Web Server controls

Label
Textbox
Checkbox
Button
Multiview
FileUpload

Start Page - Microsoft Visual Studio

File Edit View Data Tools Test Analyze Window Help

New
Open
Close
Close Solution
Save Selected Items Ctrl+S
Save Selected Items As...
Save All Ctrl+Shift+S
Export Template...
Page Setup...
Print... Ctrl+P
Recent Files
Recent Projects
Exit

Project... Ctrl+Shift+N
Web Site... Shift+Alt+N
File... Ctrl+N
Project From Existing Code...

Visual Studio Team System 2008

Recent Projects
WebSite1
WindowsFormsApplication20
WindowsFormsApplication15
WindowsFormsApplication19
WindowsFormsApplication18
WindowsFormsApplication17
Open: Project...
Create: Project...

Visual Studio Developer News
Updating the news channel...

Getting Started
What's new in Visual C#?
Create Your First Application
How Do I ... ?
Learn Visual C#
Download Additional Content
MSDN Forums

Solution Expl...

Error List
0 Errors 0 Warnings 0 Messages
Description File Line C. Project

Ready

start lecture u95.zip - WinRAR (ev... Microsoft PowerPoint ... Start Page - Microsof... 10:19 AM

Microsoft Visual Studio

File Edit View Data Tools Test Analyze Window Help

Toolbox

General

There are no usable controls in this group. Drag an item onto this tex add it to the toolbox.

Solution Expl...

New Web Site

Templates: .NET Framework 3.5

Visual Studio installed templates

- ASP.NET Web Site
- ASP.NET Web Service
- Empty Web Site
- WCF Service
- ASP.NET Reports Web Site
- ASP.NET Crystal Reports Web Site

My Templates

- Search Online Templates...

A blank ASP.NET Web site (.NET Framework 3.5)

Location: File System C:\Documents and Settings\Administrator\My Documents\Visual St Browse...

Language: Visual C#

OK Cancel

Error List

0 Errors 0 Warnings

Description

start lecture u95.zip - WinRAR (ev... Microsoft PowerPoint ... Microsoft Visual Studio 10:20 AM

An ASP.NET Web Form can contain any or all of the following types of server controls:

- **HTML server controls** Programmable HTML controls. HTML server controls expose an object model that maps very closely to the HTML elements that they render.

- **Web server controls** Programmable controls with more built-in features than HTML server controls. Web server controls include not only controls for user input, but also **special-purpose controls such as a calendar, menus, and a tree view control**. Web server controls expose an object model specifically intended for programming Web pages.

- **Validation controls** Controls that incorporate logic to allow validation of user input. Validation controls enable checking for a required field, and testing against a specific value or pattern of characters, between ranges, and so on. For more information about validation controls, see Web Forms Validation.

- **User Controls** User-defined controls that are mainly used for embedding in other ASP.NET Web pages. User controls provide an easy way to create reusable page elements, **such as a search box with built-in user-defined search logic.**

<asp:label> Control

Displays static text on the Web Forms page and allows you to manipulate it programmatically.

- The **Label** control is typically used when text in the page needs to change at run time, such as in response to a button click.
- A **Label** control is better for simply displaying text than a **TextBox** control (or another control) because the resulting text is static on the page; users cannot edit it.
- The Text property of a **Label** control can be set statically at design time or dynamically at run time.
- The **Text** property can also be bound to a data source to display database information on a page.

```
<asp:label id="Message1" runat="server">Hello </asp:label>
```

id attribute

- The id attribute is used to uniquely identify the <asp:label> control so you can refer to it in your ASP.NET code.

runat="server" attribute

- The runat="server" attribute tells the server to process the control and generate HTML code to be sent to the client.

forecolor

Let's look at another example. If you want to set the color of a text message to red, you could set it like this:

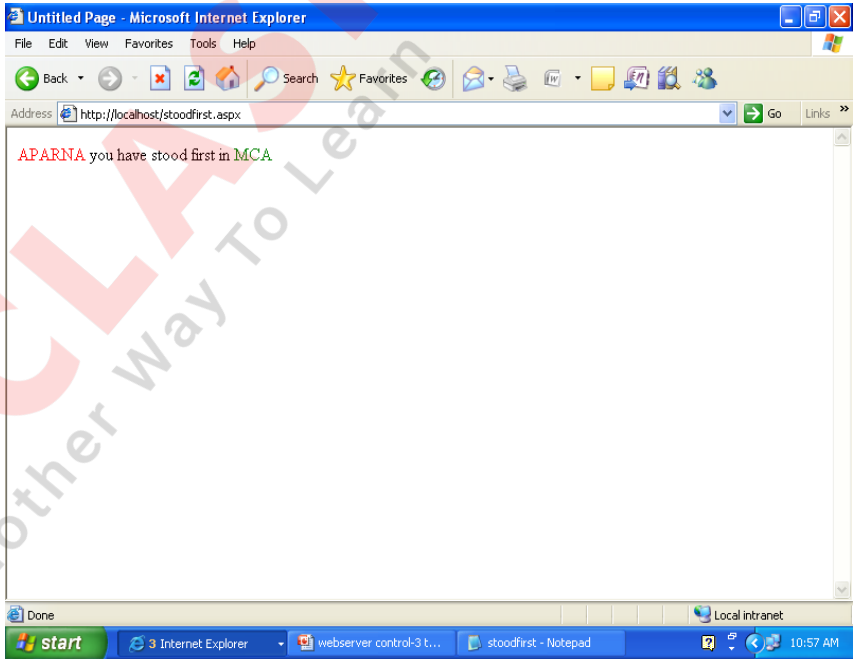
```
<asp:label id="Message1" forecolor="red" runat="server">Hello</asp:label>
```

Text attribute

- Alternatively, you can use the. This way, everything can be contained within the opening tag, in which case you need to close the tag in the following way:

```
<asp:label id="Message1" forecolor="red"  text="Hello" runat="server" />
```

```
<script language="C#" runat="server">
 void Page_Load()
 {
   Message1.Text = "APARNA";
   Message2.Text = "MCA";
 }
</script>
<html >
<head>
   <title>Untitled Page</title>
</head>
<body>
<form id="form1" runat="server">
<asp:label id="Message1" forecolor="red"
     runat="server" />
you have stood first in
<asp:label id="Message2"
     forecolor="green" runat="server" />
 </form>
</body>
</html>
```



Button Web Server Controls

- Buttons in a Web Forms page allow the user to send a *command*. They submit the form to the server and cause it to be processed along with any pending events.
- Web server controls include three types of buttons:
  - a standard command button (**<asp:Button>** control),
  - a hyperlink-style button (**<asp:LinkButton>** control), and
  - a graphical button (**<asp:ImageButton>** control) .
- All three provide similar features, but each offers a different look.

Types of Buttons

- You can create these types of Web server control buttons:
  1. **command Button** Presents a standard command button, rendered as an HTML submit button.

```
<asp:button text="Click Me" onClick="doSomething"
    runat="server" />
```

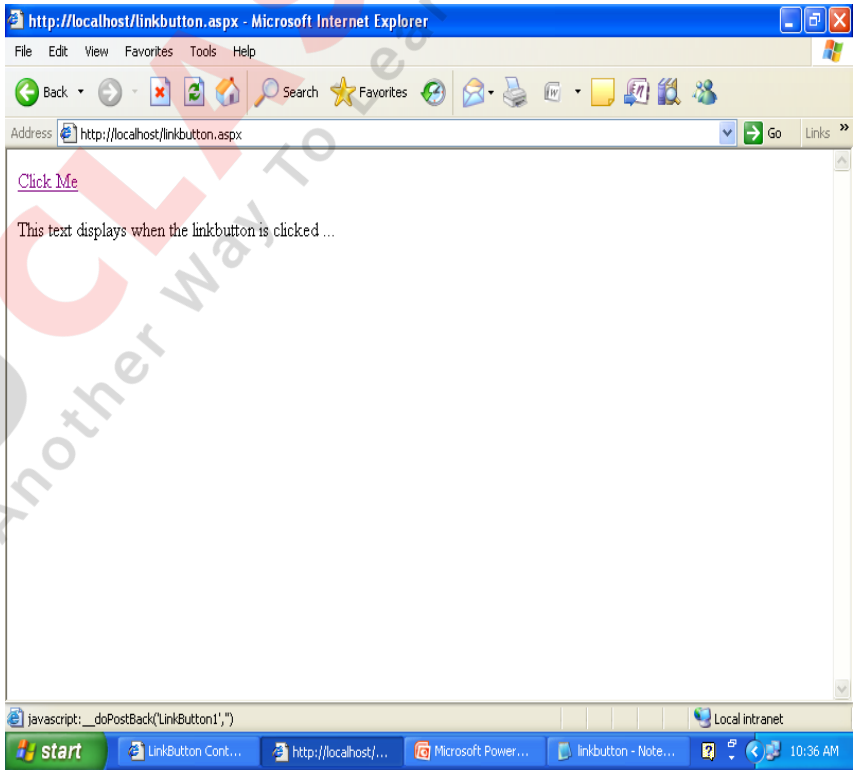
LinkButton

- Renders as a hyperlink in the page. However, it contains client-side script that causes the form to be posted back to the server. (You can create a true hyperlink using the HyperLink Control .)
- The **LinkButton** control requires some client-side script support in order to function.
- The **LinkButton** control has the same appearance as a **HyperLink** control, but has the same functionality as a **Button** control. Use the **HyperLink** control if you want to link to another Web page when the control is clicked.

- Creates a hyperlink-style button on the Web Forms page.
-

```
<asp:LinkButton id="LinkButton1" Text = "Click Me"
    onClick = "LinkButtonClickHandler" runat="server" />
```

```
<html>
<body>
<script language="C#" runat="server">
  void LinkButtonClickHandler ( Object
     sender, EventArgs e ) {
    msgLabel.Text = "This text displays when
     the linkbutton is clicked ...";
  }
</script>
</body>
<head>
<form runat=server>
  <asp:LinkButton id="LinkButton1" Text
     = "Click Me"
    onClick = "LinkButtonClickHandler"
     runat="server" />
  <p><asp:Label id="msgLabel"
     runat=server />
</form>
</head>
</html>
```



ImageButton

- Allows you to specify a graphic as a button.
- This is useful for presenting a rich button appearance. **ImageButton** controls also pinpoint where in the graphic a user has clicked, which allows you to use the button as an image map.

- <asp:**ImageButton** runat="server" ImageUrl="myPic.jpg" onClick="doSomething" />

TextBox Control

The TextBox control generates single-line and multiline text boxes.

- The **TextBox** control is an input control that lets the user enter text.
- By default, the **TextMode** property is set to **SingleLine**, which creates a text box with only one line.
- You can also set the property to **MultiLine** or **Password**.
- **MultiLine** creates a text box with more than one line.
- **Password** creates a single-line text box that masks the value entered by the user.
- The display width of the text box is determined by its **Columns** property. If the text box is a multiline text box, the display height is determined by the **Rows** property.

http://localhost/second.aspx - Microsoft Internet Explorer

File Edit View Favorites Tools Help

Back Search Favorites

Address http://localhost/second.aspx

TextBox Sample

hello shubhi

Copy Text to Label

hello shubhi

```
<html>
<head>
   <script language="C#" runat="server">
      void SubmitBtn_Click(Object Sender, EventArgs e) {
         Label1.Text = Text1.Text;
      }
   </script>
</head>
<body>
   <h3><font face="Verdana">TextBox Sample</font></h3>
   <form runat="server">
    <asp:TextBox id="Text1“  Width="200px" runat="server"/>
    <asp:Button OnClick="SubmitBtn_Click" Text="Copy Text to Label"
   Runat="server"/>
    <p>
         <asp:Label id="Label1" Text="Label1" runat="server"/>
   </form>
</body>
</html>
```

Properties

1. **TextMode** Indicates whether the text box is in single-line, multi-line, or password mode. Possible values are **Single**, **MultiLine**, and **Password**.
2. **MaxLength** The maximum number of characters allowed within text box. This property has no effect unless the **TextMode** property is set to **SingleLine** or **Password**.
3. **Rows** Number of rows within the text box. This property has no effect unless the **TextMode** property is set to **MultiLine**.
4. **Text** The text that the user has entered into the box.

Program

- Creates a check box control that allows the user to switch between a *true* or *false* state.

```
</head>

<body>
   <h3><font face="Verdana">CheckBox Example</font></h3>
   <form runat=server>
      <asp:CheckBox id=Check1 Text="CheckBox 1"
   runat="server" />
        
      <asp:button text="Submit" OnClick="SubmitBtn_Click"
   runat=server/>
      <p>
         <asp:Label id=Label1 Font-Names="arial" font-
   size="10pt" runat="server"/>
      </form>
</body>
</html>
```

```
<html>
<head>
  <script language="C#" runat="server">
   void SubmitBtn_Click(Object Sender, EventArgs e) {

      if (Check1.Checked == true) {
        Label1.Text = "Check1 is checked!";
      }
      else {
        Label1.Text = "Check1 is not checked!";
      }
    }
  </script>
```

http://localhost/checkbox1.aspx - Microsoft Internet Explorer

File Edit View Favorites Tools Help

Back Search Favorites

Address http://localhost/checkbox1.aspx

CheckBox Example

☑ CheckBox 1 Submit

Check1 is checked!

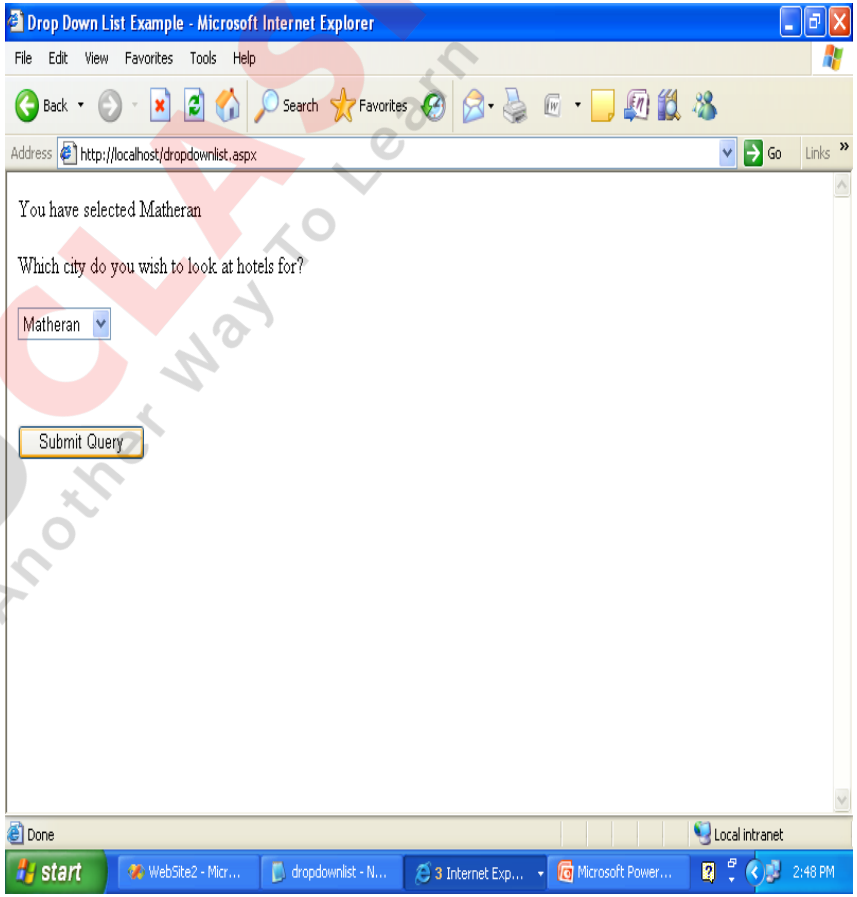
The <asp:dropdownlist> Control

- The <asp:dropdownlist> control is one of the best controls for demonstrating the usefulness of having a form control processed on the server side.

```
<asp:dropdownlist id="list1" runat="server">
 <asp:listitem>Male</asp:listitem >
 <asp:listitem >Female</asp:listitem >
</asp:dropdownlist >
```

```
<html>
 <head>
  <title>Drop Down List Example</title>
 </head>
 <body>
  <asp:label id="Message" runat="server"/>
  <br />
  <form runat="server">
      Which city do you wish to look at hotels for?<br /><br />
      <asp:dropdownlist id="list1" runat="server">
       <asp:listitem>Mount abu</asp:listitem>
       <asp:listitem>Matheran</asp:listitem>
       <asp:listitem>Ooty</asp:listitem>
      </asp:dropdownlist>
  <input type="Submit“>   </form>  </body></html>
```

```
<script runat="server"
   language="C#">
 void Page_Load()
  {
    if (Page.IsPostBack) {
   Message.Text = "You have
   selected " +
   list1.SelectedItem.Text;
    }
  }
</script>
```



```
<html>
 <head>
  <title>Drop Down List Example</title>
 </head>
 <body>
  <asp:label id="Message" runat="server"/>
  <form runat="server">
  Which city do you wish to look at hotels
    for?<br /><br />
        <asp:listbox id="list1" runat="server"
         selectionmode="multiple">
         <asp:listitem>Madrid</asp:listitem>
         <asp:listitem>Oslo</asp:listitem>
         <asp:listitem>Lisbon</asp:listitem>
        </asp:listbox>
  <input type="Submit“>   </form> </body>
</html>
```

```
<script runat="server" language="C#">
  void Page_Load()
  {
 string msg = "You have selected: <br />";
    if (list1.Items[0].Selected) {
  msg = msg + list1.Items[0].Text + "<br />";
 }
     if (list1.Items[1].Selected) {
   msg = msg + list1.Items[1].Text + "<br />";
 }
     if (list1.Items[2].Selected) {
  msg = msg + list1.Items[2].Text + "<br />";
 }
   Message.Text = msg;
  }
</script>
```

Drop Down List Example - Microsoft Internet Explorer

File Edit View Favorites Tools Help

Back Search Favorites

Address http://localhost/listbox.aspx Go Links

You have selected:
Oslo

Which city do you wish to look at hotels for?

Madrid
Oslo
Lisbon

Submit Query

Done Local intranet

start WebSite2 ... 2 Notepad 3 Intern... Microsoft ... Resume[1]... 4:13 PM

```
<html >
<head id="Head1" runat="server">
  <title>Untitled Page</title>
</head>
<body>
  <form id="form1" runat="server">
    <div>
      <h3>
        <font face="Verdana">MultiView with 3 Views</font>
      </h3>
      <asp:DropDownList ID="DropDownList1" runat="server" AutoPostBack="True"
   OnSelectedIndexChanged="IndexChanged">
        <asp:ListItem Value="0">View 1</asp:ListItem>
        <asp:ListItem Value="1">View 2</asp:ListItem>
        <asp:ListItem Value="2">View 3</asp:ListItem>
      </asp:DropDownList><br />
      <hr />
```

```
<asp:MultiView ID="MultiView1" runat="server"
  ActiveViewIndex="0">


<asp:View ID="View1" runat="server">
            Now showing View #1<br />
            <asp:TextBox ID="TextBox1"
  runat="server"></asp:TextBox><strong> </strong>
            <asp:Button ID="Button1" runat="server"
  Text="Button" /></asp:View>
```

```
<asp:View ID="View2" runat="server">
    Now showing View #2<br />
<asp:HyperLink ID="HyperLink1" runat="server"
  NavigateUrl="http://www.asp.net">
HyperLink
</asp:HyperLink>

<asp:HyperLink ID="HyperLink2" runat="server"
  NavigateUrl="http://www.asp.net">
HyperLink
</asp:HyperLink>

</asp:View>
```

```
<asp:View ID="View3" runat="server">
        Now showing View #3<br />
         <asp:Calendar ID="Calendar1" runat="server">
</asp:Calendar>

 </asp:View>

   </asp:MultiView></div>

   </form>
</body>
</html>
```

```
<%@ Page Language="C#" %>
<script runat="server">

  protected void IndexChanged(object sender, EventArgs e)
  {
    MultiView1.ActiveViewIndex =
   Convert.ToInt32(DropDownList1.SelectedValue);
  }

</script>
```

MultiView with 3 Views

View 1

Now showing View #1

Button

MultiView with 3 Views

View 3

Now showing View #3

<			August 2009			>
Sun	Mon	Tue	Wed	Thu	Fri	Sat
26	27	28	29	30	31	1
2	3	4	5	6	7	8
9	10	11	12	13	14	15
16	17	18	19	20	21	22
23	24	25	26	27	28	29
30	31	1	2	3	4	5

MultiView with 3 Views

View 2

Now showing View #2

HyperLink HyperLink

- A very useful file access feature under ASP.NET is its file upload capability.
- You can present a form containing a text box and browse button for users to navigate their local PC directories to locate a file for uploading.
- By clicking a button users can copy the local file to a server directory.
- This feature is illustrated by creating a file upload form for copying book pictures to the c:\eCommerce\BookPictures directory.
- The upload function is disabled, but you should get a good sense of how it works. The following form restricts uploads to JPG files no larger that 100 KB.

```
<asp:FileUpload id="id" Runat="Server"/>
```

File Upload Properties and Methods

- An <asp:FileUpload> control has several properties and methods to manage file uploading.

Properties:

- *Control*.**HasFile** -
- *Control*.**FileName**
  Control.**PostedFile.ContentLength**

Methods:

- *Control*.**SaveAs**("*path*")

- Whether or not a file has been selected and appears in the input box is given by the control's **HasFile** property. This property returns True or False for a script to determine whether to proceed with file uploading.
- The **FileName** property gives the name of the file selected for uploading. This is a reference only to the file name, not to the entire file path showing in the input area.
- The **PostedFile** property is a reference to the file chosen for uploading.
  - This file object's **ContentLength** property returns the size of the file in bytes.
- The control's **SaveAs(**"*path*"**)** method writes the uploaded file to a server directory given by the *path* string. The FileName property is appended to this path to copy the file to the designated directory.

A check is made on the size of the upload file as given by its FileUp.PostedFile.ContentLength property. Files that are larger than 100,000 bytes in length are rejected.

```
If  FileUp.PostedFile.ContentLength > 100000

    Message.Text = "Uploaded file size must be less than 100 KB."

Else

    { FileUp.SaveAs(Path+FileName)

Message.Text = "<b>File Uploaded</b><br/>"

Message.Text &= "File Name: “+ FileUp.FileName & "<br/>"

Message.Text &= "File Size:" + FileUp.PostedFile.ContentLength "
```

If **InStr(UCase(FileUp.FileName), ".JPG") = 0**

“,message”

Else

“upload process”

A check is made to ensure that only JPG files are uploaded. The FileUp.FileName property is checked to make sure that it contains the substring ".JPG". If not, then a message is written to the Message control and the script ends. The file name is converted to upper-case characters for the test to ensure that both ".JPG" and ".jpg" files are accepted.

FileUpLoad control

- The **FileUpLoad** control enables you to upload file to the server. It **displays a text box control** and a **browse button** that allow users to select a file to upload to the server.

- The **FileUpload control does not automatically save a file to the server** after the user selects the file to upload.
- **You must explicitly provide a control or mechanism to allow the user to submit the specified file.**

```
</head>
<body>
  <h3><font face="Verdana">File Upload</font></h3>
  <form runat=server>
    <asp:FileUpLoad id="FileUpLoad1"
  AlternateText="You cannot upload files"
  runat="server" />
    <asp:Button id="Button1" Text="Upload"
  OnClick="Button1_Click" runat="server" />
    <asp:Label id="Label1" runat="server" />
  </form>
</body>
</html>
```

```
<html>
<head>
   <script language="C#" runat="server">
      void Button1_Click(object sender, EventArgs e)
   {
    if (FileUpLoad1.HasFile)
        Label1.Text = “<br>Received =" +
   FileUpLoad1.FileName + “<br>Content Type=" +
   FileUpLoad1.PostedFile.ContentType + “<br>Length="
   + FileUpLoad1.PostedFile.ContentLength;
else
    Label1.Text = "No uploaded file";
}
 </script>
```

File Upload

Browse... Upload

File Upload

D:\meeyaaoo\10.jpg Browse... Upload

File Upload

Browse... Upload

Received= 10.jpg
Content Type= image/pjpeg
Length= 43879

```
<asp:FileUpLoad id="FileUpLoad1"
   AlternateText="You cannot upload files"
   backcolor="wheat" width=300
   borderstyle="ridge" borderwidth=1
   runat="server" />
```

File Upload

Browse... Upload

Received= 11.jpg
Content Type= image/pjpeg
Length= 43204

HasFile and SaveAs

- Hasfile: Gets a value indicating whether the FileUpload control contains a file.
- If the HasFile returns true, call the SaveAs method. If it returns false, display a message to the user indicating that the control does not contain a file.
- SaveAs Saves the contents of an uploaded file to a specified path on the Web server.

```
<script language="C#" runat="server">
  protected void UploadButton_Click ( object sender, EventArgs e ) {
    // specify the path on the server to save the uploaded file to.
    String savePath = @"c:\tmp\";

    // before attempting to perform operations on the file,
    // verify that the FileUpload control contains a file
    if ( fileUploader.HasFile ) {
      // get the name of the file to upload.
      String fileName = fileUploader.FileName;

      // append the name of the file to upload to the path.
      savePath += fileName;

      // call the SaveAs method to save the uploaded file to the specified path.
      fileUploader.SaveAs ( savePath );

      // notify the user of the name of the file was saved under.
      uploadStatusLabel.Text = "Your file was saved as " + fileName;
    } else {
      // notify the user that a filename is needed
      uploadStatusLabel.Text = "You did not specify a file to upload.";
    }
  }
</script>
```

Exercise

- The following example demonstrates how to create a FileUpload control that performs error checking. Before the file is saved, the HasFile method is called to verify that a file to upload exists. In addition, the File..::.Exists method is called to check whether a file that has the same name already exists in the path. If it does, the name of the file to upload is prefixed with a number before the SaveAs method is called. This prevents the existing file from being overwritten.

```
<%@ Page Language="C#" %>
<html  >
<head>
   <title>FileUpload.SaveAs Method Example</title>
<script runat="server">
   protected void  UploadButton_Click(object sender, EventArgs e)
   {
      // Before attempting to save the file, verify
      // that the FileUpload control contains a file.
      if (FileUpload1.HasFile)
        // Call a helper method routine to save the file.
        SaveFile(FileUpload1.PostedFile);
      else
        // Notify the user that a file was not uploaded.
        UploadStatusLabel.Text = "You did not specify a file to upload.";
   }
```

```
void SaveFile(HttpPostedFile file)
  {
    // Specify the path to save the uploaded file to.
    string savePath = "c:\\temp\\";
    // Get the name of the file to upload.
    string fileName = FileUpload1.FileName;
    // Create the path and file name to check for duplicates.
    string pathToCheck = savePath + fileName;
    // Create a temporary file name to use for checking
 duplicates.
    string tempfileName = "";
```

```
// Check to see if a file already exists with the
    // same name as the file to upload.
    if (System.IO.File.Exists(pathToCheck))
    {  int counter = 2;
      while (System.IO.File.Exists(pathToCheck))
      {   // if a file with this name already exists,
        // prefix the filename with a number.
        tempfileName =counter.ToString() +fileName ;
        pathToCheck = savePath + tempfileName;
        counter ++;
      }

       fileName = tempfileName;
```

```
    // Notify the user that the file name was changed.
UploadStatusLabel.Text = "A file with the same name already exists." +
        "<br />Your file was saved as " + fileName;
    }
    else
    { // Notify the user that the file was saved successfully.
     UploadStatusLabel.Text = "Your file was uploaded successfully.";
    }
    // Append the name of the file to upload to the path.
    savePath += fileName;
    // Call the SaveAs method to save the uploaded file to the specified directory.
    FileUpload1.SaveAs(savePath);
  }
</script>
```

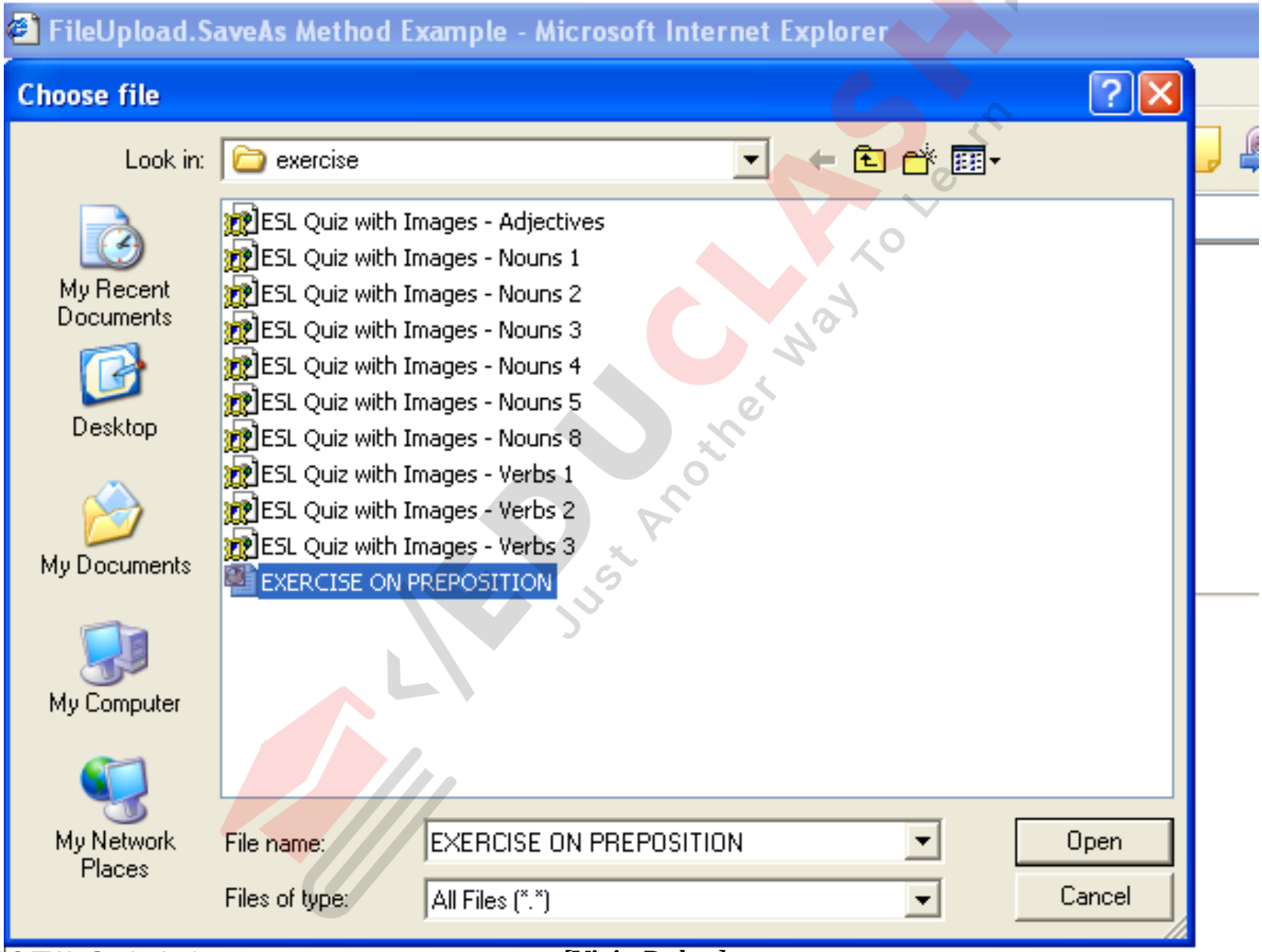
```
</head>
<body>
  <h3>FileUpload.SaveAs Method Example</h3>
  <form id="Form1" runat="server">
    <h4>Select a file to upload:</h4>
     <asp:FileUpload id="FileUpload1“ runat="server">
     </asp:FileUpload>  <br /><br />
    <asp:Button id="UploadButton"   Text="Upload file“
   OnClick="UploadButton_Click“   runat="server">
    </asp:Button>
    <hr />
    <asp:Label id="UploadStatusLabel“   runat="server">
    </asp:Label>
  </form>
</body>
</html>
```

FileUpload.SaveAs Method Example

Select a file to upload:

Browse...

Upload file


FileUpload.SaveAs Method Example - Microsoft Internet Explorer
Choose file
Look in:
exercise
ESL Quiz with Images - Adjectives
ESL Quiz with Images - Nouns 1
ESL Quiz with Images - Nouns 2
ESL Quiz with Images - Nouns 3
ESL Quiz with Images - Nouns 4
ESL Quiz with Images - Nouns 5
ESL Quiz with Images - Nouns 8
ESL Quiz with Images - Verbs 1
ESL Quiz with Images - Verbs 2
ESL Quiz with Images - Verbs 3
EXERCISE ON PREPOSITION
My Recent Documents
Desktop
My Documents
My Computer
My Network Places
File name:
EXERCISE ON PREPOSITION
Files of type:
All Files (*.*)
Open
Cancel

FileUpload.SaveAs Method Example - Microsoft Internet Explorer

File Edit View Favorites Tools Help

Back Search Favorites

Address http://localhost/fileupload_msdn.aspx Go Links

FileUpload.SaveAs Method Example

Select a file to upload:

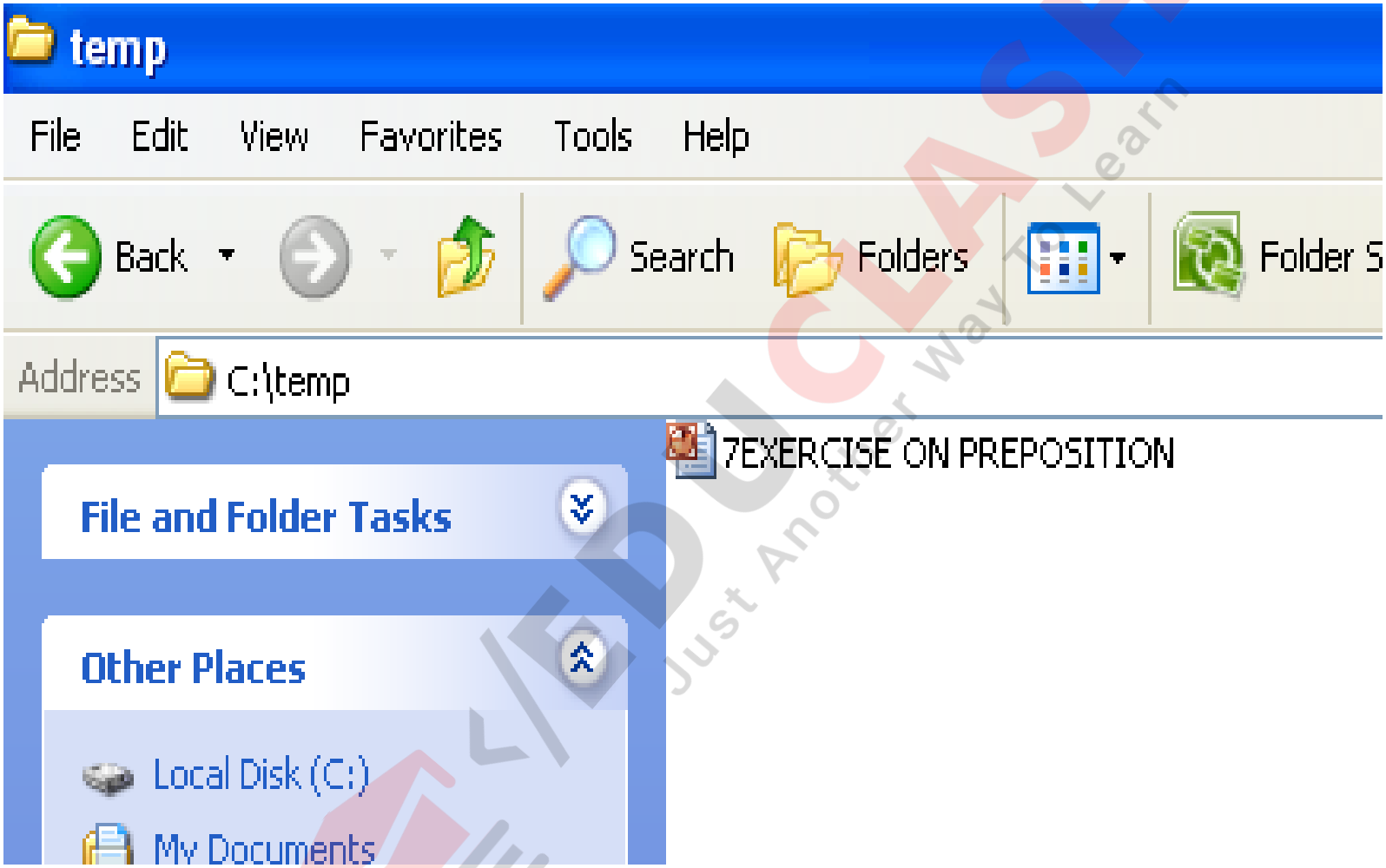
Browse...

Upload file

A file with the same name already exists.
Your file was saved as 7EXERCISE ON PREPOSITION.ppt

Done Local intranet

start 2 Inter... fileuploa... untitled -... Removab... fileuploa... 11:49 AM


temp
File
Edit
View
Favorites
Tools
Help
Back
Search
Folders
Folder S
Address
C:\temp
7EXERCISE ON PREPOSITION
File and Folder Tasks
Other Places
Local Disk (C:)
My Documents