CSIS 4135

Forms, Web Applications, and an Introduction to ASP.NET

Web Applications

- Making an application run on the Web is not a simple task
- amazon.com is not at all like MS Word
- Code generally needs to be written for both client-side and server-side processing
  - Much more than plain HTML
  - Web apps need *forms*

HTML Forms

- Enables creating interactive user interface elements
  - Buttons
  - Text boxes
  - Drop down lists
  - Check boxes
- User fills out the form and submits it
- Form data is sent to the Web server via HTTP

HTML Forms Example

```
<form>
  <input type="text" name="op1" />
   +
  <input type="text" name="op2" />
  <input type="submit" value=" = " />
</form>
```

HTML Forms (with GET)

When the user clicks the submit (=) button for the previous example, the browser issues a request like

```
GET /calc.html?op1=2&op2=7 HTTP/1.1
  …
```

- See the URL in a browser window when this is done

HTML Forms (with POST)

Example:

```
<form method="post">
  <input type="text" name="op1" />
   +
  <input type="text" name="op2" />
  <input type="submit" value=" = " />
</form>
```

- Sends a request like:

  POST /calc.html HTTP/1.1
  …
  [blank line]
  op1=2&op2=7

Processing Form Data

- How does the form input data get processed?
- Lots of ways to do this
  - CGI
  - ISAPI
  - ASP
  - ASP.NET
  - JSP
  - Etc…
- All of these require server-side processing

Processing Form Data

Whatever kind of processing is done, we want the result to look like this:

```
<html>
  <body>
     <form>
       <input type="text" name="op1" value="2" />
       +
       <input type="text" name="op2" value="7" />
       <input type="submit" value=" = " />
       9
     </form>
  </body>
</html>
```

Active Server Pages (ASP)

- Supports mixing processing code in an HTML document
- A block of ASP code is delimited by
  `<% ASP code %>`
- Examples:

```
Current time: <% = now()%>
Two plus two is <% = 2+2 %>
```

Postback

- When the user clicks the submit button, form data is *posted back* to the server
- This term will be used extensively in ASP.NET programming

Javascript

- Developed by Netscape
- Only thing in common with Java is the first 4 letters of the name (and some syntax)
- Does client-side processing, so the browser must support it

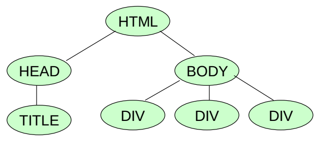
Javascript Example

```
<html>
<head>
<script language="javascript">
function MsgBox(textstr) {
  alert("You typed: " + textstr) }
</script>
</head>
<body>
 <form>
   <input name="text1" type="text">
   <input name="submit" type="button" value="Show me"
          onClick="MsgBox(form.text1.value)">
  </form>
</body>
</html>
```

The DOM

- Javascript and other client-side scripting languages have access to the page as an in-memory DOM (Document Object Model) tree maintained by the browser
- This gives the browser the ability to change parts of the rendered page without reloading the page from the server

The DOM



HTML vs. XML

- HTML is *not* XML
- HTML is used to control the layout of a document
- XML is used to structure the data in a document
- The DOM tree is actually an XML document maintained by the browser
- We’ll study XML later

Background
Web Development Technologies

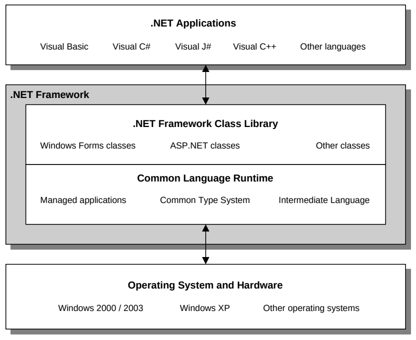
Client-side
- HTML, DHTML, JavaScript

Server-side
- ASP.NET, JSP, PHP, …

ASP.NET Overview

- ASP.NET provides services to allow the creation, deployment, and execution of Web Applications and Web Services
- ASP.NET is a server-side technology
- Web Applications are built using Web Forms
- Web Forms are designed to make building web-based applications easy

The .NET Framework



.NET Framework Components

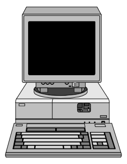
Class Library
- Pre-written classes available to all .NET programming languages
- Organized into groups called namespaces
- The classes that support ASP.NET web programs are in the System.Web namespace.

Common Language Runtime (CLR)
- Manages the execution of .NET programs
- Includes the Common Type System that ensures that all .NET applications use the same data types

Microsoft Intermediate Language

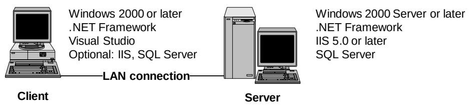
- All .NET programs are compiled into Microsoft Intermediate Language (MSIL).
- MSIL is stored on disk in an assembly.
- The assemblies are run by the CLR.

Standalone development

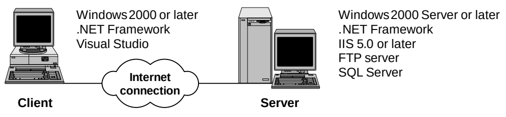


Windows 2000 or later
.NET Framework
Visual Studio
SQL Server Express
Optional: IIS, SQL Server

Local area network development



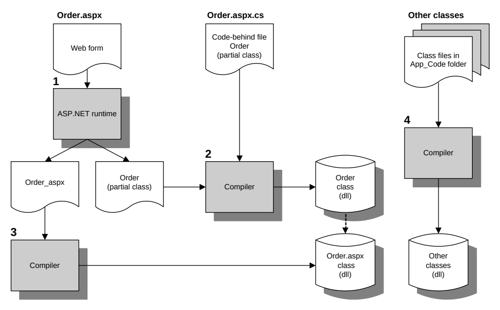
Internet development



Web form concepts

- For each web form, ASP.NET keeps two files
- The file with the aspx extension holds the HTML code and the asp tags for the server controls
- The file with the aspx.cs extension is the code-behind file that contains the C# code for the form
- If an ASP.NET application requires other classes, they are kept in the App_Code folder

How an ASP.NET application is compiled



When an ASP.NET page is requested for the first time

1. The aspx file for the web form gets divided into a full class and a partial class.
2. The partial class is compiled with the code-behind file (another partial class) into an assembly (.dll).
3. The remaining class gets compiled into another assembly that inherits the class in the first assembly.
4. Any other class files in the application's App_Code folder are compiled into a single assembly.
5. ASP.NET creates an instance of the page from the page's final assembly.
6. ASP.NET raises the appropriate events, and the page generates the HTML that's passed back to IIS for the response.

What happens when an ASP.NET page is requested again

- ASP.NET creates an instance of the page from the page's final assembly.
- ASP.NET raises the appropriate events, and the page generates the HTML that's passed back to IIS for the response.

  Note: The classes aren't recompiled.

HTML Concepts

- The HTML that is sent to the browser consists entirely of standard HTML tags.
- All of the ASP.NET tags are converted to standard HTML when the page is rendered.
- View state data is stored in a hidden input field within the HTML.
- The view state data is encrypted so you can't read it.
- User entries are returned to the browser as part of the HTML for the page.

Event Handling

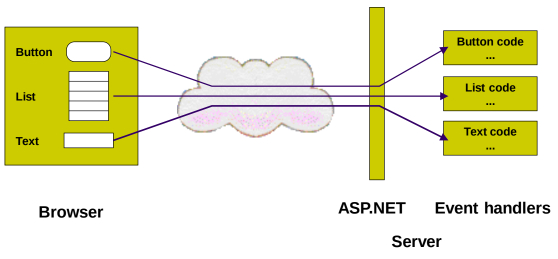
- ASP.NET sends each server control a series of messages called *events*
- If a control has a matching *event handler*, ASP.NET will run it

```
public void Submit_Click (object sender, System.EventArgs e) {
  Label1.Text = "Hello, the time is " + DateTime.Now;
}
```

Server Controls

- An ASP.NET page itself is a server control of type `Page`
- Every server control has a `Controls` collection that can store other controls
- Program code can manipulate server controls, create new ones, etc.

Programming Model
Controls and Events



ASP.NET Event Model

- Events are typically raised by the client
  - Some cause immediate posting to server
  - Some are stored until next time the page is posted back to server
  - Event message transmitted to the server via HTTP POST
- All events are handled on the server
  - Events handled by *delegates*
  - Void method with 2 params (sender & args)

Page Life Cycle Summary

- ASP.NET retrieves the .aspx file and loads it into memory. If a tag has a runat="server" attribute, the corresponding server control is loaded into memory. Otherwise an ordinary HTML tag is saved unchanged
- Program code for server controls is run in response to various events
- When all server control event handlers are finished, each control renders itself as HTML and sends the results to the client

Programming Basics
Page Event Lifecycle

Initialize	`Page_Init`
Restore Control State	
Load Page	`Page_Load`
Control Events	
1. Change Events	`Textbox1_Changed`
2. Action Events	`Button1_Click`
Save Control State	
Render	
Unload Page	`Page_Unload`

Programming Basics
Page Loading

- `Page_Load` fires at the beginning of a request after controls are initialized
  - Input control values already populated

```
protected void Page_Load(object s, EventArgs e) {
  message.Text = textbox1.Text;
}
```

Programming Basics
Page Events

- Pages are structured using events
  - Enables clean code organization (no “Monster IF” statements)
- Code can respond to page events
  - e.g. `Page_Load`, `Page_Unload`
- Code can respond to control events
  - `Button1_Click`
  - `Textbox1_Changed`

Programming Basics
Server Control Events

- Change Events
  - By default, these execute only on next action event
  - E.g. `OnTextChanged`, `OnCheckedChanged`
  - Change events fire in arbitrary order
- Action Events
  - Cause an immediate postback to server
  - E.g. `OnClick`
- Works with any browser
  - No client script required, no applets, no ActiveX®

ASP.NET Program Execution

- The C# code is compiled into a .NET assembly
- .aspx pages contain a @Page directive that links the page to the appropriate class (w/ same name as web page)
- The ASP.NET runtime passes requests and additional info to event handler
- Page is constructed and sent to client. Page construction affected by
  - HTML
  - Server controls
  - Programmatically by code in event handlers

Programming Model
Postbacks

- A postback occurs when a page generates a form whose values are posted back to the server
- A common technique for handling form data
- Helps in creating a rich UI

Programming Model
Postbacks Maintain State

- By default, ASP.NET maintains the state of all server-side controls during a postback
- Server-side control objects are automatically populated during postback
- Works with all browsers