1，什么是正则表达式：

正则表达式是对字符串操作的一种逻辑公式，就是用事先定义好的一些特定字符，及这些特定字符的组合，组成一个“规则字符串”，这个“规则字符串”用来表达对字符串的一种过滤逻辑。

2，使用正则表达式的目的。

1. 给定的字符串是否符合正则表达式的过滤逻辑（称作”匹配”）
2. 可以通过正则表达式，从字符串中获取我们想要的特定部分。

3，正则表达式的特点：

1. 灵活性，逻辑性和功能性非常的强。
2. 可以迅速地极简单的方式达到字符串的复杂控制。

**4，创建正则表达式的语法：**

第一种方法：

var reg=/设定的规则/;

第二种方法：

var reg = new RegExp（‘设定的规则’）;

RegExp对象的常用方法：

|  |  |
| --- | --- |
| exec（str） | 检索字符串中指定的值。返回找到的值，并且确定其位置。 |
| test（str） | 检索字符串中是否满足指定条件的值。返回true或者false。 |

**5，正则表达式所支持的常用通配符：**

|  |  |
| --- | --- |
| **.** | 可以匹配任何字符 |
| **\d** | 匹配0-9的所有数字 |
| **\D** | 匹配非数字 |
| **\s** | 匹配所有空白字符，包含空格，制表符，换行符，回车符等。 |
| **\S** | 匹配所有非空白字符。 |
| **\w** | 匹配所有的单词字符，包含0-9数字，26个英文字母和下划线 |
| **\W** | 匹配所有的非单词字符 |
| **\b** | 匹配单词边界 |
| **\B** | 匹配非单词边界 |

|  |  |
| --- | --- |
| **[abc]** | 查找方括号之间的任何字符 |
| **[^abc]** | 查找任何不在方括号之间的字符 |
| **[0-9]** | 查找任何从0至9的数字 |
| **[a-z]** | 查找任何从小写a到小写z的字符 |
| **[A-Z]** | 查找任何从大写A到大写Z的字符 |
| **[A-z]** | 查找任何从大写A到小写z的字符 |
| **（red|blue|green）** | 查找指定的选项 |

**7,正则表达式量词：**

|  |  |
| --- | --- |
| n+ | 匹配任何包含至少一个n的字符串 |
| n\* | 匹配任何包含零个或多个n的字符串 |
| n？ | 匹配任何包含零个或一个n的字符串 |
| n｛X｝ | 匹配包含X个n的序列的字符串 |
| n｛X,Y｝ | 匹配包含X或Y个n的序列的字符串 |
| n｛X，｝ | 匹配包含至少X个或X个1以上n的序列的字符串 |
| n$ | 匹配任何结尾为n的字符串 |
| ^n | 匹配任何开头为n的字符串 |

n为具体要判断的字母或数字.

**8，正则表达式的修饰符：**

|  |  |
| --- | --- |
| i | 执行对大小写不敏感的匹配 |
| g | 执行全局匹配（查找所有匹配而非在找到第一个匹配后停止） |
| m | 执行多行匹配 |