Web Config Manager

# About

I created this library to encapsulate writing and removing of web.config entries.

It is installed to the GAC via a solution, or you may just drop the .dll directly in the GAC.

The solution was created using the Smart Template addin for Visual Studio 2008 ([smarttemplates.codeplex.com](http://smarttemplates.codeplex.com))

It has these public properties that match up with the SPWebConfigModification class.

public string Name { get; set; }

public string XPath { get; set; }

public string Value { get; set; }

public SPWebConfigModification.SPWebConfigModificationType ModificationType { get; set; }

You may also set them directly in the constructor

WebConfigManager(string name, string xpath, string value, SPWebConfigModification.SPWebConfigModificationType modificationType)

There is one public method that creates the SPWebConfigModification from the internal data and the owner & sequence that are passed in.

public SPWebConfigModification GetModification(string owner,uint sequence)

There are two static methods that are used to add & remove the entries from the web.config. Typically, you would call the add on the feature activate and the remove on the feature deactivate.

public static void AddConfigModifications(SPWebApplication webApp, string owner, List<WebConfigManager> modifications)

public static List<SPWebConfigModification> RemoveConfiguration(SPWebApplication webApp, string owner)

# Example Usage:

using System;

using System.Collections.Generic;

using System.Text;

using Microsoft.SharePoint;

using Microsoft.SharePoint.Administration;

namespace MyLocalBroadband.WebConfigManager

{

class ExampleUsageFeatureReceiver : SPFeatureReceiver

{

#region Member Data

private const string \_MODIFICATIONOWNER = "SecurityProvider.SecProviderFeatureReceiver";

private const string \_CONNECTIONSTRINGNAME = "SecProviderConnectionString";

private const string \_CONNECTIONSTRING = @"<add name=""SecProviderConnectionString"" connectionString=""data source=127.0.0.1;Integrated Security=SSPI;Initial Catalog=SecProviderDB"" />";

#endregion

#region Public Overrides

public override void FeatureActivated(SPFeatureReceiverProperties properties)

{

SPWebApplication webApp = properties.Feature.Parent as SPWebApplication;

if (webApp != null)

{

WebConfigManager.AddConfigModifications(webApp, \_MODIFICATIONOWNER, GetWebConfigMods());

}

}

public override void FeatureDeactivating(SPFeatureReceiverProperties properties)

{

SPWebApplication webApp = properties.Feature.Parent as SPWebApplication;

if (webApp != null)

{

WebConfigManager.RemoveConfiguration(webApp, \_MODIFICATIONOWNER);

}

}

public override void FeatureInstalled(SPFeatureReceiverProperties properties)

{

// No Implementation

}

public override void FeatureUninstalling(SPFeatureReceiverProperties properties)

{

// No Implementation

}

#endregion

#region Private Methods

private List<WebConfigManager> GetWebConfigMods()

{

List<WebConfigManager> modifications = new List<WebConfigManager>();

WebConfigManager mod = new WebConfigManager();

mod.Name = "connectionStrings";

mod.XPath = "configuration";

mod.Value = "connectionStrings";

mod.ModificationType = SPWebConfigModification.SPWebConfigModificationType.EnsureSection;

modifications.Add(mod);

mod = new WebConfigManager();

mod.Name = "add[@name='" + \_CONNECTIONSTRINGNAME + "']";

mod.XPath = "configuration/connectionStrings";

mod.Value = \_CONNECTIONSTRING;

mod.ModificationType = SPWebConfigModification.SPWebConfigModificationType.EnsureChildNode;

modifications.Add(mod);

return modifications;

}

#endregion

}

}

# Author

Ryan L. Miller is a SharePoint Developer/Architect in Columbus, Ohio.

[rmiller@mylocalbroadband.com](mailto:rmiller@mylocalbroadband.com)

Company: [www.mylocalbroadband.com](http://www.mylocalbroadband.com)

Blog: [www.intranoggin.com](http://www.intranoggin.com)