**Map() object in JavaScript**

Map objects are collections of key-value pairs. A key in the Map is unique & may only occur once in the Map's collection.

**Objects vs. Maps** :-

Object is similar to Map—both let you set keys to values, retrieve those values, delete keys, and detect what is stored at a key.

**Differences that make Map preferable in some cases as below**:

**1. Accidental Keys** -

A Map does not contain any keys by default.

An Object has a prototype, so it contains default keys that could collide with your own keys if you're not careful.

**2. Key Types** -

A Map's keys can be any value (including functions, objects, or any primitive).

The keys of an Object must be either a String or a Symbol.

**3. Key Order** -

The keys in Map are ordered in a simple, straightforward way (in the order of entry insertion)

Although the keys of an ordinary Object are not ordered.

**4. Size** -

The number of items in a Map is easily retrieved from its size property.

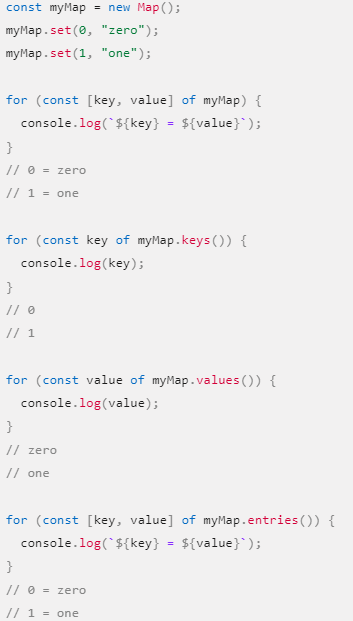
A common way to do it is through the length of the array returned from **"Object.keys(objname).length"**

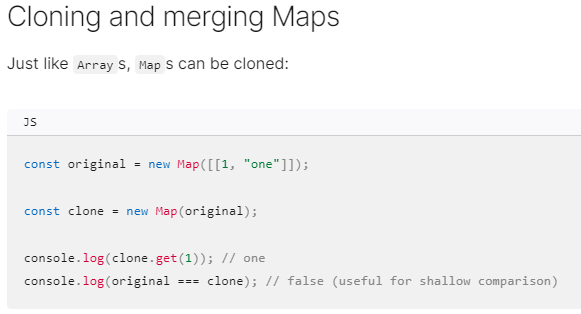
**(ref -** [**https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global\_Objects/Map**](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/Map)**)**

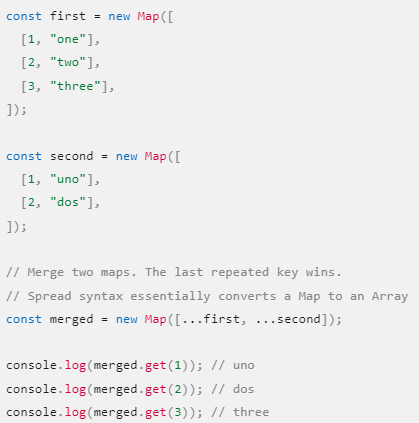
**(Local ref - C:\Users\DESKTOP\OneDrive\Desktop\June 2024 job Plan\Javascript + React July 2024\Javascript\Map.js)**













---------------------------------------------------------- End of Map()------------------------------------------ -------------------

**Objects in JavaScript**

Object is a Non-primitive data type in JavaScript used to store various keyed collections and more complex entities.

Nearly all Objects in JavaScript are instance of Object; that inherits properties from "**Object.prototype**", although these properties may be shadowed (a.k.a. overridden).

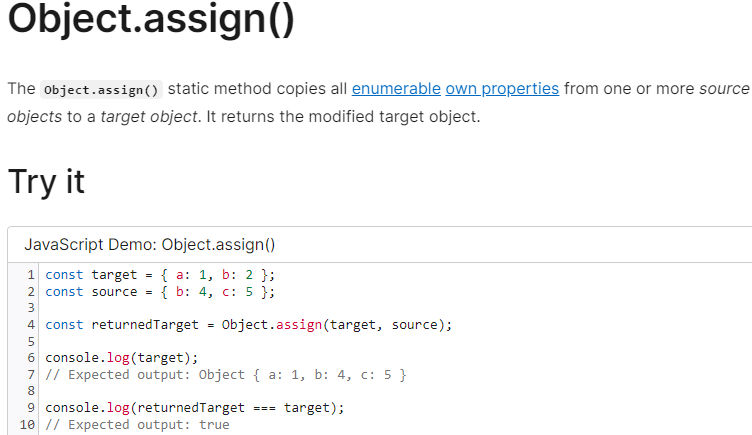
The only objects that don't inherit from Object.prototype are those with **null-prototype**, or descended from other null prototype objects.

Ex - const obj = Object.create(null); const obj2 = { \_\_proto\_\_: null };

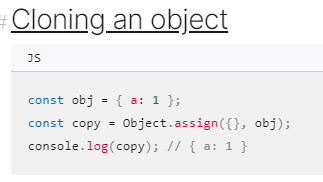
***Note: An object with a null prototype can behave in unexpected ways, because it doesn't inherit any object methods from “Object.prototype”.***

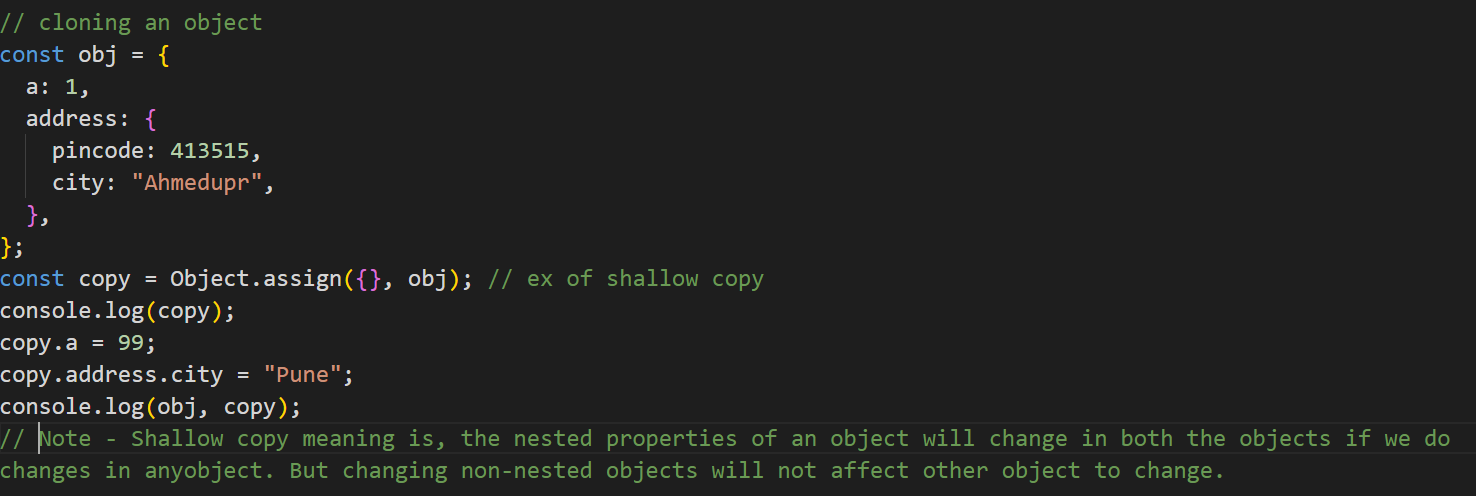
***(ref -*** [***https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global\_Objects/Object***](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/Object)***)***

***(Local ref - C:\Users\DESKTOP\OneDrive\Desktop\June 2024 job Plan\Javascript + React July 2024\Javascript\Objects.js)***

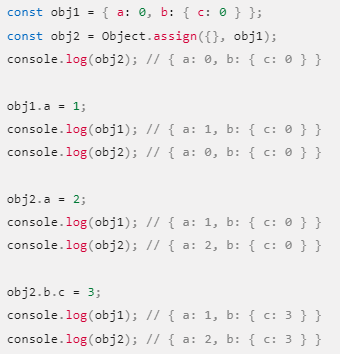
******

***--------------------------------***

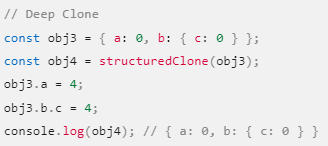
******

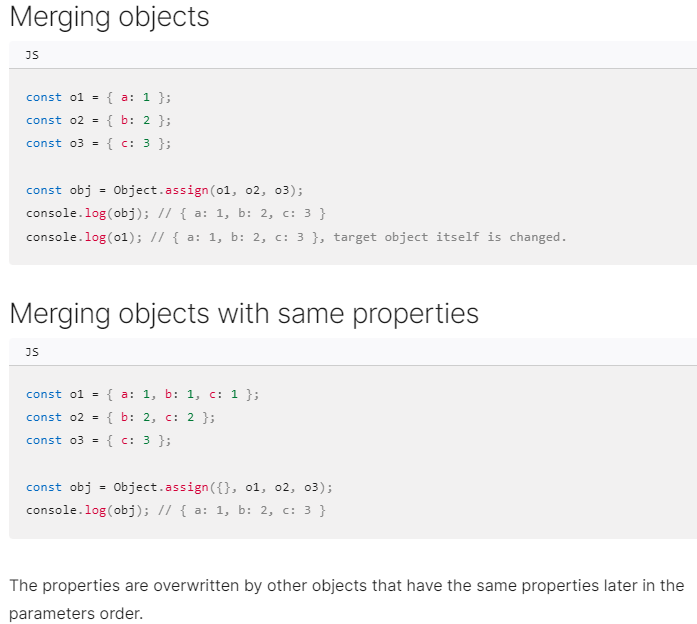
******

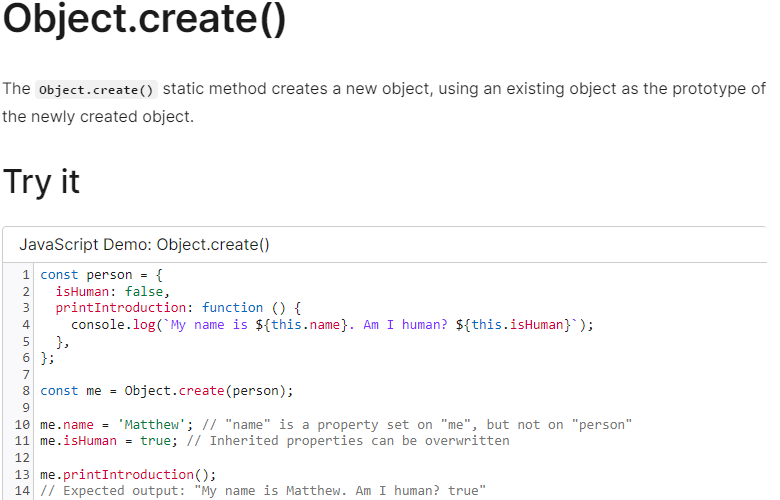
***In this ex 99 will be changed to copied obj whereas city will change in both the objects***

******

***Deep clone example see below ->***

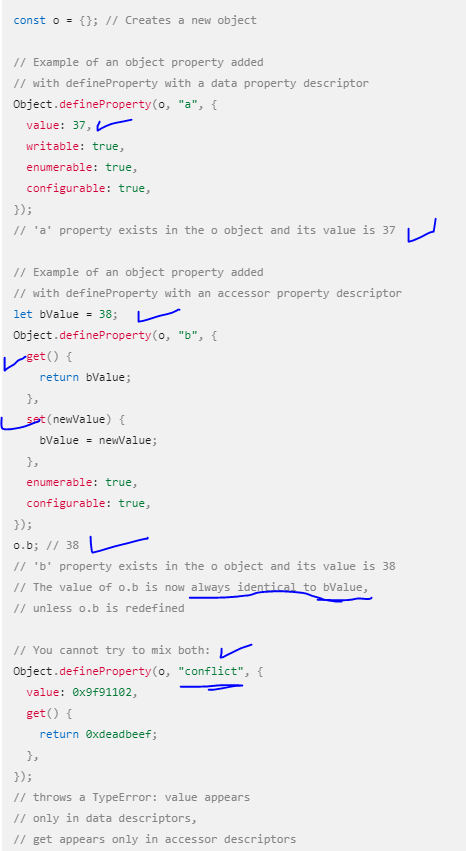
******

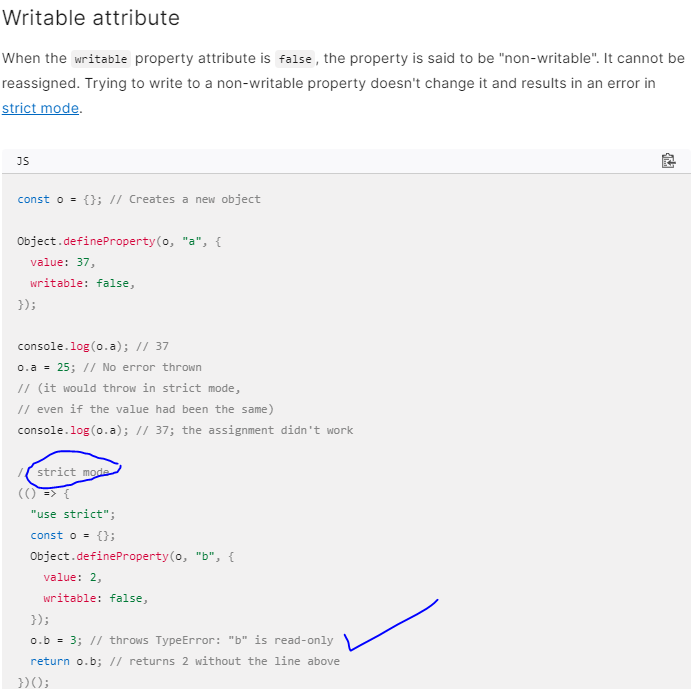
******

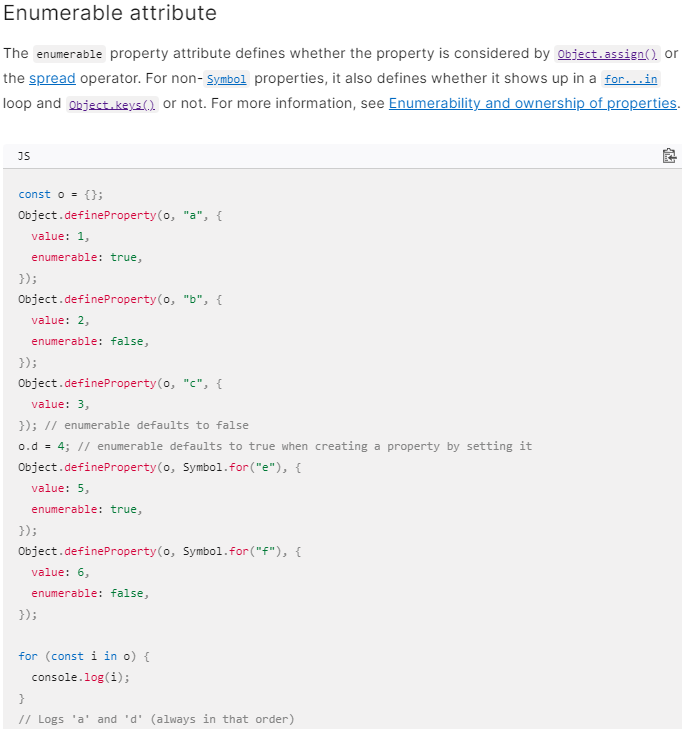
******

******

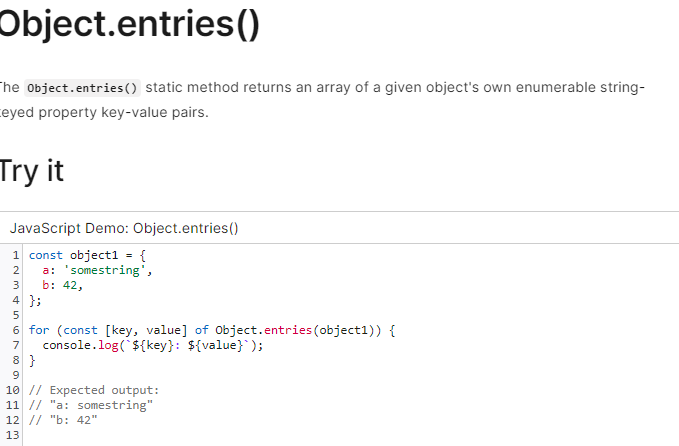
******

******

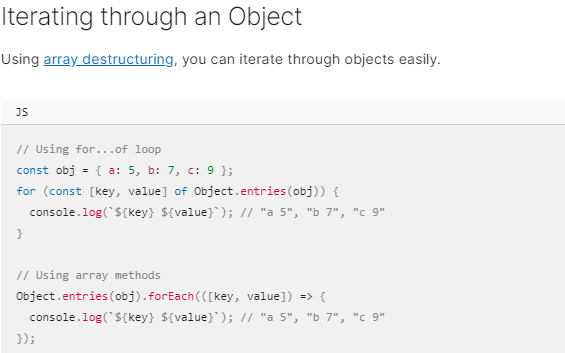
******

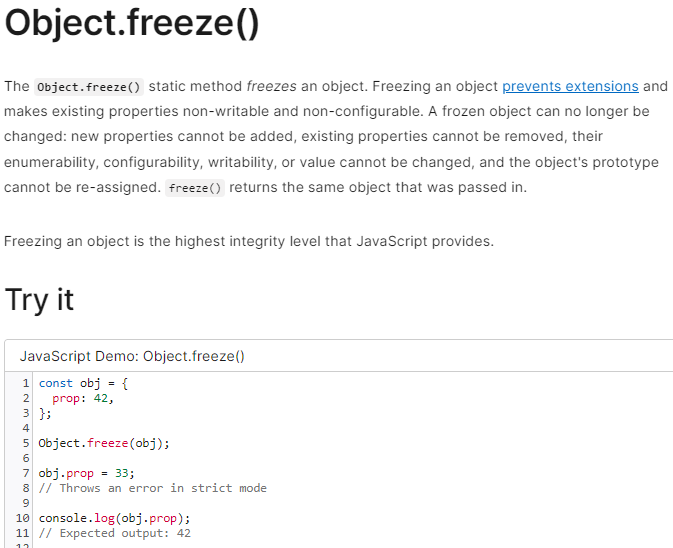
******

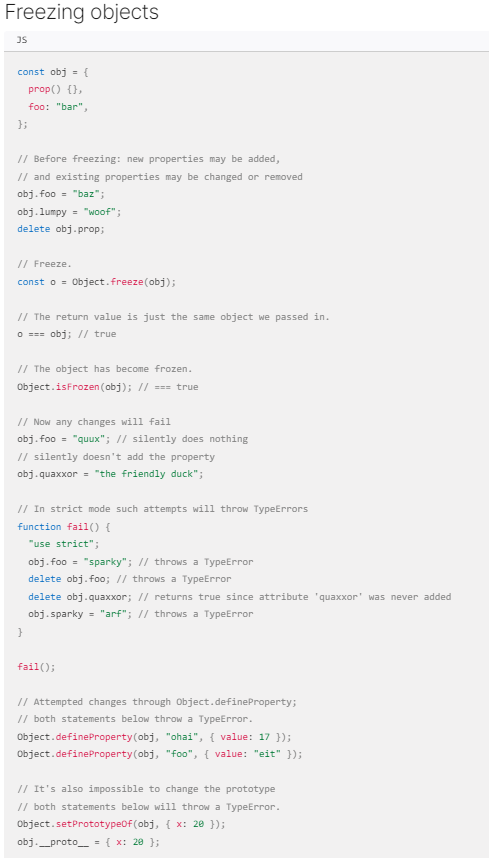
******

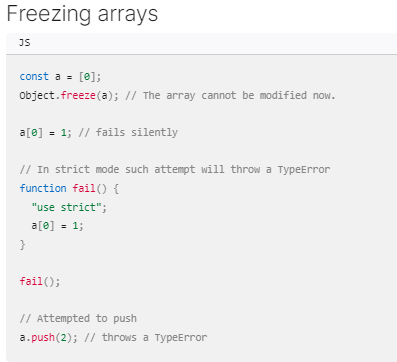
******

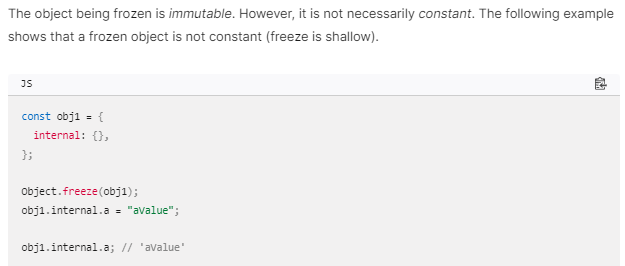
******

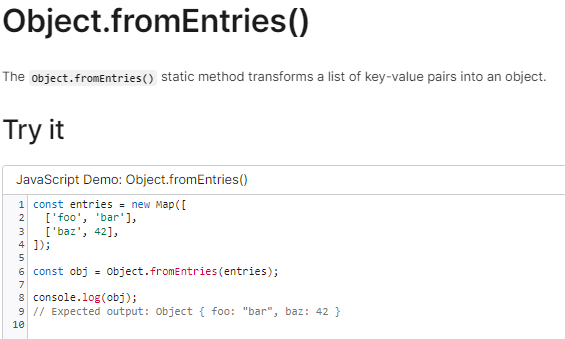
******

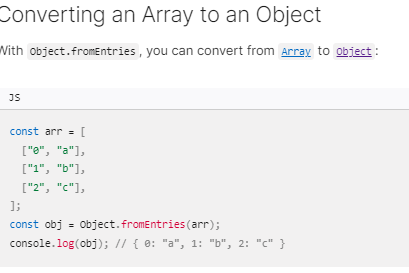
******

******

******

******

******

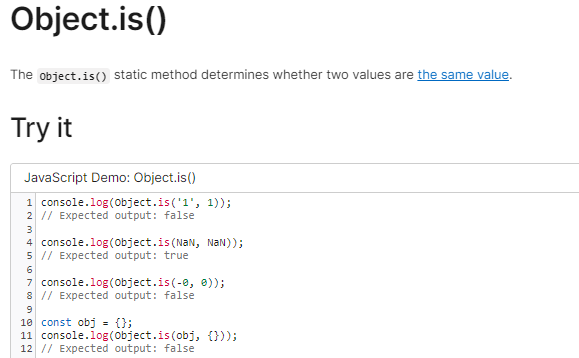
******

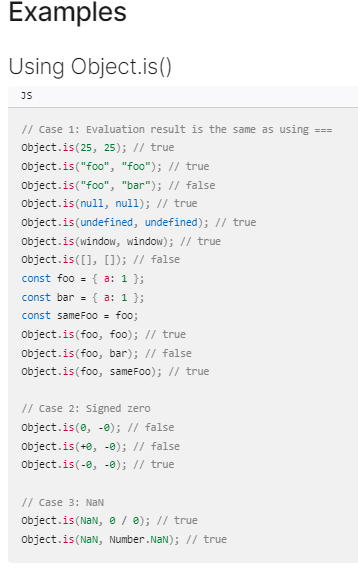
******

******

******

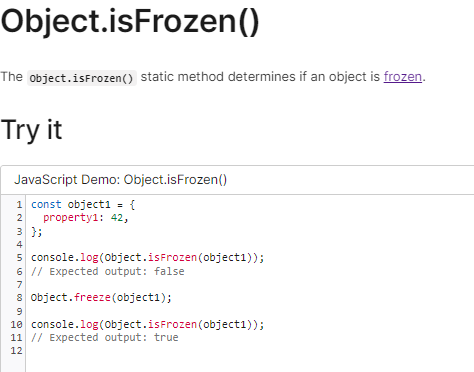
******

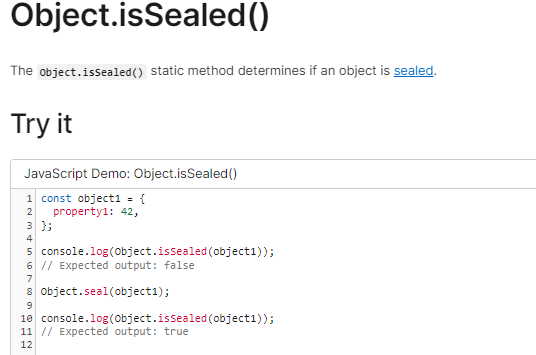
******

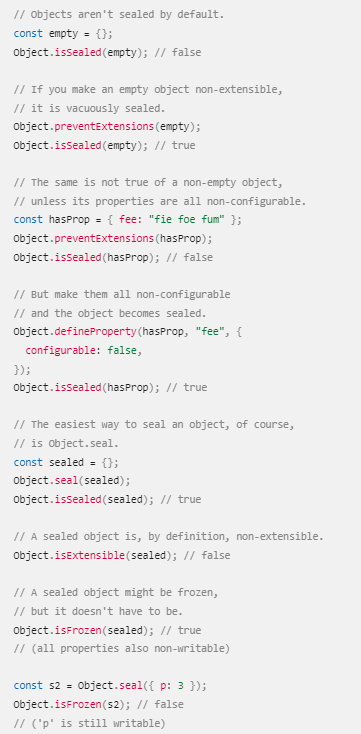
******

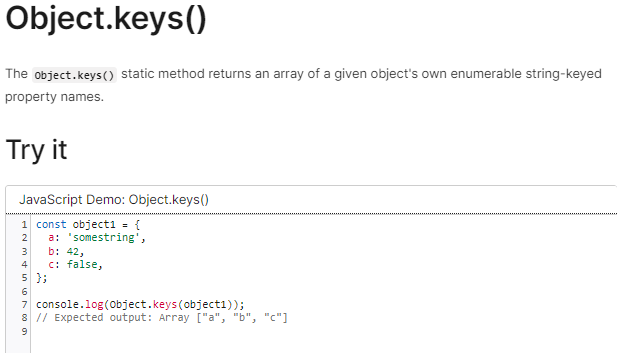
******

******

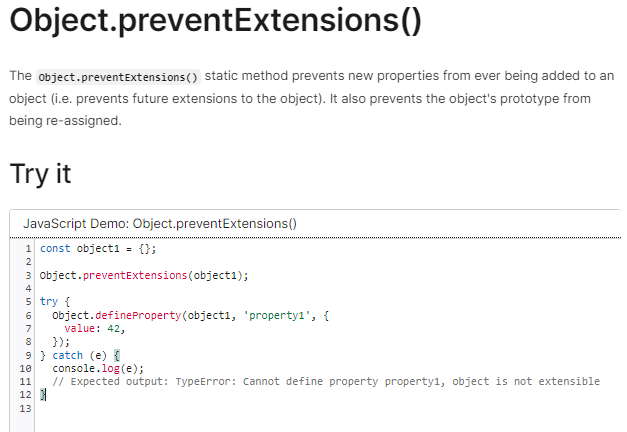
******

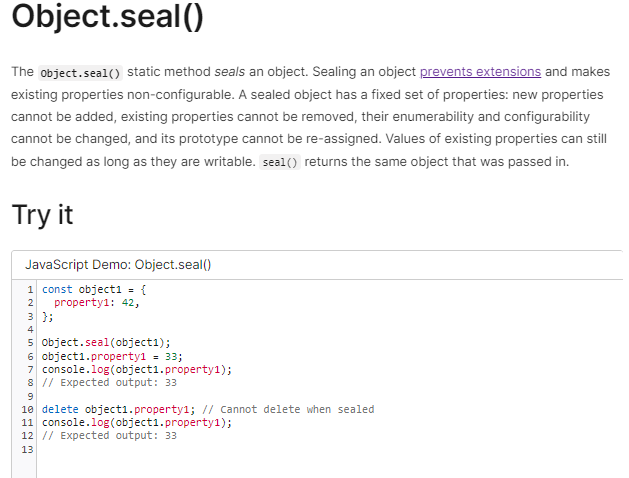
******

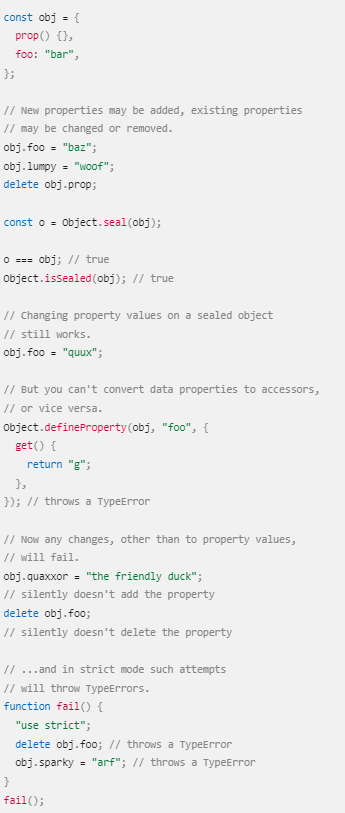
******

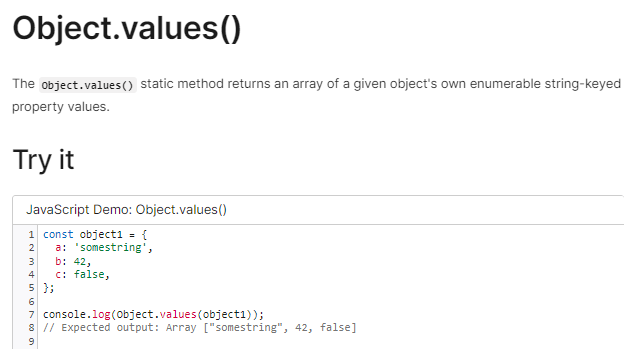
******

******

******

******

******

******

******