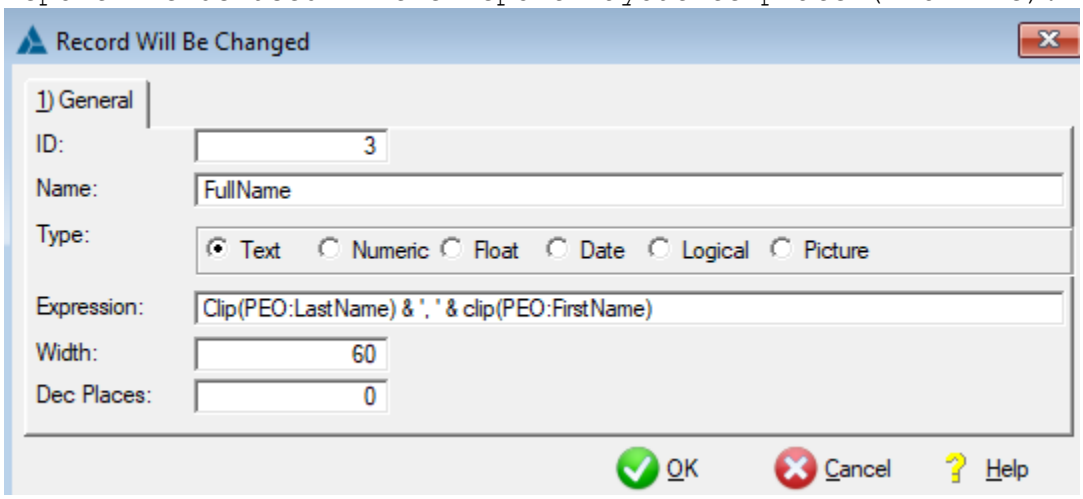
REPORTING IN CLARION USING REPORTEASE PLUS

ReportEase Plus consists of two components. The first component is the Report Designer. The Report Designer allows you to develop report layouts. The second component is the report executor. The report executor is used to print a report using a specified report template. Report templates are stored as individual files with the FPC extension. Report template can have multiple detail section, page headers and footers and report header and footer. Check product help for more information. Your Clarion application will need to store report fields information, so these fields can be populated and saved with the report template. During report execution your application will need to prepare and feed data before printing each detail section

INTEGRATING WITH REPORT DESIGNER

Your application needs to store and provide information about report fields used in the report layout template (FPC file).



This should include

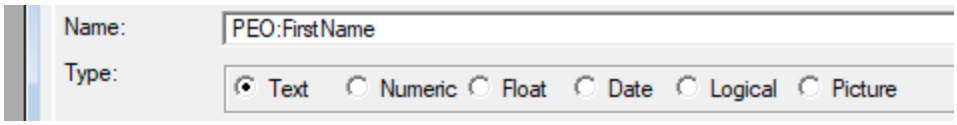
Unique Field ID – Can be numeric UID from a table where you would store report field data

Filed Type – Numeric value for ReportEase data types. Should be one of the following

```
TYPE_NUM              EQUATE(1)   !Numeric field no decimal, Clarion LONG/ULONG
TYPE_DBL              EQUATE(2)   !Double field equals to Clarion REAL
TYPE_TEXT             EQUATE(3)   !Text field
TYPE_LOGICAL          EQUATE(4)   !Logical field, Yes/No
TYPE_DATE             EQUATE(5)   !Date field
TYPE_PICT             EQUATE(6)   !Picture field
```

Field Name – Unique String value. Can (but not have to) be real name of a Clarion variable, field name or Clarion expression.

Expression (optional) – You can provide Clarion expression for a field which will be EVALUATEd during report execution.

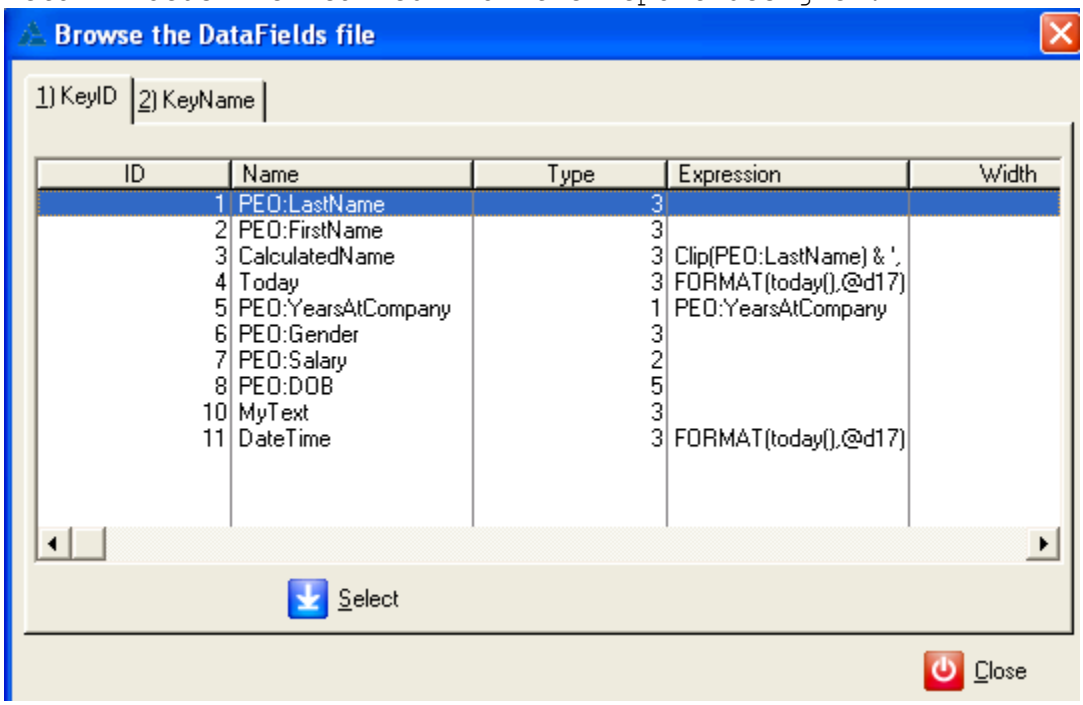


If valid Clarion expression is used as a field name or as an Expression parameter then it can be EVALUATEd during report execution and printed. Variables used in expressions need to be BINDed.

Field Width – number of characters for the TEXT fields

Decimal Places – Number of decimal places for the Double fields

You need to create your own field selection window procedure which can be standard Clarion browse with the request set to SelectRecord. Browse window should be set to modal so it would retain focus when called from the report designer.



This procedure will be called after selecting Insert-Data Field… option in designer menu. Call to the browse procedure needs to be wrapped into source procedure with the following prototype (UNSIGNED,LONG,SIGNED),SIGNED,PASCAL

Inside this procedure after successful selection of a record with report field information you would call SetReport wrapper method to pass field info to the designer. So, your code would look similar to:

```
PickDataField          FUNCTION (hWindow,lpStrField,ilFieldNo) !
Declare Procedure
RetVal                 LONG(FALSE)
RE                     ReportEase
FilesOpened       BYTE(0)
  CODE
  !This procedure passes our field info to be stored in FPC
report template file
```

```
  GlobalRequest = SelectRecord
  !Browse with our field data. You can create your own one
  BrowseFieldsModal()
  If GlobalResponse = RequestCompleted
    !Call method below to populate field data
    Retval =
RE.SetReportField(lpStrField,Dat:Type,Dat:ID,Dat:Name,Dat:Express
ion,Dat:Width,Dat:DecPlaces)
  END
  RETURN RetVal
```

Before calling report designer this procedure needs to be registered with the **RegisterPickFieldProcedure** method:

```
RE.RegisterPickFieldProcedure(Address(PickDataField))
```

You also need to register field validation procedure. It will be called by the Report Designer whenever it needs to verify a data field name as entered by the user. This procedure should have the following prototype
(UNSIGNED,LONG,LONG),SIGNED,PASCAL
And code should be similar to the one below

```
ValidateDataField    FUNCTION (hWindow, FieldPtr, SortFieldNo) !
Declare Procedure
bRetVal          SIGNED                          ! Return value
RE               ReportEase
FilesOpened     BYTE(0)
  CODE
  ! Assume field invalid
  bRetVal = FALSE
  ! Validate field name and check data type
  CLEAR(Dat:Record)
  Dat:Name = RE.GetFieldNameFromStructure(FieldPtr)
  Get(DataFields,Dat:KeyName)
  IF NOT ErrorCode() AND INRANGE(Dat:Type,1,6)
    bRetVal = TRUE
  END
  RETURN bRetVal
```

Before calling report designer this procedure needs to be registered with the **RegisterValidateFieldProcedure** method:

```
RE.RegistervalidateFieldProcedure(Address(ValidateDataField))
```

This is sequence of calls to edit report layout stored in FPCName variable

```
  RE.RegisterPickFieldProcedure(Address(PickDataField))
  RE.RegistervalidateFieldProcedure(Address(ValidateDataField))
  RE.OpenDesigner(FPCName)
```

INTEGRATING WITH REPORT EXECUTOR

Report Executor is used when report is printed, previewed or saved to disk file. Your application provides the data records. The report executor applies the data records to the chosen report layout template to produce the output. The following methods have been provided to interface with the report executor.

InitReport PROCEDURE(<STRING pOutFile>),LONG,VIRTUAL,PROC

Your application calls this method to initialize the report executor. Before calling this method, you need to set ReportTemplate and ReportOutput properties. ReportOutput property supports the following values:

'P' = Printer
'S' = Screen
'F' = Native format File
'R' = RTF file (.RTF extension).
'H' = HTML file.
'T' = Ascii text file.
'C' = Comma delimited export file.
'B' = Tab delimited export file
'A' = Ask User.
'D' = PDF file.

For example, to print report to default printer using PEOPLE.FPC template file your code would look like

```
RE.ReportTemplate = 'PEOPLE.FPC'
RE.ReportOutput = 'P'
RE.InitReport()
```

Optional pOutFile parameter is used when report is saved to file. For example, to save report to pdf file you code would look like:

```
RE.ReportTemplate = 'PEOPLE.FPC'
RE.ReportOutput = 'D'
RE.InitReport('mysavedreport.pdf')
```

This method returns a unique report id

PrintRecord PROCEDURE(),**LONG**,VIRTUAL,PROC

This method is called for each record in your data set. For example, it can be called in a process as "Activity for Each Record". Printed values are defined by field Name or Expression which are set during report design process. You can also call **SetFieldValue** method to set values for particular field(s) before printing.

EndReport PROCEDURE()

This method ends the report. Your application calls this method to print the ending totals and free up the resources.

AVAILABLE PROPERTIES

OutFile STRING(300) – File name if report output is set to disk file. Can be overridden with the pOutFile parameter in the ReportInit method.
ReportTemplate STRING(300) – File name of the report layout template file.
ReportOutput STRING(1) – Type of the report output. Must be one of the following
'P' = Printer
'S' = Screen
'F' = Native format File
'R' = RTF file (.RTF extension).
'H' = HTML file.
'T' = Ascii text file.
'C' = Comma delimited export file.
'B' = Tab delimited export file
'A' = Ask User.
'D' = PDF file.

These properties need to be set before calling InitReport method.

AVAILABLE METHODS

InitReport PROCEDURE(<STRING pOutFile>),LONG,VIRTUAL,PROC

Initialize the report executor.
pOutFile – File name when output is set to disk file

PrintRecord PROCEDURE(),LONG,VIRTUAL,PROC

Prints current data record

EndReport PROCEDURE()

Ends the report printing.

SetFieldValue PROCEDURE(STRING pFieldName, STRING pValue)

Sets printable field value before printing.
pFieldName – field Name
pValue – Value to set

OpenDesigner PROCEDURE(<**STRING** pFPCName>)**,LONG**

Opens report designer

pFPCName - optional file name of the report layout template

SetReportField PROCEDURE(**LONG** lpStrField, **LONG** FieldType, **STRING** FieldID, **STRING** FieldName, <**STRING** Expression>, |
LONG FieldWidth = **20**, **LONG** DecPlaces=**0**),**LONG**

You would call this method inside the source data field selection procedure after successful field selection.
lpStrField - pointer to the ReportEase StrField structure.
Other parameters are described in the INTEGRATING WITH REPORT DESIGNER chapter.

Adding ReportEase class to Application

GLOBAL EMBEDS

In Global Embeds, After Global INCLUDEs, add the following line:
INCLUDE('ReportEase.INC'),ONCE
At Before Global Data embed put
INCLUDE('ReportEase.EQU'),ONCE

Project Settings
For DLL version of the class set
CLAREPEASEDllMode=>1
CLAREPEASELinkMode=>0
project defines.
Add CLAREPEASE.lib as an external lib file.

Class Declaration

RE ReportEase