Few JavaScript Object Functions

Object.assign()

Object.assign() is a javascript version of the common "extend" function that copies properties from one or more source objects into a single target object.

Example:

```
const obj1 = { a: 1, b: 2}
const obj2 = {c: 3, d: 4}
const destObj = Object.assign( { }, obj1, obj2)
console.log(destObj) // {a: 1, b:2, c: 3, d: 4}
```

The first parameter in the function is an empty object({ }), that describes the return value is an object

- ★ Object.assign() works through the source objects from **Left-to-Right** order. ⟨ Object.assign({ }, obj1, obj2) ⟩
- ★ Due to the **Left-to-Right process,** if two or more source objects have a value with the same key property**,** the last from the right object value will be included.

```
const obj1 = { a: 1, b: 2}
const obj2 = {b: "new b", c: 3}
const destObj = Object.assign( { }, obj1, obj2)
console.log(destObj) // {a: 1, b: "new b", c: 3}
```

- ★ Object.assign() skips if there is any **null** or **undefined** value in the parameters

Object.is()

Object.is() compares two values and returns either **true** or **false.**
It is similar to the JavaScript's strict equality (===) comparison except, the below 2 use cases

NaN

```
( Nan === Nan )          //false
Object.is(NaN, NaN) // true
```

Signed 0 (-0/+0)

```
( +0 === -0 )      // true
Object.is(+0, -0) //false
```

Examples:

```
Object.is( 10, 10 ).  // true
Object.is(null, null) // true

const obj1 = { a: "a" }
const obj2 = { b: "b" }
 Object.is (obj1, obj2) // false
 Object.is (obj1, obj1) // true

Object.is(null, undefined) // false
```

NaN use case

```
Object.is(NaN, 0/0) // true
Object.is(NaN, NaN) // false
Object.is(NaN, Number.Nan) // true
```

Signed Zero use case

```
Object.is(0, -0) // false
Object.is(+0, 0) // true
Object.is(+0, -0) // false
Object.is(-0, 0) // false
```

Object.keys()

Object.keys() returns the array of all the key properties from the given object.

Example:

```
const obj1 = { a: 1, b: 2, c: 3, d: 4 }
console.log(Object.keys(obj1)).
 // [ a, b, c, d ]
```

Object.values()

Object.values() returns the array of all the value propertys from the given object.

Example:

```
const obj1 = { a: 1, b: 2, c: 3, d: 4 }
console.log(Object.values(obj1)).
 // [ 1, 2, 3, 4 ]
```

Object.entries()

Object.entries() converts a given object into an array of ["key", value] type arrays.
In each array, the first element is the property key (which is always a string).
the second element is the property value.

Example:

```
const obj1 = { a: 1, b: 2, c: 3 }
console.log(Object.entries(obj1))
// [ ["a", 1], ["b", 2], ["c", 3] ]
```

Object.fromEntries()

Object.fromEntries() is converse of **Object.entries,** which takes list of key/value pair arrays and convert them into an object

Example:

```
console.log(Object.fromEntries( [ ["a", 1], ["b", 2], ["c", 3] ] ))
// { a: 1, b: 2, c: 3 }
```