期中答疑 | name(){}与name: function() {}，两种写法有什么区别吗？ | 极客时间

winter 2019-03-30

你好，我是 winter。

随着专栏进度过半，我们专栏的评论区留言量也日渐上涨。除了大家的小作业和学习心得，我还看见很多同学们在学习过程中提出了不少问题。

这其实是一种很好的学习方式，通过问题，我们可以对这部分知识记得更为牢固。

所以，我鼓励你在阅读文章之外，多思考，多提问，把自己不懂的地方暴露出来，及时查缺补漏，这样可以更好地吸收知识。同时，你也可以通过回答别人的问题来检验自己对知识的掌握情况。

我们一起来看看，大家都提出什么问题。

1. 老师你好！我语义化标签用得很少，多数用到的是 header、footer、 nav 等语义化标签，想问老师 section 和 div 混合使用，会不会效果不好呢?

答：不会效果不好的，因为本来就是这么用的。遇到不确定的情况，请千万不要乱用标签，用 div 和 span 就好。

2. 我一直看见闭包这个词，但是一直也没有弄清楚它是什么东西，老师可以简单概括一下什么是闭包吗?

答：你可以这样理解，闭包其实就是函数，还是能访问外面变量的函数。

3.“事实上，JavaScript 中的“类”仅仅是运行时对象的一个私有属性，而 JavaScript 中是无法自定义类型的。”

- **文中说“类”是私有属性，可以具体表现是什么，不是很能理解具体含义?**

答：私有属性当然是你无法访问的属性了，但是具体表现的话，还是有的，那就是 Object.prorotype.toString.call(x) 的行为。

- **无法自定义类型？请问如下编码是属于什么操作，应该怎么理解这个“类”？**

```
function Person（） {}
var person = new Person（）；
□复制代码
```

答：这个代码是定义类的操作，这里注意一下，你千万不要把类和类型的概念混淆。

4. 请教老师在对象中`name(){}` 等同于`name: function() {}`，这两个写法有什么区别呢?

答：这两个写法在使用上基本没什么区别。只有一点区别，就是函数的 name 属性不一样。可以看下这段代码：

```
var o = {
   myfunc(){}
}
console.log(o.myfunc.name)
□复制代码
```

我们这里按照你的第一种方法定义了方法，然后输出它的 name 属性，我们看到 name 属性是"myfunc"。

值得一提的是，如果我们给你的第二种方法添加了名字，行为还是不一样，区别在于能否在函数内用名字递归，我们看看代码：

```
var o2 = {
    myfunc(){
        consoe.log(myfunc); //error
    }
}
var o1 = {
    myfunc: function myfunc(){
        consoe.log(myfunc); //function myfunc
    }
}
o1.myfunc();
o2.myfunc();
```

□复制代码

这段代码中，我们试着在用两种方式定义的方法中输出函数自身的名字变量，结果是不一样的。

不过现实中，我们几乎不会关心函数的 name 属性，所以不用太在意两种定义方式的区别。

5. 我对于 JavaScript 中 Number 安全整数有个疑问。

MDN 中是（-(2^53-1)~(2^53-1)），犀牛书中是（-2^53~2^53）感觉都有道理。

JavaScript 中采用 IEEE754 浮点数标准进行存储， 1 个符号位，11 位指数位， 52 位尾数位。

按照分析，不考虑符号位，尾数位取值 52 个 1 就是表示的最大值了，不会有精度损失，此时指数位代表数值是 52+1023=1075，此时即为 (-(2^53-1)~(2^53-1))。

但是 2^53 这个值，存储的时候尾数是 52 个 0， 指数位为 53+1023=1076，这个值也是刚好没有精度损失的，这时表示的就是（-2^53~2^53）。

用 Math.isSafeInteger() 判断安全数范围和 MDN 中描述一样。所以被问到这个的时候， 感觉两个都是有道理的吧！老师你说对吗?

答：你分析得非常好，我觉得我都没啥可补充的了。这个地方 JavaScript 标准写得也非常模糊，我简单瞄了一下，似乎是用实验的方式来给出的安全数范围。考虑到犀牛书的时效性肯定不如 MDN，应该是参考了某一版本旧引擎给出来的数据。

所以，这类行为我们还是以实测为准吧，我们不必纠结。

6. 老师您好，下面这个自己练习的例子希望您能帮解答：

```
console.log('sync1');


setTimeout(function () {
    console.log('setTimeout1')
}, 0);


var promise = new Promise(function (resolve, reject) {
    setTimeout(function () {
        console.log('setTimeoutPromise')
```

```
    }, 0);
    console.log('promise');
    resolve();
});



promise.then(() => {
    console.log('pro_then');
    setTimeout(() => {
        console.log('pro_timeout');
    }, 0)
})

setTimeout(function () {
    console.log('last_setTimeout')
}, 0);
console.log('sync2');
```

□复制代码

答：这个例子挺经典的，虽然我觉得这样设计面试题非常不合适，但是我们可以以它为例，学习一下分析异步的方法。

首先我们看第一遍同步执行，这是第一个宏任务。

第一个宏任务中，调用了三次 setTimeout（Promise 中的代码也是同步执行的），调用了一次 resolve，打印了三次。

所以它产生了三个宏任务，一个微任务，两次打印。

那么，首先显示的就是 sync1、promise 和 sync2。这时，setTimeout1，setTimeoutPromise，last_setTimeout 在宏任务队列中，pro_then 在微任务队列中。

接下来，因为微任务队列没空，第一个宏任务没有结束，继续执行微任务队列，所以 pro_then，被显示出来，然后又调用了一次 setTimeout，所以 pro_timeout 进入宏任务队列，成为第 5 个宏任务。

然后，没有微任务了，执行第二个宏任务，所以接下来顺次执行宏任务，显示 setTimeout1，setTimeoutPromise，last_setTimeout，pro_timeout。

最终显示顺序是这样的。

- **宏任务 1**
- **宏任务 5**
  - pro_timeout

7. 为什么 promise.then 中的 settimeout 是最后打印的？不用管是宏任务依次执行吗？

答：因为 then 是第一个宏任务中最后执行的微任务，所以它发起的宏任务是最后入队的，依次执行就是最后。

8. 怎么确定这个微任务属于一个宏任务呢，JavaScript 主线程跑下来，遇到 setTImeout 会放到异步队列宏任务中，那下面的遇到的 promise 怎么判断出它是属于这个宏任务呢？

答：resolve 在哪个宏任务中调用，对应的 then 里的微任务就属于哪个宏任务。宏任务没有从异步队列中取出，中间所碰到的所有微任务都属于这个宏任务。

9. 为什么要设计微任务（micro task），我知道这样 JavaScript 引擎可以自主地执行任务，但这样的好处是什么？提高性能吗？

答：不是，微任务是 JavaScript 引擎内部的一种机制，如果不设计微任务，那么 JavaScript 引擎中就完全没有异步了呀，所以必须要设计微任务。

10. 现在浏览器多数实现是从右往左匹配的，那么无法保证选择器在 DOM 树构建到当前节点时，已经可以准确判断当前节点是否被选中。现在浏览器又是怎么实现在生成 DOM 树，同时进行 CSS 属性计算？

答：其实现代浏览器已经为`:empty`、`:last`等伪元素写了很多例外了，不过你说的从右往左匹配，左边的要么是当前节点的父元素，要么是前置元素，所以是可以保证准确判断的呀。

11. 请问老师，页面资源的预加载是不是可以用 link 标签实现，还有其他的方式吗？

答：预加载的方法就多啦，还可以用 JavaScript 代码预加载，甚至用本地存储缓存。

12. 老师，我有一个疑问：“词法环境”和“词法作用域”这两个概念的区别是什么？希望你能帮我解惑。

答：词法环境是运行时概念，词法作用域是语言概念，就是说，作用域指的是变量生效的那段代码，而词法环境是指运行起来之后，你这段代码访问的存储变量的内存块。

13. 想问一个问题：import 进来的引用为什么可以获取到最新的值，是类似于 getter 的机制吗？

答：这个地方略微有些复杂，我们在运行时并没有讲 import 的运行时机制，这里涉及了一个叫做 ImportEntry Record 的机制，它比 getter 的实现更底层。

我想这个地方我们没有必要去深究模块的运行时机制，它很复杂而且并不是经常要用到。你如果想了解的话，可以查阅一下。

14. 请问老师，JavaScript 的 call stack size 是多少，这个 size 的单位是啥，是调用栈中函数的个数，还是一个存储单位，比如 MB 之类的。如果调用栈中就一个函数，这个函数的参数有 100 万个，浏览器端依然会溢出，看起来是存储单位，但是没得到验证。

答：这个似乎并没有什么特别规定，我知道 JSC 里面这个东西是可以用 C++ 代码来调整的，至于浏览器调用 JavaScript 引擎的时候会怎么做，还真不好说。

不过，从编码风格上建议，不要把这种事情用函数解决啦，真要干这样的事，数组可能都不合适了，请老老实实写 ArrayBuffer 吧。

15. 老师您好，我一直有一个困惑，浏览器的鼠标事件是怎么识别到的，是碰撞检测的吗？

答：这个问题很不错，我后面在浏览器 API 的事件部分会详细讲，可以先简单说一下，这里的检测方式是从外到内，逐级分配给子元素，所以我们的事件会有捕获过程。

16. 有个问题，如果我 javaScript 代码改变了 DOM 树元素的位置，需要启动重新排版（位置改变的元素只会影响其他部分元素的位置，甚至不影响其他元素的位置。），这时会导致这棵 DOM 树的所有元素都需要重新排版、绘制和渲染吗？

答：排版应该是会重新排的，但是如果有些元素的尺寸没有改变，那么它内部不需要重排，当然也就更不需要重新渲染了，但是绘制应该是要重绘的，目前来看，浏览器还没有那么智能。

17. 老师，我是 12 年左右踏进半只脚到前端领域的，后来考研就放弃了，觉得前端不够高深，和传统工程师来说觉得门槛低很多，甚至前期我都觉得自己不是个程序员。

直到研究生毕业，才又选择前端，这是三大框架风靡，我却有点迷惘，感觉和自己认知的前端不一样，直到现在工作了差不多两年，才悟出了点道道。

作为工程师，我始终觉得前端也应该熟练算法和数据结构、数据库这些所谓的后端知识，但是平时工作场景中用到又少，不知如何学习?

答：算法和数据结构可不是什么后端知识呀，是所有程序员的基本技能。

算法主要是靠大量练习提高，数据结构可以一个一个学习，不要指望工作中用到恰巧就学了，毕竟学习要教学费而工作是领工资的，哪里会有这样的好事呢，所以还是自己多多练习呀。

18. 重学前端是夯实前端基础，那前端进阶方向在哪里? 还是一定要修一门后端语言扩展服务端，希望老师可以指点迷津。

答：我觉得任何编程相关岗位的进阶方式都是做出某某东西，而不是学会某某东西。我会在专栏课程的第四模块会讲到一些进阶可能的方向，你可以关注一下。

19. 我主业是后端，工作中也会带着做前端，自认还是能完美还原设计师的设计。但是现在感觉很多时候提前端就是 vue 等，而我还是在用 jQuery，想请老师说说看，我是不是落伍了?

答：落伍的问题不是你用什么框架，而是你在做什么东西，学什么东西。

框架不是赶时髦，追潮流，每个框架都有解决的问题，我觉得你该焦虑的不是你用的框架为什么这么老，而是你该知道这些新框架要解决什么问题，以及这些问题为什么在你的工作中不存在。

最后，我们来看看我在 JavaScript 类型那一篇中给你留的实践问题。

如果我们不用原生的 Number 和 parseInt，用 JavaScript 代码实践 String 到 Number，该怎么做呢?

答：其实这个问题我在后台没看到特别满意的答案，好像大家都很喜欢偷懒啊。

我这里给你留个例子，处理十进制整数。

```
function atoi(a){

    let chars = a.split("").map(e => e.charCodeAt(0) - "0".charCodeAt(0));
```

```
    let n = 0;
    for(var char of chars) {
        n *= 10;
        n += char;
    }
    return n;
}
atoi("1001")
```

□复制代码

我比较期待大家有人能写出来带小数，甚至带科学计数法的代码，你可以尝试一下。

好了，今天的答疑环节就进行到这里，你也可以把自己想要解答的问题留言。

bd2star