**Namespace and Assembly ?**

A .Net Namespace provides the fundamental unit of logical code grouping while an assembly provides a fundamental unit of physical code grouping.

**Namespace**

Namespaces is a logical group of related classes that can be used by any other language targeting the Microsoft .Net framework . It is more used for logical organization of your classes. Namespaces are a way of grouping type names and reducing the chance of name collisions.

Assembly will contain Namespaces, Classes, Data types it's a small unit of code for deployment. Assembly defines the name of the .dll file.It also avoids dll hell problem.  
  
Namespace is used in order to avoid conflict of user defined classes.

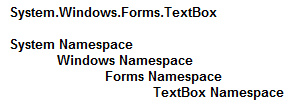
**Namespace:**  
1) it is a Collection of names wherein each name is Unique.  
2) They form the logical boundary for a Group of classes.  
3) Namespace must be specified in Project-Properties.

**Assembly:**  
1) It is an Output Unit.  
2) It is a unit of Deployment & a unit of versioning.  
3) Assemblies contain MSIL code.  
4) Assemblies are Self-Describing. [e.g. metadata,manifest]  
5)An assembly is the primary building block of a .NET Framework application.  
6) It is a collection of functionality that is built, versioned, and deployed as a single implementation unit (as one or more files).  
7) All managed types and resources are marked either as accessible only within their implementation unit, or by code outside that unit.

**Hierarchy and Fully-Qualified Names**

The fully qualified name of a class is constructed by concatenating the names of all the namespaces that contain the type. For e.g. the fully qualified name of the TextBox class is System.Windows.Forms.TextBox . That means TextBox class is contained in the Forms namespace that is contained in the Windows namespace that is contained in the root System namespace.

Textbox namespace Hierarchy



**Assembly**

An assembly is a collection of types and resources that are built to work together and form a logical unit of functionality. It is an Output Unit, that is .exe or .dll file. It is a unit of Deployment and a unit of versioning and also it contain MSIL (Microsoft Intermediate Language) code. Assemblies are self describing, it contains all the metadata about the modules, types, and other elements in the form of a manifest.

Assemblies are of two types: Private and Shared Assemblies.

**Private Assemblies**

Private assembly is intended only for one application. The files of that assembly must be placed in the same folder of the application.

**Shared Assemblies**

Shared assembly is to be made into a Shared Assembly, then the naming conventions are very strict since it has to be unique across the entire system

An assembly's name is stored in metadata and has a significant impact on the assembly's scope and use by an application. The display name of an assembly is obtained using the Assembly.FullName property. The runtime uses this information to locate the assembly and differentiate it from other assemblies with the same name.

using System;

using System.Windows.Forms;

namespace WindowsFormsApplication1

{

public partial class Form1 : Form

{

public Form1()

{

InitializeComponent();

}

private void button1\_Click(object sender, EventArgs e)

{

Type t = typeof(System.Text.Encoding );

string s = t.Assembly.FullName.ToString();

MessageBox.Show ("Assembly Name" + s.ToString ());

}

}

}