JS第三、四天

逻辑判断

if（ ）{} else{}判断

语法1：if（条件）{ js语句}
- 如果条件成立，就执行大括号中的JS代码，如果不成立就不执行大括号里的内容；
- if括号中的判断条件，最后会转换成布尔值true或false；即：true表示成立。false表示不成立

语法2；if(条件){js语句一}else{JS语句二}
- 如果条件成立 则执行语句一，如果条件不成立，则执行语句二

语法3 if(条件1){语句一} elseif（条件二）{语句二} else{语句三}

- 如果条件1成立则只执行语句一，条件1不成立则判断条件2，如果条件2成立的画，则执行语句二，如果条件二也不成立，则执行语句三；
  - elseif条件可以写无限多；

注意:例如，src，href，等这些地址类的属性，不能用来直接判断；

for循环

语法1：for(;条件;){ 语句 }
步骤：

- 1、变量初始化 `let i = 0;` 只需初始化一次
- 2、进条件判断，如果条件成立进循环体，否则就结束循环 `for（;条件;）`
- 3、进循环体 `{循环体}`
- 4、设置下次循环的条件 `i++`
- 5 然后继续进2.3.4步，直到判断条件不成立为止；

例如如下：**隔行变色；**

```
<button id="btn">点击切换</button>
    <ul>
        <li>1</li>
        <li>2</li>
```

```
        <li>3</li>
        <li>4</li>
        <li>5</li>
        <li>6</li>
        <li>7</li>
    </ul>
```

```
    <script>
        let lis = document.getElementsByTagName('li');
        btn.onclick = function(){
            for(var i=0;i<lis.length;i++){
                if(i%2){
                     lis[i].style.background = 'pink';

                }else{
                    lis[i].style.background = 'yellow';
                }
            }
        }
    </script>
```

语法1简化 for(变量初始化；条件；设置下次循环) {循环体}

for嵌套for

步骤：先执行外层大循环一次，在把里边的小循环执行完，然后在执行大循环的第二次，在执行一遍小循环，再执行大循环第三次。。。。。以此类推；

```
for(let i = 0;i < 4;i++){
            alert('大循环'+i)
            for(let i = 0;i<5;i++){
                alert("小循环"+i)
            }
        }
```

函数

- 定义：在js中属于代码块，函数内写功能的，目的为了复用；
- 组成：函数主要由函数声明和函数调用两个部分组成
- **定义函数原理：当定义一个函数的时候就是开辟了一个新的堆内存，然后把函数中的代码转成字符串存到堆内存中，最后把堆内存的地址，赋值给函数名或者变量。以便调用；**

- 调用函数原理：>在函数调用的时候，函数内形成一个执行栈—然后把函数内的字符串代码拷贝一份放到执行栈中执行–(执行过程：参数赋值—执行上下文—预解析的机制—函数内的变量销毁)
- 函数声明有三种声明方式：
  - 一、函数声明—function 自定义函数名（ ）{ }
    - 调用时可以在声明之前调用
  - 二、函数表达式—let 自定义名称 = function（ ）{ }
    - 调用时必须在声明之后才可以；不然报错；
    - 把匿名函数用`()`包起来，也是函数表达式，调用的时候直接在括号外加（ ）；
  - 三、类声明—let 自定义名称 = new function（ ）；
    - 函数内部（括号内部）要为字符串，调用可以在声明之前；
      如果函数不带名字，属于匿名函数，直接声明匿名函数会报错。想要避免报错的话，1、加上名字。2、匿名函数要为一个表达式；
- 调用方式：（只有函数才能调用，不是函数就报错）
  - 函数名 + 括号调用
    `函数名（）`
  - 事件调用
    `btn.onclick = function()`点击事件
    一般是赋值一个匿名函数，或者赋值一个函数（名）地址；
  - 定时器调用
    - setTimeout(function(){},1000)
- 匿名函数自执行调用方法
  - （匿名函数）()

函数参数

在定义函数的时候，自定义的参数名叫**形参**；（形式上的）（当做变量来看）

```
funcction fn(a,b,c,d){
console.log(a)
}
```

例如a,b,c,d是形参
在调用函数的时候，括号中传的数据叫**实参**；（实际上的）

```
fn（1,2,3,4）
```

例如1,2,3,4对应着形参的。就是实参；
参数与参数之间用逗号隔开，参数是可以为多个的；

注意：形参是对应实参的（相当于变量名对应着赋值）；他们是一一对应的；如果形参没有对应的实参，实参必须占位；**实参可以是任何数据类型**；

函数传参（重要）

```
<script>
        function a(b){
            // console.log('这是第一个函数')
            b( function(d){
                d(function(f){
                   f()
                    // console.l  og('这是第五个函数')
                })
                // console.log('这是第三个函数')
            });
        }
        a( function(c){
            c(function(e){
                e(function(){
                    console.log('这是第六个函数')
                });
                // console.log('这是第四个函数')
            })
            // console.log('这是第二个函数')
        })
    </script>
```

arguments[0]是一个在函数内的类数组；（长得像数组，但不是数组），这个数组代表的是实参的集合；
arguments是和形参一一对应的；
通过下标获取某个实参；
一般实参有很多个，能用到arguments；

```
function fn(){
        // console.log(a+b+c);
        // console.log(arguments[2]);
        let num = 0;
        for(let i=0;i<arguments.length;i++){
            num += arguments[i];
        }
        console.log(num);
    }
    fn(1,2,3,4,5,6,7,78,8,2,34,6,7,83,9);
```

函数返回值

函数内的参数或者变量**默认**是不能被外界所访问的
函数内如果没有某个参数或者变量，会去函数外查找，知道window结束；
当一个函数在调用（函数名+括号）的时候，做了两件事：
1、执行函数内的代码
2、函数返回值
- 函数返回值默认（没写undefined）为undefined；
- 通过`return`去设置函数返回值；return后边是什么，函数返回值就是什么，如果return后边没有值，那么返回值也是undefined；
- **注意**：`return`会终止后面的代码执行，return下面的代码都不会执行；
- 如果函数中使用了循环，并且也使用了return，那么return会终止循环；

```
// function sum(a,b){
    //     console.log(a+b);
    //     return a+b;
    // }
    // let x = sum(1,6);
    // console.log(x);//7
```

while循环

for in循环

通过枚举对象身上的属性名，来做到循环的目的
循环的次数跟属性的个数有关系；
只要是遍历对象的情况下；用for in循环；

```
/*
        通过下面的对象，生成对应的样式。
    */
    box = document.getElementById('box')
    let obj = {
        width:'100px',
        height:'200px',
        background:'skyblue',
        border:'1px solid #000'
    }
    for(let a in obj){
        console.log(a);
        box.style[a]= obj[a];
    }
```

例如以上代码，获取obj对象中的属性名个数为循环次数。

a就是获取的对象中的属性名；
console.log(obj[a]),输出的是obj对象中的a属性的值；

switch 判断

语法：switch（变量名）{
case 条件一：
语句一
break；
case 条件二
语句二
break；
default：
语句三
break；
}
break必须写，不然会穿透（当一个条件成立之后，还会执行下边的代码）
能用switch一定能用if判断，能用if判断的地方，不一定能用switch；