Filter

Learn how `Array.filter` works by learning how to write it yourself. It's almost as powerful as `map` and used ubiquitously throughout code.

Array.filter

The idea here is similar to Array.map, except instead of transforming individual values, we want to filter existing values. Without any functions (besides Array.push), say we want to filter out values in an array that are less than 5:

for-loop

```
const arr = [2, 4, 6, 8, 10];
const filteredArr = [];

for(let i = 0; i < arr.length; i++) {
    if(arr[i] >= 5) {
        filteredArr.push(arr[i]);
    }
}

console.log(filteredArr); // -> [6, 8, 10]
```

Let’s abstract this to a function so we can remove values below 5 in any array.

```
const arr = [2, 4, 6, 8, 10];

function filterLessThanFive(arr) {
    const filteredArr = [];

    for(let i = 0; i < arr.length; i++) {
        if(arr[i] >= 5){
            filteredArr.push(arr[i]);
        }
    }

    return filteredArr;
}
```

```
const filteredArr = filterLessThanFive(arr);
console.log(filteredArr); // -> [6, 8, 10]
```

Let’s make it so we can filter out all values below any arbitrary value.

```
function isGreaterThan5(item) {
    return item > 5;
}

function filterLessThanFive(arr) {
    const filteredArr = [];

    for(let i = 0; i < arr.length; i++) {
        if(isGreaterThan5(arr[i])) {
            filteredArr.push(arr[i]);
        }
    }

    return filteredArr;
}

const arr = [2, 4, 6, 8, 10];
const filteredArr = filterLessThanFive(arr);

console.log(filteredArr); // -> [6, 8, 10]
```

→ Abstracting out the filtering functionality

```
function filterBelow(arr, greaterThan) {
    const filteredArr = [];

    for(let i = 0; i < arr.length; i++) {
        if(greaterThan(arr[i])) {
            filteredArr.push(arr[i]);
        }
    }

    return filteredArr;
}

const originalArr = [2, 4, 6, 8, 10];
```

→ Filtering out anything below 5, using `filterBelow`

```
function isGreaterThan5(item) {
    return item > 5;
    return item > 5;
}


const newArr = filterBelow(originalArr, isGreaterThan5);
console.log(newArr); // -> [6, 8, 10];
```

→ Filtering out anything below 7, using `filterBelow`

```
function isGreaterThan7(item) {
    return item > 7;
}

const newArr2 = filterBelow(originalArr, isGreaterThan7);
console.log(newArr2); // -> [8, 10];
```

filter

So we have a function `filterBelow` that will filter out anything below a certain value, based on the `greaterThan` function we give it (this is identical to the `filterBelow` function above):

```
function filterBelow(arr, greaterThan) {
    const filteredArr = [];

    for(let i = 0; i < arr.length; i++) {
        if(greaterThan(arr[i])) {
            filteredArr.push(arr[i]);
        }
    }

    return filteredArr;
}
```

Let’s rename it.

```
function filter(arr, testFunction) {
    const filteredArr = [];

    for(let i = 0; i < arr.length; i++) {
        if(testFunction(arr[i])) {
            filteredArr.push(arr[i]);
        }
    }
    return filteredArr;
}
```

And we've written `filter`. It's basically the same as `Array.filter`, again except for usage:

```
// Although not visible, we have access to the
// 'filter' function from the above block.

const arr = ['abc', 'def', 'ghijkl', 'mnopuv'];

function longerThanThree(str) {
    return str.length > 3;
}

const newArr1 = filter(arr, longerThanThree);
const newArr2 = arr.filter(longerThanThree);

console.log(newArr1); // -> ['ghijkl', 'mnopuv']
console.log(newArr2); // -> ['ghijkl', 'mnopuv']
```

Again, `Array.filter` passes in the index and the original array to our function.

```
function log(item, index, arr) {
    console.log(item, index, arr);
}

const arr = ['abc', 'def', 'ghi'];
arr.filter(log); // -> 'abc', 0, ['abc', 'def', 'ghi']
                 // -> 'def', 1, ['abc', 'def', 'ghi']
                 // -> 'ghi', 2, ['abc', 'def', 'ghi']
```

So we should make our function do the same.

```
function filter(arr, testFunction) {
    const filteredArr = [];

    for(let i = 0; i < arr.length; i++) {
        if(testFunction(arr[i], i, arr)) {
            filteredArr.push(arr[i]);
        }
    }

    return filteredArr;
}
```

Wow. That’s it. You’ve learned how to use *and write* `Array.map` and `Array.filter`.