JAVASCRIPT

CHEAT-SHEET FOR OBJECTS

Method	Description

object.create()	The Object.create() method creates a new object, using an existing object as the prototype of the newly created object.
	ex.
	const person = {
	name : ‘John’
	age: 20
	};
	const p1 = Object.create(person);
	p1.name = 'Rutvik';
	p1.age = 21;
	
	output: {name : ‘Rutvik ’ ,age :21}

Object.assign()	The Object.assign() method copies all enumerable own properties from one or more source objects to a target object. It returns the modified target object.
	
	const target = { a: 1, b: 2 };
	const source = { b: 4, c: 5 };
	const returnedTarget = Object.assign(target, source);
	console.log(target);
	output: Object { a: 1, b: 4, c: 5 }
	console.log(returnedTarget);
	output: Object { a: 1, b: 4, c: 5 }

Object.keys()	The Object.keys() method returns an array of a given object's own enumerable property names.
	
	const object1 = {
	a: 'somestring',
	b: 42,
	c: false
	};
	console.log(Object.keys(object1));
	output: Array ["a", "b", "c"]

Object.values()	The Object.values() method returns an array of a given object's own enumerable property values.
	const object1 = {
	a: 'somestring',
	b: 42,
	c: false
	};
	console.log(Object.values(object1));
	// expected output: Array ["somestring", 42, false]

Object.entries()	The Object.entries() method returns an array of a given object's own enumerable string-keyed property [key, value] pairs.
	
	const object1 = {
	a: 'somestring',
	b: 42
	};
	for (const [key, value] of Object.entries(object1)) {
	console.log(`${key}: ${value}`);
	}
	output:
	// "a: somestring"
	// "b: 42"

Object.getOwnProperty Names()	- The Object.getOwnPropertyNames() method returns an array of all
	- Non-enumerable properties are the properties which we can't access using iteration.
	properties (including nonenumerable properties) found directly in a given object.
	
	const object1 = {
	a: 1,
	b: 2,
	c: 3
	};
	console.log(Object.getOwnPropertyNames(object1));
	// expected output: Array ["a", "b", "c"]

Object.freeze()	The Object.freeze() method freezes an object. A frozen object can no longer be changed; freezing an object prevents new properties from being added to it, existing properties from being removed.
	
	const obj = {
	prop: 42
	};
	Object.freeze(obj);
	obj.prop = 33;
	// Throws an error in strict mode
	console.log(obj.prop);
	// expected output: 42

Object.fromEntries()	The Object.fromEntries() method transforms a list of key-value pairs into an object.
	
	const entries = new Map([
	['foo', 'bar'],
	['baz', 42]
	]);
	const obj = Object.fromEntries(entries);
	console.log(obj);
	// expected output: Object { foo: "bar", baz: 42 }