POWERING POTENTIAL

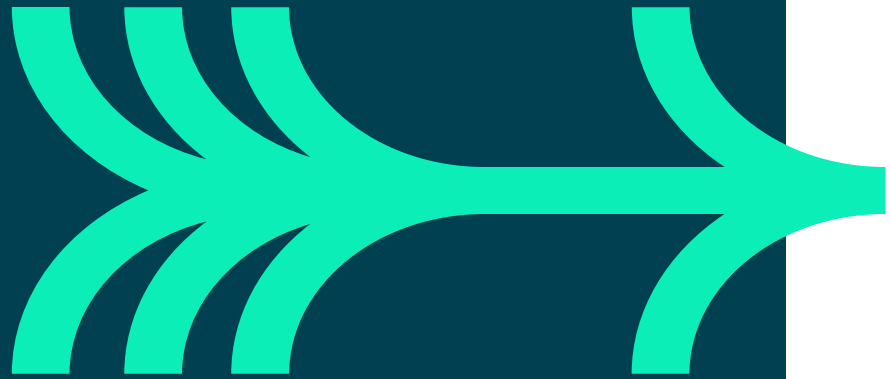
Collections

JavaScript Fundamentals

Introduction

Collections

- Maps & Sets
  - Creating
  - Accessing



Maps

- Key / Value pairs where both Key and Value can be any type

```
let myMap = new Map([[1,"bananas"],[2,"grapefruit"],[3,"apples"]]);
```

- With some helpful methods

```
console.log(myMap.size) //3

myMap.set(4, "strawberries");
console.log(myMap.size); //4

console.log(myMap.get(4)); //"strawberries"
console.log(myMap.has(2)); //true

myMap.delete(3);
console.log(myMap.size); //3

myMap.clear();
console.log(myMap.size); //0
```

Maps: Iterating

- We can iterate over a map using for...of

```
// log all key/value pairs in the map
for (let [key, value] of myMap) {
    console.log(`key: ${key} value: ${value}`);
}
// log all keys in the map
for (let key of myMap.keys()) {
    console.log(`key: ${key}`);
}
// log all values in the map
for (let value of myMap.values()) {
    console.log(`value: ${value}`);
}
// log all entries (key/value pairs) in the map
for (let [key, value] of myMap.entries()) {
    console.log(`key: ${key} value: ${value}`);
}
```

Sets

- Sets allow you to store unique values of any type

```
let mySet = new Set();
```

- With some helpful methods

```
mySet.add("apples")
mySet.add("bananas")
console.log(mySet.size) //2

mySet.add("apples")
console.log(mySet.size) //2 (the 2nd apples is not unique)

console.log(mySet.has("apples")); //true

mySet.delete("apples");
console.log(mySet.size); //1

mySet.clear();
console.log(myMap.size); //0
```

Sets: Iterating

- We can iterate over a set using for...of

```
// log all key/value pairs in the set
for (let item of mySet) {
    console.dir(item);
}
// log all values in the set
for (let value of mySet.values()) {
    console.log(`value: ${value}`);
}
// same as above for values()
for (let key of mySet.keys()) {
    console.log(`key: ${key}`);
}
// log all entries (key/value pairs) in the set where key and value are the same
for (let [key, value] of mySet.entries()) {
    console.log(`key: ${key} value: ${value}`);
}
```

WeakSets and WeakMaps

- Behave exactly like Map and Set but:
  - Do not support iteration methods
  - Values in a WeakSet and keys in a WeakMap must be objects
- This allows the garbage collector to collect dead objects out of weak collections!

```
// keep track of what DOM elements are moving
let element = document.querySelector(".animateMe");

if (movingSet.has(element)) {
    smoothAnimations(element);
}
movingSet.add(element);
```

QuickLab 7 - Maps

Creating and Managing Maps

REVIEW

Arrays and Objects are essential collections that allow us to gather data under one roof that can then be acted upon in a coherent and concise manner