$\underline{\ } ... \ \ | Programaci\'onIes 2023 \ | LaPaulita - \ copia \ | LaPaulita . Sales \ | Value Object \ | Order Detail.cs \ | Copia \$

```
1 namespace LaPaulita.Sales.BusinessRules.ValueObject
2 {
3
       public class OrderDetail : IEquatable<OrderDetail>
 4
 5
           #region "Campos"
 6
           readonly int _productId;
 7
           readonly decimal _productPrice;
 8
           readonly short _productQuantity;
9
           #endregion
10
11
           #region "Propiedades"
12
           //public int ProductId { get { return _productId; } }
13
           public int ProductId => _productId; //expresión Lambda
14
           public decimal ProductPrice => _productPrice;
15
           public short ProductQuantity => _productQuantity;
           #endregion
16
17
           #region "Cosntructor"
18
19
           public OrderDetail(int productId, decimal productPrice, short productQuantity)
20
21
               _productId = productId;
22
               _productPrice = productPrice;
23
               _productQuantity = productQuantity;
24
25
           #endregion
26
27
           #region"Equals and GetHashCode"
28
           public override bool Equals(object? obj)
29
           {
30
               return Equals(obj as OrderDetail);
31
           }
32
           public bool Equals(OrderDetail? other)
33
34
35
               //return other is not null &&
                        ProductId == other.ProductId &&
36
               //
                        ProductPrice == other.ProductPrice &&
37
               //
38
               //
                        ProductQuantity == other.ProductQuantity;
39
40
               //**************
               // Otra forma usando GetHashCode.
41
42
               //**************
43
               if (other != null) { return GetHashCode() == other.GetHashCode(); }
44
               else { return false; }
           }
45
46
47
           public override int GetHashCode()
48
               return HashCode.Combine(ProductId, ProductPrice, ProductQuantity);
49
50
51
           #endregion
52
       }
53 }
54
...rogramaciónIes2023\LaPaulita - copia\LaPaulita.Sales\ValueObject\OrdersDetails.cs
                                                                                                                      1
1 namespace LaPaulita.Sales.BusinessRules.ValueObject
2 {
3
      public record class OrdersDetails(int ProductId, decimal ProductPrice, short ProductQuantity);
4 }
5
```

 $\underline{\dots} ce \ \ Programaci\'on I es 2023 \ \ La Paulita - copia \ \ La Paulita . Sales \ \ \ \ Create Order. cs$

```
1 using LaPaulita.Sales.BusinessRules.DTOs;
2 using LaPaulita.Sales.BusinessRules.ValueObject;
4 namespace LaPaulita.Sales.BusinessRules.Agregates
5 {
6
       public class CreateOrder : OrderHeader
7
8
           //************************
9
           // Esto si lo hago con una clase que es preparada por nosotros para ser
10
11
           //**************************
12
13
           // Campo
14
15
           readonly List<OrderDetail> records = new List<OrderDetail>();
           // Propiedad
16
           public IReadOnlyList<OrderDetail> Details => records;
17
18
19
           public void AddDetail(OrderDetail record)
20
               var ExistingOrderDetail = records.FirstOrDefault(r => r.ProductId == record.ProductId);
21
               if (ExistingOrderDetail != default)
22
23
24
                   records.Add(new OrderDetail(record.ProductId, record.ProductPrice,
25
                       (short)(record.ProductQuantity + ExistingOrderDetail.ProductQuantity)));
26
                   records.Remove(ExistingOrderDetail);
27
               }
28
               else
29
               {
30
                   records.Add(record);
31
               }
           }
32
33
           public void AddRecord(int productId, decimal productPrice, short productQuantity)
34
35
               AddDetail(new OrderDetail(productId, productPrice, productQuantity));
36
37
           }
38
39
           public static CreateOrder From(OrderHeaderDto orderHeaderDto)
40
               // Aqui realizamos el mapeo de las propiedades del DTO con la Entidad.
41
42
               CreateOrder createOrder = new CreateOrder
43
44
                   ClientId = orderHeaderDto.ClientId,
45
                   ShippingAddress = orderHeaderDto.ShippingAddress,
46
                   ShippingCity = orderHeaderDto.ShippingCity,
47
                   ShippingCountry = orderHeaderDto.ShippingCountry,
48
                   ShippingZip = orderHeaderDto.ShippingZip
               };
49
50
               foreach (var item in orderHeaderDto.OrderDetails)
51
52
               {
                   createOrder.AddRecord(item.ProductId, item.ProductPrice, item.ProductQuantty);
53
54
55
               return createOrder;
           }
56
       }
57
58 }
59
```

...\ProgramaciónIes2023\LaPaulita - copia\LaPaulita.Sales\Agregates\CreatesOrders.cs

```
1 using LaPaulita.Sales.BusinessRules.DTOs;
2 using LaPaulita.Sales.BusinessRules.ValueObject;
4 namespace LaPaulita.Sales.BusinessRules.Agregates
5 {
 6
       public class CreatesOrders : OrderHeader
7
8
           //************************
           // Esto si lo hago con una clase tipo record inmutable.
9
10
           //**************************
11
12
           readonly List<OrdersDetails> records = new List<OrdersDetails>();
           public IReadOnlyList<OrdersDetails> Details => records;
13
14
           /// <summary>
15
           /// Método que agrega un nuevo record a <c>OrdersDetails</c> de una orden de compra
16
17
           /// <c>OrderHeader</c>.
18
           /// Si el nuevo record a agregar tiene un <c>ProductId</c> que ya existe en
19
           /// <c>OrdersDetails</c>, solo se modifica la cantidad del producto adicinando al
20
           /// valor existente la nueva cantidad.
21
           /// </summary>
22
           /// <param name="record">Objeto que contiene los datos del nuevo registro a agregar.</param>
           public void AddRecord(OrdersDetails record)
23
24
               var ExistingOrderDetail = records.FirstOrDefault(r => r.ProductId == record.ProductId);
25
26
               if (ExistingOrderDetail != default)
27
28
                   // Con with creamos una copia del objetoque esta a su izquierda pero con
                   // las propiedades con valores modificado (entre la llaves)
29
30
                   // Es decir agrega a la lista records una copia del objeto ExistingOrderDetail
                   // pero modificando el valor de la propiedad ProductQuantity, asignandole como valor
31
32
                   // lo que tiene ExistingOrderDetail.ProductQuantity sumado a lo que tiene
33
                   // record.ProductQuantity
34
35
                   records.Add(ExistingOrderDetail with
36
                   {
37
                       ProductQuantity = (short)(ExistingOrderDetail.ProductQuantity + record.ProductQuantity)
38
                   });
                   records.Remove(ExistingOrderDetail);
39
               }
40
               else
41
42
               {
                   records.Add(record);
43
44
               }
45
           }
46
47
           public void AddRecord(int _productId, decimal productPrice, short productQuantity)
48
           {
49
               AddRecord(new OrdersDetails(_productId, productPrice, productQuantity));
50
           }
51
           public static CreateOrder From(OrderHeaderDto orderHeaderDto)
52
53
54
               // Aqui realizamos el mapeo de las propiedades del DTO con la Entidad.
55
               CreateOrder createOrder = new CreateOrder
               {
56
57
                   ClientId = orderHeaderDto.ClientId,
                   ShippingAddress = orderHeaderDto.ShippingAddress,
58
59
                   ShippingCity = orderHeaderDto.ShippingCity,
                   ShippingCountry = orderHeaderDto.ShippingCountry,
60
                   ShippingZip = orderHeaderDto.ShippingZip
61
62
               };
63
               foreach (var item in orderHeaderDto.OrderDetails)
65
                   createOrder.AddRecord(item.ProductId, item.ProductPrice, item.ProductQuantty);
66
67
               }
68
               return createOrder;
           }
69
70
       }
71 }
72
```

12

```
...urce\ProgramaciónIes2023\LaPaulita - copia\LaPaulita.Sales\DTOs\OrderHeaderDto.cs
 1 namespace LaPaulita.Sales.BusinessRules.DTOs
 2 {
 3
        public class OrderHeaderDto
 4
 5
            public int ClientId { get; set; }
 6
            public string ShippingAddress { get; set; }
 7
            public int ShippingCity { get; set; }
 8
            public int ShippingCountry { get; set; }
 9
            public string ShippingZip { get; set; }
10
11
            public List<OrderDetailDto> OrderDetails { get; set; }
12
        }
13 }
...urce\ProgramaciónIes2023\LaPaulita - copia\LaPaulita.Sales\DTOs\OrderDetailDto.cs
1 namespace LaPaulita.Sales.BusinessRules.DTOs
2 {
3
      public class OrderDetailDto
4
5
          public int ProductId { get; set; }
          public decimal ProductPrice { get; set; }
6
          public short ProductQuantty { get; set; }
8
9 }
...Ies2023\LaPaulita - copia\LaPaulita.Sales\DTOs\ValidatorDTO\ValidationErrorDto.cs
                                                                                                                             1
1 namespace LaPaulita.Sales.BusinessRules.DTOs.ValidatorDTO
2 {
3
       public class ValidationErrorDto
4
5
           public string PropertyName { get; set; }
           public string ErrorMessage { get; set; }
6
7
       }
8 }
9
...gramaciónIes2023\LaPaulita - copia\LaPaulita.Sales\Wrappers\WrappersSalesOrder.cs
1 namespace LaPaulita.Sales.BusinessRules.Wrappers
2 {
3
       public class WrappersSalesOrder
4
           public int OrderId { get; set; }
5
6
           public List<ValidationErrorDto> Errors { get; set; }
7
       }
8 }
9
\underline{\dots} La Paulita - copia \\ La Paulita. Sales \\ \\ Interface \\ \\ Controllers \\ \\ ICreate \\ \\ Order \\ Controller. \\ \\ cs
1 using LaPaulita.Sales.BusinessRules.DTOs;
2 using LaPaulita.Sales.BusinessRules.Wrappers;
4 namespace LaPaulita.Sales.BusinessRules.Interface.Controllers
5
6
       public interface ICreateOrderController
7
8
           Task<WrappersSalesOrder> CreateOrder(OrderHeaderDto order);
9
       }
10 }
11
...2023\LaPaulita - copia\LaPaulita.Sales\Interface\Getways\ICreateOrderInputPort.cs
   using LaPaulita.Sales.BusinessRules.DTOs;
   namespace LaPaulita.Sales.BusinessRules.Interface.Getways
4 {
5
       public interface ICreateOrderInputPort
6
7
            Task Handle(OrderHeaderDto createOrderDto);
8
   }
9
10
...023\LaPaulita - copia\LaPaulita.Sales\Interface\Getways\ICreateOrderOutputPort.cs
using LaPaulita.Sales.BusinessRules.Wrappers;
3 namespace LaPaulita.Sales.BusinessRules.Interface.Getways
4
       public interface ICreateOrderOutputPort
5
6
       {
 7
           //ValueTask Handle(int orderId);
           ValueTask Handle(WrappersSalesOrder order);
8
           ValueTask ValidationFaild(List<ValidationErrorDto> validationError);
9
10
       }
11 }
```

```
\underline{\dots 3} \\ La Paulita - copia \\ La Paulita . Sales \\ Interface \\ \underline{Presenters} \\ ICreate \\ 0 \\ resenter. \\ cs
 1 using LaPaulita.Sales.BusinessRules.Wrappers;
 3 namespace LaPaulita.Sales.BusinessRules.Interface.Presenters
 4 {
        public interface ICreateOrderPresenter : ICreateOrderOutputPort
 5
 6
 7
            //int OrderId { get; }
 8
            //List<ValidationErrorDto> ErrorsList { get; }
 9
            WrappersSalesOrder Order { get; }
10
11 }
12
\underline{\dots} Paulita - copia \\ La Paulita. Sales \\ Interface \\ Repositories \\ ISales \\ Command \\ Repository. \\ cs
1 using LaPaulita.Entity.Interfaces;
 2 using LaPaulita.Sales.BusinessRules.Agregates;
 3
 4 namespace LaPaulita.Sales.BusinessRules.Interface.Repositories
 5
 6
        public interface ISalesCommandRepository : IUnitOfWork
 7
 8
            Task CreateOrder(CreateOrder order);
 9
        }
10 }
11
...ProgramaciónIes2023\LaPaulita - copia\LaPaulita.Sales\Interface\ISpecification.cs
 1 using LaPaulita.Sales.BusinessRules.DTOs.ValidatorDTO;
 3 namespace LaPaulita.Sales.BusinessRules.Interface
 4 {
 5
        public interface ISpecification<T> where T : class
 6
 7
            List<ValidationErrorDto> IsValid();
        }
 8
 9 }
10
```