ThePower business school

🇪🇸 Numbers JS 2️⃣🤖

Los datos de tipo primitivo **number** representan valores numéricos enteros, negativos y decimales.

```
const age = 30;
const temperature = -5;
const budget = 100.50;
```

Métodos

Estos son algunos de los métodos más utilizados con el tipo de dato **number**.

isNaN: El método **isNaN** determina si el valor es de tipo number o no, recordemos que NaN es “Not a Number”. Si no es un número devolverá true, y si detecta un número o un valor convertible a número devolverá false.

```
isNaN(true);   // true
isNaN("hola"); // false
isNaN(10);     // false
isNaN("12");   // false
```

isInteger: El método **isInteger** nos indica a través de un booleano si el valor pasado es un número entero, si no nos devolverá false.

```
const integer = 10;
const decimal = 12.5;

Number.isInteger(integer); //true
Number.isInteger(decimal); //false
```

parseInt: El método **parseInt** convierte un valor de tipo string a número entero.

```
const age = "18";
const ageToNumber = parseInt(age);
console.log(ageToNumber);
// Retorna 18 en formato number
```

parseFloat: El método **parseFloat** es muy parecido al parseInt, pero convierte si fuera necesario un valor de tipo string a un número decimal si encontrará un punto en el mismo.

```
const budget = "1.250";
const budgetInFloat = parseFloat(budget);
console.log(budgetInFloat);
// Retorna 1.25 en número decimal
```

toString: El método **num.toString(base)** devuelve la representación **numérica** en una cadena, en el sistema numérico con la **base** especificada.

```
let num = 255;

alert( num.toString(16) );  // ff
alert( num.toString(2) );   // 11111111
```