CSIS 4135

Data transfer over HTTP,
Query Strings, Cookies,
Managing State

Data Transfer over HTTP

- URL Encoding (Query strings)
- Form Data
- Cookies
- Redirection

URL Encoding

Parameters are key-value pairs appended to the end of a URL

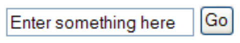
http://indra.stockton.edu/hw/Submit.aspx?class=2101&proj=hw1

Start of parameter list

Pairs separated by "&"

Encoded parameters are sent as part of the HTTP header

Form Data



```
<form name="form1" method="post" action="">
  <input type="text" name="textfield"
                      value="Enter something here">
  <input type="submit" name="Submit" value="Go">
  <input type="hidden" name="hidField" value="secret">
</form>
```

Form Data

There are two methods for sending form data.

- GET

  Form field name/value pairs are added as URL parameters
- POST

  The encoded form input is sent as part of the request message (read from standard input on server).

GET vs. POST

- GET
  - Data must be handled as name/value pairs
  - Relatively short fields
  - Security is not an issue
- POST
  - Data must be handled as name/value pairs
  - Suitable when lengthy parameters need to be passed
  - Encryption of the request is possible

ASP.NET Form Data

Server code has access to form data using the Request object

```
Using GET:  mypage.aspx?item1=something&item2=nothing

String s1 = Request["item1"];
String s2 = Request["item2"];
```

```
ASP.NET Controls:   String s = Request["TextBox1"];
```

Cookies

- A mechanism to store a small amount of data (up to 4KB) on the client
- A cookie is
  - associated with a specific web site
  - sent in HTTP header with each HTTP request
- A cookie can
  - last for only one session (until browser is closed) or
  - can persist across sessions and expire some time in the future

Cookies

```
telnet www.photo.net 80
Trying 10.101.0.100...
Connected to prd0103-006-100.
Escape character is '^]'.
GET / HTTP/1.0

HTTP/1.0 200 OK
MIME-Version: 1.0
Content-Type: text/html
Set-Cookie: ad_browser_id=87717925; Path=/; Expires=Fri, 01-Jan-2010 01:00:00 GMT
Set-Cookie:
ad_session_id=87717926%2c0%2cOLzCSFR5iruy1Seta3mAEyBG7U4JCr3h%2c986130268; Path=/; Max-Age=3600 ...
```

C# Cookie Class

- Easy to process cookies in C#
- `Response` maintains a `CookieCollection`
- Create a cookie:

```
HttpCookie cookie =
        new HttpCookie("ZipCode","08240");

Response.Cookies.Add(cookie);
```

Name

Value

Note: Value should not include comma's or semicolon's

Replacing a Cookie

Adding a cookie with the same name replaces the old cookie:

```
cookie.Value = cookie.Value + "-0195";
Response.Cookies.Add(cookie);
```

Getting Cookie Info

```
HttpCookie cookie =
            Request.Cookies.Get("ZipCode");
if (cookie == null)
{
    // cookie with name "ZipCode" doesn't exist
}
else
{
    // cookie exists, can access cookie.Value
}
```

Cookie Persistence

- By default, a cookie lasts for one session (until browser is closed)
- A cookie can persist beyond this by setting an expiration time
  - Set cookie to expire in one year

```
cookie.Expires = DateTime.Now.AddYears(1);
Response.Cookies.Add(cookie);
```

Cookie Persistence

Kill a persistent cookie by setting its expiration date to sometime in the past

```
cookie.Expires = DateTime.Now.AddHours(-1);
Response.Cookies.Add(cookie);
```

Managing State

- Recall that the HTTP protocol doesn't support maintaining state information (variables)
  - Each request/response is independent and nothing is remembered between subsequent pages
- Cookies and query strings are ways to pass data between request & response, thus maintaining state information

Maintaining State in ASP.NET

Several innovative ways were developed

- view state
- session state
- application state

We'll look at view state and session state for now

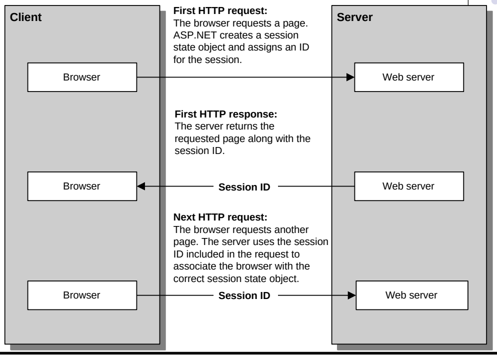
View state concepts

- *View state* is an ASP.NET feature for retaining the values of page and control properties from one execution of a page to another
- It is a collection of key/value pairs that represents the page and control properties
- You can also add your own data to the view state
- Before ASP.NET sends a page back to the client, it determines what changes the program has made to the page, encodes them in a string, and assigns the string to the value of a hidden input field named _VIEWSTATE

Session state concepts

- For each user session, ASP.NET creates a *session state object*.
- The session state object includes a session ID that is sent back to the browser as a *cookie*.
- The browser automatically returns the session ID cookie to the server with each HTTP request.
- The session ID lets the server find the right session state object.
- The session state object can be used to store and retrieve items that can be used by any of the pages in the application.

How ASP.NET maintains the state of a session


Client
Server
First HTTP request:
The browser requests a page. ASP.NET creates a session state object and assigns an ID for the session.
Browser
Web server
First HTTP response:
The server returns the requested page along with the session ID.
Browser
Session ID
Web server
Next HTTP request:
The browser requests another page. The server uses the session ID included in the request to associate the browser with the correct session state object.
Browser
Session ID
Web server

Typical uses for session state

- **To keep information about the user**, such as the user's name or whether the user has registered
- **To save objects the user is working with**, such as a shopping cart or a customer record
- **To keep track of pending operations**, such as what steps the user has completed while placing an order

Session State Examples

Add or update a session state item

```
Session["EMail"] = email;
```

Retrieve the value of a session state item

```
string email = Session["EMail"].ToString();
```

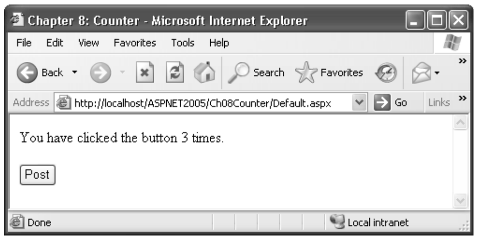
Removes an item from session state

```
Session.Remove("EMail");
```

Retrieves a session state item from a non-page class

```
string email =
  HttpContext.Current.Session["EMail"].ToString();
```

An application that counts the times the user clicks the button


Chapter 8: Counter - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Search Favorites
Address http://localhost/ASPNET2005/Ch08Counter/Default.aspx
Go Links
Post
Done
Local intranet

You have clicked the button 3 times.

Code for the Counter App

```
private int sessionCount;

protected void Page_Load(object sender, EventArgs e)
{
    if (Session["Count"] == null)
        sessionCount = 0;
    else
        sessionCount = Convert.ToInt32(Session["Count"]);
}

protected void Post_Click(object sender, EventArgs e)
{
    sessionCount++;
    SessionClicks.Text = "You have clicked the button "
                       + sessionCount + " times.";
}

protected void Page_PreRender(object sender, EventArgs e)
{
    Session["Count"] = sessionCount;
}
```

Options for tracking session IDs

Cookie-based session tracking (the default)

But if a browser doesn't support cookies, this doesn't work.

Cookieless session tracking

Encodes the session ID as part of the URL. So cookieless session state works whether or not the browser supports cookies.