

Programación Web .NET Core

Módulo 1

Ensamblados

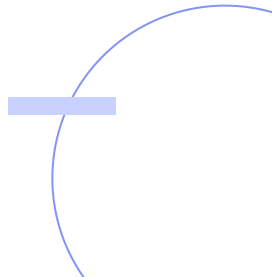
Ensamblado (*ASSEMBLY*)

Los ensamblados son los bloques de creación de las aplicaciones .NET Framework. Constituyen la **unidad fundamental de implementación**, control de versiones, reutilización, ámbitos de activación y permisos de seguridad. Un ensamblado es una **colección de tipos y recursos compilados para funcionar en conjunto y formar una unidad lógica de funcionalidad**.

Los ensamblados proporcionan a *Common Language Runtime* la información necesaria para conocer las implementaciones de tipos. Para la ejecución, un tipo no existe fuera del contexto de un ensamblado.

Ejemplos de Ensamblados:

- Librería de clases **DLL**.
- Archivo ejecutable **“.exe”**



Bibliotecas o Librerías de Clase

Las **DLL** o **Dynamic Link Library** (**Bibliotecas o Librerías de Enlace Dinámico**) son archivos con código ejecutable, en general, con **funcionalidad específica** (por ejemplo, una librería con funciones de gráficas de interfaz de usuario, o funciones matemáticas, o de conectividad de acceso a datos), que se cargan a memoria bajo demanda por parte del sistema operativo o aplicación.

Estas librerías **pueden ser reutilizadas usando cualquier lenguaje de programación.**

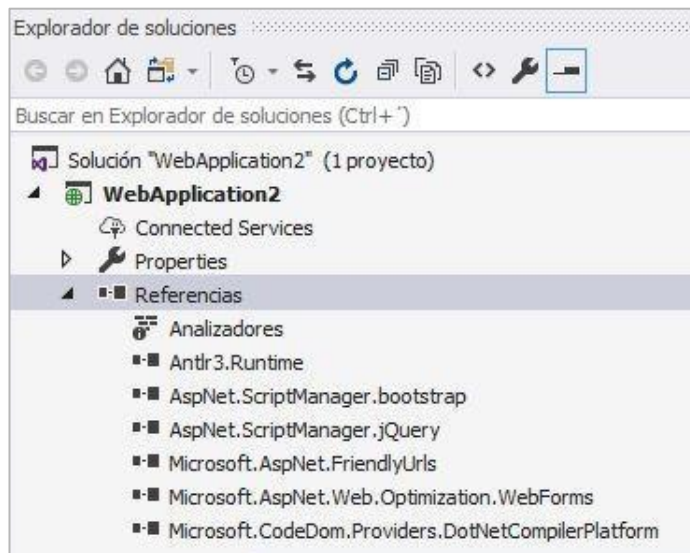


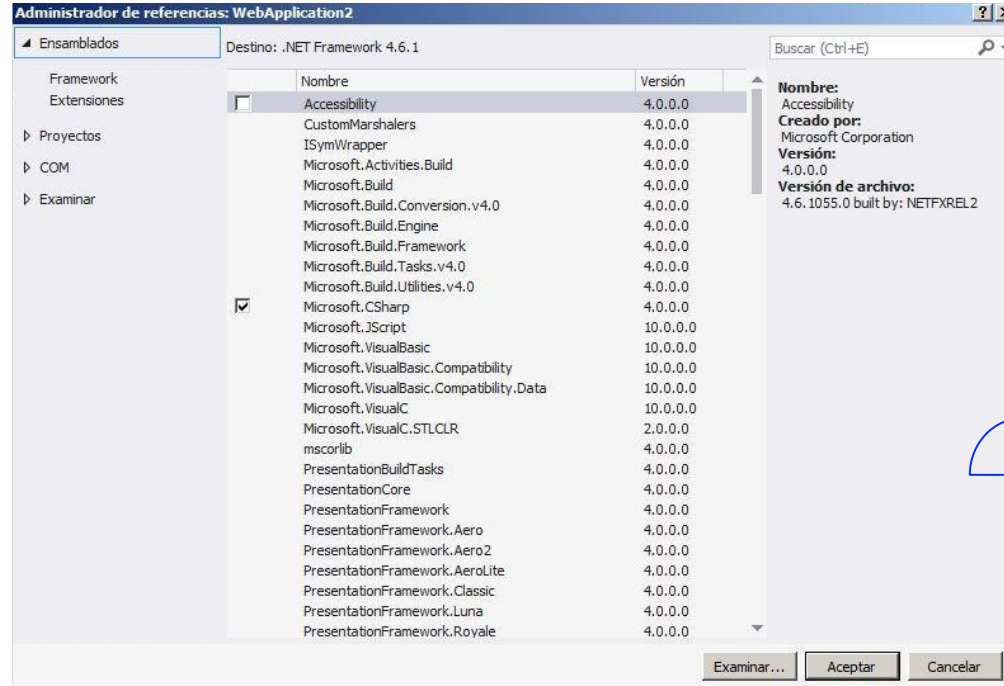
Referencias del proyecto

Para usar un componente o librería en una aplicación, se debe agregar primero una referencia a ese componente.

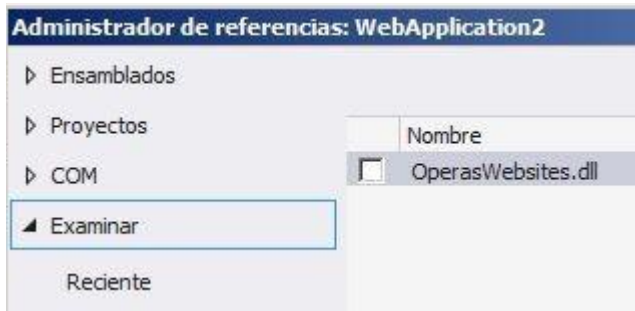
En la imagen de la derecha, podemos observar, en la carpeta **Referencias**, las referencias al proyecto **WebApplication2**.

Visual Studio proporciona cinco opciones en el cuadro de diálogo **Agregar referencia**, que se accede con click derecho desde la carpeta **Referencias**. Veámoslo en la slide a continuación.





- **Ensamblados:** Enumera todos los componentes de .NET Framework disponibles para poder hacer referencias.
- **Proyectos:** Enumera todos los componentes reutilizables creados en proyectos locales, que estén dentro de la misma solución.
- **COM:** Enumera todos los componentes de COM disponibles para hacer referencias.
- **Examinar:** Permite buscar un componente en el sistema de archivos.
- **Reciente:** Contiene una lista de componentes recién agregados a proyectos de su equipo.



El número de opciones en la parte superior del cuadro de diálogo **Agregar referencia** varía en función del tipo de proyecto abierto y de los recursos que éste utiliza.

Es posible que algunos componentes de la lista no aparezcan, depende de la versión de .NET Framework del proyecto. Esta desincronización puede surgir al tratar de referenciar componentes de versiones anteriores del .NET Framework.

Para más detalles, consulte:

[Guía de migración a .NET Framework 4.7, 4.6 y 4.5.](#)



Using

La palabra clave **using** tiene dos usos iniciales:

Como directiva:

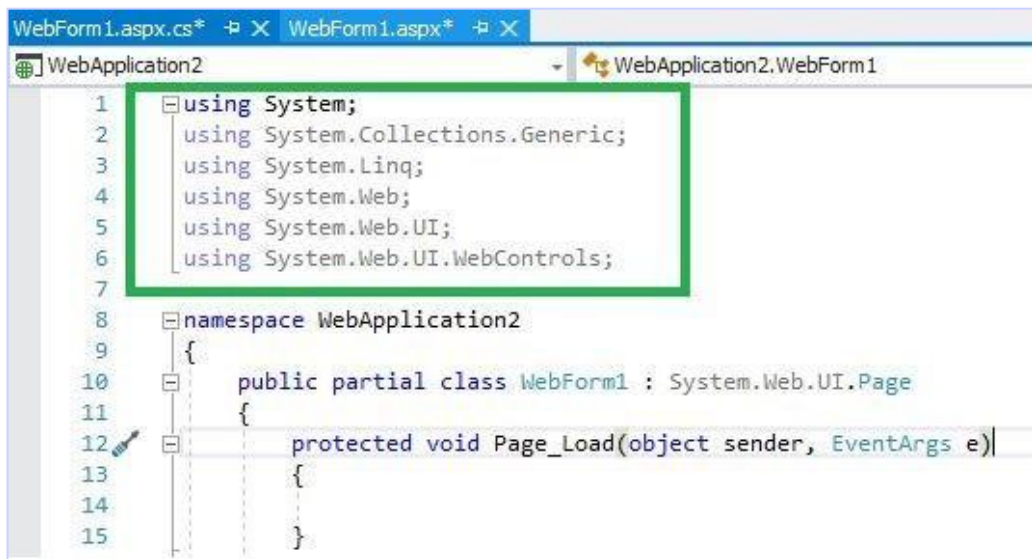
Cuando se utiliza para crear un alias para un espacio de nombres (**namespace**) o para importar tipos definidos en otros espacios de nombres.

Utilizada al comienzo del archivo, se coloca junto al espacio de nombres que se quiere importar. De esta manera, todas las clases contenidas en ese **namespace** podrán ser accedidas, sin necesidad de colocar el nombre completo del ensamblado a cada momento.



.NET
Core

Para que se reconozca un **Namespace**, debe existir dentro del ensamblado o estar presente en una referencia agregada.



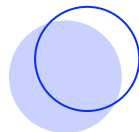
```
1  using System;
2  using System.Collections.Generic;
3  using System.Linq;
4  using System.Web;
5  using System.Web.UI;
6  using System.Web.UI.WebControls;
7
8  namespace WebApplication2
9  {
10     public partial class WebForm1 : System.Web.UI.Page
11     {
12         protected void Page_Load(object sender, EventArgs e)
13         {
14         }
15     }
16 }
```

Como instrucción:

Cuando define un ámbito o alcance al final del cual el objeto se destruye. Se puede utilizar para forzar al compilador a eliminar todo lo que exista dentro de un bloque **using**, una vez que alcanza su última línea de código.

Veamos el siguiente ejemplo, donde la variable **conexión** existe dentro del bloque **using**:

```
using (varconexion = newSqlConnection(Utils.ConnectionString))
{
    conexion.Open();
    //Aquí hacer algo con la conexión a base de datos
    //...
} //Aquí la variable conexion ya no existe, quedó fuera de alcance
```



**¡Sigamos
trabajando!**