RDLC Report Custom Code Utility Functions

This module provides comprehensive utility functions for RDLC reports including string manipulation, number conversion to words with Indian currency format, global data management, and logging capabilities.

Source: Merged from frontlook-admin/RDLCReport_CustomCode

Table of Contents

Getting Started - Setup Guide

Follow these steps to implement global data management in your RDLC reports.

Step 1: Add Custom Code Functions

Open your RDLC report in SQL Report Builder, go to **Report Properties** → **Code** tab, and paste the complete code from `RdlcReportCode.vb` file.

The code includes:

- Global variables (`GlobalDict`, `Data1`, `Data2`, `Data3`)
- `SetGlobalData()` and `GetVal()` functions (improved approach)
- `SetData()` and `GetData()` functions (legacy NAV way)
- Helper functions: `AddKeyValue()`, `SetDataAsKeyValueList()`, `GetVal2()`
- Logging functions
- String concatenation functions
- Number to words conversion functions

Step 2: Create Key-Value List in C/AL or AL

Add this helper procedure to your C/AL or AL code:

```
local procedure AddKeyValue(VAR KeyValueListAsText: Text; _Key: Text; _Value:
Text)
var
    Chr177: Text[1];
    NewPair: Text;
begin
    Chr177[1] := 177;
    NewPair := _Key + Chr177 + _Value + Chr177;
    KeyValueListAsText += NewPair;
end;
```

Create a procedure to build your global data fields:

```
local procedure GetGlobalDataFields(SalesHeader : Record "Sales Header"; Addr :
Array[8] of Text) KeyValueList : Text
begin
    AddKeyValue(KeyValueList, 'CompanyName', CompanyInfo.Name);
    AddKeyValue(KeyValueList, 'CompanyAddress', CompanyInfo.Address);
    AddKeyValue(KeyValueList, 'Address1', Addr[1]);
    AddKeyValue(KeyValueList, 'Address2', Addr[2]);
    AddKeyValue(KeyValueList, 'ReportDate', Format(Today));
    AddKeyValue(KeyValueList, 'ReportTitle', 'Sales Invoice');
    // Add more fields as needed
end;
```

Step 3: Add Dataset Column

Add the key-value list as a column in your dataset:

In AL:

```
dataset
{
    dataitem("Sales Header"; "Sales Header")
    {
        [...]
        column(GlobalData; GetGlobalDataFields("Sales Header", Addr))
        { }
        [...]
    }
}
```

In C/AL:

```
Sales Header - OnAfterGetRecord()
GlobalData := GetGlobalDataFields("Sales Header", Addr);
```

Step 4: Add Hidden Control in RDLC

Option A: Manual XML Edit (Recommended)

1. Open your `.rdl` or `.rdlc` file in a text editor
2. Search for `<ReportItems>`
3. Paste the following XML code below it (inside your main tablix if needed):

```
<Tablix Name="SetGlobalDataTable">
  <TablixBody>
    <TablixColumns>
      <TablixColumn>
        <Width>0.3cm</Width>
      </TablixColumn>
    </TablixColumns>
    <TablixRows>
      <TablixRow>
        <Height>0.3cm</Height>
        <TablixCells>
          <TablixCell>
            <CellContents>
              <Textbox Name="SetGlobalDataTextbox">
                <CanGrow>true</CanGrow>
                <KeepTogether>true</KeepTogether>
                <Paragraphs>
                  <Paragraph>
                    <TextRuns>
                      <TextRun>
                        <Value />
                        <Style />
                      </TextRun>
                    </TextRuns>
                    <Style />
                  </Paragraph>
                </Paragraphs>
                <rd:DefaultName>SetGlobalDataTextbox</rd:DefaultName>
                <Visibility>
                  <Hidden>=Code.SetGlobalData(Fields!GlobalData.Value)</Hidden>
                </Visibility>
                <Style>
                  <Border>
                    <Style>None</Style>
                  </Border>
                </Style>
              </Textbox>
            </CellContents>
```

```
          </TablixCell>
        </TablixCells>
      </TablixRow>
    </TablixRows>
  </TablixBody>
  <TablixColumnHierarchy>
    <TablixMembers>
      <TablixMember />
    </TablixMembers>
  </TablixColumnHierarchy>
  <TablixRowHierarchy>
    <TablixMembers>
      <TablixMember>
        <Group Name="Details" />
      </TablixMember>
    </TablixMembers>
  </TablixRowHierarchy>
  <DataSetName>DataSet_Result</DataSetName>
  <Height>0.3cm</Height>
  <Width>0.3cm</Width>
  <Style>
    <Border>
      <Style>None</Style>
    </Border>
  </Style>
</Tablix>
```

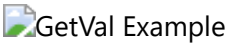
Option B: Using Report Builder

1. Add a small textbox in the body section
2. Set its `Hidden` property to: `=Code.SetGlobalData(Fields!GlobalData.Value)`
3. Make it very small (0.3cm x 0.3cm) and position it where it won't interfere

Step 5: Use GetVal to Retrieve Data

Now you can use the data in your report headers, footers, or body:

```
=Code.GetVal("CompanyName")'
=Code.GetVal("CompanyAddress")'
=Code.GetVal("ReportDate")'
=Code.GetVal("ReportTitle")'
```



⚠ **IMPORTANT:** Always end your expressions with an apostrophe (') or you will lose the arguments when copy & pasting textboxes from one instance of SQL Report Builder to another!

Why Do We Need This?

Understanding Report Rendering Order

RDLC reports render in a specific order:

1. **Body section** is rendered first
2. **Header and Footer** are rendered after the body

This creates a problem: How do you display data in the header/footer that depends on the current page's body content?

Solution: Use `SetGlobalData()` in a hidden control in the body to store values, then retrieve them in header/footer using `GetVal()`.

The Improvement Over NAV Way

The traditional NAV approach (`SetData`/`GetData`) has drawbacks:

Problem	Old NAV Way	New Improved Way
Finding Values	Must count position numbers	Use descriptive names
Readability	`=Code.GetData(5, 1)` - what is item 5?	`=Code.GetVal("CompanyName")` - clear!
Arguments	Two arguments (position, group)	One argument (name or index)
Maintenance	Hard to manage 3 separate lists	Single collection with named keys
Case Sensitivity	Case-sensitive	Case-insensitive keys

The Three Improvement Targets:

1. **Named Indexes** - Use `Microsoft.VisualBasic.Collection()` to support named keys instead of position numbers
2. **Single Argument** - `GetVal("Name")` instead of `GetData(5, 1)`
3. **Easier Maintenance** - Manage the field list in C/AL/AL procedures, not in RDLC

Logging Functions

WriteLog (Simple)

Writes a log message to a file with timestamp. Always generates a new path (stateless).

Parameters:

- `message` (String): The message to log
- `filePath` (String, Optional): Directory path for the log file. Default: `"C:\Temp"`
- `fileName` (String, Optional): Base name for the log file. Default: `"CliReportDebug_yyyyMMdd"`

Behavior:

- If no filename is provided, creates: `CliReportDebug_20251011.log`
- If filename is provided, creates: `CustomName_20251011.log`
- Automatically creates the directory if it doesn't exist
- Each log entry is timestamped with format: `yyyy-MM-dd HH:mm:ss.fff`
- Logging errors are silently ignored

```
' Default usage - logs to C:\Temp\CliReportDebug_20251011.log
WriteLog("This is a test message")

' Custom filepath - logs to D:\Logs\CliReportDebug_20251011.log
WriteLog("Custom path message", "D:\Logs")

' Custom filepath and filename - logs to D:\Logs\MyReport_20251011.log
WriteLog("Custom file message", "D:\Logs", "MyReport")
```

WriteLogCached (Performance)

Cached version of WriteLog for better performance with multiple log entries in same session.

Benefits:

- Caches log file path for repeated writes
- Automatically detects date changes and updates path
- Ideal for multiple log writes in same report execution

```
' Multiple logs in same session - more efficient
WriteLogCached("Starting report processing")
WriteLogCached("Processing item 1")
WriteLogCached("Processing item 2")
WriteLogCached("Report completed")
```

Global Data Management

Transfer data from report body to headers/footers using named key-value pairs.

SetGlobalData

Sets global data from a key-value list. Call this in a hidden tablix cell.

Usage in RDLC:

```
=Code.SetGlobalData(Fields!GlobalData.Value)
```

In C/AL or AL (to create the key-value list):

```
local procedure AddKeyValue(VAR KeyValueListAsText: Text; _Key: Text; _Value:
Text)
var
    Chr177: Text[1];
    NewPair: Text;
begin
    Chr177[1] := 177;
    NewPair := _Key + Chr177 + _Value + Chr177;
    KeyValueListAsText += NewPair;
end;

local procedure GetGlobalDataFields() KeyValueList : Text
begin
    AddKeyValue(KeyValueList, 'CompanyName', CompanyInfo.Name);
    AddKeyValue(KeyValueList, 'CompanyAddress', CompanyInfo.Address);
    AddKeyValue(KeyValueList, 'ReportDate', Format(Today));
end;
```

GetVal

Retrieves a value from global data by name or index.

Parameters:

- `Key` (String or Number): The key name (case-insensitive) or numeric index (1-based)

Returns:

- The value, or error message if not found (e.g., `?KeyName?`)

Usage in RDLC:

```
=Code.GetVal("CompanyName")'
=Code.GetVal("ReportDate")'
=Code.GetVal(1)'
```

Note: End expressions with an apostrophe (') to preserve arguments when copy/pasting textboxes.

AddKeyValue

Adds or updates a key-value pair in a collection.

Parameters:

- `Data` (Collection): The collection to modify
- `Key` (String): The key (case-insensitive)

- `Value` (Object): The value to store

Legacy NAV Way (SetData & GetData)

For backward compatibility with traditional NAV reports using numbered data groups.

SetData

Sets data in one of three global variables (Data1, Data2, or Data3).

Parameters:

- `NewData` (String): String with Chr(177) as separator
- `Group` (Integer): Which global variable to use (1, 2, or 3)

Usage:

```
=Code.SetData(Fields!GlobalData.Value, 1)
=Code.SetData(Fields!HeaderData.Value, 2)
```

GetData

Gets data by position number from one of three global variables.

Parameters:

- `Num` (Integer): Position number of the value (1-based)
- `Group` (Integer): Which global variable to use (1, 2, or 3)

Usage:

```
=Code.GetData(1, 1)  ' Gets first value from Data1
=Code.GetData(3, 2)  ' Gets third value from Data2
```

Note: The improved SetGlobalData/GetVal approach is recommended over SetData/GetData as it provides:

- Named keys instead of position numbers
- Better readability (`GetVal("CompanyName")` vs `GetData(5, 1)`)
- Single global collection instead of three separate variables
- Case-insensitive key access

String Manipulation Functions

These functions help concatenate strings with various separators, filtering out empty values.

ConcatenateNonEmptyWithCrLf

Concatenates non-empty strings from an array with CRLF (new line) characters.

```
Dim result As String = ConcatenateNonEmptyWithCrLf(New String() {"Hello", "",
"World"})
' Result: "Hello<CRLF>World"
```

ConcatenateNonEmptyWithCrLfAndDelimiter

Concatenates non-empty strings with the specified delimiter.

```
Dim result As String = ConcatenateNonEmptyWithCrLfAndDelimiter(New String()
{"Hello", "", "World"}, ",")
' Result: "Hello,World"
```

ConcatenateWithCrLf

Joins all strings with CRLF (new line) characters, including empty strings.

```
Dim result As String = ConcatenateWithCrLf(New String() {"Hello", "", "World"})
' Result: "Hello<CRLF><CRLF>World"
```

Number to Words Conversion

ToWordsIn (Double)

Converts a numeric value to its word representation in Indian format with optional currency formatting.

```
Dim result As String = ToWordsIn(1234.56, True, True)
' Result: "Rupees One Thousand Two Hundred Thirty-Four And Fifty-Six Paise
Only"
```

ToWordsIn (Long)

Converts a Long integer to its word representation using the Indian numbering system.

```
Dim result As String = ToWordsIn(1234567)
' Result: "Twelve Lakh Thirty-Four Thousand Five Hundred and Sixty-Seven"
```

FL_NumberToWordsMinimised

Creates a shorter representation of numbers using appropriate Indian units.

```
Dim result As String = FL_NumberToWordsMinimised(150000)
' Result: "1.5 Lakh"
```

Complete Usage Example

In C/AL or AL Code

```
local procedure AddKeyValue(VAR KeyValueListAsText: Text; _Key: Text; _Value:
Text)
var
    Chr177: Text[1];
    NewPair: Text;
begin
    Chr177[1] := 177;
    NewPair := _Key + Chr177 + _Value + Chr177;
    KeyValueListAsText += NewPair;
end;

local procedure GetGlobalDataFields() KeyValueList : Text
begin
    AddKeyValue(KeyValueList, 'CompanyName', CompanyInfo.Name);
    AddKeyValue(KeyValueList, 'Address', CompanyInfo.Address);
    AddKeyValue(KeyValueList, 'ReportDate', Format(Today));
end;
```

In RDLC Report

```
' Hidden tablix cell to set global data
=Code.SetGlobalData(Fields!GlobalData.Value)

' Header/Footer - Get values by name
=Code.GetVal("CompanyName")'
=Code.GetVal("Address")'
=Code.GetVal("ReportDate")'

' Logging
=Code.WriteLog("Report generated for: " & Fields!CustomerName.Value)

' String concatenation
=Code.ConcatenateNonEmptyWithCrLf(New String() {Fields!Line1.Value,
Fields!Line2.Value})

' Number to words
=Code.ToWordsIn(Fields!TotalAmount.Value)
```

Files

- **RdlcReportCode.vb** - Main code file (copy to RDLC Custom Code section)
- **RdlcReportCode_WithComments.vb** - Fully documented version with XML comments
- **RdlcVBCode_Usage** - Usage examples for all functions
- **Readme.md** - This documentation file

Credits

- Global Data Management functions from: frontlook-admin/RDLCReport_CustomCode
- Original concept by Andreas Rascher: AndreasRascher/RDLCReport_CustomCode