




15条高效的JS技巧，你千万要收藏好了

学致-web前端

已认证的官方帐号

关注他

9 人赞同了该文章

1、延迟函数delay

```
const delay = ms => new Promise((resolve, reject) => setTimeout(resolve, ms))

const getData = status => new Promise((resolve, reject) => {
  status ? resolve('done') : reject('fail')
})

const getRes = async (data) => {
  try {
    const res = await getData(data)
    const timestamp = new Date().getTime()
    await delay(1000)
    console.log(res, new Date().getTime() - timestamp)
  } catch (error) {
    console.log(error)
  }
}

getRes(true) // 隔了1秒
```

2、分割指定长度的元素数组

```
const listChunk = (list, size = 1, cacheList = []) => {
  const tmp = [...list]
  if (size <= 0) {
    return cacheList
  }
  while (tmp.length) {
    cacheList.push(tmp.splice(0, size))
  }
  return cacheList
}

console.log(listChunk([1, 2, 3, 4, 5, 6, 7, 8, 9])) // [[1], [2], [3], [4], [5], [6],
console.log(listChunk([1, 2, 3, 4, 5, 6, 7, 8, 9], 3)) // [[1, 2, 3], [4, 5, 6], [7,
console.log(listChunk([1, 2, 3, 4, 5, 6, 7, 8, 9], 0)) // []
console.log(listChunk([1, 2, 3, 4, 5, 6, 7, 8, 9], -1)) // []
```

```
const intersection = (list, ...args) => list.filter(item => args.every(list => list.in

console.log(intersection([2, 1], [2, 3])) // [2]
console.log(intersection([1, 2], [3, 4])) // []
```

4、函数柯里化

```
const curring = fn => {
  const { length } = fn
  const curried = (...args) => {
    return (args.length >= length
      ? fn(...args)
      : (...args2) => curried(...args.concat(args2)))
  }
  return curried
}

const listMerge = (a, b, c) => [a, b, c]
const curried = curring(listMerge)
console.log(curried(1)(2)(3)) // [1, 2, 3]

console.log(curried(1, 2)(3)) // [1, 2, 3]

console.log(curried(1, 2, 3)) // [1, 2, 3]
```

5、字符串前面空格去除与替换

```
const trimStart = str => str.replace(new RegExp('^([\\s]*)(.*)$'), '$2')
console.log(trimStart(' abc ')) // abc
console.log(trimStart('123 ')) // 123
```

6、字符串后面空格去除与替换

```
const trimEnd = str => str.replace(new RegExp('^.*?)([\\s]*$'), '$1')
console.log(trimEnd(' abc ')) // abc
console.log(trimEnd('123 ')) // 123
```

7、获取当前子元素是其父元素下子元素的排位

```
const getIndex = el => {
  if (!el) {
    return -1
  }
  let index = 0
  do {
    index++
  } while (el = el.previousElementSibling);
  return index
}
```

8、获取当前元素相对于document的偏移量

▲

赞同 9

▼

分享

```
        top,
        left
    } = el.getBoundingClientRect()
    const {
        scrollTop,
        scrollLeft
    } = document.body
    return {
        top: top + scrollTop,
        left: left + scrollLeft
    }
}
```

9、获取元素类型

```
const dataType = obj => Object.prototype.toString.call(obj).replace(/^\[object (.+)\]\$/,
```

10、判断是否是移动端

```
const isMobile = () => 'ontouchstart' in window
```

11、fade动画

```
const fade = (el, type = 'in') {
    el.style.opacity = (type === 'in' ? 0 : 1)
    let last = +new Date()
    const tick = () => {
        const opacityValue = (type === 'in'
                                ? (new Date() - last) / 400
                                : -(new Date() - last) / 400)
        el.style.opacity = +el.style.opacity + opacityValue
        last = +new Date()
        if (type === 'in'
            ? (+el.style.opacity < 1)
            : (+el.style.opacity > 0)) {
            requestAnimationFrame(tick)
        }
    }
    tick()
}
```

12、将指定格式的字符串解析为日期字符串

```
const dataPattern = (str, format = '-') => {
    if (!str) {
        return new Date()
    }
    const dateReg = new RegExp(`^((\\d{2})${format})(\\d{2})${format}(\\d{4})$`)
    const [, month, day, year] = dateReg.exec(str)
    return new Date(`-${month}, ${day} ${year}`)
}
```

```
console.log(dataPattern('12-25-1995')) // Mon Dec 25 1995 00:00:00 GMT+0800 (中国标准时
```

▲

赞同 9

▼

分享

```
const html = document.querySelector('html') html.oncopy = () => false html.onpaste = (
```

14、input框限制只能输入中文

```
const input = document.querySelector('input[type="text"]')
const clearText = target => {
  const {
    value
  } = target
  target.value = value.replace(/^[^\u4e00-\u9fa5]/g, '')
}
input.onfocus = ({target}) => {
  clearText(target)
}
input.onkeyup = ({target}) => {
  clearText(target)
}
input.onblur = ({target}) => {
  clearText(target)
}
input.oninput = ({target}) => {
  clearText(target)
}
```

15、去除字符串的html代码

```
const removehtml = (str = '') => str.replace(/<[\/\!]*[^\>]*>/ig, '')
console.log(removehtml('<h1>哈哈哈哈<呵呵呵</h1>')) // 哈哈哈哈<呵呵呵
百度网盘：http://6tt.co/uNn7 提取码：gbut 2020年最新web前端资料分享·
萌新入门必备
```

本文完~

作者：[尘不正经丶](#)

原文地址：[Node JS 的未来是什么？](#)

来源：今日头条

文章著作权归作者所有，如有侵权，请联系小编删除。

编辑于 08-25

[前端开发](#) [JS 调试](#) [前端入门](#)

文章被以下专栏收录



前端知识
学习、收集和整理

关注专栏

推荐阅读

JS 语言的下一代标准，已经在 2015年6月正式发布了。它的目标，是使得 JS 语言可以用来编写复杂的大型应用程序，成为企业级开发语言。接下来咱们来看看 20 ...
爱前端不爱... 发表于web前端...

= user.address && user.address.street;有了这个新语法，你可以写成 var street = user.address?.street你可能还写过这样的代码 var foolInput = ...
方应杭 发表于饥人谷前端...

过这种感觉 你
用？代码使用
巧？命名和编
些都是不良编
篇文章中，我
爱前端不爱...

1 条评论

切换为时间排序

写下你的评论...



成都的大熊猫

08-25

延迟函数直接setTimeout(fuction:{},1000)



赞

