# List of Most Common Regular Expressions and Their Uses

Manish Kumar Choudhary Aug 18, 2019

163.9k 16 8

**Introduction**

A regular expression is a pattern that could be matched against an input text. The following is the important list of regular expressions that we use widely in our applications.

1. Email id validation
2. URL validation
3. Password strength validation
4. Mobile number validation
5. String pattern validation

For using a regular expression in C# server side, we need to use the following namespace.

using System.Text.RegularExpressions;

Remember to remove / at the start and the end of the string to convert a JavaScript Regex to C#.

**1. Email id validation regular expression**

Regular expression:

i. /^\w+([-+.']\w+)*@\w+([-.]\w+)*\.\w+([-.]\w+)*$/ (Email Id)
ii. /^([\w-\.]+@(?!gmail.com)(?!yahoo.com)(?!hotmail.com)([\w- ]+\.)+[\w-]{2,4})?$/ (free/domain specific email id)

**2. URL validation regular expression**

Regular expression:

i. /(http(s)?://)?([\w-]+\.)+[\w-]+[.com]+(/[/?%&=]*)?/ (with or without http)

ii. /((www\.|(http|https|ftp|news|file)+\:\/\/)[_.a-z0-9-]+\.[a-z0-9\/_:@=.+?,##%&~-]*[^.|\'|\# |!|\(|?|,| |>|

Regular expression:

i. / ^[a-z0-9\.@#\$%&]+$/ (only contains letter [a-z] digits[0-9], special characters(@#$%&))

ii. / ^(?=.*[A-Za-z])(?=.*\d)[A-Za-z\d]{8,}$/ (Minimum 8 characters at least 1 Alphabet and 1 Number)

iii. / ^(?=.*[a-z])(?=.*[A-Z])(?=.*\d)(?=.*[$@$!%*?&])[A-Za-z\d$@$!%*?&]{8,}/ (Minimum 8 characters at least 1 Uppercase Alphabet, 1 Lowercase Alphabet, 1 Number and 1 Special Character)

iv. / ^(?=.*[a-z])(?=.*[A-Z])(?=.*\d)(?=.*[$@$!%*?&])[A-Za-z\d$@$!%*?&]{8,10}/ (Minimum 8 and Maximum 10 characters at least 1 Uppercase Alphabet, 1 Lowercase Alphabet, 1 Number and 1 Special Character)

v. / ^[a-zA-Z0-9\s]{7,16}$/ (Minimum length 7 and Maximum length 16 Characters allowed [a–z] [A-Z] [0-9])

**4. Mobile number validation regular expression**

Regular expression:

i. / ^((\+){0,1}91(\s){0,1}(\-){0,1}(\s){0,1}){0,1}9[0-9](\s){0,1}(\-){0,1}(\s){0,1}[1-9]{1}[0-9]{7}$/ (without +91 or 0)

ii. /^((\\+91-?)|0)?[0-9]{10}$/ (with or without +91 or 0)

iii. ^((\\+|00)(\\d{1,3})[\\s-]?)?(\\d{10})$/ (split the number and the country code)

**5. String pattern validation regular expression**

Regular expression:

i. /(?s)^((?!manish).)*$/ (string contains manish)
ii. \d/ (at list one digit )
iii. /(.)*(\\d)(.)* / (contains number)
iv. /^\d$/ (contains only number )
v. /^\d{11}$/ (contains only 11 digit number )
vi. /^[a-zA-Z]+$/ (contains only letter )
vii. /^[a-zA-Z0-9]+$/ (contains only letter and number )

**Use of the regular expressions**

Use the preceding regular expressions in the following ways.

In the following example, I am showing an email validation in various ways. Just replace the regular expression and use any of the others to use another validation.

**Using JavaScript**

```
<script type="text/javascript">
        function validateEmailId(email) {
            var reg = regular expression above pattern
            if (reg.test(email)) {
                mesg.innerHTML = "";
                return true;
            }
            else {
                mesg.style.color = "red";
                mesg.innerHTML = "Please provide a valid email address";
                return false;
            }
        }
    </script>
```

Call the preceding method like.

Email Address:

```
<asp:TextBox ID="txtemail" runat="server" onblur="validateEmailId(this.valu
</asp:TextBox>
<span id="mesg" style="font-size: small; position: relative;">
 </span>
```

**Using C# server side**

Using a normal function:

ASP.NET

Email Address

```
<asp:TextBox ID="txtemail" runat="server" ></asp:TextBox>
    <asp:Label ID="lblmsg" runat="server" ></asp:Label>
    <br/>
    <asp:Button ID="btnsubmit" runat="server" Text="Submit"
        onclick="btnsubmit_Click"  />
```

C#

```
private bool validateEmailId(string emailId)
{
    return Regex.IsMatch
    (
        emailId,
        @"^\w+([-+.']\w+)*@\w+([-.]\w+)*\.\w+([-.]\w+)*$",
        RegexOptions.IgnoreCase
    );
}
```

Call the preceding function in a button click.

```
protected void btnsubmit_Click(object sender, EventArgs e)
{
    if (validateEmailId(txtemail.Text.Trim()))
    {
        lblmsg.Text = string.Empty;
    }
    else
    {
        lblmsg.Text = "Please provide a valid email address";
        lblmsg.ForeColor = System.Drawing.Color.Red;
    }
}
```

**Using RegularExpressionValidator**

ASP.NET

Email Address:

```
<asp:TextBox ID="txtemail" runat="server" ></asp:TextBox>
     <asp:RegularExpressionValidator ID="RegularExpressionValidator1" ru
          ErrorMessage="Please provide a valid email address"
          ToolTip="Please provide a valid email address"
          ValidationExpression="^\w+([-+.']\w+)*@\w+([-.]\w+)*\.\w+
([-.]\w+)*$"
          ControlToValidate="txtemail" ForeColor="Red">Please provide a v
```

**Using CustomValidator**

ASP.NET

Scripts

```
<script type="text/javascript">
          function validateEmailId(oSrc, args) {
               if (args.Value > 0) {
                    args.IsValid = (args.Value.match(/^([\w-\.]+)@((\[[0-
9]{1,3}\.[0-9]{1,3}\.[0-9]{1,3}\.)|(([\w-]+\.)+))([a-zA-Z]{2,4}|[0-9]
{1,3})(\]?)$/));
               }
          }
     </script>
```

Email Address:

```
<script type="text/javascript">
        function validateEmailId(oSrc, args) {
            if (args.Value > 0) {
                args.IsValid = (args.Value.match(/^([\w-\.]+)@((\[[0-9]{1,3}\.[0-9]{1,3}\.[0-9]{1,3}\.)|(([\w-]+\.)+))([a-zA-Z]{2,4}|[0-9]{1,3})(\]?)$/));
            }
        }
    </script>
   Email Address:
    <asp:TextBox ID="txtemail" runat="server" ></asp:TextBox>
    <asp:CustomValidator ID="CustomValidator1" runat="server"
        ErrorMessage="Please provide a valid email address"
        ClientValidationFunction="validateEmailId" ControlToValidate="t
        Display="Dynamic" ForeColor="Red"
        ToolTip="Please provide a valid email address">Please provide a
```

Change the preceding regular expression in case you need to use another expression.

Use regular expression with Linq:

Regex.Match(hrefValue, @"^([\w-\.]+)@((\[[0-9]{1,3}\.[0-9]{1,3}\.[0-9]{1,3}\.)|(([\w-]+\.)+))([a-zA-Z]{2,4}|[0-9]{1,3})(\]?)$").Success

Using MVC:

**Data Annotations**

Suppose we have a student class like follows:

```
public partial class tblstudent
{
   public string Studentname { get; set; }
   public string Emailid { get; set; }
}
```

We can apply a regular expression like:

```
[RegularExpression(@"^([\w-\.]+)@((\[[0-9]{1,3}\.[0-9]{1,3}\.[0-9]{1,3}\.)|(([\w-]+\.)+))([a-zA-Z]{2,4}|[0-9]{1,3})(\]?)$", ErrorMessage = "Please provide a valid email address")]
public string Emailid { get; set; }
```

We can also extract a word or group of words from a string using a regular expression.

Suppose we want to extract a domain name and user name from an email id, then by using the following method we can do it.

Using C#:

```
string hrefValue = txtemail .Text .Trim ();
Match m = Regex.Match(hrefValue, @"^(\w+([-+.']\w+)*)@(\w+([-.]\w+)*\.\w+
([-.]\w+)*)$");
            Response.Write(string .Format ("UserName : {0}",  m.Groups[1].\
            Response.Write("<br/>");
            Response.Write(string.Format("Domain : {0}",  m.Groups[3].Value
```

Using JavaScript:

```
<script type="text/javascript">
         function validateEmailId(email) {
             var reg = /^(\w+([-+.']\w+)*)@(\w+([-.]\w+)*\.\w+
([-.]\w+)*)$/;

             var matches = email.match(reg);
             UserId.innerHTML ='UserName : '+ matches[1];
             Domain.innerHTML ='Domain : '+  matches[3];
         }
    </script>
```

Email Address:

```
<asp:TextBox ID="txtemail" runat="server" onblur="validateEmailId(this.valu
</asp:TextBox>
                           <br/>
                           <span id="UserId" style="font-
size: small; position: relative;"></span>
                           <br/>
                           <span id="Domain" style="font-
size: small; position: relative;"></span>
```

We need to make a group inside an expression using ( ) characters and extract that group value using the preceding process. For checking a regular expression click here.

## Summary

In this illustration you came to understand the various types of regular expressions and their uses. Here is a detailed tutorial on C# Regex class and its usage, Top C# Regex Code Examples.

Client Side Regular Expressions    Server side Regular Expressions in ASP.NET.

SIMILAR ARTICLES

Using Compiled Regular Expressions

Validation in HTML5 Using Regular Expressions

What is a Regular Expression in C#?

Regular Expressions In .NET

Formula Generator - A Regular Expression Generator Class in C# and .NET

Manish Kumar Choudhary TOP 500

Manish Kumar Choudhary is a passionate and pragmatic software engineer specializing in web application development with ASP.NET MVC, Web API, Entity Framework, NHibernet, Angular, Backbone, HTML5, and CSS. He started pro... Read more

173 5.3m 2

16 View All Comments

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