A ledger-sheet maker for .NET Framework

Reports.NET

Programmer’s Manual

7th edition

July 27, 2011

Pao@Office

Pao@Office Ltd.
3-29-2-401 yatsu, Narahino, Chiba
Zip Code 275-0026

http://www.pao.ac/en/

Table of Contents

Introduction 1
Functions 3
Operating Condition 4
How to Use 6
How to Use Reports.NET from Apprication Program 7
  Sample Program as an Example 7
  Example for C# 9
  Example for VB.NET 10
  How to Create instance of the object of print or preview 11
  Read Report Definition File 12
  Declare Start and End of a Page 13
  Data Set for an Object (VB.NET) 17
  Order for Print and Preview 20
  Save and Read Print Data 21
  Obtain Compressed Print Binary Data 22
Programmer's Reference 23
  ReportCreator Class 24
  IObjects Interface / z_Objects Object 25
  LoadDefFile Method 26
  PageStart Method 27
  Write Method 29
  void Write(string name, string value) Method 30
  void Write(string name, string value, int index) Method 31
  void Write(string name, int index) Method 32
  Sub Write(name As String, value As String) Method 33
  Sub Write(name As String, value As String, index As Long) Method 34
  Sub Write(name As String, index As Long) Method 35
  Output Method 36
  Output() Method 37
  Output(System.Drawing.Printing.PrinterSettings setting) Method 38
  SaveXMLFile Method 39
  LoadXMLFile Method 40
  SaveData Method 41
  LoadData Method 42
  SaveSVGFile Method 43

SaveSVGZFile Method 44
SavePDF Method (Stream) 45
SavePDF Method (File) 46
DisplayDialog Property 47
DisplayPrinting Property 48
PreviewDialog Property 49
AccessFile Property 50
MarginTop Property 51
MarginLeft Property 52
z_Objects Property / IObjects Interface 53
SetObject (string　objName) Method 54
z_Text Property / ZText Class 55
z_Line Property / ZLine Class 56
z_Square Property / ZSquare Class 57
z_Circle Property / ZCircle Classs 58
z_Image Property / ZImage Class 59
z_Barcode Property / ZBarcode Class 60
z_ArtText Property / ZArtText Class 61
z_FontAttr Property / ZFontAttr Class 63
z_LineAttr Property / ZLineAttr Class 64
GetPreview Method 65
GetReport Method 66
GetPdf Method 67
GetImagePdf Method 68
Modification History 69

Introduction

Hello all programmers who create program under the .NET developing environment.

The interface of Reports.NET as a class is so simple and easy that it requires little effort.

*It means that a report definition XML file as a design part plays a major role.

There are a few classes and methods such as:

```
IReports Interface・・・Common interface for print or preview
 │
 ├── LoadDefFile Method     ---Read a report definition file
 ├── PageStart Method       ---Declare a start of a page
 ├── Write Method           ---Write print data
 ├── PageEnd Method        ---Declare an end of a page
 ├── Output Method          ---Order print / preview
 ├── LoadXMLFile Method --- Read print data file
 ├── SaveXMLFile Method --- Write print data file
 ├── SaveData Method         ---Return compressed print binary data ┐  ┌──────────────────┐
 │                                                                  ├─ │Transfer format with│
 ├── LoadData Method         ---Read compressed print binary data  ┘  │Web service         │
 │                                                                     └──────────────────┘
 ├── SaveSVGFile Method     ---Write print data in SVG format
 ├── SaveSVGZFile Method --- Write print data in SVGZ format
 └── SavePDF Method          --- Write print data in PDF format

ReportCreator Class       --- Return instance (object) of printing or previewing
 │       (above IReports form)
 ├── GetPreview Method      ---Return preview object
 ├── GetReport Method       ---Return print object
 ├── GetPdf Method          ---Return PDF object
 └── GetImagePdf Method      ---Return image PDF object
```

That's it.　That's all you need.

*You will also see ReportStartImpl class but please don't worry about it.　It is only for preview reboot.

What do you think?　It looks like nothing to worry about.

Now let's move on to the details of each class and method with some examples such as coding.

I sincerely hope that all programmers will enjoy programming easily with this software.

Creator

As of February 24, 2011, following properties have been added to interfaces introduced in the previous page.
These were all implemented in order to meet cusomers’ needs.

IReports Interface ---Common interface for print or preview

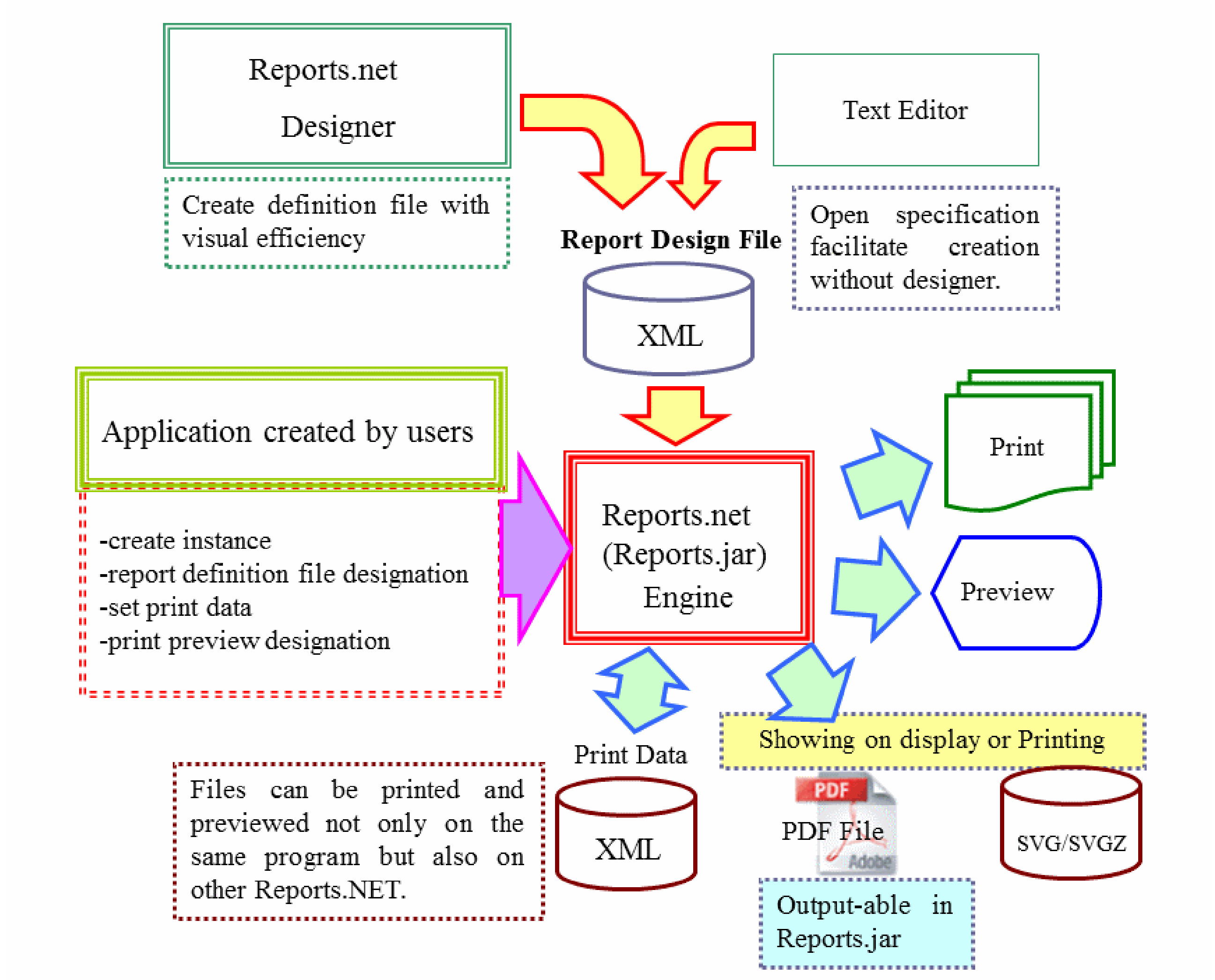
- bool DisplayDialog ---Display or hide print dialog
- bool DisplayPrinting --- Display or hide printing (the number of page)
- bool PreviewDialog --- Display or hide preview dialog
- bool AccessFile ---Allow or disallow file access (saving in file)
- float MarginTop ---Set head margin by millimeter (effective only in printing or previewing)
- float MarginLeft --- Set left margin by millimeter (effective only in printing or previewing)
- IObjects z_Objects ---Obtain an attribute of an object in designing. Static class for setting

Functions

[Monolithic function]

The core product of Reports.net and Reports.jar is Engine. The Engine offers functions to create ledger sheets defined by the report definition file for .Net application. Users can manage the Engine with arbitrary applications then preview and print ledger sheets and write print data.

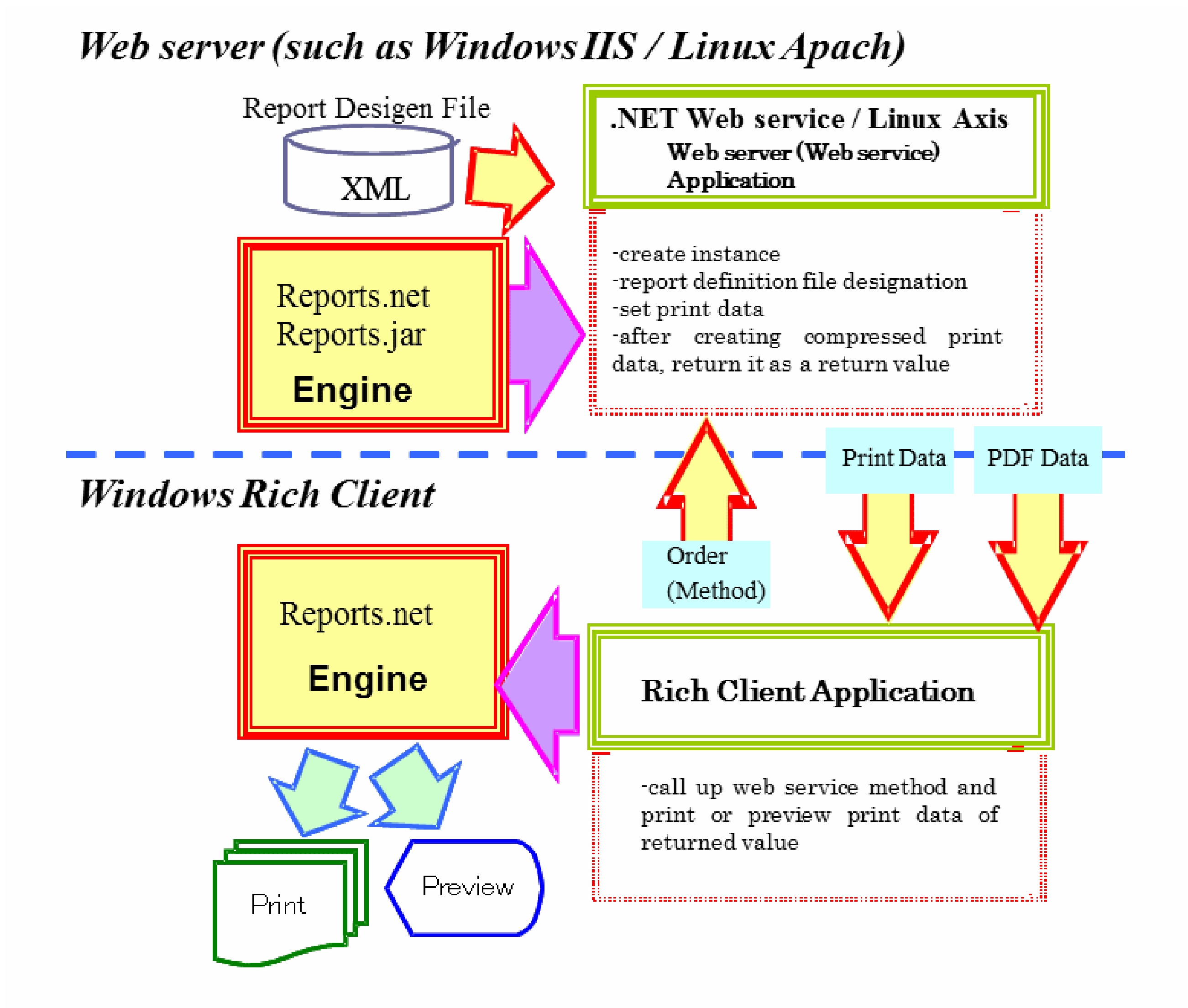
The data can be outputted in PDF or SVG/SVGZ and also be previewed and printed out by Web browser.


Reports.net Designer
Create definition file with visual efficiency
Text Editor
Open specification facilitate creation without designer.
Report Design File
XML
Application created by users
-create instance
-report definition file designation
-set print data
-print preview designation
Reports.net (Reports.jar) Engine
Print
Preview
Showing on display or Printing
Print Data
XML
Files can be printed and previewed not only on the same program but also on other Reports.NET.
PDF File
SVG/SVGZ
Output-able in Reports.jar

[Linkage with web services]

Binary data for printing can be obtained and printed by just one order or one call from a client based on a Windows platform to a web service such as .NET Web service or Axis on IIS or UNIX servers including Linux. After receiving the order from the client, the server accesses the database by itself to create print data and returns it to the client as binary data; byte [] type variable. The client then prints it out.

Developed languages include not only .NET but Java thus allowing the web server platform to grow in diversity. **(Reports.jar)**


Web server (such as Windows IIS / Linux Apach)
Report Desigen File
XML
.NET Web service / Linux Axis
Web server (Web service)
Application
-create instance
-report definition file designation
-set print data
-after creating compressed print data, return it as a return value
Reports.net
Reports.jar
Engine
Print Data
PDF Data
Windows Rich Client
Order
(Method)
Reports.net
Engine
Rich Client Application
-call up web service method and print or preview print data of returned value
Print
Preview

Operating Condition

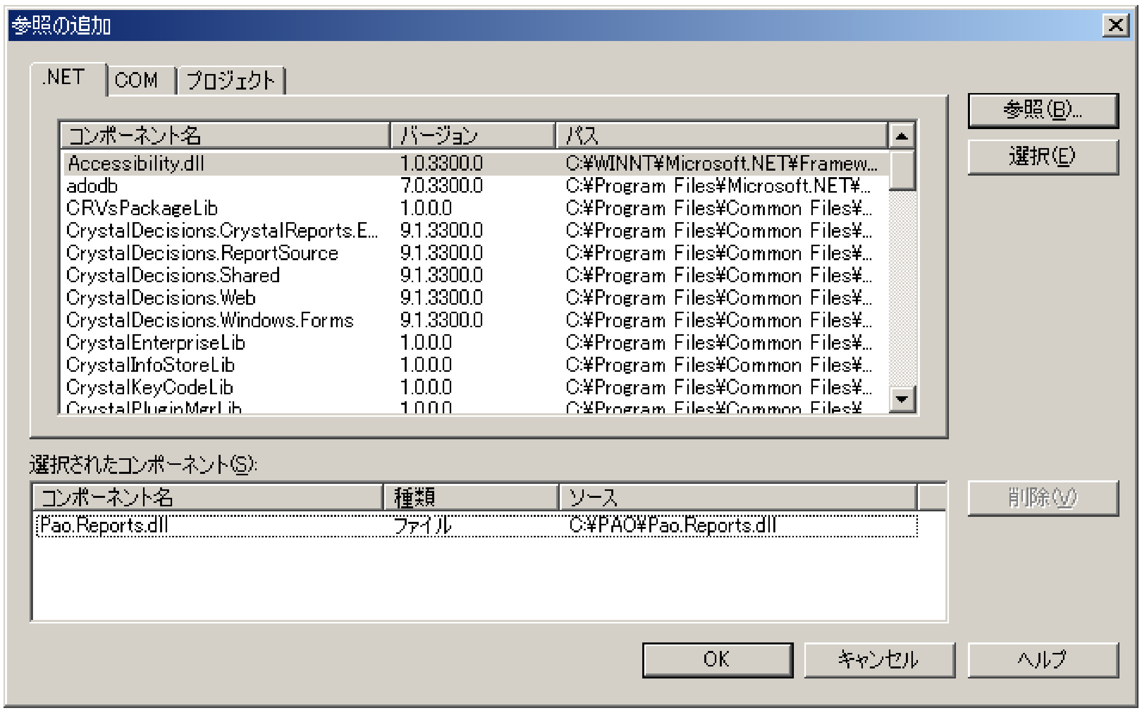
In order to use this software, a computer which meets the following requirements is needed.

OS	Systems that operate Microsoft.NET Framework sufficiently.
Computer Memory	Equivalent to the memory allocation which allows Microsoft .NET Framework to operate sufficiently.
Recommended Screen Resolution	Engine：no special limit Designer: resolution should be over 1024×768 and font size is standard small font.
Developing Environment	Microsoft Visual Studio .NET should be installed.

How to Use

1. Copy Pao.Reports.dll to arbitrary directory. The latest edition is offered on our Web site at all times.
http://www.pao.ac/en/reports.net/
(User Information File will be sent to the customer who officially regiser as a user. By copying the User Information File to the same place as Pao.Reports.dll, the software will operate as an official vertion.)

2. Add Pao.Reports.dll to the reference of the project in which you would like to use Reports.NET.



コンポーネント名	バージョン	パス
Accessibility.dll	1.0.3300.0	C:¥WINNT¥Microsoft.NET¥Framew...
adodb	7.0.3300.0	C:¥Program Files¥Microsoft.NET¥...
CRVsPackageLib	1.0.0.0	C:¥Program Files¥Common Files¥...
CrystalDecisions.CrystalReports.E...	9.1.3300.0	C:¥Program Files¥Common Files¥...
CrystalDecisions.ReportSource	9.1.3300.0	C:¥Program Files¥Common Files¥...
CrystalDecisions.Shared	9.1.3300.0	C:¥Program Files¥Common Files¥...
CrystalDecisions.Web	9.1.3300.0	C:¥Program Files¥Common Files¥...
CrystalDecisions.Windows.Forms	9.1.3300.0	C:¥Program Files¥Common Files¥...
CrystalEnterpriseLib	1.0.0.0	C:¥Program Files¥Common Files¥...
CrystalInfoStoreLib	1.0.0.0	C:¥Program Files¥Common Files¥...
CrystalKeyCodeLib	1.0.0.0	C:¥Program Files¥Common Files¥...
CrystalPluginMgrLib	1.0.0.0	C:¥Program Files¥Common Files¥

コンポーネント名	種類	ソース
Pao.Reports.dll	ファイル	C:¥PAO¥Pao.Reports.dll

3. Define "using" for C# or "Imports" for VB.NET if needed.

For C#
using Pao.Reports;

For VB.NET
Imports Pao.Reports

How to Use Reports.NET from Apprication Program

Sample Program as an Example

In all this section, How to Use Reports.NET from Apprication Program, explanations will be made with sample programs.　Please keep the following in mind.

<About program>

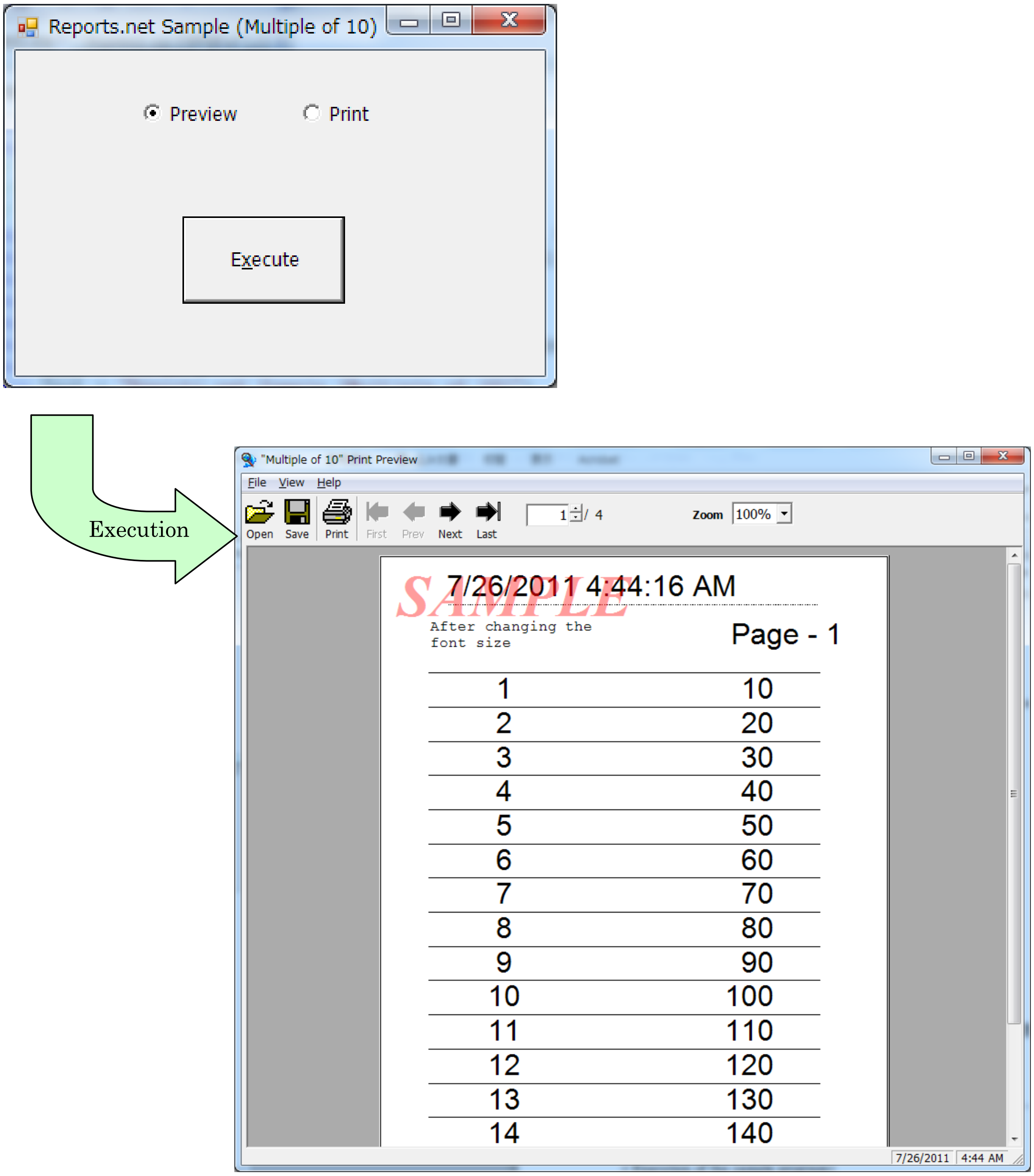
- Choose print or preview by checking radio button (option button) on a screen and click Execute, and then program starts.
- Write time and date and the number of pages to the header of each page of ledger sheet.
- In the detail part, the numbers of lines looped 60 times and the decuple values of the each number will be written in a chart.
- Each line of the detail part will be separated by horizontal ruled lines.
- A page will be broken with 15 lines so there will be 4 pages all together.
- After drawing stated above, take the next step of printing or previewing following the directions on the screen.
- Finaly, save the print data which was previously printed or previewed to a print data file, reread the file again and preview the print data.

The sample program which performs the above process is made and offered with C#.NET/VB.NET for reference.　The sample also has some comment and they would be helpful.

Please keep the processing flow of the sample program in mind.

This sample program is found in the folder, sample¥programers, in a compressed file of the product.

< Execution of the sample program>



SAMPLE

7/26/2011 4:44:16 AM

After changing the font size

Page - 1

1	10
2	20
3	30
4	40
5	50
6	60
7	70
8	80
9	90
10	100
11	110
12	120
13	130
14	140

Example for C#

```
// IReport Declaration with interface (Prepare a holder which can be used for both printing and
previewing)
        IReport paoRep = null;

        if(radioButtonPreview.Checked)// If preview is chosen in radio button
        {
                // Create the instance of preview object
                paoRep = ReportCreator.GetPreview();
        }
        else
        {
                // Create the instance of print object
                paoRep = ReportCreator.GetReport();
        }

        // Read(Load) report definition file
        paoRep.LoadDefFile("report definition file.xml");

        int page = 0; // Define the number of pages
        int line = 0; // Define the number of lines

        for (int i = 0; i < 60; i++)
        {
                if (i % 15 == 0) // Start a page with 15 lines
                {
                        // Declare a star of a page
                        paoRep.PageStart();
                        page++;             // Increment the number of pages
                        line = 0;   // Reset the number of lines

                        // *** Header setting***
                        // Set character strings
                        paoRep.Write("time and date", System.DateTime.Now.ToString());
                        paoRep.Write("the number of pages", "Page - " + page.ToString());
                }
                line++; // Increment the number of lines

                // ***Detail setting***
                // Set repeated character strings
                paoRep.Write("line number", (i+1).ToString() , line);
                paoRep.Write("decuple number", ((i+1)*10).ToString() , line);
                // Set repeated figure (horizontal line)
                paoRep.Write("horizontal line", line);

                if (((i+1) % 15) == 0) paoRep.PageEnd(); // Declare an end of the page with 15
lines
        }

        // Execute print or preview
        paoRep.Output();

        paoRep.SaveXMLFile("print data.XML"); // Save the print data

        // reobtain the instance of preview object (reset once)
        paoRep = ReportCreator.GetPreview();

        paoRep.LoadXMLFile("print data.XML"); // Read(Reload) the print data

        paoRep.Output(); // Execute preview
```

Example for VB.NET

```
 'IReport Declaration with interface (Prepare a holder which can be used for both printing and
previewing)
   Dim paoRep As IReport = Nothing

   If radioButtonPreview.Checked = True Then 'If preview is chosen in radio button
         'Create the instance of preview object
         paoRep = ReportCreator.GetPreview()
   Else
         'Create the instance of print object
         paoRep = ReportCreator.GetReport()
   End If

   'Read(Load) report definition file
   paoRep.LoadDefFile("report definition file.xml")

   Dim page As Integer = 0 'Define the number of pages
   Dim line As Integer = 0 'Define the number of lines
   Dim i As Integer
   For i = 1 To 60
         If ((i - 1) Mod 15 = 0) Then 'Start a page with 15 lines
               'Declare a star of a page
               paoRep.PageStart()
               page = page + 1    'Increment the number of pages
               line = 0 'Reset the number of lines

               '***Header setting***
               'Set character strings
               paoRep.Write("time and date ", System.DateTime.Now.ToString())
               paoRep.Write("the number of pages ", "Page - " + page.ToString())

         End If
         line = line + 1 'Increment the number of lines

         '***Detail setting***
         'Set repeated character strings
         paoRep.Write("line number ", i.ToString(), line)
         paoRep.Write("decuple number ", (i * 10).ToString(), line)
         'Set repeated figure (horizontal line)
         paoRep.Write("horizontal line", line)

         If ((i Mod 15) = 0) Then paoRep.PageEnd() 'End of the page with 15 lines
   Next i

   'Execute print or preview
   paoRep.Output()

   paoRep.SaveXMLFile("print data file.xml") ' Save the print data

   'Reobtain the instance of preview object (reset once)
   paoRep = ReportCreator.GetPreview()

   paoRep.LoadXMLFile("print data file.xml") ' Read(Reload) the print data

   paoRep.Output() 'Execute preview
```

How to Create instance of the object of print or preview

Because Reports.net's class of print and preview have same methods,

After declaring an object using the IReport interface,

By calling one of the following static methods of ReportCreator class, you can create an instance of the print or preview object.

- IReport GetPreview () -> Create instance of Preview
- IReport GetReport () -> Create instance of Print

＜Example for C#＞

```
// IReport Declaration with interface
// (Prepare a holder which can be used for both printing and previewing)
IReport paoRep = null;

if(radioButtonPreview.Checked)// If preview is chosen in radio button
{
        // Get the instance of preview object
        paoRep = ReportCreator.GetPreview();
}
else
{
        // Get the instance of print object
        paoRep = ReportCreator.GetReport();
}
```

＜Example for VB.NET＞

```
'IReport Declaration with interface
'(Prepare a holder which can be used for both printing and previewing)
Dim paoRep As IReport = Nothing

If radioButtonPreview.Checked = True Then' If preview is chosen in radio button
    'Get the instance of preview object
    paoRep = ReportCreator.GetPreview()
Else
    'Get the instance of print object
    paoRep = ReportCreator.GetReport()
End If
```

Of course, if you want to only preview, It is possible to do the following.

```
--
IReport paoRep = ReportCreator.GetPreview();
Dim paoRep As IReport = ReportCreator.GetPreview()
--
```

Read Report Definition File

When you set data to a ledger sheet from a program, the first step is to read a report definition file such as the one created with Designer.
*Definisions for a ledger sheet, including which coordinate has which object, are written in XML file format in report definition file. For details, please refer to the Report Definition XML File Specification Document.

Use the LoadDefFile method to read a report definition file from a program implemented in IReport interface. Set the path of the report definition file which is to be read to the first argument of LoadDefFile method.

In the example, the relative path is seen, however we recommend the absolute path because it is constrained to where the program would run.

＜Example for C#.NET＞

```
//Read a report definition file
paoRep.LoadDefFile("report definition file.xml");
```

＜Example for VB.NET＞

```
'Read a report definition file
paoRep.LoadDefFile("report definition file.xml")
```

Declare Start and End of a Page

When you set date to a ledger sheet from a program, a report definition file should be read and declaration of start and end at each page should be made.
Set the ledger sheet data between the declaration of the start and the end. There is no need for the data setting when outputting the ledger sheet as showed in the report definition file created by Designer.

In other words, the minimum steps of outputting a ledger sheet by reading a report definition from a program are:
1. Create a print or a preview instance.
2. Read a report definition file.
3. Declare a start of a page.
4. Declare an end of the page.
5. Set the printing or previewing

In most cases, the logic is inserted which set a ledger sheet data between "3. Declare a start of a page" and "4. Declare an end of the page".

To declare start and end of a page, use PageStart/PageEnd method implemented in IReport interface.
There is no argument.

<Example for C#.NET>

```
//Declare a star of a page
paoRep.PageStart();
        ⋮
//Declare an end of the page
paoRep.PageEnd();
```

Write() --- process for print data set

<Example for VB.NET>

```
'Declare a star of a page
paoRep.PageStart()
        ⋮
'Declare an end of the page
paoRep.PageEnd()
```

Write() --- process for print data

Data Set for an Object (C#.NET)

This section explains, with C#.NET, how to put a value to each object designated by report definition file and how to draw horizontal ruled lines in a chart repeatedly. Data set for an object should be created between the start and the end declarations of the page (between PageStart and PageEnd).

When data is set from a program to a ledger sheet, use the Write method implemented in IReport interface.　The Write method implements three patterns.

(1) void Write(string name, string value)

This sets a character string to an object.

Use this to set a value of unrepeated persistent objects such as header and footer.

string name

This specifies a name of an object in report definition file.

The intentded（intended） object types are Text (character string), ArtText (decorated character string) and Barcode (barcode) because this sets character string.

In deleting objects, specify object other than Text (character string) or ArtText (decorated character string)

string value

This specifies a character string to set.

If an object other than Text (character string) or ArtText (decorated character string) is set with a blank (""), the object will be deleted.

(2) void Write(string name, string value, int index)

This specifies a drawing position for an object and set a character string.

Use this for objects to set repeated values such as lines in a chart.

When using the method for this pattern, IntervalX or IntervalY in the report definition file should be entered with a value more than 1.

IntervalX means intervals repeated in a transverse direction by milimeter.

IntervalY means intervals repeated in a longitudinal direction by millimeter. This is mainly used for lines of a chart.

string name

This specifies a name of an object in report definition file.

The intended object types are basically Text (character string), ArtText (decorated character string) and Barcode (barcode) because this sets character string.

In deleting objects, specify object other than Text (character string) or ArtText (decorated character string)

string value

This specifies a character string to set.

If an object other than Text (character string) or ArtText (decorated character string) is set with a blank (""), the object will be deleted.

int index

This refers to the drawing position in the page set in a transverse and a longitudinal direction intervals by IntervalX／IntervalY. The values get bigger from upper left to lower right.

For example, a chart with a value in IntervalY has a drawing position such as:

(the first position of an object) + InterbalY × (index –1).

As for a chart, 1 is in the first line, 2 in the second and 3 in the third.

(3) void Write(string name, int index)

This specifies a drawing position for an object. Use this for objects to set repeated values such as horizontal ruled lines in a chart.

When using the method for this pattern, IntervalX or IntervalY in report definition file should be put with a value more than 1.

IntervalX means intervals repeated in a transverse direction by milimeter.

IntervalY means intervals repeated in a longitudinal direction by millimeter. This is mainly used for lines of a chart.

string name

This specifies a name of an object in report definition file.

All objects can be applied because any objects can be drawn repeatedly.

int index

This means printing position in the page set in a transverse and a longitudinal direction intervals by IntervalX／IntervalY. The values get bigger from upper left to lower right.

For example, a chart with a value in IntervalY has a painting position such as:

(the first position of an object) + InterbalY × (index –1).

As for a chart, 1 is in the first line, 2 in the second and 3 in the third.

＜Expample for C#.NET＞

```
int page = 0; //Define the number of pages
int line = 0; // Define the number of lines
for (int i = 0; i < 60; i++)
{
            if (i % 15 == 0) //Start a page with 15 lines
            {
                        //Declare a star of a page
                        paoRep.PageStart();
                        page++;                 //Increment the number of pages
                        line = 0;  //Reset the number of lines
                        //＊＊＊Header setting＊＊＊
                        //Set character strings
                        paoRep.Write("time and date", System.DateTime.Now.ToString());
                        paoRep.Write("the number of pages", "Page - " + page.ToString());
            }
            line++; //Increment the number of lines

            //＊＊＊Detail setting＊＊＊
            //Set repeated character strings
            paoRep.Write("line number", (i+1).ToString() , line);
            paoRep.Write("decuple number", ((i+1)*10).ToString() , line);
            //Set repeated figure (horizontal line)
            paoRep.Write("horizontal line", line);

            if (((i+1) % 15) == 0) paoRep.PageEnd(); //Declare an end of the page with 15 lines
}
```

Data Set for an Object (VB.NET)

This section explains, with VB.NET, how to put a value to each object designated by report definition file and how to draw horizontal ruled lines in a chart repeatedly.
Data set for an object should be created between the start and the end declarations of the page (between PageStart and PageEnd).

When data is set from a program to a ledger sheet, use Write() method implemented in IReport interface.　The Write() method is overloaded for three patterns.

(1) Sub Write(name As String, value As String)
  This sets a character string to an object.
  Use this to set a value of unrepeated persistent objects such as header and footer.

  name As String
    This specifies a name of an object in report definition file.
    The intentded （intended） object types are Text (character string), ArtText (decorated character string) and Barcode (barcode) because this sets character string.
    In deleting objects, specify object other than Text (character string) or ArtText(decorated character string)

  value As String
    This specifies a character string to set.
    If an object other than Text (character string) or ArtText (decorated character string) is set with a blank (“”), the object will be deleted.

(2) Sub Write(name As String, value As String, index As Long)

This specifies a drawing position for an object and set a character string.
Use this for objects to set repeated values such as lines in a chart.
When using the method for this pattern, IntervalX or IntervalY in report definition file should be put with a value more than 1.
IntervalX means intervals repeated in a transverse direction by milimeter.
IntervalY means intervals repeated in a longitudinal direction by millimeter. This is mainly used for lines of a chart.

name As String

This specifies a name of an object in report definition file.
The intended object types are only Text (character string), ArtText (decorated character string) and Barcode (barcode) because this sets character string.
In deleting objects, specify object other than Text (character string) or ArtText (decorated character string)

value As String

This specifies a character string to set.
If an object other than Text (character string) or ArtText (decorated character string) is set with a blank (“”), the object will be deleted.

index As Long

This means drawing position in the page set in a transverse and a longitudinal direction intervals by IntervalX／IntervalY. The values get bigger from upper left to lower right.
For example, a chart with a value in IntervalY has a drawing position such as:
(the first position of an object) + InterbalY × (index –1).
As for a chart, 1 is in the first line, 2 in the second and 3 in the third.

(3) Sub Write(name As String, index As Long)

This specifies a drawing position for an object

Use this for objects to set repeated values such as horizontal ruled lines in a chart.

When using the method for this pattern, IntervalX or IntervalY in report definition file should be put with a value more than 1.

IntervalX means intervals repeated in a transverse direction by milimeter.

IntervalY means intervals repeated in a longitudinal direction by millimeter.　This is mainly used for lines of a chart.

name As String

This specifies a name of an object in report definition file.

All objects can be applied because any objects can be drawn repeatedly.

index As Long

This means printing position in the page set in a transverse and a longitudinal direction intervals by IntervalX／IntervalY.　The values get bigger from upper left to lower right.

For example, a chart with a value in IntervalY has a printing position such as: (the first position of an object) + InterbalY × (index –1).

As for a chart, 1 is in the first line, 2 in the second and 3 in the third.

<Example for VB.NET>

```
Dim page As Integer = 0 'Define the number of pages
Dim line As Integer = 0 ' Define the number of lines
For i = 1 To 60
    If ((i - 1) Mod 15 = 0) Then 'Start a page with 15 lines
        'Declare a star of a page
        paoRep.PageStart()
        page = page + 1    'Increment the number of pages
        line = 0 'Reset the number of lines

        '＊＊＊Header setting＊＊＊
        'Set character strings
        paoRep.Write("time and date ", System.DateTime.Now.ToString())
        paoRep.Write("the number of pages ", "Page - " + page.ToString())
    End If
    line = line + 1 'Increment the number of lines

    '＊＊＊Detail setting＊＊＊
    'Set repeated character strings
    paoRep.Write("line number ", i.ToString(), line)
    paoRep.Write("decuple number ", (i * 10).ToString(), line)
    'Set repeated figure (horizontal line)
    paoRep.Write("horizontal line ", line)

    If ((i Mod 15) = 0) Then paoRep.PageEnd() ' Declare an end of the page with 15 lines
Next i
```

Order for Print and Preview

After the data set to objects of each ledger sheet and the end declaration of the the last page (PageEnd) is made, then print and preview can be made.
In order to print or preview from a program, use Output method implemented in IReport interface.　There is no argument.

＜Example for C#.NET＞

```
paoRep.Output(); //Execute print or preview
```

＜Example for VB.NET＞

```
paoRep.Output() 'Execute print or preview
```

Save and Read Print Data

Reports.NET can save print data directly to XML file and read it. For example, in communication between Web application and a client, it is possible to create a ledger sheet searching database in a server and to reserve it at the client.
The timing of saving print data is the same as the order for print or preview after the data set of the ledger sheet. Saving can be done before and after print or preview.
Reading can be chosen anytime as long as print or preview instance has been created. For example, the print data read under the following procedure can be printed or previewed.
1. Create instance for print and preview
2. Read print data file
3. Order for print and preview

In order to save print data from a program, use SaveXMLFile method implemented in IReport interface. There is no argument.
In order to read print data from a program, use LoadXMLFile method implemented in IReport interface. There is no argument.

<Example for C#.NET>

```
paoRep.SaveXMLFile("print data.XML"); //Save the print data

//reobtain the instance of preview object (reset once)
paoRep = ReportCreator.GetPreview();

paoRep.LoadXMLFile("print data.XML"); //Read the print data

paoRep.Output(); //Execute preview
```

<Example for VB.NET>

```
  paoRep.SaveXMLFile("print data file.xml") 'Save the print data

  'reobtain the instance of preview object (reset once)
  paoRep = ReportCreator.GetPreview()

  paoRep.LoadXMLFile("print data file.xml") 'Read the print data

  paoRep.Output() 'Execute preview
```

Obtain Compressed Print Binary Data

For Web

In order to obtain a compressed print binary data from a program, use SaveData method implemented in IReport interface.　There is no argument.
In order to read a compressed print binary data from a program, use LoadData method implemented in IReport interface.　The argument is the file name of print data (ZIP format).

Save SVG, SVGZ and PDF Print Data

In order to write SVG format print data from a program, use SaveSVGFile method implemented in IReport interface.　The argument is the file name used for saving SVG format data.　(The extension is html.)
In order to write SVGZ format print data from a program, use SaveSVGZFile method implemented in IReport interface.　The argument is the file name used for saving SVGZ format data.　(The extension is html.)
In order to write PDF format print data from a program, use SavePDF method implemented in IReport interface.　The argument is the file name used for saving print data or stream (System.IO.Stream).

Programmer's Reference

IReport Interface

IReport Interface is the interface which has all methods controlling Reports.NET.
It is possible to create instance using GetPreview method or GetReport method in ReportCreator class. Create instance with GetPreview for previewing and with GetReport for printing.

Constructor
There is no argument.

Public Method

LoadDefFile	Read a report definition file
PageStart	Declare a start of a page
PageEnd	Declare an end of a page
Write	Write print data
Output	Order print or preview
SaveXMLFile	Write a print data file
LoadXMLFile	Read a print data file
SaveData	Return compressed print binary data
LoadData	Read compressed print binary data
SaveSVGFile	Write print data in SVG format
SaveSVGZFile	Write print data in SVGZ format
SavePDF	Write print data in PDF format

Public Property

DisplayDialog	Display or hide [Print] dialog in printing (Output).
DisplayDialog	Display printing (the number of page).
PreviewDialog	Display or hide preview dialog.
AccessFile	Permit file access from preview window.
MarginTop	Set head margin by millimeter (effective only in printing or previewing)
MarginLeft	Set left margin by millimeter (effective only in printing or previewing)
z_Objects	Obtain an attribute of an object in designing. Static class for setting

ReportCreator Class

ReportCreator class is implemented with a method which returns object to print or preview.

This contains IReport type of GetPreview or GetReport method.

Call GetPreview method for previewing, and GetReport method for printing.

Public Method

GetPreview	Retrutn a preview object.
GetReport	Retrutn a print object.
GetPdf	Retrutn a PDF object.
GetImagePdf	Retrutn an image PDF object.

IObjects Interface / z_Objects Object

IObjects Interface and z_Objects Object are objects and its property class to obtain and set each property in each object in designing.
It is possible to change properties such as object color, position and font by the use of this class during runtime.

Public Method

Set Object	Object setting to edit property by specifying object name

Public Property

z_Text	Property for character string object
z_Line	Property for ruled line object
z_Square	Property for square object
z_Circle	Property for circle object
z_Image	Property for image object
z_Barcode	Property for barcode object
z_ArtText	Property for decorated character object

LoadDefFile Method

A report definition file is read with the LoadDefFile Method.
We recommend the absolute path because it is not sure where a program would run.

<C#.NET>
void LoadDefFile(string name)
string name
    report definition file name

<VB.NET>
Sub LoadDefFile(name As String)
name As String
    report definition file name

＜Example for C#.NET＞

```
// Read report definition fine
paoRep.LoadDefFile("C:¥¥ report definition fine.xml");
```

＜Example for VB.NET＞

```
' Read report definition fine
paoRep.LoadDefFile("C:¥ report definition fine.xml")
```

Reference

IReport interface

PageStart Method

Declare a start of a page with PageStart Method.
Put a code to set print data between start and end (PageEnd) declarations of a page.

＜C#.NET＞

void PageStart()

＜VB.NET＞

Sub PageStart()

＜Example for C#.NET＞

```
//Declare a star of a page
paoRep.PageStart();
      :
      :
//Declare an end of the page
paoRep.PageEnd();
```

Write() ---processing of print data set

＜Example for VB.NET＞

```
'Declare a star of a page
paoRep.PageStart()
      :
      :
'Declare an end of the page
paoRep.PageEnd()
```

Write()---processing of print data set

Reference

IReport interface

PageEnd Method

Declare an end of a page with PageEnd Method.
Put a code to set print data between start (PageStart) and end declarations of a page (this method).

<C#.NET>
void PageEnd()

<VB.NET>
Sub PageEnd()

<Example for C#.NET>

```
//Declare a star of a page
paoRep.PageStart();
    :            Write()---processing of print data set
    :
//Declare an end of the page
paoRep.PageEnd();
```

<Example for VB.NET>

```
'Declare a star of a page
paoRep.PageStart()
    :            Write()---processing of print data set
    :
'Declare an end of the page
paoRep.PageEnd()
```

Reference

IReport interface

Write Method

With Write Method, operate object specified by report definition file, such as writing characters or drawing horizontal ruled lines repeatedly to the object specified by report definition file.

List of Overload

＜C#.NET＞

void Write(string name, string value)

This sets a character string to an object.Use this to set a value of unrepeated persistent objects such as header and footer.

void Write(string name, string value, int index)

This specifies a drawing position for an object and set a character string.

Use this for objects to set repeated values such as lines in a chart.

void Write(string name, int index)

This specifies a drawing position for an object

Use this for objects to set repeated values such as horizontal ruled lines in a chart.

＜VB.NET＞

Sub Write(name As String, value As String)

This sets a character string to an object.

Use this to set a value of unrepeated persistent objects such as header and footer.

Sub Write(name As String, value As String, index As Long)

This specifies a drawing position for an object and set a character string.

Use this for objects to set repeated values such as lines in a chart.

Sub Write(name As String, index As Long)

This specifies a drawing position for an object

Use this for objects to set repeated values such as horizontal ruled lines in a chart.

Reference

IReport interface

void Write(string name, string value) Method

This sets a character string to an object.
Use this to set a value of unrepeated persistent objects such as header and footer.

string name

This specifies a name of an object in report definition file.
The intentded (intended) object types are basically Text (character string), ArtText (decorated character string) and Barcode (barcode) because this sets character string.
In deleting objects, specify object other than Text (character string) or ArtText (decorated character string)

string value

This specifies a character string to set.
If an object other than Text (character string) or ArtText (decorated character string) is set with a blank (""), the object will be deleted.

<Example>

```
//Set character strings
paoRep.Write("time and date", System.DateTime.Now.ToString());
```

Reference

IReport interface

void Write(string name, string value, int index) Method

This specifies a drawing position for an object and set a character string.
Use this for objects to set repeated values such as lines in a chart.
When using the method for this pattern, IntervalX or IntervalY in report definition file should be put with a value more than 1.
IntervalX means intervals repeated in a transverse direction by milimeter.
IntervalY means intervals repeated in a longitudinal direction by millimeter.　This is mainly used for lines of a chart.

string name

This specifies a name of an object in report definition file.
The intended object types are basically Text (character string), ArtText (decorated character string) and Barcode (barcode) because this sets character string.
In deleting objects, specify object other than Text (character string) or ArtText (decorated character string).

string value

This specifies a character string to set.
If an object other than Text (character string) or ArtText (decorated character string) is set with a blank (“”), the object will be deleted.

int index

This means drawing position in the page set in a transverse and a longitudinal direction intervals by IntervalX／IntervalY.　The values get bigger from upper left to lower right.
For example, a chart with a value in IntervalY has a drawing position such as:
(the first position of an object) + InterbalY　×　(index –1).
As for a chart, 1 is in the first line, 2 in the second and 3 in the third.

<Example>

```
//Set repeated character strings
paoRep.Write("No.", “1” , 1);
```

Reference

IReport interface

void Write(string name, int index) Method

This specifies a drawing position for an object
Use this for objects to set repeated values such as horizontal ruled lines in a chart.
When using the method for this pattern, IntervalX or IntervalY in report definition file should be put with a value more than 1.
IntervalX means intervals repeated in a transverse direction by milimeter.
IntervalY means intervals repeated in a longitudinal direction by millimeter.　This is mainly used for lines of a chart.

string name

This specifies a name of an object in report definition file.
All objects can be applied because any objects can be drawn repeatedly.

int index

This means printing position in the page set in a transverse and a longitudinal direction intervals by IntervalX／IntervalY.　The values get bigger from upper left to lower right.
For example, a chart with a value in IntervalY has a printing position such as:
(the first position of an object) + InterbalY　×　(index –1).
As for a chart, 1 is in the first line, 2 in the second and 3 in the third.

<Example>

```
//Set repeated character strings
paoRep.Write("horizontal line", 1);
```

Reference

IReport interface

Sub Write(name As String, value As String) Method

This sets a character string to an object.
Use this to set a value of unrepeated persistent objects such as header and footer.

name As String

This specifies a name of an object in report definition file.
The intentded (intended) object types are basically Text (character string), ArtText (decorated character string) and Barcode (barcode) because this sets character string.
In deleting objects, specify object other than Text (character string) or ArtText (decorated character string)

value As String

This specifies a character string to set.
If an object other than Tex (character string) or ArtText (decorated character string) is set with a blank (""), the object will be deleted.

<Example>

```
'Set character strings
paoRep.Write("time and date", System.DateTime.Now.ToString())
```

Reference

IReport interface

Sub Write(name As String, value As String, index As Long) Method

This specifies a drawing position for an object and set a character string.
Use this for objects to set repeated values such as lines in a chart.
When using the method for this pattern, IntervalX or IntervalY in report definition file should be put with a value more than 1.
IntervalX means intervals repeated in a transverse direction by milimeter.
IntervalY means intervals repeated in a longitudinal direction by millimeter. This is mainly used for lines of a chart.

name As String

This specifies a name of an object in report definition file.
The intended object types are basically Text (character string), ArtText (decorated character string) and Barcode (barcode) because this sets character string.
In deleting objects, specify object other than Text (character string) or ArtText (decorated character string)

value As String

This specifies a character string to set.
If an object other than Text (character string) or ArtText (decorated character string) is set with a blank (""), the object will be deleted.

index As Long

This means drawing position in the page set in a transverse and a longitudinal direction intervals by IntervalX／IntervalY. The values get bigger from upper left to lower right.
For example, a chart with a value in IntervalY has a drawing position such as:
(the first position of an object) + InterbalY × (index –1).
As for a chart, 1 is in the first line, 2 in the second and 3 in the third.

<Example>
```
paoRep.Write("No.", "1", 1)
```

Reference

IReport interface

Sub Write(name As String, index As Long) Method

This specifies a drawing position for an object
Use this for objects to set repeated values such as horizontal ruled lines in a chart.
When using the method for this pattern, IntervalX or IntervalY in report definition file should be put with a value more than 1.
IntervalX means intervals repeated in a transverse direction by milimeter.
IntervalY means intervals repeated in a longitudinal direction by millimeter.　This is mainly used for lines of a chart.

name As String

This specifies a name of an object in report definition file.
All objects can be applied because any objects can be drawn repeatedly.

index As Long

This means printing position in the page set in a transverse and a longitudinal direction intervals by IntervalX／IntervalY.　The values get bigger from upper left to lower right.
For example, a chart with a value in IntervalY has a printing position such as:
(the first position of an object) + InterbalY　×　(index –1).
As for a chart, 1 is in the first line, 2 in the second and 3 in the third.

<Example>
```
paoRep.Write("horizontal line", 1)
```

Reference

IReport interface

Output Method

This makes a ledger sheet printed out from a printer or displayed with preview window.

List of Overload

<C#.NET>

bool Output()

This orders print or preview to a default printer in default settings.

bool Output(System.Drawing.Printing.PrinterSettings setting)

This orders print or preview in the printer settings specified by argument.

<VB.NET>

Function Output() As Boolean

This orders print or preview to a default printer in default settings.

Function Output(setting As System.Drawing.Printing.PrinterSettings) As Boolean

This orders print or preview in the printer settings specified by argument.

Rerefence

IReport interface

Output() Method

This orders print or preview to a default printer in default settings.

<C#.NET>

```
bool Output()
```

<VB.NET>

```
Function Output() As Boolean
```

<Example for C#.NET>

```
paoRep.Output(); //Execute print or preview
```

<Example for VB.NET>

```
paoRep.Output() 'Execute print or preview
```

Reference

IReport interface

Output method

Output(System.Drawing.Printing.PrinterSettings setting) Method

This orders print or preview in the printer settings specified by argument.

<C#.NET>

bool Output(System.Drawing.Printing.PrinterSettings setting)

<VB.NET>

Function Output(setting As System.Drawing.Printing.PrinterSettings) As Boolean

<Example for C#.NET>

```
System.Drawing.Printing.PrinterSettings setting
      = new System.Drawing.Printing.PrinterSettings();
setting.PrinterName = “printer name”;
paoRep.Output(setting); //Execute print or preview
```

<Example for VB.NET>

```
Dim setting As System.Drawing.Printing.PageSettings
      = New System.Drawing.Printing.PageSettings()
setting.PrinterName = “printer name”
paoRep.Output(setting) 'Execute print or preview
```

Rerefence

IReport interface

Output method

SaveXMLFile Method

This method saves print data to XML file.
Saved files can be read with a program (LoadXMLFile) or from preview window.

< C#.NET>
bool SaveXMLFIle(string name)
string name
    Print data XML file path name to be saved

<VB.NET>
SaveXMLFIle(name As String) As Boolean
name As String
    Print data XML file path name to be saved

<Example for C#.NET>

```
paoRep.SaveXMLFile("print data.XML"); //Save the print data
```

<Example for VB.NET>

```
paoRep.SaveXMLFile("print data file.xml") 'Save the print data
```

Reference

IReport interface

LoadXMLFile Method

With this method, a print data XML file saved by SaveXMLFile is read.
Read print data can be printed out or previewed (Output).

< C#.NET >
bool LoadXMLFIle(string name)
string name
    Print data XML file path name to be read

< VB.NET >
LoadXMLFIle(name As String) As Boolean
name As String
    Print data XML file path name to be read

<Example for C#.NET >

```
paoRep.LoadXMLFile("print data.XML"); //Read the print data
```

<Example for VB.NET >

```
paoRep.LoadXMLFile("print data file.xml") 'Read the print data
```

Reference

IReport interface

SaveData Method

This returns compressed print binary data.
Use this method to create print data to be returned to a rich client at Web service.

< C#.NET >
```
byte[] SaveData()
```

< VB.NET >
```
SaveData() As Byte()
```

<Example for C#.NET >
```
byte[] b = paoRep. SaveData(); //Return compressed print binary data
```

<Example for VB.NET >
```
Dim b As Byte() = paoRep. SaveData 'Return compressed print binary data
```

Rerefence
IReport interface

LoadData Method

With this method, a compressed print binary data created by SaveData is read.
Use this method at a rich client to read a print data created at Web service.

< C#.NET >

bool LoadData(string name)

string name

Print data XML file path name to be read

< VB.NET >

LoadData(name As String) As Boolean

name As String

Print data XML file path name to be read

<Example for C#.NET >

```
byte[] data = webService.getPrintData();
IReport paoRep = ReportCreator.GetPreview()     //Create preview object
paoRep.LoadData(data); //Read compressed print binary data
paoRep.Output(); //Preview
```

<Example for VB.NET >

```
Dim data As Byte() = webTest.getledger sheet data() 'Obtain print data
Dim paoRep As IReport = ReportCreator.GetPreview() 'Create preview object
paoRep.LoadData(data) ' Read print data
paoRep.Output() ' Execute preview
```

Reference

IReport interface

SaveSVGFile Method

This method writes a print data in SVG format.

< C#.NET >
bool SaveSVGFile(string name)
string name
  Print data html file path name to be written
  Specify the html file name to read SVG file because SVG files will be created as many as its numbers of pages.

< VB.NET >
SaveSVGFile(name As String) As Boolean
name As String
  Print data SVG file path name to be written
  Specify the html file name to read SVG file because SVG files will be created as many as its numbers of pages.

<Example for C#.NET >

```
paoRep.SaveSVGFile("print data.html"); //Write SVG data
```

<Example for VB.NET >

```
paoRep. SaveSVGFile("print data file.html") 'Write SVG data
```

Reference
IReport interface

SaveSVGZFile Method

This method writes a print data in SVGZ format.

< C#.NET >
bool SaveSVGZFile(string name)
string name
Print data html file path name to be written
Specify the html file name to read SVGZ file because SVGZ files will be created as many as its numbers of pages.

< VB.NET >
SaveSVGZFile(name As String) As Boolean
name As String
Print data SVGZ file path name to be written
Specify the html file name to read SVGZ file because SVGZ files will be created as many as its numbers of pages.

<Example for C#.NET >

```
paoRep.SaveSVGZFile("print data.html"); //Write SVGZ data
```

```
paoRep.SaveSVGZFile("print data file.html") 'Write SVGZ data
```

Reference
IReport interface

SavePDF Method (Stream)

This method writes a print data in PDF format. (Stream)

< C#.NET >
bool SavePDF (System.IO.Stream stream)
System.IO.Stream stream
    Print data PDF Stream to be written

< VB.NET >
SavePDF (name As System.IO.Stream) As Boolean
name As System.IO.Stream
    Print data PDF Stream to be written

<Example for C#.NET >

```
paoRep.SavePDF(anyStream); //Write PDF data
```

<Example for VB.NET >

```
paoRep.SavePDF(anyStream) 'Write PDF data
```

Reference

IReport interface

SavePDF Method (File)

This method writes a print data in PDF format. (File)

< C#.NET >
bool SavePDF (string name)
string name
    Print data PDF file path name to be written

< VB.NET >
SavePDF (name As String) As Boolean
name As String
    Print data PDF file path name to be written

<Example for C#.NET >

```
paoRep.SavePDF("print data.PDF"); //Write PDF data
```

<Example for VB.NET >

```
paoRep.SavePDF("print data file.pdf") 'Write PDF data
```

Reference

IReport interface

DisplayDialog Property

When printing with the use of Output method, this method specifies whether or not to display [print] dialog box. The default setting is true which means display. This property has its effect only in printing but no effect when it's specified in preview.

< C#.NET >

bool DisplayDialog

true: Display a print dialog box in printing. (default value)

false: Hide a print dialog box in printing.

< VB.NET >

DisplayDialog As Boolean

True: Display a print dialog box in printing. (default value)

False: Hide a print dialog box in printing.

<Example for C#.NET >

```
paoRep.DisplayDialog = false;    //Hide a print dialog
paoRep.Output(); //Print out
```

<Example for VB.NET >

```
paoRep.DisplayDialog = False    'Hide a print dialog
paoRep.Output() 'Print out
```

Reference

IReport interface

DisplayPrinting Property

When printing with the use of Output method, this method specifies whether or not to display "in printing (the number of pages)". The default setting is true which means display. This property has its effect only in printing but no effect when it's specified in preview.

< C#.NET >

bool DisplayPrinting

true: Display "in printing (the number of pages)" in printing. (default value)

false: Hide "in printing (the number of pages)" in printing.

< VB.NET >

DisplayPrinting As Boolean

True: Display "in printing (the number of pages)" in printing. (default value)

False: Hide "in printing (the number of pages)" in printing.

<Example for C#.NET >

```
paoRep.DisplayPrinting = false;   //Hide  "in printing (the number of pages)"
paoRep.Output(); //Print out
```

<Example for VB.NET >

```
paoRep.DisplayPrinting   = False  'Hide  "in printing (the number of pages)"
paoRep.Output() 'Print out
```

Reference

IReport interface

PreviewDialog Property

When printing with the use of Output method, this method obtains or specifies whether or not to display preview window. The default setting is true which means dialog display. When the property is set to false, multipul preview windows can be activated simultaneously because those are runned as usual forms. In other words, it's modeless form.

< C#.NET >

bool PreviewDialog

true: Activate a dialog window (modeless (model) form) in previewing

false: Activate a usual form (modeless form) in previewing.

< VB.NET >

PrviewDialog As Boolean

True: Activate a dialog window (modeless (model) form) in previewing

False: Activate a usual form (modeless form) in previewing.

<Example for C#.NET >

```
paoRep.PreviewDialog = false;   //If you would like ot run multiple preview windows
simultaneously
```

<Example for VB.NET >

```
paoRep. PreviewDialog = False   'If you would like ot run multiple preview windows
simultaneously
```

Reference

IReport interface

AccessFile Property

When printing with the use of Output method, this method specifies whether or not to allow a file access such as file saving from a preview window. The default setting is true which means display.

< C#.NET >

bool AccessFile

  true: Allow a file access from a preview window in printing. (default value)

  false: Disallow a file access from a preview window in printing.

< VB.NET >

AccessFile As Boolean

  True: Allow a file access from a preview window in printing. (default value)

  False: Disallow a file access from a preview window in printing.

<Example for C#.NET >

```
paoRep.AccessFile = false;    //Disallow a file access from a preview window
```

```
paoRep. AccessFile = False    'Disallow a file access from a preview window
```

Reference

IReport interface

MarginTop Property

When printing with the use of Output method, this method set head margin by millimeter in previewing.
It is effective only in printing or previewing.
This method facilitates fine adjustment for different output results depending on printers.

< C#.NET >
float MarginTop

< VB.NET >
MarginTop As float

<Example for C#.NET >

```
paoRep. MarginTop = 10;    //Set head margin to 1cm
```

<Example for VB.NET >

```
paoRep. MarginTop = 10   ‘Set head margin to 1cm
```

Reference

IReport interface

MarginLeft Property

When printing with the use of Output method, this method set left margin by millimeter in previewing.
It is effective only in printing or previewing.
This method facilitates fine adjustment for different output results depending on printers.

< C#.NET >
float MarginLeft

< VB.NET >
MarginLeft As float

<Example for C#.NET >

```
paoRep. MarginLeft = 10;    //Set left margin to 1cm
```

<Example for VB.NET >

```
paoRep. MarginLeft = 10    'Set left margin to 1cm
```

Reference

IReport interface

z_Objects Property / IObjects Interface

This is used to set or obtain values of each property in designing each object.
With this property, it is possible to change object color, position and font during runtime.

<Example for C#.NET >

```
//Change character position, font size and bold of character string object
paoRep.z_Objects.SetObject("character string");
paoRep.z_Objects.z_Text.TextAlign = PmAlignType.Right;
paoRep.z_Objects.z_Text.z_FontAttr.Size = 8;
paoRep.z_Objects.z_Text.z_FontAttr.Bold = true;
```

<Example for VB.NET >

```
'Change character position, font size and bold of character string object
paoRep.z_Objects.SetObject("character string ")
paoRep.z_Objects.z_Text.TextAlign = PmAlignType.Right
paoRep.z_Objects.z_Text.z_FontAttr.Size = 8
paoRep.z_Objects.z_Text.z_FontAttr.Bold = True
```

Reference

IReport interface

z_Text property

z_Line property

z_Square property

z_Circle property

z_Barcode property

z_Image property

z_ArtText property

SetObject (string　objName) Method

This method specifies which object property will be obtained or set.
Specify the object name used in designing as an argument.
After calling this method, property value of the object specified by an argument can be obtained or set.

< C#.NET >
bool SetObject(string objName)

< VB.NET >
Function SetObject(String objName) As Boolean

<Example for C#.NET >

```
paoRep.z_Objects.SetObject("Object name"); //Specify object to be edited with its property
```

<Example for VB.NET >

```
paoRep.z_Objects.SetObject("Object name") 'Specify object to be edited with its property
```

Reference
IObjects interface / z_Objects property

z_Text Property / ZText Class

This class object is one class below the z_Objects. In order to obtain and set each property values of character (text) string object, obtain and set values of the property under z_Text.

It is possible to obtain and set the following property values under Z_Text.

Type(C#)	Type(VB.NET)	Property Name	Explanation
Float	Single	**Angle**	Angle of rotation
System.Drawing.Color		**BackColor**	Background color (for character strings and images)
Float	Single	**Height**	Range of drawing (height)
Float	Single	**IntervalX**	Interval of drawing (to x-coordinate)
Float	Single	**IntervalY**	Interval of drawing (to y-coordinate)
Bool	Boolean	**IsElastic**	Stretchable
System.Drawing.Color		**OutLineColor**	Outline color
Float	Single	**OutLineWidth**	Outline width
Int	Integer	**Repeat**	Repeated times
String	String	**Text**	Displayed character string
Pao.Reports.PmAlignType		**TextAlign**	Display position
Float	Single	**Width**	Range of drawing (width)
Float	Single	**X**	x-coordinate of origin (upper left)
Float	Single	**Y**	y-coordinate of origin (upper left)
Pao.Reports.ZFontAttr		**z_FontAttr**	Font attribute

Reference

IObjects interface / z_Objects property

ZFontAttr class / z (Font) Attr property

z_Line Property / ZLine Class

This class object is one class below the z_Objects. In order to obtain and set each property values of ruled line object, obtain and set values of the property under z_Line.

It is possible to obtain and set the following property values under Z_Line.

Type(C#)	Type(VB.NET)	Property Name	Explanation
float	Single	**EndX**	x-coordinate of ruled line end
float	Single	**EndY**	y-coordinate of ruled line end
float	Single	**IntervalX**	Interval of drawing (to x-coordinate)
float	Single	**IntervalY**	Interval of drawing (to y-coordinate)
int	Integer	**Repeat**	Repeated times
float	Single	**Thick**	Thickness of ruled line circular arc
float	Single	**X**	x-coordinate of origin (upper left)
float	Single	**Y**	y-coordinate of origin (upper left)
Pao.Reports.ZLineAttr		**z_LineAttr**	Ruled line attribute

Reference

IObjects interface / z_Objects property

ZLineAttr class / z_LineAttr property

z_Square Property / ZSquare Class

This class object is one class below the z_Objects. In order to obtain and set each property values of square object, obtain and set values of the property under z_Square.

It is possible to obtain and set the following property values under Z_Square.

Type(C#)	Type(VB.NET)	Property Name	Explanation
Float	Single	**Angle**	Angle of rotation
Pao.Reports.ZCornerType		**CornerType**	Type of square corner
Int	Integer	**HatchDensity**	Density of hatching (%)
Float	Single	**Height**	Range of drawing (height)
Float	Single	**IntervalX**	Interval of drawing (to x-coordinate)
Float	Single	**IntervalY**	Interval of drawing (to y-coordinate)
System.Drawing.Color		**PaintColor**	Color used to paint
Float	Single	**R**	Value describing roundness of square corner
Int	Integer	**Repeat**	Repeated times
Float	Single	**Width**	Range of drawing (width)
Float	Single	**X**	x-coordinate of origin (upper left)
Float	Single	**Y**	y-coordinate of origin (upper left)
Pao.Reports.ZLineAttr		**z_LineAttr**	Ruled line attribute

Reference

IObjects interface / z_Objects property

ZLineAttr class / z_LineAttr property

z_Circle Property / ZCircle Classs

This class object is one class below the z_Objects. In order to obtain and set each property values of circle object, obtain and set values of the property under z_Circle.

It is possible to obtain and set the following property values under Z_Circle.

Type(C#)	Type(VB.NET)	Property Name	Explanation
Float	Single	**Angle**	Angle of rotation
Int	Integer	**HatchDensity**	Density of hatching (%)
Float	Single	**Height**	Range of drawing (height)
Float	Single	**IntervalX**	Interval of drawing (to x-coordinate)
Float	Single	**IntervalY**	Interval of drawing (to y-coordinate)
System.Drawing.Color		**PaintColor**	Color used to paint
Int	Integer	**Repeat**	Repeated times
Float	Single	**Width**	Range of drawing (width)
Float	Single	**X**	x-coordinate of origin (upper left)
Float	Single	**Y**	y-coordinate of origin (upper left)
Pao.Reports.ZLineAttr		**z_LineAttr**	Ruled line attribute

Reference

IObjects interface / z_Objects property

ZLineAttr class / z_LineAttr property

z_Image Property / ZImage Class

This class object is one class below the z_Objects. In order to obtain and set each property values of image object, obtain and set values of the property under z_Image.

It is possible to obtain and set the following property values under z_Image.

Type(C#)	Type(VB.NET)	Property Name	Explanation
Float	Single	**Angle**	Angle of rotation
System.Drawing.Color		**BackColor**	Background color (for character strings and images)
Float	Single	**Height**	Range of drawing (height)
Pao.Reports.PmImgAlignType		**ImageAlign**	Image psition
String	String	**ImageData**	Path or data of image file
Pao.Reports.PmImgRevType		**ImageRev**	Image reversal
Float	Single	**IntervalX**	Interval of drawing (to x-coordinate)
Float	Single	**IntervalY**	Interval of drawing (to y-coordinate)
Int	Integer	**Repeat**	Repeated times
Float	Single	**Width**	Range of drawing (width)
Float	Single	**X**	x-coordinate of origin (upper left)
Float	Single	**Y**	y-coordinate of origin (upper left)
Pao.Reports.ZLineAttr		**z_LineAttr**	Ruled line attribute

Reference

IObjects interfac / z_Objects property

ZLineAttr class / z_LineAttr property

z_Barcode Property / ZBarcode Class

This class object is one class below the z_Objects. In order to obtain and set each property values of image object, obtain and set values of the property under z_Barcode.

It is possible to obtain and set the following property values under z_Barcode.

Type(C#)	Type(VB.NET)	Property Name	Explanation
Float	Single	**Angle**	Angle of rotation
Bool	Boolean	**DispStartStop**	Whether or not to display start/stop code.
Float	Single	**Height**	Range of drawing (height)
Float	Single	**IntervalX**	Interval of drawing (to x-coordinate)
Float	Single	**IntervalY**	Interval of drawing (to y-coordinate)
Bool	Boolean	**IsWriteDirect**	Whether or not to draw directly
Pao.Reports.PmBarcodeType		**Kind**	The kind of bardoce
Bool	Boolean	**Kintou**	Wheter or not to space subscript equally
Int	Integer	**KuroBar**	Adjust the width of black bars by dot
Float	Single	**Point**	Point of postal customer barcode
String	String	**QrErrCorrect**	Error-correcting level for QE code (L/M/Q/H)
Int	Integer	**QrVersion**	QR code version (1-40)
Int	Integer	**Repeat**	Repeated times
Int	Integer	**ShiroBar**	Adjust the width of white bars by dot
Bool	Boolean	**Soeji**	Wheter or not to display subscript
Float	Single	**Width**	Range of drawing (width)
Float	Single	**X**	x-coordinate of origin (upper left)
Float	Single	**Y**	y-coordinate of origin (upper left)
Pao.Reports.ZFontAttr		**z_FontAttr**	Font attribute

Reference

IObjects interface / z_Objects property

ZFontAttr class / z_FontAttr property

z_ArtText Property / ZArtText Class

This class object is one class below the z_Objects. In order to obtain and set each property values of decorated character string object, obtain and set values of the property under z_ArtText.

It is possible to obtain and set the following property values under z_ArtText.

Type(C#)	Type(VB.NET)	Property Name	Explanation
Float	Single	Angle	Angle of rotation
System.Drawing.Color		BackColor	Background color (for character strings and images)
Int	Integer	CharAngle	Angle of charecter rotation
System.Drawing.Color		Color	Character color
Int	Integer	DelimiterPileRatef	Digit grouping and overlaying rate_front
Int	Integer	DelimiterPileRater	Digit grouping and overlaying rate_back
Bool	Boolean	DelimiterProcess	Digit grouping process
String	String	DelimiterString	Target character of digit grouping
Bool	Boolean	FontBold	Bold font
String	String	FontName	Font name
Float	Single	Height	Range of drawing (height)
Float	Single	IntervalX	Interval of drawing (to x-coordinate)
Float	Single	IntervalY	Interval of drawing (to y-coordinate)
System.Drawing.Color		Color	Outline color
Float	Single	OutLineWidth	Outline width
Bool	Boolean	PileOrderLeftFront	Overlaying left front
Int	Integer	PileRate	Overlaying rate
Bool	Boolean	ProjectionX	Flip vertical
Bool	Boolean	ProjectionY	Flip horizontal
Int	Integer	Repeat	Repeated times
Bool	Boolean	RevText	Reversal
System.Drawing.Color		Color	Shadowed character color
System.Drawing.Color		Color	Shadowed line color

Float	Single	ShadowLineWidth	Shadowed line width
Bool	Boolean	ShadowStretch	Shadow and stretch
Float	Single	ShadowX	Shadow X position
Float	Single	ShadowY	Shadow Y position
Bool	Boolean	ShearStretch	Stretch by oblique
Float	Single	ShearX	Oblique type_horizontal
Float	Single	ShearY	Oblique type_vertical
String	String	Text	Displayed character string
Float	Single	Width	Range of drawing (width)
Bool	Boolean	WriteVertically	Vertical writing
Float	Single	X	x-coordinate of origin (upper left)
Float	Single	Y	y-coordinate of origin (upper left)
Pao.Reports.ZFontAttr		z_FontAttr	Font attribute

Reference

IObjects interface / z_Objects property

ZFontAttr class / z_FontAttr property

z_FontAttr Property / ZFontAttr Class

This class is for font property of each object with font attribute. By obtaining and setting property values under z_FontAttr, it is also possible to obtain and set values of the character string property used in each object.

It is possible to obtain and set the following property values under z_FontAttr.

Type(C#)	Type(VB.NET)	Property Name	Explanation
Bool	Boolean	**Bold**	True in case of bold
System.Drawing.Color		**Color**	Character color
Bool	Boolean	**Italic**	True in case of italic
String	String	**Name**	Font name
Float	Single	**Size**	Font size
Bool	Boolean	**Strikeout**	Strike-thorough
Bool	Boolean	**UnderLine**	Under line
System.Drawing.GraphicsUnit		**Unit**	Unit of font hight

Reference

ZText class / z_Text object

ZBarcode class / z_Barcode object

ZArtText class / z_ArtText object

z_LineAttr Property / ZLineAttr Class

This class is for font property of each object with font attribute. By obtaining and setting property values under z_LineAttr, it is also possible to obtain and set values of the ruled line property used in each object.

It is possible to obtain and set the following property values under z_LineAttr.

Type(C#)	Type(VB.NET)	Property Name	Explanation
System.Drawing.Color		Color	Ruled line color
float	Single	DashLine	Length of dash line
float	Single	DashPattern	Dash line pattern
float	Single	DashSpace	Length of blank between dashes
Pao.Reports.PmLineStyle		Style	Ruled line style
Pao.Reports.PmLineType		Type	Type
float	Single	Width	Ruled line width

Reference

ZSquare class / z_Square object

ZCircle class / z_Circle object

ZLine class / z_Line object

ZImage class / z_Image object

GetPreview Method

This method is to return object controlling preview.
Use GetReport method to print out directly.

< C#.NET >
IReport GetPreview()

< VB.NET >
Function GetPreview() As IReport

<Example for C#.NET >
```
//Obtain an instance of preview object
paoRep = ReportCreator.GetPreview();
```

<Example for VB.NET >
```
'Obtain an instance of preview object
paoRep = ReportCreator.GetPreview()
```

Reference

ReportCreator class

GetReport Method

This method is to return object controlling print.
Use GetPreview method to preview.

< C#.NET >
IReport GetReport()

< VB.NET >
Function GetReport() As IReport

<Example for C#.NET >
```
//Obtain an instance of print object
paoRep = ReportCreator.GetReport();
```

<Example for VB.NET >
```
'Obtain an instance of print object
paoRep = ReportCreator.GetReport()
```

Reference

ReportCreator class

GetPdf Method

This method is to return PDF object.

< C#.NET >

IReport GetPdf()

< VB.NET >

Function GetPdf() As IReport

<Example for C#.NET >

```
//Obtain an instance of PDF object
paoRep = ReportCreator.GetPdf();
```

<Example for VB.NET >

```
'Obtain an instance of PDF object
paoRep = ReportCreator.GetPdf()
```

Reference

ReportCreator class

GetImagePdf Method

This method is to return image PDF object.

< C#.NET >

IReport GetImagePdf()

< VB.NET >

Function GetImagePdf () As IReport

<Example for C#.NET >

```
//Obtain an instance of image PDF object
paoRep = ReportCreator.GetImagePdf ();
```

<Example for VB.NET >

```
' Obtain an instance of image PDF object
paoRep = ReportCreator.GetImagePdf ()
```

Reference

ReportCreator class

Modification History

Version	Release Date	Modification
1	May 25, 2003	First release
2	June 10, 2003	Compliant with QR code, Web service and PDF.
3	August 5, 2006	Compliant with ZIP, SVG and SVGZ. Addition of properties such as print dialog.
4	March 2, 2006	Addition of delete function of objects by setting blank in a value of Write method.
5	November 9, 2010	SavePDF method: addition of Stream output
6	February 28, 2011	Functional addition of obtaining and setting object property in designing. (z_Objects)
7	July 1, 2011	English version release