Tree Node Template:

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

**OnExpanded**="async node => await OnExpanded(node)">

<**Template**>

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

<span class="t-grid-delete">

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

</span>

</**TreeNodeTemplate**>

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

<span style="padding:0 8px 0 0;" class="t-grid-edit" @onclick="() => ShowEditForm(context, true)">

<span class="t-icon t-edit fa fa-pencil"></span>

</span>

<span class="t-grid-delete" @onclick="async () => await DeleteCity(context)">

<span class="t-icon t-delete fa fa-trash"></span>

</span>

</**TreeNodeTemplate**>

</**Template**>

</**CTreeView**>

Search:

<**CTreeView** **TEntity**="TreeViewItem" **Source**="searchSource" />

@code

{

//----- Custome Template -----

IList<TreeViewItem> source;

TreeViewItem node;

WindowStatus status;

City city;

//----Search------------

IList<TreeViewItem> searchSource;

IControl firstControl;

string strSearch;

async Task<bool> UpsertCity()

{

using var scope = CreateScope();

var service = new CityService(scope.ServiceProvider);

TreeViewItem tempNode = null;

if (city.Id == 0)

{

await service.AddAsync(city);

///نود انتخابی استان هست و باید تمامی فرزندان نود(شهرها) بارگذاری شوند

tempNode = node;

}

else

{

await service.UpdateAsync(city);

///نود انتخابی شهر است و باید تمامی فرزندان نود پدر (فرزندان استان) بارگذاری شوند

tempNode = node.Parent;

}

await service.SaveChangesAsync();

status = WindowStatus.Close;

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

.Select(t => new TreeViewItem()

{

Collabsable = false,

Expanded = false,

Text = t.Title,

Value = t.Id.ToString(),

ShowTemplate = true

}).ToListAsync();

StateHasChanged();

return true;

}

async Task ShowEditForm(TreeViewItem nodeView, bool isUpdate)

{

///باز شدن ویندو

city = new City();

city.ActiveType = ActiveType.Enable;

if (isUpdate)

{

///در حالت ویرایش

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

using var scope = CreateScope();

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

city.Id = cityId;

city.ProvinceId = old.ProvinceId;

city.Title = old.Title;

}

else

{

///در حالت ثبت

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

}

///نود انتخاب شده

node = nodeView;

status = WindowStatus.Open;

}

protected override async Task OnInitializedAsync()

{

//----- Custome Template -----

//این قسمت انتخابی است و میتواند به دو صورت بارگذاری در ابتدا و یا بارگذاری در باز کردن نود اعمال شود

//برای اطلاعات بیشتر مراجعه شود به منوی قبلی

using var scope = CreateScope();

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

{

Collabsable = true,

Text = t.Title,

Value = t.Id.ToString(),

ShowTemplate = true

}).ToListAsync();

await SearchOnTree(null);

}

void OnCollapsed(TreeViewItem node)

{

///در صورت بارگذاری در ابتدا نیازی به این بخش نیست

node.Items = null;

node.Selected = false;

}

async Task OnExpanded(TreeViewItem node)

{

///در صورت بارگذاری در ابتدا نیازی به این بخش نیست

using var scope = CreateScope();

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

node.Items = await new CityService(scope.ServiceProvider).GetAll()

.Where(t => t.ProvinceId == provinceId).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();

///پس از خذف شهر باید تمامی شهرهای استان(گره پرنت) دوباره بارگذاری شوند

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

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

{

Collabsable = false,

Expanded = false,

Text = t.Title,

Value = t.Id.ToString(),

ShowTemplate = true

}).ToListAsync();

}

else

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

}

//-----Search Method ---

async Task SearchOnTree(string str)

{

if (str == "")

str = null;

using var scope = CreateScope();

var query = new ProvinceService(scope.ServiceProvider).GetAll();

if (str.HasValue())

query = query.Where(t => t.Cities.Any(u => u.Title.Contains(str)));

searchSource = await query.Select(t => new TreeViewItem()

{

Collabsable = true,

Expanded = true,

Text = t.Title,

Value = t.Id.ToString(),

Items = t.Cities.Where(u => str == null || u.Title.Contains(str)).Select(u => new TreeViewItem()

{

Collabsable = false,

Text = u.Title,

Value = u.Id.ToString()

}).ToList()

}).ToListAsync();

await base.OnInitializedAsync();

}

}