Objectives

- CSS Review
- Discussion: Google Search
- HTML Forms
- Discussion: What Can You (Legally) Take From the Web?
- Intro to Project

CSS Review

- Why CSS?
- What is the syntax of a CSS rule?
- What is the order of applying rules in the cascade?
- How can we apply a rule to a particular element?

Identify the Errors

```html
<style>
body { background-color: white
h1, { gray; font-family: sans-serif; }
h2, p { color: }
<em> { font-style: italic; }
</style>
```

Fixed

```html
<style>
body { background-color: white
h1 { color: gray; font-family: sans-serif; }
h2, p { color: blue; }
<em> { font-style: italic; }
</style>
```

Could be other ways to fix the errors

Google Search Discussion

- How does Google’s search work?
- What are some of its novel features?
- What are its limitations?
  - How can it be manipulated?
- What features would you like to add to Google’s searching technique (or other search engine)?

Google Search Discussion

- Knowing more about Google search, will you change your queries any?
  - How can you make them more effective?
  - How can you save yourself time?
- Now that you’re an author of HTML documents, will you change them to get a higher rank on Google?
  - How could you do that?
**Simplified Google PageRank**

- Developed by Larry Page and Sergei Brin
  - Then graduate students at Stanford University
- How do we know what a page is about?
  - One approach: What text do people use to link to it?
- How should we rank pages?
  - By number of links to it
  - By popularity of pages that link to it

**Approaches to Software Design**

- Inside-out
  - Develop a system
  - Add an interface
- Outside-in
  - Develop the interface
  - Then build the system to support it
- When design decisions are made, either the developer must conform to the user, or the user must conform to the developer.

**Web Interfaces**

- Menus
  - Sidebars, links, navigation
- GUIs
- Forms
  - Today’s focus

**HTML form Tag**

- Used to provide web application input from the user
- Contains various **controls or widgets** (sources of input) and labels for controls
- Must have a **submit** button that transmits all input data to server as a query string

**Examples of Forms/Input**
**HTML form Tag**

- **form attributes:**
  - `<action>` (required): URL to send request to
  - `Relative or absolute`
  - `<method>`: get or post
    - Default: Get
    - More on differences later...

**Input From Forms**

- Types of input data
  - Text, Checkbox, Radio buttons, Select from list, password

- Data is identified by a name, has a value
  - Input fields have a `name` attribute

---

**Simplified Form for Google Search**

```html
<form action="/search">
  <p>
    <input type="text" name="q" size="55"/>
    <input type="submit" value="Google Search" name="btnG"/>
  </p>
</form>
```

- Form is submitted to Google’s `/search` page with data `q` set to whatever user entered in box, e.g.,
  - `http://www.google.com/search?q=user_query`

---

**Another Way to Invoke A Web Application**

- Example: My Digital Library
  - Keeps publications, presentations

- Create a link to
  - `http://servo.cs.wlu.edu/dspace/browse-author?starts_with=M`

- Browser sends a **GET** request to the server’s `dspace/browse-author` page
  - Data’s `name` is `starts_with`, which has value `M`

---

**Query Strings**

- Do you always want your input data to show up in the URL?

---

**get vs post**

- **get** passes parameters to server as a query string
  - Limited to browser’s URL length, ~100-200 characters

- **post** embeds the parameters in **HTTP request**
  - Not in the URL

- Advantages of **post**
  - Information is more private (not shown in browser)

- Disadvantages of **post**
  - Can’t be bookmarked
  - Browser can’t easily go back (POSTDATA error)
**input Tag**
- Used to create many of the widgets
  - **type** attribute specifies the type of widget
  - Must be inside a block-level element
  - Always close tag when opened
    - Contains attributes
  - Examples:
    - text, checkbox, radio
  - Often requires **name** attribute
    - Names the data that will be sent to the Web application

**Text input**
- A horizontal box that the user can input text into
  ```html
  <form action="">
    <p>
      <input type="text" name="name" size="25"/>
    </p>
  </form>
  ```

**Attributes:**
- **size**: Width of text box in characters; scrolls if user types more
- **maxlength**: Maximum number of characters browser accepts in a box
- **value**: Provide a default value

**Password input**
- A horizontal box that the user can input text into but the text displays as *s or ●
  ```html
  <form action="">
    <p>
      <input type="password" name="mypassword" size="10" maxlength="10"/>
    </p>
  </form>
  ```

**Attributes:**
- **size**: Width of text box in characters; scrolls if user types more
- **maxlength**: Maximum number of characters browser accepts in a box

**Labeling input fields**
- Label a control with the **label** tag
  - Better than labeling with other text because
    - Can get separated during maintenance
    - **label** has special presentation
  ```html
  <p>
    <label>Password:
      <input type="password" name="mypassword" size="10"
        maxlength="10"/>
    </label>
  </p>
  ```

**Multiple Choice Input: checkbox**
- Use when user has multiple choices for something and can choose >= 1
  ```html
  <form action="">
    <p>
      <input type="checkbox" name="groceries"
        value="milk" checked="checked"/>
    </p>
  </form>
  ```

- All checkboxes in a group have the same **name** attribute
  - **value** attribute is submitted in the form data if f the checkbox is 'checked'
  - To make a checkbox checked, the **checked** attribute is set to `checked`

**Textbox.html**

- textbox.html
Multiple Choice Input: checkbox

- Discussion: When designing a form, when should a checkbox be `checked` by default?

Multiple Choice Input: radio

- Only one radio button in a group can be on or pressed
- Groups of radio buttons are identified with the same name

Menus with `<select>`

- Displays large number of options more compactly
- Can emulate radio buttons (only one selection, default) or checkboxes (multiple selections)

<table>
<thead>
<tr>
<th>name</th>
<th>Name of the data</th>
</tr>
</thead>
<tbody>
<tr>
<td>size</td>
<td># of items to display</td>
</tr>
<tr>
<td>multiple</td>
<td>Allows multiple selections if value is multiple</td>
</tr>
</tbody>
</table>

```xml
<select name="age">
  <option value="under20" selected=selected>0-19</option>
  <option value="20-35">20-35</option>
  <option value="over65">&gt;65</option>
</select>
```
select Tag Example

- Emulating checkboxes

```html
<select name="groceries" multiple="multiple">
  <option value="milk" selected="selected">Milk</option>
  <option value="bread">Bread</option>
  <option value="eggs">Eggs</option>
</select>
```

Using select

- Any advantages or disadvantages to using `select` rather than radio buttons or checkboxes?

Option Groups: optgroup

- Tag used to group options with a label
  - Can also apply a style to label

```html
<select name="entree">
  <optgroup label="Seafood">
    <option value="crabcakes">Crab Cakes</option>
  </optgroup>
  <optgroup label="Vegetarian">
    <option value="spaghetti">Spaghetti</option>
  </optgroup>
</select>
```

textarea Tag

- Allows users to input multiple lines of text

```html
<p>Please provide your yearbook memories: </p>
textarea name="memories" rows="3" cols="40" > (Be brief and concise.)
</textarea>
```

Alternative for label

- Use `for` attribute to specify which control you’re labeling
  - `for`’s value is the control’s `id`

```html
<label for="age.under18">0-17</label>
<input id="age.under18" type="radio" name="age" value="under18"/>
```
Grouping Input: fieldset, legend
- fieldset: groups related input fields
- legend: supplies an optional caption

```html
<fieldset>
<legend>Credit cards: </legend>
<input type="radio" name="creditcards" id="visa"/>
<label for="visa">Visa</label><br/>
<input type="radio" name="creditcards" id="mastercard"/>
<label for="mastercard">MasterCard</label><br/>
<input type="radio" name="creditcards" id="amex"/>
<label for="amex">American Express</label><br/>
</fieldset>
```

submit and reset Buttons
- type = submit
  - When clicked, browser sends parameters to the server
  - Browser shows server’s response
- type = reset
  - When clicked, browser changes the controls back to their original state

```html
<input type="submit" value="Submit Query"/>
<input type="reset" value="Reset"/>
```

Using Firebug
- buttons.html

Styling Forms with CSS Attribute Selectors
```html
input[type="text"] {
  color: blue;
  font-style: italic;
  margin-bottom: 2em;
}
```
- CSS attribute selector affects an element only if it has the given attribute set to the given value
- Often used with forms because input element represents many different controls

Considerations for Choosing Controls
- How easy is it for the user to enter the input?
- What is the range of possible values?
- How many possible values are there?
- How much flexibility does the user have?
  - How much should they have?
- Should the data be hidden in the browser?

What input would you use?

<table>
<thead>
<tr>
<th>Input Data</th>
<th>Input Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>Gender</td>
<td></td>
</tr>
<tr>
<td>Sensitive data</td>
<td></td>
</tr>
<tr>
<td>Date (month, day, and year)</td>
<td></td>
</tr>
<tr>
<td>Abstract for a paper</td>
<td></td>
</tr>
<tr>
<td>Title for a song</td>
<td></td>
</tr>
<tr>
<td>Household Income Bracket</td>
<td></td>
</tr>
</tbody>
</table>
What Can You (Legally) Take From the Web?

- **What** you take, **why** you take, **who** you are, what technology you use
  - Commercial or non-profit
  - Amount and substantiality of portion used
  - Nature of copyrighted work
- **What is fair use?** *Tough question!*
  - Rules change as technologies change

Time for the Computer Science...

- Project overview: Web application for use in Professor Gregory’s logic class
- Software engineering
  - Design, development, implementation in cycles
  - Feedback from client
  - Collaborative software tools

Project Overview

- Servlets, JSPs, ...
- Backend (Java)
- State
- DB or XML or text files
- Automatically generate UI (HTML), Handle user requests
- Data structures, helper classes

Project Overview

- Different types of users using the application
  - Different functionality for each
  - Accessing (at least some of) the same data
- Professors
- Administrators
- Students

Planning Stage

- Talk to client
  - Gather requirements of application
    - What is the application’s functionality
    - What must it do and what would they like it to do?
    - Where are their priorities?
    - Clarify as much as possible!
- Analyze requirements
  - Is it possible? Within the time frame?
  - If there are multiple ways to implement something, which should you do?
  - Anticipate difficulties (technology, implementation, …)
Planning Stage

- Write out requirements
  - Get rid of any ambiguities as soon as possible
  - Know all functionality
    - Required input/output
  - Clarify as much as possible
    - Otherwise, disputes with client
- Develop Work Plan
  - Steps to complete task
  - High-level, on Course Web Page
  - Drill down: divide up responsibilities

Project Overview

- Think about what needs to be clarified
  - What can the application do?
  - What do you need to know to make the UI, the backend?
  - Any hidden assumptions?
- Monday
  - Discuss your questions
  - Meet with Professor Gregory at 2:45 p.m.

Approaches to Software Design

- Inside-out
  - Develop a system
  - Add an interface
- Outside-in
  - Develop the interface
  - Then build the system to support it
- When design decisions are made, either the developer must conform to the user, or the user must conform to the developer.

Our approach to the project

Project Deliverables

- Deliverable 0: Project Requirements, Design, Work Plan
- Deliverable 1: Static HTML Mockup
  - Clarify flow, right results
  - Feedback on presentation, usability
- Deliverable 2: Web application Implementation, I
  - High-priority functionality implemented
- Deliverable 3: Web application Implementation, Final
- Deliverable 4: Documentation
  - For users and for system administrator
- Deliverable 5: Demonstration

ODO

- Lab 2: HTML Forms
  - Due Monday
- Read project description
  - Bring your questions to discuss on Monday