Regular Expression: Word Boundaries

Summary: in this tutorial, you'll learn how to use the word boundary in regular expressions.

The `(\b)` is an anchor like the caret (`^`) and the dollar sign (`$`). It matches a *position* that is called a "word boundary". The word boundary match is zero-length.

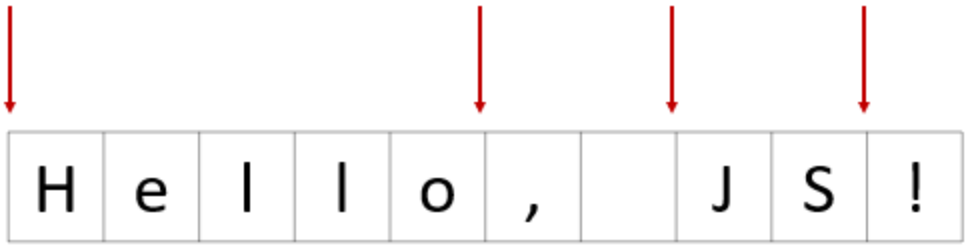
The following three positions are qualified as word boundaries:

- Before the first character in a string if the first character is a word character.
- After the last character in a string if the last character is a word character.
- Between two characters in a string if one is a word character and the other is not.

Simply put, the word boundary `\b` allows you to match the whole word using a regular expression in the following form:

```
\bword\b
```

For example, in the string `Hello, JS!` the following positions qualify as a word boundary:



The following example returns `'JS'` because `'Hello, JS!'` matches the regular expression `/\bJS\b/`:

```
console.log('Hello, JS!'.match(/\bJS\b/));
```

Output:

```
['JS', index: 7, input: 'Hello, JS!', groups: undefined]
```

However, the `'Hello, JScript'` doesn't match `/\bJS\b/` :

```
console.log('Hello, JSscript!'.match(/\bJS\b/));
```

Output:

```
null
```

Note that without `\b` , the `/JS/` matches both `'Hello, JS'` and `'Hello, JScript'` :

```
console.log('Hello, JSscript!'.match(/JS/));
console.log('Hello, JS!'.match(/JS/));
```

Output:

```
['JS', index: 7, input: 'Hello, JSscript!', groups: undefined]
['JS', index: 7, input: 'Hello, JS!', groups: undefined]
```

It's possible to use the word boundary with digits.

For example, the regular expression `\b\d\d\d\d\b` matches a 4-digit number surrounded by characters different from `\w` :

```
console.log('ES 2015'.match(/\b\d\d\d\d\b/));
```

Output:

```
['2015', index: 3, input: 'ES 2015', groups: undefined]
```

The following example uses the word boundary to find the time that has the format `hh:mm` e.g., `09:15` :

```
let str = 'I start coding JS at 05:30 AM';
let re = /\b\d\d:\d\d\b/;
let result = str.match(re);

console.log(result);
```

Output:

```
['05:30', index: 21, input: 'I start coding JS at 05:30 AM', groups: undefined]
```

> It's important to note that the `\b` doesn't work for non-Latin alphabets.

As you have seen so far, the patterns `\d\d\d\d` and `\d\d` has been used to match a four-digit or a two-digit number.

It'll be easier and more flexible if you use quantifiers that will be covered in the quantifiers tutorial. You can use `\d{4}` instead of `\d\d\d\d` , which is much shorter.

Summary

- The `\b` anchor matches a word boundary position.