Object.seal()

Object.seal() is a method in that prevents new properties from being added to an object. When an object is seal, its properties cannot be **added** and **removed**. Any attempts to add or remove properties, will result in a **TypeError** being thrown.

- The Object.seal() method can be called on any object, including arrays and functions.

Here's an example of how you might use Object.seal():

Object.seal()

```
const obj = { a: 1, b: 2 };

Object.seal(obj);

// This will throw a TypeError
obj.c = 3;

// This will work
obj.a = 5;
```

You can check whether an object is seal or not by using **Object.isSeal()**

Object.seal()

```
const obj = { x: 1, y: 2 };

console.log(Object.isSeal(obj)); //false

Object.seal(obj);
```

```
console.log(Object.isSeal(obj)); //true
```

It should be noted that while the object itself is seal, its properties are not. If you want to also seal the properties inside of object, you need to **recursively seal** the properties.

Object.seal()

```
function deepSeal(obj) {
  // Retrieve the property names defined on obj
  var propNames =
Object.getOwnPropertyNames(obj);
  // Seal properties before sealing self
  propNames.forEach(function(name) {
    var prop = obj[name];

    // Seal prop if it is an object
    if (typeof prop == 'object')
      deepSeal(prop);
  });

  // Seal self (no-op if already sealed)
  Object.seal(obj);
}
```

. . .

Thanks for reading and keep learning !!

jscodelover