# Anotaciones Javascript

## Javascript Types

Son los diferentes tipos de contenidos que javascript puede recibir y leer, son la unidad más básica que existe en el motor del código, también se le llaman ***primitive types***:

- **Number**(Los números que normalmente se usan)
- **String**(Lo que conocemos como texto, javascript usa caracteres especiales para texto)
- **Boolean**(Lógica comparativa a lo que decimos true(verdadero) o false(falso), también el código lo toma como 1 ó 0, como algo con valor o algo sin valor o vacío).
- **Undefined**(es la manera de javascript de avisar que un archuvo o variable no tiene un valor definido o está vacío cuando no debería)
- **Null**(Es lo mismo que undefined pero es usado de manera voluntaria por parte del programador para asignar como vacío o nulo deliberadamente un objeto, así le hace saber al motor de javascript y al programador que analice después el código que ese valor(Null) fué desginado por el programador por alguna razón).
- **Object**(significa objeto, )

## Data Structures / Estructura de datos

- Arrays
- Objects

## Arrays

Las arrays son listas en javascript que pueden guardar una serie de elementos de diferente tipo, aunque es recomendado guardar solo elementos del mismo tipo en una sola lista, generalmente mezclar elementos de diferente tipo en una sola lista puede generar conflictos en la lógica del programa.

#### Javascript Array Methods / Métodos

Algunos métodos a tener en cuenta para agregar, remover, concatenar, cambiar,etc, las arrays:

- *splice()*, se escribe primero el nombre de la variable que contiene el array, más u punto seguido, e inmediatamente se escribe '*splice*' seguido de unos paréntesis en donde se escribe el código para remover o agregar uno o varios elementos de la lista o array, dentro de los paréntesis se escribe en orden de izquierda a derecha:
  `array.splice( 4, 0, 'taste');`
  el primer caracter tiene que ser un número indica en que lugar se quiere agregar o remover contando desde 0 en adelante en la posición de los itms de la lista, el siguiente caracter es otro número que indicará cauántos items

de la lista se borrarán, si no se desea borrar ningún item de la lista, simplemente escribir 0, después del segundo número se puede escribir entre paréntesis el item que se quiere agregar en la lista, el código quedaría así:

```
let array = ['Banana', 'Apples', 'Oranges',  'Bluberries'];

array.splice( 4, 0, 'taste');

console.log(array);
```

- *sort();*, Se usa para ordenar una lista o array para que sea de manera alfabética o numérica de forma ascendente de menos a más (ejm. `[1,2,3,4,etc][a,b,c,d,f,etc]`).
- *`array.reverse();`* , se usa para ordenar al revés una lista o array de más a menos.
- *`array.sort((a, b) => a > b);`* , se usa para reordenar de la forma deseada una lista o array, se escribe el nombre de la variable que contiene el array luego un punto, luego el nombre sort más el resto entre paréntesis, las letras a y b dentro del primer paréntesis son para indicar el orden primero en que se desea ordenar los items de la lista, y las letras a > b, se usan para indicar el orden final o el que se verá reflejado en el resultado, el signo mayor que(>) es para indicar hacis donde va el orden de la lista, a > b indicaría que la lista va de manera ascendente de primero hacia abajo desde la primera letra del alfabeto hasta la última, indicando un orden progresivo dependiendo del tipo de items en la lista esto significa un orden ascendente como de 1 a 100, si se cambia de b < a significa que queremos que la lista esté ordenada desde la última letra del alfabeto o item de la lista hasta la primera.

Ejercicios usando **array properties**

Usar la siguiente variable:

```
const array = ['Banana', 'Apples', 'Oranges', 'Bluberries'];
```

- Remover el item Banana del array.
- Ordenar el array usando sort.
- Poner 'Kiwi' al final del array.
- Remover 'Apples' del array.
- Ordenar al revés el array.(No en orden alfabetico, solo al revés ejm: ['a', 'c', 'b'] al revés sería ['b', 'c', 'a'])
  Debería dar como resultado al final:
  ['Kiwi', 'Oranges', 'Blueberries']
- Usando el siguiente array acceder a 'Oranges':

```
const array2 = ["Banana", ["Apples", ["Oranges"], "Blueberries"]];
```

Resultado:

```
const array = ['Banana', 'Apples', 'Oranges',  'Bluberries'];
const array2 = ["Banana", ["Apples", ["Oranges"], "Blueberries"]];

array.splice(0, 1);
array.sort((a, b) => a > b);
array.splice(3, 0, 'Kiwi');
array.splice(0, 1);
array.reverse();

console.log(array); // -> Array(3) [ "Kiwi", "Oranges", "Bluberries" ]
console.log(array2[1][1]); // -> Array [ "Oranges" ]
```

# Objects

Los objects(objetos), son parecidos a las arrays, pues almacenan una lista de varios items, la diferencia es que en objects se pueden guardar una propiedad y un valor por cada item. ejm:

arrays:

```
const list = ['apple', 'banana', 'orange'];  // -> solo guarda valores por item.
```

objects:

```
const user = {
    name: 'Jhon', // -> guarda propiedades(name) y valores para cada propiedad('Jhon')
    age: 34,
    hobby: 'soccer',
};  ````
```