


# Basic HTML Form Elements



# No One Likes Forms

“I’m bored, so I think I will go  
fill out some forms.”

– Said no one, ever

No one enjoys making forms, or filling them out, for that matter. However they remain one of the main pieces of interactivity on the web. Poorly designed forms can lead to very frustrated users.

# Starting a Form

Forms are one of the most important ways users interact with content on the web. We will look at the various important form elements and discuss how they work.

A form must be surrounded by the `<form>` element tags.

The form tag usually is accompanied with these two attributes. The method attribute tells the form how to send the data from the form, the action says where to send it. You could include other attributes such as ID, class or name.

```
<body>
```

```
<form method="post" action="processor.php">
```

```
</form>
```

```
</body>
```

# Adding a Text Field

The most common element found in a form is the text field. This is created with an `<input>` tag element.

Notice that it is self closing. The type attribute is what makes it a text field.

To get data out of the form, you MUST include a name attribute. This is how the data is going to get to the server.

It is fairly common, but not required that an ID attribute is added. This MUST be unique.

```
<form method="post" action="processor.php">  
    <input type="text" name="fname">  
  
</form>
```

```
<form method="post" action="processor.php">  
    <input type="text" name="fname">  
  
</form>
```

```
<form method="post" action="processor.php">  
    <input type="text" name="fname" id="fname">  
  
</form>
```

# Optional: Add a Value Attribute

The value attribute can be added to put a value in the field when the page is loaded. Other attributes can be added as well, such as class and tabindex.

```
<form method="post" action="processor.php">
```

```
    <input type="text" name="fname" id="fname" value="some value here">
```

```
</form>
```

# Add a Label

To give the input tag a label, you use the label tag. The for attribute in the label tag connects the label to the input tag. The value in the for attribute matches the value in the ID attribute (not the name attribute).

```
<form method="post" action="processor.php">  
  <label for="fname">First Name</label>  
  <input type="text" name="fname" id="fname" value="some value here">  
  
</form>
```

# Alternative Label Markup

It is also possible to do it this way. Notice no for attribute is needed if you put the entire input element inside the label element.

However, doing it this way somewhat limits the styling options.

```
<form method="post" action="processor.php">  
  <label>First Name  
    <input type="text" name="fname" id="fname" value="some value here">  
  </label>  
  
</form>
```

# HTML5 Additional Form Input Types

- color
- date
- datetime-local
- email
- month
- number
- range
- search
- tel
- time
- url
- week

```
<form method="post" action="processor.php">  
    <label for="favcolor">Favorite Color</label>  
    <input type="color" name="favcolor" id="favcolor">  
  
</form>
```

However, these are not supported evenly across all browsers.



# Radio Button Input Type

For radio buttons and checkboxes, it is considered better practice to put the label after the input element.

Notice the name attribute for these two radio buttons match. That makes them a set. Checked makes this one checked by default.

```
<input type="radio" name="spam" id="spam-yes" value="yes">  
<label for="spam-yes">Yes, send me spam!</label>
```

```
<input type="radio" name="spam" id="spam-no" value="no" checked>  
<label for="spam-no">No, don't send me spam!</label>
```

# Use Fieldset to Group Fields

The fieldset tag allows you to group the set of radio buttons together semantically, and the legend element allows you to give that set a caption.

Do You Want Spam?\_\_\_\_\_

☐ Yes, send me spam!

☒ No, don't send me spam!

```
<fieldset>
  <legend>Do You Want Spam?</legend>
  <input type="radio" name="spam" id="spam-yes" value="yes">
  <label for="spam-yes">Yes, send me spam!</label>

  <input type="radio" name="spam" id="spam-no" value="no" checked>
  <label for="spam-no">No, don't send me spam!</label>
</fieldset>
```

# Using the Checkboxes Input Type

Checkboxes are slightly different because the user can select one or many for any group.

This means you have to process checkbox data a little differently from radio button data, where only one value can be selected.

One common way to deal with it is to send the data for checkboxes to the server in an array.

```
<fieldset>
  <legend>Favorite Animals?</legend>
  <input type="checkbox" name="animal[]" id="cat" value="cat">
  <label for="cat">Cat</label>

  <br>

  <input type="checkbox" name="animal[]" id="dog" value="dog">
  <label for="dog">Dog</label>

  <br>

  <input type="checkbox" name="animal[]" id="llama" value="llama">
  <label for="llama">Llama</label>
</fieldset>
```

# Select List Input Type

Select lists work like this. The select element gets the name attribute.

We are still including an ID to work with our label.

The option tag gets a value attribute that will be passed to the processor if that attribute is selected.

```
<label for="cheese">Select Favorite Cheese</label>
<select name="cheese" id="cheese">

    <option value="American">American</option>
    <option value="Swiss">Swiss</option>
    <option value="Brie">Brie</option>

</select>
```

# Using Textarea Elements

Textarea elements are different from input elements. First, they have an opening and closing tags, rather than self closing like the input element.

Secondly, they need the cols and rows attribute, which is a bit unusual. This can be overridden by setting a width and height in CSS, but these two attributes are considered required.

Of course the ID is needed and name attributes so that the label works and the data can be retrieved.

```
<label for="comments">Comments?</label>  
<textarea id="comments" name="comments" rows="10" cols="50"></textarea>
```

# The Submit Input Type

The final piece we will look at now is the submit button. It uses the same `<input>` element we use for text and other types of input. The name attribute can be important because we may need it when we process the form.

The value is the the text that shows up inside the button. You can add class, id and other attributes to the submit type as well.

If you want to use an image as a submit button, you will use `<input type="image">` instead.

```
<input type="submit" name="send" value="send it!">
```

# Form Styling

You can spend A LOT of time styling forms. Even a simple form can take a while to get right. Here are a few basic styles for this one.

```
<style>

    label, input {
        display: block;
    }

    input {
        margin-bottom: 20px;
    }

    fieldset input, fieldset label {
        display: inline;
    }

    fieldset label {
        margin-bottom: 20px;
    }

</style>
```

# HTML Form Validation

Some basic form validation has been making its way into HTML. You can do some basic front end validation this way, but not all browsers support it to the same extent.

If you want to take full control of front end form validation, that is where JavaScript comes to the rescue.

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