Why won't my Template Strings work?

Posted At : October 25, 2018 10:00 AM | Posted By : Jeffry Houser
Related Categories: JavaScript, Professional

As you probably know, I have been doing a lot of work with Angular, TypeScript, ES6, and the surrounding ecosystem. I've started using template strings a bunch.

This was the old way to create a string from many variables:

```
myNewValue = "http://" + myDomain + myPath + "?value=" + myValue;
```

It worked but got tedious. ES6 introduced something called template strings. Instead of using all those plus signs to concatenate variables, we can do use a friendly syntax:

```
myNewValue = `http://${myDomain}${myPath}"?value=${myValue}`;
```

I've read about this, done a few proof of principle samples, but for the first time I'm actually using this on a project and integrating it with code.

These are the two mistakes I make most common.

Use the Grave Accent

A template string must be enclosed in a grave accent, or backtick. This is the same key you use when formatting inline code in a StackOverflow comment or inside Slack. If you use double quotes or regular single quotes, your string will not be recognized as a template string and the variables will not turned into real actual values.

This works:

```
`http://${myDomain}${myPath}"?value=${myValue}`
```

But, this will not:

```
"http://${myDomain}${myPath}"?value=${myValue}"
```

Nor will this:

```
'http://${myDomain}${myPath}"?value=${myValue}'
```

I picked this one up the hard way after a particularly frustrating debug session wondering why all my unit tests suddenly broke.

Use Curly Brackets, not Parenthesis

The values that you inject in the string must be surrounded by curly brackets, like an object in JavaScript. For some reason my mind wants to use parenthesis, as if calling a function.

This works:

```
`http://${myDomain}${myPath}"?value=${myValue}`
```

But, this will fail miserably:

```
`http://$(myDomain)$(myPath)"?value=$(myValue)`
```

Despite these two minor roadblocks, I'm finding a lot of value in template strings because the new way of string concatenation is a lot easier to read than the old way.