**张东**

**zhangdong@tedu.cn**

**准备:**

**1. 书: 犀牛书——JavaScript权威指南**

**2. 微信公众号: 前端大全**

**3. 上届笔记——打印**

**禁止记笔记！**

**正课:**

**1. 正则表达式:**

**2. String的正则API:**

**3. RegExp对象**

**1. 正则表达式（Regular Expression）:**

**什么是: 规定字符串中字符出现规律的规则**

**何时: 2种:**

**1. 模糊查找多种关键词：**

**2. 表单验证: 格式验证：**

**如何:**

**1. 最简单的正则表达式其实是关键词本身**

**2. 字符集:**

**什么是: 规定一位字符上备选字符列表的集合**

**何时: 如果一位字符，有多种备选字时**

**如何: [备选字列表]**

**匹配时: 只要这一位置上的字符，匹配中括号中其一，就算符合规则。**

**简写: 如果[]内部分字符是连续的，可用-省略中间字符**

**比如: [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**

**[0-9A-Za-z] 一位字母或数字**

**[\u4e00-\u9fa5] 一位汉字**

**车牌号: 一个汉字一位大写字母·5位大写字母或数字**

**[\u4e00-\u9fa5][A-Z]·[A-Z0-9]{5}**

**3. 预定义字符集:**

**什么是: 对常用字符集的一种简写:**

**包括: \d 一位数字 => [0-9]**

**\w 一位字母或数字或\_ => [0-9A-Za-z\_]**

**\s 一位空字符: 空格，Tab，...**

**. 通配符**

**4. 量词:**

**什么是: 专门规定一位字符集出现次数的规则**

**何时: 只要规定一位字符集出现的次数**

**如何: 2种:**

**1. 有明确数量边界的: 3组:**

**{n,m} 至少n次，最多m次**

**{n,} n次以上，多了不限**

**{n} 必须n次**

**2. 没有明确数量边界的: 3组:**

**? 可有可无，最多一次**

**\* 可有可无，多了不限**

**+ 至少一个，多了不限**

**5. 选择和分组:**

**分组: 将多个字符集分为一组**

**为什么: 量词默认仅修饰相邻的前一个字符集**

**何时: 希望量词同时修饰多个字符集时**

**如何: (多个字符集)**

**选择: 或 在多个子规则中，选其一匹配即可**

**如何: 规则1|规则2**

**身份证号: 15位数字 2位数字 1位数字或X或x**

**后三位整体可有可无，最多一次**

**\d{15}(\d{2}[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**

**3，4，5，6，7，8 中选一个**

**9位数字**

**(\+86|0086)?\s\*1[3-8]\d{9}**

**6. 匹配特殊位置:**

**字符串开头 ^ 比如: 开头的空字符: ^\s+**

**字符串结尾 $ 比如: 结尾的空字符: \s+$**

**何时: 只要仅匹配开头或结尾的关键词时**

**再比如: 同时匹配开头或结尾的空字符: ^\s+|\s+$**

**单词边界 \b**

**包括: 开头，结尾，空格，标点符号**

**何时: 2种:**

**1. 仅匹配单词时 比如: \bno\b 仅匹配单词no**

**2. 仅匹配单词中特定位置的字符**

**比如: 每个单词首字母: \b[a-z]**

**作业: 微信，weixin wei xin w x**

**作业: 百度上找一个简单的你能看懂的电子邮件格式，记住！**

**2. String的正则API:**

**1. 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",starti)**

**在str中，从starti位置开始，找下一个"关键词"的位置i**

**如果找不到，返回-1**

**var i=str.lastIndexOf("关键词")**

**在str中，找"关键词"最后出现的位置**

**问题: indexOf不支持正则，无法模糊匹配多种关键词**

**2. 模糊查找一个关键词出现的位置:**

**var i=str.search(/正则表达式/i)**

**在str中从头开始查找符合/正则表达式/要求的"关键词"的位置i**

**如果找到，返回关键词的下标，否则，返回-1**

**问题1: 正则表达式默认区分大小写**

**解决: 在正则表达式第二个/后加后缀i, 表示ignore忽略大小写**

**问题2: 只能获得关键词的位置，无法获得关键词的内容**

**3. 获得关键词的内容:**

**var arr=str.match(/正则表达式/i)**

**在str中找第一个符合/正则表达式/要求的关键词的内容和位置**

**返回值: 如果找到，返回一个数组:**

**[ "0": "关键词" , "index": i ]**

**如果找不到，返回null**

**强调: 如果一个函数有可能返回null，则必须先验证不是null，才能用！**

**问题: 如果一句话中包含多个敏感词，但正则默认仅匹配一个**

**解决: 在第二个/后加后缀g, 表示global全部**

**但是: 只能返回所有关键词的内容，无法返回关键词的位置**

**4. 即查找每个关键词的内容，又查找每个关键词的位置**

**reg.exec()**

**3. 替换: 2种:**

**1. 简单替换: 将所有关键词都替换为相同的新值**

**str=str.replace(/正则表达式/g, "新值")**

**将str中符合/正则表达式/要求的关键词，都替换为统一的新值**

**2. 高级替换: 根据每次找到的关键词的不同，动态选择不同的新值替换**

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

**//没找到一个关键词，就自动调用一次函数**

**//每次调用时，自动传入本次找到的关键词给kword**

**//根据kword不同，返回不同的新值！**

**})**

**衍生操作: 删除: 将关键词替换为""**

**3. 切割: 将一个字符串，按指定字符切割为多段子字符串**

**2种:**

**1. 简单切割: 分隔符是固定的情况:**

**arr=str.split("切割符")**

**固定套路: 将字符串打散为字符数组**

**arr=str.split("")**

**2. 高级切割: 分隔符不是固定的:**

**arr=str.split(/正则表达式/)**

**3. RegExp对象:**

**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: 今后只要在程序中保存和使用正则表达式，都用正则表达式对象**

**如何:**

**创建: 2种:**

**1. 用/创建: var reg=/正则表达式/ig**

**何时: 如果正则表达式是一成不变的。**

**2. 用new RegExp创建: var reg=new RegExp("正则表达式","ig")**

**何时: 如果正则表达式不是固定的，需要动态生成**

**API:**

**1. 验证:**

**2. 查找:**