JOHN ABBOTT
CEGEP/COLLEGE

(420-PS4-AB)
ASP .NET Features

Aref Mourtada

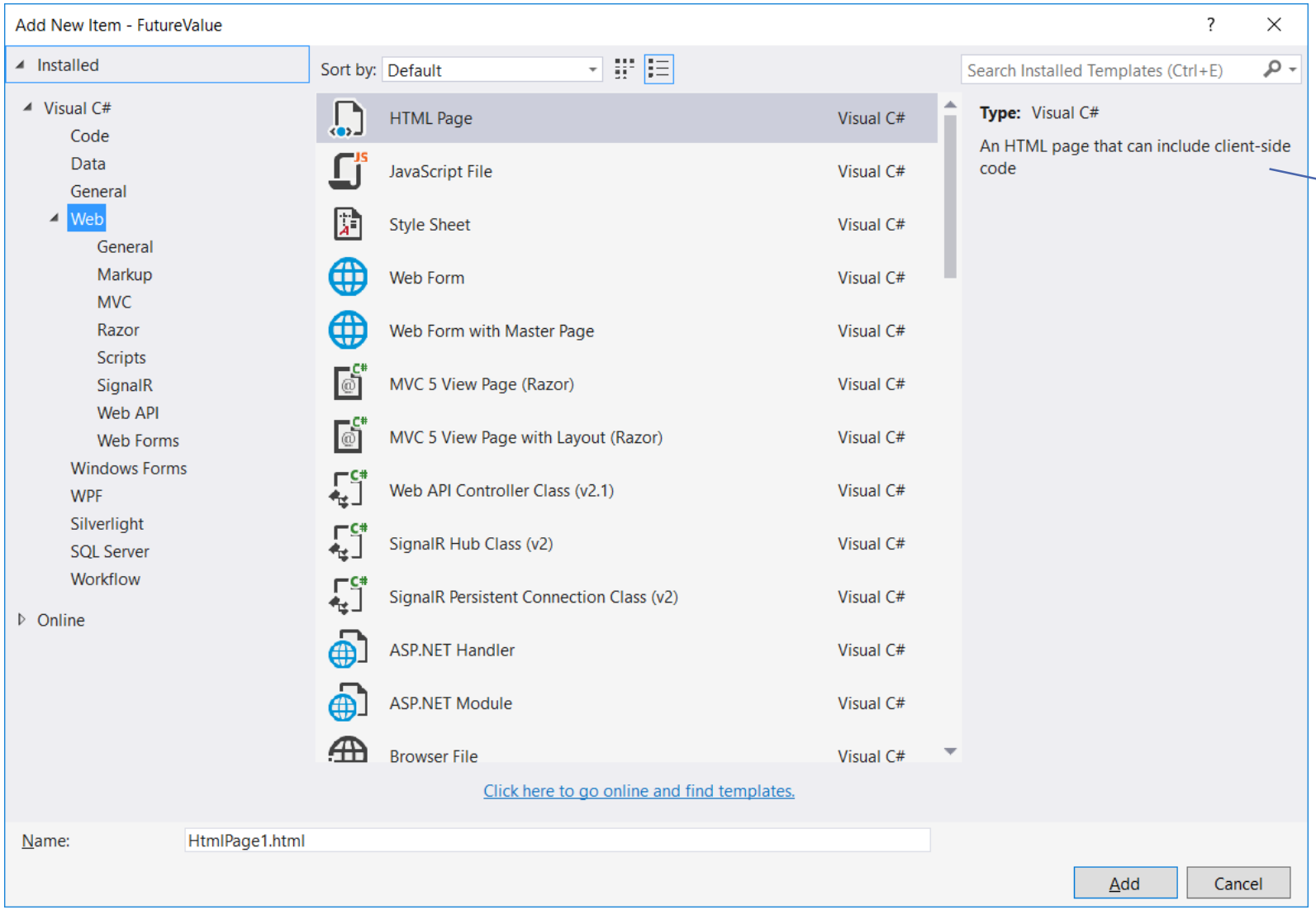
Fall 2017

Working with Files in Your Project

- Many file types of an ASP .NET each offering a distinct functionality.

- To add files:
  - Solution Explorer: right-click your project and choose Add → Add New Item
  - Project menu: Add New Item

Add New Item



Description of item

Adding Existing Files

- Already existing file that you might have created before and want to reuse such as a images, logo, CSS, scripts, ..etc.

- To add such files, use any of the following
  - Project Menu: add existing item
  - Solution Explorer: right click → add existing item
  - Copy from file explorer and paste in the Solution Explorer
  - Drag from file explorer and drop into the Solution Explorer

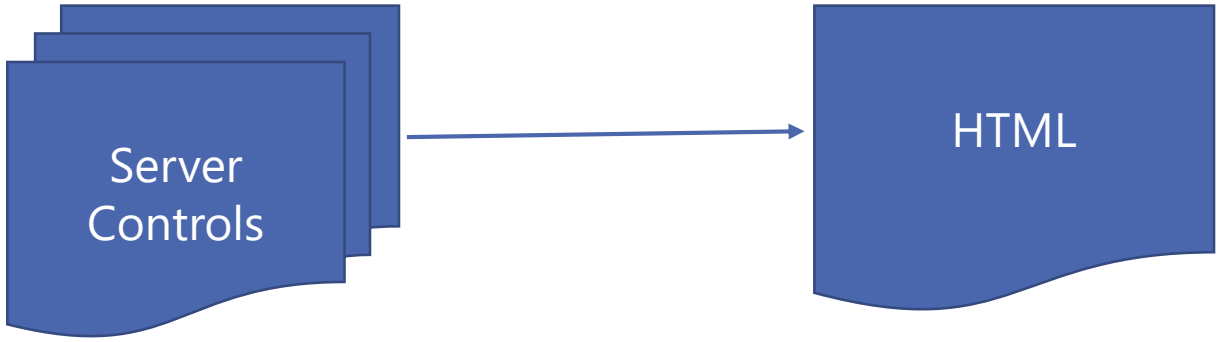
Organizing Your Site

- Because of the many files that make up your site, it's often a good idea to group them by function in separate folders.
- Example:
  - Style sheet files could go in a folder called Styles
  - .js files containing JavaScript could go in Scripts
  - User controls could go in a Controls folder
  - Master pages could be stored in a folder called MasterPages.
- This is a matter of personal preference, but structured and well-organized sites are easier to manage and understand.

Web Server Controls

- Boost productivity compared to html

- Web control ←→ Server Control
  (inter changeable terminologies)
- Main purpose of all controls is to generate HTML



Page Directive

- The Page directive is added to the top of each ASP.NET page.

```
<%@ Page Language="C#" %>
```

- Page Key Page directive features:
  - **Title**: Set the page title
  - **Language**: Specify the page's language
  - **MaintainScrollPositionOnPostback**: Maintain scrollbar positions
  - **CodeBehind**: Identify code file paths
  - **Trace**: Turn on or off tracing (logging)
  - **MasterPageFile**, **Theme**: Identify themes or master pages used.
  - **ErrorPage**: Identify an error page
  - **MetaDescription**: improve search-engine listings

ASP. NET Web Server Controls

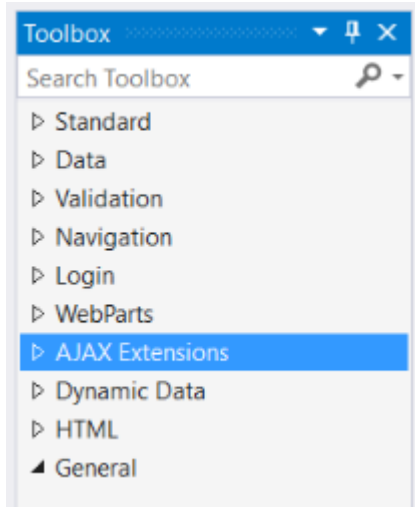
- Controls provide ASP .NET with different capabilities
  - Collect data
  - Display data
  - Validate data

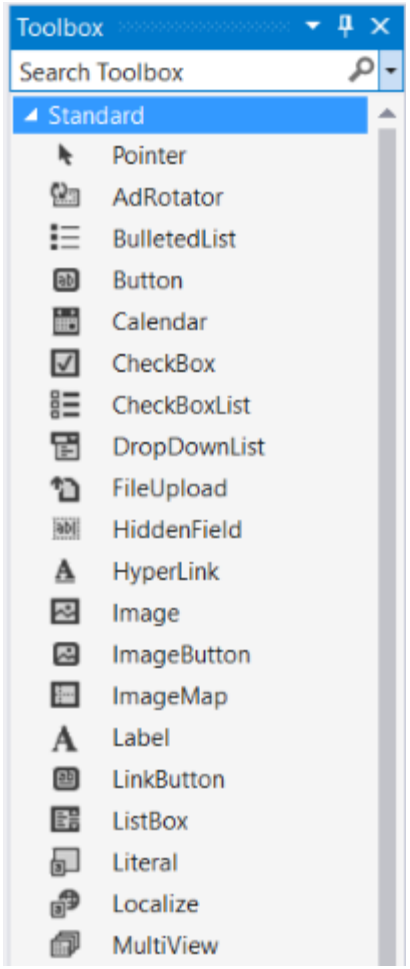
- Server controls are classed with properties, methods and events.
- Server controls can be static or dynamically generated.
- Adaptive: target browser.

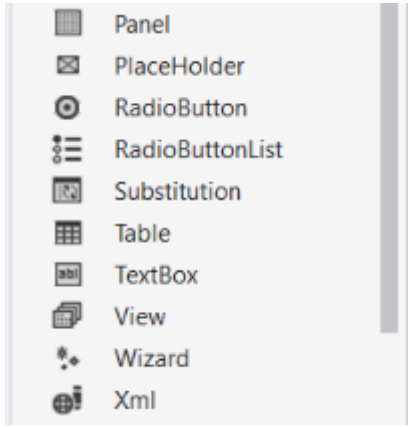
ASP.NET Control Types

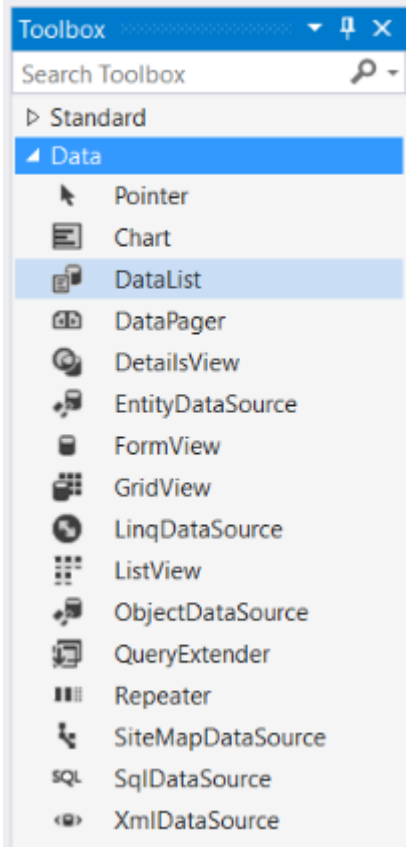
- ASP.NET controls are a key technology used by the Page class to dynamically generate HTML output.
- Four basic types of server controls exist:
  - **Web Server Controls**
    - Strongly-typed programmable objects.
  - **HTML Server Controls**
    - Similar to regular HTML elements.
    - By adding runat="Server", can be manipulated from the server side
  - **Validation Controls**
    - JavaScript
    - Used to validate Web Form submissions.
  - **User Controls**
    - Custom controls such as headers, footers and menus.

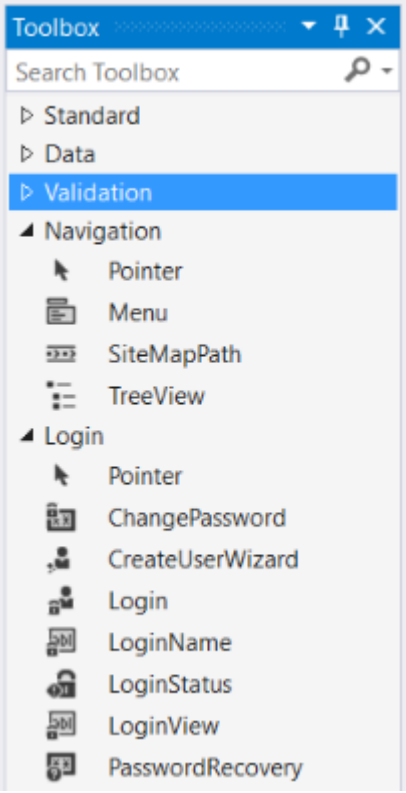
ASP .NET Web Server Controls











ASP.NET Control Examples

- Web Server Controls:

```
<asp : TextBox id=" txtName" runat=" server"  />
```

- HTML Server Controls:

```
<input type=" hidden" id="hidVal" name="hidVal" runat=" server" />
```

- Validation Controls:

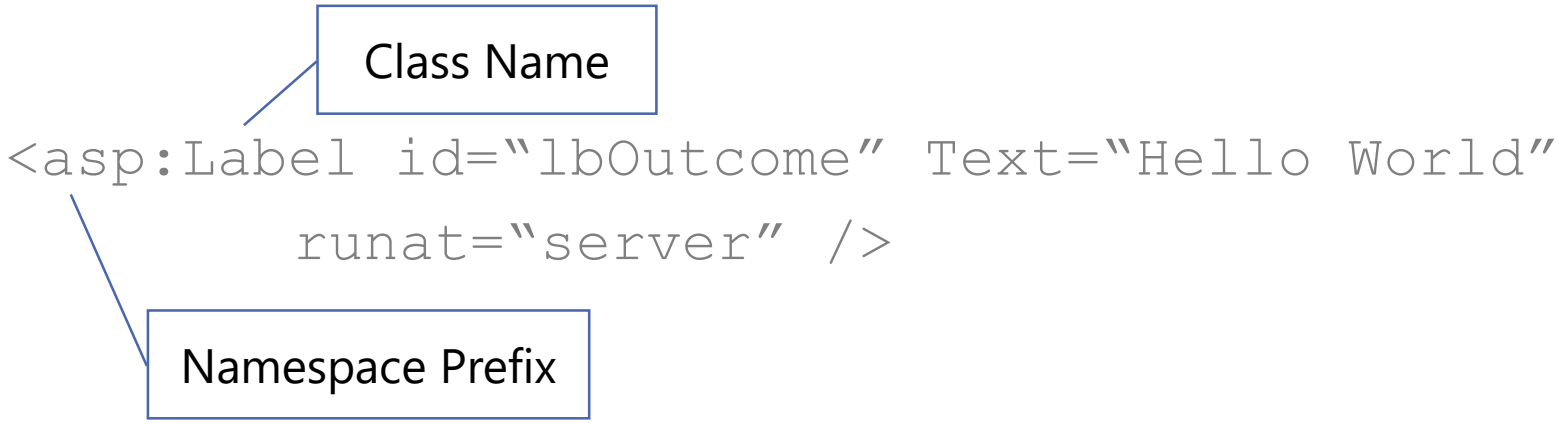
```
<asp : RequiredFieIdVaIidator id="rfvName" runat=" server"
ControlToValidate="txtName" />
```

- User Controls: (acx files)

```
<acme : Header id="ucHeader" runat="server" />
```

Declaring Server Controls

- Server Controls are used inside a web form by prefixing the control name with “asp” namespace prefix:



  - Remember that I need to define an ASP control so I would need to dynamically change on server side, otherwise just use HTML.

Web Server Control Events

- Server Controls expose different events that can be handled in Web Forms
- The OnClick attribute can be added to hook a Button Web Server control to a Click event handler:

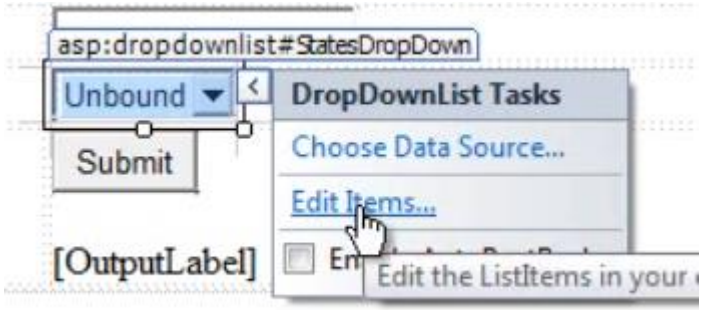

```
public void btnSubmit_Click (object sender, EventArgs e)
{
      lblMessage. Text = "You clicked btnSubmit! " ;
}

<asp:Button id="btnSubmit" OnClick="btnSubnit Click"
      runat="server" Text:" Submit" />
```

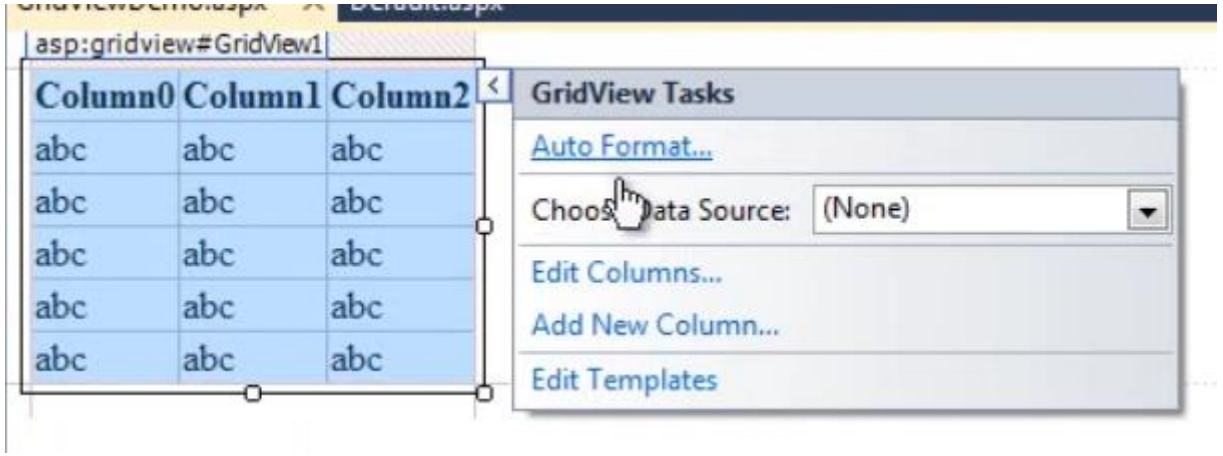
Build Project

- If you make changes in the .aspx file, no need to build the project.
- Changes in the aspx file will reflect automatically in browser.

- If make changes to the .cs file, you will need to build the project to reflect the changes in the code on the ASP .NET application.

Smart Tags on Controls





Demo: UserInformation (1)

- Write a web form to ask user for the following:
  - Name
  - Date of Birth
  - Email
  - Provence (drop down)
  - City (drop down)
  - Print to label

Demo: UserInformation (2)

- Add the following:
  - Hidden value to track number of entries in the current session
  - Clear form upon submitting (write a method)
  - Track all submitted users in a ListBox

ASP .NET Validation Controls

- Validation Controls allow user input submitted through Web Forms to validated easily.
- Supports client-side and server-side validation.

- Types of validation available:
  - Required entry
  - Validating specific criteria (value range, etc.)
  - Comparing control values (text1= text2 ?)
  - Range checking
  - Pattern matching
  - Custom Validation

ASP .NET Validation Controls

- Validation Controls that can be used in ASP .NET Web Forms include:

<asp:CompareValidator>
Multi-purpose: Comparing two values, validate dates, integers..etc.
<asp:RangeValidator>
Range of values (max & min)
<asp:RegularExpressionValidator>
Pattern matching: zip codes, IP addresses, phone numbers, email addresses
<asp:RequiredFieldValidator >
If a field is required to be enter
<asp:CustomValidator >
Write your own script
<asp:ValidationSummary>
Ties everything together

ASP .NET Validation Controls – Server Side

- Scripts at the client side can be stopped or stripped.
  - Property: EnableClientValidation (False)

- To force validation at the server:

```
Page.Validate();
if(Page.Isvalid)
{
    //your code
}
```

Demo: UserInformation (3)

- Add validation:
  1. Required field for all fields (including drop down menus)
  2. Date & email format validation.
  3. Add a summery validation.
  4. Add server-side validation.
- Add code to select a user from the ListBox and edit the information and update list box.

Setting a Default Button

- Setting the default button when a user hits the "enter" key can be done using the defaultButton attribute:

```
<form defaultButton="btnSearch" runat="server">
```

- The <asp:panel> control can override the defaultButton specified when the panel has focus:

```
<asp: Panel runat=" server" defaultButton="btnOK">
…
 </ asp : Panel>
```

Setting the Default Focus

- Setting the default focus for a page can be done using the defaultFocus attribute:

```
<form defaultFocus=" txtName" runat="server">
```

- Programmatic support for validating groups:
  - Page.SetFocus(control)
  - Page.SetFocus("ClientID")
  - txtName.Focus()

Demo: UserInformation (4)

- Set default button to submit.
- Set default focus to name text box.

ASP.NET User Controls

- User Controls allow commonly used UI functionality to be consolidated and re-used across a site
- User Controls can be registered and used in ASP.NET Web Form pages
- Examples of user controls include.
  - Login pages used in multiple places
  - Headers or Footers in Web Forms (example: copy rights)
  - Repeating Menus

Creating User Controls

- User Controls are pages created by adding a special directive at the top of the page:

```
<%@ Control Language="C#"%>
```

- The file containing the User Control directive must be saved with a ".ascx" extension
- Cannot be viewed in browser by itself.
- To use it, just simply drag and drop into the page.

Demo: UserInformation (5)

- Create a user control named Header and add to the form.

Dynamically Add Web Controls

- You can add a new control on the run.
  - But you do not want to be added in the right place in the page.

```
<asp:PlaceHolder ID="tempPlaceHolder" runat="server" />
```

- Use PlaceHolder to add
  - ASP .NET Control: button, label ..etc.
  - A user defined control: Page.LoadControl("~/path/name.ascx);

Web Control Status

- The EnableViewState property is set to True for all controls.
- EnableViewState enable a control to automatically save its state for the use is round-trips (post backs).
- If you need a control to reset to default set it to False.
- Not all controls rely on View State all the time because they are rendered as standard HTML form controls in the browser once the aspx page loads.

Q & A