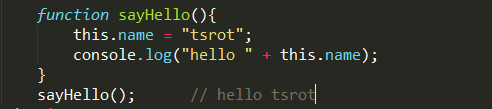
# JavaScript中this指向

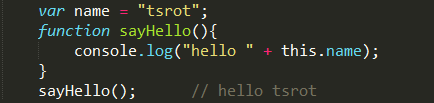
对于JavaScript初学者来说，this指针的指向问题一直是很混乱的问题。在不同的场景下，this会化身不同的对象。有一种观点认为，只有正确掌握了 JavaScript 中的 this 关键字，才算是迈入了 JavaScript 这门语言的门槛。在主流的面向对象的语言中（例如Java,C#等)，this 含义是明确且具体的，即指向当前对象。一般在编译期绑定。而 JavaScript 中this 在运行期进行绑定的，这是JavaScript 中this 关键字具备多重含义的本质原因。

随着函数使用场合的不同，this的值会发生变化。但是有一个总的原则，那就是**this指的是，调用函数的那个对象**。

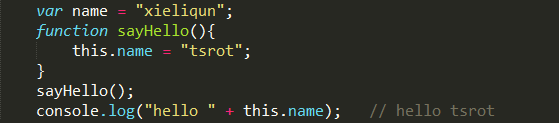
## 指向window的隐式指向



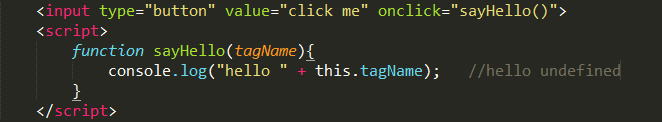
此时的变量name相当于window.name，因为调用sayHello函数的对象为window，相当于window.sayHello()。和下面两种情况是一样的：

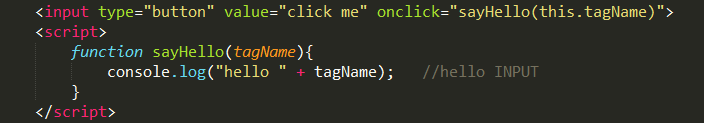


下面的name相当于在window下赋值了两次，谁后面执行就取谁。如果把var name = “xieliqun”放到sayHello（）后面，此时输出的就是 hello xieliqun。



注：当函数在DOM上调用时，未传入this，此时，this也指向window。当传入this时，this指向当前DOM input，例如下面情况：

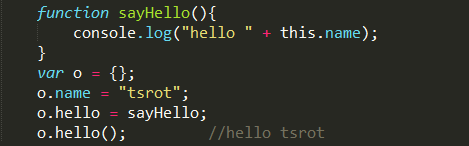




## 指向当前对象的显式调用

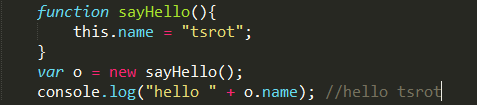
### （1）、作为对象方法的调用

函数还可以作为某个对象的方法调用，这时this就指这个上级对象。



### （2）、作为构造函数调用

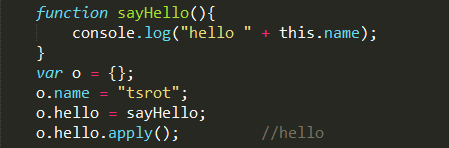
所谓构造函数，就是通过这个函数生成一个新对象（object）。这时，this就指这个新对象。



## 指向当前对象的隐式调用

### call、apply调用

call（）、apply()都是函数对象的一个方法，它们的作用是改变函数的调用对象，它们的第一个参数就表示改变后的调用这个函数的对象。因此，this指的就是这第一个参数。



apply()的参数为空时，默认调用全局对象。因此，这时的运行结果为0，证明this指的是全局对象。

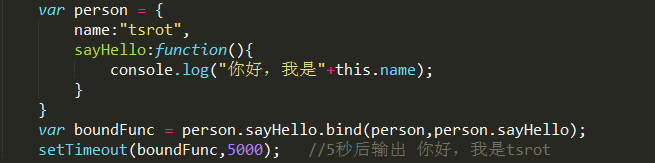
如果把最后一行代码修改为



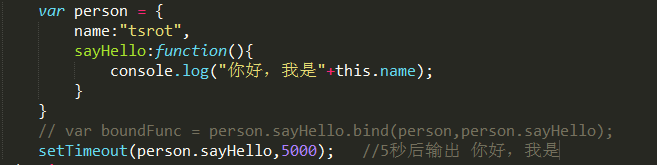
此时this就指向对象o了。

### 原生Function方法bind（）调用

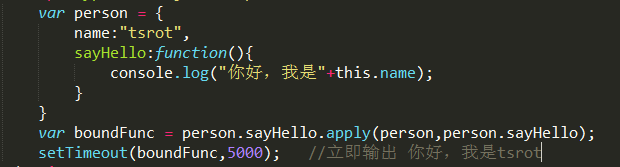
很多人不知道原生js也有bind（）方法，一直以为bind（）的方法只有jQuery有，我也是最近看别人博客知道的。其实原生的bind和jQuery的bind是不同的，原生的bind相当于apply和call。



下图代码中person.sayHello,相当于在window.person.sayHello，所以this指向window。

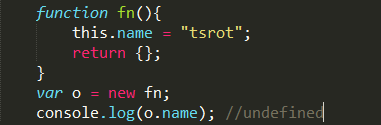


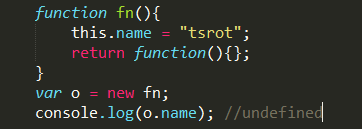
用apply和call调用时，函数将立即执行



## 当this在构造函数有return时

如果返回值是一个对象，那么this指向的就是那个返回的对象，如果返回值不是一个对象那么this还是指向函数的实例。





当return null和undefined时

