RLIB Programmers Manual

SICOM Systems, INC.

4140 Skyron Drive, Doylestown PA 18901

Bob Doan

Robert Kratz

Chet Heilman

RLIB Programmers Manual

by **SICOM Systems, INC., 4140 Skyron Drive, Doylestown PA 18901**
by Bob Doan

by Robert Kratz

by Chet Heilman

Table of Contents

1. Introduction 1

RLIB in a nutshell 1
What is a Report Writer 1
This Manual 1

2. Basic Training 2

A little Terminology 2
The DATE variable type 2
Fixed Point 3
Data Sources 3
Report Layout 4
Breaks Explained 4
Report Variables 5
Defining a Report 5
SAMPLE XML FILE 5
Report 6
Output 7
HorizontalLine 7
Image 7
Line 7
Literals 8
Field 8
ReportHeader 8
ReportFooter 8
PageHeader 8
PageFooter 8
Detail 8
Variables 9
Breaks 9

3. RLIB Function Table 10

..... 10
+ 10
- 10
* 10
/ 10
% 10
^ 10
<= 10
< 10
>= 11
> 11
== 11
!= 11
&& 11
|| 11
abs(expN) 11
ceil(expN) 12
floor(expN) 12
round(expN) 12
sin(expN) 12
cos(expN) 12
ln(expN) 12
exp(expN) 12
atan(expN) 12
sqrt(expN) 12
val(expS) 13
fxpval(expS, expN) 13
str(expN1, expN2, expN3) 13

iif(exp1, exp2, exp3) 13
stodt(expS) 13
stodtsql(expS) 13
stod(expS) 13
tstod(expS) 13
dtos(expD) 14
year(expD) 14
month(expD) 14
day(expD) 14
dim(expD) 14
wiy(expD) 14
wiyo(expD, expN) 14
date() 14
dateof(expD) 15
timeof(expD) 15
chgdateof(expD1, expD2) 15
chgtimeof(expD1, expD2) 15
gettimeinsecs(expD) 15
settimeinsecs(expD, expN) 15
upper(expS) 15
lower(expS) 16
proper(expS) 16
isnull(expS) 16
left(expS, expN) 16
right(expS, expN) 16
mid(expS, expN1, expN2) 16
dtosf(expD, expS) 16
format(exp_, expS) 16
true 17
yes 17
false 17
no 17

4. Formatting 18

Foreground Color 18
Background Color 18
COLORS IN GENERAL 18
Format Strings 18
New Style Format Strings with "!" prefix 19
String Format Strings 19
Number Format Strings 19
Number Format Strings (!# format) 19
Date Format Strings (all including !@ format) 19
Money Format String (!$ format) 21
Default Format Strings 21

5. Graphing 22

Overview 22
Creating Graphs 23
Graph Parameters 23
type 23
subtype 23
width 23
height 23
title 24
x_axis_title 24
y_axis_title 24
y_axis_title_right 24
Plot Parameters 24
axis 24
field 24

label 25
side 25
6. C API 26
rlib_init_with_environment 26
rlib_init 26
rlib_add_datasource_mysql 26
rlib_add_datasource_postgre 26
rlib_add_datasource_odbc 26
rlib_add_datasource_xml 26
rlib_add_query_as 26
rlib_add_report 27
rlib_add_report_from_buffer 27
rlib_set_output_format 27
rlib_set_output_format_from_text 27
rlib_execute 27
rlib_get_content_type 27
rlib_signal_connect 27
rlib_signal_connect_string 27
rlib_query_refresh 28
rlib_set_output_parameter 28
rlib_spool 28
rlib_get_output 28
rlib_get_output_length 28
rlib_add_parameter 28
rlib_free 28
rlib_add_resultset_follower 28
rlib_add_resultset_follower_n_to_1 29
rlib_version 29
rlib_set_output_encoding(**rlib *rlib_ptr**, const char *encoding) 29
rlib_set_datasource_encoding(**rlib *rlib_ptr**, char *input_name, const char *encoding) ..29
rlib_set_locale(**rlib *rlib_ptr**, const char *locale) 29
rlib_graph_add_bg_region(**rlib *rlib_ptr**,char *graph_name, char *region_label, char *color, float start, float end) 29
rlib_graph_clear_bg_region(**rlib *rlib_ptr**,char *graph_name) 29
rlib_graph_set_x_minor_tick(**rlib *rlib_ptr**,char *graph_name, char *x_value) 30
rlib_graph_set_x_minor_tick_by_location(**rlib *rlib_ptr**,char *graph_name, int location) 30
SAMPLE 30
7. PHP API 31
rlib_init 31
rlib_add_datasource_mysql 31
rlib_add_datasource_postgre 31
rlib_add_datasource_odbc 31
rlib_add_datasource_xml 31
rlib_add_query_as 31
rlib_add_report 31
rlib_add_report_from_buffer 32
rlib_set_output_format_from_text 32
rlib_execute 32
rlib_get_content_type 32
rlib_signal_connect 32
rlib_signal_connect_string 32
rlib_query_refresh 32
rlib_set_output_parameter 32
rlib_spool 33
rlib_add_parameter 33
rlib_free 33
rlib_add_resultset_follower 33
rlib_add_resultset_follower_n_to_1 33

rlib_version 33
rlib_set_output_encoding(**rlib**, encodingstring) 33
rlib_set_report_output_encoding(**rlib**, reportnumber, encodingstring) 33
rlib_set_datasource_encoding(**rlib**, input_name, encoding) 34
rlib_set_locale(**rlibr**, locale string) 34
rlib_graph_add_bg_region(**rlib *rlib_ptr**,char *graph_name, char *region_label, char *color, float start, float end) 34
rlib_graph_clear_bg_region(**rlib *rlib_ptr**,char *graph_name) 34
rlib_graph_set_x_minor_tick(**rlib *rlib_ptr**,char *graph_name, char *x_value) 34
rlib_graph_set_x_minor_tick_by_location(**rlib *rlib_ptr**,char *graph_name, int location) 34
SAMPLE 34
8. Examples **36**
Example 1 36
A. GNU Free Documentation License **39**
PREAMBLE 39
APPLICABILITY AND DEFINITIONS 39
VERBATIM COPYING 40
COPYING IN QUANTITY 40
MODIFICATIONS 41
COMBINING DOCUMENTS 42
COLLECTIONS OF DOCUMENTS 42
AGGREGATION WITH INDEPENDENT WORKS 42
TRANSLATION 43
TERMINATION 43
FUTURE REVISIONS OF THIS LICENSE 43
ADDENDUM: How to use this License for your documents 43
B. SWIG License **45**
LICENSE 45

Chapter 1. Introduction

RLIB in a nutshell

RLIB is a report generation library/language. It takes advantage of today's best web technology including PHP, SQL, and XML. Because speed is an extremely important with web applications, and requests must be completed with sub second response time, c was chosen as the language in which to implement RLIB.

One of the main advantages to RLIB is that you don't have to be a programmer in order to use it. The file format for describing reports is XML. RLIB supports full expression evaluation in human readable format so it is easy to follow the logic behind a report.

This manual assumes you have prior knowledge of XML and SQL, and also basic PHP. DON'T PANIC if you don't know what these technologies are. We will cover in detail the exact XML you are expected to know, and the PHP api to RLIB (which is all you need to know). We won't cover SQL queries, except for the examples given.

RLIB is released under the GNU General Public License (GPL). This means that you are free to modify, copy, and redistribute it as long as you adhere to the terms of the GPL. If you or your organization do not want to make your code free, an alternative commercial license is available. You can review the license at http://www.gnu.org/copyleft/gpl.html.

RLIB is open source. Because of that, you might take the time to review the code and possibly make changes to improve it. SICOM Systems would be happy to review your changes, and possibly include them in future releases of RLIB. To find out more please visit the "Community" section on the RLIB web site.

What is a Report Writer

There are many report writers that exist, however the basic principles are the same. Loop through data sets printing out records in a specified format. Also allow for intelligent page breaking, report headers and footers, page headers and footers, allow for data grouping and breaking for subtotal. Be able to define variables that operate on columns (sum, average) that may reset on breaks. Also allow report columns to be mathamatical equations supporting a variety of functions.

RLIB supports all of this and more.

This Manual

The RLIB manual will cover all aspects of RLIB programming. Chapter 2 will cover RLIB terminology. Chapter 3 will cover rlib XML file format. Chapter 4 will cover RLIB data types and functions. Chapter 5 will discuss the PHP api to RLIB.

Chapter 2. Basic Training

A little Terminology

In addition to being a report writer, RLIB is also an interpreted language (like Basic and FOX-PRO). RLIB handles all memory management for you. If you are familiar with FOXPRO, then you will feel at home with RLIB. However, if you don't know foxpro see the function reference (symbol table) to see all of the functions available to you. In a nutshell its pretty powerful. You can do stuff like this:

```
'hi' + ' ' + 'there' + database.stringfield
1+2*87+(7*2)/98.2/database.numberfield
dtos(database.somedatefield)-7
```

RLIB has 3 data types that it uses internally. STRING, DATE, and NUMBER. There are functions to turn datatypes into other datatypes, plus all sorts of other goodies. Almost all parameters in the RLIB xml file are expressions. This means that the parameters are computed on the fly. The result must be a STRING, DATE, or NUMBER (usually STRING or NUMBER). In our examples you will see stuff like expN, expS, expD. expN means number expressions, expS means a string expression, and expD means a date expression.

STRINGS are in single quotes

Please note that STRINGS in rlib are between single quotes. This is because XML uses double quotes.

lets take an example.

The round function is defined as round(expN). This means that it must be given a number to round. In rlib you an do the following:

```
round(7.2)
round(7.2+33)
round(val('7.2')*9/2)
```

As long as it gets passed a expN.. got it? Good. By the way, val(expS) takes a string and turns it into a number.

The DATE variable type

The RLIB DATE variables can hold either a date, a time or both. Mathematical operations on DATE variables are supported. When doing date/time arithmetic, the operations performed depend on whether the date component, time component or both are validly set to a value. For example, assuming date() returns a DATE variable and both time and date are set to 1/1/2004 at 10:31:45:

```
date() + 15
```

is 1/1/2004 at 10:32:00

```
chgdateof(date(), dateof(date()) + 15)
```

is 1/16/2004 at 10:31:45

```
timeof(date()) + 15
```

is 10:32:00

```
dateof(date()) + 15
```

is 1/16/2004

Fixed Point

Numerical accuracy is important in RLIB, therefore all mathematical calculations are done with fixed point using 8 decimal places of accuracy. As you might know, floating point operations in computing are inherently inaccurate. It is especially true if you are looping through data doing computations. Therefore, rlib uses FIXED POINT calculations internally in order to preserve accuracy. Note, all of this is internal to rlib, and you never have to worry about it when writing reports.

Data Sources

RLIB has many data sources; databases, environment variables, rlib internal variables, and rlib user variables.

The first query you give rlib is the main query. In the XML you can reference your fields as the field names without specifying the data source name ie (fld1+fld2). For the main data source, just put the field names in from the result set. You can still reference by the "RESULT SET NAME". Refer to the api "rlib_add_query_as".

Note

All Result set fields come into RLIB as STRINGS! you will have to frequently do VAL or FXPVAL. to turn them into numbers

All other queries are secondary queries. You **must** specify the data source name when you reference them. ie (result.fld1+result.fld2)

Environment variables are STRINGS coming from the environment that you can pick up to put on your report. If in php you had

```
$start_date = "2003-01-01";
```

In RLIB you reference thing as m.name. So you would have

```
m.start_date
```

rlib has a few internal variables like pageno, lineno, detailcnt (detail line count). Reference them with r.name. It would be r.pageno, r.lineno, and r.detailcnt in your code. RLIB has one more variable r.value, which is the result of a "FIELD"'s expression.. This is good so say in the color field, you DON'T HAVE TO RECALCULATE THE VALUE, you can use the "POINTER" to the value in order to color the field. like if it is < 0 make it red, else make it black....

Finally, rlib variables (discussed below) are referenced with a r.name, where name is the name you gave them.

So you could have an expression like this in rlib

```
str((val(m.some_environment_value)+r.some_sum_value*val(field1)),7,2) + ' hi'
```

Report Layout

Reports are laid out into logical sections.

Report Header - Appears at the top of the first page of a report. Report headers commonly have information such as organization name.

Page Header - Appears at the top of every page except the first page, where it appears below the Report Header. Page headers commonly list the report name.

Page Footer - Appears at the bottom of each page. Page number is commonly found on Page Footers.

Report Footer - Appears on the last page below the last element of data. Report Data Totals are found in the Report Footer

Data Lines - Appear for each row of data you have in your main result set.

Break Header(s) - A header record for your data breaks appear before the Data Lines that are in the break sub section.

Break Footers(s) - Appear below the data lines for the break sub section. Might include sub totals for the data subsection.

Breaks Explained

It is often necessary to group data on a report in logical sections. For example imagine a report with a data scope of a list of stores. The stores are part of a hierarchy. In this example the hierarchy could be 4 levels deep. ie: "District, Market, Region, Another Level". It would be wasteful to have 4 columns on a report showing the same information over and over again. If the names were 20 characters long we would have just wasted at least 80 columns on our report. Aside from wasting space, it is often necessary to produce sub totals on groups of data in a report. Breaks are good for this also. The following example is a report that is breaking on a hierarchy schema. There are count and amount subtotals for each break. It should also be noted here that RLIB knows how much room is left on a page. It will not start a break at the bottom of a page if at least 1 data line will fit. Instead it will end the page and start the break on the next page.

Breaks are defined using the **Break** tag. Breaks happen when the **BreakField** value changes. The break name is important form RLIB variables. The name can be given to the variable **resetonbreak** paramater.

```
<Breaks>
 <Break name="break0" newpage="no" headernewpage="yes">
  <BreakHeader>
   <Output>
    <Line>
     <field value="group" width="20" align="left" col="1"/>
    </Line>
   </Output>
  </BreakHeader>
  <BreakFields>
   <BreakField value="group"/>
  </BreakFields>
  <BreakFooter>
   <Output>
    <HorizontalLine size="4" bgcolor="'white'"/>
    <HorizontalLine size="2" bgcolor="'black'"/>
```

```
    <HorizontalLine size="4" bgcolor="'white'"/>
    <Line>
     <field value="group" width="20" align="left" col="1"/>
    </Line>
    <HorizontalLine size="4" bgcolor="'white'"/>
    <HorizontalLine size="2" bgcolor="'black'"/>
    <HorizontalLine size="4" bgcolor="'white'"/>
   </Output>
  </BreakFooter>
 </Break>
</Breaks>
```

Report Variables

Report Variables are are expressions rlib evaulates for every detail line, and resets automatically if asked to on breaks. Report variables are useful things like COUNT, SUM, AVG, HIGHEST, LOWEST. You may also use report variables to simplify expressions that will be used (normally more than once) in other calculations. To do this use the type EXPRESSION. Report variable can be "RESET" on the event of a BREAK (this does not apply to EXPRESSION). Here are some examples

```
<Variable name="customer_count" value="val(ei_count)+val(to_count)+val(dt_count)"
  type="expression"/>
<Variable name="daily_percent" value="v.daily_diff / v.this_year_netsales * 100"
  type="expression"/>
<Variable name="this_year_mtd" value="v.this_year_netsales"
  type="sum" resetonbreak="break5"/>
<Variable name="last_year_mtd" value="v.last_year_netsales"
  type="sum" resetonbreak="break5"/>
<Variable name="monthly_diff" value="v.this_year_mtd - v.last_year_mtd"
  type="expression"/>
<Variable name="monthly_percent" value="v.monthly_diff / v.this_year_mtd * 100"
  type="expression"/>
```

Defining a Report

RLIB reports are defined in the RLIB xml file. The XML file encompases all topics discussed above. Here is a sample XML file.

SAMPLE XML FILE

```
<?xml version="1.0"?>
<!DOCTYPE report >

<Report fontSize="9" orientation="landscape">
 <ReportHeader>
  <Output>
   <Image value="'logo.jpg'" type="'jpeg'" width="50" height="50"/>
   <Line/>
   <Line fontSize="12">
    <literal width="8"/>
    <field value="header.name" align="left" col="1"/>
   </Line>
   <Line fontSize="12">
    <literal width="8"/>
```

```
    <field value="header.name2" align="left" col="1"/>
   </Line>
   <Line/>
   <Line fontsize="4"/>
   <HorizontalLine size="4" bgcolor="'white'"/>
   <HorizontalLine size="2" bgcolor="'black'"/>
   <HorizontalLine size="4" bgcolor="'white'"/>
  </Output>
 </ReportHeader>

 <PageHeader>
  <Output>
   <Line fontSize="11">
    <field value="header.report_name" width="40" align="left" col="1"/>
   </Line>
   <HorizontalLine size="4" bgcolor="'white'"/>
  </Output>
 </PageHeader>

 <Detail>
  <FieldHeaders>
   <Output>
    <HorizontalLine size="1" bgcolor="'black'"/>
    <Line bgcolor="'0xe5e5e5'">
     <literal width="15" col="1">Number</literal>
     <literal width="1"/>
     <literal width="20" col="2">Name</literal>
     <literal width="1"/>
     <literal width="10" col="3">Type</literal>
     <literal width="1"/>
     <literal width="10" col="4">Category</literal>
    </Line>
    <HorizontalLine size="1" bgcolor="'black'"/>
    <HorizontalLine size="4" bgcolor="'white'"/>
   </Output>
  </FieldHeaders>
  <FieldDetails>
   <Output>
    <Line bgcolor="iif(r.detailcnt%2,'0xe5e5e5','white')">
     <field value="plunum" width="15" align="left" col="1"/>
     <literal width="1"/>
     <field value="name" width="20" align="left" col="2"/>
     <literal width="1"/>
     <field value="type" width="10" align="left" col="3"/>
     <literal width="1"/>
     <field value="category" width="10" align="left" col="4"/>
    </Line>
   </Output>
  </FieldDetails>
 </Detail>

 <PageFooter>
  <Output>
   <Line>
    <literal>Page: </literal>
    <field value="r.pageno" width="3" align="right"/>
   </Line>
  </Output>
 </PageFooter>

 <ReportFooter>
 </ReportFooter>
</Report>
```

Report

All Reports are made up of the Report tag.

fontSize - Font size .. 6-100

orientation - "portrait" or "landscape"

topMargin - how much space to leave at the top

leftMargin - how much space to leave at the left

bottomMargin - how much space to leave at the bottom

paperType - defines the type of paper you are targeting with your report. This field is a STRING Expression. values are: 'LETTER', 'LEGAL', 'A4', 'B5', 'C5', 'DL', 'EXECUTIVE', 'COMM10', 'MONARCH', and 'FILM35MM'

detail_columns - How many "columns" of detaul outputs will be printed per row of data. Useful for mailing labels.

xml_column_pad - Extra space between columns. Useful for mailing labels

iterations - How many times the report should be run. Useful if using callbacks the refresh the datasource when using fixed part layouts.

pagesAcross - How many pages across a report is (Normally 1). When using this feature you need to specify page=x in the <Output> tag.

suppressPageHeaderFirstPage - Suppress the page Header on the 1st Page.

Output

All main sections of a report contain output. Output is made up of Images, Lines, and Horizontal Lines.

HorizontalLine

bgcolor - background color

size - how tall the line is

indent - offset the line from the left margin (NUMBER VALUE.. number of characters)

length - the length of the line (NUMBER VALUE.. number of characters)

fontSize - part of the math on the length of and indent of a line. This is here for finer control because indent and length are given in number of characters.

Image

value - expS with name of file [REQUIRED]

type - expS with type of file (jpeg, gif) [REQUIRED]

width - expN width of image [REQUIRED]

height - expN with height of image [REQUIRED]

Line

Lines contain literals and fields.

fontSize - expN default font size for literals and fields on the line

color - expS default foreground color for literals and fields on the line

bgcolor - expS default background color for fields on the line

Literals

Contains plain text data. The actual displayed text goes between the tag
width - expN width of string
align - expS left, right, center...
color - expS default foreground color
col - expN column for csv output
bgcolor - expS background color
link - expS .. a url to link to
link - expS .. a url to link to

Field

Contains Data
value - EXPRESSION of any kind
col - expN column for csv output
width - expN width of string
align - expS left, right, center
format - expS see format section
color - expS foreground color
bgcolor - expS background color
link - expS .. a url to link to

ReportHeader

Contains Output

ReportFooter

Contains Output

PageHeader

Contains Output

PageFooter

Contains Output

Detail

Contains Field Headers and Field Details. Headers are essentially the column headers. Field details are the lines repeated over and over again.

Variables

Variables are used to manuiplate data on the floor and save it. Variables can sum, count, average, or can be simple expressions. Variables have a name, a type and optionally when they should reset their value.

name - expS the name of the variable, referenced in expressions as v.name [REQUIRED]

value - Any expression. If it is a sum, average, count it must be a expN

resetonbreak - name of break that the variable should reset

Breaks

Breaks are hierarchial groupings of data on a report.

name - expS the name of the variable, referenced in expressions as v.name [REQUIRED]

value - Any expression If it is a sum, average, count it must be a expN

newpage = [yes/no] No is default should the break start on a new page

headernewpage = [yes/no] Yes is default. The break header will always appear on the top of every page

breaks have BreakHeader and BreakFooter and both contain output. ONE NICE FEATURE IN RLIB IS THAT WHEN A BREAK EVENT HAPPENS THE OUTPUT will be done while on current row (FOR FOOTER AND HEADER). This makes subtotals much easier!

Breaks also have BreakFields. An expression on how it breaks. There can be more then one and it has a value. The value is an expression.

Chapter 3. RLIB Function Table

In the definitions below; expN means a number expressions, expS means a string expression, and expD means a date expression.

+

expN1 + expN2 **Returns** NUMBER containing the result of expN1 plus expN2

expD + expN or expN + expD **Returns** DATE contains the result of expD + expN seconds or days. If both datetimes have valid time, the result is expressed in SECONDS. Otherwise the result returned is in DAYS.

expS1 + expS2 **Returns** STRING containing the result of the concatenation of expS1 and expS2

-

expN1 - expN2 **Returns** NUMBER contain the result of expN1 minus expN2

expD - expN **Returns** if DATE contains a valid time portion then the result is expD - expN SECS. Otherwise it returns expD - expN DAYS.

expD - expD **Returns** NUMBER containing the number of SECONDS or DAYS difference in the dates. If the time portion is valid, the result is the number of SECONDS difference. If both times and both dates are valid the seconds count will include the difference in the days. Otherwise if the date portions are valid it returns the number of DAYS difference.

*

expN1 * expN2 **Returns** NUMBER containing the result of expN1 times expN2

/

expN1 / expN2 **Returns** NUMBER containing the result of expN1 divided by expN2

%

expN1 % expN2 **Returns** NUMBER containing the result of expN1 mod expN2

^

expN1 ^ expN2 **Returns** NUMBER containing the result of expN1 to the expN2

<=

expN1 <= expN1 **Returns** NUMBER 1 if true.. 0 if false

expS1 <= expS2 **Returns** NUMBER 1 if true if expS1 is alphabetically less then expS2.. 0 if false

expD1 <= expD2 **Returns** NUMBER 1 if true if expD1 is less then expD2.. 0 if false. Both date and time are compared if both have valid components. The dateof and timeof functions can be used to select one or the other.

<

expN1 < expN2 **Returns** NUMBER 1 if true.. 0 if false

expS1 < expS2 **Returns** NUMBER 1 if true.. 0 if false

expD1 < expD2 **Returns** NUMBER 1 if true.. 0 if false Both date and time are compared if both have valid components. The dateof and timeof functions can be used to select one or the other.

>=

expN1 >= expN2 **Returns** NUMBER 1 if true.. 0 if false

expS1 >= expS2 **Returns** NUMBER 1 if true.. 0 if false

expD1 >= expD2 **Returns** NUMBER 1 if true.. 0 if false Both date and time are compared if both have valid components. The dateof and timeof functions can be used to select one or the other.

>

NUMBER > NUMBER **Returns** NUMBER 1 if true.. 0 if false

STRING > STRING **Returns** NUMBER 1 if true.. 0 if false

expD1 > expD2 **Returns** NUMBER 1 if true.. 0 if false Both date and time are compared if both have valid components. The dateof and timeof functions can be used to select one or the other.

==

NUMBER == NUMBER **Returns** NUMBER 1 if true.. 0 if false

STRING == STRING **Returns** NUMBER 1 if true.. 0 if false

expD1 == expD2 **Returns** NUMBER 1 if true.. 0 if false Both date and time are compared if both have valid components. The dateof and timeof functions can be used to select one or the other.

!=

NUMBER != NUMBER **Returns** NUMBER 1 if true.. 0 if false

STRING != STRING **Returns** NUMBER 1 if true.. 0 if false

expD1 != expD2 **Returns** NUMBER 1 if true.. 0 if false Both date and time are compared if both have valid components. The dateof and timeof functions can be used to select one or the other.

&&

LOGICAL "AND". iif(m.test1 == 22 && m.test2 == 17, 'passed', 'failed')

NUMBER && NUMBER **Returns** NUMBER 1 if true.. 0 if false

STRING && STRING **Returns** NUMBER 1 if true.. 0 if false .. True if both strings are not null

||

NUMBER || NUMBER **Returns** NUMBER 1 if true.. 0 if false

STRING || STRING **Returns** NUMBER 1 if true.. 0 if false . True if at least 1 of the strings is not null

abs(expN)

Absolute Value.

Returns NUMBER

ceil(expN)

Round up to the nearest integer.

Returns NUMBER

floor(expN)

Round down to the nearest integer.

Returns NUMBER

round(expN)

Round to nearest integer

Returns NUMBER

sin(expN)

The sin() function Returns the sine of expN, where expN is given in radians.

Returns NUMBER

cos(expN)

The cos() function Returns the cosine of expN, where expN is given in radians.

Returns NUMBER

ln(expN)

The ln() function Returns the natural logarithm of of expN.

Returns NUMBER

exp(expN)

The exp() function Returns the value of e (the base of natural logarithms) raised to the power of expN

Returns NUMBER

atan(expN)

The atan() function calculates the arc tangent of expN; that is the value whose tangent is expN.

Returns NUMBER

sqrt(expN)

The sqrt() function Returns the non-negative square root of expN. Don't pass it a negative number or it will fail I think

Returns NUMBER

val(expS)

Converts a string into a number.. respects the decimal point in a string

Returns NUMBER

fxpval(expS, expN)

Converts a string into a number.. assumes no decimal places in the string.. expN says where the decimal place should go for instance if in your database you store all values in cents.. and not dollars

Returns NUMBER

str(expN1, expN2, expN3)

Convert a NUMBER into a STRING of length expN2 with decimal precision of expN3

Returns STRING

iif(exp1, exp2, exp3)

In-line If. Basically exp1 is evaluated. If it is true (NUMBER != 0, STRING != NULL, DATE is always true).. exp2 is evaluated and returned otherwise exp3 is evaluated and returned

Returns exp2 if TRUE... exp3 if FALSE

stodt(expS)

Converts a string into a date time.. must be in the format of YYYYMMDDHHMMSS (format sql timestamp type)

Returns DATE

stodtsql(expS)

Converts a string into a date time.. must be in the format of YYYY-MM-DD HH:MM:SS (format sql datetime type)

Returns DATE

stod(expS)

Converts a string into a date.. must be int he format of YYYY-MM-DD (format sql date type)

Returns DATE

tstod(expS)

expS is a time in string format such as HH:MM, HH:MM:SS, HH:MMp, HH:MM:SSp, HHMM, HHMMSS, HHMMp, HHMMSSp. This function will return a DATE value with the time set to the time in the string (the date part of the DATE variable is set to 1/1/1980).

Returns DATE

dtos(expD)

Converts a date into a string. The string will be YYYY-MM-DD

Returns STRING

year(expD)

Returns YYYY as NUMBER

Returns NUMBER

month(expD)

Returns MM as NUMBER

Returns NUMBER

day(expD)

Returns DD as NUMBER

Returns NUMBER

dim(expD)

Returns NUMBER containing the day in the month (1-31)

Returns NUMBER

wiy(expD)

Returns NUMBER containing the week number of the year. Range 00 to 53, starting with the first Sunday as the first day of week 01

Returns NUMBER

wiyo(expD, expN)

Returns NUMBER containing the week number of the year. Range 00 to 53, starting with the first expN as the first day (1=Monday, 2=Tuesday...)

Returns NUMBER

date()

Returns the current date time in RLIB DATE Format. This value is set to a constant when report generation starts and remains unchanged for the duration of the report. **This is a change from previous versions of RLIB which would reset this value to the current time on each invocation.** Locking this value prevents the current date/time from changing from page to page when the current date/time is repeated on multiple pages.

Returns DATE

dateof(expD)

A function that converts a datetime to a date only variable. Use this function with comparisons and DATE arithmetic to select only the DATE portion for use in the expression.

Returns DATE

timeof(expD)

A function that converts a datetime to a time only variable. Use this function with comparisons and TIME arithmetic to select only the TIME portion for use in the expression.

Returns DATE

chgdateof(expD1, expD2)

Changes the date portion of expD1 to equal that of expD2.

Returns DATE

chgtimeof(expD1, expD2)

Changes the time portion of expD1 to equal that of expD2.

Returns DATE

gettimeinsecs(expD)

Returns the time portion of the datetime variable as the number of seconds past 00:00:00

Returns NUMBER

This function in conjunction with settimeinsecs can be used to perform mathematical calculations on time.

settimeinsecs(expD, expN)

Returns a datetime with the time portion of the datetime changed to a value obtained by adding expN seconds to 00:00:00. For example: "settimeinsecs(expD, (gettimeinsecs(expD) / 3600) * 3600)" returns a datetime that has been truncated to an even hour value.

Returns DATE

see settimeinsecs function for details and example.

upper(expS)

Returns expS as a upper case string

Returns STRING

lower(expS)

Returns expS as a lower case string

Returns STRING

proper(expS)

Returns expS as a proper string.. ie 1str char caps.. all the rest lower

Returns STRING

isnull(expS)

Returns expN 1 if expS is null or expN 0 if expS is not null

Returns NUMBER

left(expS, expN)

Returns the leftmost expN characters of the string expS.

Returns STRING

right(expS, expN)

Returns the rightmost expN characters of the string expS.

Returns STRING

mid(expS, expN1, expN2)

expN1 is an index into the string expS starting at 0. expN2 is the maximum length of the result string.

Returns STRING

dtosf(expD, expS)

DEPRECATED! This function may be removed in future versions of RLIB. Please use the 'format' function instead.

Convert date expD to string using the specified format expS. expS uses the formatting controls listed below and can be used to display either the date or time of the DATE variable.

Returns STRING

format(exp_, expS)

Formats the passed exp_ (may be expD, expS or expN) using the expS parameter as a format string. Format strings **must** be in the new '!' format.

This function will format the passed variable using the passed format string. Character strings may be formatted using the format symbol '!!'. Errors will be generated if the type for the format string does not match the variable type. format headers are: '!!' for strings, '!@' for datetimes, '!#' for numbers and '!$' for currency.

For example: "format('xyx', '!!%6s')" prints ' xyz'.

Returns STRING

true

A predefined NUMBER variable with the value 1.

Returns NUMBER

yes

A predefined NUMBER variable with the value 1.

Returns NUMBER

false

A predefined NUMBER variable with the value 0.

Returns NUMBER

no

A predefined NUMBER variable with the value 0.

Returns NUMBER

Chapter 4. Formatting

Foreground Color

color = "exprS"

exprS must be either a valid color name or hex color triplet ie 0xFFFFFF.

Use the forground color to change the color for literals and fields

Background Color

bgcolor = "exprS"

exprS must be either a valid color name or hex color triplet ie 0xFFFFFF.

Use the background color to change the color for literals and fields

COLORS IN GENERAL

Colors must be a STRING an specified as either a named color. See chart for color names.

OR as a hex color tripplet like 0xFFFFFF

so in the XML it would be either color="'0xFFFFFF'" OR color="'red'"

Format Strings

Yup RLIB has them. They are very similar to C. Numbers, Dates, or Strings. Here is the neat part. If you mess up format strings, rlib will put an error message in the field. They are...

!ERR_F means that rlib was not given a format string and can't automatically make something of your data

!ERR_F_D means you asked rlib to format as a number but it was not given a number data type

!ERR_F_S means you asked rlib to format as a string but it was not given a string data type

!ERR_F_F means you asked rlib could not interperate your format string expression / or what it interpreted it to was not a string

It works a lot like c. You can do stuff like "'You have %d apples'"

If you do something like %$5.2d this means put commas in so you will get 12,345.67

New Style Format Strings with "!" prefix

The new style format strings can be intermixed with old style formatting and may be used anywhere a format string is needed. They provide locale aware formatting for date/time, money and numbers. The first 2 characters in the new format strings must begin with either "!@", "!$" or "!#", respectively. Following this is an appropriate 'C' style format string for one of the functions strftime, strfmon, or sprintf (for numbers). All numerics in the money and number format strings must be represented using the 'e', 'f' or 'g' format codes. For example "!#$.2g", "!$%n", "!@%m/%d/%Y" are valid format strings using the new style for a number, a money amount and a date. There are also additional error codes added as follows:

!ERR_DT_D means you asked rlib to format a date but the date field of the datetime is not set.

!ERR_DT_T means you asked rlib to format a time but the time field of the datetime is not set.

!ERR_DT_NO means there were no valid format codes in a format string for a date/time.

String Format Strings

Exactly like c. %[optional number]s where optional number is how big to make the string

Number Format Strings

Close to c. %[optional number 1][.][optional number 2]d ... where optional number 1 is how big should the left side be and optional number 2 is how many decimal places...

Number Format Strings (!# format)

Identical to C. All numbers must be represented by the e, f or g format types.

Please refer to your systems printf formatting codes.

Date Format Strings (all including !@ format)

Date codes and time codes should be consecutive, i.e. don't have a timecode a datecode and then another timecode. This will not work. Rlib internally splits the datetime string into a date format string and a time format string. It uses the first transition from date-to-time or time-to-date as the split point for the date/time.

Dates from 1/1/1 through 1/1/8000+ can be represented.

%a The abbreviated weekday name according to the current locale.

%A The full weekday name according to the current locale.

%A The full weekday name according to the current locale.

%b The abbreviated month name according to the current locale.

%B The full month name according to the current locale.

%c The preferred date and time representation for the current locale.

%C The century number (year/100) as a 2-digit integer. (SU)

%d The day of the month as a decimal number (range 01 to 31).

%D Equivalent to %m/%d/%y. (Yecch - for Americans only. Americans should note that in other coun- tries %d/%m/%y is rather common. This means that in international context this format is ambigu- ous and should not be used.) (SU)

%e Like %d, the day of the month as a decimal number, but a leading zero is replaced by a space. (SU)

%E Modifier: use alternative format, see below. (SU)

%F Equivalent to %Y-%m-%d (the ISO 8601 date format). (C99)

%G The ISO 8601 year with century as a decimal number. The 4-digit year corresponding to the ISO week number (see %V). This has the same format and value as %y, except that if the ISO week num- ber belongs to the previous or next year, that year is used instead. (TZ)

%g Like %G, but without century, i.e., with a 2-digit year (00-99). (TZ)

%h Equivalent to %b. (SU)

%H The hour as a decimal number using a 24-hour clock (range 00 to 23).

%I The hour as a decimal number using a 12-hour clock (range 01 to 12).

%j The day of the year as a decimal number (range 001 to 366).

%k The hour (24-hour clock) as a decimal number (range 0 to 23); single digits are preceded by a blank. (See also %H.) (TZ)

%l The hour (12-hour clock) as a decimal number (range 1 to 12); single digits are preceded by a blank. (See also %I.) (TZ)

%m The month as a decimal number (range 01 to 12).

%M The minute as a decimal number (range 00 to 59).

%n A newline character. (SU)

%O Modifier: use alternative format, see below. (SU)

%p Either ‘AM’ or ‘PM’ according to the given time value, or the corresponding strings for the cur- rent locale. Noon is treated as ‘pm’ and midnight as ‘am’.

%P Like %p but in lowercase: ‘am’ or ‘pm’ or a corresponding string for the current locale. (GNU)

%r The time in a.m. or p.m. notation. In the POSIX locale this is equivalent to ‘%I:%M:%S %p’. (SU)

%R The time in 24-hour notation (%H:%M). (SU) For a version including the seconds, see %T below.

%s The number of seconds since the Epoch, i.e., since 1970-01-01 00:00:00 UTC. (TZ)

%S The second as a decimal number (range 00 to 61).

%t A tab character. (SU)

%T The time in 24-hour notation (%H:%M:%S). (SU)

%u The day of the week as a decimal, range 1 to 7, Monday being 1. See also %w. (SU)

%U The week number of the current year as a decimal number, range 00 to 53, starting with the first Sunday as the first day of week 01. See also %V and %W.

%V The ISO 8601:1988 week number of the current year as a decimal number, range 01 to 53, where week 1 is the first week that has at least 4 days in the current year, and with Monday as the first day of the week. See also %U and %W. (SU)

%w The day of the week as a decimal, range 0 to 6, Sunday being 0. See also %u.

%W The week number of the current year as a decimal number, range 00 to 53, starting with the first Monday as the first day of week 01.

%x The preferred date representation for the current locale without the time.

%X The preferred time representation for the current locale without the date.

%y The year as a decimal number without a century (range 00 to 99).

%Y The year as a decimal number including the century.

%z The time-zone as hour offset from GMT. Required to emit RFC822-conformant dates (using "%a, %d %b %Y %H:%M:%S %z"). (GNU)

%Z The time zone or name or abbreviation.

%+ The date and time in date(1) format. (TZ)

%% A literal '%' character.

Money Format String (!$ format)

The money formatter is locale aware and will use appropriate symbols and formats for the designated locale. It is a string in the form: %[=f ^ (+ ! -][fieldwidth][#leftprecision][.rightprecision][n or i] [] indicates optional sections.

=f f is a character to use as the numeric fill character. Default is ' '.

^ ignore grouping if specified in the locale. This is usually thousands groupings.

(Put negative amounts in ().

+ Show + sign on positive numbers.

! Omit the currency symbol.

- Left justify all fields.

n Display in national format. Like: $1.25

i Display in international format. Like USD 1.25

%% represents a % sign within the format specification string.

Default Format Strings

For STRING it is "%s"

for NUMBER it is "%d"

for DATE it is "%m/%d/%Y"

Chapter 5. Graphing

Overview

Rlib support for graphing is enabled via the **Graph** entity. Graphs are defined by, among other parameters, **type** and **subtype**. Multiple plot entities (lines,bars, etc.) are distinguished by unique colors from a pre-determined color set.

The following is a list of supported types with corresponding sub-types:

- Line

  type="'line'"

  Sub-Types:

  - subtype="'normal'"

    Normal: Standard line graph.

  - subtype="'percent'"

    Percent: The data points are presented as a percentage of the total.

  - subtype="'stacked'"

    Stacked: Similiar to the Percent graph except the Stacked Plot presents each data point as a proportion of the total.

- Row (Bar)

  type="'row'"

  Sub-Types:

  - subtype="'normal'"

    Normal: Standard Bar Chart.

  - subtype="'percent'"

    Percent: The data points are presented as a percentage of the total.

  - subtype="'stacked'"

    Stacked: Similiar to the Percent graph except the Stacked Plot presents each data point as a proportion of the total.

- Pie

  type="'pie'"

  Sub-Types:

- subtype="′normal′"
  Normal: Stanard Pie Chart.

- subtype="′offset′"
  Offset: Pieces of the "pie" are not connected to each other.

For both Line and Bar type charts, Double Y-Axis plots (the ability to use different scales for the left and right Y-Axis) are available.

Creating Graphs

Graphs are created through the use of the <Graph> and <Plot> tags. The <Graph> tags define the graph itself and <Plot> tag defines the data for the graph.

<Graph param1 param2 ... paramN >

<Plot param1 param2 ... paramN />

.

.

<Plot param1 param2 ... paramN />

</Graph>

Graph Parameters

<Graph type="′row′" subtype="′normal′" width="370" height="250" title="′This is the title !!!′" x_axis_title="′X-Axis Label′" y_axis_title="′Y-Axis Label′" y_axis_title_right="′Right Hand Y-Axis Label′" >

The following parameters define the Graph:

type

Sets the graph type. This is a **required** parameter.

Usage: type="′row′"

subtype

Sets the graph sub-type. This is a **required** parameter.

Usage: subtype="′normal′"

width

Sets the graph width. This is a **required** parameter.

Usage: width="370"

height

Sets the graph height. This is a **required** parameter.

Usage: height="250"

title

Sets the graph title. This is an **optional** parameter.

Usage: title="'This is a title'"

x_axis_title

Labels the x-axis. This is an **optional** parameter.

Usage: title="'This the X-Axis'"

y_axis_title

Labels the y-axis. This is an **optional** parameter.

Usage: title="'This the Y-Axis'"

y_axis_title_right

Labels the right hand side y-axis. This is an **optional** parameter.

Usage: title="'This the right hand Y-Axis'"

Plot Parameters

<Plot axis="'y'" field="val(sales)" label="'Sales Data" side="'left'" />

The data input to a graph is the result of a valid data source. The following parameters define the Graph's data:

axis

Defines the data for the specified **axis**.

- x and y are the only valid choices.
- Only one X-Axis <Plot /> tag may be defined.
- Multiple Y-Axis <Plot /> tags may be defined.

This is a **required** parameter.

Usage: axis="'x'"

field

Defines the data set to be plot on the indicated axis. Inputs must be a numerical data type. This is an **required** parameter.

Usage: field="val(sales)"

label

Defines the label to be used on the Graph legend. This is an **optional** parameter.

Usage: label="'SALES'"

side

Defines which side the y-axis data set belongs. Valid choices are left or right (Default is left). This is an **optional** parameter. Required when using double Y-Axis plots.

Usage: side="'left'"

Chapter 6. C API

rlib_init_with_environment

rlib * rlib_init_with_environment(struct environment_filter *environment)

Create an instance of RLIB. You will normally pass NULL to init. However.. at some point we need to better document what the environment_filter does... this is normally used for 3rd part bindings like PHP, PERL, or PYTHON.

Returns a pointer to a **rlib**

rlib_init

rlib * rlib_init()

calls rlib_init_environment with a NULL pointer

Returns a pointer to a **rlib**

rlib_add_datasource_mysql

rlib_add_datasource_mysql(**rlib * rlib_ptr**, char *datasource_name, char *hostname, char *username, char *password, char *database)

Add a mysql datasource. The datasource_name is used in rlib_add_query_as to tell rlib which datasource to run the query with.

This function is only available if rlib is compiled with mysql support.

rlib_add_datasource_postgre

rlib_add_datasource_postgre(**rlib * rlib_ptr**, char *datasource_name, char *connection_string)

Add a postgre datasource. The datasource_name is used in rlib_add_query_as to tell rlib which datasource to run the query with. The connection_string is the standard postgre connection string.. which might contain user and password information, among other things.

This function is only available if rlib is compiled with postgre support.

rlib_add_datasource_odbc

rlib_add_datasource_odbc(**rlib * rlib_ptr**, char *datasource_name, char *user_name, char *password)

Add a odbc datasource. The datasource_name is used in rlib_add_query_as to tell rlib which datasource to run the query with. The user name and password are that of your database you are connecting to

This function is only available if rlib is compiled with odbc support.

rlib_add_datasource_xml

rlib_add_datasource_xml(**rlib * rlib_ptr**, char *datasource_name)

Adds a XML datasource to RLIB. Pass XML files to rlib instead of queries.

rlib_add_query_as

rlib_add_query_as(**rlib * rlib_ptr**, char *datasource_name, char *query, char *rlib_query_name)

The query is added to an execution queue, but it is not executed at this time. The name is important because you can reference result sets directly in your rlib xml files. The first query added is assumed to be the main loop query. The datasource name must match a datasource the your provided rlib, such as one of the mysql or postgre datasources.

rlib_add_report

rlib_add_report(**rlib *rlib_ptr**, char *rlib_xml_file)

A report is added to the report execution queue but not compiled at this time.

rlib_add_report_from_buffer

rlib_add_report_from_buffer(**rlib *rlib_ptr**, char *xml_in_memory)

A report is added to the report execution queue but not compiled at this time.

rlib_set_output_format

rlib_set_output_format(**rlib *rlib_ptr**, int type)

Type can be one of the following: RLIB_FORMAT_PDF, RLIB_FORMAT_HTML, RLIB_FORMAT_TXT, RLIB_FORMAT_CSV

rlib_set_output_format_from_text

rlib_set_output_format_from_text(**rlib *rlib_ptr**, char *name)

Type can be one of the following: "pdf", "html", "csv", "txt"

rlib_execute

rlib_execute(**rlib *rlib_ptr**)

Connects to the database, runs queries, compiles xmls and buffers up a report.

rlib_get_content_type

char *rlib_get_content_type(**rlib *rlib_ptr**)

This will return a string content type Use it with the **php header function**. Even if you ask for a PDF you might not get a PDF because errors might occur. If this is the case, rlib defaults to html and sends out error messages.

rlib_signal_connect

rlib_signal_connect(**rlib * rlib_ptr**, int signal_number, int (*signal_function)(rlib *, void *), void * data);

Connect a user callback function to a RLIB Signal. This is usally to manuipulate data and then refresh the datasource.

rlib_signal_connect_string

rlib_signal_connect(**rlib * rlib_ptr**, char *signal_name, int (*signal_function)(rlib *, void *), void * data);

Connect a user callback function to a RLIB Signal. This is usally to manuipulate data and then refresh the datasource.

rlib_query_refresh

rlib_query_refresh(**rlib * rlib_ptr**);

Causes RLIB to refresh all queries and put the row pointer back at the 1st row.

rlib_set_output_parameter

rlib_set_output_parameter(**rlib * rlib_ptr**, char *parameter, char *value);

Set an output paramteter. SEE OUTPUT PARAMETERS.

rlib_spool

rlib_spool(**rlib * rlib_ptr**)

Rlib will send the output out stdout.

rlib_get_output

char * rlib_get_output(**rlib * rlib_ptr**)

Returns the output buffer (COULD BE NON NULL TERMINATED STRING)

rlib_get_output_length

long rlib_get_output_length(**rlib * rlib_ptr**)

Returns the length of the output buffer

rlib_add_parameter

int rlib_add_parameter(**rlib *rlib_ptr**, const char *name, const char *value)

Adds the name/value pair to the memory parameters. Values added in this manner supercede values passed in the environment. The names are searched in a case sensitive manner. Both name and value are stored by value, so the passed arguments do not need to persist after the call.

rlib_free

rlib_free(**rlib *rlib_ptr**)

Free rlib's memory that it allocated

rlib_add_resultset_follower

rlib_add_resultset_follower(**rlib *rlib_ptr**, char *leader, char *follower)

Adds the ability to have more then one main loop query. leader and follower are the names of the queries you set in rlib_add_query_as.

rlib_add_resultset_follower_n_to_1

rlib_add_resultset_follower(**rlib *rlib_ptr**, char *leader, char *leader_field, char *follower, char *follower_field)

Adds the ability to have more then one main loop query. leader and follower are the names of the queries you set in rlib_add_query_as. The leader_fields and follower_fields are normal RLIB expressions. Use this feature when the rows in the datasets are uneven and needs to be linked together using a common criteria.

rlib_version

char *rlib_version(void);

Returns a string containing the version of RLIB being used.

rlib_set_output_encoding(rlib *rlib_ptr, const char *encoding)

Sets the output character encoding, overriding any encoding that is set in the current Locale. By default RLIB will use the character encoding indicated in the current Locale settings. All reports will use this encoding unless overriden by a call to rlib_set_report_output_encoding. If the encoding is NULL, or a null string, the output will be left in UTF-8 encoding.

rlib_set_datasource_encoding(rlib *rlib_ptr, char *input_name, const char *encoding)

Sets the output character encoding, overriding any encoding that is set in the current Locale. By default RLIB will use the character encoding indicated in the current Locale settings. All reports will use this encoding unless overriden by a call to rlib_set_report_output_encoding. If the encoding is NULL, or a null string, the output will be left in UTF-8 encoding.

rlib_set_locale(rlib *rlib_ptr, const char *locale)

Sets the locale to the passed locale. The locale must be one of the values returned by the shell command: **locale -a**.

Returns true if the locale was successfully set.

rlib_graph_add_bg_region(rlib *rlib_ptr,char *graph_name, char *region_label, char *color, float start, float end)

Add a region of color with a label to a graph. Useful when the X axis is time and you want to indicate an event occured over that time.

rlib_graph_clear_bg_region(rlib *rlib_ptr,char *graph_name)

Remove regions. (Use if you are iterating graphs and have different X axis values)

rlib_graph_set_x_minor_tick(rlib *rlib_ptr,char *graph_name, char *x_value)

Set a x value on a graph as a minor tick

rlib_graph_set_x_minor_tick_by_location(rlib *rlib_ptr,char *graph_name, int location)

Set a x value on a graph as a minor tick. Location starts at 0.

SAMPLE

Here is a example.

```
#include <rlib.h>

char *query ="SELECT * FROM plu";
rlib *r;

r = rlib_init();
rlib_add_datasource_mysql(r, "mysql", "localhost", "user", "password", "database");
rlib_add_query_as(r, "mysql", query, "woot");
rlib_add_report(r, "report.xml");
rlib_set_output_format(r, $format);
rlib_execute(r);
rlib_spool(r);
rlib_free(r);
```

Chapter 7. PHP API

rlib_init

rlib_init()

Start RLIB

Returns a pointer to a **rlib**

rlib_add_datasource_mysql

rlib_add_datasource_mysql(**rlib**, datasource_name, hostname, username, password, database)

Add a mysql datasource. The datasource_name is used in rlib_add_query_as to tell rlib which datasource to run the query with.

This function is only available if rlib is compiled with mysql support.

rlib_add_datasource_postgre

rlib_add_datasource_postgre(**rlib**, datasource_name, connection_string)

Add a postgre datasource. The datasource_name is used in rlib_add_query_as to tell rlib which datasource to run the query with. The connection_string is the standard postgre connection string.. which might contain user and password information, among other things.

This function is only available if rlib is compiled with postgre support.

rlib_add_datasource_odbc

rlib_add_datasource_odbc(**rlib**, datasource_name, user_name, password)

Add a odbc datasource. The datasource_name is used in rlib_add_query_as to tell rlib which datasource to run the query with. The user name and password are that of your database you are connecting to

This function is only available if rlib is compiled with odbc support.

rlib_add_datasource_xml

rlib_add_datasource_xml(**rlib * rlib_ptr**, datasource_name)

Adds a XML datasource to RLIB. Pass XML files to rlib instead of queries.

rlib_add_query_as

rlib_add_query_as(**rlib**, datasource_name, query, rlib_query_name)

The query is added to an execution queue, but it is not executed at this time. The name is important because you can reference result sets directly in your rlib xml files. The first query added is assumed to be the main loop query. The datasource name must match a datasource the your provided rlib, such as one of the mysql or postgre datasources.

rlib_add_report

rlib_add_report(**rlib**, rlib_xml_file)

A report is added to the report execution queue but not compiled at this time.

rlib_add_report_from_buffer

rlib_add_report_from_buffer(**rlib**, xml_in_memory)

rlib_set_output_format_from_text

rlib_set_output_format_from_text(**rlib**, type)

Type can be one of the following: html, pdf, txt, or csv.

rlib_execute

rlib_execute(**rlib**)

Connects to the database, runs queries, compiles xmls and buffers up a report.

rlib_get_content_type

rlib_get_content_type(**rlib**)

This will return a string content type Use it with the **php header function**. Even if you ask for a PDF you might not get a PDF because errors might occur. If this is the case, rlib defaults to html and sends out error messages.

rlib_signal_connect

rlib_signal_connect(**rlib * rlib_ptr**, int signal_number, callback_function);

Connect a user callback function to a RLIB Signal. This is usally to manuipulate data and then refresh the datasource.

rlib_signal_connect_string

rlib_signal_connect(**rlib * rlib_ptr**, signal_name, callback_function);

Connect a user callback function to a RLIB Signal. This is usally to manuipulate data and then refresh the datasource.

rlib_query_refresh

rlib_query_refresh(**rlib * rlib_ptr**);

Causes RLIB to refresh all queries and put the row pointer back at the 1st row.

rlib_set_output_parameter

rlib_set_output_parameter(**rlib * rlib_ptr**, char *parameter, char *value);

Set an output paramteter. SEE OUTPUT PARAMETERS.

rlib_spool

rlib_spool(**rlib**)

Rlib will send the output out stdout.

rlib_add_parameter

int rlib_add_parameter(**rlib**, namestring, valuestring)

Adds the name/value pair to the memory parameters. Values added in this manner supercede values passed in the environment. The names are searched in a case sensitive manner.

rlib_free

rlib_free(**rlib**)

Free rlib's memory that it allocated

rlib_add_resultset_follower

rlib_add_resultset_follower(**rlib**, leader, follower)

Adds the ability to have more then one main loop query. leader and follower are the names of the queries you set in rlib_add_query_as.

rlib_add_resultset_follower_n_to_1

rlib_add_resultset_follower(**rlib**, leader, leader_field, follower, follower_field)

Adds the ability to have more then one main loop query. leader and follower are the names of the queries you set in rlib_add_query_as. The leader_fields and follower_fields are normal RLIB expressions. Use this feature when the rows in the datasets are uneven and needs to be linked together using a common criteria.

rlib_version

rlib_version()

Returns a string containing the RLIB version number or the libarary.

rlib_set_output_encoding(rlib, encodingstring)

Sets the output character encoding, overriding any encoding that is set in the current Locale. By default RLIB will use the character encoding indicated in the current Locale settings. All reports will use this encoding unless overriden by a call to rlib_set_report_output_encoding. If the encoding is NULL, or a null string, the output will be left in UTF-8 encoding.

rlib_set_report_output_encoding(rlib, reportnumber, encodingstring)

Sets the output character encoding for the indicated report. This setting will override the default setting. If this is not set or encoding is NULL or a null string, the **default rlib encoding is used**.

rlib_set_datasource_encoding(rlib, input_name, encoding)

Sets the output character encoding, overriding any encoding that is set in the current Locale. By default RLIB will use the character encoding indicated in the current Locale settings. All reports will use this encoding unless overriden by a call to rlib_set_report_output_encoding. If the encoding is NULL, or a null string, the output will be left in UTF-8 encoding.

rlib_set_locale(rlibr, locale string)

Sets the locale to the passed locale. The locale must be one of the values returned by the shell command: **locale -a**.

Returns true if the locale was successfully set.

rlib_graph_add_bg_region(rlib *rlib_ptr,char *graph_name, char *region_label, char *color, float start, float end)

Add a region of color with a label to a graph. Useful when the X axis is time and you want to indicate an event occured over that time.

rlib_graph_clear_bg_region(rlib *rlib_ptr,char *graph_name)

Remove regions. (Use if you are iterating graphs and have different X axis values)

rlib_graph_set_x_minor_tick(rlib *rlib_ptr,char *graph_name, char *x_value)

Set a x value on a graph as a minor tick

rlib_graph_set_x_minor_tick_by_location(rlib *rlib_ptr,char *graph_name, int location)

Set a x value on a graph as a minor tick. Location starts at 0.

SAMPLE

Here is a example.

```
dl ("rlib.so");
$query = "SELECT * FROM plu";
$rlib = rlib_init();
rlib_add_datasource_mysql($rlib, "mysql", "localhost", "user", "password", "database");
$format = "PDF";
rlib_add_query_as($rlib, "mysql", $query, "topline");
rlib_add_report($rlib, "report.xml");
rlib_set_output_format_from_text($rlib, $format);
rlib_execute($rlib);
```

```
header(rlib_get_content_type($rlib));
rlib_spool($rlib);
rlib_free($rlib);
```

Chapter 8. Examples

Example 1

It is assumed that you have a working PHP, APACHE, and MySQL setup. You will have to substitute the MySQL host, username, password, and database to what ever you created. The example is in three sections. First creating the database. Second we create the PHP source. Third we create the RLIB XML file.

MySQL Table Creation

```
DROP TABLE IF EXISTS example;

CREATE TABLE example (
 rn INT NOT NULL AUTO_INCREMENT,
 name VARCHAR(30) NOT NULL DEFAULT "",
 type INT NOT NULL,
 price FLOAT NOT NULL,
 PRIMARY KEY (rn),
 KEY (name)
);

INSERT INTO example (name, type, price)
 VALUES
 ("Hammer", 1, 10.00),
 ("Screw Driver", 1, 7.00),
 ("Bolts", 1, 2.00),
 ("Hot Dog", 2, 1.50),
 ("Soda", 2, 1.00),
 ("Chips", 2, 1.00),

 ("Jaguar", 3, 50000.00),
 ("Lexus", 3, 60000.00),
 ("Pinto", 3, 2000.00);
```

PHP Source

```
<? dl ("rlib.so");

 $format = "pdf";
 $sql_host = "localhost";
 $sql_users = "username";
 $sql_password = "password";
 $sql_database = "tablename";

 $rlib = rlib_init();
    rlib_add_datasource_mysql($rlib, "mysql", $sql_host, $sql_users, $sql_password, $sql_databas
    rlib_add_query_as($rlib, "mysql", "select * from example", "example");
    rlib_add_report($rlib, "report.xml");
    rlib_set_output_format_from_text($rlib, $format);
    rlib_execute($rlib);
    header( rlib_get_content_type($rlib));
    rlib_spool($rlib);
    rlib_free($rlib);
?>
```

RLIB XML SOURCE

```
<?xml version="1.0"?>
```

```
<!DOCTYPE report >
<Report fontSize="9" orientation="landscape">
 <ReportHeader>
  <Output>
   <Image value="'logo.jpg'" type="'jpeg'" width="50" height="50"/>
   <Line/>
   <Line fontSize="12">
    <literal width="8"/>
    <literal>REPORT HEADER........</literal>
   </Line>
   <Line/>
   <Line/>
   <Line fontsize="4"/>
   <HorizontalLine size="4" bgcolor="'white'"/>
   <HorizontalLine size="2" bgcolor="'black'"/>
   <HorizontalLine size="4" bgcolor="'white'"/>
  </Output>
 </ReportHeader>

 <PageHeader>
  <Output>
   <Line fontSize="11">
    <literal>Page Header (Example Report)</literal>
   </Line>
   <HorizontalLine size="4" bgcolor="'white'"/>
  </Output>
 </PageHeader>

 <Detail>
  <FieldHeaders>
   <Output>
    <HorizontalLine size="1" bgcolor="'black'"/>
    <Line bgcolor="'0xe5e5e5'">
     <literal width="30" col="1">Name</literal>
     <literal width="1"/>
     <literal width="5" col="2">Type</literal>
     <literal width="1"/>
     <literal width="10" col="3" align="right">Price</literal>
    </Line>
    <HorizontalLine size="1" bgcolor="'black'"/>
    <HorizontalLine size="4" bgcolor="'white'"/>
   </Output>
  </FieldHeaders>
  <FieldDetails>
   <Output>
    <Line bgcolor="iif(r.detailcnt%2,'0xe5e5e5','white')">
     <field value="name" width="30" align="left" col="1"/>
     <literal width="1"/>
     <field value="type" width="5" align="left" col="2"/>
     <literal width="1"/>
     <field value="val(price)" width="10" format="'%$.2d'" align="right" col="3"/>
    </Line>
   </Output>
  </FieldDetails>
 </Detail>

 <PageFooter>
  <Output>
   <Line>
    <literal>Page: </literal>
    <field value="r.pageno" width="3" align="right"/>
   </Line>
  </Output>
 </PageFooter>

 <ReportFooter>
```

```
  <Output>
   <Line fontSize="11">
    <literal>REPORT FOOTER</literal>
   </Line>
  </Output>
 </ReportFooter>
</Report>
```

REPORT HEADER.....

Page Header (Example Report)

Name	Type	Price
Hammer	1	10.00
Screw Driver	1	7.00
Bolts	1	2.00
Hot Dog	2	1.50
Soda	2	1.00
Chips	2	1.00
Jaguar	3	50,000.00
Lexus	3	60,000.00
Pinto	3	2,000.00

REPORT FOOTER

Appendix A. GNU Free Documentation License

PREAMBLE

The purpose of this License is to make a manual, textbook, or other functional and useful document "free" in the sense of freedom: to assure everyone the effective freedom to copy and redistribute it, with or without modifying it, either commercially or noncommercially. Secondarily, this License preserves for the author and publisher a way to get credit for their work, while not being considered responsible for modifications made by others.

This License is a kind of "copyleft", which means that derivative works of the document must themselves be free in the same sense. It complements the GNU General Public License, which is a copyleft license designed for free software.

We have designed this License in order to use it for manuals for free software, because free software needs free documentation: a free program should come with manuals providing the same freedoms that the software does. But this License is not limited to software manuals; it can be used for any textual work, regardless of subject matter or whether it is published as a printed book. We recommend this License principally for works whose purpose is instruction or reference.

APPLICABILITY AND DEFINITIONS

This License applies to any manual or other work, in any medium, that contains a notice placed by the copyright holder saying it can be distributed under the terms of this License. Such a notice grants a world-wide, royalty-free license, unlimited in duration, to use that work under the conditions stated herein. The "Document", below, refers to any such manual or work. Any member of the public is a licensee, and is addressed as "you". You accept the license if you copy, modify or distribute the work in a way requiring permission under copyright law.

A "Modified Version" of the Document means any work containing the Document or a portion of it, either copied verbatim, or with modifications and/or translated into another language.

A "Secondary Section" is a named appendix or a front-matter section of the Document that deals exclusively with the relationship of the publishers or authors of the Document to the Document's overall subject (or to related matters) and contains nothing that could fall directly within that overall subject. (Thus, if the Document is in part a textbook of mathematics, a Secondary Section may not explain any mathematics.) The relationship could be a matter of historical connection with the subject or with related matters, or of legal, commercial, philosophical, ethical or political position regarding them.

The "Invariant Sections" are certain Secondary Sections whose titles are designated, as being those of Invariant Sections, in the notice that says that the Document is released under this License. If a section does not fit the above definition of Secondary then it is not allowed to be designated as Invariant. The Document may contain zero Invariant Sections. If the Document does not identify any Invariant Sections then there are none.

The "Cover Texts" are certain short passages of text that are listed, as Front-Cover Texts or Back-Cover Texts, in the notice that says that the Document is released under this License. A Front-Cover Text may be at most 5 words, and a Back-Cover Text may be at most 25 words.

A "Transparent" copy of the Document means a machine-readable copy, represented in a format whose specification is available to the general public, that is suitable for revising the document straightforwardly with generic text editors or (for images composed of pixels) generic paint programs or (for drawings) some widely available drawing editor, and that is suitable for input to text formatters or for automatic translation to a variety of formats suitable for input to text formatters. A copy made in an otherwise Transparent file format whose markup, or absence of

markup, has been arranged to thwart or discourage subsequent modification by readers is not Transparent. An image format is not Transparent if used for any substantial amount of text. A copy that is not "Transparent" is called "Opaque".

Examples of suitable formats for Transparent copies include plain ASCII without markup, Texinfo input format, LaTeX input format, SGML or XML using a publicly available DTD, and standard-conforming simple HTML, PostScript or PDF designed for human modification. Examples of transparent image formats include PNG, XCF and JPG. Opaque formats include proprietary formats that can be read and edited only by proprietary word processors, SGML or XML for which the DTD and/or processing tools are not generally available, and the machine-generated HTML, PostScript or PDF produced by some word processors for output purposes only.

The "Title Page" means, for a printed book, the title page itself, plus such following pages as are needed to hold, legibly, the material this License requires to appear in the title page. For works in formats which do not have any title page as such, "Title Page" means the text near the most prominent appearance of the work's title, preceding the beginning of the body of the text.

A section "Entitled XYZ" means a named subunit of the Document whose title either is precisely XYZ or contains XYZ in parentheses following text that translates XYZ in another language. (Here XYZ stands for a specific section name mentioned below, such as "Acknowledgements", "Dedications", "Endorsements", or "History".) To "Preserve the Title" of such a section when you modify the Document means that it remains a section "Entitled XYZ" according to this definition.

The Document may include Warranty Disclaimers next to the notice which states that this License applies to the Document. These Warranty Disclaimers are considered to be included by reference in this License, but only as regards disclaiming warranties: any other implication that these Warranty Disclaimers may have is void and has no effect on the meaning of this License.

VERBATIM COPYING

You may copy and distribute the Document in any medium, either commercially or noncommercially, provided that this License, the copyright notices, and the license notice saying this License applies to the Document are reproduced in all copies, and that you add no other conditions whatsoever to those of this License. You may not use technical measures to obstruct or control the reading or further copying of the copies you make or distribute. However, you may accept compensation in exchange for copies. If you distribute a large enough number of copies you must also follow the conditions in section 3.

You may also lend copies, under the same conditions stated above, and you may publicly display copies.

COPYING IN QUANTITY

If you publish printed copies (or copies in media that commonly have printed covers) of the Document, numbering more than 100, and the Document's license notice requires Cover Texts, you must enclose the copies in covers that carry, clearly and legibly, all these Cover Texts: Front-Cover Texts on the front cover, and Back-Cover Texts on the back cover. Both covers must also clearly and legibly identify you as the publisher of these copies. The front cover must present the full title with all words of the title equally prominent and visible. You may add other material on the covers in addition. Copying with changes limited to the covers, as long as they preserve the title of the Document and satisfy these conditions, can be treated as verbatim copying in other respects.

If the required texts for either cover are too voluminous to fit legibly, you should put the first ones listed (as many as fit reasonably) on the actual cover, and continue the rest onto adjacent pages.

If you publish or distribute Opaque copies of the Document numbering more than 100, you must either include a machine-readable Transparent copy along with each Opaque copy, or state in or with each Opaque copy a computer-network location from which the general network-using

public has access to download using public-standard network protocols a complete Transparent copy of the Document, free of added material. If you use the latter option, you must take reasonably prudent steps, when you begin distribution of Opaque copies in quantity, to ensure that this Transparent copy will remain thus accessible at the stated location until at least one year after the last time you distribute an Opaque copy (directly or through your agents or retailers) of that edition to the public.

It is requested, but not required, that you contact the authors of the Document well before redistributing any large number of copies, to give them a chance to provide you with an updated version of the Document.

MODIFICATIONS

You may copy and distribute a Modified Version of the Document under the conditions of sections 2 and 3 above, provided that you release the Modified Version under precisely this License, with the Modified Version filling the role of the Document, thus licensing distribution and modification of the Modified Version to whoever possesses a copy of it. In addition, you must do these things in the Modified Version:

A. Use in the Title Page (and on the covers, if any) a title distinct from that of the Document, and from those of previous versions (which should, if there were any, be listed in the History section of the Document). You may use the same title as a previous version if the original publisher of that version gives permission.

B. List on the Title Page, as authors, one or more persons or entities responsible for authorship of the modifications in the Modified Version, together with at least five of the principal authors of the Document (all of its principal authors, if it has fewer than five), unless they release you from this requirement.

C. State on the Title page the name of the publisher of the Modified Version, as the publisher.

D. Preserve all the copyright notices of the Document.

E. Add an appropriate copyright notice for your modifications adjacent to the other copyright notices.

F. Include, immediately after the copyright notices, a license notice giving the public permission to use the Modified Version under the terms of this License, in the form shown in the Addendum below.

G. Preserve in that license notice the full lists of Invariant Sections and required Cover Texts given in the Document's license notice.

H. Include an unaltered copy of this License.

I. Preserve the section Entitled "History", Preserve its Title, and add to it an item stating at least the title, year, new authors, and publisher of the Modified Version as given on the Title Page. If there is no section Entitled "History" in the Document, create one stating the title, year, authors, and publisher of the Document as given on its Title Page, then add an item describing the Modified Version as stated in the previous sentence.

J. Preserve the network location, if any, given in the Document for public access to a Transparent copy of the Document, and likewise the network locations given in the Document for previous versions it was based on. These may be placed in the "History" section. You may omit a network location for a work that was published at least four years before the Document itself, or if the original publisher of the version it refers to gives permission.

K. For any section Entitled "Acknowledgements" or "Dedications", Preserve the Title of the section, and preserve in the section all the substance and tone of each of the contributor acknowledgements and/or dedications given therein.

L. Preserve all the Invariant Sections of the Document, unaltered in their text and in their titles. Section numbers or the equivalent are not considered part of the section titles.

M. Delete any section Entitled "Endorsements". Such a section may not be included in the Modified Version.

N. Do not retitle any existing section to be Entitled "Endorsements" or to conflict in title with any Invariant Section.

O. Preserve any Warranty Disclaimers.

If the Modified Version includes new front-matter sections or appendices that qualify as Secondary Sections and contain no material copied from the Document, you may at your option designate some or all of these sections as invariant. To do this, add their titles to the list of Invariant Sections in the Modified Version's license notice. These titles must be distinct from any other section titles.

You may add a section Entitled "Endorsements", provided it contains nothing but endorsements of your Modified Version by various parties--for example, statements of peer review or that the text has been approved by an organization as the authoritative definition of a standard.

You may add a passage of up to five words as a Front-Cover Text, and a passage of up to 25 words as a Back-Cover Text, to the end of the list of Cover Texts in the Modified Version. Only one passage of Front-Cover Text and one of Back-Cover Text may be added by (or through arrangements made by) any one entity. If the Document already includes a cover text for the same cover, previously added by you or by arrangement made by the same entity you are acting on behalf of, you may not add another; but you may replace the old one, on explicit permission from the previous publisher that added the old one.

The author(s) and publisher(s) of the Document do not by this License give permission to use their names for publicity for or to assert or imply endorsement of any Modified Version.

COMBINING DOCUMENTS

You may combine the Document with other documents released under this License, under the terms defined in section 4 above for modified versions, provided that you include in the combination all of the Invariant Sections of all of the original documents, unmodified, and list them all as Invariant Sections of your combined work in its license notice, and that you preserve all their Warranty Disclaimers.

The combined work need only contain one copy of this License, and multiple identical Invariant Sections may be replaced with a single copy. If there are multiple Invariant Sections with the same name but different contents, make the title of each such section unique by adding at the end of it, in parentheses, the name of the original author or publisher of that section if known, or else a unique number. Make the same adjustment to the section titles in the list of Invariant Sections in the license notice of the combined work.

In the combination, you must combine any sections Entitled "History" in the various original documents, forming one section Entitled "History"; likewise combine any sections Entitled "Acknowledgements", and any sections Entitled "Dedications". You must delete all sections Entitled "Endorsements".

COLLECTIONS OF DOCUMENTS

You may make a collection consisting of the Document and other documents released under this License, and replace the individual copies of this License in the various documents with a single copy that is included in the collection, provided that you follow the rules of this License for verbatim copying of each of the documents in all other respects.

You may extract a single document from such a collection, and distribute it individually under this License, provided you insert a copy of this License into the extracted document, and follow this License in all other respects regarding verbatim copying of that document.

AGGREGATION WITH INDEPENDENT WORKS

A compilation of the Document or its derivatives with other separate and independent documents or works, in or on a volume of a storage or distribution medium, is called an "aggregate" if the copyright resulting from the compilation is not used to limit the legal rights of the compilation's users beyond what the individual works permit. When the Document is included in an aggregate, this License does not apply to the other works in the aggregate which are not themselves derivative works of the Document.

If the Cover Text requirement of section 3 is applicable to these copies of the Document, then if the Document is less than one half of the entire aggregate, the Document's Cover Texts may be placed on covers that bracket the Document within the aggregate, or the electronic equivalent of covers if the Document is in electronic form. Otherwise they must appear on printed covers that bracket the whole aggregate.

TRANSLATION

Translation is considered a kind of modification, so you may distribute translations of the Document under the terms of section 4. Replacing Invariant Sections with translations requires special permission from their copyright holders, but you may include translations of some or all Invariant Sections in addition to the original versions of these Invariant Sections. You may include a translation of this License, and all the license notices in the Document, and any Warranty Disclaimers, provided that you also include the original English version of this License and the original versions of those notices and disclaimers. In case of a disagreement between the translation and the original version of this License or a notice or disclaimer, the original version will prevail.

If a section in the Document is Entitled "Acknowledgements", "Dedications", or "History", the requirement (section 4) to Preserve its Title (section 1) will typically require changing the actual title.

TERMINATION

You may not copy, modify, sublicense, or distribute the Document except as expressly provided for under this License. Any other attempt to copy, modify, sublicense or distribute the Document is void, and will automatically terminate your rights under this License. However, parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance.

FUTURE REVISIONS OF THIS LICENSE

The Free Software Foundation may publish new, revised versions of the GNU Free Documentation License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns. See http://www.gnu.org/copyleft/.

Each version of the License is given a distinguishing version number. If the Document specifies that a particular numbered version of this License "or any later version" applies to it, you have the option of following the terms and conditions either of that specified version or of any later version that has been published (not as a draft) by the Free Software Foundation. If the Document does not specify a version number of this License, you may choose any version ever published (not as a draft) by the Free Software Foundation.

ADDENDUM: How to use this License for your documents

To use this License in a document you have written, include a copy of the License in the document and put the following copyright and license notices just after the title page:

> Copyright (c) YEAR YOUR NAME. Permission is granted to copy, distribute and/or modify this document under the terms of the GNU Free Documentation License, Version 1.2 or any later version published by the Free Software Foundation; with no Invariant Sections, no Front-Cover Texts, and no Back-Cover Texts. A copy of the license is included in the section entitled "GNU Free Documentation License".

If you have Invariant Sections, Front-Cover Texts and Back-Cover Texts, replace the "with...Texts." line with this:

> with the Invariant Sections being LIST THEIR TITLES, with the Front-Cover Texts being LIST, and with the Back-Cover Texts being LIST.

If you have Invariant Sections without Cover Texts, or some other combination of the three, merge those two alternatives to suit the situation.

If your document contains nontrivial examples of program code, we recommend releasing these examples in parallel under your choice of free software license, such as the GNU General Public License, to permit their use in free software.

Appendix B. SWIG License

LICENSE

RLIB is distributed with language bindings that allow you to use rlib (the c library) in other languages. Some of the bindings were created using SWIG (www.swig.org). SWIG generated source is not GPL, however is still is free software.

Simplified Wrapper and Interface Generator (SWIG)

SWIG is distributed under the following terms:

==========

I.

This software includes contributions that are Copyright (c) 1998-2002 University of Chicago. All rights reserved.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution. Neither the name of the University of Chicago nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE UNIVERSITY OF CHICAGO AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE UNIVERSITY OF CHICAGO OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

II.

Copyright (c) 1995-1998 The University of Utah and the Regents of the University of California All Rights Reserved

Permission is hereby granted, without written agreement and without license or royalty fees, to use, copy, modify, and distribute this software and its documentation for any purpose, provided that (1) The above copyright notice and the following two paragraphs appear in all copies of the source code and (2) redistributions including binaries reproduces these notices in the supporting documentation. Substantial modifications to this software may be copyrighted by their authors and need not follow the licensing terms described here, provided that the new terms are clearly indicated in all files where they apply.

IN NO EVENT SHALL THE AUTHOR, THE UNIVERSITY OF CALIFORNIA, THE UNIVERSITY OF UTAH OR DISTRIBUTORS OF THIS SOFTWARE BE LIABLE TO ANY PARTY FOR DIRECT, INDIRECT, SPECIAL, INCIDENTAL, OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OF THIS SOFTWARE AND ITS DOCUMENTATION, EVEN IF THE AUTHORS OR ANY OF THE ABOVE PARTIES HAVE BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

THE AUTHOR, THE UNIVERSITY OF CALIFORNIA, AND THE UNIVERSITY OF UTAH SPECIFICALLY DISCLAIM ANY WARRANTIES,INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE SOFTWARE PROVIDED HEREUNDER IS ON AN "AS IS" BASIS, AND THE AU-

THORS AND DISTRIBUTORS HAVE NO OBLIGATION TO PROVIDE MAINTENANCE, SUPPORT, UPDATES, ENHANCEMENTS, OR MODIFICATIONS.