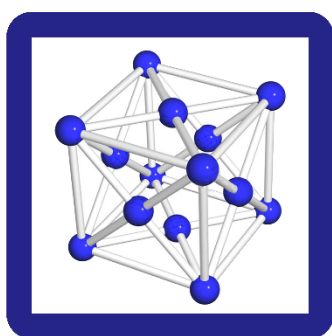




UNIVERSITÀ DEGLI STUDI DI SALERNO



UNIVERSITÀ DEGLI STUDI DI SALERNO
DIPARTIMENTO DI INFORMATICA



earth minerals
EMC
EUROPEAN METALS CORPORATION

CORSO DI INGEGNERIA DEL SOFTWARE PROF. A. DE LUCIA

PROGETTO EMC

UNIT TEST

2020/2021

PARTECIPANTI	MATRICOLA
ALESSANDRA POTESA'	06188
ROSARIO ANNUNZIATA	05810
GIOVANNI TAVOLO	05912

Introduzione

Per testing di unità si intende il testing delle singole unità software del sistema. Il test verrà effettuato con il framework JUnit insieme alla libreria Mockito.

Relazione con altri documenti

Per Individuare i test da effettuare si utilizzerà la tecnica Black-Box, quindi ci baseremo sui documenti prodotti precedentemente:

- TestPlanEMC
- TestCaseEMC

Approccio

Il testing di unità verrà diviso in testingDAO, testingBean e testingServlet, dove verranno testati singolarmente i dao, i bean e le servlet del sistema.

1 - Testing di unità

1.0 - TestDAO

1.0.1 – Test Categoria

The screenshot displays an IDE with the following components:

- Project Explorer:** Shows a project structure with various servlets and a 'test' directory containing several test classes, including 'CategoriaDAOTest'.
- Editor:** Displays the 'CategoriaDAO.java' file with the following code:

```
19 public void doRetrieveAll() {  
20  
21     List<Categoria> categoria = new ArrayList<>();  
22     assertNotNull(categoria, categoriaDAO.doRetrieveAll());  
23  
24 }  
25  
26  
27  
28 @Test  
29 public void doSave() {  
30  
31     Categoria categoriaValida = new Categoria(id: 10, nome: "Categorial", descrizione: "Descrizione1");  
32     Categoria categoriaNonValida = new Categoria(id: 11, nome: "Categorial", descrizione: "Descrizione1");  
33  
34     int succ = categoriaDAO.doSave(categoriaValida);  
35     assertEquals( expected: 1, succ);  
36  
37 }  
38  
39  
40 @Test  
41 public void doUpdate() {  
42  
43     Categoria categoriaUpdate = new Categoria(id: 4, nome: "CategoriaNuova", descrizione: "DescrizioneNuova");  
44     boolean succ = categoriaDAO.doUpdate(categoriaUpdate);  
45  
46     assertEquals( expected: true, succ);  
47  
48 }
```
- Run Console:** Shows the execution of 'CategoriaDAOTest (tw.test)' with the following results:

Test Method	Duration
doSave	950 ms
doDelete	4 ms
doUpdate	7 ms
doRetrieveByDescrizione	1 ms
doRetrieveByNome	1 ms
doRetrieveAll	2 ms

Tests passed: 6 of 6 tests - 965 ms
C:\Users\RioKurama\Desktop\jdk-13.0.1\bin\java.exe ...
Process finished with exit code 0

1.0.2 – Test Ordine

The screenshot displays an IDE with the following components:

- Project Explorer (Left):** A tree view of the project structure. The 'test' package is expanded, showing various test classes. 'OrdineDAOTest' is selected and highlighted.
- Code Editor (Center):** Displays the source code for 'OrdineDAOTest.java'. The code includes:
 - Package declaration: `package tsw.test;`
 - Imports: `import ...`
 - Class definition: `public class OrdineDAOTest {`
 - Instance creation: `OrdineDAO ordineDAO = new OrdineDAO();`
 - Test method `doSave()`:
 - Annotation: `@Test`
 - Logic: `int pippo = ordineDAO.doSave(indirizzo: "indirizzo", idu: 2, prezzo: "prezzo");`
 - Assertion: `assertEquals(expected: 1, pippo);`
 - Test method `doRetrieveByUtente()`:
 - Annotation: `@Test`
 - Logic: `List<Ordine> ordineList = new ArrayList<>();`
 - Assertion: `assertNotEquals(ordineList, ordineDAO.doRetrieveByUtente(2));`
 - Class closure: `}`
- Run Console (Bottom):** Shows the execution results of the tests.
 - Summary: `Tests passed: 2 of 2 tests - 951 ms`
 - Test Results Table:

Test Name	Duration
OrdineDAOTest (tsw.test)	951 ms
doSave	946 ms
doRetrieveByUtente	5 ms
 - Exit Code: `Process finished with exit code 0`

1.0.3 – Test Prodotto

The screenshot displays an IDE with a project explorer on the left, a code editor in the center, and a test runner window at the bottom.

Project Explorer: The 'test' package is expanded, showing various test classes. 'ProdottoDAOTest' is selected.

Code Editor: The file 'ProdottoDAOTest.java' is open. It contains three test methods:

```
30
31
32 @Test
33 public void doRetrieveByCategoria() {
34     List<Prodotto> prodottoList = new ArrayList<>();
35     assertNotNull(prodottoList, prodottoDAO.doRetrieveByCategoria(1));
36 }
37
38
39 @Test
40 public void doSave() {
41
42     Prodotto prodotto= new Prodotto( id: 11, nome: "nome", descrizione: "descrizione", prezzoBase: 3, iva: 2, idcategoria: 2);
43     int pippo = prodottoDAO.doSave(prodotto);
44     assertEquals( expected: 1, pippo);
45 }
46
47
48 @Test
49 public void doUpdate() {
50
51     Prodotto prodotto= new Prodotto( id: 3, nome: "nome", descrizione: "descrizione", prezzoBase: 3, iva: 2, idcategoria: 2);
52     boolean succ = prodottoDAO.doUpdate(prodotto);
53     assertEquals( expected: true, succ);
54 }
55
56
```

Test Runner: The 'Run' window shows the execution of 'ProdottoDAOTest (tsw.test)'. It lists 12 tests, all of which passed. The total execution time was 977 ms.

Test Method	Duration
ProdottoDAOTest (tsw.test)	977 ms
doRetrieveByVia	937 ms
doSave	10 ms
doRetrieveById	4 ms
doDelete	6 ms
doUpdate	4 ms
doRetrieveByNomeOrDescrizione	3 ms
doRetrieveByDescrizione	2 ms
doRetrieveByNome	1 ms
doRetrieveByPrezzo	2 ms
doRetrieveByCategoria	2 ms
doRetrieveAll	4 ms
doRetrieveByNomeSingolo	2 ms

Process finished with exit code 0

1.0.4 – Test Utente

The screenshot shows an IDE with the following components:

- Project Explorer:** Lists various servlets and tests. The `test` folder is expanded, showing `UtenteDAOTest` selected.
- Code Editor:** Displays the `UtenteDAOTest.java` file with the following code:

```
64 public void doSave() {  
65  
66     Utente utente = new Utente( id: 5, username: "Prova", passwordhash: "Prova", nome: "Email", email: "Prova", admin: false);  
67  
68     boolean succ = utenteDAO.doSave(utente);  
69     assertEquals( expected: true, succ);  
70  
71 }  
72  
73  
74 @Test  
75 public void doUpdate() {  
76  
77     int succ = utenteDAO.doUpdate( username: "Prova1", nome: "Email1", email: "Prova1", id: 5);  
78     assertEquals( expected: 1, succ);  
79  
80  
81 }  
82  
83  
84 @Test  
85 public void doDelete() {  
86  
87     int succ = utenteDAO.doDelete( id: 1);  
88     assertEquals( expected: 1, succ);  
89  
90 }  
91 }
```
- Run Console:** Shows the execution of `UtenteDAOTest` with the following output:

```
Run: UtenteDAOTest  
Tests passed: 8 of 8 tests - 1 sec 147 ms  
C:\Users\RioKurama\Desktop\jdk-13.0.1\bin\java.exe ...  
Process finished with exit code 0
```

1.1 - TestBean

1.1.0 – Test Carrello

The screenshot displays an IDE with the `CarrelloTest.java` file open. The left sidebar shows a project tree with a 'test' folder containing various test classes, including `CarrelloTest`. The main editor shows the following code:

```
1 package tsw.test;
2
3 import ...
4
5
6
7
8 public class CarrelloTest {
9
10
11     @Test
12     void testCarrelloBean() {
13
14         int quantita = 2;
15         Prodotto prodotto = new Prodotto( id: 10, nome: "Prova", descrizione: "Desc", prezzoBase: 20, iva: 22, idcategoria: 1);
16
17         Carrello.ProdottoQuantita prodottoQuantita = new Carrello.ProdottoQuantita(prodotto, quantita);
18         String test = prodottoQuantita.getProdotto().toString();
19
20
21         assertEquals( expected: 10, prodottoQuantita.getProdotto().getId());
22         assertEquals( expected: "Prova", prodottoQuantita.getProdotto().getNome());
23         assertEquals( expected: "Desc", prodottoQuantita.getProdotto().getDescrizione());
24         assertEquals( expected: 20, prodottoQuantita.getProdotto().getPrezzoBase());
25         assertEquals( expected: 22, prodottoQuantita.getProdotto().getIva());
26         assertEquals( expected: 1, prodottoQuantita.getProdotto().getIdcategoria());
27         assertEquals(test, prodottoQuantita.getProdotto().toString());
28     }
29 }
30
31 }
```

The bottom panel shows the 'Run' output for `CarrelloTest`. It indicates that 1 of 1 tests passed in 82 ms. The test results table shows:

Test Results	Time	Location
CarrelloTest	82 ms	C:\Users\RioKurama\Desktop\jdk-13.0.1\bin\java.exe ...
testCarrelloBean()	82 ms	

The output also states: 'Process finished with exit code 0'.

1.1.1 – Test Categoria

The screenshot displays an IDE with the following components:

- Project Explorer:** A list of project files on the left. The `test` folder is expanded, showing various test classes. `CategoriaTest` is selected and highlighted.
- Editor:** The main window shows the source code of `CategoriaTest.java`. The code defines a `testCategoriaBean()` method that:
 - Imports `assertEquals` from `org.junit`.
 - Creates a `Categoria` object with `id = 1`, `nome = "Gina"`, and `descrizione = "La gallina"`.
 - Creates a second `Categoria` object, `categoria1`, with the same values.
 - Converts both objects to strings (`test` and `test1`).
 - Uses `assertEquals` to verify that the objects and their string representations are equal.
- Run Console:** At the bottom, it shows the execution results:
 - Summary: `Tests passed: 1 of 1 test - 36 ms`
 - Test Results Table:

Test Name	Duration
Test Results	36 ms
CategoryTest	36 ms
testCategoriaBean()	36 ms
 - Output: `C:\Users\RioKurama\Desktop\jdk-13.0.1\bin\java.exe ...` and `Process finished with exit code 0`.

1.1.2 – Test Login

The screenshot displays an IDE interface with the `LoginTest.java` file open. The left sidebar shows a project structure with various servlets and tests. The main editor shows the code for `LoginTest`, which includes a `@Test` method `testLogin()`. The code creates a `Login` object, sets its attributes, and performs several assertions to verify its state.

```
7  
8 public class LoginTest {  
9  
10  
11     @Test  
12     void testLogin() {  
13  
14         String id = "pippo";  
15         int idutente = 1;  
16         String token = "token";  
17  
18         Login login = new Login(id, idutente, token);  
19         Login login1 = new Login(id, idutente, token);  
20         assertNotNull(login);  
21         String test = login.toString();  
22         String test1 = login1.toString();  
23  
24         assertEquals(id, login.getId());  
25         assertEquals(idutente, login.getIdutente());  
26         assertEquals(token, login.getToken());  
27         assertEquals(test, login.toString());  
28  
29         login1.setId(id);  
30         login1.setIdutente(idutente);  
31         login1.setToken(token);  
32         assertEquals(id, login1.getId());  
33         assertEquals(idutente, login1.getIdutente());  
34         assertEquals(token, login1.getToken());  
35         assertEquals(test1, login.toString());  
36  
37     }  
}
```

The bottom panel shows the execution results for `LoginTest`. The test `testLogin()` passed successfully in 46 ms. The command prompt shows the process finished with exit code 0.

Run: LoginTest

Tests passed: 1 of 1 test – 46 ms

Test Results	Time
Test Results	46 ms
LoginTest	46 ms
testLogin()	46 ms

Process finished with exit code 0

1.1.3 – Test Ordine

The screenshot displays an IDE with the following components:

- Project Explorer:** A list of Java classes and test files. The 'test' folder is expanded, showing various test classes. 'OrdineTest' is selected.
- Editor:** The 'OrdineTest.java' file is open, showing the following code:

```
1 package tsw.test;
2
3 import ...
10
11 public class OrdineTest {
12
13     @Test
14     void testOrdine() {
15
16         int id = 1;
17         String indirizzo = "Indirizzo";
18         String prezzo = "token";
19         Utente utente = new Utente(id, "Pippo", "pass", "Nome", "Email", true);
20         Ordine ordine = new Ordine(id, indirizzo, prezzo, utente);
21
22         assertNotNull(ordine);
23         String test = ordine.toString();
24
25         assertEquals(id, ordine.getId());
26         assertEquals(indirizzo, ordine.getIndirizzo());
27         assertEquals(prezzo, ordine.getPrezzotot());
28         assertEquals(utente, ordine.getUtente());
29         assertEquals(test, ordine.toString());
30
31     }
32
33 }
34
35
36 }
```
- Run Console:** Shows the execution of the test. The output is: "Tests passed: 1 of 1 test - 63 ms". Below this, a table shows the results for the 'OrdineTest' class and its 'testOrdine()' method, both of which passed in 63 ms.
- Test Results:** A summary of the test results, showing "Tests passed: 1".

1.1.4 – Test Prodotto

The screenshot shows an IDE with a project named 'web' on the left. The 'test' folder is expanded, showing various test classes. The 'ProdottoTest' class is selected, and its code is displayed in the main editor. The code defines private fields for 'prezzoBase', 'iva', 'categorie', and 'Rame', and implements the 'testProdotto()' method. The method creates two 'Prodotto' objects and asserts their properties. The bottom panel shows the test results, indicating that the test passed successfully.

```
18 private long prezzoBase;
19 private int iva;
20 private List<Categoria> categorie;
21 private Categoria Rame = new Categoria();
22
23
24 @Test
25 void testProdotto() {
26
27     int id = 1;
28     String nome = "Indirizzo";
29     String descrizione = "token";
30     long prezzoBase = 2000;
31     int iva = 22;
32     int idcategoria = 1;
33
34     Prodotto prodotto = new Prodotto(id, nome, descrizione, prezzoBase, iva, idcategoria);
35     Prodotto prodotto1 = new Prodotto(id, nome, descrizione, prezzoBase, iva, idcategoria);
36     String test = prodotto.toString();
37     String test1 = prodotto1.toString();
38
39
40     assertEquals(id, prodotto.getId());
41     assertEquals(nome, prodotto.getNome());
42     assertEquals(descrizione, prodotto.getDescrizione());
43     assertEquals(prezzoBase, prodotto.getPrezzoBase());
44     assertEquals(iva, prodotto.getIva());
45     assertEquals(idcategoria, prodotto.getIdcategoria());
46     assertEquals(test, prodotto.toString());
47
48 }
```

Run: **ProdottoTest** Tests passed: 1 of 1 test - 49 ms

Test Results	49 ms	C:\Users\RioKurama\Desktop\jdk-13.0.1\bin\java.exe ...
ProdottoTest	49 ms	
testProdotto()	49 ms	Process finished with exit code 0

Tests passed: 1

1.1.5 – Test Utente

The screenshot displays an IDE interface with the following components:

- Project Explorer (Left):** A list of project files including servlets (e.g., ChiSiamoServlet, CookieLoginFilter), exceptions (MyServletException), and a 'test' directory containing various test classes. 'UtenteTest' is selected at the bottom.
- Editor (Center):** Shows the code for 'UtenteTest.java'. The code defines a class with a single test method 'testUtente()' that creates a user object, sets its attributes, and performs assertions.
- Run Console (Bottom):** Displays the execution results. It shows 'Tests passed: 1 of 1 test - 63 ms' and 'Process finished with exit code 0'.

```
public class UtenteTest {  
  
    @Test  
    void testUtente() {  
  
        int id = 1;  
        String username = "Indirizzo";  
        String passwordhash = "token";  
        String nome = "Nome";  
        String email = "Email";  
        boolean admin = false;  
  
        Utente utente = new Utente(id, username, passwordhash, nome, email, admin);  
        Utente utente1 = new Utente(id, username, passwordhash, nome, email, admin);  
        String test = utente.toString();  
        String test1 = utente1.toString();  
  
        assertEquals(id, utente.getId());  
        assertEquals(username, utente.getUsername());  
        assertEquals(passwordhash, utente.getPasswordhash());  
        assertEquals(nome, utente.getNome());  
        assertEquals(email, utente.getEmail());  
        assertEquals(admin, utente.isAdmin());  
        assertEquals(test, utente.toString());  
  
        utente1.setId(id);  
        utente1.setUsername(username);  
        utente1.setPasswordhash(passwordhash);  
        utente1.setNome(nome);  
        utente1.setEmail(email);  
    }  
}
```

Run: UtenteTest

Tests passed: 1 of 1 test - 63 ms

C:\Users\RioKurama\Desktop\jdk-13.0.1\bin\java.exe ...

Process finished with exit code 0