数据类型

目录 Contents

1. 数据类型简介

1.1 为什么需要数据类型

在计算机中，不同的数据所需占用的存储空间是不同的，为了便于把数据分成所需内存大小不同的数据，充分利用存储空间，于是定义了不同的数据类型。

简单来说，数据类型就是数据的类别型号。比如姓名“张三”，年龄18，这些数据的类型是不一样的。

1. 数据类型简介

1.2 变量的数据类型

变量是用来存储值的所在处，它们有名字和数据类型。变量的数据类型决定了如何将代表这些值的位存储到计算机的内存中。**JavaScript 是一种弱类型或者说动态语言。**这意味着不用提前声明变量的类型，在程序运行过程中，类型会被自动确定。

```
var age = 10;          // 这是一个数字型
var areYouOk = '是的';  // 这是一个字符串
```

在代码运行时，变量的数据类型是由 JS引擎 根据 = 右边变量值的数据类型来判断 的，运行完毕之后， 变量就确定了数据类型。

JavaScript 拥有动态类型，同时也意味着相同的变量可用作不同的类型：

```
var x = 6;            // x 为数字
var x = "Bill";       // x 为字符串
```

1. 数据类型简介

1.3 数据类型的分类

JS 把数据类型分为两类：

- 简单数据类型 （Number,String,Boolean,Undefined,Null）
- 复杂数据类型 （object）

目录 Contents

2. 简单数据类型

2.1 简单数据类型（基本数据类型）

JavaScript 中的简单数据类型及其说明如下：

简单数据类型	说明	默认值
Number	数字型，包含 整型值和浮点型值，如 21、0.21	0
Boolean	布尔值类型，如 true 、false，等价于 1 和 0	false
String	字符串类型，如 "张三" 注意咱们js 里面，字符串都带引号	""
Undefined	var a; 声明了变量 a 但是没有给值，此时 a = undefined	undefined
Null	var a = null; 声明了变量 a 为空值	null

2. 简单数据类型

2.2 数字型 Number

JavaScript 数字类型既可以用来保存整数值，也可以保存小数(浮点数）。

```
var age = 21;        // 整数
var Age = 21.3747;   // 小数
```

2. 简单数据类型

2.2 数字型 Number

1. 数字型进制

最常见的进制有二进制、八进制、十进制、十六进制。

```
// 1.八进制数字序列范围：0~7
var num1 = 07;   // 对应十进制的7
var num2 = 019;  // 对应十进制的19
var num3 = 08;   // 对应十进制的8
// 2.十六进制数字序列范围：0~9以及A~F
var num = 0xA;
```

现阶段我们只需要记住，在JS中八进制前面加0，十六进制前面加 0x

2. 简单数据类型

2.2 数字型 Number

2. 数字型范围

JavaScript中数值的最大和最小值

```
alert(Number.MAX_VALUE); // 1.7976931348623157e+308
alert(Number.MIN_VALUE); // 5e-324
```

- 最大值：Number.MAX_VALUE，这个值为： 1.7976931348623157e+308
- 最小值：Number.MIN_VALUE，这个值为：5e-32

2. 简单数据类型

2.2 数字型 Number

3. 数字型三个特殊值

```
alert(Infinity);  // Infinity
alert(-Infinity); // -Infinity
alert(NaN);       // NaN
```

- Infinity，代表无穷大，大于任何数值
- -Infinity，代表无穷小，小于任何数值
- NaN，Not a number，代表一个非数值

2.2 数字型 Number

4. isNaN()

用来判断一个变量是否为非数字的类型，返回 true 或者 false


```
var usrAge = 21;
var isOk = isNaN(userAge);
console.log(isNum);             // false , 21 不是一个非数字
var usrName = "andy";
console.log(isNaN(userName));   // true , "andy"是一个非数字
```

2. 简单数据类型

2.3 字符串型 String

字符串型可以是引号中的任意文本，其语法为 双引号 "" 和 单引号''

```
var strMsg = "我爱北京天安门~";   // 使用双引号表示字符串
var strMsg2 = '我爱吃猪蹄~';     // 使用单引号表示字符串
// 常见错误
var strMsg3 = 我爱大肘子;         // 报错，没使用引号，会被认为是js代码，但js没有这些语法
```

因为 HTML 标签里面的属性使用的是双引号，JS 这里我们更推荐使用单引号。

2. 简单数据类型

2.3 字符串型 String

1. 字符串引号嵌套

JS 可以用单引号嵌套双引号，或者用双引号嵌套单引号 (**外双内单，外单内双**)

```
var strMsg = '我是"高帅富"程序猿';    // 可以用''包含""
var strMsg2 = "我是'高帅富'程序猿";   // 也可以用"" 包含''
// 常见错误
var badQuotes = 'What on earth?"; // 报错，不能 单双引号搭配
```

2.3 字符串型 String

2. 字符串转义符

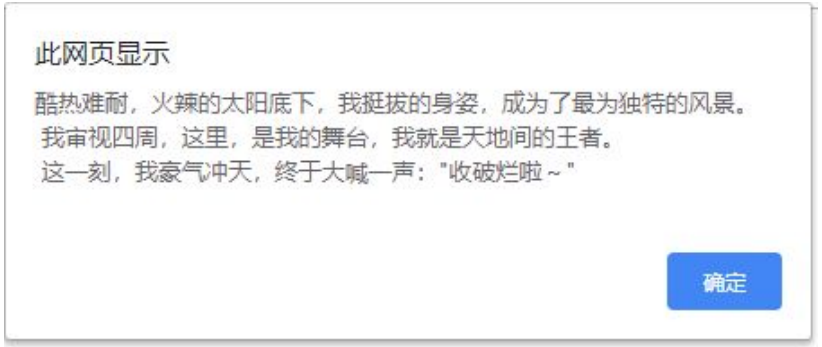
类似HTML里面的特殊字符，字符串中也有特殊字符，我们称之为转义符。

转义符都是 \ 开头的，常用的转义符及其说明如下：

转义符	解释说明
\n	换行符，n 是 newline 的意思
\ \	斜杠 \
\'	' 单引号
\"	" 双引号
\t	tab 缩进
\b	空格 ， b 是 blank 的意思

2. 简单数据类型

案例：弹出网页警示框



酷热难耐，火辣的太阳底下，我挺拔的身姿，成为了最为独特的风景。我审视四周，这里，是我的舞台，我就是天地间的王者。这一刻，我豪气冲天，终于大喊一声："收破烂啦～"

2.3 字符串型 String

3. 字符串长度

字符串是由若干字符组成的，这些字符的数量就是字符串的长度。通过字符串的 length 属性可以获取整个字符串的长度。

```
var strMsg = "我是帅气多金的程序猿！";
alert(strMsg.length); // 显示 11
```

2.3 字符串型 String

4. 字符串拼接

- 多个字符串之间可以使用 + 进行拼接，其拼接方式为 字符串 + 任何类型 = 拼接之后的新字符串
- 拼接前会把与字符串相加的任何类型转成字符串，再拼接成一个新的字符串

```
//1.1 字符串 "相加"
alert('hello' + ' ' + 'world'); // hello world
//1.2 数值字符串 "相加"
alert('100' + '100'); // 100100
//1.3 数值字符串 + 数值
alert('11' + 12);      // 1112
```

+ 号总结口诀：数值相加 ， 字符相连

2.4 字符串型 String

5. 字符串拼接加强

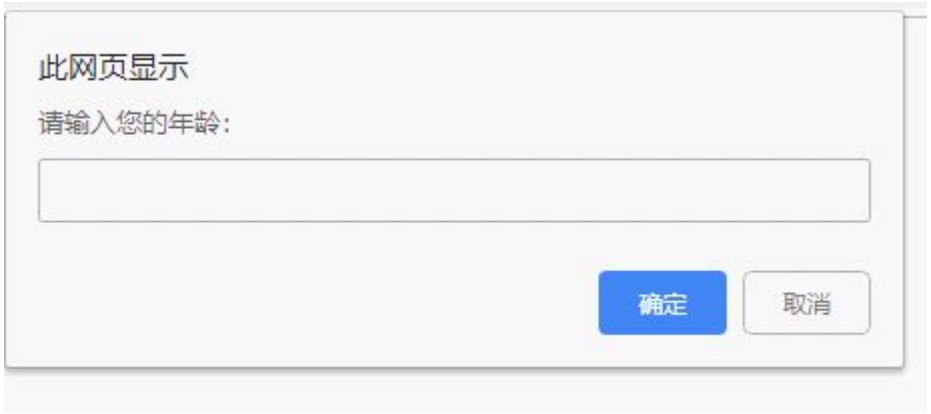
```
console.log('pink老师' + 18);            // 只要有字符就会相连
var age = 18;
// console.log('pink老师age岁啦');        // 这样不行哦
console.log('pink老师' + age);           // pink老师18
console.log('pink老师' + age + '岁啦');  // pink老师18岁啦
```

- 我们经常会将字符串和变量来拼接，因为变量可以很方便地修改里面的值
- 变量是不能添加引号的，因为加引号的变量会变成字符串
- 如果变量两侧都有字符串拼接，口诀“引引加加”，删掉数字，变量写加中间

2. 简单数据类型

案例：显示年龄

弹出一个输入框，需要用户输入年龄，之后弹出一个警示框显示“您今年 xx 岁啦”（xx 表示刚才输入的年龄）



2. 简单数据类型

案例分析

这是利用 JS 编写的一个非常简单的交互效果程序。

你喜欢我吗

2. 简单数据类型

案例分析

交互编程的三个基本要素：

1. 你喜欢我吗？→ 这是 用户输入
2. 女孩想了想 → 这是 程序内部处理
3. 最后给了你一巴掌 → 这是 输出结果

那么在程序中要如何实现呢？

① 弹出一个输入框（prompt），让用户输入年龄（用户输入）
② 把用户输入的值用变量保存起来,把刚才输入的年龄与所要输出的字符串拼接 （程序内部处理）
③ 使用alert语句弹出警示框（输出结果）

2. 简单数据类型

案例代码

```
// 弹出一个输入框（prompt），让用户输入年龄（用户输入）
// 把用户输入的值用变量保存起来,把刚才输入的年龄与所要输出的字符串拼接 （程序内部处理）
// 使用alert语句弹出警示框（输出结果）

var age = prompt('请输入您的年龄');

var str = '您今年已经' + age + '岁了';

alert(str);
```

2.5 布尔型 Boolean

布尔类型有两个值：true 和 false，其中 true 表示真（对），而 false 表示假（错）。

布尔型和数字型相加的时候， true 的值为 1，false 的值为 0。

```
console.log(true + 1);  // 2
console.log(false + 1); // 1
```

2.6 Undefined 和 Null

一个声明后没有被赋值的变量会有一个默认值 undefined（如果进行相连或者相加时，注意结果）

```
var variable;
console.log(variable);           // undefined
console.log('你好' + variable);  // 你好undefined
console.log(11 + variable);     // NaN
console.log(true + variable);   //  NaN
```

一个声明变量给 null 值，里面存的值为空（学习对象时，我们继续研究null）

```
var vari = null;
console.log('你好' + vari);  // 你好null
console.log(11 + vari);     // 11
console.log(true + vari);   //  1
```

目录 Contents

3. 获取变量数据类型

3.1 获取检测变量的数据类型

typeof 可用来获取检测变量的数据类型

```
var num = 18;
console.log(typeof num) // 结果 number
```

不同类型的返回值

类型	例	结果
String	typeof "小白"	"string"
Number	typeof 18	"number"
Boolean	typeof true	"boolean"
Undefined	typeof undefined	"undefined"
Null	typeof null	"object"

3. 获取变量数据类型

3.2 字面量

字面量是在源代码中一个固定值的表示法，通俗来说，就是字面量表示如何表达这个值。

- 数字字面量：8, 9, 10
- 字符串字面量：'黑马程序员', "大前端"
- 布尔字面量：true，false

目录 Contents

4. 数据类型转换

4.1 什么是数据类型转换

使用表单、prompt 获取过来的数据默认是字符串类型的，此时就不能直接简单的进行加法运算，而需要转换变量的数据类型。通俗来说，就是把一种数据类型的变量转换成另外一种数据类型。

我们通常会实现3种方式的转换：

- 转换为字符串类型
- 转换为数字型
- 转换为布尔型

4. 数据类型转换

4.2 转换为字符串

方式	说明	案例
toString()	转成字符串	var num= 1; alert(num.toString());
String() 强制转换	转成字符串	var num = 1; alert(String(num));
加号拼接字符串	和字符串拼接的结果都是字符串	var num = 1; alert(num+ "我是字符串");

- toString() 和 String() 使用方式不一样。
- 三种转换方式，我们更喜欢用第三种加号拼接字符串转换方式， 这一种方式也称之为隐式转换。

4. 数据类型转换

4.3 转换为数字型（重点）

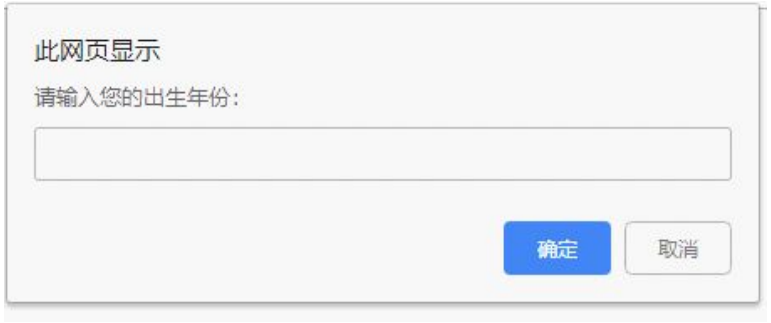
方式	说明	案例
parseInt(string) 函数	将string类型转成整数数值型	parseInt('78')
parseFloat(string) 函数	将string类型转成浮点数数值型	parseFloat('78.21')
Number() 强制转换函数	将string类型转换为数值型	Number('12')
js 隐式转换(- * /)	利用算术运算隐式转换为数值型	'12' - 0

- 注意 parseInt 和 parseFloat 单词的大小写，这2个是重点
- 隐式转换是我们在进行算数运算的时候，JS 自动转换了数据类型

4. 数据类型转换

案例 1：计算年龄

此案例要求在页面中弹出一个输入框，我们输入出生年份后， 能计算出我们的年龄。



4. 数据类型转换

案例分析

① 弹出一个输入框（prompt），让用户输入出生年份 （用户输入）

② 把用户输入的值用变量保存起来，然后用今年的年份减去变量值，结果就是现在的年龄 （程序内部处理）

③ 弹出警示框（alert）， 把计算的结果输出 （输出结果）

4. 数据类型转换

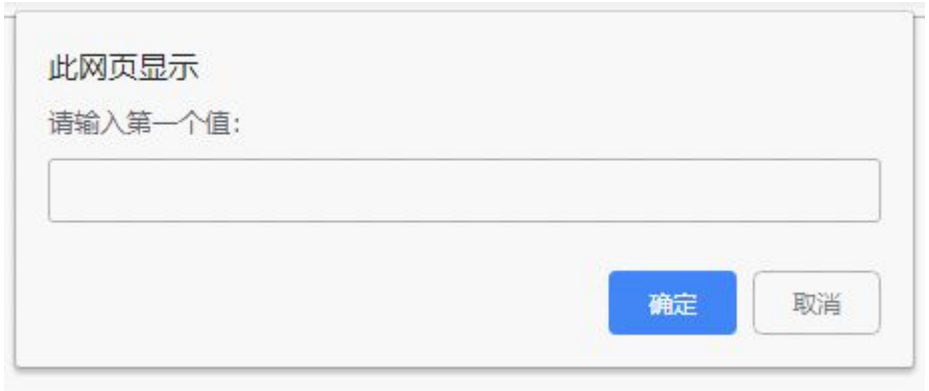
案例代码

```
// 1. 弹出输入框，输入出生年份，并存储在变量中
var year = prompt('请输入您的出生年份：');  // 用户输入
// 2. 用今年减去刚才输入的年份
var result = 2019 - year;                  // 程序内部处理
// 3. 弹出提示框
alert('您的年龄是:' + result + '岁');      // 输出结果
```

4. 数据类型转换

案例 2：简单加法器

计算两个数的值， 用户输入第一个值后，继续弹出第二个输入框并输入第二个值，最后通过弹出窗口显示出两次输入值相加的结果。



4. 数据类型转换

案例分析

① 先弹出第一个输入框，提示用户输入第一个值 保存起来
② 再弹出第二个框，提示用户输入第二个值 保存起来
③ 把这两个值相加，并将结果赋给新的变量（注意数据类型转换）
④ 弹出警示框（alert)， 把计算的结果输出 （输出结果）

4. 数据类型转换

案例代码

```
// 1. 先弹出第一个输入框，提示用户输入第一个值
 var num1 = prompt('请输入第一个值：');
// 2. 再弹出第二个框，提示用户输入第二个值
 var num2 = prompt('请输入第二个值：');
// 3. 将输入的值转换为数字型后，把这两个值相加，并将结果赋给新的变量
 var result = parseFloat(num1) + parseFloat(num2);
// 4. 弹出结果
 alert('结果是:' + result);
```

4. 数据类型转换

4.4 转换为布尔型

方式	说明	案例
Boolean()函数	其他类型转成布尔值	Boolean('true');

- 代表空、否定的值会被转换为 false ，如 ''、0、NaN、null、undefined
- 其余值都会被转换为 true

```
console.log(Boolean('')); // false
console.log(Boolean(0)); // false
console.log(Boolean(NaN)); // false
console.log(Boolean(null)); // false
console.log(Boolean(undefined)); // false
console.log(Boolean('小白')); // true
console.log(Boolean(12)); // true
```

传智播客旗下高端IT教育品牌