plain concepts

Rediscover the meaning of

TECHNOLOGY

.NET CORE FUNDAMENTALS

1

.NET CORE

ÁNGEL CARLOS LÓPEZ QUEVEDO

Desarrollador en Plain Concepts
@_aclopez
aclopez@plainconcepts.com
https://github.com/acnagrellos

Web & App

Cloud

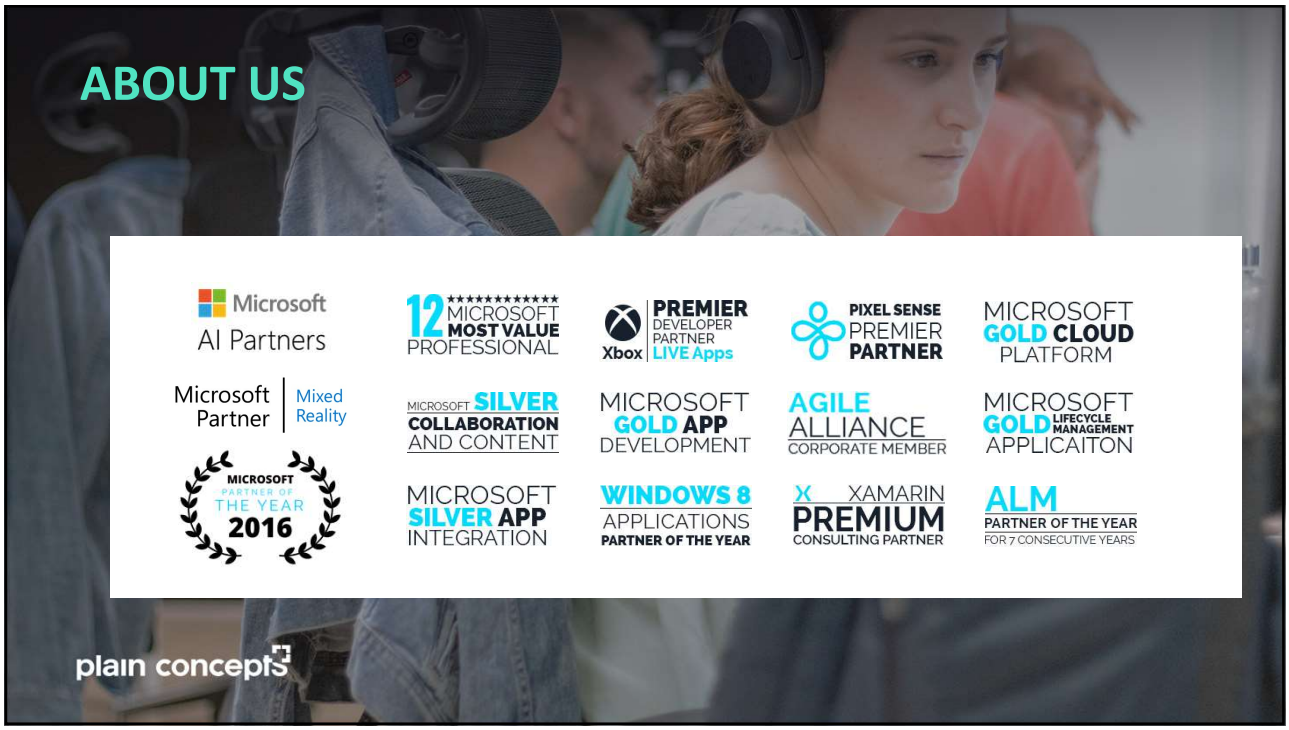
Devops

plain concepts

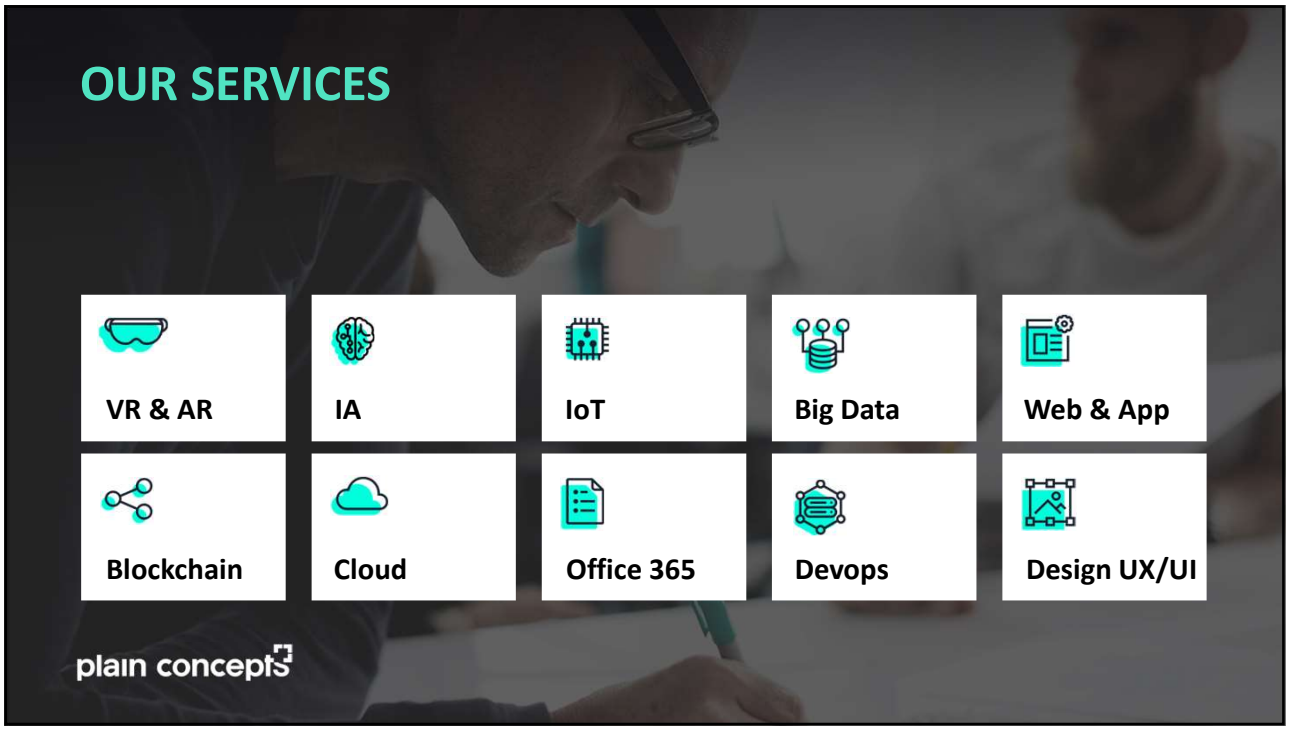
2


WE'RE HERE AND THERE
BARCELONA
SEVILLA
LONDON
BILBAO
DUBÁI
SEATTLE
MADRID
LEÓN
AMSTERDAM
FRANKFURT
CORUÑA
plain concepts

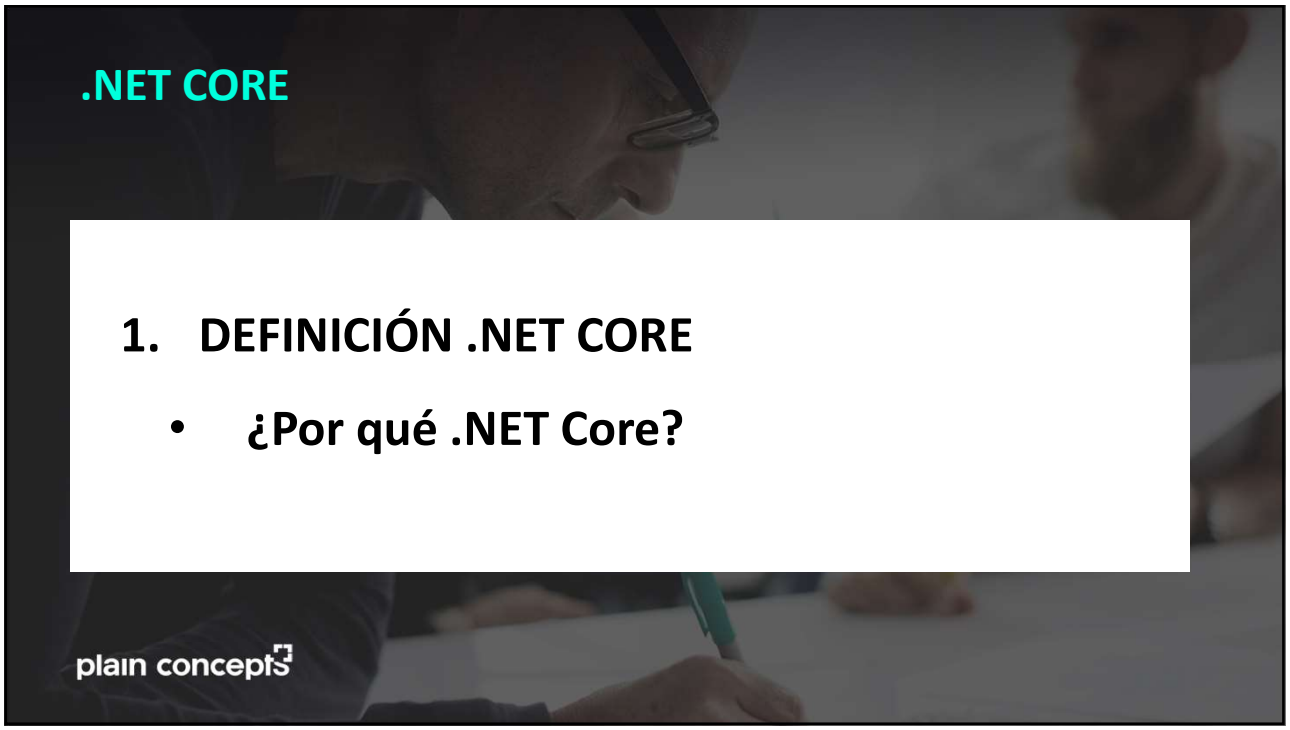
3


ABOUT US
Microsoft AI Partners
Microsoft Partner | Mixed Reality
MICROSOFT PARTNER OF THE YEAR 2016
12 MICROSOFT MOST VALUE PROFESSIONAL
Xbox PREMIER DEVELOPER PARTNER LIVE Apps
PIXEL SENSE PREMIER PARTNER
MICROSOFT GOLD CLOUD PLATFORM
MICROSOFT SILVER COLLABORATION AND CONTENT
MICROSOFT GOLD APP DEVELOPMENT
AGILE ALLIANCE CORPORATE MEMBER
MICROSOFT GOLD LIFECYCLE MANAGEMENT APPLICAITON
MICROSOFT SILVER APP INTEGRATION
WINDOWS 8 APPLICATIONS PARTNER OF THE YEAR
XAMARIN PREMIUM CONSULTING PARTNER
ALM PARTNER OF THE YEAR FOR 7 CONSECUTIVE YEARS
plain concepts

4


OUR SERVICES
VR & AR
IA
IoT
Big Data
Web & App
Blockchain
Cloud
Office 365
Devops
Design UX/UI
plain concepts

5

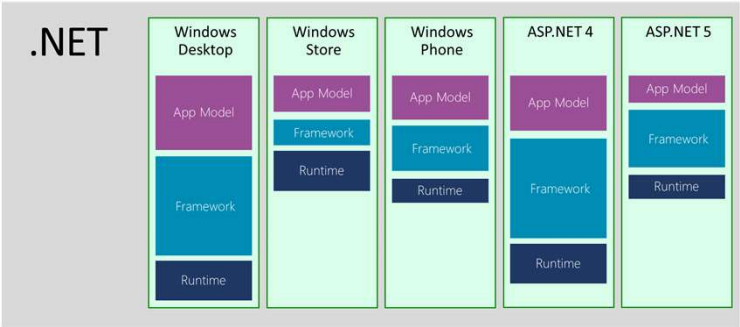

.NET CORE
1. DEFINICIÓN .NET CORE
• ¿Por qué .NET Core?
plain concepts

6

¿Por qué .NET Core?

.NET Framework

- Framework tradicional creado en 2002
- Solo tiene soporte en Windows
- Framework monolítico.
- High memory footprint. (sobre todo en móviles)
- Problemas en los deployments
- Su evolución dio lugar a un conjunto de Frameworks que tenían toda la estructura vertical con su propio Runtime y Framework dentro de ellos

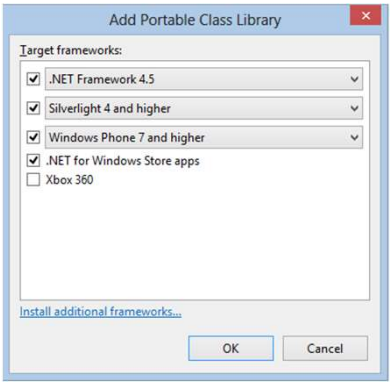


7

¿Por qué .NET Core?

PCL (Portable Class Library)

- Para resolver los problemas de fragmentación nacieron las **PCLs**
- Ofrece a los desarrolladores un **contrato común para todas las plataformas** que elijamos usando un único proyecto
- Se crea un **perfil de compilación**, que es una combinación de plataformas que comparten un API común
- La *desventaja* es que este perfil es **inmutable** en el tiempo.



8

¿Por qué .NET Core?

.NET Core

9

¿Qué es .NET Core?

.NET Core: Cross-platform

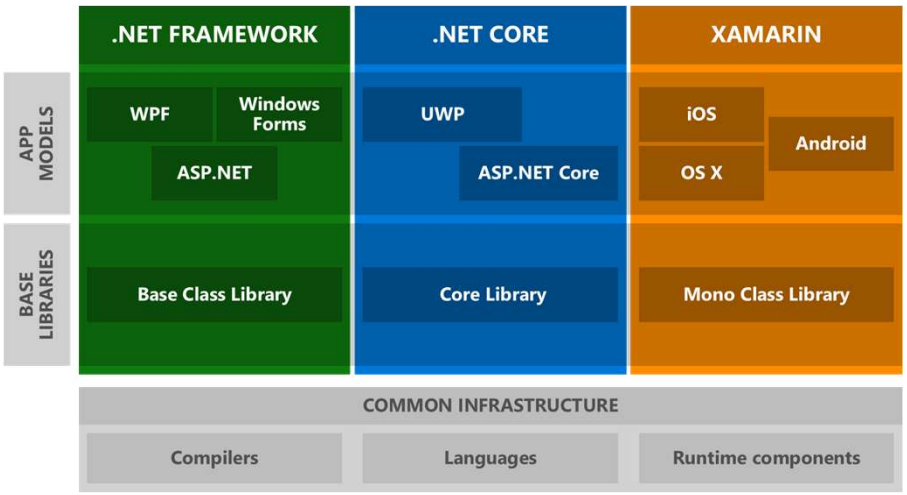
Cross-platform - .NET Core es soportado por los tres principales sistemas operativos: Linux, Windows and OS X. Hay otros proyectos para FreeBSD and Alpine.

- Las librerías .NET Core pueden ejecutarse sin modificaciones en los sistemas operativos compatibles..
- Las aplicaciones deben recompilarse por entorno
- Los usuarios pueden elegir el mejor entorno compatible con .NET Core para sus aplicaciones

10

¿Por qué .NET Core?

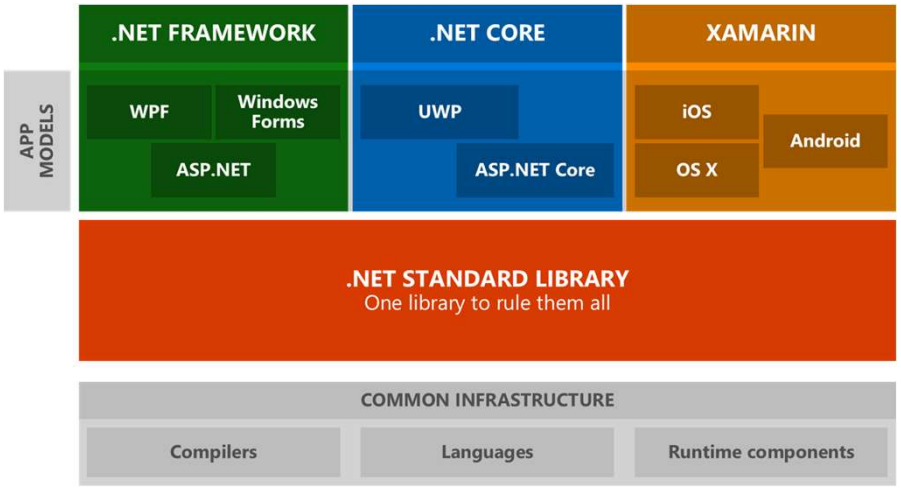
.NET Core: Nuevo ecosistema



11

¿Por qué .NET Core?

.NET Core: .NetStandard Library



12

¿Por qué .NET Core?

.NETStandard

.NET Platform	.NET Standard							
	1.0	1.1	1.2	1.3	1.4	1.5	1.6	2.0
.NET Core	→	→	→	→	→	→	1.0	vNext
.NET Framework	→	4.5	4.5.1	4.6	4.6.1	4.6.2	vNext	4.6.1
Xamarin.iOS	→	→	→	→	→	→	→	vNext
Xamarin.Android	→	→	→	→	→	→	→	vNext
Universal Windows Platform	→	→	→	→	10.0	→	→	vNext
Windows	→	8.0	8.1					
Windows Phone	→	→	8.1					
Windows Phone Silverlight	8.0							

13

¿Qué es .NET Core?

.NET Core: Open source

Open Source - .NET Core está disponible en Github, con licencia con el MIT y Apache 2. También hace uso de un conjunto significativo de dependencias de la industria de código abierto (ver notas de la versión). Ser OSS (Open Source Software) es fundamental para tener una comunidad próspera más "imprescindible" para muchas organizaciones donde OSS es parte de su estrategia de desarrollo.

14

¿Qué es .NET Core?

.NET Core: Natural acquisition

Natural acquisition - .NET Core está distribuido en una serie de paquetes Nugets que el usuario puede escoger acorde a sus necesidades. El runtime y la base del framework también se pueden obtener en NuGet y otros OS-specific package managers, como APT, Homebrew and Yum. Docker images están disponibles en docker hub. Las librerías del framework de alto nivel están disponibles en Nuget

15

¿Qué es .NET Core?

.NET Core: Smaller deployment footprint

Smaller deployment footprint:

- Incluso cuando en v1.0 el tamaño de .NET Core es mucho más pequeño que .NET Framework, tenga en cuenta que el tamaño total de .NET Core no tiene la intención de ser más pequeño que .NET Framework con el tiempo, pero dado que es de pago -para jugar, la mayoría de las aplicaciones que utilizan solo partes de CoreFX tendrán una huella de implementación más pequeña.

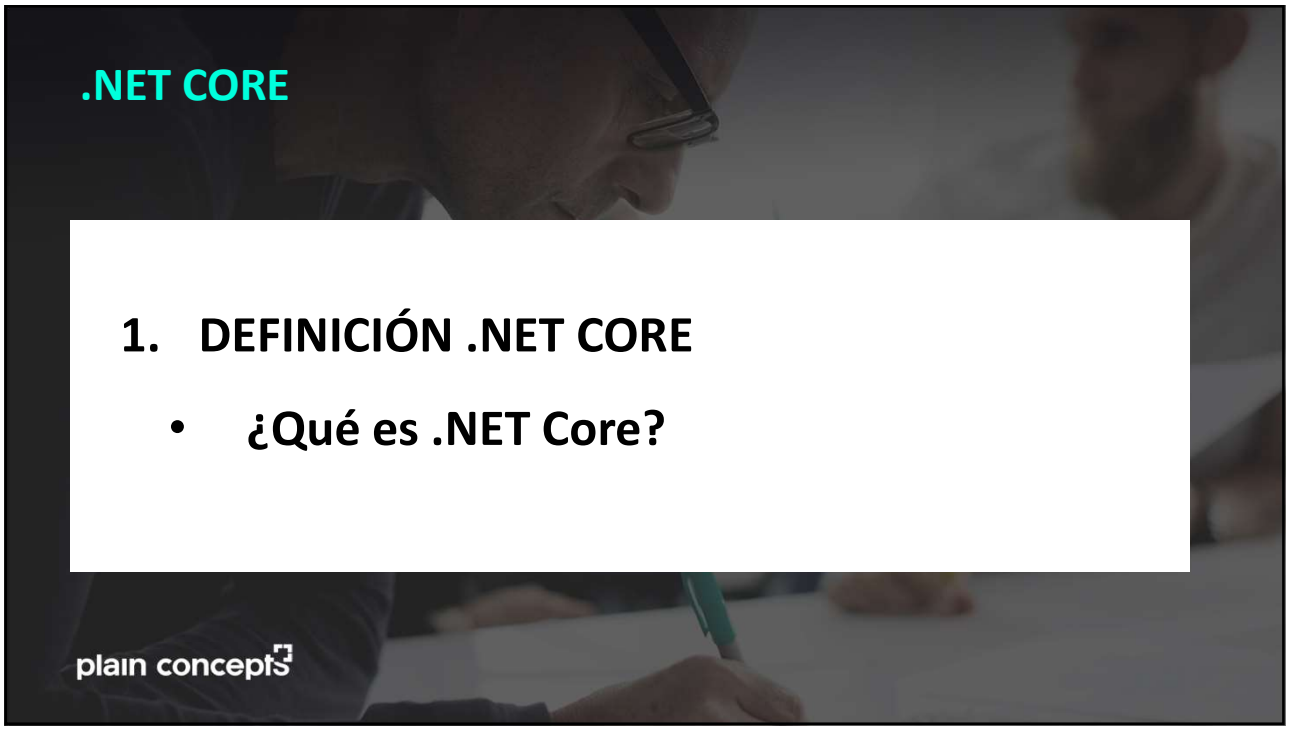
16

¿Por qué .NET Core?

Fast release cycles of .NET Core

Fast release cycles of .NET Core – .NET Core modular architecture plus its OSS nature provide a modern and much faster release cycles (even per NuGet package) compared to slow release cycles from larger monolithic frameworks. This approach allows a much faster innovation pace from Microsoft and the OSS .NET community than what was traditionally possible with the .NET Framework

17



18

¿Qué es .NET Core?

.NET Core

- Nueva version de .NET que es **cross-platform**, **open source** y **modular**
- Permite crear aplicaciones web, microservicios, librerías y aplicaciones de consola.
- Es un framework desacoplado de Windows, pudiendolo ejecutar en entorno non-Windows.
- Ahora la gran parte de su Core está en los paquetes Nugets, fuera del SDK.
- Es más ligero que .NET Framework.

19

¿Por qué .NET Core?

.NET Core: CoreFX

- A modular collection of libraries (Class library):
  - http://github.com/dotnet/corefx
  - Distributed as NuGet packages.
  - Less granularity in the last “change”.

20

¿Por qué .NET Core?

.NET Core: CoreCLR

- .NET Core runtime:
  - http://github.com/dotnet/coreclr
  - JIT compiler (RyuJIT)
    - 64 bits compiler
    - 2x faster
  - Garbage collector
  - MIT license
  - CoreCLR can be packed and deployed together with the application

21

¿Por qué .NET Core?

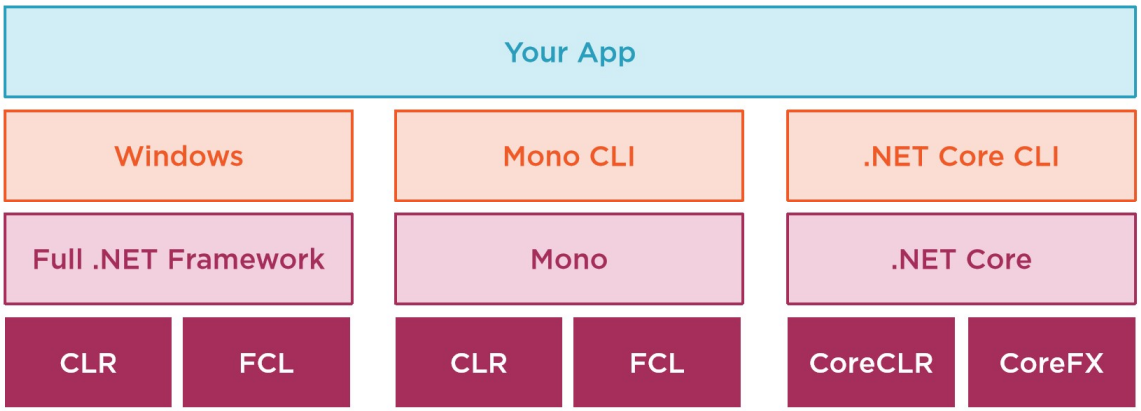
.NET Core: CoreRT

- NET Core runtime optimized for AOT (ahead of time compilation) scenarios.
  - http://github.com/dotnet/corert
  - AOT compiler
    - Single executable file.
    - Faster, faster, faster…
    - Xcopy deployments
    - Docker everywhere
    - Go territory – a cross-platform, natively-compiled, garbage-collected programming language, but C# rulez!!!

22

¿Por qué .NET Core?

.NET Core: Estructura



23

.NET CORE

1. DEFINICIÓN .NET CORE
   - Instalación

plain concepts

24

Instalación .NET Core

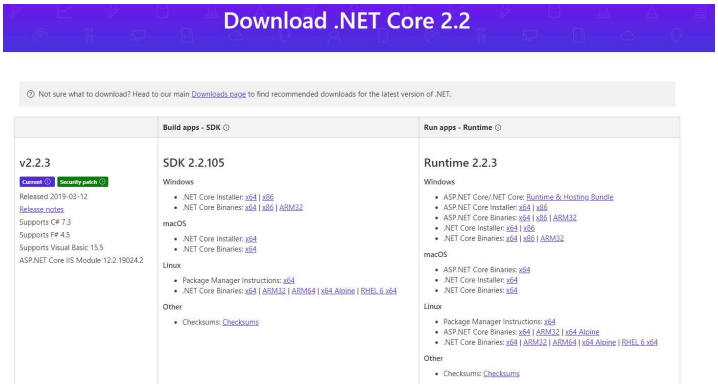
SDK/Tools

- Para desarrollar una app

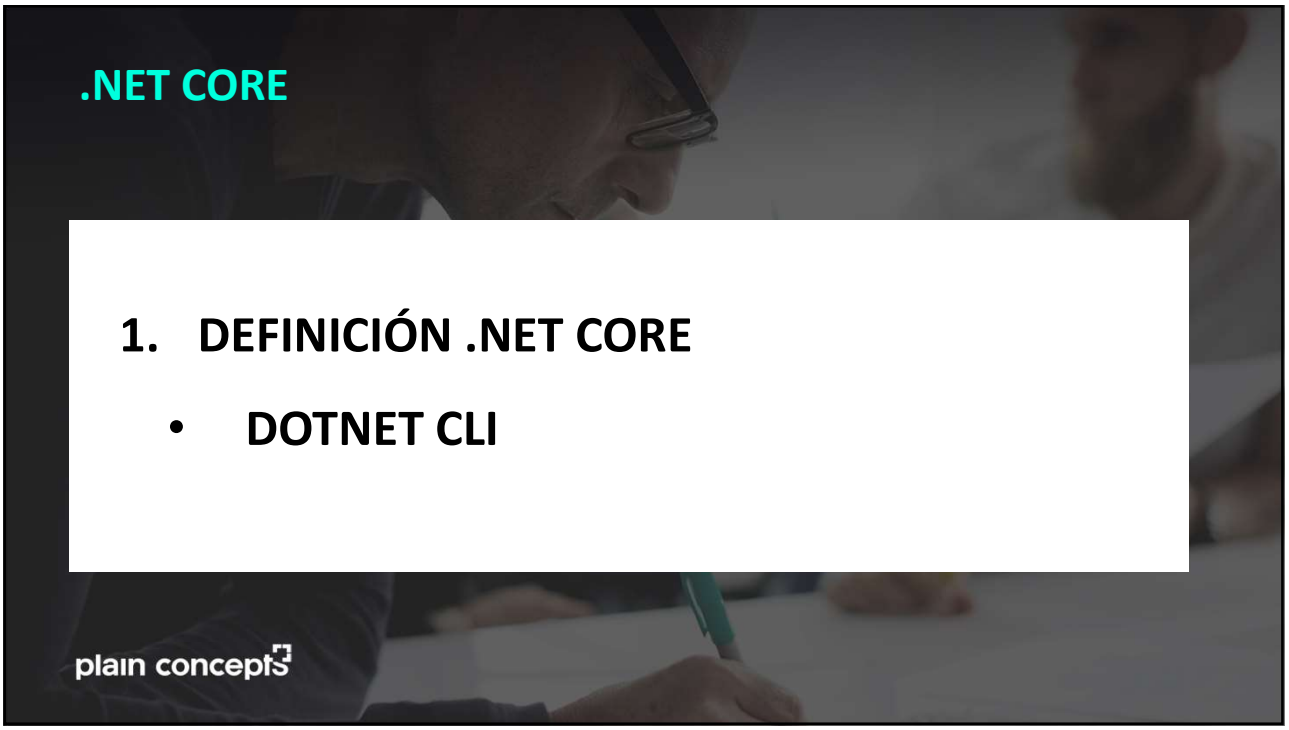
Runtime

- Para ejecutar una app en la máquina correspondiente.

https://dotnet.microsoft.com/download/dotnet-core/2.2



25



26

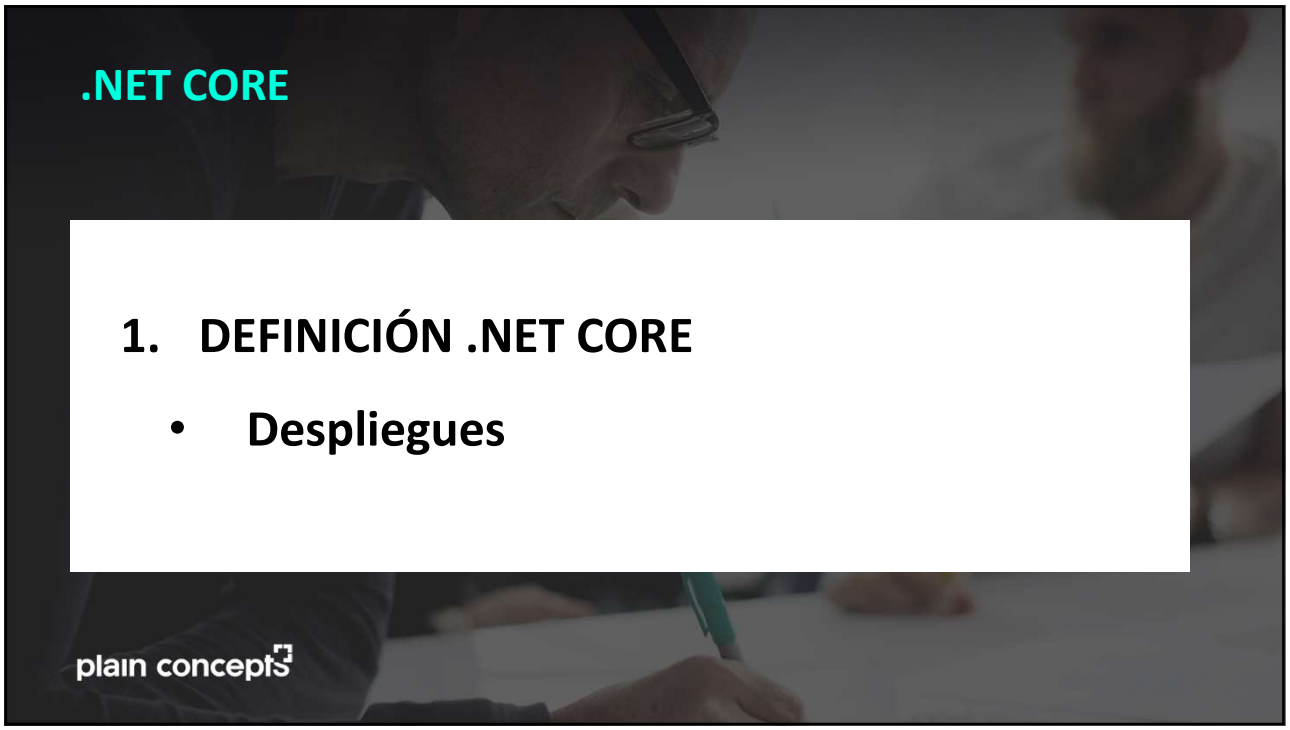
DOT NET CLI

Comandos

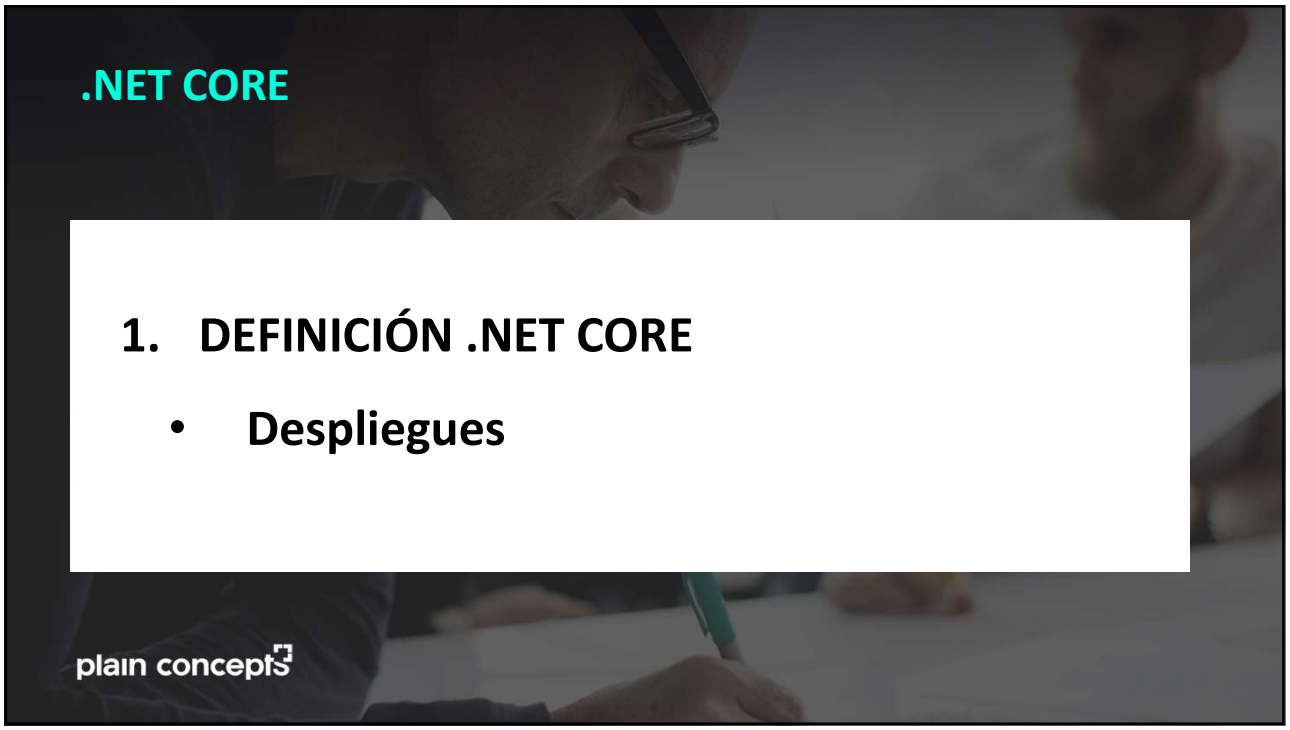
- dotnet
- dotnet new
- dotnet run
- dotnet build
- dotnet publish

27

28


.NET CORE
1. DEFINICIÓN .NET CORE
• Despliegues
plain concepts

29


.NET CORE
1. DEFINICIÓN .NET CORE
• Despliegues
plain concepts

30

.NET CORE Deployment

Deployment

Se pueden hacer dos tipos de Deployment en las aplicaciones .NET Core:

- **Framework-dependent deployment** (A.K.A. **FDD**): Este sería el deployment que depende de que en la máquina donde se vaya a desplegar la aplicación éste instalado el Runtime de .NET Core para que todo pueda funcionar bien. La aplicación a desplegar solo contendría su propio código y el código de las librerías que se han usado de terceros; pero no el código propio de .NET Core. (Es el enfoque por defecto).
- **Self-container deployment** (A.K.A. **SCD**): Al contrario que FDD este despliegue contiene todas las librerías propias de .NET Core y no hay ninguna dependencia en donde pongamos el despliegue para que pueda funcionar.

31

.NET CORE Deployment

Framework-dependent deployment

Ventajas:

- No se tiene que definir los sistemas operativos de destino con los que se ejecutará su aplicación .NET Core por adelantado
- El tamaño del despliegue es más pequeño.
- Varias aplicaciones utilizan la misma instalación .NET Core, lo que reduce tanto el espacio en disco como el uso de memoria en los sistemas host.

Desventajas:

- Su aplicación solo puede ejecutarse si la versión de .NET Core a la que apunta, o una versión posterior, ya está instalada en el sistema host.
- Es posible que el tiempo de ejecución .NET Core y las bibliotecas cambien sin su conocimiento en futuras versiones. En casos raros, esto puede cambiar el comportamiento de su aplicación.

32

33

.NET CORE Deployment

Self-container deployment

Ventajas:

- Tiene el control exclusivo de la versión de .NET Core que se implementa con su aplicación. .NET Core y solo tú la puedes cambiar o restaurar.
- Puede estar seguro de que el sistema de destino puede ejecutar su aplicación .NET Core, ya que está proporcionando la versión de .NET Core con la que se ejecutará.

Desventajas:

- Debido a que .NET Core está incluido en el paquete de implementación, debes seleccionar las plataformas de destino para las que se creen paquetes de implementación con anticipación.
- El tamaño de la implementación es mayor
- La implementación de numerosas aplicaciones .NET Core autocontenidas en un sistema puede consumir una gran cantidad de espacio en disco, ya que cada aplicación duplica los archivos .NET Core.

34

35