目录

[已解决的问题 2](#_Toc376621014)

[1. 函数中定义与形参同名变量的问题。 2](#_Toc376621015)

[问题描述 2](#_Toc376621016)

[原因分析 2](#_Toc376621017)

[解决方案 2](#_Toc376621018)

[2. 对象字面量属性互相访问的问题。 2](#_Toc376621019)

[问题描述 2](#_Toc376621020)

[原因分析 3](#_Toc376621021)

[解决方案 3](#_Toc376621022)

[参考资料 4](#_Toc376621023)

[已解决但不知道原因的问题 5](#_Toc376621024)

[未解决的问题 6](#_Toc376621025)

[注意事项 7](#_Toc376621026)

[1. 如果想插入节点而又不想更改原节点，则可以使用clone来复制节点，然后再插入复制的节点（如果不复制的话，原节点会从老地方移走）。 7](#_Toc376621027)

## 已解决的问题

1. 函数中定义与形参同名变量的问题。

#### 问题描述

function test1(t){

vat t = t;

console.log(t);

}

test1(1); //1

function test2(t){

vat t = null;

t = t;

console.log(t);

}

test(1); //null

#### 原因分析

第一个函数test1中的“var t = t”令局部变量t等于形参t，而第二个函数test2中的“t = t”令局部变量t等于局部变量t(null)！

#### 解决方案

1. 对象字面量属性互相访问的问题。

#### 问题描述

var foo = {

a: 5,

b: 6,

c: this.a + this.b // undefined

};

#### 原因分析

此处的this并不是指向foo，而是指向当前上下文（假设foo为全局对象，则this指向window，而window.a和window.b为undefined！所以foo.c为undefined）。

#### 解决方案

1. You could do something like:

var foo = {

a: 5,

b: 6,

init: function() {

this.c = this.a + this.b;

return this;

}

}.init();

This would be some kind of one time initialization of the object.

Note that you are actually assigning the return value of init() to foo, therefore you have to return this.

1. Well, the only thing that I can tell you about are getters:

var foo = {

a: 5,

b: 6,

get c () {

return this.a + this.b;

}

};

foo.c; // 11

This is a syntactic extension introduced by the ECMAScript 5th Edition Specification, the syntax is supported by most modern browsers (including IE9).

1. 可以将c改为一个函数，如：

|  |  |
| --- | --- |
|  | var foo = {   a: 5,  b: 6,      c: function (){ return this.a +  this.b;}  } |

1. 在foo外定义a和b，如：

|  |  |
| --- | --- |
|  | var a =1,      b= 2;  var foo = {    a: a,    b: b,    c: a +  b  } |

更多参照“参考资料”

#### 参考资料

[Self-references in object literal declarations](http://stackoverflow.com/questions/4616202/self-references-in-object-literal-declarations)

[JavaScript对象字面量属性间的互相访问问题](http://bbs.csdn.net/topics/390526557)

## 已解决但不知道原因的问题

## 未解决的问题

## 注意事项

1. 如果想插入节点而又不想更改原节点，则可以使用clone来复制节点，然后再插入复制的节点（如果不复制的话，原节点会从老地方移走）。

网上关于jquery的clone的说明：

.clone()方法深度复制所有匹配的元素，包括所有匹配元素、匹配元素的下级元素、文字节点。 当和插入方法联合使用时，.clone()对于复制页面上的元素很方便。请看下面的HTML：

<div class="container">

<div class="hello">Hello</div>

<div class="goodbye">Goodbye</div>

</div>

像我们讨论[.append()](http://www.pcxyz.com/#p=append)一样，通常我们将页面上一个元素插入到DOM里另立个地方，它会被从老地方移走：

$('.hello').appendTo('.goodbye');

得到的DOM结构如下：

<div class="container">

<div class="goodbye">

Goodbye

<div class="hello">Hello</div>

</div>

</div>

但是我们如果需要的是复制而不是移除，我们可以像下面这样写代码：

$('.hello').clone().appendTo('.goodbye');

结果会是：

<div class="container">

<div class="hello">Hello</div>

<div class="goodbye">

Goodbye

<div class="hello">Hello</div>

</div>

</div>

可以指定clone(true)来复制节点的事件处理函数和子节点。

这里需要注意的是，如果复制的节点为p，则它的子节点（块元素）不会被复制（即使使用clone(true)），只会复制为行内元素的子节点。

这是因为p和pre只能包含纯文本或行内元素，不能包括p、h1、div等块元素！（详见《重构HTML》 p109）

如：

<p id=”a1”>

<span>

A

</span>

<p>

B

</p>

<p>

C

</p>

</p>

<div id=”b2”></div>

<script>

$(“#a1”).clone(true).appendTo($(“#b2”))

</script>

运行结果为：

<p id=”a1”>

<span>

A

</span>

<p>

B

</p>

<p>

C

</p>

</p>

<div id=”b2”>

<p>

<span>

A

</span>

</p>

</div>