EntitySpaces
Web Admin Grids

Table of Contents

Introduction 1

GridLoader Projects 2

Template Usage 5

Tables Tab 6

Settings Tab 6

Browse Tab 7

Details Tab 9

Detail Grids Tab 10

Detail Lookups Tab 11

Search Tab 12

Edit Tab 13

MyGeneration Project Files 14

Conclusion 15

Introduction

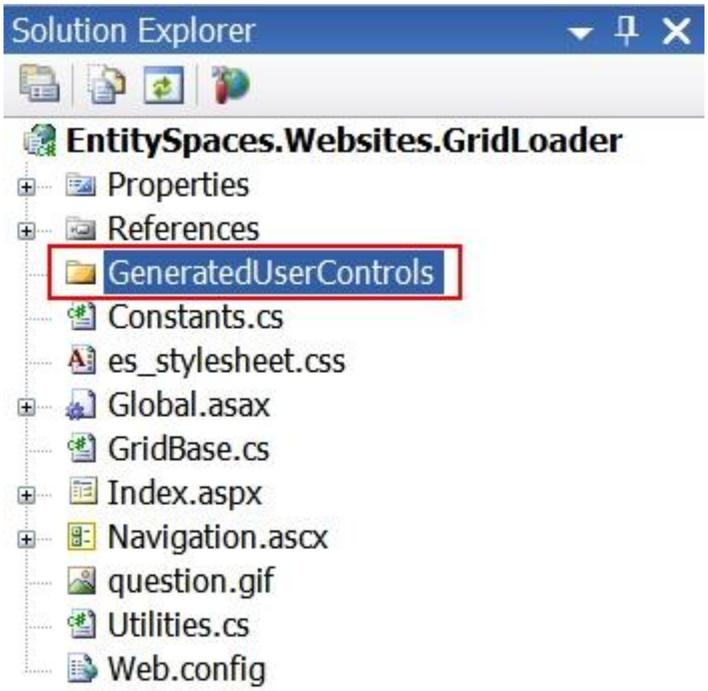
These templates are intended to provide the ability to administer databases table data over the web. These are not UI templates meant to be used to interact with the public, these are for administrative purposes only. This chore is typically overlooked although in many cases it is soon realized that it is needed. Hand coding such admin screens can eat up quite a bit of development time, are not that much fun to code, and are ideal for code generation.

These templates are not currently supported on the same level as the templates that generate the EntitySpaces classes. EntitySpaces will not be bogged down in supporting these or allow them to effect our EntitySpaces 2008 release which is fundamental to our long term strategy.

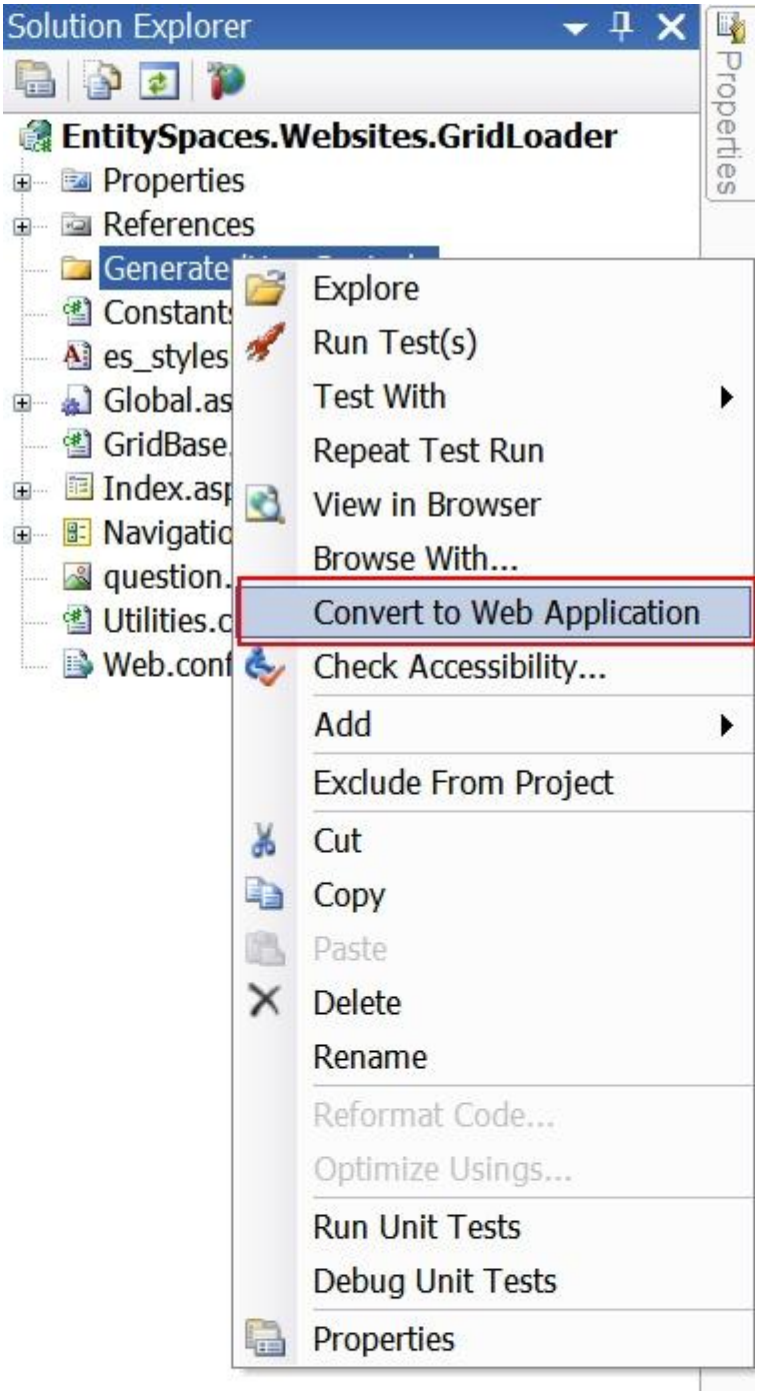
These templates are a retooling of our existing web admin grid templates to facilitate ease of future enhancements, better modularity in the generated admin grid code, and correction of ignored user metadata specified in MyGeneration. The first and last of those changes are pretty self explanatory so we will not cover those anymore. However, the second one is worth spending a little more time on as it will help you understand what is going on under the hood of our web admin grids. Beginning with this release the same physical template will allow you to target the standard ASP.NET platform or via setting (more on this later) the DotNetNuke platform. In order to accomplish this with the same template the generated web admin grid code are created as user controls. We then provide an appropriate GridLoader Web Application Project (WAP) for your environment to dynamically load the user controls for operations.

GridLoader Projects

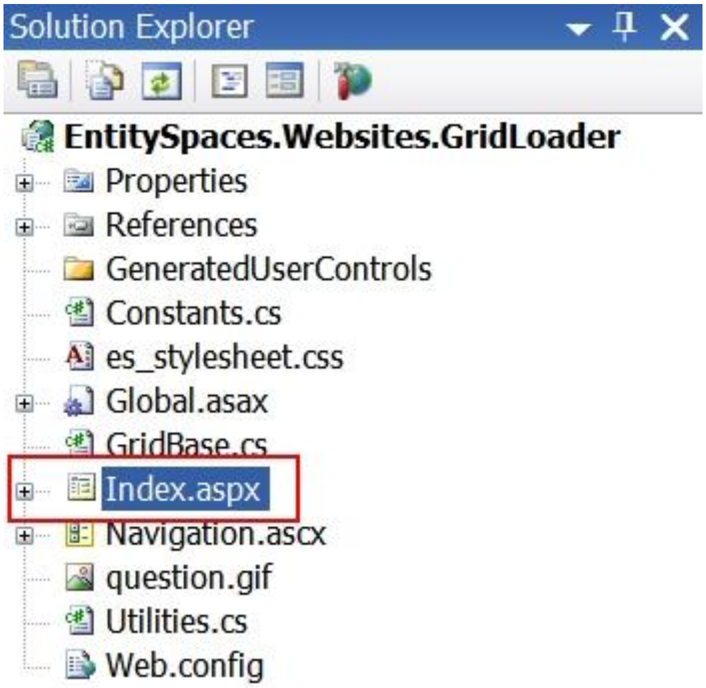
Now that you have a high level overview of what is new and why it was done let's get into the GridLoader application specifics. All of the GridLoader applications operate very much the same way. We provide the GridLoader projects in both C# & VB.net so you can choose the appropriate one for your language, and we also offer them in ASP.NET & DNN formats. Since all the generated code for the web admin grids are user control based now this adds a high degree of modularity for the platform targeted. For the ASP.NET platform you will find a WAP project to hold your generated user controls. For the DNN platform you will find a DNN module which is also a WAP based project to hold your generated user controls. In both cases the generated web admin grid code should be added to the existing folder 'GeneratedUserControls' in your platforms grid loader project.



These templates do not generate .designer files that Visual Studio generates for WAP type projects. In order to have these generated for you after you have added your generated user controls right click on the 'GeneratedUserControls' folder and select 'Convert To Web Application'.



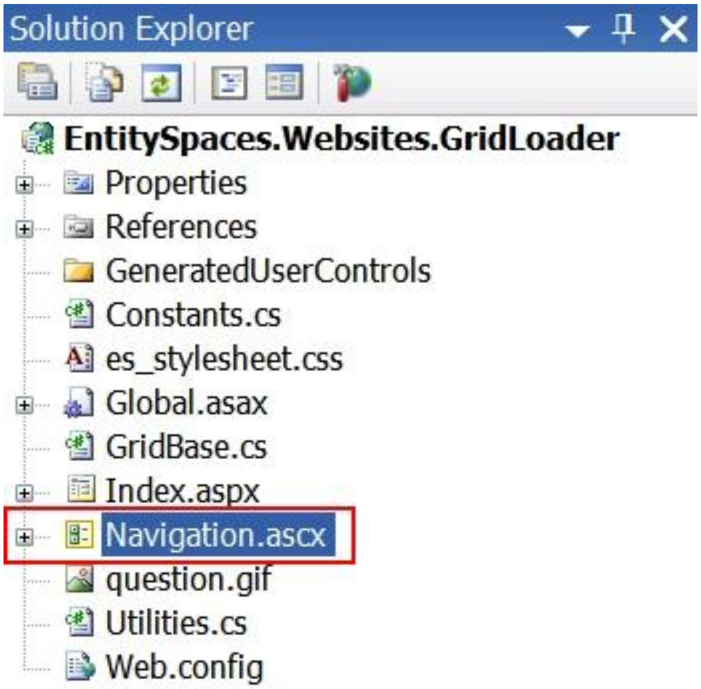
This will generate all the necessary .designer files for WAP compilation. This is a very important step and your project will not compile until you complete it. This is a side effect of Microsoft's decision to not include WAP style projects for Visual Studio 2005 which they later reversed and provided with Service Pack 1. Within your GridLoader project you will find a page called 'Index.aspx' for ASP.NET or 'Index.ascx' for DNN.



Solution Explorer

EntitySpaces.Websites.GridLoader
Properties
References
GeneratedUserControls
Constants.cs
es_stylesheet.css
Global.asax
GridBase.cs
Index.aspx
Navigation.ascx
question.gif
Utilities.cs
Web.config

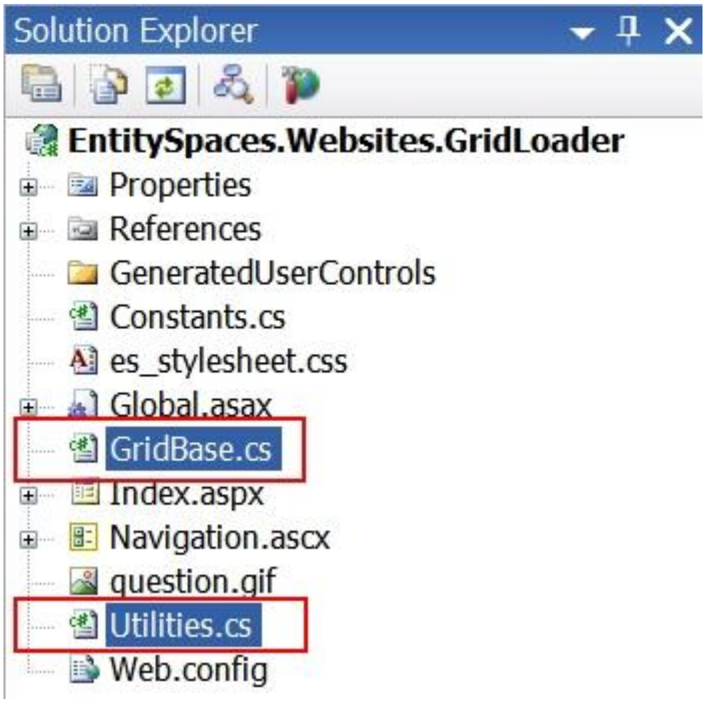
This page is responsible for dynamically loading your generated user controls from the 'GeneratedUserControls' folder when requested, also registered within this page you will find another user control called 'Navigation.ascx'.



Solution Explorer

EntitySpaces.Websites.GridLoader
Properties
References
GeneratedUserControls
Constants.cs
es_stylesheet.css
Global.asax
GridBase.cs
Index.aspx
Navigation.ascx
question.gif
Utilities.cs
Web.config

This user control iterates through all the user controls found in the 'GeneratedUserControls' folder and creates a navigation strip above your admin grids to allow you to easily navigate them all. This is the basic operation of the GridLoader project. To serve up your generated user controls and allow you to easily navigate between them. You will also find a couple of additional classes within the GridLoader projects named GridBase and Utilities.



EntitySpaces.Websites.GridLoader
Properties
References
GeneratedUserControls
Constants.cs
es_stylesheet.css
Global.asax
GridBase.cs
Index.aspx
Navigation.ascx
question.gif
Utilities.cs
Web.config

If you have changed the namespace of your admin grids/business objects from the default 'BusinessObjects' then you must change GridBase's namespace to match. The Utilites class is responsible for URL construction and need not be modified. Now that we have covered how the provided GridLoader projects enable loading of your generated user controls we will take a look at how you go about generating your user controls for your platform of choice.

Template Usage

We will now walk through each tab of the template and explain that tab does and the various options associated. You will find the template located at:

'C:\Program Files\MyGeneration\Templates\EntitySpaces\C#\Web\EntitySpaces_C#_WebAdmin.csgen'

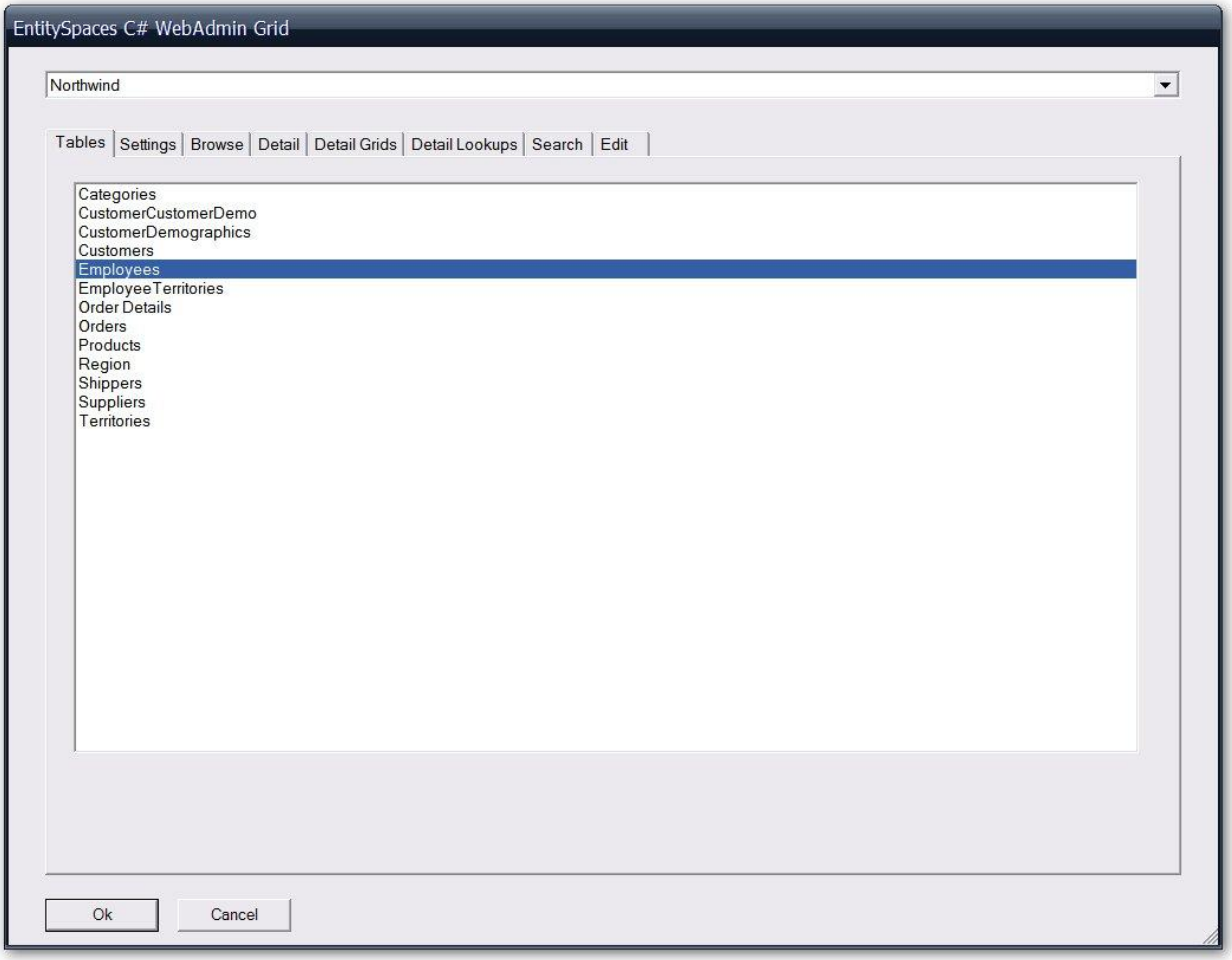
Or

C:\Program Files\MyGeneration\Templates\EntitySpaces\VB\WebEntitySpaces_C#_WebAdmin.csgen'

Assuming you accepted the default install location. Within MyGeneration the title of the template will be 'EntitySpaces C# WebAdmin Grid' or 'EntitySpaces C# WebAdmin Grid' respective to your language choice. Right click on your template choice and hit 'Execute' and we are ready to begin inputting our options for admin grid user control creation.

Tables Tab

The first step is to select the desired table. This template has a multi-tab user interface that requires quite a few settings per table. Because of this you can only generate the code for a single table at a time.

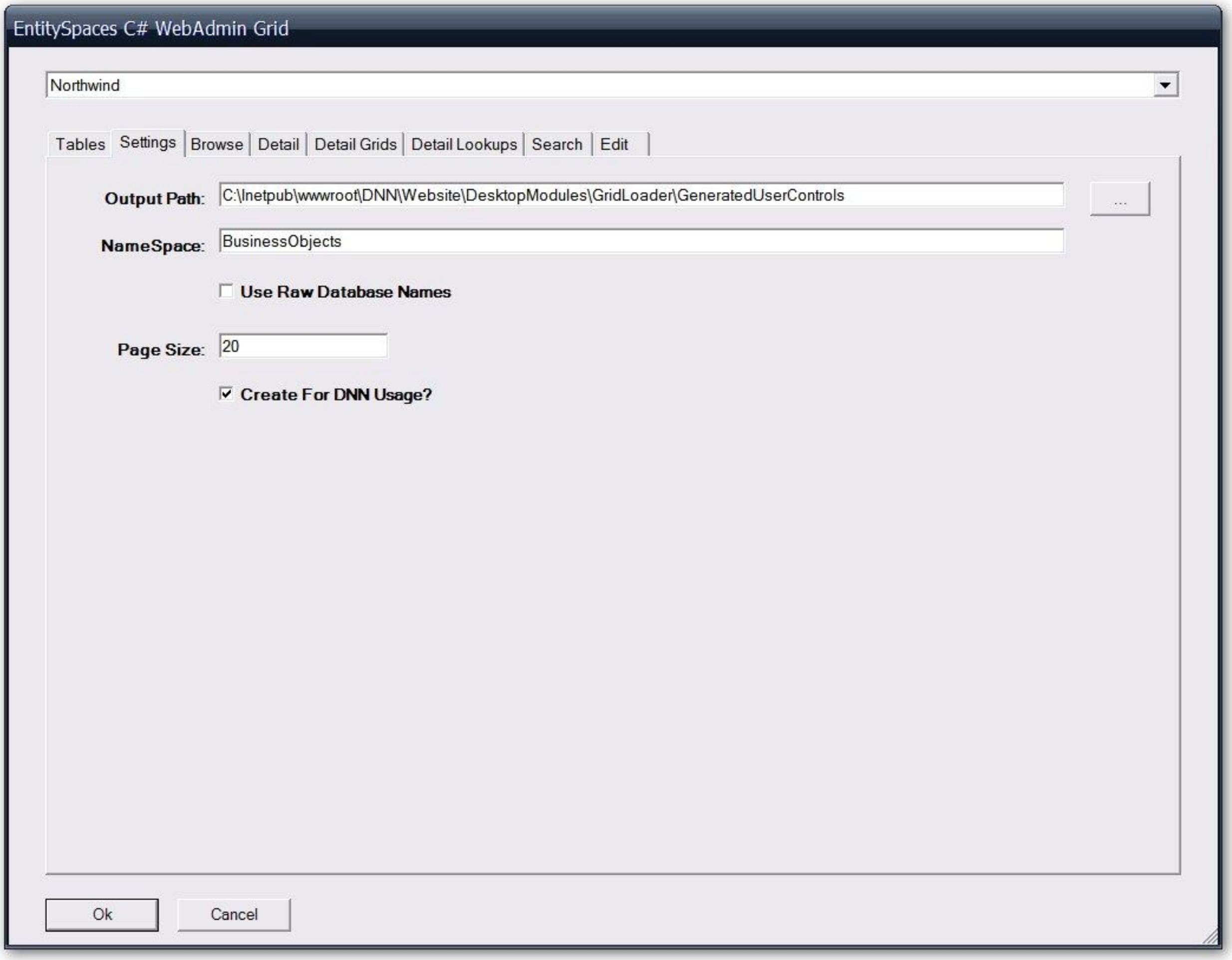


Northwind

Categories
CustomerCustomerDemo
CustomerDemographics
Customers
Employees
EmployeeTerritories
Order Details
Orders
Products
Region
Shippers
Suppliers
Territories

Settings Tab

The Settings tab allows you to control various aspects of the code generation process.



Output Path: C:\Inetpub\wwwroot\DNN\Website\DesktopModules\GridLoader\GeneratedUserControls

NameSpace: BusinessObjects

□ Use Raw Database Names

Page Size: 20

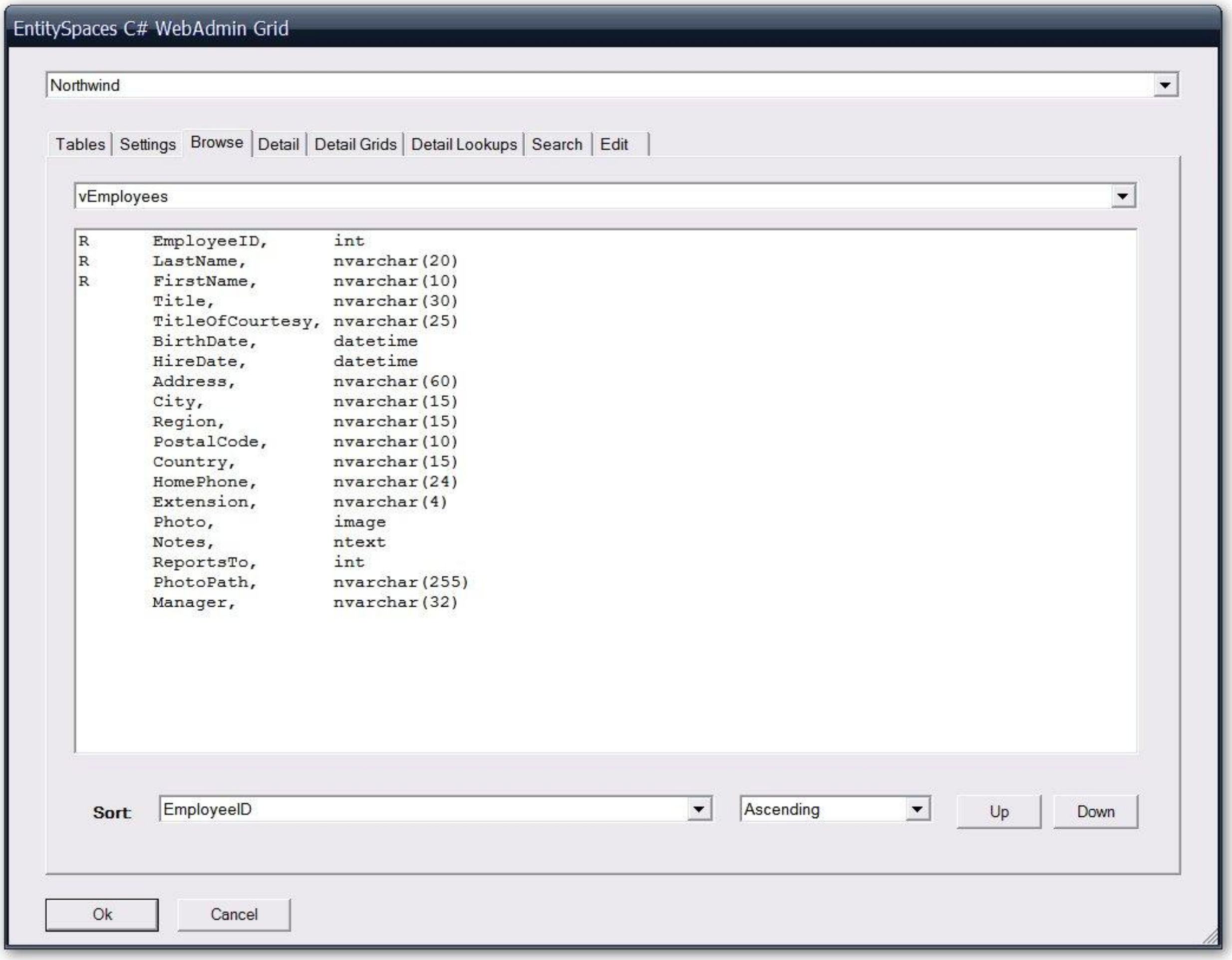
☑ Create For DNN Usage?

Miscellaneous Settings

- Output Path – This determines where the generated .ASPX and .CS files will be saved on disk.
- NameSpace – This should match the namespace of your EntitySpaces classes.
- Use Raw Database Name (Deprecated) – We highly recommend you not use this.
- PageSize – This controls the number of rows in the GridView for the browse mode, the sub grids are hard-coded to 10 in the template itself (easily changed.)
- Create For DNN Usage – This should be checked if you want to use your generated user controls on the DNN platform.

Browse Tab

The Browse tab is the main grid for the table selected in the Tables tab. On this tab you can choose which columns you want to be displayed in the browse grid. You can determine the order the columns are shown in the grid by using the up/down buttons. You can also set the default sort column and order. (All columns in the generated grid can be sorted by the user at runtime by clicking the Column Header.)



Northwind

vEmployees

```
R       EmployeeID,       int
R       LastName,         nvarchar(20)
R       FirstName,        nvarchar(10)
        Title,            nvarchar(30)
        TitleOfCourtesy,  nvarchar(25)
        BirthDate,        datetime
        HireDate,         datetime
        Address,          nvarchar(60)
        City,             nvarchar(15)
        Region,           nvarchar(15)
        PostalCode,       nvarchar(10)
        Country,          nvarchar(15)
        HomePhone,        nvarchar(24)
        Extension,        nvarchar(4)
        Photo,            image
        Notes,            ntext
        ReportsTo,        int
        PhotoPath,        nvarchar(255)
        Manager,          nvarchar(32)
```

Sort: EmployeeID Ascending

Notice the combo-box above that has 'vEmployees' selected. This is a view that was created so that we could show the actual Manager represented by the ReportsTo column, which is a self referencing key back to the Employees table.

CREATE VIEW [dbo].[vEmployees]

AS

SELECT Employees.*, Superior.LastName + ', ' + Superior.FirstName as [Manager]

FROM Employees

LEFT OUTER JOIN Employees Superior

ON Superior.EmployeeID = Employees.ReportsTo

GO

If you look at the Empoyees_Admin.ascx page after it is generated, you can see how these columns are displayed. This is a great way to present a very readable representation of your data model. Remember, you will need to have generated the necessary EntitySpaces classes for the vEmployees view.

There are also special codes to help you when laying out your web pages. Notice the “R” in the grid above, which indicates that the column is required. Here are the special codes used.

R = Required

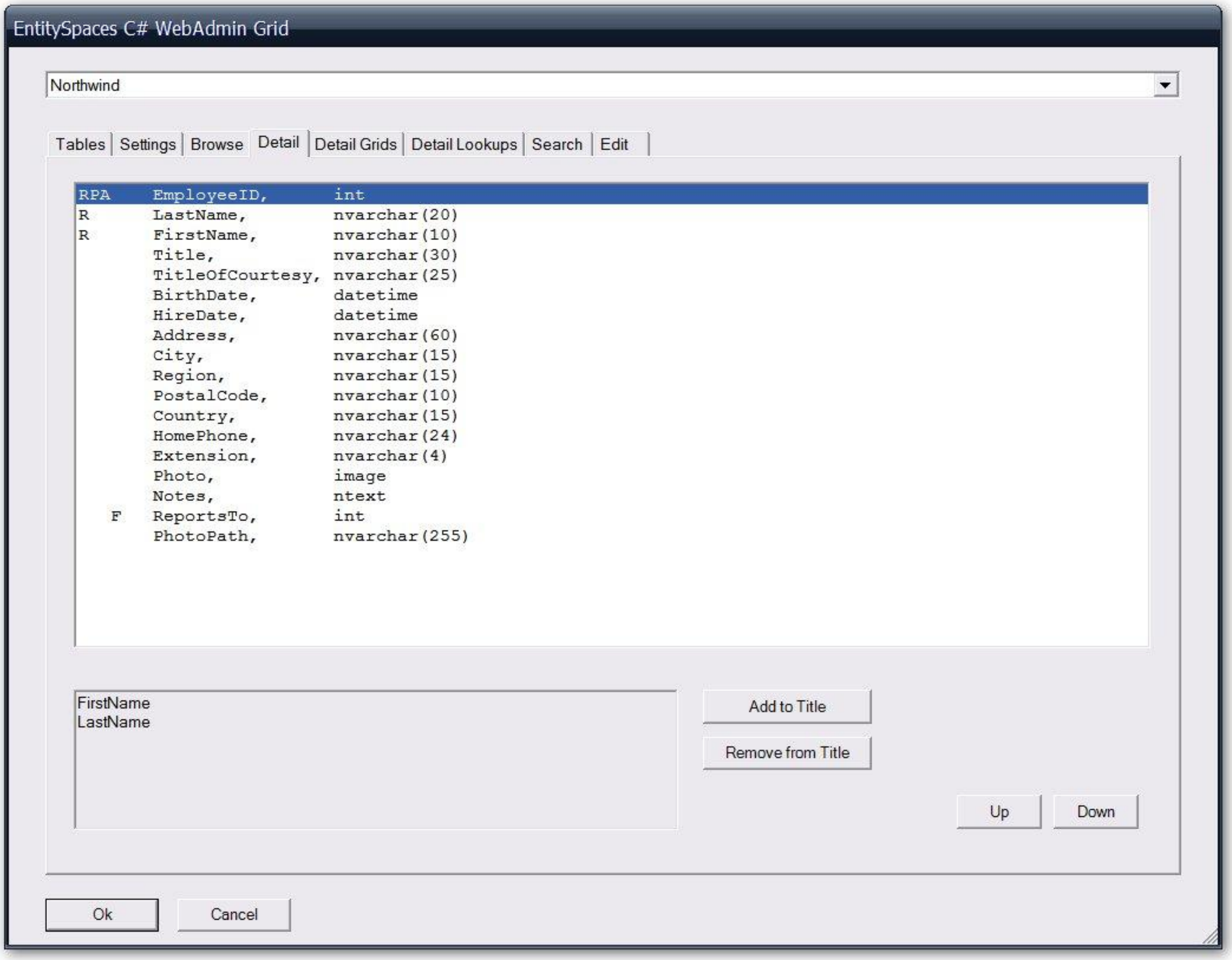
P = Primary Key

A = Auto-Identity column (auto-increment)

F = Foreign Key

Details Tab

The Details Tab controls what is displayed when the user clicks on the 'Select' button in the browse grid. You can control what columns are displayed and in what order by using the Up/Down buttons. Also, when the user selects a particular row (in our case an Employee) it’s nice to indicate what record is being displayed. That is what the 'Add to Title' and 'Remove from Title' are for. In the example below the employees FirstName and LastName will be displayed.



Northwind

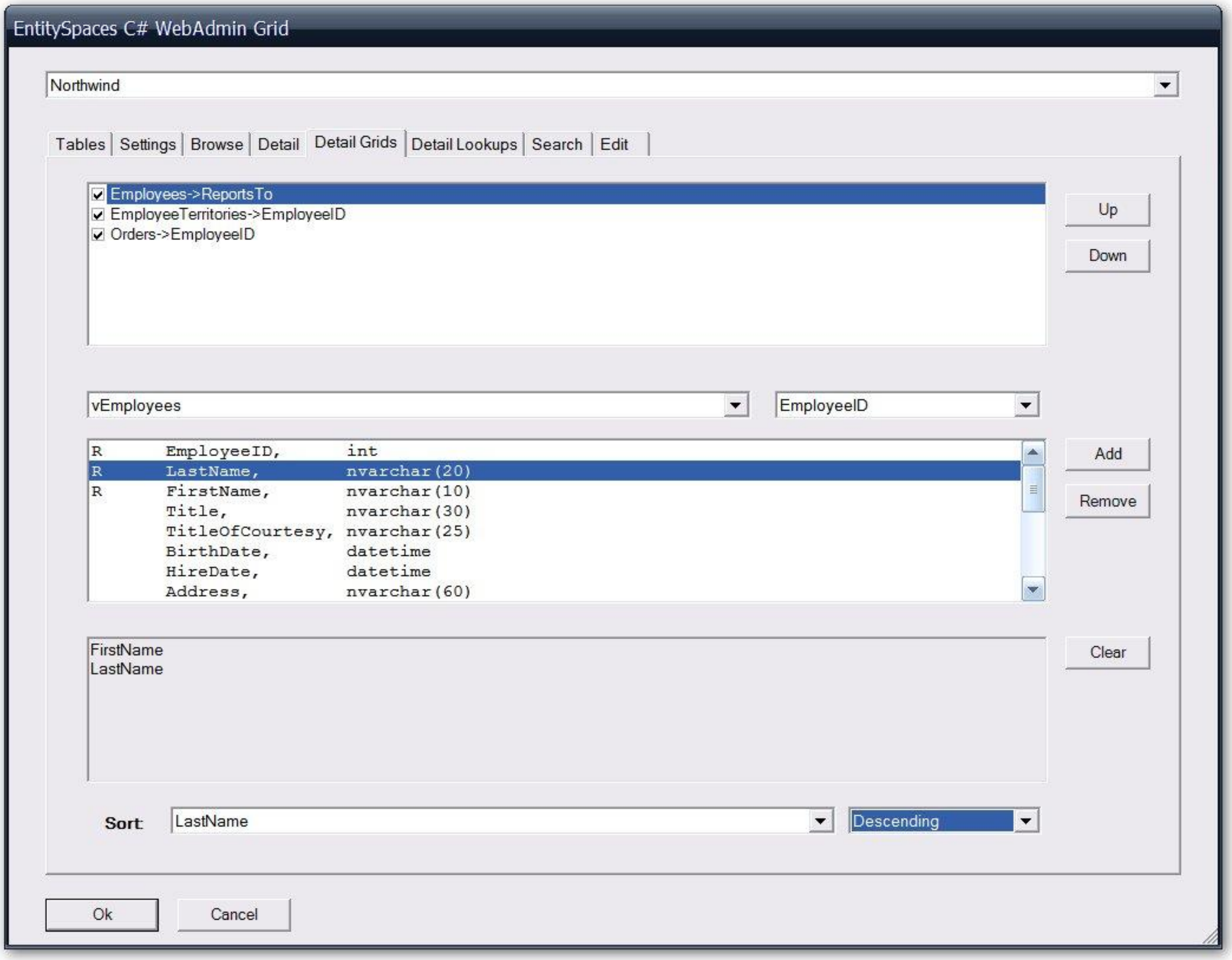
```
RPA    EmployeeID,       int
R      LastName,         nvarchar(20)
R      FirstName,        nvarchar(10)
       Title,            nvarchar(30)
       TitleOfCourtesy,  nvarchar(25)
       BirthDate,        datetime
       HireDate,         datetime
       Address,          nvarchar(60)
       City,             nvarchar(15)
       Region,           nvarchar(15)
       PostalCode,       nvarchar(10)
       Country,          nvarchar(15)
       HomePhone,        nvarchar(24)
       Extension,        nvarchar(4)
       Photo,            image
       Notes,            ntext
   F   ReportsTo,        int
       PhotoPath,        nvarchar(255)
```

FirstName
LastName

Notice that we have selected 'ReportsTo' and that it is a foreign key. Instead of displaying the actual value of ReportsTo (which is an int), we will instead turn it into a link to display the record represented by the value. This is done in the 'Detail Lookups' tab discussed later. Columns with an image data type cannot be viewed or edited.

Detail Grids Tab

The Detail Grids tab allows us to have sub-grids displayed when we navigate to our Detail View. In the Checklist box are all of the ForeignKeys that point to the table you have selected in the Tables tab. If you want to have a sub-grid for that relationship you need to check the checkbox. Then, while you have that row selected, select each column that you want to display, one at a time, and click the Add button to have them show up in the sub-grid. You can also set the default sort column and order.



☑ Employees->ReportsTo
☑ EmployeeTerritories->EmployeeID
☑ Orders->EmployeeID

vEmployees | EmployeeID

```
R       EmployeeID,       int
R       LastName,         nvarchar(20)
R       FirstName,        nvarchar(10)
        Title,            nvarchar(30)
        TitleOfCourtesy,  nvarchar(25)
        BirthDate,        datetime
        HireDate,         datetime
        Address,          nvarchar(60)
```

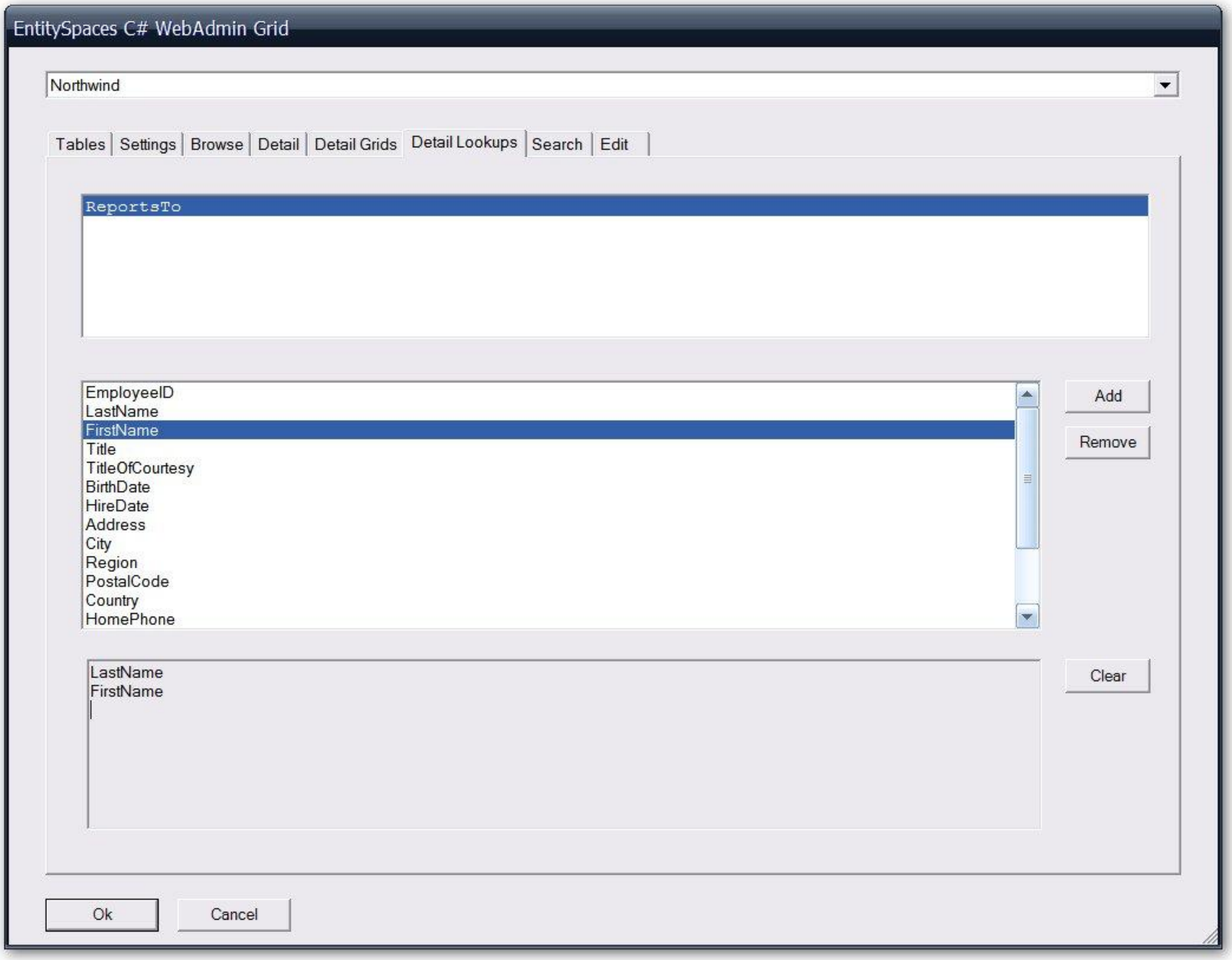
FirstName
LastName

Sort: LastName | Descending

Notice we have, again, chosen to use our vEmployees view rather than the Employees table. It wasn't really necessary but we did this demonstrate the capability.

Detail Lookups Tab

The Detail Lookups tab provides you the ability to replace any foreign key values that would normally be displayed in the Detail View with an actual link to that record. You can see this when you select an employee.



Northwind

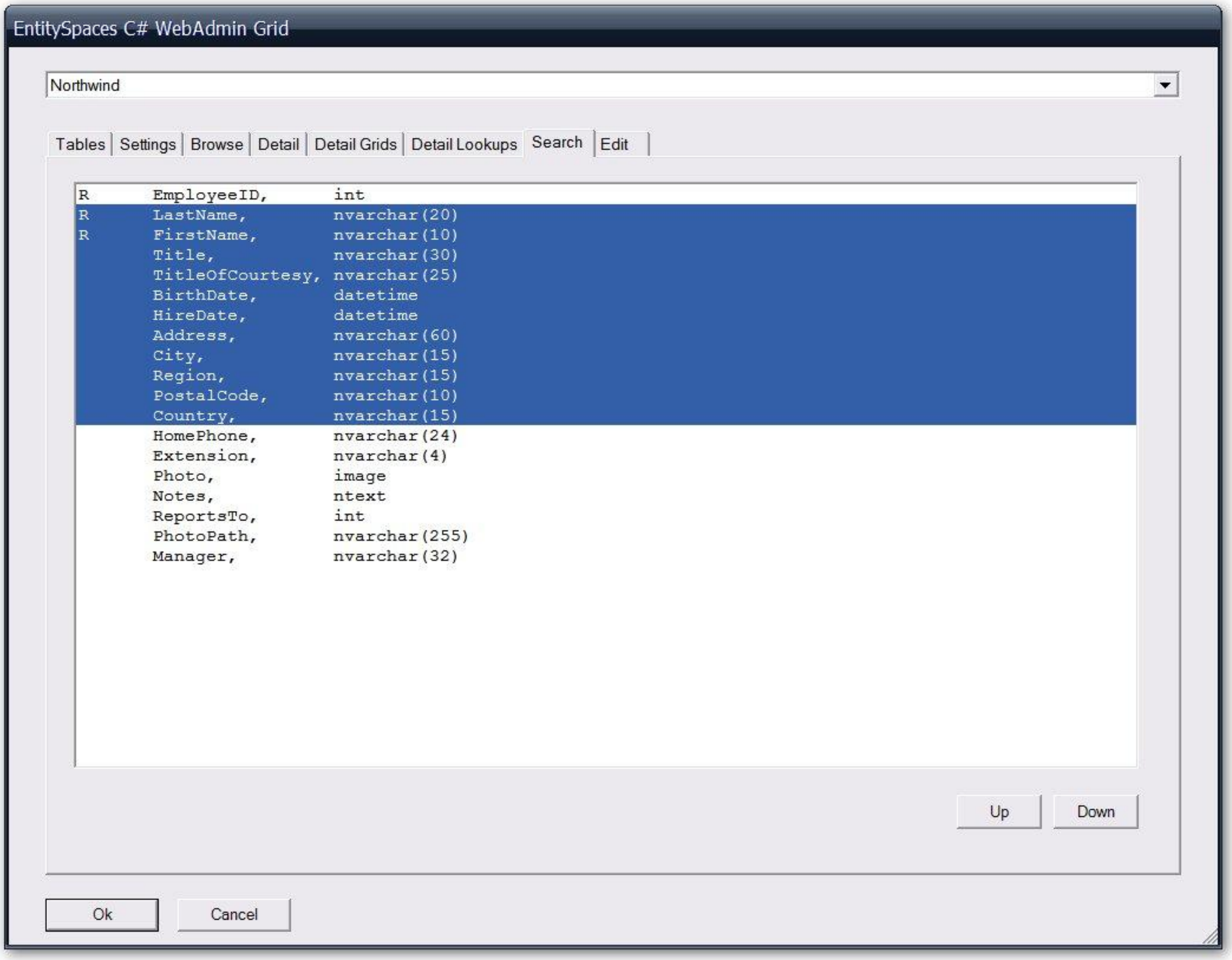
ReportsTo

EmployeeID
LastName
FirstName
Title
TitleOfCourtesy
BirthDate
HireDate
Address
City
Region
PostalCode
Country
HomePhone

LastName
FirstName

Search Tab

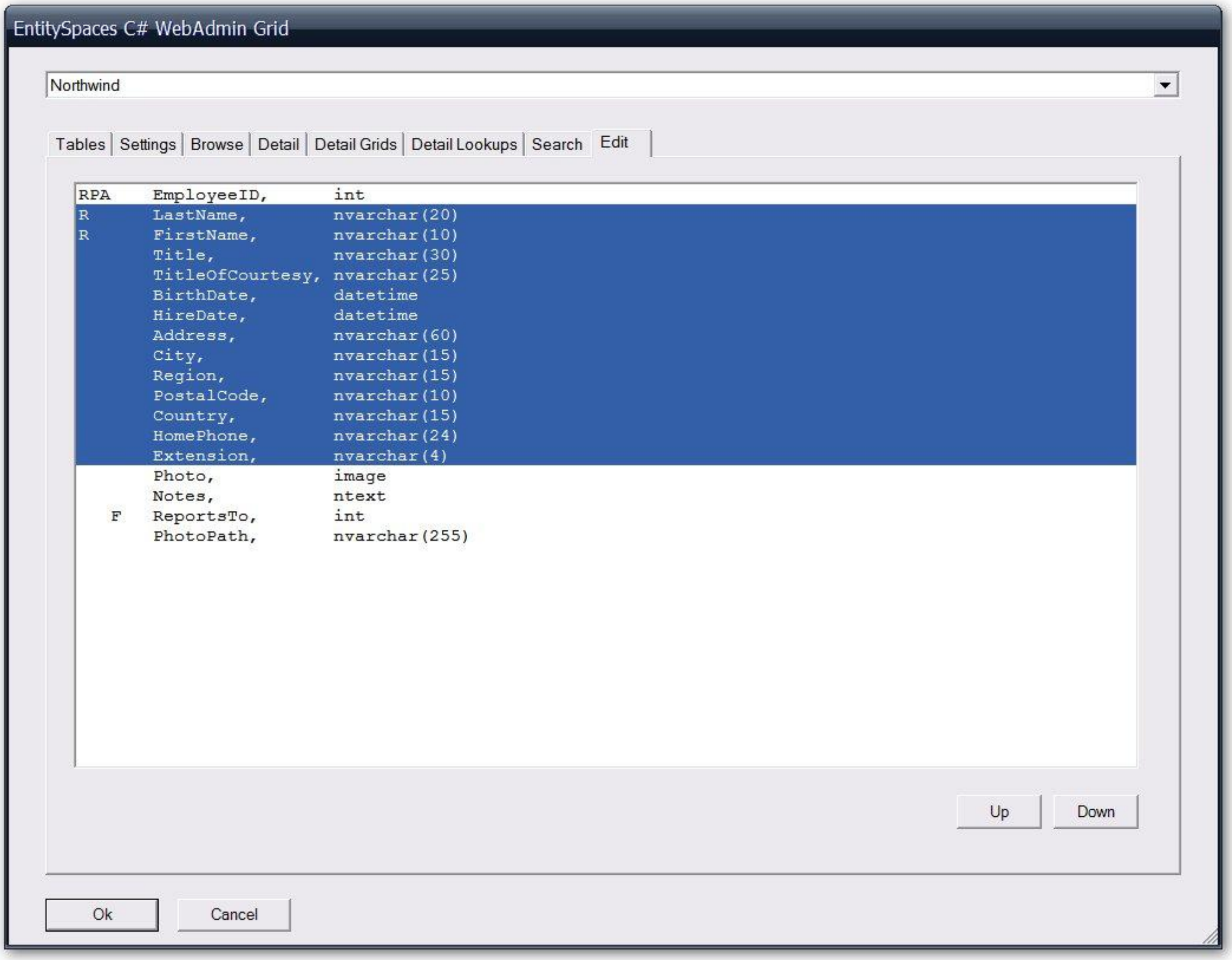
The Search Tab allows you to choose which columns appear on the Search Page. Columns chosen on the Browse Tab will have checkmarks displayed next to them on the Search Page, but all columns selected here will be searchable. You can move the columns up/down to control the order in which they are displayed in the search view as well.



```
R       EmployeeID,       int
R       LastName,         nvarchar(20)
R       FirstName,        nvarchar(10)
        Title,            nvarchar(30)
        TitleOfCourtesy,  nvarchar(25)
        BirthDate,        datetime
        HireDate,         datetime
        Address,          nvarchar(60)
        City,             nvarchar(15)
        Region,           nvarchar(15)
        PostalCode,       nvarchar(10)
        Country,          nvarchar(15)
        HomePhone,        nvarchar(24)
        Extension,        nvarchar(4)
        Photo,            image
        Notes,            ntext
        ReportsTo,        int
        PhotoPath,        nvarchar(255)
        Manager,          nvarchar(32)
```

Edit Tab

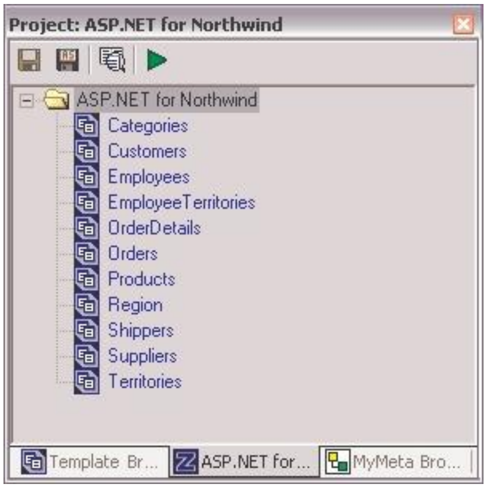
The Edit tab is driven by the Table you have selected in the Tables tab. Again you can control the order of the columns displayed with the up/down buttons. Notice that here we did not select the EmployeeID for display. It is an autoincrement field and not user entered. We chose to hide that from the user. You can choose to show them, if desired. Only the required columns are really needed. Any foreign keys chosen will be presented to the user as combo-boxes as mapped on the Detail Lookups tab.



```
RPA     EmployeeID,       int
R       LastName,         nvarchar(20)
R       FirstName,        nvarchar(10)
        Title,            nvarchar(30)
        TitleOfCourtesy,  nvarchar(25)
        BirthDate,        datetime
        HireDate,         datetime
        Address,          nvarchar(60)
        City,             nvarchar(15)
        Region,           nvarchar(15)
        PostalCode,       nvarchar(10)
        Country,          nvarchar(15)
        HomePhone,        nvarchar(24)
        Extension,        nvarchar(4)
        Photo,            image
        Notes,            ntext
   F    ReportsTo,        int
        PhotoPath,        nvarchar(255)
```

MyGeneration Project Files

Using MyGeneration project files is the best way to use the EntitySpaces ASP.NET Templates. There are a lot of settings and you'll want to regenerate each ASPX page 3 or 4 times until you have it laid out as you desire. Reconfiguring each time from scratch can become very cumbersome. Take the 10 minutes needed to get familiar with MyGeneration projects files, and then you can simply re-edit your last run with modifications, and not have to go through the entire setup each time.



ASP.NET for Northwind

- Categories
- Customers
- Employees
- EmployeeTerritories
- OrderDetails
- Orders
- Products
- Region
- Shippers
- Suppliers
- Territories

The project file above has all of the settings saved for the Northwind demo we are hosting on the EntitySpaces site. To regenerate all of the pages simply click the green arrow and all eleven ascx user controls are generated in about 10 seconds. You can imagine how valuable this is when actually working on the templates.

Conclusion

We hope you have found the web admin grid template documentation helpful. As you can see these templates are very flexible and will save you a great deal of time when administering table data via the web is a requirement.