1. 什么是正则表达式?

正则表达式是提供了一种在文本中进行搜索和替换的强大的方式的模式。

在 JavaScript 中，我们可以通过 RegExp 对象使用它们，也可以与字符串方法结合使用。

正则表达式(Regular Expression)是一种**文本模式**，包括普通字符（如，a-z,A-Z,0-9等）和特殊字符（称为"元字符"，如：+,?,*等）。

正则表达式使用单个字符串来描述、匹配一系列匹配某个句法规则的**字符串。**

正则表达式是繁琐的，但它是强大的，学会之后的应用会让你除了提高效率外，会给你带来绝对的成就感。

许多程序设计语言都支持利用正则表达式进行字符串操作。C#， java, PHP

正则表达式使用场景：校验字符串是否满足正则表达式条件；如：表单验证

- 登录校验
- 注册校验
- 验证邮箱
- 验证手机
- 验证密码强度
- 等等...

2. JS中如何定义正则表达式?

定义两种方式：字面量和RegExp()对象
如：

```
var reg1 = /[0-9]/;      // 字面量定义方式，推荐使用
var reg2 = new RegExp('[0-9]', 'i');
```

RegExp对象及其API参考：
https://developer.mozilla.org/zh-CN/docs/Web/JavaScript/Reference/Global_Objects/RegExp

正则的方法

- test()
- exec()

3. 正则表达式语法? （重点）

[0-9]匹配0-9之间的单个数字，其中[]表示范围
[a-z]匹配a-z之间的单个字母
[A-Z]匹配A-Z之间的单个字母

+ 匹配1个或多个字符（1次或多次）
? 匹配0个或一个字符（0次、或1次）
* 匹配0个或多个字符（0次、或1次、或多次）
^ 匹配以xxx字符开头，特别注意在[]中使用时表示取反操作。
$ 匹配以xxx字符结尾
{} 表示匹配的字符串的长度范围 {2, 6} 相当:str.length >= 2 && str.length <= 6

() 表示一个整体.
| 表示或者 or

\s 匹配任何空白字符 ''，包括空格(按下空格键)、制表符、换页符等等。等价于 [\f\n\r\t\v]。
\S 匹配任何非空白字符。等价于 [\^\f\n\r\t\v]
\w 匹配数字,字母,下划线 ====[a-zA-Z0-9*]*
\W 匹配非数字,非字母,非下划线 ====[\^a-zA-Z0-9]
\d 匹配一个数字字符。等价于 [0-9]。
\D 匹配一个非数字字符。等价于 [\^0-9]。

修饰符

g【global】全局
i【ignore】忽略大小写

以下的了解：

\ 转义字符 如：.
\f 匹配一个换页符(了解) PageDown下一页 PageUp上一页
\n 匹配一个换行符 (软换行,自动换行,常用)
\r 匹配一个回车符 (强换行,相当于按下enter键,常用)
\t 匹配一个制表符。(了解,相当于按下tab键)
\v 匹配一个垂直制表符。(了解)
.....
各种语法可以组合使用。

详细语法参考文档：

https://www.runoob.com/regexp/regexp-syntax.html

匹配中文字符： [\u4e00-\u9fa5]

断言[必须包含]：?=.*

4. 正则表达式模式? （了解）

贪婪模式 * 和 +、懒惰模式 ？

https://www.cnblogs.com/study-everyday/p/7426862.html

5. 常用的正则表达式

https://blog.csdn.net/ZYC88888/article/details/98479629
http://c.runoob.com/front-end/854

6. 字符串的方法

- split() 【重要】
- replace() 【重要】
- replaceAll()
- match() 【重要】
- matchAll()

- search()

7.正则的方法

- test()【重要】
- exec()