



Entity Framework

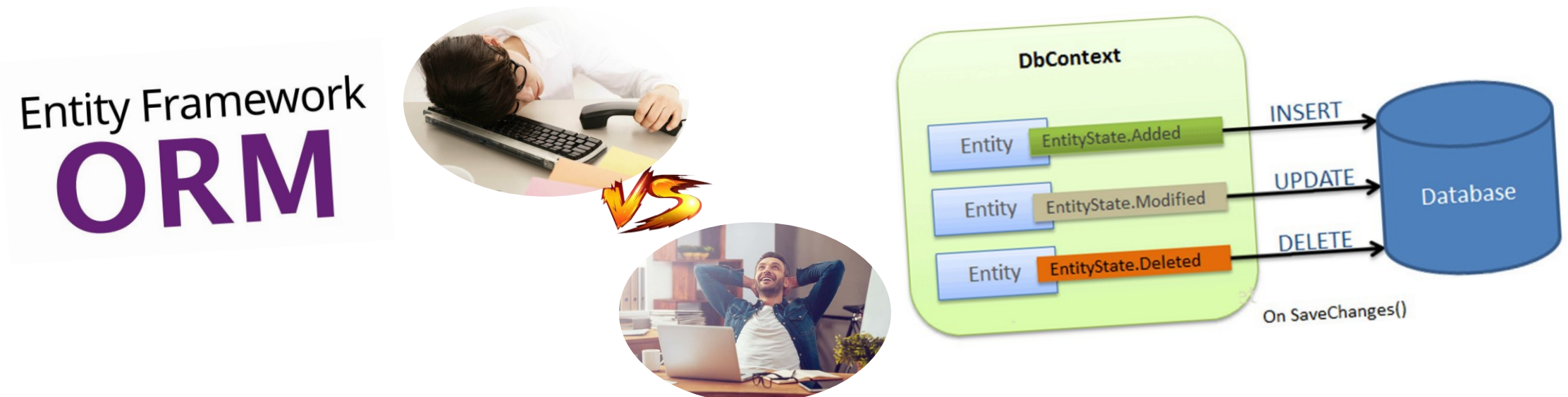


¿Que es Entity Framework?

Entity Framework

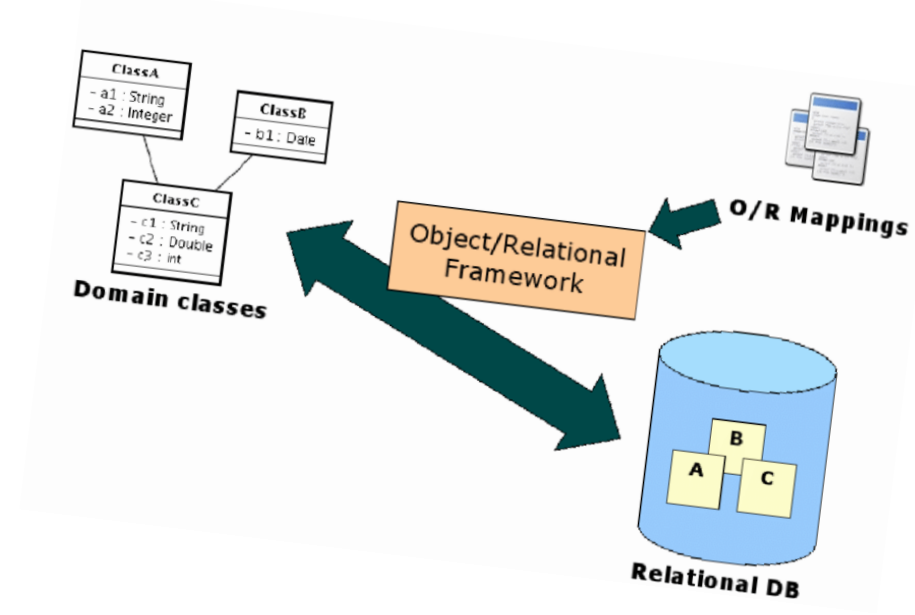
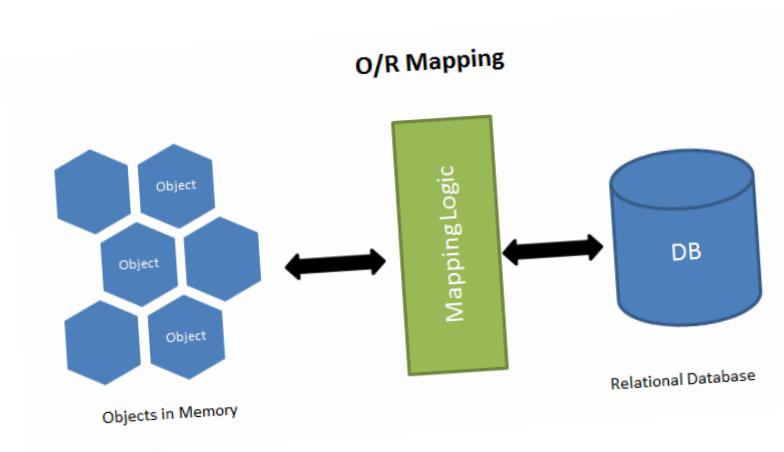
Escribir y administrar el código ADO.Net para el acceso a datos es un trabajo tedioso y monótono.

Microsoft ha proporcionado un marco O / RM llamado "**Entity Framework**" para automatizar las actividades relacionadas con la base de datos para su aplicación.



Entity Framework

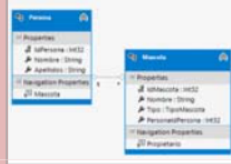


Entity framework es un marco Object / Relational Mapping (O / RM). Es una mejora de ADO.NET que brinda a los desarrolladores un mecanismo automatizado para acceder y almacenar los datos en la base de datos.



¿Qué es O / RM?

ORM es una herramienta para almacenar datos de objetos de dominio en la base de datos relacional como SQL Server, de manera automatizada, sin mucha programación.

Flujos de trabajo con EF

		<pre>public class VeterinarioContext: DbContext { public DbSet<Mascota> Mascotas { get; set; } public DbSet<Persona> Personas { get; set; } }</pre>
	Model First <ul style="list-style-type: none"> • Crear modelo en diseñador • BD creada desde modelo • Clases auto-generadas desde modelo 	Code First (nueva BD) <ul style="list-style-type: none"> • Define clases y mapeo en código • BD creada desde el modelo • Usar Migrations para cambiar BD
	Database First <ul style="list-style-type: none"> • Ingeniería inversa de BD a modelo • Clases auto-generadas desde modelo 	Code First (BD existente) <ul style="list-style-type: none"> • Usar herramientas de ingeniería inversa • Mapeo y clases definidos en código



O / RM incluye tres partes principales:

1. Objetos de clase de dominio
2. Objetos de base de datos relacionales
3. Información de asignación sobre cómo los objetos de dominio se asignan a objetos de base de datos relacionales (por ejemplo , tablas, vistas y procedimientos almacenados)

Ejemplo de Code First

```
[Table("Mascota", Schema = "dbo")]
public class Mascota
{
    [Key,
    DatabaseGenerated(DatabaseGeneratedOption.Identity)]
    public int IdMascota { get; set; }

    [MinLength(2), MaxLength(50), Required]
    public string Nombre { get; set; }

    [Required]
    public TipoMascota Tipo { get; set; }

    [Required]
    public virtual Persona Propietario
    {
        get; set; }
}

public class Persona
{
    public int PersonaId { get; set; }

    public string Nombre { get; set; }

    public string Apellidos { get; set; }

    public virtual List<Mascota> Mascotas { get; set; }
}
```

Atributos

Convencion

Clave

Relación

Ejemplito de código de EF

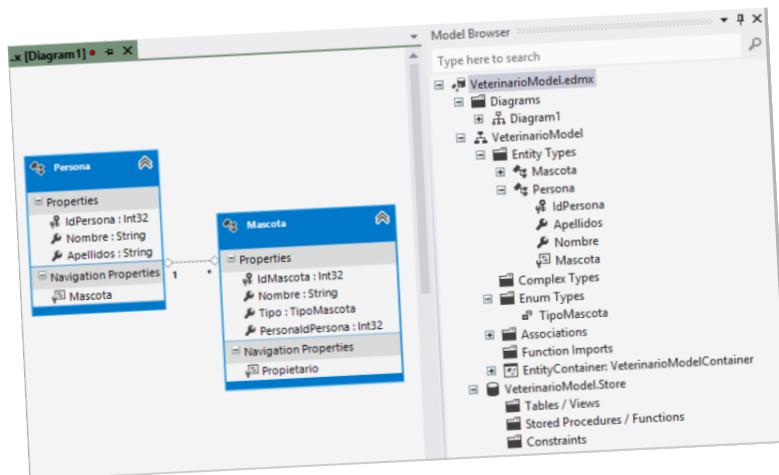
```
Persona pepe = new Persona {
    Nombre = "Pepe",
    Apellidos = "Pérez",
    Mascotas = new List<Mascota>()
};
Mascota chusky = new Mascota {
    Nombre = "Chusky",
    Tipo = TipoMascota.Perro
};
Mascota michu = new Mascota {
    Nombre = "Michu",
    Tipo = TipoMascota.Gato
};
pepe.Mascotas.Add(chusky);
pepe.Mascotas.Add(michu);

using (DefaultContext ctx
    = new DefaultContext())
{
    ctx.Personas.Add(pepe);
    ctx.SaveChanges();
}

Persona personaDeBd;
using (DefaultContext ctx = new DefaultContext())
{
    personaDeBd = ctx.Personas
        .Include(p => p.Mascotas).First();
    Console.WriteLine(personaDeBd.Nombre + " "
        + personaDeBd.Apellidos + " tiene las mascotas:");
    foreach (Mascota m in personaDeBd.Mascotas)
    {
        Console.WriteLine(m.Tipo + ": "
            + m.Nombre + " (de " + m.Propietario.Nombre + ")");
    }
}
```

Leer

Guardar



Trabajar con Entity Framework

Descargar paquete NuGet de EF
 Crear modelo (EDM o Code First)
 Instanciar el contexto (derivado del modelo)
 Se trabaja con LINQ a Entidades
 Llamar a SaveChanges

Resultado: no hay que escribir sentencias SQL

Referencias

- [entity-framework-es.pdf \(riptutorial.com\)](#)
- [Entity Framework 6 Para qué sirve? - PDF Descargar libre \(docplayer.es\)](#)