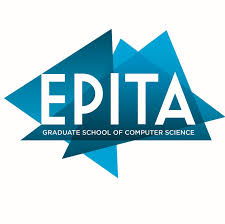
JDBCDAO

TUTORIAL

****

ASHEETH MANU

ME-2018

Table of Contents

[1 What is JDBC - DAO? 2](#_Toc474875462)

[1.1 Java Database Connectivity (JDBC) 2](#_Toc474875463)

[1.2 Data Access Object (DAO) 3](#_Toc474875464)

[2 JDBC-DAO in action in iAM – Identity Monitoring Application 4](#_Toc474875465)

[2.1 User Login 4](#_Toc474875466)

[2.2 Main Menu 5](#_Toc474875467)

[2.3 Create New User 6](#_Toc474875468)

[2.4 Modify User Record 7](#_Toc474875469)

[2.5 Delete User Record 8](#_Toc474875470)

[2.6 List All Users 9](#_Toc474875471)

[3 Bibliography 10](#_Toc474875472)

# What is JDBC - DAO?

## Java Database Connectivity (JDBC)

Java Database Connectivity (or JDBC, as it’s commonly referred to) was first introduced by Sun Microsystems as part of it’s Java Development Kit (or JDK) 1.1 in February, 1997. Since then it has become a standard part of the Java platform.

JDBC is an application programming interface (or API) for the JAVA language, which defines how a client may access a database. It provides methods to query and update data in a database (CRUD – Create, Read, Update and Delete), and is oriented towards relational databases. JDBC technology also provides a common base on which tools and alternate interfaces can be built.

The JDBC API is comprised of two packages :

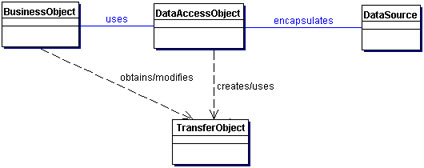
* java.sql
* javax.sql

JDBC allows multiple implementations to exist and be used by the same application. The Driver Manager is used as a connection factory for creating JDBC connections. JDBC supports creation and execution of SQL statements, SELECT statements or stored procedures.

## Data Access Object (DAO)

A data access object (or DAO) is an object that provides an abstract interface to a database. By mapping application calls to a “persistence” layer, the DAO provides specific data operations without exposing details of the database. It separates “what” data access the application needs, from “how” these needs can be specified with a specific database. While this is a design pattern equally applicable to all programming languages, it’s traditionally associated with Java applications with relational databases (using JDBC API).

The figure below shows the class diagram representing the relationships for the DAO pattern –



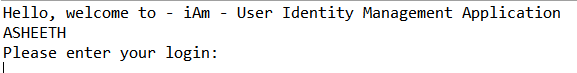
Since the DAO creates a separation between the application and the database itself, it’s possible to implement changes in business logic by making changes in only one DAO implementation, while leaving the database itself unchanged. DAO move data back and forth between objects and database records, thereby making the tests non-dependent on the persistence layer.

Potential disadvantages of using DAO are code duplication, resulting in multiple queries hitting the database for information which could have possibly be more efficiently satisfied by a single query. If an application requires multiple DAO’s, leading to repetitive codes, one way to optimize may be to implement a generic code to handle common operations.

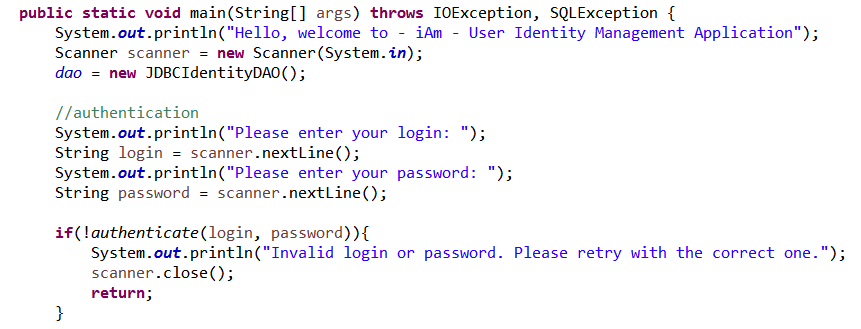
# JDBC-DAO in action in iAM – Identity Monitoring Application

## User Login

When the application is launched we see the welcome and the login screen –

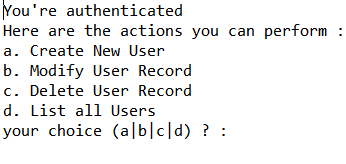


This corresponds to the following code–

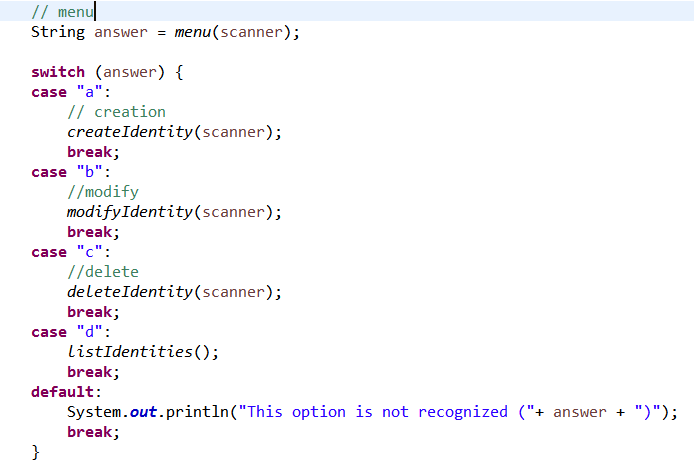


## Main Menu

Once the login has been successfully completed, the main menu is displayed

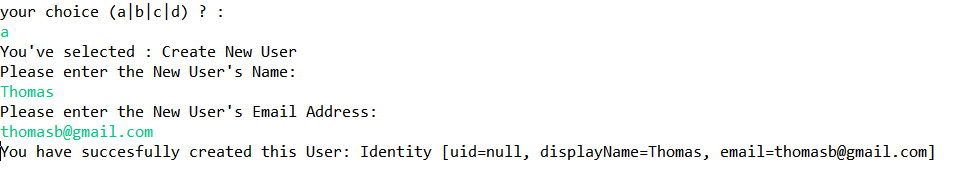


This is supported by the following code –

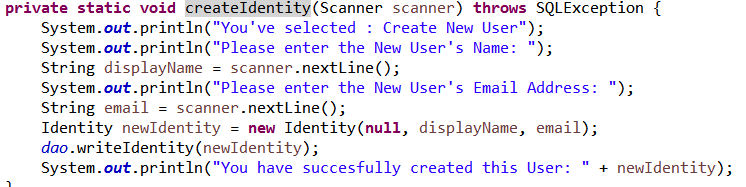


## Create New User

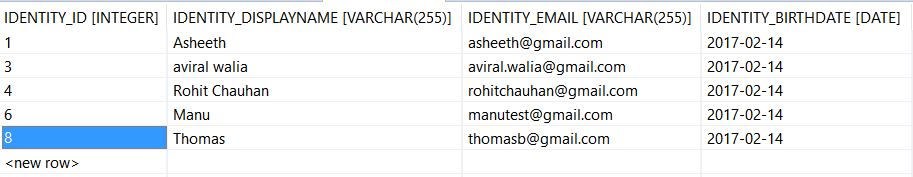
Once option “a” is selected, a new user can be added –



The corresponding code is –

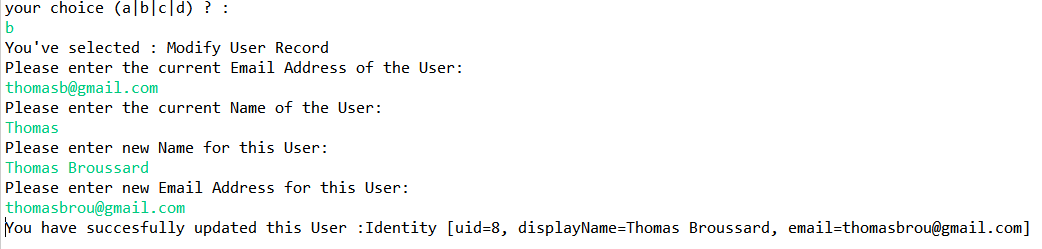


And the corresponding new entry is created in the database –

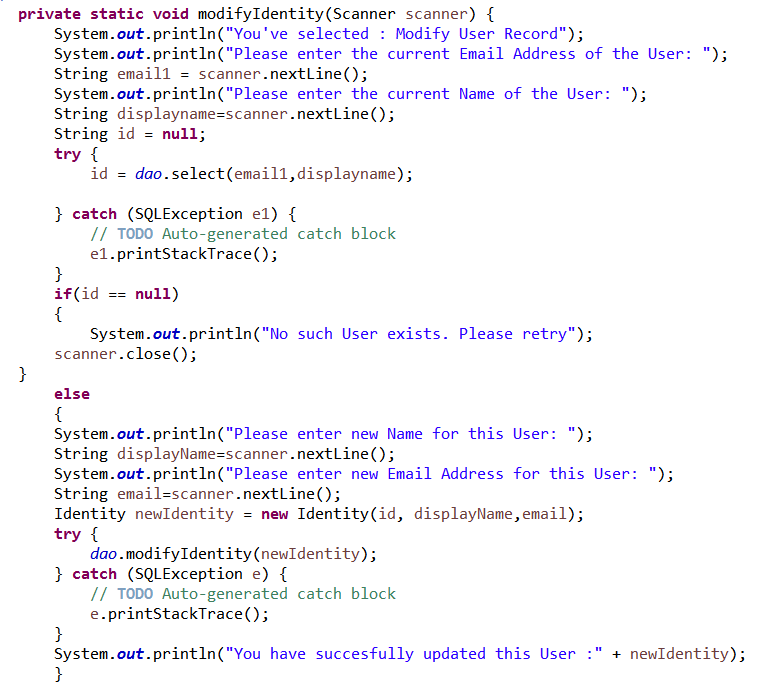


## Modify User Record

Similarly, when we modify a user,



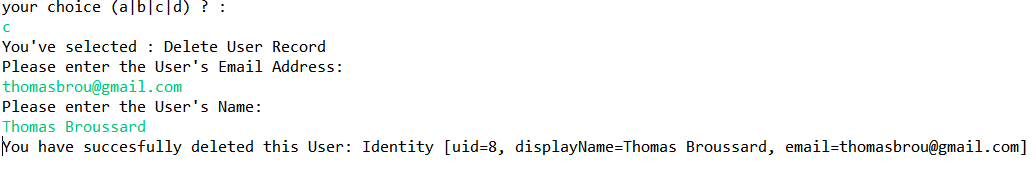
The following code is executed –



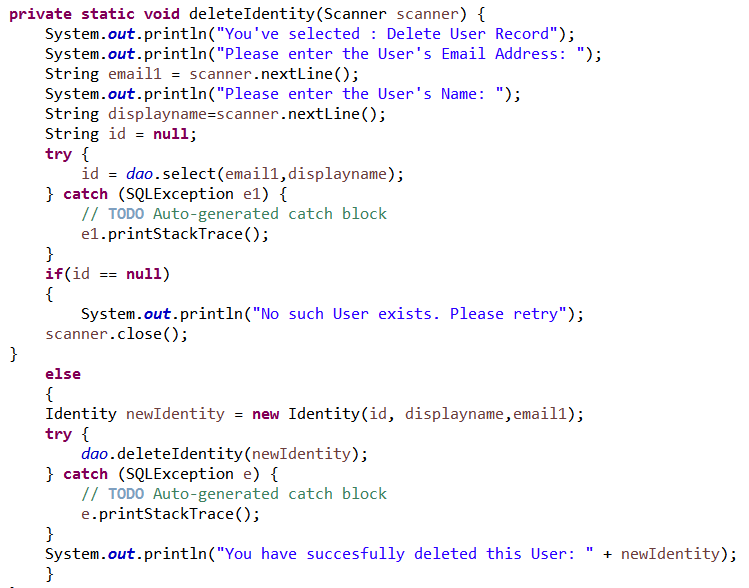
And the database entry for “Thomas” created in the previous step is updated –

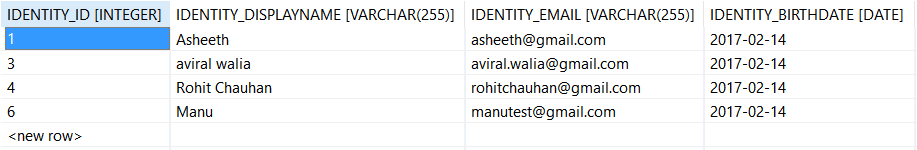
## Delete User Record

To delete a user, option “c” is selected, and the requisite information given –



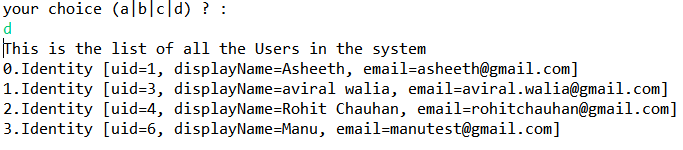
The following code is executed and the information is updated (record removed) in the database.

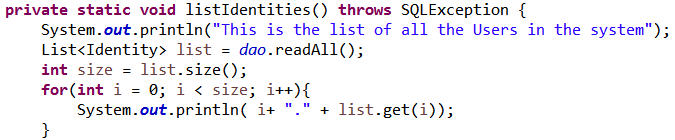




## List All Users

To list all users, the fourth option – “d” is selected, and it displays all the records in the database.





# Bibliography

* <https://en.wikipedia.org/wiki/Java_Database_Connectivity>
* <https://docs.oracle.com/javase/8/docs/technotes/guides/jdbc/>
* <https://en.wikipedia.org/wiki/Data_access_object>
* <http://www.oracle.com/technetwork/java/dataaccessobject-138824.html>
* <http://thomas-broussard.fr/index.xhtml>