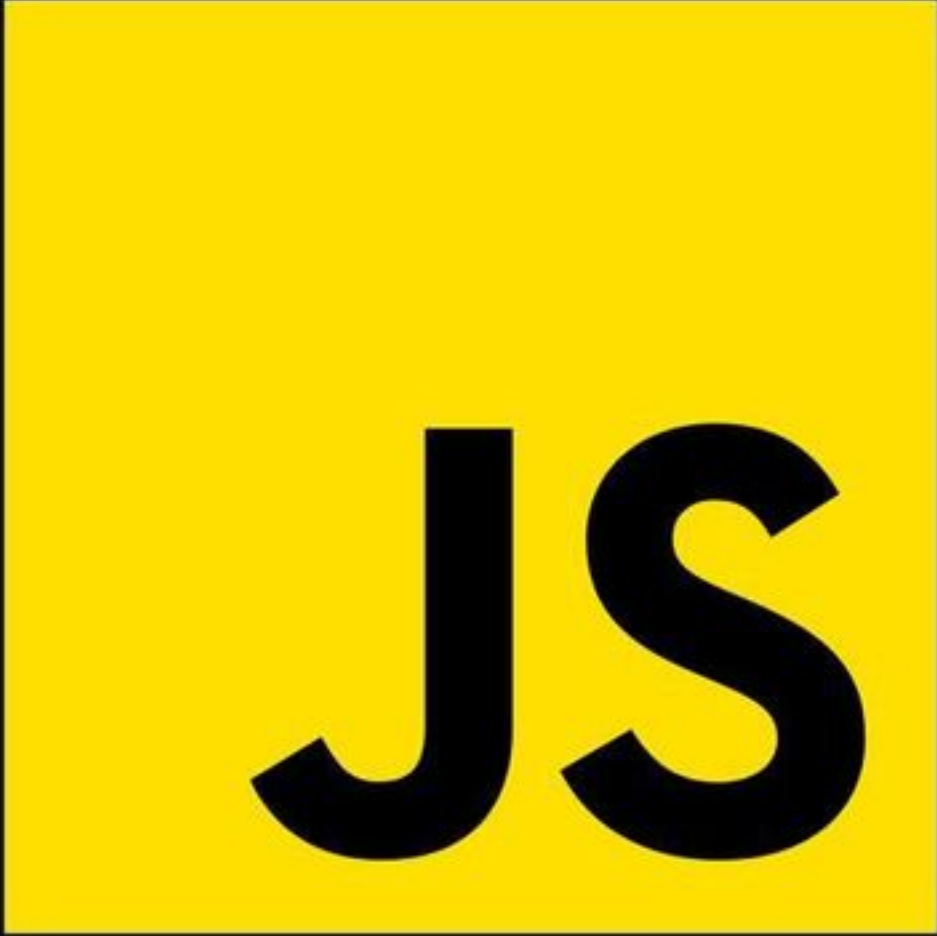


JS

JS Regex CheatSheet

A large yellow square containing the letters 'JS' in a bold, black, sans-serif font, representing the JavaScript logo.

Saad Irfan
@DevWithSaad



Create Regex

There are two ways you can write regular expression in JS.

1. Write it between two forward slashes
2. Use RegExp constructor

```
index.js

// 1. using forward slashes
const regex = /pattern/;

// 2. using RegExp constructor
const regex = new RegExp('pattern');
```



Matching a Regex

You can use the `.test()` method of regex to check if a string matches the pattern.



index.js

```
const regex = /hello/;
const str = 'Hello, World!';
console.log(regex.test(str)); // false

// using "i" flag for case-insensitive matching
const regex = /hello/i;
const str = 'Hello, World!';
console.log(regex.test(str)); // true
```



Saad Irfan
@DevWithSaad



Character Classes

A character class is a way to match any one of a set of characters. It allows you to specify a group of characters that you want to match within a larger pattern.

```
index.js

const regex = /[aeiou]/; // matching any vowel
console.log(regex.test('hello')); // true
console.log(regex.test('world')); // false

const regex = /[0-9]/; // matching any digit
console.log(regex.test('hello')); // false
console.log(regex.test('123')); // true
```



Quantifiers

a quantifier specifies how many times a particular character or group of characters should appear in the input string.

```
index.js

// matching "a" followed by one or more "b"
const regex = /ab+/;
console.log(regex.test('ab')); // true
console.log(regex.test('abb')); // true
console.log(regex.test('a')); // false

// matching "a" repeated 2 to 4 times
const regex = /a{2,4}/;
console.log(regex.test('a')); // false
console.log(regex.test('aa')); // true
console.log(regex.test('aaa')); // true
console.log(regex.test('aaaa')); // true
console.log(regex.test('aaaaa')); // false
```



Saad Irfan
@DevWithSaad



Anchors

Anchors are used to match a particular position within the input string, rather than a specific character or group of characters.

```
index.js

// matching "hello" at the beginning of a string
// (^) caret matches the beginning of a string
const regex = /^hello/;
console.log(regex.test('hello, world')); // true
console.log(regex.test('world, hello')); // false

// matching "world" at the end of a string
// ($) dollar matches the beginning of a string
const regex = /world$/;
console.log(regex.test('hello, world')); // true
console.log(regex.test('world, hello')); // false
```



Saad Irfan
@DevWithSaad



Groups

Groups allow you to group together multiple characters or sub-patterns and apply quantifiers and other operators to the entire group. You create a group with ().



index.js

```
// matching "hello" or "world"
const regex = /(hello|world)/;

console.log(regex.test('hello, world')); //
console.log(regex.test('world, hello')); //
console.log(regex.test('hi')); // false
```



Saad Irfan
@DevWithSaad





Saad Irfan

@DevWithSaad

Did You Find it Useful?

Share Your thoughts.

