IS 413
Containers
JComponents

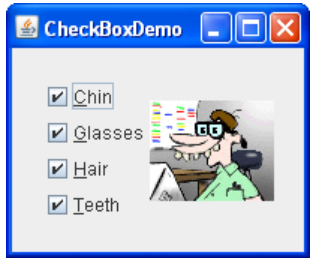
Benjamin Houdeshell
houdesh1@umbc.edu

Agenda

- GUI Components
- GUI Containers
  - JComponent
  - Command Line Arguments
  - Exceptions
  - Homework/Lab

GUI Components

- A *GUI component* is an object that represents a screen element such as a button or a text field
  - Java Components
- GUI-related classes are defined primarily in the `java.awt` and the `javax.swing` packages
  - Most components are subclasses of JComponent



- The *Abstract Windowing Toolkit* (AWT) was the original Java GUI package
- The *Swing* package provides additional and more versatile components
- Both packages are needed to create a Java GUI-based program

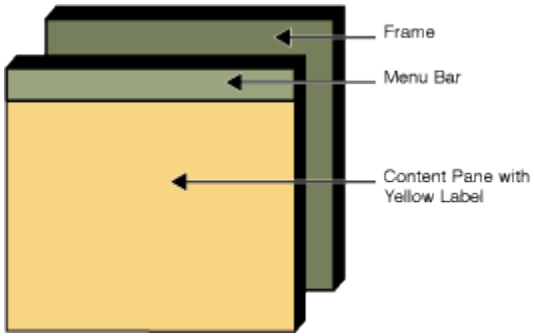
GUI Containers

- A *GUI container* is a component that is used to hold and organize other components
- A *frame* is a GUI container that is used to display a GUI-based Java application
  - javax.swing.JFrame

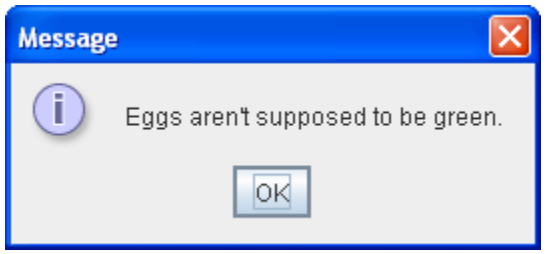
- A frame is displayed as a separate window with a title bar – it can be repositioned and resized on the screen as needed
- A *panel* is a GUI container that cannot be displayed on its own but is used to organize other components
  - javax.swing.JPanel
- A panel must be added to another container to be displayed

Top Level Containers

- Every GUI component must be part of a *containment hierarchy*
  - The root of a *containment hierarchy* is a top-level container
  - Each top-level container has a content pane that, contains (directly or indirectly) the visible components in that top-level container's GUI.



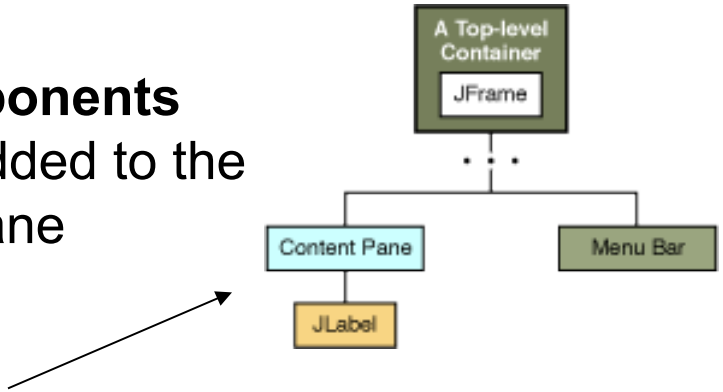
Top Level Containers



JDialog

- Top Level Containers
  - JFrame, JApplet, JDialog

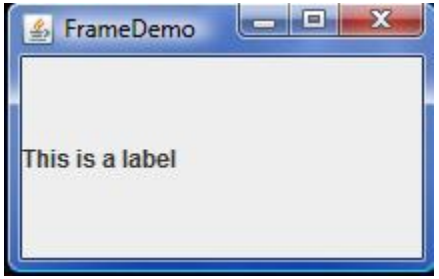
GUI Components
Must be added to the Content pane



Menu bar can be added Directly to the top-level container

```
frame.getContentPane().add(myLabel, BorderLayout.CENTER);
```

JFrame : Step by Step



- FrameDemo.java

```
//1. Create the frame.
JFrame frame = new JFrame("FrameDemo");

//2. Optional: What happens when the frame closes?
frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);

//3.  Create components
//...create  label and give it a size
JLabel myLabel = new JLabel("This is a label");
myLabel.setPreferredSize(new Dimension(200, 100));

//4. Add component to the content pane
frame.getContentPane().add(myLabel, BorderLayout.NORTH);

//5. Size the frame.
frame.pack();

//6. Show it.
frame.setVisible(true);
```

The JComponent class

- All Swing GUI components inherit from `javax.swing.JComponent`
  - Top level containers do not inherit from JComponent
- JComponent is an abstract class
- Rich functionality available to all GUI components
  - Tool tips
  - Painting and Borders
  - Layout
  - Pluggable Look and Feel
  - Custom Appearance
  - Event Handling
  - Others
  - See The JComponent Class

Exceptions (review)

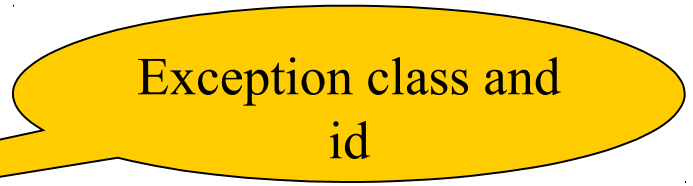
- An *exception* is an object that describes an unusual or non-standard situation
  - Potential to recover
- Exceptions are *thrown* by a program, and may be *caught* and *handled* by another part of the program
- A program can be separated into a normal execution flow and an *exception execution flow*
- Exceptions are represented as Objects in Java
  - See java.lang.Exception
- An *error* can also represented as an object in Java, but usually represents a unrecoverable situation

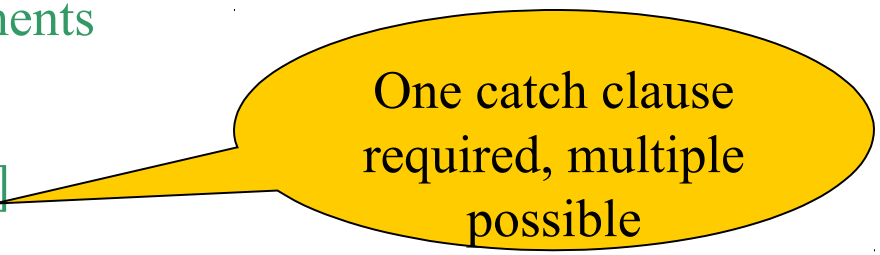
Exception Handling

- If an exception is ignored by the program, the program will terminate abnormally and produce an exception message
  - Unhandled Exception
- The message includes a *call stack trace* that:
  - indicates the line on which the exception occurred
  - shows the method call trail leading to the attempted execution of the offending line
- Throw
  - `public String convert(String s) throws Exception`
- Handling an Exception
  - try ….catch Block

Recap: Exception Form

```
try
{
    // Statements
}
catch  (ExceptionClassName exceptionIdentifier)
{
    // Exception handler statements
}
:   // [additional catch clauses]
[ finally
{
   // Statements
} ]
```





Command Line Arguments

- Provides the capability to pass parameters into a program
  - `public static void main(String[] args)`

    `args[]` is an array of strings

- The length of the array args depends on the number of arguments entered on the command line

J:\is413\week3_JFrame\code\hw_celsius_convert>java MyProgram a b c 1 2 3

```
args[0] = "a"    args[3] = "1"
args[1] = "b"    args[4] = "2"
args[2] = "c"    args[5] = "3"
```

Assignment

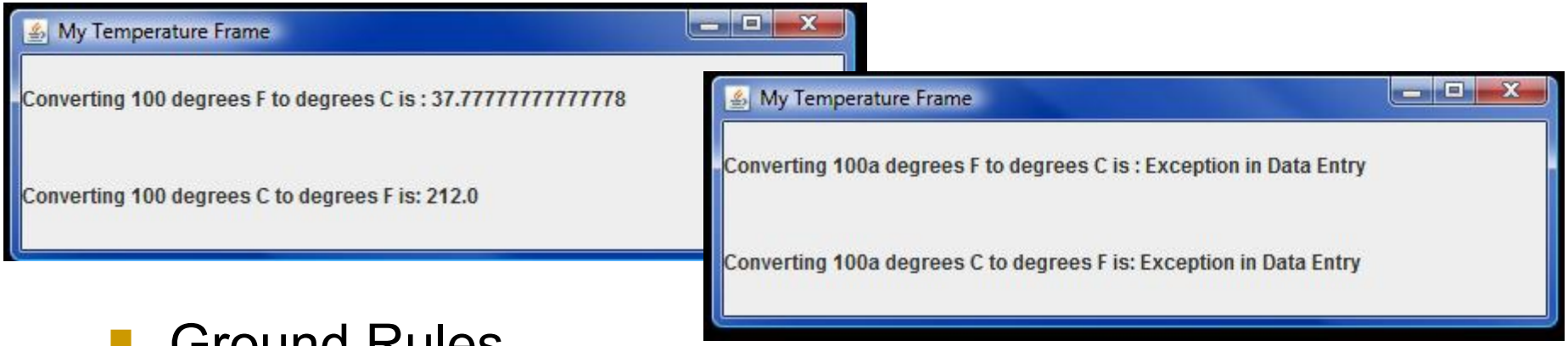
- Homework
  - Create three classes : TestTemp, Fahrenheit, Celsius
    - `public class Fahrenheit`
      - Contains a constructor that accepts a temperature
      - Contains a private member that represents a temperature
      - Contains a public method called `convert` that converts the temperature to celsius and returns this value
      - c = 5/9 * (f-32)
        - 212 degrees f
        - c = 5/9 (212 -32) = 100
    - `public class Celsius`
      - Does the same for a temperature in celsius
        - f = (9/5 * c) + 32;
          - 100 degrees c
          - f = (9/5 * 100) + 32 = 212

Assignment

- Homework (continued)
  - public class TestTemp
    - Creates an object of type Fahrenheit and an object of type Celsius
    - Accepts an single argument from the command line
      - Please do not use a Scanner or any other form of console I/O
    - Converts the command line argument from F to C and prints result
    - Converts the command line argument from C to F and prints result
  - Work with a partner
  - Use a text editor
  - **Compile and run at cmd line using javac**
  - Demo in class next week
  - **To turn in:** Please bring a hard copy of your source code. Please put your names in the commented source code
    - Emailed homework will not be accepted

HW Assignment Part II

- Create a JFrame
- Using the classes you created in Part I, create a Fahrenheit object and a Celsius object
- Use JLabel objects to display the command line argument and it's conversion from F to C and C to F
- Display an error message if invalid data was passed in on the command line (use Exceptions)



- Ground Rules
  - Demo in class next week
  - You may work with a partner if you like
  - Build and run the program from the command line
  - Turn in hard copy of your source code next week
    - Make sure your name and your partners name is on the listing

Reading

- Using Swing Components
  http://java.sun.com/docs/books/tutorial/uiswing/components/index.html
  - Read the subsections:
    - Using Top-Level Containers
    - The JComponent Class
  - How to Make Frames
    - http://download.oracle.com/javase/tutorial/uiswing/components/frame.html
  - Exceptions
    - http://java.sun.com/docs/books/tutorial/essential/exceptions/index.html
- For this weeks homework
  - Convert String to Double or Float
    - degreesDouble = Double.parseDouble(myString);
    - degreesFloat = Float.parseFloat(myString);

Summary

- Java GUI components are added to containers
  - Top Level Containers
    - JFrame, JDialog, JApplet
- All GUI components are part of a containment hierarchy
  - Add components to the *content pane*
    - Common Practice: Add components to a JPanel
    - Add JPanel to Content Pane
- Java GUI components are descendants of JComponent

- NOTE: Some of what we have looked at is handled automatically in IDE environments
  - Useful to know how things work, independent of the tools

Summary

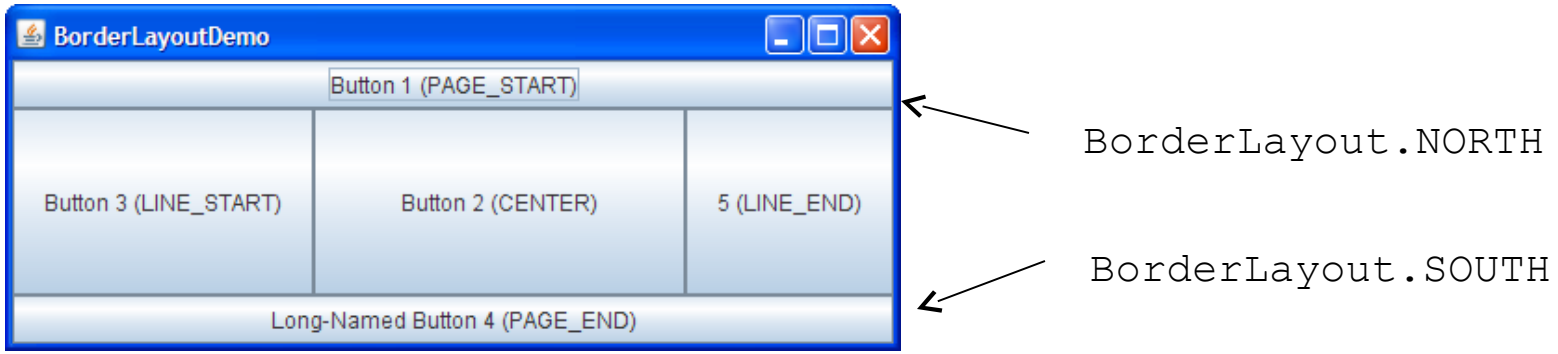
- Exceptions are a Nuts and Bolts part of the java language
  - Code that can cause an exception must be in a try …catch block
    - or **Throw** the Exception
    - To whom? – calling method
    - If an exception is never caught, the application terminates
- See the supplemental reading for more info and examples on exceptions
  - http://java.sun.com/docs/books/tutorial/essential/exceptions/index.html

Reference

- Reading
  - The Java Tutorial : Classes and Objects
    - Read the sections titled
      - Classes (and all subsections)
      - Objects (and all subsections)
  - The Java Tutorial: Packages
    - Read the sections titled:
      - Creating and Using Packages (and all subsections)
      - Creating a Package (and all subsections)

Reference BorderLayout

- Using Border Layout
  - http://java.sun.com/docs/books/tutorial/uiswing/layout/border.html



- BorderLayout is a Layout Manager
  - Positions components
    - `frame.getContentPane().add(myButton,BorderLayout.PAGE_START);`
    - NOTE: `BorderLayout.NORTH, BorderLayout.EAST, BorderLayout.WEST, BorderLayout.CENTER, BorderLayout.SOUTH` are equivalent.