

字面量

掌握常见的字面量写法

JavaScript 书写语法

掌握 JavaScript 的基本语法

- 概念
- 数字字面量
- 字符串字面量

字面量

- 字面量 (literal) 是用于表达一个固定值的表示法, 又叫常量。
- 通俗的理解, 字面就是所见即所得, js 程序执行到代码中的字面量, 会立即知道它是什么类型的数据, 值是多少。
- 可以用于表示固定值, 比如: 数字、字符串、undefined、布尔类型的字面值等。

- 数值字面量: 8, 9, 10
- 字符串字面量: "大前端"
- 布尔字面量: true, false

JavaScript 书写语法

掌握 JavaScript 的基本语法

- 概念
- **数字字面量**
- 字符串字面量

数字字面量

- 这里的数字就是数学意义上的数字。
- 数字字面量区分：整数字面量、浮点数字面量（小数）、特殊值。
- 书写时直接书写字面量，不需要添加任何辅助符号。

整数

- 整数字面量写法区分进制。
- 整数可以被表示成十进制（基数为10）、八进制（基数为8）以及十六进制（基数为16）。
- **十进制**是最基本的数值字面量格式，可以直接在代码中输入。
- **八进制**字面值必须带前导0、00、0o。八进制整数只能包括数字0-7。
- **十六进制**的前缀是0x或0X。后面可以包含数字（0-9）和字母a~f或A~F。
- 在进行算术计算时或者参与程序，**所有八进制和十六进制的数字都会被转换成十进制。**

整数的进制

- 十进制：逢十进一，每个位数只能是0-9之间的数字。
- 八进制：逢八进一，每个位数上只能是0-7之间的数字，而且必须添加前缀，0、00、0o。
- 十六进制：逢十六进一，每个位数上必须是0-9、a-f之间的符号，必须写前缀，0x、0X开头。

整数

- 特殊的：八进制中，如果以0开头，每个位数上有超过0-7之间的数字出现，也就是8/9，强制忽视前面的0，直接将后面数字当做十进制。

浮点数

- 就是数学概念中的小数。
- 包含：整数、小数点、小数部分。
- 浮点数不区分进制，所有的浮点数都是十进制下的数字。
- 注意：如果浮点数是大于0 小于1的，可以省略小数点前面的0不写。

浮点数的精度问题

- 浮点数值的最高精度是 17 位小数，但在进行算术计算时其精确度远远不如整数
- 例如：0.1 + 0.2; 结果不是 0.3，而是：0.30000000000000004

Infinity 无穷

- Infinity：无穷的意思。
- 由于计算机计算能力有限，如果高于最大计算值直接显示为正无穷 Infinity，如果低于最小计算值直接显示为 -Infinity。
- Infinity 本身就是一个数字。

最小值：Number.MIN_VALUE，这个值为： 5e-324

最大值：Number.MAX_VALUE，这个值为： 1.7976931348623157e+308

无穷大：Infinity

无穷小：-Infinity

NaN

- NaN: not a number表示不是一个正常的数, 但是还是一个 Number 类型的数字。这个数字没办法用前面的表示方法表示。
- NaN 与任何值都不相等, 包括他本身。
- isNaN(): 判断一个数据是不是一个NaN。

JavaScript 书写语法

掌握 JavaScript 的基本语法

- 概念
- 数字字面量
- **字符串字面量**

字符串字面量

- 字符串是由任意个数的有序或无序的字符组成的串，类似人类的语言，在 JS 中有自己特殊的写法。
- 组成：字母、汉字、特殊符号、空白等。
- 字符串字面量写法：是用一对单引号（' '）或双引号（" "）及引号内的字符构成，引号中间的字符可以有任意多个，也可以是没有字符的空字符串。
- 注意：字符串中如果字符包含了双引号，则其外部应该由单引号标示，反之相同。

转义符号 \

- 字符串中，有一些特殊功能的字符不能直接书写，还有一些特殊效果不能直接书写。
- 这时，可以使用转义符 \ 对这些字符进行转义。
- 在字符串中可以使用转义符 \ 加普通字母，替代一些特殊字符

\n 换行

\t Tab制表

- 字符串中可以使用转义符 \ 将特殊功能字符变为普通字符。

\' 单引号

\" 双引号

\\ 反斜杠

变量

掌握变量的用法

什么是变量

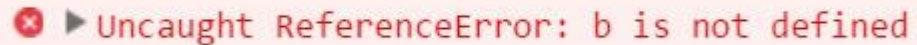
- 变量 (variables) 是计算机内存中存储数据的标识符, 根据变量名称可以获取到内存中存储的数据。
- 变量相当于一个容器, 内部可以存储任意类型的数据, 使用变量时, 用的是内部存储的数据。

为什么要使用变量

- 使用变量可以方便的获取或者修改内存中的数据。

变量声明

- 变量声明又叫做定义变量、创建变量。
- 变量在使用前，必须先有定义，如果没有定义，会出现引用错误。
- 定义方法：使用一个 `var` 的关键字进行定义，后面必须加一个空格，空格后面自定义变量名。

A screenshot of a JavaScript error message. It features a red 'x' icon in a circle on the left, followed by a right-pointing triangle. The text 'Uncaught ReferenceError: b is not defined' is displayed in a red monospace font on a light pink background.

Uncaught ReferenceError: b is not defined

变量的命名规则和规范

- 规则 - 必须遵守的，不遵守会报错

由字母、数字、下划线、\$ 符号组成，不能以数字开头。

字母区分大小写，A 和 a 表示不同。

不能是关键字和保留字，关键字指的是js中有特殊功能的小词语，比如 var、for 等；

保留字指的是现在没有特殊功能，但是将来新语法中有可能作为关键字使用。

- 规范 - 建议遵守的，不遵守不会报错

变量名必须有意义

遵守驼峰命名法。多个单词组合而成的，第一个单词首字母小写，后面单词的首字母需要大写。

例如：userName、userPassword

变量赋值

- 变量定义之后，初始时没有进行赋值，内部有一个默认存储的值叫 undefined（未定义），表示变量内部未赋值，可以存储数据了。
- 变量赋值的方式：通过等号 = 赋值，等号右边的值赋值给左边的变量。
- 注意：书写时，等号 = 两侧习惯书写一个空格。

```
// 变量定义  
var a;  
// 变量赋值  
a = 3;  
// 变量调用  
console.log(a);
```

变量赋值的几种情况

- 变量赋值时：内部可以存储任意类型的数据，甚至是一个变量。赋值过程中，等号右侧的变量使用的是存储的数据。
- 注意：变量参与赋值过程时，**等号左变右不变**。等号左侧会被赋值，将来值发生变化，等号右侧的变量使用内部的值参与运算，自身不会发生变化。

```
var a;  
a = 3;  
var b;  
b = a + 3;
```

变量赋值的几种情况

- 变量的赋初值过程可以与声明过程写在一起。

```
// 变量定义同时赋初值  
var c = "hello";
```


变量赋值的几种情况

- 变量内部的值，可以通过多次赋值的方法，进行更改。
- 变量一次定义，可以多次等号赋值。

```
var c = "hello";  
c = 3;  
c = c + 2;  
console.log(c);
```

变量赋值的几种情况

- 一个关键字 var 可以同时定义多个变量，并且都赋初值。多个变量之间用逗号进行分隔，最后一个变量后面使用分号进行结尾。

```
var a = 1,  
    b = 2,  
    c = 3;
```

数据类型

理解数据类型的含义和转换方法

JavaScript 中的值，无论是字面量还是变量，都有明确的类型。

数据类型

理解数据类型的含义和转换方法

- 简单数据类型
- 检测数据类型
- 数据类型转换

简单数据类型

- Number 数字类型
- String 字符串类型
- undefined undefined 类型
- Boolean 布尔类型
- null null 类型

另外，还有复杂数据类型

- Object 对象类型，后期课程详细介绍

Number 类型

- 数字类型，不区分整数、浮点数、特殊值，都是 Number 类型。

String 类型

- 字符串类型，所有的字符串都是 String 类型。

Boolean 类型

- Boolean 字面量： 只有 true 和 false 两个字面量的值，必须是小写字母。
- 计算机内部存储： true为1， false为0

Undefined 类型

- undefined 本身是一个数据，表示未定义。
- 变量只声明的时候值默认是 undefined。

Null 类型

- null 本身是一个数据。
- 从逻辑角度，null 值表示一个空对象指针。
- 如果定义的变量准备在将来用于保存对象，最好将该变量初始化为 null。

数据类型

理解数据类型的含义和转换方法

- 简单数据类型
- **检测数据类型**
- 数据类型转换

检测数据类型

- 使用 typeof 的方法进行数据检测。
- 检测方式：在 typeof 后面加小括号 () 执行，将要检测的数据放在小括号内部。

```
console.log(typeof(1));  
console.log(typeof("汉字内容"));  
console.log(typeof(undefined));  
console.log(typeof(true));  
console.log(typeof(false));  
console.log(typeof(null));
```

检测数据类型

- 也可以将 `typeof` 作为关键字，后面加空格，空格后添加数据的方式，检测数据。

```
console.log(typeof 67);
```

变量的数据类型

- JavaScript 语言是一门动态类型的语言，变量并没有一个单独的数据类型，而是会随着内部存储数据的变化，数据类型也会发生变化。
- 变量的数据类型，与内部存储数据有关。
- 将来使用变量时，需要知道内部存储的数据是什么类型，避免程序出错。

```
// 定义变量
var a = 1;
console.log(typeof(a));
// 变量赋值发生变化
a = "haha";
console.log(typeof(a));
```

提示

如何使用谷歌浏览器控制台，快速的查看数据类型？

- 字符串的颜色是黑色的，数值类型是蓝色的，布尔类型也是蓝色的，undefined和null是灰色的

数据类型

理解数据类型的含义和转换方法

- 简单数据类型
- 检测数据类型
- **数据类型转换**

不同的数据类型之间，可以进行互相转换。

转换成字符串类型

- 数据.toString() 方法
- String(数据) 方法，有些值没有toString()，这个时候可以使用String()。比如：undefined和null
- + 号拼接字符串方式

num + ""，当 + 两边一个操作符是字符串类型，一个操作符是其它类型的时候，会先把其它类型转换成字符串再进行字符串拼接，返回字符串。

+ 号的特殊性

1. 两边只要有一个是字符串，那么 + 就是字符串拼接功能
2. 两边如果都是数字，那么就是算术功能。

转换成数值类型

- Number(数据) 方法

转型函数Number()可以用于任何数据类型，将其他数据类型转为数字。

字符串：纯数字字符串转为对应数字，空字符串和空白字符串转为 0，非空非纯数字字符串转为 NaN。

布尔值：true 转为 1，false 转为 0。

undefined：转为 NaN。

null：转为 0。

转换成数值类型

- parseInt() 方法：字符串转整数方法

作用：第一，对浮点数进行取整操作；第二，将字符串转为整数数字。

①对数字取整功能，直接舍弃小数部分，只保留整数。

②将字符串转为整数数字，也包含取整功能。

字符串中，必须是纯数字字符串或者数字字符开头的字符串，才能转换为正常数字，且只取整数部分，如果不是数字打头的字符串，会转换为 NaN。

转换成数值类型

- parseFloat() 方法：字符串转浮点数方法

作用：将字符串转为浮点数数字。

要求：满足浮点数数字字符必须在字符串开始，如果不在开始返回值都是NaN。

应用

- 工作中，利用变量接收 `prompt()` 语句返回的用户输入的数据，检测数据类型后，得到的是字符串类型。
- 如果想要获取的是数字类型的数据，则需要先将得到的字符串转数字，避免出现字符串参与数学加法运算等。

```
var num = parseInt(prompt("请输入你今年的年龄"));  
var age = num + 50;
```


转换成布尔类型

- Boolean(数据) 方法

转型函数 Boolean() 可以用于任何数据类型，将其他数据类型转为布尔类型的值。

转为 false：NaN、0、""空字符串、null、undefined

转为 true：非0 非NaN数字、非空字符串

拉勾教育

— 互联网人实战大学 —



下载「拉勾教育App」
获取更多内容