正课:

String

1. 获得指定位置的字符

2. 获取子字符串

3. \*\*\*查找关键词

4. \*替换

5. \*切割

\*\*\*\*正则表达式

String

1. 获得指定位置的字符: ——了解

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

php: substr($str, i, 1) 获得$str中i位置的1个字符

获得指定位置字符的unicode号:

str.charCodeAt(i)

将unicode号反向转回字:

String.fromCharCode(unicode号)

2. 获取子字符串:

str.substring(starti,endi+1) => str.slice(starti,endi+1)

substring vs slice

相同: 1. 含头不含尾

2. 省略第二个参数，表示一直选取到结尾

不同: 不支持负数参数！

变通: 用str.length-n 代替 -n

str.substr(starti,n) 根据个数，获取子字符串

php: substr($str, i, n) 获得$str中i位置开始的n个字符

简写: i支持负数参数

省略n表示到结尾

3. 查找关键词: 4种:

1. 查找一个固定的关键词出现的位置:

var i=str.indexOf("关键词",开始位置)

在str中，从“开始位置”开始，查找下一个"关键词"的位置

返回值: 下一个"关键词"的第一个字的下标位置

如果找不到了，返回-1

如果省略第二个参数-开始位置: 默认从0开始找

查找最后一个关键词的位置:

var i=str.lastIndexOf("关键词")

问题: 1. 区分大小写!

解决: 在indexOf之前，先将str转为小写，再将关键词也转为小写，再查找

问题: 2. 只能查找一个固定的关键词!

解决: 正则表达式

php: strpos -> indexOf

stripos 忽略大小写版本的indexOf

strrpos -> lastIndexOf

strripos 忽略大小写的lastIndexOf版本

$i=stripos(完整字符串,关键词[,开始位置])

2. str.search():

什么是: search是用正则表达式查找str中第一个匹配的关键词的位置

何时: 只要判断一句话中\*是否包含\*符合正则要求的敏感词时

如何: var i=str.search(/正则表达式/);

返回值: 返回第一个找到的匹配的关键词的位置

如果没找到返回-1

变通: 验证: 只要将正则表达式，前加^后加$，search也可做验证

问题: 1. 永远只能找第一个

2. 只能返回位置，无法返回关键词内容

3. str.match():

什么是: 查找str中所有和正则表达式匹配的关键词的内容

何时: 只要获得敏感词的\*内容\*时

如何: var arr=str.match(/正则表达式/)

返回值: 包含所有找到的敏感词内容的数组

如果找不到: 返回null!

强调: 如果一个函数有可能返回null，就必须先验证不是null，再使用!

问题: 1. 默认正则是区分大小写的！

解决: 在第二个/后加i, 意为忽略: ignore

2. 默认正则只匹配第一个敏感词！

解决: 在第二个/后加g, 意为全部: global

缺: 只能获得每个关键词的内容，无法获得每个关键词的位置

4. 即查找每个关键词的内容，又查找每个关键词的位置:

String做不了!

reg.exec()

4. 替换:

什么是: 将找到的敏感词替换为指定的内容

如何: 2种:

1. 简单替换: 将所有敏感词都替换为统一的替换值

str=str.replace(/正则表达式/,"替换值")

强调: replace无权直接修改原字符串，只能返回新字符串，所以，必须用变量接住替换后的新内容。

2. 高级替换: 根据每个敏感词的不同，动态选择对应的替换值

str=str.replace(/正则表达式/,function(kword){

//kword: 自动获得本次找到的敏感词

return 根据kword的不同，选择不同的替换值返回;

})

衍生: 删除: 将找到的敏感词，替换为“”

5. 切割:

什么是: 将字符串，按某种子字符串，分隔为多段

如何: 2种:

1. 简单切割: 切割符是一个固定不变的字符串

var substrs=str.split("切割符");

强调: 切割后的结果中，不包含切割符

2. 复杂切割: 切割符可能有多种情况

var substrs=str.split(/正则表达式/)

\*\*\*\*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. 用规则，模糊匹配多种敏感词时

2. 用规则验证字符串的格式: 密码强度，邮箱格式，日期。。。

如何: 语法:

1. 最简单的正则就是关键词原文本身

2. 字符集:

什么是: 1位字符上的备选字符的集合

何时: 只要一位字符上有多种可能的备选字时

如何: [备选字符列表]

强调: 1. 不能加逗号分隔

2. 每个[]，只能匹配一位字符

简写: 如果字符集中部分字符的unicode号是连续的，可用-省略中间字符

常用: [0-9] 一位数字

[a-z] 一位小写字母

[A-Z] 一位大写字母

[A-Za-z] 一位字母，大小写都行

[0-9A-Za-z] 一位字母或数字

[\u4e00-\u9fa5] 一位汉字

车牌号: [\u4e00-\u9fa5] [A-Z] [A-Z0-9] [A-Z0-9] [A-Z0-9] [A-Z0-9] [A-Z0-9]

除了: [^74]

匹配范围太大！

比如匹配HTML元素的开始标签: <a[^>]+>

3. 预定义字符集:

什么是: 对常用字符集的简写: 固定4种:

包括: \d 1位数字

\w 1位字母，数字或\_

\s 1位空字符: 空格，tab，...

. 一切字符

强调: 只有规则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

4. 量词:

什么是: 专门规定一位字符集出现的次数

何时: 今后只要规定字符集的出现次数时，必须用量词

如何: 字符集量词

强调: 量词默认只修饰相邻的前一个字符集

包括: 2大类:

1. 有明确数量:

{n,m} 至少n个，最多m个

{n,} 至少n个，多了不限

{n} 必须n个，不能多也不能少

2. 没有明确数量:

? 可有可无，最多1个

\* 可有可无，多了不限

+ 至少1个，多了不限

5. 选择和分组:

选择: 等效于程序中的或

何时: 只要在两组规则间任意匹配其一即可

如何: 规则1|规则2

分组: 将多个子规则用()，包裹为一组

何时: 2种:

1. 只要一个量词需要同时修饰多个子规则，必须将多个子规则包裹为一组

2. 需要获得关键词中的部分子内容时

如何: (规则1...)

比如:

身份证号:

15位数字: \d{15}

2位数字: \d{2}

1位数字或X,x : [0-9Xx]

后三位整体: 可有可无，最多一次

手机号:

+86或0086 (\+86|0086)

至少一个空字符 \s+

地区代码和空字符整体 可有可无，最多1次

1

必须在3,4,5,7,8之间选一个 [34578]

9位数字 \d{9}

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

"微信" "微 信" "weixin" "wei xin" "w x" "wechat"

(微|w(ei)?)\s\*(信|x(in)?)

6. 匹配特殊位置: 3个位置:

开头 ^

结尾 $

何时: 只要要求匹配的关键词必须出现在开头或结尾时

比如: 只匹配开头的空字符: ^\s+

只匹配结尾的空字符: \s+$

即匹配开头，又匹配结尾: ^\s+|\s+$

正则表达式的问题:

如果验证时，只要部分匹配，就会返回true！

解决:

只要验证，必须同时前加^后加$，表示必须从头到尾完整匹配。

单词边界\b:

何时: 专门用于匹配一个独立的单词

包含: 空格，开头，结尾，标点符号