

Test Driven Development

More JUnit Tests for the DVD app

Produced Mairead Meagher
by: Dr. Siobhán Drohan



Waterford Institute *of* Technology
INSTITIÚID TEICNEOLAÍOCHTA PHORT LÁIRGE

Department of Computing and Mathematics
<http://www.wit.ie/>

Topic List

- DVD and DVDTest.java
- JUnit Testing of Library.java (which includes testing of XML reading/writing)
- Testing Driver.java

```
public class DVD
{
    private String title;

    public DVD(String title){
        setTitle(title);
    }

    public void setTitle(String title) {
        if (title.length() <= 20){
            this.title = title;
        }
        else{
            this.title = title.substring(0,20);
        }
    }

    public String getTitle() {
        return title;
    }

    public String toString() {
        return "DVD Title is: " + title;
    }
}
```

DVD.java

DVDTest.java

DVDTest.java

```
1 import static org.junit.Assert.*;
2 import org.junit.After;
3 import org.junit.Before;
4 import org.junit.Test;
5
6 public class DVDTest {
7
8     private DVD dvd1, dvd2, dvd3;
9
11    public void setUp() {}
16
18    public void tearDown() {}
20
22    public void testConstructors() {}
27
29    public void testGetTitle() {}
34
36    public void testSetTitle() {}
46
48    public void testToString() {}
53
54 }
```

DVDTest.java

DVDTest.java

```
1 import static org.junit.Assert.*;
2 import org.junit.After;
3 import org.junit.Before;
4 import org.junit.Test;
5
6 public class DVDTest {
7
8     private DVD dvd1, dvd2, dvd3;
9
10    @Before
11    public void setUp() {
12        dvd1 = new DVD("The Hobbit(Director)"); //title with 20 characters
13        dvd2 = new DVD("The Steve Jobs Film"); //title with 19 characters
14        dvd3 = new DVD("Avatar: Directors Cut"); //title with 21 characters
15    }
16
17    @After
18    public void tearDown() {
19    }
20
22    public void testConstructors() {
27
29    public void testGetTitle() {
34
36    public void testSetTitle() {
46
48    public void testToString() {
53
54 }
```

DVDTest.java

DVDTest.java

```
1 import static org.junit.Assert.*;
2 import org.junit.After;
3 import org.junit.Before;
4 import org.junit.Test;
5
6 public class DVDTest {
7
8     private DVD dvd1, dvd2, dvd3;
9
10    @Before
11    public void setUp() {
12        dvd1 = new DVD("The Hobbit(Director)"); //title with 20 characters
13        dvd2 = new DVD("The Steve Jobs Film"); //title with 19 characters
14        dvd3 = new DVD("Avatar: Directors Cut"); //title with 21 characters
15    }
16
18    public void tearDown() {}
19
21    @Test
22    public void testConstructors() {
23        assertEquals("The Hobbit(Director)", dvd1.getTitle());
24        assertEquals("The Steve Jobs Film", dvd2.getTitle());
25        assertEquals("Avatar: Directors Cu", dvd3.getTitle());
26    }
27
29    public void testGetTitle() {}
30
34
36    public void testSetTitle() {}
37
46
48    public void testToString() {}
49
53
54 }
```

```
1 import static org.junit.Assert.*;
2 import org.junit.After;
3 import org.junit.Before;
4 import org.junit.Test;
5
6 public class DVDTest {
7
8     private DVD dvd1, dvd2, dvd3;
9
10    @Before
11    public void setUp() {
12        dvd1 = new DVD("The Hobbit(Director)"); //title with 20 characters
13        dvd2 = new DVD("The Steve Jobs Film"); //title with 19 characters
14        dvd3 = new DVD("Avatar: Directors Cut"); //title with 21 characters
15    }
16
17
18    public void tearDown() {}
19
20
21
22    public void testConstructors() {}
23
24
25
26
27
28    @Test
29    public void testGetTitle() {
30        assertEquals("The Hobbit(Director)", dvd1.getTitle());
31        assertEquals("The Steve Jobs Film", dvd2.getTitle());
32        assertEquals("Avatar: Directors Cu", dvd3.getTitle());
33    }
34
35
36    public void testSetTitle() {
37
38
39
40
41
42
43
44
45
46
47
48    public void testToString() {}
49
50
51
52
53
54 }
```

DVDTest.java

DVDTest.java

```
1 *import static org.junit.Assert.*;
5
6 public class DVDTest {
7
8     private DVD dvd1, dvd2, dvd3;
9
10    @Before
11    public void setUp() {
12        dvd1 = new DVD("The Hobbit(Director)"); //title with 20 characters
13        dvd2 = new DVD("The Steve Jobs Film"); //title with 19 characters
14        dvd3 = new DVD("Avatar: Directors Cut"); //title with 21 characters
15    }
16
18    public void tearDown() {}
20
22    public void testConstructors() {}
27
29    public void testGetTitle() {}
34
35    @Test
36    public void testSetTitle() {
37        dvd1.setTitle("The Hobbit");
38        assertEquals("The Hobbit", dvd1.getTitle());
39        dvd1.setTitle("The Hobbit (Director)"); //attempting to set title to 21 characters
40        assertEquals("The Hobbit (Director", dvd1.getTitle());
41        dvd1.setTitle("The Hobbit(Director)"); //attempting to set title to 20 characters
42        assertEquals("The Hobbit(Director)", dvd1.getTitle());
43        dvd1.setTitle("The Hobbit:Director"); //attempting to set title to 20 characters
44        assertEquals("The Hobbit:Director", dvd1.getTitle());
45    }
46
48    public void testToString() {}
53
54 }
```


DVDTest.java

```
*DVDTest.java
1 import static org.junit.Assert.*;
5
6 public class DVDTest {
7
8     private DVD dvd1, dvd2, dvd3;
9
10    @Before
11    public void setUp() {
12        dvd1 = new DVD("The Hobbit(Director)"); //title with 20 characters
13        dvd2 = new DVD("The Steve Jobs Film"); //title with 19 characters
14        dvd3 = new DVD("Avatar: Directors Cut"); //title with 21 characters
15    }
16
18    public void tearDown() {}
20
22    public void testConstructors() {}
27
29    public void testGetTitle() {}
34
36    public void testSetTitle() {}
46
47    @Test
48    public void testToString() {
49        assertEquals("DVD Title is: The Hobbit(Director)", dvd1.toString());
50        assertEquals("DVD Title is: The Steve Jobs Film", dvd2.toString());
51        assertEquals("DVD Title is: Avatar: Directors Cu", dvd3.toString());
52    }
53
54 }
```

Topic List

- DVD and DVDTest.java
- JUnit Testing of Library.java (which includes testing of XML reading/writing)
- Testing Driver.java

Create a new JUnit Test Case within the “test” folder

Call the test class, **LibraryTest**

Generate the default `setUp()` and `tearDown()` methods.

New JUnit 3 test New JUnit 4 test

Source folder: DVDLibrary3.1/test

Package: (default)

Name: LibraryTest

Superclass: java.lang.Object

Which method stubs would you like to create?

setUpBeforeClass() tearDownAfterClass()
 setUp() tearDown()
 constructor

Do you want to add comments? (Configure templates and default value [here](#))
 Generate comments

Class under test:

Generated LibraryTest.java

```
DVDTTest.java  Library.java  LibraryTest.java ✕
1+ import static org.junit.Assert.*;
6
7 public class LibraryTest {
8
9-     @Before
10    public void setUp() throws Exception {
11    }
12
13-    @After
14    public void tearDown() throws Exception {
15    }
16
17-    @Test
18    public void test() {
19        fail("Not yet implemented");
20    }
21
22 }
23
```

Library.java

- Library.java

- Library

- dvds

- Library()

- add(DVD) : void

- getDVDs() : ArrayList<DVD>

- isValidIndex(int) : boolean

- listDVDs() : String

- load() : void

- save() : void

- setDVDs(ArrayList<DVD>) : void

We need to write at least one test for each of these methods.

Library.java – testing the constructor

- Library.java
 - Library
 - dvds
 - Library()
 - add(DVD) : void
 - getDVDs() : ArrayList<DVD>
 - isValidIndex(int) : boolean
 - listDVDs() : String
 - load() : void
 - save() : void
 - setDVDs(ArrayList<DVD>) : void

To test the constructor, we will create a Library object. Then we will call the getDVDs() method and ensure that the returned ArrayList has a size of 0.

```
private ArrayList<DVD> dvds;  
  
public Library(){  
    dvds = new ArrayList<DVD>();  
}
```

Library.java – testing the constructor

The screenshot displays an IDE interface with two main panels. The left panel shows the Package Explorer and a JUnit test runner window. The test runner window indicates that the test 'testConstructor' in 'LibraryTest' passed successfully, taking 0.000 seconds. The right panel shows the source code for 'LibraryTest.java' (labeled as DVDTTest.java in the tab), which includes imports for JUnit, a class definition for 'LibraryTest', and test methods for setup, teardown, and the test itself.

```
1 import static org.junit.Assert.*;
2 import org.junit.After;
3 import org.junit.Before;
4 import org.junit.Test;
5
6 public class LibraryTest {
7
8     private Library library;
9
10    @Before
11    public void setUp() throws Exception {
12        library = new Library();
13    }
14
15    @After
16    public void tearDown() throws Exception {
17    }
18
19    @Test
20    public void testConstructor() {
21        assertEquals(0, library.getDVDs().size());
22    }
23
24 }
25
```

Package Explorer JUnit

Finished after 0.093 seconds

Runs: 1/1 Errors: 0 Failures: 0

LibraryTest [Runner: JUnit 4] (0.000 s)

testConstructor (0.000 s)

Failure Trace

Library.java – testing add(DVD)

- Library.java
 - Library
 - dvds
 - Library()
 - add(DVD) : void
 - getDVDs() : ArrayList<DVD>
 - isValidIndex(int) : boolean
 - listDVDs() : String
 - load() : void
 - save() : void
 - setDVDs(ArrayList<DVD>)

```
public void add(DVD dvd){  
    dvds.add(dvd);  
}
```



```
9 public class LibraryTest {
10
11     private Library library;
12     private DVD dvd1;
13
14     @Before
15     public void setUp() throws Exception {
16         library = new Library();
17         dvd1 = new DVD("The Hobbit(Director)"); //title with 20 characters
18     }
19
21     public void tearDown() throws Exception {}
22
23
25     public void testConstructor() {}
26
27
29     @Test
30     public void testAdd()
31     {
32         //Testing the ArrayList is Empty
33         assertEquals(0, library.getDVDs().size());
34
35         //Testing the adding of the first dvd
36         library.add(dvd1);
37         assertEquals(1, library.getDVDs().size());
38         assertEquals("The Hobbit(Director)", library.getDVDs().get(0).getTitle());
39
40         //Testing the adding of the second dvd
41         library.add(new DVD("Peppa Pig"));
42         assertEquals(2, library.getDVDs().size());
43         assertEquals("Peppa Pig", library.getDVDs().get(1).getTitle());
44     }
```

Library.java – testing getDVDs and setDVDs

Library.java

Library

dvds

Library()

add(DVD) : void

getDVDs() : ArrayList<DVD>

isValidIndex(int) : boolean

listDVDs() : String

load() : void

save() : void

setDVDs(ArrayList<DVD>) : void

```
public ArrayList<DVD> getDVDs(){  
    return dvds;  
}
```

```
public void setDVDs(ArrayList<DVD> dvds){  
    this.dvds = dvds;  
}
```

```
public class LibraryTest {

    private Library library, populatedLibrary;
    private DVD dvd1, dvd2, dvd3;
    private ArrayList<DVD> emptyDVDs, populatedDVDs;

    @Before
    public void setUp() throws Exception {

        library = new Library();
        emptyDVDs = new ArrayList<DVD>();

        populatedDVDs = new ArrayList<DVD>();
        dvd1 = new DVD("The Hobbit(Director)"); //title with 20 characters
        dvd2 = new DVD("The Steve Jobs Film"); //title with 19 characters
        dvd3 = new DVD("Avatar: Directors Cut"); //title with 21 characters
        populatedDVDs.add(dvd1);
        populatedDVDs.add(dvd2);
        populatedDVDs.add(dvd3);
        populatedLibrary = new Library();
        populatedLibrary.setDVDs(populatedDVDs);
    }
}
```

```
@Test
public void testGetDVDs()
{
    //The new library object returns an empty ArrayList of DVD
    assertEquals(0, library.getDVDs().size());
    assertEquals(emptyDVDs, library.getDVDs());

    //The populated library object returns an ArrayList with three items
    assertEquals(3, populatedLibrary.getDVDs().size());
    assertEquals(populatedDVDs, populatedLibrary.getDVDs());
}
```

Library.java – testing listDVDs()

Library.java

Library

• dvds

• Library()

• add(DVD) : void

• getDVDs() : ArrayList<DVD>

• isValidIndex(int) : boolean

• listDVDs() : String

• load() : void

• save() : void

• setDVDs(ArrayList<DVD>) : void

```
public String listDVDs(){
    if (dvds.size() == 0){
        return "No DVDs.";
    }
    else{
        String listDVDs = "";
        for (int i = 0; i < dvds.size(); i++){
            listDVDs = listDVDs + (i + ":" + dvds.get(i)) + "\n";
        }
        return listDVDs;
    }
}
```

```
public class LibraryTest {

    private Library library, populatedLibrary;
    private DVD dvd1, dvd2, dvd3;
    private ArrayList<DVD> emptyDVDs, populatedDVDs;

    @Before
    public void setUp() throws Exception {

        library = new Library();
        emptyDVDs = new ArrayList<DVD>();

        populatedDVDs = new ArrayList<DVD>();
        dvd1 = new DVD("The Hobbit(Director)"); //title with 20 characters
        dvd2 = new DVD("The Steve Jobs Film"); //title with 19 characters
        dvd3 = new DVD("Avatar: Directors Cut"); //title with 21 characters
        populatedDVDs.add(dvd1);
        populatedDVDs.add(dvd2);
        populatedDVDs.add(dvd3);
        populatedLibrary = new Library();
        populatedLibrary.setDVDs(populatedDVDs);
    }
}
```

```
@Test
public void testListDVDs()
{
    //The new library object returns an empty String
    assertEquals("No DVDs.", library.listDVDs());

    //The populated library object returns an String listing three items
    assertEquals("0:DVD Title is: The Hobbit(Director)\n"
        + "1:DVD Title is: The Steve Jobs Film\n"
        + "2:DVD Title is: Avatar: Directors Cu\n",
        populatedLibrary.listDVDs());
}
```

Library.java – testing save and load

```
@SuppressWarnings("unchecked")
public void load() throws Exception
{
    XStream xstream = new XStream(new DomDriver());
    ObjectInputStream is = xstream.createObjectInputStream
        (new FileReader("dvds.xml"));
    dvds = (ArrayList<DVD>) is.readObject();
    is.close();
}
```

```
public void save() throws Exception
{
    XStream xstream = new XStream(new DomDriver());
    ObjectOutputStream out = xstream.createObjectOutputStream
        (new FileWriter("dvds.xml"));
    out.writeObject(dvds);
    out.close();
}
```

Library.java

Library

- dvds

- Library()

- add(DVD) : void

- getDVDs() : ArrayList<DVD>

- isValidIndex(int) : boolean

- listDVDs() : String

- load() : void

- save() : void

- setDVDs(ArrayList<DVD>) : void

Library.java – testing save and load

```
public class LibraryTest {

    private Library library, populatedLibrary;
    private DVD dvd1, dvd2, dvd3;
    private ArrayList<DVD> emptyDVDs, populatedDVDs;

    @Before
    public void setUp() throws Exception {

        library = new Library();
        emptyDVDs = new ArrayList<DVD>();

        populatedDVDs = new ArrayList<DVD>();
        dvd1 = new DVD("The Hobbit(Director)"); //title with 20 characters
        dvd2 = new DVD("The Steve Jobs Film"); //title with 19 characters
        dvd3 = new DVD("Avatar: Directors Cut"); //title with 21 characters
        populatedDVDs.add(dvd1);
        populatedDVDs.add(dvd2);
        populatedDVDs.add(dvd3);
        populatedLibrary = new Library();
        populatedLibrary.setDVDs(populatedDVDs);
    }
}
```

Library.java – testing save and load

```
@Test
public void testSaveAndLoad() throws Exception
{
    //TESTING AN EMPTY ARRAYLIST
    //-----
    //Saving a new library object with an empty ArrayList of DVD
    assertEquals(0, library.getDVDs().size());
    assertEquals(emptyDVDs, library.getDVDs());
    library.save();
    //Load the file into another library object and compare it to emptyDVDs
    Library library2 = new Library();
    library2.load();
    assertEquals(library2.getDVDs().size(), library.getDVDs().size());

    //TESTING A POPULATED ARRAYLIST
    //-----
    //Saving a library object with a populated ArrayList of DVD
    assertEquals(3, populatedLibrary.getDVDs().size());
    assertEquals(populatedDVDs, populatedLibrary.getDVDs());
    populatedLibrary.save();
    //Load the file into another library object and compare it to populatedLibrary
    Library library3 = new Library();
    library3.load();
    assertEquals(library3.getDVDs().size(), populatedLibrary.getDVDs().size());
    assertEquals(library3.getDVDs().get(1).getTitle(), populatedLibrary.getDVDs().get(1).getTitle());
    assertEquals(library3.getDVDs().get(2).getTitle(), populatedLibrary.getDVDs().get(2).getTitle());
}
```


Library.java – testing isValidIndex(int)

- Library.java
 - Library
 - dvds
 - Library()
 - add(DVD) : void
 - getDVDs() : ArrayList<DVD>
 - isValidIndex(int) : boolean
 - listDVDs() : String
 - load() : void
 - save() : void
 - setDVDs(ArrayList<DVD>) : void

```
public boolean isValidIndex(int index)
{
    return ( (index >= 0) && (index < dvds.size()) );
}
```

Library.java – testing isValidIndex(int)

```
@Test
public void testIsValidIndex() {
    //Boundary testing the library object, which has an empty ArrayList of DVDs.
    assertEquals(0, library.getDVDs().size());
    assertEquals(false, library.isValidIndex(-1));
    assertEquals(false, library.isValidIndex(0));
    assertEquals(false, library.isValidIndex(1));

    //Boundary testing the library object which has an ArrayList of three DVDs.
    //lower boundary tests / fence post error testing
    assertEquals(3, populatedLibrary.getDVDs().size());
    assertEquals(false, populatedLibrary.isValidIndex(-1));
    assertEquals(true, populatedLibrary.isValidIndex(0));
    assertEquals(true, populatedLibrary.isValidIndex(1));
    //upper boundary tests / fence post error testing
    assertEquals(true, populatedLibrary.isValidIndex(2));
    assertEquals(false, populatedLibrary.isValidIndex(3));
}
```

```
public boolean isValidIndex(int index)
{
    return ( (index >= 0) && (index < dvds.size()) );
}
```

Topic List

- DVD and DVDTest.java
- JUnit Testing of Library.java (which includes testing of XML reading/writing)
- Testing Driver.java

Driver.java

- JUnit is not used to test the class that takes input from the console.
- Why do you think this is?

**Any
Questions?**





Except where otherwise noted, this content is licensed under a Creative Commons Attribution-NonCommercial 3.0 License.

For more information, please see <http://creativecommons.org/licenses/by-nc/3.0/>



Waterford Institute of Technology
INSTITIÚID TEICNEOLAÍOCHTA PHORT LÁIRGE

Department of Computing and Mathematics
<http://www.wit.ie/>