正则表达式

1. 基础介绍：

jump:匹配”jump” jump(ing)?匹配 jump 或 jumping ？匹配 0 或 1 次

保留字：[()*+?{}^$.|/]

如果想搜索这些字符中一个，需用反斜杠\转义

代码中\.将显示为\\.，表定义了字符串”\.”

截获圆括号：3(pm|am)会匹配，当需选择性匹配特定字符串时，如文中查”November”,但可能简写 Nov,则可定义模式 Nov(ember)表括号内容可选

字符组：相当于一组字符中匹配单个字符。如 t[aeiou]匹配 te 或 ti，10[0-9]相当于 10[0123456789]，但更简洁，匹配 100，101，t[^0]匹配包含 t 并且后面紧跟非 0 字符

操作符：|或 *匹配 0 或多次 +匹配 1 或多次优先多次 ?匹配 0 或 1 次优先 1 次 *?匹配 0 或多次 +?匹配 1 或多次优先 1 次 ??匹配 0 或 1 次优先 0 {n} {n}? {n}+ {n,} {n,m} {n,m}? 对于”0000” “0+?”匹配单个 0 而“0＋”匹配 0000

Anchors: ^行始 $行止 \A \Z \z \

.匹配任一字符,如 P.P 匹配 POP 等,但不包括\r,\n 如果要匹配全部字符[\s\S]

\d 匹配数字[0-9]，例:\d\d:\d\d 可匹配 9:30 或 12:45 等

\b 匹配空格、标点 例:空格换行 to \b 会匹配 to the moon 或 to !在单词匹配方便

\s 匹配空白字符，如空格: hello\s 匹配 “hello “

{}包含匹配最大和最小数，10{1,2}1 会匹配 101，或 1001

He[LI]{2,}会匹配 HeLLO 和 HeLLIO 等

2. 用法示例

```
//纯数字判断(正则表达式与NSPredicate连用)
- (BOOL)validateNumber:(NSString *)textString{
   NSString *number = @"^[0-9]+$";
   NSPredicate *predicate = [NSPredicate predicateWithFormat:@"SELF MATCHES %@",number];
   return [predicate evaluateWithObject:textString];
}
```

```objc
//手机号验证(NSString方法)
- (BOOL)validatePhoneNumber:(NSString *)textString{
    NSRange range = [textString rangeOfString:@"^1[3]\\d{9}$" options:NSRegularExpressionSearch];
    if (range.location != NSNotFound) {
        return YES;
    }
    return NO;
}
//获取字符串中的数字(NSString方法)
- (NSString *)getNumbersOfString:(NSString *)textString{
    NSRange range = [textString rangeOfString:@"[0-9]+" options:NSRegularExpressionSearch];
    if(range.location != NSNotFound){
        return [textString substringWithRange:range];
    }
    return nil;
}

//纯数字判断(正则表达式类)
- (BOOL)isValidateNumber:(NSString *)textString{
    NSError *error = NULL;
    NSRegularExpression *regex = [NSRegularExpression regularExpressionWithPattern:@"^\\d+$" options:NSRegularExpressionCaseInsensitive error:&error];
    NSTextCheckingResult *result = [regex firstMatchInString:textString options:0 range:NSMakeRange(0, [textString length])];
    if(result){
        NSLog(@"%@",[textString substringWithRange:result.range]);
        return YES;
    }
    return NO;
}
```

```objc
//获取字符串的数字匹配所有结果(正则表达式类)
- (NSArray *)filterdigistsWithString:(NSString *)textString{
    NSError *error = NULL;
    NSMutableArray *tempArr = [NSMutableArray array];
    NSRegularExpression *regex = [NSRegularExpression regularExpressionWithPattern:_pattern.text options:NSRegularExpressionCaseInsensitive error:&error];
    NSLog(@"%@", _pattern.text);
    for(NSTextCheckingResult *result in [regex matchesInString:textString options:0 range:NSMakeRange(0, textString.length)]){
        [tempArr addObject:[textString substringWithRange:result.range]];
    }
    return [NSArray arrayWithArray:tempArr];
}
```

3. 零宽断言

(1) 简介

零宽断言用于查找在某些内容(但不包括这些内容)之前或之后但东西，也就是说指定一个位置，这个位置应满足一定条件(断言)，断言用来声明一个应为真的事实。正则表达式只当断言为真时才会继续进行匹配。

(2) 几种断言理解

1) 零宽度正预测先行断言(?=exp)匹配符合 exp 前的位置

例：[a-z]*(?=ing)可以匹配 cooking 和 singing 中的 cook 和 sing

注：先行断言执行步骤为先从匹配字符串最后面表达式，如不匹配往右继续查第二个 ing

2) 零宽度正回顾后发断言(?<=exp)匹配 exp 后的位置

例：*(?=ing)匹配 cookingsinging 中的 cookingsing，而后发断言(?<=ing 匹配 singing)

注：先从最左端找到第一个 ing，再匹配后面表达式，如不匹配，往右查第二个 ing

3) 零宽度负预测先行断言(?!exp)匹配 exp 前的位置

例：\d{3}(?!\d)匹配后面不是数字的三位数字

4) 零宽度负回顾后发断言(?<!exp)匹配前面不是 exp 的位置

例：(?<![a-z])\d{7}匹配前面不是小写字母的七位数字

口诀：先行由后往前匹配前面正则，后发由前往后匹配后面正则；正预测正回顾?=负预测负回顾?!