Guava学习之Splitter

Splitter:在Guava官方的解释为：Extracts non-overlapping substrings from an input string, typically by recognizing appearances of a separator sequence. This separator can be specified as a single character, fixed string, regular expression or CharMatcher instance. Or, instead of using a separator at all, a splitter can extract adjacent substrings of a given fixed length.

从输入的字符串中抽取不重复的子串，通常是分析给定的分割序列；这个分割符可以是单个的字符（on(char separator)）、字符串（on(final String separator)）、正则表达式（on(final Pattern separatorPattern)）或者是一个CharMatcher实例（on(final CharMatcher separatorMatcher)）。当然，也可以不传入分隔符（fixedLength(final int length)），从而将给定的字符串分割为长度为length的子字符串。

比如：Splitter.on(',').split("foo,bar,qux")，返回的结果是Iterable接口类型的数据，内容为foo,bar,qux，并且分割得到的子字符串顺序和其在规定字符串中出现的顺序是一致的。

通常，Splitter的行为是很简单的，并且毫无假设性，如果你有以下的语句Splitter.on(',').split(" foo,,, bar ,")，那么得出来的子字符串序列为 [" foo", "", "", " bar ", ""]，其中包含了空的字符串，这可能不是我们需要的，可以通过Splitter类中的trimResults()方法去掉子串的空格，以及omitEmptyStrings()方法去掉空的子串。运行之后的结果为["foo", "bar"]。

需要注意的：Splitter对象是immutable的，所以在你为Splitter对象添加配置方的时候，你必须用一个新的Splitter对象去接收，如下:

```
public static Iterable<String> test(){
    // Do NOT do this
    Splitter splitter = Splitter.on('/');
    splitter.trimResults(); // does nothing!
    return splitter.split("wrong / wrong / wrong");
}
```

得到的结果是[wrong , wrong , wrong]，可以看出，对splitter对象配置的trimResults()方法并没有对下面的结果起到作用。

Splitter类还提供了limit(int limit)方法，当分割的子字符串达到了limit个时，则停止分割，如下：
Splitter.on(',').limit(3).split("a,b,c,d") 结果为["a", "b", "c,d"]
当需要去掉空的子字符串时，这个空的子字符串是不记入到limit中的，如下：
Splitter.on(',').limit(3).omitEmptyStrings().split("a,,,b,,,c,d")结果为 ["a", "b", "c,d"]

Splitter
可以用指定的分隔符来分割给定的字符串，和Java中String类中的split()相比，Splitter更加灵活：
（1）、Splitter
可以支持单个字符、字符串、正则表达式或者
CharMatcher实例来分割给定的字符串，而Strin
g类中的split()方法只支持用正则表达式来分割给定的字符串，用途很有限。
（2）、我们都知道，用正则表达式来分割字符串的效率很低，对于大量的字符串分割如果用String类中的split()方法势必影响程序的效率，建议使用Splitter类提供的方法。
（3）、String类中的split()方法返回的是String数组类型，而Splitter类返回的是Iterator类型的接口，可以直接用在集合中（Lists.newArrayList(Splitter.on(',').split("_a
,_b_,c__")); ）
（4）、Splitter类用了GwtCompatible注释，它是Google Web Toolkit（GWT）兼容的。
（5）、我们还可以通过Splitter类中的trimResults()方法去掉子串中的空格，以及omitEmpty Strings()方法去掉空的子串，大大简化了用户的使用。
（6）、Splitter类可以在结果中继续分割，比如：Map<String,String> split = Splitter.on(';').trimResults().withKeyValueSeparator("=").split("a=2;b=3");
直接返回Map的键值对，这比String中的split方法强大多了，很值得用。