

# FORM BASICS

---

## FORM BASICS

---

# LEARNING OBJECTIVES

- › Be able to differentiate the different types of inputs and why/where we would use each
- › Explain how to group elements by name.
- › Explain how to connect a label with an input field

---

**FORM BASICS**

---

# FORM ELEMENT

---

# FORMS

---

<https://codepen.io/melodyserra/pen/RLxQZo>

---

# FORMS

---

All form elements used to collect user input go in the form element.

```
<form>
```

```
  <!--Data collection elements go here-->
```

```
</form>
```

---

## FORM BASICS

---

# BUTTONS

---

# FORM BUTTONS

---

Sign Up

---

# SUBMITTING A FORM

---

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



**TYPE**

Specifies that this  
button should  
**submit** a form

`<form>`

`<input type="text" name="full-name" placeholder="Enter your full name">`

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

`</form>`



---

## FORM BASICS

---

# TEXT INPUTS

---

# GETTING INFO — INPUTS

---

First Name

Last Name

Your Email

Your Password

---

## GETTING INFO — INPUTS

---

- How we get content from users.
- Should go inside our form tags.

<form>

<input type="text">

<button type="submit">Continue</button>

</form>

← **NO CLOSING TAG**

---

# INPUTS — ATTRIBUTES

---

`<input type="text">`



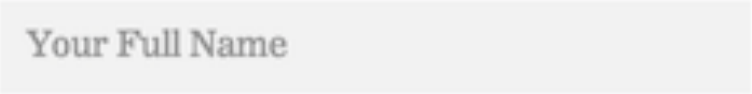
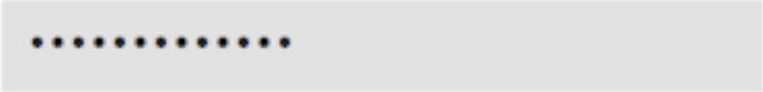
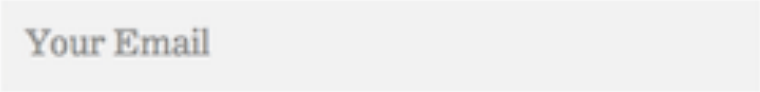

## TYPE

Type of data that is  
being input (text,  
email, password,  
etc.)

---

# TEXT INPUTS — DESCRIPTIONS

---

FIELD TYPE	HTML	WIDGET
<i>Plain Text</i>	<input type="text">	
<i>Password</i>	<input type="password">	
<i>Email</i>	<input type="email">	
<i>Text Area</i>	<textarea></textarea>	

---

# INPUTS — ATTRIBUTES

---

First Name

```
<input type="text" placeholder="First Name">
```

## PLACEHOLDER

Prompt to let user  
know what to enter  
into the field

---

## FORM BASICS

---

# TEXTAREAS

---

# TEXTAREAS — ATTRIBUTES

---

## PLACEHOLDER

Prompt to let user know  
what to enter into the  
field

\_\_\_\_\_

<textarea placeholder="Message"></textarea>

CLOSING TAG



---

## FORM BASICS

---

# STYLING INPUTS AND BUTTONS

---

# STYLING INPUTS

---

You can use the name of the element to add styles:

```
input {  
  border: 1px solid grey;  
}
```

```
textarea {  
  padding: 20px;  
}
```

```
button {  
  padding: 20px;  
}
```

---

## STYLING INPUTS

---

Question: By default inputs and textareas have a **display** of **inline-block** so they will all be on one line. If we wanted each input/textarea to start on a new line, what could we do?

---

## REVIEW — BLOCK

---

Make an inline element act like a block-level element:

### BEFORE:

[Link](#)

[Link](#)

[Link](#)

```
a {  
  display: block;  
}
```

[Link](#)

[Link](#)

[Link](#)

- Elements will stack on top of each other
- We can add all dimensions (width, height, padding, margin)

---

# STYLING PLACEHOLDER TEXT

---

We need separate rules for each browser to style placeholder text for different browsers. The one place where we can't use a comma-separated list (unfortunately!)

```
::-webkit-input-placeholder { /* WebKit, Blink, Edge */  
    color: #dfdfdf;  
}  
:-moz-placeholder { /* Mozilla Firefox 4 to 18 */  
    color: #dfdfdf;  
    opacity: 1;  
}  
::-moz-placeholder { /* Mozilla Firefox 19+ */  
    color: #dfdfdf;  
    opacity: 1;  
}  
:-ms-input-placeholder { /* Internet Explorer 10-11 */  
    color: #dfdfdf;  
}
```

[Read more here](#)

# ACTIVITY

---



## EXERCISE

### **KEY OBJECTIVE**

- ▶ Identify input types, add styles to a form

### **LOCATION**

- ▶ Starter Code > contact\_form

### **TIMING**

*15 min*

1. Review contact\_form.png
2. Write the html for the contact form
3. Style the form using contact\_form.png as a guide.

### **BONUS**

- ▶ Add a hover effect to the button! See the gif in the contact\_form folder for an example.
- ▶ Use a transition so that the new button color animates in.

---

## FORM BASICS

---

# LABELS

---

# LABELS

---

Email

Your Email



---

# LABELS

---

We can add labels for each form field like so:

**FOR**

Which form field is  
this label for?



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

```
<input type="text" id="name" name="name" placeholder="Enter your full name">
```



**ID**

Unique identifier,  
will match the label

---

## FORM BASICS

---

# CHECKBOXES AND RADIOS

---

# INPUTS — MAKING CHOICES

---

☐ I am a U.S. citizen or Permanent Resident.

- ☐ Small
- ☐ Medium
- ☐ Large

---

# INPUTS — MAKING CHOICES

---

FIELD TYPE	HTML	WIDGET
<i>Checkbox</i>	<input type="checkbox">	<input type="checkbox"/> Remember me
<i>Radio</i>	<input type="radio">	<input type="radio"/> Small <input type="radio"/> Medium <input type="radio"/> Large

# CHECKBOXES

---

☐ Remember me

`<input type="checkbox" id="save-login" name="save-login">`

**TYPE**

Widget to display  
(checkbox)

**ID**

Unique identifier,  
will match the label

---

## ALL TOGETHER NOW! CHECKBOXES

---

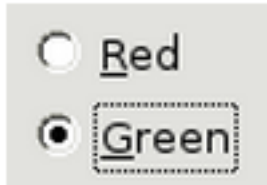
Be sure to include a label! Otherwise the user won't know what the random checkbox is for.

```
<input type="checkbox" id="save-login" name="save-login">  
<label for="save-login">Remember Me</label>
```

☐ Remember me

# RADIO BUTTONS

---



## NAME

Used to connect  
radio buttons

`<input type="radio" name="color" id="red" value="red">`

## TYPE

Widget to display  
(radio)

## ID

Use this to connect  
a radio to its label.

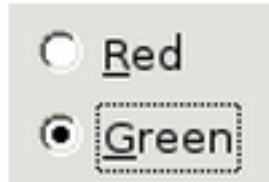
---

# RADIO BUTTONS

---

Radios allow our users to select one option.

We can connect radios so that our user can only select one option by giving them the same **name** attribute



```
<input type="radio" name="color" value="red" id="red">
```

```
<input type="radio" name="color" value="green" id="green">
```



---

# ALL TOGETHER NOW! CHECKBOXES

---

Be sure to include a label for each radio. Otherwise the user won't know what they are selecting!

```
<input type="radio" name="color" value="red" id="red">  
<label for="red">Red</label>
```



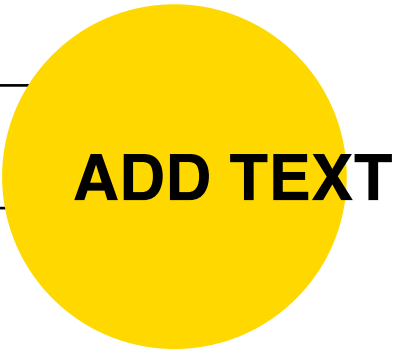
---

## FORM BASICS

---

# SELECT MENUS

# INPUTS — MAKING CHOICES



FIELD TYPE	HTML	ATTRIBUTES	WIDGET
<i>Select</i>	<pre>&lt;select&gt;   &lt;!-- options go here --&gt; &lt;/select&gt;</pre>	<i>id</i>	<p>Where are you thinking of taking this course?</p> <p>Chicago</p>
<i>Option</i>	<pre>&lt;option&gt;Friend&lt;/option&gt;</pre>	<i>value</i>	<div><div>Atlanta</div><div>Austin</div><div>Beeton</div><div>✓ Chicago</div><div>Hong Kong</div><div>London</div><div>Los Angeles</div><div>Melbourne</div><div>New York City</div><div>San Francisco</div><div>Seattle</div><div>Sydney</div><div>Washington D.C.</div></div>

---

## STYLING INPUTS — PART 2

---

Want an easy way to style specific inputs in your CSS without adding a bunch of ids or classes to each one? The attribute selector is just the ticket!

```
input[type="text"] {  
  border: 1px solid grey;  
}
```

```
input[type="password"] {  
  padding: 20px;  
}
```

---

# ACTIVITY

---

Structure questions:

1. Identify types of inputs (text, email, password, checkbox, radios)
2. Identify any textarea elements
3. Identify any select menus. How should those be structured (how can we get a dropdown with choices?)
4. How can we connect any labels and their corresponding fields?
5. How can we connect the radio inputs?

Style questions:

1. Talk through styles. Are most elements on their own line? The same line? What are the exceptions?
2. How can we center the form against the blue background?

# ACTIVITY

---



## EXERCISE

### KEY OBJECTIVE

- Identify input types, add styles to a form

### LOCATION

- Starter Code > Application form

### TIMING

*30 mins*

1. Write HTML for the form
2. Style the form with CSS. Focus on getting the form centered and getting the information on the right rows, and then add other styles if you have time.

### BONUS

- Google and integrate the **fieldset** element
- Add hover effects (and transitions) to the button

---

**FORM BASICS**

---


# HOW FORMS WORK

---

# FORMS

---

Forms are used to get data from users.

 **GENERAL ASSEMBLY**

Sign in

FRONT-END WEB DEVELOPMENT

**APPLY NOW**

Where are you thinking of taking this course?

**CONTINUE TO APPLICATION**

Fill out some basic information and complete the following application to be considered for the course.




---

# FORMS

---

1. The user fills out the form and presses the submit button (or hits the return key)



FRONT-END WEB DEVELOPMENT

## APPLY NOW

Where are you thinking of taking this course?

New York City

CONTINUE TO APPLICATION

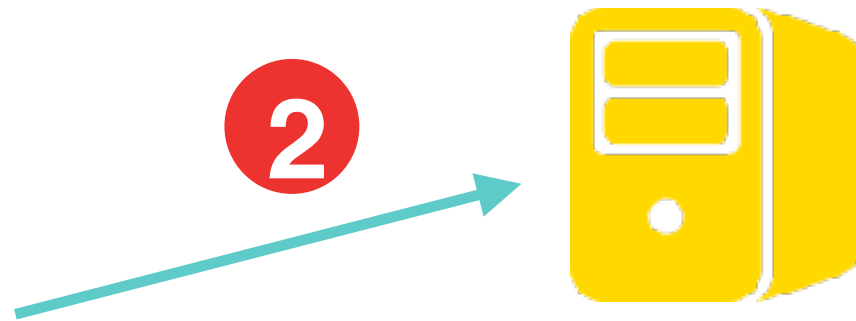
Fill out some basic information and complete the following application to be considered for the course.

---

# FORMS

---

- 
2. The name of each form field is sent to the server along with the value the user entered or selected



**FRONT-END WEB DEVELOPMENT**  
**APPLY NOW**

Where are you thinking of taking this course?

Fill out some basic information and complete the following application to be considered for the course.

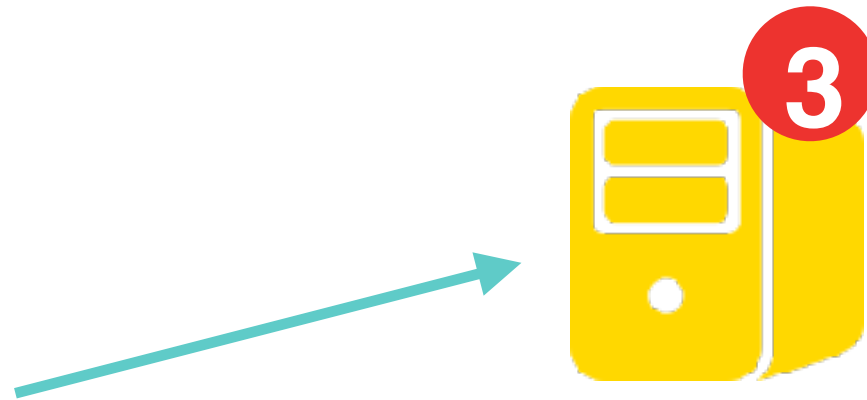
**CONTINUE TO APPLICATION**

---

# FORMS

---

3. The server processes the data using a language such as PHP, C# or Java. It may also store the information in a database



**FRONT-END WEB DEVELOPMENT**  
**APPLY NOW**

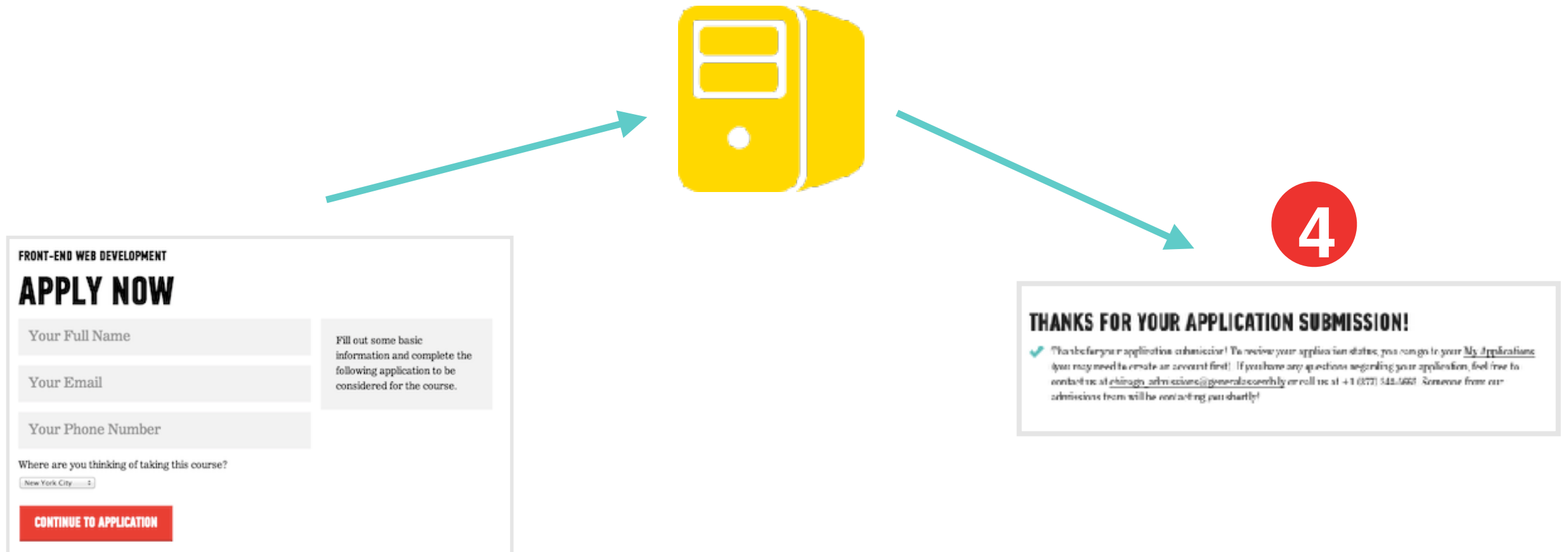
Where are you thinking of taking this course?

Fill out some basic information and complete the following application to be considered for the course.

**CONTINUE TO APPLICATION**

# FORMS

- The server creates a new page to send back to the browser based on the information received.



---

## FORM BASICS

---

# SENDING FORM DATA

---

# FORM ATTRIBUTES

---

When data is sent to a server, the form will have two attributes:

**ACTION — WHERE TO SEND**

**DATA (URL)**

**METHOD — HOW TO SEND IT**



```
<form action="http://www.example.com/login.php" method="post">  
  <!--Data collection elements go here-->  
</form>
```

Since we are not sending data to a server today, you can skip adding these for now.

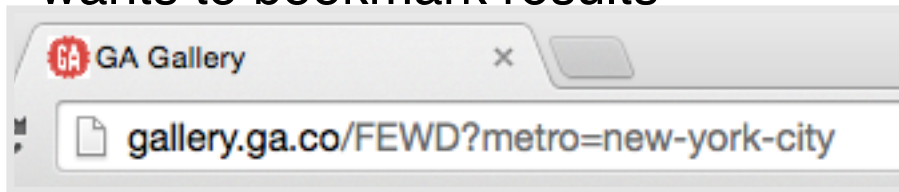
---

# FORMS — METHODS

---

## GET

- Short forms (such as search fields)
- Appended to URL in name/value pairs
- Never use for sensitive info!!!
- Useful for form submissions when user wants to bookmark results



## POST

- Data not shown in URL
- For sensitive data
- No size limitations

## METHOD — HOW TO SEND IT

```
<form action="http://www.example.com/login.php" method="post">  
  <!--Data collection elements go here-->  
</form>
```

---

# HOW DATA IS SENT AND SAVED

---

Information is sent from the browser to the server using name/value pairs.

**username=billyj@gmail.com**  
**save-login=true**  
**size=small**  
**city=chicago**



# INPUTS

---

- For any inputs where the user is entering text, the name will come from the name attribute and the value will be what the user typed in.

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

**NAME**

holden@gmail.com|

**VALUE (WHAT THE USER ENTERED)**

[username=holden@gmail.com](#)

---

# RADIO BUTTONS

---

## NAME

The key used to describe this data  
when it is sent to a server

`<input type="radio" name="color" value="red">`  
`<input type="radio" name="color" value="green">`

## VALUE

The value that will be sent to the server if this radio  
is selected (since the user is not typing a value).

`color=red`

---

# CHECKBOXES

---

## NAME

The key used to describe this data  
when it is sent to a server

  
<input type="checkbox" name="save-login">

If checkbox is checked: **save-login=true**

If checkbox is not checked: **save-login=false**

---

# ALL TOGETHER NOW! SELECT AND OPTIONS

---

```
<select name="user-location">
  <option value="atlanta">Atlanta</option>
  <option value="boston">Boston</option>
  <option value="chicago">Chicago</option>
</select>
```

Where are you thinking of taking this course?

Chicago

Atlanta  
Austin  
Boston  
✓ Chicago  
Hong Kong  
London  
Los Angeles  
Melbourne  
New York City  
San Francisco  
Seattle  
Sydney  
Washington D.C.

**user-location=atlanta**

---

## FORM BASICS

---

# LEARNING OBJECTIVES

- Be able to differentiate the different types of inputs and why/where we would use each
- Explain how to group elements by name.
- Explain how to connect a label with an input field

---

**HTML BASICS**

---

# EXIT TICKETS