

TimeMachine.java

```
1 import com.cete.dynamicpdf.*;
2 import com.cete.dynamicpdf.pageelements.*;
3 import com.cete.dynamicpdf.pageelements.barcoding.Ean13Sup5;
4 import java.io.BufferedReader;
5 import java.io.FileNotFoundException;
6 import java.io.FileReader;
7 import java.io.IOException;
8 import java.text.SimpleDateFormat;
9 import java.util.Date;
10 import java.util.GregorianCalendar;
11 import javax.servlet.ServletConfig;
12 import javax.servlet.ServletException;
13 import javax.servlet.ServletOutputStream;
14 import javax.servlet.http.HttpServlet;
15 import javax.servlet.http.HttpServletRequest;
16 import javax.servlet.http.HttpServletResponse;
17
18 public class TimeMachine extends HttpServlet {
19
20     ServletOutputStream sOut;
21     Template footerTemplate;
22     EvenOddTemplate headerTemplate;
23
24     public void init(ServletConfig servletConfig) throws ServletException {
25         super.init(servletConfig);
26
27     }
28     public void doGet(HttpServletRequest req, HttpServletResponse res)
29             throws IOException, ServletException {
30
31
32         sOut = res.getOutputStream();
33         footerTemplate = new Template();
34         headerTemplate = new EvenOddTemplate();
35         // Create a document and set it's properties
36         Document objDocument = new Document();
37         objDocument.setCreator("TimeMachine.java");
38         objDocument.setAuthor("H. G. Wells");
39         objDocument.setTitle("The Time Machine");
40         objDocument.setTemplate(headerTemplate);
41
42         // Adds elements to the header and footer groups
43         setPageHeaderTemplate();
44         setPageFooterTemplate();
45
46         // Sets up outline hierarchy
47         Outline objTitlePageOutline = objDocument.getOutlines().add("The Time "+
48             "Machine", new XYDestination(1, 0, 0));
49
```

TimeMachine.java

```
50    Outline objChaptersOutline = objTitlePageOutline.getChildOutlines()
51                                .add("Chapters");
52
53    // Builds the report
54    buildDocument(objDocument, objTitlePageOutline, objChaptersOutline);
55
56    // Outputs the document to the current web page
57    objDocument.drawToWeb(req, res, sOut, "TimeMachine.pdf");
58    sOut.close();
59 }
60
61 private void buildDocument(Document document, Outline titlePage,
62                           Outline chapters) {
63    // Adds Title page to document
64    document.getSections().begin(NumberingStyle.NONE, "Cover");
65    addTitlePage(document);
66    document.getSections().begin(NumberingStyle.NUMERIC, "Page",
67                                 footerTemplate);
68
69    // Adds Chapters to the document
70    addChapter(document, getServletContext().getRealPath("/text/TimeMachine"+
71               "_Chapter1.txt"), "1", "Chapter 1", chapters);
72    addChapter(document, getServletContext().getRealPath("/text/TimeMachine"+
73               "_Chapter2.txt"), "2", "Chapter 2", chapters);
74    addChapter(document, getServletContext().getRealPath("/text/TimeMachine"+
75               "_Chapter3.txt"), "3", "Chapter 3", chapters);
76    addChapter(document, getServletContext().getRealPath("/text/TimeMachine"+
77               "_Chapter4.txt"), "4", "Chapter 4", chapters);
78    addChapter(document, getServletContext().getRealPath("/text/TimeMachine"+
79               "_Chapter5.txt"), "5", "Chapter 5", chapters);
80    addChapter(document, getServletContext().getRealPath("/text/TimeMachine"+
81               "_Chapter6.txt"), "6", "Chapter 6", chapters);
82    addChapter(document, getServletContext().getRealPath("/text/TimeMachine"+
83               "_Chapter7.txt"), "7", "Chapter 7", chapters);
84    addChapter(document, getServletContext().getRealPath("/text/TimeMachine"+
85               "_Chapter8.txt"), "8", "Chapter 8", chapters);
86    addChapter(document, getServletContext().getRealPath("/text/TimeMachine"+
87               "_Chapter9.txt"), "9", "Chapter 9", chapters);
88    addChapter(document, getServletContext().getRealPath("/text/TimeMachine"+
89               "_Chapter10.txt"), "10", "Chapter 10", chapters);
90    addChapter(document, getServletContext().getRealPath("/text/TimeMachine"+
91               "_Chapter11.txt"), "11", "Chapter 11", chapters);
92    addChapter(document, getServletContext().getRealPath("/text/TimeMachine"+
93               "_Chapter12.txt"), "12", "Chapter 12", chapters);
94
95    // Add the Epilogue to the document
96    addChapter(document, getServletContext().getRealPath("/text/TimeMachine"+
97               "_Epilogue.txt"), "Epilogue", "Epilogue", titlePage);
98 }
```

TimeMachine.java

```
99
100    private void setPageHeaderTemplate() {
101
102        // Adds elements to the header group
103        headerTemplate.getOddElements().add(new Label("The Time Machine",
104            -18, -18, 324, 11, Font.getTimesRoman(), 11, TextAlign.LEFT));
105        headerTemplate.getOddElements().add(new Label("H. G. Wells", 18, -18,
106            324, 11, Font.getTimesRoman(), 11, TextAlign.RIGHT));
107        headerTemplate.getEvenElements().add(new Label("H. G. Wells", -18, -18,
108            324, 11, Font.getTimesRoman(), 11, TextAlign.LEFT));
109        headerTemplate.getEvenElements().add(new Label("The Time Machine", 18,
110            -18, 324, 11, Font.getTimesRoman(), 11, TextAlign.RIGHT));
111    }
112
113    private void setPageFooterTemplate() {
114        // Adds elements to the footer group
115        PageNumberingLabel objPageNumberLabel = new PageNumberingLabel
116            ("-%>SP%<-", 0, 478, 324,
117            11, Font.getTimesRoman(),
118            11, TextAlign.CENTER);
119        objPageNumberLabel.setPageOffset(-1);
120        footerTemplate.getElements().add(objPageNumberLabel);
121    }
122
123    private void addTitlePage(Document document) {
124        // Adds a title page to the document
125        Page objPage = new Page(396, 540, 36);
126
127        String strDisclaimer = "This document is in the public domain. " +
128            "Permission to use, copy, modify, and " +
129            "distribute this document for any purpose and " +
130            "without fee is hereby granted, without any " +
131            "conditions or restrictions. The barcode below " +
132            "is for demonstration purposes only.";
133
134        GregorianCalendar gc = new GregorianCalendar();
135        Date date = gc.getTime();
136        SimpleDateFormat sd1 = new SimpleDateFormat("yyyy.MM.dd");
137        SimpleDateFormat sd2 = new SimpleDateFormat("HH.mm.ss");
138        String strGenerated = "Generated by\nDynamicPDF Generator\non "
139            + sd1.format(date) + ",\nat " + sd2.format(date)
140            + " EST";
141        objPage.getElements().add(new Label("The Time Machine", 36, 36, 252, 30,
142            Font.getTimesBold(), 30, TextAlign.CENTER));
143        objPage.getElements().add(new Label("by H. G. Wells", 36, 96, 252, 22,
144            Font.getTimesBold(), 22, TextAlign.CENTER));
145        objPage.getElements().add(new Label("1895", 36, 148, 252, 22,
146            Font.getTimesBold(), 22, TextAlign.CENTER));
147        try {
```

TimeMachine.java

```
148         objPage.getElements().add(new Image(getServletContext().getRealPath
149                             (" /images/DPDFLogo.png"), 62, 208, 0.21f));
150     } catch (FileNotFoundException ex) {
151         System.err.println("cannot load the image :" + ex);
152     }
153     objPage.getElements().add(new Label(strGenerated, 132, 208, 182, 54,
154                                     Font.getTimesRoman(), 11));
155     objPage.getElements().add(new Label(strDisclaimer, 36, 276, 252, 65,
156                                     Font.getTimesRoman(), 11, TextAlign.JUSTIFY));
157     objPage.getElements().add(new Ean13Sup5("201234567890", "90000", 82,
158                                         360));
159     objPage.setApplyDocumentTemplate(false);
160
161     document.getPages().add(objPage);
162 }
163
164 private void addChapter(Document document, String filePath, String title,
165                         String bookmarkText, Outline parentOutline) {
166     // Retrieves the text from the sections file
167     String strSectionText = getTextFromFile(filePath);
168
169     // Adds the first page of the section
170     Page objPage = addSectionHeaderPage(document, title, bookmarkText,
171                                         parentOutline);
172     objPage.setApplyDocumentTemplate(false);
173
174     // Creates a TextArea for the sections text
175     TextArea objTextArea = new TextArea(strSectionText, 0, 146, 324, 322,
176                                         Font.getTimesRoman(), 11);
177     objTextArea.setLeading(14);
178     objTextArea.setParagraphSpacing(20);
179     objPage.getElements().add(objTextArea);
180     document.getPages().add(objPage);
181
182     // Creates a TextArea for the overflow text
183     objTextArea = objTextArea.getOverflowTextArea(0, 0, 324, 468);
184
185     // Loops until no overflow text is found.
186     while (objTextArea != null) {
187         // Adds a new page to the document
188         objPage = new Page(396, 540, 36);
189
190         objPage.getElements().add(objTextArea);
191         // Adds new page to the document
192         document.getPages().add(objPage);
193         // Creates a TextArea for the overflow text
194         objTextArea = objTextArea.getOverflowTextArea();
195     }
196 }
```

TimeMachine.java

```
197
198     private Page addSectionHeaderPage(Document document, String title,
199                                         String bookmarkText,
200                                         Outline parentOutline) {
201         // Adds the first page of a section to the document
202         Page objPage = new Page(396, 540, 36);
203
204
205
206         objPage.getElements().add(new Bookmark(bookmarkText, 0, 0,
207                                               parentOutline));
208         objPage.getElements().add(new Label("The Time Machine", 0, 36, 324, 30,
209                                         Font.getTimesBold(), 30,
210                                         TextAlign.CENTER));
211         objPage.getElements().add(new Label(title, 0, 96, 324, 22,
212                                         Font.getTimesBold(), 22,
213                                         TextAlign.CENTER));
214         objPage.getElements().add(new Line(120, 128, 204, 128));
215
216         return objPage;
217     }
218
219     private String getTextFromFile(String filePath) {
220         // Opens a text file and returns the text from it.
221         StringBuffer contents = new StringBuffer();
222
223         BufferedReader reader = null;
224         try {
225             reader = new BufferedReader(new FileReader(filePath));
226             String line = null;
227             while ((line = reader.readLine()) != null){
228                 contents.append(line);
229                 contents.append(System.getProperty("line.separator"));
230             }
231         } catch (FileNotFoundException ex1) {
232             System.err.println("Invalid file path :" +ex1);
233         } catch (IOException ex2){
234             System.err.println("cannot read from the file :" +ex2);
235         } finally {
236             try {
237                 reader.close();
238             } catch (IOException ex3) {
239                 System.err.println("cannot close the file :" +ex3);
240             }
241         }
242         return contents.toString();
243     }
244
245 }
```

TimeMachine.java

```
246  
247 }
```