



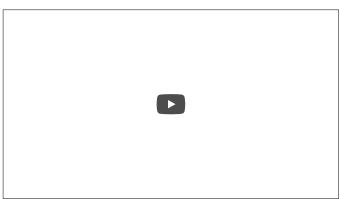




 \equiv

- ✓ 1. Intro to Arrays
- ✓ 2. Donuts to Code
- ✓ 3. Creating an Array
- ✓ 4. Accessing Array Elements
- 5. Array Index
- ✓ 6. Quiz: UdaciFamily (6-1)
- √ 7. Quiz: Building the Crew (6-2)
- ✓ 8. Quiz: The Price is Right (6-3)
- 9. Array Properties and Methods
- ✓ 10. Length
- ✓ 11. Push
- ✓ 12. Pop
- ✓ 13. Splice
- 14. Quiz: Colors of the Rainbow (6-4)
- ✓ 15. Quiz: Quidditch Cup (6-5)
- √ 16. Quiz: Joining the Crew (6-6)
- √ 17. Quiz: Quiz: Checking out the Docs ...
- ✓ 18. Array Loops
- ✓ 19. The forEach Loop
- ✓ 20. Quiz: Another Type of Loop (6-8)
- ✓ 21. Map
- 22. Quiz: I Got Bills (6-9)
- 23. Arrays in Arrays
- 24. 2D Donut Arrays
- 25. Quiz: Nested Numbers (6-10)
- 26. Lesson 6 Summary





Using <code>forEach()</code> will not be useful if you want to permanently modify the original array. <code>forEach()</code> always returns <code>undefined</code>. However, creating a new array from an existing array is simple with the powerful <code>map()</code> method.

With the map() method, you can take an array, perform some operation on each element of the array, and return a new array.

```
var donuts = ["jelly donut", "chocolate donut", "glazed donut"];
var improvedDonuts = donuts.map(function(donut) {
    donut += " hole";
    donut += donut.toUpperCase();
    return donut;
));
```

donuts array: ["jelly donut", "chocolate donut", "glazed donut"]
improvedDonuts array: ["JELLY DONUT HOLE", "CHOCOLATE DONUT HOLE", "GLAZED DONUT
HOLE"]

Oh man, did you just see that? The map() method accepts one argument, a function that will be used to manipulate each element in the array. In the above example, we used a function expression to pass that function into map(). This function is taking in one argument, donut which corresponds to each element in the donuts array. You no longer need to iterate over the indices anymore. map() does all that work for you.