Tutorial Teaching Week 4

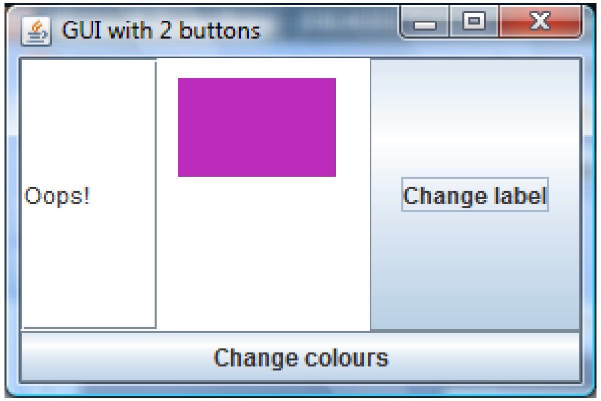
Topics:

- GUI
- Exception Handling
- File I/O

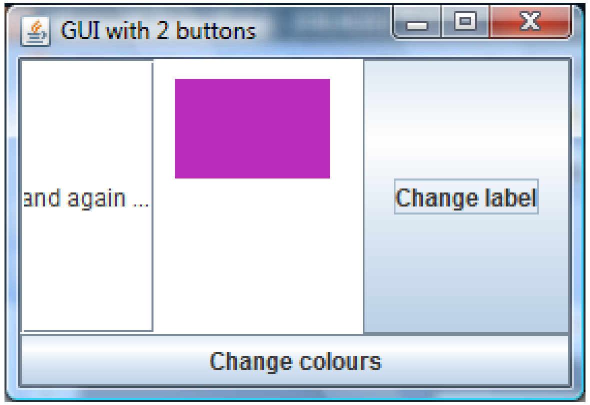
GUI

Note: The colours are *randomly generated*.

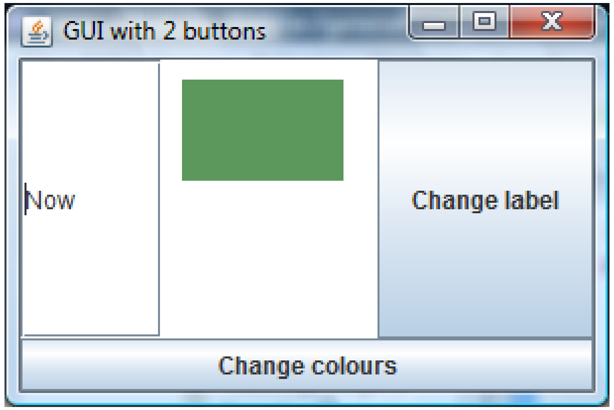
- Modify and/or add to the code below for a GUI Java program with *event handling and 2 buttons*. The expected output is:

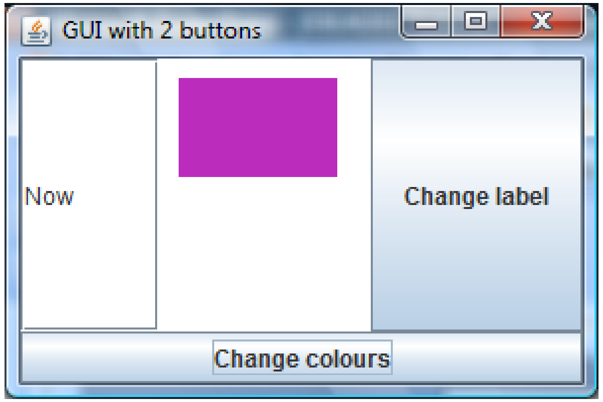


```
public class GUIwithTwoButtons {
 JFrame aFrame; JTextField aTxtField;
 public static void main(String[] args) {
   GUIwithTwoButtons myGui = new GUIwithTwoButtons();
   myGui.go();
 }
 public void go() {
   aFrame = new JFrame("GUI with 2 buttons");
   JButton labelButton = new JButton("Change label");
   JButton colorButton = new JButton("Change colours");
   aTxtField = new JTextField("Now", 6);
   MyDrawings drawing = new MyDrawings();
   aFrame.setSize(300, 200);
   aFrame.setVisible(true);
 }
}
```



The code above does not compile.





A possible solution … (1/3)

```
public class GUIwithTwoButtons {
 JFrame aFrame;
 JTextField aTxtField;
 public static void main(String[] args) {
   GUIwithTwoButtons myGui = new GUIwithTwoButtons();
   myGui.go();
 }
```

A possible solution … (2/3)

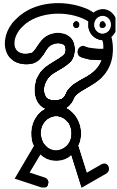
```
  public void go() {
     aFrame = new JFrame("GUI with 2 buttons");

     JButton labelButton = new JButton("Change label");

     JButton colorButton = new JButton("Change colours");

     aTxtField = new JTextField("Now", 6);
     MyDrawings drawing = new MyDrawings();





     aFrame.setSize(300, 200);
     aFrame.setVisible(true);
   }
 }
```

A possible solution … (3/3)

Exception Handling

- Consider a command-line calculator application. Write a program to deal with non-numeric operands, so it displays a message telling the user of the wrong operand type before exiting using an *exception handler*.

```
public class Calculator {
  public static void main(String[] args) {
    if (args.length != 3) {
      System.out.println(“Usage: java Calculator op1 operator op2”);
      System.exit(0);
    }
    int result = 0;
    switch (args[1].charAt(0)) {
      case ‘+’: result = Integer.parseInt(args[0]) +
                          Integer.parseInt(args[2]); break;
      case ‘-’: result = Integer.parseInt(args[0]) –
                          Integer.parseInt(args[2]); break;
      // Similarly for the ‘*’ and ‘/’ operations ...
    }
    System.out.println(args[0] + args[1] + args[2] + “ = ” + result);
  }
}
```



Can you think of other potential problems with this program?

A possible solution ...

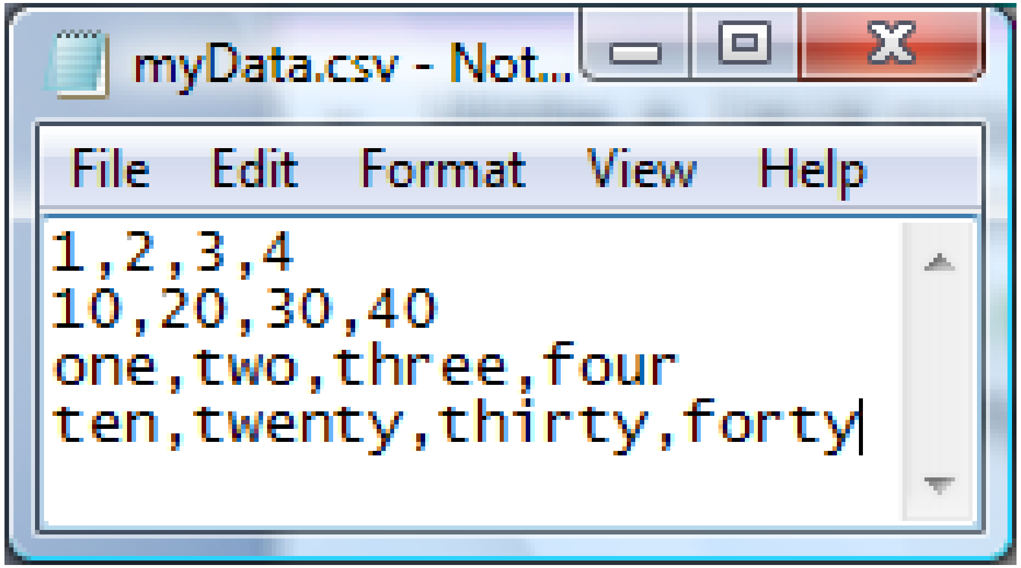
Could we be more specific when catching the exceptions?

```
public class BetterCalculator {
  public static void main(String[] args) {

    if (args.length != 3) {
      System.out.println(
         "Usage: java BetterCalculator operand1 operator
          operand2");
      System.exit(0);
    }



    switch (args[1].charAt(0)) { // Code as before ... }
    System.out.println(args[0] + args[1] + args[2] + " = " +
                       result);
  }
}
```

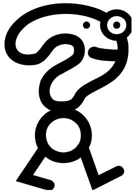
File I/O

- Write a Java program that reads each line of the data in a file called **myData.csv** and stores it in an array, after which it *displays the first element of the array onto the console* for each line of the file. Below is the expected output for running the program with the file shown:

```
> java FileProcessor
1st element of line 1 = 1
1st element of line 2 = 10
1st element of line 3 = one
1st element of line 4 = ten
```



```
1,2,3,4
10,20,30,40
one,two,three,four
ten,twenty,thirty,forty
```



What will happen if the file **myData.csv** does not exist?

A possible solution …