View Javadoc

```
1   package model.element.mobile;
2
3   import java.awt.Image;
4   import java.io.IOException;
5
6   import javax.imageio.ImageIO;
7
8   import contract.ElementType;
9   import contract.IElement;
10  import contract.ILevelMap;
11  import model.element.LevelMap;
12
13  /**
14   * The Class Diamond.
15   *
16   * @author Group 5
17   */
18  public class Rock implements IElement {
19
20      /** The score */
21      private int score = 0;
22
23      /** The exist boolean */
24      private boolean exist = true;
25
26      /** The elementType */
27      private ElementType elementType = ElementType.ROCK;
28
29      /** The x */
30      private int x;
31
32      /** The y */
33      private int y;
34
35      /** The levelmap */
36      private ILevelMap levelmap;
37
38      /** The image */
39      private static Image image;
40
41      /** The imageName */
42      private String imageName = "Rocher";
43
44      /**
45       * constructor to build and place Rock
46       *
47       * @param x
48       *              The x.
49       * @param y
50       *              The y.
51       * @param levelMap
52       *              The levelMap.
53       *
54       */
55      public Rock(final int x, final int y, LevelMap levelMap) {
56              this.setX(x);
57              this.setY(y);
58              this.setImageName(imageName);
59              this.loadImage();
60              this.setLevelmap(levelMap);
61
62      }
63
64      /**
65       * Get x position of Rock
66       *
67       * @return x
68       *
69       */
70      @Override
71      public int getX() {
72
73              return this.x;
74      }
75
76      /**
77       * Set x position of Rock
78       *
79       * @param x
80       *              The x.
81       *
82       */
83      @Override
84      public void setX(int x) {
85
86              this.x = x;
87      }
88
89      /**
90       * Get y position of Rock
91       *
```

```
92       * @return y
93       *
94       */
95      @Override
96      public int getY() {
97
98              return this.y;
99      }
100
101     /**
102      * Set y position of Rock
103      *
104      * @param y
105      *              The y.
106      *
107      */
108     @Override
109     public void setY(int y) {
110
111             this.y = y;
112     }
113
114     /**
115      * Move up the player when up key are pressed
116      */
117     @Override
118     public void moveUp() {
119
120             this.setY(this.getY() - 1);
121
122             this.levelmap.setElement(this.getX(), this.getY(), this);
123             this.levelmap.removeElement(getX(), getY()+1);
124
125     }
126
127     /**
128      * Move down the player when down key are pressed
129      *
130      */
131     @Override
132     public void moveDown() {
133             this.setY(this.getY() + 1);
134
135             this.levelmap.setElement(this.getX(), this.getY(), this);
136             this.levelmap.removeElement(getX(), getY()-1);
137     }
138
139     /**
140      * Move left the player when left key are pressed
141      *
142      */
143     public void moveLeft() {
144             this.setX(this.getX() - 1);
145
146             this.levelmap.setElement(this.getX(), this.getY(), this);
147             this.levelmap.removeElement(getX()+1, getY());
148     }
149     /**
150      * Move right the player when right key are pressed
151      *
152      */
153     @Override
154     public void moveRight() {
155             this.setX(this.getX() + 1);
156
157             this.levelmap.setElement(this.getX(), this.getY(), this);
158             this.levelmap.removeElement(getX()-1, getY());
159     }
160
161     /**
162      * do nothing the player when player don't move
163      *
164      */
165     @Override
166     public void doNothing() {
167             this.setY(this.getY());
168
169             this.levelmap.setElement(this.getX(), this.getY(), this);
170     }
171
172     /**
173      * Get image of Rock
174      *
175      * @return image
176      *
177      */
178     @Override
179     public Image getImage() {
180
181             return Rock.image;
182     }
183
184     /**
```

```
185      * Set image of Rock
186      *
187      * @param image
188      *                      The image.
189      *
190      */
191     @Override
192     public void setImage(Image image) {
193
194             Rock.image = image;
195     }
196
197     /**
198      * Load image of Rock
199      *
200      */
201     @Override
202     public void loadImage() {
203
204             Image img = null;
205             try {
206                     img = ImageIO.read(getClass().getClassLoader().getResourceAsStream("images/" + this.getImageName() + ".png"));
207             }
208             catch(IOException e) {
209                     e.printStackTrace();
210             }
211             this.setImage(img);
212     }
213
214     /**
215      * Get image name of Rock
216      *
217      * @return imgaName
218      *
219      */
220     @Override
221     public String getImageName() {
222
223             return this.imageName;
224     }
225
226     /**
227      * Set image name of Rock
228      *
229      * @param imageName
230      *                              The image name.
231      *
232      */
233     @Override
234     public void setImageName(String imageName) {
235
236             this.imageName = imageName;
237     }
238
239     /**
240      * check existing of Rock
241      *
242      * @return exist
243      *
244      */
245     @Override
246     public boolean isExist() {
247
248             return this.exist;
249     }
250
251     /**
252      * set exist verification of Rock
253      *
254      * @param exist
255      *                      The exist state.
256      *
257      */
258     @Override
259     public void setExist(boolean exist) {
260             this.exist = exist;
261     }
262
263     /**
264      * Get level
265      *
266      * @return level map
267      *
268      */
269     public ILevelMap getLevelmap() {
270             return levelmap;
271     }
272
273     /**
274      * Set level
275      *
276      * @param levelmap
277      *                      The LevelMap.
```

```
278      *
279      */
280     public void setLevelmap(ILevelMap levelmap) {
281             this.levelmap = levelmap;
282     }
283
284     /**
285      * Get score of collected diamond
286      *
287      * @return score
288      *
289      */
290     @Override
291     public int getScore() {
292             return score;
293     }
294
295     /**
296      * Set score of collected diamond
297      *
298      * @param score
299      *                      The score.
300      *
301      */
302     @Override
303     public void setScore(int score) {
304             this.score = score;
305     }
306
307     /**
308      * Get element type of Rock
309      *
310      * @return element type
311      *
312      */
313     @Override
314     public ElementType getElementType() {
315             return elementType;
316     }
317
318     /**
319      * Set element type of Rock
320      *
321      * @param elementType
322      *                              The elementType.
323      *
324      */
325     @Override
326     public void setElementType(ElementType elementType) {
327             this.elementType = elementType;
328     }
329
330 }
```