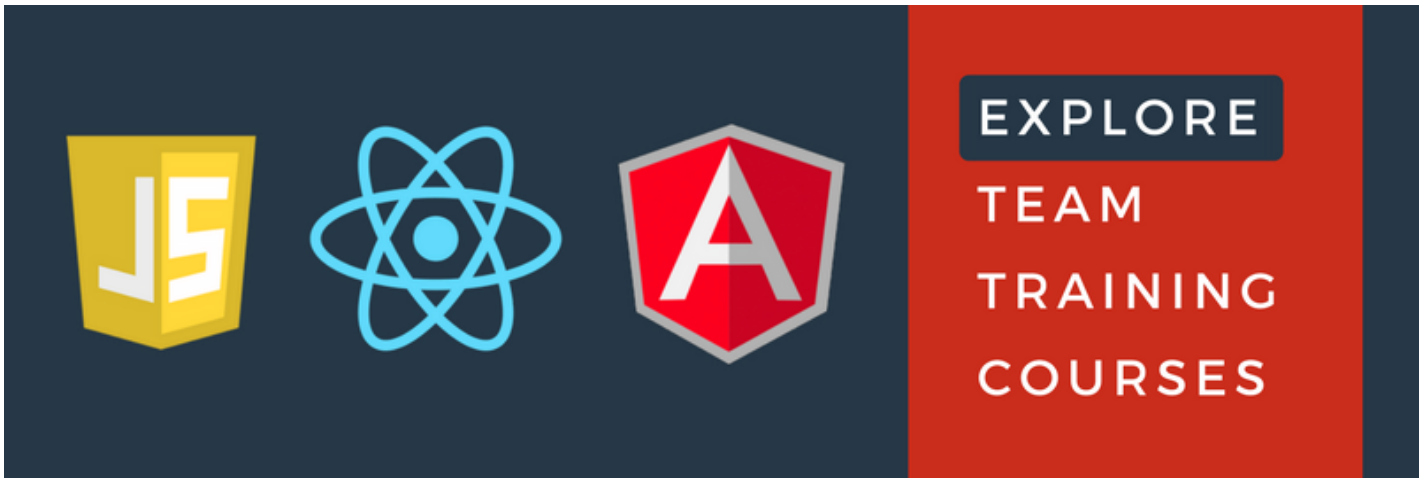
Using JavaScript forEach to do Array Iterations



This tutorial demonstrates using the Array.forEach() method with some examples.

forEach() is designed to run a function on each indexed element in an array. Starting at index[0] a function will get called on index[0], index[1], index[2], etc... forEach()

this_context is an optional parameter to set scope, and a more in depth explanation can be found at the following Array.prototype.forEach() MDN article

forEach is great for looping over an array of values to run a function on each value. Also, forEach() is a good alternative to using a for() loop. forEach let's us reuse a callback function for readability. The below example compares the readability of a for loop versus using the forEach() method.

```
var my_array = ['a', 'b', 'c'];

        for (var i=0; i<my_array.length; i++) {
            console.log(my_array[i]);
            //a b c
        }

        my_array.forEach(function(current_value) {
            console.log(current_value);
            //a b c
        });
```

Let's start with an example using forEach() to loop through an array of numbers. This example checks if each number in the javascript array is even or odd.

```
var array_of_numbers = [5, 7, 1, 9, 8, 5];

        array_of_numbers.forEach(function(current_value, inde
x, initial_array) {
            if (current_value % 2) {
                console.log('odd');
```

```
function separate_evens_from_odds(value) {
    if ( value % 2 ) {
        odd_numbers.push(value);
    }
    else {
        even_numbers.push(value);
    }
}

var array_of_numbers = [5, 7, 1, 9, 8, 5];

array_of_numbers.forEach(separate_evens_from_odds);

console.log(even_numbers); //[8]
console.log(odd_numbers); //[5, 7, 1, 9, 5]
//
```

What if we had a couple of shopping carts of items? Say you wanted to total the cost of your shopping cart. The following example totals the cost of all items from our shopping carts using the forEach() method.

```
var total_cost = 0;

function add_to_total_cost(amount) {
    total_cost += amount.cost;
}

var shopping_cart_1 = [
    {
        item: 'shirt',
```

```
            item: 'milk',
            cost: 3
        },
        {
            item: 'eggs',
            cost: 2
        }
    ]

    shopping_cart_1.forEach(add_to_total_cost);
    shopping_cart_2.forEach(add_to_total_cost);

    console.log(total_cost);
    //57
```

For more examples of looping through or iterating over javascript arrays or objects checkout this 2ality post on Iterating over arrays and objects.

While the forEach() method can be useful, forEach() does come with some disadvantages. Maybe you are wondering how to break out of a forEach() loop early. One of the negative issues with using forEach() is that there is no way to terminate or break out of the loop without throwing an exception. Since a forEach() returns a value of undefined, forEach() is not chainable like map(), filter(), or reduce().

forEach is great for looping over all values in an array to execute a function on each of those values. forEach() is also a nice alternative to using a for() loop for code reuse and readability.

Comments

Comment

Name

Email

Add Comment