A Quick Look at Octal and Binary Literals in ES6

Summary: in this tutorial, you will learn how to represent the octal and binary literals in ES6.

ES5 provided numeric literals in octal (prefix `0`), decimal (no prefix), and hexadecimal (`0x`). ES6 added support for binary literals and changed how it represents octal literals.

Octal literals

To represent an octal literal in ES5, you use the zero prefix (`0`) followed by a sequence of octal digits (from 0 to 7). For example:

```
let a = 051;
console.log(a); // 41
```

If the octal literal contains a number that is out of range, JavaScript ignores the leading 0 and treats the octal literal as a decimal, as shown in the following example:

```
let b = 058; // invalid octal
console.log(b); // 58
```

In this example, since `8` is an invalid digit for representing the octal number, JavaScript ignores the 0 and treats the whole number as a decimal with a value of 58.

Note you can use the octal literals in non-strict mode. If you use them in strict mode, JavaScript will throw an error.

```
"use strict"
let b = 058; // invalid octal
console.log(b);
```

Here is the error message:

```
SyntaxError: Decimals with leading zeros are not allowed in strict mode.
```

ES6 allows you to specify the octal literal by using the prefix `0o` followed by a sequence of octal digits from 0 through 7. Here is an example:

```
let c = 0o51;
console.log(c); // 41
```

If you use an invalid number in the octal literal, JavaScript will throw a `SyntaxError` as shown in the following example:

```
let d = 0o58;
console.log(d); // SyntaxError
```

Binary literals

In ES5, JavaScript didn't provide any literal form for binary numbers. To parse a binary string, you use the `parseInt()` function as follows:

```
let e = parseInt('111',2);
console.log(e); // 7
```

ES6 added support for binary literals by using the `0b` prefix followed by a sequence of binary numbers (0 and 1). Here is an example:

```
let f = 0b111;
console.log(f); // 7
```

Summary

- Octal literals start with `0o` followed by a sequence of numbers between 0 and 7.

- Binary literals start with `0b` followed by a sequence of numbers 0 and 1.