Asp.Net

Ch. 01

Validation Controls in ASP.Net

Introduction to Validation controls:

- Validation controls are most important and useful feature introduced by **ASP.Net**
- Validation controls are used to control and validate the given data.
- Validating a form is most important part of Website development.
- To check that given data is in specific format or not we can use validation controls.
- Using validation controls data validation could be done in couple of minutes.
- Validation controls will not use long coding for validating data.

Introduction to Validation controls:

- Form validation can be done in two ways :
  - Client Side Validation
  - Server Side Validation
- Client Side Validation :
  - Client Side validation means that checking of data format or validation is done at Client Side.
  - Client side validation is done by browser itself.
  - If all the action of user goes to server side, then the processing would be done on server side and then revert back the response.

Introduction to Validation controls:

- To send request each time and got response each time then it would take too much time to get the page processed.
- To avoid long time processing we can used client side validation.
- The client side validation gives faster response to Client.
- The client side validation is written in a script section.
- In ASP.NET several controls are available to carry out Client Side Validation.

Introduction to Validation controls:

- Server Side Validation :
  - Server side validation means the checking of data format or validation is carried out at Server Side.
  - This is more safe then client side validation.
  - ASP.NET provides some powerful controls to maintain Server Side Validations.

Validation Controls :

- The Controls that are used to validate the data against some input format are known as Validation Controls.
- If the data does not pass the validation, it will display an error message to the user and user would not be able to submit the data.
- There are 6 (six) validation controls :

RequiredFieldValidator

CompareValidator

RangeValidator

RegularExpression Validator

CustomValidator

ValidationSummary

Validation Controls :

- All the validation controls are derived from BaseValidator Class.
- Properties of BaseValidator class can be as :

Properties	Meaning
ControlToValidate	This property is used to set the control which we want to validate. Like, textbox, listbox, dropdownlist and many other.
ErrorMessage	This property is used to set error message which will be displayed when validator control fails.

Validation Controls :

Properties	Meaning
IsValid	It returns either TRUE or FALSE after checking the validation control. If any of the validation fails, then IsValid property returns false.
Enabled	It is used to enable or disable automatic checking of validation control.
Text	Used to set error message, which would only be displayed on validator control.
CausesValidation property of Button	We can enable and disable validation check on button click

Required Field Validator :

- RequiredFieldValidator Control :
  - Required Field Validator control is used when we want some value must be entered for the particular control with a proper validation.
  - It makes us to enter a value into a form control before submitting the form.
  - If we do not enter a value in the form, it gives error.
  - As an Example of using RequiredFieldValidator is username for login must not blank.

Required Field Validator :

- Syntax of RequireFieldValidator:

```
<asp:RequiredFieldValidator
  ID="RequiredFieldValidator1"
  runat="server"
  ControlToValidate="TextBox1"
  ErrorMessage="UserName Must not blank">
</asp:RequiredFieldValidator>
```

Required Field Validator : GUI

- We can also set RequiredFieldValidator using GUI (Graphical User Interface) too.
- Add a text control in your design view.
- Now drag and drop a new RequiredFieldValidator control on the form. (Put it at a position where you want to display output for required field validator).
- Now set the properties for RequiredFieldValidator as given

  ControlToValidate=<Name of Control>

  ErrorMessage=<Message Of Error>

Different types of Validation Controls :

- CompareValidator Control :
  - CompareValidator control is used to check value of two controls.
  - It is used to check whether both controls have got same value or not.
  - If not then it gives an error message else nothing.
  - Example of CompareValidator is Password and Confirm Password.
  - In this example we compare whether both the passwords entered are same or not.

Different types of Validation Controls :

- CompareValidator not only compare two controls, but it also checks the data type of the control.
- For example if we want that control have only integer value or string value, it is possible using this control.
- Some example of CompareValidator are:
  - Confirm email address must be same
  - Confirm password must be same

Different types of Validation Controls :

- Properties of CompareValidator can be as :

Properties	Meaning
ControlToCompare	This property is used to set the control which you want to compare with given control. Values of ControlToValidate and ControlToCompare must be same.
ValueToCompare	This property is used to set specific value to be compared with this control.

Different types of Validation Controls :

- Properties of CompareValidator can be as :

Properties	Meaning
ControlToValidate	This property is used to set the control which you want to validate with given control. Values of ControlToValidate and ControlToCompare must be same.
ValueToCompare	This property is used to set specific value to be compared with this control.

Different types of Validation Controls :

Properties	Meaning
Type	It is used to specify the type of value which you are comparing with the control. It checks that both the controls which are compare should have same type. Type can be String, Date, Currency or Integer.
Operator	The type of comparison which would be performed. Some of the operators are: ❖Equal, GreaterThan, GreaterThan Equal, LessThan, LessThan Equal, NotEqual.

Compare Validator :

- Syntax of CompareValidator: **Two Controls**

```
<asp:CompareValidator
   ID=“CompareValidator1”
   runat=“server”
   ControlToCompare=“TextBox1”
   ControlToValidate=“TextBox2”
   Operator=“Equal”
   ErrorMessage=“Password Must be Same”>
</asp:CompareValidator>
```

Different types of Validation Controls :

- RangeValidator Control :
  - RangeValidator control is used to check value of the control falls between specific range or not.
  - The value can be of type numeric, date or string.
- Some example of RangeValidator Control are:
  - 1. Input age within a specific range.
  - 2. Input Birth date within a range.

Different types of Validation Controls :

- Properties of RangeValidator can be as :

Properties	Meaning
MinimumValue	This property is used to set the Minimum value that this controls can process.
MinimumValue	This property is used to set the Minimum value that this controls can process.
Type	It is used to specify the type of value which you are comparing with the control. It checks that both the controls which are compared should have same datatype. Type can be String, Double, Date, Currency or Integer.

RangeValidator :

- Syntax of RangeValidator:

```
<asp:RangeValidator
   ID=“RangeValidator1” runat=“server”
   ControlToCompare=“TextBox1”
   ErrorMessage=“Age Between 18 to 80”>
   Type=“Integer”
   MaximumValue=“80” MinimumValue=“18”
</asp:RangeValidator>
```

Different types of Validation Controls :

- RegularExpressionValidator Control :
  - RegularExpressionValidator control is used to check value of given control against a specified expression.
  - It is used to check the value of control against string value, numeric value, alphanumeric value etc.

Different types of Validation Controls :

- Some example of RegularExpressionValidator Control are:

1. To input mobile number in specific format such as +91–9999999999.
2. To input a pin code, code must be of 6 digits only such as 361002.
3. To input an email id such as monarch.edu@gmail.com
4. To input a web URL such as http://google.co.in

- ValidationExpression property is used to set the expression format which user wants.

Different types of Validation Controls :

- To use RegularExpressionValidator control, you should be aware different types of RegularExpression symbols. Some of characters are specified as follows:

Characters	Meaning
\w	Specifies any word character (alphabets, numbers or underscore)
\d	Specifies any digit.
\D	Specifies any character that is a digit.
\s	Specifies any white space character like tab or space.

Different types of Validation Controls :

Characters	Meaning
\S	Specifies any non white space character.
|	Specifies one of the given choices.
[ ]	Provides a range of characters or set of characters.
[^]	Provides a range of characters not to be included in the input list.
*	Specifies zero or more number of characters.

Different types of Validation Controls :

Characters	Meaning
+	Specifies one or more numbers of characters.
{ , }	Provides a range of character or set of characters to be included in input list.

- Some other symbols are used in the format are as shown below:

*	-	Zero or more times
+	-	One or more times
?	-	Zero or more times

- If specified format is matched then it will return **TRUE** else it will return **FALSE.**

Different types of Validation Controls :

- Some of examples can be given as:

Characters	Meaning
\w{8,15}	Allows characters length between 8 to 15.
\d{6}	Allows digits length of 6 digit only.
[Y|y|N|n]	Allows one of characters from Y, y, N or n.
[B-E]	Allows characters between B and E.
[^B-E]	Allows characters except between B and E.

Different types of Validation Controls :

Characters	Meaning
[B-E]*	Allows zero or more characters between B and E.
[B-E]+	Allows one or more characters between B and E.

RegularExpression Validator :

- Syntax of RegularExpressionValidator:

```
<asp:RegularExpressionValidator
  ID=“RegularExpressionValidator1”
     runat=“server”
     ControlToValidate=“TextBox1”
     ErrorMessage=“Enter Only CapitalLetter”>
     ValidationExpression=“^[A-Z]*$”>
</asp:RegularExpressionValidator>
```

Custom Validator Controls :

- Custom validator control is used to validate an input with user defined function.
- Custom validator is used to define User Define Function which would validate the user input.
- We can distribute these function in two type
  - Client Side Validation Function
  - Server Side Validation Function

Custom Validator Controls :

- **Server Side Validation Function**
  - This function is executed on server side and is written in code behind file in any of .NET supported programming language.
  - The name of function is set in the OnServerValidate property of CustomValidator control.

Custom Validator Controls :

- **Server Side Validation Function**
  - This function has two arguments :
    - Object Source :
      - This gives the source name of customvalidator control.
    - ServerValidateEventArgs :
      - This argument has two important properties about the control Value and IsValid.
        - Value : Gives you the value of control
        - IsValid : Can be set as true or false.

Custom Validator :

- Syntax of CustomValidator:

```
<asp:CustomValidator
  ID=“CustomValidator1”  runat=“server”
    ControlToValidate=“TextBox1”
    ErrorMessage=“Text Error Message”>
    OnServerValidate=“<functionName>”>
</asp:CustomValidator>
```

Custom Validator Controls :

- **Client Side Validation Function**
  - Client side validation function is executed on Client Side and is written in languages like, JavaScript, VBScript …
  - The name of Client Function is specified in Client Validation Function property of CustomValidator control.
  - The Client function has two arguments
    - Object Source (written as objSource)
    - Object Arguments (written as objArgs)

Custom Validator :

- Syntax of CustomValidator:

```
<asp:CustomValidator
   ID=“CustomValidator1”  runat=“server”
   ErrorMessage=“Text Error Message”
   EnableClientScript=“True”
   ControlToValidate=“TextBox1”
   ClientValidationFunction=“<functionName>”>
</asp:CustomValidator>
```

ValidationSummary Control :

- ValidationSummary control is used to show the summary of all the validation controls which are included in the current web page.
- This control is useful when we have a large web form which has more controls visible in a single screen.
- This control is used to get the summary of validation on the same page.

ValidationSummary Control :

- Each validation control has two property named ErrorMessage and Text.
- ErrorMessage property is used by ValidationSummary control.
- ValidationSummary displays the message that are shown, when a ValidationSummary control is validated.

ValidationSummary Control :

- Some of the property of ValidationSummary

Property	**Meaning**
DisplayMode	It is used to display the summary of all the error messages in different ways such as List, BulletList or SingleParagraph. By default, the display mode is of BulletList.
HeaderText	It is used to specify the header text for ValidationSummary control.

ValidationSummary Control :

- Some of the property of ValidationSummary

Property	Meaning
ShowMessageBox	It is Boolean property which is used to see error list as MessageBox.
ShowSummary	It is Boolean property which is used to see error list as Summary.

Validation Summary :

- Syntax of Validation Summary(validator):

```
<asp:CustomValidator
  ID="CustomValidator1"  runat="server"
  ErrorMessage="Text Error Message"
  EnableClientScript="True"
  ControlToValidate="TextBox1"
  ClientValidationFunction="<functionName>
  > * </asp:CustomValidator>
```

Validation Summary :

- Syntax of Validation Summary:

  <asp:**ValidationSummary**

  ID=“ValidationSummary1” runat=“server”

  DisplayMode=“BulletList”

  HeaderText=“Enter Following Details”**/**>

  NOTE : Must Have a Command Button

Validation Group Property :

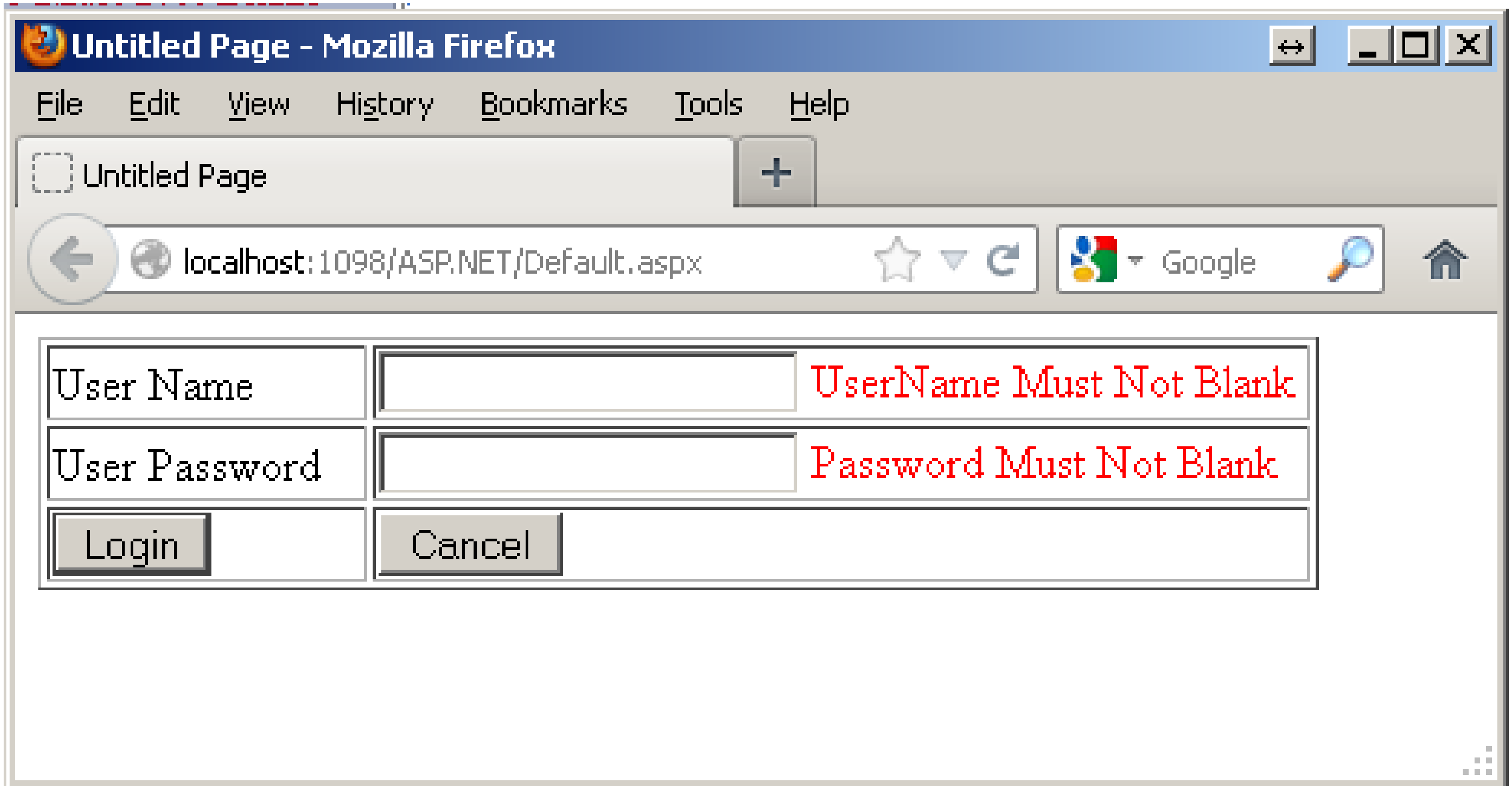
- Validation Group is a new feature that was included in ASP.NET 2.0 version.
- The validation group helps you to group the controls of a single page into separate groups, so that you can submit each group separately.
- Each group will have separate validation controls.

Validation Group Property :

- Step – 1
  - First Add Various controls as per your requirements.
- Step – 2
  - Define various Validation for controls.
  - Define Validation Summary.
- Step – 3
  - Select a control and specify Validation Group using properties. Set **ValidationGroup="GroupName"** for all controls.

Def :

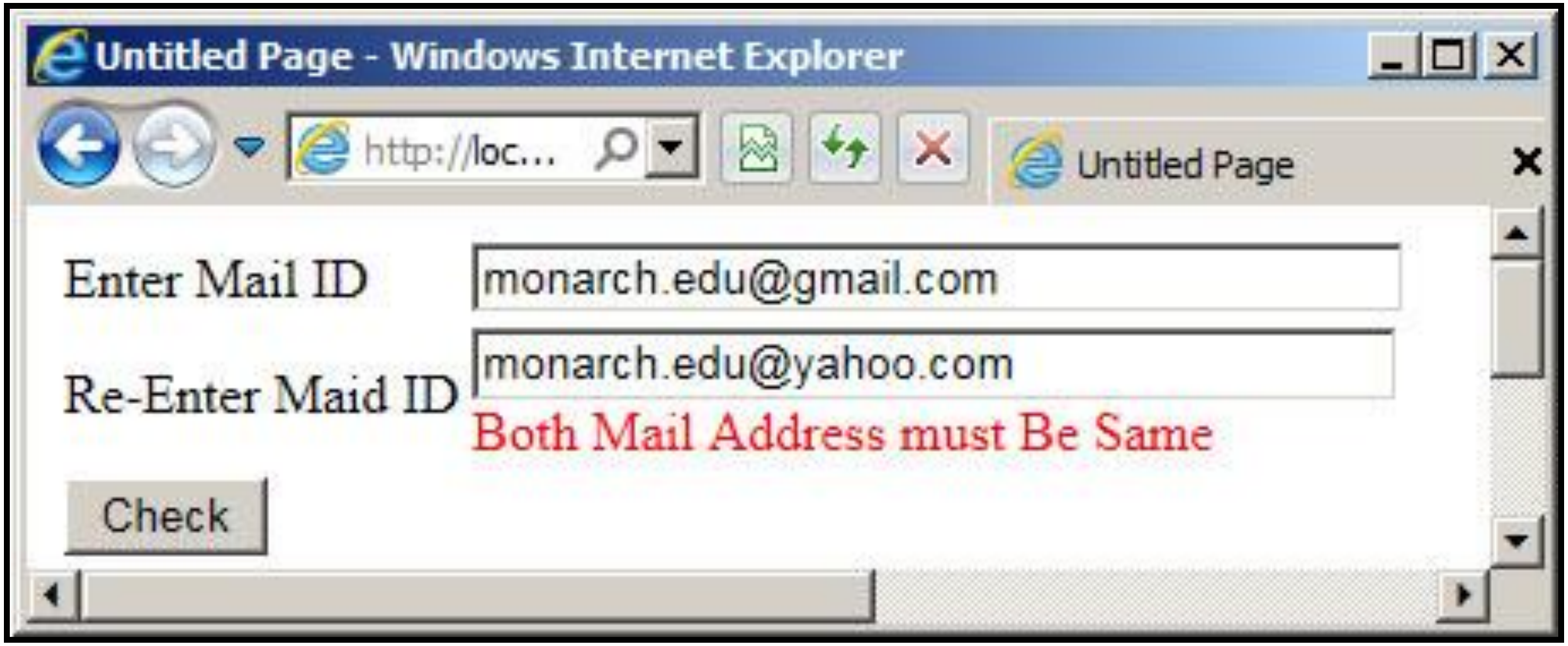
- Design a Login Form with **requiredfieldvalidator** where username and password must not blank.


Untitled Page - Mozilla Firefox
File
Edit
View
History
Bookmarks
Tools
Help
Untitled Page
+
localhost:1098/ASP.NET/Default.aspx
Google

User Name		UserName Must Not Blank
User Password		Password Must Not Blank
Login	Cancel	

Def.

- Design a form to enter user mail address and re-enter mail address again. If both are different then it will display a proper error message (CompareValidator).



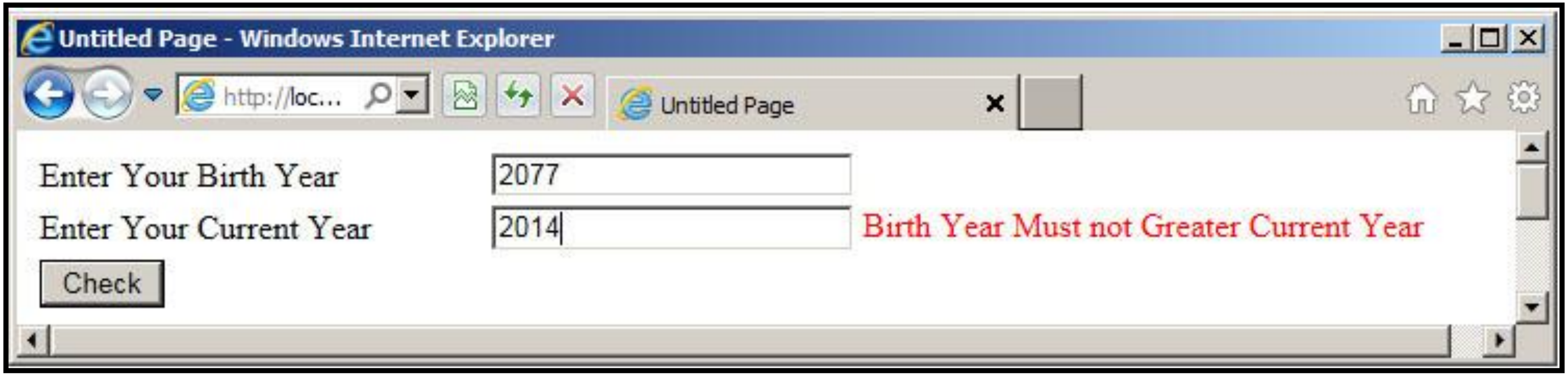
Enter Mail ID monarch.edu@gmail.com

Re-Enter Maid ID monarch.edu@yahoo.com

Both Mail Address must Be Same

Def.

- Design a form to enter your birth year and then current year...
- If birth year is greater then Current year then provide a proper validation message as given.(CompareValidator)



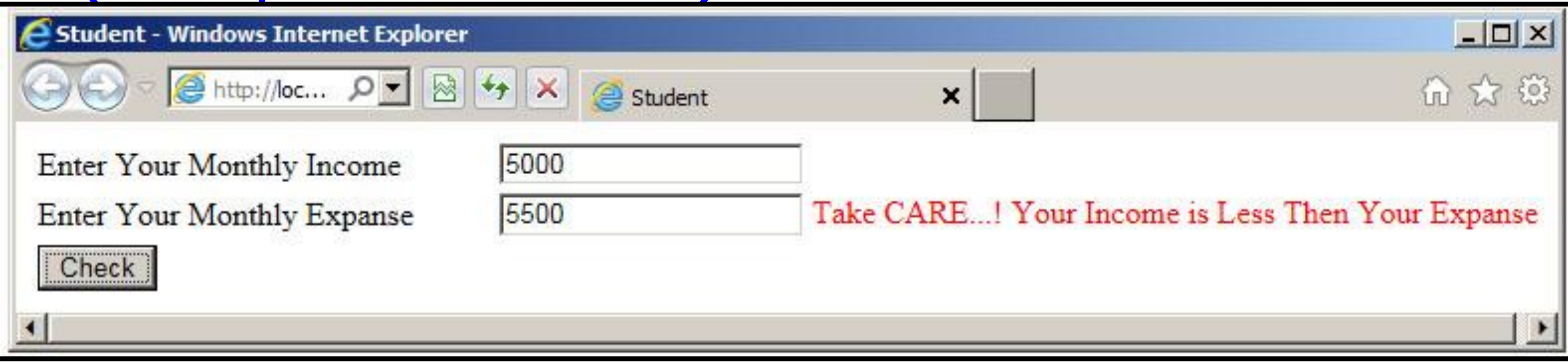
Enter Your Birth Year 2077

Enter Your Current Year 2014 Birth Year Must not Greater Current Year

Check

Def.

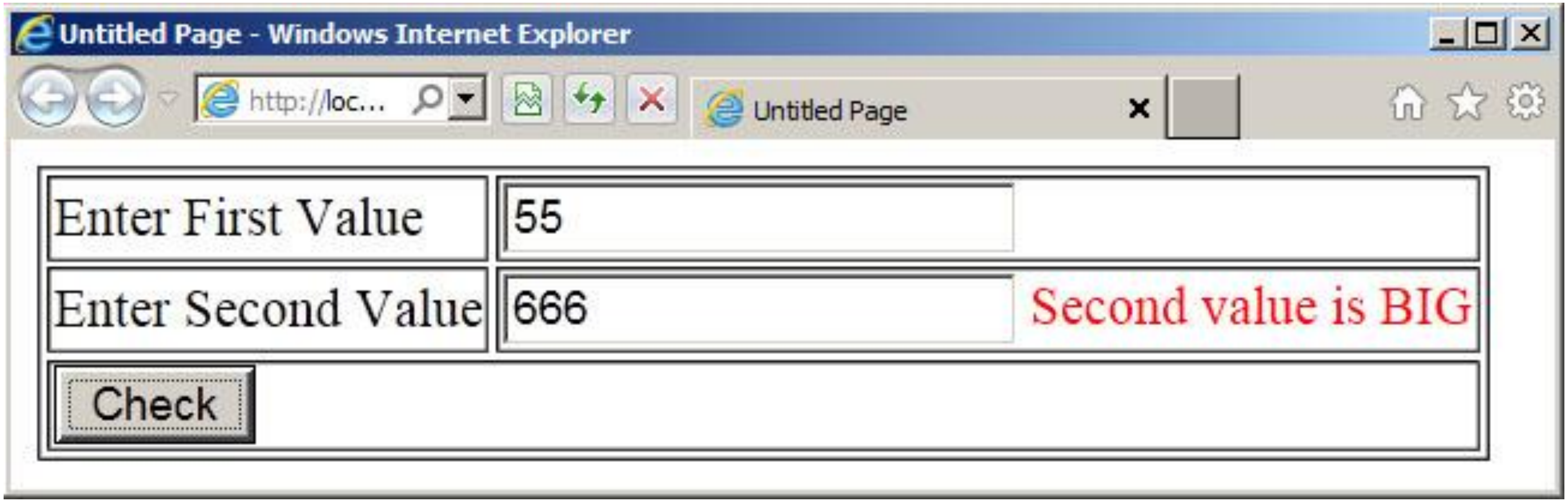
- Design a form to enter your monthly income and monthly expense.
- If expanse is greater then your income then it will give a Warning message. (CompareValidator)



Enter Your Monthly Income	5000
Enter Your Monthly Expanse	5500 Take CARE...! Your Income is Less Then Your Expanse

Def.

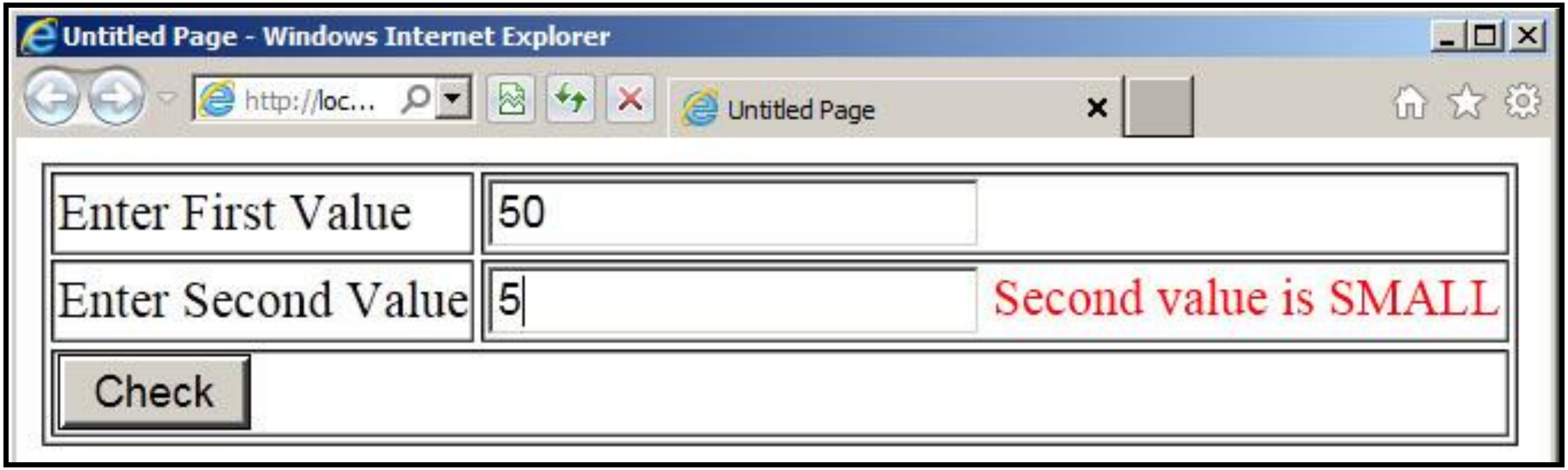
- Design a form to enter To values.
- Check that which number is GREATER? (CompareValidator)



Enter First Value	55	
Enter Second Value	666	Second value is BIG
Check		

Def.

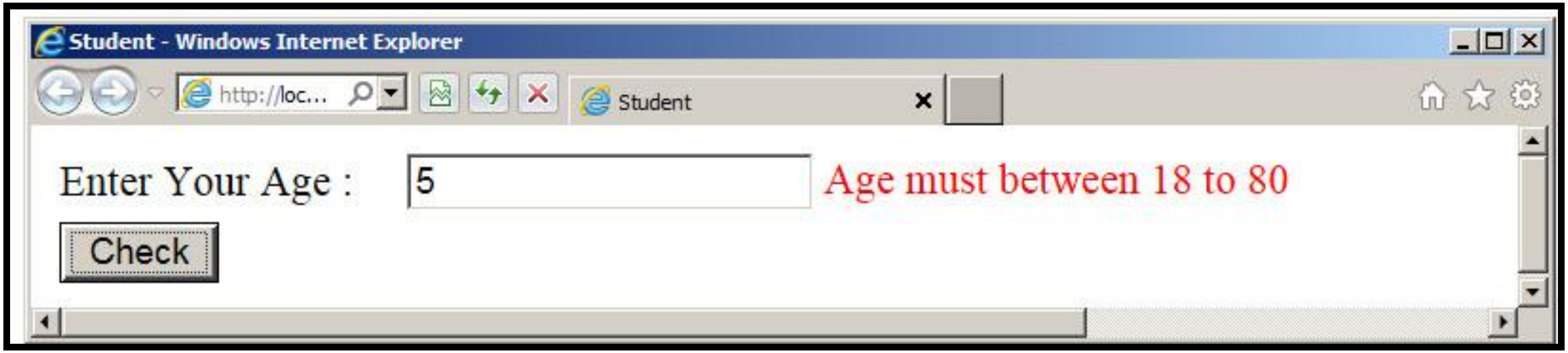
- Design a form to enter To values.
- Check that which number is SMALL?
- (CompareValidator)



Enter First Value	50
Enter Second Value	5 Second value is SMALL
Check	

Def.

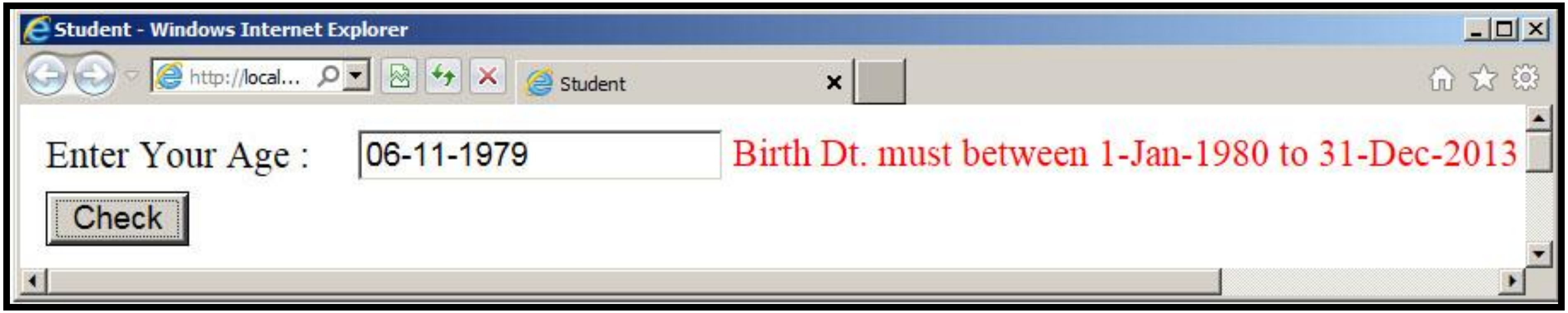
- Design a form to enter your age, if age is between 18 to 80 then valid or Give Error message “Age must between 18 to 80”
- Range Validator



Enter Your Age : 5 Age must between 18 to 80

Def.

- Design a form to enter your birth date within range of 1-Jan-1980 to 31-Dec-2013.
- If date is out of this range then it will display a proper message.
- Range Validator

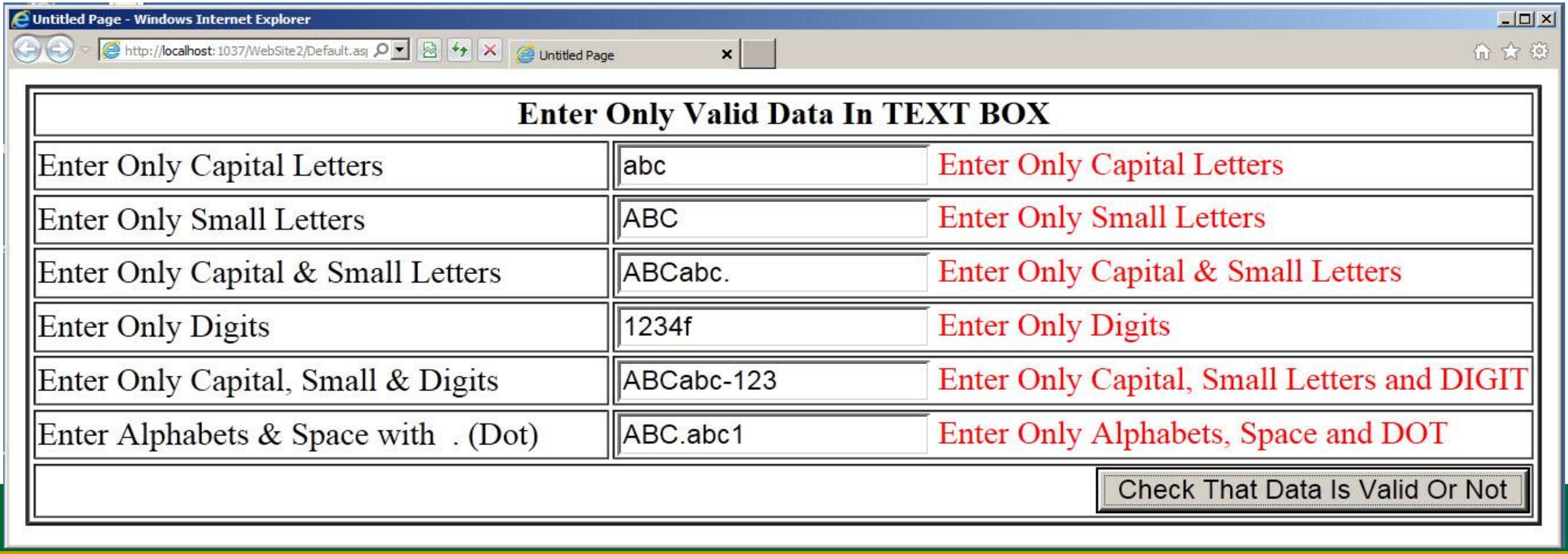


Enter Your Age : 06-11-1979 Birth Dt. must between 1-Jan-1980 to 31-Dec-2013

Def.

Design a form with

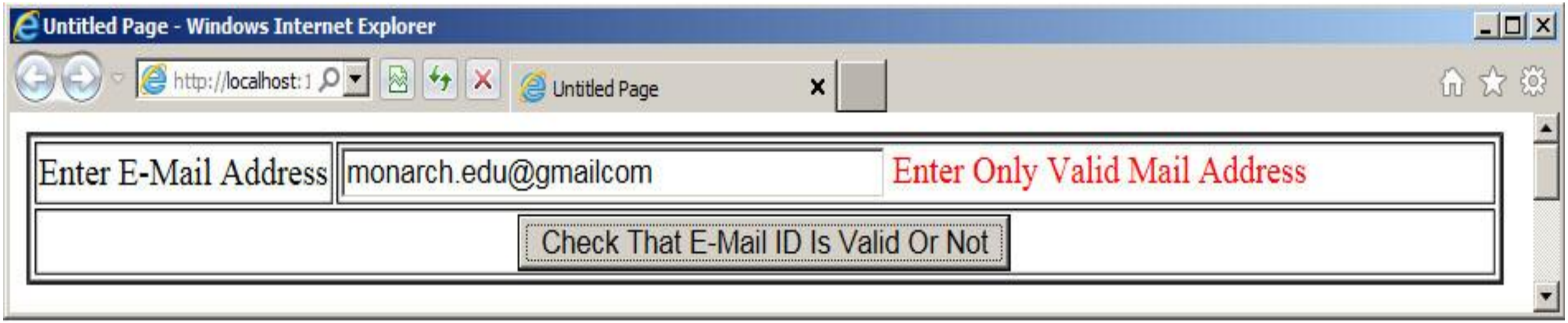
- Text Box to Enter Only Capital Letters
- Text Box to Enter Only Small Letters
- Text Box to Enter Capital & Small Letters
- Text Box to Enter Only Digits
- Text Box to Enter Capital, Small Letters & Digits only
- Text Box to Enter alphabets with . (DOT)


Untitled Page - Windows Internet Explorer
http://localhost:1037/WebSite2/Default.as|
Untitled Page

Enter Only Valid Data In TEXT BOX		
Enter Only Capital Letters	abc	Enter Only Capital Letters
Enter Only Small Letters	ABC	Enter Only Small Letters
Enter Only Capital & Small Letters	ABCabc.	Enter Only Capital & Small Letters
Enter Only Digits	1234f	Enter Only Digits
Enter Only Capital, Small & Digits	ABCabc-123	Enter Only Capital, Small Letters and DIGIT
Enter Alphabets & Space with . (Dot)	ABC.abc1	Enter Only Alphabets, Space and DOT
		Check That Data Is Valid Or Not

Def.

- Validate an e-mail address using Regular Expression Validatore…



Untitled Page - Windows Internet Explorer
http://localhost:1
Untitled Page

Enter E-Mail Address	monarch.edu@gmailcom	Enter Only Valid Mail Address

Check That E-Mail ID Is Valid Or Not

WebFor (Web Forms)

- Web form are the heart of ASP.Net.
- They provide user interface elements that give the web applications their look and feel.
- They are similar to Windows Forms in that they provide properties, methods and events for the controls that are placed onto them.

WebFor (Web Forms)

- Web Forms are made of two components :
  - Visual File (**.aspx** file)
  - Code behind File (**.aspx.cs**)
- Visual File (**.aspx** file)
  - Visual file provides the User Interface to the web page.
- Code behind File (**.aspx.cs**)
  - Code Behind File provides a place to code various events.

Features of ASP.NET Web Forms :

- Server Controls :
  - They are objects on ASP.NET web pages which run when the page is requested and that render markup to the browser.
  - Many Web Server controls are quite similar to HTML elements, such as text boxes and buttons.
- Master Pages :
  - Master pages are used to create a consistent Page layout for the Web application.

Features of ASP.NET Web Forms :

- A single master page would define the look and feel and standard behaviour that would be applicable to all the pages in the Web Application.
- We can create individual content pages that contain which we want to display.
- When users request the content pages, a combination of content page and master page would produce output that combines the layout of master page with the content from content page.

Features of ASP.NET Web Forms :

- Working with Data :
  - ASP.NET provides multiple options for storing, retrieving and displaying date. Through controls place on ASP.NET web forms.
  - We can use data bound controls to automate the presentation or input of data in web page UI elements such as text boxes and drop down lists.

Features of ASP.NET Web Forms :

- Membership :
  - ASP.NET Identity stores users’ credentials in a database created by the application. When users log in, the application validates their credentials by reading the database.
  - Project’s Account folder contains the files that implement the various parts of membership: registration, loggin in, change password and authorizing access.

Features of ASP.NET Web Forms :

- Membership :
  - Additionally, ASP.NET web forms supports OAuth and OpenID.
  - These helps us to log into our site using existing credentials, from such accounts as Facebook, Twitter, Windows Live, and Google.

Features of ASP.NET Web Forms :

- Client Script and Client Frameworks :
  - We can enhance the server based features of ASP.NET by including client script functionality in ASP.NET Web Form Pages.
  - We can use client scripts to provide a richer, more responsive user interface to users. We can also use client script to make asynchronous calls to the web server while a page is running in the browser.

Features of ASP.NET Web Forms :

- State Management :
  - ASP.NET Web Forms includes several options that help us to preserve data on both a per-page basis and an application-wide basis.
- Security :
  - Through ASP.NET web forms, we can add extensibility points and configuration options which enable us to customize various security behaviours in ASP.NET web forms.

Features of ASP.NET Web Forms :

- Performance :
  - ASP.NET web forms enables us to modify performance related to page and server control processing, state management, data access, application configuration and loading, and efficient coding practices.

Features of ASP.NET Web Forms :

- Debugging and Error Handling :
    - ASP.NET Web forms also support Debugging and error handling features which enables us in making Web Form application more efficient.

Features of ASP.NET Web Forms :

- Deployment and Hosting :
  - Visual Studio, ASP.NET and IIS provide tools that help us to deploy and host Web Forms application easily and efficiently by following simple steps.

File Upload Control

- ASP.NET Web Forms FileUpload control.
- It is an input controller which is used to upload file to the server.
- It creates a browse button on the form that pop up a window to select the file from the local machine.
- To implementFileUpload we can drag it from the toolbox in visual studio.