

# 深入理解类数组以及多种方式实现类数组转真数组



jCodeLife

2020.08.12 11:35:22 字数 1,147 阅读 3

编辑文章

我们经常会听人提到类数组，也经常会遇到类数组转真数组的怎么转的问题？要理解类数组，那得说一下真数组

我们知道，js数组是用一个变量存储多个数据的一种特殊的数据结构，可以通过数组下标获取对应位置的数据，并且js提供了一系列的属性和方法来操作数组。

我理解的类数组其实就是类似数组的对象；本质是长得很像的两个东西 一个对象一个数组，因为它长得像数组，使用起来也挺像数组，所以大家后面习惯把它叫做类数组。



长得相似的行星

类数组有几个必要组成部分：

1. 属性要为索引（数字）属性；
2. 必须有length属性
3. 最好加上数组的push和splice方法

类数组并不陌生，函数中的arguments就是属于类数组，在DOM操作中，我们也经常会得到一个类数组（比如document.getElementsByClassName()）

那再来深入看看，到底里面都有什么？

先定义一个普通方法并执行，把打印 arguments看看

```
1 function fn(){
2   console.log(arguments)
3 }
4 fn('a','b','c')
```

结果如下：

```
▼ Arguments(3) ["a", "b", "c", callee: f, Symbol(Symbol.iterator): f]
  0: "a"
  1: "b"
  2: "c"
  callee: f fn()
  length: 3
  Symbol(Symbol.iterator): f values()
  __proto__: Object
```

这验证了我们上面说的两点：1.属性要为索引属性；2.必须有length属性

接下来继续试试，看类数组能不能像数组那样进行一些操作



jCodeLife

总资产 4 (约0.41元)

深入call,apply,bind到手动封装

阅读 45

深入理解类数组以及多种方式实现类数组转真数组

阅读 3

手动封装ES5数组新增方法-indexOf和lastIndexOf

阅读 2

## 推荐阅读

当“小姐”的那五年，我心中永远抹不去的痛。

阅读 25,488

女生真的不在乎男生有没有钱吗？

阅读 581

27岁女白领，公开夫妻私生活引热议：纵欲上瘾，正在榨干年轻人

阅读 37,703

【梦忆清裳】39东风（10）大结局（上）这是本分

阅读 5,314

做一个有上进心的人

阅读 1,269



结果如下

```
0: "通过下标修改已有数据"  
1: "b"  
2: "c"  
3: "通过下标添加新的数据"  
callee: f fn()  
length: 3  
Symbol(Symbol.iterator): f values()  
__proto__: Object
```

发现我们通过下标的方式修改、查看和添加类数组的数据，但是这种方式不会影响length。仔细一想，这更像对象的基本特性，就是将下标当他属性来存取。只是他看起来像数组一样。

接着看看能不能用数组的push方法

```
1 arguments.push("push");
```

答案跟你想的一样，是不可以的！

它会一个错误：Uncaught TypeError: arguments.push is not a function

其实你只要想清楚一个事就很容易理解了，

我问问大家，我们在数组中使用的一些列方法（如push、pop、shift、splice、join、concat...等等）都是定义在哪里的？

是不是定义在构造函数Array的原型上的，这样所有的实例数组都能使用！而这里arguments即没有添加对应的方法，也没有继承自Array.prototype，自然没有这写数组方法。

接下来我们来给arguments添加一个push方法

```
1 function fn() {  
2   arguments.push = Array.prototype.push;  
3   arguments.push('d');  
4   console.log(arguments)  
5 }  
6 fn('a', 'b', 'c')  
7
```

结果：添加push方法的类数组，push数据的时候会动态的增长length属性。

```
0: "a"  
1: "b"  
2: "c"  
3: "d"  
push: f push()  
callee: f fn()  
length: 4  
Symbol(Symbol.iterator): f values()  
__proto__: Object
```

除了arguments我们可以自定义一个类数组

```
1 //自定义数组  
2 var obj = {  
3   1: "a",  
4   2: 'b',  
5   3: 'c',  
6   length: 3,  
7   push: Array.prototype.push,  
8   splice: Array.prototype.splice  
9 }
```



数组打印出来时看上去跟数组一模一样。不过我目前的最新谷歌版本是打印出来还是类数组的样子：

```
▼ Object(3) [empty, "a", "b", 3: "c", push: f, splice: f]
  1: "a"
  2: "b"
  3: "c"
  length: 3
  ► push: f push()
  ► splice: f splice()
  ► __proto__: Object
```

不过这样不要紧，splice加不加都行。

这里插播一个笔试题吧：

```
1 var obj = {
2   2: 'a',
3   3: 'b',
4   length : 2,
5   push: Array.prototype.push,
6 }
7 obj.push('c');
8 obj.push('d');
9 console.log(obj);
```

答案可能出乎你得预料：

```
  2: "c"
  3: "d"
  length: 4
  ► push: f push()
  ► __proto__: Object
```

其实的理解push的内部原理，得回到我们之前手动封装push方法的时候

```
1 Array.prototype._push = function(){
2   for(var i=0;i<arguments.length;i++){
3     this[this.length] = arguments[i];
4     this.length++
5   }
6   return this.length;
7 }
```

里头核心操作 `this[this.length] = arguments[i]` 和 `this.length++`，就是跟数组的length位置添加数据，然后让length++。所以有了上面的答案。

最后，如果你希望将数组转成真数组，那继续往下看：

## 类数组转成真数组的几种方法

### 1. 用Array.prototype.slice.call(obj)方法转化为数组

在讲数组的时候有提到，通过slice方法可以拷贝得到新数组

```
1 var arr = [1,2,3,4,5];
2 var newArr = arr.slice()
3 console.log(newArr); // 截取整个数组 [1,2,3,4,5];
```

### 2. Array.from()方法



```
1 | var arr= [...obj];
```

以上是最简单也最常见的3种方式。还有几种利用call、apply、bind和map forEach几个方法配合实现，想了解可以点击下方第一个链接

参考资料：

[类数组对象转换为数组的六种方法—作者：霓裳依旧](#)

[数组 类数组—腾讯课堂渡一教育](#)

[JS数组及手动封装ES3的数组核心方法—作者：jCodeLife](#)



1人点赞 >



渡一领跑计划



"小礼物走一走，来简书关注我"

赞赏支持

还没有人赞赏，支持一下



jCodeLife 书山有路勤为径，学海无涯苦作舟

总资产4 (约0.41元) 共写了9.0W字 获得101个赞 共19个粉丝



写下你的评论...

全部评论 0

只看作者

关闭评论

按时间倒序

按时间正序

被以下专题收入，发现更多相似内容

投稿管理

+ 收入我的专题



渡一领跑计划笔记

写下你的评论...

评论 0

赞 1

...