Map & Set

WeakMap & WeakSet

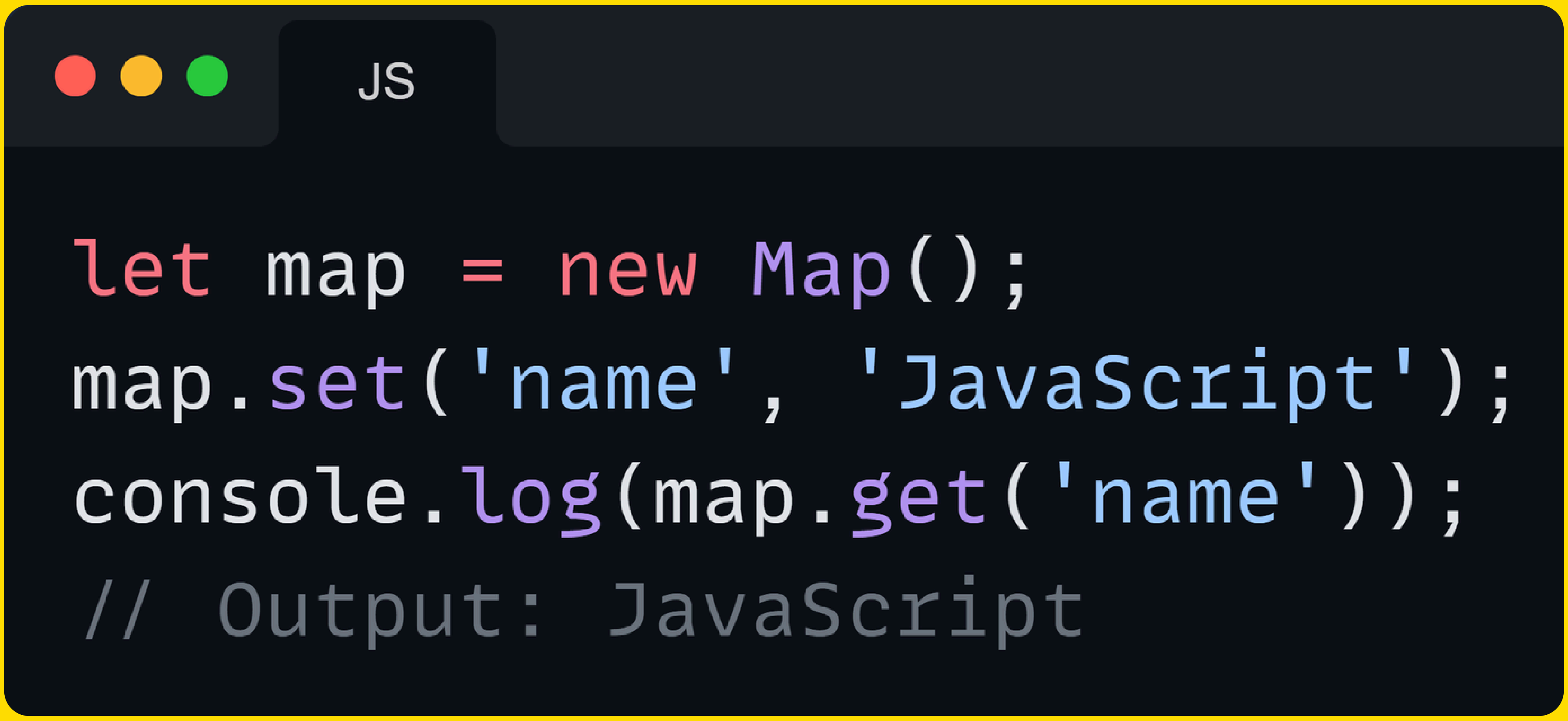
in JavaScript

[Inside]

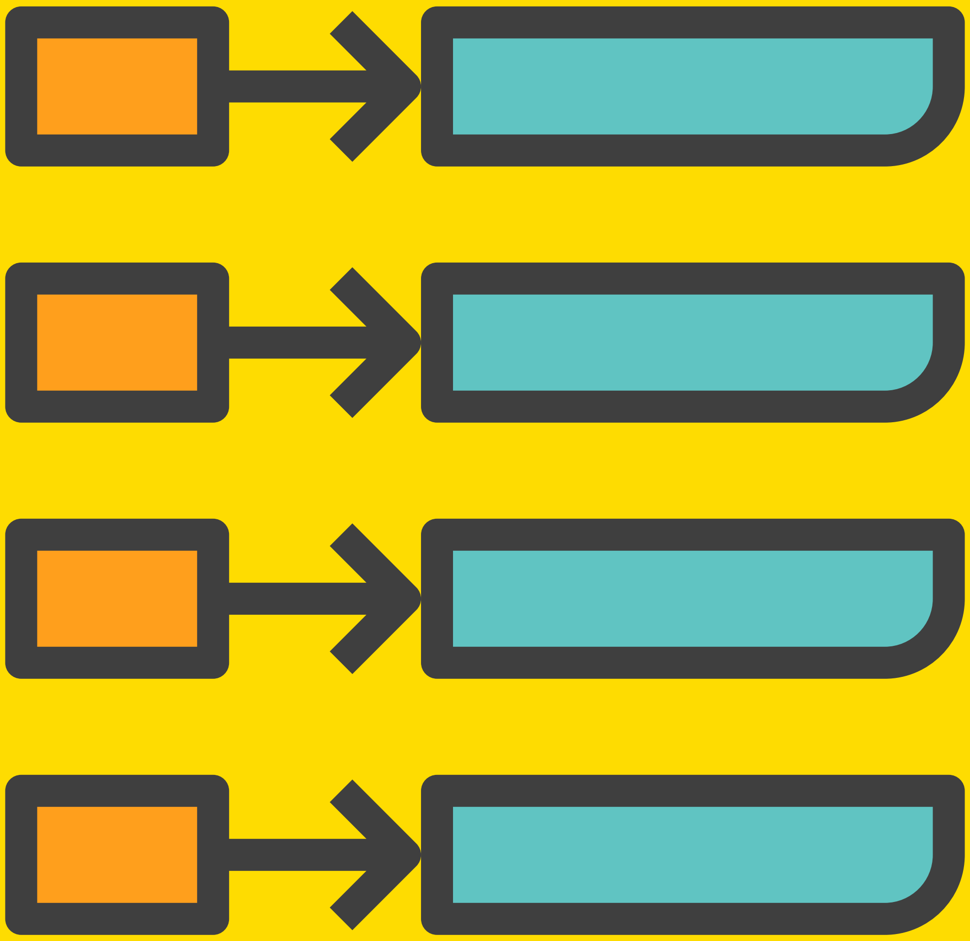
www.scribbler.live →

Map

- Map is a key-value pair structure.

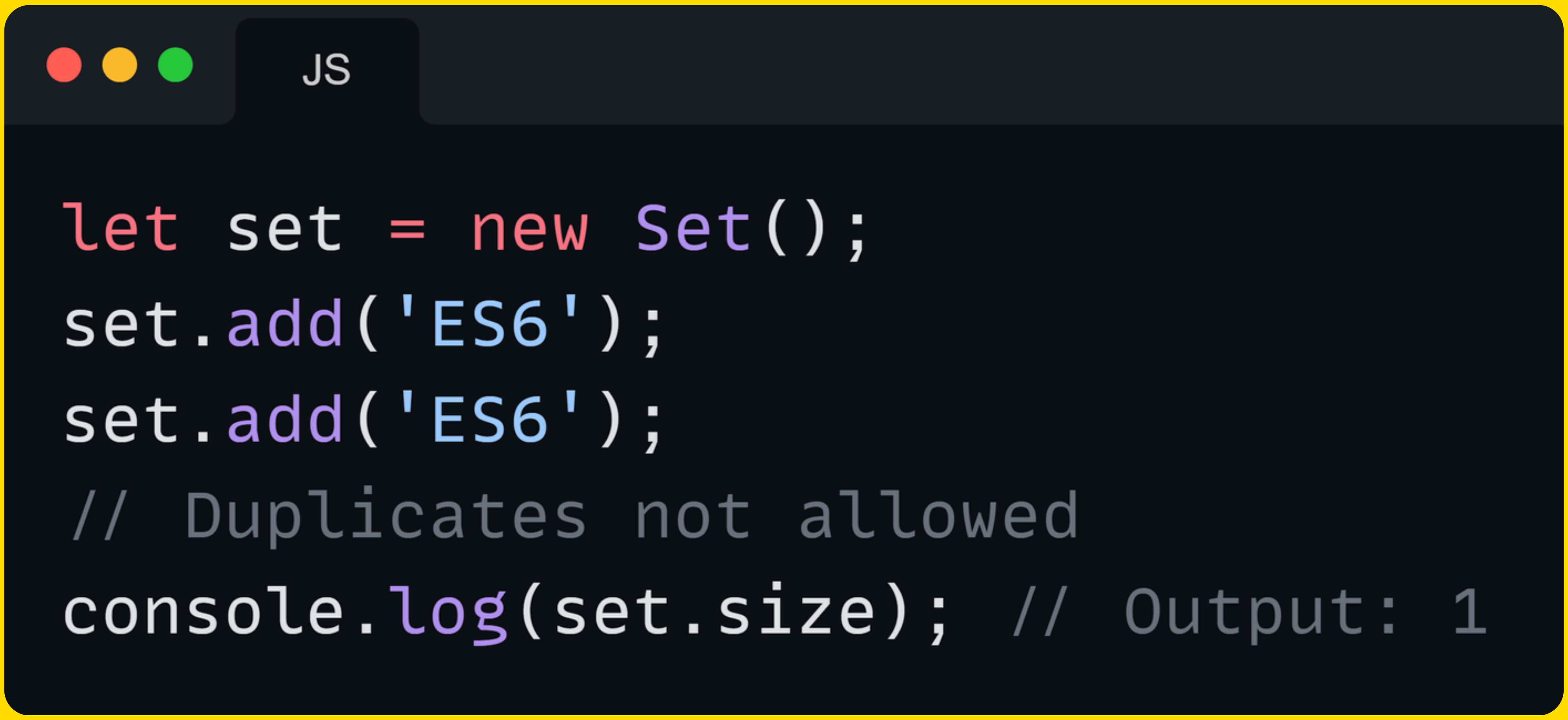


```
let map = new Map();
map.set('name', 'JavaScript');
console.log(map.get('name'));
// Output: JavaScript
```

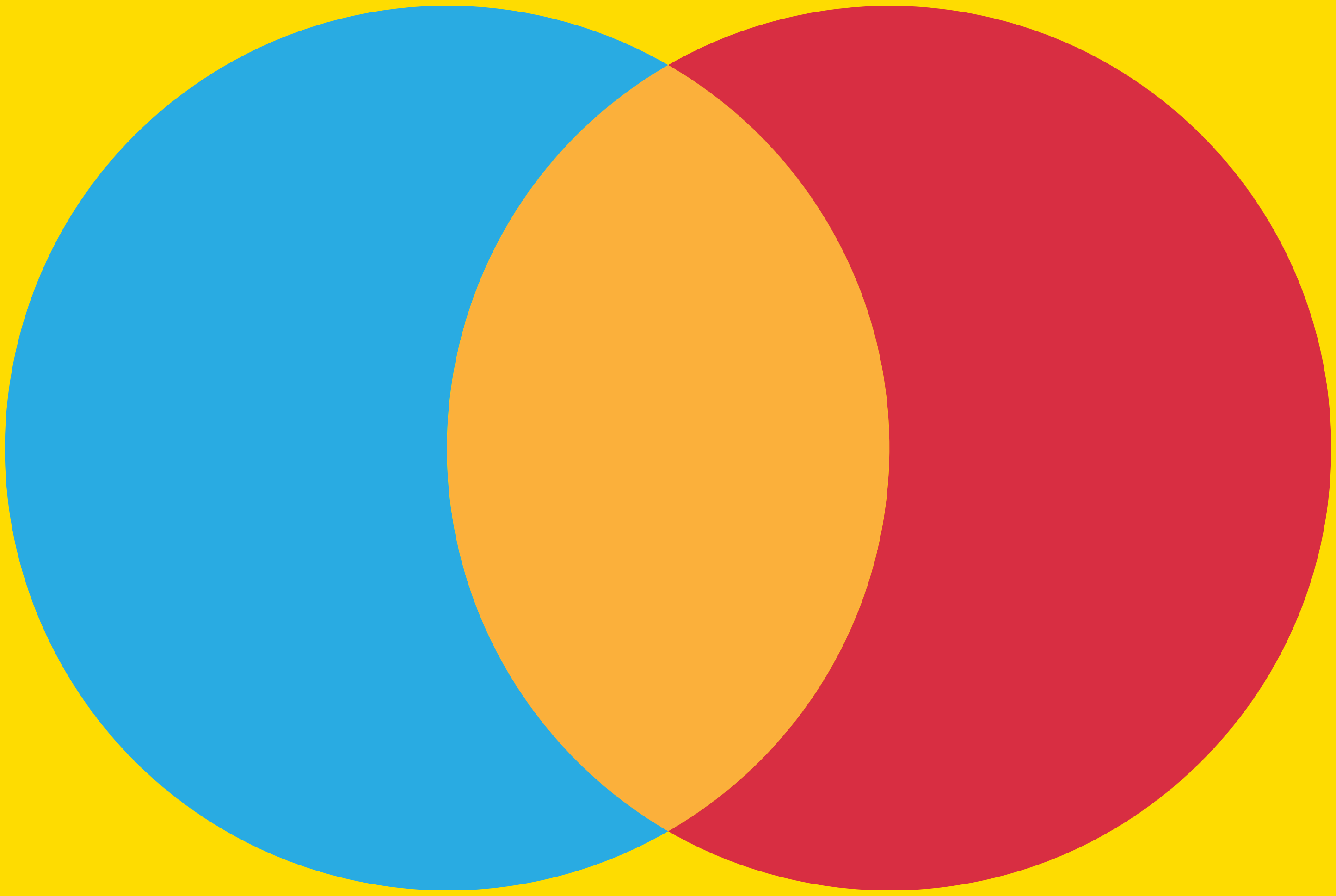


Set

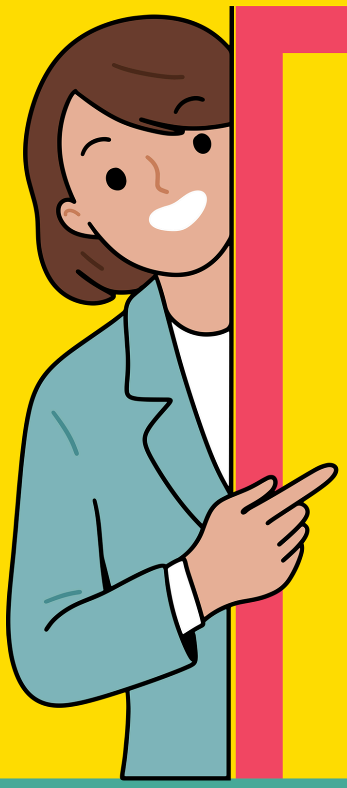
- Set stores unique values.



```js
let set = new Set();
set.add('ES6');
set.add('ES6');
// Duplicates not allowed
console.log(set.size); // Output: 1
```



Use Cases



Map

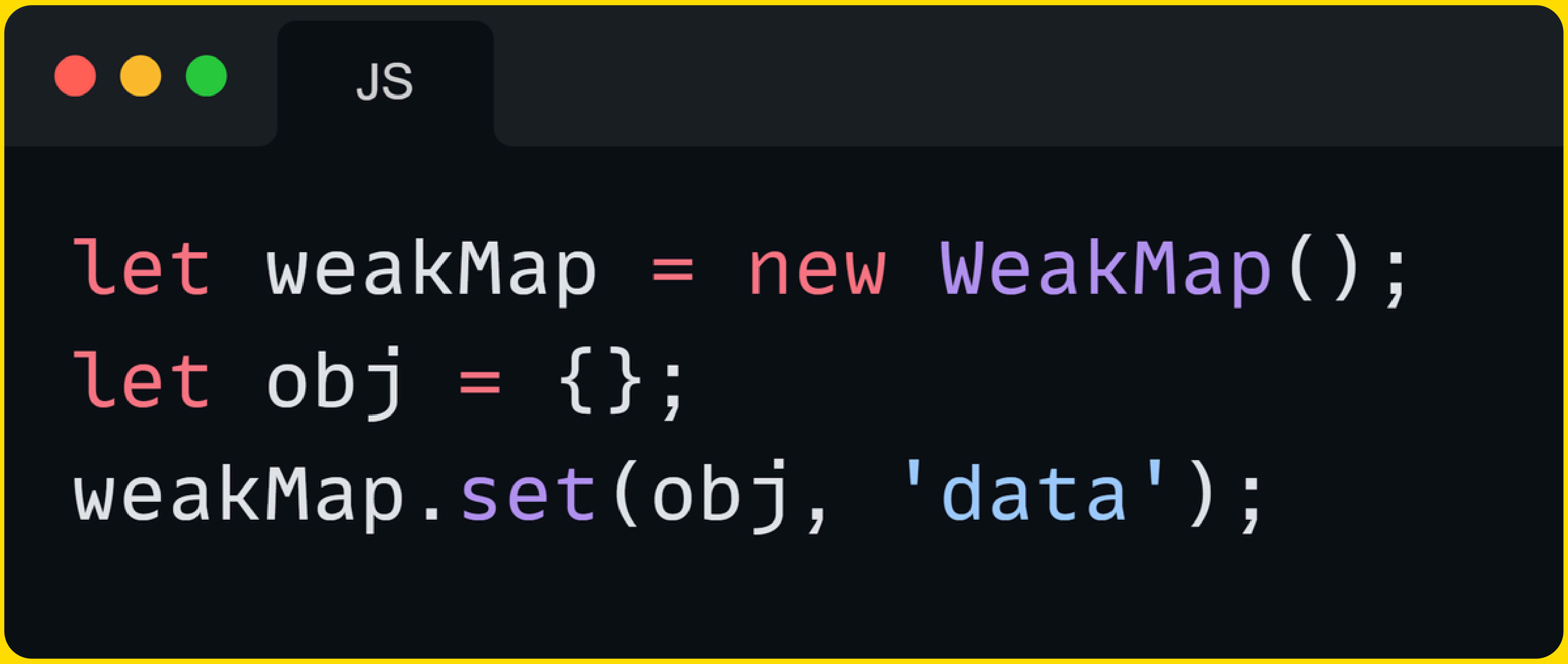
Ideal for situations where you need key-value pairs, especially when keys could be non-string.

Set

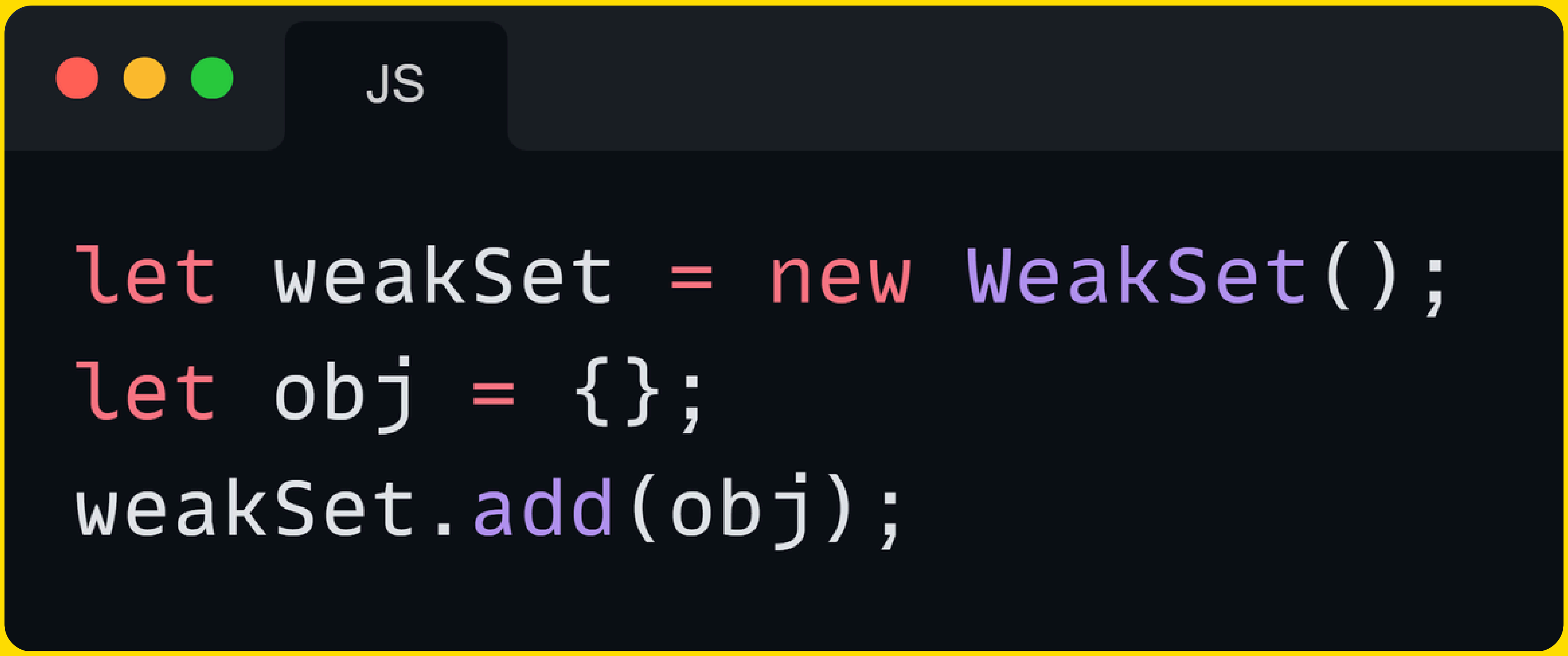
Great for managing unique collections of values (e.g., ensuring no duplicates).

WeakMap & WeakSet

- WeakMap and WeakSet are optimized for memory management by allowing objects to be garbage-collected when no other references exist.

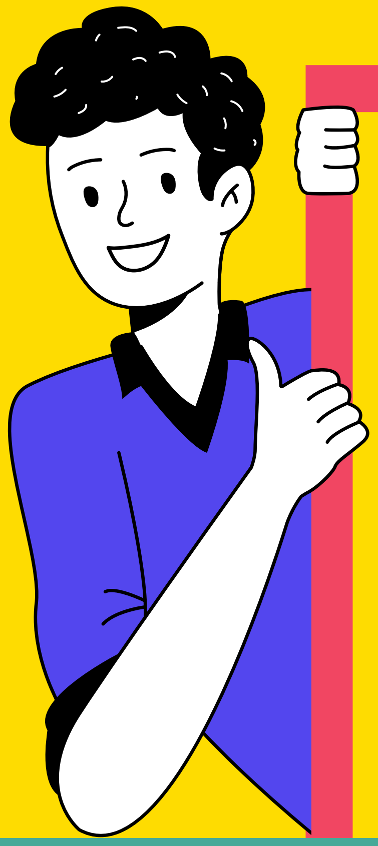


```
let weakMap = new WeakMap();
let obj = {};
weakMap.set(obj, 'data');
```



```
let weakSet = new WeakSet();
let obj = {};
weakSet.add(obj);
```

Use Cases



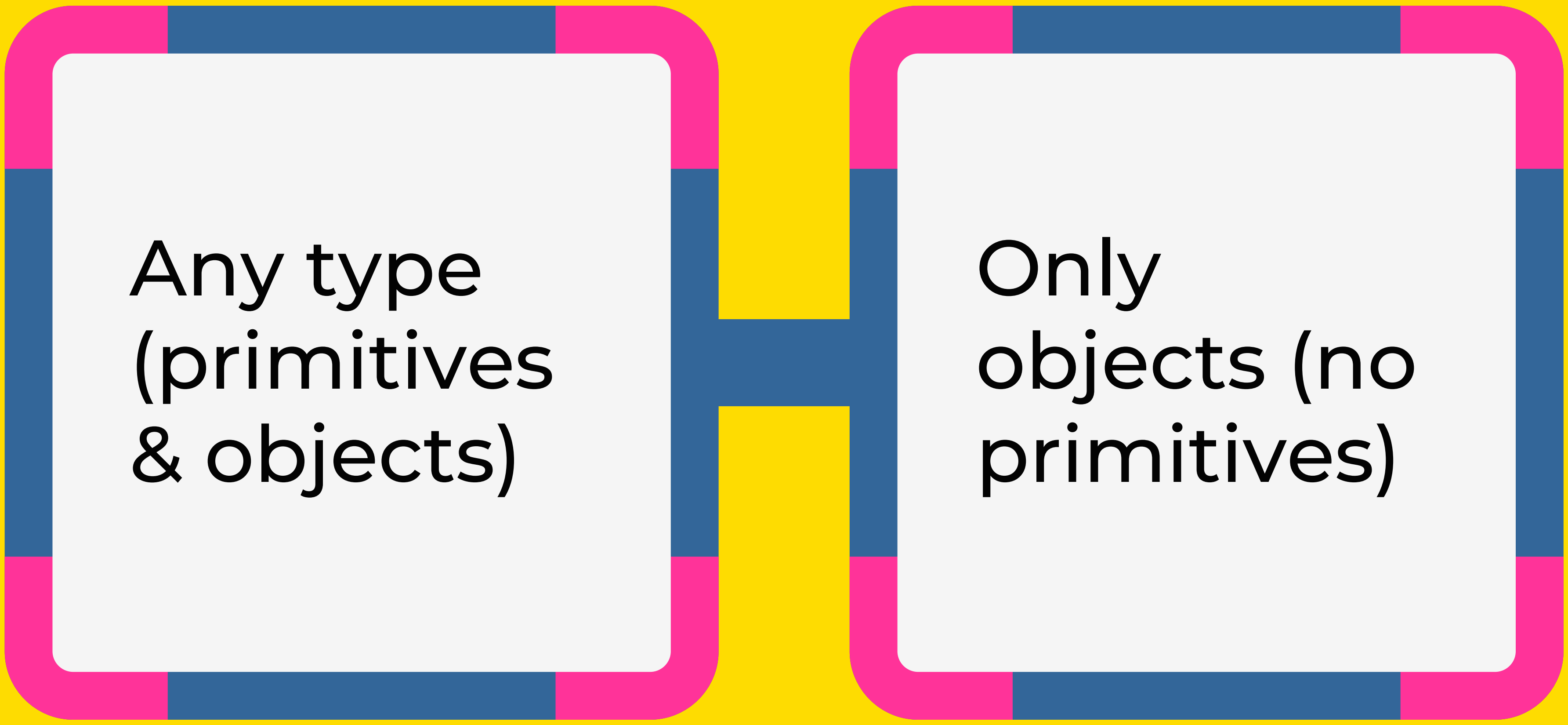
Ideal for managing metadata or tracking the state of objects without memory leaks.

WeakMap

WeakSet

Perfect for tracking objects (e.g., DOM elements) without retaining unnecessary memory.

Key/Value Type

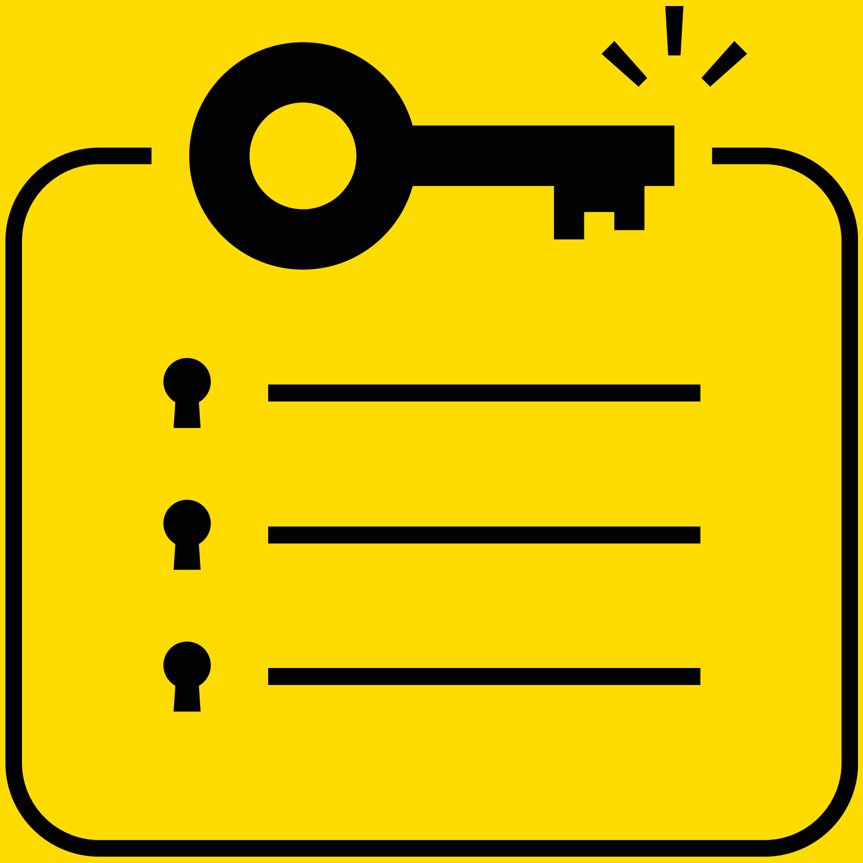


Any type (primitives & objects)

Only objects (no primitives)

Map / Set

WeakMap / WeakSet



Garbage Collection

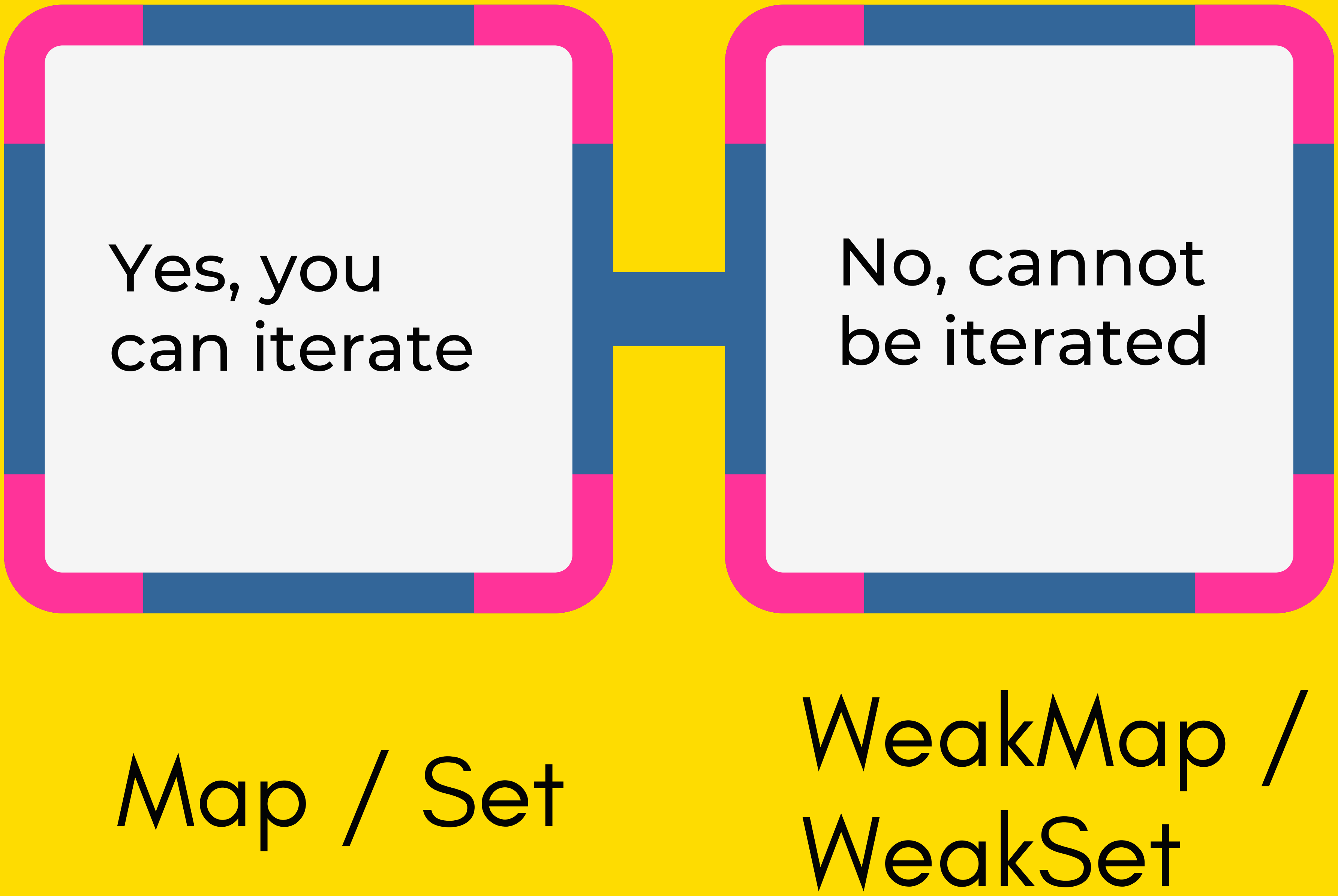
No, keeps references

Yes, keys/objects are weakly held

Map / Set

WeakMap / WeakSet

Iteration

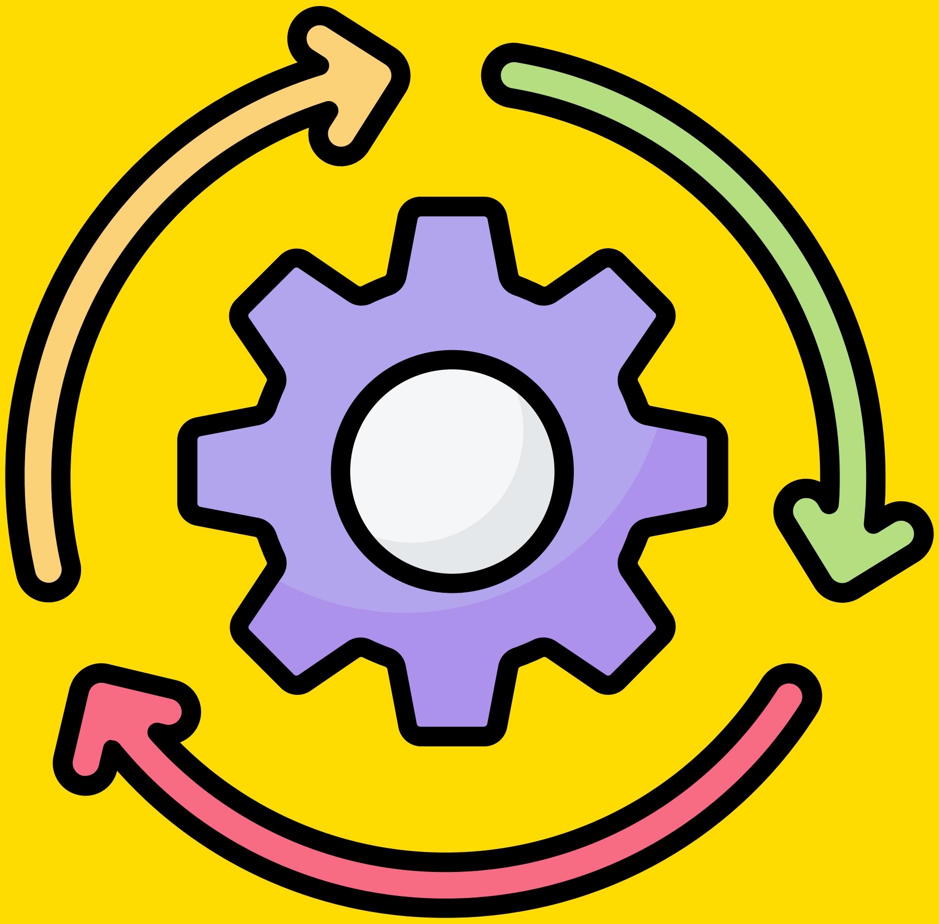


Yes, you can iterate

No, cannot be iterated

Map / Set

WeakMap / WeakSet



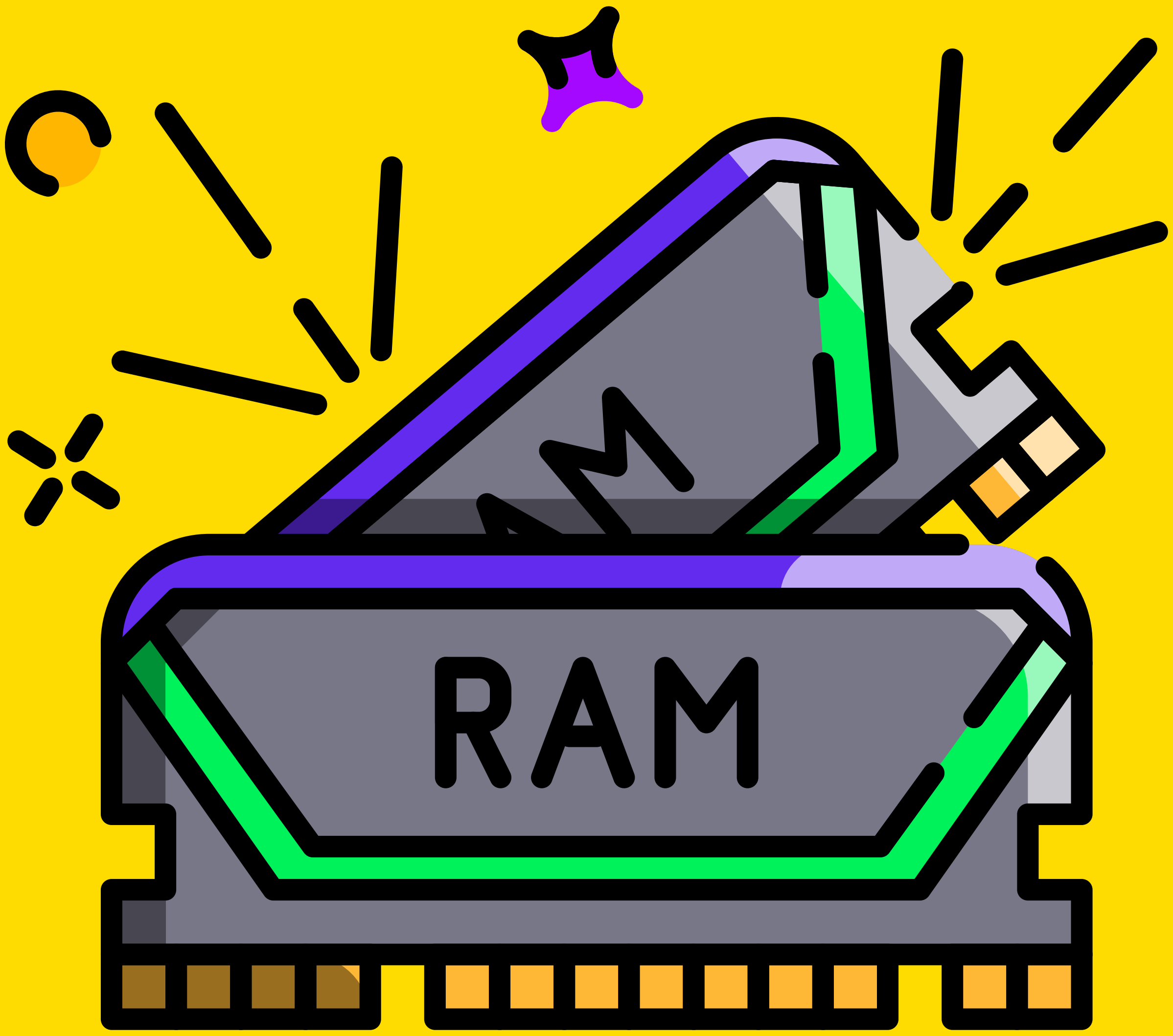
Memory Management

Retains all values

Map / Set

Frees up memory when objects are no longer referenced

WeakMap / WeakSet



Keep Exploring Javascript with us!

Share this with a friend who needs it and make sure to practice these in scribbler.

Scribbler.live

Free and Open Interface to experiment JavaScript