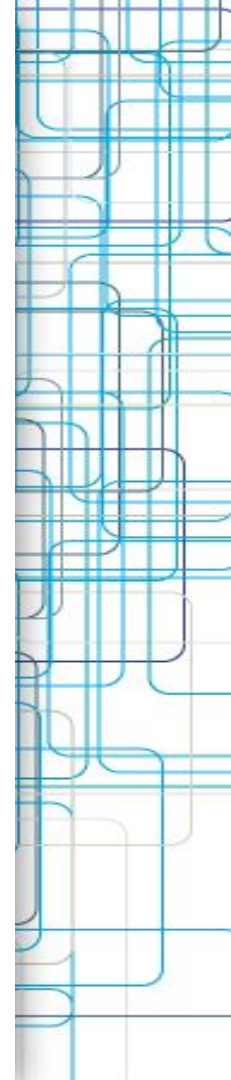


UT 3 - Componentes

INTRODUCCIÓN C#

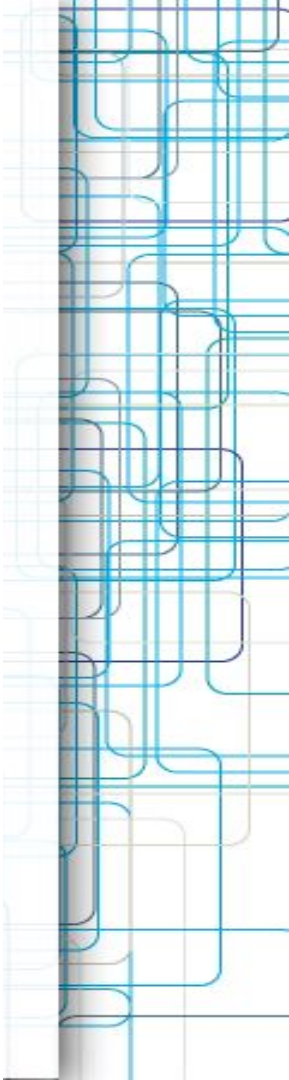


Índice

- Introducción
 - Características de .NET
 - Componentes .NET
 - Plataforma .NET
 - Ejemplo
- 

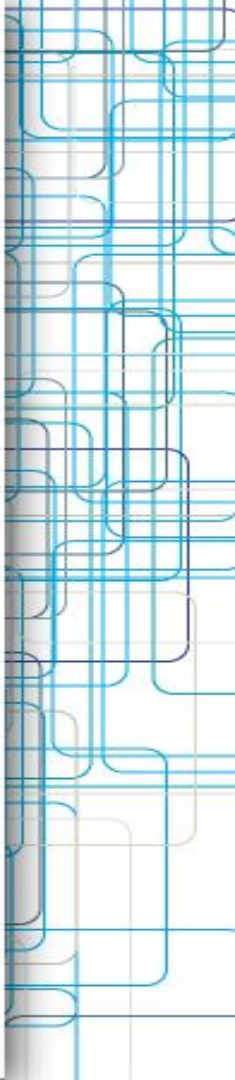



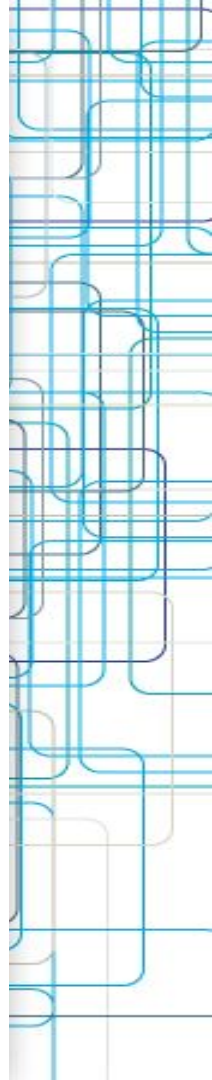
Introducción

- Microsoft .NET es el conjunto de nuevas tecnologías en las que Microsoft ha estado trabajando durante los últimos años
 - .NET ofrece una plataforma sencilla y potente para distribuir el software en forma de servicios que puedan ser suministrados remotamente y que puedan comunicarse y combinarse unos con otros de manera totalmente independiente de la plataforma, lenguaje de programación y modelo de componentes con los que hayan sido desarrollados.
 - www.microsoft.com/net/
- 



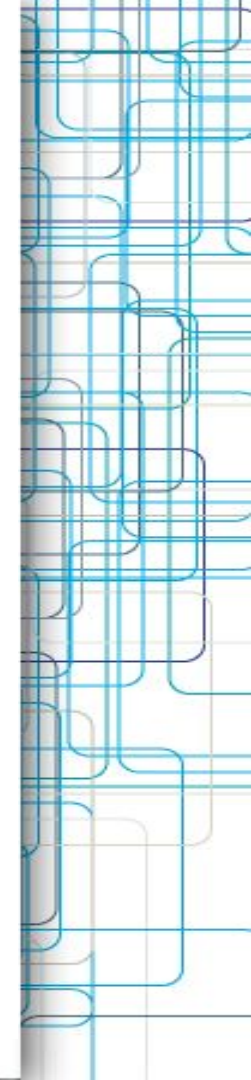
Características de .NET

- .NET es una nueva plataforma para el desarrollo y explotación de aplicaciones “gestionadas” o “administradas” (managed) modernas y orientadas a objetos.
 - Las aplicaciones .NET se pueden desarrollar en cualquier lenguaje de programación que se ajuste a .NET
 - .NET soporta una extensa colección de librerías de clases independientes del lenguaje de programación.
 - .NET soporta la creación de componentes.
 - .NET ofrece integración multi-lenguaje, reutilización de componentes, y herencia entre componentes desarrollados en diferentes lenguajes.
- 

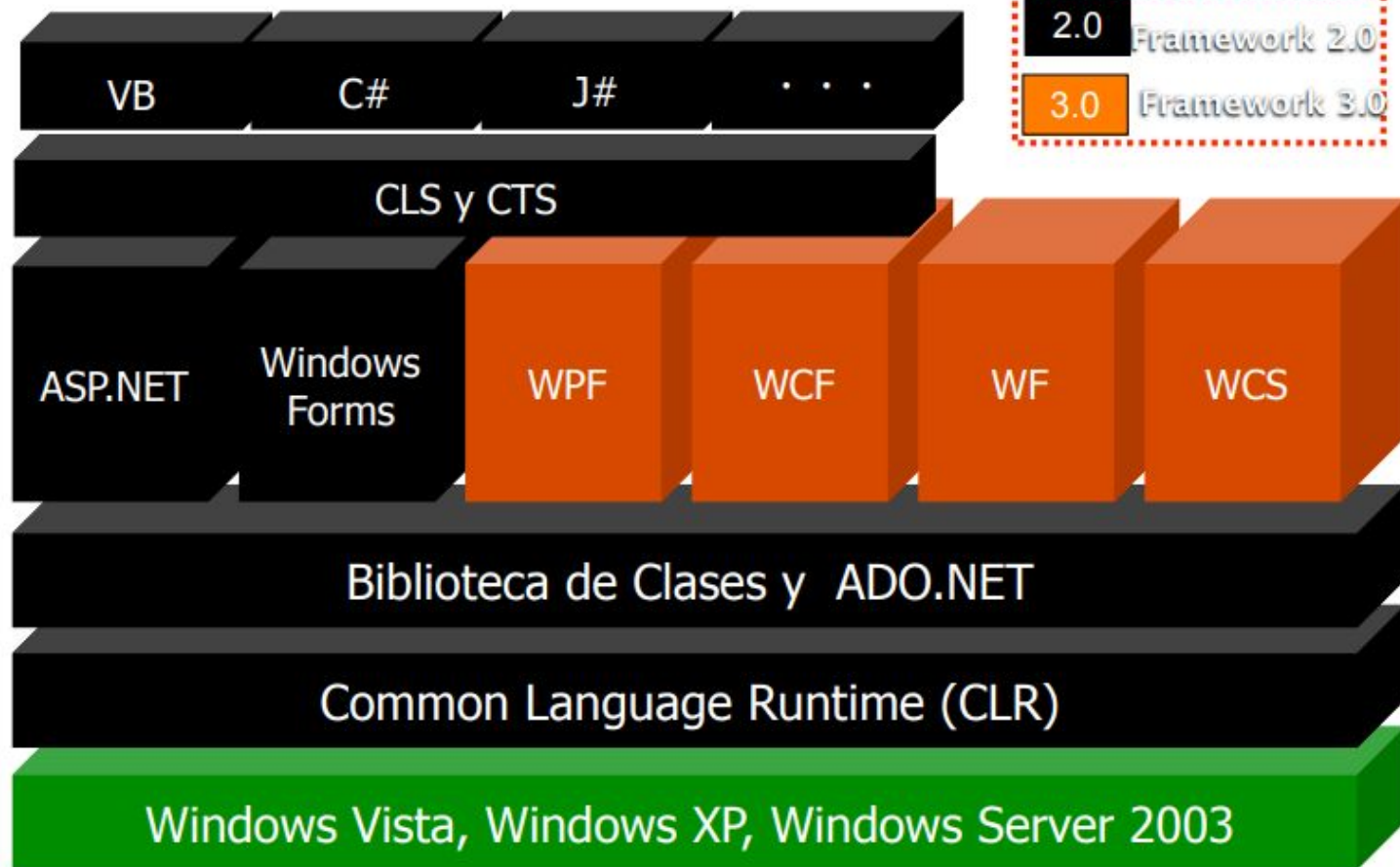
- 
- .NET ofrece una nueva manera de desarrollar aplicaciones gráficas usando WPF (Windows Presentation Foundation)
 - .NET ofrece una nueva manera de desarrollar aplicaciones basadas en navegador Web a través de ASP.NET
 - Las clases ADO.NET proveen una arquitectura desconectada para acceso a datos a través de Internet
 - .NET soporta la creación de Servicios Web XML independientes de la plataforma, a través de SOAP (Simple Object Access Protocol) y WSDL (Web Services Description Language)
 - .NET ofrece una nueva arquitectura para el desarrollo y explotación de objetos remotos WCF (Windows Communication Foundation)
 - .NET permite el desarrollo de RIA (Rich Internet Applications) a través de Silverlight
- 



Componentes de .NET

- Microsoft .NET está compuesto de:
 - Plataforma .NET
 - .NET Framework SDK
 - Visual Studio .NET
 - Servicios Web (Microsoft .NET myServices)
 - Servidores para empresas (SQL Server.NET...)
- 

Plataforma .NET




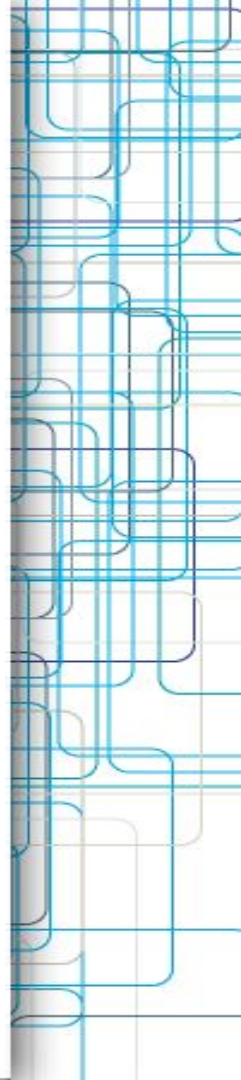


Common Language Runtime (CLR)

- Un **Runtime** es un entorno en el que se ejecutan los programas.
- El **CLR** se encarga de **gestionar** la ejecución de las aplicaciones .NET.
- Al código escrito para ejecutarse en la plataforma .NET se le llama **código gestionado**.
- Al código escrito para ejecutarse directamente se le llama **código no gestionado** o **código nativo**.
- El CLR realiza una compilación **Just in Time (JIT)** que traduce el código gestionado en código nativo sobre la arquitectura de hardware sobre la que se ejecuta.


Modelo de ejecución


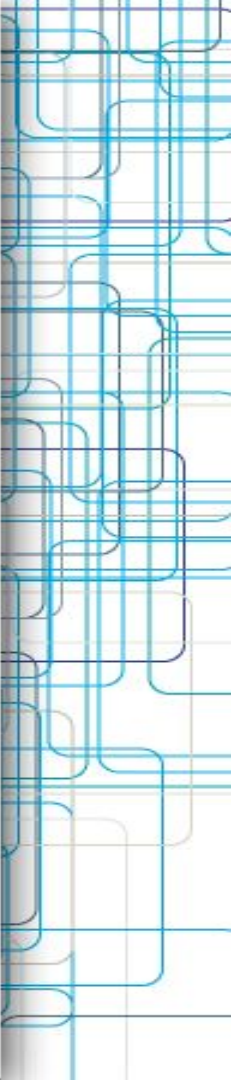



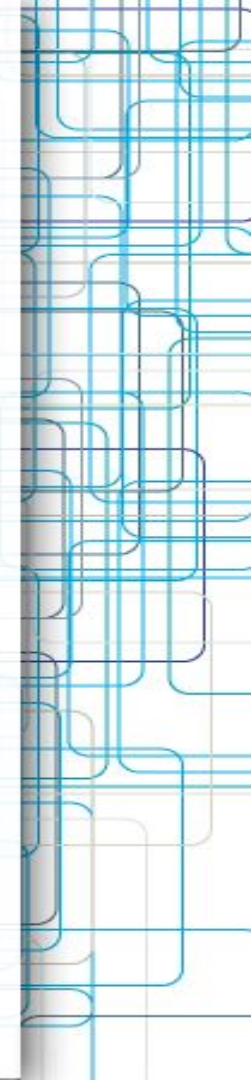
- 
- Características y servicios que ofrece:
 - Ejecución multiplataforma
 - Integración de lenguajes
 - Gestión de memoria (Recolector de basura)
 - Tratamiento de excepciones
 - Soporte multi-hilo (multi-threading)
 - Distribución transparente
 - Interoperabilidad con código antiguo
- 

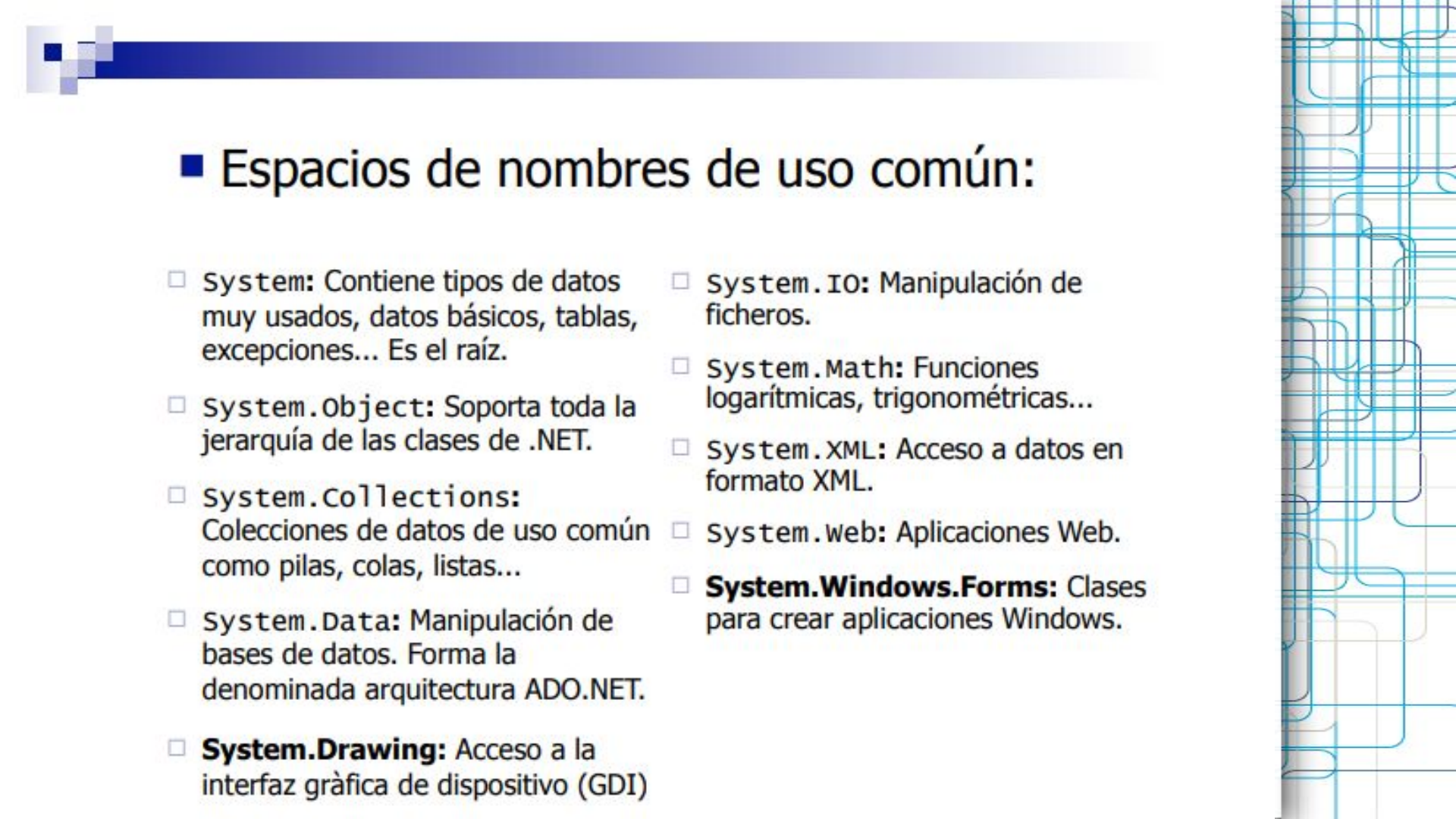


Clases de la plataforma .NET (FCL)

- La librería de clases (**Framework Class Library**) es una librería formada por cientos de tipos que permiten acceder a los servicios ofrecidos por el CLR y a sus funcionalidades.
 - Además, el programador puede crear nuevas clases que extiendan su funcionalidad y se integren perfectamente con el resto de las clases de la FCL.
- 

- 
- Esta librería de clases está escrita en **MSIL**
 - Cualquier lenguaje cuyo compilador genere MSIL podrá usarla.
 - Con esta librería podemos crear todo tipo de aplicaciones: aplicaciones de consola, de ventanas, servicios Web, ASP.NET...
- 

- 
- Dada la amplitud de la FCL, ha sido necesario organizar sus clases en **espacios de nombres (Namespace)** que agrupan clases con funcionalidades similares.
 - Se organiza de forma jerárquica.
 - El espacio de nombres **System** es el espacio raíz del que cuelgan todos los demás.
- 



■ Espacios de nombres de uso común:

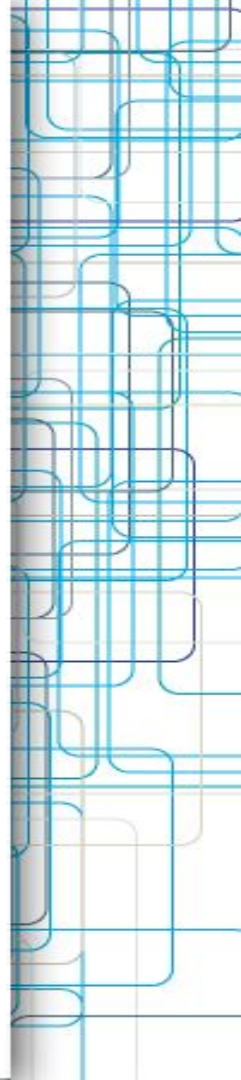
- **System:** Contiene tipos de datos muy usados, datos básicos, tablas, excepciones... Es el raíz.
- **System.Object:** Soporta toda la jerarquía de las clases de .NET.
- **System.Collections:** Colecciones de datos de uso común como pilas, colas, listas...
- **System.Data:** Manipulación de bases de datos. Forma la denominada arquitectura ADO.NET.
- **System.Drawing:** Acceso a la interfaz gráfica de dispositivo (GDI)
- **System.IO:** Manipulación de ficheros.
- **System.Math:** Funciones logarítmicas, trigonométricas...
- **System.XML:** Acceso a datos en formato XML.
- **System.Web:** Aplicaciones Web.
- **System.Windows.Forms:** Clases para crear aplicaciones Windows.



.NET Framework SDK


- Microsoft distribuye este kit de desarrollo como parte del paquete .NET. Disponible en:


<http://www.microsoft.com/downloads>


- Este kit contiene documentación sobre la plataforma, ejemplos y código fuente, y una serie de utilidades que sirven para desarrollo y prueba de aplicaciones .NET.
- 



Visual Studio.NET

- Es un entorno gráfico que permite a los desarrolladores crear, probar y depurar aplicaciones desarrolladas o no para la plataforma .NET.
 - La última versión es la 2008.
- 

- 
- Este entorno incorpora los siguientes lenguajes de programación:
 - Visual Basic.NET: Versión .NET de VBasic 6.0.
 - Visual C++.NET: Evolución de Visual C++.
 - Visual J#.NET: Adaptación de Visual J++.
 - Visual C#.NET: El nuevo lenguaje de POO.
 - Empresas ajenas han creado compiladores de sus lenguajes para la plataforma .NET:

Eiffel, Perl, Python, Haskell, Pascal, Oberon...
- 



Desarrollando para la plataforma .NET

- .NET es independiente del lenguaje de programación, no estamos restringidos a desarrollar en Visual C++ y Basic:
 - Existen compiladores para C#, Visual Basic.NET, Jscript.NET, COBOL, Python, Perl, etc.
 - .NET soporta estos lenguajes no soportando realmente ninguno de ellos: **.NET sólo entiende Microsoft Intermediate Language (MSIL)**



Una plataforma independiente del lenguaje

- En .NET se traduce código fuente a IL (Intermediate Language) + Metadatos
 - IL es compilado antes de ser ejecutado y no está diseñado para un lenguaje en particular como en Java.
 - Las sentencias IL manipulan tipos comunes compartidos por todos los lenguajes .NET (Common Type System – CTS).
 - El Common Language Runtime (CLR) es responsable de cargar y ejecutar una aplicación .NET:
 - Usa compilación **JIT** (Just-In-Time) para traducir **IL** a código nativo
 - IL es siempre compilado y nunca interpretado
 - Cada método del código IL sólo es compilado una vez cuando es indicado.

Ejemplo: Hola Mundo en C# .NET

// fichero: hola.cs

```
using System;
```

```
class Hola {
```

```
    public static void Main() {
```

```
        Console.WriteLine("Hola Món");
```

```
    }
```

```
}
```

- Todo programa en C# contiene al menos **una** clase.
- Main es el punto de entrada del programa .
- `Console.WriteLine("...")`, visualiza un mensaje, invocando método `writeLine` de la clase `Console`.
- El espacio de nombres `System` pertenece a .NET FCL (Framework Class Library).
- FCL contiene muchas clases útiles, que pueden usarse desde nuestras aplicaciones .NET.