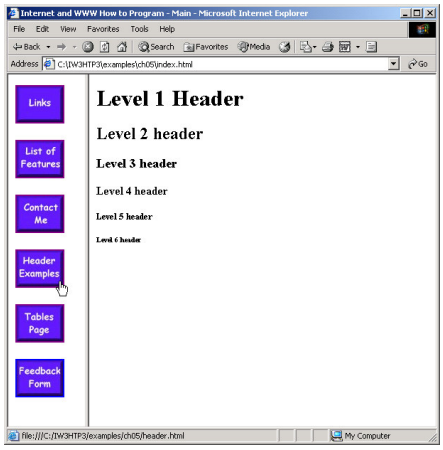
IT350 Web and Internet Programming

Fall 2008

SlideSet #15: Frames & Cookies

Frames Example



Benefits of Frames

Problems with Frames

Result: XHTML 1.1 does not support frames

Making Pages with Frames

- Two kinds of pages
  1. "Frameset" page: A page with a <frameset> (usually the index.html page)
     DTD: must use "XHTML 1.0 Frameset"

  2. "Content" page
     DTD: must use "XHTML 1.0 Transitional"

"Frameset" page

index.html

```
<?xml version = "1.0"?>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Frameset//EN"
   "http://www.w3.org/TR/xhtml1/DTD/xhtml1-frameset.dtd">

<html xmlns = "http://www.w3.org/1999/xhtml">
   <head>
      <title>Internet and WWW How to Program - Main</title>
   </head>

   <!-- the <frameset> tag sets the frame dimensions     -->
   <frameset cols = "110,*">

      <!-- frame elements specify which pages -->
      <!-- are loaded into a given frame      -->
      <frame name = "leftframe" src = "nav.html" />
      <frame name = "main" src = "main.html" />

      <noframes>  <body>
            <p>This page uses frames, but your br /owser does not
            support them.</p>

            <p>Please, <a href = "nav.html">follow this link to
            br /owse our site without frames</a>.</p>
       </body> </noframes>

   </frameset>
</html>
```

Content page

nav.html
(1 of 2)

```
1  <?xml version = "1.0"?>
2  <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
3     "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
4
5  <!-- Fig. 5.10: nav.html          -->
6  <!-- Using images as link anchors -->
7
8  <html xmlns = "http://www.w3.org/1999/xhtml">
9
10    <head>
11       <title>Internet and WWW How to Program - Navigation Bar
12       </title>
13    </head>
14
15    <body>
16
17       <p>
18          <a href = "links.html" target = "main">
19             <img src = "buttons/links.jpg" width = "65"
20                height = "50" alt = "Links Page" />
21          </a><br />
22
23          <a href = "list.html" target = "main">
24             <img src = "buttons/list.jpg" width = "65"
25                height = "50" alt = "List Example Page" />
```

5.10 Nested framesets

Nested Framesets

```
<frameset cols = "110,*">

     <frame name = "leftframe" src = "nav.html" />

     <frameset rows = "175,*">
        <frame name = "picture" src = "picture.html" />
        <frame name = "main" src = "main.html" />
     </frameset>

     <noframes>
        <body>
           <p>This page uses frames, but your browser does not
           support them.</p>

           <p>Please, <a href = "nav.html">follow this link to
           browse our site without frames</a>.</p>
        </body>
     </noframes>

  </frameset>
```


Exercise #1

Suppose the following is part of a valid XHTML document. Using this code, answer questions on this and next slide.

```
<frameset rows = "40%, *">
   <frame name="pane1" src="dog.html" />
   <frameset cols = "33%, 33%, *">
       <frame name = "pane2" src="cat.html" />
       <frame name = "pane3" src="mouse.html" />
       <frame name = "pane4" src="fox.html" />
   </frameset>
   <noframes> (omitted to save space) </noframes>
</frameset>
```

1. Draw a rough picture of what the browser window will look like. Label each frame by the file that will fill that frame.

Exercise #1 continued

2. Which of the frames given is most likely to contain navigation links?

3. Write a code snippet to create 2 links:
   a. A link to load “hippo.html” into ‘pane3’

   b. A link to load “boar.html” into the full browser window, removing frames. (hint: see the textbook)

4. What should the DOCTYPE be for the document that holds those 2 links from #3?

Exercise #2

- How might you re-create some of the benefits of frames while eliminating some of the disadvantages? What would you need?

Cookies – Again!

- Cookie
  - Data stored on user's computer to maintain information about client during and between browser sessions
  - Can be accessed through `cookie` property
  - Set expiration date through `expires` property
  - Use `escape` function to convert non-alphanumeric characters to hexadecimal escape sequences
  - `unescape` function converts hexadecimal escape sequences back to English characters

Storing Cookies – Simple Version

```
document.writeln("<br/>Cookie is: "+document.cookie);
```

```
document.cookie = "name=" + escape("J Smith");
document.writeln("<br/>Cookie is: "+document.cookie);
```

```
document.cookie = "rank=" + escape("Captain");
document.writeln("<br/>Cookie is: "+document.cookie);
```

Cookie Example #1

```
// reset the document's cookie if wrong person
function wrongPerson() {
   // reset the cookie
   document.cookie= "name=null;" + " expires=Thu, 01-Jan-95 00:00:01 GMT";

   // after removing the cookie reload the page to get a new name
   location.reload();
}

// determine whether there is a cookie
if ( document.cookie ) {
   var myCookie = unescape( document.cookie );

   // split the cookie into tokens using = as delimiter
   var cookieTokens = myCookie.split( "=" );

   // set name to the part of the cookie that follows the = sign
   name = cookieTokens[ 1 ];
}
else {
   // if there was no cookie then ask the user to input a name
   name = window.prompt( "Please enter your name", "Paul" );
   document.cookie = "name=" + escape( name );
}
document.writeln("<h1>Hello, " +name + ". </h1>" );
document.writeln( "<a href= 'javaScript:wrongPerson()' > " +
   "Click here if you are not " + name + "</a>" );
```


Cookie Example #2

```
// reset the document's cookie if wrong person
function wrongPerson() {
   // reset the cookie
   document.cookie= "name=null;" + " expires=Thu, 01-Jan-95 00:00:01 GMT";

   // after removing the cookie reload the page to get a new name
   location.reload();
}

// determine whether there is a cookie
if ( document.cookie )
{
   var cookie = document.cookie;
   var cookieTokens = cookie.split( "=" );

   // set name to the part of the cookie that follows the = sign
   name = cookieTokens[ 1 ];
   name = unescape(name);
}
else {
   // if there was no cookie then ask the user to input a name
   name = window.prompt( "Please enter your name", "Paul" );
   document.cookie = "name=" + escape( name );
}
document.writeln("<h1>Hello, " +name + ". </h1>" );
document.writeln( "<a href= 'javaScript:wrongPerson()' > " +
   "Click here if you are not " + name + "</a>" );
```

Storing Cookies – More Realistic

- By default, cookies expire when close browser
- Set "expires" attribute to make stick around longer

```
function createCookie(name,value,days) {
  if (days) {
    var date = new Date();
    date.setTime(date.getTime()+(days*24*60*60*1000));
    var expires = "; expires="+date.toGMTString();
  }
  else
     var expires = "";
   document.cookie = name+"="+escape(value)+expires;

}
function eraseCookie(name) {
  createCookie(name,"",-1);
}
```

(modified from http://www.quirksmode.org/js/cookies.html)

Parsing Cookies – More Realistic

```
// Return the 'value' of the cookie variable with name 'desiredVar'
// returns null if no match found.
function parseCookie(desiredVar) {
    // First split the pairs apart on ';'
    var pairs  = document.cookie.split(";");

    // Now split each pair on '='.  Check if have a match
    for (var i=0; i < pairs.length; i++) {
       var aPair = pairs[i];

       // remove any leading spaces
       while (aPair.charAt(0) == ' ')
          aPair = aPair.substring(1, aPair.length );

       // split into desired parts and check for match
       var cookieTokens = aPair.split("=");
       var name  = cookieTokens[0];
       var value = cookieTokens[1];
       if (name == desiredVar) {
           // found desired variable -- return value
           return unescape(value);
       }
    }
    return null;   // no match;
}
```