JavaScript let 和 const

JavaScript this 关键字

面向对象语言中 this 表示当前对象的一个引用。

但在 JavaScript 中 this 不是固定不变的，它会随着执行环境的改变而改变。

- 在方法中，this 表示该方法所属的对象。
- 如果单独使用，this 表示全局对象。
- 在函数中，this 表示全局对象。
- 在函数中，在严格模式下，this 是未定义的(undefined)。
- 在事件中，this 表示接收事件的元素。
- 类似 call() 和 apply() 方法可以将 this 引用到任何对象。

实例

```
var person = {
firstName: "John",
lastName : "Doe",
id : 5566,
fullName : function() {
return this.firstName + " " + this.lastName;
}
};
```

尝试一下 »

方法中的 this

在对象方法中， this 指向调用它所在方法的对象。

在上面一个实例中，this 表示 person 对象。

fullName 方法所属的对象就是 person。

实例

```
fullName : function() {
return this.firstName + " " + this.lastName;
}
```

尝试一下 »

单独使用 this

单独使用 this，则它指向全局(Global)对象。

在浏览器中，window 就是该全局对象为 [**object Window**]:

实例

```
var x = this;
```

尝试一下 »

严格模式下，如果单独使用，this 也是指向全局(Global)对象。

实例

```
"use strict";
var x = this;
```

尝试一下 »

函数中使用 this（默认）

在函数中，函数的所属者默认绑定到 this 上。

在浏览器中，window 就是该全局对象为 [**object Window**]:

实例

```
function myFunction() {
return this;
}
```

尝试一下 »

函数中使用 this（严格模式）

严格模式下函数是没有绑定到 this 上，这时候 this 是 **undefined**。

实例

```
"use strict";
function myFunction() {
return this;
}
```

尝试一下 »

事件中的 this

在 HTML 事件句柄中，this 指向了接收事件的 HTML 元素：

实例

```
<button onclick="this.style.display='none'">
点我后我就消失了
</button>
```

尝试一下 »

对象方法中绑定

下面实例中，this 是 person 对象，person 对象是函数的所有者：

实例

```
var person = {
firstName : "John",
lastName : "Doe",
id : 5566,
myFunction : function() {
return this;
}
};
```

尝试一下 »

实例

```
var person = {
firstName: "John",
lastName : "Doe",
id : 5566,
fullName : function() {
return this.firstName + " " + this.lastName;
}
};
```

尝试一下 »

说明: **this.firstName** 表示 **this** (person) 对象的 **firstName** 属性。

显式函数绑定

在 JavaScript 中函数也是对象，对象则有方法，apply 和 call 就是函数对象的方法。这两个方法异常强大，他们允许切换函数执行的上下文环境（context），即 this 绑定的对象。

在下面实例中，当我们使用 person2 作为参数来调用 person1.fullName 方法时, **this** 将指向 person2, 即便它是 person1 的方法：

实例

```
var person1 = {
fullName: function() {
return this.firstName + " " + this.lastName;
}
}
var person2 = {
firstName:"John",
```

```
lastName: "Doe",
}
person1.fullName.call(person2); // 返回 "John Doe"
```

尝试一下 »

JavaScript let 和 const

点我分享笔记