Java Techniki Programowania

Zestaw 6.

1 KantorExport

Do pierwszego zadania *Kantor* z zestawu 3. dodać opcję (i odpowiedni przycisk) umożliwiający zapis tabeli zawierającej kursy walut (z bazy danych *MySQL*) w pliku w jednym z poniższych formatów. Tabela powinna mieć odpowiedni nagłówek, obramowanie komórek oraz zróżnicowany kolor tekstu/tła odpowiednich kolumn. Wystarczy aby program obsługiwał tylko jeden z poniższych formatów (**do wyboru**).

1.1 Microsoft Excel .xls, Apache POI HSSF

Aby móc używać bibliotek z projektu *Apache POI* należy ściągnąć ze strony z zadaniami i dołączyć archiwum `poi-3.2-FINAL-20081019.jar`. Obsługa formatu *Microsoft Excel .xls* znajduje się w pakiecie `org.apache.poi.hssf.usermodel` (komponent HSSF (Horrible SpreadSheet Format), dawniej wchodząca w skład projektu *Jakarta*).

Należy zaimportować odpowiednie klasy z w.w. pakietu.
Opis użycia pakietu znajduje się w skryptach z wykładu. Poniżej znajduje się przykładowy kod tworzący dokument, arkusz oraz przykładową komórkę:

```
FileOutputStream fo  = new FileOutputStream("output.xls");
HSSFWorkbook wb  = new HSSFWorkbook();
HSSFSheet sh  = wb.createSheet("Tablica");
HSSFRow row = sh.createRow(0);
HSSFCell c   = row.createCell(1);
HSSFCellStyle cs = wb.createCellStyle();

c.setCellValue("Test");
cs.setFillForegroundColor(HSSFColor.GREY_25_PERCENT.index);
cs.setFillPattern(HSSFCellStyle.SOLID_FOREGROUND);
cs.setBorderLeft(HSSFCellStyle.BORDER_THICK);
cs.setBorderTop(HSSFCellStyle.BORDER_THICK);
cs.setBorderRight(HSSFCellStyle.BORDER_THIN);
cs.setBorderBottom(HSSFCellStyle.BORDER_THIN);
c.setCellStyle(cs);
```

1.2 Microsoft Word 97 .doc, Apache POI HPSF

Kolejnym formatem obsługiwanym przez *Apache POI* jest *Microsoft Word 97*. Należy ściągnąć i dołączyć archiwum `poi-scratchpad-3.2-FINAL-20081019.jar`
oraz zaimportować pakiet `import org.apache.poi.hwpf` (Horrible Word Processor Format, komponent jest we wstępnej fazie). Aby utworzyć tabele, można użyć klasy
`org.apache.poi.hwpf.usermodel.Table`. Ponadto wymagany jest już istniejący (może być pusty) dokument `.doc`.
Przykład użycia pakietu:

```
FileOutputStream fo    = new FileOutputStream("empty.doc");
HWPFDocument     doc   = new HWPFDocument();
CharacterRun     run   = new CharacterRun();
Range            range = doc.getRange();
```

```
run.setBold(true);
run.setItalic(true);
run.setCapitalized(true);
range.insertBefore("Test", run);
doc.write(fo);
fo.close();
```

1.3 Portable Document Format .pdf, JasperReports

Narzędziem pozwalającym tworzyć raporty bezpośrednio na podstawie informacji zawartych w bazie danych *SQL* jest JasperReports. Pozwala ono na zapis do plików m.in. w formatach *PDF, HTML, Excel, RTF, CSV*. W tym celu należy dołączyć archiwum `jasperreports-3.5.0.jar` (JasperReports API),
a także `commons-digester-2.0.jar` (Klasa do przetwarzania dokumentów XML),
`commons-logging-1.1.1.jar` (Logging classes),
`commons-collections-3.2.1.jar` (Collections framework extension classes),
`commons-beanutils-1.8.0.jar` (JavaBeans utility classes),
`iText-2.1.5.jar` (Biblioteka do obsługi PDF).
Wszystkie powyższe archiwa są umieszczone na stronie ćwiczeń. Należy także utworzyć plik w formacie XML zawierający instrukcje dla pakietu *JasperReports*. Przykładowy plik `Kantor.jrxml` jest umieszczony na stronie ćwiczeń. Pakiet umożliwia automatyczne pobieranie danych z bazy, wystarczy przekazać odpowiedni obiekt klasy `Connection`. Na podstawie pliku `.jrxml` biblioteka odpowiednio przetwarza informacje z bazy danych. Należy zaimportować pakiet `net.sf.jasperreports.engine` oraz obsłużyć odpowiednie wyjątki. Opis użycia pakietu znajduje się w skryptach z wykładu. Przykład tworzący raport do pliku *.pdf* i *.html*:

```
Connection con = DriverManager.getConnection
  ("jdbc:mysql://" + host + "/" + dataBase, userName, password);
Map        map = new HashMap();
JasperReport jr = JasperCompileManager.compileReport("Kantor.jrxml");
JasperPrint jp = JasperFillManager.fillReport(jr, map, con);
JasperExportManager.exportReportToPdfFile(jp, "Kantor.pdf");
JasperExportManager.exportReportToHtmlFile(jp, "Kantor.html");
```

1.4 Excel .XLS, JasperReports

Aby móc wygenerować raport i zapisać go do pliku w formacie `.xls` należy dołączyć bibliotekę *Apache POI HSSF*. Oprócz tego należy użyć klasy `JRXlsExporter` z pakietu `net.sf.jasperreports.engine.export`:

```
OutputStream ouputStream = new FileOutputStream(new File("Kantor.xls"));
ByteArrayOutputStream byteArrayOutputStream = new ByteArrayOutputStream();
JRXlsExporter exporterXLS = new JRXlsExporter();

exporterXLS.setParameter(JRXlsExporterParameter.JASPER_PRINT, jp);
exporterXLS.setParameter(JRXlsExporterParameter.OUTPUT_STREAM, byteArrayOutputStream);
exporterXLS.exportReport();
ouputStream.write(byteArrayOutputStream.toByteArray());
ouputStream.flush();
ouputStream.close();
```

1.5 PDF, iText

Aby utworzyć dokument w formacie `.pdf` (Portable Document Format) można użyć biblioteki *iText* (do ściągnięcia ze strony ćwiczeń). Należy dołączyć pakiety `com.lowagie.text` oraz `com.lowagie.text.pdf`. Należy obsłużyć odpowiednie wyjątki. Opis użycia pakietu znajduje się w skryptach z wykładu. Poniżej znajduje się prosty przykład tworzący plik `.pdf` zawierający tabele:

```
Document doc = new Document();
PdfWriter.getInstance(doc, new FileOutputStream("output.pdf"));
PdfPTable table = new PdfPTable(2);
PdfPCell cell = new PdfPCell(new Paragraph("Tabela"));

doc.open();
cell.setColspan(2);
table.addCell(cell);
table.addCell("Raz");
table.addCell("Dwa");
doc.add(table);
doc.close();
```

Podobnie można tworzyć dokumenty w formacie `.html` i `.rtf`.

Andrzej Görlich
atg@th.if.uj.edu.pl