

Criando suas próprias bibliotecas para Arduino

Por **Fábio Souza** - 13/06/2014



ÍNDICE DE CONTEÚDO



Este post faz parte da série [Primeiros Passos com Arduino](#). Leia também os outros posts da série:

- [Introdução ao Arduino - Primeiros passos na plataforma](#)
- [Usando os pinos digitais do Arduino](#)
- [Entendendo as Entradas Analógicas do Arduino](#)
- [Usando as saídas PWM do Arduino](#)
- [Arduino - Comunicação Serial](#)
- [Criando suas próprias bibliotecas para Arduino](#)

A plataforma Arduino trouxe a facilidade de criar projetos com microcontroladores, principalmente para iniciantes, devido à abstração de código de baixo nível. Quando você utiliza uma função para escrita ou leitura de um sinal digital, você não precisa se preocupar com a correta manipulação dos registradores internos do microcontrolador que está utilizando em seu projeto. Essa abstração apresenta vantagens, como a portabilidade de código, e desvantagens, como, por exemplo, o possível aumento do código de programa gerado e o consequente atraso para manipulação de sinais.

Outra vantagem de se utilizar bibliotecas é a abstração de hardware, uma vez que você pode manipular o hardware com métodos mais próximos à nossa linguagem.

Nesse tutorial vamos apresentar como criar uma biblioteca para manipular um pino de saída de forma mais alto nível.

As bibliotecas para Arduino são feitas em C++. Inicialmente deve-se criar uma pasta com o mesmo nome que vai ser chamada a biblioteca.

A pasta *examples* conterá os exemplos que apareceram na IDE do Arduino. O Arquivo *keywords.txt* servirá para as palavras da biblioteca mudarem de cor na IDE. Os arquivos *Saida.h* e *Saida.cpp* conterão os códigos da biblioteca. Vamos começar a estruturar nossa biblioteca Saida.

Inicialmente vamos pensar nas ações que uma saída digital poderá demonstrar:

- ligar;
- desligar;
- inverter o seu estado.

Agora temos que editar o arquivo *Saida.h*, que é o arquivo de cabeçalho da nossa biblioteca. Primeiro vamos inserir as seguintes diretivas de compilação:

```
1 #ifndef SAIDA_H
2 #define SAIDA_H
3
4
5
6 #endif
```

Essas diretivas não deixarão as declarações/definições da biblioteca serem inseridas mais de uma vez em um projeto.

Para ter acesso às funções do Arduino, é necessário fazer uso da biblioteca Arduino, inserindo o arquivo de cabeçalho *Arduino.h*:

```
1 #ifndef SAIDA_H
2 #define SAIDA_H
3
4 #include <Arduino.h>
5
6
7
8 #endif
```

Agora vamos criar a classe *Saida*:

```
1 #ifndef SAIDA_H
2 #define SAIDA_H
3
4 #include <Arduino.h>
5
6 class Saida
7 {
8 public:
9     Saida(int pin);
10    void liga();
11    void desliga();
12    void inverte();
13
14 private:
15    int pino;
16 };
17
18 #endif
```

Como pode ser observado, a classe *Saida* possui um construtor que recebe como parâmetro o pino correspondente a saída. Possui também 3 métodos públicos que poderão ser acessados por quem for utilizar a biblioteca e um atributo privado que só poderá ser acessado dentro da classe.

Vamos agora para a codificação dos métodos da classe no arquivo *Saida.cpp*:

```
1 #include "Saida.h"
2
3 Saida::Saida(int pin)
4 {
5     pinMode(pin, OUTPUT);
6     pino = pin;
7 }
8
9 void Saida::liga()
10 {
11     digitalWrite(pino, HIGH);
12 }
13
14 void Saida::desliga()
15 {
16     digitalWrite(pino, LOW);
17 }
18
19 void Saida::inverte()
20 {
```

```
21  digitalWrite(pino, !digitalRead(pino));  
22 }
```

O construtor **Saida::Saida(int pin)** configura o pino passado como parâmetro como saída e depois atribui o seu valor à variável privada *pino*, de modo que esse pino possa ser utilizado pelos métodos da classe futuramente.

Os métodos demonstram as ações que os seus nomes propõem, fazendo uso das funções da biblioteca Arduino.

O arquivo *keywords.txt* deve ficar da seguinte forma:

```
1 Saida KEYWORD1  
2 liga KEYWORD2  
3 desliga KEYWORD2  
4 inverte KEYWORD2
```

O nome da classe deve estar na linha de KEYWORD1 e os métodos serão KEYWORD2.

Agora que nossa biblioteca está pronta, basta adicionar a pasta *libraries* no diretório do Arduino e vamos criar dois exemplos para testes. Esses exemplos devem ser salvos na pasta *examples* da nossa biblioteca.

O primeiro exemplo é SaidaBlink, que consiste em piscar um led em intervalos de 1 segundo:

```
1 #include <Saida.h>  
2  
3 // Instancia um objeto chamado LED no pino 13  
4 Saida LED(13);  
5  
6 void setup(){  
7 }  
8  
9 void loop()  
10 {  
11  LED.liga();           // liga o led  
12  delay(1000);         // aguarda 1 segundo  
13  LED.desliga();       // desliga o Led  
14  delay(1000);         // aguarda 1 segundo  
15 }
```

No segundo exemplo é criada uma saída chamada rele, que inverte seu estado quando uma tecla for pressionada:

```
1  #include <Saida.h>
2
3
4
5  // Cria um rele passando como parâmetro o pino no qual está ligado
6
7  Saida rele(8);
8  const byte tecla = 2;
9
10 // Configura arduino
11
12 void setup()
13 {
14   pinMode(tecla, INPUT);
15 }
16
17
18 // Loop principal
19 void loop()
20 {
21   if (digitalRead(tecla) == LOW)
22   {
23     while (digitalRead(tecla) == LOW); // Aguada tecla ser liberada
24     rele.inverte();
25   }
26 }
```

Conclusão

O exemplo apresentado para criação de bibliotecas foi bem simples e servirá de base para que você possa criar as suas próprias bibliotecas, ou caso você queira entender como uma biblioteca é criada. Você pode adicionar outros métodos a essa biblioteca e testar com o seu hardware, por exemplo, um método que retorne o estado atual de uma saída, etc.

Saiba mais

Arduino - O documentário

Arduino UNO

Referências

Download da biblioteca [↗](#)

<http://arduino.cc/en/Hacking/LibraryTutorial> [↗](#)

<https://www.inkling.com/read/arduino-cookbook-michael-margolis-2nd/chapter-16/recipe-16-4> [↗](#)

Outros artigos da série

[<< Arduino - Comunicação Serial](#)

Este post faz da série [Primeiros Passos com Arduino](#). Leia também os outros posts da série:

- [Introdução ao Arduino - Primeiros passos na plataforma](#)
- [Usando os pinos digitais do Arduino](#)
- [Entendendo as Entradas Analógicas do Arduino](#)
- [Usando as saídas PWM do Arduino](#)
- [Arduino - Comunicação Serial](#)
- [Criando suas próprias bibliotecas para Arduino](#)


NEWSLETTER

Receba os melhores conteúdos sobre sistemas eletrônicos embarcados, dicas, tutoriais e promoções.


E-mail

CADASTRAR E-MAIL

Fique tranquilo, também não gostamos de spam.

Esta obra está licenciada com uma Licença [Creative Commons Atribuição-Compartilhamento 4.0 Internacional](#) .

Fábio Souza

<http://lattes.cnpq.br/5893595257492724> 

Engenheiro, especialista em sistemas embarcados. Hoje é diretor de operações do portal Embarcados, onde trabalha para levar conteúdos de eletrônica, sistemas embarcados e IoT para o Brasil. Também atua no ensino eletrônico e programação. É entusiasta do movimento maker, da cultura DIY e do compartilhamento de conhecimento, publica diversos artigos sobre eletrônica e projetos open hardware. Com iniciativas como o projeto Franzininho e projetos na área de educação, leva a cultura maker para o Brasil capacitando e incentivando professores e alunos a usarem tecnologia em suas vidas. Participou da residência hacker 2018 no Red Bull Basement.



Este site utiliza cookies. Ao usá-lo você concorda com nossos [Termos de Uso](#).
[Saiba mais](#).

Continuar