JavaScript Regex Lookbehind

Summary: in this tutorial, you'll learn how to use JavaScript regex lookbehind in regular expressions to match `X` if it is preceded by `Y`.

Introduction to the JavaScript regex lookbehind

In regular expressions, a lookbehind matches an element if there is another specific element before it. A lookbehind has the following syntax:

```
(?<=Y)X
```

In this syntax, the pattern match `X` if there is `Y` before it.

For example, suppose you want to match the number `900` not the number `1` in the following string:

```
'1 computer costs $900'
```

To do it, you use a lookahead in the regular expression as follows:

```
/(?<=\$)\d+/
```

In this regular expression:

- The `(?<=\$)` matches an element if there is a literal string `$` before it. Because `$` is a special character in the regex, we need to use the backslash `\` to escape it. By doing this, the regex engine treats `\$` as a literal character `$`.
- The `\d+` matches one or more digits.

The following example illustrates how to use a lookbehind in a regular expression to match a number that has the `$` sign before it:

```
const s = '1 computer costs $900';
const pattern = /(?<=\$)\d+/;

const match = s.match(pattern);
console.log(match);
```

Output:

```
[ '900', index: 18, input: '1 computer costs $900', groups: undefined ]
```

Negative lookbehind

To negate a lookbehind, you use a negative lookbehind with the following syntax:

```
(?<!Y)X
```

In this syntax, the regex engine matches `X` if there is **no** `Y` before it. The following example uses a regular expression with a negative lookbehind to match a number that doesn't have the `$` letter before it:

```
const s = '1 computer costs $900';
const pattern = /(?<!\$)\d+/;

const match = s.match(pattern);
console.log(match);
```

Output:

```
1
```

Summary

- A lookbehind `(?<!Y)X` matches `X` only if is is preceded by `Y`.
- A negative lookbehind `(?<!Y)X` matches `X` only if it is not preceded by `Y`.