Object

What is the purpose of **Object.keys()**, **Object.values()**, and **Object.entries()**?

**Object.keys()**: This method returns an array of the object's own enumerable property keys. You can then iterate over the array.

Ex : const obj = {a: 1, b: 2, c: 3};

const keys = Object.keys(obj);

keys.forEach((key) => {

console.log(key, obj[key]);

});

**Object.values()**: This method returns an array of the object's own enumerable property values. You can iterate over the array.

Ex : const obj = {a: 1, b: 2, c: 3};

const values = Object.values(obj);

values.forEach((value) => {

console.log(value);

});

**Object.entries()**: This method returns an array of the object's own enumerable [key, value] pairs. You can iterate over the array.  
Ex: const obj = {a: 1, b: 2, c: 3};

const entries = Object.entries(obj);

entries.forEach(([key, value]) => {

console.log(key, value);

});

Structured Clone

**structuredClone** is a method in JavaScript that creates a deep copy (clone) of an object or value. It is useful when you need to duplicate complex data structures like nested objects, arrays, Maps, Sets, and other types without copying the reference to the original data. This means the clone and the original object are independent of each other.

### Key Features of structuredClone:

* **Deep Copy**: The method performs a deep copy of the object, meaning it clones not only the top-level object but also any nested objects or arrays within it.
* **Supports Various Data Types**: It can clone objects, arrays, dates, RegExp, Maps, Sets, and more.
* **No Shared References**: The cloned object is completely independent of the original, so changes to the clone do not affect the original object, and vice versa.
* **Handling Circular References**: It can handle objects with circular references (where an object refers back to itself)

Ex: const originalObject = {

a: 1,

b: {

c: 2

},

d: [3, 4, 5]

};

const clonedObject = structuredClone(originalObject);

// Modifying clonedObject does not affect originalObject

clonedObject.b.c = 42;

clonedObject.d.push(6);

console.log(originalObject.b.c); // 2

console.log(originalObject.d); // [3, 4, 5]