

Para saber mais: permitindo o uso de LojaContext com outros providers

Suponha que sua aplicação deverá suportar o MySQL. Atualmente o código de LojaContext está acoplado diretamente ao SQL Server. Como podemos tornar LojaContext independente do banco de dados empregado na aplicação?

Usando o princípio da Injeção de Dependência! Criaremos um construtor que receberá um objeto do tipo `DbContextOptions<LojaContext>` com as opções necessárias para usar o MySQL. Ele chamará o construtor da classe base `DbContext` através da palavra reservada `base`, passando as opções como argumento. Para que o código continue compilando, manteremos o construtor sem argumentos.

Quando tanto o construtor como o método `OnConfiguring` são usados, o último sobrescreve as opções no contexto. Por isso precisamos colocar uma condição no método `OnConfiguring` perguntando se as opções já foram configuradas, o que é obtido através da verificação da propriedade booleana `IsConfigured`.

O código final fica assim:

```
using Microsoft.EntityFrameworkCore;

namespace Alura.Loja.Testes.ConsoleApp
{
    public class LojaContext : DbContext
    {
        public DbSet<Produto> Produtos { get; set; }

        public LojaContext()
        { }

        public LojaContext(DbContextOptions<LojaContext> options): base(options)
        { }

        protected override void OnConfiguring(DbContextOptionsBuilder optionsBuilder)
        {
            if (!optionsBuilder.IsConfigured)
            {
                optionsBuilder
                    .UseSqlServer("Server=(localdb)\\mssqllocaldb;Database=LojaDB;Trusted_Connection=...");
            }
        }
    }
}
```

Para maiores detalhes e outras informações, leia a [página \(https://docs.microsoft.com/en-us/ef/core/miscellaneous/configuring-dbcontext\)](https://docs.microsoft.com/en-us/ef/core/miscellaneous/configuring-dbcontext) (em inglês) sobre configuração do `DbContext` no site da Microsoft.