

12.E. Cierre de conexión y excepciones.

Sitio: [Formación Profesional a Distancia](#)

Curso: Programación

Libro: 12.E. Cierre de conexión y excepciones.

Imprimido por: Iván Jiménez Utiel

Día: domingo, 17 de mayo de 2020, 16:28

Tabla de contenidos

[1. Cierre de conexiones.](#)

[2. Excepciones.](#)

1. Cierre de conexiones.

Las conexiones a una base de datos consumen muchos recursos en el sistema gestor por ende en el sistema informático en general. Por ello, conviene cerrarlas con el método `close()` siempre que vayan a dejar de ser utilizadas, en lugar de esperar a que el garbage collector de Java las elimine.

También conviene cerrar las consultas (`Statement` y `PreparedStatement`) y los resultados (`ResultSet`) para liberar los recursos.

2. Excepciones.

En todas las aplicaciones en general, y por tanto en las que acceden a bases de datos en particular, nos puede ocurrir con frecuencia que la aplicación no funciona, no muestra los datos de la base de datos que deseábamos, etc.

Es importante capturar las excepciones que puedan ocurrir para que el programa no aborte de manera abrupta. Además, es conveniente tratarlas para que nos den información sobre si el problema es que se está intentando acceder a una base de datos que no existe, o que el servicio MySQL no está arrancado, o que se ha intentado hacer alguna operación no permitida sobre la base de datos, como acceder con un usuario y contraseña no registrados, ...

Por tanto es conveniente emplear el [método](#) `getMessage()` de la [clase](#) `SQLException` para recoger y mostrar el mensaje de error que ha generado MySQL, lo que seguramente nos proporcionará una información más ajustada sobre lo que está fallando.

Cuando se produce un error se lanza una [excepción](#) del tipo `java.sql.SQLException`.

- Es importante que **las operaciones de acceso a base de datos** estén **dentro de un bloque try-catch** que gestione las excepciones.
- Los objetos del tipo `SQLException` tienen dos métodos muy útiles para obtener el código del error producido y el mensaje descriptivo del mismo, `getErrorCode()` y `getMessage()` respectivamente.

El [método](#) `getMessage()` imprime el mensaje de error asociado a la [excepción](#) que se ha producido, que aunque esté en inglés, nos ayuda a saber qué ha generado el error que causó la [excepción](#). El [método](#) `getErrorCode()`, devuelve un número entero que representa el código de error asociado. Habrá que consultar en la documentación para averiguar su significado.

Autoevaluación

El cierre de las conexiones y la gestión de excepciones sólo hay que efectuarla con bases de datos MySQL.

- ☐ Verdadero.
- ☐ Falso.