

CAMSEG Report Language

API Documentation

Version 2.0

The CAMSEG Report Language (CRL) is the language used to create reports in CAMSEG SCM. It's based on HTML, with special language tags. To be recognized by CAMSEG SCM, a report has to be in the Reports directory of the application. All reports are organized in a directory structure.

1. Base Specifications

Every report is a HTML file with special tags. The `<?crl ?>` tags contains JavaScript code that is used to generate the report. For example, loops are used in many reports where the number of rows is variable.

1.1 Base JavaScript Functions

There are three major functions that are essential to generate the reports :

```
print( )
```

This function prints text to the report. For example, the code line `<?crl print("Hello World !"); ?>` would be replaced by `Hello World` in the HTML code.

```
tr( )
```

This function translate the text in the application's current language. For example, in a French CAMSEG SCM, the code line `<?crl print(tr("Hello")); ?>` would be replaced by `Salut` in the HTML code. All the literal strings should be translated by the `tr()` function. Note : the translated strings can be translated in the Report Traductor dialog of CAMSEG SCM.

```
query( )
```

This function is used to access the data of the campaign. It accepts CRL Queries (see section 2). For example, if the current campaign is names *Test*, the code line `<?crl print(query("Campaign.Name")); ?>` would be replaced by `Test` in the HTML code.

1.2 The `<meta>` tags

All reports should have at least the following meta tags in its `<head>` section :

```
<meta name="Content-Type" content="text/html; charset=UTF-8" /> : The charset
<meta name="report_name" content='NAME'> : The report name
<meta name="report_author" content='AUTHOR'> : The author name
<meta name="report_description" content='DESCRIPTION'> : A description of the report (optional)
<meta name="report_version" content='VERSION'> : The report version
<meta name="report_icon" content='ICON'> : The report icon (used in the menus). It should be in the Icons
directory of CAMSEG SCM.
```

Note : the only tag allowing CRL Queries is `report_name`, and the only available modules are `App` and `Strings`

A report can be *Single* or *Multiple*. In a *Single* report, there is only one version of the report (ex : Customer Report), whereas in a *Multiple* report, there is as much reports as there are objects on the selected type in the campaign (ex : Sales by Customer). When generating a Multiple report, a dialog will be prompted to the user to select an object of the selected type.

To define a report multiple, you need to include the following meta tags :

```
<meta name="count_type" content="multiple" />
<meta name="object_type" content="Customer" />
```

The `object_type` tag contains the object type. The following types are allowed :

Customer	PaymentMode
Vendor	Supplier
Service	Tax
Product	User
PLU	Delivery
CustomerCat	StockAdjustment
VendorCat	CashAdjustment
ProductCat	Discount
PLUCat	Unit
Order	Donation
Payment	Donator

In *Multiple* reports, the `OBJECT_ID` variable contains the ID of the selected object. For example, if the type is *Customer*, it's possible to get the selected customer's name with the following code :

```
query("Campaign.Customer.OBJECT_ID.Name").
```

1.3 CSS Themes

When the report is generated, a link to the campaign's report theme (CSS file) will be included in the `<head>` section.

The CSS theme file defines some base classes and IDs :

- `#infos` The report's top-right informations
- `#inv_org` In an invoice, it's used to display the organization's infos (name, address, etc.). It's located in the top-left corner of the report.
- `#inv_billto` In an invoice, it's used to display the bill-to organization's infos (name, address, etc.). It's located under the `#inv_org`.
- `#inv_type` In an invoice, it's used to display the invoice type (Order, Invoice, ...). It's located in the top-right corner.
- `tr.alt` The alternate row background color. Most base reports use this system to have an alternating row background (ex : white, gray, white, gray, etc.). If used in a loop, use the following code to have alternating row backgrounds :

```
if (i % 2 == 0) {
    print("<tr>");
} else {
    print("<tr class='alt'>");
}
```
- `td.sumlabel` This class is used for the summary rows at the bottom of some tables, when the cell shouldn't have any borders.

2. Report Language (CRL)

This language is used in the report tags. It can access the data, traduce strings, etc.

2.1 Basic types

It can contain :

- Strings : "ABC"
- Decimal values : 123
- Language Queries : `Campaign.Customer.Count`

2.2 Operations

2.2.1 Numerical operations

The following operations can be used with numerical values :

- Addition : `123 + 123`
- Subtraction : `1234 - 123`
- Multiplication : `123 * 123`
- Division : `1234 / 123`

If a value returned by a language query is a number, it can be used in operations.

Ex : `Campaign.Vendor.Count + 1` will return 8 if there's 7 vendors.

But, if the returned value is not a numerical, it will simply concatenate the values :

Ex : `Campaign.Customer.1.Name + 2` will return "Doe2" if the customer's name is "Doe"

2.2.2 String operations

The only operation that can be used on strings is the concatenation (+)

Ex : `Campaign.Customer.1.Name + ", " + Campaign.Customer.FirstName` will return "Doe, John"

2.3 Language Queries

Main Modules :

Campaign : Data about the campaign

Report : Data about the report infos

App : Data about the application

Strings : Translations

Date : Current date

Time : Current time

2.3.1 The Campaign module

See Annexe II for this module

2.3.2 The Report Module

Query	Description
<code>Report.Name</code>	The report name
<code>Report.Icon</code>	The report icon
<code>Report.Version</code>	The report version
<code>Report.Author</code>	The author of the report
<code>Report.Description</code>	The report description

2.3.3 The App Module

Query	Description
<code>App.Name</code>	The application name
<code>App.Version</code>	The application version
<code>App.Codename</code>	The application codename
<code>App.LibVersion</code>	The library version

2.3.3 The Strings Module

Query	Description
<code>Strings.tr("ABC")</code>	This function returns a translation of the string passed in parameter (here, it would return a translated version of "ABC"). The language used for the translation is the same as the current application language.

2.3.3 Other modules

Query	Description
<code>Date</code>	Returns the current date
<code>Time</code>	Returns the current time

Annexe I : Report Example (Customer Report)

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
  <meta name="Content-Type" content="text/html; charset=UTF-8" />
  <meta name="report_name" content='Strings.tr("Customer Report")' />
  <meta name="report_author" content="CAMSEG Technologies" />
  <meta name="report_description" content="" />
  <meta name="report_version" content="2.0" />
  <meta name="report_icon" content="clients.png" />
</head>
<body>
  <h1><?crl print(query("Report.Name")); ?></h1>
  <h2><?crl print(query("Campaign.Name")); ?></h2>
  <div id="infos">
    <?crl print(query("Campaign.Customer.Count") + " " + tr("customers")); ?>
  </div>
  <br />
  <table>
    <tr>
      <th><?crl print(tr("Code")); ?></th>
      <th><?crl print(tr("Name")); ?></th>
      <th><?crl print(tr("First Name")); ?></th>
      <th><?crl print(tr("Category")); ?></th>
      <th><?crl print(tr("Total Amount")); ?></th>
      <th><?crl print(tr("Amount With Taxes")); ?></th>
    </tr>
    <?crl
for (var i = 0; i < query("Campaign.Customer.Count"); ++i)
{
  if (i % 2 == 0)
  {
    print("<tr>");
  }
  else
  {
    print("<tr class='alt'>");
  }

  print("<td>");
  print(query("Campaign.Customer." + i + ".ID"));
  print("</td>");

  print("<td>");
  print(query("Campaign.Customer." + i + ".Name"));
  print("</td>");

  print("<td>");
  print(query("Campaign.Customer." + i + ".FirstName"));
  print("</td>");

  print("<td>");
  print(query("Campaign.Customer." + i + ".Category"));
  print("</td>");

  print("<td>");
  print(query("Campaign.Customer." + i + ".TotalAmount"));
  print("</td>");

  print("<td>");
  print(query("Campaign.Customer." + i + ".AmountWithTaxes"));
  print("</td>");

  print("</tr>");
}
?>
</table>
</body>
</html>
```

Annexe II : Campaign Module Queries

1. Global Queries

Query	Description
Campaign.Organization.Name	Returns the organization name
Campaign.Organization.Website	Returns the organization website URL
Campaign.Organization.Picture	Returns the organization picture
Campaign.Organization.Location.value	Returns the organization location (replace <i>value</i> by a location type, such as Address)
Campaign.Name	Returns the campaign name
Campaign.TotalCashAdjustments	
Campaign.TotalVendorPay	
Campaign.TotalExpectedDeliveryCosts	
Campaign.TotalCurrentDeliveryCosts	
Campaign.TotalGrossSales	
Campaign.TotalExpectedCosts	
Campaign.TotalCurrentCosts	
Campaign.TotalExpectedRevenues	
Campaign.TotalCurrentRevenues	
Campaign.TotalExpectedProfits	
Campaign.TotalCurrentProfits	
Campaign.TotalExpectedProfitability	
Campaign.TotalCurrentProfitability	
Campaign.TotalTaxes	
Campaign.TotalProceeds	
Campaign.TotalAmountPaid	
Campaign.TotalAmountDue	
Campaign.PercentagePaid	
Campaign.ActualStockValue	
Campaign.TotalActualStock	