**Custom template*:***

<**CTreeView** **TEntity**="TreeViewItem" **Source**="source">

<**Template**>

<**TreeNodeTemplate** **Level**="1">

<span class=" d-tree-icon d-tree-add">

<span class="fa fa-plus" @onclick="() => ShowEditForm(context, false)"></span>

</span>

</**TreeNodeTemplate**>

<**TreeNodeTemplate** **Level**="2">

<span class="d-tree-icon d-tree-edit" @onclick="() => ShowEditForm(context, true)">

<span class="fa fa-pencil"></span>

</span>

<span class="d-tree-icon d-tree-delete" @onclick="async () => await DeleteCity(context)">

<span class="fa fa-trash"></span>

</span>

</**TreeNodeTemplate**>

</**Template**>

</**CTreeView**>

<**Window** **Title**="City" @bind-Status="status" **OnOpen**="() => firstControl.Focus()">

<**Content**>

<**CaspianForm** **Model**="city" **OnValidSubmit**="async t => await UpsertCity()">

<div class="row ps-4 pe-4 c-controls">

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

<label>Title</label>

</div>

<div class="col-md-9">

<**StringTextBox** @ref="firstControl" @bind-Value="city.Title" />

</div>

<div class="col-md-3"></div>

<div class="text-center col-md-9">

<**WindowCommandButtons** />

</div>

</div>

</**CaspianForm**>

</**Content**>

</**Window**>

@code

{

IControl firstControl;

IList<TreeViewItem> source;

TreeViewItem selectedNode;

WindowStatus status;

City city;

protected override async Task OnInitializedAsync()

{

using var scope = CreateScope();

source = await new CountryService(scope.ServiceProvider).GetAll().Select(t => new TreeViewItem()

{

Collabsable = true,

Text = t.Title,

Value = t.Id.ToString(),

ShowTemplate = true,

Items = t.Cities.Select(u => new TreeViewItem()

{

Collabsable = false,

Text = u.Title,

Value = t.Id.ToString(),

ShowTemplate = true

}).ToList()

}).ToListAsync();

}

}

async Task ShowEditForm(TreeViewItem nodeView, bool isUpdate)

{

///window opening

city = new City();

city.ActiveType = ActiveType.Enable;

if (isUpdate)

{

///Edit mode

var cityId = Convert.ToInt32(nodeView.Value);

using var scope = CreateScope();

var old = await new CityService(scope.ServiceProvider).SingleAsync(cityId);

city.Id = cityId;

city.CountryId = old.CountryId;

city.Title = old.Title;

}

else

{

///Insert mode

city.CountryId = Convert.ToInt32(nodeView.Value);

}

///Selected node

selectedNode = nodeView;

status = WindowStatus.Open;

}

async Task UpsertCity()

{

using var scope = CreateScope();

var service = new CityService(scope.ServiceProvider);

TreeViewItem tempNode = null;

if (city.Id == 0)

{

await service.AddAsync(city);

///selected node is country and must reload cities of country after upsert.

tempNode = selectedNode;

tempNode.Expanded = true;

}

else

{

await service.UpdateAsync(city);

/// selected node is city and must reload cities of country (parent node) after upsert.

tempNode = selectedNode.Parent;

}

await service.SaveChangesAsync();

status = WindowStatus.Close;

tempNode.Items = await service.GetAll().Where(t => t.CountryId == city.CountryId)

.Select(t => new TreeViewItem()

{

Collabsable = false,

Expanded = false,

Text = t.Title,

Value = t.Id.ToString(),

ShowTemplate = true

}).ToListAsync();

}

async Task DeleteCity(TreeViewItem node)

{

var cityId = Convert.ToInt32(node.Value);

using var scope = CreateScope();

var service = new CityService(scope.ServiceProvider);

var city = await service.SingleAsync(cityId);

var result = await service.ValidateRemoveAsync(city);

if (result.IsValid)

{

await service.RemoveAsync(city);

await service.SaveChangesAsync();

///After removing the city, all the cities of the country (parent node) must be reloaded.

node.Parent.Items = await service.GetAll()

.Where(t => t.CountryId == city.CountryId).Select(t => new TreeViewItem()

{

Collabsable = false,

Expanded = false,

Text = t.Title,

Value = t.Id.ToString(),

ShowTemplate = true

}).ToListAsync();

}

else

ShowMessage(result.Errors.First().ErrorMessage);

}

}