidade

Nome: _____ Matric.: _____

O sensor DH11

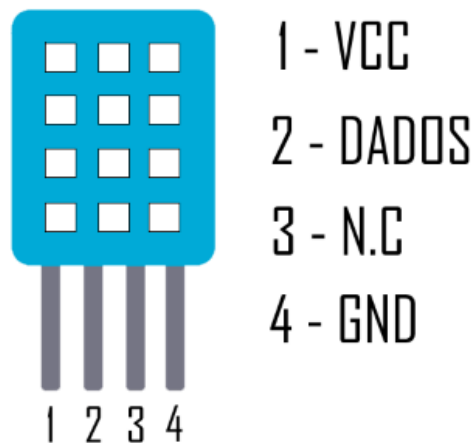
Abaixo, uma imagem do sensor que usaremos nessa aula. Este é o DH11



Este sensor é capaz de medir a umidade do ar com uma precisão de $\pm 5\%$ UR, em uma faixa de 20% a 90%UR. Já com relação a temperatura, ele é capaz de medir uma faixa entre 0 a 50°C, com uma precisão de $\pm 2^\circ\text{C}$. O tempo de resposta é de até 5s.

Pinos

Os pinos usados para a comunicação e alimentação do sensor estão descritos na imagem abaixo:

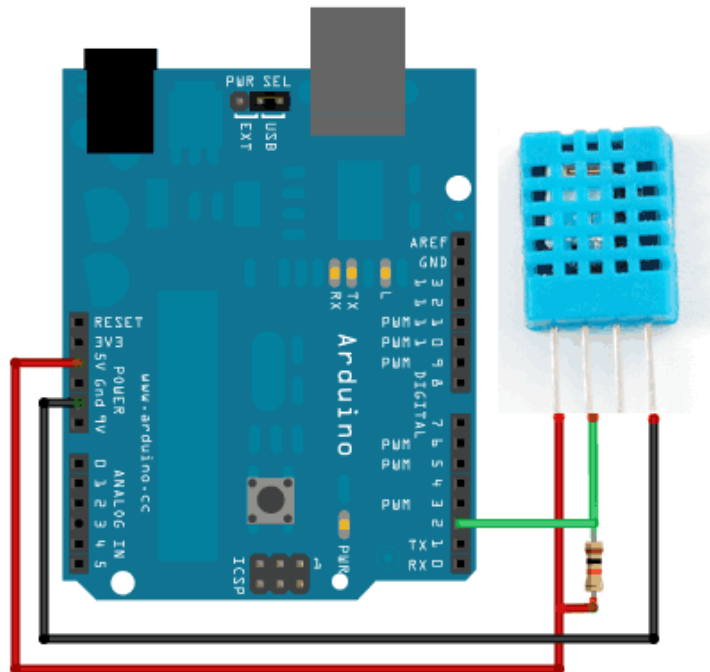


Relembrando a aula anterior, temos:

- **VCC** – Pino de alimentação. Para este sensor, a alimentação pode variar de 3 a 5 VDC.
- **Dados** – Este pino é responsável pelo envio de dados do sensor para a placa Arduino.
- **N.C.** – Este pino não será usado.
- **GND** – Aterramento.

Circuito

Para esta aula, vamos utilizar a seguinte ligação entre o sensor e a placa Arduino:



Assim temos:

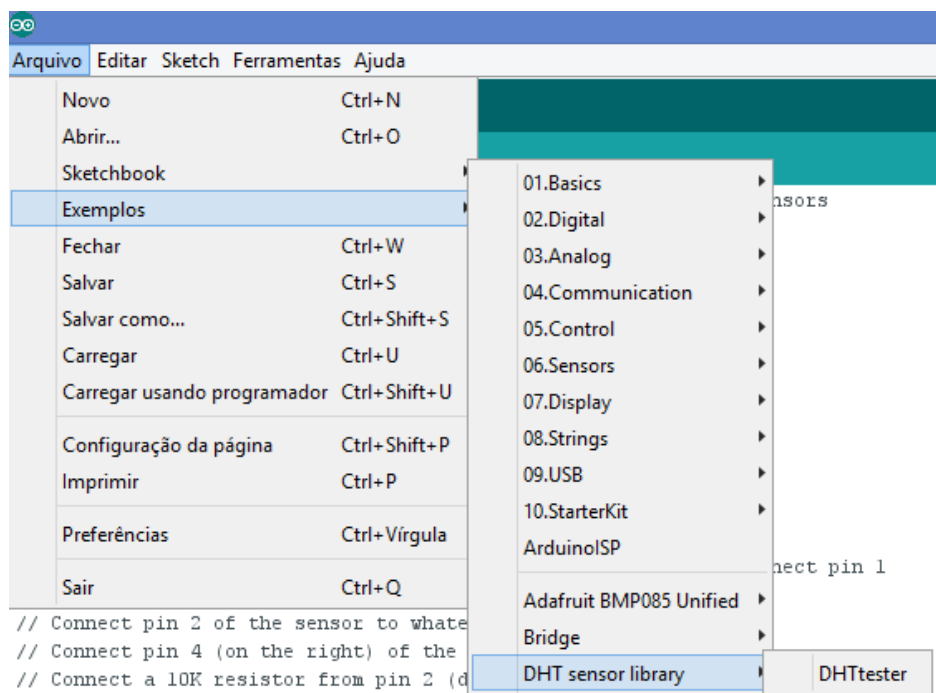
- **Fio Vermelho:** Pino 1 do DH11 (VCC) e o pino de alimentação de 5V da placa.
- **Fio Verde:** Pino 2 do DH11 (Dados) e o pino 2 da entrada digital da placa.
- **Fio Preto:** Pino 4 do DH11 (GND) e o pino GND da placa.

Entre o pino VCC e o pino de dados, adicione um resistor de 10 KΩ.

ATENÇÃO: NÃO FAÇA NENHUMA LIGAÇÃO OU ALTERAÇÃO DO CIRCUITO SEM A SUPERVISÃO DO PROFESSOR.

Um exemplo:

Depois de instalar a biblioteca que usaremos para ler os dados do sensor (certifique-se com o professor de que essa biblioteca está instalada) vamos carregar um exemplo de uso do sensor. No menu Arquivo, vá em Exemplos > DHT sensor library > dht tester, como na figura abaixo:



O seguinte exemplo deverá ser carregado:

```
#include "DHT.h"
#define DHTPIN 2      // what pin we're connected to

// #define DHTTYPE DHT11    // DHT 11
#define DHTTYPE DHT22    // DHT 22  (AM2302)
// #define DHTTYPE DHT21    // DHT 21  (AM2301)

// Initialize DHT sensor for normal 16mhz Arduino
DHT dht(DHTPIN, DHTTYPE);

void setup() {
  Serial.begin(9600);
  Serial.println("DHTxx test!");

  dht.begin();
}

void loop() {
  // Wait a few seconds between measurements.
  delay(2000);



  // Reading temperature or humidity takes about 250 milliseconds!
  // Sensor readings may also be up to 2 seconds 'old' (its a very slow sensor)
  float h = dht.readHumidity();
  // Read temperature as Celsius
  float t = dht.readTemperature();
  // Read temperature as Fahrenheit
  float f = dht.readTemperature(true);

  // Check if any reads failed and exit early (to try again).
  if (isnan(h) || isnan(t) || isnan(f)) {
    Serial.println("Failed to read from DHT sensor!");
    return;
  }
  // Compute heat index
  // Must send in temp in Fahrenheit!
  float hi = dht.computeHeatIndex(f, h);

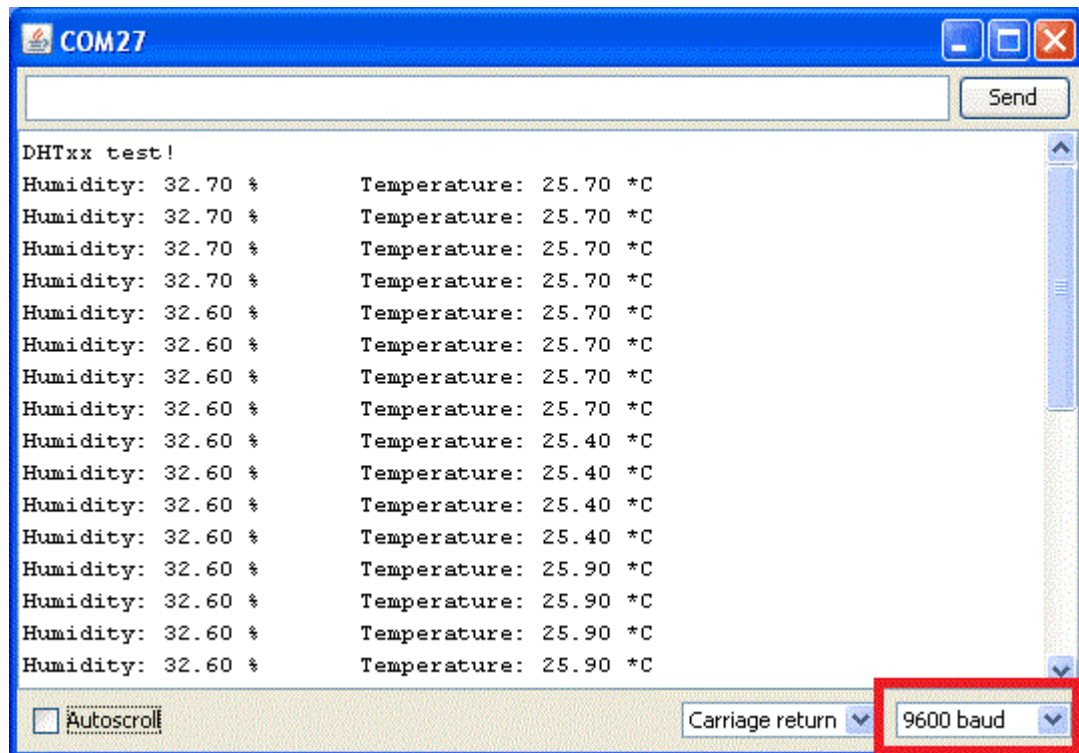
  Serial.print("Humidity: ");
  Serial.print(h);
  Serial.print(" %\t");
  Serial.print("Temperature: ");
  Serial.print(t);
  Serial.print(" *C ");
  Serial.print(f);
  Serial.print(" *F\t");
  Serial.print("Heat index: ");
  Serial.print(hi);
  Serial.println(" *F");
}
```

Procure pela linha `#define DHTTYPE DHT22` e adicione um comentário a ela, isto é, inclua `//` em sua frente. Procure pela linha `// #define DHTTYPE DHT11` e remova o comentário dela, isto é, remova `//` em sua frente. O resultado final deve ser parecido com este:

```
#define DHTTYPE DHT11    // DHT 11
// #define DHTTYPE DHT22    // DHT 22  (AM2302)
```

Clique no botão  para verificar se não existem erros no código. Em seguida, clique em  para mandar o programa para a placa Arduino.

Caso tenha sucesso, abra o Monitor serial, pelo menu Ferramentas > Monitor Serial.



No canto inferior direito da tela (marcado com o quadrado), selecione 9600 baud. Feche e abra novamente o Monitor Serial. Se tudo ocorrer corretamente, um resultado similar a esta imagem será obtido.

Você pode ter uma boa variação dos valores respirando bem perto do sensor, como se você estivesse em frente a uma janela e fazendo ela ficar embaçada.

Entendendo o código

As funções explicadas nas práticas anteriores não serão abordadas.

A diretiva DEFINE

A diretiva DEFINE é usada para definir constantes na linguagem de programação. Caso não lembre o que são constantes, revise a aula anterior.

Sintaxe:

#DEFINE *token* *VALOR*

Token – Essa palavra será usada para substituir *VALOR* no código.

VALOR – Este valor substituirá o *token* no momento da compilação do programa.

EXEMPLO:

No código exemplo, temos:

```
#define DHTPIN 2      // TOKEN = DHTPIN; VALUE = 2
#define DHTTYPE DHT11 // TOKEN = DHTTYPE; VALUE = DHT11
DHT dht(DHTPIN, DHTTYPE);
```

Após a compilação, este será o código mandado para a placa Arduino

```
#define DHTPIN 2      // TOKEN = DHTPIN; VALUE = 2
#define DHTTYPE DHT11 // TOKEN = DHTTYPE; VALUE = DHT11
DHT dht(2, DHT11);
```

DHT dht(*pino*, *sensormodel*)

Declara uma ‘variável’ chamada dht do tipo DHT que representa o sensor. Este tipo (DHT) está definido dentro da biblioteca. Para declarar essa ‘variável’, é necessário inicializa-la com 2 parâmetros:

- **Pino** – Número do pino da placa Arduino em que o pino de dados do sensor está conectado.
- **Sensormodel** – Modelo do sensor. Cada modelo é identificado por uma constante que está definida dentro do código da biblioteca. As seguintes constantes são definidas:
 - **DHT11** – modelo DH11 (modelo que estamos usando nesta prática)
 - **DHT22** – modelo DH22
 - **DHT21 ou AM2301** – Modelo DH21

Serial.begin(*velocidade*)

Abre a porta Serial e configura a taxa de dados em bits por segundo (baud) para a transmissão de dados serial.

Parâmetros

Velocidade – velocidade da taxa em bits por segundo.

Retorno: Nenhum

Serial.println(*valor*)

Imprime dados na porta Serial e quebra uma linha.

Parâmetros

valor – valor que será impresso.

Retorno: Tipo long. Número de bytes escritos. Porém, a leitura desse número é opcional.

dht.begin()

Configura o pino em que a variável que representa o sensor (no nosso caso, ela se chama dht) foi iniciado (na nossa montagem, foi o pino 2 representado pela constante DTHPIN) como modo INPUT e o nível do pino como HIGH. (Caso não lembre dessas constantes, revise a aula anterior).

Parâmetros: Nenhum

Retorno: Nenhum

dht.readHumidity()

Lê o valor de umidade coletado pelo sensor

Parâmetros - Nenhum

Retorno: Tipo float. O valor da humidade lido pelo sensor.

dht.readTemperature(*escala*)

Lê o valor de temperatura coletado pelo sensor

Parâmetros

Escala: **true** para Fahrenheit e **false** para Celsius.

Retorno: Tipo float. O valor da temperatura lido pelo sensor, de acordo com a escala escolhida

isnan(*valor*)

Verifica se o valor passado por parâmetro é um número ou não.

Parâmetros

Valor – valor que será analisado.

Retorno: **true** se NÃO for número e **false** se for número.

dht.computeHeatIndex(*temperatura*, *humidade*)

Calcula o Índice de calor.

Parâmetros

Temperatura – temperatura em Fahrenheit.

Humidade – porcentagem de humidade do ar.

Retorno: Tipo float. O valor do Índice de Calor.

Serial.print(valor)

Imprime dados na porta Serial mas não quebra linha.

Parâmetros

valor – valor que será impresso.

Retorno: Tipo long. Número de bytes escritos. Porém, a leitura desse número é opcional.

Faça você mesmo

Agora que já sabemos como o programa exemplo funciona, mude o código para realizar as seguintes tarefas:

1. Caso a humidade fique acima de 50%, o programa deve imprimir “Humidade acima de 50%”, além do valor da humidade.
2. O programa deve calcular e imprimir a média dos valores da temperatura e da humidade que foram lidos.