COMPEX COMMERCE

Process Specification

ReportDSL

Version 0.1

16. 11. 2015

Index

1 Introduction

1.1 Vaadin

Vaadin is a web application framework for Java. In contrast to Javascript libraries and browser-plugin based solutions, Vaadin features a complete stack that includes a robust server-side programming model as well as client-side development tools based on GWT and HTML5.

More information:

https://vaadin.com/home

1.2 Birt Report

The Business Intelligence and Reporting Tools (BIRT) Project is an open source software project that provides reporting and business intelligence capabilities for rich client and web applications, especially those based on Java and Java EE. BIRT is a top-level software project within the Eclipse Foundation, an independent not-for-profit consortium of software industry vendors and an open source community.

The project's stated goals are to address a wide range of reporting needs within a typical application, ranging from operational or enterprise reporting to multi-dimensional online analytical processing (OLAP). Initially, the project has focused on and delivered capabilities that allow application developers to easily design and integrate reports into applications.

The project is supported by an active community of users at BIRT Developer Center and developers at the Eclipse.org BIRT Project page.

BIRT has two main components: a visual report designer within the Eclipse IDE for creating BIRT Reports, and a runtime component for generating reports that can be deployed to any Java environment. The BIRT project also includes a charting engine that is both fully integrated into the report designer and can be used standalone to integrate charts into an application.

BIRT Report designs are persisted as XML and can access a number of different data sources including JDO datastores, JFire Scripting Objects, POJOs, SQL databases, Web Services and XML.

https://en.wikipedia.org/wiki/BIRT_Project

http://developer.actuate.com/about/

2 ReportDSL

ReportDSL generates the vaddin s ui and birt report.

The main semantic elements of the ReportDSL are:

- “package” – the root element that contains all the other elements. A model can contain multiple packages.
- “import” declarations – used to import external models or even Java classes.
- “externalCssURI” – define the optional external CSS URI for all report .
- “pagetemplate” – define the report page template, e.g. page size, orientation, margin, header, footer, etc.
- “formatters” – define the formatter types for all report .
- “colors” – define the color types for all report .
- “fonts” – define the font types for all report .
- “medias” – define the media types for all report .
- “styles” – define the style types for all report .
- “report” – define the data source details for the report, e.g. row/column configurations, legend details and tooltips configurations.

2.1 Syntax

2.1.1 Package definition

```
package <package name>  {
      [import <import models/class name>]
      ...
      [externalCssURI <URI string> [from bundle <bundle string>]]
      [pagetemplate <pagetemplate name>
       {
            ...
      }]
      ...

      formatters {...}
      colors {...}
      fonts{...}
      medias{...}
      styles{...}

      [report <report name>
      {
            ...
      }]
      ...
}
```

- **externalCssURI is used to define the external CSS for the all report in this package.**
- **pagetemplate is used to define the page template formate for all report in this package.**
- **User can also define different formatters, colors, fonts, medias and styles template for all reports in this package. All these definitions could be defined in a separated package and import to other report-packages as well.**
- **For each report definition, a *<reportname>+Report.java* file will be generated, in which a java class named *<reportname>+Report* extended from java class `CCBaseReport` is defined. In this class, report and report configurations are defined. Additionally a *<reportname>.rptdesign* file will be also gererated, in which a birt report is defined.**

e.g. birt report layout:

supplier_report

company	headquarter city	headquarter zipcode	rating
Company1	A Jupiters beDescending	112345	-1
Company2	San Francisco	99786	-1
Company3	Moon Helios	99786	-1
Company4	Moon	112345	-1
Company5	Jupiter	4e557	5
Company7	Munich	112345	3
Company8	Moon	99786	-1
Company9	San Francisco	7775423	-1
Company10	Jupiter	7775423	-1
Company11	Munich	112345	-1
Company12	Jupiter	7775423	-1
Company13	San Francisco	99786	-1
Company14	Moon	112345	-1
Company15	San Francisco	7775423	-1
Company16	Jupiter	4e557	-1
Company17	Moon or Sun	7775423	-1
Company18	Jupiter	4e557	-1
Company19	Vienna	4e557	-1
Company20	San Francisco	99786	-1
Company21	Jupiter	99786	-1
Company22	Vienna	7775423	-1
Company23	San Francisco	7775423	-1
Company24	Moon	4e557	-1
Company25	Moon	7775423	-1
Company26	Moon	76844	-1
Company27	Jupiter	4e557	-1
Company28	Jupiter	7775423	-1
Company29	Munich	7775423	-1
Company30	Jupiter	7775423	-1
Company31	Moon	4e557	-1
Company32	San Francisco	112345	-1
Company33	Vienna	76844	-1
Company34	Jupiter	4e557	-1
Company35	San Francisco	112345	-1
Company36	Jupiter	7775423	-1
Company37	Munich	76844	-1
Company38	Moon	76844	-1
Company39	Vienna	76844	-1
Company40	Vienna	76844	-1
Company41	Jupiter	112345	-1

2.1.2 CCElement

This element is used more than once in the package definition, syntax is as following defined:

```
CCElement:
      title <title name> [style <existed Style ID>]
      | subtitle <subtitle name> [style <existed Style ID>]
      | subsubtitle <subsubtitle name> [style  <existed Style ID>]
      | label <label name> [style <existed Style ID>]
      | text <text> [style <existed Style ID>]
      | autotext AutoTextEnum [style <existed Style ID>]
      | image ImageSizeEnum [scale <number>] ImageSourceEnum <file name>
      | datamarttable <existed Datamart> [as <datamart table ID>] [style
      <existed Style ID>] {
            [detailheader [style <existed Style ID>] ]
            [detailfooter [style <existed Style ID>] ]
            [group <group ID> by ([<existed Datamart property> {
                  (header [style <existed Style ID>] {
                        row [as <row ID>] [style <existed Style ID>] {
                              cell [as <cell ID>] [columnspan <number>] {
                                    CCElement
                              }
                        }
                  }]
                  [detailheader [style <existed Style ID>] ]
                  [detailfooter [style <existed Style ID>] ]
                  [footer [style <existed Style ID>] {
                        row [as <row ID>] (style <existed Style ID>] {
                              cell [as <cell ID>] [columnspan <number>] {
                                    CCElement
                              }
                        }
                  }]
            }]
            details [style <existed Style ID>] [all |
            [{CCDatamartTableProperty}]]
      }
      | CCDatamartTableProperty
      | grid [as <grid ID>] {
            row [as <row ID>] [style <existed Style ID>] {
                  cell [as <cell ID>] [columnspan <number>] {CCElement}
            }
      }
CCDatamartTableProperty:
      property <existed Datamart property>
      |aggregate
            CCPureAggregationType
            | CCUnaryAggregationType of <existed Datamart property>
            | concat of ([<existed Datamart property>] )
            | divide of (<existed Datamart property> <existed Datamart
            property>)
      [on-group <datamart group ID>] [title <title name>]
            [style <existed Style ID>]
            [image path <path name> [[dynamic] [hidelabel] [resize <size>]]]
            [intervals [hidelabel] {
                  [up to <interval number> CCTableRangeElement
                  |days in past <interval number>CCTableRangeElement]
            }]
            [lookups [hidelabel] {
                  [number <lookup number> CCTableRangeElement
```

```
                |string <lookup string> CCTableRangeElement
                |days in past <lookup number> CCTableRangeElement]
        }]

CCTableRangeElement:
      textcolor <existed Color ID>
      | cellcolor <existed Color ID>
      | icon <icon name>
      | trend TrendIconEnum


AutoTextEnum: page-number | total-page | page-number-unfiltered | total-page-
unfiltered | page-variable
ImageSizeEnum: size-to-image | scale-to-item | clip
ImageSourceEnum: file | url | embed
CCPureAggregationType : count | running-count
CCUnaryAggregationType: sum | average | minimum | maximum | running-sum
TrendIconEnum: rising | bad-rising | sloping | good-sloping | stagnating
```

☞ **title, subtitle, subsubtitle, label text, autotext, image, datamarttable, datamarttableproperty and grid could be defined detailed using/ not using existed style template in this element.**

☞ **All details of report are defined in xx.rptdesign file according to birt report syntax and semantics.**

e.g.:

```
title "Employee Salary" style headerarea
datamarttable EmployeeSalary as salaryGrouped style bootstrap {
      detailheader style grouping_1_head
      group byEducationLevel by education_level {
            header style grouping_1_head {
                  row {
                        cell columnspan 7 {property education_level}
                  }
            }
            footer style grouping_1_foot { row {} }
      }
      group byPositionTitle by position_title {
            header style grouping_2_head {
                  row {
                        cell {}
                        cell {property position_title}
                  }
            }
            detailheader style grouping_2_head
            footer style grouping_2_foot {
                  row {
                        cell columnspan 4 {label "sum"}
                        cell {aggregate count style integer}
                        cell {}
                  }
                  row {
                        cell columnspan 5 {label "average"}
                        cell {aggregate average of min_scale style ^currency}
                        cell {aggregate average of salary style ^currency}
```

```
                        cell {aggregate average of max_scale style ^currency}
                  }
                  row style defaultrow { cell {} }
            }
      }
      details style defaultrow {
            property full_name
            property gender lookups hidelabel {
                  string "M" icon "gender_male"
                  string "F" icon "gender_female"
            }
            property marital_status lookups hidelabel {
                  string "M" icon "marital_married"
                  string "S" icon "marital_single"
            }
            property min_scale style ^currency
            property salary style ^currency intervals {
                  up to 2000 textcolor red
                  up to 2500 cellcolor orange
                  up to 2500 textcolor darkblue
                  up to 3000 cellcolor white
                  up to 100000 cellcolor black
                  up to 100000 textcolor lightblue
            }
            property max_scale style ^currency
            property hire_date style ^date
      }
}
```

2.1.3 Pagetemplate

Pagetamplete is used to define the basic format for different page layouts, it is defined as following:

```
pagetemplate <pagetemplate name> {
      type PageSizeEnum
      orientation OrientationEnum
      topmargin <topMargin number> UnitEnum
      leftmargin <leftMargin number> UnitEnum
      bottommargin <bottomMargin number> UnitEnum
      rightmargin <rightMargin number> UnitEnum
      [header {
            [showOnLast]
            height <height number> UnitEnum
            CCElement
      }]
      [footer {
            [showOnLast]
            height <height number> UnitEnum
            CCElement
      }]
}

UnitEnum: mm | cm | pt | inch | pc | em | ex | px | %
PageSizeEnum: a4 | a3 | a5 | us-letter | us-legal | us-ledger | us-super-b
OrientationEnum: portrait | landscape
```

☞ **type is used to define the page size, e.g. a4, a3, ect.**

☞ **orenetation is used to define the page orientation, e.g. portrait or landscape.**

☞ **topmargin, leftmargin, bottommargin and rightmargin are used to define the margin size.**

☞ **Header and footer are used to define the header and footer details, e.g.: height, if it will be show on last, ect.**

e.g.:

```
pagetemplate A4_Landscape {
	type a4
	orientation landscape
	topmargin 11.5 mm
	leftmargin 12 mm
	bottommargin 13 mm
	rightmargin 14 mm
	header {
		// showOnFirst
		height 0.6 inch
		label "a A4 landscape page"
	}
	footer {
		showOnLast
		height 10 mm
		grid {
			row {
				cell {label "empty"}
			}
			row {
				cell {label "page"}
				cell {autotext page-number}
				cell {label "of"
				cell {autotext total-page}
				cell {label "end of the page"
			}
		}
	}
}
```

2.1.4 Formatters

Formatters is used to define the report format in detail, it is defined as following:

```
formatters {
	[CCSomeFormat]
}

CCSomeFormat:
	uomo <uomo ID>{
		ui <ui name> report <string> textalign TextAlign
	}
	| number <number ID>{
		ui <ui name> report NumberFormatCategory <string> textalign
		TextAlign
	}
	| currency <currency ID>{
```

```
            ui <ui name> report CurrencyFormatCategory <string> textalign
            TextAlign
      }
      | date <date ID>{
            ui <ui name> report custom <string> textalign TextAlign
      }
      | date+time <date+time ID>{
            ui <ui name> report custom <string> textalign TextAlign
      }
      | time <time ID>{
            ui <ui name> report custom <string> textalign TextAlign
      }

TextAlign: left | center | right
NumberFormatCategory:unformatted|general-number|fixed|percent|scientific|custom
CurrencyFormatCategory: unformatted | currency | custom
```

☞ **The defined formatters are usually used in the definition for styles.**

☞ **ui and report defines the layout format in ui and report.**

☞ **textalign defines the text align definition on page.**

e.g.:

```
formatters {
      /** empty UOMO format */
      uomo uomo_test {
            ui "test"
            report "test"
            textalign left
      }

      /** money in EURO, rounding half-up */
      currency money_euro {
            ui "###,##0.00 €"
            report currency "#,##0.00 EUR{RoundingMode=HALF_UP}"
            textalign right
      }

      /** date without time with weekday */
      date+time shortdate {
            ui "SHORTDATE"
            //report custom "EE dd. MM.yyyy"
            report custom "dd.MM.yyyy"
            textalign left
      }

      /** integer */
      number integer {
            ui "##0"
            report percent "##0"
            textalign right
      }

      /** percent */
      number ^percent {
            ui "##0.0'%'"
```

```
            report percent "##0.0'%'"
            textalign right
      }

}
styles {
      /** Bootstrap style */
      style bootstrap {
            textcolor schwarz
            backgroundcolor weiss
            font arial_8pt_regular
            media big {
                  font arial_12pt_regular
            }

      }
      /** Standardzeile (unterste Ebene) */
      style defaultrow {
            extends bootstrap
            font arial_8pt_regular
            backgroundcolor weiss alternate hellblau
            padding-left 2 pt
            padding-right 2 pt
            media big {
                  font arial_12pt_regular
            }

      }
      /** if a currency has to be rendered */
      style ^currency {
            extends defaultrow
            formatter money_euro
      }

}
```

2.1.5 Colors

Colors is used to define the report colors in detail, it is defined as following:

```
colors {
       [color <color name>
            ["<color string>"
            | darkens <existed Color ID> by-percent <number>
            | lightens <existed Color ID> by-percent <number>
            | transforms <existed Color ID> by-percent <number> towards <existed
      Color ID>
            ]
      ]

}
```

☞ **The defined colors are usually used in the definition for styles.**

☞ **color is defined with the RGB string.**

☞ **darkens, lightens, and transforms could be used to defined the different color darkness of existed color type.**

e.g.:

```
colors {
	/** the basic text color: BLACK */
	color schwarz "#000000"
	/** grau rgb(128,128,128) */
	color grau_128 "#808080"
	/** grau rgb(136,136,136) */
	color grau_136 "#888888"
	/** grau rgb(220,220,220) */
	color grau_220 "#dcdcdc"
	/** grau rgb(235,235,235) */
	color grau_235 "#ebebeb"
	/** grau rgb(250,250,250) */
	color grau_250 "#fafafa"
	/** weiß */
	color weiss "#ffffff"
	/** hellblau, laut Style-Guide eigentlich rgb(244,252,255), aber am
Bildschirm zu schwach! */
	color hellblau "#e4ecff"
	color red "#ff3b30"
	color orange "#ffcc00"
	color darkblue "#007aff"
	color white darkens grau_136 by-percent 50
	color black lightens schwarz by-percent 30
	color lightblue "#52edc7"

}
styles {
	/** Bootstrap style */
	style bootstrap {
		textcolor schwarz
		backgroundcolor weiss
		font arial_8pt_regular
		media big {
			font arial_12pt_regular
		}

	}
}
```

2.1.6 Fonts

Fonts is used to define the report fonts in detail, it is defined as following:

```
fonts {
	[font <font name> {
		[extends <existed Font ID>]
		[family BuildInFontFamily | <string>]
		[FontStyle]
		[bold]
		[size <size number> UnitEnum]
	}]

}
BuildInFontFamily: monospace | sans-serif | serif
FontStyle: normal | italic | oblique
UnitEnum: mm | cm | pt | inch | pc | em | ex | px | %
```

☞ **The defined fonts are usually used in the definition for styles.**

☞ **New font could be defined based on existed font type using extends.**

☞ **family is used to define type of font and if it is monospace or sans-serif or serif.**

☞ **If it is normal or italic or oblique or bold could be also defined directly.**

☞ **size is used to define the font size.**

e.g.:

```
fonts {
	font arial_14pt_bold {
		family "Arial"
		bold
		size 14 pt
	}

	font arial_12pt_regular {
		family "Arial"
		normal
		size 12 pt
	}

	font arial_12pt_bold {
		family "Arial"
		bold
		size 12 pt
	}

	font arial_8pt_regular {
		family "Arial"
		normal
		size 8 pt
	}

	font arial_7pt_bold {
		family "Arial"
		bold
		size 7 pt
	}
}
styles {
	/** Bootstrap style */
	style bootstrap {
		textcolor schwarz
		backgroundcolor weiss
		font arial_8pt_regular
		media big {
			font arial_12pt_regular
		}

	}
}
```

2.1.7 Medias

Medias is used to define the report media in detail, it is defined as following:

```
medias {
       [media <media name>]

}
```

☞ **The defined formatters are usually used in the definition for styles.**

☞ **It defines if the same report used in different kind of workplace, how to layout. E.g. in warehouse, it should be print with bigger size of font, ect.**

e.g.:

```
medias {
      /** Formatierungen für tabellarische Reports z.B. zur Nutzung im Büro */
      media middle
      /** Formatierungen für tabellarische Reports z.B. zur Nutzung im Lager */
      media big
      /** Formatierungen für tabellarische Reports z.B. zur Nutzung für Büro
und Ablage */
      media small
      /** Formatierungen für Reports wie Rechnungen, Lieferscheine etc. */
      media ^report

}
styles {
      /** Bootstrap style */
      style bootstrap {
            textcolor schwarz
            backgroundcolor weiss
            font arial_8pt_regular
            media big {font arial_12pt_regular}
      }

      /** header */
      style headerarea {
            extends bootstrap
            font arial_10pt_regular
            media big {font arial_12pt_regular}
            media small {font arial_8pt_regular}

      }

}
```

2.1.8 Styles

Formatters is used to define the report format in detail, it is defined as following:

```
styles {
       [style <style name> {
            [extends <existed Style ID>]
            [ [formatter <existed Formatter ID>]
              [font <existed Font ID>]
              [backgroundcolor <existed Color ID> [alternate <existed Color ID>]
      ]
              [textcolor <existed Color ID>]
              [textalign TextAlign]
              [border-top CCBorderStyle]
              [border-bottom CCBorderStyle]
```

```
                [border-left CCBorderStyle]
                [border-right CCBorderStyle]
                [padding-top <paddingTop number> UnitEnum]
                [padding-bottom <paddingBottom number> UnitEnum]
                [padding-left <paddingLeft number> UnitEnum]
                [padding-right <paddingRight number> UnitEnum]
                [margin-top <marginTop number> UnitEnum]
                [margin-bottom <marginBottom number> UnitEnum]
                [margin-left <marginLeft number> UnitEnum]
                [margin-right <marginRight number> UnitEnum]
            ]
            [media <existed Media ID>{
                [ [formatter <existed Formatter ID>]
                  [font <existed Font ID>]
                  [backgroundcolor <existed Color ID> [alternate <existed Color
            ID>] ]
                  [textcolor <existed Color ID>]
                  [textalign TextAlign]
                  [border-top CCBorderStyle]
                  [border-bottom CCBorderStyle]
                  [border-left CCBorderStyle]
                  [border-right CCBorderStyle]
                  [padding-top <paddingTop number> UnitEnum]
                  [padding-bottom <paddingBottom number> UnitEnum]
                  [padding-left <paddingLeft number> UnitEnum]
                  [padding-right <paddingRight number> UnitEnum]
                  [margin-top <marginTop number> UnitEnum]
                  [margin-bottom <marginBottom number> UnitEnum]
                  [margin-left <marginLeft number> UnitEnum]
                  [margin-right <marginRight number> UnitEnum]
                ]

            }]

    }]
}

CCBorderStyle:
      BorderType <width number> UnitEnum <existed Color ID>

TextAlign: left | center | right
UnitEnum: mm | cm | pt | inch | pc | em | ex | px | %
BorderType: none | solid | dotted | dashed | double | groove | ridge | inset |
outset
```

☞ **New style could be defined based on existed style type using extends.**

☞ **formatter, font, backgroundcolor, textcolor could be defined using existed definitions.**

☞ **Textalign, border details, padding details and margin details could be also defined in detail.**

☞ **Formatter, font, color, textalign, border details, padding details and margin details are redefinedable for existed media.**

e.g.:

```
styles {
```

```
/** Bootstrap style */
style bootstrap {
	textcolor schwarz
	backgroundcolor weiss
	font arial_8pt_regular
	media big {
		font arial_12pt_regular
	}
}

/** Headline */
style headline {
	extends bootstrap
	font arial_14pt_bold
}

/** header */
style headerarea {
	extends bootstrap
	font arial_10pt_regular
	media big {
		font arial_12pt_regular
	}
	media small {
		font arial_8pt_regular
	}
}

/** Tabellenkopf */
style tabellenkopf {
	extends bootstrap
	font arial_10pt_bold
	backgroundcolor grau_220
	border-top solid 1 pt schwarz
	border-bottom solid 1 pt schwarz
	border-left solid 1 pt grau_128
	border-right solid 1 pt grau_128
	media big {
		font arial_12pt_bold
	}
}

/** 1. Gruppierungsebene Kopfzeile */
style grouping_1_head {
	extends bootstrap
	font arial_10pt_bold
	backgroundcolor grau_235
	border-top solid 1 pt schwarz
	border-bottom solid 1 pt grau_136
	media big {
		font arial_12pt_bold
	}
	media small {
		font arial_8pt_bold
	}
}
```

```
/** 2. und weitere Gruppierungsebenen Kopfzeile */
style grouping_2_head {
	extends bootstrap
	font arial_10pt_bold
	backgroundcolor grau_250
	border-top solid 1 pt grau_136
	media big {
		font arial_12pt_bold
	}
	media small {
		font arial_8pt_bold
	}
}

/** Standardzeile (unterste Ebene) */
style defaultrow {
	extends bootstrap
	font arial_8pt_regular
	backgroundcolor weiss alternate hellblau
	padding-left 2 pt
	padding-right 2 pt
	media big {
		font arial_12pt_regular
	}
}

/** if a currency has to be rendered */
style ^currency {
	extends defaultrow
	formatter money_euro
}

/** if a date with weekday has to be rendered */
style ^date {
	extends defaultrow
	formatter shortdate
}

/** a pure integer value */
style integer {
	extends defaultrow
	formatter integer
}

/** 2. und weitere Gruppierungsebenen Fußzeile */
style grouping_2_foot {
	extends bootstrap
	font arial_10pt_regular
	backgroundcolor grau_250
	border-top solid 1 pt grau_136
	border-bottom solid 1 pt grau_136
	media big {
		font arial_12pt_regular
	}
	media small {
		font arial_8pt_regular
	}
```

```
	}

	/** 1. Gruppierungsebene Fußzeile */
	style grouping_1_foot {
		extends bootstrap
		font arial_10pt_regular
		backgroundcolor grau_235
		border-top solid 1 pt schwarz
		border-bottom solid 1 pt schwarz
		media big {
			font arial_12pt_regular
		}
		media small {
			font arial_8pt_regular
		}
	}

	/** Tabellenende */
	style tabellenende {
		extends bootstrap
		font arial_10pt_regular
		backgroundcolor weiss
		border-top solid 1 pt schwarz
		border-bottom solid 1 pt schwarz
		media big {
			font arial_12pt_regular
		}
		media small {
			font arial_8pt_regular
		}
	}

	/** Fußzeile */
	style fusszeile {
		extends bootstrap
		font arial_7pt_bold
	}
}
```

2.1.9 Report

Formatters is used to define the report format in detail, it is defined as following:

```
report <report name> [described by <discription>] {
	[datamart <datamart name>]
	rendering RenderingEnum
	[externalCssURI <Css URI string> [from bundle <bundle string>] ]
	pagetemplate <existed PageTemplate ID>
	media <existed Media ID>
	file <file name>
	| template {
		[described by <discription>]
		[header {
			[showOnLast]
			height <height number> UnitEnum
			CCElement
		}]
		detail { CCElement }
```

```
            [footer {
                  [showOnLast]
                  height <height number> UnitEnum
                  CCElement
            }]
      }
}

UnitEnum: mm | cm | pt | inch | pc | em | ex | px | %
RenderingEnum: html | pdf
```

☞ **Generate a <*reportname*>+*Report.java* file, in which a java class named <*reportname*>+*Report* extended from java class `CCBaseReport` is defined. In this class, report and report configurations are defined.**

☞ **Generate a <*reportname*>.*rptdesign* file, in which a birt report is defined.**

☞ **rendering is used to define the output format of report, e.g.: HTML or PDF.**

☞ **If the existed report should be used, it could be defined with keyword file with path+filename string.**

☞ **template is used to define the header, footer and all other specifically detail for the report.**

e.g.:

```
report EmployeeSalaryViaFile {
      /** datamart Employee Salary */
      datamart EmployeeSalary
      /** rendering als HTML, auch PDF möglich */
      rendering html
      pagetemplate A4_Landscape
      media small
      /** das rptDesign verwenden */
      file "rptdesign/CCReportFile.rptdesign"
}

report PlainText {
      datamart EmployeeSalary
      rendering html
      /** an optional external Css URI */
      externalCssURI "css/html/ccng-birt2.css" from bundle
"FoodMartReportDSLPlugin2"
      pagetemplate A4_Portrait
      media big
      template {
            detail {
                  title "it's not me"
                  label "with the logo via url..."
                  image size-to-image url "http://www.compex-
commerce.com/export/system/modules/de.compex.internet/resources/img/logo/logo_c
ompex-commerce.gif"
                  label "now the logo via file..."
                  image size-to-image file
"platform:/plugin/de.compex.utils/img/ccngdesign/compexlogo.png"
                  label "and what about the logo embedded?"
                  image size-to-image embed
"platform:/plugin/de.compex.utils/img/ccngdesign/compexlogo.png"
```

```
                subtitle "sub title #1"
                label "it's me again, but hopefully running, and modified too
:/"
                subtitle "sub title #2"
                label "this label has more to say or what?.."
                text "a text will be used to display some data!"
            }
            footer {
                showOnLast
                height 20 mm
                label "there is no page number here, sorry!"
            }
        }
}
```

3 TODO

For the updated version of ReportDSL, this file should be also accordingly modified.