Importance & Effective Use of Configuration File in Software Development (Window /Web) in .NET

[1]Tanmay Kasbe
[1]Research Scholar in Computer Science & Application
[1] Indore , India

***Abstract*—In this my research paper I will explain in brief about configuration file. As we know most of the technologies uses configuration file but in this paper I am considering only .NET technology. TOP MNC like Microsoft , TCS , Infosys etc and small software companies which working on .NET they have used very effectively configuration file in their projects. This research paper is very useful for those students who working in .NET project either it could be window or web. Initially most of the top software companies will provide training & this part of configuration file is highly effective as a training part. I will not explain only importance of configuration file but also I am providing effective implementation in software development process.**

***Index Terms*—.NET, Configuration file, MNC, Software projects** *(keywords)*

I. INTRODUCTION

The .NET Framework, by the help of configuration files, gives developers and administrators control and flexibility over the way applications run. Configuration files are XML files that can be changed as needed. An administrator can control which protected resources an application can access, which versions of assemblies an application will use, and where remote applications and objects are located. Developers can put settings in configuration files, eliminating the need to recompile an application every time a setting changes. One simple example of config file setting is when we declare a database connection in config file then that connection is used only once in complete project & database connection does not require on every forms.

Files that contain the .config file extension are used by Microsoft .NET framework & are created during application development. The config file used by variety of .NET frameworks. Proper use of config file is not only provide the flexibility but also it reduce the code which required again & again in software development coding & lots of application setting which is very important for operating system as well as when we host our web application in server. The .NET framework provides a rich set of classes, and techniques, to simplify application configuration. Essentially, all of these classes make it easy to read and write that configuration information from an XML configuration file. The configuration file includes a number of standard sections, some custom sections for common .NET features, and also allows the developer to create their own custom configuration sections.

II. TYPES OF CONFIGURATION FILE

There are three types of configuration file in .NET.

A. Application configuration file (app.config)
B. Machine configuration file (machin.config)
C. Web based configuration file (web. Config)

a.) Application Configuration file-

An application configuration file is an XML file used to control assembly binding. It can redirect an application from using one version of a side-by-side assembly to another version of the same assembly. This is called per-application configuration. An application configuration file applies only to a specific application manifest and dependent assemblies. Isolated components compiled with an embedded ISOLATIONAWARE_MANIFEST_RESOURCE_ID manifest require a separate application configuration file. Manifests managed with **CreateActCtx** require a separate application configuration file.

.NET gives an easy way to store configuration information in an Application Configuration File. In the simple implementation, you can store information as Key-Value pairs. For example, consider a case where you have to use a data source in your application. If you hardcore the data source information in your code, you will have a bad time when you have to change this data source. You have to change your source code and re-compile it. This won't work every time you give your product to different customers or when you run your application in different machines!

In earlier days, programmers used to store this information in special files called *.ini* files or in system registry. The application can read the information from the *.ini* file or registry and no need to re-compile the code when the values are changed in *.ini* file or registry. But this is a pain most of the time. It is not fun opening the registry, locate your entries and make appropriate changes. It is quite possible that you may mess up with some important entries in the registry and make your system

not run any more. In fact, in secured systems, administrator may deny access to the registry and users will not have the choice to edit the registry at all.

.NET gives you a simple and easy solution for this problem - the *Application Configuration File*. Each application can have a configuration file, which is actually an XML file. You can use any text editor (including Notepad) to open the configuration file and change the values. The application will load the values from this configuration file and you do not have to change your source code every time you change your data source or any other information stored in the configuration file.

By default, the *app.config* file will have the following content:

```
<?xml version="1.0" encoding="utf-8" ?>
<configuration>
</configuration>
```

To store values in the configuration file, you can create XML elements in the format:

```
<add key="MyKey" value="MyValue" />
```

Example:

```
<?xml version="1.0" encoding=   "utf-8" ?>
<configuration>
<appSettings>
   <add key="DatabasePath" value="c:\\projects\data\tanmay.mdb" />
<add key="Email" value="tanmay.kasbe@gmail.com" />
</appSettings>
</configuration>
```

b.) Machine Configuration file-

The Machin.config file contains settings for all sites running on the machine provided another .config further up the chain does not override any of these settings. Although machin.config provides a global configuration option, you can use .config files inside individual website directories to provide more granular control. Between these two poles you can set a number of other .config files with varying degree of applicable scope.

Configuration files are applied to an executing site based on a hierarchy. There is a global configuration file for all sites in a given machine which is called machin.config. This file is typically found in the C:\WINDOWS\Microsoft.NET\Framework\v2.0.50727\CONFIG directory. Only one machin.config file can exist on a server. Without the machin.config file Application cannot be executed.

c.) Web Configuration file-

Configuration file is used to manage various settings that define a website. The settings are stored in XML files that are separate from your application code. In this way you can configure settings independently from your code. Generally a website contains a single Web.config file stored inside the application root directory. However there can be many configuration files that manage settings at various levels within an application. ASP.NET Configuration system is used to describe the properties and behaviors of various aspects of ASP.NET applications. Configuration files help you to manage the many settings related to your website. Each file is an XML file (with the extension .config) that contains a set of configuration elements. Configuration information is stored in XML-based text file.

An application however, can override the default value by creating web.config files in its roots folder. The web.config file is a subset of the machin.config file. If the application contains child directories, it can define a web.config file for each folder. Scope of each configuration file is determined in a hierarchical top down manner.

There are number of important settings that can be stored in the configuration file. Some of the most frequently used configurations, stored conveniently inside Web.config file are:

A. Database connections
B. Caching settings
C. Session States

D. Error Handling
E. Security

Basic Syntax of web.config are:

```
<configuration>

  <!-- Configuration section-handler declaration area. -->
    <configSections>
      <section name="section1" type="section1Handler" />
      <section name="section2" type="section2Handler" />
    </configSections>
  <!-- Configuration section settings area. -->

  <section1>
    <s1Setting1 attribute1="attr1" />
  </section1>
<section2>

    <s2Setting1 attribute1="attr1" />
  </section2>
  <system.web>
    <authentication mode="Windows" />
  </system.web>
</configuration>
```

a.) Authentication-

It configures the authentication support. The basic syntax is as given:

```
<authentication mode="[Windows|Forms|Passport|None]">
  <forms>...</forms>
  <passport/>
</authentication>
```

b.) Authorization-

It configures the authorization support. The basic syntax is as given:

```
<authorization>
  <allow .../>
  <deny .../>
</authorization>
```

c.) Caching

It Configures the cache settings. The basic syntax is as given:

```
<caching>
  <cache>...</cache>
  <outputCache>...</outputCache>
  <outputCacheSettings>...</outputCacheSettings>
  <sqlCacheDependency>...</sqlCacheDependency>
</caching>
```

d.) CustomErrors

It defines custom error messages. The basic syntax is as given:

```
<customErrors defaultRedirect="url" mode="On|Off|RemoteOnly">
  <error. . ./>
</customErrors>
```

e.) Deployment

It defines configuration settings used for deployment. The basic syntax is as follows:

```
<deployment retail="true|false" />
```

f.) Hosting Environment

It defines configuration settings for hosting environment. The basic syntax is as follows:

```
<hostingEnvironment idleTimeout="HH:MM:SS" shadowCopyBinAssemblies="true|false"
shutdownTimeout="number" urlMetadataSlidingExpiration="HH:MM:SS" />
```

g.) Identity

It configures the identity of the application. The basic syntax is as given:

```
<identity impersonate="true|false" userName="domain\username"
  password="<secure password>"/>
```

h.) MachineKey

It configures keys to use for encryption and decryption of Forms authentication cookie data.
It also allows configuring a validation key that performs message authentication checks on view-state data and forms authentication tickets. The basic syntax is:

```
<machineKey validationKey="AutoGenerate,IsolateApps" [String]
  decryptionKey="AutoGenerate,IsolateApps" [String]
  validation="HMACSHA256" [SHA1 | MD5 | 3DES | AES | HMACSHA256 |
  HMACSHA384 | HMACSHA512 | alg:algorithm_name]
  decryption="Auto" [Auto | DES | 3DES | AES | alg:algorithm_name]
                                        />
```

i.) Membership

This configures parameters of managing and authenticating user accounts. The basic syntax is:

```
<membership defaultProvider="provider name"
  userIsOnlineTimeWindow="number of minutes" hashAlgorithmType="SHA1">
  <providers>...</providers>
</membership>
```

j.) Profile

It configures user profile parameters. The basic syntax is:

```
<profile enabled="true|false" inherits="fully qualified type reference"
  automaticSaveEnabled="true|false" defaultProvider="provider name">

  <properties>...</properties>
  <providers>...</providers>
  </profile>
```

k.) Role Manager

It configures settings for user roles. The basic syntax is:

```
<roleManager cacheRolesInCookie="true|false" cookieName="name"
  cookiePath="/" cookieProtection="All|Encryption|Validation|None"
  cookieRequireSSL="true|false " cookieSlidingExpiration="true|false "
  cookieTimeout="number of minutes" createPersistentCookie="true|false"
  defaultProvider="provider name" domain="cookie domain">
  enabled="true|false"
  maxCachedResults="maximum number of role names cached"
```

```
  <providers>...</providers>
</roleManager>
```

l.) Security Policy

It configures the security policy. The basic syntax is:

```
<securityPolicy>
  <trustLevel />
</securityPolicy>
```

m.) Url Mappings

It defines mappings to hide the original URL and provide a more user friendly URL. The basic syntax is:

```
<urlMappings enabled="true|false">
  <add.../>
  <clear />
  <remove.../>
</urlMappings>
```

n.) Web Controls

It provides the name of shared location for client scripts. The basic syntax is:

```
<webControls clientScriptsLocation="String" />
```

III. IMPORTANCE OF CONFIGURATION FILE

As above mention all types of configuration file are very important while developing any type of software, either it is web based application or window based application. If we utilize the entire attribute in efficient way then we must developed good software. In .NET configuration file is very important to know all attribute & what are their uses.

IV. CONCLUSION

I have written this paper to keep lots of important point related configuration file. Most of the student & also freelancers does not know about configuration file. May be they know about configuration file but they didn't know how to effective use. Configuration file is highly effective while developing software's but it is also recommending that some of the attribute running well when we work on local machine but when we deploy our application into server or client machine then same attribute not works so we must change that attributes.

V. ACKNOWLEDGEMENT

I would like to thank all the software engineer who written articles on configuration file. I mixed their experience & my experience to write this paper.

I express regret if any reference has been missed out.

REFERENCES

[1] Puran Mehra, ASP.NET web configuration file.
http://www.c-sharpcorner.com/uploadfile/puranindia/Asp-Net-web-configuration-file/

[2] T Manjaly, Configuration setting file for providing application configuration data
http://www.codeproject.com/Articles/6538/Configuration-Settings-File-for-providing-applicat

[3] MSDN , Configuration Apps by using configuration files
http://www.codeproject.com/Articles/6538/Configuration-Settings-File-for-providing-applicat

[4] http://www.tutorialspoint.com/asp.net/asp.net_configuration.htm