



++

45697056

# Sistemas para Internet

## Desenvolvimento Mobile e IOT - Android

45697056

Referências da linguagem

45697056

...



Prof. Douglas Cabral <[douglas.cabral@fiap.com.br](mailto:douglas.cabral@fiap.com.br)>  
<https://www.linkedin.com/in/douglascabral/>

A estrutura básica de um sketch é formada por duas funções principais:

**void setup()** - inicialização/configuração inicial. Definimos por exemplo quais serão os pinos utilizados e o modo de uso (INPUT/OUTPUT). Essa função é executada apenas uma vez, ao iniciar o programa.

**void loop()** - executada indefinidamente - ciclo de repetição. Nele inserimos a lógica principal de nosso programa que será executado de forma repentina até que o desligamento do Arduino ocorra.

Para definir uma variável, devemos especificar o tipo dessa variável.

**Ex: declaração de variável inteira.**

`int ledPin = 13;`

Diagram illustrating the components of the variable declaration `int ledPin = 13;`:

- int**: Tipo (Type)
- ledPin**: Nome (Name)
- = 13**: Valor (Value)

Outros tipos de dados disponíveis
boolean
byte
char
int
float
String

Para definir constantes, declaramos a variável normalmente e colocamos no início da declaração a palavra reservada **const**. Por padrão, colocamos os nomes das constantes sempre em **MAIÚSCULO**.

```
const int LED = 13;
```

O Arduino nos disponibiliza algumas constantes pré-definidas, sendo elas:

**HIGH e LOW** - valores referentes à tensão nos pinos digitais. HIGH = 5V e LOW = 0V.

**INPUT e OUTPUT** - definem o estado de um pino. INPUT = entrada e OUTPUT = saída.

**TRUE e FALSE** - referências para valores lógicos, ou seja, verdadeiro e falso respectivamente.

Símbolo	Operação
+	Adição
-	Subtração
*	Multiplicação
/	Divisão
=	Atribuição
++	Incremento
--	Decremento

Símbolo	Operação
<b>+=</b>	Operação composta de adição
<b>-=</b>	Operação composta de subtração
<b>*=</b>	Operação composta de multiplicação
<b>/=</b>	Operação composta de divisão

Símbolo	Operação
==	Igual
!=	Diferente
>	Maior
<	Menor
>=	Maior ou igual
<=	Menor ou igual

Símbolo	Operação
<b>&amp;&amp;</b>	E (AND)
<b>  </b>	OU (OR)
<b>!</b>	NÃO (NOT)



Símbolo	Operação
&	E (AND)
	OU (OR)
~	NÃO (NOT)
^	Ou Exclusivo (XOR)
<<	Deslocamento de bit à esquerda
>>	Deslocamento de bit à direita

Para declaração de arrays utilizamos a seguinte sintaxe com os valores separados por vírgula dentro de um par de chaves:

```
int durations[] = {200, 300, 200, 500, 200, 300, 200, 500};
```

**Para acessar um determinado valor do array:**

`duration[0];` // retornará o valor 200 - o primeiro item do array é sempre o índice 0

Podemos utilizar funções para agrupar comandos que serão reutilizados posteriormente em outros trechos de nosso código. As funções podem ter parâmetros se necessário seguindo a seguinte sintaxe:

<tipo do retorno> nomeDaFuncao([<tipo do parametro 1> nomeDoParametro1, ...])

### Exemplo:

```
void minhaFuncao(int parametro1, int parametro2, boolean parametro3) {  
    // comandos aqui  
}
```

Para comentarmos nosso código-fonte podemos utilizar uma das seguintes formas:

`// Comentários de uma linha`

`/* Comentários de  
múltiplas linhas */`

## Estrutura de controle - if



```
if ( expressao1 ) {  
    //bloco de comandos 1  
} else if ( expressao2 ) {  
    // bloco de comandos 2  
} else {  
    // bloco de comandos 3  
}
```

A instrução “else if” e “else” são opcionais.

## Estrutura de controle - switch



```
switch (variavel) {  
    case valor1:  
        //bloco de comandos 1  
        break;  
    case valor2:  
        //bloco de comandos 2  
        break;  
    default:  
        //bloco de comandos 3  
        break;  
}
```

## Loops - while

O loop while será executado enquanto a condição for verdadeira.

```
while (expressao) {  
    //bloco de comandos  
}
```

## Loops - do while

Executa pelo menos uma vez o bloco de comandos e continua repetindo enquanto a condição for verdadeira.

```
do {  
    //bloco de comandos  
} while (condicao);
```



O loop **for** é utilizado quando sabemos o número de repetições que queremos. A sintaxe é:

```
for (int x = 0; x < 10; x++) {  
    //bloco de comandos  
}
```

++

45697096

# Dúvidas?

◇ ◇ ◇



Copyright © 2017 Prof. Douglas Cabral <[douglas.cabral@fiap.com.br](mailto:douglas.cabral@fiap.com.br)> <https://www.linkedin.com/in/douglascabral/>

Todos direitos reservados. Reprodução ou divulgação total ou parcial deste documento é expressamente proibido sem o consentimento formal, por escrito, do Professor (autor).