Regular Expression: Quantifiers

Summary: in this tutorial, you'll learn to use quantifiers to match several instances of a character, group, or character class in a string.

Quantifiers specify how many instances of a character, group, or character class must appear in the input string for a match to be found.

Quantity

Exact count {n}

A number in curly braces `{n}` is the simplest quantifier. When you append it to a character or character class, it specifies how many characters or character classes you want to match.

For example, the regular expression `/\d{4}/` matches a four-digit number. It is the same as `/\d\d\d\d/` :

```
let str = 'ECMAScript 2020';
let re = /\d{4}/;

let result = str.match(re);

console.log(result);
```

Output:

```
['2020', index: 11, input: 'ECMAScript 2020', groups: undefined]
```

The range {n,m}

The range matches a character or character class from `n` to `m` times.

For example, to find numbers that have two, three, or four digits, you use the regular expression `/\d{2,4}/g` :

```
let str = 'The official name of ES11 is ES2020';
let re = /\d{2,4}/g;

let result = str.match(re);
console.log(result);
```

Output:

```
["11", "2020"]
```

Because the upper limit is optional, the `{n,}` searches for a sequence of `n` or more times. For example, the regular expression `/\d{2,}/` will match any number that has two or more digits.

```
let str = 'The official name of ES6 is ES2015';
let re = /\d{2,}/g;

let result = str.match(re);
console.log(result);
```

Output:

```
["2015"]
```

The following example uses the regular expression `/\d{1,}/g` to match any numbers that have one or more digits in a phone number:

```
let numbers = '+1-(408)-555-0105'.match(/\d{1,}/g);
console.log(numbers);
```

Output:

```
["1", "408", "555", "0105"]
```

Shorthands

+

The quantifier `{1,}` means one or more which has the shorthand as `+`. For example, the `\d+` searches for numbers:

```
let phone = "+1-(408)-555-0105";
let result = phone.match(/\d+/g);

console.log(result);
```

Output:

```
["1", "408", "555", "0105"]
```

?

The quantifier `?` means zero or one. It is the same as `{0,1}`. For example, `/colou?r/` will match both `color` and `colour`:

```
let str = 'Is this color or colour?';
let result = str.match(/colou?r/g);

console.log(result);
```

Output:

```
["color", "colour"]
```

*

The quantifier `*` means zero or more. It is the same as `{0,}`. The following example shows how to use the quantifier `*` to match the string `Java` followed by any word character:

```
let str = 'JavaScript is not Java';
let re = /Java\w*/g

let results = str.match(re);
console.log(results);
```

Output:

```
["JavaScript", "Java"]
```

We often use quantifiers to form complex regular expressions. The following shows some regular expression examples that include quantifiers:

- Whole numbers: `/^\d+$/`
- Decimal numbers: `/^\d*.\d+$/`
- Whole numbers and decimal numbers: `/^\d*(.\d+)?$/`
- Negative, positive whole numbers & decimal numbers: `/^-?\d*(.\d+)?$/`

Summary

The following table lists the quantifiers:

Quantifier	Description
*	Match zero or more times.
+	Match one or more times.
?	Match zero or one time.
{ *n* }	Match exactly *n* times.

Quantifier	Description
{ *n* , }	Match at least *n* times.
{ *n* , *m* }	Match from *n* to *m* times.