WORKING WITH THE JASPER_REPORTS MODULE

Module explained

Abstract

This document's goal is to explain users looking for a reporting tool with a descent graphical report designer environment a better explanation on how this module can be used.

Eric Christensen (Stretcha bvba)

Eric.Christensen@stretcha.be

Contents

1 Working with the jasper_reports module

The jasper_reports module is a module available as an add-on to the Odoo ERP software solution. It provides the user with the ability to design his own high quality reports and make exports to, for example xls, quite easily.

Undelying engine for the module as one can expect the JasperReports®Server.

Remark: most of the document is made out of the Jasper Studio perspective, iReport slightly differs in the way the different required elements can be defined but the for the greater part is still correct.

1.1 Version of the underlying JasperReports® Server

The product currently is running on the JasperReports® Server 5.0.0 release. By using Jasper Studio versions with a higher release number then the JasperReports® Server in the module it is possible that some features will not be supported properly of even cause errors.

Prerequisite here is the installation of the Java DK Version on your linux machine.

1.2 iReport and or Jasper Studio

1.2.1 iReport

iReport requires a JRE 6 or 7 to be installed on your workstation, it also requires you to point the configuration file of iReport to the correct JRE.

The configuration file for iReport can be found in the directory:

```
C:\Program Files (x86)\Jaspersoft\iReport-5.0.0\etc
```

The file itself is named *ireport.conf* and you need to adjust the *jdkhome* parameter to reflect a pointer to your installed JRE:

```
# default location of JDK/JRE, can be overridden by using --jdkhome <dir> switch

jdkhome="C:\Program Files\Java\jre7"
```

This version of Jasper Reports is known to work with the following versions of iReport:

- iReport 5.0.0

1.3 Jasper Studio

No installation specific requirements known for the moment. It is known to work with:

- Jasper Studio V6.3.1
- Jasper Studio V6.3.0 (if you can live with some very annoying shortcomings of this version of the Jasper Studio software)

1.4 Jasper Reports installation

The module installs as any Odoo module but has some configuration file parameters that are worthwhile setting:

In the Odoo configuration file in the standard options section you can add the following parameters:

jasperpid	File path to a pid file (eq. /home/odoo/jasper.pid) As the jasper server is started by the user of the Odoo process it is quite likely to have no rights in the standard pid directory. Therefore, this parameter allows you to shift the pid file where the Odoo process user is allowed to create a file
jasperport	A valid port number, by default 8090. The port on which the JasperReports® Server will listen for requests
jasperunlink	Either True or False. If True, all temporary files will be unlinked.
jasperdir	Location of the header reports (default is set to the report subdirectory of the module directory)

1.4.1 Custom Reports

Upon installation the module directory contains a sub directory named *Custom Reports*, this is the directory where your uploaded report files will reside (in Jasper terminology .jrxml files). It will also contain the compiled versions of your reports (in Jasper terminology .jasper files). The latter will automatically be created from your base report files by requestion the report.

It is thus important that this directory is writable by the odoo process user.

1.5 Creating a simple one model report with Jasper Reports.

1.5.1 Create a data template

One of the big advantages of the jasper_reports module is that it is capable of using the full power of the Odoo database model. Any field type existing on the Odoo side can be used for reporting.

In order to create a data template, use the menu provided by the menu under the Jasper Reports options on the Setting – Technical tab of Odoo.

You simply select a model and the depth of field recovery and press Create. The process will create an xml based version of the requested model going 4 levels deep.

Once done, download the created xml file to a convenient place for later intergration into the Jasper Reports GDE.

Remark: although there is no theoretical limit to the depth, a depth of 4 showed to be a practical limit because otherwise the create of the xml file will take too much time generating and perhaps other solutions for the creation of your report should be looked into.

1.5.2 Create a report on the iReport or Jasper Studio side

The goal of this document is to explain and help you on the Odoo side of things, it is not intended to be a full manual on how to use the Jaspersoft suite. For that other documentation is readily available and exercising with this software is absolutely advisable.

Good documentation for the Jaspersoft Report Designers (JRD) can be found on:
http://community.jaspersoft.com/documentation (link as per Nov 2016)

This documentation is not really a nice to have but is imperative in the understanding on how to create reports with the Jaspersoft Report Designers.

1.5.2.1 Create a connector in your JRD

The Jaspersoft RDs work with a basis data entity called a datasource (iReport) or data adapter (Jasper Studio), so the first thing we need to do is to convert our xml template file into a JDR datasource.

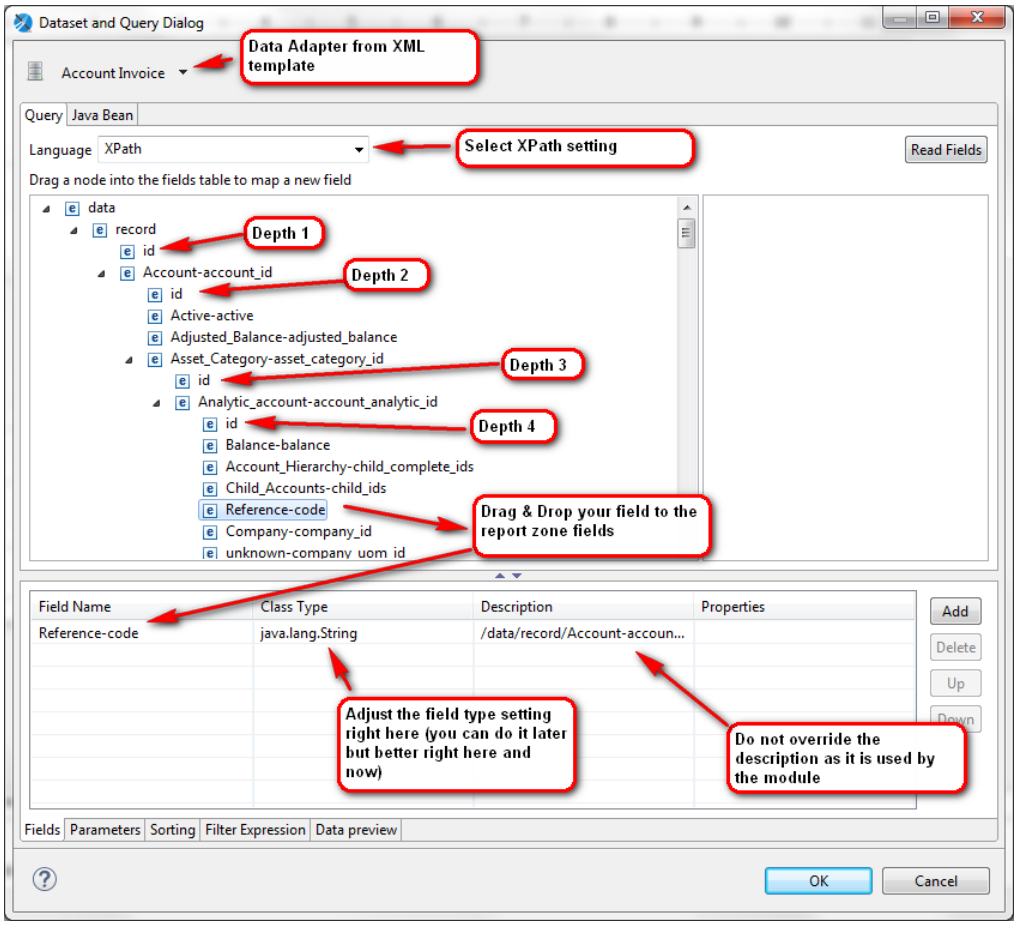
Simply create a new data adapter from the *XML Document* type, then:

- Provide a name for your adapter
- Point to the earlier collected template
- Tick the radio button: Use the report Xpath expression when filling the report

1.5.2.2 Create a first and simple report

Under the File option in the menu use the New item and you will be offered to create a new report. First thing to do is to add some fields from your data adapter into the report.

In the Outline perspective, click right on your newly created report and select the Dataset and Query option.



By this operation your chosen fields will become available under the Outline perspective – Fields section, from where you can drag and drop them to the report area.

Once your report is created, save the jrxml and create the report on the Odoo side through the menu option under Settings -> Technical -> Jasper Reports.

1.6 Parameters available for more complex reports

Some parameters have been implemented for more complex operations:

OPENERP_RELATIONS	Should be defined as a property at the report level and must contain the name of a valid field name (or valid relation chain) in the Odoo model. Example of usage: - We want to print the invoices and their lines from the base model account.invoice, of which the field invoice_line will represent the different lines of a single invoice. In normal reporting terms this can be achieved by a double loop as shown below: *For each invoice in invoices {* *# print for example the header/partner info* *For each invoice_line in invoice.invoice_line {* *# do something with the line* *}* *# print for example the footer info* *}* As Jasper only knows 1 detail section, the detail section would already be occupied by the invoice level loop with no possibility for the second level. Defining an OPENERP_RELATIONS tag will result in a report data file with one line for each relation existing (as many lines as there are lines in the invoice). Therefor you can provide your line details in the details section and use the other available sections for the partner information (i.e. the header and footer information).

OPENERP_COPIES_FIELD	Should be defined as a property at the report level and must contain the name of a valid field name in the Odoo model containing the number of extra copies required of a given report. Example: A field defined on the invoice called invoice_copies which contains the number of extra copies required. Then the parameter should look like: ['invoice_copies']
OPENERP_COPIES	Should be defined as a property at the report level and must contain an integer value equal to the number of extra copies required of a given report.
~~OPENERP_HEADER~~[1]	~~Should be defined as a property at the report level and must contain a valid name of a report in the report sub directory of the jasper module.~~ ~~Example given: portrait_header~~
OPENERP_MODEL	Should be defined as a property at the sub report level in the main report and must contain a valid model name (e.g. account.invoice.line) It can be used in cooperation with other elements to create a connection with another model: 1. Define a main report parameter with an arbitrary name referring to a JRDataSource (class = net.sf.jasperreports.engine.JRDataSource) e.g. $P{SR1_DATA_SOURCE} 2. On the sub report element in the main report define a database connection e.g. $P{SR1_DATA_SOURCE} This required because jasper_reports treats data in a csv manner towards the jasperserver 3. On the sub report element define a property OPENERP_MODEL with the name of the model you want to address 4. On the sub report element define a property (via Edit Properties) OPENERP_PREFILTER which contains a standard odoo domain clause in string format. e.g. "[('a1_id','=',718)]" The clause cannot be dynamic as parameters are only resolved later in the process Instructions 4 and 5 will lead to a filtered response for the sub report data source. If you only use OPENERP_MODEL you will receive the full content of the requested model (which you filter out with the filter expression on the report query, but the pre-filtered approach will in most cases be more efficient as it is performed before passing the data to the report). iReport will require (com.nantic.jasperreports. CsvMultiLanguageDataSource) as a prefix to the datasource connection parameter.

[1] OPENERP_HEADER was not tested in the course of establish this document, the feature is still there.

OPENERP_PREFILTER	A value in the parameter section of the sub report element representing an expression that results in a valid domain clause for Odoo See OPENERP_MODEL just above.
OPENERP_PATH_PREFIX	Should be defined as a property at the sub report level in the main report and must contain a valid field name in the report's selected model. e.g. ['invoice_line']
$P{STANDARD_DIR}	Overwrite the standard directory path (default points to the report subdirectory of the module directory)
$P{SUBREPORT_DIR}	Define the path to eventual sub reports (defaults to the same path as the main report file, i.e. the customer_reports directory)

1.7 Reporting from the same model as the main report

Reporting from the same model can be achieved by using the main report parameter $P{REPORT_DATA_SOURCE} as data source expression on the sub report element and extend the sub element report's parameter section with the following parameter:

REWIND with value: $P{REPORT_DATA_SOURCE}.moveFirst()

These instructions will pass the main report's dataset to the sub report and rewind it to the first record making it available again for processing.

2 Exporting data to XLS format

This section contains some hints and tips on exporting data to xls format from a jasper report:

- Although many suggestions can be found on how to achieve multiple worksheets in a single xls, the following tip works:
  At the top of your sub report band place an empty static field and define with it a property with the following name and value:
  net.sf.jasperreports.export.xls.break.after.row : true
  Remark: this static field will be added to the before worksheet, and the new worksheet will start with the new contents
- I only found it to work well if the different sub reports required to be in different sheets are defined in different bands
- Standard worksheets are named Page 1, Page 2, Page 3, … if you want to provide different names to the worksheets, the best way to achieve this is using a property at the main report level:
  net.sf.jasperreports.export.xls.sheet.names.{"some_name"} : Name Sheet 1/ Name Sheet 2/ …
- Or you can name the sheets at the report level when setting the following property on the sub report element in the main report:
  net.sf.jasperreports.export.xls.sheet.name : some name

Follow the following link to get a better understanding of the different properties available:

http://jasperreports.sourceforge.net/config.reference.html

3 Many ways to get to Rome

As you can understand there are many ways to achieve the same result with this module, especially when you have models in a header/lines approach.

In the Invoice example there are multiple ways to obtain almost the exact same result:

- Using OPENERP_RELATIONS, will format the reporting dataset such that all lines can be found at the main report level, just using the right values in the right bands will provide the invoice report, the advantage of this approach is the fact that no sub report is required.
- The same can off course be obtained by reporting from the account.invoice.line model, but selection of the invoice is less convenient as the model is not immediately exposed on the Odoo side.
- Using a sub report based on REPORT_DATA_SOURCE, in which the lines are extracted from the main model (account.invoice)
- Using a sub report based on OPENERP_MODEL and OPENERP_FILTER, the main report will provide the invoice level, the sub report will provide the lines level, although this approach is more meant for loosely coupled models or more complex reporting situations