正课:

1. String

2. RegExp

3. Math

4. Date

1. String

切割: 用指定的分隔符将字符串分割为多段子字符串

1. 简单切割: 分隔符是固定的

var arr=str.split("分隔符")

固定套路: 打散字符串为字符数组

var chars=str.split("")

2. 复杂切割: 分隔符不是固定的

var arr=str.split(/正则/i);

2. RegExp

什么是: 专门封装一条正则表达式，并提供用正则表达式执行验证和查找的API

何时: 只要用正则表达式执行验证和查作功能，都要用正则表达式对象

如何:

创建: 2种:

1. 用//直接量: var reg=/正则/ig;

何时: 如果正则是固定的，不需要动态生成

2. 用new创建:

var reg=new RegExp("正则","ig");/\*注意：不是//.这里需要的是字符串，有无数组方法将数组变成字符串 \*/

何时: 如果正则表达式需要动态生成

API:

1. 验证:

var bool=reg.test(str);

用reg验证str的格式是否符合要求

问题: 默认，只要str中部分和reg匹配，就返回true

解决: 凡是验证必须前加^后加$

2. 查找关键词:

var arr=reg.exec(str)

在str中查找下一个符合reg要求的关键词的内容和位置

强调: 一次只能找一个关键词

返回值: arr:[ 0: 关键词, index: 位置 ]

exec可自动找下一个:

每个reg对象上有一个lastIndex属性，开始为0，记录着下次开始查找的位置

每次找到关键词时，exec都自动修改reg.lastIndex=本次的位置+关键词长度

3. Math

什么是: 封装数学计算的常量和API的对象

何时: 只要执行数学相关计算

如何:

创建: 不能new！

所有API和常量都用Math直接调用

API:

1. 取整:

上取整: 只要超过，就取下一个整数

Math.ceil(n)

下取整: 只要超过，就舍弃小数部分

Math.floor(n)

parseInt(str) 能去掉数字后非数字字符(单位)

四舍五入取整: 够5进1，不够舍弃

Math.round(n); n.toFixed(d)

灵活: 只能取整 可按任意小数位数

返回值: number string

2. 乘方和开平方:

乘方: Math.pow(底数,幂)

开平方: Math.sqrt(n)

3. 最大值和最小值:

Math.max(值1,值2,...)

Math.min(值1,值2,...)

问题: 不支持获得数组中的最大值和最小值

解决: Math.max(...arr)

其中: ... ES6中的spread操作，打散数组为单个元素

4. 随机数:

0<=Math.random()<1

公式: min~max

parseInt(Math.random()\*(max-min+1)+min)

5. 反正切

Math.atan(对边长/临边长)

问题: 只能在正负90度之内摇摆

解决: Math.atan2(对边长, 临边长)