<u>COMO MONTAR UM RELATORIO QUE CONTENHA SUB-</u> <u>RELATORIO UTILIZANDO UM DATA-SOURCE DO TIPO</u> <u>ARRAYLIST</u>

Recentemente me deparei com um problema. Precisava montar um relatório e que neste relatório precisava montar um sub-relatorio.

Vamos supor....

No relatório principal eu preciso colocar informações de um titular e no sub-relatorio eu precise colocar os dependentes, e seus respectivos dados, deste titular A estrutura era mais ou menos a seguinte:

Dados do Titular Nome Titular: xxxxxxxxxxxxxxx CPF Titular: 111111111 Data Nascimento: 99/99/9999

Dados dos Dependentes

RG Dependente	Data Nascimento
77777777	11/11/1111
22222222	22/22/2222
5555555	66/66/6666
	G Dependente 77777777 22222222 55555555

Eu tinha que mostrar alguns dados do Titular e Todos os seus dependentes, juntamente com seus dados...

Mas o meu problema era o seguinte. Eu não podia fazer um select dentro do sub-relatorio. Eu tinha que, de alguma forma, passar os dados do titular e dos dependentes para o relatório e este apenas mostrar estes dados.

Aí que vinha o meu problema....

Bem. Depois de muito encher as paciências de um amigo e lotar sua caixa postal com muitas duvidas, ler diversos tutoriais, participar de muitos fóruns de discussão, consegui montar meu relatório. FINALMENTE.

Esse meu amigo em questão é o Roberto Furutani. Ele montou um EXCELENTE tutorial a respeito de sub-relatorios utilizando ArrayList. Mas de alguma maneira, não me pergunte o porque, não deu certo comigo. Mas serviu de base para eu solucionar meu problema. Vc pode ver o tutorial montado por ele no seguinte link: http://geocities.yahoo.com.br/robertofurutani/java/Tutorial_JasperReports/

Enfim....vamos largar mão de conversinha e vamos logo ao que interessa....

ATENÇÃO: Estou considerando que o usuário já tenha algum domínio e experiência sobre como construir relatórios no iReport. Questões básicas sobre iReport, Jasper, Classes, etc.... www.google.com/ ajuda muito!!! :D

Suponhamos que eu tenha duas classes: Titular e Dependente

Os Atributos da classe Titular são os seguintes: CPF, Nome, DataNascimento

Os atributos da classe Dependente são os seguintes: CPFT, (CPF do titular) Nome, DataNascimento, RG

Na minha aplicação tinha um método que, ao se passar o CPF do Titular como parâmetro, me devolvia um ArrayList contendo os Objetos Dependentes relacionados à esse titular. Era aí que estava meu problema...como passar este ArrayList para o relatório JASPER e ainda assim imprimir estes atributos...

Bem...Sabemos que o método que preenche o relatório é o JasperFillManager.fillReport, que possui uma serie de métodos com o mesmo nome(polimórficos) e mudando os seus argumentos. Eu usei este: <u>fillReport(JasperReport</u> jasperReport, java.util.Map parameters,

JRDataSource dataSource)

Para o primeiro parametro vc declara um JasperReport padrão...sem maiores problemas até aqui. Para o Segundo parametro eu usei uma HashTable contendo os dados do Titular como chave, dessa forma: Hashtable ht = new Hashtable(); //coloque os valores como String para ficar mais facil de manipular ht.put("CPF",titular.getCPF()); ht.put("NOME_TITLU_AP" titular.getNome());

ht.put("NOME_TITULAR",titular.getNome()); ht.put("DATA NASCIMENTO TITULAR",titular.getDataNascimento());

Bem...Vc deve criar um relatório, o principal (ou Master), e colocar nele os campos que vc quer(referentes apenas ao Titular).

No seu relatório Principal, ou master, (no iReport), onde vc vai imprimir os valores do titular, vc cria os mesmos campos, do tipo PARAMETER, para cada chave da sua Hashtable, que acabamos de criar algumas linhas acima, como na figura abaixo. Reparem que os campos do relatório(PARAMETER) devem ter o mesmo nome das chaves da sua Hashtable senão dá PAU !!!

	Maste	r values			
	Fields	Variables	Parameters	;	
		Parameter na	ame	Class type	1
ᅓ닪;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;	REPORT	PARAMETER	S_MAP	java.util.M	t
	REPORT	CONNECTIO	N .	java.sql.C	1
	REPORT	DATASOUR	E	net.sf.jas	1
	REPORT	SCRIPTLET	}	net.sf.jas	1
	REPORT	LOCALE	3	java.util.L	1
	REPORT	RESOURCE	_BUNDLE	java.util.R	1
Dados do Titular:	REPORT	MAX_COUNT	- 1 - 1 - 1 - 1 - 1 - 1 - 1 - 1 - 1 - 1	java.lang	1
Nome do Titular: \$P{NOME TITULAR}	CPF			java.lang	1
CPF do Titular: \$P{CPF}	NOME_TI	TULAR		java.lang	1
Data de Nascimento: \$P{DATA NASCIMENTO TITULAR}	DATA_NA	SCIMENTO_1	TTULAR	java.lang	1

Para o terceiro parametro do método eu declarei um JREmptyDataSource, como segue:

JREmptyDataSource ED = new JREmptyDataSource();

Pronto. Agora nós já temos os três parâmetros para o método fillReport. É só usar!!!

Legal né?!?!??! :D

Mas vc deve estar se perguntando.... "Onde que o carinha passou o ArrayList contendo os objetos Dependente pra eu mostrar no relatório ????"

Legal....Isso mostra que vc está prestando atenção...

Bem...Agora que vem a parte legal do relatório.

Para criar um relatório com sub-relatorio desta maneira como estamos tentando fazer, precisamos criar 2 relatorios JRXML no iReport. Um para o relatório principal (Master) e outro para o sub-relatorio (Detail). Isso porque um relatório com sub-relatorio nada mais eh que um relatório maior com um outro relatório dentro dele....

Até agora nós já criamos o relatório Principal, ou Máster. Precisamos criar o sub relatório (Detail) onde será mostrados os Dependentes e seus respectivos dados.

Crie um novo arquivo no iReport e coloque os campos (agora do tipo FIELDS) da maneira que vc gostaria que eles aparecessem(layout) no seu relatório principal, como na figura abaixo:



Reparem que : Os nomes dos campos(FIELDS) devem ter os mesmos nomes dos atributos da Classe Dependente. Os nomes têm que ser idênticos e os campos de tipos compatíveis (String, int, etc...). Senão já sabem. dá PAU!!! Vejam que agora eles não têm nada a ver com a Hashtable. Isso porque vc irá passar como parametro para o sub-relatorio o seu ArrayList contendo os objetos da Classe Dependente. Pronto...não precisa colocar mais nada no seu sub-relatorio(Detail). Somente estes campos(repito, do tipo FIELD).

Agora vamos voltar ao relatório Principal (Master).

Na área onde vc deseja que o sub-relatorio(Detail) apareça vc deve colocar um objeto do tipo sub-report tool, como na figura abaixo:

Celso Fernandes de Oliveira Junior - teteorp@gmail.com

95×8	342	(Uno	han	ged)	C:\CE	LSO\.	JAVA\T	UTORIA	IS\Tu	torial 9	5ub Re	eport	Cels	o\M	aster.	jrxml]	
Projec	t E	Build	Fgri	mat	Datas	ource	Tools	Plugin:	<u>s W</u> in	dow	Help						
][D	1	8	/	D	D		Di	Ţ);					Ð	E		0
3	9									~							
		•	4	-	A 3	A	b i	<u>u</u> -	E	= 3	63						
ញ្	<u>51</u>	₽	뫼	++	el e	1	E 6		••• •••		+()∔(] ₩[1⇔:	Ŧ	<u><u><u></u></u><u></u><u></u><u></u><u></u><u></u><u></u><u></u><u></u><u></u><u></u><u></u><u></u><u></u><u></u><u></u><u></u><u></u></u>	- <u>¥</u> o	= 1
1	ç	. 1	. ?		3 . 4	. 5	. 6	7.8	3 9	. 10	11	12	13	1,1	15	16	7 . 18
Sec. 1																	
			Dad	os di) Titu a	C .											
			Νоπ	ne do	Titu a	r: \$	P{NOME	E TITULA	AR}								
			CPF	do 1	'itular:	\$	P{CPF}										
			Data	a de l	Vascim	ento:	\$F{DAT	TA NASC	IMENT	D TITJ	LAR}						
			Dad	os di	os Dep	endeni	tes:		1.16	8-965			52.0				
			Norr	ne Da	pence	nte:	>	RG D	epende	nte:	5	-6	ata N	lascin	ento:	\supset	
1		-	E	7									_				1
1	-																Ì
		1	-											1000	-	-	

Reparem que os campos circulados em Azul são apenas Labels, ou estáticos. Já o campo em vermelho é o sub-report tool.

Agora que vem o pulo do gato!!!!

Voltemos ao seu código-java.

Suponhamos que vc já tenha o seu ArrayList montadinho. O que vc vai fazer com ele???? Vc vai declarar um objeto do tipo JRBeanCollectionDataSource e construí-lo com o seu ArrayList como parametro, como segue o código abaixo.

JRBeanCollectionDataSource BCD = new JRBeanCollectionDataSource(deps);

Onde deps eh o meu objeto ArrayList contendo os objetos Dependente.

Legal...Mas vc deve estar se perguntando de novo: "Mas onde que esse cara passou esse JRBeanCollectionDataSource para o relatório, sendo que o terceiro parametro do método fillReport, um cara que implemente a interface JRDataSource, ele já colocou um JREmptyDataSource????"

Agora que vem a parte legal....vc vai colocar esse JRBeanCollectionDataSource na sua Hashtable que vc vai passar como segundo parametro do seu método fillReport.

COMO????

Assim:

ht.put("deps",BCD);

Aí, na sua lista de PARAMETERS do seu relatorio principal (MASTER) vc vai colocar mais um parametro. Vc vai colocar o "deps". Veja a figura abaixo. AH.... reparem que ele deve ser do tipo Object e ter o mesmo nome da chave da sua Hashtable.

CPF do Titular: \$P{CPF} Data de Nascimento: \$P{DATA NASCIMENTO TITULAR} Dados dos Dependentes: Nome Dependente: RG Dependente: Data Nascimento:	Nome do	Titular:	\$P{NOME	E TITULAR}					
Data de Nascimento: \$P{DATA NASCIMENTO TITULAR} Dados dos Dependentes: Nome Dependente: RG Dependente: Data Nascimento:	CPF do T	itular:	\$P{CPF}						
Dados dos Dependentes: Nome Dependente: Data Nascimento:	Data de N	lascimento:	: ₿P{DAT	TA NASCIME	NTO TITULAR				
Dados dos Dependentes: Nome Dependente: Data Nascimento: Nome Dependente: RG Dependente: Data Nascimento:									
Nome Dependente: Data Nascimento:	Dados do	<u>s Depender</u>	ntes:						
						Data Ma			
	Nome De	pendente:		RG Deper	ndente:	Data Na	scimento	8 I.	
	Nome De	pendente:	1	RG Deper	ndente:	Data Na	scimento	10	-
	Nome De	pendente:		RG Deper	ndente:	Data Na	scimento		
🔁 Master values	Nome De	pendente:		RG Deper	ndente:	Data Na	scimento		
Fields Variables Parameters	Nome De	pendente: :r values		RG Depei	ndente:	Data Na	scimento	•]
	Nome De Maste	pendente: :r values Variabl	Ies P	RG Deper	ndente:	Data Na	scimento		
	Nome De Maste	pendente: :r values Variable	 Ies Pi	RG Deper	ndente:		scimento		
Parameter name Class type Is for Built-in	Nome Dep Maste	pendente: :r values Variabl Paramete	les P er name	RG Deper	ndente: Class type	Is for	Built-in	·)
Parameter name Class type Is forBuilt-in New CPF java.lang.String no no	Nome De Maste	pendente: :r values Variabl Paramete	ies P er name	RG Deper	Class type	Is for	.Built-in		New
Parameter name Class type Is for Built-in	Nome De Maste	endente: r values Variable Paramete	 les P er name	RG Deper	Class type	Is for	Built-in	 [] r]
Parameter name Class type Is for Built-in New	Nome Dep Maste	pendente: er values Variable Paramete	 Ies Pa er name	RG Deper	Class type	Is for	Built-in		New
Parameter name Class type Is for Built-in CPF java.lang.String no no	Nome De Maste	pendente: er values Variable Paramete	·· les P er name	RG Deper arameters	Class type ava.lang.String	Is for	.Built-in		New

Agora que vem a jogada legal do relatório....E onde eu tomei por base o tutorial do Roberto Furutani. É nesse momento que o relatório principal (MASTER) passa o ArrayList para o sub-relatorio(DETAIL).

Voltando ao Sub-report tool ...

Dê um duplo click no seu objeto sub-report tool. Na Aba SubReport vc vai colocar "Use datasource expression"

Celso Fernandes de Oliveira Junior - teteorp@gmail.com

E no campo de baixo, vc coloca $P\{deps\}$, ou seja, o seu parametro deps. Veja a figura:

	Parameters Man Expression
Dados do Titular:	1
Nome do Titular: \$P{NI	
CPF do Titular: \$P{CI	
Data de Nascimento: \$P{	
Dados dos Dependentes:	Connection / Datasource Expression
Nomo Dopondonto:	
	\$P{deps}
	the set

Ainda na mesma janela, agora na Aba Subreport(Other), vc vai colocar o caminho absoluto(C:\diretório\detail.jasper, por exemplo) do seu sub-relatorio compilado (.JASPER) No exemplo, meu sub relatório compilado se chama Sub.jasper Veja a figura:

- Concer			×
Common Suk	preport Subreport	(Other) All	_
Image Expressio	n Class	10 - 82	
java.lang.String			
Subreport Expre	ssion		
1			
Subreport param	ieters		
Subreport param Para	eters Imeter	Expression	

Pronto. Agora acabou. Basta vc compilar os relatórios e montar seu código...

Segue um trechinho do meu código servlet de exemplo:

```
....
....
//Supondo a classe ja construida com seus metodos
Titular titular = new Titular(12345678911);//Cosntrutor da classe Titular, passando o CPF
ArrayList deps = new ArrayList(titular.buscaDependentes(12345678911)); //ArrayList com
objetos da Classe Dependente
Hashtable ht = new Hashtable():
ht.put("CPF",titular.getCPF());
ht.put("NOME_TITULAR",titular.getNome());
ht.put("DATA_NASCIMENTO_TITULAR",titular.getDataNascimento());
try
{
  JREmptyDataSource ED = new JREmptyDataSource();
  JRBeanCollectionDataSource BCD = new JRBeanCollectionDataSource(deps);
  ht.put("deps",BCD);
  InputStream jp = getServletContext().getResourceAsStream("/Master.jasper");
  JasperReport relatorio = (JasperReport) JRLoader.loadObject(jp);
  JasperPrint impressao = new JasperPrint();
  impressao = JasperFillManager.fillReport(relatorio,ht,ED);
  byte[] buffer = JasperExportManager.exportReportToPdf(impressao);
  if (buffer != null)
   {
   ServletOutputStream outputStream = p_response.getOutputStream();
   outputStream.write(buffer, 0, buffer.length);
   outputStream.flush();
   outputStream.close();
   ł
}
catch(Exception e1)
  System.out.println("Erro ao gera relatorio! "+e1.toString());
  return:
}
return;
```

Ok...

Acho que agora vai dar tudo certo.

Bem, essa foi a maneira que eu achei de resolver meu problema.....funcionou legal.... Se vc quiser colocar mais sub-relatorios..siga o mesmo raciocínio...Crie mais ArrayList, crie um JRBeanCollectionDataSource com o ArrayList como parametro e coloque o JRBeanCollectionDataSource na Hashtable...

Espero que tenha ajudado.