```
 1  // Example 9.1 SerializationDemo.java
 2  //
 3  //
 4  // Copyright 2003 Bill Campbell and Ethan Bolker
 5
 6  import java.io.*;
 7  import java.util.*;
 8
 9  // Test of Java serialization.
10  //
11  // %> java SerializationDemo
12  // Wrote: round blue Mon Jan 06 20:14:44 EST 2003
13  // Read:  round null Mon Jan 06 20:14:44 EST 2003
14  //
15  // interesting observations:
16  //
17  // %> wc -c SerializationDemo* tmp
18  //    1207 SerializationDemo$Circle.class
19  //    1611 SerializationDemo.class
20  //    3221 SerializationDemo.java
21  //     271 tmp
22  //
23  // the "strings" command finds ascii strings in a file
24  //
25  // %> strings SerializationDemo.class | wc
26  //    45      45     813
27  //
28  // %> strings tmp
29  // SerializationDemo$Circle?
30  // attributest
31  //     Ljava/util/Map;L
32  // selft
33  //      LSerializationDemo$Circle;xpsr
34  // java.util.HashMap
35  // loadFactorI
36  // thresholdxp?@
37  // datesr
38  // java.util.Datehj
39  // thisq
40  // namet
41  // roundxq
42  //
43  // %> strings tmp | wc -c
44  // 180
45
46  public class SerializationDemo
47  {
48      public static void main (String[] args)
49      {
50          Circle circle = new Circle("round");
51          write(circle,"tmp");
52          System.out.println("Wrote: "+ circle);
53
54          Circle circleCopy = (Circle)read("tmp");
55          System.out.println("Read:  " + circleCopy);
56      }
```



```
57
58      public static void write (Object obj, String pathname)
59      {
60          try {
61              FileOutputStream f = new FileOutputStream(pathname);
62              ObjectOutput s = new ObjectOutputStream(f);
63              s.writeObject(obj);
64              s.flush();  s.close();  f.close();
65          }  catch (Exception e) { e.printStackTrace(); }
66      }
67
68      public static Object read(String pathname)
69      {
70          try {
71              Object obj;
72              FileInputStream in = new FileInputStream(pathname);
73              ObjectInputStream s = new ObjectInputStream(in);
74              obj = s.readObject();
75              s.close(); in.close();
76              return(obj);
77          }
78          catch (Exception e) {
79              e.printStackTrace();
80          }
81          return(null);
82      }
83
84      // To implement the Serializable interface you just _say_ so.
85      // You don't have to _do_ anything, although you may choose to
86      // overwrite the writeObject() and readObject() methods.
87
88      private static class Circle implements Serializable
89      {
90          private Circle self;    // a circular reference
91          private Map attributes; // saved with its contents
92
93          // Don't bother saving whatever the current color
94          //(user settable) happens to be:
95          transient private String color;
96
97          Circle(String name)
98          {
99              attributes = new HashMap();
100             attributes.put("this", this); // a circular reference
101             attributes.put("name", name);
102             attributes.put("date", new Date());
103             this.color = "blue";
104             self = this;
105         }
106
107         public void setColor(String color)
108         {
109             this.color = color;
110         }
111
112         // NOTE: serialization does not call toString-- it calls
```

```
113          // a smarter serialization method.
114          public String toString()
115          {
116              return( (String)attributes.get("name") + " " +
117                      color + " " + (Date)attributes.get("date") );
118          }
119      }
120  }
```