

# #C1435



## Lecture 9

# HTML5 and CSS forms

# This lecture will cover...

- Role of forms on websites
- HTML form elements
  - The **input** element and input **types**
  - Form controls
  - HTML5 form validation
  - Other form elements
- Styling forms with CSS
  - Responsive forms
- Reading and resources

# Why forms are important

***Essential*** for web functionality and interactivity

- Converting website browsers into customers and contributors
  - Engaging users – registering, contributing content
- Drivers for web application workflow
  - Collecting data from users
- Examples –
  - **e-commerce** – shopping cart, orders and payments
  - **e-gov** – vehicle licensing, student finance
  - **blogs** – comments, content management



**Book It**

### Sign Up

First Name

Last Name

Email Address

Password

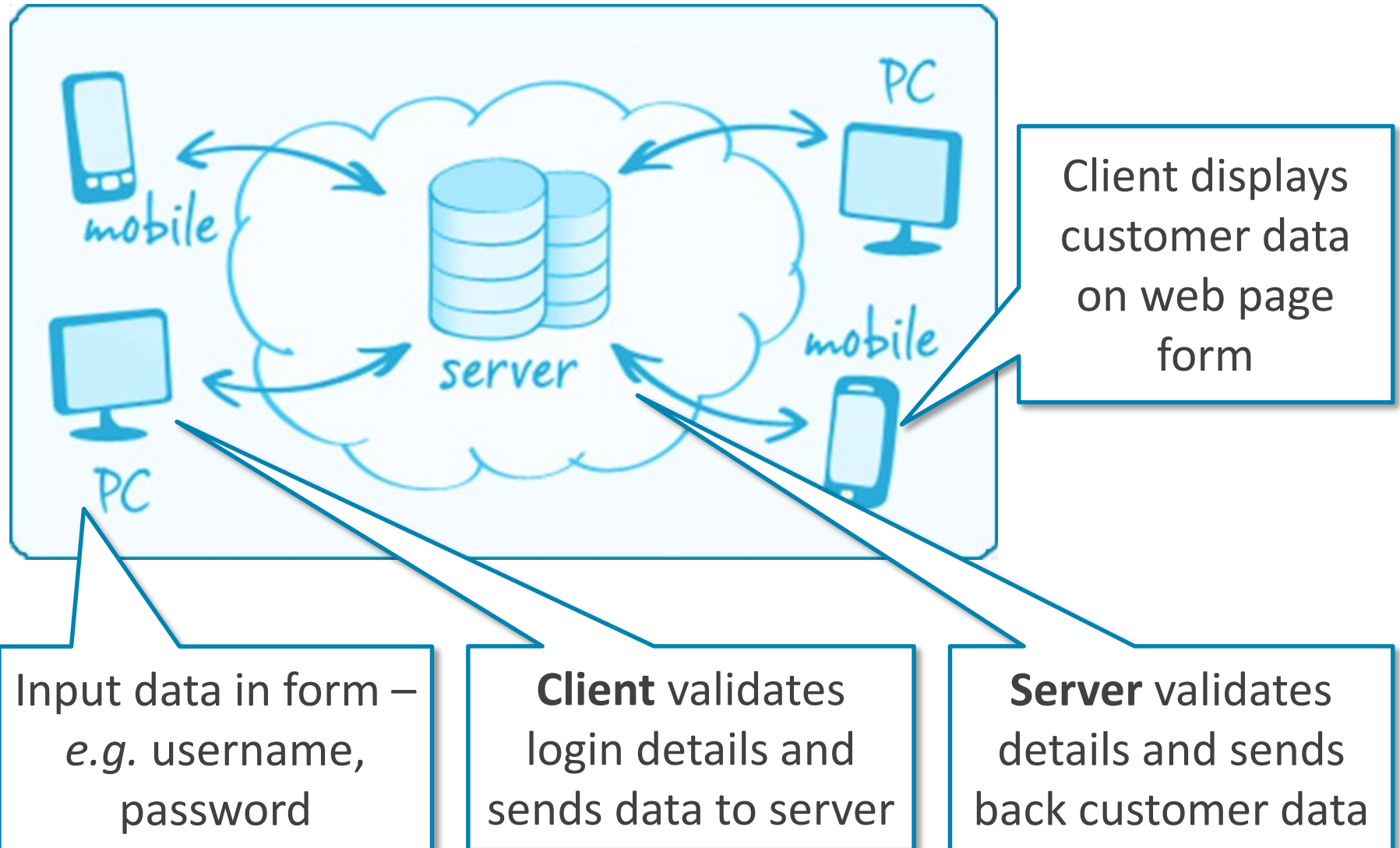
Confirm Password

**Submit**

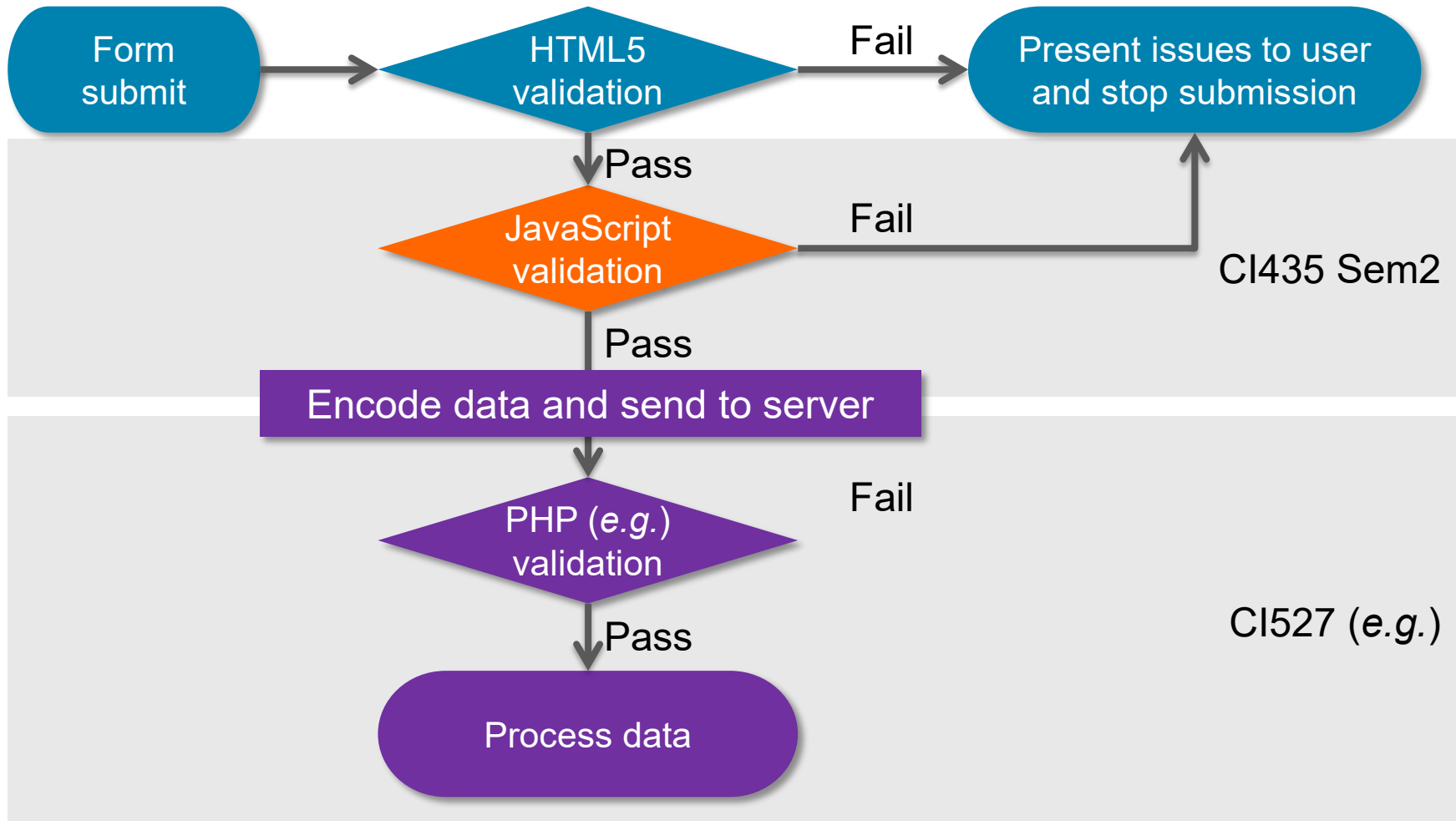
# Making forms interactive

- Scripting languages are needed to make forms functional – *i.e.* to exchange data between the users (front end **client** computer) and the website owner (back end **server** computer)
  - **JavaScript** – client side scripts that can check accuracy of data entered in the form before it is sent to the server
  - Server side scripts such as **PHP**, **ASP.NET** process the data provided by the user, check it and send the appropriate response back to the user
- **Semester 2** – covers JavaScript and client side data processing
- **Lab tutorial 9** – developing an HTML and CSS form, but without any scripting or functionality

# Client server application architecture



# Form processing



# Designing usable forms

- Forms are interactive and functional – so it's very important that they provide a good user experience
- But often badly designed, not usable and inaccessible
- Worse - often used to 'trick' the user – *e.g.* by having to *deselect* a pre-ticked checkbox to opt *out* of spam email
- Forms should –
  - be laid out clearly and logically
  - give clear guidance and **feedback** to users
  - have consistent design throughout a website
  - use the correct semantic HTML form elements



# HTML <form> element

- All form elements are nested inside the **<form>** element –  
    <body>  
        <form action="/processing-form-page" method="post">  
            </form>  
    </body>
- The **action attribute** is required: it is used to specify the Uniform Resource Indicator (URL) of a web page that processes information submitted via the form
- The **method attribute** specifies the HTTP method for submitting form data to the server – it can be 'get' or 'post'
- In your form the **values** for action and method can be empty

# HTML `<input>` element

- The HTML **`<input>`** element is used for the interactive controls through which users can enter data into the form
- Input elements have a **`type`** attribute which is used to specify the type of **control** ('widget') to display and the type of **data** that can be input through the control, *e.g.*—  
button, checkbox, radio, password, reset
- Input types are associated with specific **controls** – the means through which the data is input
- For full list of HTML5 `<input>` element types see the excellent Mozilla Developer Network **HTML Forms Guide** - <https://developer.mozilla.org/en-US/docs/Web/HTML/Element/input>

# HTML `<input>` types - examples

Input type keyword	Data type	Control type
text	Text with no line breaks	Text field
password	Text with no line breaks (sensitive information)	Text field that obscures data entered
email	Email address	Text field
date	A date (year, month, day)	Date control
checkbox	Set of zero or more values from a predefined list	Checkbox
radio	An enumerated value	Radio button
submit	An enumerated value that initiates form submission	Button
button	N/A	Button

# <input> types – text

**Text - input type** for text data

```
<form>
```

```
  <label for="name">Name:</label>
```

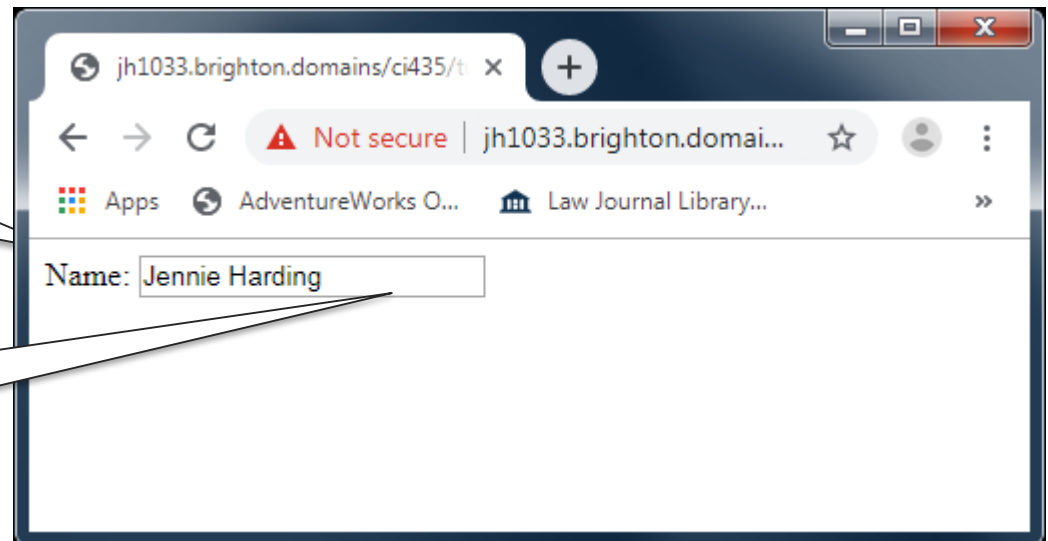
```
  <input type="text" id="name" />
```

```
</form>
```

Self-closing tag

Label for the  
<input> element

The **control** is a  
text field



# <input> types – checkbox

- **Checkboxes** are used to select zero *or more* options from a predefined list - *i.e.* set zero or more to 'true'

<p>What pets do you have? (tick all that apply)</p>

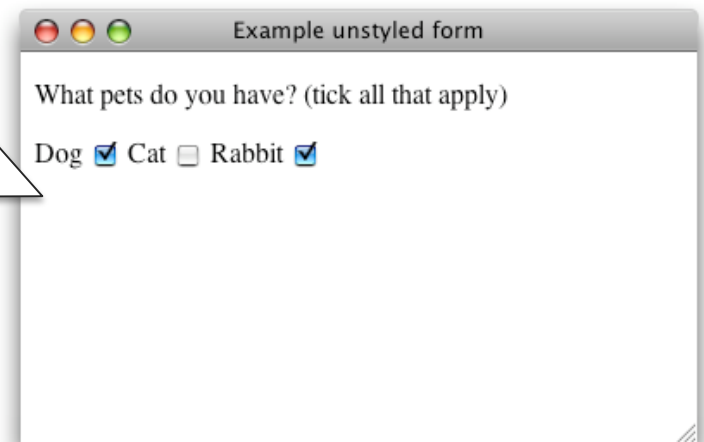
<label for="dog">Dog</label>

<input type="checkbox" id="dog" />

<label for="cat">Cat</label>

<input type="checkbox" id="cat" />

<input> is an **inline** element.  
Don't use <p> <br /> to put elements on separate lines - declare **input {display: block;}** in the CSS stylesheet



Example unstyled form

What pets do you have? (tick all that apply)

Dog ☒ Cat ☐ Rabbit ☒

# <input> types – radio button

- A group of **radio buttons** is used when *only one option* in a list can be checked - *i.e.* set one option to 'true'

```
<p>Do you like dogs or cats?</p>
```

```
  <label>Dogs<input type="radio" name="likes"  
  value="dog" checked />  
</label>
```

```
  <label>Cats<input type="radio" name="likes"  
  value="cat"/></label>
```

- The **name** attribute links buttons in a group and enforces the single option selection
- The **checked** attribute pre-selects a button

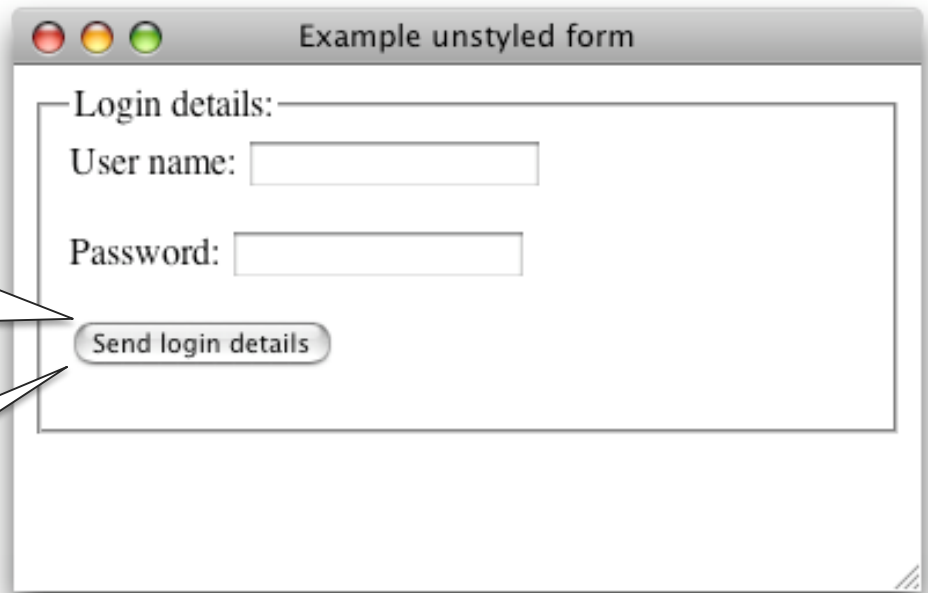
# <input> types – submit

- The submit input type has a **button** control: when activated it sends the form data to the web server, using the action and method specified in the <form> element attributes

```
<input type="submit" id="send"  
value="Send login details" />
```

Submit text should be short, but must provide a clear '**call to action**' for the user

Submit control styled by browser



The screenshot shows a web browser window with the title "Example unstyled form". Inside the window is a form titled "Login details:". The form contains two input fields: "User name:" and "Password:". Below these fields is a submit button with the text "Send login details". The button has a light gray background and a thin border, which is the default styling provided by the browser. Two callout boxes from the text on the left point to the button: one points to the text "Send login details" and the other points to the button's border.

# <input> types – button, reset button

- The **button** input type is a button with *no default behaviour*. It has to invoke a JavaScript function when activated to make it behave in a specific way – *e.g.* to display a message when clicked (semester 2)

```
<input type="button" id="send" value="Send details"/>
```

- The **reset** input type is a button that when activated resets the form to its default state – *i.e.* returns all the values in the input elements to how they were before the user entered any data

```
<input type="reset" id="reset" value="Clear the form"/>
```



# HTML5 <input> types

HTML5 added **13** form input types with new controls *e.g.* –

- **date** – control is a **calendar**
- **color** – control is a **colour picker**
- **number** – control is a **spinner** for entering a number, can also define 'min' and 'max' values as attributes
- **range** – slider for entering a number in a range (precise value may not be required)
- Read this article to find out more -

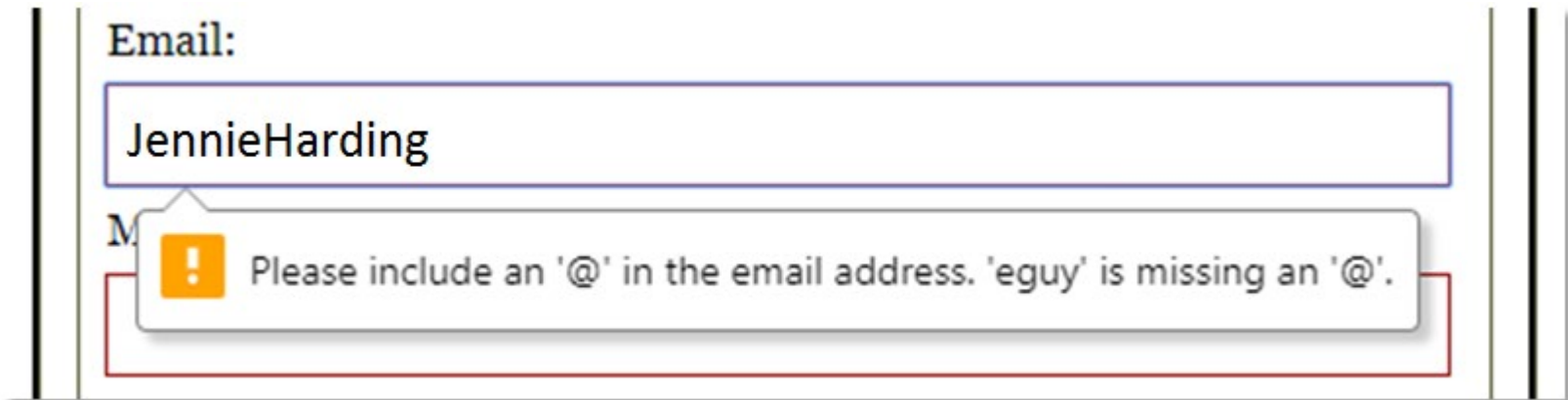
<http://html5doctor.com/html5-forms-input-types/>

# HTML5 <input> types

- Some of the HTML5 controls are not supported in the latest browsers; refer to <https://caniuse.com/#feat=forms>
- My example form should be viewed in **Chrome**, which gives the best support, to see the new controls -  
<http://jh1033.brighton.domains/ci435/tutorials/tutorial09/contact.html>
- The browser fallback when an input type control is not supported is to display a text input type - this still allows data to be entered

# Native HTML5 form validation

- Input element types are associated with a specific data type – so browsers 'know' what data values are valid
- Native - *i.e.* in the browser - form validation is increasingly a viable alternative to client side validation using JavaScript
- Browser can check an email address entered in the **email input type** text field against a built-in pattern of what an email should include. Displays an error message if it is invalid.



The screenshot shows a web form with a label "Email:" and a text input field containing "JennieHarding". Below the input field, a red border highlights the area where an error message is displayed. The error message is contained within a white box with a grey border and a shadow. It features a red square icon with a white exclamation mark on the left, followed by the text: "Please include an '@' in the email address. 'eguy' is missing an '@'." The input field has a blue border, and the error message box has a red border on its left and bottom sides.

Email:

JennieHarding

M

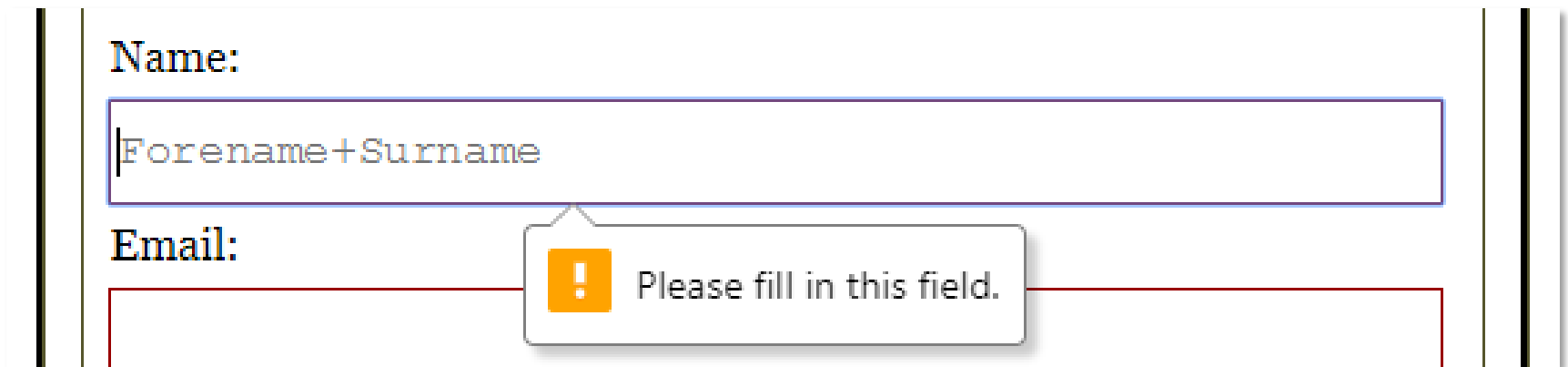
! Please include an '@' in the email address. 'eguy' is missing an '@'.

# Validation – 'required' attribute

- For form fields that **must be completed** - use the Boolean (*i.e.* True or False) attribute **'required'**

```
<label for="name">Name: </label>
```

```
<input id="name" type="text" required ...>
```

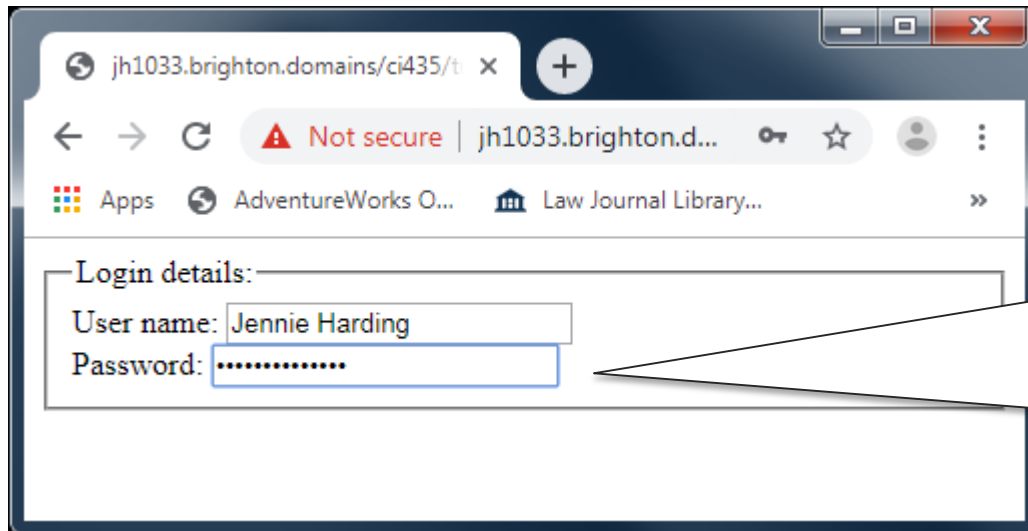


The screenshot shows a web form with two input fields. The first field is labeled 'Name:' and contains the placeholder text 'Forename+Surname'. The second field is labeled 'Email:'. A red border highlights the 'Email:' label and its corresponding input field. A yellow warning icon with an exclamation mark is positioned above the 'Email:' field, with a tooltip message that reads 'Please fill in this field.'.

Tested in **Chrome** – the error message is the browser default

# Example – HTML login form

```
<form action="" method="post" id="contact">
  <fieldset id="login">
    <legend>Login details:</legend>
    <label for="username">User name: </label>
    <input type="text" id="name" required />
    <label for="password">Password: </label>
    <input type="password" id="pwd" required/>
  </fieldset>
</form>
```



The screenshot shows a web browser window with the address bar displaying 'jh1033.brighton.domains/ci435/t'. The page content includes a 'Login details:' section with two input fields. The first field, labeled 'User name:', contains the text 'Jennie Harding'. The second field, labeled 'Password:', contains masked characters '.....'. A callout box points to the password field, explaining that the input type is 'password'.

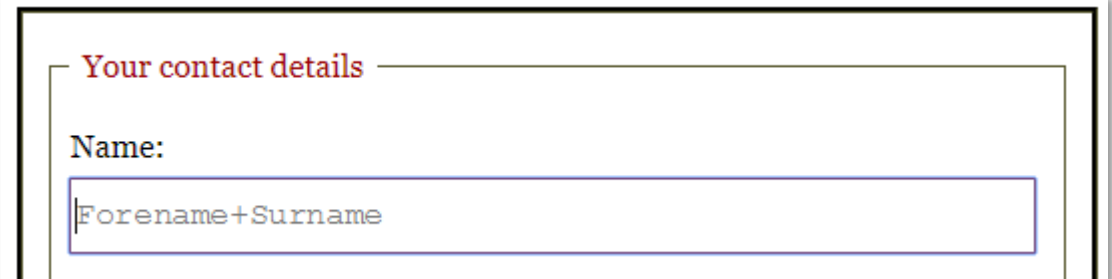
Both text fields are 'required'. When the password is entered it is not shown because the input type is "password"

# HTML5 forms – placeholder

- Display text in a form field to show the user how to complete it, using the 'placeholder' attribute -
  1. When a form field has no value, show placeholder text
  2. When user focuses on the field, remove placeholder text
  3. If user removes focus from the field, without completing it, reinstate placeholder text

```
<label for="name">Name: </label>
```

```
<input name="name" id="name" type="text"  
placeholder="Forename+Surname">
```



# Other form elements - <select>

- A drop down list control for selecting from a set of options

```
<label for="list">What pet would you like for Xmas?</label>  
<select name="list" id="list" size="4">  
  <option value="dog" selected="selected">Dog</option>  
  <option value="cat">Cat</option>  
  <option value="rabbit">Rabbit</option>  
  <option value="tortoise">Tortoise</option>  
</select>
```

'Dog' is selected  
initially; list is 4  
lines long



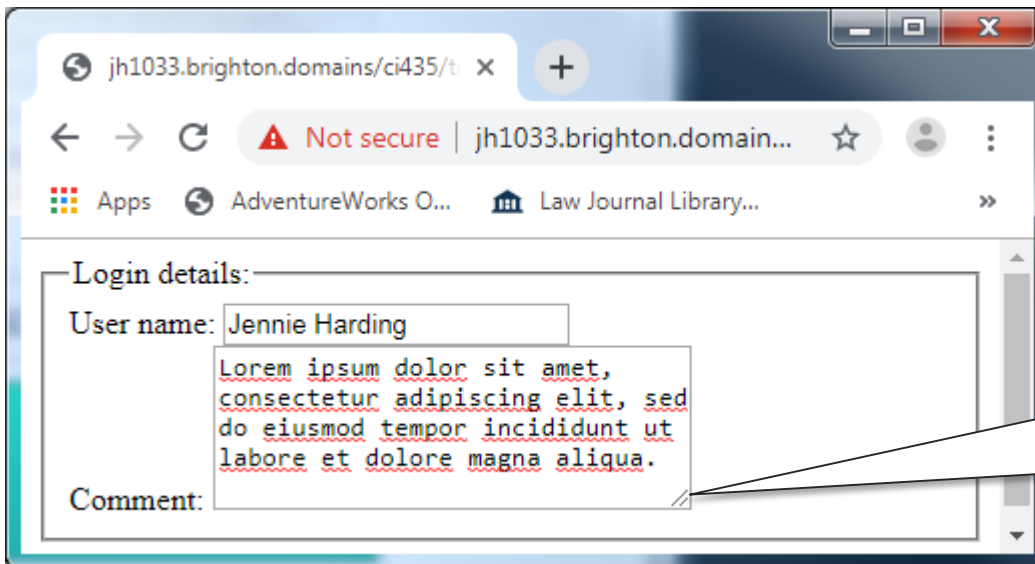
# Other form elements - <textarea>

- Multiline plain text edit control for entering text data of a variable and unspecified length

`<label for="comment">Comment: </label>`

`<textarea></textarea>`

- Specify size of <textarea> in CSS



A screenshot of a web browser window. The address bar shows the URL "jh1033.brighton.domains/ci435/t". The browser's address bar and tabs are visible. The page content includes a "Login details:" section. Under "User name:", there is a text input field containing "Jennie Harding". Below this, there is a "Comment:" label followed by a large text area. The text area contains the text "Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua." The text area has a small, light blue, L-shaped handle in its bottom-right corner, which is used for resizing the text box.

Browser applies a draggable corner to resize the text box



# HTML form elements

Other form elements improve accessibility and usability


- **<fieldset>** - group related form elements or blocks of data
- **<legend>** - for the name of a fieldset
- **<label>** - for describing a form element to the user and associating the description with the **<input>** element and control

# Browser styling

- Platforms (Mac, PC) and browser stylesheets style form elements differently – especially form **controls** (widgets) - buttons, checkboxes *etc.*
- Testing on different browsers is essential
- It's possible to override much of the default browser display with CSS3...
- ... older browsers will not render this – but the fallback of the browser styled element is acceptable ...
- ... and it might be better to stick with the native browser styling as this is consistent with the users' experience and expectations

# Browser styling

- A form styled by the browser, without CSS, has poor visual appearance, layout and User eXperience
- You need to style your forms to get a good grade



The screenshot shows a web browser window with a single tab titled 'Contact form'. The address bar indicates a 'Not secure' connection to 'jh1033.brighton.domai...'. The browser's bookmark bar shows 'Apps', 'AdventureWorks O...', and 'Law Journal Library...'. The page content features a heading 'Contact Jennie Harding' followed by the subheading 'Please get in touch'. Below this is a bulleted list of links: 'Learning Journal', 'Tutorial', and 'Contact me'. A section titled 'Your contact details' contains four input fields: 'Name (required)', 'Email address (required)', 'Website address', and a field with 'http://' pre-filled. Below the contact details is a 'Your comments:' section with a 'Message:' label and a text area. At the bottom of the form is a section 'Would you like more information?' with two radio buttons: 'Yes please' and 'No thanks' (which is selected). A 'Click to send' button is located below the radio buttons. The footer of the page reads '© 2019, Jennie Harding'.

Contact form

Not secure | jh1033.brighton.domai...

Apps AdventureWorks O... Law Journal Library...

## Contact Jennie Harding

### Please get in touch

- [Learning Journal](#)
- [Tutorial](#)
- [Contact me](#)

Your contact details

Name (required)  Email address (required)

Website address

http://

Your comments:

Message:

Would you like more information?

☐ Yes please ☒ No thanks

Click to send

© 2019, Jennie Harding

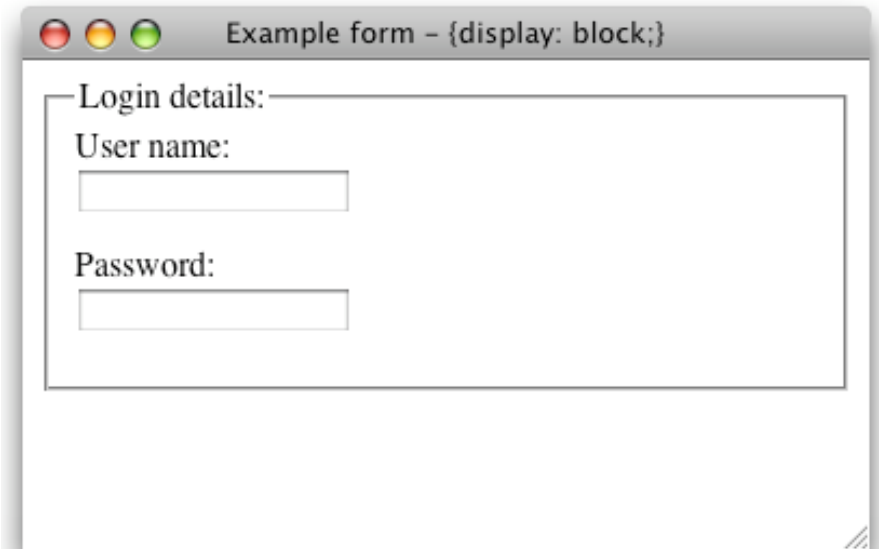
# Styling forms with CSS: layout

- *Don't* add `tables` and `divs` for form layout and styling – not necessary
- Input elements are inline – use CSS to display them as block elements *i.e.* laid out vertically rather than horizontally
- Define `width` property for `<input>` element types which have text field controls - *e.g.* the `'text'` input type)
- Define `margin` and `padding` properties for elements to create white space around them and control layout

# Styling forms with CSS: layout

- `<label>` is also an inline element and by default it will be on the same line as the input element it is associated with
- Create a CSS rule to display it as a block element –  

```
input, label {  
  display: block;  
}
```



Example form – {display: block;}

Login details:

User name:

Password:

# Making forms responsive

- Form elements should be fluid – *i.e.* sized in % or ems
- Use media queries to serve up different form layouts depending on device viewport width

The image illustrates a responsive form design. On the left, a desktop layout shows four sections: 'Two-Up' (Username and Password), 'Two-Up Wider' (Username and Password), 'Three-Up' (First, Middle, and Last name), and 'Zip, City, State' (ZIP code, City, and State dropdown). On the right, a mobile layout shows the same sections but with the 'Two-Up' and 'Two-Up Wider' sections stacked vertically, and the 'Three-Up' section with the First and Middle name fields stacked vertically.

# Making forms usable

Design for touchscreens by default

- Fingers are less precise than a mouse point and click – enlarge buttons and target areas for touch
  - Apple guidelines state that 44 x 44 pixels is the **minimum** size for usable buttons

<http://www.smashingmagazine.com/2012/02/21/finger-friendly-design-ideal-mobile-touchscreen-target-sizes/> - article says that controls such as buttons should be at least 57 pixels wide, the pixel width of an average finger; 72 pixels wide for buttons for thumb use

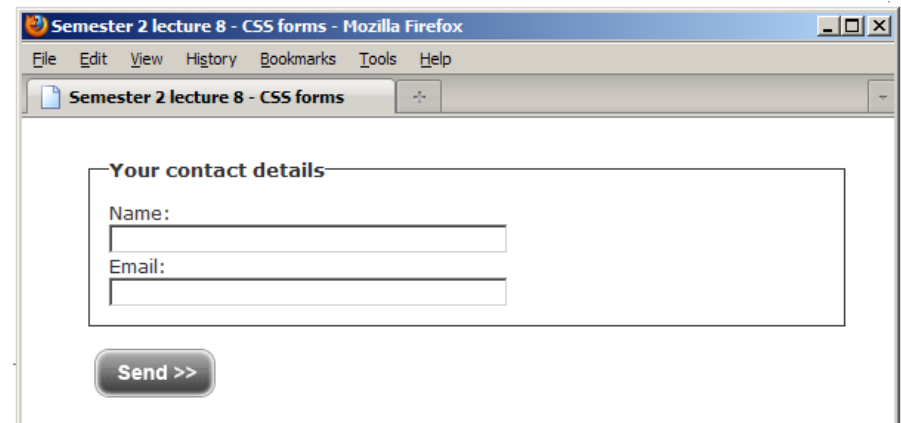


## 57 Pixel Touch Target

Index finger fits snugly inside. Target edges give visual feedback. Finger pad is used instead of fingertip.

# Styling the <button> element

- There is an alternative to using `<input type="submit" />` or `<input type="button" />` for the button control on the form
- The `<button>` element, with the **type attribute**, can be used together with an image and the **alt attribute** –  
`<button type="submit">`  
``  
`</button>`
- Advantage is that the button element can be styled with an image or icon – either in the HTML or a CSS background image





# Styling the `<button>` element

- Pure CSS buttons are an even better option than an image  
`<button type="submit">Send login details</button>`
- Create a CSS rule to select the `<button>` element ...
- ...and pseudo-class rules for **`:hover`**, **`:active`** and **`:focus`** states to create a rollover, clickable effect
- Advantage of this method is that there is text content in the HTML rather than an image - for accessibility
- Could also use CSS3 **`border-radius`**, **`box-shadow`**, **`text-shadow`** and **`gradient`** properties...
- Google "**CSS forms**" for lots of great examples and tutorials

# Resources and reading

- Mozilla Developer Network, *HTML Forms Guide* - <https://developer.mozilla.org/en-US/docs/Web/Guide/HTML/Forms>
  - Includes **My First HTML Form** - [https://developer.mozilla.org/en-US/docs/Web/Guide/HTML/Forms/My first HTML form](https://developer.mozilla.org/en-US/docs/Web/Guide/HTML/Forms/My_first_HTML_form)
- Clark, R., 2013. HTML5 forms input types. *HTML5 Doctor*. <http://html5doctor.com/html5-forms-input-types/> Essential reading - this article covers **13** new HTML5 form input types.
  - <http://html5doctor.com/html5-forms-introduction-and-new-attributes/>
- Can I use ... HTML5 form features. <https://caniuse.com/#feat=forms>