**Regular expression attribute** is great for pattern matching validation. Let's understand using Regular expression attribute with an example. We will be continuing with the example that we started in [Part 80](http://csharp-video-tutorials.blogspot.com/2013/08/part-80-stringlength-attribute-in.html) of the [asp.net mvc tutorial.](http://www.youtube.com/user/kudvenkat/videos?view=1&flow=grid)  
  
   
  
**Here is the requirement for validating Name property**  
**1.** Name can contain first and last name with a single space.   
**2.** Last name is optional. If last name is not present, then there shouldn't be any space after the first name.  
**3.** Only upper and lower case alphabets are allowed.  
  
This requirement can be very easily met using **RegularExpression**attribute. In Employee.cs class file, decorate Name property with RegularExpression attribute.    
[RegularExpression(@"^(([A-za-z]+[\s]{1}[A-za-z]+)|([A-Za-z]+))$")]  
public string Name { get; set; }  
  
Notice that, we are passing regular expression string to the attribute constructor. Regular expressions are great for pattern matching and ensures that, the value for name property is in the format that we want. Also, notice that we are using a verbatim literal(@ symbol) string, as we don't want escape sequences to be processed. We discussed verbatim literal in [Part 4](http://csharp-video-tutorials.blogspot.com/2012/06/part-4-c-tutorial-string-type.html) of [C# tutorial](http://www.youtube.com/user/kudvenkat/videos?view=1&flow=grid).  
  
Understanding and writing regular expressions is beyond the scope of this video. If you are interested in learning to write regular expressions, here is a link from MSDN  
<http://msdn.microsoft.com/en-us/library/az24scfc.aspx>  
  
The following website is very helpful, for writing and testing regular expressions. This website also contains commonly used regular expressions. Infact, I have picked up the regular expression for validating Name property from here.  
<http://gskinner.com/RegExr/>  
  
Let's discuss another example of using validation attribute. A valid internet email address should have an @ and a DOT symbol in it. To match this pattern, use the following regular expression.  
^[\w-\.\_\+%]+@(?:[\w-]+\.)+[\w]{2,6}$  
  
In **Employee.cs** class file, decorate Email property with **RegularExpression**attribute as shown below.  
[RegularExpression(@"^[\w-\.\_\+%]+@(?:[\w-]+\.)+[\w]{2,6}$", ErrorMessage = "Please enter a valid email address")]  
public string Email { get; set; }