



# Javascript

## 3 - Conditionais

< Por Isis =) >





01 { ..

Se... se não

< Executando comandos se uma condição for  
atendida >



} ..



# Utilizando condições

Muitas vezes, precisamos criar um código que só seja executado caso uma condição seja verdadeira.

Para isso, utilizamos o `if`.  
O `if` executa um trecho de código se a condição for atendida.

```
if (a > b) {  
    console.log("a é maior que  
b");  
}
```





# E se não?

O `else` é utilizado após o `if`, e o código dentro dele é executado se a condição for falsa.

O `else` sempre é precedido por um `if`, porque nele não há uma condição própria.

```
if (a > b) {  
    console.log("a é maior que  
b");  
}  
else {  
    console.log("a é menor ou  
igual a b");  
}
```






# Mas eu quero outra opção

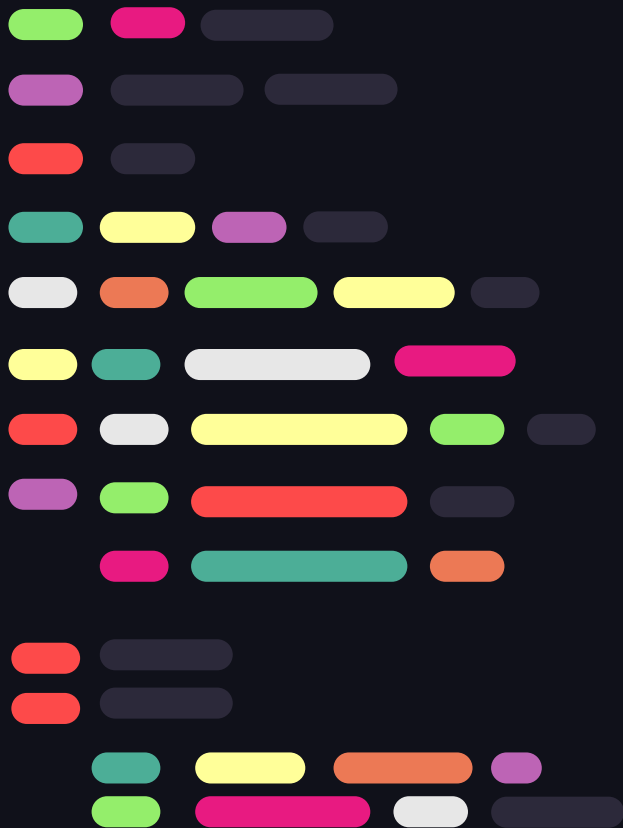
Entre o `if` e o `else`, podemos colocar o `else if`.

Ele funciona como um meio termo, tendo uma condição própria, mas só executando caso as condições anteriores não sejam atingidas.



```
if (a > b) {  
    console.log("a é maior que  
b");  
}  
else if (a > c) {  
    console.log("a é  
menor/igual a b e maior que c");  
}  
else {  
    console.log("a é  
menor/igual a b e c");  
}
```





# Pera! }

< Dependendo da ordem e do colocação dos seus if/else if/else, a execução do código pode variar bastante! >





# Encontre a diferença

```
if (a > b) {  
    console.log("a é maior que  
b");  
}  
if (a > c) {  
    console.log("a é maior que  
c");  
}
```

```
if (a > b) {  
    console.log("a é maior que  
b");  
}  
else if (a > c) {  
    console.log("a é  
menor/igual a b e maior que c");  
}
```





02 { ..

## Switch case

< Quando é necessário realizar comparação do mesmo objeto várias vezes seguidas >

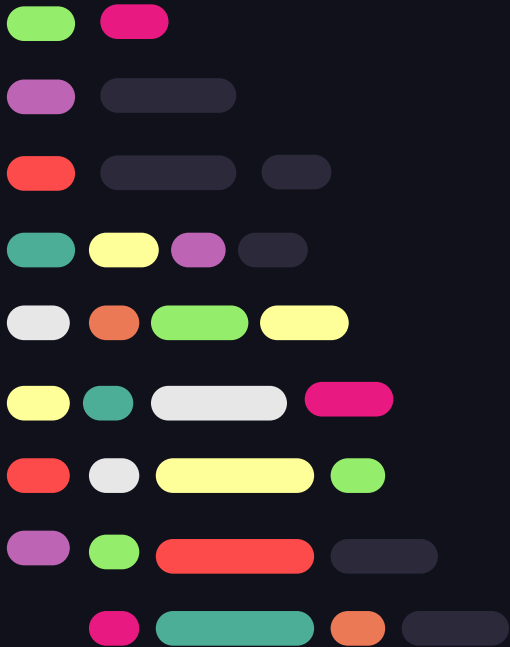


} ..





# Várias comparações



Em casos específicos, comparamos várias e várias vezes uma mesma variável com valores diferentes.

Essa é uma ocorrência tão comum na programação, que foi criado um método específico para esse tipo de condição: o `switch`.






# Switch case

O `switch` é bem simples: basta informar a variável a ser comparada, e cada possível valor que ela pode assumir.

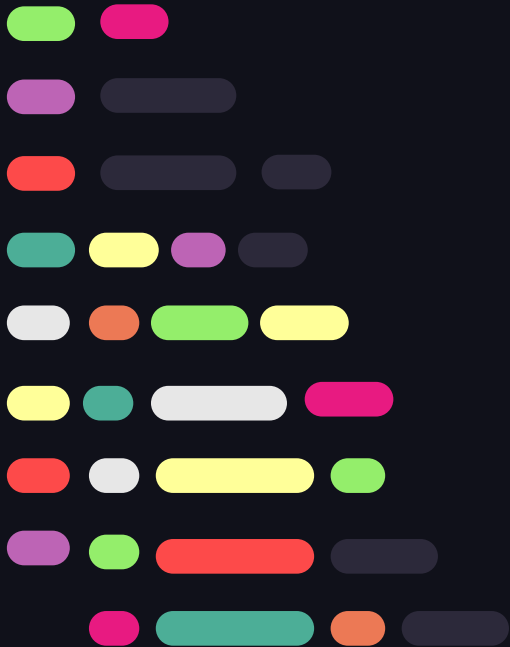
Cada `case` representa uma possível execução, dependendo do valor da variável comparada.

```
switch (a) {  
    case 1:  
        alert("a é 1");  
        break;  
    case 2:  
        alert("a é 2");  
        break;  
    default:  
        alert("nenhum dos  
anteriores");  
}
```





# 0 default



O `default` é a última opção, caso nenhum dos outros valores seja o da variável.

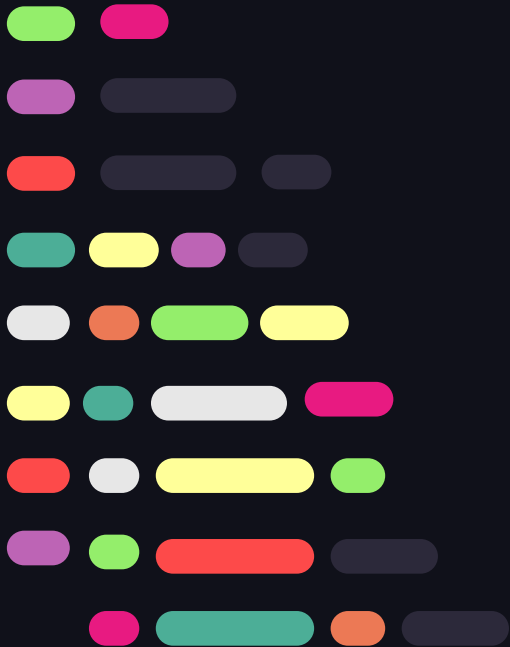
Em funcionamento, ele é bastante similar ao `else`.

Em geral, sempre colocamos o `default`, como boa prática, para não termos pontas soltas no código.





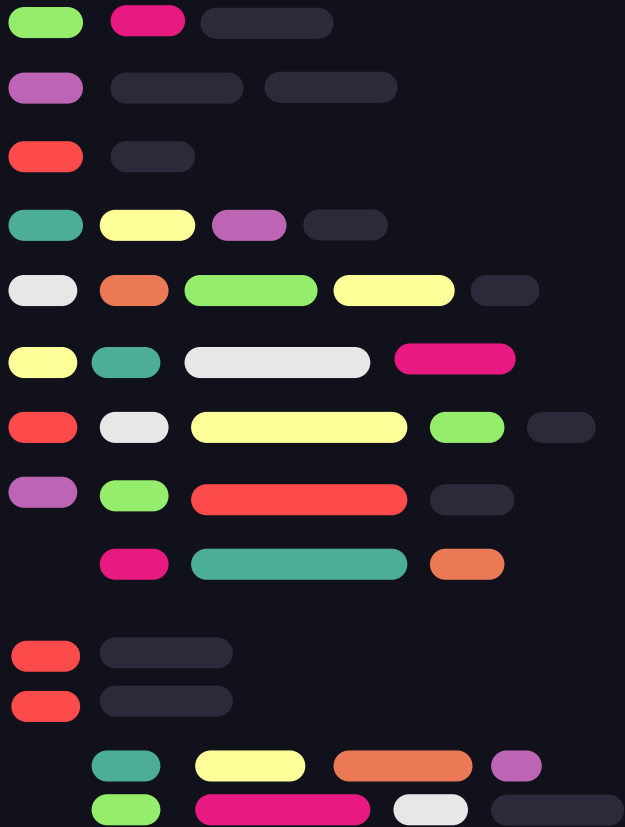
# 0 comando `break`



Se olhar o exemplo anterior, pode ver que, ao final de cada case, há um comando `break`.

Esse comando é responsável por encerrar o `switch`. Se não utilizá-lo, o programa continua executando o que vem abaixo, então é sempre bom lembrar de colocá-lo ao final do case.





# Yey! }

< Com estes conceitos na manga, já  
podemos fazer programas mais  
interessantes ;) >

