**The INPUT Tag: Radio Buttons, Checkboxes, and Submit Buttons**

Form elements, such as radio buttons, checkboxes, and submit buttons, are created and inserted into a FORM using the INPUT tag. The INPUT tag doesn't require a closing tag.

Tag: **INPUT**  
Description: the INPUT tag creates form elements within the confines of a FORM tag. Note: INPUT tags may ONLY be inserted as child elements of a FORM tag; they won't work outside of a FORM tag.

INPUT tags require different attributes, depending on what sort of form element you're trying to create. Every INPUT tag, however, REQUIRES the TYPE attribute, which tells the browser what type of form element you want.

Tag: **INPUT**  
Attribute: **TYPE**  
Value: **text** (default), **radio**, **checkbox**, **password**, **submit**, **image**, **reset**, **button**, **hidden**, **file**  
Description: the TYPE attribute sets the INPUT tag to become the desired type of form element.  
Example (abbreviated): **<input type="submit" />** (makes a submit button)

There are two other attributes of INPUT that are GENERALLY, though not universally, required: NAME and VALUE.

Tag: **INPUT**  
Attribute: **NAME**  
Value: any name, NO SPACES IN THE NAME (and following the naming convention rules laid out in previous modules)  
Description: the NAME attribute of the INPUT tag identifies that INPUT for analysis in a script. Most INPUT tags require the NAME attribute.

Tag: **INPUT**  
Attribute: **VALUE**  
Value: any plain text (alphanumeric) information; spaces are permissible although not always recommended.  
Description: the VALUE attribute of the INPUT tag allows you to assign some piece of information to a named INPUT; this information can be retrieved by a script upon submission of the form. (Note that HTML5 does not require the ending / for the input statement).

Example (INPUT tag only):

<input type="checkbox" name="pets" value="dog" />

The above example creates a checkbox named "pets" whose value is "dog". Upon submission of the form, a CGI or JavaScript script could analyze whether or not the "pets" checkbox was checked by the user; if the "pets" checkbox was checked, the script could extract the value "dog" from that checkbox.

**Checkboxes**

Let's take a closer look at the "pets" checkbox from the previous example. Here is a complete HTML page, including a FORM tag, which uses the "pets" checkbox code:

<html>

<head>

<title>Example Form</title>

</head>

<body>

<form>

<input type="checkbox" name="pets" value="dog" />

</form>

</body>

</html>

Here it is displayed.

Top of Form



Bottom of Form

Notice how there is NOTHING visible but the checkbox? You can't see the name "pets" or the value "dog"; these things are invisible to the user. The INPUT tag creates ONLY the form element itself; it does not create any context for the form element. Context for a form element must be created using ordinary HTML code and text.

Here is an example of the plainest possible HTML formatting for a set of checkboxes:

<html>

<head>

<title>Example Form</title>

</head>

<body>

<form>

<p>Choose one or more of your favorite pets: <br />

<input type="checkbox" name="pets" value="dog" /> Dog<br />

<input type="checkbox" name="pets" value="cat" /> Cat<br />

<input type="checkbox" name="pets" value="bird" /> Bird

</p>

</form>

</body>

</html>

Here it is displayed.

Choose one or more of your favorite pets:  
 Dog  
 Cat  
 Bird

The use of the P tag to mark the entire set of inputs, including their opening line, is very standard. Notice how all of the checkboxes in this group have the same NAME, while the VALUEs are all different? You will usually name all checkboxes in a single group the same thing, or they will not operate properly when being analyzed by most scripts. An exception to this rule is when using a php script with checkboxes. There must be NOTHING else on the HTML page which uses that NAME, however; each named element on an HTML page must have a UNIQUE name (a named element being anything using the NAME attribute, regardless of the tag).

**Submit Buttons**

As mentioned earlier, a form isn't a form without a submit button to allow the user to submit the form for processing by a script. The submit button is created using the INPUT tag, **type="submit"**.

Example (INPUT only):

<input type="submit" value="Submit Me, Please" />

The VALUE of a submit button is the text which appears on the button itself. Here is the above submit button in context with the set of checkboxes from the previous example:

<html>

<head>

<title>Example Form</title>

</head>

<body>

<form>

<p>Choose one or more of your favorite pets: <br />

<input type="checkbox" name="pets" value="dog" /> Dog<br />

<input type="checkbox" name="pets" value="cat" /> Cat<br />

<input type="checkbox" name="pets" value="bird" /> Bird

</p>

<input type="submit" value="Submit Me, Please" />

</form>

</body>

</html>

Here it is displayed.

Choose one or more of your favorite pets:  
 Dog  
 Cat  
 Bird



Again, notice how the submit INPUT tag's VALUE is the text appearing in the button itself?

**Radio Buttons**

Radio buttons are coded identically to checkboxes, with one exception: the TYPE attribute must be set equal to **"radio"**.

Example:

<html>

<head>

<title>Example Form</title>

</head>

<body>

<form>

<p>Choose your favorite type of pet:<br />

<input type="radio" name="pets" value="dog" />Dog<br />

<input type="radio" name="pets" value="cat" />Cat<br />

<input type="radio" name="pets" value="bird" />Bird /

</p>

<input type="submit" value="Submit Me, Please" />

</form>

</body>

</html>

Here it is displayed.

Choose your favorite type of pet:  
 Dog  
 Cat  
 Bird



Radio buttons behave differently from checkboxes in one important respect: the user may check only ONE from a group of radio buttons, whereas they may check as many as they like from a group of checkboxes. That is the essential nature of radio buttons.

It is important that all radio buttons in a group have the same NAME, or they will absolutely NOT function properly for the user. In the following example, I have NAMED each of my radio buttons something different; witness the sad result for yourself.   
  
Choose your favorite type of pet:  
 Dog  
 Cat  
 Bird  
  
  
  
Once you've checked all the buttons once, you can't UNCHECK them, nor can you get them to behave like proper radio buttons.

Here is another example which you may find illuminating. I have created two sets of radio buttons, one named **"pets"** and one named **"colors"**. In the **"colors"** set of radio buttons, I have "accidentally" named one of the inputs **"pets"** by mistake. Here is the code:

<html>

<head>

<title>Example Form</title>

</head>

<body>

<form>

<p>Choose your favorite type of pet:<br />

<input type="radio" name="pets" value="dog" /> Dog<br />

<input type="radio" name="pets" value="cat" /> Cat<br />

<input type="radio" name="pets" value="bird" /> Bird

</p>

<p>Choose your favorite color:<br />

<input type="radio" name="colors" value="red" /> Red<br />

<input type="radio" name="colors" value="green" /> Green<br />

<input type="radio" name="pets" value="blue" /> Blue </p>

<input type="submit" value="Submit Me, Please" />

</form>

</body>

</html>

And here is the above example displayed.

Choose your favorite type of pet:  
 Dog  
 Cat  
 Bird

Choose your favorite color:  
 Red  
 Green  
 Blue

   
  
If you click on the "pets" radio buttons, everything seems fine. But if you click on the "blue" radio button in the "colors" group (which was named "pets" by mistake), it operates in tandem with the "pets" radio buttons rather than the "colors" radio buttons, causing the "pets" buttons to lose their checked choice.

It doesn't matter how far apart these INPUT tags are in the FORM, or where they're located or what they're grouped with; if they have the same NAME attribute, they are going to operate together as a group, regardless of their context. Since you don't want this sort of bizarre behavior to occur, you need to be very careful when coding your radio buttons and checkboxes, making CERTAIN that all of the NAMEs in a given group are indeed identical, and that no other group contains a repeat of that NAME.

Here is the corrected example:

<html>

<head>

<title>Example Form</title>

</head>

<body>

<form>

<p>Choose your favorite type of pet:<br />

<input type="radio" name="pets" value="dog" /> Dog<br />

<input type="radio" name="pets" value="cat" /> Cat<br />

<input type="radio" name="pets" value="bird" /> Bird

</p>

<p>Choose your favorite color:<br />

<input type="radio" name="colors" value="red" /> Red<br />

<input type="radio" name="colors" value="green" /> Green<br />

<input type="radio" name="colors" value="blue" /> Blue

</p>

<input type="submit" value="Submit Me, Please" />

</form>

</body>

</html>

And here is that corrected example displayed.

Choose your favorite type of pet:  
 Dog  
 Cat  
 Bird

Choose your favorite color:  
 Red  
 Green  
 Blue



Students always ask me, "Can I make radio buttons behave like checkboxes, or checkboxes behave like radio buttons?" The answer is NO, you can not; radio buttons behave and look like radio buttons and may not be anything else, and the same holds true for checkboxes. These form elements and their behaviors are built into the browsers as common RESOURCES for web programmers, so that there are consistent-looking-and-acting interface elements available to users, and they may not be altered arbitrarily.

The second question students ask me is "Can I change the appearance of these form elements, or use my own images in place of the boring buttons?" The answer to that question is NO, not significantly. Some form elements may be altered in appearance SLIGHTLY using CSS, but not in any substantive way, and it is physically impossible to substitute custom graphics for any form element except for the submit button (which we'll talk about later).