

String基本包装类型

slice() 只是截取, **splice()**, 选取删除+插入 **split()**分开

```
// 下面代码的问题?  
// s1是基本类型, 基本类型是没有方法的  
var s1 = 'zhangsan';  
var s2 = s1.substring(5);  
  
// 当调用s1.substring(5)的时候, 先把s1包装成String类型的临时对象, 再调用substring方法, 最后销毁临时对象, 相当于:  
var s1 = new String('zhangsan');  
var s2 = s1.substring(5);  
s1 = null;
```

- 字符串的不可变

```
var str = 'abc';  
str = 'hello';  
// 当重新给str赋值的时候, 常量'abc'不会被修改, 依然在内存中  
// 重新给字符串赋值, 会重新在内存中开辟空间, 这个特点就是字符串的不可变  
// 由于字符串的不可变, 在大量拼接字符串的时候会有效率问题
```

- 创建字符串对象

```
var str = new String('Hello World');  
  
// 获取字符串中字符的个数|  
console.log(str.length);
```

- 字符串对象的常用方法

字符串所有的方法，都不会修改字符串本身(字符串是不可变的)，操作完成会返回一个新的字符串

```
// 1 字符方法
charAt()          //获取指定位置处字符
charCodeAt()      //获取指定位置处字符的ASCII码
str[0]            //HTML5, IE8+支持 和charAt()等效
// 2 字符串操作方法
concat()          //拼接字符串，等效于+，+更常用
slice()           //从start位置开始，截取到end位置，end取不到
substring()       //从start位置开始，截取到end位置，end取不到
substr()          //从start位置开始，截取length个字符
// 3 位置方法
indexOf()          //返回指定内容在元字符串中的位置
lastIndexOf()      //从后往前找，只找第一个匹配的
// 4 去除空白
trim()            //只能去除字符串前后的空白
// 5 大小写转换方法
toLocaleUpperCase() //转换大写
toLocaleLowerCase() //转换小写
// 6 其它
search()
replace()
split()
fromCharCode()
// String.fromCharCode(101, 102, 103); //把ASCII码转换成字符串
```

{代替python数组，js键值对唯一

使用花括号 {} 创建的是对象字面量，用于创建一个新的空对象。而使用方括号 [] 创建的是数组字面量，用于创建一个新的空数组。

every(),forEach()看fiveComplicatedFunction.js文件

splice(0,array.length) // 清空

splice(start,endNoInclude, 插入东西)

删除前面startToEndNoInclude, 并插入东西，谓之拼接

Array对象

```
// 1. 使用构造函数创建数组对象
// 创建了一个空数组
var arr = new Array();
// 创建了一个数组，里面存放了3个字符串
var arr = new Array('zs', 'ls', 'ww');
// 创建了一个数组，里面存放了4个数字
var arr = new Array(1, 2, 3, 4);

// 2. 使用字面量创建数组对象
var arr = [1, 2, 3];

// 获取数组中元素的个数
console.log(arr.length);
```

- toString()/valueOf()
 - toString() 把数组转换成字符串，逗号分隔每一项
 - valueOf() 返回数组对象本身
- 数组常用方法
演示：push()、shift()、unshift()、reverse()、sort()、splice()、indexOf()

```
> ar
< ▶ (4) [1, 2, 3, 'a']
> ar.toString()
< '1,2,3,a'
> ar.valueOf()
< ▶ (4) [1, 2, 3, 'a']
```

Date 对象

```
> var now = new Date();  
< undefined  
> now.valueOf()  
< 1696427719408  
> now.toString()  
< 'Wed Oct 04 2023 21:55:19 GMT+0800 (China Standard Time)'  
> now instanceof Date  
< true
```

- 日期格式化方法

```
toString()    // 转换成字符串  
valueOf()     // 获取毫秒值  
// 下面格式化日期的方法，在不同浏览器可能表现不一致  
toDateStrin()  
getTimeStrin()  
toLocaleDateStrin()  
toLocaleTimeStrin()
```

- 获取日期指定部分

```
getTime()      // 返回毫秒数和valueOf()结果一样  
getMilliseconds()  
getSeconds()   // 返回0-59  
getMinutes()   // 返回0-59  
getHours()     // 返回0-23  
getDay()       // 返回星期几 0周日 6周六  
getDate()      // 返回当前月的第几天  
getMonth()     // 返回月份，***从0开始***  
getFullYear() // 返回4位的年份 如 2016
```

Math对象 静态成员

```
Math.PI           // 圆周率
Math.random()     // 生成随机数
Math.floor()/Math.ceil() // 向下取整/向上取整
Math.round()      // 取整，四舍五入
Math.abs()        // 绝对值
Math.max()/Math.min() // 求最大和最小值

Math.sin()/Math.cos() // 正弦/余弦
Math.power()/Math.sqrt() // 求指数次幂/求平方根
```

内置对象

JavaScript中的对象分为3种：内置对象、浏览器对象、自定义对象

JavaScript 提供多个内置对象：Math/Array/Number/String/Boolean...

<https://developer.mozilla.org/zh-CN/docs/Web>

Ajax JQuery科普

AJAX 是异步 JavaScript 和 XML 的缩写(Asynchronous JavaScript and XML)。

jQuery是一个常见的 Javascript 库。以下是 jquery 的一些主要特点：

数组 循环和C都很像


```
function fn(b) {  
    var a = 10;  
    return function () {  
        alert(a+b);  
    }  
}  
fn(15)();
```

匿名函数


匿名函数是指没有名称的函数。在 JavaScript 中，你可以创建匿名函数并将其分配给变量、传递给其他函数或立即执行。

以下是几种创建匿名函数的常见方式：


1. 将匿名函数分配给变量：

```
javascript  Copy  
  
const myFunction = function() {  
    // 函数体  
};  
...
```

2. 作为函数参数传递：

```
javascript  Copy  
  
function doSomething(callback) {  
    callback();  
}  
  
doSomething(function() {  
    // 匿名函数作为回调函数  
});  
...
```

3. 立即执行的匿名函数（Immediately Invoked Function Expression, IIFE）：

```
javascript  Copy  
  
(function() {  
    // 函数体  
})();  
...
```

函数一般都用来干一件事情，需用使用动词+名词，表示做一件事情 tellStory sayHello等

调试

- 调试步骤

浏览器中按F12-->sources-->找到需要调试的文件-->在程序的某一行设置断点

- 调试中的相关操作

Watch：监视，通过watch可以监视变量的值的变化，非常的常用。

F10：程序单步执行，让程序一行一行的执行，这个时候，观察watch中变量的值的变化。

F8：跳到下一个断点处，如果后面没有断点了，则程序执行结束。

do..while循环和while循环非常像，二者经常可以相互替代，但是do..while的特点是不管条件成不成立，都会执行一次。

break:立即跳出整个循环，即循环结束，开始执行循环后面的内容（直接跳到大括号）

continue:立即跳出当前循环，继续下一次循环（跳到i++的地方）

支持if ;else if ;else x?x:x 还有switch case break,default break（和C一样的）

C也是这样的

运算符的优先级

优先级从高到底

1. () 优先级最高
2. 一元运算符 ++ -- !
3. 算数运算符 先* / % 后 + -
4. 关系运算符 > >= < <=
5. 相等运算符 == != === !==
6. 逻辑运算符 先&& 后||
7. 赋值运算符

= += -= *= /= %=

Compare

```
> '55' == 55
```

```
< true
```

```
> '55' !== 55
```

```
< true
```

```
> '55' === 55
```

++ - 和C语言一样

转换

Obj.toString()

String()函数存在的意义：有些值没有toString()，这个时候可以使用String()。比如：undefined和null

xx.Number()

- parseInt() `var num1 = parseInt("12.3abc");` // 返回12，如果第一个字符是数字会解析知道遇到非数字结束
- `var num2 = parseInt("abc123");` // 返回NaN，如果第一个字符不是数字或者符号就返回NaN

- parseFloat() 把字符串转换成浮点数
- parseFloat()和parseInt非常相似，不同之处在与
- parseFloat会解析第一个. 遇到第二个.或者非数字结束
- 如果解析的内容里只有整数，解析成整数

- +, -0等运算 `var str = '500';`
- `console.log(+str);` // 取正 会自动转换
- `console.log(-str);` // 取负
- `console.log(str - 0);`

转换成布尔类型

- Boolean()

`console.log(str.length);`
`'11' + '我' + 1 -> 11我1`
 true为1，false为0

复杂数据类型

Object

如何使用谷歌浏览器，快速的查看数据类型？

字符串的颜色是黑色的，数值类型是蓝色的，布尔类型也是蓝色的，undefined和null是灰色的

type

Number、String、Boolean、Undefined、Null

浮点数

`var n = 5e-324;` // 科学计数法 5乘以10的-324次方

不要判断两个浮点数是否相等，有误差

最小值：Number.MIN_VALUE，这个值为： 5e-324

最大值: Number.MAX_VALUE, 这个值为:
1.7976931348623157e+308

无穷大: Infinity

无穷小: -Infinity

- NaN: not a number
 - NaN 与任何值都不相等, 包括他本身
- isNaN: is not a number

用typeof(xx)来看类型

“ ” “ ” 不区分, 大小写区分

字 面 量	含 义
<code>\n</code>	换行
<code>\t</code>	制表
<code>\b</code>	空格
<code>\r</code>	回车
<code>\f</code>	进纸
<code>\\</code>	斜杠
<code>\'</code>	单引号 ('), 在用单引号表示的字符串中使用。例如: <code>'He said, \'hey.\''</code>
<code>\"</code>	双引号 ("), 在用双引号表示的字符串中使用。例如: <code>"He said, \"hey.\""</code>
<code>\xnn</code>	以十六进制代码 <code>nn</code> 表示的一个字符 (其中 <code>n</code> 为 0 ~ F)。例如, <code>\x41</code> 表示 "A"
<code>\unnnn</code>	以十六进制代码 <code>nnnn</code> 表示的一个Unicode字符 (其中 <code>n</code> 为 0 ~ F)。例如, <code>\u03a3</code> 表示希腊字符 Σ

想法

直接看文档, 不看课, 看完文档做项目, 不懂的不多就算了, 反正真正做到也是有问题 = 目标去写小程序还有另一套东西

声明变量

- 规则 - 必须遵守的, 不遵守会报错

- 由字母、数字、下划线、\$符号组成，不能以数字开头
- 不能是关键字和保留字，例如：for、while。
- 区分大小写

- var age, name, sex;

age = 10;

name = 'zs';

JavaScript的书写位置

- 写在行内

```
<input type="button" value="按钮" onclick="alert('Hello World')" />
```

- 写在script标签中

```
<head>
<script>
  alert('Hello World!');
</script>
</head>
```

- 写在外部js文件中，在页面引入

```
<script src="main.js"></script>
```

外部的js不需要写script标签

•

BOM DOM

- 1 BOM (Browser Object Model) : BOM是浏览器对象模型的缩写，用于描述浏览器提供的对象和方法，通过这些对象和方法，可以与浏览器窗口进行交互并操作浏览器的各个部分。BOM提供了诸如window、

document、navigator等对象，可以用于操作浏览器窗口、访问和修改文档内容、获取浏览器信息等。

- 2 DOM (Document Object Model) : DOM是文档对象模型的缩写，用于描述网页文档的结构和内容的编程接口。DOM将文档表示为一棵树形结构，其中每个节点都代表文档中的一个元素、属性或文本。通过使用DOM，可以以编程方式访问和操作网页文档的内容、结构和样式。

JavaScript现在的意义(应用场景)

JavaScript 发展到现在几乎无所不能。

- 1 网页特效
- 2 服务端开发(Node.js)
- 3 命令行工具(Node.js)
- 4 桌面程序(Electron)
- 5 App(Cordova)
- 6 控制硬件-物联网(Ruff)
- 7 游戏开发(cocos2d-js)
- 8

TS全栈工程师能做的

我列出了以下一些TS目前可做的东西

以下也是一个TS全栈开发者应该可以做的

- RN开发移动端
- React开发SPA, 中后台等
- Nextjs,Remix开发SSR网站
- Taro开发跨平台小程序
- Electron开发桌面软件
- Fastify,Nestjs开发后端
- Yargs与周边一些工具可构建强大的CLI
- 还有, 微服务, 爬虫, 区块链等等,Node都是一把好手

如果TS工程师配合上Golang用于补足Node性能的不足, 那么就是一名真正的无懈可击的全栈工程师了

TS+React+Node.js/Nestjs