Multiline Strings and Templates

After completing this section you will be able to:

- [ ] Create multiline strings
- [ ] Display Multiline Strings
- [ ] Understand String Interpolation with Individual Variables
- [ ] Use String Interpolation with Objects
- [ ] Create a Function that uses String Interpolation

Introduction

ES6 has introduced some convenience features for working with strings. Many of these features have existed in other languages for decades and have been requested by JavaScript developers for a long time.

In this section, we'll look at two of those features. First, we'll examine multi-line strings. Then we'll explore templates, a powerful feature that allows you to better separate content from logic in your JavaScript projects.

Multiline Strings

Multiline strings are a convenience that has existed in other programming environments but has been missing in JavaScript up until the introduction of ES6. Many strings that you have to work within JavaScript are longer than a single line, and the multiline string feature provides a convenient way for you to store them.

Consider the following string:

"*Roses are red*
Violets are Blue
I like chocolate
And so do you"

Before multiline strings, we could either ignore the carriage returns in the string or, embed HTML to display the string. (This would change the nature of our string and integrate the actual HTML characters- a far from perfect solution!)

However, with multiline strings we can store the entire string easily:

```
let poem = `Roses are red
Violets are Blue
I like chocolate
And so do you`;
```

Note the character wrapping the poem itself is not one we've seen before with strings. The backtick character looks like an apostrophe facing the wrong way and is often on the upper-left hand side of your keyboard.

Multiline Strings and HTML

Remember that if you display the string in the browser window as we created it above, it will still display on a single line. The browser doesn't respect embedded carriage returns and the `<br>` tag or other HTML would need to be used to create the carriage returns.

String Interpolation (Templates)

String interpolation (aka templating) makes creating strings from variables and string components easier. With string interpolation, you can retrieve the value of a variable from inside the string using special notation. This avoids the inconvenience of repeatedly concatenating string fragments together.

Creating an Object

First, let's create an object that we can use to create our string. We'll create an object that models a few properties of an employee:

```
let employee = {
    firstName: "John",
    lastName: "Johnson",
    position: "Production Assistant",
    location: "Norwalk, CT",
    age: 30
};

let fullName = employee.firstName + " " + employee.lastName;
```

Here we created a simple employee object. Next we access the `firstName` and `lastName` properties of the `employee` object and store them in a variable called `fullName` using the concatenation method.

Now let's use interpolation to access object properties and variables in our code.

```
console.log(`The employees name is ${fullName}`);
console.log(`The employees position is ${employee.position}`);
```

Notice that by using a combination of the curly bracket, backtick, and dollar sign we can access both the properties of an object and the value stored in a variable.

Interpolation and Functions

Interpolation also works within the functino context. Consider the following fuction:

```
let employeeInfo = function(object){
    let output = `${object.name} is a ${object.position} in our
    ${object.location} offfice and is ${object.age} years old.";
    return output;
}
```

In this function the object is passed in to the function and then each of the properties is accessed via the `object` reference. This is a bit clearer than concatenating a bunch of string fragments together.

> The implications of String Interpolation are vast. This feature allows for easy templating of content. When working with large, enterprise applications, it is often recommended that the content is separated from business logic, which is facilitated by string interpolation. Also, consider the implications for internationalization where a single codebase is used to serve users in many different languages.

Debug This:

When working correctly, the following code outputs `Greetings, I'm Tom Green` to the console. Make any necessary corrections to get this code working correctly.

```
const person : [
    firstName: "Tom"
    lastName: "Green",
    function welcome(){
        return `Greetings, I'm ${this.firstName} ${this.lastName}`;
    ]
}

console.log(person.welcome);
```

Submit This: Madder Libs

Create a program that prompts the user for the following information and stores each response in a unique variable (You may use a form or the prompt() command).

- A piece of Clothing (A)

- Part of Body (B)
- Part of Body (C)
- Verb Base Form (D)
- Part of Body (E)
- Verb Base Form (F)
- Noun Plural (G)
- Noun Plural (H)
- Verb Base Form (I)

Using String Interpolation and the Multiline String techniques taught in this section output instructions for "Flossing" using the words entered by the user:

```
How to do The Floss Dance

1 ) Put on some music
2 ) Put on a *A* ( optional )
3 ) Stand with your *B* ever so slightly bent
4 ) Put your hands in fists
5 ) Place your *C*, relaxed, at the side of your body
6 ) Begin to *D* your *E* left and right around your torso in the opposite direction to your h
7 ) Keep doing it and increase speed and *F* *G* occasionally for effect
After building confidence, you can also *H* *I* with your arms if desired, but this is not mar
```

Please save your file in the following format to ensure proper credit: `LastName_ES6_Strings.html`.

For this course visit https://www.dropbox.com/request/Nl7bSaAe11Zl0PgLRonA to submit your assignments.