FastReport VCL 6
用户手册

翻译：灰太郎（QQ:1206475516；Email：wxb_km@qq.com）

© 1998-2018 Fast Reports Inc.

手册版本6.0.0

目录

第一章 设计器 2

1控制键 3
2 鼠标控制 3
3 工具栏 4
设计器模具栏 4
标准工具栏 4
文字工具栏 5
Fram e工具栏 6
对齐工具栏 7
4 设计器选择 8
5 报表设置 9
6 页面选项 12

第二章 创建报表 16

1报表对象 16
2 “你好，世界！”报表的例子 16
3 “文本”对象 17
4 “文本”对象中的HTML标记 19
5 在“Text”对象的帮助下显示表达式 20
6 FastReport中的段（band） 21
7 数据段（Data Band） 22
8 TfrxDBDataSet组件 23
9 “客户列表”报表 23
10 借助“Text”对象显示DB字段 26
11 别名 27
12 变量 28
13 “图片”对象 30
14 用图片报道 31
15 多行文字显示 33
16 数据分割 35
17对象的文本换行 37
18 以表格的形式显示数据 39
19 打印标签 41
20 子段 43
21 移动物体 45
22 报表有两个数据级别（主 - 详细信息） 46

©1998-2018FastReportsInc.

23 数据带的页眉和页脚 49
24 多页报表 51
25 RowCount和PageCount属性 52

第三章 组和聚合 54

1 与小组一起报表 54
2 其他组功能 56
3 重置页码 58
4 深入研究小组 58
5 行号 59
6 聚合函数 60
7 页面和报表总计 63
8 插入聚合函数 64

第四章 格式化，突出显示 67

1 格式化值 67
2 内联格式 68
3 有条件的突出显示 69
4 着色备用数据行 72

第五章 嵌套报表（子报表） 75

1 嵌套报表（子报表） 75
2 并排的子报表 75
3 使用子报表的限制 76
4 PrintOnParent选项 77

第六章 脚本 79

1 脚本的味道 79
2 脚本的结构 82
3 “你好，世界！”脚本 84
4 在脚本中使用对象 84
5 从报表的变量列表中调用变量 85
6 引用数据库字段 86
7 在脚本中使用聚合函数 87
8 在报表中显示变量的值 87
9 活动 87
10 使用“OnBeforePrint”事件的示例 89
11 在组标题中打印组和 91
12 “OnAfterData”事件 95
13 服务对象 96
“报表”对象 96

©1998-2018FastReportsInc.

“引擎”对象 97
“大纲”对象 98
14 使用“引擎”对象 99
15 锚 102
16 使用“大纲”对象 103
17 “OnManualBuild”页面事件 107
18在脚本中创建对象 111
第七章 交叉表报表 114
1 构建交叉表报表 115
2 改变外观 118
3 使用功能 120
4 排序值 121
5 带有复合标题的表 121
6 调整单元格宽度 123
7 字体颜色和突出显示 125
8 在脚本中管理交叉表 126
9 调整行/列大小 130
10 手动填写表格 131
11 将外部对象添加到表中 133
12一些有用的设置 136
第八章 表类型报表 140
1配置列 140
2 管理列的大小 140
3配置行 141
4 管理行的大小 141
5 配置单元格 142
6 加入和分裂细胞 142
7 在单元格中插入对象 143
第九章 图表 145
1 限制图表值的数量 149
2 一些有用的设置 150
3 手动输入值的图表 150
4 从脚本完成图表 151
5 打印使用Delphi构建的图表 152
第十章 地图 154
1个地图元素 154
2 用鼠标控制地图 155

©1998-2018FastReportsInc.

3 “地图”对象编辑器 155
添加m ap层 156
设置图层外观 157
设置标签显示 158
将m ap层连接到数据 158
使用颜色突出显示数据 160
使用大小突出显示数据 162

第十一章 互动报道 165

1超链接 165
2 超链接配置 166
3 链接到URL 166
4 链接到页码 167
5 链接到锚点 167
6链接到详细报表 167
7 链接到详细页面 168
8自定义链接 168

第十二章 点阵报表 170

1 点阵中的交叉表 173
2 点阵报表打印 174
3 “命令”对象 175

第十三章 对话形式 178

1 控制 178
2 “你好，世界！”报表 179
3 输入参数并将其传输到报表中 180
4 与控件交互 181
5 几种对话形式 182
6 管理对话形式 183

第十四章 数据访问组件 186

1 组件说明 187
TfrxDBLookupComboBox 187
TfrxADOTable 188
TfrxADOQuery 190
TfrxADODataBase 192
2 报表设计器 192
3 简单报表“列表”类型 193
4 使用参数查询报表 194
5 其他有用的功能 196

第十五章 报表继承 198

1 创建基本报表 198

©1998-2018FastReportsInc.

2 更改基本报表 200
3 继承控制 201

第十六章 奇才 204

1 新报表向导 204
2 新连接向导 208
3 新表向导 209
4 新查询向导 210
5 查询构造 210
使用Query构造函数 213
复杂查询构建 215

第十七章 预览，打印，导出 219

1 控制键 220
2 鼠标控制 221
3 报表打印 221
4 报表中的文本搜索 223
5 保存报表 224
通过Em ail发送报表 226
导出到FTP 229
导出到Dropbox 230
导出到OneDrive 231
导出到Google云端硬盘 232
导出到Box.com 234
6 报表导出 235
导出为PDF表格 236
导出到Open Docum ent 237
导出到RTF表格 238
导出到Word 2007 239
导出到Excel 97/2000 / XP 239
导出到Excel XML表单 241
导出到Excel 2007 241
导出到Pow erPoint 2007 242
导出到CSV表格 243
导出到HTML表单中 244
导出到文本表单 245
出口到Jpeg, BMP, Gif, Tiff Graphic Form ats 246
7 报表设计注意事项 247

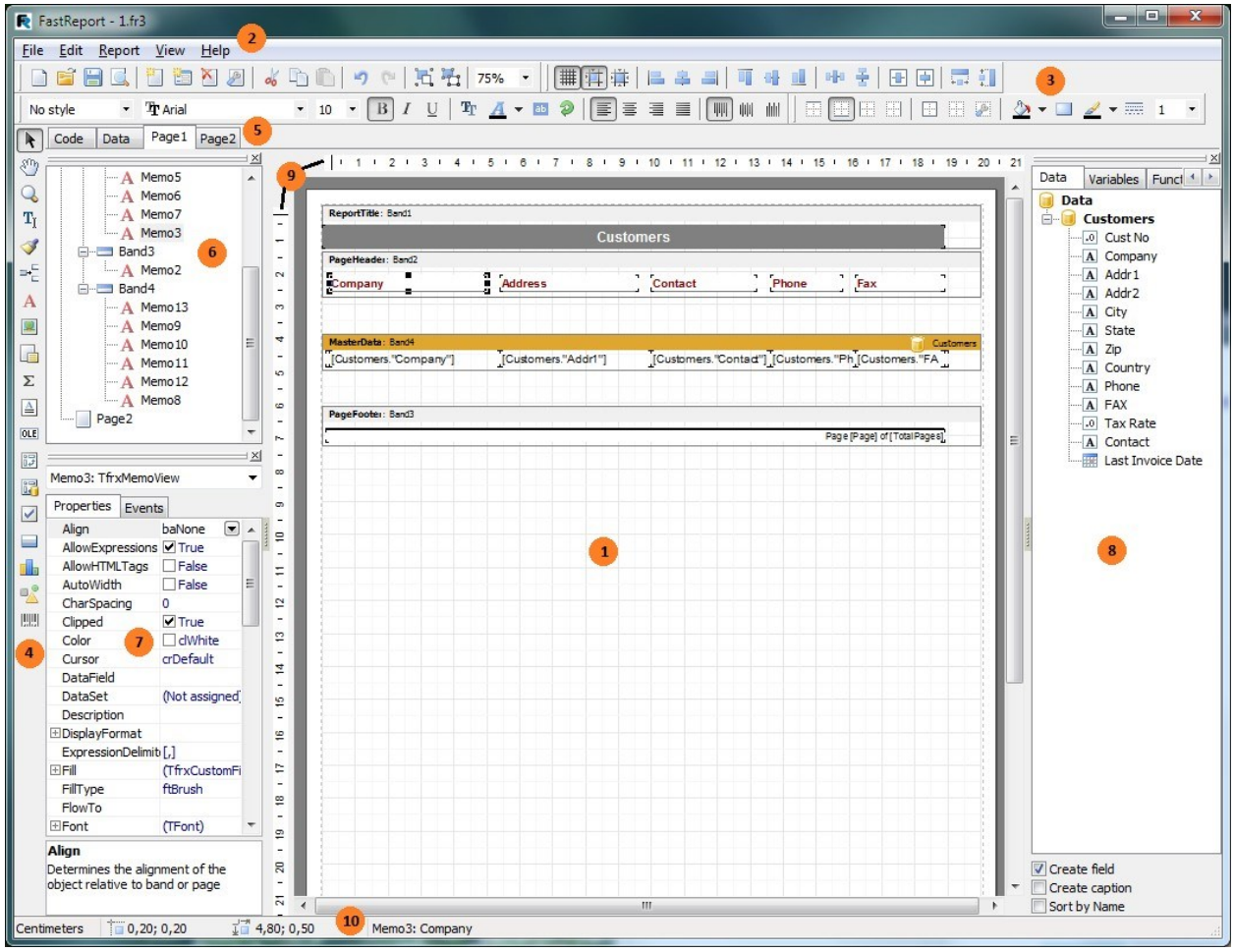
©1998-2018FastReportsInc.

章节 1

设计器

报表组件提供了嵌入式可视化报表设计器，可以通过双击TfrxReport组件在设计时打开它。设计人员为用户提供设计和预览报表所需的所有工具。设计人员的界面符合当前要求。它包含几个工具栏，可以在任何需要的地方停靠。每次打开设计器时都会恢复这些工具栏位置。工具栏位置以及其他设计器设置存储在注册表中，或者如果已分配，则存储在ini文件中。

为了使项目的最终用户能够设计报表，您应该将FastReport组件面板中的“TfrxDesigner”组件放到Delphi表单上，或者将“frxDesgn”单元添加到单元的Uses列表中。在运行时使用设计器允许用户更改报表的外观，以及编辑完成的报表。
注意：您还应该放置将在Delphi表单上使用的任何其他Tfrx组件。



报表设计器功能的关键：
1 –报表设计工作区
2 –菜单栏
3 –工具栏
4 –对象工具栏
5 –报表页面标签

6 – “报表树”窗格
7 – “对象检查器”窗格
8 – “数据树”窗格：可以将元素从此窗格拖到报表页面上
9 – 标尺：可以将标尺拖到报表页面上以在页面上创建蓝色指南（对象与附近的指南对齐）
10 – 状态栏

1.1 控制键

按键	描述
按Ctrl + O	“文件>打开...”菜单命令
按Ctrl + S	“文件>保存”菜单命令
按Ctrl + P	“文件>预览”菜单命令
按Ctrl + Z	“编辑>撤消”菜单命令
按Ctrl + C	“编辑>复制”菜单命令
按Ctrl + V	“编辑>粘贴”菜单命令
按Ctrl + X	“编辑>剪切”菜单命令
按Ctrl + A	“编辑>全选”菜单命令
箭头，标签	在对象之间移动
德尔	删除所选对象
输入	选定对象的打开编辑器
Shift +箭头	修改所选对象的大小
按Ctrl +箭头	移动所选对象
Alt +箭头	将所选对象以指定方向附加到相邻对象

1.2 鼠标控制

手术	描述
左键	选择对象；粘贴新对象；移动或调整对象大小 对于选定对象，通过拖动所选对象组左下角的红色方块来放大和缩小
右键	选定对象的上下文菜单
双击	打开对象的编辑器 双击空白区域可打开“页面设置”对话框
鼠标滚轮	滚动报表页面
Shift +左键	切换对象选择

按Ctrl +左键	通过移动鼠标创建框架 释放按钮选择框架中捕获的所有对象也可以单击空白区域， 并根据需要移动鼠标
Alt +左键	如果选择“文本”对象，则编辑内容

1.3 工具栏

1.3.1 设计器模式栏

“模式”工具栏与设计器窗口左侧边缘的“对象”工具栏集成在一起，并具有以下按钮：

图标	名称	描述
	对象选择	标准操作模式：鼠标光标选择对象，修改其大小等。
	手	允许拖动整个报表页面
	放大	允许左键单击放大 （如果已经25-75%，则增加25%，否则增加 100%）或右键单击缩小 （减去25%，如果已经50-100%，否则减去100%）按住鼠标左 键同时拖动放大到选定区域
	文本编辑器	允许就地编辑“Text”对象 按住鼠标左键并移动光标会创建并调整新的“文本”对象并打开其 编辑器
	格式化复制	允许从一个“Text”对象格式化复制到其他对象：选择“Text”对 象， 单击格式模式按钮， 依次选择目标“文本”对象 退出通过单击任何其他模式按钮格式化复制模式

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	从文件热键组合打开现有报表 “Ctrl+O”

	保存报表	将报表保存到文件 热键组合 – “Ctrl + S键”
	预习	预览报表 热键组合 – “Ctrl+P”
	新报表	添加新页面进行报表
	新对话页面	添加新的对话形式进行报表
	删除页面	删除当前页面
	页面设置	打开页面属性对话框
	切	将所选对象剪切到剪贴板热键组合 – “Ctrl+X”
	复制	将所选对象复制到剪贴板热键组合 – “Ctrl+C”
	糊	从剪贴板热键组合中粘贴对象 – “Ctrl+V”
	解开	撤消上次操作 热键组合 – “Ctrl+Z”
	重做	重做上次取消（撤消）操作热键组合 – “按Ctrl + Y”
	组	将所选对象分组
	取消组合	取消组合所选对象
75%	放大	设置缩放系数

1.3.3 文字工具栏



图标	名称	描述
没有风格	样式	显示所选“文本”对象的样式 从下拉列表中选择以更改“报表>样式”中所选对象的样式定义样式
宋体	字体名称	显示所选“文本”对象的字体 从下拉列表中选择以更改所选对象的字体，显示列表头部使用的最后五种字体
10	字体大小	显示所选“文本”对象的字体大小 从下拉列表中选择以更改所选对象的字体大小 也可以直接在编辑框中输入数字
B	大胆（切换）	切换字体大胆打开/关闭“文本”对象内容的样式
I	斜体（切换）	切换字体斜体“文本”对象内容的斜体样式

	下划线（切换）	切换字体打开/关闭“文本”对象内容的下划线样式
	字体设置	打开字体设置对话框
	字体颜色	下拉字体颜色选择网格
	突出	打开高亮对话 可以设置突出显示的条件
	文字轮换	从下拉列表中更改文本旋转
	左对齐	left在“Text”对象的框架内对齐文本
	居中对齐	center将文本对齐在“Text”对象的框架内
	对齐	右对齐“文本”对象框架内的文本
	辩解	证明“文本”对象框架内的文本
	顶部对齐	top在“Text”对象框架内对齐（垂直）文本
	对齐中间	中间在“文本”对象的框架内对齐（垂直）文本
	对齐底部	底部对齐“文本”对象框架内的（垂直）文本

1.3.4 框架工具栏



图标	名称	描述
	顶线（切换）	打开/关闭“文本”对象的顶部框架线
	底线（切换）	打开/关闭“文本”对象的底部框架线
	左线（切换）	打开/关闭“文本”对象的左框架线
	右线（切换）	打开/关闭“文本”对象的右框架线
	所有框架线	设置“Text”对象的所有四个框架线
	没有框架	设置“文本”对象的所有四个框架线关闭
	帧编辑器	调用帧编辑器对话框
	背景颜色	从下拉列表中设置背景颜色。
	填充编辑器	调用填充编辑器对话框
	框架颜色	下拉框架线颜色选择网格
	框架样式	下拉框架线样式选择列表
1	框架宽度	从下拉列表中设置框线宽 也可以直接在编辑框中输入数字（0.1到10）

1.3.5 对齐工具栏

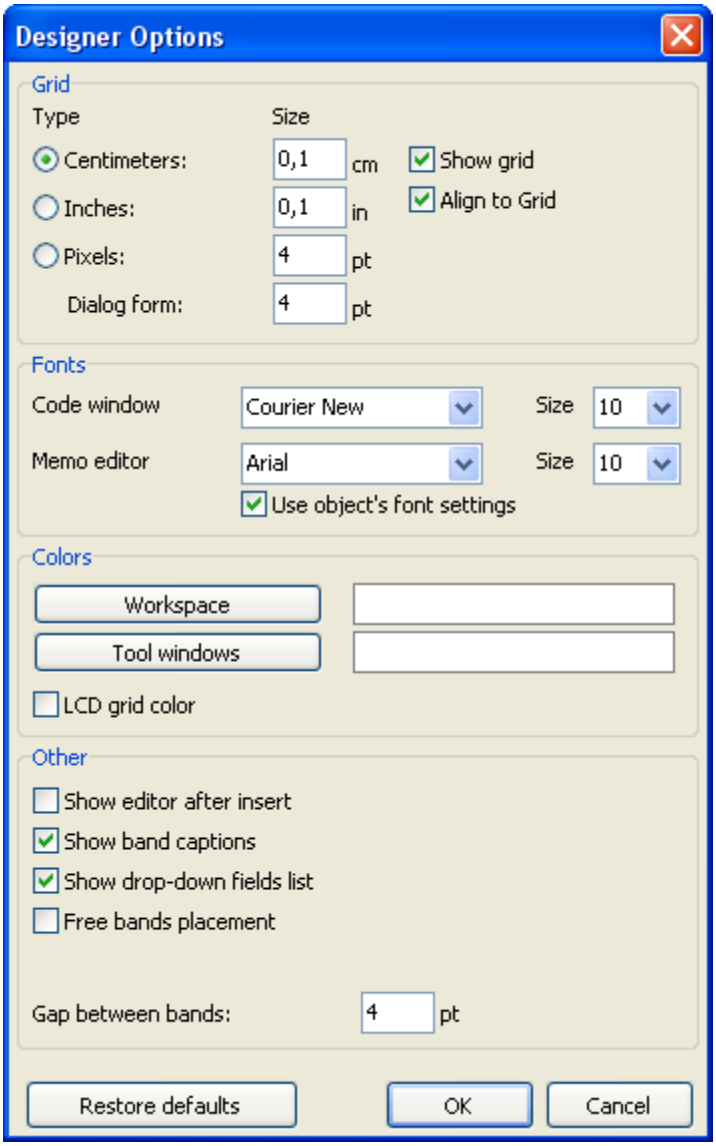


请注意，当选择多个对象时，某些按钮将处于活动状态。

图标	描述
	显示网格（切换开/关）
	对齐网格（切换开/关）
	适合网格
	对齐左边（到第一个选定的对象）
	对齐水平中心（到第一个选定的对象）
	对齐右边缘（到第一个选定的对象）
	对齐顶边（到第一个选定的对象）
	对齐垂直中心（到第一个选定的对象）
	对齐底边（到第一个选定的对象）
	水平方向上的空间相等
	垂直方向上的空间相等
	在段内水平地将每个物体对中
	在段内垂直地将每个物体对中
	设置相等的宽度（与第一个选定的对象相同）
	设置相等的高度（与第一个选定的对象相同）

1.4 设计器选择

通过“视图>选项...”菜单命令设置设计器选项。



在这里，您可以设置首选单位（厘米，英寸，像素）和网格间距。提示：您还可以通过双击状态栏左侧显示当前单位的部分，在设计器中循环显示单位。

您可以将网格可见性和对齐设置为网格。这也可以通过设计器内“标准”工具栏中的按钮完成。

您可以为代码编辑器窗口和“文本”对象编辑器设置字体。如果启用了“使用对象的字体设置”选项，则文本编辑器窗口中的字体将与正在编辑的对象的字体匹配。

可以通过“工作区”和“工具窗口”按钮修改设计器工作区和工具窗口的默认白色背景。

“LCD网格颜色”选项可稍微增加网格线的对比度，并提高其在LCD显示器上的可见性。

“插入后显示编辑器”选项控制插入新对象时发生的情况。如果启用该选项，则每次插入对象时都会显示其编辑器。创建大量空白对象时，建议暂时禁用该选项。

禁用“显示段标题”选项会隐藏段标题，从而在设计页面上节省一些空间。禁用时，段标题显示在段工作区内。

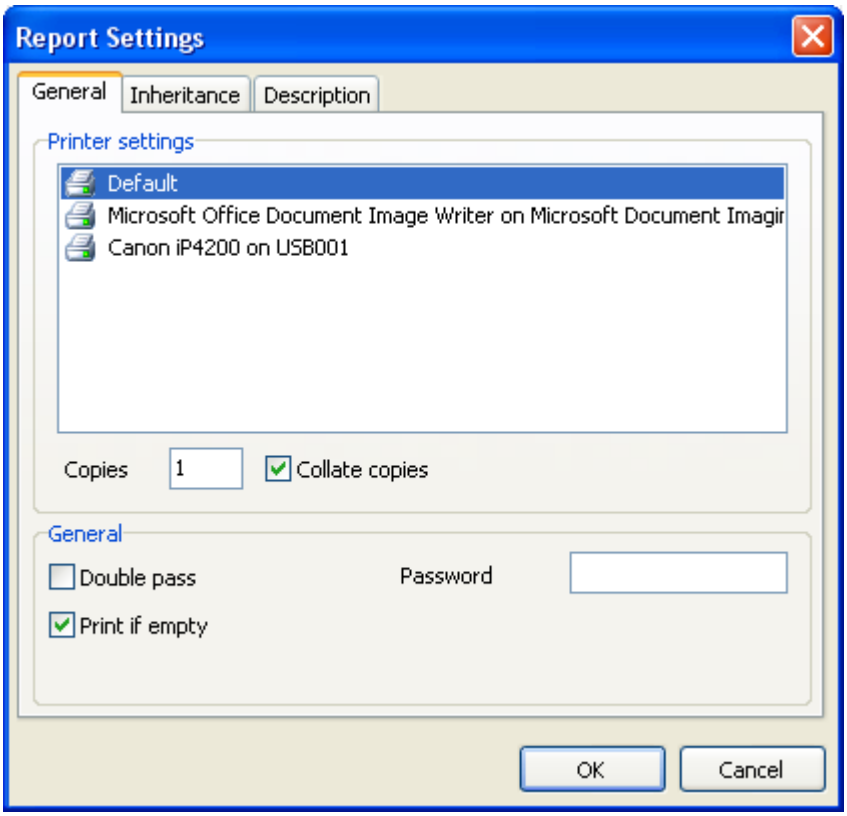
“显示字段下拉列表”选项控制在使用鼠标指向连接到数据字段的“文本”对象时是否可以访问下拉列表。如果段中有许多狭窄的“文本”对象，则可能需要这样做

“自由段放置”选项禁用对页面的捕捉。默认情况下禁用此选项，并根据功能自动将页面分组到页面上。频带之间的间隙在“频带间隙”字段中设置。

1.5 报表设置

“报表设置”对话框可从“报表>选项...”菜单中获得。对话有三页。

在第一页上，您可以看到报表的常规设置：



您可以将报表绑定到系统中安装的其中一台打印机。这意味着在打印该报表时，所选打印机将成为默认打印机。在系统中有多个不同的打印机的情况下，这可能很有用；例如，文本文档可以绑定到单色打印机，而具有图形的文档可以绑定到彩色打印机。打印机设置中列出了“默认” - 如果选择此选项，报表将不会绑定到任何特定的打印机，而是打印在系统的默认打印机上。

您还可以设置要打印的报表副本数以及是否整理输出。打印报表时，此对话框中设置的值将显示在“打印”对话框中。

如果检查“双通”标志，则将分两步执行报表生成。在第一次传递期间，会创建报表草稿并将其分为多个页面，但无法预览。在第二轮中，草稿报表将转换为标准报表，然后保存在预览流中。

为什么需要两次通行证？大多数情况下，此选项用于报表需要访问报表中总页数的情况，即用作“第1页，共15页”。总页数在第一次通过期间计算，并通过“TOTALPAGES”系统变量提供。最常见的错误是尝试在单遍报表中使用此变量，此时变量将返回0。

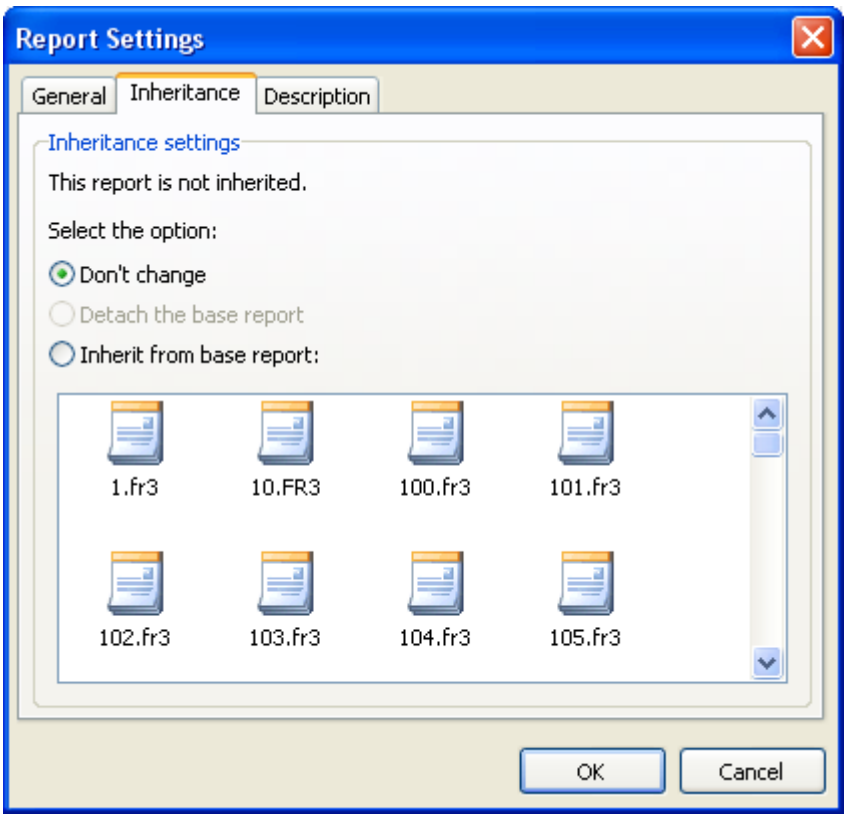
两次通过的另一个用途是在第一遍中执行一些计算并在第二遍中显示结果。例如，当要在组头中而不是在通常的组页脚中显示总和时。通过在对象的“OnBeforePrint”事件中编写报表脚本代码来完成此类型的计算。

“Print if empty”标志允许创建不包含数据行的报表。如果是这个选项

将不会创建禁用的空白报表。

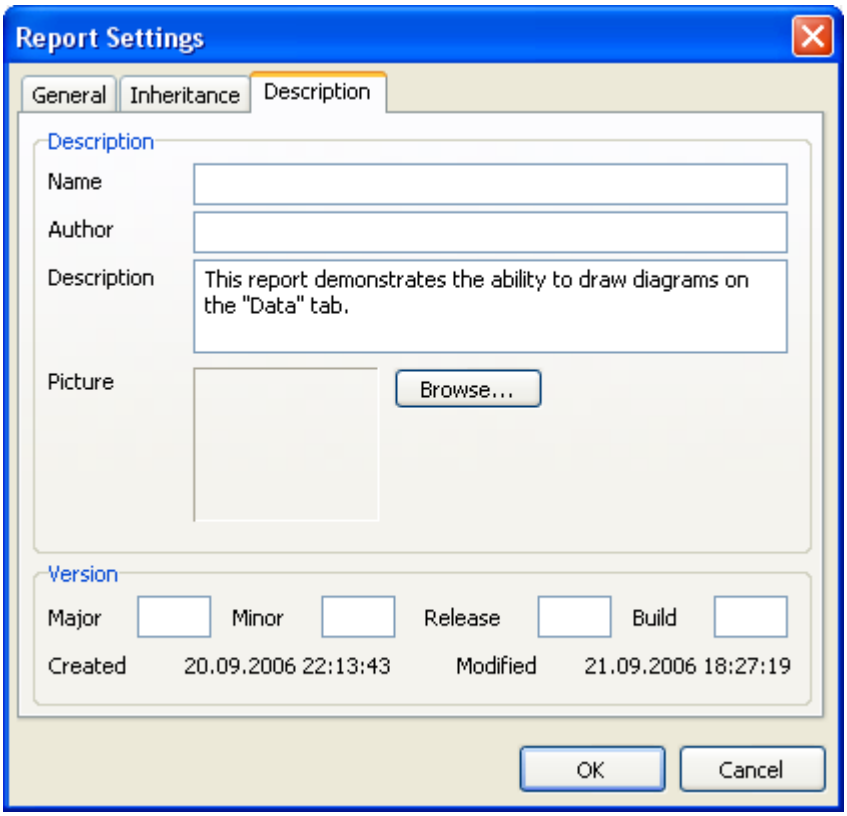
“密码”字段可以设置密码，必须在打开报表之前输入密码。

在第二页上，您可以设置报表继承选项：



稍后您将在“报表继承”一章中了解继承。在此对话框中，您可以看到基本报表的名称（如果报表是继承的），则分离基础报表（在这种情况下，您的报表将是独立的和非继承的）或从其中一个可用的基础报表继承报表。

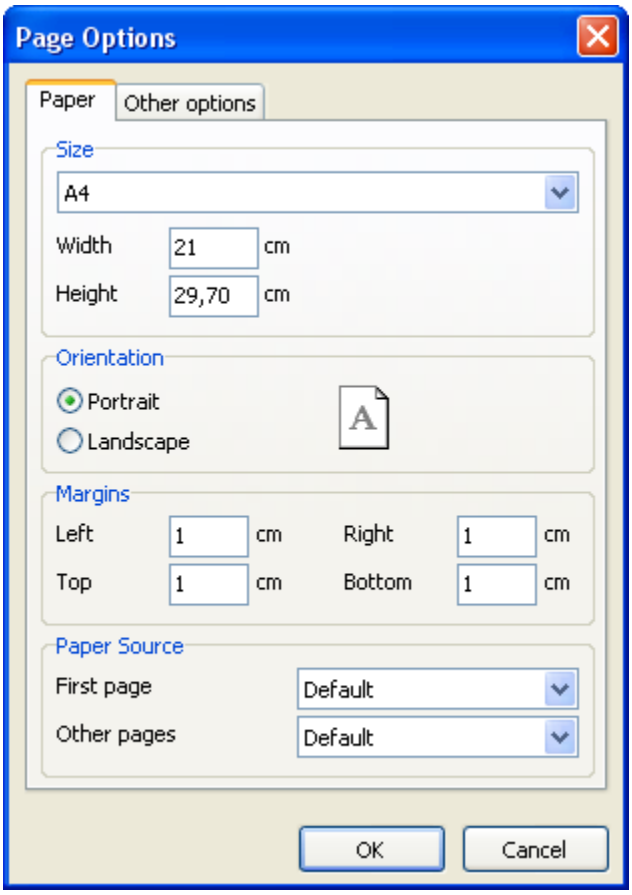
对话框的第三页允许您为报表设置描述性属性：



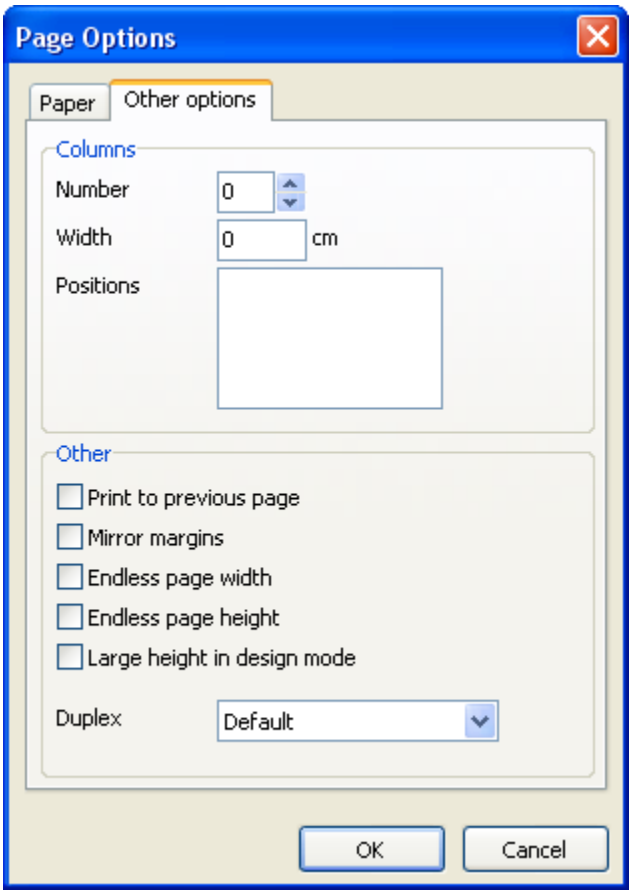
本页面上的所有字段仅供参考，不可打印。

1.6 页面选项

“页面选项”可通过“文件>页面设置...”菜单或双击页面上的空白区域获得。对话有两页：



在第一个对话页面上，您可以设置纸张的大小和对齐方式以及边距。在“纸张来源”中，下拉列表允许为第一页和其余报表页选择打印机托盘。



在第二个对话框页面上，您可以设置多列报表的列数。当前设置也会显示在设计器中。

“打印到上一页”标志允许您在不启动新页面的情况下打印页面，而是从上一页的任何空白处开始。当报表模板由多个页面组成或打印批量（复合）报表时，此选项非常有用。

“镜像边距”选项可在预览或打印报表期间切换偶数页面的左右边距。

“无限页面宽度和高度”选项会根据页面上的数据记录数（运行报表时）增加页面大小。设置后，报表将显示在预览窗口中的一个大页面上，而不是显示在多个标准尺寸页面上。

“设计模式下的大高度”选项会增加设计器中的页面高度。如果页面上有许多段，则此功能非常有用：在使用重叠段时必须使用此功能。它仅在设计模式下影响页面高度。

章节 2

创建报表

2.1 报表对象

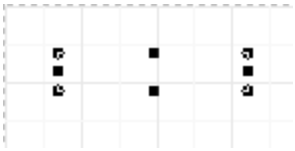
FastReport Designer将报表表示为原理图页面的集合。对象放置在报表页面的任何位置，用于定义报表的外观并显示各种信息，如文本和图形。标准包中包含的FastReport对象是：

/ - “Band”对象：设计页面上根据其类型行为的区域（例如，标题段，数据段）
- “文本”对象：在矩形区域内显示一行或多行文本
- “图片”对象：以“BMP”，“JPEG”，“ICO”，“WMF”或“EMF”格式显示图形文件
- “线”对象：显示水平或垂直线
- “系统文本”对象：显示系统信息（日期，时间，页码等）或聚合值
- “子报表”对象：允许在主机页面中插入另一个报表设计页面
- “绘制”类别对象：显示各种几何形状
（对角线，矩形，圆角矩形，椭圆，三角形和菱形）
- “图表”对象：以各种图表格式显示数据（饼图，直方图等）
- “RichText”对象：以RTF格式显示文本（RTF）
- “CheckBox”对象：显示带有刻度线或十字形的复选框
- “条形码”对象：将数据显示为多种条形码类型之一
- “OLE”对象：使用OLE技术显示任何对象。

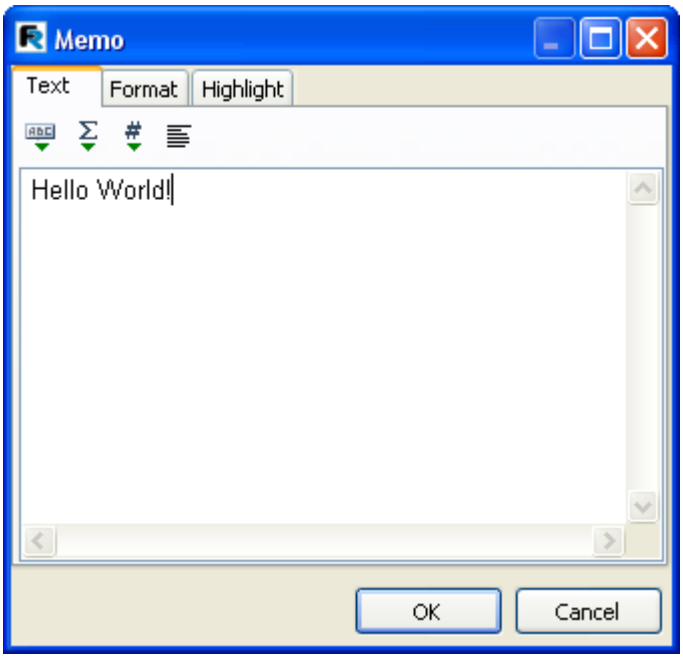
最常用的基本对象是“Band”和“Text”对象。您将在本章后面详细了解它们的功能。

2.2 “你好，世界！”报表的例子

示例报表将只包含一条信息：“Hello，World！”。打开报表设计器，然后单击“对象”工具栏中的“文本”按钮。将鼠标光标移到页面上，然后再次单击。将对象插入鼠标位置。



文本编辑器窗口将立即打开；如果它没有自动出现，则双击该对象。可以在设计器设置中配置文本编辑器打开 - 请参阅后面的内容。键入“Hello，World！”，然后单击ＯＫ按钮。

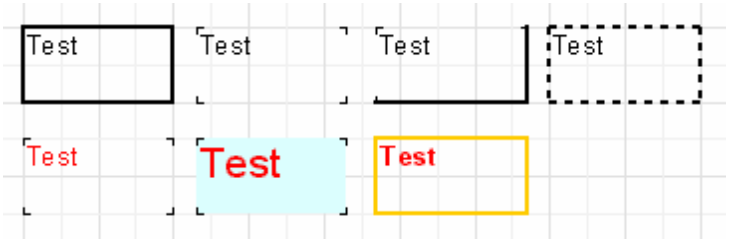


报表设计完成。要预览报表，请选择“文件>预览”菜单项或单击工具栏中的相应按钮。将出现包含带有“Hello，World！”文本的报表页面的预览窗口。此报表可以打印出来，保存到文件（*.fp3），或导出为支持的导出格式之一。

2.3 “文本”对象

“Text”对象具有许多功能。它可以在框架中显示文本并填充颜色。可以使用任何大小和样式的任何字体显示文本。可以借助工具栏直观地设置所有属性。

以下是一些文本设计示例：



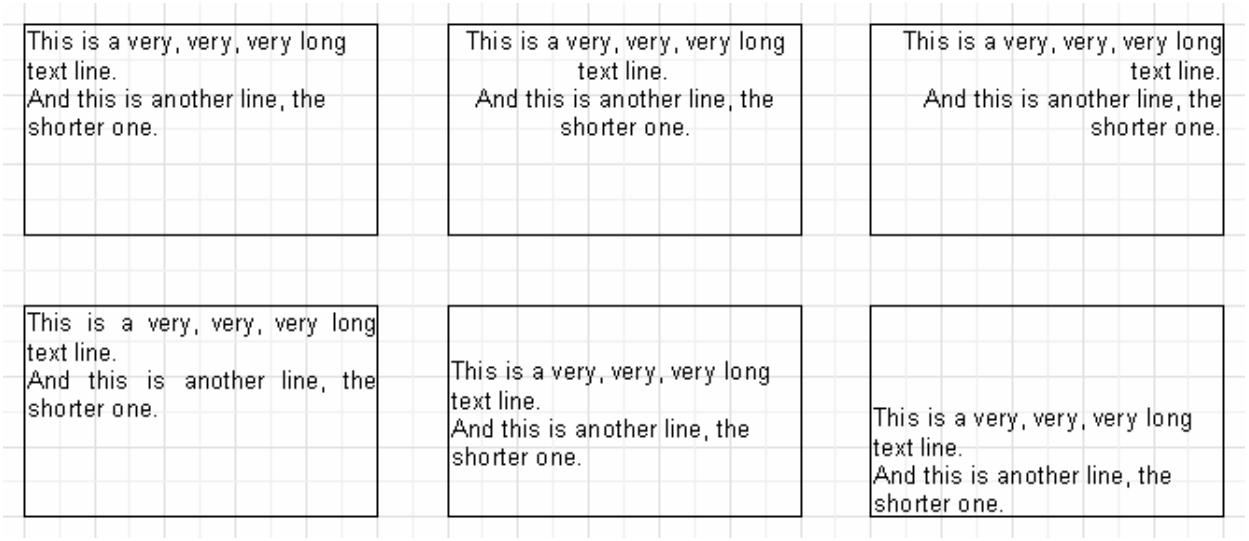
现在让我们看看这个基本对象的其他功能。例如，让我们创建一个新的“Text”对象来显示两行文本：

这是一个非常非常非常长的文本行。这是
另一条线，较短的线。

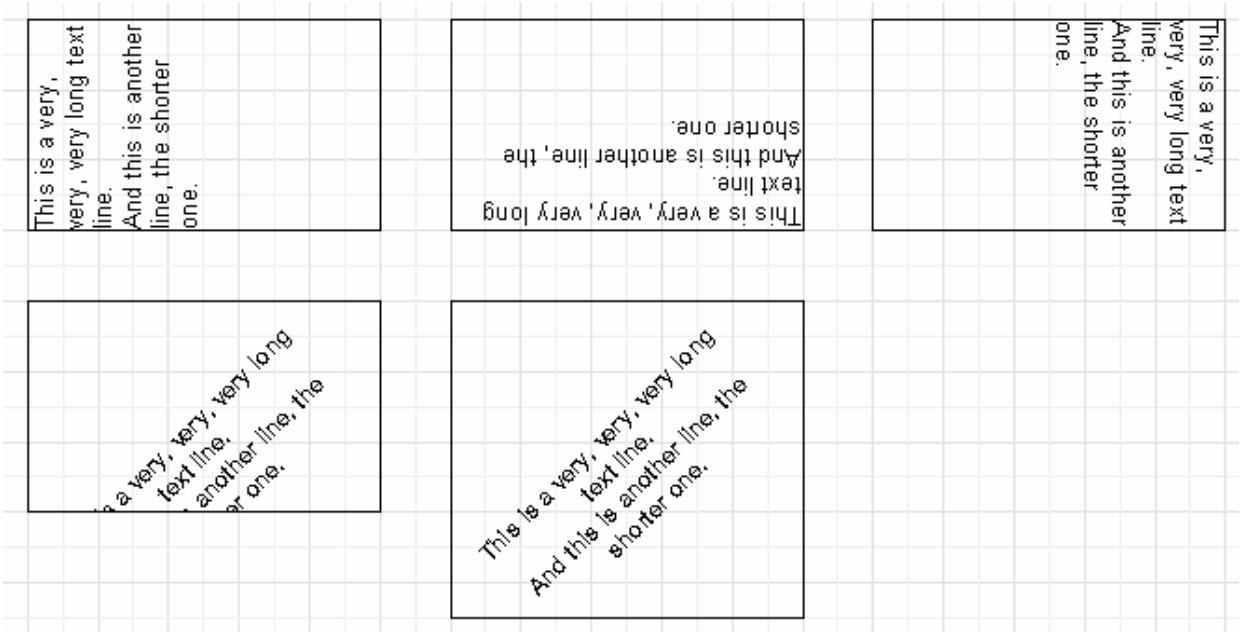
从工具栏启用对象框，并使用鼠标将对象重新调整为9x3 cm。我们看到该对象不仅可以显示单行，还可以显示多行文本。现在

将物体宽度减小到5厘米。很明显，长线不适合整个物体，因此被包裹。这由“WordWrap”对象属性控制。如果它被禁用（在对象检查器中或通过对象上下文菜单），任何长行都将被简短地缩短。

现在让我们看看对象内部的文本对齐是如何工作的。对齐按钮位于“文本”工具栏中，可控制水平或垂直文本对齐。请注意“Justify”按钮，它允许您将段落对齐到两个对象边缘 - 为此，必须启用“WordWrap”。



对象中的所有文本都可以旋转到0..360°范围内的任何角度。“文本”工具栏中的/按钮允许您快速将文本旋转到0,45,90,180或270°的预设。对于任何其他值，在对象检查器的“旋转”属性中设置所需的角度。当将文本旋转到90°，180°或270°以外的角度时，文本可能会被对象的框架截断，如下所示。为了解决这个问题，请稍微增加对象高度，以便所有文本都适合对象。



让我们简要地看一下影响其外观的其他“Text”对象属性。大多数这些属性仅在对象检查器中可用：

- BrushStyle：对象填充的类型
- CharSpacing：在字符之间进行拼写，以像素为单位
- GapX，GapY：左边和上边的文本缩进，以像素为单位
- LineSpacing：行之间的间距（以像素为单位）
- ParagraphGap：段落的第一行缩进，以像素为单位

2.4 “文本”对象中的HTML标记

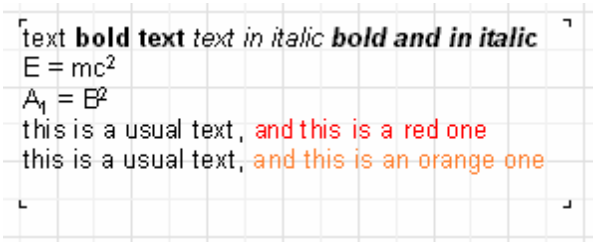
“Text”对象确实理解了一些简单的HTML标记。标签可以位于对象的文本中。默认情况下禁用标记，但要启用它们，请在对象上下文菜单中选择“允许HTML标记”，或在对象检查器中启用“AllowHTMLTags”属性。以下是支持的标记列表：

<b>：粗体文字
<i>：斜体文字
<u>：带下划线的文字
<sub>：下标
<sup>：上标
<font color>：字体颜色
<nowrap>：启用“WordWrap”时未分割的文本，整个文本将转移到下一行

请注意，只支持几个标签，但对于大多数应用程序来说这应该足够了。无法通过HTML标记修改字体大小或名称。

以下示例显示了如何使用这些标记。

```
text <b>粗体文字</ b><i>斜体文字</ i><b><i>粗体和斜体</ b></ i>
E = mc <sup> 2 </ sup>
A<sub>1</sub> = B<sup>2</sup>
这是一个通常的文字，<font color = red>，这是一个红色的</ font>
这是一个通常的文字，<font color =“#FF8030”>，这是一个橙色的</ font>
```



2.5 在“Text”对象的帮助下显示表达式

这个基本对象最重要的特性之一是它不仅能够显示静态文本，还能显示表达式。表达式可以位于与普通文本混合的对象内。让我们看一个如何工作的简单示例。在对象中键入以下内容：

你好，世界！今天是[日期]。

当报表运行时，我们可以得到这样的结果：

你好，世界！今天是01.01.2004。

这是怎么发生的？当FastReport创建报表并遇到括在方括号中的表达式时，报表引擎会计算表达式的值，并将此值插入到文本中以代替表达式。“文本”对象可以包含与普通文本混合的任意数量的表达式。复杂表达式可以包含括号（例如[1 + 2 * (3 + 4)]）。常量，变量，函数和DB字段都可以在表达式中使用。我们将在本章后面详细了解这些内容。

FastReport自动识别文本中方括号中的表达式。但是如果我们的普通文本包含方括号而我们不想被视为表达式会发生什么呢？例如，如果我们需要显示以下内容：

a[1] := 10

FastReport会将[1]视为表达式并将文本显示为：

a1 := 10

当然，这不是我们想要的。避免这种情况发生的一种方法是禁用表达式识别。禁用“AllowExpressions”属性（或上下文菜单中的“Allow Expressions”），将忽略文本中的所有表达式。在我们的示例中，FastReport将准确显示我们需要的内容：

a[1] := 10

但有时文本需要包含表达式和带方括号的普通文本，例如：

a [1]: = [myVar]

禁用“AllowExpressions”可以让我们在所需的位置显示方括号，但它也会禁用表达式的处理。在这种情况下，FastReport允许您使用另一组符号来指定表达式。默认情况下，“ExpressionDelimiters”属性“[,]”负责此操作。在我们的例子中，我们可以使用尖括号表示表达式而不是方形表达式：

a [1]: = <myVar>

必须在逗号所在的“ExpressionDelimiters”属性中设置“<, >”值

需要分开开始和结束符号。另一个要求是打开和关闭符号不能相同，因此“%，%”将不起作用。可以使用复杂符号，例如“<%，%>”。所以我们的例子看起来像这样：

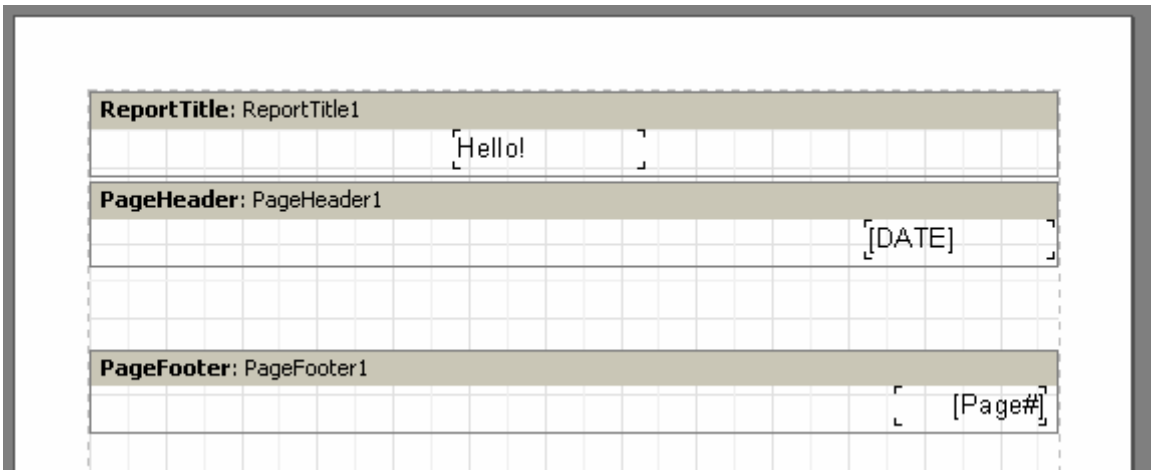
```
a [1]: = <%myVar%>
```

2.6 FastReport中的段

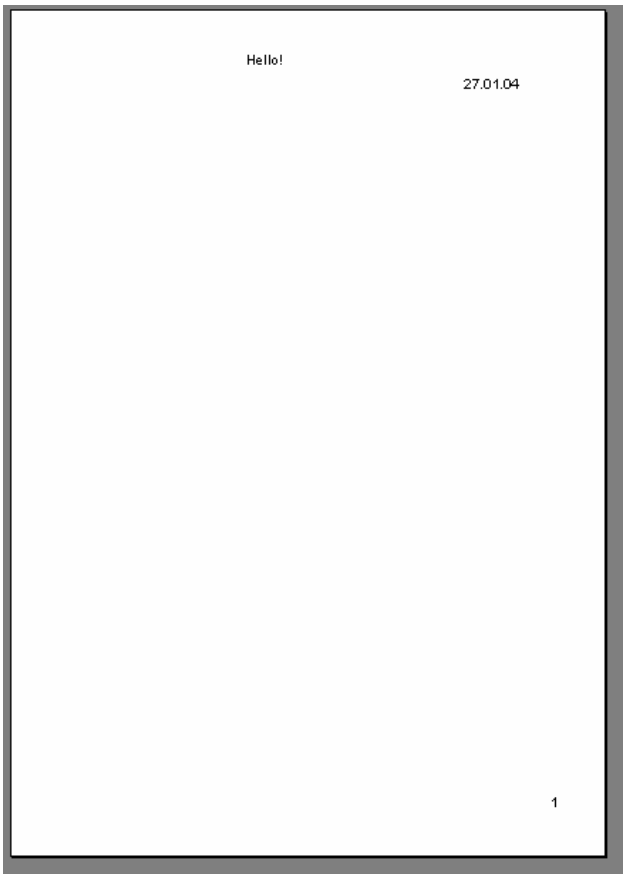
段用于将它们包含的对象放置在输出页面上的特定位置。将对象放在“PageHeader”区域时，我们告诉报表引擎必须在完成的报表中的每个页面的顶部显示给定对象。同样，“PageFooter”段中的对象显示在每个页面的底部。让我们用一个例子来证明这一点。我们将在页面顶部创建一个包含“Hello！”的报表，右侧的当前日期和右侧页面底部的页码。

打开FastReport设计器，然后单击工具栏中的“新建报表”按钮。您将看到一个已包含三个段的报表模板：“ReportTitle”，“MasterData”和“PageFooter”。让我们删除“MasterData”段一段时间（单击段内部或其标题上的任何空闲空间，然后使用“删除”键或上下文菜单中的“删除”将其删除）。现在让我们添加一个新的段（“PageHeader”）。单击对象工具栏上的“添加段”按钮，然后从下拉列表中选择“PageHeader”。我们看到页面中添加了一个新的段。同时现有的频段向下移动。 FastReport设计器自动在页面上定位段 - 顶部的标题段，中间的数据段和底部的页脚段。

现在让我们添加一些对象。将“系统文本”对象添加到“PageHeader”区域，并在其编辑器中从“系统变量”下拉列表中选择“[日期]”（您应该记住，当前日期也可以通过“文本”对象显示在其编辑器中输入“[DATE]”。接下来，在“ReportTitle”段中添加一个包含“Hello！”的“Text”对象。显示页码的“文本”对象已自动添加到“PageFooter”频段。



运行报表时，您将看到完成报表中的对象打印在页面上的正确位置。



因此，段负责在页面上定位对象。根据段的类型，我们可以在页面的顶部或底部以及第一页或最后一页上打印对象。大多数报表所需的基本频段如下：

- “PageHeader”频段：显示在每个页面的最顶部
- “PageFooter”段：显示在每页的最底部
- “ReportTitle”频段：显示在第一页的顶部 – 这可以在“PageHeader”频段之前或之后，具体取决于页面的“TitleBeforeHeader”属性（在点击页面上的任何可用空间后在对象检查器中找到）
- “ReportSummary”段：显示在报表最后的自由空间中

2.7 数据段

现在我们将学习如何从数据库表或查询中打印数据。什么被认为是'表'或'查询'？它们由组织成行（记录或行）的数据组成，这些行包含一个或多个列（字段）。要打印这种数据，FastReport使用一种特殊类型的段，各种段名为“...数据”。要打印整个表或仅打印某些行和字段，必须将一个或多个这些段添加到报表中，将它们连接到表并在其中放置字段对象。当FastReport构建报表时，段将打印在输出页面上，对表中的每个记录打印一次。如果某个段的输出页面上没有剩余可用空间，则报表引擎将在继续之前创建一个新的输出页面。

2.8 TfrxDBDataSet组件

Delphi FastReport组件面板中的“TfrxDBDataSet”连接器组件用于将表（或任何其他数据源）连接到FastReport。此组件充当数据源和FastReport核心之间的信使。该组件负责记录导航和现场参考。这允许FastReport核心独立于任何数据访问库。 FastReport可以同时使用“BDE”，“IB_Objects”（具有非标准实现，与TDataSet不兼容）和其他库，以及来自未连接到数据库的源的数据，例如数组或文件。“TfrxDBDataSet”组件用于处理与TDataSet兼容的数据源（例如BDE，ADO，IBX和大多数其他库）。“TfrxIBODataSet”组件用于处理IB_Objects。“TfrxUserDataSet”组件/与其他数据源（数组，文件等）一起使用。

使用“TfrxDBDataSet”组件非常容易。通过设置“DataSet”属性（用于直接连接到表或查询）或“DataSource”属性（用于通过“TDataSource”组件进行连接）将其连接到数据源。这两种方法是等效的并且是互斥的，尽管第一种方法允许数据管理而不需要“TDataSource”组件。

要使报表可以使用Delphi组件（及其连接的数据），必须启用报表中使用的任何数据集。通过FastReport设计器中的“Report> Data ...”菜单项执行此操作，在打开的对话框中选择所需的数据集。



2.9 “客户列表”报表

我们的第二份报表将比第一份报表复杂得多（它将包含公司客户的数据库表）。要生成此报表，请使用Delphi分发工具包中包含的演示数据库DBDEMOS。在Delphi中创建一个新项目，在表单上放置一个“TTable”组件并设置它的一些属性：

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

要使表的数据可用于FastReport，请在表单中添加“TfrxDBDataSet”组件，然后设置一个属性：

数据= table1

最后，在表单中添加“TfrxReport”（FastReport的基本组件），双击它以打开FastReport设计器，然后单击“新建报表”按钮，自动创建一个带有三个段的基本报表设计（“报表标题”，“”主数据“和”页脚“）。为了使我们的表可用，我们必须在FastReport中启用它。通过单击“报表>数据...”菜单项，检查frxDBDataset1（它是当前列出的唯一数据集）并单击“确定”来执行此操作。对话框关闭后，DBDataset及其连接的表字段将在IDE的“数据树”窗格中显示。
现在让我们设计报表。首先，将包含“客户列表”的“文本”对象添加到“ReportTitle”频段。

接下来，将“MasterData”段连接到我们的数据集。这可以通过以下三种方式之一完成：

- 双击段
- 在段的上下文菜单中选择“编辑...”
- 或者在对象检查器中单击段的“DataSet”属性

现在在段上放置四个“文本”对象（将显示数据集中的客户端编号，客户名称，电话和传真字段）。让我们用几种不同的方式来演示FastReport设计器的一些功能。

第一种方法是在段上放置一个“Text”对象，并在其中键入“[frxDBDataSet1。”CustNo“]”。这是最不方便的方法，因为必须手动输入字段链接，可能会错误地键入文本。使用表达式设计器稍微容易一点 - 双击“Text”对象，然后单击打开的编辑器窗口的“Text”选项卡上最左边的按钮。要插入我们的字段，请双击表达式设计器中的字段名称。单击“确定”按钮关闭对话框，然后查看插入“文本”对象的字段链接。

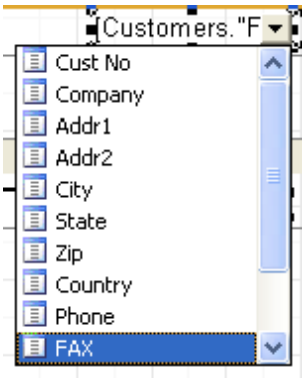
将DB字段添加到“Text”对象的第二种方法是在对象检查器中设置两个属性。在段上放置第二个“Text”对象，而不在编辑器窗口中写入任何内容。使用对象检查器设置对象的属性：

DataSet = frxDBDataSet1
DataField = 'Company'

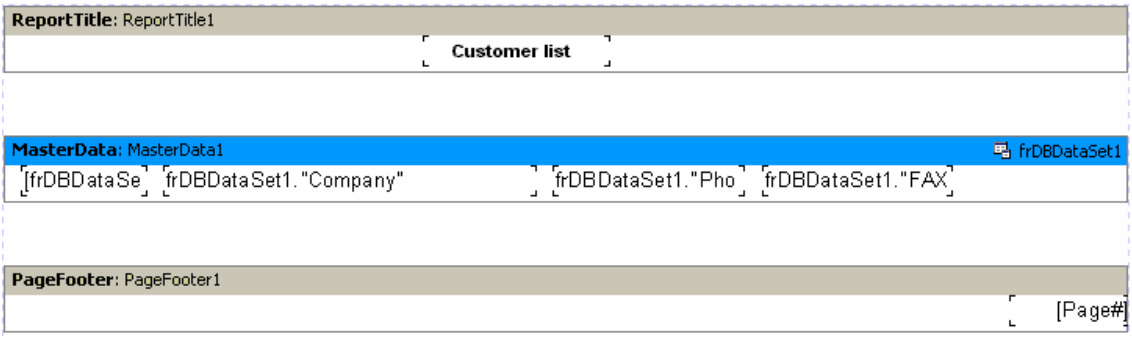
这两个属性都显示为列表，因此只需使用鼠标从下拉列表中选择所需的值。

第三种方法是将所需字段从“数据树”窗格“拖放”到报表中。这是最简单和最简单的方法，但在执行此操作之前，请取消选中“数据树”窗格底部的“创建标题”复选框，否则将创建除字段外的第二个“文本”对象，其中包含字段的名称链接对象本身。以这种方式选择“电话”字段并将其拖到频段上。

第四种方法需要设置设计器选项“显示下拉字段列表”标志（通过主菜单：“视图>选项”）和已连接到数据集的频段。在带上放置一个空白的“文本”对象，将光标悬停在对象上，右下角会出现一个下拉按钮。单击此按钮可打开数据库字段列表，然后选择“传真”字段。



我们的报表设计已经完成。



单击“预览”按钮以查看结果。



Customer list

1221	Kauai Dive Shoppe	808-555-0269	808-555-0278
1231	Unisco	809-555-3915	809-555-4958
1351	Sight Diver	357-6-876708	357-6-870943
1354	Cayman Divers World Unlimited	011-5-697044	011-5-697064
1356	Tom Sawyer Diving Centre	504-798-3022	504-798-7772
1380	Blue Jack Aqua Center	401-609-7623	401-609-9403
1384	VIP Divers Club	809-453-5976	809-453-5932
1510	Ocean Paradise	808-555-8231	808-555-8450
1513	Fantastique Aquatica	057-1-773434	057-1-773421
1551	Marmot Divers Club	416-698-0399	426-698-0399
1560	The Depth Charge	800-555-3798	800-555-0353
1563	Blue Sports	610-772-6704	610-772-6898
1624	Makai SCUBA Club	317-649-9098	317-649-6787
1645	Action Club	813-870-0239	813-870-0282
1651	Jamaica SCUBA Centre	011-3-697043	011-3-697043
1680	Island Finders	713-423-5675	713-423-5676

2.10 借助“Text”对象显示DB字段

如您所见，“Text”对象可以显示来自DB的数据以及静态文本和表达式。我们还看到它可以通过两种方式完成：将DB字段的链接放入对象文本，或者通过“DataSet”和“DataField”属性将对象连接到必需字段。第一种方法适用于在同一对象中显示字段内容和限定文本。例如：

联系人：[frxDBDataSet1。“Contact_Person”]

链接到DB字段需要特殊语法：[datasetname。“fieldname”]。字段名称（以及数据集名称）可以包含空格，但“点”和“引号”符号之间不能有任何空格。

此外，我们还可以将计算操作应用于表达式中的字段，如下所示：

*长度（cm）：[<frxDBDataSet1。“Length_in”> * 2.54]*

请注意必须使用方括号和尖括号。请记住，默认情况下使用方括号来分隔对象文本中的表达式。如果需要，方括号可以替换为任何其他打开/关闭字符对（请参阅“在”文本“对象的帮助下显示表达式”部分）。在表达式中使用尖括号来描绘FastReport变量或DB字段。逻辑上我们应该写：

联系人：[<frxDBDataSet1。“Contact_Person”>]

代替

联系人：[frxDBDataSet1。“Contact_Person”]

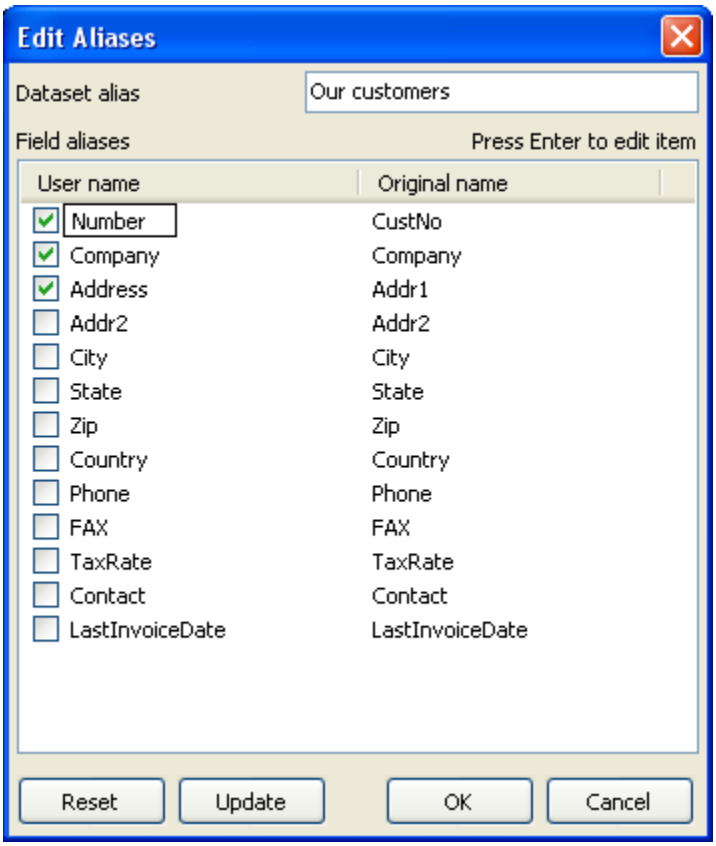
但是，这两种表示法都是正确的，因为FastReport不需要尖括号，其中表达式只包含变量或仅包含DB字段。在所有其他情况下，角括号是必需的，如：

长度（cm）：[〈frxDBDataSet1。“Length_in”〉＊2.54]

2.11 别名

在上一个报表中，数据源名为frxDBDataSet1，字段名为“CustNo”，“Company”，“Phone”和“FAX”，我们在报表中使用类似“[frxDBDataSet1。”CustNo“”的内容引用它们。“这是否易于理解？不是真的。如果数据源和字段分别命名为”我们的客户“和”数字“，那就更清楚了。有一个问题，”frxDBDataSet1“是组件的名称，不支持空格的“CustNo”是数据库中字段的名称，如果没有数据库重组，则无法更改。但是有一种解决方法。我们可以对这些项使用假名或别名。在FastReport中，数据集及其字段拥有称为UserName的第二个名称属性，即别名，可以轻松更改。每当组件被赋予别名（即UserName从其默认值更改）时，此别名必须在FastReport中使用 - 组件的名称属性不是r ecognized。

在FastReport中为数据源及其字段设置别名非常容易。它是在Delphi环境中完成的。要打开别名编辑器，请双击frxDBDataSet1组件或使用其上下文菜单。您可以在编辑器中修改数据集和字段别名，还可以指定报表中需要哪些字段。让我们重命名数据集和字段，如下所示：



也可以在不使用别名编辑器的情况下修改数据集的别名，从而更改frxDBDataSet1组件的“UserName”属性。

完成此操作后，我们现在需要修改报表，因为数据集和字段的名称已更改。要修改报表对象中的字段名称，最简单的方法是使用“客户列表”报表“部分中描述的第四种方法。将鼠标光标移到“文本”对象上，以便显示对象右端的按钮，单击按钮并从列表中选择字段。在此之后，数据集及其字段中数据的来源更加明显。

注意：最好在报表设计的最开始分配别名，以避免以后重命名报表中的所有字段。

2.12 变量

除了别名之外，还有另一种方法可以帮助报表设计者为数据库字段和其他信息元素设置更易理解的名称。 DB字段名称或任何表达式可以与变量相关联。要在FastReport中创建和使用变量，请选择“报表/变量...”菜单项。

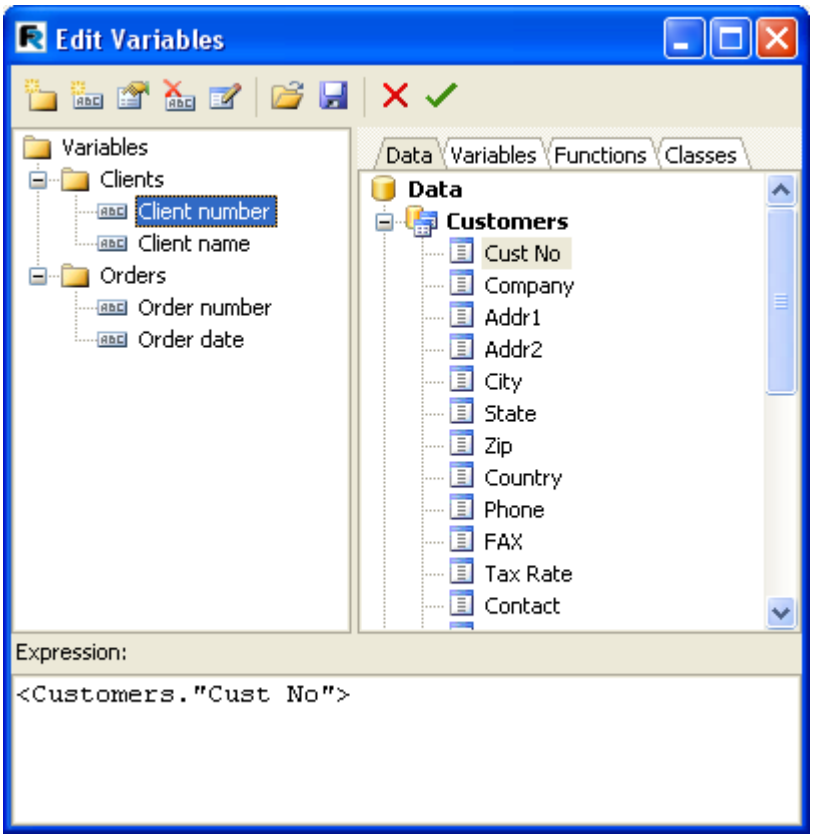
FastReport中的变量列表具有两级结构。第一级只包含类别，第二级包含变量本身。当变量列表很长时，这种变量分类很方便。变量列表必须至少包含一个

category，表示变量不能位于顶层。此外，类别名称不能包含在报表中，因此列表中的每个名称都必须是唯一的。

让我们用以下示例说明变量的使用。假设我们有两个数据源：第一个是带有“CustNo”和“Name”字段的“frxDBDataSet1”，第二个是带有“OrderNo”和“Date”字段的“frxDBDataSet2”。我们可以将以下变量列表与字段相关联：

客户端
客户编号客户名
称订单
订单号订购日
期

“客户”和“订单”分为两类。打开变量编辑器并使用“新类别”，“新变量”和“编辑”按钮创建此结构。要将变量链接到数据库字段，请选择一个变量，然后双击右侧窗格中所需的数据库字段。DB窗口的链接将显示在底部窗格中。变量现在与此表达式相关联，因此变量的值将成为表达式的值。如有必要，可以手动编辑或修改表达式，并且可以在其中使用任何FR函数或其他变量。请记住，类别不得与任何表达式相关联。



创建变量列表后，关闭变量编辑器。现在我们可以将变量插入到报表中。与DB字段相比，插入方法更少。我们可以通过输入“[Client number]”手动将变量插入到对象文本中，或者我们可以拖动一个

从“数据树”窗格到报表页面的变量，在这种情况下切换到此窗格中的“变量”选项卡。

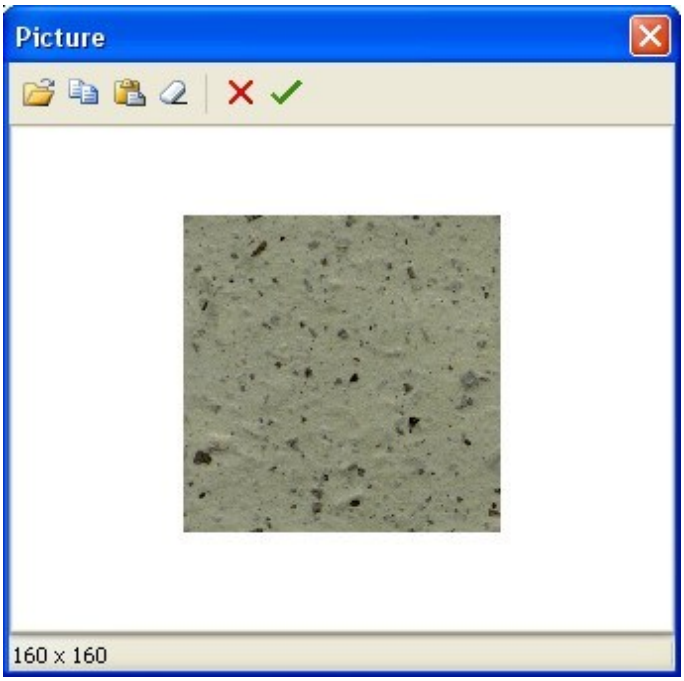
让我们回顾一下到目前为止我们学到了什么

- 报表设计由设计页面组成
- 页面可能包含FR对象，可放置在页面上或带内
- 段是设计页面上的占位符，并根据段类型控制它们包含的对象在输出页面上的位置
- “Text”对象包含我们想要在给定位置输出的文本，它们可以是多行的，可以包含静态文本，数据字段，变量，表达式或所有这些的组合
- 数据类型段（Master，Detail，Subdetail等）连接到TfrxDbDataset时，控制这些段出现的次数（行），并与报表引擎一起确定报表输出的已完成页数。注意：即使这些数据段可能具有Master，Detail等类型，但这只是输出页面上段分层位置的位置关系。实际的数据关系取决于连接的frxdbdatasets中的表/查询关系。每个数据段级别都需要单独的“TfrxDbDataset”或等效关联。

2.13 “图片”对象

下一个要查看的对象是“图片”对象。它也经常用于报表中。使用此对象，您可以插入徽标，照片（例如员工的照片）或任何其他图片信息。该对象可以以“BMP”，“JPEG”，“ICO”，“WMF”或“EMF”格式显示图形。

我们来看看这个对象的功能。创建一个空白报表并在报表页面上放置“图片”对象。如果图片编辑器未自动打开，请双击“图片”对象。您可以从图形文件加载任何图片或清除现有图片，然后单击绿色勾选关闭编辑器。



上下文菜单以及对象检查器中提供了一些对象的属性：

- 自动尺寸
- 拉伸：默认启用
- 中央
- KeepAspectRatio：默认启用

启用“自动调整大小”时，对象的大小将调整为其包含的图片大小。有时，如果显示不同大小的数据库图片，这可能很有用。默认情况下禁用“AutoSize”。

默认情况下启用“拉伸”。此属性可拉伸对象内的图片。使用鼠标更改对象的大小，您将看到整个图片始终显示在对象的框架内。如果禁用“拉伸”，则图片（或仅部分图片）将以其原始大小显示。此行为与“AutoSize”的不同之处在于您可以控制对象的帧大小，该大小可以大于或小于其包含的图片。启用“AutoSize”后，您无法控制对象的帧大小。

“Center”属性对齐对象内的图片。

默认情况下启用“KeepAspectRatio”：当对象的大小发生变化时，它会使图片不会扭曲。此属性仅在启用“拉伸”时有效。启用“KeepAspectRatio”后，无论“图片”对象的大小如何，绘制的圆圈都会保持圆形而不会变成椭圆形。拉伸的图片仅占据对象内部空间的那部分，以正确的宽高比显示它。禁用时，图片将被拉伸以填充对象的帧，如果帧的宽高比与图片的宽高比不同，则会失真。

仅在对象检查器中可用的另一个有用属性是“FileLink”。输入文件名（例如c：\ picture.bmp）或包含文件名的变量（例如[picture_file]）将在运行报表时从指定文件加载图片。

2.14 用图片报道

与“快速报表”中的许多对象一样，“图片”对象可以显示来自数据库的数据。通过在对象检查器中设置“DataSet”和“DataField”属性，可以将此对象连接到DB字段。与“Text”对象相比，这是将此对象连接到其数据的唯一方法。

让我们用一份报表来证明这一点，它给出了一些鱼的名字和图像。我们将再次需要Delphi分发工具包中包含的“DBDEMOS”数据库。

在Delphi中创建一个空白项目，在表单上放置一个“TTable”组件并设置一些属性：

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

要在FastReport中使用此表，请添加“TfrxDBDataSet”组件并设置以下属性：

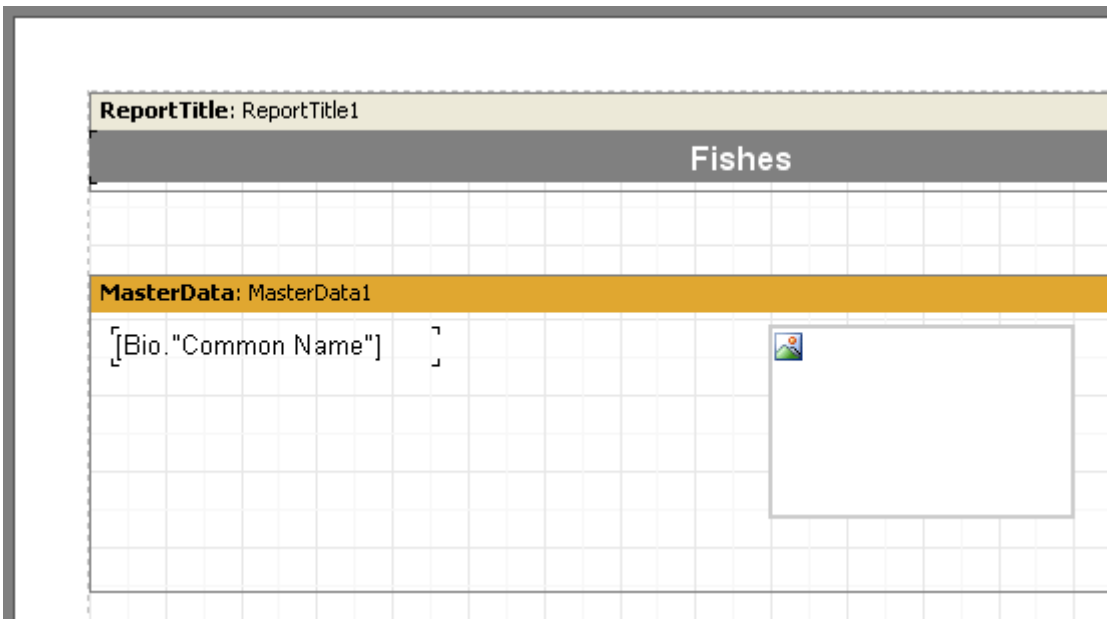
DataSet = Table1
UserName = 'Bio'

最后，在表单中添加“TfrxReport”组件。打开报表设计器并单击“新建报表”按钮以在FastReport中创建基本设计。现在启用报表中frxDBDataset及其连接表的使用 - 从菜单中选择“报表>数据”，选择“生物”数据集并单击“确定”。

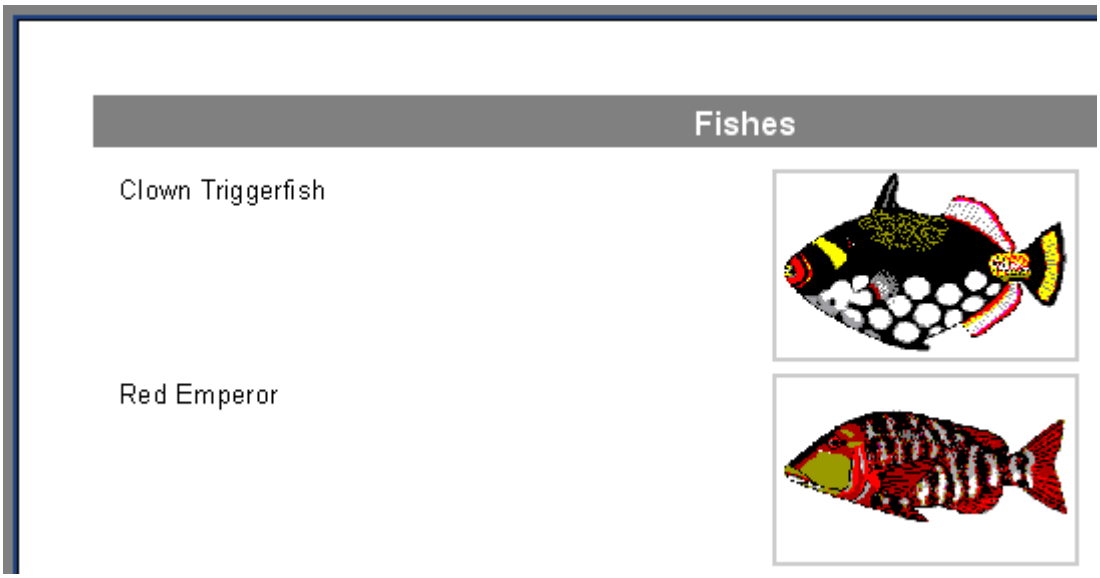
现在我们将向报表页面添加对象。在“ReportTitle”段中放置一个包含“Fish”的“Text”对象。将“主数据”段连接到数据源（双击段并从列表中选择“生物”）。通过向下拖动条段底部或使用对象检查器将条段高度增加到5厘米。在段中放置一个“Text”对象，并使用前面描述的任何方法将其连接到“CommonName”字段。之后，将“图片”对象放在旁边，并通过在对象检查器中设置其属性将其连接到“图形”字段：

DataSet = Bio
DataField = 'Graphic'

请注意，这两个属性都属于“List”类型，可以使用鼠标将其设置为所需的值。要为图片腾出空间，请将物体拉伸至4 x 2.5厘米。



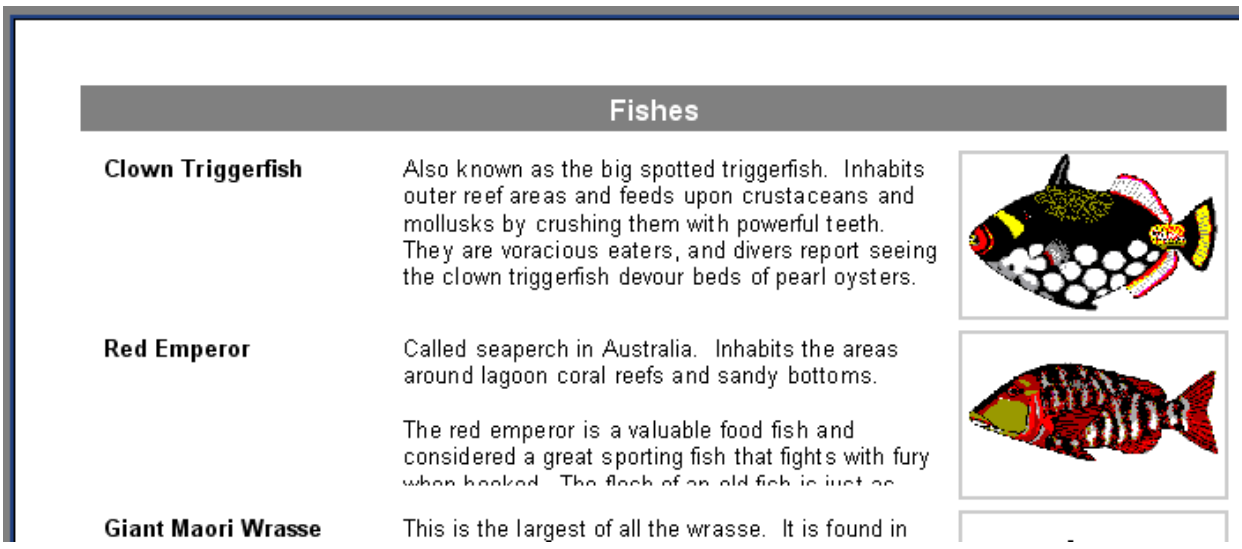
报表设计现已完成，并生成此处显示的报表：



2.15 多行文字显示

我们将改进前面的例子“Biolife”表有一个“注释”字段，其中包含每条鱼的详细描述。通过向其添加此字段来更新我们的报表。

乍一看，这看起来很简单 - 将“Text”对象添加到现有对象之间的数据段，将其连接到“Notes”字段，并将对象的大小设置为8 x 2.5 cm。但是，报表预览并不完全符合我们的要求：



FastReport完全按照指示执行。“注释”字段包含不同长度的多行文本，但显示该字段信息的“文本”对象具有固定大小。这就是为什么有些线条会被切掉的原因。关于这个还能做什么？

当然，可以增加对象的大小或者可以减小其字体大小。

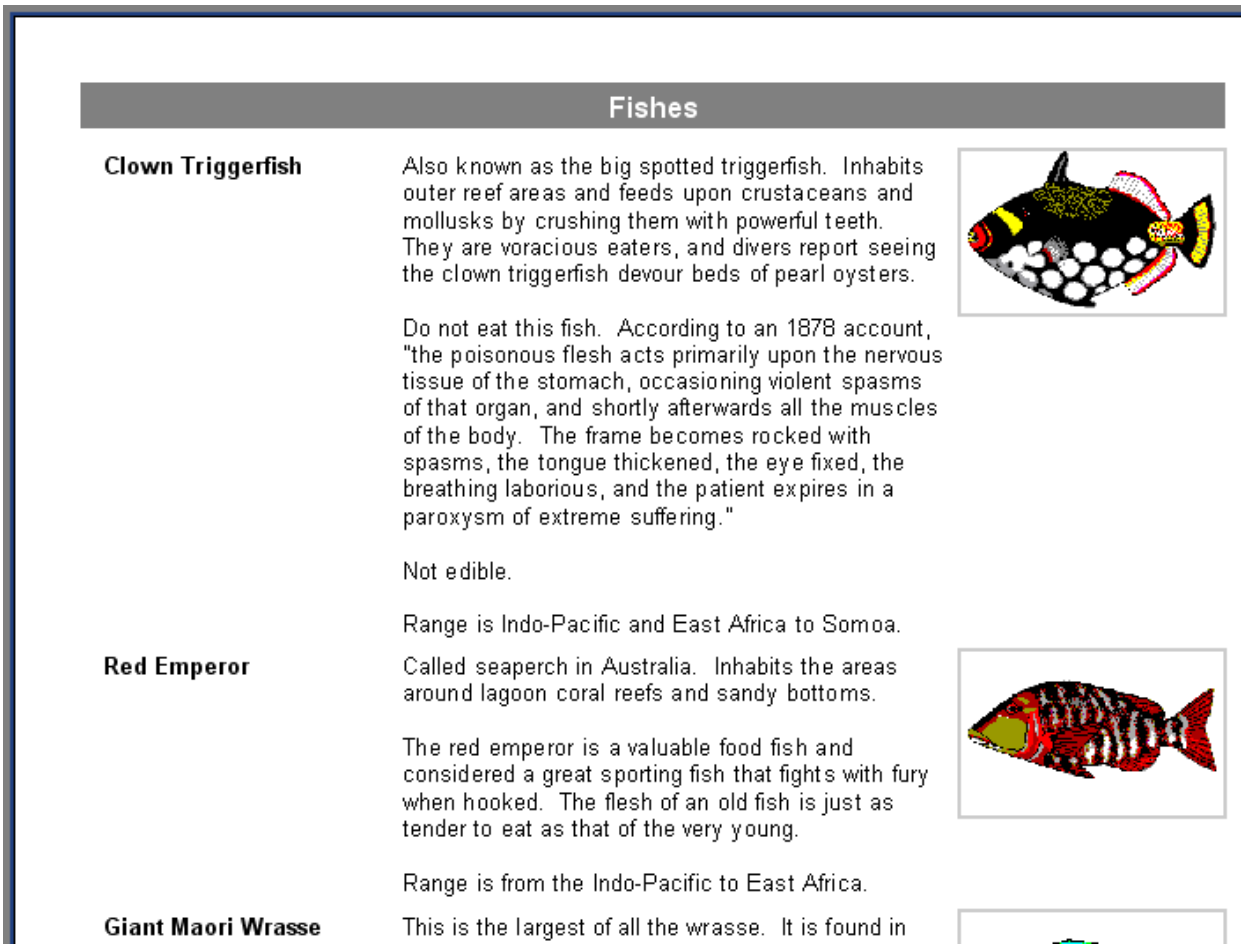
然而，这可能导致输出页面上的空间浪费，因为一些鱼有很长的描述，而其他鱼有较短的描述。FastReport有一些属性可以让我们解决这个问题。

这些属性允许段或对象自动调整其高度，以便为给定记录（行）创建必要的空间。为此，我们只需要启用band和“Text”对象的“Stretch”属性。但是，这并不是全部，因为具有较长文本的“Text”对象应该能够自行伸展，我们还需要设置其他一些属性。

“Text”对象可以自动设置其高度和宽度，以便为其内容腾出空间。“AutoWidth”和“StretchMode”可用于此目的。“AutoWidth”允许“Text”对象改变其宽度，以便所有行都可以创建空格而不会分割任何单词。当对象具有单个文本行并且向右增长不会影响其他对象时，此模式很有用。“Stretch”属性允许对象的高度增长以适应文本而不更改对象的宽度。“拉伸”有几种模式可以在对象检查器中选择：

smDontStretch – 不要拉伸对象（默认）
smActualHeight – 拉伸对象，使其为整个文本腾出空间
smMaxHeight – 拉伸物体，使其底部到达放置它的段的底部 - 我们稍后会看到这个模式

这里我们对“Text”对象的“Stretch”属性感兴趣。使用对象的上下文菜单或将“StretchMode”设置为smActualHeight启用它。同时启用段的“Stretch”属性。预览报表并确保一切都按预期进行。

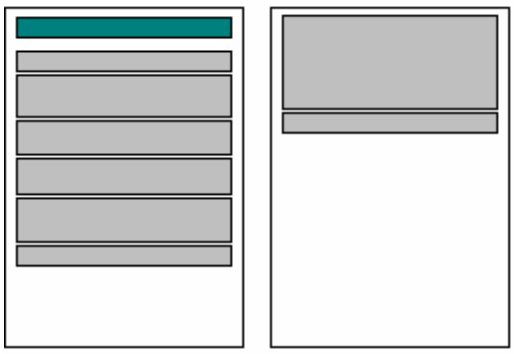


如您所见，在构造报表时，FastReport使用数据填充对象，并在启用“拉伸”时拉伸它们。然后计算段的高度，使每个物体都有足够的空间。如果禁用了段的“拉伸”属性，则不会执行此高度调整，段将以设计中设置的高度显示。在这种情况下，我们会看到具有较长文本的对象仍将被拉伸但是段不会被拉伸，这会导致文本重叠，因为后一个段在前一个段之后立即显示。

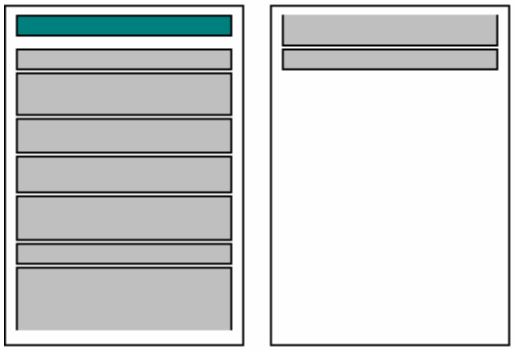
2.16 数据分割

让我们来看看这份报表的特点：页面底部有很多空白区域。为什么是这样？创建报表时，FastReport引擎会使用段填充页面的空白区域。显示每个频段后，当前位置向下移动。当FastReport发现没有足够的空白区域来显示下一个段（其高度大于页面上剩余的空白区域）时，FastReport会创建一个新页面并从那里继续进行段显示。此序列与数据集中的记录一样多次。

我们的报表包含一个大文本的对象，这就是段高度非常大的原因。此外，如果一个大的段在页面上没有足够的空间，它会转移到下一个空间，在页面底部留下大量未使用的空间，如下所示：



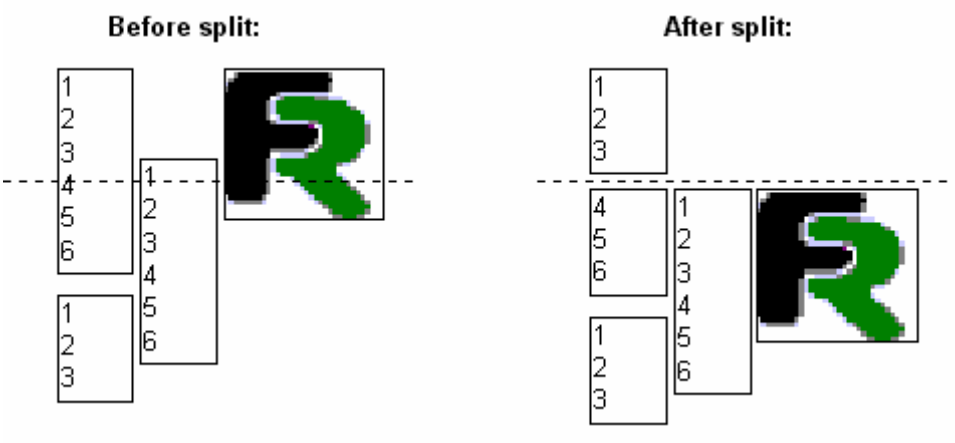
为了限制纸张浪费，让我们使用FastReport功能，该功能可以从段的内容中创建段落。我们所需要做的就是启用“First level data”频段的“AllowSplit”属性。您将看到报表页面底部的空白现在减少了：



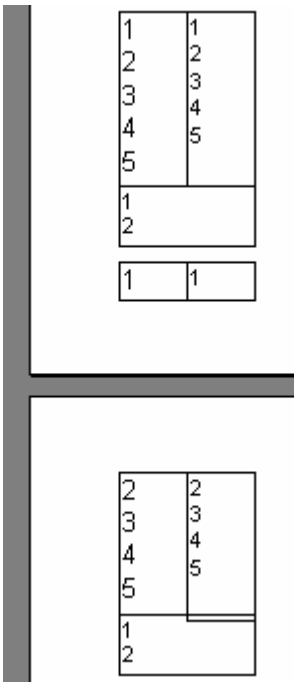
这个段分裂是如何工作的？ FastReport中有一些对象支持此功能。它们是“Text”，“Line”和“RichEdit”对象。它们可以“分裂”，而其他物体则不能。当FastReport需要拆分一个段时，它会按以下方式执行：

- 显示在空白区域中具有空间的不可拆分对象
- 部分显示可分割对象（“文本”对象以所有行在对象中都有空间的方式显示）
- 创建一个新页面并继续在段中显示对象
- 如果不可拆分的对象在空白区域没有空间，则转移到下一页；同时，位于其下方的所有对象都根据需要移动
- 该过程继续，直到完全显示所有段对象。通过查看此示例，拆分算法

将变得更加清晰：



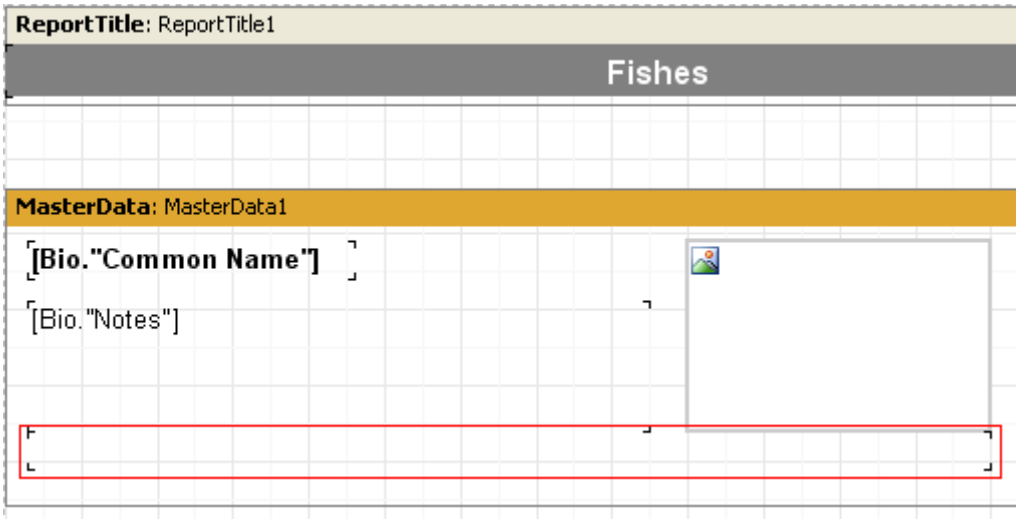
应该注意的是，分裂算法并不完美，最终输出可能不如预期的那样。如果拆分段上的对象以复杂的方式分组，和/或它们的字体大小不同，则应非常小心地使用此选项。以下是可生成内容的示例：



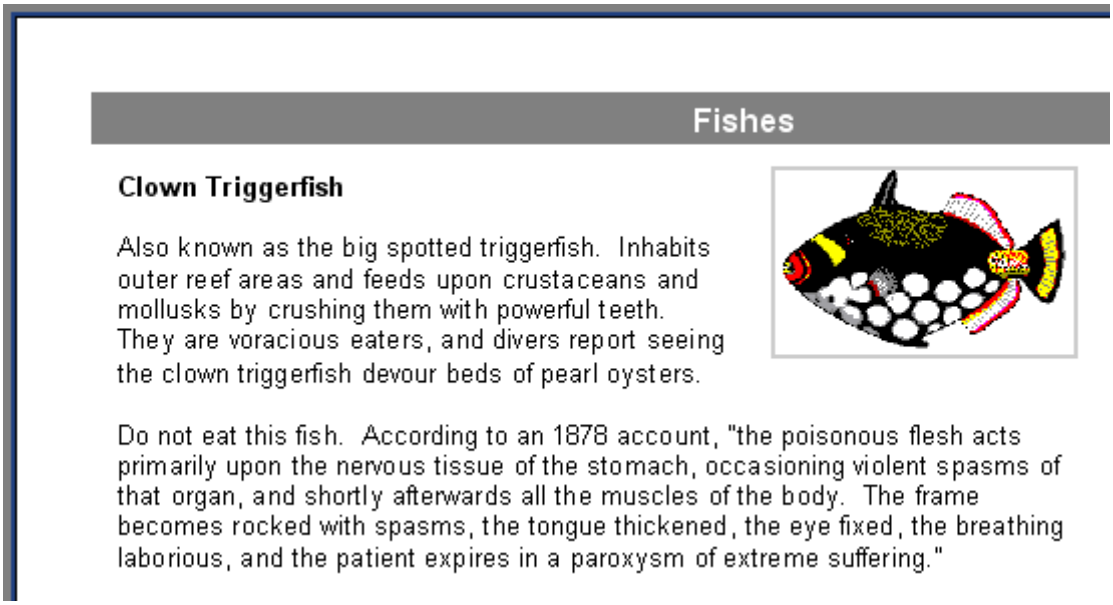
2.17 对象的文本换行

在某些报表设计中，可能需要文本环绕其他对象（通常在使用图片时）。让我们演示FastReport如何在我们当前的例子中做到这一点。

将另外一个“Text”对象添加到Bio。“Notes”对象下面的数据段中，如下所示：



我们将禁用Bio。“Notes”对象的拉伸并为底部对象启用它。要使文本从Bio。“Notes”对象“流动”到底部，请在对象检查器中设置Bio。“Notes”对象的“FlowTo”属性 - 它是一个下拉列表。从此列表中选择底部对象的名称。生成的报表如下所示：



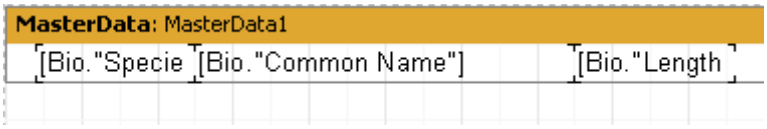
生成报表时，如果文本不适合顶部对象，则多余部分将移动到底部对象。通过在图片周围布置两个对象，实现文本包装的效果。

请注意：在插入链接的主对象之前必须将主对象插入到报表中，否则文本流可能无法正常工作！如果发生这种情况，请选择链接对象并使用“编辑>置于前面”菜单项将其置于前面。

2.18 以表格的形式显示数据

有时需要以框架表的形式显示数据。此类报表的示例可能是价目表。要在FastReport中创建此类报表，只需要为位于数据段中的对象启用帧。让我们通过一个例子演示几种框架的变体。

创建类似于此的报表：



将“文本”对象并排放置在条段上，并最小化条段的高度。

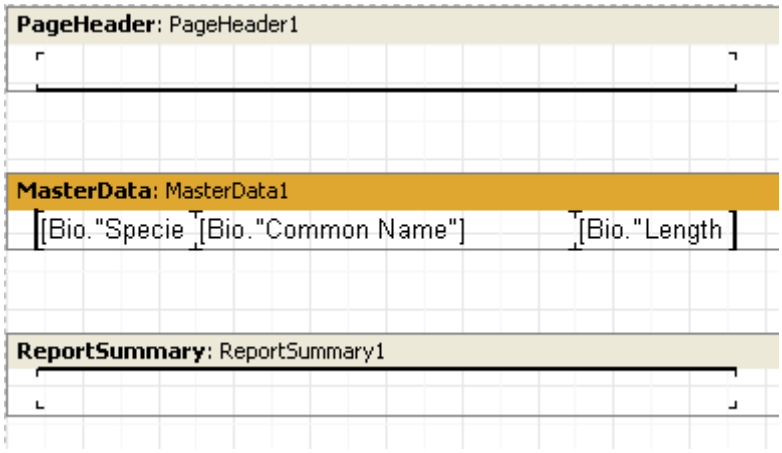
第一种和最简单的表格具有完整的单元格框架。要实现此目的，请为每个对象启用所有框架线（在Frame.Typ属性中找到）：

90020	Clown Triggerfish	50
90030	Red Emperor	60
90050	Giant Maori Wrasse	229
90070	Blue Angelfish	30
90080	Lunartail Rockcod	80

下一种类型的框架仅绘制水平或仅垂直的单元格行，同样通过Frame.Typ属性：

90020	Clown Triggerfish	50
90030	Red Emperor	60
90050	Giant Maori Wrasse	229
90070	Blue Angelfish	30
90080	Lunartail Rockcod	80

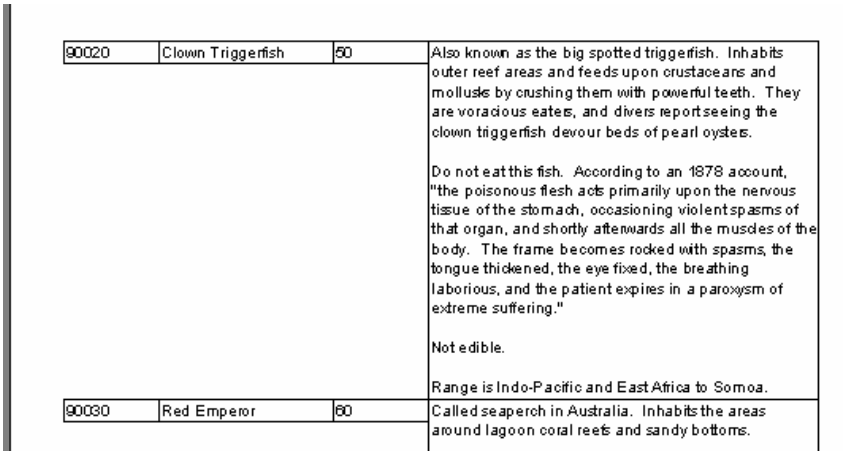
最后，要仅向表中绘制外部框架，报表需要进行修改：



你可以看到我们添加了两个“Text”对象，一个在pageheader中，另一个在pagefooter band中。已启用沿数据段边缘的对象的相应框架线，导致报表如下所示：

90020	Clown Triggerfish	50
90030	Red Emperor	60
90050	Giant Maori Wrasse	229
90070	Blue Angelfish	30
90080	Lunartail Rockcod	80
90090	Firefish	38

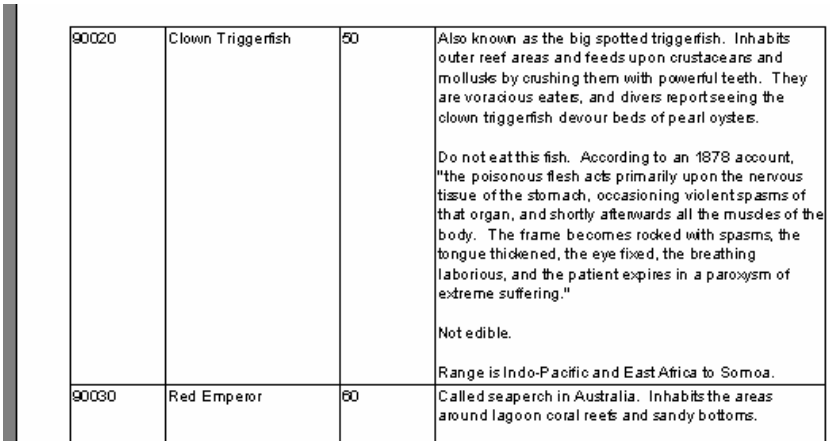
所有这些例子都包含具有固定尺寸的段。当段拉伸时如何显示表格？让我们通过一个例子来解释。在我们的报表中添加一个新字段（来自Bio。“Notes”的多行文本）。如您所知，必须为此对象和对象所在的段启用“拉伸”属性，以便更改段高度，具体取决于“文本”对象中文本的大小。生成的报表如下所示：



90020	Clown Triggerfish	50	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. 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." Not edible. Range is Indo-Pacific and East Africa to Somoa.
90030	Red Emperor	60	Called seaperch in Australia. Inhabits the areas around lagoon coral reefs and sandy bottoms.

这与我们需要的有点不同 - 如果相邻对象的帧也伸展，它看起来更好。 FastReport可以轻松解决这个问题。在对象检查器中为要拉伸的所有对象和FastReport将“StretchMode”属性设置为smMaxHeight

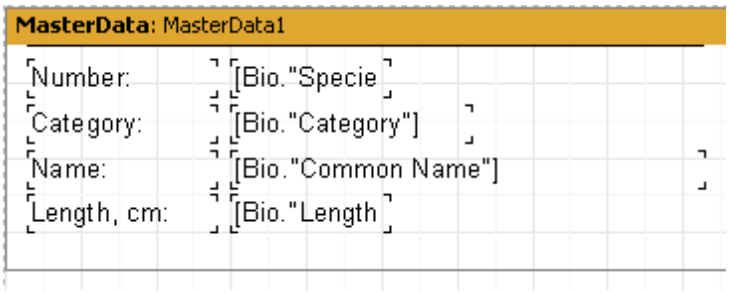
core将首先计算最大band高度，然后将“stretch”对象启用伸展到band的下边缘。由于对象框与对象一起拉伸，因此报表的外观会发生变化：



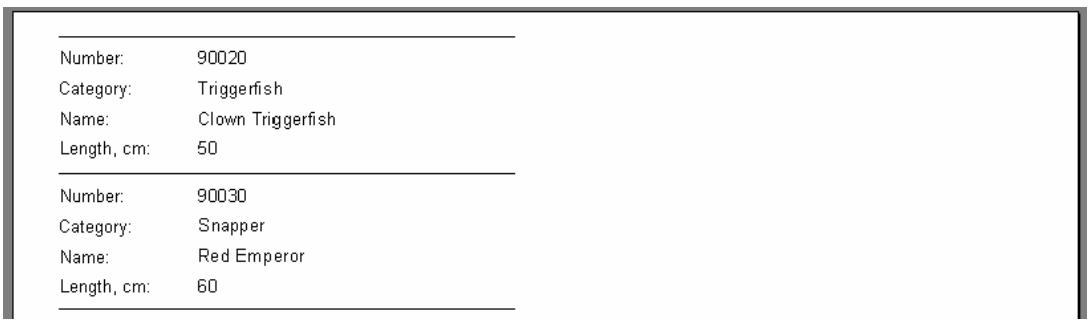
90020	Clown Triggerfish	50	Also known as the big spotted triggerfish. Inhabits outer reef areas and feeds upon crustaceans and mollusks by crushing them with powerful teeth. They are voracious eaters, and divers report seeing the clown triggerfish devour beds of pearl oysters. Do not eat this fish. According 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." Not edible. Range is Indo-Pacific and East Africa to Somoa.
90030	Red Emperor	60	Called seaperch in Australia. Inhabits the areas around lagoon coral reefs and sandy bottoms.

2.19 打印标签

与表类型报表相比，其他报表（例如标签打印报表）可能将数据排列在另一个字段下。让我们看一下此类报表的示例，该报表显示有关鱼的数据，如上例所示。报表数据以标签形式呈现，具有以下结构：



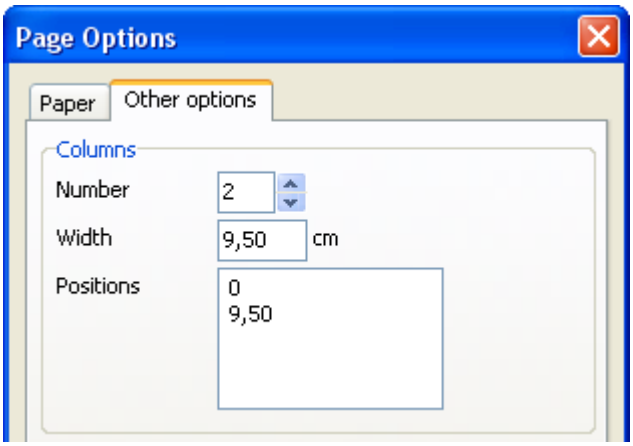
预览时，我们会看到以下输出：



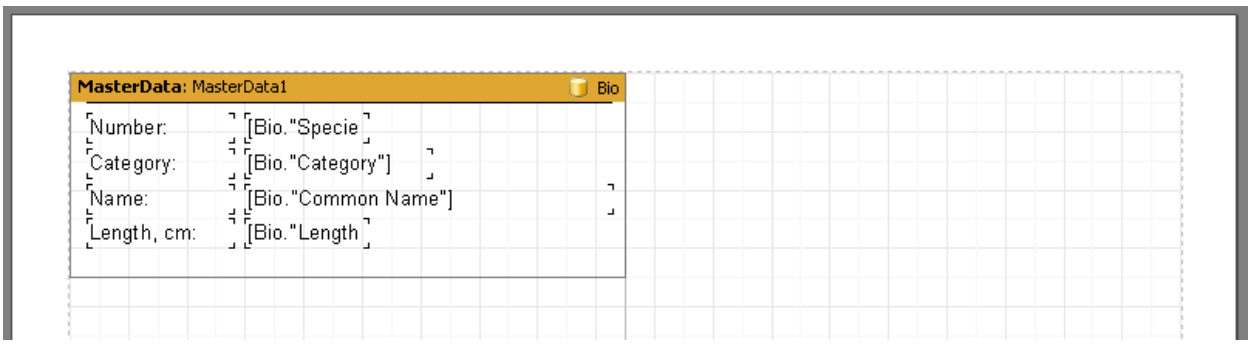
Number:	90020
Category:	Triggerfish
Name:	Clown Triggerfish
Length, cm:	50
Number:	90030
Category:	Snapper
Name:	Red Emperor
Length, cm:	60

请注意，页面右侧有很多空白区域。要使用整个页面，可以在报表页面设置中设置显示数据的列数。去做

这可以双击设计页面上的空白区域，也可以使用菜单上的“文件>页面设置...”。



列数，宽度和位置在对话框的“其他选项”选项卡上设置。在我们的示例中，只需将Number设置为2，因为FastReport会自动调整其他选项。列框在设计器中显示为细垂直线：



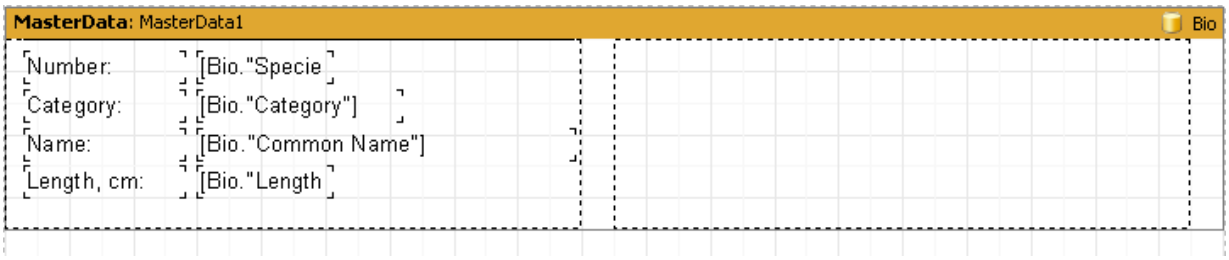
现在，报表按以下方式创建。只要页面底部有空白区域，FastReport就会重复“第一级数据”。之后，将创建同一页面上的新列，并且数据段将从新列的顶部继续重复：这与简单报表形成对比，后者将创建新页面而不是新列。填充第二列后，将创建第三列，依此类推 - 我们的示例只有两列。当所有列都已填满后，FastReport会创建一个新页面并继续显示从第一列开始的数据。

我们的两栏报表应如下所示：

Number:	90020	Number:	90140
Category:	Triggerfish	Category:	Cod
Name:	Clown Triggerfish	Name:	Lingcod
Length, cm:	50	Length, cm:	150
Number:	90030	Number:	90150
Category:	Snapper	Category:	Sculpin
Name:	Red Emperor	Name:	Cabezon
Length, cm:	60	Length, cm:	99

对象检查器中可用于所有数据段的“列”属性是设置列数的另一种方法。但是，如果此属性从零更改，则仅为选定的数据绑定设置列数，而不为整个页面设置列数（如上例所示）。效果是首先从“从左到右，从上到下”显示数据，与上面显示的“从上到下，从左到右”输出形成对比。

禁用“页面选项”对话框中的列（将“列号”设置为1）并在数据和“列”属性中输入2。请注意，还必须从默认的零更改“ColumnWidth”属性，以防止第2列覆盖第1列；可选地，也可以更改“Column Gap”属性。然后FastReport将列框显示为虚线：



此设计以“从左到右，从上到下”的顺序显示数据。

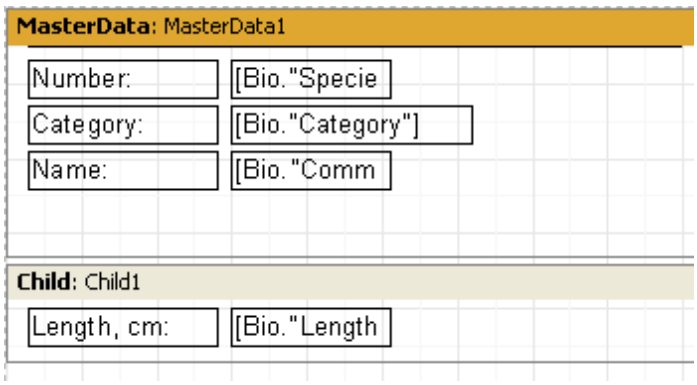
2.20 子段

当标签类型报表中的一个字段具有可变长度的内容时，可能会出现问题。为了在我们的例子中模拟这个，让我们将Bio。“Common Name”对象的宽度减小到2.5 cm并为该对象和“First level data”段启用“Stretch”属性。启用所有对象的所有框线，以便可以清楚地看到拉伸功能的效果。设计现在输出如下报表：

Number:	90020
Category:	Triggerfish
Name:	Clown Triggerfish
Length, cm:	50
Number:	90030
Category:	Snapper
Name:	Red Emperor
Length, cm:	60

这里第一个Bio。“Common Name”字段对象包含大量文本，并被拉伸为两行。这导致位于其下方的Bio。“长度（cm）”场对象向下移动。发生这种情况是因为默认情况下所有对象的“ShiftMode”属性都设置为smAlways，这意味着如果有一个可伸缩对象（启用了“Stretch”属性的“Text”对象），它们会向下移动。移动的距离取决于上面的物体被拉伸的程度。

但这并不是我们想要在我们的标签上发生 - 我们希望“长度，厘米”对象也可以移动相同的量。这可以通过使用称为“Child”段的特殊FastReport段类型来实现。“Child”段与其父段相关联（并在其之后显示）。在设计中添加“Child”段并将两个“Text”对象拖入其中，如下所示：



通过在对象检查器中将其“Child”属性设置为“Child1”，将MasterData band链接到Child band。现在，每次打印MasterData段时，子段会立即打印出来：

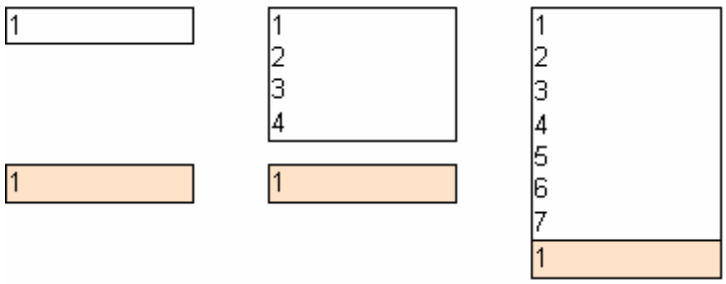
Number:	90020
Category:	Triggerfish
Name:	Clown Triggerfish
Length, cm:	50
Number:	90030
Category:	Snapper
Name:	Red Emperor
Length, cm:	60

“Length, cm: ”标题现在与其值字段“50”完全对齐。如果页面上的空白区域不足（并且从其父段变为“孤立的”孤立状态），则阻止子段移动到下一页，请在对象检查器中启用父段的“KeepChild”属性。

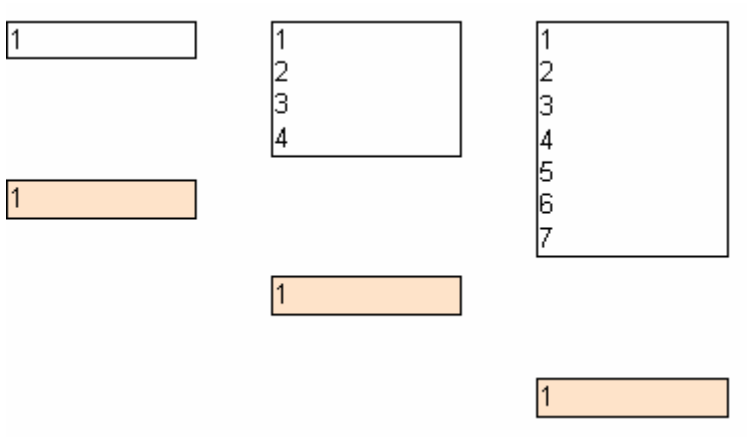
2.21 移动物体

您已经了解了“smAlways”如何适用于“ShiftMode”属性。让我们看看下一个转变模式’smWhenOverlapped’。在此模式下，当上面的对象拉伸并重叠下面的对象时，会发生对象移位。这是两个方案：

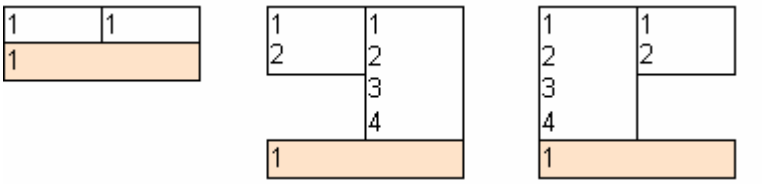
1：三个顶部对象启用了“拉伸”，三个底部对象将“ShiftMode”设置为“smAlways”。底部对象只移动足够远，以便为其上方的拉伸对象留出空间：



2：三个顶部对象启用了“拉伸”，三个底部对象将“ShiftMode”设置为“smWhenOverlapped”。底部对象移动以允许拉伸对象的空间，并且还保持两个对象的设计分离：



这允许创建复杂的报表，特别是在对象可以同时与其上方的几个其他对象重叠的情况下。在下面的示例中，所有上部对象都包含可伸缩文本，所有下部对象都处于“smWhenOverlapped”模式。无论上面对象中的文本长度如何，下面的对象将始终显示在上面的对象附近：



2.22 报表有两个数据级别（主 - 详细信息）

到目前为止，我们的示例报表仅使用一个数据段（“第一级数据”或“MasterData”）来控制数据输出。这足以从一个DB表输出数据。 FastReport还允许在一个设计页面上设计具有多达六个数据级别的报表。此外，通过使用“子报表”对象可以实现报表中无限数量的数据级别 - 稍后将介绍此对象。大多数报表通常只需要一个，两个或三个数据级别，大量数据级别很少见。

我们来看看如何设计两个数据级报表。该报表将从演示表中输出数据：“客户”和“订单”。第一个表是客户列表，第二个表是这些客户下达的订单列表。表包含以下字段中的数据：

顾客：

CustNo	公司
1221	考艾潜水专柜
1231	联合船代
1351	视线潜水员

. . . 。命

令：

订单号	CustNo	发售日期
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

....

如您所见，第二个表包含所有客户下达的所有订单的列表。要查看Orders表中由一个客户在Customers表中放置的订单，这两个表在“CustNo”字段上链接，这两个表都是通用的。此数据的报表输出应如下所示:

1221 考艾潜水专柜
1023 01.07.1988
1123 24.08.1993

1231 联合船代
1060 28.02.1989

1351 视线潜水员
1003 12.04.1988
1052 06.01.1989
1055 04.02.1989

我们来设计报表。在Delphi中创建一个新项目，在表单上放置两个“TTable”，一个“TDataSource”，两个“TfrxDBDataSet”和一个“TfrxReport”组件。设置组件属性，如下所示:

表格1:
DatabaseName = 'DBDEMOS'
TableName = 'Customer.db'

表2:
DatabaseName = 'DBDEMOS'
TableName = 'Orders.db'

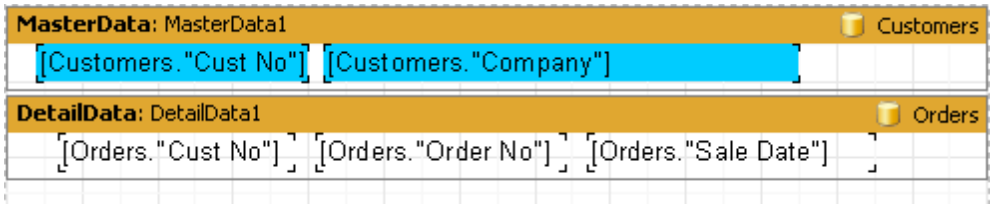
DataSource1:
数据=表

frxDBDataSet1:
DataSet = Table1
UserName = 'Customers'

frxDBDataSet2:
DataSet = Table2
UserName ='Orders'

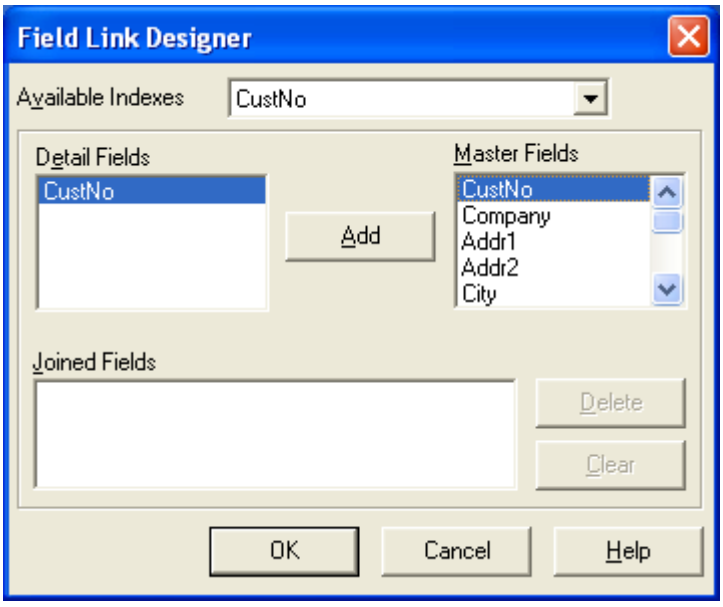
在报表设计器中，在“报表>数据...”对话框中启用数据源。现在添加一个

“MasterData”和“DetailData”段到页面：



请注意，“MasterData”频段必须放在“DetailData”频段上方！如有必要，将其拖到那里。如果将Master放在Detail下，则FastReport将在您预览报表时生成错误消息。

如果您现在预览报表，您会看到每个客户的订单列表保持不变，并包含“订单”表中的所有记录。这是因为我们没有设置“Orders”表的MasterSource属性。在Delphi表单上为“Table2”组件设置“MasterSource = DataSource1”。现在我们已经建立了一个“主 - 细节”关系。之后，我们选择要链接的字段。设置“Table2”组件的“MasterFields”属性：



我们需要将两个来源中的“CustNo”字段链接在一起。为此，请选择所需的字段，然后单击“添加”按钮。“字段”链接将显示在底部窗格中。使用OK关闭编辑器完成。

创建预览时FastReport执行以下操作。从主表（Customer）输出记录后，它会在详细信息表（Orders）上设置过滤器。只有符合’Orders.CustNo = Customer.CustNo’条件的记录才会保留在表格中。这意味着对于每个客户，只有那些属于当前客户的订单才会显示在详细信息区域中。这是一个需要掌握的重要概念。尽管数据段可以是主数据段或详细类型，但它们仅控制输出页面上数据的定位（显示的顺序和次数）。由条段中的对象显示的数据取决于对象链接到哪些字段以及两个表的外部链接。

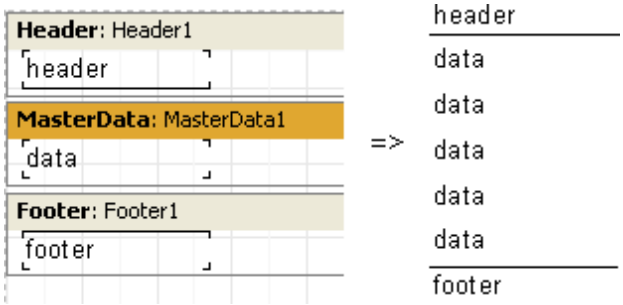
这是最终输出：

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

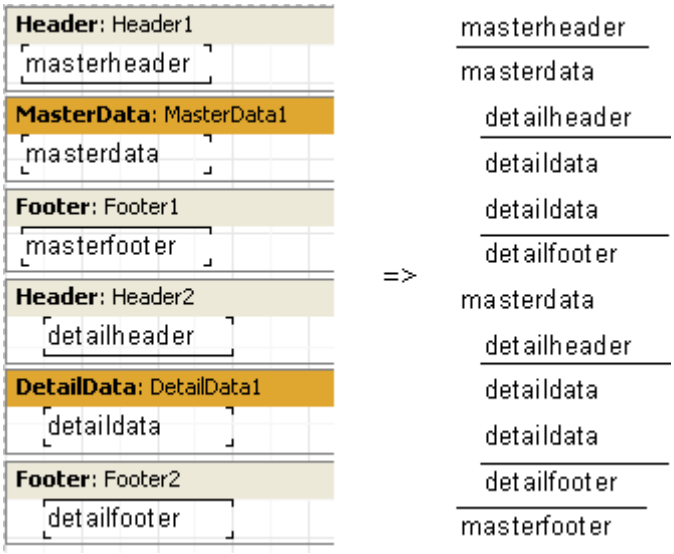
最多包含6个数据级别的报表可以以类似的方式构建。

2.23 数据段的页眉和页脚

每个数据段可以具有页眉和页脚。首先输出标题，然后输出数据段中的所有记录，最后输出页脚。以下是页眉和页脚如何在简单报表中工作的示例：

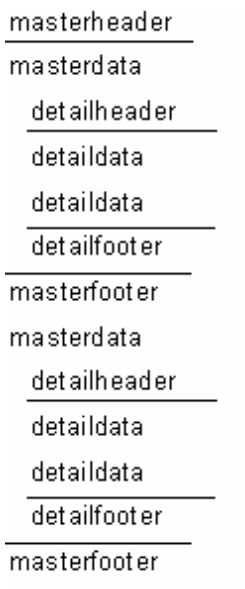


让我们看一个使用两个数据级别的更复杂的例子 – 主数据和详细信息：



如您所见，标头在所有数据段记录之前输出。因此，主数据头在报表开始时输出一次，并且在属于当前主记录段的每组细节段之前输出详细数据头。在输出所有主数据段记录之后，直到属于主记录段和主页脚的细节段组才输出之后输出细节页脚。

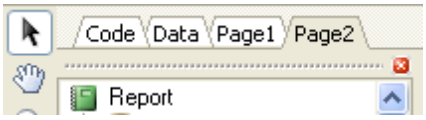
使用数据段的FooterAfterEach属性，我们可以覆盖此行为。将此属性设置为True（您也可以使用数据段的上下文菜单 - “每行后的页脚”）在每个数据行之后导致页脚输出。这在设计一些主 - 详细报表时可能很有用。如果主数据段的FooterAfterEach设置为True，则报表将如下所示：



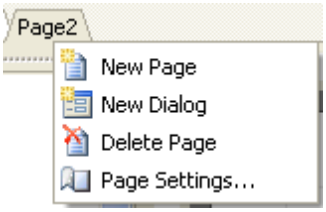
2.24 多页报表

FastReport报表可以包含多个设计页面。多页设计允许调整每个页面的大小和方向等属性，以及允许在页面上放置对象和条段的位置。当输出这种类型的报表时，将显示来自第一个设计页面的所有段，然后显示来自第二个页面的段，依此类推。

在设计器中创建新报表时，默认情况下它包含一个页面。您可以通过单击工具栏中的/按钮或选择“文件>新建页面”菜单命令来添加新页面。然后您会看到设计器中出现一个新页面选项卡：

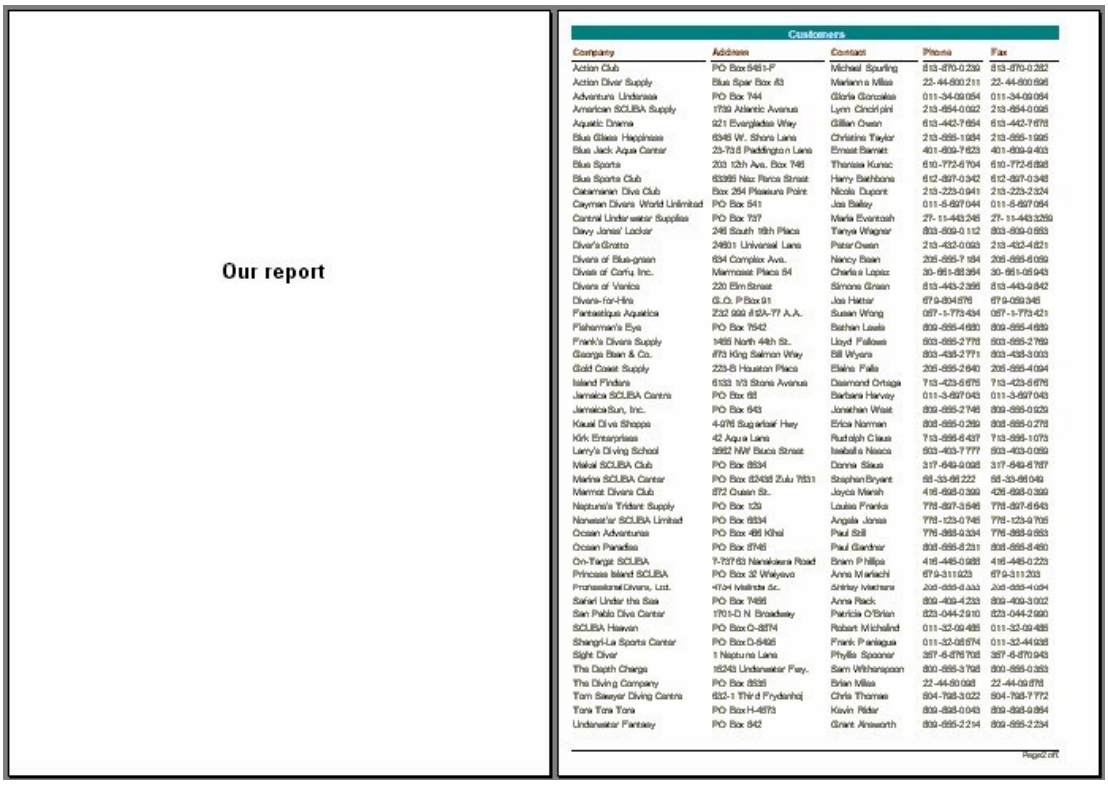


您可以通过单击页面选项卡轻松切换页面。可以拖动页面选项卡（“拖放”）以轻松更改其打印顺序。可以使用工具栏中的/按钮或选择“编辑>删除页面”菜单命令删除不必要的页面。您也可以通过右键单击页面选项卡来调用上下文菜单：



报表中的设计页面数量不受限制。通常，附加页面用于标题页，或者在更复杂的报表中，用于来自许多数据源的数据。

以下是创建标题页的简单示例。让我们使用我们之前的报表，它有一个数据级别。向其添加新页面，它将作为第二页添加。通过使用鼠标抓住page2选项卡并将其拖到第一页选项卡page1上，将其移动到报表的前面。这会更改页面顺序。选择新页面并在页面中间放置一个包含“我们的报表”的“文本”对象。这就是所需要的。标题页的报表已完成：



但是，请注意多页报表的一个功能。如果在第二个输出页面的对象检查器中启用了“PrintOnPreviousPage”属性，则第二个输出页面的对象将开始在第一个输出页面的空白区域上打印，而不是在新的输出页面上打印。

2.25 RowCount和PageCount属性

有时需要多次显示静态数据，例如在打印“空白”名片或明信片时。为此，数据段具有“RowCount”属性，报表页面具有“PageCount”属性。

这两个属性控制报表中的段/页重复的程度，而不受报表数据的影响。

章节 3

组和聚合

3.1 与小组一起报表

在前面的示例中，我们基于两个表中的数据构建了一个两级报表。看起来相同的另一个报表可以在FastReport中构建，这次基于从连接查询获得的数据集。

为此，需要一个SQL查询，它从两个表返回数据并以特定方式排序。在我们的示例中，表将连接在两个表中的“CustNo”字段中。查询可能是：

从客户，订单中选择*
orders.CustNo = customer.CustNo
客户订购。没有

需要“order by”行来对“CustNo”字段上的记录进行排序。此查询返回如下数据集：

CustNo	公司	saledate -ORDERNO.	
1221	考艾潜水专柜	1023	01.07.1988
1221	考艾潜水专柜	1123	24.08.1993
1231	联合船代	1060	28.02.1989
1351	视线潜水员	1003	12.04.1988
1351	视线潜水员	1052	06.01.1989
1351	视线潜水员	1055	04.02.1989

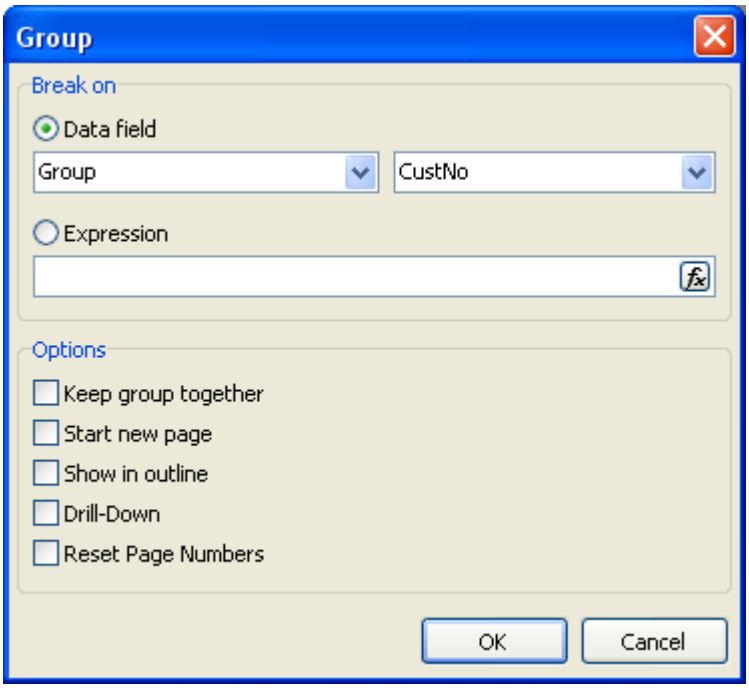
如何使用此数据设计多级报表？在FastReport中有一个特殊的段，即“Group Header”。为段指定条件（DB字段值或表达式）；每当条件值改变时输出段。以下示例说明了这一点。

让我们在Delphi中创建一个新项目，并在表单上放置“TQuery”，“TfrxReport”和“TfrxDBDataSet”组件。设置其属性如下所示：

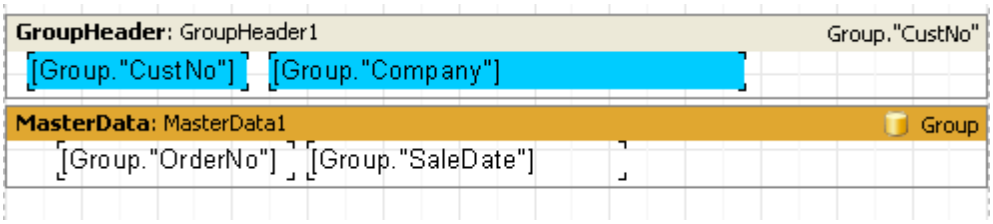
查询1：
DatabaseName = 'DBDEMOS'
SQL =
*从客户，订单中选择**
其中orders.CustNo = customer.CustNo order
by customer.CustNo

frxDBDataSet1:
DataSet = Query1
UserName ='Group'

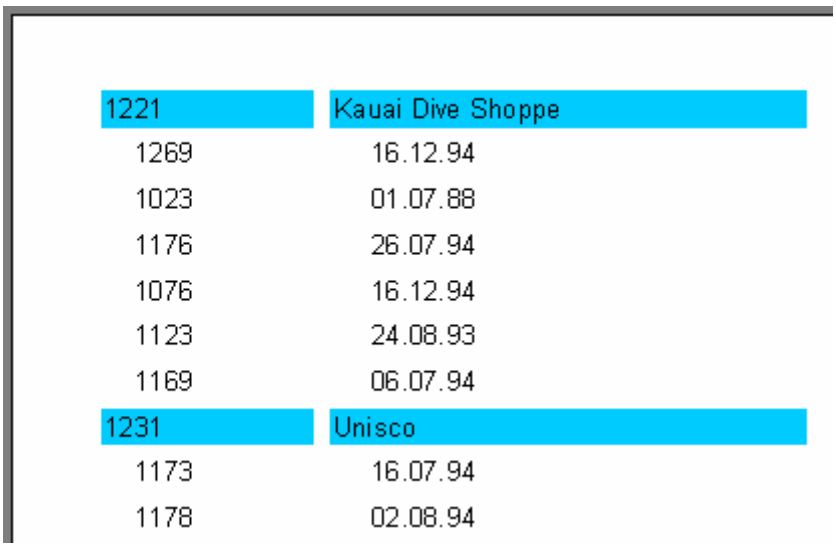
打开报表设计器并创建新报表。将我们的数据源连接到报表。之后，在报表中添加“组标题”区域。在“组标题”段编辑器中设置条件（在本例中为“Group.CustNo”数据字段）：



还将数据段链接到“组”数据源，并如图所示排列一些对象（请注意，组头必须放在数据段上方）：



在预览报表时，我们获得类似于此的输出：



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

如您所见，“Group header”段仅在与其链接的字段时输出

价值的变化。否则，将显示连接到“组”数据集的数据段。如果我们将此报表与之前构建的主详细报表进行比较，很明显订单编号根本没有排序。通过更改SQL查询的“order by”子句可以很容易地纠正这个问题：

```
从客户，订单中选择*
orders.CustNo = customer.CustNo
由customer.CustNo, orders.OrderNo订购
```

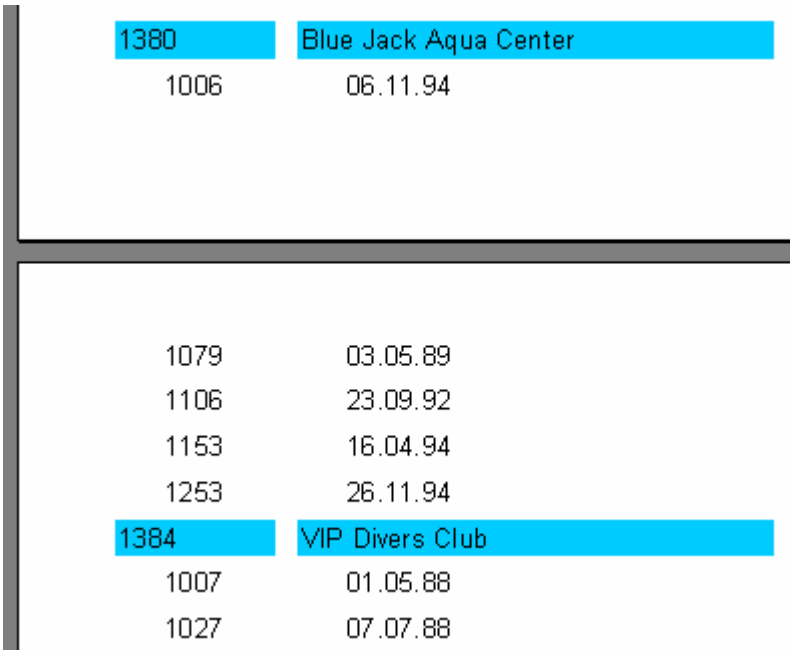
具有嵌套组的报表可以以类似的方式设计。组的嵌套深度是无限的。使用组的报表与主 - 详细信息类型的报表相比具有一些优势：

- 整个报表只需要一个数据集（查询）
- 数据分组级别的数量是无限的
- 数据排序成为可能
- 更优化地使用数据库资源（查询只返回一个数据集用于输出，并通过查询进行过滤）。

唯一的缺点是需要用SQL语言编写查询。但是，对于使用数据库的任何程序员来说，SQL的基本知识几乎是必须的。

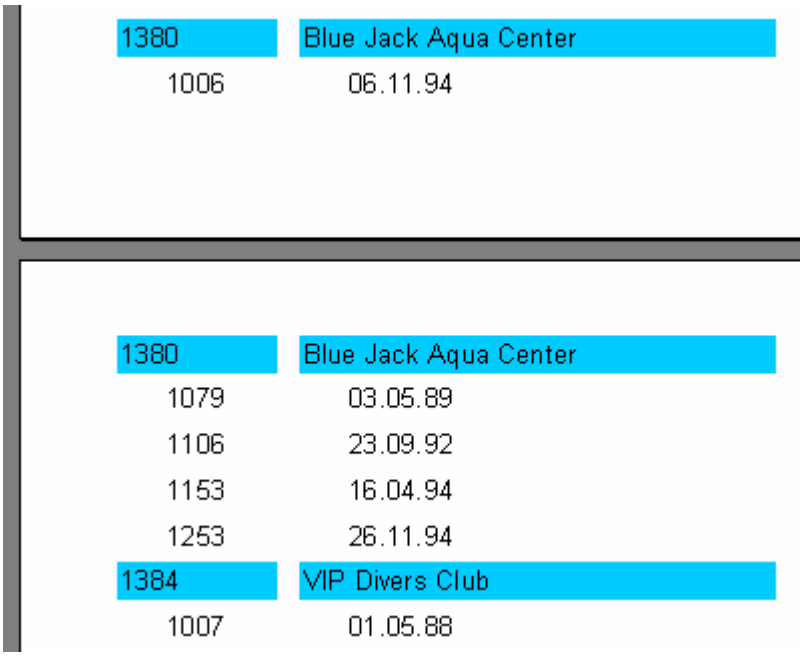
3.2 其他组功能

让我们看一下当报表调用新页面时如何输出组：

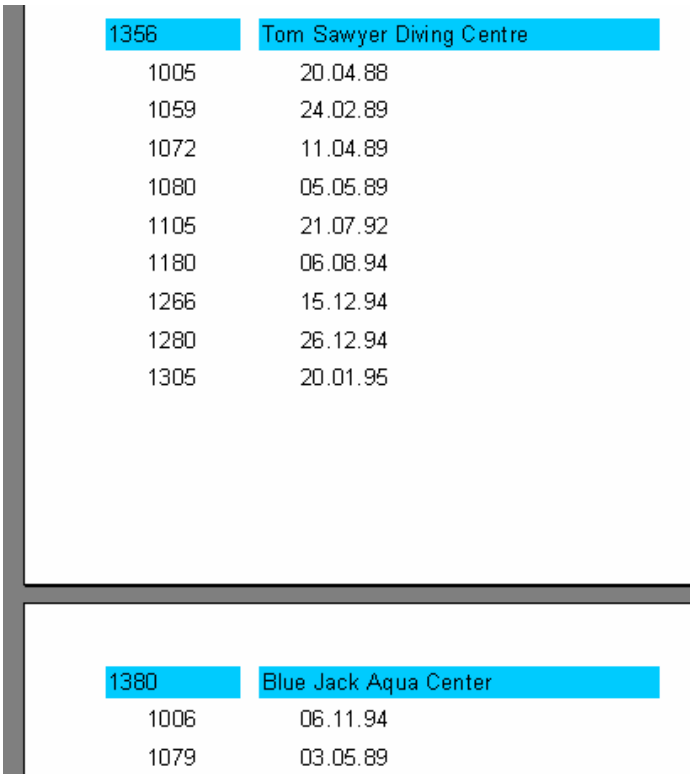


查看报表，不清楚第二页顶部的订单列表是指哪个客户。 FastReport允许在新页面上输出组头（在我们的例子中）

识别客户端）。为此，请使用对象检查器或上下文菜单为“组标题”区域启用“ReprintOnNewPage”属性。这将改变报表，如下所示：



还有另一种方法可以避免在页面边界处破坏组。这是为对象检查器或上下文菜单中的组头启用“KeepTogether”属性。然后，如果整个组不适合输出页面上的可用空间，则将其作为整体移动到新页面。在我们的示例中，出现如下：



某些页面上可能会留下大量空白，但如果可能，该组将在一页上显示完整。

“StartNewPage”组标题属性允许在不同页面上输出组。它可能会导致纸张的浪费，但在某些情况下可能会有用。

3.3 重置页码

“组标题”区域具有“ResetPageNumbers”属性，允许我们在打印组时重置页码。它是为了什么？

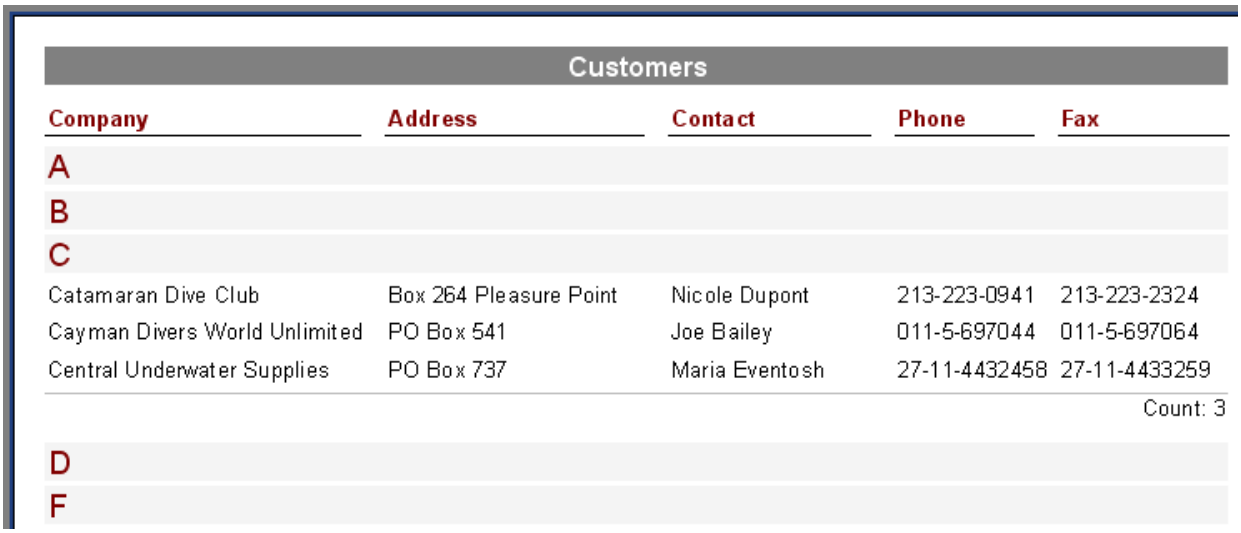
下面是一个示例：您创建了一个报表，该报表将客户名称放在数据段中的组头和客户订单中。现在，您需要打印报表并将其发送给所有客户，每个客户只需获得引用它们的报表页面。遗憾的是，报表中的页码编号是连续的，因此获得编号为50到52的页面的客户会询问“前49页在哪里？”。为了避免这种情况，您必须使用自己的顺序为每个客户的页面编号。在报表内，每个组的页面编号为1。

请注意：如果将“ResetPageNumbers”设置为True，则还应将“StartNewPage”设置为True，以便每个组都将在新页面上启动。要打印页码或总页数，应使用[Page]和[TotalPages]系统变量，而不是[Page#]和[TotalPages#]。

3.4 深入研究小组

组头有一个名为DrillDown的属性。如果将其设置为True，则该组将变为交互式。这意味着您可以单击预览窗口中的组标题，组将展开（显示组中的所有记录）或折叠（仅显示标题，如果ShowFooterIfDrillDown为True，则显示页脚）。

以下是具有一个扩展标头的此类组的示例：



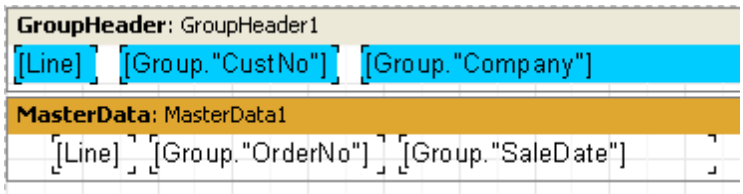
Customers

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				

您可以控制在报表首次运行时是折叠还是展开所有组。默认情况下，组已折叠，但如果要扩展，则可以将ExpandDrillDown设置为True。您还可以使用预览的上下文菜单一次展开或折叠所有组。

3.5 行号

让我们使用我们的示例来说明如何对组中的行进行编号。为此，我们将两个包含系统变量[Line]的“Text”对象添加到我们的两个段（这最容易通过从“数据树”窗格的“变量”选项卡中拖放来完成）。

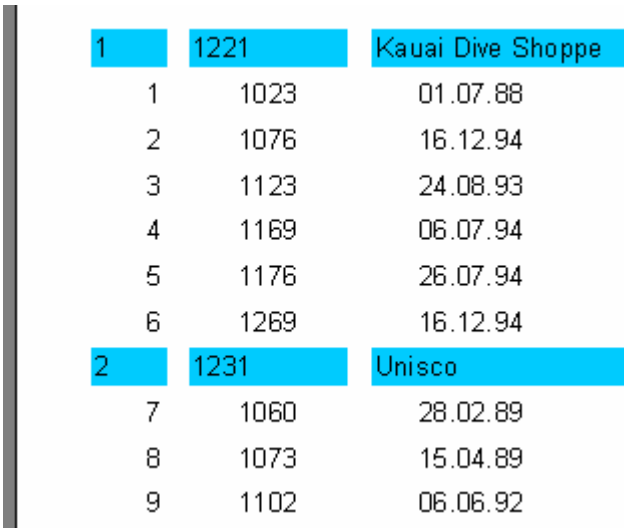


在预览报表时，我们可以看到两个数据级别现在都有自己的行号：



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

要连续编号第二级数据行，请在数据段上的“文本”对象中使用[Line#]变量而不是[Line]。结果将如下所示：



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 聚合函数

在大多数情况下，组报表需要显示一些摘要信息（例如：“组的总数”，“组元素的数量”等）。FastReport提供聚合函数，用于计算某些数据范围内的聚合值。聚合函数是：

和 返回表达式的总和
分钟 返回表达式MAX的最小值 返回表达式AVG
的最大值 返回表达式的平均值
COUNT返回数据范围中的行数（行）

所有聚合函数（COUNT除外）的语法类似于SUM函数的语法：SUM（表达式，band，flags）
SUM（表达式，段）
SUM（表达式）参数

是：

表达 – 要计算的表达式
段 – 执行计算的数据段的名称 一个位字段，段有值
– 1：在计算中包含不可见的段
2：累计聚合作为运行总计
（当前数据跨度重置时不要重置聚合）3 :(前两个选项都是）

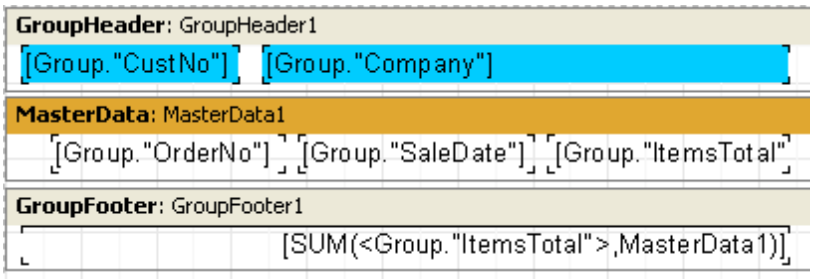
表达式是唯一的必需参数，其他两个是可选的。尽管如此，为了避免出错，建议始终给出频段参数。

“COUNT”聚合函数具有以下语法：COUNT（band，flags）
COUNT（段）

其中参数与上述含义相同。

所有聚合函数都有一个通用规则：聚合只能在数据段上计算，并且只能在该段的页脚中使用，该页脚可以是以下之一：页脚，页脚，组页脚，列页脚或报表页脚（摘要段）。

聚合函数如何工作？我们将使用我们的小组报表示例来查看此内容。让我们在报表中添加一些新元素：



Group。数据段中的“ItemsTotal”字段显示当前订单总数。在组页脚中放置一个“文本”对象，其中包含上面显示的聚合SUM。它将显示给定客户下达的所有订单的总和。使用计算器，我们可以检查结果是否正确：

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

聚合函数的工作方式如下：在输出报表之前，FastReport会扫描“Text”对象内容以查找任何聚合函数。找到的聚合链接到其参数中的数据段（在我们的示例中，“SUM”链接到“MasterData1”段）。在报表输出期间（显示数据段时），计算链接到它的聚合的值。在我们的例子中，Group。“ItemsTotal”字段值被累积。一旦输出显示聚合的组页脚，聚合值将重置为零，并为下一组重复循环，依此类推。

聚合函数中可选“Flags”参数的用途是什么？报表可能隐藏了部分或全部数据段。但是，我们可能需要计算所有数据段的聚合，无论是否可见。在我们的示例中，将数据段的“Visible”属性设置为false，以防止其显示。为了使这个隐藏的数据段包含在计算中，我们必须在函数调用中设置第三个可选参数，如图1所示，即：

[SUM(<基团。“ItemsTotal”>, MasterData1,1)]

这会生成一个类似于tis的报表：

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

当“Flags”参数值设置为2时，聚合值在显示后不会立即重置：聚合成为每个连续输出的“运行”计算。让我们修改函数调用，如下所示：

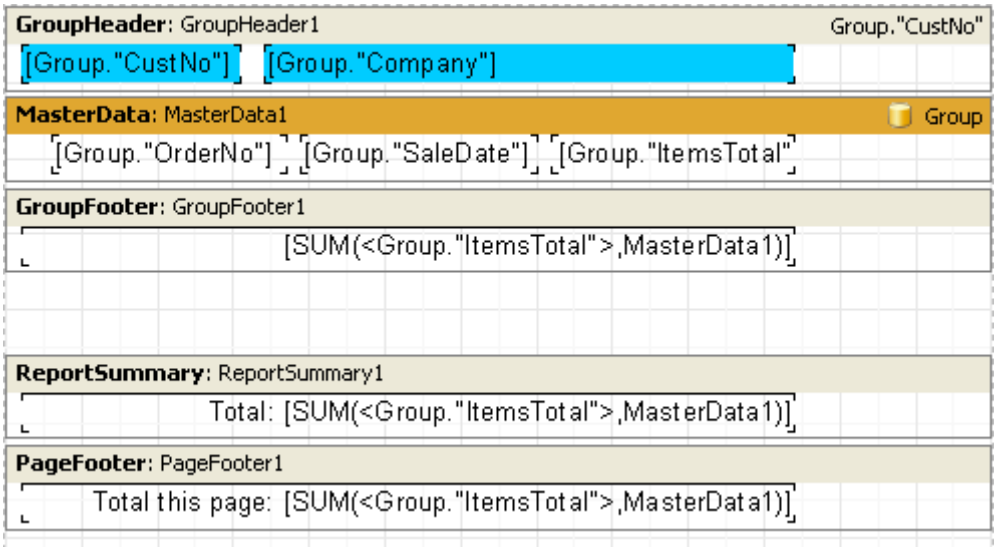
[SUM(<基团。“ItemsTotal”>, MasterData1,3)]

值“3”是“1”和“2”的位组合，这意味着我们需要包括不可见的段，而不需要在每个组之后重置总数。因此，我们有：

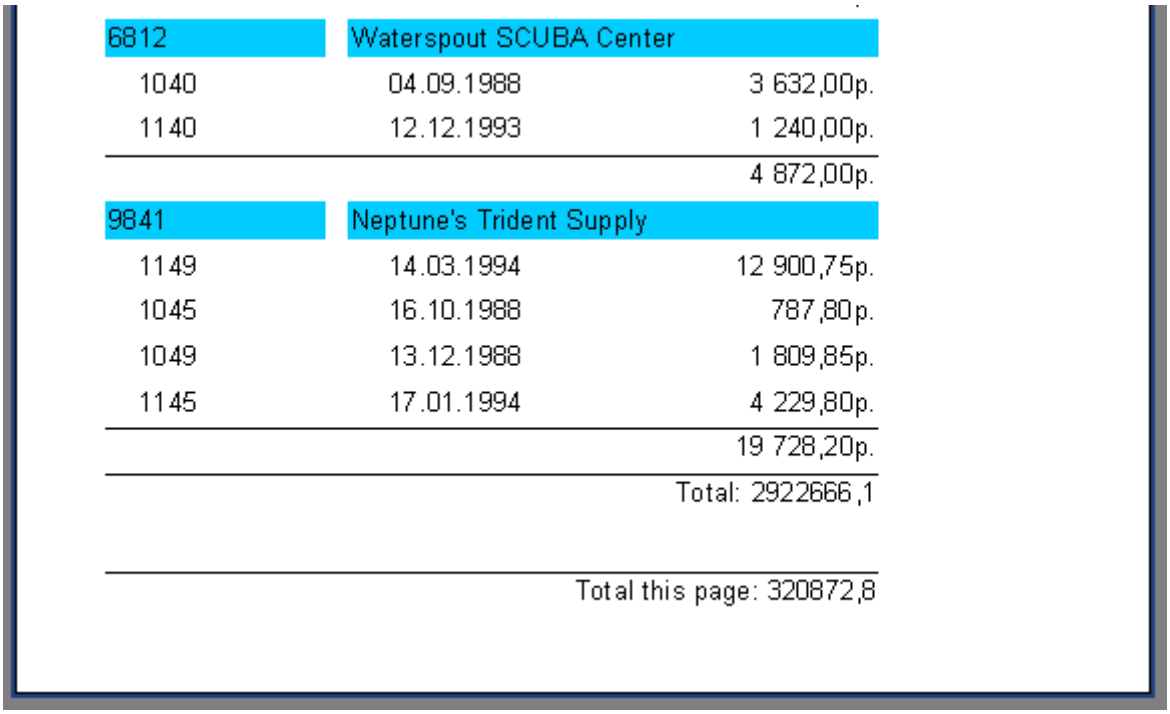
1221	Kauai Dive Shoppe
	51450,8
1231	Unisco
	137094,4
1351	Sight Diver
	398670,2

3.7 页面和报表总计

通常，我们需要显示页面或整个报表的摘要总计。我们也可以在这种情况下使用聚合函数。我们将通过对我们的示例进行一些更改来显示：



正如您所看到的，我们添加了一个“报表摘要”区域，其中包含“文本”对象，其中聚合SUM指向“报表摘要”区域和“页脚”区域。这就是所需要的：

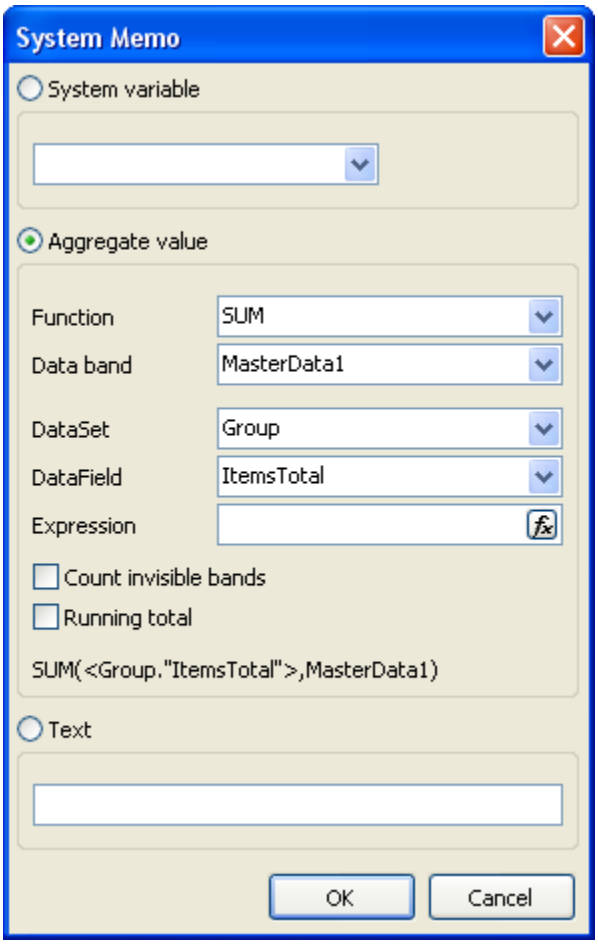


6812	Waterspout SCUBA Center	
1040	04.09.1988	3 632,00p.
1140	12.12.1993	1 240,00p.
		4 872,00p.
9841	Neptune's Trident Supply	
1149	14.03.1994	12 900,75p.
1045	16.10.1988	787,80p.
1049	13.12.1988	1 809,85p.
1145	17.01.1994	4 229,80p.
		19 728,20p.
		Total: 2922666,1
		Total this page: 320872,8

3.8 插入聚合函数

到目前为止，我们已将聚合函数手动插入“Text”对象。现在我们将看看插入聚合函数的其他方法。

首先，我们可以使用“系统文本”对象来输出聚合。事实上，这个对象类似于 “Text”对象，它有自己的特殊编辑器，可以更轻松地指定系统变量或聚合函数：



选择一个函数类型，然后选择一个数据段（在其上计算聚合），最后选择一个DB字段或要计算其值的表达式。如果需要，您还可以设置“计数不可见段”和“运行总计”标记。

第二种方法是使用“Text”对象并单击其编辑器中的/按钮：这将打开类似于“系统文本”对象编辑器的对话框。单击“确定”按钮时，会将对聚合函数的调用插入到对象的文本中。

章节 4

格式化，突出显示

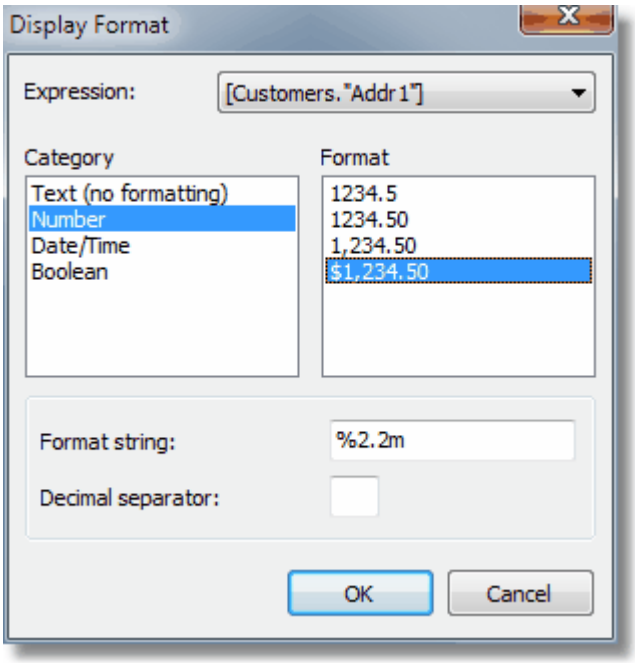
4.1 格式化值

聚合函数的一个特性是返回的数值没有格式化，如第一个例子所示，它使用“SUM”：

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

数据字段通常返回格式化的值，该值仅由“Text”对象显示而不做任何更改。要将格式应用于“SUM”结果，让我们使用FastReport中的值格式设置工具。

选择包含Sum的对象，并在其上下文菜单中的“Display Format ...”或对象检查器中的“DisplayFormat”属性中打开格式编辑器。



此编辑器列出左侧的格式类别，右侧的相应格式以及下面所选类别和格式的格式字符串和小数点分隔符。我们将选择“数字”类别和“$ 1,234.50”格式。格式字符串是Delphi“格式”函数的参数，FastReport用它来实现数字格式。格式字符串和小数分隔符可以更改。如果小数分隔符留空，则使用当前的区域设置值。

单击OK并预览报表后，您将看到报表中的Sum现在已正确格式化：

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

请注意对话框顶部的组合框。如果我们在一个对象中有多个表达式，我们可以为每个表达式设置不同的格式。

4.2 内联格式

内联格式允许您为对象中包含的每个表达式设置不同的格式。它曾用于以前版本的FastReport。现在它已经过时了（使用格式化对话框为每个表达式设置不同的格式）。

使用该示例，重新调整页脚及其对象的大小，并将对象文本更改为：

```
总计：[SUM（<Group。“ItemsTotal”>，MasterData1）]
编号：[COUNT（MasterData1）]
```

订单的总数和订单数将显示在对象中。

在报表预览中，这两个值均以货币格式显示，我们之前已设置。这是不正确的：

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

要以正确的格式显示每个值，需要单独格式化。为此，我们使用格式标记，它位于表达式的右方括号之前。在我们的示例中，禁用对象的格式（在格式编辑器中选择“文本（无格式化）”类别）。现在我们需要为第一个表达式指定格式，因为默认情况下第二个表达式将正确显示（即作为整数）。更改对象文本，如下所示：

```
总和：[SUM（<Group。“ItemsTotal”>，MasterData1）#n%2,2m]
数量：[COUNT（MasterData1）]
```

预览报表以确保正确显示对象：

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

格式标记的一般语法是：

```
[表达式#formattag]
```

请注意，表达式和“#”符号之间的空格字符是必需的！格式标记本身可能如下所示：

#n*format_string* – 数字格式
#d*format_string* – 日期/时间格式
#bFalse，真 – 布尔格式

在每种情况下，Format_string是用于格式化的函数的参数。因此，对于数字格式化，使用Delphi Format函数，并使用FormatDateTime函数的日期/时间。这些函数的语法可以在Delphi帮助系统中找到。以下是FastReport中使用的几个值：

用于数字格式化：
%g – 小数点后面的最小位数
%2.2f – 小数点后固定位数的数字
%2.2n – 如前所述，但有千位分隔符
%2.2m – 货币格式，由Windows操作系统接受，取决于控制面板中的区域设置

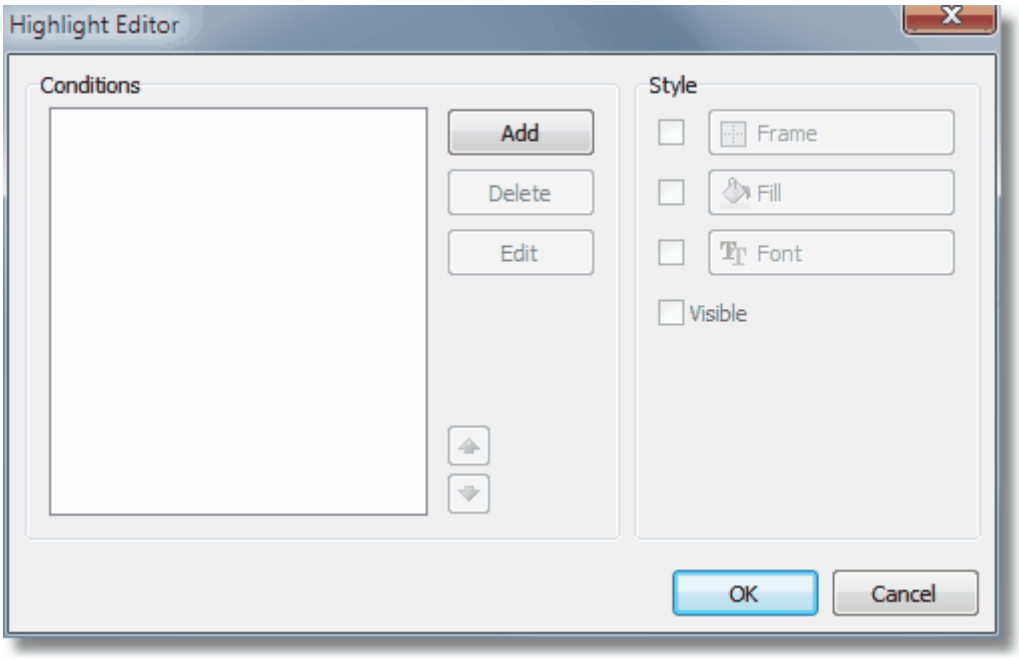
日期/时间格式：dd.mm.yyyy
– 日期为
’23 .12.2003’
dd mmm – 日期为’2003年11月23日’
dd 日期为2003年11月23日’hh：mm 时间为’23：12’
HH：MM：SS – 时间为’23：12：00’
dd mmmm yyyy，hh：mm – 日期和时间为’2003年11月23日，23：12’

可以使用逗号或破折号代替format_string中的点来进行数字格式化。此符号用作整数和值的小数部分之间的分隔符。不允许任何其他角色。

对于使用“#b”类型（布尔值）进行格式化，format_string作为两个以逗号分隔的值输入。第一个值对应于“False”，第二个值对应于“True”。

4.3 有条件的突出显示

根据给定的条件，可以更改“文本”对象的外观。例如，如果对象具有负值，则可以用红色突出显示该对象。
此功能称为“条件突出显示”。要进行设置，请选择“文本”对象，然后单击“文本”工具栏上的按钮。您将看到以下对话框窗口：



可以定义一个或多个条件并为每个条件设置样式。样式可以包含一个或多个设置：

- 帧；
- 填；
- 字体；
- 对象的可见性。

您可以指示在满足条件时需要更改哪些设置。为此，请使用复选框检查所需的设置。

要创建新条件，请单击“添加”按钮。您将看到一个表达式编辑器。在这里，可以编写任何返回布尔结果的表达式。在许多情况下，您将使用“Value”变量，该变量包含当前的打印值。

让我们看看下面的例子：我们有一个“Text”对象，我们在其中打印库存产品的数量：

[产品。“库存量”]

如果产品数量= 0，我们想要将对象绘制为红色。为此，我们创建以下条件：

值= 0

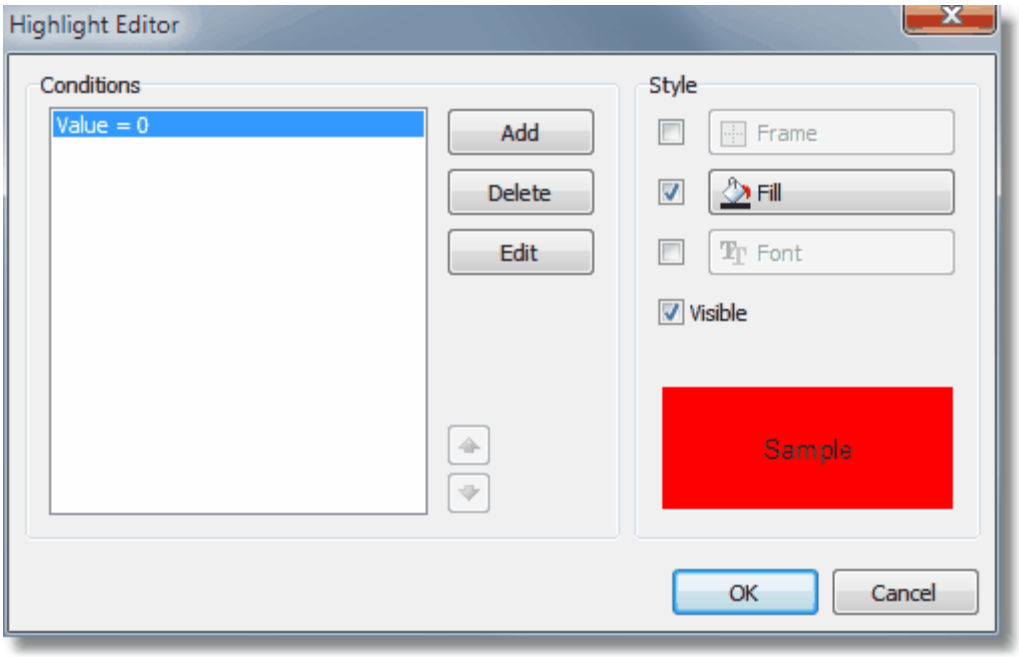
注意：如果您已选择C ++ Script作为脚本语言（请参阅“脚本”一章中的更多详细信息），则应使用C ++脚本编写条件：

值== 0

在给定的情况下，我们使用了“Value”变量，它具有打印值。如果对象中有多个表达式，则此变量将具有最后一个表达式的值。您可以使用数据列代替“值”：

〈Products。“UnitsInStock”〉= 0

为给定条件配置样式，使得只能使用填充，并选择红色：



打印具有零值的对象时，它将为红色。让我们让我们的例子更复杂，我们将添加另一个条件。如果库存中的单位小于10，则必须打印为黄色。为此，请打开条件编辑器并单击“添加”按钮。第二个条件是这样的：

值〈10

如果指示了多个条件，FastReport将从第一个条件开始检查所有条件。如果满足某个条件，FastReport会将其样式设置应用于该对象，并且该过程将停止。以正确的顺序排列条件非常重要。我们在这个例子中看到的顺序是正确的：

1. 值= 0
2. 值〈10

如果我们交换条件，那么突出显示将错误地工作。

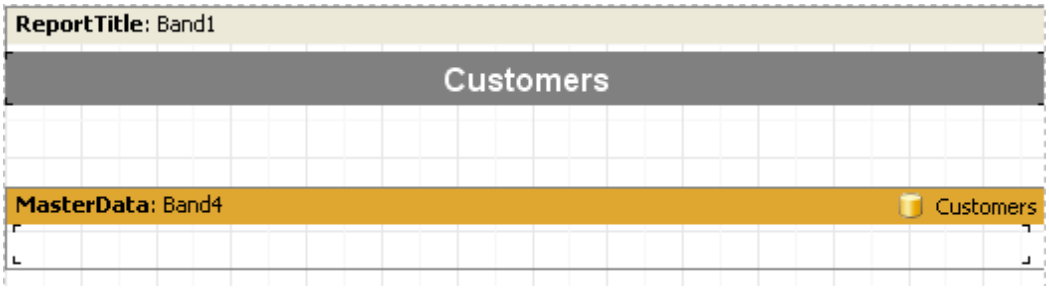
1. 值〈10
2. 值= 0

在给定的情况下，“Value = 0”将不会被执行，因为当值为零时，则为第一个条件将得到满足。要更改条件的顺序，请使用 和 按钮。

4.4 着色备用数据行

使用条件突出显示，可以轻松创建具有“段状”外观的报表，其中数据行交替着色。为了省力，让我们使用之前设计的“客户列表”示例。

从“MasterData”段中删除所有“Text”对象。在数据段上放置一个“文本”对象并将其拉伸以覆盖几乎所有的段空间：



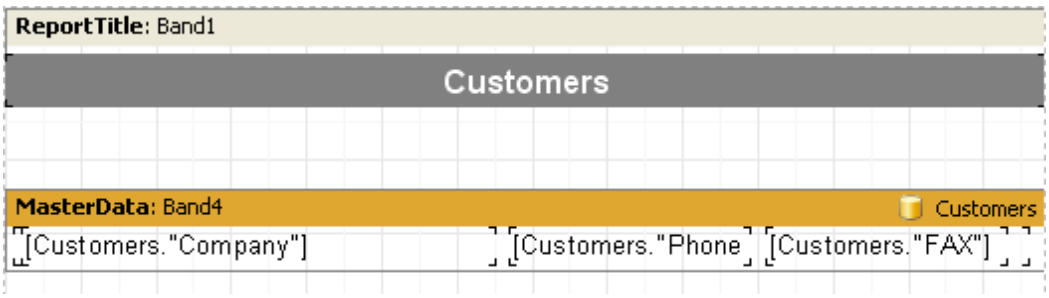
此对象将根据数据行号更改其颜色。选择对象并在对象编辑器的“突出显示”选项卡中设置以下条件表达式：

〈线〉mod 2 = 1

注意：如果您已选择C ++ Script作为脚本语言（请参阅“脚本”一章中的更多详细信息），则应使用C ++脚本编写条件：

〈Line〉%2 == 1

选择灰色作为突出显示的颜色，颜色不要太饱和，但要接近白色。现在可以将其他对象添加到第一个空“Text”对象之上的数据段中：



在预览时，报表生成此输出：



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

章节 5

嵌套报表（子报表）

5.1 嵌套报表（子报表）

有时需要非常复杂的报表，其中包含在设计中的特定点插入的附加数据块。虽然其中许多报表可以使用FastReport频段的排列进行设计，但有时它会变得太复杂。在这些情况下，有必要使用“子报表”对象/。

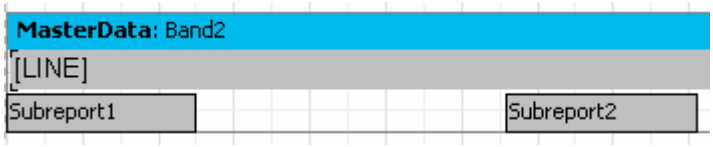
当“子报表”对象插入到设计中时，FastReport会自动添加一个新页面，该页面连接到“子报表”。这样的嵌套报表在设计结构方面类似于多页报表。唯一的区别是嵌套报表显示在基本设计页面上的特定位置，而不是在它之后。输出此报表时，只要遇到“子报表”对象，报表引擎就会完整地输出连接的子报表页面。之后，输出继续进行基本设计页面的其余部分。

可以将更多“子报表”对象插入子报表设计页面，从而增加嵌套深度。可以在演示程序中找到嵌套报表的示例 - “子报表”报表。

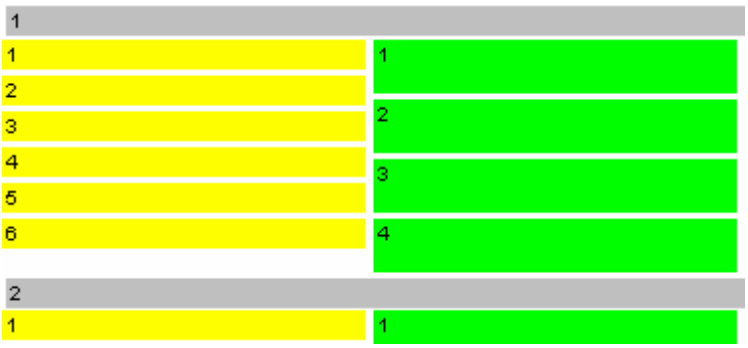
值得注意的是，FastReport使用子报表的能力可以实现深度嵌套数据。请记住，当使用Detail数据段而不是“Subreport”对象时，FastReport中的数据级别数仅限于6。

5.2 并排的子报表

两个或多个“子报表”对象可以并排放置在同一数据段上：



此设计允许报表，其中每个“子报表”的数据输出具有不同长度的行/记录，或不同的高度或拉伸程度：

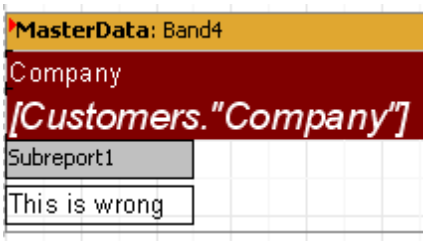


如图所示，FastReport仅在最长的子报表完成后才继续输出基本设计页面。 Vertical Alignment属性还可用于调整每个子报表中的“Text”对象对齐。

5.3 使用子报表的限制

由于子报表放在基本设计页面上，因此它们不能包含以下任何段：“ReportTitle / ReportFooter”，“PageTitle / PageFooter / PageBackground”或“ColumnTitle / ColumnFooter”。如果这些段中的任何一个放置在嵌套报表页面上，则无法识别它们。出于同样的原因，更改任何嵌套的报表页面选项没有意义，因为基本报表页面的选项会覆盖任何嵌套页面的选项。

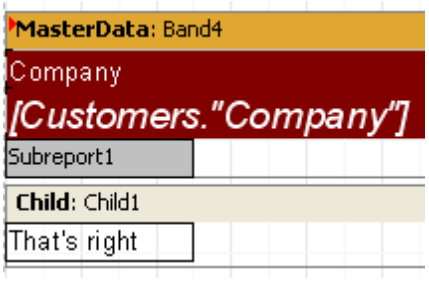
不要在“Subreport”对象下面放置对象：



如果这样做，那么在子报表中创建的对象将覆盖主设计页面上子报表对象下面的所有内容，输出将如下所示：



要在嵌套报表下方或之后显示对象，请使用子段：

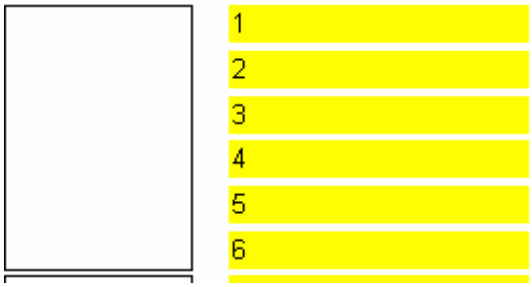


当多个子报表必须一个放在另一个之下时，也会使用此方法。为每个子报表使用子段并将它们链接在一起，将Child1的子属性设置为Child2，依此类推。

5.4 PrintOnParent选项

“Subreport”对象具有“PrintOnParent”属性，有时可能很有用。默认情况下，此属性为False。

通常，子报表在基本报表页面上作为一组段输出。如果是这样，包含“子报表”对象的父段的高度不受子报表中的段控制，即它不能被拉伸。如果子对象检查器或上下文菜单中的子报表“PrintOnParent”属性设置为True，则子报表中的对象将物理打印在包含“子报表”对象的段上。这个段子可以拉伸，并可以放置拉伸的物体：



章节 6

脚本

脚本是一种用高级语言编写的程序，它是报表的一部分。报表运行时，脚本也会运行。脚本可以使用FastReport核心的正常操作无法处理数据；例如，脚本可以根据预定义的条件隐藏冗余数据。脚本还可用于控制作为报表一部分的对话表单的属性。

脚本使用脚本引擎支持的一种语言（FastScript）编写。目前，这些是：

- PascalScript
- C ++脚本
- BasicScript
- JScript中

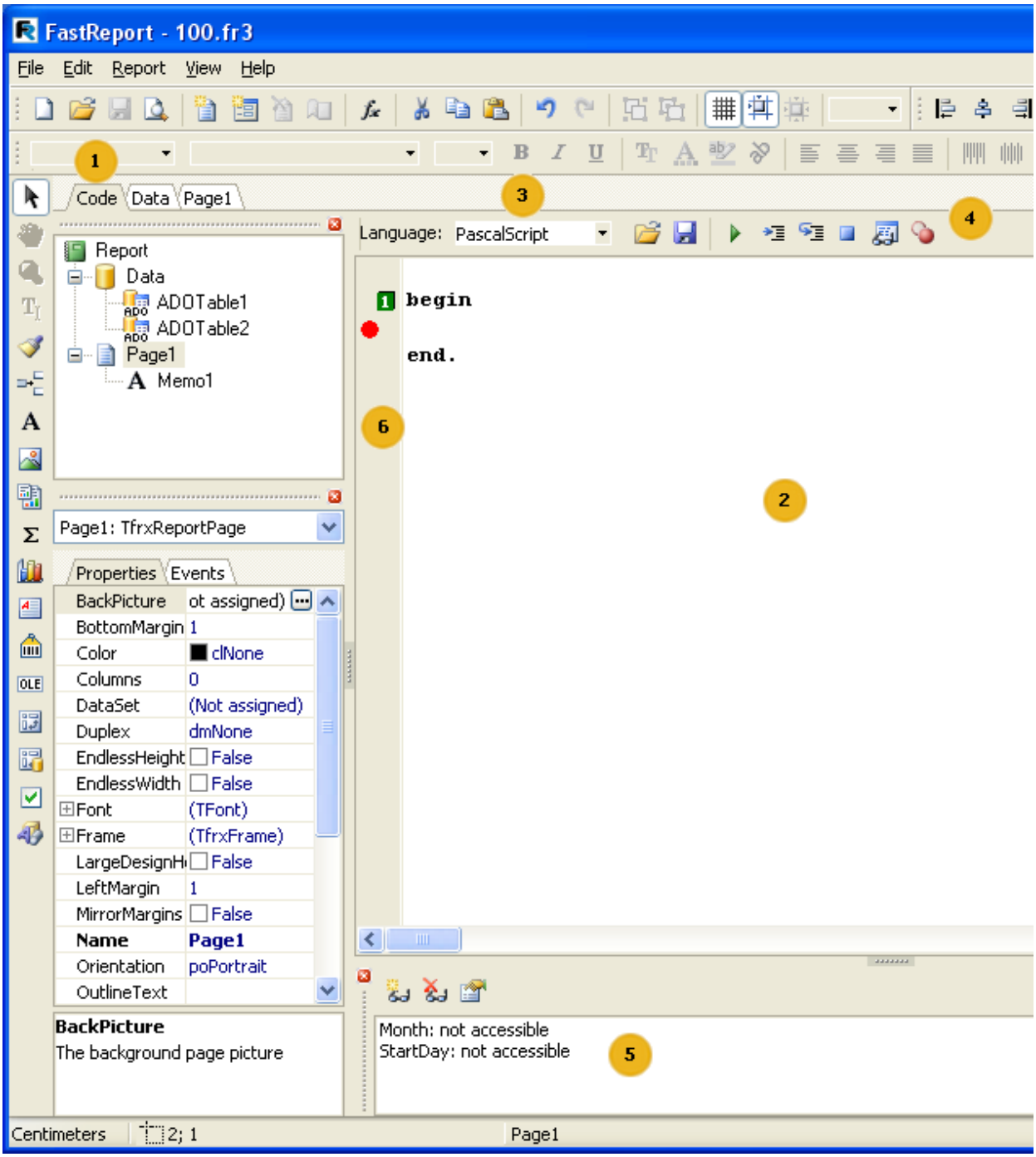
FastScript引擎支持以下功能：
- 标准语言集：变量，常量，过程，函数（可以嵌套并具有变量，常量，默认参数），所有标准运算符（包括case，try，finally，except，with），类型（整数，分数，逻辑，字符，行，多维数组，变体），类（包含方法，事件，属性，索引和默认属性）
- 类型兼容性检查
- 访问任何报表的对象

但是，FastScript不支持以下内容：
- 这些类型的声明：记录，类
- 指针，集合（但'IN'运算符可用于表达式，例如“a'['a'...'c'，'d']中的a”）
- 短裤型
- 无条件跳跃（GOTO）

可以在FastReport设计器中创建脚本，该设计器包含段语法突出显示的脚本编辑器。还有一个嵌入式调试器，它具有以下功能：“Step”，“Breakpoint”，“Run to cursor”和“Evaluate”。

6.1 脚本的味道

使用脚本的工具位于FastReport设计器的“代码”选项卡中。切换到此选项卡时，IDE如下所示：



上述标签的关键：

1 – “代码”标签
2 – 脚本编辑器窗格
3 – 用于选择编写脚本的语言的下拉列表
4 – 调试工具栏
- 在调试模式下运行报表（F9）
- 运行到光标（F4）
- 执行常规代码行（Step into，F7）

- 中断运行脚本（Ctrl + F2）
- 表达式的预览值（Evaluate，Ctrl + F7）
- 切换断点（F5）5 -

“手表”窗格

6 – 书签和断点显示在此列中；此外，这里突出显示执行代码行

可以在脚本编辑器中使用的快捷键列表：

键	含义
光标箭头	移动光标
PageUp，PageDown	转到上一页/下一页
按Ctrl + PageUp键	转到文本的开头
按Ctrl + PageDown键	转到文本的末尾
家	去行的开头
结束	走到尽头
输入	转到下一行
删除	删除光标位置的符号；删除所选文字
退格	删除光标左侧的符号
按Ctrl + Y	删除当前行
按Ctrl + Z	撤消上次操作（最多32个事件）
Shift +光标箭头	选择一个文本块
按Ctrl + A	选择全文
按Ctrl + U	将所选块向左移动2个符号
按Ctrl + I	将选定的块向右移动2个符号
Ctrl + C，Ctrl + Insert	将所选块复制到剪贴板
Ctrl + V，Shift + Insert	粘贴剪贴板中的文本
Ctrl + X，Shift + Delete	将所选块剪切到剪贴板
CTRL + SHIFT + <编号>	在当前行上设置段<number> 0..9的书签
CTRL + <编号>	跳转到书签<编号>
按Ctrl + F	搜索一行
Ctrl+R	替换一条线
F3	从光标位置重复搜索/替换
F4	设置要运行的脚本的断点（运行到游标）
按Ctrl + F2	重置脚本
按Ctrl + F7	变量的预览值（Evaluate）
F9	运行脚本（运行）

F7或F8	执行代码行 (Step into)
Ctrl +空格	显示对象的方法和属性列表
按Ctrl + Shift + Delete键	删除光标右侧的单词
按Ctrl + Shift + Backspace键	删除光标左侧的单词

6.2 脚本的结构

脚本的结构取决于使用的语言；但是每种语言都有一些共同的元素。这些是脚本的标题和正文，以及报表运行时将执行的主要过程。以下是所有四种受支持语言的脚本示例：

PascalScript结构：

```
#language PascalScript //可选
程序MyProgram;                    // 可选的

//"使用"章节应位于任何其他章节之前
使用'unit1.pas', 'unit2.pas';

是                                //"变量"章节可以放在任何地方
  我, j: 整数;

常量                              //"常量"章节
  PI = 3.14159;

程序p1;                           //程序和功能
是
  我: 整数;

  程序p2;                         //嵌套程序
  开始结束;

开始结束;

开始                              //主程序
结束。
```

C ++脚本结构：

```
#languageC++脚本                  // 可选的

//"包含"章节应放在任何其他章节之前
#include"unit1.cpp", "unit2.cpp"
```

```
int i, j = 0;                    //"变量"章节可以放在任何地方

#定义PI = 3.14159                //"常量"章节

void p1 ()                       // 功能
{                                //没有嵌套程序
}

{                                //主程序
}
```

JScript结构:

```
#语言JScript                     //可选

//"导入"章节应该在任何其他章节之前
import"unit1.js", "unit2.js"

var i, j = 0;                    //"变量"章节可以位于任何地方

功能p1 ()                        // 功能
{                                //
}
                                        //主程序
p1();
for (i = 0; i <10; i ++) j ++;
```

BasicScript结构:

```
#语言BasicScript                 '可选

'the"import"章节应位于任何其他章节导入"unit1.vb", "unit2.vb"之前

Dim i, j = 0                     ''变量'章节可以放在任何地方函数p1 ()    ' 功能
{                                '
}
                                      '主要程序。
对于i = 0到10 p1 ()
下一个
```

有关FastScript引擎的更详细说明，请参见其文档。此用户手册中不再重复此信息：
- 每种支持语言的语法图表
- 支持的数据类型
- 具有类，属性，方法和事件的操作
- 嵌套函数

- 枚举和集合

稍后我们将看一下用“PascalScript”语言编写的脚本示例。创建新报表时，默认情况下会选择此语言。

6.3 “你好，世界！”脚本

我们已经看到了一个“Hello，World！”报表的例子；现在让我们看看如何创建一个显示具有相同问候语的窗口的简单脚本。

打开设计器并单击“新建报表”按钮，以便FastReport自动创建基本模板。切换到“代码”选项卡并编写以下脚本：

PascalScript：

```
开始
  ShowMessage ('Hello, World!') ;
结束。
```

C ++脚本：

```
{
  ShowMessage ( “Hello, World! ” ) ;
}
```

之后运行报表。正如预期的那样，FastReport会显示一个段问候语的小对话：



我们来解释一些细节。我们创建了一个由单个“begin..end”块组成的脚本。所以我们的脚本结构非常简单；它只包含一个主程序（参见上面的“脚本结构”部分）。报表运行后立即执行主程序。在这种情况下，它显示了问候对话；该程序在对话结束后立即结束。主程序运行完毕后，正常的报表构建开始。

6.4 在脚本中使用对象

可以从脚本访问任何报表对象。因此，如果存在“Page1”页面和“Memo1”对象，则可以在脚本中使用它们，并按名称调用它们：

PascalScript：

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

C ++脚本:

```
memo1。颜色= clred
```

可从脚本访问的报表对象列表显示在“报表树”窗格中。脚本可以使用哪些对象属性？答案很简单：在对象检查器中可见的所有内容。对象检查器还显示底部每个属性的提示。使用脚本时，两个窗格（报表树和对象检查器）都可用。要获取有关对象属性和方法的详细帮助，请使用分发工具包中包含的FastReport帮助文件。

这是一个简单的例子。将名为 “MyTextObject”并包含 “Test”的 “Text”对象放在报表设计页面上。然后写这个脚本：

PascalScript:

```
开始
  MyTextObject.Color: = clRed
结束。
```

C ++脚本:

```
{
  mytextobject。颜色= clred
}
```

运行报表，看到对象的颜色为红色。

6.5 从报表的变量列表中调用变量

可以在脚本中引用报表变量列表中定义的任何变量（“Report> Variables ...”菜单项）。变量的名称应该用尖括号括起来：

PascalScript:

```
如果<我的变量> = 10则...
```

C ++脚本:

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

另一种方法是使用“获取”功能：

PascalScript:

```
如果Get ('我的变量') = 10则...
```

C ++脚本:

```
if (Get ("我的变量") == 10) {...}
```

变量的值仅通过"Set"过程更改:

PascalScript:

```
设置('我的变量', 10);
```

C ++脚本:

```
设置("我的变量", 10);
```

值得注意的是，要为变量分配字符串值，您必须在值周围添加引号:

PascalScript:

```
设置('my variable', ''''+'String'+'''');
```

C ++脚本:

```
设置("我的变量", "\"字符串\"");
```

系统变量，例如"Page#"，应以完全相同的方式引用:

PascalScript:

```
如果<Page#> = 1则...
```

C ++脚本:

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

6.6 引用数据库字段

与变量一样，在引用报表中的数据库字段时应使用尖括号:

PascalScript:

```
if <Table1。 "Field1"> = Null然后......
```

C ++脚本:

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

或者，“Get”函数可用于访问DB字段（实际上，当计算括在尖括号中的表达式时，FastReport会隐式使用此函数）。

6.7 在脚本中使用聚合函数

聚合函数的一个特性是它们必须在“Text”对象中使用；一旦以这种方式使用，它们就可以在脚本本身中使用。如果聚合函数仅出现在脚本中（不出现在“Text”对象中），则会生成错误消息。这是因为聚合函数必须连接到特定的段；一旦这样连接它将正常工作。

6.8 在报表中显示变量的值

可以在脚本中本地声明和使用变量。声明后，脚本变量可以赋值给它。以下是正在使用的脚本变量的简单示例：

PascalScript：

```
是
  MyVariable: String; 开
始
  MyVariable: =’你好！’;
结束。
```

C ++脚本：

```
字符串MyVariable;

{
  MyVariable =“你好！”;
}
```

变量的值可以显示在“Text”对象中，例如，通过在对象中键入“[MyVariable]”。

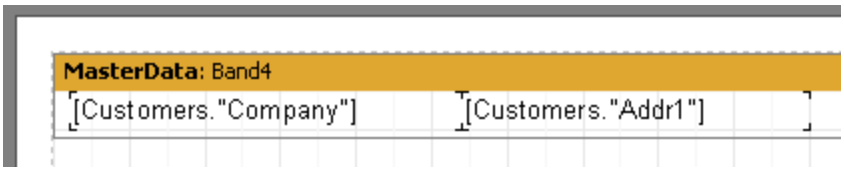
变量的名称必须是唯一的。这意味着名称不得复制任何其他报表对象，标准函数或常量的名称。如果脚本中存在错误，则在运行报表时将显示一条消息，并且将停止报表构造。

6.9 活动

到目前为止，我们只查看了一个主程序的脚本，该程序在报表开始运行时执行。在主程序中，可以进行初始设置并初始化变量。但是，这还不足以完全控制报表生成过程。为了在报表生成上实现尽可能多的控制，每个对象都有几个事件

可以分配处理程序（即脚本中的过程）。例如，将处理程序连接到数据段使得能够过滤记录，从而可以根据满足的特定条件隐藏或显示该段。

让我们演示报表创建过程以及通过包含一个页面并具有一个“MasterData”段的简单报表触发的事件，段上有两个 “Text”对象：



如上所述，在运行报表的最开始调用脚本的主过程。之后，报表建设的要点就开始了。首先调用 “Report”对象的 “OnStartReport”事件。然后，在创建输出页面之前，调用 “OnBeforePrint”页面事件。对于报表模板中的每个设计页面，都会调用此事件一次（不应将设计页面与报表的输出页面混淆！）。在我们的示例中，事件被调用一次，因为报表设计只包含一个设计页面。

然后按以下顺序调用数据段的事件：

1. 段的“OnBeforePrint”活动被召唤
2. 调用包含在段中的每个对象的“OnBeforePrint”事件
3. 每个对象都填充了数据（在我们的示例中，包含“Company”和 “Addr1”DB字段的值）
4. 调用每个对象的“OnAfterData”事件
5. 诸如在段子上定位物体的动作（如果它们之间有可伸缩的物体），计算段高度并拉伸它（如果它是可拉伸的）被执行
6. 段的“OnAfterCalcHeight”事件被调用
7. 如果段在页面的空白处没有足够的空间，则会创建一个新的输出页面
8. 段及其所有对象都显示在输出页面上
9. 调用每个band对象的 “OnAfterPrint”事件
10. 调用段本身的“OnAfterPrint”事件

只要连接到段的源有数据，段就会继续打印。在该报表打印停止之后，将调用报表页面的 “OnAfterPrint”事件，最后调用 “Report”对象的 “OnStopReport”事件。

因此，通过使用不同对象的事件，几乎可以管理报表创建过程的每个步骤。使用事件的关键是对频段输出过程的全面了解，这将在接下来的九个部分中讨论。大多数动作只能使用段的 “OnBeforePrint”事件来执行；对对象所做的任何修改都会立即显示。但是，如果段是可伸缩的，则在这种情况下无法说明段将在哪个页面打印，因为段高度的计算是在步骤5中执行的。但是，可以在 “OnAfterCalcHeight”中完成。步骤6中的事件或步骤9中的 “OnAfterPrint”事件。请注意，在最后一个事件中，已经输出了段，因此对对象的修改将不会产生任何可见效果。

必须清楚地理解输出频段的“何时何地”，并了解每个事件的时间（呼叫顺序）。同样地，对于包含在其中的每个对象

段。

6.10 使用“OnBeforePrint”事件的示例

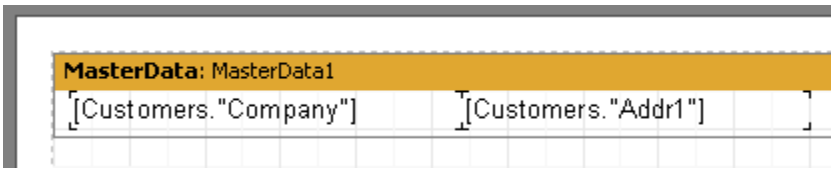
要演示此事件，请创建表示客户端列表的报表。此报表仅包括名称以字母“A”开头的公司。

让我们在Delphi中创建一个新项目，在表单上放置“TTable”，“TfrxDBDataSet”和“TfrxReport”组件并设置这些属性：

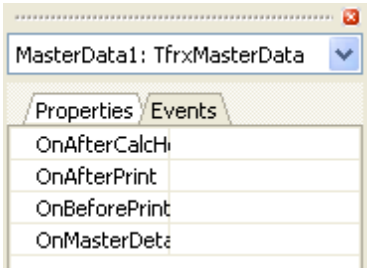
表格1：
DatabaseName = 'DBDEMOS'
TableName = 'customer.db'

frxDBDataSet1：
DataSet = Table1
UserName = 'Customers'

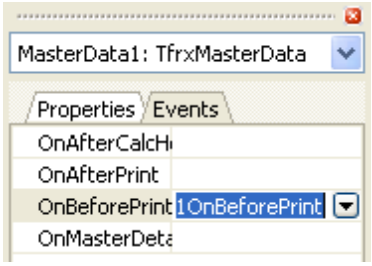
打开报表设计器并创建如下报表：



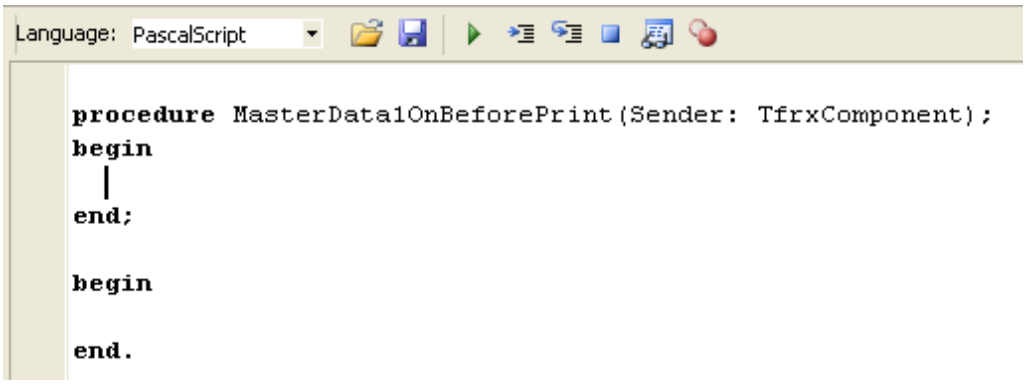
选择数据段并切换到对象检查器中的“事件”选项卡：



要创建“OnBeforePrint”事件处理程序（最适合我们），请双击事件名称右侧的空白字段：



这会向脚本添加一个空白处理程序，设计器会切换到“代码”选项卡。



现在需要的只是在处理程序的主体中键入以下代码：

PascalScript：

```
如果复制（<Customers。 “Company”>，1,1）='A'则
  MasterData1.Visible: = True
其他
  MasterData1.Visible: = False;
```

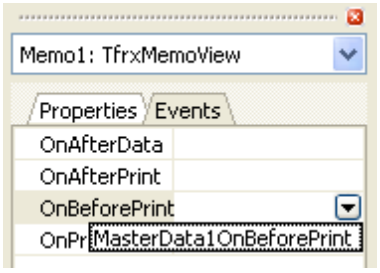
C ++脚本：

```
if（复制（<Customers。 “Company”>，1,1）==“A”）
  MasterData1.Visible = true;
其他
  masterdata1可视=假；
```

运行报表并确保脚本正常工作：

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

我们来解释几件事。可以将一个处理程序分配给多个对象的事件
- “Sender”参数显示哪个对象已启动事件。要将现有处理程序分配给事件，请将其直接键入对象检查器，或从下拉列表中选择它：



可以轻松删除指向处理程序的链接 – 在对象检查器中选择已分配的处理程序，然后按“删除”键。

6.11 在组标题中打印组和

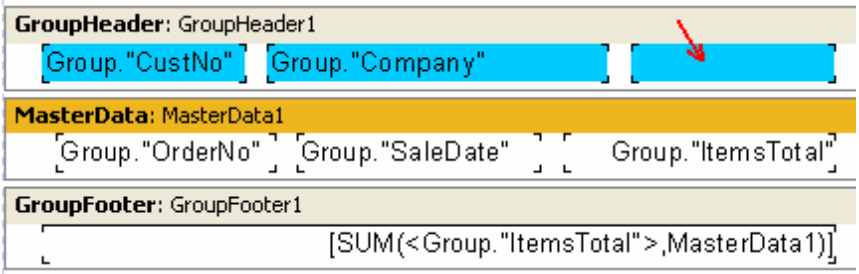
此方法经常使用并且需要使用脚本，因为只有在处理了组中的所有记录之后才知道组和的值。要在组头中显示总和（即在将组输出到报表之前），使用以下算法：
- 打开两遍报表选项（“报表>选项...”菜单项）
- 在第一遍中，计算每个组的总和并将它们保存在一个数组中
- 在第二遍中，从数组中提取值并将其显示在组头中

让我们展示两种执行此任务的方法。首先在Delphi中创建一个新项目，在表单上放置“TQuery”，“TfrxReport”和“TfrxDBDataSet”组件，并按如下方式设置它们：

查询1：
DatabaseName = 'DBDEMOS'
SQL =
*从客户，订单中选择**
其中orders.CustNo = customer.CustNo order
by customer.CustNo, orders.OrderNo

frxDBDataSet1:
DataSet = Query1
UserName ='Group'

打开设计器并将数据源连接到报表。在报表的设置中启用双遍（“报表>选项...”菜单项）。在报表中添加两个段：“GroupHeader”和“MasterData”。在“GroupHeader”频段的编辑器中，输入Group。“CustNo”数据字段。将数据段连接到“Group”数据源，然后按以下方式排列一些对象：



为了显示总和，我们在设计中使用了箭头对象（在我们的例子中，它被命名为“Memo8”）。

第一种方法。

我们将使用“TStringList”类作为存储和的数组 - 我们将数值存储为字符串.StringList中的第一项将对应于第一组的总和，如下所示。整数变量（我们将在打印每组后增加）用于计算组的索引号。

所以我们的脚本将如下所示：

PascalScript:

```
是
  清单：TStringList；我：
  整数；

procedure frReport1OnStartReport (Sender: TfrxComponent) ;
开始
  List: = TStringList.Create;
结束；

procedure frReport1OnStopReport (Sender: TfrxComponent) ;
开始
  自由列表；
结束；

程序Page1OnBeforePrint (发件人：TfrxComponent) ；
开始
  i := 0;
结束；

procedure GroupHeader1OnBeforePrint (Sender: TfrxComponent) ;
开始
  如果是Engine.FinalPass那么
    Memo8.Text: ='Sum: '+ List [i];
结束；

procedure GroupFooter1OnBeforePrint (Sender: TfrxComponent) ;
开始
  如果不是Engine.FinalPass那么
    List.Add (FloatToStr (SUM (<组“ItemsTotal”>, MasterData1) ) ) ；公司
   (i) ;
结束；开始

结束。
```

C ++脚本:

```
TStringList列表;
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 (<组"ItemsTotal">, MasterData1) ) ) ; 我++;
}

{

}
```

脚本中的过程名称显示了我们使用过的事件。它们是："报表。 OnStartReport", " Report.OnStopReport", " Page1.OnBeforePrint", " GroupHeader1。 OnBeforePrint"和" GroupFooter1.OnBeforePrint"。前两个事件分别在报表的开头和结尾调用。要为这两个事件创建处理程序，请在对象检查器的下拉列表中选择"报表"对象，其属性将显示在对象检查器中。然后切换到对象检查器的"事件"选项卡并创建处理程序。

为什么我们不在脚本的主程序中创建"List"变量？我们在"OnStartReport"事件中创建它，因为在报表完成后应该销毁动态创建的变量。在"OnStartReport"事件中创建动态变量并在"OnStopReport"事件中销毁它们是合乎逻辑的。在其他情况下（当脚本完成时不需要释放内存时）可以使用脚本的主过程来初始化变量。

"List"变量的创建和销毁是直截了当的。现在让我们看看脚本是如何工作的。在页面的开头，当前组的计数器（变量"i"）被重置为零，并且在每个组打印后（在"GroupFooter1.OnBeforePrint"事件中）递增。在计数器之前，计算的总和将添加到此事件中的"列表"中

递增。“GroupHeader1.OnBeforePrint”事件在第一次传递期间不执行任何操作（如果是“Engine.FinalPass”条件），但在第二次传递期间（当“List”填充了值时），从“List”中检索对应于当前组的总和“并输出到”Memo8“对象以显示组头中的总和。在完成的报表中，它显示如下：

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

这个算法很简单。但是，它可以简化。

第二种方法。

我们将使用报表变量的集合作为存储组和的数组。请记住，可以通过“获取”和“设置”功能访问报表变量。使用这些函数还可以使我们不必显式创建和销毁这些变量。我们的脚本如下：

PascalScript：

```
procedure GroupHeader1OnBeforePrint (Sender: TfrxComponent) ;
开始
  如果是Engine.FinalPass那么
    Memo8.Text: =’Sum: ’+ Get (<Group。 “CustNo”>) ;
结束；

procedure GroupFooter1OnBeforePrint (Sender: TfrxComponent) ;
开始
  设置 (<Group。 “CustNo”>, FloatToStr (SUM (<Group。
    “ItemsTotal”>, MasterData1) ) ) ;
结束；开始

结束。
```

C ++脚本：

```
void GroupHeader1OnBeforePrint (TfrxComponent Sender)
{
  if (Engine.FinalPass)
    Memo8.Text =“Sum: ”+ Get (<Group。 “CustNo”>) ;
}
```

```
void GroupFooter1OnBeforePrint (TfrxComponent Sender)
{
  设置 (<Group。 “CustNo”>, FloatToStr (SUM (<Group。
    “ItemsTotal”>, MasterData1) ) ) ;
}

{

}
```

如您所见，此脚本稍微简单一些。“GroupFooter1.OnBeforePrint”处理程序中的代码设置具有从客户端编号派生的名称的变量的值（或者可以使用明确标识客户端的任何其他标识符，例如<组。“公司”>）。如果没有已存在该名称的变量，则脚本会自动创建它；否则，如果确实存在则更新其值。在“GroupHeader1.OnBeforePrint”处理程序中，检索适当变量的值。

6.12 “OnAfterData”事件

在报表对象填充了与其连接的源的数据后，将触发此事件。使用此事件可以访问DB字段值或对象中包含的表达式。此值放在“Value”系统变量中，该变量仅在此事件中可用。因此，如果两个“Text”对象包含表达式[Table1。“Field1”]和[<Table2。“Field1”>
+ 10]这些表达式的值可以通过引用对象的“Value”变量来使用：

PascalScript：

```
如果值> 3000那么
  Memo1.Color: = clRed
```

C ++脚本：

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

这比写这样的东西简单：

PascalScript：

```
if <Table1。“Field1”>> 3000然后
  Memo1.Color: = clRed
```

C ++脚本：

```
if (<Table1。“Field1”>> 3000)
  Memo1.Color = clRed;
```

使用“Value”而不是表达式可以为。编写一个多用途处理程序

“OnAfterData”事件可以连接到多个对象。

请注意其他内容 - 如果对象包含多个表达式（例如'[expr1] [expr2]'），则它是传递给“Value”变量的最后一个表达式的值。

“OnAfterData”事件非常适合计算对象的高度和宽度，例如“文本”。也就是说，如果脚本中需要包含表达式的拉伸“Text”对象的确切高度，则可以在“OnAfterData”事件中使用此代码：

PascalScript：

```
是
  MemoHeight: 扩展;

开始
  MemoHeight: = TfrxMemoView (发件人) .CalcHeight;

结束;
```

C ++脚本：

```
浮动MemoHeight;
MemoHeight = TfrxMemoView (发件人) .CalcHeight;
```

如果在“OnBeforePrint”事件中使用此代码，则结果将是在计算表达式之前包含表达式的对象的高度，而不是打印时的实际值。

6.13 服务对象

除了普通的报表对象（如页面，段，“文本”和其他对象）之外，还有一些其他系统对象可以在脚本中使用。它们可能有助于报表构建的管理。前一章中使用的“Engine”对象就是这样一个对象，所有这些对象都列在这里：

- 报表：“报表”对象
- 引擎：报表“引擎”的链接
- 大纲：预览报表中“报表树”元素的链接让我们看一下这些对象。

6.13.1 “报表”对象

此对象表示当前报表的链接。在“报表树”窗口中选择“报表”元素时，可以看到此对象的属性。

方法：

方法	描述

功能Calc (const Expr: String) : 变体	返回“Expr”的值，这是一个表达式，例如， Report.Calc ('1 + 2') 返回 “3” 任何有效的fastReport表达式都可以在参数中传递
函数GetDataSet (const Alias: String) : TfrxDataSet	返回具有指定名称的数据集 数据集必须出现在报表的数据集列表中（“报表>数据...”对话框）

6.13.2 “引擎”对象

这是最有用和最有趣的对象，它代表了引擎的链接（FastReport的核心，它管理报表构建）。通过使用引擎的属性和方法，可以创建异乎寻常的报表设计布局

此对象的方法和属性。

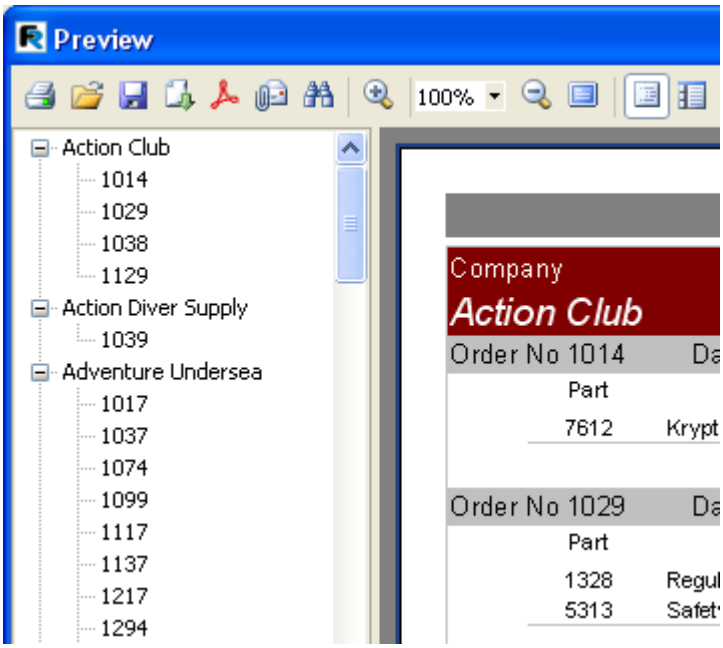
属性	类型	描述
CurColumn	整数	多列报表中当前列的索引可以将值分配给此属性。
CurX	扩展	X轴上的当前打印位置 可以为此属性分配值。
居里	扩展	Y轴上的当前打印位置 可以为此属性分配值。
DoublePass	布尔	如果报表是类似于Report.EngineOptions.DoublePass的两遍，则等于 “True”。
FinalPass	布尔	在两遍报表的最后一次传递中等于“真”
依赖	扩展	可打印区域的高度，以像素为单位
页宽	扩展	可打印区域的宽度（以像素为单位）
开始日期	TDateTime类型	报表运行的时间 与<Date>系统变量相同
开始时间	TDateTime类型	报表运行的时间 与<Time>系统变量相同
总页数	整数	报表中的页数 与<TotalPages>系统变量相同 如果使用此变量，则报表应为两遍
SecondScriptcall	布尔	flag返回事件的'repeat-call'状态（在某些情况下，可以在分组期间重复调用事件） 如果为True则脚本已被调用

方法：

方法	描述
过程AddAnchor (const Text: String)	将“Text”添加到锚点列表中，请参阅下面的内容
程序NewColumn	在最后一列自动分页后，在多列报表中创建一个新列 插入
程序NewPage	创建一个新页面（分页符）
程序ShowBand (Band: TfrxBand)	显示具有指定名称的段 显示段后，“CurY”位置自动递增
功能FreeSpace: 扩展	返回页面上剩余的空白高度（以像素为单位）。
function GetAnchorPage (const Text: String): Integer	返回已放置指定锚点的页码

6.13.3 “大纲”对象

此对象表示预览报表中的“报表树”控件元素。



“大纲”显示完成报表的树状结构。单击任何树节点时，预览窗格将跳转到显示此节点的页面。要显示“大纲”，它应该是

通过单击预览窗口工具栏中的按钮或将“Report.PreviewOptions.OutlineVisible”属性设置为True来启用。也可以在那里设置“Outline”的宽度（以像素为单位）：“Report.PreviewOptions.OutlineWidth”。

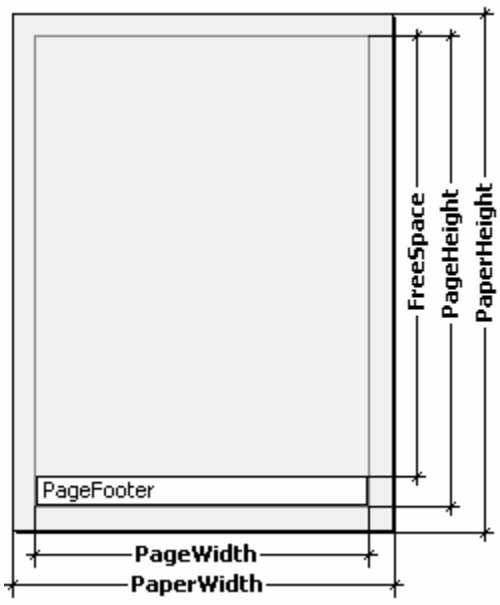
“大纲”方法是：

方法	描述
过程AddItem (const Text: String)	在当前树位置添加具有“文本”名称的元素，页面上的当前报表页面和位置被链接 对元素
程序LevelRoot	将树中的当前位置移动到根级别
程序LevelUp	将树中的当前位置向上移动一级

6.14 使用“引擎”对象

我们已经说过，“Engine”对象代表报表的引擎，它管理报表的构建。可以使用引擎的属性和方法来管理在页面上排列band的过程。首先是一些理论。

下图显示了报表页面的各个维度。



页面的物理尺寸是“PaperWidth”和“PaperHeight”属性，在选择页面时在对象检查器中可见。因此A4页面的尺寸为210 x 297mm。

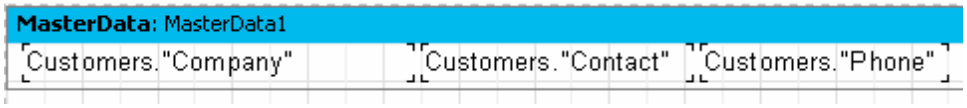
“PageWidth”和“PageHeight”是可打印区域的尺寸，通常小于页面的物理尺寸。可打印区域的大小取决于报表页面属性“LeftMargin”，“TopMargin”，“RightMargin”和“BottomMargin”。可打印区域的大小（以像素为单位）由“Engine.PageWidth”和“Engine.PageHeight”函数返回。

最后，“FreeSpace”是页面上可用空间的高度。如果有一个“页脚”段

在页面上，计算“FreeSpace”时会考虑其高度。“Engine.FreeSpace”函数以像素为单位返回此高度。请注意，在显示下一个段后，页面上的可用空间会减少，这在计算“FreeSpace”时会被考虑在内。

报表页面是如何构建的？只要有足够的可用空间，FastReport核心就会在页面上显示段。当没有剩余可用空间时，打印“页脚”段（如果需要）并创建一个新的空白页。如上所述，在显示下一个段后，可用空间的高度减小。此外，显示下一个段从当前位置开始，该位置由X轴和Y轴上的坐标定义。当前位置分别由“Engine.CurX”和“Engine.CurY”返回。打印完下一个段后，CurY自动增加打印段的高度。创建新页面后，“CurY”等于“0”。打印多列报表时，“CurX”会更改。

“Engine.CurX”和“Engine.CurY”不仅可用于阅读，还可用于书写。这意味着可以通过递增或递减这些值来移动段。例如，在类似于此的报表中：



它可以通过以下方式打印：

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

这是通过为段的“OnBeforePrint”事件编写处理程序来实现的：

PascalScript：

```
procedure MasterData1OnBeforePrint (Sender: TfrxComponent) ;
开始
  Engine.CurX: = Engine.CurX + 5;
结束;
```

C ++脚本：

```
void MasterData1OnBeforePrint (TfrxComponent Sender)
{
  engine.curx = engine.curx + 5;
}
```

更改“CurY”可以使段重叠，例如：



通过此脚本实现：

PascalScript：

```
procedure MasterData1OnBeforePrint (Sender: TfrxComponent) ;
开始
  Engine.CurY: = Engine.CurY - 15;
结束;
```

C ++脚本：

```
void MasterData1OnBeforePrint (TfrxComponent Sender)
{
  engine.cury = engine.cury - 15;
}
```

“Engine.NewPage”方法在报表中的任何所需位置插入分页符，然后从新输出页面的顶部继续打印。在我们的示例中，可以在打印第二条记录后插入一个break：

PascalScript：

```
procedure MasterData1OnAfterPrint (发件人: TfrxComponent) ;
开始
    如果<Line> = 2则
     Engine.NewPage;
结束;
```

C ++脚本：

```
void MasterData1OnAfterPrint (TfrxComponent Sender)
{
  if (<Line> == 2)
    Engine.NewPage () ;
}
```

请注意，我们使用了“OnAfterPrint”事件（也就是说，在打印段之后）。另请注意，“Line”系统变量返回记录的序号。

“Engine.NewColumn”方法在多列报表中插入分栏符。如果页面上没有剩余的空闲列，则会创建一个新页面。

6.15 锚

Anchor是超链接系统中的一个元素，它可以通过单击（在预览窗口中）跳转到连接到已完成报表对象的任何元素。

可以通过“Engine.AddAnchor”方法设置锚点。 Anchor在报表页面中具有名称和位置。要跳转到具有指定名称的锚点，请在任何报表对象的URL属性中键入以下行：

#AnchorName

要么

#[AnchorName]

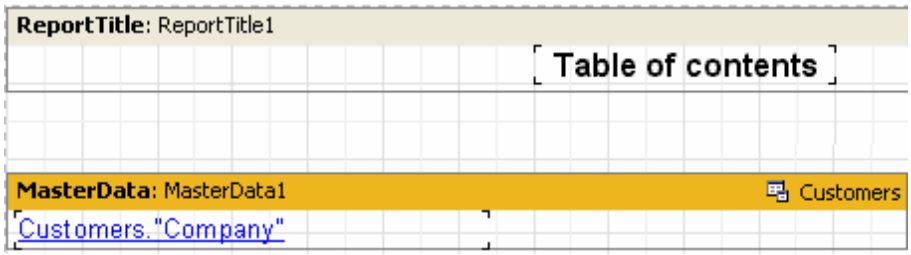
在第二种情况下，FastReport替换表达式的值。

单击该对象将执行跳转到添加锚点的报表部分。

在构建“内容”表时使用锚点，例如使用指向相应章节的链接。让我们在下面的例子中说明这一点。首先，我们需要熟悉的“客户”表。

我们的报表将是一个多页面（有两个设计页面）。我们将“内容”表放在第一页上，将客户列表放在第二页上。单击任何内容行将执行跳转到相应的报表元素。

第一个设计页面：



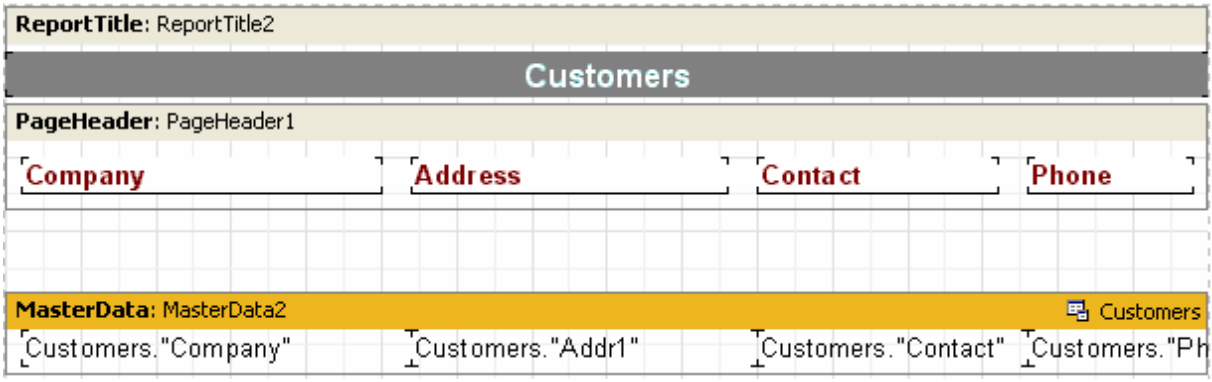
将以下文本放在数据段中包含的“Text”对象的URL属性中

#[客户。“公司”]

并将字体属性设置为蓝色和段下划线，以模拟超链接的外观。第二个

管理员
2018-10-18 09:27:25

应该是锚类型



要添加锚点，请在脚本中创建“MasterData2.OnBeforePrint”事件处理程序：

PascalScript：

```
procedure MasterData2OnBeforePrint (Sender: TfrxComponent) ;
开始
  Engine.AddAnchor (<顾客“公司”。>) ;
结束;
```

C ++脚本：

```
void MasterData2OnBeforePrint (TfrxComponent Sender)
{
  Engine.AddAnchor (<顾客“公司”。>) ;
}
```

管理员
2018-10-18 09:26:02

错误！应该是：Engine.AddAnchor <Customers。“Company”>) ;

这就是所需要的。预览报表并确保“超链接”有效。

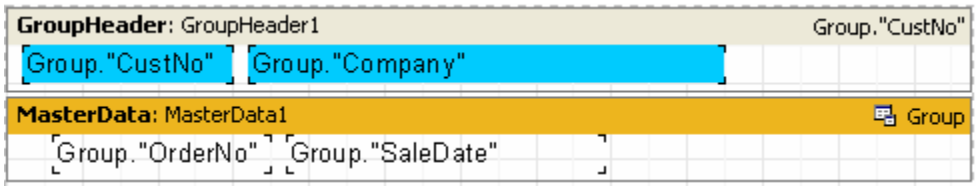
最后要提到的是“Engine.GetAnchorPage”函数。此函数返回添加了相应锚点的页面编号，在创建“内容”表时非常有用。报表必须是两遍的，否则不能使用此功能。

6.16 使用“大纲”对象

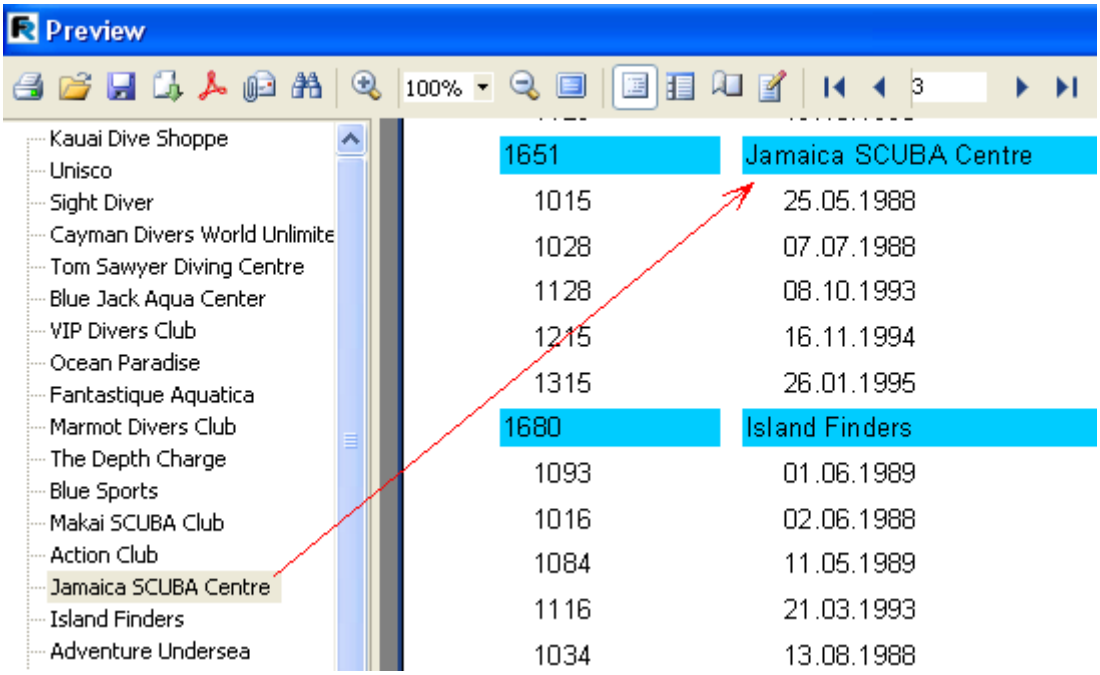
如前所述，“Outline”对象表示可以在预览窗口中显示的报表树。单击树中的元素将跳转到包含相应元素的报表输出页面。没有必要使用脚本来处理“大纲”，因为一些段可以自动创建树。让我们看两个例子，展示如何在段和剧本的帮助下使用“大纲”。

几乎所有band都具有“OutlineText”属性，以包含自动创建树的文本表达式。在创建报表时评估表达式，并在打印段时将其值添加到树中。因此，树中元素的层次结构类似于报表中的段的层次结构，这意味着树将具有与报表中的主要和从属段相对应的主要和从属元素（例如，报表具有两个级别的数据或有组）。我们将使用前面的示例来显示

在具有组的报表中操作树。

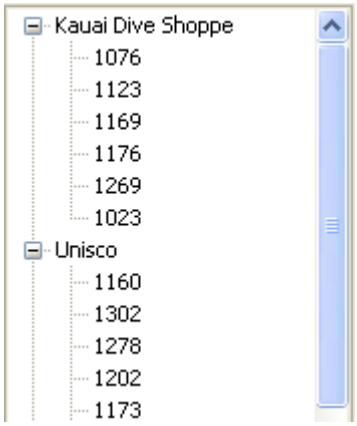


将“GroupHeader1.OutlineText”band属性的值设置为“<Group。”Company“>。”要在预览窗口打开后立即显示树，请设置“PreviewOptions”。 OutlineVisible“Report”对象的属性为True。在预览报表时，您将看到以下内容：



单击树中的任何元素将跳转到报表中的相应页面，窗口顶部将显示所选元素。

让我们将第二级添加到报表树中。只需将“MasterData”段的“OutlineText”属性设置为“<Group。”OrderNo“>”，树将更改为：



很明显，导航到订单号是可能的，并且树中元素的层次结构类似于报表中的元素。

现在我们将创建一个类似的树，但使用脚本而不是“OutlineText”属性。在报表中清除两个段的“OutlineText”属性并创建两个事件处理程序：“GroupHeader1.OnBeforePrint”和“MasterData1.OnBeforePrint”：

PascalScript:

```
procedure GroupHeader1OnBeforePrint (Sender: TfrxComponent) ;
开始
  Outline.LevelRoot;  Outline.AddItem (<组
  “公司”。>) ;
结束;

procedure MasterData1OnBeforePrint (Sender: TfrxComponent) ;
开始
  Outline.AddItem (<组“OrderNo”>) ;
  Outline.LevelUp;
结束；开

始结束。
```

C ++脚本:

```
void GroupHeader1OnBeforePrint (TfrxComponent Sender)
{
  Outline.LevelRoot;  Outline.AddItem (<组
  “公司”。>) ;
}

void MasterData1OnBeforePrint (TfrxComponent Sender)
{
  Outline.AddItem (<组“OrderNo”>) ;
  Outline.LevelUp;
}
```

```
{

}
```

预览报表以确保它的工作方式与之前自动创建树的方式相同。让我们看看脚本是如何创建树的。

“Outline.AddItem”方法将子节点添加到当前树节点，然后使子节点成为当前节点。因此，如果连续多次调用“AddItem”，它将创建此处显示的“梯形图”：

第一条第
二Item3
...

“LevelUp”和“LevelRoot”Outline方法用于控制哪个元素是当前元素。第一种方法将光标移动到位于一级的元素。所以这个脚本：

```
Outline.AddItem ('项目1');
Outline.AddItem ('项目2');
Outline.AddItem ('项目3');
Outline.LevelUp;
Outline.AddItem ('项目4');
```

构造这样的树：

第一条第
二项规定
Item3

这表明“Item4”是“Item2”元素的子元素。另一方面，“LevelRoot”方法将光标向上移动到树的根。例如，脚本：

```
Outline.AddItem ('项目1');
Outline.AddItem ('项目2');
Outline.AddItem ('项目3');
Outline.LevelRoot;
Outline.AddItem ('项目4');
```

构造这个树：

第一条第
二项规定
Item3

了解这一点很清楚报表是如何运作的。在输出每个组标题（公司名称）之前，树的根成为当前元素。之后，输出订单列表，每个订单被添加为公司的子元素。确保所有订单都位于一个订单上

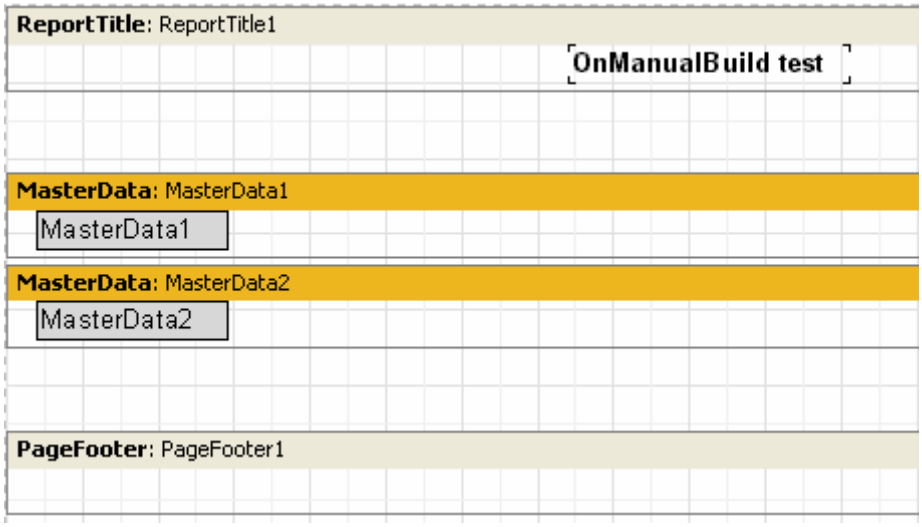
如果没有显示为“梯形图”，则在每次添加订单后调用“Outline.LevelUp”方法，将光标移回公司级别。

6.17 “OnManualBuild”页面事件

FastReport核心通常负责报表构建。它按特定顺序显示报表段，这是数据所需的次数，从而创建完整的报表。有时需要以非标准形式显示FastReport核心无法完成的报表。在这种情况下，可以使用报表设计页面的“OnManualBuild”事件手动构建报表。如果定义了此事件的处理程序，则FastReport核心在需要数据输出时将控制权转移给它。同时FastReport核心自动处理位于页面上的那些段的显示，例如“报表标题”，“页面标题”，“列标题”，“报表页脚”，“页脚”，“列页脚“和”背景“。核心还处理新页面和列的创建。“OnManualBuild”事件处理程序的目的是以用户控制的顺序显示数据段，它们的标题和页脚。

也就是说，“OnManualBuild”处理程序的本质是向FastReport核心发出命令，以便在特定时间显示段。核心完成剩下的工作：只要当前没有空闲空间，处理附加到事件的脚本等，它就会创建新页面。

让我们使用一个简单的例子演示一个处理程序。此报表有两个未连接到数据的主数据段：



处理程序将以交替顺序显示这些段（每个段六次）。显示六个段后，将插入一个小间隙。

PascalScript:

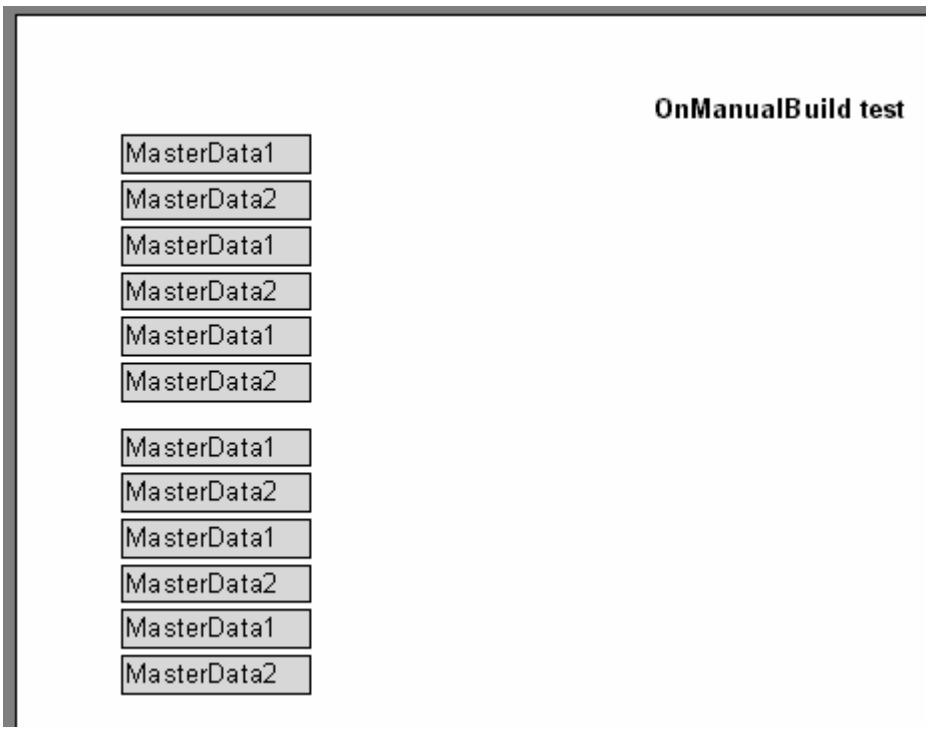
```
程序Page1OnManualBuild（发件人：TfrxComponent）；
是
  我：整数；
```

```
开始
  对于i: = 1到6确实开始
    {show two bands} Engine.ShowBand
     (MasterData1) ;  Engine.ShowBand
     (MasterData2) ;
    {做出差距}
    如果我= 3那么
      Engine.CurY: = Engine.CurY + 10;
  结束;
结束;
```

C ++脚本:

```
void Page1OnManualBuild (TfrxComponent Sender)
{
  int i;

  for (i = 1; i <= 6; i ++)
  {
    //显示两个段Engine.ShowBand
     (MasterData1) ;  Engine.ShowBand
     (MasterData2) ;
    //缩小差距
    if (i == 3)
      engine.cury = engine.cury + 10;
  }
}
```



以下示例显示相同的段，第二个副本向右移动。

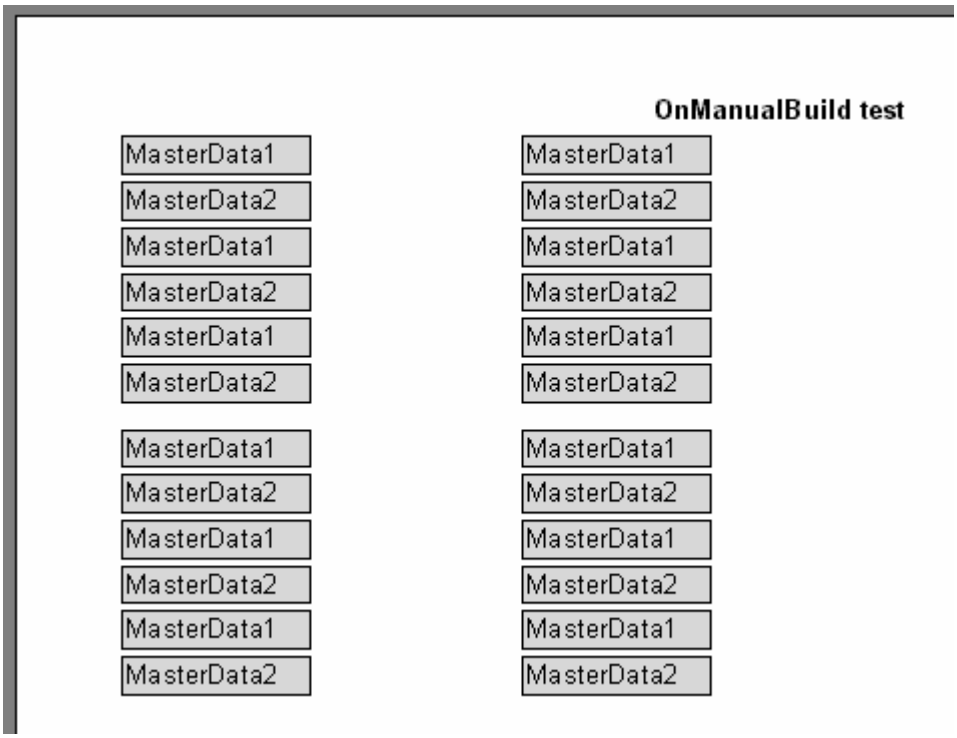
PascalScript:

```
程序Page1OnManualBuild (发件人: TfrxComponent) ;
是
  我, j: 整数;  SaveY:
  扩展;
开始
  SaveY: = Engine.CurY; 对
  于j: = 1到2确实开始
    对于i: = 1到6确实开始
      Engine.ShowBand  ( MasterData1 )  ;
      Engine.ShowBand (MasterData2) ; 如
      果我= 3那么
        Engine.CurY: = Engine.CurY + 10;
    结束;
    Engine.CurY: = SaveY;
    Engine.CurX: = Engine.CurX + 200;
  结束;
结束;
```

C ++脚本:

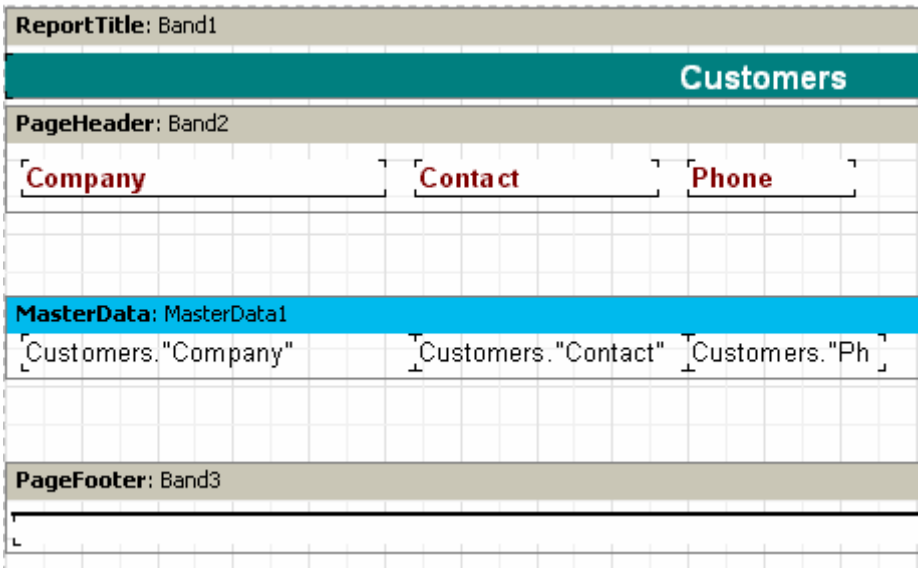
```
void Page1OnManualBuild (TfrxComponent Sender)
{
  int i, j; 延长保存;

  悟性= 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 =理解;
    engine.curx = engine.curx + 200;
  }
}
```



如您所见，在这些示例中，我们仅控制数据段的输出。其余的段（在我们的例子中是“报表标题”）是自动输出的。

最后，我们将展示如何使用“OnManualBuild”事件构建一个段有“客户列表”（我们之前已经显示过几个版本）的报表。这次将数据段连接到数据源。



并使用此脚本：

PascalScript：

```
程序Page1OnManualBuild (发件人: TfrxComponent) ;
是
  DataSet: TfrxDataSet;
开始
  DataSet: = MasterData1.DataSet;
  DataSet.First;
  而不是DataSet.Eof开始
    Engine.ShowBand (MasterData1) ;
    DataSet.Next;
  结束;
结束;
```

C ++脚本:

```
void Page1OnManualBuild (TfrxComponent Sender)
{
  tfrxdataset数据集;

  DataSet = MasterData1.DataSet;
  DataSet.First () ;
  而 (! DataSet.Eof)
  {
    Engine.ShowBand (MasterData1) ;
    DataSet.Next () ;
  }
}
```

预览报表以确保脚本生成与标准报表相同的报表。请注意我们如何获得数据集的链接 - 我们使用以下代码将数据集变量连接到数据源:

```
DataSet: = MasterData1.DataSet;
```

如果MasterData频段未连接到数据源，则可以通过以下方式链接到所需数据源:

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

当然，我们感兴趣的数据源必须在菜单"Report> Data ..."对话框中启用。

6.18 在脚本中创建对象

可以使用脚本将新对象添加到报表中。让我们用一个简单的例子来说明这是如何完成的。创建一个空白报表，并在脚本的主过程中输入以下代码:

PascalScript:

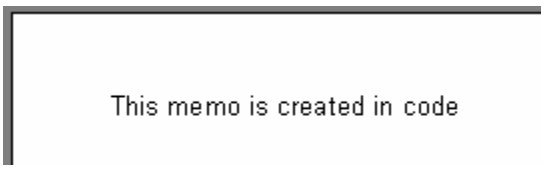
```
是
  段: TfrxReportTitle;
```

```
  备注: TfrxMemoView;
开始
  Band: = TfrxReportTitle.Create (Page1) ;
  Band.Height: = 20;
  备注: = TfrxMemoView.Create (段) ;
  Memo.SetBounds (10,0,100,20) ;
  Memo.Text: =’这个备忘录是用代码创建的’;
结束。
```

C ++脚本:

```
TfrxReportTitle段;
TfrxMemoView备忘录;
{
  Band = TfrxReportTitle.Create (Page1) ;
  Band.Height = 20;
  备注= TfrxMemoView.Create (段) ;
  Memo.SetBounds (10,0,100,20) ;
  Memo.Text =“此备忘录是用代码创建的”;
}
```

预览报表:



请注意，我们没有销毁我们在此示例中创建的FastReport对象。这不是必需的，因为报表完成后，Delphi应用程序会自动销毁FastReport对象。另请注意，当我们在脚本中创建标准Delphi对象（例如TStringLists）时，我们还必须在脚本中销毁它们，因为这不会由应用程序自动完成。

章节 7

交叉表报表

这种报表具有表格结构，这意味着它由行和列组成。在设计时，不知道输出表将具有多少行和列。这就是为什么报表不仅会向下发展（如前面所述的报表类型），还会横向发展。交叉表报表的典型示例如下所示。

我们来看看表中的元素：

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

在图示中，我们看到一个包含两行（行）和四列的表，其中“a”和“b”是行标题，“1”，“2”，“3”和“4”是列标题，“ a1“.....”a4“和”b1“.....”b4“是细胞。要构建这样的报表，我们只需要一组数据（来自查询或表），它有三个字段并包含以下值：

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

您可以看到第一个字段包含一个行号，第二个字段包含一个列号，第三个字段包含指定行和列交叉处的单元格内容。输出报表时，FastReport在内存中创建一个表并用数据填充它。因此，表格会动态扩展，创建尚不存在的行和列。

标题可以出现在多个级别，如下所示：

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

在此示例中，列的数字或索引是复合的，即它由两个值组成。此报表由以下数据生成：

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

这里第一个字段包含行索引，如前所述，第二个和第三个字段包含列索引，最后一个字段包含单元格值。在处理具有复杂标题的交叉表数据时，查看FastReport如何构建内存表：

	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

从此内存表输出报表时，FastReport将连接具有相同值且位于同一级别的标题单元格。

这是一个更复杂的交叉表报表，包含中间和总计：

	10			20			Total
	1	2	Total	1	2	Total	
a	a10.1	a10.2	a10.1+a10.2	a20.1	a20.2	a20.1+a20.2	sum(a)
b	b10.1	b10.2	b10.1+b10.2	b20.1	b20.2	b20.1+b20.2	sum(b)
Total	a10.1+ b10.1	a10.2+ b10.2	a10.1+b10.1+ a10.2+b10.2	a20.1+ b20.1	a20.2+ b20.2	a20.1+b20.1+ a20.2+b20.2	sum(a)+ sum(b)

此报表源自与以前相同的数据。以新颜色突出显示的单元格中的值将自动计算，并且不会出现在原始数据集中。

7.1 构建交叉表报表

现在让我们从理论转向实践。我们将构建一份简单的交叉表报表，显示员工四年的工资。为此，我们需要“crosstest”表，它位于FastReport“DEMOS \ MAIN”文件夹中。该表包含如下数据：

姓名年份 薪水
安 1999 3300
本 2002 2000
....

在Delphi中创建一个新项目，在表单上放置“TTable”，“TfrxDBDataSet”和“TfrxReport”组件并设置它们的属性：

表格1：
DatabaseName ='c: \ Program Files \ FastReport 4 \ Demos \ Main' TableName ='crosstest.db'

当然，DatabaseName属性必须与FastReport的路径相对应

安装文件夹!

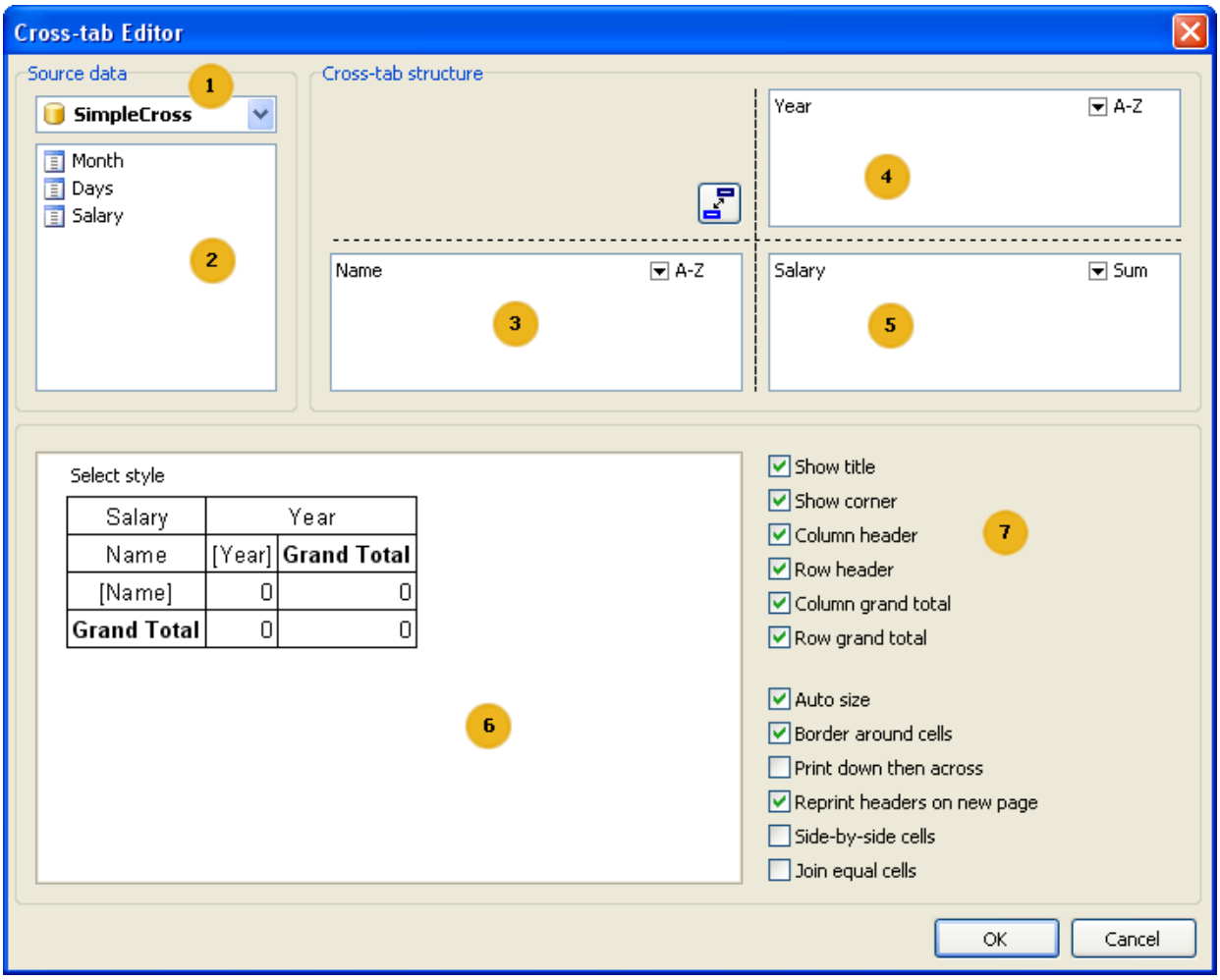
frxDBDataSet1:
数据=表
UserName = 'SimpleCross'

FastReport组件面板中的“TfrxCrossObject”组件用于设计交叉表报表。只需将其放在Delphi表单上，就不需要更改任何属性。当编译Delphi应用程序时，包含交叉表功能的“frxCross”单元被添加到“uses”列表中。

打开报表设计器。首先使用“Report> Data ...”菜单项连接数据源。然后从设计器的对象工具栏中选择“DB cross-tab”对象，并单击设计页面将对象放在那里:



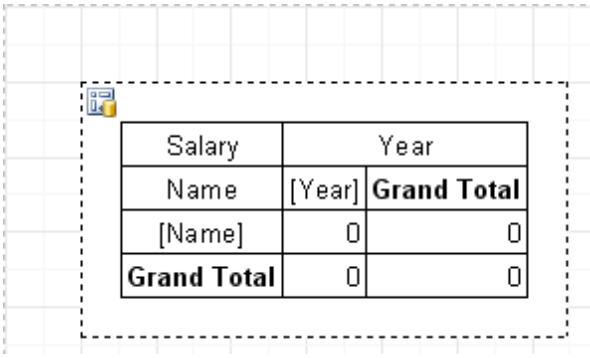
所有设置均使用交叉表编辑器进行。双击对象打开它:



上面显示的项目的关键：

1 – 可用数据源的下拉列表
2 – 所选数据源中的字段列表；此列表中的字段可以拖动到编号为3, 4或5的列表中
3 – 生成行（行）标题的字段列表
4 – 生成列标题的字段列表
5 – 生成表格单元格的字段列表
6 – 表结构预览
7 – 结构选项：显示标题，总数等

您只能在此编辑器中使用鼠标进行更改。对于我们的示例，只需要将列表2中的字段拖到列表3, 4和5（在上图中）。之后，单击OK按钮关闭编辑器。交叉表对象现在显示其结构：



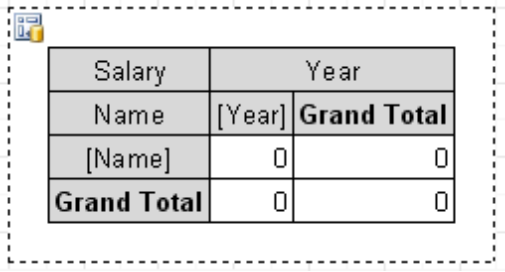
Salary	Year	
Name	[Year]	**Grand Total**
[Name]	0	0
Grand Total	0	0

预览报表时，您将看到类似于此的表格：

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

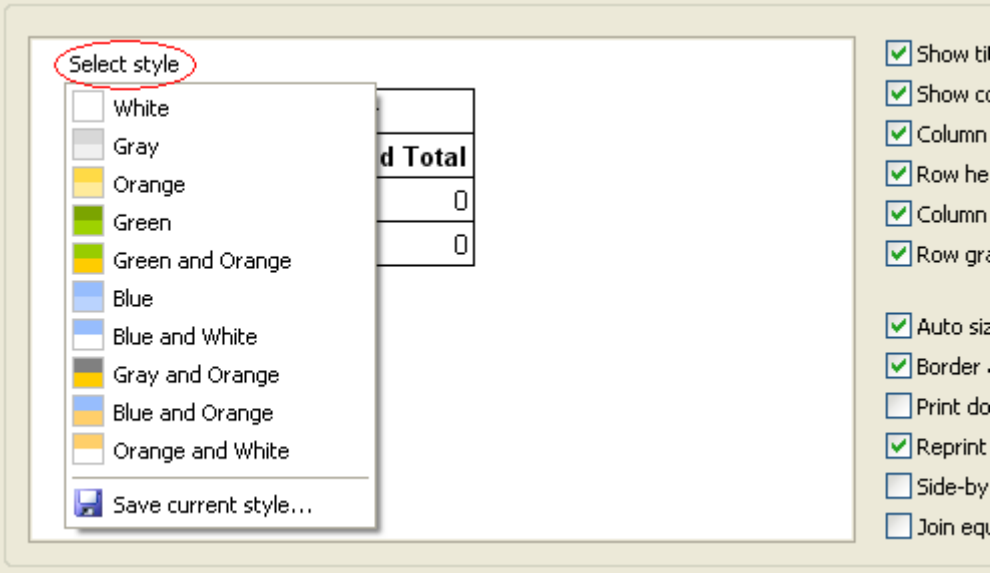
7.2 改变外观

让我们修改交叉表对象的外观。我们要做的第一件事是更改标题颜色并显示“总计”而不是“总计”。这很容易 – 将标题颜色更改为灰色依次单击“年”，“名称”和“总计”单元格，然后使用工具栏上的背景按钮/选择灰色。

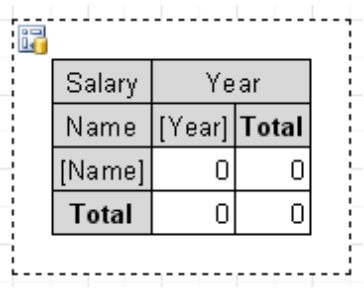


Salary	Year	
Name	[Year]	**Grand Total**
[Name]	0	0
Grand Total	0	0

我们还可以使用一组预定义的样式。这些在交叉表编辑器中可用 – 单击“选择样式”并选择一个。

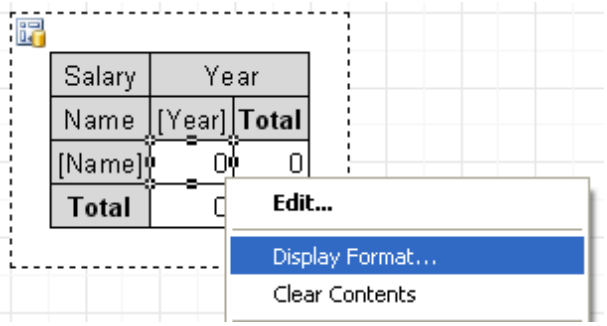


要更改两个“Grand Total”文本，请双击每个单元格，这将打开熟悉的文本编辑器，我们可以在其中键入 “Total”：



Salary	Year	
Name	[Year]	**Total**
[Name]	0	0
Total	0	0

要格式化货币值，请选择第一个单元格（在我们的示例中为[Name]和[Year]的交集），右键单击以显示上下文菜单，然后选择 “Display Format ...”：



选择所需的格式并关闭格式编辑器。所有这些都会产生报表：

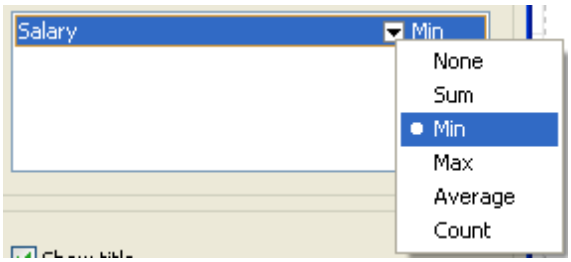
Salary	Year				
Name	1999	2000	2001	2002	**Total**
Ann	$3 300,00	$2 700,00	$3 100,00	$1 700,00	$10 800,00
Ben	$3 900,00	$2 100,00		$1 800,00	$7 800,00
Catherine	$6 100,00	$3 200,00			$9 300,00
Den		$3 999,00	$8 100,00		$12 099,00
Total	$13 300,00	$11 999,00	$11 200,00	$3 500,00	$39 999,00

7.3 使用功能

在我们的示例中，我们在“总计”行中看到每个员工四年以上工资的总和。可以使用以下任何聚合函数：

和 – 值之和MIN 最
小值MAX 最
大值AVG 平
均值
计数 – 值的数量

我们在示例中使用“MIN”函数。打开交叉表编辑器，在列表5（“Salary”字段项）中单击向下箭头。



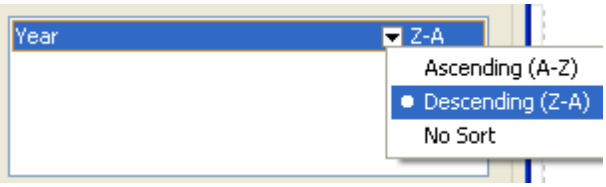
从下拉列表中选择“MIN”功能。现在我们可以将总单元格中的文本从“总计”更改为“最小”。
完成的报表如下所示：

	1999	2000	2001	2002	**Minimum**
Ann	3300	2700	3100	1700	**1700**
Ben	4300	2400		2000	**2000**
Catherine	6100	3200			**3200**
Den		3999	8100		**3999**
Minimum	**3300**	**2400**	**3100**	**1700**	**1700**

7.4 排序值

默认情况下，行和列按数字或字母顺序按升序排列，具体取决于数据类型。可以为行和列单独设置排序模式。排序模式是：“按升序排列”，“按降序排列”和“不执行排序”。如果没有排序，则行/列将以默认数据库顺序显示。

让我们在示例中更改列排序。让年份按递减顺序排列。要执行此操作，请打开交叉表编辑器，选择“年”列元素并通过单击向下箭头并选择降序来更改排序模式：



关闭编辑器并预览报表。它看起来像这样：

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

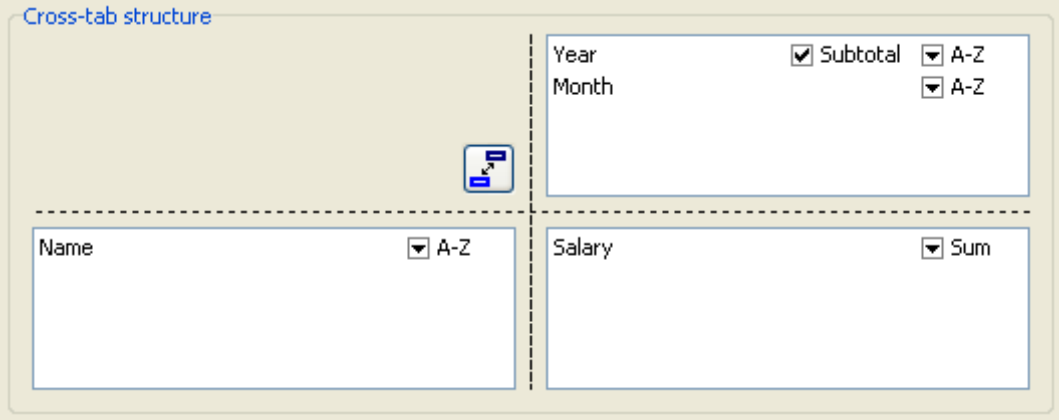
7.5 带有复合标题的表

我们之前的示例包含每行和单列标题的单个值。让我们看一下具有多个值的复合头的交叉表设计。源数据具有以下格式的数据：

名称	年	月	天	薪
安	1999	2	3	1000
本	2002	1	5	2000
….				

我们分别添加了包含月份编号和工作日数的“月”和“天”字段。可以从这些数据中构建几个不同的报表，例如：每年的员工工资，按月分解。

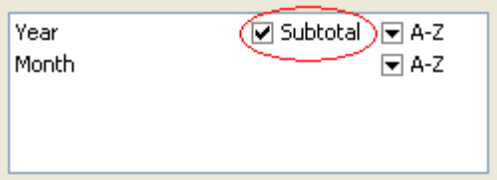
我们会得到什么样的报表？它必须类似于前一个示例中的报表，但将年度数据按月细分。必须以与以前相同的方式设置交叉表对象。这次我们还将“Month”字段拖到列标题列表中，如下所示：



在预览中，我们看到以下报表：

	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**

请注意，FastReport已自动添加一列中间总计，每年后显示一列。通过取消选择“Year”列元素的“Subtotal”标志，可以在交叉表编辑器中关闭它：



另请注意，列标题列表中的最后一个列元素从不具有“小计”标记（包括单个元素的大小写）。在我们的示例中，我们不需要每个月的中间总计，因此可以关闭“小计”标记。

如果使用中间总计还有另一个特征：最好将中间总数作为“年+年总数”而不是“总计”。在报表页面上的交叉表对象中，双击中间的总单元格，然后在文本编辑器中键入：

[价值]总计

在报表构造中，“[Value]”表达式将替换为上面单元格中列标题的实际值：

	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**

7.6 调整单元格宽度

查看前面的插图，很明显FastReport会自动调整单元格宽度，以便更大的单元格值确实适合单元格。然而，在某些情况下，这可能并不理想，因为超宽的列可能看起来很难看。关于这个还能做什么？让我们看看控制单元宽度的三种方法。

控制单元格宽度的最简单方法是在中间总计的文本中添加换行符，即：

[价值]总
计

结果表更紧凑：

	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**

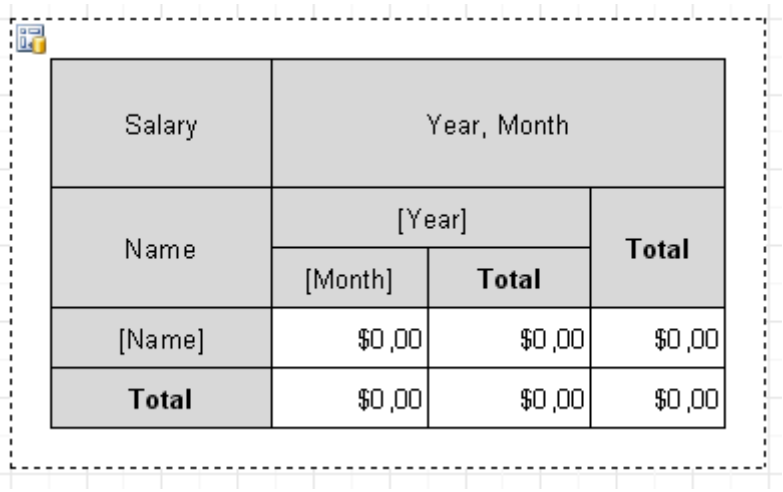
但是，有些情况下很难或不可能手动地断线。因此，交叉表对象具有“MinWidth”和“MaxWidth”属性（指单元格宽度）。这两个属性只能通过对象检查器访问。

默认情况下，“MinWidth”为0，“MaxWidth”为200，这在大多数情况下是足够的。控制单元宽度的第二种方法是根据特殊要求改变这些值。

因此在我们的示例中，我们可以将“MinWidth”和“MaxWidth”都设置为50.这意味着数据单元必须至少为50像素宽，即使单元格值适合较少的像素。对于大单元格值，单元格宽度限制为“MaxWidth”值，单元格中的文本将根据需要中断。所以我们的例子现在看起来像这样：

	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**

控制单元宽度的第三种方法是手动更改单元格宽度。为此，必须将AutoSize属性设置为False。然后可以使用鼠标更改交叉表单元格宽度。在报表页面交叉表对象中，鼠标光标在单元格边框上更改形状，因此允许拖动边框。以下是可以实现的示例：



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

请记住，如果关闭自动调整大小，则交叉表不会自动调整单元格的宽度和高度，您可能会在报表预览中看到类似的内容：

Salary			
Name	1999		
	2	10	11
Ann	$1 000,00		$1 100,00
Ben		$1 900,00	$2 000,00
Catherine		$3 000,00	$3 100,00

如果是这样，那么只需稍微增加单元格宽度。

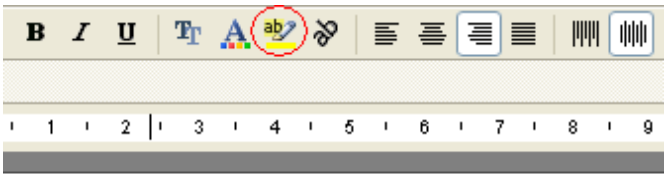
7.7 字体颜色和突出显示

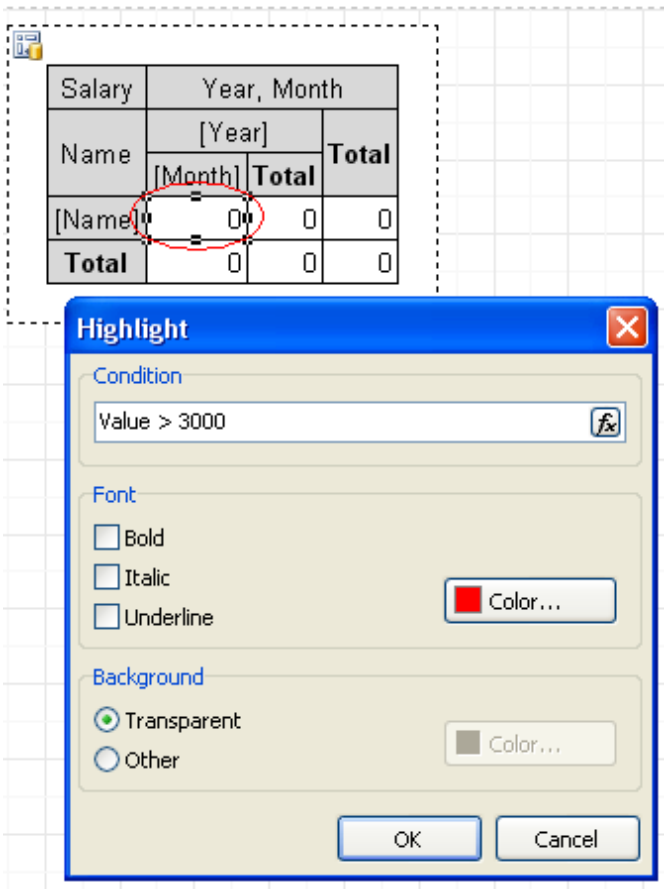
有时需要突出显示值和/或更改字体颜色。我们在组报表示例中查看了这一点，其中我们对“Text”对象使用了条件突出显示。条件突出显示对我们来说也很有用。

要在示例报表中添加突出显示，假设我们需要更改大于3000的值的字体颜色 – 选择表示表格单元格的对象，并通过单击工具栏上的突出显示按钮/来设置突出显示参数。熟悉的突出显示编辑器窗口将打开可以设置此条件的位置：

价值> 3000

并将字体颜色设置为红色：





这就是所需要的。单击“确定”按钮关闭编辑器并预览报表：

	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，如果需要，可以突出显示总值，还可以使用背景和框架颜色按钮突出显示单元格和线条。

7.8 在脚本中管理交叉表

如果上面显示的任何方法无法实现所需的报表，则可以使用报表脚本。“交叉表”对象具有以下事件：

事件	描述
OnAfterPrint	在打印表格后调用event
OnBeforePrint	在打印表之前调用event
OnCalcHeight	在计算表中行的高度之前调用event，事件处理程序可以设置为所需的高度 如果需要隐藏行，则为“0”
OnCalcWidth	在计算表中的列宽之前调用event，事件处理程序可以设置为所需的宽度 如果需要隐藏列，则为“0”
OnPrintCell	在显示表格单元格之前调用event 事件处理程序可以修改单元格设计或其内容
OnPrintColumnHeader	在显示列标题之前调用event 事件处理程序可以修改标题单元格的设计或内容
OnPrintRowHeader	在显示行标题之前调用event 事件处理程序可以修改标题单元格的设计或内容

我们可以在这些事件中使用“交叉表”对象的以下方法：

方法	描述

function ColCount: 整数	返回表中的列数
function RowCount: Integer	返回表中的行数
function IsGrandTotalColumn (Index: Integer) : Boolean	如果'Index'列是总计，则返回“True”
function IsGrandTotalRow (Index: Integer) : Boolean	如果'Index'行是总数，则返回“True”
function IsTotalColumn (Index: Integer) : Boolean	如果“索引”列是小计，则返回“真”
function IsTotalRow (Index: Integer) : Boolean	如果'Index'行是小计，则返回“True”
过程AddValue (const行，列，单元格 : Variant数组)	为表添加一个值

让我们展示如何突出显示第三列（在我们的示例中为“1999年11月”日期）。在报表设计页面上选择交叉表对象，在对象检查器中单击事件选项卡，找到“OnPrintCell”事件并通过双击事件右侧的空列表在代码页上创建处理程序名称。脚本编辑器将显示为您创建的基本声明，然后在声明的空“begin ... end”块中添加所需的代码：

Pascal脚本：

```
过程Cross1OnPrintCell (备注: TfrxMemoView; RowIndex,
  ColumnIndex, CellIndex: Integer; RowValues,
  ColumnValues, Value: Variant) ;
开始
    如果ColumnIndex = 2那么
      Memo.Color: = clRed;
结束;
```

C ++脚本：

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

预览报表时，我们会看到以下内容：

	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**

要突出显示列标题，请以类似方式创建“OnPrintColumnHeader”事件处理程序：

Pascal脚本：

```
过程Cross1OnPrintColumnHeader (备注: TfrxMemoView; HeaderIndexes,
  HeaderValues, Value: Variant) ;
开始
  if (VarToStr (HeaderValues [0]) ='1999') 和
     (VarToStr (HeaderValues [1]) ='11') 然后
    Memo.Color: = clRed;
结束;
```

C ++脚本：

```
void Cross1OnPrintColumnHeader
 (TfrxMemoView备忘录,
Variant HeaderIndexes,
Variant HeaderValues,
Variant Value)
{
  if ( (VarToStr (HeaderValues [0]) ==“1999”) &&
       (VarToStr (HeaderValues [1]) ==“11”) )
  {
    颜色= clred备忘录;
  }
}
```

报表预览现在是：

	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**

这是脚本的工作原理：在表的数据区域中打印单元格之前调用“OnPrintCell”事件处理程序（请注意，表标题中的单元格调用“OnPrintColumnHeader”或“OnPrintRowHeader”处理程序）。“OnPrintCell”处理程序参数包括：指向“文本”的链接

表示单元格的对象（“备注”参数）和单元格的“地址”作为行，列和单元格的位置（如果交叉表包含多级单元格，则单元格相关）作为“RowIndex”，分别为“ColumnIndex”和“CellIndex”参数。参数列表还具有指定为Variants的标题值（“RowValues”和“ColumnValues”参数）和保存单元格内容的“Value”Variant参数。

在我们的示例中，使用“RowIndex”和“ColumnIndex”指定“地址”更容易。列和行的编号从“0”开始，因此“ColumnIndex = 2”指的是第三列。我们还可以通过查看其数据内容来指定正确的列（我们需要1999年的第11个月）：

Pascal脚本：

```
过程Cross1OnPrintCell（备注：TfrxMemoView; RowIndex,
  ColumnIndex, CellIndex: Integer; RowValues,
  ColumnValues, Value: Variant）;
开始
  if（VarToStr（ColumnValues [0]）='1999'）和
     （VarToStr（ColumnValues [1]）='11'）然后
    Memo.Color: = clRed;
结束;
```

C ++脚本：

```
void Cross1OnPrintCell
 （TfrxMemoView备忘录，
int RowIndex, int
ColumnIndex, int
CellIndex,
Variant RowValues,
Variant ColumnValues,
Variant Value）
{
  if （（VarToStr（ColumnValues [0]）==“1999”）&&
       （VarToStr（ColumnValues [1]）==“11”））
  {
    颜色= clred备忘录;
  }
}
```

“RowValues”和“ColumnValues”参数是Variant类型的数组，具有零基数。“0”元素位于表格标题的最高级别，“1”元素位于下一级别，等等。在我们的示例中，“ColumnValues [0]”包含年份，“ColumnValues [1]”包含月份。

为什么需要“VarToStr”功能？这可以防止类型转换期间出错。使用Variant类型时，FastReport尝试自动将字符串转换为数字格式，这反过来可能会在转换“Total”和“Grand Total”列值时导致错误。

在输出列标题单元格期间调用“OnPrintColumnHeader”事件处理程序。参数列表类似于“OnPrintCell”处理程序的参数列表，但在这种情况下，单元格的“地址”（“HeaderIndexes”和“HeaderValues”参数）采用不同的形式。“HeaderValues”参数与“ColumnValues”和“RowValues”中的值保持相同

“OnPrintCell”处理程序。“HeaderIndexes”参数也是Variant类型的值数组，并且包含不同形式的标题单元格的地址：“0”元素是表标题中最高级别的索引，“1”是为了阐明细胞编号的原理，请参考下图：

	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**

在我们的示例中，使用“HeaderValues”参数更容易，但可以使用以下处理程序：

Pascal脚本：

```
过程Cross1OnPrintColumnHeader (备注: TfrxMemoView; HeaderIndexes,
  HeaderValues, Value: Variant) ;
开始
  if (HeaderIndexes [0] = 0) 和 (HeaderIndexes [1] = 2) 然后
    Memo.Color: = clRed;
结束;
```

C ++脚本：

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

7.9 调整行/列大小

可以使用“OnCalcWidth”和“OnCalcHeight：”事件处理程序调整表列和行的宽度和高度。让我们看看如何通过以下示例增加保持1999年日月份的列的宽度。 创建一个“OnCalcWidth”事件处理程序：

Pascal脚本：

```
procedure Cross1OnCalcWidth (ColumnIndex: Integer;
  ColumnValues: Variant; var Width: Extended) ;
开始
  如果 (VarToStr (ColumnValues [0]) ='1999') 和
     (VarToStr (ColumnValues [1]) ='11') 则宽度: =
    100;
结束;
```

C ++脚本:

```
void Cross1OnCalcWidth (int
ColumnIndex, variant
ColumnValues, Extended&
Width)
{
  if ( (VarToStr (ColumnValues [0]) ==“1999”) &&
     (VarToStr (ColumnValues [1]) =“11”) )
    {
      宽度= 100;
    }
}
```

报表变成:

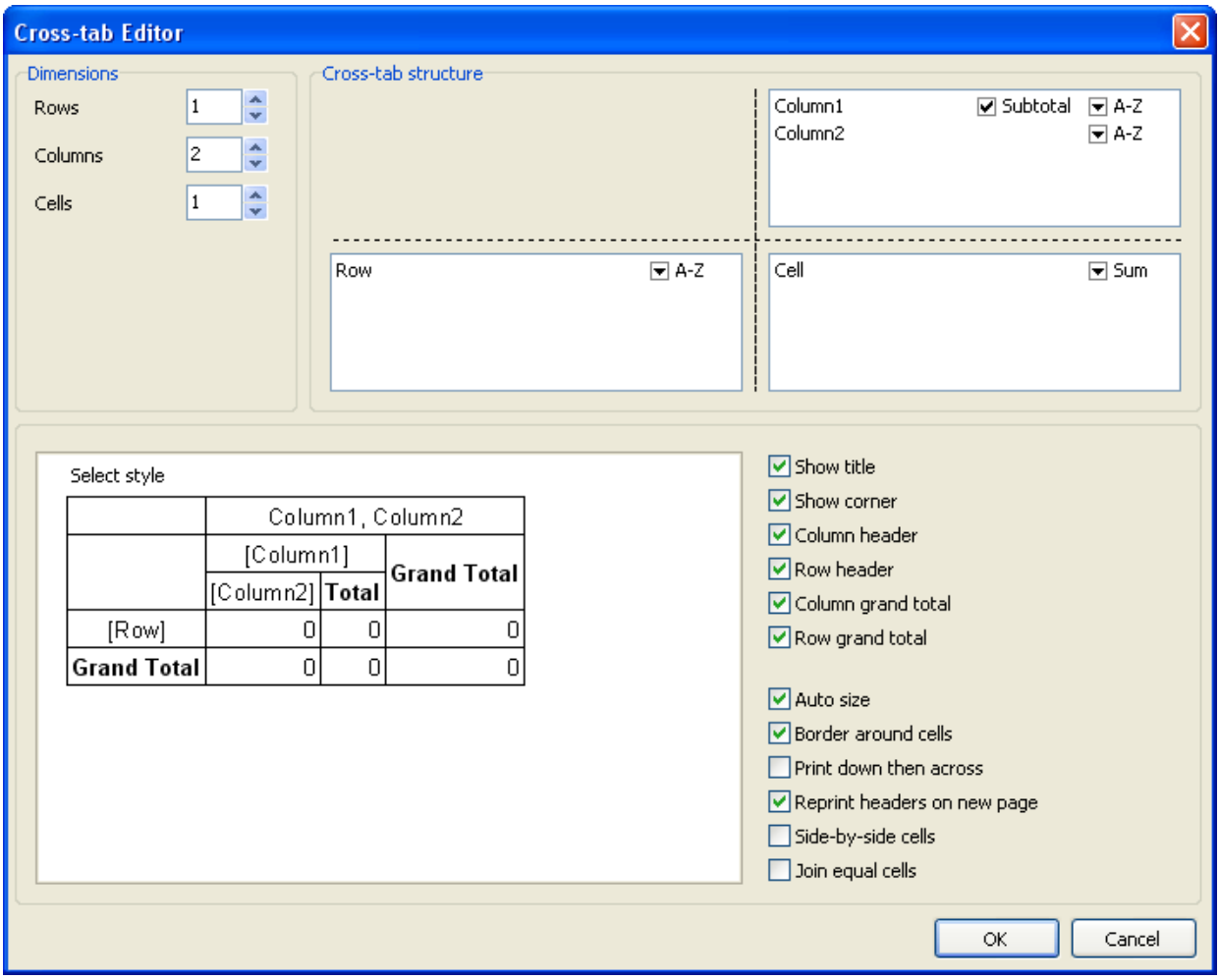
	1999				
	2	10	11	12	**Total**
Ann	1000		1100	1200	**3300**
Ben		2100	2200		**4300**
Catherine		3000	3100		**6100**
Den					**0**
Grand Total	**1000**	**5100**	**6400**	**1200**	**13700**

要在我们的示例中隐藏列，只需将“宽度”设置为零。请注意，不会重新计算总计，因为此时表已填充了值。

7.10 手动填写表格

交叉表对象有两个版本，“DB cross-tab”和“Cross-tab”。到目前为止，我们已经处理了第一个对象，它附加到数据库表中的数据。报表运行后，对象会自动填充。让我们看看第二个对象“交叉表”。

“交叉表”对象未附加到数据库表。因此必须手动填充数据。“交叉表”对象具有与“DB交叉表”对象类似的编辑器，但不同之处在于必须指定表的标题和单元格的维度，而不是由DB字段设置:



让我们通过一个例子展示“交叉表”对象的用法。在报表设计页面上放置一个“交叉表”对象，并设置其属性如上所示：行标题中的级别数为1，列标题中为2，单元格中为1. 让我们填写表格使用“OnBeforePrint”事件处理程序的数据：

PascalScript:

```
程序Cross1OnBeforePrint (发件人: TfrxComponent) ;
开始
  与Cross1一起开始
    的AddValue ([ '安
        '],
    的AddValue ([ '安
        '],
    的AddValue ([ '安
        '],

    的AddValue ([ '奔
        '],

    的AddValue ([ '传
        '],
    的AddValue ([ '传
        '],
  结束;
结束;
```

C ++脚本：

```
void Cross1OnBeforePrint (TfrxComponent Sender)
{
    Cross1.AddValue ([“安”], [2001, 2], [1500]);
    Cross1.AddValue ([“安”], [2001, 3], [1600]);
    Cross1.AddValue ([“安”], [2002, 1], [1700]);

    Cross1.AddValue ([“本”], [2002, 1], [2000]);

  Cross1.AddValue ([“小室”], [2001, 1], [4000]);
  Cross1.AddValue ([“小室”], [2001, 2], [4100]);
}
```

在处理程序中，数据通过“TfrxCrossView.AddValue”方法添加到表中。此方法有三个参数，每个参数都是Variant类型的数组。第一个参数是行值，第二个参数是列值，第三个参数是单元格值。请注意，每个数组中的值的数量必须与对象的设置匹配！在我们的示例中，对象在行标题中有一个级别，列标题中有两个级别，一个单元格级别，因此AddValue方法的Variant数组参数需要一个值用于行，两个值用于列，一个值用于单元格。

在预览时，报表输出为：

	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**

“AddValue”方法也可用于“DB交叉表”对象。这允许插入不是从附加到对象的数据源导出的数据。如果以这种方式添加任何数据，则还会使用数据源中的数据进行汇总。

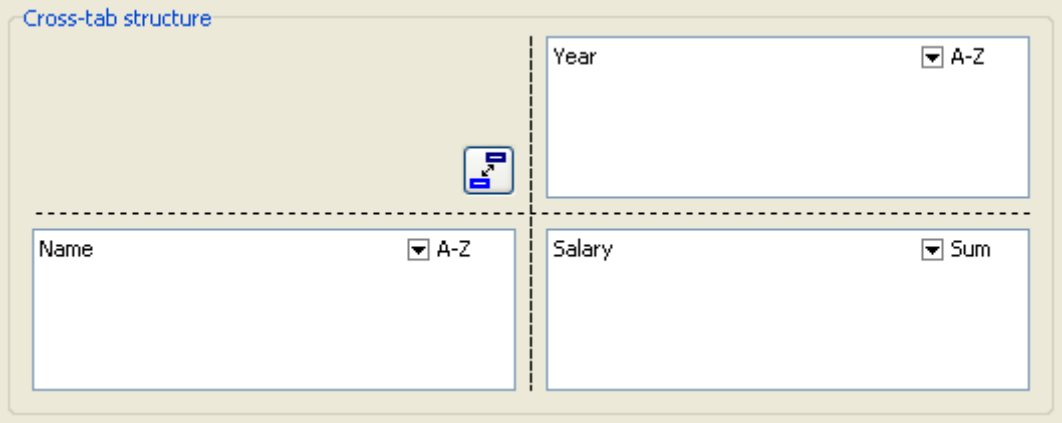
7.11 将外部对象添加到表中

外部对象（如线条，形状，图片）可以放在交叉表中。例如，您可能需要以图形形式显示某些数据。让我们看一个使用形状来显示基本进度条的示例：

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

如果单元格值小于100，则显示深红色条，如果小于3000，则显示黄色;如果大于3000，则显示绿色。

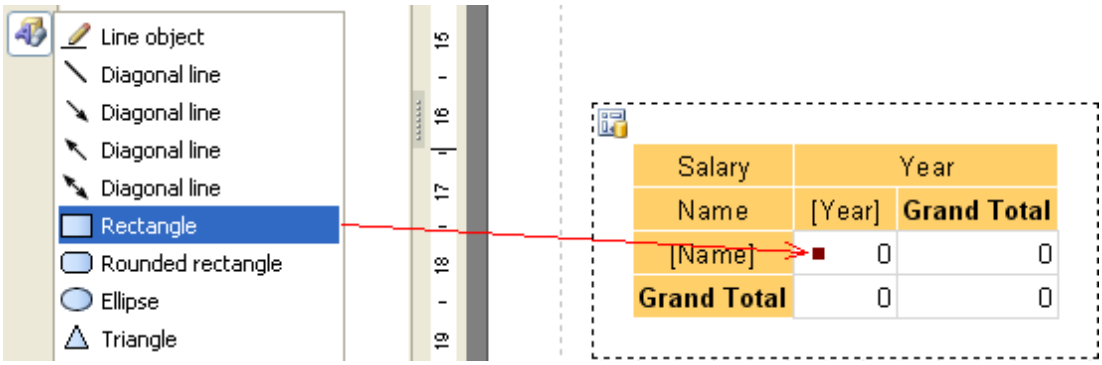
让我们从我们的报表开始。在报表页面上放置一个“DB Cross-tab”对象，并设置如下所示的属性：



关闭“自动调整大小”属性并设置列宽，如下所示：

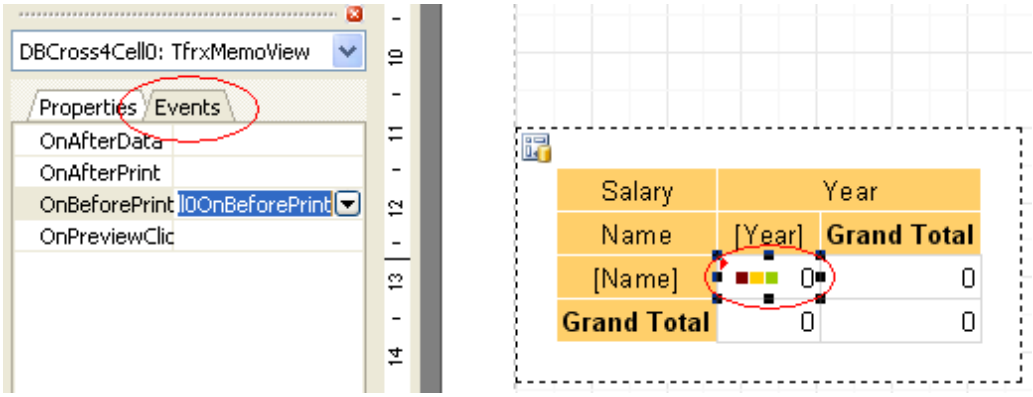
Salary	Year	
Name	[Year]	**Grand Total**
[Name]	0	0
Grand Total	0	0

现在，通过在对象工具栏上选择“矩形”对象并将其放在单元格中，将形状对象添加到表中：



将其高度和宽度更改为0.2厘米并设置其顶部和左侧属性。再添加两个相似的矩形。

现在创建一个脚本，显示正确数量的彩色形状（取决于单元格值）。为此，请选择单元格并创建“OnBeforePrint”事件处理程序：



在事件处理程序中编写以下代码（注意形状名称，以便它们与您的对象匹配）：

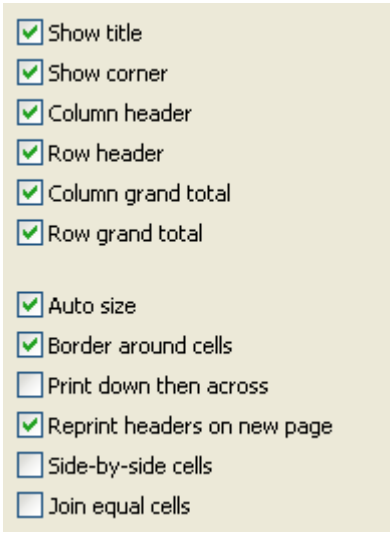
```
procedure DBCross1Cell0OnBeforePrint (Sender: TfrxComponent);
开始
  //'Value'是当前单元格的值
  如果值<100则开始
    //第一个形状对象
    DBCross1Object1.Color: = clMaroon;  // 深红
    //第二个形状对象
    DBCross1Object2.Color: = clWhite;
    //第三个形状对象
    DBCross1Object3.Color: = clWhite;
  结束
  否则，如果值<3000则开始
    DBCross1Object1.Color: = $ 00CCFF;        //黄色
    DBCross1Object2.Color: = $ 00CCFF;
    DBCross1Object3.Color: = clWhite;
  结束
```

```
    别的开
    始
      DBCross1Object1.Color: = $ 00CC98;          //绿色
      DBCross1Object2.Color: = $ 00CC98;
      DBCross1Object3.Color: = $ 00CC98;
    结束;
  结束;
```

这就是全部 - 预览报表，这与本节顶部显示的类似。

7.12 一些有用的设置

让我们看一下交叉表编辑器中提供的其他一些设置。



前六个选项允许您显示或隐藏各种表格元素。

“自动尺寸”选项已众所周知。它允许我们手动设置表格宽度和高度。

“细胞边界”选项允许在细胞元素周围绘制框架。这是一个带边框的示例表（请注意，单元格本身没有边框）：

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

“向下打印”选项确定如何在多个页面上打印表格。以下两个示例显示了此选项的工作原理（请注意页面顺序）：

1) “打印然后跨越”已关闭：

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

2) “打印然后跨越”是：

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

“在新页面上重新打印标题”选项确定是否在每个新预览页面上打印表格标题。

如果表格单元格中有两个或更多值，则使用“并排单元格”选项。它确定这些单元格值是并排打印还是一个堆叠在另一个之上（默认值）。

如果它们包含相同的值，则“加入相等单元格”选项会水平合并单元格：

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

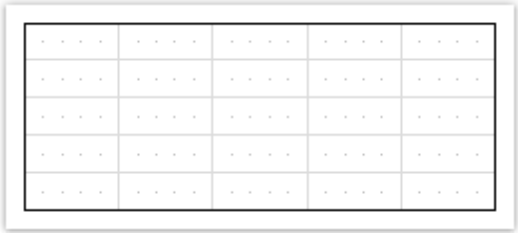
对象检查器中还提供了一些其他属性：

- AddWidth, AddHeight：向单元格宽度或高度添加指定的空间量。 FastReport引擎计算单元格大小时会考虑它（必须启用“自动大小”选项）
- NextCross：指向另一个交叉表对象的指针，该对象将打印到此一侧
- NextCrossGap：两个相邻交叉表对象之间的间隙

章节 8

表类型报表

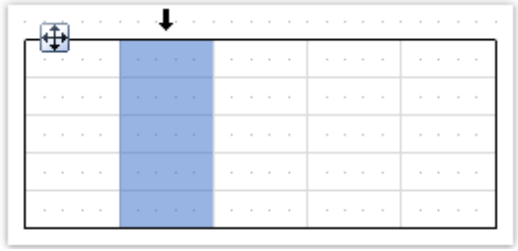
“表”对象由行，列和单元格组成。它是Microsoft Excel表的简化模拟。它看起来如下：



8.1 配置列

您可以在上下文菜单的帮助下添加或删除列。为了那个原因：

- 选择表或其任何元素，并将光标放在所需的列上。光标的形状变为小黑箭头。



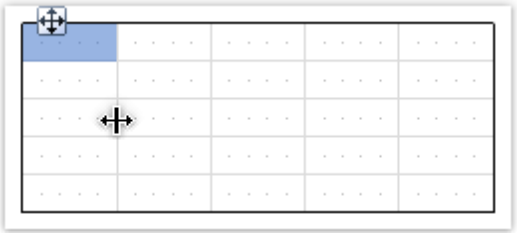
- 左键单击，以选择列；
- 右键单击以显示列的上下文菜单；
- 如果您需要选择多个相邻列，请单击鼠标左键，然后在不离开的情况下将鼠标向右或向左移动，以便选择相邻的列。

也可以在“报表树”窗口中调用列的上下文菜单。打开窗口，选择所需的列并右键单击鼠标。

8.2 管理列的大小

您可以使用以下方法之一设置列的宽度：

- 选择表或其任何元素，并将光标放在两列之间的边框上。光标的形式变为水平分割器：



- 选择一列并在“宽度”属性中指明所需的宽度。可以在“属性”窗口中访问此属性。

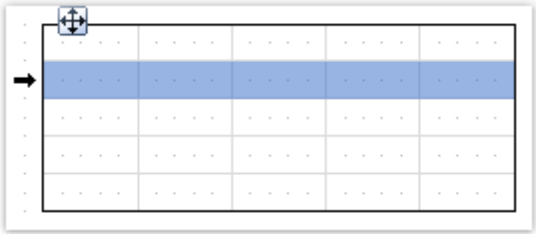
您还可以启用“AutoSize”列属性。运行报表时，将自动计算列的宽度。为了限制列的宽度，您可以指定“MinWidth”和“MaxWidth”属性。

列的宽度绝不应大于页面的宽度。

8.3 配置行

行以相同的方式配置。要选择行，请执行以下操作：

- 选择表或其任何元素，并将光标放在所需行的左侧。光标的形式变为一个小的黑色箭头：



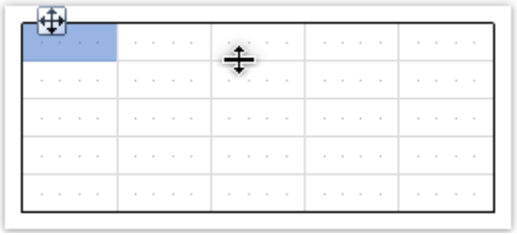
- 左键单击鼠标，以选择行；
- 右键单击鼠标以显示行的上下文菜单。

如果需要选择多个相邻的行，则左键单击鼠标，不离开它，将鼠标向右或向左移动，以便选择相邻的行。

8.4 管理行的大小

您可以使用以下方法之一设置行的高度：

- 选择表或其任何元素，并将光标放在两行之间的边框上。光标的形式变为垂直分割器：



左键单击并移动鼠标，以更改行的大小。

- 选择行并在“高度”属性中指明所需的高度。可以在“属性”窗口中访问此属性。

您还可以启用行的“AutoSize”属性。运行报表时，将自动计算行的高度。为了限制行的高度，您可以使用最小高度（“MinHeight”）和最大高度（“MaxHeight”）属性：

行的高度绝不应大于页面的高度。

8.5 配置单元格

单元格是文本对象。实质上，cells类是从“Text”对象继承的。以上关于“文本”对象的所有内容也适用于表格的单元格。

编辑单元格的文本可以像“文本”对象一样完成。此外，您可以将元素从“数据”窗口拖放到单元格中。

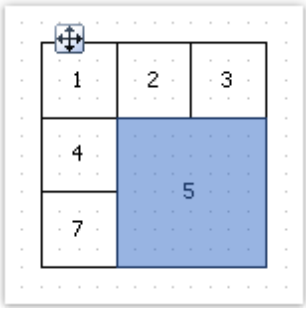
可以使用“边框和填充”工具栏配置单元格的边框和填充。要调用单元格的上下文菜单，

只需右键单击单元格即可。

8.6 加入和分裂细胞

您可以加入表格的相邻单元格。结果，将有一个大单元格。为此：

- 借助鼠标选择第一个细胞；
- 左键单击，不离开它，移动鼠标以选择单元格组；
- 在所选区域上，右键单击鼠标以显示单元格的上下文菜单；在上下文菜单中，选择“加入单
- 元格”项。



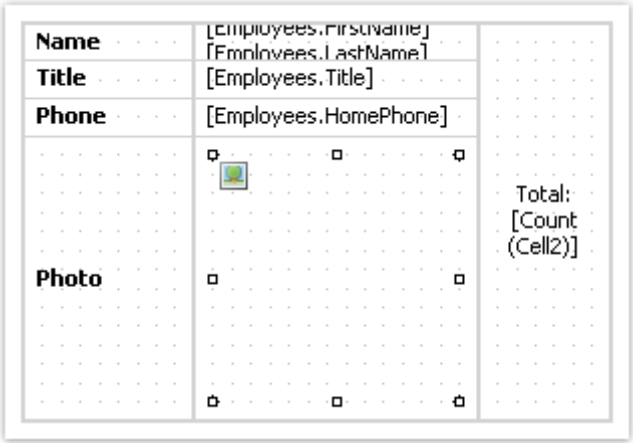
要拆分单元格，请调用其上下文菜单并选择“拆分单元格”项。

8.7 在单元格中插入对象

在单元格中，您可以插入其他对象，例如图片。永远不能将以下对象添加到单元格中：

- “表”；
- “跨标签”；
- “子报表”。

要将对象添加到单元格中，只需将其拖动到单元格内即可。您可以在单元格之间自由移动对象，并将其带回到表格的边界之外。



单元格用作放置在其中的对象的容器。这意味着，您可以使用单元格内对象的“对齐”属性。这允许在单元格的大小改变时改变对象的大小。

章节 9

图表

FastReport可以将图表插入报表中。“TfrxChartObject”  来自Delphi中的FastReport组件面板的对象用于此purpoae。该组件基于“TeeChart”库，该库包含在Delphi分发工具包中。或者，可以单独购买“TeeChartPro”库。

让我们用简单的图表制作一个示例报表。该图表将使用Delphi附带的“DBDEMOS”数据库中的“Country”表。该表包含有关国家，地区和人口的数据：

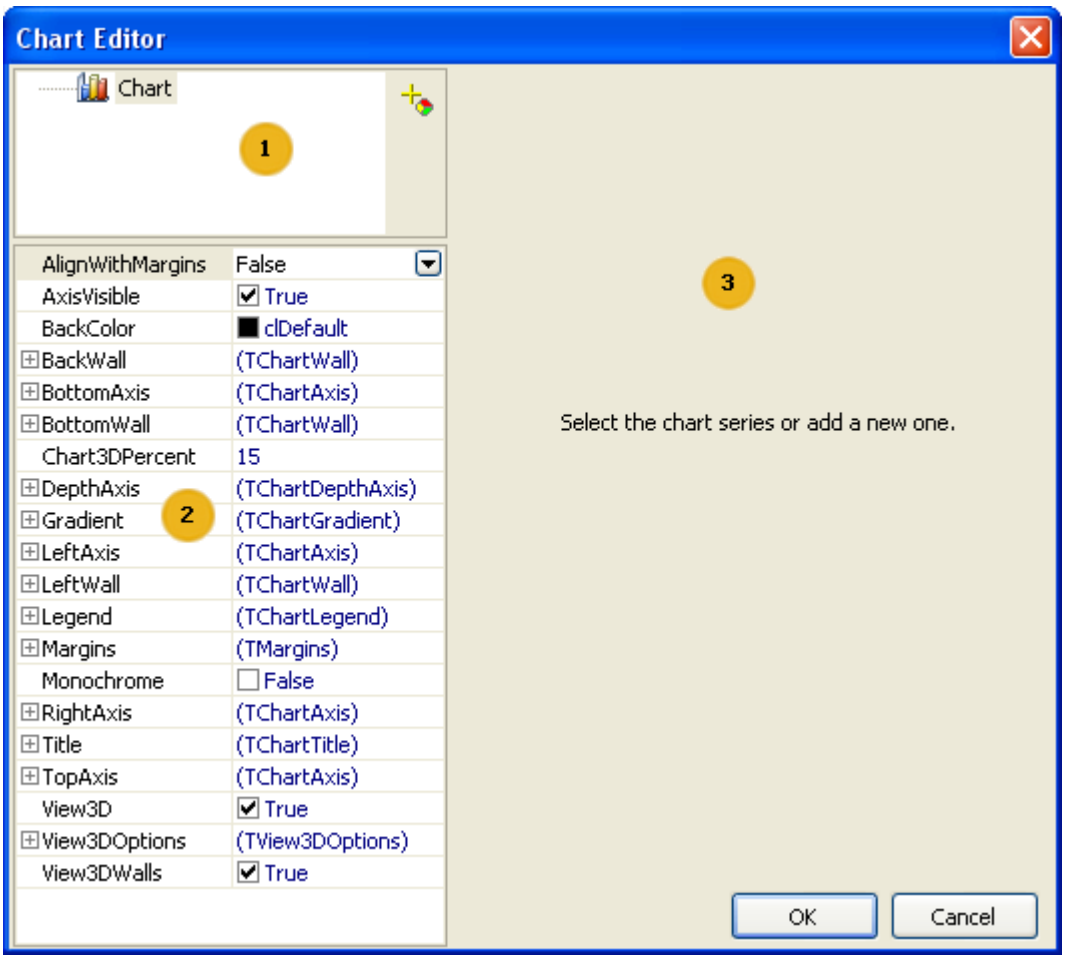
名称	区域	人口
阿根廷	2 777 815	32 300 003
玻利维亚	1 098 575	7 300 000
….		

在Delphi中创建一个新项目。在表单上放置“TTable”，“TfrxDBDataSet”，“TfrxChart”和“TfrxReport”组件并设置这些属性：

表格1：
DatabaseName = 'DBDEMOS'
TableName = 'country.db'

frxDBDataSet1:
DataSet = Table1
UserName ='Country'

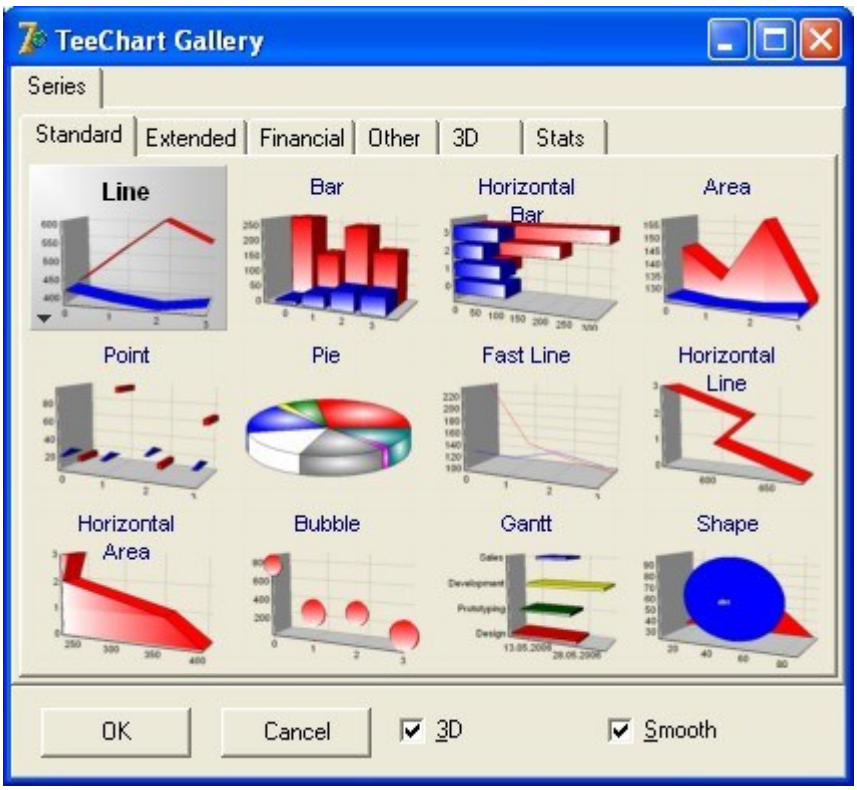
打开报表设计器，创建新报表并在“报表>数据...”对话框中连接数据源。将“图表”对象添加到报表设计页面，并将其大小设置为18厘米宽x 8厘米高。双击打开其对象编辑器。



以上图表编辑器的关键：

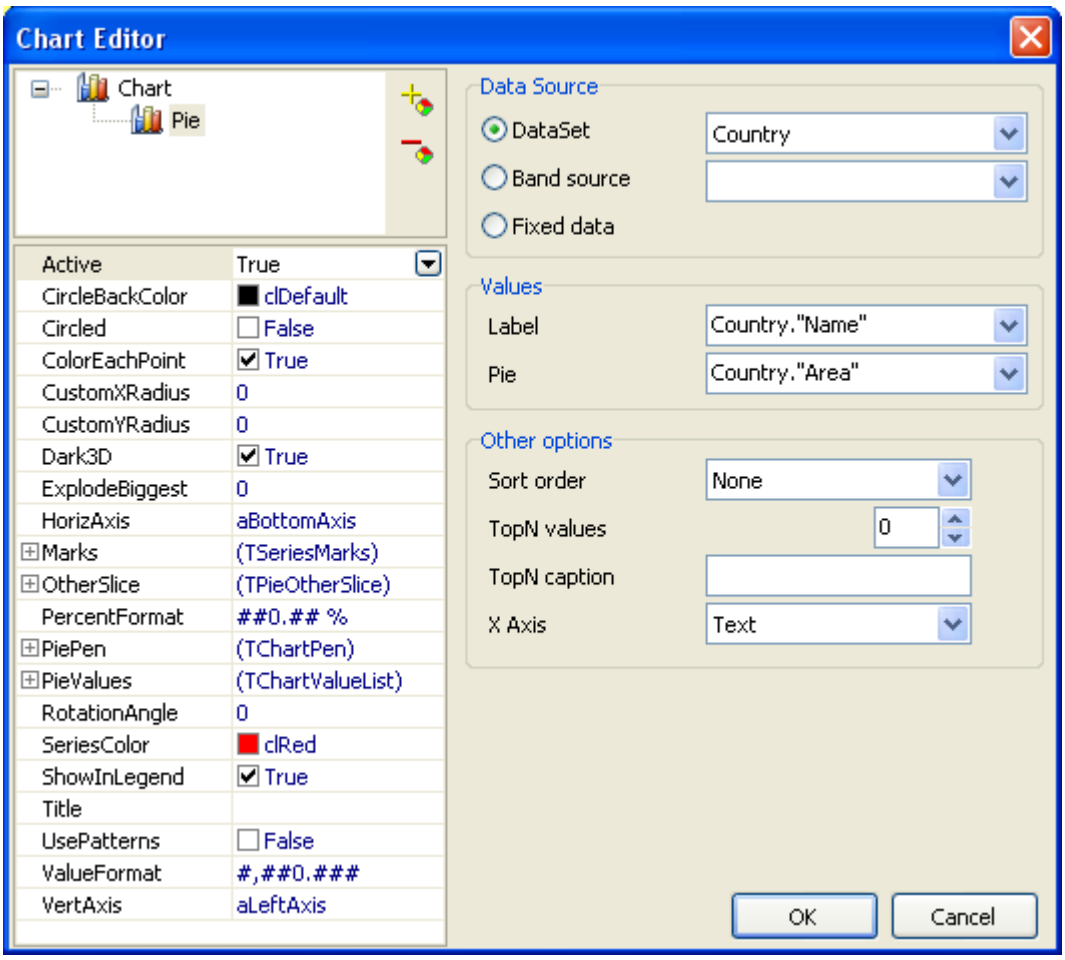
1 –图表结构；图表可以包含一个或多个系列
2 –对象检查器，显示在窗口中选择的元素的属性；在此处设置图表属性
3 –用于连接系列和数据的选项区域；一旦选择了区域1中的一系列，它就会被激活

首次打开时，图表编辑器将显示在上图所示的图像中。第一个任务是向图表添加一个或多个系列（在我们的示例中只是一个系列）。通过单击添加按钮/并在库中选择饼图来执行此操作：

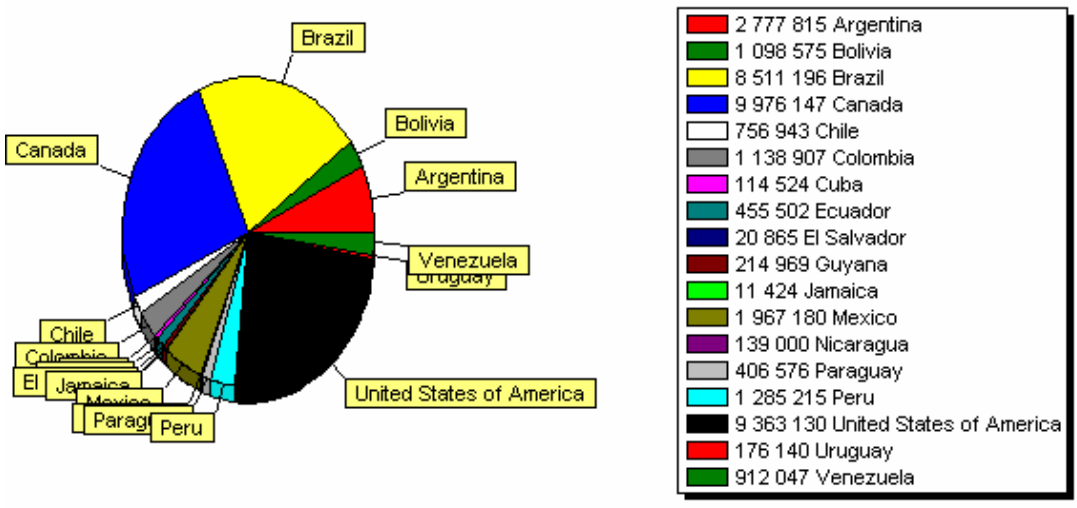


垂直箭头更改列表中序列的顺序。要更改系列的名称，请选择系列，然后再过一秒钟再次单击它（请注意，这不是双击）。

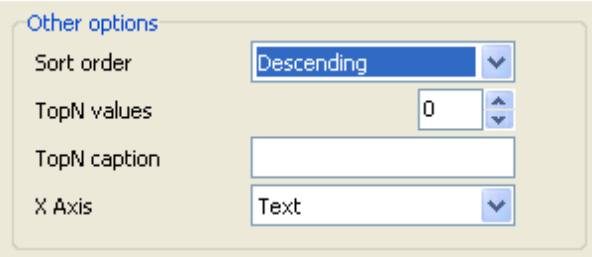
有许多不同类型的系列可供选择。添加一系列后，选项区域3变为活动状态。在此处指定应使用哪些数据绘制图表。首先，让我们在“DataSet”下拉列表中选择数据集。然后使用各自的下拉列表选择“标签”和“饼图”字段，如下所示：



单击“确定”关闭编辑器并预览报表：



本报表中可以改进哪些内容？按降序对群体进行排序会很不错。再次打开图表编辑器，在图表结构中选择系列，并将排序顺序从None更改为Descending：

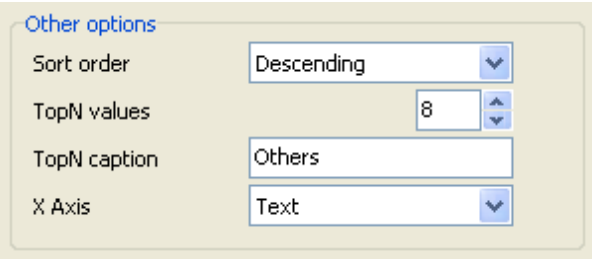


如果我们现在预览报表，我们会看到图例表中的数据已按降序排序。

9.1 限制图表值的数量

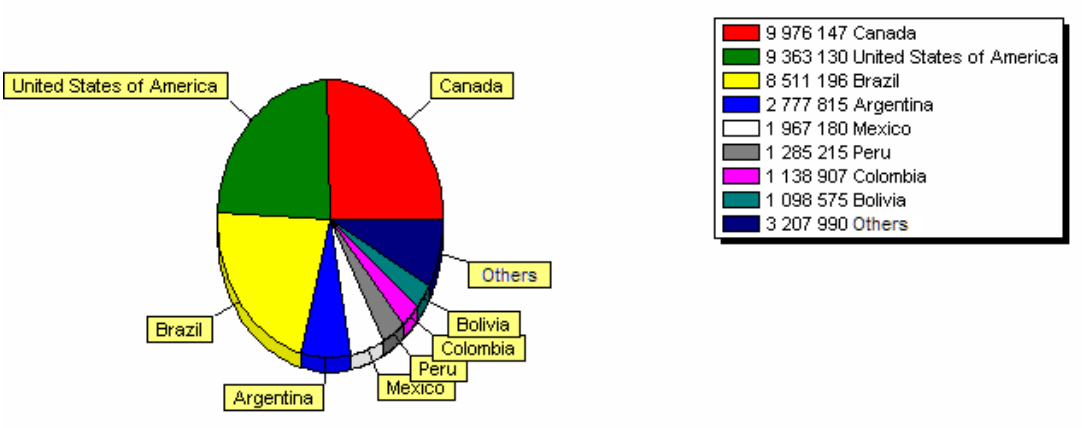
我们的图表过于拥挤，有许多小的价值观，导致许多人变得无形。 FastReport允许您限制图表中显示的值的数量。超过设定数量的所有值都显示为单个复合值，由所有这些值的总和组成。

在我们的例子中，图表有18个值，但实际上只能看到其中的8个。打开图表编辑器并将限制设置为8：



如果“TopN”设置为零，则没有限制。应在“TopN标题”中输入名称，以便在图例列表中正确标识聚合值。排序模式无关紧要，因为默认情况下值始终按降序排序。

该报表将如下所示：



9.2 一些有用的设置

让我们看一下控制图表外观的一些有用设置。只能通过图表编辑器的对象检查器访问这些设置。

在图表结构中选择“图表”时，这些基本属性可用：

- 渐变：渐变背景填充的设置。启用 “Gradient.Visible”属性以显示渐变
- 图例：图例列表的设置。可以通过 “Legend.Visible”属性隐藏List，并通过 “Legend.Alignment” 属性设置其位置。

选择系列后，可以使用以下属性：

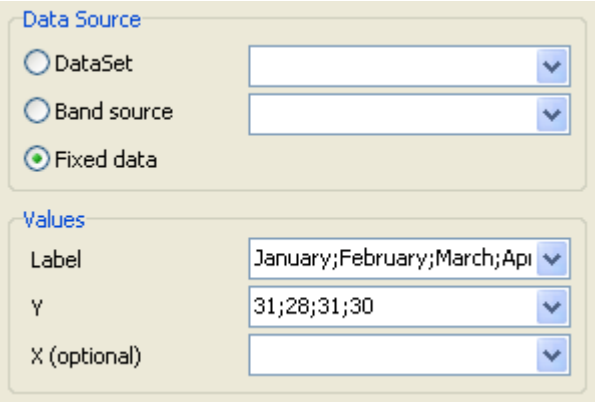
- ColorEachPoint：使用不同的颜色为每个值着色
- ExplodeBiggest：开始最大值（仅适用于饼图）
- 标记：图表标签外观的设置
- ValueFormat：格式化值的行

请注意，所有图表功能都可以在TeeChart Pro库中找到（可以从www.teechart.com单独购买）。该库包含许多类型的图表，包括方便的图表和系列编辑器。

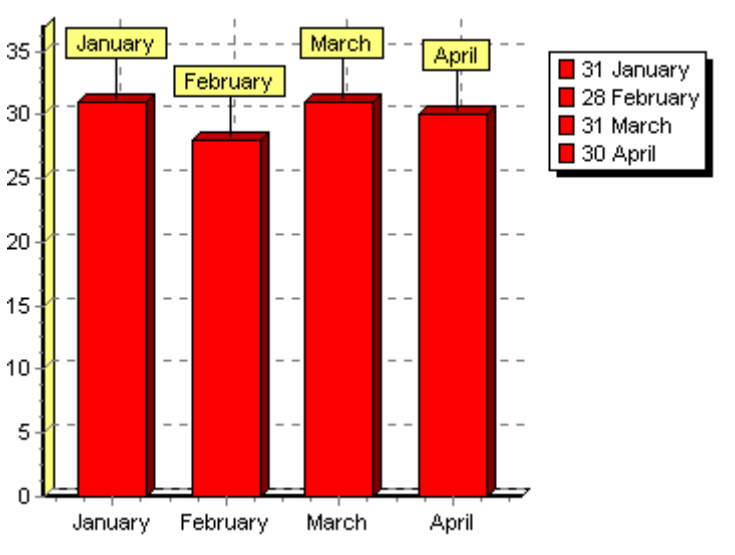
9.3 手动输入值的图表

在前面的示例中，我们使用DB表中的数据创建了一个图表。也可以从手动输入的数据创建图表。如果图表很小，这种方法很方便。

让我们用一个简单的例子演示它是如何工作的。在报表设计页面上放置图表并打开图表编辑器。添加一系列 “条形图” 类型并设置这些属性，使用分号分隔各个值：



结果图表是：



9.4 从脚本完成图表

让我们看看如何使用脚本构建上一个图表。在图表编辑器中将Label和Y字段留空。在报表脚本中输入以下代码：

PascalScript：

```
开始
  Chart1.SeriesData [0] .XSource: =’Jan; Feb; Mar; Apr’;
  Chart1.SeriesData [0] .YSource: = ’31; 28; 31; 30’;
结束。
```

C ++脚本：

```
{
  Chart1.SeriesData [0] .XSource =“Jan; Feb; Mar; Apr”;
  Chart1.SeriesData [0] .YSource =“31; 28; 31; 30”;
```

```
}
```

在这种情况下，SeriesData [0]允许我们为图表中的第一个系列设置参数。如果图表有多个系列，则用SeriesData [1]等引用它们。

9.5 打印使用Delphi构建的图表

如果您已经在Delphi代码中构建了一个图表并想要在报表中打印它，则需要使用FastReport对象工具栏中的“图片”对象。将对象放在报表设计页面上的所需位置，并在Delphi应用程序中创建以下“TfrxReport.OnBeforePrint”事件处理程序：

```
procedure TForm1.frxReport1BeforePrint (Sender: TfrxReportComponent) ;
开始
  如果Sender.Name ='Picture1'那么
    TfrxPictureView (发件人) .Picture.Assign
      (Chart1.TeeCreateMetafile (False,
        Rect (0,0, Round (Sender.Width) , Round (Sender.Height) ) ) ) ;
结束;
```

其中Picture1是“Picture”对象的名称，Chart1是Delphi图表。

注意：如果在Delphi应用程序中为TfrxReport组件的事件处理程序分配了代码，则必须通过运行已编译的Delphi应用程序来预览报表。您无法在FastReport报表设计器中预览报表。

章节 10

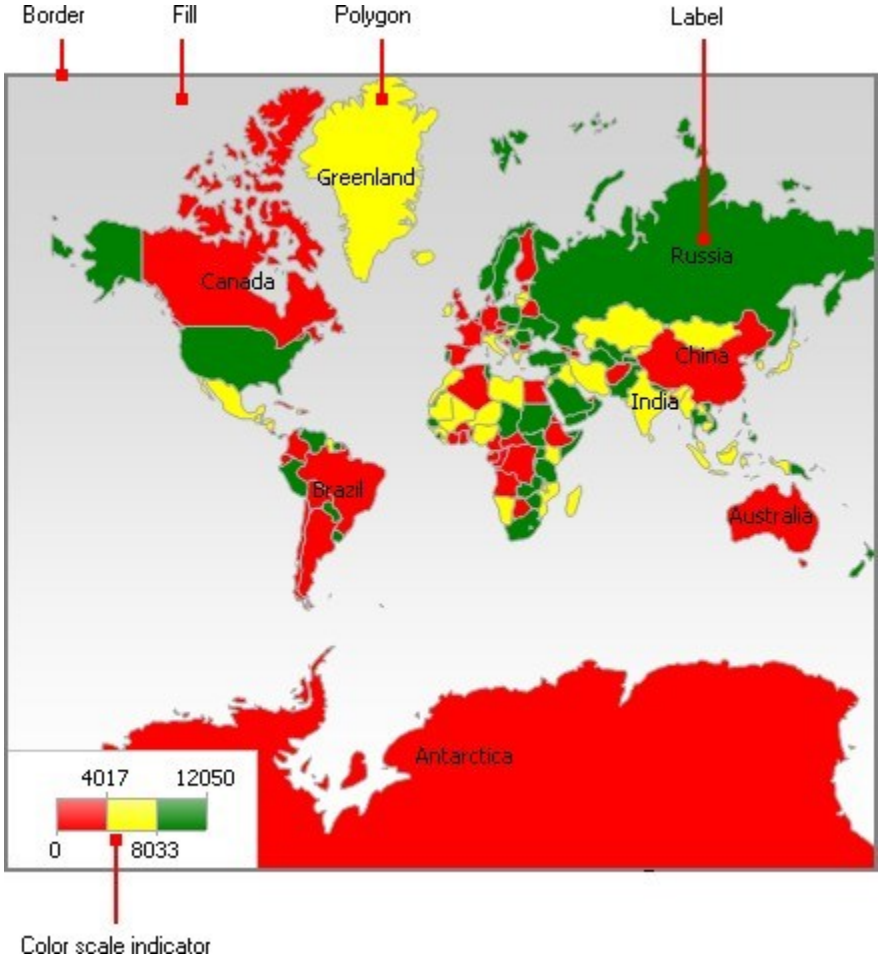
地图

“Map”对象可以显示ESRI shapefile格式（.shp / .dbf），Open Street Map（.osm），GPS-track（GPS Exchange File，.gpx）的二维地图。在此处查看有关这些格式的更多详情

http: / / /wiki / shapefile的维基百科
http: / / / /维基百科维基型百科
OpenStreetMap http: / / /wiki /
GPx

10.1 地图元素

“Map”对象由以下元素组成：



一个“地图”对象可以显示一个或多个图层。每个图层都包含自己的地图。

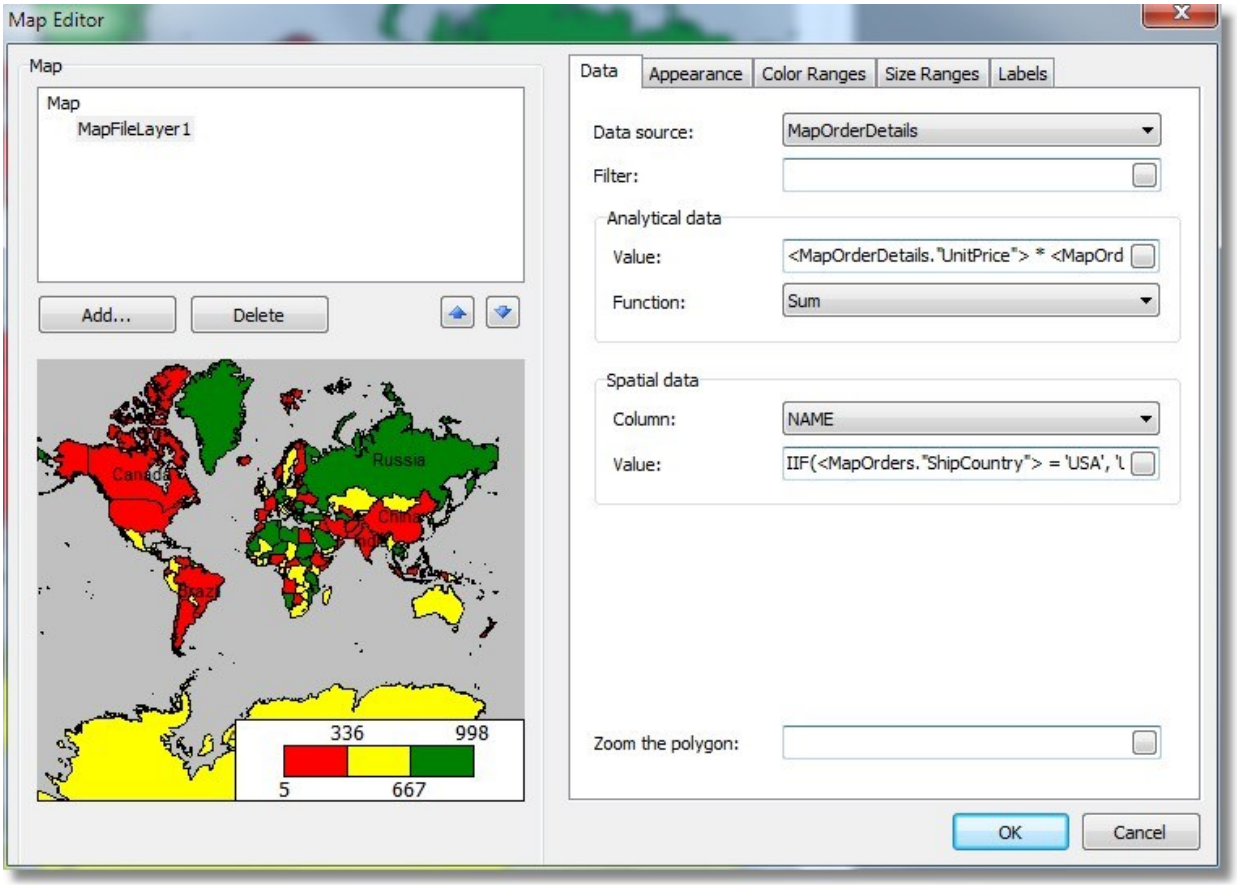
10.2 用鼠标控制地图

在报表设计器和预览窗口中，可以通过鼠标控制地图：
- 使用鼠标滚轮放大和缩小
- 通过拖动鼠标左键来平移地图
- 通过在地图多边形内单击鼠标左键，在“对象检查器”窗口中更改多边形的属性

可以在MinZoom和MaxZoom属性中设置最小和最大缩放值。这些属性在“对象检查器”窗口中更改。

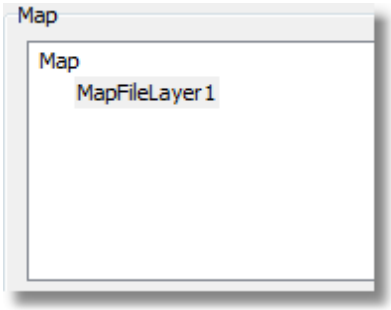
10.3 “地图”对象编辑器

“Map”对象具有许多属性，可以在对象编辑器中设置。通过双击对象或从其上下文菜单中打开编辑器：

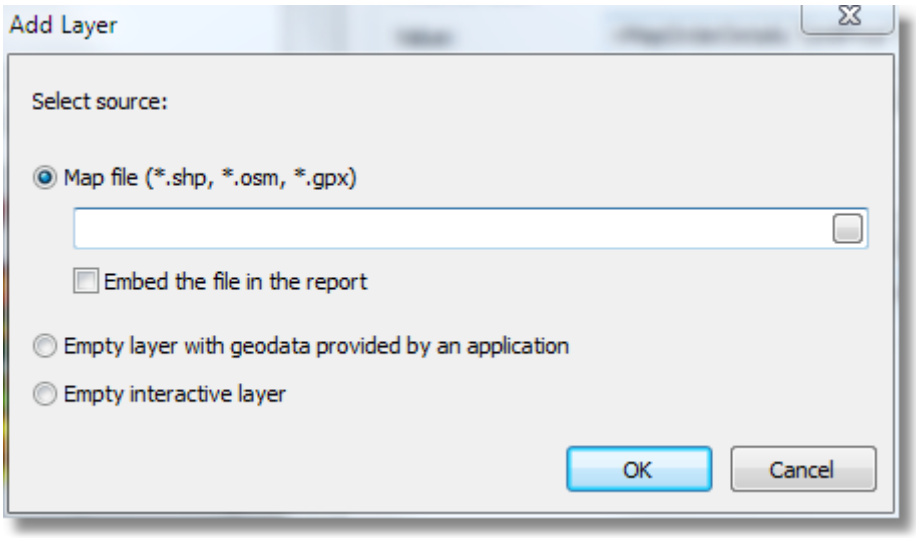


10.3.1 添加地图图层

“Map”对象可以包含一个或多个图层。图层结构显示在编辑器窗口的左上角：



要添加新图层，请单击“添加...”按钮，这将打开以下对话框：



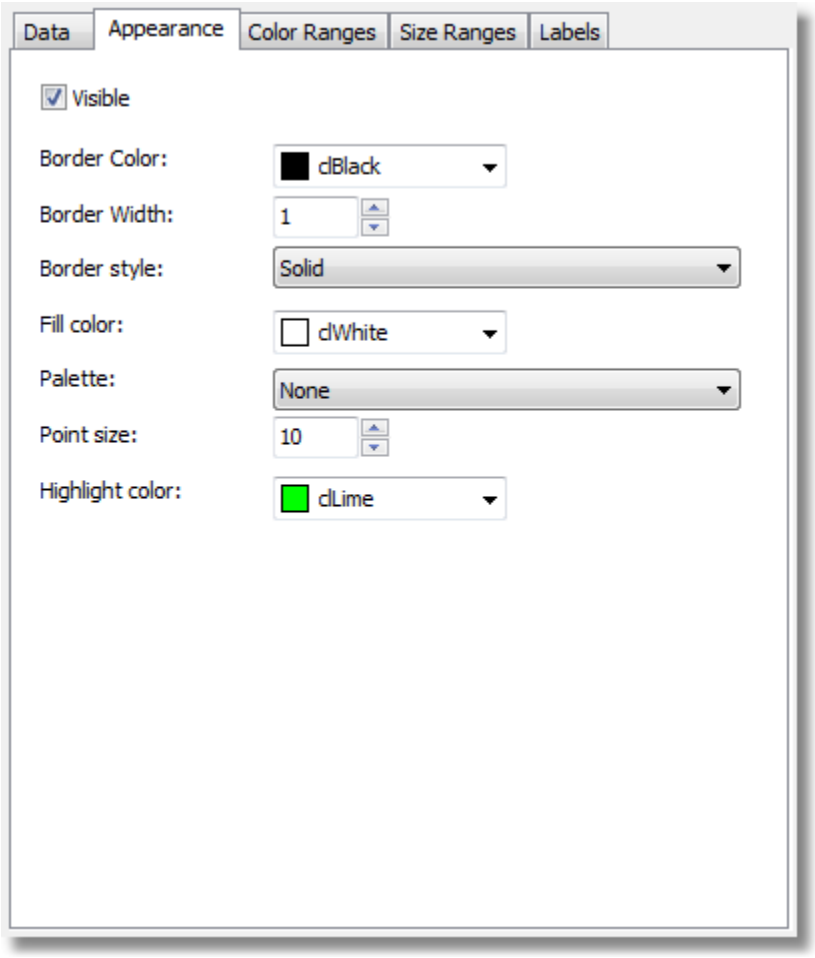
此时，从以下位置选择图层的类型：
- 来自ESRI shapefile的地图。这是地图最常用的类型。例如，您可以显示世界地图并突出显示某些国家/地区的颜色，具体取决于这些国家/地区的销售水平；
- 来自您的应用程序的地理数据。您的应用程序必须提供以下数据：纬度，经度，名称和值。此数据显示为图层上的一个小点。该点可以有标题，也可以具有可变的大小/颜色，具体取决于提供的值。实际上，这种类型的层与“ESRI shapefile”类型的基础层一起使用。基础层（第一个）配置为显示国家/地区地图（例如），第二个层（来自应用程序的地理数据）显示点 - 销售发生的城市名称。可以配置点的颜色和大小以反映销售水平。

如果选择“ESRI shapefile”图层类型，请从以下位置选择地图数据的存储方式：
- 整个地图数据嵌入在报表文件中。在这种情况下，报表文件（.frx）可能很大；
- 报表文件包含对映射文件（.shp / .dbf）的引用。如果多个报表使用相同的映射文件，则此模式很有用。

注意：大型地图文件（超过30Mb）或包含大量多边形（超过20,000）的地图文件会减慢报表生成速度。

10.3.2 设置图层外观

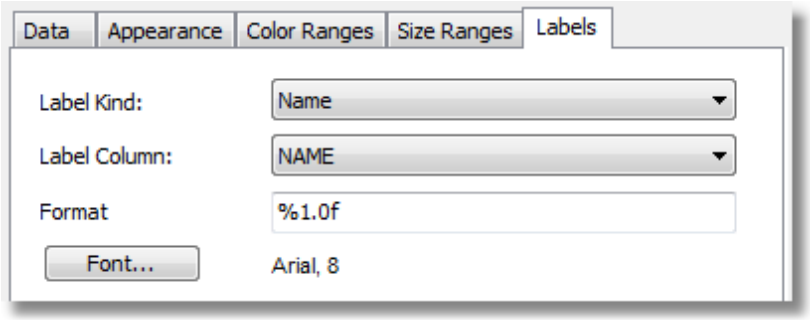
要设置图层外观，请选择图层并切换到“外观”选项卡：



设置地图多边形的边框颜色和样式，然后选择调色板。请注意，如果配置色标，则会忽略调色板（稍后会详细介绍）。

10.3.3 设置标签显示

地图可以显示标签，例如国家/地区名称。在“标签”选项卡上设置标签类型和外观：



如果选择“ESRI shapefile”图层类型，请设置包含显示信息的字段名称。在大多数情况下，它是一个“NAME”字段。 FastReport演示程序中包含的世界地图包含以下字段：
- 姓名（例如：德国）
- ABBREV（例如：Ger。）
- ISO_A2（例如：DE）
- ISO_A3（例如：DEU）

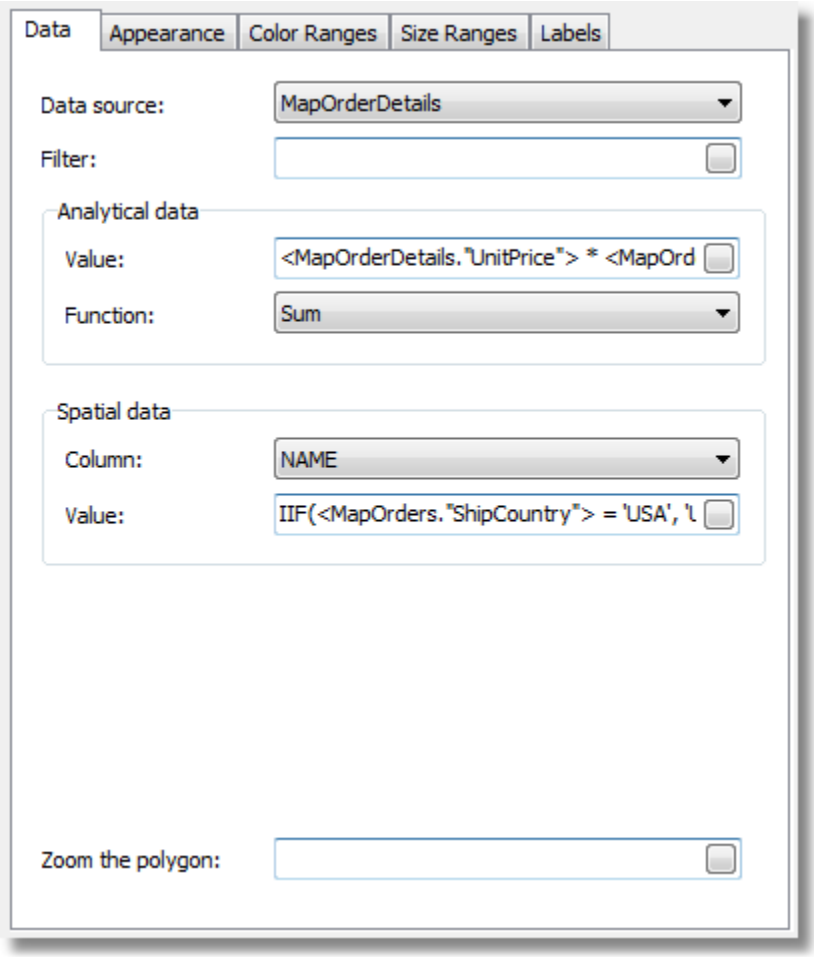
其他地图将具有不同的字段集。

如果选择“应用程序地理数据”图层类型，请设置显示标签的最小缩放值。默认值为1，表示始终显示标签。

10.3.4 将地图图层连接到数据

大多数报表使用“地图”对象来显示分析数据，例如各个国家/地区的销售水平。因此，必须使用地图对象编辑器窗口将地图图层连接到数据源。从图层树中选择图层，然后切换到“数据”选项卡。必须提供适合地图图层类型的数据：

- 对于“ESRI shapefile”类型的地图图层，“数据”选项卡如下所示：



所需数据是：
- 姓名（例如：国名）；
- 数值（例如：此国家/地区的销售量）。

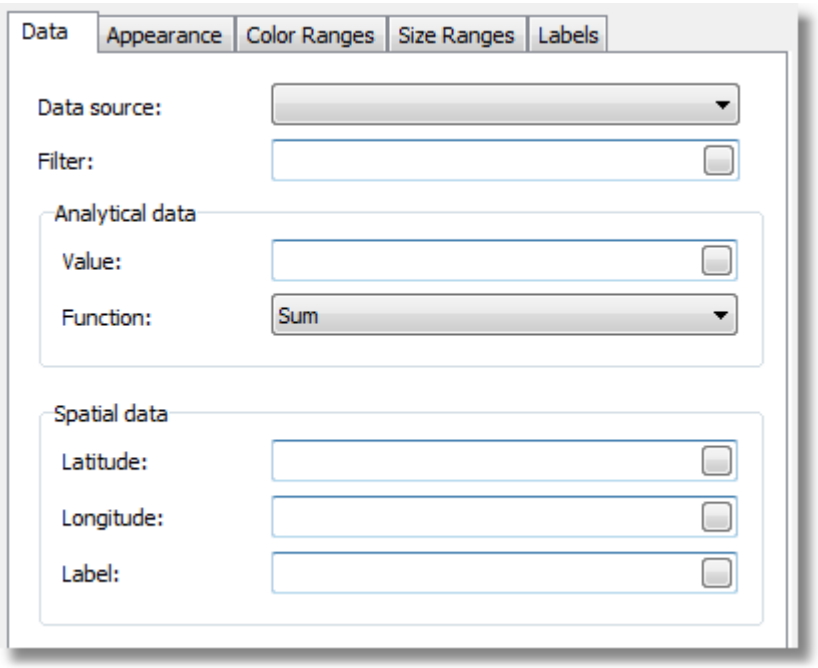
例如，“Sales”数据源可能包含以下数据：

国家	SalesTotal
-------	- - - - - 美
国	500000
德国	1200000
俄国	300000

按以下方式设置“数据”选项卡：
- 数据来源：销售
- 空间数据，列名：选择包含国家名称的列；通常这是“NAME”列
- 空间数据，价值：[Sales.Country]
- 分析数据，值：[Sales.SalesTotal]
- 分析数据，功能：“总和”；如果每个国家/地区有多个值，则使用此功能。

“缩放多边形”编辑框允许使用给定名称缩放多边形，因此它占据整个“地图”对象工作空间。例如，要在世界地图上缩放德国，请在此编辑框中键入“德国”（带引号）。

- 对于“来自应用程序的地理数据”类型的地图图层，“数据”选项卡如下所示：



所需数据是：
- 空间数据：经度和纬度；
- 标签（例如：城市名称）；
- 数值（例如：此城市的销售量）。

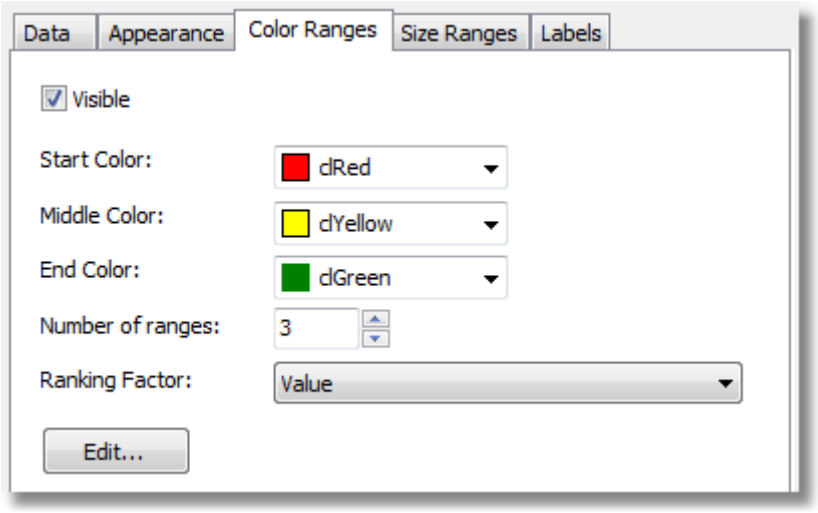
例如，“Sales”数据源可能包含以下数据：

纬度	经度	城市名称	SalesTotal
48.13641	11.57753	慕尼黑	50000
50.94165	6.95505	科隆	36000

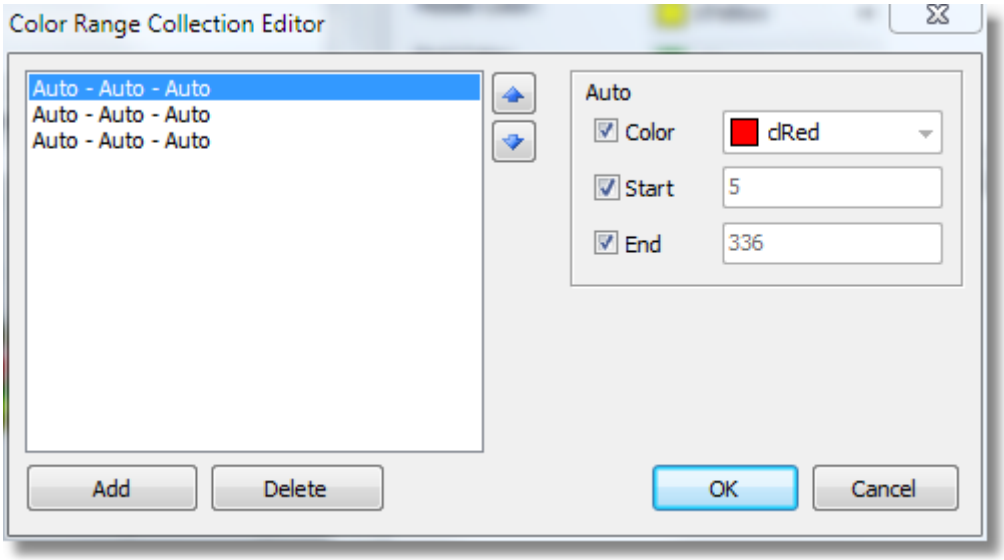
按以下方式设置“数据”选项卡：
- 数据来源：销售
- 空间数据，纬度：[Sales.Latitude]
- 空间数据，经度：[Sales.Longitude]
- 空间数据，标签：[Sales.CityName]
- 分析数据，值：[Sales.SalesTotal]
- 分析数据，功能：“总和”；如果每个城市有多个值，则使用此功能。

10.3.5 使用颜色突出显示数据

将图层连接到其数据后，下一个问题是 - 如何显示分析数据（例如各个国家/地区的销售量）？最简单的方法是设置显示国家/地区名称及其销售额的标签（请参阅“设置标签显示”）。通过使用颜色绘制每个国家，可以实现更多的视觉效果，具体取决于其销售量。例如，低销售额为红色，中间销售为黄色，高销售额为绿色。这可以通过在“色阶”选项卡上设置色标来完成：

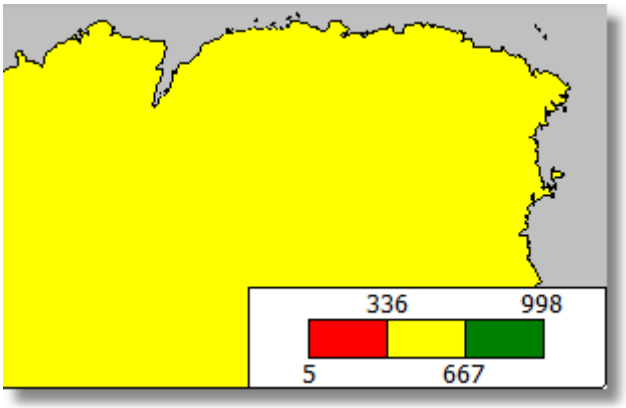


色标由几个范围组成。每个范围都具有以下属性：最小值，最大值和颜色。您可以根据需要使用多个范围。要设置色标，请先设置范围数，然后设置每个范围的属性。

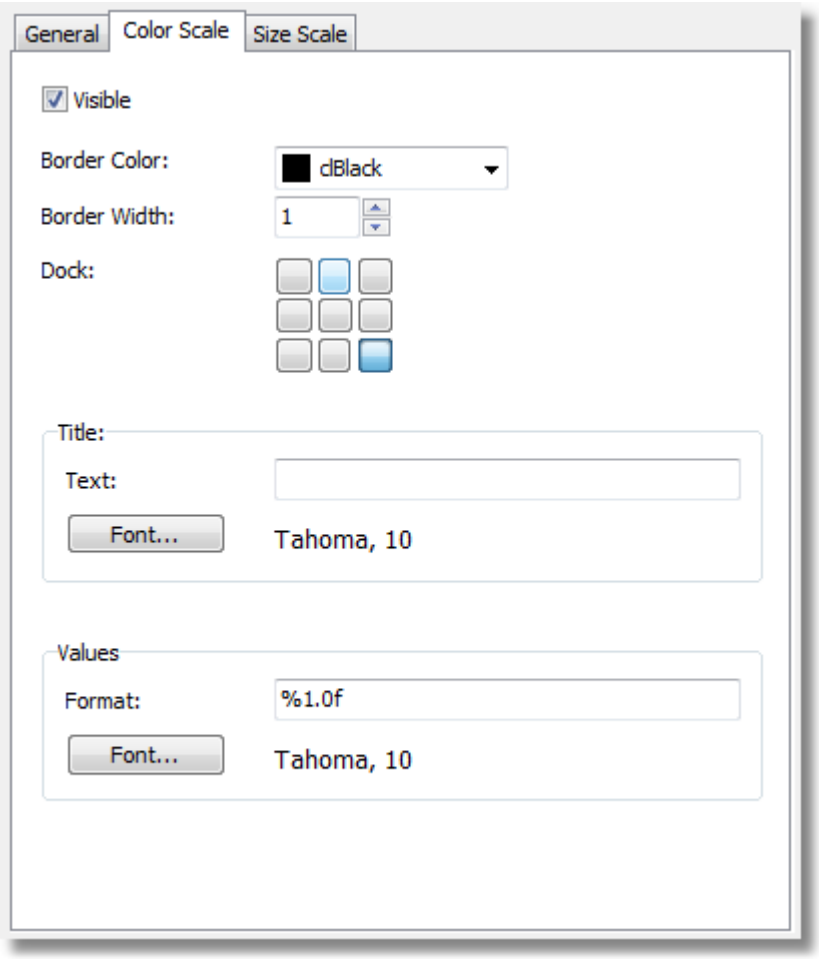


默认情况下，所有范围属性都设置为“自动”，这意味着FastReport会自动计算每个范围的最小值和最大值。自动颜色从三个预设中选择（“开始颜色”，“中间颜色”，“结束颜色”）。在大多数情况下，“自动”模式可能是合适的。

设置色标时，“Map”对象底部会显示一个指示器控件：

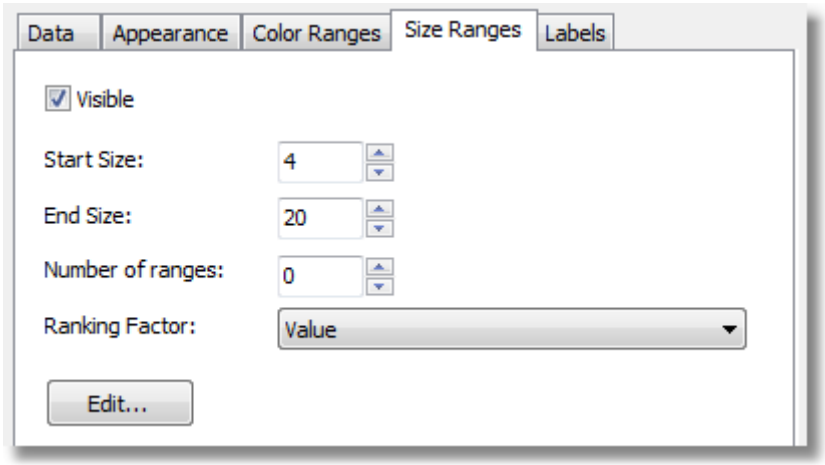


要设置指示器的外观和位置，请在图层树控件中选择“Map”元素，然后切换到“Color scale”选项卡：



10.3.6 使用大小突出显示数据

使用图层类型“来自应用程序的地理数据”时，数据显示为带有标题的小圆圈。圆的大小可以与使用颜色突出显示数据的方式绑定到数据。要执行此操作，请在“大小范围”选项卡上设置大小范围：



大小范围具有以下属性：最小值，最大值和大小（以像素为单位）。您可以根据需要使用多个范围。首先设置范围数，然后设置每个范围的属性。

默认情况下，所有范围属性都设置为“自动”，这意味着FastReport会自动计算每个范围的最小值和最大值。自动尺寸从两个预设中选择（“开始尺寸”，“结束尺寸”）。在大多数情况下，“自动”模式可能是合适的。

章节 11

互动报道

FastReport准备的报表可以互动。这意味着，它将对预览窗口中用户的操作做出反应。您可以使用以下交互：

- 单击报表对象时，将执行某种操作。例如，您可以运行详细报表并在单独的窗口中显示；
- 预览窗口可以显示报表大纲，可用于在报表上导航。

11.1 超链接

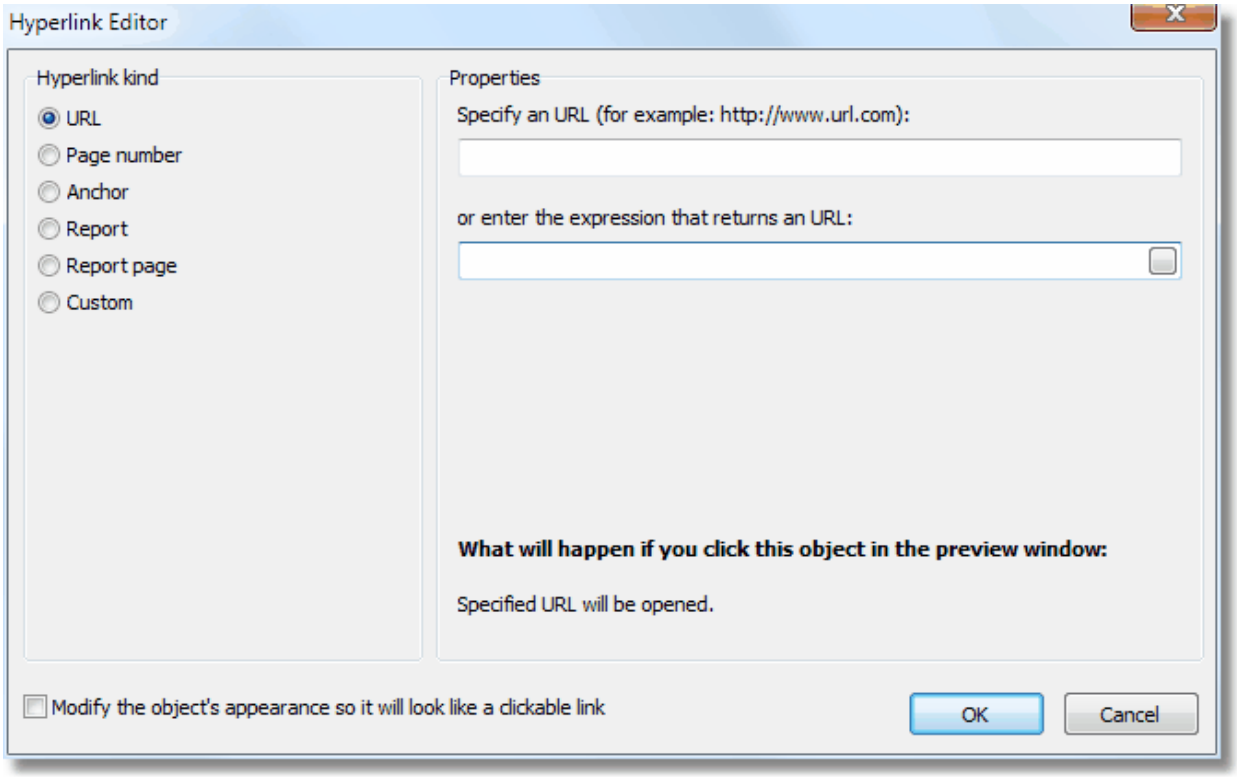
几乎所有报表对象都具有“超链接”属性。使用此属性，您可以在预览窗口中定义对象对鼠标单击的反应。

单击此类对象时，可能会出现以下情况之一：

- 导航到URL地址；发送电子邮件；
- 执行任何类型的系统命令；
- 导航到带有指定编号的报表页面；导航到锚点，添加
- 到脚本中；
- 在单独的预览窗口中运行详细报表；自定义操作，
- 在脚本中定义。

11.2 超链接配置

要配置超链接，请选择要进行交互的对象，然后右键单击它。在上下文菜单中，选择“超链接...”项。超链接编辑器窗口将打开：



通过选择窗口左侧的单选按钮来选择超链接的类型。完成后，您可以单击窗口底部的“修改对象的外观...”复选框。对象的外观将以下列方式更改：

- 将为文本设置蓝色，并将加下划线；将设置手形光标
- 表单。

在某些情况下，需要在预览窗口中显示超链接，但不需要打印它。如果配置“可见性”对象属性，这很容易做到。这可以在“对象检查器”窗口中完成。

11.3 链接到URL

使用此类链接，您可以：

- 导航到给定的互联网地址；
- 执行某种系统命令，例如“mailto：”用于发送电子邮件。您可以使用两种方法指示链

接的值：

- 直接表示该值，例如“http: / / www.fast-report.com";
- 表示一个表达式，它返回链接的值。将计算此表达式

当您运行报表时。

11.4 链接到页码

通过使用此链接类型，您可以在准备好的报表的页面上组织导航。通常，使用导航到第一页。为此，请指示页码（在给定的情况下为1）作为链接值。

您可以使用两种方法指示页码：

- 直接表示数字，例如1；
- 表示一个表达式，它返回页码。运行报表时将计算此表达式。

11.5 链接到锚点

通过使用此类链接，您可以导航到在脚本中添加的锚点。锚在准备好的报表中有一个名称和一个确定的位置（页面上的页码和位置）。当按名称移动到锚点时，您将导航到指示的位置。

要使用此链接类型，首先需要定义锚点。为此，请使用脚本和Engine.AddAnchor方法（请参阅“脚本/锚点”主题中的更多内容）。现在，您可以在超链接配置窗口中指明其名称。这可以通过使用两种方法来完成：

- 直接指出锚的名称；
- 表示一个返回锚名称的表达式。例如，这可以是数据列。运行报表时将计算此表达式。

11.6 链接到详细报表

使用此链接类型，您可以执行另一个报表并在单独的预览窗口中显示它。您必须为此类型的超链

接设置以下参数：

- 详细报表的名称；
- 报表变量的名称，它将取超链接的值；超链接值。
-

单击链接后，将进行以下操作：

- 指定的报表将被加载；
- 报表的变量将设置为超链接的值；
- 报表将在单独的预览窗口中构建和运行。报表变量的值可以使用

两种方法指示：

- 直接表示价值；
- 表示一个返回值的表达式。运行报表时将计算此表达式。

11.7 链接到详细页面

此链接类型的工作方式相同，只是当前报表中的另一个页面用作详细报表。为此，您的报表必须至少包含两个页面：一个包含主报表，另一个包含详细报表。

您必须为此类型的超链接设置以下参数：

- 该报表中的页面名称；
- 报表变量的名称，它将取超链接的值；超链接值。
-

单击链接后，将进行以下操作：

- 报表的变量将设置为超链接的值；
- 指定的报表页面将被构建并显示在单独的预览窗口中。报表变量的值可以使用两种方

法指示：

- 直接表示价值；
- 表示一个返回值的表达式。运行报表时将计算此表达式。

选择报表页面时，其“可见”属性将重置为false。这意味着，当构建主报表时，将跳过此页面。

11.8 自定义链接

使用这种类型的链接，您可以定义自己对单击鼠标的反应。为此，请使用对象的 “OnClick”事件处理程序。

章节 12

点阵报表

早些时候，我们查看了用于在标准现代打印机（手写笔，激光等）上打印的报表。发送到点阵式打印机的报表打印速度非常慢。 FastReport允许创建用于点阵打印机的特殊报表，其中只输出标准字体符号和图形元素；这导致更快的打印速度。

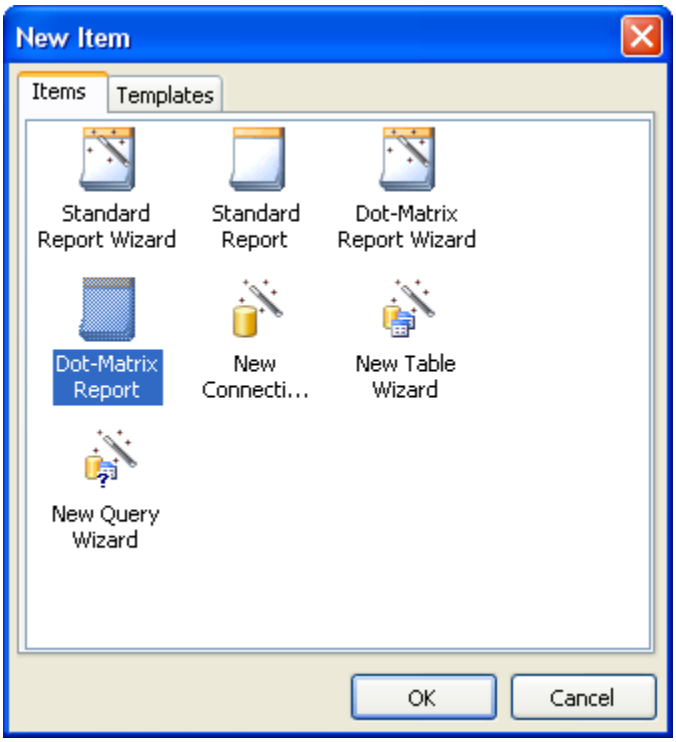
让我们看看如何构建用于点阵打印的“列表”类型的报表。之前我们创建了这种报表，请参阅“客户列表”报表。我们将使用相同的数据进行报表。

因此，在Delphi中启动一个新项目，在表单上放置“TTable”，“TfrxDBDataSet”，“TfrxReport”和“TfrxDotMatrixExport”组件并设置它们的属性：

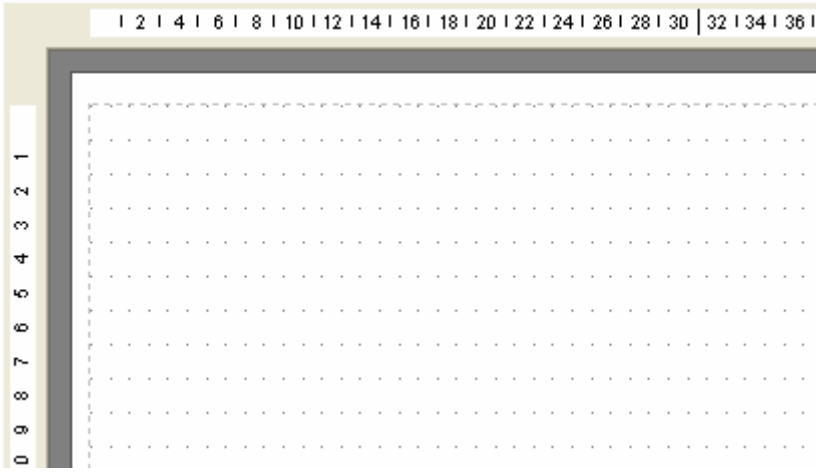
TTable的：
DatabaseName = 'DBDEMOS'
TableName = 'Customer.db'

TfrxDBDataSet：
DataSet = Table1 UserName ='Customers'

打开报表设计器。选择“文件>新建...”以打开报表向导对话框并选择“点阵报表”项：



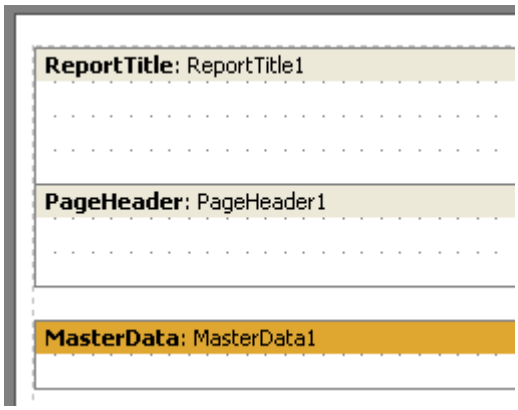
单击OK时，将显示一个空的设计页面，标记为点阵字体：



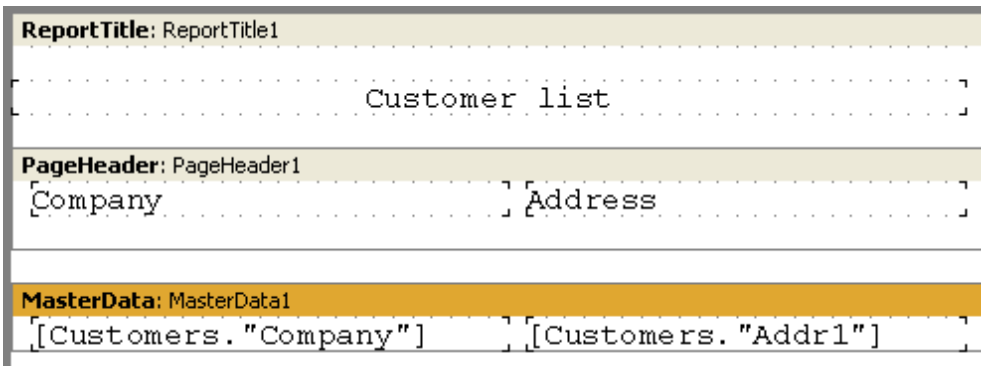
对象工具栏也会更改以显示可用于点阵打印的对象，这些对象是“Band”，“Text”，“Line”，“ESC-Command”，“Subreport”，“Cross-tab”和“DBCross-tab”“对象。其他对象不能用于点阵打印机。



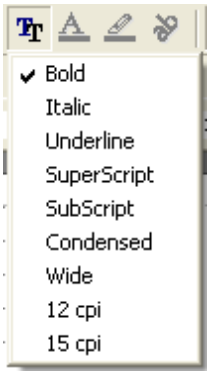
在报表页面上放置“报表标题”，“页眉”和“主数据”段：



将“Text”对象放在段上，如下所示：



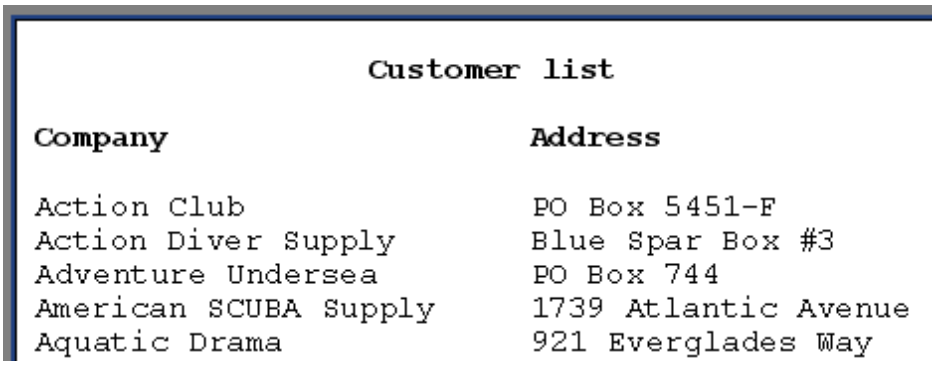
点阵对象的放置类似于普通报表中的放置。但是，这些物体的位置和外观受到严格限制。对象捕捉到网格，字体大小（高度）无法更改，并且无法着色。但是，可以通过选择“文本”对象并单击工具栏上的“Tt”按钮来修改某些字体属性：



这些字体属性特定于点阵打印。除了段之外，报表页面和所有点阵对象都具有这些属性。

注意：在设计器和预览中，只有“Bold”，“Italic”和“Underline”属性在屏幕上实现。整个属性集仅在打印时实现。

让我们使用标题的“粗体”样式修改我们的报表。报表已完成并准备好进行预览：



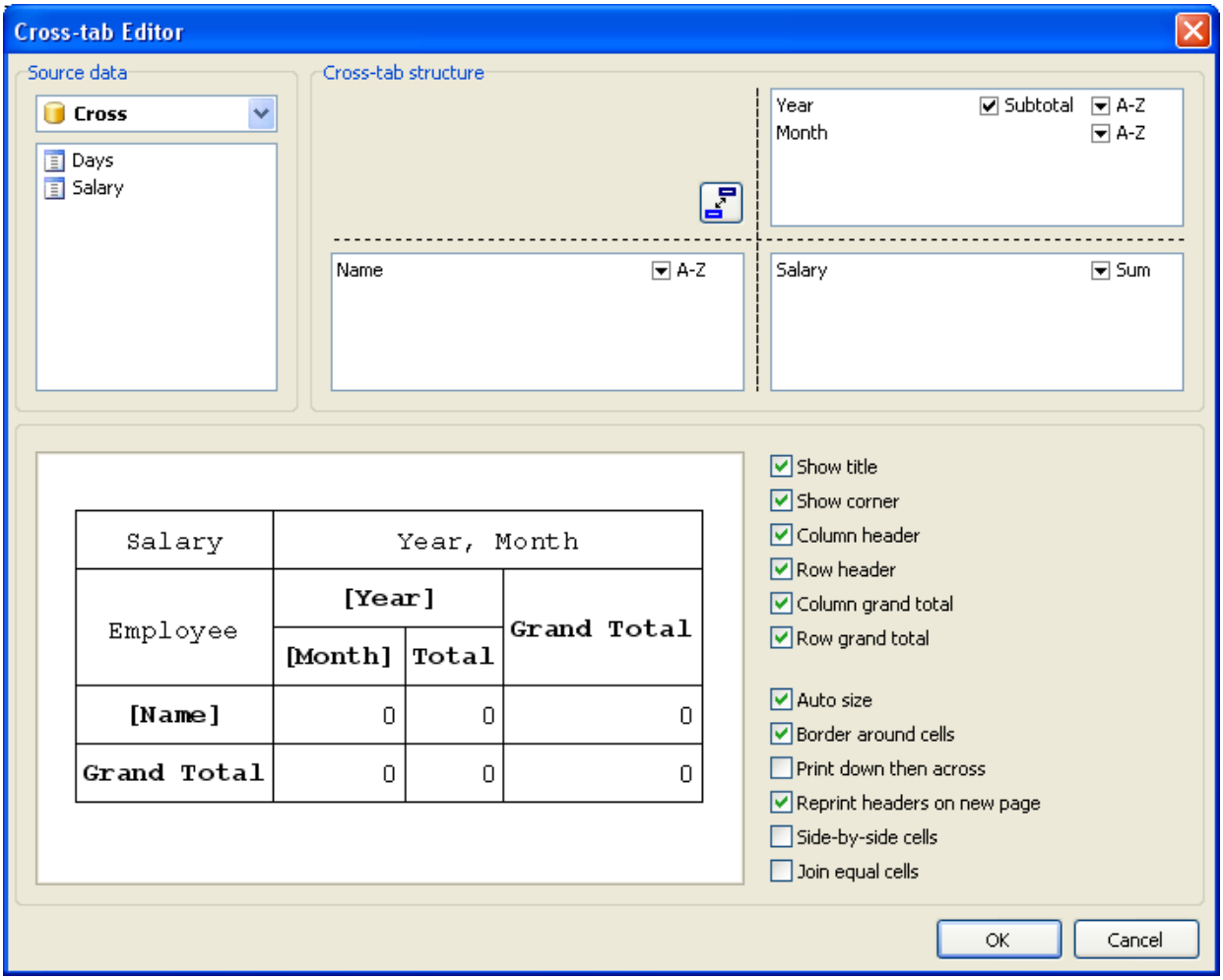
Customer list

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

12.1 点阵中的交叉表

点阵报表中可用的对象范围仅限于可以文本形式显示的对象。其中包括两个“交叉表”对象。让我们看一个简单的交叉表报表，类似于之前在“带有复合头的表”部分中构建的报表。

创建一个点阵报表，如上一节所述，选择“点阵报表”。在报表页面上放置一个“DB cross-tab”对象并打开其编辑器：

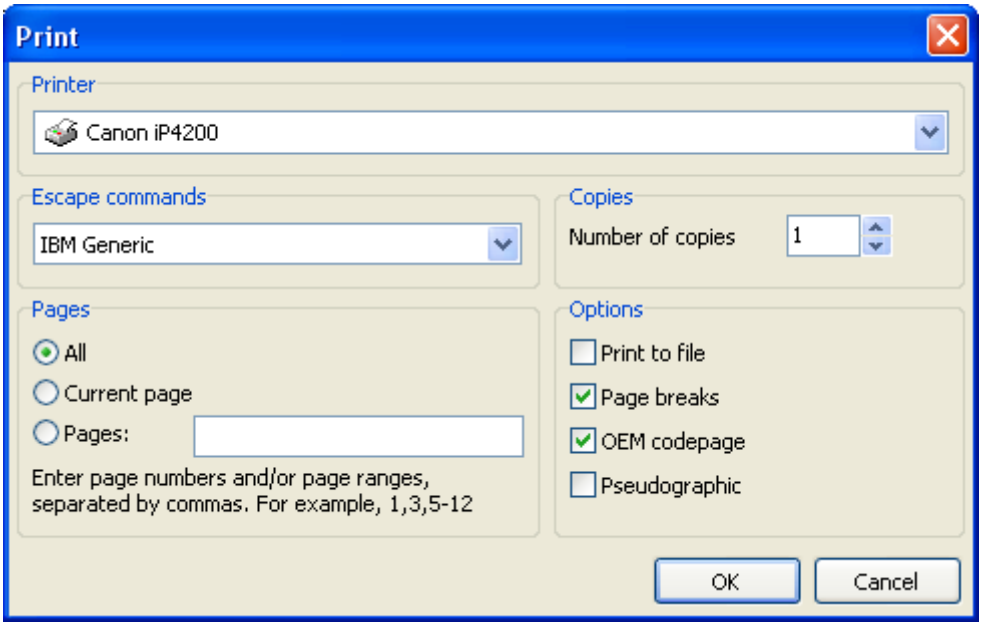


交叉表编辑器以点阵模式显示输出表的结构。可以使用工具栏中的“Tt”按钮设置单元格字体属性。在所有其他方面，交叉表对象的行为如前所述。预览的报表如下所示：

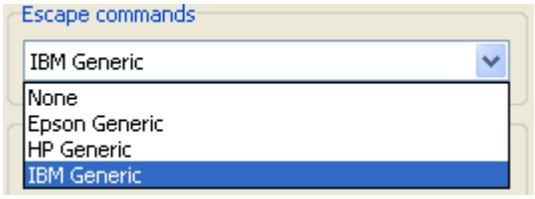
	1999					2000				2001		
	2	10	11	12	Total	1	2	3	Total	1	2	3
Ann	1000		1100	1200	3300	1300	1400		2700		1500	1600
Ben		2100	2200		4300		2400		2400			
Catherine		3000	3100		6100			3200	3200			
Den					0	3999			3999	4000	4100	
Grand Total	1000	5100	6400	1200	13700	5299	3800	3200	12299	4000	5600	1600

12.2 点阵报表打印

要启用预览点阵报表的实际打印，必须首先从Delphi项目表单上的“FastReport”组件面板中放置TfrxDotMatrixExport /组件。此组件处理将预览报表转换为点阵打印机所需的文本格式。它还用点阵打印机的特殊打印机对话取代了普通的打印机对话：



打印机对话框与普通打印机对话框类似，但具有点阵打印机细节。打印前必须从以下列表中选择打印机的命令系统（ESC命令）：

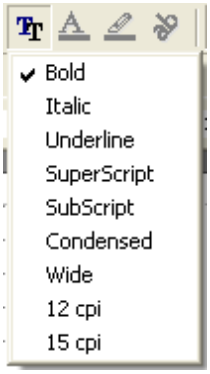


点阵打印还有其他一些选择：

- 打印到文件：是否将打印流发送到硬盘上的文件。如果启用，则会出现正常的“另存为...”对话框
- 分页符：是否在到达页面底部时发送“分页符”控制命令。如果禁用，则在连续信纸上打印不会中断
- OEM-codepage：是否执行符号转换
- Pseudographic：指定如何绘制垂直和水平线。如果禁用，则使用 - ，|绘制线条和+符号

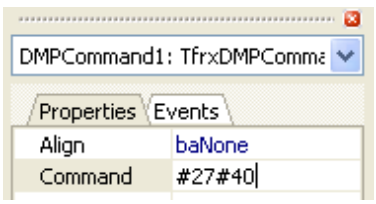
12.3 “命令”对象

如前所述，可以为点阵报表对象设置这些属性：



这是所有型号点阵打印机都能理解的标准装置。某些打印机型号可以支持标准装置中不存在的其他命令，例如以每英寸20个字符的分辨率打印。为了能够将这些额外的属性发送到打印机，请使用报表中的“ESC-Command”对象/。

将“ESC-Command”对象放在报表页面上的正确位置，在使用非标准属性的任何对象之前（例如，在页面的左上角）。要设置命令，请在对象检查器中编辑对象的Command属性：

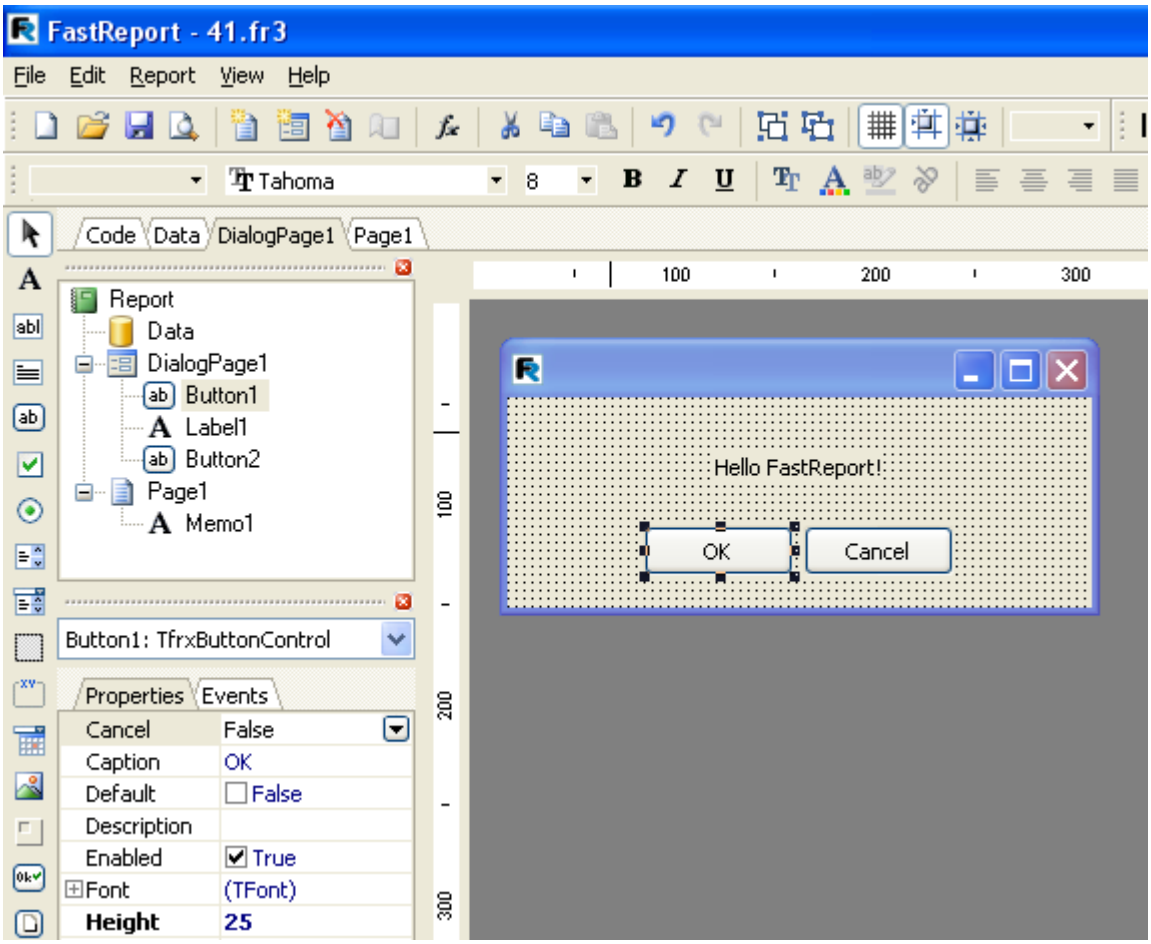


可以使用以下两种格式之一设置属性：十进制（例如#27#40）或十六进制（例如1B28）。

章节 13

对话形式

与通常的报表设计页面一样，您可以在报表中使用对话表单。使用设计器工具栏中的/按钮在通常的报表设计器中创建对话表单：该按钮为报表添加了一个新的对话页面选项卡。切换到对话框页面选项卡时，设计器工作区将更改为显示表单，对象工具栏将更改为显示可以放置在表单上的控件对象：



13.1 控制

要在报表中使用Dialogue表单控件，请在Delphi项目表单上的FastReport组件面板中放置“TfrxDialogControls”/组件。或者，将“frxDCtrl”添加到“Uses”列表中。然后，以下控件可用于报表：

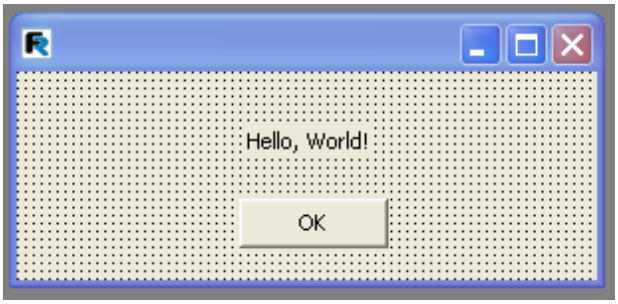
元件	名称	描述
A	TfrxLabelControl	控制在对话表格上显示固定文本
abl	TfrxEditControl	控制使用键盘输入文本

	TfrxMemoControl	控制使用键盘输入多行文本
	TfrxButtonControl	控制代表一个按钮
	TfrxCheckBoxControl	控制代表一个标志，有两个状态 – 启用或禁用；有一个相邻的标签
	TfrxRadioButtonControl	控制代表一个单选按钮。表单上必须有多个单选按钮控件
	TfrxListBoxControl	控制表示可以从中选择一个项目的列表
	TfrxComboBoxControl	控制表示可从中选择一个项目的下拉列表
	TfrxDateEditControl	控制表示具有日期条目的下拉日历的字段
	TfrxGroupBoxControl	控制表示可以包含其他对象的标记框
	TfrxPanelControl	控制表示可以包含其他对象的面板
	TfrxBitBtnControl	控制表示包含图片的按钮
	TfrxSpeedButtonControl	控制表示包含图片的速度按钮
	TfrxMaskEditControl	控制使用键盘输入文本并符合模板
	TfrxCheckListBoxControl	控制表示通过复选框选择的项目列表
	TfrxBevelControl	用于在表单上显示行和框架的控件
	TfrxImageControl	控制表示“BMP”，“ICO”，“WMF”或“EMF”格式的图片

所有控件都与Delphi中使用的控件类似。有关每个控件的属性，事件和方法的信息，请参阅FastReport组件帮助。

13.2 “你好，世界！”报表

在此示例中，我们将创建一个报表，在创建预览之前，使用对话框窗口显示问候窗口。在Delphi中创建一个新项目，并在表单上放置“TfrxReport”和“TfrxDialogControls”组件。通过双击“TfrxReport”组件打开FastReport设计器，并在报表中添加一个对话框。在表单上放置“TfrxLabelControl”和“TfrxButtonControl”对象：



设置对象属性：

TfrxLabelControl：
Caption ='你好，世界！'

T*frxButtonControl：*
Caption ='OK'默认=
True ModalResult =
mrOk

为表单设置"BorderStyle = bsDialog"属性。控件和表单都具有与相应Delphi控件相同的属性集。

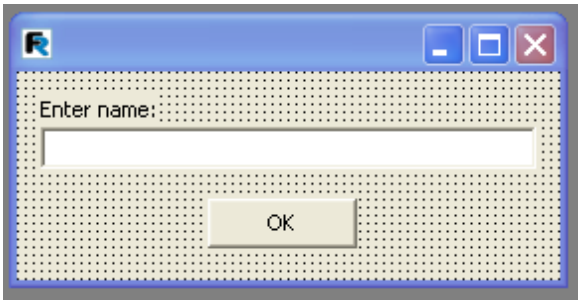
完成对话框设计后，返回到报表设计Page1选项卡，并在页面上放置包含问候语的"文本"对象。预览报表，您将看到对话框：



单击"确定"按钮后，对话框将关闭，并构建并显示报表。如果通过单击系统菜单"X"按钮关闭对话框，则不会构建报表。 FastReport的工作方式如下：如果报表中有对话表单，则仅在使用ModalResult = mrOk关闭每个对话框后构建报表，即在此示例中单击OK按钮。这就是为什么按钮的"ModalResult"属性必须设置为"mrOk"。

13.3 输入参数并将其传输到报表中

让我们让这个例子更复杂，以展示如何将对话中输入的值传输到报表中。修改对话框如下：



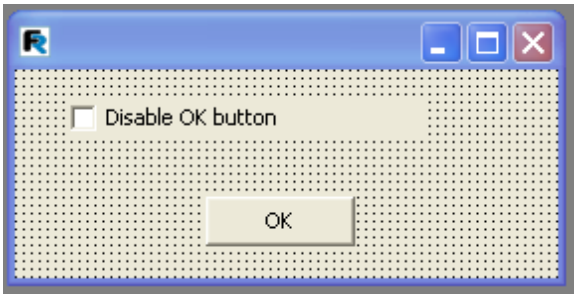
在报表Page1上放置一个包含此文本的“Text”对象，并将其“AutoWidth”设置为True：

您输入了：[Edit1.Text]

预览报表并确保在报表中成功显示输入的文本。可以以类似的方式访问对话中的其他对象。由于每个对象的名称在整个报表中都是唯一的，因此可以在同一报表中的任何位置使用。

13.4 与控件交互

通过使用脚本，复杂的逻辑可以合并到对话的操作中。让我们用一个简单的例子来说明这一点：修改表单如下：



双击“CheckBox”对象以创建“OnClick”事件处理程序并输入以下脚本：

PascalScript：

```
程序CheckBox1OnClick（发件人：TfrxComponent）；
开始
  Button1.Enabled：=不是CheckBox1.Checked;
结束；
```

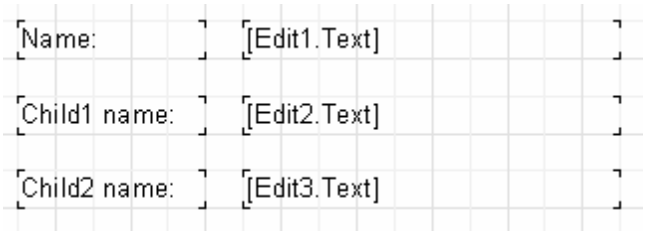
C ++脚本：

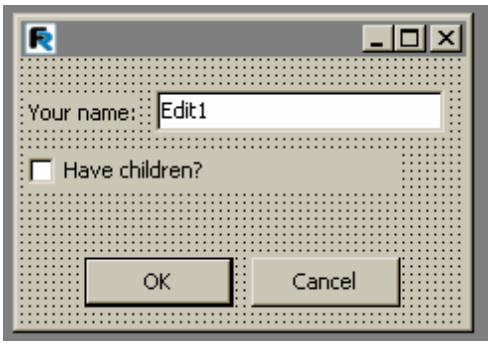
```
void CheckBox1OnClick (TfrxComponent Sender)
{
  Button1.Enabled =! CheckBox1.Checked;
}
```

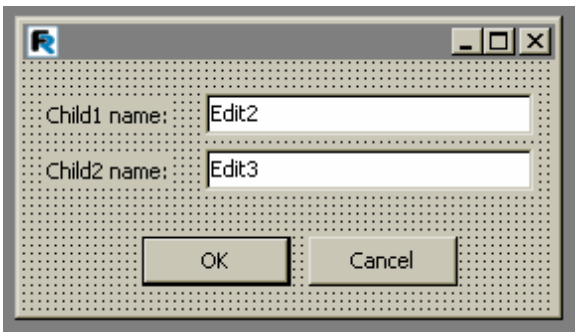
此代码与Delphi中使用的代码相同。在运行报表时，按钮响应复选框的状态。

13.5 几种对话形式

让我们看看有两个对话的报表是如何工作的。创建包含两个对话框和一个设计页面的报表：







将“确定”和“取消”按钮的“ModalResult”属性分别设置为mrOk和mrCancel。现在运行报表。首先，用户将被要求回答第一个对话（姓名，任何孩子）的问题，然后点击OK，从第二个对话（孩子的名字）。在第二个对话框中单击OK后，将构建报表。 FastReport以这种方式处理多个对话：对话以其创建顺序显示；每个对话框仅在使用“ModalResult = mrOk”关闭前一个对话框后显示（在本例中，单击每个OK按钮）。如果使用“取消”按钮或系统菜单“X”按钮关闭任何对话框，则会阻止构建报表。

13.6 管理对话形式

在前面的示例中，无论是否“检查孩子”，都会显示第二个对话框。让我们展示如何在未选中此标志时隐藏第二个对话框。在第一个对话框上为OK按钮创建一个“OnClick”处理程序（双击按钮创建处理程序）：

PascalScript：

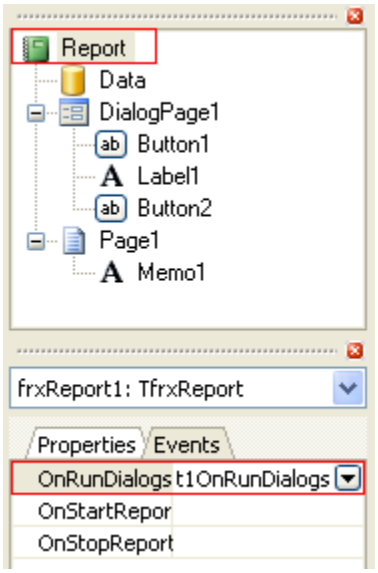
```
过程Button1OnClick（发件人：TfrxComponent）；
开始
  DialogPage2.Visible：= CheckBox1.Checked;
结束；
```

C ++脚本：

```
void Button1OnClick (TfrxComponent Sender)
{
  dialogpage2 CheckBox1。检查可见=；。
}
```

如果未选中该标志，则此代码隐藏第二个对话框（DialogPage2）。预览报表以查看其是否正常工作。

管理表单的另一种方法是使用“OnRunDialogs”报表事件。要创建此事件处理程序，请在报表树或对象检查器中选择“报表”对象，然后切换到对象检查器中的“事件”选项卡。双击“OnRunDialogs”事件以创建处理程序：



在处理程序中编写以下代码：

PascalScript：

```
procedure frxReport1OnRunDialogs (var Result: Boolean) ;
开始
  结果: = DialogPage1.ShowModal = mrOk;
  如果结果然后开始
    如果CheckBox1.Checked那么
      结果: = DialogPage2.ShowModal = mrOk;
  结束; 结
束;
```

C ++脚本:

```
void frxReport1OnRunDialogs (bool&Result) ;
{
  dialogpage1.showmodal mrok result＝＝＝;
  如果 (结果)
  {
    if (CheckBox1.Checked)
      dialogpage2.showmodal mrok result＝＝＝;
  }
}
```

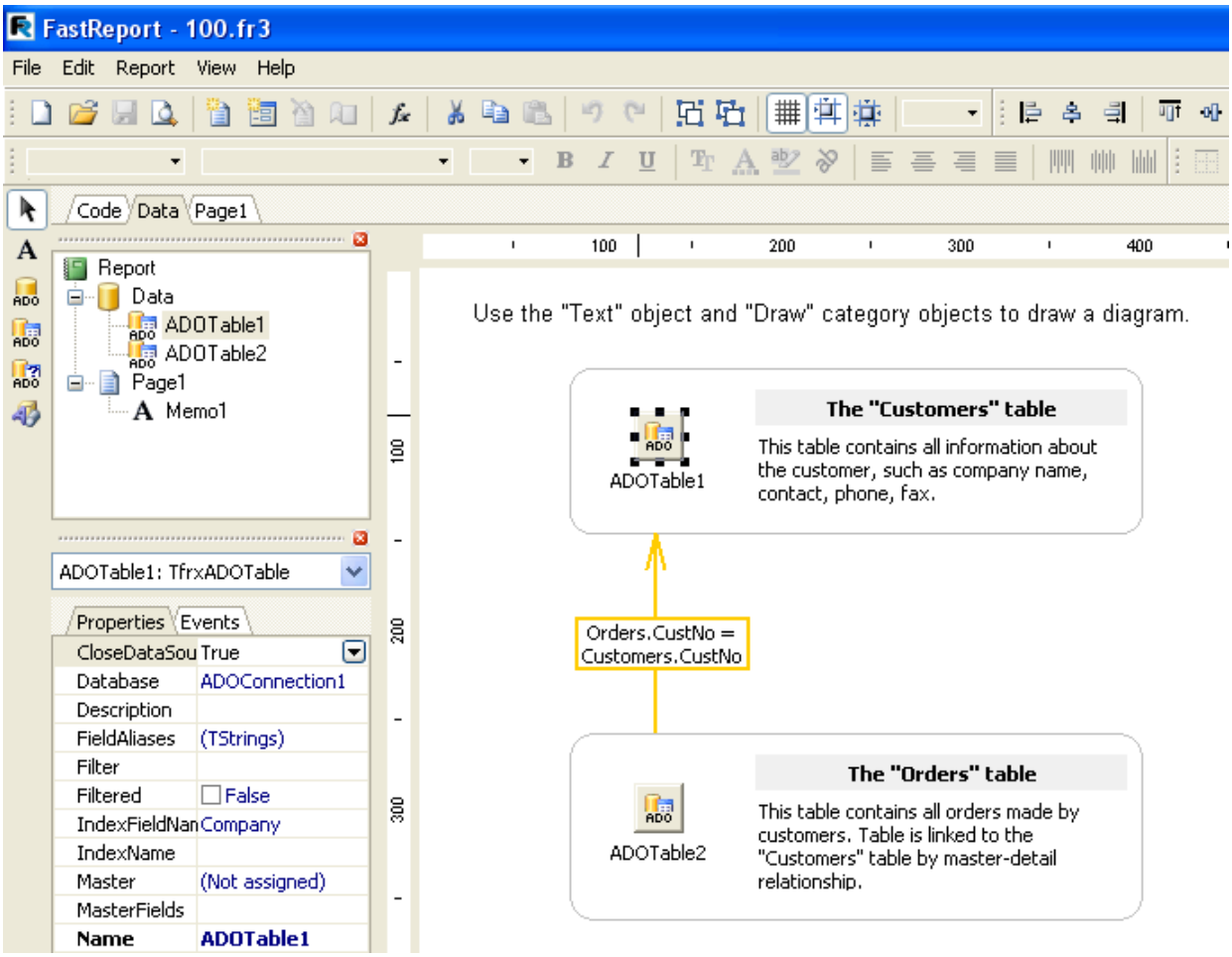
处理程序的工作方式如下: 显示第一个对话框: 如果通过OK按钮关闭, 则查看CheckBox1的状态: 如果此状态为Checked, 则显示第二个对话框: 如果第二个对话框通过 O K 按钮关闭则将Result设置为True。如果处理程序返回Result
= True然后构建预览: 如果Result = False, 则报表停止运行而不构建预览。

章节 14

数据访问组件

通常，大多数报表都基于来自数据库的数据。 Delphi提供了各种用于链接到DB数据的组件，而FastReport则使用这些链接。这里我们讨论使用“TTable”和“TQuery”组件作为报表的数据源，但通常可以使用任何TDataSet后代。

除了Delphi项目的数据访问组件，例如我们在目前的示例中使用的“TfrxDBDataset”，FastReport中有几个特定的数据库组件可以在报表中使用，哪些用于依赖于安装过程中做出的选择 FastReport中的数据访问原则与Delphi环境中使用的原理大致相同。就像在Delphi中一样，组件放在对话框中，其属性在对象检查器中设置。组件设计非常灵活，可以轻松创建支持不同数据库引擎的新组件（请参阅“开发人员手册”）。在“TfrxDesigner”组件的支持下，它们为应用程序的最终用户提供了在运行时设计报表的能力。



14.1 组件说明

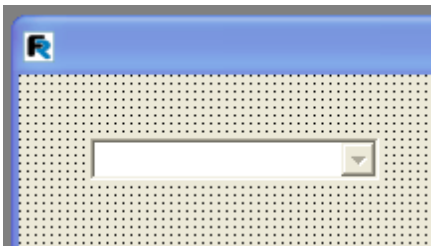
让我们看看组件如何通过ADO用于数据访问。通过将Delphi FastReport选项板中的“TfrxADOComponents”/组件添加到项目表单中，可以在FastReport中使用它们。当您切换到报表设计器中的“数据”选项卡时，对象工具栏上将显示以下对象：“TfrxADOTable”，“TfrxADOQuery”和“TfrxADODataBase”。这些组件在功能方面类似于相应的Delphi组件（“TADOTable”，“TADOQuery”和“TADOConnection”）。还有一个“TfrxDBLookupComboBox”控件，可用于对话形式。

图标	名称	描述	用于
	TfrxDBLookupComboB牛	用于从目录中选择值的控件	对话形式
	TfrxADOTable	用于访问数据库表的控件	“数据”标签
	TfrxADOQuery	用于执行SQL查询的控件	“数据”标签
	TfrxADODataBase	用于连接数据库的控件	“数据”标签

我们来看看每个组件。

14.1.1 TfrxDBLookupComboBox

该组件用于从查找数据集中选择值。



该组件具有以下属性：

属性	描述
数据集	控件所连接的数据源
ListField	控件中显示的DB字段的名称
关键字段	标识所选记录的DB关键字段的名称
核心价值	从列表中选择返回的DB键字段的值
文本	列表中显示的DB列表字段的值
AutoOpenDataSet	什么时候 真正 该 连接的数据 资源 is 在对话的OnActivate事件后自动打开

要将控件连接到查找数据集，请输入三个属性的值：“DataSet”，“ListField”和“KeyField”。

请注意，返回的值可通过“Text”或“KeyValue”属性获得，这两个属性都不会出现在对象检查器中。它们只能通过代码访问。可以使用“KeyValue”在代码中设置查找数据集中光标的初始位置。

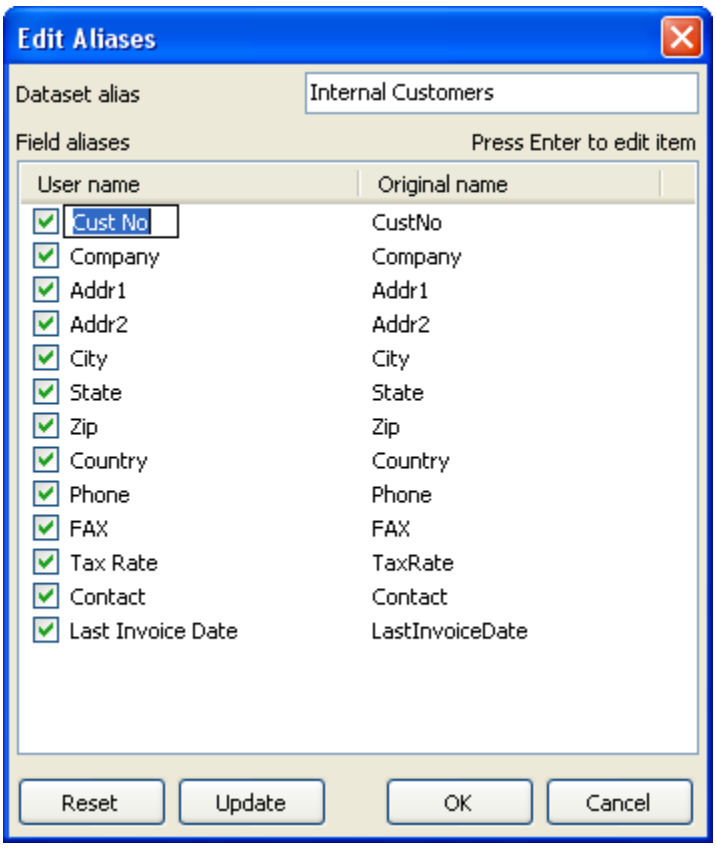
14.1.2 TfrxADOTable

该组件用于使用ADO访问数据库表。该组件具有以下属性：

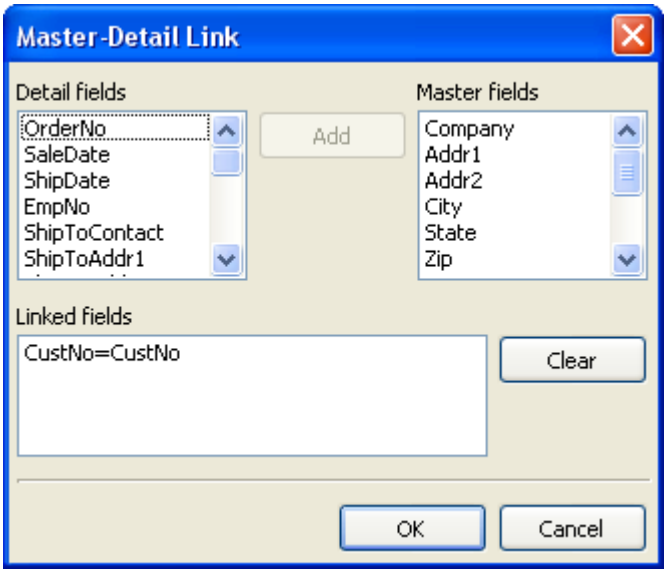
属性	描述
数据库名称	连接名称（TfrxADODatabase组件的名称）
FieldAliases	允许为数据集字段设置别名
过滤	过滤记录的表达式
过滤	是否过滤
IndexFieldNames	索引字段的名称（用于排序）
索引名	二级索引名称
MasterFields	与主数据集关联的字段
主	主数据集
表名	数据库表名
用户名	代码中使用的数据集的别名

这些属性类似于Delphi的“TADOTable”组件。要将组件连接到DB表，只需设置“DatabaseName”和“TableName”属性。通过将“Active”属性设置为True或通过调用“Open”方法打开表。

从对象检查器或双击组件打开“FieldAliases”属性编辑器。编辑器允许选择哪些字段可用于代码以及为这些字段和整个数据集设置别名。



“MasterFields”属性编辑器用于在两个表之间创建主从细节连接。要使用master-detail关系连接两个表，必须在依赖表的“Master”属性中设置第二个表，并打开“MasterFields”属性编辑器。如果表具有要使用的二级索引，则预先设置“IndexName”属性。



编辑器直观地绑定数据集的“主”和“详细信息”字段。当两个数据集以主数据集内的“主 - 细节”关系运动连接时

自动过滤细节数据集，以便仅显示属于主数据集的当前记录的记录。

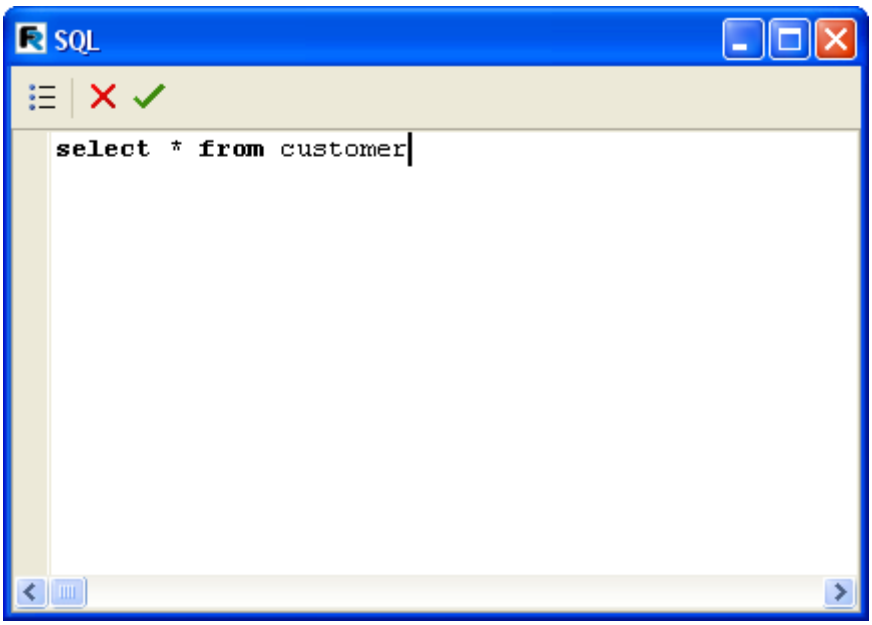
要连接两个数据集中的字段，请从左侧列表中选择一个字段（详细数据集），然后从右侧列表中选择一个字段（主数据集），然后单击“添加”按钮。然后，两个字段之间的链接显示在底部列表中。要清除底部列表，请使用“清除”按钮。链接字段必须是兼容类型并且要编制索引。

14.1.3 TfrxADOQuery

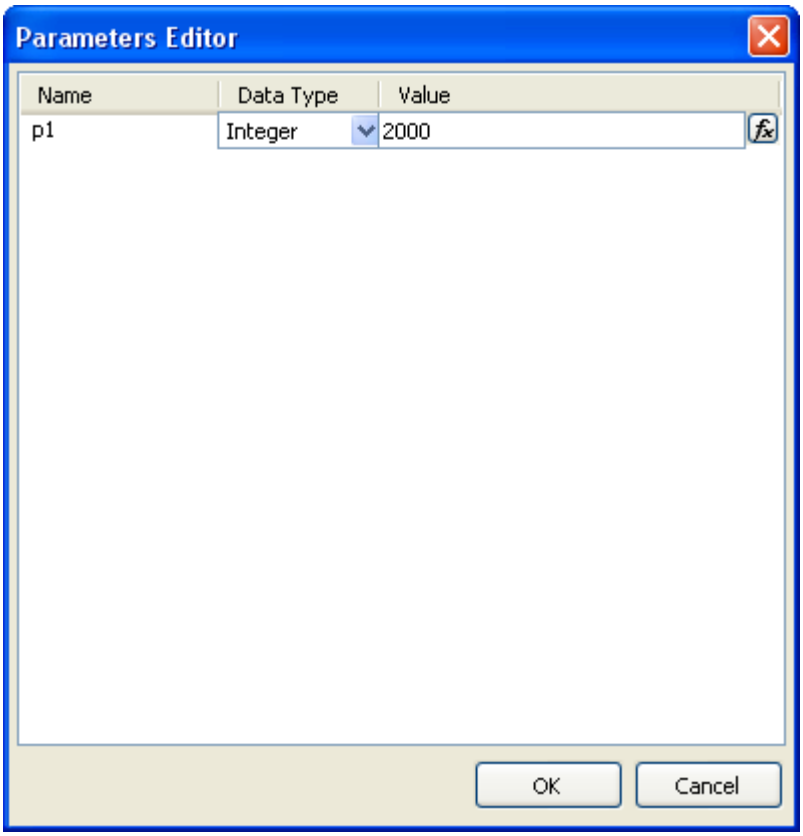
该组件在DB上执行SQL查询。它具有以下属性：

属性	描述
数据库名称	连接名称（TfrxADODatabase组件的名称）
FieldAliases	允许为数据集字段设置别名
过滤	过滤记录的表达式
过滤	是否过滤
主	主数据集
Params	查询参数列表
SQL	查询文字
用户名	代码中使用的数据集的别名
IgnoreDupParams	当True允许具有重复名称的参数时。不会在参数编辑器中编辑Query参数的名称

“活动”，“DatabaseName”，“FieldAliases”，“过滤器”，“过滤”和“主”属性类似于上述“TfrxADOTable”组件的属性。“SQL”属性具有自己的编辑器，用于输入SQL查询。



“Params”属性也有自己的编辑器。当Query文本包含参数时，它会启用。



参数可以是以下两种类型之一：其值可以从主源派生，也可以设置为离散值（绝对值，如上所示，或链接到变量或对象属性）。

当从数据主集导出参数时，必须设置“TfrxADOQuery.Master”属性。此数据集必须包含与参数同名的字段。既不必指定参数类型也不指定其值。

14.1.4 TfrxADODataBase

该组件用于连接数据库。它的功能类似于“TADOConnection”Delphi组件。该组件具有以下属性：

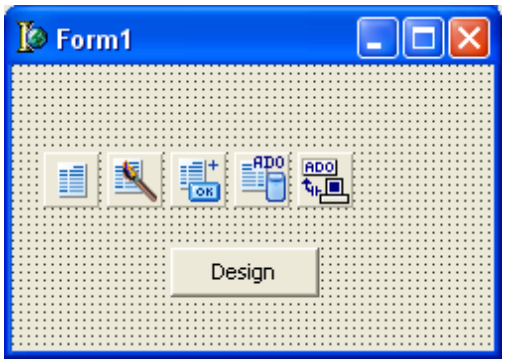
属性	描述
连接的	当True时，连接被激活
数据库名称	ADO连接字符串
LoginPrompt	是否在连接到DB时提示输入密码

“LoginPrompt”属性定义在连接到DB时是否提示输入密码。当“LoginPrompt”为False时，ADO连接字符串中必须包含用户名和密码。

14.2 报表设计器

让我们看一下在运行时使用数据访问组件的简单报表的设计。它将使用FastReport附带的Demo.mdb数据库作为数据源 - {FR}
\演示\ MAIN \ demo.mdb。

创建一个新的Delphi项目，并在表单中添加“TfrxReport”，“TfrxDesigner”，“TfrxDialogControls”，“TfrxADOComponents”，“TADOConnection”和“TButton”组件。



通过双击“TADOConnection”，并选择“构建连接字符串”，提供程序（“Microsoft Jet 4.0 OLE DB提供程序”）和数据库（{FR} \ Demos \ Main \ demo.mdb）来设置数据库连接。单击确定关闭连接对话框并在此处设置组件属性：

ADOConnection1:
loginprompt =假

frxADOComponents1:
DefaultDatabase = ADOConnection1

为“设计”按钮定义以下处理程序：

```
程序TForm1.Button1Click (发件人: TObject) ;
开始
  frxreport1.designreport;
结束;
```

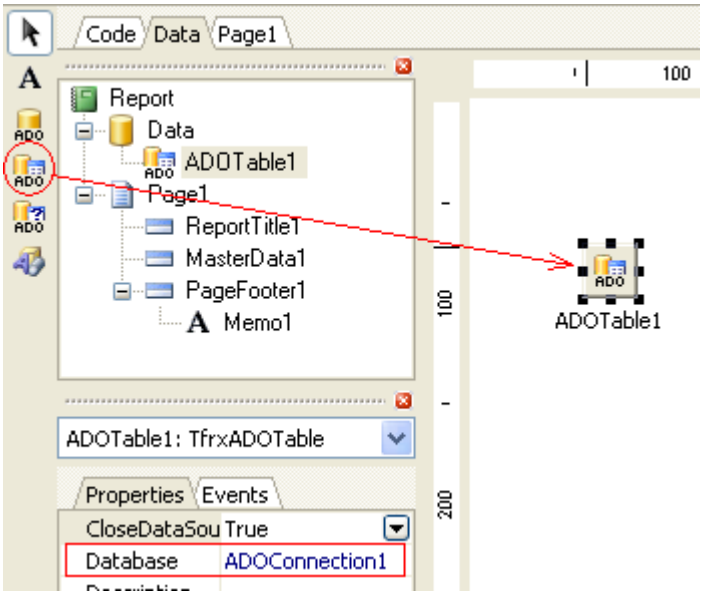
之后，编译并运行该项目。这就是创建最终用户运行时报表设计器所需要做的全部工作。

单击“设计”按钮后，将打开FR报表设计器，其中包含空白报表。让我们看一下这个环境中简单报表的设计。

14.3 简单报表“列表”类型

此报表将包含来自单个数据库表的数据。要构建报表，请执行以下步骤：

单击设计器工具栏上的“新建报表”按钮/ - FastReport创建一个包含“代码”，“数据”和“Page1”选项卡的空报表。切换到“数据”选项卡并在页面上放置“ADO表”组件：



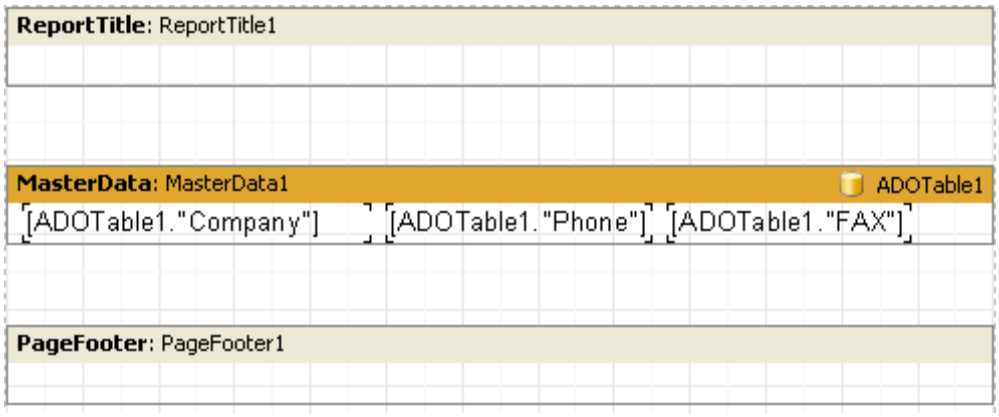
请注意，“Database”属性已连接到我们的数据库，因为这是在“TfrxADOComponents.DefaultDatabase”属性中指定的。但表名必须是

现在设定：

TableName ='客户'

切换到Page1选项卡。通过双击“MasterData”段并在对话框中选择“ADOTable1”将其连接到桌面。

将下面显示的字段从“数据树”窗口拖到报表页面，然后看起来大致如下所示：



使用工具栏上的“预览”按钮/预览报表。

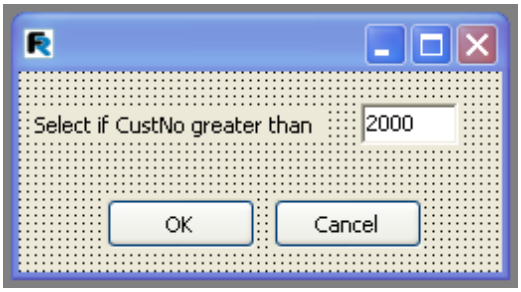
14.4 使用参数查询报表

让我们设计一个更复杂的报表，在报表创建之前，在对话框中请求查询参数。使用与上一节中相同的项目，然后单击报表设计器中的“新建报表”按钮以清除旧报表。

切换到“数据”选项卡并在页面上放置“ADO查询”组件。双击它以打开其编辑器并输入以下SQL文本：

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

在报表中添加一个对话框，并在对话框中放置一个“标签”，一个“编辑”和两个“按钮”组件：



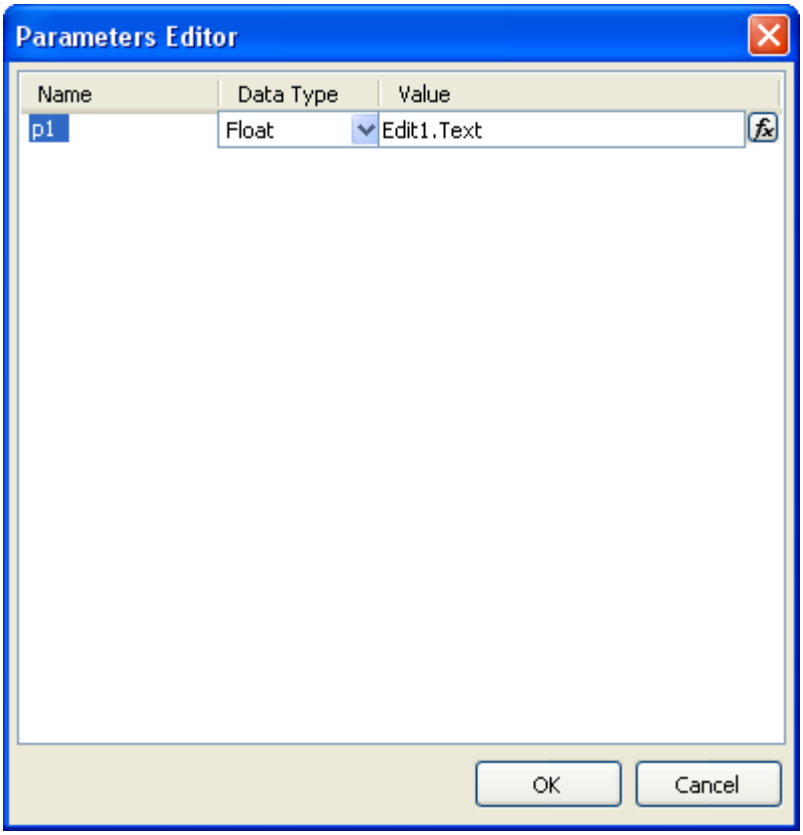
设置组件属性：Label1：
Caption ='选择CustNo是否大于'

EDIT1：
Text = '2000'

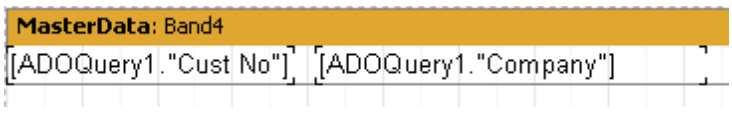
Button1的：
Caption = 'OK'
ModalResult = mrOk

BUTTON2：
Caption ='取消'ModalResult
= mrCancel

打开"Query"组件的 "Params"属性编辑器并设置参数：

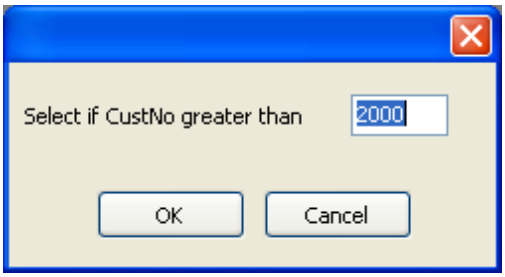


之后切换到报表设计Page1并创建报表，如上例所示：



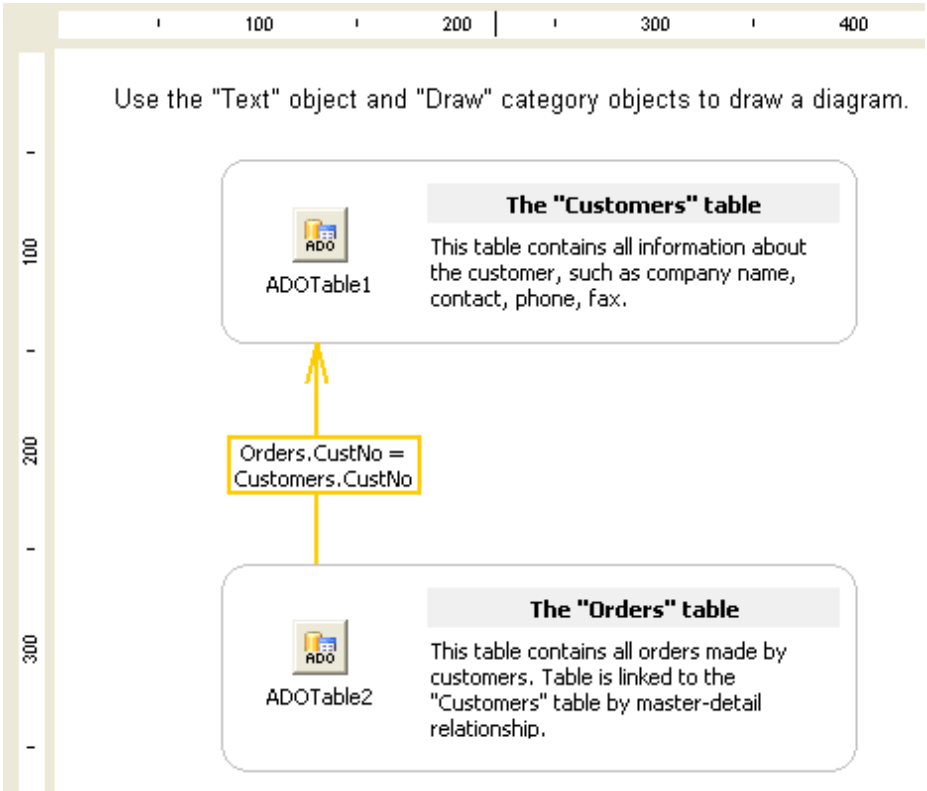
预览报表时，将显示提示输入客户编号的对话框。后

输入一个数字，并使用创建报表的OK按钮关闭对话框。显示CustNos大于输入的所有客户。



14.5 其他有用的功能

“Text”和“Draw”元素可以放在“Data”页面上。使用这些元素可以绘制简单的解释图，如下所示：



章节 15

报表继承

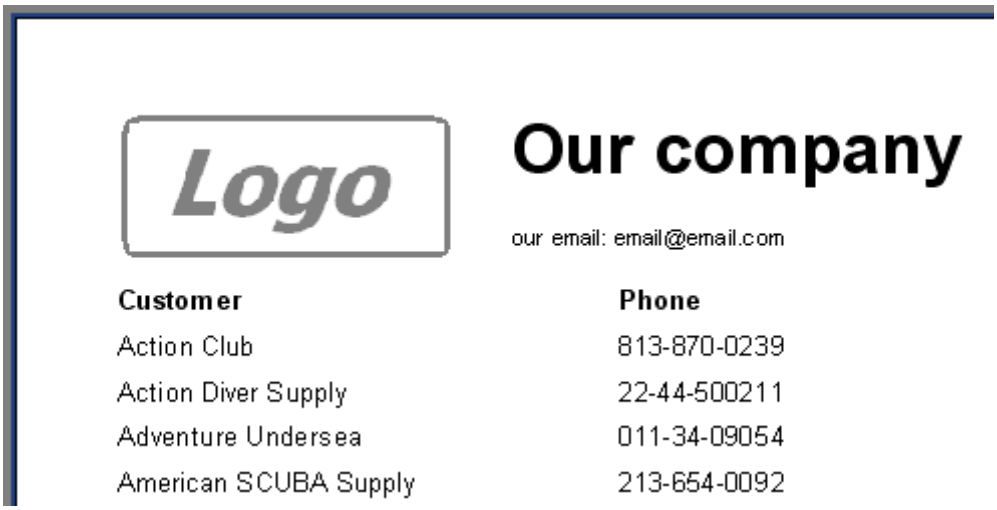
通常，一组报表共享一些常见数据 – 例如，带有公司徽标的页眉/页脚或其他数据，如电子邮件或地址等。如果出现某些公司数据需要更改的情况，例如电子邮件地址，则它必须在每个报表中完成！为避免这种繁琐的任务，可以使用报表继承。什么是报表继承？

例如，报表通常包含通常放在报表标题和/或页眉中的元素（徽标，公司名称，电子邮件等）。可以设计基本报表，其仅包含这些共同元素。然后，所有其他报表都可以使用基本报表，从而包含所有常见元素，以及专门添加到每个报表中的其他元素。

如果需要更改某些内容（例如徽标或电子邮件），则会打开基本报表并在那里进行必要的更改。然后，将自动修改从基础报表继承的所有报表。实际上，当打开基于继承的报表时，首先打开基本报表，然后打开派生报表。

15.1 创建基本报表

让我们创建一个使用继承的简单报表。我们完成的报表应如下所示：

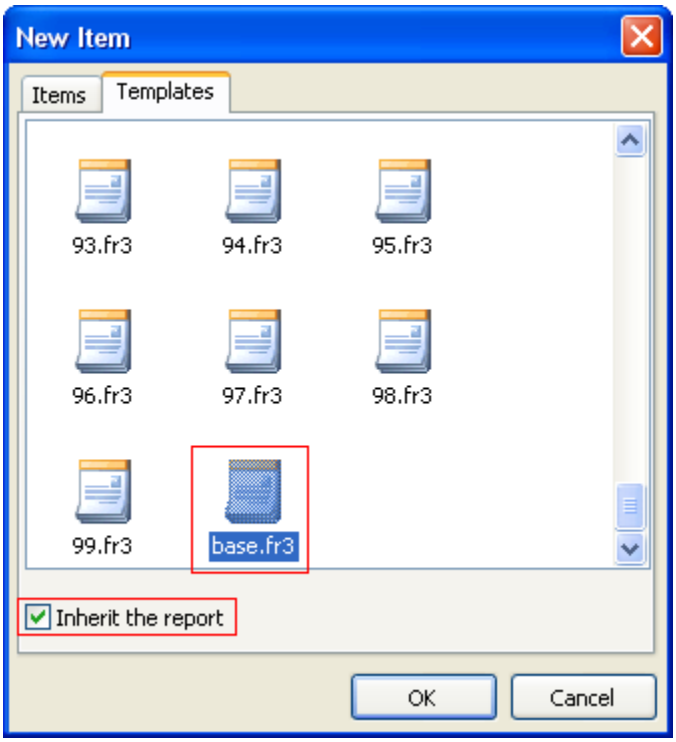


首先，必须创建基础报表。它必须包含哪些元素？它们是徽标位图，“我们公司”的标题和电子邮件地址。创建一个新报表并将公共对象放在“ReportTitle”中：



将报表另存为“base.fr3”。在哪个文件夹？这取决于您如何设置“TfrxDesigner”组件。默认情况下，FastReport 在包含应用程序的.exe 文件的文件夹中搜索基本报表。或者，可以在“TfrxDesigner.TemplateDir”属性中指定模板的文件夹。

现在创建派生报表。要执行此操作，请使用“文件>新建”，在对话框中选择“模板”选项卡并搜索基本报表（“base.fr3”）。单击“继承报表”复选框，然后按“确定”：



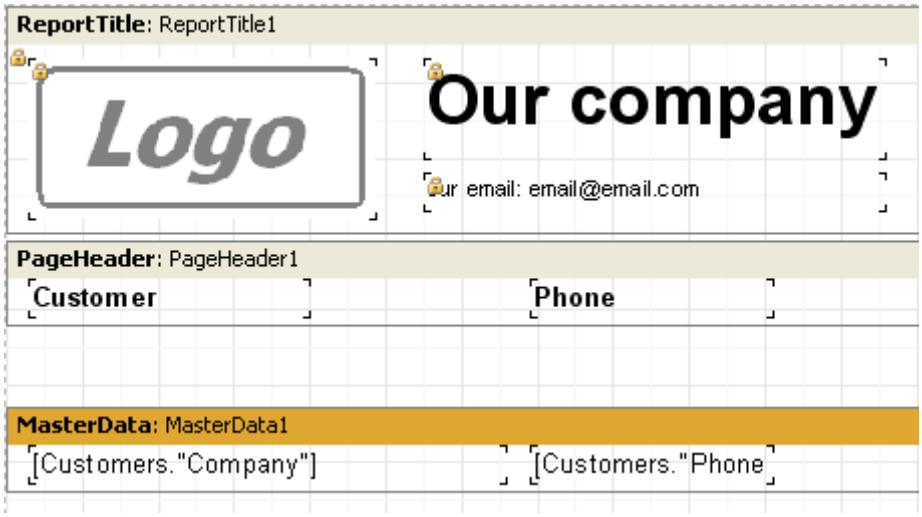
FastReport将创建一个包含基本报表中所有对象的报表。它们标有“锁定”符号：



“锁定”符号是什么意思？这意味着无法重命名或删除这些对象，也无法将它们移动到另一个段。可以对任何其他属性（例如文本，颜色或框架）进行更改。请注意，如果更改锁定对象的某些属性（例如颜色），则此更改将存储在派生报表中。如果随后在基本报表中更改此对象的颜色，则派生报表将忽略该更改。例如：打开派生报表，将“我们公司”的颜色更改为红色并保存报表。现在打开基础报表并将“我们公司”的颜色设置为绿色。打开派生报表时

再次，“我们公司”的颜色仍然是红色。因此，最好在基本报表中而不是在派生报表中更改具有“锁定”符号的对象的属性。

让我们完成我们的报表。所需的只是添加页眉和主数据段：



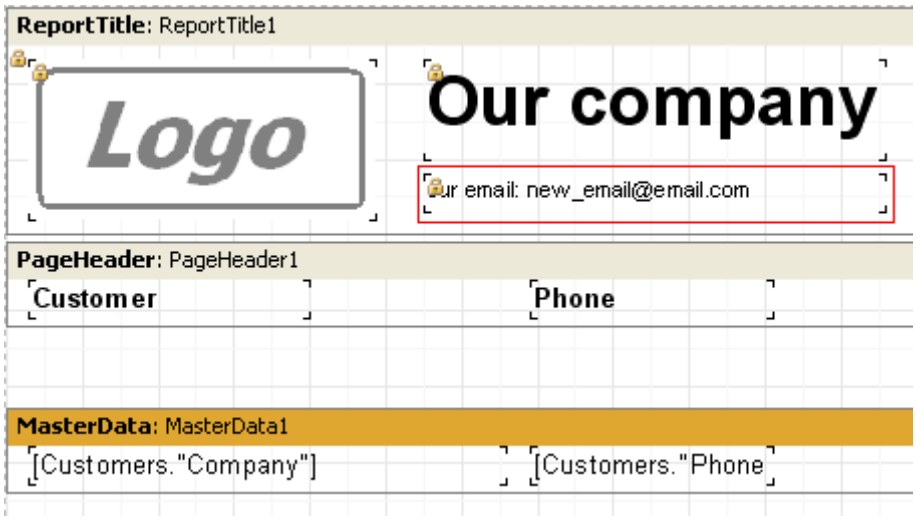
报表结束了。

15.2 更改基本报表

我们来看看更改基本报表。打开基本报表（在我们的示例中为“base.fr3”）并更改一些字段。我们改变电子邮件地址：



保存报表。现在打开派生报表，看到该报表中的电子邮件地址也已更改：

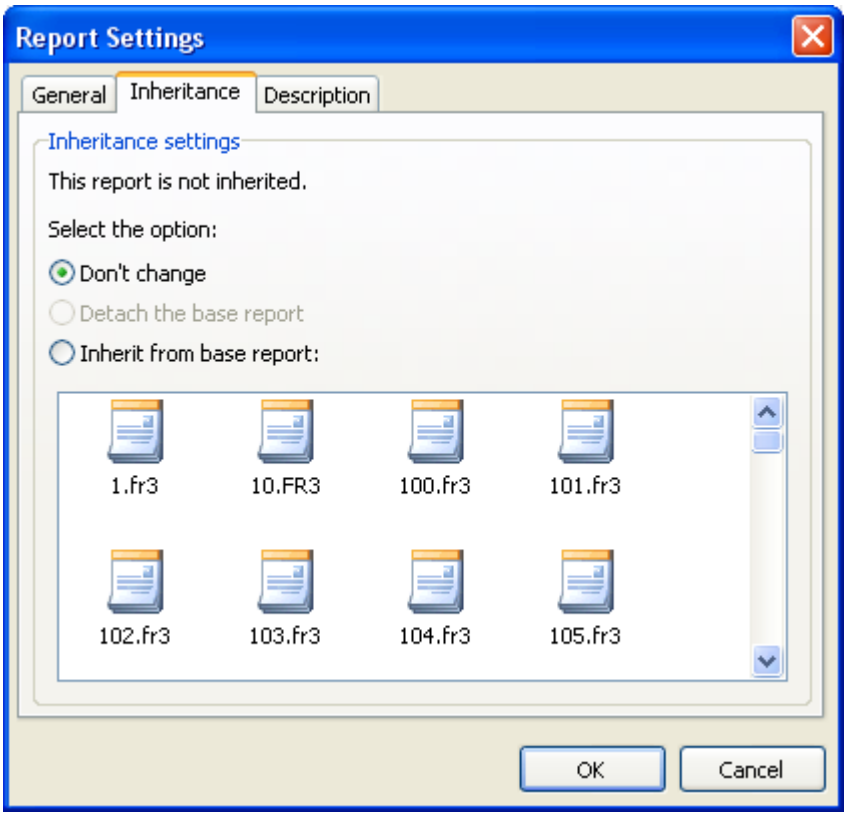


如果必须将某些对象添加到基本报表中该怎么办？有一个简单的规则：基本报表和派生报表不能包含任何具有相同名称的对象。在更改基本报表时，可能不知道有多少报表使用基本报表，也未知在这些派生报表中使用了哪些对象名称。因此，一个简单的策略是：在向基本报表添加对象时，使用“ReportName_ObjectName”之类的模板命名对象。在我们的示例中，在报表中添加“Text”对象，并将其名称设置为“BaseMemo3”。

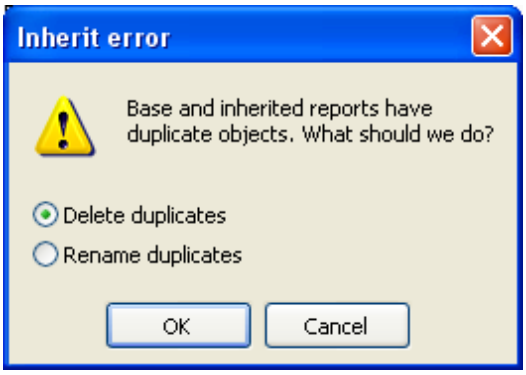
从基本报表中删除对象或移动它们没有任何限制。

15.3 继承控制

我们从头开始创建派生报表。如果要将现有报表更改为从其他报表继承，会发生什么？要执行此操作，请打开报表，然后从“报表>选项...”中选择“继承”选项卡：



选择"从基本报表继承"选项，然后从列表中选择基本报表。如有必要，请更改模板路径以查看不同的模板集。按OK，FastReport将合并这两个报表。可能会出现以下错误消息：

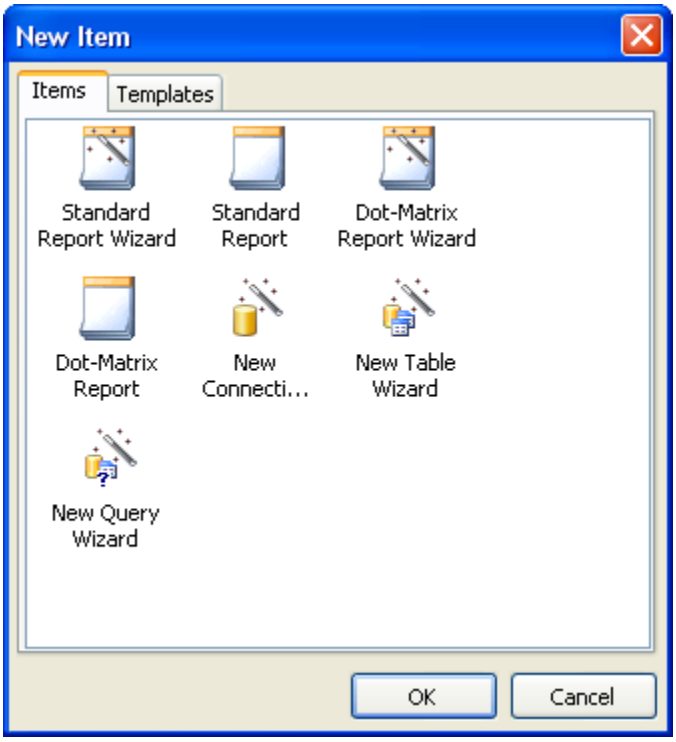


如果两个报表包含具有相同名称的对象，则会发生这种情况具有重复名称的对象可以从派生报表中删除，也可以在派生报表中重命名。

章节 16

向导

FastReport提供了一些简化报表创建过程的向导。向导位于“文件>新建...”下：



16.1 新报表向导

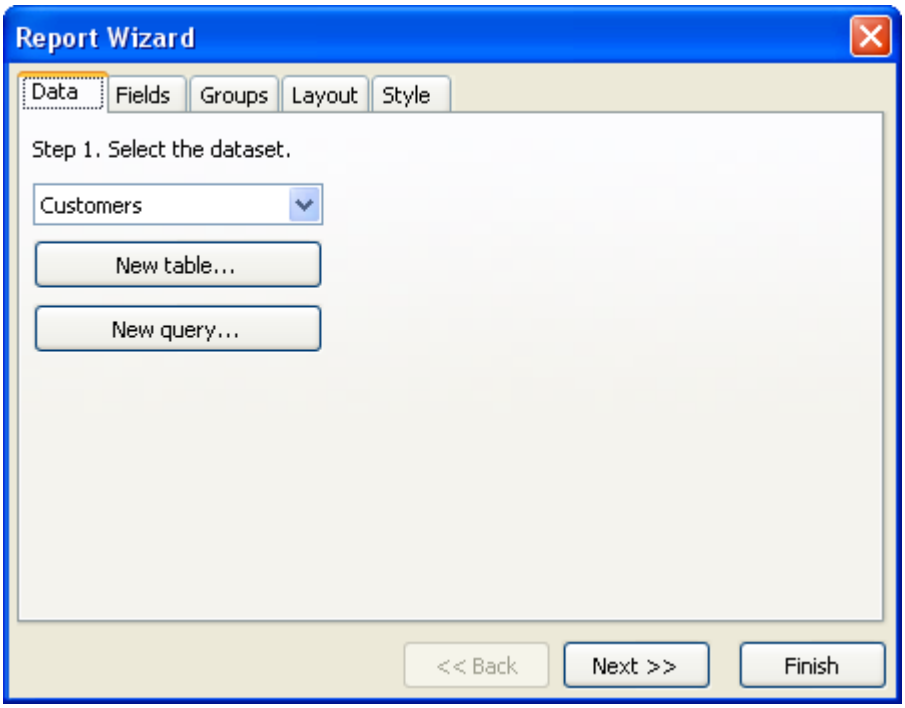
有两个图标可供帮助创建新报表的向导和两个用于直接创建新报表的图标：

- 标准报表向导
- 点阵报表向导
- 标准报表
- 点阵报表

“标准报表”和“点阵报表”类型的图标分别创建空标准和点阵报表（前面几节中有关于点阵报表的更多信息）。这些新报表包含一个空白页面。

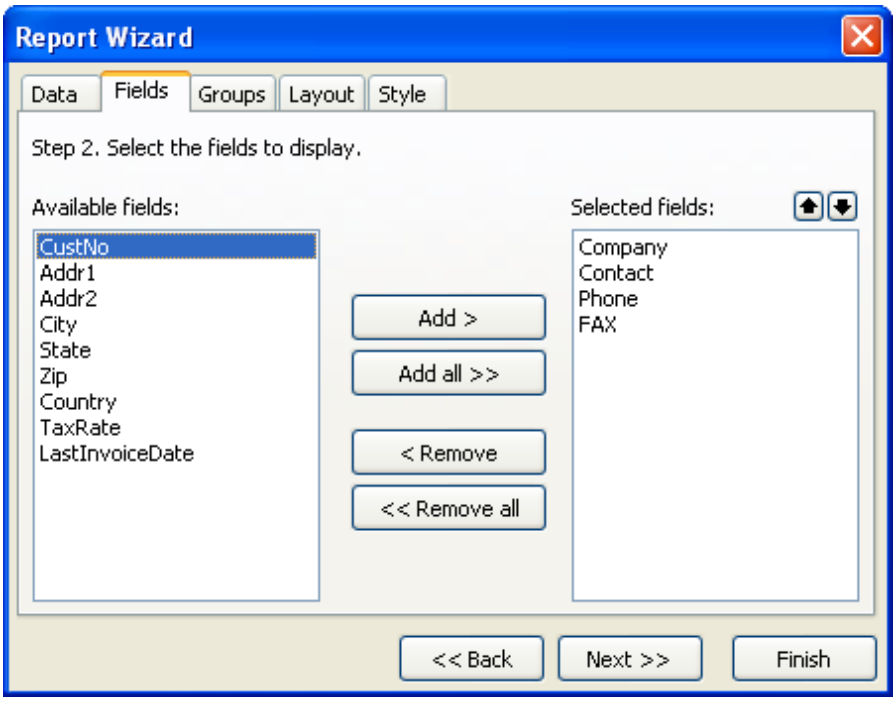
“标准报表向导”和“点阵报表向导”类型的图标指导了选择报表所需的数据集和字段，可选的组创建和数据布局选择的过程。让我们看看在“标准报表向导”的帮助下创建的报表。

选择“文件>新建...”和“标准报表向导”图标。出现报表向导对话框：



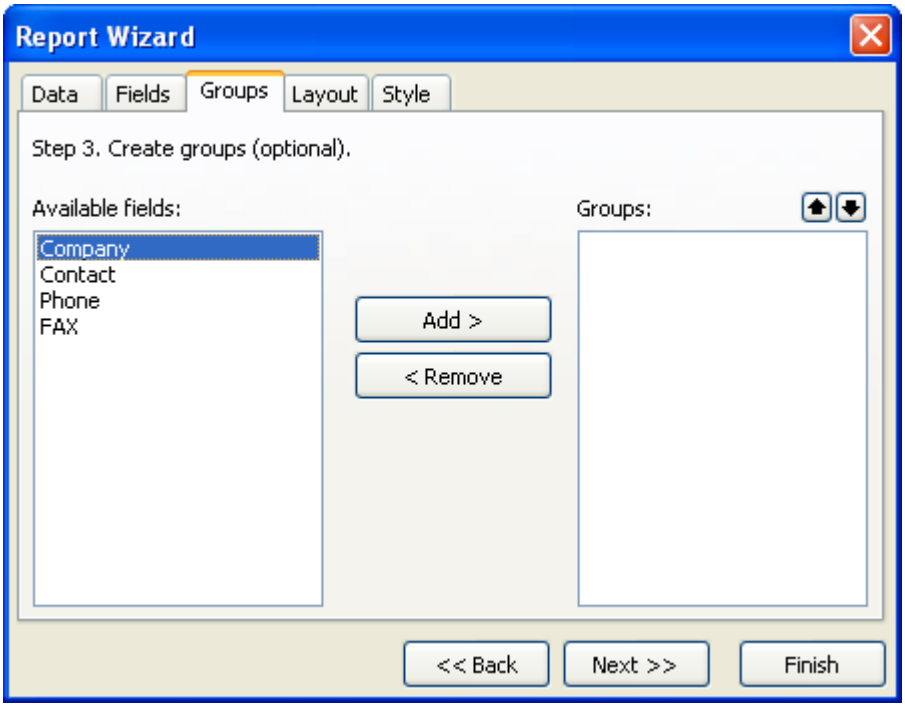
对话有几个标签。在第一个选项卡上，选择报表的数据源。此处列出了应用程序中可用的所有数据源（TfrxDBDataSet组件）。可以使用“新表”或“新查询”按钮创建新数据源（表或查询）。这些按钮打开“新建表/查询”向导（本章稍后介绍）。我们选择Customers表。然后按“下一步>>”按钮。

在下一个选项卡上，选择要在报表中显示的字段：



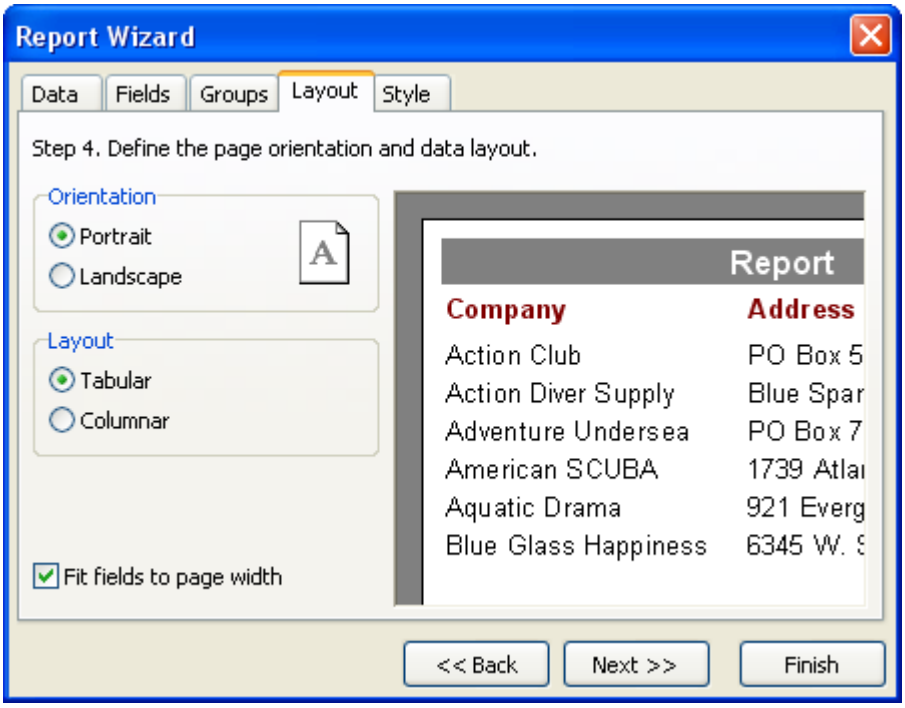
左侧是可用字段列表；在右侧，已选择要在报表中显示的字段列表。使用“添加>”，“添加全部>>”，“<删除”和“<<全部删除”按钮将字段从一个列表移动到另一个列表。使用/按钮在列表中向上或向下移动选定的字段。让我们将“公司”，“联系人”，“电话”和“传真”字段添加到所选字段列表中，然后按“下一步>>”按钮。

在下一个选项卡上，可以创建一个或多个组。 FastReport会将Group标题和组页脚区域添加到报表中。



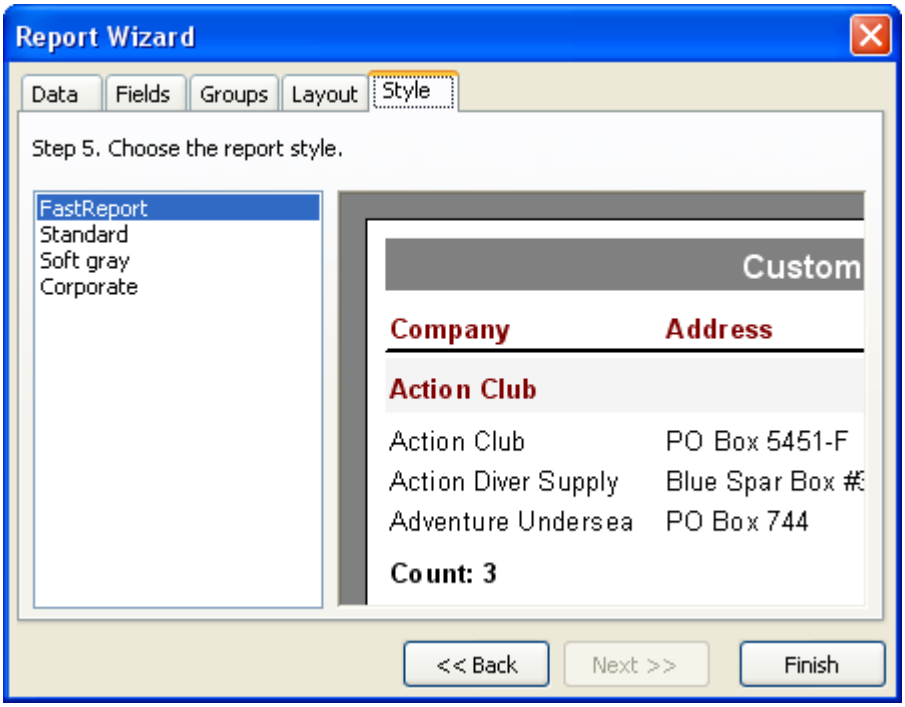
组创建是可选的。按“下一步>>”按钮跳过此处。

下一个选项卡设置页面方向和两种数据布局之一 - 表格和列式：

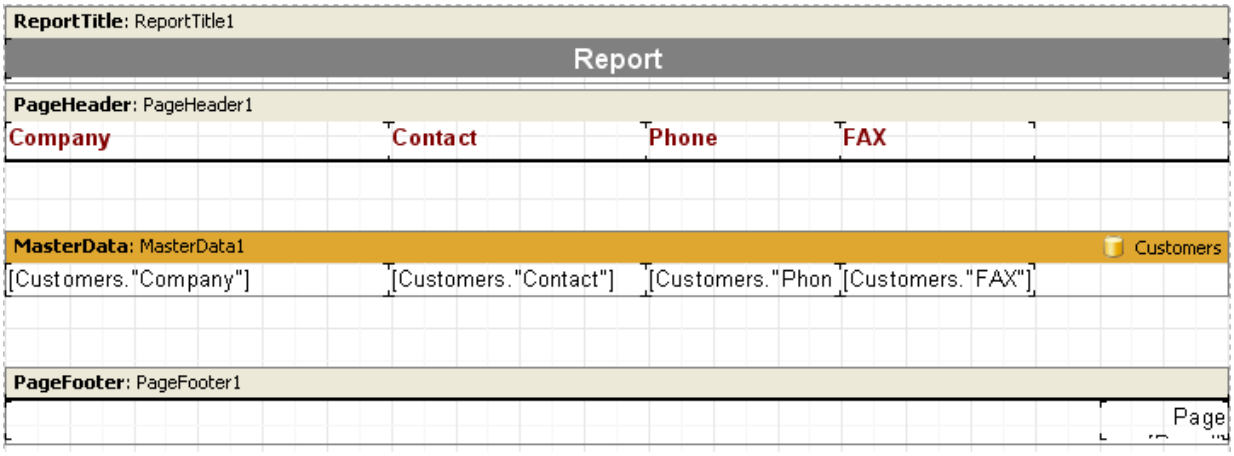


所选布局显示在对话框的右侧。

最后，最后一个选项卡列出了报表的可用颜色方案，再次在对话框的右侧进行了说明。



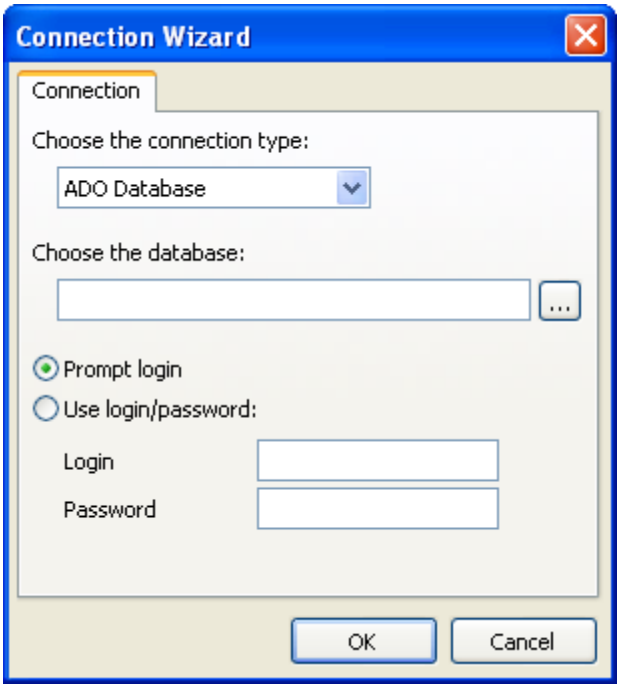
按下“完成”按钮后，向导将创建以下报表：



现在可以预览该报表。

16.2 新连接向导

此向导将新数据库连接添加到现有报表。如果要在报表中显示来自两个或多个数据库的数据，则可能需要两个或多个连接。该向导将数据库组件（例如“TfrxADODatabase”）添加到报表中。



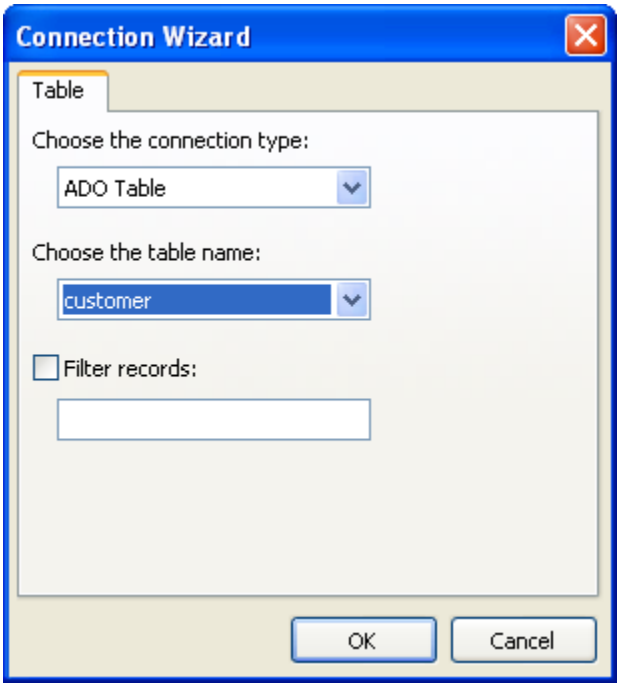
必须使用/按钮创建连接字符串，这将打开标准Windows连接对话框以设置数据库和连接参数。之后，如有必要，可以设置用户名和密码。

注意：可以手动建立新连接 – 只需放置“TfrxADODatabase”组件即可

报表的数据选项卡。

16.3 新表向导

该向导将新数据库表添加到现有报表中。



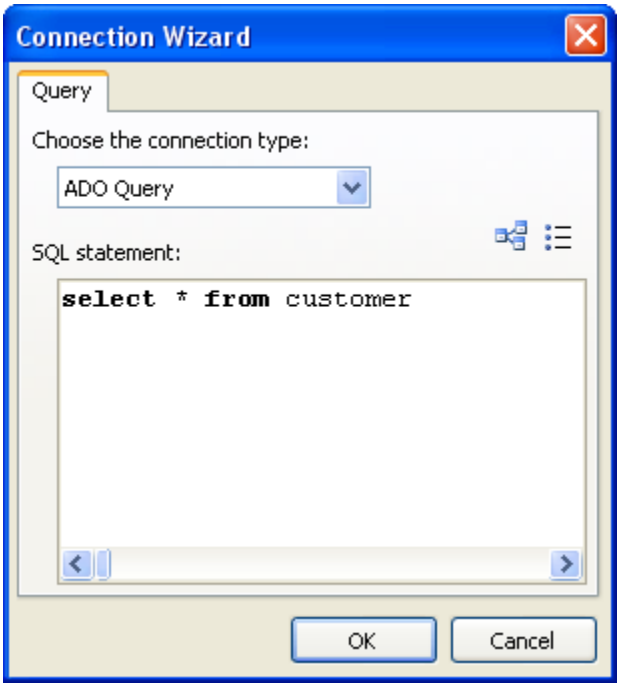
选择表名称。如果需要，还可以定义过滤器，例如：

(CustNo> 2000) 和 (CustNo <3000)

注意：可以通过在报表的“数据”选项卡上放置“TfrxADOTable”组件来手动创建新表。

16.4 新查询向导

此向导将新SQL查询添加到现有报表。

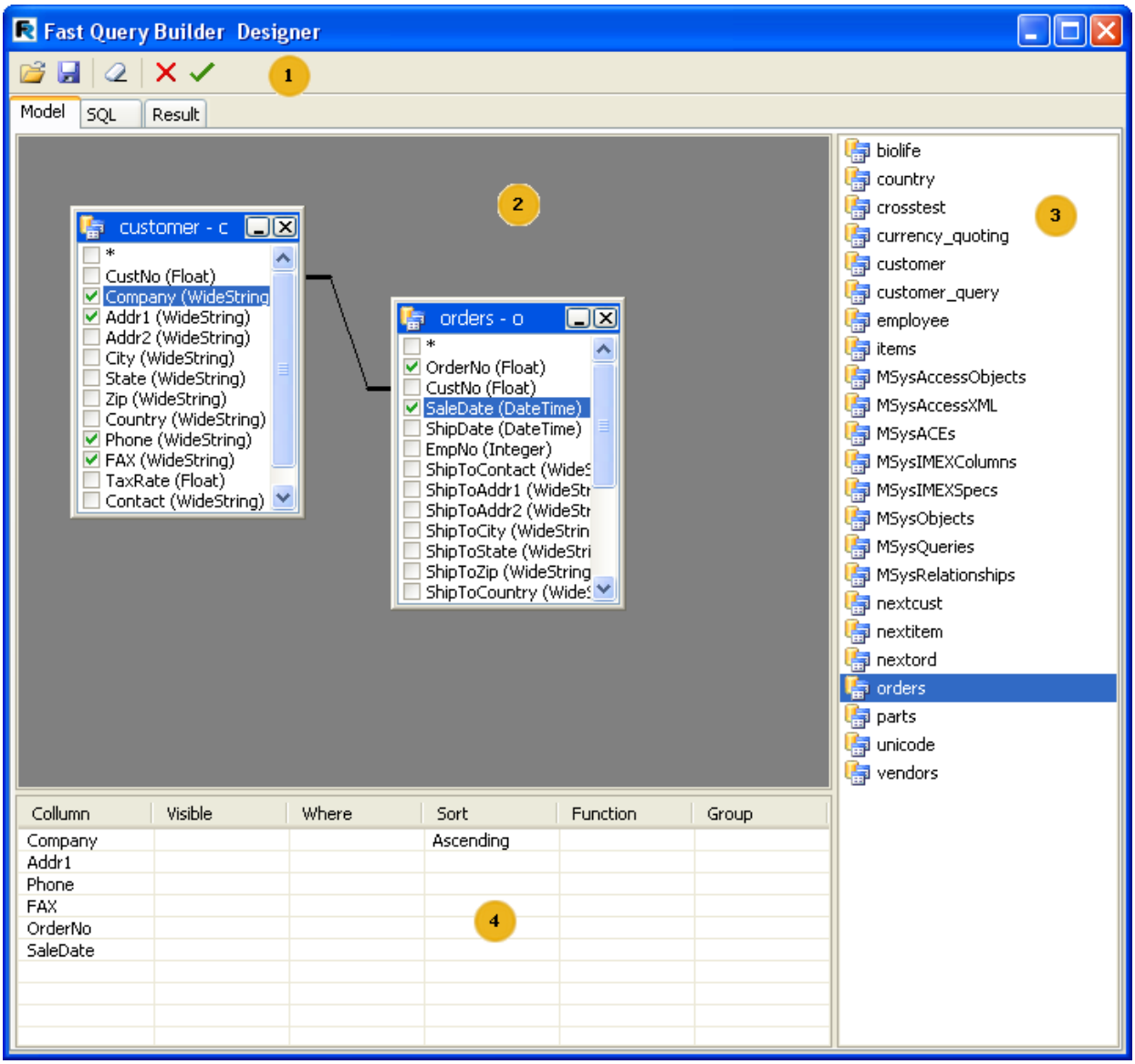


必须在此处编写SQL查询。可视化查询构建器可用于执行此操作 - 单击/按钮。查询构建器将在本章后面介绍。

注意：可以通过在报表的"数据"选项卡上放置"TfrxADOQuery"组件来手动创建新查询。

16.5 查询构造

FastQueryBuilder可用于直观地构成查询。 FastQueryBuilder包含在FastReport专业版和企业版中，或者可以作为独立产品购买。查询构建器以SQL语言可视化地构建查询。建设者如下图所示：



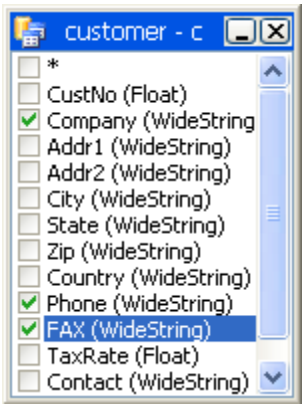
1 – 工具栏
2 – 构建器工作区
3 – 可用表列表
4 – 选定的表字段属性

工具栏:

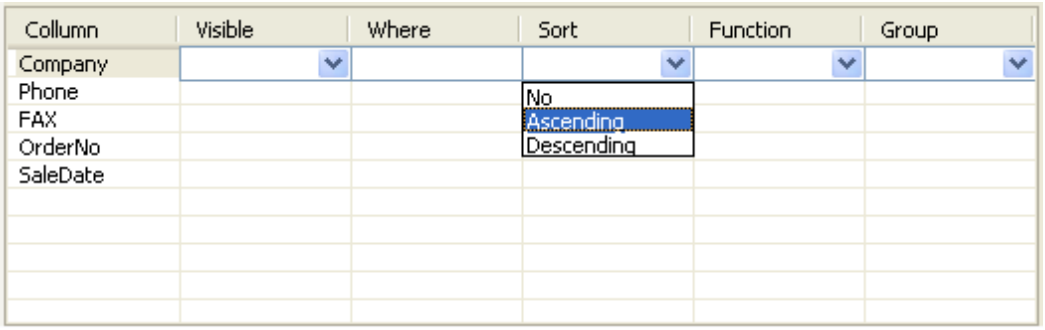
- 打开SQL文件
- 将查询保存到文件（查询图也保存在文件中）
- 清除构建器工作区
- OK按钮：保存并退出构建器
- 取消按钮：退出构建器而不保存

构建器的工作空间和可用表的列表支持拖放，即可以使用鼠标将表拖动到工作空间中。或者，双击可用表列表中的表。

要在查询中包含表中的字段，请在列表中选择它：

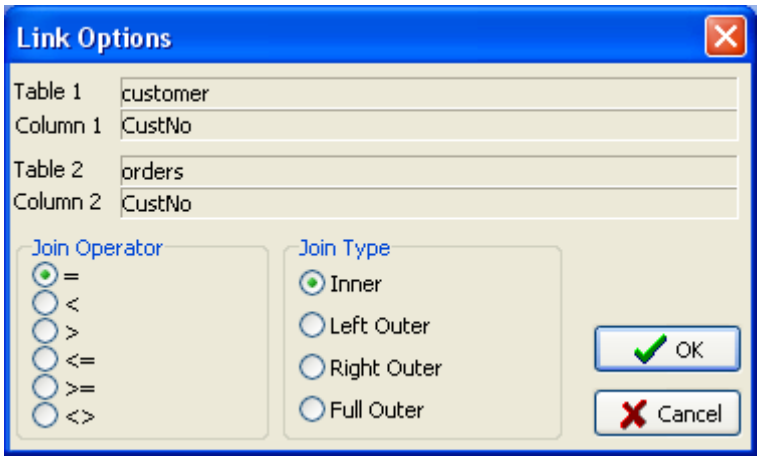


选定的字段显示在字段属性空间（4）中：



- 可见：字段是否出现在输出中
- 其中：字段选择条件，例如'> 5'
- 排序：字段的排序方向
- 功能：应用于字段的功能
- 集团：在该领域进行分组

通过在工作空间中的表之间"拖动"字段（2），出现“连接线”。加入字段时，将检查连接字段类型的兼容性。构建器可防止类型不兼容的字段之间的连接。要更改连接参数，请将光标悬停在“连接线”上，右键单击并选择“选项”项。将打开“链接选项”对话框，可以在其中配置连接，如下所示：

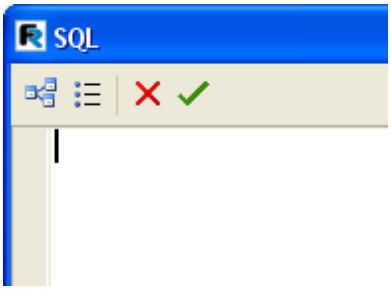


16.5.1 使用Query构造函数

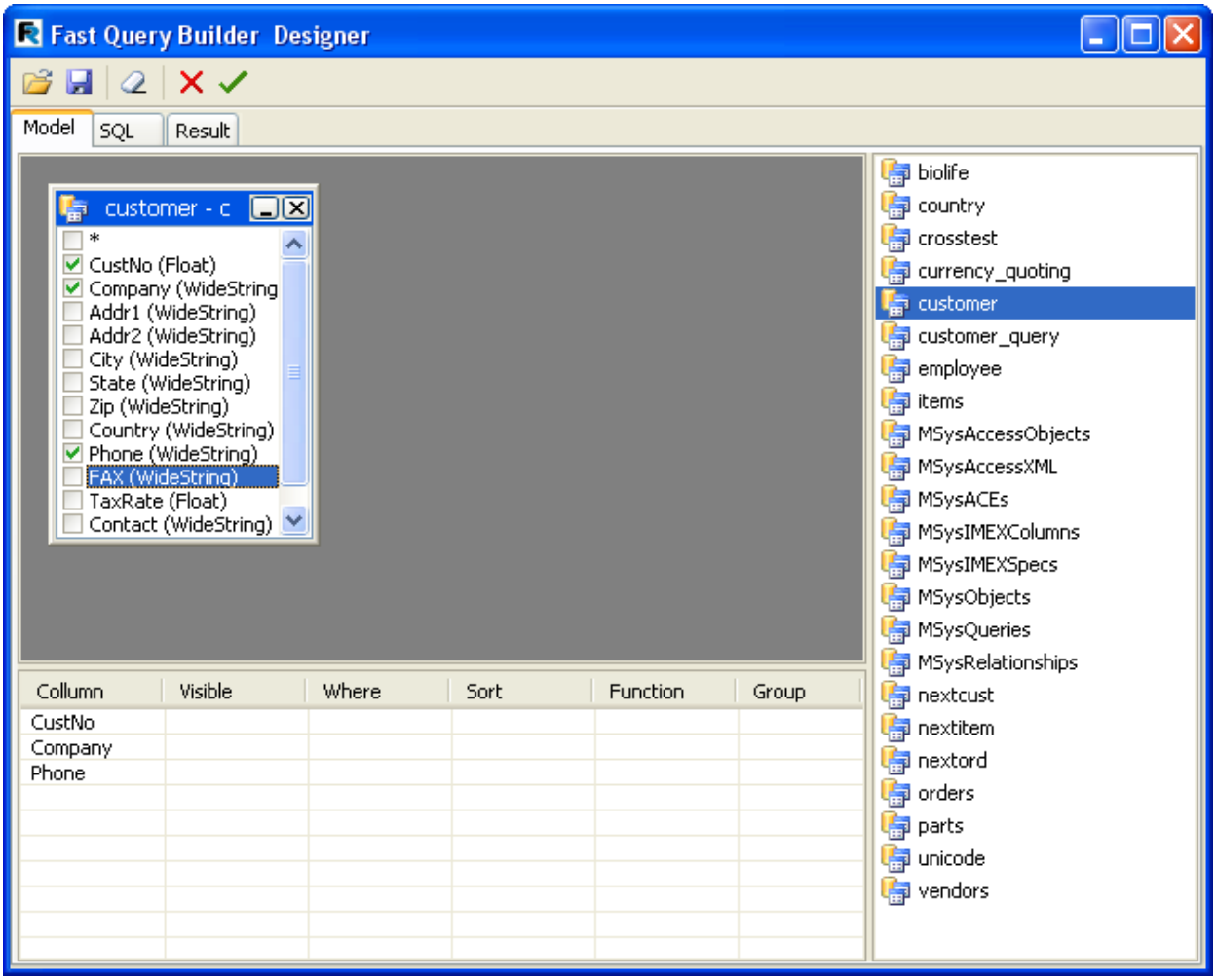
让我们使用查询构建器创建一个简单的报表。

单击设计器工具栏上的“新建报表”/以创建具有“报表标题”，“第一级数据”和“页脚”条段的报表页面。

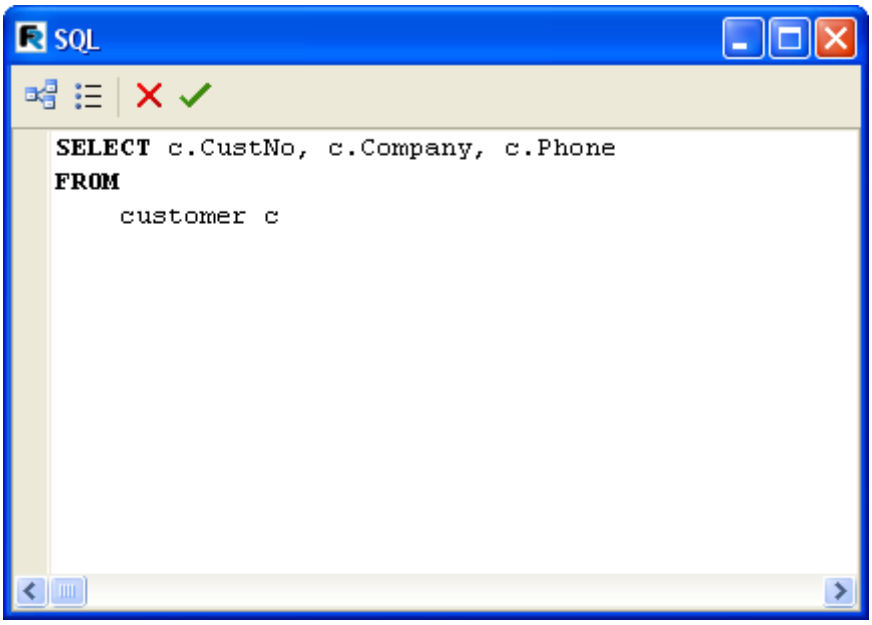
在“数据”选项卡上放置“ADO查询”组件。双击组件以打开查询编辑器。



单击查询编辑器中的/按钮以打开查询构建器窗口。在表格列表中选择Customer表（3）并将其拖到工作区（或者双击表格）。选择CustNo，Company和Phone字段：



这就是查询构建所需的全部内容。查询文本显示在SQL选项卡上，结果选项卡显示查询返回的数据。单击/关闭构建器并返回到查询编辑器，现在显示查询文本：



请注意！如果在查询编辑器中更改了查询文本，则表和联接的查询图将丢失。请勿手动更改查询文本，始终打开查询构建器并直观地修改图表。

单击查询编辑器中的/将返回到报表设计器。剩下要做的就是将“MasterData”频段连接到数据源，并将所需字段放在“MasterData”频段上。

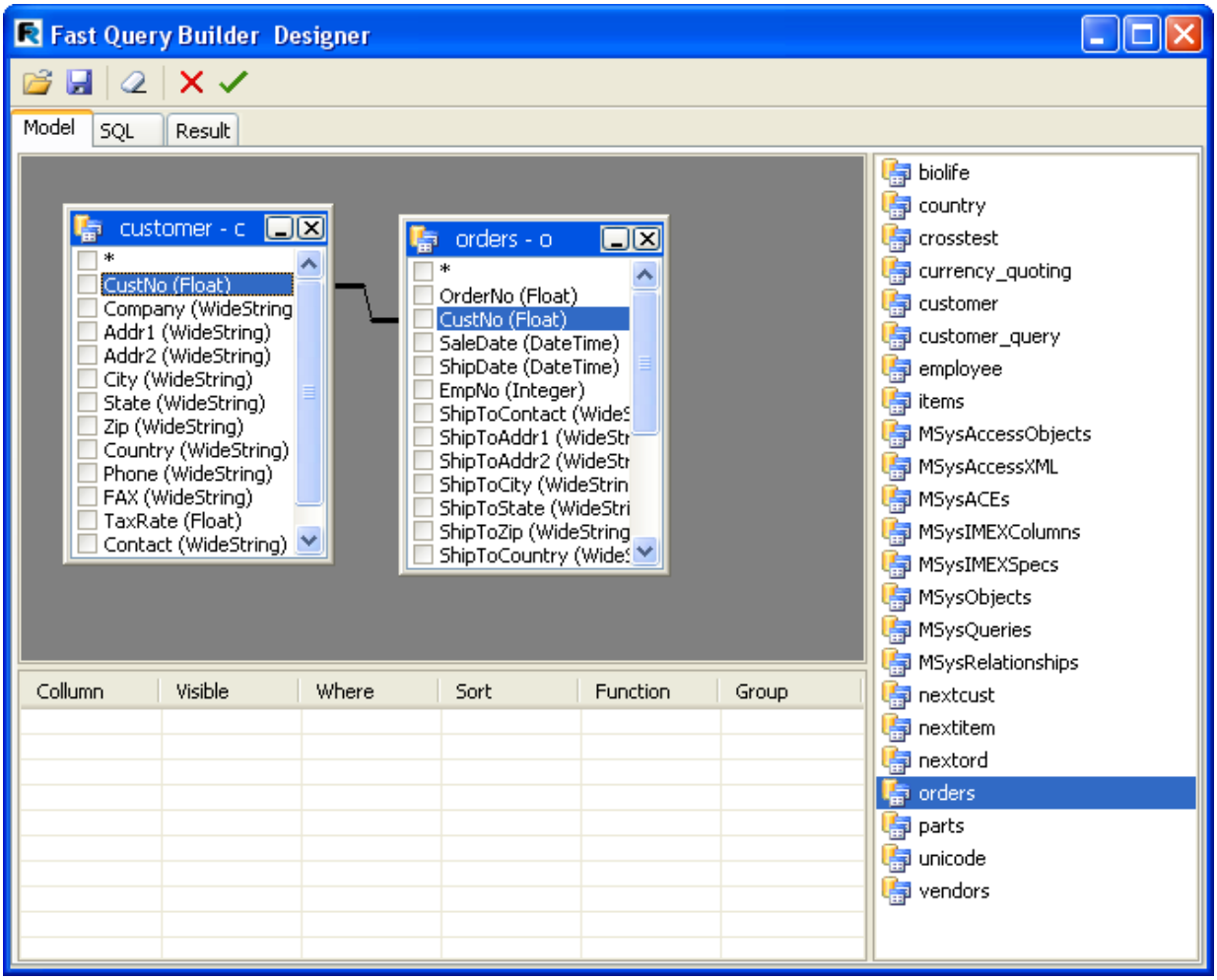
16.5.2 复杂的查询构建

在上一个例子中，我们基于一个表构建了一个报表。现在让我们看看数据来自两个表的查询构建。

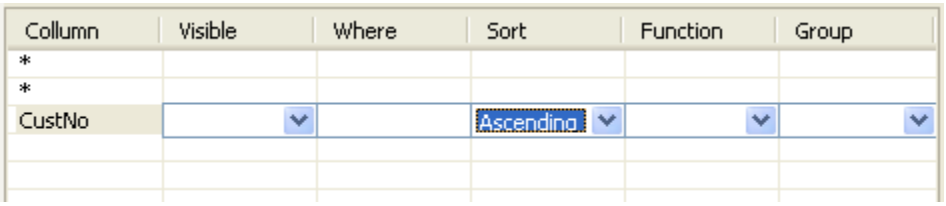
之前我们查看过与小组合作的报表。让我们使用查询构建器为此报表构建查询。我们需要在SQL中编写一个查询，它将从两个表中返回数据，并将数据分组到特定条件。在我们的示例中，条件将是两个表中的CustNo字段。

与前面的示例一样，创建一个新报表并在页面上放置一个“TADOQuery”组件。打开查询编辑器，然后打开查询构建器。

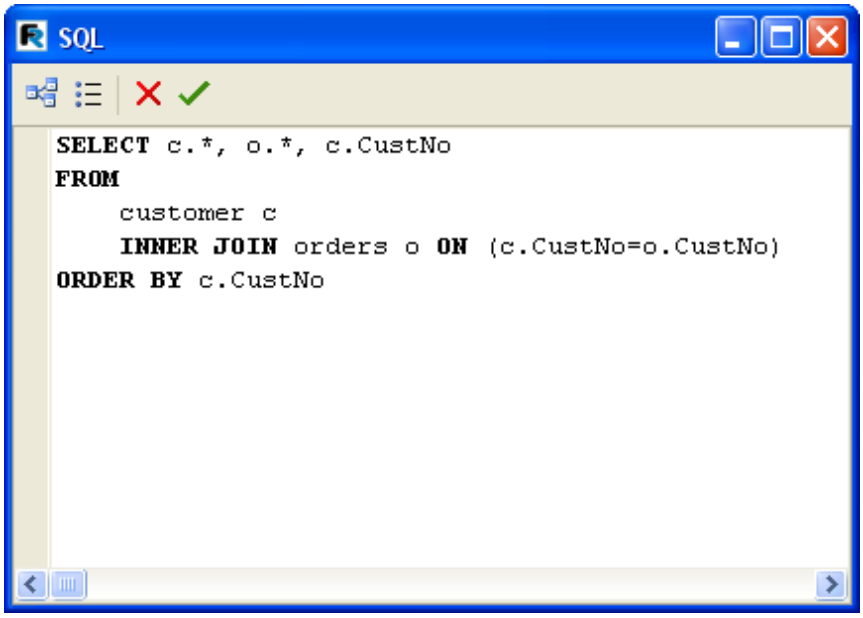
将两个表拖到工作区Customers和Orders。两个表都有一个CustNo字段，我们将使用它来加入它们。将CustNo字段从一个表拖到另一个表以在两个表之间创建连接：



现在需要设置要显示的字段和排序字段。检查两个表中的“*”字段，然后检查Customer表中的CustNo字段。选定的字段显示在字段参数列表中。选择CustNo字段的排序顺序：



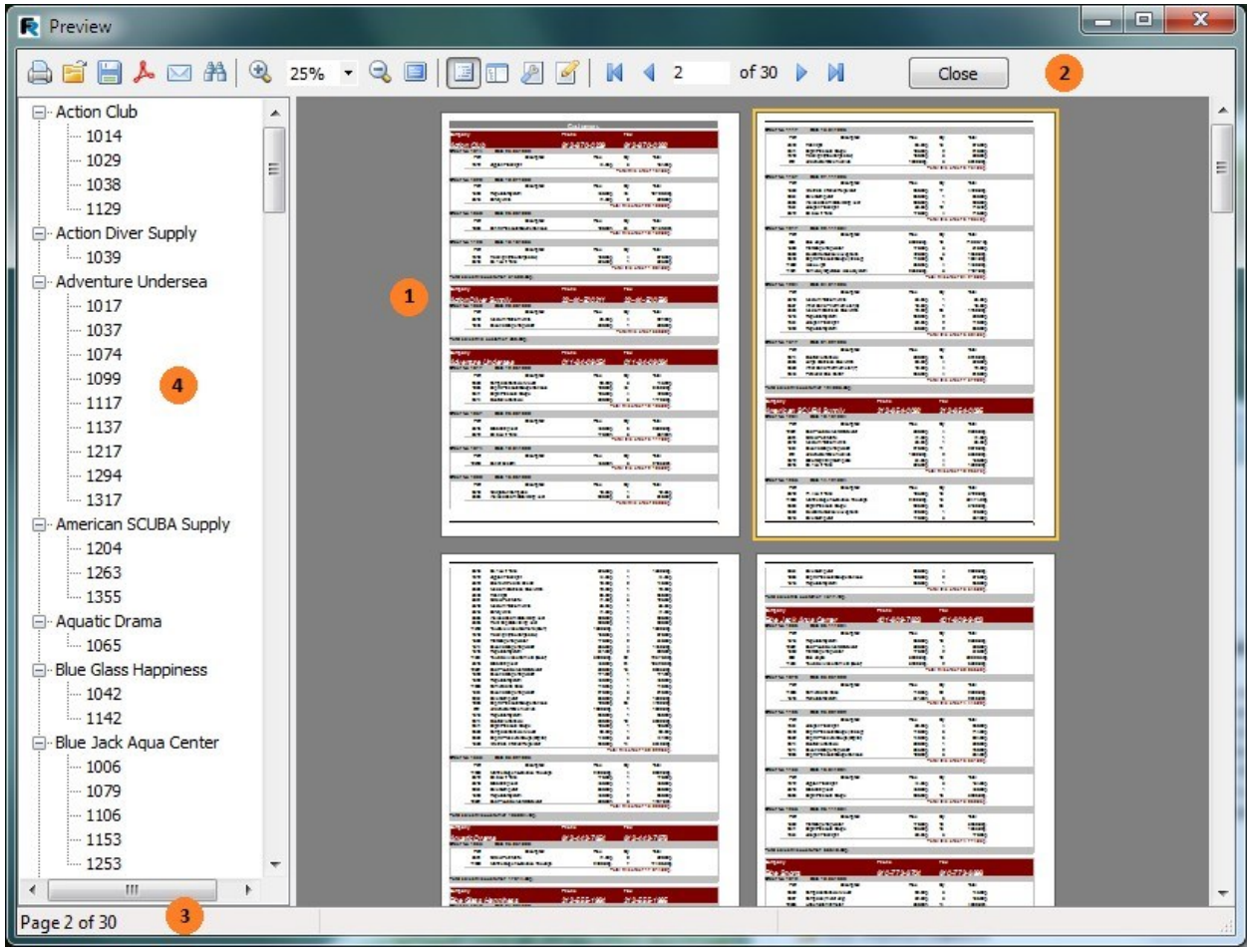
这就是完成查询所需的全部内容。 SQL代码如下所示：


SQL
SELECT c.*, o.*, c.CustNo
FROM
customer c
INNER JOIN orders o ON (c.CustNo=o.CustNo)
ORDER BY c.CustNo

章节 17

预览，打印，导出

可以显示和打印构建的报表，或将其导出为支持的格式之一。一切都可以在预览窗口中完成。

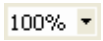
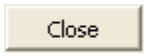


键：

1 – 完成报表页面
2 – 工具栏
3 – 状态栏
4 – 大纲树的轮廓空间（如上所示）或缩略图工具栏包含以下按钮：



图标	名称	描述
	打印报表	打印报表 - 热键：Ctrl + P.
	打开报表	打开包含已完成报表的文件（*.fp3）
	保存报表	将报表保存到文件（*.fp3）或将报表导出为一个

		支持的格式
	导出为PDF	如果安装了相应的导出过滤器，则导出报表到Adobe Acrobat文件（*.pdf）可见
	通过电子邮件发送	如果安装了相应的导出过滤器，则导出将报表为其中一种受支持的格式，并通过电子邮件将其发送为可见的机箱
	文字搜索	报表中的文本搜索 - 热键：Ctrl + F.
	放大	放大预览
	规模	选择任意比例缩放
	缩小	放大预览
	全屏	以全屏显示报表 双击报表返回正常大小
	大纲	显示或隐藏报表大纲
	缩略图	显示或隐藏缩略图视图
	页面属性	打开页面设置对话框
	编辑页面	编辑当前页面
	开始	跳转到第一个报表页面
	上一页	跳转到上一个报表页面
	页码	跳转到报表页码类型编号，然后按Enter键
	下一页	跳转到下一个报表页面
	结束	跳转到上一个报表页面
	关闭窗口	关闭预览

17.1 控制键

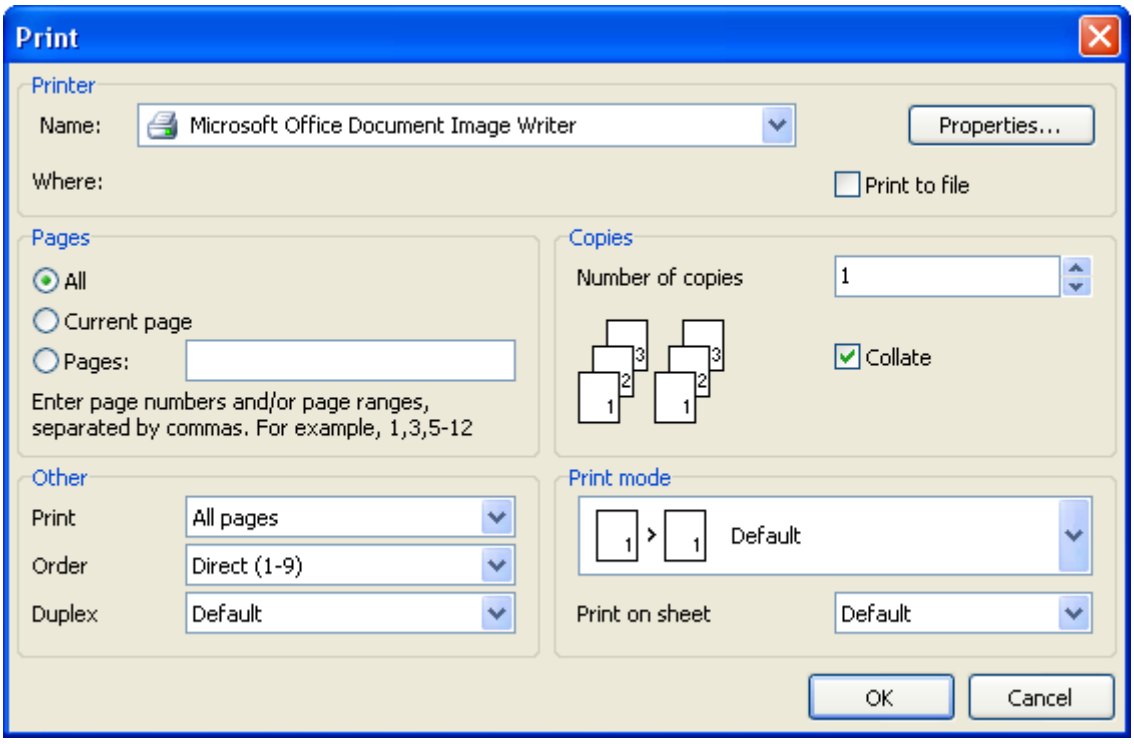
按键	描述
按Ctrl + S	将报表保存到*.fp3文件
按Ctrl + P	打印报表
按Ctrl + F	文字搜索
F3	再次搜索
箭头	流畅的报表滚动
PageUp, PageDown	上/下报表滚动
Ctrl + PageUp, PageDown	下一页/上一页报表滚动
家	报表开始
结束	报表结束

17.2 鼠标控制

行动	描述
左键	单击所选对象（在交互式报表中） 报表在“手动”模式下滚动（按住按钮移动鼠标）在“放大镜”模式下放大
右键	上下文菜单 缩小“放大镜”模式
双击	在全屏模式下恢复正常大小
鼠标滚轮	报表滚动

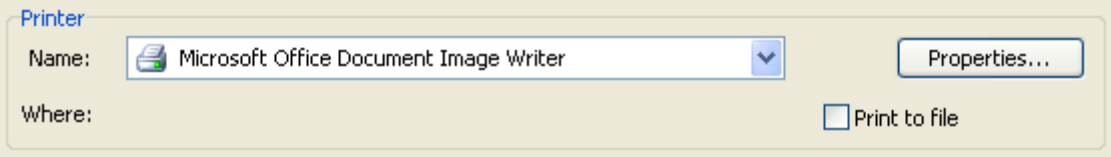
17.3 报表打印

要打印报表，请单击/按钮（或Ctrl + P热键）。打开标准打印对话框。

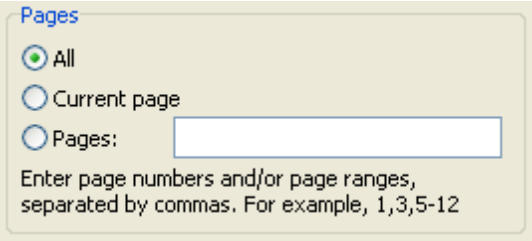


我们来看看这个对话中可用的选项。

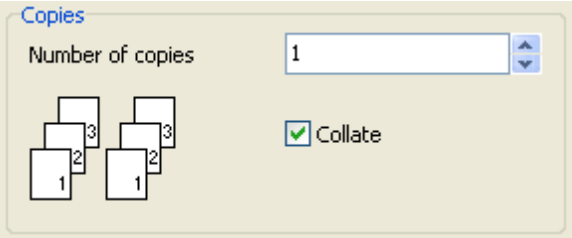
“打印机“面板：选择要在其上打印报表的打印机；设置打印机属性，例如打印质量；选择打印到文件。



“页面“面板：选择要打印的页面（全部，当前或选定范围）。

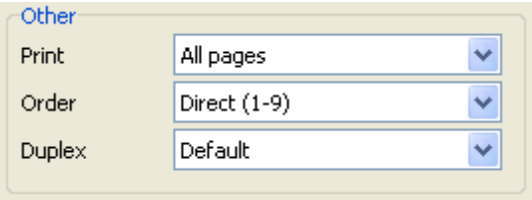


“复制“面板：设置要打印的份数。如果打印多个副本并选中自动分页，则第一个副本将完整打印，然后第二个副本完整打印，等等。如果未选中自动分页，则打印第一页的所有副本，然后打印第二页的所有副本，等等

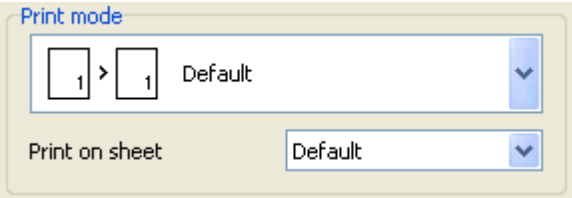


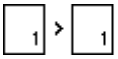
“其他“小组：

- 打印：选择要打印的页面（所有页面，偶数页面，奇数页面）
- 订购：以直接或相反的顺序打印页面（首页到最后一页，最后一页到第一页）
- 双工：默认情况下处理双工（使用报表设置）或选择双工选项之一：垂直，水平，单面

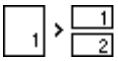


“打印模式“面板：选择打印模式。

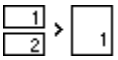



Default

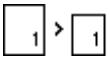
- 默认模式：在报表中定义的工作表上打印。每张纸上都打印一个预览页面


Split big pages

- 拆分大页面：如果在A4纸张上打印A3报表，此模式非常有用。一个预览页面打印在多个工作表上。选择此模式时，还必须指定纸张尺寸（“在纸张上打印”）。


Join small pages

- 加入小页面：如果在A3纸张上打印A4报表，此模式非常有用。在一张纸上打印两个或更多预览页面。选择此模式时，还必须指定纸张尺寸（“在纸张上打印”）。

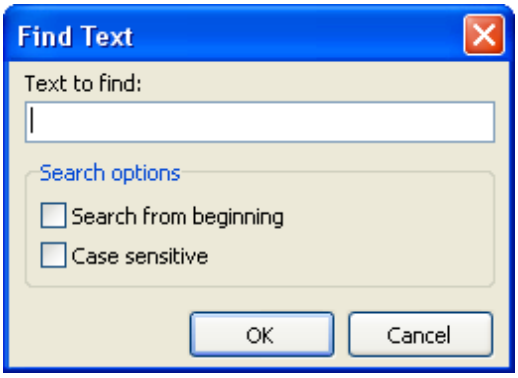

Scale

- 缩放模式：报表打印在指定大小的工作表上。所有报表输出都按比例缩放。一张预览页面打印在一张纸上。选择此模式时，还必须指定纸张尺寸（“在纸张上打印”）。

单击OK时，将开始报表打印。如果选中“打印到文件”，则会打开标准的“另存为...”对话框。报表将以*.prn扩展名保存到此文件中。该文件包含发送到打印机的信息的副本。

17.4 报表中的文本搜索

FastReport可以在预览报表的文本中搜索给定的短语。使用工具栏上的/按钮（或Ctrl + F热键）进行搜索。搜索对话框打开：



输入要搜索的短语，并根据需要选择选项：

- 从头开始搜索：从头开始搜索报表；否则搜索是从当前页面开始的
- 区分大小写：搜索时匹配搜索短语（下部或上部）的大小写

点击OK后，搜索开始，第一次出现（如果有）突出显示：



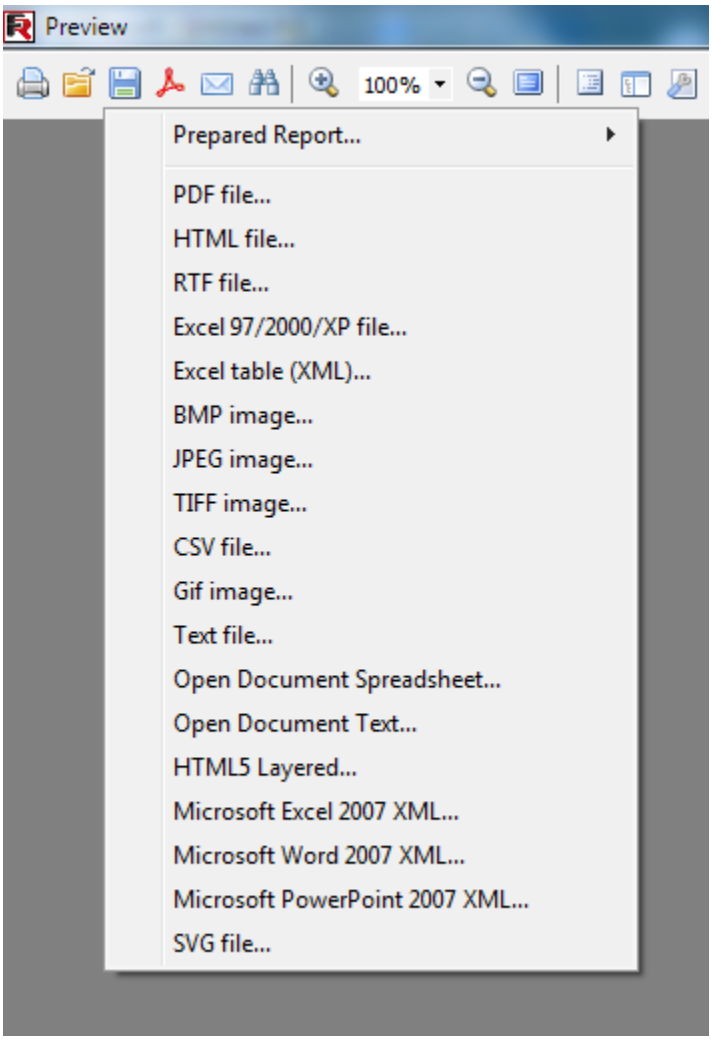
要继续搜索，请单击F3。下一个出现（如果有）将突出显示

17.5 保存报表

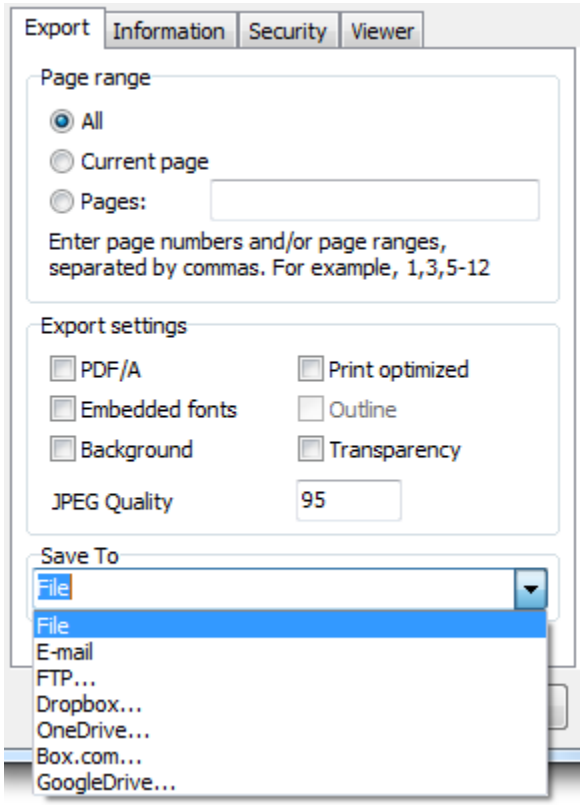
FastReport使用所谓的传输将准备好的报表或导出的文件保存到以下位置之一：

- 文件；
- 电子邮件；
- FTP；
- Dropbox的；
- 一个驱动器；
- Google云端硬盘；
- Box.com。

要保存准备好的报表（.fp3文件），请在预览窗口中按“保存”按钮，然后选择“准备报表”菜单项：



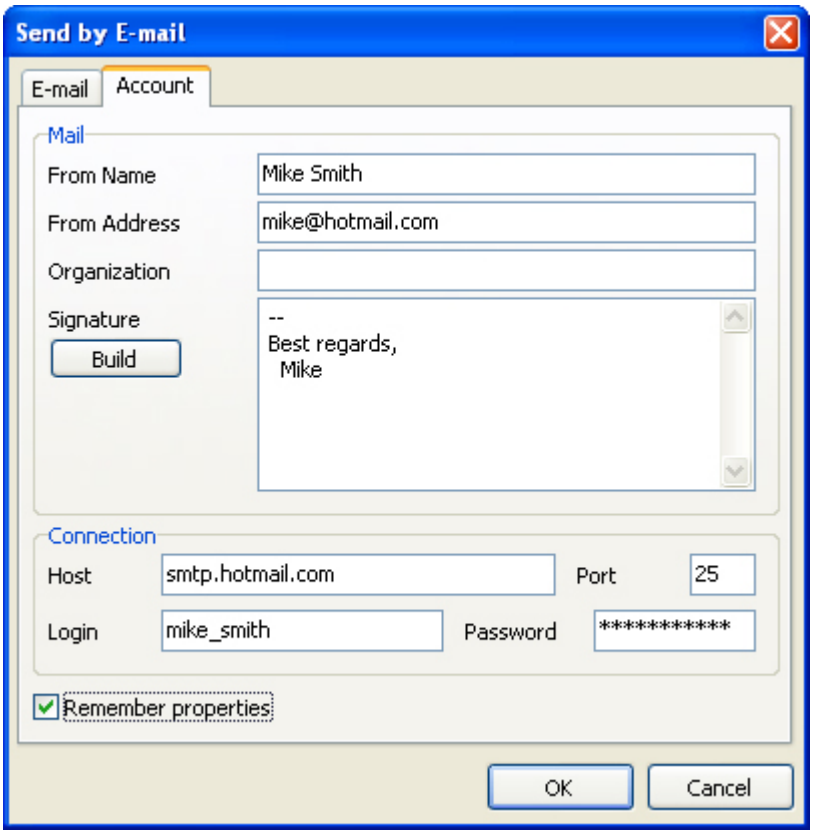
要导出报表并保存结果，请在预览窗口中按“保存”按钮，然后选择所需的导出。在导出对话窗口中，选择“保存到”选项：



17.5.1 通过电子邮件发送报表

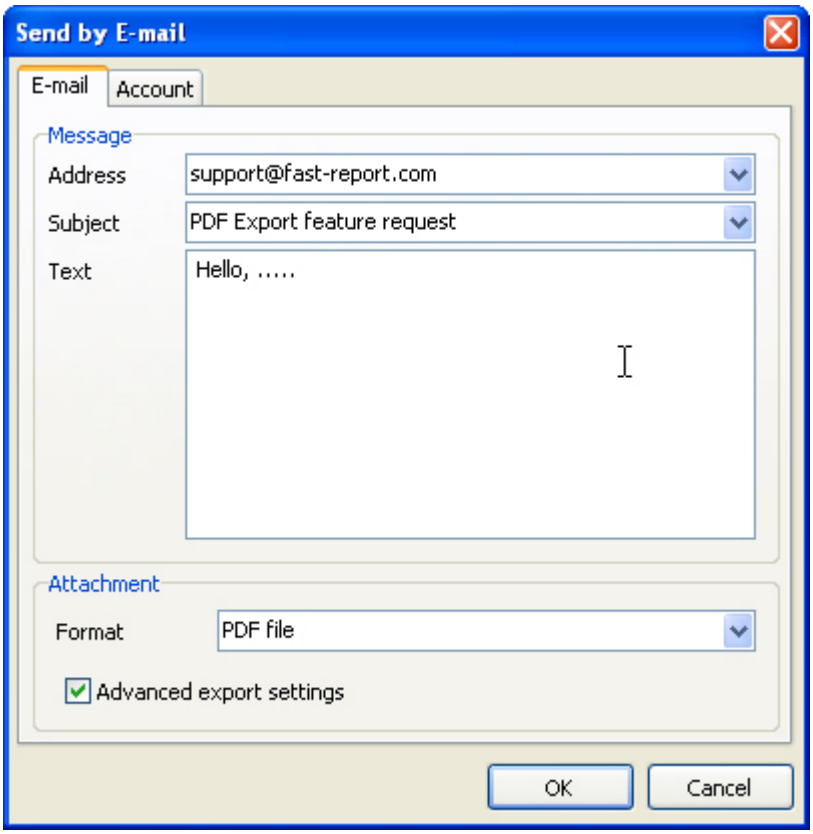
FastReport可以通过电子邮件以任何所需格式发送完成的报表，而无需单独的邮件应用程序。

通过电子邮件导出时，将打开电子邮件设置对话框。在导出开始之前，应在“帐户”标签上设置发件人的电子邮件帐户详细信息：



- 来自姓名： – 发件人姓名
- 来自地址：发件人的电子邮件地址
- 组织：发件人的组织
- 签名：电子邮件签名；填写前面的字段后，可以通过单击“构建”按钮生成它
- 主机：SMTP服务器服务
- 端口：SMTP服务器端口
- 登录：如果指定的SMTP服务器需要，则访问SMTP服务器上的授权名称
- 密码：授权密码，如果指定的SMTP服务器需要
- 记住属性：记住所有字段以供进一步使用

填写“帐户”选项卡上的必填字段后，必须填写“电子邮件”选项卡上的邮件字段：

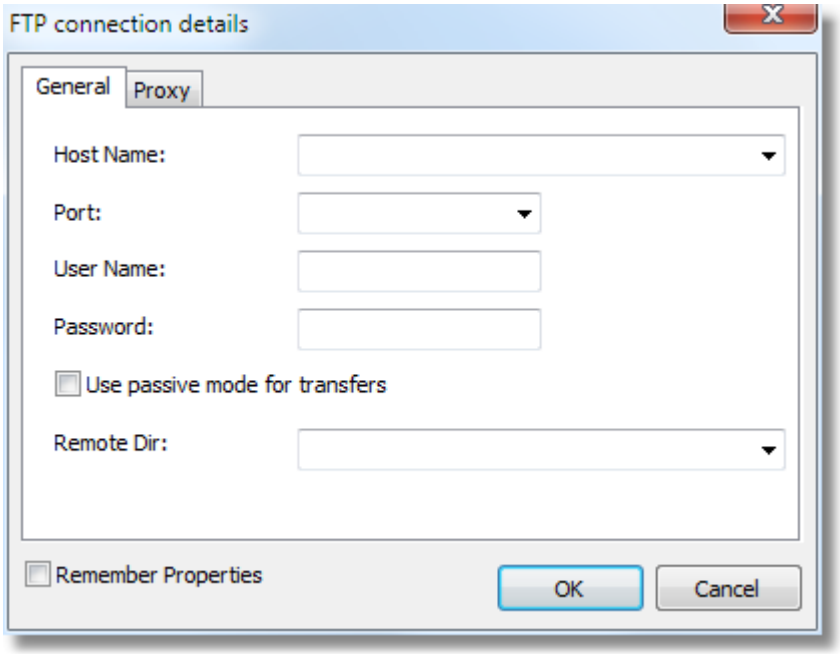


- 地址：收件人的电子邮件地址；可以从下拉列表中选择以前使用的地址
- 主题：信息主题；可以从下拉列表中选择以前使用的主题
- 文字：信息文字
- 格式：附加到电子邮件的导出报表的格式；选择一种可用的导出格式； FastReport (FR3) 格式也可供选择
- 高级导出设置：启用后，“确定”按钮将打开相应的导出格式设置对话框；禁用时使用默认导出设置

通过电子邮件功能导出：仅支持SMTP服务器上的纯文本身份验证。如果不需要验证，则无需输入“登录”和“密码”。

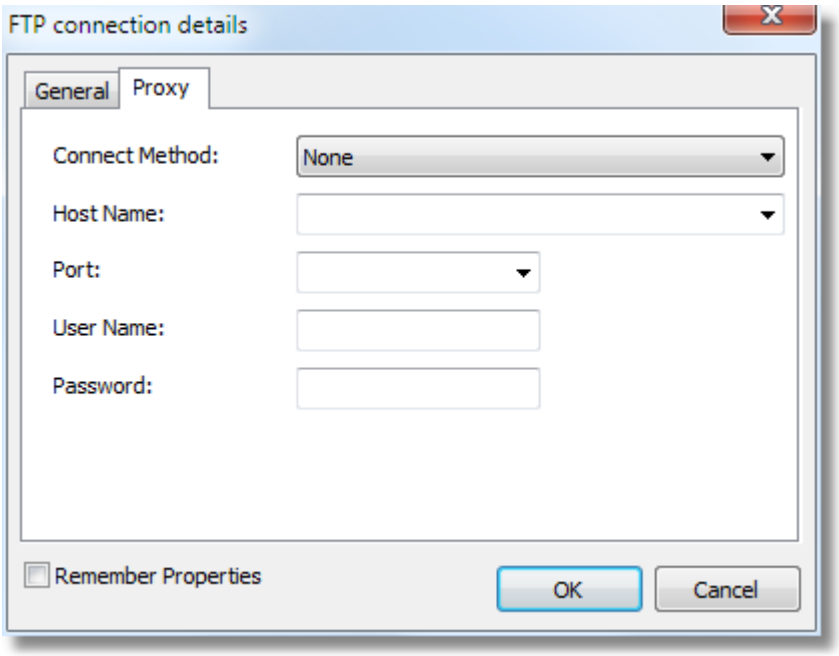
17.5.2 导出到FTP

将报表保存到FTP时，将显示以下对话窗口：



“常规”选项卡包含以下字段：
- 主机名：输入FTP服务器的URL地址。
- 端口：进入端口。
- 用户名和密码：输入您的用户名和密码。
- 使用被动模式：选择FTP连接模式。
- 远程目录：选择服务器上的远程目录（可选）。

如果使用代理服务器，则代理服务器的URL地址，端口，用户名和密码将在“代理”选项卡上设置：



完成所有设置后，单击“确定”按钮将文件保存到FTP服务器。

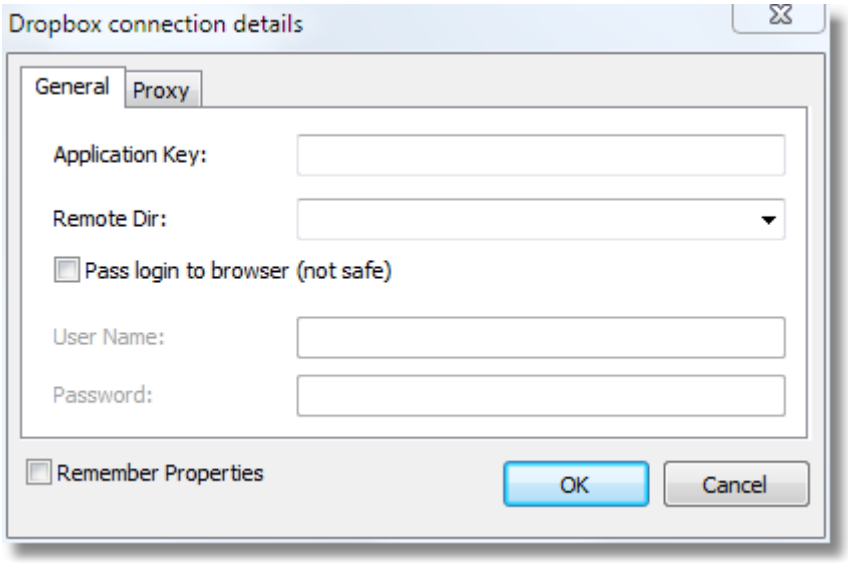
17.5.3 导出到Dropbox

在将报表保存到Dropbox之前，必须在Dropbox帐户中创建应用程序。通过登录Dropbox帐户并执行以下步骤来执行此操作：

- 点击“更多”按钮：它位于Dropbox主页的底部
- 在下拉列表中选择“开发人员”：您将被引导到开发人员的页面
- 转到“App Console”：它会将您引导至应用程序列表
- 单击“创建应用程序”按钮：Dropbox将检查您的电子邮件：单击“发送电子邮件”。在您的邮件系统收件箱中，您会看到一封带有“确认”按钮的电子邮件：单击按钮确认您的电子邮件地址。

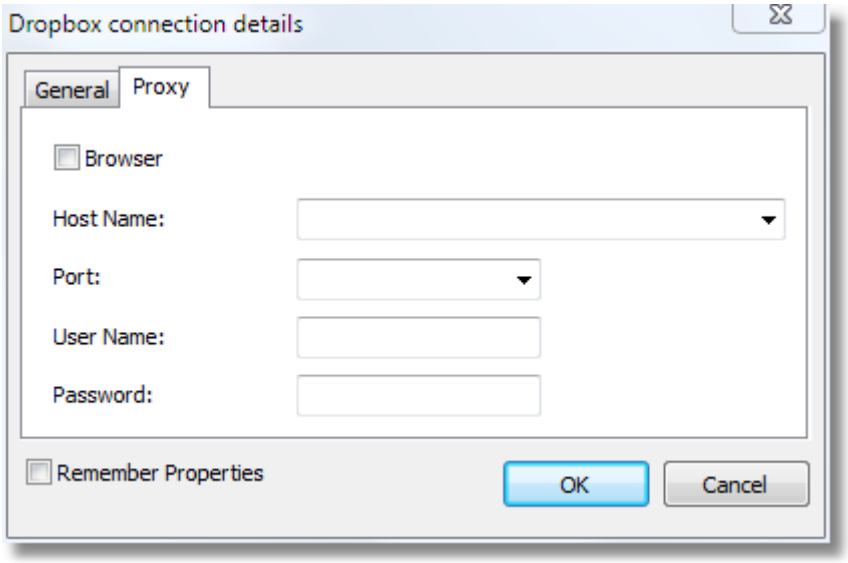
最后，您将被定向到“创建一个新的Dropbox平台应用程序”。
选择“Dropbox API应用”并回答“您的应用需要在Dropbox上存储哪种类型的数据？”的问题。
通过选择“文件和数据存储”。
回答问题“你的应用程序可以限制在自己的私人文件夹中吗？”通过选择两个提议的答案中的任何一个。
此页面还要求您输入应用程序的名称。
单击“创建应用程序”按钮后，系统将创建应用程序。
结果，打开了应用程序设置页面。在这里，您可以看到导出到Dropbox时所需的“应用程序密钥”和“应用程序密钥”。

导出到Dropbox时，将显示以下对话窗口：



输入上面获得的“应用程序密钥”。您也可以选择服务器上的远程目录。

如果使用代理服务器，则代理服务器的URL地址，端口，用户名和密码将在“代理”选项卡上设置：



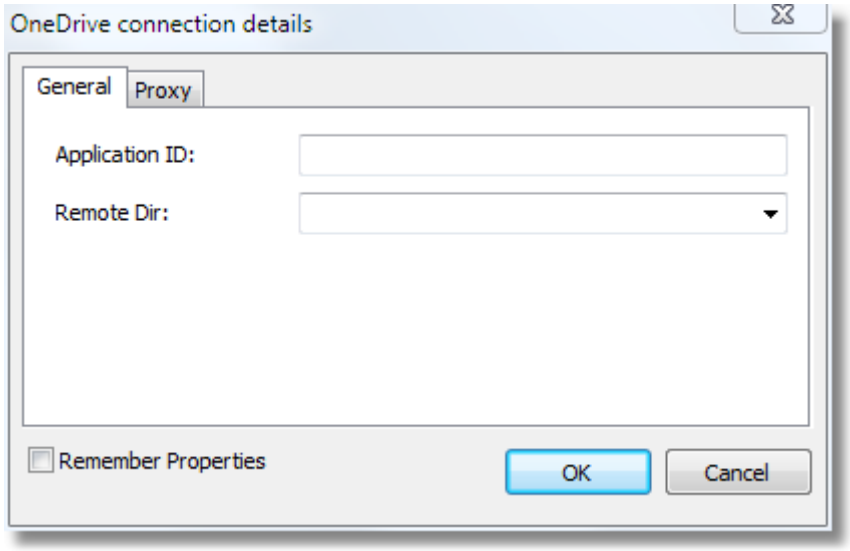
完成所有设置后，单击“确定”按钮。将打开浏览器窗口以登录Dropbox。然后该文件将保存到Dropbox。

17.5.4 导出到OneDrive

在将报表保存到OneDrive之前，必须在OneDrive帐户中创建应用程序。通过转到OneDrive主页并单击“开发人员”来完成此操作。
在开发中心单击“我的应用程序”，然后单击“创建应用程序”。
输入应用程序的名称并选择一种语言。阅读“使用条款”和“隐私声明”，然后单击“我接受”。
因此，下一页是应用程序的“设置”页面，其中显示“客户端ID”和“客户端密钥”。

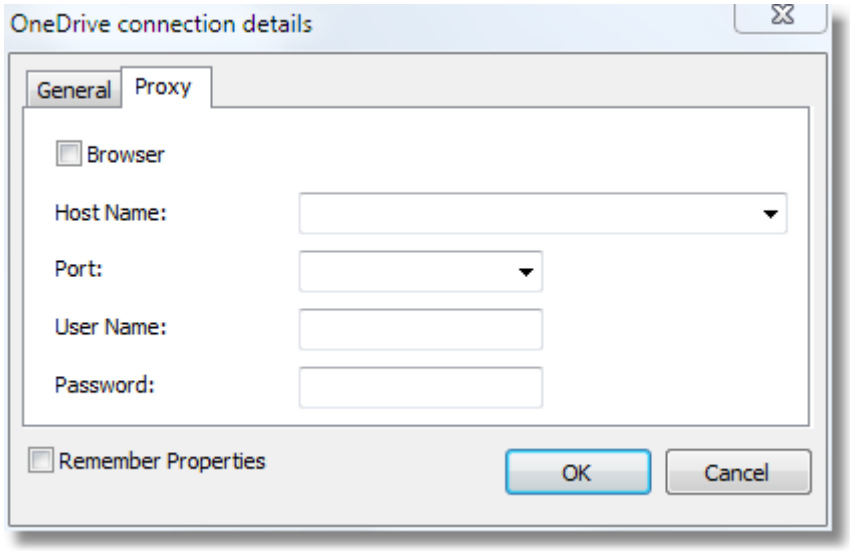
进入重定向域，然后单击“保存”。

导出到OneDrive时，将显示以下对话窗口：



输入上面获得的“应用程序ID”。您也可以选择服务器上的远程目录。

如果使用代理服务器，则代理服务器的URL地址，端口，用户名和密码将在“代理”选项卡上设置：



完成所有设置后，单击“确定”按钮。将打开浏览器窗口以登录OneDrive。然后该文件将保存到OneDrive。

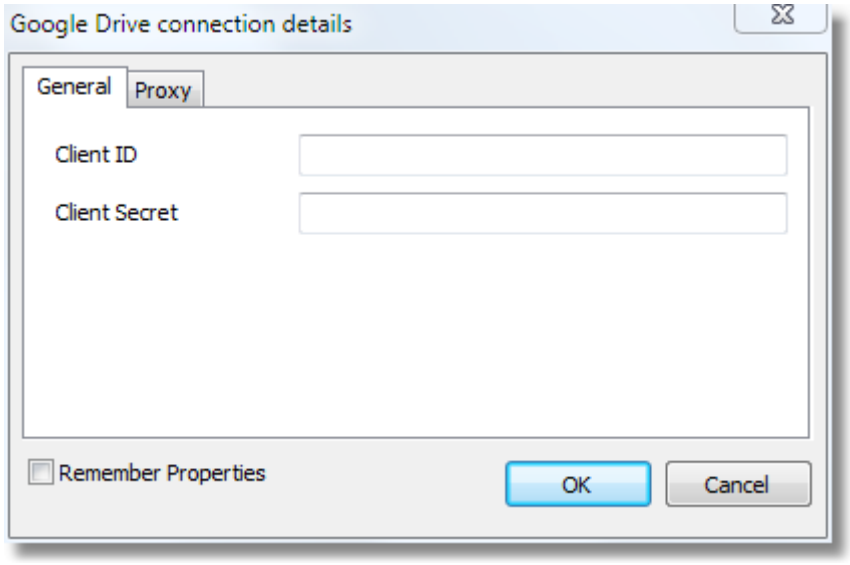
17.5.5 导出到Google云端硬盘

在将报表保存到Google云端硬盘之前，必须在您的Google云端硬盘帐户中创建应用。
通过访问https://code.google.com/apis/console/来执行此操作
- 此页面具有许可协议，并指向“项目设置”页面。

转到“服务”标签并激活“云端硬盘API”。
转到“API访问”标签，然后点击“创建OAuth 2.0客户端ID”。
在“品牌信息”部分中输入应用程序的名称，然后单击“下一步”。在“客户端ID设置”中，选择以下内容：
- 应用程序类型的“已安装的应用程序”
- 已安装的应用程序类型的“其他”。单击“创建客户端ID”。

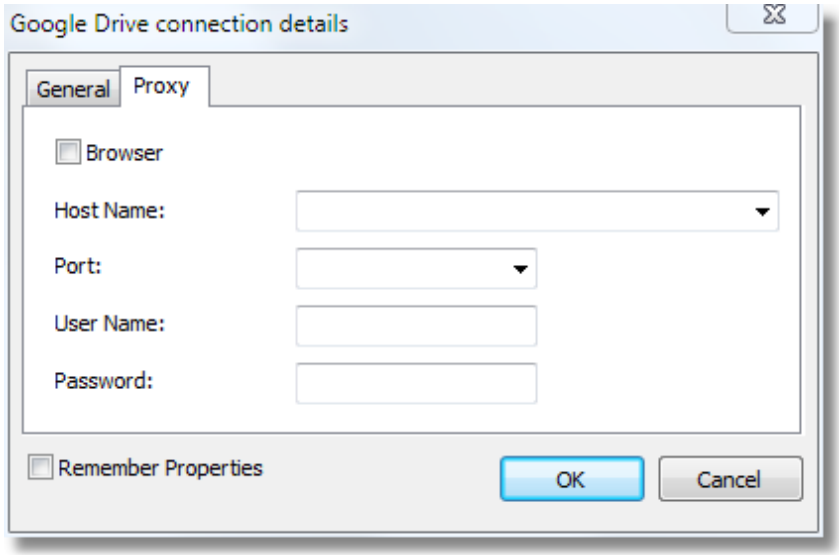
结果，下一页显示“客户端ID”和“客户端密钥”。

导出到Google云端硬盘时，将显示以下对话窗口：



输入上面获得的“客户端ID”和“客户端密钥”。

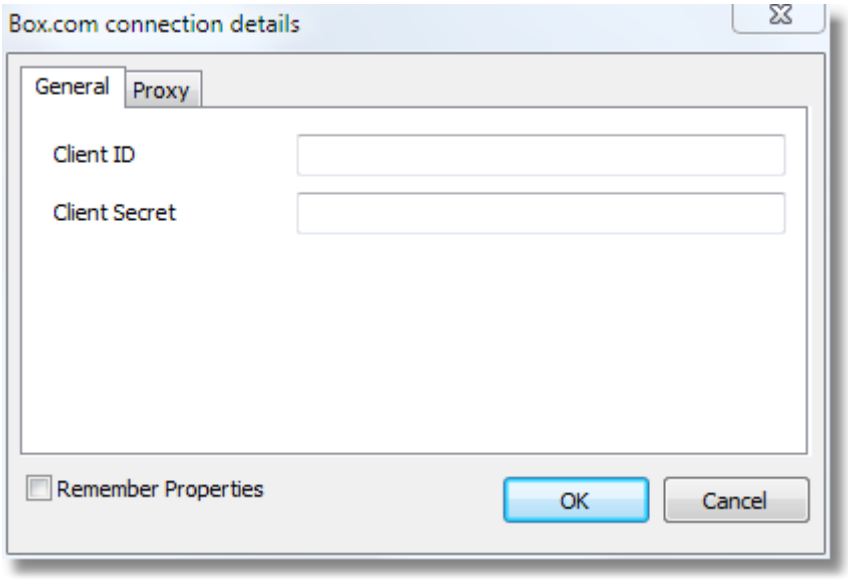
如果使用代理服务器，则代理服务器的URL地址，端口，用户名和密码将在“代理”选项卡上设置：



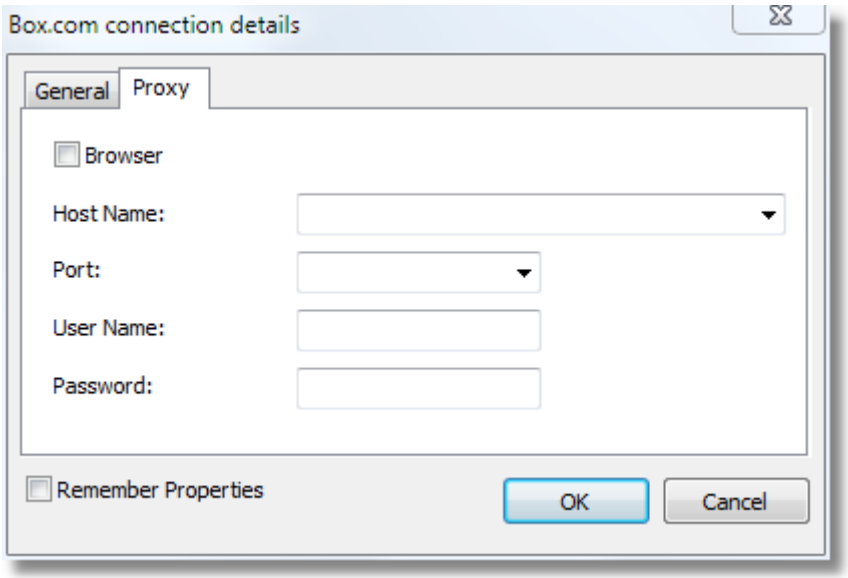
完成所有设置后，单击“确定”按钮将文件保存到Google云端硬盘。

17.5.6 导出到Box.com

导出到Box.com时，将显示以下对话窗口：



如果使用代理服务器，则代理服务器的URL地址，端口，用户名和密码将在“代理”选项卡上设置：

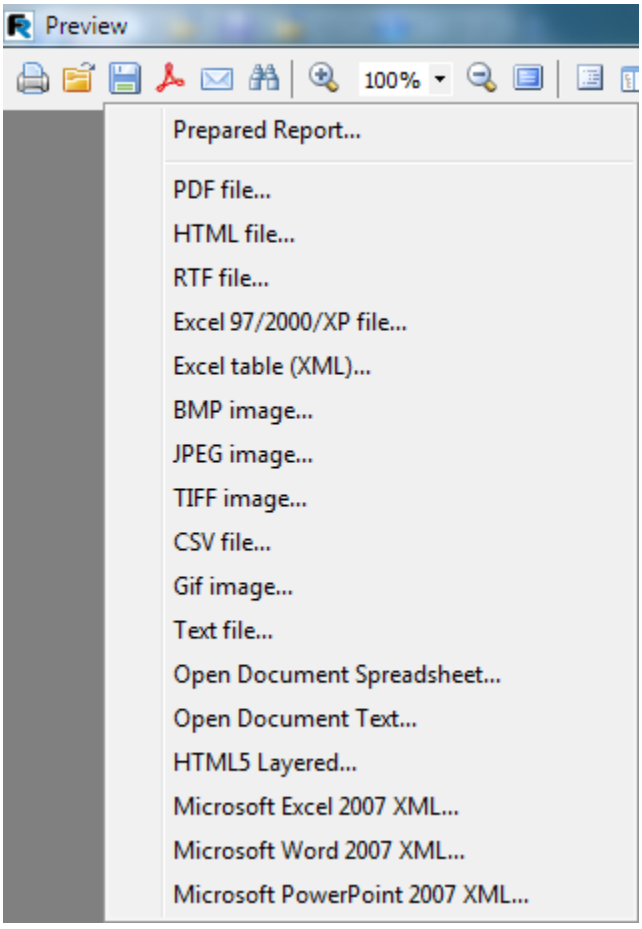


完成所有设置后，单击“确定”按钮将文件保存到Box.com。

17.6 报表导出

FastReport可以将预览报表导出为各种格式，以便进一步编辑，存档或通过电子邮件发送等。要启用导出，必须将相应的FR导出组件添加到Delphi表单中。

FastReport可以导出为以下格式：PDF，Open Document Spreadsheet，Open Document Text，Excel 97/2000 / XP，Excel XML，Excel 2007，RTF，Word 2007，PowerPoint 2007，HTML，text，CSV，BMP，Jpeg，Tiff和Gif。此外，报表可以通过电子邮件以任何这些列出的格式发送。



FastReport使用以下三种方法之一导出报表：

-'Layer'：每个报表对象都导出到一个单独的图层。导出的输出近似于原始预览。
-'Table'：将对象导出到输出文件是通过在内存中创建过渡网格然后输出此网格。导出的输出与

原始预览紧密匹配，基于遵循良好报表设计原则的假设（请参阅“报表设计注意事项”一章）。

- ’绘图’：从页面图像中捕获导出的对象。导出的输出是预览的直接副本。导出为图形格式时使用此方法。

17.6.1 导出为PDF格式

PDF（可移植文档格式）：由Adobe Systems创建的独立于平台的电子文档格式。免费的Adobe Reader包用于查看。这种格式很灵活，允许包含所需的字体，矢量和位图图像；它是一种分发和存储用于查看和/或打印的文档的方法。

Export方法是“逐层”。

导出为PDF格式时，会打开一个对话框，请求输出文件设置。



导出设置：

- 压缩：输出文件被压缩，文件大小减少但导出时间增加
- 嵌入字体：报表中使用的所有字体都包含在PDF输出文件中，允许在没有字体的计算机上准确呈现；输出文件大小显着增加
- 背景：分配给页面背景的图形图像导出为PDF文件；输出文件大小显着增加
- 打印优化：以高分辨率输出图形图像，实现精确打印；只有当文档包含图形并且将被打印时，才需要此选项；输出文件大小显着增加
- 大纲：仅当报表包含大纲时才启用选项；大纲导出为PDF文件
- 导出后打开：使用计算机上安装的默认PDF查看器（例如，Adobe Reader）在导出后立即打开导出的文件。

导出功能：RichText对象导出为图形。

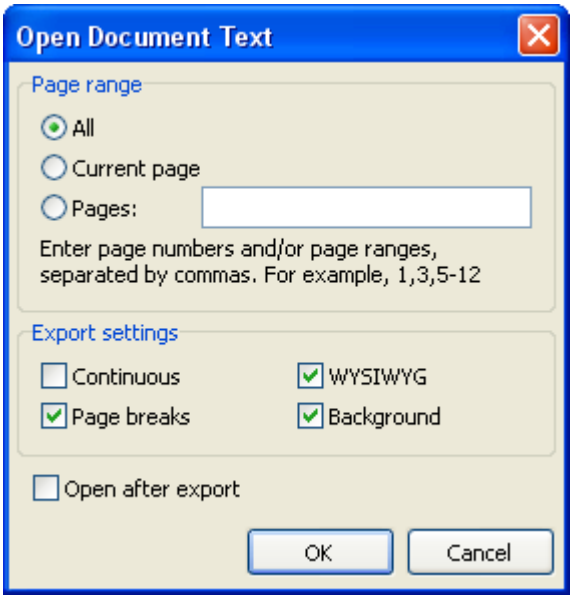
17.6.2 导出到打开文档

开放文档格式（ODF，OASIS Office应用程序的开放文档格式）由OASIS设计，基于OpenOffice中使用的XML格式。

FastReport支持导出到表（.ods）和文本（.odt）文件。可以在OpenOffice中打开这些文件。

Export方法是'table'。

导出为ODF格式时，会打开一个对话框，请求输出文件设置。



导出设置：

- 连续：导出为连续文档，没有分页符，没有页眉/页脚
- 分页符：在文档中启用分页符
- 所见即所得：预览报表的准确再现；允许禁用优化时，减少导出文件中的行数和列数
- 背景：分配给页面背景的图形图像被导出到ODF文件；输出文件大小显着增加
- 导出后打开：导出后立即打开导出的文件。

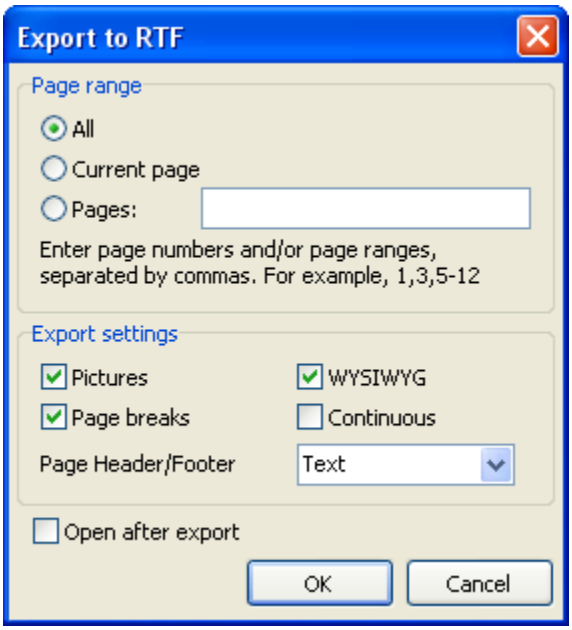
导出功能：RichText对象导出为简单文本，支持导出图形图像。

17.6.3 导出为RTF格式

RTF（RTF格式）由Microsoft开发，作为文本文档交换的标准。许多现代文本编辑器和操作系统都支持RTF文档。

Export方法是'table'。

导出为RTF格式时，会打开一个对话框，请求输出文件设置。



导出设置：

- 图片：可以将图形图像导出到文件
- 分页符：在文档中启用分页符
- 所见即所得 – 准确呈现预览报表；允许禁用优化时，减少导出文件中的行数和列数
- 连续：导出为连续文档，没有分页符，没有页眉/页脚
- 页眉/页脚：页眉/页脚导出模式；模式是：文本（h / f导出为普通文本），页眉/页脚（h / f导出）和无（h / f未导出）
- 导出后打开：导出后使用计算机上安装的默认RTF查看器立即打开导出的文件（例如，Microsoft WordPad）

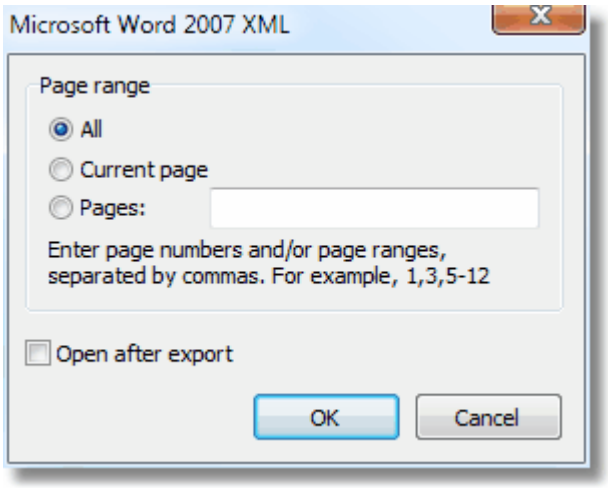
导出功能：RichText对象完全集成在RTF格式中；渲染和文件大小的准确性取决于报表的设计方式，请参阅“报表设计注意事项”一章。

17.6.4 导出到Word 2007

Word 2007是一个处理文本文档的应用程序。它包含在Microsoft Office 2007中。

导出方法是'table'。

导出为Word 2007格式时，将打开一个对话框，请求输出文件设置。



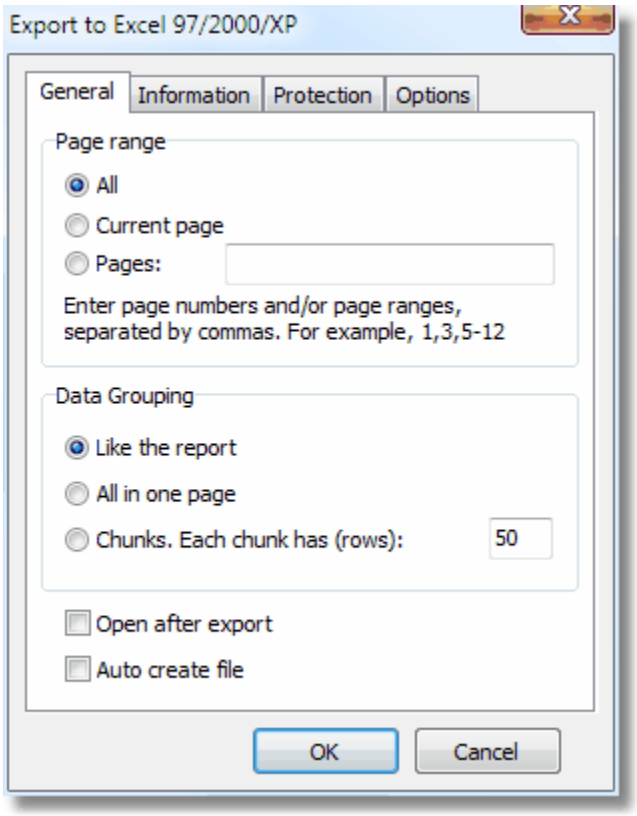
导出设置:

- 出口后打开 – 输出文件将在导出后立即打开。

17.6.5 导出到Excel 97/2000 / XP

高强 – 申请使用电子电子表格。它包含在Microsoft Office中。 Export方法是'table'。

导出为Excel格式时，会打开一个对话框，请求输出文件设置。



数据分组：

- 与报表一样 – 准备好的报表的每一页都在单独的Excel表格中导出；
- 全部在一页 – 生成连续文档，没有分页符和页眉/页脚；
- 块。每个块都有（行） – 每个块都在单独的Excel工作表上导出。导出设置：

- 所见即所得：完全符合报表的外观；允许禁用优化时，减少导出文件中的行数和列数
- 图片 – 包括图形图像导出到输出表；
- 网格线 – 打开/关闭Excel网格线；
- 调整页面大小 – 调整单元格大小以适合内容；
- 删除空行 – 从输出表中删除空行；
- 导出公式 – 如果单元格文本以“=”符号开头，则将其导出为Excel公式。
- 导出后打开Excel：导出后立即打开导出的文件。

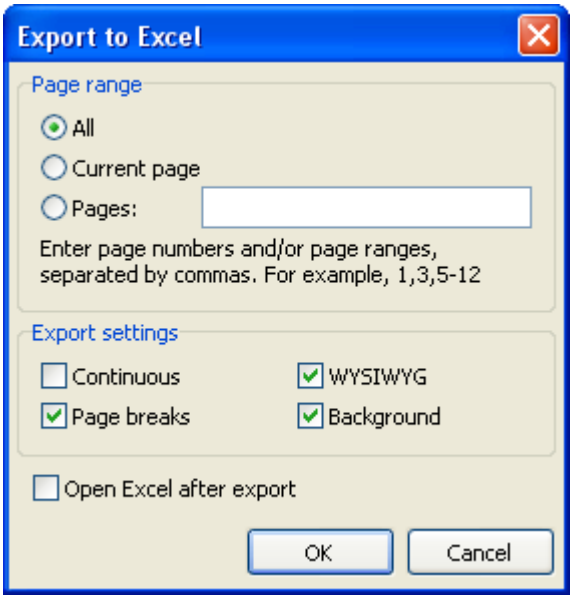
导出功能：RichText对象作为简单文本传输，支持图形图像传输。

17.6.6 导出为Excel XML格式

XML（可扩展标记语言）是一种可扩展的标记语言。 XML用于结构化数据存储，也用于不同程序之间的信息交换。 FastReport使用XML格式将数据传输到Excel ver。 2003年及以后。

导出方法是'table'。

在导出为XML格式时，将显示输出文件参数设置的对话框。



出口参数：

- 连续 - 生成连续文档，不包含分页符和页眉/页脚
- 分页符 – 启用文档中的分页符
- 所见即所得 – 准确呈现预览报表；允许禁用优化时，减少导出文件中的行数和列数
- 背景：导出到电子表格的报表页面的背景颜色
- 导出后打开Excel：导出后立即打开导出的文件。

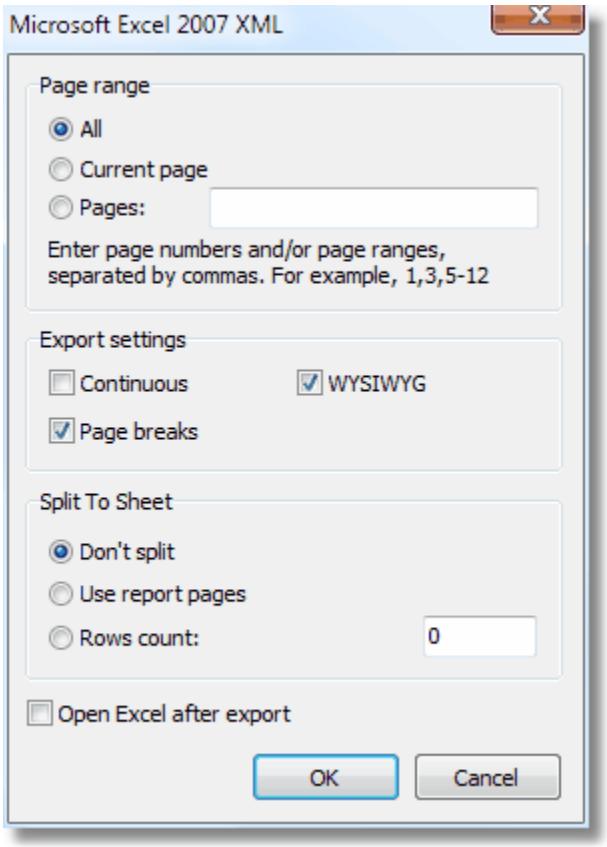
导出功能：RichText对象导出为简单文本；不支持图形图像。

17.6.7 导出到Excel 2007

Excel 2007是一个处理电子工作表的应用程序。它包含在Microsoft Office 2007中。

导出方法是'table'。

导出为Excel格式时，会打开一个对话框，请求输出文件设置。



拆分为表格：

- 不要拆分 – 所有报表页面都导出到单个Excel工作表中；
- 使用报表页面 – 准备好的报表的每一页都在单独的Excel工作表中导出；
- 行数 – 每组行都在单独的Excel工作表中导出。出口参数：

- 连续 – 生成连续文档，没有分页符和页眉/页脚；
- 分页符 – 包括结果文件中的分页符；
- 所见即所得 – 准确呈现预览报表；当允许禁用优化时，减少导出文件中的行数和列数；
- 导出后打开Excel – 导出后立即打开导出的文件。

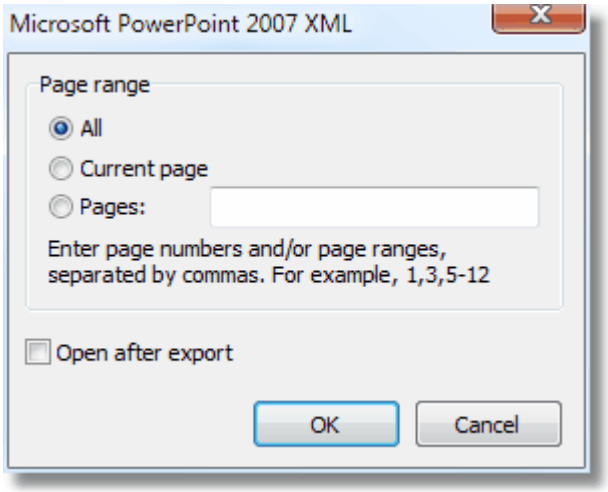
导出功能：RichText对象作为简单文本传输，支持图形图像。

17.6.8 导出到PowerPoint 2007

PowerPoint 2007是一个处理电子演示文稿的应用程序。它包含在Microsoft Office 2007中。

导出方法是分层的。

导出为PowerPoint格式时，会打开一个对话框，请求输出文件设置。



导出设置：

– 出口后打开 – 导出文件将在导出后立即打开。

17.6.9 导出为CSV格式

CSV文件包含纯文本格式的表格数据（数字和文本）。在列值之间插入指定的分隔符，并且每行在新行上开始。这种格式可以由各种表/图编辑器打开。

Export方法是'table'。

导出为CSV格式时，将打开一个对话框，请求输出文件设置。



导出设置：

- OEM代码页：OEM编码导出的文件

- 分隔符：列之间使用的分隔符
- 导出后打开：导出后使用计算机上安装的默认CSV查看器立即打开导出的文件。

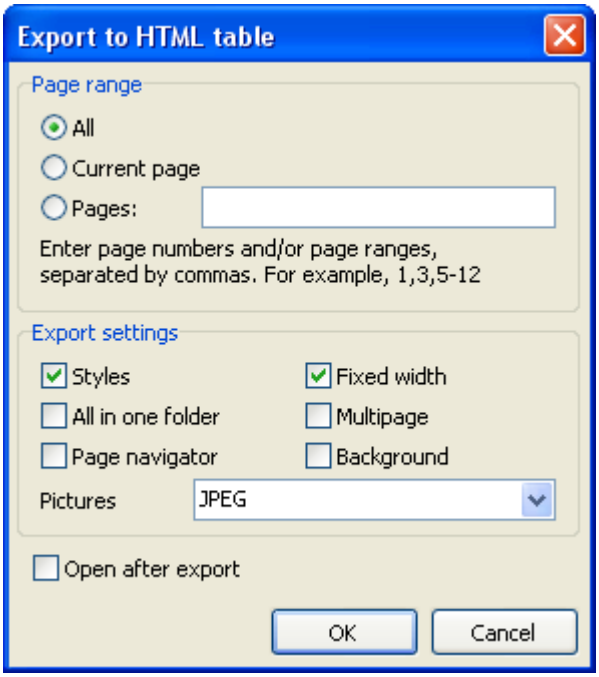
导出功能：导出文件中不包含布局信息（即报表设计）；不支持图形图像。

17.6.10 导出为HTML格式

HTML（超文本标记语言）被认为是因特网上文档的标准语言。它用于创建相对简单但设计良好的文档。HTML支持超文本链接以及简单的文档布局。

Export方法是'table'。

导出为HTML格式时，会打开一个对话框，请求输出文件设置。



导出设置：

- 样式：导出的文本对象的设计样式；当禁用导出速度增加时，但会减少电子表格外观
- 所有在一个文件夹中：所有其他文件都保存在与主文件相同的文件夹中
- 页面导航器：用于在创建的页面之间快速跳转的特殊导航器
- 固定宽度：在更改显示窗口大小时阻止自动表/图表宽度调整
- 多页：每个页面作为单独的文件导出
- 背景：分配给页面背景的图形图像被导出为HTML文件
- 图片：图形图像导出为HTML文件

- 导出后打开：导出后使用计算机上安装的默认HTML查看器立即打开导出的文件。

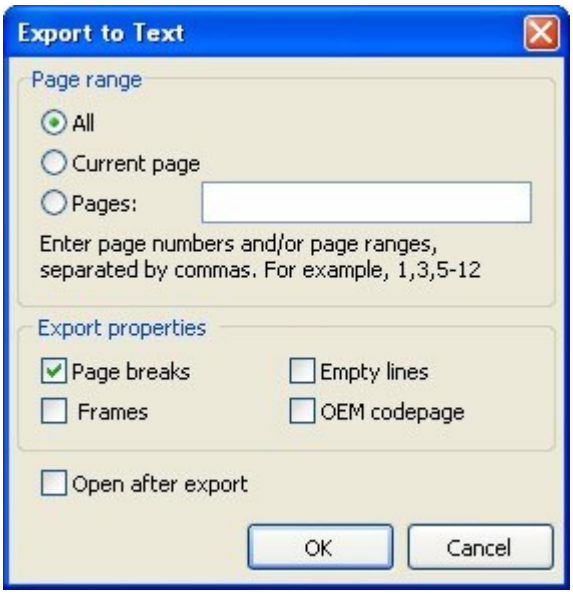
导出功能：导出可能包含多个文件；每个图形图像导出到自己的文件； RichText对象导出为简单文本；渲染精度和文件大小取决于报表的设计方式，请参阅“报表设计注意事项”一章。

17.6.11 导出为文本格式

普通纯文本文件没有任何图形。 Export

方法是’table’。

导出为文本格式时，会打开一个对话框，请求输出文件设置



导出设置：

- 分页符：将分页符导出到结果文件
- 空行：在文档中启用分页符
- 框架：启用文本对象框架导出
- OEM代码页：OEM编码导出的文件
- 导出后打开：导出后使用计算机上安装的默认文本查看器立即打开导出的文件。

导出功能：导出文件中不包含布局信息（即报表设计）；不支持图形图像：页面宽度根据报表页面上的文本对象类型自动设置。

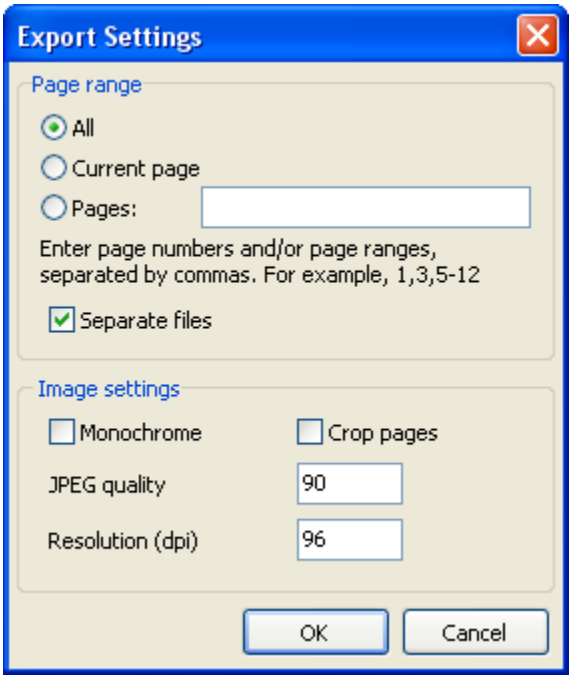
17.6.12 导出为Jpeg，BMP，Gif，Tiff图形格式

FastReport导出为图形格式。

- JPEG（联合图像专家组）：基于记录像素之间差异的算法的压缩格式。它的特点是高压缩，但牺牲图形精度。
- BMP（Windows设备独立位图）：用于存储Windows中使用的位图图像。 Windows控制下的计算机的标准文件格式。
- GIF（图形交换格式）：开发了硬件无关格式，用于通过网络传输位图图像。非常适合压缩同质内容（徽标，铭文，方案）。
- TIFF，TIF（目标图像文件格式）：硬件独立格式。今天，它是印刷和传真传输中最广泛和最可靠的之一。

Export方法是'enveloping'。

以上述图形格式导出时，会打开一个对话框，请求输出文件设置。



导出设置：

- 单独的文件：启用后，每个报表页面都会导出到一个单独的文件中；文件名是通过后缀下划线+页码从指定的文件名派生的

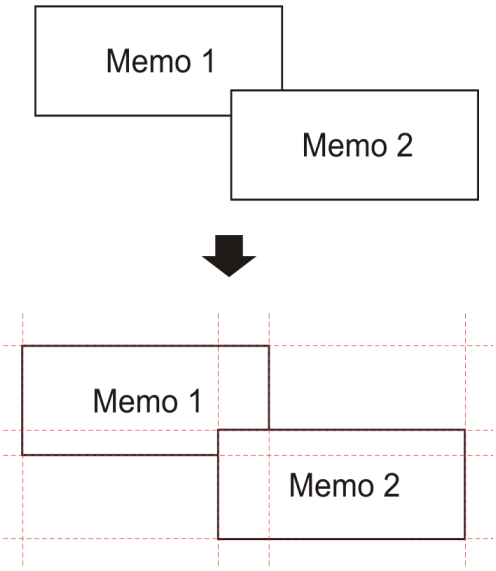
- 单色：导出为单色图像
- 裁剪页面：裁剪页面边缘的空白区域
- JPEG品质：JPEG压缩比；仅在导出为JPEG格式时启用
- 分辨率：导出图像的图形分辨率

导出功能：禁用单独文件时，会创建一个非常大的文件。

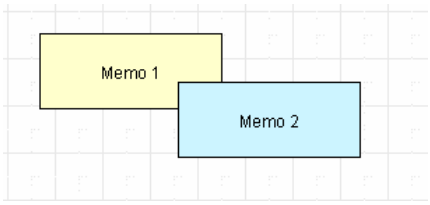
17.7 报表设计注意事项

任何格式的出口产出质量在很大程度上取决于原始报表设计的质量。 FastReport可以在创建报表期间以多种方式操作对象。这具有快速开发报表和后续打印的优点。打印文档看起来就像预览一样。这是FastReport报表引擎的主要目标。这种开发自由的缺点是将FastReport文档导出为不同数据格式的复杂性，每种格式都有自己的，有时是复杂的要求和限制。在本章中，将讨论用于出口的报表的特殊设计要求。

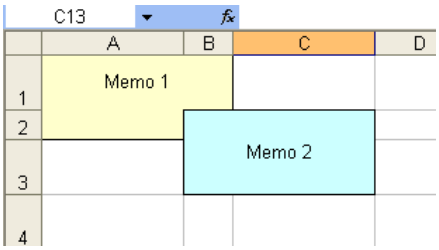
许多格式使用表格数据表示，例如HTML，XLS，XML，RTF和CSV。与FastReport页面设计中允许的自由度相反，当导出为这些格式时，为这些表格创建的输出单元格不能相交或按层排列。导出过滤器通常在FastReport导出对象时，通过使用处理交叉点的特殊算法和单元格的最佳放置来考虑这些要求。在对象交叉点处，在输出表中创建新的列和行。这是使FastReport准确定位对象并获得与原始预览页面的最佳对应关系所必需的。报表设计中的大量相交对象会在输出表中导致大量其他列和行。这反过来可能导致需要在适合进一步使用之前在其自己的编辑器中编辑导出的文件。



例如，获取一个报表，其中设计在同一段中两个对象略有重叠，报表中的记录数为150.在导出为RTF格式时，将创建450行（每行150行）和150行为了交集）。如果我们删除重叠，那么只有300行。对于具有大量对象的大型报表，差异会更大。当然，这会影响输出文件的大小。



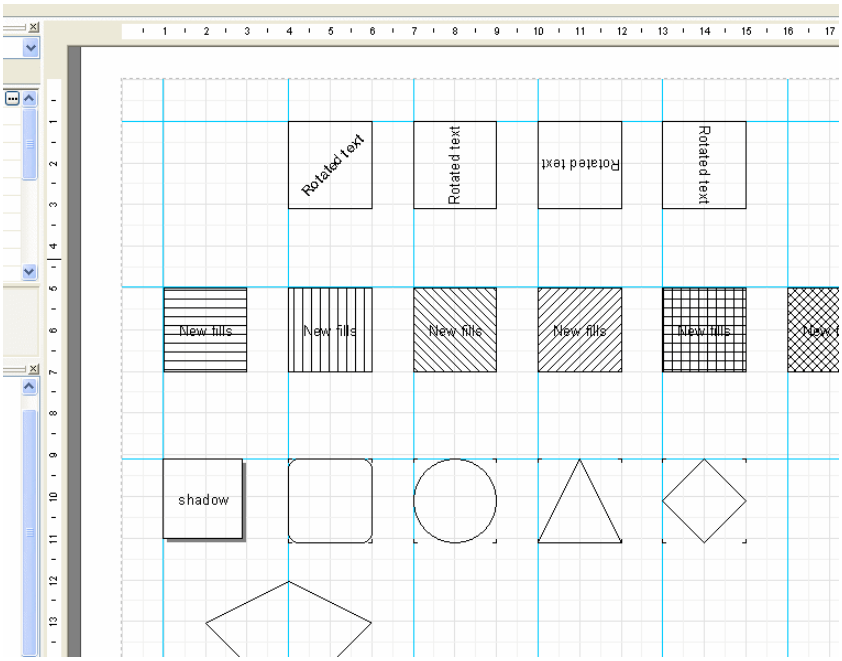
报表中的对象



导出到Excel – 结果

在设计要导出为使用“表格”输出方法的任何格式的报表时，请牢记这一点。

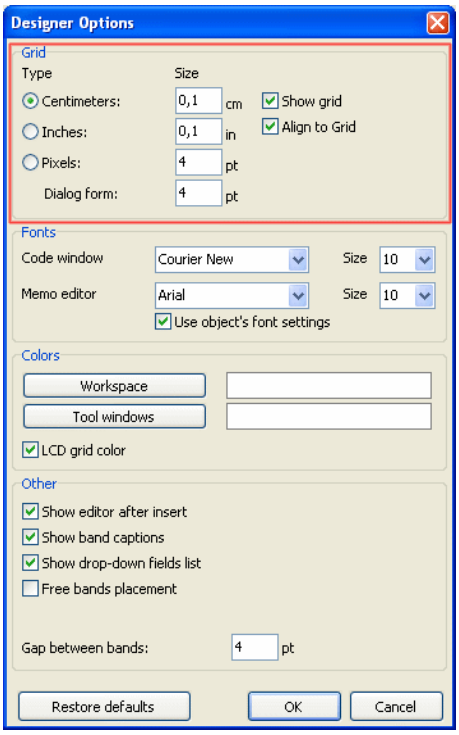
在报表中设计表格时，请注意相邻单元格的边界。重要的是细胞不重叠并且不以层的形式排列。导出算法可能以意想不到的方式处理单元格，并给出远离预期的结果。最好以垂直和水平放置的方式排列物体。指南可以帮助实现这一目标。



使用设计器中的指南

要在FastReport设计器中使用指南，只需单击报表页面顶部或左侧边缘的水平或垂直标尺，然后将标尺拖动到页面上的所需位置，此处将显示指南。然后，您将能够放置与这些水平和垂直指南对齐的对象。

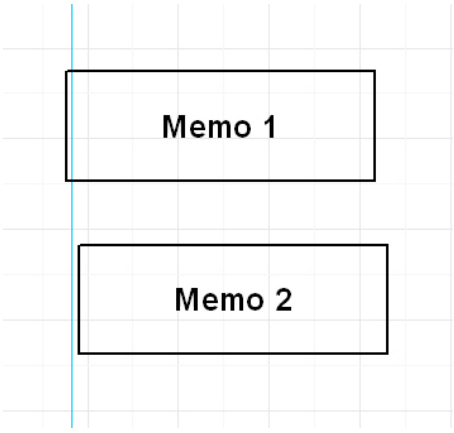
网格对齐也有助于放置“文本”对象以避免它们重叠。网格在设计器选项中启用，其中也可以调整音高：“视图>选项>网格”。



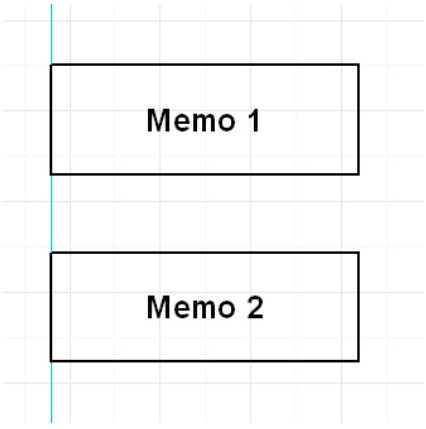
在“文本”对象周围使用框架时，最好使用对象的框架属性，而不是在文本周围添加线条和矩形等图形对象。另外，尽量不要在透明文本对象下面的背景中使用对象。

记住这些简单的规则将帮助您创建一个报表，该报表在导出到使用基于表的输出方法的任何格式后看起来都很完美。

下面是一些对象排列好与坏的例子。

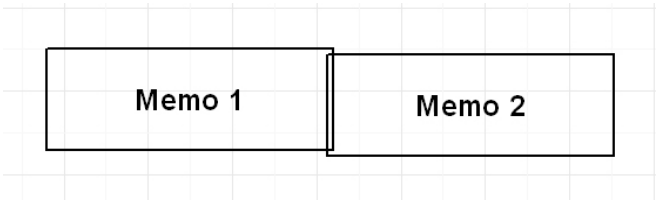


坏

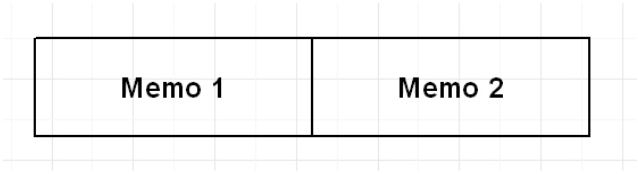


好

物体水平移动 – 它们不与垂直准线水平对齐。



坏



好

对象重叠 - 在导出到表/图表格式时，会创建其他不必要的行和列以及重叠区域中的三个附加单元格。

建议研究FastReport安装中包含的演示报表，以帮助掌握良好报表设计的基本原则。