3.2 Forms

<form> tag

The ***<form>*** tag allows the web browser to submit information from the user to the server. The `<form>` tag has two primary attributes:

1. The ***action attribute*** indicates the URL where the form data should be sent. Typically the URL uses HTTPS so the form data is encrypted.
2. The ***method attribute*** indicates the HTTP request type the browser will use to communicate with the server. The method is either GET or POST. GET is the default method if no method is specified.

Figure 3.2.1: Partial HTML form sending data to Twitter using a secure HTTP POST request.

```
<form action="https://twitter.com/status"
method="POST">
  ...
</form>
```

Feedback?

PARTICIPATION ACTIVITY 3.2.1: Submitting form data to a server.

Start 2x speed

First Name: Sarah

Last Name: Connor

Submit

Database on server

First name	Last name	...
John Jane Buck Sarah	Doe Doe Rogers Connor	... ...

Captions

1. The user enters information in the form.
2. The user clicks the submit button.
3. The browser collects the data and sends the data to the server.

Feedback?

PARTICIPATION ACTIVITY 3.2.2: <form> tags.

Refer to the HTML below.

```
<form action="https://google.com/">
</form>
```

1) To which server will the browser send the form data?

Check Show answer

Answer

google.com

The action attribute indicates which server the browser contacts.

2) Which method will the browser use to communicate with the server?

Check Show answer

Answer

GET

When the method attribute is not set or invalid, the browser by default uses an HTTP GET request to communicate with the server.

Feedback?

GET method

The ***GET method*** is a technique used by a web browser to submit information to a web server by altering the URL of the HTTP request. When a user clicks the submit button in a form that uses the GET method, the browser performs the following steps:

1. Collect all data from the form fields into a query string. The ***query string*** is a set of name=value pairs separated by the ampersand character (&). Each name is specified as an attribute of the HTML field, and the value is the user-entered data. Ex: The first and last field names and values in the animation below become the string: `first=Rick&last=Deckard`
2. Create a URL with the server page and name=value pairs. The URL is composed of the action attribute specified in the form, the question mark character (?), and the query string. Ex: `http://example.com/apply?first=Rick&last=Deckard`
3. Use the newly created URL to create and send an HTTP GET request.

4. Display or update the webpage using the HTTP response received from the server.

PARTICIPATION ACTIVITY 3.2.3: Using the GET method to submit form data to a server.

Start 2x speed

Thank you, Rick Deckard!

Web server

example.com
GET request

https://example.com/apply?first=Rick&last=Deckard

```
<body>
  <form action="https://example.com/apply">
    <p>
      <label for="first">First Name:</label>
      <input type="text" name="first" id="first">
    </p>
    <p>
      <label for="last">Last Name:</label>
      <input type="text" name="last" id="last">
    </p>
    <input type="submit">
  </form>
</body>
```

Captions

1. The user enters information in the form and clicks the submit button.
2. The browser creates a query string of name=value pairs for each form field, separated by &.
3. The query string is appended to the URL from the form action.
4. The browser uses the new URL to submit the form data to the web server.
5. The web server responds with a new webpage.

Feedback?

Warning

Because submitting form data with the GET method places the form data in the URL, the data is visible to anyone who sees the URL. The GET method should not be used to submit private information like phone numbers, credit card information, etc.

PARTICIPATION ACTIVITY 3.2.4: GET method.

Refer to the form below.

```
<form method="GET" action="http://example.org/">
   <input type="text" name="item">
   <input type="text" name="price">
   <input type="submit">
</form>
```

1) What is the query string if the user submits "bananas" and "2.50" in the text boxes?

- `bananas=2.50`
- `method=bananas&action=2.50`
- `item=bananas&price=2.50` (selected)

Correct

Two name=value pairs are created from the text box inputs and separated by `&`.

2) What URL does the browser request if the user submits "bananas" and "2.50" in the text boxes?

- http://example.org/
- http://example.org/item=bananas&price=2.50
- http://example.org/?item=bananas&price=2.50 (selected)

Correct

The query string appears after the `?` in the URL when the form submits data with the GET method.

Feedback?

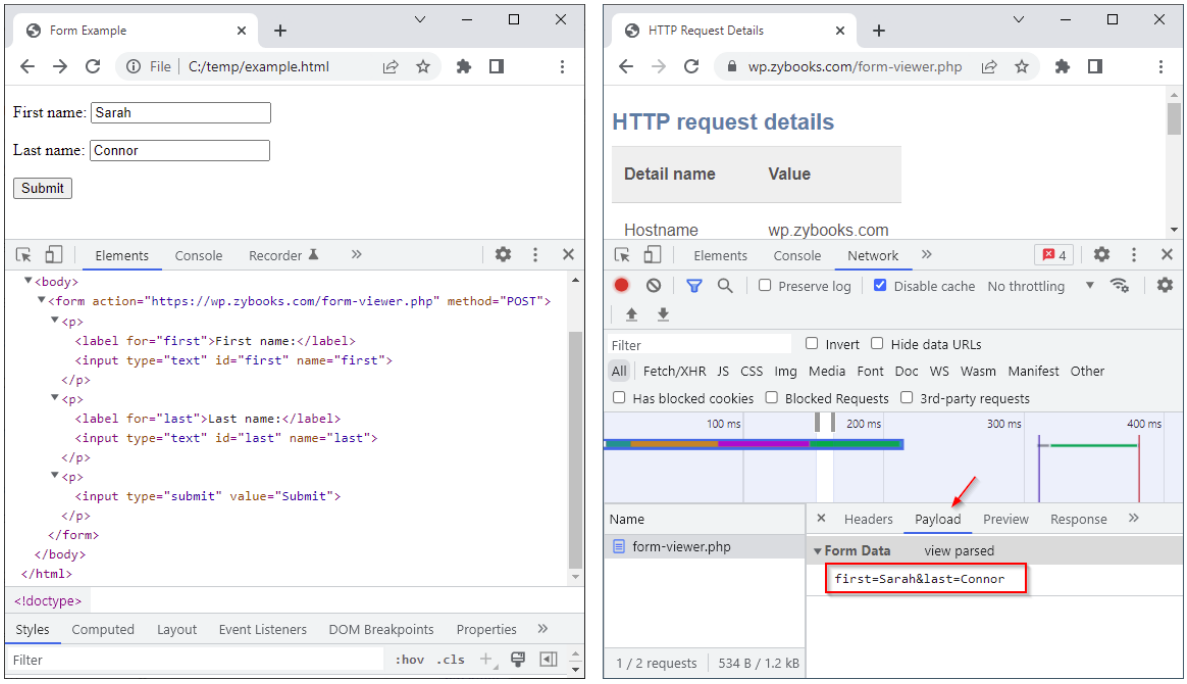
POST method

The ***POST method*** is a technique used by a web browser to submit information to a web server by sending the information in the HTTP request body. When the user clicks the submit button in a form that uses the POST method, the browser performs the following:

1. Create an HTTP POST request using the URL from the form's `action` attribute.
2. Create a query string from the form data. Ex: `first=Sarah&last=Connor`
3. Place the query string in the HTTP request message body, and send the request.
4. Display or update the webpage using the HTTP response received from the server.

The left side of the figure below shows a webpage opened in Chrome. The DevTools show the page's form, which is POSTing to a zyBooks URL. The user entered her name Sarah Connor before pressing Submit. On the right, the user has pressed Submit, and the DevTools show the details of the HTTP request. The Payload tab shows the query string created from the form data.

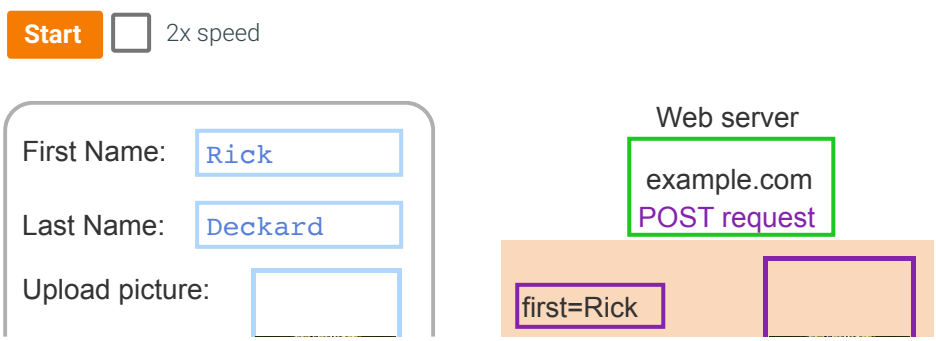
Figure 3.2.2: Chrome DevTools showing form data in POST request.

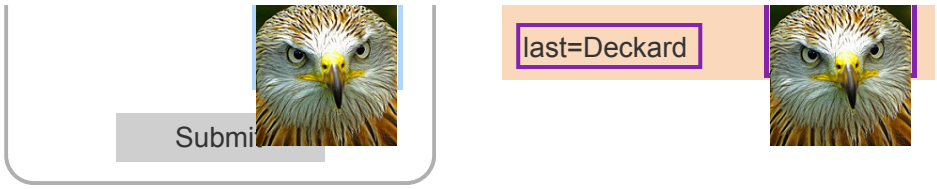


Feedback?

If a form field contains binary data such as an image, the normal format of the query string is not sufficient to encode the binary data. To accommodate binary data, a POST request can be split into multiple parts. The `<form>` tag's ***enctype attribute*** value "multipart/form-data" indicates the web browser should split a POST request into multiple parts, where each input field is sent as a separate part of the HTTP request message.

PARTICIPATION ACTIVITY 3.2.5: Using the POST method to submit form data to a server.






```
<body>
  <form action="https://example.com/apply" method="POST"
        enctype="multipart/form-data">
    <p>
      <label for="first">First Name:</label>
      <input type="text" name="first" id="first">
    </p>
    <p>
      <label for="last">Last Name:</label>
      <input type="text" name="last" id="last">
    </p>
    <p>
      <label for="pic">Upload picture:</label>
      <input type="file" name="pic" id="pic">
    </p>
    <input type="submit">
  </form>
</body>
```

Captions

1. The user enters information in the form, including a picture to upload to the server, and presses Submit.
2. The browser collects the form data into multiple parts and adds each part to the POST request.
3. The POST request is then sent to the server.

Feedback?

PARTICIPATION ACTIVITY 3.2.6: POST method.

Refer to the form below.

```
<form method="POST" action="http://example.org/">
   <input type="text" name="item">
   <input type="text" name="price">
   <input type="submit">
</form>
```

1) What is the query string if the user submits "oranges" and "1.99" in the text boxes?

   - `oranges=1.99`
   - `method=oranges&action=1.99`
   - `item=oranges&price=1.99`

Correct

Two name=value pairs are created from the text box inputs and separated by `&`. The same query string is formed regardless of which method (GET or POST) is used.

2) What URL does the browser request if the user submits "oranges" and "1.99" in

Correct

the text boxes?

- ◉ http://example.org/
- ○ http://example.org/oranges=1.99
- ○ http://example.org?item=oranges&price=1.99

The query string appears in the HTTP request body when the form submits data with the POST method.

3) What changes about the form data when `enctype="multipart/form-data"` is added to the `<form>` tag?

- ○ Nothing changes since binary data is not submitted.
- ◉ Each name/value pair is separated into multiple parts.
- ○ The form data is encrypted.

Correct

The form data is split into multiple parts instead of creating a single query string. The multipart/form-data is only necessary when submitting binary data with `<input type="file">`.

Feedback?

Escaping form data

The &, ?, and = characters have special meaning in a query string. If a user enters characters like &, ?, =, or white space characters like space, newline, or tab, the characters must be ***escaped***, meaning the characters must be transformed into other representations. The browser rules for escaping form data are as follows:

- Space → +
- All other reserved characters, newline, and tab → %XX where XX is the ASCII hex value of the character

Ex: If "1 + 2 = ?" is entered into a textbox, the browser escapes the values producing "1+%2B+2+%3D+%3F". 2B is the ASCII hex value for "+", 3D is the ASCII value for "=", and 3F is the ASCII value for "?".

The web server ***unescapes*** the form data to determine what the original values are.

Form widgets

A ***widget*** is an interactive component (usually graphical) that the browser uses to interact with a user. Ex: Buttons, drop-down menus, and data entry fields.

The ***<input>*** tag allows the user to enter information into a webpage. The `<input>` tag cannot enclose any additional page content, and thus does not have a closing tag. The `<input>` tag has five primary attributes:

- The ***type attribute*** indicates the widget type. Common types include text, password, submit, and button.
- The ***name attribute*** names the widget and sends the widget's value when the widget's form is submitted.
- The ***id attribute*** is used to give a widget a unique identifier.
- The ***placeholder attribute*** specifies text that first appears in a text widget, typically for giving the user a hint as to the expected value.
- The ***value attribute*** specifies a default value for a widget.

A ***text box*** widget is an `input` element with the `type` attribute of "text" that allows users to enter a single line of text.

The web browser displays a ***submit button*** widget for an `<input>` tag with the `type` attribute of "submit", which sends the associated form's data to the server when clicked. A submit button uses the `value` attribute to specify the button's text.

The HTML below asks for a message to tweet. The text box widget does not use the `value` attribute because no default tweet message makes sense. The submit button does not use the `name` attribute because the submit button's value is not needed by the web server.

Figure 3.2.3: Complete HTML form sending status to Twitter using a secure HTTP POST request.

```
<form action="https://twitter.com/status" method="POST">
  <input type="text" name="status" id="status" placeholder="Your
status">
  <input type="submit" value="Send status">
</form>
```

Feedback?

PARTICIPATION ACTIVITY 3.2.7: <input> attributes.

Match each `<input>` attribute to the corresponding effect.

If unable to drag and drop, refresh the page.

type	Indicates which kind of widget is displayed by the browser. Some examples of the `type` attribute include "radio" for radio buttons, "checkbox" for checkboxes, and "text" for textual input.	Correct
name	Allows the server to identify which data came from which widget. The browser sends the data and name to the server, so the server can associate the data received with a specific form input.	Correct
id	Uniquely identifies the specific input tag to the browser. Often the id and name attributes are the same. However, the id attribute may be different from the name attribute. Ex: The same webpage may have two forms that send data using the same name.	Correct
value	Allows the input to start with a default value. Some inputs may be optional or previously entered by the user, so the value attribute allows the browser to send data from the form without direct user interaction.	Correct
placeholder	Provides a hint to the user about the information being requested. Ex: A widget asking for the user's email address may set the placeholder to "user@email.com".	Correct

Placeholders should not be used to replace labels because screen readers do not always read placeholder text to users.

Reset

Feedback?

Labels and text areas

The ***<label>*** tag displays descriptive text associated with a specific widget. A label has a ***for attribute*** whose value should match the `id` attribute for the widget being labeled. Labels help people using screen readers understand what input is expected.

Figure 3.2.4: HTML for a label associated with a text box.

```
<label for="username">Username:
</label>
<input type="text" id="username">
```

Username:

Feedback?

A ***text area*** widget is an input element specified by ***<textarea>*** opening and closing tags that allows users to enter multiple lines of text. A `<textarea>` tag has optional ***rows*** and ***cols attributes*** to specify the initial size of the text area.

Figure 3.2.5: HTML for a textarea.

```
<textarea name="summary" rows="4" cols="50">To summarize...
</textarea>
```

To summarize...

Feedback?

PARTICIPATION ACTIVITY 3.2.8: Text inputs.

The following HTML form contains a text box, text area, and submit button. Pressing the submit button submits the form data to form-viewer.php, which displays the submitted form data.

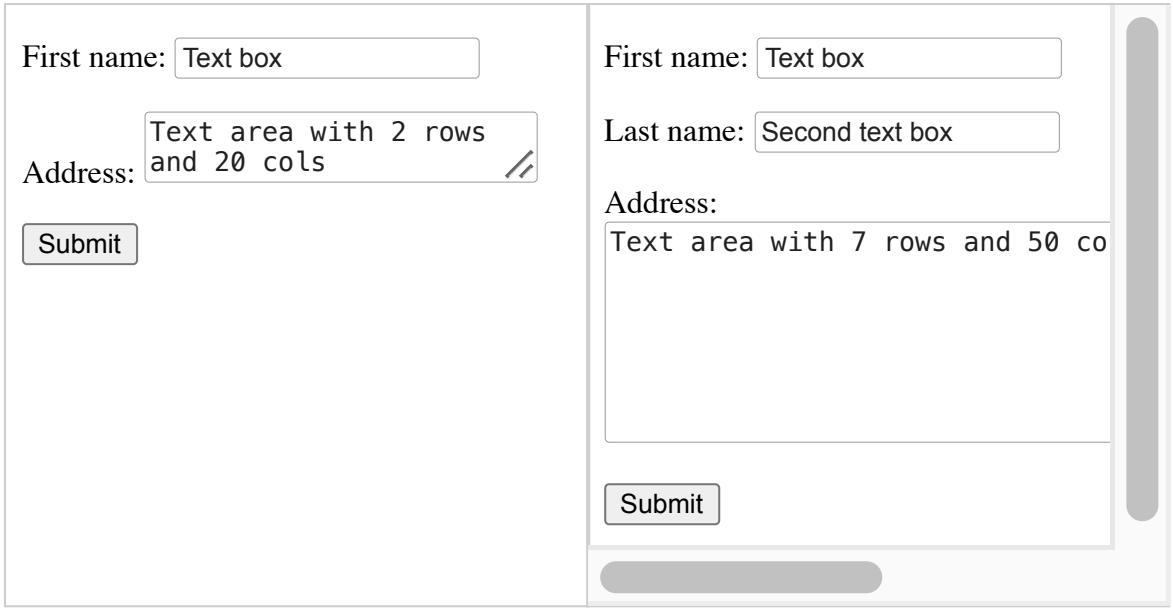
1. Add another text box with a "last" `name`. Also change the size of the text area so that the text area has 7 rows and 50 columns of text. Make your webpage match the expected webpage.
2. Type some data into the form and press Submit. Note that the form data appears in the query string of the resulting webpage because the form uses the GET method.
3. Change the form's `method` from "GET" to "POST". Render the webpage, type some data into the form, and press Submit. The form data no longer appears in the query string of the resulting webpage although the submitted data is present.

```
<form action="https://wp.zybooks.com/form-viewer.php" target="_b
   <p>
      <label for="first">First name:</label>
      <input type="text" id="first" name="first" placeholder="Te
   </p>
   <p>
      <label for="address">Address:</label>
      <textarea id="address" name="address" placeholder="Text ar
   </p>
   <p>
      <input type="submit" value="Submit">
   </p>
</form>

```

Render webpage Reset code

Your webpage **Expected webpage**



▼ View solution

Explain

```
--- START FILE: HTML ---

<form action="https://wp.zybooks.com/form-viewer.php"
target="_blank" method="POST">
   <p>
      <label for="first">First name:</label>
      <input type="text" id="first" name="first"
placeholder="Text box">
   </p>
   <p>
      <label for="last">Last name:</label>
      <input type="text" id="last" name="last"
placeholder="Second text box">
   </p>
   <p>
      <label for="address">Address:</label>
      <textarea id="address" name="address"
placeholder="Text area with 7 rows and 50 cols" rows="7"
cols="50"></textarea>
   </p>
   <p>
      <input type="submit" value="Submit">
   </p>
</form>
```

```
--- END FILE: HTML ---
```

Feedback?

PARTICIPATION ACTIVITY 3.2.9: Text inputs.

1) Which tag creates a text box widget?
   - ◉ `<input>`
   - ○ `<text>`
   - ○ `<textarea>`

Correct

The `<input>` tag creates a text box widget.

2) Which attribute must be set to create a text box widget?
   - ○ `type`
   - ◉ no attribute
   - ○ `name`
   - ○ `id`

Correct

The default value for the `type` attribute is "text", so no attribute must be set to create a text box widget.

3) Which tag creates a widget capable of holding multiple lines of text?
   - ○ `<input>`
   - ○ `<text>`
   - ◉ `<textarea>`

Correct

The text area widget can hold any number of text lines. A vertical scrollbar appears when the number of text lines exceeds the text area size.

4) Which attribute inside the `<label>` tag is used to associate the label with a widget?
   - ○ `id`
   - ◉ `for`
   - ○ `name`

Correct

The `for` attribute contains the `id` attribute of the input element to be labeled.

Feedback?

CHALLENGE ACTIVITY 3.2.1: Building forms.

530096.4000608.qx3zqy7

Jump to level 1

Add a <label> with content "First name", associated to a text <input> with name and id of firstName, and placeholder of Ann. **SHOW EXPECTED**

```
<form action="https://wp.zybooks.com/form-viewer.php" target="_b
   <p>

      <!-- Your solution goes here -->
      <label for = "firstName">First name</label>
      <input type = "text" id = "firstName" name = "firstName" p

   </p>
   <p>
      <input type="submit" value="Submit">
   </p>
</form>
```

1 2 3

Check Next

✔

✔ Testing number of <label> tags

Yours `1`

✔ Testing contents of the <label>

Yours `First name`

✔ Testing the "for" attribute of <label>

Yours `firstName`

✔ Testing number of <input> tags

Yours `2`

✔ Testing type attribute of <input>

Yours `text`

✔ Testing name attribute of <input>

Yours `firstName`

✔ Testing placeholder attribute of <input>

Yours `Ann`

✔ Testing id attribute of <input>

Yours `firstName`

Your webpage

First name Ann

Submit

View your last submission

```
    <!-- Your solution goes here -->
<label for="fullName">Full name</label>
<input type = "text" id="fullName" name="fullName" placeholder="Jan Smi
```

Feedback?

Exploring further:

- enctype attribute from W3Schools
- Sending form data from Mozilla Developer Network

How was this section? Provide section feedback