CS 350: Computer/Human Interaction
Lecture 02 Overview

- Basic XHTML structural and form elements
- Using PHP for CGI scripting
- PHP data types and operations
- PHP control structures
- References: HRAG, HW3S, HVAL, PINT, PZND, PW3S

CS 350: Computer/Human Interaction
HTML, XHTML, XML???

- HTML is the original markup language used on the web. XML is a more general markup language that can be used to describe data as well as text. XHTML is HTML defined in terms of XML. XHTML is almost identical to HTML 4.01. XHTML is stricter and cleaner than HTML and was intended to replace HTML.
- W3C is working on HTML 5 to unify HTML 4 and XHTML 1.

CS 350: Computer/Human Interaction
HTML and XHTML Differences

- Documents must be well-formed and elements must be properly nested
- Tags must be lower case.
  E.g. `<body>` not `<BODY>`
- All elements must be closed
  E.g. `<p> ... </p>`
- Empty elements must be closed
  E.g. `<br />`

CS 350: Computer/Human Interaction
Web Page Format

```html
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" lang="en" xml:lang="en">
<head>
  <title>Title of page</title>
  <meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
</head>
<body>
<!-- This is a comment. Page contents go here. -->
</body>
</html>
```

CS 350: Computer/Human Interaction
In-class Exercise 1

- Log into csserver and create directory `www_home/cgi-bin`
- Copy file `~hwang/cs350/lecture02.tar` to this directory and untar it. These files are example forms and PHP scripts.
- We will edit `index.html` which contains the HTML code on the previous slide.
CS 350: Computer/Human Interaction
Text Formatting

The following are XHTML phrase elements:

- `<p>...</p>` Defines a paragraph
- `<br />` Forces a line break
- `<hr />` Draws a horizontal line (preserves line breaks)
- `<h1>...</h1>` Largest header text
- `<h6>...</h6>` Smallest header text
- `<strong>...</strong>` Emphasis/bold
- `<em>...</em>` Emphasis/italic

CS 350: Computer/Human Interaction
Links

- Link to another page
  `<a href="http://www.google.com">Google</a>`
- Relative link to another page in the same directory
  `<a href="pizza1.html">Purple Pizza Parlor Order Form, Version 1</a>`
- Link to a different spot on the same page
  `<a href="#Refs">Jump to References</a>`

  `<h2 id="Refs">References</h2>`

CS 350: Computer/Human Interaction
Images

- Display an image in a web page
  `<img src="pizza-logo.png" width="50" height="50" alt="Pizza" />`
- Use an image as a link
  `<a href="pizza1.html"><img src="pizza-logo.png" width="50" height="50" alt="Pizza Logo" /></a>`
- Width and height attributes are optional, but it is a good practice to include them, so browsers can render the page more quickly.

CS 350: Computer/Human Interaction
Forms

- Forms are used to collect data from users
  `<form action="URL" method="post"> <!-- form elements can only appear in a form --> </form>`
  Where URL is the location of the CGI script
- Look at file `pizza1.html`

CS 350: Computer/Human Interaction
Input Element

- Input element has a type attribute
  - One line of text input
    `<input type="text" name="name" size="65" />
  - Submit button
    `<input type="submit" value="Submit" />`
- Name attribute is associated with the input value
- Value attribute is text label for button type inputs

CS 350: Computer/Human Interaction
Textarea Element

- Textarea element is used to process multi-line text input
  `<textarea cols="80" rows="3" name="address"> <!-- Default value --> </textarea>`
**CS 350: Computer/Human Interaction**  
**What is CGI?**

- CGI (Common Gateway Interface) programs run on the web server and are used to process web form data. The program’s standard output is sent to the browser.
- The web server must be configured to run CGI programs. They are not permitted on all sites.

**What is CGI?**

- Web servers usually require that CGI programs live in a particular directory and have certain filename extensions (e.g. `.cgi` or `.php`).
- On csserver CGI scripts must be in: `/home/username/www_home/cgi-bin`  
- CGI programs can be written in any language (shell, C, C++, Java, Perl, etc.) We will use only PHP.

**What is CGI?**

- When a form is posted, each element’s data is sent as encoded "name=value" pairs separated by &.  
- A CGI script receives this data as one long string through standard input.

**What is CGI?**

- CGI script must decode the input data into a usable form. Fortunately, PHP does this automatically.  
- After processing the data, a response page must be sent by writing XHTML code to standard output.

**What is PHP?**

- PHP is a portable (UNIX, Windows, Mac) scripting language with C++-like syntax with more high-level features than C++ especially designed for web programming.  
- PHP scripts are embedded within XHTML code (called code islands), which allows the static parts of a web page to be coded naturally.  
- Look at file `pizza1.php`

**Basic PHP Script**

```php
<!-- HTML code may be here -->
<?php  
// This is a comment  
// PHP code goes here  
?>
<!-- More HTML code may be here -->
```

- Output from the script replaces the script. When used as a webpage, the browser interprets it. PHP can be used stand-alone, if the CLI (Command Line Interpreter) is installed.
**CS 350: Computer/Human Interaction**

**PHP Variables**

- Seven data types: integer, float, boolean, string, array, object, and resource
  
  ```php
  $msg = "Hello, world!\n";
  $msg[0] = "H";
  print $msg;   // Hello, world!
  $age = 37;
  $data = array(1,"abc",$age);  // indexed
  print $data[2];  // 37
  $fruits = array("a"=>"apple", // assoc.
  "b"=>"banana","c"=>"cantaloupe");
  print $fruits["c"] ; // cantaloupe
  ```

**Form Values**

- Posted form input is processed and placed in the superglobal array $_POST. This is an associative array with the name attributes as keys. Generally, it is a good idea to copy the values into local variables.
  ```php
  $name = $_POST["name"];  
  $size = $_POST["size"]; 
  ```

**Here Documents**

- Here documents (heredocs) are useful for large amounts of static output. E.g., in constructing response pages.
  ```php
  print <<<EOF
  <p><strong>Name:</strong> $name</p>
  <p><strong>Address:</strong><br/>
  $addressEcho</p>
  EOF;
  ```

**More Input Element Types**

- Look at file pizza2.html
- Radio buttons
  ```html
  <input type="radio" name="size" value="Small" />Small
  <input type="radio" name="size" value="Large" checked="checked" />Large
  ```
- Check boxes
  ```html
  <input type="checkbox" name="toppings[]" value="Pepperoni" />Pepperoni
  ```

**Control Structures**

- "" (empty string), "0", 0, 0.0 are false (but "0.0" is true). Operators are as in C++. 
  ```
  
  === (identity) can be used to check for unwanted type conversions.
  ```

- There is a foreach loop
  ```php
  foreach ($parts as $word) {
    print $word;
  }
  ```
  ```php
  foreach ($fruits as $key =>$fruit)
  print "$key = $fruit";
  ```

**Functions**

- Functions as in C++ without the types. Has both value and reference parameters.
  ```php
  function foo ($arg1, & $arg2) {
  ...
  return $result
  }
  ```

- Look at file pizza2.php
**CS 350: Computer/Human Interaction**  
**String and Array Tricks**

```php
$msg = "Hello, ", $name; // concatenate
$line = "abc\ndef\nghi\nkl\nmnop";
$parts = explode("\n", $line);
// array of elements between separator
$size = count($parts); // 4
$parts[] = "qrst"; // add to end
$partlist = implode("", $parts);
// string of elements with separator
$lineEcho = nl2br($line);
// insert HTML break before newline
$input = trim($input);
// remove trailing newline
```

**CS 350: Computer/Human Interaction**  
**Select Boxes**

- Look at file `pizza3.html`, uses `pizza2.php`
- Pull-down menu
  ```html
  <select name="size">
    <option value="Large" selected="selected">Large</option>
    </select>
  ```
- Multi-selection
  ```html
  <select name="toppings[]" multiple="multiple">
    <!-- options the same as above -->
  </select>
  ```

**CS 350: Computer/Human Interaction**  
**In-class Exercise 2**

- Modify **one** of the HTML pages `pizza2.html` or `pizza3.html`, and PHP script `pizza2.php`:
  - Add an input element for a phone number
  - Add more sizes and/or toppings
  - Split the toppings into two lists: meats and veggies
  - Add pricing information: for size, for meats, for veggies
  - Compute the price of the ordered pizza
  - Output this new info in the response page

**CS 350: Computer/Human Interaction**  
**Programming Notes**

- Errors in CGI PHP code are written to standard output, so they show up in the response page along with whatever XHTML is rendered.
- PHP has implicit variable declarations, so watch out for typos. Uninitialized variables cause warnings. The function `isset($var)` can be used to determine if a variable has been initialized.