SEP Individual Testing Report

Yatong ZHOU

ID: 1204471

21 October 2012

0.1 Introduction

As I did not participate much in the map data structure development. I was responsible for testing the pixel structure of our code. Pixel is the basic element structure of our map. Since pixel was involved in the path finding algorithm, it has complex structure.

My test cases will create a 10 * 10 map structure at the beginning. Some pixel elements will be obtained from the map, they are used to test the methods of the pixel structure.

All the tests are using Java Junit test tool,and these tests will be purely black box observational testing as the results can only be observed by eye.

Tests URL:
https://version-control.adelaide.edu.au/svn/sep2012-13/Code/Working Copy-/Version 0.3/src/Tests/PixelTest.java

0.2 Test Cases

0.2.1 Test 1: Pixel position test

Find a pixel from the map structure in a random location, for example, the pixel which has coordinate (6, 4), and check if its x, y coordinates match the location of the map.
Pass/Fail Criteria:
The actual coordinates of x and y are matched to the location of the map, the test passes. Otherwise, this test fails.
Result: Pass

0.2.2 Test 2: Pixel neighbours check test

Find an random pixel's four neighbours' coordinates and check if they are actual equal the parameters of the target pixel's neighbours.

Pass/Fail Criteria:
If all of the four pixels' coordinates are equal the structure of the target, the test passes. Otherwise, it fails.
Result: Pass

0.2.3 Test 3: Pixel as wall test

Find a random pixel and set it as wall, check if its value is equal to the wall value.

Pass/Fail Criteria:
This test will pass if the value of the pixel is equal to the value stand for the wall, otherwise, it will fail.
Result: Pass

0.2.4 Test 4: Pixel as NoGoZone test

Find a random pixel and set it as NoGoZone, check if its value is equal to the NoGoZone value.

Pass/Fail Criteria:
This test will pass if the value of the pixel is equal to the value stand for the NoGoZone, otherwise ,it will fail.
Result: Pass

0.2.5 Test 5: Parent pixel test

Set a pixel as the parent of another one, and check if it the same as the get parent method value from the child pixel.

Pass/Fail Criteria:
This test will pass if the two pixel object are the same, otherwise, it will fail.
Result: Pass

0.2.6 Test 6: Buffer zone value test

Set a target pixel as the central pixel of an area of buffer zone, randomly choose some of the pixel closed to the target pixel and check if it an buffer zone pixel.

Pass/Fail Criteria:
This test will pass if all of the chosen pixels are marked as the buffer zone pixels, otherwise, this test fails.

Result: Pass

0.3 JUnit Test Cases Code

```
package Tests;

import static org.junit.Assert.*;
import org.junit.Test;
import MapStructure.Map;
import MapStructure.Pixel;

/**
 *
 * @author Yatong ZHOU
 *
 */
public class PixelTest {

        Map map =  new MapStructure.Map("MAP00000", 10, 10, 0, 0, 1);

        Pixel centralPixel = map.findPixel(6, 4);
        Pixel northPixel = map.findPixel(6, 3);
        Pixel southPixel = map.findPixel(6, 5);
        Pixel eastPixel = map.findPixel(7, 4);
        Pixel westPixel = map.findPixel(5, 4);

        @Test
        public void testPixelPosition() {
                assertTrue(centralPixel.getxPos() == 6);
                assertTrue(centralPixel.getyPos() == 4);
        }

        @Test
        public void fourDirectionTest() {
                assertTrue("North Test", centralPixel.getN().equals(northPixel));
                assertTrue("South Test", centralPixel.getS().equals(southPixel));
                assertTrue("East Test", centralPixel.getE().equals(eastPixel));
                assertTrue("West Test", centralPixel.getW().equals(westPixel));
        }

        @Test
        public void setWallTest() {
                centralPixel.setWall();
                assertEquals(centralPixel.getValue(), Pixel.WALL);
        }

        @Test
        public void noGoZoneTest() {
                centralPixel.setNoGoZone();
                assertEquals(centralPixel.getValue(), Pixel.NOGOZONE);
        }

        @Test
        public void parentTest() {
                centralPixel.setParent(eastPixel);
                assertEquals(centralPixel.getParent(), eastPixel);
        }

        @Test
        public void testBufferedZone() {
                centralPixel.setBufferZone();
                assertEquals(map.findPixel(centralPixel.getxPos() - 3, centralPixe
                assertEquals(map.findPixel(centralPixel.getxPos() + 3, centralPixe
                assertEquals(map.findPixel(centralPixel.getxPos() - 1, centralPixe
        }

}
```