# Configuring the SQL Server Provider

If you want to use the SQL Server provider to store application feature data in a SQL Server database, you must first configure SQL Server by creating the appropriate database. ASP.NET includes a command-line utility named aspnet\_regsql.exe that performs this task for you.

The ***aspnet\_regsql.exe*** executable is located in the ***WINDOWS\Microsoft.NET\Framework\versionNumber*** folder on the Web server. The aspnet\_regsql.exe utility is used to both create the SQL Server database and add or remove options from an existing database.

You can run the ***aspnet\_regsql.exe*** executable without any command-line arguments to run a wizard that will help you with specifying connection information for SQL Server and installing or removing the database elements for all supported features. You can also run the aspnet\_regsql.exe executable as a command-line utility to configure database elements for individual features.

To run the ***aspnet\_regsql.exe*** wizard, run the ***aspnet\_regsql.exe*** executable file without any command-line arguments, as shown in the following example.

***[%system root%]\Microsoft.NET\Framework\versionNumber\aspnet\_regsql.exe***

To view online Help for additional options that are available with the aspnet\_regsql.exe utility, use the **/?** option.

# Actions

## JQuery ‘post’ function

### Tariff Controller – Tariff View

<script type="text/javascript">

$(document).ready(function () {

$('#ddlResource').change(function () {

var resourceId = $("#ddlResource").val();

$.post("/Tariff/Tariffs", { "id": resourceId },

function (data) {

var table = $("#tblData > tbody");

table.hide("slow", function () {

table.empty();

for (var i = 0; i < data.length; i++) {

table.append(

'<tr> ' +

'<td class="center">' + data[i].Price +

'</td>' +

'<td class="center">' + FormattedDate(data[i].Date) +

'</td>' +

'<td class="buttonpane\_two">' +

'<form action="/Tariff/Edit/"' + data[i].Id + '" method = "post">' +

'<button class="left img\_button img\_button\_edit"></button>' +

'</form>' +

'<form action="/Tariff/Delete/"' + data[i].Id + '" method = "post">' +

'<button class="right img\_button img\_button\_delete"></button>' +

'</form>' +

'</td>' +

'</tr>');

}

table.show("slow");

});

});

});

});

</script>

### Tariff Controller – Tariff (Post) Method

[HttpPost]

public ActionResult Tariffs(int id)

{

return Json(\_energyEntities.Tariffs.

Where(i => i.IdfResource == id).

OrderByDescending(i => i.Date).

Select(i => new { Id = i.Id, Price = i.Price, Date = i.Date }));

}

## Razor ‘BeginForm’ directive: HTML Helpers

### Tariff Controller – Edit (Get) Method

public ActionResult Edit(int id = 0)

{

Tariff tariff = \_energyEntities.Tariffs.Single(t => t.Id == id);

if (tariff == null)

{

return HttpNotFound();

}

ViewBag.ResourceIdf = new SelectList(\_energyEntities.Resources, "Id", "Name", tariff.IdfResource);

return View(tariff);

}

### Tariff Controller – Edit View

@using (Html.BeginForm()) {

@Html.ValidationSummary(true)

<fieldset>

<legend>Tariff</legend>

@Html.HiddenFor(model => model.Id)

<div class="editor-label">

@Html.LabelFor(model => model.Price)

</div>

<div class="editor-field">

@Html.EditorFor(model => model.Price)

@Html.ValidationMessageFor(model => model.Price)

</div>

<div class="editor-label">

@Html.LabelFor(model => model.Date)

</div>

<div class="editor-field">

@Html.EditorFor(model => model.Date)

@Html.ValidationMessageFor(model => model.Date)

</div>

<p>

<input type="submit" value="Save" />

</p>

</fieldset>

}

### Tariff Controller – Edit (Post) Method

[HttpPost]

public ActionResult Edit(Tariff tariff)

{

if (ModelState.IsValid)

{

\_energyEntities.Tariffs.Attach(tariff);

\_energyEntities.ObjectStateManager.ChangeObjectState(tariff, EntityState.Modified);

\_energyEntities.SaveChanges();

return RedirectToAction("Index");

}

ViewBag.ResourceIdf = new SelectList(\_energyEntities.Resources, "Id", "Name", tariff.IdfResource);

return View(tariff);

}

## Razor ‘BeginForm’ directive: HTML Elements

### Home Controller – Create (Get) Method

public ActionResult Create()

{

Article article = new Article() { EventDate = DateTime.Now, ExpiryDate = DateTime.Now.AddDays(10) };

return View(article);

}

### Home Controller – Create View

@using (Html.BeginForm("Create", "Home", FormMethod.Post, new { enctype = "multipart/form-data" }))

{

<div class="body-title row">

<div class="col-xs-12 title reflected">Breaking News</div>

</div>

<div class="body-body">

<div class="row">

<div class="col-xs-12 form-group">

<label class="control-label" for="news\_title">@Html.DisplayNameFor(model => model.Title)</label>

<input type="text" class="form-control" id="news\_title" name="Title" value="@Model.Title"/>

@Html.ValidationMessageFor(model => model.Title)

</div>

</div>

<div class="body-footer row">

<div class="col-xs-12">

<input type="submit" name="submit" value="Save" class="button shadowed rounded redirect"/>

@Html.ActionLink("Cancel", "Manager", new object { }, new { @class = "button rounded shadowed" })

</div>

</div>

}

### Home Controller – Create (Post) Method

[HttpPost]

public ActionResult Create(Article article, HttpPostedFileBase file)

{

if (ModelState.IsValid)

{

\_entities.Articles.AddObject(article);

\_entities.SaveChanges();

FtpServer server = new FtpServer();

article.Picture = server.Upload(article.Id, file);

\_entities.ObjectStateManager.ChangeObjectState(article, EntityState.Modified);

\_entities.SaveChanges();

return RedirectToAction("Manager");

}

return View(article);

}

## Razor ‘ActionLink’ directive

@Html.ActionLink("Back to List", "Index")

@Html.ActionLink("... more", "Article", new {Id = article.Id})

@Html.ActionLink("News Manager", "Manager", new object {}, new { @class = "button rounded shadowed" })

## HTML <form> tag

<form action="/Tariff/Edit/"' + Id + '" method = "post">

<button class="left img\_button img\_button\_edit"></button>

</form>

## HTML <a> tag

<a href="../Reference/Reference" class="caption redirect">References</a>

<a href="@Url.Action("Article", "Home", new {Id = article.Id})">

# Sections

### Reference Controller – Index View

@section Sidebar

{

<ul class="sidebar left">

<li class="sidebar button"><a href="/">Home</a></li>

<li class="sidebar button"><a href="../Reference/References">References</a></li>

</ul>

}

### Organization Controller – Index View

@section Sidebar

{

<ul class="sidebar left">

<li class="sidebar button"><a href="/">Home</a></li>

<li class="sidebar button"><a href="../Organization/Organizations">Organizations</a></li>

</ul>

}

### \_Layout View

@if (IsSectionDefined("Sidebar"))

{

@RenderSection("Sidebar")

}

# Partial Views

# Helpers

## Truncate

### Script

@helper Truncate(string input, int length)

{

if(input.Length <= length) {

@input

} else {

@input.Substring(0, length)<text> ...</text>

}

}

### HTML

<div class="news-body">@Truncate(article.Content,100)</div>

## Data Annotation

[DisplayName("Event Date")]

[DataType(DataType.Date, ErrorMessage="It should be a date")]

[DisplayFormat(DataFormatString = "{0:MMMM dd, yyyy}")]

[Required(ErrorMessage = "Please fill in this field")]

[StringLength(50, ErrorMessage="Not more than 50 characters long")]