张东

zhangdong@tedu.cn

准备:

1. 犀牛书: JAVASCRIPT权威指南

2. 微信公众号: 前端大全

3. 上届笔记和思维导图:

谁笔记记的勤，谁学的越烂！

正课:

1. String:

2. \*\*\*正则表达式:

1. String

什么是: 由多个字符组成的只读字符数组

vs 数组:

相同: 1. 下标, 2. .length, 3. for循环遍历

4. slice 选取子内容

不同: 类型不同! API不通用!

API: 所有API无权直接修改字符串的内容，只能返回新字符串。

大小写转换:

toUpperCase()

toLowerCase()

何时: 只要不区分大小写时，都要先转为一致的大小写，再比较或判断

获取指定位置的字符:

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

获取指定位置字符的unicode号

var unicode号=str.charCodeAt(i)

将一个unicode号反向转回字

var 字=String.fromCharCode(unicode号)

选取子字符串:

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

区别: substring不支持负数参数

str.substr(starti,n)

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"[,fromi=0]);

从fromi位置向后，找str中下一个关键词的位置i

返回值: 关键词第一个字的位置i

如果找不到，就返回-1

var i=str.lastIndexOf("关键词")

查找str中最后一个关键词的位置

问题: 只能查找一种固定的关键词

2. 用正则判断是否包含关键词:

var i=str.search(/正则/i)

在str中查找第一个和正则匹配的关键词的位置i

返回值: 第一个关键词的位置

如果找不到，返回-1

问题: 1. 只能返回位置，无法返回内容

2. 只能找第一个，无法找之后的关键词

3. 查找所有关键词的内容:

var kwords=str.match(/正则/ig);

返回所有关键词的数组

如果找不到，返回null

\*\*\*如果将来一个函数，有可能返回null，必须先验证不是null，再使用！

kwords.length表示关键词的个数

问题: 1. 正则表达式默认只匹配第一个关键词

解决: 第二个/后加后缀g global

2. 正则表达式区分大小写

解决: 第二个/后加后缀i ignore

3. 只能获得关键词内容，无法获得每个关键词位置

4. 即查找每个关键词的内容，又查找每个关键词的位置:

?

替换: 将查找到的关键词，替换为指定的新内容

2种:

1. 简单替换: 将所有关键词都替换为统一的新词

str=str.replace("关键词"或/正则/g,"新值");

用指定的新值，替换找到的关键词

问题: 1. replace无权修改原字符串

解决:必须通过赋值的方式替换原字符串的内容

2. 正则默认只查找和替换第一个关键词

解决: g

3. 默认区分大小写

解决: i

2. 高级替换: 根据关键词的不同，动态选择不同的新词替换。

str=str.replace(/正则/ig,function(kword){

return 根据kword的不同,动态选择不同的值

})

切割:

2. 正则表达式:(regular expression)

什么是: 规定一个字符串中字符出现规律的表达式

何时: 1. 用规则模糊匹配多种关键词

2. 用规则验证字符串的格式

如何:

1. 其实最简单的规则，就是关键词原文

2. 字符集:

什么是: 规定一位字符上的备选字符列表

何时: 只要一位字符上，有多种备选字

如何: [备选字列表]

强调: 一个字符集无论包含多少备选字符，都只能匹配一个字符。多选一!

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

比如: 1位数字: [0-9]

1位小写字母: [a-z]

1位大写字母:[A-Z]

1位字母:[A-Za-z]

1位字母或数字: [0-9A-Za-z]

1位汉字: [\u4e00-\u9fa5]

比如: 车牌号:

[\u4e00-\u9fa5][A-Z]·[A-Z0-9][A-Z0-9][A-Z0-9][A-Z0-9][A-Z0-9]

"除了": [^xx]

3. 预定义字符集:

对常用字符集的更简化写法:

\d => [0-9] 一位数字

\w => [0-9A-Za-z\_] 一位字母数字或下划线

\s => 一切空字符: 空格 tab ...

. => 通配符

4. 量词:

专门固定一位字符出现次数的规则

何时: 今后只要规定一位字符出现的次数

如何: 字符集量词

包括: 2大类：

1. 有明确数量边界的:

字符集{n,m} 至少n个，最多m个

字符集{n,} 至少n个，多了不限

字符集{n} 必须n个，不能多也不能少

2. 没有明确数量边界:

字符集? 可有可无, 最多一个

字符集\* 可有可无，多了不限

字符集+ 至少一个, 多了不限

5. 选择和分组:

选择: 或 |

何时: 只要在两个规则之间任选其一匹配即可

如何: 规则1|规则2

分组: 用()将多个子规则分为一组

为什么: 量词默认只修饰相邻的前一个字符集

何时: 如果希望量词可同时修饰之前的多个字符集

如何: (规则1规则2)量词

身份证号:

15位数字 2位数字 1位数字或Xx

后三位整体可有可无,最多1次

\d{15} (\d{2} [0-9Xx])?

手机号:

0086或+86 整体可有可无，最多一次

空字符 可有可无，不限个数

1

[34578]

9位数字

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

微信 weixin wei xin wx

微 或 wei (其中ei可有可无,最多一个)

空字符 可有可无，数量不限

信 或 xin(其中in可有可无,最多一个)

(微|w(ei)?)\s\*(信|x(in)?)

7. 匹配特殊位置:

字符串中，开头和结尾最重要:

^ 开头

$ 结尾

如何: ^规则 匹配开头的xxx

规则$ 匹配结尾的xxx

固定套路:

1. 开头的空字符: ^\s+

2. 结尾的空字符: \s+$

3. 开头和结尾的空字符: ^\s+|\s+$

英文中特殊位置: 单词边界

包括: 空格 标点 开头和结尾

\b

何时: 只要匹配单个单词时