

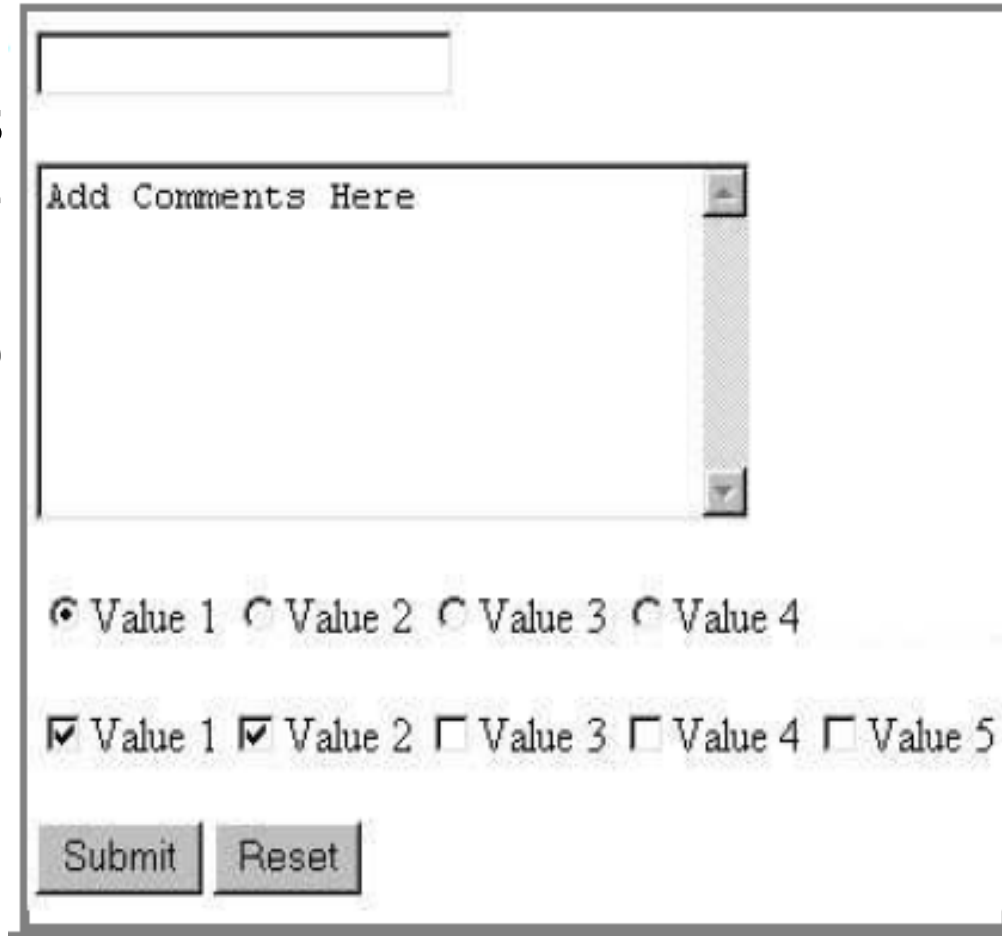
# Forms

EGR 223 - SE Approach to HCI

# HTML Forms

# HTML forms

- **form**: a group of UI controls that accepts information from the user and sends the information to a web server
- the information is sent to the server as a query string



The image shows a screenshot of a web form. At the top is a single-line text input field. Below it is a multi-line text area with the placeholder text "Add Comments Here". Under the text area are four radio buttons labeled "Value 1", "Value 2", "Value 3", and "Value 4". Below the radio buttons are five checkboxes labeled "Value 1", "Value 2", "Value 3", "Value 4", and "Value 5". At the bottom of the form are two buttons: "Submit" and "Reset".

# HTML form: <form>

```
<form action="destination URL">  
  form controls  
</form>
```

*HTML*

- required action attribute gives the URL of the page that will process this form's data
- when form has been filled out and **submitted**, its data will be sent to the action's URL

# Form example

```
<form action="http://www.google.com/search">  
  <div>  
    Let's search Google:  
    <input name="q" />  
    <input type="submit" />  
  </div>  
</form>
```

*HTML*

Let's search Google:

- Wrap the form's controls in a block element such as `div`

# Form controls: <input>

```
<!-- 'q' happens to be the name of Google's required  
parameter -->  
<input type="text" name="q" value="Colbert Report" />  
<input type="submit" value="Booyah!" />
```

HTML

- `input` element is used to create many UI controls
- `name` attribute specifies name of query parameter to pass to server

# Form controls: <input>

```
<!-- 'q' happens to be the name of Google's required  
parameter -->  
<input type="text" name="q" value="Colbert Report" />  
<input type="submit" value="Booyah!" />
```

HTML

Colbert Report

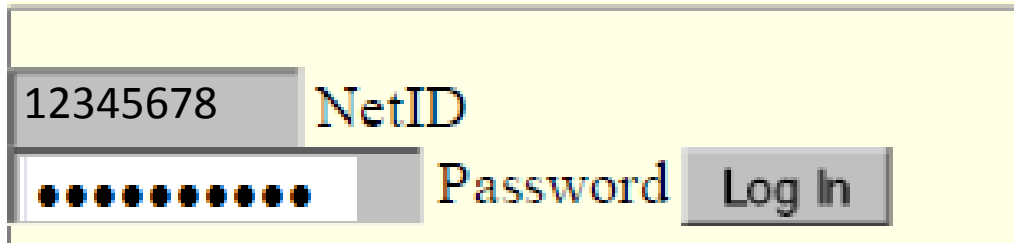
Booyah!

- `type` can be button, checkbox, file, hidden, password, radio, reset, submit, text, ...
- `value` attribute specifies control's initial text

# Text fields: `<input>`

```
<input type="text" size="10" maxlength="8" /> NetID <br />  
<input type="password" size="16" /> Password  
<input type="submit" value="Log In" />
```

HTML



- `input` attributes: `disabled`, `maxlength`, `readonly`, `size`, `value`
- `size` attribute controls onscreen width of text field
- `maxlength` limits how many characters user is able to type into field



# Form example

```
<form action="hw13.php">  
<div>  
Search for title:  
<input name="title" />  
<input type="submit" value ="Search" />  
</div>  
</form>
```

*hw13.php*

Search for title:

- Submit a form to the same page

# Form example

`https://egr223-drim-cbu.c9users.io/1_HTML_Basics/hw13.php`

- [Hello by Adele](#)
- [Speak Life by TobyMac](#)
- [All of Me by John Legend](#)

Search for title:

- Once you submit the form, the form data is appended into the URL in name/value pairs

`https://egr223-drim-cbu.c9users.io/1_HTML_Basics/hw13.php?title=Hello`

- [Hello by Adele](#)

Search for title:

# Recall Query parameters: \$\_GET

```
http://example.com/login.php?username=hello&id=1234567
```

```
$user_name = $_GET["username"];  
$id_number = (int) $_GET["id"];  
$seats_meat = FALSE;  
if (isset($_GET["meat"])) {  
    $seats_meat = TRUE;  
}
```

*PHP*

- `$_GET["parameter name"]` returns an **HTTP GET parameter's** value as a string
- parameters specified as `http://....?name=value&name=value` are **GET** parameters
- can test whether a given parameter was passed with **isset()**

# Link with Parameters

- [All of Me by John Legend](#)
- [Hello by Adele](#)
- [Speak Life by TobyMac](#)

Search for title:

[Sort by song title](#)

```
<a href="hw13.php?sort=TRUE">Sort by song title</a>
```

*PHP*

```
https://egr223-drim-cbu.c9users.io/1 HTML Basics/hw13.php?sort=TRUE
```

*URL*

# HW13 due before next class

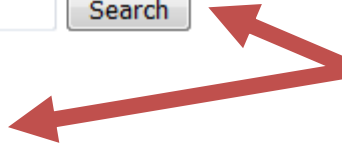
- Extend your HW12 to have the following in **blog.php**
  - An input form to search for a title
  - A link to sort by titles alphabetically, and unsorted (default)

[https://php-mysql-drim-cbu.c9users.io/1\\_HTML/blog.php](https://php-mysql-drim-cbu.c9users.io/1_HTML/blog.php)

- [All of Me by John Legend](#)
- [Hello by Adele](#)
- [Speak Life by TobyMac](#)

Search for title:

[Sort by song title](#)   [Unsorted](#)



- Use PHP to create a single function called  
`function printSongs($searchTitle="", $doSort="FALSE")`
- And depending on the user event, display the songs in sorted list or display the title the user searched for.

# Expected Output

## My Blog

*Because I need to share my thoughts to random people*

[About Me](#)  
[My Blog](#)  
[Favorite Music](#)  
[Verse of the Day](#)

### 2/24/ - Music Musik Musico Musica Musickcoca

My faviorite music:

Search for title:

Sort by song title    [Unsorted](#)

- i. [Hello by Adele](#)
- ii. [Speak Life by TobyMac](#)
- iii. [All of Me by John Legend](#)
- iv. [Oceans by Hillsong](#)

### 2/4 - Burgers...mmm

Today, I have a sudden craving for some good burgers. What do you say: In-N-Out vs. The Habit?



2/24

- Music Musik Musico Musica Musickcoca

My faviorite music:

Search for title:

[Sort by song title](#)

[Unsorted](#)

i. [Hello by Adele](#)

- Search by song title and show only the matched song.
- If user searches for empty string, show all songs.

2/24,

- Music Musik

My faviorite music:

Search for title:

[Sort by song title](#)

[Unsorted](#)

**My list in sorted order:**

- [All of Me by John Legend](#)
- [Hello by Adele](#)
- [Oceans by Hillsong](#)
- [Speak Life by TobyMac](#)

- Sorted =  
sort by  
song title

2/24

- Music Musik M

My faviorite music:

Search for title:

[Sort by song title](#)

[Unsorted](#)

- [Hello by Adele](#)
- [Speak Life by TobyMac](#)
- [All of Me by John Legend](#)
- [Oceans by Hillsong](#)

- Unsorted =  
as it  
appears in  
music.txt