

PossibleError.java

```
2 * Licensed to the Apache Software Foundation (ASF) under one or more
17 package com.google;
18
19 import java.io.BufferedReader;
20 import java.io.FileReader;
21 import java.io.IOException;
22 import java.util.ArrayList;
23 import java.util.Arrays;
24 import java.util.Collections;
25 import java.util.List;
26
27 /**
28 * Possible Errors.
29 *
30 * @author lihg
31 * @version 2013-6-11
32 */
33 public class PossibleError {
34     private List<String> names;
35
36     /**
37      * List is stored <font color="red"><b>without copying</b></font>.
38      *
39      * @param names
40      */
41     public void setNames(List<String> names) {
42         this.names = names;
43     }
44
45     /**
46      * Avoid <font color="red"><b>null</b></font> return values.
47      *
48      * @param l
49      * @return
50      */
51     public List<String> nullReturn(List<String> l) {
52         if (l.isEmpty()) {
53             // avoid
54             return null;
55             return Collections.emptyList();
56         }
57         return l;
58     }
59
60     /**
61      * <pre>
62      * 1, Check Type in <font color="red"><b>equals</b></font>.
63      * 2, Overloaded equals.
64      * </pre>
65      */
66     @Override
67     public boolean equals(Object obj) {
68         // 性能优化: 比较操作有可能很昂贵
69         if (this == obj)
70             return true;
71
72         if (! (obj instanceof PossibleError))
73             return false;
74
75         PossibleError other = (PossibleError) obj;
76         return names == null ? other.names == null : names.equals(other.names);
77     }
78 }
```

## PossibleError.java

```

77     }
78
79     /**
80      * {@link AbstractList#hashCode()}已实现生成{@link List}的hash值功能.
81      *
82      * 参考{@link Arrays#hashCode(Object[])}}实现.
83      */
84     @Override
85     public int hashCode() {
86         if (names == null)
87             return 0;
88
89         return names.hashCode();
90     }
91
92     /**
93      * <pre>
94      * 1, <font color="red"><b>Close</b></font> in <b>finally</b>.
95      * 2, Throw in <b>finally</b>.
96      * </pre>
97      *
98      * @param fileName
99      * @return
100     * @throws IOException
101     */
102     public static String bufferedRead(String fileName) throws IOException {
103         BufferedReader in = null;
104
105         try {
106             in = new BufferedReader(new FileReader(fileName));
107
108             StringBuilder content = new StringBuilder(1024); // 2KB
109             String l = null;
110             while ((l = in.readLine()) != null) {
111                 content.append(l);
112             }
113             return content.toString();
114         } finally {
115             if (in != null) {
116                 in.close();
117             }
118             // don't throw exception in finally!
119         }
120     }
121
122     /**
123      * <pre>
124      * 1, <font color="red"><b>Empty</b></font> <b>catch</b> clause.
125      * 2, Handle numeric parsing errors.
126      * </pre>
127      *
128      * @param s
129      * @return
130      */
131     public static long parseLong(String s) {
132         try {
133             long l = Long.parseLong(s, 10);
134             return l;
135         } catch (NumberFormatException nfe) {
136             // empty catch
137             return -1;

```

PossibleError.java

```
138     }
139 }
140
141 private String content;
142
143 /**
144  * <pre>
145  * 1, <b>Field</b> might have <font color="red"><b>null</b></font> value.
146  * 2, Use <font color="red"><b>==</b></font> to compare with <b>null</b>.
147  * </pre>
148  *
149  * @return
150  */
151 public String getContent() {
152     if (content == null)
153         content = "";
154
155     return content;
156 }
157
158 /**
159  * <font color="red"><b>Improper conversion</b></font> of <b>Array to String</b>.
160  *
161  * @param ids
162  * @return
163  */
164 public static String arrayToString(long[] ids) {
165 //     return ids.toString();
166     return Arrays.toString(ids);
167 }
168 }
169
```