Reporting

Creating your own reports for Data Crow

Author: Robert Jan van der Waals
Date: June 13, 2007
Version: 0.1
Based on Data Crow version 2.12

1 Introduction

Data Crow allows users to add their own report or to modify the existing reports. Knowledge of XML and XSLT (XSL Transformations) is needed to successfully create a new report. Extensive information on these subjects can be found here:
http://www.w3.org/TR/xslt
http://www.w3schools.com/xsl
http://en.wikipedia.org/wiki/XML

This document will not delve deep into these subjects. The purpose of this document is a brief introduction to the default available reports, how to change certain aspect of the reports and how to use the fields of Data Crow in the reports.

2 Who can create reports?

The good news is that you don't need Java programming skills to create a new report. However, the slightly less good news is that you will need to know a certain script language. The script language is not overly complicated and someone with common sense and maybe a little bit of programming skills will easily get into it. There are tons of resources and courses available on the internet to help you out. A little bit of HTML knowledge might also come in handy.

3 Used Technologies

The technology used to create the reports is based on XML (Extensible Markup Language) and XSLT (XSL Transformations). When creating a report an XML document is created which is transformed via the XSLT document to a HTML or PDF document. When you want to add a new report to Data Crow you will have to provide a XSLT document. The XML document itself is created by the reporting functionality.

3.1 XML

All Data Crow reports are based on XML files. The reporting functionality exports the items to an XML file and generates a XML Schema (XSD) which is used test the XML document for validity. If you don't know anything about XML I suggest you go to the following webpage for a brief introduction:
http://en.wikipedia.org/wiki/XML

You don't actually need to know how to create a XML document or a XML Schema as Data Crow creates these for you. The reports (XSLT) however are also XML documents, so little knowledge about XML documents might prove useful.

3.2 XSLT

XSLT stands for XSL Transformations. Via a scripting language the XML source document is converted to another document type, like HTML and PDF. The XSLT documents are the actual Data Crow report documents; creating a new report solely involves writing a XSLT document.

3.3 Java

Data Crow uses libraries from the Apache Foundation for the XML transformations:

For HTML documents:
http://xml.apache.org/xalan-j/

For PDF documents:
http://xmlgraphics.apache.org

You don't need to know anything about of these libraries in order to create a report.

4 Data Crow Reports

When I mention a report I do not mean the generated version but the template used to create the report. This template is a simple XSLT file and is not an invention of me and is neither a Data Crow specific implementation. It's XSLT in its purest form.

4.1 File structure

The reports are stored in a folder called "reports", situated in the Data Crow installation directory. You will find several sub directories here, one for each module to be exact. There the division does not stop. Each module directory also contains a sub directory called "html" and "PDF". Why? The templates for creating a PDF or a HTML are substantially different. For now, just believe me on this on.

The templates always have the extension "XSL".

Tip 1 If you are a beginner I would advise you to start with a html report rather then a PDF template.

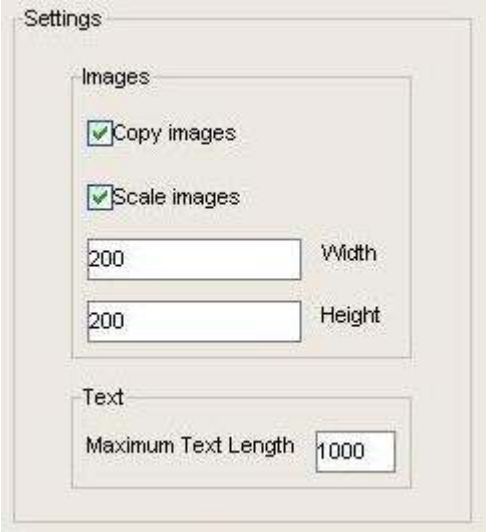
Tip 2 Always create a copy of the template you want to modify. You can select the copied version from the report dialog as well.

4.2 Report properties

You might have noticed report having a separate properties file. These properties files are created automatically by Data Crow. These files live alongside the "XSL" file; <*filename*>.properties.

Property	Description	Allowed values
image_height	Maximum image height	Any numeric value
image_width	Maximum image width	Any numeric value
report_name	Name of the report (as shown on screen)	Any text value
max_text_length	Maximum number of characters before the text gets truncated	Any numeric value
scale_images	Indicates if the images should be scaled	True / false
copy_images	Indicates if the images should be copied to the report location. A sub-directory is created: <report-name>-images	True / false

All of these options can be set via the report dialog (except the name of the report).
So instead of creating this file by hand you can go to the report dialog, select your report and adjust its properties. A properties file for the selected report will be created for you with the selected values filled in.



Tip 3 Set the properties via the report dialog

5 Data Crow fields

How do you know which fields to add to a report? Which name of the field should you be using? To make this easy for you a dictionary file is generated every time you close the report dialog.

Below you find a sample of this file:

```
module                  [Software Item]
xsl reference name      [software-item]
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
  Picture CD            = picture-cd
  Platform              = platform
  Storage Medium        = storage-medium
  User Numeric Field 1  = user-numeric-field-1
  Screenshot One        = screenshot-one
```

From the information above you now know that the field with label "Picture CD" can be added via the name "picture-cd" (CaSe SeNsItIvE). A Software Item should be referenced via the name "software-item".

5.1 Printing a field

Printing the content of a field is accomplished thru the statement below.

```
<xsl:value-of select=" storage-medium "/>
```

To print an actual image you have to do something special. This is described in the next paragraph.

5.2 Printing an Image

Image fields are somewhat special. The image field contains a fully qualified path to the picture (either the copied version or the original version). You can display the path to this file or show the image itself. The latter is more challenging:

HTML report image definition

```
<img alt="">
  <xsl:attribute name="src"><xsl:value-of select="picture-cd" />
</img>
```

PDF report image definition

```
<fo:block>
  <fo:external-graphic content-width="4cm" content-height="4cm" scaling="uniform">
    <xsl:attribute name="src">url('<xsl:value-of select="picture-cd" />')</xsl:attribute>
  </fo:external-graphic>
</fo:block>
```

Sometimes you might want to (to overcome layout problems) print a value even when the picture is empty.

HTML report image definition

```
<xsl:if test="picture-cd != ''">
  <img alt="">
    <xsl:attribute name="src"><xsl:value-of select="picture-cd" />
  </img>
</xsl:if>

<xsl:if test="picture-cd = ''">
   
</xsl:if>
```

PDF report image definition

```
<fo:block>
  <xsl:if test="picture-cd != ''">
    <fo:external-graphic content-width="4cm" content-height="4cm" scaling="uniform">
      <xsl:attribute name="src">url('<xsl:value-of select="picture-cd" />')</xsl:attribute>
    </fo:external-graphic>
  </xsl:if>
  <xsl:if test="picture-cd = ''"> </xsl:if>
</fo:block>
```

6 Document structure

In this chapter a brief introduction is given of the existing report files.

6.1 PDF report

```
01 <?xml version="1.0" encoding="UTF-8"?>
02 <xsl:stylesheet version="1.1" xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
xmlns:fo="http://www.w3.org/1999/XSL/Format" exclude-result-prefixes="fo">
03 <xsl:output method="xml" version="1.0" omit-xml-declaration="no" indent="yes"/>
04 <xsl:param name="versionParam" select="'1.0'"/>
05 <xsl:template match="/">
06   <fo:root xmlns:fo="http://www.w3.org/1999/XSL/Format">
07     <fo:layout-master-set>
08       <fo:simple-page-master master-name="A4" page-height="29.7cm" page-width="21cm"
margin-top="2cm" margin-bottom="2cm" margin-left="2cm" margin-right="2cm">
09         <fo:region-body/>
10         <fo:region-after />
11       </fo:simple-page-master>
12     </fo:layout-master-set>
13
14     <xsl:for-each select="data-crow-objects/software-item">
15     <fo:page-sequence master-reference="A4">
16
17       <fo:static-content flow-name="xsl-region-after">
18         <fo:block text-align="right" font-size="10pt">page <fo:page-number/></fo:block>
19       </fo:static-content>
20
21       <fo:flow flow-name="xsl-region-body">
22
23         <fo:block font-size="11pt" font-weight="bold"><xsl:value-of select="title"/></fo:block>
24
24         <fo:block font-size="11pt">
25           <fo:table table-layout="fixed" width="100%" border-collapse="separate" space-before="10">
26
27             <fo:table-column column-width="3cm"/>
28             <fo:table-column column-width="15cm" />
29
30             <fo:table-body>
31
32               <fo:table-row height="1cm">
33                 <fo:table-cell><fo:block font-weight="bold">Description</fo:block></fo:table-cell>
34                 <fo:table-cell><fo:block><xsl:value-of select="description"/></fo:block></fo:table-
cell>
35               </fo:table-row>
36
37             </fo:table-body>
38           </fo:table>
39         </fo:block>
40
41     </fo:flow>
42   </fo:page-sequence>
43   </xsl:for-each>
44 </fo:root>
45
46 </xsl:template>
47 </xsl:stylesheet>
```

01 – 06 Definition of the XML version and default definitions for this XSL style sheet. These can safely be re-used for any template.

07 – 12 Definition of the page layout. In this case A4. The actual paper size is not based on the name "A4" (as any name is allowed) but by the page-height and page-width. The name of the layout (in this case "A4") is as a reference in line 15.

Tip 4 By swapping the values of the page width and height you can create a landscape report.

09 – 10 Defines the page structure. In this case the body of the page is printed and a region called "after". The "region after" is used in this report to display the current page number.

Tip 5 To remove the page-numbering remove line 10 (region after definition) and line 17 to 19.

14 We are going to loop-thru the report items. The statements between line 14 and 43 are executed for each item in the report. Items are always referenced via "data-crow-objects/<module name>". The "module name" can be found in the dictionary.

15 Indicates the beginning of the actual report generation, the content, and holds a reference to the page definition as defined in line 07 to 12. Using the "A4" page definition the reporting functionality will know when a new page has to be created.

17 – 19 A static block for the "region after" section. This static block is printed out for each and every page. In this report, for every page the page number is printed out.

21 Marks the beginning of the document body.

23 – 39 With these lines the attributes of the item are printed out in a table.

23 The title of the current software item is printed out in a bold font

25 – 28 A table is created. The width 100% means it is allowed to take the full page width. In this report the table has only two columns, one has a width of 3cm and one has a width of 15cm.

30 Indicates that information will be added to the table from this point onwards.

32 – 35 A single table row is added to the table. The row has two cells (corresponding to the column definition as seen in lines 27 and 28). The first cell prints the static text "Description". The second cell (with a maximum width of 15cm) prints the description of the current software item. More rows could be added to this table, for example for the year of the software item or the developer.

General Rules

1. Tags, such as “<fo:layout-master-set>” are always ended with “</tag-name>” (in this example “</fo:layout-master-set>”.
2. Anything that should be printed to the report must be surrounded by “<fo:block>” and “</fo:block>”. Tables, contents of table cells, all should be surrounded by this tag.

6.2 HTML report

```
01 <?xml version="1.0" encoding="ISO-8859-1"?>
02 <xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
03 <xsl:template match="/">
04 <html>
05 <head>
06   <style type="text/css" MEDIA="screen">
07      body {    font-family: Arial, Verdana;
08                font-size: 11px;}
09      h1 {      font-size: 13px;
10                font-weight: bolder;}
11      tr, td.label,td.text {
12                height: 35px;
13                vertical-align: top;}
14      td.label {
15                font-weight: bolder;
16                font-size: 12px;
17                background-color: #FAFAD2;}
18      td.text {
19                font-size: 12px;
20                background-color: #FFFFCC;}
21   </style>
22 </head>
23
24 <body>
25  <h1>Software Items</h1>
26
27   <xsl:for-each select="data-crow-objects/software-item">
28      <h1><xsl:value-of select="title"/></h1>
29      <table width="600" style="border:0;" cellspacing="0" cellpadding="10">
30         <colgroup valign="top" align="left">
31            <col width="100" />
32            <col width="500" />
33         </colgroup>
34
35         <tr><td class="label">Description</td>
36               <td class="text"><xsl:value-of select="description"/></td></tr>
37      </table>
38
39     <br /><br />
40   </xsl:for-each>
41  </body>
42 </html>
43 </xsl:template>
44 </xsl:stylesheet>
```

01 – 03 Definition of the XML version and default definitions for this XSL style sheet. These can safely be re-used for any template.

04 – 22 Start of the HTML document. This is just normal HTML, nothing specific. In the head the style is specified; the font, font size and background colour. You can check what you can use here by searching the internet for “CSS” (Cascading Style Sheets).

24 Start of the body of HTML document (the actual content).

25 Prints the static text “Software Items” as a header for this document.

27 We are going to loop-thru the report items. The statements between line 27 and 40 are executed for each item in the report. Items are always referenced via “data-crow-objects/<module name>”. The “module name” can be found in the dictionary.

28 The title of the current software item is printed to document.

29 – 33 A table is defined. The table has two columns, the first has a width of 100 and the second has a width of 500. The table itself has a maximum width of 600.

35 – 36 A single table row is added to the table. The row has two cells (corresponding to the column definition as seen in lines 30 to 33). The first cell prints the static text “Description”. The second cell (with a maximum width of 500) prints the description of the current software item. More rows could be added to this table, for example for the year of the software item or the developer.