

JavaScript 基础第二天

类型转换、语句(分支、循环)

isNaN只是判断这个参数能否转成数字而已,并不是判断是否严格等于NaN,有隐式转换。 ES6中提供了一个Number.isNaN()方法用于判断一个值是否严格等于NaN,不存在转换类型的行为。





多一句没有,少一句不行,用更短时间,教会更实用的技术!

模块	说明	单词	作用
		Number('12')	转换为数字型
	转换为数字型	parseInt('12px')	转换为整数数字型
ソ 피 # + + 4		parseInt('12.5px')	转换为小数数字型
类型转换 	转换为字符串型	String(12)	转换为字符串型
		变量.toString()	转换为字符串型
	转换为布尔型	Boolean()	转换为布尔型
		ifelse	if分支语句
	分支语句	条件? 表达式1:表达式2	三元表达式
		switch case	switch分支语句
语句		while	while循环
	循环语句	for	for循环
		break	中止循环
		continue	中止本次循环继续下一次循环





- ◆ 类型转换
- ◆ 语句
- ◆ 综合案例





类型转换

- 显式转换
- 隐式转换



类型转换

类型转换: 把一种数据类型转换成另外一种数据类型

为什么需要类型转换呢?

例如: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

此时需要转换数据类型

数据类型转换可以分为: 显示转换和隐式转换



1.1 显式转换

概念:

自己手动写代码告诉系统该转成什么类型(数据类型明确、程序员主导)

格式:

Number('1') // 1

类型转换主要转换为数字型、字符串型、布尔型



1.1 显式转换

- 1. 转换为数字型
- ➤ Number(数据)
 - ✔ 转换成功返回一个数字类型
 - ✓ 转换失败则返回 NaN (例如数据里面包含非数字)
- ➤ parseInt(数据)
 - ✓ 只保留整数
 - ✓ 如果数字开头的字符串,只保留整数数字 比如 12px 返回 12
- ➤ parseFloat(数据)
 - ✓ 可以保留小数
 - ✓ 如果数字开头的字符串,可以保留小数 比如 12.5px 返回 12.5

parseInt和parseFloat专门用来将字符串转换为数值的两个方法

布尔型转换为数字: true 为 1 , false 为 0 null 转换为数字为 0, undefined 为 NaN



1.1 显式转换

转换为字符型

- String(数据)
 - > 返回字符串类型
- 变量.toString(进制)
 - ▶ 可以有进制转换

null和undefined没有toString方法

转换为布尔值

- ●Boolean(数据)
 - ▶ 返回 true 或者 false
 - ▶ 如果值为 false、0、"、null 、undefined、NaN,则返回 false,其余返回为 true



1.2 隐式转换

某些运算符被执行时,系统内部自动将数据类型进行转换,这种转换称为隐式转换。

学了隐式转换可以帮我们解决很多疑惑~



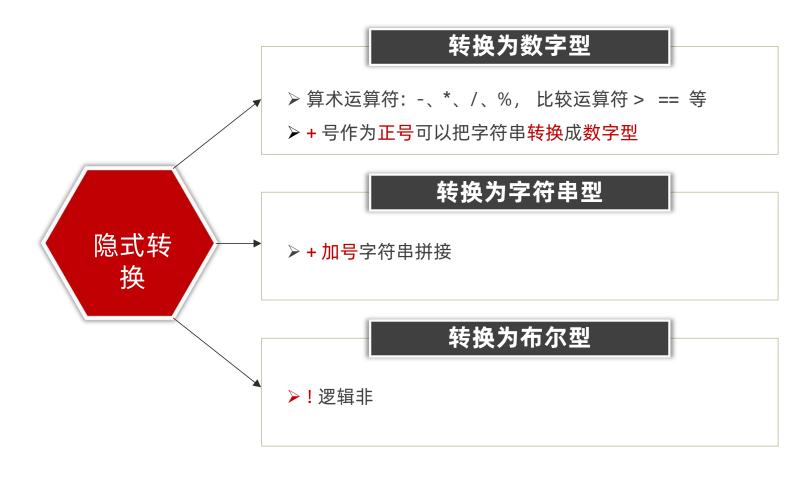


- ==和===
- ==比较的是值,通过 valueOf 进行隐式转换,通常由 JS 在后台自动调用,并不显式地出现在代码中
- ===除了比较值,还比较类型

商品名称	商品价格	商品数量	总价	收货地址
小米青春版PLUS	1999元	2	3998元	黑马程序员



1.2 隐式转换



注意: 隐式转换类型不明确, 靠经验才能总结, 尽量数据类型统一再做计算





- ◆ 数据类型
- ◆ 语句
- ◆ 综合案例



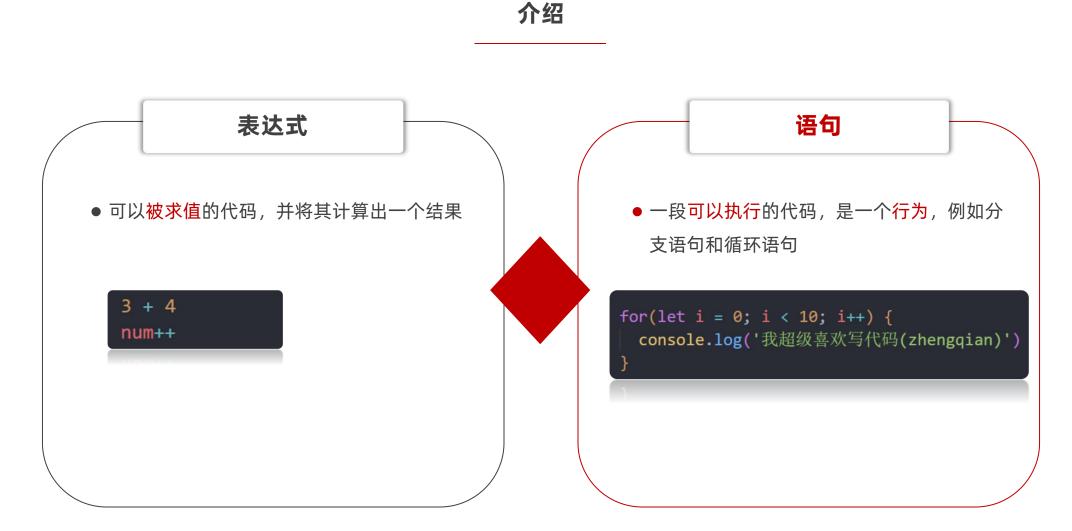


语句

- 表达式和语句
- · 分支语句
- 循环语句



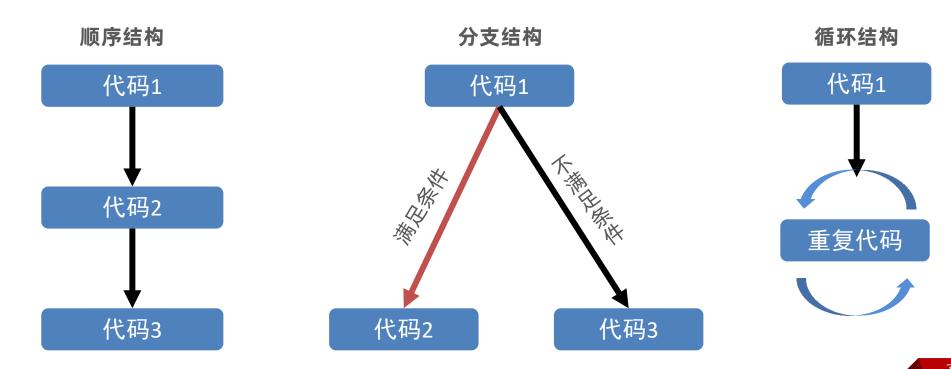
2.1 表达式和语句



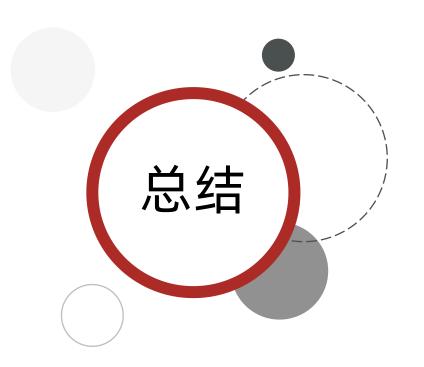


程序三大流程控制语句

- 以前我们写的代码,写几句就从上往下执行几句,这种叫顺序结构
- 有的时候要根据条件选择执行代码,这种就叫分支结构
- 某段代码被重复执行, 就叫循环结构







1. 什么是表达式? 什么是语句?

▶ 表达式:可以被求值的代码

▶ 语句:一段可以执行的代码,是一个行为

2. 程序三大流程控制语句分别是什么?

- ▶ 顺序结构(从上往下依次执行)
- ▶ 分支结构(选择执行)
- 循环结构(重复执行)



2. 分支语句

- 分支语句可以根据条件判定真假,来选择性的执行想要的代码
- 分支语句包含:
 - ▶ if分支语句(重点)
 - ▶ 三元运算符
 - ➤ switch 语句





2.1 if 语句

• 使用语法:

```
if (条件) { 满足条件要执行的代码 }
```

- ▶ 小括号内的条件结果是布尔值,为 true 时,进入大括号里执行代码;为 false,则不执行大括号里面代码
- ▶ 小括号内的结果若不是布尔类型时,会发生类型转换为布尔值,类似Boolean()
- ▶ 如果大括号只有一个语句,大括号可以省略,但是,俺们不提倡这么做~





if语句

●课堂案例1: 用户输入高考成绩, 如果分数大于等于700分, 则提示 '恭喜考入黑马程序员'



2.1 if 语句

• if 双分支语句:

```
let score = +prompt('请您输入高考成绩:')
if (score >= 700) {
    alert('恭喜您考入黑马程序员')
}
```





判断用户登录案例

需求:用户输入,用户名:刘德华,密码:123456,则提示登录成功,否则提示登录失败

分析:

①: 弹出输入框,分别输入用户名和密码

②:判断如果用户名是刘德华 并且密码是 123456,则提示登录成功,否则提示登录失败

a	I			(m	
a				里马程	
1. 明可	2.阿 3.吖	4. 嗖	5. 锕 ◀▶ 🖸	黑马程 www.itheim	a.com



2.1 if 语句

• if 多分支语句:

使用场景: 适合于有多个条件的时候, 比如学习成绩可以分为

if (条件1) { 代码1 } else if (条件2) { 代码2 } else if (条件3) { 代码3 } else { 代码n

①: 成绩90以上是优秀

②: 成绩70~90是 良好

③: 成绩是60~70之间是 及格

④: 成绩60分以下是不及格

释义:

- ▶ 先判断条件1,若满足条件1就执行代码1,其他不执行
- ▶ 若不满足则向下判断条件2,满足条件2执行代码2,其他不执行
- 若依然不满足继续往下判断,依次类推
- ➤ 若以上条件都不满足,执行else里的代码n





输入成绩输出评语案例

需求:根据输入不同的成绩,反馈不同的评价

注:

①:成绩90以上是优秀

②: 成绩70~90是 良好

③: 成绩是60~70之间是 及格

④:成绩60分以下是不及格



2. 分支语句

- 分支语句可以让我们有选择性的执行想要的代码
- 分支语句包含:
 - ➤ If分支语句
 - ▶ 三元运算符
 - ➤ switch 语句



2.2 三元运算符

● 使用场景: 一些简单的双分支,可以使用 三元运算符(三元表达式),写起来比 if else双分支 更简单

• **符号:** ? 与: 配合使用

● 语法:

x + y 二元运算 符

条件 ? 表达式1: 表达式2

x++ 一元运算符

● **执行过程:** 如果条件为真,则执行表达式1; 如果条件为假,则执行表达式2





· 判断2个数的最大值(导师讲解)

需求: 用户输入2个数, 页面弹出最大的值

分析:

①: 用户输入2个数

②: 利用三元运算符输出最大值

此网页显示		
请输入第一个数		
I		
	确定	取消





数字补0案例

需求:用户输入1个数,如果数字小于10,则前面进行补0,比如 09 03 等目的:

①: 练习三元运算符

②: 为后期页面显示时间做铺垫

22:00 点场 距结束

00:02:18



2. 分支语句

- 分支语句可以让我们有选择性的执行想要的代码
- 分支语句包含:
 - ▶ If分支语句
 - ▶ 三元运算符
 - ➤ switch 语句



2.3 switch语句

使用场景: 适合于有多个条件的时候,也属于分支语句,大部分情况下和 if多分支语句 功能相同

语法:

```
switch (表达式) {
 case 值1:
   代码1
   break
 case 值2:
   代码2
   break
 // 可以有多个case
 default:
   代码n
```

释义:

- ▶ 找到跟小括号里数据全等的case值,并执行里面对应的代码
- ➤ 若没有全等 === 的则执行default里的代码

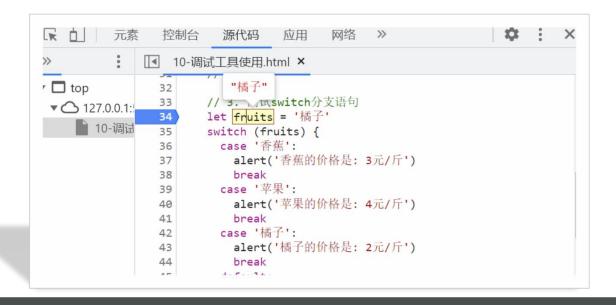
注意事项

- 1. switch case语句一般用于等值判断, if 适合于区间判断
- 2. switch case一般需要配合break关键字使用 没有break会造成case穿透
- 3. if 多分支语句开发要比switch更重要,使用也更多



断点调试-chrome调试工具

- 作用: 学习时可以帮助更好的理解代码运行,工作时可以更快找到bug
- 浏览器打开调试界面
 - 1. 按F12打开开发者工具
 - 1. 点到<mark>源代码一栏(sources)</mark>
 - 1. 选择代码文件
- 断点:在某句代码上加的标记就叫断点,当程序执行到这句有标记的代码时会<mark>暂停</mark>下来







语句

- 表达式和语句
- 分支语句
- 循环语句



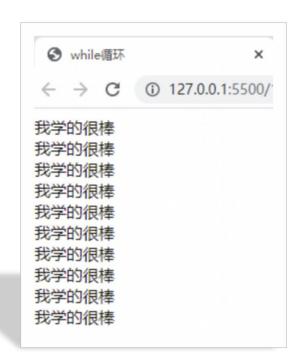
2.3 循环语句

使用场景: 重复执行 指定的一段代码, 比如我们想要输出10次 '我学的很棒'

学习路径:

1. while循环

1. for 循环 (重点)





1. while 循环

while: 在....期间, 所以 while循环 就是在满足条件期间, 重复执行某些代码。

● while 循环基本语法:

```
while (循环条件) {
 要重复执行的代码(循环体)
}
```

释义:

- ▶ 跟if语句很像,都要满足小括号里的条件为true才会进入循环体执行代码
- ▶ while大括号里代码执行完毕后不会跳出,而是继续回到小括号里判断条件是否满足,若满足又执行大括号里的代码,然后再回到小括号判断条件,直到括号内条件不满足,即跳出



1. while 循环

● while 循环三要素:

- 1. 初始值 (经常用变量)
- 2. 循环条件
- 3. 变量计数 (常用自增或者自减)





在页面中打印输出10句"月薪过万"(学生独立练习)

需求:使用while循环,页面中打印,可以添加换行效果



2. for 循环 (重点)

• 作用: 重复执行指定的一段代码

```
初始值
while (循环条件) {
    // 循环体
    变量计数
}
```

```
for(初始值;循环条件;变量计数) {
// 满足条件执行的循环体
}
```

● **好处:** 把声明初始值、循环条件、变量计数写到一起,让人一目了然,它是<mark>最常使用</mark>的循环形式







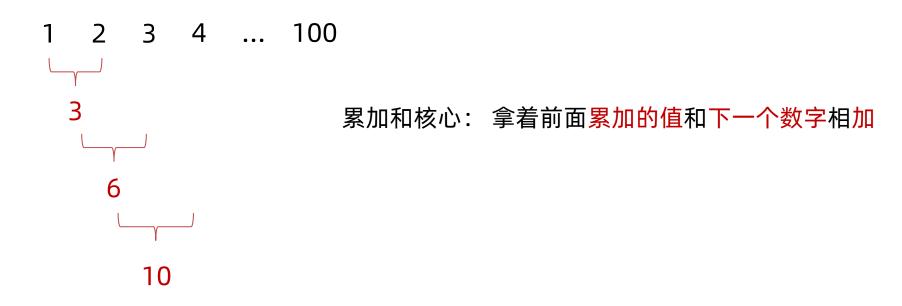
- for 循环练习
- 1. 页面输出5个小星星 ★★★★★
- 2. 页面输出年龄(1~100岁)
- 3. 页面输出1~100的偶数







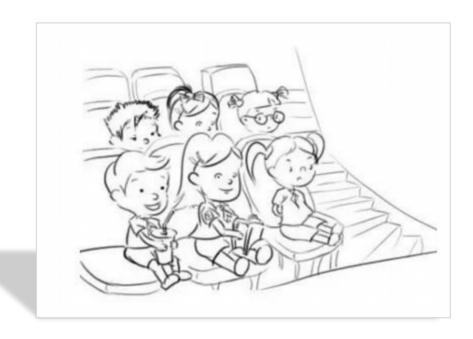
- for 循环练习
- 4. 求1-100的累加和 (1+2+3...+100的和)
- 5. 求1-100之间所有的偶数和





2. for 循环

- 中止循环
- ▶ break 中止整个循环,一般用于结果已经得到,后续的循环不需要的时候可以使用(提高效率)
- > continue 中止本次循环,一般用于排除或者跳过某一个选项的时候







2. for 循环

● 无限循环

- 1. while(true)来构造"无限"循环,需要使用break退出循环。
- 2. for(;;) 也可以来构造"无限"循环,同样需要使用break退出循环。



2. for 循环

● 无限循环

- 1. while(true)来构造"无限"循环,需要使用break退出循环。
- 2. for(;;) 也可以来构造"无限"循环,同样需要使用break退出循环。





- ◆ 运算符
- ◆ 语句
- ◆ 综合案例





简易ATM存取款机案例

需求: 用户可以选择存钱、取钱、查看余额和退出功能





简易ATM存取款机案例

需求:用户可以选择存钱、取钱、查看余额和退出功能

1. 取款			
 存款 查看余额 			
4. 退出			
	т		
	7		

分析:

①:提示输入框写到循环里面(无限循环)

②:用户输入4则退出循环 break

③:提前准备一个金额预先存储一个数额 money

④:根据输入不同的值,做不同的操作

(1) 取钱则是减法操作,存钱则是加法操作,查看余额则是直接显示金额

(2) 可以使用 if else if 多分支 来执行不同的操作





模块	说明	单词	作用
		Number('12')	转换为数字型
	转换为数字型	parseInt('12px')	转换为整数数字型
¥ =11 ++ +4		parseInt('12.5px')	转换为小数数字型
类型转换 	转换为字符串型	String(12)	转换为字符串型
		变量.toString()	转换为字符串型
	转换为布尔型	Boolean()	转换为布尔型
		ifelse	if分支语句
	分支语句	条件? 表达式1:表达式2	三元表达式
		switch case	switch分支语句
语句	循环语句	while	while循环
		for	for循环
		break	中止循环
		continue	中止本次循环继续下一次循环



传智教育旗下高端IT教育品牌