目录

- 对象：`var obj = {}`
- `Math`对象：`random() max() min() abs() round() ceil() floor() pow() sqrt() Math.PI num.toFixed(2)`
- `Date`对象
- `setInterval(函数,毫秒数);` `clearInterval(返回值编号);`

1.认识对象

1. 编程的发展历史
    - 编程语言：汇编、C语言 面向过程语言
    - 编码语言：Java、C++、Javascript、object-C、Python 面向对象语言
2. 思想
    - 面向过程编程思想：只考虑数学逻辑
    - 面向对象编程思想：直接将生活逻辑映射到我们的程序
        1. 分析有哪些实体
        2. 设计实体的属性功能
        3. 实体间的相互作用
    - 例子：
        - 有一辆车，时速60km/h，一条路1000km
            - 问题：如果让这辆车跑完这条路，需要多长时间
        - 面向过程编程思想：只考虑数学逻辑

```
var hours = 1000 / 60;
alert(hours);
```

        - 面向对象编程思想：
            - 车
                - 属性: speed 60km/h
                - 功能: 可以跑在路上
            - 路
                - 属性: length 1000km
    - 例子2：
        - 贪吃蛇：
            - 面向对象编程思想：
                - 蛇 ： 可以行走，控制方向
                - 食物 ： 随机出现的坐标，不同的食物有不同的功能
                - 砖块 ： 让蛇死亡，有宽高
3. 语法（在Javascript没有 类 这个概念，只有对象，ECMA6新增了类的概念）
    - 类：一类具有相同特征事物的 抽象概念。 (人类，灵长类)

- 对象：具体某一个 个体，唯一的实例。

类	对象
狗	你遇到的那只
电脑	你桌上的那台

4. 代码
   - **声明对象**
     1. 通过`new`运算符声明对象

```
var obj1 = new Object();
```

     2. 省略`new`声明对象

```
var obj2 = Object();
```

     3. 对象常量赋值(对象字面量)

```
var obj3 = {};  // (使用大括号{},代表对象)  (中括号[],代表数组)
```

   - **对象属性** (使用起来和变量没有任何区别)

```
obj3.username = '钢铁侠';
obj3.age = 18;
alert(obj3.username);   //钢铁侠
alert(obj3.age);        //18
```

   - **对象方法** (使用起来和普通函数没有任何区别)

```
obj3.show = function(){
  alert("我的名字叫" + obj3.username);
}
obj3.show();    //括号内传参
```

   - 对象属性 和 对象方法的 其他添加方式
     - 通过中括号添加，中括号填写的必须是字符串

```
obj3['username'] = '钢铁侠';
obj3['username']; // '钢铁侠'

obj3['show'] = function(){
    alert("我的名字叫" + obj3.username);
}
obj3['show']; // "我的名字叫钢铁侠"
```

     - 通过对象常量赋值，在声明时直接赋值 或 按照下面方法赋值

```
obj3 = {
  username:"钢铁侠",
  age: 18,
}
obj3.username; // "钢铁侠"
obj3["age"]; // 18
```

    - 对象的属性和方法可以通过 `obj.属性名` / `obj["属性名"]` 调用
  - 关键字 delete

```
delete obj3.username
```

    - 作用: 删除对象的属性或者方法

5. 数据结构
   - 基本数据类型(存储一个值)
   - 数组(处理批量的数据)
   - 对象(即可以存储数据，又可以存储函数)

2.Math对象

- 在JS中一切皆对象
- 在JS有很多关于数学运算的函数，直接一个 `Math` 对象提供

```
Math.random()       //返回0-1 之间 的随机数 取不到0 和 1
Math.max(num1,num2) //返回较大的数
Math.min(num1,num2) //返回较小的数
Math.abs(num)       //绝对值
Math.round(3.49)    //四舍五入(成整数，只看小数点后一位) 3
Math.ceil(19.3)     //向上取整  20
Math.floor(11.8)    //向下取整  11
Math.pow(x,y)       //x的y次方
Math.sqrt(num)      //开平方    16的开平方是+-4/

// 传入参数是：弧度
Math.sin()  // 正弦    对边与斜边的比
Math.cos()  // 余弦    邻边比三角形的斜边
Math.tan()  // 正切    对边与邻边的比值

Math.PI = 180弧度
1弧度 = Math.PI / 180

数值.toFixed(2);        保留两位小数
```

3.日期对象

- 日期对象的声明
  - 不传参

```
var d = new Date(); // 没有传入参数，默认当前系统时间
```

    - 如果系统时间和现实不相符，获取的就不准确
  - 传参

```
  "2000-01-01/09:21:34"
var d = new Date("2000-01-01/09:21:34");

  "2000/01/01/19:21:34"
var d = new Date("2000/01/01/19:21:34");

// 按照顺序，分别传入参数   年 月 日 时 分 秒 毫秒
  //【注】在国外月份是从0开始数的 0~11
var d = new Date(2000, 0, 1, 9, 21, 34);
```

    - 也可以传入毫秒数

```
// 1秒 = 1000毫秒  (以1970年 1月1日 0:0:0 为参照时间进行换算)
var d = new Date(1000);  //1970/1/1/08:00:01  北京时区要加8小时
```

4.日期对象的格式化方法，了解即可

```
var d = new Date();

//以特定的格式显示 星期几 月 日和年
d.toDateString();    //Fri Jun 26 2020

//以特定的格式显示 时分秒 和 时区
d.toTimeString();    //19:58:47 GMT+0800 (中国标准时间)

//以特定的地区格式显示 星期几 月 日和年
d.toLocaleDateString();//2020/6/26

//以特定的地区格式显示 时分秒 和 时区
d.toLocaleTimeString();//下午8:00:10

//以特定的格式显示完整的UTC日期
d.toUTCString();     //Fri, 26 Jun 2020 12:02:55 GMT
```

- 系统给我们的不一定是最好的，最好是自定义

5.日期对象的方法(重点)

`d.set/getDate()` ： 从Date对象中返回一个**月**中的某一天(1~31)
`d.getDay()` ： 从Date对象返回一**周**中的某一天(0~6) 0是周日
`d.set/getMonth()` ： 从Date对象中返回**月份**(0~11) 0是12月
`d.set/getFullYear()` ： 从Date对象以**四位数**返回**年份**
`d.ste/getHours()` ： 返回Date对象的**小时**(0~23)
`d.set/getMinutes()` ： 返回Date对象中的**分钟**(0~59)
`d.set/getSeconds()` ： 返回Date对象中的**秒数**(0~59)
`d.set/getMilliseconds()` ： 返回Date对象的**毫秒**

`d.set/getTime()` ： 返回1970年1月1日至今的毫秒数
`d.getTimezoneOffset()` ： 返回本地与格林威治标准时间(GMT)的分钟差

- 所有的参照时间点都是1970年1月1日
  - `Date.parse(日期对象)` ： 将当前日期转为毫秒数
  - `d.getTime()` / `d.setTime()`; 将当前日期转为毫秒数，能精确到毫秒

```
alert(Date.parse(d));       //1593253203000
alert(d.getTime());         //1593253203311
```

- 每个set只能修改对应区域

```
d.setTime(1000);    // 1970年1月1日1秒
d.setDate(3);   2020/7/3    // 修改的是当前月的天数，遵循日期规则
d.setMonth(1);  2020/2/     // 从0开始计算月份，同样修改的是当前年的月份
```

- 日期的练习(答案看6_日期的练习)
  - 获取两个日期相差的天数
    - 规定，传入日期格式： `"2000-01-01"` 或 `"2000/01/01"`

6.定时器

- `setInterval(函数,毫秒数);` -> `1000毫秒 = 1秒`
  - 功能：每格对应毫秒数，执行对应函数
  - 返回值：启动定时器，系统分配的编号
- `clearInterval(返回值编号);`
  - 功能：关闭定时器

```
var i = 0;
var timer =  setInterval(show,1000);
function show(){
  //在i等于5的时候，关闭定时器
  if(i == 5){
    clearInterval(timer);
  }
  i++;
  document.write(i + "<br/>");
}
```

    - 精简版

```
var i = 0;
var timer =  setInterval(function(){
  if(i == 5){
    clearInterval(timer);
  }
  i++;
  document.write(i + "<br/>");
},1000);
```

7.定时器实现秒表

- 实现功能：暂停 开始 复位

```
window.onload = function(){
  // 写在这里的代码，是整个页面加载完成以后才会运行的(固定格式)
}
```

- 简化获取ID,封装函数

```
function $(id){
  return document.getElementById(id);
}
$("id名")

//写在行间的onclick升级版
$("start").onclick = function(){ }
```

- 暂停开始整合成一个按钮，通过If和改变判断是否执行定时器