Custom Model Validation In ASP.NET Core 3.1

Farhan Ahmed Updated date, Feb 17, 2020 20.6k 0 5

This article explain how to create custom model validation in ASP.net core. We have model validation in System.ComponentModel.DataAnnotations namespace. Validation attributes let you specify validation rules for model properties. We can create custom validation attributes, create a class that inherits from ValidationAttribute, and override the IsValid method. Here is the code.

The `IsValid` method accepts an object named *value*, which is the input to be validated. An overload also accepts a `ValidationContext` object, which provides additional information, such as the model instance created by model binding.

ValidationAttribute class

This class is the base class for all validation attributes in System.ComponentModel.DataAnnotations namespace.

Methods	Description
GetValidationResult(Object, ValidationContext)	This checks whether the specified value is valid with respect to the current validation attribute.

IsValid(Object)	Determines whether the specified value of the object is valid.
IsValid(Object, ValidationContext)	This validates the specified value with respect to the current validation attribute.
MemberwiseClone()	This creates a shallow copy of the current Object. It is Inherited from Object
ToString()	This returns a string that represents the current object. It is Inherited from Object
Validate(Object, String)	This validates the specified object.
Validate(Object, ValidationContext)	This validates the specified object.

Step 1

Start-up Visual Studio 2019. Now click on create new project and Choose ASP.NET Core Web Application and click on “Next”.

Custom Model Validation In ASP.NET Core 3.1

After clicking next, another wizard will open. Under the project name, give a meaningful name to your project and click on create.

Custom Model Validation In ASP.NET Core 3.1

That will open up another new wizard. Select ASP.Net Core 3.1 from the dropdown. If not, select default. Choose Web Application (Model-View-Controller) template and click on create which will create ASP.Net Core Application.

Custom Model Validation In ASP.NET Core 3.1

Step 2

Now right click on Models folder and “Add” class and name it Student.

```
using System.ComponentModel.DataAnnotations;

namespace MvcCoreCustomModelValidation_Demo.Models
{
    public class Student
    {
        [Key]
        public int Id { get; set; }

        [Required(ErrorMessage = "Please enter name")]
        public string Name { get; set; }

        [Required(ErrorMessage = "Please choose admission date.")]
        [Display(Name = "Admission Date")]
        [DataType(DataType.Date)]
        [CustomAdmissionDate(ErrorMessage = "Admission Date must be less than or equal to Today's D
        public DateTime AdmissionDate { get; set; }

        [Display(Name = "Date of Birth")]
        [DataType(DataType.Date)]
        [Min18Years]
        public DateTime DateofBirth { get; set; }
    }
}
```

Step 3

Right click on project and “Add” folder CustomValidation. Now “Add” two classes, CustomAdmissionDate and Min18Years, respectively. Inherit from ValidationAttribute class override method IsValid with bool property.

```
using System;
using System.ComponentModel.DataAnnotations;

```

```
    public class CustomAdmissionDate : ValidationAttribute
    {
        public override bool IsValid(object value)
        {
            DateTime dateTime = Convert.ToDateTime(value);
            return dateTime <= DateTime.Now;
        }
    }
}
```

```
using MvcCoreCustomModelValidation_Demo.Models;
using System;
using System.ComponentModel.DataAnnotations;

namespace MvcCoreCustomModelValidation_Demo.CustomValidation
{
    public class Min18Years : ValidationAttribute
    {
        protected override ValidationResult IsValid(object value, ValidationContext validationConte
        {
            var student = (Student)validationContext.ObjectInstance;

            if (student.DateofBirth == null)
                return new ValidationResult("Date of Birth is required.");

            var age = DateTime.Today.Year - student.DateofBirth.Year;

            return (age >= 18)
                ? ValidationResult.Success
                : new ValidationResult("Student should be at least 18 years old.");
        }
    }
}
```

Now open HomeController which is added when we created the new project. Write the following code for New IActionResult methods.

```
using Microsoft.AspNetCore.Mvc;
using MvcCoreCustomModelValidation_Demo.Models;

namespace MvcCoreCustomModelValidation_Demo.Controllers
{
    public class HomeController : Controller
    {
        public IActionResult Index()
        {
            return View();
        }

        public IActionResult New()
        {
            return View();
        }

        [HttpPost]
        [ValidateAntiForgeryToken]
        public IActionResult New(Student student)
        {
            if (ModelState.IsValid)
            {
                RedirectToAction("Index");
            }
            return View();
        }
    }
}
```

Step 5

```
@model MvcCoreCustomModelValidation_Demo.Models.Student

@{
    ViewData["Title"] = "New";
}

<div class="card">
    <div class="card-header">
        <h4 class="text-uppercase">Student Information</h4>
    </div>
    <div class="card-body">
        <form asp-action="New">
            <div class="form-group">
                <label asp-for="Name" class="label-control"></label>
                <input asp-for="Name" class="form-control" />
                <span asp-validation-for="Name" class="text-danger"></span>
            </div>
            <div class="row">
                <div class="col-md-6">
                    <div class="form-group">
                        <label asp-for="AdmissionDate" class="label-control"></label>
                        <input asp-for="AdmissionDate" class="form-control" />
                        <span asp-validation-for="AdmissionDate" class="text-danger"></span>
                    </div>
                </div>
                <div class="col-md-6">
                    <div class="form-group">
                        <label asp-for="DateofBirth" class="label-control"></label>
                        <input asp-for="DateofBirth" class="form-control" />
                        <span asp-validation-for="DateofBirth" class="text-danger"></span>
                    </div>
                </div>
            </div>
            <div class="form-group">
                <button type="submit" class="btn btn-sm btn-primary rounded-0">Submit</button>
        </div>
    </div>
```

Step 6

Built and run the application; ctrl+F5.

Custom Model Validation In ASP.NET Core 3.1
Custom Model Validation In ASP.NET Core 3.1

Brought to you by: Embed Analytics and Dashboards into your product with a JavaScript SDK. Free trial.

Next Recommended Article
Getting Started With ASP.NET Core 3.1 - Part One

ASP.NET Core 3.1 | Custom Model Validation | Custom Model Validation In ASP.NET Core

Farhan Ahmed TOP 100

ASP.NET Full-Stack Developer | Blogger

87 4.5m 2

5 0

Type your comment here and press Enter Key (Minimum 10 characters)