TSHTHA094_CSC2002S_PCP2

Thabelo Tshikalange

August 2022

1 HungryWordMover

```
1  package typingTutor;
2
3  import java.util.concurrent.*;
4  import java.util.concurrent.atomic.AtomicBoolean;
5
6  public class HungryWordMover extends Thread {
7      private FallingWord myWord;
8      private AtomicBoolean done;
9      private AtomicBoolean pause;
10     private Score score;
11     private FallingWord[] words;
12     CountDownLatch startLatch; //so all can start at once
13
14     HungryWordMover( FallingWord word) {
15         myWord = word;
16     }
17
18     HungryWordMover( FallingWord word,WordDictionary dict,
           Score score,
19             CountDownLatch startLatch, AtomicBoolean d,
                   AtomicBoolean p) {
20         this(word);
21         this.startLatch = startLatch;
22         this.score=score;
23         this.done=d;
24         this.pause=p;
25     }
26
27     HungryWordMover( FallingWord word,WordDictionary dict,
           Score score,
28     CountDownLatch startLatch, AtomicBoolean d, AtomicBoolean p
           ,FallingWord[] words) {
29         this(word);
30         this.startLatch = startLatch;
31         this.score=score;
```

```
32          this.done=d;
33          this.pause=p;
34          this.words = words;
35      }
36
37      public void run() {
38
39          //System.out.println(myWord.getWord() + " falling speed
                = " + myWord.getSpeed());
40          try {
41              System.out.println(myWord.getWord() + " waiting to
                    start " );
42              startLatch.await();
43          } catch (InterruptedException e1) {
44              // TODO Auto-generated catch block
45              e1.printStackTrace();
46          } //wait for other threads to start
47          System.out.println(myWord.getWord() + " started" );
48          while (!done.get()) {
49              //animate the word
50              //System.out.println("The hungry word is " +
                    TypingTutorApp.hungrywWord.getWord()+ " is the
                    same as "+myWord.getWord() );
51              while (!myWord.droppedx() && !done.get()) {
52                  if (myWord.equals(TypingTutorApp.hungrywWord)){
53                      myWord.dropx(10);
54                      for (int i = 0 ; i <words.length; i++){
55                          if(!words[i].equals(TypingTutorApp.
                                hungrywWord)){
56                              if (words[i].getY() == myWord.getY
                                    ()){
57                                  //System.out.println("The
                                        hungry word is at " + myWord
                                        .getY()+ "the other at"+
                                        words[i].getY());
58                                  if ((myWord.getX()+myWord.
                                        getWord().length()/2 -words[
                                        i].getX()) < 1){
59                                      score.missedWord();
60                                      words[i].resetWord();
61                                  }
62                                  if ((myWord.getWord().length()
                                        /2- myWord.getX() -words[i].
                                        getX()) < 1){
63                                      score.missedWord();
64                                      words[i].resetWord();
65                                  }
66                              }
67                          }
68                      }
```

```
69                  }
70                      try {
71                          sleep(myWord.getSpeed());
72                      } catch (InterruptedException e) {
73                          // TODO Auto-generated catch block
74                          e.printStackTrace();
75                      };
76                      while(pause.get()&&!done.get()) {};
77              }
78              if (!done.get() && myWord.droppedx()) {
79                  score.missedWord();
80                  myWord.resetWordx();
81              }
82              myWord.resetWordx();
83          }
84      }
85
86  }
```

2 Changed Classes

2.1 CatchWord

```
87  while (i<noWords) {
88          while(pause.get()) {};
89          int counter = 0;
90          int Y_value = 0;
91          if(words[i].getWord().equals(target)){
92              for(int j = 0; j < noWords; j++ ){
93                  if (words[j].getWord().equals(target)){
94                      if (words[j].getY()>Y_value){
95                          Y_value = words[j].getY();
96                          counter = j;
97                      }
98                  }
99              }
100             if (words[counter].matchWord(target)) {
101                 System.out.println( " score! '" + target); //
                        for checking
102                 score.caughtWord(target.length());
103                     /FallingWord.increaseSpeed();
104                 break;
105             }
106         }
107     i++;
108 }
```

And you can reference line **??** in the code!

2.2 TypingTutorApp

The changes made to the TypingTutorApp where to be able to make new instances of the HungryWordMover such that we can implement the mover class and allow use to change how the hungry word moves across the window.

```
109 for (int i=0;i<noWords;i++) {
110             if(i == random){
111                 words[i]=new FallingWord(dict.getNewWord());
112                 words[i].setPos(0, gameWindow.getHeight()/2+1);
113                 hungrywWord = words[random];
114             //System.out.println( "This is word "+hungrywWord.
                    getWord()+ " this is position X "+ hungrywWord.
                    getX()+" this is position Y "+ hungrywWord.getY
                    ());
115             }
116             else{
117             words[i]=new FallingWord(dict.getNewWord(),
                    gameWindow.getValidXpos(),yLimit);
118             }
119         }
120         //create threads to move them
121         for (int i=0;i<noWords;i++) {
122                 wrdShft[i] = new WordMover(words[i],dict,score,
                        startLatch,done,pause);
123                 wordMovers[i] = new HungryWordMover(words[i],
                        dict,score,startLatch,done,pause,words);
124         }
125         //word movers waiting on starting line
126         for (int i=0;i<noWords;i++) {
127             wrdShft[i].start();
128             wordMovers[i].start();
129         }
130     }
```