In this video, we will discuss, **creating custom validation attribute** in asp.net mvc. We will be working with the example, that we started in [Part 80](http://csharp-video-tutorials.blogspot.com/2013/08/part-80-stringlength-attribute-in.html). Please watch [Part 80](http://csharp-video-tutorials.blogspot.com/2013/08/part-80-stringlength-attribute-in.html), and [Part 81](http://csharp-video-tutorials.blogspot.com/2013/08/part-81-range-attribute-in-aspnet-mvc.html) before proceeding.   
  
   
  
At the moment, any value outside the range of **"01/01/2000"** and **"01/01/2010"** for HireDate filed, will raise a validation error.   
[Range(typeof(DateTime), "01/01/2000", "01/01/2010")]  
[DisplayFormat(DataFormatString = "{0:d}", ApplyFormatInEditMode = true)]  
public DateTime HireDate { get; set; }  
  
But, let's say, we want the end date to be today's date instead of the hardcode "01/01/2010" value. To achieve this we would be tempted to use **DateTime.Now.ToShortDateString**() as shown below.  
[Range(typeof(DateTime), "01/01/2000", DateTime.Now.ToShortDateString())]  
[DisplayFormat(DataFormatString = "{0:d}", ApplyFormatInEditMode = true)]  
public DateTime HireDate { get; set; }  
  
At this point, if you compile, you will get an error - An attribute argument must be a constant expression, typeof expression or array creation expression of an attribute parameter type.  
  
**To fix this, we can create a custom DateRangeAttribute. Here are the steps**  
**1.** Right click on the project name in solution explorer, and add **"Common"** folder.  
**2.** Right click on the **"Common"** folder and add a class file with name = **DateRangeAttribute.cs**  
**3.** Copy and paste the following code in DateRangeAttribute.cs class file.   
using System;  
using System.ComponentModel.DataAnnotations;  
  
namespace MVCDemo.Common  
{  
    public class DateRangeAttribute : RangeAttribute  
    {  
        public DateRangeAttribute(string minimumValue)  
            : base(typeof(DateTime), minimumValue, DateTime.Now.ToShortDateString())  
        {  
  
        }  
    }  
}  
**4.** Finally decorate **"HireDate"** property with our custom **DateRangeAttribute** as shown below. Notice that, we are only passing the minimum date value. Maximum date value will be today's date. Please note, DateRangeAttribute is present in MVCDemo.Common namespace.  
[DateRange("01/01/2000")]  
[DisplayFormat(DataFormatString = "{0:d}", ApplyFormatInEditMode = true)]  
public DateTime HireDate { get; set; }  
  
**Let's now look at another example of creating a custom validation attribute.** Let's say our business rules have changed, and the HireDate property should allow any valid date that is <= Today's Date. This means, there is no minimum value restriction and the maximum value should be less than or equal to Today's date. To achieve this, let's add another custom validation attribute. Here are the steps  
**1.**Right click on the "Common" folder and add a class file with name = CurrentDateAttribute.cs

**2.** Copy and paste the following code in CurrentDateAttribute.cs class file.   
using System;  
using System.ComponentModel.DataAnnotations;  
  
namespace MVCDemo.Common  
{  
    public class CurrentDateAttribute : ValidationAttribute  
    {  
        public override bool IsValid(object value)  
        {  
            DateTime dateTime = Convert.ToDateTime(value);  
            return dateTime <= DateTime.Now;  
        }  
    }  
}  
**3.**Decorate **"HireDate"** property with our custom CurrentDateAttribute as shown below.  
[CurrentDate]  
[DisplayFormat(DataFormatString = "{0:d}", ApplyFormatInEditMode = true)]  
public DateTime HireDate { get; set; }  
  
Please note that the validation error message can be customised using named parameter **"ErrorMessage"** as shown below.  
[CurrentDate(ErrorMessage = "Hire Date must be less than or equal to Today's Date")]  
[DisplayFormat(DataFormatString = "{0:d}", ApplyFormatInEditMode = true)]  
public DateTime HireDate { get; set; }