

[View Javadoc](#)

```

1  package model.element;
2
3  import java.util.Observable;
4
5  import contract.IElement;
6  import contract.ILevelMap;
7  import entity.Level;
8  import model.element.mobile.*;
9  import model.element.motionless.*;
10
11  /**
12   * The Class Model.
13   *
14   * @author Group 5
15   */
16  public class LevelMap extends Observable implements ILevelMap {
17
18      /** The Levelmap; */
19      private IElement[][] levelmap = new IElement[40][22];
20
21      /**
22       * Instantiates a new LevelMap.
23       *
24       * @param Level
25       *             The Level.
26       */
27      public LevelMap(Level level) {
28
29          for(int x=0; x<40; x++) {
30              for(int y=0; y<22; y++) {
31
32                  String E = level.getElement(x, y);
33                  switch(E) {
34                      case "B":
35                          this.setElement(x, y, new UnbreakableBlock(x, y, this));
36                          break;
37                      case "C":
38                          this.setElement(x, y, new Block(x, y, this));
39                          break;
40                      case "R":
41                          this.setElement(x, y, new Rock(x, y, this));
42                          break;
43                      case "E":
44                          this.setElement(x, y, new Enemy(x, y, this));
45                          break;
46                      case "S":
47                          this.setElement(x, y, new Exit(x, y, this));
48                          break;
49                      case "P":
50                          this.setElement(x, y, new Player(x, y, this));
51                          break;
52                      case "D":
53                          this.setElement(x, y, new Diamond(x, y, this));
54                          break;
55                      default :
56                          this.setElement(x, y, null);
57
58                  }
59
60              }
61
62          }
63
64      }
65
66      /**

```

```

67      * (non-Javadoc)
68      *
69      * @see contract.ILevelMap#getPlayer()
70      */
71  @Override
72  public IElement getPlayer() {
73
74      for(int x=0; x<40; x++) {
75          for(int y=0; y<22; y++) {
76
77              IElement element = this.getElement(x, y);
78
79              if(element instanceof Player) {
80                  return element;
81              }
82          }
83      }
84
85      return null;
86  }
87
88  /*
89  * (non-Javadoc)
90  *
91  * @see contract.ILevelMap#getElement()
92  */
93  @Override
94  public IElement getElement(int x, int y) {
95
96      return this.levelmap[x][y];
97  }
98
99  /*
100  * (non-Javadoc)
101  *
102  * @see contract.ILevelMap#setElement()
103  */
104  @Override
105  public void setElement(int x, int y, IElement element) {
106      this.levelmap[x][y] = element;
107      this.setChanged();
108      this.notifyObservers();
109  }
110
111  /*
112  * (non-Javadoc)
113  *
114  * @see contract.ILevelMap#removeElement()
115  */
116  @Override
117  public void removeElement(int x, int y) {
118
119      this.setElement(x, y, null);
120  }
121
122  /*
123  * (non-Javadoc)
124  *
125  * @see contract.ILevelMap#getObservable()
126  */
127  public Observable getObservable() {
128      return this;
129  }
130
131  /*
132  * (non-Javadoc)
133  *
134  * @see contract.ILevelMap#popDiamond()
135  */

```

```
136 @Override
137 public void popDiamond(int a, int b) {
138
139     for(int x=a-1; x<a+2; x++) {
140         for(int y =b-1; y<b+2; y++) {
141             this.setElement(x, y, new Diamond(x, y, this));
142         }
143     }
144 }
145
146 }
```

Copyright © 2019. All rights reserved.