src/stuff/Sample.java

```
1 package stuff;
2
3 import java.awt.event.ActionEvent;
4 import java.awt.event.ActionListener;
5 import java.util.Iterator;
6 import java.util.LinkedList;
7 import java.util.List;
8 import java.util.Map;
9
10 /**A sample class to demonstrate the functionalities of the J2L Compiler.
11 * @see C1.jj
12 * @version 1.0*/
13 public abstract class Sample<T,E> implements SampleInter<T,E>,SampleInter2
extends Object{
14      /**Constant test for Octints.*/
15      public static int OCT_INT=0400;
16      /**Constant test for Hexints.*/
17      protected static int HEX_INT=0x400;
18      /**Constant test for Strings.*/
19      private static String KONSTANTE_TEST="Konstantentest";
20      /**Tests the implementation of transient.*/
21      public transient float floattest;
22      /**Tests the implementation of final.*/
23      protected boolean booltest;
24      /**Tests the implementation of volatile.*/
25      private volatile double doubletest;
26      /**Tests the implementation of byte.*/
27      private byte[] bytetest;
28      /**Tests the implementation of char.*/
29      private char chartest;
30      /**Tests the implementation of int.*/
31      private int inttest;
32      /**Test the implementation of a class (String).*/
33      private String stringtest;
34      /**Tests the generic types of a field.*/
35      private List<String> list;
36      /**Tests the implementation of an enum definition.*/
37      private enum  enumtest{ABC, DEF, GHI};
38      /**Tests the implementation of a Constructor.
39      * @param floattest float parameter
40      * @param booltest boolean parameter
41      * @param doubletest double parameter
42      */
43      public Sample (float floattest, boolean booltest, double doubletest){
```

```
44          super();
45          this.floattest=15.0f;
46          this.booltest=true;
47          this.doubletest=15.23;
48          this.stringtest=new String("Hallo");
49      }
50
51      @Override
52      public synchronized byte hello(int you){
53          return 0;
54      }
55
56      @Override
57      public double testExceptions(int test){
58          try {
59                  Integer.parseInt("Hallo");
60          }catch(NumberFormatException event){
61                  event.getStackTrace();
62          }finally{
63                  this.chartest='b';
64          }
65          if (--test>2){
66              throw new  NullPointerException();
67          }
68          new String();
69          return 1.0;
70      }
71
72      @Override
73      public boolean testExpressions(int abc){
74          this.chartest='a';
75          abc|=2;
76          abc=abc|2;
77          abc&=2;
78          abc=abc&2;
79          abc^ =2;
80          abc=abc^ 2;
81          abc=abc<<2;
82          abc<<=2;
83          abc=abc>>2;
84          abc>>=2;
85          abc=abc>>>2;
86          abc>>>=2;
87          this.floattest-=2;
88          this.floattest=this.floattest-2;
89          this.floattest%=2;
```

```
90          this.floattest=this.floattest%2;
91          this.doubletest+=4;
92          this.doubletest=this.doubletest+4;
93          this.floattest/=2;
94          this.floattest=this.floattest/2;
95          this.floattest*=2;
96          this.floattest=this.floattest*2;
97          switch (4){
98              case 'a':
99                      break;
100             case 4:
101             case OCT_INT:
102                             chartest='b';
103             default:
104                     break;
105         }
106         return true;
107     }
108
109     @Override
110     public String testIfElse(){
111         if (this.stringtest==null){
112             this.stringtest=Sample.KONSTANTE_TEST;
113         }
114         else if (this.stringtest.equals("ABC") || this.stringtest.equals("DEF")){
115             this.stringtest=Sample.KONSTANTE_TEST;
116         }
117         else if (this.stringtest instanceof String){
118             this.stringtest="Cool!";
119         }
120         if (!this.booltest && this.inttest==4){
121             this.floattest=0;
122         }
123         else if ( this.inttest>0){
124             return "false";
125         }
126         else if (this.floattest!=5){
127             this.stringtest="!5";
128         }
129         else if (this.doubletest<=2){
130             this.doubletest=0.0;
131         }
132         else if (this.doubletest>=2){
133             this.chartest='t';
134         }
135         else if (this.doubletest<1){
```

```
136                 this.doubletest=1.0;
137             }
138             else {
139                 this.stringtest="";
140             }
141             this.inttest=(int )this.floattest;
142             final String str=this.booltest ? this.getStringtest() : null;
143             return str;
144         }
145 
146         @Override
147         public List<String> testIncrements(){
148             return new LinkedList<String>();
149         }
150 
151         @Override
152         public int testInnerClasses(){
153             final ActionListener act=new ActionListener(){
154                 @Override
155                 public void actionPerformed(ActionEvent arg0){
156                     arg0.toString();
157                 }
158 
159             };
160             return 0;
161         }
162 
163         @Override
164         public boolean testLoops(final float test, Sample<T,E> sample)throws
NullPointerException{
165             sample=this;
166             final List<String> strlist=new LinkedList<String>();
167             int count=0;
168             String [] strarray=new String[5];
169             while(count<100){
170                 ++count;
171             }
172             do{
173                 count++;
174             }while(count<100);
175             for(String str : strlist){
176                 testExpressions(count);
177             }
178             for(Iterator<String> iter=strlist.iterator();iter.hasNext()){
179                 iter.next();
180             }
```

```
181             for(int i=0;i<10;i++){
182                 i--;
183             }
184             for(;;){
185                 return testExpressions(count);
186             }
187         }
188 
189         /**Gets the bytetest value.
190         * @return the bytetest
191         */
192         public byte[] getBytetest(){
193             return bytetest;
194         }
195 
196         /**Gets the chartest value.
197         * @return the chartest
198         */
199         public char getChartest(){
200             return chartest;
201         }
202 
203         /**Gets the doubletest value.
204         * @return the doubletest
205         */
206         public double getDoubletest(){
207             return doubletest;
208         }
209 
210         /**Gets the floattest value.
211         * @return the floattest
212         */
213         public float getFloattest(){
214             return floattest;
215         }
216 
217         /**Gets the Stringtest value.
218         * @return the stringtest
219         */
220         public String getStringtest(){
221             return this.stringtest;
222         }
223 
224         /**Returns the boolean value of booltest.
225         * @return the booltest value
226         */
```

```
227     public boolean isBooltest(){
228         return booltest;
229     }
230 
231     /**Sets the bytetest value.
232     * @param bytetest the bytetest to set
233     */
234     public void setBytetest(byte[] bytetest){
235         this.bytetest=bytetest;
236     }
237 
238     /**Sets the chartest value.
239     * @param chartest the chartest to set
240     */
241     public void setChartest(char chartest){
242         this.chartest=chartest;
243     }
244 
245     /**Sets the doubletest value.
246     * @param doubletest the doubletest to set
247     */
248     public void setDoubletest(double doubletest){
249         this.doubletest=doubletest;
250     }
251 
252     /**Sets the floattest value.
253     * @param floattest the floattest to set
254     */
255     public void setFloattest(float floattest){
256         this.floattest=floattest;
257     }
258 
259     /**
260     * @param stringtest the stringtest to set
261     */
262     public void setStringtest(String stringtest){
263         this.stringtest=stringtest;
264     }
265 
266 }
```