**String类的知识点**

1. String(字符串常量)---不可变的，每次对string类型改变等同于生成一个对象，造成jvm工

作负荷增大，导致速度变慢，所以一旦对string类型改变次数较多时，不建议使用string类型。

StringBuffer(字符串变量，线程安全)---stringbuffer每次结果都会对对象本身进行操作，

相对String较快，但在字符串拼接时，String速度上又占优势

StringBuild(字符串变量，非线程安全)---1.5以上新增的，被设计用作stringbuffer的一个简单替换，速度相对stringbuffer较快

1. String是一个Final修饰的类，不能被继承改变，字符串传递的是它的引用，其值本

身无法改变，每一个看起来会修改String值得方法实际上都是创建一个全新的String对象；

1. 持有成员变量
2. String类型是一个持有私有访问权限的字符串数组（char[]），它实现Serializable拥有序列化id
3. String中的字符串是连续不可变的，并通过实例构造器将字符串编译为String类型的实例对象
4. String类型持有一个整型，作为hash值
5. 实例构造器
6. 无参构造和有参构造

有参构造实际上同时对字符串和字符串hash地址进行实例化

public String(String original){

this.value=original.value;

this.hash=original.hash;

}

1. String重载“+”和StringBuilder
2. 用于String的“+”和“+=”是Java中仅有的两个重载的操作符，“+”能将字符

串拼接

1. 格式化输出
2. javase引用format方法，类似于print
3. 例. Formatter f=new Formatter(System.out);

f.format(“Row 1:[%d%f]\n”,x,y)

将x和y分别插到%d和%f的位置,x和y使用逗号分别打印，不能使用print方法

1. Formatter类

Formatter是一个类，他将你的格式化字符与数据翻译成你需要的结果

创建一个Formatter对象时，必须向其构造器传递一些信息

new Formatter(System.out)

1. 格式化说明符

例. 控制空格与对齐

format("%-15s %5s %10s\n", "Item", "Qty", "Price");

常用转意字符

b---boolean值；e ---浮点数（科学技术）；x---整数（十六进制）

h---散列码（十六进制）

e. String.format()----静态方法，类似Formatter.formatter(),返回String对象

1. 正则表达式
2. String类内建功能

matches(“规则”)

例.System.out.println("-1234".matches("-?\\d+"));

-?: 开头为一个或零个负号

\d: 范围为[0~9]

+: 表示有多个数

spilt(“规则”)-----将字符串按规则分开，需要用数组来接收

例. System.out.println(Arrays.toString(str.split(regex)));

补充：String.spilt()存在一个重载的版本，允许限制分割次数

replaceFirst(“正则”,”替换字符”)

例. System.out.println(str.replaceFirst("f\\w+", "located"));

以字母f开头，后面跟一个或多个字母（注意这里的w是小写的）并且只替换掉第一个匹配的部分，所以“found”被替换成“located”

1. 逻辑操作符和边界操作符

逻辑操作符

XY Y跟在X后面 X|Y X或Y （X） 捕获组（capturing group） 可以在表达式中用\i引用第i个捕获组。

边界匹配符

^ 一行起始 \B 非词的边界 $ 一行结束 \G 前一个匹配结束 \b词的边界

1. 量词

例如：abc+

看起来它似乎应该匹配1个或多个abc序列，如果我们把它应用于输入字符abcabcabc则实际上会获得3个匹配。然而，这个表达式实际上表示的是：匹配ab,后面跟随1个或多个c.要匹配1个或多个完整的abc字符串，我们必须这样表示：（abc）+

1. pattern和matcher

Pattern.compile(“正则”) 编译正则，生成pattern对象

Matcher()是pattern对象的方法，用于检索匹配

例. Pattern p = Pattern.compile("\\w+");//将字符串划分为单词

Matcher m = p.matcher("Evening is full of the linnet's wings")

m.find(“可接收整数”): matcher的一个方法，遍历输入的字符串，可接收整数作为参数，表示字符串中字符的位置，但每次查找完成后不能再次进行第二次查找，若要进行第二次查找则需要使用reset（）方法刷新操作，将现有的matcher对象应用于一个新的字符序列

m.groups()： 可根据组的编号来引用某个组，可接受整数作为参数，代表组号。

start（）和end（）分别返回先前匹配的起始位置的索引和最后的索引加一