MailingLabels.java

```
import com.cete.dynamicpdf.*;
import com.cete.dynamicpdf.pageelements.TextArea;
import java.io.IOException;
import java.sql.Connection;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.sql.SQLException;
import javax.servlet.ServletConfig;
import javax.servlet.ServletException;
import javax.servlet.ServletOutputStream;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;


public class MailingLabels  extends HttpServlet {
    // Set page dimensions
    int topMargin;
    int bottomMargin;
    float rightMargin;
    float leftMargin;
    // Set the number of labels per page
    int maximumColumns;
    int maximumRows;
    // Set the spacing between the labels
    int horizontalSpace;
    int verticalSpace;
    // These margins are on the labels themsleves
    int labelTopBottomMargin;
    int labelLeftRightMargin;

    Document document;
    Page page;
    float currentColumn, currentRow, labelWidth, labelHeight;
    String companyName;
    String contactName;
    String phone;
    String fax;
    ServletOutputStream sOut;
    Connection connection;

    public void init(ServletConfig servletConfig) throws ServletException {
        super.init(servletConfig);
        topMargin = 36;
        bottomMargin = 36;
        rightMargin = 13.5f;
        leftMargin = 13.5f;
        maximumColumns = 3;
        maximumRows = 10;
```

```
        horizontalSpace = 9;
        verticalSpace = 0;
        labelTopBottomMargin = 5;
        labelLeftRightMargin = 15;
    }
    public void doGet(HttpServletRequest req, HttpServletResponse res)
    throws IOException,ServletException {
        CeTeConnection ceTe = (CeTeConnection)getServletContext().getAttribute("cetecon");
        connection = ceTe.getConnection();
        sOut = res.getOutputStream();

        // Create a document and set it's properties
        document = new Document();
        page = new Page(PageSize.LETTER, PageOrientation.PORTRAIT);
        document.setCreator("MailingLabels.java");
        document.setAuthor("ceTe Software");
        document.setTitle("Mailing Labels");

        // Entrypoint for the labels
        currentRow = 1;
        currentColumn = 1;

        ResultSet data = null;
        // Creates a ResultSet for the report
        try {
            PreparedStatement ps = connection.prepareStatement("Select CompanyName," +
                    "ContactName, Phone, Fax FROM Customers");
            data = ps.executeQuery();
        } catch (SQLException ex1) {
            ex1.printStackTrace(System.err);
        }

        // Loop over the ResultSet and add each label
        try {
            while (data.next()) {
                companyName = safeDBNull(data.getString("CompanyName"), data);
                contactName = safeDBNull(data.getString("ContactName"), data);
                phone = safeDBNull(data.getString("Phone"), data);
                fax = safeDBNull(data.getString("Fax"), data);
                addLabel();
            }
        } catch (SQLException ex2) {
            ex2.printStackTrace(System.err);
        }
        if (page.getElements().size() > 0) {
            document.getPages().add(page);
        }
        // Outputs the MailingLabels to the current web page
        document.drawToWeb(req, res, sOut, "MailingLabels.pdf");
```

```
99          ceTe.close();
100         sOut.close();
101     }
102
103     private float findLabelHeight() {
104         return (page.getDimensions().getHeight() - (page.getDimensions().getTopMargin()
105         + page.getDimensions().getBottomMargin()) - ((maximumRows - 1)
106         * verticalSpace)) / maximumRows;
107     }
108
109     private float findLabelWidth() {
110         return (page.getDimensions().getWidth() - (page.getDimensions().getRightMargin()
111         + page.getDimensions().getLeftMargin()) - ((maximumColumns - 1)
112         * horizontalSpace)) / maximumColumns;
113     }
114
115     private void addLabel() {
116         // Add a new page if you are beyond the maximum Rows
117         if (currentRow == maximumRows + 1) {
118             document.getPages().add(page);
119             currentRow = 1;
120         }
121         // Determines if the the label belongs in the first row or first column of the page
122         if (currentColumn > 1 & currentRow > 1) {
123             addToPage();
124         } else if (currentColumn > 1 & currentRow == 1)    {
125             addToFirstRow();
126         } else if (currentColumn == 1 & currentRow > 1)    {
127             addToFirstColumn();
128         } else    {
129             page = new Page(PageSize.LETTER, PageOrientation.PORTRAIT);
130
131             page.getDimensions().setTopMargin(topMargin);
132             page.getDimensions().setBottomMargin(bottomMargin);
133             page.getDimensions().setRightMargin(rightMargin);
134             page.getDimensions().setLeftMargin(leftMargin);
135             labelWidth = findLabelWidth();
136             labelHeight = findLabelHeight();
137             addToFirstRowColumn();
138         }
139
140         // Incremment your row if you are beyond the maximum columns
141         if (currentColumn == maximumColumns + 1) {
142             currentRow = currentRow + 1;
143             currentColumn = 1;
144         }
145     }
146
147     // Adds the label on at least row 2 column 2 of the page
```

```
148     private void addToPage() {
149         float x;
150         float y;
151         y = (currentRow - 1) * (labelHeight + verticalSpace);
152         x = (currentColumn - 1) * (labelWidth + horizontalSpace);
153         addLabelInfo(x, y);
154         currentColumn = currentColumn + 1;
155     }
156 
157     // Adds the label on the first row of labels
158     private void addToFirstRow() {
159         float x;
160         float y;
161         y = 0;
162         x = (currentColumn - 1) * (labelWidth + horizontalSpace);
163         addLabelInfo(x, y);
164         currentColumn = currentColumn + 1;
165     }
166 
167     // Adds the label to the First column of labels
168     private void addToFirstColumn() {
169         float x;
170         float y;
171         y = (currentRow - 1) * (labelHeight + verticalSpace);
172         x = 0;
173         addLabelInfo(x, y);
174         currentColumn = currentColumn + 1;
175     }
176 
177     // Adds only the first label of every page (row 1 column 1)
178     private void addToFirstRowColumn() {
179         float x;
180         float y;
181         y = 0;
182         x = 0;
183         addLabelInfo(x, y);
184         currentColumn = currentColumn + 1;
185     }
186 
187     // This is where you format the look of each label
188     private void addLabelInfo(float x, float y)    {
189         TextArea txt1 = new TextArea(companyName, x + labelLeftRightMargin,
190                 y + labelTopBottomMargin, labelWidth
191                 - (labelLeftRightMargin * 2), 11,
192                 Font.getTimesRoman(), 11);
193         TextArea txt2 = new TextArea(contactName, x + labelLeftRightMargin,
194                 y + labelTopBottomMargin + 12, labelWidth
195                 - (labelLeftRightMargin * 2), 11,
196                 Font.getTimesRoman(), 11);
```

```
197         TextArea txt3 = new TextArea(phone, x + labelLeftRightMargin,
198                 y + labelTopBottomMargin + 24, labelWidth
199                 - (labelLeftRightMargin * 2), 11,
200                 Font.getTimesRoman(), 11);
201         TextArea txt4 = new TextArea(fax, x + labelLeftRightMargin,
202                 y + labelTopBottomMargin + 36, labelWidth
203                 - (labelLeftRightMargin * 2), 11,
204                 Font.getTimesRoman(), 11);
205 
206         page.getElements().add(txt1);
207         page.getElements().add(txt2);
208         page.getElements().add(txt3);
209         page.getElements().add(txt4);
210     }
211 
212     private String safeDBNull(String value, ResultSet data) {
213         try {
214             if (data.wasNull()) {
215                 return "";
216             } else {
217                 return value;
218             }
219         } catch (SQLException ex3) {
220             ex3.printStackTrace(System.err);
221         }
222         return "";
223     }
224 }
```