Report Generator (RG) Language

# Basic Conventions

## File extensions

Formats are text files with extensions of .b with "p" appended for the primary or control format e.g. LtdFullGAAP.bp

The extensions are just for convention, and to facilitate formatting within an editor e.g. NotePad++.

The compiler does not make any use of the extension.

Format names must NOT include spaces.

## Directory structure

The directory structure is:

Admin root /Formats/Corp for corporate formats

Admin root /Formats/Unincorp for unicorporated formats

Admin root /Formats/Shared for shared or common or library formats

The RG compiler searches in these three directories for a given format.

Only if a format is in a subdirectory of any of these directories is directory information needed in its name,

e.g.

#include genFuncs.b would find the format Admin/Formats/Shared/genFuncs.b

#include farmers/notes.b would find the format Admin/Formats/Unincorp/farmers/notes.b

## Processing

Formats are processed on a line by line basis.

## Compiling

Compiler command lines start with # as the first non white space character. The command can be followed by a comment after at least one space.

## Language statements

Language statements are enclosed within []s. A format line can include multiple statements.

Anything else is ignored so white space, blank lines, and comments can be used anywhere, with no need for comment characters.

Strings with spaces in them should be quoted using either single or double quotes.

White space within statements is ignored, unless enclosed within "" or ''s

Tab characters should not be used i.e. spaces should be used for indenting.

Multiple lines in the format may be extended to form a single processing "line" via an "&" character at the end of lines before the last one.

Anything else is ignored so white space, blank lines, and comments can be used anywhere, with no need for comment characters.

White space within statements is ignored, unless enclosed within ""s

## Compiler

All compiler commands and statements are in lower case.

Compiler Commands:

Only one compiler command can be used per line, with no [] statements on the same line.

The available commands are:

\* #include formatName

\* #constant constantName {=} value where value can be a number, a quoted string, or an already defined constant

#if

#else

#end

#include formatName

-------------------

Example:

#include notes.b

Formats can be included at any point in a format and be nested to any depth.

A format can be included more than once if that should be appropriate to the report.

formatName does not need to be quoted though it can be.

#constant constantName {=} value

--------------------------------

Examples:

#constant YearC = 2011

#constant FredC 'Fred Smith'

#constant fredC FredC

constantName: See Names and RG Data Types below for the naming rules.

{=} A = sign between name and value is optional

value: Must be scalar i.e. a single number, a quoted string, or an already defined constant name which is not quoted.

It cannot be a more complex expression. Note that strings must be quoted. (A non quoted string is assumed

Once defined a constant can be used in any statement expression.

Names

=====

RG names:

- must start with a letter

- can be of any length

- cannot contain white space, $, or quote characters

- are case sensitive, thus fredC is a different constant from FredC in the constant examples above.

- must have a final character as specified below to indicate its type

#include formatName

#constant constantName expr where expr can be any expression, whether numeric or string

## Data types

The final character of a RG data name/construct indicates its type. A type character is mandatory. The types are:

Preset or defined by the RG program

|  |  |  |
| --- | --- | --- |
| T | T seT  (or Tuple) | Values that the RG knows how to process re skipping empty values, tags, contexts etc e.g. DirectorNamesT |

Either preset or can be defined by formatting

|  |  |  |
| --- | --- | --- |
| B | Boolean | Value 1 or 0 i.e. True or False, Yes or No.  (Actually anything other than 0 means True/Yes.) |
| C | Constant |  |
| D | Date |  |
| H | Heading | Heading (or text) from the Headings subsystem, processed by the RG as a string |
| I | Integer | Integer number |
| M | Money | Money number (processed internally as a 64 bit integer with 2 places of implied decimals, used to avoid rounding issues) |
| R | Real | Real or floating point number for percentages, ratios, or other calculated values with possible decimal fractions |
| S | String |  |
| X | Cross reference |  |

**Defined by formatting**

|  |  |  |
| --- | --- | --- |
| F | Function |  |
| P | Para(graph) |  |

## Expressions

The RG supports 4 types of expressions:

**integer**

**money**

**real**

**boolean**

**string**

These may be used together in any sensible way, with conversion from one type to another being automatic.

### Operators

**+ - / \***

**&** (concatenation of strings)

**=**

**or**

**and**

### Precedence

From lowest to highest:

**+ -**

**/ \***

**or**

**and**

optionally controlled by ()'s

### Language Statements

Conventions:

|  |  |
| --- | --- |
| dD | a date variable |
| <a|b|...> | one item to be chosen e.g. a or b etc. {...} optional |
| intExpr | an integer expression which can be just a number |
| stringExpr | A string expression, meaning a string either inline as "double quotes string text which can include single quotes e.g. O'Reilly", 'single quotes text which can include double quotes "like this"', a string variable, a para call, a call to a function which generates output, or a nested statement [nl] or d[ ...] which generates output. Any number of these elements can be strung together one after the other, with space separators if needed. If a space is not needed to tell the compiler where one element ends and another starts e.g. when an element is quoted or is enclosed within []s, the use of the space separator is optional.  String expressions may include numeric elements  ()s may be included in the expression if desired, |
| Expr | An expression of any type i.e. string, integer, money, boolean, or real |
| Cccsl | CSS class comma separated list, though most usually just one class name would be used with no comma needed |

#### CCCSL

The full set of CSS classes to be defined.

A start is:

**Built in "fixed" classes**:

|  |  |
| --- | --- |
| b | Bold |
| i | Italics |
| u | Underlined |
| nb | No bold.  Typically used for a non-bold section within a heading that almost always will be bold |
| lj | Left justify |
| rj | Right justify |
| ctr | Centre |
| hide | to hide (not show) content |

**Classes editable via user preferences** for font, size, bold, italics, underline, colour

|  |  |
| --- | --- |
| h1 | class for h1 headings |
| h2 | class for h2 headings |
| h3 | class for h3 headings |
| h4 | class for h4 headings |
| f0 | font (or style) 0 - default or body text |
| f1 | font (or style) 1 |
| f2 | font (or style) 2 |
| f3 | font (or style) 3 |
| f4 | font (or style) 4 |

Statements:

|  |  |
| --- | --- |
| [c<1|...>{,intExpr} {c:cccsl} Expr] | Output Expr result to table column c<1|...>, optionally across intExpr columns.  The RG takes care of generating any necessary rowspan, colspan, or empty <td> html attributes/statement. |
| [centred] | Set output to be centred on the page until turned off or the format/header/footer/para/block ends. |
| [centred off] | Turn centring off |
| [df dD] | Output date dD in full form e.g. 03 February 2011 |
| [ds dD] | Output date dD in short form e.g. 03 Feb 2011 or 03/02/2011 or 03.02.2011 as per preferences |
| [dx dD] | Output date dD in iXBRL form e.g. 2011-02-03 |
| [dy dD] | Output date dD in year form e.g. 2011 |
| [def block] | Define a block to be output on one page  i.e. to be bumped to the next page if it won't fit on the current page. |
| **{statements}** |  |
| **[end block]** |  |
| [def footer] | Define a footer to be output at the bottom of pages.  Can be changed or cancelled by defining an empty one. |
| **{statements}** | Footers survive across formats |
| **[end footer]** |  |
| [def header] | Define a header to be output at the top of pages.  Can be changed or cancelled by defining an empty one. |
| **{statements}** | Headers survive across formats |
| **[end header]** |  |
| [def function funcNameF ...] | Define a function. Parameters and return value staements still to be defined |
| **{statements}** |  |
| **[end function]** |  |
| [def para paraNameP] | Define a "paragraph" of text which can be called by name anywhere a string can be used.  A para can contain any statements other than [def para ...] or [def function ...] statements.  A para is like a function without parameters or return values.  If a para is called within a stringExpr its output is not output directly but is returned to the stringExpr being built up. |
| **{statements}** |  |
| **[end block]** |  |
| [h1 {c:cccsl} stringExpr] | Output the stringExpr as an html h1 heading. Can include nested statements. |
| [h2 {c:cccsl} stringExpr] | Output the stringExpr as an html h2 heading. Can include nested statements. |
| [h3 {c:cccsl} stringExpr] | Output the stringExpr as an html h3 heading. Can include nested statements. |
| [h4 {c:cccsl} stringExpr] | Output the stringExpr as an html h4 heading. Can include nested statements. |
| [line] | Output a blank line |
| [lines intExpr] | Output intExpr blank lines |
| [np] | Start a new page and output the header if defined |
| [nl] | New line. Will cause a <br/>, <p>, or <tr> in the html according to context. |
| [nonl] | No new line i.e. suppress a new line |
| [p {c:cccsl} stringExpr] | Output the stringExpr as an html paragraph i.e. <p> in body text as the default. Can include nested statements. |
| [page# {c:cccsl}] | Output the current page # |
| [table] | Start a table. A table statement is not needed for balance output statements which are automatically output in tabular form. Table rows are created via [c# ...] statements.  A table ends when the format/header/footer/para/block ends or a non table statement comes along. |
| [toc] | Start a two column Table of Contents table |
| [toc formatName] | Insert the tocName defined in format formatName as a link plus page # as a table of contents row |
| [tocName stringExpr] | Name of the page for the toc. Does not cause any output to occur. |

Type:

|  |  |  |
| --- | --- | --- |
| C | Control | flow control or Conditional statements |
| F | Formatting | defines a Formatting option |
| L | Language | a Language statement/construct which does not directly cause output |
| O | Output | generates html Output |
| R | Reference | a Reference or bookmark in the accounts |

/- compilation working

\* Used - Can be used in a expr of type S (string), I (intExpr), E (expr) according to type of output

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Statement | Type | Used\* |  |
| \* | [c<1|..|9>{,intExpr {c:css} strExpr] | O |  | Output strExpr result to table column c<1|..|9> (c1 to c9), optionally across intExpr columns, optionally with css class(es) included |
|  |  |  |  | The RG takes care of generating any necessary rowspan, colspan, or empty <td> html attributes/statements. |
| \* | [block] | L |  | Start of a block. See Function Constructs section below. |
| \* | [centre {off}] | F |  | Set output to be centred on the page until turned off, or turn centre state off with the off attribute. |
|  |  |  |  | Stays in force until turned off i.e. through page breaks and formats, except in headers and footers, |
|  |  |  |  | which maintain their own 'centre' state and should use their own [centre] statement if required. |
|  |  |  |  | There is no need to turn centre off in a header or footer. |
| \* | [d {<f|s|x|y>} {c:css} dD] |  |  | Output date formatted according the the optional <f|s|x|y> attribute with none meaning f, |
|  |  |  |  | optionally with css class(es) included. The first attribute means: |
|  |  | O | S | f: Output date dD in full form e.g. 03 February 2011 |
|  |  | O | S | s: Output date dD in short form e.g. 03 Feb 2011 or 03/02/2011 or 03.02.2011 as per preferences |
|  |  | O | S | x: Output date dD in iXBRL form e.g. 2011-02-03 |
|  |  | O | E | y: Output date dD in year form e.g. 2011 |
|  | [else] | C |  | Start of an alternative if branch. See Flow Control Statements below. |
|  | [else if intExpr] | C |  | Alternative if test. See Flow Control Statements below. |
| \* | [end] | L |  | End of an if, else if, block, header, footer, function, para, table, or toc construct. See sections below. |
| \* | [expr] | O |  | Evaluate the expr and output the result if any. (This statement could be used to call a function which |
|  |  |  |  | does not generate output, or to call a para which is empty.) |
|  |  |  |  | If output is generated the statement acts like [p strExpr]. |
|  | [function nameF{(...)}] | L |  | Start of a function. See Function Constructs section below. |
| \* | [footer] | L |  | Start of a footer. See Function Constructs section below. |
| \* | [header] | L |  | Start of a header. See Function Constructs section below. |
| \* | [h<1|..|4> {c:css} strExpr | ] O |  | Output the strExpr result as an html h1 to h4 heading, optionally with css class(es) included. |
|  | [if intExpr] | C |  | If test. See Flow Control Statements below. |
| \* | [line] | O |  | End any output in progress and output a blank line |
| \* | [lines intExpr] | O |  | End any output in progress and output intExpr blank lines |
| \* | [nl] | O | Y | New line, which really means end the current line so that any following output will be on a new line. |
|  |  |  |  | To achieve vertical space in the report use [line], [lines intExpr], or a suitable css class. |
|  |  |  |  | If [nl] is used within a string expression it causes a <br/> (break or new line) tag to be inserted in the string. |
| \* | [np] | O |  | Start a new page and output the header if defined |
| \* | [p {c:css} strExpr] | O |  | Output strExpr result as an html <p> paragraph in body text as the default, optionally with css class(es) included |
| \* | [page# {c:css}] | O |  | Output the current page #, optionally with css class(es) included |
| \* | [para nameP] | L |  | Start of a para. See Function Constructs section below. |
|  | [return] | L |  | Return from (or end) a function or para. |
|  | [span {c:css} strExpr] | O | S | Output the strExpr result within an html span tag, optionally with css class(es) included. Most commonly |
|  |  |  |  | [span ...] would be used for the purpose of attaching css to a segment of output within other output, |
|  |  |  |  | though it could also be used to define text for a [xref xrefNameX] statement. |
|  |  |  |  | e.g. [span c:b 'today'] generates <span class='b'>today</span> |
|  |  |  |  | [span c:b,i AgentNameS] generates <span class='b i'>Woodgate & Co.</span> or whatever the Agent's name is. |
| \* | [table] | O |  | Start a table. A table statement is not needed for balance output statements which are automatically output |
|  |  |  |  | in tabular form. Table rows are created via [c# ...] statements. |
|  |  |  |  | A table ends on an [end] statement. |
| \* | [toc {c:css} {c:css} | O |  | Insert a Table of Contents table. [toc could also be used for other cross reference tables besides tables of |
|  | {xrefNameX}] |  |  | contents tables. A Table of Contents could also be constructed via [table] and [c#...] statements but |
|  |  |  |  | [toc provides a shorthand method. |
|  |  |  |  | [toc] i.e. without attributes means start the table. |
|  |  |  |  | [toc xrefNameX] means insert a two column table of contents row for the text and page number defined by |
|  |  |  |  | xrefNameX with the text as a link. |
|  |  |  |  | The two optional css attributes override preferences for the text and page number columns. |
|  |  |  |  | A table of contents table ends on an [end] statement. |
| \* | [xref <target|text|page| both|link> {c:css} nameX] | R |  | A cross reference for toc, link, or page number/text cross referencing use, according to the first attribute: target: Defines the target of the cross reference. The page number and the next text from a p, h, or span statement (up to the first line end if multi line) are recorded for use elsewhere in the report, either before or after the [xref target location. Example: [xref target notesX] |
|  |  | O | S | text: Outputs the text of the target of the xref |
|  |  | O | E | page: Outputs the page number of the target of the xref |
|  |  | O | S | both: Outputs "text on page x" where 'text' and 'x' are the text and page number of the target |
|  |  | O | S | link: Outputs the text of the target of the xref as a link to the target |
|  |  |  |  | Optionally css classes can be used to override preferences in the output cases. |

Pagination Control

|  |  |  |
| --- | --- | --- |
| [block] {statements [end] | L | Define a block to be output on one page i.e. to be bumped to the next page if it won't fit on the current page.  Cannot contain footer, header, function, or para statements, though it may use these. |

Function Constructs

The code which defined functions cannot be nested. Footers and headers execute automatically

|  |  |  |
| --- | --- | --- |
|  |  |  |
| [footer]{statements}[end] | L | Define a footer to be output at the bottom of pages.  Stays active across formats amd pages until changed or cancelled by defining an empty one. |
| [header] {statements}[end] | L | Define a header to be output at the top of pages.  Stays active across formats amd pages until changed or cancelled by defining an empty one. |
| [function nameF{(...)}]  {statements}  {[return]}  [end] | L | Define a named function, optionally with any number of comma separated parameters to be passed to it.  A function can contain any other statements except block and other function statements.  A function may optionally include one or more [return] statements. ([end] acts like [return] so a [return] statement isn't necessary but can be convenient if the function includes different execution paths.)  A function is used or called by name either directly e.g. [nameF{(...)}] or by within an expr.  A direct function call like [nameF] acts like a [p nameF] statement if the function generates output.  If a functiion is called within an expr:  - its output is not output directly to the report but is returned to the expr as a number or string depending on what is output by the function.  - if the function includes multiple output statements, the output is treated as a string and concatenated with a space separator  - line feeds (<br/> tags) from [NL] statements are preserved, but lines from different format lines do not result in <br/> tags. Thus  [function FredF]  ["Now is[nl]the hour"]  ["for all good men"]  [end]  [FredS = FredF]  results in FredS holding "Now is<br/>the hour when all good men"  and  [p FredF] of [FredF]  results in output of <p>Now is<br/>the hour when all good men</p> |
| [para nameP] {statements}  [end] | L | Define a named "paragraph" of text.  A para is like a function without parameters.  See [function ...] above for how a para operates. |
|  |  |  |

# Still to be defined

Balance output statements

Notes related statements

Logic statements e.g. if/else, case, loops, function details

# Report Generator (RG) Data

User Data Maintained via "Agent Details" page

RG Name

-------

FirmNameS

FirmAddress1S

FirmAddress2S

FirmAddress3S

FirmAddressCityOrTownS

FirmAddressCountyS

FirmAddressCountryS

FirmAddressPostcodeS

User Preferences Maintained via "My Preferences" page

================

Layout stuff:

All the CSS classes

toc item CSS class

toc page CSS class

header default CSS class

footer default CSS class

page number default CSS class

page number position

Full date style

Short date style

Default thousands separator

Default decimal separator

Default currency symbol

...

Headings by client type

Data for a user maintained via "My Preferences" page

Company

|  |  |  |
| --- | --- | --- |
| RG Name | Default | Comments |
| AccountsFullUnauditedH | Report of the Directors and Unaudited Financial Statements |  |
| AccountsFullAuditedH | Report of the Directors and Audited Financial Statements |  |
| AccountsShortH | Financial Statements |  |
| AccountsPeriodH | Period of Accounts |  |
| CompanyInformationH | Company Information |  |
| DirectorH | Director | Singular. Plural generated by program. |
| CompanyRegistrationNumberH | Company Registration Number |  |
| CompanySecretaryH | Company secretary |  |
| RegisteredOfficeH | Registered office address |  |

INSERT INTO Headings (AgentId, Ref, Heading, AddT)

VALUES

(1, 'AccountsFullUnauditedH', 'Report of the Directors and Unaudited Financial Statements', NOW()),

(1, 'AccountsFullAuditedH', 'Report of the Directors and Audited Financial Statements', NOW()),

(1, 'AccountsShortH', 'Financial Statements', NOW()),

(1, 'AccountsPeriodH', 'Period of Accounts', NOW()),

(1, 'CompanyInformationH', 'Company Information', NOW()),

(1, 'CompanyRegistrationNumberH', 'Company Registration Number', NOW()),

(1, 'CompanySecretaryH', 'Company Secretary', NOW()),

(1, 'DirectorH', 'Director', NOW()),

(1, 'RegisteredOfficeH', 'Registered Office Address', NOW());

|  |  |  |
| --- | --- | --- |
|  | | |
| Set By Program | Comments |
| AgentNameS | From Agent account DB record |
| FullAccountsS | FullUnauditedAccountsH or FullAuditedAccountsH |
| DirectorsS | DirectorH with "s" added if DirectorsAtReportDateI > 1 |
| sets |  |
| AgentAddressT | From Agent account DB record |
| DirectorNamesT |  |
| FirmAddressT |  |
| RegisteredOfficeT |  |

Client Data Required from SAPA for a Company to run LtdFullGAAP.cp

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| RG Formatting Name | SAPA Export Format Name | SAPA Name | Tag | Context | Example | Comment |
| EntityNameS | EntityName | cdb.gen.2 | uk-bus:EntityCurrentLegalOrRegisteredName | YThis | iXBRL Test Ltd. | Should agree with Entity name entered when Entity is added |
| RegistrationNumberS | RegistrationNumber | cdb.sta.1.1 | uk-bus:UKCompaniesHouseRegisteredNumber | YThis | 87654321 |  |
| IncorporationCountryS | IncorporationCountry | cdb.sta.3 | uk-bus: CountryFormationOrIncorporation |  | England and Wales |  |
| CompanySecretaryS | CompanySecretary |  | uk-bus:NameEntityOfficer |  |  |  |
| RegisteredOfficeAddressT | RegisteredOfficeAddress1 | cdb.sta.4.1 | uk-bus:AddressLine1 | RegisteredOffice |  |  |
|  | RegisteredOfficeAddress2 | cdb.sta.4.2 | uk-bus:AddressLine2 | RegisteredOffice |  |  |
|  | RegisteredOfficeAddress2 | cdb.sta.4.3 | uk-bus:AddressLine3 | RegisteredOffice |  |  |
|  | RegisteredOfficeCityOrTown | cdb.sta.4.4 | uk-bus:PrincipalLocation-CityOrTown | RegisteredOffice |  |  |
|  | RegisteredOfficeCounty | cdb.sta.4.5 | uk-bus:CountyRegion | RegisteredOffice |  |  |
|  | RegisteredOfficeCountry | cdb.sta.4.6 | RegisteredOffice |  |  |  |
|  | RegisteredOfficePostcode | cdb.sta.4.7 | uk-bus:PostalCodeZip | RegisteredOffice |  |  |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| RG Name | SAPA Name | Tag | Context | Example |
| YNameS |  |  |  | 2010 i.e. "Y" = current year |
| YDescriptionS |  |  |  | Year |
| YStartDateD |  |  |  | 2010-01-01 |
| YEndDateD |  |  |  | 2010-12-31 |
| Y1NameS |  |  |  | 2009 i.e. "1" in Y1 = Y-1 |
| Y1DescriptionS |  |  |  | Year |
| Y1StartDateD |  |  |  | 2009-01-01 |
| Y1EndDateD |  |  |  | 2009-12-31 |
| Y2NameS |  |  |  | 2008 |
| Y2DescriptionS |  |  |  | year |
| Y2StartDateD |  |  |  | 2008-01-01 |
| Y2EndDateD |  |  |  | 2008-12-31 |
| Y3NameS |  |  |  | 2007 |
| Y3DescriptionS |  |  |  | year |
| Y3StartDateD |  |  |  | 2007-01-01 |
| Y3EndDateD |  |  |  | 2007-12-31 |
| Y4NameS |  |  |  | 2006 |
| Y4DescriptionS |  |  |  | year |
| Y4StartDateD |  |  |  | 2006-01-01 |
| Y4EndDateD |  |  |  | 2006-12-31 |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| RG Name | SAPA Name | Tag | Context | Example |
| DirectorsAtReportDateI | DirsAtRepDateIg |  |  |  |
| AuditedB |  |  |  |  |
|  |  |  |  |  |

Contexts

<!-- Context: Registered Office -->

<xbrli:context id="RegisteredOffice">

<xbrli:entity>

<xbrli:identifier scheme="http://www.companieshouse.gov.uk/">[RegistrationNumberS]</xbrli:identifier>

<xbrli:segment>

<xbrldi:explicitMember dimension="uk-bus:EntityContactTypeDimension">uk-bus:RegisteredOffice</xbrldi:explicitMember>

<xbrldi:explicitMember dimension="uk-bus:AddressTypeDimension">uk-bus:Postal</xbrldi:explicitMember>

</xbrli:segment>

</xbrli:entity>

<xbrli:period>

<xbrli:startDate>[dx YStartD]</xbrli:startDate>

<xbrli:endDate>[dx YEndD]</xbrli:endDate>

</xbrli:period>

</xbrli:context>

# Appendix - Discussion Notes

Removed