**Check foreign key for upsert:**

<div class="row c-controls">

<div class="ps-5 col-md-4">

<label>Country</label>

</div>

<div class="pe-5 col-md-8">

<**ComboBox** @bind-Value="upsertCountryId" **TValue**="int?" **TEntity**="SelectListItem" **OnChange**="() => upsertResult = null"

**Source**="countries" />

</div>

<div class="ps-5 col-md-4">

<label>Country id</label>

</div>

<div class="pe-5 col-md-8 text-center">

<label>@(upsertCountryId?.ToString() ?? "---")</label>

</div>

<div class="ps-5 col-md-4">

<label>Validation status</label>

</div>

<div class="pe-5 col-md-8 text-center">

<label>@(upsertResult == null ? "---" : upsertResult.IsValid == true ? "Is valid" : "Is invalid")</label>

</div>

<div class="ps-5 col-md-4">

<label>Error message</label>

</div>

<div class="pe-5 col-md-8 text-center">

<label>@(upsertResult?.IsValid == false ? upsertResult.Errors[0] : "---")</label>

</div>

<div class="pt-2 ps-5 pe-5 col-md-12 text-center">

<button class="c-btn c-primary" @onclick="async () => await CheckForeignKeyForUpsert()">Click to validate</button>

</div>

</div>

@code

{

int? upsertCountryId;

IList<SelectListItem> countries;

ValidationResult upsertResult;

protected override async Task OnInitializedAsync()

{

var scope = ServiceScopeFactory.CreateScope();

var service = new CityService(scope.ServiceProvider);

service.UserId = UserId;

countries = await new CountryService(scope.ServiceProvider).GetAll().Take(5)

.Select(t => new SelectListItem()

{

Text = t.Title,

Value = t.Id.ToString()

}).ToListAsync();

countries.Add(new SelectListItem("1256985", "Invalid Country"));

await base.OnInitializedAsync();

}

async Task CheckForeignKeyForUpsert()

{

var scope = CreateScope();

var service = new CityService(scope.ServiceProvider);

var city = new City()

{

CountryId = upsertCountryId ?? 0,

ActiveType = ActiveType.Enable,

Title = "For Test"

};

upsertResult = await service.ValidateAsync(city);

}

}

**Check foreign key for delete:**

<div class="row c-controls">

<div class="ps-5 col-md-4">

<label>Country</label>

</div>

<div class="pe-5 col-md-8">

<**ComboBox** @bind-Value="deleteCountryId" **TValue**="int?" **TEntity**="Country"

**TextExpression**="t => t.Title + '(' + t.Cities.Count() + ')'" **OnChange**="() => deleteResult = null" />

</div>

<div class="ps-5 col-md-4">

<label>Country id</label>

</div>

<div class="pe-5 col-md-8 text-center">

<label>@(deleteCountryId?.ToString() ?? "---")</label>

</div>

<div class="ps-5 col-md-4">

<label>Validation status</label>

</div>

<div class="pe-5 col-md-8 text-center">

<label>@(deleteResult == null ? "---" : deleteResult.IsValid == true ? "Is valid" : "Is invalid")</label>

</div>

<div class="ps-5 col-md-4">

<label>Error message</label>

</div>

<div class="pe-5 col-md-8 text-center">

<label>@(deleteResult?.IsValid == false ? deleteResult.Errors[0] : "---")</label>

</div>

<div class="pt-2 ps-5 pe-5 col-md-12 text-center">

<button class="c-btn c-primary" @onclick="async () => await CheckForeignKeyForDelete()">Click to validate</button>

</div>

</div>

@code

{

int? deleteCountryId;

ValidationResult deleteResult;

async Task CheckForeignKeyForDelete()

{

if (deleteCountryId == null)

ShowMessage("Please specify country to validate");

else

{

var scope = CreateScope();

var service = new CountryService(scope.ServiceProvider);

var country = await service.SingleAsync(deleteCountryId.Value);

deleteResult = await service.ValidateRemoveAsync(country);

}

}

}

**Check enume:**

<div class="row c-controls">

<div class="ps-5 col-md-4">

<label>Order Type</label>

</div>

<div class="pe-5 col-md-8">

<**DropdownList** @bind-Value="orderType" **OnChange**="() => enumResult = null"

**Source**='SelectListItem.CreateList("Salon", "Takeout", "Tel", "Internet", "Invalid order type")' />

</div>

<div class="ps-5 col-md-4">

<label>Enum value</label>

</div>

<div class="pe-5 col-md-8">

<label>@(Convert.ToInt32(orderType))</label>

</div>

<div class="ps-5 col-md-4">

<label>Validation status</label>

</div>

<div class="pe-5 col-md-8 text-center">

<label>@(enumResult == null ? "---" : enumResult.IsValid == true ? "Is valid" : "Is invalid")</label>

</div>

<div class="ps-5 col-md-4">

<label>Error message</label>

</div>

<div class="pe-5 col-md-8 text-center">

<label>@(enumResult?.IsValid == false ? enumResult.Errors[0] : "---")</label>

</div>

<div class="pt-2 ps-5 pe-5 col-md-12 text-center">

<button class="c-btn c-primary" @onclick="async () => await CheckEnum()">Click to validate</button>

</div>

</div>

@code

{

OrderType orderType = OrderType.Salon;

ValidationResult enumResult;

async Task CheckEnum()

{

var scope = CreateScope();

var service = new EnumOrderService(scope.ServiceProvider);

var order = new Order()

{

OrderType = orderType,

};

enumResult = await service.ValidateAsync(order);

}

}