PREPÁRATE
PARA SER EL
MEJOR

+ ENTREMIENTO
EXPERIENCIA

BIENVENIDOS.

+ ENTRENAMIENTO
EXPERIENCIA

GALAXY
TRAINING

ASP.NET CORE WEB APPLICATIONS: FUNDAMENTALS

Sesión 01

Ing. Erick Arostegui
Instructor
earostegui@galaxy.edu.pe

AGENDA

PLATAFORMA.NET Y AMBIENTE DE DESARROLLO

- Plataforma .NET (NET Framework, NET Core y ASP.NET Core).
- Principios SOLID y MVC Pattern (Model, View and Controller).
- ASP.NET Core MVC.
- Visual Studio 2017(soluciones, proyectos) e IIS (despliegue interno y externo).
- NET Core CLI, NuGet, Git y GitHub (consola, gestor de paquetes y versiones).

.NET

.NET es una plataforma de desarrollo gratuita, multiplataforma y de código abierto para crear muchos tipos diferentes de aplicaciones.

Con .NET, puede usar varios idiomas, editores y bibliotecas para crear para web, móvil, escritorio, juegos e IoT.

https://dotnet.microsoft.com/

Plataforma .NET (NET Framework, NET Core y ASP.NET Core).

Basico

.NET

Web
Build web apps and services for Windows, Linux, macOS, and Docker.

Mobile
Use a single codebase to build native mobile apps for iOS, Android, and Windows.

Desktop
Create beautiful and compelling desktop apps for Windows and macOS.

Microservices
Create independently deployable microservices that run on Docker containers

Gaming
Develop 2D and 3D games for the most popular desktops, phones, and consoles.

Machine Learning
Add vision algorithms, speech processing, predictive models, and more to your apps.

Cloud
Consume existing cloud services, or create and deploy your own.

Internet of Things
Make IoT apps, with native support for the Raspberry Pi and other single-board computers.

.NET Framework

Es un componente de software desarrollado por Microsoft que puede ser instalado en **Windows**. Provee soluciones pre-codificadas comunes de los programas y gestiona la ejecución de programas escritos para este Framework.

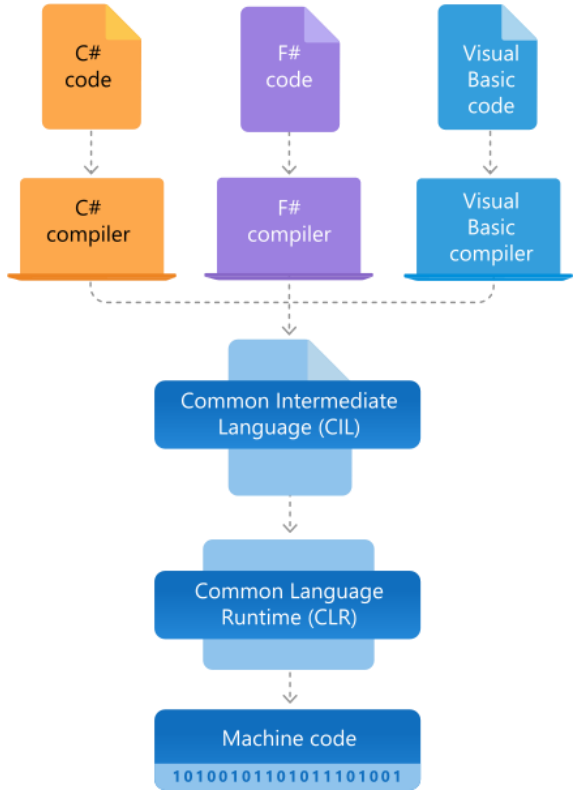
https://docs.microsoft.com/es-es/dotnet/framework/get-started/index#Introducing

.NET Frameworks

.NET Framework

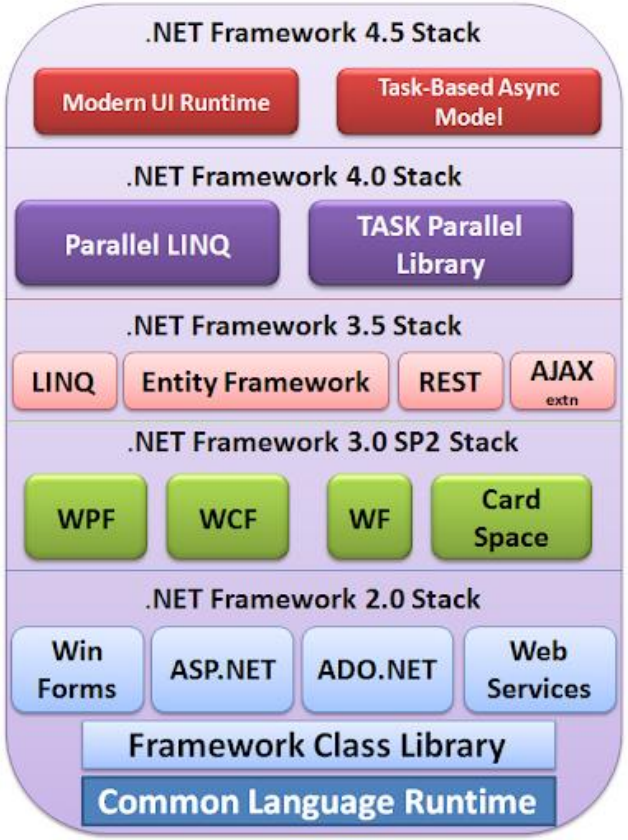
- **Common Language Runtime (CLR)** es el motor de ejecución que controla las aplicaciones en ejecución. Proporciona servicios como administración de subprocesos, recolección de elementos no utilizados, seguridad de tipos, control de excepciones y mucho más.

- **La biblioteca de clases** proporciona un conjunto de API's y tipos de funcionalidad común. Proporciona tipos para cadenas, fechas, números, etc. La biblioteca de clases incluye API's para leer y escribir archivos, conectarse a bases de datos, dibujar y mucho más.



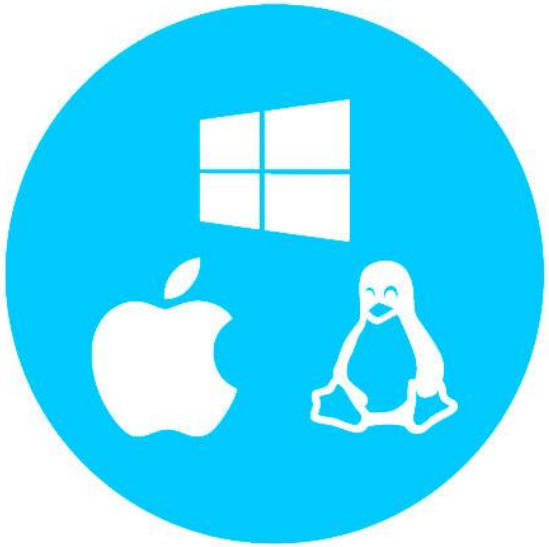
.NET Framework

Versión actual
NET Framework 4.8



.NET Core

Framework Open Source desarrollado por Microsoft para la creación de aplicaciones web y disponible para versiones Linux, Windows y Mac. Integrado con contenedores (Docker).



.NET Core 3.1

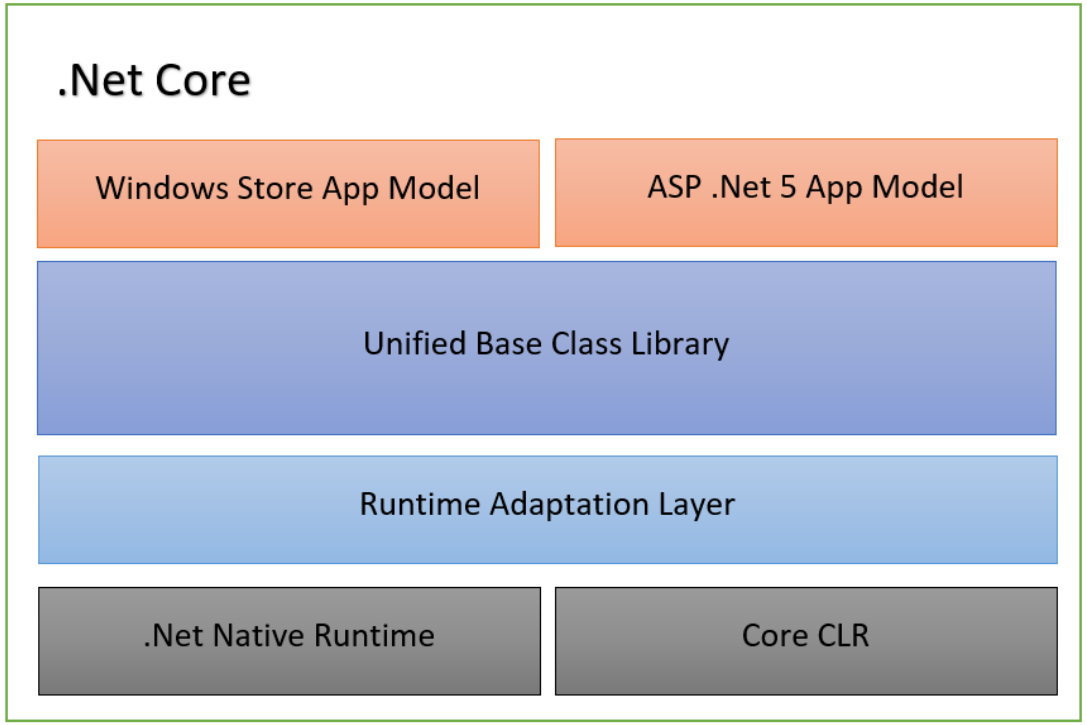
Welcome to .NET Core

https://dotnet.github.io/

.NET Core

- **Core CLR** es Common Language Runtime optimizado para implementaciones multiplataforma y basadas en la nube. Esto, junto con .Net Native Runtime, constituye la base de todas las plataformas basadas en .Net.

- **.Net Native Runtime** También conocido como Managed Runtime, .Net Native Runtime contiene las bibliotecas basadas en Windows nativas. Esto también contiene la compilación Ahead Of Time (AOT) en lugar de la compilación Just In Time (JIT) de antwhile. Esto mejora el rendimiento de las aplicaciones.



Ventajas de .NET Framework

- Si es un desarrollador de .NET que necesita compilar y lanzar algo rápido y no tiene tiempo para aprender .NET Core, .NET Framework es su elección. .NET Core tiene una curva de aprendizaje ligeramente superior.

- Si mantiene y actualiza aplicaciones .NET existentes, .NET Framework es su elección. La portabilidad de una aplicación .NET existente a una aplicación de .NET Core requiere cierto trabajo.

- .NET Framework es lo que es. Se supone que la versión actual de .NET Framework, 4.8, es la última versión de .NET Framework. No habrá más nuevas versiones de .NET Framework planeadas en el futuro.

Ventajas de .NET Core

- Si va a crear una nueva aplicación y tiene la opción entre .NET Core y .NET Framework, .NET Core es el camino a seguir.

- .NET Core 3.x ahora es compatible con WPF y Windows Forms, también admite el desarrollo cruzado entre UWP, WPF y Windows Forms. Esto proporciona a los desarrolladores la flexibilidad para incorporar interfaces modernas de UWP en Windows Forms y WPF.

- .NET Core es más adecuado para necesidades multiplataforma. Las aplicaciones de .NET Core son compatibles con Windows, Linux y macOS. El popular editor de código abierto de Microsoft, Visual Studio Code, es compatible con Windows, Linux y macOS. VS Code admite las necesidades modernas de los editores de código, incluidos IntelliSense y la depuración. La mayoría de los editores de terceros, como Sublime, Emacs y VI, trabajan con .NET Core.

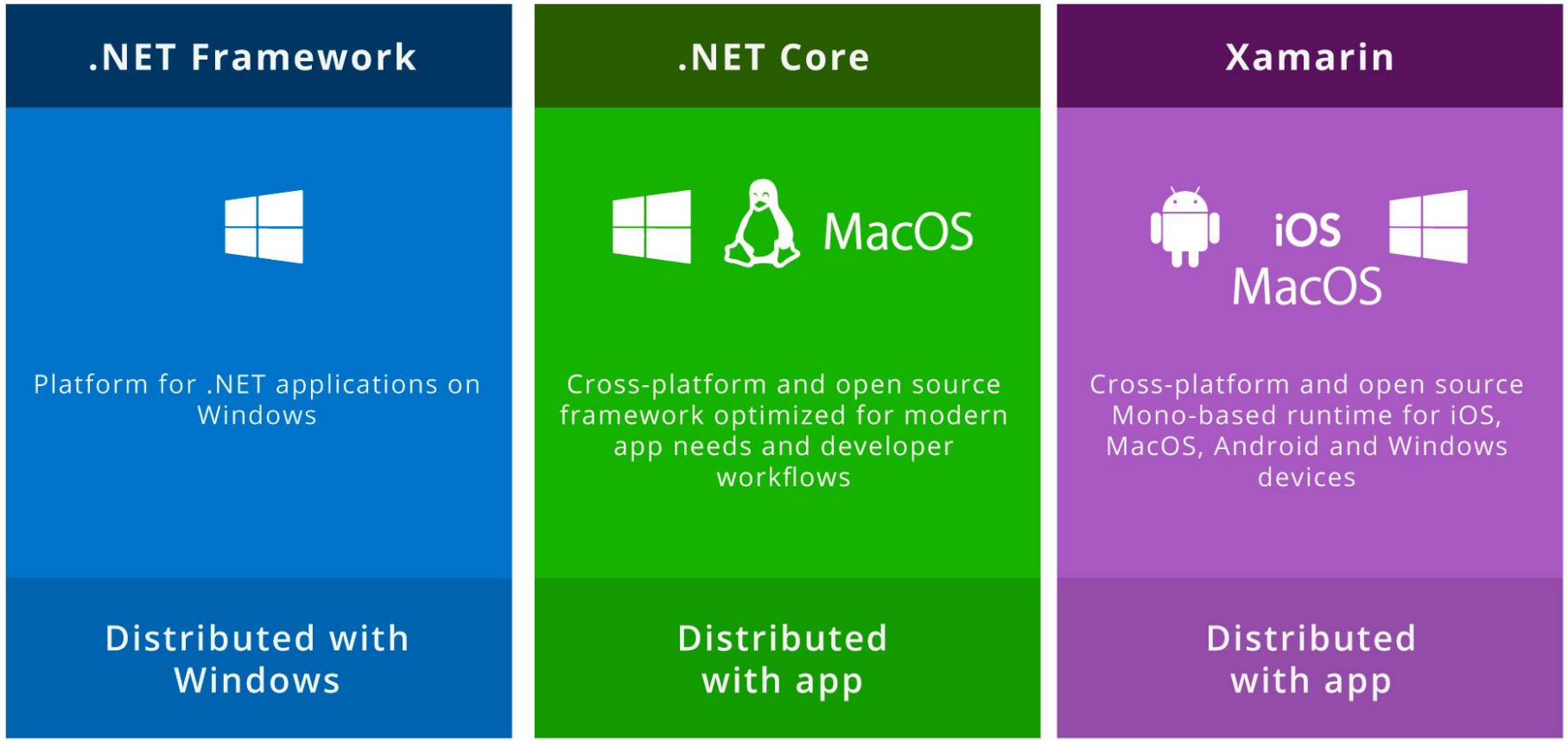
Ventajas de .NET Core

- La arquitectura de microservicios se admite en .NET Core, que permite que los servicios multiplataforma funcionen con .NET Core, incluidos los servicios desarrollados con .NET Framework, Java, Ruby u otros.

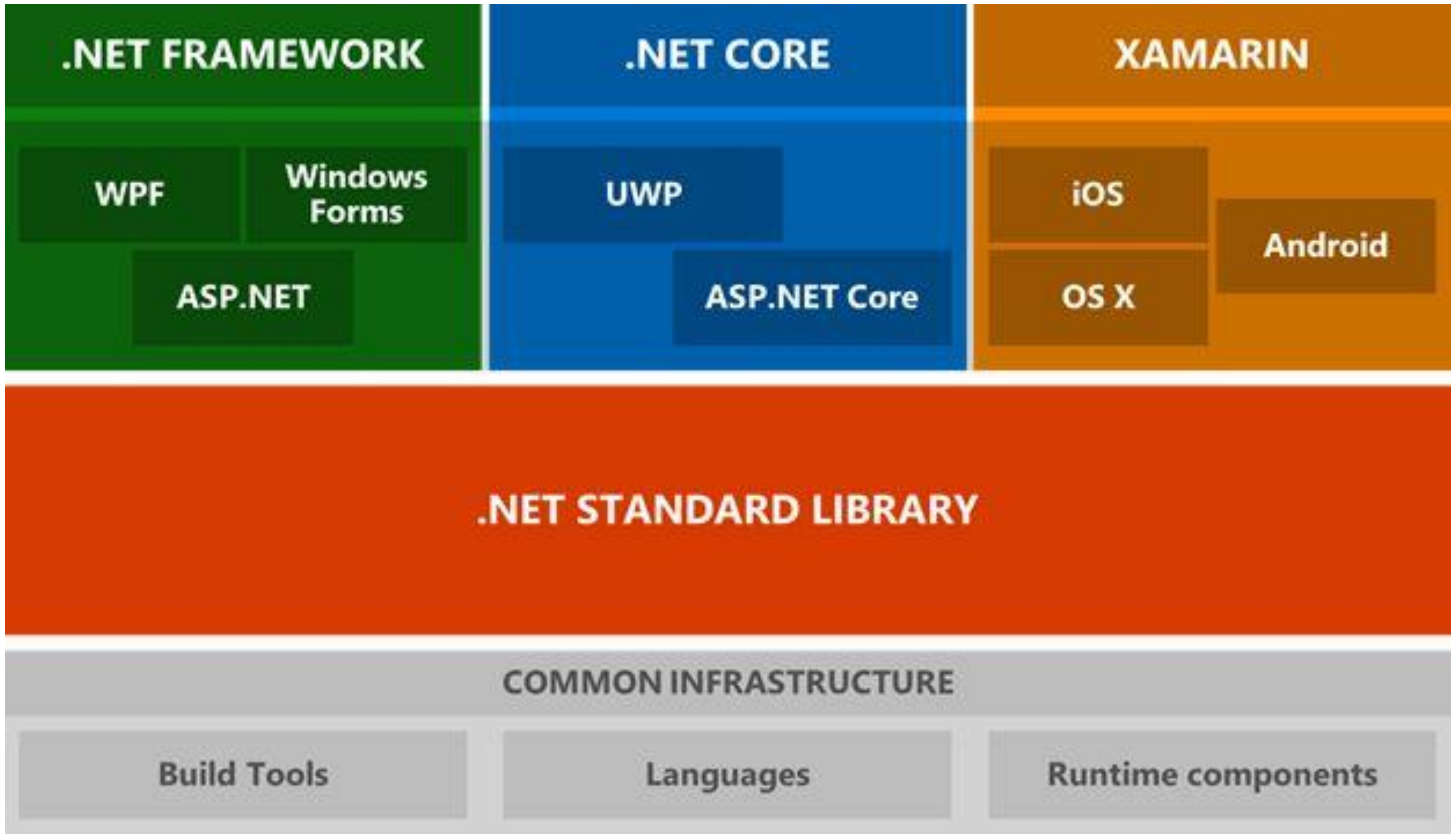
- Los contenedores son las máquinas virtuales de hoy en día. La modularidad, el peso ligero y la flexibilidad de .NET Core facilitan la implementación de aplicaciones de .NET Core en contenedores. Los contenedores se pueden implementar en cualquier plataforma, nube, Linux y Windows. .NET Core funciona bien con Docker y Azure Kubernetes Service.

- El rendimiento y la escalabilidad fueron las dos principales áreas de enfoque clave cuando se desarrolló .NET Core. .NET Core y ASP.NET Core son los marcos web de mejor rendimiento según algunos puntos de referencia.

El Ecosistema NET



El Ecosistema NET



Plataforma .NET (NET Framework, NET Core y ASP.NET Core).

Basico

.NET foundation

Openness

Community

Rapid innovation

.NET Compiler Platform ("Roslyn")
LLILC
MVVM Light Toolkit
MSBuild
ASP.NET MVC
MEF
IdentityManager
ASP.NET Core
.NET SDK for Hadoop
Kudu
.NET Core
.NET Micro Framework
Mono
Mailkit
Cecil
Xamarin.Auth
Mimekit
Umbraco
ASP.NET AJAX Control Toolkit
Open Live Writer
NuGet
Cake
WorldWide Telescope
Couchbase Lite for .NET
ASP.NET SignalR
ASP.NET Web Pages
Microsoft Azure SDK for .NET
WCF
Xamarin SDK
Entity Framework
Open XML SDK
IdentityServer
Microsoft Azure WebJobs SDK
OWIN Authentication Middleware
Microsoft Web Protection Library
ASP.NET Web API
Prism
System.Drawing
Orchard CMS
ProtoBuild
Xamarin.Mobile
Salesforce Toolkits for .NET
Orleans

Features

Protection
- Licenses
- Copyrights
- Trademarks
- Patents

Practices
- Mentorship
- Governance
- Feedback
- Co-ordination

Visibility
- Media
- Branding
- Events

Support
- Hosting
- Code signing
- CLA Management
- Swag

Entonces, ¿qué elegir?

.NET Framework es una mejor opción si:

- No tiene tiempo para aprender nuevas tecnologías.
- Necesita un entorno estable para trabajar.
- Tener horarios de lanzamiento más cercanos.
- Ya estamos trabajando en una aplicación existente y ampliando su funcionalidad.
- Ya tiene un equipo existente con experiencia en .NET y software listo para la producción.
- No desea lidiar con actualizaciones y cambios continuos.
- Creación de aplicaciones cliente de Windows mediante Windows Forms o WPF

.NET Core es una mejor opción si:

- Desea orientar sus aplicaciones en sistemas operativos Windows, Linux y Mac.
- No tienen miedo de aprender cosas nuevas.
- No tienen miedo de romper y arreglar las cosas, ya que .NET Core aún no está completamente madurado.
- Un estudiante que acaba de aprender .NET.
- Me encanta el código abierto.

Entonces, ¿qué elegir?

Un sistema escalable y de alto rendimiento sin interfaz de usuario	.NET Core es mucho más rápido.
Compatibilidad con contenedores Docker	Ambos, pero .NET Core nace para vivir en un contenedor.
Depende en gran medida de la línea de comandos	.NET Core tiene mejor compatibilidad.
Necesidades multiplataforma	.NET Core
Uso de microservicios	Ambos, pero .NET Core está diseñado para tener en cuenta las necesidades actuales.
Aplicaciones web centradas en la interfaz de usuario	.NET Framework es mejor ahora hasta que .NET Core se ponga al día.
Aplicaciones cliente de Windows con Windows Forms y WPF	.NET Framework
Ya tienen un entorno y sistemas preconfigurados	.NET Framework es mejor.
Versión estable para una necesidad inmediata de construir e implementar	.NET Framework ha existido desde 2001. .NET Core es solo un bebé.
Tener un equipo de .NET experimentado existente	.NET Core tiene una curva de aprendizaje.
El tiempo no es un problema. Los experimentos son aceptables. No hay prisa por la implementación.	.NET Core es el futuro de .NET.

El futuro de .NET

- .NET Schedule
  - Major releases every year, LTS for even numbered releases
  - Predictable schedule, minor releases if needed

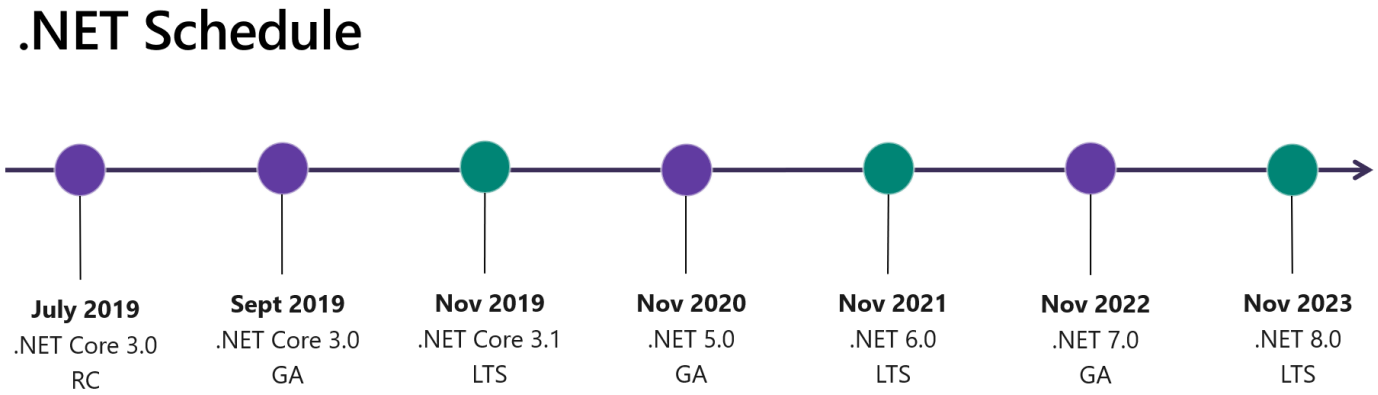
El futuro de .NET

Actualmente, hay dos versiones de .NET .NET Framework y .NET Core. En el futuro, solo habrá una versión de .NET y es decir, .NET 5. .NET 5 toma lo mejor de .NET Core, .NET Framework, Xamarin y Mono para proporcionar bibliotecas, API y tiempo de ejecución para compilar aplicaciones para dispositivos Windows, Web, Mobile e IoT.

C# 8.0 es la versión más reciente del lenguaje de C# que se admite en Visual Studio 2019. Si tiene previsto compilar una nueva aplicación en .NET, debe usar .NET Core 3.x (la versión actual) y C# 8, que más adelante se convertirá en un proyecto de .NET 5. Asumo que no tendrá cambios mínimos o no.

El futuro de .NET

El futuro de .NET



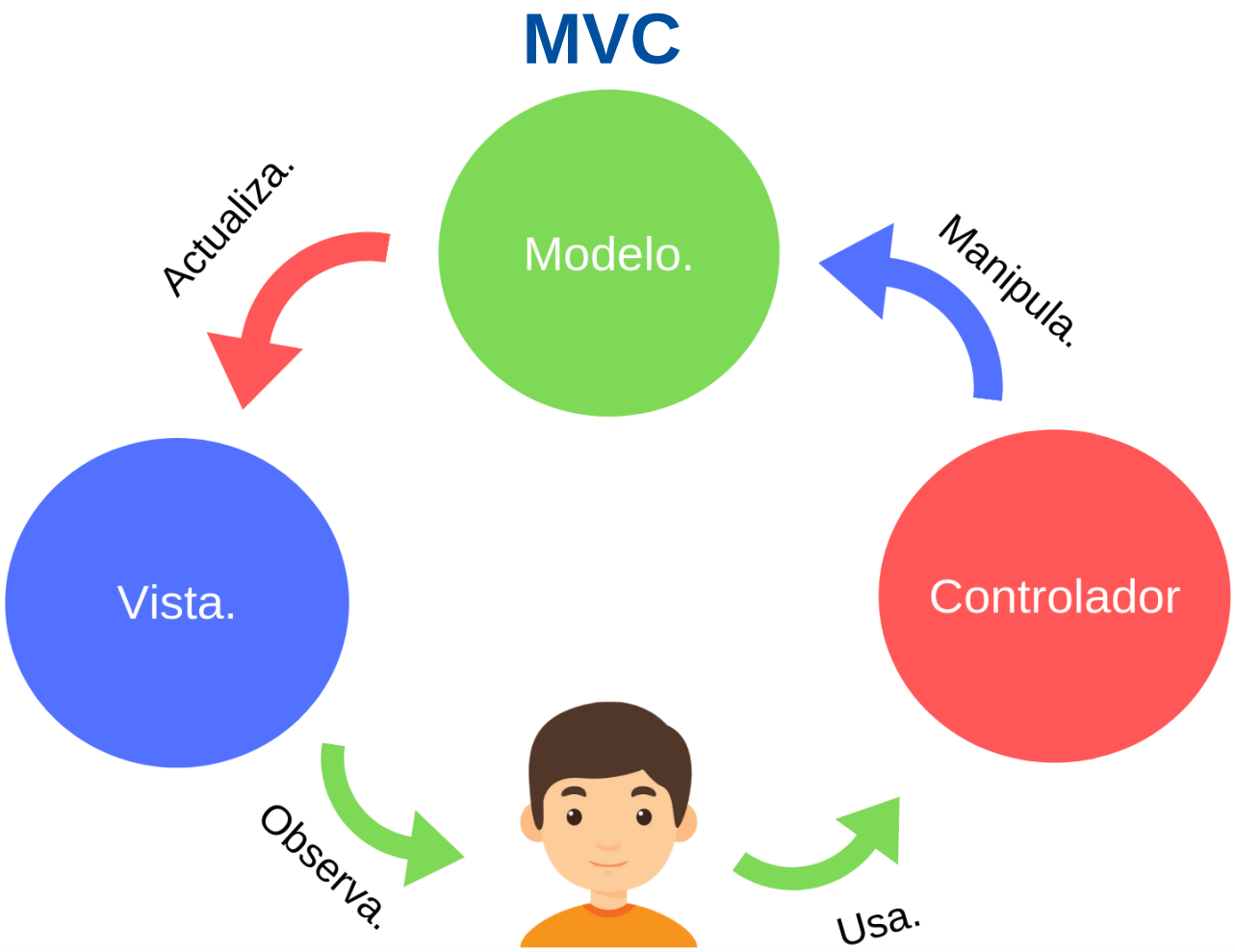
- .NET Core 3.0 release in September
- .NET Core 3.1 = Long Term Support (LTS)
- .NET 5.0 release in November 2020
- Major releases every year, LTS for even numbered releases
- Predictable schedule, minor releases if needed

https://devblogs.microsoft.com/dotnet/introducing-net-5/

Principios SOLID

- **S** • Single Responsibility Principle
- **O** • Open/Closed Principle
- **L** • Liskov Substitution Principle
- **I** • Interface Segregation Principle
- **D** • Dependency Inversion Principle

Plataforma .NET (NET Framework, NET Core y ASP.NET Core).

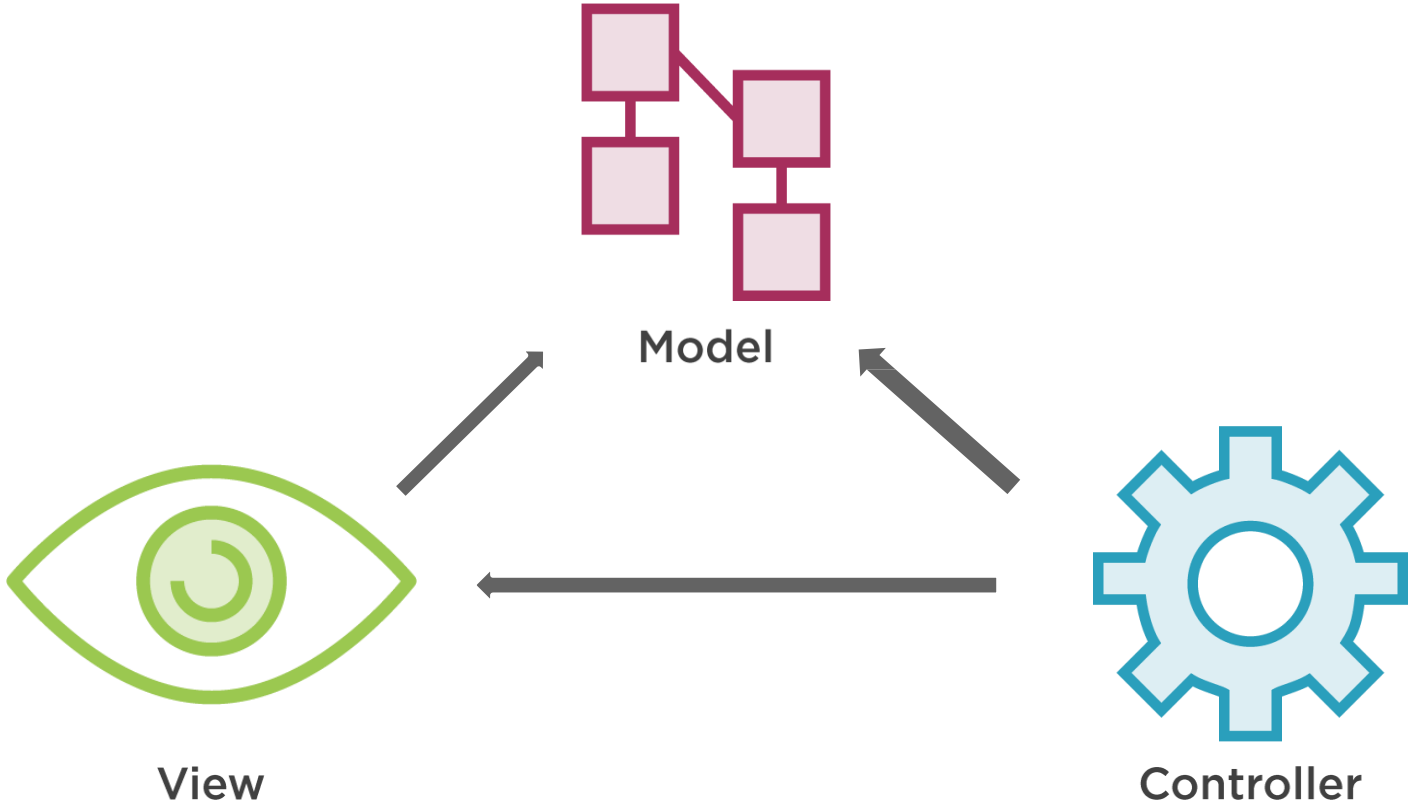


Patron MVC

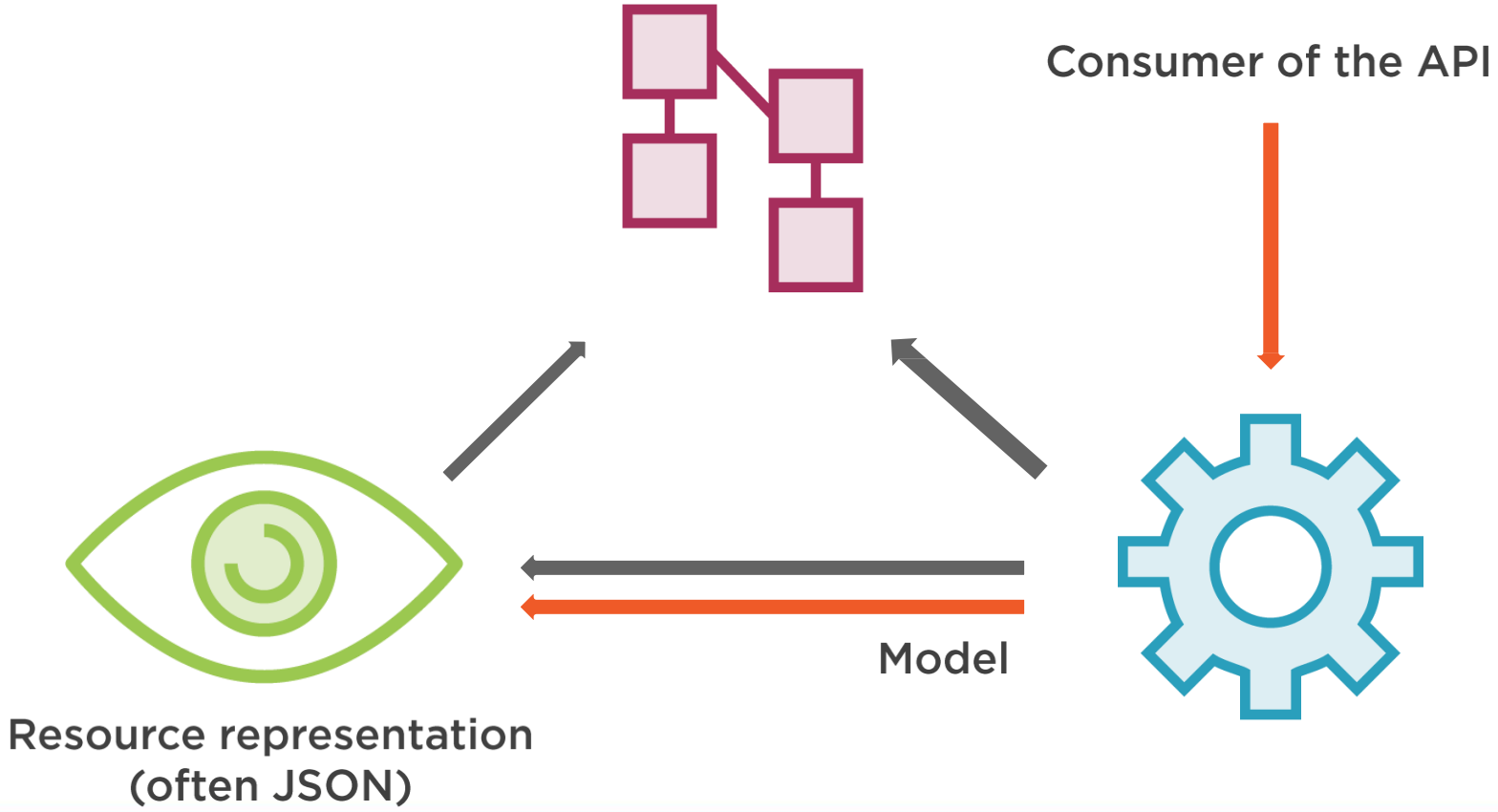
El patrón de arquitectura del controlador de vista de modelos (MVC) separa una aplicación en tres grupos de componentes principales: modelos, vistas y controladores. Este patrón permite lograr la separación de responsabilidades. Con este patrón, las solicitudes del usuario se enrutan a un controlador que se encarga de trabajar con el modelo para realizar las acciones del usuario o recuperar los resultados de consultas. El controlador elige la vista para mostrar al usuario y proporciona cualquier dato de modelo que sea necesario.

Con esta delineación de responsabilidades es más sencillo escalar la aplicación, porque resulta más fácil codificar, depurar y probar algo (modelo, vista o controlador) que tenga un solo trabajo. Es más difícil actualizar, probar y depurar código que tenga dependencias repartidas entre dos o más de estas tres áreas. Por ejemplo, la lógica de la interfaz de usuario tiende a cambiar con mayor frecuencia que la lógica de negocios. Si el código de presentación y la lógica de negocios se combinan en un solo objeto, un objeto que contenga lógica de negocios deberá modificarse cada vez que cambie la interfaz de usuario. A menudo esto genera errores y es necesario volver a probar la lógica de negocio después de cada cambio mínimo en la interfaz de usuario.

Patron Model-View-Controller (API)



Patron Model-View-Controller (API)



ASP.NET Core MVC.

El framework de ASP.NET Core MVC es un framework de presentación ligero, de código abierto y con gran capacidad de prueba, que está optimizado para usarlo con ASP.NET Core.

ASP.NET Core MVC ofrece una manera basada en patrones de crear sitios web dinámicos que permitan una clara separación de intereses. Proporciona control total sobre el marcado, admite el desarrollo controlado por pruebas (TDD) y usa los estándares web más recientes.

- Enrutamiento
- Enlace de modelos
- Validación de modelos
- Inserción de dependencias
- Filtros
- Áreas
- API web
- Capacidad de prueba
- Motor de vistas de Razor
- Vistas fuertemente tipadas
- Asistentes de etiquetas
- Componentes de vista

https://docs.microsoft.com/es-es/aspnet/core/mvc/overview?view=aspnetcore-3.1

ASP.NET Core MVC.

El framework de ASP.NET Core MVC es un framework de presentación ligero, de código abierto y con gran capacidad de prueba, que está optimizado para usarlo con ASP.NET Core.

ASP.NET Core MVC ofrece una manera basada en patrones de crear sitios web dinámicos que permitan una clara separación de intereses. Proporciona control total sobre el marcado, admite el desarrollo controlado por pruebas (TDD) y usa los estándares web más recientes.

- Enrutamiento
- Enlace de modelos
- Validación de modelos
- Inserción de dependencias
- Filtros
- Áreas
- API web
- Capacidad de prueba
- Motor de vistas de Razor
- Vistas fuertemente tipadas
- Asistentes de etiquetas
- Componentes de vista

https://docs.microsoft.com/es-es/aspnet/core/mvc/overview?view=aspnetcore-3.1

Visual Studio 2017 e Internet Information Services

Basico

Visual Studio 2017(soluciones, proyectos) e Internet Information Services (IIS) (despliegue interno y externo).

NET Core CLI, NuGet, Git y GitHub (consola, gestor de paquetes y versiones).

Es una extensión de Visual Studio que permite gestionar librerías en NET. Sus herramientas cliente permiten producir y consumir paquetes.

https://www.nuget.org/

NPM

NPM = Node Package Manager

NET Core CLI, NuGet, Git y GitHub (consola, gestor de paquetes y versiones).

Basico

- Task Runners
- Ejecucion en node.js
- Plugins
- Escribir las tareas en javascript
- Integrados con Visual Studio

NET Core CLI, NuGet, Git y GitHub (consola, gestor de paquetes y versiones).


GALAXY
TRAINING