

**SimpleQuery
V1.2
For
DotNetNuke 5.1+**

By
Paul Scarlett

TressleWorks.ca

Document Version 1.2
January 2010

Table of Contents

Introduction	1
Usage.....	2
Database	2
Template	4
No Data Template.....	8
Search.....	8
Examples	10
Exmample 1 - DesktopModules	10
Example 2 - Products with Column Sorting	12
Revision History.....	14
Support	14

Introduction

SimplyQuery is, as the name suggests, a simple module to display SQL Select data. The module uses a user supplied template to display the data. Optionally, if no template is supplied, SimpleQuery will generate a default template that then can be update by the user.

Use SimpleQuery when the SQL Select is static and the data returned is limited in volume as there is no paging support, or any column sorting. (Both of these issues can be address via JQuery)

Caution:

SimpleQuery module has **complete access** to the current database tables (all portals) so Site Administrators should be aware that cross portal data "leakage" is possible.

Usage

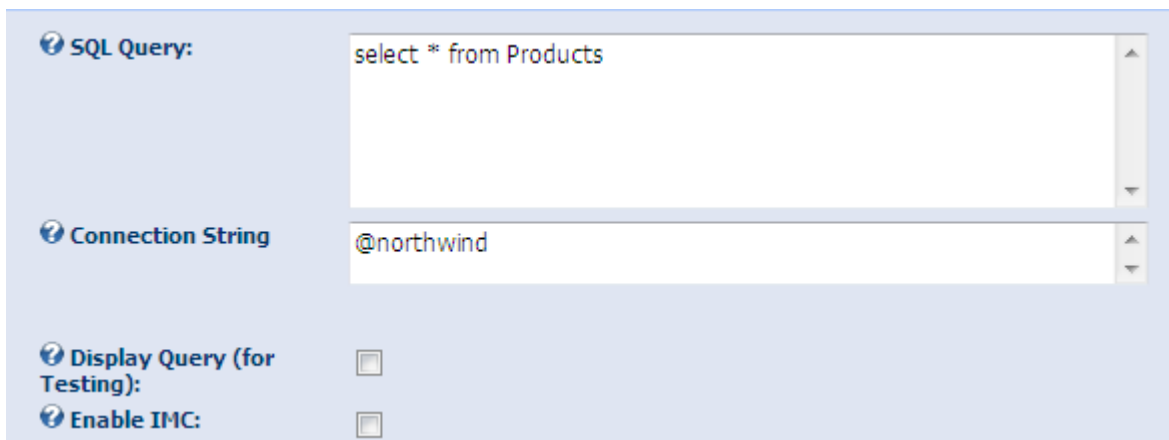
SimpleQuery requires only the SQL Select. In such a case, the module will assume the select is being performed against the local DotNetNuke database, and will also generate a default template.

However, the user can provide a specific Database connection string, and define the template to be used.

Additionally, SimpleQuery will accept Search Words and Description.

Database

SimpleQuery requires a SQL Select command for the SQL Query. This query must return a valid data table.



The screenshot shows a configuration panel with a light blue background. It contains three main sections:

- SQL Query:** A text input field containing the SQL command `select * from Products`.
- Connection String:** A text input field containing the value `@northwind`.
- Display Query (for Testing):** A checkbox that is currently unchecked.
- Enable IMC:** A checkbox that is currently unchecked.

An optional connection string is available to allow connecting to another database. If the connection string is empty, then the current database is used. The connection string will also accept a AppSetting key that contains a valid connection string. For example, `@NorthWind` would define a AppSetting key of `Northwind`. AppSetting are located in the `Web.Config` file for your site. This feature is used to hide or share implementation information details.

Several special Meta-strings will be replaced with the current value PRIOR to execution of the SQL command:

<code>[DNN:PortalID]</code>	is replaced with the current portalID (see example above)
<code>[DNN:ModuleID]</code>	is replaced with the current ModuleID

[DNN:UserID] is replaced with the current UserID
[DNN:TabID] is replaced with the current TabID

[DNN:PortalPath] is replace with the current directory path of the portal
[DNN:ModulePath] is replace with the current directory path of the module

[DNN:Date:<format>] is replace with the current date based on optionally supplied format. Standard .Net Date format string is expected.
[DNN:Time:<format>] is replace with the current time based on optionally supplied format. Standard .Net Time format string is expected

Optionally, SimpleQuery can support Inter-Module Communication. SimpleQuery understands the parameters passed from SQLGridSelectedView (another Tressleworks module). Please refer to SQLGridSelectedView User's Guide for details on how to use this feature.

SimpleQuery also supports the use parameter substitution based on Query string parameters. The Querystring substitution parameter format is:

[Query:<key>:<defaultvalue>]

Where <key> is the Query string key name and <defaultvalue> is the value to be used if the key is not found in the Query String. For example, assume there is a URL like

<http://www.mysite.com/.../default.aspx?OrderID=12345>

and a SQL Query of

```
Select * from Orders where OrderID = [Query:OrderID:0]
```

The resulting executed SQL Command would be

```
Select * from Orders where OrderID = 12345
```

Template

<input type="checkbox"/> Reset Template:	<input type="checkbox"/>
<input checked="" type="checkbox"/> Module Specific Styles:	<input type="checkbox"/>
Prefix:	<pre><style type="text/css"> .SQ_Table { color:black; background-color:white; } .SQ_Header th { color:white; background-color:navy;} .SQ_Odd td { background-color:white; } .SQ_Even td { background-color:lightgrey; }</pre>
Row Template:	<pre><tr class="SQ_Odd"> <td class="SQ_Col_ProductID">[Field:ProductID]</td> <td class="SQ_Col_ProductName">[Field:ProductName]</td> <td class="SQ_Col_SupplierID">[Field:SupplierID]</td> <td class="SQ_Col_CategoryID">[Field:CategoryID]</td> <td class="SQ_Col_QuantityPerUnit">[Field:QuantityPerUnit]</td> <td class="SQ_Col_UnitPrice">[Field:UnitPrice]</td> <td class="SQ_Col_UnitsInStock">[Field:UnitsInStock]</td> <td class="SQ_Col_UnitsOnOrder">[Field:UnitsOnOrder]</td></pre>
Alternate Row Template:	<pre><tr class="SQ_Even"> <td class="SQ_Col_ProductID">[Field:ProductID]</td> <td class="SQ_Col_ProductName">[Field:ProductName]</td> <td class="SQ_Col_SupplierID">[Field:SupplierID]</td> <td class="SQ_Col_CategoryID">[Field:CategoryID]</td> <td class="SQ_Col_QuantityPerUnit">[Field:QuantityPerUnit]</td> <td class="SQ_Col_UnitPrice">[Field:UnitPrice]</td> <td class="SQ_Col_UnitsInStock">[Field:UnitsInStock]</td> <td class="SQ_Col_UnitsOnOrder">[Field:UnitsOnOrder]</td></pre>
Suffix:	<pre></tbody> </table></pre>

SimpleQuery uses a 4-part template based on a HTML table. The Prefix contains all the required HTML codes to define the table to be displayed. The Row Template defines what will be displayed on each row. If present, the Alternate Row Template will be displayed on alternating rows. Finally the Suffix contains closing Table related HTML.

If the All of the Template textboxes are empty, then SimpleQuery will generate a default template. Alternatively, the user can request the default be generated by checking the [Reset Template](#) option

Additionally, the user can ask for [Module Specific Style](#) and when enabled all style related entries in the template will have the **[DNN:ModuleID]** added, so that the style classes are unique.

The following is the default template created for the Northwind Products table:

```
<style type="text/css">
  .SQ_Table { color:black; background-color:white; width:100% }
  .SQ_Header th { color:white; background-color:navy;}
  .SQ_Odd    td { background-color:white; }
  .SQ_Even   td { background-color:lightgrey; }

  .SQ_Col_ProductID { text-align:center; }
  .SQ_Col_ProductName { text-align:center; }
  .SQ_Col_SupplierID { text-align:center; }
  .SQ_Col_CategoryID { text-align:center; }
  .SQ_Col_QuantityPerUnit { text-align:center; }
  .SQ_Col_UnitPrice { text-align:center; }
  .SQ_Col_UnitsInStock { text-align:center; }
  .SQ_Col_UnitsOnOrder { text-align:center; }
  .SQ_Col_ReorderLevel { text-align:center; }
  .SQ_Col_Discontinued { text-align:center; }
</style>
<table id="SQ_Table" class="SQ_Table">
<thead class="SQ_Header">
  <tr>
    <th>Product ID</th>
    <th>Product Name</th>
    <th>Supplier ID</th>
    <th>Category ID</th>
    <th>Quantity Per Unit</th>
    <th>Unit Price</th>
    <th>Units In Stock</th>
    <th>Units On Order</th>
    <th>Reorder Level</th>
    <th>Discontinued</th>
  </tr>
</thead>
<tbody>
```

The generated template contains a style section that defines class for the table, header, and even and odd rows. Also generated are class for each column.

The table is defined along with a header row with the column names as defined in the database select.

The following are row and alternate row templates for the products table:

```
<tr class="SQ_Odd">
  <td class="SQ_Col_ProductID">[Field:ProductID]</td>
  <td class="SQ_Col_ProductName">[Field:ProductName]</td>
  <td class="SQ_Col_SupplierID">[Field:SupplierID]</td>
  <td class="SQ_Col_CategoryID">[Field:CategoryID]</td>
  <td class="SQ_Col_QuantityPerUnit">[Field:QuantityPerUnit]</td>
  <td class="SQ_Col_UnitPrice">[Field:UnitPrice]</td>
  <td class="SQ_Col_UnitsInStock">[Field:UnitsInStock]</td>
  <td class="SQ_Col_UnitsOnOrder">[Field:UnitsOnOrder]</td>
  <td class="SQ_Col_ReorderLevel">[Field:ReorderLevel]</td>
  <td class="SQ_Col_Discontinued">[Field:Discontinued]</td>
</tr>
```

Notice that each column references the column class defined in the prefix and the actual contents of the row's column is defined by the substitution parameter **[Field:<columnname>]** where <columnname> is the corresponding column name. For example: [Field:ProductID]

```
<tr class="SQ_Even">
  <td class="SQ_Col_ProductID">[Field:ProductID]</td>
  <td class="SQ_Col_ProductName">[Field:ProductName]</td>
  <td class="SQ_Col_SupplierID">[Field:SupplierID]</td>
  <td class="SQ_Col_CategoryID">[Field:CategoryID]</td>
  <td class="SQ_Col_QuantityPerUnit">[Field:QuantityPerUnit]</td>
  <td class="SQ_Col_UnitPrice">[Field:UnitPrice]</td>
  <td class="SQ_Col_UnitsInStock">[Field:UnitsInStock]</td>
  <td class="SQ_Col_UnitsOnOrder">[Field:UnitsOnOrder]</td>
  <td class="SQ_Col_ReorderLevel">[Field:ReorderLevel]</td>
  <td class="SQ_Col_Discontinued">[Field:Discontinued]</td>
</tr>
```

Finally the suffix that closed the table body and table

```
</tbody>
</table>
<p class="SubHead">
As of [DNN:Date:dd-MMM-yyyy] [DNN:Time:HH:mm]
</p>
```

The field substitution has several formats

[Field:<columnname>]

The resulting substitution is the value of the column name supplied.

[Field:<type>:<column>]

Where

<type> = COLUMNNAME, NAME, N, COLUMNVALUE, VALUE , V

<column> = column name or ordinal number of column

The resulting substitution will be either the column's name or value

[Field:<type>:<column>:<fieldtype>]

Where

<type> = COLUMNNAME, NAME, N, COLUMNVALUE, VALUE , V

<column> = column name or ordinal number of column

<fieldtype> = INT, INTEGER, DATE, SINGLE, REAL, BOOLEAN

The resulting substitution will be either the column's name or value and formatted based on default format for the field type.

[Field:<type>:<column>:<fieldtype>:<format>]

Where

<type> = COLUMNNAME, NAME, N, COLUMNVALUE, VALUE , V

<column> = column name or ordinal number of column

<fieldtype> = INT, INTEGER, DATE, SINGLE, REAL, BOOLEAN

<format> = {0:<formatstring>} where format string is a .NET format string.

The resulting substitution will be either the column's name or value and formatted based on supplied format. For example:

```
[Field:Value:OrderDate:Date:{0:dd-MMM-yyy HH:mm}]
```

Will display the Order date like 10-Jan-2010 14:21

```
[Field:Value:UnitCost:Real:{0:c}]
```

Will display the unit cost of 15.350 like \$15.35

Two additional Substitution parameters are available :

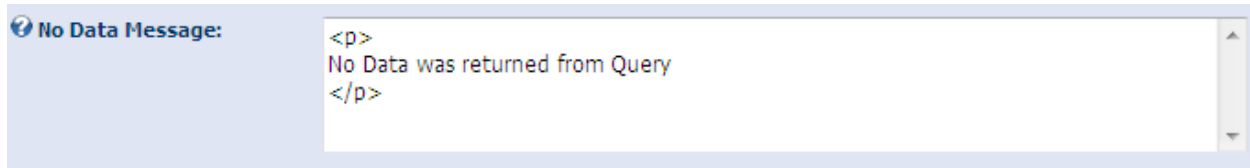
[Field:SQ_RecordNumber]

which returns the row number

[Field:SQ_RecordCount]

which returns the total number of row returned from the query

No Data Template

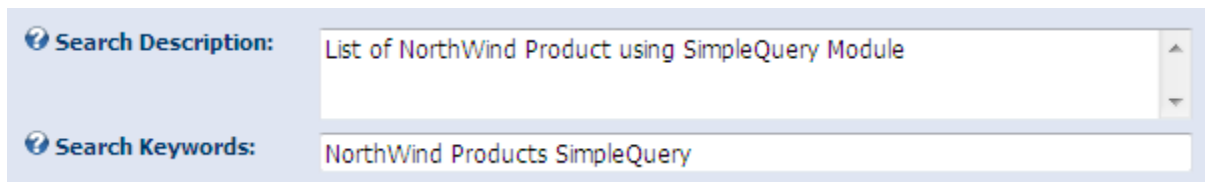


In the event that the query returns no rows, and the No Data Message Template has been defined, the No Data Message Template will be used.

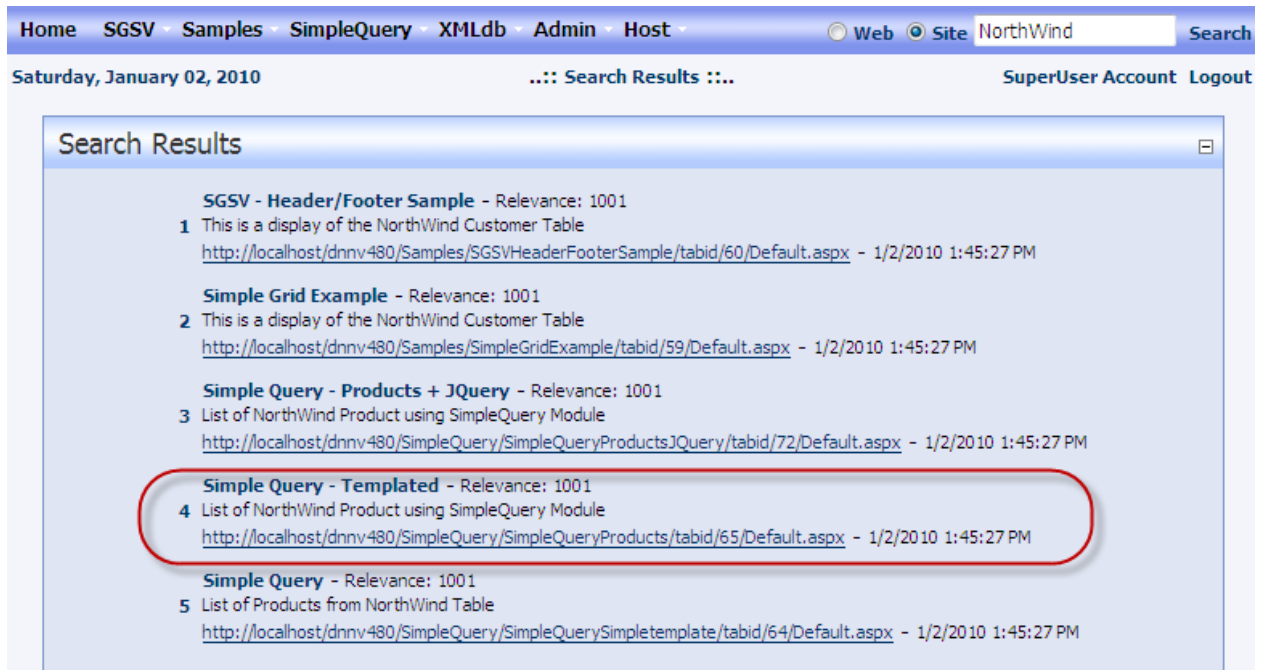
The Template must include all required HTML to display the message as no other template will be included in the display.

Search

SimpleQuery is a Searchable module which means whenever the search engine request search information, the module will respond with a Search Description and Search Keywords to be associated with the module.



In the above example, the keyword "NorthWind, Products, and SimpleQuery" will be associated with this module along with the description.



Home SGSV Samples SimpleQuery XMLdb Admin Host Web Site NorthWind Search

Saturday, January 02, 2010 ...:: Search Results ::.. SuperUser Account Logout

Search Results

- SGSV - Header/Footer Sample** - Relevance: 1001
1 This is a display of the NorthWind Customer Table
<http://localhost/dnnv480/Samples/SGSVHeaderFooterSample/tabid/60/Default.aspx> - 1/2/2010 1:45:27 PM
- Simple Grid Example** - Relevance: 1001
2 This is a display of the NorthWind Customer Table
<http://localhost/dnnv480/Samples/SimpleGridExample/tabid/59/Default.aspx> - 1/2/2010 1:45:27 PM
- Simple Query - Products + JQuery** - Relevance: 1001
3 List of NorthWind Product using SimpleQuery Module
<http://localhost/dnnv480/SimpleQuery/SimpleQueryProductsJQuery/tabid/72/Default.aspx> - 1/2/2010 1:45:27 PM
- Simple Query - Templated** - Relevance: 1001
4 List of NorthWind Product using SimpleQuery Module
<http://localhost/dnnv480/SimpleQuery/SimpleQueryProducts/tabid/65/Default.aspx> - 1/2/2010 1:45:27 PM
- Simple Query** - Relevance: 1001
5 List of Products from NorthWind Table
<http://localhost/dnnv480/SimpleQuery/SimpleQuerySimpletemplate/tabid/64/Default.aspx> - 1/2/2010 1:45:27 PM

The fourth result is the specific page.

Examples

Exmple 1 - DesktopModules

This example is straight forward and is a quick example. Enter the following SQL select into the SQL Query. Leave all other fields blank.

```
Select DesktopModuleID, FriendlyName, Description
from DesktopModules
```

The following will be displayed – actual contents depends on what modules have been installed.

Simple Query - ModuleSettings		
Desktop Module ID	Friendly Name	Description
10	Security Roles	Administrators can manage the security roles defined for their portal. The module allows you to add new security roles, modify existing security roles, delete security roles, and manage the users assigned to security roles.
11	Tabs	Administrators can manage the Tabs within the portal. This module allows you to create a new tab, modify an existing tab, delete tabs, change the tab order, and change the hierarchical tab level.
12	Site Settings	The Site Settings module represents the local options for your portal. Local settings allow you to customize your portal to meet your business requirements.
13	User Accounts	Administrators can manage their registered users. This module allows you to add new users, modify existing users, delete users, and manage the security roles for users.
14	Vendors	Administrators can manage the Vendors and Banners associated to the portal. This module allows you to add a new vendor, modify an existing vendor, and delete a vendor.
15	Banners	Banner advertising is managed through the Vendors module in the Admin tab. You can select the number of banners to display as well as the banner type.
16	File Manager	Administrators can manage the files stored in their upload directory. This module allows you to upload new files, download files, delete files, and synchronize your upload directory. It also provides information on the amount of disk space used and available.
18	Site Log	Administrators can view the details of visitors using their portal. There are a variety of

Reopening the settings will show the default template created:

Reset Template:	<input type="checkbox"/>
Module Specific Styles:	<input type="checkbox"/>
Prefix:	<pre><style type="text/css"> .SQ_Table { color:black; background-color:white; width:100%; } .SQ_Header th { color:white; background-color:navy;} .SQ_Odd td { background-color:white; } .SQ_Even td { background-color:lightgrey; }</pre>
Row Template:	<pre><tr class="SQ_Odd"> <td class="SQ_Col_DesktopModuleID">[Field:DesktopModuleID]</td> <td class="SQ_Col_FriendlyName">[Field:FriendlyName]</td> <td class="SQ_Col_Description">[Field:Description]</td> </tr></pre>
Alternate Row Template:	<pre><tr class="SQ_Even"> <td class="SQ_Col_DesktopModuleID">[Field:DesktopModuleID]</td> <td class="SQ_Col_FriendlyName">[Field:FriendlyName]</td> <td class="SQ_Col_Description">[Field:Description]</td> </tr></pre>
Suffix:	<pre></tbody> </table></pre>

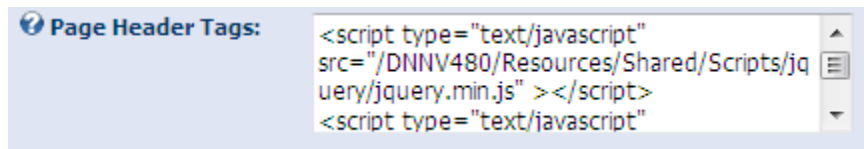
Example 2 - Products with Column Sorting

This example uses the NorthWind Sample database and a JQuery plug-in called TableSorter to implement column-sorting. See <http://tablesorter.com> for details and to download the plug-in.

Also, this example will assume you are using DNN that is NOT aware of JQuery and requires the JQuery Library be installed on the page. To do this, add the following lines (the specific directories may vary on your site – adjust according)

```
<script type="text/javascript"
src="/DotNetNuke/Resources/Shared/Scripts/jquery/jquery.min.js" >
</script>
<script type="text/javascript"
src="/DotNetNuke/js/jquery.tablesorter.min.js" >
</script>
```

To the page header tags Textbox located on the Page Settings where SimpleQuery module will be located – as follows:



For additional help see Joe Brinkman's blog on using JQuery prior to V5.0 of DotNetNuke. (<http://blog.theaccidentalgeek.com/post/2009/05/26/DotNetNuke-Tips-and-Tricks-11-Using-jQuery-in-DotNetNuke.aspx>)

If you are using DNN V5.0+ then, you will only need to add the following to the page header tags:

```
<script type="text/javascript"
src="/DotNetNuke/js/jquery.tablesorter.min.js" >
</script>
```

Or the same can be added to the prefix template text (assume only one module on the page)

Use the following SQL Query to display the products:

```
Select * from Products
```

You will need to define a Connection string to the database containing the NorthWind Sample database.

Next allow the page to be displayed, then reopen the SimpleQuery Settings and add the following javascript to beginning of the Prefix Template:

```
<script type="text/javascript">
jQuery(document).ready(function()
{ jQuery("#SQ_Table").tablesorter(
  { widgets:["zebra"],
    widgetZebra: {css: ["SQ_Odd", "SQ_Even"]}}
  } );
} );
</script>
```

This small piece of Javascript will evoke the TableSorter Plug-in on the SQ_Table and reapply the SQ_Odd and SQ_Even CSS class on the appropriate rows of data after the column has been sorted.

To sort a column. Double-click on the column header to sort ascending, and again to sort descending.

The following is a screen shot of the products table after sorting the Unit Price Column descending

Product ID	Product Name	Supplier ID	Category ID	Quantity Per Unit	Unit Price	Units In Stock	Units On Order	Reorder Level	Discontinued
9	Mishi Kobe Niku	4	6	18 - 500 g pkgs.	97.0000	29	0	0	True
41	Jack's New England Clam Chowder	19	8	12 - 12 oz cans	9.6500	85	0	10	False
45	Rogede sild	21	8	1k pkg.	9.5000	5	70	15	False
47	Zaanse koeken	22	3	10 - 4 oz boxes	9.5000	36	0	0	False
19	Teatime Chocolate Biscuits	8	3	10 boxes x 12 pieces	9.2000	25	0	5	False
23	Tunbröd	9	5	12 - 250 g pkgs.	9.0000	61	0	25	False
20	Sir Rodney's Marmalade	8	3	30 gift boxes	81.0000	40	0	0	False
75	Rhönbräu Klosterbier	12	1	24 - 0.5l bottles	7.7500	125	0	25	False
54	Tourtière	25	6	16 pies	7.4500	21	0	10	False

Revision History

Version 1.0 – 04-Dec-2009

Initial release

Version 1.1 – 02-Jan-2009

Initial public release with enhanced default template along with a DNN v5+ install version.

Version 1.2 – 10-Jan-2009

Addition of No Data Message Template

Addition of additional Field substitution formats

Addition of new substitution parameters for:

Date, Time, RecordNumber and Record Count

Addition of Support or Query String Values.

Revised Settings form to resize Templates fields.

Support

If you have any questions or comments, please send them to
SimpleQuery@tressleworks.ca