Blazor custom form validation

In this video we will discuss how to create a custom validation attribute and perform custom form validation in Blazor.

Built-in attributes for validation in Blazor

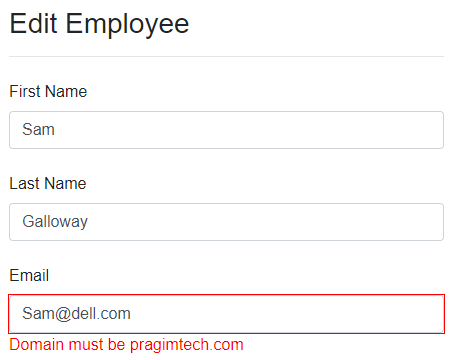
For most use cases asp.net core has several built-in attributes for model validation. Some of the common built-in attributes are listed below.



Custom validation attribute example

If we have a validation requirement that cannot be implemented using the built-in attributes, we can create a custom validation attribute.

On the following Edit Employee page, the only allowed email domain is pragimtech.com. If any other domain name is used, we want to display a validation error. We could achieve this using the built-in regular expression validator, but let's create a custom validator.



Create custom validation attribute

To create a custom validation attribute

1. Create a class that derives from the built-in abstract ValidationAttribute class and override IsValid() method.
2. IsValid() method returns null if there are no validation errors, otherwise a ValidationResult object.
3. ValidationResult accepts 2 parameters - Validation error message and the property name with which this validation error message must be associated with.
4. The public property (AllowedDomain) allows to pass the domain name instead of hard-coding it in this EmailDomainValidator class. This approach makes this validator more reusable.

using System.ComponentModel.DataAnnotations;

namespace EmployeeManagement.Models.CustomValidators

{

public class EmailDomainValidator : ValidationAttribute

{

public string AllowedDomain { get; set; }

protected override ValidationResult IsValid(object value,

ValidationContext validationContext)

{

string[] strings = value.ToString().Split('@');

if (strings[1].ToUpper() == AllowedDomain.ToUpper())

{

return null;

}

return new ValidationResult($"Domain must be {AllowedDomain}",

new[] { validationContext.MemberName });

}

}

}

Copy

Using Custom Validation Attribute in Blazor

public class Employee

{

[EmailDomainValidator(AllowedDomain = "pragimtech.com")]

public string Email { get; set; }

}

Copy

* Use the custom validation attribute just like any other built-in validation attribute.
* Email property is decorated with EmailDomainValidator attribute.
* AllowedDomain property specifies the email domain that we want to validate against.