**Databinding & search:**

<**StringTextBox** @bind-Value="@organSearch" **OnChange**="async () => await tree.ReloadAsync()"

**BindingType**="BindingType.OnInput" placeholder="Search ..." />

<**TreeView** @ref="tree" **TEntity**="OrganUnit" style="height:500px" **TextFunc**="t => t.Title"

**FilterFunc**="t => (!organSearch.HasValue() || t.Title.StartsWith(organSearch)) && t.ActiveType == activeType" />

@code

{

ActiveType activeType = ActiveType.Enable;

string organSearch;

}

**Custom template:**

<div class="row align-items-center">

<div class="col-md-1 d-tree-icon text-center pb-1">

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

</div>

<div class="col-md-11 pb-1">

<**StringTextBox** @bind-Value="@organSearch" **OnChange**="async () => await customTemplateTree.ReloadAsync()" **BindingType**="BindingType.OnInput"

placeholder="Search ..." />

</div>

<div>

<**TreeView** @ref="customTemplateTree" **TEntity**="OrganUnit" style="height:500px" **TextFunc**="t => t.Title"

**FilterFunc**="t => (!organSearch.HasValue() || t.Title.StartsWith(organSearch)) && t.ActiveType == activeType">

<**Template**>

<**TreeNodeTemplate** **AllLevels**>

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

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

</span>

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

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

</span>

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

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

</span>

</**TreeNodeTemplate**>

</**Template**>

</**TreeView**>

</div>

</div>

@code

{

string organSearch;

WindowStatus status;

OrganUnit organUnit;

TreeView<OrganUnit> databindingTree;

TreeView<OrganUnit> customTemplateTree;

IControl firstControl;

async Task DeleteOrganUnit(TreeViewItem node)

{

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

using var scope = CreateScope();

var service = new OrganUnitService(scope.ServiceProvider);

var organUnit = await service.SingleAsync(organUnitId);

var result = await service.ValidateRemoveAsync(organUnit);

if (result.IsValid)

{

await service.RemoveAsync(organUnit);

await service.SaveChangesAsync();

customTemplateTree.RemoveFromTree(organUnit);

}

else

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

}

async Task ShowEditForm(TreeViewItem nodeView, bool isUpdate)

{

if (isUpdate)

{

///در حالت ویرایش

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

using var scope = CreateScope();

organUnit = await new OrganUnitService(scope.ServiceProvider).SingleAsync(organUnitId);

}

else

{

organUnit = new OrganUnit();

organUnit.ActiveType = ActiveType.Enable;

organUnit.ParentOrganId = nodeView == null ? null : Convert.ToInt32(nodeView.Value);

}

status = WindowStatus.Open;

}

async Task<bool> UpsertOrganUnit()

{

using var scope = CreateScope();

var service = new OrganUnitService(scope.ServiceProvider);

if (organUnit.Id == 0)

await service.AddAsync(organUnit);

else

await service.UpdateAsync(organUnit);

await service.SaveChangesAsync();

customTemplateTree.UpsertInTree(organUnit);

status = WindowStatus.Close;

StateHasChanged();

return true;

}

}