PROFESSOR DANILO *ROBÓTICA – 8° ANO – 03/10/2020*

FOLHA 03

Guia de referência da linguagem do Arduino®.

#define

Descrição[1]

#define é uma diretiva muito útil da linguagem C++ que permite ao programador dar um nome a um valor constante antes de o programa ser compilado. Constantes definidas no arduino não ocupam nenhum espaço na memória de programa do chip. O compilador irá substituir referências a essas constantes pelo valor definido no tempo de compilação.

Isso pode ter alguns efeitos colaterais desagradáveis no entanto, por exemplo, se o nome de uma constante que foi definida com #defined é incluído em outra constante ou nome de uma variável. Nesse caso o texto seria trocado pelo número (ou texto) definido com #define.

Em geral, a palavra-chave const é recomendada para se definir constantes e deveria ser usada em vez de #define.

Sintaxe

#define nomeDaConstante valor
Note que o # é necessário.

Código de Exemplo

```
#define pinoLED 3
// O compilador irá substituir qualquer menção de pinoLED com o valor 3 no tempo de compilação.
```

Notas e Advertências

Não há ponto e vírgula após a diretiva #define. Se você incluir uma, o compilador irá acusar erros.

```
#define pinoLED 3;  // isso é inválido
```

Similarmente, incluir sinal de igual após #define também resultará em erros

```
#define pinoLED = 3  // também é inválido
```

pinMode

[Digital I/O]

Descrição[2]

Configura o pino especificado para funcionar como uma entrada ou saída. Veja a descrição dos pinos digitais (em Inglês) para mais detalhes sobre a funcionalidade dos pinos.

Desde a versão 1.0.1, é possível ativar os resistores internos de pull-up como o modo INPUT_PULLUP. Adicionalmente, o modo INPUT explicitamente desativa os resistores pull-up internos.

Sintaxe

pinMode(pino, modo)

Parâmetros

pino: the número do pino do Arduino no qual se quer configurar o modo

modo: o modo do pino. Este pode ser INPUT, OUTPUT ou INPUT_PULLUP; que correspondem respectivamente a entrada, saída e entrada com pull-up ativado.

Retorna

Nada

Código de Exemplo

The código configura o pino digital 13 como OUTPUT e troca seu estado entre HIGH e LOW

```
void setup() {
  pinMode(13, OUTPUT);      // configura o pino digital 13 como saída
}

void loop() {
  digitalWrite(13, HIGH); // ativa o pino digital 13
  delay(1000);            // espera por um segundo
  digitalWrite(13, LOW);  // desativa o pino digital 13
  delay(1000);            // espera por um segundo
}
```

Notas e Advertências

Os pinos de entrada analógica podem ser também usados como pinos digitais, referidos como A0, A1, etc.

digitalWrite

[Digital I/O]

Descrição[3]

Aciona um valor HIGH ou LOW em um pino digital.

Se o pino for configurado como saída (OUTPUT) com a função pinMode(), sua tensão será acionada para o valor correspondente: 5V (ou 3.3V em placas alimentadas com 3.3V como o DUE) para o valor HIGH, 0V (ou ground) para LOW.

Se o pino for configurado como entrada (INPUT), a função digitalWrite() irá ativar (HIGH) ou desativar (LOW) o resistor interno de pull-up no pino de entrada. É recomendado configurar pinMode() com INPUT_PULLUP para ativar o resistor interno de pull-up. Veja o tutorial sobre pinos digitais para mais informações.

Se você não configurar o pino com pinMode() e OUTPUT, e conectar um LED ao pino, quando chamar digitalWrite(HIGH), o LED pode aparecer um pouco apagado. Sem configurar explicitamente pinMode(), digitalWrite() irá apenas ativar o resistor de pull-up interno, que age como um grande resistor limitador de corrente.

Sintaxe

digitalWrite(pino, valor)

Parâmetros

pino: o número do pino do Arduino

valor: HIGH ou LOW

Retorna

Nada

Código de Exemplo

The código configura o pino digital 13 como OUTPUT e troca seu estado entre HIGH e LOW

[1] Texto retirado de https://www.arduino.cc/reference/pt/language/structure/further-syntax/define/
[2] Texto retirado de https://www.arduino.cc/reference/pt/language/functions/digital-io/pinmode/
[3] Texto retirado de https://www.arduino.cc/reference/pt/language/functions/digital-io/digitalwrite/

```
void setup() {
  pinMode(13, OUTPUT);    // configura o pino digital 13 como
saída
}
void loop() {
  digitalWrite(13, HIGH); // ativa o pino digital 13
  delay(1000);            // espera por um segundo
  digitalWrite(13, LOW);  // desativa o pino digital 13
  delay(1000);            // espera por um segundo
}
```

Notas e Advertências

Os pinos de entrada analógica podem ser também usados como pinos digitais, referidos como A0, A1, etc. As exceções são os pinos A6 e A7 das placas Arduino Nano, Pro Mini, e Mini, que podem ser usadas apenas como entradas analógicas.

digitalRead

[Digital I/O]

Descrição[4]

Lê o valor de um pino digital especificado, que pode ser HIGH ou LOW.

Sintaxe

digitalRead(pino)

Parâmetros

pino: o número do pino digital do Arduino que você quiser verificar

Retorna

HIGH ou LOW

Código de Exemplo

Aciona o pino 13 para o mesmo valor que o pino 7, declarado como entrada.

```
int ledPin = 13;  // LED conectado ao pino digital 13
int inPin = 7;    // botão conectado ao pino digital 7
int val = 0;      // variável para guardar o valor lido
void setup() {
  pinMode(ledPin, OUTPUT);    // configura o pino digital 13 como
saída
  pinMode(inPin, INPUT);      // configura o pino digital 7 como
entrada
}
void loop() {
  val = digitalRead(inPin);   // lê o pino de entrada
  digitalWrite(ledPin, val);  // aciona o LED com o valor lido do
botão
}
```

Notas e Advertências

Se o pino não está conectado a nada, digitalRead() pode retornar tanto HIGH como LOW (e isso pode mudar aleatoriamente).

Os pinos de entrada analógica podem ser também usados como pinos digitais, referidos como A0, A1, etc. As exceções são os pinos A6 e A7 das placas Arduino Nano, Pro Mini, e Mini, que podem ser usadas apenas como entradas analógicas.

if

[Control Structure]

Descrição[5]

O comando if checa uma condição e executas o comando a seguir ou um bloco de comandos delimitados por chaves, se a condição é verdadeira ('true').

Sintaxe

```
if (condição) {
  //comando(s)
}
```

Parâmetros

condição: uma expressão booleana, isto é, que pode resultar apenas em true ou false

Código de Exemplo

As chaves podem ser omitidas depois de um comando if. Se isso é feito, a próxima linha (definida pelo ponto e vírgula) é interpretada como o único comando condicional. Para mais de um comando, use as chaves para delimitar o bloco de comandos.

```
if (x > 120) {
  digitalWrite(pinoLED, HIGH);
}
if (x > 120) {
  digitalWrite(pinoLED, HIGH);
}
if (x > 120) {
  digitalWrite(pinoLED, HIGH);
}
if (x > 120) {
  digitalWrite(pinoLED1, HIGH);
  digitalWrite(pinoLED2, HIGH);
} // todas as formas acima estão corretas
```

Notas e Advertências

As expressões sendo testadas dentro dos parênteses geralmente requerem o uso de um ou mais dos operadores mostrados abaixo.

Operadores de comparação:

```
x == y (x é igual a y)
x != y (x é diferente de y)
x <  y (x é menor que y)
x >  y (x maior que y)
x <= y (x é menor ou igual a y)
x >= y (x é maior ou igual a y)
```

Cuidado para não usar acidentalmente o símbolo de igual simples (ex. if (x = 10)). O símbolo de igual simples é o operador de atribuição, se atribui 10 a x (coloca o valor 10 na variável x). Em vez disso, o símbolo de igual duplo (ex. if (x == 10)) deve ser usado, que é o operador de comparação, e testa se x é igaul a 10 ou não. O último é apenas verdadeiro se x é igual a 10, enquanto o primeiro comando mostrado sempre resultará em verdadeiro.

Isso acontece porque a linguagem C++ interpreta if (x=10) da seguinte forma: 10 é atribuído a x (Lembre que o símbolo de igual simples é o (operador de atribuição)), então x agora contém 10. Então o comando 'if' testa 10, o que sempre resulta true, desde que qualquer número diferente de zero resulta em true. Consequentemente, if (x = 10) irá sempre resultar em true, o que não é desejável ao se usar um comando 'if'. Além disso, a variável x irá receber o valor 10, o que também é indesejado.

Declaração de variável

Por fim, devemos lembrar que devemos declarar as variáveis que iremos usar. Se quisermos uma variável chamada, por exemplo, *statusBotao*, declaramos da seguinte forma:

int *statusBotao* = 1;

Agora é com você!

[4] Texto retirado de https://www.arduino.cc/reference/pt/language/functions/digital-io/digitalread/

[5] Texto retirado de https://www.arduino.cc/reference/pt/language/structure/control-structure/if/