正课:

1. \*\*二维数组

2. \*\*\*String

内置对象, 包装类型

3. \*\*String API

4. \*\*\*\*正则表达式

1. \*\*二维数组:

什么是: 数组中的元素，又引用了另一个子数组

何时: 1. 如果要存储的数据本身就是二维数据

2. 对原数组中的数据进行细致分类

如何:

创建: 2种:

1. 先创建空数组，再向空数组中添加子数组

var arr=[];

arr[0]=[...];

arr[1]=[...];

2. 创建数组同时，就初始化子数组;

var arr=[

[0,0,0,0],

[0,0,0,0],

[0,0,0,0],

[0,0,0,0],

]

访问: arr[r][c] 用法和普通变量完全一样

遍历: 外层循环控制行，内层循环控制列:

for(var r=0;r<arr.length;r++){

for(var c=0;c<arr[r].length;c++){

arr[r][c] //当前元素

}

}

2. \*\*\*String

什么是: 由一串文字组成的只读字符数组

vs 数组: 相同: 1. 下标; 2. .length; 3. for遍历; 4. .slice

不同: 类型不同 -> API不通用!

内置对象: ES标准中规定的，浏览器厂商已经实现的对象

包括: 11个:

Number String Boolean -> 包装类型

Array RegExp Date Math

Error

Function Object

Global (在浏览器中被window代替)

包装类型:

什么是: 专门封装原始类型的值，并提供对原始类型值进行操作的API

为什么: 原始类型的值本身什么操作也做不了

何时: 只要试图对原始类型的值调用函数时，都会自动创建该类型的包装类型对象，并调用函数。调用后，包装类型对象自动释放。

转义字符: 2种

1. \ 将字符串中和语法冲突的符号转为正文

2.具有专门功能的特殊符号: \n 回车, \t Tab

路径:

问题: windows中的\ 会被误认为是转义字符

解决: \ 改为\\

问题: \只有windows支持，/才是所有操作系统都支持的统一路径分隔符

解决: 今后只要写路径，必须/

3. \*\*字符串API:

大小写转换:

什么是: 将字符串中的字母统一都转为大写或小写

何时: 只要不区分大小写时,都要先转为一致的大小写，再比较

比如: 验证码, 邮箱, 用户名

如何: str.toUpperCase() 转大写

str.toLowerCase() 转小写

获得指定位置的字符:

str.charAt(i) -> str[i]

获得指定位置字符的unicode号:

str.charCodeAt(i) 获得i位置字符的unicode号

将一个unicode号，反向转回字:

String.fromCharCode(code)

选取指定位置的子字符串:

str.substring() => str.slice()

唯一差别: substring不支持负数参数

解决: 可用str.length-n，表示倒数第n的位置

从当前位置开始，取n个

str.substr(starti,n)

\*\*\*查找关键词:

什么是: 在字符串中查找符合条件的关键词的位置或内容

包括:

1. 查找一个固定的关键词的位置:

var i=str.indexOf("关键词",fromi)

在str中，从fromi位置开始，找下一个"关键词"的位置

强调: 每次只能找一个

返回值: 返回找到的下一个关键词的下标位置i

如果没找到，返回-1

简写: 省略fromi，默认从头开始找第一个关键词

var i=str.lastIndexOf("关键词")

在str中找最后一个关键词的位置

问题: 只能查找一个固定的关键词

解决: 用正则表达式模糊查找符合规则的多种关键词

2. 判断是否包含某类关键词:

var i=str.search(/正则表达式/i);

在str中找第一个符合正则表达式要求的关键词的位置

如果没找到，返回-1

问题: 默认，所有正则区分大小写！

解决: 如果希望不区分大小写时，在第二个/后加i

问题: 1. 仅返回关键词位置，不返回内容

2. 永远只找第一个关键词

3. 获得所有关键词的内容:

var kwords=str.match(/正则表达式/ig);

在str中找所有符合正则表达式要求的敏感词，保存在数组中返回。

如果没找到，返回null!

问题: 默认，正则表达式都只匹配第一个关键词

解决: 只要希望匹配所有，都必须加g

问题: 只能返回关键词内容，无法返回每个关键词位置

4. 即找所有关键词内容，又找所有关键词位置:

\*\*\*替换:

4. \*\*\*\*正则表达式(Regular Expression)

什么是: 规定字符串中字符出现规律的表达式

何时: 2种:

1. 查找关键词: 按规则模糊匹配多种关键词

2. 验证: 用规则验证字符串是否符合要求

如何:

1. 最简单的正则表达式其实就是关键词本身

2. 字符集: 规定字符串中一位字符上可选字符列表的集合

何时: 只要关键词中一位字符上有多种备选字时

如何: [备选字符列表] 不用加逗号分隔

强调: 别选字符再多，一个字符集也只匹配一位字符

简写: 如果备选字符列表中部分字符是连续的，可用-省略中间字符

1位数字 [0-9]

1位大写字母[A-Z]

1位小写字母[a-z]

1位字母[A-Za-z]

1位字母或数字[A-Za-z0-9]

1位汉字[\u4e00-\u9fa5]

3. 预定义字符集: 对常用字符集的简写

包括4种:

\d 一位数字

\w 一位字母/数字或\_

\s 一位空字符: 包括空格，Tab...

. 一位任意字符——通配符

强调: 规则必须和预定义字符集的规则完全匹配时，才能使用预定义字符集。否则，使用普通字符集。

4. 量词: 规定一个字符集出现次数的规则

为什么: 单靠字符集，无法灵活固定字符出现的次数

何时: 只要规定一个字符集出现的次数

如何: 紧跟在修改的字符集之后使用

包括: 2大类:

1. 有明确数量边界: 3组

字符集{m,n} 至少m次，最多n次

字符集{m,} 至少m次，多了不限！

字符集{m} 必须m次

2. 没有明确边界: 3个

字符集\* 可有可无，多了不限

字符集+ 至少一个，多个不限

字符集? 可有可无，最多一个

问题: 默认，一个量词只修改相邻的前一个字符集

5. 选择和分组:

选择: 或 | 规则1|规则2 只要匹配任意一个规则即可

分组: (多个规则) 将多个规则分为一组

为什么: 一个量词默认值修改一个字符集

何时: 只要希望一个量词修改多个字符集时

身份证号: 15位数字 2位数字 1位数字或Xx

可有可无,最多一次

\d{15}(\d\d[0-9Xx])?

手机号:

+86或0086 可有可无，最多一次

空字符 可有可无，多了不限

1

在3,4,5,7,8中选一个

9位数字

(\+86|0086)?\s\*1[34578]\d{9}

微 信 wei xin w x

(微|w(ei)?)\s\*(信|x(in)?)

6. 指定匹配位置:

开头: ^ 比如: ^\s+ 匹配开头位置的空字符

结尾: $ 比如: \s+$ 匹配结束位置的空字符

单词边界: \b 包括: ^ $ 空格 标点符号

何时: 只要要求匹配一个单词时