**package** com.google.engedu.anagrams;  
  
**import** java.io.BufferedReader;  
**import** java.io.IOException;  
**import** java.io.InputStream;  
**import** java.io.InputStreamReader;  
**import** java.util.Arrays;  
**import** java.util.ArrayList;  
**import** java.util.HashMap;  
**import** java.util.HashSet;  
**import** java.util.Random;  
  
*/\*\*  
 \*  
 \* Google - Computer Science with Android  
 \* Unit 1: Getting Started  
 \*  
 \* Extensions Applied:  
 \* 1) Two-letter mode: switch to allowing the user to add two letters to form anagrams  
 \* 2) Optimize word selection by removing words that do not have enough anagrams from the pool of possible starter words. Note that those words should still remain in wordSet since they can still be used as anagrams to other words.  
 \*  
 \*/***public class** AnagramDictionary {  
  
 *// Constant Variables* **private static final int** MIN\_NUM\_ANAGRAMS = 5;  
 **private static final int** DEFAULT\_WORD\_LENGTH = 3;  
 **private static final int** MAX\_WORD\_LENGTH = 7;  
  
 *// Incremental Word Length Variable* **private static int** wordLength = DEFAULT\_WORD\_LENGTH;  
  
 *// Random Class to generate Random Numbers* **private** Random random = **new** Random();  
  
 *// Data Structures to Hold Information* HashSet<String> wordSet = **new** HashSet<>();  
 ArrayList<String> wordList = **new** ArrayList<>();  
 HashMap<String, ArrayList<String>> lettersToWord = **new** HashMap<>();  
 HashMap<Integer, ArrayList<String>> sizeToWords = **new** HashMap<>();  
  
 **public** AnagramDictionary(InputStream wordListStream) **throws** IOException {  
 BufferedReader in = **new** BufferedReader(**new** InputStreamReader(wordListStream));  
 String line;  
 ArrayList<String> wordMapList;  
  
 **while**((line = in.readLine()) != **null**) {  
 String word = line.trim();  
 wordSet.add(word);  
 wordList.add(word);  
  
 *// Populate the sizeToWords HashMap* **if** (sizeToWords.containsKey(word.length())) {  
 wordMapList = sizeToWords.get(word.length());  
 wordMapList.add(word);  
 sizeToWords.put(word.length(), wordMapList);  
 } **else** {  
 ArrayList<String> newWordList = **new** ArrayList<>();  
 newWordList.add(word);  
 sizeToWords.put(word.length(), newWordList);  
 }  
  
 ArrayList<String> sortedList = **new** ArrayList<>();  
 String sortedWord = sortLetters(word);  
  
 *// Populate the lettersToWord HashMap* **if** ( !(lettersToWord.containsKey(sortedWord)) ) {  
 sortedList.add(word);  
 lettersToWord.put(sortedWord, sortedList);  
 } **else** {  
 sortedList = lettersToWord.get(sortedWord);  
 sortedList.add(word);  
 lettersToWord.put(sortedWord, sortedList);  
 }  
 }  
 }  
  
 *// This function checks if the word is valid.  
 // the provided word is a valid dictionary word (i.e., in wordSet), and  
 // the word does not contain the base word as a substring.* **public boolean** isGoodWord(String word, String base) {  
 **if** (wordSet.contains(word) && !(base.contains(word))) {  
 **return true**;  
 } **else** {  
 **return false**;  
 }  
  
 }  
  
 *// Default Anagram Fetcher* **public** ArrayList<String> getAnagrams(String targetWord) {  
 ArrayList<String> resultList = **new** ArrayList<>();  
 **return** resultList;  
 }  
  
 *// Anagram Fetcher for One More Extra Letter* **public** ArrayList<String> getAnagramsWithOneMoreLetter(String word) {  
 ArrayList<String> tempList;  
 ArrayList<String> resultList = **new** ArrayList<>();  
  
 **for** (**char** alphabet = **'a'**; alphabet <= **'z'**; alphabet++) {  
 String anagram = word + alphabet;  
 String sortedAnagram = sortLetters(anagram);  
  
 **if** (lettersToWord.containsKey(sortedAnagram)) {  
 tempList = lettersToWord.get(sortedAnagram);  
  
 **for** (**int** i = 0; i < tempList.size(); i++) {  
 **if** ( !(tempList.get(i).contains(word)) ) {  
 resultList.add(tempList.get(i));  
 }  
 }  
 }  
 }  
  
 **return** resultList;  
 }  
  
 *// Anagram Fetcher for Two More Extra Letters* **public** ArrayList<String> getAnagramsWithTwoMoreLetters(String word) {  
 ArrayList<String> tempList;  
 ArrayList<String> resultList = **new** ArrayList<>();  
  
 **for** (**char** firstAlphabet = **'a'**; firstAlphabet <= **'z'**; firstAlphabet++) {  
 **for** (**char** secondAlphabet = **'a'**; secondAlphabet <= **'z'**; secondAlphabet++) {  
 String anagram = word + firstAlphabet + secondAlphabet;  
 String sortedAnagram = sortLetters(anagram);  
  
 **if** (lettersToWord.containsKey(sortedAnagram)) {  
 tempList = lettersToWord.get(sortedAnagram);  
  
 **for** (**int** i = 0; i < tempList.size(); i++) {  
 **if** (!(tempList.get(i).contains(word))) {  
 resultList.add(tempList.get(i));  
 }  
 }  
 }  
 }  
 }  
  
 **return** resultList;  
 }  
  
 *// Function to pick a suitable and valid starter word,  
 // and increment for every subsequent level* **public** String pickGoodStarterWord() {  
 **int** randomNumber;  
 String starterWord;  
  
 **do** {  
 randomNumber = random.nextInt(sizeToWords.get(wordLength).size());  
 starterWord = sizeToWords.get(wordLength).get(randomNumber);  
 } **while** (getAnagramsWithTwoMoreLetters(starterWord).size() < MIN\_NUM\_ANAGRAMS);  
  
 **if** (wordLength < MAX\_WORD\_LENGTH) {  
 wordLength++;  
 }  
  
 **return** starterWord;  
 }  
  
 *// Function to sort a single string alphabetically,  
 // and returns a new string* **public** String sortLetters(String word) {  
 **char**[] characters = word.toCharArray();  
 Arrays.sort(characters);  
 String sortedWord = **new** String(characters);  
 **return** sortedWord;  
 }  
}

anagram activity

**package** com.google.engedu.anagrams;  
  
**import** android.content.Context;  
**import** android.content.res.AssetManager;  
**import** android.os.Bundle;  
**import** android.support.design.widget.FloatingActionButton;  
**import** android.support.v7.app.AppCompatActivity;  
**import** android.support.v7.widget.Toolbar;  
**import** android.text.Html;  
**import** android.text.InputType;  
**import** android.text.TextUtils;  
**import** android.view.KeyEvent;  
**import** android.view.Menu;  
**import** android.view.MenuItem;  
**import** android.view.View;  
**import** android.view.inputmethod.EditorInfo;  
**import** android.view.inputmethod.InputMethodManager;  
**import** android.widget.EditText;  
**import** android.widget.TextView;  
**import** android.widget.Toast;  
  
**import** java.io.IOException;  
**import** java.io.InputStream;  
**import** java.util.ArrayList;  
  
  
**public class** AnagramsActivity **extends** AppCompatActivity {  
  
 **public static final** String START\_MESSAGE = **"Find as many words as possible that can be formed by adding one letter to <big>%s</big> (but that do not contain the substring %s)."**;  
 **private** AnagramDictionary dictionary;  
 **private** String currentWord;  
 **private** ArrayList<String> anagrams;  
  
 @Override  
 **protected void** onCreate(Bundle savedInstanceState) {  
 **super**.onCreate(savedInstanceState);  
 setContentView(R.layout.activity\_anagrams);  
 Toolbar toolbar = (Toolbar) findViewById(R.id.toolbar);  
 setSupportActionBar(toolbar);  
 AssetManager assetManager = getAssets();  
  
 **try** {  
 InputStream inputStream = assetManager.open(**"words.txt"**);  
 dictionary = **new** AnagramDictionary(inputStream);  
 } **catch** (IOException e) {  
 Toast toast = Toast.makeText(**this**, **"Could not load dictionary"**, Toast.LENGTH\_LONG);  
 toast.show();  
 }  
  
 *// Set up the EditText box to process the content of the box when the user hits 'enter'* **final** EditText editText = (EditText) findViewById(R.id.editText);  
 editText.setRawInputType(InputType.TYPE\_CLASS\_TEXT);  
 editText.setImeOptions(EditorInfo.IME\_ACTION\_GO);  
 editText.setOnEditorActionListener(**new** TextView.OnEditorActionListener() {  
 @Override  
 **public boolean** onEditorAction(TextView v, **int** actionId, KeyEvent event) {  
 **boolean** handled = **false**;  
 **if** (actionId == EditorInfo.IME\_ACTION\_GO) {  
 processWord(editText);  
 handled = **true**;  
 }  
 **return** handled;  
 }  
 });  
 }  
  
 **private void** processWord(EditText editText) {  
 TextView resultView = (TextView) findViewById(R.id.resultView);  
 String word = editText.getText().toString().trim().toLowerCase();  
 **if** (word.length() == 0) {  
 **return**;  
 }  
 String color = **"#cc0029"**;  
 **if** (dictionary.isGoodWord(word, currentWord) && anagrams.contains(word)) {  
 anagrams.remove(word);  
 color = **"#00aa29"**;  
 } **else** {  
 word = **"X "** + word;  
 }  
  
 resultView.append(Html.fromHtml(String.format(**"<font color=%s>%s</font><BR>"**, color, word)));  
 editText.setText(**""**);  
  
 FloatingActionButton floatingActionButton = (FloatingActionButton) findViewById(R.id.fab);  
 floatingActionButton.show();  
 }  
  
 @Override  
 **public boolean** onCreateOptionsMenu(Menu menu) {  
 *// Inflate the menu; this adds items to the action bar if it is present.* getMenuInflater().inflate(R.menu.menu\_anagrams, menu);  
 **return true**;  
 }  
  
 @Override  
 **public boolean** onOptionsItemSelected(MenuItem item) {  
 *// Handle action bar item clicks here. The action bar will  
 // automatically handle clicks on the Home/Up button, so long  
 // as you specify a parent activity in AndroidManifest.xml.* **int** id = item.getItemId();  
  
 *//noinspection SimplifiableIfStatement* **if** (id == R.id.action\_settings) {  
 **return true**;  
 }  
  
 **return super**.onOptionsItemSelected(item);  
 }  
  
 **public boolean** defaultAction(View view) {  
 TextView gameStatus = (TextView) findViewById(R.id.gameStatusView);  
 FloatingActionButton fab = (FloatingActionButton) findViewById(R.id.fab);  
 EditText editText = (EditText) findViewById(R.id.editText);  
 TextView resultView = (TextView) findViewById(R.id.resultView);  
  
 **if** (currentWord == **null**) {  
 currentWord = dictionary.pickGoodStarterWord();  
 anagrams = dictionary.getAnagramsWithTwoMoreLetters(currentWord);  
 gameStatus.setText(Html.fromHtml(String.format(START\_MESSAGE, currentWord.toUpperCase(), currentWord)));  
 fab.setImageResource(android.R.drawable.ic\_menu\_help);  
 fab.hide();  
  
 resultView.setText(**""**);  
 editText.setText(**""**);  
 editText.setEnabled(**true**);  
 editText.requestFocus();  
  
 InputMethodManager inputMethodManager = (InputMethodManager) getSystemService(Context.INPUT\_METHOD\_SERVICE);  
 inputMethodManager.showSoftInput(editText, InputMethodManager.SHOW\_IMPLICIT);  
 } **else** {  
 editText.setText(currentWord);  
 editText.setEnabled(**false**);  
 fab.setImageResource(android.R.drawable.ic\_media\_play);  
  
 currentWord = **null**;  
 resultView.append(TextUtils.join(**"\n"**, anagrams));  
 gameStatus.append(**"\nHit 'Play' to start again"**);  
 }  
  
 **return true**;  
 }  
}