B.Sc. In Software Development. Year 3. Applications Programming. Using the Apache Commons DBUtil Library.

LIMERICK INSTITUTE OF TECHNOLOGY

SCHOOL OF SCIENCE, ENGINEERING & I.T.

Department of Information Technology

Introduction

What is Apache Commons?

- A project of the Apache Software Foundation.
- Broken into three parts:
    1. The Commons Proper - A repository of reusable Java components.
    2. The Commons Sandbox - A workspace for Java component development.
    3. The Commons Dormant - A repository of components that are currently inactive.

DBUtil Library

A small set of classes designed to make working with JDBC easier.

Advantages of using DBUtil.

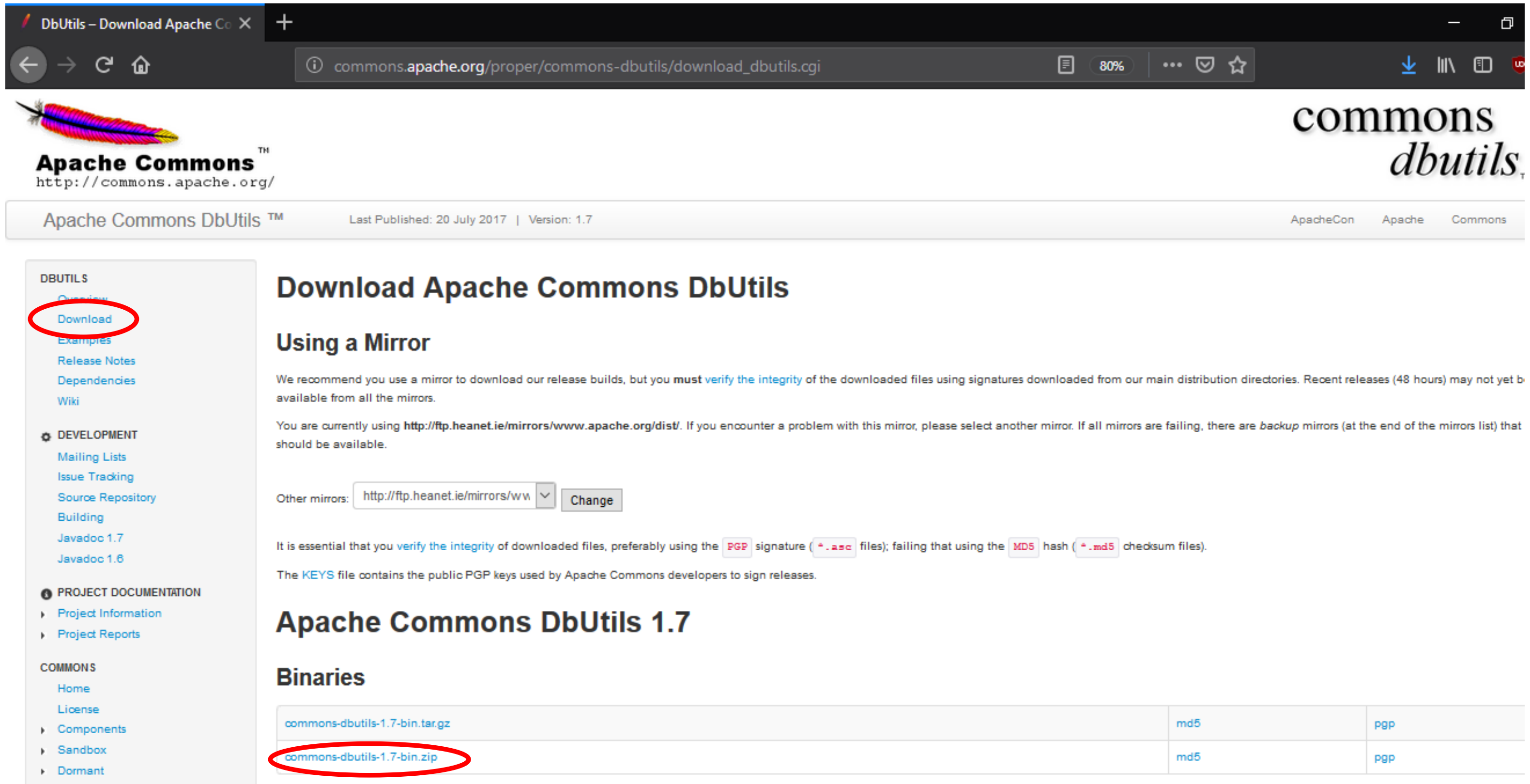
Its small size allows you to understand it fully in a short space of time.

Leads to clear and concise code.

Automatically populates JavaBean properties from ResultSets.

DBUtil Library

To use DBUtil, download the necessary jar files and add them to your project.


DbUtils – Download Apache Co

commons.apache.org/proper/commons-dbutils/download_dbutils.cgi

Apache Commons™ http://commons.apache.org/

commons dbutils

Apache Commons DbUtils ™ Last Published: 20 July 2017 | Version: 1.7 ApacheCon Apache Commons

DBUTILS
- Overview
- Download
- Examples
- Release Notes
- Dependencies
- Wiki

DEVELOPMENT
- Mailing Lists
- Issue Tracking
- Source Repository
- Building
- Javadoc 1.7
- Javadoc 1.6

PROJECT DOCUMENTATION
- Project Information
- Project Reports

COMMONS
- Home
- License
- Components
- Sandbox
- Dormant

Download Apache Commons DbUtils

Using a Mirror

We recommend you use a mirror to download our release builds, but you **must** verify the integrity of the downloaded files using signatures downloaded from our main distribution directories. Recent releases (48 hours) may not yet b available from all the mirrors.

You are currently using **http://ftp.heanet.ie/mirrors/www.apache.org/dist/**. If you encounter a problem with this mirror, please select another mirror. If all mirrors are failing, there are *backup* mirrors (at the end of the mirrors list) that should be available.

Other mirrors: http://ftp.heanet.ie/mirrors/ww Change

It is essential that you verify the integrity of downloaded files, preferably using the `PGP` signature (`*.asc` files); failing that using the `MD5` hash (`*.md5` checksum files).

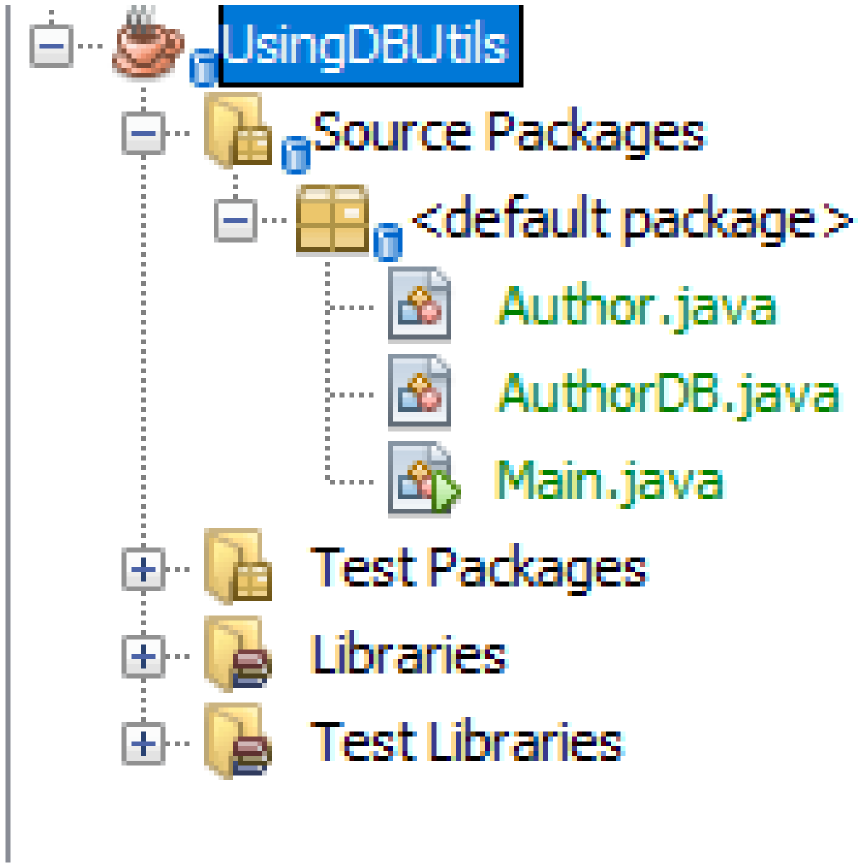
The KEYS file contains the public PGP keys used by Apache Commons developers to sign releases.

Apache Commons DbUtils 1.7

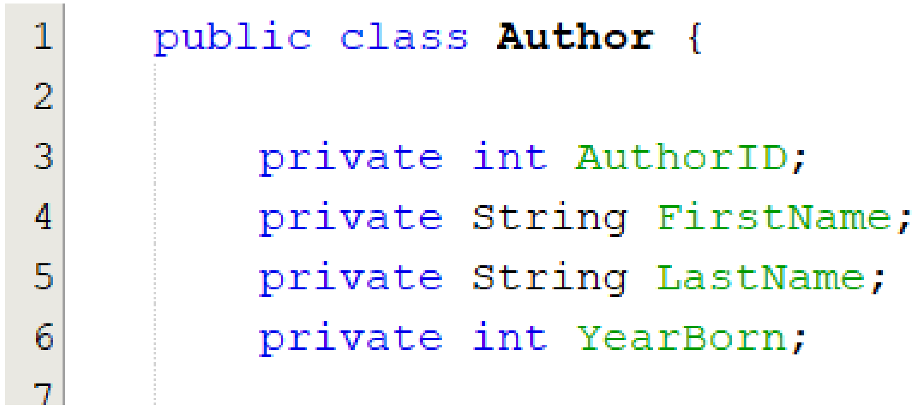
Binaries

commons-dbutils-1.7-bin.tar.gz	md5	pgp
commons-dbutils-1.7-bin.zip	md5	pgp

Anatomy of my project

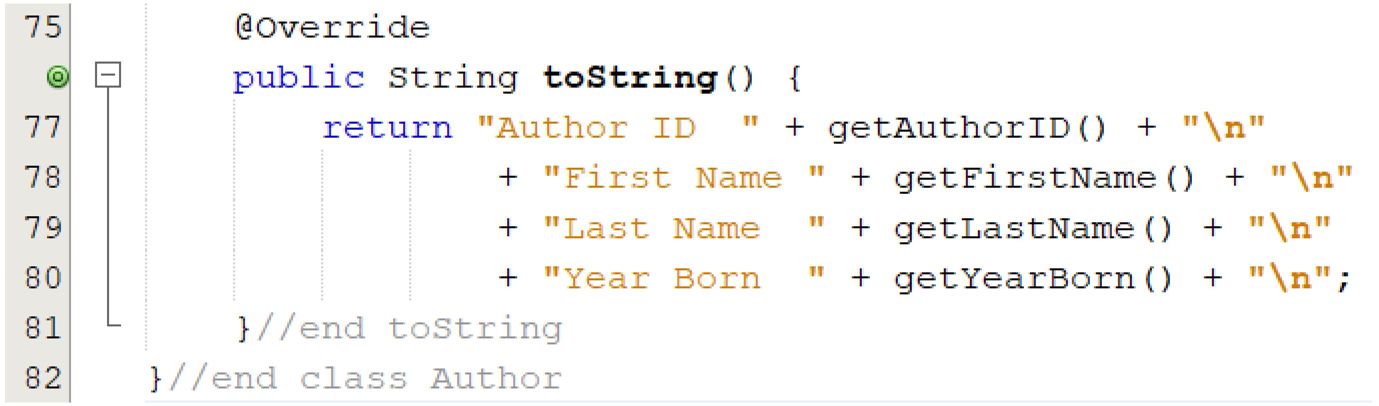


Create a JavaBean



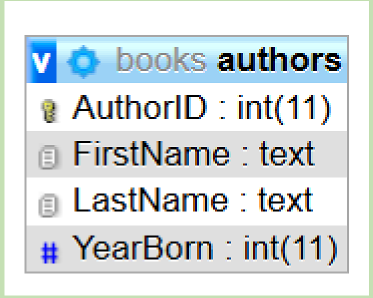
```java
public class Author {

    private int AuthorID;
    private String FirstName;
    private String LastName;
    private int YearBorn;

```

Constructors and getter/setters are omitted



```java
    @Override
    public String toString() {
        return "Author ID  " + getAuthorID() + "\n"
                + "First Name " + getFirstName() + "\n"
                + "Last Name  " + getLastName() + "\n"
                + "Year Born  " + getYearBorn() + "\n";
    }//end toString
}//end class Author
```

Structure of Authors table in the DB



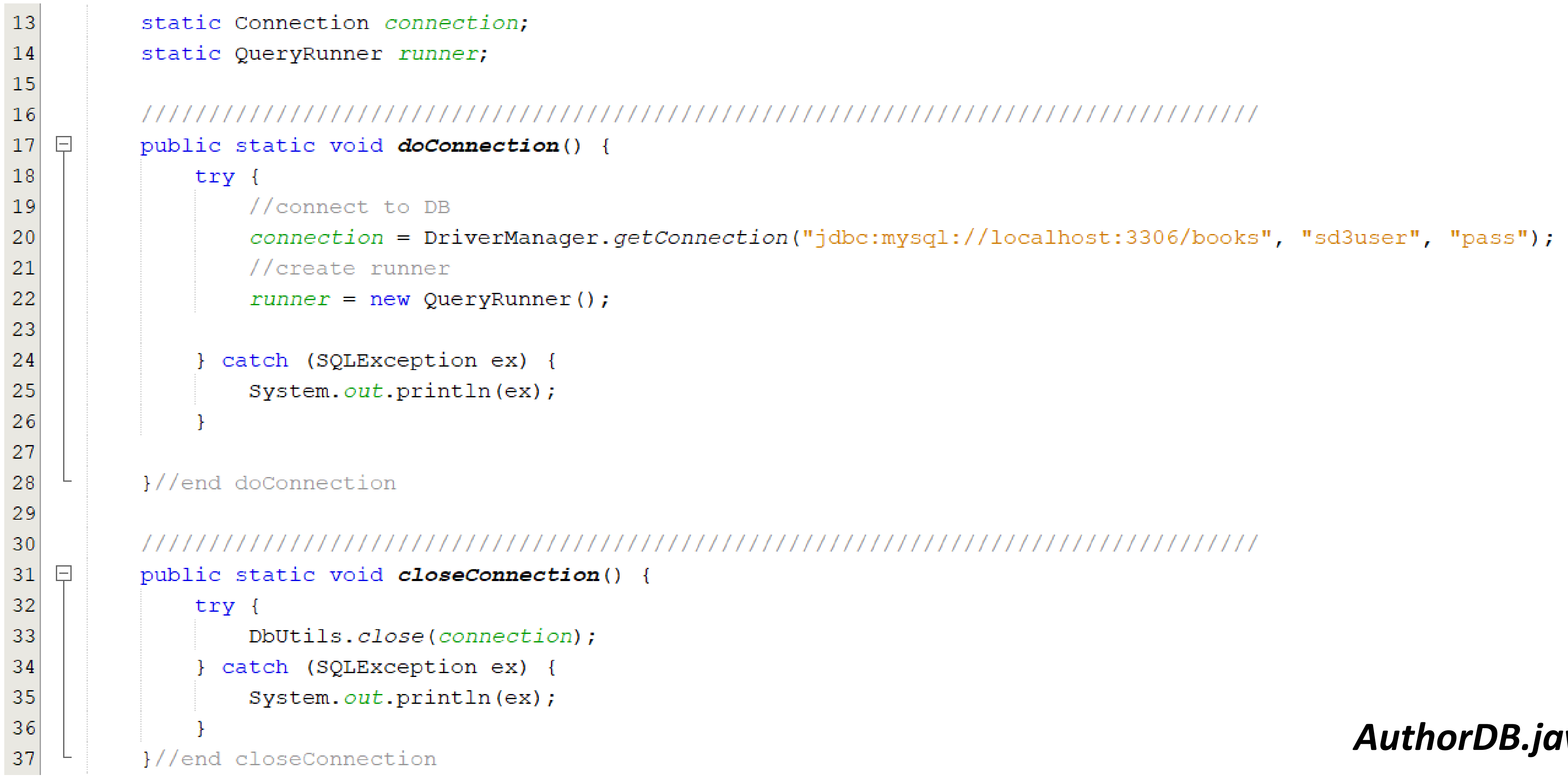
books **authors**

AuthorID : int(11)
FirstName : text
LastName : text
YearBorn : int(11)

Author.java

DBUtil Library

Connect the app to the DB.



```java
    static Connection connection;
    static QueryRunner runner;

    ///////////////////////////////////////////////////////////////////////////////////
    public static void doConnection() {
        try {
            //connect to DB
            connection = DriverManager.getConnection("jdbc:mysql://localhost:3306/books", "sd3user", "pass");
            //create runner
            runner = new QueryRunner();

        } catch (SQLException ex) {
            System.out.println(ex);
        }

    }//end doConnection

    ///////////////////////////////////////////////////////////////////////////////////
    public static void closeConnection() {
        try {
            DbUtils.close(connection);
        } catch (SQLException ex) {
            System.out.println(ex);
        }
    }//end closeConnection
```

AuthorDB.java

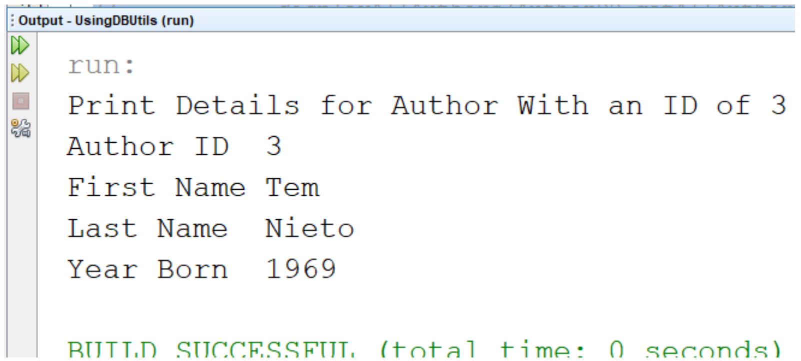
Retrieve a single record

```java
int id = 3;
System.out.println("Print Details for Author With an ID of " + id);
System.out.println(AuthorDB.getAuthorByID(id));
```

Main.java

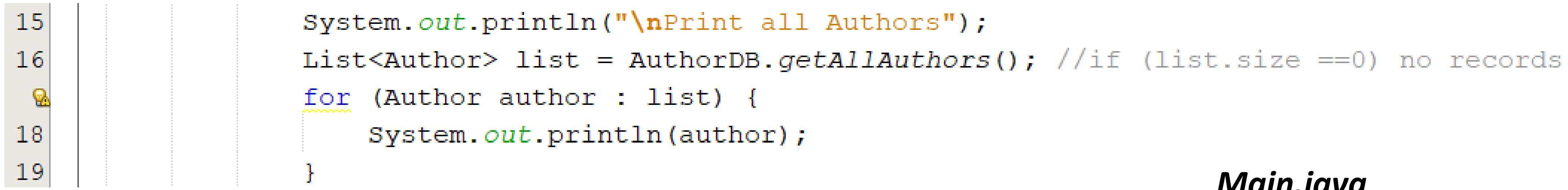
```java
public static Author getAuthorByID(int id) throws SQLException {
    ResultSetHandler<Author> handler = new BeanHandler(Author.class);
    return runner.query(connection, "SELECT * FROM authors WHERE AuthorID = ?", handler, id);
}//end getAuthorByID
```

AuthorDB.java


Output - UsingDBUtils (run)

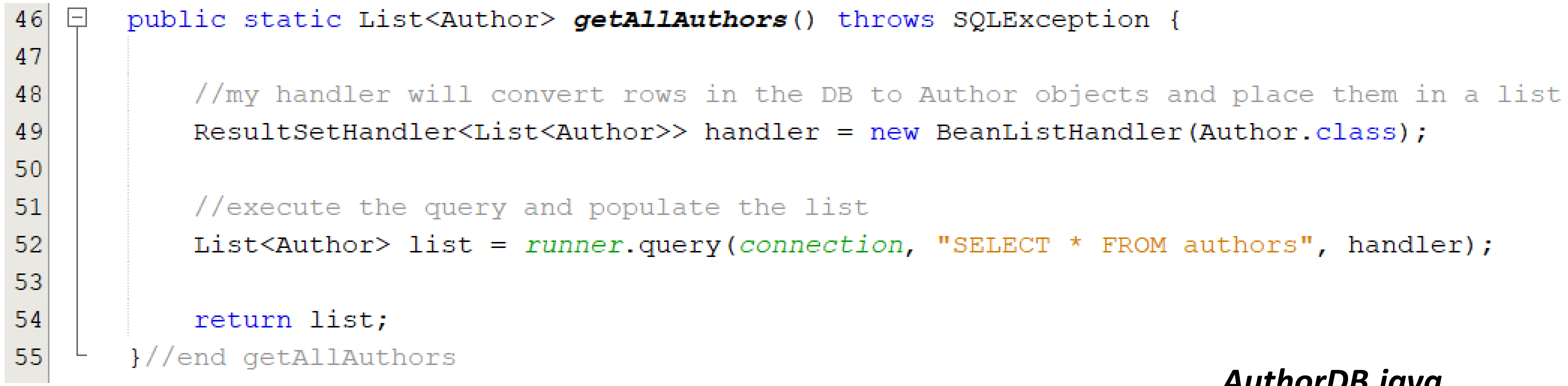
```
run:
Print Details for Author With an ID of 3
Author ID  3
First Name Tem
Last Name  Nieto
Year Born  1969

BUILD SUCCESSFUL (total time: 0 seconds)
```

Retrieve all records



```java
System.out.println("\nPrint all Authors");
List<Author> list = AuthorDB.getAllAuthors(); //if (list.size ==0) no records
for (Author author : list) {
    System.out.println(author);
}
```

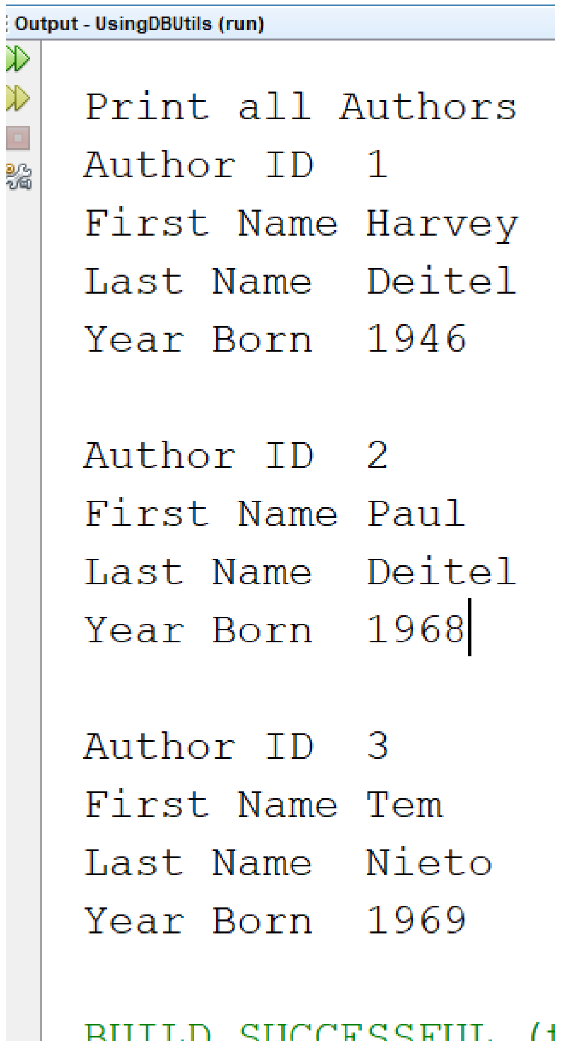
Main.java



```java
public static List<Author> getAllAuthors() throws SQLException {

    //my handler will convert rows in the DB to Author objects and place them in a list
    ResultSetHandler<List<Author>> handler = new BeanListHandler(Author.class);

    //execute the query and populate the list
    List<Author> list = runner.query(connection, "SELECT * FROM authors", handler);

    return list;
}//end getAllAuthors
```

AuthorDB.java

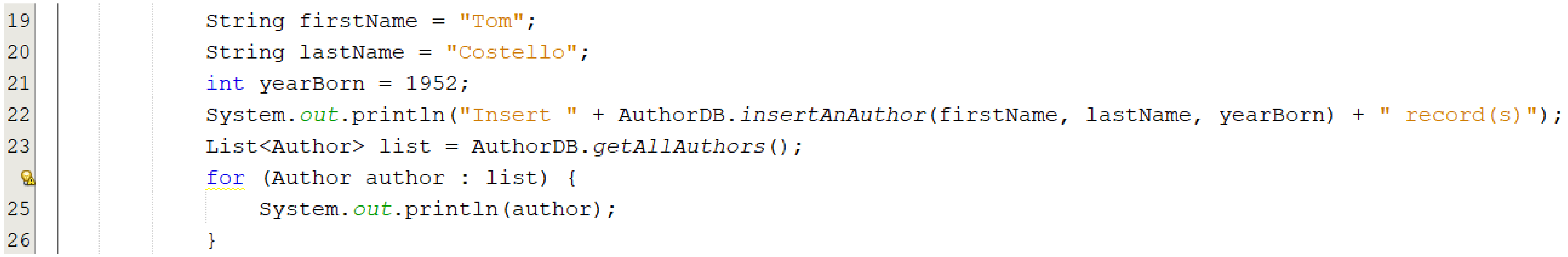
Retrieve all records



Output - UsingDBUtils (run)

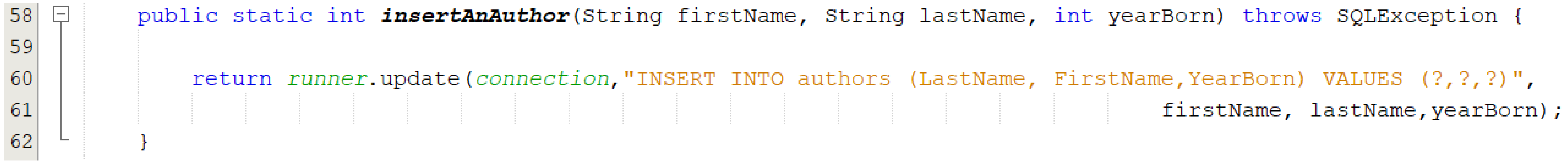
```
Print all Authors
Author ID  1
First Name Harvey
Last Name  Deitel
Year Born  1946

Author ID  2
First Name Paul
Last Name  Deitel
Year Born  1968

Author ID  3
First Name Tem
Last Name  Nieto
Year Born  1969

BUILD SUCCESSFUL (t
```

Insert a record



```java
String firstName = "Tom";
String lastName = "Costello";
int yearBorn = 1952;
System.out.println("Insert " + AuthorDB.insertAnAuthor(firstName, lastName, yearBorn) + " record(s)");
List<Author> list = AuthorDB.getAllAuthors();
for (Author author : list) {
    System.out.println(author);
}
```

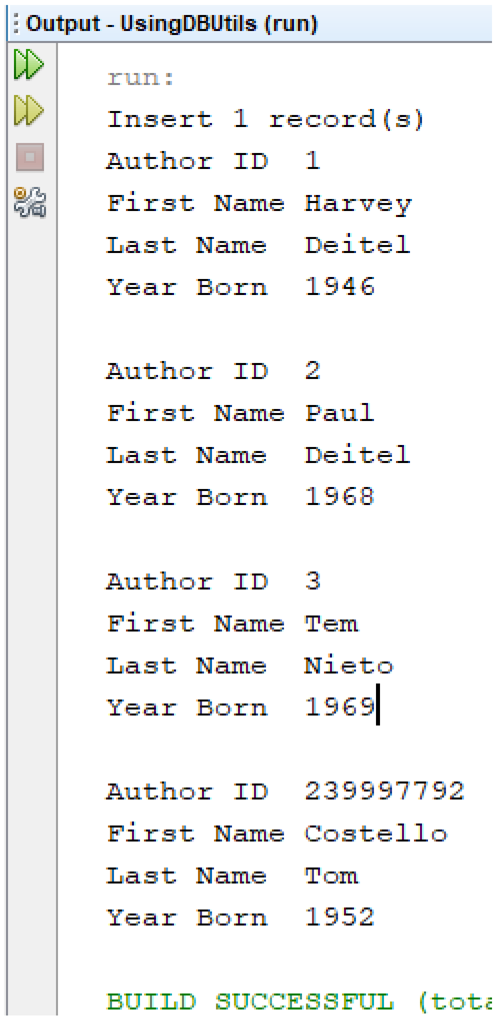
Main.java



```java
public static int insertAnAuthor(String firstName, String lastName, int yearBorn) throws SQLException {

    return runner.update(connection,"INSERT INTO authors (LastName, FirstName,YearBorn) VALUES (?,?,?)",
                                                                                firstName, lastName,yearBorn);
}
```

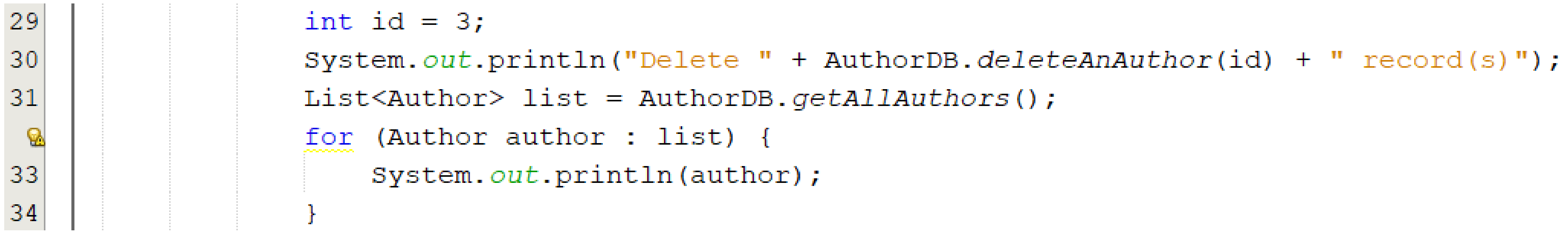
AuthorDB.java

Insert a record


Output - UsingDBUtils (run)

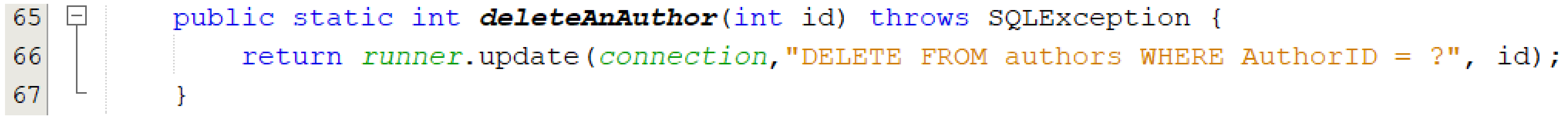
```
run:
Insert 1 record(s)
Author ID  1
First Name Harvey
Last Name  Deitel
Year Born  1946

Author ID  2
First Name Paul
Last Name  Deitel
Year Born  1968

Author ID  3
First Name Tem
Last Name  Nieto
Year Born  1969

Author ID  239997792
First Name Costello
Last Name  Tom
Year Born  1952

BUILD SUCCESSFUL (tota
```

Delete a record



```java
int id = 3;
System.out.println("Delete " + AuthorDB.deleteAnAuthor(id) + " record(s)");
List<Author> list = AuthorDB.getAllAuthors();
for (Author author : list) {
    System.out.println(author);
}
```

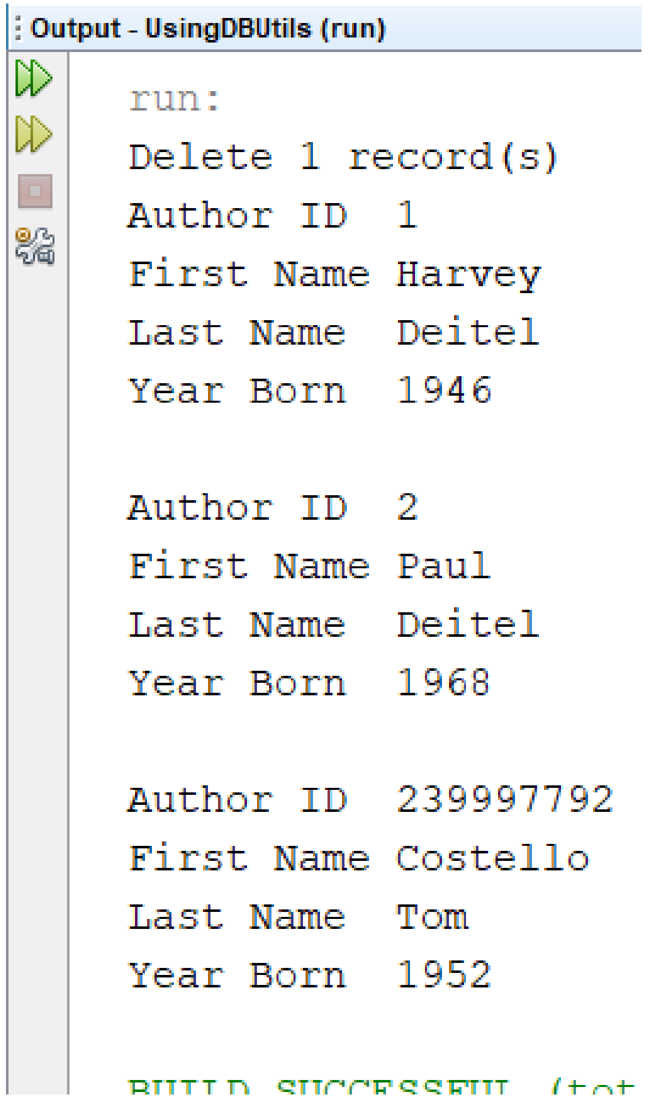
Main.java



```java
public static int deleteAnAuthor(int id) throws SQLException {
    return runner.update(connection,"DELETE FROM authors WHERE AuthorID = ?", id);
}
```

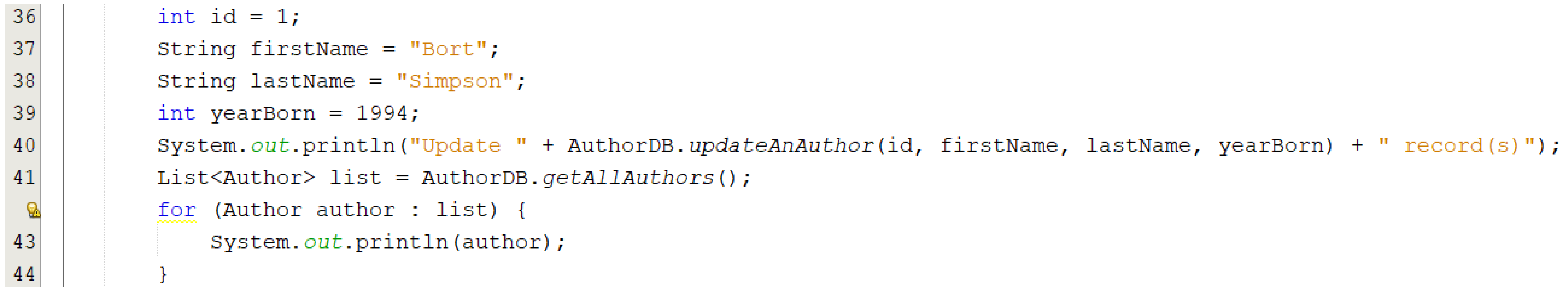
AuthorDB.java

Delete a record


Output - UsingDBUtils (run)

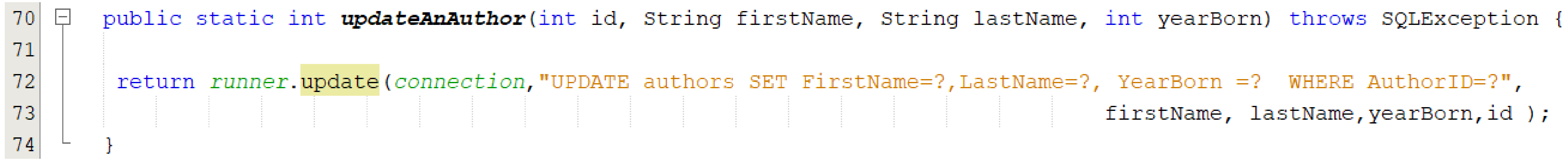
```
run:
Delete 1 record(s)
Author ID  1
First Name Harvey
Last Name  Deitel
Year Born  1946

Author ID  2
First Name Paul
Last Name  Deitel
Year Born  1968

Author ID  239997792
First Name Costello
Last Name  Tom
Year Born  1952

BUILD SUCCESSFUL (tot
```

Update a record



```
int id = 1;
String firstName = "Bort";
String lastName = "Simpson";
int yearBorn = 1994;
System.out.println("Update " + AuthorDB.updateAnAuthor(id, firstName, lastName, yearBorn) + " record(s)");
List<Author> list = AuthorDB.getAllAuthors();
for (Author author : list) {
    System.out.println(author);
}
```

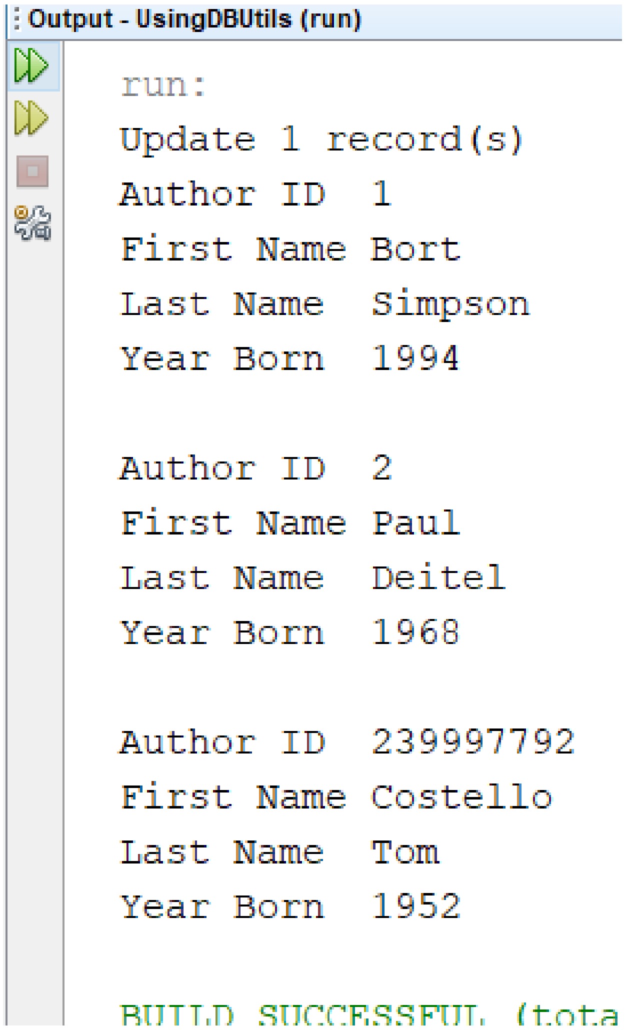
Main.java



```
public static int updateAnAuthor(int id, String firstName, String lastName, int yearBorn) throws SQLException {

 return runner.update(connection,"UPDATE authors SET FirstName=?,LastName=?, YearBorn =?  WHERE AuthorID=?",
                                                                                   firstName, lastName,yearBorn,id );
}
```

AuthorDB.java

Update a record



```
run:
Update 1 record(s)
Author ID  1
First Name Bort
Last Name  Simpson
Year Born  1994

Author ID  2
First Name Paul
Last Name  Deitel
Year Born  1968

Author ID  239997792
First Name Costello
Last Name  Tom
Year Born  1952

BUILD SUCCESSFUL (tota
```

References

http://commons.apache.org/proper/commons-dbutils/