Testing Report

Bowen Tao 1622211

2/11/2013

1 Introduction

The RobotMap file in our project plays a vital role in the procedure to transform the data between DTD map file which is defined by XML and various objects (for example Closure class). The file must read carefully from the XML file in order to decode the information stored in the file. Also, it will rebuild the data from the XML file and translate them to a format that could be passed to the objects. The different kinds of objects are based on the attributes which distinguishes from each other such as closure, obstacles, roads, disasters, intersections, and unexplored zone. When reading a valid XML file, the RobotMap will firstly load the XML file which defines map's information and then get all of attributes. Then, by reading through the XML, it will create all the attributes in the GUI (the virtual map) based on their information which is defined by XML file. Meanwhile, it will also get the unexplored area information and simultaneously the area will be divided into several temporary blocks and add them into the UnexploredZone object so that a rectangle area with gray color will be created on the map to represent the unexplored area. And if robot wants to enter this area, it can explore the block one by one.

2 Test Description

To test whether the information has been correctly transferred from the XML file, a branch of tests will be conducted by using JUnit. The following test cases have been utilized in this project to test whether the important properties such as, boundary, closure, disaster have been read correctly by the RobotMap.

1. boundaryTest: Test if the boundary gets changed during the transform process and finally all of tests are successful.
2. disaterTest: Test if the disaster coordinate and size get changed during the transform process and finally all of tests are successful.
3. intersectionTest: Test if the intersection coordinate and size get changed during the transform process and finally all of tests are successful.
4. obstaclesTest: Test if the obstacle coordinate and size get changed during the transform process and finally all of tests are successful.
5. roadsTest: Test if the road coordinate and size get changed during the transform process and finally all of tests are successful.
6. unexploredZonesTest: Test if the boundary of unexplored area gets changed during the transform process and finally all of tests are successful.
7. closureTest: Test if the closure coordinate and size get changed duringthe transform process and finally all of tests are successful.
8. RobotTest: Test if the start point where robot stands initially gets changed during the transform process and finally all of tests are successful.

A Test Code

The code below is taken from RobotMapTest.

```
1  package mapDataStructure;
2
3  import static org.junit.Assert.*;
4
5  import java.util.ArrayList;
6
7  import junit.framework.Assert;
8
9  import org.junit.Before;
10 import org.junit.Test;
11
12 /**
13  * @author Bowen Tao
14  * @filename RobotMapTest.java
15  */
16 @SuppressWarnings("deprecation")
17
18 public class RobotMapTest {
19
20   RobotMap rmap;
21
22   @Before
23   public void before(){
24     rmap = new RobotMap();
25     rmap.loadMap("src/map4.xml");
26   }
27
28   @Test
29   public void boundaryTest(){
30     double height=rmap.getHeight();
31     double width=rmap.getWidth();
32     Assert.assertEquals(180,height,0.00001);
33     Assert.assertEquals(240, width, 0.00001);
34   }
35
36   @Test
37   public void disaterTest() {
38     int disaterSize =rmap.getDisasterZones().size();
39     assertEquals(2, disaterSize);
40
41     double x =rmap.getDisasterZones().get(0).getLocation().getX();
42     double y =rmap.getDisasterZones().get(0).getLocation().getY();
43     Assert.assertEquals(70, x, 0.00001);
44     Assert.assertEquals(40, y, 0.00001);
45
46     double x1 =rmap.getDisasterZones().get(1).getLocation().getX();
47     double y1 =rmap.getDisasterZones().get(1).getLocation().getY();
48     Assert.assertEquals(120, x1, 0.00001);
49     Assert.assertEquals(80, y1, 0.00001);
50
51   }
52
53   @Test
54   public void intersectionTest() {
55     int intersectionSize =rmap.getIntersections().size();
56     assertEquals(8, intersectionSize);
57
58     double x =rmap.getIntersections().get(0).getLocation().getX();
59     double y =rmap.getIntersections().get(0).getLocation().getY();
60     Assert.assertEquals(120, x, 0.00001);
61     Assert.assertEquals(40, y, 0.00001);
62
63     double x1 =rmap.getIntersections().get(1).getLocation().getX();
64     double y1 =rmap.getIntersections().get(1).getLocation().getY();
65     Assert.assertEquals(100, x1, 0.00001);
66     Assert.assertEquals(40, y1, 0.00001);
67
68     double x2 =rmap.getIntersections().get(2).getLocation().getX();
69     double y2 =rmap.getIntersections().get(2).getLocation().getY();
70     Assert.assertEquals(100, x2, 0.00001);
71     Assert.assertEquals(80, y2, 0.00001);
72
```

```
73      double x3 =rmap.getIntersections().get(3).getLocation().getX();
74      double y3 =rmap.getIntersections().get(3).getLocation().getY();
75      Assert.assertEquals(120, x3, 0.00001);
76      Assert.assertEquals(80, y3, 0.00001);
77
78      double x4 =rmap.getIntersections().get(4).getLocation().getX();
79      double y4 =rmap.getIntersections().get(4).getLocation().getY();
80      Assert.assertEquals(160, x4, 0.00001);
81      Assert.assertEquals(40, y4, 0.00001);
82
83      double x5 =rmap.getIntersections().get(5).getLocation().getX();
84      double y5 =rmap.getIntersections().get(5).getLocation().getY();
85      Assert.assertEquals(160, x5, 0.00001);
86      Assert.assertEquals(80, y5, 0.00001);
87
88      double x6 =rmap.getIntersections().get(6).getLocation().getX();
89      double y6 =rmap.getIntersections().get(6).getLocation().getY();
90      Assert.assertEquals(100, x6, 0.00001);
91      Assert.assertEquals(120, y6, 0.00001);
92
93      double x7 =rmap.getIntersections().get(7).getLocation().getX();
94      double y7 =rmap.getIntersections().get(7).getLocation().getY();
95      Assert.assertEquals(160, x7, 0.00001);
96      Assert.assertEquals(120, y7, 0.00001);
97
98    }
99
100   @Test
101   public void obstaclesTest() {
102     int obstacleSize =rmap.getObstacles().size();
103     assertEquals(2, obstacleSize);
104
105     double x =rmap.getObstacles().get(0).getLocation().getX();
106     double y =rmap.getObstacles().get(0).getLocation().getY();
107     Assert.assertEquals(200, x, 0.00001);
108     Assert.assertEquals(40, y, 0.00001);
109
110     double x1 =rmap.getObstacles().get(1).getLocation().getX();
111     double y1 =rmap.getObstacles().get(1).getLocation().getY();
112     Assert.assertEquals(100, x1, 0.00001);
113     Assert.assertEquals(105, y1, 0.00001);
114
115   }
116
117   @Test
118   public void roadsTest() {
119     int roadSize =rmap.getRoads().size();
120     assertEquals(6, roadSize);
121
122     double startX =rmap.getRoads().get(0).getStart().getX();
123     double startY =rmap.getRoads().get(0).getStart().getY();
124     Assert.assertEquals(120, startX, 0.00001);
125     Assert.assertEquals(0, startY, 0.00001);
126     double endX =rmap.getRoads().get(0).getEnd().getX();
127     double endY =rmap.getRoads().get(0).getEnd().getY();
128     Assert.assertEquals(120, endX, 0.00001);
129     Assert.assertEquals(80, endY, 0.00001);
130
131     double startX1 =rmap.getRoads().get(1).getStart().getX();
132     double startY1 =rmap.getRoads().get(1).getStart().getY();
133     Assert.assertEquals(80, startX1, 0.00001);
134     Assert.assertEquals(40, startY1, 0.00001);
135     double endX1 =rmap.getRoads().get(1).getEnd().getX();
136     double endY1 =rmap.getRoads().get(1).getEnd().getY();
137     Assert.assertEquals(200, endX1, 0.00001);
138     Assert.assertEquals(40, endY1, 0.00001);
139
140     double startX2 =rmap.getRoads().get(2).getStart().getX();
141     double startY2 =rmap.getRoads().get(2).getStart().getY();
142     Assert.assertEquals(100, startX2, 0.00001);
143     Assert.assertEquals(40, startY2, 0.00001);
144     double endX2 =rmap.getRoads().get(2).getEnd().getX();
145     double endY2 =rmap.getRoads().get(2).getEnd().getY();
146     Assert.assertEquals(100, endX2, 0.00001);
147     Assert.assertEquals(120, endY2, 0.00001);
148
```

```
149     double startX3 =rmap.getRoads().get(3).getStart().getX();
150     double startY3 =rmap.getRoads().get(3).getStart().getY();
151     Assert.assertEquals(160, startX3, 0.00001);
152     Assert.assertEquals(40, startY3, 0.00001);
153     double endX3 =rmap.getRoads().get(3).getEnd().getX();
154     double endY3 =rmap.getRoads().get(3).getEnd().getY();
155     Assert.assertEquals(160, endX3, 0.00001);
156     Assert.assertEquals(120, endY3, 0.00001);
157
158     double startX4 =rmap.getRoads().get(4).getStart().getX();
159     double startY4 =rmap.getRoads().get(4).getStart().getY();
160     Assert.assertEquals(100, startX4, 0.00001);
161     Assert.assertEquals(80, startY4, 0.00001);
162     double endX4 =rmap.getRoads().get(4).getEnd().getX();
163     double endY4 =rmap.getRoads().get(4).getEnd().getY();
164     Assert.assertEquals(160, endX4, 0.00001);
165     Assert.assertEquals(80, endY4, 0.00001);
166
167     double startX5 =rmap.getRoads().get(5).getStart().getX();
168     double startY5 =rmap.getRoads().get(5).getStart().getY();
169     Assert.assertEquals(100, startX5, 0.00001);
170     Assert.assertEquals(120, startY5, 0.00001);
171     double endX5 =rmap.getRoads().get(5).getEnd().getX();
172     double endY5 =rmap.getRoads().get(5).getEnd().getY();
173     Assert.assertEquals(160, endX5, 0.00001);
174     Assert.assertEquals(120, endY5, 0.00001);
175
176   }
177
178   @Test
179   public void unexploredZonesTest() {
180     double x =rmap.getUnexploredZones().get(0).getLocation().getX();
181     double y =rmap.getUnexploredZones().get(0).getLocation().getY();
182     Assert.assertEquals(0, x, 0.00001);
183     Assert.assertEquals(41, y, 0.00001);
184
185
186   }
187
188   @Test
189   public void closureTest() {
190     int closureSize = 0;
191     ArrayList <Road> roads = new ArrayList<Road>();
192     roads = rmap.getRoads();
193     for (Road r: roads){
194       if (r.isClosed()){
195         closureSize++;
196       }
197     }
198     assertEquals(1, closureSize);
199   }
200
201   @Test
202   public void RobotTest(){
203     double x=rmap.getRobot().getRobotLocation().getX();
204     double y=rmap.getRobot().getRobotLocation().getY();
205     Assert.assertEquals(120, x, 0.00001);
206     Assert.assertEquals(0, y, 0.00001);
207
208   }
209
210 }
```