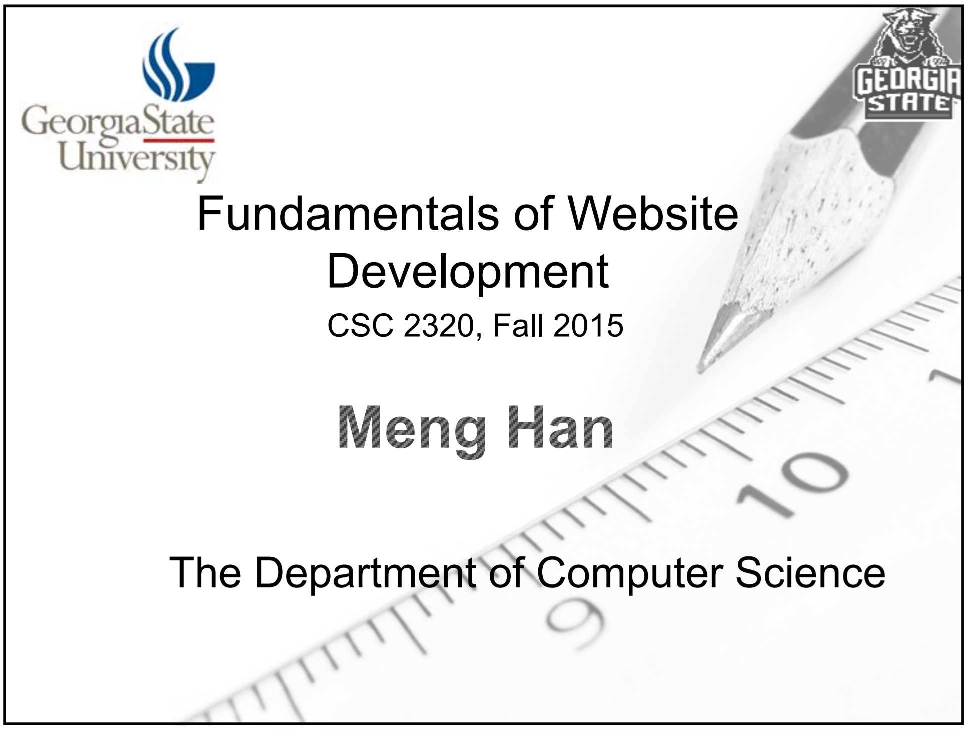
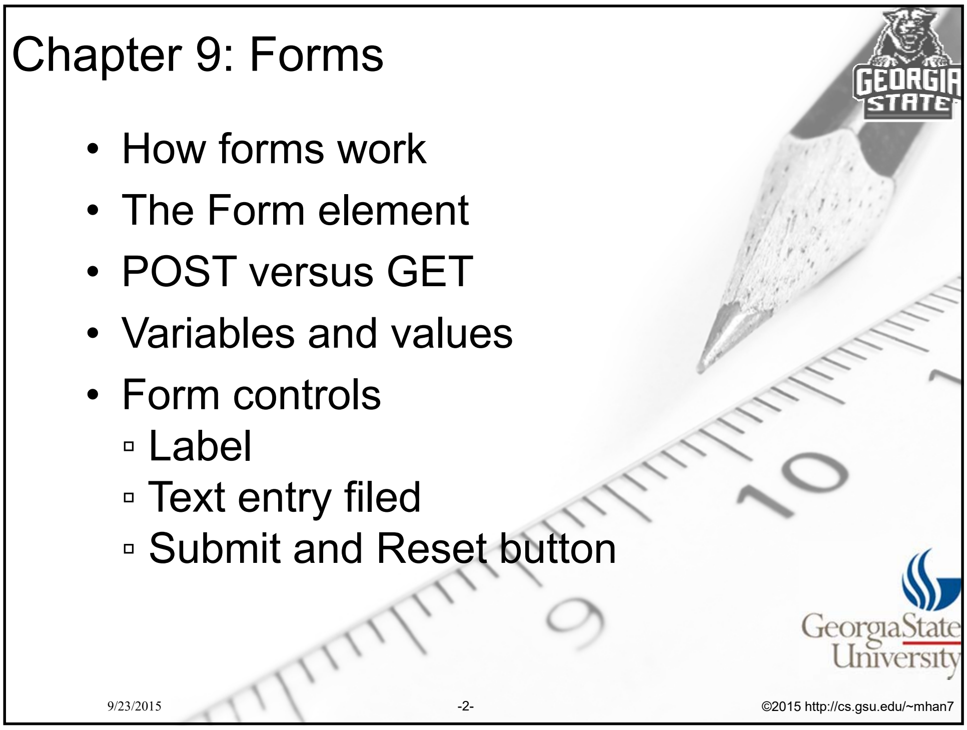


# Fundamentals of Website Development

CSC 2320, Fall 2015

**Meng Han**

The Department of Computer Science



## Chapter 9: Forms

- How forms work
- The Form element
- POST versus GET
- Variables and values
- Form controls
  - Label
  - Text entry filed
  - Submit and Reset button

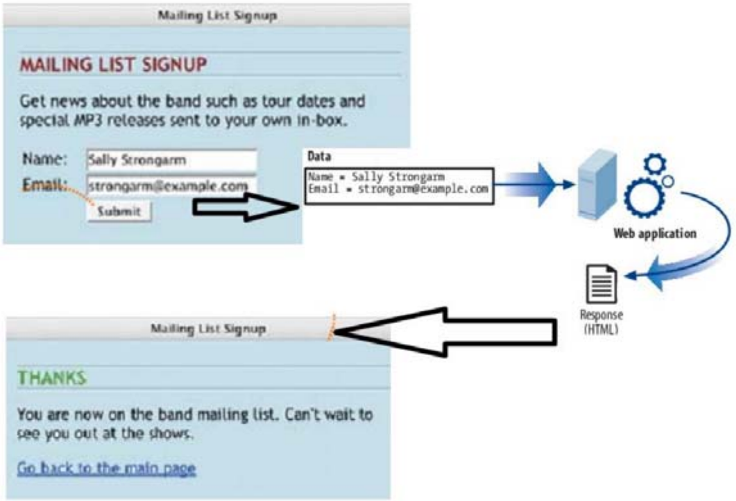
9/23/2015 -2- ©2015 http://cs.gsu.edu/~mhan7

# How Forms Work

- **Two parts:**
  - Part A: the visible form on the web page
  - Part B: an application or script on the server that PROCESSES the information submitted by the form and RETURNS appropriate response.

- Web applications and scripts make the forms work

# How Forms Work ( Example )



Mailing List Signup

MAILING LIST SIGNUP

Get news about the band such as tour dates and special MP3 releases sent to your own in-box.

Name: Sally Strongarm

Email: strongarm@example.com

Submit

Data

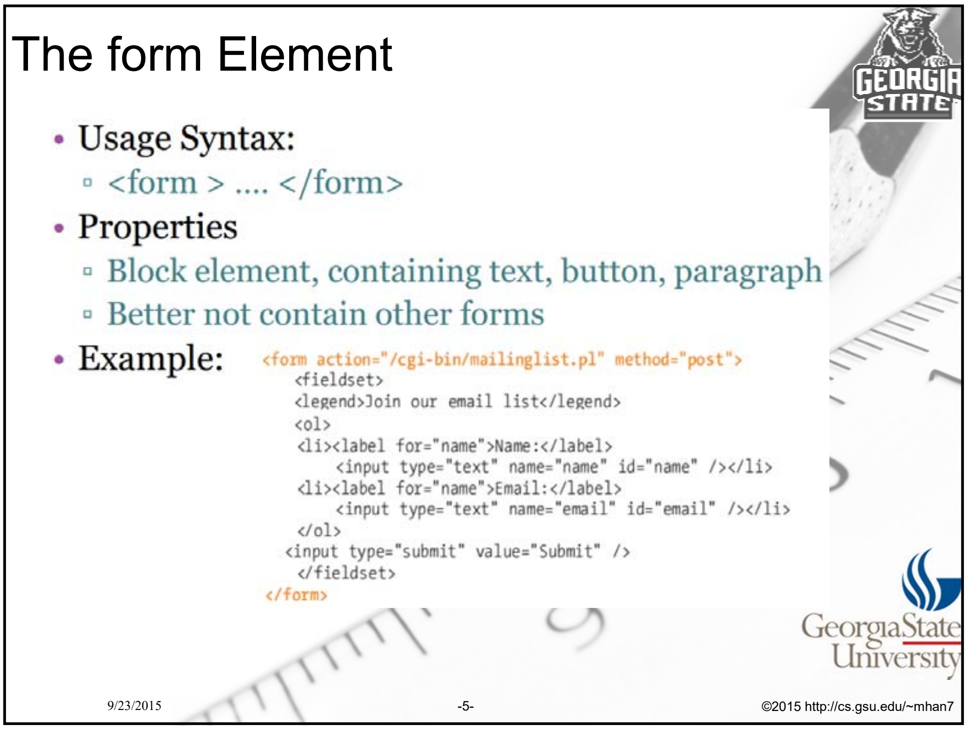
Name = Sally Strongarm
Email = strongarm@example.com

Mailing List Signup

THANKS

You are now on the band mailing list. Can't wait to see you out at the shows.

Go back to the main page

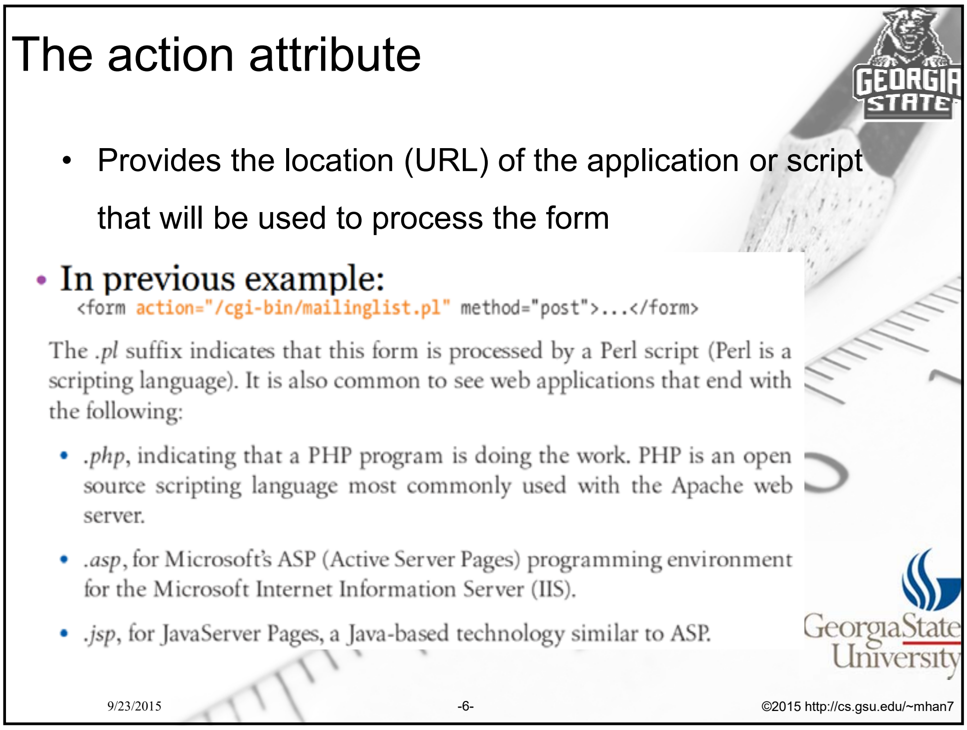


# The form Element

- Usage Syntax:
  - <form > .... </form>
- Properties
  - Block element, containing text, button, paragraph
  - Better not contain other forms
- Example:

```
<form action="/cgi-bin/mailinglist.pl" method="post">
    <fieldset>
    <legend>Join our email list</legend>
    <ol>
    <li><label for="name">Name:</label>
        <input type="text" name="name" id="name" /></li>
    <li><label for="name">Email:</label>
        <input type="text" name="email" id="email" /></li>
    </ol>
   <input type="submit" value="Submit" />
    </fieldset>
</form>
```

9/23/2015 -5- ©2015 http://cs.gsu.edu/~mhan7



# The action attribute

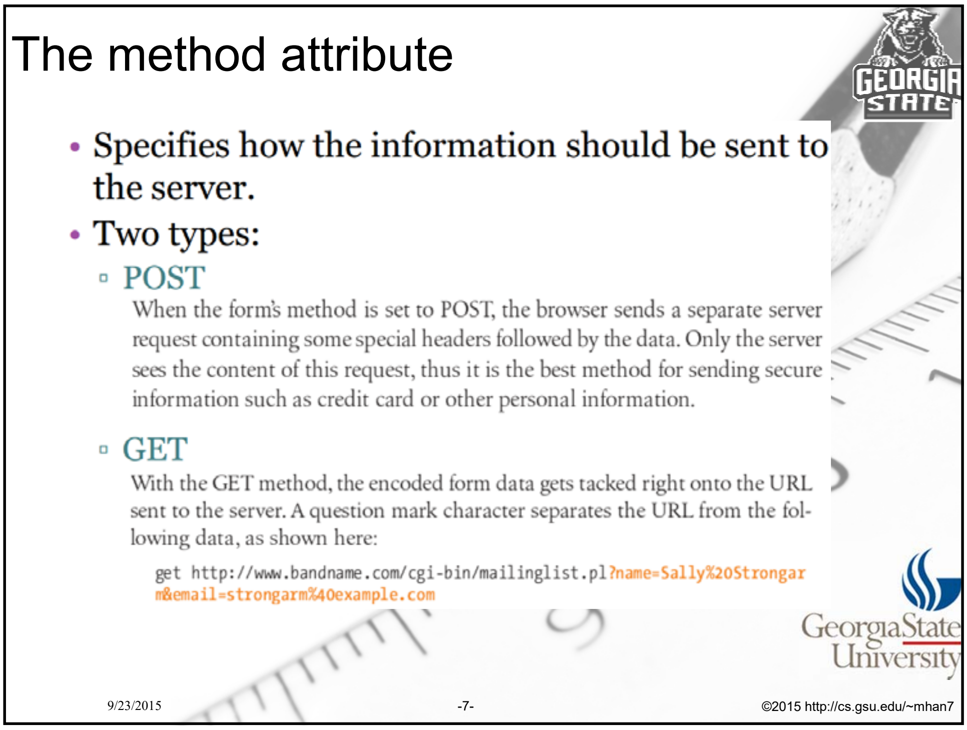
- Provides the location (URL) of the application or script that will be used to process the form
- In previous example:

```
<form action="/cgi-bin/mailinglist.pl" method="post">...</form>
```

The *.pl* suffix indicates that this form is processed by a Perl script (Perl is a scripting language). It is also common to see web applications that end with the following:

- *.php*, indicating that a PHP program is doing the work. PHP is an open source scripting language most commonly used with the Apache web server.
- *.asp*, for Microsoft's ASP (Active Server Pages) programming environment for the Microsoft Internet Information Server (IIS).
- *.jsp*, for JavaServer Pages, a Java-based technology similar to ASP.

9/23/2015 -6- ©2015 http://cs.gsu.edu/~mhan7



## The method attribute

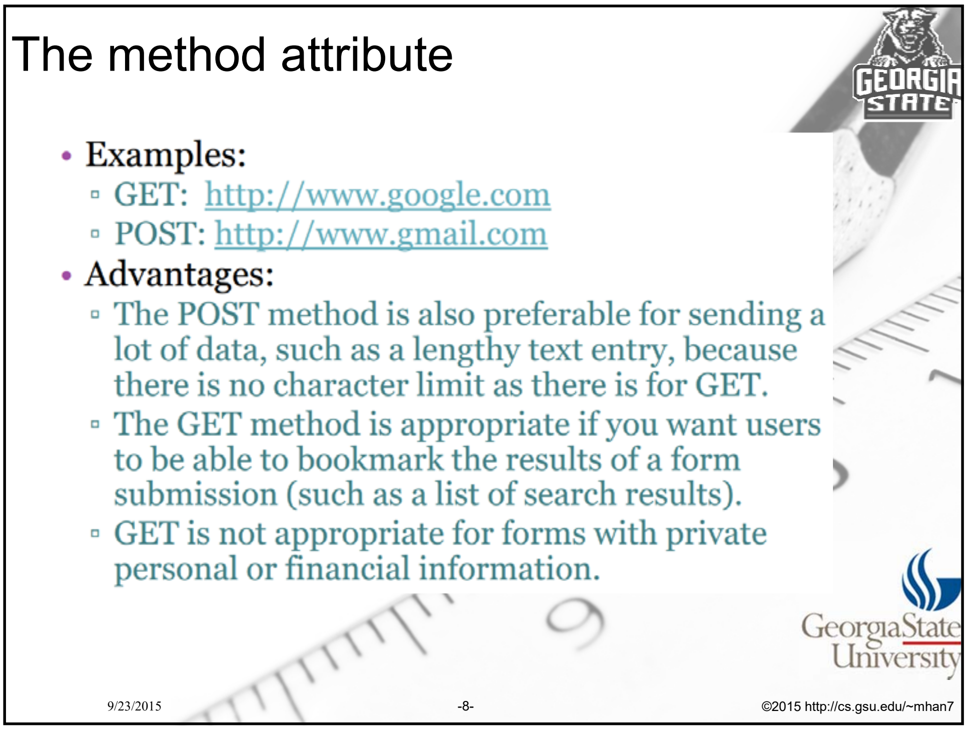
- Specifies how the information should be sent to the server.
- Two types:
  - POST

    When the form's method is set to POST, the browser sends a separate server request containing some special headers followed by the data. Only the server sees the content of this request, thus it is the best method for sending secure information such as credit card or other personal information.
  - GET

    With the GET method, the encoded form data gets tacked right onto the URL sent to the server. A question mark character separates the URL from the following data, as shown here:

```
get http://www.bandname.com/cgi-bin/mailinglist.pl?name=Sally%20Strongar
m&email=strongarm%40example.com
```

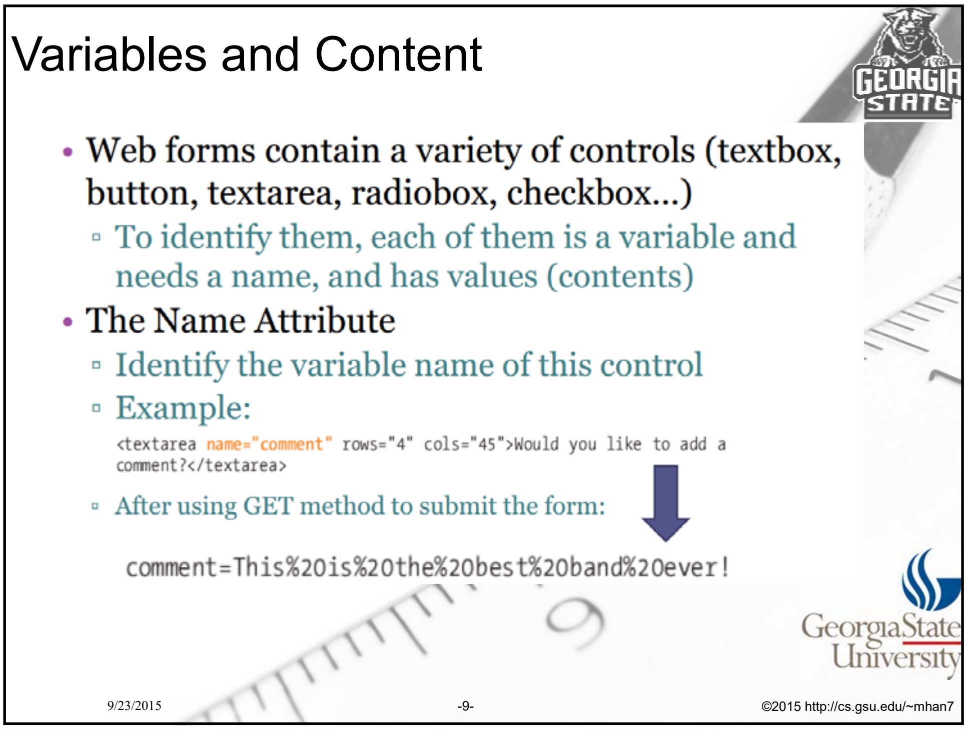
9/23/2015 -7- ©2015 http://cs.gsu.edu/~mhan7



## The method attribute

- Examples:
  - GET: http://www.google.com
  - POST: http://www.gmail.com
- Advantages:
  - The POST method is also preferable for sending a lot of data, such as a lengthy text entry, because there is no character limit as there is for GET.
  - The GET method is appropriate if you want users to be able to bookmark the results of a form submission (such as a list of search results).
  - GET is not appropriate for forms with private personal or financial information.

9/23/2015 -8- ©2015 http://cs.gsu.edu/~mhan7



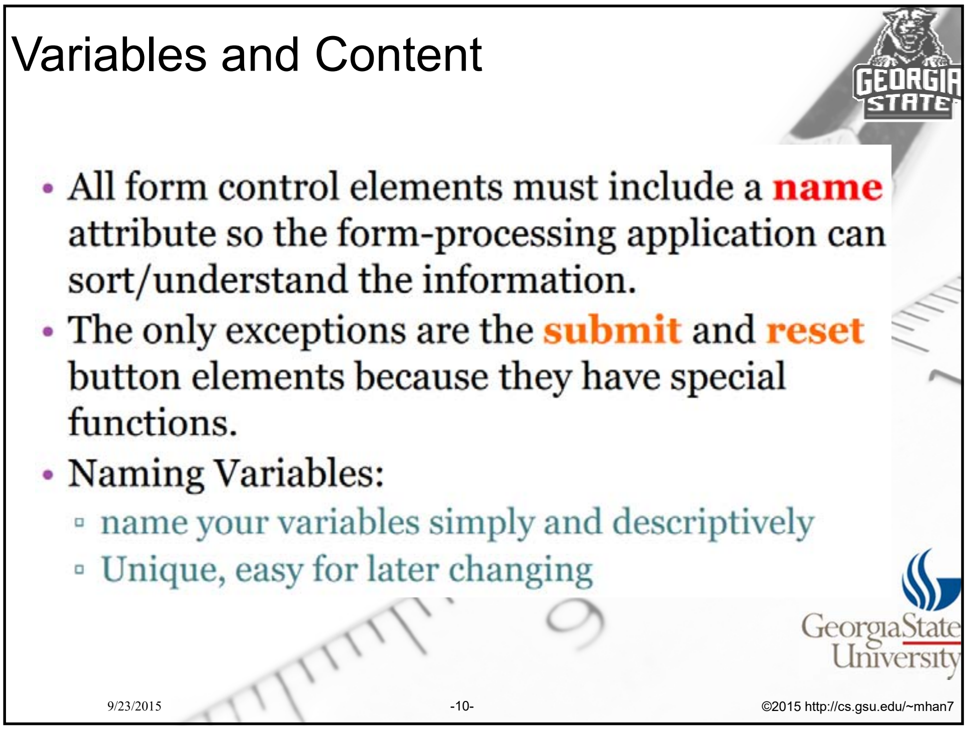
## Variables and Content

- Web forms contain a variety of controls (textbox, button, textarea, radiobox, checkbox…)
  - To identify them, each of them is a variable and needs a name, and has values (contents)
- The Name Attribute
  - Identify the variable name of this control
  - Example:

```
<textarea name="comment" rows="4" cols="45">Would you like to add a
comment?</textarea>
```

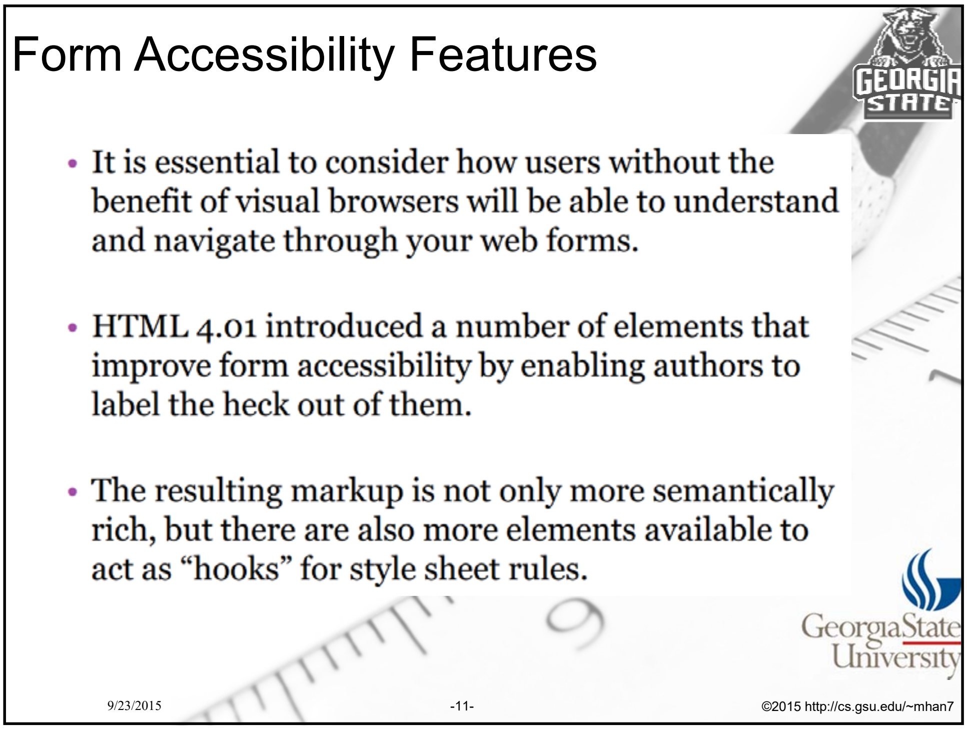
  - After using GET method to submit the form:

```
comment=This%20is%20the%20best%20band%20ever!
```



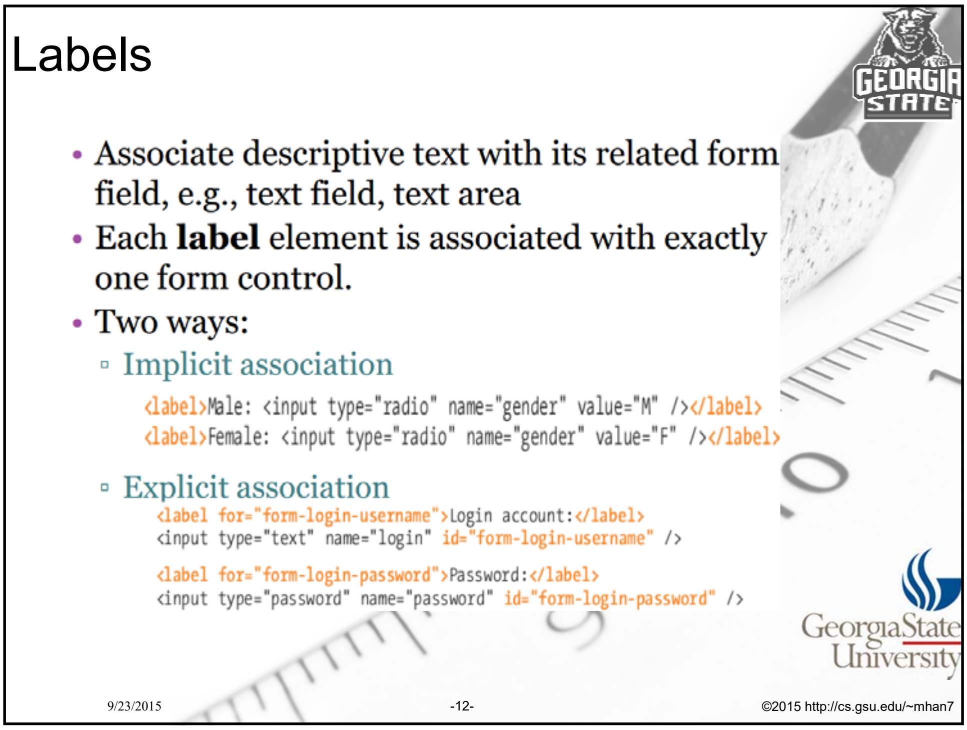
## Variables and Content

- All form control elements must include a **name** attribute so the form-processing application can sort/understand the information.
- The only exceptions are the **submit** and **reset** button elements because they have special functions.
- Naming Variables:
  - name your variables simply and descriptively
  - Unique, easy for later changing



## Form Accessibility Features

- It is essential to consider how users without the benefit of visual browsers will be able to understand and navigate through your web forms.
- HTML 4.01 introduced a number of elements that improve form accessibility by enabling authors to label the heck out of them.
- The resulting markup is not only more semantically rich, but there are also more elements available to act as “hooks” for style sheet rules.



## Labels

- Associate descriptive text with its related form field, e.g., text field, text area
- Each **label** element is associated with exactly one form control.
- Two ways:
  - Implicit association

```
<label>Male: <input type="radio" name="gender" value="M" /></label>
<label>Female: <input type="radio" name="gender" value="F" /></label>
```

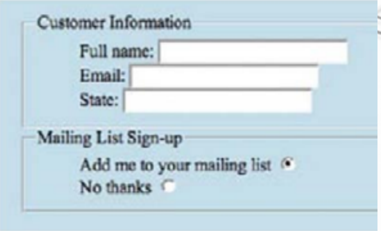
  - Explicit association

```
<label for="form-login-username">Login account:</label>
<input type="text" name="login" id="form-login-username" />

<label for="form-login-password">Password:</label>
<input type="password" name="password" id="form-login-password" />
```

# Fieldset and Legend

- The fieldset element is used to indicate a logical group of form controls.
- A fieldset may also include a **legend** element that provides a caption for the enclosed fields.

```
<fieldset>
<legend>Customer Information</legend>
 <ol>
   <li><label>Full name: <input type="text" name="name" /></l
   <li><label>Email: <input type="text" name="email" /></labe
   <li><label>State: <input type="text" name="state" /></labe
 </ol>
</fieldset>

<fieldset>
<legend>Mailing List Sign-up</legend>
<ul>
  <li><label>Add me to your mailing list <input type="radio"
name="list" value="yes" checked="checked" /></label></li>
  <li><label>No thanks <input type="radio" name="list" value=
</label></li>
</ul>
</fieldset>
```



Customer Information

Full name:

Email:

State:

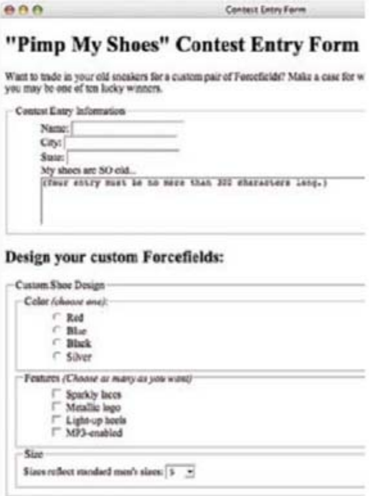
Mailing List Sign-up

Add me to your mailing list ●

No thanks ○

# The Great Form Control Round-up

- Text entry controls
- Submit and reset buttons
- Radio and checkbox buttons
- Pull-down and scrolling menus
- File selection and upload control
- Hidden controls



"Pimp My Shoes" Contest Entry Form

Want to trade in your old sneakers for a custom pair of Forcefields? Make a case for w you may be one of ten lucky winners.

Contest Entry Information

Name:

City:

State:

My shoes are SO old...

(Your entry must be no more than 300 characters long.)

Design your custom Forcefields:

Custom Shoe Design

Color (choose one):

○ Red

○ Blue

○ Black

○ Silver

Features (Choose as many as you want)

□ Sparkly laces

□ Metallic logo

□ Light-up heels

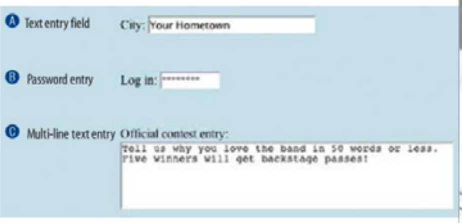
□ MP3-enabled

Size

Sizes reflect standard men's sizes: 5

# Text entry controls

- Three categories:
  - Single-line text field
  - Password entry field
  - Multiline text field



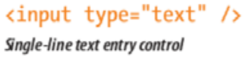
A Text entry field City: Your Hometown

B Password entry Log in: ********

C Multi-line text entry Official contest entry:

Tell us why you love the band in 50 words or less.
Five winners will get backstage passes!

- Single-line text field

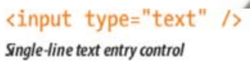


```
<input type="text" />
```

*Single-line text entry control*

- Attributes:
  - Name: required for identifying variable name
  - Value: specifies default text
  - Size: width of text filed when displaying
  - Maxlength: max number of characters can typed in this flield

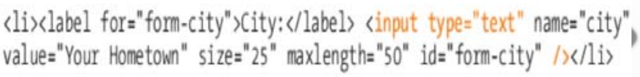
# Single-line text field



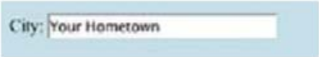
```
<input type="text" />
```

*Single-line text entry control*

- Attributes:
  - Name: required for identifying variable name
  - ID: used to association for label or identify itself
  - Value: specifies default text
  - Size: width of text filed when displaying
  - Maxlength: max number of characters can typed in this field
- Example:



```
<li><label for="form-city">City:</label> <input type="text" name="city"
value="Your Hometown" size="25" maxlength="50" id="form-city" /></li>
```



City: Your Hometown

# Password text entry field

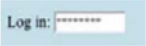
- A password field works just like a text entry field, except the characters are obscured from view using asterisk (*) or bullet (·) characters, or another character determined by the browser.
- Basic usage:

```
<input type="password" />
```

*Password text control*

- Attributes: same as single-line text field
  - Name, ID, Value, Size, Maxlength
- Example:

```
<li><label for="form-pswd">Log in:</label> <input type="password"
name="pswd" size="8" maxlength="8" id="form-pswd" /></li>
```



Log in: ********

# Multiline text entry field

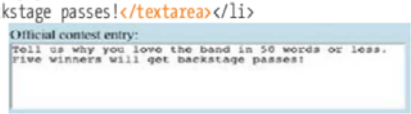
- Basic Usage:

```
<textarea>...</textarea>
```

*Multiline text entry control*

- Opening tag and Closing tag both used
- Attributes:
  - Name
  - Rows: number of displaying lines (scrollbars)
  - Cols: width of text area in number of chars
- Example:

```
<li><label for="form-entry">Official contest entry:</label><br />
<textarea name="contest_entry" rows="5" cols="100" id="form-entry">Tell
us why you love the band in 50 words or less. Five winners will get
backstage passes!</textarea></li>
```



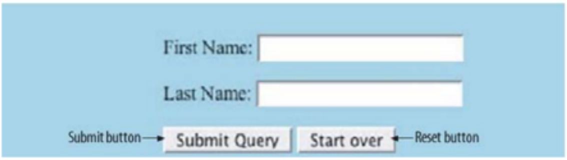
Official contest entry:

Tell us why you love the band in 50 words or less.
Five winners will get backstage passes!

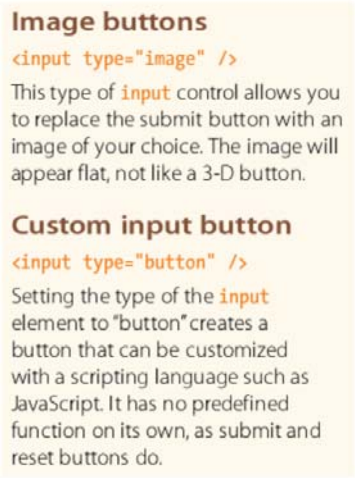
# Submit and reset buttons

- Submit button: When clicked, the submit button immediately sends the collected form data to the server for processing.
- Reset button: returns the form controls to the state they were in when the form loaded.
- Both submit and reset buttons are added using the **input** element.
- Attribute: **value**, defines the text on button
- Example:

```
<p><input type="submit" /> <input type="reset" value="Start over" /></p>
```



# More buttons …



### Image buttons

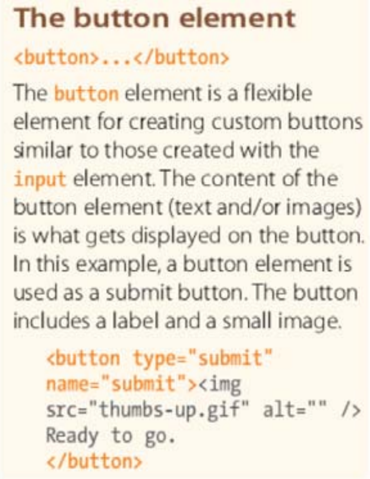
`<input type="image" />`

This type of input control allows you to replace the submit button with an image of your choice. The image will appear flat, not like a 3-D button.

### Custom input button

`<input type="button" />`

Setting the type of the input element to "button" creates a button that can be customized with a scripting language such as JavaScript. It has no predefined function on its own, as submit and reset buttons do.

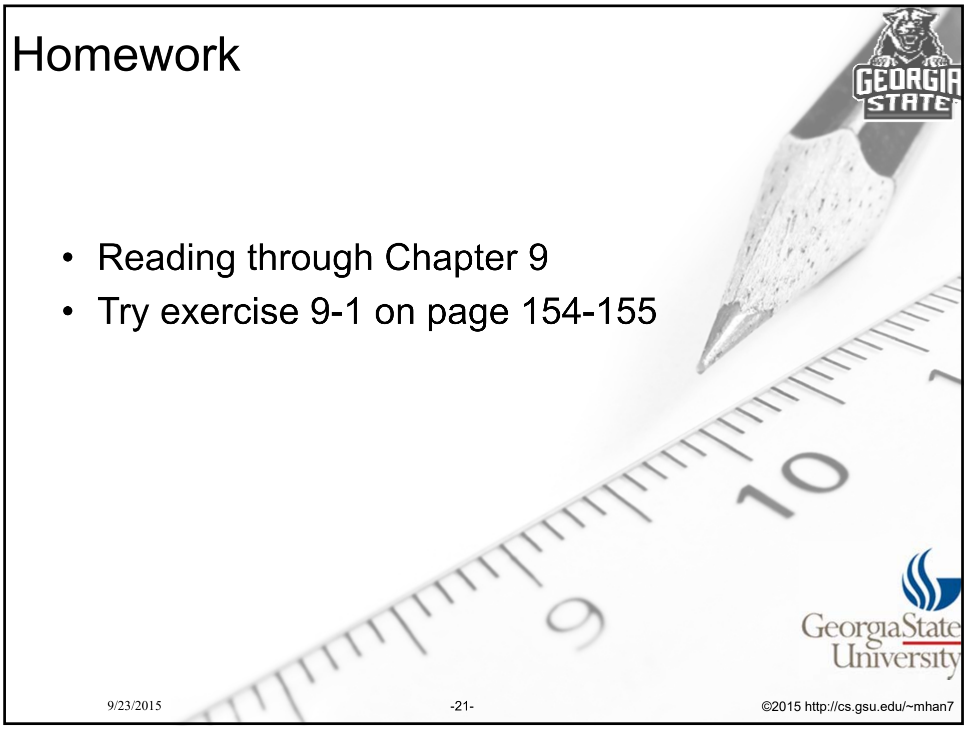


### The button element

`<button>...</button>`

The button element is a flexible element for creating custom buttons similar to those created with the input element. The content of the button element (text and/or images) is what gets displayed on the button. In this example, a button element is used as a submit button. The button includes a label and a small image.

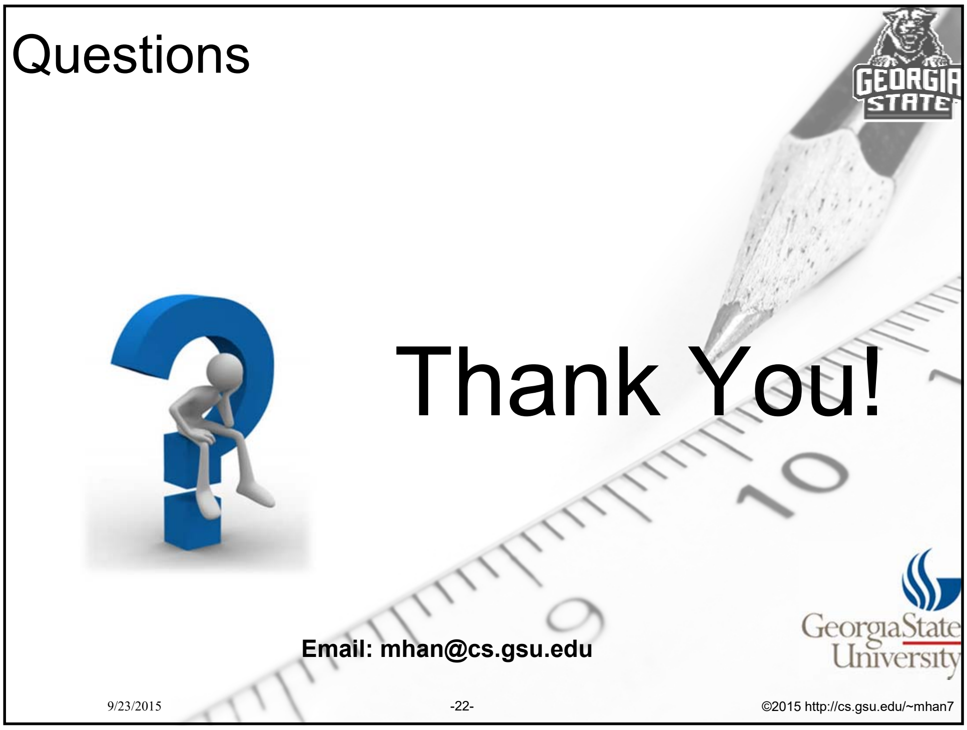
```
<button type="submit"
name="submit"><img
src="thumbs-up.gif" alt="" />
Ready to go.
</button>
```



## Homework

- Reading through Chapter 9
- Try exercise 9-1 on page 154-155

9/23/2015 -21- ©2015 http://cs.gsu.edu/~mhan7



## Questions

Thank You!

**Email: mhan@cs.gsu.edu**

9/23/2015 -22- ©2015 http://cs.gsu.edu/~mhan7