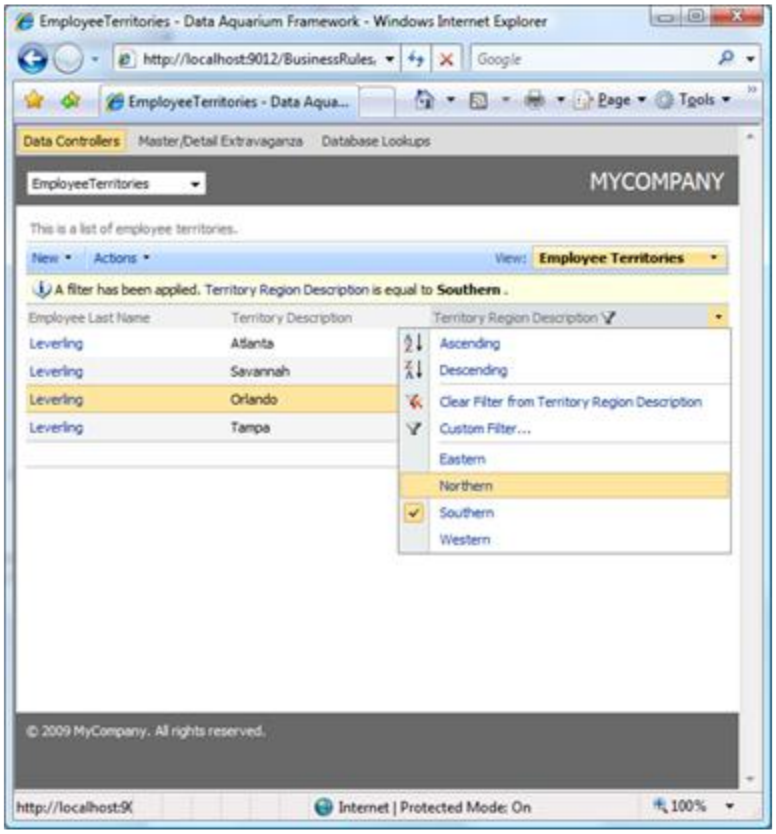
Business Rules: RowBuilder Attribute and Existing Rows

Northwind database has a cross-reference table *EmployeeTerritories* that link together an employee and a territory. Here is a user interface generated for this table by Data Aquarium premium project.

List of employee territories is filtered by a region.

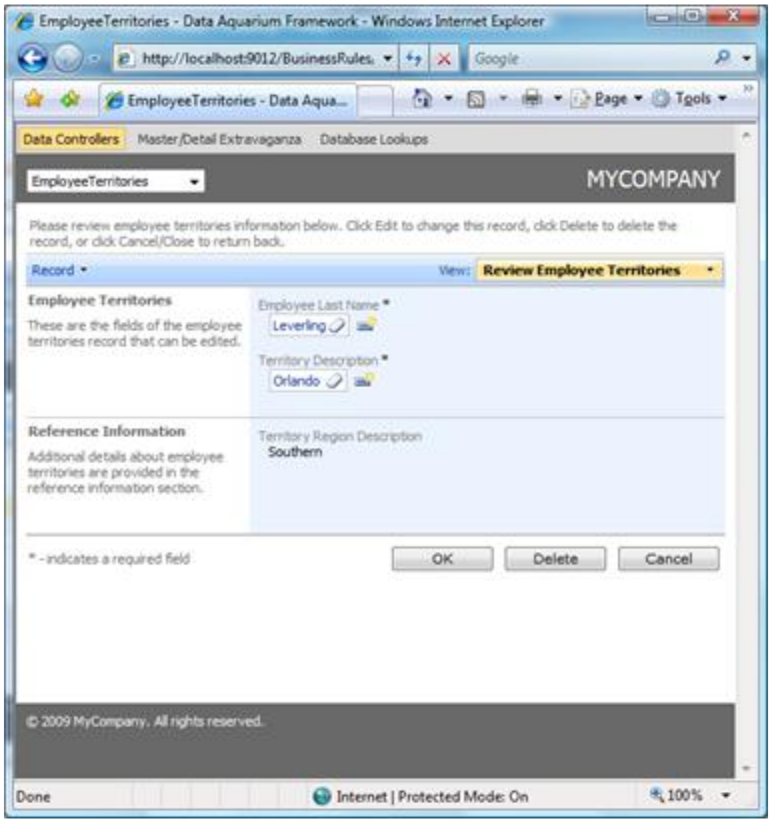

EmployeeTerritories - Data Aquarium Framework - Windows Internet Explorer
http://localhost:9012/BusinessRules.
Google
EmployeeTerritories - Data Aqua...
Page
Tools
Data Controllers
Master/Detail Extravaganza
Database Lookups
EmployeeTerritories
MYCOMPANY
New
Actions
View:
Employee Territories
Ascending
Descending
Clear Filter from Territory Region Description
Custom Filter...
Eastern
Northern
Southern
Western
© 2009 MyCompany. All rights reserved.
http://localhost:9(
Internet | Protected Mode: On
100%

This is a list of employee territories.

A filter has been applied. Territory Region Description is equal to **Southern**.

Employee Last Name	Territory Description	Territory Region Description
Leverling	Atlanta	
Leverling	Savannah	
Leverling	Orlando	
Leverling	Tampa	

Employee territory is displayed in edit form.



EmployeeTerritories - Data Aquarium Framework - Windows Internet Explorer

http://localhost:9012/BusinessRules.

Data Controllers | Master/Detail Extravaganza | Database Lookups

EmployeeTerritories

MYCOMPANY

Please review employee territories information below. Click Edit to change this record, click Delete to delete the record, or click Cancel/Close to return back.

Record ▾ View: Review Employee Territories

Employee Territories

These are the fields of the employee territories record that can be edited.

Employee Last Name *
Leverling

Territory Description *
Orlando

Reference Information

Additional details about employee territories are provided in the reference information section.

Territory Region Description
Southern

* - indicates a required field

OK | Delete | Cancel

© 2009 MyCompany. All rights reserved.

Done | Internet | Protected Mode: On | 100%

While completely function this type of user interface will still benefit if we display a collection of all territories right on the employee screen and allow user to check mark territories that is user is responsible for.

CREATING A PLACEHOLDER FIELD FOR A LIST OF TERRITORIES

Open *~/Controllers/Employees.xml* data controller and modify the text of *command1* as shown below. Notice the new field *Territories* just before the *from* clause. This field is used just as placeholder and an actual value is going to be provided by a business rule.

```
        <command id="command1" type="Text">
            <text>
                <![CDATA[
select
    "Employees"."EmployeeID" "EmployeeID"
    ,"Employees"."LastName" "LastName"
    ,"Employees"."FirstName" "FirstName"
    ,"Employees"."Title" "Title"
    ,"Employees"."TitleOfCourtesy" "TitleOfCourtesy"
    ,"Employees"."BirthDate" "BirthDate"
    ,"Employees"."HireDate" "HireDate"
    ,"Employees"."Address" "Address"
    ,"Employees"."City" "City"
    ,"Employees"."Region" "Region"
    ,"Employees"."PostalCode" "PostalCode"
    ,"Employees"."Country" "Country"
    ,"Employees"."HomePhone" "HomePhone"
    ,"Employees"."Extension" "Extension"
    ,"Employees"."Photo" "Photo"
    ,"Employees"."Notes" "Notes"
    ,"Employees"."ReportsTo" "ReportsTo"
    ,"ReportsTo"."LastName" "ReportsToLastName"
    ,"Employees"."PhotoPath" "PhotoPath",
    ,null "Territories"
from "dbo"."Employees" "Employees"
    left join "dbo"."Employees" "ReportsTo" on "Employees"."ReportsTo" =
"ReportsTo"."EmployeeID"
]]>
            </text>
        </command>
```

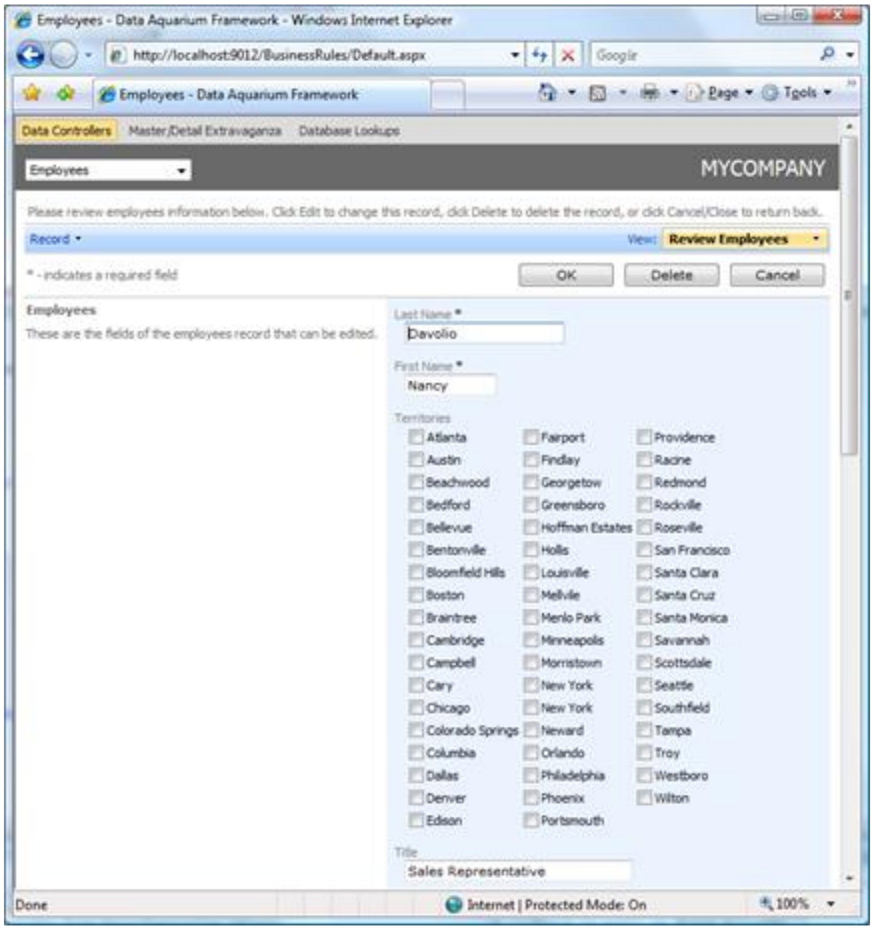
Add new field to the list of fields.

```
<field name="Territories" type="String">
    <items style="CheckBoxList" dataController="Territories"
        dataTextField="TerritoryDescription"/>
</field>
```

Modify *editForm1* to include a reference to the field in the user interface of the form right after the *FirstName* field. We have specified 3 as number of columns for the field, which is supposed to be displayed as a check box list.

```
<dataField fieldName="LastName" columns="20" />
<dataField fieldName="FirstName" columns="10" />
<dataField fieldName="Territories" columns="3"/>
```

Run the sample application and start editing any employee record in form mode. Here is what you will likely see.



Employees - Data Aquarium Framework - Windows Internet Explorer

http://localhost:9012/BusinessRules/Default.aspx

Data Controllers | Master/Detail Extravaganza | Database Lookups

Employees | MYCOMPANY

Please review employees information below. Click Edit to change this record, click Delete to delete the record, or click Cancel/Close to return back.

Record | View: Review Employees

* - indicates a required field

OK | Delete | Cancel

Employees

These are the fields of the employees record that can be edited.

Last Name *: Davolio

First Name *: Nancy

Territories

☐ Atlanta	☐ Fairport	☐ Providence
☐ Austin	☐ Findlay	☐ Racine
☐ Beachwood	☐ Georgetown	☐ Redmond
☐ Bedford	☐ Greensboro	☐ Rockville
☐ Bellevue	☐ Hoffman Estates	☐ Roseville
☐ Bentonville	☐ Hollis	☐ San Francisco
☐ Bloomfield Hills	☐ Louisville	☐ Santa Clara
☐ Boston	☐ Melville	☐ Santa Cruz
☐ Braintree	☐ Menlo Park	☐ Santa Monica
☐ Cambridge	☐ Minneapolis	☐ Savannah
☐ Campbell	☐ Morristown	☐ Scottsdale
☐ Cary	☐ New York	☐ Seattle
☐ Chicago	☐ New York	☐ Southfield
☐ Colorado Springs	☐ Neward	☐ Tampa
☐ Columbia	☐ Orlando	☐ Troy
☐ Dallas	☐ Philadelphia	☐ Westboro
☐ Denver	☐ Phoenix	☐ Wilton
☐ Edison	☐ Portsmouth	

Title: Sales Representative

Done | Internet | Protected Mode: On | 100%

None of the check boxes is checked. We will add a business rule to populate the check boxes.

CREATING A BUSINESS RULE TO POPULATE AN EXISTING ROW

Open business rules class ~/App_Code/Class1.cs(.vb) that was created as described in the RowBuilder attribute introduction. Add the following method to the class.

C#:

```
[RowBuilder("Employees", "editForm1", RowKind.Existing)]
protected void PrepareExistingEmployeeRow()
{
    int employeeId = Convert.ToInt32(SelectFieldValue("EmployeeID"));
    List<EmployeeTerritories> territories =
        EmployeeTerritories.Select(employeeId, null, null, null, null);
    StringBuilder sb = new StringBuilder();
    foreach (EmployeeTerritories et in territories)
    {
        if (sb.Length > 0)
            sb.Append(",");
        sb.Append(et.TerritoryID);
    }
    UpdateFieldValue("Territories", sb.ToString());
}
```

VB:

```
Protected Sub PrepareExistingEmployeeRow()
    Dim employeeId As Integer =
Convert.ToInt32(SelectFieldValue("EmployeeID"))
    Dim territories As List(Of EmployeeTerritories) = _
        EmployeeTerritories.Select(employeeId, Nothing, Nothing, Nothing,
Nothing)
    Dim sb As StringBuilder = New StringBuilder()
    For Each et As EmployeeTerritories In territories
        If sb.Length > 0 Then
            sb.Append(",")
        End If
        sb.Append(et.TerritoryID)
    Next
    UpdateFieldValue("Territories", sb.ToString())
End Sub
```

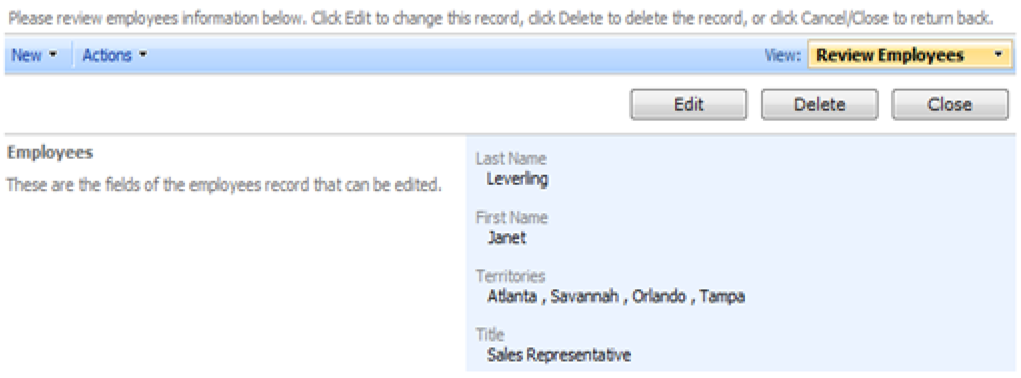
The name of the business rule method is irrelevant. The rule will be automatically invoked when data controller *Employees* is preparing data for *editForm1* view and the form will be displaying an existing row. This method is called for each row returned to the client. Data Aquarium Framework only returns the exact number of rows that are requested by a client view.

Method *SelectFieldValue* is inherited from the base class *BusinessRules* and allows to access values that will be returned for a row that is being built at this moment. We are obtaining a list of employee territories via business object *EmployeeTerritories*.

Next we are creating a comma-separated list of territory IDs and update the value of the *Territories* field with the result accumulated in an instance of *System.Text.StringBuilder*. Please make sure to add *System.Text* namespace in the list of imported namespaces.

Lookup item style *CheckBoxList* is designed to automatically handle comma separated lists of values.

Here is how *editForm1* looks when we select a row. Text corresponding to each territory of selected employee is automatically matched to an ID of each employee territory.


Please review employees information below. Click Edit to change this record, click Delete to delete the record, or click Cancel/Close to return back.

Employees

These are the fields of the employees record that can be edited.

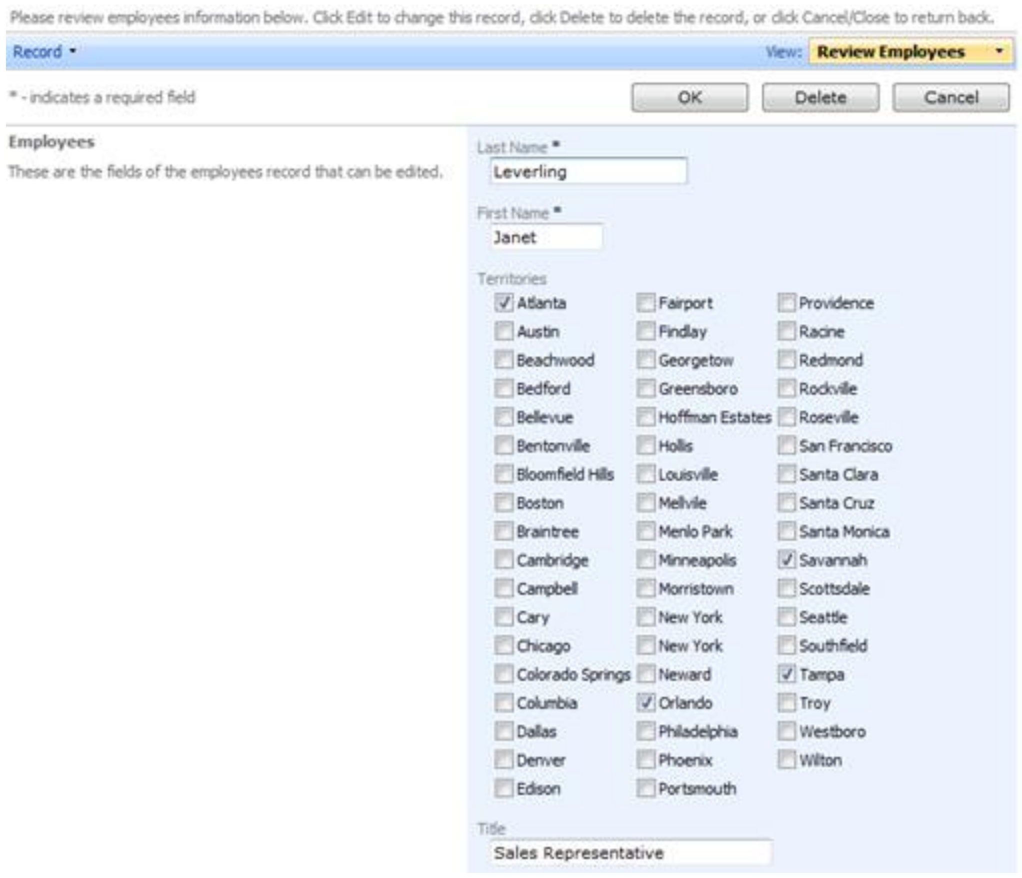
Last Name
Leverling

First Name
Janet

Territories
Atlanta , Savannah , Orlando , Tampa

Title
Sales Representative

Here is how eidtForm1 is transformed when user clicks on *Edit* button.


Please review employees information below. Click Edit to change this record, click Delete to delete the record, or click Cancel/Close to return back.

Record ▾ View: Review Employees ▾

* - indicates a required field OK Delete Cancel

Employees

These are the fields of the employees record that can be edited.

Last Name *: Leverling

First Name *: Janet

Territories

[✓ Atlanta]	Fairport	Providence
Austin	Findlay	Racine
Beachwood	Georgetow	Redmond
Bedford	Greensboro	Rockville
Bellevue	Hoffman Estates	Roseville
Bentonville	Hollis	San Francisco
Bloomfield Hills	Louisville	Santa Clara
Boston	Mellvile	Santa Cruz
Braintree	Menlo Park	Santa Monica
Cambridge	Minneapolis	[✓ Savannah]
Campbell	Morristown	Scottsdale
Cary	New York	Seattle
Chicago	New York	Southfield
Colorado Springs	Neward	[✓ Tampa]
Columbia	[✓ Orlando]	Troy
Dallas	Philadelphia	Westboro
Denver	Phoenix	Wilton
Edison	Portsmouth	

Title: Sales Representative

CONCLUSION

RowBuilder attribute allows providing calculated field values for new and existing rows returned to a client script running in a browser. You can create fields that don't actually exist in your database and figure their values on-the-fly.

There is still work to do when you need to save values of calculated fields. We will review this in the next post dedicated to *ControllerAction* attribute.

Code OnTime LLC

http://www.codeontime.com