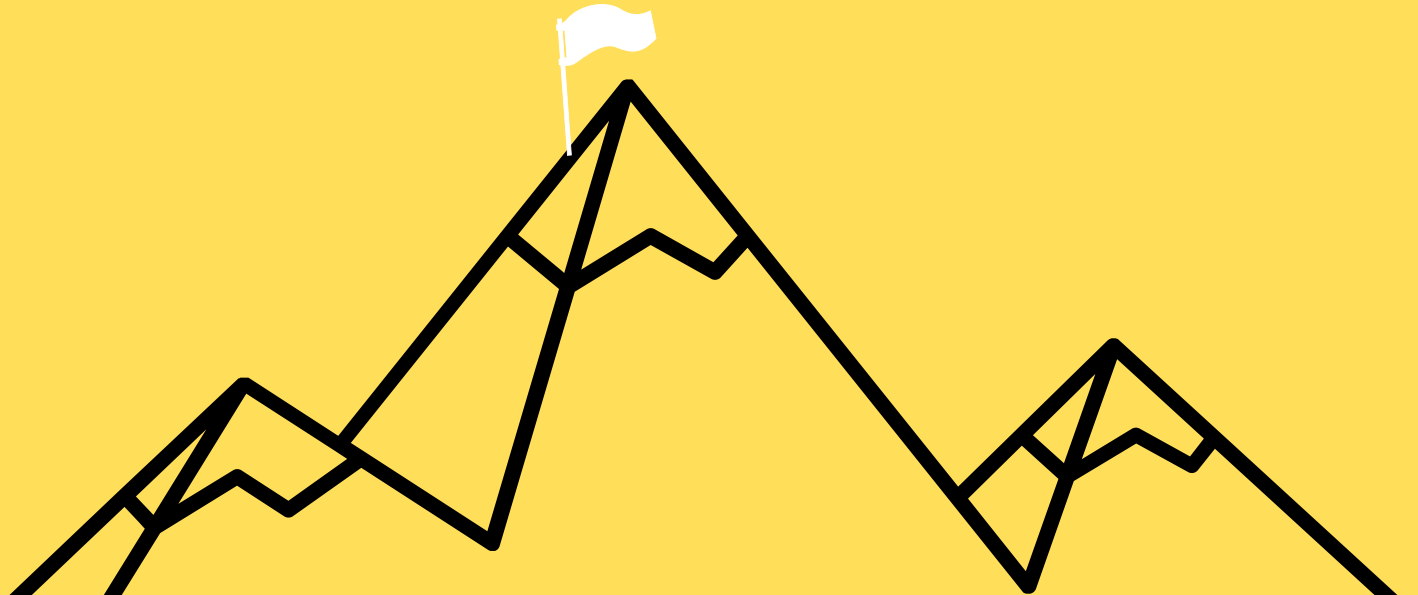


# GOALS

- Use the new arrow function syntax
- Understand and use these methods:
  - `forEach`
  - `map`
  - `filter`
  - `find`
  - `reduce`
  - `some`
  - `every`



# FOREACH

```
const nums = [9, 8, 7, 6, 5, 4, 3, 2, 1];

nums.forEach(function (n) {
  console.log(n * n)
  //prints: 81, 64, 49, 36, 25, 16, 9, 4, 1
});

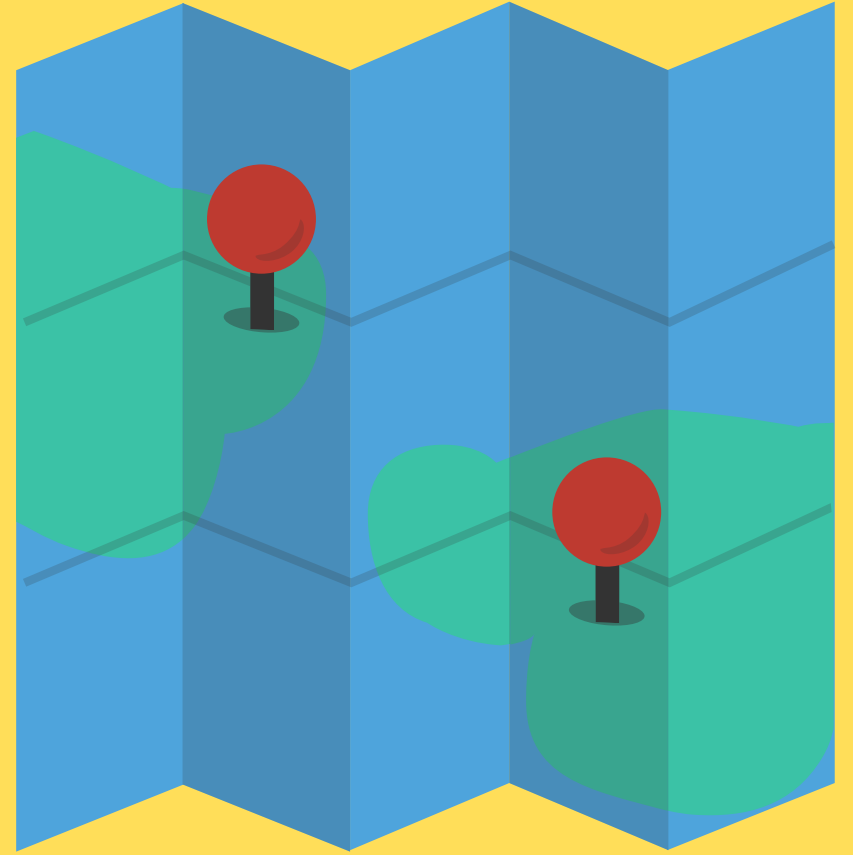
nums.forEach(function (el) {
  if (el % 2 === 0) {
    console.log(el)
    //prints: 8, 6, 4, 2
  }
})
```

Accepts a callback function.

Calls the function once per element in the array.

# MAP

Creates a new array with the results of calling a callback on every element in the array



# MAP



```
const texts = ['rofl', 'lol', 'omg', 'ttyl'];  
const caps = texts.map(function (t) {  
  return t.toUpperCase();  
})  
texts; //["rofl", "lol", "omg", "ttyl"]  
caps;  //["ROFL", "LOL", "OMG", "TTYL"]
```

# FIND


returns the value of the first element in the array that satisfies the provided testing function.

```
let movies = [
  "The Fantastic Mr. Fox",
  "Mr. and Mrs. Smith",
  "Mrs. Doubtfire",
  "Mr. Deeds"
]
let movie = movies.find(movie => {
  return movie.includes('Mrs.')
}) // "Mr. and Mrs. Smith"

let movie2 = movies.find(m => m.indexOf('Mrs') === 0);
// "Mrs. Doubtfire"
```

# FILTER

Creates a new array with all elements that pass the test implemented by the provided function.



```
const nums = [9, 8, 7, 6, 5, 4, 3, 2, 1];
const odds = nums.filter(n => {
  return n % 2 === 1; //our callback returns true or false
  //if it returns true, n is added to the filtered array
})
//[9, 7, 5, 3, 1]

const smallNums = nums.filter(n => n < 5);
//[4, 3, 2, 1]
```

# EVERY

tests whether **all** elements in the array pass the provided function. It returns a Boolean value.

```
const words = ["dog", 'dig', 'log', 'bag', 'wag'];

words.every(word => {
  return word.length === 3;
}) //true

words.every(word => word[0] === 'd'); //false

words.every(w => {
  let last_letter = w[w.length - 1];
  return last_letter === 'g'
}) //true
```

# SOME

Similar to every, but returns true if ANY of the array elements pass the test function

```
const words = ['dog', 'jello', 'log', 'cupcake', 'bag', 'wag'];

//Are there any words longer than 4 characters?
words.some(word => {
  return word.length > 4;
}) //true

//Do any words start with 'Z'?
words.some(word => word[0] === 'Z'); //false

//Do any words contain 'cake'?
words.some(w => w.includes('cake')) //true
```



# REDUCE

Executes a reducer function on each element of the array, resulting in a single value.



# SUMMING AN ARRAY



```
[3, 5, 7, 9, 11].reduce((accumulator, currentValue) => {  
  return accumulator + currentValue;  
});
```

Callback	accumulator	currentValue	return value
first call	3	5	8
second call	8	7	15
third call	15	9	24
fourth call	24	11	35

# FINDING MAX VAL



```
let grades = [89, 96, 58, 77, 62, 93, 81, 99, 73];
```

```
const topScore = grades.reduce((max, currVal) => {  
  if (currVal > max) return currVal;  
  return max;  
})
```

```
topScore; //99
```

```
//A shorter option w/ Math.max & implicit return
```

```
const topScore = grades.reduce((max, currVal) => (  
  Math.max(max, currVal)  
))
```

# INITIAL VALUE



```
[4, 5, 6, 7, 8].reduce((accumulator, currentValue) => {  
  return accumulator + currentValue;  
});  
//RETURNS: 30
```

```
[4, 5, 6, 7, 8].reduce((accumulator, currentValue) => {  
  return accumulator + currentValue;  
}, 100);  
//RETURNS: 130
```

# TALLYING



```
const votes =  
  ['y', 'y', 'n', 'y', 'n', 'y', 'n', 'y', 'n', 'n', 'n', 'y', 'y'];  
const tally = votes.reduce((tally, vote) => {  
  tally[vote] = (tally[vote] || 0) + 1;  
  return tally;  
}, {}); //INITIAL VALUE: {}  
  
tally; //{y: 7, n: 6}
```