**BUILT-IN**

**SYMBOL:**

Q: What’s that for?

A: To avoid confusion

Ex:

**const** bowl = {

'apple': { color: 'red', weight: 136.078 },

'banana': { color: 'yellow', weight: 183.15 },

'orange': { color: 'orange', weight: 170.097 }

};

The bowl contains fruit which are objects that are properties of the bowl. But, we run into a problem when the second banana gets added.

**const** bowl = {

'apple': { color: 'red', weight: 136.078 },

'banana': { color: 'yellow', weight: 183.151 },

'orange': { color: 'orange', weight: 170.097 },

'banana': { color: 'yellow', weight: 176.845 }

};

console.log(bowl);

Object {apple: Object, banana: Object, orange: Object}

Instead of adding another banana to the bowl, our previous banana is overwritten by the new banana being added to the bowl. To fix this problem, we can use symbols.

**const** bowl = {

[Symbol('apple')]: { color: 'red', weight: 136.078 },

[Symbol('banana')]: { color: 'yellow', weight: 183.15 },

[Symbol('orange')]: { color: 'orange', weight: 170.097 },

[Symbol('banana')]: { color: 'yellow', weight: 176.845 }

};

console.log(bowl);

Object {Symbol(apple): Object, Symbol(banana): Object, Symbol(orange): Object, Symbol(banana): Object}

By changing the bowl’s properties to use symbols, each property is a unique Symbol and the first banana doesn’t get overwritten by the second banana.

**ITERATION & ITERATABLE PROTOCOL:**

\_ Iterator method is a zero arguments function that returns an iterator object. Used via [Symbol.iterator]

\_ Iterator object is an object that conforms to the iterator protocol

\_ Iterator protocol means how an object products a sequence of value

ex:

const digits = [0, 1, 2, 3, 4, 5, 6, 7, 8, 9];

const arrayIterator = digits[Symbol.iterator]();

console.log(arrayIterator.next());

console.log(arrayIterator.next());

console.log(arrayIterator.next());

Object {value: 0, done: false}

Object {value: 1, done: false}

Object {value: 2, done: false}

… (and so on)

// .next() is a method that returns the value of that next value in the sequence. Done means if the iterator has reached the end of its sequence of value