JavaScript Booleans

❮ Previous Next ❯

A JavaScript Boolean represents one of two values: **true** or **false**.

Boolean Values

Very often, in programming, you will need a data type that can only have one of two values, like

- YES / NO
- ON / OFF
- TRUE / FALSE

For this, JavaScript has a **Boolean** data type. It can only take the values **true** or **false**.

The Boolean() Function

You can use the Boolean() function to find out if an expression (or a variable) is true:

Example

```
Boolean(10 > 9)        // returns true
```

Try it Yourself »

Or even easier:

Example

```
(10 > 9)            // also returns true
10 > 9              // also returns true
```

Try it Yourself »

Comparisons and Conditions

The chapter JS Comparisons gives a full overview of comparison operators.

The chapter JS Conditions gives a full overview of conditional statements.

Here are some examples:

Operator	Description	Example
==	equal to	if (day == "Monday")
>	greater than	if (salary > 9000)
<	less than	if (age < 18)

The Boolean value of an expression is the basis for all JavaScript comparisons and conditions.

Everything With a "Value" is True

Examples

```
100

3.14

-15

"Hello"
```

```
"false"

7 + 1 + 3.14
```

Try it Yourself »

Everything Without a "Value" is False

The Boolean value of **0** (zero) is **false**:

```
var x = 0;
Boolean(x);       // returns false
```

Try it Yourself »

The Boolean value of **-0** (minus zero) is **false**:

```
var x = -0;
Boolean(x);       // returns false
```

Try it Yourself »

The Boolean value of **""** (empty string) is **false**:

```
var x = "";
Boolean(x);       // returns false
```

Try it Yourself »

The Boolean value of **undefined** is **false**:

```
var x;
Boolean(x);        // returns false
```

Try it Yourself »

The Boolean value of **null** is **false**:

```
var x = null;
Boolean(x);        // returns false
```

Try it Yourself »

The Boolean value of **false** is (you guessed it) **false**:

```
var x = false;
Boolean(x);        // returns false
```

Try it Yourself »

The Boolean value of **NaN** is **false**:

```
var x = 10 / "H";
Boolean(x);        // returns false
```

Try it Yourself »

Booleans Can be Objects

Normally JavaScript booleans are primitive values created from literals:

var x = false;

But booleans can also be defined as objects with the keyword new:

var y = new Boolean(false);

Example

```
var x = false;
var y = new Boolean(false);

// typeof x returns boolean
// typeof y returns object
```

Try it yourself »

Do not create Boolean objects. It slows down execution speed.
The **new** keyword complicates the code. This can produce some unexpected results:

When using the == operator, equal booleans are equal:

Example

```
var x = false;
var y = new Boolean(false);

// (x == y) is true because x and y have equal values
```

Try it Yourself »

When using the === operator, equal booleans are not equal, because the === operator expects equality in both type and value.

Example

```
var x = false;
var y = new Boolean(false);

// (x === y) is false because x and y have different types
```

Try it Yourself »

Or even worse. Objects cannot be compared:

Example

```
var x = new Boolean(false);
var y = new Boolean(false);

// (x == y) is false because objects cannot be compared
```

Try it Yourself »

Note the difference between (x==y) and (x===y).
Comparing two JavaScript objects will always return false.

Complete Boolean Reference

For a complete reference, go to our Complete JavaScript Boolean Reference.

The reference contains descriptions and examples of all Boolean properties and methods.

❮ Previous Next ❯