educación IT
CODE YOUR FUTURE

Programación Web .NET Core

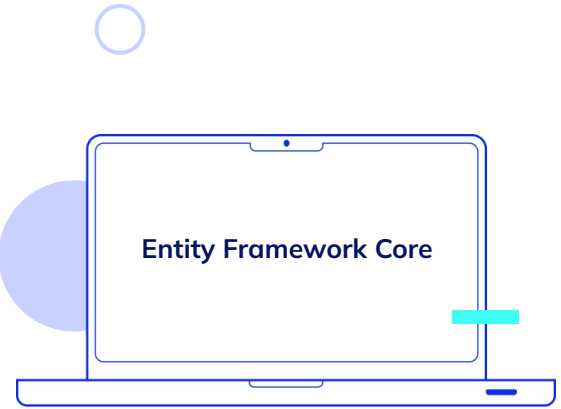
Módulo 8

Entity Framework Core

Introducción a Entity Framework Core

EF Core es un marco de **mapeo de objetos relacional (ORM)** que se puede usar al desarrollar aplicaciones **ASP.NET**. Permite trabajar con bases de datos y usar los datos en las aplicaciones **ASP.NET Core**.

Además, permite que los desarrolladores se orienten a los datos sin necesidad de concentrarse en modelar las entidades. **EF Core es una versión ligera de Entity Framework**, también extensible y multiplataforma.



Mapeo de objetos relacional (ORM)

ORM* es una técnica de programación que simplifica la interacción de la aplicación con los datos mediante un *descriptor de metadatos* para conectar el código de objeto a una base de datos relacional**. ORM proporciona una abstracción que mapea objetos de aplicación a registros de bases de datos. Hay varios marcos de ORM. Puedes elegir el que más te convenga. Uno de los más populares es ***Entity Framework. Entity Framework fue creado por Microsoft y **mapea las tablas y columnas que se hallan en una base de datos a los objetos y propiedades que se utilizan en .NET**.

Además de Entity Framework, también existen otros marcos ORM como ***Hibernate*** y ***Django***. Sin embargo, Entity Framework fue creado para .NET Framework por Microsoft como **la mejor combinación para aplicaciones .NET**. **EF Core** es una versión que puede ejecutarse en diferentes sistemas operativos y trabajar con diferentes bases de datos.

Descripción general de Entity Framework

Entity Framework proporciona una solución integral para interactuar con los datos almacenados en una base de datos.

Las clases se llaman *entidades*. Entity Framework mantiene **seguimiento de los cambios que realiza en las entidades** para habilitar la actualización de la base de datos con datos que existen en la memoria.

Entity Framework introduce una **capa de abstracción entre el esquema de la base de datos y el código de la aplicación**. Esto la hace más flexible. Con Entity Framework, se puede **acceder a objetos utilizando código fuertemente tipado** y consultarlos utilizando **LINQ**.

Enfoques de Entity Framework

Entity Framework proporciona tres enfoques generales para crear su **capa de acceso a datos** (**DAL, Data Access Layer**):

- **Base de datos primero (*DatabaseFirst*).**
- **Modelo primero (*ModelFirst*).**
- **Codifique primero (*CodeFirst*).**

En cada uno de estos enfoques, Entity Framework puede crear una nueva base de datos, de acuerdo con los datos a modelar, o utilizar una base de datos existente.

ModelFirst y DatabaseFirst

En estos enfoques, **el modelo de datos se genera mediante un diseñador en Visual Studio, y se almacena en un archivo *.edmx***. A partir del modelo se generan las ***clases de entidad*** y las ***relaciones***.

Si aún no tienes una base de datos, después de diseñar tu modelo, puedes generar scripts del modelo utilizando el diseñador de Visual Studio. Luego, puedes ejecutarlos en una nueva base de datos para crear las tablas. Por otro lado, si tu base de datos existía antes de crear el modelo, puedes utilizar el diseñador de Visual Studio para aplicar ingeniería inversa al modelo de las tablas de la base de datos.

Código primero (CodeFirst)

En este enfoque, no utilizarás un archivo *.edmx* para diseñar el modelo y no confiarás en el diseñador de Visual Studio.

El modelo de dominio es simplemente un conjunto de clases con propiedades que tú proporcionas. Entity Framework escanea tus clases de dominio y sus propiedades e intenta asignarlas a la base de datos.

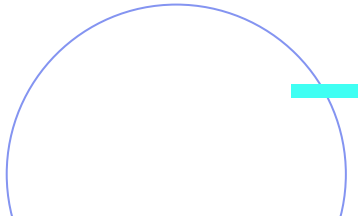
Puedes utilizar el enfoque ***CodeFirst*** tanto con nuevas bases de datos como con las existentes. Si no tienes una base de datos, el comportamiento default del enfoque CodeFirst es crear la base de datos la primera vez que ejecutes tu aplicación.

Si tu base de datos ya existe, Entity Framework se conectará a ella y usará las asignaciones definidas entre las clases de modelo del proyecto y las tablas de la base de datos existente.

Versiones de Entity Framework

Existen diferentes versiones de Entity Framework y es importante estar familiarizado con ellas:

- **Entity Framework6 (EF6)**.
- **Entity Framework Core (EF Core)**.



Entity Framework6 (EF6)

Entity Framework6 (EF6) se lanzó por primera vez en 2008. Con *EF6* se pueden controlar los datos almacenados en la base de datos y utilizarlos dentro de una aplicación. Es compatible con los enfoques ***ModelFirst***, ***DatabaseFirst*** y ***CodeFirst***, que permiten **elegir el enfoque de diseño** para la aplicación. Sin embargo, EF6 **solo es compatible con el sistema operativo Windows**.

Entity Framework Core (EF Core)

Cuando se decidió que .NET Framework, fuera **multiplataforma**, compatible con varios sistemas operativos y no solo Windows, fue necesario volver a implementar Entity Framework. Como resultado, se creó EF Core. EF Core es la versión .NET Core de Entity Framework. Es **extensible y de código abierto**, por lo que puedes encontrar soluciones o características específicas además de crear las tuyas.

Es importante tener en cuenta que EF Core **solo funciona con el enfoque *CodeFirst***. Sin embargo, es un marco moderno, avanzado y en alza que puede ser utilizado ampliamente por una gran variedad de plataformas.

En esta unidad, aprenderás a usar **EF Core** en aplicaciones **ASP.NET Core**.

Proveedores de bases de datos

Proveedores de bases de datos

EF Core es una capa entre tu código y una base de datos, que se utiliza para conectarlos. **Un *proveedor de base de datos* es una biblioteca que EF Core usa para conectarse a una base de datos**. Los proveedores de bases de datos se distribuyen como ***Paquetes NuGet***. Si deseas utilizar un proveedor de base de datos en tu aplicación, puedes instalar el correspondiente paquete *NuGet*.

El proveedor de Microsoft SQL

El proveedor de base de datos de SQL Server es muy utilizado en EF Core. SQL Server permite crear, administrar **bases de datos relacionales**.

El proveedor de SQLite

SQLite es una **base de datos de código abierto popular que es ampliamente utilizada** por los desarrolladores. Puedes usar el proveedor SQLite para conectar una base de datos SQLite a su aplicación mediante EF Core.

El proveedor de InMemory

Entity Framework Core presentó el proveedor InMemory. Es importante señalar que el proveedor InMemory **no se conecta a una base de datos relacional real y no imita una base de datos**, pero está diseñado para ser una **herramienta para probar la aplicación**. Por ejemplo, con el proveedor InMemory puedes guardar datosque no podrás guardar en una base de datos relacional real, como datos que pueden violar restricciones de integridad referenciales.

Otros proveedores

Hay otros proveedores de datos con los que es posible conectarse a distintos motores de base de datos como ***MySQL***, ***SQLite***, ***Maria DB*** y ***Db2***.Sin embargo, algunos proveedores son mantenidos por el proveedor de EF Core Project, Microsoft. Antes de usar un proveedor de base de datos, asegúrate de que cumpla con tus requisitos.

Agregar un proveedor de base de datos a su aplicación

Por lo general, un proveedor de bases de datos se distribuye como un paquete NuGet. Por lo tanto, es bastante sencillo agregarlo a una aplicación, se puede usarla **herramienta dotnet** o la ***Consola del Administrador de paquetes NuGet***.

Este es el comando **dotnet** que puedes usar para agregar un proveedor de base de datos:

```
dotnet add package provider_package_name
```

Este es el comando de la Consola del administrador de paquetes NuGet que puedes utilizar para agregar un proveedor de base de datos:

```
install-package provider_package_name
```

educación IT
CODE YOUR FUTURE

¡Sigamos trabajando!