

Java Reporting com JasperReports e iReport Open Source

"JasperReports é uma solução open source poderosa e flexível para geração de relatórios. O visual designer iReport permite tirar total vantagem do poder do JasperReports sem necessidade de conhecimento profundo do formato XML nativo JasperReports."

John Ferguson Smart

O JasperReports é um poderoso e flexível gerador de relatórios open source. É fácil de ser integrado a aplicações Java empresariais, mas carece de um editor de relatórios visual integrado. Portanto, se quiser usar diretamente o JasperReports, precisará manipular sua estrutura de relatórios XML — uma atividade relativamente técnica, com uma curva de aprendizagem alta, para dizer o mínimo.

Na realidade, escrever do começo um JasperReport completo usando só o formato XML representa uma tarefa longa, dolorosa e pouco compensadora.

Afortunadamente, existem algumas alternativas disponíveis que são muito mais fáceis de usar. A melhor de todas, é o uso de um editor visual para projetar, compilar e testar os relatórios.

Um dos editores visuais mais úteis que podemos usar é o iReport. Este artigo demonstra como usar o iReport para aproveitar todo o poder do JasperReports, sem se emaranhar em complexidades do formato XML nativo do JasperReports.

Começando

A primeira coisa a fazer, é carregar e instalar o iReport. Esta é uma aplicação Java, portanto, precisaremos de um JDK na máquina (JDK 1.4 ou maior - este tutorial usa o JDK 1.5.0):

Fazer o download do iReport no ireport.sourceforge.net.

Descomprimir o arquivo iReport.

Rodar o startup script (bin\startup.bat ou. / bin/startup.sh).

O download do iReport vem com o seu próprio pacote JasperReports (a versão mais atual - a 0.5.1 - suporta o JasperReports 1.0.1 recentemente liberado).

Uma vez que o iReport esteja rodando, podemos começar a projetar os relatórios!

O Banco de dados

Este tutorial usa um banco de dados muito simples para efeitos de demonstração (ver Figura 1). Para segui-lo passo a passo, poderemos tanto carregar os scripts para preparar este banco de dados com o MySQL e montá-lo na sua máquina, ou então, usar um banco de dados similar e traduzir as técnicas para a sua situação particular.

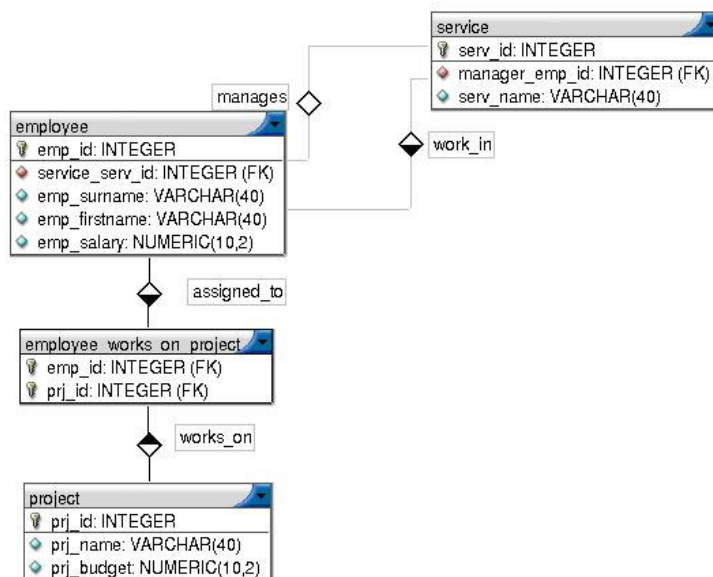


Figura 1. O Schema do Tutorial Employee Database

Adicionando uma Nova Conexão de Banco de dados

Primeiramente, adicione uma nova conexão ao seu banco de dados. Use o menu "Datasource -> Connections/Datasources" para montar uma nova conexão de banco de dados (ver Figura 2). Se selecionar o driver JDBC da lista (no exemplo escolhemos o MySQL), entre com o endereço do servidor e o nome do banco de dados, e clique no botão 'Wizard'. O iReport deverá fornecer uma URL JDBC correta para o seu banco de dados particular.

Agora que temos um datasource, está na hora de fazer alguma coisa com ele.

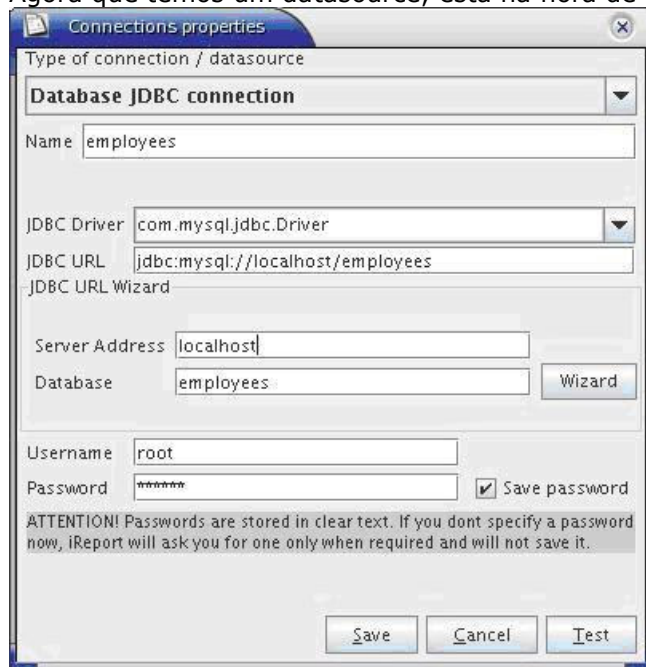


Figura 2. Adicionando uma Nova Conexão de Banco de Dados

Criando um Relatório Simples

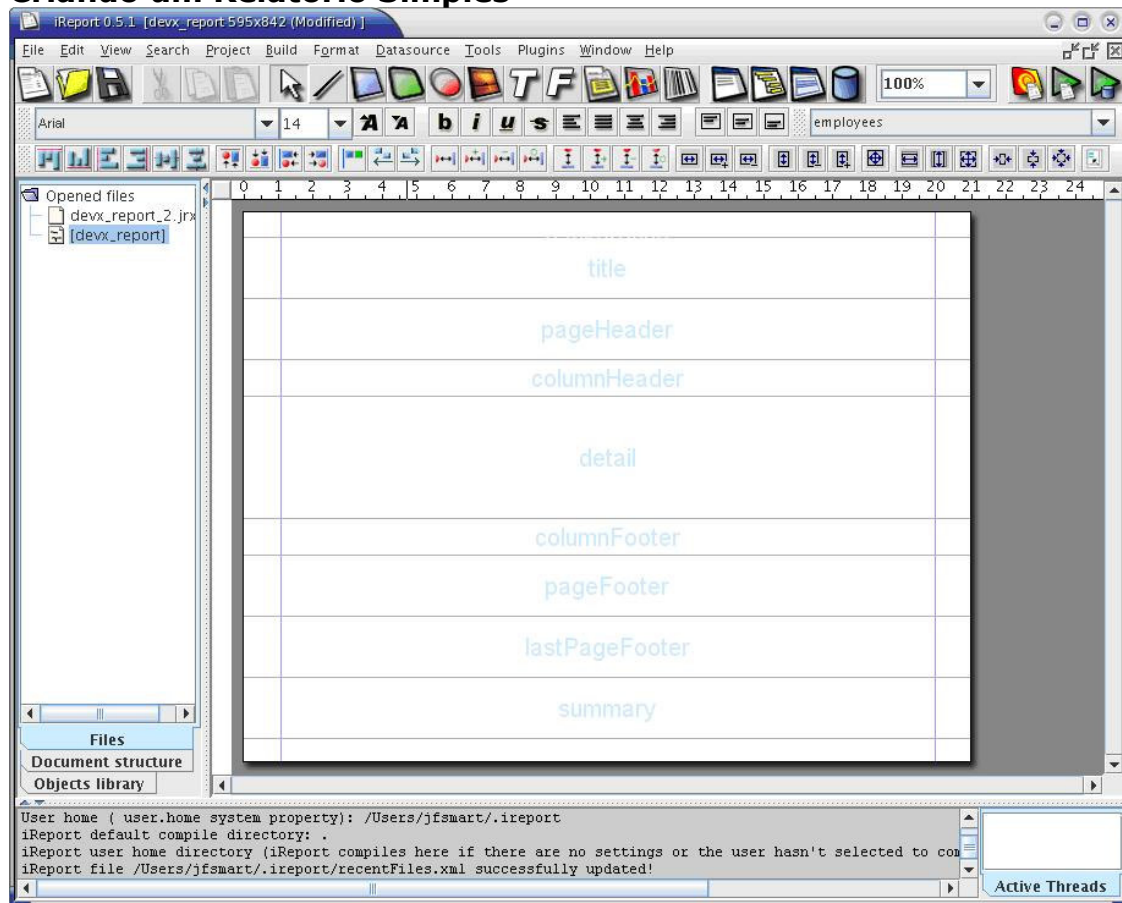


Figura 3. Um Novo Relatório JasperReports

Crie um novo documento JasperReports usando o item de menu "File/New Document". Podemos ignorar todas as demais opções por enquanto. Apenas daremos um nome ao relatório. Teremos assim um relatório vazio, como mostrado na Figura 3.

Um relatório JasperReport é dividido nas seções de exibição mostradas na tela iReport,:

- 1. Title:** como o nome o indica, esta seção contém o título do relatório;
- 2. Page Header:** esta seção aparece no topo de cada página (como esperado). É um bom lugar para colocar datas, numeração de páginas, etc.;
- 3. Column Header:** esta seção aparece no topo de cada coluna;
- 4. Detail:** nesta área serão colocadas as informações para cada item de registro. O JasperReports gera uma seção de detalhe para todo e cada registro processado;
- 5. Column Footer:** esta seção aparece ao final de cada coluna;
- 6. Page Footer:** esta seção aparece ao final de cada página;
- 7. Last Page Footer:** esta seção aparece ao final da última página;
- 8. Summary:** esta seção aparece ao término do relatório, logo após o último registro.

Para começar, especificamos uma consulta SQL apropriada para o relatório, utilizando o menu Datasource/Report Query (ver Figura 4). A consulta recuperará uma lista de todos os empregados no banco de dados:

```
select * from employee e, service s
where e.serv_id = s.serv_id
order by s.serv_name, e.emp_surname, e.emp_firstname
```

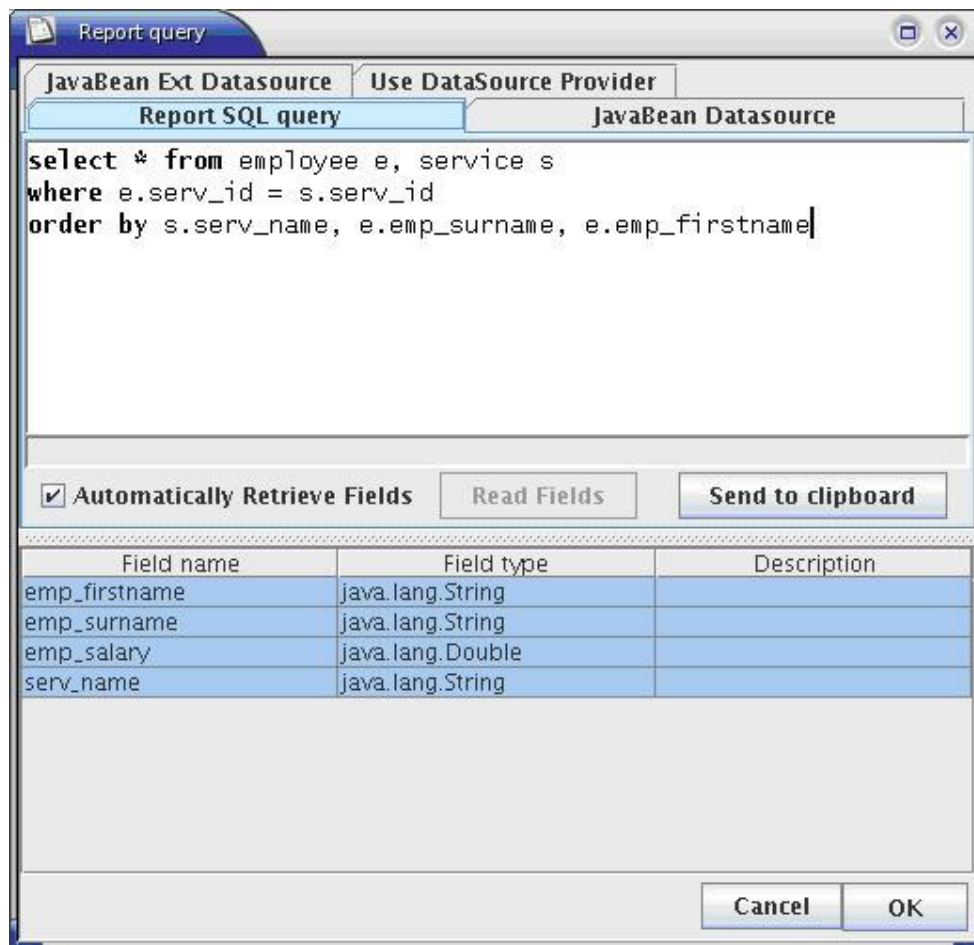


Figura 4. O Report Query

Report Fields

Cada relatório JasperReports tem uma lista de campos que são usados para colocar no layout do relatório, os dados dos registros recuperados pela consulta ao banco de dados

Podemos visualizar a lista de campos utilizando o menu "View/Report Fields" (ver Figura 5).

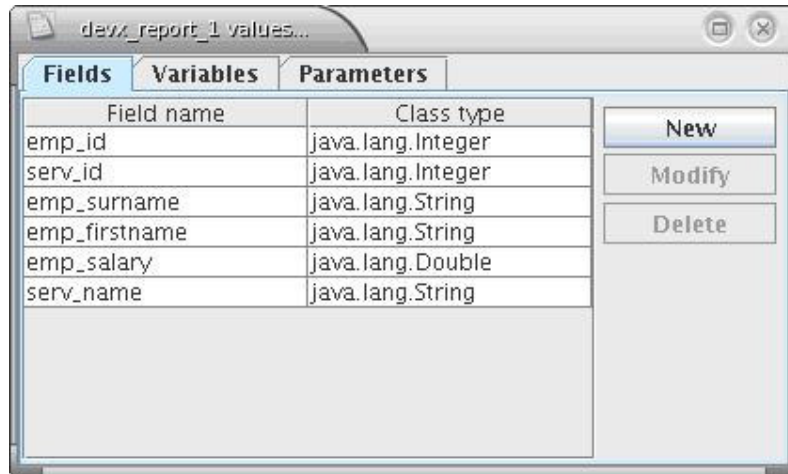


Figura 5 Report Fields

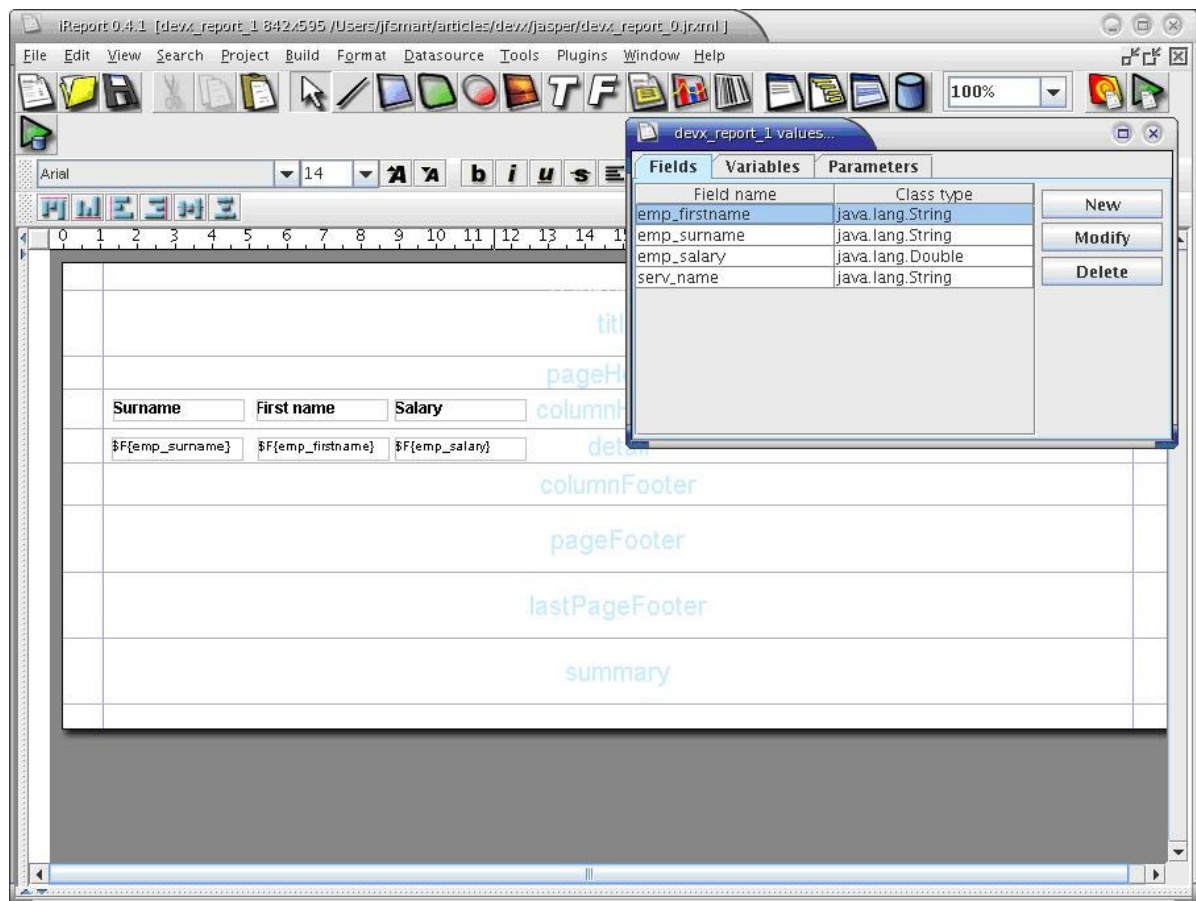
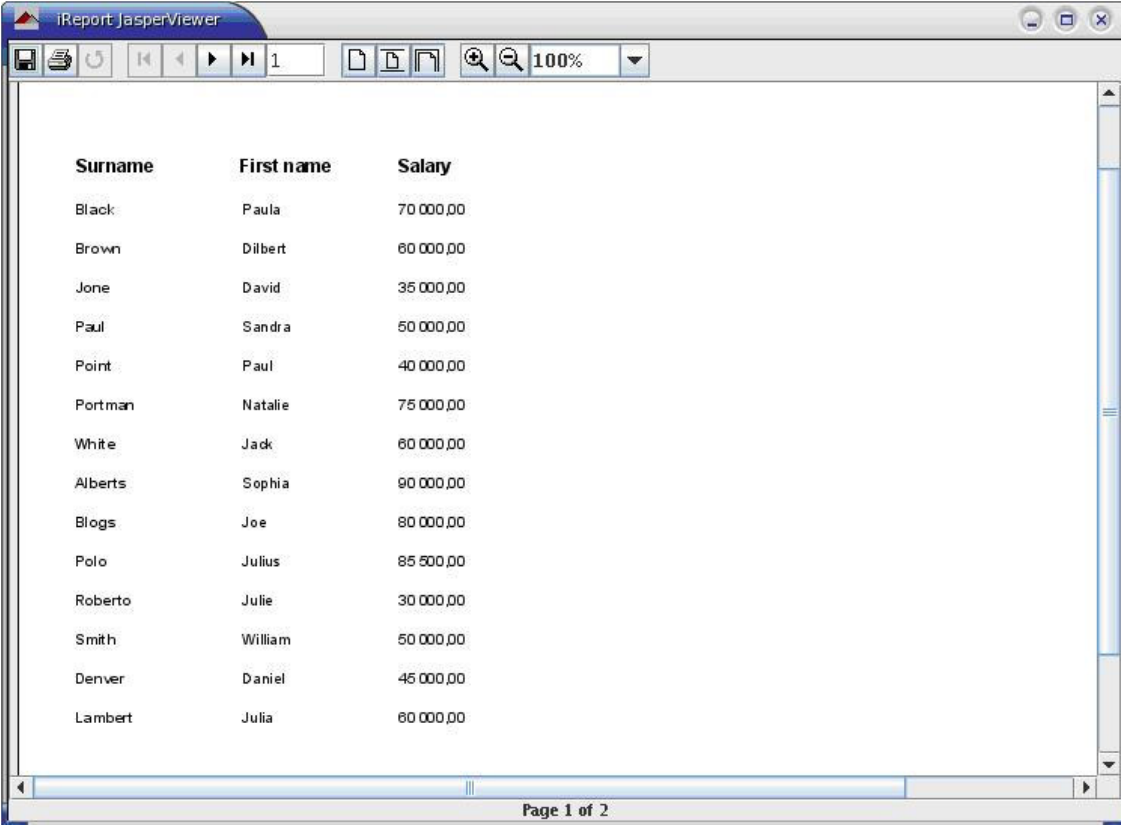


Figura 6. Inserindo os Report Fields

Quando criamos uma consulta SQL, esta lista é automaticamente atualizada com os campos que a consulta retorna. Para os casos nos quais usamos outros tipos de datasources, podemos ter que definir os campos manualmente.

A partir da janela campos de relatório (report fields window), podemos arrastar e soltar campos no relatório. Os campos geralmente entram na seção Detail, como mostrado na Figura 6. Colocamos três campos na seção Detail, junto com os títulos de coluna apropriados na seção Column Header (Cabeçalho de Colunas). Os títulos de coluna são itens de texto estáticos que podemos inserir usando o ícone "T" ou o item de menu "Insert Element/Static Text". Experimente as opções de formatação e de layout para familiarizar-se com que iReport têm a oferecer nesta área.

Para visualizarmos a saída do relatório, rodamos o mesmo utilizando o item de menu "Build/Execute Report" (usando active conn). Deveremos obter um relatório como o da Figura 7.



Surname	First name	Salary
Black	Paula	70 000,00
Brown	Dilbert	60 000,00
Jone	David	35 000,00
Paul	Sandra	50 000,00
Point	Paul	40 000,00
Portman	Natalie	75 000,00
White	Jack	60 000,00
Alberts	Sophia	90 000,00
Blogs	Joe	80 000,00
Polo	Julius	85 500,00
Roberto	Julie	30 000,00
Smith	William	50 000,00
Denver	Daniel	45 000,00
Lambert	Julia	60 000,00

Figura 7. Pré-visualização dos Resultados do Relatório

Adicionando Título de Relatório e Data

Então agora podemos gerar um relatório com dados reais. Adicionemos agora um título na barra de título. Este título só será exibido no começo do relatório. Suponhamos que o título seja "Employees/service" seguido pela data do dia. Para inserir o texto "Employees/Service", apenas colocamos uma zona de texto estática na seção title (usar "Edit/Insert Element/Static Text" ou o ícone "T").

Agora, suponhamos que desejamos também exibir a data atual. Inserimos um novo campo texto ("Edit/Insert Element/Text Field"). Fazemos duplo clique no objeto campo texto (text field object) e a seguir, na aba "TextField" da janela (ver **Figura 8**).

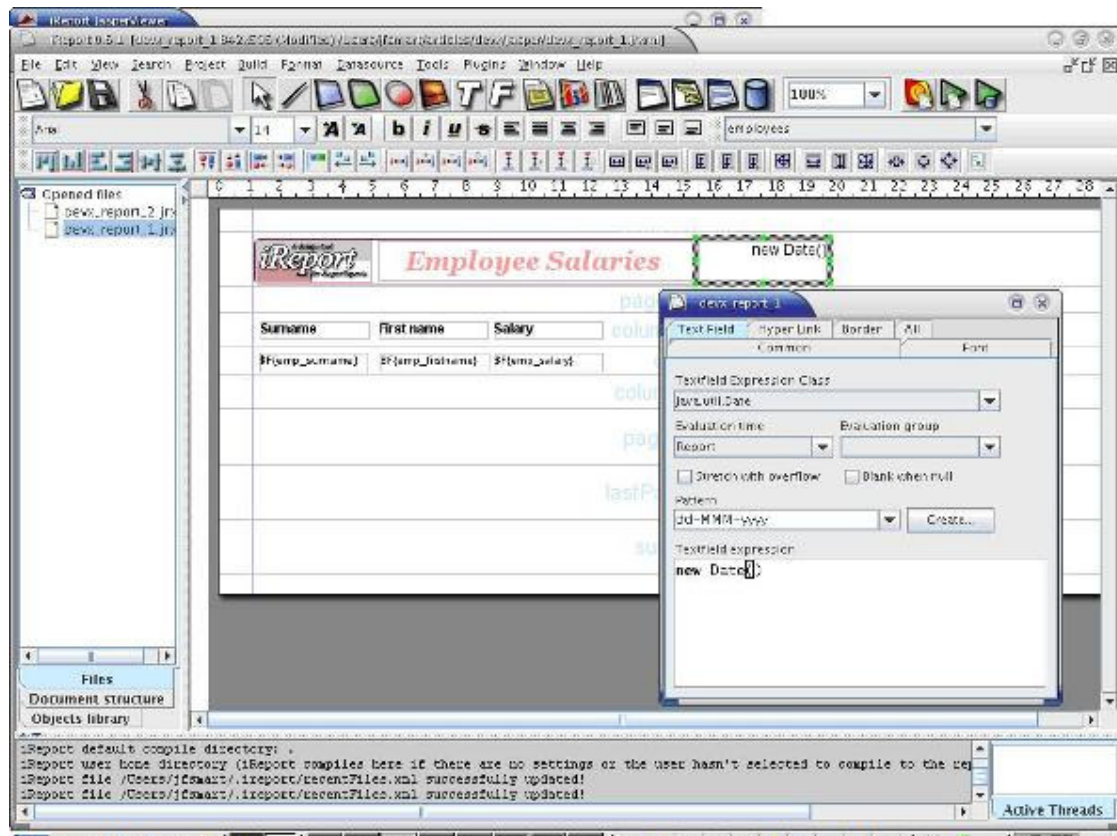


Figura 8. Adicionar um Campo Data

Neste momento temos uma idéia do poder de JasperReports: sendo o JasperReport compilado uma classe Java, podemos usar qualquer expressão Java para auxiliar na construção do relatório, assim como também campos JasperReports, variáveis e parâmetros. Por exemplo, a expressão "TextField" é uma expressão Java e será interpretada como tal.

Para exibir a data atual, criamos apenas uma novo objeto Date() que será automaticamente instanciado como a data e hora atuais. A seguir, informamos ao JasperReports o tipo de expressão a ser usado (na classe Textfield Expression: java.util.Date), quando dever[á ser avaliada a expressão (Evaluation time - momento da Avaliação) e qual formato usar ("Pattern field" - "campo Formato").

Finalizando, adicionamos uma borda transparente ao redor do título ("Edit/Insert Element/Rounded Rectangle") e uma imagem ("Edit/Insert Element/Image") e então personalizamos as cores e fontes (ver exemplo na **Figura 9**).

Surname	First name	Salary
Black	Paula	70 000.00
Jone	David	35 000.00
Point	Paul	40 000.00
White	Jack	60 000.00
Blogs	Joe	80 000.00
Smith	William	50 000.00
Lambert	Julia	60 000.00
West	Carla	55 000.00

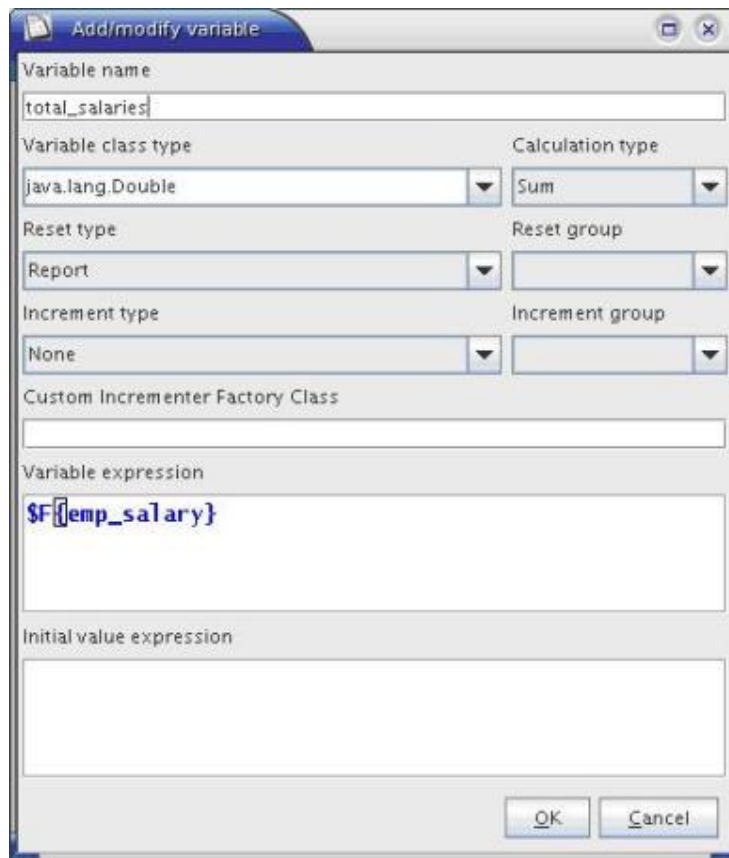
Figura 9. O Relatório com uma Barra de Título

Calculando Totais

Vamos supor agora, que desejamos exibir o custo total de todos os salários dos empregados. Para calcular estes tipos de valores com o JasperReports, precisamos usar variáveis de relatório. Usamos variáveis de relatório para armazenar e calcular qualquer valor temporário necessário para o relatório, tais como totais, subtotais, médias, etc. Vejamos alguns detalhes importantes a respeito de variáveis de relatório:

- O tipo de classe variável (variable class type) tem que ser compatível com o tipo de campo (field type), para que os cálculos funcionem corretamente. Portanto, devemos configurá-lo como Double;
- O tipo de cálculo informa ao JasperReports como calcular a variável. Neste caso, configuramos como 'Sum', para achar a soma total de todos os valores do campo emp_salary;
- A expressão variável representa o valor a ser avaliado no cálculo. Esta expressão pode ter vários formatos:
 - Uma expressão Java (por exemplo, New Date());
 - \$F para campos (por exemplo, "\$F{emp_salary}");
 - \$V para variáveis (por exemplo, "\$V{service_employee_count}");
 - \$P para parâmetros de relatório;
 - \$R para recursos localizados (localized resources).

Para exibir as variáveis de relatório, abrimos o menu "View/Report Variables". Precisamos adicionar uma nova variável para localizar o salário do empregado e calcular o valor total. Chamemos esta variável de total_salaries (ver **Figura 10**). Configuramos o tipo de classe da variável (variable class type) como Double e o tipo de cálculo (calculation type) como Sum.



The screenshot shows the 'Add/modify variable' dialog box. The 'Variable name' field contains 'total_salaries'. The 'Variable class type' dropdown is set to 'java.lang.Double'. The 'Calculation type' dropdown is set to 'Sum'. The 'Reset type' dropdown is set to 'Report'. The 'Increment type' dropdown is set to 'None'. The 'Variable expression' field contains '\$F{emp_salary}'. The 'Initial value expression' field is empty. The 'OK' and 'Cancel' buttons are at the bottom right.

Figura 10. Adicionando uma Variável Nova

Agora, precisamos avaliar o campo 'emp_salary', portanto informamos "\$F{emp_salary}" como a expressão variável (variable expression).

A seguir, arrastamos esta variável para a seção Column Footer e adicionamos um texto de título apropriado (ver o relatório gerado na **Figura 11**). Seguindo os mesmos passos, podemos adicionar facilmente outros tipos de variáveis: médias, contas, valores mínimos e máximos, etc.

Surname	First name	Salary
Black	Paula	70 000,00
Brown	Dilbert	60 000,00
Jones	David	35 000,00
Paul	Sandra	50 000,00
Point	Paul	40 000,00
Portman	Natalie	75 000,00
White	Jack	60 000,00
Alberts	Sophia	90 000,00
Boggs	Joe	80 000,00
Polo	Jules	85 000,00
Roberto	Jules	30 000,00
Smith	William	60 000,00
Denver	Daniel	45 000,00
Lambert	Jules	60 000,00
Total of all salaries		885 500,00

Figura 11. Calculando os Salários Totais

Adicionando Grupos

Suponhamos agora, que desejamos agrupar os empregados pelo tipo de serviço, e calcular um subtotal de salários para cada tipo de serviço. A primeira coisa a fazer é definir um grupo de relatório pelo menu "View/Report Groups" (ver **Figura 12**).

Figura 12. Adicionando um Novo Grupo

O campo mais importante aqui é 'Group Expression'. Cada vez que esta expressão muda, será gerado um novo grupo. Portanto, se agruparmos os registros pelo tipo de serviço, usaremos o campo nome do serviço ("F{serv_name}"). Como imaginado, para que isto funcione corretamente, temos que ordenar apropriadamente a consulta SQL ("order by serv_name,...").

Quando adicionamos um novo grupo, obtemos duas novas seções: "serviceHeader" e "serviceFooter". Estas seções são geradas, respectivamente, no começo e no fim de cada grupo de serviço. Reposicionamos os cabeçalhos de coluna estáticos na seção "serviceHeader" e colocamos o campo `$F{serv_name}` acima de estas colunas, para que ajam como um título de grupo.

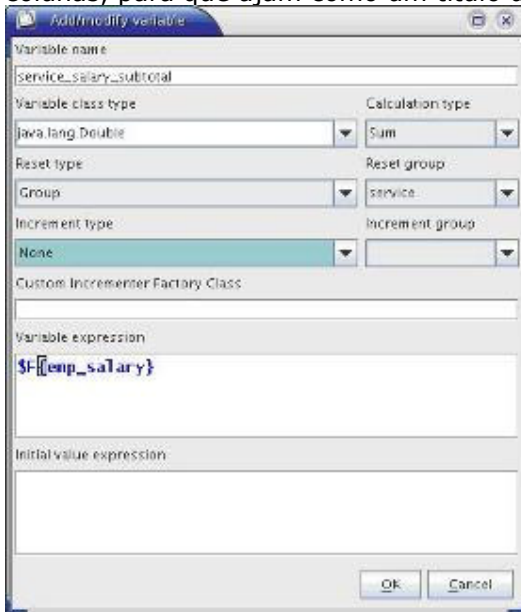


Figura 13. Adicionando um Novo Grupo

Agora criamos uma nova variável chamada `service_salary_subtotal`, como ilustrado na **Figura 13**. Esta última variável, é semelhante à anterior, porém com duas diferenças importantes: Reset Type é 'Group', e Reset Group é 'service', significando que a variável será zerada ao começo de cada novo grupo de serviço.

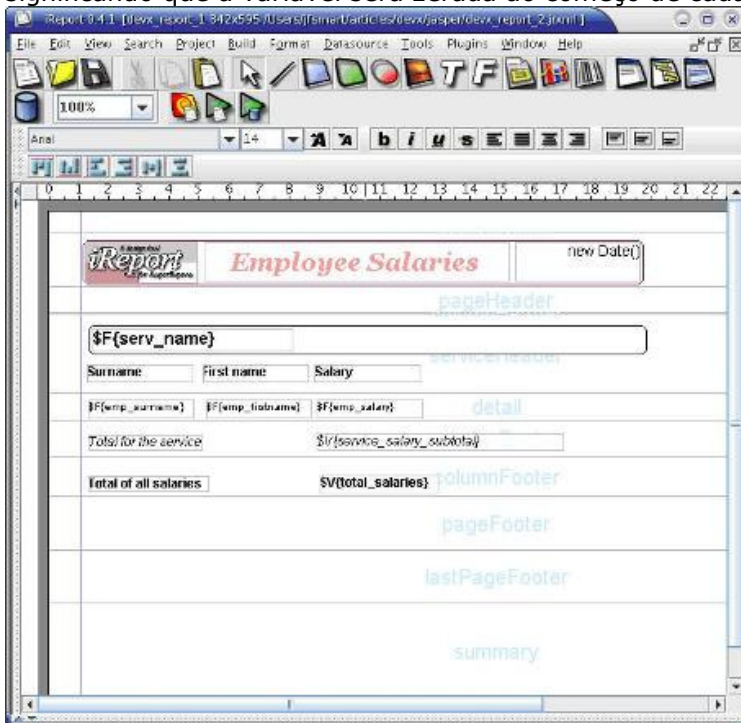


Figura 14. O Layout do Relatório Agrupado

Arrastamos esta variável para a seção "serviceFooter", para exibir o subtotal de todos os salários para cada grupo. O layout deverá ter o aspecto da **Figura 14** e o relatório gerado deverá ter o aspecto da **Figura 15**.

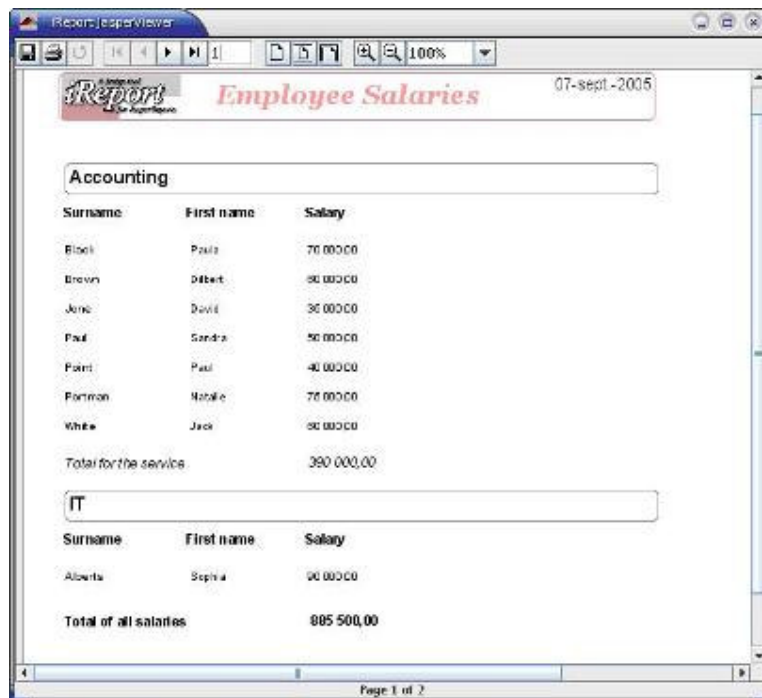


Figura 15. O Relatório Agrupado

Relatórios com Gráficos

No JasperReports 1.0.1 e no iReport 0.5.1, podemos projetar relatórios com gráficos. Suponhamos que desejamos adicionar um gráfico de torta, que mostra ao final do relatório, o custo relativo do salário para cada serviço. Teríamos que colocá-lo na seção Summary. Podemos precisar aumentá-la um pouco, de forma a caber um gráfico de bom tamanho. Portanto, adicionamos um novo gráfico nesta seção usando o item de menu "Edit/Insert Element/Chart" ou o ícone "Chart tool". Escolhemos o gráfico de torta.

Clicamos em new chart e a seguir na aba 'Chart'. A seguir, clicamos no botão 'Edit Chart Properties' e então na aba 'Chart data' (ver **Figura 16**).

Os parâmetros de gráfico são diferentes para cada tipo de gráfico. A aba 'Chart data' do gráfico de torta tem três zonas:

- **Key Expression:** identifica cada fatia. Informar "\$F{serv_name}" para que cada fatia represente um serviço;
- **Value Expression:** informe "\$V{service_salary_subtotal}" para associar ao custo de salário total de cada serviço;
- **Label Expression:** é o rótulo que será exibido para cada fatia. Informar "\$F{serv_name}" para exibir o nome do serviço.

Agora rodamos o relatório. Deveremos obter um gráfico de torta ao término do mesmo (ver **Figura 17**).

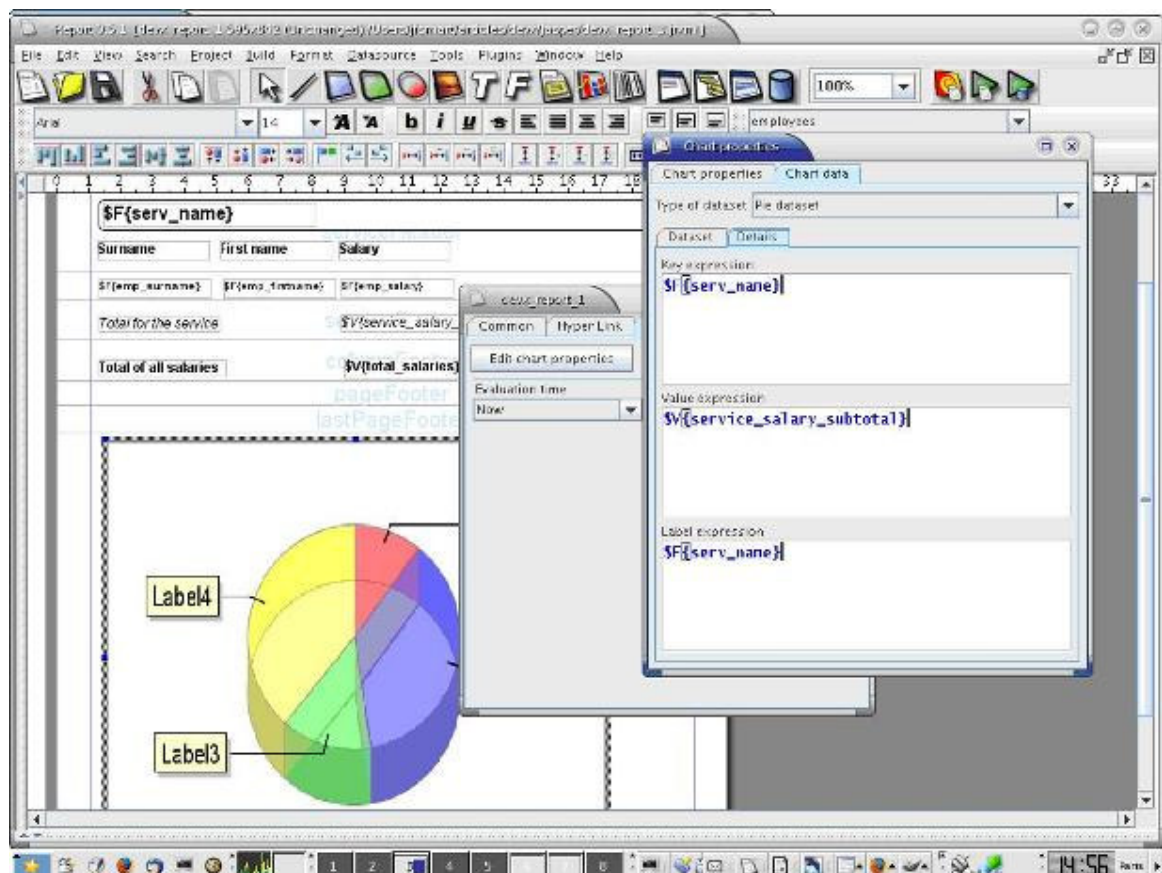


Figura 16. Chart Design

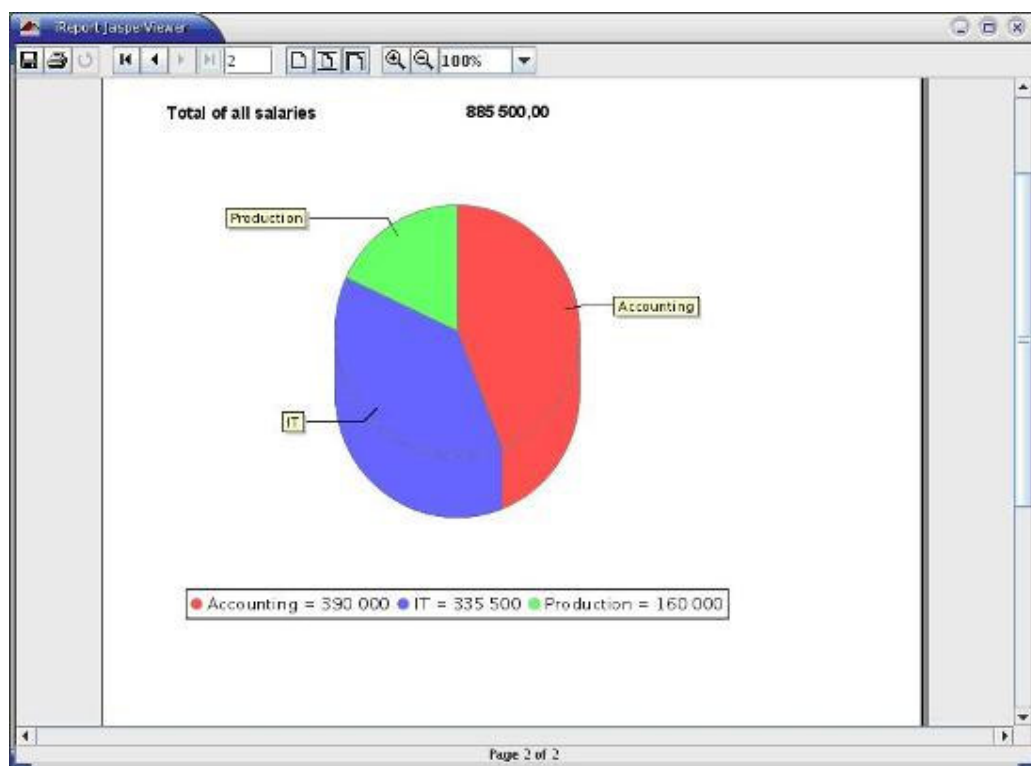


Figura 17. Chart Report

Usando o JasperReports do Java

Então, agora temos um gerador de relatórios JasperReports funcional. Como usá-lo dentro de aplicações Java?

O objeto `JasperDesign`, no pacote `net.sf.jasperreports.engine`, é a representação Java de relatórios XML projetados usando o `iReport`. Carrega-se o XML report e compila-se para um objeto `JasperReport` que realiza a geração do relatório:

```
JasperDesign jasperDesign
    = JasperManager.loadXmlDesign("MyReport.xml");
JasperReport jasperReport
    = JasperManager.compileReport(jasperDesign);
```

Em uma aplicação real, não deveríamos fazer isto toda vez que geramos um relatório, pois a geração consome tempo e o relatório pode ser facilmente armazenado em cache.

Uma vez que tivermos um relatório compilado, podemos alimentá-lo com dados e usá-lo para gerar relatórios.

Ao gerarmos o relatório, podemos fornecer parâmetros de tempo de execução via `Map`. Isto é útil para fornecer informações desconhecidas em tempo de projeto, tal como um título de relatório personalizado pelo usuário. Dentro do relatório `JasperReport`, declaramos o parâmetro na janela "View/Report Parameters" e a seguir usamos a variável de parâmetro da mesma maneira que faríamos com os outros campos e variáveis previamente vistos:

```
// Parametros de relatório em tempo de execução
Map parameters = new HashMap();
parameters.put("title", "A user-customized title");
```

É claro que precisaremos também fornecer uma conexão JDBC válida para o banco de dados alvo:

```
// Recuperar a conexão com o banco de dados
Connection conn = DBConnectionFactory.getConnection();
```

Finalmente, usamos a classe `JasperFillManager` para combinar o modelo de relatório compilado com os dados entrantes e gerar um relatório pronto para impressão:

```
JasperPrint jasperPrint
    = JasperFillManager.fillReport(jasperReport,
                                   parameters,
                                   conn);
```

Agora usamos o `JasperPrintManager` para gerar o relatório em qualquer formato desejado. O `JasperReport` suporta muitos formatos: PDF, Excel, XML, HTML, CVS, etc. Mas por enquanto, apenas escreveremos o relatório em um arquivo PDF:

```
JasperExportManager.exportReportToPdfFile(jasperPrint,
                                           "report.pdf");
```

Existem muitas outras possibilidades. Veja as API JasperReports para maiores detalhes.

Outras Ferramentas Geradoras de Relatórios

Há disponíveis várias outras ferramentas geradoras de relatórios, portanto, em que pé o JasperReports/iReport se encontra em relação a elas? A seguir, alguns dos atores principais no campo:

O Eclipse BIRT é uma nova e promissora ferramenta para projeto e geração de relatórios, com um plugin Eclipse agradável. Embora relativamente nova, tem algumas funcionalidades poderosas de projeto e geração de relatórios. Por outro lado, é menos madura do que o JasperReports e parece também menos bem integrada com o Java, pois depende de scripting JavaScript interno para otimização dos relatórios. A integração de conexões com datasources que não sejam JDBC puros, parece ser também complicada. Não obstante, vale a pena dar uma olhada. O Business Objects/Crystal Reports é uma solução BI/reporting comercial poderosa, com um desenhista gráfico experto. Uma licença Crystal Reports Server XI Edition (20 usuários) custa aproximadamente \$7,500. O Crystal Reports IX Developer Edition, uma versão mais leve e mais orientada para desenvolvimento de aplicações Web, está disponível por aproximadamente \$595 por desenvolvedor. A integração com o Java, parece porém, ser também limitada, mesmo na sua mais recente versão.

Um Duo Dinâmico

O JasperReports é uma ferramenta para geração de relatórios poderosa e flexível, fácil de integrar em um ambiente Java. O iReport elimina muito do trabalho difícil de projetar relatórios com o JasperReports — sem comprometer o seu poder. Juntos, formam um par impressionante. Experimente!

Sandro Miguel

Coordenador de Laboratório

UNAMA – Educação para o Desenvolvimento da Amazônia