**Week 3 – DQ1: Introduction**

Author: Daniel Schieberle  
Last Update: July 20th 2013  
Java Version: 1.6

**Files:**  
  
**HumanReadableComparison.java**    
Helper class providing a static method to compare two Double values and returns a human readable output.  
  
**JCustomInputPanel.java**    
Class provides a custom panel holding a label and a formatted text-field bound to a NumberFormatter.  
  
**JCustomPrecisionInputPanel.java**    
Class provides a custom panel holding a label and a slider component to set the display precision for the text-fields.  
  
**JCustomOutputPanel.java**    
Class provides a custom panel holding three labels that can be highlighted

**SchieberleWeek3DQ1.java**   
Class for DQ 1 “business logic”. Creates the custom GUI and provides handler methods to “react” to user button clicks, etc.

**SchieberleWeek3DQ1Driver.java**   
Wrapper Class. Main application entry point.

**Approach:**

Since as of this week it is “recommended” to familiarize with Swing and its components, I honestly “played” around to discover its possibilities.

SchieberleWeek3DQ1Driver.java

It was interesting to discover that Swing is not thread-safe and is depended from the Event Dispatcher Thread to deal with GUI updates and events. As a consequence the main method uses the “SwingUtilities” boilerplate to start the GUI provided in SchieberleWeek3DQ1.java

SchieberleWeek3DQ1.java

The class creates the custom GUI for the Java application allowing to input two double values and outputs the greater of both after comparison. Since I am not using JOptionPanes this time, input and output are handled on the same “frame” (Class extends JFrame). I opted to outsource the creation of the buttons into the private method “addButton” to automatically bind the current object as an “ActionListener” as well.  
For this week’s DQ1, I left input verification up to “NumberFormatter”. Since a “NumberFormatter” allows setting the precision of the values to be displayed, I decided to add a JSlider to allow adjusting the precision on the fly from 5 to 15.

HumanReadableComparison.java

The class provides a static method “onDoubles” taking advantage of Java’s compareTo method defined in the Double class. The method returns an array of strings where the greater value always “sits” in the first position as well as a human readable explanation of the comparison result. The case that both values entered are equal is covered as well.

JCustomInputPanel.java

Instead of repeating code during GUI setup, I decided to box the formatted input fields into a separate class that is instantiated twice during GUI setup. As already mentioned the text-field is backed-up by a “NumberFormatter” that conveniently also provides parsing of the input and in consequence provides access to the double primitive representation. The only exception to be caught at this point is about “null” meaning no input. In this case the corresponding label is highlighted in red to notify the user.  
The class has a static variable “precision” to allow changes across all instances consistently.

JCustomPrecisionInputPanel.java

To keep the approach consistent, I decided to “outsource” the slider as well. I might be able to reuse it in the future.

JCustomOutputPanel.java

The class provides a custom panel to display the String array provided by the “onDoubles” method in the HumanReadableCamparison. The numerically greater value is highlighted (darker color, larger font size)