

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.

The screenshot shows the Apache Commons DbUtils download page. The left sidebar contains a navigation menu with 'Download' circled in red. The main content area has a heading 'Download Apache Commons DbUtils' and a section 'Using a Mirror'. Below this is a table of mirrors. The 'Binaries' section at the bottom contains a table with download links. The link 'commons-dbutils-1.7-bin.zip' is circled in red.

Download Apache Commons DbUtils
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 be 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: [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).

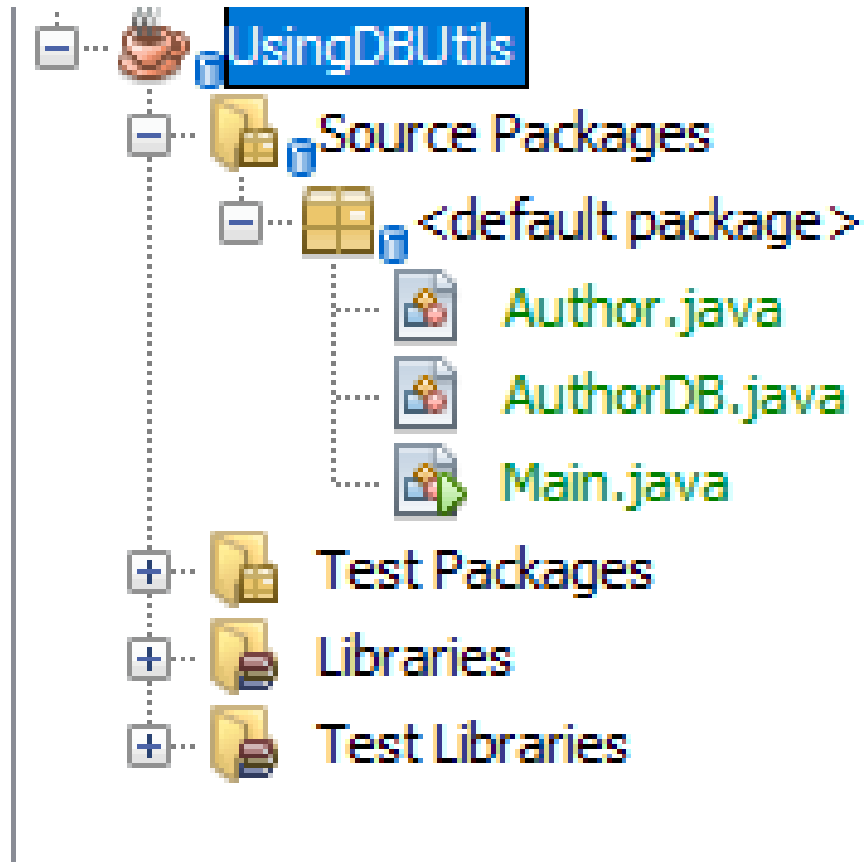
The [KEYS](#) file contains the public PGP keys used by Apache Commons developers to sign releases.

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

```
1 public class Author {  
2  
3     private int AuthorID;  
4     private String FirstName;  
5     private String LastName;  
6     private int YearBorn;  
7
```

Constructors and getter/setters are omitted

```
75     @Override  
76     public String toString() {  
77         return "Author ID " + getAuthorID() + "\n"  
78             + "First Name " + getFirstName() + "\n"  
79             + "Last Name " + getLastName() + "\n"  
80             + "Year Born " + getYearBorn() + "\n";  
81     } //end toString  
82 } //end class Author
```

DBUtil Library

Connect the app to the DB.

```
13 static Connection connection;
14 static QueryRunner runner;
15
16 ///////////////////////////////////////////////////
17 public static void doConnection() {
18     try {
19         //connect to DB
20         connection = DriverManager.getConnection("jdbc:mysql://localhost:3306/books", "sd3user", "pass");
21         //create runner
22         runner = new QueryRunner();
23
24     } catch (SQLException ex) {
25         System.out.println(ex);
26     }
27
28 } //end doConnection
29
30 ///////////////////////////////////////////////////
31 public static void closeConnection() {
32     try {
33         DbUtils.close(connection);
34     } catch (SQLException ex) {
35         System.out.println(ex);
36     }
37 } //end closeConnection
```

Retrieve a single record

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

Main.java


```
40      public static Author getAuthorByID(int id) throws SQLException {
41          ResultSetHandler<Author> handler = new BeanHandler(Author.class);
42          return runner.query(connection, "SELECT * FROM authors WHERE AuthorID = ?", handler, id);
43      } //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

```
15 System.out.println("\nPrint all Authors");
16 List<Author> list = AuthorDB.getAllAuthors(); //if (list.size ==0) no records
17  for (Author author : list) {
18     System.out.println(author);
19 }
```

Main.java

```
46 public static List<Author> getAllAuthors() throws SQLException {
47
48     //my handler will convert rows in the DB to Author objects and place them in a list
49     ResultSetHandler<List<Author>> handler = new BeanListHandler(Author.class);
50
51     //execute the query and populate the list
52     List<Author> list = runner.query(connection, "SELECT * FROM authors", handler);
53
54     return list;
55 } //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 (4
```

Insert a record

```
19 String firstName = "Tom";
20 String lastName = "Costello";
21 int yearBorn = 1952;
22 System.out.println("Insert " + AuthorDB.insertAnAuthor(firstName, lastName, yearBorn) + " record(s)");
23 List<Author> list = AuthorDB.getAllAuthors();
24 for (Author author : list) {
25     System.out.println(author);
26 }
```

Main.java

```
58 public static int insertAnAuthor(String firstName, String lastName, int yearBorn) throws SQLException {
59
60     return runner.update(connection, "INSERT INTO authors (LastName, FirstName, YearBorn) VALUES (?, ?, ?)",
61                             firstName, lastName, yearBorn);
62 }
```

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 (total time: 0s)
```

Delete a record

```
29 int id = 3;
30 System.out.println("Delete " + AuthorDB.deleteAnAuthor(id) + " record(s)");
31 List<Author> list = AuthorDB.getAllAuthors();
32 for (Author author : list) {
33     System.out.println(author);
34 }
```

Main.java

```
65 public static int deleteAnAuthor(int id) throws SQLException {
66     return runner.update(connection, "DELETE FROM authors WHERE AuthorID = ?", id);
67 }
```

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 (total
```

Update a record

```
36 int id = 1;
37 String firstName = "Bort";
38 String lastName = "Simpson";
39 int yearBorn = 1994;
40 System.out.println("Update " + AuthorDB.updateAnAuthor(id, firstName, lastName, yearBorn) + " record(s)");
41 List<Author> list = AuthorDB.getAllAuthors();
42 for (Author author : list) {
43     System.out.println(author);
44 }
```

Main.java

```
70 public static int updateAnAuthor(int id, String firstName, String lastName, int yearBorn) throws SQLException {
71
72     return runner.update(connection, "UPDATE authors SET FirstName=?, LastName=?, YearBorn =? WHERE AuthorID=?",
73                             firstName, lastName, yearBorn, id );
74 }
```

AuthorDB.java

Update a record

```
Output - UsingDBUtils (run)
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 (total
```


References

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