DAY02 逻辑分支

2023年4月19日 11:12

语句和语句块的理解

1.语句:

语句就是JavaScript指令,通过这些指令可以执行特定任务,或者设计程序的逻辑结构。

2.语句块:

就是一条或者多条语句的组合,一般写在一个花括号{}中。

一条语句(JS指令),帮助我们实现输出

程序的三大结构

1.顺序结构 - 最自然的执行方式

顺序结构表示程序中的各操作是按照它们出现的先后顺序执行的。

顺序: 上-下 左-右

2.选择结构

选择结构表示程序的处理步骤出现了分支,需要根据某一特定的条件选择其中的一个 分支执行。

选择结构有单选择、双选择和多选择三种形式。

3.循环结构

循环结构表示程序反复执行某个或某些操作,直到某条件为假(或为真)时才可终止循环。

if语句

大多数编程语言中最为常用的一个语句就是if语句。

if语句的三种格式

1.单分支

```
语法格式:
```

```
if(条件){
```

如果if语句后面的条件成立(真),执行语句块里面的代码

2.双分支

```
语法格式:
 if(条件){
 如果if语句后面的条件成立(真),执行语句块里面的代码
 }
 else{
 如果if语句后面的条件不成立(假),执行语句块里面的代码
3.多分支
 语法格式:
 if(条件1){
 满足条件1执行的代码块
 else if(条件2){
 满足条件2执行的代码块
 }
 else if(条件3){
 满足条件3执行的代码块
 }
 else{
 上面的条件都不满足时执行的代码块
 }
```

if语句的特点

1.if语句的条件可以是任意表达式,但结果一定会转换成布尔值(系统强制调用Boolean 函数进行转换)

注意:布尔值为假的情况null,undefined,0,",NaN,其他的都是真。

2.else可以省略,else可以短路(短路指的是代码不执行)

注意: 尽量将最大的可能给if语句 - 提升性能

解读:

因为如果执行了if语句,else不会执行,反之你执行了else语句,一定执行了if语句。

if语句的嵌套使用

不嵌套的写法弊端: 提示信息不够明确, 但是语句简练一些

```
// var num = prompt('请输入一个数字: ');

// if (!isNaN(num) && num % 2 === 0) {//是数字且被2整除

// alert('偶数')

// } else {

// alert('不是数字或者是奇数');

// }
```

程序的多分支结构 switch语句

switch 语句与 if 语句的关系最为密切,而且也是在其他语言中普遍使用的一种流程控制语句。

1.switch语句的语法格式

```
switch (值) {
    case 值: 语句块; break;
    case 值: 语句块; break;
    case 值: 语句块; break;
    .....
    default: 语句块;
```

解读

switch多分支语句核心关键字, switch括号里面的值和case后面的值恒等(===), 执行case后面的语句块。

case表示情形(情况,分支)的含义,具有穿透力(没有碰到break一直向下执行到碰到break或者整个程序的最后)。

break含义用来结束switch,只要碰到break,整个的switch语句就结束了。default含义是其他情形,case之外的情形,类似于else,可以省略

对比if和switch的区别 - 非常重要

- 1.这两个语句是可以相互替换的, switch-case适合比较确定值的情况, 而if-else常用于范围判断
- 2.当分支比较少的时候,if-else语句的执行效率比switch语句高;反之switch效率高,而且结构清晰。

3.重点的区别

switch语句进行条件判断后直接执行到满足case的语句(一步到位,不按顺序),效率更高。

而if-else语句,有几种条件,就有可能进行几次判断。 switch语句(switch大部分情况是具体的值,包括布尔值)

```
console.log('A');
                                                          var score = +prompt('请输入分数0-100'); //75
else if (score >= 90 && score < 100) {第二
                                                           switch (true) {
  console.log('B');
                                                            case score === 100: console.log('A'); break;
                                                            case score >= 90 && score < 100: console.log('B'); break;</pre>
else if (score >= 80 && score < 90) {第三
                                                            case score >= 80 && score < 90: console.log('C'); break;</pre>
  console.log('C');
                                                            case score >= 70 && score < 80: console.log('D'); break; case score >= 60 && score < 70: console.log('E'); break; case score >= 0 && score < 60: console.log('F'); break;
else if (score >= 70 && score < 80) {第四
  console.log('D');
                                                              console.log('输入有误');
else if (score >= 60 && score < 70) {第五步
 console.log('E');
else if (score >= 0 && score < 60) {第六
  console.log('F');
  console.log('输入有误');
```

三目运算符(取代简单的分支语句, 让简单的分支写的简单) - 重点应用

1.三目运算符(三元运算符)的语法格式:三目的符号(?:)

条件?语句1:语句2

解读:条件满足执行语句1,否则执行语句2

- 2.三目的特点
 - 2.1.三目带有返回值
 - 2.2.三目语句也可以嵌套使用,但是语句仅支持一条语句
 - 三目带有返回值

```
var n = prompt('请输入数字');
n = n < 10 ? '0' + n : n;
console.log(n);
// console.log(n < 10 ? '0' + n : n);</pre>
```

运算符的优先级

- 1.利用括号改变所有的优先级
- 2.列举的常见的符号

```
()!++--*/%+-&&||?:=+=
```

比先级	运算符	结合性
1	0 [] .	从左到右
2	!+正)-倹)~++	从右向左
3	* / %	从左向右
4	+ (加) - (咸)	从左向右
5	<< >> >>>	从左向右
6	< <= > >= instanceof	从左向右
7	g==	从左向右
8	&(按位与)	从左向右
9	•	从左向右
10	I	从左向右
11	& &	从左向右
12		从左向右
13	?:	从右向左
1470	= += -= *= /= % = &= = ^= ~= <<= >>= >>>=	从右向左