

Nreport Generic Report Engine – 3.0.0.1

Reporting for the .Net 4 environment

- Implement reporting capability to your .Net application
- Instantly converts data from a [DataTable](#) into a report
- Minimal settings required
- Easy implementation

Contents

Introduction	2
Requirements.....	2
Available Versions.....	2
How to use the Nreport.dll	4
Adding the reference to your project	4
Creating your first Windows Forms report	5
Delving further	8
Adding a Logo.....	9
Create a byte array from an image stored in a SQL table.....	9
Create a byte array from an image on disk.....	9
Create a byte array from an image embedded in your application.....	10
Functionality derived from Column Heading naming conventions	10
Common Errors.....	11
No DataSet	11
DataSet name not specified.....	12
All columns are “Group By” columns.....	12
Some columns are displaying too small.....	12
Other Reporting Products.....	12
Licence	12
Disclaimer.....	13

Introduction

The Nreport.dll is built for the .Net 4 Framework and creates Microsoft Reports on the fly. The report engine returns a memory stream containing valid report XML which can be loaded into a standard .Net Report Viewer control (either web based or forms based)

No knowledge of Microsoft Reporting or XML is required!

The only required setting is a populated [DataTable](#)

Requirements

- .Net 4
- Visual Studio 2010
- A form / web page * with a standard ReportViewer control

* NOTE: The ReportViewer control works best in IE and is not supported in other browsers. Consult Microsoft documentation for implementing a MS report in a web interface.

Available Versions

Properties	Nreport Full Version	Nreport Light (free version)
Custom Title Text	✓	✓
Custom Subtitle Text	✓	✓
Custom Logo	✓	✓
Highlighted Row Background Colour	✓	✓
Highlighted Row Text Colour	✓	✓
Highlighted Row Frequency	✓	✓
Custom Report Footer (Left)	✓	✓
Custom Report Footer (Right)	✓	✗

Advanced Properties	Nreport Full Version	Nreport Light (free version)
Paper Size	✓	✓
Logo Sizing Options	✓	✗
Margin Left, Right, Top, Bottom	✓	✗

Normal Background Colour	✓	✗
Normal Text Colour	✓	✗
Column Heading Colour	✓	✗
Column Heading Text Colour	✓	✗
Advanced Logo Sizing Options	✓	✗

Report Functionality	Nreport Full Version	Nreport Light (free version)
Standard Report	✓	✓
Group By (multiple levels)	✓	✓
Group By – starting on new page	✓	✗
Field Sum	✓	✗
Row Count	✓	✗
Export to CSV (not yet available)	✓	✗

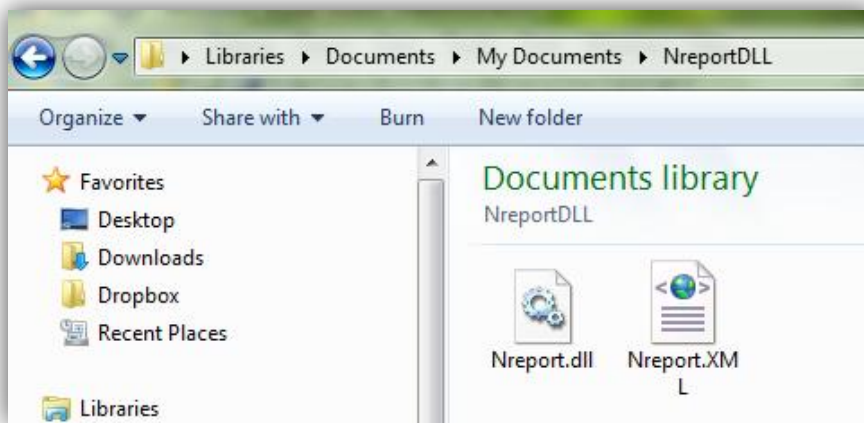
How to use the Nreport.dll

The examples below will explain how to implement Nreport.dll into your application, and how to run a simple report. The examples given are using Visual Studio 2010 and show both C# and Visual Basic syntax.

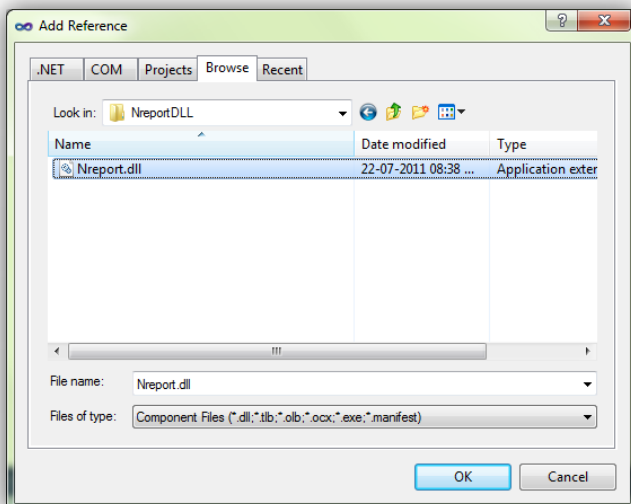
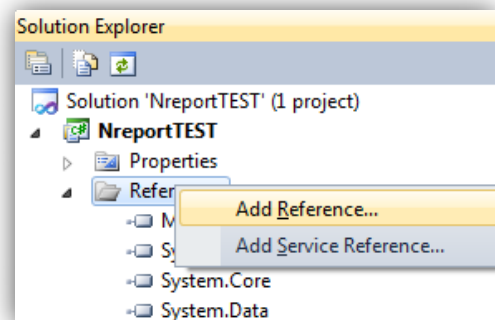
Adding the reference to your project

Save the Nreport.dll to a location on your development machine. (Save the Nreport.XML to the same directory too for added code completion hints and Object Browser details)

For this example I have saved the files to a folder called "NreportDLL" in "My Documents"



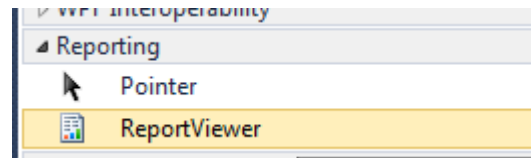
In a Visual Studio application, right-click on the References (in your Solution Explorer) and select "Add Reference..."



Now select the dll from the location where you saved it

Creating your first Windows Forms report

Using the Toolbox, add a ReportViewer to either a form or a web page. The example below is using a Windows Form.



This example will create and display a report on the Form Load event

```
C#
private void Form1_Load(object sender, EventArgs e)
{
    //Declare a DataTable
    DataTable dt;

    // A valid connection string to your database
    string connectionString = "Data Source=(local);Initial Catalog=Nreports;" +
        "Integrated Security=True";

    //Use your own application functions to populate the DataTable with data
    //Below is just a quick method to connect to a database and run a query
    //against it
    //The Nreport Dll only needs data in a DataTable - irrespective of how
    //you populate it!
    SqlConnection dbConn = new SqlConnection();
    dbConn.ConnectionString = connectionString;
    dbConn.Open();

    SqlCommand command = new SqlCommand("SELECT * FROM Users", dbConn);
    SqlDataAdapter adapter = new SqlDataAdapter(command);
    DataSet ds = new DataSet();

    try
    {
        adapter.Fill(ds);
    }
    catch { }

    adapter.Dispose();
    command.Dispose();
    dbConn.Close();
    dbConn.Dispose();

    //Here we actually get the DataSet
    dt = ds.Tables[0];

    try
    {
        //Create a new Report Definition (this allows you to change report properties)
        Nreport.ReportDefinition reportSettings = new Nreport.ReportDefinition();
        //Set the Data Table To Use
        reportSettings.DataTableToUse = dt;
        //Create a new ReportBuilder and pass the settings object
        Nreport.ReportBuilder report = new Nreport.ReportBuilder(reportSettings);

        //!!! Important !!!
        //Very important to add a DataSource to your "local report" called
```

```
//ResultDataSet, along with the DataTable!  
//NOTE: You must reference the "DataTableToUse" property here as the Nreport  
//      class has already made some changes to accomodate data column  
//headings (see documentation for more on this)  
reportViewer1.LocalReport.DataSources.Add(  
    new Microsoft.Reporting.WinForms.ReportDataSource  
        ("ResultDataSet", reportSettings.DataTableToUse)  
);  
  
//Load the report from the Report Builder  
reportViewer1.LocalReport.LoadReportDefinition(  
    report.GetRdlcStream2008Schema());  
}  
catch (Exception _e)  
{  
    //Simple error handling  
    MessageBox.Show("The error is: " + _e.Message);  
}  
  
//View Report in viewer  
this.reportViewer1.RefreshReport();  
this.reportViewer1.SetDisplayMode(  
    Microsoft.Reporting.WinForms.DisplayMode.PrintLayout);  
}
```

Visual Basic

```
Private Sub Form1_Load(ByVal sender As System.Object, ByVal e As System.EventArgs)  
Handles MyBase.Load
```

```
'Declare a DataTable  
Dim dt As DataTable
```

```
'A valid connection string to your database  
Dim connectionString As String  
connectionString = "Data Source=(local);Initial Catalog=Nreports;" +  
    "Integrated Security=True"
```

```
'Use your own application functions to populate the DataTable with data  
'Below is just a quick method to connect to a database and run a query  
'against it  
'The Nreport Dll only needs data in a DataTable - irrespective of how  
'you populate it!
```

```
Dim dbConn As SqlConnection = New SqlConnection()  
dbConn.ConnectionString = connectionString  
dbConn.Open()
```

```
Dim command As SqlCommand = New SqlCommand("SELECT * FROM Users", dbConn)  
Dim adapter As SqlDataAdapter = New SqlDataAdapter(command)  
Dim ds As DataSet = New DataSet
```

```
Try  
    adapter.Fill(ds)  
Catch ex As Exception
```

```
End Try  
adapter.Dispose()  
command.Dispose()  
dbConn.Close()  
dbConn.Dispose()
```

```
'Here we actually get the DataSet
dt = ds.Tables(0)
```

Try

```
'Create a new Report Definition (this allows you to change report properties)
Dim reportSettings As Nreport.ReportDefinition = New
    Nreport.ReportDefinition()
'Set the Data Table To Use
reportSettings.DataTableToUse = dt
'Create a new ReportBuilder and pass the settings object
Dim report As Nreport.ReportBuilder = New
    Nreport.ReportBuilder(reportSettings)
```

```
'!!! Important !!!
'Very important to add a DataSource to your "local report" called
'ResultDataSet, along with the DataTabel!
'NOTE: You must reference the "DataTableToUse" property here as the
'Nreport
'
'    class has already made some changes to accomodate data column
'headings (see documentation for more on this)
ReportViewer1.LocalReport.DataSources.Add(
    New Microsoft.Reporting.WinForms.ReportDataSource(
        "ResultDataSet", reportSettings.DataTableToUse)
    )
```

```
'Load the report from the Report Builder
```

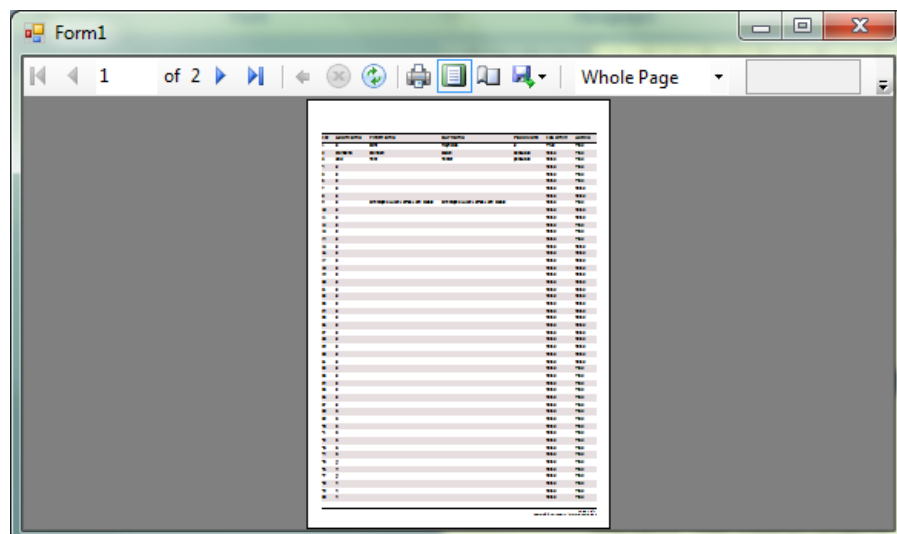
```
ReportViewer1.LocalReport.LoadReportDefinition(report.GetRdlcStream2008Schema())
```

```
Catch ex As Exception
'Simple error handling
MessageBox.Show("The error is: " + ex.Message)
End Try
```

```
'View Report in viewer
Me.ReportViewer1.RefreshReport()
Me.ReportViewer1.SetDisplayMode(
    Microsoft.Reporting.WinForms.DisplayMode.PrintLayout)
```

End Sub

The above code produces your first simple report



Delving further

Once you have managed to produce your first report you can now start setting the various properties available. (Note the section on properties per versions)

Expanding on the above example, let's add some titles and footers to the report:-

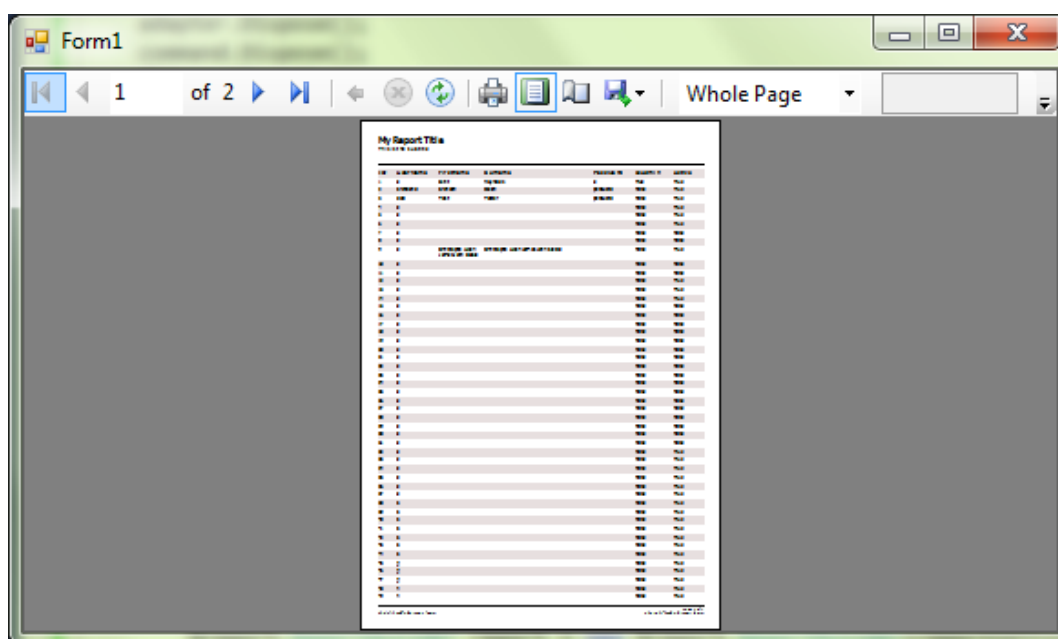
C#

```
//Create a new Report Definition (this allows you to change report properties)
Nreport.ReportDefinition reportSettings = new Nreport.ReportDefinition();
//Set the Data Table To Use
reportSettings.DataTableToUse = dt;
//Set further properties
reportSettings.Title = "My Report Title";
reportSettings.SubTitle = "This is the sub title";
reportSettings.ReportFooterLeft = "The left hand-side report footer";
reportSettings.ReportFooterRight = "The right-hand side report footer";
```

Visual Basic

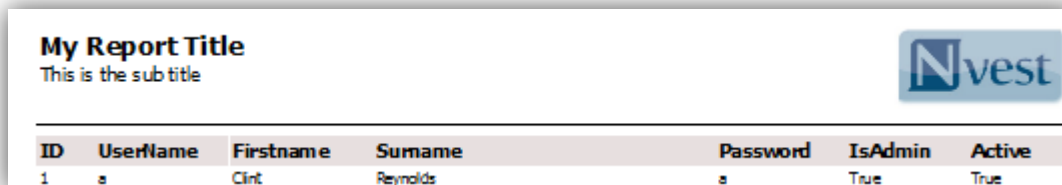
```
'Create a new Report Definition (this allows you to change report properties)
Dim reportSettings As Nreport.ReportDefinition = New
    Nreport.ReportDefinition()
'Set the Data Table To Use
reportSettings.DataTableToUse = dt
'Set further properties
reportSettings.Title = "My Report Title"
reportSettings.SubTitle = "This is the sub title"
reportSettings.ReportFooterLeft = "The left hand-side report footer"
reportSettings.ReportFooterRight = "The right-hand side report footer"
```

Running your report now will produce the following:-



Adding a Logo

The report logo will appear in the top right-hand side and the [ReportDefinition](#) requires a logo as a byte array.



Create a byte array from an image stored in a SQL table

Create a new byte array

```
C#  
byte[] _blob = new byte[0];
```

Select a field from your database that contains the image data – in SQL Server this would be an IMAGE field.

Cast the data field to a byte array

```
C#  
_blob = (byte[])_ds.Tables[0].Rows[0]["Logo"];
```

Create a byte array from an image on disk

Provided you have a valid File Name, you could use the example below

```
C#  
/// <summary>  
/// Get a byte array from an image on the file system  
/// </summary>  
/// <param name="FileName">A valid File Name of a valid image!</param>  
/// <returns></returns>  
public byte[] ImageOnFileToByte(string FileName)  
{  
    byte[] _blob = new byte[0];  
  
    //Get the Blob of the file  
    try  
    {  
        FileStream _fls;  
        _fls = new FileStream(FileName, FileMode.Open, FileAccess.Read);  
        //a byte array to read the file  
        _blob = new byte[_fls.Length];  
        _fls.Read(_blob, 0, System.Convert.ToInt32(_blob.Length));  
        _fls.Close();  
    }  
}
```

```
    catch
    { }

    return _blob;
}
```

Create a byte array from an image embedded in your application

Using a URI string such as "pack://application:,,,/Nreports;component/Images/nvestLogo.jpg" one could use a function as the example below

```
C#
/// <summary>
/// Get a byte array from an image embedded in the application
/// </summary>
/// <param name="URI">The full URI string to a valid embedded image</param>
/// <returns></returns>
public byte[] ImageInResourceToByte(string URI)
{
    byte[] _blob = new byte[0];

    //Get the Blob of the file
    try
    {
        StreamResourceInfo sri = App.GetResourceStream(new Uri(URI));

        _blob = new byte[sri.Stream.Length];
        sri.Stream.Read(_blob, 0, System.Convert.ToInt32(_blob.Length));
    }
    catch { }

    return _blob;
}
```

Functionality derived from Column Heading naming conventions

The Nreport engine has added functionality derived from the column headings supplied in the DataTable. The easiest way to set the column headings would be to use aliases within your SQL statement populating the data. Eg.

```
SELECT FirstName AS [grp!First Name] FROM dbo.Users
```

The square brackets are essential! One can also use an alias to create spaces in your column headers eg `SELECT FirstName AS [My First Name] FROM dbo.Users`

The DataSet column headings can also be set programmatically.

The Nreport engine works with the following prefixes appended to the column headings (consult table on versions available and functionality)

Prefix	Functionality
[grp!Field Name]	Add a Grouping on the report
[grnp!Field Name]	Add a Grouping that starts on a new page

[sum!Field Name]	Sum the values (per group, with grand totals)
[cnt!Field Name]	Count the records (per group, with grand totals)

The following SQL statement will produce a report with Groupings. Please note that multiple groupings are catered for, and the groupings will occur in the order of the [grp! Fields

The following simple SQL statement will produce a report with two levels of groupings:-
`SELECT UserName AS [grp!User], Project AS [grp!The Project], Resources FROM Projects`

Resources
User: Bob
The Project: Project 1
Hammer
Pliers
The Project: Project 3
Measuring Tape
User: Clint
The Project: Project 1
Nails
User: John
The Project: Project 2
Hammer
Safety Hat
The Project: Project 3
Screwdriver

Common Errors

Here follows some common errors that could be made.

No DataSet

If you try creating a report with no dataset, you will most certainly see an error!

```
C#
//Create a new Report Definition (this allows you to change report properties)
Nreport.ReportDefinition reportSettings = new Nreport.ReportDefinition();
//Set the Data Table To Use
reportSettings.DataTableToUse = dt;
```

DataSet name not specified

When you create a local report dataset (on the Report Viewer), you must specify the name "ResultDataSet", and you need to pass the ReportDefinition's DataTable

C#

```
////!! Important !!!  
//Very important to add a DataSource to your "local report" called  
//ResultDataSet, along with the DataTable!  
//NOTE: You must reference the "DataTableToUse" property here as the Nreport  
//      class has already made some changes to accomodate data column  
//headings (see documentation for more on this)  
reportViewer1.LocalReport.DataSources.Add(  
    new Microsoft.Reporting.WinForms.ReportDataSource  
        ("ResultDataSet", reportSettings.DataTableToUse)  
);
```

All columns are "Group By" columns

You need to specify at least one "detail" column (ie it is not a grouping field)

Some columns are displaying too small

You are probably trying to display too much info on your report! Reports which are more than 1 page wide are not easy to read.

Other Reporting Products

Consult www.nvest.co.za to find information the generic Nreports reporting application. This allows administrators to set up standard reports within a treeview structure, and to dynamically select parameters specific to each report.

Licence

This software may be freely distributed as part of your application installations and does not contain any restrictive licensing modules.

The "free download" version (with limited features) may be freely distributed to anyone, but may not be sold by any means or manner, nor may it be altered in any form.

The "paid for" fully featured DLL may however not be copied to others who intend using it as part of their software development. This software is strictly sold "per developer" and may be used by a single user on multiple machines. Each developer is required to purchase their own licence.

Disclaimer

THIS SOFTWARE IS PROVIDED BY NVEST DEVELOPMENT SOLUTIONS (PTY) LTD "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT HOLDER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.