

ARRAY FUNCTIONS

PROF. DAVID ROSSITER

AFTER THIS PRESENTATION

- You'll learn some more advanced array functions

WE'LL LOOK AT

`forEach()`

`map()`

FOREACH()

- You can go through every element using loop (for / while)

```
var pets = ["Dog", "Cat", "Hamster"];  
for(var i = 0; i < pets.length; i++) {  
    alert(pets[i]);  
}
```

- You can also use *array.forEach(function)*:

```
var pets = ["Dog", "Cat", "Rabbit"];  
pets.forEach(alert);  
// This shows 3 separate alerts
```

MORE ON FOREACH()

- You can think of `forEach()` in this way:

```
function forEach(theArray, fn) {  
    for(var i = 0; i < theArray.length; i++) {  
        fn(theArray[i], i, theArray);  
    }  
}
```

- So, your function should look like this, if you need all of the 3 things:

```
function yourFunction(element, index, array) {}
```

```
<!doctype html>
<html>
<body>
  <script>
    var numbers = [1, 2, 3, 4, 5];
    numbers.forEach( function(elem, idx, arr) {
      arr[idx] = elem * elem;
    });
    alert(numbers); // This shows [1,4,9,16,25];
  </script>
</body>
</html>
```

MAP()

- `map(function)` stores the result of each execution of *function* into an array it returns.

You can think of `map()` in this way:

```
function map(theArray, fn) {  
  var results = [];  
  for(var i = 0; i < theArray.length; i++) {  
    results.push(fn(theArray[i], i, theArray));  
  }  
  return results;  
}
```

```
<!doctype html>
<html>
<body>
  <script>
    var square = function(el) { return el * el; }
    var numbers = [1, 2, 3, 4, 5];
    var results = numbers.map(square);
    alert(results); // This shows [1,4,9,16,25];
  </script>
</body>
</html>
```