

```

1  /**
2  * Class Name      : SwedishAddress.java
3  * Version        : Build 01.00
4  * Date of Change   : 2000 MAR 11
5  * Author         : Team # 4
6  * Copyright      : 2000 AITP NCC 2000 Java Competition
7  * Description     : The address objects
8  *                : specific for Swedish
9  *                : addresses.
10 *
11 * Version History:
12 *
13 * Version          Date          Team # 4      Author
14 *   Build 01.00    2000 MAR 31    Team # 4
15 *                :                :                : Added
16 *   Build 00.00    2000 MAR 31    Don Baldwin
17 *                :                :                : Provided
18 *                :                :                : design framework construct.
19 *                :                :                : Add
20 *                :                :                : functionality to the framework skeleton
21 *                :                :                : that is
22 *                :                :                : provided below. You will need to make
23 *                :                :                : appropriate modifications to the code below.
24 *
25 * Known Problems: None
26 *
27 * @version Build 01.01
28 * @author Team # 4
29 */
30
31 import java.sql.*;
32 import java.sql.SQLException;
33 import java.util.StringTokenizer;
34 public class SwedishAddress extends Address
35 {
36     private String strStreet;
37     private String strCity;
38     private String strSECountryCode;
39     private String strSECountry;
40     private String strPostalRegionCode;
41     private String strPostalBlockCode;
42
43     /**
44      * Method to set the street information.
45      */
46     public void setStreet(String aStrStreet)
47         throws IllegalArgumentException
48     {
49         aStrStreet = aStrStreet.trim();// by trimmin first we avoid
50         an input of only white characters
51         if(aStrStreet == null || aStrStreet.equals(""))
52             throw new IllegalArgumentException(
53                 "Street can't be empty");
54         strStreet = aStrStreet;
55     }
56
57     /**
58      * Method to get the street information.
59      */
60     public String getStreet()

```

```
56     {
57     return strStreet;
58     }
59
60     /**
61     * Method to set the country code.
62     */
63     public void setCountryCode(String aStrSECountryCode)
64         throws IllegalArgumentException
65     {
66         aStrSECountryCode = aStrSECountryCode.trim();// by trimmin
67         first we avoid an imput of only white characters
68         if(aStrSECountryCode == null || aStrSECountryCode.equals(""))
69             throw new IllegalArgumentException(
70                 "Country code can't be empty");
71         strSECountryCode = aStrSECountryCode;
72     }
73
74     /**
75     * Method to set the country code.
76     */
77     public String getCountryCode()
78     {
79         return strSECountryCode;
80     }
81
82     /**
83     * Method to set the country name.
84     */
85     public void setCountry(String aStrSECountry)
86         throws IllegalArgumentException
87     {
88         aStrSECountry = aStrSECountry.trim();// by trimmin first we
89         avoid an imput of only white characters
90         if(aStrSECountry == null || aStrSECountry.equals(""))
91             throw new IllegalArgumentException(
92                 "Country can't be empty");
93         strSECountry = aStrSECountry;
94     }
95
96     /**
97     * Method to get the country name.
98     */
99     public String getCountry()
100    {
101        return strSECountry;
102    }
103
104    /**
105    * Method to set the city name.
106    */
107    public void setCity(String aStrCity)
108        throws IllegalArgumentException
109    {
110        aStrCity = aStrCity.trim();// by trimmin first we avoid an
111        imput of only white characters
112        if(aStrCity == null || aStrCity.equals(""))
113            throw new IllegalArgumentException("City can't be
114            empty");
115        strCity = aStrCity;
116    }
117
```

```
112     /**
113      * Method to get the city name.
114      */
115     public String getCity()
116     {
117         return strCity;
118     }
119
120     /**
121      * Method to set the postal region code.
122      */
123     public void setPostalRegionCode(String aStrPostalRegionCode)
124         throws IllegalArgumentException
125     {
126         aStrPostalRegionCode = aStrPostalRegionCode.trim();// by
127         trimmin first we avoid an imput of only white characters
128         if(aStrPostalRegionCode == null || aStrPostalRegionCode.
129         equals(""))
130             throw new IllegalArgumentException(
131             "Postal Region can't be empty");
132         strPostalRegionCode = aStrPostalRegionCode;
133     }
134
135     /**
136      * Method to get the postal region code.
137      */
138     public String getPostalRegionCode()
139     {
140         return strPostalRegionCode;
141     }
142
143     /**
144      * Method to set the postal block code.
145      */
146     public void setPostalBlockCode(String aStrPostalBlockCode)
147         throws IllegalArgumentException
148     {
149         aStrPostalBlockCode = aStrPostalBlockCode.trim();// by
150         trimmin first we avoid an imput of only white characters
151         if(aStrPostalBlockCode == null || aStrPostalBlockCode.equals(
152         ""))
153             throw new IllegalArgumentException(
154             "Postal block can't be empty");
155         strPostalBlockCode = aStrPostalBlockCode;
156     }
157
158     /**
159      * Method to get the postal block code.
160      */
161     public String getPostalBlockCode()
162     {
163         return strPostalBlockCode;
164         //return strPostalBlockCode;
165     }
166
167     /**
168      * Method to set the postal code.
169      */
170     public void setPostalCode(String aStrPostalCode)
171         throws IllegalArgumentException
172     {
173         StringTokenizer stk = new StringTokenizer(aStrPostalCode,
```

```
167         "-");// only space accepted as a valid delimiter
168         String region, block;// temporary variable to hold the postal
           region code and postal block code
169         // first check to make sure that it is two tokens separated
           by a space
170         if(stk.countTokens() != 2)
171             throw new IllegalArgumentException("Invalid postal
           code");
172         region = stk.nextToken();
173         block = stk.nextToken();
174         // make sure both region and block have the right length
175         if(region.length() != 3 || block.length() != 2)
176             throw new IllegalArgumentException("Invalid postal
           code");
177         // make sure both region and block are valid Integers
178         try {
179             Integer.parseInt(region);
180             Integer.parseInt(block);
181         } catch (NumberFormatException nfe) {
182             throw new IllegalArgumentException("Invalid postal
           code");
183         }
184         // finally, since we know they are valid--save them
185         setPostalRegionCode(region);
186         setPostalBlockCode(block);
187     }
188
189     /**
190     * Return the postal code as "RegionCode BlockCode"
191     * (xxx xx)
192     */
193     public String getPostalCode()
194     {
195         return getPostalRegionCode() + " " + getPostalBlockCode();
196     }
197
198     /**
199     * Build the mailing label which will be displayed like
200     * this in the address book.
201     */
202     public String getMailingLabel()
203     {
204         return getCompanyName() + "\n" +
205             getFirstName() + " " + getLastName() + "\n" +
206             getStreet() + "\n" +
207             getPostalCode() + " " + getCity() + "\n" +
208             getCountry();
209     }
210
211     /**
212     * Creates a new european address in the database with
213     * the data retrieved from the address object passed to
214     * the method.
215     */
216     public void writeAddress()
217         throws SQLException
218     {
219         Connection con = DBInfo.initConnection();
220         Statement stmt = con.createStatement();
221
222         stmt.executeUpdate(
223             "INSERT INTO ADDRESS" +
```

```
224
    " ( FIRSTNAME, LASTNAME, COMPANYNAME, STREET, CITY,
STATE, POSTALCODE, POSTALBLOCK, COUNTRYCODE, COUNTRY)" +
225
    " VALUES (" +
226
    " '" + getFirstName() +
227
    " ', '" + getLastName() +
228
    " ', '" + getCompanyName() +
229
    " ', '" + getStreet() +
230
    " ', '" + getCity() +
231
    " ', ' '" +
232
    " ', '" + getPostalRegionCode() +
233
    " ', '" + getPostalBlockCode() +
234
    " ', '" + getCountryCode() +
235
    " ', '" + getCountry() +
236
    " ' )"
237
    );
238
    }
239
240
241
    /**
242
    * toString() displays the class attributes,
243
    * for debugging purposes.
244
    */
245
    public String toString()
246
    {
247
    return getMailingLabel();
248
    }
249
250
    } // END SwedishAddress.java
```