HTTP POST

HTML Forms

- Write HTML to setup a form
- When a user submits the form it will send a request to your server
  - Respond with an HTML page
  - Submitting a form will reload the page

```
<form action="/form-path" method="get">
    <label for="form-name">Enter your name: </label><br/>
    <input id="form-name" type="text" name="commenter"><br/>

    <label for="form-comment">Comment: </label><br/>
    <input id="form-comment" type="text" name="comment"><br/>

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

HTML Forms

- The action attribute is the path for the form
- The method attribute is the type of HTTP request made
- When the form is submitted, and HTTP request is set to the path using this method
    - This behaves like clicking a link

```
<form action="/form-path" method="get">
    <label for="form-name">Enter your name: </label><br/>
    <input id="form-name" type="text" name="commenter"><br/>

    <label for="form-comment">Comment: </label><br/>
    <input id="form-comment" type="text" name="comment"><br/>

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

HTML Forms

- Use input elements for the user to interact with the form
- The type attribute specifies the type of input
  - This input is a text box
  - The name attribute is used when the data is sent to the server

```
<form action="/form-path" method="get">
    <label for="form-name">Enter your name: </label><br/>
    <input id="form-name" type="text" name="commenter"><br/>

    <label for="form-comment">Comment: </label><br/>
    <input id="form-comment" type="text" name="comment"><br/>

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

HTML Forms

- Its good practice to provide a label for each input
  - Helps with accessibility (ie. Screen readers)
  - Clicking the label focuses the input
  - Use ids to associate labels with inputs

```
<form action="/form-path" method="get">
    <label for="form-name">Enter your name: </label><br/>
    <input id="form-name" type="text" name="commenter"><br/>

    <label for="form-comment">Comment: </label><br/>
    <input id="form-comment" type="text" name="comment"><br/>

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

HTML Forms

- An input of type submit makes a button that will send the HTTP request when clicked
- The value attribute is the text on the button

```
<form action="/form-path" method="get">
    <label for="form-name">Enter your name: </label><br/>
    <input id="form-name" type="text" name="commenter"><br/>

    <label for="form-comment">Comment: </label><br/>
    <input id="form-comment" type="text" name="comment"><br/>

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

HTML Forms

- This sends a GET request containing the form data in a query string
  - Page reloads with the content of the response

```
GET /form-path?commenter=Jesse&comment=Good+morning%21 HTTP/1.1
```

```
<form action="/form-path" method="get">
    <label for="form-name">Enter your name: </label><br/>
    <input id="form-name" type="text" name="commenter"><br/>

    <label for="form-comment">Comment: </label><br/>
    <input id="form-comment" type="text" name="comment"><br/>

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

HTTP POST

- Sending form data in a query string can cause issues
  - Browsers have limits on the length of a URL
    - Browsers have limits on the the total length of a GET request, including headers
      - Typically a few kB
      - How to upload a file?
- Let's try POST requests

HTTP POST

- A POST request is used when the user is sending information to the server
  - As opposed to requesting (GETing) information
- A POST request will include a body
  - Read the Content-Length and Content-Type headers to know how to read the body
  - Follows same protocol as our responses

HTTP POST

- Process a POST request:
  - Read the Content-Length header
  - Find the blank line indicating the end of the headers
  - Read the length of the content number of bytes after the blank line
  - Parse the body according to the Content-Type
- This is what browsers are doing to read your responses

HTML Forms - POST

- Change the method of a form to post to send the entered data in the body of a POST request

```
<form action="/form-path" method="post">
    <label for="form-name">Enter your name: </label><br/>
    <input id="form-name" type="text" name="commenter"><br/>

    <label for="form-comment">Comment: </label><br/>
    <input id="form-comment" type="text" name="comment"><br/>

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

HTML Forms - POST

- A request is sent the path from the action attribute without a query string
- Content-Type is a url encoded string containing the entered data
    - Same format as the query string
- Read the Content-Length to know know how many bytes are in the body
    - Foreshadow: Very import when receiving more data than the size of your TCP buffer

```
POST /form-path HTTP/1.1
Content-Length: 27
Content-Type: application/x-www-form-urlencoded

commenter=Jesse&comment=Good+morning%21
```

HTML Forms - POST

- Specify multipart encoding to receive each input separately in the body

```
<form action="/form-path" method="post" enctype="multipart/form-data">
    <label for="form-name">Enter your name: </label><br/>
    <input id="form-name" type="text" name="commenter"><br/>

    <label for="form-comment">Comment: </label><br/>
    <input id="form-comment" type="text" name="comment"><br/>

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

HTML Forms - POST

- Content-Type specifies a string that separates each input
- Each input has its own headers
- Great for submitting different types of data in the same form
    - Required for file uploads

```
POST /form-path HTTP/1.1
Content-Length: 252
Content-Type: multipart/form-data; boundary=----WebKitFormBoundaryfkz9sCA6fR3CAHN4

------WebKitFormBoundaryfkz9sCA6fR3CAHN4
Content-Disposition: form-data; name="commenter"

Jesse
------WebKitFormBoundaryfkz9sCA6fR3CAHN4
Content-Disposition: form-data; name="comment"

Good morning!
------WebKitFormBoundaryfkz9sCA6fR3CAHN4--
```

HTML Inputs

- Radio Buttons:
  - Provide multiple options with the same name
  - Only one option with the same name can be chosen
  - The value property is sent to the server with this name

```
<form action="/form-path" method="post" enctype="multipart/form-data">

    <input id="option1" type="radio" name="chooseOne" value="1">
    <label for="option1"> 1</label><br/>
    <input id="option2" type="radio" name="chooseOne" value="2">
    <label for="option2"> 2</label><br/>
    <input id="option3" type="radio" name="chooseOne" value="3">
    <label for="option3"> 3</label><br/>

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

HTML Inputs

- Dropdown Menus:
  - Use the select element to create a dropdown
  - The name of the select is sent to the server with the value of the selected option

```
<form action="/form-path" method="post" enctype="multipart/form-data">

    <label for="dropdown">Select an option: </label><br/>
    <select id="dropdown" name="dropping">
        <option value="first">First</option>
        <option value="second">Second</option>
        <option value="third">Third</option>
        <option value="home">Home</option>
    </select>

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

HTML Inputs

- As always
  - There are many more input types
  - Search the documentation for more that you can add to your sites

Demos