

O estado Added

Transcrição

Aprendemos o que acontece com um dado quando é alterado, o que veremos agora é como o *ChangeTracker* age ao adicionar um produto.

Comentaremos o trecho de código onde alteramos o último produto. Em seguida adicionaremos o um novo produto, e para colocarmos sobre o controle do Entity chamaremos o método `contexto.Produtos.Add()`;

```
static void Main(string[] args)
{
    using(var contexto = new LojaContext())
    {
        var serviceProvider = contexto.GetInfrastructure<IServiceProvider>();
        var loggerFactory = serviceProvider.GetService<ILoggerFactory>();
        loggerFactory.AddProvider(SqlLoggerProvider.Create());

        var produtos = contexto.Produtos.ToList();
        foreach (var p in produtos)
        {
            Console.WriteLine(p);
        }

        Console.WriteLine("=====");
        foreach (var e in contexto.ChangeTracker.Entries())
        {
            Console.WriteLine(e);
        }

        //var p1 = produtos.Last();
        //p1.Nome = "007 - O Espiao Que Me Amava";

        var novoProduto = new Produto()
        {
            Nome = "Desinfetante",
            Categoria = "Limpeza",
            Preco = 2.99;
        };
        contexto.Produtos.Add(novoProduto);

        Console.WriteLine("=====");
        foreach (var e in contexto.ChangeTracker.Entries())
        {
            Console.WriteLine(e);
        }

        contexto.SaveChanges();

        //Console.WriteLine("=====");
        //produtos = contexto.Produtos.ToList();
        //foreach (var p in produtos)
        //{
            //    Console.WriteLine(p);
        //}
```

```

    // .....
    //}
}
}

```

Para evitarmos a repetição de código, vamos extrair o laço onde mostramos o estado das entidade no *ChangeTracker*.

```

private static void ExibeEntries(IEnumerable<EntityEntry> entries)
{
    Console.WriteLine("=====");
    foreach (var e in entries)
    {
        Console.WriteLine(e);
    }
}

```

Além do estado, mostraremos também as informações do produto.

```

private static void ExibeEntries(IEnumerable<EntityEntry> entries)
{
    Console.WriteLine("=====");
    foreach (var e in entries)
    {
        Console.WriteLine(e.Entity.ToString() + " - " + e.State);
    }
}

```

Na classe `Produto` dentro do `ToString()` , colocaremos todas as informações do produto.

```

namespace Alura.Loja.Testes.ConsoleApp
{
    internal class Produto
    {
        // ...

        public override string ToString()
        {
            return $"Produto: {this.Id}, {this.Nome}, {this.Categoria}, {this.Preco}";
        }
    }
}

```

Vamos substituir os laços de repetição por uma chamada do método `ExibeEntries()` . A classe `Program` ficará da seguinte maneira:

```

namespace Alura.Loja.Testes.ConsoleApp
{
    static void Main(string[] args)
    {
        using(var contexto = new LojaContext())
        {
            var serviceProvider = contexto.GetInfrasctructure<IServiceProvider>();

```

```

var loggerFactory = serviceProvider.GetService<ILoggerFactory>();
loggerFactory.AddProvider(SqlLoggerProvider.Create());

var produtos = contexto.Produtos.ToList();

ExibeEntries(contexto.ChangeTracker.Entries());

var novoProduto = new Produto
{
    Nome = "Desinfetante",
    Categoria = "Limpeza",
    Preco = 2.99;
};
contexto.Produtos.Add(novoProduto);

ExibeEntries(contexto.ChangeTracker.Entries());

contexto.SaveChanges();

ExibeEntries(contexto.ChangeTracker.Entries());
}
}

private static void ExibeEntries(IEnumerable<EntityEntry> entries)
{
    Console.WriteLine("=====");
    foreach (var e in contexto.ChangeTracker.Entries())
    {
        Console.WriteLine(e.Entity.ToString() + " - " + e.State);
    }
}
}

```

Após executar a aplicação, veremos os resultados. O primeiro resultado é o comando SQL `select` . Em seguida foi mostrado todos os dados cadastrados no banco, com o estado valor *Unchanged*.

O produto "Desinfetante" que adicionamos, está com o menor valor inteiro no **Id** e com o estado **Added**. Após ser salvo com o `SaveChanges()` , o resultado é o comando SQL `insert` :

```

INSERT INTO [Produtos] ([Categoria], [Nome], [Preco])
VALUES (@p0, @p1, @p2);

```

Novamente um `select` será gerado trazendo todos os dados cadastrados no banco. Agora o produto "Desinfetante" está com o **Id** gerado pelo banco e o estado *Unchanged*.