Registering custom providers, helpers or managers through your application's **web.config** file allows you to switch between different providers (custom or default) without having to edit the web project's code.

If you do not need your provider customizations to be adjustable through the web.config file, it is more efficient to register custom providers directly via the **RegisterCustomProvider**, **RegisterCustomHelper** or **RegisterCustomManager** assembly attributes (as described in Registering providers using assembly attributes).

1. Edit your web.config file and add the following section into the **<configSections>** element:

   ```
   <!-- Extensibility BEGIN -->
     <section name="cms.extensibility" type="CMS.Base.CMSExtensibilitySection, CMS.
   Base" />
   <!-- Extensibility END -->
   ```

   This defines the ***cms.extensibility*** web.config section where you can register custom providers, helpers and managers.

2. Create the actual **<cms.extensibility>** section directly under the root level of the web.config.
3. Assign your custom classes to providers, helpers or managers via **<add>** elements with the following attributes:

   - **name** - must match the name of the provider/helper/manager class that you wish to customize.
   - **assembly** - specifies the assembly where your custom class is located. Set the value to App_Code for classes placed in the App_Code folder (even on web application installations, where the actual name of the folder is Old_App_Code).
   - **type** - specifies the name of your custom class (including namespaces). If the class is in the App_Code folder, the type value must match the name parameter of the *RegisterCustomClass* attribute used to load the class.

You need to categorize the **<add>** elements under **<providers>**, **<helpers>** or **<managers>** sub-sections based on the type of the customized class.

```
<configuration>
...
<!-- Extensibility BEGIN -->
<cms.extensibility>
    <providers>
        <add name="EmailProvider" assembly="App_Code" type="CustomEmailProvider" />
        <add name="SiteInfoProvider" assembly="App_Code" type="CustomSiteInfoProvider"
/>
                 <add name="ForumPostInfoProvider" assembly="CMS.CustomProviders"
type="CMS.CustomProviders.CustomForumPostInfoProvider" />
        ...
    </providers>
    <helpers>
        <add name="CacheHelper" assembly="App_Code" type="CustomCacheHelper" />
        ...
    </helpers>
    <managers>
        <add name="WorkflowManager" assembly="App_Code" type="CustomWorkflowManager" />
        ...
    </managers>
</cms.extensibility>
<!-- Extensibility END -->
...

</configuration>
```

In the case of custom providers placed in a separate assembly, the registration is now complete. When using the web.config to register custom providers defined in the **App_Code** folder, you also need to ensure that the system can load the classes.

Loading App_Code classes

1. Edit the App_Code class containing your custom provider (repeat for all classes).
2. Add a **RegisterCustomClass** assembly attribute above the class declaration for every custom provider, helper or manager (requires a reference to the **CMS.Base** namespace).

```
using CMS;

[assembly: RegisterCustomClass("CustomEmailProvider", typeof
(CustomEmailProvider))]
[assembly: RegisterCustomClass("CustomSiteInfoProvider", typeof
(CustomSiteInfoProvider))]
[assembly: RegisterCustomClass("CustomCacheHelper", typeof(CustomCacheHelper))]
[assembly: RegisterCustomClass("CustomWorkflowManager", typeof
(CustomWorkflowManager))]
```

The **RegisterCustomClass** attribute registers custom **App_Code** classes and makes them available in the system. The attribute accepts two parameters:

- The first parameter is a string identifier representing the name of the provider. The name must match the value of the **ty pe** attribute set for the corresponding add element in the web.config extensibility section.
- The second parameter specifies the type of the custom class as a **System.Type** object.

When the system requests an App_Code provider class registered through the cms.extensibility section of the web.config, the RegisterCustomClass attribute ensures that an instance of the correct class is loaded.