



# Introducción .NET Core

ISI Quevedo Fabricio



# Conceptos

- **.NET Core está escrito prácticamente desde cero**
- **.NET Core es *open source*.** El proyecto pertenece a la .NET Foundation y puede ser modificado por la comunidad.
- **.NET Core es multiplataforma.** Capacidad real de funcionar en múltiples plataforma. Los SDK y runtimes están disponibles para una gran variedad de sistemas operativos y distribuciones (Windows, macOS, Ubuntu, RHEL, Debian, Fedora, CentOS...), lo que hace que sea posible desarrollar aplicaciones sobre cualquiera de las plataformas soportadas y ejecutarlas también sobre cualquiera ellas. es compatible con distintas arquitecturas de procesador, como x64, x86 o ARM, lo que posibilita su uso en diversos tipos de dispositivo.

ARM, lo que posibilita su uso en diversos tipos de dispositivo.




# Conceptos

- **.NET Core es modular.** .NET Core está formado por distintas piezas distribuidas a través de paquetes NuGet. De esta forma, las correcciones a bugs o mejoras en componentes concretos pueden distribuirse y actualizarse de forma independiente al resto, sólo actualizando el paquete correspondiente.
- **Las operaciones principales de .NET Core se realizan desde línea de comandos.** Para conseguir ser cross platform real, era absolutamente imprescindible que las herramientas de .NET Core estuviesen disponibles en Windows, Linux y Mac. Denominador común: la línea de comandos.
- |                                 |                           |                           |
|---------------------------------|---------------------------|---------------------------|
| <code>dotnet new console</code> | <code>dotnet build</code> | <code>dotnet build</code> |
|---------------------------------|---------------------------|---------------------------|



# Conceptos

- .NET Core puede distribuirse de varias formas. .NET Core es más flexible: el framework puede estar instalado a nivel de equipo, como .NET Framework, pero también podemos hacerlo a nivel de usuario o incluso a nivel de aplicación (que cada aplicación incluya su propia copia de .NET Core). Además, distintas versiones y revisiones pueden convivir sin problemas en el mismo equipo. Por ejemplo, la siguiente traza de consola muestra el resultado de invocar el comando `dotnet --info`, donde aparecen las distintas versiones del SDK y runtimes instalados en el equipo.



```
C:\>dotnet --info
SDK de .NET Core (reflejando cualquier global.json):
Version: 2.2.102
Commit: 96ff75a873
```

```
Entorno de tiempo de ejecución:
OS Name: Windows
OS Version: 10.0.17763
OS Platform: Windows
RID: win10-x64
Base Path: C:\Program Files\dotnet\sdk\2.2.102\
```

```
Host (useful for support):
Version: 2.2.1
Commit: 878dd11e62
```

```
.NET Core SDKs installed:
1.1.11 [C:\Program Files\dotnet\sdk]
2.1.202 [C:\Program Files\dotnet\sdk]
2.1.503 [C:\Program Files\dotnet\sdk]
2.2.102 [C:\Program Files\dotnet\sdk]
```

```
.NET Core runtimes installed:
Microsoft.AspNetCore.All 2.1.7 [C:\Program Files\dotnet\shared\Microsoft.AspNetCore.All]
Microsoft.AspNetCore.All 2.2.1 [C:\Program Files\dotnet\shared\Microsoft.AspNetCore.All]
Microsoft.AspNetCore.App 2.1.7 [C:\Program Files\dotnet\shared\Microsoft.AspNetCore.App]
Microsoft.AspNetCore.App 2.2.1 [C:\Program Files\dotnet\shared\Microsoft.AspNetCore.App]
Microsoft.NETCore.App 1.0.13 [C:\Program Files\dotnet\shared\Microsoft.NETCore.App]
Microsoft.NETCore.App 1.1.10 [C:\Program Files\dotnet\shared\Microsoft.NETCore.App]
Microsoft.NETCore.App 2.0.9 [C:\Program Files\dotnet\shared\Microsoft.NETCore.App]
Microsoft.NETCore.App 2.1.7 [C:\Program Files\dotnet\shared\Microsoft.NETCore.App]
Microsoft.NETCore.App 2.2.1 [C:\Program Files\dotnet\shared\Microsoft.NETCore.App]
```

```
To install additional .NET Core runtimes or SDKs:
https://aka.ms/dotnet-download
```



# Conceptos

- **.NET Core no soporta todos los modelos de aplicación ni todos los frameworks** con .NET Core 2.x sólo podemos crear determinados tipos de aplicaciones: consola, web (ASP.NET Core) y Xamarin.
- En .NET Core 3.x se incorpora desarrollo de aplicaciones para Windows utilizando WPF, WinForms y Entity Framework 6 con .NET Core 3.x.
- En definitiva, a la hora de desarrollar con .NET Core debemos utilizar marcos de trabajo específicos para esta plataforma, como ASP.NET Core, Entity Framework Core o SignalR Core
- **.NET Core el rendimiento es algo prioritario** Algunas cifras, algo tan simple y recurrente como utilizar los métodos IndexOf() o StartsWith() sobre una cadena son un 200% más rápidos en .NET Core que en .NET Framework. Un ToString() sobre el elemento de un enum gana un 600% de rendimiento. LINQ es hasta un 300% más eficiente en determinados puntos. Lazy<T> es un 500% más rápido... Beneficiarios directos de estas mejoras son los frameworks específicos como ASP.NET Core o Entity Framework Core.



# Concepto

- **.NET Core ha sido creado pensando en los *tiempos modernos*. .NET Core y los frameworks contruidos sobre él han sido creados teniendo en cuenta aspectos como cloud, contenedores, microservicios, etc. Sus componentes están diseñados con el enfoque *cloud-first* en mente, por lo que se integran a la perfección en infraestructuras de nube. Se trata además de una tecnología especialmente recomendada para la construcción de microservicios, entre otros aspectos, por la facilidad con que pueden ser creados y distribuidos en contenedores.**



# Conceptos

- **Se pueden desarrollar aplicaciones .NET Core con cualquier editor o IDE.** Con .NET Core tenemos libertad completa a la hora de elegir el editor que utilizaremos en el desarrollo de nuestros proyectos. Si trabajamos sobre Windows, podemos seguir utilizando Visual Studio en el día a día, pero hay quien prefiere trabajar con Visual Studio Code, o con editores alternativos como Atom, Brackets o Sublime Text. Y si trabajamos desde Linux o Mac, podemos utilizar cualquier editor disponible en dichos entornos.





## Corolario

.NET Core se está convirtiendo en la herramienta de uso diario. Aunque en principio .NET Framework seguirá existiendo y Microsoft continuará dándole soporte, la tendencia actual indica que todo está confluyendo en .NET Core: Xamarin, WPF, WinForms, Entity Framework, SignalR o MVC son sólo algunos ejemplos de frameworks que ya están o van a estar pronto basados en la nueva infraestructura.

<https://www.campusmvp.es/recursos/post/10-diferencias-entre-net-core-y-net-framework.aspx>