

快速报表系统 4.0

用
户
使
用
手
册

杰安软件工作室

第一章 设计	7
1. 1、控制键	9
1. 2、鼠标操作	9
1. 3、工具栏	10
1. 3. 1、设计模式工具栏	10
1. 3. 2、“标准”工具栏	10
1. 3. 3、“文本”工具栏	11
1. 3. 4、“边筐”工具栏	12
1. 3. 5、“对齐”工具栏	13
1. 4、设计选项	13
1. 5、报表设置	15
1. 6 页面设置	16
第二章 创建报表	19
2. 1、报表对象	20
2. 2、“世界，你好！”报表事例	20
2. 3、“Text”组件	21
2. 4、在“Text”组件中使用HTML标记	23
2. 5、通过“Text”组件显示公式	24
2. 6、FastReport中的Bands	25
2. 7、Data Band	26
2. 8、TfrxDBDataSet组件	27
2. 9、“客户列表”报表	27
2.10、通过text组件显示数据表字段	29
2.11 别名	30
2. 12、变量。	30
2. 13、“Picture”控件	32
2. 14、图形报表	33
2. 15、多行文本显示	34
2. 16、文本拆分	36
2. 17、组件的Wrap	37

2. 18、显示数据表中的数据	38
2. 19、标签式打印	40
2. 20、子bands	42
2.21、组件的移动	43
2.22、两个数据阶的报表（主—细）	43
2.23、页首和页尾数据Band	46
2.24、多页报表	47
第三章 分组集合体	50
3. 1、分组打印	51
3. 2、其他分组特性	53
3.3 页码的重设	55
3. 4、组的操作	55
3. 5、行数	56
3. 6、函数集	57
3. 7、页和报表的统计	60
3. 8、插入汇总函数	61
第四章 格式化、加强	62
4. 1、格式化输出结果	63
4. 2、内嵌格式化	63
4. 3、条件显示	65
4. 4、分行显示数据行的颜色	66
第五章 嵌套报表	68
5. 1 嵌套报表	69
5. 2、设计子报表	69
5. 3、子报表中的限制	69
5. 4、PrintOnParent选项	70
第六章 脚本	72
6. 1、体验脚本语言	73
6. 2、脚本结构	76

6. 3、“世界你好!” 脚本	78
6. 4、脚本中组件对象的使用	79
6. 5、调用报表变量列表中的变量。	79
6. 6、调用数据表子段	80
6. 7、脚本中调用汇总函数	80
6. 8、报表中变量值的显示	80
6. 9、事件	81
6. 10、一个使用 “OnBeforePrint” 事件的例子	82
6. 11、在组头打印组的汇总信息.....	84
6. 12、“OnAfterData” 事件.....	88
6. 13、 Service 组件.....	89
6. 13. 1、 Report 组件.....	89
6. 13. 2、 Engine 组件	90
6. 13. 3、“ OutLine ” 组件	91
6.13.4、引擎组件的使用	91
6.15 Anchors	94
6.16 “ Outline ” 组件的使用方法	96
6.17 “ OnManualBuild ” 页面事件	98
6.18 脚本中的组件的建立	104
第七章 交叉报表	106
7. 1、创建交叉报表	108
7. 2、改变显示	110
7. 3、使用函数	112
7. 4、对结果进行排序	112
7. 5、组合标题的表格	113
7. 6、调整单元格的宽度	115
7. 7、字体颜色和突出显示	117
7. 8、使用脚本语言管理组件	118
7. 9、调整行和列的大小	123
7. 10、手动填充表格单元	124

7. 11、在表格单元中加入其他组件.....	126
7. 12、一些有用的设置	129
第八章 制图表.....	132
8. 1、chart数据中数字的限制	137
8. 2、设置	137
8. 3、指定数字制表	138
8. 4、利用脚本进行制表	139
8. 5、在delphi环境中创建的报表的打印	139
第九章 点阵报表.....	140
9. 1、点阵报表使用交叉报表	144
9.2、点阵报表的打印	145
9. 3、命令组件	146
第十章 对话框窗体.....	147
10. 1、控件	148
10. 2、“世界你好!” 报表	149
10. 3、输入参数, 并传递到报表中	150
10. 4、组件的交互	150
10. 5、多个对话框表单	151
10. 6、对话框窗体的管理	152
第十一章 数据访问组件.....	155
11. 1、组件的描述.....	156
11. 1. 1、TfrxDBLookupCombobox组件.....	157
11. 1. 2、TfrxADOTable组件	157
11. 1. 3、TfrxAdoQuery组件.....	159
11. 1. 4、TfrxADODatabase组件	161
11. 2、创建报表.....	161
11. 3、简单的列表式报表.....	162
11. 4、参数查询报表.....	163
11. 5、其他可用配置.....	164

第十二章 报表的继承性	166
12. 1、创建报表	167
12. 2、修改基础模板	169
12. 3、组件的继承	170
第十三章 报表向导	171
13. 1、新报表向导	172
13. 2、数据连接向导	175
13. 3、新table向导	176
13. 4、新query向导	177
13. 5、查询语句生成	177
第十四章 报表的预览 打印 导出	180
14. 1、控制键	182
14. 2、鼠标控制	182
14. 3、报表的打印	183
14. 4、报表中的文字搜索	183
14. 5、报表的导出	184

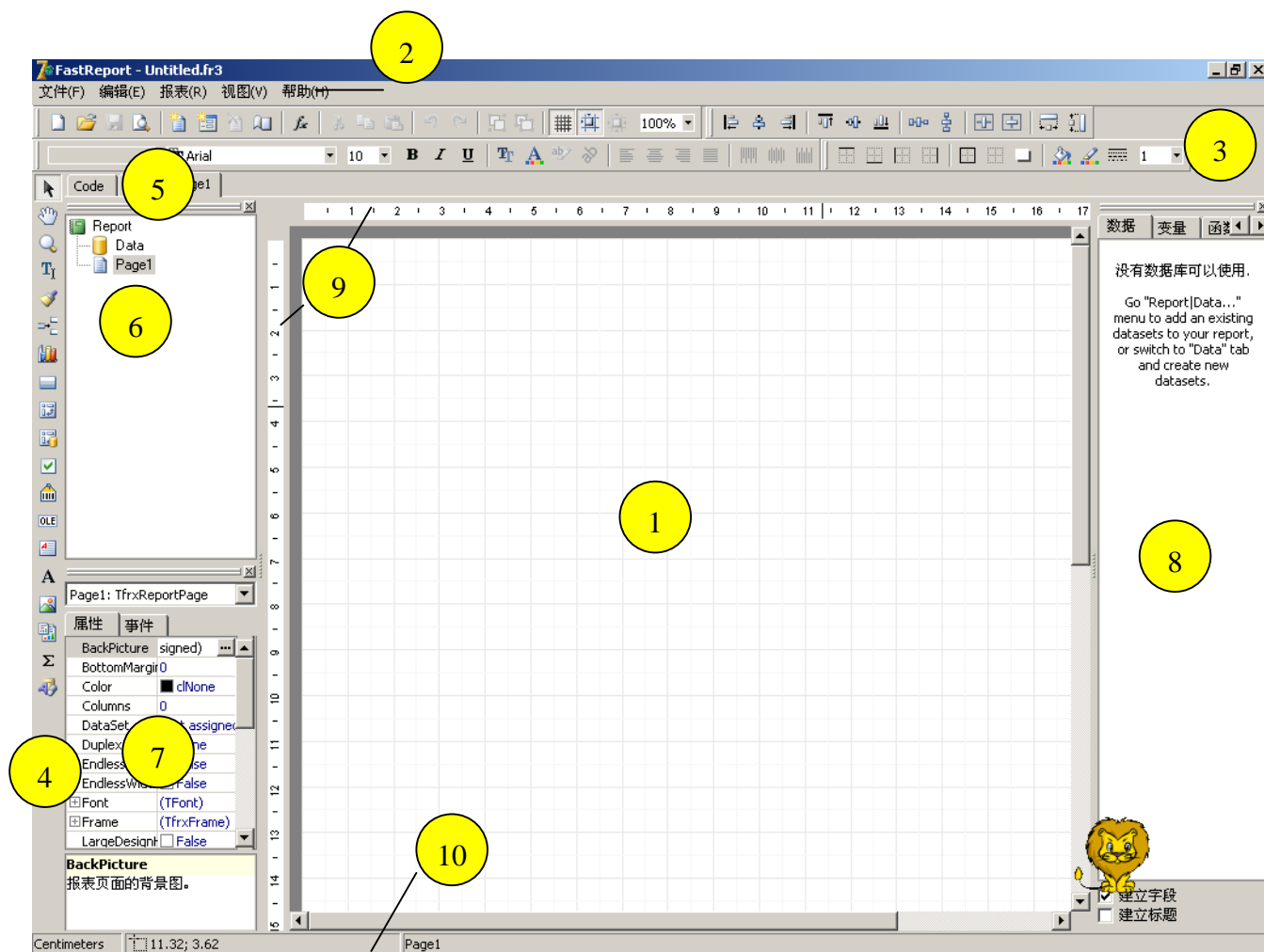
第一章

设计

报表组件在开发环境中，可以通过鼠标双击TfrxReport组件，打开报表设计器环境。设计器窗口提供给用户全部有关报表设计的工具，并同时提供预栏功能。报表设计器界面新颖。它包含有几个工具栏，并且工具栏可以停靠镶嵌在任何你想停靠的地方，并随设计器其他配置属性保存在了一个.Ini文件中，当再次打开时，恢复设计器的配置属性。

为了提供给用户运行期设计报表的功能，应该再添加“TfrxDesigner”或“frxDesign”单元到Uses列表中。这样用户就可以在运行期修改报表格式，又可以修改已经生成的报表内容信息。

注：根据报表的需要，你应该还需要添加其他的一些组件到表单上。其他组件说明不在此介绍。



图中标号说明：

- 1——报表设计区域
- 2——菜单栏
- 3——工具栏
- 4——报表对象工具栏

- 5——报表分页标签
- 6——报表树窗口
- 7——对象属性窗口
- 8——数据树窗口。可以从此窗口拖曳到报表窗口
- 9——尺标。
- 10——状态条

1. 1、控制键

控制键	描述
Ctrl+O	“文件 打开...” 菜单命令
Ctrl+S	“文件 保存” 菜单命令
Ctrl+P	“文件 预览” 菜单命令
Ctrl+Z	“编辑 撤销” 菜单命令
Ctrl+C	“编辑 复制” 菜单命令
Ctrl+V	“编辑 粘贴” 菜单命令
Ctrl+X	“编辑 剪切” 菜单命令
Ctrl+A	“编辑 全选” 菜单命令
Arrow, Tab	对象切换
Del	删除被选择的对象
Enter	打开对象编辑器
Shift+arrows	改变对象的大小
Ctrl+arrows	移动对象位置
Alt+arrows	移动对象到附近对象的位置

1. 2、鼠标操作






操作	描述
左键	选择组件对象；添加新的组件对象；移动组件对象；改变组件的大小； 对选中的对象通过组件的黑色方块可以改变组件的大小。
右键	弹出选择组件对象的浮动菜单

双击	打开对象编辑器；通过在组件的中间位置双击鼠标左键，打开属性对话框窗口。
鼠标滚轮	滚动报表。
Shift+左键	多选组件对象
Ctrl+右键	当你按着鼠标左键移动鼠标时，在窗口上划出一道方形窗口。松开鼠标按键，则在方形窗口中的组件都被选中。在被选中组件的中间位置，按住鼠标左键移动组件对象到相应的位置。
Alt+左键	如果被选中的组件是“TEXT”，则组件处于输入状态。

1. 3、工具栏


1. 3. 1、设计模式工具栏

组件工具栏和以下工具按钮。

图标	名称	描述
	对象选择	标准模式下，选中对象，鼠标箭头方向改变组件大小。
	手	单击图标，托动报表窗口。
	缩放	单击鼠标左键，可以对当前窗口放大两倍（增加 100%）；右键单击鼠标，可以缩小当前窗口。如果鼠标左键托划，则选择范围被放大。
	文本编辑	单击“TEXT”组件对象，允许编辑报表内容。当按着鼠标左键在空白处托划，松开鼠标键则在被选择范围产生一新的“Text”组件对象。
	格式刷	当“Text”对象被选中后，此按钮变成可操作状态。鼠标左键单击“Text”对象时，则拷贝格式，再单击“Text”组件对象时则复制格式到对象上。

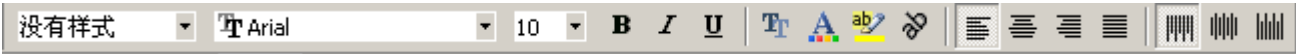
1. 3. 2、“标准”工具栏
















图标	名称	描述
	新报表	创建新的空白报表

	打开报表	打开已有的报表。快捷键：Ctrl+O;
	保存报表	保存报表到文件中。快捷键：Ctrl+S;
	预览	生成报表并进行预览。快捷键：Ctrl+P;
	新页	添加一新页到报表中。
	新对话框	添加一新的对话框表单到报表中。
	删除页	删除当前页。
	页面配置	打开页面属性配置窗口。
	变量	打开报表变量编辑器
	剪切	剪切当前选择对象到粘贴板中。快捷键：Ctrl+X
	复制	复制当前选择对象到粘贴板中。快捷键：Ctrl+C
	粘贴	粘贴复制到粘贴板中的数据到报表中。快捷键：Ctrl+V
	取消	撤销最后的操作。快捷键：Ctrl+Z
	重复	重复最后的取消的操作。快捷键：Ctrl+Y
	群组	对选择的组件对象组合成一个整体
	取消群组	将组合整体的组件分开成单个组件。
	显示栅格	在页上是否显示栅格。栅格大小可以在视图——选项窗口中进行设置。
	栅格对齐	当移动组件或改变组件大小时，自动改变组件的大小对齐到栅格。
	适配栅格	改变组件大小时，组件自动和上个大小对齐。
	缩放	设置界面缩放大小。

1. 3. 3、“文本”工具栏














图标	名称	描述
	样式	允许选择一个样式。定义样式列表，可在“报表 样式...”菜单栏。
	字体名称	允许从下拉筐中选择字体名称，保存最后应用五个可用的字体列表。

10 ▾	字体大小	允许从字体大小下拉列表筐中选择字体大小。字体大小也可以用户输入。
B	粗体	设置/取消字体粗体
<i>I</i>	斜体	设置/取消字体斜体
<u>U</u>	下划线	设置/取消字体下划线
	字体设置	显示字体设置对话框。
	字体颜色	从下拉筐中选择字体颜色。
	高亮显示	显示高亮显示文本属性对话框。
	文本方向	选择字体显示方向。
	左对齐	文本左对齐
	水平居中	文本水平居中
	右对齐	文本右对齐
	平均居中	文本自适应组件宽度
	上对齐	文本上对齐
	垂直居中	文本垂直居中
	下对齐	文本下对齐

1. 3. 4、“边筐”工具栏















图标	名称	描述
	上边线	设置 取消上边筐线
	下边线	设置 取消下边筐线
	左边线	设置 取消左边筐线
	右边线	设置 取消右边筐线
	边筐	设置边筐线
	没有变筐	取消边线
	阴影	设置 取消阴影

	背景颜色	从下拉筐中选择背景颜色
	边筐颜色	从下拉筐中选择边筐颜色。
	外筐样式	从下拉筐中选择线条样式
	边筐宽度	从下拉筐中选择线条的宽度。

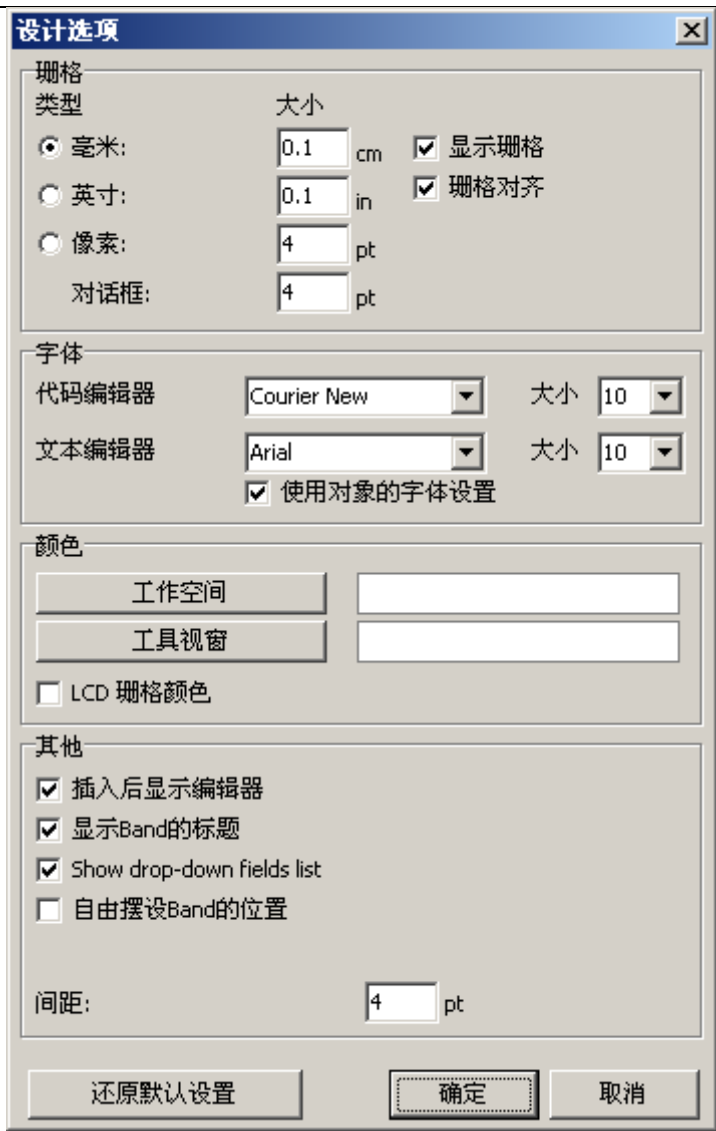
1. 3. 5、“对齐”工具栏



图标	名称	描述
	左对齐	组件左边对齐
	水平居中	组件水平中间对齐
	右对齐	组件右边对齐
	上对齐	组件上边对齐
	垂直居中	组件垂直中间对齐
	下对齐	组件下边对齐
	水平均分	组件位置水平均分
	垂直均分	组件位置垂直均分
	水平居中	组件水平居中
	垂直居中	组件垂直居中
	相同宽度	组件宽度设置成最大
	相同高度	组件高度设置成最高

1. 4、设计选项

通过“视图|选项”菜单条设置设计器选项。



在这设置需要的单位（毫米、英寸、像素）。并设置栅格大小，也可以双击状态栏中左边地一个格，也是设置这一项。

你也可以设置单个是否显示，和栅格对齐。也可以通过“标准”工具栏中的按钮进行设置。你可以设置代码编辑器和文本编辑器的字体名称、大小、颜色等。如果“使用对象的字体设置”被选中，则文本编辑器中的字体被“Text”组件的字体代替。

设计器工作区默认颜色为白色，并且可以通过工具空间和工具视窗改变颜色。

“LCD 栅格颜色”，增强栅格线对比度，并改进可视性。

“插入后显示编辑器”当增加新的组件时进行选项控制。如果被选中，增加新的组件时出现组件编辑器。当添加最大数量的组件时，此选项取消。

“显示 Band 标题”选项取消选中，则 Band 标题不被显示。

“Show drop-down fields list”选项，决定“Text”组件按钮关联数据时，当鼠标在上边移动时是否显示下拉筐显示可选择字段。

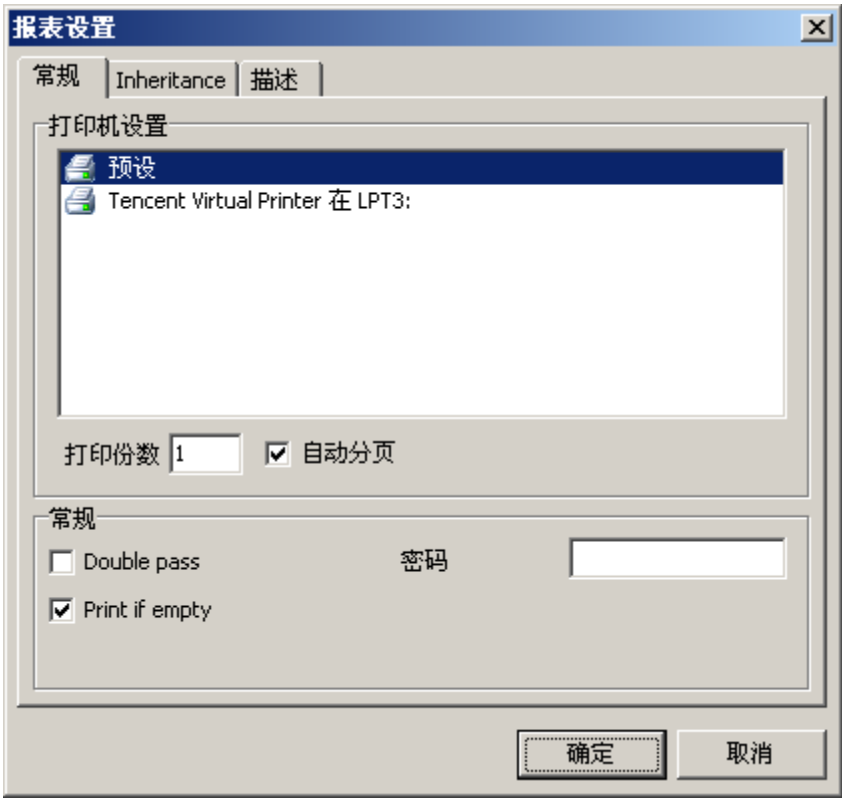
“自由摆置 band 位置”选项，决定是否粘连在报表页上。此选项默认不被选中，Band 可

以分组显示在报表页上，Band 之间的间隙可以通过“间距设置”设置。

1. 5、报表设置

通过“报表|选项”菜单条显示报表参数窗口，其一共有三个属性页。

第一页显示报表的一般性参数。



用户可以绑定报表到系统已安装的打印机中的一台。意思是报表打印时默认使用绑定的打印机输出打印。这一项当系统中有多个打印机时被使用。例如：文本可以帮定输出到单色打印机，有图形的可以输出到彩色打印机。打印机列表中“预设”打印机选项被选中，则报表不和任何打印机绑定，并且打印输出到系统默认得打印机上。

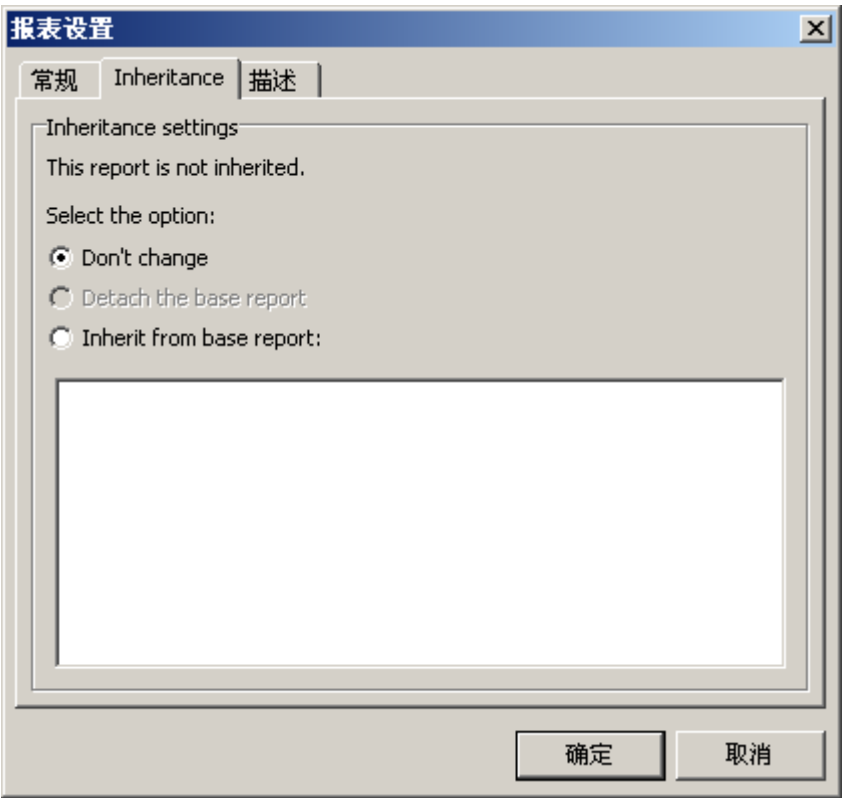
用户可以设置报表打印分数，打印份数显示在打印机选项的面板上。

如果“Double pass”选项被选中，报表的信息分为两项。第一阶段，报表形成，并分开到页中，但结果不保存。第二个阶段，标准报表信息保存到流文件中。

“print if empty”选项，决定报表空白时是否生成报表文件，如果选中，生成报表时显示空白报表。

“密码”项设置密码，当打开报表时需要输入密码。

第二页设置报表的继承性



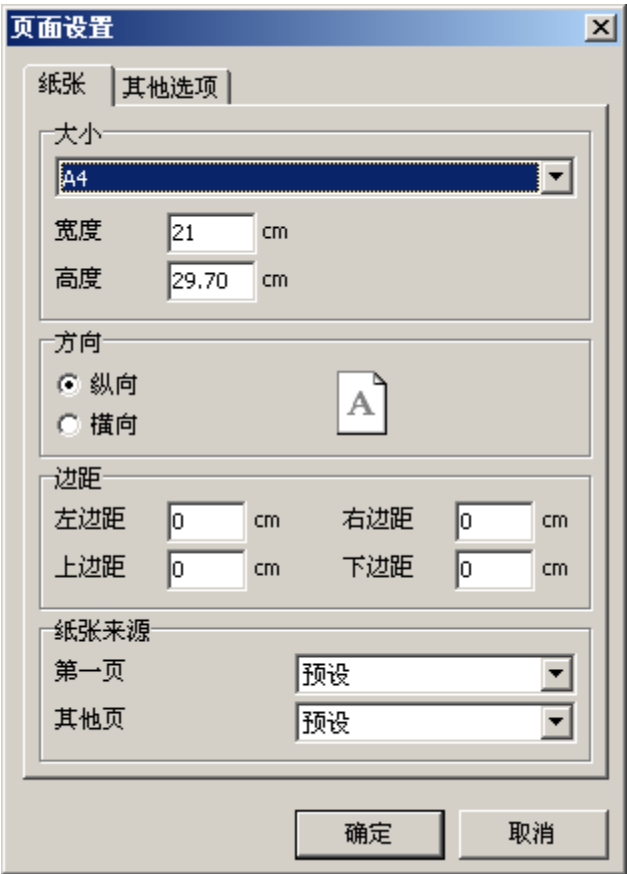
第三页用户设置报表属性描述



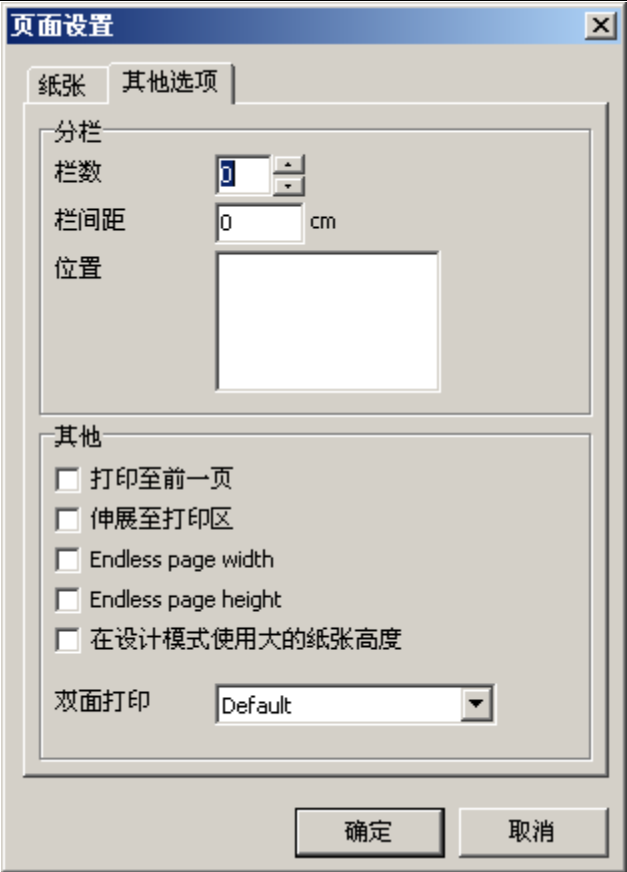
1. 6 页面设置

通过“文件|页面设置...”菜单条，或双击页面空白处，弹处页面设置对话框。这个对话框

共有两页：



在第一个页面上，用户可以设置纸面大小，和纸面方向，还有页面边距。
在纸张来源的下拉筐中选择第一页和其他页打印机的状态。



第二页设置分栏显示报表栏数、栏距和位置。当前设置在设计器中显示。

“打印至前一页”允许用户打印报表，从上一页的空白处开始。这个选项可用在一个报表有多个模板组成或批量打印时。

“伸展至打印区”选项打印奇页面时左右边界可以交换。

“Endless page width & height”选项根据多个数据报表增加页数。这样可以看到一个大型报表数据代替多个报表页面。


“在设计模式使用大的纸张高度”选项增加页面高度。


第 二 章

创建报表


2. 1、报表对象


一个空白报表就像一张白纸，在页面的任何部位都可以添加报表对象，以显示不同的信息（像文本/图形），像定义一个报表外貌。下面简单的描述一下报表组件业务，主要包含标准组件：


：“Band”对象在设计页中添加一 Band，在其设置区用户可以添加定义，依照 band 类型设置组件。

：“Text”对象，用于显示文本，在其组件范围显示一行或多行文本信息。


：“图形”组件用于显示“BMP”“JPG”“ICO”“WMF”“EMF”格式文件


：“制图”组件将数据通过不同的图例进行可视化形象化显示，如：饼形显示、柱形显示、曲线等。


：“检查筐”组件通过“√”或“×”显示两型数据。

：“OLE”组件用于通过“OLE”组件显示其它系统组件的数据。

：“Rich text”组件显示“RTF”格式的文本数据文件。

：“SubReport”组件用于在基础报表上添加一个另外的报表页。

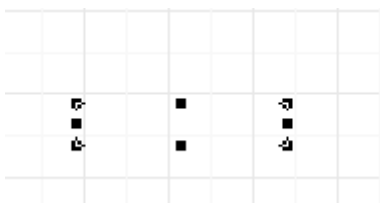
：“System text”显示几种信息（例如：日期、时间、页码等），还有像计算数据之和等。

：用于绘制不同的几何图形，如织线、斜线、矩形、圆形、椭圆、三角形、菱形等。

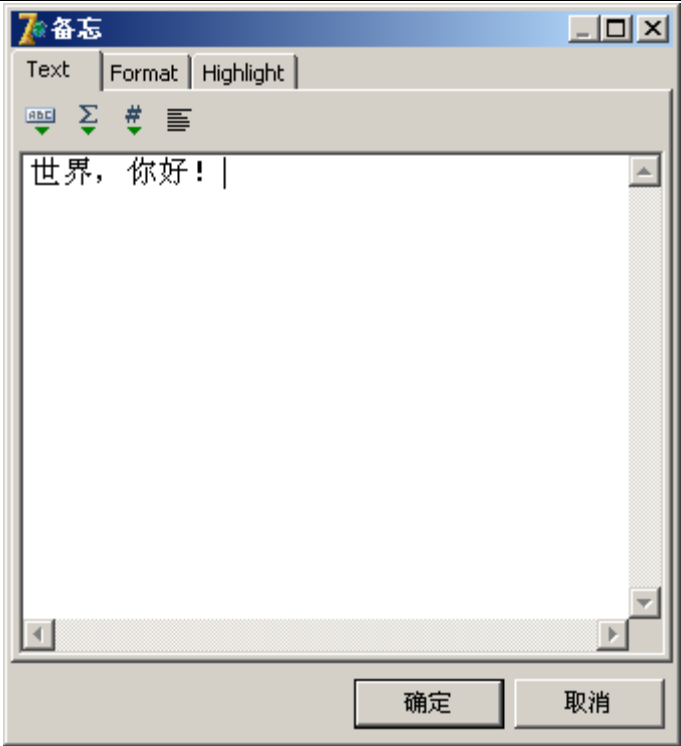
：“Barcode”组件在面板上显示不同的条形码。

2. 2、“世界，你好！”报表事例

报表只包含一个“世界，你好！”这样一个标题，打开报表设计器，在“object”设计面板中点击“Text”按钮，在报表页面上希望显示的地方，点击鼠标按键，则在相应位置生成一个组件。



文本编辑器立即出现，如果不出现（可以在设计器选项配置中进行设置），可以通过双击组件，显示文本编辑器窗口，输入“世界，你好！”文字，然后点击“确定”按钮。



报表设计完成，点击“文件|预览…”打开预览窗口，或在工具栏点击相应的预览按钮，则包含一个“世界，你好!”标题的预览窗口即便出现，报表可以通过打印机打印，可以保存为“*.fp3”文件中，或者导出到系统可以支持的其他文件格式。

2. 3、“Text” 组件

“Text” 组件有多个特性，现在我们已经知道这个组件允许用户显示文本、边框、填充颜色等，它可以通过任何字体、任何大小、任何样式进行显示。所有的参数基本上可以通过工具栏按钮的帮助进行设置完成。




这里有几种文本设置的应用事例：

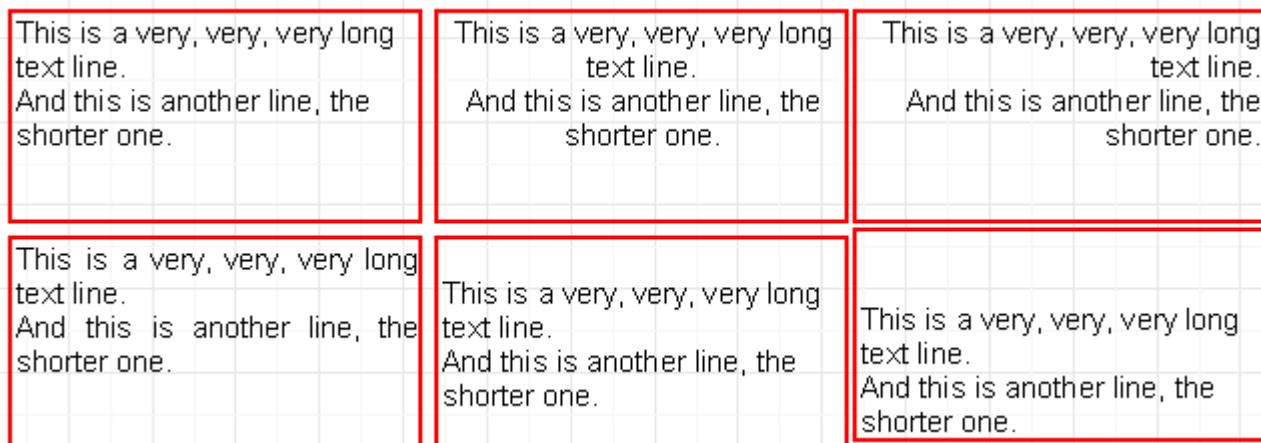



现在再让我们看看这个基础组件的其他特性，作为事例，我们首先创建一个新的对象，并在里面输入以下两行文字：

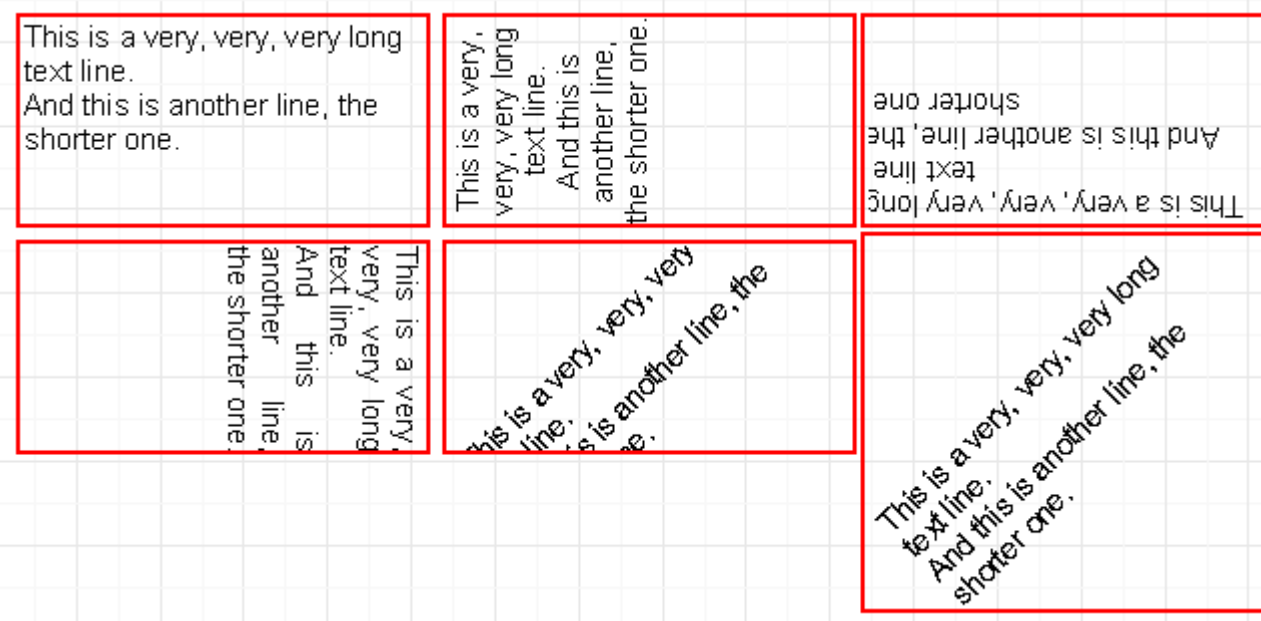
这是一个非常非常长的文本行
这一另外一行，稍短一点

设置边筐，然后通过鼠标把对象改变成 9X3 厘米的矩形。我们会看到组件不止显示一行文本文字，而是显示多行文本文字。在改变组件宽度到 5 厘米，很显然，长的行没有显示完整而是换行显示。这是因为“WordWrap”属性。如果设置成不可用，则长行显示会被剪切。

现在我们看看组件内部文字对齐的功能，对齐按钮位于“Text”工具栏上，允许用户设置文本的水平或垂直对其属性。注意按钮，这个按钮可以让段落分别向组件的两个边缘对齐，此时“WordWrap”属性必须可用才行。



组件中的文字都可以在 0 到 360 范围进行任意角度的旋转，“Text”工具栏中的按钮能让用户快速将文字旋转 45 度、90 度、180 度、270 度。如果你想让文字旋转任意角度，可以通过对象监视器。“Rotation”属性设置文字旋转的角度。如果文字旋转不在 90, 180, 270 度中，则文字可能超出组件范围，此时可以通过调整组件大小，显示文本文字。



现在我们简单看一下“Text”组件的其他属性，这些属性可能影响组件的外观显示，他们只

能在对象查看器中进行设置：

BrushStyle——对象填充类型

CharSpacing——文字间的间隙

GapX,GapY——设置文字水平方向和垂直方向缩进情况。

LineSpacing——行之间的间隙大小。

ParagraphGap——段落开始缩进情况

2. 4、在“Text”组件中使用HTML标记

是的，这个组件可以使用几种简单的 HTML 标记。HTML 标记可以添加在组件的文字中间。默认情况下，这些标记不可用，可以通过右键弹出的菜单中选择“Allow HTML Tag”菜单条，或在对象监视器中设置 AllowHTMLTag 属性。以下是系统可以支持的标记：

——加粗文字

<i>——斜体

<u>——文字下划线

<sub>——subscript

<sup>——superscript

——字体颜色

注意其他标记不支持，但是对大半应用程序已经够用了。他不能改变字体的大小和字体名称。否则文字的描述变得将更为复杂。

以下是标记的使用事例：

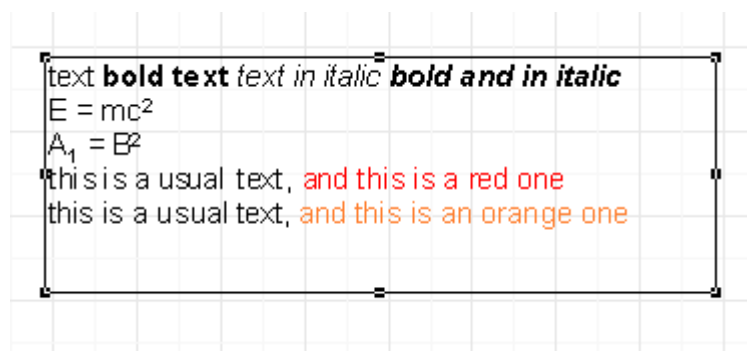
text bold text <i>text in italic</i> <i>bold and in italic</i>

E = mc²

A₁ = B²

this is a usual text, and this is a red one

this is a usual text, and this is an orange one



2. 5、通过“Text”组件显示公式

这个通用组件一个最重要的特性，不但可以显示静态的文字，而且可以显示公式。公式可以和文字混合编辑，下面让我们测试一下公式编辑的使用。

输入以下文字到组件内：

Hello, World! Today is [DATE].

当生成报表时，生成以下内容：

Hello, World! Today is 01.01.2007.

怎么形成这种结果呢？当 FastReport 生成报表时，组件中遇到有方括号里面的公式，报表引擎计算公式并将计算结果插入到公式的位置。“Text”组件可以和静态文字组合多个公式。单个变量和公式可以包含在一起[例如：1+2*(3+4)]。所有常量、变量、函数、数据库字段都可用用在公式里面。在此章的稍后我们将学习更多特性：

FastReport 自动识别文本组件方括号内的公式。如果我们不想让包含在方括号的内容被认为公式，该怎么办呢？例如，将显示如下的文字：

A[1] := 10

FastReport 包含[1]如果认为公式，则显示为如下：

A1 := 10

当然这不是我们需要的，解决这用情况的方法就是取消公式功能，取消“AllowExpressions”属性，或右键菜单中选择“Allow Expressions”菜单条，此时组件内的全部公式都被忽略。在我们的事例中，fastreport 将显示为我们需要的格式：

A[1] := 10

有时我们需要既带公式又带方括号的文本，例如：

a[1] := [myVar]

取消公式，不但我们需要的方括号内的数据可以显示，而且需要的公式也将不进行处理。

FastReport 允许用户使用另外的标记添加在公式里。默认情况下，“ExpressionDelimiters”属性等于[,], 他是可靠的。这种情况下，我们使用<>代替[]。

a[1] := <myVar>

<>值必须设置“ExpressionDelimiters”属性。正如你看见的，逗号分开开始和结束标记符号。这里有一个局限性：开始和结束标记不能相类似，所以“%, %”不能工作，设置几种标记，例如<%, %>, 这样，我们的事例将显示为：

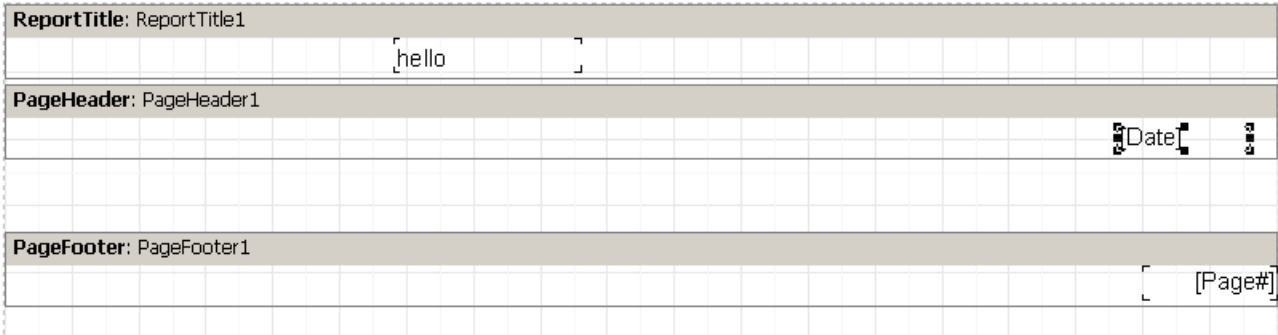

```
a[1] := <%myVar%>
```

2. 6、FastReport中的Bands

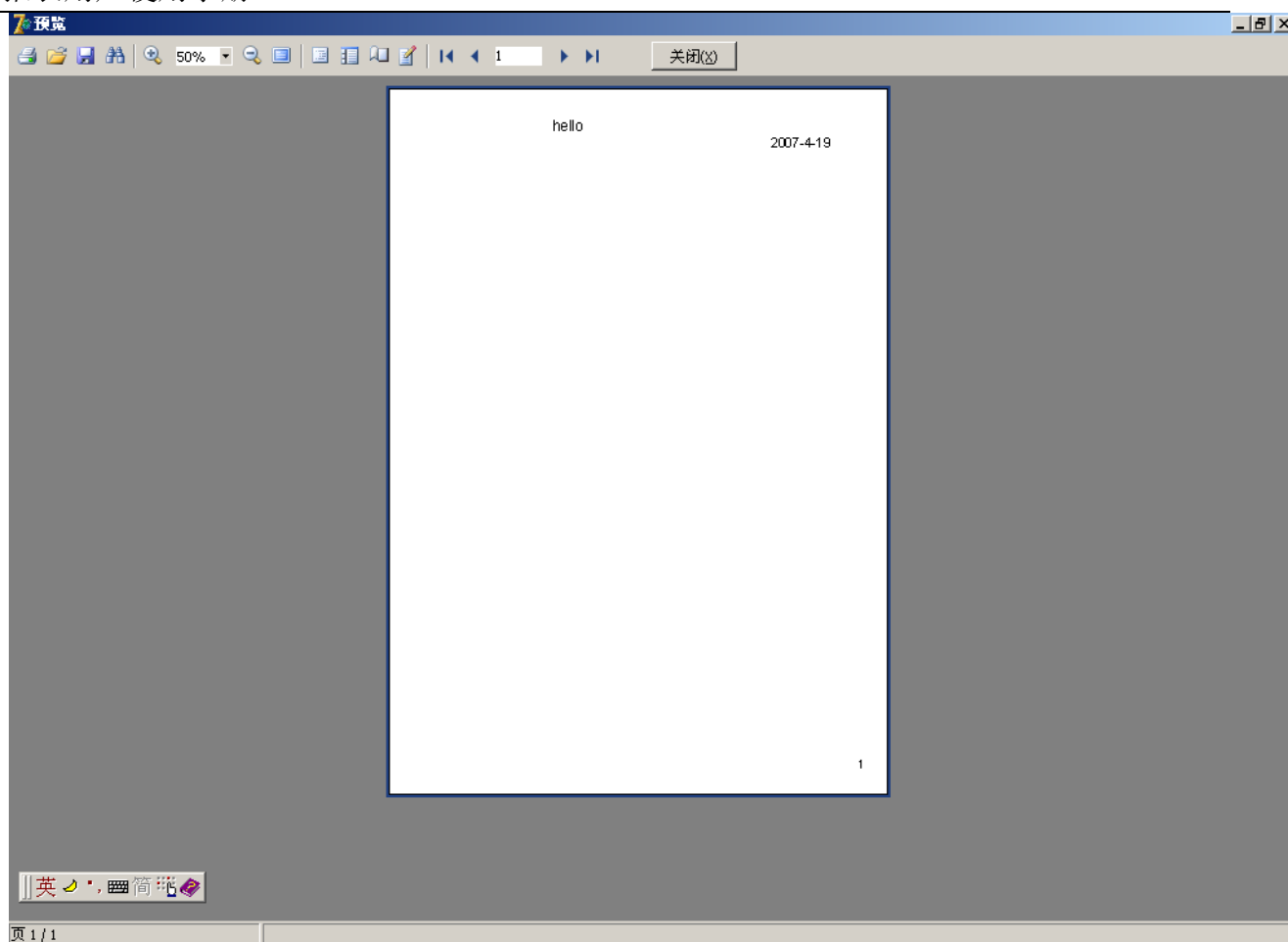
Bands用于在输出的页面上存放理论上的组件，当放置组件到Band上时，例如“Page Header”，让报表引擎在生成的报表的页头显示放置的组件内容。“Page bottom”,在报表的每个页尾显示band上的组件内容。让我们做个示范：创建一个新的报表，在页头添加一个“hello”标题，在页头的右侧显示日期，在页尾的右边显示页码：

打开FastReport报表设计器，在工具栏中点击“新报表”按钮，则生成报表模板，在上面已经有三个Bands，“Report Tile”，“Master data”，“Page footer”。先把“Master Data”删掉，选中后按del键,或从右键菜单中选择删除。现在添加一个新的Band(“Page header”),点击“Add Band”按钮，从下拉条中选择“页首”，我们看到一个新的band添加到报表设计器中。同时，已存在的Band位置往下移动。报表设计器自动摆放位置，页首在上面，数据区在中间，页尾在下面。

现在我们添加一些组件，添加“system text”到“页首”中，并从其编辑器的“system variable”中选择[DATE]。(你应该记住，日期在组件中显示为[DATE])。添加 TEXT 组件到 Report Tile 上，并输入“Hello”文本；此时你会发现页码已经显示在“Page footer”上了。



当运行报表，最终生成的报表中那些组件已经被放在了合适的位置上了。



因此，Band 是组件放置在合适位置的依赖。根据 Band 的类型，我们可以将组件放在页面的上部，或页面的下部，或第一页，或最后一页。基础 Band，他们在大部门的报表中都需要，有以下几种：

“Page header” 显示在报表每页的上面

“Page footer” 显示在报表每页的下面

“Report tile” 显示在第一页报表的上面，但在 Page header 的下面。



“Report Summary” 显示在报表的最后一页的空白处。

2. 7、Data Band

现在我们开始学习关于如何打印数据库表中的数据。怎么理解数据库表？就是有多个行（记录），每个行又有多个列（字段）组成的数据集合。打印这种类型的数据，FastReport 使用了特殊的 Band（DataBand）。这是一些名称带有“第 XX 阶数据”特征。为了打印整个表单或部分字段，需要添加这种类型的 band(s)，连接到数据库表单，并添加相关组件连接上要打印的数据

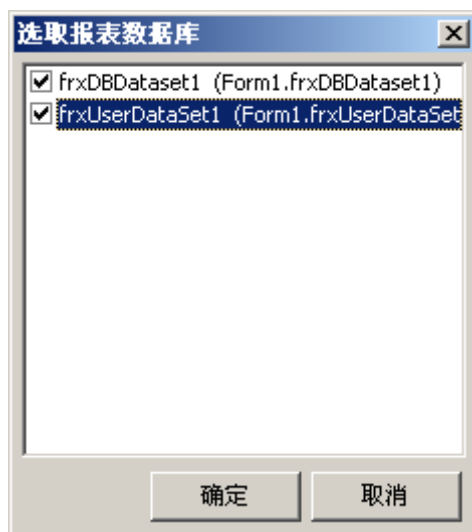
库表单的字段。当 FastReport 生成这些 bands 时，他重复打印生成这种 band，如同表单的记录数。如果打印输出页面已经没有了剩余空间，报表系统会自动增加一个新的页面用于打印输出报表。

2. 8、TfrxDBDataSet组件

在 FastReport 面板上的“TfrxDBDataSet”连接器组件用于连接数据库表或其他数据源。这个组件担当着数据源和 FastReport 核心之间的信使。这使的 FastReport 核心不依赖于任何数据访问连接库。FastReprt 可以同时工作于“BDE”、“IB_Objects”,或其他动态库，就象从没有和数据库连接的数据源中接收数据，例如：树组或文件等。工作过程中 TfrxDBDataSet 被数据源代替，他和 Tdataset 兼容（例如 BDE、ADO、IBX 等）。TfrxIBODataset 组件在工作过程中被 IB_Objects 代替；TfrxUserDataSet 通过其他数据源进行运行。

使用 TfrxDBDataSet 组件非常简单，设置他的 Dataset 属性或 DataSource 属性既可和数据源进行连接。两种连接方法是等效的，使用第一种方法可以不使用“TDatasource”组件。

为了将已经连接到数据源的组件用到报表中，报表中数据源需要进行明确的声明。在报表设计器中的点击“报表|数据”菜单，从打开的窗口中选择要连接的数据源。



2. 9、“客户列表”报表

我们的第二个报表比第一个稍微复杂一点（它包含一个数据库表单，一个公司的客户列表）。为了完成这个报表，让我们应用示范数据库 DBDEMOS。我们首先在 Delphi 中创建一个新的工程，把“TTable”组件放在 Form 上,并设置它的属性：

```
DatabaseName = 'DBDEMOS'
```

```
TableName = 'Customer.db'
```

为了使数据表的数据应用到报表中，需要在 Form 上添加一个“TfrxDBDataset”组件，并设置它的属性：

DataSet = Table1

最后在 Form 上添加一个 TfrxReport 组件，并打开报表设计器，在工具栏中点击“新报表”按钮，系统自动生成一个包含三个 Band 的报表。为了使数据表可以用在报表中，点击“报表|数据”菜单，从弹出的窗口中选择 frxDBDataset1,然后点击“确定”按钮。当窗口关闭后，数据源以及数据表字段在数据树窗口中的数据面板上即可看见。

现在开始创建报表，第一步先添加一个 Text 到 Report tile 中，并填写内容为“客户列表”，再连接“Master data”band 到我们的数据源，可以有三种方法来实现。

在 Band 中双击；

在 Band 右键菜单中选择“编辑”；

在对象监视器修改 Dataset 属性。

添加四个 Text 组件到 band 面板中，分别连接客户编号，客户名称，客户电话，客户传真。下面我们通过不同的方法进行设置。放置第一个 Text 组件，然后在编辑器中输入“[frxDBDataSet1."CustNo"]”，此数据可以人工进行手动输入，也可以通过公式编辑器进行输入。然后点击确定按钮，关闭窗口。

第二种方法就是在对象监视器中设置 Dataset 属性，放入第二个 Text 组件，在编辑器中不输入数据关闭。在对象监视器中设置：

DataSet = frxDBDataSet1

DataField = 'Company'

第三种方法就是从数据服务窗口中托拽字段到报表设计 band 中，这是最简单最容易的方法，开始之前，需要先设置取消数据服务窗口中的创建标题选项。

设置完后点击预览按钮。

客户列表			
编号	名称	电话	传真
1645	Action Club	813-870-0239	813-870-0282
3158	Action Diver Supply	22-44-500211	22-44-500596
1984	Adventure Undersea	011-34-09054	011-34-09064
3053	American SCUBA Supply	213-654-0092	213-654-0095
6312	Aquatic Drama	613-442-7654	613-442-7678
3984	Blue Glass Happiness	213-555-1984	213-555-1995
1380	Blue Jack Aqua Center	401-609-7623	401-609-9403
1563	Blue Sports	610-772-6704	610-772-6898
2118	Blue Sports Club	612-897-0342	612-897-0348
3054	Catamaran Dive Club	213-223-0941	213-223-2324
1354	Cayman Divers World	011-5-697044	011-5-697064
5151	Central Underwater	27-11-443245	27-11-4433259
2156	Davy Jones' Locker	803-509-0112	803-509-0553
3055	Diver's Grotto	213-432-0093	213-432-4821
3041	Divers of Blue-green	205-555-7184	205-555-6059
2315	Divers of Corfu, Inc.	30-661-88364	30-661-05943
4312	Divers of Venice	813-443-2356	813-443-9842
5432	Divers-for-Hire	679-804576	679-059345

2.10、通过text组件显示数据表字段

正如你所看到的，text 组件除了可以显示静态文本和公式以外，还可以显示数据表中的数据，这里有两种方法进行设置，第一就是在 text 组件中设置输入要连接的数据表字段，第二种方法就是在属性中设置 dataset 和 DataField。第一种方法可以用于显示字段的描述和字段数据同时显示。例如：

Contact person: [frxDBDataSet1."Contact_Person"]

正如你所看到的，关联数据表字段有他专有的语法结构，**数据库名称."字段名称"**。字段名称可以有空格，但在点号和引号之间不能有空格。

我们不但可以连接数据表字段到组件中，而且还可以对字段进行系统操作，例如：

*Length (cm): [<frxDBDataSet1."Length_in"> * 2.54]*

注意，在此方括号和尖括号都用了。但记住方括号被默认的作为公式的边界，它包含在了组件的文本中。如果需要，方括号可以被一组有开始/结束标记的符号所代替。**尖括号用在公式里面用于区分数据表字段和报表变量**。这里我们可以写成：

Contact person: [<frxDBDataSet1."Contact_Person">]

代替

Contact person: [frxDBDataSet1."Contact_Person"]

然而，这两种符号都是正确的，当公式中只有一个变量或数据表字段时，尖括号可以不用，然而，如果公式中有多个成员，则尖括号成为必需的了。

*Length (cm): [<frxDBDataSet1."Length_in"> * 2.54]*

2.11别名

在前面的报表中，我们使用了 frxDBDataSet1 和数据字段名称：CustNo,” “Company,” “Phone,” and “FAX.”，在报表中我们必须使用象"[frxDBDataSet1."CustNo"]"进行输入，这样输入是否直观呢？答案是否。一种方法就是修改数据源名称和字段名称。然而 frxDBDataSet1 是一个控件名称，它里面不支持空格，更不支持中文。CustNo 是数据表字段，更不能直接修改，除非修改数据表结构。那怎么办呢？使用别名。在这里非常容易的修改，假如我们设置了别名，就可以在报表中使用数据表别名否则就需要使用原始名称了。

对数据源和字段进行重命名非常简单，在 Delphi 开发环境中，双击 frxDBDataSet1 组件，打开别名编辑器。在此你可以修改数据源名称和字段名称：



注：数据源的别名的修改可以不通过别名编辑器进行修改，可以通过组件的 **UserName** 属性进行修改。

2. 12、变量。

除了别名的用法外，还有另外一种方法，让用户设置可以理解的数据库名称。一种就是就像公式一样连接数据字段名称到变量上。在 FastReport 中创建和使用变量，可以使用“报表|变量”菜单项，或在工具栏中点击“变量”按钮。

在 **FastReport** 中变量结构有两层，第一层为目录，第二层为变量本身。目录是为了当变量列表过多时便于管理。列表中至少需要包含一个目录，这表示变量本身不能直接连接在根目录上。再说，目录是变量分级必需的，因此报表中不包含目录。这是因为，**设置的变量名称在系统中必须是唯一的，在不同的目录中设置相同的变量名称是不可能。**

通过以下事例我们解释一下变量的应用。假定我们有两个数据源，一个是带有“CustNo”和“Name”字段的 frxDBDataset1，一个是带有“OrderNo”，“Date”字段的 frxDBDataset2。我们将字段组织成以下的变量列表：

Clients

客户 序号

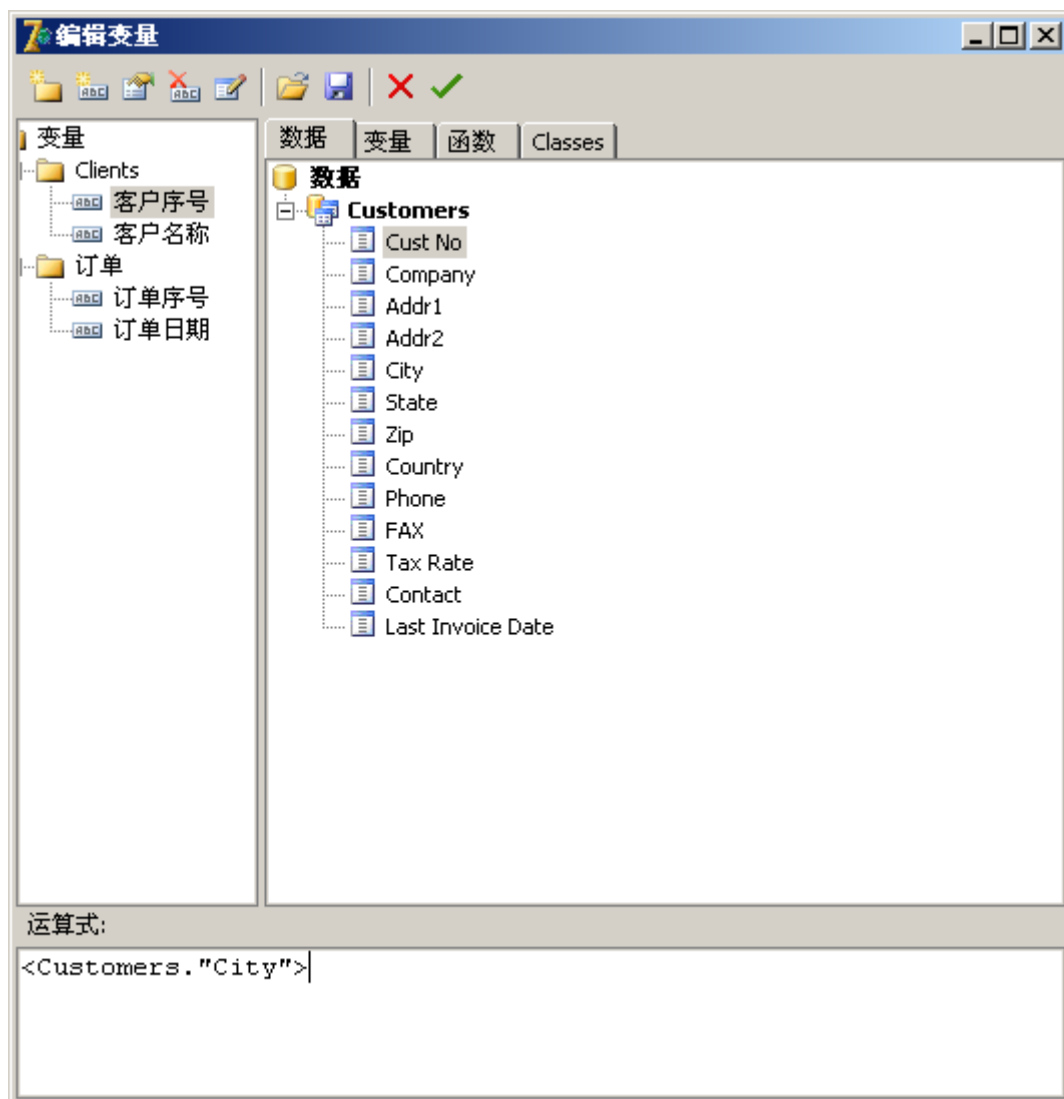
客户 名称

Orders

订单 序号

订单 日期

这里 **Clients** 和 **Orders** 是两个目录。通过新目录、新变量、编辑按钮创建一个目录和变量。连接变量到数据库字段中，首先选中要连接的变量，在窗口的右边双击要连接的数据表字段，则要连接的字段移动到下边输入筐中。这样变量就和公式建立了连接，变量的值就是公式计算的值。如果需要，公式可以通过人工进行输入修改，而且如果需要，还可以在公式中添加其它的字段或内部函数或其他变量等。目录不能建立这种连接。

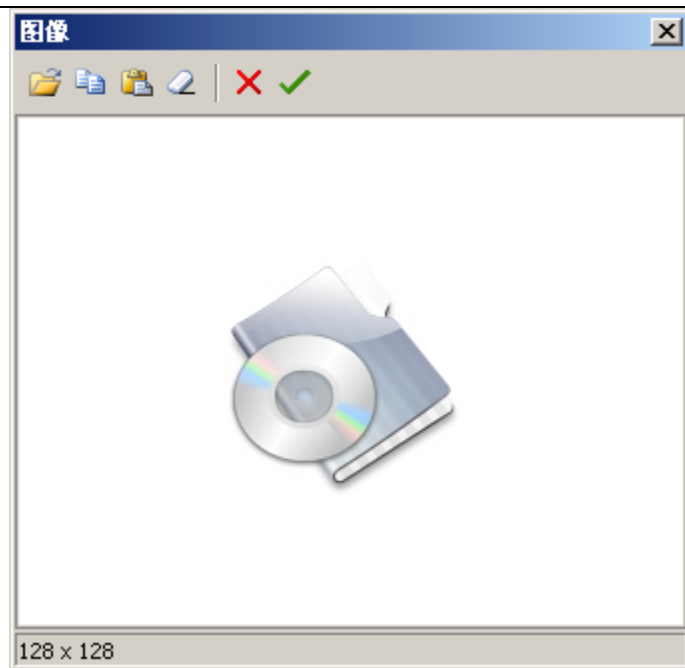


变量创建以后，关闭变量编辑窗口。现在可以插入变量到报表中。和输入字段名称一样，在此输入变量。通过键入 “[客户序号]” 人工插入变量到组件中。或通过在数据树窗口中托拽要连接的变量到报表指定的地点。

2. 13、“Picture” 控件

下一个要学习的在报表中也是经常用到的组件是 “Picture” 控件。通过这个组件，用户可以插入厂标、员工照片、或其他图形信息。“Picture” 组件可以显示的图形格式有：“BMP” “ICO” “JPG” “JPEG” “WMF” “EMF” 等。

现在看一下这个组件的性能。创建一个新的报表，并在页面上添加一个 “Picture” 组件。在组件编辑器中（如果编辑器没有自动打开，可以在组件上双击鼠标左键打开），加载需要的图片，点击确定按钮。这里可以从文件中加在图片，也可清除已有的图片。



在组件的右键菜单中有几个有用的属性，分别对应着对象查看器的属性名称。

- AutoSize：自动调整大小。
- Stretch:延伸，默认是自动延伸。
- Center:图形居中。
- KeepAspectRatio: 保持图形的分辨率。

另外一个有用的属性就是“FileLink”。你可以输入图片所在的路径名称，则报表在生成的时候，自动从文件中进行加载，生成报表。

2. 14、图形报表

在 FastReport 中，“Picture” 组件和其他组件一样，通过设置 Dataset 和 DataField 属性可以连接到数据表字段上。和“Text” 组件相比，它只能连接到图形数据上。

让我们做一个示范，显示鱼和鱼的名称。我们仍然使用“DBDEMOS”数据库。

在 Delphi 中创建一个新的工程，并添加“TTable” 组件到表单上，设置其属性：

```
DatabaseName = 'DBDEMOS'
```

```
TableName = 'Biolife.db'
```

使用 FastReport,添加“TfrxDBDataset” 组件，设置属性：

```
DataSet = Table1
```

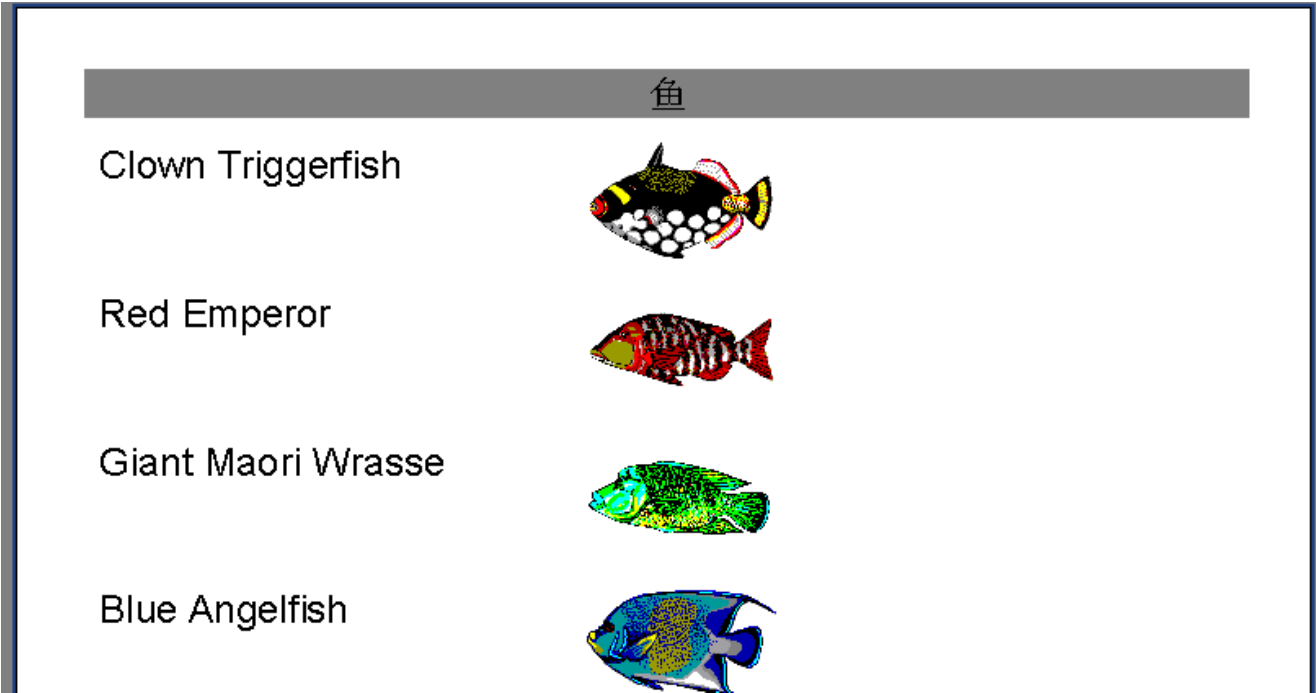
```
UserName = 'Bio'
```

在添加“TfrxReport” 组件，打开报表设计器点即“新报表” 按钮，生成一个基本报表，在“报表|数据” 菜单条打开数据窗口，选择 Bio，然后确定。

现在可以在报表中添加组件，现在“Report Tile”中添加一个“Text”组件，输入“鱼”，在调整“Master Data”Band 的高度到 5 厘米，在上边添加一个“Text”组件，让它连接数据表的 CommonName 字段，在添加一个“Picture”组件，让它连接数据表的 Graphic 字段，并调整其位置和大小。





```
DataSet = Bio
DataField = 'Graphic'
```

报表设置完成，以下是预览结果：



2. 15、多行文本显示

在上一个报表中，包含一个“Notes”字段，它里面包含各种鱼的详细描述。在添加一个“Text”组件，让它连接到“Notes”字段上，并调整大小和位置。设置完成后，预览结果如下图：

鱼		
Clown Triggerfish		Also known as the big spotted triggerfish. Inhabits outer reef areas and feeds upon
Red Emperor		Called seaperch in Australia. Inhabits the areas around lagoon coral reefs and sandy
Giant Maori Wrasse		This is the largest of all the wrasse. It is found in dense reef areas, feeding on a wide variety
Blue Angelfish		Habitat is around boulders, caves, coral ledges and crevices in shallow waters. Quipa




当然每个组件都可以调整其大小，字体的大小，然而，这样就可能在报表中出现大面积的浪费空间，毕竟每种鱼的描述长度不一样。有的描述长，有的描述短。FastReport 可以通过简单的设置几个属性，即可解决这种问题。

为了使用需要的空间，这涉及到 Bands 和组件能够自动调整高度的能力。我们需要设置 Bands 和组件的 Stretch 属性改为 Enable。但并不是全部都需要，根据需要设置一部分即可。

“Text” 组件自动调节其高度和宽度，以有足够的空间显示其内容。可以通过 “AutoWidth” 和 “StretchMode” 属性来实现。“AutoWidth” 属性让组件自动调整宽度，以显示全部内容而不需要分割。这个属性在文本只有一行并且不会影响右边组件显示的情况下非常有用。“Stretch” 属性让组件自动调整高度，而不改变宽度的情况下，以显示容纳全部文本。这个属性有多个模式，用户可以在组件查看器中设置。

- SmDontStretch——不进行拉伸操作，默认。
- SmActualHeight——自动调整每一行的高度适应全部文字。
- SmMaxHeight——自动调整高度适应Band的最大高度。

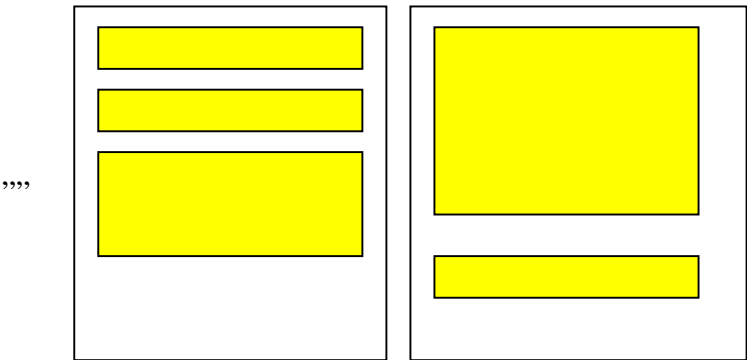
我们看一下 “Text” 组件的 “StretchMode” 属性，通过组件的右键菜单设置或通过组件监视器设置 “StretchMode = smActualHeight”，并改变Band的Stretch属性为enable；然后预览报表结果：

鱼		
Clown Triggerfish		<p>Also known as the big spotted triggerfish. Inhabits outer reef areas and feeds upon crustaceans and mollusks by crushing them with powerful teeth. They are voracious eaters, and divers report seeing the clown triggerfish devour beds of pearl oysters.</p> <p>Do not eat this fish. According to an 1878 account, "the poisonous flesh acts primarily upon the nervous tissue of the stomach, occasioning violent spasms of that organ, and shortly afterwards all the muscles of the body. The frame becomes rocked with spasms, the tongue thickened, the eye fixed, the breathing laborious, and the patient expires in a paroxysm of extreme suffering."</p> <p>Not edible.</p> <p>Range is Indo-Pacific and East Africa to Somoa.</p>
Red Emperor		<p>Called seaperch in Australia. Inhabits the areas around lagoon coral reefs and sandy bottoms.</p> <p>The red emperor is a valuable food fish and considered a great sporting fish that fights with fury when hooked. The flesh of an old fish is just as tender to eat as that of the very young.</p> <p>Range is from the Indo-Pacific to East Africa.</p>
		<p>This is the largest of all the wrasse. It is found in dense reef areas,</p>

2. 16、文本拆分

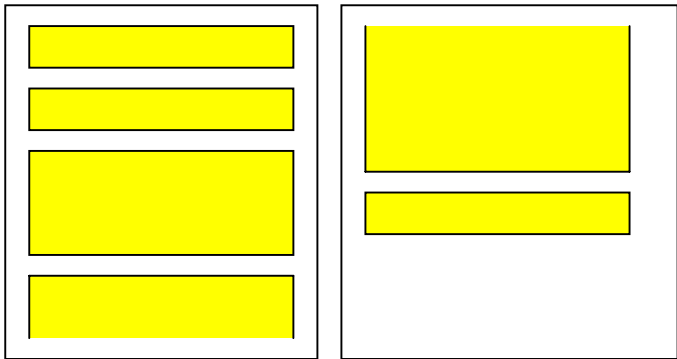
我们注意一下报表的特性，在页面的下边有好多空白位置，这是为什么？当报表生成时，FastReport 用 band 填充页面，每填充一行，当前位置自动顺序往下移，当 FastReport 发现下一个 Band 在此页中没有了足够的空间进行显示，则 FastReport 生成一个新的页面进行继续显示。根据数据表中的记录数继续生成报表。

一个报表包含有一个大型文本的组件，这是 Band 高度增高的原因。另外，如果 Band 在此页中没有发现足够的空间用于显示，则生成一个新的页面，则在这个页面的下面留有空白位置。



为了减少页面的浪费，使用 FastReport 特性，对文本内容进行分段。我们只需在第一页的

分层的数据 Band 上通过右键菜单设置 “AllowSplit” 属性。这样就减少了页面空间的浪费。

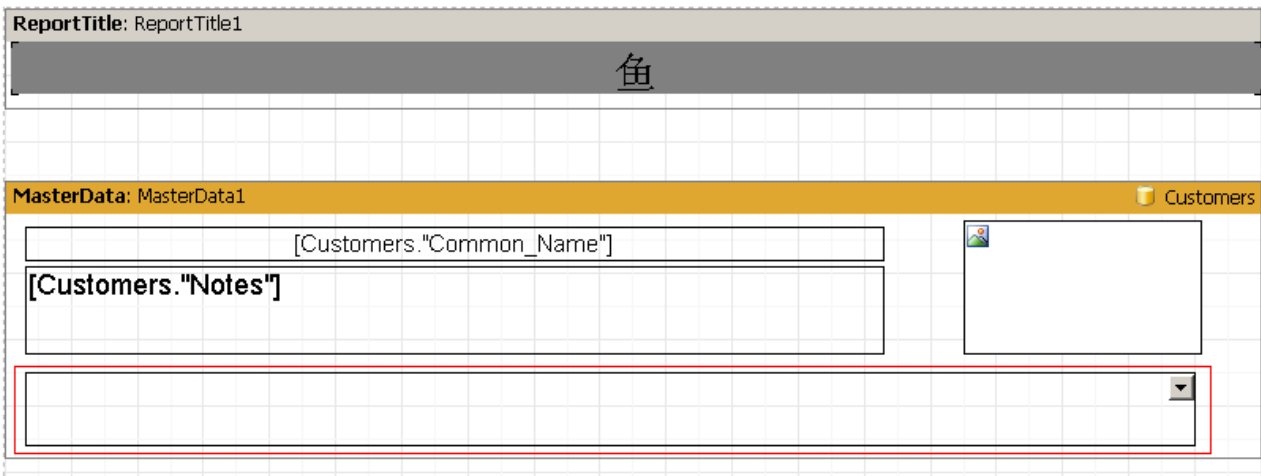


Band 是怎么拆分的呢？在 FastReport 中有多种组件支持这种特性。有 “line”，“Text”，“RichEdit”，这些可以拆分，其他组件不可以。



2. 17、组件的Wrap

报表设计时，有时需要文本信息围绕其他组件（如图形）进行显示。通过当前事例示范一下报表的这个特性。

在 “Bio.”Notes”” 组件的下方添加一个 “Text” 组件，如下：



设置 “Bio.”Notes”” 的Strech属性为否，并设置下边的组件的Strech属性。选中 “Bio.”Notes”” 组件，在对象查看器中设置FlowTo属性，从下拉筐中选择下面的组件名称。事例结果为：

鱼	
<div>Clown Triggerfish</div> <div>Also known as the big spotted triggerfish. Inhabits outer reef areas and feeds upon crustaceans and mollusks by crushing them with</div>	
<div>powerful teeth. They are voracious eaters, and divers report seeing the clown triggerfish devour beds of pearl oysters.</div> <div>Do not eat this fish. According to an 1878 account, "the poisonous flesh acts primarily upon the nervous tissue of the stomach, occasioning violent spasms of that organ, and shortly afterwards all the muscles of the body. The frame becomes rocked with spasms, the tongue thickened, the eye fixed, the breathing laborious, and the patient expires in a paroxysm of extreme suffering."</div> <div>Not edible.</div> <div>Range is Indo-Pacific and East Africa to Somoa.</div>	
<div>Red Emperor</div> <div>Called seaperch in Australia. Inhabits the areas around lagoon coral reefs and sandy bottoms.</div>	
<div>The red emperor is a valuable food fish and considered a great sporting fish that fights with fury when hooked. The flesh of an old fish is just as tender to eat as that of the very young.</div> <div>Range is from the Indo Pacific to East Africa.</div>	

注：主要显示的组件要先于联接的组件插入到报表中，如果相反，则报表不能达到预期的结果。这是可以通过联接组件右键菜单，设置置于顶层。

2. 18、显示数据表中的数据

有时我们需要带有边筐显示数据表中的数据。一个简单的例子就是物价列表。在FastReport, 用户只需修改组件的边筐属性即可。做个示范：

创建新的报表，设置如下：

MasterData: MasterData1		
[Customers."Species	[Customers."Common Name"]	[Customer

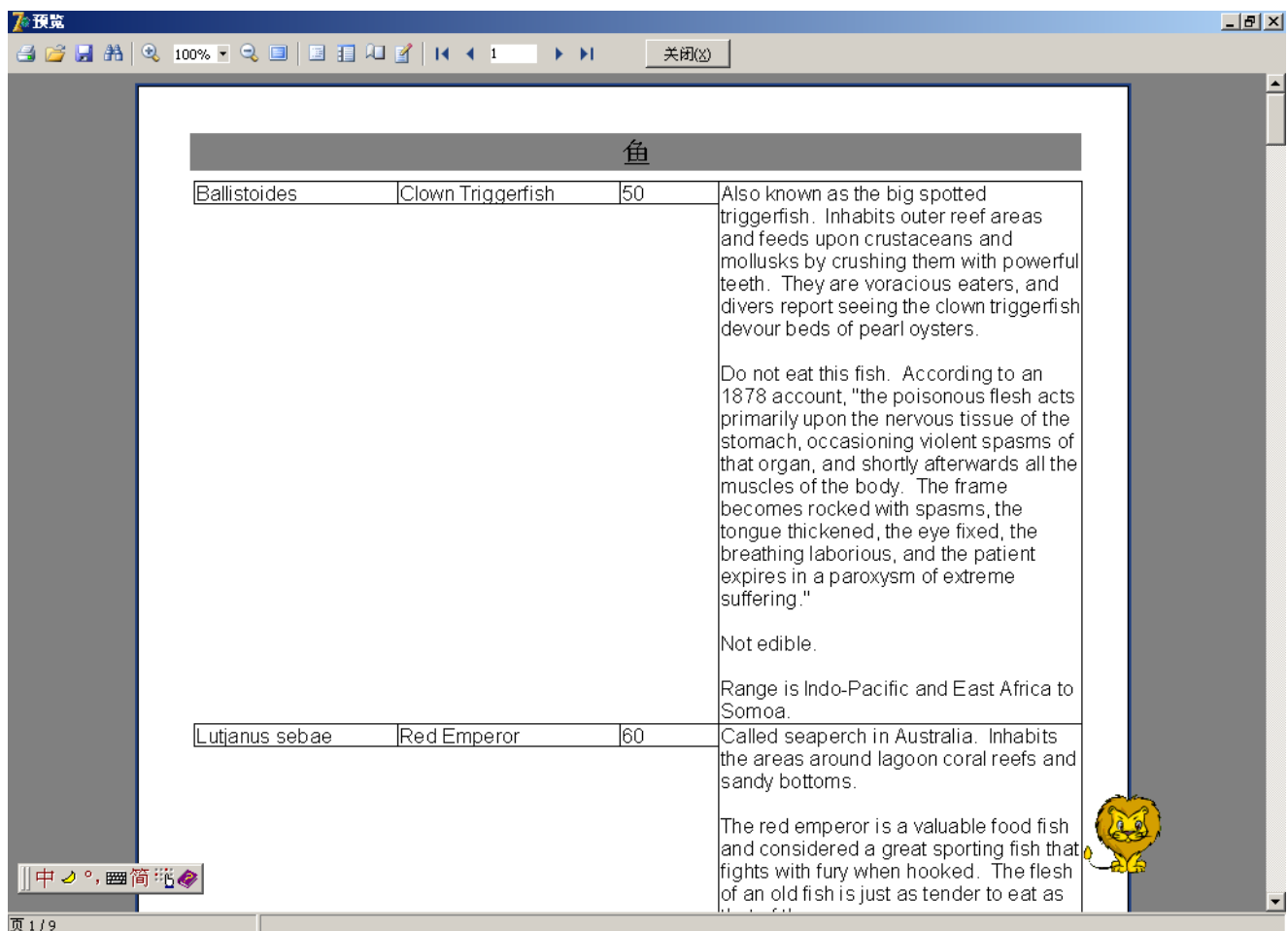
显示结果：

鱼

Ballistoides	Clown Triggerfish	50
Lutjanus sebae	Red Emperor	60
Cheilinus undulatus	Giant Maori Wrasse	229
Pomacanthus	Blue Angelfish	30
Variola louti	Lunartail Rockcod	80
Pterois volitans	Firefish	38
Chaetodon	Ornate Butterflyfish	19
Cephaloscyllium	Swell Shark	102
Myliobatis californica	Bat Ray	56
Gymnothorax mordax	California Moray	150
Ophiodon elongatus	Lingcod	150
Scorpaenichthys	Cabezon	99
Chaetodiperus faber	Atlantic Spadefish	90
Ginglymostoma	Nurse Shark	400
Aetobatus narinari	Spotted Eagle Ray	200
Ocyurus chrysurus	Yellowtail Snapper	75
Sparisoma	Redband Parrotfish	28
Sphyrna barracuda	Great Barracuda	150
Haemulon	French Grunt	30

有时只显示水平线或只显示垂直线，或显示边线，分别设置边筐的属性值即可实现。

如果报表中包含有大文本数据，设置文本的 **Stretch** 属性，显示结果：



这和我们需要的略有差别，边线的高度应该相同。我们设置其他组件的StretchMode属性，StretchMode = smMaxHeight，然后再显示：

鱼			
Ballistoides conspicillum	Clown Triggerfish	50	<p>Also known as the big spotted triggerfish. Inhabits outer reef areas and feeds upon crustaceans and mollusks by crushing them with powerful teeth. They are voracious eaters, and divers report seeing the clown triggerfish devour beds of pearl oysters.</p> <p>Do not eat this fish. According to an 1878 account, "the poisonous flesh acts primarily upon the nervous tissue of the stomach, occasioning violent spasms of that organ, and shortly afterwards all the muscles of the body. The frame becomes rocked with spasms, the tongue thickened, the eye fixed, the breathing laborious, and the patient expires in a paroxysm of extreme suffering."</p> <p>Not edible.</p> <p>Range is Indo-Pacific and East Africa to Samoa.</p>
Lutjanus sebae	Red Emperor	60	Called seaperch in Australia. Inhabits the areas around lagoon coral reefs and

2. 19、标签式打印

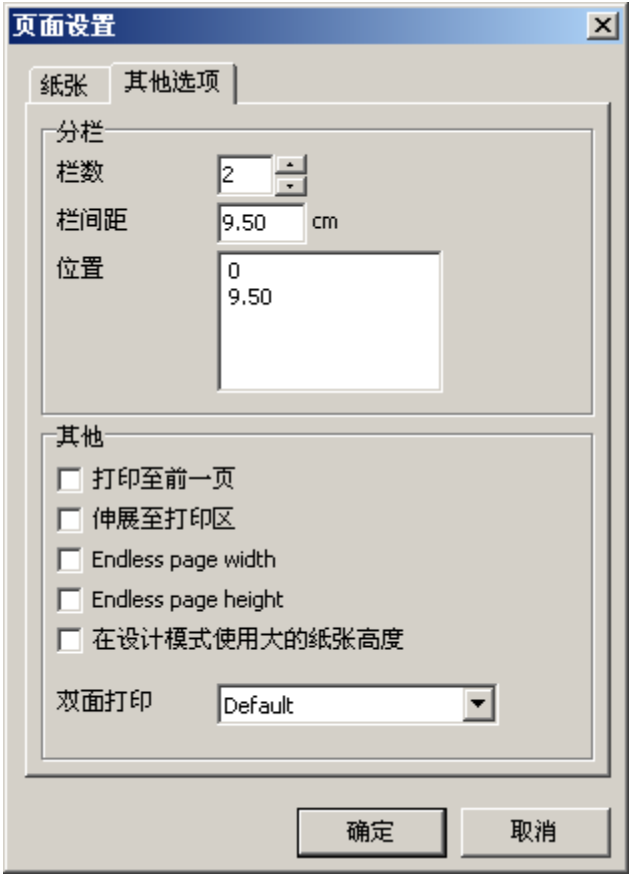
和表格式报表对比，标签式数据报表是另外一种数据分配方式。做一个示范，如下图形：

MasterData: MasterData1	
序号	[Customers."Specie"
种类	[Customers."Catego
名称	[Customers."Comm
长度	[Customers."Length"

预览结果如下：

鱼	
序号	90020
种类	Triggerfish
名称	Clown Triggerfish
长度	50
序号	90030
种类	Snapper
名称	Red Emperor
长度	60
序号	90050
种类	Wrasse
名称	Giant Maori Wrasse
长度	229
序号	90070
种类	Angelfish
名称	Blue Angelfish
长度	30

你会注意到，页面的有半部分是空白区，要在整个页面中显示数据，需要在页面配置中设置栏数。如下图：



设置后，预览结果：

鱼			
序号	90020	序号	90140
种类	Triggerfish	种类	Cod
名称	Clown Triggerfish	名称	Lingcod
长度	50	长度	150
序号	90030	序号	90150
种类	Snapper	种类	Sculpin
名称	Red Emperor	名称	Cabezon
长度	60	长度	99
序号	90050	序号	90160
种类	Wrasse	种类	Spadefish
名称	Giant Maori Wrasse	名称	Atlantic Spadefish
长度	229	长度	90
序号	90070	序号	90170
种类	Angelfish	种类	Shark
名称	Blue Angelfish	名称	Nurse Shark
长度	30	长度	400

我们可以取消页面设置中的列数，而设置 band 的 columns=2,并设置 columnwidth,columngap 值

MasterData: MasterData1		Customers	
序号	[Customers."Specie"		
种类	[Customers."Catego		
名称	[Customers."Comm		
长度	[Customers."Length"		

这种报表方式和上一种方法的区别，上一种方法从左到右显示，这种方法显示从上到下顺序进行显示。

2. 20、子bands

让我们测试标签类型的报表多个行中的一行可能有可变大小。模拟事例情形，缩小“Bio.”Common Name””组件的宽度到2.5厘米，并设置 “Strech” 属性，设置“First level data”band的Strech 属性，结果如下：

序号	1645
国家	Action Club
城市	Sarasota
电话	813-870-0239
序号	3158
国家	Action Diver Supply
城市	St. Thomas
电话	22-44-500211
序号	1984
国家	Adventure Undersea
城市	Belize City
电话	011-34-09054

2.21、组件的移动

2.22、两个数据阶的报表（主—细）

至此前边的报表系统只使用了一种数据源。FastReport 允许可以在设计器生成六中数据阶的报表。通常的报表在 1-3 个数据阶的报表。

让我们模拟一个俩个数据源的报表（主—细）。选择数据源 “Customer” 和 “orders”。第一个表是客户数据，第二个是客户的订单信息。数据表包含如下信息：

Customer:

CustNo	Company
1221	Kauai Dive Shoppe
1231	Unisco
1351	Sight Diver
....	

Orders:

OrderNo	CustNo	SaleDate
1003	1351	12.04.1988
1023	1221	01.07.1988
1052	1351	06.01.1989
1055	1351	04.02.1989
1060	1231	28.02.1989
1123	1221	24.08.1993
....		

开始设计报表，首先在 delphi 中创建一个新的工程，在表单上放两个 “TTable”，一个 “TDataSource”、两个 “TfrxDBDataset” 和一个 “TfrxReport” 组件，设置组件属性：

Table1:

```
DatabaseName = 'DBDEMOS'
TableName = 'Customer.db'
```

Table2:

```
DatabaseName = 'DBDEMOS'
TableName = 'Orders.db'
```

DataSource1:

```
DataSet = Table1
```

frxDBDataSet1:

```
DataSet = Table1
UserName = 'Customers'
```

frxDBDataSet2:

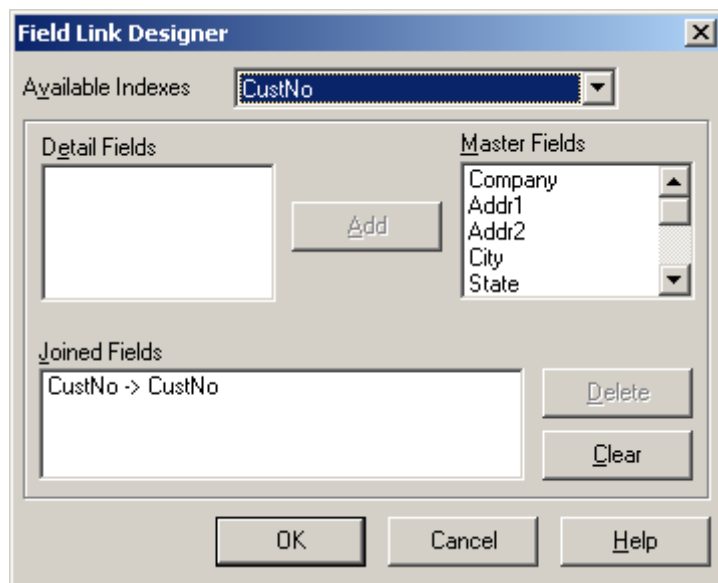
```
DataSet = Table2
UserName = 'Orders'
```

在报表设计器中，“报表|数据” 窗口设置连结数据源，并在报表页面上添加 “Master Band” 和 “Detail Band” 两个 Band。

MasterData: MasterData1		
[Customers.]	[Customers."Company"]	
DetailData: DetailData1		
[Sales."Orde"]	[Sales."Cust"]	[Sales."Sale Date"]

注意 master Band 一定要在 detail band 的上边，如果在下面，在生成报表时会产生错误。

如果此时预览，会看到报表预览全部数据源，因为没有设置order的MasterSource属性值，在Delphi设计环境中设置table2的属性 “MasterSource = DataSource1”，建立Master—detail连结。



我们连接两个数据源的 custNo 字段。主—细分别选择 custNo 字段，点击 add 按钮，添加到连结的字段列表中。点击 “OK”，关闭窗口。

预览报表数据，显示如下：

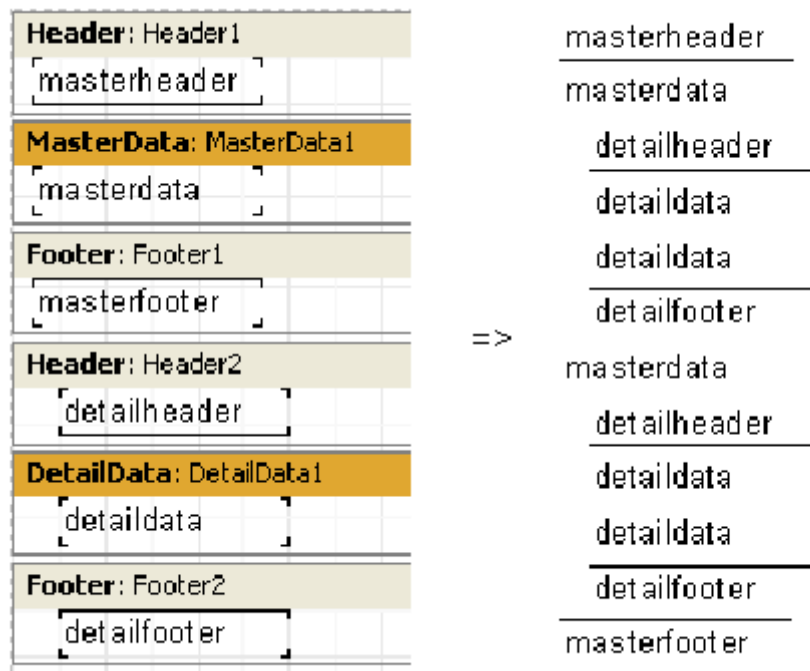
1645 Action Club		
1014	1645	1988-5-25
1029	1645	1988-7-18
1038	1645	1988-8-26
1129	1645	1993-10-19
3158 Action Diver Supply		
1039	3158	1988-8-29
1984 Adventure Undersea		
1017	1984	1988-6-12
1037	1984	1988-8-26
1074	1984	1989-4-19
1099	1984	1989-6-16
1117	1984	1993-4-13
1137	1984	1993-11-27
1217	1984	1994-11-22
1294	1984	1995-1-4
1317	1984	1995-2-1
3053 American SCUBA Supply		
1204	3053	1994-10-18
1263	3053	1994-12-14

2.23、页首和页尾数据Band

每个数据 band 都可能 有头和尾，Header 就是显示在数据 Band 前面的数据，footer 就是显示在数据 Band 后面的数据，这里设置一个简单的事例：

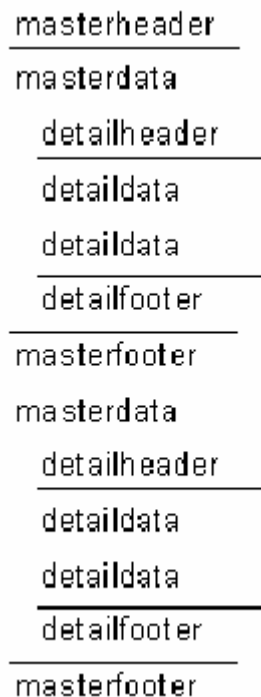
Header: Header1		头
		数据
MasterData: MasterData1		数据
[Customers."Cust [Cus		数据
Footer: Footer1		数据
		数据
		数据
DetailData: DetailData1		尾
[Sales."Orde [Sal		

然后再看一个复杂点的，包含主—细表的报表



我们可以看到，**header** 就是打印在数据记录前面的数据内容。**Master band** 前面的 **header** 在报表的开始的时候打印一次；**detailband** 前的 **header** 在详细数据分组的开始都打印一次。**Footer band** 同理，**master band** 的 **footer** 只在报表末尾打印一次，**detail band** 的 **footer** 则在每个细数据分组结束时打印一次。


“FooterAfteretch”属性可以覆盖上面的操作。他在设计主-细报表是可能非常有用。例如：

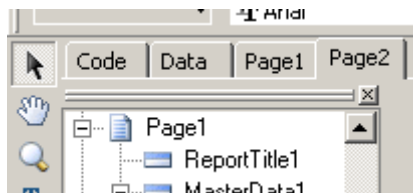



2.24、多页报表

在 FastReport 中，报表可能有多个设计页组成，用户可以为每一页调整页面大小和方向。

在上面也可以放置组件和 **Band** 组件。当这种类型的报表输出时，首先显示第一页面中的内容，然后是第二页.....

用户创建报表时，FastReport 默认有一个页，可以通过点击工具栏中的按钮或菜单”文件|新页面”添加新的页面。可以在设计器中看到一个新的页面标签。



通过点击页标签很容易的在页面之间进行切换。通过对页标签的托拽很容易的改变页面的顺序。点击按钮，可以很容易的从报表中删除选中的页。另外的一种快捷操作方式就是在页标签上点击右键，通过右键菜单进行操作。



设计页面数在报表中没有限制，一般，新添的页面不是作为封皮，就是在更为复杂的报表中应用，一般他包含有多个数据源。

做一个简单的事例，就是做一个简单的封皮。在刚才报表中再添加一个页面，此时页面为空，在上面添加一个 text 组件，输入标题：“封皮事例”，然后将页面的顺序通过鼠标的托拽操作，提到第一个页面的前面。预览结果：

封皮事例		
1645	Action Club	
1014	1645	1988-5-25
1029	1645	1988-7-18
1036	1645	1988-9-26
1129	1645	1989-10-19
3182	Action River Supply	
1039	3182	1988-9-29
1584	Adventure Unleashed	
1017	1584	1988-6-12
1037	1584	1988-9-26
1014	1584	1989-4-19
1039	1584	1989-6-16
1117	1584	1989-1-13
1137	1584	1989-11-27
1217	1584	1990-11-22
1294	1584	1995-1-4
1317	1584	1995-2-1
3053	American Soft Ball Supply	
1204	3053	1990-10-18
1263	3053	1990-12-14
1266	3053	1995-2-6
6312	Aquatic Drama	
1036	6312	1989-3-25
2684	Blue Class Hopalong	
1042	2684	1988-9-24
1142	2684	1989-12-25
1380	Blue Jack Aqua Center	
1006	1380	1989-11-6
1019	1380	1989-6-3
1106	1380	1992-9-23
1153	1380	1990-4-16
1253	1380	1990-11-26
1553	Blue Sports	
1012	1553	1988-5-19
1057	1553	1989-2-18
1051	1553	1989-3-3
1033	1553	1989-6-9
1091	1553	1989-6-28
1148	1553	1990-3-3
1151	1553	1990-6-4
1168	1553	1990-7-4
1212	1553	1990-11-14
1261	1553	1990-12-11
1283	1553	1990-12-30

多行报表还需要注意的一点是：当第二页的“print to previous page”属性设置 true,则第二页的打印不是从新的一页开始，而是从第一页的空白位置接着打印。报表续行打印。

第三章

分组 集合体

3. 1、分组打印

前面的报表我们使用的是基于两个数据源的主-细报表。FastReport 可以实现通过关联建立的数据源信息。

为了演示，我们需要一个从两个表中通过查询条件返回数据源的 sql 语言的查询语句，

```
select * from customer, orders
      where orders.CustNo = customer.CustNo
      order by customer.CustNo
```

在此 Order by 语句是必需的。返回结果：

CustNo	Company ...	OrderNo	SaleDate
1221	Kauai Dive Shoppe	1023	01.07.1988
1221	Kauai Dive Shoppe	1123	24.08.1993
1231	Unisco	1060	28.02.1989
1351	Sight Diver	1003	12.04.1988
1351	Sight Diver	1052	06.01.1989
1351	Sight Diver	1055	04.02.1989

在这样一个数据源中怎样组织一个多阶报表呢？在 FastReport 中有一个特殊的 Band——Group header。为这个 Band 制定条件。当这个 Band 数据发生变化时打印一次。下面事例说明：

在 Delphi 环境中创建一个新的功能，放入一个 "TQuery"，一个 'Tfrxreport'，一个 "TfrxDBDataset" 组件。并设置属性值：

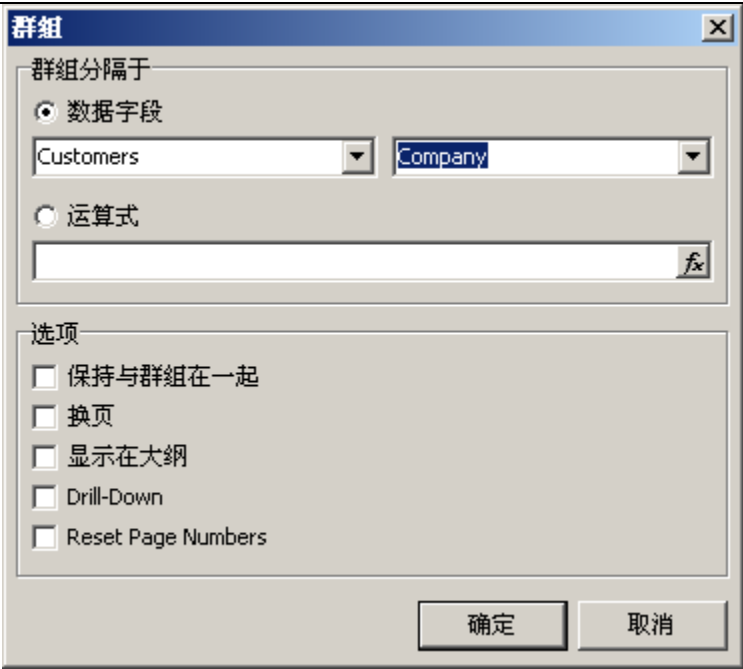
Query1:

```
DatabaseName = 'DBDEMOS'
SQL =select *from customer, orders
      where orders.CustNo = customer.CustNo
      order by customer.CustNo
```

frxDBDataSet1:

```
DataSet = Query1
UserName = 'Group'
```

打开报表设计器，连接数据源，放入一个 group header，一个 master band，并设置 group header 的显示条件。



在连结数据源到 Group header band 中。

GroupHeader: GroupHeader1		Group."CustNo"
[Group."CustNo"]	[Group."Company"]	
MasterData: MasterData1		Group
[Group."OrderNo"]	[Group."SaleDate"]	

打开预览，显示如下：

1221	Kauai Dive Shoppe
1269	16.12.94
1023	01.07.88
1176	26.07.94
1076	16.12.94
1123	24.08.93
1169	06.07.94
1231	Unisco
1173	16.07.94
1178	02.08.94

3. 2、其他分组特性

让我们看看组是怎样转移到下一页的：

1380	Blue Jack Aqua Center
1006	06.11.94
1079	03.05.89
1106	23.09.92
1153	16.04.94
1253	26.11.94
1384	VIP Divers Club
1007	01.05.88
1027	07.07.88

我们查看报表输出时，从第二页不能清晰的看出订单数据是那个客户的数据。FastReport 允许在打印新的页面时重新打新组的内容。通过修改 group header 的属性 ReprintOnNewPage 属性设为 true，或右键菜单选择 Reprint on new page 菜单项。在此预览结果：

1380	Blue Jack Aqua Center
1006	06.11.94
1380	Blue Jack Aqua Center
1079	03.05.89
1106	23.09.92
1153	16.04.94
1253	26.11.94
1384	VIP Divers Club
1007	01.05.88

还有另外一个办法避免分组拆开，就是设置 Group Header 的 Keep together 属性，这样报表当发现没有足够的空间显示组内容时，添加新的一页开始显示。结果如下图：

1356	Tom Sawyer Diving Centre
1005	20.04.88
1059	24.02.89
1072	11.04.89
1080	05.05.89
1105	21.07.92
1180	06.08.94
1266	15.12.94
1280	26.12.94
1305	20.01.95

1380	Blue Jack Aqua Center
1006	06.11.94
1079	03.05.89

这样，可能有多个页面出现大面积空白区，但分组数据显示在了一起。

结论，group header 的 “StartNewPage” 属性允许组内容打印在一个单独得页面上，这样虽然可能会产生许多空白区域，但他在一定情况下是非常有用的。

3.3页码的重设

组有一个 “ResetPageNumber” 属性，允许打印一个分组报表时从新设置页码属性。这有什么用呢？

假如有一个分组报表，就是组头时客户名称，内容为客户的订单，如果这些订单要分别发送到客户手里，按常规打印，可能有的用户收到的报表页从 50 页开始，51，52.....。前面的 49 页去那里了，客户该问了。所以需要每个客户的报表页码分别设置。

还要注意：你设置了 “ResetPageNumber” 属性，还要设置 “StartNewPage”，这样才能够每个组从新的页面开始，页码独立计算。打印页码和总页数可以使用系统变量[Page],[TotalPages]。

3. 4、组的操作

Group header 有一个叫 drilldown 的属性，如果 drilldown 属性设为 true，则在报表时变为交

交互式报表，意思是在预览窗口用户可用通过鼠标在组的标题上单击，以实现展开和关闭详细信息内容。

这里有一个事例：

Customers				
Company	Address	Contact	Phone	Fax
A				
B				
C				
Catamaran Dive Club	Box 264 Pleasure Point	Nicole Dupont	213-223-0941	213-223-2324
Cayman Divers World Unlimited	PO Box 541	Joe Bailey	011-5-697044	011-5-697064
Central Underwater Supplies	PO Box 737	Maria Eventosh	27-11-4432458	27-11-4433259
				Count: 3
D				
F				

你在第一次生成报表时可以控制全部组内容是展开还是缩回。默认情况全部组不展开。如果想展开，可以设置 ExpandDrillDown 属性为 True，或者通过右键上下文菜单进行操作。

3. 5、行数

让我们的事例显示每个组的行数，在没有 band 上添加一个 text 组件，并设置系统变量[line]。如图：

GroupHeader: GroupHeader1		
[Line]	[Group."CustNo"]	[Group."Company"]
MasterData: MasterData1		
[Line]	[Group."OrderNo"]	[Group."SaleDate"]

浏览结果，可以看到每个组分别显示序列号。

1	1221	Kauai Dive Shoppe
1	1023	01.07.88
2	1076	16.12.94
3	1123	24.08.93
4	1169	06.07.94
5	1176	26.07.94
6	1269	16.12.94
2	1231	Unisco
1	1060	28.02.89
2	1073	15.04.89
3	1102	06.06.92
4	1160	01.06.94

如果想让第三个 band 的页码成序号延续，可以在 band 中将[line]改为[line#]。显示结果如下图：

1	1221	Kauai Dive Shoppe
1	1023	01.07.88
2	1076	16.12.94
3	1123	24.08.93
4	1169	06.07.94
5	1176	26.07.94
6	1269	16.12.94
2	1231	Unisco
7	1060	28.02.89
8	1073	15.04.89
9	1102	06.06.92

3. 6、函数集

在分组报表中，一般情况下需要显示一些结论性的内容，如分组数，组内行数等。为了这个目的，FastReport 提供了一些函数集。

- Sum 返回公式的和
- Max 返回给定参数的最大值

Min 返回公式的最小值

Avg 返回公式的平均值

Count 返回数据行数

除 count 函数外的其他函数的语法如下，以 sum 为例：

sum(公式, band, 标记)

sum(公式, band)

sum(公式)

参数解释：

公式——要显示数据的公式

band——数据 Band 的名称

标记——一个字节，可以是以下数据之一或他们的和

- 1: 计算时包含不可见 Band
- 2: 对计算值进行累计

公式是必需的参数，其他时可选的，不过，有歧义时，应该使用 band 这个参数。

Count 函数的语法：

Count(band, 标记)

Count(band)

参数说明和上面类似。

对所有的函数有一个规定，就是只适用于数据 Band 或数据 Band 的 Footer Band.

集合函数是怎么工作的呢？让我们做一个事例：

GroupHeader: GroupHeader1	
[Group."CustNo"]	[Group."Company"]
MasterData: MasterData1	
[Group."OrderNo"]	[Group."SaleDate"] [Group."ItemsTotal"]
GroupFooter: GroupFooter1	
[SUM(<Group."ItemsTotal">,MasterData1)]	

数据 Band 的 “ItemTotal” 字段显示当前表单的总计，在 groupfooter 中添加一个 text 组件，输入[sum(<group.”itemstotsl”>,masterdata1)]。显示结果如下图：

1221	Kauai Dive Shoppe	
1023	01.07.88	\$4 674,00
1076	16.12.94	\$17 781,00
1123	24.08.93	\$13 945,00
1169	06.07.94	\$9 471,95
1176	26.07.94	\$4 178,85
1269	16.12.94	\$1 400,00
		51450,8

函数中“标记”参数的目的就是为了有些报表部分数据 Band 可能隐藏不可见，不管是否可见都应该计算在范围内。在我们的事例中，如果我们设置数据 Band 的 visible 属性为 false,在打印是不可见。为了计算这些不可见的数据行，需要添加上这个参数。

`[SUM(<Group."ItemsTotal">,MasterData1,1)]`

可能显示如下结果：

1221	Kauai Dive Shoppe	
		51450,8
1231	Unisco	
		85643,6
1351	Sight Diver	
		261575,8

如果设置标志参数为 2，显示以后结果不在重新设置。每个显示结果为运行期的总和，如：

`[SUM(<Group."ItemsTotal">,MasterData1,3)]`

3 就是 1 和 2 的和，意思是计算不可见的的数据，并对计算结果不进行重新复位，打印报表结果：

1221	Kauai Dive Shoppe	51450,8
1231	Unisco	137094,4
1351	Sight Diver	398670,2

3. 7、页和报表的统计

通常，我们的一个需求可能是显示一个报表或一个页面的汇总信息。这种情况下我们可以使用这些函数。将我们的例子做一下改变进行说明：

GroupHeader: GroupHeader1		Group."CustNo"
[Group."CustNo"]	[Group."Company"]	
MasterData: MasterData1		Group
[Group."OrderNo"]	[Group."SaleDate"]	[Group."ItemsTotal"]
GroupFooter: GroupFooter1		
		[SUM(<Group."ItemsTotal">,MasterData1)]
ReportSummary: ReportSummary1		
		Total: [SUM(<Group."ItemsTotal">,MasterData1)]
PageFooter: PageFooter1		
		Total this page: [SUM(<Group."ItemsTotal">,MasterData1)]

我们在报表上添加了一个“报表合计”band，并在上面添加一个 text 组件，组件中输入 sum 函数，还添加了一个 PageFooter Band.这是我们需要的。显示结果为：

6812	Waterspout SCUBA Center	
1040	04.09.1988	3 632,00p.
1140	12.12.1993	1 240,00p.
		4 872,00p.
9841	Neptune's Trident Supply	
1149	14.03.1994	12 900,75p.
1045	16.10.1988	787,80p.
1049	13.12.1988	1 809,85p.
1145	17.01.1994	4 229,80p.
		19 728,20p.
		Total: 2922666,1
		Total this page: 320872,8

3. 8、插入汇总函数

至此我们只是手工在 Text 组件中输入这些汇总函数，下面我们说说另外一种输入方法。

首先，我们可以使用 “system text” 组件输入这些汇总函数，事实上，它是和 “text” 组件是相同的。只不过他又一个快捷输入数据的方法。



我们可以一步一步的选择函数类型，数据 Band，和数据库字段，或者公式等我们需要的值。一样可以设置执行不可见的 Band 的数据 和 执行总数复选标志。

可以通过点击文本编辑器的  按钮，打开汇总函数设置窗口。

第 四 章

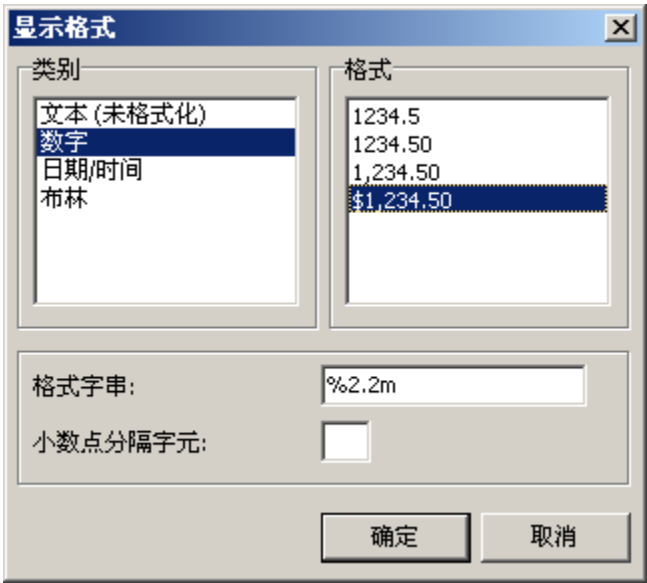
格式化 加强

4. 1、格式化输出结果

汇总函数的输出结果的一个特性就是没有格式化，就看前面的 sum 的例子就显而易见。

1176	26.07.94	\$4 178,85
1269	16.12.94	\$1 400,00
		51450,8

如果要格式化输出，可以使用 fastreport 的格式化工具进行设置。选中要格式化的组件，右键菜单选择“formatting”，或在对象查看器中设置“displayformat”属性，弹出格式化工具窗口。



可以看到，窗口左边是变量目录的列表，右边是格式化列表。我们选择数字目录，格式化选择\$1,234.50。格式化的参数参考 delphi 的 Format 函数。

点击确定按钮，再次预览：

1176	26.07.94	\$4 178,85
1269	16.12.94	\$1 400,00
		\$51450,80

4. 2、内嵌格式化

上面的事例中，是格式化一个组件。每次工作只是作用在一个组件上。如果有两个公式，并且有不同格式输出，就需要内嵌格式化。

上边的事例中，改变 footer 中组件的大小，并输入一下内容：

```
Total: [SUM(<Group."ItemsTotal">,MasterData1)]
```

Number: [COUNT(MasterData1)]

总和和订单数分别显示如下，但结果不是我们需要的：

1269	16.12.94	\$1 400,00
		Total: \$51 450,80
		Number: \$6,00

为了正确显示，需要对数字进行分别格式化。在行内使用格式化符号进行设置，写在公式方括号前。修改组件的内容：

Sum: [SUM(<Group."ItemsTotal">,MasterData1) #n%2,2m]

Number: [COUNT(MasterData1)]

预览，看一下结果是否正确：

1269	16.12.94	\$1 400,00
		Total: \$51 450,80
		Number: 6

使用格式化符号，语法如下：

[公式 #格式化符号]

注意公式和#之间应该有至少 1 个空格，格式化符号如下：

#n 格式化符号：对数字文件进行格式化

#d 格式化符号：对日期进行格式化

#bFalse,True：对布尔型结果进行格式化

格式化符号对数字类型可以参考 delphi 中的 Format 函数，对日期型可以参考 delphi 中的 FormatDatetime 函数。以下是 Format 中涉及格式：

数字格式：

%g

%2.2f

%2.2n

%2.2m

日期格式：

dd.mm.yyyy

dd mmm yyyy

dd mmmm yyyy


hh:mm

hh:mm:ss

dd mmmm yyyy, hh:mm

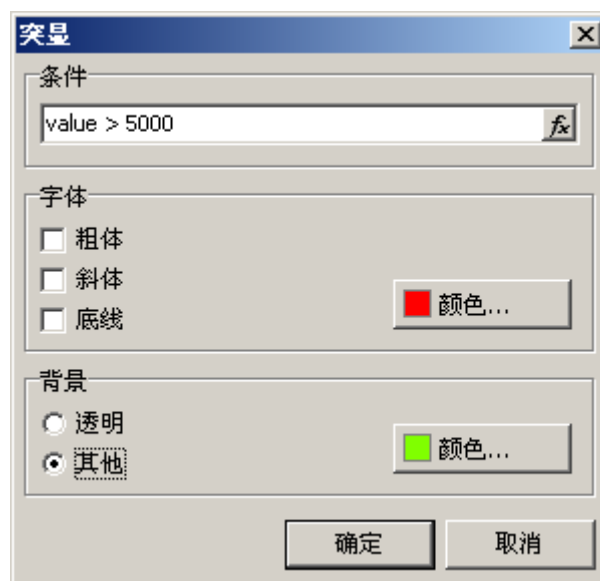
数字格式化字符串中间可以加入 “,” 或 “-”。这些符号何以用在整数和小数之间的分割符号。

#b 格式化是“布尔”型，这个用在用逗号分开的两种结果的格式化。第一个就是“否”，第二个就是“是”。

如果记不住这些格式符号和符号的意思，可以有一个自动设置环境，让用户配置。在“Text”组件的编辑器中点击“”组件。即可打开格式化编辑器。选择格式化内容，点击确定，格式化符号就插入到组件中，这样如果光标在方括号的前面或后面，格式化符号即可正确插入。

4. 3、条件显示

“Text”组件就有根据不同的条件而显示不同的颜色设置的特性。我们使用一个带分组的事例做个示范：让汇总的结果大于5000的用绿色显示。选择带“Group."ItemsTotal"”字段的组件，或点击highlight。打开条件编辑器，输入条件，指定字体和背景颜色。点击确定按钮。



预览结果显示如下：

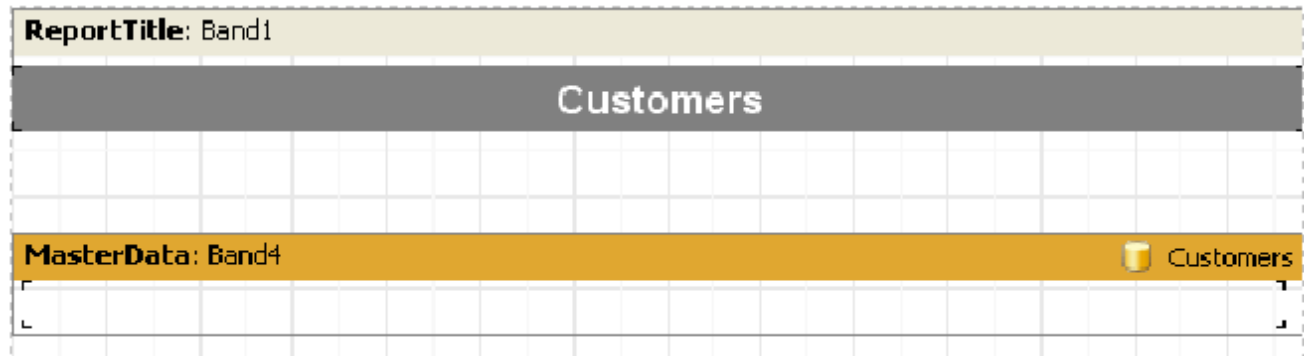
1221	Kauai Dive Shoppe	
1023	01.07.88	\$4 674,00
1076	16.12.94	\$17 781,00
1123	24.08.93	\$13 945,00
1169	06.07.94	\$9 471,95
1176	26.07.94	\$4 178,85
1269	16.12.94	\$1 400,00
		Total: \$51 450,80

注：指定的条件公式中，结果是和组件连结的数据表字段的值，同理，也可以设置“<Group."ItemsTotal"> > 5000”条件公式，

4. 4、分行显示数据行的颜色

使用条件显示，可以非常容易的生成两行具有不同颜色。

创建一个新的报表，在“Master Data” Band 中添加一个 Text 组件，调整大小为 Band 的大小，让他占用整个 Band。



根据行号设置组件的显示颜色，选择组件，在条件编辑窗口中设置条件公式如下：

```
<Line> mod 2 = 1
```

注意：如果在组件中选择 C++脚本，应该写成如下，使用 c++脚本语言：

```
<Line> % 2 == 1
```

将条件设置的颜色设为灰色，边框不可见。再在上面添加其他的组件：

ReportTitle: Band1		
Customers		
MasterData: Band4		
Customers		
[Customers."Company"]	[Customers."Phone"]	[Customers."FAX"]


预览报表，显示结果：

Customers		
Action Club	813-870-0239	813-870-0282
Action Diver Supply	22-44-500211	22-44-500596
Adventure Undersea	011-34-09054	011-34-09064
American SCUBA Supply	213-654-0092	213-654-0095
Aquatic Drama	613-442-7654	613-442-7678
Blue Glass Happiness	213-555-1984	213-555-1995
Blue Jack Aqua Center	401-609-7623	401-609-9403
Blue Sports	610-772-6704	610-772-6898

第 五 章

嵌套报表 (子报表)

5. 1嵌套报表

有时一些复杂的报表结构中,需要报表的特殊位置显示特殊的数据。通过 FastReport 的 Band 可以创建这些报表。创建子报表可以使用“子报表组件” 。

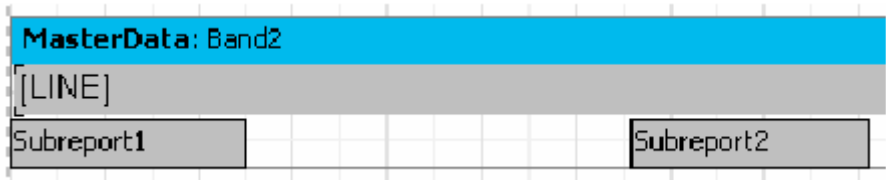
插入这种组件后,就会在设计器中添加一个新的页面,它连接着这个子报表组件。镶嵌式报表就像一系列的多页报表系统。唯一不同的就是嵌套报表显示在基础报表中特殊的位置,而不是在他的后面。当输出报表时,当 FastReport 系统遇到“SubReport”组件, FastReport 引擎,输出关联的设计页,然后输出报表的其他部分。

还可以在子报表窗面上添加子报表,增加嵌套层数。

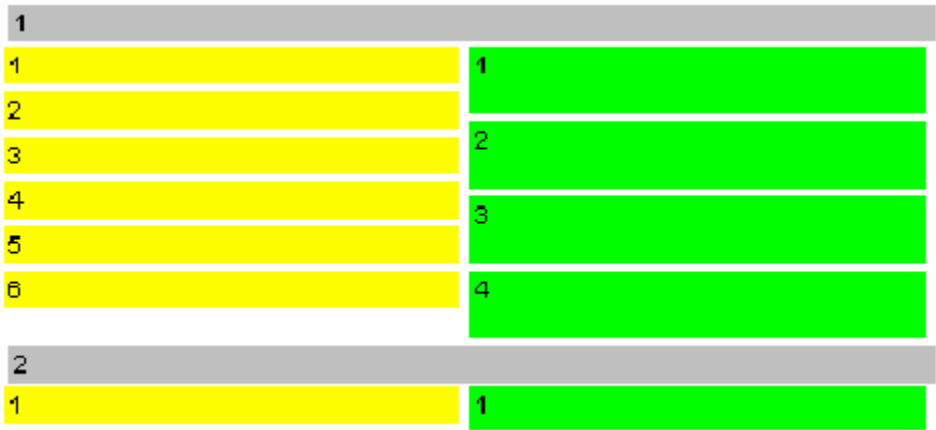
注意:可以增加嵌套报表,但最多有 6 层数据。

5. 2、设计子报表

你可以在同一 Band 上放置两个或更多的“SubReport”组件。



这样允许报表设计中,输出数据可以有不同的长度,不同的宽度,不同的高度。

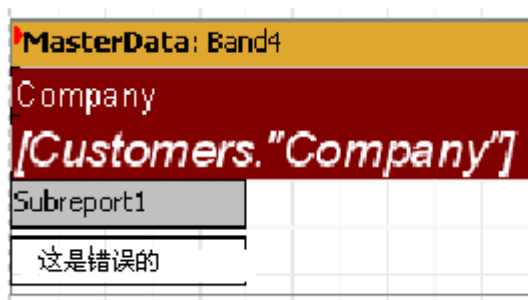


如你所见,报表在输出全部子报表后,生成报表结构。子报表也可以使用各种对齐方法属性。

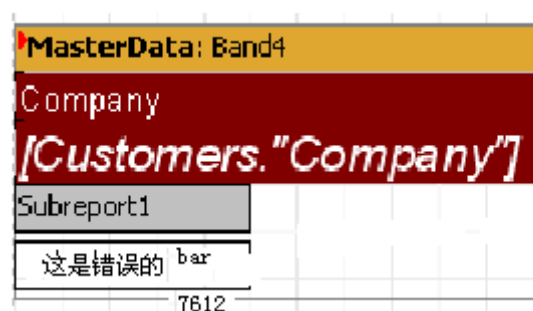
5. 3、子报表中的限制

既然子表是主报表的一部分,他又不能在包含以下Band: “ReportTitle/ReportFooter”、 “PageTitle/PageFooter/PageBackground,” 和 “ColumnTitle/ColumnFooter.”。把这些Band是可以放在组表单上,但是报表系统是不能对其进行处理。同理,嵌套表的选项设置一样不被理会,因为报表输出的时候,主要用到的是主表单的选项。

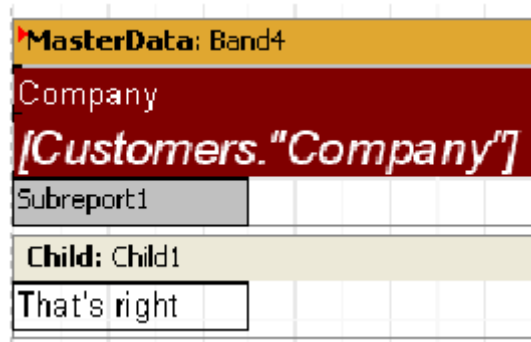
不能把组件放在“subreport”组件的下面。



当显示子报表的数据时，报表系统将用子报表的内容覆盖“subReport”组件下面的位置。打印可能会出现以下情况：



如果要在 subreport 下面显示数据，可以使用 child-band。



在需要多个子报表的情况，这种方法一样适用，

5. 4、PrintOnParent选项

“subreport”组件又一个属性“printonparent”属性，根据情况有需要。默认属性为 false。通常子表显示在主表的 band 上，父 Band 不依赖于子 Band。不进行拉伸。如果“printonparent”属性设为 true，则打印输出报表的子报表到主报表的包含有“subreport”组件的 band 上。你可以使 Band 可以随内容进行拉伸。

	1
	2
	3
	4
	5
	6

第 六 章

脚本

Script 是一种高级程序语言，他是报表的一部分。报表运行的时候，脚本语言也运行。脚本用户处理报表数据，例如，隐藏不需要的数据；脚本也用于处理对话框等属性。

脚本语言可以使用脚本组件引擎中的一种语言，当前可以支持以下几种语言：

- PascalScript
- C++Script
- BasicScript
- JScript

以下是脚本引擎脚本特性变量：

——标准语言设置：变量、常量、过程、带有变量、常量和默认参数的函数，所有标准操作符（`case`, `try`, `finally`, `except`, `with`），数据类型（`integral`, `fractional`, `logical`, `character`, `line`, `multidimensional arrays`, `set`, `variant`），类（方法，事件，属性，索引，和默认的属性）。

——类型的描述：脚本中的结构和类。没有结构，没有指针，不能跳转。

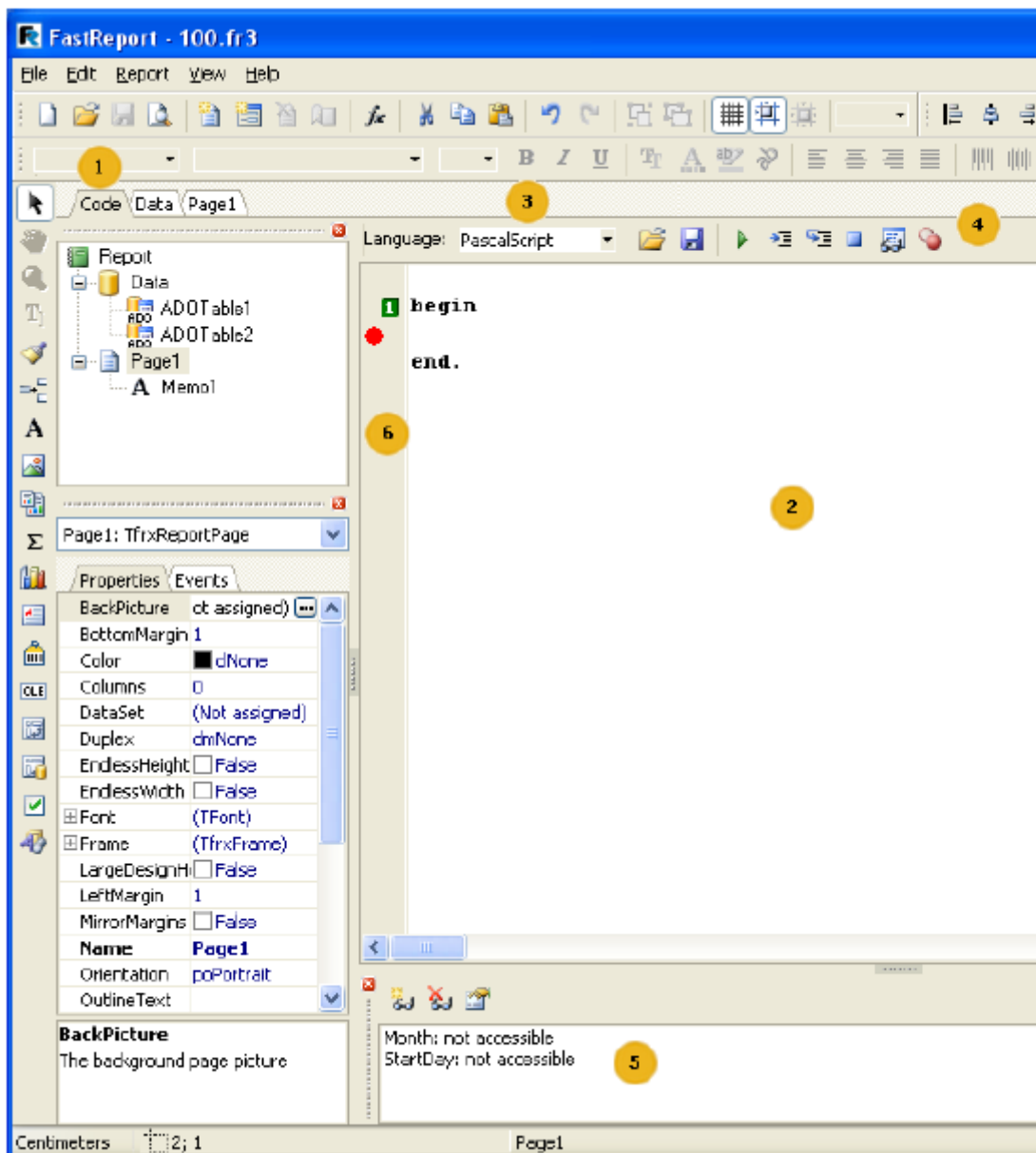
——类型的兼容性。

——可操作任何组件。

用户可以在报表设计器添加程序语言，还可以在线进行调试。


6. 1、体验脚本语言


FastReport 设计器的 **Code** 页上有关于语言的一些工具栏。切换到 **code** 页后，界面如下图：





注释:

- 1: code 切换页
- 2: 脚本编辑窗口
- 3: 选择使用的脚本语言类型
- 4: 调试工具栏

: 开始调试运行

: 运行到当前光标处

: 单步调试

: 停止调试



: 监视变量



: 设置取消断点

5: 监视窗口

6: 断点、标签标识显示区。

以下是脚本编辑器中可用的按键说明

按键	描述
Cursor Arrow	移动鼠标
PageUp,PageDown	向前、向后翻页
Ctrl+PageUp	将光标移到页首
Ctrl+PageDown	将光标移到页尾
Home	将光标移到当前行的开始
end	将光标移到当前行的最后
delete	删除光标位置的字符。或删除已选择的字符。
backspace	删除光标前面的字符。
Ctrl+Y	删除当前行
Ctrl+Z	取消最后的操作
Shift+Cursor arrows	选择字符
Ctrl+A	选择字符串
Ctrl+U	将选中的行的内容左移两个字符
Ctrl+I	将选中的行的内容右移两个字符。
Ctrl+C	复制内容到剪贴板中
Ctrl+V	将剪贴板中的字符串粘贴到当前位置。
Ctrl+X	剪切选中的内容到剪贴板中
Ctrl+Shift+<数字>	设置标签
Ctrl+<number>	跳到标签位置
Ctrl+F	查找
Ctrl+R	查找替代
F3	重复查找或查找替换

F4	设置断点
Ctrl+F2	复位
Ctrl+F7	预览变量值
F9	运行脚本
F7 or F8	单步执行代码

6. 2、脚本结构

脚本的结构和你用的语言有关，但是他们有一些通用的元素。报表生成是可以运行的脚本的标题、主体以及主函数。以下是报表支持的四种语言的例子。

PascalScript 的结构

```
#language PascalScript // 选项
program MyProgram; // 选项
/ / “uses” 章解放在其他章节的最前面。
    uses 'unit1.pas', 'unit2.pas';
var // “variables” 章节可以放在任何位置
    i, j: Integer;
const // “常量” 章节
    pi = 3.14159;
procedure p1; // 过程或函数
var
    i: Integer;
procedure p2; // 嵌套函数或过程
begin
end;
begin
end;
begin //主程序.
end.
```

C++Script的结构:

```
#language N++Script // 现象
// “include” 张解放在其他章节的最前面
#include "unit1.cpp", "unit2.cpp"
int i, j = 0; // “variables” 章节可以放在任何位置
#define pi = 3.14159 // “常量” 章节
void p1() //函数
{ // 没有嵌套函数
}
{ // 主程序.
}
```

JScript’ s 结构

```
#language JScript // 选项
// “import” 章节放在其他章节的最前面
import "unit1.js", "unit2.js"
var i, j = 0; // “variables” 章节
anywhere
function p1() // 函数
{ //
}
// 主程序
p1();
for (i = 0; i < 10; i++) j++;
```

BasicScript结构

```
#language BasicScript ' 选项
' “imports” 章节放在其他章节的最前面
imports "unit1.vb", "unit2.vb"
Dim i, j = 0 ' “variables” 章节可以放在任何位置
Function p1() ' 函数
{ ,
```

```
}  
' 主程序  
For i = 0 To 10  
  pl()  
Next
```

FastScript脚本引擎的更详细的描述分别由他们的文档资料，这个手册没有重复介绍。

——所有语言都支持的格式字符

——类、属性、函数、事件的操作

——嵌套函数

——枚举和set

接下来我们适用pasical语言生成一个报表。

6. 3、“世界你好！”脚本

前面我们演示过“世界你好”的报表事例，现在让我们再用简单脚本语言来显示一个欢迎窗口。

进入到报表设计器，点击“新报表”按钮，生成一个新报表界面，然后切换到“code”页，输入以下代码：

PascalScript:

```
begin  
  ShowMessage('世界你好!');  
end.
```

C++ Script:

```
{  
  ShowMessage("世界你好!");  
}
```

然后运行报表，窗体上出现我们期望的欢迎小界面：



让我们详细说明一下。我们创建一个只有“begin...end”组成的脚本，因此，我们的脚本非常简单，他只有一个主程序。当报表生成时，主程序即开始运行，在这个事例中，他显示一个欢迎窗口，窗口关闭以后，主程序运行完毕，开始生成报表。

6. 4、脚本中组件对象的使用

报表系统可以使用任何报表上的组件，因此，在一个“Page1”页面和一个“Memo1”组件，我们使用脚本语言，可以直接调用他们的组件名称，例如：

PascalScript:

```
Memo1.Color := clRed
```

C++Script:

```
Memo1.Color = clRed
```

在报表树窗口中是脚本语言可以访问的组件列表。脚本语言可以访问组件的属性是什么？很简单，就是在对象监视器中可以看到的选择对象的属性。

举个简单的例子。我们在窗口上放置一个“Text”组件，命名为“MyTextObject”，添入如下代码：

PascalScript:

```
begin
```

```
    MyTextObject.Color := clRed
```

```
end.
```

C++Script:

```
{
```

```
    MyTextObject.Color = clRed
```

```
}
```

运行报表，会发现组件的颜色变为了红色。

6. 5、调用报表变量列表中的变量。

我们可以调用任何报表变量列表中的变量。变量名称使用<变量>符号：

PascalScript:

```
if <my variable> = 10 then ...
```

C++ Script:

```
if (<my variable> == 10) { ... }
```

另外一个可选方法就是使用“Get”函数：

PascalScript:

```
if Get('my variable') = 10 then ...
```

C++ Script:

```
if (Get("my variable") == 10) { ... }
```

这些变量值的更新只能使用“Set”函数进行更新。

PascalScript:

```
Set('my variable', 10);
```

C++ Script:

```
Set("my variable", 10);
```

还可以访问系统变量，如“Page#”，同样的方法：

PascalScript:

```
if <Page#> = 1 then ...
```

C++ Script:

```
if (<Page#> == 1) { ... }
```

6. 6、调用数据表子段

象访问变量一样，我们使用<>来访问数据表子段：

PascalScript:

```
if <Table1."Field1"> = Null then...
```

C++ Script:

```
if (<Table1."Field1"> == Null) { ... }
```

同样可以使用“Get”函数进行操作。

6. 7、脚本中调用汇总函数

汇总函数的一个特性就是如果需要在脚本中调用需要和“Text”组件配合。如果脚本单独进行调用，运行报表时会产生错误信息。这是因为这些汇总函数总是依赖于 band 进行运行的。

6. 8、报表中变量值的显示

要在报表中显示变量的值，首先需要将变量和一个常量值进行绑定：

PascalScript:

```
var
```

```
MyVariable: String;
```

```
begin
```

```
MyVariable := 'Hello!';
```


end.

C++ Script:

```
string MyVariable;

{

    MyVariable = "Hello!";

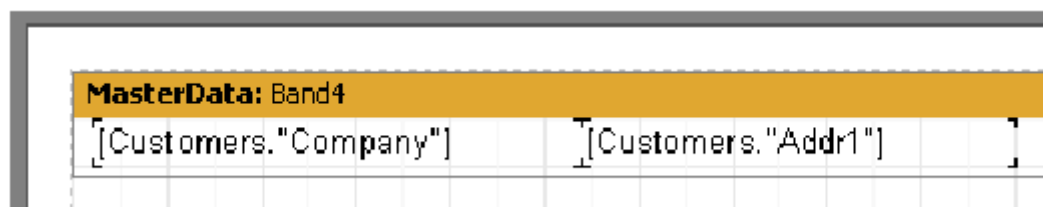
}
```

变量值可以显示在“Text”组件中，例如输入[Myvariable],变量名称应该是唯一的，意思是在报表中不能和其他组件，常量，函数名称，过程名称等相一致。如果那样，在报表时就会产生一个错误信息。

6. 9、事件

前面我们测试了报表生成开始时运行的系统主函数。他一般情况下进行一些初始化的配置，初始化变量等。但他对整个报表过程的控制是不足够的。为了尽可能的控制报表系统，每个组件都分别由自己的事件函数。例如，在 DataBand 中可以设置数据的过滤情况，这样我们在报表时就可以根据条件对数据进行隐藏或显示。

让我们示范一下报表生成过程中报表生成和事件的顺序。例如，我们生成一个简单的报表，它包含一个页，一个“MasterBand”，两个“Text”组件。



像前面规定的，主函数在报表的最前面开始运行，然后实际的报表过程才开始运行。在报表的开始，“Report”组件的“OnStartReport”事件被调用，页面打印预览之前，页面的“OnBeforePrint”事件被调用，这个事件在报表每个页面生成之间都调用一次。

Databand 生成事件调用顺序：

- 1: “OnBeforePrint”事件被调用
- 2: Band 组件上的其他组件的“OnbeforePrint”事件被调用。
- 3: 添加组件内容，然后调用组件的“OnAfterPrint”事件。
- 4: 计算 Band 的位置，高度，进行拉伸
- 5: 调用 band 的“OnAfterCaluHeight”事件。
- 6: 如果页面没有足够的空间，生成一个新的页面
- 7: 所有的 Band 和组件显示在报表中。

8: 所有 Band 上组件的 “OnAfterPrint” 事件被调用

9: Band 本身的 “OnAfterPrint” 事件被调用

当连接到 Band 的数据源进行连结后, Band 被打印显示。然后报表生成结束, 调用报表页的 “OnAfterPrint” 事件, 在调用 “Report” 组件的 “OnStopReport” 事件。

这样通过不同组件的事件, 在报表过程中可以有效的控制报表。

6. 10、一个使用 “OnBeforePrint” 事件的例子

我们创建一个报表, 显示客户信息。这个报表显示公司名以 “A” 开头的客户列表。

在 delphi 中创建一个工程, 添加一个 “TTable” 组件, 一个 “TfrxDBdata” 组件, 一个 “TfrxReport” 组件, 并设置他们的属性:

Table1:

```
DatabaseName = 'DBDEMOS'
```

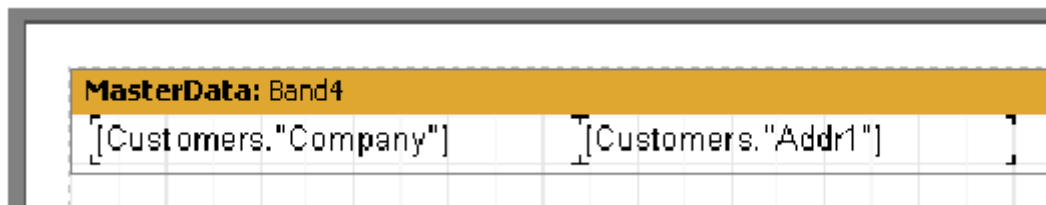
```
TableName = 'customer.db'
```

frxDBDataSet1:

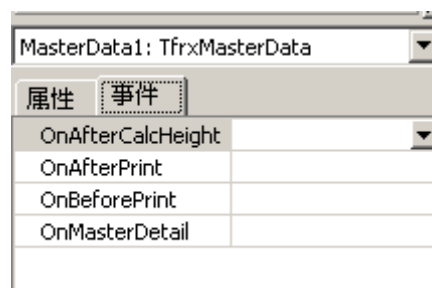
```
DataSet = Table1
```

```
UserName = 'Customers'
```

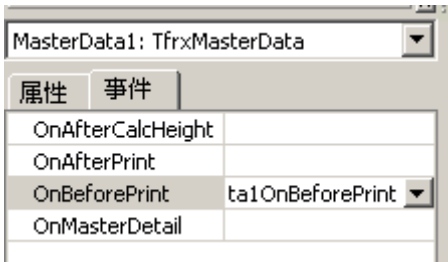
进入报表设计器, 进行编辑:



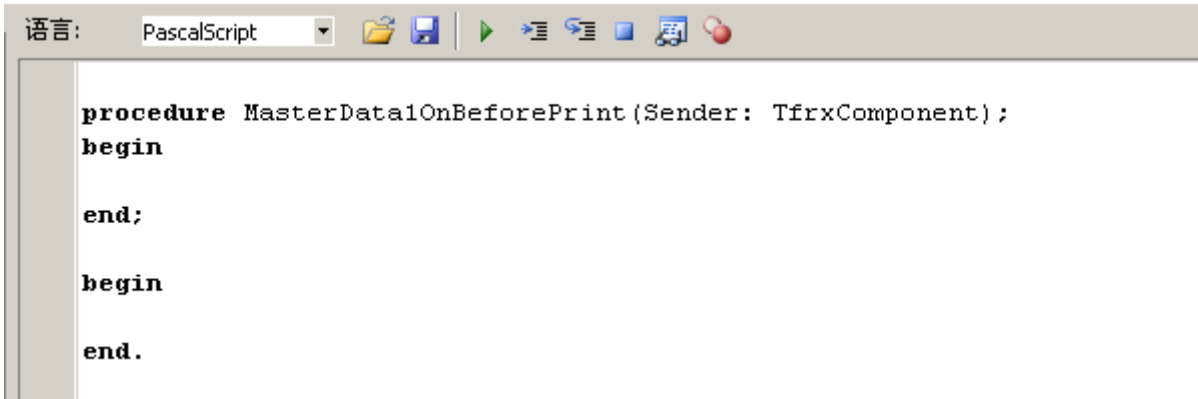
选中 DataBand, 并且换到对象查看器中的事件页:



在 OnBeforePrint 右边空白处, 双击鼠标, 生成一个事件处理器。



同时切换到 code 页，加入了一个空白处理函数。



接下来，我们在函数的空白区写下如下代码：

PascalScript:

```
if Copy(<<Customers."Company">, 1, 1) = 'A' then
    MasterData1.Visible := True else
    MasterData1.Visible := False;
```

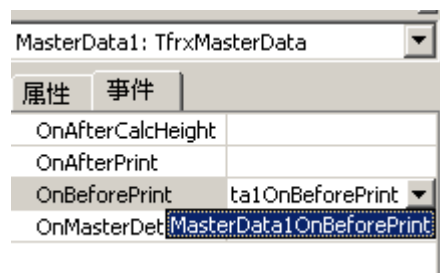
C++Script:

```
if (Copy(<<Customers."Company">, 1, 1) == "A")
    MasterData1.Visible = true;
else
    MasterData1.Visible = false;
```

运行报表：

Action Club	Michael Spurling	813-870-0239
Action Diver Supply	Marianne Miles	22-44-500211
Adventure Undersea	Gloria Gonzales	011-34-09054
American SCUBA Supply	Lynn Cinciripini	213-654-0092
Aquatic Drama	Gillian Owen	613-442-7654

我们来说明几个细节，用户可以同时将一个事件处理函数指定到多个组件上，这种情况下，函数的“sender”参数，就是调用事件的组件。设置一个已经存在的事件函数，可以通过人工录入，或在下拉筐中进行选择。



如果要取消连结，可以直接选中相关项，按“delete”即可取消。

6. 11、在组头打印组的汇总信息

这是脚本十分有用的方法，因为一般情况下，汇总信息是在报表全部信息处理完毕后进行的。要则组的开头显示汇总和的信息，一下运算方法可以使用：

- 选中报表选项中的 double pass 选项
- 第一个过程就是将每个组的计算机过保存到数组中
- 第二个过程就是将计算的结果显示在每个组的开头。

可以有两种方法来实现。首先在 delphi 中创建一个工程，在表单上添加一个“TQuery”，一个“TfrxDBData”，一个“TfrxReport”组件，设值属性：

Query1:


```
DatabaseName = 'DBDEMOS'

SQL =select * from customer, orders
      where orders.CustNo = customer.CustNo
      order by customer.CustNo, orders.OrderNo
```

frxDBDataSet1:

```
DataSet = Query1
UserName = 'Group'
```

进入报表设计器，连接数据源。设置double-pass选项（报表|选项...菜单）。在报表中添加两个Band，一个“Group header”，一个“Data band”。在“Group Header”的编辑器中指定条件 (“Group.CustNo”数据字段)，连接databand的数据源，安排组件位置：

GroupHeader: GroupHeader1		
[Group."CustNo"]	[Group."Company"]	
MasterData: MasterData1		
[Group."OrderNo"]	[Group."SaleDate"]	[Group."ItemsTotal"]
GroupFooter: GroupFooter1		
[SUM(<Group."ItemsTotal">,MasterData1)]		

为了输入汇总值，我们使用了一个“Text”组件，取名：“Memo8”

第一种方法：

我们使用“TStringList”类存储报表计算结果。把数据类型的值保存成字符串类型，同时第一行保存第一个组的值，第二行保存第二个组的值，以此类推。

脚本语言代码如下：

PascalScript:

```
var
List: TStringList;
i: Integer;
procedure frReport10nStartReport(Sender: TfrxComponent);
begin
    List := TStringList.Create;
end;
procedure frReport10nStopReport(Sender: TfrxComponent);
begin
    List.Free;
end;
procedure Page10nBeforePrint(Sender: TfrxComponent);
begin
    i := 0;
end;
procedure GroupHeader10nBeforePrint(Sender: TfrxComponent);
begin
    if Engine.FinalPass then
        Memo8.Text := 'Sum: ' + List[i];
```

```
end;  
  
procedure GroupFooter1OnBeforePrint(Sender: TfrxComponent);  
  
begin  
    if not Engine.FinalPass then  
        List.Add(FloatToStr(SUM(<Group."ItemsTotal">, MasterData1)));  
        Inc(i);  
    end;  
  
begin  
end.
```

C++ Script:

```
TStringList List;  
  
int i;  
  
void frReport1OnStartReport(TfrxComponent Sender)  
{  
    List = TStringList.Create();  
}  
  
void frReport1OnStopReport(TfrxComponent Sender)  
{  
    List.Free();  
}  
  
void Page1OnBeforePrint(TfrxComponent Sender)  
{  
    i = 0;  
}  
  
void GroupHeader1OnBeforePrint(TfrxComponent Sender)  
{  
    if (Engine.FinalPass)  
        Memo8.Text = "Sum: " + List[i];  
}  
  
void GroupFooter1OnBeforePrint(TfrxComponent Sender)
```

```
{
    List.Add(FloatToStr(SUM(<Group. "ItemsTotal">, MasterData1)));
    i++;
}
{
}
```

查看过程名称名称，可以很容易的看出我们应用的事件。他们是：“Report.OnStartReport”、“Report.OnStopReport”、“Page1.OnBeforePrint”、“GroupHeader1.OnBeforePrint”和“GroupFooter1.OnBeforePrint”。调用的前两个事件，就是说在报表开始前和结束后调用的事件。为了创建事件处理过程，在报表树中选中Report页，切换到对象查看器的事件页，双击对应的属性右边筐。

为什么没有在主函数创建“List”，而在“OnStartReport”事件中呢？那是因为报表生成以后，原有的组件将被清除。在“OnStartReport”事件中创建组件并在“OnStopReport”事件中清除组件是合理。另外，在变量的初始化可以在主函数中进行。

现在开始解释脚本代码。在页的开始，设置分组统计变量（I变量）为0，每打印一页，变量增加1。在这个事件中将计算机过添加到list变量中。在第一个阶段GroupHeader1.OnBeforePrint事件不触发；第二个阶段，将当前组的结果添加到“Memo8”中。打印预览：

1221	Kauai Dive Shoppe	Sum: 51450,8
1023	01.07.88	4 674,00
1076	16.12.94	17 781,00
1123	24.08.93	13 945,00
1169	06.07.94	9 471,95
1176	26.07.94	4 178,85
1269	16.12.94	1 400,00
		51 450,80

可以看到，方法非常简单，不过，这值简化的。

第二种方法：

使用报表变量列表存放计算结果，使用 Get 和 set 函数进行设置，代码如下：

PascalScript:

```
procedure GroupHeader1OnBeforePrint(Sender: TfrxComponent);
```

```
begin

    if Engine.FinalPass then

        Memo8.Text := 'Sum: ' + Get(<Group."CustNo">);

    end;

    procedure GroupFooter1OnBeforePrint(Sender: TfrxComponent);

    begin

        Set(<Group."CustNo">,

            FloatToStr(SUM(<Group."ItemsTotal">,MasterData1)));

    end;

    begin

    end.
```

C++ Script:

```
void GroupHeader1OnBeforePrint(TfrxComponent Sender)

{

    if (Engine.FinalPass)

        Memo8.Text = "Sum: " + Get(<Group."CustNo">);

}

void GroupFooter1OnBeforePrint(TfrxComponent Sender)

{

    Set(<Group."CustNo">,

        FloatToStr(SUM(<Group."ItemsTotal">,MasterData1)));

}

{

}
```

如你所见，这种方法更为简单，“GroupFooter1.OnBeforePrint”事件的代码，设置变量名称，如果没有这个变量，系统则生成变量，“GroupHeader1.OnBeforePrint”事件中设置其值。

6. 12、“OnAfterData”事件

当连结的报表数据源添加到报表组件后，调用“OnAfterData”事件。用这个事件分析数据

表子段或这分析组件中包含的公式。有两个 “Text” 组件，分别： [Table1."Field1"] 和 [

PascalScript:

```
if Value > 3000 then  
    Memo1.Color := clRed
```

C++ Script:

```
if (Value > 3000)  
    Memo1.Color = clRed;
```

可以写成如下代码：

PascalScript:

```
if <Table1."Field1"> > 3000 then  
    Memo1.Color := clRed
```

C++ Script:

```
if (<Table1."Field1"> > 3000)  
    Memo1.Color = clRed;
```

6. 13、Service组件

除了报表中的组件以外，还有一些组件在脚本中可以使用，如下：

- Report,Report 组件
- Engine,报表之间的联接
- OutLine,报表中和 ReportTree 的联接。

6. 13. 1、Report 组件

描述当前报表的一个联结，在报表树中的选择 “Report” 组件，可以看到这个组件的属性。

方法：

Function Calc(exp:string):variant	返回有 exp 的公式计算结果。如 Calc(1+2)，返回结果为 3。
Function GetDataSet(Alias:string):TfrxDBData	返回别名所表示的数据集，

6. 13. 2、Engine 组件

属性

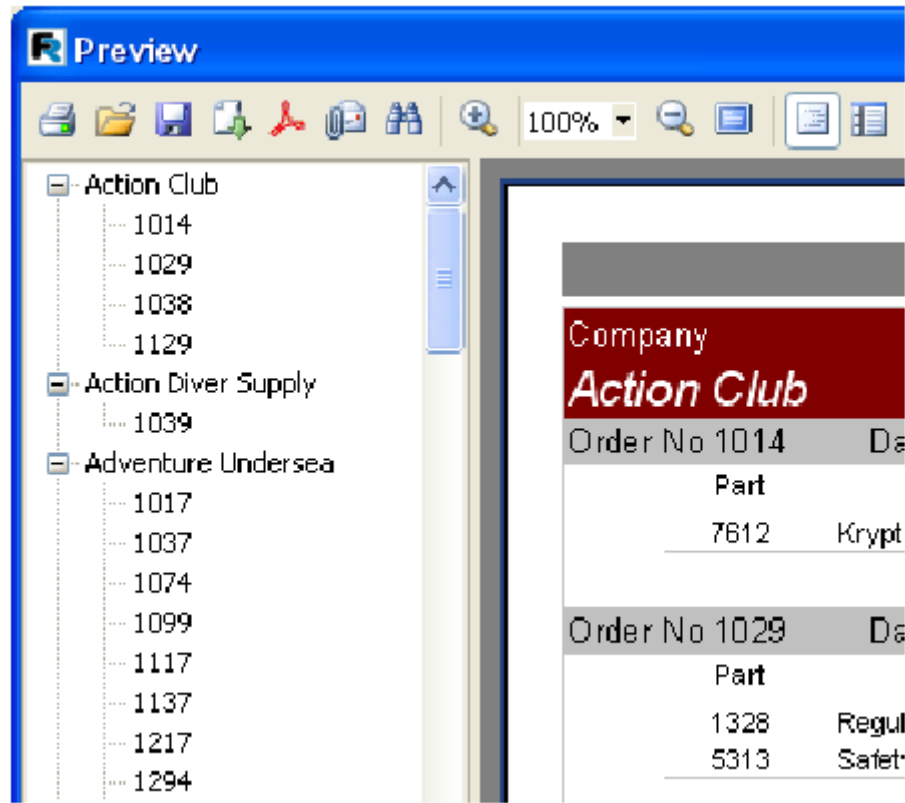
属性	类型	名称
CurColumn	Integer	当前多栏报表的栏数
CurX	Extended	当前 X 值
CurY	Extended	当前 Y 值
DoublePass	Boolean	如果是 True，是 two-pass
FinalPass	Boolean	如果是 true，是否最后一个 pass 几经运行。
PageHeight	Extended	当前页面高度
PageWidth	Extended	当前页面宽度
StartDate	TDateTime	报表开始日期
StartTime	TDateTime	报表开始时间
TotalPages	Integer	报表共有页数

方法：

方法	描述
procedure AddAnchor (const Text: String)	Adds “anchor” to the list of anchors. See more below.
procedure NewColumn	Creates a new column in a multicolumn report. After the last column, page break is automatically inserted.
procedure NewPage	Creates a new page (page break).
procedure ShowBand(Band: TfrxBand)	Displays a band with a specified name. After displaying
function FreeSpace: Extended	Returns height value of white space left on a page, in pixels.
function GetAnchorPage(const	Returns the number of the page, in which the specified

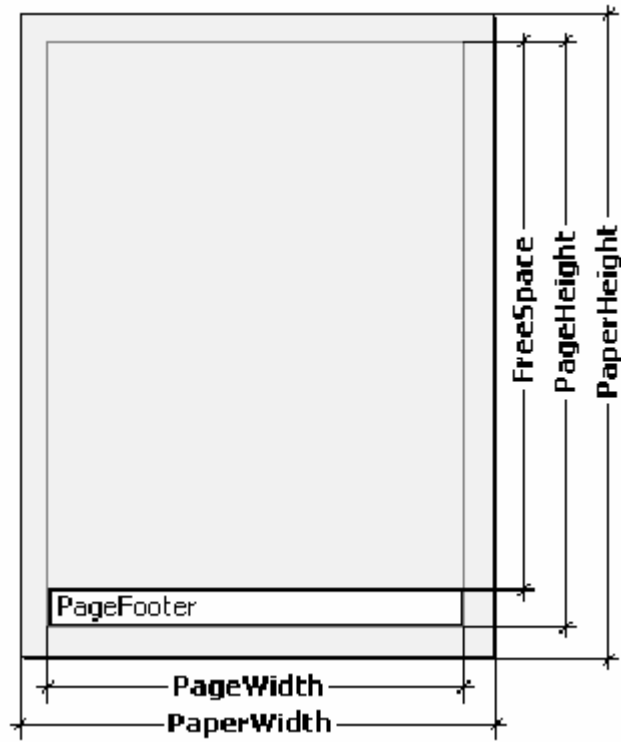
Text: String): Integer	
------------------------	--

6. 13. 3、“OutLine” 组件



6.13.4、引擎组件的使用

我们已经规定引擎组件是报表的引擎，来管理报表整体结构。通过报表引擎可以管理报表中的数据排列，先显示报表得页面和属性名称。



报表页面有“页宽”和“页高”尺寸大小，他的尺寸大小取决于选中页面，在对象查看器中显示的属性值。A4 纸的页面大小为 210X297mm。

页高和页宽决定了打印区域的范围，他不会超出页面大小，打印区域大小还依赖页面的边界（左边界LeftMargin、右边界RightMargin、上边界TopMargin、下边界BottomMargin）。使用 Engine. PageHeight和Engine.PageWidth属性，返回页面的高度和宽。

最后，“FreeSpace”属性返回一页空白区域的高。通过“Engine.FreeSpace”属性返回其值。

如何准备报表得页面，报表核心系统放置 Band 到页面有足够空间的空白位置。当除了“Footer Band”没有足够空间打印时，增加一个新的空白页面。换言之，如果打印一个 Band 区，则空白区域的高度就被减少，此外，下一个 band 的位置从当前位置开始。当前位置可以有“Engine.CurX”，“Engine.CurY”属性返回。当打印一个 band 后，则 CurY 自动增加一个当前打印的 Band 的高度。如果生成一个新页，CurY 重设为 0。CurX 在多栏报表中被修改。

“Engine.CurX”，“Engine.CurY”属性不但可以读取，而且还可以写回。意思是说，Band 的位置可以通过手动进行位移。

MasterData: MasterData1									
Customers."Company"				Customers."Contact"			Customers."Phone"		

打印结果如下：

Action Club	Michael Spurling	813-870-0239
Action Diver Supply	Marianne Miles	22-44-500211
Adventure Undersea	Gloria Gonzales	011-34-09054
American SCUBA Supply	Lynn Cinciripini	213-654-0092
Aquatic Drama	Gillian Owen	613-442-7654
Blue Glass Happiness	Christine Taylor	213-555-1984

以下是在 OnBeforePrint 中填写的代码:

PascalScript:

```
procedure MasterData1OnBeforePrint(Sender: TfrxComponent);  
  
begin  
  
    Engine.CurX := Engine.CurX + 5;  
  
end;
```

C++ Script:

```
void MasterData1OnBeforePrint(TfrxComponent Sender)  
{  
  
    Engine.CurX = Engine.CurX + 5;  
  
}
```

CurY 属性页可以被修改。形成界面如下:

Action Club	Michael Spurling	813-870-0239
Action Diver Supply	Marianne Miles	22-44-500211
Adventure Undersea	Gloria Gonzales	011-34-09054
American SCUBA Supply	Lynn Cinciripini	213-654-0092
Aquatic Drama	Gillian Owen	613-442-7654
Blue Glass Happiness	Christine Taylor	213-555-1984
Blue Jack Aqua Center	Ernest Barrett	401-609-7673
Blue Sports Club	Theresa Kuehn	813-867-8604

代码如下:

PascalScript:

```
procedure MasterData1OnBeforePrint(Sender: TfrxComponent);  
  
begin  
  
    Engine.CurY := Engine.CurY - 15;  
  
end;
```

C++ Script:

```
void MasterData1OnBeforePrint (TfrxComponent Sender)

{

    Engine.CurY = Engine.CurY - 15;

}
```

“Engine.NewPage”允许用户根据需要在任何地方进行强制分页，同时从新的页面开始打印，如每两条记录强制分页。

代码如下：

PascalScript:

```
procedure MasterData1OnAfterPrint (Sender: TfrxComponent);

begin

    if <Line> mod 2 = 0 then

        Engine.NewPage;

end;
```

C++ Script:

```
void MasterData1OnAfterPrint (TfrxComponent Sender)

{

    if (<Line> \ 2 = 0)

        Engine.NewPage();

}
```

“Engine.NewColumn”属性用在多栏报表中进行强制分栏，当页面没有空白位置进行分栏，责另起一页打印。

6.15 Anchors

Anchor 是超链接系统中的一个元素，在终极报表（预览窗口）中可以通过点击快速的调转到相应的位置。

Anchor是一种特技，可以通过Engine.AddAnchor方法添加，Anchor具有名称，对应着报表页面的位置。如果要调转到Anchor指定的位置，可以通过组件的URL属性进行设置。

#AnchorName

or

#[AnchorName]

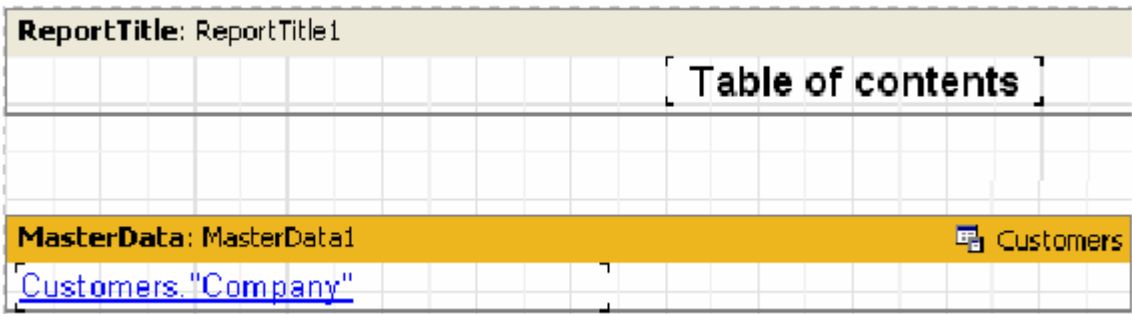
最后，FastReport 将用公式计算结果替代。

点击组件，就能调转 Anchor 标示的报表位置。

当有章节目录时，可以使用 Anchor。

例如，我们在第一页填写报表目录，客户信息从报表的第二页开始，通过报表目录能够快速浏览的相应的记录信息。

第一页设计：

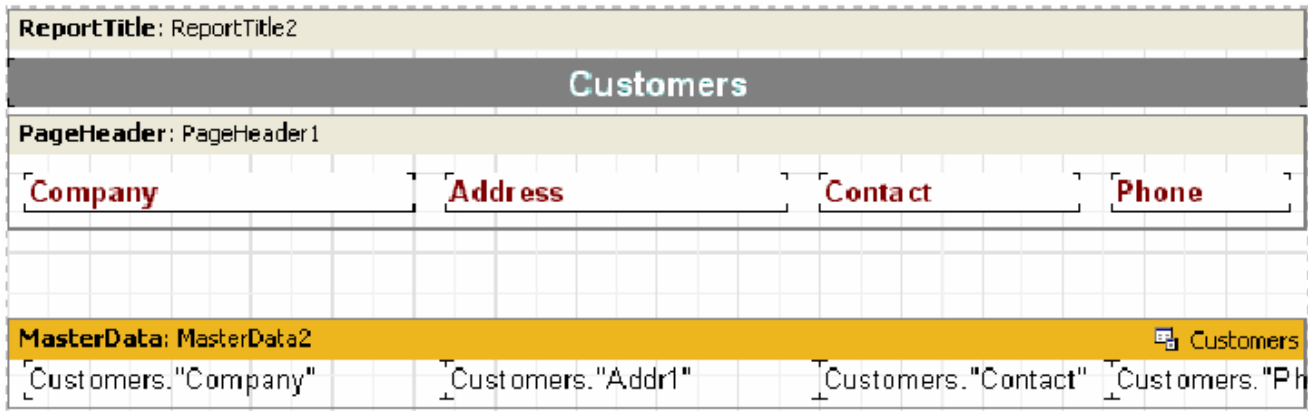


我们在 DataBand 上的“Text”组件的 URL 属性填写如下：

`#[Customers."Company"]`

并设置字体颜色，和样式为下划线字体。

第二个设计页



创建MasterData2.OnBeforePrint事件，填写代码：

PascalScript:

```
procedure MasterData2OnBeforePrint(Sender: TfrxComponent);  
  
begin  
    Engine.AddAnchor(<Customers."Company">);  
  
end;
```

C++ Script:

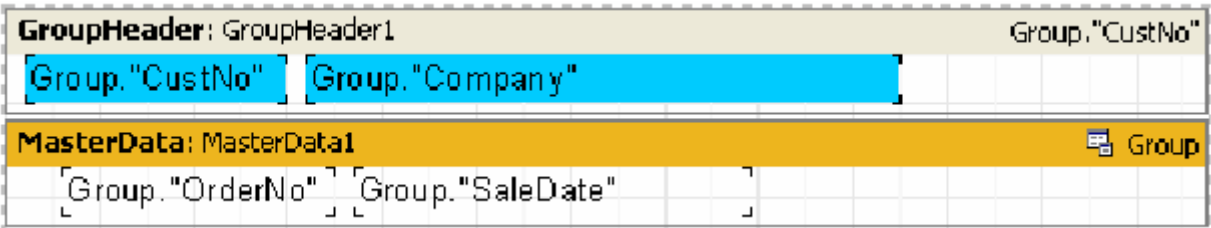
```
void MasterData2OnBeforePrint (TfrxComponent Sender)
{
    Engine.AddAnchor (<Customers."Company">);
}
```

最后一个要说明的是Engine.GetAnchorPage函数，返回Anchor名称所在的页码，这在形成目录结构中是很有用的。

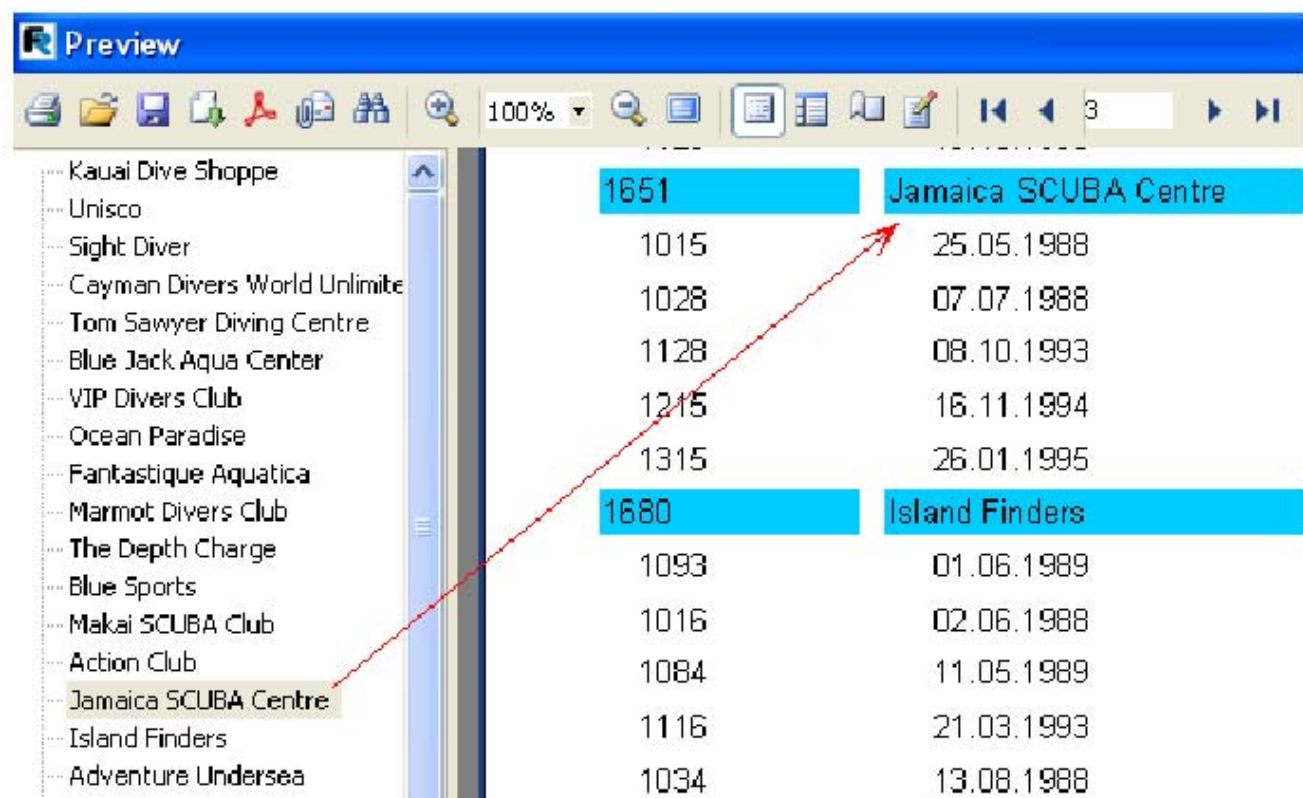
6.16 “Outline” 组件的使用方法

OutLine组件用于管理在报表预览窗口中报表树的结构，通过单击这个报表树可以快速跳转到报表的相应的页面中。使用脚本控制Outline不是必需的，因为Band中有这么一种机制，自动形成报表树的结构。

几乎所有的Band都有“OutLine Text” 属性，应用他可以自动生成报表树。这些公式在报表是会被计算，并且在打印Band的时候被添加到报表树中。例：

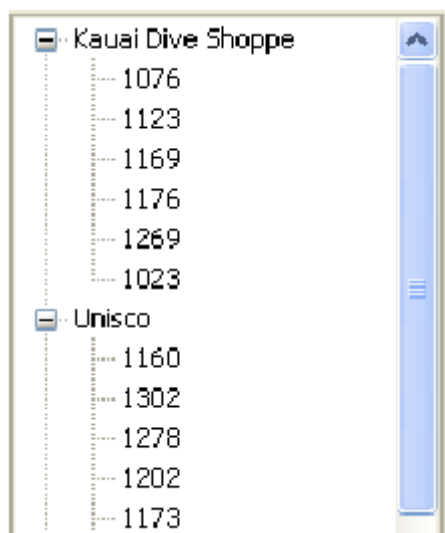


指定GroupHeader的OutLineText属性为：“<Group."Company">.”。当报表预览打开时，自动形成树。界面参考如下：



在左边树中点击节点，报表快速跳转到相应的位置，并且显示在报表的最上面。

在Master band的Outlinetext属性设置“<Group."OrderNo">.”，形成二级报表树结构，界面参考如下：



现在使用脚本而不是用属性值而生成相类似的报表树，上面的例子去掉outlinetext属性值，而添加GroupHeader1.OnBeforePrint 和 MasterData1.OnBeforePrint两个事件。

PascalScript:

```
procedure GroupHeader1OnBeforePrint(Sender: TfrxComponent);
```

```
begin

    Outline.LevelRoot;

    Outline.AddItem(<Group."Company">);

end;

procedure MasterData1OnBeforePrint(Sender: TfrxComponent);

begin

    Outline.AddItem(<Group."OrderNo">);

    Outline.LevelUp;

end;

begin

end.
```

C++ Script:

```
void GroupHeader1OnBeforePrint(TfrxComponent Sender)
{
    Outline.LevelRoot;

    Outline.AddItem(<Group."Company">);
}

void MasterData1OnBeforePrint(TfrxComponent Sender)
{
    Outline.AddItem(<Group."OrderNo">);

    Outline.LevelUp;
}

{

}
```

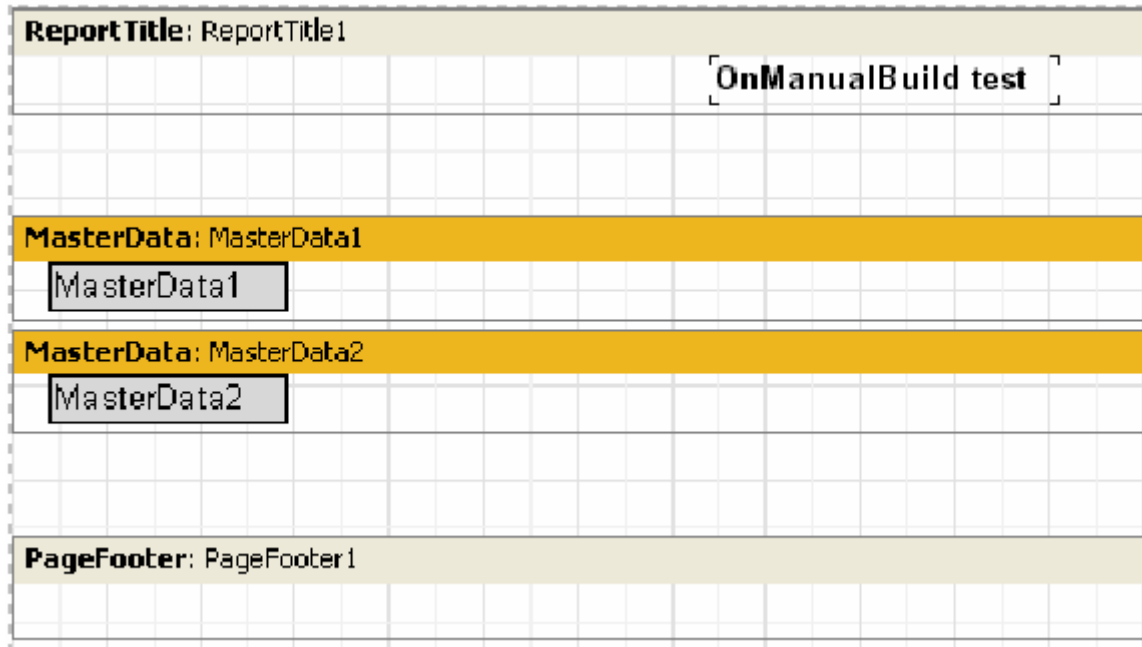
6.17 “OnManualBuild” 页面事件

FastReport核心对报表结构是可靠的，按显示定义顺序显示报表的Band。有时需要显示一些非标准得页面，这是FastReport核心不能胜任。在设计页面借助OnManualBuild事件的能力创建报表。假如这个事件已经定义，生成报表是控制权就交给了这个处理函数。同时，报表核心自动在页面上放置Band，如"Report title," "Page title," "Column title," "Report footer,""Page footer,"

"Column footer," and "Background."。报表核心一样处理新的页面和新的栏。

这说，“OnManualBuild”处理函数本质是处理显示特殊Band到FastReport的核心，核心rest，当没有空白尾之后，形成一个新的页面。处礼服在事件中的脚本。

我们做一个示范，在报表上，有两个还没有联接数据的Master 数据bands。



PascalScript:

```
procedure Page1OnManualBuild(Sender: TfrxComponent);
var
    i: Integer;
begin
    for i := 1 to 6 do
    begin
        Engine.ShowBand(MasterData1);
        Engine.ShowBand(MasterData2);
        if i = 3 then
            Engine.CurY := Engine.CurY + 10;
        end;
    end;
end;
```

C++ Script:

```
void Page1OnManualBuild(TfrxComponent Sender)
{
```

```

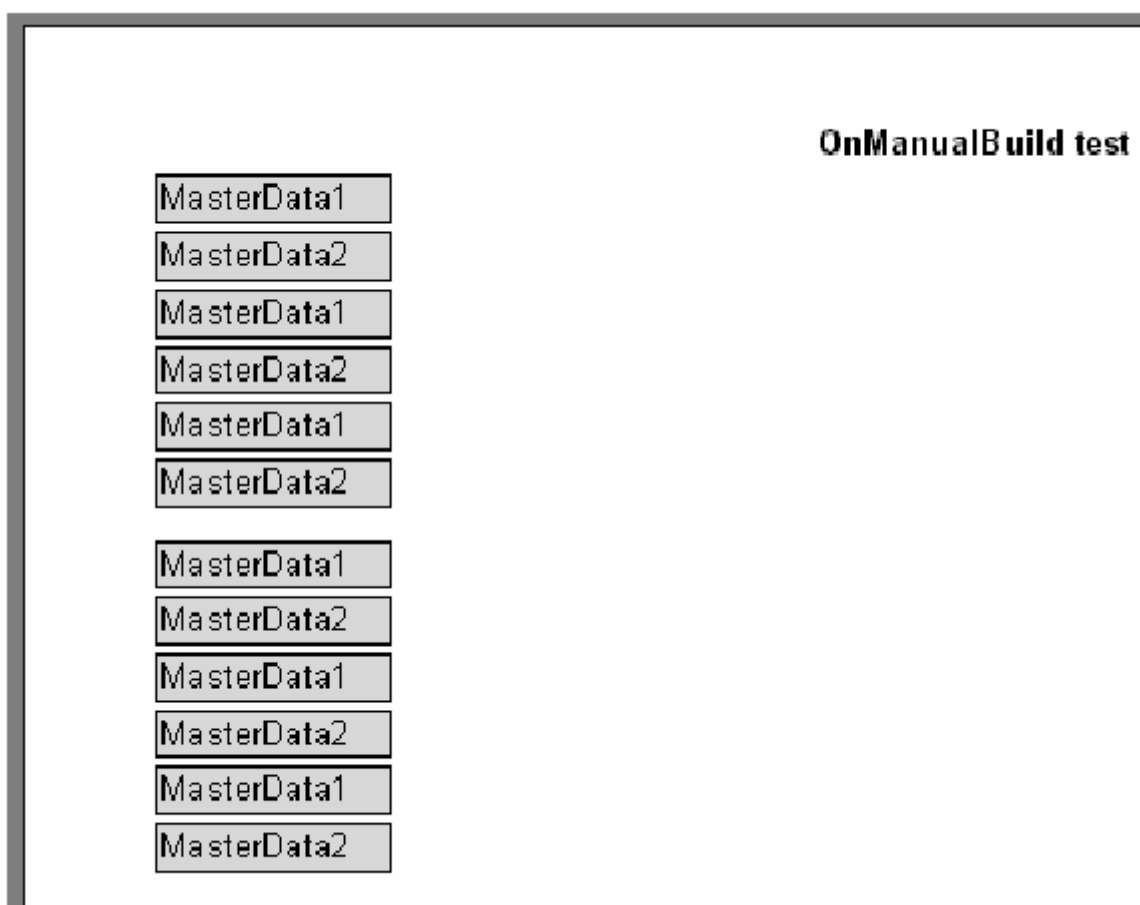
int i;

for (i = 1; i <= 6; i++)
{
    Engine.ShowBand(MasterData1);

    Engine.ShowBand(MasterData2);

    if (i == 3)
        Engine.CurY = Engine.CurY + 10;
}
}

```



以下代码显示两个band组的下一个。

PascalScript:

```

procedure Page1OnManualBuild(Sender: TfrxComponent);

var

    i, j: Integer;

```

```
SaveY: Extended;

begin

    SaveY := Engine.CurY;

    for j := 1 to 2 do

        begin

            for i := 1 to 6 do

                begin

                    Engine.ShowBand(MasterData1);

                    Engine.ShowBand(MasterData2);

                    if i = 3 then

                        Engine.CurY := Engine.CurY + 10;

                end;

            Engine.CurY := SaveY;

            Engine.CurX := Engine.CurX + 200;

        end;

    end;
```

C++Script:

```
void Page1OnManualBuild(TfrxComponent Sender)

{

    int i, j;

    Extended SaveY;

    SaveY = Engine.CurY;

    for (j = 1; j <= 2; j++)

    {

        for (i = 1; i <= 6; i++)

        {

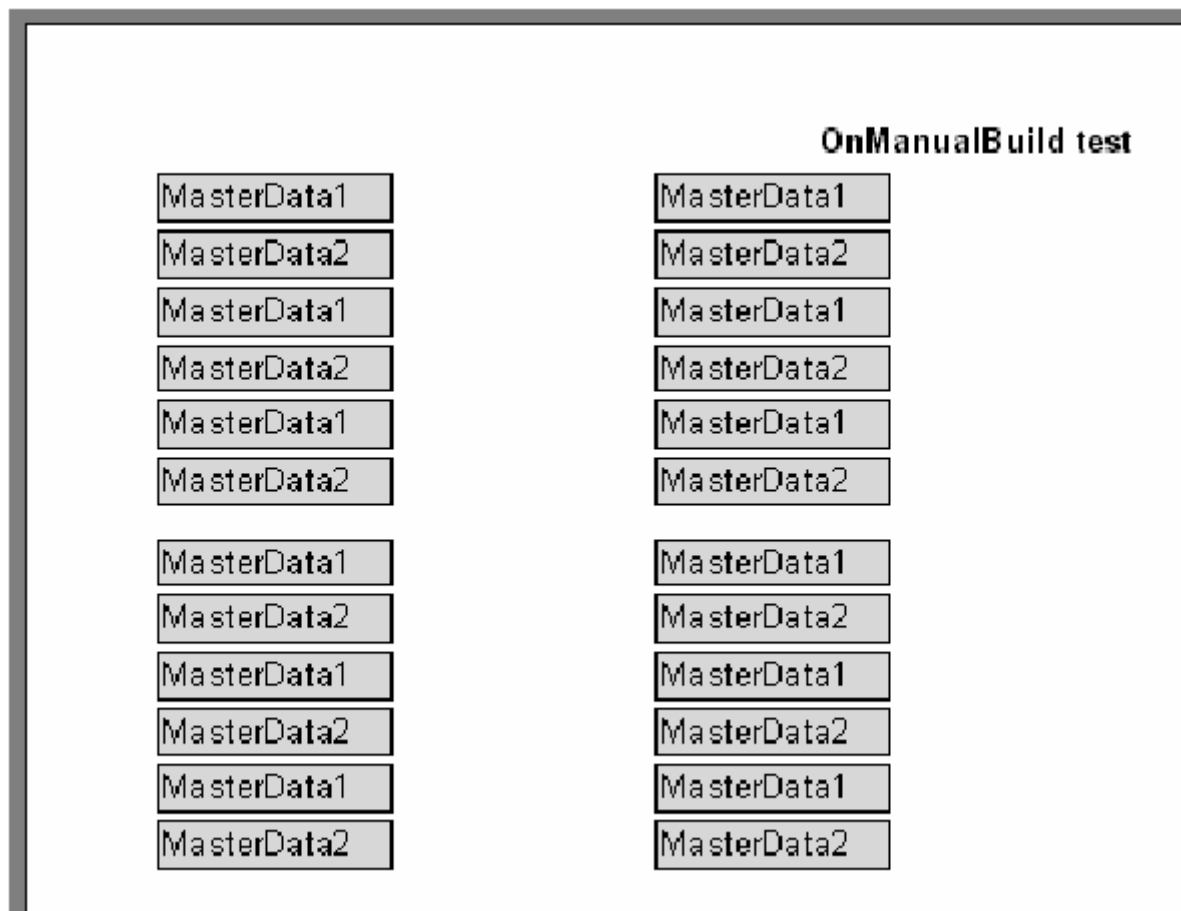
            Engine.ShowBand(MasterData1);

            Engine.ShowBand(MasterData2);

            if (i == 3)

                Engine.CurY = Engine.CurY + 10;
```

```
    }  
  
    Engine.CurY = SaveY;  
  
    Engine.CurX = Engine.CurX + 200;  
  
    }  
  
}
```



如你所见到的,在这个例子中我们只是控制数据Band。全部剩余band被自动打印。最后,我们将描述怎样使用”OnManualBuild”事件,建立一个顾客列表类型的报表。

ReportTitle: Band1		
Customers		
PageHeader: Band2		
Company	Contact	Phone
MasterData: MasterData1		
Customers."Company"	Customers."Contact"	Customers."Ph"
PageFooter: Band3		

以下是事件的脚本语言：

PascalScript:

```

procedure Page1OnManualBuild(Sender: TfrxComponent);

var

    DataSet: TfrxDataSet;

begin

    DataSet := MasterData1.DataSet;

    DataSet.First;

    while not DataSet.Eof do

        begin

            Engine.ShowBand(MasterData1);

            DataSet.Next;

        end;

    end;

```

C++Script:

```

void Page1OnManualBuild(TfrxComponent Sender)

{

    TfrxDataSet DataSet;

    DataSet = MasterData1.DataSet;

```

```
DataSet.First();

while (!DataSet.Eof)
{
    Engine.ShowBand(MasterData1);

    DataSet.Next();
}
}
```

预栏报表，确定脚本工作和使用标准报表有什么不同，注：如何得到联接的数据集，这个事例中，我们联结一个band到数据源。如果Band没有联接到数据源，需要的连接通过如下代码实现：

```
DataSet := MasterData1.DataSet;

DataSet := Report.GetDataSet('Customers');
```

6.18 脚本中的组件的建立

我们可以使用脚本在报表中添加新的组件，让我们通过一个事例，来描述一下如何实现。首先创建一个新的报表，并在主过程中写入一下代码：

PascalScript:

```
var

    Band: TfrxReportTitle;

    Memo: TfrxMemoView;

begin

    Band := TfrxReportTitle.Create(Page1);

    Band.Height := 20;

    Memo := TfrxMemoView.Create(Band);

    Memo.SetBounds(10, 0, 100, 20);

    Memo.Text := 'This memo is created in code';

end.
```

C++ Script:

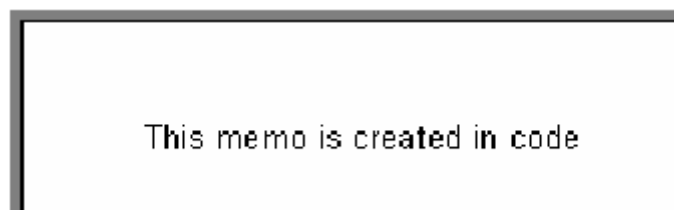
```
TfrxReportTitle Band;

TfrxMemoView Memo;
```



```
{  
  
    Band = TfrxReportTitle.Create(Page1);  
  
    Band.Height = 20;  
  
    Memo = TfrxMemoView.Create(Band);  
  
    Memo.SetBounds(10, 0, 100, 20);  
  
    Memo.Text = "This memo is created in code";  
  
}
```

预览结果是：



注：在事例中，我们没有销毁组件事例，因为也没有这个必要，当报表完成后，这些组件将自动销毁。

第 七 章

交叉报表

这种报表类型是表格结构，意思是它包含着一系列的行和列。同时它不可预知，表格有多少个行和列。这就是为什么报表不但纵向增加，而且横向增加。一下显示这种类型的报表的例子：

表格的数据如下：

	1	2	3	4
a	a1	a2	a3	a4
b	b1	b2	b3	b4

这个例子中，表格有两个行和四个列。A 和 b 行的标题，1,2,3,4 是列的标题。A1,a2..a4,b1..b4 是表格中的数据。我们创建一个数据集，有三个字段，包含如下内容：

a 1 a1
a 2 a2
a 3 a3
a 4 a4
b 1 b1
b 2 b2
b 3 b3
b 4 b4

第一个字段为行序号，第二个字段为列序号，第三个字段为表格数据。当输出报表时，FastReport 在内存创建一个表格，并填入数据。

标题如果有两层，先是数据如下：

	10		20	
	1	2	1	2
a	a10.1	a10.2	a20.1	a20.2
b	b10.1	b10.2	b20.1	b20.2

报表需要如下数据：

a 10 1 a10.1
a 10 2 a10.2
a 20 1 a20.1
a 20 2 a20.2
b 10 1 b10.1

```

b 10  2  b10.2
b 20  1  b20.1
b 20  2  b20.2

```

内存中创建的表格数据如下：

	10	10	20	20
	1	2	1	2
a	a10.1	a10.2	a20.1	a20.2
b	b10.1	b10.2	b20.1	b20.2

7. 1、创建交叉报表

现在我们从理论转到实践。我们创建一个简单的报表，数据包含四年来的雇员的薪水。这样，我们需要一个“CrossTest”的数据表，数据包含如下内容：

<u>Name</u>	<u>Year</u>	<u>Salary</u>
Ann	1999	3300
Ben	2002	2000
....		

在 delphi 中创建一个新的工程，添加“TTable”，“TfrxDBData”和“TfrxReport”组件。并设置其属性：

Table1:

```

DatabaseName = 'c:\Program Files\FastReport 4\Demos\Main'
TableName = 'crosstest.db'


```


frxDBDataSet1:

```

DataSet = Table1
UserName = 'SimpleCross'

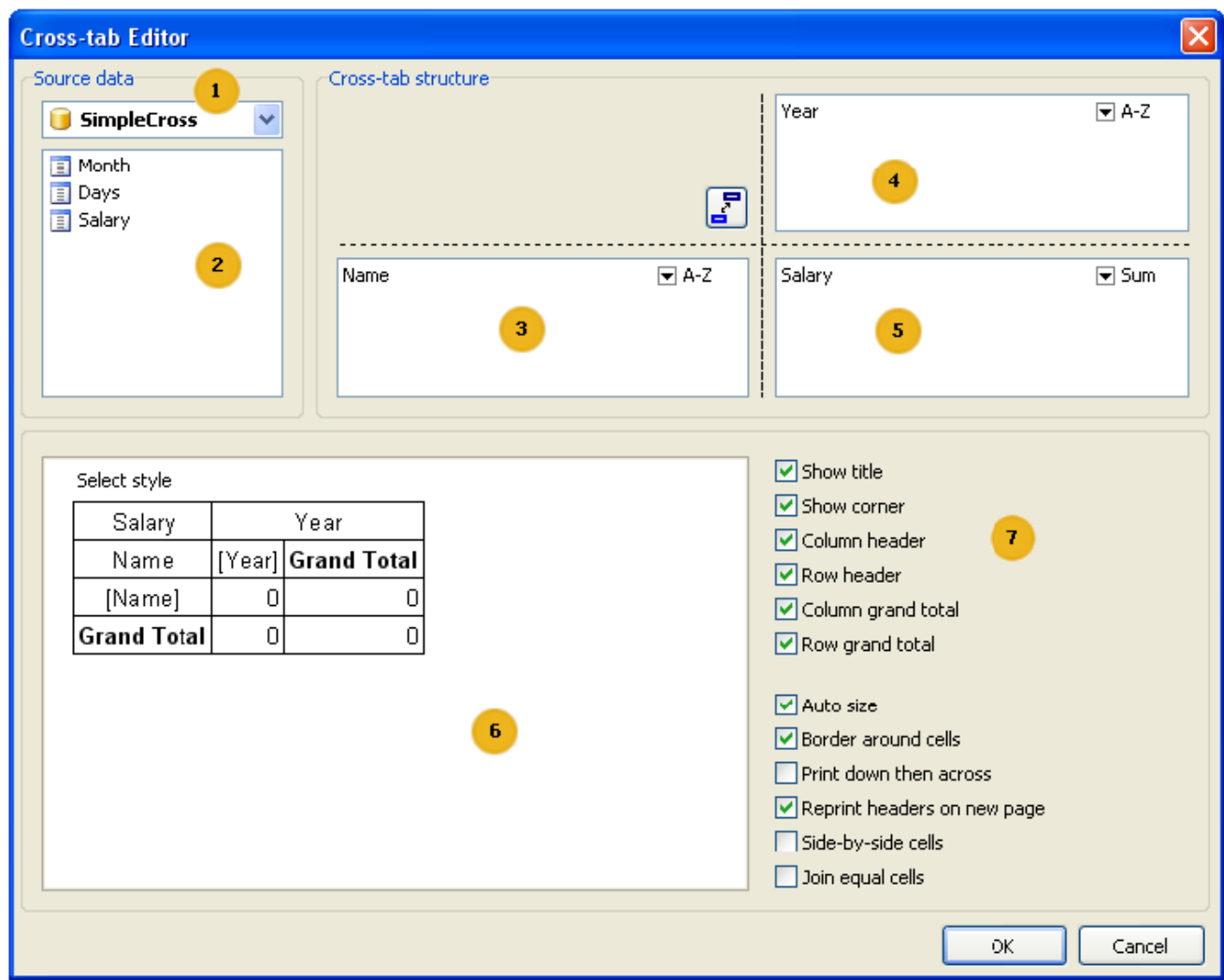
```

为了完成交叉报表，我们需要使用组件“TfrxCrossObject”，此组件在 FastReport 组件面板上。将他添加到 delphi 的表单中，不需要设置任何属性；同时，包含全部可用函数的“frxCross”单元被添加到 uses 列表中。

进入报表设计器界面，首先连接数据源，再在报表中添加“Db Cross tab”组件.



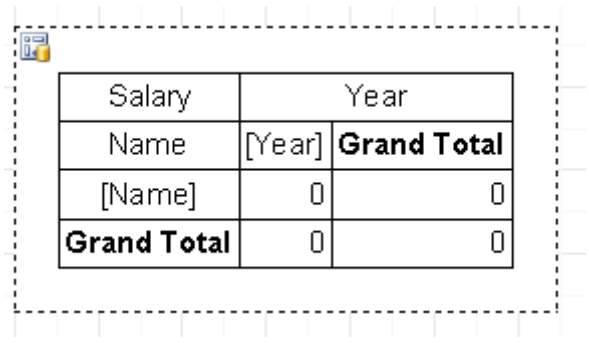
通过事件编辑器，设置组件的相关属性。双击组件可以打开编辑器界面。



图形中数字区说明

- 1: 可用的数据源下拉列表
- 2: 选择数据源的字段列表。字段可以被拖拽到 3、4、5 区域中。
- 3: 生成行标题的字段列表。
- 4: 生成列标题的字段列表。
- 5: 显示表格数据的字段列表。
- 6: 表格结构预览
- 7: 可选项。

在这个界面中，只能通过鼠标进行操作修改。在这个事例中，可以托动鼠标将 2 区域的字段列表拖拽到 3、4、5 区域。然后点击确定，显示结构如下：




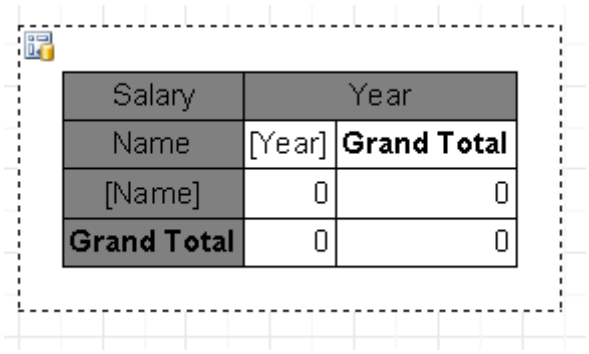
Salary	Year	
Name	[Year]	Grand Total
[Name]	0	0
Grand Total	0	0

预览报表，显示界面如下：

Salary	Year				
Name	1999	2000	2001	2002	Grand Total
Ann	3300	2700	3100	1700	10800
Ben	4300	2400		2000	8700
Catherine	6100	3200			9300
Den		3999	8100		12099
Grand Total	13700	12299	11200	3700	40899

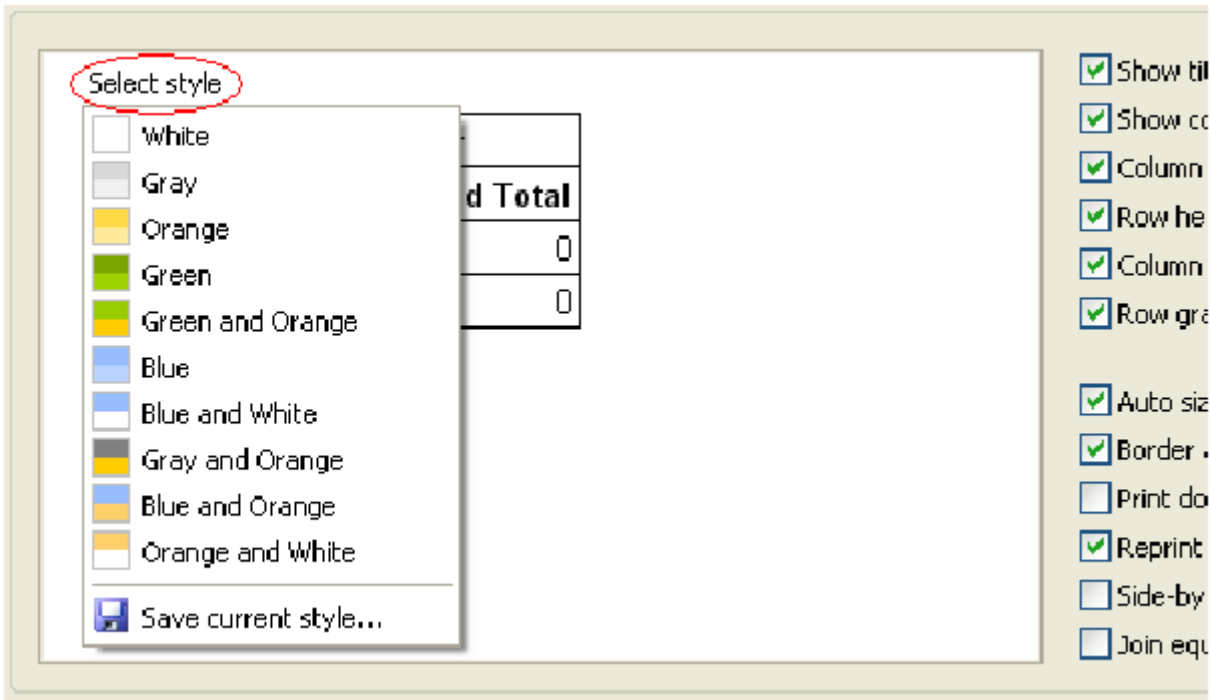
7. 2、改变显示

让我们改变组件的显示模式。首先我们要做的就是改变标题的颜色，并改变 Grand Total 改变为“汇总”。给边非常简单，一次选中“year”，“Name”，“Grand Total” 组件，点击按钮设置颜色，



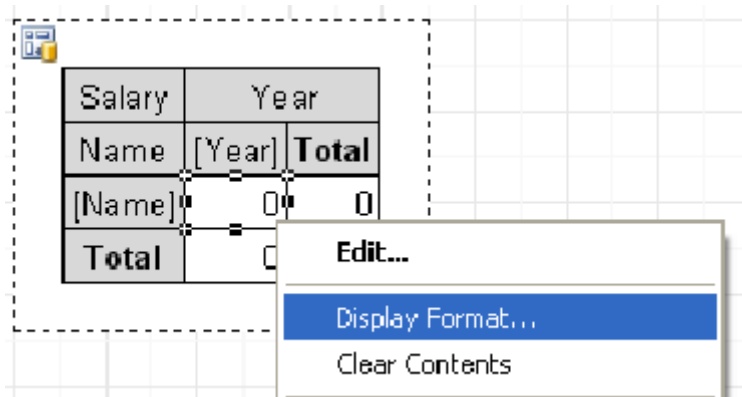
Salary	Year	
Name	[Year]	Grand Total
[Name]	0	0
Grand Total	0	0

页可以在组件的编辑器中选择你喜欢的样式。



改变“Grand Total” 标题，上击要修改的组件，从弹出的编辑窗口中修改文字即可。

格式化显示结果，可选选中第一个单元格（year 和 name 的交叉位置）。点击右键从菜单中选择“display format”。



选中需要得显示格式，然后关闭格式化界面，显示结果如下图：

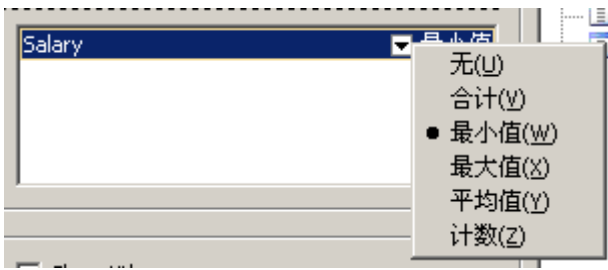
Salary	Year				
Name	1999	2000	2001	2002	Grand Total
Ann	¥ 3,300.00	¥ 2,700.00	¥ 3,100.00	¥ 1,700.00	¥ 10,800.00
Ben	¥ 4,300.00	¥ 2,400.00		¥ 2,000.00	¥ 8,700.00
Catherine	¥ 6,100.00	¥ 3,200.00			¥ 9,300.00
Den		¥ 3,999.00	¥ 8,100.00		¥ 12,099.00
Grand Total	¥ 13,700.00	¥ 12,299.00	¥ 11,200.00	¥ 3,700.00	¥ 40,899.00

7. 3、使用函数

在上面的事例中，我们可以看到员工四年的工资以及工资汇总情况，我们还可以使用一下函数：

- sum:汇总值和
- Max:求最大值
- Min:求最小值
- Avg:求平均值。
- Count:计数

我们事例 MIN 函数，打开交叉组件编辑器，在 6 区选择 salary,



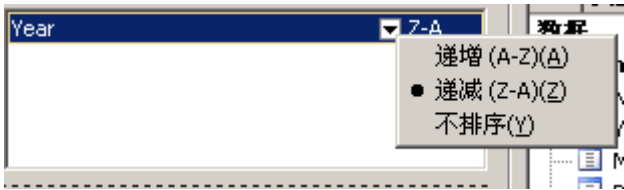
从菜单中选择最小值。把表格中的汇总字符改为最小值。然后显示结果：

Salary	Year				
Name	1999	2000	2001	2002	最小值
Ann	¥ 3,300.00	¥ 2,700.00	¥ 3,100.00	¥ 1,700.00	¥ 1,700.00
Ben	¥ 4,300.00	¥ 2,400.00		¥ 2,000.00	¥ 2,000.00
Catherine	¥ 6,100.00	¥ 3,200.00			¥ 3,200.00
Den		¥ 3,999.00	¥ 8,100.00		¥ 3,999.00
最小值	¥ 3,300.00	¥ 2,400.00	¥ 3,100.00	¥ 1,700.00	¥ 1,700.00

7. 4、对结果进行排序

对行和列进行升序排序，如果结果是数字型，则按数据进行排序，如果是字符型，则按字母顺序排序。我们可以对行和列分别进行设置排序模式。

我们做个示范，让 year 进行降序排列，设置界面如图（组件设计器）：



显示结果：

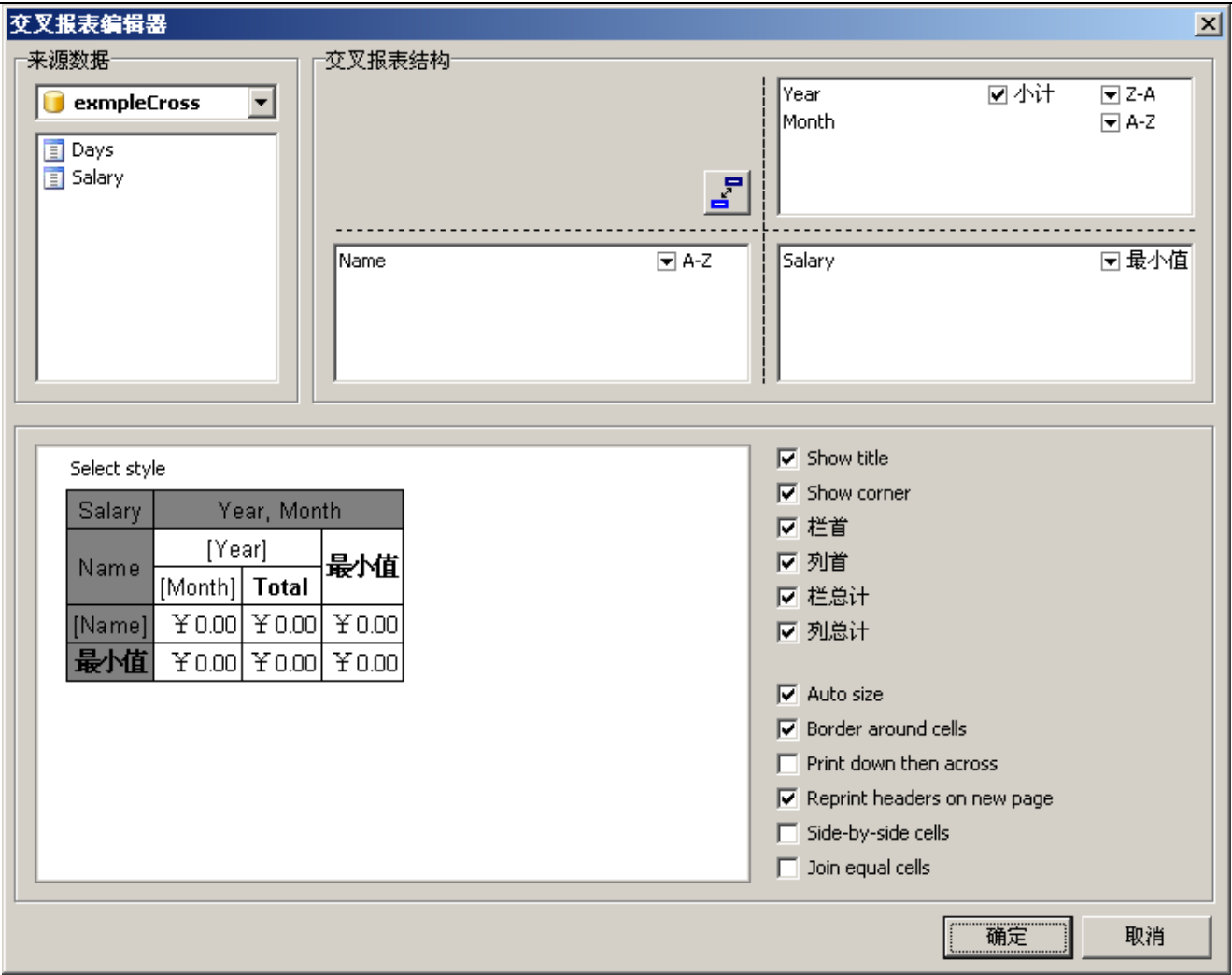
Salary	Year				
Name	2002	2001	2000	1999	最小值
Ann	¥ 1,700.00	¥ 3,100.00	¥ 2,700.00	¥ 3,300.00	¥ 1,700.00
Ben	¥ 2,000.00		¥ 2,400.00	¥ 4,300.00	¥ 2,000.00
Catherine			¥ 3,200.00	¥ 6,100.00	¥ 3,200.00
Den		¥ 8,100.00	¥ 3,999.00		¥ 3,999.00
最小值	¥ 1,700.00	¥ 3,100.00	¥ 2,400.00	¥ 3,300.00	¥ 1,700.00

7. 5、组合标题的表格

我们的事例只有一个行头和一个列首。现在我们设计一个复杂标题的报表，它可能包含两个行或列数据。表格包含如下数据内容：

<u>Name</u>	<u>Year</u>	<u>Month</u>	<u>Days</u>	<u>Salary</u>
Ann	1999	2	3	1000
Ben	2002	1	5	2000
....				

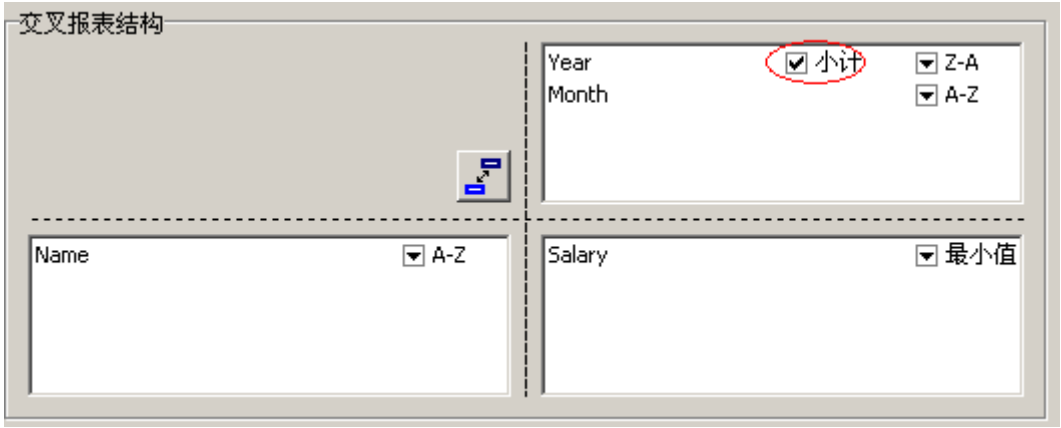
设置界面如下：



结果显示界面如下：

	1999					2000				2001				2002		Grand Total
	2	10	11	12	Total	1	2	3	Total	1	2	3	Total	1	Total	
Ann	1000		1100	1200	3300	1300	1400		2700		1500	1600	3100	1700	1700	10800
Ben		2100	2200		4300		2400		2400				0	2000	2000	8700
Catherine		3000	3100		6100			3200	3200				0		0	9300
Den					0	3999			3999	4000	4100		8100		0	12099
Grand Total	1000	5100	6400	1200	13700	5299	3800	3200	12299	4000	5600	1600	11200	3700	3700	40899

注：报表自动在每个 year 最后做一个总计，这个可以在组件编辑器中设置 year 的“小计”标识。



另外注意，在最后一列我们没有做中间的汇总统计，实际上，在我们的事例中也没有这个必要。

7. 6、调整单元格的宽度

察看前面的事例，可以很清晰的发现，单元格可以自动调整其宽度，以适应最大字符宽度。然而有时却不尽人意，如果字符特别的长，这时报表的样式就变得非常难看。如何解决这种情况呢？让我们采用 3 种方法进行处理。

最简单的方法就是字组件中，将字符设成多行。

Total

for[Value]

可以看到结果如下：

	1999				
	2	10	11	12	Total for 1999
Ann	1000		1100	1200	3300
Ben		2100	2200		4300
Catherine		3000	3100		6100
Den					0
Grand Total	1000	5100	6400	1200	13700

然而，如果行或列的结果值的字符串长度非常大时，这种方法就不适应了。这就是 cross-tab 组件为什么有”Minwidth”,”MaxWidth”属性参数了。

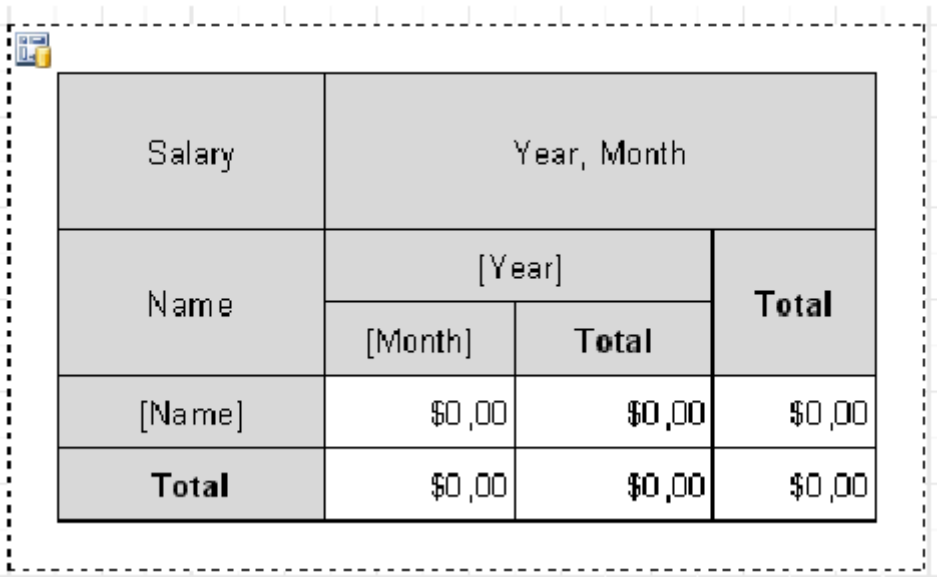
默认情况下，Minwidth 值为 0，MaxWidth 属性值是 200，这可以在大部分情况下都适用。用户可以根据实际需要进行对这个值进行设置。

这样，在我们的事例中，我们设置 Minwidth=MaxWidth=50；意思是说单元格在任何情况

下都是 50 个像素值，如果字符串比较小，系统自动调整宽度到 50，如果字符串长度比较大，系统还是调整宽度到 50，同时，长的部分被分割。显示结果如下图：

	1999				
	2	10	11	12	Total for 1999
Ann	1000		1100	1200	3300
Ben		2100	2200		4300
Catherine		3000	3100		6100
Den					0
Grand Total	1000	5100	6400	1200	13700

第三种就是手动调整单元格宽度。这时，需要设置 AutoSize 属性值为 false。现在可以通过鼠标改变组件的大小。显示界面如下：



Salary	Year, Month		
Name	[Year]		Total
	[Month]	Total	
[Name]	\$0,00	\$0,00	\$0,00
Total	\$0,00	\$0,00	\$0,00


记住，如果取消了组件的自动大小属性，则在报表时，系统不能在调整单元格的宽度和高度，预览时的结果可能如下图：

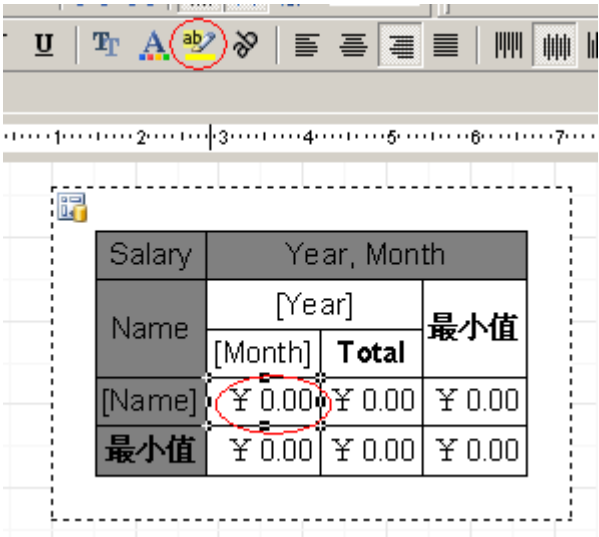
Salary				
Name	1999			
	2	10	11	
Ann	\$1 000,0 n		\$1 100,0 n	
Ben		\$1 900,0 n	\$2 000,0 n	
Catherine		\$3 000,0 n	\$3 100,0 n	

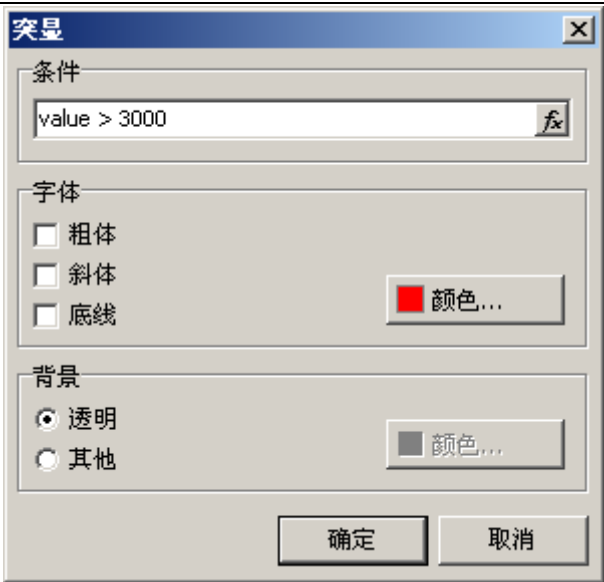
这时，稍微在调整一下单元格的宽度即可。

7. 7、字体颜色和突出显示

有时我们需要突出显示某些结果值，或通过字体颜色进行区分。我们在组报表中测试过突出显示方法。我们使用条件进行设置突出显示，这在报表中是非常有用的。

添加突出显示：在我们的事例中，我们将结果值大于 3000 的字体颜色改为红色，选中单元格，在工具览中点击按钮，弹出条件设置对话框。设置参数 value>3000，并改变字体颜色为红色。





点击确定关闭窗口，进行预览：

	1999					2000			
	2	10	11	12	Total for 1999	1	2	3	Total for 2000
Ann	1000		1100	1200	3300	1300	1400		2700
Ben		2100	2200		4300		2400		2400
Catherine		3000	3100		6100			3200	3200
Den					0	3999			3999
Grand Total	1000	5100	6400	1200	13700	5299	3800	3200	12299

同样，用户可以设置全部的单元格，行和列的值。

7. 8、使用脚本语言管理组件

如果不能够设置全部可是资源，我们还可以使用脚本语言进行设置表格的显示模式。

Cross-tab 有以下事件属性：

事件	描述
OnAfterPrint	表格打印之前调用
OnBeforePrint	表格打印之后调用
OnCalcHeight	计算某一行的的高度时调用，在事件中，可以返回需要的行的高度，如果要隐藏行，设置高度为 0。
OnCalcWidth	计算某一列的宽度时调用，在事件中，可以返回需要的列的宽度，如果要隐藏列，设置宽度为 0。

OnPrintCell	打印单元格时使用。在事件中可以改变单元格内容的显示。
OnPrintColumnHeader	打印列表头时调用，在事件中可以改变列表头内容的显示。
OnPrintRowHeader	在打印行表头时调用，在事件中可以改变行头的内容显示。

在事件中我们可以使用“cross-table”的如下方法：

方法	描述
function ColCount: Integer	返回表格的列数
function RowCount: Integer	返回表格的行数
function IsGrandTotalColumn(Index: Integer): Boolean	如果是专用的汇总列，返回 true
function IsGrandTotalRow(Index: Integer): Boolean	如果指定的行为专用回总行，返回 true
function IsTotalColumn(Index: Integer): Boolean	如果是专用列，返回 True
function IsTotalRow(Index: Integer): Boolean	如果指定的行是专用行，返回 true
procedure AddValue(const Rows, Columns, Cells: array of Variant)	添加数据到表格中

让我们演示怎样高亮显示第三列（1999 年的 11 月）。在报表设计器中选中 **cross-table** 组件，在对象查看器的事件页中，双击“OnPrintCell”属性右边的空白处，在脚本代码页中显示一个空白的 OnprintCell 处理函数。然后在 begin 和 end 中间添加代码：

Pascal script:

```

procedure Cross1OnPrintCell(Memo: TfrxMemoView;
   RowIndex, ColumnIndex, CellIndex: Integer;
    RowValues, ColumnValues, Value: Variant);
begin
    if ColumnIndex = 2 then
        Memo.Color := clRed;
end;

```

C++ Script:

```
void Cross1OnPrintCell(
    TfrxMemoView Memo,
    int RowIndex,
    int ColumnIndex,
    int CellIndex,
    Variant RowValues,
    Variant ColumnValues,
    Variant Value)
{
    if (ColumnIndex == 2) { Memo.Color = clRed; }
}
```

当打印预览时我们会看到如下结果:

	1999				
	2	10	11	12	Total
Ann	1000		1100	1200	3300
Ben		2100	2200		4300
Catherine		3000	3100		6100
Den					0
Grand Total	1000	5100	6400	1200	13700

高亮显示列标题，创建一个 OnPrintColumnHeader 事件函数，添加代码:

Pascal script:

```
procedure Cross1OnPrintColumnHeader(Memo: TfrxMemoView;
    HeaderIndexes, HeaderValues, Value: Variant);
begin
    if (VarToStr(HeaderValues[0]) = '1999') and
        (VarToStr(HeaderValues[1]) = '11') then
        Memo.Color := clRed;
end;
```

C++ Script:

```
void Cross1OnPrintColumnHeader(
```



```
TfrxMemoView Memo,  
  
Variant HeaderIndexes,  
  
Variant HeaderValues,  
  
Variant Value)  
  
{  
  
    if ((VarToStr(HeaderValues[0]) == "1999") &&  
  
        (VarToStr(HeaderValues[1]) == "11"))  
  
    {  
  
        Memo.Color = clRed;  
  
    }  
  
}
```

结果显示如下：

	1999				
	2	10	11	12	Total
Ann	1000		1100	1200	3300
Ben		2100	2200		4300
Catherine		3000	3100		6100
Den					0
Grand Total	1000	5100	6400	1200	13700

脚本是如何工作的？在表格的主体打印之前调用 “OnPrintCell” 事件。（当打印标题时，“OnprintColumnHeader”，“OnPrintRowHeader” 被调用）。同时和单元格相关联的 “Text” 组件（即 “Memo” 参数），和行号，列号，单元格序列号（RowIndex,ColumnIndex,CellIndex 参数），和 RowValues 和 ColumnValues 还有 value 参数被传到事件中。

在参数中可以使用 ColumnValues 和 RowValues 参数，代码如下：

Pascal script:

```
procedure Cross1OnPrintCell(Memo: TfrxMemoView;  
  
    RowIndex, ColumnIndex, CellIndex: Integer;  
  
    RowValues, ColumnValues, Value: Variant);  
  
begin  
  
    if (VarToStr(ColumnValues[0]) = '1999') and
```

```
(VarToStr(ColumnValues[1]) = '11') then

    Memo.Color := clRed;

end;
```

C++ Script:

```
void Cross1OnPrintCell(

    TfrxMemoView Memo,

    int RowIndex,

    int ColumnIndex,

    int CellIndex,

    Variant RowValues,

    Variant ColumnValues,

    Variant Value)

{

    if ((VarToStr(ColumnValues[0]) == "1999") &&

        (VarToStr(ColumnValues[1]) == "11"))

    {

        Memo.Color = clRed;

    }

}
```

当打印单元格列标题是调用”OnPrintColumnHeader”事件函数。和 OnprintCell 一样，有一些类似的参数。事例显示如下界面：

	0					1				2			
	0	1	2	3	4	0	1	2	3	0	1	2	3
0	1000		1100	1200	3300	1300	1400		2700		1500	1600	3100
1		2100	2200		4300		2400		2400				0
2		3000	3100		6100			3200	3200				0
3					0	3999			3999	4000	4100		8100
4	1000	5100	6400	1200	13700	5299	3800	3200	12299	4000	5600	1600	11200

生成 OnprintColumnHeader 函数，添加脚本代码：

Pascal script:

```
procedure Cross1OnPrintColumnHeader(Memo: TfrxMemoView;  
    HeaderIndexes, HeaderValues, Value: Variant);  
  
begin  
    if (HeaderIndexes[0] = 0) and (HeaderIndexes[1] = 2) then  
        Memo.Color := clRed;  
  
end;
```

C++ Script:

```
void Cross1OnPrintColumnHeader(  
    TfrxMemoView Memo,  
    Variant HeaderIndexes,  
    Variant HeaderValues,  
    Variant Value)  
{  
    if ((HeaderIndexes[0] == 0) && (HeaderIndexes[1] == 2)) { Memo.Color =  
        clRed; }  
}
```

7. 9、调整行和列的大小

用户可用通过“OnCalcWidth”“OnCalcHeight”这两个事件调整列的宽的和行的高的。我们演示如何增加列的宽的。在事例中，创建 OnCalcWidth 事件：

Pascal script:

```
procedure Cross1OnCalcWidth(ColumnIndex: Integer;  
    ColumnValues: Variant; var Width: Extended);  
  
begin  
    if (VarToStr(ColumnValues[0]) = '1999') and  
        (VarToStr(ColumnValues[1]) = '11') then  
        Width := 100;  
  
end;
```

C++ Script:

```
void Cross1OnCalcWidth(  

```

```
int ColumnIndex,

variant ColumnValues,

Extended &Width)

{

    if ((VarToStr(ColumnValues[0]) == "1999") &&

        (VarToStr(ColumnValues[1]) = "11"))

    {

        Width = 100;

    }

}
```

显示结果:

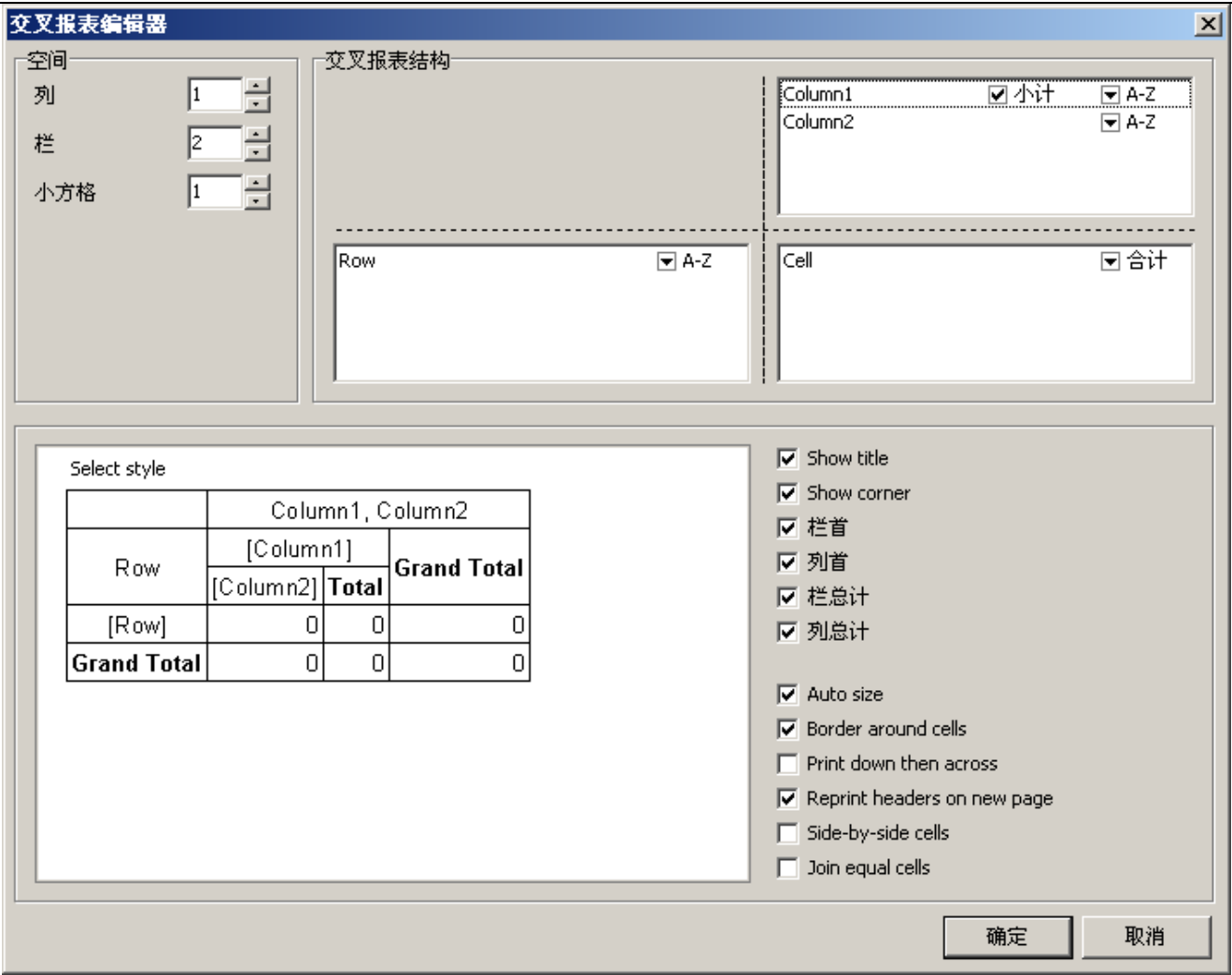
	1999				
	2	10	11	12	Total
Ann	1000		1100	1200	3300
Ben		2100	2200		4300
Catherine		3000	3100		6100
Den					0
Grand Total	1000	5100	6400	1200	13700

如果要隐藏列，可以返回 width=0，结果值矩阵改变之前，汇总函数不进行从新计算。

7. 10、手动填充表格单元

有两个 Cross-table 的版本：“DB Cross table”、“Cross-table”。大部分情况，我们在生成报表时，使用 DB Cross-table 组件进行连结数据表，自动进行填充表格的方法处理报表。现在使用 “cross-table” 组件。

这个组件不能数据表中的数据源进行关联。因此必须手动填充数据。组件编辑器也比较简单。



让我们通过事例演示 cross-table 的应用，在报表设计页面上添加一个 cross-table 组件，打开组件编辑器窗口，设置列数为 2，行数为 1，单元数为 1。创建 OnBeforePrint 事件填充表格。

PascalScript:

```
procedure Cross1OnBeforePrint(Sender: TfrxComponent);  
  
begin  
    with Cross1 do  
    begin  
        AddValue(['Ann'], [2001, 2], [1500]);  
        AddValue(['Ann'], [2001, 3], [1600]);  
        AddValue(['Ann'], [2002, 1], [1700]);  
        AddValue(['Ben'], [2002, 1], [2000]);  
        AddValue(['Den'], [2001, 1], [4000]);
```

```
        AddValue(['Den'], [2001, 2], [4100]);

    end;

end;
```

C++ Script:

```
void Cross1OnBeforePrint(TfrxComponent Sender)
{
    Cross1.AddValue(["Ann"], [2001, 2], [1500]);
    Cross1.AddValue(["Ann"], [2001, 3], [1600]);
    Cross1.AddValue(["Ann"], [2002, 1], [1700]);
    Cross1.AddValue(["Ben"], [2002, 1], [2000]);
    Cross1.AddValue(["Den"], [2001, 1], [4000]);
    Cross1.AddValue(["Den"], [2001, 2], [4100]);
}
```

在事件中我们使用 TfrxCrosstab.Addvalue 方法，添加数据到单元格中。预览：

	2001				2002		Grand Total
	1	2	3	Total	1	Total	
Ann		1500	1600	3100	1700	1700	4800
Ben				0	2000	2000	2000
Den	4000	4100		8100		0	8100
Grand Total	4000	5600	1600	11200	3700	3700	14900

在 DB Cross-table 组件中一样可以应用 AddValue 方法添加数据，此时不再和数据源关联。

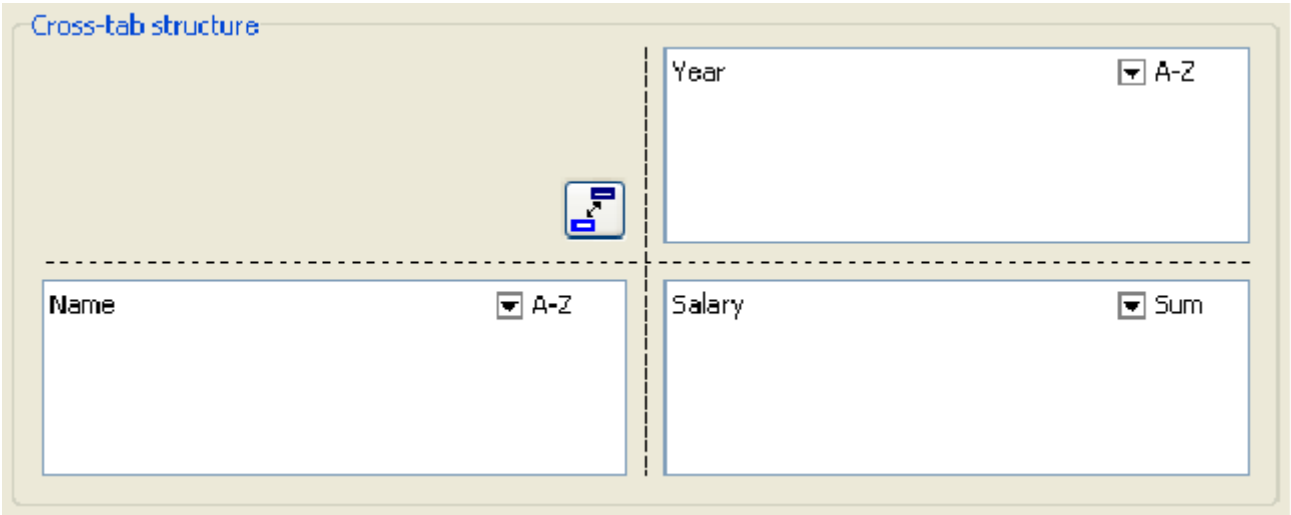
7. 11、在表格单元中加入其他组件

用户可以放置其他组件到 cross-table 组件上，干吗？有时我们需要使用图形方法显示结果值。现在演示使用图形显示进度条。


Salary	Year				
Name	1999	2000	2001	2002	Grand Total
Ann	3300	2700	3100	1700	10800
Ben	3900	2100		1800	7800
Catherine	6100	3200			9300
Den		3999	8100		12099
Grand Total	13300	11999	11200	3500	39999

小于 1000 的显示一个红色方块，小于 2000 的显示两个黄色方块，大于 3000 的显示三个绿色方块。

现在开始，在报表上放置一个 DB Cross-table 组件，并设置属性：

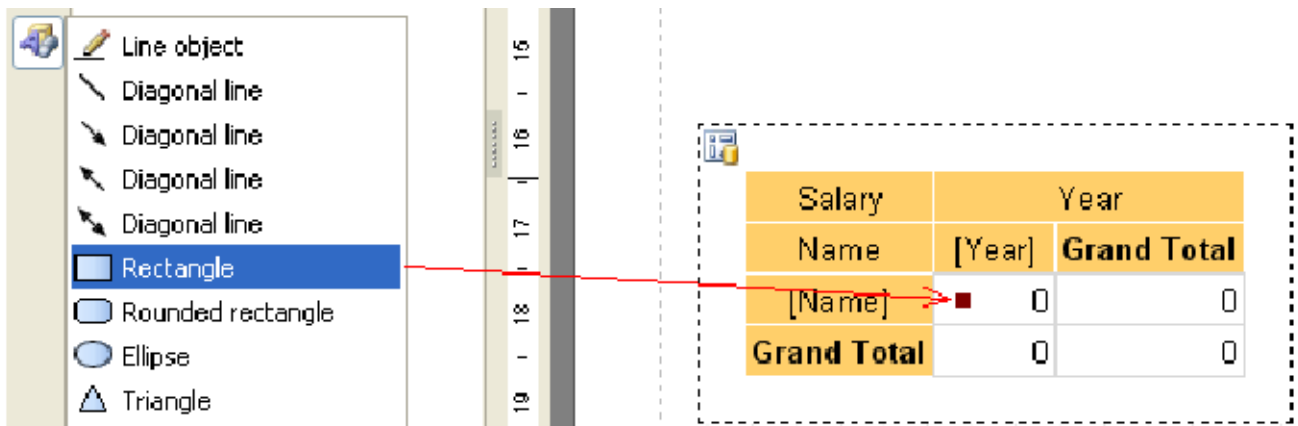


关闭自动改变大小的属性，并设置列的宽度。



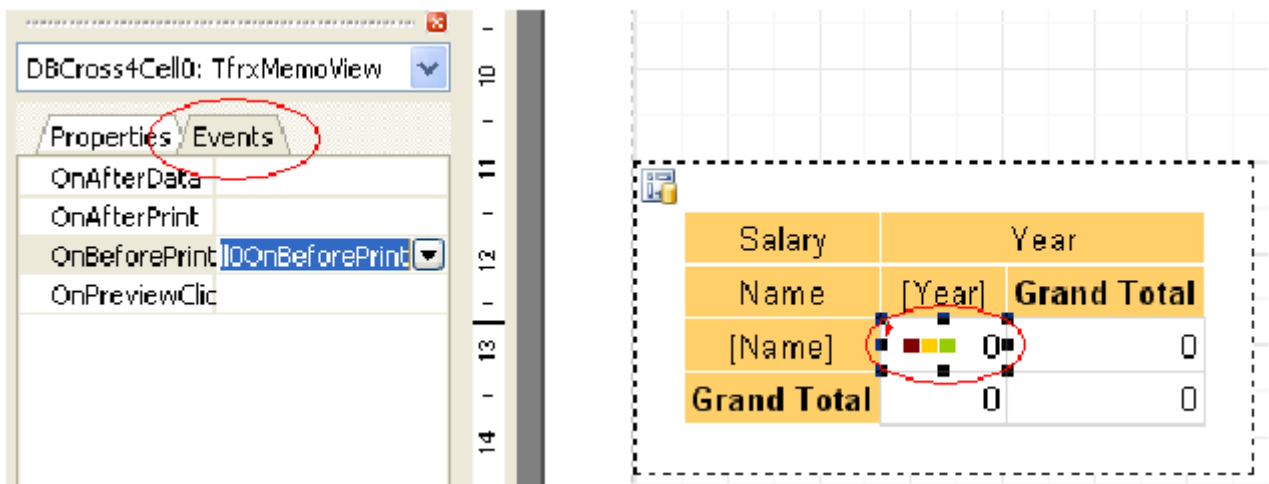
Salary	Year	
Name	[Year]	Grand Total
[Name]	0	0
Grand Total	0	0

现在加入图形到表格中，在组件面板中选择 shape，添加到 cross-table 组件中。



同样的方法添加另外两个图形。

创建脚本，根据结果数据的不同而显示不同的图形。选中单元格，并通过对象查看器创建一个 OnBeforePrint 事件函数：



在事件行数种写入如下代码：

```
procedure DBCross1Cell10OnBeforePrint(Sender: TfrxComponent);
begin
// Value it's a current cell's value
    if Value < 100 then
    begin
        // first shape object
        DBCross1Object1.Color := clMaroon; // red
        // second shape object
        DBCross1Object2.Color := clWhite;
        // third shape object
        DBCross1Object3.Color := clWhite;
```



```

end

else if Value < 3000 then

begin

    DBCross1Object1.Color := $00CCFF; // yellow

    DBCross1Object2.Color := $00CCFF;

    DBCross1Object3.Color := clWhite;

end

else

begin

    DBCross1Object1.Color := $00CC98; // green

    DBCross1Object2.Color := $00CC98;

    DBCross1Object3.Color := $00CC98;

end;

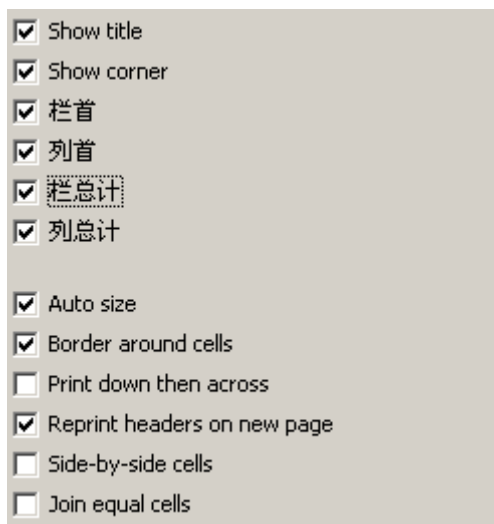
end;

```

然后运行报表。

7. 12、一些有用的设置

我们看看 Cross-table 的编辑器中可用的设置。



前六个选项是设置表格中显示或隐藏的表格元素。

Auto size 允许系统自动调整表格的高和宽。

Border around Cells: 确定是否绘制单元格的边筐，如果取消，显示界面类似：

Salary	Year				
Name	1999	2000	2001	2002	Grand Total
Ann	3300	2700	3100	1700	10800
Ben	3900	2100		1800	7800
Catherine	6100	3200			9300
Den		3999	8100		12099
Grand Total	13300	11999	11200	3500	39999

“Print down then across” 选项决定多页显示时的顺序:

1: 当选中对话框时, 显示顺序如下:

1	Salary		2	Year			
	Employee	1999	2000	2001	2002	Grand Total	
	Ann	3 300,00p.	2 700,00p.	3 100,00p.	1 700,00p.	10 800,00p.	
	Ben	3 900,00p.	2 100,00p.		1 800,00p.	7 800,00p.	
	Catherine	6 100,00p.	3 200,00p.			9 300,00p.	
	Den		3 999,00p.	3 100,00p.		12 099,00p.	
3	Grand Total	13 300,00p.	11 999,00p.	4	6 200,00p.	3 500,00p.	39 999,00p.

2: 当不选中对话框时, 显示顺序如下:

1	Salary						3	Year		
	Employee	1999	2000	2001	2002	Grand Total				
	Ann	3 300,00p.	2 700,00p.	3 100,00p.	1 700,00p.	10 800,00p.				
	Ben	3 900,00p.	2 100,00p.		1 800,00p.	7 800,00p.				
	Catherine	6 100,00p.	3 200,00p.			9 300,00p.				
	Den		3 999,00p.	3 100,00p.		12 099,00p.				
2	Grand Total	13 300,00p.	11 999,00p.	3 200,00p.	3 500,00p.	39 999,00p.	4			

“Reprint headers on new page” 决定在多个页面打印时，是否在每个页面当打印输出标题栏。

“Side by side cell”：是否允许合并单元格


“Join equal cells” 选项，当数值相等时是否可以合并单元格。

Days	Year				
Name	1999	2000	2001	2002	Grand Total
Ann	3		4	2	12
Ben	4	2		2	8
Catherine	6	3			9
Den		4	7		11
Grand Total	13	12	11	4	40

使用对象查看器的属性设置也可以设置这些内容。

第 八 章

制图表

FastReport 允许用户插入 chart 制表组件到报表中，这样，在 FastReport 面板上的“TfrxCharObject”  组件就必须应用上。这个组件基于“TeeChart”动态库，在 delphi 中需要将它包含在工程中。

我们通过以下事例先进行简单说明，连接 DBDemos 中的 countory 表，数据内容如下：

<u>Name</u>	<u>Area</u>	<u>Population</u>
Argentina	2 777 815	32 300 003
Bolivia	1 098 575	7 300 000
....		

在 delphi 环境中创建一个新的工程，添加“TTable”，“TfrxChartObject”，“TfrxDBData”，“TfrxReport”组件，并设置属性值：

Table1:

DatabaseName = 'DBDEMOS'

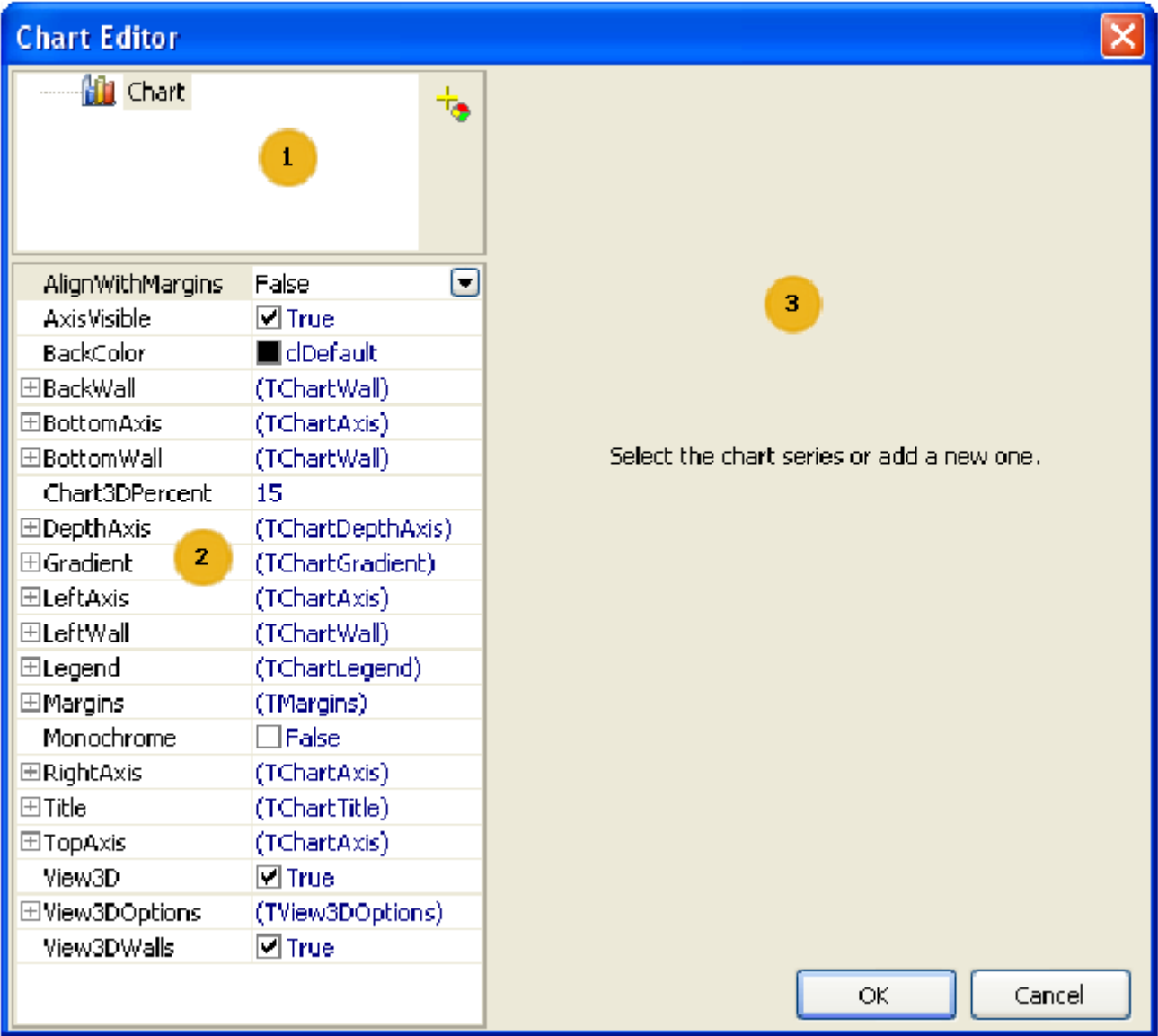
TableName = 'country.db'

frxDBDataSet1:

DataSet = Table1


UserName = 'Country'

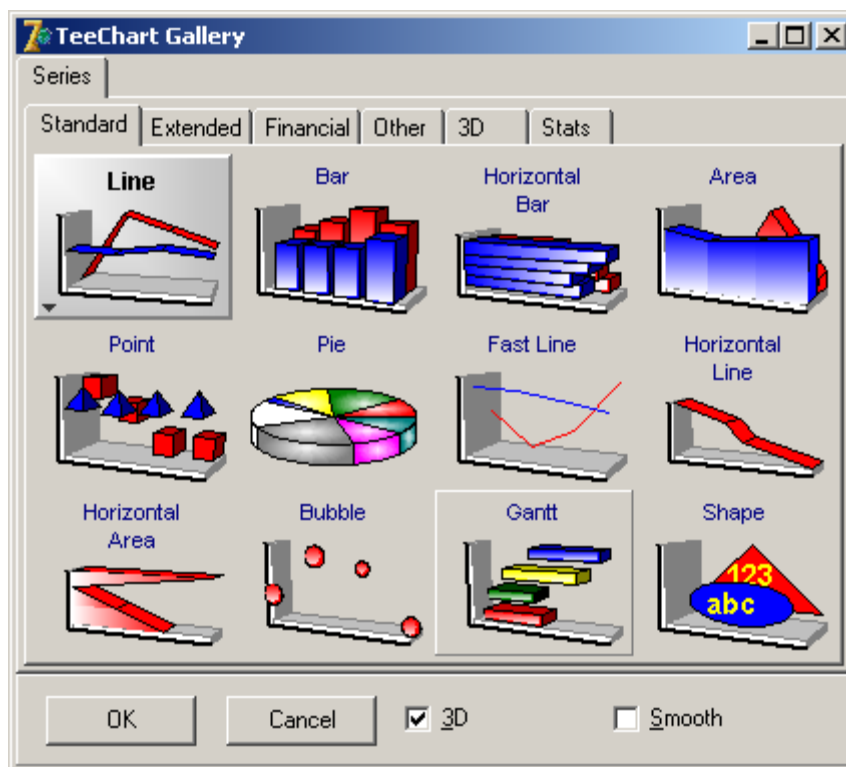
进入到报表设计器页面，并连结设置数据源。在报表的设计页上放置一个“chart”组件，并调整其大小（18X8cm）。双击组件，打开组件的编辑器窗口。



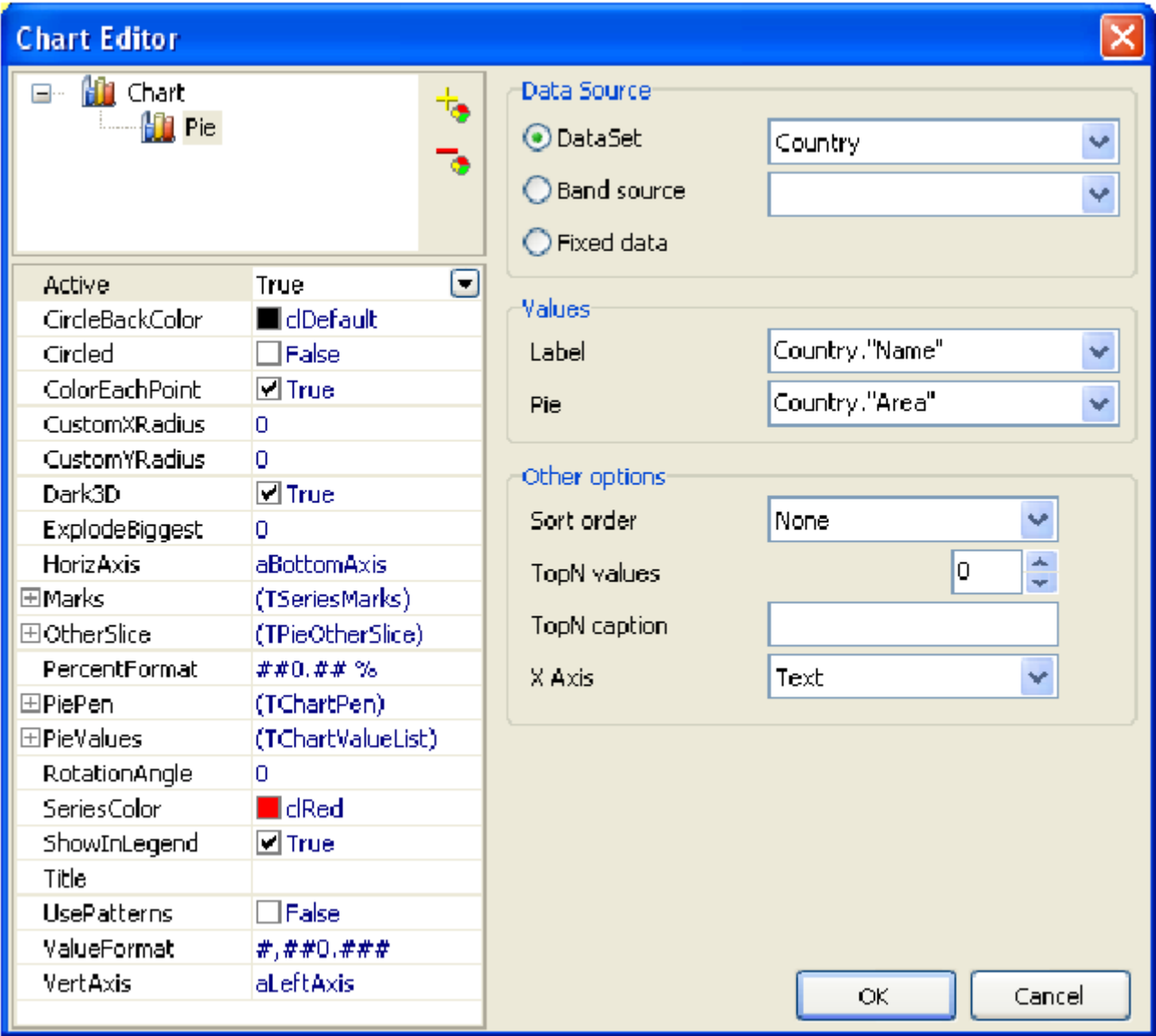
char 编辑器区域说明：

- 1：chart 结构，一个 chart 包含有几个制表系统。
- 2：组件监视器，显示所选组件的属性。
- 3：连结 chart 的 series 的数据属性。

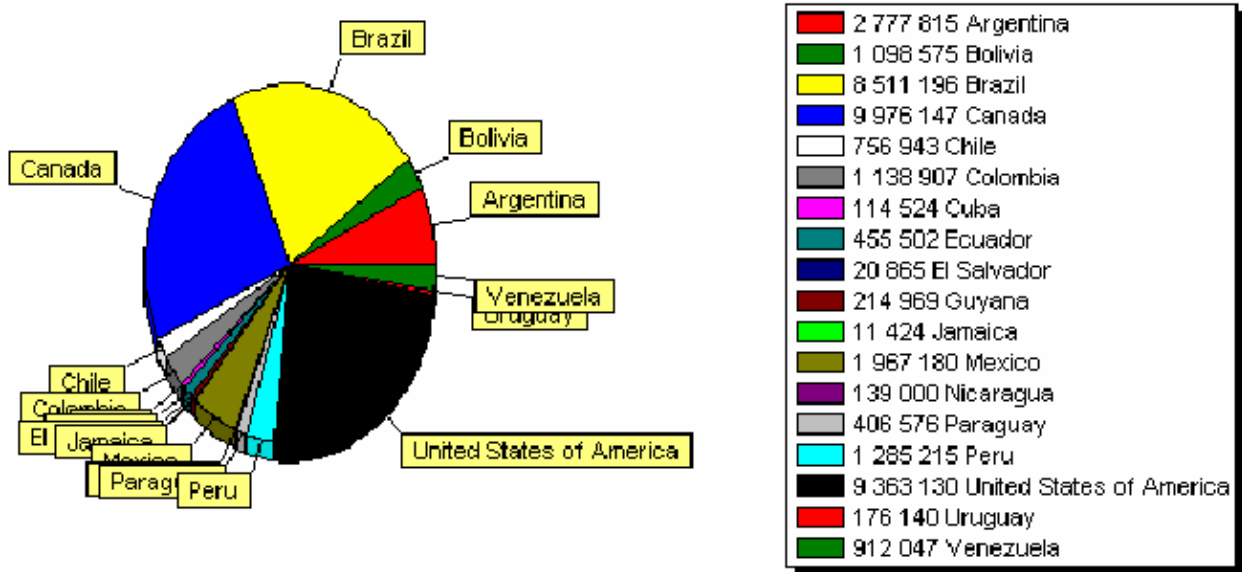
开始的界面显示如上，首先添加一个或几个series,在1区点击按钮，添加一个Pie。



有许多可用的不同的类型的 series。添加一个系列之后，3 区被激活。首先选择数据源，设置 Label 和 pie 字段。



点击确定按钮，关闭编辑器窗口，然后预览。



还如何改良以下报表呢？首先，将数据按升序方式进行排序。再次进入编辑器，选择要设置的 series，在 3 区设置选项。

其他选项

排列顺序

无

前N名的值

0

前N名的标题

X Axis

Text

8. 1、chart数据中数字的限制

上面的制表看上去非常拥挤，在 chart 中有特别多的值，而有些看不见。FastReport 可以设置显示数据的限制性。因此不被限制的值可以在制表中显示。

在我们的事例中，包含 18 个数值，而最多友 8 个能显示，键入编辑器，进行设置。

其他选项

排列顺序

无

前N名的值

8

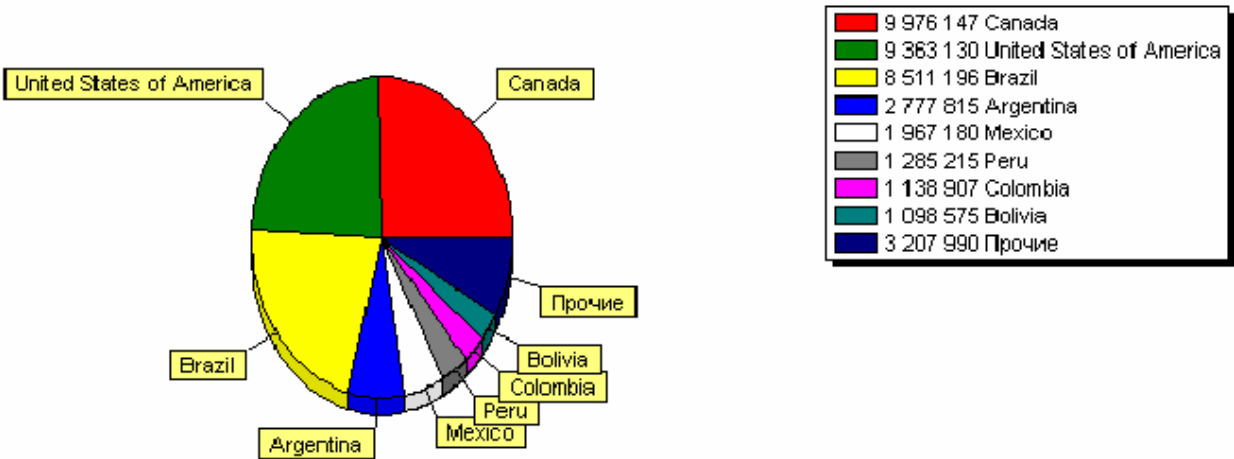
前N名的标题

other|

X Axis

Text

前几个如果不等于 0。则显示在报表中，如果排序没有设置，则采用默认排序。
结果如下：



8. 2、设置

下面说明一下几个有用的设置。这些属性只能在对象监视器中进行设置。

以下几个属性可用：

- Gradient：设置背景填充颜色。
- Legend：图例

- 一下是 series 的属性：
- ColorEachPoint: 设置每个部分一种颜色
- ExplodeBiggest: 设置最大部分分出。
- ValueFormat:数字监视格式。

8. 3、指定数字制表

在上面的制表中，我们使用的是数据表中数据进行创建的图表。还可以通过手动输入进行制表。这对创建一个小的制表系统是非常方便的。

让我们示范一下如何工作？在报表设计器中添加一个制表系统，进入编辑器，添加一个“bar chart”类型 series，并设置属性：

Data Source

☐ 数据库

☐ Band source

☒ 固定数据

Values

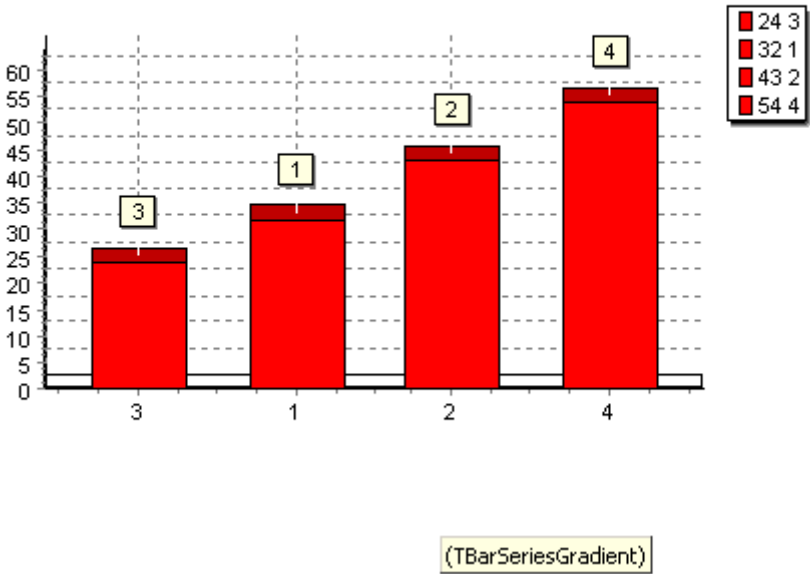
Label

1;2;3;4

Y

32;43;24;54

X (optional)



8. 4、利用脚本进行制表

将上面的 xvalue 和 yvalue 属性值取消掉，并在报表中添加如下代码：

PascalScript:

```
begin  
  
    Chart1.SeriesData[0].XSource := 'Jan;Feb;Mar;Apr';  
  
    Chart1.SeriesData[0].YSource := '31;28;31;30';  
  
end.
```

C++Script:

```
{  
  
    Chart1.SeriesData[0].XSource = "Jan;Feb;Mar;Apr";  
  
    Chart1.  
  
}
```

SeriesData[0] 允许用户设置第一个 series 参数，如果制表中有多个 series，可以通过 SeriesData[Data_Number] 进行设置。

8. 5、在delphi环境中创建的报表的打印

如果用 delphi 的代码创建一个报表，并想着打印报表，这需要一个“Picture”组件。在报表设计器的相应位置添加一个“Picture”组件。在 delphi 环境中，在 frxReport.OnBeforePrint 事件中添加如下代码：

```
procedure TForm1.frxReport1BeforePrint(Sender: TfrxReportComponent);  
  
begin  
  
    if Sender.Name = 'Picture1' then  
  
        TfrxPictureView(Sender).Picture.Assign(  
  
            Chart1.TeeCreateMetafile(False,  
  
end;
```

Picture1 是报表中 Tpicture 组件，chart1 是 delphi 中的 Tchart 组件名称。

第 九 章

点阵报表

以前的报表在普通的打印机中打印，如果发送到点阵打印机中，则打印速度非常的慢。**FastReport** 允许用户生成针对点阵打印机的报表模式，他只有标准的元素，而没有图形等其他特殊元素。这就是为什么会打印的速度要快。

我们创建一个 **List** 报表，创建点阵报表，以前的报表参考客户列表报表。我们应用相同的数据源。

在 **delphi** 环境中添加一个 **Ttable**，**TfrxDBData**，**TfrxReport**，**TfrxDotMatrixExport** 组件。并设置属性：

TTable:

DatabaseName = 'DBDEMOS'

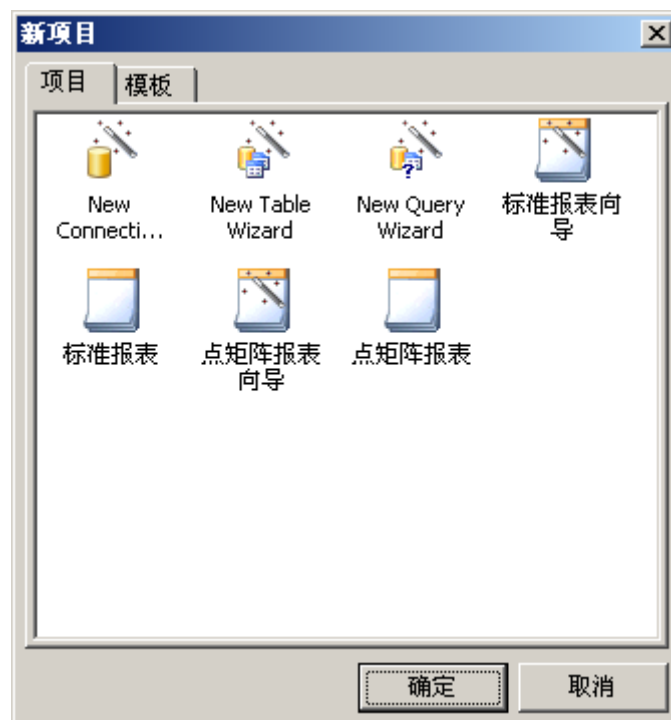
TableName = 'Customer.db'

TfrxDBDataSet:

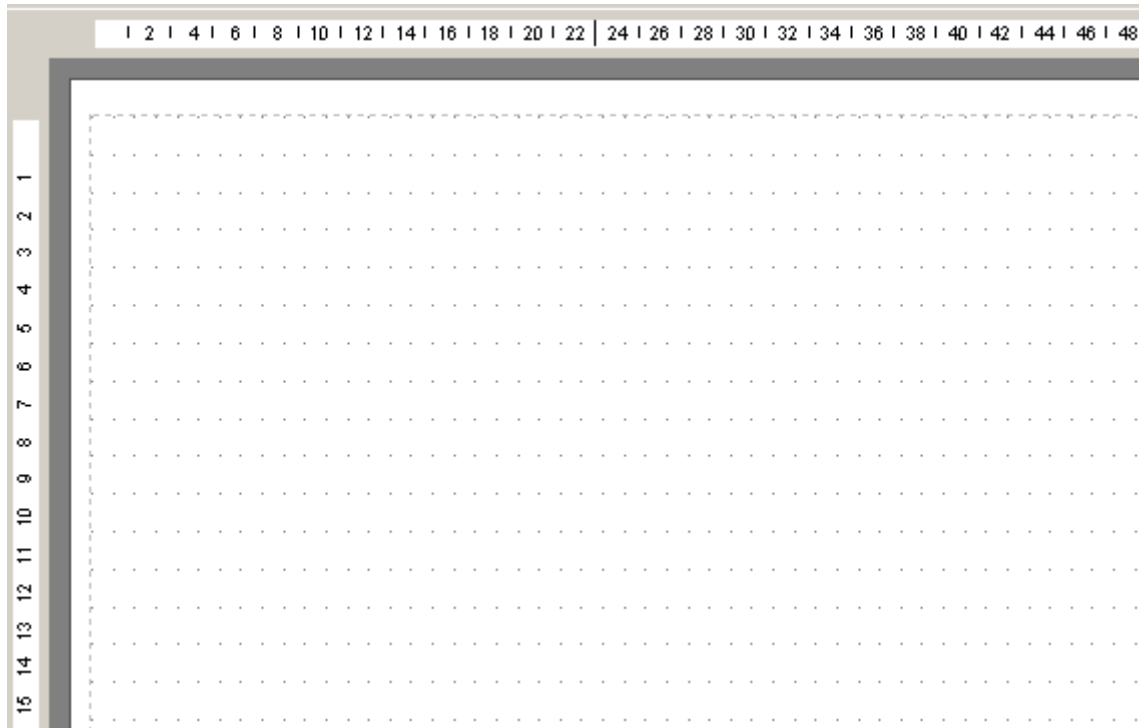
DataSet = Table1

UserName = 'Customers'

进入报表设计器，在“文件|新建...”菜单，打开报表向导窗口，选择“Dot-Matrix report”项。



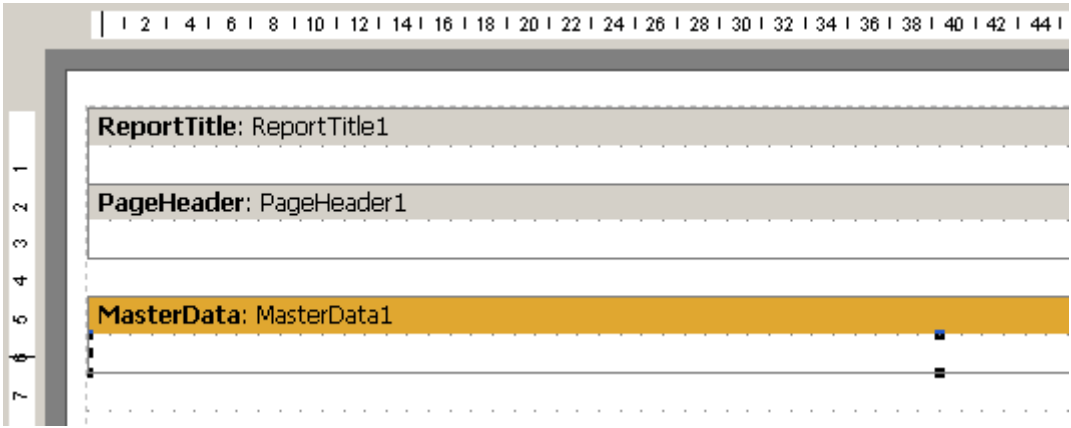
点击确定按钮，生成一个空白的报表设计界面。



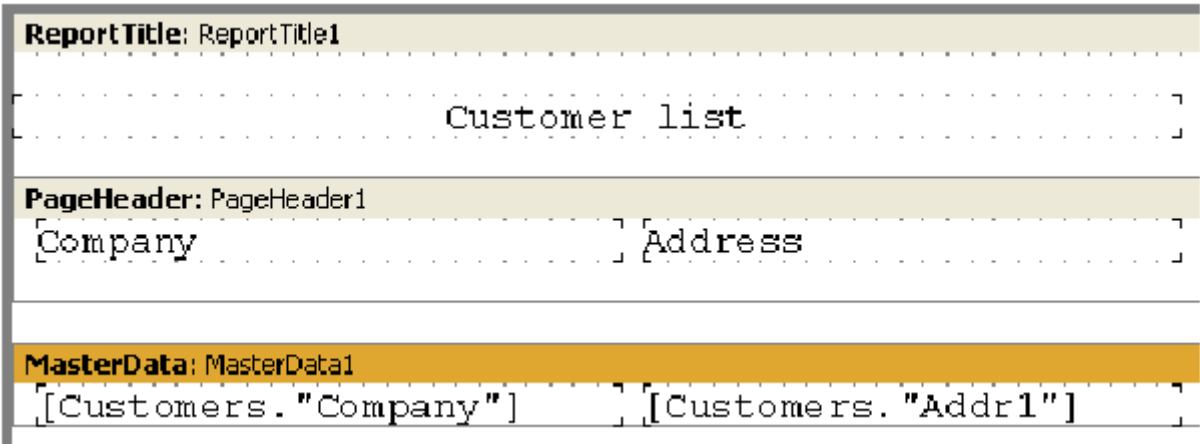
可用的组件列表发生变化，只有 band、text、line、esc-command、subreport、cross-tab 组件。
其他组件在点阵报表中不可用。



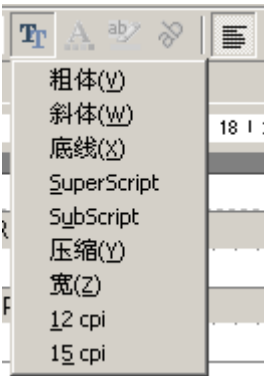
放置 Report title、Page header 、master data band 到报表中。



在 Band 上添加 “text” 组件，如下：



点阵报表中组件的放置原则和普通报表一样，不同之处，就值放置的位置严格的对应网格，不能设置字体的大小和颜色。选择 Text 组件，在工具栏中点击 “Tt” 按钮。



如你所见，能够修改字体的属性，打印预览报表。

Customer list	
Company	Address
Action Club	PO Box 5451-F
Action Diver Supply	Blue Spar Box #3
Adventure Undersea	PO Box 744
American SCUBA Supply	1739 Atlantic Avenue
Aquatic Drama	921 Everglades Way

9. 1、点阵报表使用交叉报表

点阵报表中可用的几个组件是可以放在文本报表上使用的。其中就有 cross-tab 组件。我们示范 cross-tab 组件和以前的例子相类似。

点阵报表的创建步骤和以前的报表“Empty dot-matrix wizard”相类似。放置一个 DB cross-tab 组件，并进入编辑器：

Cross-tab Editor

Source data

Cross

Days

Salary

Cross-tab structure

Year

Month

Subtotal

A-Z

A-Z

Name

A-Z

Salary

Sum

Salary	Year, Month		
Employee	[Year]		Grand Total
	[Month]	Total	
[Name]	0	0	0
Grand Total	0	0	0

☒ Show title

☒ Show corner

☒ Column header

☒ Row header

☒ Column grand total

☒ Row grand total

☒ Auto size

☒ Border around cells

☐ Print down then across

☒ Reprint headers on new page

☐ Side-by-side cells

☐ Join equal cells

OK

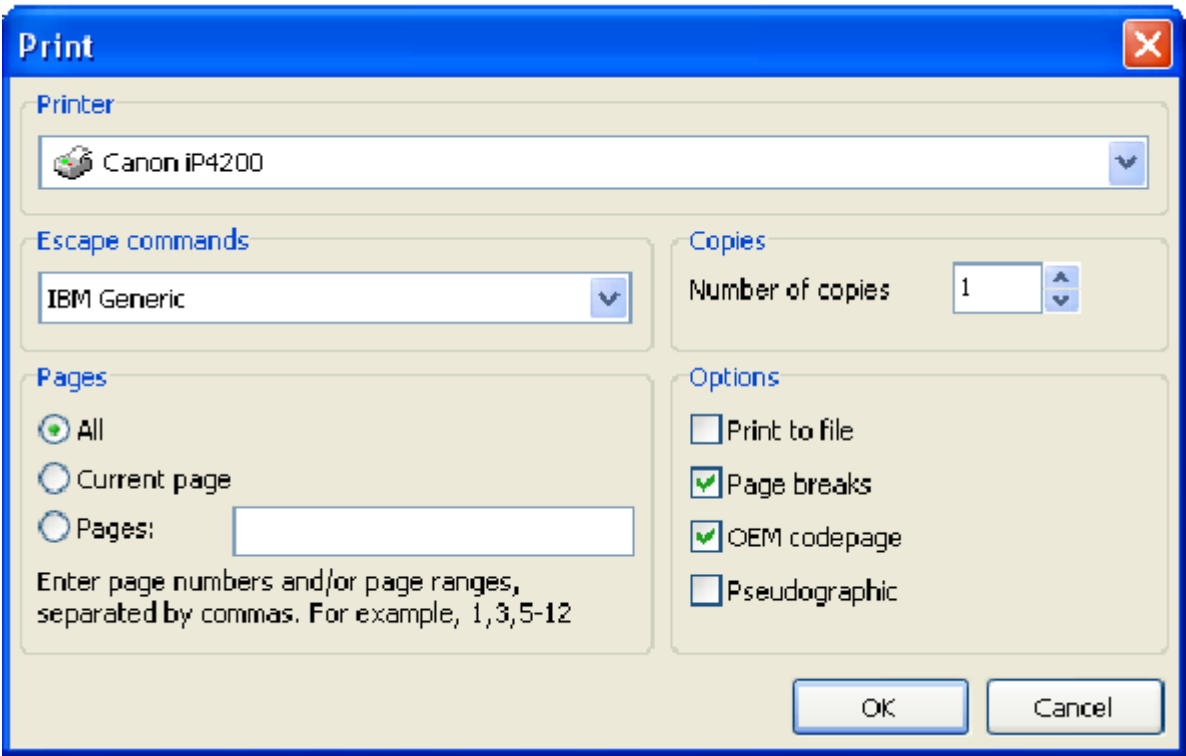
Cancel

报表预览：

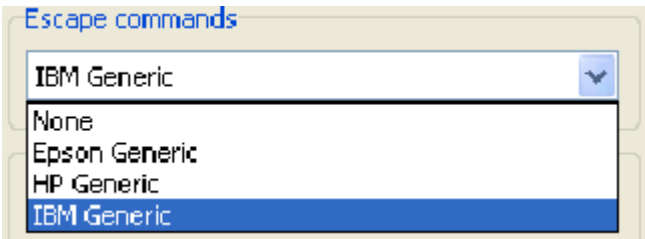
	1999					2000				2001		
	2	10	11	12	Total	1	2	3	Total	1	2	3
Ann	1000		1100	1200	3300	1300	1400		2700		1500	1600
Ben		2100	2200		4300		2400		2400			
Catherine		3000	3100		6100			3200	3200			
Den					0	3999			3999	4000	4100	
Grand Total	1000	5100	6400	1200	13700	5299	3800	3200	12299	4000	5600	1600

9.2、点阵报表的打印

以文本模式打印点阵报表，需要组件 TfrxDotMatrixExport 。这个组件将报表转换为文本模式进行打印，同时他还替代标准的打印对话框。



使用以下可用的命令：




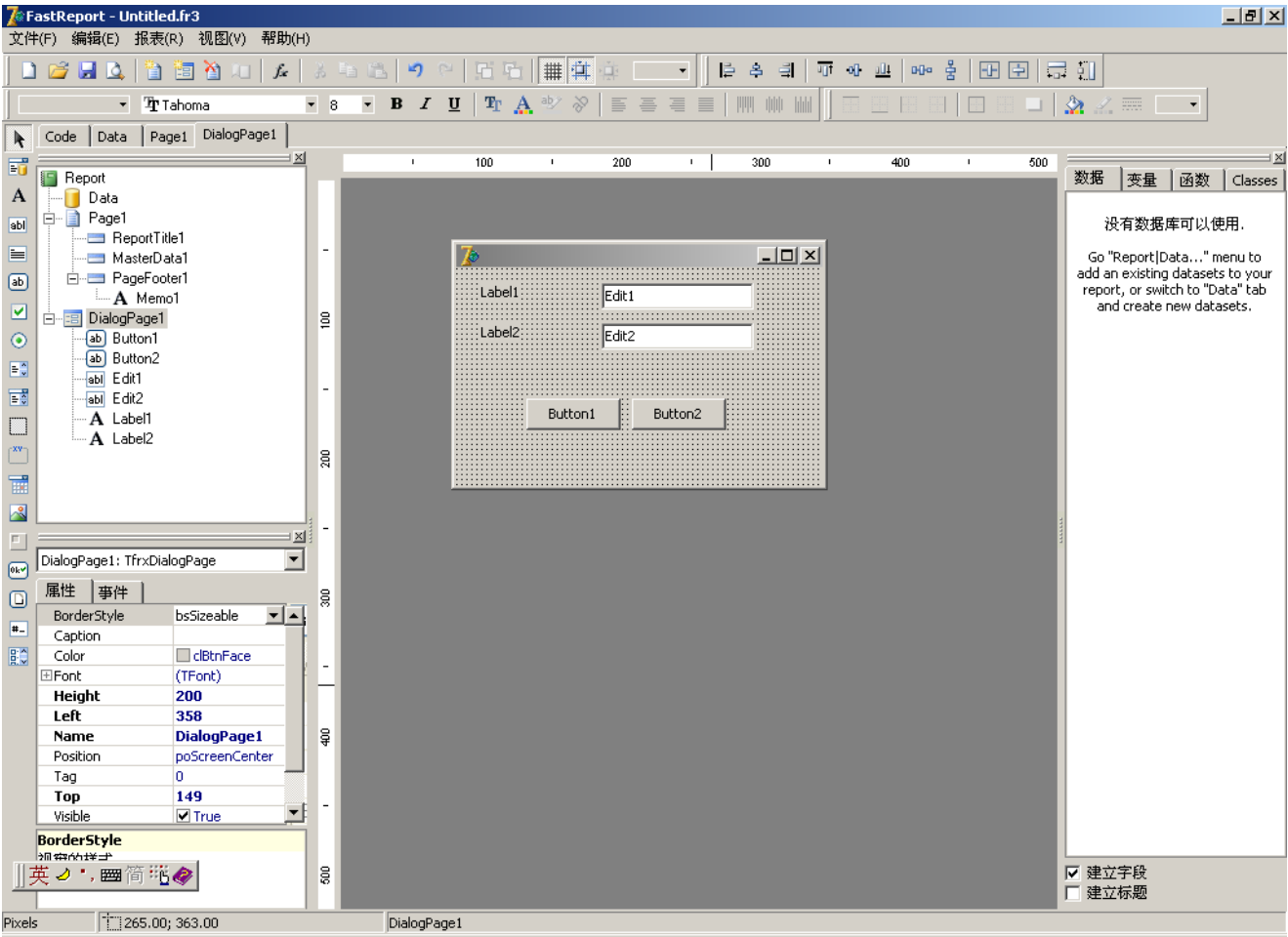
还有几个打印选项标志。

9. 3、命令组件


第 十 章

对话框窗体




除了可是设计报表页面，在报表中还可以使用对话框窗体。在报表设计器中创建对话框，使用工具览中的按钮，可以创建一个空白对话框，在上面可以添加一些组件。



10. 1、控件

为了在报表中使用这些组件，需要在 delphi 的设计窗口中添加 TfrxDialogControls  或将 frxDctrl 单元添加到 uses 列表中。以下组件在报表中即可使用。

图标	名称	描述
	TfrxLabelControl	这个组件可用在对话窗口中作为显示解释性的描述文字
	TfrxEditControl	这个组件用于键盘输入单行文本筐
	TfrxMemoControl	这个组件可是输入多行文本
	TfrxButtonControl	这个组件绘制一个按钮
	TfrxCheckBoxControl	这个组件绘制一个只有两种状态的标志，标志旁边是解释文本。
	TfrxRadioButtonControl	单选按钮

	TfrxListBoxControl	显示可以选择的多行列表
	TfrxComboBoxControl	下拉筐选择列表
	TfrxPanelControl	工具面板，上面可以放置不同的组件
	TfrxGroupBoxControl	带有描述性的工具面板。
	TfrxDateEditControl	下拉筐中可选择日期的组件
	TfrxImageControl	图形组件，可以容纳 BMP,ICO, WMF, EMF 格式图形
	TfrxBevelControl	在对话框中绘制图形
	TfrxBitBtnControl	可以支持放置图形的按钮
	TfrxSpeedButtonControl	可以支持放置图形的按钮
	TfrxMaskEditControl	带格式的文本输入窗口控件
	TfrxCheckListBoxControl	多行标志列表控件

可以发现，这些组件的使用和 delphi 环境下相应的组件相类似。

10. 2、“世界你好！”报表

在这个事例中，我们在报表输出之前，用对话框窗口显示一个欢迎窗口。在 delphi 中创建一个新的工程，放置一个 frxReport 和一个 TfrxDialogContors 组件，进入报表编辑器环境，创建一个新的窗口。在窗口上添加一个 TfrsLebalControl 和一个 TfrcButtonControl，并设置属性：

TfrxLabelControl:

Caption = ‘世界你好!’

TfrxButtonControl:

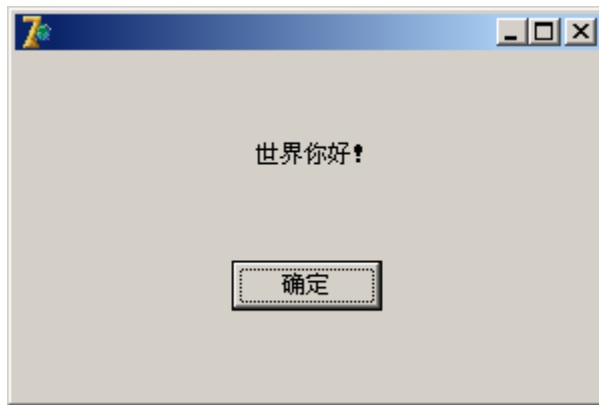
Caption = 'OK'


Default = True

ModalResult = mrOk

设置 Bodystyle=bsDialog。你会发现这些组件和窗口的属性列表和 delphi 环境的组件的属性列表基本相同。

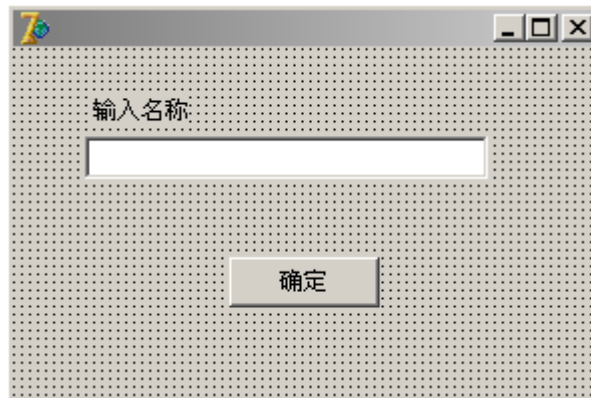
然后关闭设计环境，进行预览：



点击确定进入报表预览界面，点击系统关闭按钮，则退出报表。

10. 3、输入参数，并传递到报表中

让我们示范如何在对话框窗口中输入参数，并将其值传递到报表中输出。修改上面的事例：



在报表设计页放置一个 text 组件，并输入

你输入的名称:

[Edit1.Text]

预览报表，确认你输入的文字已经显示到报表中了。同样可以放置其他的组件到对话框中。

每个组件在整个报表中都有一个唯一的名称，可以在报表的任何部位进行使用。

10. 4、组件的交互

使用脚本可以很容易得创建一个可以交互的报表。我们改变一下事例进行示范：



双击 checkbox1 组件，创建一个 checkbox1 的 onclick 事件，并输入如下代码：

PascalScript:

```
procedure CheckBox1OnClick(Sender: TfrxComponent);  
  
begin  
    Button1.Enabled := not CheckBox1.Checked;  
  
end;
```

C++ Script:

```
void CheckBox1OnClick(TfrxComponent Sender)  
{  
    Button1.Enabled = !CheckBox1.Checked;  
}
```

10. 5、多个对话框表单

让我们示范以下两个对话框如何工作。报表设计器加入组件，并建立两个对话框。

[名称:]	[edit1.text]]
[名称1]	[edit2.text]]
[名称2]	[edit3.text]]



预览:

名称:	NameMy
名称1	First Name
名称2	Secend Name

10. 6、对话框窗体的管理

在报表中两个窗口都运行出现。怎么能够根据条件,隐藏第二个窗体呢? 创建一个 checkbox 的 Onclick 事件。填入代码:

PascalScript:

```
procedure Button1OnClick(Sender: TfrxComponent);  
  
begin  
    DialogPage2.Visible := CheckBox1.Checked;  
  
end;
```


C++Script:

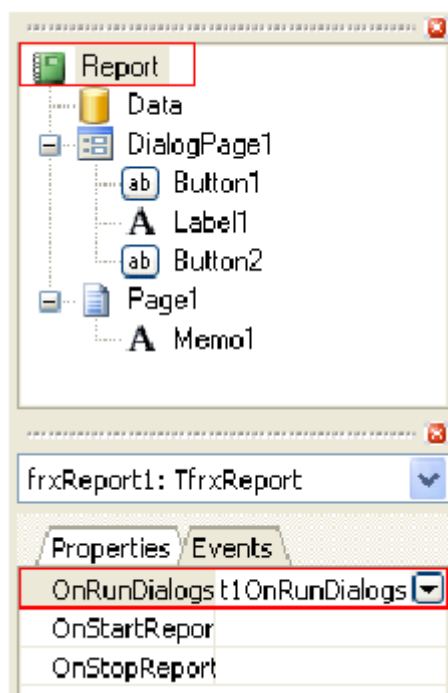
```

void Button1OnClick(TfrxComponent Sender)
{
    DialogPage2.Visible = CheckBox1.Checked;
}

```

这段代码就是隐藏第二个窗口，如果不选中，则隐藏第二个窗口。

第二种方法就是在 report 的 OnRunDialogs 事件中进行处理。在报表树中选择 report，在对象查看器的事件页中，双击 OnRunDialogs，创建事件



并加入代码:

PascalScript:

```

procedure frxReport1OnRunDialogs(var Result: Boolean);
begin
    Result := DialogPage1.ShowModal = mrOk;

    if Result then
    begin
        if CheckBox1.Checked then
            Result := DialogPage2.ShowModal = mrOk;
    end;
end;

```

```
end;
```

C++Script:

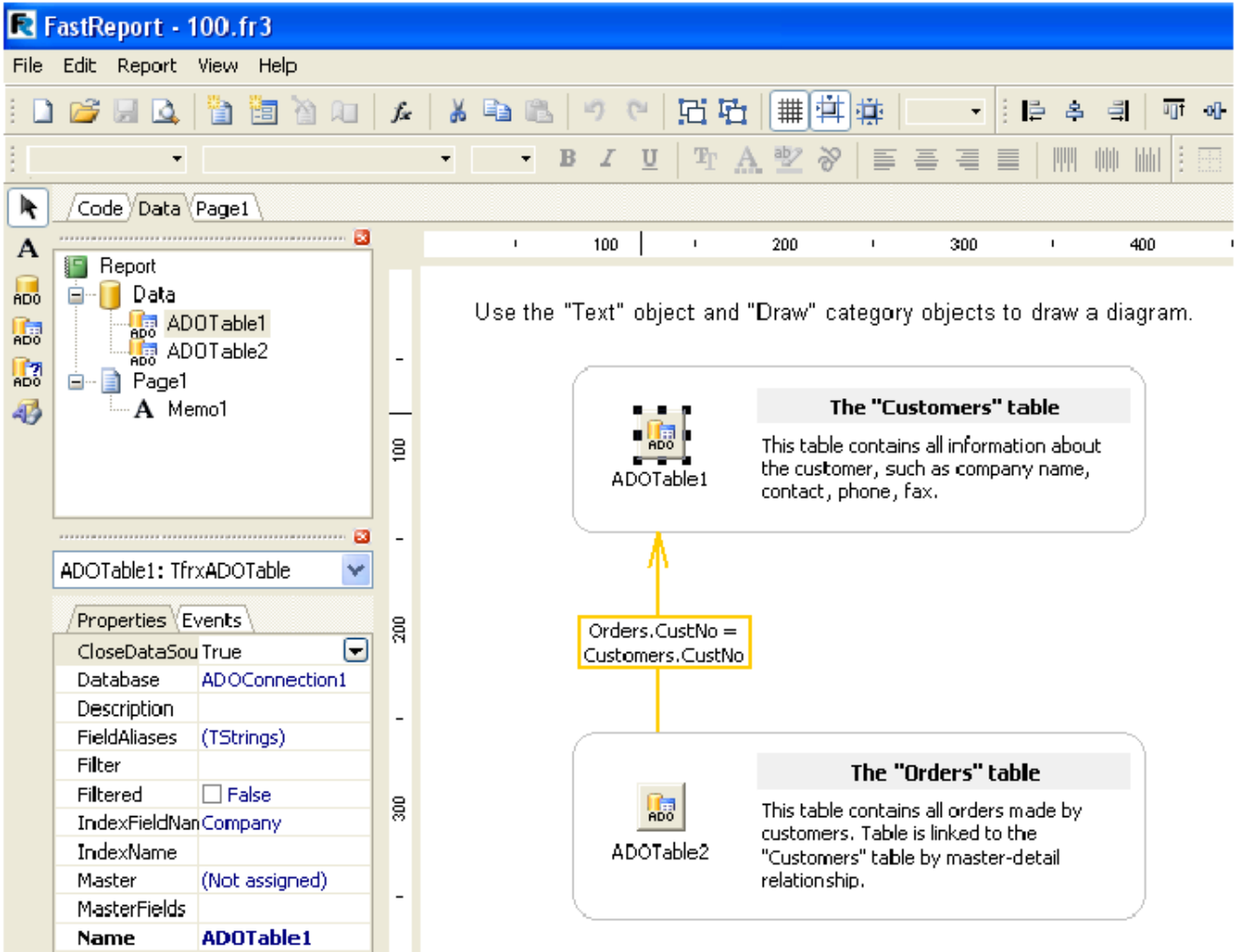
```
void frxReport1OnRunDialogs(bool &Result);  
{  
    Result = DialogPage1.ShowDialog == mrOk;  
    if (Result)  
    {  
        if (CheckBox1.Checked)  
            Result = DialogPage2.ShowDialog == mrOk;  
    }  
}
```

第十一章


数据访问组件




常规情况下，大部分报表是基于数据表中的数据进行报表。访问数据，delphi 中提供有效的机制，这些可以用在 FastReport。这涉及到“TTable”“TQurey”等数据组件，他们可以和 datasource 进行连接，他们可以为报表提供数据源。

除了用 TfrxDBData 组件在 delphi 环境设置的组件能够访问数据表外，在 report 内部通过数据访问引擎，也能够访问数据表。在 FastReport 中的访问组件和 delphi 环境的组件差不多。和 delphi 一样，将组件放在窗体上，通过对象查看器设置组件属性。组件的形态是灵活的，你可以创建一个组件，可以访问不同的数据库，加入 TfrsDesigner 组件，可以让用户在运行阶段在线进行设计。



11. 1、组件的描述

我们通过 ADO 组件来示范这些数据访问组件的应用，这需要在工程窗体上添加一个 TfrxADOComponents  组件。在报表设计器中，切换到 data 页，这时在组件面板中有“TfrxADOTable”“TfrxADOQuery”“TfrxAdoDatabase”出现，此时你就可以对话框窗体上使用 TfrxDBlookupbombox 组件了。

图标	名称	说明
	TfrxADODataBase	用户连结数据库
	TfrxADOTable	连接访问数据表
	TfrxAdoQurey	查询访问数据源
	TfrxDBlookupbombox	显示数据表中的数据。

11. 1. 1、TfrxDBLookupCombobox 组件

用户从数据表中选择数据值。

属性介绍：

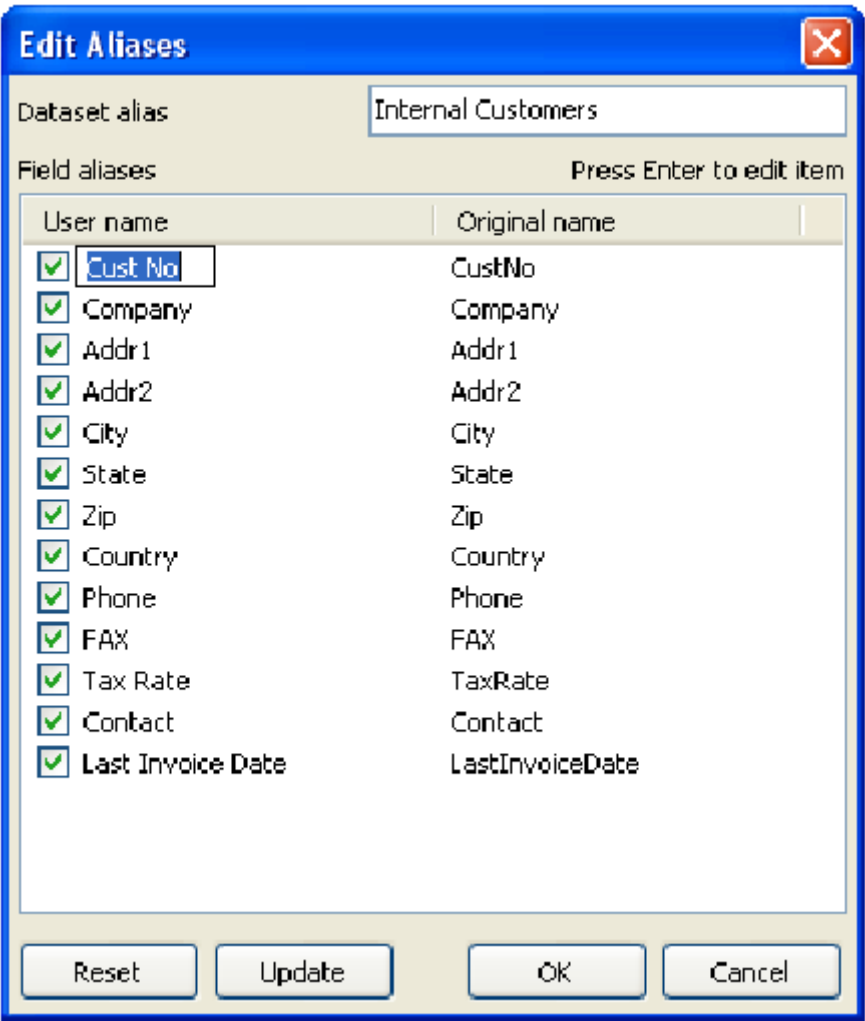
属性	描述
DataSet	选择组件可以连结的数据源
ListField	在列表中显示的数据的数据表字段
KeyField	对应 listField，返回的数据的数据表字段
Text	显示在组件上的字符串。
KeyValue	对应 Text 值的数据表中 keyField 的值。

11. 1. 2、TfrxADOTable 组件

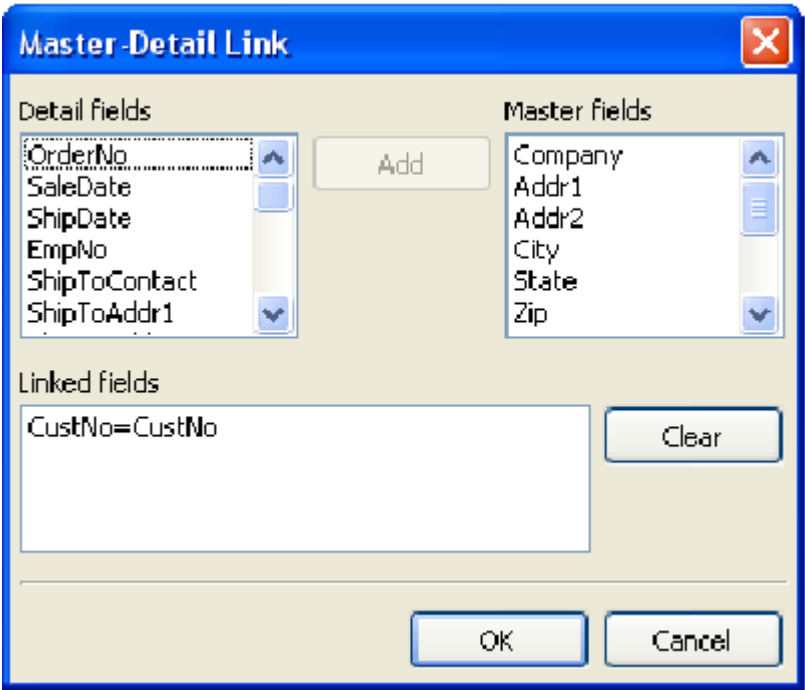
这个组件可以访问数据表，属性介绍：

属性	描述
Active	设置数据表是否被打开
DatabaseName	连结的数据库名称
FieldAliases	设置字段的别名
Filter	过滤条件
Filtered	设置是否通过过滤条件进行过滤
IndexFieldNames	索引字段名称
IndexName	二级索引
MasterFields	连接主表的字段
Master	连结的主表组件
TableName	连接访问的数据表
UserName	数据表的别名

FieldAliases 属性可以设置表和表的字段的别名



MasterFields和master属性用于建立主-细表结构的数据源连接。

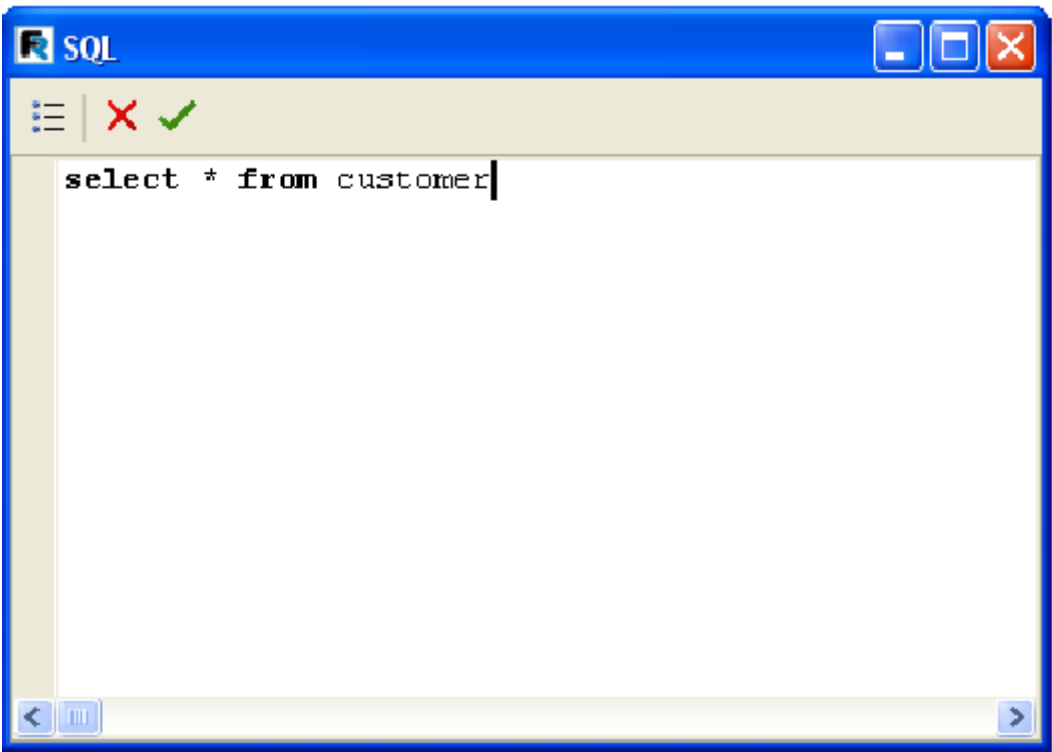


11. 1. 3、TfrxAdoQuery 组件

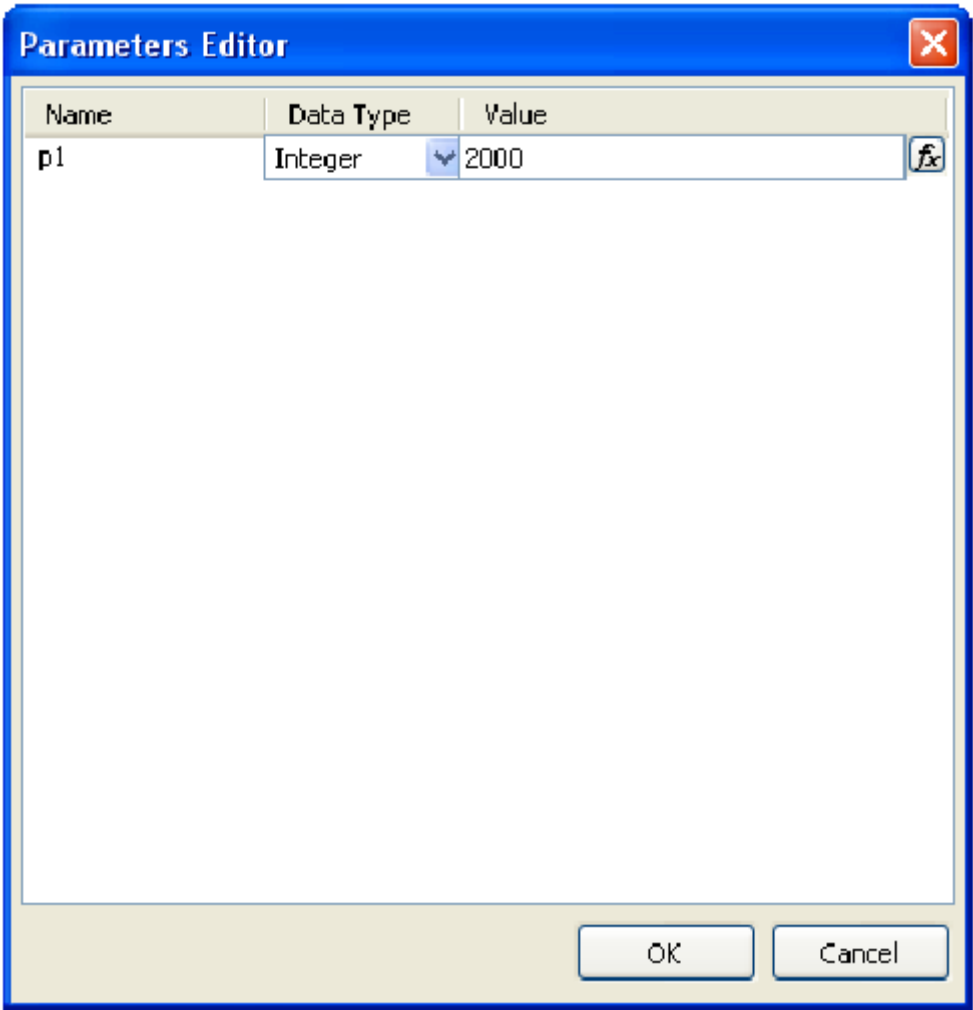
这个组件可以通过 SQL 语句进行访问数据表。属性介绍

属性	描述
Active	设置数据表是否被打开
DatabaseName	连结的数据库名称
FieldAliases	设置字段的别名
Filter	过滤条件
Filtered	设置是否通过过滤条件进行过滤
SQL	输入查询的 SQL 语句
MasterFields	连接主表的字段
Master	连结的主表组件
Params	设置连接访问的数据表时参数
UserName	数据表的别名

SQL 属性语句输入：



Params 属性也有编辑器，如果查询带参数，是用编辑器是非常有用的。



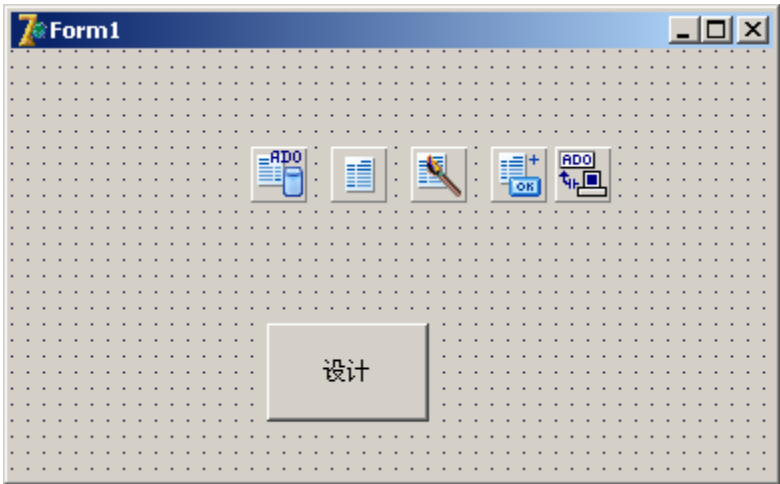
11. 1. 4、TfrxADODatabase 组件

连接数据库字段，他和 delphi 中的 TadoConnection 组件非常相似。属性介绍：

属性	说明
connected	是否对数据库进行连结
DatabaseName	连接数据库的字符串
LoginPrompt	确定每次连结数据表是否进行身份确认

11. 2、创建报表

我们做一个在运行其使用这些组件建立一个简单报表的示范。在 delphi 环境中创建一个新的工程， 并加入以下组件： TfrxADODComponents ， Tbutton ， TfrxReport ， TfrxDesigner ， TfrxDialogControls， TADOConnection。



设置属性：

```
ADOConnection1:  
LoginPrompt = False  
  
frxADODComponents1:  
DefaultDatabase = ADOConnection1
```

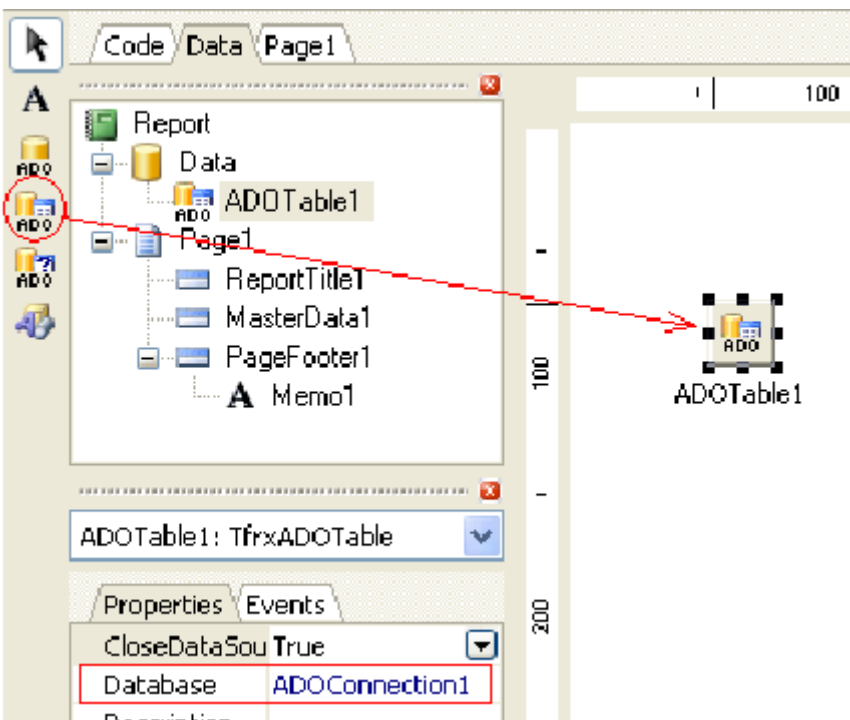
双击设计按钮， 定义一个事件函数：

```
procedure TForm1.Button1Click(Sender: TObject);  
  
begin  
    frxReport1.DesignReport;  
  
end;
```

运行 delphi， 点击设计按钮， 进入报表设计器。

11. 3、简单的列表式报表

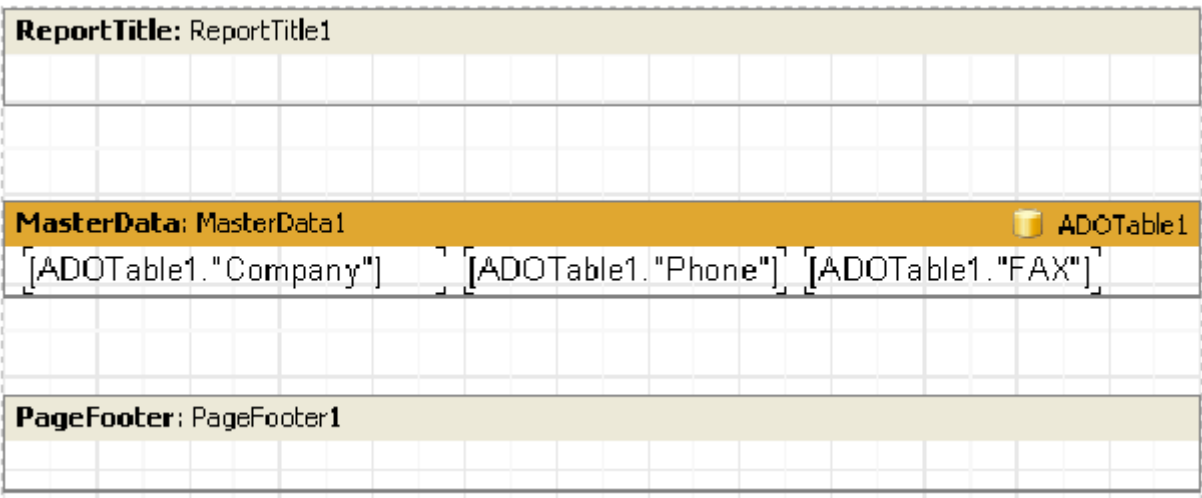
在设计器环境中点击新报表， 并且换到 Data 页， 放置一个 ADOTable 组件到页面上。



注意 database 属性已经连结到了数据库。并选择数据表名称：

```
TableName = 'Customer'
```

到报表设计页， 连接 Master band 数据源， 并在数据树中将数据源字段托到 band 上合适的位置， 调整大小。界面如下：



设计完毕， 点击预览按钮， 浏览结果。

11. 4、参数查询报表

我们创建一个比较复杂一点报表，在报表输出之前，在对话框中输入参数查询。在上面的报表中在点击新报表，重新创建一个空白报表。

切换到 data 页，放置 ADO Query 组件到面板上，双击组件单出编辑器，输入查询语句：

```
select * from Customer where CustNo > :p1
```

加入一个对话框窗体，组织组件如下：



设置组件属性：

Label1:

Caption = '选择CustNo大于'

Edit1:

Text = '2000'

Button1:

Caption = '确定'

ModalResult = mrOk

Button2:

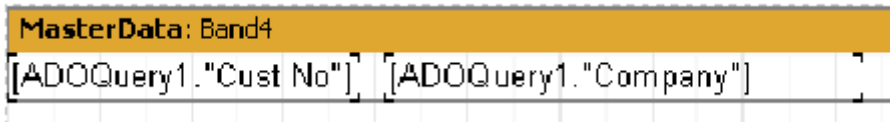
Caption = '取消'

ModalResult = mrCancel

打开 Query 的参数 params 属性编辑器对话框，设置参数：



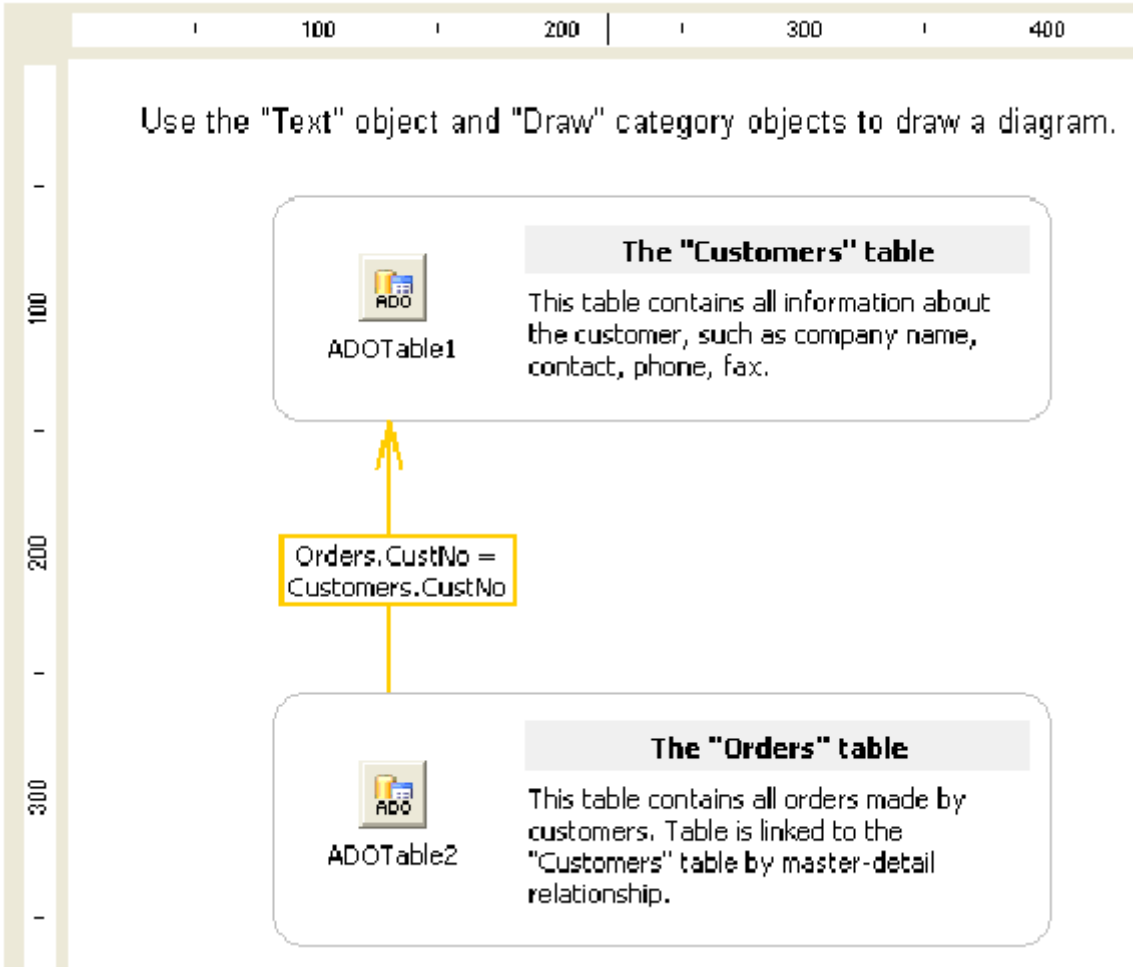
回到报表设计器页面，组织报表结构如下：



进行预览，输入参数，查看显示结果。

11. 5、其他可用配置

还可以在 data 面板上添加 text 和 draw 组件元素，创建如下图的简单的数据结构图：



第十二章

报表的 继承性

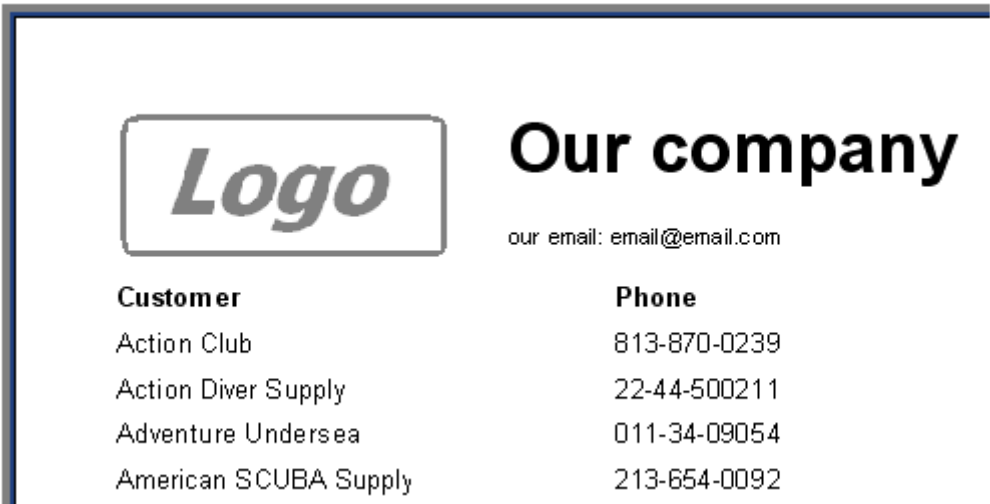
我们经常有相同格式的报表信息。例如表头和表为，或页头页尾包含公司商标和公司信息，Email，地址等。这样可以使用报表的继承性属性来实现。

例如，我们有报表需求的通用信息（email、公司商标、公司地址等）。这些信息填写在报表的页首和页尾。我们可以创建一个基本的报表，它包含着这些信息。其他报表可以使用基础报表进行创建，这样他就可以包含有这些基本信息。

如果要修改这些基本信息，只需要修改基础模板的信息即可，从他继承的报表的这些基本信息相应的就会自动修改。实际上，当打开报表时，首先打开的基础报表，然后在打开继承的报表信息。

12. 1、创建报表

现在我们使用继承来创建一个报表，报表界面如下：

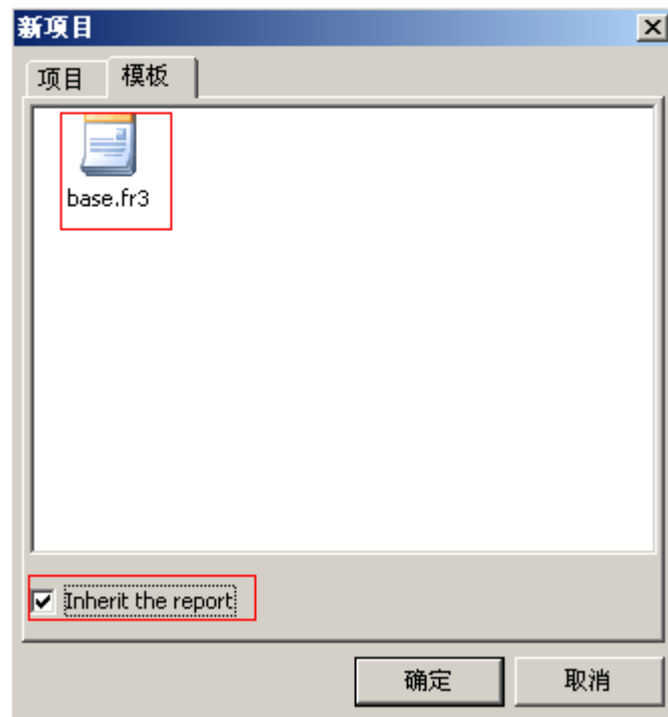


说先创建一个基本模板。组件安排如下：



保存文件为：“base.fr3”。保存路径取决于 TfrxDesigner 组件，默认情况下，系统在应用程序所在的目录搜索模板，可以通过 TfrxDesigner.templatedir 属性来指定模板路径。

现在创建一个继承报表。在文件菜单中选择“新建”。在对话框中选择“模板”页，选择报表模板（base.fr3），选中“Inherit the report”检查筐。

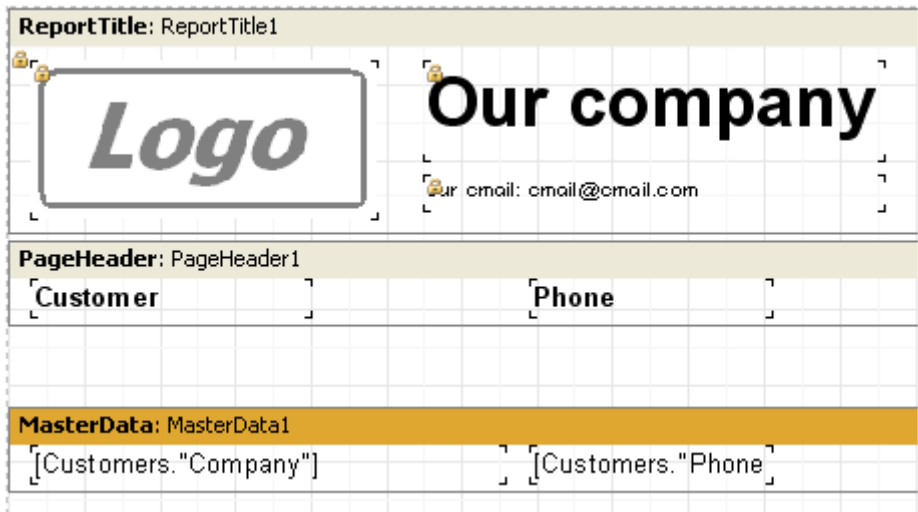


点击确定按钮，创建一个包含全部基础信息的报表，这些组件都被所定，



这就是说不能在此删除和重命名这些组件，并且不能够移动到其他的 band 上去，其他的设置属性可以修改。记住，如果在模板中修改了一些属性，如颜色，则在继承的报表中一样得到修改。如果你想只修改模板的信息，而不修改继承的报表的信息。例如，我们打开继承的报表，修改公司的颜色为红色，在打开模板，修改公司的颜色为绿色，我们再打开继承的报表，公司颜色仍然为红色。

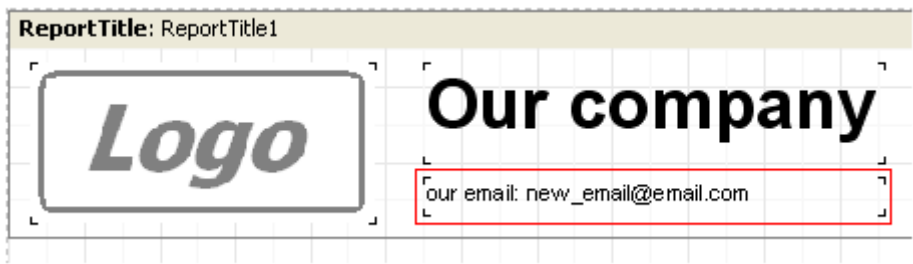
在再报表中添加其他信息。



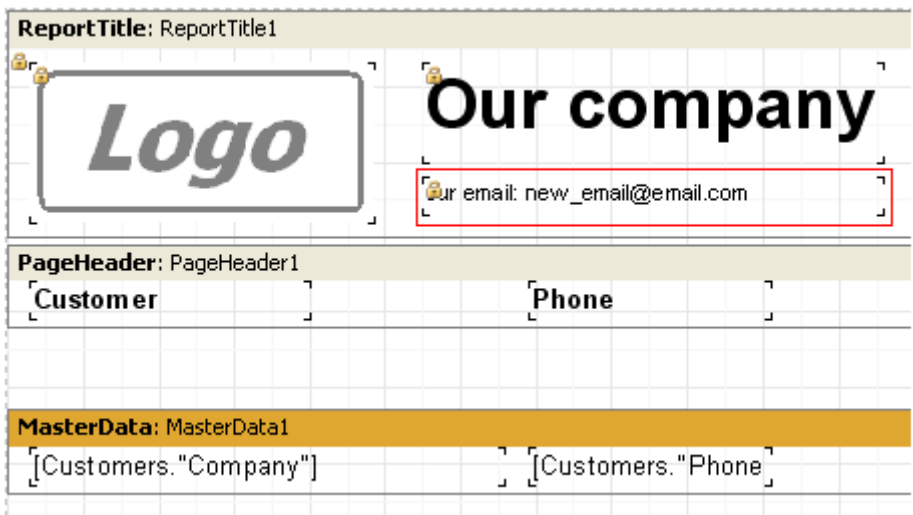
报表设计完成。

12. 2、修改基础模板

打开基础模板 base.fr3，修改 email 值



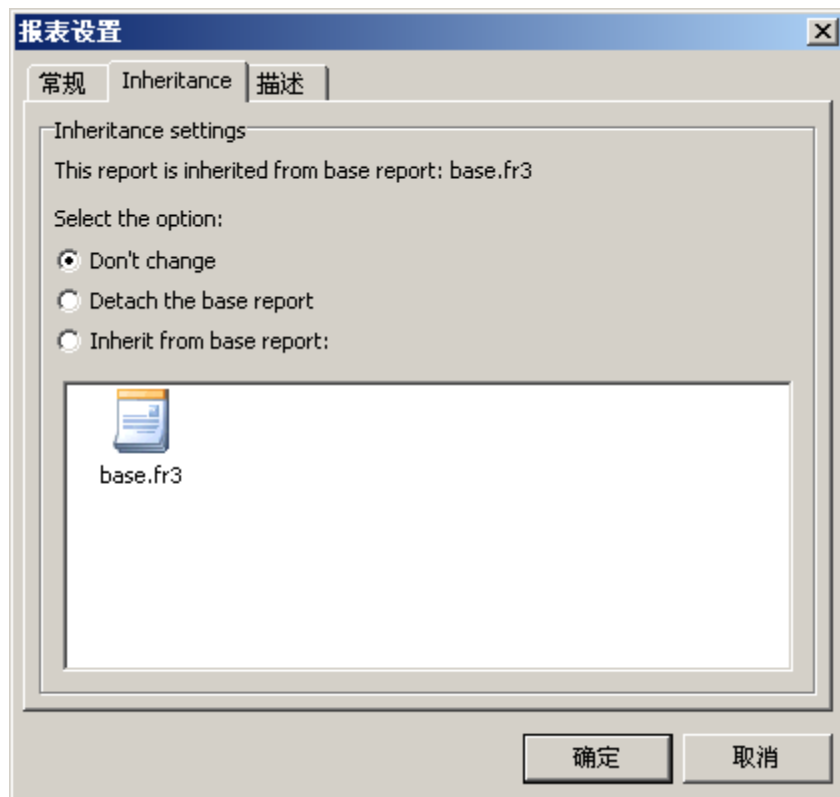
保存基础报表，打开继承的报表，查看 email 也修改了：



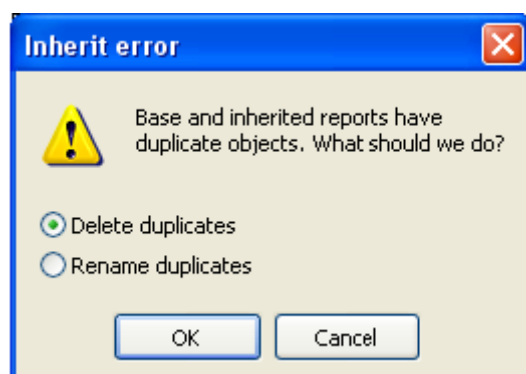
我们在基础模板中加入新的组件如何？但记住：基础模板和继承报表中的组件不能重名。又不能知道有多少个继承的报表使用了这个模板，怎么办呢？可以通过定义命名规则，如组件名称可以定义为：ReportName_ObjectName。

12. 3、组件的继承

我们已经描述了继承报表的创建。假如我们已有的报表如何设置继承模板呢？“报表|选项”菜单打开报表的选项窗口，并且换到“inheritance”页。



我们选择“Inherit from base report”选项并选择要继承的模板，然后点击确定按钮。FastReport 将合并两个报表。也许会出现如下错误信息。



用户可以删除或重命名那些重名的组件。

第十三章

报表向导

FastReport 有几个报表向导简化报表的生成过程。选择“文件|新建”菜单，界面如下：



13. 1、新报表向导

有四个创建新报表的向导。

——标准报表向导

——标准报表

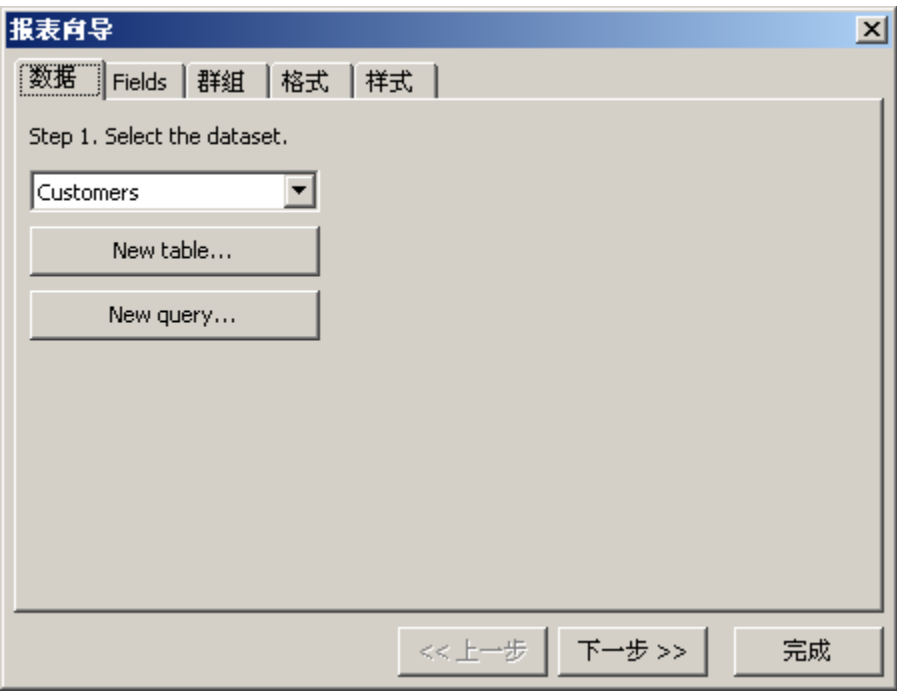
——点矩阵报表向导

——点矩阵报表

标准报表和点矩阵报表向导可以创建空白的标准报表或点矩阵报表。报表包含有一个空白页。

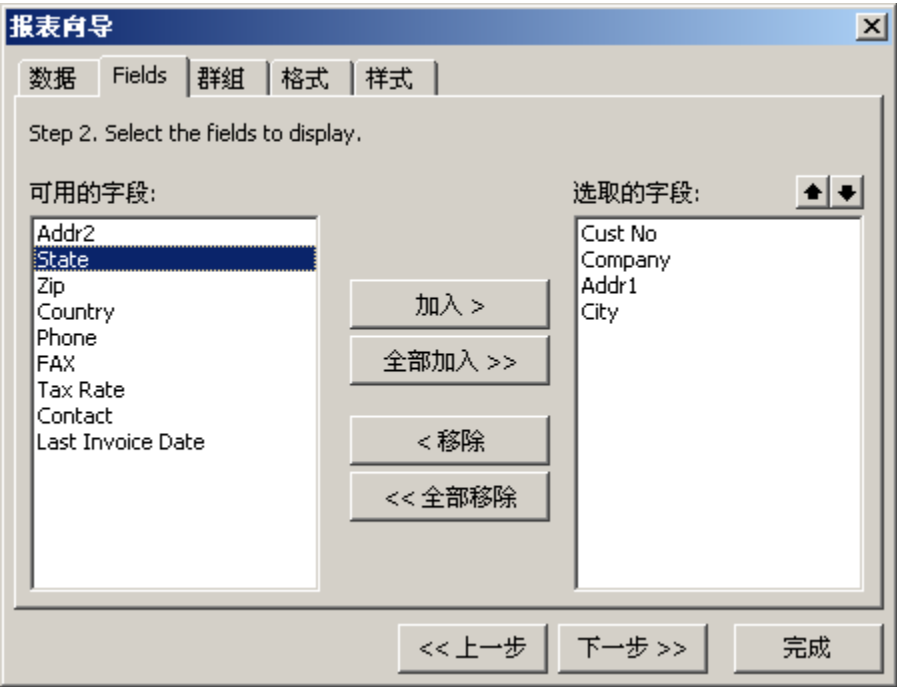
标准报表向导和点矩阵报表向导允许生成报表中选择数据字段，创建组。我们通过标准报表向导创建一个报表。

点击“文件|新建”菜单，选择标准报表向导，弹出向导窗口：



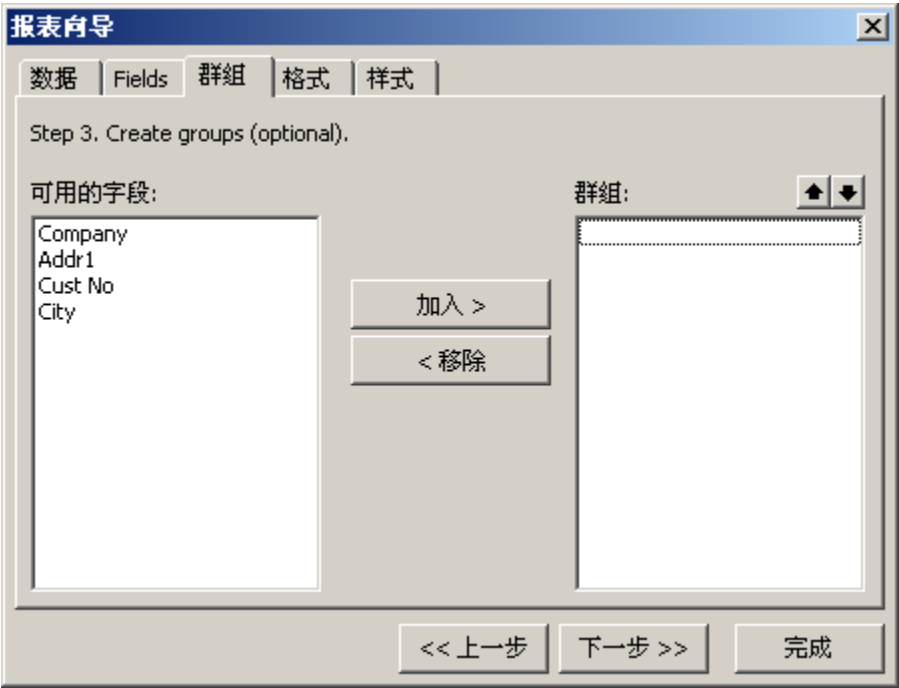
可以看到，这个窗口有多个页标签。第一个页中选择数据源；可以选择任何可用的数据源，也可以创建一个新的表——table 或 query。点击 new table 或 new query 按钮，弹出数据源向导窗口。在此我们选择 customers 数据源，点击“下一步>>”按钮。

下一页选择可以显示的字段。



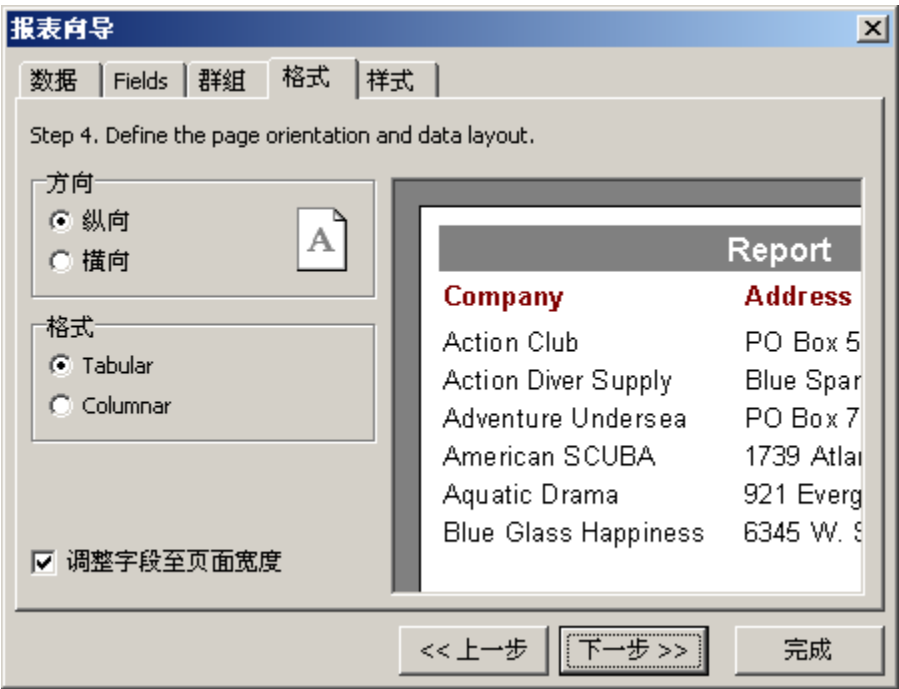
左边列表中显示可以使用的字段，右边列表中显示的是已经选择的字段。通过“加入>>”“全部加入>>”“<移除”“<<全部移除”按钮选择或移除可以显示的字段。通过▲▼按钮，调整选取的字段的顺序。

下一个页创建一个或多个组。这个事例中添加一个 Group Header，Group Footer 两个组。



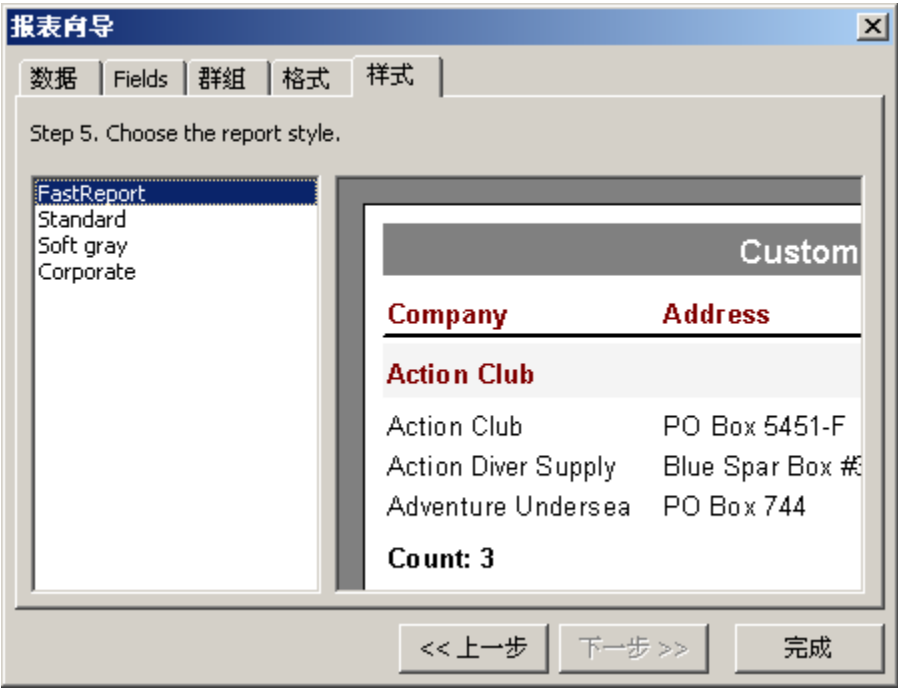
这个页我们点击“下一步>>”按钮，跳过此页。

下一页设置页面属性。

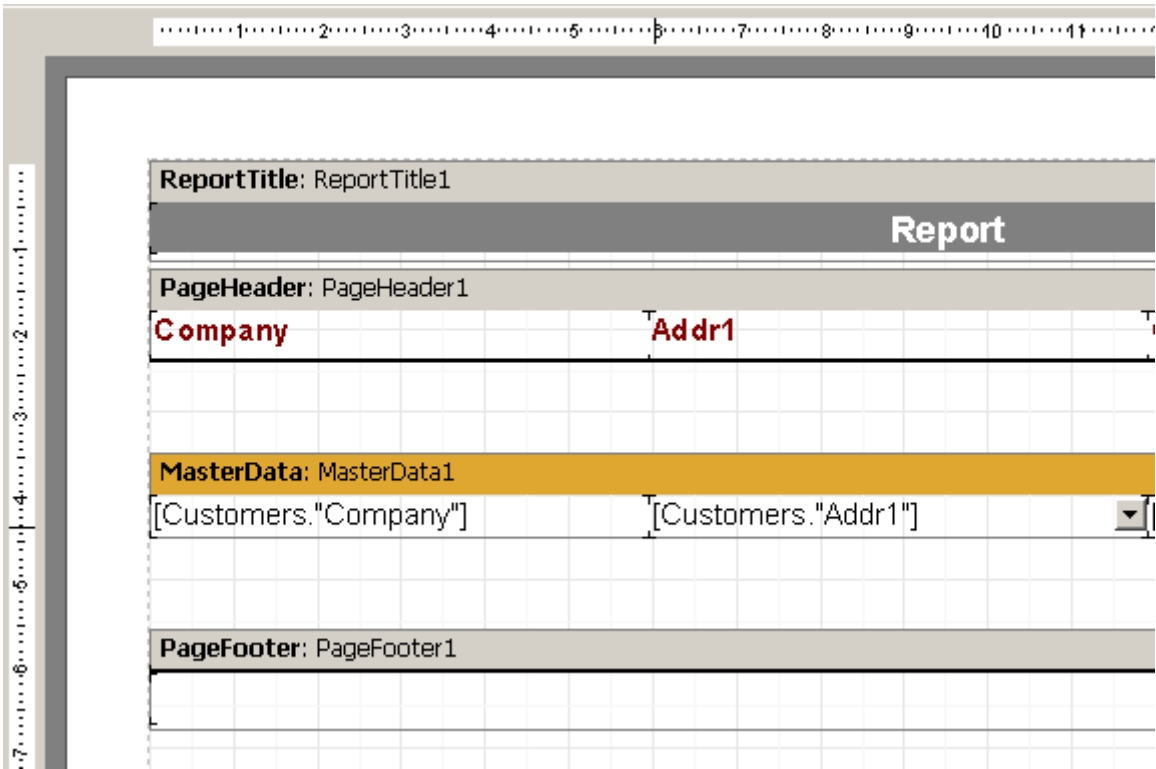


选择显示方式，在右边可以查看显示样式。

最后一页设置显示模式。

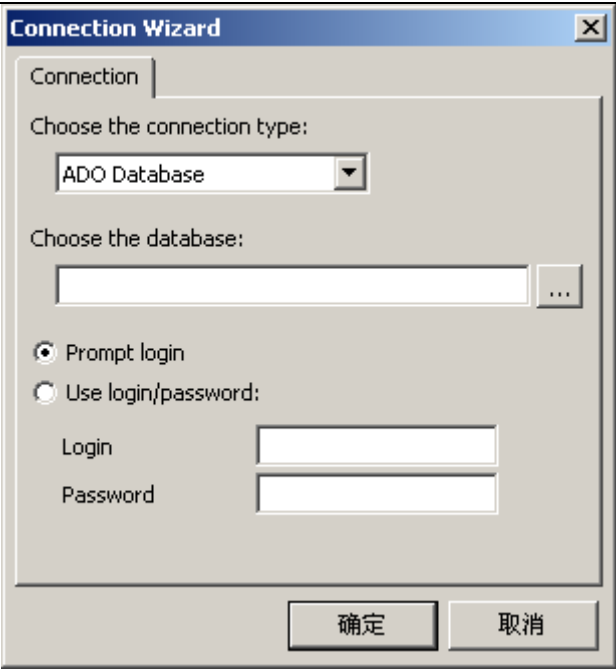


点击“完成”按钮，创建如下报表：



13. 2、数据连接向导

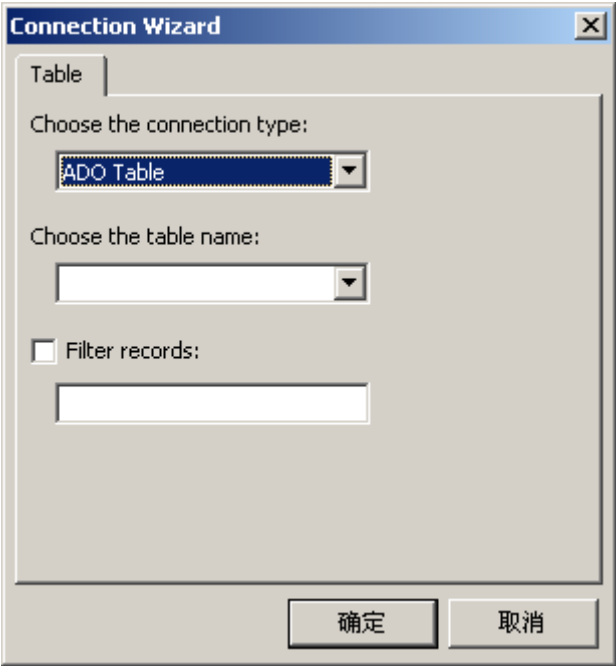
这个向导在当前报表中添加一个新的数据库连接。这个向导将添加一个 TfrxAdoDatabase 组件到报表中。



点击 “...” 按钮，弹出标准数据库连接窗口。并设置登陆用户和登陆密码。
用户可以直接添加一个 TfrxAdoDatabase 组件建立连接。

13. 3、新table向导

这个向导可以在报表中加入一个新的数据表的连接。



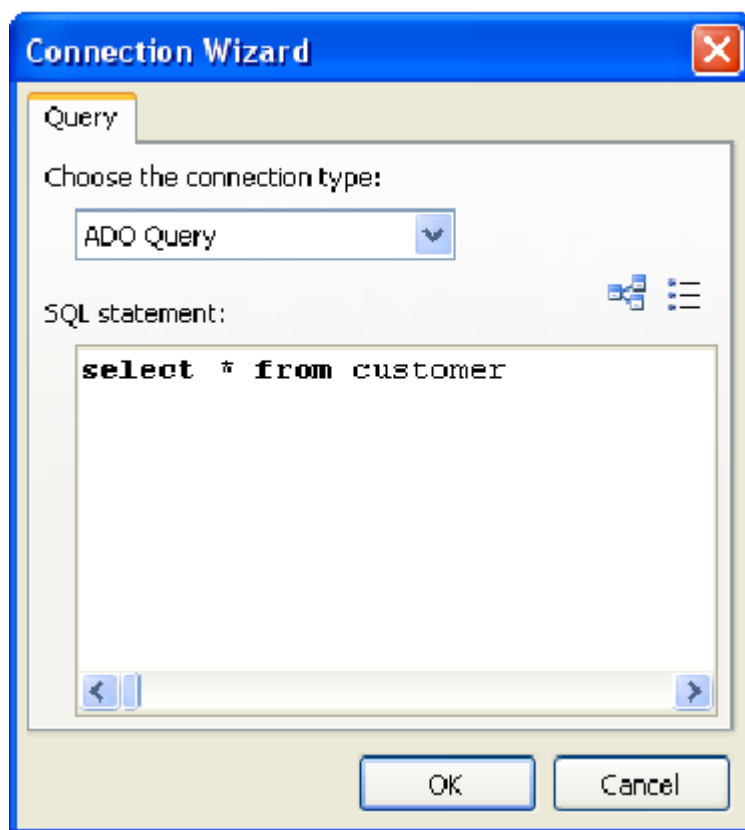
用户选择表，并可以设置过滤条件。

(CustNo > 2000) and (CustNo < 3000)

用户也可以添加一个新的 Ttable 组件到窗口中创建表连接。

13. 4、新query向导

这个向导添加一个 query。

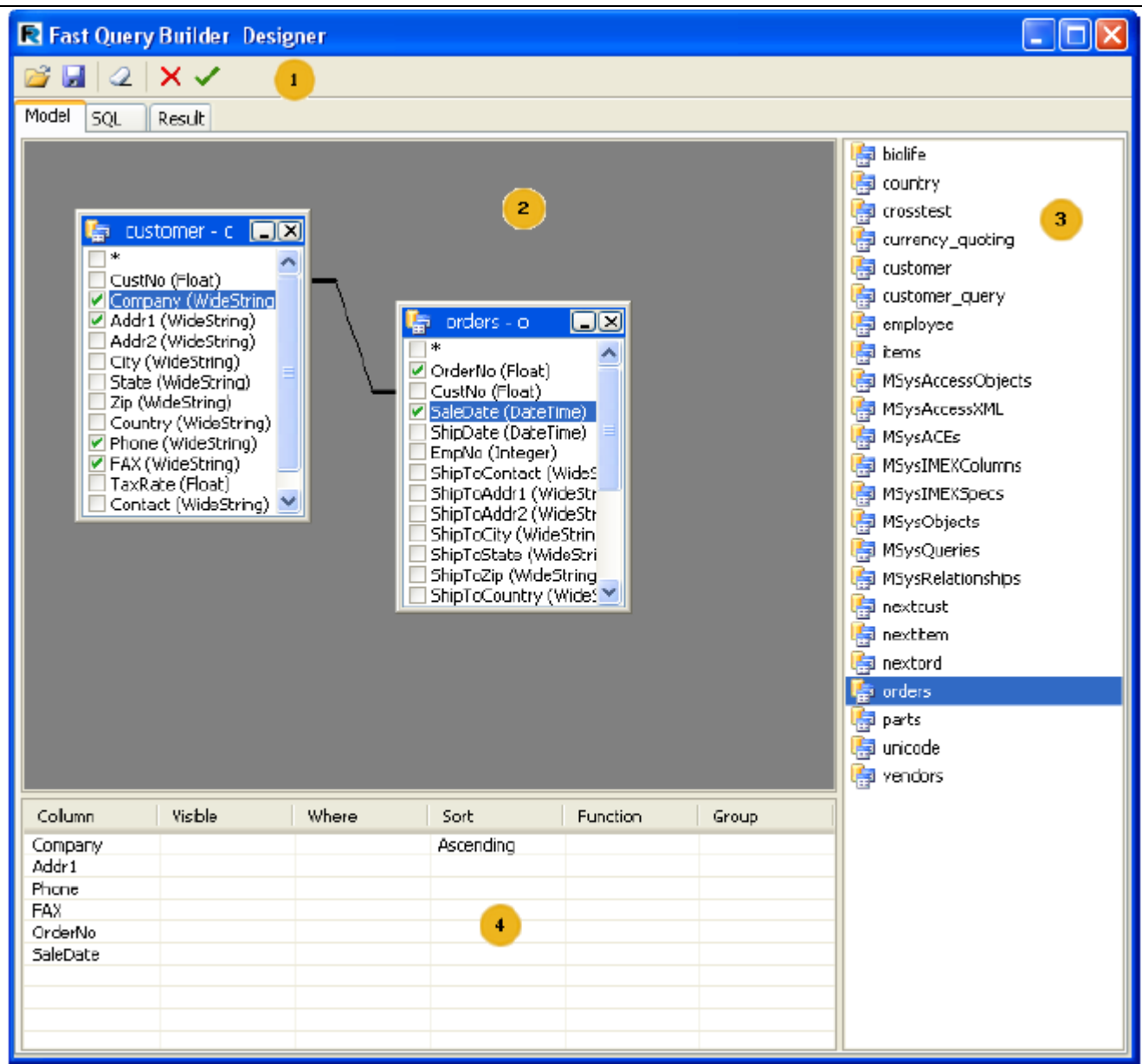


在这添加 sql 语句。可以通过按钮，可视化创建 sql 语句。

可以添加 TfrxAdoquery。

13. 5、查询语句生成

使用FastQueryBuilder可视化创建查询语句。



1: 工具栏

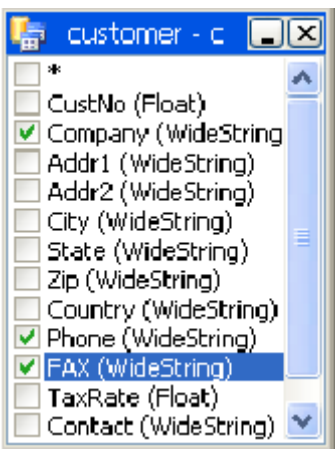
- 2: 设计工作区
- 3: 可选数据表
- 4: 选择数据表的字段

工具栏:

- : 打开 sql 语句
- : 保存
- : 清除
- : 取消

✔：确定完成

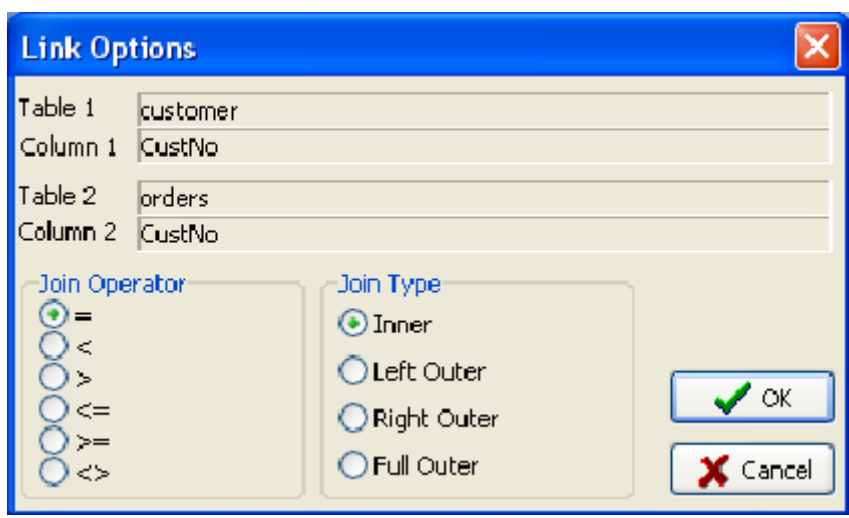
工具支持鼠标托动。在数据表双击或将其托到设计区。



在字段列表设置字段参数

Column	Visible	Where	Sort	Function	Group
Company	<input type="checkbox"/>		<input type="checkbox"/>		<input type="checkbox"/>
Phone			No		
FAX			Ascending		
OrderNo			Descending		
SaleDate					

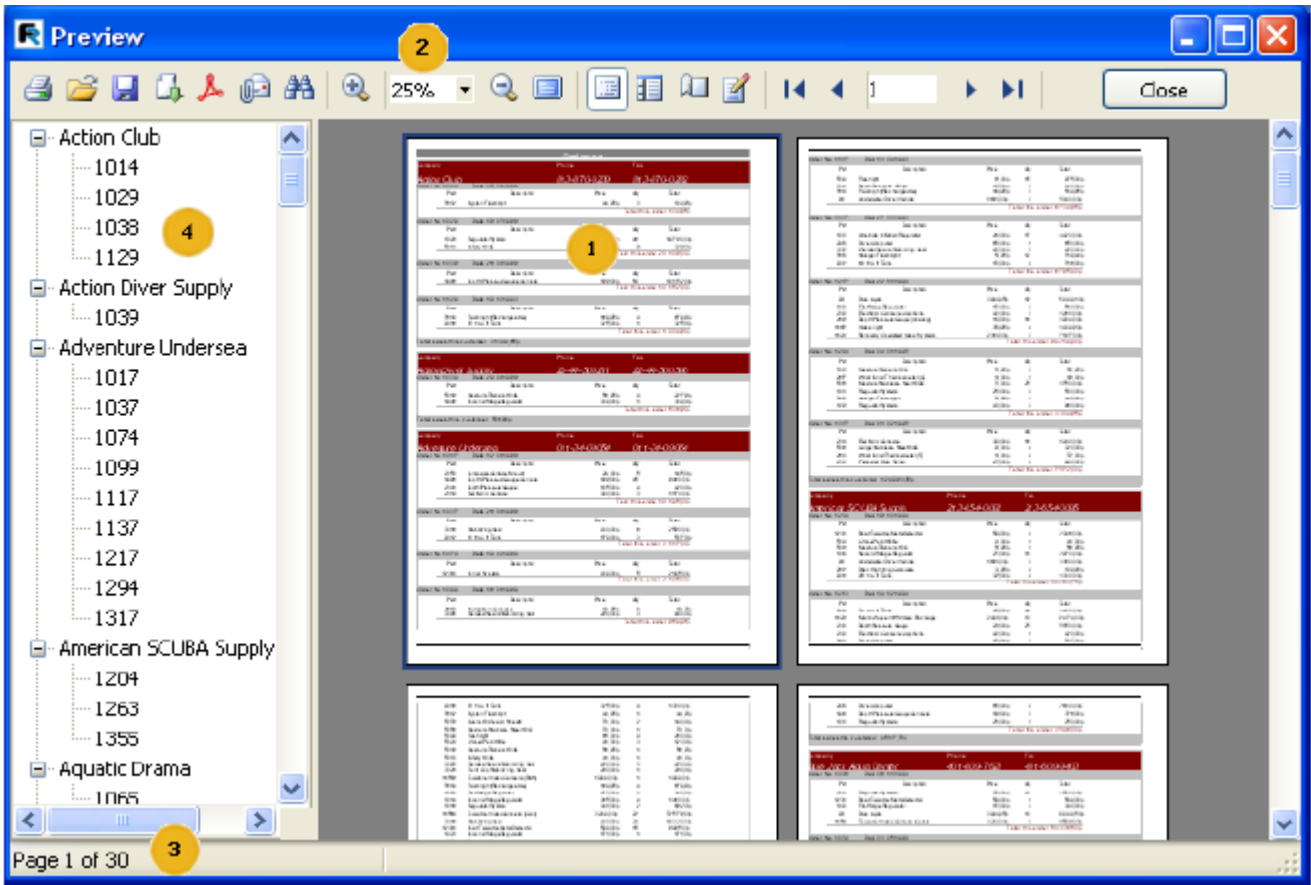
两个数据表后，可以创建表格连接：



第十四章

报表的预览 打印 导出

创建的报表可以预览，打印和导出到报表可以支持的格式。这些在报表预览界面都可以实现。






图中数字说明：

- 1：报表页
- 2：工具栏
- 3：状态栏
- 4：概要区

工具栏按钮说明：



图标	名称	说明
	打印报表	输出当前预栏报表到打印机
	打开报表	从文件中 (*.fr3) 打开预览报表
	保存报表	将当前预览的报表保存到文件中。
	文字查询	在报表预览中查询匹配文字

	放大	放大显示预览
	缩小	缩小显示预览
	全屏	全屏显示预览
	概述	
	页面设置	
	编辑	
	导出	弹出下拉条选择可以支持的导出格式

14. 1、控制键

Keys	Description
Ctrl+S	保存到文件中 “*.fr3”
Ctrl+P	打印报表
Ctrl+F	文字查找
F3	继续查找
Arrows	移动报表
PageUp, PageDown	向上/向下滚动
Ctrl+PageUp, PageDown	上一页/下一页滚动
Home	移到报表开始
End	移到报表结尾

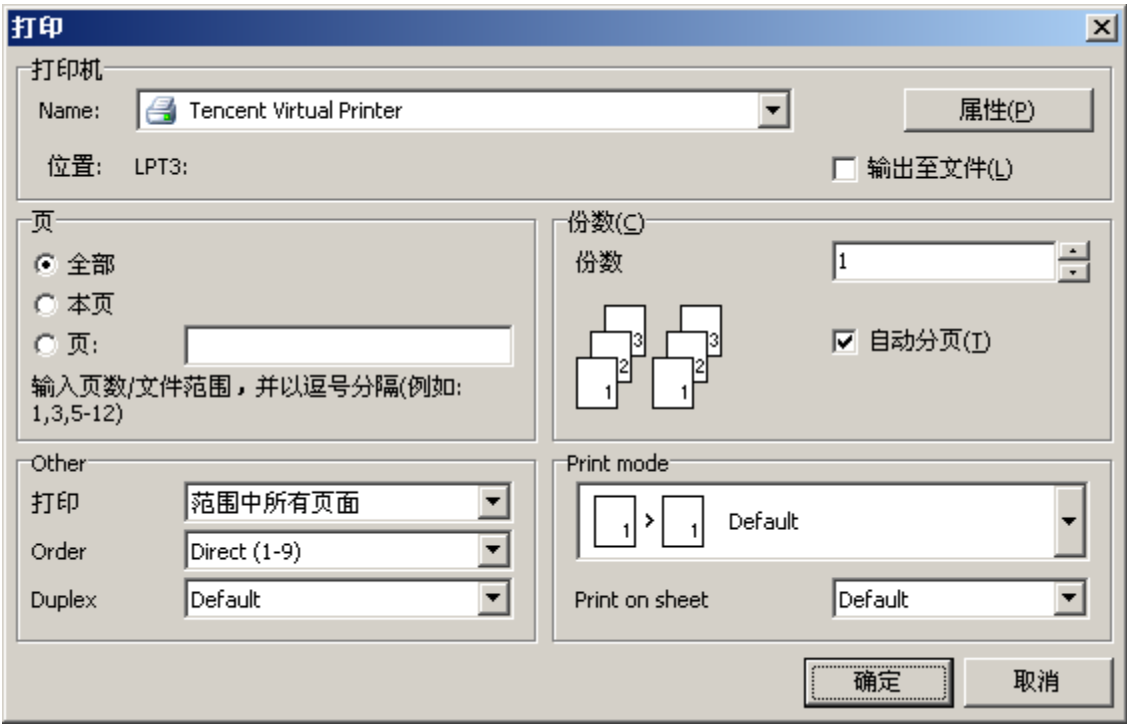
14. 2、鼠标控制

Action	说明
Left button	鼠标状态为“小手”可以移动页面，在工具栏点击放缩按钮。
Right button	弹出上下文右键菜单

Double-click	在全屏模式下双击还原到普通模式。
Mouse scroll	滚动报表页面。


14. 3、报表的打印

点击工具栏打印机图标按钮，弹出打印机选项窗体。



设置相关选项，点击确定按钮，输出到打印机。

14. 4、报表中的文字搜索

FastReport 可以在预览窗口进行文字的搜索。在工具览点击查询按钮，弹出查找设置对话框：



点击确定按钮，开始查找，找到相匹配的字符串进行高亮显示。再按 F3 可以从当前位置向

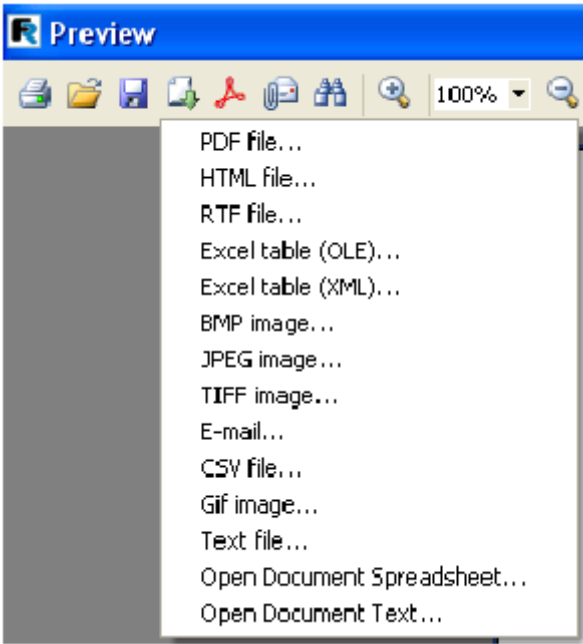
下继续查找上面设置的文字。


Company	Address
Action Club	PO Box 5451-F
Action Diver Supply	Blue Spar Box #3
Adventure Undersea	PO Box 744

14. 5、报表的导出

FastReport 可以将生成的报表到出导不同格式的文件中，可用于将来的修改。如果要支持这些导出能力，需要将 FastReport4 exports 组件面板上的组件放置到 delphi 的窗体上。

FastReport 可以支持 13 格式的导出，他们是：PDF 文件、HTML 文件、RTF 文件、excel table(OLE)、XML 格式、BMP 图形、JPEG 图形、TIFF 图形、CSV 文件，GIF 文件、文本文件、开放文本文件、开放表单文件。



点击按钮从菜单中选择要导出的文件格式，则弹出相应的导出向导窗体，如 PDF



设置相关的属性，点击确定按钮，弹出指定保存目录和保存的文件名，确定后，开始导出。

注：在导出到 PDF 文件中时，如果要支持中文的到出，在报表设计时，应该使用中文字体，如“宋体”“黑体”“隶书”等字体。

[参考]

1、英文原文 UserManual-en.pdf

2、FastReport 汉化组件

翻译：梦醒来

E-Mail:mxl326@sina.com

QQ:93872022