International study centre

AN INTRODUCTION TO OBJECT-ORIENTATION AND THE JAVA PROGRAMMING LANGUAGE

JAVA GUI DEVELOPMENT

❑Name : John Alamina

❑Email : john.alamina@hud.ac.uk

Outline

- Introduction to GUI development
- Setting up an application window
- Component Hierarchy
- JButton
- Labels and TextFields
- Basic Layouts in Java
- Event Handling
- Exception handling
- Exercises

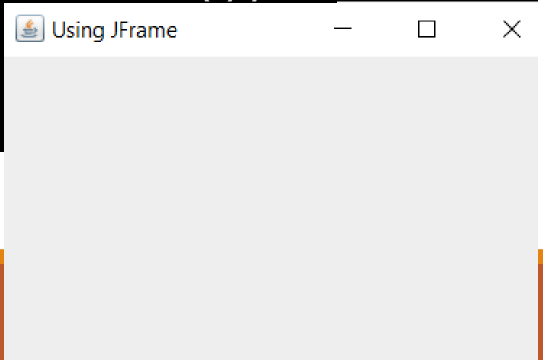
Introduction to GUI Development

- GUI is the acronym for Graphical User Interface
- The Graphical User Interface employs a WYSIWIG approach.
- WYSIWIG-what you see is what you get
- User interact with the computer using mini graphical screens called application windows occupying regions of the computer screen
- Interactions include pointing, clicking buttons, scrolling, typing into text fields/areas etc.
- There are different GUI toolkits in Java including SWT, JavaFX and Swing.

Setting up an application window

- The java swing library is the basic java GUI library.
- Swing has a number of packages.
- We will use the following packages.
  - javax.swing;
  - java.awt;
  - java.awt.event;

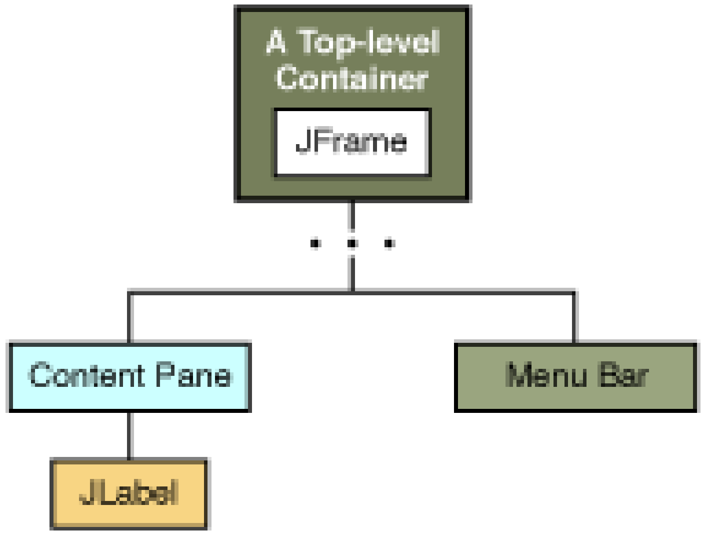
```
import javax.swing.JFrame;

public class CreateWindow{

    public CreateWindow(){
        JFrame frame = new JFrame("Usi
ng JFrame");
        frame.setSize(400,300);
        frame.setResizable(true);
        frame.setVisible(true);
    }
    public static void main(String ar[
]){
        new CreateWindow();
    }
}
```



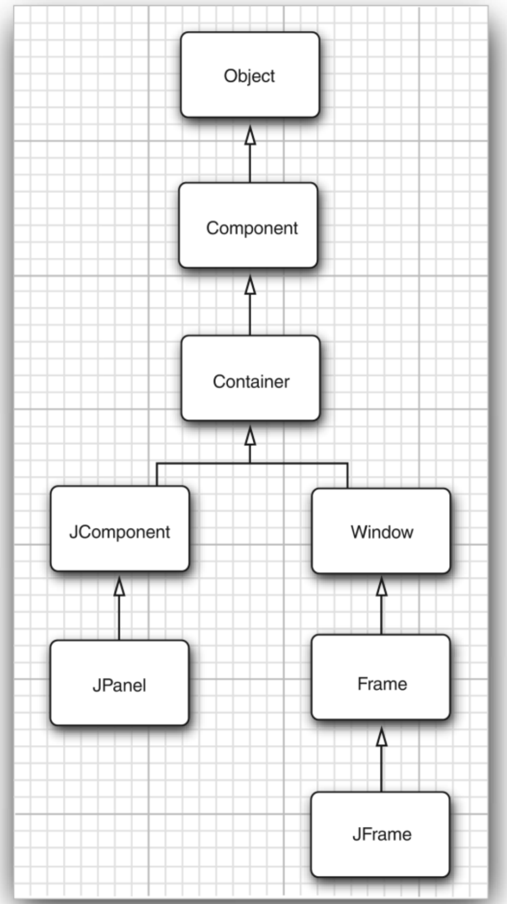
Any Questions?

Component Hierarchy

LAYING OUT COMPONENTS



COMPONENT INHERITANCE



Horstmann, C. S., & Cornell, G. (2013). *Core Java: Essential Features* (Vol. 1). Pearson Education.

Button

```java
public class BasicButton{

    public BasicButton(){
        JFrame frame = new JFrame("Using JFrame2");
        frame.setSize(400,300);
        frame.setResizable(true);
        frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);

        //creating a button
        JButton b=new JButton("Just a button");

        //add button to frame
        frame.getContentPane().add(b);

        frame.setVisible(true);
    }
    public static void main(String ar[]){
        new BasicButton();
    }
}
```

Any Questions?

Label and TextField

- A label and textfield can be created using the same process of creating a button
- TextField is an editable text component for entering text
- A label contains a non-editable text component.
- You can see how to work with different components in the java swing library here: https://docs.oracle.com/javase/tutorial/uiswing/components/componentlist.html

Behaviours inherited from the javax.swing.Component

- `comp.getWidth()` and `comp.getHeight()` are functions that give the current size of the component, in pixels
- `comp.setEnabled(true)` and `comp.setEnabled(false)` can be used to enable and disable the component.
- There is a boolean-valued function, `comp.isEnabled()` that you can call to discover whether the component is enabled.
- `comp.setVisible(true)` and `comp.setVisible(false)` can be called to hide or show the component.
- `comp.setFont(font)` sets the font that is used for text displayed on the component.
- `comp.setBackground(color)` and `comp.setForeground(color)` set the background and foreground colors for the component.
- `comp.setOpaque(true)` tells the component that the area occupied by the component should be filled with the component's background color before the content of the component is painted.
- `comp.setToolTipText(string)` sets the specified string as a "tool tip" for the component
- `comp.setPreferredSize(size)` sets the size at which the component should be displayed, if possible.

http://math.hws.edu/eck/cs124/javanotes7/c6/s5.html

Any Questions?

Basic Layouts

- There are various ways to layout components within a top-level window.
- The simplest way is by absolute positioning. The alternative is using a layout manager
- These include
  - Flow Layout
  - Border Layout
  - Grid layout
- The basic layouts can be combined together to form complex layouts.
- We can use a wireframe diagram to determine how to layout our components and what layouts can be used to achieve a specific design.

Flow and Grid Layout

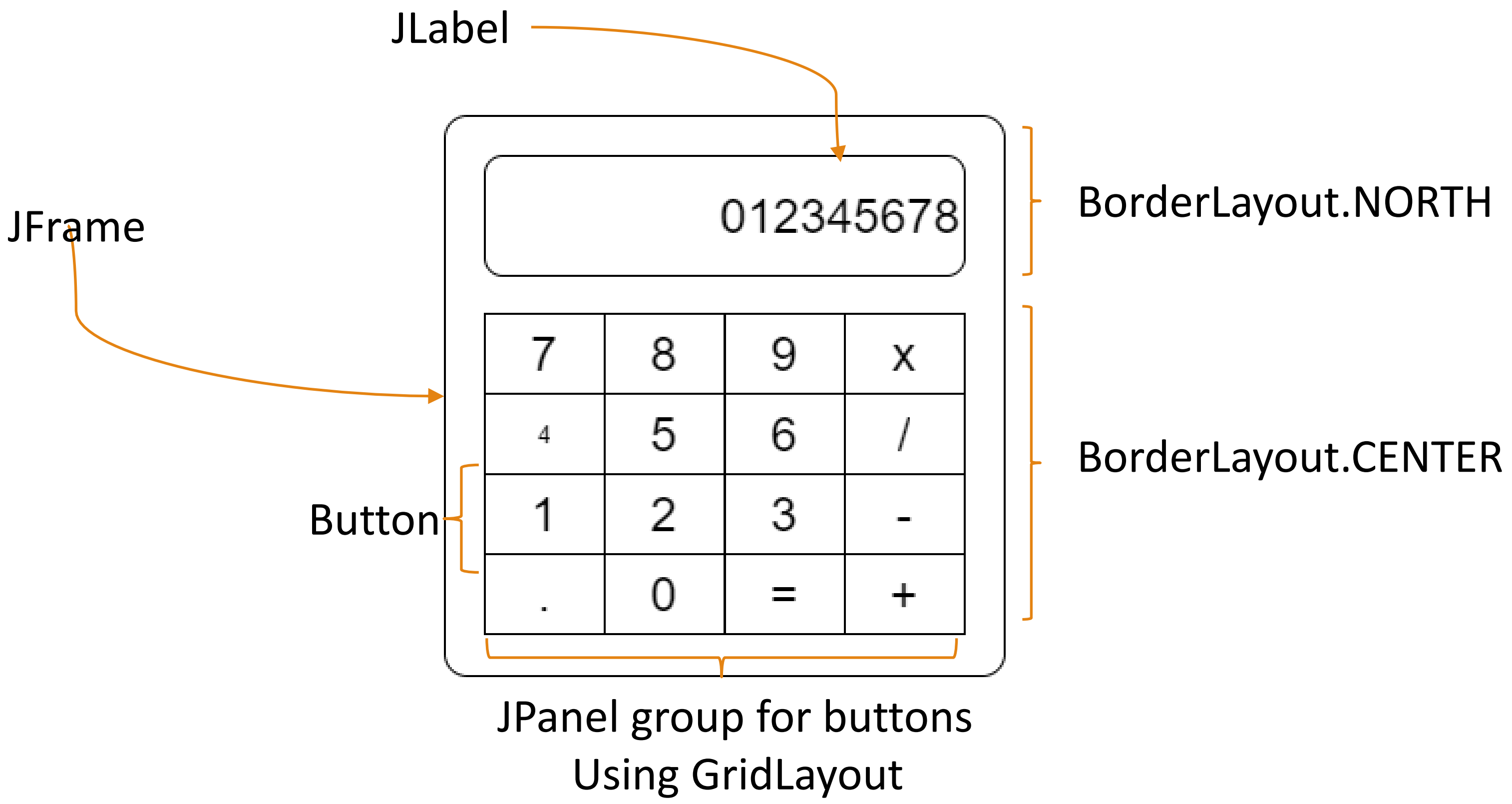
- This is the default layout manager for the JPanel class
- The JPanel class is used to group components together
- The Flow layout lays out components from left to right and then top to bottom.
- The grid layout lays out components in predefined rows and columns of a grid

Border Layout

- Border Layout is the Default layout for the JFrame class
- Border layout has 5 positions
  - Top (header) NORTH position
  - Bottom (footer) SOUTH position
  - flush right EAST position
  - flush left WEST position
  - CENTER
- The default position for the border layout is CENTER

Any Questions?

Simple Calculator wireframe



Simple calculator layout

```
import javax.swing.JFrame;
import javax.swing.JPanel;
import javax.swing.JButton;
import javax.swing.JTextField;
import java.awt.BorderLayout;

public class SimpleCalc extends JFrame{

    public SimpleCalc(){

        //initialise frame
        this.initFrame();

        //create button grid
        JPanel buttons=this.createButtonGrid();

        //creating a button
        JTextField display=new JTextField("0",1
0);

        //add contents and show frame
        getContentPane().add(buttons,BorderLayo
ut.CENTER);
        getContentPane().add(display,BorderLayo
ut.NORTH);
        pack();
        setVisible(true);
    }
}
```

```
    private JPanel createButtonGrid(){
        JPanel buttons=new JPanel();
        buttons.setLayout(new GridLayout(5,4));
        buttons.add(new JButton("A/C"));
        buttons.add(new JButton("C"));
        buttons.add(new JButton("M+"));
        buttons.add(new JButton("+/-"));
        buttons.add(new JButton("7"));
        buttons.add(new JButton("8"));
        buttons.add(new JButton("9"));
        buttons.add(new JButton("x"));
        buttons.add(new JButton("4"));
        buttons.add(new JButton("5"));
        buttons.add(new JButton("6"));
        buttons.add(new JButton("/"));
        buttons.add(new JButton("1"));
        buttons.add(new JButton("2"));
        buttons.add(new JButton("3"));
        buttons.add(new JButton("-"));
        buttons.add(new JButton("."));
        buttons.add(new JButton("0"));
        buttons.add(new JButton("="));
        buttons.add(new JButton("+"));
        return buttons;
    }
```

Any Questions?

Exception Handling

- A mechanism of recovering or gracefully exiting from anticipated runtime errors.
- We can anticipate that in a division may include a denominator of 0, resulting in a Divide-By-Zero Exception.
- We can also anticipate that a file requested by our program to read from may not exist in the operating system file path given, resulting in a File-Not-Found exception.
- There are many of such predefined Exception defined by the Java library and we can also define our own exceptions, or use the default Exception class.
- Two ways of handling exceptions including either the throws keyword on an encapsulating method or using the try-catch-finally block.
- We can also generate predefined or user-defined exceptions using by throwing an exception using the “throw” keyword.

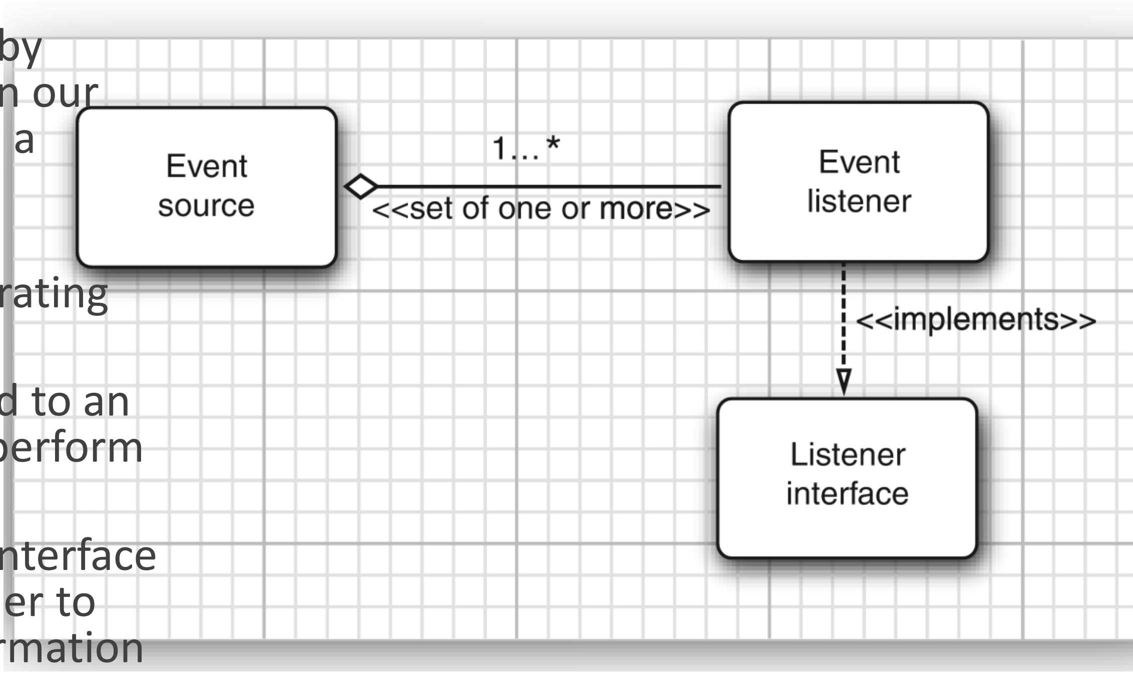
Divide-by-zero Exception handling example

```
private void div(double x){
  try{
    setVal(getVal()/x);
  } catch(Exception e){
    System.out.println(e.getMessage());
  };
}
```

Any Questions?

Event handling

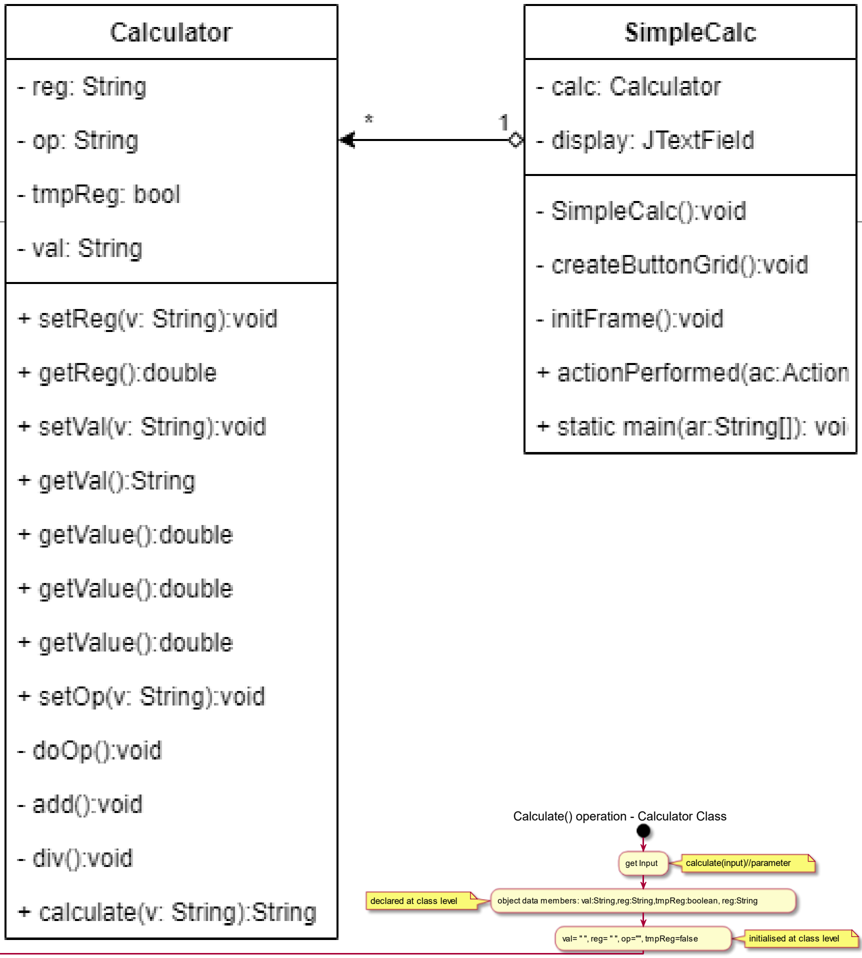
- An GUI event is a user activity that the program responds to.
- These events are activated by interactions from the user on our components such as clicking a button
- An event source is a GUI component capable of generating events.
- An event listener is attached to an event source and is able to perform action based on events
- In Java swing, the Listener interface is implemented by the listener to respond to and acquire information about subscribed events

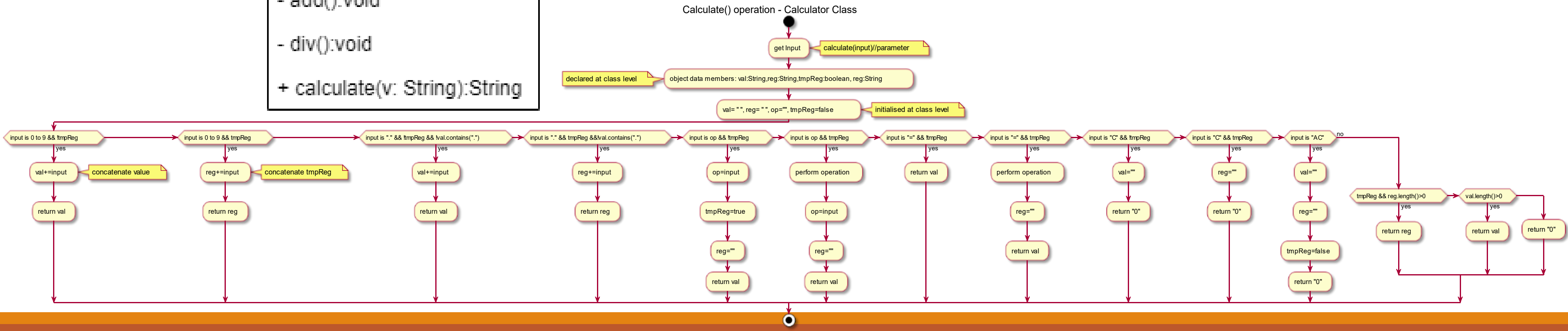


Simple Calculator events

```
void actionPerformed(ActionEvent evt){
    String op=evt.getActionCommand();
    display.setText(calc.calculate(op));
}
```

Simple Calculator Design





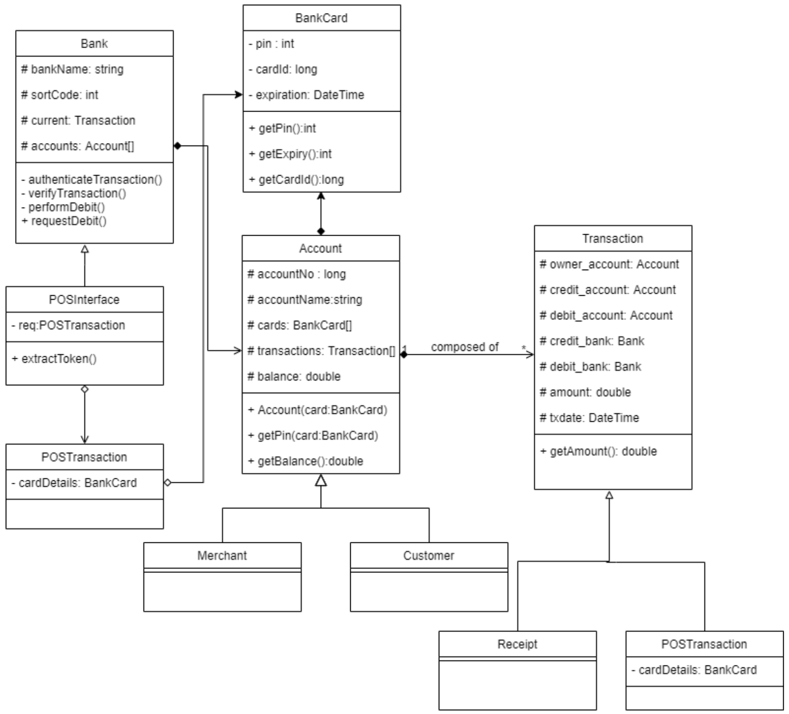
Any Questions?

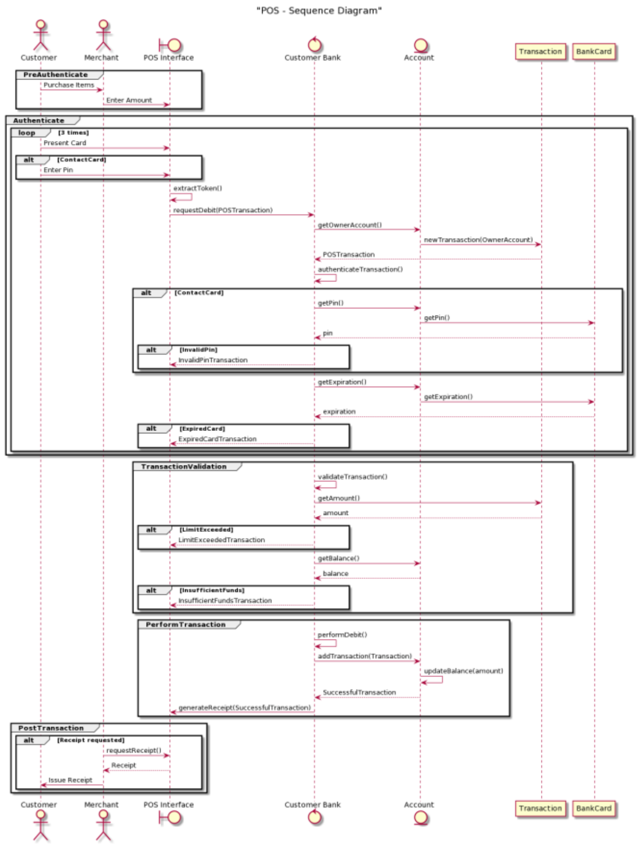
Exercises

1. Create an exception handler for the appropriate portion of the Loan Calculator program developed in Lesson 3.
2. Create a simple Calculator application using javax.swing GUI library.
3. Create a wireframe diagram for the address book program.
4. Create a GUI for the address book program.
5. (advanced) Complete your address book program by adding the appropriate event handler methods.

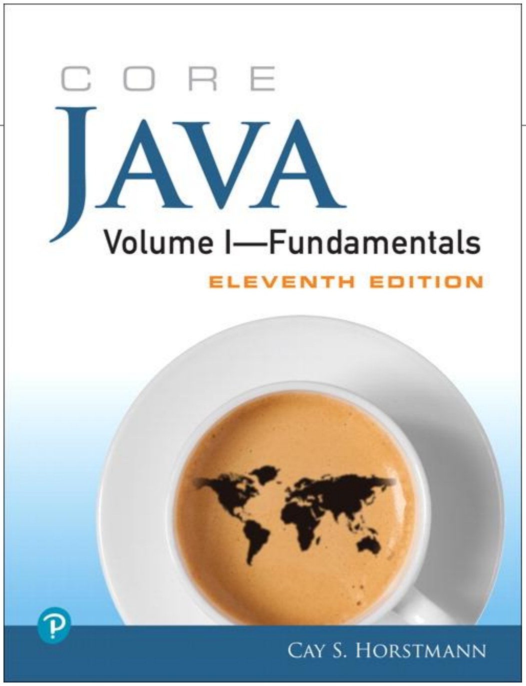
Java Bootcamp challenge

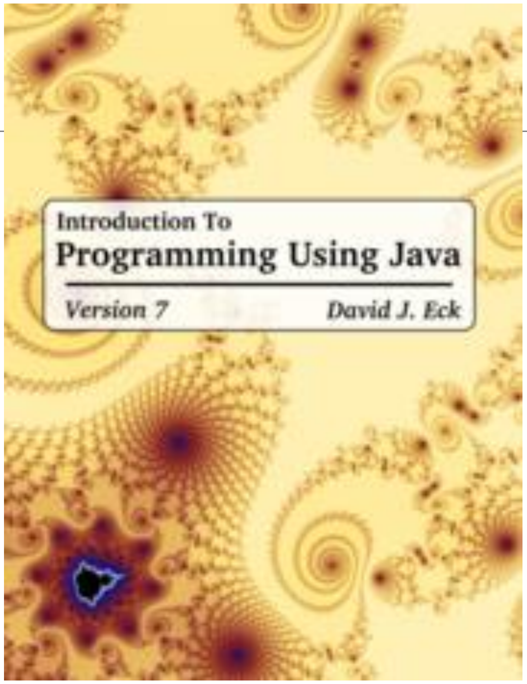
- Create a GUI application for the POS system given by the following class and sequence diagram (click on image to open in browser)





Supplementary material





- The Java Tutorial
- Java API documentation
- Link to today's Session screencast
- Link to John's Group Padlet
- Link to Kelly's Group Padlet