|  |  |
| --- | --- |
| Data Validation Testing: Why It is Important for ETL  Data validation task | FeBRUARie 2024    Adriaan Mostert  Grade 12 |

Contents

[1. GUI Images 2](#_Toc158410144)

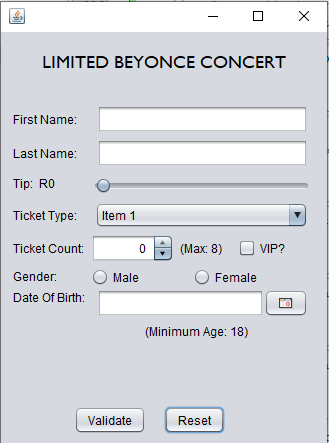
[2. Table: Description of fields 2](#_Toc158410145)

[3. Test Plan for validation 3](#_Toc158410146)

[4. Testing Evidenc 3](#_Toc158410147)

[5. Programming code 4](#_Toc158410148)

# GUI Before Data is Entered Images



# Table: Description of fields

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Input field** | **Input**  **Component** | **Reason For Input Component** | **Data Validation Method** | **Reason For Data Validation** |
| First Name (String) | JTextField | First Name Input is a string | * Presence Check * Length Check * Format Check | * Ensures data is inputted. * Ensures input does not exceed max length of 20. * Ensures no digits are used. |
| Last Name (String) | JTextField | Last Name input is a string | * Presence Check * Length Check * Format Check | * Ensures data is inputted. * Ensures input does not exceed max length of 20. * Ensures no digits are used. |
| Tip (Double) | JSlider | Tip is a real number | * None | * No validation needed. |
| Ticket Type (String) | JComboBox<String> | There are a fixed number of ticket types to choose from | * None | * No validation needed. |
| Ticket Count (Int) | JSpinner | Forces user to enter an integer | * Range Check | * Ensures tickets are within acceptable range of 1 and 8. |
| VIP (Boolean) | JCheckBox | Allows input of either one of two states | * None | * No validation needed. |
| Gender (Boolean) | JRadioButton x2  With ButtonGroup | Forces user to choose one of two options | * Presence Check | * Ensures data is inputted. |
| Date Of Birth (LocalDate) | JDateChooser | Forces user to choose a valid date | * Presence Check * Logic check | * Ensures data is inputted. * Ensures user is at least 18 years of age. |

# Test Plan for validation.

Each Field will be tested by violating the validation criteria. When the criteria are violated after pressing the validate button, a warning message will appear detailing the error for the input and the label of the incorrectly inputted field will be highlighted.

## Testing for Presence check

To test the presence check, the fields are left empty and when the data gets validated the criteria is violated. The warning message will notify the user which fields are empty, and the correct label will be highlighted.

The presence check will be tested with the First Name Field; Last Name Field; Gender Field and Date of Birth Field.

Note the Date of Birth Field would alert of violated criteria if the date entered is invalid.

## Testing for Length check

The length presence check is tested by exceeding the maximum character amount of 20 in the relevant fields. When the maximum characters are exceeded the warning message will be displayed.

The length check will be tested on the First Name Field and Last Name Field.

## Testing for Format check

The format check will be tested by entering digits in the fields that no digits are allowed. Therefore, the format is letters only. When digits are entered a warning message will be displayed.

This test applies to the First Name Field and the Last Name Field.

## Testing for Range check

The Range check will be tested by entering a value outside of the range of 1 to 8 like a 9. The ticket count is only allowed to be between 1 and 8. When it is outside of the range the criteria will be violated and a warning message will be displayed.

The test applies to the Ticket Count Field.

## Testing for Logic check

The Logic check will be tested by checking if the user is an adult, by checking their age. Therefore, the user must be eighteen or older. The Date of Birth Field will be used to calculate the user’s age. If the user is younger than eighteen the criteria is violated.

# Testing Evidence

## Testing for Presence check

**First Name Field left empty.**

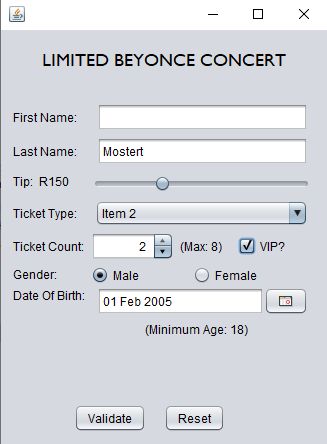
****

Figure 1: Before

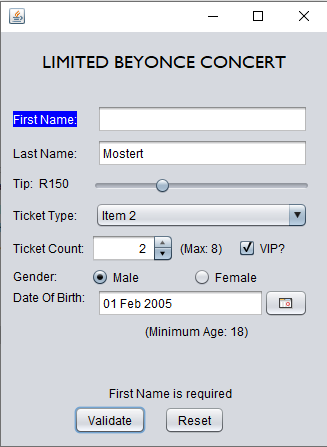
****

Figure 2: After

**Last Name Field left empty.**

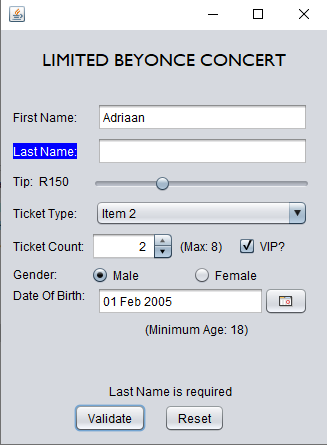
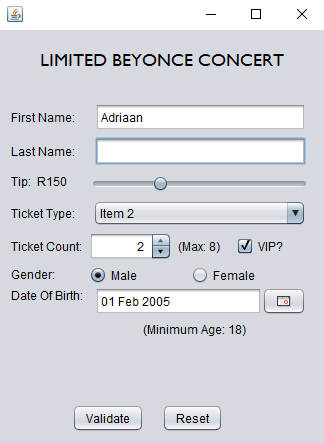
****

Figure 3: Before

Figure 4: After

**Gender Field left empty.**

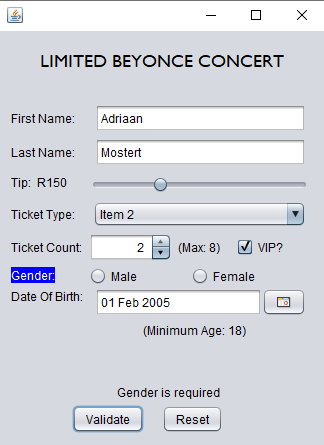
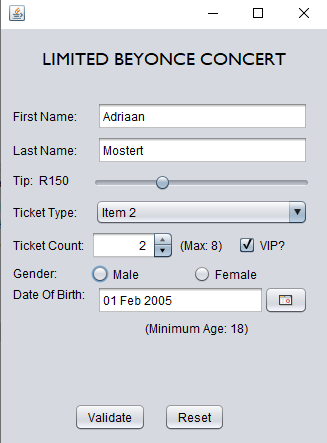
****

Figure 5: Before

Figure 6; After

**Date Of Birth Field left empty.**

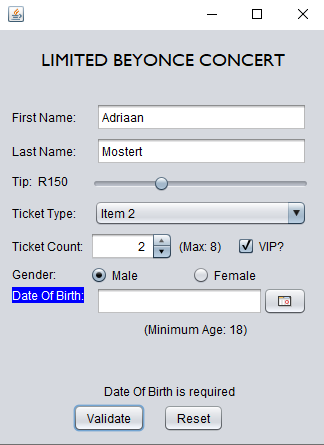
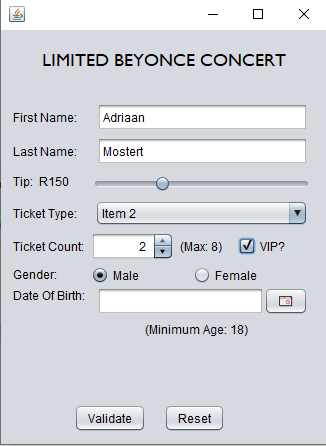
****

Figure 7: Before

Figure 8: After

## Testing for Length check

**First Name Field length check.**

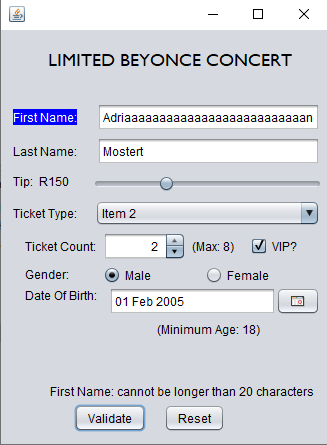
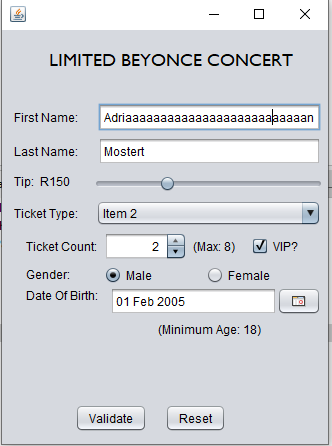
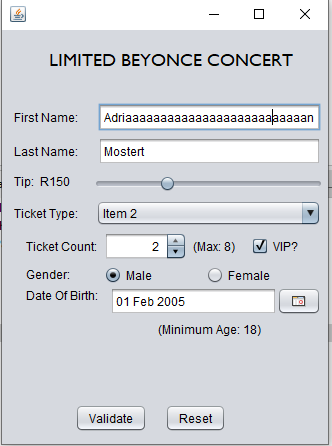
****

Figure 9: Before

Figure 10: After

**Last Name Field length check.**

**A screenshot of a concert

Description automatically generated**

Figure 11: Before

Figure 12: After

## Testing for Format check

**First Name Field format check for no digits.**

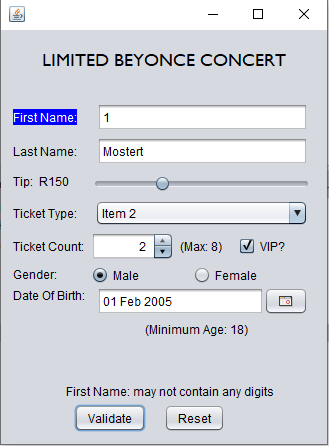
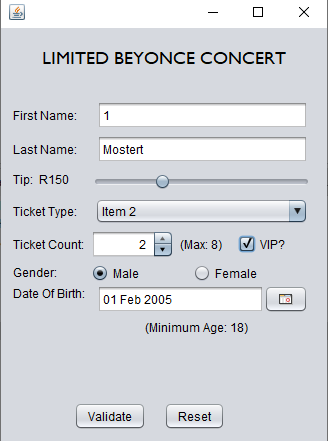
****

Figure 13: Before

Figure 14: After

**Last Name Field format check for no digits.**

**A screenshot of a concert

Description automatically generatedA screenshot of a concert

Description automatically generated**

Figure 15: Before

Figure 16: After

## Testing for Range check

**Ticket Count Field range check for between 1 and 8.**

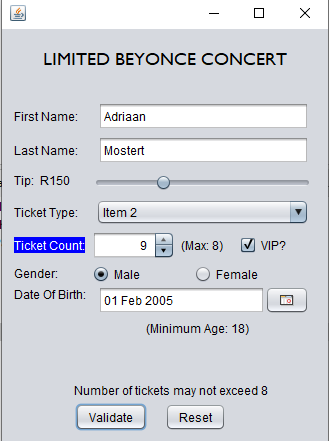
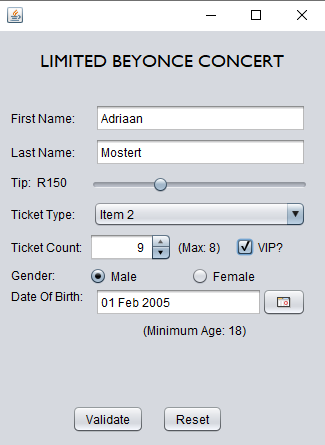
****

Figure 17: Before

Figure 18: After

## Testing for Logic check

**Date Of Birth Field Logic Check for Age 18 or over.**

**A screenshot of a concert

Description automatically generatedA screenshot of a concert

Description automatically generated**

Figure 19: Before

Figure 20: After

## When data is good

**Data is valid.**

**A screenshot of a concert

Description automatically generated**

# Programming code

/\*

\* Click nbfs://nbhost/SystemFileSystem/Templates/Licenses/license-default.txt to change this license

\* Click nbfs://nbhost/SystemFileSystem/Templates/GUIForms/JPanel.java to edit this template

\*/

package datavalidationtask.Views;

import java.awt.Color;

import java.time.LocalDate;

import java.time.ZoneId;

import javax.swing.JLabel;

import javax.swing.JTextField;

/\*\*

\*

\* @author user-pc

\*/

public class InputForm extends javax.swing.JPanel {

JLabel focusedLabel;

public InputForm() {

initComponents();

}

/\*\*

\* This method is called from within the constructor to initialize the form.

\* WARNING: Do NOT modify this code. The content of this method is always

\* regenerated by the Form Editor.

\*/

@SuppressWarnings("unchecked")

// <editor-fold defaultstate="collapsed" desc="Generated Code">

private void initComponents() {

genderBtnGroup = new javax.swing.ButtonGroup();

validateBtn = new javax.swing.JButton();

resetBtn = new javax.swing.JButton();

titleLabel = new javax.swing.JLabel();

infoPanel = new javax.swing.JPanel();

tipSlider = new javax.swing.JSlider();

birthDateLabel = new javax.swing.JLabel();

lastNameLabel = new javax.swing.JLabel();

firstNameField = new javax.swing.JTextField();

maleRadioBtn = new javax.swing.JRadioButton();

femaleRadioBtn = new javax.swing.JRadioButton();

ticketCounter = new javax.swing.JSpinner();

genderLabel = new javax.swing.JLabel();

typeLabel = new javax.swing.JLabel();

typeCombo = new javax.swing.JComboBox<>();

firstNameLabel = new javax.swing.JLabel();

vipCheckBox = new javax.swing.JCheckBox();

tipLabel = new javax.swing.JLabel();

countLabel = new javax.swing.JLabel();

tipCountLabel = new javax.swing.JLabel();

lastNameField = new javax.swing.JTextField();

maxTicketLabel = new javax.swing.JLabel();

birthDateChooser = new com.toedter.calendar.JDateChooser();

minAgeLabel = new javax.swing.JLabel();

warningLabel = new javax.swing.JLabel();

validateBtn.setText("Validate");

validateBtn.addActionListener(new java.awt.event.ActionListener() {

public void actionPerformed(java.awt.event.ActionEvent evt) {

validateBtnActionPerformed(evt);

}

});

resetBtn.setText("Reset");

resetBtn.addActionListener(new java.awt.event.ActionListener() {

public void actionPerformed(java.awt.event.ActionEvent evt) {

resetBtnActionPerformed(evt);

}

});

titleLabel.setFont(new java.awt.Font("Gill Sans MT", 0, 18)); // NOI18N

titleLabel.setHorizontalAlignment(javax.swing.SwingConstants.CENTER);

titleLabel.setText("LIMITED BEYONCE CONCERT");

tipSlider.setMaximum(500);

tipSlider.setValue(0);

tipSlider.addChangeListener(new javax.swing.event.ChangeListener() {

public void stateChanged(javax.swing.event.ChangeEvent evt) {

tipSliderStateChanged(evt);

}

});

birthDateLabel.setText("Date Of Birth:");

lastNameLabel.setText("Last Name:");

firstNameField.addActionListener(new java.awt.event.ActionListener() {

public void actionPerformed(java.awt.event.ActionEvent evt) {

firstNameFieldActionPerformed(evt);

}

});

maleRadioBtn.setText("Male");

femaleRadioBtn.setText("Female");

genderLabel.setText("Gender:");

typeLabel.setText("Ticket Type:");

typeCombo.setModel(new javax.swing.DefaultComboBoxModel<>(new String[] { "Item 1", "Item 2", "Item 3", "Item 4" }));

firstNameLabel.setText("First Name:");

vipCheckBox.setText("VIP?");

tipLabel.setText("Tip:");

countLabel.setText("Ticket Count:");

tipCountLabel.setText("R0");

lastNameField.addActionListener(new java.awt.event.ActionListener() {

public void actionPerformed(java.awt.event.ActionEvent evt) {

lastNameFieldActionPerformed(evt);

}

});

maxTicketLabel.setText("(Max: 8)");

minAgeLabel.setText("(Minimum Age: 18)");

javax.swing.GroupLayout infoPanelLayout = new javax.swing.GroupLayout(infoPanel);

infoPanel.setLayout(infoPanelLayout);

infoPanelLayout.setHorizontalGroup(

infoPanelLayout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addGroup(infoPanelLayout.createSequentialGroup()

.addContainerGap()

.addGroup(infoPanelLayout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addGroup(infoPanelLayout.createSequentialGroup()

.addGroup(infoPanelLayout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addGroup(infoPanelLayout.createSequentialGroup()

.addGroup(infoPanelLayout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addComponent(firstNameLabel)

.addComponent(lastNameLabel))

.addGap(20, 20, 20)

.addGroup(infoPanelLayout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addComponent(lastNameField)

.addComponent(firstNameField)))

.addGroup(infoPanelLayout.createSequentialGroup()

.addGroup(infoPanelLayout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addComponent(typeLabel)

.addGroup(infoPanelLayout.createSequentialGroup()

.addComponent(tipLabel)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)

.addComponent(tipCountLabel, javax.swing.GroupLayout.PREFERRED\_SIZE, 33, javax.swing.GroupLayout.PREFERRED\_SIZE)))

.addGap(18, 18, 18)

.addGroup(infoPanelLayout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addComponent(tipSlider, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, Short.MAX\_VALUE)

.addComponent(typeCombo, 0, javax.swing.GroupLayout.DEFAULT\_SIZE, Short.MAX\_VALUE))))

.addContainerGap())

.addGroup(infoPanelLayout.createSequentialGroup()

.addGap(0, 0, Short.MAX\_VALUE)

.addGroup(infoPanelLayout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addComponent(genderLabel)

.addComponent(birthDateLabel)

.addComponent(countLabel))

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)

.addGroup(infoPanelLayout.createParallelGroup(javax.swing.GroupLayout.Alignment.TRAILING)

.addGroup(infoPanelLayout.createSequentialGroup()

.addGroup(infoPanelLayout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addComponent(ticketCounter, javax.swing.GroupLayout.PREFERRED\_SIZE, 83, javax.swing.GroupLayout.PREFERRED\_SIZE)

.addComponent(maleRadioBtn, javax.swing.GroupLayout.PREFERRED\_SIZE, 64, javax.swing.GroupLayout.PREFERRED\_SIZE))

.addGroup(infoPanelLayout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING, false)

.addGroup(infoPanelLayout.createSequentialGroup()

.addGap(19, 19, 19)

.addComponent(femaleRadioBtn, javax.swing.GroupLayout.PREFERRED\_SIZE, 115, javax.swing.GroupLayout.PREFERRED\_SIZE)

.addContainerGap())

.addGroup(javax.swing.GroupLayout.Alignment.TRAILING, infoPanelLayout.createSequentialGroup()

.addGap(6, 6, 6)

.addComponent(maxTicketLabel)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED, javax.swing.GroupLayout.DEFAULT\_SIZE, Short.MAX\_VALUE)

.addComponent(vipCheckBox)

.addGap(29, 29, 29))))

.addGroup(infoPanelLayout.createSequentialGroup()

.addComponent(birthDateChooser, javax.swing.GroupLayout.PREFERRED\_SIZE, 211, javax.swing.GroupLayout.PREFERRED\_SIZE)

.addContainerGap())

.addGroup(infoPanelLayout.createSequentialGroup()

.addComponent(minAgeLabel, javax.swing.GroupLayout.PREFERRED\_SIZE, 105, javax.swing.GroupLayout.PREFERRED\_SIZE)

.addGap(64, 64, 64))))))

);

infoPanelLayout.setVerticalGroup(

infoPanelLayout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addGroup(infoPanelLayout.createSequentialGroup()

.addContainerGap()

.addGroup(infoPanelLayout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)

.addComponent(firstNameLabel)

.addComponent(firstNameField, javax.swing.GroupLayout.PREFERRED\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.PREFERRED\_SIZE))

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)

.addGroup(infoPanelLayout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)

.addComponent(lastNameLabel)

.addComponent(lastNameField, javax.swing.GroupLayout.PREFERRED\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.PREFERRED\_SIZE))

.addGap(8, 8, 8)

.addGroup(infoPanelLayout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)

.addComponent(tipLabel)

.addComponent(tipSlider, javax.swing.GroupLayout.PREFERRED\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.PREFERRED\_SIZE)

.addComponent(tipCountLabel))

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)

.addGroup(infoPanelLayout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)

.addComponent(typeLabel)

.addComponent(typeCombo, javax.swing.GroupLayout.PREFERRED\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.PREFERRED\_SIZE))

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)

.addGroup(infoPanelLayout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)

.addComponent(countLabel)

.addComponent(ticketCounter, javax.swing.GroupLayout.PREFERRED\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.PREFERRED\_SIZE)

.addComponent(vipCheckBox)

.addComponent(maxTicketLabel))

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)

.addGroup(infoPanelLayout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING, false)

.addComponent(genderLabel)

.addGroup(infoPanelLayout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)

.addComponent(femaleRadioBtn, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, Short.MAX\_VALUE)

.addComponent(maleRadioBtn)))

.addGap(3, 3, 3)

.addGroup(infoPanelLayout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addComponent(birthDateLabel)

.addComponent(birthDateChooser, javax.swing.GroupLayout.PREFERRED\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.PREFERRED\_SIZE))

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)

.addComponent(minAgeLabel)

.addContainerGap(39, Short.MAX\_VALUE))

);

warningLabel.setHorizontalAlignment(javax.swing.SwingConstants.CENTER);

warningLabel.setText("------------------------------------------------------------------------");

javax.swing.GroupLayout layout = new javax.swing.GroupLayout(this);

this.setLayout(layout);

layout.setHorizontalGroup(

layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addComponent(titleLabel, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, Short.MAX\_VALUE)

.addGroup(layout.createSequentialGroup()

.addGap(67, 67, 67)

.addComponent(validateBtn)

.addGap(18, 18, 18)

.addComponent(resetBtn)

.addContainerGap(javax.swing.GroupLayout.DEFAULT\_SIZE, Short.MAX\_VALUE))

.addGroup(layout.createSequentialGroup()

.addComponent(infoPanel, javax.swing.GroupLayout.PREFERRED\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.PREFERRED\_SIZE)

.addGap(0, 7, Short.MAX\_VALUE))

.addGroup(javax.swing.GroupLayout.Alignment.TRAILING, layout.createSequentialGroup()

.addContainerGap(javax.swing.GroupLayout.DEFAULT\_SIZE, Short.MAX\_VALUE)

.addComponent(warningLabel, javax.swing.GroupLayout.PREFERRED\_SIZE, 289, javax.swing.GroupLayout.PREFERRED\_SIZE)

.addContainerGap())

);

layout.setVerticalGroup(

layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addGroup(layout.createSequentialGroup()

.addContainerGap()

.addComponent(titleLabel, javax.swing.GroupLayout.PREFERRED\_SIZE, 37, javax.swing.GroupLayout.PREFERRED\_SIZE)

.addGap(18, 18, 18)

.addComponent(infoPanel, javax.swing.GroupLayout.PREFERRED\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.PREFERRED\_SIZE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED, 8, Short.MAX\_VALUE)

.addComponent(warningLabel, javax.swing.GroupLayout.PREFERRED\_SIZE, 14, javax.swing.GroupLayout.PREFERRED\_SIZE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)

.addComponent(validateBtn)

.addComponent(resetBtn))

.addContainerGap())

);

}// </editor-fold>

private void firstNameFieldActionPerformed(java.awt.event.ActionEvent evt) {

// TODO add your handling code here:

}

private void lastNameFieldActionPerformed(java.awt.event.ActionEvent evt) {

// TODO add your handling code here:

}

private void tipSliderStateChanged(javax.swing.event.ChangeEvent evt) {

tipLabel(tipSlider.getValue());

}

private void validateBtnActionPerformed(java.awt.event.ActionEvent evt) {

unhighlight();

validateData();

}

private void resetBtnActionPerformed(java.awt.event.ActionEvent evt) {

reset();

unhighlight();

}

//performs all the validation functions and returns when one of them fails

private void validateData() {

//executes check for data input

if (!checkIfHasData()) {

return;

}

//executes check for age

if (!checkAge()) {

warning("Minimum age of 18 is required...");

highlight(birthDateLabel);

return;

}

//executes check for ticket count

if (!checkTicketCount()) {

return;

}

//executes check of length for firstname

if (!checkLength(firstNameLabel, firstNameField, 20)) {

return;

}

//executes check of length for lastname

if (!checkLength(lastNameLabel, lastNameField, 20)) {

return;

}

//executes number check for firstname

if (!numberCheck(firstNameLabel, firstNameField)) {

return;

}

//executes number check for lastname

if (!numberCheck(lastNameLabel, lastNameField)) {

return;

}

//notifies user the input is valid

warning("Data is good");

}

//checks if all fields have data inputted

private boolean checkIfHasData() {

if (birthDateChooser.getDate() == null) {

warning("Date Of Birth is required");

highlight(birthDateLabel);

return false;

}

if (firstNameField.getText().trim().equals("")) {

warning("First Name is required");

highlight(firstNameLabel);

return false;

}

if (lastNameField.getText().trim().equals("")) {

warning("Last Name is required");

highlight(lastNameLabel);

return false;

}

if (!femaleRadioBtn.isSelected() && !maleRadioBtn.isSelected()) {

warning("Gender is required");

highlight(genderLabel);

return false;

}

return true;

}

//checks if user is at least 8 years old.

private boolean checkAge() {

LocalDate date = birthDateChooser.getDate().toInstant().atZone(ZoneId.systemDefault()).toLocalDate();

return (LocalDate.now().getYear() - date.getYear()) >= 18;

}

//ensures the ticket count is from 1 to 8

private boolean checkTicketCount() {

boolean valid = ((int) ticketCounter.getValue()) > 0;

if (!valid) {

warning("You must buy at least 1 ticket");

highlight(countLabel);

return valid;

}

valid = ((int) ticketCounter.getValue()) < 8;

if (!valid) {

warning("Number of tickets may not exceed 8");

highlight(countLabel);

}

return valid;

}

//check the length of the textfield input

private boolean checkLength(JLabel label, JTextField field, int length) {

if (field.getText().length() > length) {

warning(label.getText() + " cannot be longer than " + length + " characters");

highlight(label);

return false;

}

return true;

}

//ensures there are no numbers in names

private boolean numberCheck(JLabel label, JTextField field) {

String text = field.getText();

for (char c : text.toCharArray()) {

if (Character.isDigit(c)) {

warning(label.getText() + " may not contain any digits");

highlight(label);

return false;

}

}

return true;

}

//resets all the inputs of each field

private void reset() {

birthDateChooser.setDate(null);

genderBtnGroup.clearSelection();

firstNameField.setText("");

lastNameField.setText("");

ticketCounter.setValue(0);

typeCombo.setSelectedIndex(0);

tipSlider.setValue(0);

vipCheckBox.setSelected(false);

tipLabel(0);

warning("");

}

//highlight the labels where data is incorrectly entered

private void highlight(JLabel label) {

focusedLabel = label;

label.setBackground(Color.BLUE);

label.setOpaque(true);

label.setForeground(Color.WHITE);

label.repaint();

}

//unhighligts the highlighted labels when data is resetted

private void unhighlight() {

if (focusedLabel == null) {

return;

}

focusedLabel.setOpaque(false);

focusedLabel.setForeground(Color.BLACK);

focusedLabel.repaint();

focusedLabel = null;

}

//updates the value of the tip label

private void tipLabel(int num) {

tipCountLabel.setText("R" + num);

}

//updates the warning label to inform the user of incorrect input

private void warning(String warning) {

warningLabel.setText(warning);

}

// Variables declaration - do not modify

private com.toedter.calendar.JDateChooser birthDateChooser;

private javax.swing.JLabel birthDateLabel;

private javax.swing.JLabel countLabel;

private javax.swing.JRadioButton femaleRadioBtn;

private javax.swing.JTextField firstNameField;

private javax.swing.JLabel firstNameLabel;

private javax.swing.ButtonGroup genderBtnGroup;

private javax.swing.JLabel genderLabel;

private javax.swing.JPanel infoPanel;

private javax.swing.JTextField lastNameField;

private javax.swing.JLabel lastNameLabel;

private javax.swing.JRadioButton maleRadioBtn;

private javax.swing.JLabel maxTicketLabel;

private javax.swing.JLabel minAgeLabel;

private javax.swing.JButton resetBtn;

private javax.swing.JSpinner ticketCounter;

private javax.swing.JLabel tipCountLabel;

private javax.swing.JLabel tipLabel;

private javax.swing.JSlider tipSlider;

private javax.swing.JLabel titleLabel;

private javax.swing.JComboBox<String> typeCombo;

private javax.swing.JLabel typeLabel;

private javax.swing.JButton validateBtn;

private javax.swing.JCheckBox vipCheckBox;

private javax.swing.JLabel warningLabel;

// End of variables declaration

}