

# **Quartam Software**

---

# **Reports**

for LiveCode

**Quartam Reports  
for LiveCode  
■ version 1.1.6**

# **User Manual**

© Copyright 2003-2011, Quartam Software/Jan Schenkel. All Rights Reserved Worldwide.  
Portions © 1999-2011, Runtime Revolution Ltd. All Rights Reserved Worldwide.

No part of this manual may be reproduced, stored in a retrieval system, or transmitted in any form or by any means, without the prior written permission of the copyright holder, except in the case of brief quotations embodied in critical articles or reviews.

The author and publisher have made every effort during the preparation of this manual to ensure accuracy of the material. However, the information contained herein is provided without warranty, either express or implied. Neither the authors nor the publisher, nor its dealers or distributors will be held liable for any damages caused or alleged to have been caused either directly or indirectly by this manual.

Quartam and Quartam Reports are trademarks of Quartam Software/Jan Schenkel. All other trademarks used herein are trademarks or registered trademarks of their respective owners.

# Contents

|  |    |
|--|----|
| Foreword by Dan Shafer .....                               | 7  |
| Acknowledgements .....                                     | 9  |
| About the Author .....                                     | 11 |
| Introduction .....   | 13 |
| What is Quartam Reports ? .....                            | 14 |
| The structure of this manual .....                         | 15 |
| Part One: About Quartam Reports .....                      | 17 |
| Introduction .....   | 18 |
| Technology .....   | 19 |
| Features .....   | 21 |
| Benefits .....   | 22 |
| Two editions: Standard and Professional .....              | 23 |
| What's new in version 1.1 .....                            | 24 |
| Improvements for all editions: .....                       | 24 |
| Improvements for Professional edition only: .....          | 24 |
| Part Two: Installation .....                               | 27 |
| Chapter 1. Installing Quartam Reports .....                | 28 |
| Windows installation procedure .....                       | 28 |
| MacOS X installation procedure .....                       | 29 |
| Chapter 2. Using the Quartam Reports Library .....         | 30 |
| Important notice about standalones .....                   | 31 |
| Part Three: Tutorial .....                                 | 33 |
| Chapter 3. Getting to know the Layout Builder .....        | 34 |
| Introducing the Layout Builder .....                       | 34 |
| Reports and Label sets .....                               | 34 |
| Groups and Bands .....                                     | 36 |
| Adding items .....   | 38 |
| Inspector palette .....                                    | 41 |
| Alignment palette .....                                    | 43 |
| Chapter 4. Report building strategy .....                  | 45 |
| What do I want to print? .....                             | 45 |
| How do I want to organize it? .....                        | 45 |
| Do I need to track subtotals or similar data? .....        | 46 |
| Where do I get the data? .....                             | 46 |
| What about the font, size and style? .....                 | 47 |
| Chapter 5. Stacks and databases .....                      | 48 |
| Printing stacks .....                                      | 48 |
| Printing database records .....                            | 55 |
| Chapter 6. Combining data sources using data brokers ..... | 61 |
| Skeleton of a data broker .....                            | 61 |
| Multiple database queries .....                            | 63 |
| Internet pages .....                                       | 67 |
| Part Four: Layout Builder .....                            | 71 |
| Chapter 7. The application menubar .....                   | 72 |
| The File Menu .....  | 72 |
| The Edit Menu .....  | 72 |
| The Tools Menu .....                                       | 73 |
| The Help Menu .....  | 74 |

|   |     |
|---|-----|
| Chapter 8: The application toolbar..... | 75  |
| New .....                               | 75  |
| Open .....                              | 75  |
| Save.....                               | 75  |
| Cut .....                               | 75  |
| Copy .....                              | 75  |
| Paste.....                              | 75  |
| Undo.....                               | 76  |
| Inspect .....                           | 76  |
| Align.....                              | 76  |
| Help .....                              | 76  |
| Chapter 9: Layout Editor windows.....   | 77  |
| Toolbar .....                           | 77  |
| Document settings.....                  | 79  |
| Data groups .....                       | 80  |
| Layout bands.....                       | 81  |
| Chapter 10: Property palette .....      | 83  |
| Basic.....                              | 83  |
| Condition.....                          | 90  |
| Size & Position .....                   | 91  |
| Colors & Inks.....                      | 92  |
| Chapter 11: Alignment palette.....      | 93  |
| Nudge.....                              | 93  |
| Resize.....                             | 93  |
| Align.....                              | 93  |
| Chapter 12: Keyboard shortcuts.....     | 94  |
| Part 4 : API Reference .....            | 95  |
| License and Version .....               | 96  |
| qrtReports_InitLicense .....            | 96  |
| qrtReports_VersionNumber.....           | 96  |
| Printing LiveCode stacks.....           | 97  |
| qrtReports_PrintReportForStack .....    | 97  |
| qrtReports_PrintLabelSetForStack .....  | 97  |
| Printing database cursors .....         | 98  |
| qrtReports_PrintReportForCursor .....   | 98  |
| qrtReports_PrintLabelSetForCursor ..... | 98  |
| Printing automated queries.....         | 99  |
| qrtReports_PrintReportForQuery .....    | 99  |
| qrtReports_PrintLabelSetForQuery .....  | 99  |
| Printing with data brokers .....        | 100 |
| qrtReports_PrintReport .....            | 100 |
| qrtReports_PrintLabelSet.....           | 100 |
| qrtReports_EndOfReport.....             | 101 |
| qrtReports_ValueOfExpression .....      | 101 |
| qrtReports_MoveNextRecord .....         | 102 |
| qrtReports_MoveNextGroup.....           | 102 |
| qrtReports_MoveNextPage .....           | 103 |
| qrtReports_RecoverFromError .....       | 103 |
| Handling format errors.....             | 104 |
| qrtReports_HandleFormatError .....      | 104 |
| qrtReports_SetFormatErrorHandling.....  | 104 |

|  |     |
|--|-----|
| Miscellaneous commands and functions ..... | 105 |
| qtrReports_GetCurrentPage.....             | 105 |
| qtrReports_SetPreviewWindowStyle .....     | 105 |
| qtrReports_SetProgressWindowStyle.....     | 106 |
| Appendix A: License Agreement.....         | 107 |



## Foreword by Dan Shafer

Discovering a new programming tool is one of those happy-sad experiences for most of us. First we're happy we found something useful that is perhaps better than what we've been using. Then we're sad because it doesn't do some specific things we're used to doing with our old tool. Then we're happy again because we find ways of either doing those things or improving on them. Then we're sad because some third-party library we really need can't be ported to the new tool.

And so it goes.

My engagement with Runtime Revolution is like that in many ways. I came to Revolution from HyperCard (its distant ancestor) via Smalltalk, JavaScript and Python. In all of those last three environments, I missed the elegance of the HyperTalk syntax, which Transcript restores and then explodes magnificently.

But as I began to build applications for myself, my wife, my friends, my colleagues and ultimately for my customers, I found myself stymied by Revolution's report-generating capability which was, to put it kindly, limited. Early versions of the program included some fairly primitive tools but eventually Runtime Revolution saw the futility in providing a half-baked solution and pulled those features from the product.

Although it is possible, using just built-in Transcript and the table field object in Revolution 2.6 to lay out and print some fairly basic reports, anything requiring careful layout or sophisticated calculations can be a trying exercise.

So I was sad.

But then along came Quartam Reports. This product enables me to very quickly lay out fairly sophisticated-looking reports that can include every kind of calculation I ever need to do and to have those reports print correctly on both Macintosh and Windows systems (no mean feat, I can tell you from personal experience). On top of that, if I end up needing a calculation that isn't built into Quartam Reports, I can write a function to perform it and include it in the report layout. That's a nice feature. (By the way, Quartam Reports was written entirely in Transcript, showing you what the language is capable of in the hands of a wizard. It's not Transcript that limits printing, it's the Revolution development environment.)

I suspect that Jan Schenkel, the creator of this fine product, will probably be largely responsible for any significant penetration of the enterprise market Revolution is able to achieve in coming years because I suspect its lack of a solid and useful report generating tool is about the only thing holding it back in many corporations that adopted HyperCard only after a third party product gave that program usable reporting capabilities.

And Jan's not done. This release of the product is a giant leap forward for us Revolution developers but you can be sure Jan is in his kitchen right now cooking up another set of features for future releases of the product, anticipating the places that his customers will suddenly become sad. Because Jan likes making people happy.

Like he's made me with Quartam Reports.

Dan Shafer

Author, "Revolution: Software at the Speed of Thought"

<http://www.shafermedia.com>





## Acknowledgements

*Quartam Reports* would not have been possible without the help and the patience of a number of good people.

In particular, I would like to thank my partner *Trisha Jacobs* for putting up with my spending countless hours in front of the computer, *Yvette Latoir at Professional Software* for allowing me to finish this project, *Kevin Miller and the others at Runtime Revolution Ltd.* for making *LiveCode* (née *Revolution*) the best cross-platform software development solution on the planet, and *the crew of beta testers* who gave endless amounts of advice, pointed out problems as well as solutions, and fed me ideas for years to come.

In random order, the original private alpha and beta testers: Tuviah Snyder, Richard Gaskin, Paul Looney, Rob Cozens, Frédéric Rinaldi, Ro Nagey, Alex Rice, Ken Ray, Monte Goulding, Chipp Walters, Till Bandi, Jim Sims, Dan Shafer and Graham Samuel.



## About the Author

Some things are just bound to happen. When Jan's mother was carrying him around in her womb, she spent the eighth month of her pregnancy in France, where she was taking an advanced course in assembler-programming for a mini-computer long since forgotten.

When he was 10, his father brought home one of the first Macintosh computers on Belgian soil, and had a hard time convincing Jan that sleep was a necessity. But the arrival of HyperCard was the trigger that made his nose jump up from his books and point straight towards the computer screen. During high school, his attention was focused on Medieval history, as well as the sort of languages humans speak. But slowly his other hobby started to take up much more time: programming his Mac SE/30 using HyperCard and Think Pascal.

After some turbulent college years, he joined the country's workforce just in time to upgrade FoxBase, FoxPro and Omnis applications in the infamous Y2K era. Armed with his analytical mind and love for data structures, he moved on to become a functional analyst at VERO -- a leading Belgian software company that builds powerful Finance, ERP and CRM applications on Progress databases.

In a bold career move, he joined Professional Software, where he not only oversaw the euro-conversion of the applications and customer data, but also started the quest for a new development platform that would fulfil their needs.

"If you're looking for a development platform with database support that works on Mac and Windows, and preferably Linux and Solaris as well, one tool after another drops out of the race. In the end, we had to make a choice between Java and Revolution. The sheer jump in productivity was enough for us to pick Revolution."

A good chunk of this research and development is paying off for the Revolution/LiveCode community in the form of Quartam Reports, and there's more in store for the near future. Building a powerful tool is a swift process, but turning it into an actual solid product has proven a lot more time-consuming. But you will no doubt agree that it has been worth the wait.

Jan has extensive experience with Progress OpenEdge, FoxPro, Java, FileMaker Pro, various SQL-databases, and of course, Revolution/LiveCode.

Apart from developing tools for his peers, he is occasionally available for custom software services and project consultancy.



## Introduction

## What is Quartam Reports ?

Data is the basis of knowledge. But it isn't enough to just have a collection of individual facts : you have to track and compare these items to learn more and interpret the raw data. *Collecting data into meaningful reports creates information.* And this information allows decision makers to analyse trends, draw conclusions and act accordingly.

*LiveCode* is the multi-platform user-centric software development tool of choice for Windows, Macintosh, Linux and popular Unix systems. Using *LiveCode*, you can easily access data from all over the internet, from files on your hard disk, from millions of records in that corporate database.

The screenshot displays the Quartam Reports software interface. The main window shows a report design canvas with a yellow header area and a table of data. The table has columns for Article, Invoice, Date, Amount, Price, Total, Commission, Unit, Costprice, and Profit. The report is titled "Quartam Reports Sample 2 :: print a query".

On the right side, there are two panels:

- Properties**: This panel shows the report name "sample8.qrl" and the selected item "field 1005 of Detail line". It includes tabs for Basic, Condition, Size & Position, and Colors & Inks. The Basic tab is active, showing the field expression "det\_amount \* det\_price", the aggregate function "<None>", and the format "Numeric" with 2 decimals. It also has text formatting options like Font, Style, Size, and Margins.
- Alignment**: This panel shows a nudge tool with a value of 10 and an increment button. It also has a Resize tool and an Align tool.

But collecting the data is only the first step - so we are providing you with the tools to turn it into information : whether you need to print a simple form, a delivery note, a statistical report or integrate data from various sources, *Quartam Reports for LiveCode* is at your service.

## ***The structure of this manual***

### **Part 1: About Quartam Reports**

In this part of the manual, you will get a general overview of *Quartam Reports for LiveCode*, the technology behind it, the features and benefits to you as a developer.

### **Part 2: Installation**

In the second part of the manual, you get information on how to install *Quartam Reports* on MacOS X and Windows. You will also learn how to use the *Quartam Reports* report printing library in your own projects.

### **Part 3: Tutorial**

In the third part of the manual, the focus is on examples. First you will play with the layout builder to get a good grasp of how it works. The real work starts with a report-building strategy that I and others have used with success in the past. You will learn how to create reports that print data from LiveCode stacks and automated database queries, as well as how to script data brokers that give you full control over the report generation process.

### **Part 4: Layout Builder**

In the fourth part of the manual, we go through the different parts of the layout builder. It covers the menubar, the toolbars and the layout editor windows, as well as the inspector and alignment palettes.

### **Part 5: API Reference**

The fifth and final part of the manual describes the Application Programming Interface of the *Quartam Reports* report printing library. Every command, function and callback message is explained with syntax, examples and additional notes.





## **Part One: About Quartam Reports**

## ***Introduction***

You know the drill : all that data that goes into the system, must come out again -- as invoices, reports, label sets, and the like. You have the data from the database, but that is just the beginning ; you need to track subtotals, include headers and footers per data group, etc. ; not to mention putting it at the right spot on the paper.

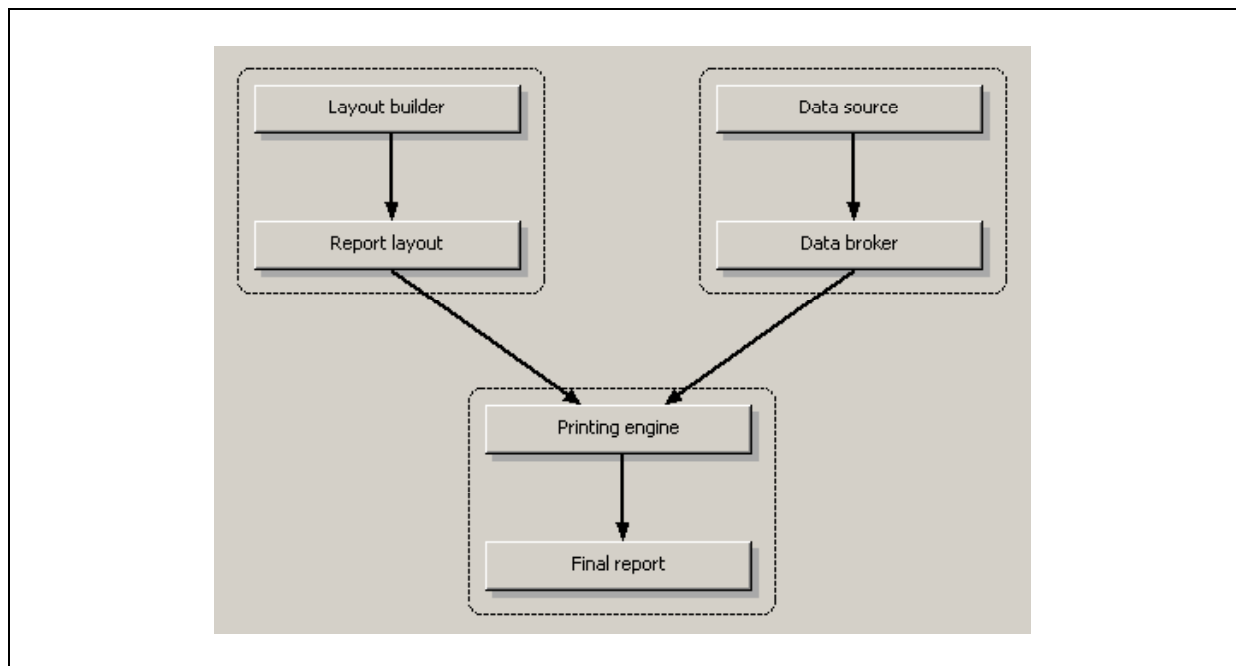
*Quartam Reports for LiveCode* is here to make your life a lot easier. It is the first professional reporting tool for this multi-platform software development tool, packing all the features you would expect from traditional report building tools :

- Use the layout builder to assemble your report using familiar tools, by placing text labels, data fields, graphics and images, dazzling everyone with colors and inks.
- Add page header and footer bands for those elements that need to be printed on every page.
- Organize your data in groups, each with their own header and footer, keeping track of statistical information through aggregate functions (sum, count, average, variance, standard deviation, etc.).
- Fill your fields with data using everyday LiveCode expressions, then format them as styled text, numbers, dates, times or using advanced Printf()-expressions.
- Unicode text printing is as easy as picking a menu option, while powerful expression editors help you adapt the data to your information needs.

*Quartam Reports* brings the advantages of traditional report building tools to the cross-platform world of *LiveCode*. It is the only tool to let you print reports and labels while offering on-screen preview, data grouping, aggregate functions, and much more, through a cross-platform library with a simple API as well as an advanced callback architecture.

## Technology

The diagram below shows how the different parts come together.



## Data Sources

Data can come from many sources : a stack, local or remote databases, a web-page somewhere on the internet, the result of a SOAP-call -- LiveCode is an excellent tool to obtain data from various sources, and prepare them for printing in complex, integrated reports.

Relational databases like *MySQL*, *Oracle*, *PostgreSQL*, *SQLite* and *Valentina*, are the most common way of storing significant amounts of structured data in an easily retrievable way. SQL-queries are generally quite readable, just like LiveCode commands and expressions they are English-like.

LiveCode also offers an easy way to connect to databases : you use the Database Query Builder to create a query, and then you link fields, checkboxes, menus and buttons.

Nevertheless, these relational database management systems (RDBMS) present a certain overhead, such as the need to install server software, and limitations in the supported platforms. That is why many LiveCode programmers prefer to store data right in stacks, as these are flexible and can be read by standalones, cross-platform and without additional software.

But in the days of the Information Age, data is everywhere and can be stored as a static web-page or a dynamic LAMP-driven website. With the advent of *XML-RPC* and *SOAP* computers can connect with one another more easily than ever to exchange information with barely more intervention from the user than making sure that the computer is connected.

**Quartam Reports** provides you with a simple yet powerful library to connect the dots : with a single command you can print data from a stack or an automated query -- and when you need more control, you can script your own data broker which fetches related information from a local file, a remote website, through Web Services, etc.

## Report Layouts

We've said it before, but it bears repeating : data requires presentation to become information -- information allows decision makers to analyse trends, draw conclusions and act accordingly. You want to group data into logical sets, providing subtotals and statistical information on the data. To improve the look of your reports, you apply colors and inks, combining images graphics and data with interesting visual effects that make the important data stand out.

The *Quartam Reports Layout Builder* is used to prepare the look of your reports and label sets. Modeled after traditional report layout tools as found in FoxPro, FileMaker Pro and Access, it allows you to create the layouts in an intuitive manner, and save them as external files. You create the look by placing graphics, labels, images and setting the font, size, style and format of fields. Like you would expect, data fields form the corner stone of your report : you can use valid LiveCode expressions such as

```
field "First name"
```

or

```
det_amount * det_price
```

to determine what is printed inside the report fields.

## Printing Engine

At the heart of *Quartam Reports* is a flexible and powerful printing engine -- a single cross-platform library that doesn't require additional software, as it was written entirely in LiveCode.

It reads the report layout file, and uses this as a template to build the individual pages, working closely together with the data broker, which evaluates the expressions that define the data in the report fields. The printing engine keeps track of variable data and offers an arsenal of statistical functions (sum, variance, etc.) so you can attain the insight you need.

The printing engine provides convenient commands for printing the content of stacks or automated database queries without the need for additional scripting.

However, we didn't stop there, but worked hard to provide you with a system of data brokers : objects that handle a well-defined set of callback messages, allowing your reports to integrate data from different sources in a flexible way.

In effect, you can combine data from a stack, a database query, an Internet URL, an XML-file, a SOAP-call, with only a modest script, without having to worry about how to place the data onto the right spot on the paper.

## Features

*Quartam Reports for LiveCode* is a powerful report building system, providing all you need to produce reports and label sets in a friendly, flexible environment. The following overview will highlight its main features.

### Layout builder :

- Layout building in bands, similar to FoxPro, FileMaker Pro, Access,...
- Support for both regular reports and label sets.
- Support for page header and footer bands.
- Support for title and summary bands.
- Support for label fields, data fields, graphics and images.
- Support for barcodes and charts. (*Professional Edition*)

### Layout items :

- Label fields for static data.
- Data fields for dynamic data.
- Graphics: rectangles, lines, ovals, polygons,...
- Images for company logos.
- Barcodes: EAN-8, EAN-13, UPC-A, Codabar, Interleaved 2 of 3, Code 3 of 9, Postnet.
- 2D Charts: bar chart, column chart, line chart, pie chart.

### Data groups :

- Related sets of data can be grouped using everyday LiveCode expressions.
- Each data group has its own header and footer band.
- Aggregate functions help you track statistical information in data fields : count, sum, average, min, max, median, variance, standard deviation.
- These aggregate data fields can be reset upon page change, upon group change or simply continue to be updated until the end of the report.

### Data fields :

- Field content is described using everyday LiveCode expressions.
- Field content can be passed as plain text, unicodeText, htmlText or rtfText.
- Field content can be formatted as numbers, dates, times or by means of a Printf() expression.

### Scripting power :

- Simple library calls for printing data from stacks, SQL result sets and automated queries.
- An advanced callback system to script complex data brokers that combine data from multiple data sources, XML-files, the internet, Web Services, and much more. (*Professional edition*)

### Miscellanea :

- LiveCode expression editor for field expressions, printing conditions and data groups.
- Printf() expression editor for complex field formats.

## **Benefits**

*Quartam Reports for LiveCode* is here to make your life a lot easier. It is the first professional reporting tool for this multi-platform software development tool, packing all the features you would expect from traditional report building tools.

**No longer do you have to painstakingly program each report individually, because...**

**... everything you need is there :**

- page headers and footers
- text labels, data fields, graphics and images
- data groups with their own headers and footers
- aggregate functions to track statistical data
- print preview for your end-users

**... it is very easy to use :**

- the layout builder helps you assemble your report using familiar tools
- no need to learn a new programming language, as all calculations are done using everyday LiveCode expressions -- we even provided an expression editor
- simplified printing commands for stacks and database records make reporting a snap

**... but packs a lot of power :**

- text formatting options include Unicode, HTML and RTF
- more formatting options for numbers, dates and times
- flexible formatting by means of Printf()-expressions

**... and comes in two editions :**

- Standard Edition gives you all you need to print reports and labels sets
- Professional Edition adds barcodes, 2D charts, text export and data brokers.

**... offering room for growth :**

- minimal scripting combines data from various sources into one comprehensive report
- the advanced callback architecture allows you to create your own *data brokers*

Using *Quartam Reports for LiveCode* you can make professional reports and label sets in no time. It is extremely easy to use for small projects, but is equally at your service for the complicated reports of tomorrow, that need to integrate data from many sources.

With *LiveCode* and *Quartam Reports* the possibilities are endless !

## ***Two editions: Standard and Professional***

Quartam Reports for LiveCode is available in two editions: Standard and Professional.

Both editions share the report layout builder and use the same report printing library.

Both editions give you the full power of data fields with LiveCode expressions, aggregate functions and complex calculations.

Both editions now offer data groups, adding group headers and footers to related sets of data.

Both editions allow you to easily print data from a stack, database cursor or automated query.

The Professional edition adds the advanced features of barcodes, 2D charts, text export and data brokers to the mix.

Barcodes are widely used in retail and warehouse management. The current version supports a number of industry-standard formats, including UPC-A and EAN-13.

The brand new 2D charts allow you to clearly visualize trends, shares, etc.

Text export takes the data from your report and makes it available for analysis in other environments like Microsoft Excel.

Data brokers form the powerful glue between the report printing library and complex data source combinations.

While the Standard edition offers all you need to produce everyday reports and label sets, the Professional edition offers advanced features and a path for growth.

So when you need that extra feature, it is easy to upgrade and you don't have to start looking for a different tool.

### **IMPORTANT NOTICE:**

Both of these licenses include a copy of the layout builder application FOR PERSONAL USE ONLY -- this means that you cannot give a copy of this application to your end-users along with the stacks or standalones that you distribute.

If you need to include a layout builder in your application so that your end-users can modify the report layouts, [contact us](#) for a special bundle license or a custom version that you can embed into your application.

## What's new in version 1.1

### Improvements for all editions:

- **Stretching data fields**  
The top feature request was data fields that would stretch to fit their content. *Quartam Reports 1.1* delivers on the promise made at the launch of version 1.0 - if the text does not fit on the current page, it is automatically split and new pages are added as needed.
- **Relative positioning**  
When you use stretching data fields, you also want to position other items in relation to this expansion - e.g. a line at the bottom of the band, below a stretching data field with remarks about the customer. *Quartam Reports 1.1* lets you position items at a fixed distance from the top or the bottom border of the band in which they reside.
- **Oval graphics**  
In addition to lines, rectangles and rounded rectangles, *Quartam Reports 1.1* allows you to draw oval graphics on the layouts of your reports and label sets.
- **Title and summary bands**  
A lot of reports benefit from a title section to provide introductory information, as well as a summary section to display aggregate information. *Quartam Reports 1.1* lets you set flags to add title and summary sections to your report layout.
- **Data groups for everyone**  
This feature used to be exclusive for the Professional edition, but is now also available in the Standard Edition. *Quartam Reports 1.1* lets you define data group, and print optional data group headers and footers, as well as track aggregate information.
- **New page upon group change**  
Sometimes you want to start printing the contents of a data group on a blank new page - e.g. when printing a commission report for your sales reps, you don't want to share the data of one sales rep with the other sales reps. *Quartam Reports 1.1* lets you set a flag for the data group, and will print the necessary data group headers on the new page.
- **Printing global variables**  
Sometimes you want to print the content of a global variable on your report, or use it in a printing condition, without having to script your own data broker. *Quartam Reports 1.1* adds an extra parameter to the simplified printing commands, so you can pass the names of the global variables that you want to use during the printing of a report or label set from the data of a stack, database cursor or automated query.

### Improvements for Professional edition only:

- **Barcodes**  
Another new feature in *Quartam Reports 1.1* is the ability to add barcodes to your reports and label sets - currently supporting the industry standards *EAN-8*, *EAN-13*, *UPC-A*, *Codabar*, *Interlaced 2 of 5* and *Code 3 of 9*, as well as *Postnet*.
- **2D Charts**  
Nothing helps you see numbers in perspective like a 2d chart (pun intended). *Quartam Reports 1.1* lets you display numeric information in the following chart styles:
  - Bar chart
  - Column chart
  - Line chart
  - Pie chart



- **Label and field text export**

You may want to take the data from your report and use it as the basis of an Excel worksheet where you can add charts, perform off-line analytical processing, etc. *Quartam Reports 1.1* lets your user export the text content of the labels and data fields to the following formats:

- Excel 2003 (Spreadsheet XML)
- HTML
- CSV
- Text (tab-delimited)

- **New data broker callbacks**

When you're implementing your own data broker, you may want to take certain actions before the report printing library moves on to printing a new page. *Quartam Reports 1.1* will now send the callback message 'qrtReports\_MoveNextPage' to your data broker after printing the page footer and before printing the header of the new page. Implementing this callback message is optional.

- **New data broker functions**

When you're implementing your own data broker, you may want to take the current page number into account when resolving data expressions. *Quartam Reports 1.1* now offers the function 'qrtReports\_GetCurrentPage', allowing your data broker to retrieve the current page number.



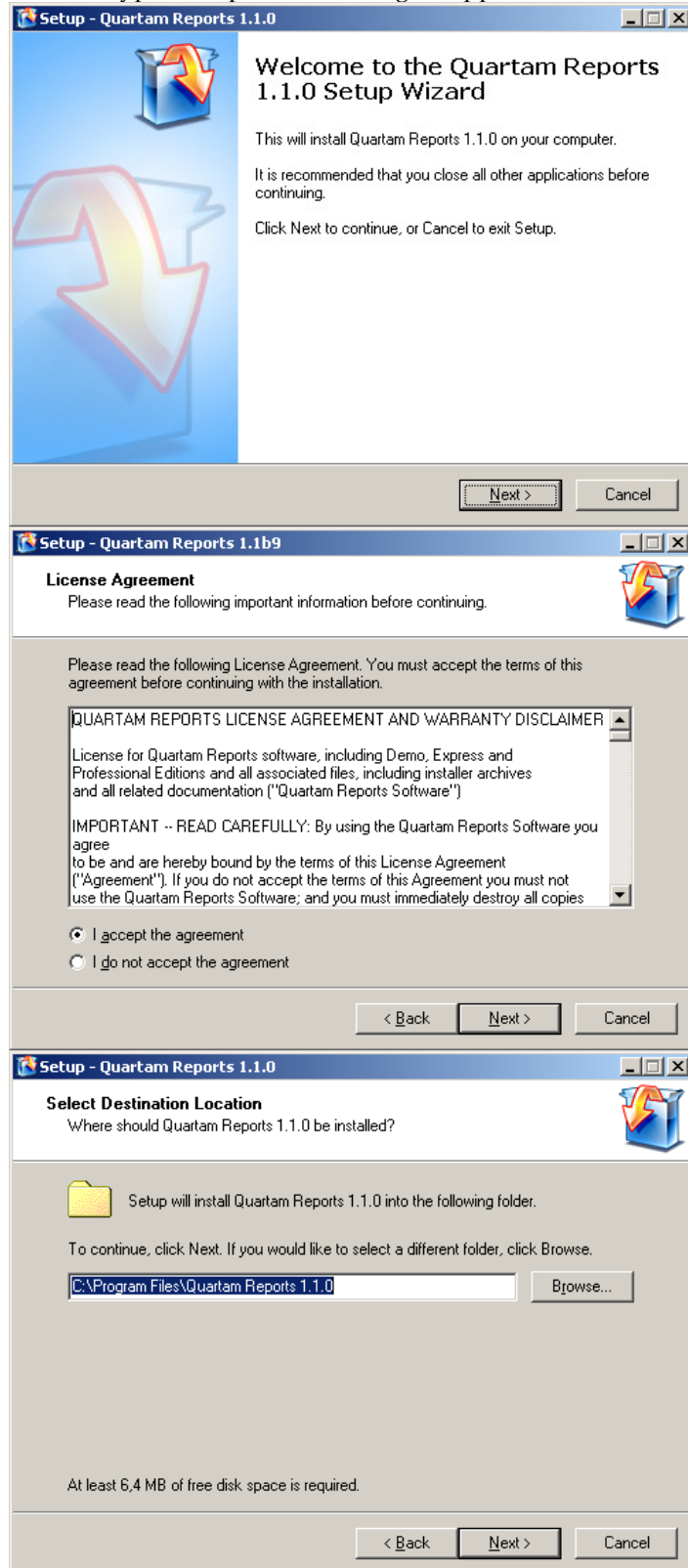
## **Part Two: Installation**

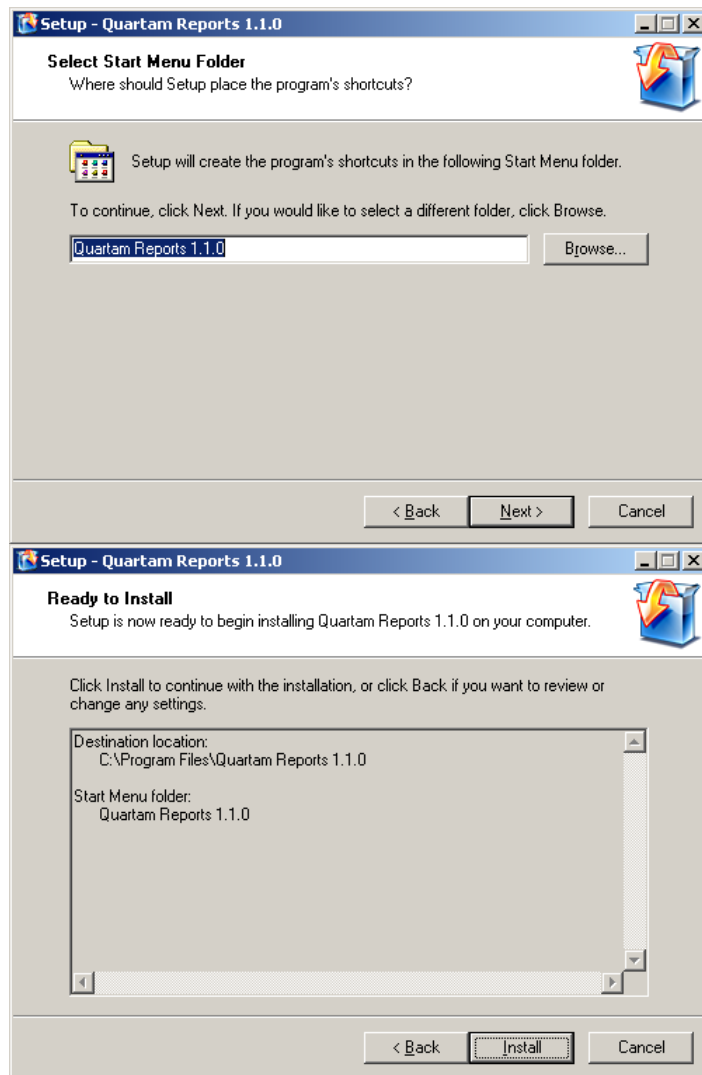
## Chapter 1. Installing Quartam Reports

### Windows installation procedure

Download the latest version for Windows at: <http://www.quartam.com/downloads.htm>

The installer comes in the form of a .EXE – executable file. Created with InstallGadget, it presents you with the typical steps for installing an application on Windows.





## MacOS X installation procedure

Download the latest version for MacOS X at: <http://www.quartam.com/downloads.htm>

The installer comes in the form of a standard .DMG disk image. Create a folder 'Quartam Reports 1.1' in your Applications folder, and drag the files from the unpacked disk image into this folder.

## Chapter 2. Using the Quartam Reports Library

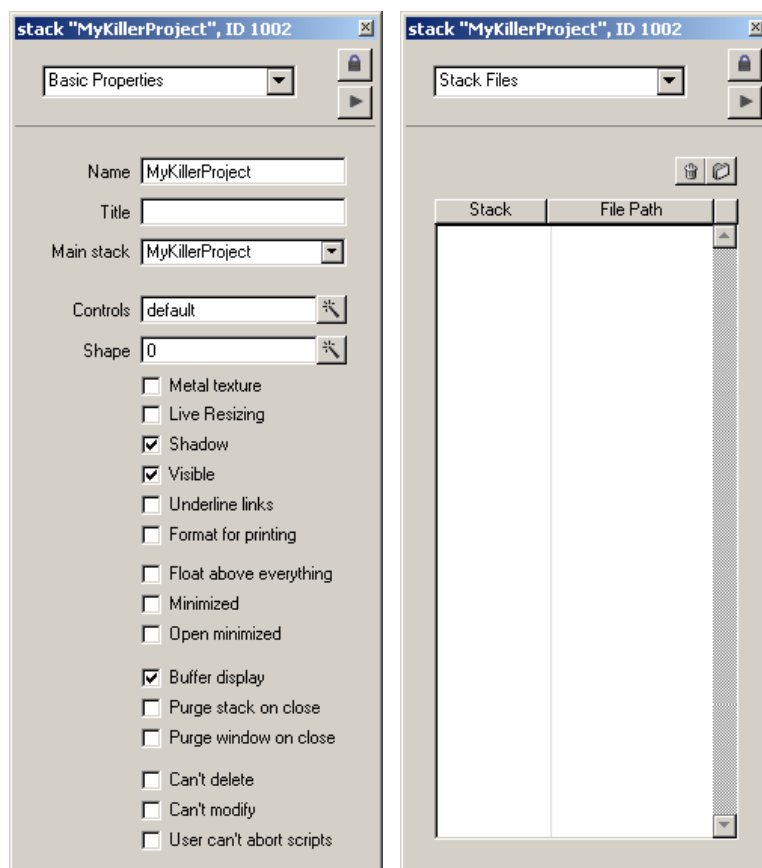
Now that you have installed the complete package, it's important that you know how to use *Quartam Reports* in your projects. Whereas the following chapters will guide you through building your first reports, this chapter focuses on using the report printing library in your projects.

### The library stack

At the core of *Quartam Reports for LiveCode* is the report printing library. This library is implemented in LiveCode and thus portable to any platform supported by LiveCode. The library and associated stacks are contained in the file "qrtReportsLib.rev" in the same location as the Layout Builder application.

We advise that you add this library stack to the stack files of your project's main stack, and load the library once at startup. As a bonus, adding it to the stack files, means that the LiveCode standalone builder will copy it when you build your project.

Open your project's main stack in LiveCode, go to menu 'Tools' and pick item 'Stack Inspector'. Then, use the option menu at the top to open the panel 'Stack Files'.



Click the small 'Folder' icon, to open a standard Open file dialog.

Traverse the file system structure to the folder containing the Quartam Reports Layout Builder, and select the file 'qrtReportsLib.rev'

#### MacOS X Hint:

If the dialog box won't let you select the stack, hold down the option-key as you click on the button – this will allow you to select any stack file and add it to the stack files.

### License key

When you purchased your copy, a license key was sent to you via email. You need this license key not only to unlock the Layout Builder application, but also to activate the report printing library. It is easiest to do this at the same time as loading the library.

## Setup script

Below you find a simple setup script that will load the library and initialize it with your license key.

```
on openStack
    if "qrtReportsLib" is not among the lines of the stacksInUse then
        start using stack "qrtReportsLib"
        qrtReports_InitLicense "<your license key here>"
    end if
end openStack

on closeStack
    stop using stack "qrtReportsLib"
end closeStack
```

Copy this into the script of your project's main stack, and you're set for printing reports and labels.

## Important notice about standalones

For the report printing library to work in your standalone applications, you have to include the standard LiveCode revXML library when building your standalone.

When you ask the standalone builder to search for inclusions, and it does not include the XML library, you will have to manually include it yourself. If you don't include the XML library, you will receive errors when trying to print reports or label sets.





## **Part Three: Tutorial**

## Chapter 3. Getting to know the Layout Builder

If you've used drawing applications or other reporting tools before, then the Layout Builder will look quite familiar: the items on your layouts are drawn using tools, properties are changed using an inspector, and you can align them with other items to make your reports shine.

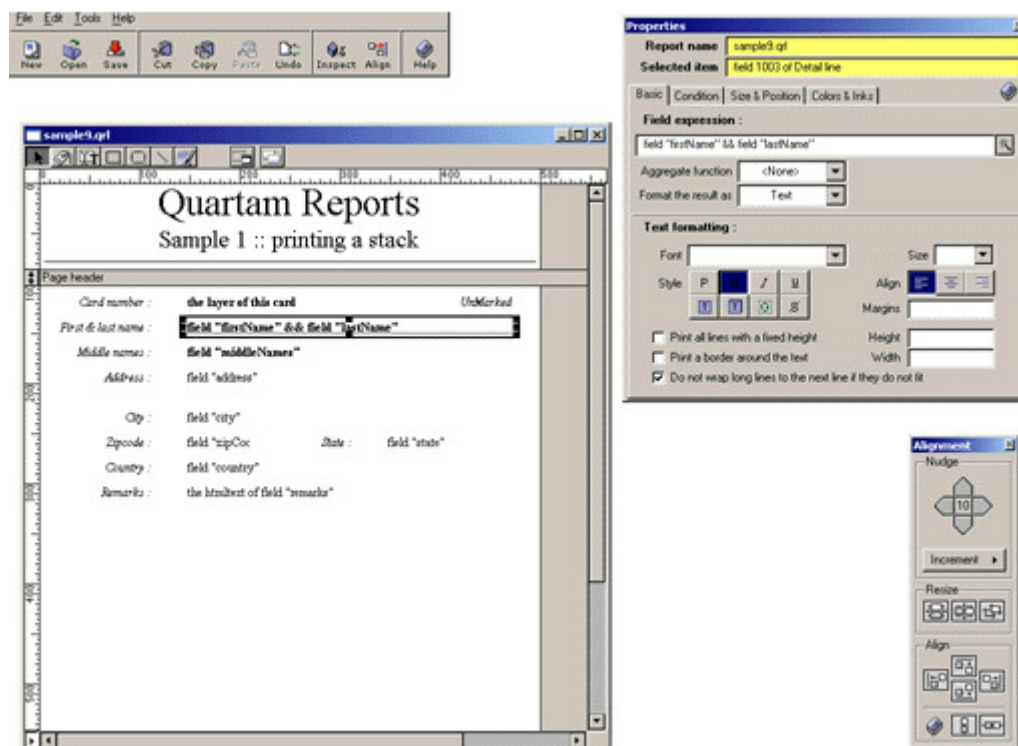
### Introducing the Layout Builder

When you launch the Layout Builder, you see the Welcome screen, where you can :

- Review the samples
- Create a new report layout
- Create a new label set layout
- Open an existing layout

But of course it's best to see the Layout Builder in action. As you can see, it sports a clean interface :

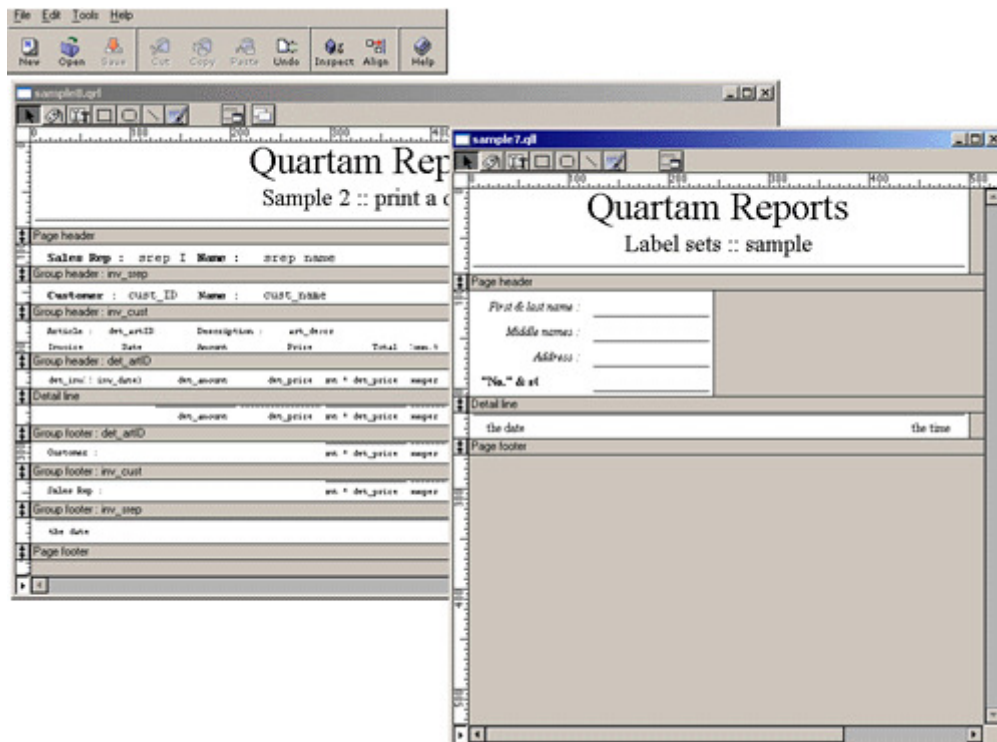
- The toolbar at the top provides access to the most commonly used features.
- Every layout window has its tools at the top.
- The items are organised in bands that you can easily resize.
- The properties palette lets you inspect the different settings of each item.
- The alignment palette helps ensure all items are in the right place.



### Reports and Label sets

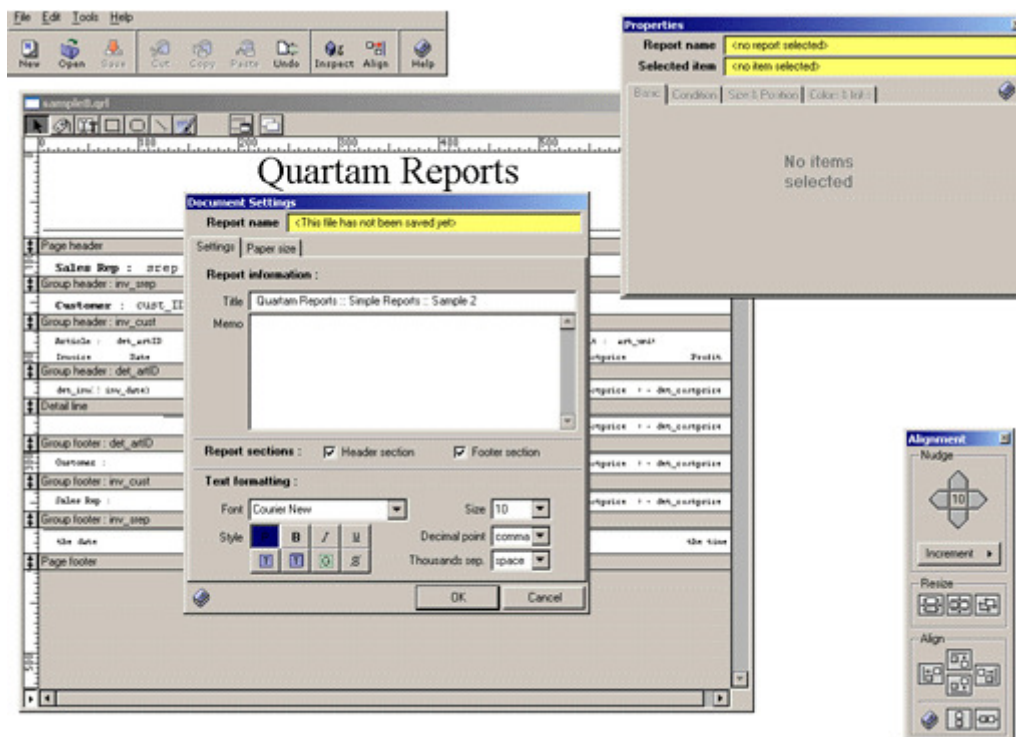
Data and presentation together create effective reports. But that's not the only type of information that gets sent to a printer : Label sets are ideal for mailing lists, price tags,...

*Quartam Reports for LiveCode* helps you make both reports and label sets.

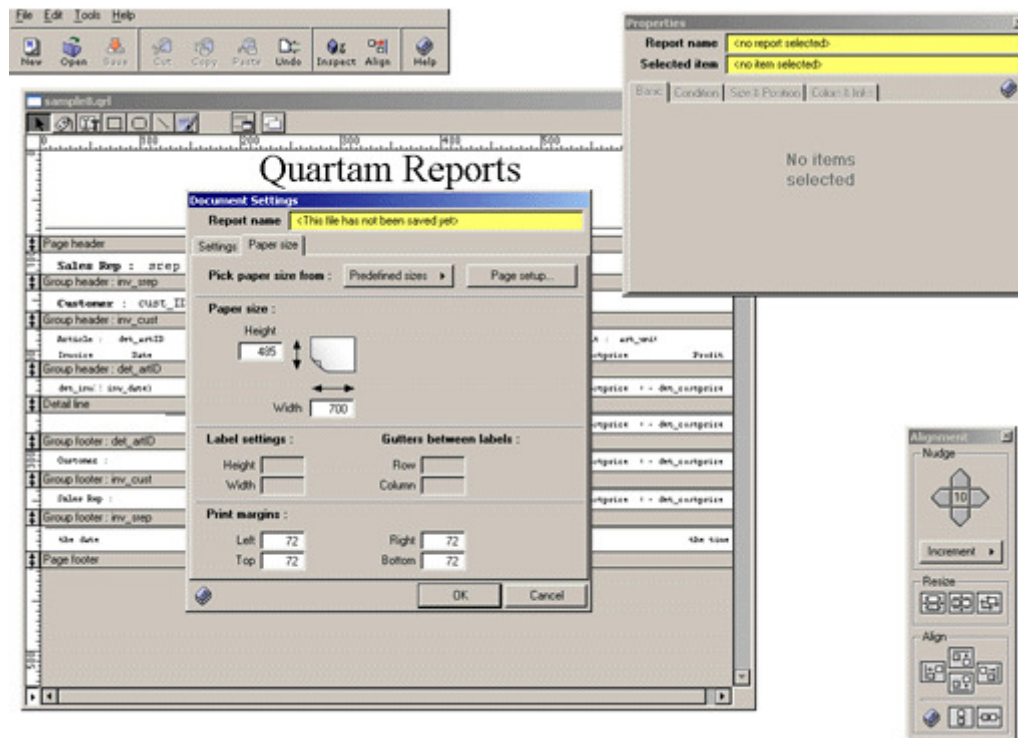


It is easy to setup reports and label sets. When you create a new layout, you are immediately presented with the 'Document Settings' dialog box. If you need to visit this dialog box afterwards, you can just click on the 'Document Settings' button at the top of your window.

In the first tab of the document settings, you can enter general report information, turn the Header and Footer sections on/off, turn the Title and Summary sections on/off, set the default font, size and style, as well as the decimal point and the thousands separator.



In the second tab of the document settings, you can set the paper size, setup your labels and determine the print margins. As soon as you click 'OK', your changes are reflected on the layout.



## Groups and Bands

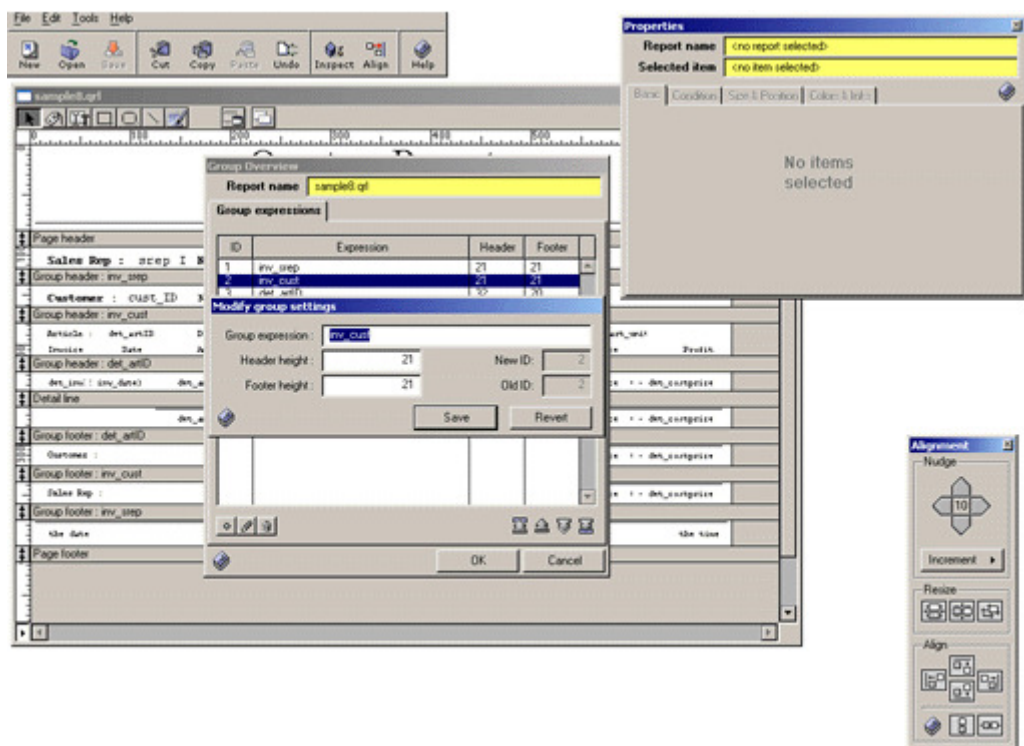
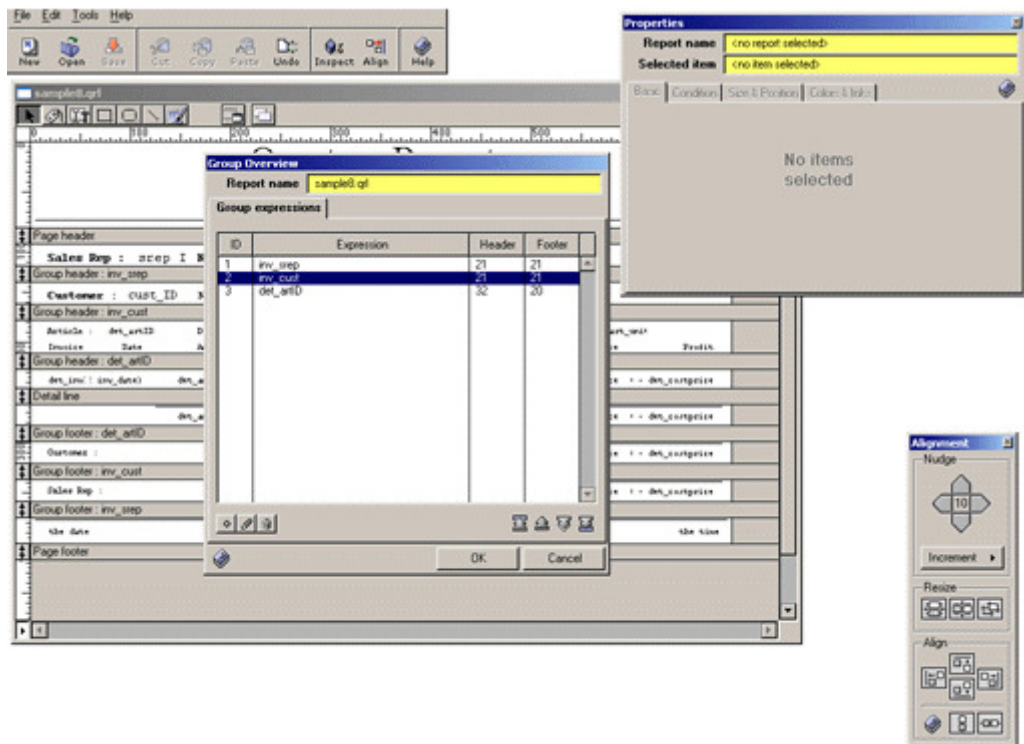
In *Quartam Reports for LiveCode*, the items on your layouts are organised into *bands*. In every report there is a *detail band*, as well as optional *header* and *footer bands*.

In your *detail band*, you provide a placeholder for every individual item that you want to print : the placeholder identifies the item and its properties, including its exact position within the band. If you're printing a stack, each printed detail band corresponds to a card. If you're printing a database cursor or automated query, each printed detail band corresponds to one record in the result set.

The optional *header* and *footer bands* are placed on every page of the report. This makes it easy to place certain items on every page : the company logo ; the title, date and time of the report ; page totals ; etc. The optional *title* and *summary bands* are placed on the first and last page of the report respectively. This makes it easy to place introductory information as well as a recap of all the data at the start and end of the report.

Many reports are based on data that is sorted into groups : the deals made by a single sales representative ; the articles sold to a single customer over a period of time ; the measurements from an Antarctic research station ; etc.

We help you organise this data into nested groups. Each group can have its own *header* and *footer bands*, so you can easily keep track of sales totals, average temperatures and other aggregate data. And all you need to know is a little LiveCode, as the data is grouped on the basis of everyday LiveCode expressions.





## Adding items

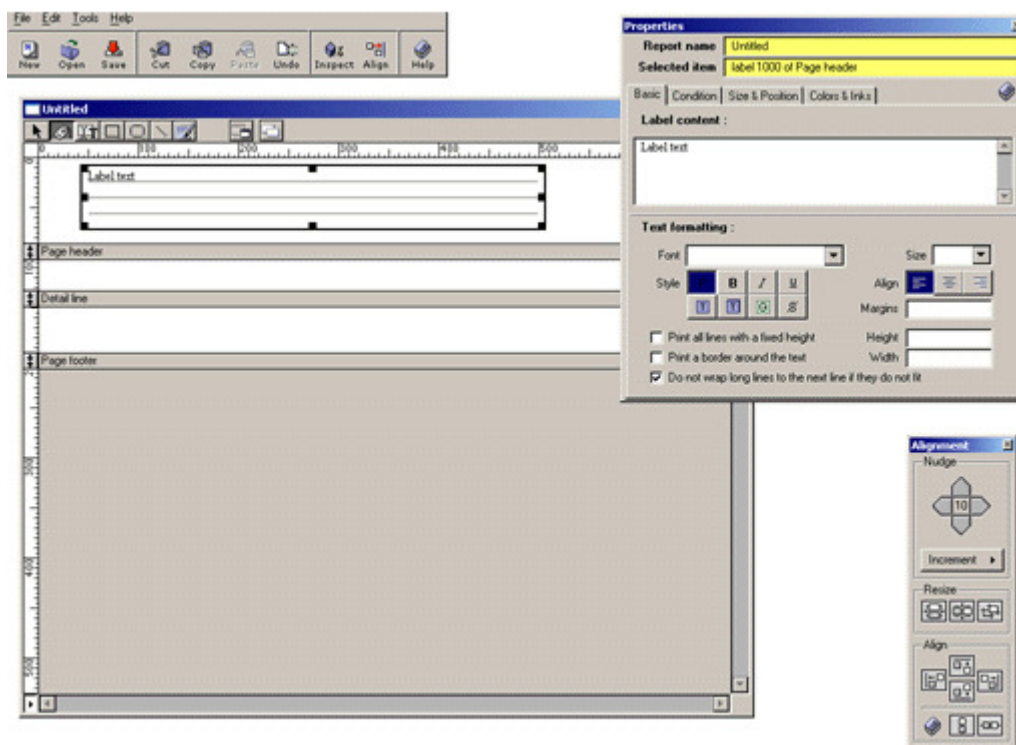
In *Quantam Reports for LiveCode*, you get support for the following item types :

- Label fields
- Data fields
- Graphics
- Images
- Barcodes (*Professional Edition*)
- 2D Charts (*Professional Edition*)

Let's start with an empty layout, and add a label field by using the *label field tool* at the top of the window.

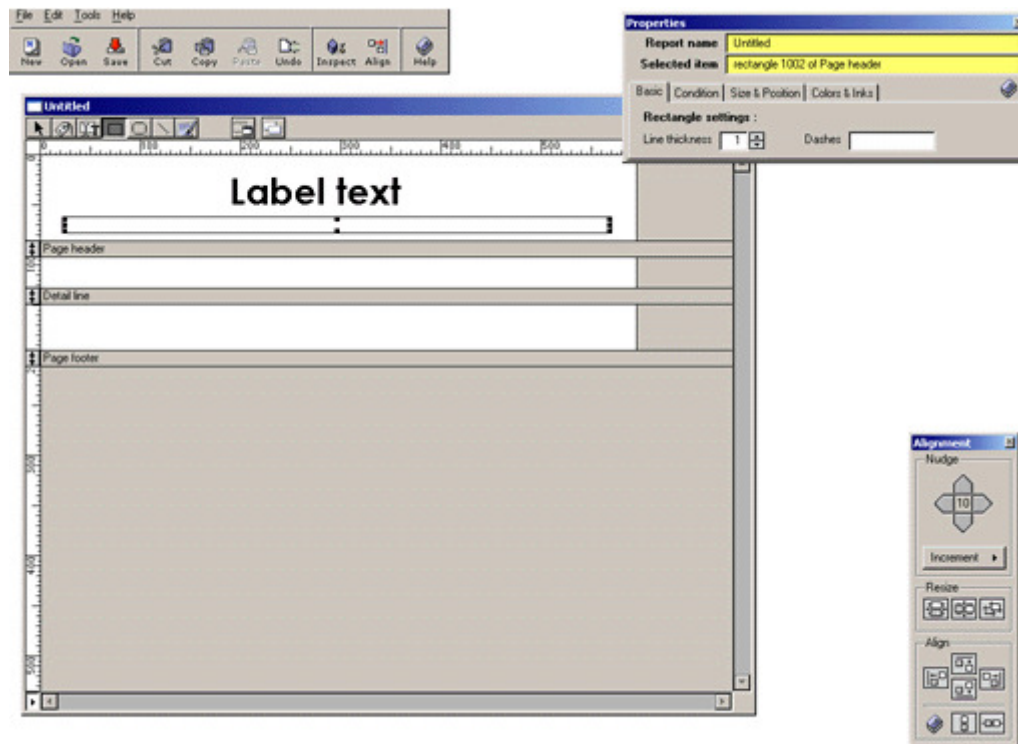
Just click on the tool icon, then click in one of the *bands* to start creating an item of that type. The starting point serves as an anchor, and you can drag in all different directions to make your new item the right size.

However, the new item cannot resize outside the bounds of the enclosing band. This ensures that you don't end up with items that are cut off when printed or are placed on, top of items in other bands.



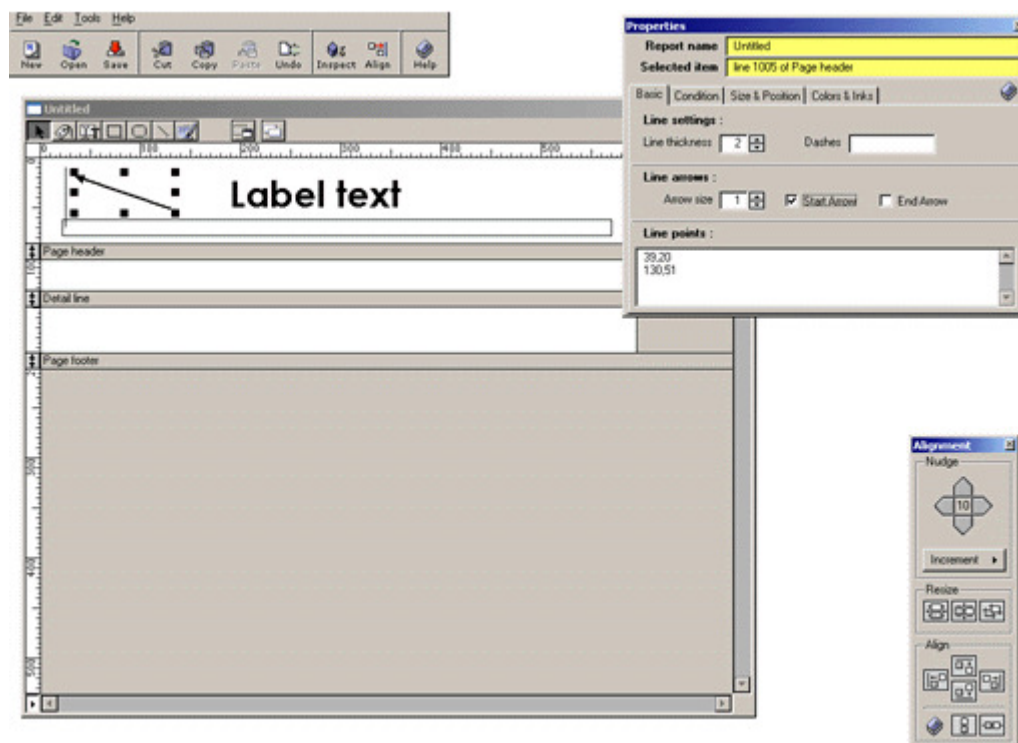
Using the Inspector Palette we can set the properties of the label field, such as the content, font, size, style,... Of course you recognise these from *LiveCode*.

After changing the properties of the label field, we'll add a rectangle by using the *rectangle tool* at the top of the window.



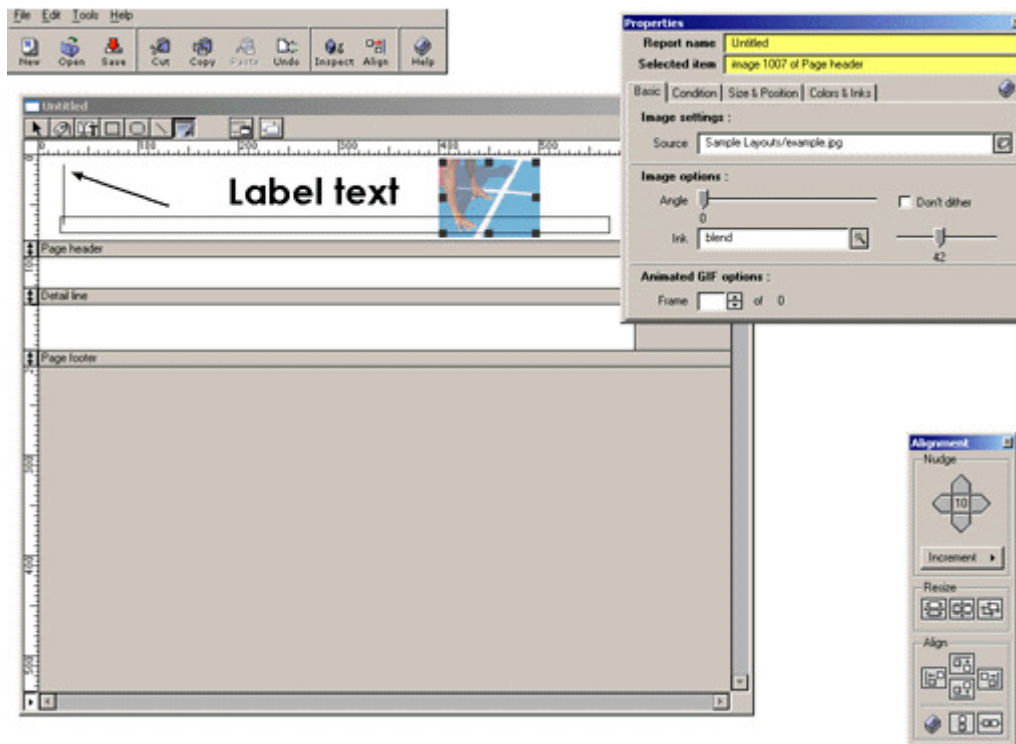
Once again, we can use the Inspector Palette to set the properties of the rectangle, such as the line thickness and the dashes pattern -- all familiar from *LiveCode*.

Now we'll add a few lines by using the *line tool* at the top of the window.

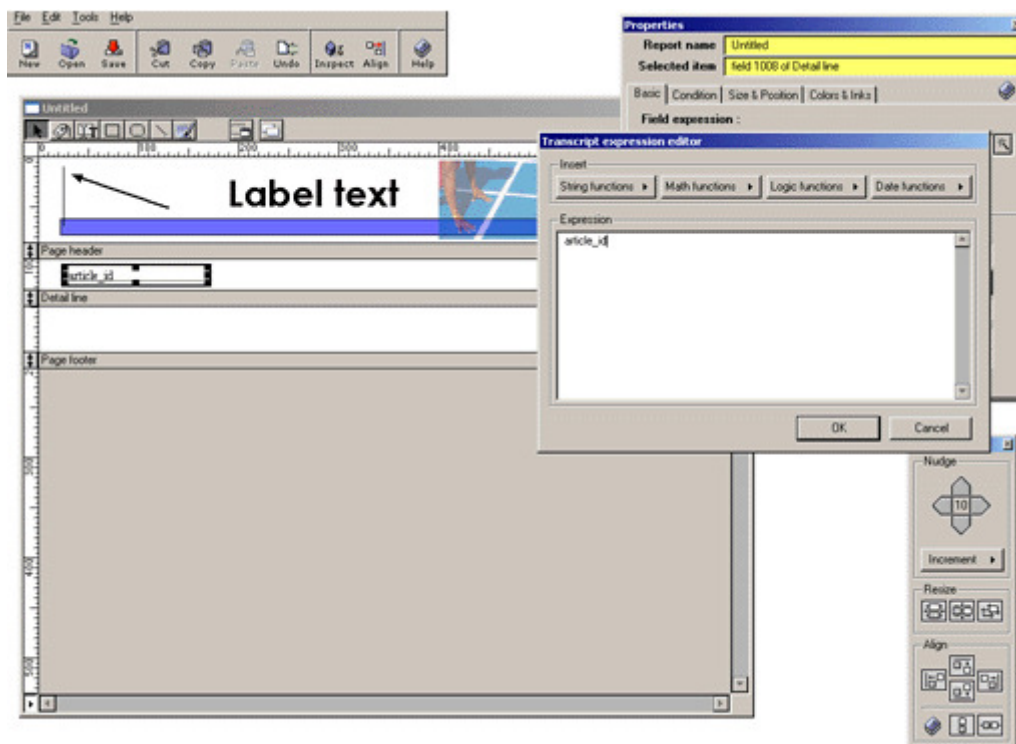


As you can see, the Inspector Palette allows us to set the properties of the lines : line thickness, arrow size, whether we want to show a start and end arrow,... -- if you can do it in *LiveCode*, you can do it in *Quartam Reports*.

Next we will add an image by using the *image tool* at the top of the window.



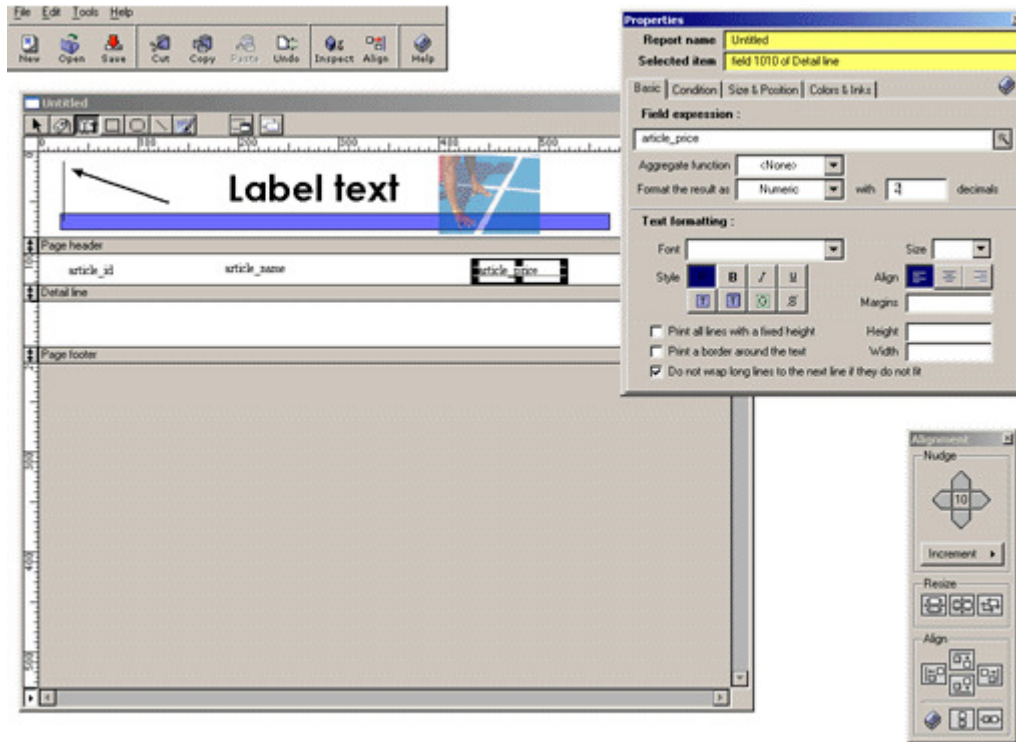
Indeed, using the Inspector Palette you can pick an image file and set more properties -- just like in *LiveCode* you can rotate images to a certain angle and even blend them with underlying items ! After all these decorative items, it is time we added some data fields by using the *data field tool* at the top of the window.





As you can see, **Quartam Reports** even provides a *LiveCode expression editor*, where you have convenient access to string, math, logic and date functions. It also checks the syntax to make sure you enter a correct expression.

But there's more : you can easily format the data as plain text, Unicode text, HTML text, RTF text, or even as numbers, dates, times or by means of an Printf()-expression -- with only a few steps, you can show the hexadecimal value of a number.

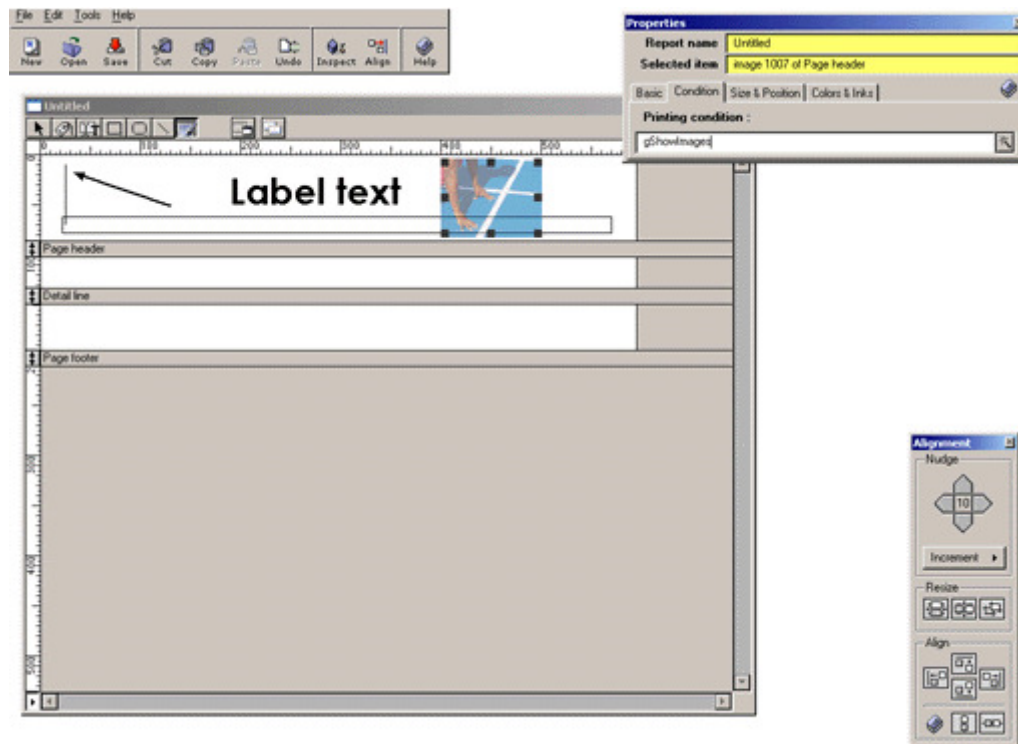


## Inspector palette

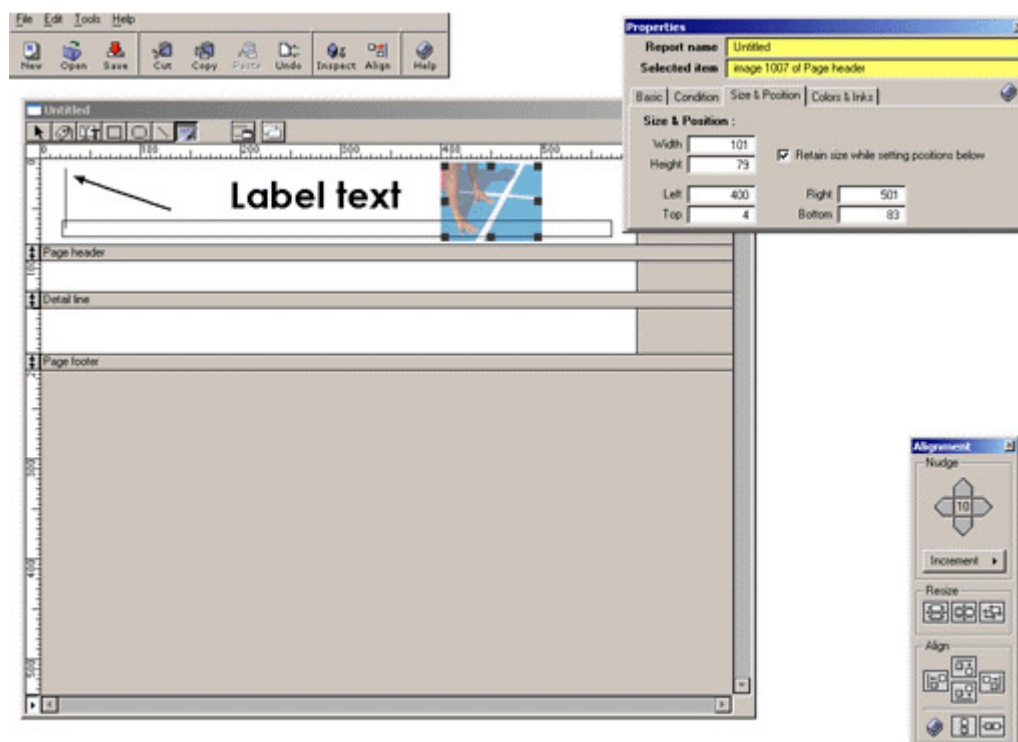
In the previous section of the **Quartam Reports** tour, you could see how the Inspector Palette allows you to set the properties of each item. We only showed you the Basic tab for some of the items, but have no doubt you wish to find out about the other tabs in the Inspector Palette.

The second tab hosts the *Condition* that has to be true in order for the item to be printed -- of course an empty condition means the item will always be printed. For example, you could test a variable to check if images should be included in your printout.

Another situation where these expressions come in handy, is if you want to highlight items that are out of the ordinary : place an arrow next to a figure, and only show it if the number is negative. Or place two data fields with the same expression, size and position, then change the color of one copy to red, and enter mutually exclusive printing conditions so that the number is printed in black if it's positive, or in red when it's negative. **Quartam Reports** puts the power where it belongs : in your hands.

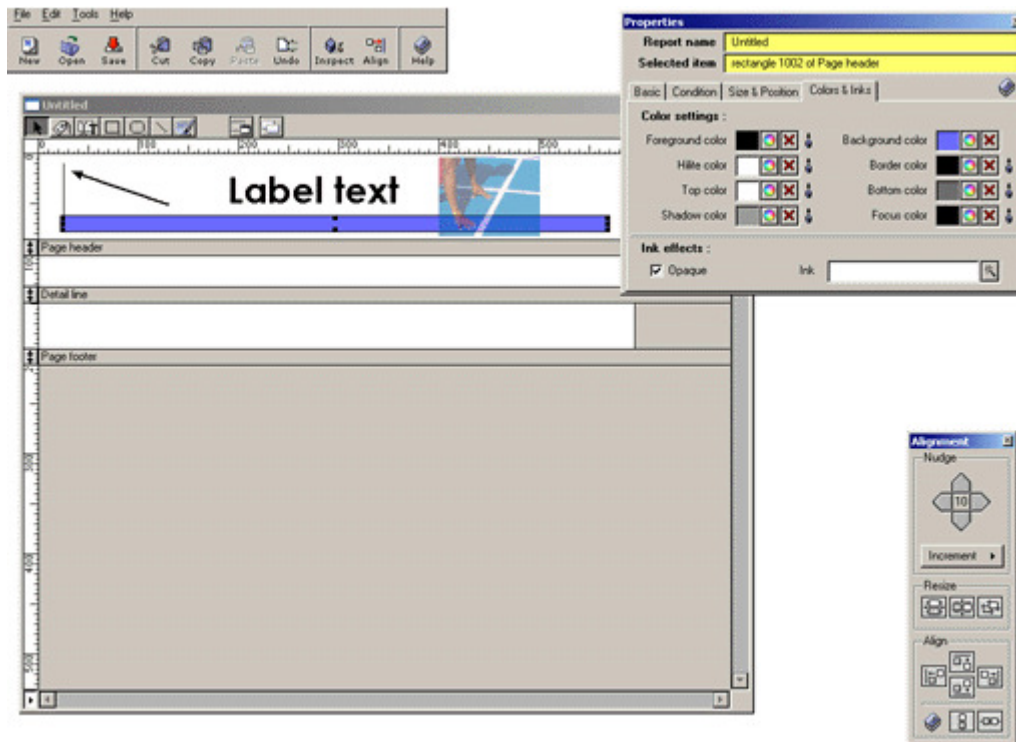


The next tab is appropriately named *Size & Position*. It makes it easy to change the size and position of your layout items down to the pixel level. However, you can also use the *arrow keys* on your keyboard to move the selected items around, or even to grow/shrink them by holding down the *option* or *alt-key*.



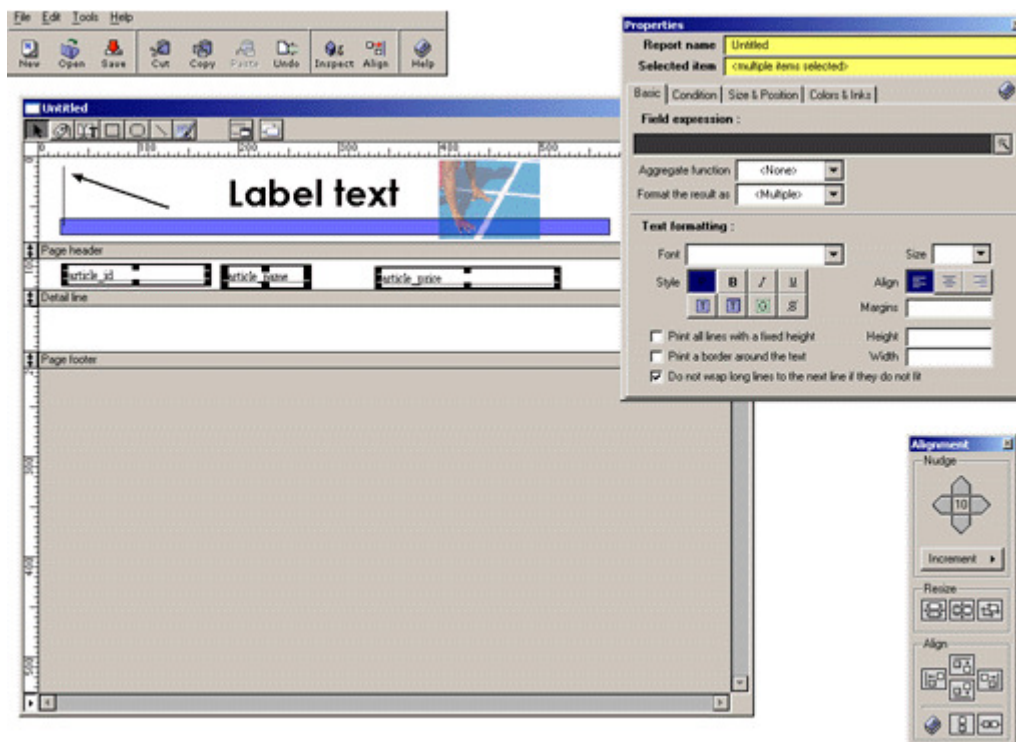
Of course you can also use the Alignment Palette to ensure that items are placed correctly, but we'll save that for the next section of the *Quartam Reports* tour.

The last tab is titled *Colors & Inks*, and this is the place where you can set the colors of your items : make your text red, fill your rectangles with your favorite shade of blue, or apply the many inks that *LiveCode* has to offer.




## Alignment palette

Earlier we added a few fields to the *detail band* of our report. Unfortunately some of the items were placed in a rather sloppy manner, and we need to make sure everything aligns well.



Fortunately, we have the Alignment Palette at our disposal for this type of mishap. As you can see, it is divided into three sections :

|   |  |
|---|--|
|  | <p><b>Nudge</b></p> <p>Select all the items you want to move, and click on the four arrows to change their positions.<br/>Use the Increment action menu to switch between moving 1-2-5-10-20 pixels at a time.</p> <hr/> <p><b>Resize</b></p> <p>Select all the items you want to resize, and make their widths, heights or rectangles equal to that of the first selected item.<br/>If you hold down the <i>option</i> or <i>alt-key</i>, all items will be resized to the widest, tallest or largest item in your selection.</p> <hr/> <p><b>Align</b></p> <p>Select all the items you want to align, and click the buttons to align them to the left, top, right, bottom, middle or center of the first selected item.<br/>If you hold down the <i>option</i> or <i>alt-key</i>, all items are aligned to the left-most, right-most, highest or lowest item, or centered horizontally or vertically in the <i>band</i>.</p> |
|---|--|

As you can see, we've taken every step to ensure that the Layout Builder is intuitive and easy to use, without compromising the power and flexibility of the Report Printing library. Now that you know that all those times spent with drawing tools was not a waste, let's move on to the next step: a strategy for building your reports.

## **Chapter 4. Report building strategy**

Maybe I ought to tell you a little about myself. I make business software for a living: applications for accounting and order-processing. In this capacity, I have to design and generate a lot of different reports, with lots of subtotals, averages and other statistical calculations.

When I design a report, I ask myself these five basic questions:

- 1) What do I want to print?
- 2) How do I want to organize it?
- 3) Do I need to track subtotals or similar calculated data?
- 4) Where do I get the data?
- 5) What about the font, size and style?

And I ask my customers to answer these very same questions.

### **What do I want to print?**

This question is designed to generate a general, textual description of the report. I'm not interested in detailed information, such as which data fields I want to see where. This is about knowing the goal.

A few examples of answers to this question might include:

- I need a list of my suppliers to tell them we're closing shop for a week, so they can organize their deliveries efficiently.
- Our quarterly magazine will be sent to our subscribers tomorrow, so I need to print address labels that I can stick onto the envelopes.
- Every month I send out reminders to those customers who are late on paying their bills – it should include an overview of the invoices and the total amount due.
- Once a week, I refill my inventory. I want a list that tells me, for each of my suppliers, which articles I need to stock up on.

As you can see, this already gives us an indication of the sort of data we'll be looking for, as well as a general idea of how it will be organized.

### **How do I want to organize it?**

Next I grab a sheet of paper and a pencil, and make a sketch of the report. If I'm sitting down with a customer, I also take out a ruler so I can draw straight lines.

At this point, I ask more questions, depending on the type of report.

If it's a label set, the obvious questions are:

- How many labels are one a single page?
  - o 8 rows of 3 columns
- Is there space above and below the labels?
- What do I need to print on each label?
  - o contact name and address
  - o article code, inventory location and price

If it's a report, the obvious questions are:

- Do I need a page header and footer?
- If so, what needs to be displayed in these parts?
  - o the report title

- the date and time of printing
  - the company logo and address information
- Is this a straightforward report or does the data need to be grouped into different sections?
- If it needs grouped, in what sort of order will the data be printed?
  - (I usually work from the detail to the outer group, but you can also work from the outer groups down to the detail sections)
  - invoice details, grouped per article, grouped per customer, grouped per sales rep
  - customer history, sorted by date, ascending
- What do I need to print in each section?
  - the invoice number, date, number of articles, unit price, discount, line total
  - the document number, document date, due date, amount, amount due
- Where do I place rectangles and other lines to group related items?
  - a rectangle around the payment information
  - a line at the end of each customer's set of invoices

After I've made this sketch, I start to think of the calculations needed to track subtotals and other statistical data.

## Do I need to track subtotals or similar data?

If you're simply printing out a customer file, you won't need any subtotals, but for business reports like an overview of the outstanding payments on your invoices, a grand total comes in handy. Or if you're analyzing trends in sales, you might want to track the average number of articles sold during a certain timeframe, such as the last 12 months, per month.

*Quartam Reports* offers an impressive array of data-aggregating and calculation functions:

- count
- sum
- average
- highest
- lowest
- median
- variance
- standard deviation

You can easily reset them to zero at the end of the report, the end of the page, or at the change of a data group, making it easy for you to track the total invoiced, the total paid, and the amount due.

## Where do I get the data?

This is where things can get complicated. Is all the data in one place, or do you need to combine it from multiple sources?

If the data is stored in stacks:

- is it all on the same card?
- is it spread over substacks or all in the same stack?
- if it is stored on multiple cards, which cards must be printed?
  - all cards, marked cards, unmarked cards, a certain range of cards
- does it need to be sorted before printing?

If the data is stored in a database:

- does data from multiple tables need to be combined?
- what columns are needed from each table?
- how is the data from these tables joined?
  - o primary and foreign keys
  - o inner joins, left-, right- and full- outer joins
- how should they be grouped and ordered?
  - o GROUP BY clause
  - o ORDER BY clause
- what records should be printed?
  - o WHERE clause
  - o HAVING clause

Most of these questions are answered already, but this is where you describe them in a more structured and formal manner, translating the user's needs into SQL-queries.

## **What about the font, size and style?**

This may sound like an afterthought, but it isn't. Data only becomes information through presentation. Make a list of the styles you want to apply to each bit of data: the font, the text size, whether a field needs to be printed bold or italic, etc.

But don't go overboard: make sure everything is legible and that the important information can be easily identified by increasing the font size or printing it in bold type.

I know that's a lot of questions to ask and answer, but preparation is half the work. If you have your sketch, it will be easier to use the Layout Builder in *Quartam Reports* to get everything positioned properly and looking as it should. Hopefully, the user's input leads to better organization.

## Chapter 5. Stacks and databases

Now that you have a strategy to tackle report printing, let's start with our first examples. In this chapter, you will learn how to print data from a stack as well as a database.

### Printing stacks

In the stack 'Report samples.rev', you will find a substack named 'qrsContacts' -- a simple contacts database. I know it doesn't look too shiny, but my graphics talents are limited to stick figures, so I prefer to use text labels for the buttons.

< screenshot of the qrsContacts stack

As you can see, it has a simple group of buttons for navigating the stack, creating and deleting cards, as well as marking them for printing and other purposes. The data is stored in a set of background fields, shared among the cards.

LiveCode stacks make for excellent embedded databases, where you can easily access and modify information without installing a hefty database server and learning SQL.

### The basic questions

Let's go through our questions one by one to prepare for the report building.

#### *What do I want to print?*

I want to print the data in my contacts stack so that I can add them as first pages to my paper files, so that I can easily see who the file is about as soon as I grab the binder.

#### *How do I want to organize it?*

- I only need one page per customer.
- The header should display a general text, the same on each page.
- The footer should display the date and time at which I printed it.
- I want to display all the data that is on the card, but I combine some of the fields.
  - o I'll print the first and last name as one line
  - o And I want to quickly see if I have marked this contact card for some reason
- It should look roughly the same as on the card, except stretch the remarks field to the bottom of the page.

#### *Do I need to track subtotals or similar data?*

Not in this example.

#### *Where do I get the data?*

All the data is stored on the cards, so the right data is already together in one spot – *Quartam Reports* has two features that makes this report easy:

- 1) the `qrtReports_PrintReportForStack` command
- 2) the ability to define data fields with everyday LiveCode expressions.

The first is a flexible command that lets you print a report, filling it with the data from one card at a time, with the option to print a specific range of cards. The latter means that our data fields can have simple expressions like

```
field "address"
```

or even combine data from different fields

```
field "firstName" && field "lastName"
```



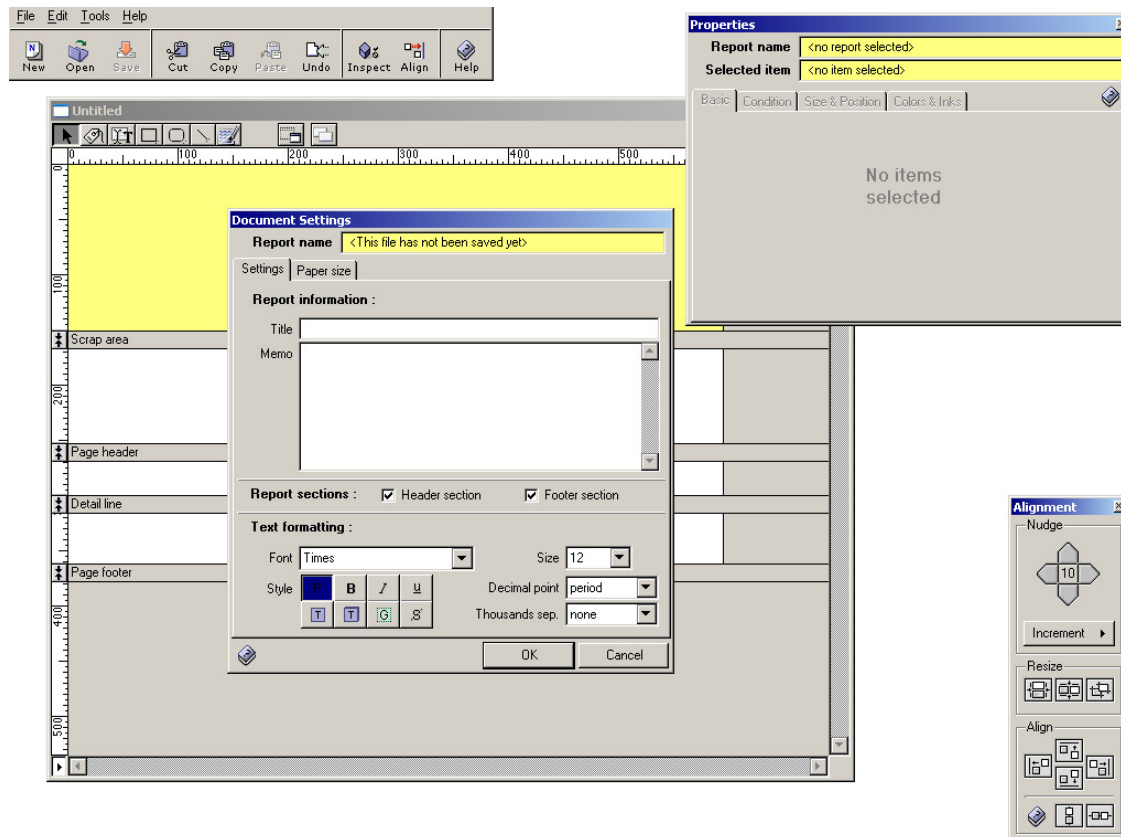
When you print the stack, all these expressions are evaluated within the context of the card that is being printed. Simple isn't it?

### *What about font, size and style?*

You can say all you want about tastes and colors, but I'm a 'Times New Roman' kind of guy. I will display the label data in bold, the rest in plain text. The title text in the header should be a little bigger. In the remarks field I often place styled text, and I want to retain that style information when printing.

## The report layout

Open the layout builder and click on the New button in the application toolbar.



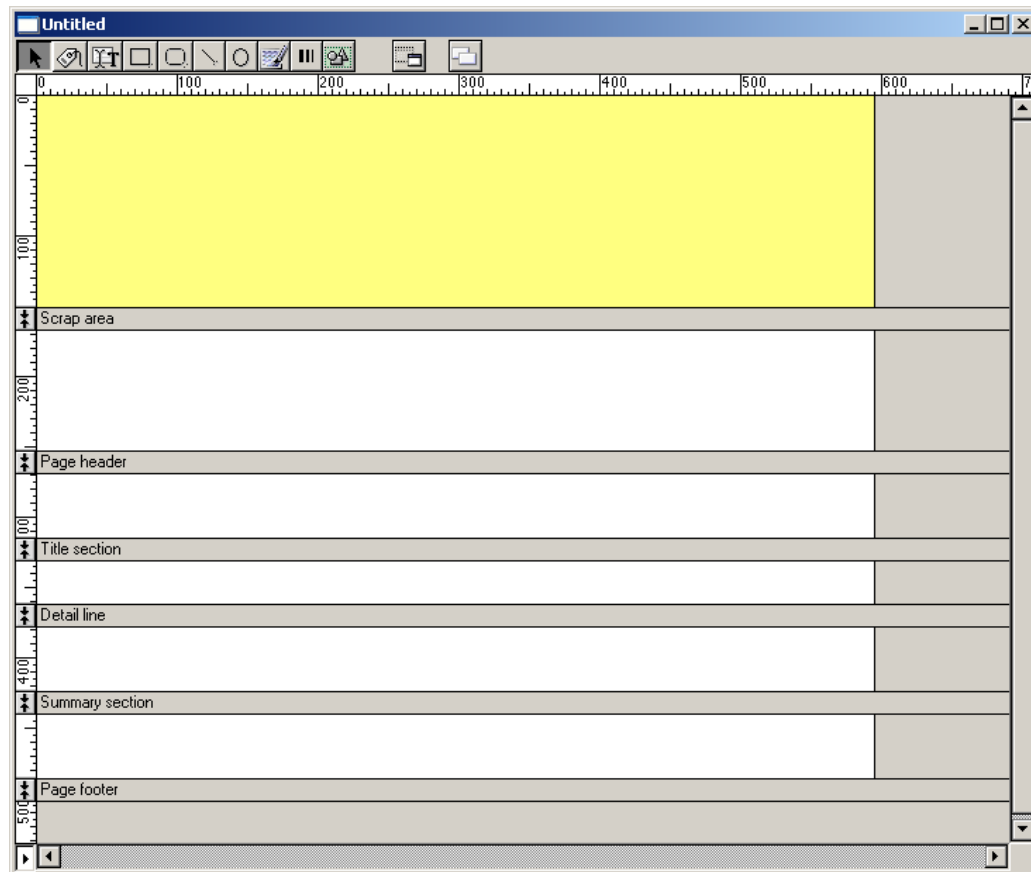
The Document settings dialog box opens immediately, so you can easily set the information at the start. The default paper size is A4 and header and footer sections are turned on, so all we need to do is tweak the default font and style settings.

Click the OK button to save the changes and go ahead with the layout of the report. If you need to go back to this dialog box at a later stage, you can always click the Document settings button in the editor window toolbar.

This empty layout editor window is ready to accept your orders. So let's add the different items.

As you can see, the layout editor window displays 6 bands that you can work in:

- the scrap area
- the page header
- the title section
- the detail band
- the summary section
- the page footer



The scrap area is where all the items are pasted. The items in this area will not be printed onto your report, so this can be a place to store items that you might want to display later on.

The page header and footer are where we place items that are printed on each page, at the top and the bottom of the page respectively.

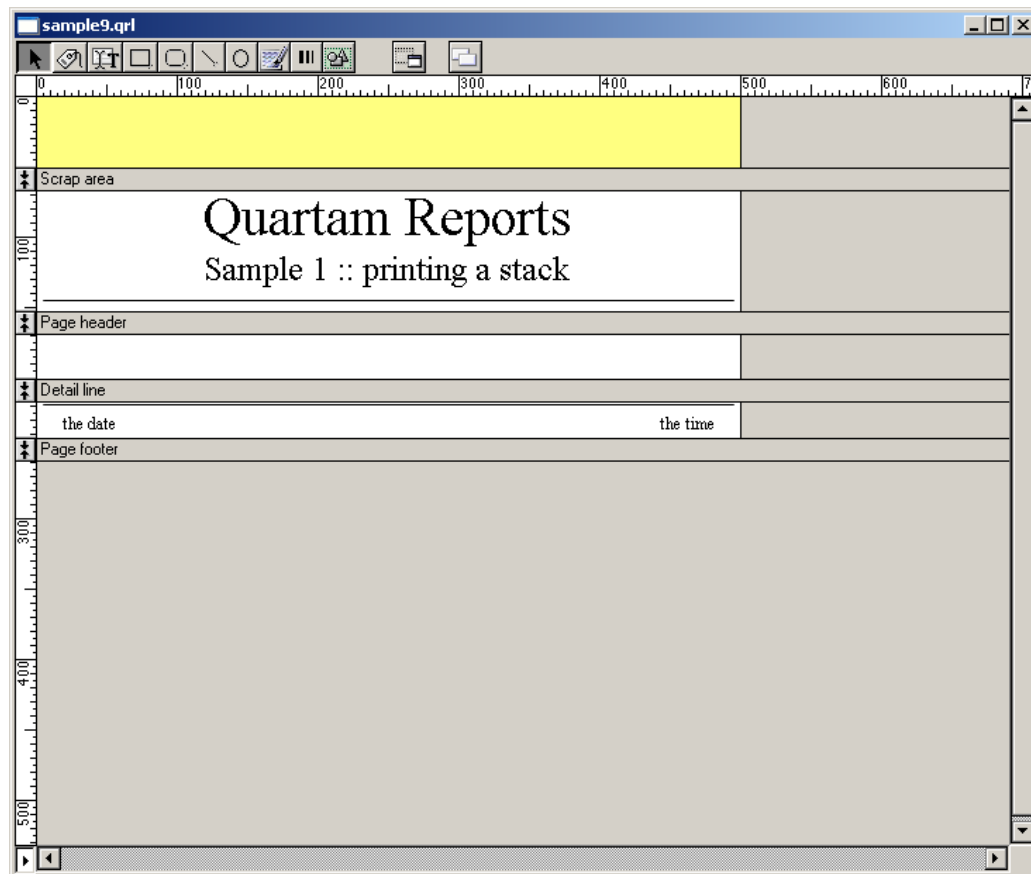
The detail band is where we place the items that we want to print for each contact.

As we don't need the Title and Summary sections for this example, click on the 'Document properties' button in the toolbar, and untick the 'Title section' and 'Summary section' checkboxes in the first tab of the dialog box.

We'll start by putting together the header and footer.

In the header, we add a label field, and use the inspector palette to set its content and change the font, size and style properties. Then we draw a line at the bottom of the band.

In the footer, we add two data fields, and use the inspector palette to set their expressions to 'the date' and 'the time' respectively. Then we draw a line at the top of the band.



Note how easy it is to see what data fields will contain, as the expression is shown like a label. Now we enlarge the detail band by dragging the vertical resizer icon on the left hand side of the band divider. Then we start adding more label and data fields, using the inspector palette to set their content and expressions.

For the “remarks” field we want to preserve the style as it is printed, so we set the expression to

```
the htmlText of field "remarks"
```

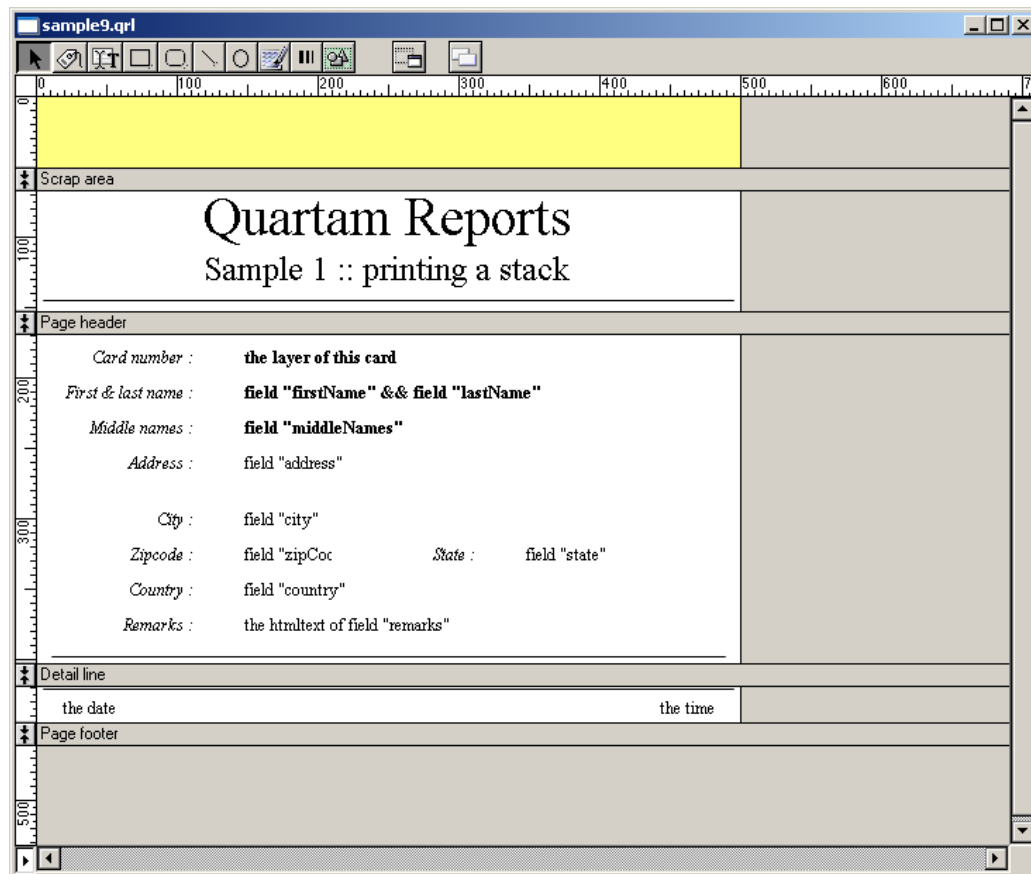
and set the Format option to ‘HTML Text’.

We also want to make sure the text can grow and that pages are automatically created, just in case it is a contact that we have a lot of comments about.

Go to the ‘Size & Position’ tab in the Inspector, and change the ‘Dynamic Resizing’ to the second option ‘Stretch the field vertically to fit the content’.

Now we also put a line underneath the “remarks” field – and that’s where we want it to stay: no matter how long the remarks text is, the line must be printed below the text..

Go to the ‘Size & Position’ tab in the Inspector, and change the ‘Relative position’ to the second option ‘Fixed distance from the bottom border of the band’.



Starting to look good, but we have a little more work to do: remember how I wanted to print a note if the card in the stack was marked or unmarked?

Let's add two label fields, one with the text "Marked" and another with the text "Unmarked". Then select the first data field and go to the second tab in the inspector palette, titled 'Condition' – here we can enter the printing condition: a LiveCode expressions that returns `true` if we want to print this item, and `false` if we don't want to print it.

In this case, we'll attach the following printing condition to the 'Marked' label field:

```
the mark of this card
```

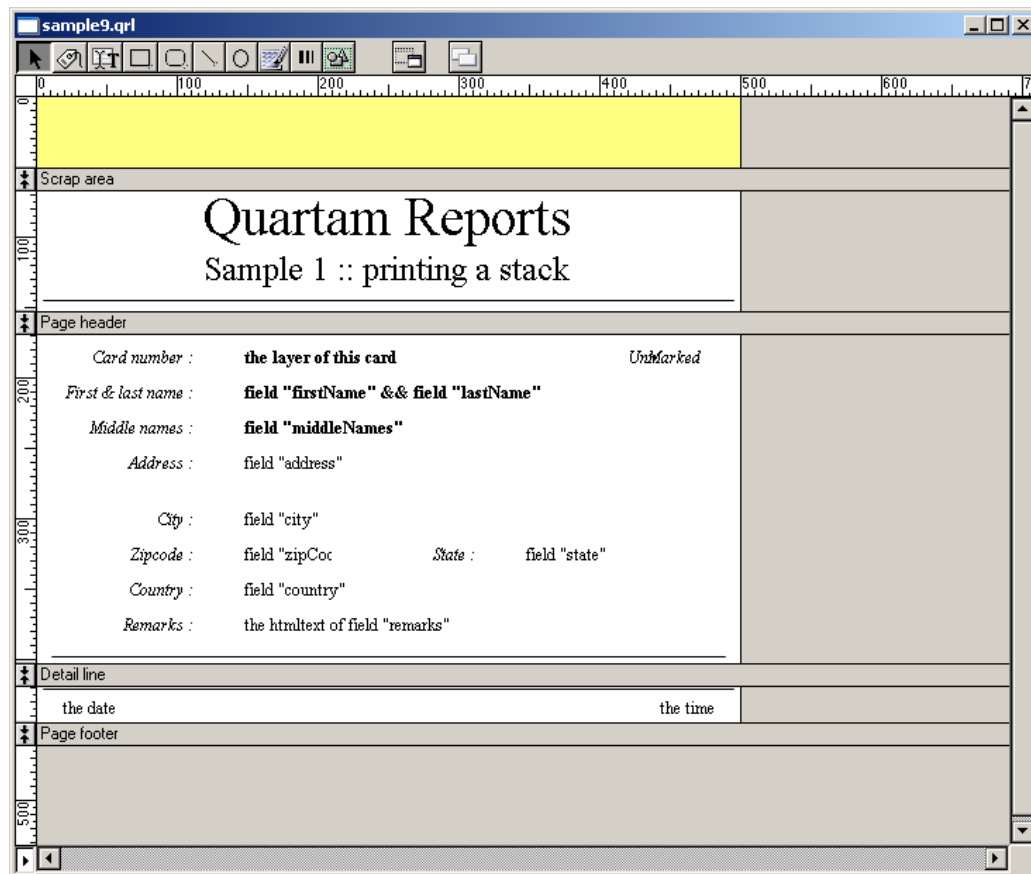
And attach the opposite printing condition to the 'Unmarked' label field:

```
not the mark of this card
```

Then we move them into the same spot, and the printing condition will make sure only either of these label fields gets printed.

**Tip:** This is an important trick to keep in mind, as you can use this to color a data field in *black* if it's a positive number or *red* if it's a negative number – simply make two data fields with the same data expression, and attach mutually exclusive printing conditions.

So, at long last, the layout is ready, and should look similar to this:



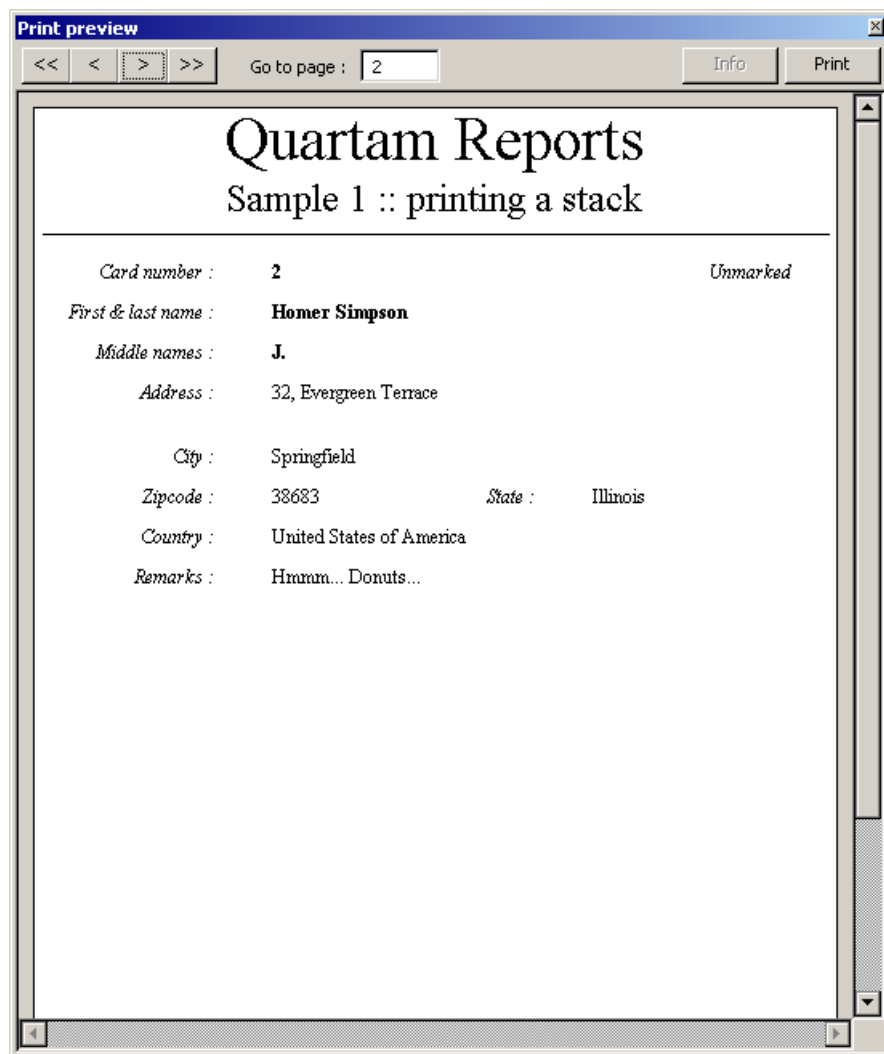
## The print script

Now that we have all the layout elements in place, it's time to add a Print button to our LiveCode project stack, right? First save the layout as "mystackreport.qrl", then return to LiveCode to make a print button.

In this case, we want to print all the cards in the stack, so add a button to your stack, and give it the following script to show a preview on screen:

```
on mouseUp
    -- define the variables
    local tLayoutFile, tStackName, tShowPreview
    -- set their values
    put "mystackreport.qrl" into tLayoutFile
    put "qrsContacts" into tStackName
    put true into tShowPreview
    -- print the report
    qrtReports_PrintReportForStack tLayoutFile, tStackName, tShowPreview
end mouseUp
```

Click the button, and you're rewarded with the following report preview:



## More ideas

The `qrtReports_PrintReportForStack` has a fourth parameter that lets you print a subset of your cards. This fourth parameter accepts the following input:

|          |  |
|----------|--|
| All      | All cards are printed (the default value)  |
| Marked   | Only the marked cards are printed          |
| Unmarked | Only the unmarked cards are printed        |
| [range]  | An MS-Word style range, such as "1-7,9,14" |

It would be nice to add these selections to your printing options, so you might want to take a good look at the script of the Stack printing example in the Report samples stack.

**Tip:** One way of letting your user select cards to print is to provide them with a list field of the contact names, and then using the *hilitedLines* property of the field as the print range.

## Printing database records

This section will show you how to print data from automated database queries.

### The basic questions

Let's go through our standard questions one by one to prepare for the report building.

#### *What do I want to print?*

I want to print a profit report where I can see how many articles were sold to which customers, at what price and what commission, with subtotals and grouped per sales representative.

Now this may seem like a complicated report, but I'll show you how to build the SQL query – just like LiveCode, it has a plain English-like syntax.

#### *How do I want to organize it?*

- The header should display a general text, the same on each page.
- The footer should display the date and time at which I printed it.
- I want to group the data per sales rep, then within the same sales rep I want the data grouped per customer, then within the same customer I want the data grouped per article
- In the detail lines of my invoices, I want to display:
  - o the invoice ID
  - o the date of the invoice
  - o the number of units sold
  - o the unit price
  - o the line total
  - o the commission percentage
  - o the commission amount
  - o the cost price
  - o the gross profit
- At the start of each article data group, I want to display:
  - o the article ID
  - o the article description.
- At the start of each customer data group, I want to display:
  - o the customer ID
  - o the customer name.
- At the start of each sales rep data group, I want to display:
  - o the sales rep ID
  - o the sales rep name.
- At the end of each group, I want to display some subtotals.

#### *Do I need to track subtotals or similar data?*

- At the end of each article data group, I want to display :
  - o the total number of units sold
  - o the average unit price
  - o the sum of the line totals
  - o the average commission percentage
  - o the sum of the commission amounts
  - o the total cost price
  - o the total gross profit
- At the end of each customer data group, I want to display:
  - o the sum of the line totals

- the average commission percentage
- the sum of the commission amounts
- the total cost price
- the total gross profit
- At the end of each sales rep data group, I want to display:
  - the sum of the line totals
  - the average commission percentage
  - the sum of the commission amounts
  - the total cost price
  - the total gross profit

### ***Where do I get the data?***

All the data is stored in 6 different tables in our database, so this requires a good SQL query to get the data onto paper – ***Quartam Reports*** has three features that makes this report less painful:

- 1) the `qrtReports_PrintReportForQuery` command
- 2) the ability to use the names of the database columns as though they were variables
- 3) the ability to define data fields with these variables in everyday LiveCode expressions.

The first is a flexible command that lets you print a report, filling it with the data from an automated database query one record at a time, with the option to print a specific range of records. The second and third mean that the data fields can refer to database columns using expressions like

```
det_costprice
```

or even combine data from different columns

```
det_amount * det_price
```

When you print the database query, all these expressions are evaluated within the context of the record that is being printed. Simple isn't it?

**Tip:** If your query combines the data from more than one field into a single column, you can use the `AS`-clause to turn that column into a valid LiveCode variable name.

**Example:** `SELECT article, amount * price AS linetotal FROM orderdetails`

So now we just have to figure out the SQL query. Please note that there are many excellent books and tutorials about SQL (Structured Query Language), and it is outside the scope of this manual to give you a crash course on database queries.

For this example, we need to combine the data from 5 tables in our database:

- salesreps
- customers
- invoices
- invdetails (short for invoice detail lines)
- articles

The relationship between these tables can be broken down as follows

- `salesreps.srep_ID = invoices.inv_salesrep` [links salesreps and invoices]
- `customers.cust_ID = invoices.inv_cust` [links customers and invoices]
- `invoices.inv_ID = invdetails.det_invID` [links invoices and invdetails]
- `articles.art_ID = invdetails.det_artID` [links articles and invdetails]

As we want to group the data per sales rep, within the sales rep per customer, and within the customer per article, and then sorted by the invoice ID, we get the following SQL query:



```
SELECT *
FROM salesreps, customers, invoices, invdetails, articles
WHERE srep_ID = inv_srep AND cust_ID = inv_cust AND
      inv_ID = det_invID AND det_artID = art_ID
ORDER BY srep_ID, cust_ID, art_ID, inv_ID
```

Note that I decided to get all the fields from all the tables by using an asterisk between SELECT and FROM – not the most efficient method if you only use a small subset of the fields in each table, but it will do for this example.

### *What about font, size and style?*

At the risk of being called conservative, I prefer this sort of report printed in Courier. As a monospaced font, it makes going through, comparing and verifying numeric data a lot easier. The exception is the report title, which should fit in the bigger scheme of the company style guide. While this doesn't make for very 'sexy' reports, I have yet to see an accounting application where reports make me tingly inside, anyway.

## The report layout

We're not going to spend time on creating a new layout. Instead, open the layout of the Query printing sample in the Report samples stack, and study how it is built. It should look like this:

The screenshot displays the 'sample8.qrl' application window. The interface includes a toolbar with various icons for navigation and editing, a ruler at the top, and a scrollable report area. The report content is as follows:

## Quartam Reports

### Sample 2 :: print a query

---

**Page header**

**Sales Rep :** srep I **Name :** srep name

**Group header : inv\_srep**

**Customer :** cust\_ID **Name :** cust\_name

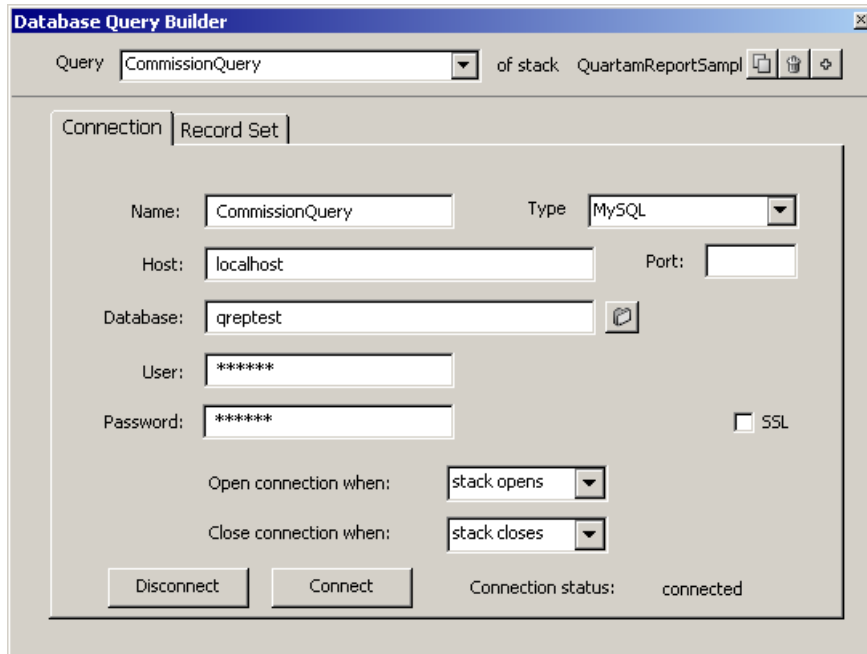
**Group header : inv\_cust**

| Article :                       | det_artID | Description : | art_descr      | Unit :                | art_unit                        |
|---------------------------------|-----------|---------------|----------------|-----------------------|---------------------------------|
| Invoice                         | Date      | Amount        | Price          | Total                 | comm %                          |
|                                 |           |               |                | Commission            | Costprice                       |
|                                 |           |               |                |                       | Profit                          |
| <b>Group header : det_artID</b> |           |               |                |                       |                                 |
| det_inv :                       | inv_date) | det_amount    | det_price      | mt * det_price        | xmper det_price / 100           |
|                                 |           |               |                | det_costprice         | ' - det_costprice               |
| <b>Detail line</b>              |           |               |                |                       |                                 |
|                                 |           | det_amount    | det_price      | mt * det_price        | xmper det_price / 100           |
|                                 |           |               |                | det_costprice         | ' - det_costprice               |
| <b>Group footer : det_artID</b> |           |               |                |                       |                                 |
| Customer :                      |           |               | mt * det_price | xmper det_price / 100 | det_costprice ' - det_costprice |
| <b>Group footer : inv_cust</b>  |           |               |                |                       |                                 |
| Sales Rep :                     |           |               | mt * det_price | xmper det_price / 100 | det_costprice ' - det_costprice |
| <b>Group footer : inv_srep</b>  |           |               |                |                       |                                 |
| the date                        |           |               |                |                       | the time                        |

## The automated query

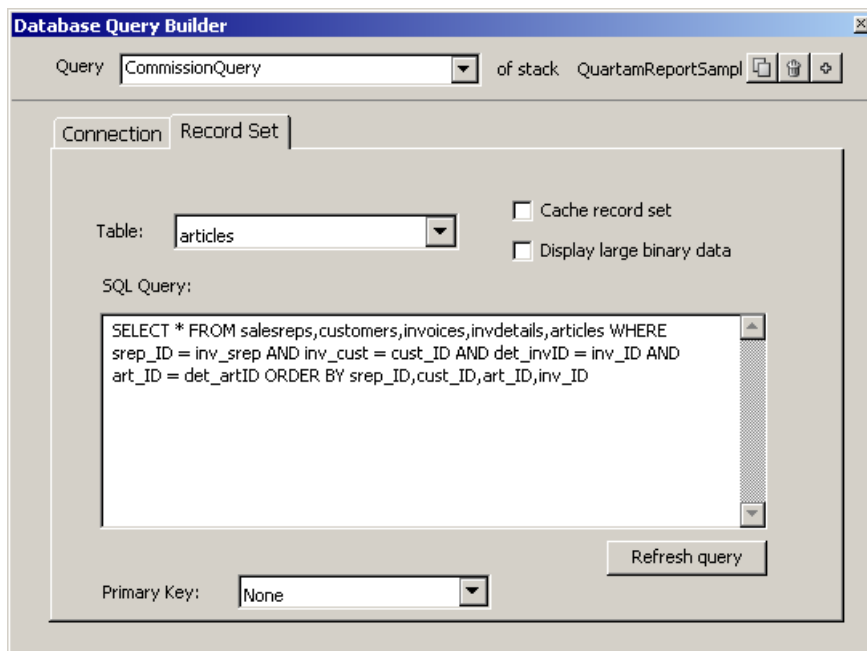
Of course we still have to define the automated database query in our stack. In LiveCode, go to the Tools menu and select the item 'Database query'.

Click on the '+' button to create a new query called "CommissionQuery" and fill in the connection parameters in the first tab – the screen will look similar to:



The screenshot shows the 'Database Query Builder' window with the 'Connection' tab selected. The 'Query' dropdown is set to 'CommissionQuery' and it is associated with the stack 'QuartamReportSampl'. The 'Record Set' tab is also visible. In the 'Connection' section, the 'Name' is 'CommissionQuery' and the 'Type' is 'MySQL'. The 'Host' is 'localhost' and the 'Port' is empty. The 'Database' is 'qreptest'. The 'User' and 'Password' fields are both masked with '\*\*\*\*\*'. There is an unchecked checkbox for 'SSL'. The 'Open connection when:' dropdown is set to 'stack opens' and the 'Close connection when:' dropdown is set to 'stack closes'. At the bottom, there are 'Disconnect' and 'Connect' buttons, and the 'Connection status:' is 'connected'.

In the second tab you enter the actual SQL query – the screen will look similar to:



The screenshot shows the 'Database Query Builder' window with the 'Record Set' tab selected. The 'Table' dropdown is set to 'articles'. There are two unchecked checkboxes: 'Cache record set' and 'Display large binary data'. The 'SQL Query' text area contains the following query: 

```
SELECT * FROM salesreps,customers,invoices,invdetails,articles WHERE  
srep_ID = inv_srep AND inv_cust = cust_ID AND det_invID = inv_ID AND  
art_ID = det_artID ORDER BY srep_ID,cust_ID,art_ID,inv_ID
```

 At the bottom, there is a 'Primary Key' dropdown set to 'None' and a 'Refresh query' button.

**Tip:** If you don't have a database server or don't want the overhead of running *MySQL* running on your local computer, you should consider *SQLite*. It ships for free with *LiveCode* 2.8.1 and higher, and just saves a single file on your local hard drive.

## The print script

Now that we have all the layout elements in place, it's time to add a Print button to our LiveCode project stack, right? First save the layout as "myqueryreport.qrl", then return to LiveCode to make a print button.

In this case, we want to print all the records in the query, so add a button to your stack, and give it the following script to show a preview on screen:

```
on mouseUp
  -- define the variables
  local tLayoutFile, tQueryName, tShowPreview

  -- set their values
  put "myqueryreport.qrl" into tLayoutFile
  put "CommissionQuery" into tQueryName
  put true into tShowPreview

  -- print the report
  qrtReports_PrintReportForQuery tLayoutFile, tQueryName, tShowPreview
end mouseUp
```

Click the button, and you're rewarded with the following report preview:

**Print preview**

<< < > >> Go to page : 2 Info Print

**Quartam Reports**  
**Sample 2 :: print a query**

|                          |           |                                      |         |          |         |            |            |          |
|--------------------------|-----------|--------------------------------------|---------|----------|---------|------------|------------|----------|
| F230006                  | 1-8-2003  | 12,250                               | 100,000 | 1 225,00 | 3,25    | 39,81      | 750,00     | 425,19   |
| F230007                  | 6-8-2003  | 12,250                               | 100,000 | 1 225,00 | 1,25    | 15,31      | 950,00     | 259,69   |
|                          |           | 24,500                               | 100,000 | 2 450,00 | 2,50    | 54,94      | 2 450,00   | 1 130,06 |
| Article : TRN-MS-PPPOINT |           | Description : MS-PowerPoint Training |         |          |         |            | Unit : hrs |          |
| Invoice                  | Date      | Amount                               | Price   | Total    | Comm. % | Commission | Costprice  | Profit   |
| F230001                  | 25-7-2003 | 12,250                               | 100,000 | 1 225,00 | 3,25    | 39,81      | 750,00     | 425,19   |
| F230006                  | 1-8-2003  | 12,250                               | 100,000 | 1 225,00 | 3,25    | 39,81      | 750,00     | 425,19   |
|                          |           | 24,500                               | 100,000 | 2 450,00 | 3,25    | 79,63      | 1 500,00   | 870,38   |
| Article : TRN-MS-WORD    |           | Description : MS-Word Training       |         |          |         |            | Unit : hrs |          |
| Invoice                  | Date      | Amount                               | Price   | Total    | Comm. % | Commission | Costprice  | Profit   |
| F230001                  | 25-7-2003 | 8,750                                | 100,000 | 875,00   | 3,25    | 28,44      | 612,50     | 234,06   |
| F230006                  | 1-8-2003  | 8,750                                | 100,000 | 875,00   | 3,25    | 28,44      | 612,50     | 234,06   |
| F230007                  | 6-8-2003  | 10,000                               | 100,000 | 1 000,00 | 1,25    | 12,50      | 698,75     | 288,75   |
|                          |           | 27,500                               | 100,000 | 2 750,00 | 2,50    | 69,38      | 1 323,75   | 756,88   |
| Article : TRN-POCKETPC   |           | Description : PocketPC/iFAQ Training |         |          |         |            | Unit : hrs |          |
| Invoice                  | Date      | Amount                               | Price   | Total    | Comm. % | Commission | Costprice  | Profit   |
| F230001                  | 25-7-2003 | 8,000                                | 100,000 | 800,00   | 2,25    | 18,00      | 700,00     | 82,00    |
|                          |           | 8,000                                | 100,000 | 800,00   | 2,25    | 18,00      | 700,00     | 82,00    |
| Customer :               |           | 13 975,00                            |         |          | 2,63    | 369,44     | 9 173,75   | 4 421,81 |
| Sales Rep :              |           | 13 975,00                            |         |          | 2,63    | 369,44     | 9 173,75   | 4 421,81 |

13-7-2005 7:40 PM

## More ideas

The `qrtReports_PrintReportForStack` has a fourth parameter that lets you print a subset of your cards. This fourth parameter accepts the following input:

|         |   |
|---------|---|
| All     | All records are printed (the default value) |
| [range] | An MS-Word style range, such as “1-7,9,14”  |

It would be nice to add these selections to your printing options, so you might want to take a good look at the script of the Query printing example in the Report samples stack.

**Tip:** One way of letting your user select records to print is to provide them with a list field of the records, and then using the *hilitedLines* property of the field as the print range.

## Chapter 6. Combining data sources using data brokers

In the days of the Information Age, data can come from many sources : a stack, local or remote databases, a static web-page somewhere on the internet or a dynamic LAMP-driven website, the result of a call to Web Services -- *LiveCode* is an excellent tool to obtain data from various sources, and prepare them for printing in complex, integrated reports.

*Quartam Reports (Professional Edition)* provides you with a simple yet powerful library to connect the dots : you can script your own *data broker* which fetches the information from its various sources and provides it to the report printing engine.

This data broker can be any control, though many developers will just place the script in the button that the user clicks in order to print the report.

### Skeleton of a data broker

Scripting a data broker seems like a complicated task, but it merely involves 3 required and 2 optional callback handlers. Below you find a skeleton script of a data broker – it assumes that you're using a button to print.

```
-- declare global variables
...

-- declare local variables
...

on mouseUp
    -- prepare everything for printing
    -- > this can mean: connecting to a database, opening a stack, calling a web service,...
    ...

    -- print the report
    qrtReports_PrintReport \
        "myreport.qrl", \      ## the layout of this report
        the long id of me, \   ## this button will handle the callbacks
        true                  ## show preview flag

    -- cleanup resources after we're done
    -- > this can mean: closing a database connection, deleting temporary files,...
    ...

end mouseUp

-- Quartam Reports data broker callbacks handlers and functions

function qrtReports_EndOfReport
    ## QR :: RETURN TRUE IF THE END OF THE REPORT HAS BEEN REACHED
    ...
end qrtReports_endOfReport
```

```

on qrtReports_MoveNextRecord
    ## QR :: PREPARE EVERYTHING FOR PRINTING THE NEXT RECORD
    ...
end qrtReports_moveNextRecord

on qrtReports_MoveNextGroup pGroupID,pGroupExpression
    ## QR :: PREPARE EVERYTHING FOR PRINTING THE NEXT GROUP
    ...
end qrtReports_moveNextGroup

on qrtReports_MoveNextPage pPageNumber
    ## QR :: PREPARE EVERYTHING FOR PRINTING THE NEXT PAGE
    ...
end qrtReports_moveNextPage

function qrtReports_ValueOfExpression pExpression
    ## QR :: EVALUATE THE EXPRESSION AND RETURN THE RESULT
    -- workaround for limitation in LiveCode engine
    put pExpression into tExpression
    replace "<QUOTE_REPLACEMENT>" with quote in tExpression
    return value(tExpression)
end qrtReports_valueOfExpression

```

From this skeleton script, you learn that you have to tell the report printing library which object will handle the callbacks (in this case, by passing ‘the long id of me’ as the second parameter), and how the script of this object has to implement the following callback handlers and functions:

- qrtReports\_EndOfReport function
- qrtReports\_MoveNextRecord handler
- qrtReports\_MoveNextGroup handler
- qrtReports\_MoveNextPage handler
- qrtReports\_ValueOfExpression function

Let’s go into a little more detail for each of these callback handlers and functions.

### **qrtReports\_EndOfReport function**

Implementing this function is mandatory. As the printing library assembles your report, it will periodically ask your data broker object if it has reached the end of the report. Your implementation should return true if all data has been printed, false otherwise.

Nothing too complicated, right? Nevertheless, make sure that you return true once you’ve reached the end of the report data, or you could send the printing process into an infinite loop!

### **qrtReports\_MoveNextRecord handler**

Implementing this handler is mandatory. As the printing library assembles your report, it will periodically signal to your data broker object that it has finished printing a detail band. Your implementation should take the opportunity to prepare for printing the next detail band. Of course, this is also a good time to increment any counter you have to keep track of the report progress – this makes it easier to tell if you’ve reached the end of the data.

## qrtReports\_MoveNextGroup handler

Implementing this handler is optional. As the printing library assembles your report, it will periodically signal to your data broker object that it has detected a change in the value of a data group expression.

The first parameter is the ID of the data group, which is the same as the group's ID in the Layout Builder. The second parameter is the expression of the data group. If you plan on fetching data separately to print in data group headers and footers, this is the best place to go and get it.

## qrtReports\_MoveNextPage handler

Implementing this handler is optional. As the printing library assembles your report, it will periodically signal to your data broker object that it has created a new page.

The only parameter is the page number, which makes it easier to print this information, as you can store it in a local variable.

**Tip:** This feature can be used to track separate page numbers per data group.

## qrtReports\_ValueOfExpression function

Implementing this function is mandatory. As the printing library assembles your report, it will frequently ask your data broker object to evaluate an expression and return the result.

This function will have access to all the local and global variables that you have declared at the top of your script, as well as all the functions that you have defined in this object or another object down *LiveCode's* message path.

Now that you have a basic idea of what a data broker script looks like, let's explore two examples.

## Multiple database queries

In the previous section we used an SQL-query which copied quite a bit of repeated information from the database to memory. Such an approach would use up a lot of resources for databases that contain vast amounts of data, not to mention hog the database as well as the network if you're running a quarterly report on the entire sales database.

In the sample script below, we make extra queries to the database "as needed" ; and because we only refresh related data when the data group changes, this is the most optimal design for large corporate databases.

```
-- define the local variables
local sConnectionID, sCursorID, sCurrentRecord, sRecordCount
local sDetail, sSalesRep, sCustomer, sArticle

on mouseUp
  -- first check if we're connected to the database or not
  put the uConnectionID of this stack into sConnectionID
  if sConnectionID is empty then
    answer information "Please connect to the database first."
    exit mouseUp
  end if
  -- if so, carry on by executing the report query
  put "SELECT * FROM invoices, invdetails" && \
```

```

        "WHERE invoices.inv_ID = invdetails.det_invID" && \
        "ORDER BY invoices.inv_srep, invoices.inv_cust," && \
        "invdetails.det_artID ;" \
    into tSQLQuery
    put revdb_query(sConnectionID, tSQLQuery) into sCursorID
    if sCursorID is not an integer then
        answer error sCursorID with "Cancel"
        exit mouseUp
    end if
    -- if it survived all these checks, move along and prepare
    -- all the data for the report
    put 1 into sCurrentRecord
    put empty into sDetail
    MyPrepNextDetail
    put empty into sSalesRep
    MyPrepNextSalesRep
    put empty into sCustomer
    MyPrepNextCustomer
    put empty into sArticle
    MyPrepNextArticle
    -- get ready to print
    local tLayoutFile, tDataBroker, tShowPreview
    put "mycomplexreport.qrl" into tLayoutFile
    put the long ID of me into tDataBroker
    put true into tShowPreview
    -- print the report
    qrtReports_PrintReport tLayoutFile, tDataBroker, tShowPreview
    -- don't forget to cleanup the cursor after we're done
    put revdb_closecursor(sCursorID) into sCursorID
    if sCursorID is not empty then
        answer error sCursorID with "Continue"
    end if
end mouseUp

-- NOTE :: these handlers map the database records onto arrays
on MyPrepNextDetail
    -- prepare the detail array
    -- by copying each column from the current record into the sDetail[] array
    put revdb_columnnames(sCursorID) into tColumnNames
    repeat for each item tColumnName in tColumnNames
        put revdb_columnbyname(sCursorID,tColumnName) into sDetail[tColumnName]
    end repeat
end MyPrepNextDetail

on MyPrepNextSalesRep
    -- prepare the sales rep array

```



```

-- by querying the database separately for the related sales rep record
put "SELECT * FROM salesreps WHERE srep_ID = :1" into tSQLQuery
put sDetail["inv_srep"] into tHolder
put revdb_query(sConnectionID, tSQLQuery, "tHolder") into tCursorID
if tCursorID is not an integer then
    answer error tCursorID with "Continue"
    exit MyPrepNextSalesRep
end if
-- now copy each column from the salesrep record into the sSalesRep[] array
put revdb_columnnames(tCursorID) into tColumnNames
repeat for each item tColumnName in tColumnNames
    put revdb_columnbyname(tCursorID,tColumnName) into sSalesRep[tColumnName]
end repeat
-- and cleanup after ourselves
put revdb_closecursor(tCursorID) into tCursorID
if tCursorID is not empty then
    answer error tCursorID with "Continue"
end if
end MyPrepNextSalesRep

on MyPrepNextCustomer
    -- prepare the customer array
    -- by querying the database separately for the related customer record
    put "SELECT * FROM customers WHERE cust_ID = :1" into tSQLQuery
    put sDetail["inv_cust"] into tHolder
    put revdb_query(sConnectionID, tSQLQuery, "tHolder") into tCursorID
    if tCursorID is not an integer then
        answer error tCursorID with "Continue"
        exit MyPrepNextCustomer
    end if
    put revdb_columnnames(tCursorID) into tColumnNames
    -- now copy each column from the customer record into the sCustomer[] array
    repeat for each item tColumnName in tColumnNames
        put revdb_columnbyname(tCursorID,tColumnName) into sCustomer[tColumnName]
    end repeat
    -- and cleanup after ourselves
    put revdb_closecursor(tCursorID) into tCursorID
    if tCursorID is not empty then
        answer error tCursorID with "Continue"
    end if
end MyPrepNextCustomer

on MyPrepNextArticle
    -- prepare the article array
    -- by querying the database separately for the related article record
    put "SELECT * FROM articles WHERE art_ID = :1" into tSQLQuery

```

```

put sDetail["det_artID"] into tHolder
put revdb_query(sConnectionID, tSQLQuery, "tHolder") into tCursorID
if tCursorID is not an integer then
    answer error tCursorID with "Continue"
    exit MyPrepNextArticle
end if
-- now copy each column from the article record into the sArticle[] array
put revdb_columnnames(tCursorID) into tColumnNames
repeat for each item tColumnName in tColumnNames
    put revdb_columnbyname(tCursorID,tColumnName) into sArticle[tColumnName]
end repeat
-- and cleanup after ourselves
put revdb_closecursor(tCursorID) into tCursorID
if tCursorID is not empty then
    answer error tCursorID with "Continue"
end if
end MyPrepNextArticle

-- NOTE :: Quartam Reports Callback Handlers and Functions

function qrtReports_EndOfReport
    ## QR :: RETURN TRUE IF THE END OF THE REPORT HAS BEEN REACHED
    -- we've reached the end if we've passed our last record
    return (sCurrentRecord > sRecordCount)
end qrtReports_endOfReport

on qrtReports_MoveNextRecord
    ## QR :: PREPARE EVERYTHING FOR PRINTING THE NEXT RECORD
    -- update our sCurrentRecord local variable
    add 1 to sCurrentRecord
    get revdb_movenext(sCursorID)
    -- and fill up the detail data
    MyPrepNextDetail
end qrtReports_moveNextRecord

on qrtReports_MoveNextGroup pGroupID,pGroupExpression
    ## QR :: PREPARE EVERYTHING FOR PRINTING THE NEXT GROUP
    switch pGroupID
    case 1 -- sales representative changed
        MyPrepNextSalesRep
        break
    case 2 -- customer changed
        MyPrepNextCustomer
        break
    case 3 -- article changed
        MyPrepNextArticle

```

```

        break
    end switch
end qrtReports_moveNextGroup

function qrtReports_ValueOfExpression pExpression
    ## QR :: EVALUATE THE EXPRESSION AND RETURN THE RESULT
    -- workaround for limitation in LiveCode engine
    put pExpression into tExpression
    replace "<QUOTE_REPLACEMENT>" with quote in tExpression
    return value(tExpression)
end qrtReports_valueOfExpression

-- NOTE :: the following function is called from within the report

function MyDBDateToDate pDBDate
    -- a simple function which converts dates in YYYYMMDD format
    -- to the LiveCode date format
    put char 1 to 4 of pDBDate & comma & char 5 to 6 of pDBDate & comma & \
        char 7 to 8 of pDBDate & ",0,0,0,0" into tDate
    convert tDate from dateItems to date
    return tDate
end MyDBDateToDate

```

In our report layout, we can now use expressions like

```
sArticle["description_french"]
```

I personally like using arrays to store the data, as it keeps everything nicely together.

One of the other interesting bits about this example, is that one of the data fields uses a function that is defined in the script of the data broker, to transform a date in YYYYMMDD format into a valid LiveCode date.

In fact, your data fields can call upon any function in library scripts – you are not limited to built-in LiveCode functions, but can expand at will!

## Internet pages

Until now, we've only shown you 'traditional' examples of what type of data can be printed. We want you to see just how big an advantage you have when you combine *LiveCode* and *Quartam Reports*. Below we've provided a sample script of a data broker that :

- connects to a local mysql database and reads article information
- connects to a cgi-script on a supplier website to find out about the number of days for delivery of each article.

```

-- define the local variables
local sConnectionID, sCursorID, sRecordCount, sCurrentRecord
local sRecordAsArray

```

```

on mouseUp

## PRINT MY COMPLEX REPORT
-- connect to the database and execute the query
put revdb_connect("mysql","localhost","mydatabase","user","password") \
    into sConnectionID
put revdb_query(sConnectionID, "SELECT * FROM articles") into sCursorID
put revdb_recordcount(sCursorID) into sRecordCount
put 1 into sCurrentRecord
-- prepare the data of the first record for printing
PrepareRecordData
-- get ready to print
local tLayoutFile, tDataBroker, tShowPreview
put "mycomplexreport.qrl" into tLayoutFile
put the long ID of me into tDataBroker
put true into tShowPreview
-- print the report
qrtReports_PrintReport tLayoutFile, tDataBroker, tShowPreview
-- clean up after printing
get revdb_closecursor(sCursorID)
get revdb_disconnect(sConnectionID)
end mouseUp

```

```

on PrepareRecordData

## PREPARE THE DATA FROM THE CURRENT RECORD FOR PRINTING
-- define the variables
local tColumnNames, tColumnName
-- copy the contents of the current record into the array
-- column per column
put revdb_columnnames(sCursorID) into tColumnNames
repeat for each item tColumnName in tColumnNames
    put revdb_columnbyname(sCursorID,tColumnName) into \
        sRecordAsArray[tColumnName]
end repeat
-- retrieve the data from the supplier website
put "http://www.example.com/cgi-bin/delivery_days.pl?article=" & \
    URLEncode(sRecordAsArray["article_id"]) into tRemoteURL
put URL tRemoteURL into sRecordAsArray["delivery_days"]
end PrepareRecordData

```

-- NOTE :: Quartam Reports Callback Handlers and Functions

```

function qrtReports_EndOfReport

## QR :: RETURN TRUE IF THE END OF THE REPORT HAS BEEN REACHED
-- we've reached the end if we've passed our last record
return (sCurrentRecord > sRecordCount)
end qrtReports_EndOfReport

```

```

on qrtReports_MoveNextRecord
    ## QR :: PREPARE EVERYTHING FOR PRINTING THE NEXT RECORD
    -- update the sCurrentRecord local variable
    add 1 to sCurrentRecord
    -- move to the next record in the cursor
    get revdb_movenext(sCursorID)
    -- and prepare the record data
    PrepareRecordData
end qrtReports_MoveNextRecord

function qrtReports_ValueOfExpression pExpression
    ## QR :: EVALUATE THE EXPRESSION AND RETURN THE RESULT
    -- workaround for limitation in LiveCode engine
    local tExpression
    put pExpression into tExpression
    replace "<QUOTE_REPLACEMENT>" with quote in tExpression
    return value(tExpression)
end qrtReports_ValueOfExpression

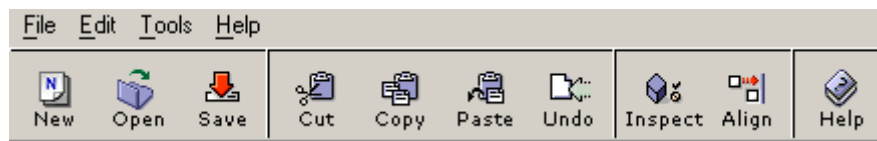
```

As you can see from the above examples, a few lines of script allow you to dynamically integrate data from multiple sources into a single report. With *LiveCode* and *Quartam Reports* the possibilities are endless !



## **Part Four: Layout Builder**

## Chapter 7. The application menubar



### The File Menu

The file menu is the place to be when creating, opening, saving and closing layout files.

#### New

The New menu item has two sub-items to create either a report or a label set layout.

In either case, the application creates a new layout editor window, and opens the 'Document settings' dialog box. There you can easily describe basic report information and paper size.

#### Open

The Open menu item displays a standard system 'Open file' dialog box from which you can browse and open existing report and label set layout files

#### Close

The Close menu item will close the active layout editor window. If you have not saved the layout before, or have made changes since it was last saved, you will be asked if you want to save the layout.

#### Save

The Save menu item lets you save the active layout editor window. If you have not saved the layout before, it opens a standard system 'Save file' dialog box.

#### Save As

The Save As... menu item lets you save a copy of the active layout editor window, and opens a standard system 'Save file' dialog box.

#### Revert

The Revert menu item lets you revert your layout to when it was last saved. You will be warned that doing so will discard all the changes made. This action cannot be undone.

#### Quit/Exit

The Quit (or Exit, depending on the operating system) menu item is the preferred way to close the application. If any of the open layout editor windows has not been saved in its current state, you will be asked to save these files.

Under MacOS X, the menu item 'Quit' can be found in the Application menu.

### The Edit Menu

The Edit menu is where you can find the usual collection of item and text manipulation features: Undo/Redo, Cut, Copy, Paste, Clear and Duplicate.

#### Undo / Redo

The Undo/Redo menu item will undo or redo the last action. This includes moving items back to their previous location, reverting changes to item properties, etc.



## **Cut**

The Cut menu item will cut the currently selected item(s) in the active layout editor window. The item(s) will be removed from the layout and placed into the clipboard.

## **Copy**

The Copy menu item will copy the currently selected items in the active layout editor window. A copy of the items will be placed into the clipboard.

## **Paste**

The Paste menu item will paste the item(s) in the clipboard into the scrap area of the active layout editor window. From there you can move them to the desired location on your layout.

## **Clear**

The Clear menu item will delete the currently selected items in the active layout editor window. The items will be removed from the layout but will not be placed into the clipboard. Use this item with caution.

## **Duplicate**

The Duplicate menu item will make an in-place copy of the currently selected item(s) in the active layout editor window. The items will be placed in the same spot as the original items. This action does not affect the clipboard.

## **Preferences**

The Preferences menu item will display the 'Preferences' dialog box.

Under MacOS X, the menu item 'Preferences' can be found in the Application menu.

## **The Tools Menu**

The Tools menu allows you to toggle the visibility of the Properties and Alignment palettes, as well as the application backdrop.

### **Properties palette**

The Properties palette menu item will open the Properties palette if it is not open or close if it is open.

Toggle the visibility of the palette to save screen real estate while viewing larger layouts.

### **Alignment palette**

The Alignment palette menu item will open the Alignment palette if it is not open or close it if it is open.

Toggle the visibility of the palette to save screen real estate while viewing larger layouts..

### **Backdrop**

The Backdrop menu item will display a backdrop behind all layout editor windows, if it is not already displayed. If the backdrop was already displayed, it will be hidden.

Toggle the visibility of the palette to reduce screen clutter from other applications.

## The Help Menu

The Help menu provides you with easy access to the online help, as well as samples for review, and license and version information.

### Layout Builder Help

The Layout Builder Help menu item opens the online help. It is currently available in HTML-format only, and is a copy of the information found in Part 3 of this manual.

### Review samples

The Review samples menu item opens the 'Report Samples' stack, which contains a number of example layouts and scripts that showcase the capabilities of *Quartam Reports*.

You can also open this stack in *LiveCode*, where you can look at the scripts.

### License information

The License information menu item will display more information about the license. From there, you can easily visit our online store, unlock your license code and print the license agreement.

### About Quartam Reports

The About Quartam Reports menu item will display more information about *Quartam Reports*. In particular, it contains the copyright and license holder information, as well as the version number. Under MacOS X, the menu item 'About Quartam Reports' can be found in the Application menu.

## Chapter 8: The application toolbar

The application toolbar provides quick access to commonly used features.



### New

The New toolbar button creates a new report layout, and opens the Document settings dialog box where you can easily set the report information and paper size.

If you want to create a label set rather than a paged report, use the 'File' menu's "New" item. It has sub-items to create either a report or label set layout.

Clicking this button is the equivalent of opening the 'File' menu and selecting the menu item 'New' and then the sub-item 'Report'.

### Open

The Open toolbar button displays a standard system Open file dialog box. There you can easily browse and open existing layout files for reports and label sets.

Clicking this button is the equivalent of opening the 'File' menu and selecting the menu item 'Open'.

### Save

The Save toolbar buttons lets you save the active layout editor window. If you have not saved the layout before, it opens a standard system 'Save file' dialog box.

Clicking this button is the equivalent of opening the 'File' menu and selecting the menu item 'Save'.

### Cut

The Cut toolbar button will cut the currently selected items in the active layout editor window. The items will be removed from the layout and placed into the clipboard.

Clicking this button is the equivalent of opening the 'Edit' menu and selecting the menu item 'Cut'.

### Copy

The Copy toolbar button will copy the currently selected items in the active layout editor window. A copy of the items will be placed into the clipboard.

Clicking this button is the equivalent of opening the 'Edit' menu and selecting the menu item 'Copy'.

### Paste

The Paste toolbar button will paste the items in the clipboard into the scrap area of the active layout editor window. From there you can move it to the desired location on your layout.

Clicking this button is the equivalent of opening the 'Edit' menu and selecting the menu item 'Paste'.

## **Undo**

The Undo toolbar button will undo the last action. This includes moving back items to their previous location, reverting changes to item properties, etc.

Clicking this button is the equivalent of opening the 'Edit' menu and selecting the menu item 'Undo / Redo'.

## **Inspect**

The Inspect toolbar button will open the Properties palette if it is not open or closes it if it is open

Toggle the visibility of the palette to save screen real estate while viewing larger layouts.

## **Align**

The Align toolbar button will open the 'Alignment' palette if it is not open or close it if it is open.

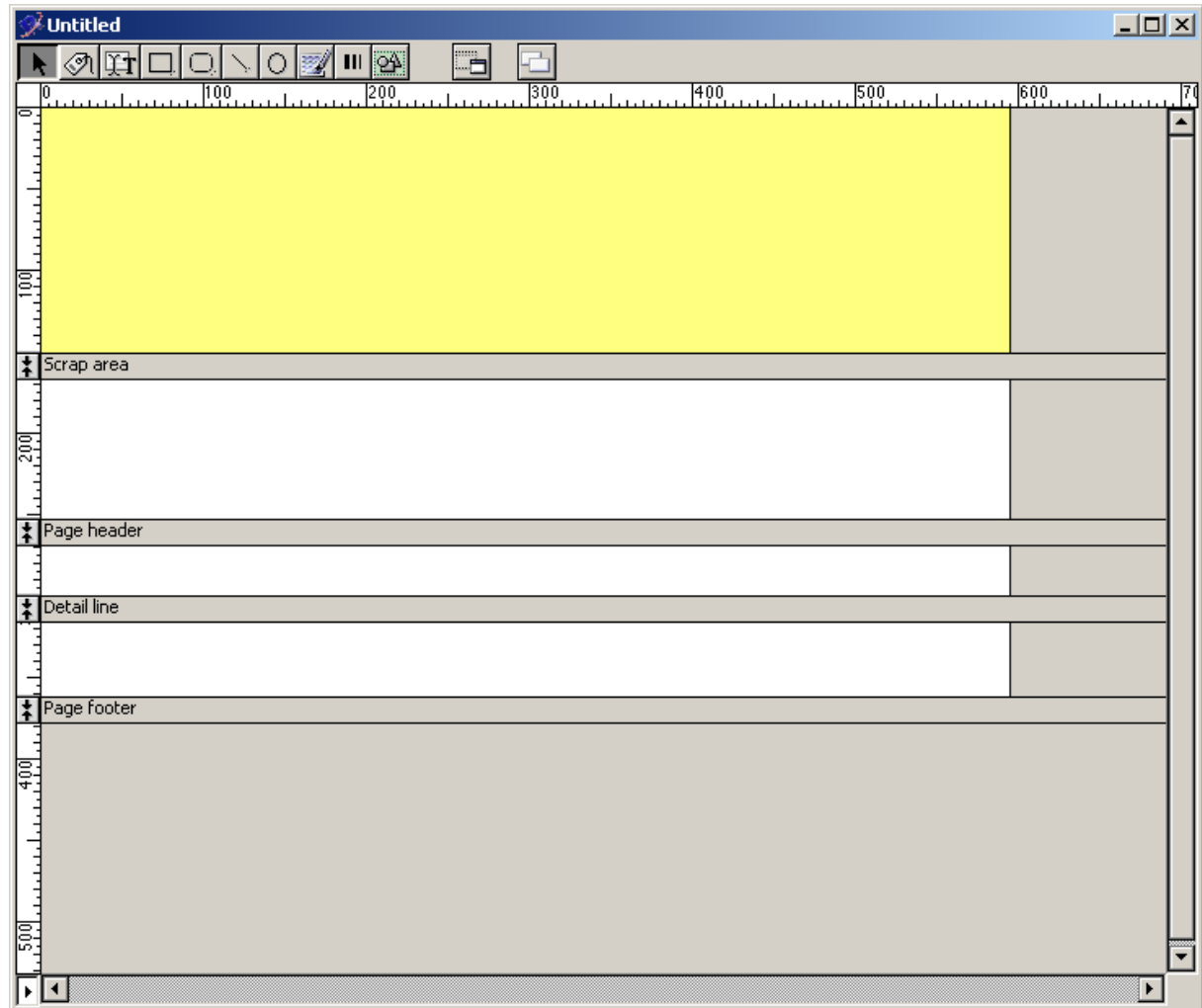
Toggle the visibility of the palette to save screen real estate while viewing larger layouts..

## **Help**

The Help toolbar button opens the online help system. Similar buttons are available in all the dialog boxes and palettes of the layout builder.

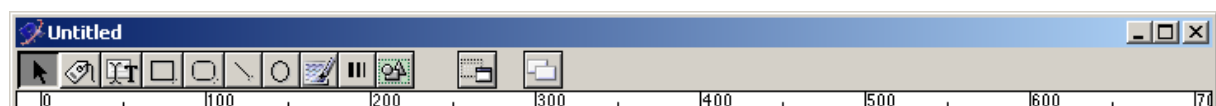
## Chapter 9: Layout Editor windows

Most of your layout editor windows will contain a few more items than the one below, but this is what it typically looks like when you're at the start of building a new report layout.



Every layout editor window has its own toolbar, document settings and data group overview.

### Toolbar



### Pointer

Use the pointer tool to select existing items, move them, etc.

Hold down the shift key while clicking to select multiple items. (Note that you can only select multiple items within the same layout band.)

As an alternative method to select multiple items in a band, click and drag to draw a virtual box around the desired items.

## **Label field**

Use the Label field tool to add label fields to your layout. Label fields display static text such as report titles, labels that tell the user what is being displayed in a field, etc.

You can set the actual text as well as the font information using the Propertiespalette. If you do not set font information, the corresponding font information settings for the document will be applied.

## **Data field**

Use the Data field tool to add data fields to your layout. Data fields are the heart of your reports, holding the variable information that you want to display.

Once you've defined a data field, you can open the Properties inspector on the field to define a LiveCode expression (if you wish to use one) as well as the formatting options and font information. If you do not set font information, the corresponding font settings for the document will be applied.

## **Rectangle**

Use the Rectangle tool to draw rectangles on your layout. Rectangles provide an easy way to visually group related data or draw attention to important information.

You can set the line thickness, color and other properties of a rectangle using the Properties palette.

## **Round rectangle**

Use the Round rectangle tool to add rectangles to your layout. Round rectangles provide an easy way to visually group related data or draw attention to important information.

You can set the line thickness, corner radius and other properties using the Properties palette.

## **Line**

Use the Line tool to add straight lines to your layout. Lines provide an easy way to visually delineate sets of related data or draw attention to important information.

You can set the line thickness, color and other properties using the Propertiespalette.

## **Oval**

Use the Oval tool to draw ovals on your layout. Ovals provide an easy way to visually draw attention to information that requires more investigation.

You can set the line thickness, color and other properties of an oval using the Properties palette.

## **Image**

Use the Image tool to add images to your layout. Static images provide an easy way to display a company logo on your reports.

You can set the file name, inks and other properties using the Properties palette.

## **Barcode (*Professional Edition*)**

Use the Barcode tool to add barcodes to your layout. Barcodes are often used in retail and warehouse information, speeding up the registration of data.

You can set the barcode type, inks and other properties using the Properties palette.

## 2D Chart (*Professional Edition*)

Use the Chart tool to add two-dimensional charts to your layout. Charts help to visualize trends, in turn aiding the decision making process.

You can set the barcode type, inks and other properties using the Properties palette.

## Document settings

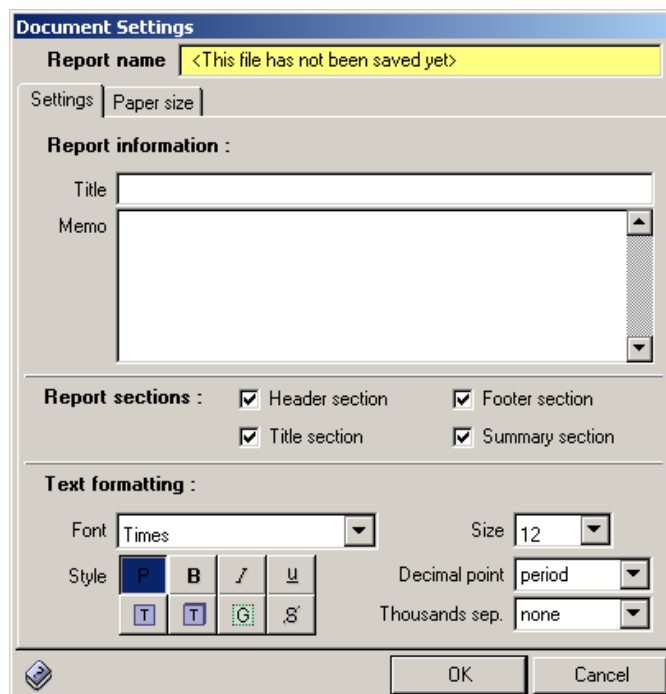
To the right of the tool buttons, you will find the Document settings button.

The Document settings dialog box is divided in two parts:

- General settings
- Paper size

You can navigate by using the tabbed button at the top.

### General settings



The screenshot shows the 'Document Settings' dialog box with the 'General' tab selected. The 'Report name' field at the top contains the text '<This file has not been saved yet>'. Below this, there are two tabs: 'Settings' and 'Paper size', with 'Settings' being the active tab. The 'Report information' section contains a 'Title' text box and a 'Memo' text area. The 'Report sections' section has four checkboxes, all of which are checked: 'Header section', 'Footer section', 'Title section', and 'Summary section'. The 'Text formatting' section includes a 'Font' dropdown menu set to 'Times', a 'Size' dropdown menu set to '12', and a 'Style' section with buttons for 'P' (Paragraph), 'B' (Bold), 'I' (Italic), 'U' (Underline), 'T' (Text), 'G' (Color), and 'S' (Size). To the right of these buttons are two dropdown menus: 'Decimal point' set to 'period' and 'Thousands sep.' set to 'none'. At the bottom of the dialog are 'OK' and 'Cancel' buttons.

Here you can set the report title as well as some memo text. You can also toggle the header and footer sections here, and set the default text formatting options, which you can override separately for each label or data field item in your report layout.

Last but not least, it allows you to define the decimal point and thousands separator for number formatting.

## Paper size

The image shows a 'Document Settings' dialog box with a blue title bar. It has two tabs: 'Settings' and 'Paper size', with 'Paper size' currently selected. At the top, there's a yellow box for 'Report name' containing the text '<This file has not been saved yet>'. Below this, there's a 'Pick paper size from :' section with a 'Predefined sizes' button and a 'Page setup...' button. The 'Paper size :' section features a diagram of a paper sheet with 'Height' set to 840 and 'Width' set to 595. To the right of this is a yellow box with an 'IMPORTANT NOTE' stating that the report will be scaled to fit the chosen paper size while preserving margins. Below the paper size section are two groups of input fields: 'Label settings :' with 'Height' and 'Width' fields, and 'Gutters between labels :' with 'Row' and 'Column' fields. The 'Print margins :' section at the bottom has four fields: 'Left' (72), 'Right' (72), 'Top' (72), and 'Bottom' (72). At the very bottom are 'OK' and 'Cancel' buttons.

Here you define the expected paper size, label size, gutters and print margins.

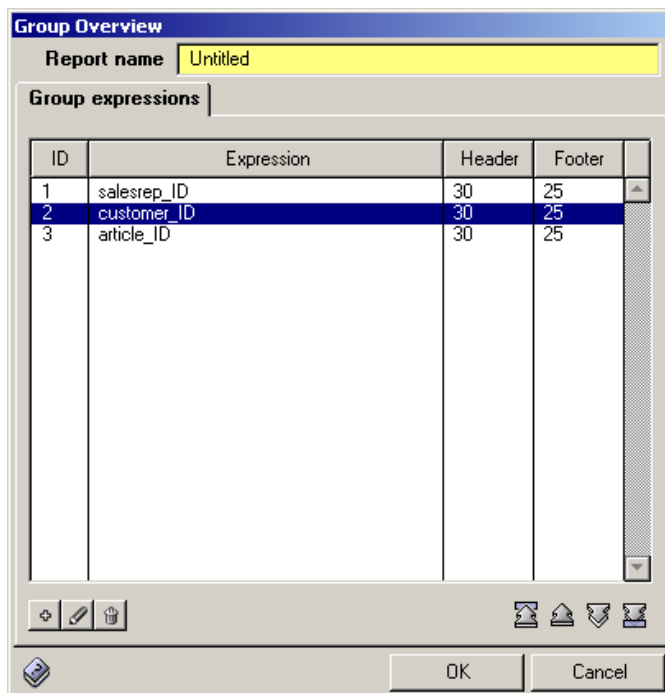
## Data groups

Quartam Reports Professional edition allows you to use *data groups*: they define sets or related data and make it easier to track statistical data, such as the total number of sold goods or average profit margins per customer.

Data in your reports is grouped in a hierarchical manner, where one group is nested within another group. Each data group can have its own header and footer band for you to display this information above and below the information on the deeper levels.



## Group overview



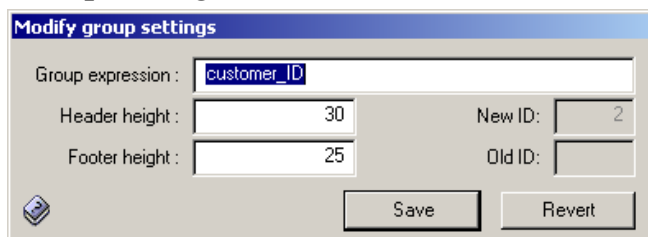
The 'Group overview' dialog box is where you create, modify, delete and rearrange data groups.

Use the '+' button on the left to create a new data group, the 'pencil' button to edit the selected group, and the 'trashcan' button to delete the currently selected group.

Use the arrow keys on the bottom right to rearrange the hierarchy of your data groups.

No changes are actually made to the report layout until you click the 'OK' button.

## Group settings

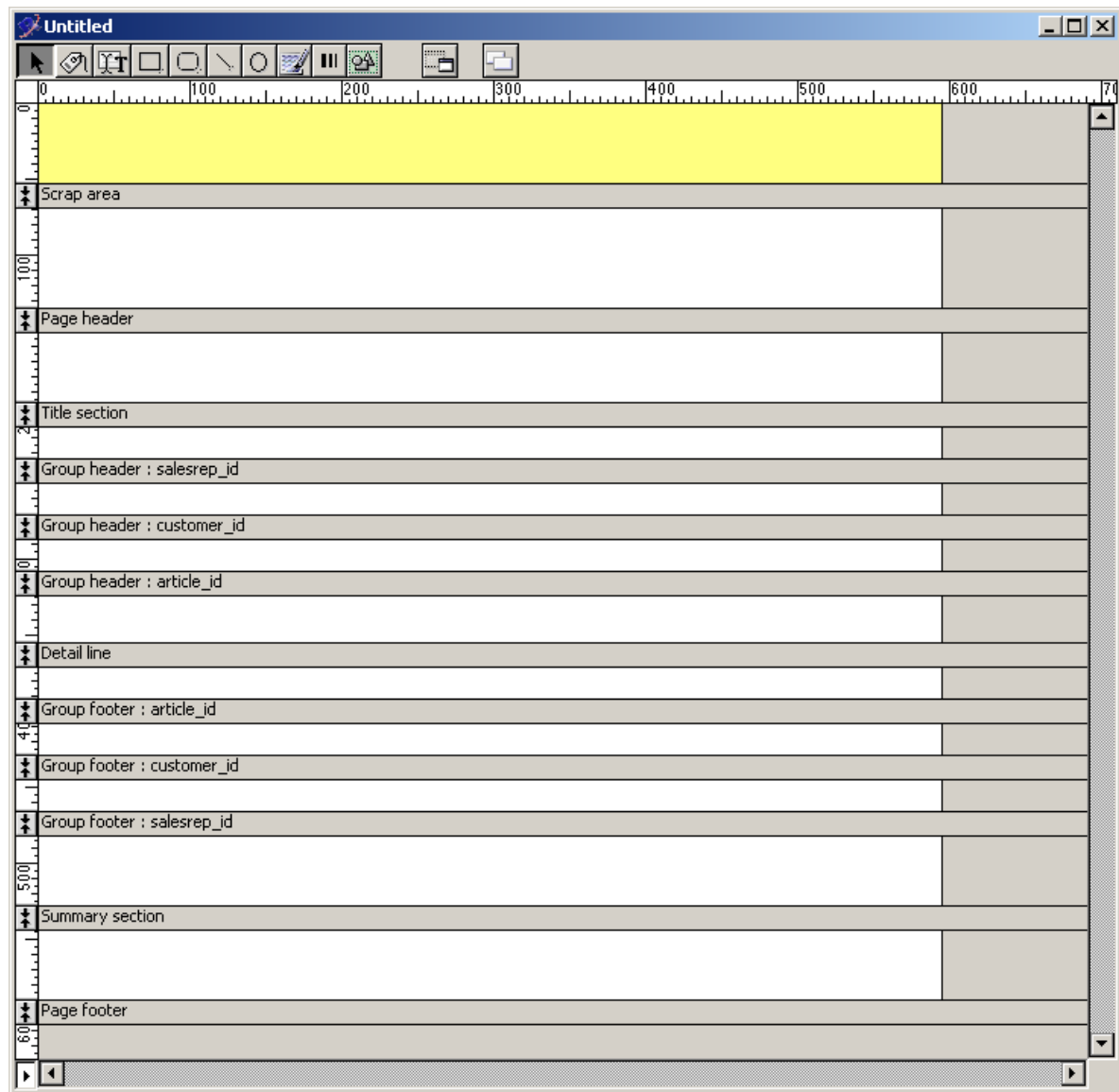


The group settings dialog box allows you to define the group expression and set the header and footer height for the data group bands.

The group expression is an everyday LiveCode expression, defining the set of data that it encompasses.

## Layout bands

As we've seen before, all the items in your report are organized in bands.



Every editor window has its own scrap area, a special band where pasted items will be relocated to so you can decide where on the report they actually need to be placed. It will resize itself automatically to fit these new items. Items in the scrap area will not be printed on your report.

Most reports also have page header and footer bands, sometimes title and summary bands as well, and each data group has its own headers and footer band where you can place items that will be printed at the start and end of each set of related data, respectively.

Each band has a resizer button on the left hand side which allows you to make the band taller or smaller by dragging the button.

## Chapter 10: Property palette

The Properties palette is where you set the properties of the items in your layouts. It breaks these properties into four groups, each represented by a single tab, regardless of the type of item you have currently selected.

These tabs are:

- Basic
- Condition
- Size & Position
- Colors & Inks

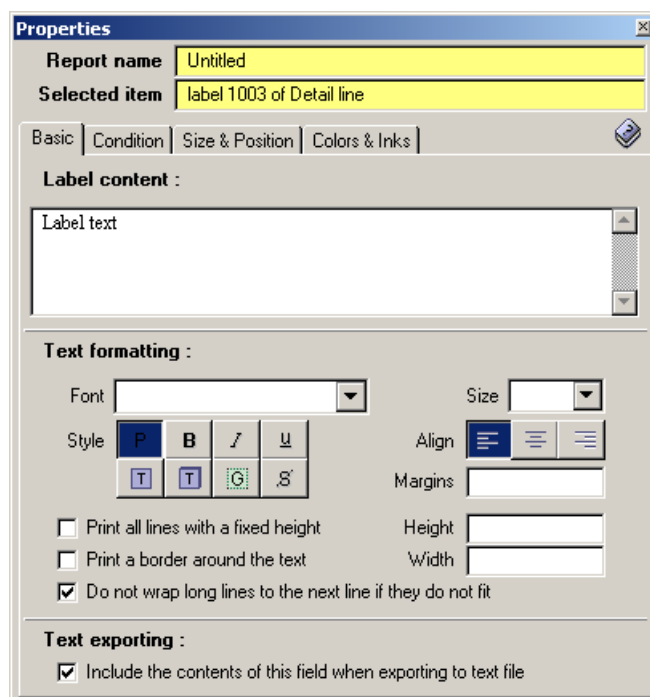
In the following paragraphs, we will cover these property groups in detail.

### Basic

In the 'Basic' tab of the Properties palette, the properties that you can set depend on the selected item in the active layout editor window.

Most of these properties map directly onto their equivalents in *LiveCode*, so you should feel right at home, even if they're concentrated into a single panel.

### Label field



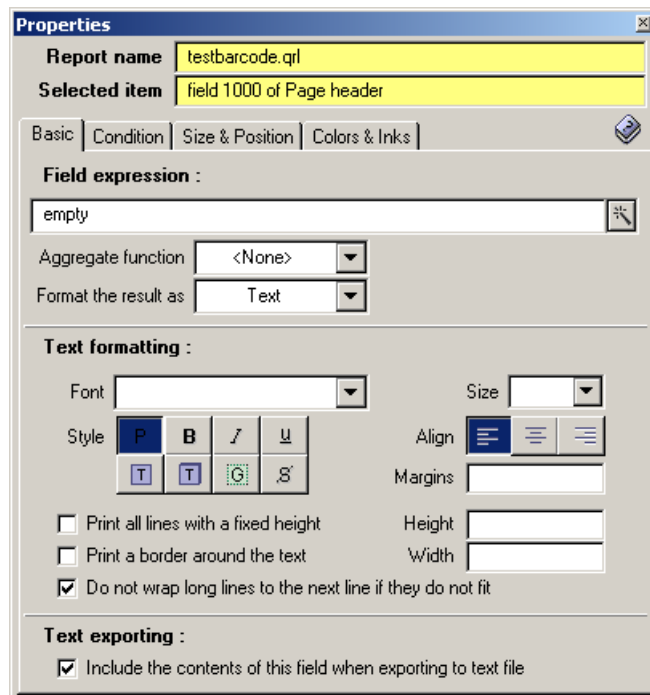
You can drag and drop formatted text from other applications into the *Label content* field, and set the various text formatting options: font, size, style, alignment and margins.

One interesting addition to the *LiveCode* text formatting settings is that you can set these individually: if you set the text size, this does not override the font or style settings as defined in your document settings.

This means that you can set the size to 10, and if you change the font settings at document level, the size will remain the same, but the font will follow along independently.

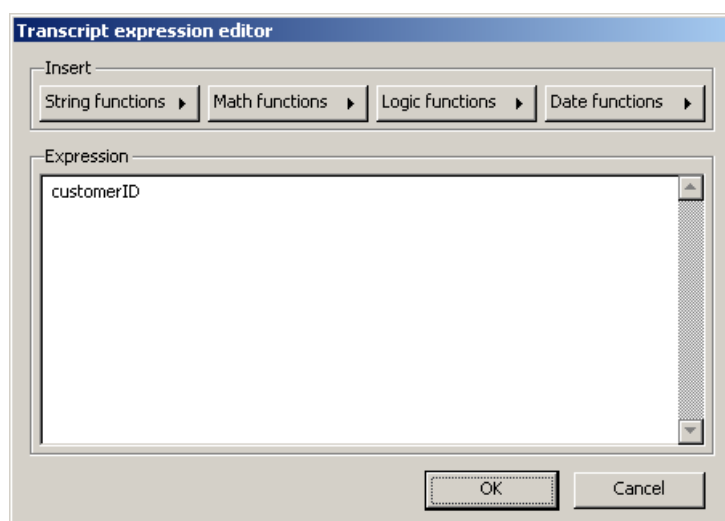
New in version 1.1 is the option to include the content of the text label when exporting the report data to a text file.

## Data field



All text formatting options as offered for Label items, also apply to data fields. Of particular interest here are the data field expression editor, the aggregate functions and result formatting options.

The expression editor helps you build a valid expression using built-in LiveCode functions.



This makes it a lot easier to achieve the data results you want to display, as you can apply the multitude of string, math, logic and date functions from the set of action menus at the top.

Select one of the following aggregate functions:

- Count
- Sum
- Average
- Lowest
- Highest
- Median
- Variance
- Standard deviation

and then select any of the reset options:

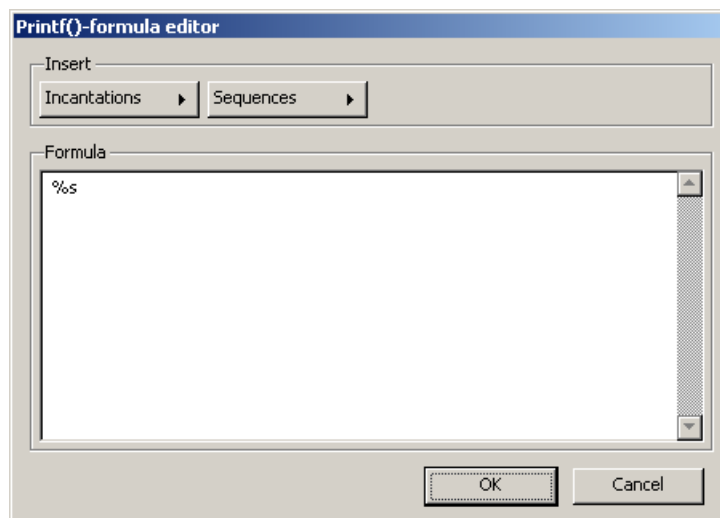
- end of report
- end of page
- a list of the data groups

To format the result, pick one of the format options:

- Text
- Unicode text
- HTML text
- RTF text
- Numeric
- Date
- Time
- Printf()-formula

The second group of these will display additional options, where you can set the number of decimals, the date format, the time format or a powerful Printf()-formula.

To help you achieve the right formatting when using a Printf()-formula, the layout builder offers you a formula editor:



Similar to the LiveCode expression editor, this makes it a lot easier to achieve the data results you want to display, as you can apply the multitude of incantations and sequences from the set of action menus at the top.

## Rectangle

The screenshot shows the 'Properties' dialog box for a 'rectangle 1000 of Page header'. The 'Report name' is 'Untitled'. The 'Selected item' is 'rectangle 1000 of Page header'. The 'Basic' tab is selected. Under 'Rectangle settings:', 'Line thickness' is set to 1 with a spinner, and 'Dashes' is an empty text field.

You can enter the line thickness directly or use the spinner to increase or decrease it, and apply a pattern of dashes by entering a comma-separated list of integers.

## Round rectangle

The screenshot shows the 'Properties' dialog box for a 'roundRect 1003 of Detail line'. The 'Report name' is 'Untitled'. The 'Selected item' is 'roundRect 1003 of Detail line'. The 'Basic' tab is selected. Under 'Round rectangle settings:', 'Line thickness' is set to 1 with a spinner, and 'Dashes' is an empty text field. Under 'Round rectangle options:', 'Corner radius' is set to 15 with a spinner.

You can enter the line thickness directly or use the spinner to increase or decrease it, and apply a pattern of dashes by entering a comma-separated list of integers.

In addition to the regular rectangle settings, you can also specify the corner radius.

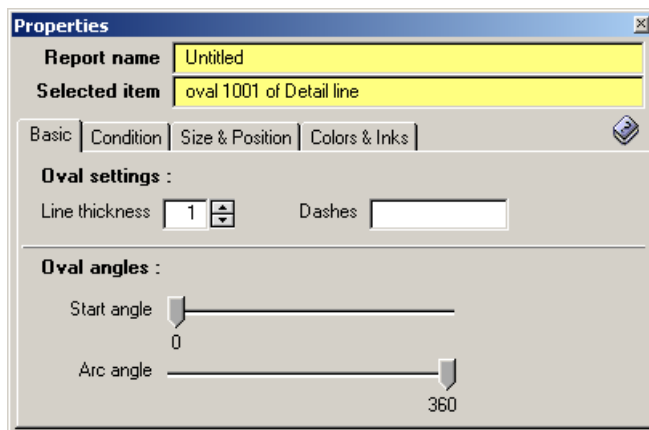
## Line

The screenshot shows the 'Properties' dialog box for a 'line 1004 of Detail line'. The 'Report name' is 'Untitled'. The 'Selected item' is 'line 1004 of Detail line'. The 'Basic' tab is selected. Under 'Line settings:', 'Line thickness' is set to 1 with a spinner, and 'Dashes' is an empty text field. Under 'Line arrows:', 'Arrow size' is set to 1 with a spinner, and there are checkboxes for 'Start Arrow' and 'End Arrow', both of which are unchecked. Under 'Line points:', there is a list box containing the coordinates '40,63' and '479,63'.

You can enter the line thickness directly or use the spinner to increase or decrease it, and apply a pattern of dashes by entering a comma-separated list of integers.

You can set the arrow size and which arrows you want to display. Furthermore, you can set the individual points of your line graphic.

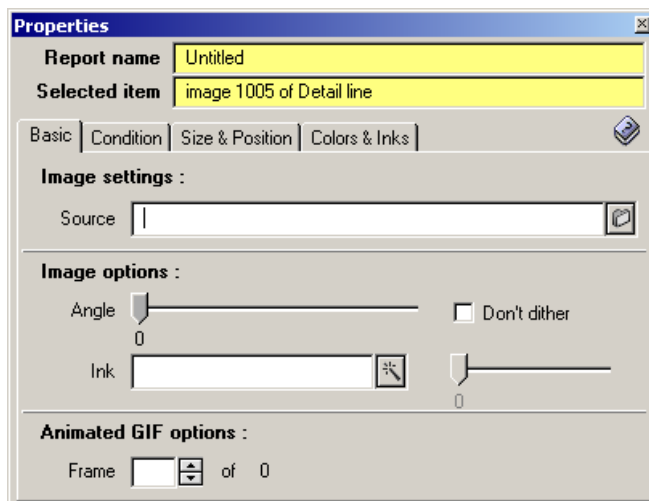
## Oval



You can enter the line thickness directly or use the spinner to increase or decrease it, and apply a pattern of dashes by entering a comma-separated list of integers.

You can change the oval into an arc by modifying the arc angle and tweaking the start angle.

## Image



You can set the image source by entering an absolute or relative path to an image file. Alternately, you can use the 'folder' button to browse for an image file on your hard drive.

The other settings allow you to rotate the image, apply an ink and fine-tune the blendlevel.

## Barcode (Professional edition only)

The screenshot shows the 'Properties' dialog box for a barcode. The 'Report name' is 'Untitled' and the 'Selected item' is 'barcode 1002 of Detail line'. The 'Basic' tab is selected. The 'Barcode expression' is 'empty'. The 'Barcode format' is 'Code 3 of 9'. The 'Text formatting' section includes a 'Font' dropdown, a 'Size' dropdown, and a 'Style' section with buttons for 'P' (Paragraph), 'B' (Bold), 'I' (Italic), 'U' (Underline), 'T' (Text), 'G' (Green), and 'S' (Strikethrough).

The ability to include barcodes was added in version 1.1, and is limited to the Professional edition. As with data fields, you can use the expression editor to build the barcode expression.

You can pick one of the following industry-standard barcode formats:

- Codabar
- Code 3 of 9
- EAN 8
- EAN 13
- Interleaved 2 of 5
- JAN
- Postnet
- UPC-A

The text formatting options apply to the display of the barcode in text format below the barcode lines (this does not apply to Postnet barcodes, which do not display their values as text).

## 2D Chart (Professional edition only)

The screenshot shows the 'Properties' dialog box for a 2D chart. The 'Report name' is 'Untitled' and the 'Selected item' is 'chart 1000 of Summary section'. The 'Basic' tab is selected. The 'Chart options' section includes a 'Chart style' dropdown set to 'Bar chart', a 'Units' dropdown, a 'Chart title' text box, and checkboxes for 'Print a border around the chart', 'Print the labels on the chart' (checked), and 'Print line markers'. The 'Label expression' is 'empty' and the 'Value expression' is 'empty'. Both expressions have a 'Reset' button and a dropdown menu set to 'End of page'.

The ability to include 2D Charts was added in version 1.1, and is limited to the Professional edition. You can choose the chart style, and enter the title and value units, which will be displayed in the chart. You can pick one of the following chart types:



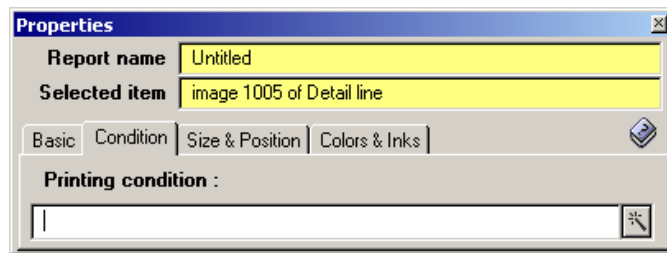
- Bar chart
- Column chart
- Line chart
- Pie chart

You can optionally print a border of a certain width around the chart, and indicate whether or not you want to print the labels (all chart types) and the markers (line charts only).

You can define both label and value expressions using the expression editor, and define when the label and value sets should be reset.

## Condition

The next tab in the Properties palette houses the Printing condition of the items.



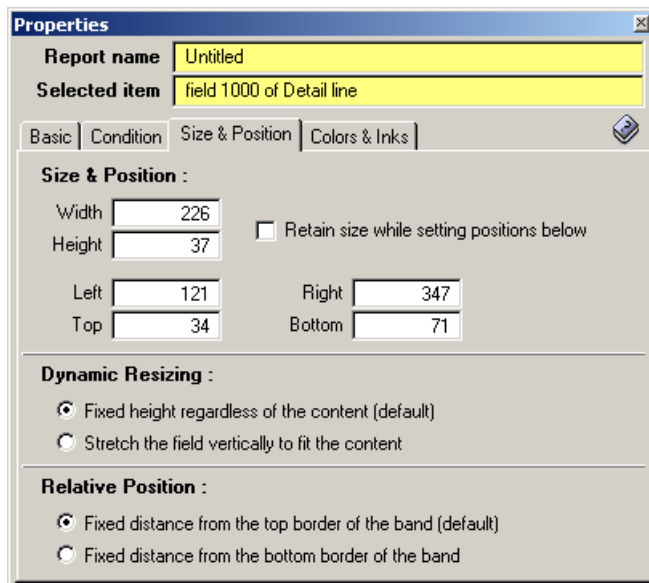
Printing conditions are everyday LiveCode expressions that determine whether an item is printed on the layout or not.

If the result of the expression is true, then the item is printed, if false then the item is not printed. While this may seem a simple concept, it gives you a lot of options.

The printing condition expression editor lets you insert a plethora of built-in LiveCode functions.

## Size & Position

The next tab in the Properties palette gives you quick access to the size and position of items.



The screenshot shows the 'Properties' palette with the 'Size & Position' tab selected. The 'Report name' is 'Untitled' and the 'Selected item' is 'field 1000 of Detail line'. The 'Size & Position' section contains input fields for Width (226), Height (37), Left (121), Right (347), Top (34), and Bottom (71). There is a checkbox 'Retain size while setting positions below' which is unchecked. The 'Dynamic Resizing' section has two radio buttons: 'Fixed height regardless of the content (default)' (selected) and 'Stretch the field vertically to fit the content'. The 'Relative Position' section has two radio buttons: 'Fixed distance from the top border of the band (default)' (selected) and 'Fixed distance from the bottom border of the band'.

An interesting option here is that by toggling the checkbox on the right, you can easily resize or move the selected item using exact numbers, without the aid of your mouse or the keyboard equivalents.

New in version 1.1 is the ability to set the resizing and positioning behaviour.

Data fields can have either a fixed height, or stretch vertically to fit the content. The printing engine will add new pages as needed.

All items can be set to remain at a fixed distance from the top border of the band or from the bottom border of the band.

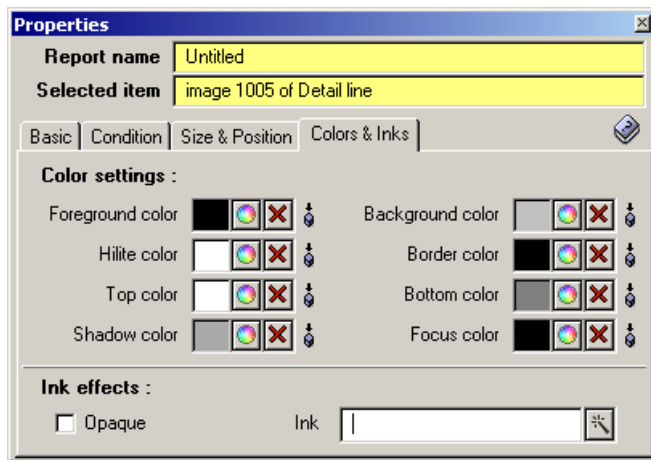
This allows you to make flexible layouts that stretch to fit the data while retaining the relative position of the items in your bands.

Combined with stretching data fields, this means that the printing order is as follows:

- (1) items that are linked to the top of the band and do not stretch ;
- (2) items that are linked to the top of the band and do stretch ;
- (3) items that are linked to the bottom of the band and do stretch ;
- (4) items that are linked to the bottom of the band and do not stretch.

## Colors & Inks

The last tab in the Properties palette is where you set the colors & ink effects for your items.



As in *LiveCode*, each type of item has a different set of supported color settings, so please refer to the *LiveCode* documentation if a setting doesn't seem to have any effect.

At the bottom, you can determine whether an item is opaque or transparent, and pick one of the cross-platform or MacOS-specific ink effects that *LiveCode* has to offer.

Which inks are offered depends on your Preferences setting, where you can choose to either display the inkmodes for Revolution 2.6 and older, or the inkmodes for Revolution 2.7 through 4.0, as well as LiveCode 4.5 and newer.

## Chapter 11: Alignment palette

It is not enough to place all the items onto the layout. To make the report as easy as possible to read and to enable recognition of data trends, it is important to make sure everything is aligned properly. Fortunately, we have the Alignment Palette at our disposal to make this a lot easier. As you can see, it is divided into three sections: Nudge, Resize and Align.



### Nudge

Select all the items you want to move, and click on the four arrows to change their positions.

Use the Increment action menu to switch between moving 1-2-5-10-20 pixels at a time.

### Resize

Select all the items you want to resize, and make their widths, heights or rectangles equal to that of the first selected item.

If you hold down the *option* or *alt-key*, all items will be resized to the widest, tallest and largest item in your selection.

### Align

Select all the items you want to align, and click the buttons to align them to the left, top, right, bottom, middle or center of the first selected item.

If you hold down the *option* or *alt-key*, all items are aligned to the left-most, right-most, highest or lowest item, or centered horizontally or vertically within the *band*.

## Chapter 12: Keyboard shortcuts

While you can do everything with your computer mouse, the layout builder offers the usual set of keyboard shortcuts, as well as some time-saving additional tricks.

| Windows               | MacOS | Description  |
|-----------------------|-------|--|
| Ctrl-N                | Cmd-N | Create a new report layout                                     |
| Ctrl-O                | Cmd-O | Open an existing report or label set layout                    |
| Ctrl-W                | Cmd-W | Close the active layout editor window                          |
| Ctrl-S                | Cmd-S | Save the active layout editor window                           |
| Ctrl-Q                | Cmd-Q | Quit the layout builder application                            |
| Ctrl-Z                | Cmd-Z | Undo the last modification                                     |
| Ctrl-X                | Cmd-X | Cut the selected items or text to the clipboard                |
| Ctrl-C                | Cmd-C | Copy the selected items or text to the clipboard               |
| Ctrl-V                | Cmd-V | Paste the items or text from the clipboard into the scrap area |
| Ctrl-D                | Cmd-D | Duplicate the selected items                                   |
| Ctrl-I                | Cmd-I | Show/Hide the properties palette                               |
| Ctrl-L                | Cmd-L | Show/Hide the alignment palette                                |
| Ctrl-K                | Cmd-K | Show/Hide the application backdrop                             |
| Delete                |       | Delete the selected items or text                              |
| Left arrow            |       | Move the selected items 1 pixel to the left                    |
| Right arrow           |       | Move the selected items 1 pixel to the right                   |
| Up arrow              |       | Move the selected items 1 pixel upwards                        |
| Down arrow            |       | Move the selected items 1 pixel downwards                      |
| Shift-Left arrow      |       | Move the selected items 10 pixel to the left                   |
| Shift-Right arrow     |       | Move the selected items 10 pixel to the right                  |
| Shift-Up arrow        |       | Move the selected items 10 pixel upwards                       |
| Shift-Down arrow      |       | Move the selected items 10 pixel downwards                     |
| Alt-Left arrow        |       | Shrink the selected items 1 pixel horizontally                 |
| Alt-Right arrow       |       | Expand the selected items 1 pixel horizontally                 |
| Alt-Up arrow          |       | Shrink the selected items 1 pixel vertically                   |
| Alt-Down arrow        |       | Expand the selected items 1 pixel vertically                   |
| Alt-Shift-Left arrow  |       | Shrink the selected items 10 pixels horizontally               |
| Alt-Shift-Right arrow |       | Expand the selected items 10 pixels horizontally               |
| Alt-Shift-Up arrow    |       | Shrink the selected items 10 pixels vertically                 |
| Alt-Shift-Down arrow  |       | Expand the selected items 10 pixels vertically                 |

## **Part 4 : API Reference**

## License and Version

### qrtReports\_InitLicense

|                |   |
|----------------|---|
| Type           | Command   |
| Introduced in  | Version 1.0   |
| Purpose        | Initialise the Quartam Reports license  |
| Syntax         | qrtReports_InitLicense <license code>   |
| Parameters     |   |
| <license code> | the license code is sent to you via email. It consists of 22 alphanumeric characters and 8 digits, separated by a dash. |

#### Example

```
qrtReports_InitLicense "ABCDABCDABCDABCDABCDAB-12345678"
```

#### Description

The qrtReports\_InitLicense command is used to unlock the report printing library. Until you unlock the library, you will not be able to preview or print reports or label sets.

Your license code also contains information regarding your license type. Different license types may have limited access to advanced features. Please refer to the website for more information – link:

<http://www.quartam.com/reports/editions.html>

#### Comments

If you pass an invalid license code, 'the result' will contain a string starting with "qrtReportsErr" – this will allow you to react accordingly.

### qrtReports\_VersionNumber

|               |  |
|---------------|--|
| Type          | Function   |
| Introduced in | Version 1.0  |
| Purpose       | Retrieve the version number of the report printing library |
| Syntax        | qrtReports_VersionNumber()                                 |

#### Example

```
if qrtReports_VersionNumber() >= "1.1.0" then
  -- it supports that shiny new feature
end if
```

#### Description

The qrtReports\_VersionNumber function is used to determine the version of the report printing library. In the future, you can use this information to ensure that the current copy of the printing library supports the feature you are using.

#### Comments

The version number consists of a major version number, a period, a minor version number, a period, and a revision number.

In keeping with traditional version numbering schemes, a major version number change implies a significant rewrite of the product, a minor version number change adds new features, and a revision number change brings tweaks and fixes.



## Printing LiveCode stacks

### qrtReports\_PrintReportForStack

|               |  |
|---------------|--|
| Type          | Command  |
| Introduced in | Version 1.0  |
| Purpose       | Print the data contained in a stack into a report, using a specified layout, reading the data one card at a time, optionally in a specified range. |
| Syntax        | qrtReports_PrintReportForStack <layout filepath>, <stack name>, <preview flag>[, <print range>[, <global variables>]]                              |

#### Parameters

|                    |   |
|--------------------|---|
| <layout filepath>  | the path to the layout file of the report                                 |
| <stack name>       | the short name of the stack to print                                      |
| <preview flag>     | true if you want to show a preview, false for printing without previewing |
| <print range>      | all   marked   unmarked   MS-Word style range                             |
| <print range>      | all   marked   unmarked   MS-Word style range                             |
| <global variables> | a comma-delimited list of global variable names                           |

#### Example

```
qrtReports_PrintReportForStack "addresslist.qrl", "Addresses", false, "marked"
qrtReports_PrintReportForStack "phonelist.qrl", "Phone book", true, "1-7,9"
```

#### Description

This command allows you to print the content of a stack without the need to script your own data broker, using a predefined report layout. You can choose to display a preview instead of printing directly, and have extensive control over the cards that are printed.

#### Comments

If an error occurs while printing the report, this command will ‘throw’ an error starting with “qrtReportsErr”. This will allow you to react accordingly from within a try-catch-end try block.

### qrtReports\_PrintLabelSetForStack

|               |   |
|---------------|---|
| Type          | Command   |
| Introduced in | Version 1.0   |
| Purpose       | Print the data contained in a stack into a label set, using a certain layout, reading the data from one card at a time, in the specified range. |
| Syntax        | qrtReports_PrintLabelSetForStack <layout filepath>, <stack name>, <preview flag>[, <print range>[, <global variables>]]                         |

#### Parameters

|                    |  |
|--------------------|--|
| <layout filepath>  | the path to the layout file of the label set           |
| <stack name>       | the short name of the stack to print                   |
| <preview flag>     | true if you want to show a preview, false for printing |
| <print range>      | all   marked   unmarked   MS-Word style range          |
| <global variables> | a comma-delimited list of global variable names        |

#### Example

```
qrtReports_PrintLabelSetForStack "addresses.qll", "Addresses", false, "all"
qrtReports_PrintLabelSetForStack "artlabels.qll", "Articles", true, "marked"
```

#### Description

This command allows you to print the content of a stack without the need to script your own data broker, using a predefined label set layout. You can choose to display a preview instead of printing directly, and have extensive control over the cards that are printed.

#### Comments

If an error occurs while printing the report, this command will ‘throw’ an error starting with “qrtReportsErr”. This will allow you to react accordingly from within a try-catch-end try block.

## ***Printing database cursors***

### **qrtReports\_PrintReportForCursor**

|                      |  |
|----------------------|--|
| <b>Type</b>          | Command  |
| <b>Introduced in</b> | Version 1.0  |
| <b>Purpose</b>       | Print the data contained in a database cursor into a report, using a certain layout, reading the data from one record at a time, in the specified range.         |
| <b>Syntax</b>        | <code>qrtReports_PrintReportForCursor &lt;layout filepath&gt;, &lt;cursor id&gt;, &lt;preview flag&gt;[, &lt;print range&gt;[, &lt;global variables&gt;]]</code> |

#### **Parameters**

|                                       |  |
|---------------------------------------|--|
| <code>&lt;layout filepath&gt;</code>  | the path to the layout file of the report              |
| <code>&lt;cursor id&gt;</code>        | the id of the database cursor to print                 |
| <code>&lt;preview flag&gt;</code>     | true if you want to show a preview, false for printing |
| <code>&lt;print range&gt;</code>      | all   MS-Word style range                              |
| <code>&lt;global variables&gt;</code> | a comma-delimited list of global variable names        |

#### **Example**

```
qrtReports_PrintReportForCursor "addresslist.qrl", tAddressesCursor, false, "all"  
qrtReports_PrintReportForCursor "phonelist.qrl", tPhonesCursor, true, "1-7,9"
```

#### **Description**

This command allows you to print the content of a database cursor without the need to script your own data broker, using a predefined report layout. You can choose to display a preview instead of printing directly, and have extensive control over the cards that are printed.

#### **Comments**

If an error occurs while printing the report, this command will ‘throw’ an error starting with “qrtReportsErr”. This will allow you to react accordingly from within a try-catch-end try block.

### **qrtReports\_PrintLabelSetForCursor**

|                      |   |
|----------------------|---|
| <b>Type</b>          | Command   |
| <b>Introduced in</b> | Version 1.0   |
| <b>Purpose</b>       | Print the data contained in a database cursor into a label set, using a certain layout, reading the data from one record at a time, in the specified range.       |
| <b>Syntax</b>        | <code>qrtReports_PrintLabelSetForStack &lt;layout filepath&gt;, &lt;cursor id&gt;, &lt;preview flag&gt;[, &lt;print range&gt;[, &lt;global variables&gt;]]</code> |

#### **Parameters**

|                                       |  |
|---------------------------------------|--|
| <code>&lt;layout filepath&gt;</code>  | the path to the layout file of the label set           |
| <code>&lt;cursor id&gt;</code>        | the id of the database cursor to print                 |
| <code>&lt;preview flag&gt;</code>     | true if you want to show a preview, false for printing |
| <code>&lt;print range&gt;</code>      | all   MS-Word style range                              |
| <code>&lt;global variables&gt;</code> | a comma-delimited list of global variable names        |

#### **Example**

```
qrtReports_PrintLabelSetForCursor "addresses.qll", tAddressesCursor, false, "all"  
qrtReports_PrintLabelSetForCursor "artlabels.qll", tArticlesCursor, true, "1,10-19"
```

#### **Description**

This command allows you to print the content of a database cursor without the need to script your own data broker, using a predefined label set layout. You can choose to display a preview instead of printing directly, and have extensive control over the records that are printed.

#### **Comments**

If an error occurs while printing the report, this command will ‘throw’ an error starting with “qrtReportsErr”. This will allow you to react accordingly from within a try-catch-end try block.

## ***Printing automated queries***

### **qrtReports\_PrintReportForQuery**

|                      |  |
|----------------------|--|
| <b>Type</b>          | Command  |
| <b>Introduced in</b> | Version 1.0  |
| <b>Purpose</b>       | Print the data contained in an automated database query into a report, using a certain layout, reading the data from one record at a time, in the specified range. |
| <b>Syntax</b>        | <code>qrtReports_PrintReportForQuery &lt;layout filepath&gt;, &lt;query name&gt;, &lt;preview flag&gt;[, &lt;print range&gt;[, &lt;global variables&gt;]]</code>   |

#### **Parameters**

|                                       |  |
|---------------------------------------|--|
| <code>&lt;layout filepath&gt;</code>  | the path to the layout file of the report              |
| <code>&lt;query name&gt;</code>       | the name of the automated database query to print      |
| <code>&lt;preview flag&gt;</code>     | true if you want to show a preview, false for printing |
| <code>&lt;print range&gt;</code>      | all   MS-Word style range                              |
| <code>&lt;global variables&gt;</code> | a comma-delimited list of global variable names        |

#### **Example**

```
qrtReports_PrintReportForQuery "addresslist.qrl", "AddressesQuery", false, "all"
qrtReports_PrintReportForQuery "phonelist.qrl", "PhonesQuery", true, "1-7,9"
```

#### **Description**

This command allows you to print the content of an automated database query without the need to script your own data broker, using a predefined report layout. You can choose to display a preview instead of printing directly, and have extensive control over the records that are printed.

#### **Comments**

If an error occurs while printing the report, this command will ‘throw’ an error starting with “qrtReportsErr”. This will allow you to react accordingly from within a try-catch-end try block.

### **qrtReports\_PrintLabelSetForQuery**

|                      |   |
|----------------------|---|
| <b>Type</b>          | Command   |
| <b>Introduced in</b> | Version 1.0   |
| <b>Purpose</b>       | Print the data contained in an automated database query into a label set, using a certain layout, reading the data from one record at a time, in the specified range. |
| <b>Syntax</b>        | <code>qrtReports_PrintLabelSetForStack &lt;layout filepath&gt;, &lt;cursor id&gt;, &lt;preview flag&gt;[, &lt;print range&gt;[, &lt;global variables&gt;]]</code>     |

#### **Parameters**

|                                       |  |
|---------------------------------------|--|
| <code>&lt;layout filepath&gt;</code>  | the path to the layout file of the label set           |
| <code>&lt;query name&gt;</code>       | the name of the automated database query to print      |
| <code>&lt;preview flag&gt;</code>     | true if you want to show a preview, false for printing |
| <code>&lt;print range&gt;</code>      | all   MS-Word style range                              |
| <code>&lt;global variables&gt;</code> | a comma-delimited list of global variable names        |

#### **Example**

```
qrtReports_PrintLabelSetForQuery "addresses.qll", "AddressesQuery", false, "all"
qrtReports_PrintLabelSetForQuery "artlabels.qll", "ArticlesQuery", true, "1,10-19"
```

#### **Description**

This command allows you to print the content of an automated database query without the need to script your own data broker, using a predefined label set layout. You can choose to display a preview instead of printing directly, and have extensive control over the records that are printed.

#### **Comments**

If an error occurs while printing the report, this command will ‘throw’ an error starting with “qrtReportsErr”. This will allow you to react accordingly from within a try-catch-end try block.

## Printing with data brokers

### qrtReports\_PrintReport

|                   |   |
|-------------------|---|
| Type              | Command   |
| Introduced in     | Version 1.0   |
| Purpose           | Print the data from a data broker into a report, using a certain layout, sending callbacks to the data broker to evaluate expressions, prepare the data for a new detail band, determine the end of the report, and signal group changes. |
| Syntax            | qrtReports_PrintReport <layout filepath>, <data broker>, <preview flag>   |
| Parameters        |   |
| <layout filepath> | the path to the layout file of the label set  |
| <data broker>     | the long name or id of the data broker object   |
| <preview flag>    | true if you want to show a preview, false for printing  |

#### Example

```
qrtReports_PrintReport "addresslist.qrl", the long name of me, false
qrtReports_PrintReport "phonelist.qrl", the long id of button "PhonesBroker", true
```

#### Description

This command is the heart and soul of the report printing library. You gain full control over all aspects of the data in the report by developing a data broker object in LiveCode to interact with the printing library. This way, you can combine data from multiple sources into one comprehensive report. See below for more information on the callback messages and functions that you will need to implement in your data broker object script.

#### Comments

If an error occurs while printing the report, this command will ‘throw’ an error starting with “qrtReportsErr”. This will allow you to react accordingly from within a try-catch-end try block.

### qrtReports\_PrintLabelSet

|                   |   |
|-------------------|---|
| Type              | Command   |
| Introduced in     | Version 1.0   |
| Purpose           | Print the data from a data broker into a label set, using a certain layout, sending callbacks to the data broker to evaluate expressions, prepare the data for a new detail band, and determine the end of the label set. |
| Syntax            | qrtReports_PrintLabelSet <layout filepath>, <data broker>, <preview flag>   |
| Parameters        |   |
| <layout filepath> | the path to the layout file of the label set  |
| <data broker>     | the long name or id of the data broker object   |
| <preview flag>    | true if you want to show a preview, false for printing  |

#### Example

```
qrtReports_PrintLabelSet "addresses.qll", the long name of me, false
qrtReports_PrintLabelSet "artlabels.qll", the long id of button "ArticlesBroker", true
```

#### Description

This command is the heart and soul of the report printing library. You gain full control over the data aspects by scripting a data broker object to interact with the printing library. This way, you can combine data from multiple sources into one comprehensive label set.

See below for more information on the callback messages and functions that you will need to implement in your data broker object script.

#### Comments

If an error occurs while printing the report, this command will ‘throw’ an error starting with “qrtReportsErr”. This will allow you to react accordingly from within a try-catch-end try block.

## qrtReports\_EndOfReport

|                      |   |
|----------------------|---|
| <b>Type</b>          | Callback function   |
| <b>Introduced in</b> | Version 1.0   |
| <b>Purpose</b>       | This function is called in your data broker by the report engine, to determine whether the processing has reached the end of the report. Return false unless all the data has been printed. |
| <b>Return value</b>  | true = all the data has been printed<br>false = there is more data to print   |

### Example

```
local sCurrentRecord, sRecordCount
on qrtReports_MoveNextRecord
    add 1 sCurrentRecord
end qrtReports_MoveNextRecord
function qrtReports_EndOfReport
    return sCurrentRecord > sRecordCount
end qrtReports_EndOfReport
```

### Comments

Make sure you test your implementation of this callback message carefully before using it in a printing command, as you can effectively send processing off into an infinite loop.

If the script of your data broker does not implement this function, the printing library will 'throw' an error starting with "qrtReportsErr". This will allow you to react accordingly from within a try-catch-end try block.

## qrtReports\_ValueOfExpression

|                      |  |
|----------------------|--|
| <b>Type</b>          | Callback function  |
| <b>Introduced in</b> | Version 1.0  |
| <b>Purpose</b>       | This function is called in your data broker by the report engine, to evaluate the expressions that can constitute field contents, printing conditions and data groups. |

### Parameters

<value expression> specially formatted version of the expression to evaluate

**Special note** Due to limitations in the value() function of LiveCode, all quotes in the expression have been replaced with the substitution string <QUOTE\_REPLACEMENT>, and your implementation will have to do the replacement.

**Return value** the value of the expression to evaluate

### Example

```
function qrtReports_ValueOfExpression pExpression
    ## this function will reformat and evaluate an expression
    ## and return the result
    put pExpression into tExpression
    replace "<QUOTE_REPLACEMENT>" with quote in tExpression
    return value(tExpression)
end qrtReports_ValueOfExpression
```

### Comments

This callback function allows you to use memory variables, custom functions and any other functionality at your disposal, to evaluate the expressions that can make up the field data in your reports and label set layouts.

If the script of your data broker does not implement this function, the printing library will 'throw' an error starting with "qrtReportsErr" – this will allow you to react accordingly from within a try-catch-end try block.

## qrtReports\_MoveNextRecord

**Type** Callback message

**Introduced in** Version 1.0

**Purpose** This message is sent to your data broker by the report engine, after printing the current detail band, as long as the end of the report hasn't been reached, so you can prepare the data for the next detail band.

### Example

```
local sCurrentRecord, sRecordCount
on qrtReports_MoveNextRecord
    add 1 sCurrentRecord
end qrtReports_MoveNextRecord
function qrtReports_EndOfReport
    return sCurrentRecord > sRecordCount
end qrtReports_EndOfReport
```

### Comments

If an error occurs while printing the report and your script doesn't respond, this command will 'throw' an error starting with "qrtReportsErr". This will allow you to react accordingly from within a try-catch-end try block.

## qrtReports\_MoveNextGroup

**Type** Callback message

**Introduced in** Version 1.0

**Purpose** This message is sent to your data broker by the report engine when the evaluation of a data group expression returns a different result when compared to the previous result, so you can prepare the data for this group change.

### Parameters

<group id> the id of the data group that has changed

<group expression> the value expression of the data group that has changed

### Example

```
on qrtReports_MoveNextGroup pID, pExpression
    switch pID
    case 1
        -- the article id has changed ; get information for the new article
        ReadArticleData
        break
    case 2
        -- the customer id has changed ; get information for the new customer
        ReadCustomerData
        break
end qrtReports_MoveNextGroup
```

### Comments

Implementing this callback message is entirely optional. No error will be thrown if your data broker script does not contain a handler for this message.

## qrtReports\_MoveNextPage

|                      |  |
|----------------------|--|
| <b>Type</b>          | Callback message   |
| <b>Introduced in</b> | Version 1.1  |
| <b>Purpose</b>       | This message is sent to your data broker by the report engine when a new page is created, and before the page header is printed, so you can prepare the data for this page change. |

### Parameters

|                   |  |
|-------------------|--|
| <new page number> | the number of the new page that has been created |
|-------------------|--|

### Example

```
local sQuoteOfTheDay
on qrtReports_MoveNextPage pNewPageNumber
  -- we display a random quote at the top of each page
  put randomQuote() into sQuoteOfTheDay
end qrtReports_MoveNextPage
```

### Comments

Implementing this callback message is entirely optional. No error will be thrown if your data broker script does not contain a handler for this message.

## qrtReports\_RecoverFromError

|                      |  |
|----------------------|--|
| <b>Type</b>          | Callback message   |
| <b>Introduced in</b> | Version 1.1.4  |
| <b>Purpose</b>       | This message is sent to your data broker by the report engine when an error has occurred, allowing your data broker to free up any resources in the event of an error. |

### Example

```
on qrtReports_RecoverFromError
  -- flag there was an error and handle it later
  put true into sErrorHappened
end qrtReports_RecoverFromError
```

### Comments

Implementing this callback message is entirely optional.

## Handling format errors

### qrtReports\_HandleFormatError

|               |  |
|---------------|--|
| Type          | Callback message   |
| Introduced in | Version 1.0  |
| Purpose       | This message is sent to your data broker by the report engine when the evaluation of a data group expression returns a different result from the most recent previous result, so you can prepare the data for this group change. |

#### Parameters

|                      |                                      |
|----------------------|--------------------------------------|
| <unformatted value>  | the data that could not be formatted |
| <format description> | the format as set in the data field  |

#### Example

```
local sFELog, sFECounter
on qrtReports_HandleFormatError pValue, pFormat
  add 1 to sFECounter
  -- log the necessary information to review and fix the formatting mistakes
  put "<" & sFECounter & ">" & tab & \
    "<" & pValue & ">" & tab & \
    "<" & pFormat & ">" & return after sFELog
  -- print the error counter instead of the unformatted value of the data
  return "** FE:" && sFECounter && "**"
end qrtReports_HandleFormatError
```

#### Comments

Implementing this callback message is entirely optional. No error will be thrown if your data broker script does not contain a handler for this message.

### qrtReports\_SetFormatErrorHandling

|               |  |
|---------------|--|
| Type          | Command  |
| Introduced in | Version 1.0  |
| Purpose       | This command lets you decide how the printing library handles formatting errors. |

#### Parameters

|                  |                       |
|------------------|-----------------------|
| <error handling> | throw   skip   ignore |
|------------------|-----------------------|

#### Example

```
qrtReports_SetFormatErrorHandling "throw" -- throw an exception upon format error
qrtReports_SetFormatErrorHandling "skip"  -- don't print if the format is wrong
qrtReports_SetFormatErrorHandling "ignore" -- print the data without formatting
```

#### Comments

If you try to set the format error handling to another method than the predefined ones, this command will ‘throw’ an error starting with “qrtReportsErr” – this is similar to how the LiveCode engine handles setting properties to invalid values.



## Miscellaneous commands and functions

### qrtReports\_GetCurrentPage

|               |  |
|---------------|--|
| Type          | Function                                     |
| Introduced in | Version 1.1                                  |
| Purpose       | Retrieve the page number of the current page |
| Syntax        | qrtReports_GetCurrentPage()                  |

#### Example

```
on qrtReports_MoveNextGroup pID, pExpression
  put qrtReports_GetCurrentPage() into sCurrentPage
  ...
end qrtReports_MoveNextGroup
-- or use it directly in a data field expression
"Page:" && qrtReports_GetCurrentPage()
```

#### Description

The qrtReports\_GetCurrentPage function is used to determine the current page in the report printing cycle. You can use this during any of the custom data broker callback events to differentiate behaviour between pages, or directly in a data field expression on your layout.

#### Comments

This function is available for layouts printed with or without the use of a custom data broker. The result is always a positive integer.

### qrtReports\_SetPreviewWindowState

|               |   |
|---------------|---|
| Type          | Command   |
| Introduced in | Version 1.1.4                                   |
| Purpose       | Set the style of the preview window             |
| Syntax        | qrtReports_SetPreviewWindowState <window style> |

#### Parameters

<window style>            toplevel | modal | modeless | palette

#### Example

```
qrtReports_SetPreviewWindowState "toplevel" -- show preview in toplevel window
qrtReports_SetPreviewWindowState "modeless" -- show preview in modeless dialog
qrtReports_SetPreviewWindowState "modal"    -- show preview in modal dialog
qrtReports_SetPreviewWindowState "palette"   -- show preview in palette window
```

#### Description

Use the qrtReports\_SetPreviewStyle command to determine how the preview window will be displayed: as a toplevel window, a modeless dialog, a modal dialog or a palette window. The default preview window style is “modeless.” This command allows you to integrate the preview with the modality of the windows in your existing application.

#### Comments

If you are previewing a report from within a modal dialog box, make sure to set the preview window style to “modal” to ensure that the preview window doesn’t hide behind your open dialog box.

## qrtReports\_SetProgressWindowStyle

|                      |  |
|----------------------|--|
| <b>Type</b>          | Command  |
| <b>Introduced in</b> | Version 1.1.4                                    |
| <b>Purpose</b>       | Set the style of the progress window             |
| <b>Syntax</b>        | qrtReports_SetProgressWindowStyle <window style> |
| <b>Parameters</b>    |  |
| <window style>       | toplevel   modeless   palette                    |

### Example

```
qrtReports_SetProgressWindowStyle "toplevel"  -- show progress in toplevel window
qrtReports_SetProgressWindowStyle "modeless"  -- show progress in modeless dialog
qrtReports_SetProgressWindowStyle "palette"    -- show progress in palette window
```

### Description

Use the `qrtReports_SetProgressStyle` command to determine how the preview window will be displayed: as a toplevel window, a modeless dialog or a palette window. The default preview window style is “modeless.” This command allows you to integrate the preview with the modality of the windows in your existing application.

### Comments

If you are previewing a report from within a modal dialog box, make sure to set the preview window style to “palette” to ensure that the preview window doesn’t hide behind your open dialog box.

# Appendix A: License Agreement

## QUARTAM REPORTS LICENSE AGREEMENT AND WARRANTY DISCLAIMER

License for Quartam Reports software, including Demo, Express and Professional Editions and all associated files, including installer archives and all related documentation ("Quartam Reports Software")

IMPORTANT -- READ CAREFULLY: By using the Quartam Reports Software you agree to be and are hereby bound by the terms of this License Agreement ("Agreement"). If you do not accept the terms of this Agreement you must not use the Quartam Reports Software; and you must immediately destroy all copies of the Quartam Reports Software.

### 1. GRANT OF LICENSE:

a) Quartam Software/Jan Schenkel ("Quartam") hereby grant you a non-exclusive non-transferable license to use the Quartam Reports Software subject to the terms set out in this Agreement:

### 2. THE DEMO EDITION

a) You may request a key to evaluate the Quartam Reports Software as a Demo Edition ("Demo Edition") for free from our web site. The Demo Edition will allow you a limited time period with the Demo Edition as specified on our web site. You may not attempt to use the Demo Edition without this key or attempt to circumvent the time limit included in the key.

b) You may use the Demo Edition strictly for evaluation purposes only. Unlike the Licensed Editions (below) you may NOT distribute software created with this edition for free, commercially, or within your organization.

### 3. THE LICENSED EDITION

a) You may upgrade from the Demo Edition to a licensed edition of Express or Professional by paying a fee ("Licensed Edition"). If you are an educational organization as defined by the terms on our web site, you may purchase an Educational Edition ("Educational Edition"), whereby a discount is given and certain additional restrictions applied. Proof of your Educational status will be required and in the event of any dispute, Quartam reserve the

absolute right not to supply, or allow a 3rd party to supply, an Educational edition with associated discount.

b) Upon purchase of a license, you will be sent a key to license the Demo Edition ("Licensing Key"), which will remove the trial period limitation by installing the Licensing Key into the license file ("License File"), causing the Quartam Reports Software to become the fully functional Licensed Edition.

c) The Licensed Edition must be used exclusively by a single individual and may not be transferred or loaned to another user. If you purchased the Professional Edition The Licensed User can be changed on completion of a license transfer form, payment of a nominal fee and the installation of a new Licensing Key.

d) If you have purchased multiple copies of the software, you must nominate a single individual as a contact ("License Contact"). Other individuals, who do not have to be named, may use copies of the Licensed Edition, up to the maximum number of users purchased with your license, each known as ("Licensed User"). At no time may more individuals use the Licensed Edition than the number of copies purchased with the license. The License Contact is responsible for enforcing this restriction.

e) The Express Edition has built in limits that restrict the use of certain features ("Limits"). Such Limits are published on our web site from time to time and may be varied at any time without notice. You may not attempt to exceed the Limits.

f) The Licensed User must not permit any other person to access the Licensing Key or License File.

g) Quartam will provide basic technical support via email, to the Licensed User in the case of a single user license, or the Licensed Contact in the case of multiple users for 90 days from date of purchase, relating to installation of the product only. If you purchased the Professional Edition additional technical support may be provided in the form of a specified number of support incidents, the terms and conditions of this support if offered at the time of purchase, are published and amended from time to time on our web site.

h) You may not resell, rent, lease or otherwise transfer rights to the Quartam Reports Software for any financial gain or remove any proprietary notices or labels on the Quartam Reports Software.

#### 4. DISTRIBUTION OF CREATED WORK

a) Permission is hereby granted to distribute your Created Software by any method to anyone, subject the terms of this Section 4 and this Agreement.

b) You may use the Licensed Edition to create report and label set layouts for your own use or to give to others either free of charge or for a fee under the terms of this Clause 4 ("Created Software") except if you purchased an Educational Edition, where you may distribute layouts but may not charge for them.

b) You may use the Licensed Edition to create report printing features for your own software, and include them in your stacks or self contained executable programs. You may not give these features to others either free of charge or for a fee under the terms of this Clause 4, these parties must purchase their own Licensed Edition. You may not attempt to get round any of these limitations or encourage users of your Created Software to attempt to do so.

c) Copyright notices, as defined in Appendix A, must appear wherever any other copyright or trademark notice appears in any standalone executable that uses this Created Software, as well as in its documentation.

d) You may not modify, translate, reverse engineer, decompile, disassemble or create derivative works based upon the Quartam Reports Software.

## 5. LAYOUT BUILDER APPLICATION

a) The Quartam Reports Software includes a copy of the layout builder application FOR PERSONAL USE ONLY -- this means that you cannot give a copy of this application to your end-users along with the stacks or standalones that you distribute. If you need to include a layout builder in your application so that your end-users can modify the report layouts, contact Quartam for a special bundle license or a custom version that you can embed into your application.

## 6. COMPETITION

a) You must not at any time during the period of this License create or distribute Created Software to be used as a generic report generation development tool in circumstances which Quartam considers will have an appreciable adverse effect on the sales of Quartam Reports Software. This clause is intended to prevent you from using the Quartam Reports Software to produce something that is directly damaging to our business. Any software that you produce that requires the end user to use a Licensed Edition of the Quartam Reports Software will not be considered in competition, nor will any Created Software that does not involve any sort of report layout editing. In the event that either you or Quartam

determines that your software is in such competition, you must withdraw the Created Software immediately. It is agreed that producing software that is in competition with the Quartam Reports Software could cause irreparable harm and injury and Quartam will be entitled, in addition to any other rights and remedies it may have at law or in equity, to an interdict, injunction or other similar relief enjoining or restraining you from doing or continuing to distribute such Created Software.

#### 7. TITLE

a) Title, ownership, rights, and intellectual property rights in and to the Quartam Reports Software and all related documentation shall remain with Quartam. The Quartam Reports Software is protected by the copyright laws of the Kingdom of Belgium and international copyright treaties. For the purposes of this license "intellectual property rights" means any and all patents, copyright, registered or unregistered design rights, trade marks, trade names, know-how, database rights or other intellectual property rights.

#### 8. DISCLAIMER OF WARRANTY

a) THE TOTAL LIABILITY WHICH QUARTAM OWES TO YOU IN RESPECT OF BREACH OF CONTRACT, NEGLIGENCE OR OTHERWISE SHALL, OTHER THAN IN CIRCUMSTANCES IN WHICH SUCH ACTS RESULT IN DEATH OR PERSONAL INJURY IN WHICH CASE LIABILITY SHALL BE UNLIMITED, BE LIMITED TO THE SUM OF €1000 IN RESPECT OF ANY ONE CLAIM OR AGGREGATE OF CLAIMS ARISING IN ANY TWELVE MONTH PERIOD;

b) THE SOFTWARE AND DOCUMENTATION ARE PROVIDED AS IS WITHOUT WARRANTY OF ANY KIND. TO THE MAXIMUM EXTENT PERMITTED BY APPLICABLE LAW, JAN SCHENKEL FURTHER DISCLAIMS ALL WARRANTIES, INCLUDING WITHOUT LIMITATION ANY IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, AND NONINFRINGEMENT OF ANY THIRD PARTY INTELLECTUAL PROPERTY RIGHTS;

c) THE ENTIRE RISK ARISING OUT OF THE USE OR PERFORMANCE OF THE SOFTWARE AND DOCUMENTATION REMAINS WITH YOU. TO THE MAXIMUM EXTENT PERMITTED BY APPLICABLE LAW, IN NO EVENT SHALL RUNTIME BE LIABLE FOR ANY CONSEQUENTIAL, INCIDENTAL, DIRECT, INDIRECT, SPECIAL, PUNITIVE, OR OTHER DAMAGES WHATSOEVER (INCLUDING, WITHOUT LIMITATION, DAMAGES FOR LOSS OF BUSINESS PROFITS, BUSINESS INTERRUPTION, LOSS OF BUSINESS INFORMATION, OR OTHER PECUNIARY LOSS) ARISING OUT OF THIS AGREEMENT OR THE USE OF OR INABILITY TO USE THE PRODUCT, EVEN IF QUARTAM HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES;

#### 9. TERMINATION

a) This license shall terminate automatically if you fail to comply with the

limitations described in this Agreement. No notice shall be required from Quartam to effectuate such termination. On termination you must destroy all copies of the Quartam Reports Software including the installer archive and any copies of it therewith. However, such termination does not prejudice the right of Quartam to obtain injunctive, equitable or any other relief in the event of such a breach.

#### 10. GOVERNING LAW

a) This License shall be governed by the laws of the Kingdom of Belgium and in the event of any dispute the parties agree to submit to the non-exclusive jurisdiction of the courts of the canton of Dendermonde.

#### 11. ENTIRE AGREEMENT

a) This License constitutes the entire agreement between Jan Schenkel and you with respect to the subject matter hereof and with the exception of any fraudulent or negligent misrepresentations supersedes all prior oral or written understandings, communications or agreements not specifically incorporated herein. The only exception to this is any explicit terms you may have agreed to before downloading the package from our web site, or any other supplementary terms made clear by us prior to your agreeing to this agreement. any such terms will form part of this agreement. This License may not be modified except in a writing duly signed by an authorized representative of Quartam and you.

#### APPENDIX A - Required Copyright Notice

You must include the following copyright notice where other such notices appear. In the event that such other notices do not appear in the Created Software, this notice must be placed in a reasonable location.

Portions (c)2003-2011 Quartam Software, All Rights Reserved Worldwide.