Template Literals

introduction to template literals and its usage

The purpose of *template literals* is to evaluate and insert values of JavaScript expressions in a template string.

```
let x = 555; //<-notice 555
let evaluatedTemplate = `${x} === 555 is ${x === 555}`;
console.log(evaluatedTemplate);
// evaluatedTemplate becomes "555 === 555 is true"

let y = '555'; //<-notice '555'
evaluatedTemplate = `${y} === 555 is ${y === 555}`;
console.log(evaluatedTemplate);
// evaluatedTemplate becomes "555 === 555 is false"
```

> One practical application of template literals is the creation of DOM markup with static structure, and dynamic values.

For more information, check out my article on Underscore templating. If you compare the ES6 syntax, you will find that it is a lot more compact than the Underscore syntax.

> Another application of template literals is to simplify writing strings that span multiple lines:
>
> ```
> `first
> second
> third
> fourth`
> ```

In the next lesson, let’s talk about how to perform a transformation on a template literal.