Introduction to HTML

Max Design - Hands-on CSS

Lesson FIVE

Max Design - Hands-on CSS

What will we cover in this lesson?

Lesson 5: Form markup

What are forms?
<form> element
<fieldset> element
<legend> element
<label> element
<input> elements
input type="text"
input type="email"
input type="url"
input type="tel"
input type="password"
input type="file"
input type="checkbox"
input type="radio"
input type="hidden"
input type="reset"
input type="image"
input type="submit"
input type="button"
<button> element
<select> & <option>
<optgroup> element
<textarea> element

Open the
exercise folder

Open the folder called “start” and then open the file called **“lesson05.htm”** using some sort of HTML editor.

Note:

What are forms?

HTML forms are forms that are placed in web pages. These forms contain special elements called “form controls” (checkboxes, radio buttons, menus, etc.) and their associated labels.

Users “complete” a form by **modifying the various form controls** (entering text, selecting menu items, etc.), and then submitting the form to the server.

Step 1:
<form> element

The **<form> element** is used to create an HTML form.

```
<form>
</form>
```

The <form> element requires a **start and end tag**.

```
<form>
</form>
```

There are a **range of attributes** you can use within the <form> element.

The **ACTION attribute** is used to specify where the form data is sent after it is submitted. The value is normally a URL.

```
<form action="submit.php">
</form>
```

The **METHOD attribute** specifies the HTTP method to be used when sending the form data. The two values are “get” and “post”.

```
<form method="get">
</form>
```

Exercise

Let’s add some form attributes...

```
<form>
    ...
</form>
```

```
<form action="#" method="get">
    ...
</form>
```

Step 2:
<fieldset> element

The **<fieldset> element** is used to group related parts of a form. It is mainly used in complex forms.

```
<fieldset>
</fieldset>
```

The <fieldset> element requires a **start and end tag**.

```
<fieldset>
</fieldset>
```

More than one <fieldset> element can be used inside a form.

```
<fieldset>
</fieldset>
<fieldset>
</fieldset>
```

The <fieldset> element can also contain **nested <fieldset> elements**.

```
<fieldset>
   <fieldset>
   </fieldset>
</fieldset>
```

By default, the <fieldset> element will **draw a box** around the content inside.

Fieldset content...

Exercise

Let’s add two fieldsets...

```
<form action="#" method="get">
    <fieldset>

    </fieldset>
    <fieldset>

    </fieldset>
</form>
```

Step 3:

<legend> elements

The **<legend> element** defines a caption for the <fieldset>.

```
<fieldset>
   <legend>Legend here</legend>
</fieldset>
```

The <legend> element requires a **start and end tag**.

```
<legend>Legend here</legend>
```

The **<legend> element** can only contain inline content.

```
<fieldset>
   <legend>Legend here</legend>
</fieldset>
```

There can only be **one <legend>** inside each <fieldset>. The <legend> must be placed directly after the <fieldset> start tag.

By default, the <legend> element is **rendered on top** of the <fieldset> border.

Legend here

Fieldset content...

Exercise

Let’s add legends to our fieldsets...

```html
<form action="#" method="get">
    <fieldset>
        <legend>Your details</legend>
    </fieldset>
    <fieldset>
        <legend>Your information</legend>
    </fieldset>
</form>
```

Note:
<label> element

The **<label> element** defines a label for individual form controls.

```
<label>Email address</label>
<input type="text">
```

The **<label> element** requires a start and end tag.

```
<label>Email address</label>
```

Tying the <label> to its form control

Ideally, the <label> element should be **tied to the relevant form control**. This can be achieved using two different methods.

Method 1: Wrap the <label> element around the relevant form control.

```
<label>
    Email address
    <input type="text">
</label>
```

Method 2: Use the FOR and ID attributes to explicitly tie the label to the form control.

```
<label for="email">Email address</label>
<input id="email" type="text">
```

Method 2 is the preferred method as the FOR attribute **specifically tells devices** that this label is “for” the relevant form control.

Label before or after?

The contents of the label should **come before all form controls** except in the case of checkboxes and radio buttons.

```
<label for="email">Email address</label>
<input id="email" type="text">
```

In the case of checkboxes and radio buttons, the contents of the label should **come after the form control**.

```
<input id="choice1" type="checkbox">
<label for="choice1">Choice 1</label>
```

Form controls that don’t require labels

Some form controls **do not require labels**.

```
<input type="submit" value="submit">
```

In the cases below, browsers use the “value” as a label, so a **<label> is not required**.

```
<input type="button" value="Submit">
<input type="reset" value="Reset">
<input type="submit" value="Submit">
```

In the cases of <button> elements, the content inside the element is displayed, so a **<label> is not required**.

```
<button value="Submit">Submit</button>
```

In the case of input type="hidden", the form control is used to send instructions to the server, so a **<label> is not required**.

```
<input type="hidden" value="instructions-
to-server-here">
```

Note:
<input> elements

The **<input> element** specifies an input field where the user can enter data.

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

The **<input> element** is a void element, so there is no end tag.

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

input types

There are a range of different <input> elements that serve different purposes. The type of input is **defined by the TYPE attribute**.

```
<input type="button">
<input type="checkbox">
<input type="file">
```

The <input type="text">
is the **default input type**.
If the type attribute is
omitted or not supported by
the device, <input
type="text"> will be used.

In HTML 4.01 there were **ten different input types**.

```
<input type="button">
<input type="checkbox">
<input type="file">
<input type="hidden">
<input type="image">
<input type="password">
<input type="radio">
<input type="reset">
<input type="submit">
<input type="text">
```

In HTML5, **thirteen new input types** have been introduced. We will cover the three most practical of these new inputs.

```
<input type="email">
<input type="url">
<input type="tel">
```

input attributes

There are a **wide range of different attributes** that can be used with the <input> element.

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

We will include the **ID attribute** to link the input to the relevant <label> element.

```
<input id="email" type="text">
```

In some cases, we will include the **NAME attribute** as this helps to describe the data being submitted to the server.

```
<label for="email">Email</label>
<input type="text" id="email" name="email">
```

In some cases, we will include the **VALUE attribute** as the name information being submitted to the server is not enough on its own.

```
<input id="brochure" name="print"
type="checkbox" value="brochure">
```

Wrapping things in a

In our exercise, you can see that we have wrapped **each label and form control inside a **, as this gives us greater control of the layout.

Step 4:

input type= "text"

The input type="text" creates a **single-line text input control**.

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

Exercise

Let’s add input type=“text”

```html
<div>
    <label for="nm">Name</label>
    <input id="nm" name="nm">
</div>
```

```html
<div>
    <label for="nm">Name</label>
    <input id="nm" name="nm" type="text">
</div>
```

Step 5:

input type= "email"

The input type="email" creates an input control for **one or more email addresses**.

```
<input type="email">
```

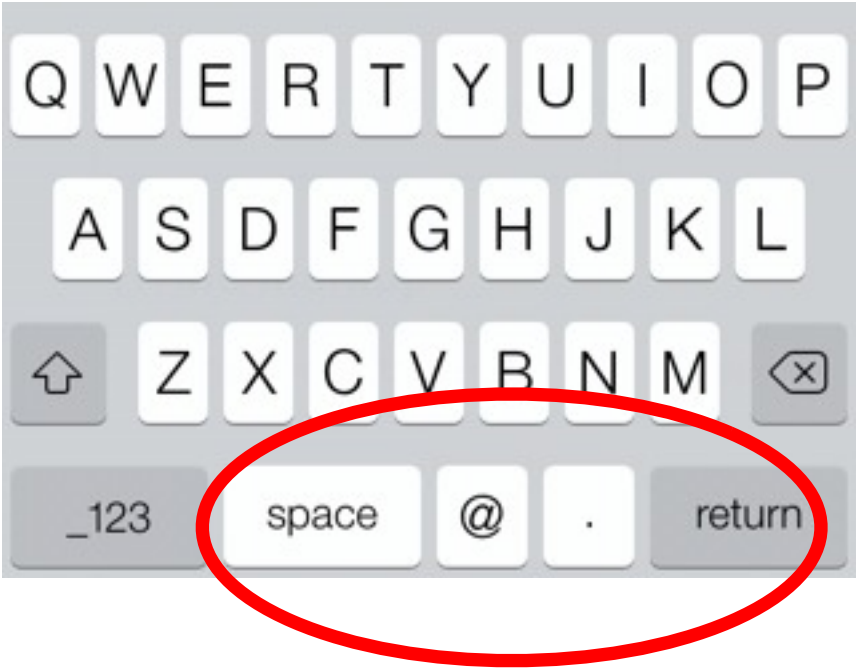
As users apply content into this field, some devices look for the “@” symbol and **flag the field red if this symbol is not present**.

Email: Jeff.smith

In some mobile devices such as the iPhone, this field triggers the keyboard to **change from a standard keyboard** to an email keyboard.

Standard keyboard

Email keyboard



Exercise

Let’s add input type=“email”

```html
<div>
    <label for="em">Email address</label>
    <input id="em" name="em" type="email">
</div>
```


```html
<div>
    <label for="em">Email address</label>
    <input id="em" name="em" type="email">
</div>
```

Step 6:

input type= "url"

The input type="url" creates an **input control for a web address**.

```
<input type="url">
```

As users apply content into this field, some devices look for the “http://” symbol and **flag the field red if this content is not present**.

URL: sample.com

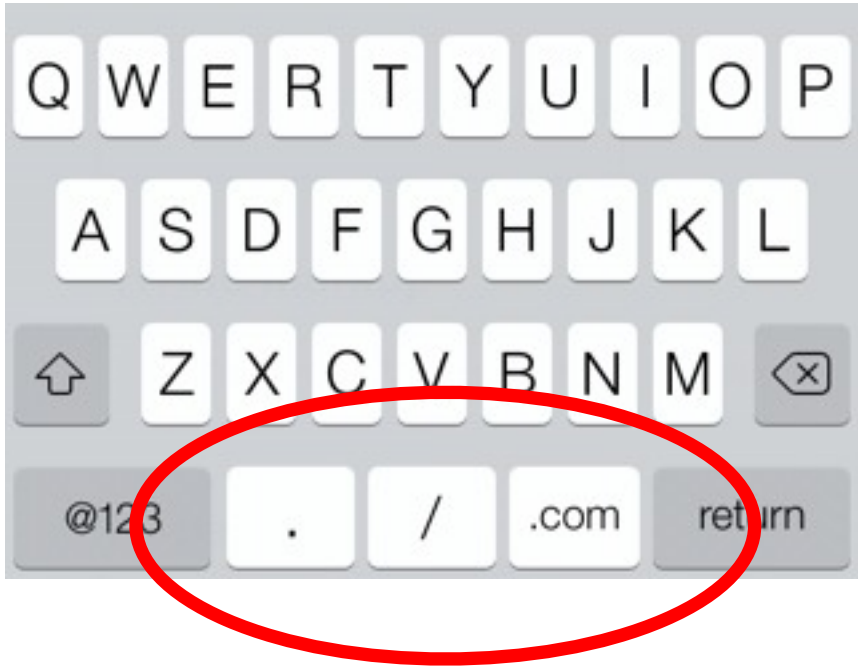
Other devices add **“http://” automatically** for users.

URL: http://sample.com

In some mobile devices such as the iPhone, this field triggers the keyboard to **change from a standard keyboard** to a URL keyboard.

Standard keyboard

URL keyboard



Exercise

Let’s add input type=“url”

```html
<div>
    <label for="wb">Website</label>
    <input id="wb" name="wb">
</div>
```

```html
<div>
    <label for="wb">Website</label>
    <input id="wb" name="wb" type="url">
</div>
```

Step 7:

input type= "tel"

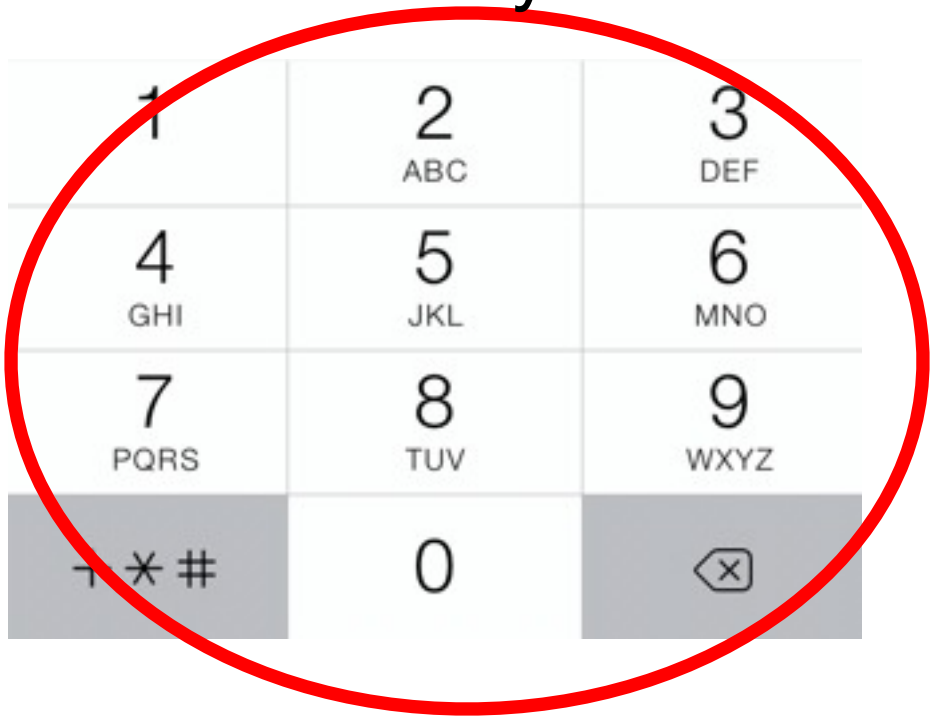
The input type="tel" creates an **input control for telephone numbers**.

```
<input type="tel">
```

In some mobile devices such as the iPhone, this field triggers the keyboard to **change from a standard keyboard** to a number keyboard.

Standard keyboard

Number keyboard



Exercise

Let’s add input type=“tel”

```html
<div>
    <label for="ph">Phone number</label>
    <input id="ph" name="ph">
</div>
```

```html
<div>
    <label for="ph">Phone number</label>
    <input id="ph" name="ph" type="tel">
</div>
```

Step 8:

input
type="password"

The input type="password" is like “text” **except input characters are blocked on screen**.

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

Exercise

Let’s add input type=“password”

```html
<div>
    <label for="ps">Password</label>
    <input id="ps" name="ps">
</div>
```

```html
<div>
    <label for="ps">Password</label>
    <input id="ps" name="ps" type="password">
</div>
```

Step 9:

input type="file"

The input type="file" creates a **file select control** that allows the user to select files and submit them with a form.

```
<input type="file">
```

The input type="file" **may look different** based on individual browsers and operating systems.

Photo Choose

Photo Choose file No file chosen

Exercise

Let’s add input type=“file”

```html
<div>
    <label for="upload">Upload photo</label>
    <input id="upload" name="upload">
</div>
```

```html
<div>
    <label for="upload">Upload photo</label>
    <input id="upload" name="upload"
type="file">
</div>
```

Step 10:

input
type="checkbox"

The input type="checkbox" is an **on/off switch that may be toggled by the user**.

```
<input type="checkbox">
```

If a checkbox is selected, the name and value attribute values will be **submitted as a name/value pair**.

```
<input id="brochure" type="checkbox"
name="print" value="brochure">
```

The <label> associated with the checkbox **should be placed after** the checkbox.

```
<input type="checkbox">
<label>Checkbox 1</label>
```

When adding multiple checkboxes into a form, an unordered list can be used to mark them up as they are **list of options**.

Exercise

Let’s add input type=“checkbox”

```
<li><input id="br" name="print" value="br">
<label for="br">Brochure</label></li>
<li><input id="po" name="print" value="po">
<label for="po">Poster</label></li>
```

```
<li><input id="br" name="print"
type="checkbox" value="br">
<label for="br">Brochure</label></li>
<li><input id="po" name="print"
type="checkbox" value="po">
<label for="po">Poster</label></li>
```

Step 11:

input type="radio"

The input type="radio" is like a checkbox except that when several inputs share the same name value, **they are mutually exclusive**.

```
<input type="radio" name="cx" value="ch1">
<input type="radio" name="cx" value="ch2">
<input type="radio" name="cx" value="ch3">
```

If a radio button is selected, the name and value attribute values will be **submitted as a name/value pair**.

```
<input id="brochure" type="checkbox"
name="delivery" value="monthly">
```

The <label> content associated with the radio button **should be placed after** the radio button.

```
<input type="radio">
<label>Radio 1</label>
```

When adding multiple radio buttons into a form, an unordered list can be used to mark them up as they are **list of options**.

Exercise

Let’s add input type=“radio”

```
<li><input id="wk" name="del"  value="wk">
<label for="wk">Weekly</label></li>
<li><input id="mn" name="del" value="mn">
<label for="mn">Monthly</label></li>
```

```
<li><input id="wk" name="del" type="radio"
value="wk">
<label for="wk">Weekly</label></li>
<li><input id="mn" name="del" type="radio"
value="mn">
<label for="mn">Monthly</label></li>
```

Step 12:

input type="hidden"

The input type="hidden" creates a control that is **not rendered** (hidden). However, its values are submitted with other form data.

```
<input type="hidden">
```

No <label> element is required for hidden inputs.

Exercise

Let’s add input type=“hidden”

```
<div>
    <input name="loc" value="en-US">
</div>
```

```
<div>
    <input type="hidden" name="loc"
    value="en-US">
</div>
```

Step 13:

input type="reset"

The input type="reset" creates a **reset button** that resets all controls to their initial values.

```
<input type="reset" value="Reset">
```

No <label> element is required for the reset input.

If used, the reset button should be placed carefully on any form as it **can be confused for a submit form**.

Users can become frustrated if they think they have clicked “submit” only to find they have **wiped all of their carefully entered data**.

Exercise

Let’s add input type=“reset”

```html
<div>
    <input value="Reset">
</div>
```

```html
<div>
    <input type="reset" value="Reset">
</div>
```

Step 14:

input type="image"

The input type="image" creates a **graphical submit button** that is used to submit form data.

```
<input type="image" src="btn.png"
alt="Submit">
```

The **SRC attribute** is required. It tells the browser where to find the image.

```
<input type="image" src="btn.png"
alt="Submit">
```

The **ALT attribute** is required. It provides a text alternative for devices that cannot load the image.

```
<input type="image" src="btn.png"
alt="Submit">
```

No <label> element is required for the image input.

Exercise

Let’s create an image...

```
<div>
    <input src="button.png">
</div>
```

```
<div>
    <input type="image" src="button.png"
alt="Submit">
</div>
```

Step 15:

input type="submit"

The input type="submit" creates a **submit button** that is used to submit form data.

```
<input type="submit" value="Submit">
```

No <label> element is required for the submit input.

Exercise

Let’s add input type=“submit”

```html
<div>
    <input value="Submit">
</div>
```

```html
<div>
    <input type="submit" value="Submit">
</div>
```

Step 16:

input type="button"

The input type="button" creates a **push button**. Unlike the submit button, push buttons have no default behavior.

```
<input type="button" value="Submit">
```

No <label> element is required for the button input.

Exercise

Let’s add input type=“image”, SRC and ALT

```html
<div>
    <input value="Submit">
</div>
```

```html
<div>
    <input type="button" value="Submit">
</div>
```

Step 17:
<button> element

The button element creates a **button** just like input type="button".

```
<button type="submit">Submit</button>
```

The button element includes an **open and closing tag**.

```
<button>Submit</button>
```

The button element is very flexible as it allows **content as well as HTML markup** inside the element.

```
<button type="submit"><em>Submit</em></
button>
```

No <label> element is required for the <button> element.

Exercise

Let’s add input type=“submit”

```html
<div>
    <button>Submit</button>
</div>
```

```html
<div>
    <button type="submit">Submit</button>
</div>
```

Step 18:
<select> and <option> elements

The **<select> element** creates a dropdown menu. Each choice offered by the menu is represented by an option element.

```
<select>
    <option>Item 1</option>
    <option>Item 2</option>
</select>
```

The <select> element includes start and end tags. The element must contain at least **one option element**.

The boolean **MULTIPLE attribute** can be used to change the element from a dropdown list to a multiple choice list.

```
<select multiple>
    <option>Item 1</option>
    <option>Item 2</option>
</select>
```

The **<option> element** is used to represent each option within the dropdown.

```
<select>
    <option>Item 1</option>
    <option>Item 2</option>
</select>
```

Each <option> element should include a unique **VALUE attribute**.

```
<select>
    <option value="item1">Item 1</option>
    <option value="item2">Item 2</option>
</select>
```

In the cases where the <option> is instructional (ie. “Choose an option”), **a blank value** can be used.

```
<select>
    <option value="">Choose</option>
    <option value="item1">Item 1</option>
    <option value="item2">Item 2</option>
</select>
```

The boolean **SELECTED attribute** can be used to determine the <option> that is displayed by default.

```
<select>
    <option value="" selected>Choose</option>
    <option value="item1">Item 1</option>
    <option value="item2">Item 2</option>
</select>
```

Exercise

Let’s add a boolean attribute

```html
<select id="car" name="car">
    <option value="">Choose</option>
    <option value="volvo">Volvo</option>
    <option value="saab">Saab</option>
    <option value="merc">Merc</option>
    <option value="audi">Audi</option>
</select>
```

```html
<select id="car" name="car">
    <option value="" selected>Choose</option>
    <option value="volvo">Volvo</option>
    <option value="saab">Saab</option>
    <option value="merc">Merc</option>
    <option value="audi">Audi</option>
</select>
```

Step 19: <optgroup> element

The **<optgroup> element** is used to group related options in a drop-down list.

```
<optgroup>
    <option value="volvo">Volvo</option>
    <option value="saab">Saab</option>
</optgroup>
<optgroup>
    <option value="merc">Merc</option>
    <option value="audi">Audi</option>
</optgroup>
```

Each <optgroup> element should be **given a LABEL attribute** to describe the group.

```
<optgroup label="Swedish Cars">
    <option value="volvo">Volvo</option>
    <option value="saab">Saab</option>
</optgroup>
<optgroup label="German Cars">
```

This label is **displayed as a greyed out label** (cannot be chosen by users) within the dropdown list.

Exercise

Let’s create some optgroups...

```
<option value="" selected>Choose</option>
<option value="volvo">Volvo</option>
<option value="saab">Saab</option>
<option value="merc">Merc</option>
<option value="audi">Audi</option>
```

```
<option value="" selected>Choose</option>
<optgroup label="Swedish Cars">
    <option value="volvo">Volvo</option>
    <option value="saab">Saab</option>
</optgroup>
<optgroup label="German Cars">
    <option value="merc">Merc</option>
    <option value="audi">Audi</option>
</optgroup>
```

Step 20:
<textarea> element

The textarea element is used to create a **multi-line text input control**.

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

The textarea element includes an **open and closing tag**. No content is allowed between the opening and closing tags.

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

Exercise

Let’s create a textarea...

```html
<div>
    <label for="cm">Add a comment</label>
</div>
```

```html
<div>
    <label for="cm">Add a comment</label>
    <textarea id="cm" name="cm">
    </textarea>
</div>
```

Russ Weakley

Max Design

Site: maxdesign.com.au

Twitter: twitter.com/russmaxdesign

Slideshare: slideshare.net/maxdesign

Linkedin: linkedin.com/in/russweakley