

```

1 // foj/4/dictionary/Dictionary.java
2 //
3 //
4 // Copyright 2003 Bill Campbell and Ethan Bolker
5
6 import java.util.*;
7
8 /**
9  * Model a dictionary with a TreeMap of (word, Definition) pairs.
10  *
11  * @see Definition
12  *
13  * @version 4
14  */
15
16 public class Dictionary
17 {
18     private TreeMap entries;
19
20     /**
21      * Construct an empty Dictionary.
22      */
23     public Dictionary()
24     {
25         entries = new TreeMap();
26     }
27
28     /**
29      * Add an entry to this Dictionary.
30      *
31      * @param word the word being defined.
32      * @param definition the Definition of that word.
33      */
34     public void addEntry( String word, Definition definition )
35     {
36         entries.put( word, definition );
37     }
38
39     /**
40      * Look up an entry in this Dictionary.
41      *
42      * @param word the word whose definition is sought
43      * @return the Definition of that word, null if none.
44      */
45     public Definition getEntry( String word )
46     {
47         return (Definition)entries.get(word);
48     }
49
50     /**
51      * Get the size of this Dictionary.
52      *
53      * @return the number of words.
54
55
56

```

```

57     */
58     public int getSize()
59     {
60         return entries.size();
61     }
62
63     /**
64      * Construct a String representation of this Dictionary.
65      *
66      * @return a multiline String representation.
67      */
68     public String toString()
69     {
70         String str = "";
71         String word;
72         Definition definition;
73         Set allWords = entries.keySet();
74         Iterator wordIterator = allWords.iterator();
75         while ( wordIterator.hasNext() ) {
76             word = (String)wordIterator.next();
77             definition = this.getEntry( word );
78             str += word + ":\n" + definition.toString() + "\n";
79         }
80         return str;
81     }
82
83 }
84

```

```
1 // fo1/4/dictionary/Definition.java
2 //
3 //
4 // Copyright 2003 Bill Campbell and Ethan Bolker
5
6 /**
7  * Model the definition of a word in a dictionary.
8  *
9  * @see Dictionary
10 *
11 * @version 4
12 */
13
14 public class Definition
15 {
16     private String definition; // the defining string
17
18     /**
19      * Construct a simple Definition.
20      *
21      * @param definition the definition.
22      */
23
24     public Definition( String definition )
25     {
26         this.definition = definition;
27     }
28
29     /**
30      * Construct a String representation of this Definition.
31      *
32      * @return the definition string.
33      */
34
35     public String toString()
36     {
37         return definition;
38     }
39 }
```

```

1 // fo1/4/dictionary/lookup.java
2 //
3 //
4 // Copyright 2003 Bill Campbell and Ethan Bolker
5
6 /**
7  * On line word lookup.
8  *
9  * @see Dictionary
10 * @see Definition
11 *
12 * @version 4
13 */
14
15 public class Lookup
16 {
17     private static Terminal t = new Terminal();
18     private static Dictionary dictionary = new Dictionary();
19
20     /**
21      * Helper method to fill the dictionary with some simple
22      * Definitions.
23      *
24      * A real Dictionary would live in a file somewhere.
25      */
26
27     private static void fillDictionary()
28     {
29         dictionary.addEntry( "shape",
30             new Definition( "a geometric object in a plane" ) );
31         dictionary.addEntry( "quadrilateral",
32             new Definition( "a polygonal shape with four sides" ) );
33         dictionary.addEntry( "rectangle",
34             new Definition( "a right-angled quadrilateral" ) );
35         dictionary.addEntry( "square",
36             new Definition( "a rectangle having equal sides" ) );
37     }
38
39     /**
40      * Helper method to print the Definition of a single word,
41      * or a message if the word is not in the Dictionary.
42      *
43      * @param word the word whose definition is wanted.
44      */
45
46     private static void printDefinition(String word)
47     {
48         Definition definition = dictionary.getEntry(word);
49         if (definition == null) {
50             t.println("sorry, no definition found for " + word);
51         }
52         else {
53             t.println(definition.toString());
54         }
55     }
56

```

```

57     /**
58      * Run the Dictionary lookup.
59      *
60      * Parse command line arguments for words to look up,
61      * "all" prints the whole Dictionary.
62      *
63      * Then prompt for more words, "quit" to finish.
64      *
65      * For example,
66      * <pre>
67      *
68      * %> java Lookup shape square circle
69      * shape:
70      * a geometric object in a plane
71      * square:
72      * a rectangle having equal sides
73      * circle:
74      * sorry, no definition found for circle
75      *
76      * look up words, "quit" to quit
77      * word> rectangle
78      * a right-angled quadrilateral
79      * word> quit
80      * %>
81      * </pre>
82
83      * @param args the words that we want looked up, supplied as
84      * command line arguments. If the word "all" is
85      * included, all words are looked up.
86      */
87
88     public static void main( String[] args )
89     {
90         // fill the dictionary (not a big one!)
91         fillDictionary();
92
93         // look up some words
94         String word;
95
96         // words specified on command line
97         for (int i = 0; i < args.length; i++) {
98             word = args[i];
99             if (word.equals("all")) {
100                 t.println("The whole dictionary ( " +
101                     dictionary.getSize() + " entries):");
102                 t.println("-----");
103                 t.println(dictionary.toString());
104                 t.println("-----");
105             }
106             else {
107                 t.println(word + ":");
108                 printDefinition(word);
109             }
110         }
111         // words entered interactively
112

```

```
113     t.println("\nlook up words, \"quit\" to quit");
114     while (true) {
115         word = t.readWord("word> ");
116         if (word.equals("quit")) {
117             break;
118         }
119         printDefinition(word);
120     }
121 }
122 }
```