Introducción al .NET Core

.NET Core

- Es una plataforma de desarrollo de propósito general mantenida por Microsoft y la comunidad .NET en GitHub.
- Es multiplataforma, compatible con Windows, MacOS y Linux, y se puede utilizar en el dispositivo, la nube, y escenarios embebidos/IoT.

.NET Core, Características

- Implementación flexible.
- Multiplataforma.
- Herramientas de línea de comando.
- Compatible.
- Código abierto.
- Soportado.

Este nuevo framework ofrece una serie de características muy interesantes. Estas características ayudan a comprender porque Microsoft ha decidido crear un nuevo framework utilizando las lecciones aprendidas en el ya conocido .NET Framework.

- Implementación flexible: Se puede incluir en su aplicación o instalar lado a lado por el usuario o de todo el equipo.

- Multiplataforma: Se ejecuta en Windows, MacOS y Linux; puede ser portado a otros sistemas operativos. Los sistemas operativos soportados (OS), CPU y los escenarios de aplicación crecerá con el tiempo, proporcionados por Microsoft, otras empresas y los individuos.

- Herramientas de línea de comando: Todos los escenarios de construcción de aplicación pueden ser ejercidos en la línea de comandos.

- Compatible: .NET Core es compatible con .NET Framework, Xamarin y Mono, a través de la biblioteca estándar de .NET (.NET Standard Library).

- Código abierto: La plataforma .NET Core es de código abierto, utilizando las

licencias MIT y Apache 2. La documentación está disponible bajo CC-BY. .NET Core es un proyecto de la Fundación .NET.

- Con el soporte de Microsoft: .NET Core es soportado por con Microsoft.

.NET Core, Constitución

- .NET Core.
  - CoreCLR.
  - CoreFX.
- Herramientas de Desarrollo.
  - .NET Core SDK.
  - dotnet.exe.

El .NET Core está constituido por:

- Un “runtime” también conocido como CoreCLR, que ofrece un sistema de tipos, la carga de ensamblados, un recolector de basura, interoperabilidad nativa y otros servicios básicos.

- Un conjunto de bibliotecas, conocidas como CoreFX, que proporcionan los tipos de datos primitivos, modelos de aplicación y los servicios fundamentales.

- Un conjunto de herramientas de SDK y compiladores de lenguaje que permiten la experiencia de desarrollo, disponible en el .NET Core SDK.

- La aplicación anfitrión (host) dotnet.exe, que se utiliza para poner en marcha aplicaciones .NET Core. Este selecciona el “runtime” (CoreCLR, .NET Framework CLR) y aloja el “runtime”, ofrece políticas para la carga de ensamblados y ejecuta la aplicación. La misma aplicación “hos” dotnet.exe también se utiliza para poner en marcha las herramientas del .NET Core SDK.

.NET Core, CoreFX

- Vista como un BCL multiplataforma.
- Subconjunto de otros API.
- .NET Standard Library.
- Compatible con .NET Standard Library.

.NET Core puede ser visualizado como una versión multiplataforma de .NET Framework, en lo que respecta a la capa de las bibliotecas de clases base (BCL) del .NET Framework. El .NET Core implementa la especificación .NET Standard Library.

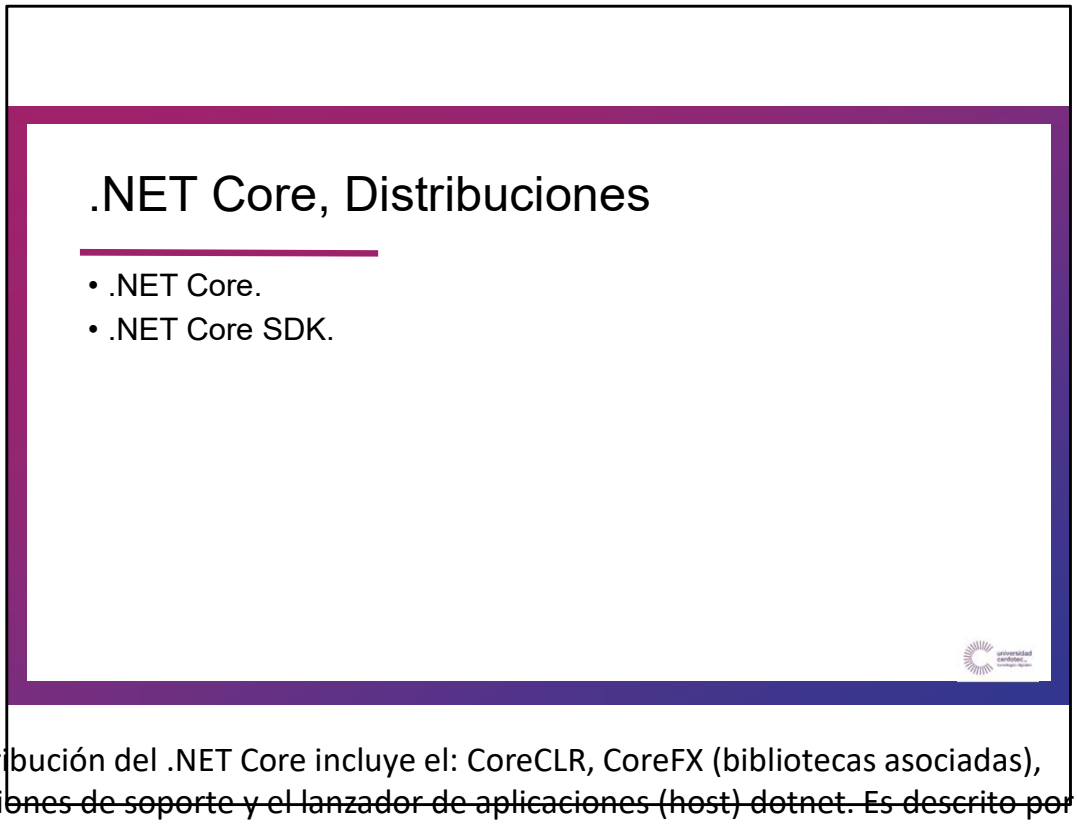
.NET Core proporciona un subconjunto de las API que están disponibles en el .NET Framework o Mono / Xamarin. En algunos casos, los tipos no se aplican plenamente (algunos miembros no están disponibles o se han movido).

La .NET Standard Library es una especificación API que describe el conjunto coherente de las API de .NET que los desarrolladores pueden esperar en cada implementación de aplicación .NET. Las implementaciones de .NET necesitan implementar esta especificación con el fin de ser considerado compatible con la .NET Standard Library y así dar soporte a las bibliotecas que orientadas al .NET Standard Library.

.NET Core implementa la .NET Standard Library, y por lo tanto es compatible con .NET Standard Library.

.NET Core, Modelos de Aplicación

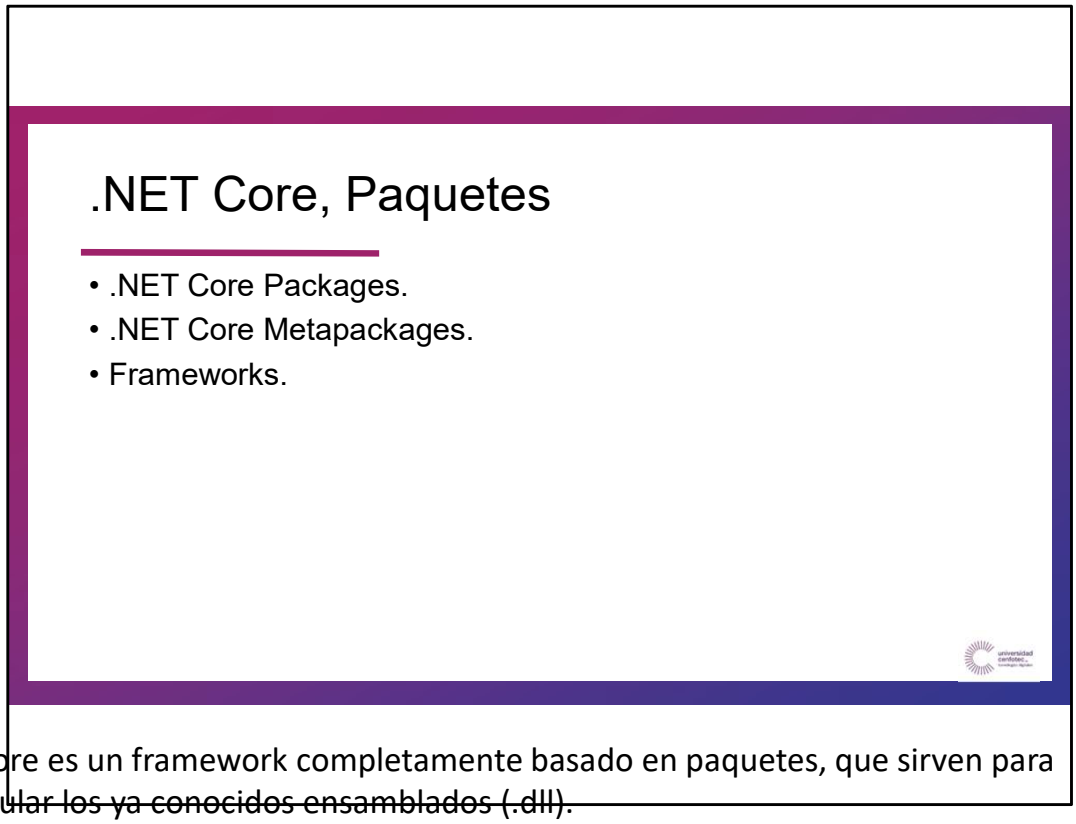
- .NET Core incorpora solamente un modelo de aplicaciones (aplicaciones de consola) que es útil para herramientas, servicios locales y juegos basados en texto.
- Modelos de aplicación adicionales se han construido en la parte superior del .NET Core para extender su funcionalidad, como:
  - ASP.NET Core
  - Windows 10 Universal Windows Platform (UWP)
  - Xamarin.Forms



La distribución del .NET Core incluye el: CoreCLR, CoreFX (bibliotecas asociadas), aplicaciones de soporte y el lanzador de aplicaciones (host) dotnet. Es descrito por el metapaquete Microsoft.NETCore.App.

La distribución de Microsoft .NET Core SDK incluye la .NET Core y un conjunto de herramientas para la restauración de paquetes NuGet, la compilación y la creación de aplicaciones.

https://www.microsoft.com/net/core#windows
https://www.microsoft.com/net/download#core



.NET Core es un framework completamente basado en paquetes, que sirven para encapsular los ya conocidos ensamblados (.dll).

.NET Core aprovecha la tendencia actual de publicar los paquetes por medio de la nube, en este caso en el sitio nuget.org, que sirve como directorio para los paquetes públicos, pero es posible contar también con directorios privadors para la publicación de paquetes.

.NET Core, Paquetes

- Los paquetes del .NET Core contienen al CoreCLR (runtime) y al CoreFX (bibliotecas básicas y sus implementaciones).
  - System.Runtime
  - System.Collections
  - System.Net.Http
  - System.IO.FileSystem
  - System.Linq
  - System.Reflection

.NET Core, Framework

- Los Frameworks describen un conjunto de API disponibles (y potencialmente otras características) que se puede aprovechar cuando el destino es un framework específico. Cambian de versión a medida que se añaden nuevas API.
- Frameworks basados en paquetes. Existe una relación bidireccional entre los frameworks y paquetes.
  - netstandard
  - netcoreapp

.NET Core, Tipos de Aplicación

- Aplicación portable.

Aplicaciones portable es el tipo predeterminado en .NET Core. Requieren que el .NET Core esté instalado en la máquina destino con el fin de que se ejecuten. Para el desarrollador, esto significa que su aplicación es portátil entre las instalaciones de .NET Core.

.NET Core, Tipos de Aplicación

- Aplicación portable con dependencias nativas.

Esta aplicación es tan portable como todas sus dependencias nativas son portables. La aplicación será capaz de ejecutar en cualquier plataforma que sus dependencias nativas se pueden ejecutar.

El primer ejemplo de esto es Kestrel, el servidor web de ASP.NET Core multiplataforma. Está construido sobre la biblioteca libuv que es su dependencia nativa.

Cuando se incluyen paquetes que no son parte del CoreFX y se encuentran publicados en Nuget.org, un vez que se obtienen por medio de Nuget quedan en una ubicación local en C:\Users\Gilberto\.nuget\packages para la plataforma de Microsoft Windows.

.NET Core, Tipos de Aplicación

- Aplicación autónoma.
- RID.

A diferencia de la aplicación portable, una aplicación autónoma no se basa en ningún componente compartido que esté presente en la máquina en la que desea ejecutar la aplicación.

Como su nombre lo indica, significa que todo el conjunto de las dependencias, incluyendo el “runtime” (CoreCLR, CLR) está empaquetado con la aplicación.

La momento de efectuar la publicación de la aplicación se deberá de especificar la plataforma sobre la cual se desea que aplicación ejecute, pues se generará un paquete específico para cada plataforma. Esto se debe indicar en la sección runtimes del archivo Project.json, el conjuntos de valores válidos son conocidos como RIDs (.NET Core Runtime IDentifier).

.NET Core, Arquitecturas

- Multiplataforma.
- Windows, MacOS y Linux.
- X64 y x86.
- ARM64 y ARM32 en progreso.

.NET Core es una implementación de .NET multiplataforma. Las preocupaciones arquitectónicas primarias específica de .NET Core están relacionados con proporcionar implementaciones específicas de la plataforma para las plataformas soportadas.

.NET Core es compatible con Microsoft Windows, MacOS y Linux. En Linux, Microsoft apoya principalmente .NET Core ejecutando en Red Hat Enterprise Linux (RHEL) y las familias de distribución de Debian, pero esto se podría ampliar en el futuro.

Actualmente .NET Core soporta CPUs x64. En Windows, x86 también es compatible. ARM64 y ARM32 están en curso.

.NET Core, Herramientas desarrollo

- Dotnet CLI (dotnet.exe).
- Visual Studio 2019.
  - Solución
  - Proyecto
- Otros.
  - Visual Studio Code.
  - Yeoman.
  - OmniSharp.

El .NET Core CLI es un nuevo conjunto de herramientas multiplataforma fundamental para el desarrollo de aplicaciones .NET Core. Es "fundamental", ya que es la capa principal en la que otras herramientas de alto nivel, tales como entornos de desarrollo integrado (IDE), editores y generadores de aplicación se sustentan.

También es multiplataforma de forma predeterminada y tiene la misma interface en cada una de las plataformas soportadas. Esto significa que cuando se aprende cómo utilizar la herramienta, se puede usar de la misma manera a partir de cualquiera de las plataformas soportadas.

El dotnet.exe se instala como parte del .NET Core y queda disponible en la ubicación: C:\Program Files\dotnet en la plataforma de Microsoft Windows.

El dotnet.exe es posible extenderlo con comandos adicionales información sobre el procedimiento puede ser encontrado en:https://github.com/dotnet/cli/blob/rel/1.0.0/Documentation/intro-to-cli.md, https://github.com/dotnet/cli/blob/rel/1.0.0/Documentation/developer-guide.md.

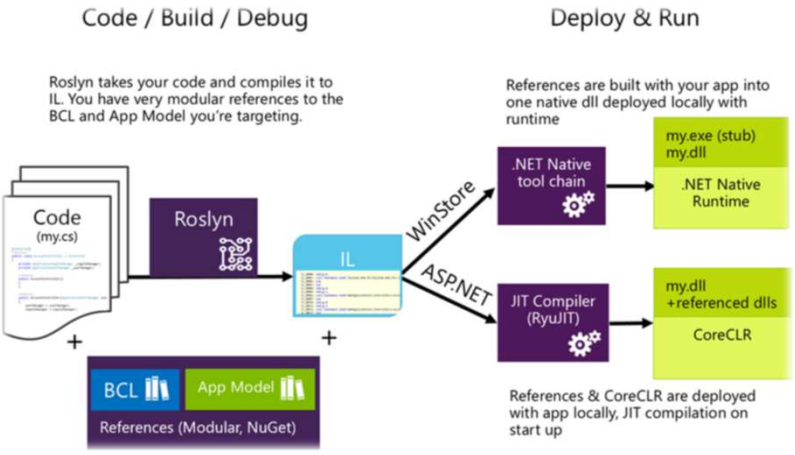
El archivo global.json sirve como base para la organización de los diferentes proyectos

de la solución. Viene a sustituir los archivos de definición de solución utilizados tradicionalmente por Visual Studio, y busca por lo tanto ser multiplataforma.

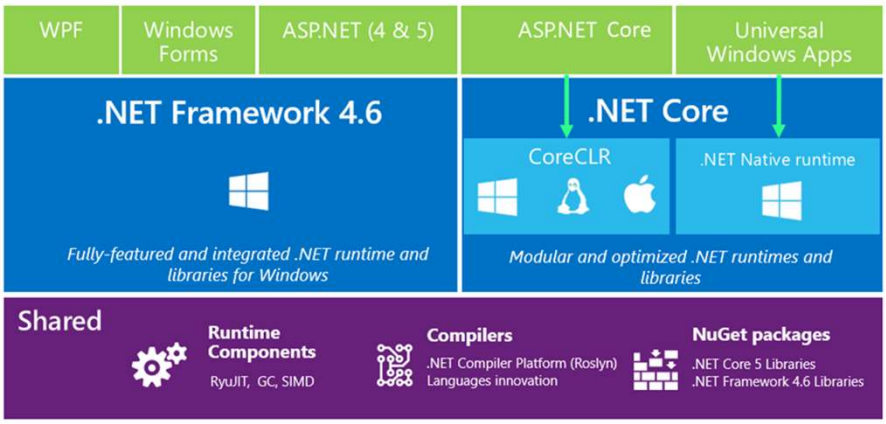
El archivo project.json permite describir elementos esenciales del paquete, entre otros cosas: versión, dependencias, y “runtimes” soportados. Al momento de efectuar un dotnet restore, se valida los paquetes realmente necesitados por la aplicación y este proceso genera una versión con las versiones verificadas de los paquetes a utilizar y el resultado es un archivo project.json.lock este archivo es mantenido por dotnet y no debería de manipularse manualmente.

En la dirección https://docs.microsoft.com/en-us/dotnet/articles/core/tools/project-json, se puede encontrar la documentación del esquema soportado por el archivo.

.NET Code, Programación



.NET Stack

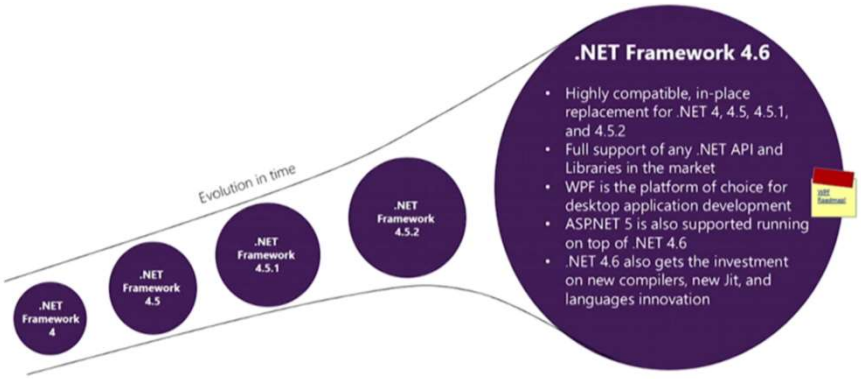

WPF
Windows Forms
ASP.NET (4 & 5)
ASP.NET Core
Universal Windows Apps
.NET Framework 4.6
Fully-featured and integrated .NET runtime and libraries for Windows
.NET Core
CoreCLR
.NET Native runtime
Modular and optimized .NET runtimes and libraries
Shared
Runtime Components
RyuJIT, GC, SIMD
Compilers
.NET Compiler Platform (Roslyn)
Languages innovation
NuGet packages
.NET Core 5 Libraries
.NET Framework 4.6 Libraries

¿Qué pasara con .NET 4.x?

- Hay muchas características de ASP.NET 4.x a las cuales se les seguirá dando soporte.
- No hay problema es que siga usando esa versión.
- Además mantiene al CLR completo.

.NET Roadmap

With over ~1B installations, we will continue to evolve .NET 4.x in a highly compatible manner




universidad
cenfotec_
tecnologías digitales