js是一门弱类型语言

数据类型

1.数值类型(number)

2.字符串(String)

3.布尔类型(Boolead)

4.函数类型(function)

5.对象类型:object(包含数组,null)

6.undefined

```
//定义变量
 //var 变量名=值
 var a = 10;
 //打印变量
 // a.log
 console.log(a);
 //打印变量的数据类型
 console.log(typeof a);
 //字符串可以使用''和""
  a = "123";
  a = '123';
  console.log(a);
  console.log(typeof a);

  a = false;
  a = true;
  console.log(a);
  console.log(typeof a);

  //数组
  a = [1, 2, 3];
  console.log(a);
  console.log(typeof a);

  //对象
  a = {
  name: '张三',
  age: 18
```

```
};
console.log(a);
console.log(typeof a);

//null
a = null;
console.log(a);
console.log(typeof a);

//undefined 未定义的值
var b;
console.log(b);
console.log(typeof b);

var c = function () {
};
console.log(c);
console.log(typeof c);

//数字类型
a = 10;
a = 1.223;
a = NaN;//报错时
a = Infinity;//无穷大
//字符串
a = '啊哈哈';
a = "哈哈哈";
//布尔类型
a = true;
a = false;
//对象类型
a = null;
a = {};
a = [];
//函数类型
a = function () {
};
//underfined
a = undefined;

```

```
69  //输出语句
70 /* console.log("123");
71  console.error("123");*/
```

程序的三大结构;(和java一样)

顺序结构 分支结构 循环结构

```
1  if (true) {//分支
2
3  }
4  if (true) {
5
6  } else {
7
8  }
9  if (true) {
10
11  } else if (true) {
12
13  } else {
14
15  }
16  switch (a) {
17   case 1:
18   break;
19   case 2:
20   break;
21   default:
22   break;
23  }
24  var a = a > b ? a : b;
25  //循环
26  for (var i = 1; i <= 10; i++) {
27   console.log(i);
28  }
```

js没有块作用域,js是函数作用域

```
1  console.log(i);//11
2  while (false) {
3
4  }
5  do {
```

```
6
7 } while (false);
```

字符串的方法

```
1 a = "Hello,world!";
2 console.log(a.toLowerCase());
3 console.log(a.toUpperCase());
4 console.log(a.trim());
5 console.log(a.concat("123"));//字符串的拼接
6 console.log(a + "123");
7 console.log(a.charAt(0));
8 // console.log(a.reverse());//反转
```

函数

/*

function 函数名(参数列表) {

    函数体

}

*/

```
1 //定义方法
2 function sum(a, b) {
3  return a + b;
4 }
5
6 //调用函数
7 var result = sum(1, 2);
8 console.log(result);
```

dom操作

dom: document object moder,文档对象模型

domcument:内置对象,代表当前html文档,文档中的元素都可以通过domcument获取.

```
1  //获取元素
2  //html元素获取
3  var html = document.documentElement;
4  console.log(html);
5  //获取body元素
```

```
var body = document.body;
console.log(body);

//通过id获取某个元素
 var p1 = document.getElementById("one");
 console.log(p1);

 //通过标签名获取多个元素
 var ps = document.getElementsByTagName("p");
 console.log(ps);

 //通过css选择器获取某个元素
 var p = document.querySelector("p");
 console.log(p)
 //通过css选择器获取多个元素
 var ps = document.querySelectorAll("p");
 console.log(ps);

 //通过元素怒关系获取元素
 var parent = p.parentElement;
 console.log(parent);
 //前一个兄弟元素
/* var p1 = document.querySelector("p-nth-child)(2)").previousElementSib
ling;
 console.log(p1);*/

 //后一个兄弟元素
 var p2 = p1.nextElementSibling;
 console.log(p2);

 //获取子元素
 var children = body.children
 console.log(children);
```

修改元素的属性

```
var img = document.getElementById("girl");
img.src = '../img/girl.jpg'
var makeup = document.querySelector("#makeup");
//鼠标进入
var h1 = document.querySelector("#makeup>h1");
var img = document.querySelector("#makeup>img");
```

```
7
8  //事件(点击事件.键盘事件,浏览器事件)
9  makeup.onmouseenter = function () {
10   //修改文本开始标签结束的属性
11   h1.innerText = "化妆后";
12   img.src = "../img/after.jpeg";
13  };
14  makeup.onmouseleave = function () {
15   h1.innerText = "化妆前";
16   img.src = "../img/before.jpeg";
17  };
```

修改样式

```
1  <form id="color-form">
2   <input class="color" type="number" placeholder="红色" min="0" max="255">
3   <input class="color" type="number" placeholder="绿色" min="0" max="255">
4   <input class="color" type="number" placeholder="蓝色" min="0" max="255">
5   <input type="submit" value="修改">
6  </form>
```

```
1  h1.style.backgroundColor="red";
2  h1.style.fontSize="50px";
3
4  //练习:输入RGB的颜色值,修改body的颜色
5   var form=document.getElementById("color-form");
6   form.onsubmit=function (ev) {
7   var inputs=document.querySelectorAll(".color");
8   var red=inputs[0].value;
9   var green=inputs[1].value;
10   var blue=inputs[2].value;
11   document.body.style.backgroundColor="rgb("+red+","+green+","+blue+")";
12   //阻止事件的默认行为
13   return false;
14   }
```

js的创建对象(js-object):在定义方法中,window:浏览器内置对象

document是window的属性

所有的全局变量都是window的属性

为了减轻window的负担,避免变量污染,我们需要创建匿名函数自执行

```
1  //创建对象
```

```
var person={
//属性
name:"张三",
age:"16"
};
console.log(parent);

//添加属性
 person.gender="男";
 console.log(person);
 //定义方法
 //方式1
 function sayHi() {
 console.log("你好");
 };
 //调用方法
 sayHi();

 //方式2
 var sayHi1=function () {
 console.log("你好!");
 };
 sayHi1();

 parent.say=sayHi();//调用方法,把后面的方法赋值给say
 parent.say=sayHi;

 var parson={
 name:"张三",
 gender:"男",
 age:"16",
 //方法:函数,把函数当成属性传给say
 say:function () {
 console.log("你好!");
 }
 };

 //window:浏览器内置对象
 //document是window的属性
 //所有的全局变量都是window的属性
```

```
42  console.log(window);
43
44
45  //全局变量
46  var a=10;
47  a1=20;
48  console.log(a);
49  console.log(a1);
50
51 //js的变量作用域:函数作用域
52 // function f() {
53  // //局部变量
54  // var a2=10;
55  // }
56
57  //减轻window的负担,避免变量污染
58
59  // function f1() {
60  // var aa=10;
61  // }
62
63  //匿名函数自执行
64  (function f2() {
65  var aa=10;
66  console.log(aa);
67  })();
```

js顺序(js-loading):js当某行出现错误时,以后的代码不执行,body里div元素在script元素前面,先执行,后加载.反之,网页加载完后执行的方法(浏览器事件)

```
1  //js当某行出现错误时,以后的代码不执行
2  var a=10;
3  console.log(a);
4  console.log(a1);
5  console.log('你好啊');
6  var a1=20;
7
8  var one=document.querySelector("#one");
9  console.log(one);
10  //当网页加载完成后执行
11  window.onload=function (ev) {
12   var one=document.querySelector("#one");
```

```
13   console.log(one);
14 }
```