Maps and Sets

Objectives

- Describe the importance of Maps and Sets
- Create, Add, Retrieve, and Delete values from a Map Data Structure and a Set Data Structure

Set => A **list** of value that cannot contain duplicates
+ You wouldn't access individual items in a Set
+ You would use a Set to see if a value was present.

Map => A **collection** of keys that correspond to specific values
+ Used as caches to store data can be quickly retrieved at a later time
+ Access values by retrieving the corresponding key

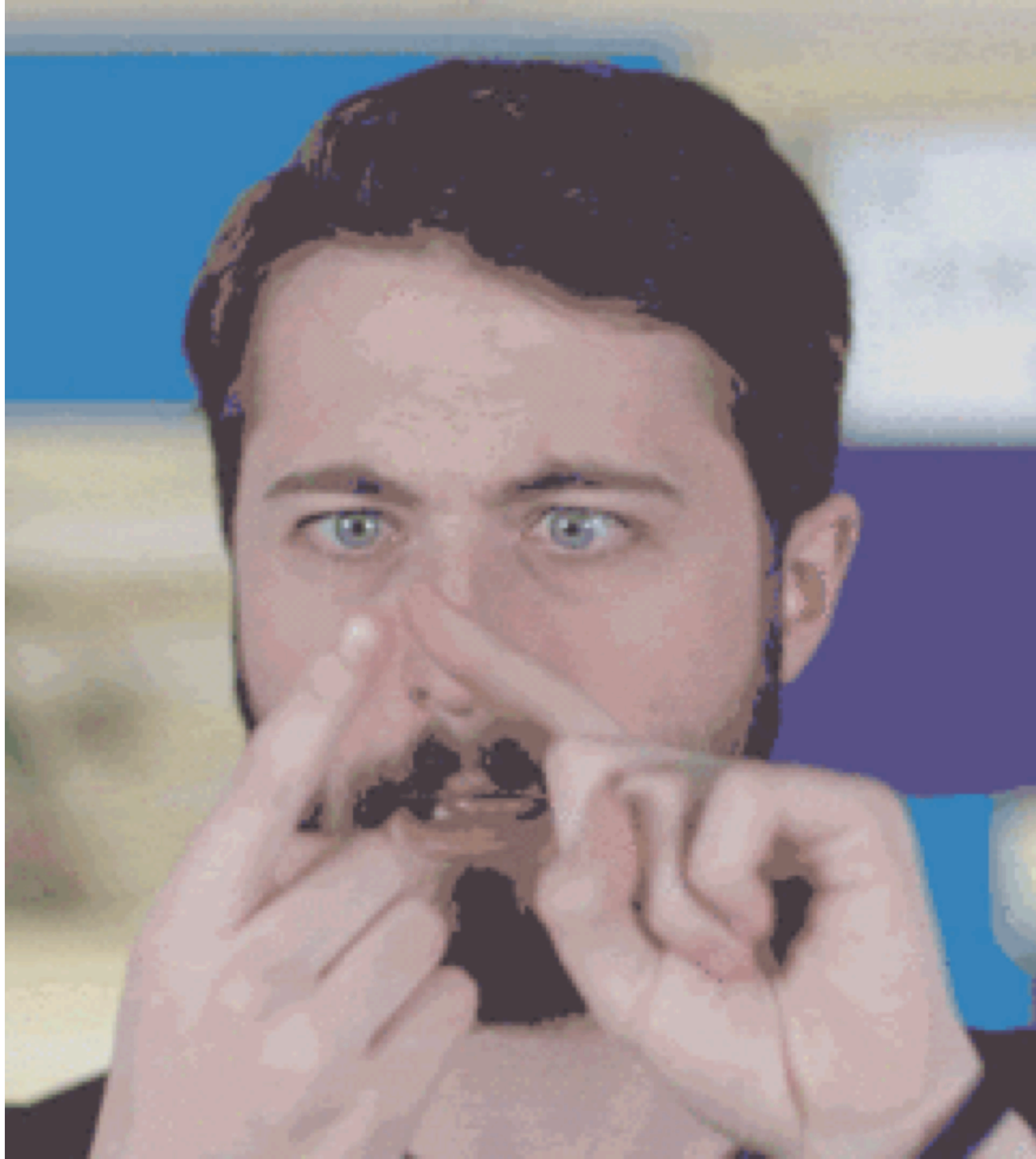
NOTE Both of these Data Structures are new to JavaScript with ES6

Sets

```
let set = new Set();
set.add(5);
set.add('green');

console.log(set);
// Set { 5, 'green' }

console.log(set.size);
// 2
```

Basic Operations:

- add
- has
- delete
- clear
- size



Creating a Set

```
let set = new Set([5,'green', {}]);
```

OR

```
let set = new Set().add(5).add('green').add({});
```

Values

Anything that can be compared to '===' can be used as a value in a Set

EXCEPTION: *NaN* which can be used as a value

- Adding elements a second time has no effect except for {} because they are never considered to be equal

Iterating

For-Of Loop

```
let set = new Set(['red', 'green', 'blue']);
for (let x of set) {
    console.log(x);
}
```

forEach()

```
let set = new Set(['red', 'green', 'blue']);
set.forEach(value=>{
    console.log(value);
});
```

Spreads and Sets

```
let set = new Set(['red', 'green', 'blue']);
let arr = [...set]; // ['red', 'green', 'blue']
```

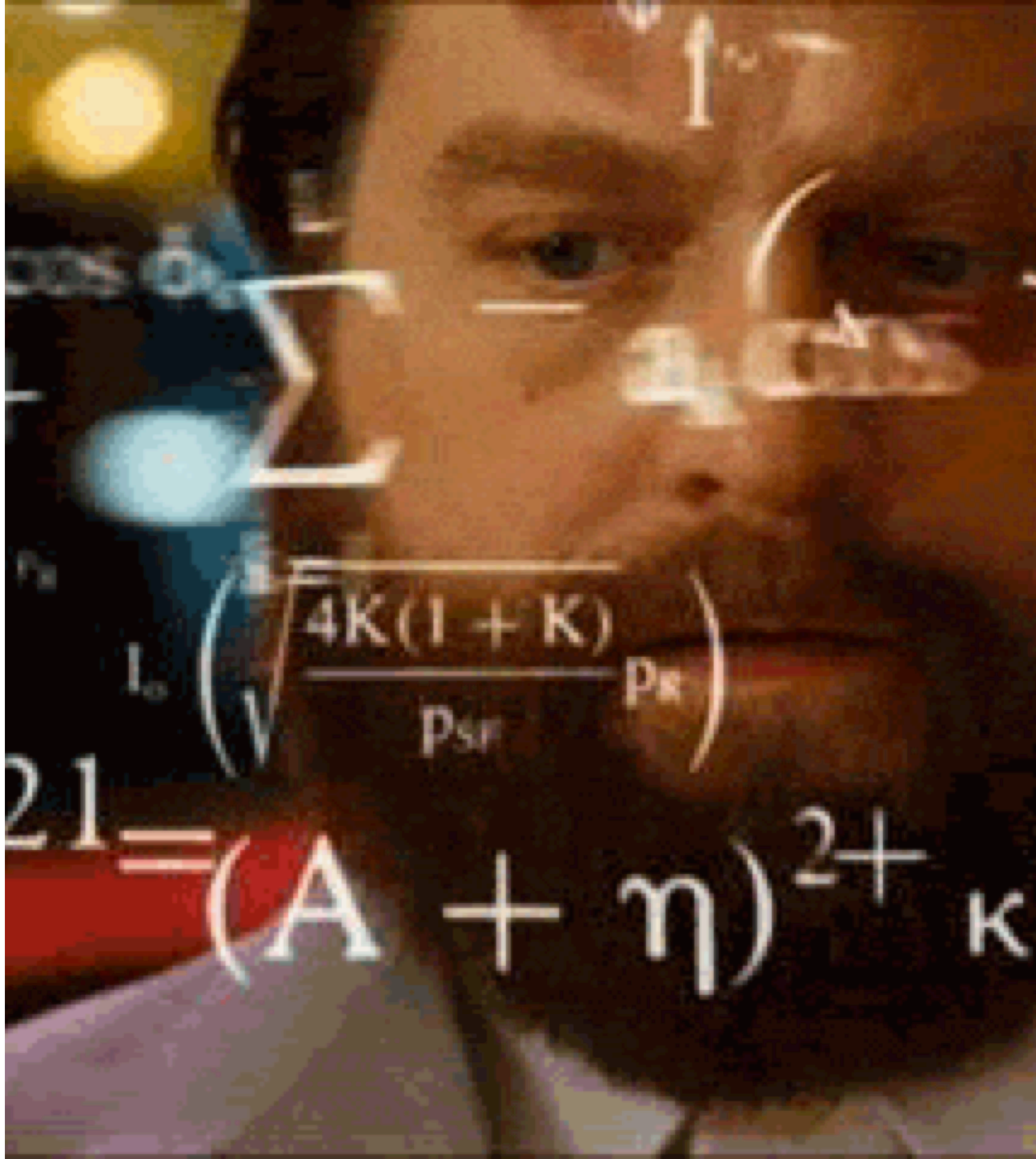
Can use the spread operator on a Set to be able to map and filter methods

Maps

```
let myMap = new Map();

let car1 = {
    make: 'Toyota',
    model: 'Camry'
}

myMap.set('car1', car1)

console.log(myMap);
// Map { 'car1' => {make: 'Toyota', model: 'Camry'} }
```

Basic Operations

- set
- get
- has
- delete
- size
- clear



Creating a Map

```
let map = new Map([
['1', 'a'],
['2', 'b'],
['3', 'c']]);
```

OR

```
let map = new Map();
map.set('1','a');
map.set('2', 'b');
map.set('3', 'c');
```

Keys

Anything can be a key in a Map

```
let map = new Map();
Let key1 = {};
map.set(key1, 1).set(-0, 'green').set(NaN, 123).set({}, 'Hello);
```

- +0, 0, -0 are considered the same value
- NaN can be used as a key just like any other value
- Different objects are always considered different

Iterating over Maps

- For-of Loops:

```
for(let key of map.keys()){
    console.log(key);
}

for(let value of map.values()){
    console.log(value);
}

for(let entry of map.entries()){
    console.log(entry[0], entry[1]);
}
```

Iterating over Maps

- forEach():

```
map.forEach((key, value) => {
  console.log(key, value);
});
```

Spreads and Maps

```
let map = new Map([
    [1, 'one'],
    [2, 'two'],
    [3, 'three'],
]);
let arr = [...map.keys()]; // [1, 2, 3]
```

Can use the spread operator to use map or filter methods

Objectives

- Describe the importance of Maps and Sets
- Create, Add, Retrieve, and Delete values from a Map Data Structure and a Set Data Structure

Resources

- ECMAScript 6: Maps and Sets
- YouTube: JavaScript ES6/ES2015 = [08]Set, Map, WeakSet, WeakMap