

2011



# **USER GUIDE**

Appointment Manager

Suppose that you need to create an appointment manager for your business. You have a receptionist in the front office and salesmen ready to service customers. Whenever a customer comes in requesting a consultation, the receptionist will make an appointment in the *Appointment Manager*. The next automatically assigned salesman will be rotated by the application. If a customer requests a specific salesman, then that appointment will not affect the rotation process.

When a new appointment is created, the salesman that performed the smallest number of appointments will be assigned. The rotation process will ensure fair distribution of appointments.

**New Appointments** [X]

Please fill this form and click OK button to create a new appointments record. Click Cancel to return to the previous screen.

**New Appointments**

Complete the form. Make sure to enter all required fields.

Client Name \*

Date \*

Salesman Full Name

Assigned Salesman Full Name Kumar Rajkumar

\* - indicates a required field

OK Cancel

If you enter a client name and press *Ok* to save the appointment, the assigned salesman will be inserted into the salesman field, and the salesman's number of appointments will increase by one. When another appointment is made, the salesman with the next lowest number of appointments will be assigned. This salesman will be different for each appointment.

**New Appointments** [X]

Please fill this form and click OK button to create a new appointments record. Click Cancel to return to the previous screen.

**New Appointments**

Complete the form. Make sure to enter all required fields.

Client Name \*

Date \*

Salesman Full Name

Assigned Salesman Full Name George Fischer

\* - indicates a required field

OK Cancel

If the customer specifically requests a salesman, a salesman will be selected by the receptionist using the lookup for the field *Salesman Full Name*. When the appointment is saved, the number of appointments will not be changed for either salesman.

**New Appointments** [X]

Please fill this form and click OK button to create a new appointments record. Click Cancel to return to the previous screen.

**New Appointments**

Complete the form. Make sure to enter all required fields.

Client Name \*

Date \*

Salesman Full Name

Assigned Salesman Full Name George Fischer

\* - indicates a required field

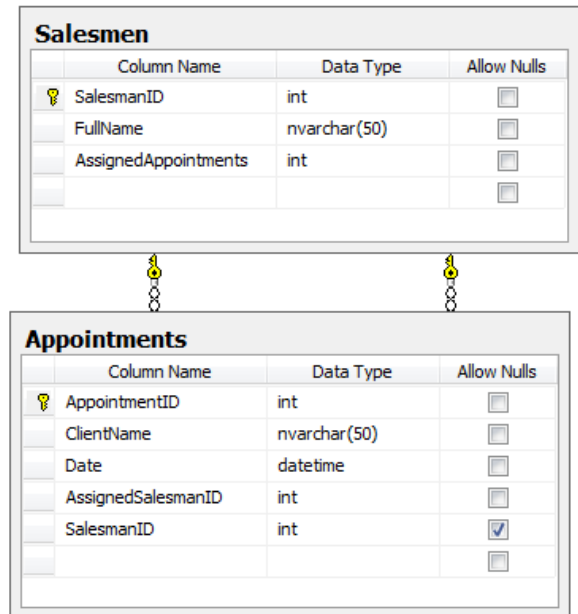
OK Cancel

We will show you how to set up the database using *Microsoft SQL Server 2008 Management Studio*, generate the application using *Code OnTime Generator*, and use the *Designer* in combination with *Visual Web Developer 2008 Express Edition* to implement this algorithm.

To start, we created a database called *ServiceDesk*. This database has two tables, *Appointments* and *Salesmen*. *Salesmen* is a table of all the salesmen in the business, and have the columns *SalesmanID* (an automatically assigned number), *FullName*, and *AssignedAppointments* (a count of how many appointments each salesperson has been assigned).

*Appointments* table is a list of the appointments, and it has *AppointmentID* (an automatically assigned number), *ClientName* (name of the customer), *Date* (the date and time the appointment was made), *AssignedSalesmanID* (indicates the assigned salesperson according to rotation), and *SalesmanID* (the ID of the salesman who actually performed the consultation). Value of *SalesmanID* is only different from *AssignedSalesmanID* if a client specifically requests a salesperson. Both *AssignedSalesmanID* and *SalesmanID* are foreign key references to the *Salesmen* table.

You can see the database schema in the picture to the right.



## SQL Script Creating the Sample Database

```
USE [ServiceDesk]
GO
/***** Object: Table [dbo].[Salesmen] *****/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE TABLE [dbo].[Salesmen](
    [SalesmanID] [int] IDENTITY(1,1) NOT NULL,
    [FullName] [nvarchar](50) NOT NULL,
    [AssignedAppointments] [int] NOT NULL,
    CONSTRAINT [PK_Salesmen] PRIMARY KEY CLUSTERED
(
    [SalesmanID] ASC
)WITH (PAD_INDEX = OFF, STATISTICS_NORECOMPUTE = OFF, IGNORE_DUP_KEY = OFF, ALLOW_ROW_LOCKS =
ON, ALLOW_PAGE_LOCKS = ON) ON [PRIMARY]
) ON [PRIMARY]
GO

/***** Object: Table [dbo].[Appointments] *****/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE TABLE [dbo].[Appointments](
    [AppointmentID] [int] IDENTITY(1,1) NOT NULL,
    [ClientName] [nvarchar](50) NOT NULL,
    [Date] [datetime] NOT NULL,
    [AssignedSalesmanID] [int] NOT NULL,
    [SalesmanID] [int] NULL,
    CONSTRAINT [PK_Appointments] PRIMARY KEY CLUSTERED
(
    [AppointmentID] ASC
)WITH (PAD_INDEX = OFF, STATISTICS_NORECOMPUTE = OFF, IGNORE_DUP_KEY = OFF, ALLOW_ROW_LOCKS =
ON, ALLOW_PAGE_LOCKS = ON) ON [PRIMARY]
) ON [PRIMARY]
GO

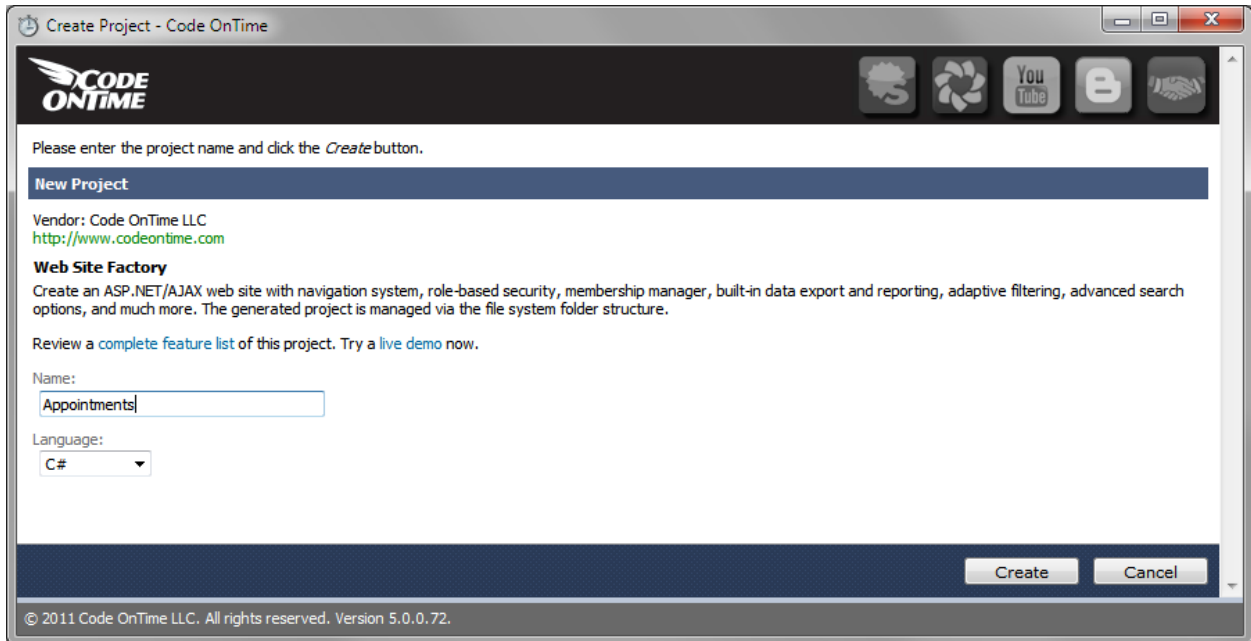
/***** Object: Default [DF_Salesmen_Appointments] *****/
ALTER TABLE [dbo].[Salesmen] ADD CONSTRAINT [DF_Salesmen_Appointments] DEFAULT ((0)) FOR
[AssignedAppointments]
GO

/***** Object: Default [DF_Appointments_Date] *****/
ALTER TABLE [dbo].[Appointments] ADD CONSTRAINT [DF_Appointments_Date] DEFAULT (getdate()) FOR
[Date]
GO

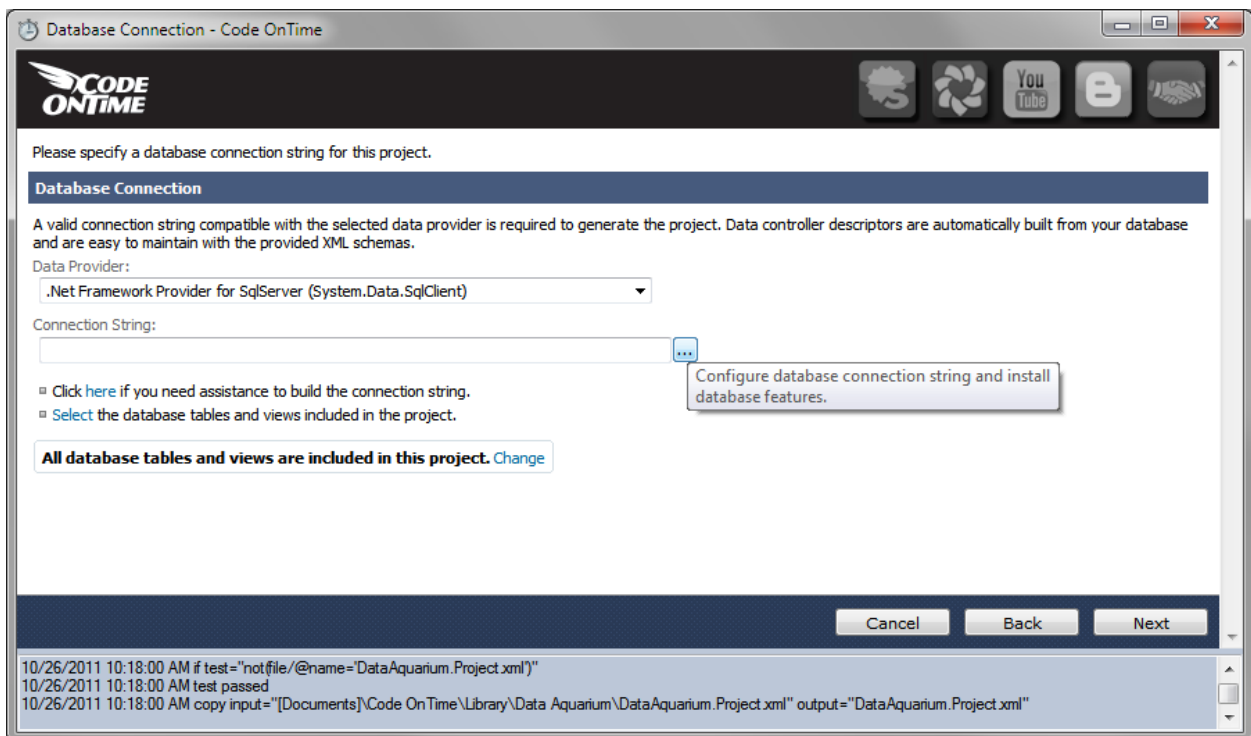
/***** Object: ForeignKey [FK_Appointments_Salesmen] *****/
ALTER TABLE [dbo].[Appointments] WITH CHECK ADD CONSTRAINT [FK_Appointments_Salesmen] FOREIGN
KEY([AssignedSalesmanID])
REFERENCES [dbo].[Salesmen] ([SalesmanID])
GO
ALTER TABLE [dbo].[Appointments] CHECK CONSTRAINT [FK_Appointments_Salesmen]
GO

/***** Object: ForeignKey [FK_Appointments_Salesmen1] *****/
ALTER TABLE [dbo].[Appointments] WITH CHECK ADD CONSTRAINT [FK_Appointments_Salesmen1] FOREIGN
KEY([SalesmanID])
REFERENCES [dbo].[Salesmen] ([SalesmanID])
GO
ALTER TABLE [dbo].[Appointments] CHECK CONSTRAINT [FK_Appointments_Salesmen1]
GO
```

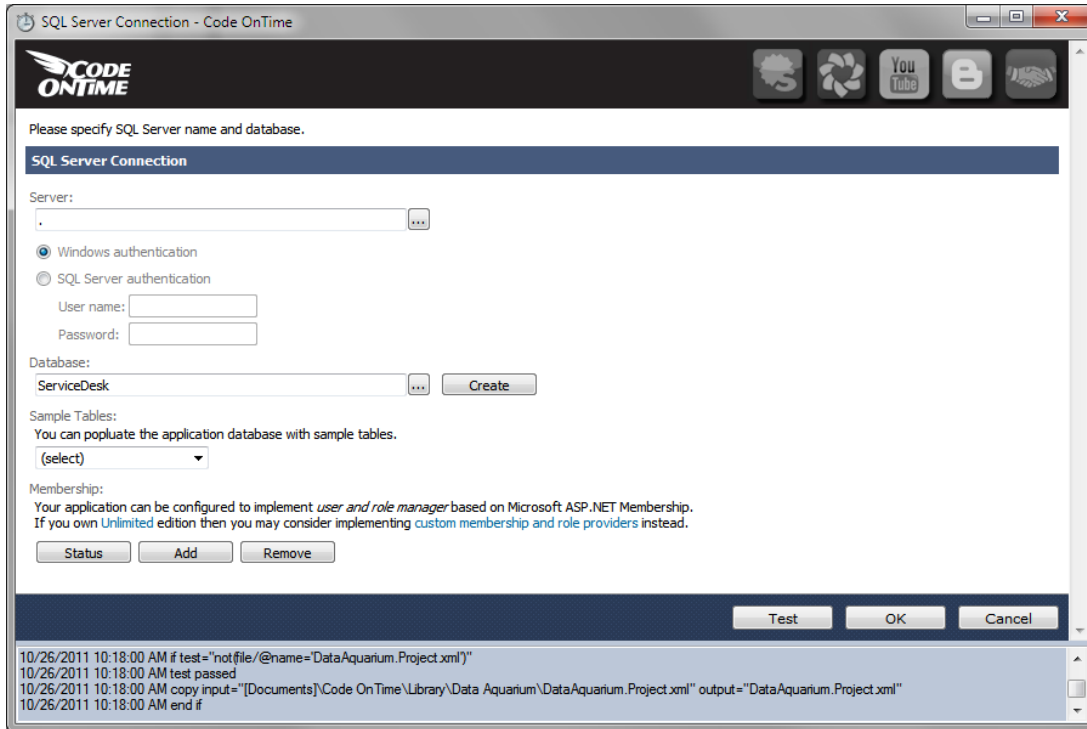
Run *Code OnTime Generator* by clicking on the shortcut on your desktop. Click on the *Create new project* link. Create a new *Web Site Factory* project by choosing it from the list. Enter *Appointments* as the project name.



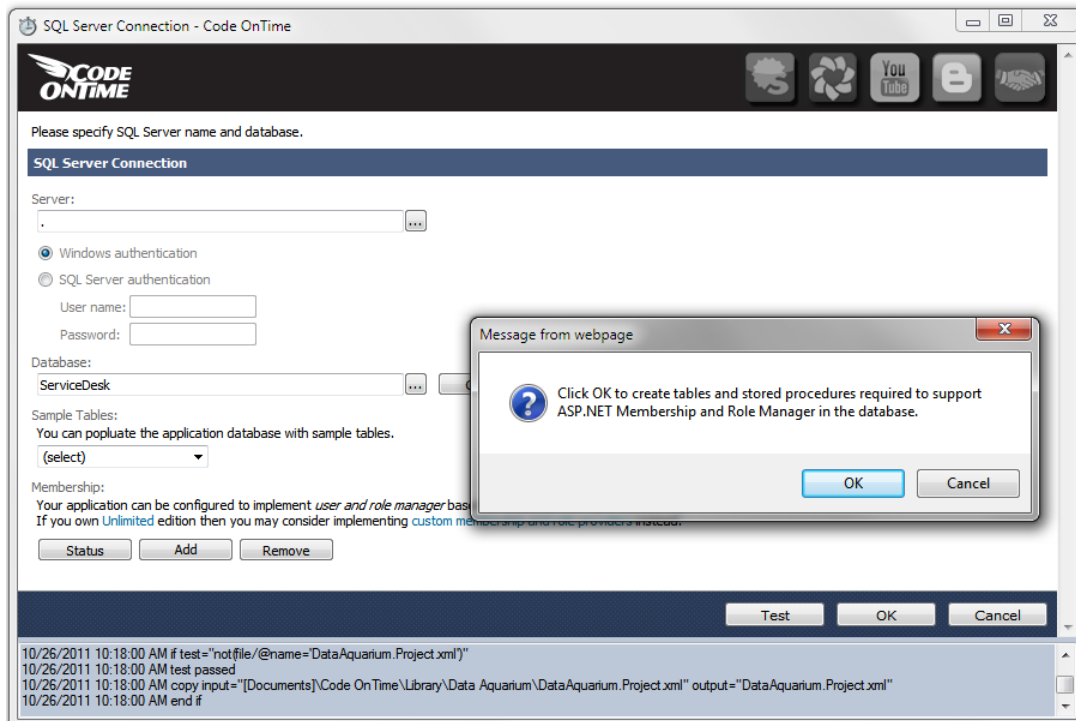
Press the *Create* button in the bottom right corner to create your project. Select .NET Framework 4.0 and press *Next*. On this page, click on the "..." button to the left of the *Connection String* field.



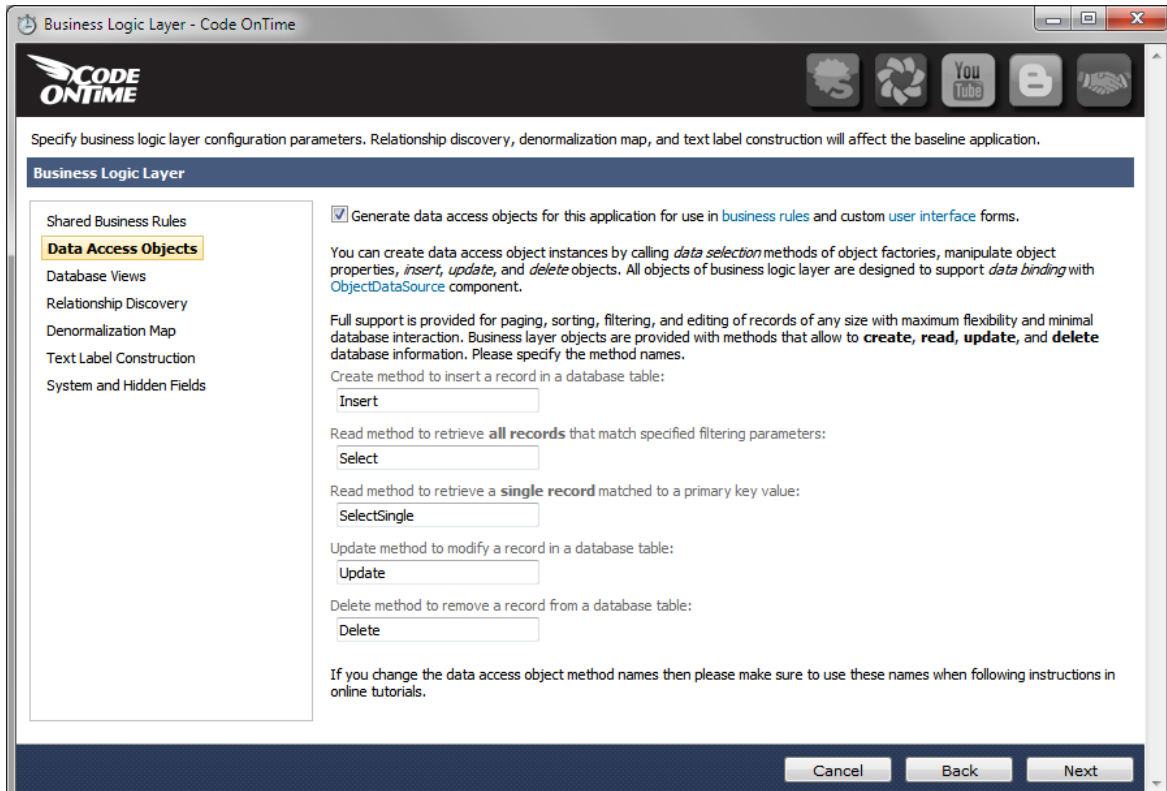
It will take you to a page where you will type in the database server name, and select the database. Press *Test* to confirm.



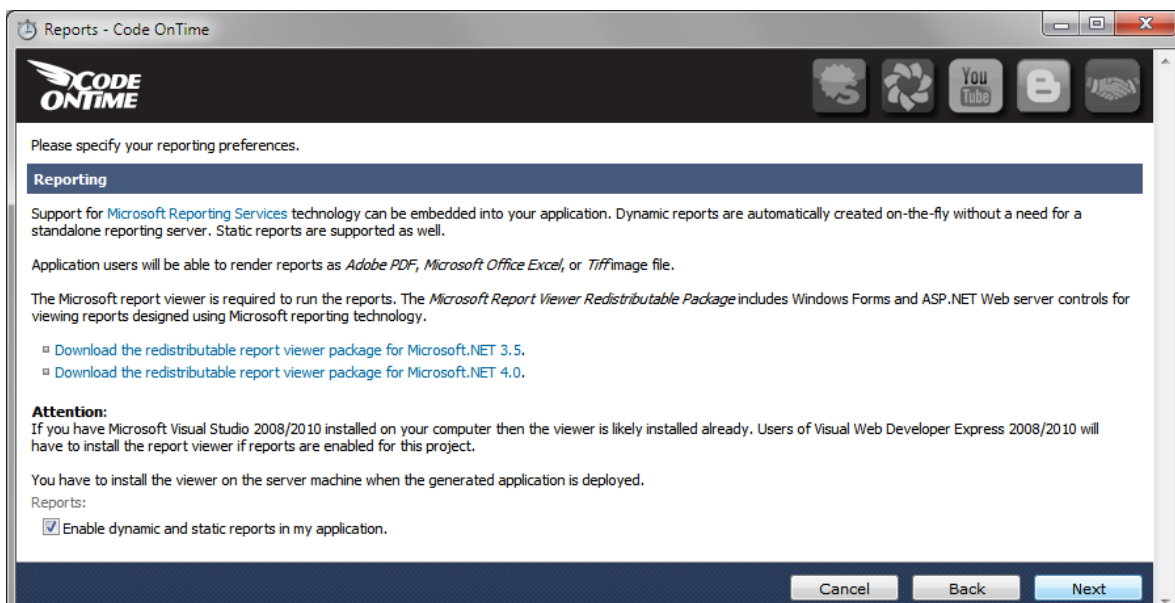
If the connection succeeds, press the “Add” button under *Membership* to add *ASP.NET Membership* to your application.



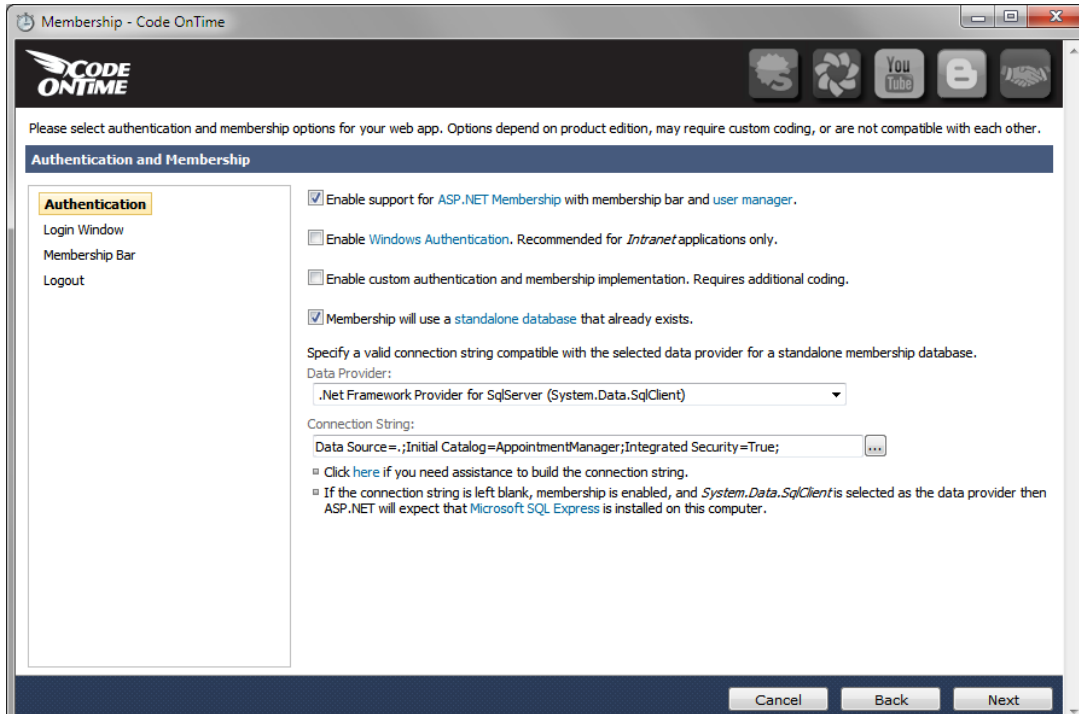
Press OK, and go back to the Database Connection page. Press Next to continue. In the list of items, switch to Data Access Objects. Enable generation of data access objects by checking the box. These will help you write data manipulation logic for the application, as explained later in this article. The options do not need to be changed beyond that on this page, so proceed by pressing the *Next* button in the lower right corner.



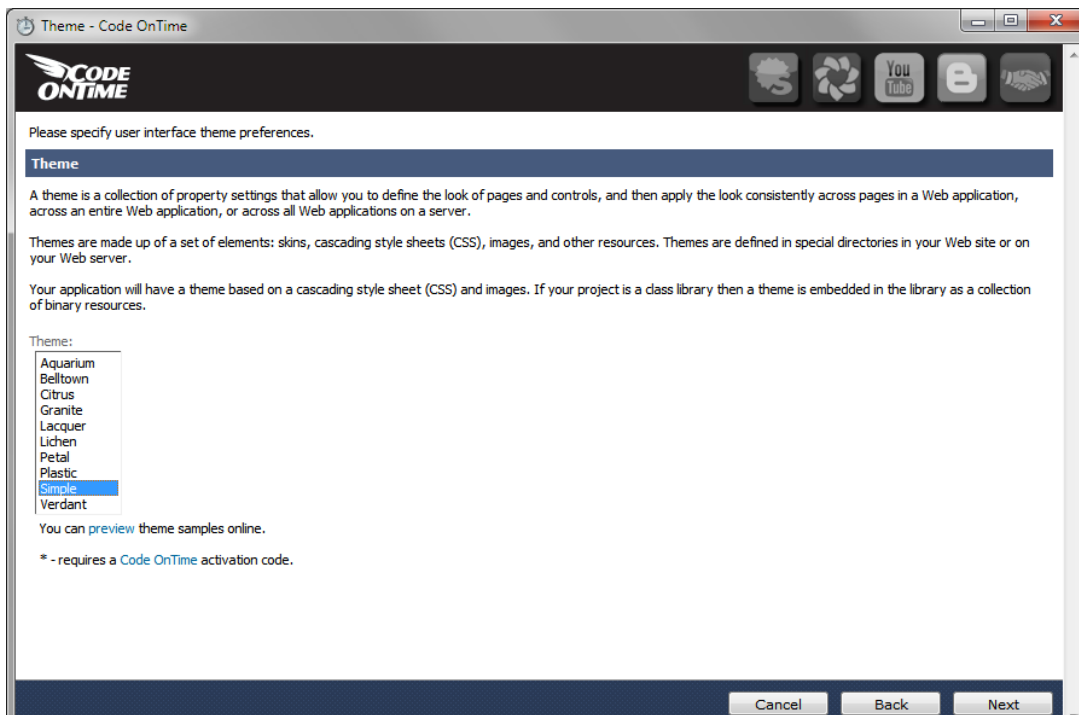
You can enable reports by checking the box on the next page. This requires *Microsoft Report Viewer 2008 Redistributable Package* to view reports. The viewer comes with *Microsoft Visual Studio 2008*.



Proceed to the *Authentication and Membership* page by pressing *Next*. If you installed ASP.NET Membership previously, then these settings should be configured correctly.

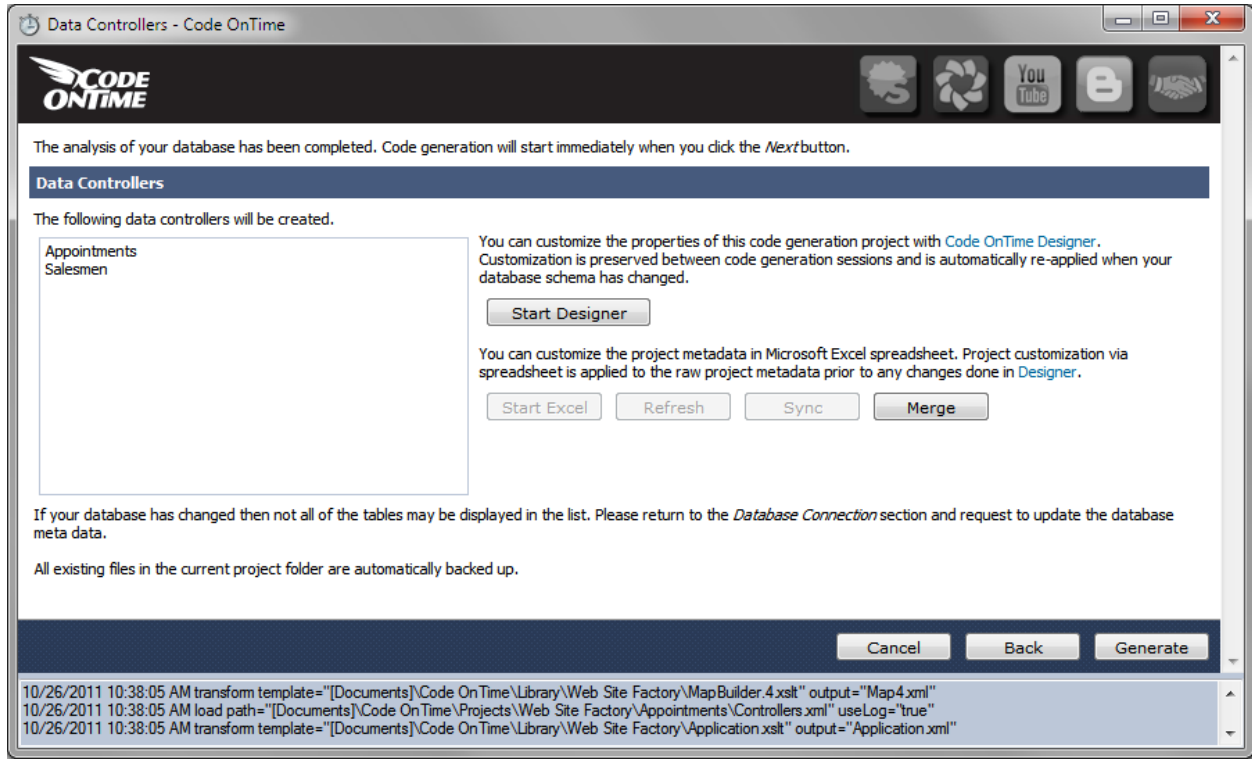


Keep on pressing *Next* until you reach the *Theme* page. This page allows you to choose the theme of the project. Themes are made up of cascading style sheets, images, and other resources. Choose *Simple* theme from the list box.





Keep on pressing the *Next* button until you reach the *Data Controllers* page. It will display a list of the data controllers on the left, and some buttons on the right side of the page.



Press the *Start Designer* button on the Data Controllers page. This will bring up the *Designer*, and display the *All Controllers* page.



Now, click on the *Salesmen* controller in the *All Controllers* list (selections are made by clicking on the link in the first column). You can see that the tab set has changed, so click on the *Fields* tab. Select *AssignedAppointments* field. Press *Edit*, and change *Code Default* field to "0". This way, the default value of the field is presented as 0 in the user interface. Press *Ok* to save changes.

The screenshot shows the 'Project Designer' application window. The breadcrumb path is 'Home > Controller: Salesmen > Field: AssignedAppointments (Int32)'. The 'Fields' tab is active. Below the breadcrumb, there are tabs for 'Field', 'Items', 'Validators', 'Data Fields', and 'Field Outputs'. A message reads: 'Please review the field information below. Click Edit to change this record, click Delete to delete the record, or click Cancel/Close to return back.' The 'Record' dropdown is set to 'Field'. There are three buttons: 'OK', 'Delete', and 'Cancel'. A note states '\* - indicates a required field'. The 'General' section contains instructions on field naming, server defaults, computed fields, and calculated fields. The configuration fields on the right are: Name: 'AssignedAppointments', Controller: 'Salesmen', Type: 'Int32', 'Allow null values' (unchecked), 'The value of this field is computed at run-time by SQL expression' (unchecked), 'The value of the field is calculated by a business rule expression' (unchecked), Server Default: '((0))', and Code Default: '0'.

Go back to the *All Controllers* page by pressing the back icon on the left-hand side of page path. Select the *Appointments* controller. Press *Edit*, and change the *Handler* field to "AppointmentBusinessRules". This will create a business rule class, which allows altering the controller's behavior.

The screenshot shows the 'Project Designer' application window. The breadcrumb path is 'Home > Controller: Appointments'. The 'Controllers' tab is active. Below the breadcrumb, there are tabs for 'Controller', 'Commands', 'Fields', 'Views', 'Categories', 'Data Fields', 'Action Groups', and 'Actions'. A message reads: 'Please review data controller information below. Click Edit to change this record, click Delete to delete the record, or click Cancel/Close to return back.' The 'Record' dropdown is set to 'Controller'. The 'General' section contains instructions on controller naming and code generation. The configuration fields on the right are: Controller Name: 'Appointments', 'Include in code generation' (checked), Conflict Detection: 'Overwrite Changes' (selected), 'Compare All Values' (unselected), Connection String Name: (empty), and Handler: 'AppointmentBusinessRules'.

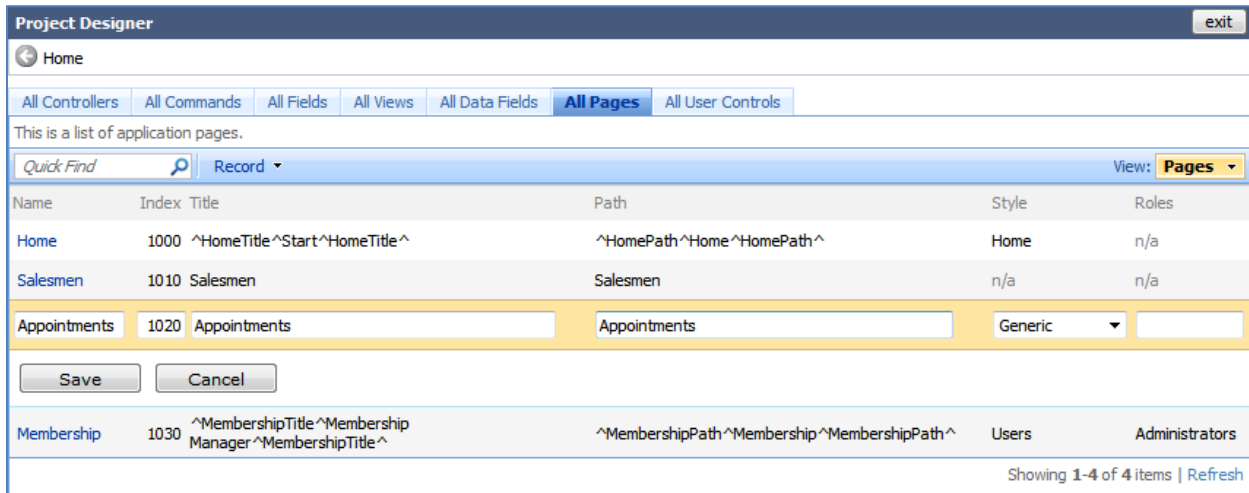
Press *Save*. Click on the *Appointments* controller. Navigate to the *Fields* tab using the tabs at the top, and select the field *Date*. In the *Data Format String* field, type in “g”, so that date and time will be displayed for the appointment date when presented in the user interface. Press *Ok* to save.

The screenshot shows the 'Project Designer' window with the 'Field: Date (DateTime)' configuration page. The 'General' section contains instructions on field properties like 'Server Default', 'Computed', 'Calculated', 'Code Default', and 'Code Value'. The 'Presentation' section explains data format strings and editors. On the right, the configuration fields are: Name: Date; Controller: Appointments; Type: DateTime; Allow null values: unchecked; Computed at run-time: unchecked; Calculated by business rule: unchecked; Server Default: (getdate()); Code Default: empty; Code Value: empty; Value is retrieved on demand: unchecked; Field included in all data views: unchecked; Label: Date; Values cannot be edited: unchecked; Show In Summary: checked; HTML encoding: checked; Data Format String: g.

Press the *back* icon in the upper left corner. Navigate to the *Views* tab, and select *grid1*. Press the *Edit* button, and change the *Sort Expression* field to “Date desc”, so that the latest appointment will always appear at the top of the list.

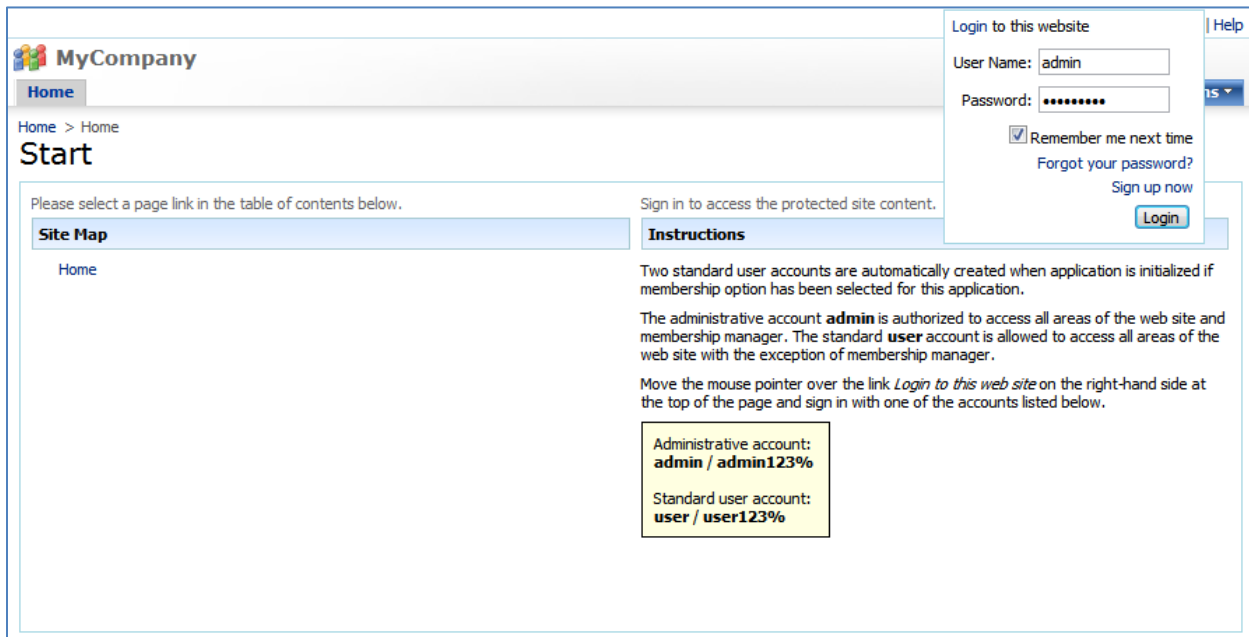
The screenshot shows the 'Sort and Filter' configuration page. The 'Sort Expression' field is set to 'Date desc'. The 'Filter Expression' field is empty. The page includes instructions on how to use sort and filter expressions.

Press *Ok* to save. Now, navigate to *All Controllers* tab using the *Back* button in the upper left corner. Then, navigate to the *All Pages* tab. Select *Appointments*, and change the *Path* field from “Salesmen | Appointments” to just “Appointments”, so that it appears on the main tabs, not underneath Salesmen.



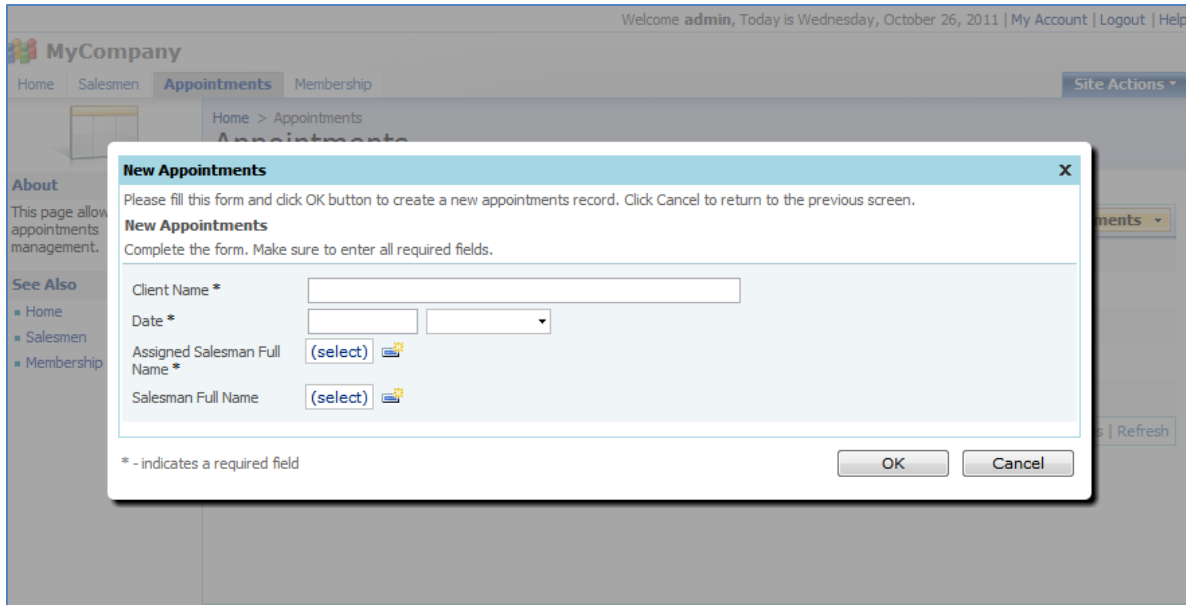
Press *Ok* to save. Press the *Exit* link in the upper right corner to go back to the Generator, and then press *Generate* to start code generation. When the code generator finishes, a webpage will pop up with your generated application.

You can see that a sophisticated web application has been generated. You will need to log in using the membership bar at the top of the page. Use standard account *admin/admin123%* to sign in.



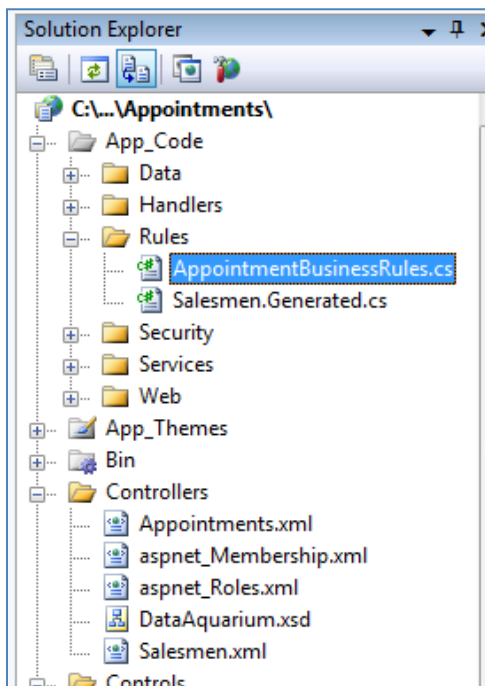
This will take you to the Home screen. Click on the *Appointments* tab. Press *New Appointment*, and you will be led to the *New Appointments* page. Here, you can see that a few optimizations still need to be

made. The date is blank, and will only be inserted when the appointment is saved. There is no automatically assigned *Salesman* or *Assigned Salesman* in the respective fields.



Press *Ok* to save the appointment. This will take you back to list view. Notice that *Salesman* field is not automatically assigned if not selected in the *New Appointment* page.

Return to code generator and click on the *Develop* link next to the name of your project. The project will open in Visual Studio. Using the *Solution Explorer*, navigate to the location *App\_Code\Rules\AppointmentBusinessRules.cs(vb)* as shown in the picture.



This is how you will implement the required functionality:

Create method called *AssignDefaultSalesman* adorned with the *RowBuilder* attribute to handle new rows in *Appointments* data controller. When a user starts entering a new row, then several things will occur. The field *Date* will be updated with the current time, using *UpdateFieldValue*. The *Salesmen* rotation will be requested from the database in ascending order of *AssignedAppointments*. The first *SalesmanID* found will be inserted into the field *AssignedSalesmanID* of the new row. Also, the salesman's *FullName* will be copied into *AssignedSalesmanFullName* of the row.

Create another method named *UpdateAppointmentSalesman* with the *ControllerAction* attribute. The attribute will ensure that the code is executed on the server for new appointments just before the record is inserted in the database. If *SalesmanID* is null, then *AssignedSalesmanID* will be copied into *SalesmanID*.

One more method needs to be created, with the name *UpdateAssignedSalesman*. After an appointment is inserted, *SalesmanID* will be compared to *AssignedSalesmanID*. If the fields have the same value, then that specific *Salesman* record will have its *AssignedAppointments* field increased by one.

The actual code is presented next in C# and VB.NET on the next pages. Copy and paste the code into your own application.

## AppointmentBusinessRules.cs

```
using System;
using System.Data;
using System.Collections.Generic;
using System.Linq;
using MyCompany.Data.Objects;
using MyCompany.Data;

namespace MyCompany.Rules
{
    public partial class AppointmentBusinessRules : MyCompany.Data.BusinessRules
    {
        [RowBuilder("Appointments", RowKind.New)]
        public void AssignDefaultSalesman()
        {
            UpdateFieldValue("Date", DateTime.Now);
            using (SqlText findSalesman = new SqlText(
                "select * from Salesmen order by AssignedAppointments"))
            {
                if (findSalesman.Read())
                {
                    UpdateFieldValue("AssignedSalesmanID", findSalesman["SalesmanID"]);
                    UpdateFieldValue("AssignedSalesmanFullName",
                        findSalesman["FullName"]);
                }
            }
        }

        [ControllerAction("Appointments", "Insert", ActionPhase.Before)]
        public void UpdateAppointmentSalesman(int assignedSalesmanID,
            FieldValue salesmanID)
        {
            if (salesmanID.Value == null)
            {
                salesmanID.NewValue = assignedSalesmanID;
                salesmanID.Modified = true;
            }
        }

        [ControllerAction("Appointments", "Insert", ActionPhase.After)]
        public void UpdateAssignedSalesman(int assignedSalesmanID, int salesmanID)
        {
            if (assignedSalesmanID == salesmanID)
            {
                Salesmen s = Salesmen.SelectSingle(salesmanID);
                s.AssignedAppointments = s.AssignedAppointments + 1;
                s.Update();
            }
        }
    }
}
```

## AppointmentBusinessRules.vb

```
Imports MyCompany.Data
Imports System
Imports System.Collections.Generic
Imports System.Data
Imports System.Linq
Imports MyCompany.Data.Objects

Namespace MyCompany.Rules

    Partial Public Class AppointmentBusinessRules
        Inherits MyCompany.Data.BusinessRules

        <RowBuilder("Appointments", RowKind.New)> _
        Public Sub AssignDefaultSalesman()
            UpdateFieldValue("Date", DateTime.Now)
            Using findSalesman As SqlText = New SqlText( _
                "select * from Salesmen order by AssignedAppointments")
                If (findSalesman.Read()) Then
                    UpdateFieldValue("AssignedSalesmanID", findSalesman("SalesmanID"))
                    UpdateFieldValue("AssignedSalesmanFullName", findSalesman("FullName"))
                End If
            End Using
        End Sub

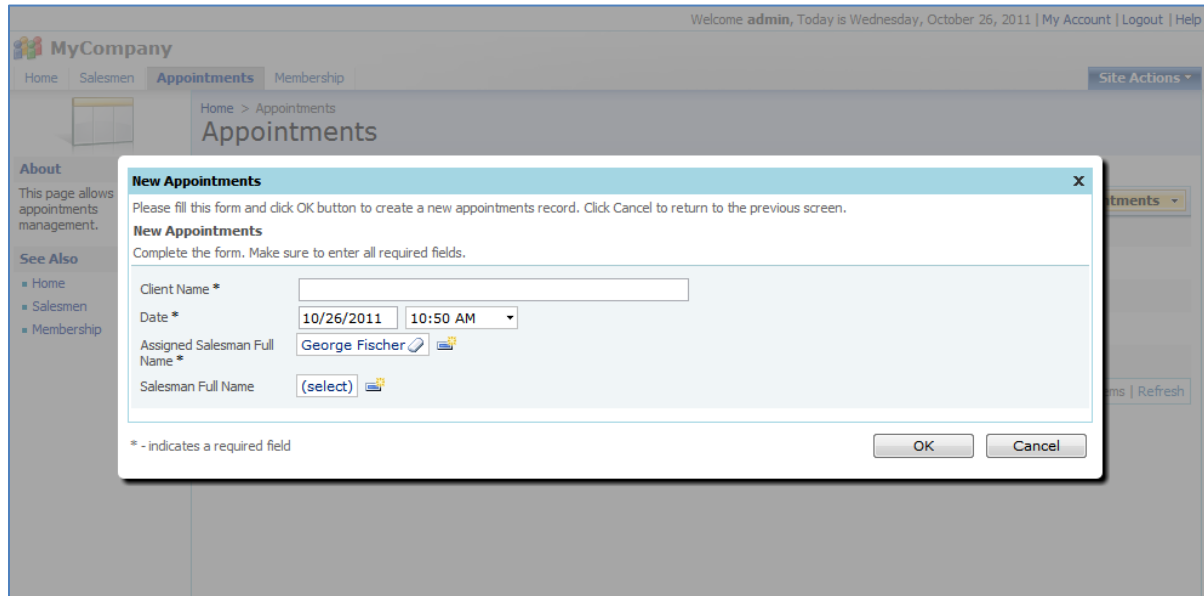
        <ControllerAction("Appointments", "Insert", ActionPhase.Before)> _
        Public Sub UpdateAppointmentsSalesman(ByVal assignedSalesmanID As Integer, _
            ByVal salesmanID As FieldValue)
            If salesmanID.Value Is Nothing Then
                salesmanID.NewValue = assignedSalesmanID
                salesmanID.Modified = True
            End If
        End Sub

        <ControllerAction("Appointments", "Insert", ActionPhase.After)> _
        Public Sub UpdateAssignedSalesman(ByVal assignedSalesmanID As Integer, _
            ByVal salesmanID As Integer)
            If (assignedSalesmanID = salesmanID) Then
                Dim s As Salesmen = Salesmen.SelectSingle(salesmanID)
                s.AssignedAppointments = s.AssignedAppointments + 1
                s.Update()
            End If
        End Sub

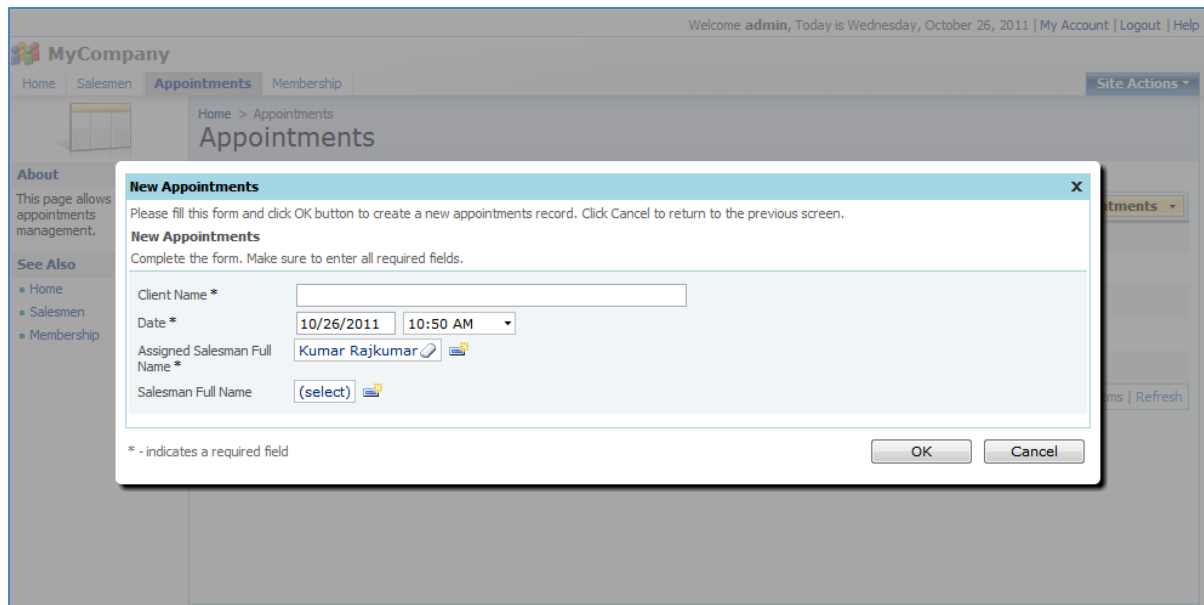
    End Class
End Namespace
```



Let's see if this code works. If you go back to the *Code OnTime Generator*, and press *Browse* next to the project name, the website will run. Navigate to the *Appointments* tab. If you press *New Appointments* on the action bar, you will be taken to the *New Appointments* page. You can see that date is automatically assigned into the field, and an *Assigned Salesman Full Name* will be automatically assigned, based on the *Assigned Appointments* of each salesman.



If you save this appointment, and then create another appointment, the next salesman in line will be automatically inserted. In the picture below, we created a new appointment, and then pressed *New Appointments* again. Notice how the *Assigned Salesman Full Name* has changed to cycle to the next salesman.



## Making Assigned Salesman Full Name a Read-only Field

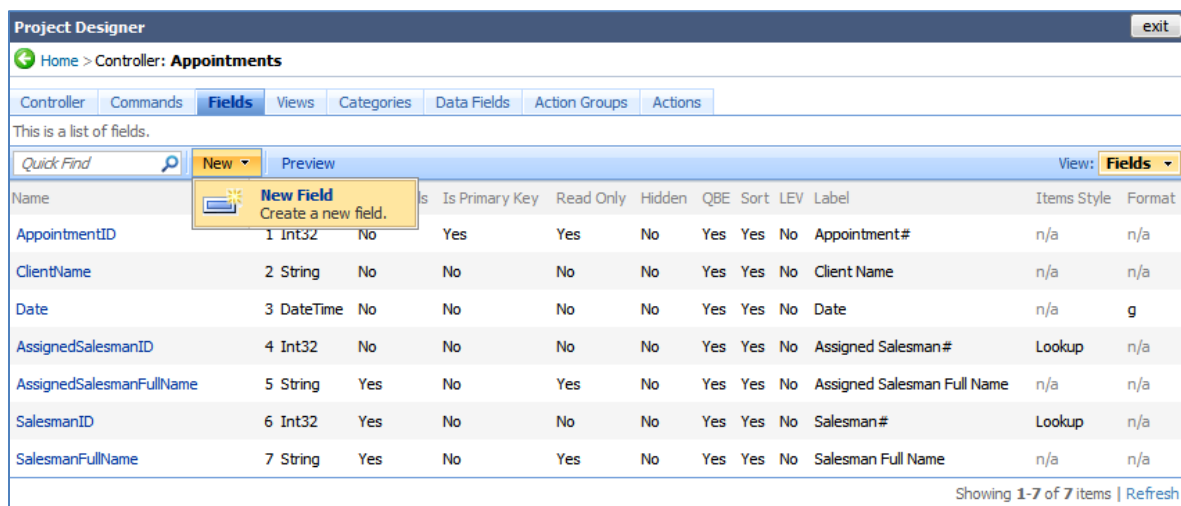
Field *AssignedSalesmanID* is calculated on the server. It travels to the client, and when the record is sent to the server for insertion into the database, the client will send the field value to the server, along with all the other editable field values, and primary key field values. It is possible that you will want to have field *AssignedSalesmanID* presented as read-only text, so that the rotation of salespeople cannot be affected by users.

If you simply mark the field as read-only in *Designer*, then the client browser will not send the field value to the server, and the calculation of *AssignedSalesmanID* will be lost.

You can overcome this by creating a computed field that uses *AssignedSalesmanFullName* as an alias. Here's how you can do that.

Switch back to the Generator. Click on the project name, and then press *Design*. On the *All Controllers* tab, click on the *Appointments* controller, and switch to the *Fields* tab.

On the action bar, press *New*, then *New Field*.



Name	Is Primary Key	Read Only	Hidden	QBE	Sort	LEV	Label	Items Style	Format
AppointmentID	Yes	Yes	No	Yes	Yes	No	Appointment#	n/a	n/a
ClientName	No	No	No	Yes	Yes	No	Client Name	n/a	n/a
Date	No	No	No	Yes	Yes	No	Date	n/a	g
AssignedSalesmanID	No	No	No	Yes	Yes	No	Assigned Salesman#	Lookup	n/a
AssignedSalesmanFullName	No	Yes	No	Yes	Yes	No	Assigned Salesman Full Name	n/a	n/a
SalesmanID	Yes	No	No	Yes	Yes	No	Salesman#	Lookup	n/a
SalesmanFullName	No	Yes	No	Yes	Yes	No	Salesman Full Name	n/a	n/a

Enter the field name as “AssignedSalesmanDisplayFullName”, choose “String” as field type. Allow null values.

Check the box next to “The value of this field is computed at run-time”. The field *SQL Formula* will appear. Enter text “NULL” as the field value.

In the *Label* field, type “Assigned Salesman”. Check the checkbox next to “Values of this field cannot be edited”.

**Project Designer** exit

Home > Controller: **Appointments**

Controller Commands **Fields** Views Categories Data Fields Action Groups Actions

Please fill this form and click OK button to create a new field record. Click Cancel to return to the previous screen.

View: **New Field** ▾

\* - indicates a required field OK Cancel

---

**New Field**

Specify field name, type, and data properties of the field.

*Server Default* is a SQL expression used as a field value when no value is provided for the field in INSERT and UPDATE statement.

Indicate that the field is *computed* if the field is not physically present in the dataset produced by controller's command. Computed field requires a mandatory *formula* that must be defined as a valid SQL expression. This expression is automatically inserted in SELECT statements when needed.

*Calculated* field values can be produced by business rule methods with attribute ControllerAction. You must list the context fields that will cause the calculation. Optional code formula is embedded into an automatically created business rule and is calculated whenever any context field is changed.

*Code Default* is an expression written in the programming language of your project. The expression is evaluated in an automatically created business rule to produce a default value for the field of the new row before it is presented in the user interface.

*Code Value* is an expression written in the programming language of your project. This expression is evaluated every time a record is saved to the database.

The field must be marked as *on-demand* if the field is a large binary object (BLOB) or text in order to speed up record retrieval.

**Name \***  
AssignedSalesmanDisplayFullName

**Type \***  
String ▾

Allow null values.

The value of this field is computed at run-time by SQL expression.

**SQL Formula**  
NULL

The value of the field is calculated by a business rule expression.

**Server Default**  
\_\_\_\_\_

**Code Default**  
\_\_\_\_\_

**Code Value**  
\_\_\_\_\_

Value is retrieved on demand

The field is included in all data views but remains hidden from users.

---

**Presentation**

Presentation interface properties of the field.

A data format string is applied to the field value on the client via JavaScript *String.format* function. You can override the data format string on the data field level in views.

**Label \***  
Assigned Salesman

Values of this field cannot be edited.

Press *Ok* to save. You will be taken back to the grid view of fields.

Click on *AssignedSalesmanDisplayFullName*, the record you just created. Navigate to the *Data Fields* tab (which should be empty), and on the action bar, press *New*, then *New Field*. For *View* lookup, choose "grid1". To use the lookup, click on (*select*), and choose your option from the list.

Station Columns

This is a list of data controller views. X

Quick Find View: **Views** ▾

Id	Type	Command	Label	Header Text	Group Selector	Report Label	Orientation	Font	Template
grid1	Grid	command1	Appointments	\$DefaultGridViewDescription	n/a	Yes	n/a	(Auto)	(Auto) No
editForm1	Form	command1	Review Appointments	\$DefaultEditViewDescription	n/a	Yes	n/a	(Auto)	(Auto) No
createForm1	Form	command1	New Appointments	\$DefaultCreateViewDescription	n/a	Yes	n/a	(Auto)	(Auto) No

Showing 1-3 of 3 items | Refresh

The *Alias* field should be *AssignedSalesmanFullName*. Use the lookup window to select the correct option.

The screenshot shows the 'Project Designer' window with the 'Data Fields' tab selected. The breadcrumb path is 'Home > Controller: Appointments > Field: AssignedSalesmanDisplayFullName (String)'. Below the breadcrumb are tabs for 'Field', 'Items', 'Validators', 'Data Fields', and 'Field Outputs'. A message reads: 'Please fill this form and click OK button to create a new data field record. Click Cancel to return to the previous screen.' The 'View' dropdown is set to 'New Data Field'. A note states '\* - indicates a required field'. The 'New Data Field' section contains the following fields: 'View \*' with the value 'grid1', 'Category' with the value '(select)', and 'Alias' with the value 'AssignedSalesmanFullName'. 'OK' and 'Cancel' buttons are visible at the top right.

Press *Ok* to save the data field. Again, press *New* on the action bar, then *New Data Field* on the dropdown menu. This time, *View* field will be "editForm1", *Category* field will be "Appointments", and the *Alias* field will be "AssignedSalesmanFullName".

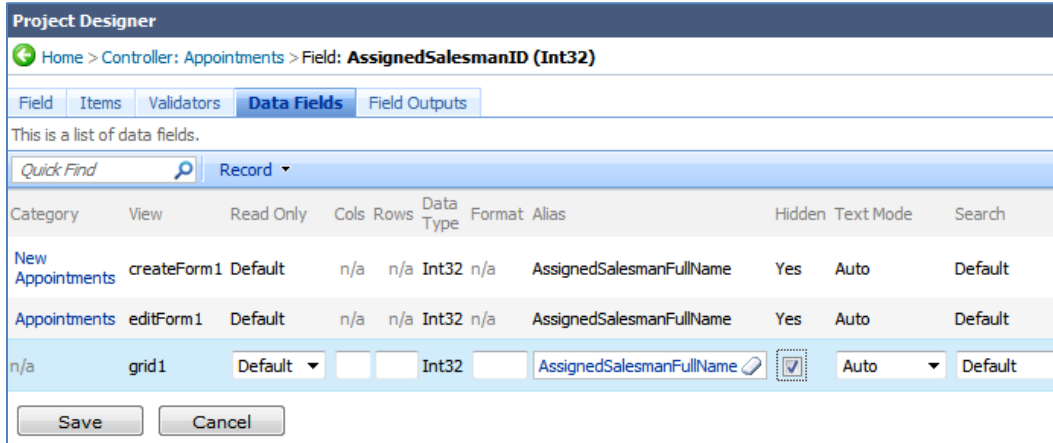
The screenshot shows the 'Project Designer' window with the 'Data Fields' tab selected. The breadcrumb path is 'Home > Controller: Appointments > Field: AssignedSalesmanDisplayFullName (String)'. Below the breadcrumb are tabs for 'Field', 'Items', 'Validators', 'Data Fields', and 'Field Outputs'. A message reads: 'Please fill this form and click OK button to create a new data field record. Click Cancel to return to the previous screen.' The 'View' dropdown is set to 'New Data Field'. A note states '\* - indicates a required field'. The 'New Data Field' section contains the following fields: 'View \*' with the value 'editForm1', 'Category' with the value 'Appointments', and 'Alias' with the value 'AssignedSalesmanFullName'. 'OK' and 'Cancel' buttons are visible at the top right.

Press the *Ok* button to save the new data field. Create one last data field by pressing *New* on the action bar, then *New Data Field*. This will have *View* field as "createForm1", *Category* field as "New Appointments", and *Alias* field as "AssignedSalesmanFullName".

The screenshot shows the 'Project Designer' window with the 'Data Fields' tab selected. The breadcrumb path is 'Home > Controller: Appointments > Field: AssignedSalesmanDisplayFullName (String)'. Below the breadcrumb are tabs for 'Field', 'Items', 'Validators', 'Data Fields', and 'Field Outputs'. A message reads: 'Please fill this form and click OK button to create a new data field record. Click Cancel to return to the previous screen.' The 'View' dropdown is set to 'New Data Field'. A note states '\* - indicates a required field'. The 'New Data Field' section contains the following fields: 'View \*' with the value 'createForm1', 'Category' with the value 'New Appointments', and 'Alias' with the value 'AssignedSalesmanFullName'. 'OK' and 'Cancel' buttons are visible at the top right.

Press *Ok* to save.

Go back to the fields list by pressing the back icon in the upper left corner. Click on the field *AssignedSalesmanID*. Go to the *Data Fields* tab. Mouse over the *New Appointments* link in the first column, activate the dropdown, and press *Edit*. Change the *Hidden* field to “Yes”. Do this for all three data fields, so that the original field *AssignedSalesmanID* will not be displayed in any of the pages.



Press the *Exit* link in the upper right corner to go back to the Generator. Press *Generate* to initiate generation. When the Generator finishes, a webpage will open with the freshly generated application.

Go to the *Appointments* page. Press *New Appointments* on the action bar. You can see that *Assigned Salesman Full Name* field is no longer editable. If you press *Ok*, the assigned salesman will be transferred into the *Salesman Full Name* field, unless you modified that field.

