Cùi Bắp @euclid Follow
★ 3.3K 209 27

Published Aug 13th, 2015 1:42 PM - 3 min read
2.0K 0 6

Quản lý kết nối chung tới Database khi sử dụng JDBC

Java MySql Algorithm

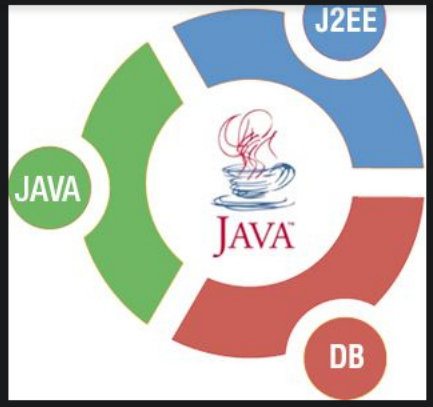
TABLE OF CONTENTS

SUGGESTED ORGANIZATIONS

1. Vấn Đề

Thời gian gần đây, do phải code ứng dụng chạy được trên cả server windows và unix nên tôi có quay trở lại code Java. (mặc dù có rất nhiều lựa chọn khác)

VIBLO Posts Questions Discussions Search Viblo Sign In/Sign up



Cảm giác ban đầu là code Java khá dài, và khoảng thời gian không dùng tới java cũng chừng 2 năm nên đôi lúc thấy rất nản. Tuy nhiên mọi việc đều suôn sẻ cho đến khi ứng dụng đến với end-user. (facepalm)

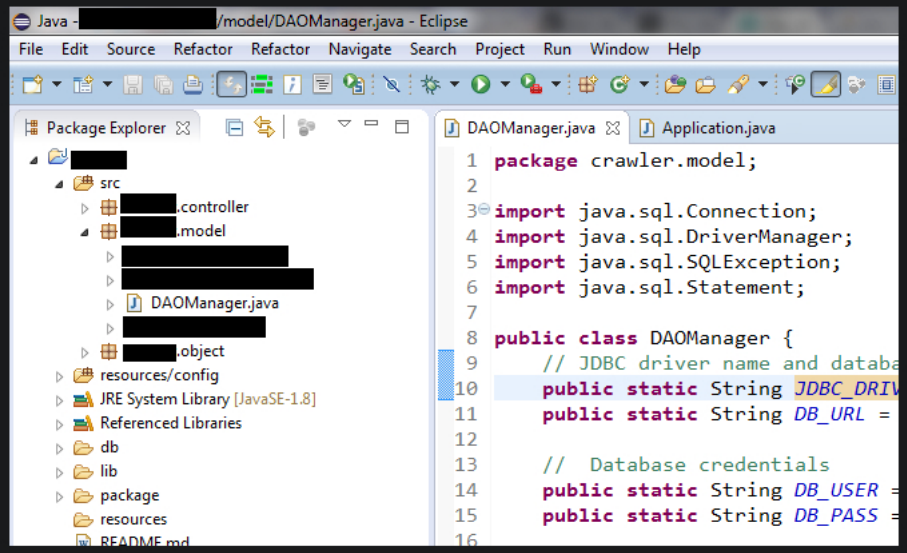
ứng dụng sử dụng JDBC để truy xuất và ghi dữ liệu, do trong lúc implement *class* Connection đã không chú ý, dẫn tới việc tạo ra quá nhiều instance, open nhiều hơn một connection tới db khi có một user sử dụng hệ thống. Dẫn tới database thường xuyên bị overload. (yaoming)

2. Phương án giải quyết

của kết nối, nếu đã có kết nối được tạo sẽ sử dụng kết nối đó, ngược lại sẽ mở kết nối mới tới db.

2.2. Cấu Trúc Ứng Dụng

Ở đây tui sử dụng DAO (Data Access Object) pattern, một mẫu kiến trúc rất thông dụng trong java, giúp chia nhỏ ứng dụng ra từng layer, nhằm quản lý dễ dàng hơn.



VIBLO Posts Questions Discussions Search Viblo Sign In/Sign up

+4

```
Import Java.Sql.Connection;
Import Java.Sql.DriverManager;
Import Java.Sql.SQLException;
Import Java.Sql.Statement;

Public Class DAOManager {
    // JDBC Driver Name And Database URL
    Public Static String JDBC_DRIVER = "Com.Mysql.Jdbc.Driver";
```

```
    Public Static String DB_URL = "Jdbc:mysql://Localhost/db_name";

    //  Database Credentials
    Public Static String DB_USER = "db_username";
    Public Static String DB_PASS = "db_password";

    // Database Connection
    Connection Conn = Null;
    Statement Stmt = Null;

}
```

2.3. Connect Database

Chúng ta thêm vào các method open, close phục vụ cho việc mở kết nối và đóng kết nối


```
public DAOManager() throws Exception
{
    try {
        Class.forName("com.mysql.jdbc.Driver");
        System.out.println("Created DB Connection....");
    } catch (ClassNotFoundException e) {
        // Handle errors for Class.forName
        throw e;
    } catch (Exception e) {
        // Handle errors for Exception
        throw e;
    }
}

public void open() throws SQLException {
    try {
        if (this.conn == null && this.stmt == null) {
            this.conn = DriverManager.getConnection(DB_URL, DB_USER, DB_PASS);
            this.stmt = conn.createStatement();
        }
    } catch (SQLException e) {
        // Handle errors for JDBC
        e.printStackTrace();
        throw e;
    }
}
```



```
        if (this.conn != null)
            this.conn.close();
    } catch(SQLException e) {
        // Handle errors for JDBC
        e.printStackTrace();
```

```
            throw e;
        }
    }
```

Đến đây chúng ta đã có thể chạy thử:

```
DAOManager daoMngr = new DAOManager();
daoMngr.open();
daoMngr.close();
```

Tuy nhiên chúng ta cần nhiều hơn thế, cho dù có nhiều instance object được tao ra thì vẫn chỉ có 1 kết nối duy nhất tới DB, việc này tránh lãng phí resource, nhất là khi connection open nhưng lại ko được close.

```
DAOManager daoMngr1 = new DAOManager();
DAOManager daoMngr2 = new DAOManager();
daoMngr1.open();
daoMngr2.open();
```


Như đã nói ở trên, để có duy nhất một kết nối, chúng ta trở về với ***Singleton Pattern***. Việc sử dụng **Singleton** cho phép **một và chỉ một instance cho mọi object**.

- Phải chuyển đổi public constructor sang private one. DAOManager trở thành một factory class!
- Thêm một private class chưa singleton.

↑ **getInstance()** được tạo ra để đáp ứng yêu cầu trên.

Tuy nhiên Singleton không phải là lý tưởng, thậm chí nó còn bị gọi là anti pattern, lấy ví dụ như: Nếu ứng dụng xử lý theo multithread, rõ ràng việc sử dụng Singleton để tạo ra duy nhất một instace cho toàn bộ ứng dụng là vô lý, khi đó multithread sẽ vẫn dùng chung một connection

(huytsao) Để giải quyết vấn đề đó **Java** cung cấp cho chúng ta một class tên là **ThreadLocal**. Mỗi ThreadLocal variable sẽ cho một instance trên mỗi thread. Dựa vào nguyên lý đó ta có thể giải quyết được vấn đề phát sinh.

http://docs.oracle.com/javase/6/docs/api/java/lang/ThreadLocal.html

```
/**
 * Implementation for one connection per user by singleton class
 *
```

```
    public static ThreadLocal<DAOManager> INSTANCE;
    static
    {
        ThreadLocal<DAOManager> dm;
        try {
            dm = new ThreadLocal<DAOManager>() {
                @Override protected DAOManager initialValue() {
                    try {
                        return new DAOManager();
                    } catch (Exception e) {
                        e.printStackTrace();
                        return null;
                    }
                }
            };
        } catch (Exception e) {
            e.printStackTrace();
            dm = null;
        }
        INSTANCE = dm;
    }
}

/**
 * A private Constructor prevents any other class from instantiating.
 *
 */
private static DAOManager getInstance() {
```

VIBLO Posts Questions Discussions Search Viblo Sign In/Sign up

+4

Để sử dụng DAOManager thực sự rất đơn giản:

```
DAOManager dao = DAOManager.getInstance();

// Open database connections
dao.open();

// Do something
...

// Close database connections
dao.close();
```

(thankyou) **for reading.**

3. Tham khảo

3. Tham khảo

- https://docs.oracle.com/javase/tutorial/jdbc/basics/processingsqlstatements.html
- http://docs.oracle.com/javase/6/docs/api/java/lang/ThreadLocal.html
- http://stackoverflow.com/questions/12812256/how-do-i-implement-a-dao-manager-using-jdbc-and-connection-pools

Related

Học Singleton Pattern trong 5 phút.
Doan Van Toan
6 min read
11750 18 5 29

Multithreading: Race Conditions, Critical Section...
Cùi Bắp
7 min read
3222 7 0 7

Tản mạn đôi chút về việc import dữ liệu lớn với Ruby...
Nguyen Trung Hieu
7 min read
125 2 0 3

Singleton Pattern trong java
Trần Văn Thành
3 min read
2840 1 2 2

More from Cùi Bắp

Go-lang: Anonymous fields in structs
247 11 1 12

Twitter đã deploy widgets JavaScript của họ như thế...
492 12 1 12

Angular 2 architecture overview (P2)
1537 17 1 11

Angular 2 architecture overview (P1)
3894 27 1 19

VIBLO Posts Questions Discussions Search Viblo Sign In/Sign up

Comments

Login to comment