

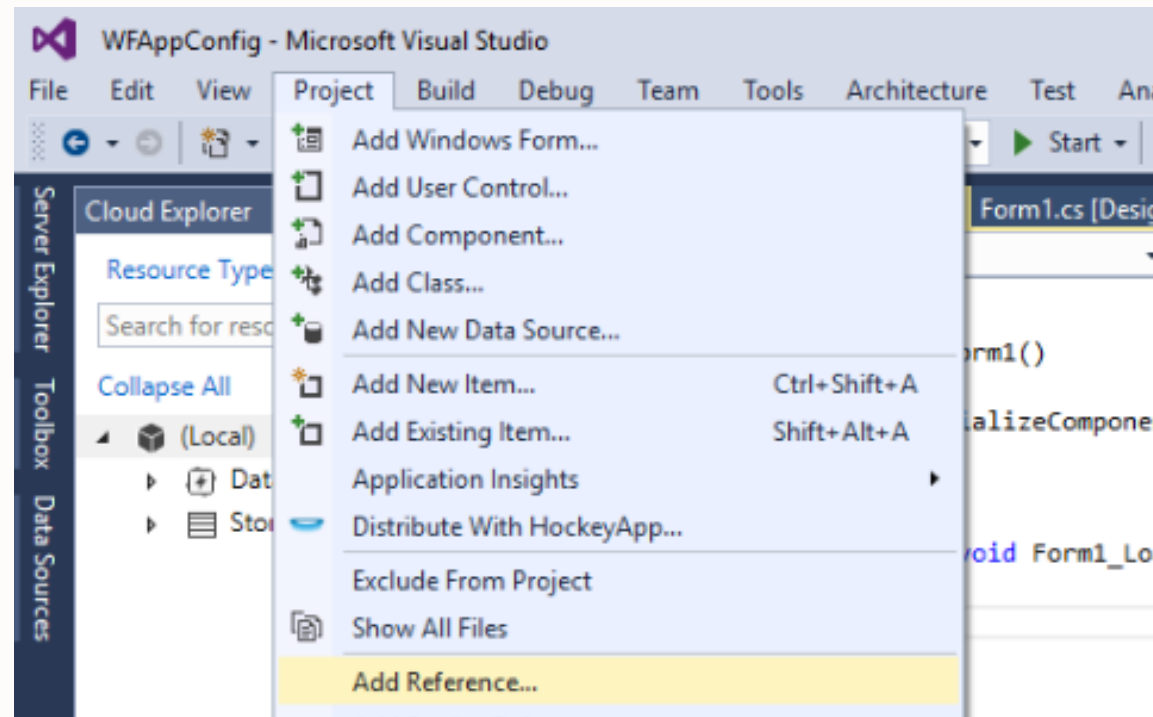
How to work with Configuration Manager class and App.config

Lab 2 Helper

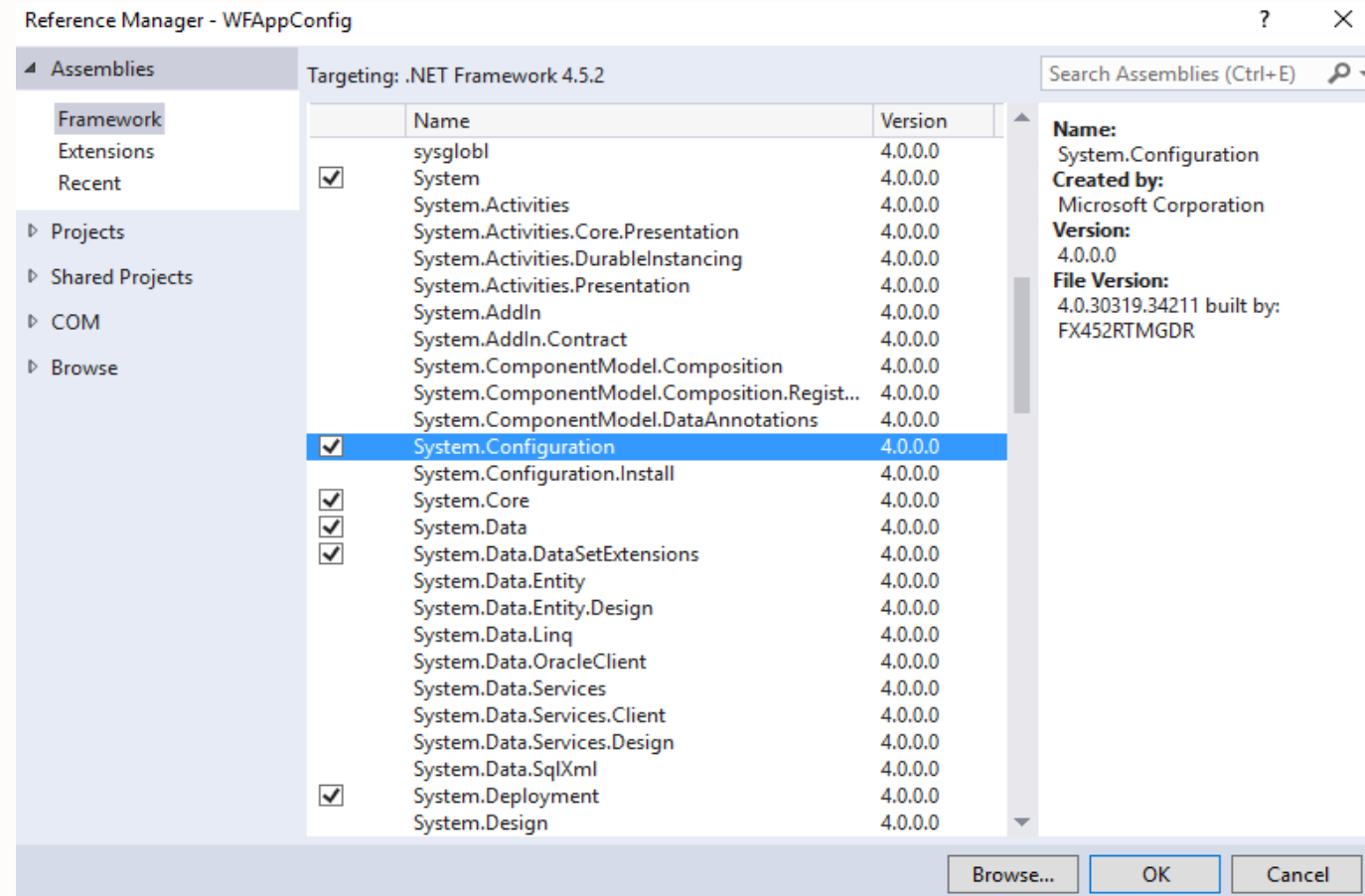
by Camelia Andor

Configuration Manager class and App.config

- If you want to access data from App.config file inside your application, you must first add a reference to the assembly System.Configuration.dll:

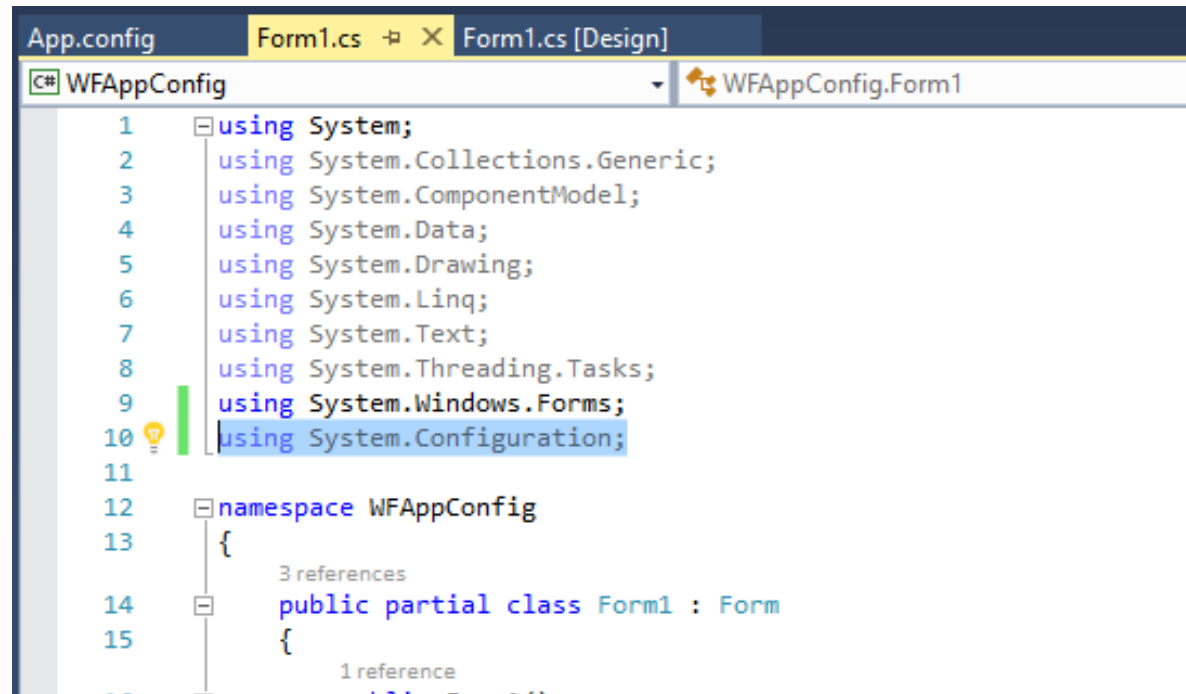


Configuration Manager class and App.config



Configuration Manager class and App.config

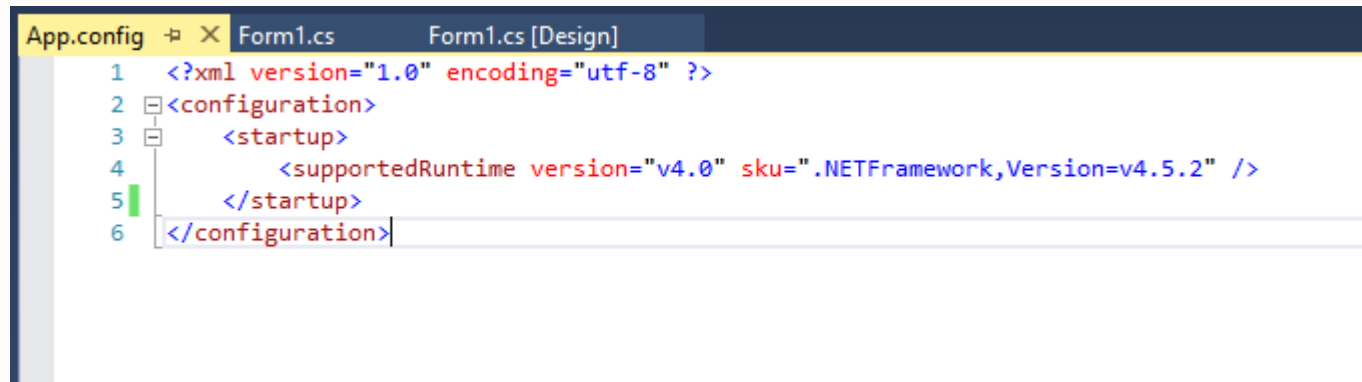
- After the reference is added, you must add the System.Configuration namespace (in Form1.cs file):



```
App.config  Form1.cs  Form1.cs [Design]
C# WAppConfig WAppConfig.Form1
1  using System;
2  using System.Collections.Generic;
3  using System.ComponentModel;
4  using System.Data;
5  using System.Drawing;
6  using System.Linq;
7  using System.Text;
8  using System.Threading.Tasks;
9  using System.Windows.Forms;
10 using System.Configuration;
11
12 namespace WAppConfig
13 {
14     3 references
15     public partial class Form1 : Form
16     {
17         1 reference
```

Configuration Manager class and App.config

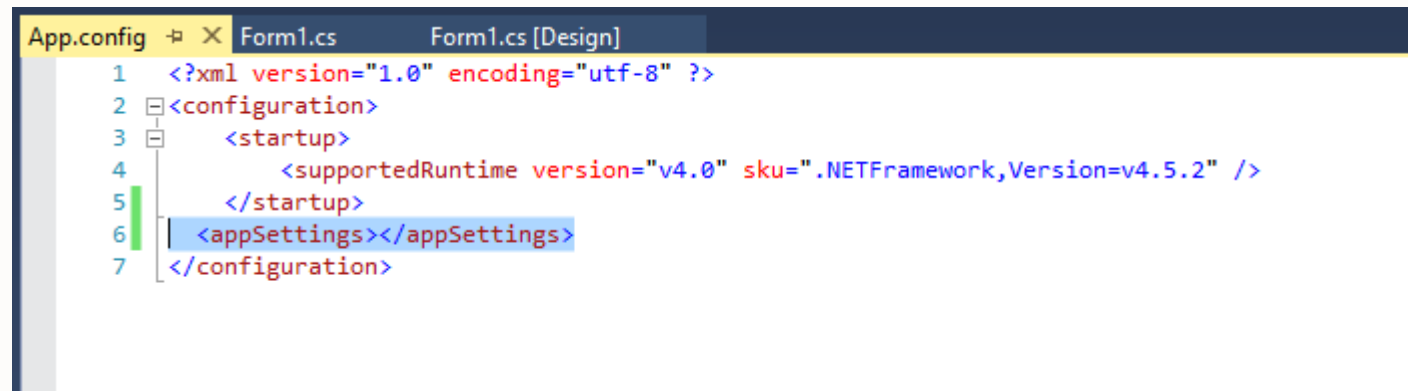
- Now, let's see what we currently have in App.config:



```
1 <?xml version="1.0" encoding="utf-8" ?>
2 <configuration>
3   <startup>
4     <supportedRuntime version="v4.0" sku=".NETFramework,Version=v4.5.2" />
5   </startup>
6 </configuration>
```

Configuration Manager class and App.config

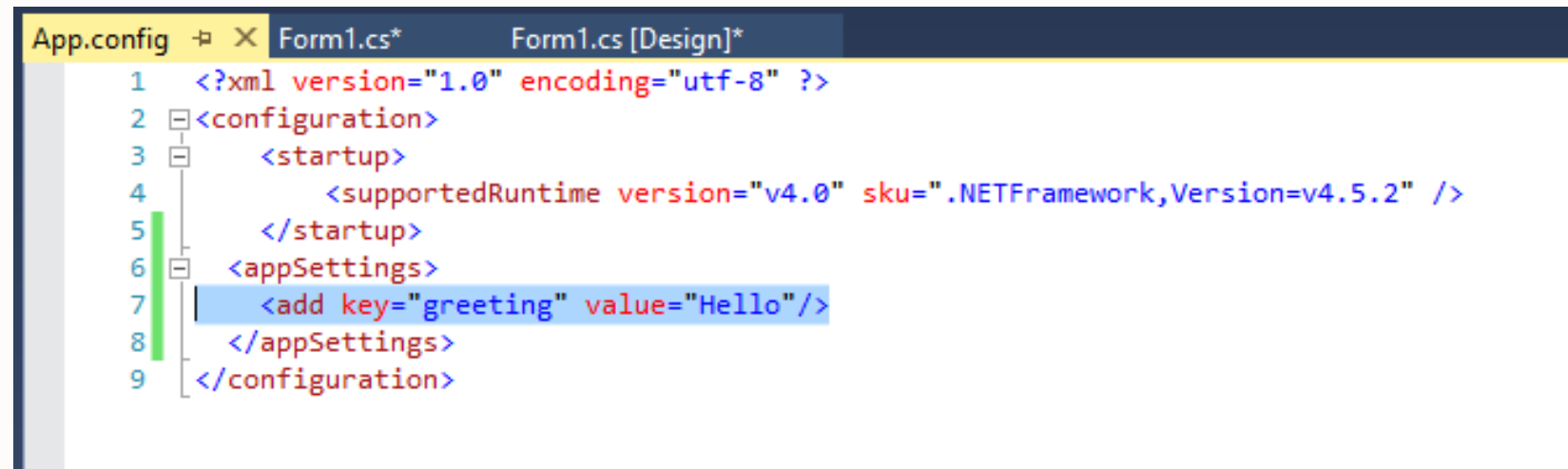
- Now, if we want to add some configuration settings to our application, we will include the set of tags corresponding to the AppSettings property of ConfigurationManager class (<appSettings> </appSettings>):

A screenshot of a Visual Studio code editor window. The title bar shows three tabs: 'App.config' (active), 'Form1.cs', and 'Form1.cs [Design]'. The 'App.config' tab is selected, and the file content is displayed. The code is XML and includes a configuration section with a startup element and an appSettings element. The 'appSettings' element is highlighted with a blue selection background. The code is as follows:

```
1 <?xml version="1.0" encoding="utf-8" ?>
2 <configuration>
3   <startup>
4     <supportedRuntime version="v4.0" sku=".NETFramework,Version=v4.5.2" />
5   </startup>
6   <appSettings></appSettings>
7 </configuration>
```

Configuration Manager class and App.config

- Between the start tag and end tag of AppSettings we can define new elements that are needed in order to configure the application:

A screenshot of a code editor window showing the content of an App.config file. The window has three tabs: 'App.config', 'Form1.cs*', and 'Form1.cs [Design]*'. The 'App.config' tab is active, displaying XML code. The code is as follows:

```
1 <?xml version="1.0" encoding="utf-8" ?>
2 <configuration>
3   <startup>
4     <supportedRuntime version="v4.0" sku=".NETFramework,Version=v4.5.2" />
5   </startup>
6   <appSettings>
7     <add key="greeting" value="Hello"/>
8   </appSettings>
9 </configuration>
```

The line containing the <add key="greeting" value="Hello"/> element is highlighted in blue. A green vertical line is positioned to the left of the code, between lines 5 and 6.

Configuration Manager class and App.config

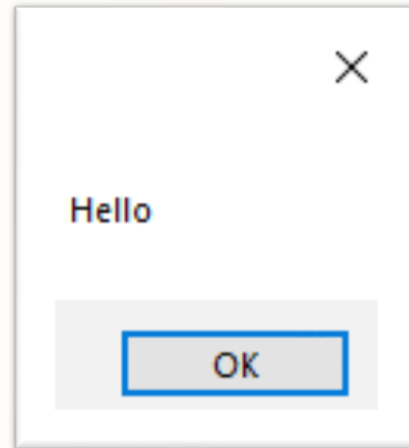
- If we want to use the configuration element defined in App.config inside our application, we can do so by using the property AppSettings of ConfigurationManager class (and we specify the key of the element):

```
namespace WFAAppConfig
{
    3 references
    public partial class Form1 : Form
    {
        1 reference
        public Form1()
        {
            InitializeComponent();
        }

        1 reference
        private void Form1_Load(object sender, EventArgs e)
        {
            MessageBox.Show(ConfigurationManager.AppSettings["greeting"]);
        }
    }
}
```


Configuration Manager class and App.config

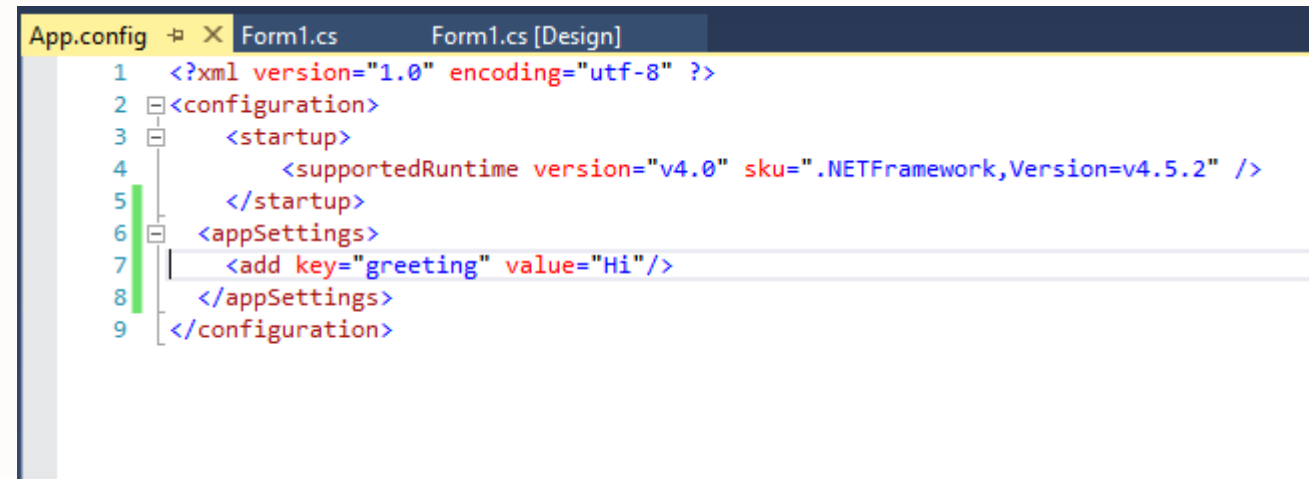
- When we run the application, we get the following output:



- We can observe that the MessageBox contains the value corresponding to the key called “greeting”

Configuration Manager class and App.config

- Now, if we want to change the value of the key called “greeting”, we must only modify its value inside App.config (we don’t have to modify it inside Form1.cs file):



```
1 <?xml version="1.0" encoding="utf-8" ?>
2 <configuration>
3   <startup>
4     <supportedRuntime version="v4.0" sku=".NETFramework,Version=v4.5.2" />
5   </startup>
6   <appSettings>
7     <add key="greeting" value="Hi" />
8   </appSettings>
9 </configuration>
```

- This is very useful when we use a setting element in more than one place in our application’s code

Configuration Manager class and App.config

- If we run our application again, we get the updated value of the key “greeting”, without modifying anything inside Form1.cs file:

