By default, child web applications inherit the configuration of their parent application. This can prevent the child application from starting when certain sections of the **web.config** aren't compatible across applications. For example, if the child application doesn't share the same libraries, handlers, modules, master pages or themes with the parent application. This page describes how you can solve some of the situations resulting from nesting different applications together.

Breaking web.config inheritance in child applications

Use the **location** tag with the **inheritInChildApplications** attribute set to **false** to ensure that the configuration applies to the parent web application only. This can be useful when you don't want to inherit the parent application's settings to the child application. The path attribute set to "." ensures that the location tag covers the default path.

Using the location tag break configuration inheritance

```
<location path="." inheritInChildApplications="false" >
        <!-- Additional settings -->
</location>
```

Example

The following example shows how you can break inheritance for **controls**, **namespaces**, **assemblies**, **modules**, and **handlers** in your web.config file. None of the settings enclosed in the **location** tag with **inheritInChildApplications** set to **false** are inherited by the child applications.

Using the location tag break configuration inheritance

```
<location path="." inheritInChildApplications="false" >
          <system.web>
            <controls>
                        <!-- Removed for brevity -->
                </controls>
                <namespaces>
                        <!-- Removed for brevity -->
                </namespaces>
                <assemblies>
                        <!-- Removed for brevity -->
                </assemblies>
        </system.web>
</location>

<location path="." inheritInChildApplications="false" >
        <system.webServer>
            <modules>
                        <!-- Removed for brevity -->
                </modules>
                <handlers>
                        <!-- Removed for brevity -->
                </handlers>
        </system.webServer>
</location>
```

A different approach to removing inheritance is using the **<remove>** and **<clear>** tags. The tags remove irrelevant modules in the child web application. However, this approach is not supported and implemented the same way across all the elements and items in the web.config file.

Specifying different web.config configurations for child applications

By specifying a value for the **path** attribute in the **location** tag, you can apply different configuration settings to specific folders and files. This way, you can distribute parent or child application settings through one configuration file.

The default value for the **path** attribute is ".". You can only specify one value in the **path** attribute.

```
<location path="Folder1" >
    <section1 ... />
    <section2 ... />
</location>
<location path="Folder2" >
    <section1 ... />
    <section2 ... />
</location>
```

Example

The following example shows a configuration that specifies different settings for two specific folders. One of the folders belongs to a parent application and the nested folder belongs to a child application:

```
<!-- Configuration for 'Default Web Site' -->
<location path="Default Web Site/ParentApplication" >
    <namespaces>
        <add namespace="System.Web.Mvc" />
        <add namespace="System.Web.Mvc.Ajax" />
        <add namespace="System.Web.Mvc.Html" />
        <add namespace="System.Web.Routing" />
      </namespaces>
</location>

<!-- Configuration for 'ChildApplication' -->
<location path="Default Web Site/ParentApplication/ChildApplication" >
      <namespaces>
              <add namespace="System.IO" />
        <add namespace="System.Text" />
      </namespaces>
</location>
```