EXERCISE 3, EXCEPTIONS

Exercise 3, Exceptions

Objective

The primary objectives for this lab are to be able to throw and catch exceptions and have a full understanding of the Java runtime's propagation mechanism for exceptions until a handler is found.

Overview

Firstly, you will handle the exceptions by throwing an exception object in your Account class if balance goes into red.

In a second practical you will enhance the MotorwaySimulator to throw an exception when the RegistrationFactory runs out of registration plates.

In the third part of this lab you'll handle Java's Checked exceptions.

Part 1 – Step by step

Using try/throw/catch in the Account class

1. Add a new package to the **Labs** project called **lab03**.
2. Create a class in this package called **Program** with a *main()* method.
3. Create a class called Account with the following fields and methods
   a. int **id**; double **balance**; String **owner**
   b. Create a constructor to set these fields
   c. void **withdraw**(double amount) // reduces the balance by amount
   d. void **deposit**(double amount) // increases the balance by amount
   e. void **close**()
      just display a message like "account 123 is closed" where 123 is the Account's **id**.
   f. String **getDetails**() to get details of id,balance and owner.

      Add suitable code to the withdraw, deposit and getDetails() methods
4. Back in the main() method create an account with balance of £100.
5. Withdraw £50 and then display the account using the getDetails() method
6. Withdraw £60 and display the account's details
   Please note, there no withdrawal limit. Such an excellent account!
7. Fix the withdrawal() method so that it throws an `IllegalArgumentException` when the balance is negative,

8. Catch this exception in main() and display a suitable user friendly message.
9. Build and run the program now. Make sure you get the Account class to throw the exception and make sure it is handled in main().
10. Add a finally clause to the exception handler in main() to close the account no matter if there is an exception or not.

Part 2

11. Let's get back in the motorway simulator lab which you created in chapter 2.
12. The factory class will at some time run out of registration plates.
    How would you handle this situation?
    How would you inform the Vehicle class that we've ran out of reg plates?

    How would the Vehicle class inform the caller we cannot create a vehicle?

    Implement the required exception mechanisms the best you can

Part 3

13. Create a method in the Program class to write a message to a log file.
14. Type the following code:

```
private static void log(String msg){
        File file = new File("log.txt");
        FileWriter fr = new FileWriter(file, true);
        BufferedWriter br = new BufferedWriter(fr);
        br.write(msg + "\r\n");
        br.close();
        fr.close();
}
```

15. Call the log method from main()
16. Your code will not compile unless you change the signature of the log() method. It should indicate that it throws IOException
17. Catch this exception in main() and display a IOException's message, should the log() method fail.

** End **