**Batch-Custom:**

@page "/Demo/Batch/Custom/Details"

@page "/Demo/Batch/Custom/Details/{MasterId:int}"

@inherits MasterDetailPage<Order, OrderDeatil>

<div class="col-md-4 row pe-3">

<**CaspianForm** @ref="Form" **Model**="UpsertData">

<div class="row c-controls ps-3 pe-3">

<div class="col-md-4">

<label>Order date</label>

</div>

<div class="col-md-8">

<**DatePicker** @bind-Value="UpsertData.Date" />

</div>

<div class="col-md-4">

<label>Order type</label>

</div>

<div class="col-md-8">

<**DropdownList** @bind-Value="UpsertData.OrderType" />

</div>

<div class="col-md-4">

<label>Customer</label>

</div>

<div class="col-md-8">

<**ComboBox** **TEntity**="Customer" **TValue**="int?" @bind-Value="UpsertData.CustomerId"

**TextExpression**="@(t => t.CustomerType == CustomerType.Real ? (t.FName + " " + t.LName) : t.CompanyName)" />

</div>

<div class="col-md-12">

<**FormCommandButton** />

</div>

</div>

</**CaspianForm**>

<div class="pt-3 text-center pe-3">

<div class="pt-3">

<button @onclick="InsertedProducts" class="c-btn c-primary">Inserted products</button>

</div>

<div class="pt-3">

<button @attributes="btnAttr" @onclick="UpdatedProducts" class="c-btn c-primary">Updated products</button>

</div>

<div class="pt-3">

<button @attributes="btnAttr" @onclick="DeletedProducts" class="c-btn c-primary">Deleted products</button>

</div>

</div>

</div>

<div class="col-md-8 ps-3">

@if (products != null)

{

<div class="ps-4">

@foreach (var product in products)

{

<span @onclick="async () => await AddUpsertProduct(product)" class="product">@product.Title</span>

}

</div>

}

<div class="pt-4">

<**DataGrid** @ref="Grid" **TEntity**="OrderDeatil" **Batch** **ConditionExpr**="t => t.OrderId == MasterId" **TableHeight**="250" **HideInsertIcon**>

<**Columns**>

<**GridColumn** **Field**="t => t.Product.ProductCategory.Title" **Title**="Product category" />

<**GridColumn** **Field**="t => t.Product.Title" **Title**="Product" />

<**GridColumn** **Field**="t => t.Price.Seprate3Digit()" **Title**="Price" />

<**GridColumn** **Field**="t => t.Quantity" **Title**="Quantity">

<**Template**>

<**IconButton** **IconType**="IconType.Plus" @onclick="async () => await IncQuantity(context.Data)" />

<span>@context.Data.Quantity</span>

<**IconButton** **IconType**="IconType.Minus" @onclick="async () => await DecQuantity(context.Data)" />

</**Template**>

</**GridColumn**>

</**Columns**>

</**DataGrid**>

</div>

</div>

@code

{

IList<Product> products;

IDictionary<string, object> btnAttr;

protected override Task OnMasterEntityCreatedAsync()

{

UpsertData.Date = DateTime.Now.Date;

return base.OnMasterEntityCreatedAsync();

}

protected async override Task OnInitializedAsync()

{

var scope = CreateScope();

products = await new ProductService(scope.ServiceProvider).GetAll().ToListAsync();

foreach (var product in products)

product.Image = null;

btnAttr = new Dictionary<string, object>();

if (MasterId == 0)

btnAttr.Add("disabled", "disabled");

await base.OnInitializedAsync();

}

async Task UpdatedProducts()

{

var products = Grid.GetUpdatedEntities();

if (products.Count == 0)

await Alert("No products have been updated");

else

{

var str = $"Updated products count: {products.Count}\nUpdated products Id: {{ ";

var isFirst = true;

foreach (var product in products)

{

if (!isFirst)

str += ", ";

str += product.ProductId;

isFirst = false;

}

await Alert(str + " }");

}

}

async Task DeletedProducts()

{

var products = Grid.GetDeletedEntities();

if (products.Count == 0)

await Alert("No products have been deleted");

else

{

var str = $"Deleted products count: {products.Count}\n Deleted products Id {{ ";

var isFirst = true;

foreach (var product in products)

{

if (!isFirst)

str += ", ";

str += product.ProductId;

isFirst = false;

}

await Alert(str + " }");

}

}

async Task InsertedProducts()

{

var products = Grid.GetInsertedEntities();

if (products.Count == 0)

await Alert("No products have been added");

else

{

var str = $"Added Products count: {products.Count}\n Added products Id: {{";

var isFirst = true;

foreach (var product in products)

{

if (!isFirst)

str += ", ";

str += product.ProductId;

isFirst = false;

}

await Alert(str + " }");

}

}

async Task AddUpsertProduct(Product product)

{

var old = Grid.GetAllRecords().SingleOrDefault(t => t.ProductId == product.Id);

if (old == null)

{

await Grid.InsertAsync(new OrderDeatil()

{

Quantity = 1,

Price = product.Price,

ProductId = product.Id,

OrderId = MasterId

});

}

else

{

old.Quantity = old.Quantity + 1;

await Grid.UpdateAsync(old);

}

}

async Task IncQuantity(OrderDeatil detail)

{

detail.Quantity++;

await Grid.UpdateAsync(detail);

}

async Task DecQuantity(OrderDeatil detail)

{

detail.Quantity--;

if (detail.Quantity == 0)

await Grid.RemoveAsync(detail);

else

await Grid.UpdateAsync(detail);

}

}