## 变量与常量

**变量**

在 JavaScript 中，我们可以使用*变量*来存放一个值，这个值随时可以被改变。变量的定义像下面这样：

let a;  
a = 10;

我们使用关键字 let 来表示一个变量声明的开始。let a 会产生一个名为 a 的变量，我们将这一步叫做**变量声明**，随后的 a = 10; 会把 10 这个值放进变量 a 中，这个过程叫**赋值**。

变量声明与赋值是两个过程，一个变量只能被声明一次，但是可以被多次赋值。正如它的名称所示，它的值是可以变的。第一次的声明与赋值可以被写到一行语句中：

let a = 10;

我们可以像这样改变它的值：

a = 100;

a = "Hi";

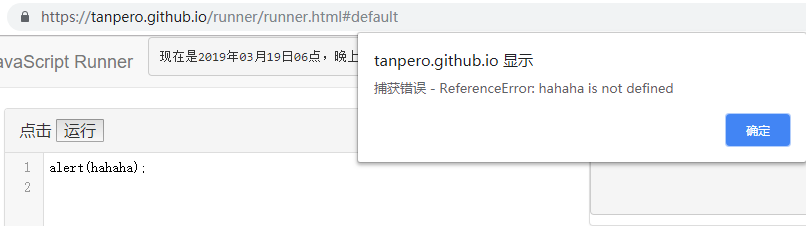
a = 12 \* 3;

为变量赋值之后，我们可以像通常的表达式那样使用它：

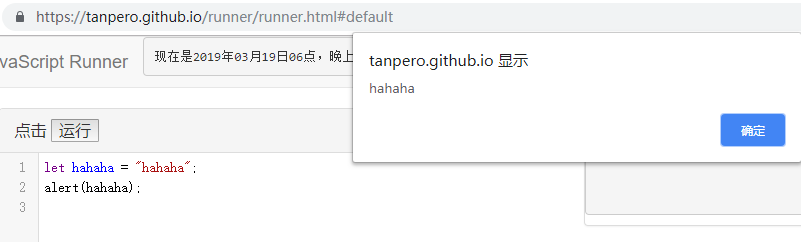
let a = 10;  
alert(a);  
a = a + 1;  
alert(a);  
alert(a + 2);

显示的结果分别为 10,11 和 13。

假如我们使用了一个从未出现的变量，会产生一个错误：



如果我们已经给 hahaha 赋值，那么可以使用它，不会产生错误，但是我们非常不推荐这种做法，使用变量前一定要记得声明。



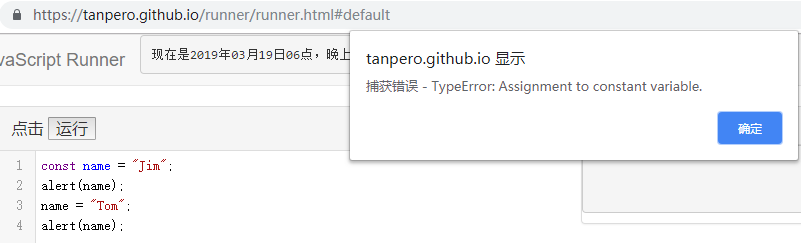
你或许曾在别的地方见过另一种声明变量的方式：

var a = 10;

var 关键字是 JavaScript 曾经使用的变量声明方式，但是这种方式存在一些缺陷，在2015年发布的新标准中被 let 所取代，一些细节我们会在后面谈到。我们理解它的含义即可，不建议使用它。

**常量**

顾名思义，*常量*与变量类似，但是它的值是不可改变的。也就是说，只能赋值一次，之后如果尝试改变它的值，会产生一个错误。

第一次调

用 alert 会正常显示 name 的值 "Jim" 。随后的赋值语句会带来如图所示的错误，并中断代码的执行，因此最后一行语句不会被执行。

常量的使用场景一般是强制某些不应改变的值**不可改变**，可以避免失误造成原有值被覆盖的情况。为了带来明显的视觉效果以与通常的变量区分，我们一般会使用大写字母来书写常量名称，并使用下划线“\_”来连接常量名中包含的单词。

const MY\_BIRTHDAY = "2005-2-6";  
alert(MY\_BIRTHDAY);

**标识符**

JavaScript 中的变量名或常量名必须是合法的*标识符*。一个合法的标识符应当以以下字符开头：

A B C D E F G H I J K L M N O P Q R S T U V W X Y Z

a b c d e f g h i j k l m n o p q r s t u v w x v z $ \_

即二十六个英文大小写字母以及美元符号和下划线。

而其后跟随的字符可以是：

A B C D E F G H I J K L M N O P Q R S T U V W X Y Z

a b c d e f g h i j k l m n o p q r s t u v w x v z $ \_

0 1 2 3 4 5 6 7 8 9

即二十六个英文大小写字母、美元符号、下划线和十个阿拉伯数字。

除此之外的字符一般是非法的，使用非法字符会造成不可预知的错误。

一个合法的标识符不能与已有的保留字发生冲突。保留字是一些单词，已经语言内定义以起到特殊目的，或者可能在将来被使用，因此被语言保留。如果你使用它们作为标识符，会发生语法错误。

**JavaScript 保留字**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  |  |  |  |  |
| abstract | arguments | boolean | break | byte |
| case | catch | char | class | const |
| continue | debugger | default | delete | do |
| double | else | enum | eval | export |
| extends | false | final | finally | float |
| for | function | goto | if | implements |
| import | in | instanceof | int | interface |
| let | long | native | new | null |
| package | private | protected | public | return |
| short | static | super | switch | synchronized |
| this | throw | throws | transient | true |
| try | typeof | var | void | volatile |
| while | with | yield |  |  |

除此之外，有一些 JavaScript 中预定义的函数、常量等也不应当被用作标识符，包括 Object Ininity undefined isNaN 等。