KIV/PIA Labs JDBC Project

Spring implementation of a simple web application.

1. Study the following section on handling database connections.
2. Study the section on passing arguments to queries.

Database Connection Management

This section provides overview of possible ways to close database connections maintained by JDBC. Many good reasons to do so can be found elsewhere.

Pre-JDK 7

In Java 6 and earlier, you have to close your resources manually:

```
Connection conn = null;
PreparedStatement stmt = null;
ResultSet set = null;

try {

    conn = DriverManager.getConnection("some url");
    stmt = conn.prepareStatement("some query");
    set = stmt.executeQuery();

    //process results

} catch (SQLException e) {
    //handle exception
} finally {
    if(set != null) {
        try {
            set.close();
        } catch (SQLException e) {
            //nothing left to do here, log this event and continue
        }
    }

    if(stmt != null) {
        try {
            stmt.close();
        } catch (SQLException e) {
            //nothing left to do here, log this event and continue
        }
    }

    if(conn != null) {
        try {
            conn.close();
        } catch (SQLException e) {
            //nothing left to do here, log this event and continue
        }
    }
}
```

Using Existing Utilities

We can use the `DbUtils` class for closing the resources:

```
Connection conn = null;
PreparedStatement stmt = null;
ResultSet set = null;

try {

    conn = DriverManager.getConnection("some url");
    stmt = conn.prepareStatement("some query");
    set = stmt.executeQuery();

    //process results

} catch (SQLException e) {
    //handle exception
} finally {
    DbUtils.closeQuietly(conn, stmt, set);
}
```

JDK 7 and Later

JDK 7 has introduced so-called try-with-resources statement.
The new language construct can be used to call the `close()` method of `Closeable` objects implicitly at the end of the try-catch block execution.

```
try (Connection conn = DriverManager.getConnection("some url");
    PreparedStatement stmt = conn.prepareStatement("some query");
    ResultSet set = stmt.executeQuery();) {

    //process results

} catch (SQLException e) {
    //handle exception
    // this is run after the resources have been closed
} finally {
    // this is run after the resources have been closed
}
```

Working with Prepared Statements

There are several good reasons for using *PreparedStatement*.

```
PreparedStatement stmt = conn.prepareStatement("SELECT * FROM users WHERE user_id = :?");
stmt.setLong(1, 33l);
```

Tasks

1. Implement org.danekja.edu.pia.dao.UserDaoJDBC.findByUsername using JDBC.
2. Test that registration and login work.

License