# JS 第四天

## 昨日内容回顾

### 判断类型

typeof
用来显示当前内容的基础类型

| | |
|---|---|
| 数字 | number |
| 文字 | string |
| 真假 | boolean |
| null | object |
| undefined | undefined |
| NaN | number |
| 函数 | function |
| 数组 | object |
| 对象 | object |

,

### 类型转换

#### 隐式类型转换(自动类型转化)

```
隐式类型转换(自动类型转化)
*
* 在程序的解释和执行的过程当中,
* 自动根据需要进行的类型转化
* */
console.log(10 - "5");

console.log(123 + "asd");

console.log(123 + "");

console.log( + "asd");

console.log( + "123");
```

#### 显式类型转化(强制类型转化)

```
console.log(Number("123"));     //123
console.log(Number("asd"));     //NaN
console.log(Number("123asd"));  //NaN
console.log(Number(null));      //0
console.log(Number(undefined)); //NaN
console.log(Number(true));      //1
console.log(Number(false));     //0
```

```
parseInt()
*
*  1.在转化数字的时候，会前面的空格忽略
*  2.在转化的过程中，如果内容中间部分存在无法转化的内容
*       会从前向后，尽可能的去转化
*       如果第一个字符就是无法转化的内容，则返回 NaN
*  3.parseInt() 在使用的过程中其实可以放置两个参数
*       * 需要转化的内容
*       * 需要转化的内容的进制
*       如果未填写第二个参数，则默认以十进制转化
```

```
console.log(parseFloat("12.3"));
console.log(parseFloat("       12.3"));
console.log(parseFloat("12.3+12.4"));
console.log(parseFloat("12..3"));
console.log(parseFloat(".12"));
```

```
Number 和 parseInt 的区别
*
* 1.parseInt 是对数字进行取整
*   Number 只是对数字进行转化
* 2.parseInt 从前向后进行转化，直至不能识别的内容为止
*   Number 只要后面存在无法转化的，直接返回 NaN
```

```
/*
 toFixed
 *
 * 可以按照指定的小数去返回数值
 * 返回的内容为 字符串
 * */
var num = 10;
console.log(num.toFixed(2));
```

```
console.log(Boolean(1));        //true
console.log(Boolean(0));        //false
console.log(Boolean(null));     //false
console.log(Boolean(undefined));///false
console.log(Boolean("123"));    //true
console.log(Boolean(""));       //false
console.log(Boolean(true));     //true
console.log(Boolean(false));    //false
console.log(Boolean(NaN));      //false
```

```
console.log(String("123"));
console.log(String("ads"));
console.log(String(null));
console.log(String(undefined));
console.log(String(true));
console.log(String(false));
console.log(String(NaN));
console.log(String(""));
```

```
转化成文字
*
*  1.String()
*  2. +
*  3.toString()
*
*
*  使用 String(),我们需要将转化的内容放置在括号内部
*  使用 toString(),我们需要让哪一个变量变成字符串,
*  就在这个变量后面去书写 .tostring()
```

### 提示框的使用

```
提示框的使用
* */

var num1 = parseInt(prompt("请输入第一个数值"));
console.log(num1);

var num2 = parseInt(prompt("请输入第二个数值"));
console.log(num2);

alert("您当前输入两个数的和是: \n" + (num1 + num2));
```

### 程序流程控制

#### 结构化程序

- 顺序结构
  - 顺序结构是一种线型的，有序的一种结构，它会依次去执行各个语句模块
- 选择结构
  - 选择结构是根据条件是否成立，来去选择程序执行的通路（需要条件语句）
- 循环结构
  - 循环结构是重复执行一个或者多个模块，直至满足某一条件为止（需要循环语句）

#### 语句

平时书写中，任意一行代码其实都是一条语句。
程序都是由若干条语句去组成的

### 选择结构

```
选择结构
*
*       1.if..
*       2.if..else..
*       3.if..else if..
*       4.if..else if..else..
*       5.switch
```

```
if 判断
*
*      if(condition){
*          条件为真时触发的行为
*      }
*  注意:
*      1.condition（条件）可以是任意的表达式
*        表达式的值不一定要为 布尔值
*      2.如果 condition 不是 布尔值，则会
*        尝试使用 Boolean() 去转化为 布尔值
*      3.如果 condition 成立，才会去执行
*        大括号内部的内容，否则不会执行
```

```
if..else..
*
*      if(condition){
*          条件为真时执行的行为
*      }
*      else{
*          条件为假时执行的行为
*      }
```

```
if..else if..
*
*       if(condition1){
*           满足条件1成立时执行
*       }
*       else if(condition2){
*           满足条件2成立时执行
*       }
*       ...
```

```
if..else if..else..
*
*       if(condition1){
*           condition1成立执行
*       }
*       else if(condition2){
*           condition2成立执行
*       }
*       else{
*           以上所有条件都不成立执行
*       }
```

### 昨日作业

3. 编程实现：请输入一个4位数的会员号，求这四位数字之和，如果大于20，则该会员是幸运会员 ，弹出消息框显示该会员是否为幸运会员

```
<!DOCTYPE html>
<html>

<head>
<meta charset="UTF-8">
<title></title>
</head>

<body>
<p>请输入一个4位数的会员号</p>
<input type="text" id="num"/>
<input type="button" value="确定" id="submit"/>
<script type="text/javascript">
var num = document.getElementById("num");
document.getElementById("submit").onclick = function()
{
var v = num.value;
var ge,shi,bai,qian;
ge = v%10;
shi = parseInt(v/10)%10;
bai = parseInt(v/100)%10;
qian = parseInt(v/1000);
if(parseInt(v) !== Number(v)){
alert("输入错误！请重新输入");
num.value = "";
}else if(ge + shi + bai + qian > 20){
alert("恭喜您成为幸运会员");
}else{
alert("很遗憾，未中奖");
}
}

</script>
</body>

</html>
```

## JS 常用的循环

- for 循环
- while 循环
- do..while 循环

### 三种循环的使用场景

- for
  - 一般用于精确控制循环次数的场景
- while
  - 一般用于循环次数不明确的场景
- do..while
  - 一般用于循环次数不明确的场景
  - 至少需要执行一次的场景