Grow with Google: Lesson 15

Arrays

1. Intro to Arrays

- When you want to work with a lot of data at once, use arrays.
- Arrays are a data structure that can hold multiple data values, similar to a list.

2. Donuts to Code

- Arrays help us cut down on variables and consolidate them under one name.
  - Arrays are like an ordered list that starts at 0.
  - Items in an array are delimited by commas.
  - The last item in an array does not require a comma.
- I really wish he had explained those keyboard shortcuts.

3. Creating an Array

- Arrays can store any mixture of data types, including other arrays.
- Typically, you will not store mixed data.
- Nested arrays are very useful.

```
var donuts = []; // empty array

var donuts = ["chocolate", "sprinkled", "glazed"]; // array with 3 strings

var mixedData = ["abcd", 1, true, undefined, null]; // array with different
data types

var arraysInArrays = [[1, 2, 3], ["Julia", "James"], [true, false, true,
false]];
// an array with 3 arrays of varying length

var arraysInArrays = [
  [1, 2, 3],
  ["Julia", "James"],
  [true, false, true, false]
];
// nested arrays with improved formatting
```

4. Accessing Array Elements

- Individual pieces of data in an array are elements.
- An index references the location of an element in an array.
- Indices permit us to access any value in an array.

5. Array Index

- Arrays are indexed started at zero.
  - When attempting to access an element outside the bounds of an array, at an index that does not exist, a value of `undefined` will be returned.

```
var donuts = ["glazed", "powdered", "sprinkled"];
console.log(donuts[1]); // returns the second element, "powdered"

donuts[1] = "glazed crueller"; // overwrites the second element
console.log(donuts[1]); // returns the new second element. "glazed crueller"
```

6. Quiz: UdaciFamily (6-1)

```
var udaciFamily = ["Julia", "James", "Asia"];
console.log(udaciFamily); // [ 'Julia', 'James', 'Asia' ]
```

7. Quiz: Building the Crew (6-2)

```
var captain = "Mal";
var second = "Zoe";
var pilot = "Wash";
var companion = "Inara";
var mercenary = "Jayne";
var mechanic = "Kaylee";

var crew = [captain, second, pilot, companion, mercenary, mechanic];
console.log(crew); // [ 'Mal', 'Zoe', 'Wash', 'Inara', 'Jayne', 'Kaylee' ]
```

8. Quiz: The Price is Right (6-3)

```
var prices = [1.23, 48.11, 90.11, 8.50, 9.99, 1.00, 1.10, 67.00];
```

```
prices[0] = 2.42;
prices[2] = 4.57;
prices[6] = 3.99;

console.log(prices); // [ 2.42, 48.11, 4.57, 8.5, 9.99, 1, 3.99, 67 ]
```

9. Array Properties and Methods

- JavaScript has several built-in methods for manipulating arrays.
  - MDN Array
- Methods are pre-defined functions that a data structure can call.
  - Length: Used in an earlier lesson on strings.
  - Reverse: Reverse the order of an array.
  - Sort: Sort an array.
  - Push & Pop: Add or remove items from the array.
- Arrays are a special object.

10. Length

- Returns the number of elements in an array.

```
var donuts = ["glazed", "powdered", "sprinkled"];
console.log(donuts.length); // 3
```

- Arrays inside of an array count as one array in the parent array.

11. Push

- Appends an element to the end an array.

```
var donuts = ["glazed", "chocolate frosted", "Boston creme", "glazed cruller",
"cinnamon sugar", "sprinkled"];
donuts.push("powdered"); // Returns 7. Item appended at index 6.
```

12. Pop

- Removes items from the end of an array.

```
var donuts = ["glazed", "chocolate frosted", "Boston creme", "glazed cruller",
"cinnamon sugar", "sprinkled", "powdered"];
```

```
donuts.pop(); // Returns the removed item, 'powdered.`
```

13. Splice

- Add or remove elements from anywhere within an array.
  - MDN Array Splice
- Syntax `array.splice(start[, deleteCount[, item1[, item2[, ...]]]])`
  - If `start > array.length`, `splice()` will behave like `push()`.
  - If `start < 0`, `splice()` will begin that many elements from the end of the array.
    - For example, `array.splice(-2, 1);` will remove the second to last item.
    - If the absolute value is greater than the length of the array, it will be set to 0.
  - If `deleteCount === undefined || deleteCount > array.length - start`, then all of the items from `start` through the end of the array will be deleted.
  - If `deleteCount <=== 0`, no elements are removed.
  - If no items are specified for addition, `splice()` will only remove elements from the array.

```
var donuts = ["glazed", "chocolate frosted", "Boston creme", "glazed cruller"];
donuts.splice(1, 1, "chocolate cruller", "creme de leche");
// Remove 1 item at index 1. Returns "chocolate frosted."
// Insert "chocolate crueller" and "creme de leche" at index 1.

var donuts = ["cookies", "cinnamon sugar", "creme de leche"];
donuts.splice(-2, 0, "chocolate frosted", "glazed");
// Remove no items at index 1. Nothing is returned.
// Insert "chocolate frosted" and "glazed" at index -2 (i.e. index 1).
```

14. Quiz: Colors of the Rainbow (6-4)

- It accepts multiple solutions.

```
var rainbow = ["Red", "Orange", "Blackberry", "Blue"];

rainbow.splice(2, 2, "Yellow", "Green", "Blue", "Purple");
// rainbow.splice(rainbow.length, 0, "Purple" );
// rainbow.push("Purple");
console.log(rainbow);
```

15. Quiz: Quidditch Cup (6-5)

- Only worked without an explicit else condition.
- Ternary operators throw a syntax error with returns.

```
function hasEnoughPlayers(team) {
    if(team.length >= 7) {
        return true;
    } return false;
}

var team = ["Oliver Wood", "Angelina Johnson", "Katie Bell", "Alicia Spinnet",
"George Weasley", "Fred Weasley", "Harry Potter"];
console.log(hasEnoughPlayers(team));
```

16. Quiz: Joining the Crew (6-6)

```
var captain = "Mal";
var second = "Zoe";
var pilot = "Wash";
var companion = "Inara";
var mercenary = "Jayne";
var mechanic = "Kaylee";

var crew = [captain, second, pilot, companion, mercenary, mechanic];

var doctor = "Simon";
var sister = "River";
var shepherd = "Book";

crew.push(doctor, sister, shepherd);
console.log(crew);
```

17. Quiz: Checking out the Docs (6-7)

- The `sort()` method converts integers to Unicode strings, so double digit integers precede single digit.
- Combine the elements of an array into a string with the `join()` method.

18. Array Loops

```
var donuts = ["jelly donut", "chocolate donut", "glazed donut"];

for (var i = 0; i < donuts.length; i++) {
  donuts[i] += " hole"; // Append " hole" to each string in the array.
}
```

19. The forEach Loop

- A special method for looping over the elements in an array.
  - The function passed to the `forEach()` method can take up to 3 parameters.
  - The parameters can be called element, index, and array, or anything you lik
  - The method is called *for each* element in the array.

```
var donuts = ["jelly donut", "chocolate donut", "glazed donut"];

// Inline Function
donuts.forEach(function(donut) {
  donut += " hole";
  console.log(donut);
});

// Another Example
words = ["cat", "in", "hat"];
words.forEach(function(word, num, all) {
  console.log("Word " + num + " in " + all.toString() + " is " + word);
});
/*
Word 0 in cat,in,hat is cat
Word 1 in cat,in,hat is in
Word 2 in cat,in,hat is hat
*/
```

20. Quiz: Another Type of Loop (6-8)

```
var test = [12, 929, 11, 3, 199, 1000, 7, 1, 24, 37, 4,
    19, 300, 3775, 299, 36, 209, 148, 169, 299,
```

```
    6, 109, 20, 58, 139, 59, 3, 1, 139
];

// For Loop
// for (var i = 0; i < test.length; i++) {
//     if (test[i] % 3 === 0) {
//         test[i] += 100;
//     }
// }

// ForEach Function
test.forEach(function(element, i) {
    if (element % 3 === 0) {
        test[i] += 100;
    }
});

console.log(test);
```

21. Map

- Using `forEach()` is not useful in cases where the original array should be modified.
  - Returns `undefined.`
- The `map()` method can be used to perform an operation on each element and return a new array

```
var donuts = ["jelly donut", "chocolate donut", "glazed donut"];

var improvedDonuts = donuts.map(function(donut) {
  donut += " hole";
  return donut;
});
```

22. Quiz: I Got Bills (6-9)

```
var bills = [50.23, 19.12, 34.01,
    100.11, 12.15, 9.90, 29.11, 12.99,
    10.00, 99.22, 102.20, 100.10, 6.77, 2.22
```

```
];

var totals = bills.map(function(bill) {
   bill *= 1.15;
   return Number(bill.toFixed(2));
});

console.log(totals);
```

23. Arrays in Arrays

- An array of arrays can be represented as a grid-like structure.
  - Each element corresponds to a row of sales.
  - Iterating over a row or cells can be tricky.
  - A single loop can access each row of an array.
  - A nested loop can access each cell of the row.

24. 2D Donut Arrays

```
var donutBox = [
  ["glazed", "chocolate glazed", "cinnamon"],
  ["powdered", "sprinkled", "glazed cruller"],
  ["chocolate cruller", "Boston creme", "creme de leche"]
];

for (var row = 0; row < donutBox.length; row++) {
  for (var column = 0; column < donutBox[row].length; column++) {
      console.log(donutBox[row][column]);
  }
 }

/* Prints:
"glazed"
"chocolate glazed"
"cinnamon"
"powdered"
"sprinkled"
"glazed cruller"
"chocolate cruller"
```

```
"Boston creme"
"creme de leche"
*/
```

25. Quiz: Nested Numbers (6-10)

```
var numbers = [
    [243, 12, 23, 12, 45, 45, 78, 66, 223, 3],
    [34, 2, 1, 553, 23, 4, 66, 23, 4, 55],
    [67, 56, 45, 553, 44, 55, 5, 428, 452, 3],
    [12, 31, 55, 445, 79, 44, 674, 224, 4, 21],
    [4, 2, 3, 52, 13, 51, 44, 1, 67, 5],
    [5, 65, 4, 5, 5, 6, 5, 43, 23, 4424],
    [74, 532, 6, 7, 35, 17, 89, 43, 43, 66],
    [53, 6, 89, 10, 23, 52, 111, 44, 109, 80],
    [67, 6, 53, 537, 2, 168, 16, 2, 1, 8],
    [76, 7, 9, 6, 3, 73, 77, 100, 56, 100]
];

for (var row = 0; row < numbers.length; row++) {
    for (var column = 0; column < numbers[row].length; column++) {
        if (numbers[row][column] % 2 === 0) {
            numbers[row][column] = "even";
        } else {
            numbers[row][column] = "odd";
        }
    }
}

console.log(numbers);
```

26. Lesson 6 Summary

- Other data types also have methods.
- You can create your own objects