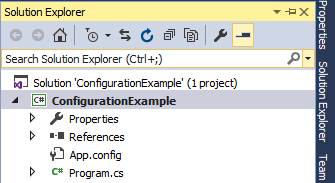
This article will demonstrate how we can get/read the configuration setting from Web.Config or App.Config in C#. There are different ways to set the values inside the configuration file and read their values, which are based on the defined keys. We define those values inside the configuration section, which might be needed to make it more secure. It can be some secret keys or the value, which should be received frequently.

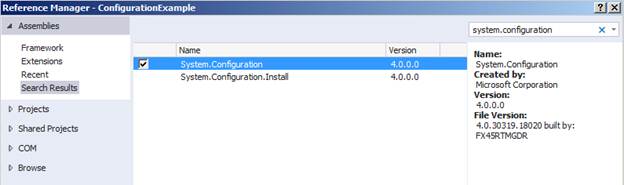
Today, I will show you four different ways to get the values from the configuration section.

For this demonstration, I am going to create a simple console Application and provide the name as “ConfigurationExample”. Just create one console Application, as shown below.

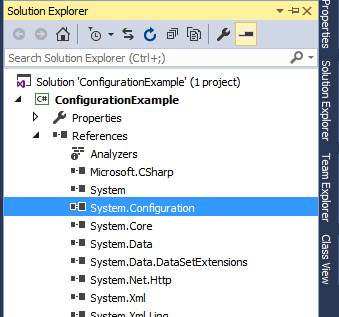
**New Project > Visual C# > Console Application**



We need to add System.Configuration assembly reference to access configuration settings, using ConfigurationManager. To add the reference, just right click References and click to add references.



Now, we can see that System.Configuration reference has been added successfully to our project.



Thus, let’s move to different ways to add the values inside the config file and the approach we follow to get it.

First approach

Let’s take one example, where we need to add some Application level settings and access them based on their keys. We can add these settings either inside Web.Config or App.Config but we need to add <appSettings> section inside the configuration section.

Just follow the example given below, where inside the appSettings section; we have defined few keys and their values.

**App.config**

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

<configuration>

<startup>

<supportedRuntime version="v4.0" sku=".NETFramework,Version=v4.5" />

</startup>

<appSettings>

<add key="Title" value="Configuration Example"/>

<add key="Language" value="CSharp"/>

</appSettings>

</configuration>

Markup

Copy

To access these values, there is one static class named ConfigurationManager, which has one getter property named AppSettings. We can just pass the key inside the AppSettings and get the desired value from AppSettings section, as shown below.

public static void GetConfigurationValue()

{

var title = ConfigurationManager.AppSettings["title"];

var language = ConfigurationManager.AppSettings["language"];

Console.WriteLine(string.Format("'{0}' project is created in '{1}' language ", title, language));

}

C#

Copy

When we implement the code given above, we get the output, as shown below.

Four Ways To Read Configuration Setting In C#

Second approach

Let’s move to the next example. If we need to add the settings inside the section for the separation, in this situation, we can create a custom section inside the configuration section in App.Config/Web.Config, as shown below. This section can make your data more readable and understandable based on your section name.

In the example given below, we have just created one custom section named ApplicationSettings and added all the key/value pairs separately.

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

<configuration>

<configSections>

<[section](https://webplatform.github.io/docs/html/elements/section) [name](https://webplatform.github.io/docs/html/attributes/name)="ApplicationSettings" [type](https://webplatform.github.io/docs/html/attributes/type)="System.Configuration.NameValueSectionHandler"/>

</[configSections](https://webplatform.github.io/docs/html/elements/configSections" \t "_blank)>

<[ApplicationSettings](https://webplatform.github.io/docs/html/elements/ApplicationSettings" \t "_blank)>

<[add](https://webplatform.github.io/docs/html/elements/add) [key](https://webplatform.github.io/docs/html/attributes/key)="ApplicationName" [value](https://webplatform.github.io/docs/html/attributes/value)="Configuration Example Project"/>

<[add](https://webplatform.github.io/docs/html/elements/add) [key](https://webplatform.github.io/docs/html/attributes/key)="Language" [value](https://webplatform.github.io/docs/html/attributes/value)="CSharp"/>

<[add](https://webplatform.github.io/docs/html/elements/add) [key](https://webplatform.github.io/docs/html/attributes/key)="SecretKey" [value](https://webplatform.github.io/docs/html/attributes/value)="012345"/>

</[ApplicationSettings](https://webplatform.github.io/docs/html/elements/ApplicationSettings" \t "_blank)>

<[startup](https://webplatform.github.io/docs/html/elements/startup" \t "_blank)>

<[supportedRuntime](https://webplatform.github.io/docs/html/elements/supportedRuntime" \t "_blank) [version](https://webplatform.github.io/docs/html/attributes/version)="v4.0" [sku](https://webplatform.github.io/docs/html/attributes/sku)=".NETFramework,Version=v4.5" />

</[startup](https://webplatform.github.io/docs/html/elements/startup" \t "_blank)>

</[configuration](https://webplatform.github.io/docs/html/elements/configuration)>

Markup

Copy

To access custom section settings, we first need to find out the section, using GetSection method, which is defined inside the ConfigurationManager class and cast the return value as NameValueCollection. It will return all the keys available inside this custom section and based on the keys, we can get the values easily, as shown below.

//Approach Two

public static void GetConfigurationUsingSection()

{

var applicationSettings = ConfigurationManager.GetSection("ApplicationSettings") as NameValueCollection;

if (applicationSettings.Count == 0)

{

Console.WriteLine("Application Settings are not defined");

}

else

{

foreach (var key in applicationSettings.AllKeys)

{

Console.WriteLine(key + " = " + applicationSettings[key]);

}

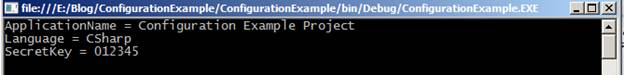
}

}

C#

Copy

When we implement the code given abbove, we get the output given below.



Third approach

Now, let's move to some tough stuff. Here, we are going to create a section inside the group, so that, if required, we can add multiple sections in the same group. It is basically grouping the same type of section in a group.

In the example given below, we have created one group named BlogGroup and inside it, we have defined one section, named it “PostSetting” and its type as a NameValueSectionHandler. “PostSetting” section contains all the key/value pair separately, as shown below.

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

<[configuration](https://webplatform.github.io/docs/html/elements/configuration)>

<[configSections](https://webplatform.github.io/docs/html/elements/configSections" \t "_blank)>

<[sectionGroup](https://webplatform.github.io/docs/html/elements/sectionGroup" \t "_blank) [name](https://webplatform.github.io/docs/html/attributes/name)="BlogGroup">

<[section](https://webplatform.github.io/docs/html/elements/section) [name](https://webplatform.github.io/docs/html/attributes/name)="PostSetting" [type](https://webplatform.github.io/docs/html/attributes/type)="System.Configuration.NameValueSectionHandler"/>

</[sectionGroup](https://webplatform.github.io/docs/html/elements/sectionGroup" \t "_blank)>

<[section](https://webplatform.github.io/docs/html/elements/section) [name](https://webplatform.github.io/docs/html/attributes/name)="ProductSettings" [type](https://webplatform.github.io/docs/html/attributes/type)="ConfigurationExample.ProductSettings, ConfigurationExample"/>

</[configSections](https://webplatform.github.io/docs/html/elements/configSections" \t "_blank)>

<[BlogGroup](https://webplatform.github.io/docs/html/elements/BlogGroup" \t "_blank)>

<[PostSetting](https://webplatform.github.io/docs/html/elements/PostSetting" \t "_blank)>

<[add](https://webplatform.github.io/docs/html/elements/add) [key](https://webplatform.github.io/docs/html/attributes/key)="PostName" [value](https://webplatform.github.io/docs/html/attributes/value)="Getting Started With Config Section in .Net"/>

<[add](https://webplatform.github.io/docs/html/elements/add) [key](https://webplatform.github.io/docs/html/attributes/key)="Category" [value](https://webplatform.github.io/docs/html/attributes/value)="C#"></[add](https://webplatform.github.io/docs/html/elements/add)>

<[add](https://webplatform.github.io/docs/html/elements/add) [key](https://webplatform.github.io/docs/html/attributes/key)="Author" [value](https://webplatform.github.io/docs/html/attributes/value)="Mukesh Kumar"></[add](https://webplatform.github.io/docs/html/elements/add)>

<[add](https://webplatform.github.io/docs/html/elements/add) [key](https://webplatform.github.io/docs/html/attributes/key)="PostedDate" [value](https://webplatform.github.io/docs/html/attributes/value)="28 Feb 2017"></[add](https://webplatform.github.io/docs/html/elements/add)>

</[PostSetting](https://webplatform.github.io/docs/html/elements/PostSetting" \t "_blank)>

</[BlogGroup](https://webplatform.github.io/docs/html/elements/BlogGroup" \t "_blank)>

<[ProductSettings](https://webplatform.github.io/docs/html/elements/ProductSettings" \t "_blank)>

<[DellSettings](https://webplatform.github.io/docs/html/elements/DellSettings" \t "_blank) [ProductNumber](https://webplatform.github.io/docs/html/attributes/ProductNumber)="20001" [ProductName](https://webplatform.github.io/docs/html/attributes/ProductName)="Dell Inspiron" [Color](https://webplatform.github.io/docs/html/attributes/Color)="Black" [Warranty](https://webplatform.github.io/docs/html/attributes/Warranty)="2 Years" ></[DellSettings](https://webplatform.github.io/docs/html/elements/DellSettings" \t "_blank)>

</[ProductSettings](https://webplatform.github.io/docs/html/elements/ProductSettings" \t "_blank)>

<[startup](https://webplatform.github.io/docs/html/elements/startup" \t "_blank)>

<[supportedRuntime](https://webplatform.github.io/docs/html/elements/supportedRuntime" \t "_blank) [version](https://webplatform.github.io/docs/html/attributes/version)="v4.0" [sku](https://webplatform.github.io/docs/html/attributes/sku)=".NETFramework,Version=v4.5.2"/>

</[startup](https://webplatform.github.io/docs/html/elements/startup" \t "_blank)>

</[configuration](https://webplatform.github.io/docs/html/elements/configuration)>

Markup

Copy

To read these types of configuration settings, we need to access the section. Based on the section group, we can get all the keys and their values, as shown below.

//Approach Three

public static void GetConfigurationUsingSectionGroup()

{

var PostSetting = ConfigurationManager.GetSection("BlogGroup/PostSetting") as NameValueCollection;

if (PostSetting.Count == 0)

{

Console.WriteLine("Post Settings are not defined");

}

else

{

foreach (var key in PostSetting.AllKeys)

{

Console.WriteLine(key + " = " + PostSetting[key]);

}

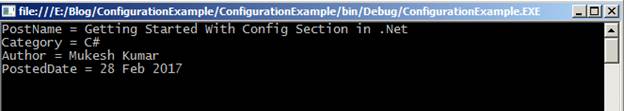
}

}

C#

Copy

When we implement the code given above, we get the output, as shown below.



Fourth approach

At last, we are on an advanced stage of the configuration settings. Sometimes, it is required to set up your all key/value pairs based on the custom class behavior, so that we can control the behavior from outside.

See the following class “DellFeatures”, which shows some custom properties of Dell laptops and we need to add it inside the configuration section. The following class contains some default values, if the value is not available in the configuration section.

using System;

using System.Collections.Generic;

using System.Configuration;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace ConfigurationExample

{

public class DellFeatures : ConfigurationElement

{

[ConfigurationProperty("ProductNumber", DefaultValue = 00000, IsRequired = true)]

public int ProductNumber

{

get

{

return (int)this["ProductNumber"];

}

set

{

value = (int)this["ProductNumber"];

}

}

[ConfigurationProperty("ProductName", DefaultValue = "DELL", IsRequired = true)]

public string ProductName

{

get

{

return (string)this["ProductName"];

}

set

{

value = (string)this["ProductName"];

}

}

[ConfigurationProperty("Color", IsRequired = false)]

public string Color

{

get

{

return (string)this["Color"];

}

set

{

value = (string)this["Color"];

}

}

[ConfigurationProperty("Warranty", DefaultValue = "1 Years", IsRequired = false)]

public string Warranty

{

get

{

return (string)this["Warranty"];

}

set

{

value = (string)this["Warranty"];

}

}

}

}

C#

Copy

To return this setting, we are going to create one more class, which returns this as a property. Here, we can also add multiple classes as the properties.

using System;

using System.Collections.Generic;

using System.Configuration;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace ConfigurationExample

{

public class ProductSettings : ConfigurationSection

{

[ConfigurationProperty("DellSettings")]

public DellFeatures DellFeatures

{

get

{

return (DellFeatures)this["DellSettings"];

}

set

{

value = (DellFeatures)this["DellSettings"];

}

}

}

}

C#

Copy

To implement it inside the configuration section, we are going to change the type of “ProductSettings” as “ConfigurationExample.ProductSettings”, which will return all the properties of DellFeatures class.

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

<[configuration](https://webplatform.github.io/docs/html/elements/configuration)>

<[configSections](https://webplatform.github.io/docs/html/elements/configSections" \t "_blank)>

<[section](https://webplatform.github.io/docs/html/elements/section) [name](https://webplatform.github.io/docs/html/attributes/name)="ProductSettings" [type](https://webplatform.github.io/docs/html/attributes/type)="ConfigurationExample.ProductSettings, ConfigurationExample"/>

</[configSections](https://webplatform.github.io/docs/html/elements/configSections" \t "_blank)>

<[BlogGroup](https://webplatform.github.io/docs/html/elements/BlogGroup" \t "_blank)>

<[PostSetting](https://webplatform.github.io/docs/html/elements/PostSetting" \t "_blank)>

<[add](https://webplatform.github.io/docs/html/elements/add) [key](https://webplatform.github.io/docs/html/attributes/key)="PostName" [value](https://webplatform.github.io/docs/html/attributes/value)="Getting Started With Config Section in .Net"/>

<[add](https://webplatform.github.io/docs/html/elements/add) [key](https://webplatform.github.io/docs/html/attributes/key)="Category" [value](https://webplatform.github.io/docs/html/attributes/value)="C#"></[add](https://webplatform.github.io/docs/html/elements/add)>

<[add](https://webplatform.github.io/docs/html/elements/add) [key](https://webplatform.github.io/docs/html/attributes/key)="Author" [value](https://webplatform.github.io/docs/html/attributes/value)="Mukesh Kumar"></[add](https://webplatform.github.io/docs/html/elements/add)>

<[add](https://webplatform.github.io/docs/html/elements/add) [key](https://webplatform.github.io/docs/html/attributes/key)="PostedDate" [value](https://webplatform.github.io/docs/html/attributes/value)="28 Feb 2017"></[add](https://webplatform.github.io/docs/html/elements/add)>

</[PostSetting](https://webplatform.github.io/docs/html/elements/PostSetting" \t "_blank)>

</[BlogGroup](https://webplatform.github.io/docs/html/elements/BlogGroup" \t "_blank)>

<[ProductSettings](https://webplatform.github.io/docs/html/elements/ProductSettings" \t "_blank)>

<[DellSettings](https://webplatform.github.io/docs/html/elements/DellSettings" \t "_blank) [ProductNumber](https://webplatform.github.io/docs/html/attributes/ProductNumber)="20001" [ProductName](https://webplatform.github.io/docs/html/attributes/ProductName)="Dell Inspiron" [Color](https://webplatform.github.io/docs/html/attributes/Color)="Black" [Warranty](https://webplatform.github.io/docs/html/attributes/Warranty)="2 Years" ></[DellSettings](https://webplatform.github.io/docs/html/elements/DellSettings" \t "_blank)>

</[ProductSettings](https://webplatform.github.io/docs/html/elements/ProductSettings" \t "_blank)>

<[startup](https://webplatform.github.io/docs/html/elements/startup" \t "_blank)>

<[supportedRuntime](https://webplatform.github.io/docs/html/elements/supportedRuntime" \t "_blank) [version](https://webplatform.github.io/docs/html/attributes/version)="v4.0" [sku](https://webplatform.github.io/docs/html/attributes/sku)=".NETFramework,Version=v4.5.2"/>

</[startup](https://webplatform.github.io/docs/html/elements/startup" \t "_blank)>

</[configuration](https://webplatform.github.io/docs/html/elements/configuration)>

Markup

Copy

To access this type of configuration, we need to get the custom section first and the rest of it will be accessible very easily, as shown below.

//Approach Four

public static void GetConfigurationUsingCustomClass()

{

var productSettings = ConfigurationManager.GetSection("ProductSettings") as ConfigurationExample.ProductSettings;

if (productSettings == null)

{

Console.WriteLine("Product Settings are not defined");

}

else

{

var productNumber = productSettings.DellFeatures.ProductNumber;

var productName = productSettings.DellFeatures.ProductName;

var color = productSettings.DellFeatures.Color;

var warranty = productSettings.DellFeatures.Warranty;

Console.WriteLine("Product Number = " + productNumber);

Console.WriteLine("Product Name = " + productName);

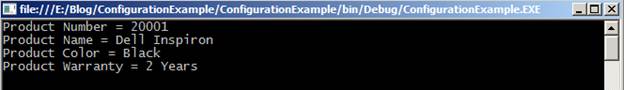
Console.WriteLine("Product Color = " + color);

Console.WriteLine("Product Warranty = " + warranty);

}

}

When we implement the code given above, we get the output, as shown below.



We have seen different ways to define the configuration setting inside the configuration file and access/read it.