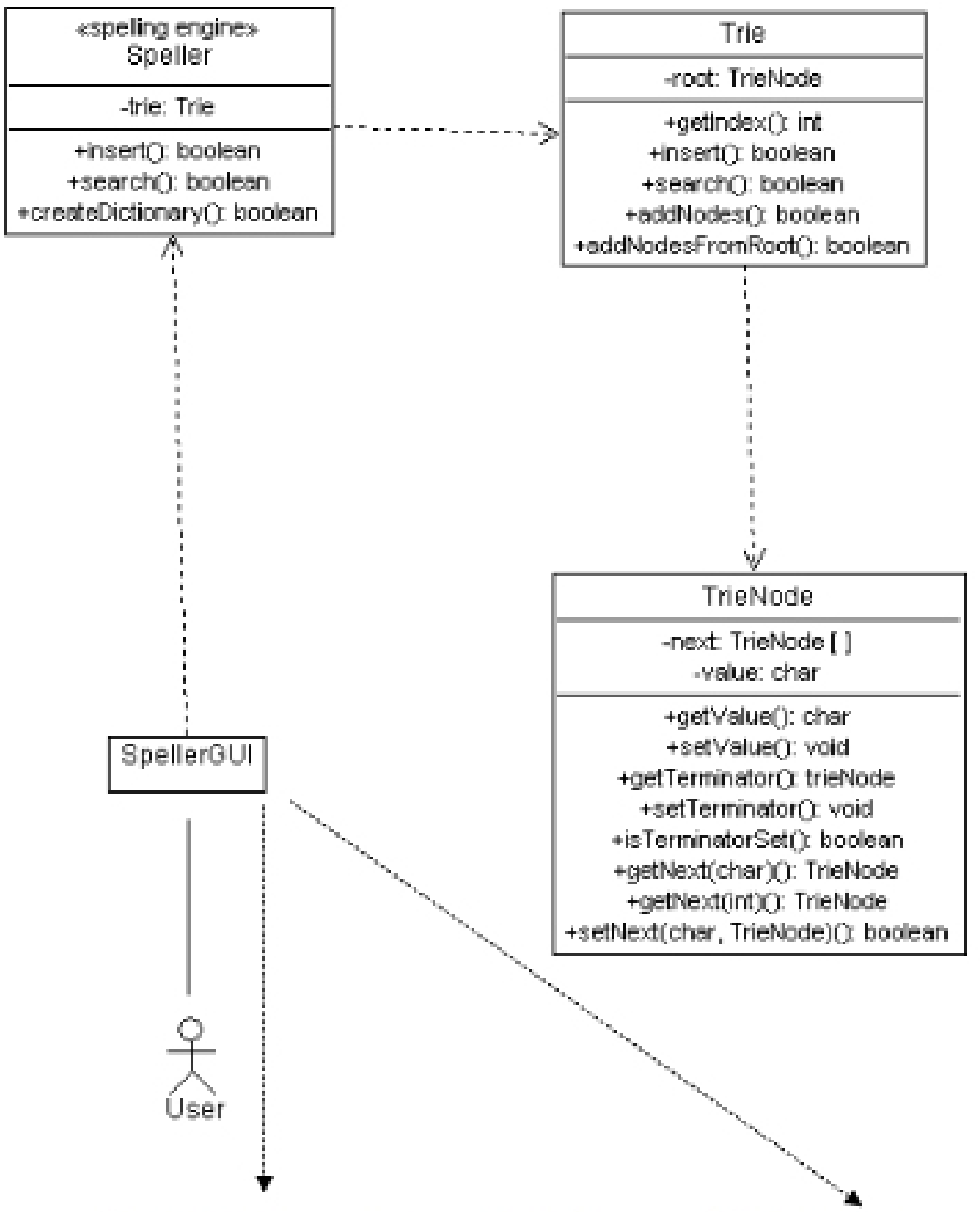
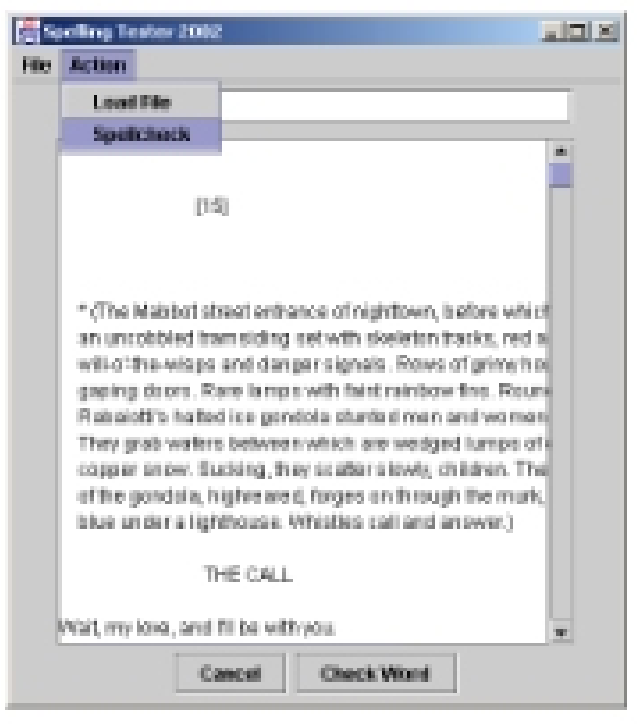
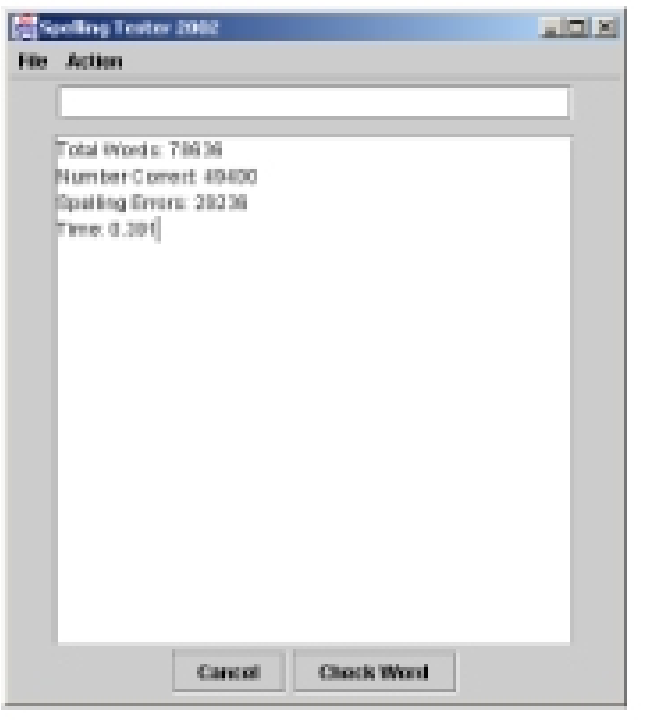
SpellChecker Program Using Java Trie Implementation







```
// Author Julius Dichter
// © 2002


import java.util.*;
import java.io.*;

public class Speller {

Trie trie;

public Speller() {

RandomAccessFile file = null;
        trie = new Trie();

try {
        file = new RandomAccessFile("words.all","r"); }
catch (IOException ioe) { System.err.println("File not found"); }

try {
for (int i=0; ; i++) {
        String word = file.readLine();
        if (word == null) break;
        if (word.indexOf((int)'\'')!= -1) continue;
        trie.insert(word);        }

        trie.insert("aardvark");
}
catch (EOFException eofe) { }
catch (IOException ioe) { ioe.printStackTrace(); }}


public void insert (String string) {
        trie.insert(string); }

public boolean search(String string) {
        return trie.search(string); }

public static void main(String [] s) {
        Speller speller = new Speller(); }

}
```

```
public class Trie {

// make private in implementation
public TrieNode root;


public Trie() {
        root = new TrieNode(); }

// returns index 1-26 for valid words, -1 for
public static int getIndex(char ch) {

if  (ch >= 'a' && ch <= 'z')
        return (int)ch - 'a' + 1;
else if  (ch >= 'A' && ch <= 'Z')
   return (int)ch - 'A' + 1;
else
  return -1;  }


public boolean insert(String string) {

if (string == null) return false;

char ch = string.charAt(0);

TrieNode curr = root.getNext(ch);

if (curr == null) { // 1st word starting with given letter
        return addNodesFromRoot(string); }

TrieNode prev = null;

// for each character in the string
for (int i = 0; curr != null && i < string.length(); i++) {
        ch = string.charAt(i);

        // set node value to character
        curr.setValue(ch);

        if (curr != null) { // curr NOT null: we have a character match
                // is it the end of the string (a match)

        if (i == string.length() -1)  {   // end of string
                if (curr.getTerminator() != null)  // already exists
                        return false;
```