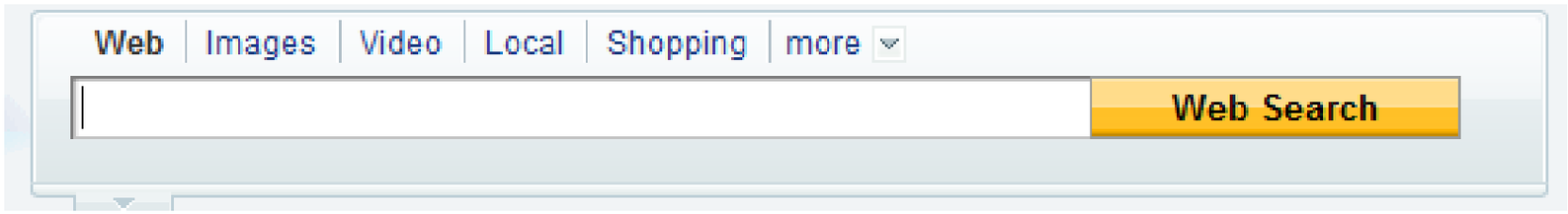
Forms

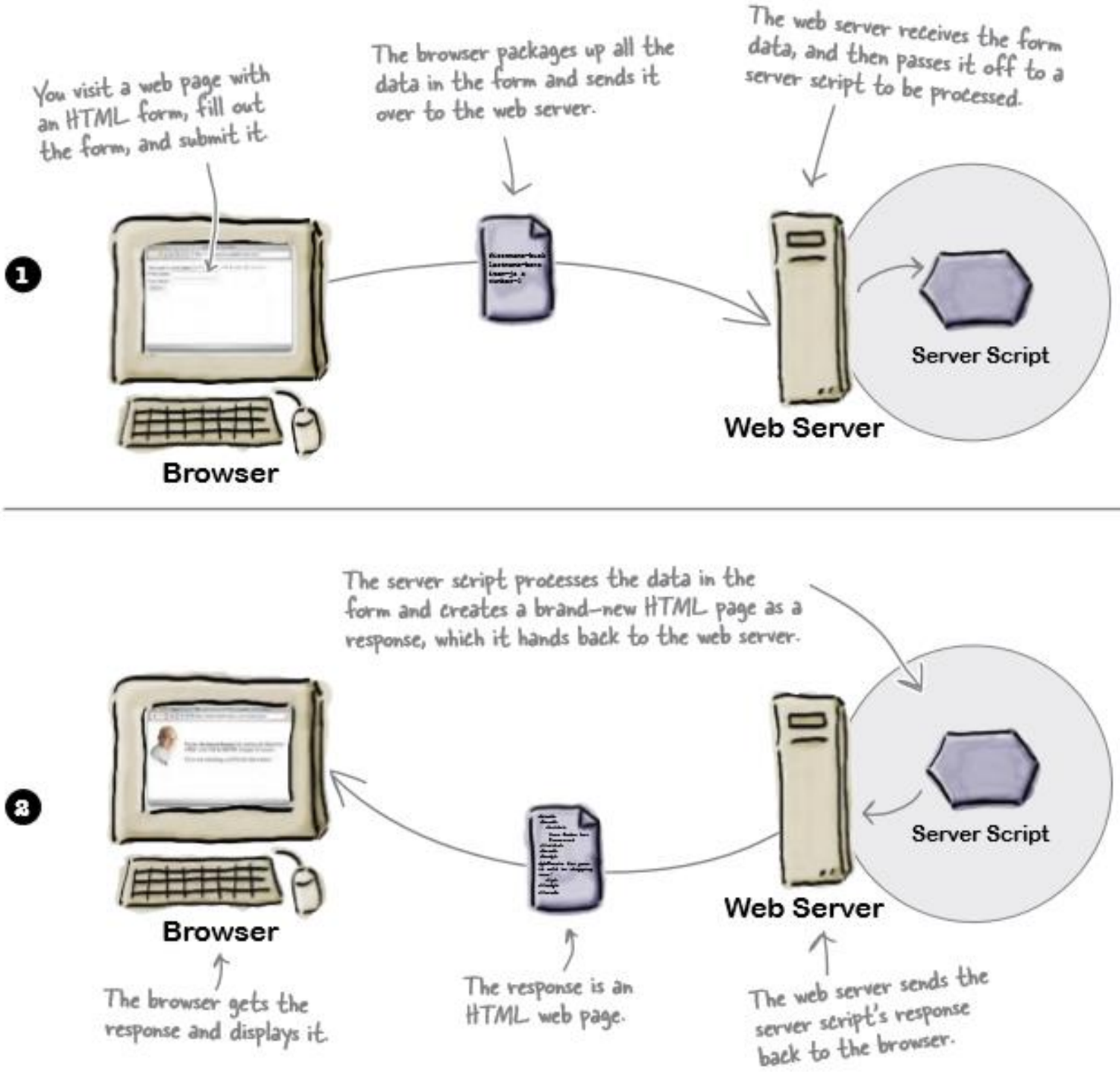
Overview of forms

- Forms are used all over the Web to
  - Accept information.
  - Provide interactivity.



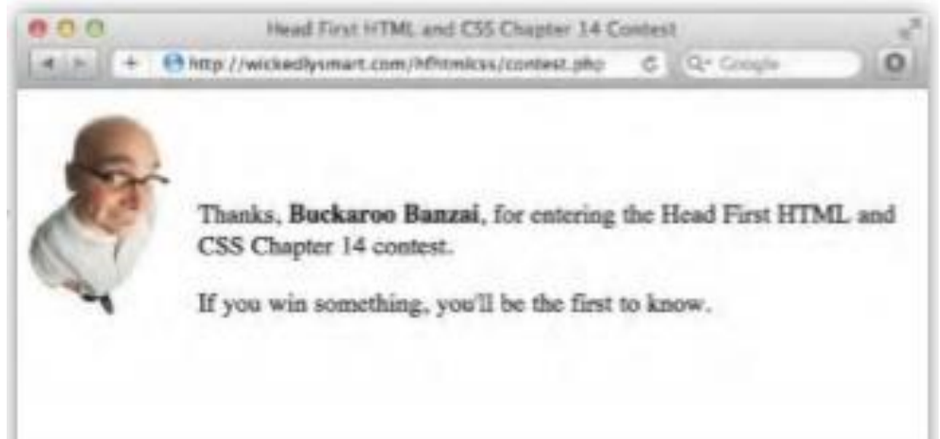
- Types of forms:
  - Search form, Order form, Newsletter sign-up form, Survey form, Add to Cart form, and so on…

Server side


You visit a web page with an HTML form, fill out the form, and submit it.
The browser packages up all the data in the form and sends it over to the web server.
The web server receives the form data, and then passes it off to a server script to be processed.
1
Server Script
Web Server
Browser
The server script processes the data in the form and creates a brand-new HTML page as a response, which it hands back to the web server.
8
Server Script
Web Server
Browser
The browser gets the response and displays it.
The response is an HTML web page.
The web server sends the server script's response back to the browser.

How forms work in the browser

- The browser loads the page, it creates controls on the page that allow you to input various kinds of data.
- You enter data, type a single line of text, click an option, checkbox control etc.
- You submit the form by clicking a submit button.
- The server responds by receiving the data and processing it and presenting a new html page.



Form elements

```
<form action="http://wickedlysmart.com/hfhtmlcss/contest.php"
      method="POST">
   <p>Just type in your name (and click Submit) to
      enter the contest: <br>

    First name: <input type="text" name="firstname" value=""> <br>
    Last name: <input type="text" name="lastname" value=""> <br>
    <input type="submit">

   </p>
</form>
```

A

B

C

D

We've got the <form> element itself...

...and a bunch of elements nested inside it

Form elements

What can go on a form?

text input

The text <input> element is for entering one line of text. Optional attributes let you set a maximum number of characters and the width of this control.

Name: Buckaroo Banzai

An <input> element with a type attribute of "text" creates a one-line control in the browser page.

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

- The password <input> element works just like the text <input> element except that the text is masked.

••••••

What can go on a form?

- Radio buttons create a single control of several buttons of which only one can be chosen.

radio input

The radio <input> element creates a single control with several buttons, only one of which can be selected at any time. These are like old-time car radio buttons; you "push" one in, and the rest "pop out."

hot
not

The radio control allows only one of a set of choices.

Use a radio <input> for each choice.

All the radio buttons associated with a given set of choices must have the same name...

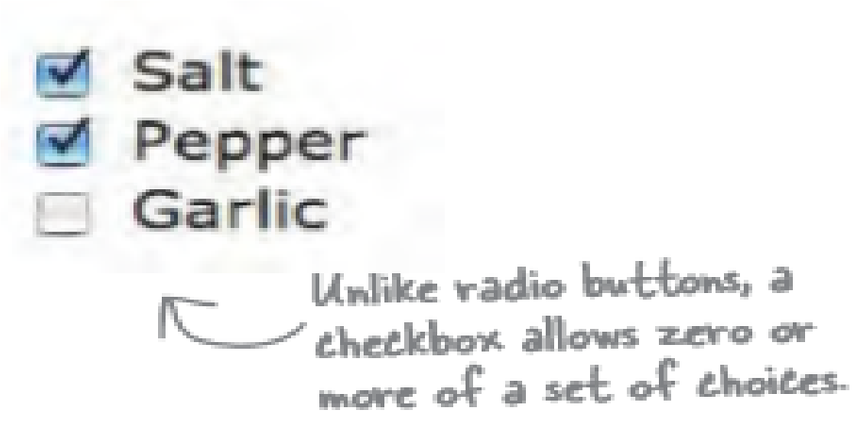
...but each choice has a different value.

```
<input type="radio" name="hotornot" value="hot">
<input type="radio" name="hotornot" value="not">
```

What can go on a form?

checkbox input

A checkbox <Input> element creates a checkbox control that can be either checked or unchecked. You can use multiple checkboxes together, and if you do, you can check as many or few as you like.



Like radio, you use one checkbox <input> element for each choice.

Related checkboxes also share a common name.

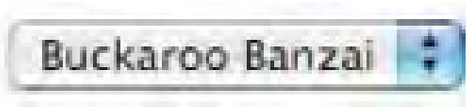
Each checkbox has a different value.

```
<input type="checkbox" name="spice" value="Salt">
<input type="checkbox" name="spice" value="Pepper">
<input type="checkbox" name="spice" value="Garlic">
```

What can go on a form?

select

The <select> element creates a menu control in the web page. The menu provides a way to choose between a set of choices. The <select> element works in combination with the <option> element below to create a menu.



The select element creates a menu that looks like this (although the look will vary depending on the browser you're using).

After clicking on the menu, the menu items drop down.

The <select> element goes around all the menu options to group them into one menu.

Just like the other form give the select element a name using the name att

```
<select name="characters">
    <option value="Buckaroo">Buckaroo Banzai</option>
    <option value="Tommy">Perfect Tommy</option>
    <option value="Penny">Penny Priddy</option>
    <option value="Jersey">New Jersey</option>
    <option value="John">John Parker</option>
</select>
```

What can go on a form?

HTML5 adds more input types:

number input

The number <input> element restricts input to numbers. You can even specify a min and max number that is allowed with optional attributes.

7

Some browsers show arrows next to the input area you can use to increase or decrease the number.

The "number" type means you're expecting a number only, not text.

Use the max and min attributes to restrict the numbers allowed.

```
<input type="number" min="0" max="20">
```

range input

The range <input> element is similar to number except that it displays a slider instead of an input box.

Both number and range have an optional step attribute you can use to specify the number of intervals for the values.

```
<input type="range" min="0" max="20" step="5">
```

What can go on a form?

color input

Use the color <input> to specify a color. When you click on the control, a color picker pops up that allows you to select a color rather than having to type in the color name or value.

If the color input is not supported by the browser, you'll just get a regular text input instead.


```
<input type="color">
```

email input

The email <input> element is just a text input, but on some mobile browsers, you'll get a custom keyboard for email when you start typing.

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

Email: Buckaroo Banzai

tel input

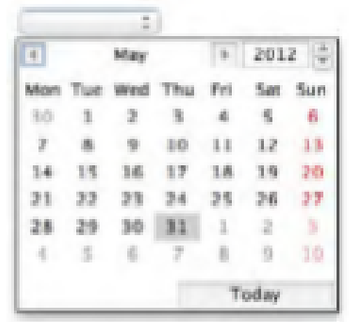
The tel <input> element is also just a text input, but like email, causes a custom keyboard to pop up on mobile devices.

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

Phone: 555-1212

date input

Use the date <input> element to specify a date, with a date picker control. The control creates a valid date format string to send to the server script.

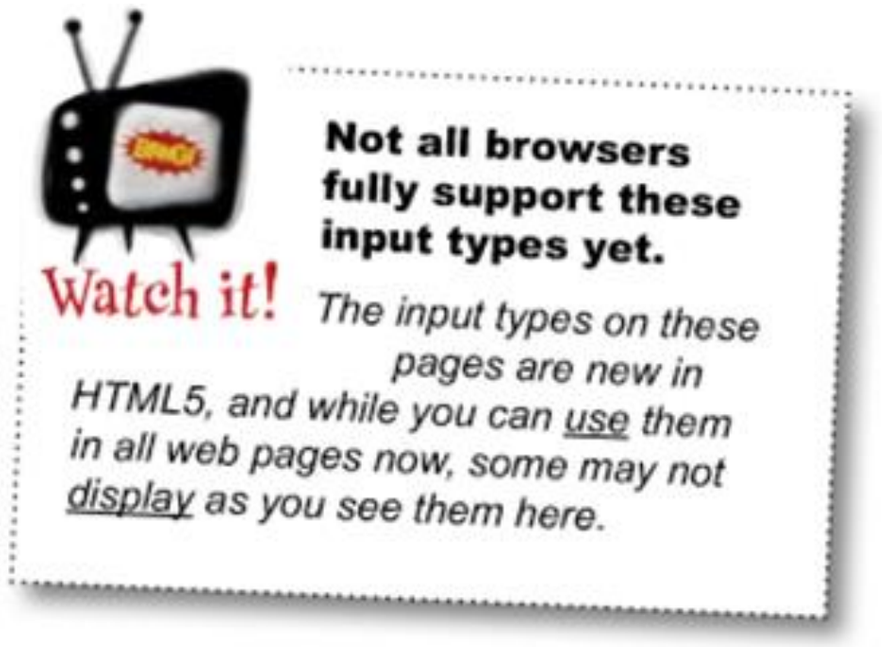



```
<input type="date">
```

Like with color, if the date input isn't supported by the browser yet, you'll get a regular text input instead.

What can go on a form?

- Even with these specialised types, it's up to you to make sure you know what values the server script is expecting and use the right <input> type.



What can go on a form?

- Not every form element is an <input> element there are some others for typing more than one line of text:

textarea

The <textarea> element creates a multiline text area that you can type into. If you type more text than will fit into the text area, then a scroll bar appears on the right side.

The <textarea> element is not an empty element, so it has both opening and closing tags.

Use the name attribute to give the element a unique name.

The cols attribute tells the browser how many characters wide to make the text area.

```
<textarea name="comments" rows="10" cols="48"></textarea>
```

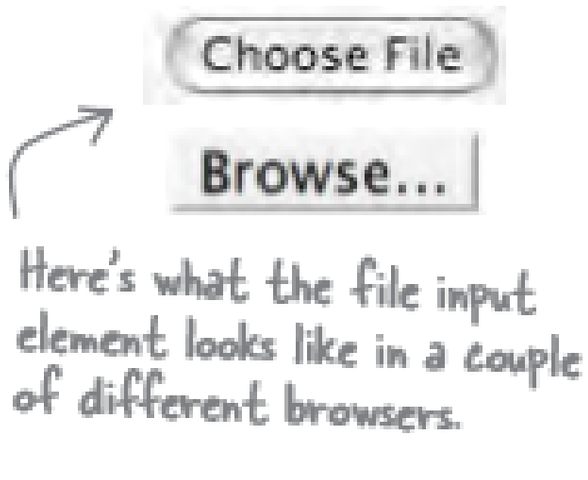
The rows attribute tells the browser how many characters tall to make the text area.

Any text that goes between the opening and closing tags becomes the initial text in the browser's text area control.

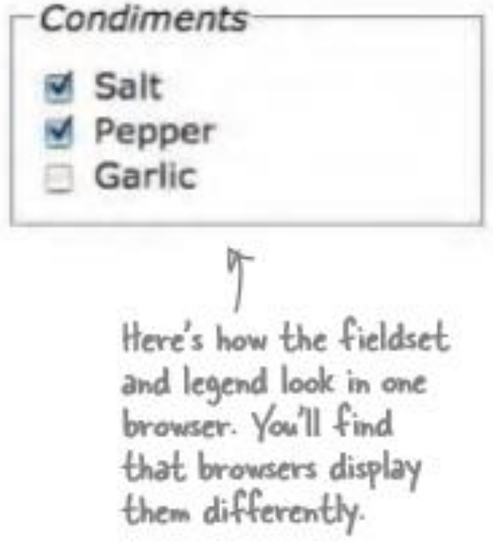
You can also specify the width and height of a textarea using CSS.

What more could go into a form?

- If you need to send an entire file you can set the input type to "**file**", this creates a control that allows you to choose a file and the contents when the form is submitted is sent to the server.

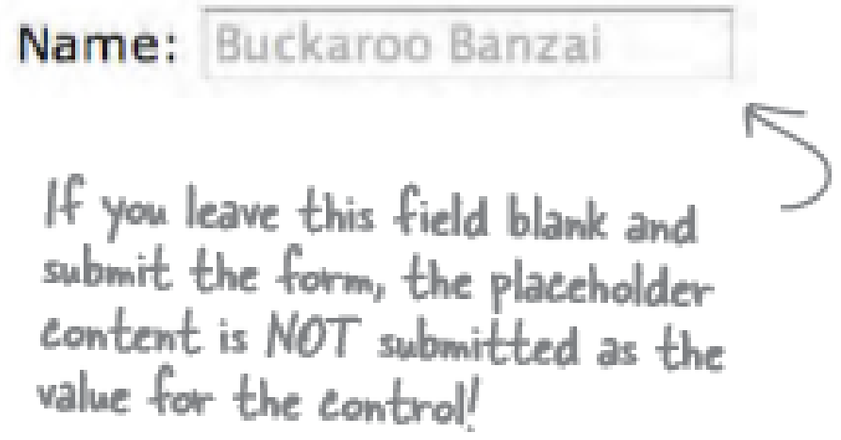


- A **<fieldset>** element can be used to group together common elements.
- It then can also make use of an element called **<legend>**.

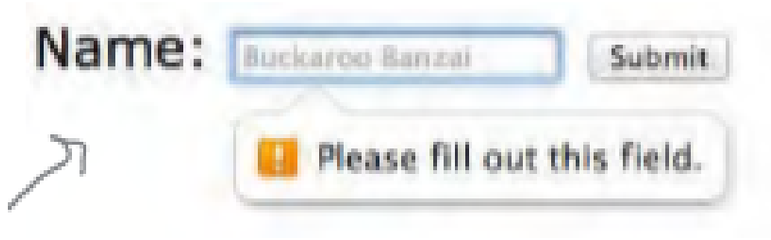


What more could go on a form?

- The **placeholder** attribute can be used with most <input> types in a form, it gives a hint about the kind of content expected.



- The **required** attribute forces the user to enter content. An error will show if the user leaves it blank.

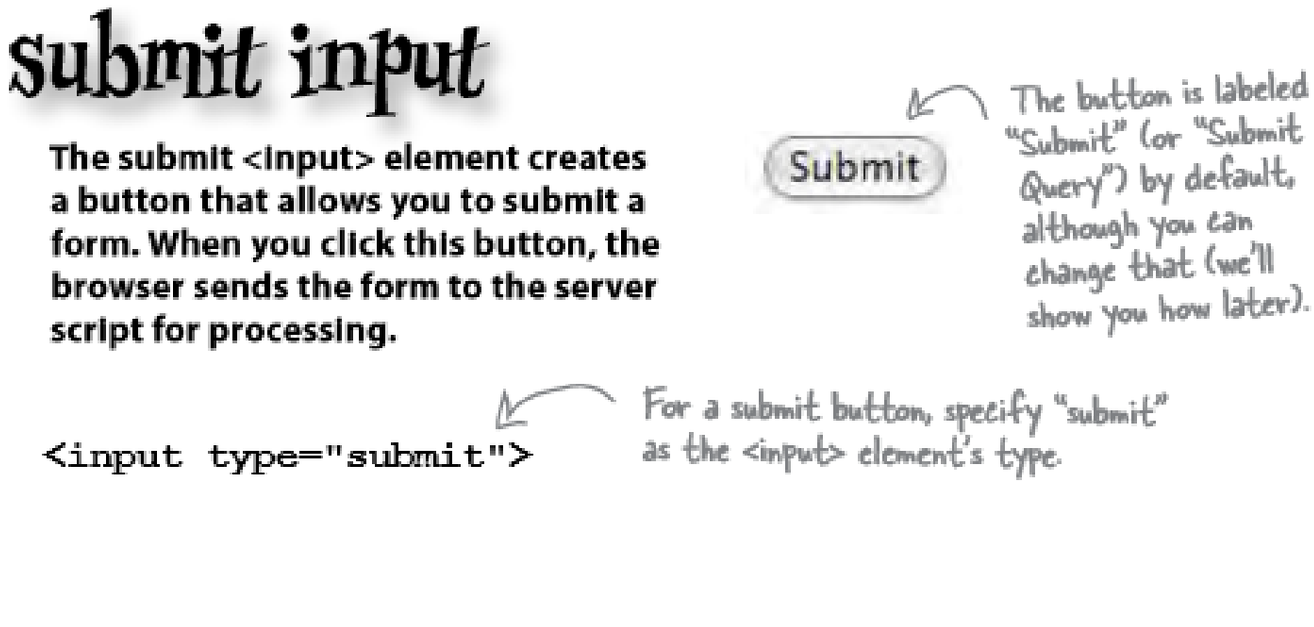


- If a form element includes the **autofocus** attribute then the cursor is in the form element when the form opens.

What more could go on a form?

- We can also include a **Pattern**. This allows us to specify the type of data, and the format required.
  - pattern="[a-zA-Z0-9]": All alphanumeric characters.
  - pattern="[A-Za-z]{3}" : Three letter code.
  - pattern = "[A-Z][a-z]+([A-Z][a-z]+)*": Name (with optional second name, and more names...).
  - pattern ="\S{6,}": At least 6 characters (excluding space, tab, newline, carriage return, form feed).
  - pattern="\d{4,8}": Matches at least 4 digits but no more than 8 digits.
- Add a **title** attribute to add an error message and it is displayed when the cursor moves over the respective form element.

What more could go on a form?

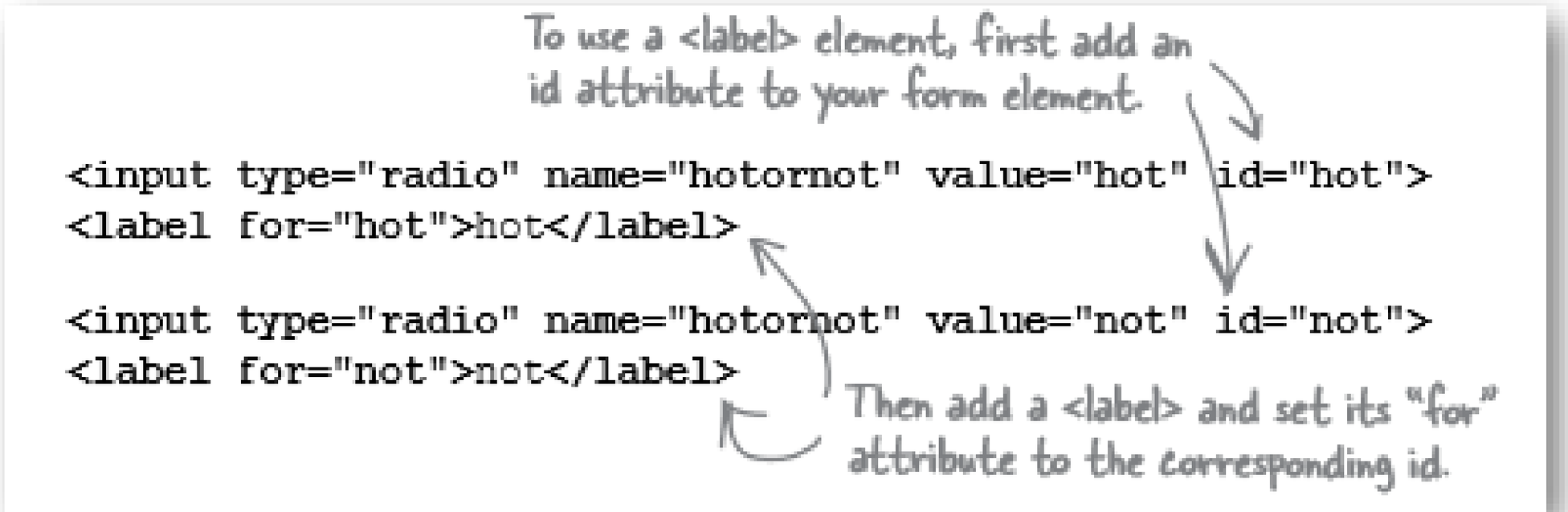



```
<input type=”reset”>
```

- The submit button will send the fields to the server script for processing, whereas the reset button will return all fields to their default values.

Accessibility

- So far we have been labelling our form using text. We should really be using the **<label>** element to mark up these labels.
- The **<label>** element provides further information about the structure of your page, allows you to style your labels using CSS more easily, and helps screen readers for the visually impaired to correctly identify form elements.
- Labels don't look any different from normal text. But they can make a big difference when it comes to accessibility.



Form layout

- Add the **form** class to the <form> tag.
- Enclose each combined/related label and text input with an element (such as a) with the field class.

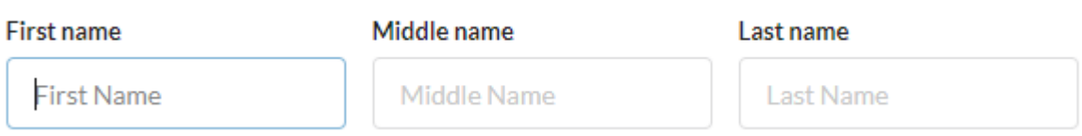
```
<div class="field">
  <label for="firstname">First Name</label>
  <input type="text" id="firstname"
    name="firstname">
</div>
```

Name

Form layout

- Fields can be grouped together by enclosing the fields inside an element (such as a) with the class **fields**.

```
<div class="fields">
  <div class="field">
    <label for = "fname">First name</label>
    <input type="text" id = "fname" name = "fname" placeholder="First Name" autofocus>
  </div>
  <div class="field">
    <label for = "mname">Middle name</label>
    <input type="text" id = "mname" name = "mname"  placeholder="Middle Name">
  </div>
  <div class="field">
    <label for = "lname">Last name</label>
    <input type="text" id = "lname" name = "lname"  placeholder="Last Name">
  </div>
</div>
```

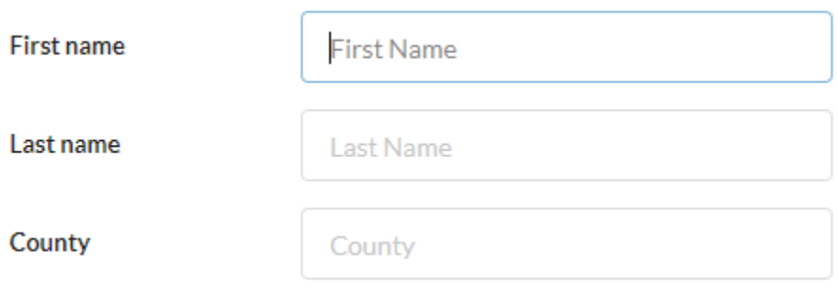


Form layout

- A horizontal form is a form with the label and input element on the same row/line.
- You can use the **inline fields** or **inline field** classes to create horizontal forms.

Form layout

```
<div class="inline field">
  <label for = "fname" class = "two wide field">First name</label>
  <input type="text" id = "fname" name = "fname" placeholder="First Name" class = "four wide field" autofocus>
</div>
<div class="inline field">
  <label  for = "lname" class = "two wide field">Last name</label>
  <input type="text" id = "lname" name = "lname" placeholder="Last Name" class = "four wide field">
</div>
<div class="inline field">
  <label  for = "county" class = "two wide field">County</label>
  <input type="text" id = "county" name = "county" placeholder="County" class = "four wide field">
</div>
```



Form layout

- To output checkboxes and radio buttons in a readable manner you can use the **checkbox** class, and add other classes to it as necessary.

```
<div class="ui checkbox">
  <input type="checkbox" id = "TandC"  name ="TandC">
  <label for = "TandC">I agree to the Terms and Conditions</label>
</div>
```

I agree to the Terms and Conditions

- Toggle I agree to the Terms and Conditions
- Slider I agree to the Terms and Conditions

Form layout

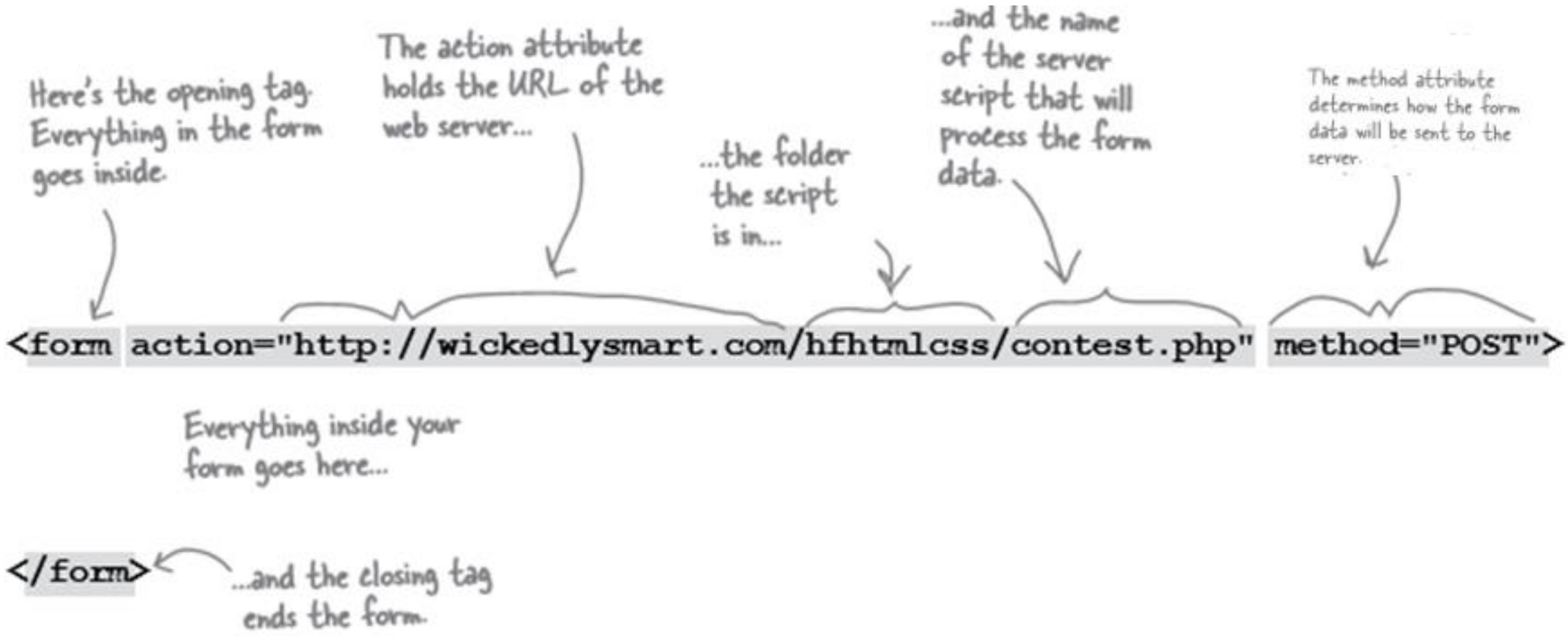
- Radio

```
<div class="grouped fields">
  <label for="fruit">Select your favourite fruit:</label>
  <div class="field">
   <div class="ui radio checkbox">
    <input type="radio" name="fruit" checked="checked" id = "apple" value = "apple">
    <label for = "apple" >Apples</label>
   </div>
  </div>
  <div class="field">
   <div class="ui radio checkbox">
    <input type="radio" name="fruit"  id = "orange" value = "orange">
    <label for = "orange">Oranges</label>
   </div>
  </div>
  ...
</div>
```

Select your favourite fruit:

- (•) Apples
- ( ) Oranges
- ( ) Pears
- ( ) Grapefruit

More on how the <form> element works



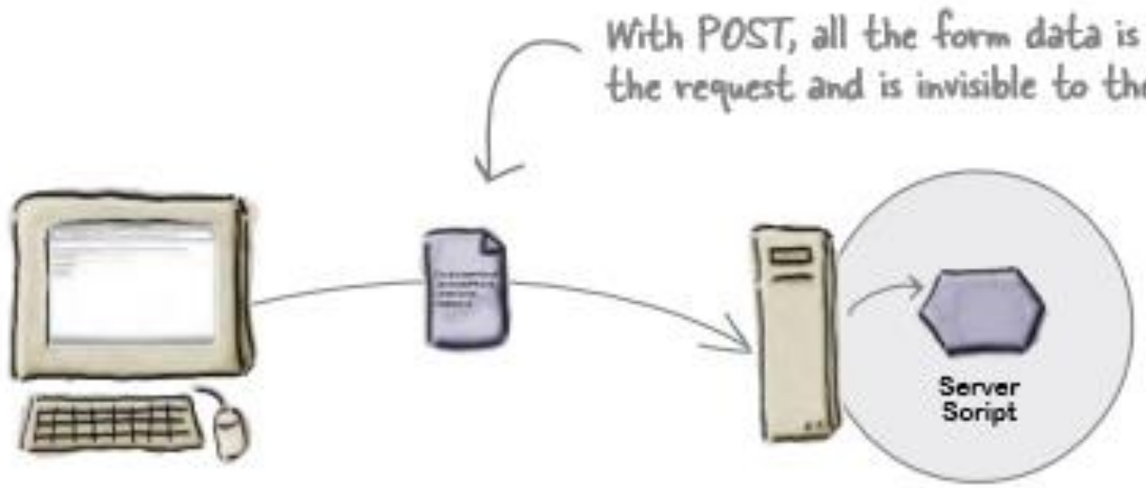
Sending the data to the server

There are two primary methods the browser uses: POST and GET.

POST and **GET** accomplish the same thing—getting your form data from the browser to a server—but in two different ways. **POST** packages up your form variables and sends them behind the scenes to your server, while **GET** also packages up your form variables, but appends them on the end of the URL before it sends a request to the server.

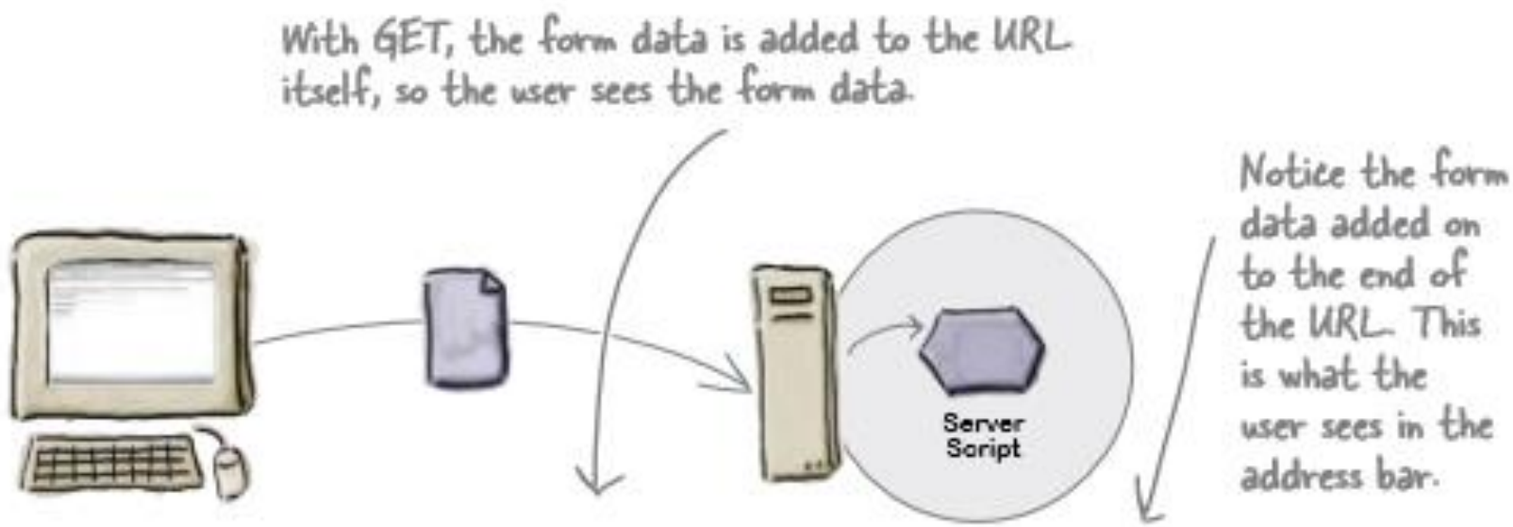
Sending the data to the server

POST




```
http://wickedlysmart.com/hfhtmlcss/contest.php
```

GET




```
http://wickedlysmart.com/hfhtmlcss/contest.php?firstname=buckaroo&lastname=banzai
```