****

**Trabajo practico N°1**

**Alumno: Maximiliano Garrett**

**Profesor: Gabriel Turich**

**Materia: .NET**

1- **que es .NET:** se podría definir como un conjunto de herramientas, tecnologías y servicios que facilita la construcción de todo tipo de aplicaciones.

.NET trae consigo una serie de beneficios como:

* Disminuir el tiempo de desarrollo de los proyectos.
* Simplificar el mantenimiento de las aplicaciones desarrolladas en esta plataforma
* Poder utilizar las numerosas funcionalidades ya diseñadas que permite a los diseñadores realizar un proyecto de importación y adaptación a las necesidades en lugar de un desarrollo desde cero.

2- **características principales:**

* Programación orientada a objetos.
* Soporte para múltiples lenguajes.
* Fácil desarrollo basado en componentes.
* Soporte para biblioteca de clases (base clase library).
* Soporte para comunicación con servicios web.

3-**.NET Framework:** es un entorno para construir, instalar y ejecutar servicios web y otras aplicaciones. Se compone de tres partes principales: el Common Lenguage Runtime, las clases Framework y ASP.NET.

El Microsoft .NET Framework, es un componente de software que puede ser o es incluido en los sistemas operativos Microsoft Windows. Provee soluciones pre-codificadas para requerimientos comunes de los programas y gestiona la ejecución de programas escritos específicamente para este framework.   
  
Microsoft desea que todas las aplicaciones creadas para la plataforma Windows, sean basadas en el .NET Framework. Su objetivo es crear un marco de desarrollo de software sencillo, reduciendo las vulnerabilidades y aumentando la seguridad de los programas desarrollados.

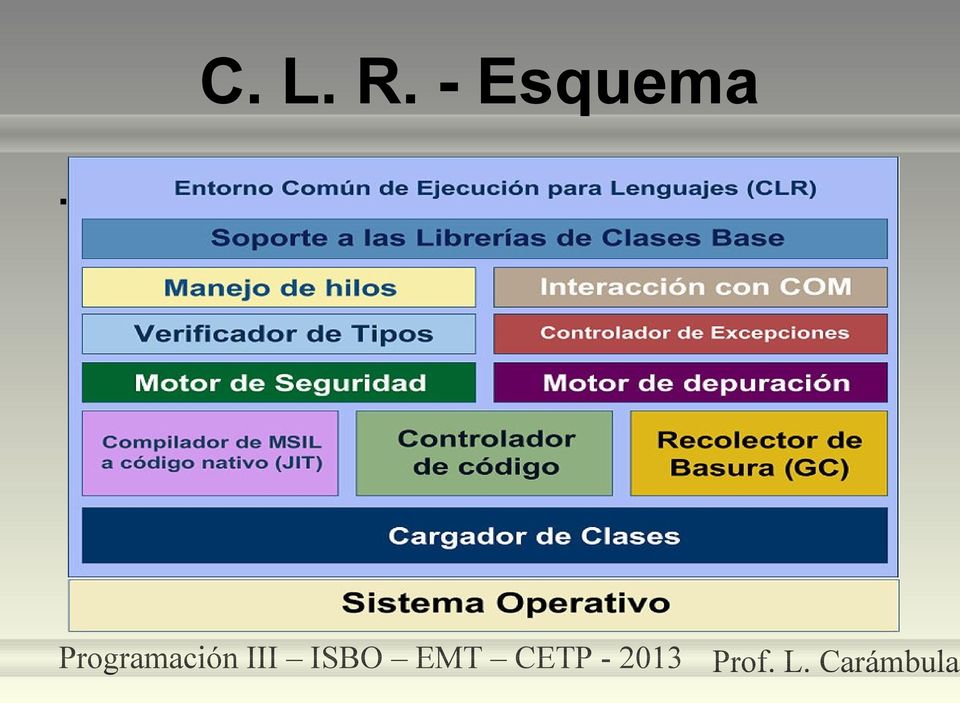
4-**versiones de .NET framework.**

| **Versiones de sistema operativo** | **Versiones de Microsoft .NET Framework** | | | | |
| --- | --- | --- | --- | --- | --- |
| **.NET Framework 2.0 SP2** | **.NET Framework 3.0 SP2** | **.NET Framework 3.5 SP1** | **.NET Framework 4.0** | **.NET Framework 4.5** |
| Windows Server 2003 Web Edition (32 bits) | Sí | Sí | Sí | Sí | No |
| Windows Server 2003 SE (32 bits) | Sí | Sí | Sí | Sí | No |
| Windows Server 2003 EE (32 bits) | Sí | Sí | Sí | Sí | No |
| Windows Server 2003 Datacenter Edition (32 bits) | Sí | Sí | Sí | Sí | No |
| Windows Server 2003 SE (64 bits) | Sí | Sí | Sí | Sí | No |
| Windows Server 2003 EE (64 bits) | Sí | Sí | Sí | Sí | No |
| Windows Server 2003 Datacenter Edition (64 bits) | Sí | Sí | Sí | Sí | No |
| Windows Server 2003 R2 | Sí | Sí | Sí | Sí | No |
| Windows Server 2008 Web Edition (32 bits) | Sí | Sí | Sí | Sí | En Service Pack 2  En Windows Server 2008 Service Pack 2, no se admite .NET Framework 4.5 en el rol Server Core. |
| Windows Server 2008 SE (32 bits) | Sí | Sí | Sí | Sí | En Service Pack 2  En Windows Server 2008 Service Pack 2, no se admite .NET Framework 4.5 en el rol Server Core. |
| Windows Server 2008 EE (32 bits) | Sí | Sí | Sí | Sí | En Service Pack 2  En Windows Server 2008 Service Pack 2, no se admite .NET Framework 4.5 en el rol Server Core. |
| Windows Server 2008 Datacenter Edition (32 bits) | Sí | Sí | Sí | Sí | En Service Pack 2  En Windows Server 2008 Service Pack 2, no se admite .NET Framework 4.5 en el rol Server Core. |
| Windows Server 2008 Web Edition (64 bits) | Sí | Sí | Sí | Sí | En Service Pack 2  En Windows Server 2008 Service Pack 2, no se admite .NET Framework 4.5 en el rol Server Core. |
| Windows Server 2008 SE (64 bits) | Sí | Sí | Sí | Sí | En Service Pack 2  En Windows Server 2008 Service Pack 2, no se admite .NET Framework 4.5 en el rol Server Core. |
| Windows Server 2008 EE (64 bits) | Sí | Sí | Sí | Sí | En Service Pack 2  En Windows Server 2008 Service Pack 2, no se admite .NET Framework 4.5 en el rol Server Core. |
| Windows Server 2008 Datacenter Edition (64 bits) | Sí | Sí | Sí | Sí | En Service Pack 2  En Windows Server 2008 Service Pack 2, no se admite .NET Framework 4.5 en el rol Server Core. |
| Windows Server 2008 R2 Web Edition (64 bits) | Sí | Sí | Sí | Sí | Sí En Windows Server 2008 R2 .NET Framework 4.5 se admite en el rol Server Core con Service Pack 1 o posterior. |
| Windows Server 2008 R2 SE (64 bits) | Sí | Sí | Sí | Sí | Sí En Windows Server 2008 R2 .NET Framework 4.5 se admite en el rol Server Core con Service Pack 1 o posterior. |
| Windows Server 2008 R2 EE (64 bits) | Sí | Sí | Sí | Sí | Sí En Windows Server 2008 R2 .NET Framework 4.5 se admite en el rol Server Core con Service Pack 1 o posterior. |
| Windows Server 2008 R2 Datacenter Edition (64 bits) | Sí | Sí | Sí | Sí | Sí En Windows Server 2008 R2 .NET Framework 4.5 se admite en el rol Server Core con Service Pack 1 o posterior. |
| Windows Server 2012 Datacenter Edition | Sí | Sí | Sí | Sí | Sí |
| Windows Server 2012 Standard Edition | Sí | Sí | Sí | Sí | Sí |
| Windows Server 2012 Essentials Edition | Sí | Sí | Sí | Sí | Sí |
| Windows Server 2012 Foundation Edition | Sí | Sí | Sí | Sí | Sí |
| Máquinas virtuales que se crean en Hyper-V R2 Server | Sí | Sí | Sí | Sí | Sí |
| Los dos agentes de .NET Framework de 32 bits y 64 bits supervisan las versiones 2.0, 3.0, 3.5 y 4.5 de .NET Framework. | | | | |  |

5- **entorno de ejecución de CLR:** El **Common Language Runtime** o **CLR** ("entorno en tiempo de ejecución de lenguaje común") es un [entorno de ejecución](https://es.wikipedia.org/wiki/Sistema_en_tiempo_de_ejecuci%C3%B3n) para los códigos de los [programas](https://es.wikipedia.org/wiki/Programa_inform%C3%A1tico) que corren sobre la plataforma [Microsoft .NET](https://es.wikipedia.org/wiki/Microsoft_.NET). El CLR es el encargado de [compilar](https://es.wikipedia.org/wiki/Compilador) una forma de código intermedio llamada [Common Intermediate Language](https://es.wikipedia.org/wiki/Common_Intermediate_Language) (CIL, anteriormente conocido como MSIL, por Microsoft Intermediate Language), al [código de maquina](https://es.wikipedia.org/wiki/C%C3%B3digo_de_maquina) nativo, mediante un [compilador en tiempo de ejecución](https://es.wikipedia.org/wiki/Compilaci%C3%B3n_en_tiempo_de_ejecuci%C3%B3n). No debe confundirse el CLR con una [máquina virtual](https://es.wikipedia.org/wiki/M%C3%A1quina_virtual), ya que una vez que el código está compilado, corre nativamente sin intervención de una capa de abstracción sobre el hardware subyacente. Es una implementación del estándar [Common Language Infrastructure](https://es.wikipedia.org/wiki/Common_Language_Infrastructure) (CLI).

6-**funcionamiento básico de CLR:** es una especificación estandarizada que describe un entorno virtual para la ejecución de aplicaciones, cuya principal característica es la de permitir que aplicaciones escritas en distintos lenguajes de alto nivel puedan luego ejecutarse en múltiples plataformas tanto de hardware como de software sin necesidad de reescribir o recompilar su código fuente.

7-**esquema de ejecución de CLR:**



8-**metadata:** metadatos son un conjunto de datos que describen el contenido informativo de un recurso, de archivos o de información de los mismos. Es decir, es información que describe otros datos. No obstante, la palabra metadatos no tiene una definición exclusiva, ya que son varias las expresiones con las que se conoce, tales como informaciones sobre datos, informaciones sobre informaciones o datos sobre informaciones.

A pesar de que la palabra metadatos se emplea en la actualidad, generalmente para el ámbito de la informática, el concepto es anterior a Internet, si bien el interés que ha suscitado ha hecho que se utilice en este entorno. El término metadatos es principalmente muy usado en el ámbito de la informática, enfocado a las empresas debido a la adquisición de una relevancia única en los últimos tiempos como consecuencia de las grandes cantidades de información que se manejan actualmente en las compañías. Por ello se emplean los metadatos como ayuda para gestionar los datos o información con que trabajan para una mayor efectividad y rendimiento.

9-**principales servicios de CLR:**

* **Modelo de programación consistente:**A todos los servicios y facilidades ofrecidos por el CLR se accede de la misma forma: a través de un modelo de programación orientado a objetos. Esto es una diferencia importante respecto al modo de acceso a los servicios ofrecidos por los algunos sistemas operativos actuales (por ejemplo, los de la familia Windows), en los que a algunos servicios se les accede a través de llamadas a funciones globales definidas en DLLs y a otros a través de objetos (objetos COM en el caso de la familia Windows)
* **Modelo de programación sencillo:** Con el CLR desaparecen muchos elementos complejos incluidos en los sistemas operativos actuales (registro de Windows, GUIDs, HRESULTS, IUnknown, etc.) El CLR no es que abstraiga al programador de estos conceptos, sino que son conceptos que no existen en la plataforma .NET
* **Eliminación del “infierno de las DLLs”:**En la plataforma .NET desaparece el problema conocido como “infierno de las DLLs” que se da en los sistemas operativos actuales de la familia Windows, problema que consiste en que al sustituirse versiones viejas de DLLs compartidas por versiones nuevas puede que aplicaciones que fueron diseñadas para ser ejecutadas usando las viejas dejen de funcionar si las nuevas no son 100% compatibles con las anteriores. En la plataforma .NET las versiones nuevas de las DLLs pueden coexistir con las viejas, de modo que las aplicaciones diseñadas para ejecutarse usando las viejas podrán seguir usándolas tras instalación de las nuevas. Esto, obviamente, simplifica mucho la instalación y desinstalación de software.
* **Ejecución multiplataforma:**El CLR actúa como una máquina virtual, encargándose de ejecutar las aplicaciones diseñadas para la plataforma .NET. Es decir, cualquier plataforma para la que exista una versión del CLR podrá ejecutar cualquier aplicación .NET. Microsoft ha desarrollado versiones del CLR para la mayoría de las versiones de Windows: Windows 95, Windows 98, Windows ME, Windows NT 4.0, Windows 2000, Windows XP y Windows CE (que puede ser usado en CPUs que no sean de la familia x86) Por otro lado Microsoft ha firmado un acuerdo con Corel para portar el CLR a Linux y también hay terceros que están desarrollando de manera independiente versiones de libre distribución del CLR para Linux. Asimismo, dado que la arquitectura del CLR está totalmente abierta, es posible que en el futuro se diseñen versiones del mismo para otros sistemas operativos.
* **Integración de lenguajes:**Desde cualquier lenguaje para el que exista un compilador que genere código para la plataforma .NET es posible utilizar código generado para la misma usando cualquier otro lenguaje tal y como si de código escrito usando el primero se tratase. Microsoft ha desarrollado un compilador de C# que genera código de este tipo, así como versiones de sus compiladores de Visual Basic (Visual Basic.NET) y C++ (C++ con extensiones gestionadas) que también lo generan y una versión del intérprete de JScript (JScript.NET) que puede interpretarlo. La integración de lenguajes es tal que es posible escribir una clase en C# que herede de otra escrita en Visual Basic.NET que, a su vez, herede de otra escrita en C++ con extensiones gestionadas.
* **Gestión de memoria:**El CLR incluye un **recolector de basura** que evita que el programador tenga que tener en cuenta cuándo ha de destruir los objetos que dejen de serle útiles. Este recolector es una aplicación que se activa cuando se quiere crear algún objeto nuevo y se detecta que no queda memoria libre para hacerlo, caso en que el recolector recorre la memoria dinámica asociada a la aplicación, detecta qué objetos hay en ella que no puedan ser accedidos por el código de la aplicación, y los elimina para limpiar la memoria de “objetos basura” y permitir la creación de otros nuevos. Gracias a este recolector se evitan errores de programación muy comunes como intentos de borrado de objetos ya borrados, agotamiento de memoria por olvido de eliminación de objetos inútiles o solicitud de acceso a miembros de objetos ya destruidos.
* **Seguridad de tipos:** El CLR facilita la detección de errores de programación difíciles de localizar comprobando que toda conversión de tipos que se realice durante la ejecución de una aplicación .NET se haga de modo que los tipos origen y destino sean compatibles.
* **Aislamiento de procesos:** El CLR asegura que desde código perteneciente a un determinado proceso no se pueda  acceder a código o datos pertenecientes a otro, lo que evita errores de programación muy frecuentes e impide que unos procesos puedan atacar a otros. Esto se consigue gracias al sistema de seguridad de tipos antes comentado, pues evita que se  pueda convertir un objeto a un tipo de mayor tamaño que el suyo propio, ya que al tratarlo como un objeto de mayor tamaño podría accederse a espacios en memoria ajenos a él que podrían pertenecer a otro proceso. También se consigue gracias a que no se permite acceder a posiciones arbitrarias de memoria.
* **Tratamiento de excepciones:**En el CLR todos los errores que se puedan producir durante la ejecución de una aplicación se propagan de igual manera: mediante excepciones. Esto es muy diferente a como se venía haciendo en los sistemas Windows hasta la aparición de la plataforma .NET, donde ciertos errores se transmitían mediante códigos de error en formato Win32, otros mediante HRESULTs y otros mediante excepciones.

10-**BCL:** a Biblioteca de Clases (BLC por sus siglas en ingles) maneja la mayoría de las operaciones básicas que se encuentran involucradas en el desarrollo de aplicaciones, incluyendo entre otras:

* Interacción con los dispositivos periféricos
* Manejo de datos (ADO.NET)
* Administración de memoria
* Cifrado de datos
* Transmisión y recepción de datos por distintos medios (XML, TCP/IP)
* Administración de componentes web que corren tanto en el servidor como en el cliente (ASP.NET)
* Manejo y administración de excepciones
* Manejo del sistema de ventanas
* Herramientas de despliegue de gráficos (GDI+)
* Herramientas de seguridad e integración con la seguridad del sistema operativo
* Manejo de tipos de datos unificado
* Interacción con otras aplicaciones
* Manejo de cadena de caracteres y expresiones regulares
* Operaciones aritméticas
* Manipulación de fechas, zonas horarias y periodos de tiempo
* Manejo de arreglos de datos y colecciones
* Manipulación de archivos de imágenes
* Aleatoriedad
* Generación de código
* Manejo de idiomas
* Auto descripción de código
* Compilación de código

Esta funcionalidad se encuentra organizada por medio de espacios de nombres jerárquicos.

La Biblioteca de Clases Base se clasifica, en cuatro grupos clave:

* ASP.NET y Servicios Web XML
* Windows Forms
* ADO.NET
* .NET

11-**Namespace:** Los espacios de nombres se utilizan como un sistema de la organización. Proporcionan una manera de clasificar y presentar elementos de programación que se exponen a otros programas y aplicaciones. Tenga en cuenta que un espacio de nombres no es un tipo en el sentido de que es una clase o estructura, no se puede declarar un elemento de programación que el tipo de datos de un espacio de nombres.