



Boolean

O tipo Boolean é primitivo, imutável e representado pelas palavras reservas **true** e **false**

A screenshot of a Mac OS X desktop environment showing a terminal window and a code editor side-by-side.

The terminal window on the right is titled "boolean_1.js — javascriptmasterclass". It shows the command "node" being run, followed by the output of the script:

```
rodrigobranas:javascriptmasterclass $ node
> true;
true
> false;
false
> █
```

The code editor window on the left has a tab bar with "boolean_1.js" selected. The file contains the following code:

```
1 true;
2 false;
3
```

A screenshot of a Mac OS X desktop environment showing a terminal window. The window title is "boolean_2.js — javascriptmasterclass". The left pane contains the code for "boolean_2.js":

```
JS boolean_2.js ×  
1 new Boolean(true);  
2 new Boolean(false);  
3
```

The right pane is a terminal window titled "TERMINAL" with tab "1: node". It shows the output of running the Node.js script:

```
rodrigobranas:javascriptmasterclass $ node  
> new Boolean(true);  
[Boolean: true]  
> new Boolean(false);  
[Boolean: false]  
> █
```

A screenshot of a Mac OS X desktop environment showing a terminal window and a code editor window side-by-side.

The terminal window on the right shows the command `node boolean_3.js` being run, followed by the output "The condition is true".

The code editor window on the left displays the contents of the file `boolean_3.js`:

```
JS boolean_3.js ×
1 let condition = true;
2 if (condition) {
3     console.log("The condition is true");
4 } else {
5     console.log("The condition is false");
6 }
7
```

A screenshot of a Mac OS X desktop environment showing a terminal window and a code editor side-by-side.

The terminal window on the right shows the command `node boolean/boolean_4.js` being run, followed by the output "The condition is false".

The code editor window on the left displays the file `boolean_4.js` with the following content:

```
JS boolean_4.js ×
1 let condition = false;
2 if (condition) {
3     console.log("The condition is true");
4 } else {
5     console.log("The condition is false");
6 }
7
```

A screenshot of a Mac OS X desktop environment showing a terminal window and a code editor side-by-side.

The terminal window on the right shows the command `node boolean/boolean_5.js` being run, followed by the output "The condition is true".

The code editor window on the left displays the file `boolean_5.js` with the following content:

```
1 let condition = new Boolean(true);
2 if (condition) {
3     console.log("The condition is true");
4 } else {
5     console.log("The condition is false");
6 }
7
```

A screenshot of a Mac OS X desktop environment. On the left is a code editor window titled "boolean_6.js — javascriptmasterclass". The file contains the following JavaScript code:

```
JS boolean_6.js x
1 let condition = new Boolean(false);
2 if (condition) {
3     console.log("The condition is true");
4 } else {
5     console.log("The condition is false");
6 }
7
```

The code uses a constructor function to create a Boolean object with the value `false`. An `if` statement then checks this value. Since `Boolean(false)` is `false`, the condition is false, and the code inside the `else` block is executed, printing "The condition is false" to the console.

To the right of the code editor is a terminal window titled "TERMINAL" with a tab labeled "1: bash". The terminal shows the command `node boolean/boolean_6.js` being run, followed by the output "The condition is true".



CAUTION

Cuidado com a coersão de tipo

A coersão de tipo acontece quando um tipo de dado é utilizado em um contexto onde ele é **convertido de forma implícita ou explícita**

A screenshot of a Mac OS X desktop environment showing a terminal window and a code editor. The terminal window is titled 'boolean_7.js — javascriptmasterclass' and contains the command 'node' followed by the contents of the file 'boolean_7.js'. The code editor shows the file 'boolean_7.js' with seven lines of code, each demonstrating the double-not operator (!!) on a different type: 0, NaN, "", false, undefined, and null.

The terminal output is as follows:

```
rodrigobranas:javascriptmasterclass $ node
> !!0;
false
> !!NaN;
false
> !!!";
false
> !!false;
false
> !!undefined;
false
> !!null;
false
> █
```

A screenshot of a macOS terminal window titled "boolean_8.js — javascriptmasterclass". The window has three panes: a left pane with code, a right pane with a terminal session, and a bottom pane which is mostly empty.

The code in the left pane is:

```
1  !!-10;
2  !!!"JavaScript";
3  !!!{};
4  !!![];
5  !!!/JavaScript/;
6  !!!new Date();
7  !!!function () {};
```

The terminal session in the right pane shows the output of running the script with Node.js:

```
rodrigobranas:javascriptmasterclass $ node
> !!-10;
true
> !!!"JavaScript";
true
> !!!{};
true
> !!![];
true
> !!!/JavaScript/;
true
> !!!new Date();
true
> !!!function () {};
true
> 
```