Embedding JasperReports into JBoss Applications

Teodor Danciu
Founder
JasperReports

February 22, 2005

Agenda

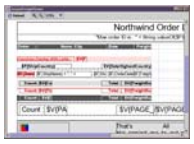
- JasperReports Product Description
- JasperReports for Jboss
- Sample JasperReports/JBoss Integration

JasperReports

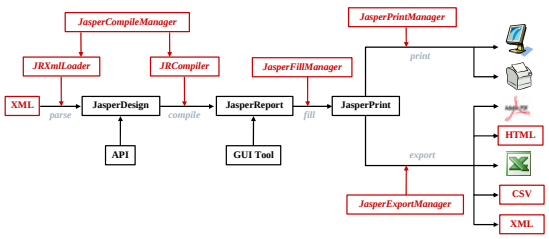
- ✓ Rich information presentation
- ✓ Open-source Java-based library
- ✓ Seamlessly-embeddable
- ✓ Runtime environment agnostic
- ✓ Data source agnostic
- ✓ Flexible calculation capabilities
- ✓ Pixel-perfect, complex layouts
- ✓ Wide range of output channels

Report Creation Process

- Report Design
  - ✓ JRXML – XML-based report definition language
  - ✓ Alternative methods for JRXML editing
    - Swing and Eclipse based visual report designers
    - Programmatic API
    - XML Editors
- Report compilation
  - ✓ Pluggable report compilers
  - ✓ Compiles design to ready-to-use form
- Report filling
  - ✓ Populate with data from any source
  - ✓ Data supplied by parent application
- Report storage (optional)
  - ✓ Serializable as a *JasperPrint* object
  - ✓ PDF, HTML, XLS, XML, CSV output options



Library API Overview



Data Sources

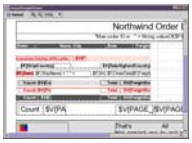
- Runtime environment agnostic
  - ✓ Jasper does not make any assumptions about type of data source
  - ✓ All data supplied as either input parameters, or from a data source
  - ✓ Parent application is responsible for supplying fill data
- Parameterized query support
  - ✓ Partial or total dynamic/runtime query construction using report parameters
- Pluggable data model
  - ✓ Simple and common data supplier interface
    - *JRDataSource* interface
  - ✓ Pre-packaged JDBC data source implementation
    - Report embedded SQL query
  - ✓ JavaBeans based data source implementations for wrapping EJBs, Hibernate or POJOs
  - ✓ XML data source

Report Layout

- Section oriented and multi-column report layouts
  - ✓ Title, detail, summary, page, column and group headers and footers, etc.
- Data grouping
  - ✓ Any number of nested groups
  - ✓ Inserts group footer/header sections on each group break
- Wide range of graphic and text elements
  - ✓ Lines, rectangles, images, static texts, dynamic text fields, etc
  - ✓ Comprehensive control over report element style properties
- Pixel-perfect layout
  - ✓ Designed with printing in mind
  - ✓ All report elements can be precisely positioned and sized
- Drill-down / Drill-through
  - ✓ Reports can contain various types of hyperlinks
  - ✓ Enables drill-down reports and drill-through to external document

Complex Layouts

- Complex report requirements may include:
  - ✓ Simultaneous iteration through multiple data sets
  - ✓ Aggregation of multiple pieces of information

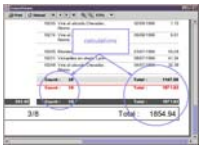


- Complex report requirements are handled via "subreports"
  - ✓ Any report template can be used as a subreport inside another template
  - ✓ Allows report designers to achieve arbitrarily complex report layout

Report Expressions

- Report expressions defined using Java
  - ✓ Provides robust expression definition
  - ✓ Enables existing code to be leveraged
- JasperReports special syntax
  - ✓ Enables references within expression parameters, fields, or report variables
- Expressions are compiled to bytecode
  - ✓ Faster evaluation vs. interpreter or scripting language

Calculation Engine



- Built-in and user-defined variables
  - ✓ Page number, record counts, etc
  - ✓ Late-fill capability
- Built-in and calculations
  - ✓ Implemented using arbitrarily scalable algorithms: sum, count, min, max, average, variance, standard deviation
- User-defined calculations
  - ✓ Built on top of expressions
  - ✓ Pluggable calculation interface
- Flexible scope of calculations
  - ✓ Report, page, column, or group level
- Scriptlet API allows custom-code execution during report-filling
  - ✓ Convenient way to place some business logic associated with the report data
  - ✓ E.g. prepare data for a chart

Internationalization

- Leverages Java platform internationalization support
  - ✓ Unicode
  - ✓ Locales
  - ✓ Resource bundles
- Single report design can be rendered in different languages/locales
  - ✓ Text element translation
  - ✓ Date/number/time/currency formatting

Report Viewers

- Embeddable report viewers
  - ✓ Built-in Java/Swing report viewer
  - ✓ Available open source Eclipse/SWT report viewer

Requirements

- Java JDK
  - ✓ JDK 1.2.2 or higher
- XML
  - ✓ JAXP 1.1 XML Parser
- Jakarta Commons Digester Component (version 1.1 or later)
  - ✓ http://jakarta.apache.org/commons/digester/
- Jakarta Commons BeanUtils Component (version 1.1 or later)
  - ✓ http://jakarta.apache.org/commons/beanutils/
- Jakarta Commons Collections Component (version 1.0 or later)
  - ✓ http://jakarta.apache.org/commons/collections/
- Jakarta Commons Logging Component (version 1.0 or later)
  - ✓ http://jakarta.apache.org/commons/logging/
- JDBC
  - ✓ JDBC 2.0 Driver
- PDF
  - ✓ iText (version 1.01 or later)
    - http://www.lowagie.com/iText/
- XLS
  - ✓ Jakarta POI (version 2.0 or later)
    - http://jakarta.apache.org/poi/

JasperReports for JBoss

- ✓ Working with static report templates
- ✓ Ad-hoc report templates
- ✓ Data sources
- ✓ Reports in HTML format
- ✓ PDF and XLS output

Static Report Templates

- Report compilation using ANT

```
<target name="compile-reports">
  <taskdef name="jrc" classname="net.sf.jasperreports.ant.JRAntCompileTask">
    <classpath refid="classpath"/>
  </taskdef>
  <jrc srcdir="./reports" destdir="./build/reports" tempdir="./build/temp"
    xmlvalidation="true">
    <classpath refid="classpath"/>
  </jrc>
</target>
```

- Deploying and locating report files

```
File reportFile =
  new File(application.getRealPath("/reports/WebappReport.jasper"));

...

JasperReport jasperReport =
  (JasperReport)JRLoader.loadObjectFromLocation("com/mycomp/rpt/Invoice.jasper")
```


Ad-hoc Report Templates

- Dynamic report structure
  - ✓ User input required for report layout
  - ✓ Higher level or simplified report design framework
- Building report templates using the API
  - ✓ Specialized package and object model
  - ✓ Runtime report template compilation
- Altering report templates without requiring recompilation
  - ✓ Style related and non-validation related modifications

Data Sources

- JavaBeans compatible data source implementations
  - ✓ JRBeanCollectionDataSource
  - ✓ JRBeanArrayDataSource
- Access to nested object properties
- Reporting directly from DTOs, EJBs, Hibernate or simple Java data objects

Reports in HTML Format

- Paginated HTML output
- Flow oriented HTML output
- Working with images in HTML

```
JRHtmlExporter exporter = new JRHtmlExporter();

Map imagesMap = new HashMap();
session.setAttribute("IMAGES_MAP", imagesMap);

exporter.setParameter(JRExporterParameter.JASPER_PRINT, jasperPrint);
exporter.setParameter(JRExporterParameter.OUTPUT_STRING_BUFFER, sbuffer);
exporter.setParameter(JRHtmlExporterParameter.IMAGES_MAP, imagesMap);
exporter.setParameter(JRHtmlExporterParameter.IMAGES_URI, "image.jsp?image=");
exporter.setParameter(JRExporterParameter.PAGE_INDEX, new Integer(pageIndex));

exporter.exportReport();
```

PDF and XLS Output

- PDF Output

```
bytes[] bytes = JasperExportManager.exportReportToPdf(jasperPrint);
...
response.setContentType("application/pdf");
response.setContentLength(bytes.length);
```

- XLS Output

```
JRXlsExporter exp = new JRXlsExporter();
exp.setParameter(JRExporterParameter.JASPER_PRINT, jasperPrint);
exp.setParameter(JRExporterParameter.OUTPUT_STREAM, baos);
exp.setParameter(JRXlsExporterParameter.IS_ONE_PAGE_PER_SHEET, Boolean.FALSE);
exp.setParameter(JRXlsExporterParameter.IS_REMOVE_EMPTY_SPACE_BETWEEN_ROWS,
    Boolean.FALSE);
exp.setParameter(JRXlsExporterParameter.IS_WHITE_PAGE_BACKGROUND,
    Boolean.FALSE);
exp.exportReport();
...
response.setContentType("application/vnd.ms-excel");
response.setHeader("Content-Disposition", "inline; filename=\"file.xls\"");
response.setContentLength(bytes.length);
```

Questions