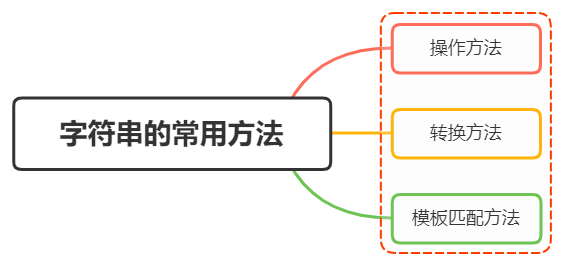
# 面试官：JavaScript字符串的常用方法有哪些？



## 一、操作方法

我们也可将字符串常用的操作方法归纳为增、删、改、查，需要知道字符串的特点是一旦创建了，就不可变

### 增

这里增的意思并不是说直接增添内容，而是创建字符串的一个副本，再进行操作

除了常用+以及${}进行字符串拼接之外，还可通过concat

#### concat

用于将一个或多个字符串拼接成一个新字符串

let stringValue = "hello ";  
let result = stringValue.concat("world");  
console.log(result); // "hello world"  
console.log(stringValue); // "hello"

### 删

这里的删的意思并不是说删除原字符串的内容，而是创建字符串的一个副本，再进行操作

常见的有：

* slice()
* substr()
* substring()

这三个方法都返回调用它们的字符串的一个子字符串，而且都接收一或两个参数。

let stringValue = "hello world";  
console.log(stringValue.slice(3)); // "lo world"  
console.log(stringValue.substring(3)); // "lo world"  
console.log(stringValue.substr(3)); // "lo world"  
console.log(stringValue.slice(3, 7)); // "lo w"  
console.log(stringValue.substring(3,7)); // "lo w"  
console.log(stringValue.substr(3, 7)); // "lo worl"

### 改

这里改的意思也不是改变原字符串，而是创建字符串的一个副本，再进行操作

常见的有：

* trim()、trimLeft()、trimRight()
* repeat()
* padStart()、padEnd()
* toLowerCase()、 toUpperCase()

#### trim()、trimLeft()、trimRight()

删除前、后或前后所有空格符，再返回新的字符串

let stringValue = " hello world ";  
let trimmedStringValue = stringValue.trim();  
console.log(stringValue); // " hello world "  
console.log(trimmedStringValue); // "hello world"

#### repeat()

接收一个整数参数，表示要将字符串复制多少次，然后返回拼接所有副本后的结果

let stringValue = "na ";  
let copyResult = stringValue.repeat(2) // na na

#### padEnd()

复制字符串，如果小于指定长度，则在相应一边填充字符，直至满足长度条件

let stringValue = "foo";  
console.log(stringValue.padStart(6)); // " foo"  
console.log(stringValue.padStart(9, ".")); // "......foo"

### toLowerCase()、 toUpperCase()

大小写转化

let stringValue = "hello world";  
console.log(stringValue.toUpperCase()); // "HELLO WORLD"  
console.log(stringValue.toLowerCase()); // "hello world"

### 查

除了通过索引的方式获取字符串的值，还可通过：

* chatAt()
* indexOf()
* startWith()
* includes()

#### charAt()

返回给定索引位置的字符，由传给方法的整数参数指定

let message = "abcde";  
console.log(message.charAt(2)); // "c"

#### indexOf()

从字符串开头去搜索传入的字符串，并返回位置（如果没找到，则返回 -1 ）

let stringValue = "hello world";  
console.log(stringValue.indexOf("o")); // 4

#### startWith()、includes()

从字符串中搜索传入的字符串，并返回一个表示是否包含的布尔值

let message = "foobarbaz";  
console.log(message.startsWith("foo")); // true  
console.log(message.startsWith("bar")); // false  
console.log(message.includes("bar")); // true  
console.log(message.includes("qux")); // false

## 二、转换方法

### split

把字符串按照指定的分割符，拆分成数组中的每一项

let str = "12+23+34"  
let arr = str.split("+") // [12,23,34]

## 三、模板匹配方法

针对正则表达式，字符串设计了几个方法：

* match()
* search()
* replace()

### match()

接收一个参数，可以是一个正则表达式字符串，也可以是一个RegExp对象，返回数组

let text = "cat, bat, sat, fat";  
let pattern = /.at/;  
let matches = text.match(pattern);  
console.log(matches[0]); // "cat"

### search()

接收一个参数，可以是一个正则表达式字符串，也可以是一个RegExp对象，找到则返回匹配索引，否则返回 -1

let text = "cat, bat, sat, fat";  
let pos = text.search(/at/);  
console.log(pos); // 1

### replace()

接收两个参数，第一个参数为匹配的内容，第二个参数为替换的元素（可用函数）

let text = "cat, bat, sat, fat";  
let result = text.replace("at", "ond");  
console.log(result); // "cond, bat, sat, fat"