CS F213 - Object Oriented Programming

J. Jennifer Ranjani

email: jennifer.ranjani@pilani.bits-pilani.ac.in

Chamber: 6121 P, NAB

Consultation: Appointment by e-mail

https://github.com/JenniferRanjani/Object-Oriented-Programming-with-Java

BITS Pilani

Pilani Campus

innovate achieve lead

BITS Pilani

Pilani Campus

Query asked during the previous class

Query asked in previous class

- In AWT, why are the AWT methods are invoked within an constructor?
- In the following case, the code works perfectly fine.

```
import java.awt.*;
import java.awt.event.*;
public class test {
    public static void main(String[] args)     {
    Frame f = new Frame("Adapter Example");
    f.setSize(300,300);
    f.setVisible(true);
    f.addWindowListener(new WindowAdapter() {
    public void windowClosing(WindowEvent e) {
    System.exit(0);    }
    });
    }
}
```

Query asked in previous class

```
import java.awt.*;
import java.awt.event.*;
public class test extends Frame {
   public static void main(String[] args)     {
    setSize(300,300);
    setVisible(true);
    addWindowListener(new WindowAdapter() {
    public void windowClosing(WindowEvent e) {
    System.exit(0);    }    });
    }
}
```

- This does not work because, non static methods cannot be invoked within a static method.
- Paint method is found in java.awt.Component class

Card Layout

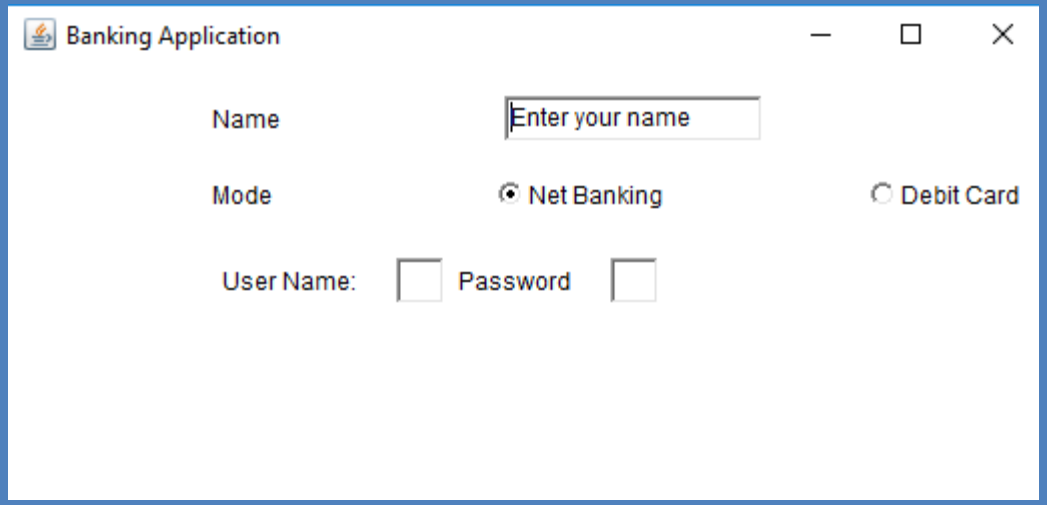
- It can be used to store several other layouts.
- Each layout can be thought of as being on a separate index card in a deck that can be shuffled.
- Other layouts can be prepared and kept ready to be activated when required.
- Cards are held in a Panel object and must have CardLayout selected as layout manager.
- A panel is created for each card in the deck. Add appropriate components in each card.
- Add these cards to the panel for which CardLayout is the layout manager.
- Add this panel to the window.

Methods used

- void add(Component *panelRef*, Object *name*);

- void first(Container *deck*)
- void last(Container *deck*)
- void next(Container *deck*)
- void previous(Container *deck*)
- void show(Container *deck*, String *cardName*)

Screen Shot


Banking Application
Name
Enter your name
Mode
Net Banking
Debit Card
User Name:
Password

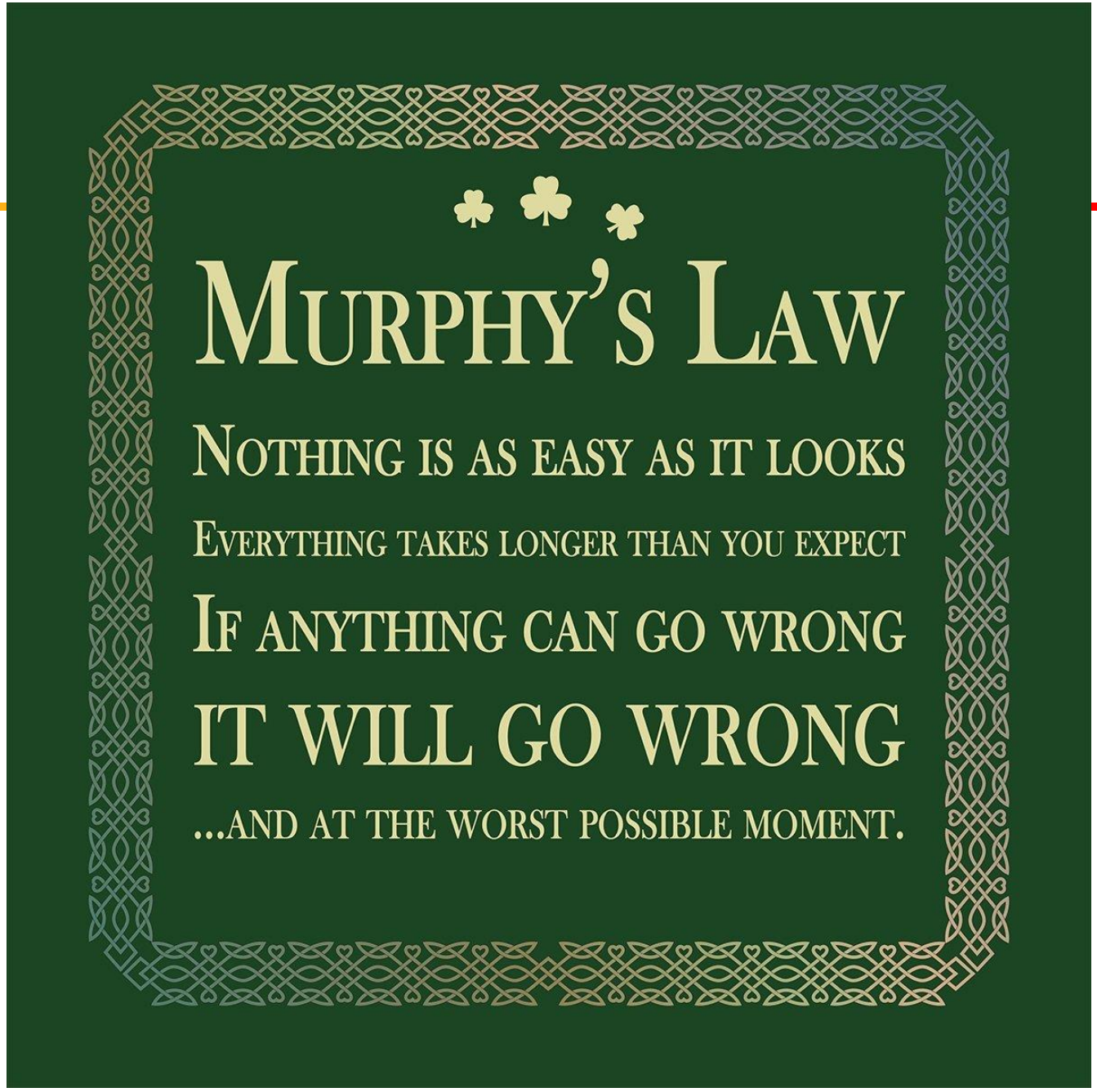

Banking Application
Name
Enter your name
Mode
Net Banking
Debit Card
Card No.
cw

innovate achieve lead

BITS Pilani

Pilani Campus

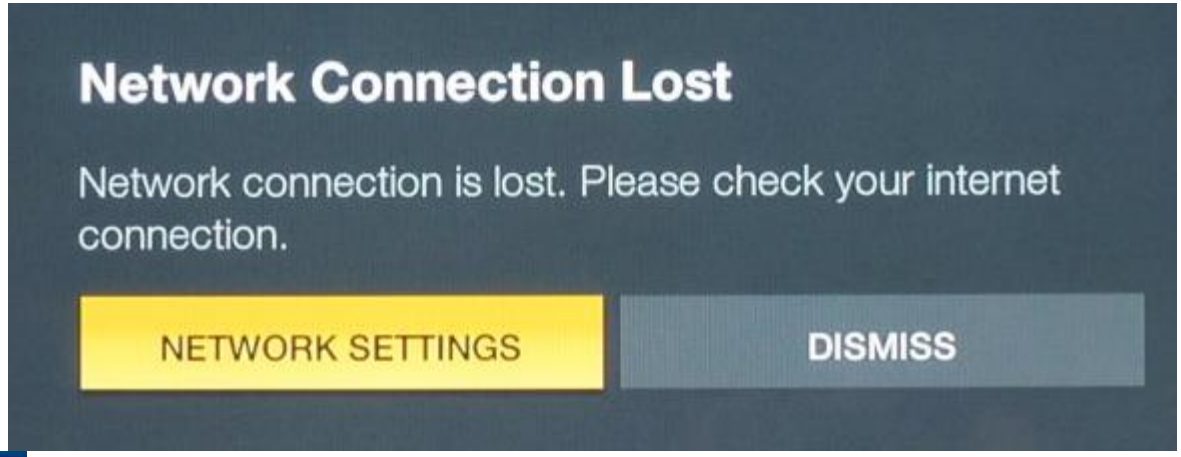
Exception Handling

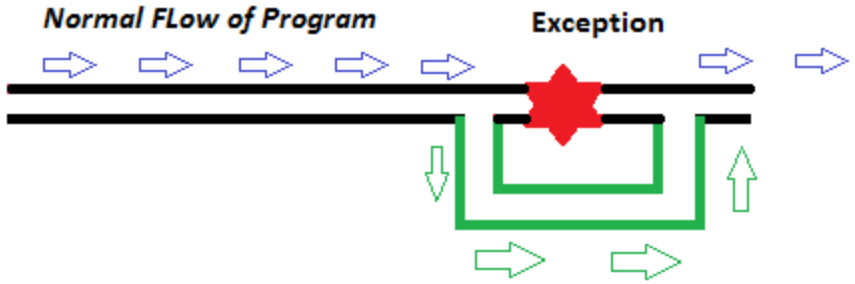

MURPHY'S LAW
NOTHING IS AS EASY AS IT LOOKS
EVERYTHING TAKES LONGER THAN YOU EXPECT
IF ANYTHING CAN GO WRONG
IT WILL GO WRONG
...AND AT THE WORST POSSIBLE MOMENT.

PCI :: Invalid Data

This transaction cannot be processed. Please enter a valid credit card number and type.

Go back and fix





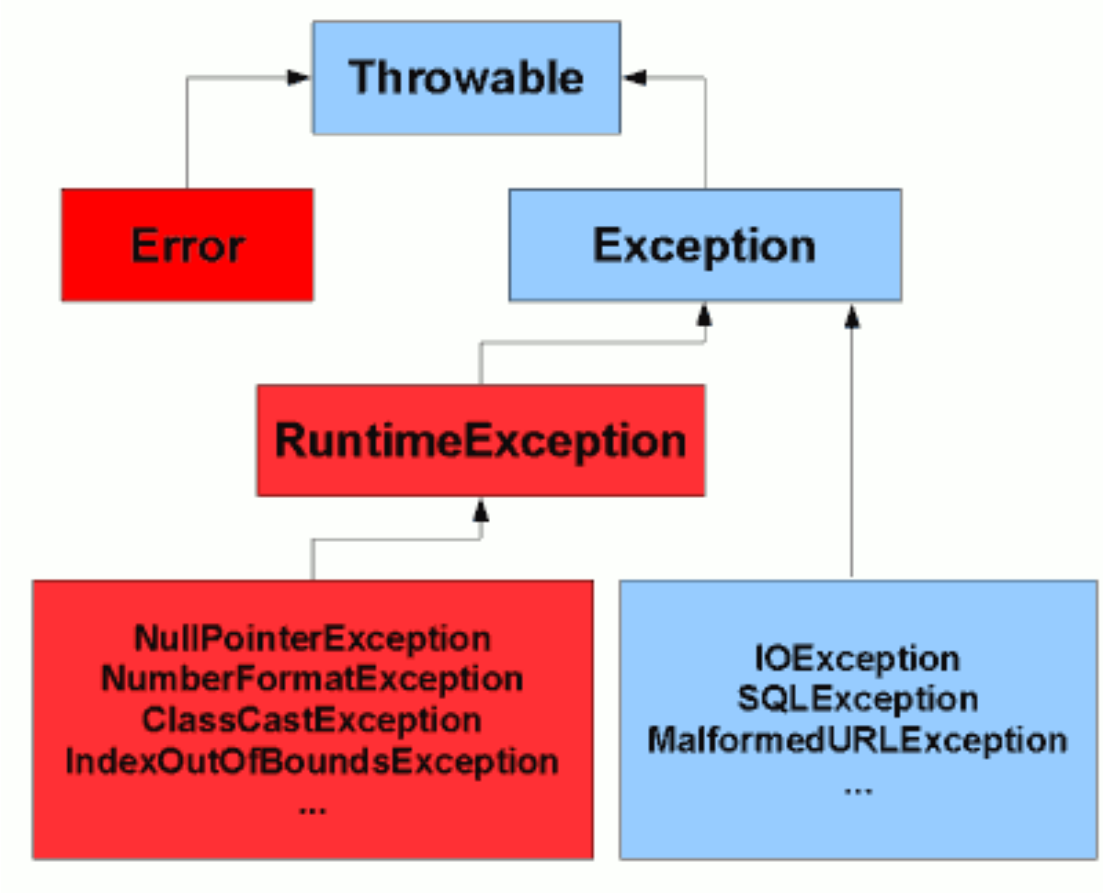
EXCEPTION HANDLING

Alternate way to continue flow of program

Fundamentals

- Exception is an object describing an exceptional condition.
- When exception arises, an object representing the exception is thrown in the method that caused the error.
- The method may chose to handle the exception by itself or pass it on.
- At some point the exception is caught and processed.
- Exception can be generated by JAVA run-time system or they can be manually generated by your code.
- Keywords: try, catch, throw, throws, finally

Exception Hierarchy



Exception Types

- All exceptions are sub class of the built in class Throwable
- Exception class is used for exceptional conditions the program should catch.
- Error defines exceptions that are not expected to be caught under normal circumstances.
  - Error is created in response to catastrophic failures that cannot be handled by the program
  - Eg. stack overflow

Checked vs. Unchecked Exceptions

innovate achieve lead

- **Checked exceptions (**compile time exceptions) – it occurs at the compile time. Cannot be ignored and the programmer should handle them.

```
FileTest.java:8: error: unreported exception FileNotFoundException;
must be caught or declared to be thrown

        FileReader fr = new FileReader(file);

^ 1 error
```

Checked vs. Unchecked Exceptions

innovate achieve lead

- **Unchecked exceptions (Runtime Exceptions)** − it occurs at the time of execution. These include programming bugs, such as logic errors or improper use of an API. Runtime exceptions are ignored at the time of compilation.

```
Exception in thread "main"
java.lang.ArrayIndexOutOfBoundsException: 5 at
Exceptions.FileTest.main(FileTest.java:8)
```

General Exception Handling Block