

70094 14

Mock Objects

Submitters

sf23

Shihan Fu

zj123

Ze Jin

Emarking

Final Tests**TestSummary.txt: 1/1****Shihan Fu - sf23:v5**

```
1: Final Tests: Summary for sf23 of v5
2: -----
3:
4:   Public Tests:
5:     Compiles:           1 / 1
6:     Tests Pass:         1 / 1
7:     Coverage and Style Checks: 1 / 1
8:
9: Git Repo: git@gitlab.doc.ic.ac.uk:lab2324_spring/SE_Design_Ex4_sf23.git
10: Commit ID: 69779
```

**Good.****Great work. See some comments below. 17/20**

Final Tests

CameraTest.java: 1/2

Shihan Fu - sf23:v5

```

1: package ic.doc.camera;
2:
3: import org.jmock.Expectations;
4: import org.jmock.integration.junit4.JUnitRuleMockery;
5: import org.junit.Rule;
6: import org.junit.Test;
7: import static org.junit.Assert.assertFalse;
8: import static org.junit.Assert.assertTrue;
9:
10: public class CameraTest {
11:
12:     @Rule public JUnitRuleMockery context = new JUnitRuleMockery();
13:
14:     Sensor sensor = context.mock(Sensor.class);
15:     MemoryCard memoryCard = context.mock(MemoryCard.class);
16:
17:     @Test
18:     public void switchingTheCameraOnPowersUpTheSensor() {
19:         Camera camera = new Camera(sensor, memoryCard);
20:         // write your test here
21:         context.checking(
22:             new Expectations() {
23:                 {
24:                     exactly(1).of(sensor).powerUp();
25:                 }
26:             });
27:         camera.powerOn();
28:         assertTrue(camera.get_power());
29:     }
30:
31:     @Test
32:     public void switchingTheCameraOffPowersDownTheSensor() {
33:         Camera camera = new Camera(sensor, memoryCard);
34:         // write your test here
35:         context.checking(
36:             new Expectations() {
37:                 {
38:                     exactly(1).of(sensor).powerUp();
39:                     exactly(1).of(sensor).powerDown();
40:                 }
41:             });
42:         camera.powerOn();
43:         camera.powerOff();
44:         assertFalse(camera.get_power());
45:     }
46:
47:     @Test
48:     public void pressShutterWhenCameraPowerOff() {
49:         Camera camera = new Camera(sensor, memoryCard);
50:
51:         context.checking(
52:             new Expectations() {
53:                 {
54:                     never(sensor);
55:                 }
56:             });
57:         // assumption: after creating a camera, its mode is power off
58:         camera.pressShutter();
59:     }
60:
61:     @Test
62:     public void pressingShutterWithPowerOnCopiesData() {
63:         Camera camera = new Camera(sensor, memoryCard);
64:         context.checking(
65:             new Expectations() {
66:

```

Tell, don't ask.

Final Tests

CameraTest.java: 2/2

Shihan Fu - sf23:v5

```

67:         exactly(1).of(sensor).powerUp();
68:         exactly(1).of(sensor).readData();
69:         will(returnValue(new byte[0]));
70:         exactly(1).of(memoryCard).write(with(any(byte[].class)));
71:     }
72: });
73:
74: camera.powerOn();
75: camera.pressShutter();
76: assertTrue(camera.get_isWritingData());
77: }
78:
79: @Test
80: public void switchingTheCameraOffWithoutPowersDownTheSensor() {
81:     Camera camera = new Camera(sensor, memoryCard);
82:     context.checking(
83:         new Expectations() {
84:             {
85:                 exactly(1).of(sensor).powerUp();
86:                 exactly(1).of(sensor).readData();
87:                 will(returnValue(new byte[0]));
88:                 oneOf(memoryCard).write(with(any(byte[].class)));
89:                 never(sensor).powerDown();
90:             }
91:         });
92:
93:     camera.powerOn();
94:     camera.pressShutter();
95:     camera.powerOff();
96: }
97:
98: @Test
99: public void completeWritingPowerDown() {
100:     Camera camera = new Camera(sensor, memoryCard);
101:     context.checking(
102:         new Expectations() {
103:             {
104:                 exactly(1).of(sensor).powerUp();
105:                 exactly(1).of(sensor).readData();
106:                 will(returnValue(new byte[0]));
107:                 oneOf(memoryCard).write(with(any(byte[].class)));
108:                 exactly(1).of(sensor).powerDown();
109:             }
110:         });
111:     camera.powerOn();
112:     camera.pressShutter();
113:     camera.powerOff();
114:     camera.writeComplete();
115:     assertFalse(camera.get_isWritingData());
116: }
117: }

```

memory card should be written with exactly what was read

Final Tests

Camera.java: 1/2

Shihan Fu - sf23:v5

```

1: package ic.doc.camera;
2:
3: public class Camera implements WriteListener {
4:
5:     private final Sensor sensor;
6:     private final MemoryCard memoryCard;
7:     private boolean power = false;
8:     private boolean isWritingData = false;
9:
10:    /**
11:     * return the status of isWritingData.
12:     *
13:     * @return boolean
14:     */
15:    public boolean get_isWritingData() {
16:        return this.isWritingData;
17:    }
18:
19:    /**
20:     * constructor.
21:     *
22:     * @param sensor Sensor
23:     * @param memoryCard MemoryCard
24:     */
25:    public Camera(Sensor sensor, MemoryCard memoryCard) {
26:        this.memoryCard = memoryCard;
27:        this.sensor = sensor;
28:    }
29:
30:    /** press shutter. */
31:    public void pressShutter() {
32:        if (power) {
33:            isWritingData = true;
34:            memoryCard.write(sensor.readData());
35:        }
36:    }
37:
38:    /** camera power on. */
39:    public void powerOn() {
40:        sensor.powerUp();
41:        this.power = true;
42:    }
43:
44:    /** camera power off. */
45:    public void powerOff() {
46:
47:        if (!isWritingData) {
48:            sensor.powerDown();
49:        }
50:        this.power = false;
51:    }
52:
53:    /** write complete. */
54:    @Override
55:    public void writeComplete() {
56:        isWritingData = false;
57:        if (!power) {
58:            sensor.powerDown();
59:        }
60:    }
61:
62:    /**
63:     * return the status of power.
64:     *
65:     * @return boolean
66:     */

```

Final Tests

Camera.java: 2/2

Shihan Fu - sf23:v5

```

67:     public boolean get_power() {
68:         return this.power;
69:     }
70: }

```

Final Tests

testResults.txt: 1/1

Shihan Fu - sf23:v5

```
1: ----- Test Output -----
2: Running LabTS build... (Tue 6 Feb 16:32:01 UTC 2024)
3:
4: Submission summary...
5: You made 2 commits
6:   - 30a909d implement the basic function and test file [2 files changed, 146 insertions, 7 deletions]
7:   - 6977908 implement the testing logic [1 file changed, 19 insertions, 6 deletions]
8:
9: Preparing...
10:
11: BUILD SUCCESSFUL in 961ms
12:
13: Compiling...
14: BUILD SUCCESSFUL in 11s
15:
16: Running tests...
17:
18: ic.doc.camera.CameraTest > switchingTheCameraOffWithoutPowersDownTheSensor PASSED
19:
20: ic.doc.camera.CameraTest > completeWritingPowerDown PASSED
21:
22: ic.doc.camera.CameraTest > pressShutterWhenCameraPowerOff PASSED
23:
24: ic.doc.camera.CameraTest > switchingTheCameraOffPowersDownTheSensor PASSED
25:
26: ic.doc.camera.CameraTest > pressingShutterWithPowerOnCopiesData PASSED
27:
28: ic.doc.camera.CameraTest > switchingTheCameraOnPowersUpTheSensor PASSED
29:
30: BUILD SUCCESSFUL in 2s
31:
32: Checking test coverage and code style...
33: BUILD SUCCESSFUL in 7s
34: Finished auto test. (Tue 6 Feb 16:32:36 UTC 2024)
35:
36: ----- Test Errors -----
37:
```