第二天笔记

- 第二天笔记
  - 基本数据类型和引用数据类型的区别
  - 字符串拼接
  - js中数据类型的检测
    - typeof value
    - instanceof
    - constructor
    - Object.prototype.toString.call()
  - 数据类型之间的比较
  - 判断语句
    - if else
    - switch
    - 三元运算符 （条件运算符/三目运算符）
  - 循环
    - for循环
      - break出现在循环体中，break后面的代码不再执行，整个循环结束
      - continue出现正在循环体中，循环体continue后面的代码不执行，但是循环依然正常
    - for in循环
  - 复习

基本数据类型和引用数据类型的区别

基本数据类型直接对值进行操作
引用数据类型操作的是内存地址

> 1、浏览器提供给代码运行的环境，代码在这里从上到下执行
> 2、当遇到引用数据类型的时候，浏览器会开辟一个新的内存空间，将引用数据类型中的代码当做字符串存到这个里面
> 3、开辟的空间会返回给对象一个地址，那么此时这个对象储存的不是代码而是一个地址
> 4、当将这个对象赋值给另一个对象的时候，其实赋值过程是将地址传递给另一个对象了
> 也就是说，这两个对象将要访问同一个地址，也就是说他们操作的是同一份代码

字符串拼接

在js中其他数据类型+=一个字符串的话，全部会变成字符串

```
1.      var str1='';
2.      str1+=true;
3.      console.log(str1);//'true'
```

js中数据类型的检测

typeof value

返回值永远是一个**字符串**里面包着数据类型

```
1. typeof undefined --> 'undefined'
2. typeof 1 --> 'number'
3. typeof '' --> 'string'
4. typeof true --> 'boolean'
5. typeof []--> 'object'
6. typeof {} --> 'object'
7. typeof null--> 'object'
8. typeof function (){}--> 'function'
9. typeof 不能具体细分object下面的数据类型
10. typeof 人为null也是object下的数据类型
11.
12. /*typeof (typeof (typeof (typeof (typeof
    undefined))))
13.     'string'*/
```

instanceof

运算返回值是布尔类型的值
判断已知对象数据类型的方法
（ 判断一个实例是否属于这个类–>例如：[] 是不是属于数组类 ）
instanceof 在检测的时候 基本数据类型不能检测

```
value instanceof Class
```

constructor

运算返回值是布尔类型的值

```
[1,2,3].constructor === Array
```

=== 绝对比较 左右两侧的数据类型完全一致

Object.prototype.toString.call()

```
(Object.prototype.toString.call('珠峰培训') === '[object String]')---> true
```

数据类型之间的比较

- 对象==对象 永远为false

    - ({}=={}) []==[] {}==[]

- 数字==字符串 字符串调用Number()方法，将字符串转为数字，然后再进行比较

```
1==='1'
1==Number('1')
```

- 数字==布尔 布尔转为数字数据类型，调用Number() true = 1 false = 0

```
1==false
1==Number(false)
1==0
```

- 数字==对象 对象调用.toString()变成字符串，字符串调用Number()方法变成数字

```
1==[]
1==[].toString() --> 1=='' --> 1==Number('')--> 1==0 --> false

1==[1]
1==[1].toString() --> 1=='1' --> 1==Number('1')--> 1==1 --> true

1=={}
1=={}.toString() --> 1=="[object Object]" --> 1==Number("[object Object]")--> 1==NaN --> false

{}.toString() --> "[object Object]"
```

- 对象==字符串 对象调用.toString()变成字符串

```
''==[]
''==[].toString()-->''=='' -->true

''=={}
''=='[object Object]' --> false
```

- 字符串==布尔 双方都变成数字数据类型

```
12==true  -- > false
```

- 对象==布尔 双方转化为数字

- null==undefined true
- null===undefined false ===要求等号两侧的数据类型一致
- NaN==NaN false

> 对象和其他数据类型比较的时候，最先都要转为字符串
> 最终比较的时候，其实都是数字数据类型在比较

判断语句

if else

```
(条件) --> true/false

1 if(条件){条件成立执行的代码}

2 if(条件){条件成立执行的代码}else{条件不成立执行的代码}

3 if(条件1){
    条件1成立执行的代码
}else if(条件2){
    条件2成立执行的代码
}else if(条件3){
    条件3成立执行的代码
}....else{
    以上条件都不成立执行的代码
}
```

switch

是绝对比较（条件和值进行比较的时候必须是相同数据类型），用在不同值的不同操作上面
ps：如果成立的条件后面没有break，那么这个判断会从成立的那个条件开始，一直执行到最后

```
var n=1;
      switch (n){
          case '0':
              alert('0');
              break;//如果break跟着的这个条件成立，就跳出（中断）判断，不会继续执行判断了
          case '1':
              alert('1');
              break;
          default:
              alert('以上都不成立');
      }
```

三元运算符（条件运算符/三目运算符）

基于某些条件对变量进行赋值的条件运算符

声明 变量名=(条件)?条件成立时将这个值给变量:条件不成立将这个值给到变量;
var num=(!0)?2:3; – > num=2

!0?alert(‘ok’):void (0);
void (0) 用来占位

!0?(alert(‘ok’),console.log(‘ok’)):(alert(‘no’),console.log(‘no’));
如果要执行的是多个语句，需要将这些语句放在一个括号中，用逗号隔开

循环

for循环

for循环适用于已知循环次数

- 声明一个变量 设置初始值
- 设定循环范围
- 执行循环体中的代码
- 设置初始值的累加操作

```
for(var i=0;i<=3;i++){
    循环体中的代码
    //i++;
}
```

break出现在循环体中，break后面的代码不再执行，整个循环结束

```
for (var i=0;i<3;i++){
        console.log(i);// 0
        break;
        i++;//两个i++都不会再执行了
    }
    console.log(i);// 0
```

continue出现正在循环体中，循环体continue后面的代码不执行，但是循环依然正常

```
for (var i=0;i<3;i++){
      console.log(i);// 0 1 2
         continue;
      i++;//只有这一行受到了影响
      }
      console.log(i);// 3
```

> i++ –> 先赋值 再累加
> ++i –> 先在自身累加 再赋值

for in循环

作用：遍历对象的属性名和属性值

```
var obj={
        name:'珠峰',
        age:'9年',
        url:'www.zhufengpeixun.cn'
    };
    for (var key in obj){
        //key 变量  代表这个对象的属性名
        console.log(obj[key]);
        //在遍历对象时，不知道这个对象的属性名是否存在数字，所以只能用对象名[变量名]的形式获取属性值
        //obj[key] 中括号中是变量 而不是一个属性 所以不能加引号
    }
    //for in循环只能遍历可枚举属性 （自定义属性）
```

1 隔行变色
2 点击弹出
3 自定义属性
4 99
5 选项卡

数组的基础知识

复习

js的定义
js的引入方式
js组成
js的命名规范
变量 属性名
数据类型
基本数据类型：数字 字符串 布尔 null undefined
引用数据类型：
对象数据类型：数组类（Array） 对象类（Object） 正则类（RegExp） 数学函数类（Math）
函数数据类型：function
js的输出
变量：
是用来储存值和代表值，弱类型语言所以变量在js中什么数据类型的值都可以放置
声明 变量名=值；
var ES5 let ES6

对象
对象名.属性名
对象名[‘属性名’]

Number()
parseInt()
parseFloat()
数字或NaN
isNaN()
true false

Boolean() <==> !!
! 先进行布尔运算后取反
如何判断一个值是真是假：0 “” (空字符串) NaN null undefined

![] – > false
!!{} – >true

只要两侧有引号的字符都是字符串，字符串在js中没有意义

null 预留一个位置
undefined 本应有值 但是没有