

DBUNIT

```
1 public class AlumnoDAOIT {
2     private IDatabaseTester databaseTester;
3     private IDatabaseConnection connection;
4     @BeforeEach
5     public void setUp() throws Exception {
6
7         databaseTester = new MiJdbcDatabaseTester(,cadena_conexionDB, "root", "ppss");
8         //obtenemos la conexión con la BD
9         connection = databaseTester.getConnection();
10    }
11    @Test
12    public void testB1() throws Exception {
13        //Inicializamos el dataSet con los datos iniciales de la tabla
14        IDataset dataSet = new FlatXmlDataFileLoader().load("/tabla2.xml");
15        //Inyectamos el dataset en el objeto databaseTester
16        databaseTester.setDataSet(dataSet);
17        //inicializamos la base de datos con los contenidos del dataset
18        databaseTester.onSetup();
19        //invocamos a nuestro SUT
20        Assertions.assertDoesNotThrow(()->new FactoriaDAO().getAlumnoDAO().delAlumno("11111111A"));
21
22        //recuperamos los datos de la BD después de invocar al SUT
23        IDataset databaseDataSet = connection.createDataSet();
24        //Recuperamos los datos de la tabla alumnos
25        ITable actualTable = databaseDataSet.getTable("alumnos");
26
27        //creamos el dataset con el resultado esperado
28        IDataset expectedDataSet = new FlatXmlDataFileLoader().load("/tabla4.xml");
29        ITable expectedTable = expectedDataSet.getTable("alumnos");
30
31        Assertion.assertEquals(expectedTable, actualTable);
32    }
33    @Test
34    public void testRetrieve() throws Exception {
35        //Inicializamos el dataSet con los datos iniciales de la tabla cliente
36        IDataset dataSet = new FlatXmlDataFileLoader().load("/cliente-esperado.xml");
37        //Inyectamos el dataset en el objeto databaseTester
38        databaseTester.setDataSet(dataSet);
39        //inicializamos la base de datos con los contenidos del dataset
40        databaseTester.onSetup();
41        //invocamos a nuestro SUT
42        Cliente cl= Assertions.assertDoesNotThrow(()->clienteDAO.retrieve(1));
43        Assertions.assertAll(
44            ()->Assertions.assertEquals(cl.getId(),1),
45            ()->Assertions.assertEquals(cl.getNombre(), "John"),
46            ()->Assertions.assertEquals(cl.getApellido(), "Smith"),
47            ()->Assertions.assertEquals(cl.getDireccion(), "1 Main Street"),
48            ()->Assertions.assertEquals(cl.getCiudad(), "Anycity")
49        );
50        //recuperamos los datos de la BD después de invocar al SUT
51        IDataset databaseDataSet = connection.createDataSet();
52        //Recuperamos los datos de la tabla cliente
53        ITable actualTable = databaseDataSet.getTable("cliente");
54
55        //creamos el dataset con el resultado esperado
56        IDataset expectedDataSet = new FlatXmlDataFileLoader().load("/cliente-esperado.xml");
57        ITable expectedTable = expectedDataSet.getTable("cliente");
58
59        Assertion.assertEquals(expectedTable, actualTable);
60    }
61 }
```

WebDriver



```
1  public class LoginPage {
2      WebDriver driver;
3      @FindBy(name="uid") WebElement userID;
4      @FindBy(name="password") WebElement password;
5      @FindBy(cssSelector="h1 > p") WebElement login;
6      @FindBy(className="barone") WebElement loginTitle;
7
8      public LoginPage(WebDriver driver){
9          this.driver = driver;
10         this.driver.get("http://demo.guru99.com/V4");
11     }
12     //nos logueamos
13     public ManagerPage login(String user, String pass) {
14         userID.sendKeys("user");
15         password.sendKeys("pass");
16         login.click();
17         //retur de la pagina destino
18         return PageFactory.initElements(driver,ManagerPage.class);
19     }
20     //cogemos el titulo
21     public String getTile(){
22         return driver.getTitle();
23     }
24 }
25
26 public class TestLoginPage {
27     WebDriver driver;
28     LoginPage poLogin;
29     ManagerPage poManagerPage;
30
31     @BeforeEach
32     public void setup(){
33         driver = new ChromeDriver();
34         poLogin = PageFactory.initElements(driver, LoginPage.class);
35     }
36     @Test
37     public void test_Login_Correct(){
38         //cogemos el
39         String loginPageTitle = poLogin.getTile();
40         Assertions.assertEquals("guru99 bank",loginPageTitle);
41         //nos loguemos y obtenemos la case MAnage
42         poManagerPage = poLogin.login("mngr34733", "AbEvydU");
43         //obtenemos el nombre de usuario
44         Assertions.assertEquals("manger id : mngr34733",poManagerPage.getUserName());
45     }
46 }
47 @AfterEach
48 public void setup(){
49     driver.close();
50 }
51 }
```

POM Integración DBunit



```
1 <project xmlns="http://maven.apache.org/POM/4.0.0" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
2   xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 http://maven.apache.org/xsd/maven-4.0.0.xsd">
3   <modelVersion>4.0.0</modelVersion>
4   <groupId>ppss</groupId>
5   <artifactId>matriculacion</artifactId>
6   <version>1.0-SNAPSHOT</version>
7   <!--modulos del proyecto-->
8   <modules>
9     <module>matriculacion-comun</module>
10    <module>matriculacion-dao</module>
11    <module>matriculacion-proxy</module>
12    <module>matriculacion-bo</module>
13  </modules>
14  <!-->
15  <packaging>pom</packaging><!--package para prouectos multimodulo-->
16  <name>matriculacion</name>
17  <properties>
18    <project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>
19    <maven.compiler.source>11</maven.compiler.source>
20    <maven.compiler.target>11</maven.compiler.target>
21  </properties>
22  <dependencies>
23    <dependency>
24      <groupId>org.junit.jupiter</groupId>
25      <artifactId>junit-jupiter-engine</artifactId>
26      <version>5.8.2</version>
27      <scope>test</scope>
28    </dependency>
29  </dependencies>
30  <build>
31    <plugins>
32      <plugin>
33        <groupId>org.apache.maven.plugins</groupId>
34        <artifactId>maven-compiler-plugin</artifactId>
35        <version>3.9.0</version>
36      </plugin>
37      <plugin>
38        <groupId>org.apache.maven.plugins</groupId>
39        <artifactId>maven-surefire-plugin</artifactId>
40        <version>3.0.0-M5</version>
41      </plugin>
42      <!--Plugin failsafe para la integracion-->
43      <plugin>
44        <groupId>org.apache.maven.plugins</groupId>
45        <artifactId>maven-failsafe-plugin</artifactId>
46        <version>3.0.0-M5</version>
47        <executions>
48          <execution>
49            <!--Asignamos la goal a la fase integration-test y verify-->
50            <goals>
51              <goal>integration-test</goal>
52              <goal>verify</goal>
53            </goals>
54          </execution>
55        </executions>
56      </plugin>
57    <!-->
58  </plugins>
59 </build>
60 </project>
```

```

1  <?xml version="1.0" encoding="UTF-8"?>
2  <project xmlns="http://maven.apache.org/POM/4.0.0"
3          xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
4          xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 http://maven.apache.org/xsd/maven-4.0.0.xsd">
5      <!--Padre del proyecto-->
6      <parent>
7          <artifactId>matriculacion</artifactId>
8          <groupId>ppss</groupId>
9          <version>1.0-SNAPSHOT</version>
10     </parent>
11     <!-->
12     <modelVersion>4.0.0</modelVersion>
13     <artifactId>matriculacion-dao</artifactId>
14     <dependencies>
15         <!--dependencias del proyecto-->
16         <dependency>
17             <groupId>${project.groupId}</groupId>
18             <artifactId>matriculacion-comun</artifactId>
19             <version>${project.version}</version>
20         </dependency>
21         <!-->
22         <dependency>
23             <groupId>mysql</groupId>
24             <artifactId>mysql-connector-java</artifactId>
25             <version>8.0.27</version>
26             <scope>test</scope>
27         </dependency>
28         <dependency>
29             <groupId>org.slf4j</groupId>
30             <artifactId>slf4j-simple</artifactId>
31             <version>1.7.36</version>
32         </dependency>
33         <dependency>
34             <groupId>org.dbunit</groupId>
35             <artifactId>dbunit</artifactId>
36             <version>2.7.3</version>
37             <scope>test</scope>
38         </dependency>
39     </dependencies>
40     <build>
41         <plugins>
42             <!--plugin SQL-->
43             <plugin>
44                 <groupId>org.codehaus.mojo</groupId>
45                 <artifactId>sql-maven-plugin</artifactId>
46                 <version>1.5</version>
47                 <dependencies>
48                     <dependency>
49                         <groupId>mysql</groupId>
50                         <artifactId>mysql-connector-java</artifactId>
51                         <version>8.0.27</version>
52                     </dependency>
53                 </dependencies>
54                 <configuration><!--Cadena de coneion-->
55                     <driver>com.mysql.cj.jdbc.Driver</driver>
56                     <url>jdbc:mysql://localhost:3306/?useSSL=false</url>
57                     <username>root</username>
58                     <password>ppss</password>
59                 </configuration>
60                 <executions>
61                     <!--añadimos la goal execute a la fase pre-integration-test para ejecutar comando SQL -->
62                     <execution>
63                         <id>create-alumnos-table</id>
64                         <phase>pre-integration-test</phase>
65                         <goals>
66                             <goal>execute</goal>
67                         </goals>
68                         <configuration>
69                             <srcFiles>
70                                 <srcFile>src/test/resources/sql/matriculacion.sql</srcFile>
71                             </srcFiles>
72                         </configuration>
73                     </execution>
74                     <!-->
75                 </executions>
76             </plugin>
77         </plugins>
78         <!--plugin SQL-->
79     </build>
80 </project>

```

Plugin de JACOCO

```
1 <plugin>
2   <groupId>org.jacoco</groupId>
3   <artifactId>jacoco-maven-plugin</artifactId>
4   <version>0.8.7</version>
5   <!--asociamos las goals al plugin-->
6   <executions>
7     <execution>
8       <id>default-prepare-agent</id>
9       <goals>
10        <goal>prepare-agent</goal>
11      </goals>
12    </execution>
13    <execution>
14      <id>default-report</id>
15      <goals>
16        <goal>report</goal>
17      </goals>
18    </execution>
19    <!--Aqui se define las reglas de las cobertura-->
20    <execution>
21      <id>default-check</id>
22      <goals>
23        <goal>check</goal>
24      </goals>
25      <configuration>
26        <rules>
27          <!--BUNDLE PACKAGE CLASS METHOD-->
28          <rule>
29            <!--nivel de app-->
30            <element>BUNDLE</element>
31            <limits>
32              <limit>
33                <!--COMPLEXITY, INSTRUCTION, CLASS, LINE, BRANCH-->
34                <counter>COMPLEXITY</counter>
35                <!--TOTALCOUNT, COVEREDCOUNT, MISSEDCOUNT, COVEREDRATIO, MISSEDRATIO-->
36                <value>COVEREDRATIO</value>
37                <minimum>0.85</minimum>
38              </limit>
39              <limit>
40                <counter>INSTRUCTION</counter>
41                <value>COVEREDRATIO</value>
42                <minimum>0.80</minimum>
43              </limit>
44            </limits>
45          </rule>
46          <!--excluimos una clase-->
47          <rule>
48            <element>CLASS</element>
49            <excludes>
50              <exclude>
51                ppss.ejercicio2.MyClass
52              </exclude>
53            </excludes>
54            <limits>
55              <limit>
56                <counter>LINE</counter>
57                <value>COVEREDRATIO</value>
58                <minimum>0.75</minimum>
59              </limit>
60            </limits>
61          </rule>
62        </rules>
63      </configuration>
64    </execution>
65    <!------>
66  </executions>
67  <!------>
68 </plugin>
```