Java 字符串处理

实验编号 exp0101

0.1 实验目的

熟悉并掌握 String 类、StringBuffer 类和 StringTokenizer 类的使用，并能对字符串变量进行操作。

0.2 实验要求

要求对一个 Java 程序进行简单的词法分析，找出其中用到的关键字、变量类型、变量名、方法名以及操作运算符，并分别打印出来。

注意 考虑如果 (、{ 字符与关键字或变量名相连的情况下该怎么处理，尽可能将代码写完善。

具体要求：

1. 自行学习相关类的使用说明，并概要编写说明手册；
2. 完成代码编写并测试功能；
3. 总结程序无法完成分析的场景并给出描述。

0.3 问题分析

通过定义关键字、变量类型、操作运算符数组，利用文件输入流读入一个文件，对 Java 程序中的语句进行简单的词法分析，找出所要求的元素并打印在终端。

对于变量名和方法名处理稍微复杂，尽可能提出方案并解决。例如，是否可以将 {和} 替换为空格以避免与关键字和变量名紧紧相连的情形，是否可以基于对字符串尾字符是否为 (的判断以确定方法名。

简单的读入文件并分割字符串的参考代码如下：

```
1 package sample.string;
```

```
3  import java.io.FileReader;
4  import java.io.IOException;
5  import java.util.StringTokenizer;

7  public class CharToSting {
8    public static void main(String[] args) throws IOException {
9      char[] charBuffer = new char[30];
10     FileReader fr = new FileReader("hello.txt");
11     StringBuffer sb = new StringBuffer();

13     while(fr.read(charBuffer) != -1) {
14       sb.append(String.valueOf(charBuffer));
15     }

17     StringTokenizer st = new StringTokenizer(sb.toString());

19     while(st.hasMoreTokens()) {
20       System.out.println(st.nextToken().trim().toString());
21     }
22   }
23 }
```

0.4 类用法说明

0.4.1 StringTokenizer

StringTokenizer 属于 java.util 包，用于分隔字符串。
StringTokenizer 构造方法：

- StringTokenizer(String str)
  构造一个用来解析 str 的 StringTokenizer 对象。Java 默认的分隔符是空格、制表符、换行符、回车符。

- StringTokenizer(String str, String delim)
  构造一个用来解析 str 的 StringTokenizer 对象，并提供一个指定的分隔符。

- StringTokenizer(String str, String delim, boolean returnDelims)
  构造一个用来解析 str 的 StringTokenizer 对象，并提供一个指定的分隔符，同时，指定是否返回分隔符。

StringTokenizer 常用方法：

1. int countTokens()：返回 nextToken 方法被调用的次数。
2. boolean hasMoreTokens()：返回是否还有分隔符。
3. boolean hasMoreElements()：返回是否还有分隔符。

4. String nextToken()：返回从当前位置到下一个分隔符的字符串。

5. Object nextElement()：返回从当前位置到下一个分隔符的字符串。

6. String nextToken(String delim)：与 4 类似，以指定的分隔符返回结果。

代码示例：

```
import java.util.*;

public class Main {
  public static void main(String[] args) {
    String str = "runoob,google,taobao,facebook,zhihu";
    // 以 , 号为分隔符来分隔字符串
    StringTokenizer st=new StringTokenizer(str,",");
    while(st.hasMoreTokens()) {
      System.out.println(st.nextToken());
    }
  }
}
```

0.4.2 StringBuffer

StringBuffer 又称为可变字符序列，它是一个类似于 String 的字符串缓冲区，通过某些方法调用可以改变该序列的长度和内容。我们可以使用 StringBuffer 的 append 方法将制定字符串追加到字符串序列。

```
StringBuffer sb = new StringBuffer();
sb.append("Hello");
  sb.append(", ");
sb.append("StringBuffer");
```

0.4.3 FileReader

FileReader 类从 InputStreamReader 类继承而来，该类按字符读取流中数据。从文件名创建一个 FileReader 并读出数据示例代码：

```
char[] charBuffer = new char[30];
FileReader fr = new FileReader("hello.txt");
while(fr.read(charBuffer) != -1) {
  System.out.println(charBuffer);
}
```