A Drip of JavaScript

Truthy and Falsy Values in JavaScript

Originally published in the A Drip of JavaScript newsletter.

Longtime JavaScript developers often toss around the terms "truthy" and "falsy", but for those who are newer to JavaScript these terms can be a bit mystifying.

When we say that a value is "truthy" in JavaScript, we don't just mean that the value is `true`. Rather, what we mean is that the value coerces to `true` when evaluated in a boolean context. Let's look at what that means.

```
function logTruthiness (val) {
    if (val) {
        console.log("Truthy!");
    } else {
        console.log("Falsy.");
    }
}
```

This function takes the `val` parameter and evaluates it in a boolean context (the condition of the `if` statement.) So let's try it out on some values.

```
// Outputs: "Truthy!"
logTruthiness(true);

// Outputs: "Truthy!"
logTruthiness({});
```

```
// Outputs: "Truthy!"
logTruthiness([]);

// Outputs: "Truthy!"
logTruthiness("some string");

// Outputs: "Truthy!"
logTruthiness(3.14);

// Outputs: "Truthy!"
logTruthiness(new Date());
```

As you can see, there are a lot of truthy values in JavaScript. And there are many more than could be listed here. On the other side, though, there are only six falsy values. In fact, because the list of falsy values is so short, memorizing that list is the easiest way to tell whether a value is truthy or falsy. Here is the list:

```
// Outputs: "Falsy."
logTruthiness(false);

// Outputs: "Falsy."
logTruthiness(null);

// Outputs: "Falsy."
logTruthiness(undefined);

// Outputs: "Falsy."
logTruthiness(NaN);

// Outputs: "Falsy."
logTruthiness(0);

// Outputs: "Falsy."
logTruthiness("");
```

That's a pretty straightforward list. But how can we actually use truthiness? Let's look at an example.

```
function reportAttitude (person) {
    if (person.skepticism) {
        console.log(person.name +
                    " is skeptical about " +
                    person.skepticism);
    } else {
        console.log(person.name + " wants to believe.");
    }
}

var mulder = {
    name: "Fox Mulder"
};

var scully = {
    name: "Dana Scully",
    skepticism: "UFOs & conspiracy theories"
};

var frohikey = {
    name: "Melvin Frohikey",
    skepticism: ""
};

// Outputs: "Fox Mulder wants to believe."
reportAttitude(mulder);

// Outputs: "Dana Scully is skeptical about UFOs and conspiracy theorie
reportAttitude(scully);

// Outputs: "Melvin Frohikey wants to believe."
reportAttitude(frohikey);
```

Taking advantage of truthiness can make your code a little bit more concise. We don't need to explicitly check for `undefined`, `""`, etc. Instead we can just check whether `person.skepticism` is truthy. However, there some caveats to keep in mind.

Firstly, this approach only works if **all** of the falsy values should be excluded (or included.) For instance, if a value of `0` or `""` was meaningful, then the `if` above would not function correctly.

Secondly, it is important to keep in mind that truthiness is not the same as `== true`. The algorithm for loose equality is much more complicated than for truthiness. We'll go into more detail about that in a future drip.

I hope this introduction to truthiness and falsiness helps you to write clearer, more concise JavaScript.

Thanks for reading!

Josh Clanton
