**ES6 ==> ECMA Script or 2015**

**ES6 🡺 Babel 🡺 ES5** (Supported by all browsers)

**# Array Helpers Methods –**

1. forEach
2. map
3. filter
4. find
5. every
6. some
7. reduce

https://stephengrider.github.io/JSPlaygrounds/

1. **forEach**

|  |
| --- |
| //create an array of numbers  var numbers= [1,2,3,4,5,6,7];  //Create a variable to hold sum  var sum=0;  function adder(number){  sum+=number  }  //Loop over the array  numbers.forEach(adder);  //Print Sum  sum |

1. **map**

|  |
| --- |
| **var numbers= [1,2,3,4,5,6,7];**  **var doubled = numbers.map(function(number){**  **return number\*2;**  **});**    **//Print New Array**  **doubled** |

1. **filter**

|  |
| --- |
| **var products= [**  **{ name: 'orange', type: 'fruit', quantity: 0, price: 10 },**  **{ name: 'celery', type: 'vegetable' ,quantity: 5, price: 15 },**  **{ name: 'banana', type: 'fruit', quantity: 3, price: 15 },**  **{ name: 'cucumber', type: 'vegetable', quantity: 13, price: 30 }**  **];**  **products.filter(function(product) {**  **return product.type === 'fruit' && product.price >10 ;**  **});** |

|  |
| --- |
| **var post= {id:4, title: 'New Post'};**  **var comments= [**  **{ postId: 4, content: 'awesome post'},**  **{ postId: 3, content: 'awesome post'},**  **{ postId: 4, content: 'ok'},**  **{ postId: 4, content: 'good'}**  **];**  **function commentsForPost(post, comments) {**  **return comments.filter(function(comment) {**  **return comments.postId === post.Id;**  **});**  **}**  **commentsForPost(post, comments);** |

1. **find**

this helper fetches the first match and break the loop

|  |
| --- |
| **var users = [**  **{ name: 'Jill' },**  **{ name: 'Alex' },**  **{ name: 'Bill' }**  **]**  **users.find(function(user) {**  **return user.name === 'Alex';**  **});** |

1. **every and some**

every works as and operator

some works as or operator

|  |
| --- |
| var names = [  "rajat",  "sonu",  "dhubham"  ]  names.every(function(user) {  return user.length > 4;  });  names.some(function(user) {  return user.length > 4;  }); |

1. **reduce**

takes two argument, second argument is the initial value

|  |
| --- |
| var primaryColors = [  { color: 'red' },  { color: 'blue' },  { color: 'green' },  ];  primaryColors.reduce(function(previous, primaryColor) {  previous.push(primaryColor.color);    return previous  }, []); |

|  |
| --- |
| var numbers = [ 10, 20, 30];  numbers.reduce(function(sum, number) {  return sum + number  }, 0) |

|  |
| --- |
| Balanced Parenthesis Problem |
| function balancedParens(string) {  return !string.split("").reduce(function(previous, char) {  if(previous < 0 ) { return previous; }  if(char === "(" ) { return ++previous; }  if(char === ")" ) { return --previous; }  }, 0);  }  balancedParens("()()()()"); |