Client-Side Web Programming: JavaScript (Part 2)

Robert M. Dondero, Ph.D.

Princeton University

Objectives

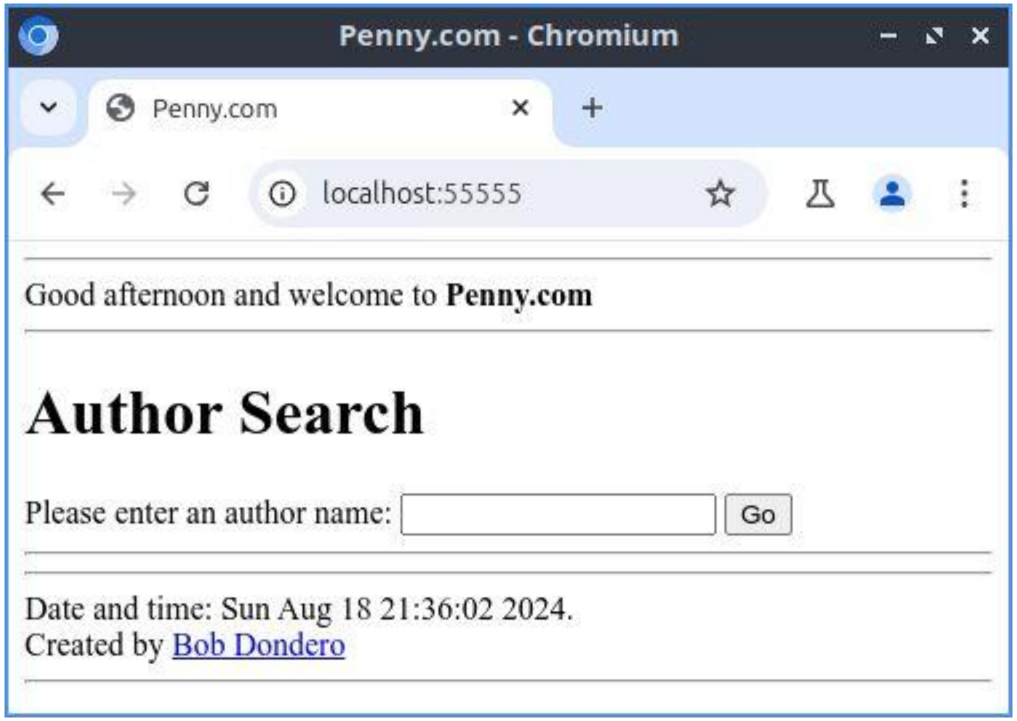
- We will cover:
  - Baseline example
  - JavaScript client-side web programming
  - AJAX

Agenda

- **Baseline example**
- JavaScript client-side web pgmming
- AJAX
- AJAX via XMLHttpRequest
- AJAX via XMLHttpRequest enhancements
- AJAX wrap-up

Baseline Example

- See **PennyOnePage** app



Good afternoon and welcome to **Penny.com**

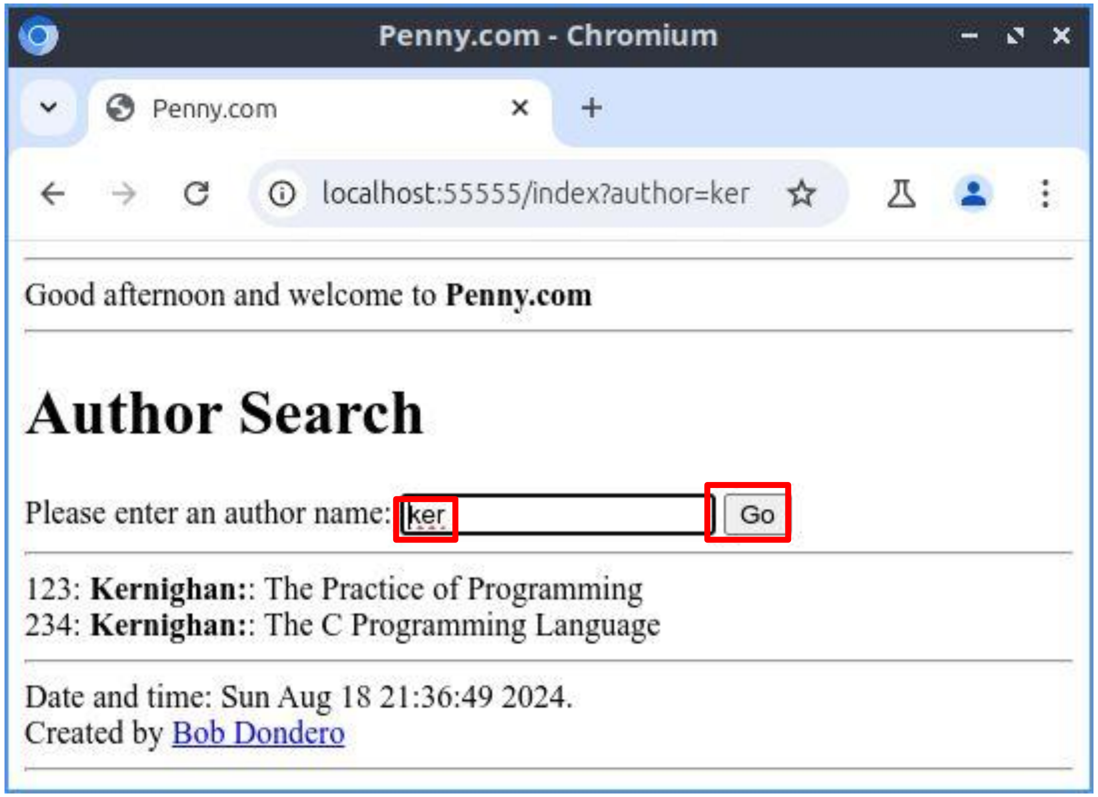
Author Search

Please enter an author name:

Date and time: Sun Aug 18 21:36:02 2024.
Created by Bob Dondero

Baseline Example

- See **PennyOnePage** app (cont.)



Good afternoon and welcome to **Penny.com**

Author Search

Please enter an author name: ker

123: **Kernighan:**: The Practice of Programming
234: **Kernighan:**: The C Programming Language

Date and time: Sun Aug 18 21:36:49 2024.
Created by Bob Dondero

Baseline Example

- See **PennyOnePage** app (cont.)
  - runserver.py
  - penny.sql, penny.sqlite
  - database.py
  - **penny.py**
  - **index.html**

PennyOnePage/penny.py (Page 1 of 1)

```python
#!/usr/bin/env python

#-----------------------------------------------------------------------
# penny.py
# Author: Bob Dondero
#-----------------------------------------------------------------------

import time
import flask
import database

#-----------------------------------------------------------------------

app = flask.Flask(__name__, template_folder='.')

#-----------------------------------------------------------------------

def get_ampm():
    if time.strftime('%p') == "AM":
        return 'morning'
    return 'afternoon'

def get_current_time():
    return time.asctime(time.localtime())

#-----------------------------------------------------------------------

@app.route('/', methods=['GET'])
@app.route('/index', methods=['GET'])
def index():

    author = flask.request.args.get('author')
    if author is None:
        author = ''
    author = author.strip()

    if author == '':
        books = []
    else:
        books = database.get_books(author) # Exception handling omitted

    html = flask.render_template('index.html',
        ampm=get_ampm(),
        current_time=get_current_time(),
        author=author,
        books=books)
    response = flask.make_response(html)
    return response
```

PennyOnePage/index.html (Page 1 of 1)

```html
<!DOCTYPE html>
<html>

   <head>
      <title>Penny.com</title>
   </head>

   <body>
      <hr>
      Good {{ampm}} and welcome to <strong>Penny.com</strong>
      <hr>

      <h1>Author Search</h1>
      <form action="/index" method="get">
         Please enter an author name:
         <input type="text" name="author" value="{{author}}" autoFocus>
         <input type="submit" value="Go">
      </form>
      <hr>
      {% for book in books: %}
         {{book['isbn']}}:
         <strong>{{book['author']}}:</strong>:
         {{book['title']}}<br>
      {% endfor %}

      <hr>
      Date and time: {{current_time}}.<br>
      Created by <a href="https://www.cs.princeton.edu/~rdondero">
      Bob Dondero</a>
      <hr>
</body>

</html>
```

Baseline Example

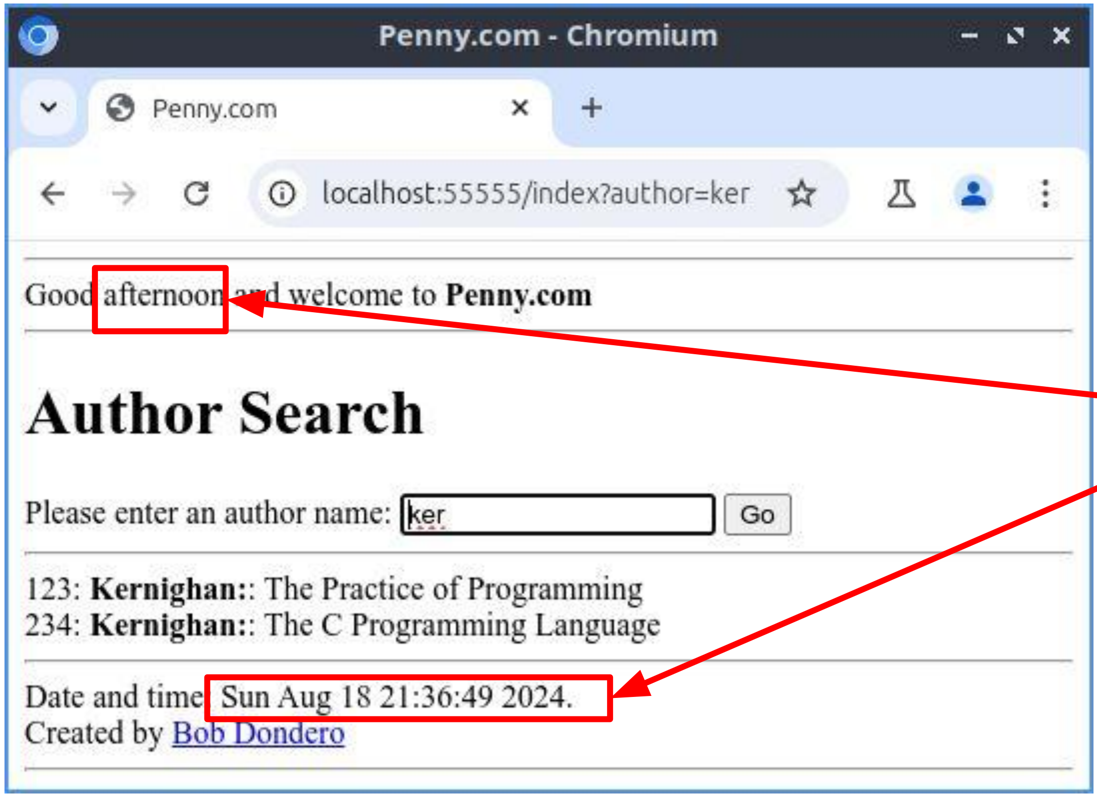
- PennyOnePage vs. PennyFlaskJinja:
  - (con) Doesn’t illustrate multiple Flask routes (endpoints)
  - (con) Doesn’t illustrate state handling
  - (pro) Users prefer?
  - (pro) Better example for this lecture!

Agenda

- Baseline example
- **JavaScript client-side web pgmming**
- AJAX
- AJAX via XMLHttpRequest
- AJAX via XMLHttpRequest enhancements
- AJAX wrap-up

JS Client-Side Web Pgmming

- **Problem**



Good afternoon and welcome to **Penny.com**

Author Search

Please enter an author name: ker Go

123: **Kernighan:**: The Practice of Programming
234: **Kernighan:**: The C Programming Language

Date and time: Sun Aug 18 21:36:49 2024.
Created by Bob Dondero

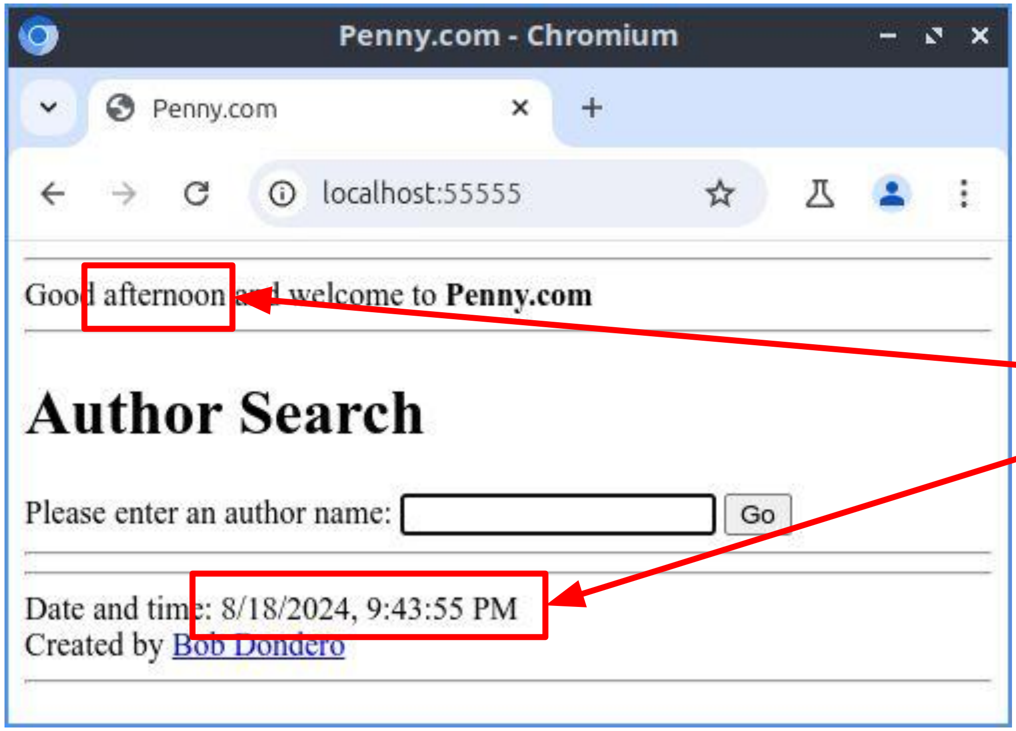
Computed once by server!

JS Client-Side Web Pgmming

- **Solution**
  - Client-side web programming
  - That is, program the browser…

JS Client-Side Web Pgmming

- See **PennyJavaScript** app



Good afternoon and welcome to **Penny.com**

Author Search

Please enter an author name:

Date and time: 8/18/2024, 9:43:55 PM
Created by Bob Dondero

Computed repeatedly by client

JS Client-Side Web Pgmming

- See **PennyJavaScript** app (cont.)
  - runserver.py
  - penny.sql, penny.sqlite
  - database.py
  - **penny.py**
  - **index.html**

PennyJavaScript/penny.py (Page 1 of 1)

```python
#!/usr/bin/env python

#-----------------------------------------------------------------------
# penny.py
# Author: Bob Dondero
#-----------------------------------------------------------------------

import flask
import database

#-----------------------------------------------------------------------

app = flask.Flask(__name__, template_folder='.')

#-----------------------------------------------------------------------

@app.route('/', methods=['GET'])
@app.route('/index', methods=['GET'])
def index():

    author = flask.request.args.get('author')
    if author is None:
        author = ''
    author = author.strip()

    if author == '':
        books = []
    else:
        books = database.get_books(author) # Exception handling omitted

    html_code = flask.render_template('index.html',
        author=author,
        books=books)
    response = flask.make_response(html_code)
    return response
```

PennyJavaScript/index.html (Page 1 of 1)

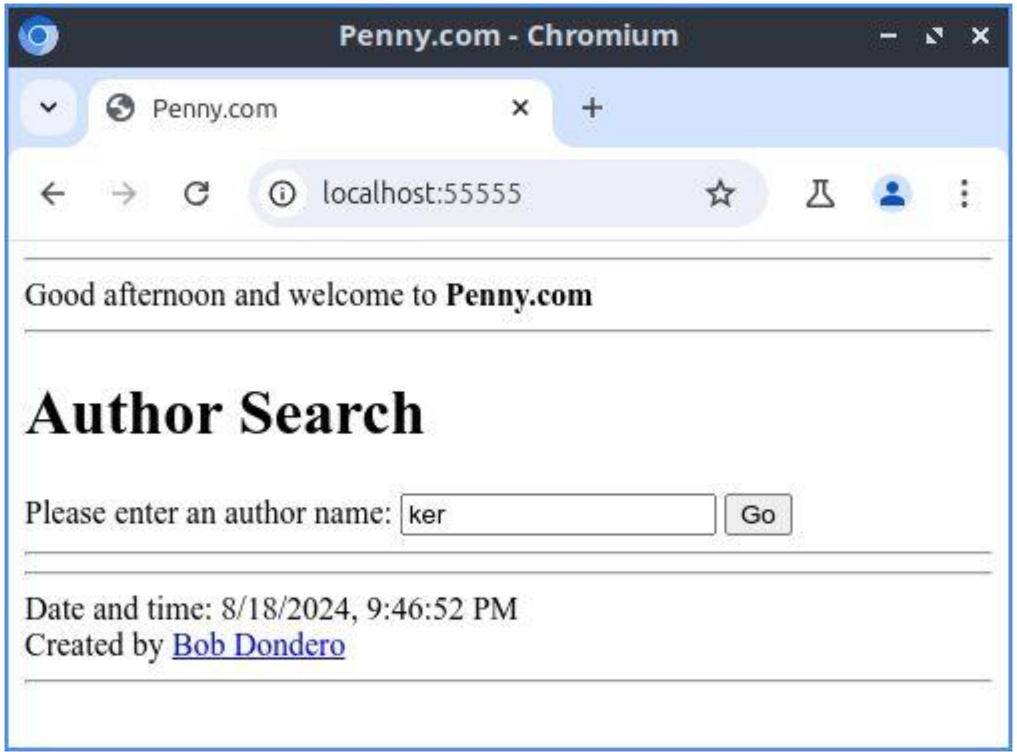
```html
<!DOCTYPE html>
<html>
   <head>
      <title>Penny.com</title>
   </head>

   <body>

      <hr>
      Good <span id="ampmSpan"></span> and welcome to
      <strong>Penny.com</strong>
      <hr>

      <h1>Author Search</h1>
      <form action="/index" method="get">
         Please enter an author name:
         <input type="text" name="author" value="{{author}}" autoFocus>
         <input type="submit" value="Go">
      </form>
      <hr>
      {% for book in books: %}
         {{book['isbn']}}:
         <strong>
         {{book['author']}}:
         </strong>
         {{book['title']}}<br>
      {% endfor %}

      <hr>
      Date and time: <span id="datetimeSpan"></span><br>
      Created by <a href="https://www.cs.princeton.edu/~rdondero">
      Bob Dondero</a>
      <hr>

      <script>

         'use strict';

         function getAmPm() {
            let dateTime = new Date();
            let hours = dateTime.getHours();
            let amPm = (hours < 12) ? 'morning': 'afternoon';
            let ampmSpan = document.getElementById('ampmSpan');
            ampmSpan.innerHTML = amPm;
         }

         function getDateTime() {
            let dateTime = new Date();
            let datetimeSpan = document.getElementById('datetimeSpan');
            datetimeSpan.innerHTML = dateTime.toLocaleString();
         }

         function setup() {
            getAmPm();
            window.setInterval(getAmPm, 1000);
            getDateTime();
            window.setInterval(getDateTime, 1000);
         }

         document.addEventListener('DOMContentLoaded', setup);

      </script>
   </body>
</html>
```

Agenda

- Baseline example
- JavaScript client-side web pgmming
- **AJAX**
- AJAX via XMLHttpRequest
- AJAX via XMLHttpRequest enhancements
- AJAX wrap-up

AJAX

- **Problem**:
  - Page state sometimes is inconsistent
    - Example: User types “ker”, but doesn’t yet click Go



Good afternoon and welcome to **Penny.com**

Author Search

Please enter an author name: ker Go

Date and time: 8/18/2024, 9:46:52 PM
Created by Bob Dondero

AJAX

- **Solution**:
  - Revert to multi-page app, or
  - Stick with one-page app, and update the page with each keystroke…

AJAX

- **Problem**:
  - Inefficient to fetch an **entire** new page with each keystroke
- **Solution**:
  - Update **part of** the current page – the output element – with each keystroke

AJAX

- **Problem:**
  - Shouldn’t update part of page **synchronously**; GUI would be “laggy”
- **Solution**:
  - Should update part of page **asynchronously**, while GUI remains responsive

- But how???

AJAX

- ***AJAX: Asynchronous JavaScript and XML***
  - **JavaScript**
    - AJAX is accomplished via function calls embedded in JavaScript code
  - **Asynchronous**
    - With AJAX, the browser communicates with the server asynchronously, and so remains responsive
  - **XML**
    - With AJAX, the response sent by the server is often (but not necessarily) a XML document

Agenda

- Baseline example
- JavaScript client-side web pgmming
- AJAX
- **AJAX via XMLHttpRequest**
- AJAX via XMLHttpRequest enhancements
- AJAX wrap-up

Aside: JSON in JavaScript

To convert a JSON doc to a JavaScript data structure:

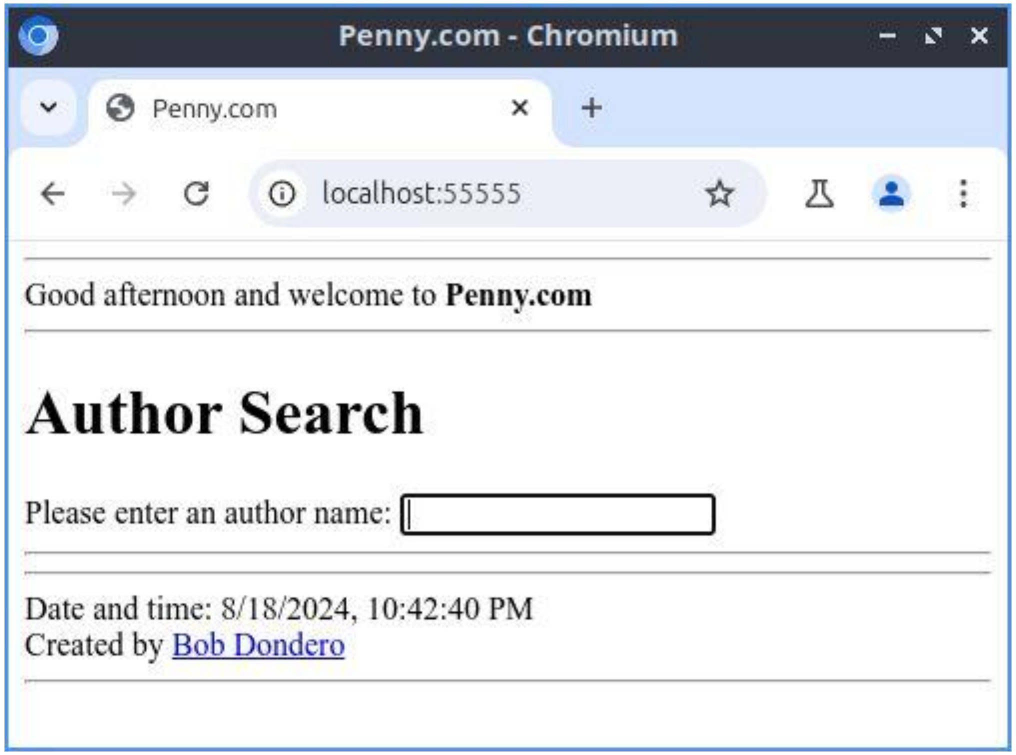
```
ds = JSON.parse(json_doc);
```

To convert a JavaScript data structure to a JSON doc:

```
json_doc = JSON.stringify(ds);
```

AJAX via XMLHttpRequest

- See **<u>PennyAjax1</u>** app



Good afternoon and welcome to **Penny.com**

Author Search

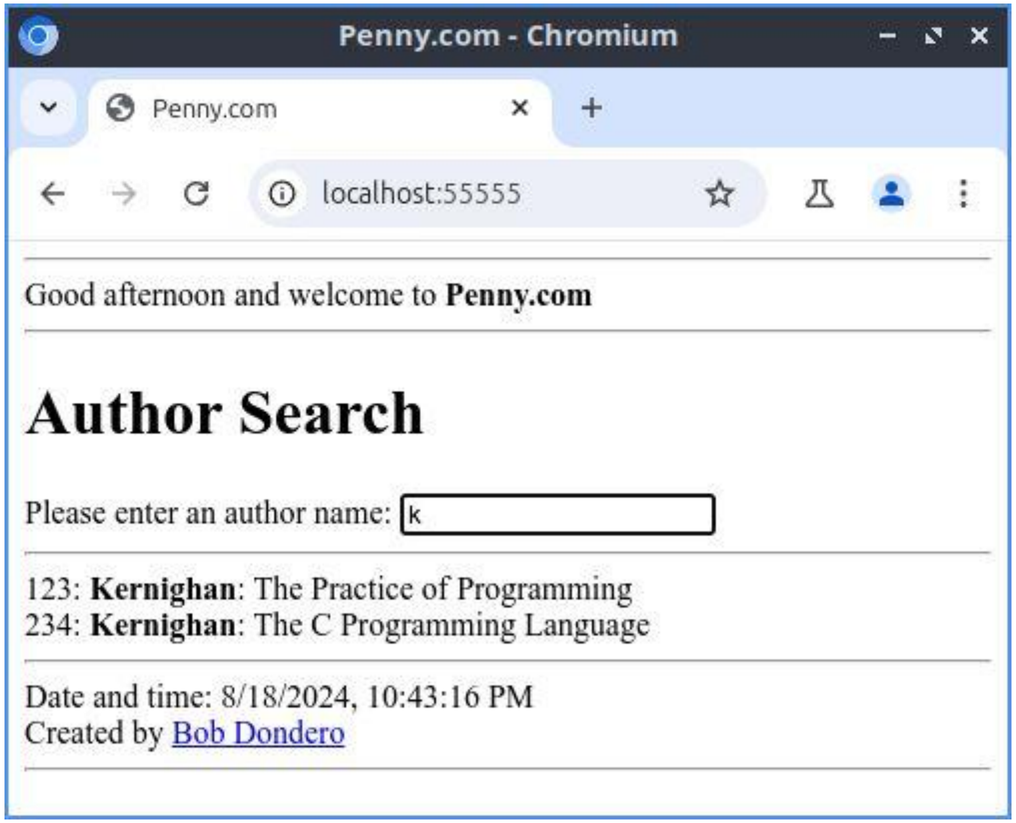
Please enter an author name:

Date and time: 8/18/2024, 10:42:40 PM
Created by Bob Dondero

No
“Go”
button

AJAX via XMLHttpRequest

- See **PennyAjax1** app (cont.)



Good afternoon and welcome to **Penny.com**

Author Search

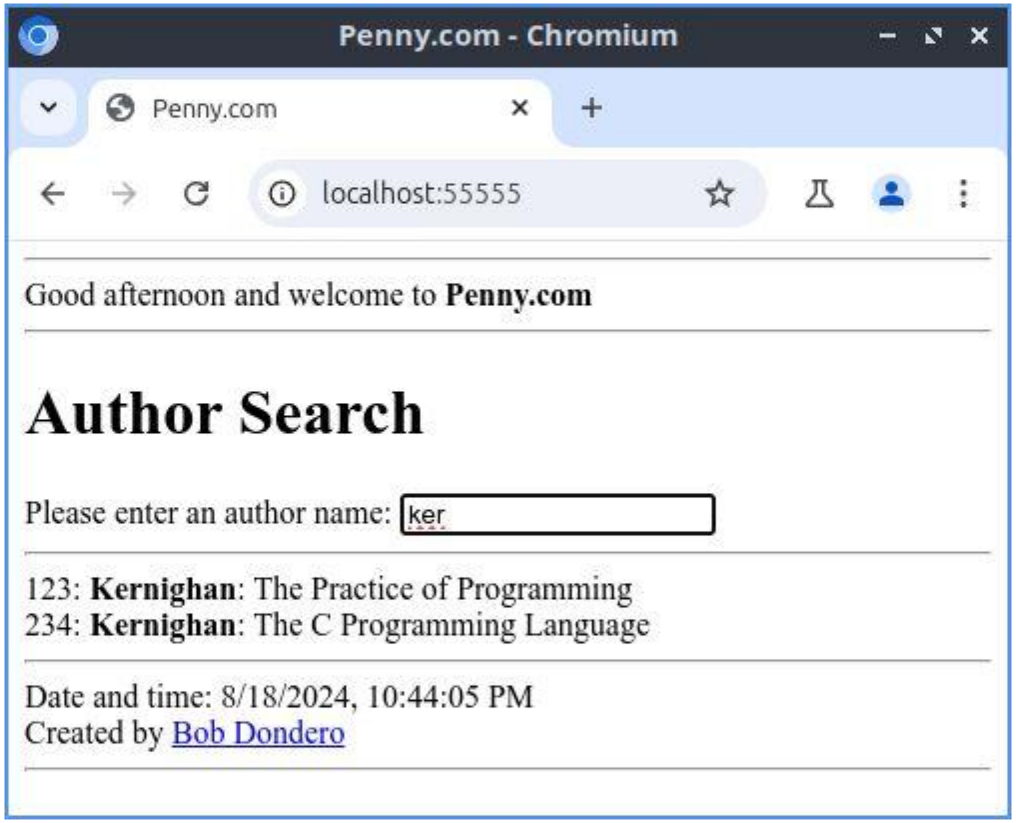
Please enter an author name: k

123: **Kernighan**: The Practice of Programming
234: **Kernighan**: The C Programming Language

Date and time: 8/18/2024, 10:43:16 PM
Created by Bob Dondero

AJAX via XMLHttpRequest

- See **PennyAjax1** app (cont.)



Good afternoon and welcome to **Penny.com**

Author Search

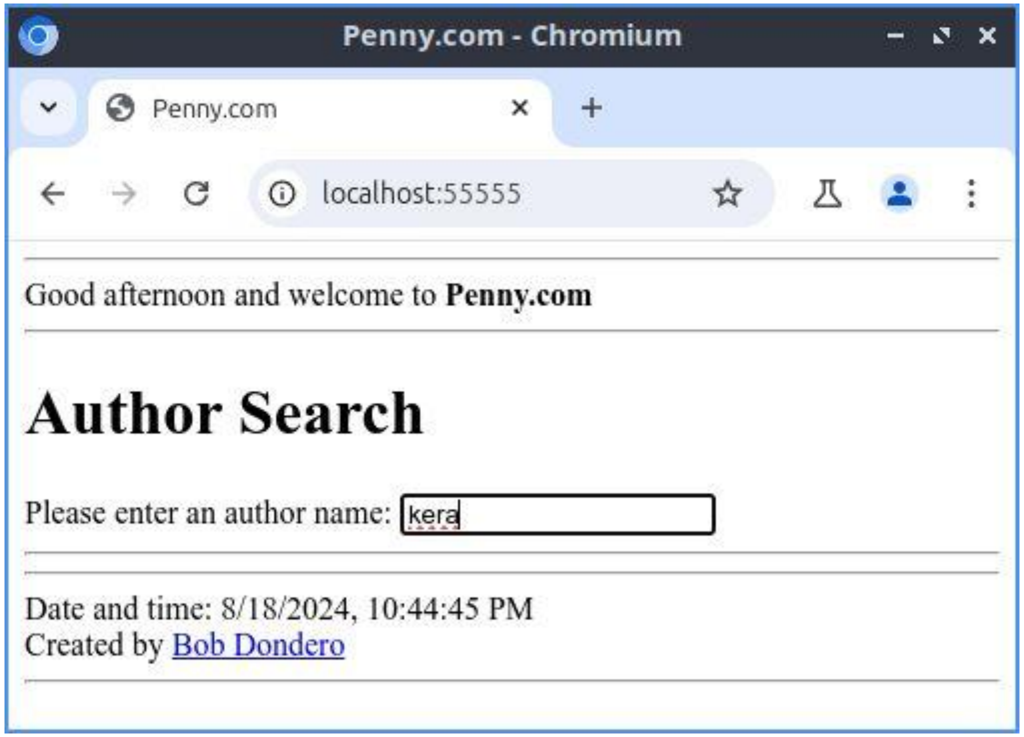
Please enter an author name: ker

123: **Kernighan**: The Practice of Programming
234: **Kernighan**: The C Programming Language

Date and time: 8/18/2024, 10:44:05 PM
Created by Bob Dondero

AJAX via XMLHttpRequest

- See **PennyAjax1** app (cont.)



Good afternoon and welcome to **Penny.com**

Author Search

Please enter an author name: kera

Date and time: 8/18/2024, 10:44:45 PM
Created by Bob Dondero

AJAX via XMLHttpRequest

- See <u>**PennyAjax1**</u> app (cont.)
  - runserver.py
  - penny.sql, penny.sqlite
  - database.py
  - **penny.py**
  - **index.html**

PennyAjax1/penny.py (Page 1 of 1)

```python
#!/usr/bin/env python

#-----------------------------------------------------------------------
# penny.py
# Author: Bob Dondero
#-----------------------------------------------------------------------

import json
import flask
import database

#-----------------------------------------------------------------------

app = flask.Flask(__name__)

#-----------------------------------------------------------------------

@app.route('/', methods=['GET'])
@app.route('/index', methods=['GET'])
def index():

    return flask.send_file('index.html')

#-----------------------------------------------------------------------

@app.route('/searchresults', methods=['GET'])
def search_results():

    author = flask.request.args.get('author')
    if author is None:
        author = ''
    author = author.strip()

    if author == '':
        books = []
    else:
        books = database.get_books(author) # Exception handling omitted

    json_doc = json.dumps(books)
    response = flask.make_response(json_doc)
    response.headers['Content-Type'] = 'application/json'
    return response
```

blank (Page 1 of 1)

```
This page is intentionally blank.
```

PennyAjax1/index.html (Page 1 of 2)

```
<!DOCTYPE html>
<html>
   <head>
      <title>Penny.com</title>
   </head>

   <body>

      <hr>
      Good <span id="ampmSpan"></span> and welcome to
      <strong>Penny.com</strong>
      <hr>

      <h1>Author Search</h1>
      Please enter an author name:
      <input type="text" id="authorInput" autoFocus>
      <hr>
      <div id="resultsDiv"></div>

      <hr>
      Date and time: <span id="datetimeSpan"></span><br>
      Created by <a href="https://www.cs.princeton.edu/~rdondero">
      Bob Dondero</a>
      <hr>

      <script>

         'use strict';

         function getAmPm() {
            let dateTime = new Date();
            let hours = dateTime.getHours();
            let amPm = (hours < 12) ? 'morning' : 'afternoon';
            let ampmSpan = document.getElementById('ampmSpan');
            ampmSpan.innerHTML = amPm;
         }

         function getDateTime() {
            let dateTime = new Date();
            let datetimeSpan =
               document.getElementById('datetimeSpan');
            datetimeSpan.innerHTML = dateTime.toLocaleString();
         }

         function escape(s) {
            s = s.replace('&', '&amp;');
            s = s.replace('<', '&lt;');
            s = s.replace('>', '&gt;');
            s = s.replace('"', '&quot;');
            s = s.replace("'", '&apos;');
            return s;
         }

         function convertToHtml(books) {
            let html = '';
            for (let book of books) {
               html += escape(book.isbn) + ': ';
               html += '<strong>';
               html += escape(book.author);
               html += '</strong>: ';
               html += escape(book.title) + '<br>';
            }
            return html;
         }

```

PennyAjax1/index.html (Page 2 of 2)

```
         function handleResponse() {
            if (this.status !== 200) {
               alert('Error: Failed to fetch data from server');
               return;
            }
            let books = JSON.parse(this.response);
            let html = convertToHtml(books);
            let resultsDiv = document.getElementById('resultsDiv');
            resultsDiv.innerHTML = html;
         }

         function handleError() {
            alert('Error: Failed to fetch data from server');
         }

         function getResults() {
            let authorInput = document.getElementById('authorInput');
            let author = authorInput.value;
            let encodedAuthor = encodeURIComponent(author);
            let url = '/searchresults?author=' + encodedAuthor;
            let request = new XMLHttpRequest();
            request.onload = handleResponse;
            request.onerror = handleError;
            request.open('GET', url);
            request.send();
         }

         function setup() {
            getAmPm();
            window.setInterval(getAmPm, 1000);
            getDateTime();
            window.setInterval(getDateTime, 1000);
            let authorInput = document.getElementById('authorInput');
            authorInput.addEventListener('input', getResults);
         }

         document.addEventListener('DOMContentLoaded', setup);

      </script>
   </body>
</html>
```

AJAX via XMLHttpRequest

- See <u>**PennyAjax1**</u> app (cont.)
  - Note:
    - Could design `search_results()` to return a **HTML fragment** instead of a JSON doc
    - That would be more convenient if the client is a browser
    - That would be less convenient if the client is:
      - A desktop app
      - An Android app
      - An iOS app

Agenda

- Baseline example
- JavaScript client-side web pgmming
- AJAX
- AJAX via XMLHttpRequest
- **AJAX via XMLHttpRequest enhancements**
- AJAX wrap-up

AJAX Enhancements

- **Problem**:
  - Code to convert JavaScript data structure to HTML doc is ugly, inefficient
- **Solution:**
  - Use a **template engine**

AJAX Enhancements

- Python
  - Mustache, CheetahTemplate, Django, Genshi, **Jinja2**, Kid, Topsite, …
- JavaScript
  - **Mustache**, Squirrelly, Handlebars, …
- Java
  - Mustache, FreeMarker, Hamlets, Tiles, Thymeleaf, WebMacro, WebObjects, Velocity, …

https://en.wikipedia.org/wiki/Web_template_system

AJAX Enhancements

- See **PennyAjax2** app
  - runserver.py
  - penny.sql, penny.sqlite
  - database.py
  - penny.py
  - **index.html**

PennyAjax2/index.html (Page 1 of 2)

```
<!DOCTYPE html>
<html>
   <head>
      <title>Penny.com</title>
   </head>

   <body>

      <hr>
      Good <span id="ampmSpan"></span> and welcome to
      <strong>Penny.com</strong>
      <hr>

      <h1>Author Search</h1>
      Please enter an author name:
      <input type="text" id="authorInput" autoFocus>
      <hr>
      <div id="resultsDiv"></div>

      <hr>
      Date and time: <span id="datetimeSpan"></span><br>
      Created by <a href="https://www.cs.princeton.edu/~rdondero">
      Bob Dondero</a>
      <hr>

      <script src=
         "https://cdn.jsdelivr.net/npm/mustache@4.2.0/mustache.min.js">
      </script>

      <!-- <script src="/static/mustache.min.js"></script> -->

      <script>

         'use strict';

         function getAmPm() {
            let dateTime = new Date();
            let hours = dateTime.getHours();
            let amPm = (hours < 12) ? 'morning' : 'afternoon';
            let ampmSpan = document.getElementById('ampmSpan');
            ampmSpan.innerHTML = amPm;
         }

         function getDateTime() {
            let dateTime = new Date();
            let datetimeSpan =
               document.getElementById('datetimeSpan');
            datetimeSpan.innerHTML = dateTime.toLocaleString();
         }

         function convertToHtml(books) {
            let template = `
               {{#books}}
                  {{isbn}}:
                  <strong>{{author}}</strong>:
                  {{title}}
                  <br>
               {{/books}}
            `;
            let map = {books: books};
            let html = Mustache.render(template, map);
            return html;
         }

         function handleResponse() {
```

PennyAjax2/index.html (Page 2 of 2)

```
            if (this.status !== 200) {
               alert('Error: Failed to fetch data from server');
               return;
            }
            let books = JSON.parse(this.response);
            let html = convertToHtml(books);
            let resultsDiv = document.getElementById('resultsDiv');
            resultsDiv.innerHTML = html;
         }

         function handleError() {
            alert('Error: Failed to fetch data from server');
         }

         function getResults() {
            let authorInput = document.getElementById('authorInput');
            let author = authorInput.value;
            let encodedAuthor = encodeURIComponent(author);
            let url = '/searchresults?author=' + encodedAuthor;
            let request = new XMLHttpRequest();
            request.onload = handleResponse;
            request.onerror = handleError;
            request.open('GET', url);
            request.send();
         }

         function setup() {
            getAmPm();
            window.setInterval(getAmPm, 1000);
            getDateTime();
            window.setInterval(getDateTime, 1000);
            let authorInput = document.getElementById('authorInput');
            authorInput.addEventListener('input', getResults);
         }

         document.addEventListener('DOMContentLoaded', setup);

      </script>
   </body>
</html>
```

AJAX Enhancements

- **How to fetch the Mustache library…**

- **Option 1**
  - Command browser to fetch Mustache library from the **cdn** website
- **Option 2**
  - Command browser to fetch Mustache library from *your website*

Aside: Mustache

- Template (informally)
  - HTML string with placeholders
  - Each placeholder is identified by a Mustache variable

```
Hello <strong>{{username}}</strong>
and welcome
```

Aside: Mustache

- To instantiate a template:

```
let map = {somevar: someval, …};
let html = Mustache.render(sometemplate, map);
```

  - For each placeholder identified by `somevar` in `sometemplate`, replaces the placeholder with `someval`
- Automatically escapes `someval`
- Returns the resulting string

Aside: Mustache

- Template can contain:
  - Variables

```
… {{author}} …
```

Aside: Mustache

- Template can contain:
  - Iteration constructs

```
{{#books}}
   <strong>{{author}}</strong>
   …
{{/books}}
```

Note:

- Unusual implicit specification of iteration object
- If books is falsy, then block is not rendered

Aside: Mustache

- Template can contain:
  - Selection constructs

```
{{#books}}
    …
{{/books}}
{{^books}}
    …
{{/books}}
```

If books is truthy, then first block is rendered
If books is falsy, then the second block is rendered

Aside: Mustache

- Template can contain:
  - Includes of other templates

```
…
{{>header}}
…
…
{{>footer}}
…
```

Aside: Mustache

- There is more to Mustache
- For more information:
  - https://github.com/janl/mustache.js

AJAX Enhancements

- **Problem:**
  - Server will respond to requests in arbitrary order
- **Solution:**
  - Abort previous request

AJAX Enhancements

- See **PennyAjax3** app
  - runserver.py
  - penny.sql, penny.sqlite
  - database.py
  - penny.py
  - **index.html**

PennyAjax3/index.html (Page 1 of 2)

```
<!DOCTYPE html>
<html>
   <head>
      <title>Penny.com</title>
   </head>

   <body>

      <hr>
      Good <span id="ampmSpan"></span> and welcome to
      <strong>Penny.com</strong>
      <hr>

      <h1>Author Search</h1>
      Please enter an author name:
      <input type="text" id="authorInput" autoFocus>
      <hr>
      <div id="resultsDiv"></div>

      <hr>
      Date and time: <span id="datetimeSpan"></span><br>
      Created by <a href="https://www.cs.princeton.edu/˜rdondero">
      Bob Dondero</a>
      <hr>

      <script src=
         "https://cdn.jsdelivr.net/npm/mustache@4.2.0/mustache.min.js">
      </script>

      <script>

         'use strict';

         function getAmPm() {
            let dateTime = new Date();
            let hours = dateTime.getHours();
            let amPm = (hours < 12) ? 'morning' : 'afternoon';
            let ampmSpan = document.getElementById('ampmSpan');
            ampmSpan.innerHTML = amPm;
         }

         function getDateTime() {
            let dateTime = new Date();
            let datetimeSpan =
               document.getElementById('datetimeSpan');
            datetimeSpan.innerHTML = dateTime.toLocaleString();
         }

         function convertToHtml(books) {
            let template = `
               {{#books}}
                  {{isbn}}:
                  <strong>{{author}}</strong>:
                  {{title}}
                  <br>
               {{/books}}
            `;
            let map = {books: books};
            let html = Mustache.render(template, map);
            return html;
         }

         function handleResponse() {
            if (this.status !== 200) {
               alert('Error: Failed to fetch data from server');
```

PennyAjax3/index.html (Page 2 of 2)

```
               return;
            }
            let books = JSON.parse(this.response);
            let html = convertToHtml(books);
            let resultsDiv = document.getElementById('resultsDiv');
            resultsDiv.innerHTML = html;
         }

         function handleError() {
            alert('Error: Failed to fetch data from server');
         }

         let request = null;

         function getResults() {
            let authorInput = document.getElementById('authorInput');
            let author = authorInput.value;
            let encodedAuthor = encodeURIComponent(author);
            let url = '/searchresults?author=' + encodedAuthor;
            if (request !== null)
               request.abort();
            request = new XMLHttpRequest();
            request.onload = handleResponse;
            request.onerror = handleError;
            request.open('GET', url);
            request.send();
         }

         function setup() {
            getAmPm();
            window.setInterval(getAmPm, 1000);
            getDateTime();
            window.setInterval(getDateTime, 1000);
            let authorInput = document.getElementById('authorInput');
            authorInput.addEventListener('input', getResults);
         }

         document.addEventListener('DOMContentLoaded', setup);

      </script>
   </body>
</html>
```

AJAX Enhancements

- **Problem**:
  - Server could be overwhelmed with requests
- **Solution:**
  - ***Debounce*** the requests

AJAX Enhancements

- See **PennyAjax4** app
  - runserver.py
  - penny.sql, penny.sqlite
  - database.py
  - penny.py
  - **index.html**

PennyAjax4/index.html (Page 1 of 2)

```
<!DOCTYPE html>
<html>
   <head>
      <title>Penny.com</title>
   </head>

   <body>

      <hr>
      Good <span id="ampmSpan"></span> and welcome to
      <strong>Penny.com</strong>
      <hr>

      <h1>Author Search</h1>
      Please enter an author name:
      <input type="text" id="authorInput" autoFocus>
      <hr>
      <div id="resultsDiv"></div>

      <hr>
      Date and time: <span id="datetimeSpan"></span><br>
      Created by <a href="https://www.cs.princeton.edu/˜rdondero">
      Bob Dondero</a>
      <hr>

      <script src=
         "https://cdn.jsdelivr.net/npm/mustache@4.2.0/mustache.min.js">
      </script>

      <script>

         'use strict';

         function getAmPm() {
            let dateTime = new Date();
            let hours = dateTime.getHours();
            let amPm = (hours < 12) ? 'morning' : 'afternoon';
            let ampmSpan = document.getElementById('ampmSpan');
            ampmSpan.innerHTML = amPm;
         }

         function getDateTime() {
            let dateTime = new Date();
            let datetimeSpan =
               document.getElementById('datetimeSpan');
            datetimeSpan.innerHTML = dateTime.toLocaleString();
         }

         function convertToHtml(books) {
            let template = `
               {{#books}}
                  {{isbn}}:
                  <strong>{{author}}</strong>:
                  {{title}}
                  <br>
               {{/books}}
            `;
            let map = {books: books};
            let html = Mustache.render(template, map);
            return html;
         }

         function handleResponse() {
            if (this.status !== 200) {
               alert('Error: Failed to fetch data from server');
```

PennyAjax4/index.html (Page 2 of 2)

```
               return;
            }
            let books = JSON.parse(this.response);
            let html = convertToHtml(books);
            let resultsDiv = document.getElementById('resultsDiv');
            resultsDiv.innerHTML = html;
         }

         function handleError() {
            alert('Error: Failed to fetch data from server');
         }

         let request = null;

         function getResults() {
            let authorInput = document.getElementById('authorInput');
            let author = authorInput.value;
            let encodedAuthor = encodeURIComponent(author);
            let url = '/searchresults?author=' + encodedAuthor;
            if (request !== null)
               request.abort();
            request = new XMLHttpRequest();
            request.onload = handleResponse;
            request.onerror = handleError;
            request.open('GET', url);
            request.send();
         }

         let timer = null;

         function debouncedGetResults() {
            clearTimeout(timer);
            timer = setTimeout(getResults, 500);
         }

         function setup() {
            getAmPm();
            window.setInterval(getAmPm, 1000);
            getDateTime();
            window.setInterval(getDateTime, 1000);
            let authorInput = document.getElementById('authorInput');
            authorInput.addEventListener('input', debouncedGetResults);
         }

         document.addEventListener('DOMContentLoaded', setup);

      </script>
   </body>
</html>
```

AJAX Enhancements

- **Bonus:**
  - Debouncing reduces (but does not eliminate) the need to abort requests!

Question (13webjavascript2)

- Does debouncing eliminate the need to abort previous AJAX requests? Answer “yes” or “no”.
  - Browse to https://cos333attend.cs.princeton.edu to answer

[2 points]

Agenda

- Baseline example
- JavaScript client-side web pgmming
- AJAX
- AJAX via XMLHttpRequest
- AJAX via XMLHttpRequest enhancements
- **AJAX wrap-up**

AJAX Wrap-Up

AJAX Implementation	Browser Built-In or Library?	COS 333 Coverage?
XMLHttpRequest	Built-in	This lecture
fetch & AbortController	Built-in	Appendix
Axios	Library	None
jQuery	Library	Next lecture

AJAX Wrap-Up

AJAX Implementation	Firefox	Chrome
XMLHttpRequest	12+ (2012)	31+ (2013)
fetch	39+ (2015)	42+ (2015)
AbortController	57+ (2017)	66+ (2018)
Axios	12+ (2012)	31+ (2013)
jQuery	12+ (2012)	31+ (2013)

AJAX Wrap-Up

- PennyAjax app is a ***single page app (SPA)***

- SPAs are enabled by AJAX

Summary

- We have covered:
  - Baseline example
  - JavaScript client-side web programming
  - AJAX
- See also:
  - **Appendix 1**: AJAX via fetch

Appendix 1: AJAX via fetch

AJAX via fetch

- **Option 1:**
  - **`fetch()`** function
    - Uses **promises**

AJAX via fetch

- See **PennyAjaxFetch1** app
  - runserver.py
  - penny.sql, penny.sqlite
  - database.py
  - penny.py
  - **index.html**

PennyAjaxFetch1/index.html (Page 1 of 2)

```
<!DOCTYPE html>
<html>
   <head>
      <title>Penny.com</title>
   </head>

   <body>

      <hr>
      Good <span id="ampmSpan"></span> and welcome to
      <strong>Penny.com</strong>
      <hr>

      <h1>Author Search</h1>
      Please enter an author name:
      <input type="text" id="authorInput" autoFocus>
      <hr>
      <div id="resultsDiv"></div>

      <hr>
      Date and time: <span id="datetimeSpan"></span><br>
      Created by <a href="https://www.cs.princeton.edu/˜rdondero">
      Bob Dondero</a>
      <hr>

      <script src=
         "https://cdn.jsdelivr.net/npm/mustache@4.2.0/mustache.min.js">
      </script>

      <script>

         'use strict';

         function getAmPm() {
            let dateTime = new Date();
            let hours = dateTime.getHours();
            let amPm = (hours < 12) ? 'morning' : 'afternoon';
            let ampmSpan = document.getElementById('ampmSpan');
            ampmSpan.innerHTML = amPm;
         }

         function getDateTime() {
            let dateTime = new Date();
            let datetimeSpan =
               document.getElementById('datetimeSpan');
            datetimeSpan.innerHTML = dateTime.toLocaleString();
         }

         function usingResponseGetText(response) {
            if (! response.ok)
               throw new Error();
            return response.text();
         }

         function convertToHtml(books) {
            let template = `
               {{#books}}
                  {{isbn}}:
                  <strong>{{author}}</strong>:
                  {{title}}
                  <br>
               {{/books}}
            `;
            let map = {books: books};
            let html = Mustache.render(template, map);
```

PennyAjaxFetch1/index.html (Page 2 of 2)

```
            return html;
         }

         function usingTextUpdateResultsDiv(text) {
            let books = JSON.parse(text);
            let html = convertToHtml(books);
            let resultsDiv = document.getElementById('resultsDiv');
            resultsDiv.innerHTML = html;
         }

         function handleError(err) {
            if (err.name !== 'AbortError')
               alert('Error: Failed to fetch data from server');
         }

         let controller = null;

         function getResults() {
            let authorInput = document.getElementById('authorInput');
            let author = authorInput.value;
            let encodedAuthor = encodeURIComponent(author);
            let url = '/searchresults?author=' + encodedAuthor;
            if (controller !== null)
               controller.abort();
            controller = new AbortController();
            fetch(url, {signal: controller.signal})
               .then(usingResponseGetText)
               .then(usingTextUpdateResultsDiv)
               .catch(handleError);
         }

         let timer = null;

         function debouncedGetResults() {
            clearTimeout(timer);
            timer = setTimeout(getResults, 500);
         }

         function setup() {
            getAmPm();
            window.setInterval(getAmPm, 1000);
            getDateTime();
            window.setInterval(getDateTime, 1000);
            let authorInput = document.getElementById('authorInput');
            authorInput.addEventListener('input', debouncedGetResults);
         }

         document.addEventListener('DOMContentLoaded', setup);

      </script>
   </body>
</html>
```

AJAX via fetch

```
fetch(url)
    .then(usingResponseGetText)
    .then(usingTextUpdateResultsDiv)
    .catch(handleError);
```

- Fetch a response from `url`
- After that's finished, call `usingResponseGetText`, passing it the value returned by `fetch`
- After that's finished, call `usingTextUpdateResultsDiv`, passing it the value returned by `usingResponseGetText`
- If an exception occurs, call `handleError`, passing it the Error object

AJAX via fetch

```
if (this._controller !== null)
   this._controller.abort();
this._controller = new AbortController();

fetch(url, {signal: this._controller.signal})
   .then(usingResponseGetText)
   .then(usingTextUpdateResultsDiv)
   .catch(handleError);
```

Use of `AbortController` allows abort of request

AJAX via fetch

- **Option 2:**
  - **`fetch()`** function
    - Uses **promises**
  - **`Async`** and **`await`**

AJAX via fetch

- See **PennyAjaxFetch2** app
  - runserver.py
  - penny.sql, penny.sqlite
  - database.py
  - penny.py
  - **index.html**

PennyAjaxFetch2/index.html (Page 1 of 2)

```
<!DOCTYPE html>
<html>
   <head>
      <title>Penny.com</title>
   </head>

   <body>

      <hr>
      Good <span id="ampmSpan"></span> and welcome to
      <strong>Penny.com</strong>
      <hr>

      <h1>Author Search</h1>
      Please enter an author name:
      <input type="text" id="authorInput" autoFocus>
      <hr>
      <div id="resultsDiv"></div>

      <hr>
      Date and time: <span id="datetimeSpan"></span><br>
      Created by <a href="https://www.cs.princeton.edu/~rdondero">
      Bob Dondero</a>
      <hr>

      <script src=
         "https://cdn.jsdelivr.net/npm/mustache@4.2.0/mustache.min.js">
      </script>

      <script>

         'use strict';

         function getAmPm() {
            let dateTime = new Date();
            let hours = dateTime.getHours();
            let amPm = (hours < 12) ? 'morning' : 'afternoon';
            let ampmSpan = document.getElementById('ampmSpan');
            ampmSpan.innerHTML = amPm;
         }

         function getDateTime() {
            let dateTime = new Date();
            let datetimeSpan =
               document.getElementById('datetimeSpan');
            datetimeSpan.innerHTML = dateTime.toLocaleString();
         }

         function convertToHtml(books) {
            let template = `
               {{#books}}
                  {{isbn}}:
                  <strong>{{author}}</strong>:
                  {{title}}
                  <br>
               {{/books}}
            `;
            let map = {books: books};
            let html = Mustache.render(template, map);
            return html;
         }

         async function fetchBooks(url) {
            try {
               let response =
```

PennyAjaxFetch2/index.html (Page 2 of 2)

```
                  await fetch(url, {signal: controller.signal});
               if (! response.ok)
                  throw new Error();
               let text = await response.text();
               let books = JSON.parse(text);
               let html = convertToHtml(books);
               let resultsDiv = document.getElementById('resultsDiv');
               resultsDiv.innerHTML = html;
            }
            catch (err) {
               if (err.name !== 'AbortError')
                  alert('Error: Failed to fetch data from server');
            }
         }

         let controller = null;

         function getResults() {
            let authorInput = document.getElementById('authorInput');
            let author = authorInput.value;
            let encodedAuthor = encodeURIComponent(author);
            let url = '/searchresults?author=' + encodedAuthor;
            if (controller !== null)
               controller.abort();
            controller = new AbortController();
            fetchBooks(url);
         }

         let timer = null;

         function debouncedGetResults() {
            clearTimeout(timer);
            timer = setTimeout(getResults, 500);
         }

         function setup() {
            getAmPm();
            window.setInterval(getAmPm, 1000);
            getDateTime();
            window.setInterval(getDateTime, 1000);
            let authorInput = document.getElementById('authorInput');
            authorInput.addEventListener('input', debouncedGetResults);
         }

         document.addEventListener('DOMContentLoaded', setup);

      </script>
   </body>
</html>
```