# Investigación: Valores Truthy y Falsy en Javascript.

Rodrigo Emmanuel Esquivias Quirarte
e-mail: rodrigo.esquivias@alumnos.udg.mx

**RESUMEN:** *Es bien conocido que en los lenguajes de programación, tenemos los valores "booleanos" verdadero (True) y falso (False) entonces puede surgir la pregunta, ¿A qué se refiere entonces truthy y falsy? En palabras sencillas podemos decir que cada elemento del lenguaje tiene un valor booleano intrínseco-primitivo.*

**PALABRAS CLAVE**: Falsy, False, Javascript, Primitivo, Truthy, True.

## 1 INTRODUCCIÓN

En JavaScript las variables están débilmente y dinámicamente tipeadas es decir al lenguaje no le importa cómo se declaran o cambian de valor:

```
var a;
a = 1;   // a es un número.
a = '1'; // a es un string.
a = [1];  // a es un arreglo.
```

Valores aparentemente "diferentes" equivalen a verdadero cuando se compara con == porque JavaScript convierte cada valor a una representación de cadena antes de la comparación:

```
// Todos verdaderos.
100 == '100';
100 == [100];
'100' == [100];
```

En cambio si utilizamos el operador de igualdad estricto (===) se obtendrán resultados falsos porque se considera el tipo de dato de los valores:

```
// Todos falsos
100 === '100';
100 === [100];
'100' === [100];
```

Internamente, JavaScript establece un valor de uno de los seis tipos de datos primitivos para nuestras variables:

Undefined (una variable sin valor definido).
Null (un simple valor nulo)
Boolean (verdadero [true] o falso [false])
Números (incluyendo infinito [infinity] y no número [NaN])
String (datos de texto)
Symbol (tipo de dato cuyo valor es único e inmutable)

## 2 ¿Qué es Truthy y Falsy?

Además de un tipo, cada valor también tiene un valor booleano inherente esto es que es esencial y permanente, que forma parte de su naturaleza, generalmente conocido como truthy o falsy. Algunas de las reglas son un tanto extrañas, así que comprender los conceptos y el efecto de la comparación ayuda a la hora de utilizar estos valores.

Los siguientes valores son siempre falsy:

false.
0 (cero)
" ó "" (string vacío)
null.
undefined.
NaN (E.j el resultado de 1/0)

Todo lo demás es truthy, eso incluye:

'0' (una cadena que contenga un simple 0)
'false' (un string que contenga el texto "false")
[] (un arreglo vacío)
{} (un objeto vacío)
function(){} (una función vacía)

Por lo tanto, se puede usar un único valor dentro de las condiciones, por ejemplo:

```
if (valor) {
  // el valor es truthy.
}
else {
  // el valor es falsy
  // podría ser false, 0, '', null, undefined ó NaN.
}
```

Las reglas son:

False, cero y cadenas vacías son todas equivalentes.

Null y undefined son equivalentes a ellos mismos y entre ellos, pero nada más.
NaN no es equivalente a nada, incluido otro NaN.
Infinity es truthy, pero no se puede comparar con verdadero o falso.
Un arreglo vacío es truthy, peo si lo comparamos con true nos da false, y si se compara con false nos da true.

Por ejemplo:

```
// Todos verdaderos
false == 0;
0 == '';
null == undefined;
[] == false;
!![0] == true;

// Todos falsos.
false == null;
NaN == NaN;
Infinity == true;
[] == true;
[0] == true;
```

Los valores truthy y falsy pueden dar dolores de cabeza incluso a los desarrolladores más experimentados.

A continuación, algunos ejemplos cuando resultan especialmente útil estos valores.

Raramente es necesario comparar dos valores de truthy y falsy cuando un único valor siempre equivale a verdadero o falso:

```
// en lugar de
if (a == false) // ...
// Se ejecuta si a es false, 0, '', ó []

// Use
if (!a) // ...
// Se ejecuta si a es false, 0, '', NaN, null ó undefined.
```

Utilice una === igualdad estricta (o estricta desigualdad !==) para comparar valores y evitar problemas de conversión de tipo:

```
// En lugar de
if (a == b) // ...
// Se ejecuta si a y b son ambos truthy ó ambos falsy.
// Por ejemplo a = null y b = undefined
// Use
if (a === b) // ...
// Se ejecuta si a y b son idénticos...
// Excepto cuando ambos son NaN.
```

Cualquier valor puede convertirse a un valor booleano real usando una doble negación. Esto nos dará la certeza de que un falso se genera solo por falso, 0, “”, nulo, indefinido y NaN:

```
// En lugar de
if (a === b) // ...
// Se ejecuta si a y b son idénticos...
// Excepto cuando ambos son NaN.

// Use
if (!!a === !!b)
// Se ejecuta si a y b son idénticos...
// incluso cuando cualquiera o ambos son NaN.
```

## 3 REFERENCIAS

[1] 36 Javascript – Coerción y valores Truthy & Falsy – Aprende a Programar – Codejobs. (2015). Codejobs.biz. Consultado el 3 de diciembre de 2017, en https://www.codejobs.biz/es/blog/2015/08/12/36-javascript-coercion-y-valores-truthy-falsy

[2] Valores Truthy y Falsy. (2016). La Espora Del Hongo. Consultado el 3 de diciembre de 2017, en http://laesporadelhongo.com/valores-truthy-falsy/

[3] Buckler, C. (2017). Truthy and Falsy: When All is Not Equal in JavaScript. SitePoint. Consultado el 3 de diciembre de 2017, en https://www.sitepoint.com/javascript-truthy-falsy/

[4] Sergio Rodriguez Loza. (2018). Valores Truthy y Falsy en Javascript. No todo es lo que parece. 16/02/2018, de michelletorres.mx Sitio web: http://michelletorres.mx/valores-truthy-y-falsy-en-javascript-no-todo-es-lo-que-parece/