# 对象简介

* JavaScript：是一门基于**对象**的语言
* 具有面向对象的一部分特征
* 在JavaScript中，**一切都是对象**
* 某个具体的**实物**是一个对象（Object）比如：一只狗
* 对象具有一些**属性**（比如一只狗的名字叫做Lili，品种：哈士奇，体重：60斤）和行为（会做跳跃运动）
  + 属性：通过变量来表示

例： var name = “Lili”;

var breed = “Husky“;

var weight = 60；

* 行为：通过函数来实现

例： function jump( ){ }

* JavaScript中的对象
* 一系列相关属性和方法的集合

属性：**与对象相关的值**

方法：能够在对象上执行的**动作**

* 是一种数据类型

# 对象的基本使用方法

## 定义对象

### 方法一：利用var的方法定义一个对象

var dog = {

name : "Lili",

breed : "Husky",

weight : 60,

jump : function (){

alert("She is jumping");

}

}//定义了一个对象，对象的名字叫做dog

注意事项：

* 使用一对大括号表示**对象**，属性和方法写在之内
* 属性包括属性名和属性值，之间使用**冒号**分隔
* 属性和属性之间使用**逗号**分隔（最后一个后面可以没有符号）
* **方法同属性类似**，**只是值部分为一个函数体**

### 方法二：利用new的方法来进行对象的定义

var dog = new Object();//创建了一个空的对象,新生成了一个对象.

dog.name = "Lili";

dog.breed = "Husky";

dog.weight = 60;//在给对象的某个属性或者方法进行命名的时候，可以采用dog[‘weight’] = 60

dog.jump = function(){

alert("She is jumping");

}

var a = 'age';//或者直接dog['age'] = 4;

dog[a] = 4;//

alert(dog.a);// undefined

alert(dog.age);// 4

方式一：通过 对象名.属性名 对象名.方法名

方式二：通过 对象名['属性名' ]//**必须加引号**

## 访问对象属性/方法

方式一：通过 对象名**.**属性名 对象名**.**方法名( )

方式二：通过 对象名[“属性名”]

//访问对象属性

alert(dog.name);

alert(dog['breed']);

//使用对象方法

dog.jump();

dog['jump']();

## 修改对象属性

修改对象属性就是通过直接赋值给对象属性。

dog.weight = 70;

dog[‘weight’] = 70;

## 遍历对象属性

使用for in 语句

for （var *i* in *objName*）{

// i 在循环体内部的每次循环中代表一个属性名

// 通过 objName[ i ] 访问到每一个属性

}

<script>

//定义对象

*var* dog = {

name: "Lili",

breed: "Husky",

weight: 60,

loves: ['eat', 'sleep'],

jump: *function*() {

alert("She is jumping");

}

}

//遍历对象

for (*var* i in dog) {

//访问每一个属性名

alert(i); //访问的只是属性名，不是属性值

//访问每一个属性值

alert(dog[i]); //访问的是属性值，跟在属性值的后面

//访问数组的每一个元素

if (i == "loves") {

*var* len = dog[i].length;

for (*var* j = 0; j < len; j++) {

alert(dog[i][j]);

} //原因是因为i == loves 此时的loves是一个数组，需要依次访问的话，则需要进行遍历

}

}

</script>

添加和删除对象属性

对象创建之后，要向对象添加属性，只需要为新属性赋值即可

dog.age = 4;//或者是dog[‘age’] = 4;

alert(dog.age);

删除对象属性，使用delete关键字

delete dog.weight;

alert(dog.weight);//undefined

## 实例

天堂电影院最近播放《战狼2》和《羞羞的铁拳》两部电影。使用JS程序的对象分别来存储这两部电影的数据，包括：标题、类别、评分和播放时间（每天播放3次，时间自定）。

（1）思考并写出这两个对象

（2）增加一项功能能够显示该电影下一次播放的时间

方式一：定义一个函数getNextShowing

方式二：扩展电影对象，增加一个方法getNextShowing

1. 定义两个对象

var movie1 = {

name : '战狼2',

type : "战争",

scoer : 9.6,

showtime : ['8:00','10:00','15:00']

}

var movie2 = {

name : '羞羞的铁拳',

type : "喜剧",

scoer : 9.0,

showtime : ['9:00','12:00','17:00']

}

1. 计算下一次播放的时间

方法一：

function getNextshowing(movie){

var now = '10:00';

for(var i = 0;i < movie.showtime.length;i++){

if(parseInt(now) < parseInt(movie.showtime[i]))

break;

}

alert('下一次播放的时间是'+movie.showtime[i]);

}

getNextshowing(movie2);

方法二：

var movie2 = {

name : '羞羞的铁拳',

type : "喜剧",

scoer : 9.0,

showtime : ['9:00','12:00','17:00'],

getNextshowing:function(){

var now = '10:00';

for(var i = 0;i < this.showtime.length;i++){

if(parseInt(now) < parseInt(this.showtime[i]))

break;

}

alert('下一次播放的时间是'+this.showtime[i]);

}

}

## this关键字

this关键字在对象内部使用，指代当前对象

# 构造函数

## JavaScript构造函数的定义

* 能够创建出**对象**的一个特殊函数
* 类似一个工厂一样，能够**生产对象**
* 我们需要在工厂中设定“产品（对象）”的“模板（结构）”
* 在“生产产品（创建对象）”时**指定产品的“具体参数（属性值）**”

（1）使用**function**关键字声明，为了区别普通函数，将**构造函数首字母大写**

（2）构造函数的**形参**用于赋给对象的属性

（3）在构造函数中，使用**this**指代对象，同时构造函数里面都是语句

（4）对象的属性和方法**必须通过this**访问

function Movie(title, showtime) {

this.title = title;

this.showtime = showtime;

this.getNextShowing = function(){

var now = "10:00";

//对象内部，使用this指代本对象

for (var i=0; i<this.showtime.length; i++){

if (parseInt(now) < parseInt(this.showtime[i])){

alert(this.title + "下一次播放时间是" + this.showtime[i]);

break;

}

}

}

}

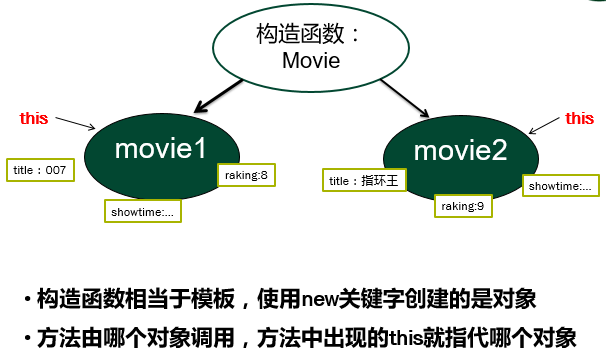
## 使用构造函数创建对象

* 构造函数不能被调用执行
* 通过**new 关键字**来通过构造函数创建对象
* 将要赋给对象属性的值通过构造函数的**参数传递**给对象

//通过构造函数来创建对象

var movie1 = new Movie("007", ["8:00", "13:00", "16:00"]);

var movie2 = new Movie("指环王", ["9:00", "14:00", "19:00"]);



# 内置对象

## JavaScript中的对象

JavaScript中的对象

自定义对象 内置对象 浏览器对象（BOM） 文档对象（DOM）

## 内置对象

### String对象的使用

* String对象处理字符串相关操作
  + 创建String对象，例：

var str = new String(“some string here”);

* 常用操作
  + 获取字符长度（length属性）
  + 判断子串位置（substr方法）
  + 截取子串（indexOf方法）

### Array对象的使用

* Array对象处理数组相关操作
  + 创建Array对象，例：

var movies = new Array(“007”, “指环王”);

* 常用操作
  + 获取数组长度（length属性）
  + 向数组末尾增加元素（push方法）
  + 从数组末尾获取元素（pop方法）

### Date对象的使用

* Date对象**包含了一系列的日期时间处理的功能**
  + 创建Date对象，例： var now= new Date( );
* 常用操作
  + 获取当前日期时间： toLocaleString( )
  + 获取年份、月份、日期：getFullYear()、 getMonth()、getDate()
  + 获取小时、分钟、秒钟： getHours()、 getMinutes()、 getSeconds()

### Math对象的使用

* Math对象包含了一系列的数学运算的功能
  + Math对象不需要创建，直接使用
* 常用操作
  + 四舍五入：Math. round( )
  + 向下取整：Math.floor( )
  + 生成随机数：Math.random( )
  + 取到最大值：Math.max( )
  + 取到最小值：Math.min( )

|  |  |
| --- | --- |
| 属性 | 说明 |
| Math.E | 自然对数的底数，即常量e的值 |
| Math.LN10 | 10的自然对数 |
| Math.LN2 | 2的自然对数 |
| Math.LOG2E | 以2为底e的对数 |
| Math.LOG10E | 以10为底e的对数 |
| Math.PI | ∏的值 |
| Math.SQRT1\_2 | 1/2的平方根 |
| Math.SQRT2 | 2的平方根 |

（1）舍入方法

Math.ceil()执行向上舍入，即它总是将数值向上舍入为最接近的整数；

Math.floor()执行向下舍入，即它总是将数值向下舍入为最接近的整数；

Math.round()执行标准舍入，即它总是将数值四舍五入为最接近的整数；

（2）random（）方法

Math.random()方法返回介于 0 到 1 之间一个随机数，不包括 0 和 1。如果想大于这个范

围的话，可以套用一下公式：

值 = Math.floor(Math.random() \* 总数 + 第一个值)

alert(Math.floor(Math.random() \* 10 + 1)); //随机产生 1-10 之间的任意数

Math.floor(Math.random() \* 10 + 5); //5-14 之间的任意数

# 理解“一切都是对象”

## 字符串变量和字符串对象

* 直接声明一个字符串变量和通过String对象创建的变量，在使用上有什么区别？
  + 声明一个字符串变量，该变量也具有String对象的一系列属性和方法
  + 字符串变量实质就是String对象
* 同字符串一样，数组、布尔、数值**类型的变量**，本质上也都是对象